SECURITY FOR SERVICES

COMPUTER NETWORK SECURITY

Fatih Kıyıkçı
fatihkykc@gmail.com

Monday 25th May, 2020

This article is to understand, configure and maintain the security of ftp, ssh and web servers. It includes automation of some fundamental security maintenance and enhancement methods and maintaining the security of a server, computer.

- Apache
- Open-SSH
- FTP

1 Introduction

The first step after creating any server structure, must be to set the security protocols. Unwanted access to the server could be crucial. By applying fundamental security protocols to the servers, unwanted access can be reduced to minimum and abusers can be discouraged. Applying the protocols and configuring the security manually every time can be time consuming and not trustable. Therefore there is definitely benefits in automating the security process.

2 Topics

The article includes installing, testing and configuring the mentioned servers, as well as checking the security logs, implementing automated security checkers and finding unusual activities.

2.1 (“Be familiar with the user and sudo functions”)

”sudo” is a keyword in unix based operating systems, that runs a command on root level. Acronym stands for ”super user do”. ”sudo” can run any command, mostly used for writing to system files, log files or executing scripts that must work with those files.

2.2 ”user commands”

- ls -la : list all files in the current working directory, including hidden files.
- mkdir : create a directory
- cd : change current directory
- echo : print the output of a command

2.3 ”sudo commands”

- sudo adduser : add a new user
- sudo apt install : install a new package from apt
- sudo reboot : reboot the system

2.4 (“Installing the mentioned services”)

2.4.1 ”openSSH”

OpenSSH server packages are already available in the apt repository of ubuntu. To install;

```
sudo apt install openssh-server
```

Then you need to allow the ssh server from the ubuntu firewall with;

```
sudo ufw allow ssh
```

2.4.2 ”Apache”

Just like openSSH, apache server can be installed by typing;

```
sudo apt install apache2
```

In order to accept incoming traffic, ufw needs to allow apache, by typing;

```
sudo ufw allow "Apache Full"
```

Apache is by default works on ports 80 and 443

2.4.3 ”VSFTPD”

VSFTPD is a file transfer service, it is widely used and open source, to install;

```
sudo apt install vsftpd
```

When it is installed, enable it with;

```
sudo systemctl start vsftpd
sudo systemctl enable vsftpd
```

In order to allow the ftp traffic, ufw must be configured;

```
sudo ufw allow 20/tcp
sudo ufw allow 21/tcp
```

2.5 ("Testing the services")

2.5.1 OpenSSH

To see if the OpenSSH is enabled on your machine, type;

```
sudo systemctl status ssh
```

To connect your machine and see if it is working, use ssh over LAN from any machine on your network, type;

```
ssh username@ip-address
```

2.5.2 Apache

To see if the Apache web server is active on your machine, type;

```
sudo systemctl status apache2
```

To see if Apache and OpenSSH is allowed by the ufw, type;

```
sudo ufw status
```

To see the default landing page of Apache, go to your browser and type;

```
http://localhost:80
```

You should see the Apache2 Ubuntu Default Page

2.5.3 VSFTPD

Check if the vsftpd service is active on the system;

```
sudo systemctl status vsftpd
```

Check if the firewall allows the vsftpd.

```
sudo ufw status
```

Connect to vsftpd;

```
sudo ftp your.ip.address
```

2.6 (“Configuring the services and enhancing security”)

2.6.1 ”OpenSSH”

The configuration file for openSSH is on ”/etc/ssh/sshd˙config”.Before making any changes to the configuration file, it is important to keep a copy of the file in case anything goes wrong. Let’s start by making a copy of the conf file and making it read-only.

```
sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.factory-defaults
sudo chmod a-w /etc/ssh/sshd_config.factory-defaults
```

By default, openSSH uses password authentication for the server. Since most attackers brute-force try passwords after they found an ssh server, it will be wise to change the authentication method from password to ssh Key. Open the ”etc/ssh/sshd˙config” file and change the line;

```
PasswordAuthentication no
```

After disabling password authentication, you can now access only by the public key. To generate a key pair type;

```
ssh-keygen -t rsa
```

You now have 2 files in the /.ssh hidden folder, named id˙rsa and id˙rsa.pub. in the command line, type;

```
ssh-copy-id username@your.ip.adress
```

