

18+

ЗАРАБОТОК ДЛЯ ХАКЕРА

Полностью
переработанное
издание
2025–2026

Михаил Тарасов

Михаил Тарасов

**Заработок для хакера.
Полностью переработанное
издание 2025—2026**

«Издательские решения»

Тарасов М.

Заработок для хакера. Полностью переработанное издание 2025—2026 / М. Тарасов — «Издательские решения»,

«Заработок для хакера» — это RedTeam-мануал по легальной монетизации взлома в 2026 году. Bug Bounty, фриланс, найм от джуна до Lead, ИП и налоги, NDA, AI-атаки и квантовая криптография — без инфоцыганщины, с реальными командами и payload'ами. 388 страниц жесткого практикума: как превратить свой скилл в белый, стабильный и очень жирный доход.

© Тарасов М.

© Издательские решения

Содержание

Введение	6
— Зачем хакерам деньги и почему рынок кибербезопасности взорвался	6
— Как читать эту книгу: карты монетизации и дорожные карты	9
— Честный разговор: что реально, что миф	12
Часть I: Фундамент — кто ты и сколько стоишь	15
Глава 1. Портрет современного хакера-профессионала	15
— Спектр от Script Kiddie до Senior Researcher: где деньги?	15
— Специализации с самым высоким чеком в 2025—2026	18
— Как рынок оценивает твои навыки (грейды, сертификаты, портфолио)	20
Глава 2. Стек навыков = стек доходов	24
— Языки, которые реально нужны: Python, Rust, Go, JavaScript, Bash	24
— Операционные системы и лабораторная база: Твой хакерский бункер	27
— Soft skills хакера: почему «объяснить клиенту» стоит дороже, чем «сломать»	29
Часть II: Bug Bounty — охота за головами	33
Глава 3. Bug Bounty с нуля до первого чека	33
— Как устроены программы: scope, правила, выплаты	33
— Платформы 2026: HackerOne, Bugcrowd, Intigriti, Standoff 365, BI. ZONE Bug Bounty	36
— Методология поиска: разведка, перебор, эксплуатация, репорт	38
Глава 4. Прокачка репутации на Bug Bounty	42
— Как писать репорты, чтобы не получить N/A	42
— От \$100 до \$50k: какие баги реально платят	45
— Работа с приватными программами и VIP-инвайтами	47
— Кейсы: реальные выплаты за SSRF, XXE, Auth Bypass в 2024—2025	50
Конец ознакомительного фрагмента.	51

Заработок для хакера. Полностью переработанное издание 2025—2026

Михаил Тарасов

© Михаил Тарасов, 2026

ISBN 978-5-0071-3219-0

Создано в интеллектуальной издательской системе Ridero

Введение

— Зачем хакерам деньги и почему рынок кибербезопасности взорвался

Давай начистоту. Если ты открыл эту книгу, значит, романтика бессонных ночей за консолью ради одного лишь чувства превосходства начала понемногу выветриваться. Ты умеешь находить скули (SQLi), крутить XSS, собирать цепочки уязвимостей и, возможно, даже писать свои простенькие эксплойты. Твои глаза привыкли к темной теме, а пальцы помнят синтаксис Nmap лучше, чем дни рождения родственников. Это круто. Это база. Но в какой-то момент возникает закономерный вопрос: а где, собственно, профит?

В хакерском сообществе долгое время существовало негласное, почти сектантское правило: тру-хакеры работают за идею. За фан, за уважение на андеграундных бордах, за красивый дефейс или строчку в зале славы (Hall of Fame). Деньги считались чем-то грязным, уделом «корпоративных пиджаков», которые продали душу за стабильный оклад и соцпакет. «Информация должна быть свободной» — кричали манифесты 90-х и нулевых.

Забудь об этом. Мы в 2026 году.

Сегодня информация — это не просто свободные байты, летающие по оптоволокну. Это нефть, золото и уран XXI века в одном флаконе. И если ты умеешь добывать эту информацию, защищать её или показывать другим, как легко её можно украсть — ты обладаешь навыком, за который рынок готов платить огромные, иногда неприличные деньги.

Зачем хакерам деньги? Вопрос звучит почти философски, но ответ предельно прагматичен.

Во-первых, инфраструктура стоит дорого. Хорошее железо для брутфорса хешей или поднятия десятка виртуалок для тестов, платные подписки на OSINT-сервисы (тот же Shodan или Maltego), аренда серверов под собственные C2-инфраструктуры (Command and Control) для Red Team операций — всё это требует постоянных вливаний кэша. Ты не сможешь конкурировать с топовыми багхантерами, если твой гесоп-пайплайн крутится на старом ноутбуке с 8 гигами оперативки, который задыхается при запуске Burp Suite. Профессиональный инструмент требует профессионального бюджета.

Во-вторых, время. Самый ценный невосполнимый ресурс. Если ты вынужден работать курьером, эникеем или переключать JSON-ы на скучной галере по 8 часов в день, чтобы оплатить счета, у тебя физически не останется сил на качественный ресерч. Чтобы находить 0day (уязвимости нулевого дня) или раскапывать сложную бизнес-логику в банковских приложениях, нужен фокус. А фокус появляется только тогда, когда у тебя закрыт базовый уровень по пирамиде Маслоу. Финансовая независимость — это свобода выбирать, что ломать сегодня: скучный энтерпрайз за большой чек или интересный open-source проект просто для души.

В-третьих, легализация статуса. Деньги, заработанные легальным (или хотя бы серым) путем, позволяют выйти из тени. Тебе больше не нужно трястись при каждом стуке в дверь, ожидая маски-шоу и изъятия техники по статье 272 УК РФ. Легальный заработок открывает двери к нормальной жизни: покупке недвижимости, путешествиям, созданию семьи. Да, это звучит скучно для 18-летнего скрипт-кидди, но поверь, к 25—30 годам эти вещи становятся критически важными.

Теперь давай разберемся с главным феноменом последних лет: почему рынок кибербезопасности не просто вырос, а буквально взорвался?

Если отмотать время на 10—15 лет назад, ИБ (информационная безопасность) в большинстве компаний финансировалась по остаточному принципу. Это была статья расходов, которую резали в первую очередь. «Зачем нам пентест? У нас и так стоит антивирус и фаервол!» — так рассуждал типичный топ-менеджмент. Хакеров воспринимали как мифическую угрозу из голливудских фильмов.

Всё изменилось, когда мир тотально переехал в цифру, а атаки стали не просто хулиганством, а высокомаржинальным бизнесом. Эпоха шифровальщиков (ransomware) перевернула игру. Когда корпорации уровня Colonial Pipeline, Garmin или JBS останавливают работу на недели из-за того, что кто-то кликнул не на ту ссылку, а убытки исчисляются десятками миллионов долларов — до бизнеса наконец-то доходит: дешевле заплатить миллион превентивно «белым» хакерам, чем отдать десять миллионов выкупа «чёрным» плюс потерять репутацию и получить гигантские штрафы от регуляторов.

К 2025—2026 годам сошлись сразу несколько глобальных трендов, которые превратили кибербез в золотую жилу:

Тотальная цифровизация и рост Attack Surface (поверхности атаки). Облака, микросервисы, IoT-устройства, удаленка, интеграции по API. Кода стало так много, что контролировать его безопасность силами внутренних команд физически невозможно. Разработчики пишут фиичи быстро, безопасность всегда отстает. Этот зазор — и есть твоя зона профита. Каждый новый релиз корпоративного софта генерирует новые баги.

Геополитика и кибервойны. Государства (APT-группировки) стали открыто использовать кибероружие для шпионажа и саботажа. Это заставило крупный бизнес (банки, телеком, энергетику, госсектор) вливать миллиарды в защиту и нанимать Red Team команды для симуляции реальных атак. Если раньше пентест был формальностью для галочки, то сейчас заказчикам нужна реальная проверка боем: смогут ли их сломать целеустремленные профи?

Бум ИИ (искусственного интеллекта) и LLM (больших языковых моделей). Интеграция AI во все подряд создала совершенно новый класс уязвимостей — от prompt injection до отравления обучающих выборок. Никто пока толком не знает, как надежно защищать эти системы, а значит, исследователи, которые разберутся в этом первыми, сорвут джекпот. С другой стороны, AI дал хакерам инструменты для автоматизации разведки и написания социальной инженерии, что сделало атаки дешевле и массовее. Защитникам приходится отвечать тем же, покупая дорогие AI-решения и нанимая людей, способных ими управлять.

Давление регуляторов. Законы о защите персональных данных (GDPR в Европе, уже сточения ФЗ-152 и введение оборотных штрафов за утечки в России) сделали утечки не просто неприятными, а фатальными для бизнеса. Штрафы стали настолько огромными, что выгоднее нанять штат безопасников и запустить программу Bug Bounty, чем однажды утром увидеть базу своих клиентов в свободном доступе в Telegram.

Кадровый голод. По данным различных аналитических агентств, в мире не хватает миллионов специалистов по кибербезопасности. В России эта проблема усугубилась оттоком кадров и импортозамещением — нужно срочно переписывать и тестировать горы отечественного софта. Когда спрос многократно превышает предложение, цена на твои услуги летит в космос.

Что мы имеем в сухом остатке? Компании боятся. У них есть бюджеты. И они готовы платить тем, кто докажет, что может найти дыру раньше, чем это сделает кто-то с плохими намерениями.

Рынок Bug Bounty (поиск уязвимостей за вознаграждение) из маргинальной тусовки превратился в индустрию, где топовые хантеры зарабатывают миллионы долларов в год, покупают Porsche и выступают на международных конференциях. Фриланс-платформы забиты заказами на аудит смарт-контрактов, OSINT-расследования и настройку защищенных архитектур. Зар-

платы Senior-пентестеров или AppSec-инженеров в продуктовых IT-компаниях давно перешагнули планку, о которой обычные разработчики могут только мечтать.

Эта книга — твой навигатор по этому перегретому рынку. Я не буду учить тебя, как писать эксплойты под нулевые кольца (Ring 0) или обходить последние версии EDR (Endpoint Detection and Response) — для этого есть технические мануалы и CVE-репорты. Я покажу тебе схемы монетизации. Мы разберем, как конвертировать твои технические скиллы в звонкую монету, оставаясь при этом на свободе и не продавая душу дьяволу.

Будет жестко, прагматично и без иллюзий. Готов? Тогда закрывай YouTube, открывай редактор и погнали читать. Твои знания стоят дорого. Пора заставить их работать на тебя.

Практическое задание

Составь личный «хакерский профиль»:

Что ты реально умеешь делать руками прямо сейчас (без Гугла и подсказок ChatGPT)?

Сколько ты хочешь зарабатывать на этих навыках через 6 и 12 месяцев (назови конкретную сумму)?

Какой формат тебе ближе: стабильный найм, охота за баунти (Bug Bounty), фриланс или свой продукт? Запиши это. Это твой baseline — отправная точка, к которой мы вернемся в конце книги.

— Как читать эту книгу: карты монетизации и дорожные карты

Большинство технических книг по кибербезопасности страдают одной и той же болезнью: они написаны либо слишком академично — как диссертация, которую читают трое и засыпают на второй странице, — либо слишком поверхностно — как рекламный буклет курсов, где «ты за три месяца станешь хакером и будешь зарабатывать миллион». Эта книга — не то и не другое.

Перед тобой рабочий инструмент. Как хороший Burp Suite: с виду немного страшновато, зато когда разберёшься — без него уже не представляешь, как вообще раньше жил. Чтобы получить из неё максимум, важно понять логику устройства и как ею пользоваться. Уделишь пять минут этому разделу — сэкономишь часы хаотичного перелистывания.

Общая архитектура книги

Книга разделена на восемь частей, каждая из которых — самостоятельный блок монетизации. Ты можешь читать её линейно, с первой страницы до последней — тогда ты получишь полную картину рынка, начиная с азов и заканчивая стратегическим планированием карьеры. Но можешь и прыгать сразу в нужный раздел, если понимаешь, чего хочешь прямо сейчас.

Вот как это работает:

— Если ты **полный новичок**, который только-только решил, что хочет зарабатывать в ИБ, но ещё не понимает, с чего начать — читай строго по порядку. Части I и II дадут тебе фундамент и первые быстрые деньги через Bug Bounty.

— Если ты уже **умеешь делать что-то конкретное** (например, находить XSS и SQLi), но не понимаешь, как это монетизировать — листай сразу в Части II—IV. Там про Bug Bounty, найм и фриланс.

— Если ты **опытный спец**, которому нужен следующий уровень дохода через контент, продукты или консалтинг — твои главы в Частях V и VI.

— Если тебе нужна **юридическая и финансовая база** (как оформить ИП, как работать с клиентами официально, как не схлопотать 272-ю статью) — Часть VII всегда ждёт тебя, причём желательно прочитать её до того, как ты подпишешь первый договор с клиентом, а не после.

Что такое карта монетизации и зачем она тебе

В конце книги, в Приложении А, ты найдёшь карты монетизации по трём уровням: Junior, Middle и Senior. Это не мотивационные постеры для рабочего стола. Это жёсткие схемы: какой навык → какой формат заработка → сколько реально платят на российском рынке в 2026 году → что нужно докрутить, чтобы перейти на следующий уровень.

Карта монетизации — это твой главный артефакт из этой книги. Объясню на примере:

Предположим, ты умеешь делать ручной веб-пентест на уровне Middle. На карте ты увидишь, что прямо сейчас тебе доступны четыре вектора дохода одновременно: публичные Bug Bounty программы на Standoff 365 и BI. ZONE (выплаты от 10 000 до 500 000 рублей за критический баг), фриланс-аудиты для МСБ (малый и средний бизнес) по 80 000—150 000 рублей за проект, найм в пентест-компанию на 150 000—250 000 рублей в месяц, а также ведение Telegram-канала с write-up'ами как фундамент личного бренда.

Ты не обязан делать всё сразу. Карта показывает, что тебе уже доступно и куда двигаться. Это снимает самую болезненную проблему большинства специалистов — паралич выбора.

Что такое дорожная карта и чем она отличается от карты монетизации

Если карта монетизации отвечает на вопрос «за что мне платят», то дорожная карта отвечает на вопрос «как мне добраться до следующего уровня оплаты».

В Главе 22 собраны четыре детальные дорожные карты под разные стартовые точки:

1. Web Hacker → Bug Bounty Hunter → Senior Vulnerability Researcher Для тех, кто хочет охотиться за уязвимостями независимо, без начальника над головой и дресс-кода в офисе.

2. Sysadmin / DevOps → Pentester → Red Teamer Для тех, у кого уже есть понимание инфраструктуры и сетей, и кто хочет перейти на «тёмную» (на самом деле серую и очень прибыльную) сторону.

3. Developer → AppSec Engineer → Product Security Lead Для разработчиков, которые хотят остаться в коде, но начать зарабатывать существенно больше за счёт фокуса на безопасности приложений.

4. Студент или джун без опыта → первый оффер за 12 месяцев Самый честный и самый жёсткий трек. Без прикрас расписано, что реально можно успеть за год, если тратить на прокачку от 10 часов в неделю.

Каждая дорожная карта — это не абстрактный список советов в духе «учи больше, работай усерднее». Это конкретный список: что изучить (с названиями ресурсов и курсов), какие лабы прокрутить (HackTheBox, TryHackMe, Standoff Cybersecurity Arena), какие сертификаты взять и в какой последовательности, какие первые проекты добавить в портфолио, какие конференции посетить и в каких CTF поучаствовать для нетворкинга.

Как устроены практические задания

В конце каждой главы — блок **«Практическое задание»**. Это не факультативная история «если хочешь — сделай». Это обязательная часть книги, потому что рынок кибербезопасности не покупает знания — он покупает навыки и результаты. Разница между человеком, который прочитал книгу, и человеком, который прочитал книгу и выполнил все задания, — это разница между тем, кто рассуждает о пентесте в теории, и тем, кто показывает клиенту реальный отчёт с реальными находками.

Практические задания выстроены по принципу нарастающей сложности и связанности: задание из Главы 3 создаёт фундамент для задания из Главы 5, а задание из Главы 10 использует артефакты, которые ты создал в Главе 6. К концу книги у тебя будет не просто набор выполненных упражнений, а полноценное портфолио: написанные репорты, опубликованные write-up'ы, первый инструмент на GitHub, базовый шаблон коммерческого предложения и задокументированные результаты лабораторных работ.

Это — твоё портфолио. В современном российском ИБ-рынке, где работодатели и заказчики уже пересытились красивыми резюме, именно портфолио открывает двери.

Что ты НЕ найдёшь в этой книге

Честность с читателем — обязательный элемент хорошей книги. Поэтому скажу прямо, чего здесь нет:

Здесь нет инструкций по незаконным действиям. Не потому что автор наивен или испугался. А потому что нелегальный заработок в ИБ — это худший ROI (возврат на инвестиции), который только можно себе представить. Кратковременная прибыль против реального уголовного дела по статьям 272—274 УК РФ, конфискации оборудования, репутационного краха и закрытого горизонта будущей карьеры — это сделка, на которую соглашаются только те, кто не умеет считать. Рынок легального ИБ в России в 2026 году предлагает настолько жирные возможности, что криминальный путь — просто нерациональный выбор с точки зрения элементарной математики.

