ПРАКТИЧЕСКИЙ УЧЕБНИК

БЕЛЫЙ ХАКИНГ С НУЛЯ

Основы кибербезопасности
и этичного тестирования



От первых команд до первого отчёта

Иван Гром

Белый хакинг с нуля

«Автор»
2026

Гром И.

Белый хакинг с нуля / И. Гром — «Автор», 2026

Как устроены компьютеры и сети? Как исследовать веб-приложение, распознавать уязвимости и понятно объяснять их владельцам? Эта книга поможет пройти путь от первых команд в терминале до учебного проекта по тестированию безопасности.Вы разберётесь в Linux и Windows, сетевых протоколах, HTTP, инструментах Nmap и Burp Suite, контроле доступа, SQL-инъекциях, XSS, паролях, Active Directory и основах обратного инжиниринга. На примерах и пошаговых практикумах вы научитесь работать в собственной лаборатории, проверять гипотезы, разбирать результаты и составлять отчёты с рекомендациями по исправлению.Материал рассчитан на новичков: сложные темы объясняются последовательно, а задания помогают закрепить знания. Все практические проверки выполняются только в собственной лаборатории или на системах, тестирование которых явно разрешено.

© Гром И., 2026
© Автор, 2026

Содержание

Содержание 8
Часть I. Основной текст книги 10
Для абсолютного новичка: начните отсюда 11
Правда, которую полезно знать сразу 12
Что нужно, чтобы начать 13
Мини-словарь самых базовых слов 14
Самое важное правило всей книги 15
Как читать эту книгу, чтобы не бросить 16
Ваши первые три шага прямо сейчас 17
Куда смотреть, если запутались 18
Как читать эту книгу 19
Что такое «белый хакинг» 20
Чем на самом деле занимается белый хакер 21
Чего в этой книге нет 22
Как устроена книга 23
Главный совет перед началом 24
Глава 1. Закон, этика и как не сесть в тюрьму 25
Простая аналогия 26
Что считается нарушением 27
Что даёт вам право тестировать 28
Личные правила, которые спасут вам карьеру 29
Практика к главе 1 30
Частые ошибки новичка 31
Глава 2. Как вообще работает компьютер и сеть 32
Компьютер в четырёх словах 33
Сеть: как данные доходят из точки А в точку Б 34
HTTP — язык веба 36
Шифрование простыми словами 37
Практика к главе 2 38
Частые ошибки новичка 39
Глава 3. Операционные системы: Linux и Windows глазами хакера 40
Почему Linux 40
Структура файловой системы Linux 41
Права доступа — самое важное в Linux 42
Windows глазами специалиста по безопасности 43
Практика к главе 3 44
Частые ошибки новичка 45
Углублённо: что ещё важно знать о сети 46
Глава 4. Терминал Linux: ваш главный инструмент 47
Как устроена команда 48
Навигация 49
Работа с файлами 50
Поиск — то, что вы будете делать постоянно 51
Конвейеры и перенаправление — суперсила 52
Права и процессы 53
Сеть из терминала 54

Установка программ 55
Горячие клавиши, которые экономят часы 56
Практика к главе 4 57
Частые ошибки новичка 58
Глава 5. Инструменты белого хакера 59
Карта инструментов по этапам работы 60
Четыре инструмента, которые надо освоить первыми 61
Kali Linux и аналоги 62
Как не утонуть в инструментах 63
Практика к главе 5 64
Углублённо: текстовая обработка и автоматизация 65
Глава 6. Собираем собственную лабораторию 67
Зачем нужна виртуализация 68
Что устанавливаем 69
Пошаговая установка 70
OWASP Juice Shop через Docker 72
Бесплатные онлайн-площадки (альтернатива и 73
Практика к главе 6 74
Частые ошибки новичка 75
Глава 7. Разведка: искусство собирать информацию 76
Два вида разведки 77
OSINT — разведка по открытым источникам 78
Как это складывается в картину 80
Практика к главе 7 81
Частые ошибки новичка 82
Глава 8. Сканирование портов и сервисов с Nmap 83
Идея простыми словами 84
Этап 1. Кто жив (host discovery) 85
Этап 2. Какие порты открыты 86
Этап 3. Что за сервисы и версии — самое ценное 87
Этап 4. ОС и скрипты 88
Рекомендуемая связка для новичка 89
Чтение результата: пример рассуждения 90
Масштаб и скорость 91
Практика к главе 8 92
Частые ошибки новичка 93
Углублённо: от одной команды к методике 94
Глава 9. Как устроено веб-приложение и где оно ломается 95
Три слоя любого сайта 96
Главный принцип всех веб-уязвимостей 97
Доверенная и недоверенная сторона 98
OWASP Top 10 — карта того, что искать 99
Инструменты для веб-работы 100
Методика: как подходить к незнакомому сайту 101
Практика к главе 9 102
Частые ошибки новичка 103
Глава 10. Burp Suite: прокси между вами и сайтом 104
Зачем это нужно 105
Установка и первый запуск 106

Главные вкладки 107
Рабочий цикл: Proxy → Repeater 108
Пример: разбор запроса в Repeater 109
HTTP history как журнал 110
Decoder — работа с кодировками 111
OWASP ZAP — бесплатная альтернатива 112
Практика к главе 10 113
Частые ошибки новичка 114
Глава 11. Инъекции: SQL-инъекция на понятном примере 115
Откуда берётся SQL-инъекция 116
Как это обнаруживают (на разрешённой цели) 117
Конец ознакомительного фрагмента. 118

Иван Гром
Белый хакинг с нуля

БЕЛЫЙ ХАКИНГ С НУЛЯ

Полный учебный курс для начинающих

От основ компьютера и сетей до лабораторной проверки и отчёта

Исходная книга и расширенная практическая часть

Содержание

Часть I. Основной текст книги
Для абсолютного новичка: начните отсюда
Как читать эту книгу
Глава 1. Закон, этика и как не сесть в тюрьму
Глава 2. Как вообще работает компьютер и сеть
Глава 3. Операционные системы: Linux и Windows глазами хакера
Глава 4. Терминал Linux: ваш главный инструмент
Глава 5. Инструменты белого хакера
Глава 6. Собираем собственную лабораторию
Глава 7. Разведка: искусство собирать информацию
Глава 8. Сканирование портов и сервисов с Nmap
Глава 9. Как устроено веб-приложение и где оно ломается
Глава 10. Burp Suite: прокси между вами и сайтом
Глава 11. Инъекции: SQL-инъекция на понятном примере
Глава 12. XSS: когда чужой скрипт попадает на страницу
Глава 13. Контроль доступа и IDOR — самая «денежная» тема новичка
Глава 14. Остальные веб-уязвимости: краткий практикум
Глава 15. Пароли, хеши и аутентификация
Глава 16. Эксплуатация и Metasploit в учебной среде
Глава 17. Сетевые атаки и Wi-Fi: что надо понимать
Глава 18. Active Directory: как устроены корпоративные сети
Глава 19. Обратный инжиниринг (reverse engineering)
Глава 20. Социальная инженерия: взлом человека
Глава 21. Немного программирования: Bash и Python для хакера
Глава 22. Отчёт — ваш главный продукт
Глава 23. Карьера, баг-баунти и сертификаты
Глава 24. План обучения на 9 месяцев
Шпаргалка команд
Глоссарий
Ресурсы для обучения
Шпаргалка по обратному инжинирингу
Чеклист защиты от социальной инженерии
Практикум: задачи с решениями
Часть II. Подробный учебный курс и практикумы
Глава 1. Правила безопасной практики
Глава 2. Компьютер и операционная система
Глава 3. Сеть: адреса, порты и DNS
Глава 4. HTTP и веб-приложения
Глава 5. Linux и терминал
Глава 6. Текст, Python и Git
Глава 7. Учебная лаборатория
Глава 8. Методика исследования
Глава 9. Разведка и карта приложения
Глава 10. Сканирование своей лаборатории с Nmap
Глава 11. Веб-прокси
Глава 12. Аутентификация и контроль доступа

Глава 13. Валидация и инъекции
Глава 14. XSS и защита вывода
Глава 15. Пароли и секреты
Глава 16. Логика и конфигурация
Глава 17. Отчёты и исправление
Глава 18. Итоговый проект и план обучения
Часть VI. Практикум с пошаговым разбором
Часть VII. Теория, на которой держится практика
Приложение. Шаблон журнала
Краткий словарь
Источники для дальнейшего обучения

Часть I. Основной текст книги

Основной текст исходного издания сохранён. Заголовки и абзацы размечены стилями для удобной навигации и адаптивной электронной вёрстки.

Для абсолютного новичка: начните отсюда

Если вы открыли эту книгу и уже на первых страницах встретили непонятные слова — не пугайтесь. Эта короткая глава написана специально для тех, кто начинает с полного нуля: кто ни разу не открывал терминал, не знает, что такое IP-адрес, и вообще не уверен, «потянет ли». Потянете. Давайте разложим всё по полочкам.

Правда, которую полезно знать сразу

Никто не родился со знанием хакинга. Каждый специалист когда-то не знал, что такое cd. Разница между новичком и профи — только в количестве часов практики.

Непонимание — это нормальная часть учёбы, а не признак, что «это не моё». Вы будете много раз что-то не понимать. Это и есть процесс. Перечитали, попробовали руками, спросили — поняли.

Не нужно понимать всё сразу. Достаточно понимать текущий шаг. Остальное прояснится позже само.

Главное — делать руками. Чтение без практики похоже на просмотр видео о плавании: в воде это не поможет.

