BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Information System Security 2020/2021

http://fit.vut.cz

The FITfather - Documentation

Matúš Liščinský, xlisci02@stud.fit.vut.cz,
Faculty of Information Technology, Brno University of Technology

1. Introduction

The aim of this project was to connect via ssh to the BIS server on a specified port and get as many secrets as possible hidden on private servers in the internal network.

To connect to the BIS server, I had to supply the received private key to an SSH client. Standard Linux built-in SSH client refused the key because the key has an invalid format. It turns out that the problem was because of CRLF, not LF line ending. By running `dos2unix <key>` I converted the private key text file from DOS/MAC format to UNIX format, and so the connection could be established.

Before revealing secrets, it was necessary to map the internal network. Using the command `ip addr` I found out the IP address of the BIS server, which was 192.168.122.107/24. Based on that I could start the network mapping using the Nmap tool[1]: `nmap -sn 192.168.122.0/24`.

```
Nmap scan report for s2(192.168.122.5)
Nmap scan report for s5(192.168.122.36)
Nmap scan report for s3(192.168.122.55)
Nmap scan report for s4(192.168.122.211)
Nmap scan report for s1(192.168.122.234)
```

Option `-sn` tells Nmap to only print out the available hosts that responded to the host discovery probes (often known as a "ping scan"). Previous releases of Nmap uses `-sP` option for this purpose.

To obtain running services on these servers, I ran Nmap with the `-Pn` option including the hostname of each server: `nmap -Pn s1 s2 s3 s4 s5`

```
Nmap scan report for s1(192.168.122.234)
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
111/tcp open  rpcbind
900/tcp open  omginitialrefs

Nmap scan report for s2(192.168.122.5)
PORT   STATE SERVICE
22/tcp open  ssh

Nmap scan report for s3(192.168.122.55)
PORT   STATE SERVICE
22/tcp open  ssh

Nmap scan report for s4(192.168.122.211)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
3306/tcp open  mysql

Nmap scan report for s5(192.168.122.36)
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
111/tcp open  rpcbind
```

2. Revealing secrets

2.1 Secret A

I decided to look at the files and directories stored in `$HOME` directory of the BIS server (bis.fit.vutbr.cz). Going over files, I came across on ssh configuration file named `config` in `.ssh` directory. The configuration file contained information about two hosts {`s1`, `s2`}: hostname (ip address) and username for SSH connection. So with the command `ssh s1` I easily connect to the `s1` server. There I found suspicious hidden directory `.secret` with the text file `cipher` and bash script `generate_secret_from_decrypted_cipher`. According to the script's code, its work was to compute SHA256 hash from the given input text file. However, it required decrypted cipher from the `cipher` file.

At first, I had to find out what cipher was used, then decrypt the cipher, and finally send the decrypted cipher to the script. To identify the cipher, online tool[2] suggested that it could be Columnar Transposition Cipher. Columnar Transposition Cipher Tool[3] and its auto solve function guessed the key `abcde` and so the result of decrypting looked like this: `RWANDANEPAL ERITREAJAMAICAPHILIPPINESARGENTINA`.

[1] nmap(1) - Linux man page

[2] Cipher Identifier and Analyzer

[3] Columnar Transposition Cipher Decoder and Encoder

I assumed the key was correct because if we add spaces in the right places of the result, we get a list of countries. The final step was to execute the script with the decrypted text as a parameter. In the script output, I got Secret A.

2.2 Secret B

Following the `Nmap` scan output, I saw that on `s1` server was also running `http` service. Hence I was trying to look at the `/var/www/html` directory, but I couldn't because of a lack of reading permissions. However, I downloaded the web content from `s1` server using `curl` tool. The web page contained a form that was using `POST` method and required some url. When I forced `curl` to print also response headers (`curl -i s1`), I found out that the website was powered by the `PHP/7.4.10`. This information was included in the X-Powered-By header.

Sending POST request by running command `curl -d "url=google.com" s1` I received an array of DNS answers (IP address, IPv6 address, and hostnames of mail servers), so it seemed like the purpose of this web page was to perform DNS lookup. All this information and internet research led me to the idea of command injection attack. I wanted to know the contents of the whole `html` folder, and running `curl -d "url=;ls" s1`, caused that the DNS lookup command was ended and the server execute also `ls` command. I received the output of the `ls` command, which was an array with two file names: `index.php` and `secret.txt`. So finally, Secret B was revealed by running `curl s1/secret.txt` command.

2.3 Secret C

I already mentioned that in SSH configuration file at the BIS server was also recorded `s2` host information. Analogously to the previous case, I connected to this server with the `ssh s2` command without having to provide a password.

On the `s2` server, I found not empty `joe` file in `/var/mail`, but unfortunately, as user `server2`, I had no read permissions to that file. The directory `/var/mail` was only a symbolic link to another directory: `/var/spool/mail`. There I saw one non-empty file: `joe`. In the context of traditional Unix mail clients, mail inbox of `USER` is included in `/var/spool/mail/$USER` file. File `joe` was then the mail inbox of user `joe` and only user `joe` could access it. Switching the user with `su joe` works without having to enter a password. Thus I acquired reading permissions to `joe` file and after opening it, I found Secret C.