When prompted, enter the password that you set when you are creating the key pair. As the response from terminal says, check if you can login to the machine with ssh by;

```
ssh username@your.ip.address
```

For further security the users to access the server can be specified, by adding the line to the conf file;

```
AllowUsers username1, username2
```

Since most attackers brute-force try the passwords to log in, setting up a limit can be wise. To set a limit, type in the command line;

```
sudo ufw limit ssh
```

To get more detailed log information, which can be useful since the default does not record the failed authorizations, you can increase the level of detail in the logs by changing the line in conf;

```
LogLevel VERBOSE
```

Root logins are not adviced for ssh connections, since they can do a lot more damage than a standard user. To prevent logging as root, change the line in conf;

```
PermitRootLogin no
```

OpenSSH has 2 protocols to use, protocol 1 is not as secure as protocol 2, you can change the settings to only use protocol 2, by uncommenting and changing the line;

```
Protocol 2
```

By default OpenSSH uses the port number 22. Since it is very common and known by many vulnerability scanners that there is the ssh server on port 22. To change the port number, add the following line, (you can change the number with any port that is not used.)

```
Port 173
```

2.6.2 Apache

To configure apache, head to the configuration files in "/etc/apache2/apache2.conf" First of all, most of the vulnerabilities and exploits are specific to a version. Mostly the older versions suffer from exploits. Updating apache whenever a new version is released is crucial. If for some reason, you did not updated or even if you updated your server, it is important to not give the version information to the public. One can delete this information from request header by adding the lines to the apache2.conf;

```
ServerTokens Prod
ServerSignature Off
```

By default, if there is no index page, apache shows the files and directories on its location. This can leak information about your setup, version and source code, we don't want that. To disable, find the ¡Directory¿ and Options in the conf and change "Indexes" to;

`"None"` or `"-Indexes"`

Etag HTTP response header is an identifier for a specific version of a resource, but it can leak information about child processes or the server structure through inode's. To disable it, add the following to the configuration;

```
FileETag None
```

By default, apache runs as nobody or the daemon, but changing it to a user that is specific for the server can be benefitial, it can prevent the access to other services running under same user. Change the following lines;

```
User apache
Group apache
```

Permissions for the binary and conf, is 755 by default, users can see the conf. To disable this, change the permission of the bin and conf folders;

`chmod {R 750 bin conf` In default configuration, users can override the configuration, to disable, find the ¡Directory¿ and under Options, add;

```
AllowOverride None
```

By default, apache uses some request methods that you may not need, decide the methods that you will need and disable the rest by adding the line under Directory;

```
 <LimitExcept GET POST HEAD> deny from all </LimitExcept>
```

Apache comes with enabled trace method, it can be used to steal cookie information, let's disable it by addin the line;

```
TraceEnable off
```

2.6.3 "VSFTPD"

Basic Security
The configuration file is on "/etc/vsftpd.conf" let's access and make changes;

```
sudo nano /etc/vsftpd.conf
```

In the conf file, local users are allowed and anonymous users are denid by default. To allow users to upload files uncomment the line;

```
write_enable=YES
```

Then to jail the users to their directory and prevent them from accessing outside their directory tree, uncomment the line;

```
chroot_local_user=YES
```

To use this configuration for every user, add the lines;

```
user_sub_token=$USER
local_root=/home/$USER/ftp
```

To only allow users to access the ftp server when explicitly added to the userlist, add the lines;

```
userlist_enable=YES
userlist_file=/etc/vsftpd.userlist
userlist_deny=NO
```

Then add the specific users to allow them to access the ftp server. Open the userlist and write their usernames, then restart the service. `sudo nano "/etc/vsftpd.userlist"`

```
sudo systemctl restart vsftpd
```

Configuring SSL

By default, vsftpd does not encrypt the data in transition, this can expose the user credentials to abusers. Optionally one can enable TLS (Transport Layer Security) to secure the transmission. In order to do that, first, let's create a SSL Certificate for vsftpd, you can specify the expiration date.