Что нужно, чтобы начать

1. Компьютер с 8 ГБ оперативной памяти (лучше 16). Подойдёт обычный ноутбук.
2. Интернет — скачать бесплатные программы.
3. Тетрадь или файл для заметок — записывать команды и что они делают.
4. Время — лучше по часу регулярно, чем 8 часов раз в месяц.
5. Терпение — результат приходит через недели, а не за вечер.

Денег не нужно: всё программное обеспечение в книге бесплатное.

Мини-словарь самых базовых слов

Прежде чем читать главы, познакомьтесь с десятком слов, которые будут встречаться постоянно. Не надо заучивать — просто прочитайте, чтобы они перестали пугать.

Слово Простыми словами

Терминал (консоль) Чёрное окно, где команды печатают текстом, а не кликают

мышкой

Команда Текстовая инструкция компьютеру, например ls — «покажи

файлы»

Слово Простыми словами

Сервер Компьютер, который постоянно работает и что-то «отдаёт»

(сайт, почту)

IP-адрес «Адрес дома» компьютера в сети, например 192.168.1.10

Порт «Номер квартиры» в этом доме: за одним адресом много

служб

Сайт / веб-приложение Программа, работающая через браузер

Браузер Программа для просмотра сайтов (Chrome, Firefox)

Уязвимость Ошибка в программе, которой можно воспользоваться

Эксплойт Способ воспользоваться уязвимостью

Виртуальная машина (ВМ) «Компьютер внутри компьютера» в отдельном окне — наш

безопасный полигон

Хакер (белый) Специалист, который ищет уязвимости легально, с

разрешения, и помогает их закрыть

Если встретите незнакомое слово дальше — загляните в большой глоссарий (Приложение Б) в конце книги.

Самое важное правило всей книги

ВНИМАНИЕ

Всё, чему учит эта книга, разрешено применять только к своим учебным машинам, специальным тренировочным сайтам и системам, на проверку которых у вас есть письменное разрешение. Проверять чужой сайт или чужую сеть «просто посмотреть» — это нарушение закона, даже если вы ничего не сломали и хотели помочь. Это не формальность: именно это отличает профессию от преступления. Подробно — в главе 1.

Как читать эту книгу, чтобы не бросить

1. Идите по порядку. Главы выстроены от простого к сложному. Перепрыгивать — значит не понимать.

2. Каждую главу закрепляйте практикой. В конце каждой главы есть задания. Глава без практики не считается пройденной.

3. Не застревайте надолго. Не поняли абзац с третьего раза — отметьте и идите дальше, часто становится ясно позже.

4. Ведите журнал. Записывайте команды и что они делают. Через месяц это ваш личный учебник.

5. Повторяйте. Вернуться к главе второй раз — это не стыдно, а правильно.

«Я сломаю компьютер». Нет. Мы работаем в виртуальной машине — отдельном «компьютере в окне». Что бы там ни случилось, ваш настоящий компьютер в безопасности, а ВМ откатывается к сохранённому состоянию за секунды.

«Я не программист, у меня не получится». Программирование понадобится, но позже и в небольшом объёме. Начинают все без него.

«Это слишком сложно». Сложно выглядит целое. По одному шагу — посильно каждому. Книга и разбивает всё на шаги.

«У меня нет мощного компьютера». Для старта хватает обычного. Многое можно делать и в бесплатных онлайн-тренажёрах (TryHackMe, PortSwigger Academy) прямо в браузере.

Ваши первые три шага прямо сейчас

1. Прочитайте главу 1 (про закон и этику) — это фундамент, без которого нельзя.
2. Прочитайте главу 2 (как работают компьютер и сеть) — не спеша, с заметками.
3. Зарегистрируйтесь на TryHackMe и пройдите любую бесплатную вводную «комнату» — почувствуете вкус практики.

Куда смотреть, если запутались

Глоссарий (Приложение Б) — расшифровка терминов.

Шпаргалка команд (Приложение А) — быстрый справочник.

Практикум с решениями (в конце книги) — задачи с полным разбором ответов по каждой теме. Если глава не уложилась в голове, порешайте задачи оттуда — на практике всё встаёт на места.

Готовы? Тогда вперёд, в главу 1. И помните: вы не обязаны понимать всё сразу — вы обязаны только не бросать.

Как читать эту книгу

Эта книга написана для человека, который никогда не работал в терминале, не знает, что такое IP-адрес, и просто услышал слова «белый хакер» и захотел понять, с чего начать. Никакого предварительного образования не требуется. Требуется только компьютер и желание разбираться.

Что такое «белый хакинг»

Хакер — это не преступник. Изначально слово означало человека, который глубоко понимает, как устроена техника, и умеет заставить её делать то, что не предусмотрел производитель. Дальше появились цвета шляп:

Цвет Кто это Что делает

Белая шляпа (white Этичный хакер, Ищет уязвимости с разрешения

hat) пентестер, специалист по владельца, отдаёт отчёт, получает за

ИБ это деньги легально

Чёрная шляпа (black Преступник Взламывает ради кражи,

hat) вымогательства, вандализма. Это

уголовная статья

Серая шляпа (grey hat) «Я только посмотреть» Лезет без разрешения, но без злого

умысла. Юридически это всё равно

нарушение

Белый хакер делает ровно то же технически, что и чёрный, но с двумя принципиальными отличиями: у него есть письменное разрешение и он сообщает о найденном владельцу, а не использует находку.

ЗАМЕТКА

Вся разница между хорошо оплачиваемой профессией и уголовным делом — один документ. Договор на тестирование, scope баг-баунти программы или ваш собственный ноутбук в вашей собственной локальной сети. Никакого третьего варианта нет.

Чем на самом деле занимается белый хакер

Представьте, что вы строитель, которого наняли проверить новый дом. Вы не ломаете стены ради удовольствия. Вы методично проверяете: закрываются ли окна, выдержит ли балкон, не замурован ли пожарный выход, можно ли открыть входную дверь скрепкой. Потом вы пишете хозяину список: «вот тут замок бумажный, тут проводка опасна, вот такой ремонт сделать срочно, а такой — можно потом».

Белый хакер делает то же самое с сайтами, мобильными приложениями, серверами и сетями. Типичные направления работы:

Веб-пентест — поиск уязвимостей в сайтах и веб-приложениях. Самый массовый и самый доступный новичку путь.

Пентест инфраструктуры — внутренние сети компании, серверы, Active Directory, принтеры, камеры.

Мобильный пентест — Android- и iOS-приложения.

Баг-баунти — свободная охота по программам компаний (HackerOne, Bugcrowd, Яндекс, VK, Standoff 365), оплата за каждую найденную уязвимость.

Red Team — имитация реальной атакующей группировки на длинной дистанции, с социальной инженерией и обходом защиты.

Blue Team / SOC-аналитик — защита: слежение за логами, реакция на инциденты. Знания атак обязательны и здесь.

AppSec / DevSecOps — встраивание безопасности в процесс разработки.

Сколько это стоит и сколько платят

Честно: вход в профессию занимает от 6 до 18 месяцев регулярной практики, если вы начинаете с нуля. Первые деньги в баг-баунти обычно приходят через 3–8 месяцев после начала практики, и это будут небольшие суммы. Джуниор-позиция пентестера в 2026 году — реальная цель через год упорного обучения. Денег на старте почти не нужно: всё ПО в этой книге бесплатное, нужен только компьютер с 8 ГБ оперативной памяти (лучше 16).

Чего в этой книге нет

Готовых инструментов для взлома чужих систем и обхода антивирусов.

Инструкций «как взломать Wi-Fi соседа» или «как прочитать чужой WhatsApp». Это преступления, и на них здесь ответов нет.

Волшебной кнопки. Хакинг — это чтение документации, терпение и много-много попыток.

Как устроена книга

Часть I (главы 1–3) — фундамент: закон, как работает компьютер и сеть. Часть II (главы 4–6) — Linux, терминал и собственная лаборатория. Часть III (главы 7–14) — разведка, сканирование и веб-уязвимости на практике. Часть IV (главы 15–19) — пароли, эксплуатация, сети, автоматизация. Часть V (главы 20–22) — отчёты, карьера и план обучения. Приложения — шпаргалки, глоссарий и ресурсы.

В каждой главе есть:

простое объяснение на бытовых аналогиях;

примеры команд с разбором, что означает каждый флаг;

что вы должны увидеть в ответ;

практика — задание, которое надо выполнить своими руками;

частые ошибки новичка.

ВНИМАНИЕ

Все практические задания выполняются только на: (1) ваших собственных виртуальных машинах из главы 6, (2) специально созданных тренировочных площадках вроде TryHackMe, HackTheBox, PortSwigger Web Security Academy, (3) системах, на тестирование которых у вас есть письменное разрешение. Сканирование даже «просто посмотреть» чужого сайта — уже потенциальное правонарушение в большинстве стран.

Главный совет перед началом

Не пытайтесь выучить всё сразу. Самая частая причина, по которой новички сдаются, — попытка одновременно читать про Linux, криптографию, Active Directory, реверс-инжиниринг и Wi-Fi. Идите по главам последовательно и выполняйте практику. Глава без практики не считается прочитанной.

Глава 1. Закон, этика и как не сесть в тюрьму

Эта глава первая не случайно. Технику можно освоить за год. Загубленную биографию не исправить.

Простая аналогия

Вы умеете вскрывать замки. Это навык, он не запрещён. Вскрыть замок на своей двери — нормально. Вскрыть замок на двери, которую вам показал владелец и попросил проверить, — работа. Вскрыть замок в чужом подъезде, ничего не взяв и уйдя, — уже взлом с проникновением. «Я ничего не украл» и «я хотел помочь» не являются оправданием нигде в мире.

Что считается нарушением

Почти во всех правовых системах наказуемо само неправомерное получение доступа, даже без ущерба:

Подбор или использование чужого пароля.