Здесь нет магии «станешь богатым за три месяца». Первые деньги в Bug Bounty можно получить быстро. Первый фриланс-проект — через два-три месяца при правильном подходе. Но стабильный высокий доход — это результат системной работы, которая займёт от года до трёх лет в зависимости от твоей стартовой точки. Кто обещает тебе иначе — продаёт курс.

Здесь нет теоретической жижи. Минимум истории и контекста — ровно столько, чтобы понять почему что-то работает. Максимум конкретики: инструменты, платформы, суммы, шаблоны, команды.

Твой первый шаг прямо сейчас

Прежде чем перейти к следующей главе, открой любой текстовый редактор (Notion, Obsidian, простой блокнот — без разницы) и создай файл с названием **«Мой хакерский трек»**. Он станет твоим личным журналом прогресса на протяжении всей книги. В него ты будешь записывать результаты практических заданий, накапливать артефакты портфолио и отслеживать, как меняются твои цели и доходы.

Это не лирика. Это инструмент. Люди, которые записывают цели и промежуточные результаты, достигают их статистически значимо чаще тех, кто всё держит в голове. Твоя голова — это RAM. Заметки — это persistent storage. Не надейся на оперативку там, где нужен жёсткий диск.

Поехали.

Практическое задание

Открой файл «Мой хакерский трек» и заполни четыре блока:

Где я сейчас — честно опиши свой текущий уровень: что умеешь, сколько зарабатываешь на навыках ИБ прямо сейчас (возможно, ноль — это нормально).

Куда я хочу попасть — выбери один из четырёх дорожных треков (из Главы 22) и запиши конкретную должность или формат работы, который хочешь получить.

Мои сроки — поставь дату через 6 месяцев и через 12 месяцев. Напиши, где ты должен быть к каждой из них.

Мои ограничения — сколько часов в неделю ты реально можешь посвящать прокачке? Не идеально, а реально, с учётом работы, семьи и сна.

Сохрани. Не закрывай. Мы вернёмся к этому файлу в финальном задании книги.

— Честный разговор: что реально, что миф

Если ты пришёл в кибербезопасность за деньгами, добро пожаловать в клуб. Здесь их много. Но на пути к первому серьёзному чеку стоит стена из иллюзий, инфоцыганского булшита и устаревших стереотипов десятилетней давности.

Прежде чем мы начнём разбирать конкретные схемы заработка, эксплойты для твоего кошелька и векторы атаки на рынок труда, нам нужно зачистить твою голову от ментального мусора. В ИБ цена ошибки высока: поверив в красивую сказку, ты можешь слить годы на изучение неликвидных технологий или, что хуже, уехать по 272-й статье, искренне веря, что «в даркнете никто не найдёт».

Давай проведём хардкорный аудит самых популярных мифов о заработке хакеров. Разберём, что из этого — откровенный скам, что — полуправда, а как дела обстоят на самом деле в суровой реальности 2026 года.

Миф 1. «Bug Bounty — это лёгкий и быстрый заработок»

Как это продают: «Сидишь на Бали с ноутбуком, попиваешь смузи, запускаешь сканер, находишь XSS в Гугле, получаешь 10 000 долларов и идёшь сёрфить».

Реальность: Bug Bounty — это самый высококонкурентный, выматывающий и непредсказуемый вид легального заработка в ИБ.

Когда ты заходишь на платформу (HackerOne, Bugcrowd или наши Standoff 365 и BI.ZONE), ты соревнуешься не с системой. Ты соревнуешься с тысячами других таких же голодных и умных парней со всего мира. И многие из них написали свою автоматизацию три года назад.

Если ты думаешь, что можно натравить дефолтный профиль Acunetix или Nuclei на публичный score и сорвать куш — забудь. В 99% случаев ты получишь статус «Duplicate» (дубликат — кто-то нашёл это до тебя) или «N/A» (Not Applicable — не является уязвимостью в рамках этой программы).

Реальные деньги в Bug Bounty делают две категории людей:

Короли автоматизации. Те, кто пишет собственные уникальные сканеры, мониторит появление новых поддоменов у корпораций в реальном времени и бьёт туда свежими CVE (уязвимостями) в первые 5 минут после их публикации.

Мастера глубокого бурения (Manual Hackers). Те, кто берёт одно конкретное приложение (например, банковский портал) и неделями ковыряет его бизнес-логику, находя сложные цепочки (IDOR -> SSRF -> RCE), которые ни один автоматический сканер никогда не найдёт.

Bug Bounty — это круто. Это отличная школа. Но рассчитывать на это как на *стабильный* доход в первый год — это как рассчитывать жить на выигрыши в покер, едва выучив комбинации карт.

Миф 2. «Чтобы много зарабатывать, нужно быть гением-программистом»

Как это продают: «Настоящий хакер пишет эксплойты на C++ в блокноте с закрытыми глазами, знает ассемблер наизусть и компилирует ядра Linux по выходным».

Реальность: Для 80% высокооплачиваемых задач в коммерческой безопасности тебе вообще не нужно быть Senior-разработчиком.

Безусловно, уметь читать чужой код (Python, JavaScript, Go, PHP) — это критически важный навык. Написать скрипт для автоматизации рутины — база. Но рынок ИБ в 2026 году настолько огромен, что в нём полно ниш, где платят за *другие* мозги.

Посмотри на социальную инженерию и Red Team. Там платят за психологию, понимание корпоративных процессов и умение обмануть человека, а не машину. Посмотри на OSINT (разведку по открытым источникам) — топовые аналитики зарабатывают сотни тысяч, просто

виртуозно сопоставляя данные из реестров, соцсетей и утечек, не написав за жизнь ни строчки сложного кода. Посмотри на GRC (Governance, Risk, and Compliance) — это бумажная безопасность, аудиты по стандартам. Звучит скучно? Зато чеки за приведение банка в соответствие требованиям ЦБ РФ пробивают потолок здравого смысла.

Индустрия платит за *решение проблемы бизнеса*, а не за то, на каком языке ты умеешь писать циклы.

Миф 3. «Настоящие деньги — только в даркнете»

Как это продают: «Легальные безопасники работают за копейки на дядю, а нормальные пацаны на теневого форумах продают дампы баз и доступы к корпоративным сетям за миллионы в крипте».

Реальность: Даркнет — это болото, где продолжительность «карьеры» стремится к нулю, а уровень паранойи сжирает психику быстрее, чем кокаин.

Давай считать ROI (возврат инвестиций). В 2026 году российский рынок ИБ-услуг легально пухнет от бюджетов. За хороший легальный пентест корпоративной сети компания готова заплатить от 1 до 5 миллионов рублей. Ты делаешь это официально, спишь спокойно, кладёшь кейс в публичное портфолио и повышаешь свой рейтинг на следующий заказ.

Теперь чёрный рынок: ты ломаешь ту же сеть, чтобы продать доступ (Initial Access Broker) картелю шифровальщиков. Продаёшь за \$5000. Получаешь крипту, которую ещё нужно отмыть и вывести (потеряв процент). А взамен получаешь:

Риск попасть под разработку Управления «К» МВД или ФСБ (а они работают гораздо лучше, чем принято думать на форумах).

Постоянный риск кидка от «партнёров» по даркнету — там нет арбитража и судов.

Невозможность легализовать и потратить крупные суммы. Купишь Porsche за наличные — к тебе придут с вопросами из налоговой.

Нелегальный хакинг в 2026 году — это удел либо государственно-спонсируемых АPT-групп, либо дураков, не умеющих считать риски. Легальный рынок платит больше, стабильнее и не требует прятать лицо.

Миф 4. «Сертификаты гарантируют работу и высокую зарплату»

Как это продают: «Сдай экзамен на CEN (Certified Ethical Hacker) или OSCP (Offensive Security Certified Professional), и HR-ы выстроятся в очередь у твоего подъезда с чемоданами денег».

Реальность: Сертификат — это просто фильтр для HR, который доказывает, что ты не совсем идиот и способен довести дело до конца.

В реальном мире (особенно в России, где западные вендоры отвалились, а отечественные сертификации только формируются) работодатель смотрит на три вещи в таком порядке:

Реальный опыт и портфолио (твой GitHub, статьи на Хабре, write-up'ы, профили на Bug Bounty платформах).

Как ты мыслишь на техническом собеседовании (сможешь ли ты объяснить цепочку атаки).

Бумажки (дипломы и сертификаты).

Никто не заплатит тебе 300 000 рублей в месяц только за то, что у тебя есть OSCP. Тебе заплатят их, если ты сможешь показать, как твои навыки принесут компании пользу или сэкономят деньги. Сертификат открывает дверь на собеседование, но продавать себя внутри кабинета придётся тебе самому.

Миф 5. «Искусственный интеллект скоро заменит пентестеров»

Как это продают: «LLM-модели (типа ChatGPT или Claude) уже умеют писать код и находить уязвимости. Через пару лет хакеры будут не нужны, ИИ всё автоматизирует».

Реальность: ИИ заменит не хакеров. ИИ заменит хакеров, *которые не используют ИИ*.

Да, базовый сканирующий макак-труд стремительно обесценивается. Если вся твоя работа заключается в том, чтобы запустить Nessus, выгрузить PDF-отчёт и отправить клиенту — у меня для тебя плохие новости, ты уже безработный.

Но ИИ не умеет понимать бизнес-контекст. LLM может найти SQL-инъекцию в куске кода, но она не поймёт, что перевод денег с отрицательным значением баланса на бонусный счёт в рамках конкретной программы лояльности ритейлера — это критическая уязвимость бизнес-логики.

Более того, ИИ сам по себе стал новой гигантской поверхностью для атак (Attack Surface). Кто-то должен тестировать эти модели на prompt injection, отравление данных (data poisoning) и утечку конфиденциальной информации. Спрос на AI Red Teaming сейчас бьёт все рекорды. Тот, кто разберётся в этом первым, будет диктовать свои цены рынку.

Миф 6. «Клиентам нужен взлом. Чем круче взлом — тем больше платят»

Как это продают: «Я получил root-права на контроллере домена за два часа! Заказчик должен быть в экстазе и завалить меня деньгами!»

Реальность: Заказчику плевать на твой root. Заказчику нужен понятный отчёт о том, как закрыть дыру до того, как в неё пролезут плохие парни.

Это самая болезненная иллюзия начинающих пентестеров. Ты продаёшь не взлом. **Ты продаёшь отчёт и рекомендации.**

Если ты разнёс инфраструктуру банка за час, но потом написал невнятный отчёт на три страницы без шагов воспроизведения (Steps to Reproduce) и с рекомендациями уровня «обновите Windows» — ты сработал в минус. Клиент не может передать это админам в работу. В следующий раз наймут не тебя, а зануду, который ломал неделю, но выдал идеальный 50-страничный документ с Executive Summary для директора и чёткими инструкциями для IT-отдела.

Деньги в ИБ платят за снижение рисков, а не за демонстрацию твоего эго.

Подводим итог раздела

Кибербезопасность — это не магия. Это ремесло. Такое же, как инженерия, архитектура или юриспруденция.

Здесь работают те же законы капитализма:

Если ты делаешь то, что умеют делать миллионы школьников по туториалам с YouTube (запустить скрипт) — тебе будут платить копейки.

Если ты решаешь сложные, нестандартные задачи, берёшь на себя ответственность и умеешь переводить технический сленг на язык бизнеса и денег — твой чек не имеет потолка.

Хватит верить в сказки про лёгкие миллионы из даркнета или волшебные кнопки «Взломать всё». Пора относиться к своим навыкам как к бизнес-активу. В следующих главах мы начнём разбирать этот актив на винтики и смотреть, как именно его продавать.

Практическое задание

Инвентаризация иллюзий.

Открой свой файл «Мой хакерский трек» (который ты создал в предыдущем задании).

Напиши заголовок «Мои иллюзии».

Честно выпиши 3—5 мифов или стереотипов об ИБ, в которые ты верил до прочтения этой главы. Возможно, ты думал, что Bug Bounty даст тебе первые 1000\$ через неделю? Или что без знания Ассемблера не стоит лезть в веб-пентест?

Рядом с каждым пунктом напиши, как это знание ограничивало тебя в действиях (например: «Думал, что нужен OSCP -> не откликнулся на вакансии джунов»).

Осознание того, где ты обманывал сам себя — первый шаг к реальным деньгам.

Часть I: Фундамент — кто ты и сколько стоишь

Глава 1. Портрет современного хакера-профессионала

— Спектр от Script Kiddie до Senior Researcher: где деньги?

В IT-индустрии привыкли мерить людей линейкой «Junior — Middle — Senior». Это удобно для HR-ов, которые закупают программистов на галеры. Но в кибербезопасности, особенно в её атакующей части (Offensive Security), эта линейка часто даёт сбой. Можно быть Senior Python Developer'ом, но при этом абсолютным нулём в понимании векторов атак. И наоборот, можно в 19 лет находить критические 0day-уязвимости в корпоративных продуктах, не имея ни дня официального стажа.

Поэтому хакерский рынок правильнее измерять не годами высиживания стула в офисе, а **глубиной понимания систем и способностью создавать ценность (или угрозу, которую нужно предотвратить).**

Давай разберём этот спектр от самого низа до абсолютного топа. Посмотрим, кто чем занимается, сколько за это платят в 2026 году и, главное, *почему* рынок оценивает эти навыки именно так.

Уровень 1: Script Kiddie (Оператор чужого кода)

Исторически это ругательство. Так называли школьников, которые качали готовые эксплойты с Exploit-DB, запускали их по мануалам с YouTube и радовались дефейсам сайтов на WordPress.

Сегодня Script Kiddie эволюционировали. Они больше не ломают сайты ради фана. Они пытаются на этом зарабатывать. Это те самые ребята, которые:

Запускают Nessus/Acunetix/Nuclei с дефолтными профилями на всё подряд.

Сканируют публичные пулы IP-адресов в поисках открытых портов RDP (Remote Desktop Protocol) с простыми паролями (брутфорс).

Пытаются участвовать в Bug Bounty, отправляя сотни мусорных репортов о том, что у сайта нет заголовка X-Frame-Options или открыт листинг директорий без конфиденциальных данных.

Где здесь деньги? А денег здесь, по сути, нет. Вернее, они случайные и мизерные.

На Bug Bounty такие репорты закрываются со статусом N/A (не применимо) или Informative (принято к сведению, без выплаты). В лучшем случае, если программа совсем щедрая, можно получить \$50 или 5 000 рублей на Standoff 365 за какую-нибудь мелочь.

В найм таких не берут даже на позицию Junior. Кому нужен сотрудник, который умеет только нажимать кнопку «Scan» и генерировать тысячестраничные PDF-отчёты, в которых 99% — это ложные срабатывания (False Positives)? Настоящая работа безопасника начинается там, где заканчивается работа сканера — нужно верифицировать уязвимость руками, доказать её эксплуатабельность и написать рекомендации. Скрипт-кидди этого не умеют.

Финансовый потолок: 0 — 30 000 рублей в месяц (если очень повезёт поймать что-то сканером на широком скоупе).

Уровень 2: Junior Security Specialist / Базовый Пентестер

Это уже профессионал. Главное отличие Junior'a от Script Kiddie в том, что Junior понимает, *как* работают инструменты, и умеет интерпретировать их вывод. Он знает базу: сети (TCP/IP, маршрутизация), устройство операционных систем (Windows/Linux) и основы веб-технологий (HTTP, сессии, куки).

Junior делает рутину:

Проводит базовые аудиты защищённости по методикам (например, OWASP Top 10 для веба).

Умеет руками раскрутить простую SQL-инъекцию (SQLi) или настроить обход базового WAF (Web Application Firewall) для Cross-Site Scripting (XSS).

Собирает данные из разных сканеров (Nmap, Burp Suite, Metasploit), отфильтровывает мусор и пишет связный отчёт.

Где здесь деньги? Это вход в легальную профессию. В 2026 году российский рынок испытывает жесточайший кадровый голод, потому что бизнесу (особенно финансовому сектору и ритейлу) нужно срочно тестировать огромные объёмы нового отечественного ПО.

Зарплаты Junior-пентестеров в найме сейчас стартуют от 80 000 до 150 000 рублей. Это хорошие деньги для старта. Но проблема в том, что на этом уровне ты продаёшь своё время оптом. Твоя добавленная стоимость невелика — ты делаешь понятную, алгоритмизированную работу.

На фрилансе или в Bug Bounty Junior может зарабатывать нестабильно: сегодня нашёл логический баг на 50 000 рублей, а потом два месяца глухо. Конкуренция на простых уязвимостях огромная.

Финансовый потолок: 80 000 — 150 000 рублей в месяц.

Уровень 3: Middle / Advanced Pentester & Red Teamer

Здесь начинается серьёзная игра и серьёзные деньги. Middle-специалист не просто ищет баги по чеклисту. Он мыслит сценариями (Attack Paths) и понимает бизнес-логику целевой системы.

Что умеет Middle:

Цепочки уязвимостей (Chaining). Найти SSRF (Server-Side Request Forgery) в облачной инфраструктуре, вытащить через него токены AWS/Яндекс. Облака, авторизоваться, найти бакет с бэкапами базы данных и слить оттуда хеши паролей администраторов. Это не одна уязвимость, это вектор.