2.4 Secret D

This secret was hidden on `s2` server again. In `$HOME` directory there was an executable file `secret_app`. Opening it in `vim` editor and searching for substring `Tajemstvi` I revealed Secret D.

2.5 Secret E

Looking for secrets on `s2` server I discovered SSH configuration file in `.ssh` directory, containing information about hosts `s3` and `s4`. With `ssh s3` I tried to connect to `s3` as user `joe`, but to continue I had to provide a password. I searched for the list of the most common passwords on the internet and got 20 items long lists on Wikipedia[4]. Attempt with password `password1` was successful and Secret E was hidden in file `$HOME/secret.txt`.

2.6 Secret F

Browsing the `s3` server I found a suspicious directory `/database_backup` and file `2020_dump` inside it. First lines of this file said, that it's a GDBM dump file. These files are created using `gdbm_dump`[5] utility that creates a flat file dump of a GDBM database from the source database file.

The created dump can be given as argument to the `gdbm_load`[6] utility to re-create an exact copy of the source database file. By running `gdbm_load /database_backup/2020_dump` I created source database file `secret_db.gdbm`. Opening it and searching for substring `Tajemstvi` I revealed Secret F.

2.7 Secret G

From `s2` server I ran `ssh s4` and connected to `s4` without providing a password. In the listing of the `$HOME` directory I found `libgd` directory, which I decided to explore. When I listed its contents by running `ls -a` inside it, I noticed hidden `.git` directory, what indicated that it was a git repository. I looked at the commit history with `git log --oneline` and I figured that the local master branch is ahead of the origin[7] by one commit, and this commit carried the message `Super secret commit message`. With command `git show HEAD` I wanted to find out what had changed with this last commit, but in the command output, I found Secret G.

[4] List of the most common passwords
[5] gdbm_dump (1) - Linux Man Pages
[6] gdbm_load (1) - Linux Man Pages
[7] GD Graphics (Draw) Library

2.8 Secret H

According to the Nmap results, on `s4` server were also running `http` and `mysql` services. In the directory `/var/www/html` I found file `index.php`, which I did not have read permissions to. However, using the command `curl s4` I obtained the contents of this file. There was a form for entering the user's name and password. I thought that the web page could be connected with a `mysql` database and I could use an SQL injection attack. By running `curl -d 'name=" or ""=" &password=" or ""="' s4` inspired by the web page about SQL Injection[8] I obtained the data about all users of the database and among them was Secret H. The trick is in the fact that `or ""=""` is always `true`, and thus I got the entire table that had been queried.

2.9 Secret I

Network scan results showed that there was a `ftp` service running on the `s5` server. When I tried to log in, I received an information that `s5` was running `vsFTPd` (very secure FTP daemon) `2.3.4.`, which is vulnerable to a smiley face attack.

The backdoor is initiated in response to a `:)` character combination in the username which represents a smiley face. The password does not matter at all. So I try to connect again with the username containing substring `:)` and empty password and I got the message `220 Opened port <port_number>, take a look ;).`

Then I used another terminal to connect using the given port (`ftp s5 <port_number>`) and after that, I revealed Secret I.

2.10 Secret J

The last secret was hidden on `s5` server, and the process of its revealing was not that straightforward. On `s5` was running `rpcbind` service, which is generally used for communication between processes on different network nodes. The next step was to get a list of all registered RPC programs on `s5` server, which I got by command `rpcinfo -p s5` from the BIS server. There I could see running `ypbind`[9] on port 1016. According to manual pages, `ypbind` is used for finding the server for NIS domains and maintaining the NIS binding information. The client (`s5` in our case) could get the information over RPC from `ypbind` or read the binding files.

Then I noticed, that `rpcbind` service was also running on server `s1`. Therefore, I asked for the same list as on `s5` and got the information, that there was running `ypserv`[10] on port 899. According to the fact, that `ypserv` runs only on NIS master server machine with a complete NIS database, I look to `/var/yp` directory on `s1` server. There, in the `Makefile`, I found suspicious modifications of the paths to the files from which the NIS database was built. It includes files `passwd`, `shadow` and `gshadow` in the `$HOME` directory. However, I have writing permissions only to one of them: the `shadow` file.

The `shadow` file stored encrypted password of user `bis_user`. Using `mkpasswd`[11] I generated a new password encrypted with the `sha-512` algorithm and then I replaced the originally encrypted password in the `shadow` file with a newly-generated one.

After that, I need to inform `s5` server that the password of `bis_user` had changed. In the Makefile I obtain the path to the yp binaries, that was assigned to the `YPBINDIR` variable: `/usr/lib64/yp`. From there, I run `./init -m` to build new maps. This was followed by connecting to the `s5` server: `ssh bis_user@s5` with the new password. The connection was successful and I found the Secret J in the text file `$HOME/.secret/secret.txt`.

After revealing the Secret J, I logged out of `s5` and returned the `shadow` file to its original form by following the instructions from the `shadow_backup` file.

3. Conclusion

All the secrets were successfully revealed and they are included in the `secrets.txt` file.

[8] SQL Injection
[9] ypbind(8) - Linux man page
[10] ypserv(8) - Linux man page
[11] mkpasswd(1) - Linux man page