Сканирование портов и уязвимостей чужой инфраструктуры без разрешения (в ряде стран трактуется как подготовка к доступу).

Эксплуатация уязвимости на чужом сайте, даже если вы потом написали владельцу.

Скачивание чужих данных, даже если они «лежали открыто».

DoS/DDoS — отказ в обслуживании, в любом масштабе.

Создание и распространение вредоносного ПО.

ВНИМАНИЕ

Классическая ловушка новичка: нашёл уязвимость на чужом сайте, сам её проэксплуатировал «для доказательства», написал владельцу — и получил не благодарность, а заявление в полицию. Если у компании нет публичной программы баг-баунти или политики раскрытия (/.well-known/security.txt), вы не имеете права ничего проверять.

Что даёт вам право тестировать

1. Договор на пентест — отдельный документ со scope (что можно), сроками, контактами и подписями. Без него работа не начинается никогда.

2. Публичная программа баг-баунти — её правила и есть ваше разрешение. Читать scope до первой команды, а не после.

3. Политика ответственного раскрытия (responsible disclosure / VDP) —

4. Ваша собственная лаборатория — виртуальные машины на вашем компьютере в изолированной сети.

5. Тренировочные платформы — TryHackMe, Hack The Box, PortSwigger Academy, OWASP Juice Shop, Standoff 365, root-me. Созданы, чтобы их ломали.

Как читать scope баг-баунти

Scope — это список целей, разрешённых к тестированию. Пример типичной формулировки:

IN SCOPE (можно):

*.example.com — все поддомены

api.example.com — API

Android app com.example — мобильное приложение

OUT OF SCOPE (нельзя):

blog.example.com — работает на сторонней CMS

payments.example.com — платёжный контур

Любой DoS / нагрузочное тестирование

Социальная инженерия сотрудников

Автоматические сканеры на продакшене

Физический доступ в офисы

Правило: если цели нет в списке «можно» — значит нельзя. Не «не написано, что нельзя», а именно нельзя.

ЗАМЕТКА

Отдельно обратите внимание на запрет автоматических сканеров. Многие новички запускают по цели агрессивный сканер, роняют сайт и вместо выплаты получают бан и претензию. Сначала руками, аккуратно, в один поток.

Личные правила, которые спасут вам карьеру

Не трогаю то, на что нет письменного разрешения. Даже один пакет.

Не выхожу за пределы scope, даже если «там рядом явно дырка».

Минимально достаточное доказательство. Нашёл SQL-инъекцию — доказал версией базы, а не выгрузкой таблицы с клиентами.

Нашёл чужие персональные данные — остановился, не скачивал, зафиксировал факт, сообщил.

Никогда не меняю и не удаляю данные на чужой системе.

Не публикую находку, пока её не исправили и не разрешили публиковать.

Логирую свои действия: что, когда, с какого адреса. Это защищает и меня тоже.

Не использую рабочие инструменты «по приколу» против знакомых.

Практика к главе 1

1. Зарегистрируйтесь на HackerOne и откройте любую публичную программу.

2. Найдите у трёх крупных компаний файл политики: откройте https://<домен>/.well-known/security.txt. Посмотрите, у кого он есть и что там написано.

3. Своими словами напишите 5 строк: чем ваша будущая работа отличается от преступления. Это будет ваш личный кодекс.

Частые ошибки новичка

«Я учился, я не знал» — не работает как защита.

Использование чужих аккаунтов «для теста».

Тестирование сайта работодателя без разрешения, чтобы «показать инициативу». Увольнение и заявление — типичный финал.

Запуск найденных на форумах скриптов, не читая их код. Половина таких «инструментов» — трояны, которые украдут ваши же данные.

Глава 2. Как вообще работает компьютер и сеть

Нельзя взламывать то, чего не понимаешь. Девять из десяти неудач новичка — не от незнания хакерских трюков, а от незнания основ. Эта глава даёт минимальный фундамент простыми словами.

Компьютер в четырёх словах

Процессор считает. Он умеет только очень простые операции, но миллиарды в секунду.

Оперативная память (RAM) — рабочий стол. Всё, что сейчас работает, лежит там. Выключили питание — стол чист.

Диск — шкаф. Хранит файлы и после выключения.

Операционная система (ОС) — администратор, который решает, какая программа получит процессор, память и доступ к файлам.

Ключевая мысль для будущего хакера: программа — это просто данные, которые процессор согласился исполнить. Большинство серьёзных уязвимостей — это ситуации, когда данные пользователя случайно оказались в роли команды. Запомните эту фразу, она объясняет SQL-инъекции, XSS, command injection и переполнение буфера одновременно.

Пользователи и права

ОС делит всех на пользователей. У каждого файла есть владелец и права. Есть один особый пользователь с абсолютной властью: root в Linux, Administrator / SYSTEM в Windows.

Почти любая атака на систему — это два шага:

1. Получить любой доступ (хоть самый ограниченный).
2. Повысить его до административного (privilege escalation).

Сеть: как данные доходят из точки А в точку Б

Представьте обычную почту.

IP-адрес — адрес дома. Например, 192.168.1.10.

Порт — номер квартиры в доме. Веб-сервер обычно живёт в квартире 80 или 443, SSH — в 22.

Пакет — конверт с письмом: адрес отправителя, адрес получателя, содержимое.

Маршрутизатор (роутер) — почтовое отделение, которое знает, куда передать конверт дальше.

Модель OSI и TCP/IP — без зауми

Данные упаковываются слоями, как матрёшка. Для практики достаточно знать четыре уровня:

Уровень Что делает Примеры Где ломают

Канальный (L2) Передача внутри Ethernet, Wi-Fi, ARP ARP-спуфинг, атаки

одного сегмента сети на Wi-Fi

по MAC-адресам

Сетевой (L3) Доставка между IP, ICMP подмена IP,

сетями по IP маршрутизация

Транспортный (L4) Доставка TCP, UDP сканирование портов

конкретному

приложению по

порту

Прикладной (L7) Сам смысл данных HTTP, DNS, SMTP, SSH 90% работы

веб-пентестера

TCP и UDP

TCP — звонок по телефону: сначала «алло, слышно меня?», потом разговор, потом «до связи». Гарантирует доставку и порядок. Используется для сайтов, почты, SSH.

Установление соединения называется «тройное рукопожатие»:

Клиент → SYN → Сервер «Можно поговорить?»

Клиент ← SYN/ACK ← Сервер «Да, можно»

Клиент → ACK → Сервер «Отлично, начинаю»

UDP — открытка: бросил в ящик и забыл. Быстро, без гарантий. Используется для DNS, видео, игр, VoIP.

ЗАМЕТКА

Понимание рукопожатия нужно практически: сканер nmap в «тихом» режиме -sS отправляет SYN и, получив SYN/ACK, обрывает соединение, не досылая ACK. Так он узнаёт, что порт открыт, не устанавливая полноценную связь.

Частные и публичные адреса

Некоторые диапазоны IP зарезервированы для внутренних сетей и не маршрутизируются в интернете:

10.0.0.0 – 10.255.255.255

172.16.0.0 – 172.31.255.255

192.168.0.0 – 192.168.255.255

127.0.0.1 — localhost, «я сам».

Ваш домашний роутер даёт устройствам адреса из 192.168.x.x, а в интернет выходит через один публичный адрес с помощью NAT (подмена адреса). Именно поэтому нельзя «просто так» подключиться к компьютеру внутри чужой домашней сети — и именно поэтому атаки чаще начинаются с того, что жертва сама открыла ссылку или вложение.

Маска подсети и CIDR

Запись 192.168.1.0/24 означает: первые 24 бита — это адрес сети, остальные 8 бит — адреса устройств. То есть диапазон 192.168.1.0–192.168.1.255, примерно 254 рабочих адреса. Это то, что вы будете писать в сканере, когда захотите проверить всю свою лабораторную сеть.

Запись Количество адресов Когда используется

/32 1 Один конкретный хост

/24 256 Типичная домашняя или офисная

подсеть

/16 65 536 Большая корпоративная сеть

/8 16 777 216 Огромный блок, сканировать не надо

HTTP — язык веба

Браузер и сервер общаются текстом. Запрос выглядит так:
GET /profile?id=42 HTTP/1.1

Host: shop.example.com

User-Agent: Mozilla/5.0

Cookie: session=a1b2c3d4e5

Accept: text/html
Ответ:
HTTP/1.1 200 OK

Content-Type: text/html; charset=utf-8

Set-Cookie: session=a1b2c3d4e5; HttpOnly; Secure

Content-Length: 1534
<html><body>Привет, Иван</body></html>
Разберём по частям:
GET — метод. Основные: GET (получить), POST (отправить данные), PUT (заменить), DELETE (удалить), OPTIONS (узнать возможности).
/profile?id=42 — путь и параметр. id=42 — это то, чем управляет пользователь, а значит первое место, куда смотрит хакер.
Host — какой сайт нужен (на одном IP могут жить сотни сайтов).
Cookie — ваш «пропуск», доказывающий серверу, что вы уже вошли.
Коды ответов, которые надо знать на память
Код Значение Что это говорит хакеру
200 OK Запрос сработал
301 / 302 Перенаправление Куда-то уводят, стоит посмотреть куда
400 Плохой запрос Сломали синтаксис — иногда так

находят инъекции
401 Нужна аутентификация Есть закрытая зона
403 Запрещено Ресурс есть, но доступ закрыт — цель

для обхода
404 Не найдено Нет такого
405 Метод не разрешён Попробуйте другой метод
500 Ошибка сервера Золото. Что-то упало от ваших данных
ЗАМЕТКА
Код 500 после подставленной кавычки в параметр — классический первый признак SQL-инъекции. Код 403 там, где ожидалась админка, — признак, что админка существует.