Обход защитных механизмов (Evasion). Middle умеет писать свои пейлоады (нагрузки), которые не детектируются популярными антивирусами и EDR (Endpoint Detection and Response) системами.

Red Teaming. Участие в комплексных симуляциях кибератак (когда ломают не конкретный сервис, а всю компанию целиком — через фишинг, социальную инженерию, физическое проникновение в офис и захват Active Directory).

Разработка. Умеет писать собственные инструменты и скрипты для автоматизации сложных рутинных задач (например, кастомный фаззер для специфического API).

Где здесь деньги? Везде. На этом уровне ты становишься востребованным активом.

В найме (консалтинг, банки, IT-гиганты) зарплаты Middle-пентестеров и Red Teamer'ов в 2026 году составляют от 150 000 до 250 000 рублей. И это только оклад. Многие компании доплачивают бонусы за успешно сданные проекты или найденные критические баги во внутренних системах.

Во фрилансе Middle может брать заказы на аудит небольших компаний (МСБ) с чеком от 100 000 до 300 000 рублей за проект (который делается за 1—2 недели чистого времени).

В Bug Bounty это уровень, где начинаются стабильные выплаты. Middle-хантер уже не конкурирует с толпой на поверхности. Он лезет в сложные места: тестирует десктопные клиенты, ковыряет толстые клиенты на Java, анализирует протоколы аутентификации (OAuth/SAML) и находит критические уязвимости (RCE, IDOR, утечки ПДн). Выплаты на российских платформах за такие баги составляют от 50 000 до 500 000 рублей за штуку.

Финансовый потолок: 150 000 — 400 000 рублей в месяц (если комбинировать найм и Bug Bounty/фриланс).

Уровень 4: Senior Vulnerability Researcher / Эксперт

Это элита. Вершина пищевой цепи. Senior Researcher не просто использует известные техники атак — он их *создаёт*.

Что отличает Senior'a:

Реверс-инжиниринг (Reverse Engineering) и 0day. Он может разобрать бинарный файл коммерческого антивируса, маршрутизатора или IoT-устройства, найти в нём переполнение буфера (Buffer Overflow) или Use-After-Free, и написать рабочий эксплойт, обходящий современные механизмы защиты памяти (ASLR, DEP).

Исследование архитектур. Он находит уязвимости не в конкретном сайте, а в самих протоколах или фреймворках (например, обход аутентификации в популярном SSO-решении, которое используют тысячи компаний).

Threat Intelligence и Forensic. Глубокое понимание того, как работают топовые правительственные хакеры (APT-группировки), анализ сложной малвари и проектирование систем защиты уровня Enterprise.

Управление и консалтинг. Senior может прийти к CISO (директору по ИБ) крупного банка и на языке денег объяснить, почему их текущая архитектура — решето, и как её перестроить, чтобы снизить бизнес-риски на миллионы долларов.

Где здесь деньги? Здесь деньги перестают быть зарплатой и становятся гонорарами или долями.

В найме топовые эксперты (Lead Security Researcher, Head of Offensive Security) в России зарабатывают от 300 000 до 600 000 рублей в месяц, а иногда и до 1,5 миллионов (если это уникальные спецы с именем). Их хантят не через HeadHunter, а через личные знакомства на конференциях вроде Positive Hack Days или OffZone.

На Bug Bounty Senior-хантеры забирают максимальные выплаты. Они находят RCE (Remote Code Execution) в корневых системах или обходы бизнес-логики, которые стоят от 300 000 до 1 000 000 рублей за один репорт (а на частных западных программах — десятки тысяч долларов).

Но самое интересное начинается в смежных нишах. Senior может:

Продавать найденные 0day-уязвимости легальным брокерам (например, Zerodium, если мы говорим о мировом рынке, где RCE под iOS стоит до \$2 000 000).

Открывать свой бутиковый консалтинг (аудиты смарт-контрактов в крипте, где чек за аудит одного контракта может составлять \$20 000 — \$50 000).

Создавать свои образовательные продукты или SaaS-решения для безопасников.

Финансовый потолок: Не ограничен (от 500 000 рублей в месяц и до бесконечности, зависящей от предпринимательской жилки).

Почему рынок платит так нелинейно?

Посмотри на цифры выше. Разница в доходе между Junior и Senior — не в два раза, а в десять, а то и в сто раз. Почему так происходит?

Ответ прост: **рынок платит за уникальность и масштаб предотвращённого ущерба.**

Когда Junior находит XSS, он предотвращает потенциальную кражу сессии одного пользователя. Это стоит 10 000 рублей. Когда Senior находит RCE на сервере процессинга кредитных карт, он предотвращает кражу миллионов рублей, полную остановку бизнеса и гигантские штрафы от регуляторов. Компания легко отдаст ему 500 000 рублей в виде Bug Bounty, потому что для неё это копейки по сравнению с потенциальным ущербом.

Деньги лежат там, где сложно. Там, куда не дотягиваются автоматические сканеры, где нужно читать документацию, реверсить код и понимать бизнес-процессы. Если ты хочешь зарабатывать больше — перестань конкурировать с толпой на уровне простых тулзов. Начинай копать вглубь.

Практическое задание

Определи свою точку на спектре.

Открой свой файл «Мой хакерский трек».

Создай раздел «Мой уровень в 2026 году».

Честно (без завышения и самоуничтожения) отнеси себя к одному из 4 уровней: Script Kiddie, Junior, Middle или Senior.

Напиши 3 конкретных навыка, которых тебе прямо сейчас не хватает, чтобы перейти на следующий уровень. (Например, если ты Junior, тебе может не хватать: умения обходить WAF руками, базового знания Python для написания скриптов, понимания Active Directory).

Напротив каждого из 3 навыков напиши, как ты будешь его прокачивать в ближайший месяц (книга, курс, лабораторная на HackTheBox).

Не ври себе. Точная диагностика — это половина успешного эксплойта.

— Специализации с самым высоким чеком в 2025—2026

Кибербезопасность — это не монолит. Это огромный супермаркет профессий, где на одной полке лежат дошираки (скучный комплаенс за копейки), а на другой — черная икра (эксклюзивный ресёрч за миллионы). Проблема большинства новичков в том, что они хватают первое попавшееся, не глядя на ценник.

В 2025—2026 годах рынок России трансформировался. Ушли западные вендоры, пришли отечественные экосистемы. Выросли атаки через цепочки поставок (Supply Chain). Искусственный интеллект стал рабочим инструментом и новой дырой в безопасности одновременно.

Вот топ-5 специализаций, где спрос превышает предложение настолько, что работодатели готовы платить выше рынка, лишь бы закрыть вакансию.

1. Application Security Engineer (AppSec) / DevSecOps

Чек: 250 000 — 500 000+ рублей/мес.

Это короли современной разработки. Раньше безопасники приходили в конце проекта, находили баги и тормозили релиз на месяц. Бизнес это ненавидел. Сейчас безопасников встраивают прямо в процесс разработки (CI/CD конвейер).

AppSec-инженер — это гибрид. Он понимает код (Python, Java, Go, JS) так же хорошо, как разработчик, но при этом видит в нём уязвимости. Он не просто находит XSS сканером. Он:

Встраивает SAST/DAST инструменты в GitLab/Jenkins, чтобы баги ловились автоматически при каждом коммите.

Учит разработчиков писать безопасный код (Security Champions программы).

Проводит архитектурное ревью (Threat Modeling) ещё до того, как написана первая строка кода.

Почему платят много? Потому что один AppSec экономит компании десятки миллионов рублей на исправлении багов до релиза. Исправить уязвимость на этапе архитектуры стоит \$1. Исправить её в продакшене — \$10 000. Бизнес умеет считать. Плюс, найти человека, который и кодит, и ломает — адски сложно.

2. Cloud Security Architect / Engineer

Чек: 300 000 — 600 000+ рублей/мес.

Весь крупный бизнес переехал в облака (Яндекс. Облако, VK Cloud, SberCloud). Но админы по привычке настраивают облака как обычные сервера «железе». И тут начинается праздник для хакеров: открытые S3-бакеты, торчащие наружу Kubernetes API, IAM-роли с правами администратора у стажёра.

Cloud Security инженер знает, как безопасно настроить Kubernetes, Docker, Terraform и облачные сервисы. Он понимает специфику контейнеризации и микросервисов.

Почему платят много? Потому что утечка из облака — это сразу всё: база клиентов, исходный код, ключи шифрования. Одна ошибка в конфиге Terraform может стоить компании существования. Специалистов, которые глубоко понимают безопасность K8s и российских облаков — единицы.

3. Red Teamer / Senior Pentester

Чек: 250 000 — 550 000+ рублей/мес. (плюс бонусы за проекты).

Это элита атакующей безопасности. Если пентестер проверяет конкретный сервис по списку (OWASP), то Red Team проверяет *всю компанию* на прочность.

Red Teamer имитирует реальную АPT-группировку. Он может:

Отправить фишинговое письмо бухгалтеру.

Пройти в офис под видом курьера и воткнуть «малину» (Raspberry Pi) в сетевую розетку.

Сбрутить пароль админа через забытый шлюз.

Тихо сидеть в сети месяцами, повышая привилегии и выкачивая данные, пока SOC (Blue Team) его не заметит.

Почему платят много? Потому что это *реальная* проверка боем. Топ-менеджмент банков и корпораций готов платить огромные деньги, чтобы их «взломали» свои, показали дыры в процессах и людях, а не только в софте. Это страховка от катастрофы. Квалифицированный Red Teamer должен знать не только технический взлом, но и тактику, психологию и OPSEC (операционную безопасность).

4. Smart Contract Auditor / Web3 Security

Чек: \$5 000 — \$20 000+ за аудит (проектная работа).

Крипта и блокчейн никуда не делись, они трансформировались. DeFi (децентрализованные финансы), токенизация активов (ЦФА) в России — это огромный рынок. Одна ошибка в смарт-контракте на Solidity или Rust может стоить миллионы долларов, которые невозможно вернуть (блокчейн помнит всё).

Аудитор читает код контракта, ищет логические ошибки (Reentrancy, Front-running, манипуляции с оракулами) и математические неточности.

Почему платят много? Потому что цена ошибки здесь абсолютна. Взломанный банк можно откатить по бекапам. Взломанный смарт-контракт — это потеря денег навсегда. Аудиторов, которые реально понимают глубокую математику DeFi и безопасность EVM (Ethereum Virtual Machine), в мире очень мало, а в России — тем более.

5. AI Security Specialist / ML Sec Ops

Чек: Формируется, но уже стартует от 300 000 рублей/мес.

Самая свежая и горячая ниша. Компании массово внедряют LLM (Large Language Models) для поддержки, аналитики и генерации контента. Но никто не знает, как их защищать.

Специалист по AI Security занимается:

Prompt Injection: как заставить чат-бота выдать конфиденциальные данные или проигнорировать инструкции.

Data Poisoning: защита обучающих выборок от внедрения вредоносных паттернов.

Adversarial Attacks: обман систем компьютерного зрения и классификаторов.

Secure AI Pipeline: безопасное развёртывание моделей.

Почему платят много? Потому что это Wild West. Угроз много, стандартов нет, экспертов нет. Если ты разберёшься, как ломать и защищать AI сейчас — через 2 года ты будешь стоить миллионы. Это инвестиция в будущее.

Как выбрать свою нишу?

Не пытайся объять необъятное. Выбери *одно* направление и бей в точку.

Любишь код и автоматизацию? Иди в **AppSec/DevSecOps**.

Нравится ломать, социальная инженерия и адреналин? Твой путь — **Red Teaming**.

Фанат инфраструктуры, Linux и контейнеров? **Cloud Security** ждёт.

Хочешь работать удалённо за валюту и любишь математику? Учи **Smart Contracts**.
Хочешь быть на острие прогресса? Копай **AI Security**.

Деньги есть везде. Но самые большие деньги — там, где ты уникален.

Практическое задание

Выбери свою специализацию.

Открой файл «Мой хакерский трек».

Перечитай описания 5 специализаций выше.

Выбери **одну** (максимум две смежных), которая тебе ближе всего по духу и навыкам.

Зайди на HeadHunter, Хабр Карьеру или LinkedIn. Найди 3—5 вакансий по этой специализации.

Выпиши в файл список требований из этих вакансий:

Какие языки программирования нужны?

Какие инструменты (Burp, K8s, Solidity)?

Какие сертификаты (OSCP, СКА, CISSP)?

Это твой персональный roadmap обучения на ближайшие полгода. Теперь ты знаешь, что учить не «всё подряд», а конкретно под высокий чек.

— Как рынок оценивает твои навыки (грейды, сертификаты, портфолио)

Представь ситуацию: на одну и ту же вакансию Middle-пентестера с вилкой 200 000 — 250 000 рублей откликаются два кандидата.

Первый — Вася. Он пять лет просидел в отделе ИБ регионального банка, настраивая антивирусы и раз в год запуская сканер уязвимостей. У него в резюме написано: «Опыт работы 5 лет, глубокие знания сетевых технологий, уверенный пользователь Burp Suite». Второй — Илья. Ему 21 год. У него вообще нет официального опыта работы в трудовой книжке. Но в его резюме три ссылки: на профиль HackerOne (где у него десяток выплаченных баунти от [Mail.ru](https://mail.ru) и Яндекса), на его GitHub (где лежит форк популярного фаззера с его оптимизациями) и на статью на Хабре, где он подробно разобрал цепочку эксплуатации свежей CVE в Microsoft Exchange.

Угадай, кому предложат верхнюю границу вилки, а чьё резюме полетит в корзину ещё на этапе скрининга?

В кибербезопасности, в отличие от классической разработки или юриспруденции, **выслуга лет не значит практически ничего**. Год, проведённый за переключением бумажек и настройкой фаерволов по инструкции, не делает тебя специалистом по атакующей безопасности. Рынок оценивает не время, которое ты потратил, а плотность твоего боевого опыта и способность доказать этот опыт в моменте.

Давай разберём три главных столпа, на которых строится твоя стоимость на рынке: грейды, сертификации и портфолио. И посмотрим, как расставить приоритеты так, чтобы максимизировать доход.

1. Грейды (Junior, Middle, Senior): Иллюзия объективности

Грейдирование — это попытка корпораций впихнуть хаос человеческих навыков в понятную табличку Excel. Проблема в том, что в ИБ нет единого стандарта, что должен уметь Middle, а что — Senior.

В интеграторе (компании, которая продаёт пентесты на сторону) ты можешь быть «Сеньором», потому что умеешь красиво общаться с клиентами и быстро штамповать отчёты по шаблону. Но если ты с этим же грейдом придёшь на собеседование в продуктовую Red Team

команду Яндекса или Тинькофф, тебя могут раскатать техническими вопросами до уровня крепкого Джуна.

Как рынок *на самом деле* определяет твой грейд в деньгах?

Junior (до 150 000 руб.): Тебе нужно ставить задачи. Тебе нужно давать чёткий score (рамки) и методологию. Если что-то идёт не по инструкции (например, WAF блокирует стандартные пейлоады), ты поднимаешь руки и идёшь к наставнику. Ты — исполнитель рутины.

Middle (150 000 — 300 000 руб.): Тебе дают цель, а путь ты выбираешь сам. Ты можешь взять проект «под ключ» от разведки до написания финального отчёта. Ты умеешь гуглить, адаптировать чужие эксплойты под текущую задачу и обходить базовые средства защиты. Ты — самостоятельная боевая единица.

Senior (от 300 000 руб. и до бесконечности): Ты сам ставишь задачи. Ты видишь архитектурные изъяны системы до того, как начнёшь её ломать. Ты создаёшь новые векторы атак, пишешь собственные инструменты (потому что существующих не хватает) и, самое главное, можешь обучать Мидлов и Джунов. Ты — генератор прибыли и снижатель глобальных рисков для бизнеса.

Твой грейд на собеседовании определяется не ответом на вопрос «Чем TCP отличается от UDP?», а тем, как ты решаешь архитектурные и нестандартные кейсы, которые тебе дают в качестве тестового.

2. Сертификаты: Ускоритель, а не двигатель

В IT-тусовке принято либо молиться на сертификаты, либо презирать их. Истина, как всегда, посередине и строго привязана к деньгам.

Сертификат — это независимое подтверждение того, что в условиях стресса (экзамена) ты способен продемонстрировать заявленный уровень навыков. Работодатель любит сертификаты по одной простой причине: они снижают риск найма идиота. Если у тебя есть сложный практический сертификат, значит HR может смело пропустить тебя на техническое интервью — ты точно знаешь базу.

В 2026 году на российском рынке (в условиях изоляции от некоторых западных систем оплат и вендоров) котируются следующие бумажки:

Золотой стандарт Offensive Security:

OSCP (Offensive Security Certified Professional). Это классика. Экзамен длится 24 часа, за которые ты должен взломать несколько машин в изолированной сети и написать отчёт. Наличие OSCP автоматически открывает двери на большинство Middle-позиций по всему миру. Получить его из РФ сейчас сложно (через крипту и сервиса на три буквы), но он всё ещё ценится выше всего.

OSWE (Offensive Security Web Expert). Высшая лига веб-безопасности (анализ исходного кода, написание кастомных эксплойтов). Это прямой путь к зарплатам от 350 000 рублей.