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/vsftpd.pem
-out /etc/ssl/private/vsftpd.pem
```

Then open the vsftpd configuration file "/etc/vsftpd.conf". Uncomment the following lines;

```
rsa_cert_file=/etc/ssl/private/vsftpd.pem
rsa_private_key_file=/etc/ssl/private/vsftpd.pem
```

To force every user the usage of SSL, change the following line to "YES";

```
ssl_enable=YES
```

Adding the following lines will prevent anonymous users from accessing even if they are using SSL, and force logins and all data transmissions to happen over SSL.

```
allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
```

To disable the reuse of the SSL, and require high encryption (128 bit or higher), add the lines;

```
require_ssl_reuse=NO
ssl_ciphers=HIGH
```

Restart your service with `sudo systemctl restart vsftpd`, Forcing SSL, will cause the ftp server to not accept your access from terminal, since terminal doesn't support SSL. Instead, you should use a FTP Client that supports SSL.

2.7 ("Checking the system logs")

2.7.1 Apache system logs

To find the apache error log location, you can check the apache2.conf file by;

```
grep ErrorLog /etc/apache2/apache2.conf
```

You can see the location on the output, you can specify the log directory by changing the variable [APACHE_LOG_DIR]. By default, all system logs are on `/var/log/apache2/`.

2.7.2 OpenSSH system logs

OpenSSH system log is at `/var/log/auth.log`. You can grep the latest logs about sshd by;

```
grep 'sshd' /var/log/auth.log
```

or you can debug in real time by;

```
tail -f -n 500 /var/log/auth.log | grep 'sshd'
```

2.7.3 VSFTPD

VSFTPD log is at `/var/log/vsftpd.log`
You can see the logs in real time by;
`tail -f -n 500 /var/log/vsfptd.log`

2.8 ("Keeping track of the critical files")

In linux, since everything is a file, the critical system files are probably more exposed compared to the other "black box" OS's (these files can be service configurations, such as apache or ssh). You may want to monitor these files and check if they are still the same. Changing these files can open the way for serious attacks, or it might effect the services, in a bad way. This is called "File Integrity Monitoring".

Mostly the working principle of File Integrity Monitoring is to get a hash for the file, store it in a file, and when you call the script for FIM, or in some scheduled time, the hashes are computed again, And it compares the hashes, if they are same, no problem, but if it's changed and you don't know who did it, there is a major problem. I wrote a very basic implementation of a FIM service, it is nowhere near a secure application, but it can demonstrate the underlying logic.

usage of the script i implemented;
After saving the .sh files, give execution permission by;
`chmod +x checkIntegrityRecord.sh`

Then run the checkIntegrityRecord script with the folder or file you want to record.
`./checkIntegrityRecord.sh /var/log/ssh`

running the checkIntegrityRecord.sh file again for the same folder, will give you the difference between the previous run and the current run.

There are opensource tools for File system integriy, one of the most common used one is "inotify-tools". You can install with;
`sudo apt install inotify-tools`
Then to watch a folder, run the command;
`sudo inotifywait -m "PATH/TO/WATCH"`
-m parameter stands for indefinite watch, meaning it will not exit the program and run continuously, so you can check the folder in real time. You can test if it is working or not by simply running;
`sudo apt update`
From another terminal and watch the output of inotifywait.

2.9 ("Checking the last connections via ssh or telnet")

In unix, one can see the last successfull logins and the ip addresses via the command;
`last --ip`

You can check the failed login attempts, in order to check if there is any abusers that are trying their way in. In the configuration part, we have talked about why we don't want to allow users

to login with a password, If for some reason, you need to validate your users with password, you can grep the failed attempts with;

```
grep "Failed password" /var/log/auth.log
```

2.10 (“Preventing users from gaining root access via ssh”)

In the ”configuration and enhancing security” step, We have disabled the root login through ssh, Which means no one can ssh as root, even if they have the private key.

Other than this, if the user that is logged in as a ”sudoer”, they can switch to root if they have the password, or they can use sudo command. To prevent, One should not add the new users to the sudoers list. To see who is on the sudoers list, run;

```
getent group sudo | cut -d: -f4
```

If there are users you accidently gave superuser priviliges, you can remove their priviliges by;

```
sudo deluser USERNAME sudo
```

2.11 (“Scripts and commands.”)

2.11.1 unusual accounts

The attached checkAcc.sh tries to catch users with no password, if it does, it locks user, then it looks for users with the user id ”0”, there are any other users than root, it alerts. usage;