Шифрование простыми словами

Хеш — односторонняя «мясорубка». Из пароля получается строка фиксированной длины, обратно восстановить нельзя. Используется для хранения паролей. Примеры: MD5 (устарел), SHA-1 (устарел), SHA-256, bcrypt, argon2 (современные и правильные).

Симметричное шифрование — один ключ и на закрытие, и на открытие. AES.

Асимметричное шифрование — два ключа: публичным закрывают, приватным открывают. RSA, ECDSA. Так работают HTTPS и SSH-ключи.

TLS/HTTPS — конверт, который нельзя прочитать по дороге. Защищает содержимое, но не защищает сайт от уязвимостей. HTTPS-сайт ломается ровно так же, как HTTP-сайт.

Хеш против шифрования

Пароль: qwerty123

MD5: 3f7b2a18b6f0c4b0e1a9e8b1d2c3a4b5 (нельзя расшифровать)

↑ но можно подобрать перебором, если пароль слабый

AES(ключ="secret"): 8af2... (можно расшифровать ключом)

Это различие критично: когда вы «взламываете хеш», вы не расшифровываете его — вы перебираете варианты, хешируете каждый и сравниваете. Поэтому длинный

Практика к главе 2

1. Откройте командную строку и выполните ping example.com. Запишите, какой IP-адрес отвечает и сколько миллисекунд идёт ответ.

2. Выполните nslookup example.com (или dig example.com). Найдите в выводе IP и сервер, который дал ответ.

3. Узнайте свой внутренний адрес: ipconfig в Windows, ip a в Linux. Какая у вас подсеть? Какой адрес у роутера (шлюза)?

4. Откройте любой сайт, нажмите F12 → вкладка Network → перезагрузите страницу. Найдите первый запрос и изучите его заголовки запроса и ответа. Найдите Set-Cookie.

5. Выполните tracert example.com (Windows) или traceroute example.com (Linux) и посмотрите, через сколько «почтовых отделений» идёт ваш пакет.

Частые ошибки новичка

Пропустить эту главу и потом не понимать, почему сканер «ничего не находит» (а он просто сканирует не ту подсеть).

Путать порт и IP-адрес.

Думать, что HTTPS защищает сайт от взлома.

Считать, что хеш можно «расшифровать».

Глава 3. Операционные системы: Linux и Windows глазами хакера

Почему Linux

Почти вся серверная часть интернета работает на Linux. Все профессиональные инструменты написаны для Linux в первую очередь. Linux бесплатен, прозрачен и полностью управляется из терминала — а значит, автоматизируется.

Это не значит «выбросьте Windows». Рабочая схема для новичка: Windows как основная система + Linux в виртуальной машине. Как её поставить — в главе 6.

Структура файловой системы Linux

В Windows есть диски C:, D:. В Linux всё растёт из одного корня /.
Путь Что там Зачем хакеру
/ Корень всего —
/home/имя Домашние папки Личные файлы, ключи SSH, история

пользователей команд
/root Домашняя папка root Главная цель
/etc Все настройки системы /etc/passwd, /etc/shadow, конфиги

сервисов
/var/log Логи Что видел сервер, следы
/var/www Файлы сайтов Код веб-приложения
/tmp Временные файлы, Куда загрузить свой скрипт

доступны всем на запись
/usr/bin, /bin Программы Чем можно воспользоваться на цели
/proc Виртуальная информация Что запущено, с какими правами

о процессах
Два файла, которые надо знать
/etc/passwd — список пользователей, читается всеми:
root:x:0:0:root:/root:/bin/bash

www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin

ivan:x:1000:1000:Ivan:/home/ivan:/bin/bash

Поля через двоеточие: имя, пароль (x = лежит в другом файле), UID, GID, описание, домашняя папка, оболочка. UID 0 означает полную власть. Если вы видите второго пользователя с UID 0 — это уже находка.

root:6xyz123$aBcD...:19800:0:99999:7:::

6 означает SHA-512. y — yescrypt, $2b$ — bcrypt. Если вы как пентестер получили доступ к этому файлу — вы фактически получили систему.

Права доступа — самое важное в Linux

$ ls -l

-rw-r--r-- 1 ivan ivan 1024 Oct 23 10:00 notes.txt

drwxr-xr-x 2 ivan ivan 4096 Oct 23 10:00 projects

-rwsr-xr-x 1 root root 44512 Oct 23 10:00 passwd

Читаем первую строку символ за символом:

- — обычный файл (d — папка, l — ссылка).

rw- — права владельца: читать, писать, не исполнять.

r-- — права группы: только читать.

r-- — права всех остальных: только читать.

Числовая запись: r=4, w=2, x=1. Значит rw-r--r-- = 644, rwxr-xr-x = 755.

Бит SUID — почему он важен для защиты

Посмотрите на третий файл в примере выше: -rwsr-xr-x. Вместо x у владельца стоит s. Это бит SUID: программа выполняется с правами своего владельца, а не того, кто её запустил. Для некоторых системных программ (например passwd, которой нужно менять защищённый файл паролей) это нормально и необходимо.

Почему это важно специалисту по безопасности: если администратор по ошибке поставит SUID-root на программу, которая умеет запускать другие команды или редактировать произвольные файлы, то обычный пользователь системы сможет выйти за рамки своих прав. Поэтому одна из стандартных проверок при аудите защищённости Linux — составить список всех SUID-программ и сверить его со списком «так и должно быть».

Как такой список составляют при аудите собственной (или разрешённой к проверке) машины:

find / -perm -4000 -type f 2>/dev/null

find / — искать от корня;

-perm -4000 — где установлен бит SUID;

-type f — только файлы;

2>/dev/null — спрятать ошибки «нет доступа».

ЗАМЕТКА

Справочник GTFOBins (gtfobins.github.io) перечисляет, какие стандартные программы опасно оставлять с лишними правами. Защитнику он нужен, чтобы знать, что убрать из автозапуска и с каких бинарников снять SUID. Это открытая база знаний для

Windows глазами специалиста по безопасности

Windows устроен иначе, но логика та же: пользователи, права, привилегированные учётные записи.

Вместо текстовых конфигов — реестр (regedit): дерево настроек.

Главные привилегированные аккаунты — Administrator и системный SYSTEM.

Домены и Active Directory — централизованное управление тысячами компьютеров в компании. Это отдельная большая тема (глава 18).

Пароли хранятся в виде хешей (NTLM) в базе SAM и в памяти процесса lsass.

Полезные команды для изучения своей машины: whoami /priv, systeminfo, net user, ipconfig /all, tasklist.

Для защитника Windows важнее всего понимать, что в корпоративной среде компрометация одной рабочей станции редко бывает целью — целью почти всегда является доступ к контроллеру домена. Поэтому защита строится «вглубь»: сегментация сети, минимум прав у пользователей, мониторинг событий входа.

Практика к главе 3

1. В виртуальной машине с Linux (поставим в главе 6) выполните ls -l / и разберите права на 3–4 папки.

2. Посмотрите cat /etc/passwd и найдите всех пользователей с оболочкой /bin/bash.

3. Выполните команду поиска SUID-файлов из этой главы на своей учебной машине и выпишите список. Сравните его с типовым — лишнего быть не должно.

4. В Windows выполните whoami /priv и прочитайте, какие привилегии есть у вашей учётной записи.

Частые ошибки новичка

Бояться терминала. Терминал — это просто текстовый способ отдавать те же команды.
Работать под root/Administrator постоянно. Это опасно и мешает понять механику прав.
Учить Windows и Linux «потом». Реальные цели смешанные, нужны обе системы.

Углублённо: что ещё важно знать о сети

NAT, порты и почему вас «не видно» из интернета

Домашние устройства имеют частные адреса (192.168.x.x). В интернет они выходят через роутер, который подменяет адрес (NAT) и ведёт таблицу «какой

снаружи виден только роутер, а не ваши устройства напрямую. Это объясняет, почему удалённые атаки на домашние машины обычно требуют, чтобы жертва сама инициировала соединение (открыла ссылку, установила программу), — и почему «проброс портов» на роутере надо делать осознанно.

DNS подробнее: цепочка разрешения имени

Браузер → кэш ОС → DNS-резолвер провайдера → корневые серверы

→ серверы зоны (.com) → авторитетный сервер домена → IP

Понимание этой цепочки пригодится в разведке (глава 7): разные записи (A, MX, TXT, NS) рассказывают об инфраструктуре организации.

Базовые протоколы прикладного уровня

Протокол Порт Назначение Безопасная замена

HTTP 80 Веб без шифрования HTTPS (443)

FTP 21 Передача файлов, SFTP/FTPS

открытый текст

Telnet 23 Удалённая консоль, SSH (22)

открытый текст

SMTP 25/587 Отправка почты с TLS

DNS 53 Разрешение имён DoH/DoT

SMB 445 Файлы/принтеры с ограничениями

Windows доступа

RDP 3389 Удалённый рабочий через VPN, с MFA

стол Windows

Знание «кто на каком порту живёт» ускоряет чтение результатов сканирования: увидели 3389 — вероятно, Windows с удалённым столом; 445 — файловые службы; 22 — SSH.

Модель «запрос-ответ» и состояние

HTTP сам по себе «без памяти» (stateless): сервер не помнит вас между запросами. Чтобы «узнавать» вошедшего пользователя, применяют cookie с идентификатором сессии. Отсюда вытекает целый класс тем: кража/подделка сессий, безопасность cookie (HttpOnly, Secure, SameSite), токены. Это фундамент для глав про веб.

Глава 4. Терминал Linux: ваш главный инструмент

Терминал пугает новичков, но на деле это просто окно, где вы печатаете команду и жмёте Enter. Освоив 30 команд из этой главы, вы будете чувствовать себя в Linux увереннее большинства.