Практические альтернативы:

PNPT (Practical Network Penetration Tester) от TCM Security. Более дешёвая, современная и очень практичная альтернатива OSCP. Экзамен имитирует реальный пентест корпоративной сети на 5 дней + живая защита отчёта. Отлично котируется на рынке.

CRT0 (Certified Red Team Operator) от Zero-Point Security. Фокус на работу с Cobalt Strike, обход антивирусов и симуляцию APT-атак. Must-have для Red Team.

Отечественные реалии: В России сейчас активно формируются собственные системы сертификации. Успешные выступления на **Standoff** (как часть атакующей команды) или наличие статуса топ-хантера на платформах **BI. ZONE Bug Bounty** часто заменяют любые международные бумажки, потому что это доказательство работы в реалиях российского интерпрайза.

Важное правило: Никогда не получай сертификат ради сертификата. Сертификат должен окупаться (ROI — Return on Investment). Если OSCP стоит ~\$1600, он должен принести тебе прибавку к зарплате минимум в 50 000 рублей в месяц, чтобы окупиться за первый квартал.

Если ты идёшь сдавать теорию СЕН (Certified Ethical Hacker) просто чтобы «было» — ты сливаешь деньги.

3. Портфолио: Твой главный актив в 2026 году

Мы подошли к самому важному. В 2026 году твоё резюме читают по диагонали. Твои сертификаты проверяют галочкой. Но твоё портфолио изучают под микроскопом.

Портфолио в кибербезе — это доказательство того, что ты уже делал то, за что просишь деньги.

Что должно входить в сильное портфолио хакера, который хочет дорого стоить:

Публичные профили Bug Bounty (HackerOne, Bugcrowd, Standoff 365). Даже если там нет выплат на сотни тысяч, наличие там подтверждённых багов (Resolved) показывает, что ты умеешь работать с реальными продакшен-системами, а не только с тепличными лабами.

Профиль на HackTheBox (HTB) / TryHackMe. Статус Pro Hacker или Omniscient на HTB — это маркер того, что ты постоянно держишь мозги в тонусе и следишь за новыми векторами атак.

GitHub / GitLab. Это критически важно. Умение кодить для пентестера — это множитель зарплаты на 1.5x. Твой GitHub не обязан содержать ядро новой операционной системы. Достаточно 3—5 репозиториев:

Скрипт на Python для автоматизации сбора поддоменов.

Кастомный плагин для Burp Suite (написанный на Java или Python).

Форк известного инструмента, в котором ты пофиксил баг или добавил фичу.

Технические статьи и Write-up'ы. Умение внятно излагать мысли письменно стоит огромных денег, потому что финальный продукт пентеста — это текст (отчёт). Если ты завёл блог на Хабре, Medium или свой Telegram-канал, где детально, с кусками кода и скриншотами разбираешь сложные уязвимости — ты снимаешь главный страх работодателя: «А сможет ли этот гений нормально объяснить клиенту, что он сломал?».

CVE (Common Vulnerabilities and Exposures). Высший пилотаж портфолио. Если ты нашёл уязвимость в Open Source проекте или коммерческом софте, сообщил разработчикам и получил именной CVE-идентификатор — ты навсегда вписал своё имя в историю ИБ. Это бетонный аргумент на любом собеседовании.

Синтез стоимости: как это работает на практике

Рынок оценивает тебя по пересечению этих трёх факторов.

Если у тебя есть только теоретические знания и голый диплом вуза — ты стажёр, который будет работать за еду и опыт. Если у тебя есть знания и пара сертификатов (например, PNPT) — ты крепкий Джун/Мидл, претендующий на 150 000 рублей. Но если ты приходишь на собеседование и говоришь: «Вот мой профиль на HTB (топ-100 по России), вот мой скрипт на GitHub, который автоматизирует эксплуатацию недавней CVE-2025-XXXX, и вот мой write-up, где я разобрал, как обошёл WAF на реальном Bug Bounty проекте» — собеседование заканчивается. Начинаются торги по зарплате. И диктовать условия будешь ты.

В этой книге мы будем строить именно третью модель. Каждая практика, которую ты будешь выполнять в следующих главах, направлена на создание артефактов для твоего портфолио. Потому что слова стоят дёшево. Доказательства стоят дорого.

Практическое задание

Собери своё первичное портфолио.

Открой файл «Мой хакерский трек». Создай раздел «Моё портфолио (Текущее состояние)».

Собери все ссылки, которые у тебя есть прямо сейчас:

Ссылка на профиль GitHub. Если там пусто — напиши «Пусто, нужно залить первый скрипт».

Ссылки на профили HackTheBox / TryHackMe / RootMe. Укажи свой ранг.

Ссылки на профили Bug Bounty платформ.

Ссылки на любые твои технические тексты (посты в форумах, статьи, write-up'ы).

Посмотри на этот список глазами HR-директора, который готов платить тебе 250 000 рублей в месяц. Выглядит убедительно?

Задача на неделю: Создай один новый артефакт. Если нет профиля на НТВ — зарегистрируйся и пройди первую машину. Если есть машина — напиши по ней подробный write-up и выложи в открытый доступ. Если умеешь кодить — напиши простенький сканер портов или парсер ответов HTTP и залей на GitHub с хорошим README.

Сделай это. Рынок не поверит тебе на слово.

Глава 2. Стек навыков = стек доходов

— Языки, которые реально нужны: Python, Rust, Go, JavaScript, Bash

В мире существуют сотни языков программирования. Но хакеру для заработка нужны не все. Тебе не нужно знать Haskell (если ты не пишешь смарт-контракты на Cardano) или 1С (хотя за аудит 1С платят жирно, но это узкая ниша боли).

В 2026 году есть конкретный набор инструментов, который покрывает 99% задач: от быстрого написания эксплойта «на коленке» до разработки сложной малвари, обходящей EDR. Этот набор — твой швейцарский нож. Чем острее лезвия, тем толще слой масла на твоём бутерброде.

1. Python: Lingua Franca хакеров

Статус: Обязательная база. Без него ты не хакер, а юзер. **Зачем нужен:** Автоматизация, PoC (Proof of Concept), ML/AI скрипты.

Python — это клей интернета и главный язык кибербезопасности. 90% эксплойтов на Exploit-DB написаны на Питоне. Все топовые инструменты (Impacket, Scapy, Pwntools) — это Питон.

Где деньги:

Автоматизация рутины: Написать скрипт, который парсит 10 000 JSON-файлов из утечки и находит там пароли — дело 15 минут. Руками ты бы делал это неделю.

Написание PoC: Вышла новая CVE. Ты первый написал рабочий эксплойт на Python и залил на GitHub или продал на платформе — профит.

Взлом AI/ML: Библиотеки PyTorch и TensorFlow — это Python. Хочешь ломать нейронки через adversarial attacks? Учи питон.

Твой уровень для заработка: Не нужно быть Senior Python Developer. Тебе нужно уметь работать с библиотекой requests (для веба), socket (для сетей), pwntools (для бинарщины) и argparse (чтобы твои тулзы запускались из консоли по-человечески).

2. Bash: Язык выживания в Linux

Статус: Жизненно необходим. **Зачем нужен:** Работа с серверами, пайплайны, «жизнь в терминале».

Ты взломал сервер. Получил доступ к консоли (shell). У тебя нет графического интерфейса, нет мышки, нет браузера. Только чёрный экран и мигающий курсор. Если ты не знаешь Bash, ты беспомощен.

Где деньги:

One-liners (однострочники): Умение одной строкой найти все файлы с конфигами, grep отсюда пароли и отправить себе на сервер — это основа пост-эксплуатации.

Recon pipelines: Связать subfinder, httpx и nuclei в один конвейер, который работает 24/7 на VPS и шлёт уведомления в Telegram, когда находит новый баг — это делается на Bash.

DevSecOps: CI/CD пайплайны в GitLab часто состоят из bash-скриптов. Понимать их = находить дыры в деплое.

Твой уровень для заработка: Пайпы (|), перенаправление потоков (>, 2> &1), циклы for, работа с grep, awk, sed. Этого хватит, чтобы чувствовать себя богом в чужой системе.

3. Go (Golang): Новый стандарт скоростного сканирования

Статус: Выбор профессионалов для High-Load атак. **Зачем нужен:** Быстрые сканеры, микросервисы, кроссплатформенная малварь.

За последние 5 лет Go вытеснил Python из ниши высокопроизводительных инструментов. ProjectDiscovery (авторы Nuclei, Subfinder) пишут на Go. Почему? Потому что он компилируется в один бинарник (не нужно тащить зависимости и устанавливать python на жертву) и он *безумно* быстр благодаря многопоточности (goroutines).

Где деньги:

Разработка своих сканеров: Если тебе нужно просканировать весь интернет на наличие открытого порта за час — Python умрёт. Go справится. Уникальный быстрый сканер — это конкурентное преимущество в Bug Bounty.

C2 и малварь: Go легко кросс-компилируется. Написал код на Make — скомпилировал под Windows, Linux и ARM одной командой. Антивирусы до сих пор плохо детектят Go-бинарники из-за их специфической структуры.

Облачная безопасность: Kubernetes и Docker написаны на Go. Чтобы искать в них Oday, нужно читать код на Go.

Твой уровень для заработка: Понимание горутин, каналов, структур и умение собрать статический бинарник без зависимостей.

4. JavaScript / TypeScript: Ключ от всех дверей веба

Статус: Главный язык атакуемого периметра. **Зачем нужен:** Client-side атаки (XSS), анализ веб-приложений, Node.js бэкенды.

Веб захватил мир. А веб — это JavaScript. Если ты хочешь ломать сайты (а это 80% рынка Bug Bounty и пентеста), ты обязан читать JS как родной язык.

Где деньги:

XSS (Cross-Site Scripting): Чтобы раскрутить XSS до перехвата аккаунта или RCE (через Electron), нужно писать на JS. Пейлоады типа alert (1) денег не приносят. Пейлоады, воруящие токены и обходящие CSP — приносят тысячи долларов.

Code Review: Современные фронтенды (React, Vue, Angular) содержат кучу логики на клиенте. Часто там забывают секретные ключи или прописывают дырявую логику авторизации. Умеешь читать минифицированный JS в DevTools — найдёшь баги, которые пропустили сканеры.

Server-side JS (Node.js): Огромная часть бэкендов сейчас пишется на Node. Prototype Pollution, RCE через десериализацию — это специфичные для JS уязвимости.

Твой уровень для заработка: Понимание DOM, Fetch API, LocalStorage, замыканий и асинхронности. И умение читать чужой «спагетти-код».

5. Rust: Оружие Судного дня 2026 года

Статус: Элитарный, сложный, для самых дорогих задач. **Зачем нужен:** Обход антивирусов (AV/EDR evasion), разработка эксплойтов, системное программирование.

Rust — это новый C++, только безопасный (для разработчика) и убийственный (для защитника). В 2026 году вирусописатели и Red Teamer'ы массово мигрируют на Rust.

Где деньги:

Bypassing EDR: Современные системы защиты (Kaspersky, CrowdStrike) научились хорошо ловить Python, PowerShell и даже Go. Rust пока остаётся «слепой зоной» во многих

случаях. Написать лодер (загрузчик) для Cobalt Strike на Rust — значит повысить шансы на успешную атаку в разы.

Разработка C2 имплантов: Инструменты типа Mythic C2 активно используют агенты на Rust.

Высокопроизводительный фаззинг: Искать уязвимости в памяти (Memory Corruption) лучше всего инструментами на Rust.

Твой уровень для заработка: Rust сложен. У него высокий порог входа (Borrow Checker выпьет твою кровь). Но знание Rust автоматически переводит тебя из категории «Скрипт-кидди» в категорию «Инженер-разработчик эксплойтов». Зарплата растёт соответственно.

Вердикт: Что учить прямо сейчас?

Не пытайся выучить всё сразу — мозг взорвётся. Иди по стеку доходов:

Уровень «Хочу кушать» (Junior):

Python: Чтобы писать простые скрипты.

Bash: Чтобы не тупить в консоли Linux.

Результат: Ты можешь автоматизировать рутину и пользоваться чужими эксплойтами.

Уровень «Хочу на Бали» (Middle):

JavaScript: Чтобы глубоко копать веб и находить сложные XSS/DOM-based уязвимости.

Go: Чтобы написать свой быстрый сканер и обогнать конкурентов на Bug Bounty.

Результат: Ты находишь баги, которые другие пропускают, и делаешь это быстрее.

Уровень «Хочу яхту» (Senior / Red Team):

Rust: Чтобы писать малварь, которую не видит антивирус, и создавать инструменты enterprise-уровня.

C/C++ (бонус): Для реверса и понимания, как работает память (Windows API, syscalls).

Результат: Ты можешь сломать то, что считается защищённым, и твои инструменты стоят тысячи долларов на приватном рынке (или приносят тебе офферы от топ-компаний).

Язык — это рычаг. Чем прочнее материал рычага, тем тяжелее камень (проблему заказчика) ты можешь сдвинуть. А платят нам именно за вес сдвинутых камней.

Практическое задание

Выбери свой «Боевой язык».

Открой файл «Мой хакерский трек».

Честно оцени свои знания по шкале 0—5 для каждого из пяти языков (Python, Bash, JS, Go, Rust).

Выбери **один** язык, который станет твоим приоритетом на ближайший месяц.

Если ты совсем новичок -> Python.

Если хочешь в веб-пентест -> JavaScript.

Если метишь в Red Team -> Go (потом Rust).

Задача-минимум: Найди на GitHub любой популярный инструмент на выбранном языке (например, sqlmap для Python или nuclei для Go), открой исходный код и попытайся понять, как работает *одна конкретная функция*. Напиши комментарий к ней для себя — что она делает.

Задача-максимум: Напиши «Hello World» в мире хакеров — **Port Scanner**.

На Python (с использованием socket).

Или на Bash (с использованием /dev/tcp).

Или на Go (с использованием горутин).

Код сканера не должен быть идеальным. Он должен *работать*. Залей его в свой GitHub. Это первая строчка твоего кода, которая приносит пользу.

— Операционные системы и лабораторная база: Твой хакерский бункер

Давай сразу проясним: твой основной ноутбук (хостовая машина), с которого ты смотришь YouTube, играешь в игры и заходишь в онлайн-банк — **никогда не должен использоваться для реальных атак или тестирования вирусов**. Одно неверное движение, один клик по чужому эксплойту с бэкдором (а на GitHub такого мусора полно) — и ты сам становишься чьей-то жертвой. Твои пароли уплывают, а твой IP светится в логах атакованной компании.

Профессионал отличается от любителя тем, что у него есть чётко выстроенная, изолированная и легко восстанавливаемая лабораторная база. Это твой полигон, твой завод по производству эксплойтов и твой щит.

Давай разберём, как собрать инфраструктуру, которая соответствует уровню заработка в 250 000+ рублей в месяц в 2026 году.

1. Операционные системы: Триада хакера

Тебе придётся жить в трёх мирах одновременно. Если ты знаешь только одну ОС — ты слеп на две трети.

А) Linux (Offensive & Server)

Твоя роль: Дом родной.

Зачем: Это система, из которой ты атакуешь, и система, на которой крутятся 90% веб-серверов в мире. Не знать архитектуру Linux (где лежат логи, как работают права доступа chmod/chown, что такое systemd и cron) — значит не уметь закрепиться в системе после взлома (Post-Exploitation).

Что ставить:

Kali Linux / Parrot OS: Классика для пентеста. Всё предустановлено, всё работает из коробки. Идеально для старта, чтобы не тратить неделю на сборку тулзов из исходников. Ставь на виртуалку.

Ubuntu / Debian: Это то, что ты будешь встречать на серверах жертв. Ты должен уметь поднимать на них веб-серверы, базы данных (PostgreSQL, MySQL) и Docker-контейнеры с закрытыми глазами, чтобы понимать, где админы обычно косячат с конфигами.

Б) Windows (Enterprise Target)

Твоя роль: Территория врага (или самого жирного клиента).

Зачем: 99% корпоративных сетей в России в 2026 году (несмотря на импортозамещение) всё ещё сидят на Active Directory (AD). Взломать веб-сайт компании — это баунти на 50 000 рублей. Сдампить базу доменных хешей (NTDS.dit) и получить права Enterprise Admin — это Red Team проект на 2 миллиона рублей.

Что ставить: Тебе нужна локальная лаборатория с Windows Server (2019/2022) и парочкой клиентских машин на Windows 10/11. Ты должен понимать, как работают групповые политики (GPO), протоколы аутентификации (Kerberos, NTLM) и как работает Windows API (чтобы писать малварь на Rust/C++, которая обходит Defender).

В) macOS (Daily Driver)

Твоя роль: Мобильный штаб.

Зачем: Это субъективно, но большинство топовых хантеров и секьюрити-инженеров сидят на макбуках. Под капотом у мака — UNIX (Darwin), что даёт тебе нормальный терминал (zsh, bash), но при этом у тебя не отваливается Wi-Fi после обновления ядра, как это часто бывает на десктопном Linux. Это идеальная хостовая машина, на которой крутятся виртуалки.

2. Виртуализация: Изоляция — это жизнь

Забудь про Dual Boot (две системы на одном диске). Это неудобно, долго и не позволяет тестировать сетевые атаки между машинами. Твой лучший друг — гипервизоры.