```
sudo ./checkAcc.sh
```

you can check the accounts that are created by humans on your machine with;

```
awk -F: '$3 >= 1000 && $1 != "nobody" { print $1 }' /etc/passwd
```

You can check the users with uid 0 with;

```
cat /etc/passwd | awk -F: '($3 == 0) { print $1 }'
```

2.11.2 unusual log entries

You can use the command below to check for any failed logins;

```
grep "Failed password" /var/log/auth.log
```

Or you can use the attached checkLogs.sh script to look at auth.log and check for the differences in the success ip’s and failure ip’s. usage;

```
./checkLogs.sh
```

2.11.3 sluggish system performance

With the script i attached as checkMemCpu.sh, you can check the usage for cpu and memory. Give a threshold as an input, for Memory and cpu usage percentage, if the usage percentages are above the threshold, it will alert and list the top 5 consuming processes. Usage;

```
./checkMemCPU 50 80
```

Or you can check most consuming processes instantly with;

```
ps -eo pid,comm,%cpu --sort=-%cpu | head -n 5
ps -eo pid,comm,%mem --sort=-%mem | head -n 5
```

2.11.4 Excessive memory use

You can see the memory usage percentage with;

```
ps -eo pid,comm,%mem --sort=-%mem | head -n 5
```

2.11.5 Decrease in Disk space

To check the decrease in the disk space at any time, i wrote the script checkDiskDecrease.sh , making it a scheduled cron job and logging the results can be useful, or you can check the total disk space instantly, memorize and check again later, using the command;

```
df -k /
```

2.11.6 Unusual process and services

You can see the current services on your machine with;

```
service --statuall
```

To see the running processes, use the command, which will list all processes with their pid's and usage stats;

```
top
```

You can use the attached checkServices.sh to control the services, It stores the services at first run time, then compares with the current services in the second run. usage;

```
./checkServices.sh
```

2.11.7 Unusual files

One can check the unusual files with helper tools like inotify, AIDE, or the script i provided in the "2.6, keeping track of the critical files".

If a user is deleted, you can list the files with no owners by;

```
find PATH -nouser
```

or you can list files with nouser and nogroup, or both.

```
find / -nogroup -nouser
```

2.11.8 Unusual network usage

"VNSTAT" is a network tracker tool, it works as a daemon in the background and records network data, it can be help identify the anormality in the network. You can check daily, monthly network stats by running;

```
vnstat
```

If there is a lot of difference in a particular date and others, there might be someone or some script using your network.

2.11.9 Unusual scheduled tasks

Scheduled tasks can be used in unix system by cron jobs. They execute a script or a command on a recurrent time, which you can set when configuring the cron jobs. Cron jobs are one of the most popular places for "malware" scripts. Since they operate in the background and run recurrently. To prevent the cron jobs that you are not aware of, you can use file integrity monitoring the cron job files. It should check the cron jobs and report when a new one is added. inotify, AIDE, or the script i provided. Cron jobs are defined inside "/etc/", as daily, weekly or hourly. to list all cronjobs;

```
ls -lR /etc/cron*
```

3 Results

This report covers the installation, testing and configuring of openSSH, apache2, vsftpd, and very common security techniques to maintain the security of a server, or a computer. I have attached some scripts and commands which are basic demonstration implementations for security steps.

4 Conclusion

Maintaining the security of a system is not an easy task to do. Even though it is not much possible to "secure" a system completely, File integrity monitoring and keeping an eye on the system will definitely make it harder for an attacker, or a bot. If not, these can help you to be aware of the and reduce the reaction time.

References

[1] https://www.askubuntu.com.

[2] https://www.stackoverflow.com.

[3] James M. Aquilina. Malware forencics: Investigating and analyzing malicious code., 2006.

[4] Vivek Gite. Find files that do not have any owners or do not belong to any user under linux/unix. https://www.cyberciti.biz/, 2006.

[5] Silver Moon. 18 commands to monitor network bandwidth on linux server. https://www.binarytides.com/, 2014.

[6] Paul Troncone. Cybersecurity ops with bash: Attack, defend, and analyze from the command line, 2006.

[7] Jake Walters. Troubleshooting slow servers: How to check cpu, ram, and disk i/o. https://www.redhat.com/, 2019.