Как устроена команда

команда -флаги аргументы

| | |

что как над чем
Например ls -l /home: ls — показать содержимое, -l — подробно, /home — что именно.

Навигация

Команда Что делает Пример
pwd Показать, где я сейчас pwd
ls Список файлов ls -la (включая скрытые)
cd Перейти в папку cd /var/log
cd .. На уровень вверх cd ..
cd ~ В домашнюю папку cd ~
tree Дерево папок tree -L 2
ЗАМЕТКА

Скрытые файлы в Linux начинаются с точки (.bashrc, .ssh). Их показывает только ls -a. При исследовании системы именно в них часто лежат ключи, токены и история — поэтому ls -la важнее, чем простой ls.

Работа с файлами

Команда Что делает
cat file Показать весь файл
less file Листать большой файл (q — выход)
head -n 20 file Первые 20 строк
Команда Что делает
cp a b Копировать
mv a b Переместить/переименовать
rm file Удалить (осторожно, корзины нет)
mkdir dir Создать папку
touch file Создать пустой файл
nano file Простой текстовый редактор

Поиск — то, что вы будете делать постоянно

```
find / -name "*.conf" 2>/dev/null # найти все .conf по имени

grep -r "password" /etc/ 2>/dev/null # найти строку в файлах

grep -i "error" app.log # без учёта регистра

locate nmap # быстрый поиск по базе

which python3 # где лежит программа
```

grep — самый важный инструмент фильтрации. Он ищет строки по шаблону. Будущему хакеру он нужен, чтобы найти пароли в конфигах, интересные строки в логах, свои результаты в гигабайтах вывода сканера.

Конвейеры и перенаправление — суперсила терминала

Вывод одной команды можно передать на вход другой через | (пайп):

cat access.log | grep "admin" | wc -l

Читается так: взять лог → оставить строки со словом admin → посчитать их количество. Три простые программы решили задачу, для которой в Windows пришлось бы искать отдельную утилиту.

Перенаправление в файл:

nmap -sV 192.168.56.0/24 > scan.txt # записать вывод в файл echo "заметка" >> notes.txt # дописать в конец command 2>/dev/null # выбросить ошибки

Права и процессы

chmod +x script.sh # сделать файл исполняемым

chmod 644 file # выставить права числом

chown ivan file # сменить владельца (нужен root)

sudo команда # выполнить от имени root

ps aux # все процессы

sudo — «выполни это от имени администратора». Правильная привычка: работать обычным пользователем и писать sudo только там, где действительно нужны права root.

Сеть из терминала

ip a # мои адреса

ip route # таблица маршрутизации

ping -c 4 8.8.8.8 # проверить доступность

curl http://site.com # скачать страницу как текст

wget http://site/f.zip # скачать файл

ss -tulpn # какие порты слушает моя машина

curl вы полюбите: он делает HTTP-запросы из командной строки и показывает сырой ответ сервера — то, что скрывает браузер.

curl -I https://example.com # только заголовки ответа curl -X POST -d "user=ivan" URL # POST-запрос с данными curl -H "Authorization: Bearer X" URL # со своим заголовком

Установка программ

В Debian/Ubuntu/Kali используется менеджер пакетов apt:

sudo apt update # обновить списки пакетов sudo apt install nmap # установить программу sudo apt upgrade # обновить всё

Горячие клавиши, которые экономят часы

Tab — автодополнение имён файлов и команд. Жмите постоянно.
Стрелка вверх — предыдущая команда.
Ctrl+C — прервать текущую программу.
Ctrl+R — поиск по истории команд.
Ctrl+L — очистить экран.
history — показать все прошлые команды.

Практика к главе 4

1. Создайте папку lab, зайдите в неё, создайте файл notes.txt, запишите в него строку командой echo.

2. Найдите на своей учебной машине все файлы .log в /var/log и посчитайте их количество одной строкой с пайпом.

3. Выполните curl -I https://example.com и определите, какой веб-сервер отвечает (заголовок Server).

5. Напишите команду, которая из /etc/passwd выводит только имена пользователей (подсказка: cut -d: -f1).

Частые ошибки новичка

Копировать команды из интернета, не понимая каждого флага. Разбирайте по кусочкам.
rm -rf без раздумий. В Linux нет корзины.
Забывать sudo и винить систему в «ошибке доступа».
Не пользоваться Tab и набирать длинные пути руками с опечатками.

Глава 5. Инструменты белого хакера

Инструмент — это не магия, а усилитель. Он делает быстро то, что вы в принципе могли бы сделать руками. Поэтому важно понимать, что он делает, а не просто запускать.

Карта инструментов по этапам работы

Этап Задача Основные инструменты
Разведка Собрать информацию, не theHarvester, Amass, whois, Google

трогая цель dorking
Сканирование Найти живые хосты, nmap, masscan, naabu

порты, сервисы
Веб-анализ Изучить веб-приложение Burp Suite, OWASP ZAP, браузер +

DevTools
Поиск директорий Найти скрытые страницы ffuf, gobuster, dirsearch
Уязвимости Проверить известные Nikto, nuclei, OWASP ZAP

слабости
Пароли Проверить стойкость hashcat, John the Ripper, hydra
Эксплуатация Подтвердить уязвимость Metasploit (в учебной среде)
Пост-анализ Разобрать и описать своя голова, документация находку

Четыре инструмента, которые надо освоить первыми

1. Nmap — «кто дома и какие двери открыты»

Сканер сети. Показывает, какие устройства в сети живы, какие порты открыты и какие программы за ними стоят. Это первое, что делает пентестер, получив разрешённый диапазон адресов. Подробно — в главе 8.

2. Burp Suite — «микроскоп для веба»

Прокси, который встаёт между браузером и сайтом и позволяет видеть и изменять каждый HTTP-запрос. Главный инструмент веб-пентестера. Есть бесплатная Community-версия, её хватит надолго. Подробно — в главе 10.

3. ffuf / gobuster — «перебор скрытых дверей»

Берёт список возможных имён (словарь) и проверяет, какие страницы и папки реально существуют на сайте. Так находят /admin, /backup, /.git — то, на что нет ссылок, но что существует.

F12 в браузере. Вкладки Network, Console, Application, Storage. Половину простых уязвимостей в веб-приложениях видно прямо здесь, без единого платного инструмента.

Kali Linux и аналоги

Kali Linux — дистрибутив Linux, в котором сотни инструментов уже установлены. Для новичка это удобно: не надо ставить каждый по отдельности. Альтернативы — Parrot OS и BlackArch. В главе 6 мы поставим Kali в виртуальную машину.

ВНИМАНИЕ

Kali — это инструмент для работы по разрешённым целям, а не «система для взлома». Сам факт её установки ничего не разрешает. Все правила главы 1 действуют в полном объёме.

Как не утонуть в инструментах

Новичок видит сотни программ в меню Kali и теряется. Делайте проще: на первые три месяца вам нужны только nmap, Burp, ffuf, curl и браузер. Всё остальное изучается по мере появления конкретной задачи. Инструмент без понимания задачи бесполезен.

Практика к главе 5

1. Составьте для себя таблицу «этап → инструмент → что делает» из этой главы и держите под рукой.

2. Посмотрите на YouTube одно короткое demo Burp Suite Community, просто чтобы увидеть интерфейс (запускать пока не надо).

3. Выпишите пять инструментов, которые вы освоите первыми, и напротив каждого — одну задачу, которую он решает.

Углублённо: текстовая обработка и автоматизация

Три кита обработки текста

Вывод любой программы — это текст, и умение его «резать» отличает уверенного пользователя.

grep "шаблон" файл # оставить строки по шаблону

awk '{print $2}' файл # работать по столбцам

sed 's/a/b/g' файл # заменять/редактировать потоково

Пример разбора лога веб-сервера: найти топ-10 адресов по числу запросов.

awk '{print $1}' access.log | sort | uniq -c | sort -rn | head

Читается так: взять первый столбец (IP) → отсортировать → посчитать повторы → программу.

Регулярные выражения — минимум

. любой символ ^ начало строки

* 0 и более повторов $ конец строки

[0-9] цифра \. точка буквально

+ 1 и более (grep -E) | или (grep -E)

Пример: найти в тексте что-то похожее на email.

grep -E "[a-z0-9._]+@[a-z0-9.]+\.[a-z]+" файл

Переменные, подстановка и циклы

TARGET="192.168.56.101"

echo "Сканирую $TARGET"

for p in 22 80 443; do echo "порт $p"; done

files=$(ls *.txt)

Фоновые процессы и длительные задачи

команда & # запустить в фоне

jobs # список фоновых задач

nohup команда & # не прервётся при выходе из сессии

tmux # менеджер сессий: не терять длинные сканы

tmux особенно полезен: запустили долгий скан, отключились, вернулись — всё на месте.

Права числом: быстрый разбор

chmod 755 file # rwxr-xr-x владельцу всё, остальным читать+выполнять

chmod 644 file # rw-r--r-- владельцу чтение/запись, остальным чтение

chmod 600 file # rw------- только владелец (так держат приватные ключи)

chmod +x s.sh # добавить право на выполнение
SSH — рабочая лошадка удалённого доступа
ssh user@host # подключиться

ssh -i key.pem user@host # по ключу

scp file user@host:/path # скопировать файл на хост

ssh-keygen -t ed25519 # сгенерировать свою пару ключей

Ключи SSH безопаснее паролей: приватный ключ (права 600!) не передаётся по сети. Понимание SSH-ключей пригодится и как пользователю, и в разведке (найденные ключи — ценная находка в разрешённом тесте).