Локальная виртуализация (VMware Workstation / VirtualBox): Это база. У тебя должен быть настроен изолированный виртуальный коммутатор (Host-Only Network), внутри которого живут твои атакующие и уязвимые машины. Они не должны иметь прямого доступа в твою домашнюю сеть, пока ты им это не разрешишь.

Снапшоты (Snapshots): Перед тем как запустить сомнительный бинарник или начать ломать сложную машину, делай снимок состояния. Сломал систему в процессе экспериментов? Откатился за 5 секунд. Это экономит сотни часов жизни.

3. Облачная инфраструктура (VPS/VDS): Выход в большой мир

Если ты занимаешься Bug Bounty или внешним пентестом, твоего домашнего IP-адреса недостаточно. Во-первых, домашний провайдер может блокировать нестандартные порты. Во-вторых, светить свой домашний IP в логах чужих серверов — это грубейшее нарушение OPSEC (Operational Security).

В 2026 году у тебя должна быть распределённая облачная инфраструктура:

Recon-сервер (Разведка): Мощный VPS (от 4 ядер, 8 ГБ RAM), желательно за пределами РФ (чтобы обходить геоблокировки западных сервисов), на котором крутятся инструменты сбора данных (Axiom, Subfinder, Amass). Разведка требует широкого канала и времени. Твой домашний ноут может уснуть, а VPS будет молотить интернет 24/7.

C2-сервер (Command and Control): Если ты тренируешься в Red Teaming, тебе нужен отдельный, хорошо защищённый сервер, куда будут «стучаться» твои пейлоады. Этот сервер должен быть скрыт за цепочкой редиректоров (например, через Cloudflare или AWS CloudFront), чтобы защитники не вычислили настоящий IP твоей базы.

Collaborator/OOB (Out-of-Band) сервер: Для поиска слепых уязвимостей (Blind XSS, Blind SSRF, OOB SQLi). Когда ты отправляешь пейлоад, который заставляет сервер жертвы сделать DNS-запрос или HTTP-запрос к *твоему* серверу. Можно использовать встроенный Burp Collaborator, но профессионалы поднимают свой (например, ProjectDiscovery Interactsh). Это сильно повышает успешность находок на приватных программах.

4. Лабораторная база для тренировок

Где легально ломать, пока у тебя нет клиентов?

HackTheBox (HTB) / TryHackMe (THM): Это спортзалы кибербеза. HTB жёстче, THM более дружелюбен к новичкам. Наличие VIP-подписки на одной из этих платформ в 2026 году — это обязательные инвестиции в себя (около \$10—15 в месяц). Если тебе жалко этих денег на своё развитие — закрой книгу, ты не заработаешь здесь ничего.

VulnHub: Отличный бесплатный ресурс с готовыми образами уязвимых виртуалок, которые ты скачиваешь и ломаешь локально.

Сборка собственной AD-лаборатории: Это переход на Senior-уровень. Подними руками контроллер домена (Domain Controller), настрой пару пользователей, имитируй уязвимости (например, Kerberoasting или AS-REP Roasting) и попробуй их проэксплуатировать с помощью инструмента BloodHound.

Синтез: Как выглядит инфраструктура профи, зарабатывающего 300к+

Хост: MacBook Pro или мощный ThinkPad с зашифрованным диском (BitLocker / FileVault).

Локальные виртуалки: Kali Linux (для тестов), Windows 10 (для анализа малвари), Windows Server (для тестов AD).

Внешние сервера:

VPS на DigitalOcean/Hetzner (или российских аналогах Selectel/Timeweb для локальных задач) под 24/7 разведку.

Собственный сервис на три буквы (WireGuard/Outline) для шифрования трафика между ноутбуком и рабочим VPS.

Инструментарий: Лицензия Burp Suite Professional (да, это стоит денег, но окупается первой же найденной уязвимостью), настроенный терминал (tmux/screen) и аккаунты на НТВ/ТНМ.

Это твой завод. Если станки не настроены, ты не сможешь выпустить продукт. Настрой свою базу один раз по уму, и она будет кормить тебя годами.

Практическое задание

Разверни свой полигон.

Открой файл «Мой хакерский трек». Создай раздел «Моя Инфраструктура».

Задача-минимум (если у тебя ничего нет):

Скачай и установи VirtualBox.

Скачай образ Kali Linux и разверни виртуалку.

Настрой сетевой адаптер в режим NAT или Host-Only.

Сделай первый Snapshot (снимок состояния) чистой системы.

Задача-оптимум (для заработка):

Арендуй самый дешёвый VPS (виртуальный сервер) на Ubuntu (за 200—300 рублей в месяц).

Настрой подключение по SSH по ключам (запрети вход по паролю — это база OPSEC).

Установи на VPS Docker. Это позволит тебе поднимать любые уязвимые приложения для тестов за 10 секунд одной командой.

ЗадOCUMENTИРУЙ: Запиши в файл IP-адреса своих виртуалок и серверов. Напиши памятку: «Я никогда не атакую цели с хостовой машины без сервиса на три буквы/VPS».

Сделал? Отлично. У тебя появился свой бункер. Теперь пора поговорить о самом важном навыке хакера, который не связан с консолью, но приносит больше всего денег.

— Soft skills хакера: почему «объяснить клиенту» стоит дороже, чем «сломать»

В IT-тусовке есть вредный стереотип: «Я технический специалист. Моё дело — ломать (или защищать) системы. А разговаривать с клиентами, писать красивые бумажки и защищать бюджеты — это работа менеджеров. Пусть они этим и занимаются».

Это позиция человека, который добровольно отказывается от половины своего потенциального дохода.

Давай посмотрим правде в глаза. В 2026 году технические навыки (Hard Skills) — знание Python, умение раскрутить RCE через десериализацию Java, владение Burp Suite — это

гигиенический минимум. Это твой билет на стадион. Но то, на какую трибуну ты сядешь — на галёрку к джунам или в VIP-ложу к сеньорам и консультантам — определяют исключительно твои Soft Skills (гибкие навыки).

Почему так происходит? Потому что **кибербезопасность не существует в вакууме**. Она обслуживает бизнес. А бизнес разговаривает на языке денег, рисков и сроков, а не на языке эксплойтов, CVE и шеллкодов.

Разрыв перевода: Почему технарей не понимают (и не платят им больше)

Представь классическую ситуацию: ты, как внешний пентестер, нашёл уязвимость в интернет-банке. Ты приходишь на встречу с IT-директором (CIO) и директором по информационной безопасности (CISO) заказчика и с горящими глазами рассказываешь:

— *«Короче, я нашёл уязвимость в вашем API. Там эндпоинт /v1/transfer уязвим к Insecure Direct Object Reference (IDOR). Если подменить параметр user_id в JSON-нагрузке POST-запроса, то из-за отсутствия проверки на сервере (Access Control Flaw) можно обойти механизм авторизации и обратиться к чужому объекту базы данных!»*

Для тебя это звучит как победа. Ты доказал свою крутость. Но что в этот момент слышит бизнес? Бизнес слышит белый шум. Директор думает: «Опять этот хакер птичьим языком ругается. Наверное, это важно, раз он так машет руками, но как это влияет на мою квартальную премию? Надо ли мне сейчас останавливать релиз нового мобильного приложения ради этого вашего IDOR?».

А теперь представь, что на ту же встречу приходит Senior-консультант. Он нашёл ту же самую уязвимость, но говорит иначе:

— *«Коллеги, у нас критическая проблема в логике переводов. Прямо сейчас любой авторизованный пользователь может перевести деньги со счёта другого клиента на свой, просто подменив одну цифру в запросе. Я написал скрипт, который автоматизирует этот процесс. Потенциальный ущерб — прямые финансовые потери в размере миллионов рублей в час, плюс 100% вероятность отзыва лицензии ЦБ РФ в случае массовой эксплуатации. Релиз нового приложения необходимо остановить до внедрения патча. Патч займёт у разработчиков 4 часа».*

Уязвимость одна и та же. Техническая работа проделана одинаковая. Но второй специалист перевёл технический баг на язык **бизнес-рисков**. Он показал, сколько денег компания потеряет, что скажет регулятор, и сколько времени нужно на решение проблемы.

Заказчик заплатит первому специалисту по базовому тарифу (потому что он просто нашёл баг). Заказчик заплатит второму специалисту премию, наймёт его на постоянный консалтинг и порекомендует партнёрам (потому что он спас их бизнес от краха).

Три главных Soft Skills для хакера

Если ты хочешь расти в деньгах, тебе нужно прокачать три конкретных навыка коммуникации:

1. Написание отчётов (Reporting & Documentation)

Это самый недооценённый навык в ИБ. Большинство хакеров ненавидят писать отчёты. Для них взлом — это искусство, а отчёт — скучная бюрократия.

Но парадокс в том, что **клиент покупает не взлом, клиент покупает отчёт**. Клиент не видел, как ты полночи потел над обходом WAF. Клиент видит только PDF-документ, который ты ему прислал.

Хороший отчёт (за который платят миллионы) состоит из двух частей:

Executive Summary (для менеджмента): Выжимка на 1—2 страницы. Без технического сленга. Только суть: что плохо, чем грозит (в деньгах/репутации), насколько сложно починить.

Technical Details (для IT-отдела): Подробное описание уязвимости. Обязательно с пошаговой инструкцией воспроизведения (Steps to Reproduce), кусками кода (PoC) и чёткими рекомендациями по устранению (Remediation).

Если разработчик клиента не может воспроизвести баг по твоему отчёту — твой отчёт мусор, а ты зря потратил время заказчика.

2. Навыки ведения переговоров (Negotiation & De-escalation)

В пентесте или Red Team ты часто будешь приносить плохие новости. Ты будешь приходить к разработчикам и говорить, что их код, который они писали полгода, — решето. Ты будешь приходить к админам и показывать, что их домен захватывается за 15 минут.

Люди не любят, когда им указывают на ошибки. Включается защитная реакция: «У нас всё нормально, это вы читерите, в реальной жизни так не бывает».

Твоя задача — не самоутверждаться за их счёт («Смотрите, какие вы тупые, а я Д'Артаньян»). Твоя задача — быть союзником. Навык деэскалации конфликта — это умение сказать: «Ребята, архитектура отличная, но вот здесь есть узкое место. Давайте я покажу, как это можно эксплуатировать, и мы вместе подумаем, как это закрыть».

Специалист, который умеет находить общий язык с IT-отделом заказчика (а не воевать с ним), стоит на рынке золота. Таких людей делают Team Lead'ами и руководителями направлений.

3. Публичные выступления и личный бренд (Public Speaking & Branding)

Рынок ИБ в России в 2026 году — это большая деревня. Все всех знают. Если ты хочешь получать приватные инвайты на вкусные Bug Bounty программы или дорогие заказы на фрилансе, тебя должны знать.

Умение упаковать свой технический ресёрч в интересную статью на Хабре или в динамичный доклад на конференции (ZeroNights, OFFZONE, PHDays) — это мощнейший мультипликатор дохода.

Когда ты выступаешь, ты показываешь не только свои технические скиллы, но и способность структурировать информацию и удерживать внимание аудитории. После хорошего доклада к тебе подходят HR-ы топовых компаний с готовыми офферами, а потенциальные заказчики — с визитками.

Формула дорогого эксперта

Давай соберём это в формулу. Доход = (Технические навыки) * (Умение донести ценность) * (Количество людей, которые тебя знают)

Если у тебя гениальные технические навыки (10/10), но ты не умеешь общаться (1/10) и тебя никто не знает (1/10) — твой результат $10 * 1 * 1 = 10$. Ты будешь вечным мидлом-исполнителем.

Если у тебя хорошие (но не гениальные) технические навыки (7/10), ты умеешь писать понятные отчёты и объяснять риски (8/10), и ты регулярно публикуешь write-up'ы в своём Telegram-канале (6/10) — твой результат $7 * 8 * 6 = 336$. Ты будешь зарабатывать в десятки раз больше гения-социофоба.

Твоя задача — не стать лучшим кодером в мире. Твоя задача — стать лучшим переводчиком с языка технологий на язык бизнеса.

Практическое задание

Тренировка перевода (Техника -> Бизнес).

Открой файл «Мой хакерский трек». Создай раздел «Тренировка Soft Skills».

Возьми любую известную уязвимость (например, SQL-инъекцию, XSS или ту, в которой ты разбираешься лучше всего).

Напиши три абзаца об этой уязвимости для трёх разных аудиторий:

Абзац 1 (Для коллеги-хакера): Опиши уязвимость с использованием максимального количества технического сленга (пейлоады, байпассы, WAF, векторы).

Абзац 2 (Для разработчика): Объясни, *почему* в коде возникла эта уязвимость (отсутствие санитизации, неправильное использование ORM) и как именно её исправить на уровне кода. Без эмоций, только факты.

Абзац 3 (Для Директора/Бизнеса): Опиши уязвимость *вообще без технических терминов*. Используй только понятия: «риск», «деньги», «репутация», «штрафы», «остановка бизнес-процессов».

Прочитай Абзац 3 вслух. Если твоя мама или бабушка поймёт, в чём угроза для бизнеса — ты справился. Если нет — переписывай, пока не уйдёт весь птичий язык.

Это твой первый шаг к тому, чтобы продавать свои мозги в 3 раза дороже.

Часть II: Bug Bounty — охота за головами

Глава 3. Bug Bounty с нуля до первого чека

— Как устроены программы: score, правила, выплаты

Давай забудем всё, что ты читал в восторженных статьях формата «Студент нашёл баг в Google и купил квартиру». Bug Bounty (BB) в 2026 году — это индустрия со своими жёсткими законами, корпоративной бюрократией и тысячами профессиональных игроков, которые автоматизировали 90% рутины.

Чтобы получить свой первый чек, тебе нужно перестать вести себя как хакер-энтузиаст и начать думать как QA-инженер на сдельщине. Первый шаг к этому — умение читать и анализировать условия программы (Policy) до того, как ты запустил хотя бы один сканер.

Каждая публичная или приватная программа Bug Bounty состоит из трёх фундаментальных блоков: Score (рамки дозволенного), Rules (правила игры) и Rewards (деньги). Давай препарируем каждый из них.

1. Score (Скоуп): Где твои деньги, а где уголовный кодекс

Score — это самое важное слово в словаре багхантера. Это чётко очерченная территория, на которой тебе официально разрешено ломать. Всё, что находится за пределами скоупа (Out of Score), ломать **категорически запрещено**.

Если ты найдёшь RCE (удалённое выполнение кода) на сервере, который не указан в скоупе, в лучшем случае тебе скажут «Спасибо, но денег не дадим», а в худшем — безопасники компании воспримут это как реальную атаку и передадут твои IP-адреса в правоохранительные органы. И никакая отмазка «я же из добрых побуждений» в суде не сработает.

Как читать Score:

In Scope (В рамках): Это список доменов (например, *.example.com), мобильных приложений (ссылки на App Store / Google Play), исходных кодов на GitHub или конкретных API-эндпоинтов, за которые платят.

*Широкий скоуп (Wildcard *.domain.com):* Это рай для новичка и специалиста по автоматизации (Recon). Твоя задача — найти забытый поддомен (например, dev-test-old.example.com), который админы подняли 3 года назад и забыли обновить. Там обычно самые лёгкие баги.

Узкий скоуп (app.example.com): Сюда загоняют новые продукты или критичные системы (например, биллинг). Автоматизация здесь почти бесполезна, всё уже отсканировано до тебя тысячами людей. Здесь работают только мозги, ручной анализ бизнес-логики и глубокое понимание архитектуры.

Out of Scope (Вне рамок): Это то, куда лезть нельзя. Обычно это сторонние сервисы (например, форма поддержки на базе Zendesk, прикрученная к сайту компании), тестовые среды подрядчиков или информационные блоги на WordPress, которые не несут бизнес-рисков. Если ты потратил неделю на взлом блога, который Out of Scope — ты работал бесплатно.

Лайфхак 2026 года: Всегда проверяй, кому *реально* принадлежит IP-адрес или сервис, который ты атакуешь. Часто компании используют SaaS-решения (типа Shopify или AWS S3). Найти баг в AWS S3, который использует компания — это баг в AWS, а не в компании. Тебе за него не заплатят (если только это не мисконфигурация самого бакета).

2. Правила (Rules): Как не получить бан на платформе

Правила пишутся кровью безопасников (Blue Team), которым багхантеры роняли продакшен-сервера. Если ты нарушаешь правила, тебя банят на платформе (HackerOne, Bugcrowd, BI. ZONE, Standoff), и твоя карьера багхантера заканчивается, не успев начаться.

Типичные «Красные флаги» в правилах:

Запрет на DoS / DDoS (Отказ в обслуживании). Тебе запрещено проверять, упадёт ли сервер, если отправить на него 10 миллионов запросов в секунду. Даже если ты уверен, что упадёт — тестировать это нельзя. Использование агрессивных фаззеров (ffuf, dirb) с сотнями потоков без настройки Rate Limiting (ограничения скорости) часто приравнивается к DDoS. Настраивай свои тулзы на 5—10 запросов в секунду.

Запрет на Социальную инженерию и Фишинг. Ты тестируешь *железо и код*, а не людей. Нельзя звонить в техподдержку и выманивать пароли. Нельзя отправлять письма сотрудникам.

Запрет на физический доступ. Нельзя приходить в офис компании с флешкой.

