

### **Individual Project Report**

Student ID	B1809707
Name	Nguyen Chi Hoang Minh
Email address	minhb1809707@student.ctu.edu.vn
Class	Thursday – M01
OS	<u>Linux</u>

## **1. TITLE**

A study on security configurations should be implemented of the Linux server.

## **2. PURPOSE OF THE STUDY**

The purpose of this project is to explore and highlight basic security configurations that should be implemented to strengthen the security status of the default Linux operating system installation. This document is by no means a complete security guide for the Linux operating system; however, it outlines the basic hardening of the Linux system, so it may not be a vulnerable target. Many system administrators do not realize that the default Linux installation is vulnerable to various attacks. Therefore, this document describes the basics used to protect Linux servers and some of the most popular security and industry best practices application services that usually run on Linux servers.

This research project explored key weaknesses and default configurations that have never changed when building a production Linux server, making the server an easy target for hackers on the Internet. By following some industry best practices and adjusting some security configurations, Linux servers can be well protected. This research project explores and proposes general hardening best practices for common Linux services (such as Secure) Shell (SSH), Apache Web Server, and host-based firewall (IPTABLES) to block Connect to unnecessary ports and block bad traffic. The project also explored and outlined How open-source host-based intrusion detection and prevention tools (OSSEC) can help Raise the security, auditing and monitoring of Linux servers to a new level.

The proposed outcome of the project is to identify common mistakes and weaknesses in configuring a production Linux Server and the result of such weakness. Many businesses are compromised as a result of such common mistakes, and this project is expected to explore and suggest best practices to enhance the security posture of Linux Servers.

## **3. Contents and scope**

### **3.1 Contents**

Linux Operating System is widely used as server operating system around the world. Like other operating systems, it has its own pros and cons. Security is one of the aspects that are overlooked. Many system administrators assume that Linux itself is secure and they leave many services at the default configurations leaving the server vulnerable and making an easy target for the hackers. Therefore, this project is intended to outline common default settings for common services like SSH, Apache, and IPTABLES, etc. that should be changed and configured properly to harden the server. This project also recommends some additional configurations and installations that will help enhance the security posture of the server. This project will also cover the installation and configuration of a HIDPS (Host-Based Intrusion Detection and Prevention System) which will help in better monitoring and preventing intrusions. The project will additionally, outline best practices for enhancing the security of the Operating System and common services running on the server.

I will follow the KISS principle which says “Keep It as Simple as Possible”. It is my belief that by implementing this type of monitoring with common best practices and

changing default configurations to a more secure counterpart can highly reduce the risk of easily being compromised. This project will not address the security and configurations of all the tools and services available for Linux operating systems and or network security. This project will only cover Linux Server & Hardening Security Linux operating systems and the most common services on Linux Servers. This study alone will not make a Linux server completely secure from attacks or vulnerabilities; however it will try to point out common settings and configurations that will harden the server security.

### **3.2 Scope**

- There are various operating systems that may be used in server systems; However, this project will focus on Linux Operating System, Ubuntu 20.04 is selected as the operating system for this project.
- In the scope of this project common services include Secure Shell or SSH which is used to remotely administer a server, Apache web server used for web application services, IPTABLES which is used as host-based firewall.
- Finally, I will explore the basic installation and configuration of OSSEC, an open source and freely available Host-Based Intrusion Detection and Prevention System

## **4. Result of study**

### **4.1 Meaning of brute force attack [1]**

A brute force attack is a popular cracking method: by some accounts, brute force attacks accounted for five percent of confirmed security breaches. A brute force attack involves 'guessing' username and passwords to gain unauthorized access to a system. Brute force is a simple attack method and has a high success rate.

Some attackers use applications and scripts as brute force tools. These tools try out numerous password combinations to bypass authentication processes. In other cases, attackers try to access web applications by searching for the right session ID. Attacker motivation may include stealing information, infecting sites with malware, or disrupting service.