Глава 6. Собираем собственную лабораторию

Это самая важная практическая глава в начале книги. Лаборатория — это несколько виртуальных машин на вашем компьютере, соединённых в изолированную сеть. В ней вы можете атаковать всё, что угодно, не нарушая ни одного закона, потому что это ваши машины в вашей сети.

Зачем нужна виртуализация

Виртуальная машина (ВМ) — это компьютер внутри компьютера: отдельная ОС, работающая в окне. Плюсы для нас:

Полная изоляция: что бы ни сломалось внутри, ваш основной компьютер цел.

Снимки (snapshots): сделали «фото» состояния, наэкспериментировали, откатились назад за секунды.

Безопасность: уязвимую учебную машину можно держать в сети, куда нет выхода в интернет.

Что устанавливаем

Компонент Роль Где взять
VirtualBox (бесплатно) Сам «движок» virtualbox.org

или VMware виртуализации

Workstation Player
Kali Linux (образ для Машина атакующего kali.org/get-kali

VirtualBox)
Metasploitable 2 / 3 Нарочно уязвимая мишень SourceForge / Rapid7
OWASP Juice Shop Уязвимое веб-приложение через Docker или npm
(опционально) Мишень на Windows microsoft.com Evaluation Center

Windows 10/11

Evaluation
ВНИМАНИЕ

Metasploitable — это машина, специально напичканная уязвимостями. Её категорически нельзя подключать к интернету или к реальной домашней сети. Только изолированная внутренняя сеть виртуальных машин. Иначе вы откроете дыру в собственный дом.

Пошаговая установка

1. Скачайте VirtualBox с официального сайта под вашу ОС.

2. Установите, соглашаясь со значениями по умолчанию.

3. Установите Extension Pack с того же сайта (нужен для USB и сети).

Шаг 2. Kali Linux

Самый простой путь — готовый образ:

1. На kali.org в разделе Virtual Machines скачайте образ именно для VirtualBox (файл .7z с .vbox и .vdi внутри).

2. Распакуйте в отдельную папку.

3. В VirtualBox: Машина → Добавить → выберите .vbox.

4. Логин/пароль по умолчанию обычно kali / kali — сразу смените пароль командой passwd.

Проверьте, что всё работает:

cat /etc/os-release # должно быть Kali GNU/Linux

ip a # есть сетевой адрес

sudo apt update # пакеты обновляются

Шаг 3. Мишень Metasploitable 2

1. Скачайте архив, распакуйте в отдельную папку.

2. В VirtualBox создайте новую машину типа Linux, при создании диска выберите «использовать существующий» → укажите .vmdk из архива.

3. Не запускайте пока. Сначала настроим изолированную сеть.

Шаг 4. Изолированная сеть — критический момент

Цель: Kali и мишени видят друг друга, но мишени не видят интернет и вашу реальную сеть.

Для каждой ВМ: Настройки → Сеть → Адаптер 1 → выбрать «Внутренняя сеть» (Internal Network), задать всем одно имя сети, например labnet.

Если Kali всё же нужен интернет (для обновлений), дайте ей второй адаптер в режиме NAT, а мишеням — только внутреннюю сеть.

Kali: [Внутренняя: labnet] + [NAT для интернета]

Metasploitable:[Внутренняя: labnet] ← и только она

Juice Shop: [Внутренняя: labnet]

Шаг 5. Проверка связи

Запустите обе машины. На Kali узнайте адрес мишени и проверьте связь:

ip a # смотрим свою подсеть, напр. 192.168.56.x

nmap -sn 192.168.56.0/24 # кто ещё живёт в этой подсети

ping -c 3 192.168.56.101 # пингуем найденную мишень

Если пинг идёт — лаборатория готова. Если нет — проверьте, что у обеих машин одинаковое имя внутренней сети.

ЗАМЕТКА

Сразу после настройки сделайте снимок (snapshot) каждой машины в состоянии «чистая, настроенная». Любой эксперимент потом откатывается к этому снимку за секунды, и вам не придётся переустанавливать всё заново.

OWASP Juice Shop через Docker

Juice Shop — современное уязвимое веб-приложение для тренировки. Проще всего через Docker на Kali:

sudo apt install docker.io -y

sudo systemctl start docker

sudo docker pull bkimminich/juice-shop

sudo docker run -d -p 3000:3000 bkimminich/juice-shop

Откройте в браузере Kali http://localhost:3000 — увидите интернет-магазин, который и будете исследовать в главах про веб.

Бесплатные онлайн-площадки (альтернатива и дополнение)

Если не хочется возиться с ВМ — начните с готовых площадок в браузере. Они легальны и созданы для обучения:

Площадка Для чего Уровень

PortSwigger Web Веб-уязвимости, лучший с нуля

Security Academy бесплатный курс

TryHackMe Пошаговые комнаты с с нуля

подсказками

Hack The Box Практика, ближе к начальный–средний

(Academy и Starting реальности

Point)

OverTheWire (Bandit) Основы Linux и терминала с нуля

в игре

root-me.org Множество задач по все

категориям

Standoff 365 Русскоязычный начальный–средний

киберполигон и

баг-баунти

Оптимальная связка для новичка: PortSwigger Academy (теория и веб-практика) + TryHackMe (общие навыки) + своя ВМ-лаборатория (свобода экспериментов).

Практика к главе 6

1. Установите VirtualBox и запустите в нём Kali. Смените пароль, обновите пакеты.
2. Поднимите Metasploitable во внутренней сети и убедитесь, что Kali его пингует, а выхода в интернет у мишени нет (проверьте ping 8.8.8.8 с мишени — не должно работать).
3. Сделайте снимки «чистого» состояния всех машин.
4. Запустите Juice Shop и откройте его в браузере Kali.
5. Зарегистрируйтесь на TryHackMe и пройдите бесплатную вводную комнату.

Частые ошибки новичка

Подключить уязвимую мишень к реальной сети или интернету. Главная ошибка безопасности.

Не сделать снимки и потом переустанавливать всё после первой же поломки.

Выделить ВМ слишком мало памяти (ставьте Kali минимум 2–4 ГБ).

Перепутать режимы сети и потом удивляться, что машины не видят друг друга.

Глава 7. Разведка: искусство собирать информацию

Любой грамотный тест начинается не с атаки, а с разведки. Чем больше вы знаете о цели заранее, тем точнее и безопаснее работа. В реальных проектах на разведку уходит до трети всего времени.

Два вида разведки

Пассивная — сбор информации из открытых источников, не трогая саму цель. Вы читаете то, что и так опубликовано: домены, сотрудники в соцсетях, утёкшие базы, кэш поисковиков. Цель даже не знает, что вы смотрите.

Активная — вы напрямую взаимодействуете с целью: сканируете порты, запрашиваете страницы. Это уже «активность на цели», и она разрешена только в рамках scope.

Начинают всегда с пассивной: она безопаснее и часто даёт больше, чем кажется.

OSINT — разведка по открытым источникам

OSINT (Open Source Intelligence) — сбор данных из публичных источников. Для пентеста полезно узнать:

Какие домены и поддомены есть у организации.
Какие IP-диапазоны ей принадлежат.
Имена и почты сотрудников (для оценки риска фишинга).
Какие технологии используются.
Нет ли утёкших паролей и ключей в публичных репозиториях.
Whois и DNS
whois example.com # кто владелец домена, когда создан

dig example.com ANY # все DNS-записи

dig example.com MX # почтовые серверы

dig example.com TXT # текстовые записи (часто SPF, проверки)

nslookup example.com # простой аналог
Записи DNS, которые важны:
Тип Что означает Чем полезно
A Домен → IPv4 Где физически сайт
AAAA Домен → IPv6 То же для IPv6
MX Почтовые серверы Инфраструктура почты
NS DNS-серверы домена Кто управляет зоной
Тип Что означает Чем полезно
TXT Произвольный текст SPF, DKIM, верификации сервисов
CNAME Псевдоним Связи с внешними сервисами
Поиск поддоменов
У крупной организации десятки поддоменов, и часть из них забыта и уязвима. Их ищут:
пассивно, через публичные базы сертификатов и источники

amass enum -passive -d example.com

subfinder -d example.com
онлайн без установки: crt.sh (прозрачность сертификатов)

в браузере: https://crt.sh/?q=%25.example.com
ЗАМЕТКА

Сертификаты TLS публично логируются (Certificate Transparency). Через crt.sh видно все поддомены, для которых когда-либо выпускался сертификат. Это полностью пассивный и очень богатый источник.

Google Dorking
Поисковые операторы позволяют находить то, что случайно попало в индекс:
site:example.com все страницы домена в индексе

site:example.com filetype:pdf документы

site:example.com inurl:admin страницы со словом admin в адресе

site:example.com intitle:index.of открытые листинги папок

"example.com" ext:log логи, попавшие в индекс

Это легально — вы используете обычный поиск. Но найденное может оказаться случайно опубликованной чувствительной информацией. В рамках разрешённого проекта это фиксируется как находка «утечка информации».

Поиск секретов в открытом коде

Разработчики иногда случайно публикуют на GitHub ключи, пароли, токены. Для разрешённого аудита это проверяют так:

на github через веб-поиск по организации:

org:target-company password

org:target-company api_key

инструменты для своих/разрешённых репозиториев:

trufflehog github --org=target-company

gitleaks detect --source=.

ВНИМАНИЕ

Найдя чужой действующий ключ или пароль, не используйте его. Зафиксируйте факт и место, сообщите владельцу. Использование — это уже несанкционированный доступ со всеми последствиями из главы 1.

Определение технологий сайта

whatweb https://example.com # сервер, CMS, фреймворки

curl -I https://example.com # заголовки: Server, X-Powered-By

расширение браузера Wappalyzer показывает стек визуально

Зная, что сайт на конкретной версии конкретной CMS, вы сможете осмысленно проверить, нет ли для неё известных публичных уязвимостей (по базе CVE).