Правила работы с чужими данными (Data Privacy). Это критично. Если ты нашёл SQL-инъекцию (SQLi) или IDOR, позволяющий читать чужие профили — **остановись**. Не скачивай всю базу данных (Дамп). Создай *два своих* тестовых аккаунта (Attacker и Victim) и покажи, как Attacker ворует данные Victim. Если ты сольёшь реальные данные пользователей — это статья 272 УК РФ, даже в рамках Bug Bounty.

Safe Harbor (Безопасная гавань). Ищи эту строчку в политике. Она означает, что если ты действовал строго по правилам, компания гарантирует, что не будет преследовать тебя юридически. Если Safe Harbor нет — ты работаешь на свой страх и риск.

3. Выплаты (Rewards): За что реально платят

Платят не за сложность взлома. Платят за **Impact (Влияние на бизнес)**.

Ты можешь написать гениальный эксплойт обхода WAF в 100 строк кода, чтобы сделать красивый alert (1) (Reflected XSS) на странице поиска, которой никто не пользуется. И получить \$50 (или статус Informative). А можешь просто заметить, что при регистрации нового пользователя параметр «role»: «user» в JSON-запросе ничем не защищён, поменять его в Burp Suite на «role»: «admin», получить права администратора (Massive Privilege Escalation) и забрать \$10 000. Трудозатраты — 2 минуты. Импакт — тотальная компрометация системы.

Матрица выплат строится по шкале CVSS (Common Vulnerability Scoring System) или кастомной шкале угрозы:

Low (Низкая угроза): Мелкие недочёты, утечка некритичной информации (например, версии сервера в заголовках), Open Redirect (если он не ведёт к краже токенов), простенькие XSS без возможности захвата сессии.

Чек: 5 000 — 15 000 рублей (или \$50 — \$200).

Medium (Средняя угроза): Уязвимости, требующие сложных условий для эксплуатации, или затрагивающие только одного пользователя. Например, CSRF (Cross-Site Request Forgery) на некритичные действия, IDOR на чтение не самых важных данных.

Чек: 15 000 — 50 000 рублей (или \$200 — \$1000).

High (Высокая угроза): Серьёзные дыры. Кража сессий (Stored XSS), чтение чужих финансовых данных (IDOR), SSRF (без возможности чтения локальных файлов сервера), чтение файлов базы данных.

Чек: 50 000 — 150 000 рублей (или \$1000 — \$3000).

Critical (Критическая угроза): Джекпот. Remote Code Execution (RCE — выполнение своего кода на сервере), SQL-инъекции в критичных базах данных, обход аутентификации (Authentication Bypass), XXE (с чтением файлов), полный захват аккаунтов (Account Takeover) без взаимодействия с жертвой.

Чек: От 150 000 до 500 000+ рублей на российских платформах (и от \$5000 до \$100 000+ на западных частных программах).

Что такое N/A и Duplicate (Главная боль багхантера)

Duplicate (Дубликат): Ты нашёл баг, оформил идеальный репорт, отправил... и узнал, что кто-то другой нашёл его на 2 часа раньше тебя. Выплата достаётся первому. Тебе — ничего (иногда дают немного репутации). Это часть игры. Чтобы избежать дублей, нужно искать там, где сложно, или там, где только что выкатили новый функционал.

N/A (Not Applicable) / Informative: Уязвимость есть, но компания считает, что она не несёт бизнес-риска (Accepted Risk). Например, ты нашёл отсутствие защиты от брутфорса на форме логина, а компания говорит: «У нас мониторинг заблокирует IP через 10 попыток, мы этот риск принимаем». За это не платят. Читать раздел «Out of Scope Vulnerabilities» в правилах программы нужно *до* начала тестов, чтобы не искать то, за что заведомо не заплатят.

Bug Bounty — это не лотерея. Это навык правильной приоритизации своих усилий. Ты должен тратить время только на те программы, где скоуп позволяет развернуться, правила адекватны, а выплаты соответствуют твоему уровню экспертизы.

Практическое задание

Анализ реальной цели (Target Profiling).

Открой файл «Мой хакерский трек». Создай раздел «Bug Bounty: Разбор Политики».

Зайди на любую доступную тебе платформу (HackerOne, Bugcrowd, Standoff 365, BI.ZONE).

Выбери одну публичную программу известной компании (например, Яндекс, VK, Авито, Ozon — если они есть на платформе).

Не запуская никаких инструментов, прочитай политику программы от первой до последней строчки и выпиши в свой файл:

3 элемента In Scope: (Например, *.api.target.com, мобильное приложение Android, основной сайт).

3 элемента Out of Scope: То, куда лезть категорически запрещено (Например, blog.target.com, физическая безопасность, DDoS).

2 уязвимости, за которые НЕ заплатят: (Обычно они перечислены в разделе Exclusions. Например, Clickjacking на страницах без чувствительных действий, или отсутствие DMARC-записей).

Максимальная выплата: Какую сумму обещают за Critical?

Напиши вывод: стоит ли тебе (с твоим текущим уровнем навыков) тратить время на эту программу, или скоуп слишком узкий/сложный?

Умение читать политику экономит тебе месяцы впустую потраченного времени и убеждёт от уголовного преследования.

— Платформы 2026: HackerOne, Bugcrowd, Intigriti, Standoff 365, BI. ZONE Bug Bounty

Если ты погуглишь «как заработать на Bug Bounty», 90% статей из поисковика отправят тебя регистрироваться на HackerOne. Это отличный совет... если ты читаешь эту статью в 2021 году.

В 2026 году геополитическая реальность диктует свои правила игры. Глобальный рынок багхантинга раскололся на два лагеря. И если ты не понимаешь, как устроена эта фрагментация, ты рискуешь либо нарваться на блокировку с заморозкой честно заработанных тысяч долларов, либо упустить золотую лихорадку на локальном рынке.

Давай разберём топ платформ: куда можно идти, куда нельзя, и где сейчас самая высокая плотность денег на одну найденную уязвимость.

1. HackerOne и Bugcrowd: Закрытый клуб

HackerOne (H1) и Bugcrowd — это титаны индустрии. Там находятся программы Пентагона, Google, Apple, Tesla и десятков тысяч мировых корпораций. Выплаты за критические баги там могут легко пробивать \$100 000.

Но для граждан РФ и РБ двери туда официально закрыты с марта 2022 года. Платформы в одностороннем порядке отключили пользователей из этих стран, заморозив на их счетах сотни тысяч долларов заработанных баунти (вознаграждения были перенаправлены в благотворительные фонды).

Можно ли там работать сейчас? Технически — да, если ты используешь сложный OPSEC. Тебе потребуется:

Регистрация аккаунта на дропа (подставное лицо) с паспортом другой страны (Казахстан, Грузия, Сербия).

Использование выделенных сервисов на три буквы/VPS с чистыми IP-адресами (чтобы не спалить российскую локацию).

Приём денег на иностранные банковские счета или в крипте через посредников.

Стоит ли оно того? Для Senior-хантеров, которые уже переросли локальный рынок — возможно. Но для новичка или мидла риск того, что платформа (которая регулярно проверяет IP, отпечатки браузера и налоговые данные) заподозрит подвох и заблокирует аккаунт прямо перед выплатой в \$10 000 — неоправданно высок.

2. Intigriti и YesWeHack: Европейская альтернатива (с нюансами)

Европейские платформы (Intigriti базируется в Бельгии, YesWeHack — во Франции) менее политизированы, чем американские H1 и Bugcrowd. Они предлагают отличные программы европейских компаний (Red Bull, Lufthansa, Brussels Airlines) с хорошими чеками в евро.

Они не банили старые российские аккаунты так агрессивно, но *вывод денег* оттуда в российские банки невозможен из-за санкций. Если у тебя есть ВНЖ или счёт в зарубежном банке — это отличные платформы с более низкой конкуренцией, чем на HackerOne. Если зарубежного счёта нет — ты будешь ломать за «фантики», которые не сможешь потратить.

3. Standoff 365 Bug Bounty: Российский лидер и золотая жила 2026 года

Когда западные платформы ушли, в России образовался вакуум. Российский бигтех (Яндекс, VK, Госуслуги, банки) остался без легальных каналов для проверки безопасности тысячами хакеров. Эту нишу молниеносно заняли отечественные платформы.

Безоговорочный лидер рынка в 2026 году — **Standoff 365** (от Positive Technologies).

Почему тебе нужно быть здесь:

Масштаб и деньги: К началу 2026 года на платформе зарегистрировано более 32 000 багхантеров. Суммарные выплаты за всё время перевалили за 242 миллиона рублей, а за один только 2025 год багхантеры заработали 160 миллионов (рост почти на 50% к прошлому году).

Рекордные чеки: Забудь про миф, что в РФ платят копейки. Максимальная разовая выплата на Standoff 365 в 2025 году приблизилась к 5 миллионам рублей (4,97 млн руб.) за одну уязвимость. Средний чек за принятый валидный репорт сейчас составляет около 65 000 рублей.

Кто там есть: Практически весь крупный бизнес РФ. Госсектор (Минцифры), VK, Тинькофф, Азбука Вкуса, HeadHunter (который предлагает до 500 000 рублей за крит) и десятки других.

Легальность: Все выплаты абсолютно белые. Деньги приходят на карту российского банка, налоги (НДФЛ для самозанятых) платятся автоматически. Никаких проблем с Финмониторингом.

4. BI. ZONE Bug Bounty: Комфорт и скорость

Платформа от BI. ZONE (дочерняя структура Сбера) — главный конкурент Standoff 365. Она была запущена в 2022 году и сейчас занимает прочную вторую строчку на рынке.

Фишки платформы:

Сверхбыстрые выплаты: BI. ZONE решила главную боль хакеров — долгое ожидание денег. Через интеграцию с платформой «Консоль» (для самозанятых) деньги падают на твою обычную банковскую карту в течение 30 часов после подтверждения уязвимости вендором. На западных платформах иногда приходится ждать месяц.

Интересный скоуп: Кроме самого Сбера и BI. ZONE, здесь много среднего и крупного бизнеса, ритейла и промышленных компаний, чья инфраструктура часто тестировалась хуже, чем у классических IT-гигантов.

Триаж (Triage): У BI. ZONE отличная команда аналитиков, которые проверяют твои репорты перед отправкой клиенту. Если ты написал что-то непонятно, они помогут докрутить репорт, а не просто влепят статус N/A.

5. Частные программы (Self-Hosted)

Не все компании хотят платить комиссию платформам (которая составляет 20—30% сверху от суммы баунти). Многие крупные игроки (например, Яндекс со своей программой «Охота за ошибками» или Ozon) держат свои программы на собственных сайтах.

Плюсы: Вся сумма выплаты идёт тебе, нет конкуренции с толпой сканеров, которые мониторят платформы через API. **Минусы:** Тебе придётся подписывать договор с каждой компанией отдельно, а в случае спора (если компания скажет «это не баг») у тебя не будет платформы-арбитра, которая могла бы защитить твои интересы.

Стратегия багхантера в 2026 году

Твоя стратегия должна быть прагматичной:

Если ты живёшь в РФ/РБ и не хочешь играть в шпиона: Иди на Standoff 365 и BI. ZONE. Рынок горячий, бюджеты на импортозамещение огромные, вендоры отечественного ПО (Astra Linux, МойОфис и т.д.) сейчас активно покупают услуги багхантеров, чтобы доказать свою надёжность.

Регистрируйся как самозанятый: Это займёт 10 минут, но это обязательное условие для получения легальных выплат на российских платформах.

Не распыляйся: Не нужно регистрироваться на 10 программах сразу. Выбери одну компанию на Standoff или BI. ZONE с широким скоупом, и потрать на неё минимум месяц. Успех в ВВ — это не широта охвата, это глубина погружения.

Практическое задание

Легализация и выбор первой цели.

Открой файл «Мой хакерский трек». Создай раздел «Мой Bug Bounty Старт».

Шаг 1: Зайди на сайты bugbounty.standoff365.com и bugbounty.bi.zone. Зарегистрируй аккаунты.

Шаг 2: Если у тебя ещё нет статуса самозанятого (НПД) — скачай приложение «Мой налог» и зарегистрируйся. Это бесплатно и не обязывает тебя платить налоги, пока у тебя нет дохода. Без этого ты не сможешь вывести заработанные деньги. Запиши в файл: «Статус самозанятого: Оформлен».

Шаг 3: Открой список программ на обеих платформах. Найди **две программы**, которые существуют больше года, но где выплаты за Critical уязвимость составляют не менее 150 000 рублей.

Запиши названия этих компаний в файл. Это твои мишени на ближайший месяц.

Никакого взлома Пентагона через прокси. Мы работаем там, где платят реальные рубли прямо на твою карту без риска сесть в тюрьму.

— Методология поиска: разведка, перебор, эксплуатация, репорт

Bug Bounty — это не творчество. Это завод. А на заводе должен быть конвейер. Если ты пытаешься найти RCE (Remote Code Execution) до того, как понял, какие технологии использует сервер, ты пытаешься собрать машину, не имея чертежей.

Профессиональная охота за уязвимостями всегда делится на четыре чётких этапа: Recon (Разведка), Fuzzing/Discovery (Перебор и поиск), Exploitation (Эксплуатация) и Reporting (Отчётность). Давай разберём каждый этап так, чтобы ты перестал тратить время впустую.

Этап 1. Recon (Разведка): Кто владеет информацией, тот владеет деньгами

Разведка — это 70% успеха в Bug Bounty, особенно на программах с широким score (например, *.target.com). Твоя задача на этом этапе — найти то, о чём забыли админы компании. Старый тестовый сервер, забытую панель администратора без пароля, открытый API-эндпоинт для мобильного приложения, которое уже удалили из AppStore.

Что мы ищем:

Субдомены (Subdomains). Чем больше субдоменов ты найдёшь, тем шире твоя поверхность атаки.

Инструменты: Amass (медленный, но находит всё), Subfinder (молниеносный), assetfinder. Не забудь про dnsx для резолва (проверки, живы ли эти домены).

IP-адреса и порты. Субдомен может вести на балансировщик нагрузки (Cloudflare), за которым прячется реальный сервер (Origin IP). Если ты найдёшь Origin IP, ты сможешь атаковать сервер напрямую, обойдя WAF (Web Application Firewall).

Инструменты: Nmap (для точечного сканирования), Masscan или Naabu (для сканирования тысяч IP за минуты). Ищи открытые порты, отличные от 80 и 443 (например, 8080, 8443, 3306 для баз данных).

Технологический стек. На чём написан сайт? PHP, Java, Ruby, Node.js? Какая версия веб-сервера (Nginx 1.14)? Используются ли старые версии фреймворков (например, уязвимый Apache Struts)?

Инструменты: Wappalizer (расширение для браузера), httpx (с флагами -tech-detect и -status-code).

Скрытые директории и файлы (Content Discovery). Поиск административных панелей (/admin), бэкапов конфигурации (.git/, config.bak., env), старых версий API (/api/v1/).

Инструменты: ffuf (самый быстрый фаззер на Go), dirsearch, gobuster. Главное здесь — не инструмент, а хороший словарь (wordlist). Используй словари от SecLists.

Лайфхак: Настрой уведомления в Telegram на появление новых поддоменов у твоей цели. Компании постоянно выкатывают новые фичи. Если ты отсканируешь новый сервер в первые 10 минут после его запуска, ты соберёшь все лёгкие баги до того, как туда придут конкуренты.

Этап 2. Перебор и поиск (Fuzzing & Discovery): Где тонко, там и рвётся

У тебя есть список живых адресов и технологий. Теперь нужно понять, как приложение реагирует на неадекватное поведение. На этом этапе в игру вступает Burp Suite (твой главный рабочий инструмент).

Что мы делаем:

Маппинг приложения (Spidering). Пройдись по всему сайту руками. Кликай на все ссылки, заполняй все формы, регистрируй аккаунты с разными ролями. Твоя задача — заставить Burp Suite записать в свою историю (HTTP History) максимальное количество уникальных запросов.

Поиск параметров (Parameter Discovery). Иногда разработчики оставляют скрытые параметры, которые включают дебаг-режим или меняют логику. Например, /user/profile?debug=true.

Инструменты: Arjun, ParamMiner (плагин для Burp).

Тестирование бизнес-логики. Это то, что не могут сделать автоматические сканеры. Задай себе вопросы:

Могу ли я добавить товар в корзину с отрицательной ценой?

Могу ли я обойти шаг оплаты в процессе оформления заказа?

Что будет, если я отправлю запрос на изменение пароля для пользователя ID=2, будучи авторизованным под ID=1 (это и есть IDOR)?

Фаззинг входов (Input Fuzzing). Отправляй спецсимволы (>>, <<, <,>, %00, \x00) в каждый параметр каждого запроса (в URL, в тело POST-запроса, в заголовки). Смотри на реакцию сервера. Упал с 500 ошибкой? Отлично, копаем глубже. Вывел твой пейлоад на страницу без фильтрации? Привет, XSS.

Этап 3. Эксплуатация (Exploitation): Превращаем ошибку в чек

Ты нашёл аномалию. Сервер вернул ошибку синтаксиса SQL при вводе кавычки. Это ещё не баг, за который платят. Это зацепка. Теперь тебе нужно доказать Impact (ущерб).