While some attackers still perform brute force attacks manually, today almost all brute force attacks today are performed by bots. Attackers have lists of commonly used credentials, or real user credentials, obtained via security breaches or the dark web. Bots systematically attack websites and try these lists of credentials, and notify the attacker when they gain access.

### **4.2 Meaning of SSH [2]**

SSH or Secure Shell is a network communication protocol that enables two computers to communicate (c.f http or hypertext transfer protocol, which is the protocol used to transfer hypertext such as web pages) and share data. An inherent feature of ssh is that the communication between the two computers is encrypted meaning that it is suitable for use on insecure networks.

SSH is often used to "login" and perform operations on remote computers but it may also be used for transferring data.

#### **4.3 Meaning of HIDS [3]**

An intrusion detection system is a hardware or software application that detects and alerts administrators when a malicious activity has been detected. HIDS stands for "host-based intrusion detection system". HIDS mainly focus on monitoring and analyzing log files in order to detect anomalies and unauthorized alterations based on predefined policies and a set of rules. In other words, the HIDS is as effective as the pre-established rules you've added. With a large number of stored logs, extracting meaningful information is critical to detect anomalies. The extracted information should be accurate. Therefore, ensuring the security of those logs is essential to defend against log manipulation.

#### **4.4 Meaning of Firewall [4]**

A Linux firewall is a device that inspects Network traffic ( Inbound /Outbound connections ) and makes a decision to pass or filter out the traffic. Iptables is a CLI tool for managing firewall rules on a Linux machine.

Network Security evolved with different types of Linux firewall in the era. Traditional packet-filtering firewalls deal with Routing and filtering packets ( OSI Layers 3 and 4 ), Where else NGFWs will work with additional functions as with OSI layers ( L4-L7 of OSI model ).

#### **4.5 Meaning of Web server [5]**

A web server is software and hardware that uses HTTP (Hypertext Transfer Protocol) and other protocols to respond to client requests made over the World Wide Web. The main job of a web server is to display website content through storing, processing and delivering webpages to users. Besides HTTP, web servers also support SMTP (Simple Mail Transfer Protocol) and FTP (File Transfer Protocol), used for email, file transfer and storage.

Web server hardware is connected to the internet and allows data to be exchanged with other connected devices, while web server software controls how a user accesses hosted files. The web server process is an example of the client/server model. All computers that host websites must have web server software.

Web servers are used in web hosting, or the hosting of data for websites and web-based applications -- or web applications.

#### **4.6 Meaning of SQL injection [6]**

An SQL injection is a type of cyber attack in which a hacker uses a piece of SQL (Structured Query Language) code to manipulate a database and gain access to potentially valuable information.

#### 4.7 Meaning of XSS [7]

Cross-Site Scripting (XSS) attacks occur when:

Data enters a Web application through an untrusted source, most frequently a web request.

The data is included in dynamic content that is sent to a web user without being validated for malicious content.

#### 4.8 Meaning of DDoS [8]

Distributed Network Attacks are often referred to as Distributed Denial of Service (DDoS) attacks. This type of attack takes advantage of the specific capacity limits that apply to any network resources – such as the infrastructure that enables a company's website. The DDoS attack will send multiple requests to the attacked web resource – with the aim of exceeding the website's capacity to handle multiple requests... and prevent the website from functioning correctly.

#### 4.9 Real situation of Linux server security in the world [9]

With more and more servers moving beyond the enterprise boundary and into the cloud, network protection at the host-level becomes increasingly important, as workloads need to defend themselves vs. having a perimeter around them. And remember, workloads include the applications that sit on top of Linux...it's more than just the OS.

In a recent Shodan survey, it showed that Heartbleed was still an available vulnerability on more than 180,000 servers around the world, with the majority of them in the US!



Shodan scans counting vulnerable servers show the Heartbleed bug has not been eradicated.

If you run a web server on Linux (running on at least 37 percent of the web servers out there according to W3Techs), you need protection against vulnerabilities affecting them, including Apache, Nginx, etc.