Как это складывается в картину

Представьте: вы начали с одного домена example.com. Через crt.sh нашли dev.example.com и old.example.com. Whatweb показал, что old. работает на древней версии CMS. Google выдал забытый PDF с внутренними адресами. Вы ещё ни разу не «атаковали», но уже знаете, где самое слабое место. Это и есть сила разведки.

Практика к главе 7

1. Для разрешённой учебной цели (или собственного домена) соберите: whois, все DNS-записи, список поддоменов через crt.sh.

2. Через Google dorks найдите все проиндексированные PDF на site:gov.ru filetype:pdf (публичные документы — тренируемся на открытом).

3. Определите технологии трёх публичных сайтов через Wappalyzer или whatweb.

4. Составьте «карту цели»: домены, поддомены, технологии, почты — в одном файле. Это заготовка будущего отчёта.

Частые ошибки новичка

Сразу бежать сканировать, пропустив пассивную разведку.
Не записывать находки — потом невозможно вспомнить, откуда что взялось.
Путать «нашёл в открытом доступе» с «можно использовать».
Сканировать активно то, что можно было узнать пассивно, и этим шуметь зря.

Глава 8. Сканирование портов и сервисов с Nmap

Nmap — главный инструмент активной разведки. Если разведка отвечала на вопрос «что это за организация», то сканирование отвечает на «какие машины живы, какие двери открыты и что за ними стоит».

Идея простыми словами

Представьте дом. Сканирование хостов — это обход квартала с вопросом «кто дома?». Сканирование портов — проверка, какие двери (порты) не заперты. Определение сервисов — выяснение, что за каждой дверью: прихожая, кухня, сейфовая комната.

ВНИМАНИЕ

Сканирование — это уже активное взаимодействие с целью. Запускать его можно только по своей лаборатории или по целям в явном scope. Агрессивное сканирование способно вывести слабый сервис из строя, поэтому по разрешённым продакшен-целям начинают мягко и в один поток.

Этап 1. Кто жив (host discovery)

nmap -sn 192.168.56.0/24
-sn — только проверка «живости», без сканирования портов;
192.168.56.0/24 — вся подсеть.
Вы получите список активных адресов. С него начинается работа в сети.

Этап 2. Какие порты открыты

nmap 192.168.56.101
По умолчанию проверяются 1000 самых частых портов. Пример вывода:
PORT STATE SERVICE

22/tcp open ssh

80/tcp open http

139/tcp open netbios-ssn

445/tcp open microsoft-ds

3306/tcp open mysql
Состояния портов:
STATE Значение
open Порт слушает, сервис доступен
closed Порт отвечает, но сервиса нет
filtered Ответа нет — обычно фаервол
Полезные режимы
nmap -p- 192.168.56.101 # все 65535 портов (дольше, полнее) nmap -p 80,443,8080 цель # конкретные порты nmap -F цель # быстро, 100 портов nmap --top-ports 20 цель # 20 самых популярных

Этап 3. Что за сервисы и версии — самое ценное

nmap -sV 192.168.56.101
-sV определяет версии программ:
PORT STATE SERVICE VERSION

22/tcp open ssh OpenSSH 7.2p2 Ubuntu

80/tcp open http Apache httpd 2.4.18

3306/tcp open mysql MySQL 5.7.33

Версия — это ключ. Зная Apache 2.4.18, вы можете проверить по базе CVE, нет ли у этой конкретной версии известных уязвимостей. Пентест во многом — это сопоставление найденных версий с публичными базами уязвимостей.

Этап 4. ОС и скрипты

nmap -O цель # определить операционную систему

nmap -sC цель # стандартные безопасные скрипты

nmap -A цель # всё сразу: -sV -O -sC + трассировка

-sC запускает набор стандартных скриптов NSE (Nmap Scripting Engine): они аккуратно собирают дополнительную информацию — заголовки HTTP, доступные SMB-шары, сертификаты.

Рекомендуемая связка для новичка

Две команды, закрывающие большинство задач в учебной сети:
1. Быстрый обзор: что вообще открыто

nmap -p- --min-rate 1000 192.168.56.101 -oN ports.txt
2. Глубоко по найденным портам

nmap -sV -sC -p 22,80,3306 192.168.56.101 -oN detail.txt
-oN файл — сохранить результат в читаемый файл. Всегда сохраняйте вывод — он нужен для отчёта.
Форматы сохранения
-oN file.txt обычный, для чтения -oG file.gnmap «grepable», удобно фильтровать grep-ом -oX file.xml XML, для импорта в другие инструменты -oA basename сразу все форматы

Чтение результата: пример рассуждения

Вы просканировали учебную мишень и видите: открыт 21/tcp (FTP, vsftpd 2.3.4), 80/tcp (Apache), 3306/tcp (MySQL, доступен снаружи). О чём это говорит:

1. FTP старой версии — повод проверить по базе известных уязвимостей для неё.

2. MySQL, открытый наружу, — почти всегда ошибка конфигурации: база не должна смотреть в сеть.

3. Веб-сервер — повод переходить к веб-анализу (главы 9–13).

Вы ещё ничего не сломали, но уже составили план проверки и список вероятных проблем для отчёта.

Масштаб и скорость

Для больших диапазонов nmap медленноват. Тогда применяют быстрый masscan для первичного поиска открытых портов, а затем точечно nmap -sV по найденному. Но для домашней лаборатории и большинства учебных задач хватает одного nmap.

Практика к главе 8

1. В лаборатории найдите все живые хосты своей подсети (-sn).

2. Полностью просканируйте Metasploitable (-p-), сохраните в файл.

3. По открытым портам определите версии сервисов (-sV -sC).

4. Выберите один сервис и его версию, найдите в интернете, какие уязвимости для неё известны (сайт CVE или поиск по версии). Только изучение, без эксплуатации на этом этапе.

5. Оформите результаты как мини-отчёт: цель, открытые порты, версии, предварительные наблюдения.

Частые ошибки новичка

Сканировать не ту подсеть (перепроверьте ip а).
Запускать -A по продакшену в баг-баунти — это шумно и часто запрещено правилами.
Не сохранять вывод и терять результаты.
Останавливаться на списке портов, не определяя версии. Версии — это и есть ценность.
Доверять filtered как «ничего нет»: там может быть сервис за фаерволом.

Углублённо: от одной команды к методике

Тюнинг скорости и «вежливости»

Nmap имеет шаблоны таймингов -T0..-T5. -T4 — разумно быстро для лаборатории; -T1/-T2 — медленно и незаметно для чувствительных целей; -T5 — агрессивно, может ронять слабые сервисы.

nmap -sV -T4 цель # быстро, для лаборатории

nmap -sV -T2 --max-retries 2 цель # мягко, для хрупкой цели

NSE-скрипты: больше, чем версия

Движок скриптов NSE расширяет nmap. Категории: safe, default, discovery, vuln и др.

nmap --script=default цель # стандартный безопасный набор

nmap --script=http-title,http-headers цель

nmap --script=vuln цель # проверка известных уязвимостей (шумно!)

nmap --script=smb-os-discovery цель # сбор данных по SMB

ВНИМАНИЕ

Категория vuln активно проверяет уязвимости и может быть расценена как атака и даже навредить. В баг-баунти её применение по продакшену обычно запрещено. В лаборатории — пожалуйста.

Читаем «фильтрованные» порты

filtered означает «ответа нет» — почти всегда фаервол. Приёмы прояснить картину: другой тип скана (-sA для анализа правил фаервола), другой порт-источник, UDP-скан (-sU) для сервисов на UDP (DNS, SNMP), которые TCP-скан пропускает.

Связка инструментов на практике

1. masscan — быстро по огромному диапазону найти открытые порты

2. nmap -sV -sC — точечно по найденным портам узнать сервисы

3. по веб-портам → переход к главам 9–14 (Burp, ffuf)

4. по остальным → определить версии и свериться с базами уязвимостей

Чеклист интерпретации результата

[ ] Правильная ли подсеть? (сверка с ip a)

[ ] Сохранён ли вывод в файл?

[ ] Нет ли сервисов, которые не должны смотреть в сеть (БД, админки)?

[ ] Есть ли устаревшие версии для проверки по CVE?

[ ] Отмечены ли веб-порты для отдельного веб-анализа?

Глава 9. Как устроено веб-приложение и где оно ломается

Веб — самая доступная для новичка область и самая востребованная. Почти всё, что вы будете находить в баг-баунти первые год-два, — это веб. Поэтому разберём устройство веб-приложения подробно.

Три слоя любого сайта

[БРАУЗЕР] ——HTTP——► [СЕРВЕР ПРИЛОЖЕНИЯ] ——► [БАЗА ДАННЫХ]

клиент, здесь логика: хранит

что видит проверки, обработка данные

пользователь запросов

Клиент (frontend) — HTML, CSS, JavaScript в браузере. Пользователь всё это видит и может менять. Поэтому любая проверка только на клиенте — не защита.

Сервер (backend) — PHP, Python, Node.js, Java. Здесь настоящая логика. Сюда приходят данные от пользователя, и здесь их должны правильно проверять.

База данных — MySQL, PostgreSQL, MongoDB. Хранит пользователей, заказы, пароли.

Главный принцип всех веб-уязвимостей

Вернёмся к мысли из главы 2: уязвимость возникает там, где данные пользователя воспринимаются как команда или код. Почти вся эта часть книги — вариации одной темы:

Данные пользователя попали в SQL-запрос → SQL-инъекция.

Данные пользователя попали в HTML-страницу → XSS.

Данные пользователя попали в системную команду → command injection.

Данные пользователя попали в путь к файлу → path traversal.

Данные пользователя указали, какой ресур показать, без проверки прав → IDOR.