Что мы делаем:

Раскрутка уязвимости.

Была ошибка SQL? Напиши запрос, который вытащит версию базы данных (например, UNION SELECT @@version), а лучше — имя текущего пользователя БД. (Не сливай таблицы с данными пользователей, помни про правила!).

Был XSS? Напиши пейлоад, который не просто вызывает окошко alert (1), а ворует CSRF-токен или отправляет запрос на смену email-адреса жертвы.

Был SSRF? Попробуй обратиться к внутренним метаданным облака (например, http://169.254.169.254/latest/meta-data/ в AWS) и вытащить ключи доступа.

Обход защиты (Bypass). Если твой пейлоад блокирует WAF (Cloudflare, Qrator), не сдавайся сразу. Изучи техники обхода. Закодируй пейлоад (URL-encoding, Base64, Unicode), используй альтернативные HTML-теги для XSS, разбей SQL-запрос комментариями (UNI/**/ON).

Создание PoC (Proof of Concept). Это скрипт (обычно на Python) или чёткая последовательность действий в Burp Suite Repeater, которая гарантированно и без сбоев воспроизводит уязвимость. Твой PoC должен работать как швейцарские часы. Если триажер платформы не сможет его повторить, ты получишь статус N/A.

Этап 4. Отчёт (Reporting): Как продать свой взлом дорого

Это финал. Ты проделал колоссальную работу, и теперь тебе нужно её «продать» команде безопасности заказчика. Плохой отчёт может срезать выплату вдвое или вообще отправить репорт в корзину. Хороший отчёт заставляет триажера уважать тебя и быстрее подтверждать баг.

Структура идеального репорта (Золотой стандарт):

Title (Заголовок). Должен быть чётким. Формат: [Тип уязвимости] в [Эндпоинт/Функция] позволяет [Импакт].

Плохо: Нашёл баг на сайте.

Хорошо: IDOR в эндпоинте /api/v2/user_docs позволяет неавторизованному пользователю читать чужие паспорта.

Vulnerability Description (Описание уязвимости). Кратко объясни суть проблемы (2—3 предложения).

Steps to Reproduce (Шаги для воспроизведения). Это самая важная часть. Пиши так, будто объясняешь стажёру.

Шаг 1: Авторизуйтесь под пользователем А.

Шаг 2: Перехватите запрос на редактирование профиля в Burp Suite.

Шаг 3: Измените параметр user_id с 100 на 101.

Шаг 4: Отправьте запрос. Вы получите данные пользователя Б.

Proof of Concept (PoC). Приложи скриншоты (обязательно с видимым URL и реакцией сервера) или короткое видео (MP4/GIF). Приложи HTTP-запрос и ответ текстом.

Impact (Влияние на бизнес). Объясни, что будет, если этот баг найдёт хакер из дарк-нета. «Данная уязвимость позволяет злоумышленнику массово выгрузить персональные данные 100 000 клиентов, что приведёт к нарушению 152-ФЗ, оборотному штрафу и репутационным потерям для компании».

Remediation (Рекомендации по устранению). (Опционально, но сильно повышает лояльность). Напиши, как это исправить. Например: «Необходимо реализовать проверку прав доступа на уровне сервера (Access Control), проверяя, принадлежит ли запрашиваемый user_id текущей сессии».

Если ты научишься прогонять каждую цель через этот конвейер, твои шансы на стабильный заработок в Bug Bounty вырастут кратно. Ты перестанешь надеяться на удачу и начнёшь работать системно.

Практическое задание

Собери свой первый Recon-pipeline.

Открой файл «Мой хакерский трек». Создай раздел «Мой Конвейер».

Если ты работаешь в Linux (Kali/Ubuntu) или MacOS, открой терминал.

Скачай и установи три базовых инструмента от ProjectDiscovery (они написаны на Go и ставятся одной командой, если у тебя установлен язык Go):

```
go install -v github.com/projectdiscovery/subfinder/v2/cmd/subfinder@latest
go install -v github.com/projectdiscovery/httpx/cmd/httpx@latest
go install -v github.com/projectdiscovery/nuclei/v3/cmd/nuclei@latest
```

Свяжи их в один пайплайн (однострочник):

Выбери публичную программу со Standoff 365, разрешающую сканирование (например, программу самого Standoff 365 или Give Me Public). Допустим, их домен — target.com.

Запусти команду: `subfinder -d target.com -silent | httpx -silent -status-code -title | tee alive_subdomains.txt`

Эта команда найдёт поддомены, проверит, работают ли там веб-сервера, выведет код ответа и заголовок страницы, а результат сохранит в файл.

Запиши в свой трек: сколько живых поддоменов ты нашёл? Какие из них возвращают 200 OK, а какие 403 Forbidden?

Это твой первый шаг к автоматизации. В следующей главе мы будем учиться, как превращать этот сырой список в репорты на \$1000+.

Глава 4. Прокачка репутации на Bug Bounty

— Как писать репорты, чтобы не получить N/A

Давай представим рабочий день триажера (Triage Engineer) на платформе вроде Standoff 365 или BI. ZONE. Этот человек сидит на зарплате. В его очереди (queue) висит 150 новых репортов от багхантеров.

50 из них — это автоматический мусор со сканеров (отсутствие заголовка Strict-Transport-Security).

Ещё 40 — это попытки выдать фишу за баг («Я могу зарегистрировать 100 аккаунтов с одного IP! Платите мне за отсутствие Rate Limit!»).

Ещё 30 — это нечитаемый текст через Google Translate без скриншотов и внятных шагов воспроизведения.

Триажер устал. Он хочет закрыть смену и пойти пить пиво. И тут он открывает твой репорт. Твоя главная задача — сделать так, чтобы проверка твоего бага заняла у него **ровно две минуты**, после которых он с облегчением нажмёт кнопку «Triage» (Передано вендору на исправление).

Если триажер не понял твой баг за две минуты, если у него не получилось воспроизвести твой эксплойт с первого раза — он не будет разбираться. Он не будет звонить тебе и просить: «Слушай, бро, а какой пейлоад ты тут использовал?». Он просто нажмёт N/A (Not Applicable) или Needs More Info, и твой чек отложится на недели, а то и навсегда (если за это время баг найдёт кто-то другой).

Давай разберём анатомию идеального репорта, который закрывает все возражения и гарантирует выплату.

1. Title (Заголовок): Продаём с первой строчки

Заголовок — это тема письма. Он должен мгновенно отвечать на три вопроса: Что случилось? Где случилось? Чем это грозит?

Ужасный заголовок: Уязвимость на сайте, SQL injection найдена, Я могу взломать вашего пользователя. *Почему это плохо:* Триажер не понимает приоритет. Он оставит этот репорт на потом.

Идеальный заголовок: [IDOR] Уязвимость в эндпоинте /api/v2/profile позволяет неавторизованному пользователю менять email любого клиента (Account Takeover). *Почему это хорошо:* Ты сразу назвал класс уязвимости (IDOR), указал точное место (/api/v2/profile) и продал импакт (Account Takeover — полный угон аккаунта). Этот репорт триажер откроет первым, потому что это критическая угроза (P1).

2. Summary (Резюме): Для ленивых менеджеров

Summary читает не только триажер, но и менеджер со стороны компании-заказчика, который будет согласовывать выплату. Этот человек может вообще не понимать технический сленг.

В Summary (2—3 предложения) ты должен объяснить суть проблемы простым языком.

Пример: «В ходе тестирования веб-приложения app.target.com была обнаружена уязвимость типа Insecure Direct Object Reference (IDOR). Приложение не проверяет права доступа при обновлении профиля пользователя. Это позволяет любому зарегистрированному клиенту (Attacker) изменить адрес электронной почты любого другого клиента (Victim), зная только его числовой ID. После смены email атакующий может запросить восстановление пароля и получить полный контроль над чужим аккаунтом.»

Всё. Никаких дампов HTTP-запросов на этом этапе. Только чистая бизнес-логика.

3. Steps to Reproduce (Шаги воспроизведения): Защита от дурака

Это ядро твоего репорта. Если шаги написаны криво — ты получишь N/A с комментарием «Не воспроизводится» (Can't Reproduce).

Правило: Пиши шаги так, будто ты пишешь инструкцию для своей бабушки, которая первый раз открыла Burp Suite. Каждый клик, каждый перехваченный запрос, каждое изменённое значение должно быть задокументировано.

Пример идеальных шагов:

Зарегистрируйте два аккаунта на сайте target.com:

Account A (Attacker): attacker@email.com

Account B (Victim): victim@email.com (запомните его User ID, например, 1054)

Авторизуйтесь под Account B, перейдите в раздел «Настройки» и убедитесь, что профиль работает штатно.

Откройте другой браузер (или приватное окно), авторизуйтесь под Account A.

Включите перехват трафика в Burp Suite (Intercept is On).

Нажмите кнопку «Изменить Email» в профиле Account A и введите hacked@email.com.

В Burp Suite поймите POST-запрос к https://api.target.com/v1/update_email.

В теле JSON-запроса найдите параметр «user_id»: 1053 (ID Attacker) и измените его на «user_id»: 1054 (ID Victim).

Отправьте изменённый запрос (Forward).

Обратите внимание, что сервер вернул ответ 200 OK.

Обновите страницу профиля Account B — email успешно изменён на hacked@email.com.

Важно: Если твой эксплойт требует специфических настроек (например, старой версии Firefox или отключённого JavaScript), **обязательно** укажи это в шагах.

4. Proof of Concept (PoC): Бетонные доказательства

Словам не верят. Верят логам.

В блоке PoC ты должен прикрепить сырые HTTP-запросы и ответы.

Пример:

```
POST /v1/update_email HTTP/1.1 Host: api.target.com Authorization: Bearer
<Attacker-Token> Content-Type: application/json
{ «user_id»: 1054, «new_email»: "hacked@email.com" }
```

И ответ сервера:

```
HTTP/1.1 200 OK Content-Type: application/json
{ «status»: «success», «message»: «Email updated for user 1054» }
```

Скриншоты и Видео:

Прикладывай скриншоты из Burp Suite. Обведи красным маркером изменённый параметр в запросе и статус 200 OK в ответе.

Для сложных цепочек уязвимостей (Chained Bugs) всегда записывай короткое видео (до 1 минуты) без звука. Это снимает 99% вопросов у триажеров. Загрузи видео на защищённый хостинг (или прикрепи прямо в репорт на платформе). Ни в коем случае не выкладывай видео на публичный YouTube!

5. Impact (Влияние на бизнес): Как выбить максимальный чек

Триажеры и вендоры часто пытаются занижить уровень критичности уязвимости (Severity), чтобы сэкономить бюджет. Например, назвать IDOR не High, а Medium, потому что «ну это же нужно знать ID пользователя, его сложно угадать».

Твоя задача в блоке Impact — доказать, что баг смертельно опасен. Переведи техническую ошибку в деньги и юридические риски.

Пример: «Данная уязвимость позволяет полностью скомпрометировать аккаунт любого клиента платформы. Поскольку User ID является последовательным целым числом (Sequential

ID), атакующий может написать простой скрипт на Python и за 10 минут изменить email у всех 50 000 зарегистрированных пользователей, перехватив их аккаунты. Это приведёт к:

Массовой краже персональных данных (нарушение 152-ФЗ и GDPR).

Полной остановке бизнес-процессов платформы.

Огромным репутационным и финансовым потерям.»

Когда безопасник компании читает такой Impact, он понимает, что не может закрыть репорт как Informative. Он обязан выдать тебе премию по максимальному тарифу.

6. Remediation (Рекомендации): Бонус в карму

Это не обязательно, но если ты напишешь, как исправить баг, вендор может накинуть тебе бонус (Bounty Bonus) просто за сэкономленное время разработчиков.

Пример: «Не доверяйте параметру `user_id`, передаваемому со стороны клиента (Client-Side). Для обновления данных профиля берите ID пользователя исключительно из валидного JWT-токена (Server-Side сессии).»

Почему репорты всё равно получают N/A? (Чеклист самопроверки)

Прежде чем нажать кнопку «Submit», проверь свой репорт по этому списку:

Баг действительно в Score? Точно ли домен `blog.target.com` разрешён к тестированию?

Это реальная уязвимость, а не фича? Отсутствие лимита на количество символов в поле «О себе» — это не баг, если оно не приводит к отказу в обслуживании сервера (DoS).

Требуется ли взаимодействие с пользователем (User Interaction)? Если твой XSS работает, только если жертва сама вставит вредоносный код в консоль разработчика (Self-XSS) — это 100% N/A. Тебе за это не заплатят. Эксплойт должен работать без участия жертвы (или с минимальным участием, например, клик по ссылке).

Не сломал ли ты реальные данные? Если ты для демонстрации SQLi удалил (DROP) таблицу пользователей в продакшене — ты получишь не баунти, а повестку в суд. Всегда используй команду `SELECT` (чтение), а лучше `SLEEP (10)` (задержка по времени) для безопасной демонстрации.

Написание репортов — это навык продаж. Ты продаёшь свой труд платформе и компании. Упакуй его красиво, и твой ROI (возврат инвестиций) взлетит в космос.

Практическое задание

Напиши шаблон своего идеального репорта.

Открой файл «Мой хакерский трек». Создай раздел «Мой Шаблон Репорта».

Скопируй в этот раздел структуру из текста выше:

Title:

Summary:

Steps to Reproduce:

PoC (HTTP Request/Response):

Impact:

Remediation:

Главная задача: Возьми любую виртуальную уязвимость (например, SQL-инъекцию в параметре `id` на странице `target.com/product.php?id=1`).

Заполни этот шаблон для придуманной уязвимости. Пропиши выдуманные шаги, придумай фейковый HTTP-запрос со сливом версии базы данных и напиши убедительный Impact.

Сохрани этот шаблон. Когда ты найдёшь свой первый реальный баг на платформе (а в состоянии адреналина руки обычно дрожат и мысли путаются), ты просто откроешь этот файл, подставишь реальные данные и отправишь идеальный репорт за 5 минут.

Это твой первый шаг к статусу профессионала, которого уважают триажеры.

— От \$100 до \$50k: какие баги реально платят

Bug Bounty — это рынок, а на любом рынке есть свои дешёвые и дорогие товары. Стоимость уязвимости определяется одним простым словом: **Импакт (Impact)**.

Импакт — это ответ на вопрос: «Что будет с бизнесом, если этот баг найдёт и использует плохой парень из даркнета?». Если ответ «Пользователю покажется всплывающее окошко» — ты получишь свои \$100. Если ответ «Мы потеряем базу кредитных карт всех клиентов, получим оборотный штраф от Роскомнадзора и наше приложение удалят из App Store» — ты получишь \$50 000.

Проблема в том, что сканеры (Acunetix, Nessus) хорошо ищут дешёвые баги. Дорогие баги ищутся руками, потому что они завязаны на уникальную бизнес-логику конкретного приложения.

Давай разложим прайс-лист Bug Bounty 2026 года на четыре ценовые категории и посмотрим, куда тебе стоит направлять свои усилия.

Категория 1: «Сдача на проезд» (\$50 — \$300 / 5 000 — 30 000 руб.)

Это баги уровня Low или слабый Medium. Их находят тысячи новичков каждый день. Здесь самая высокая конкуренция и самый высокий процент дубликатов (Duplicate).

За что платят копейки:

* **Reflected XSS (Отражённый межсайтовый скриптинг).** Ты нашёл поле поиска, ввёл туда `<script> alert (1) </script>`, и сервер вернул этот скрипт на страницу. Круто. Но чтобы эта атака сработала, жертва должна сама кликнуть по специально сформированной (и очень подозрительной) ссылке. Импакт низкий. Платят от 5 000 до 15 000 рублей на российских платформах (BI. ZONE, Standoff 365).

Open Redirect (Открытое перенаправление). Ты нашёл параметр `?url=`, который позволяет перекинуть пользователя с доверенного сайта `bank.com` на фишинговый сайт `evil.com`. Импакт: фишинг. Если с помощью этого редиректа нельзя украсть OAuth-токены (а чаще всего нельзя), то выплата будет минимальной.

Слабая парольная политика / Отсутствие Rate Limit. Ты можешь брутфорсить форму логина бесконечно. За это редко платят больше \$100, потому что компании считают это «принятым риском» (Accepted Risk) — у них часто стоят внешние системы защиты (WAF/Anti-DDoS), которые заблокируют тебя на 101-м запросе.

Утечка некритичной информации (Information Disclosure). Ты нашёл забытый файл `phpinfo()`, который показывает версию PHP и пути на сервере. Это полезно для разведки, но само по себе денег не стоит (или стоит очень мало).

Стратегия: Не трать время на целенаправленный поиск этих багов. Если нашёл случайно — репорти. Но не делай из поиска Reflected XSS дело всей жизни.

Категория 2: «Месячная зарплата» (\$500 — \$3 000 / 50 000 — 300 000 руб.)

Это уровень крепкого Medium и High. За этими багами охотятся мидлы. Конкуренция здесь ниже, потому что сканеры такие баги пропускают в 90% случаев.

За что платят хорошо:

Stored XSS (Хранимый скриптинг). Ты нашёл поле в профиле (например, «О себе»), куда можно вставить вредоносный JS-код. Этот код сохраняется в базе данных. Теперь *любой* пользователь, который зайдёт на твою страницу, автоматически выполнит этот код. С помощью Stored XSS можно массово воровать сессионные куки (Cookies) или CSRF-токены. Это уже серьёзно.

IDOR (Insecure Direct Object Reference) на чтение ПДн. Классика. Ты меняешь свой `user_id` в запросе на `user_id` другого человека и получаешь в ответе его имя, телефон, домашний

адрес и историю заказов. Это прямое нарушение законов о защите персональных данных. Чек стартует от 50 000 рублей.

Логические ошибки (Business Logic Flaws). Это самое вкусное. Интернет-магазин продаёт iPhone за 100 000 рублей. Ты перехватываешь запрос на оплату в Burp Suite, меняешь параметр «price»: 100000 на «price»: 1, и сервер успешно списывает с твоей карты 1 рубль, отправляя заказ в доставку. Сканер никогда не поймёт, что iPhone не может стоить 1 рубль. За такие баги платят очень щедро.

CSRF (Cross-Site Request Forgery) на критичные действия. Ты делаешь зловредную HTML-страничку. Жертва (уже авторизованная на сайте банка) заходит на твою страничку, и её браузер незаметно для неё отправляет POST-запрос на перевод денег или смену пароля.

Стратегия: Фокусируйся на бизнес-логике. Заведи два аккаунта (с разными ролями, например, User и Manager) и сравнивай, какие запросы им разрешено выполнять. Ищи IDOR'ы и обходы процессов оплаты/подписки.

Категория 3: «Покупка машины» (\$5 000 — \$15 000 / 500 000 — 1 500 000 руб.)

Это уровень Critical. Такие баги находят редко, но метко. Здесь играют Senior-хантеры.

За что платят отлично:

SQL Injection (SQLi) в продакшене. Если ты нашёл слепую (Blind) или Error-based SQL-инъекцию, которая позволяет читать базу данных пользователей (с хешами паролей) — это джекпот. Это полный компромат на систему.

IDOR на модификацию критичных данных (Account Takeover — ATO). Ты можешь поменять пароль или email *любого* пользователя, не зная его текущего пароля. Это полный угон аккаунта.

SSRF (Server-Side Request Forgery) с внутренним доступом. Ты заставляешь внешний веб-сервер компании сделать HTTP-запрос во внутреннюю сеть компании (которая недоступна из интернета). Например, к базе данных Redis (<http://127.0.0.1:6379>) или к метаданным облака AWS (<http://169.254.169.254/latest/meta-data/iam/security-credentials/>), чтобы украсть административные токены облачной инфраструктуры. За облачный SSRF платят по верхней границе вилки.

XXE (XML External Entity) с чтением файлов. Если приложение парсит XML-файлы (например, при загрузке SOAP-запросов или SVG-картинок), ты можешь заставить парсер прочитать локальные файлы на сервере (например, /etc/passwd в Linux) и вернуть их тебе в ответе.

Стратегия: Если ты видишь функционал загрузки файлов, парсинга XML/JSON, генерации PDF из HTML или функционал веб-хуков (Webhooks) — копай туда до последнего. Это самые вероятные места для Critical-багов.

Категория 4: «Изменение жизни» (\$20 000 — \$100 000+ / 2 000 000+ руб.)

Это баги, которые попадают в новости. В России чеки такого размера встречаются на частных программах или на специальных ивентах (вроде Standoff Hacks). На западных платформах это стандартная цена за критику в FAANG (Fb, Apple, Amazon, Netflix, Google).

За что платят космические деньги:

RCE (Remote Code Execution) без аутентификации. Ты отправляешь один HTTP-запрос на публичный сервер компании, и этот сервер выполняет твою команду операционной системы (например, скачивает и запускает твой шелл). Это полная капитуляция сервера. 100% Impact.

Уязвимости в цепочке поставок (Supply Chain Attacks) или CI/CD. Ты нашёл способ подменить пакет в частном pм-репозитории компании или получил доступ к её GitHub Actions. Это позволяет внедрить вредоносный код прямо в исходники продукта, который потом скачают миллионы пользователей (как это было с SolarWinds).

0-click Account Takeover в массовых сервисах. Например, уязвимость в WhatsApp или Telegram, позволяющая получить доступ к переписке жертвы, просто отправив ей специально сформированное сообщение (без необходимости жертве кликать по ссылкам).

Bypass механизмов аутентификации (SAML, OAuth, JWT). Ты нашёл способ подделывать криптографическую подпись JSON Web Token (JWT) из-за слабого секретного ключа (или уязвимости алгоритма) и теперь можешь генерировать админские токены «из воздуха».

Стратегия: Ищи RCE через небезопасную десериализацию (Java, .NET, PHP), уязвимости загрузки файлов (File Upload to RCE) и ошибки в интеграции сложных протоколов авторизации.

Главный секрет ценообразования

Ты можешь искусственно повысить стоимость своего бага (Сделать Chain — цепочку). Если ты нашёл XSS — это 10 000 рублей. Если ты нашёл SSRF, который позволяет обращаться только к локальному хосту — это 100 000 рублей. Но если ты используешь XSS на странице администратора, чтобы от его имени выполнить внутренний SSRF-запрос и получить ключи от облака — это цепочка (XSS -> SSRF -> Cloud Compromise). За неё заплатят не 10 000 рублей, а 500 000 рублей, потому что ты показал полный путь эксплуатации до самого ядра бизнеса.

Всегда задавай себе вопрос: «Что я могу сделать с этим багом дальше? Как я могу усилить урон?». Импакт — это твои деньги.

Практическое задание

Сфокусируй зрение на деньгах.

Открой файл «Мой хакерский трек». Создай раздел «Мои Целевые Уязвимости».

Проанализируй свой текущий уровень навыков (ты оценивал его в Главе 1).

Выбери две категории уязвимостей из списка выше, на которых ты сосредоточишься в ближайшие 3 месяца.

Например, если ты начинающий веб-пентестер: выбери **IDOR** (Категория 2) и **Stored XSS** (Категория 2). Они логически понятны и встречаются очень часто.

Если ты уже имеешь опыт разработки: выбери **SSRF** (Категория 3) и **Ошибки бизнес-логики** (Категория 2).

Запиши свой выбор в файл. Напиши рядом: «Я не трачу больше 15 минут на поиск Reflected XSS. Я ищу баги с чеком от 50 000 рублей».

Задание на ресёрч: Зайди на ресурс hackerone.com/hackactivity (или найди разборы багов со Standoff 365 в Telegram-каналах). Найди **три реальных отчёта** по тем уязвимостям, которые ты выбрал (например, три репорта про IDOR). Внимательно прочитай, как другие хантеры их нашли и сколько за это получили.

Это научит твой мозг замечать паттерны, которые приносят реальный профит.

— Работа с приватными программами и VIP-инвайтами

Давай будем честны: публичные программы (Public Programs) — это мясорубка. Когда Яндекс, VK или Госуслуги выкатывают свою программу в паблик на Standoff 365, в первую же минуту туда устремляются тысячи сканеров. Через час всё, что лежало на поверхности (XSS, открытые .git директории, старые CVE), уже выгребено и зарепорчено. Тебе остаётся либо ковырять бетонную стену бизнес-логики неделями, либо глотать пыль, получая бесконечные статусы «Duplicate».

Но у платформ есть другой слой — **Private Programs (Приватные программы)**.

Это программы, которых нет в публичном каталоге. Ты не найдёшь их через поиск. Ты попадёшь туда, только если платформа сама пришлёт тебе на почту заветное письмо: «*You've been invited to a private bug bounty program*».

Почему компании делают программы приватными?

Скрытность. Не все хотят публично кричать «Эй, мы платим за баги!», привлекая внимание не только этичных хакеров, но и черных шляп.

Контроль нагрузки. Запуск публичной программы часто означает DDoS-атаку от армии скрипт-кидди, которые запускают DirBuster в 1000 потоков. Приватная программа позволяет пригласить только 50—100 проверенных специалистов, которые будут работать аккуратно.

Бета-тестирование. Часто в приват выкатывают совершенно новые продукты, которые ещё не релизнуты для обычных пользователей. Это клондайк: код свежий, багов много, конкуренции ноль.

Как устроена экономика привата

На приватных программах соотношение «потраченное время / заработанные деньги» (ROI) пробивает потолок.

Нулевая конкуренция на старте. Если тебя пригласили в день запуска привата (Launch Day), ты соревнуешься не с 10 000 человек, а с 50. Вероятность получить Duplicate стремится к нулю.

Свежий Scope (Fresh Surface). Разработчики только что выкатили новый API. Там ещё никто не фаззил параметры и не проверял права доступа (IDOR). Это не выжженное поле публичной программы, которая висит 5 лет.

Высокие чеки и бонусы. Компании в привате часто платят больше, потому что они экономят на обработке мусорных репортов. Плюс, часто бывают множители (например, x1.5 за критический баг в первый месяц).

VIP-эвенты (Live Hacking Events). Высшая лига привата. Платформы (и западные, и наши вроде Standoff Hacks) регулярно проводят закрытые мероприятия. Тебе оплачивают перелёт, гостиницу в хорошем отеле (или приглашают онлайн), дают эксклюзивный доступ к инфраструктуре крупного клиента (которая недоступна из интернета), и ты ломаешь её в компании других топовых хакеров в течение нескольких дней. Выплаты там космические.

Алгоритм получения инвайтов: Как заставить платформу тебя заметить

Инвайты не раздают случайно. Платформа (вернее, алгоритмы распределения инвайтов и комьюнити-менеджеры) оценивает твой профиль по конкретным метрикам. Твоя задача — взломать этот алгоритм.

Метрика 1: Репутация (Signal / Reputation) Это главный показатель твоей адекватности. Репутация растёт, когда твои баги принимают (Resolved), и падает (иногда очень больно), когда ты шлешь мусор (N/A, Spam). *Как прокачать:* На старте забудь про деньги. Твоя цель — репутация. Найди 3—5 мелких, но 100% валидных багов на публичных программах (даже бесплатных, VDP — Vulnerability Disclosure Programs). Идеально подходят логические ошибки или обход авторизации на некритичных эндпоинтах. Оформи идеальные репорты (как мы учились в предыдущем разделе). Получи свои первые очки репутации (Signal). *Красный флаг:* Если твой Signal (отношение валидных репортов к невалидным) падает ниже определённого порога (например, больше 30% твоих репортов закрывают как N/A), система автоматически исключает тебя из списков на рассылку инвайтов. Лучше отправить один 100% баг в месяц, чем 10 спорных.

Метрика 2: Импакт (Impact / Severity) Платформам нужны люди, которые находят критику, а не только любители XSS. Если у тебя в профиле есть хотя бы один High или Critical баг (даже на бесплатной программе), твои шансы на инвайт в жирный приват удваиваются.

Алгоритм видит: «Этот парень умеет находить RCE/SQLi. Добавим его в программу к банку, им нужны спецы по базам данных».

Метрика 3: Активность (Activity) Инвайты живут недолго. Если тебя пригласили, а ты не отправил ни одного репорта (или даже не принял инвайт), в следующий раз тебя могут проигнорировать. Алгоритмы отдадут приоритет активным хантерам. *Лайфхак:* Принимай все инвайты, даже если сейчас нет времени. Найди хотя бы полчаса, чтобы поверхностно пробежаться по скоупу и скинуть простенький P4 (Low) баг (утечку версии сервера, слабый SSL-сертификат, если это разрешено). Это покажет платформе, что ты живой и вовлечённый.

Стратегия перехода в приват (Roadmap 2026)

Если ты только начинаешь, твой путь выглядит так:

Песочница (Месяц 1—2): Идёшь на публичные программы или VDP (программы без денежных выплат, только за очки репутации/мерч). Твоя единственная цель — 5 валидных репортов. Забудь про чеки, работай на зачётку.

Первые инвайты (Месяц 3—4): Как только твой Signal станет зелёным, на почту начнут падать первые приваты. Обычно это программы с узким скоупом или небольшими выплатами (от стартапов). Бери их все.

Закрепление в привате (Месяц 5—6): Ищи логические баги в этих первых приватах. Они там точно есть, потому что конкуренции мало. Получай первые хорошие чеки (High/Critical) и повышай свой Impact-рейтинг.

VIP-клуб (Полгода+): Имея в профиле зелёный Signal и несколько принятых Critical-багов, ты начнёшь получать инвайты в «жирные» программы энтерпрайз-уровня и приглашения на Live Hacking Events. На этом этапе ты можешь вообще перестать открывать вкладку «Public Programs».

Bug Bounty — это игра в долгую. Не пытайся сорвать куш в первый день. Построй репутацию, докажи платформе, что ты не спамер, и она сама принесёт тебе на блюде деньги, где лежат настоящие деньги.

Практическое задание

Аудит своего профиля.

Открой файл «Мой хакерский трек». Создай раздел «Мой Путь в Приват».

Если у тебя уже есть аккаунт на платформе (Standoff/BI. ZONE/H1):

Выпиши свои текущие метрики: сколько отправлено репортов? Сколько из них Resolved (Принято)? Сколько N/A (Отклонено)?

Посчитай свой Signal (Процент успешных репортов от общего числа). Если он ниже 50% — у тебя проблемы.

План действий: Если Signal низкий, напиши обязательство: «Следующие 3 репорта я отправляю только тогда, когда уверен в баге на 200%. Никаких догадок, только железобетонный PoC». Твоя задача — вытащить репутацию из ямы.

Если ты только начинаешь (Чистый профиль):

Зайди в список публичных программ.

Найди программу типа **VDP** (Vulnerability Disclosure Program — платят только репутацией или мерчем). Обычно там низкая конкуренция, потому что профи за бесплатно не работают.

Твоя цель: За следующий месяц найти там 2 логических бага или IDOR'a (они там точно есть). Это будет твоя инвестиция в репутацию, которая окупится инвайтами через пару месяцев. Запиши эту цель в файл.

Не гонись за деньгами на старте. Репутация — это валюта, которая покупает инвайты.

— Кейсы: реальные выплаты за SSRF, XXE, Auth Bypass в 2024—2025

Когда ты читаешь политику программы и видишь строчку «Critical — до 500 000 рублей», твой мозг рисует сцену из фильма: зелёный код бежит по экрану, ты взламываешь Пентагон за 30 секунд.

В реальности самые дорогие баги часто выглядят до смешного просто. Они прячутся не в многослойной криптографии, а в банальной человеческой лени разработчиков, которые забыли проверить права доступа или доверились пользовательскому вводу.

Давай разберём три реальных сценария (на базе публичных write-up'ов и отчётов с платформ BI. ZONE и Standoff 365 за последние два года), чтобы ты понял, куда именно нужно смотреть.

Кейс 1: Authentication Bypass (Обход аутентификации) в B2B-портале

Уязвимость: Insecure Direct Object Reference (IDOR) + Business Logic Flaw. **Импакт:** Account Takeover (Угон аккаунта) любого корпоративного клиента. **Выплата (примерная оценка рынка):** 300 000 — 450 000 рублей (Critical).

Как это было: Крупная логистическая компания выкатила новый личный кабинет для B2B-клиентов. Багхантер (назовём его Alex) начал тестировать процесс восстановления пароля.

Обычно процесс выглядит так: ты вводишь email -> сервер присылает тебе на почту секретный токен (ссылку) -> ты переходишь по ссылке и задаёшь новый пароль.

Alex перехватил запрос на сброс пароля в Burp Suite. Запрос выглядел примерно так:
POST /api/v1/password/reset HTTP/1.1 Host: b2b.target.com Content-Type: application/json
{«email»: "alex@attacker.com»}

Сервер ответил 200 OK. На почту пришла ссылка: <https://b2b.target.com/reset?token=1a2b3c4d...>

Alex начал фаззить параметры (добавлять неожиданные данные в JSON). Он попробовал добавить второй email в запрос (техника HTTP Parameter Pollution / Mass Assignment):

{«email»: "alex@attacker.com», «email»: "admin@target-client.com»}

Не сработало. Тогда он попробовал передать массив:

{«email»: ["admin@target-client.com», "alex@attacker.com»]}

Бинго! Разработчики написали кривой бэкенд на Node.js. Сервер взял *первый* email из массива (admin@target-client.com), сгенерировал для него токен сброса пароля в базе данных, а затем отправил письмо с этим токеном на *второй* email из массива (alex@attacker.com).

Alex получил на свою почту валидную ссылку для сброса пароля от чужого аккаунта. Он поменял пароль и вошёл в профиль чужой компании, получив доступ к накладным, счетам и договорам.

Урок: Сканер никогда не догадается передать массив вместо строки в поле email. Это чистый ручной ресёрч бизнес-логики.

Кейс 2: SSRF (Server-Side Request Forgery) в облачной инфраструктуре

Уязвимость: SSRF с доступом к внутренним метаданным облака (Cloud Metadata). **Импакт:** Компрометация облачного аккаунта (AWS/Yandex Cloud) + доступ к внутренним базам данных. **Выплата (примерная оценка рынка):** 400 000 — 500 000 рублей (Critical).

Как это было: Популярный сервис онлайн-образования (EdTech) запустил фичу: преподаватель мог вставить ссылку на PDF-документ или картинку с другого сайта, а сервер сервиса скачивал этот файл и сохранял у себя для предпросмотра.

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.