Если вы запомните этот принцип, вы будете понимать не отдельные трюки, а саму причину. А значит, сможете находить и новые уязвимости.

Доверенная и недоверенная сторона

Всё, что приходит от пользователя, — недоверенные данные: параметры URL, тело POST, заголовки, cookie, загружаемые файлы, даже User-Agent. Ошибка проектирования почти всегда звучит так: «разработчик поверил данным пользователя».

Доверять нельзя: URL-параметры, формы, cookie, заголовки, JSON, имена файлов, скрытые поля формы, значения из JS. Можно доверять: только тому, что сервер сам проверил и очистил.

OWASP Top 10 — карта того, что искать

OWASP Top 10 — это список самых критичных категорий веб-уязвимостей, обновляемый сообществом. Его знать обязательно. Версия 2021 (актуальная база на 2026):

Категория Простыми словами

A01 Broken Access Control Можно получить чужое/запретное

(IDOR, обход прав)

A02 Cryptographic Failures Данные плохо или не зашифрованы

A03 Injection Инъекции: SQL, команды, XSS

A04 Insecure Design Небезопасно спроектировано в

принципе

A05 Security Misconfiguration Небезопасные настройки, дефолты,

лишнее включено

A06 Vulnerable Components Старые уязвимые библиотеки и

компоненты

A07 Identification/Auth Failures Слабая аутентификация, сессии

A08 Software/Data Integrity Нет проверки целостности

обновлений/данных

A09 Logging/Monitoring Failures Атаку не видно, потому что ничего не

логируется

A10 SSRF Сервер заставляют делать запросы за

атакующего

В этой части книги мы разберём самые важные и частые на практике: Access Control / IDOR, Injection (SQL и XSS), а также затронем остальные.

Инструменты для веб-работы

Браузер + DevTools (F12) — бесплатно, обязательно.
Burp Suite Community — прокси, сердце веб-пентеста (глава 10).
OWASP ZAP — бесплатная альтернатива Burp с автосканером.
ffuf / gobuster — перебор скрытых путей.

Методика: как подходить к незнакомому сайту

1. Осмотреться как пользователь. Пройдите все функции: регистрация, вход,
данных.

2. Составить карту точек ввода. Каждый параметр, каждая форма, каждый заголовок.

3. Понять роли. Есть ли обычный пользователь и админ? Что видит каждый? Здесь живут ошибки контроля доступа.

4. Проверять по одной идее за раз. Подставили спецсимвол → посмотрели реакцию → сделали вывод.

5. Фиксировать всё. Запрос, ответ, вывод. Это будущий отчёт.

Практика к главе 9

1. Откройте Juice Shop. Пройдите его как обычный покупатель и выпишите все функции, где вы вводите данные.

2. Для каждой функции запишите, какие параметры уходят на сервер (смотрите во вкладке Network).

3. Прочитайте официальную страницу OWASP Top 10 2021 и своими словами опишите каждую категорию одной фразой.

4. Для трёх знакомых вам сайтов предположите, к каким категориям Top 10 они потенциально уязвимее всего и почему.

Частые ошибки новичка

Искать «уязвимость вообще», не понимая функций приложения. Сначала карта, потом поиск.

Проверять защиту только на клиенте и радоваться «обходу», который ничего не меняет на сервере.

Пытаться применить сразу все техники к одному полю. По одной идее за раз.

Глава 10. Burp Suite: прокси между вами и сайтом

Burp Suite — главный инструмент веб-пентестера. Суть проста: он садится посередине между вашим браузером и сайтом и показывает каждый запрос, позволяя его рассмотреть и изменить до отправки. Браузер скрывает «кухню» HTTP — Burp её открывает.

Зачем это нужно

Браузер не даст вам легко изменить скрытое поле формы, подставить другой id, убрать клиентскую проверку. Burp даёт. Вы видите сырой HTTP-запрос и можете поменять в нём что угодно перед отправкой на сервер. А именно сервер — то место, где живут настоящие уязвимости.

Установка и первый запуск

Burp Suite Community Edition бесплатен и уже установлен в Kali (меню → Web Application Analysis). Для других систем — скачайте с portswigger.net.

Настройка браузера на Burp

Проще всего использовать встроенный браузер Burp: вкладка Proxy → Intercept → Open Browser. Он уже настроен на прокси и с сертификатом. Ничего настраивать не нужно.

Если хотите свой браузер: установите прокси 127.0.0.1:8080 и импортируйте сертификат Burp (http://burp → CA Certificate), иначе HTTPS-сайты будут ругаться.

Главные вкладки

Вкладка Назначение
Proxy Перехват и просмотр трафика между браузером и сайтом
Target Карта сайта: все увиденные страницы и параметры
Repeater Отправить один запрос много раз, меняя его. Самый

используемый
Intruder Автоматический перебор значений (в Community замедлен)
Decoder Кодирование/декодирование (URL, Base64, hex)
Comparer Сравнение двух ответов

Рабочий цикл: Proxy → Repeater

1. Intercept. Во вкладке Proxy включите Intercept. Теперь каждый запрос браузера останавливается и ждёт вашего решения: отпустить (Forward) или выбросить (Drop).

2. Поймайте интересный запрос. Например, открытие своего профиля GET /profile?id=42.

3. Отправьте в Repeater. Правой кнопкой по запросу → Send to Repeater (или Ctrl+R).

4. Экспериментируйте. В Repeater меняйте id=42 на id=43, жмите Send, смотрите ответ. Так проверяется контроль доступа (следующая глава).

ЗАМЕТКА

Intercept удобно держать выключенным, а весь трафик смотреть в Proxy → HTTP history. Там накапливаются все запросы, и любой можно отправить в Repeater задним числом. Постоянно включённый Intercept только мешает сёрфингу.

Пример: разбор запроса в Repeater

Поймали запрос входа:
POST /rest/user/login HTTP/1.1

Host: localhost:3000

Content-Type: application/json

Content-Length: 44
{"email":"test@test.com","password":"test123"}
В Repeater вы можете:
менять email и пароль и смотреть разные ответы (200 против 401);
подставить спецсимволы в email и искать признаки инъекции (глава 11);
убрать или изменить заголовки и смотреть реакцию.
Каждое нажатие Send — отдельный эксперимент с мгновенным ответом сервера рядом. Это и делает Repeater незаменимым.

HTTP history как журнал

Вкладка Proxy → HTTP history — это полный лог всего, что прошло через Burp. Фильтры помогают отсечь шум (картинки, статику) и оставить только запросы к API. Привыкайте просматривать историю: часто интересный запрос замечаешь не сразу.

Decoder — работа с кодировками

Данные в вебе часто закодированы. Decoder переводит туда-обратно:
Текст: admin:password

Base64: YWRtaW46cGFzc3dvcmQ=
Это нужно постоянно: cookie сессии, токены, параметры нередко в Base64 или URL-кодировке, и их надо уметь читать.

OWASP ZAP — бесплатная альтернатива

Если Burp покажется тяжёлым, попробуйте OWASP ZAP: полностью бесплатный, с автоматическим сканером (которого нет в Burp Community). Логика та же — прокси посередине. Многие новички начинают с ZAP, а потом переходят на Burp.

Практика к главе 10

1. Запустите Burp, откройте встроенный браузер, зайдите в Juice Shop. Убедитесь, что запросы появляются в HTTP history.

2. Залогиньтесь в Juice Shop и найдите в истории запрос login. Отправьте его в Repeater.

3. В Repeater измените пароль на неверный и сравните код ответа с верным входом.

4. Найдите запрос, который отдаёт ваш профиль или корзину. Запишите все параметры в нём.

5. Через Decoder раскодируйте любой Base64-токен, который найдёте в запросах.

Частые ошибки новичка

Держать Intercept всегда включённым и бесконечно жать Forward. Выключите его и работайте через историю.

Забыть импортировать сертификат и решить, что «HTTPS-сайты не работают».

Не пользоваться Repeater и менять запросы в браузере вручную.

Не сохранять интересные запросы. Комментируйте и помечайте их цветом.

Глава 11. Инъекции: SQL-инъекция на понятном примере

Инъекция — самая классическая и до сих пор одна из самых опасных уязвимостей. Разберём её на SQL, потому что это нагляднее всего. Цель главы — чтобы вы поняли механизм, умели его обнаружить на разрешённой цели и, главное, знали, как его чинят.

Откуда берётся SQL-инъекция

База данных понимает язык запросов SQL. Приложение формирует запросы из данных пользователя. Если оно склеивает запрос простой конкатенацией строк, данные пользователя превращаются в часть команды.

Уязвимый код (так писать нельзя):

НЕПРАВИЛЬНО: пользовательский ввод вклеен прямо в запрос

query = "SELECT * FROM users WHERE name = '" + user_input + "'"

Если пользователь ввёл обычное имя Ivan, получится нормальный запрос:

SELECT * FROM users WHERE name = 'Ivan'

Но пользователь контролирует user_input. Если он введёт значение, содержащее кавычку, структура запроса сломается — данные «вытекут» в код команды. Это ровно тот принцип из главы 9: данные стали командой.

Как это обнаруживают (на разрешённой цели)

Первый и самый безобидный тест — подставить одиночную кавычку в параметр и посмотреть на реакцию:

http://site/product?id=5'

Три типичных исхода:

Реакция Что означает

Ошибка SQL или код 500 Ввод влияет на запрос — вероятна инъекция

Страница «сломалась» иначе, Стоит копать глубже

чем от неверного id

Ничего не изменилось Либо фильтруется, либо параметр не в запросе

Появление ошибки вида You have an error in your SQL syntax — классический индикатор. Это и есть «обнаружение»: вы показали, что ваш символ долетел до

Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, купив полную легальную версию на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.