	<b>Vulnerabilities Covered in and after 2014 (approx.)</b>	<b>Before 2014 (approx.)</b>	<b>Total</b>
<b>Non-Windows OS and Core Services</b>	<b>80</b>	<b>230</b>	<b>310</b>
<b>Web Servers</b>	<b>114</b>	<b>472</b>	<b>586</b>
<b>Application Servers</b>	<b>255</b>	<b>319</b>	<b>574</b>
<b>Web Console/Management Interfaces</b>	<b>113</b>	<b>453</b>	<b>566</b>
<b>Database Servers</b>	<b>10</b>	<b>218</b>	<b>228</b>
<b>DHCP, FTP, DNS servers</b>	<b>9</b>	<b>82</b>	<b>91</b>

#### **4.10 Real situation of Linux server security in Vietnam [10]**

According to risk statistics, attacks in Vietnam, in the past month, there were 346 cases of attacks on Vietnamese websites/portals. Specifically, there are 4 cases of attack to change the interface, 77 cases of phishing attacks and 265 cases of attacks to install malicious code.

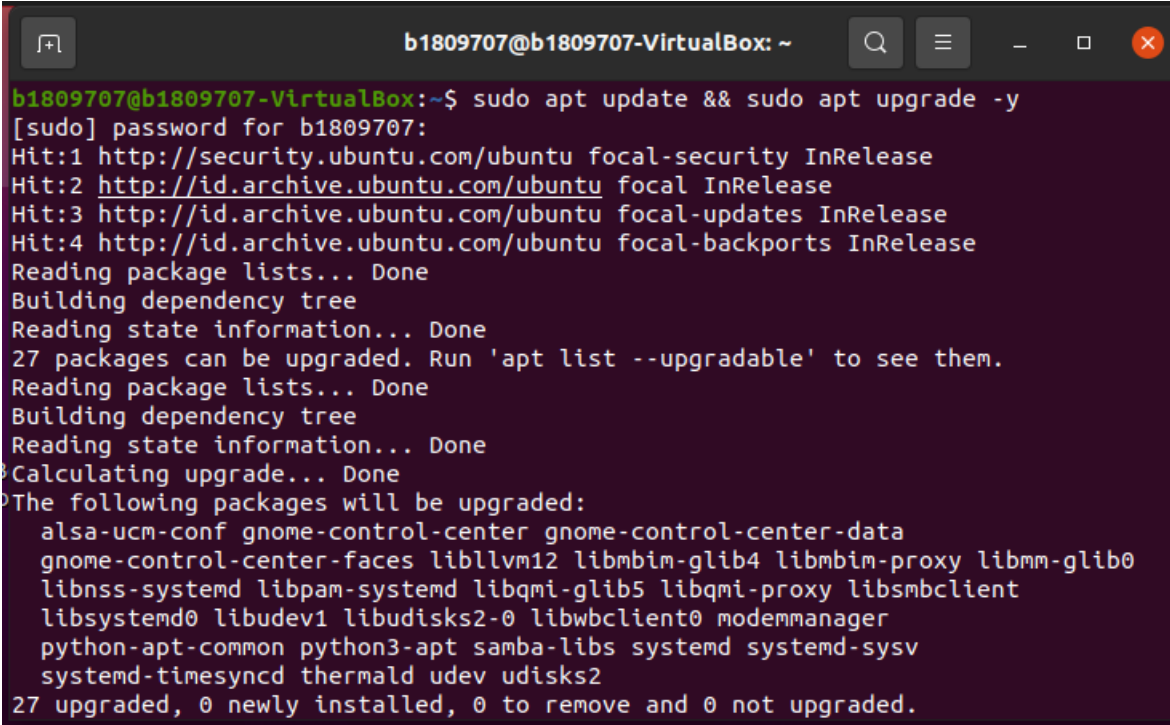
## 5. Problems and Solutions

### 5.1 Updating the System

**Problem:** Malicious activities and hacking are occurred due to the systems with vulnerabilities. Normally the hackers will find the version of the software installed in web application/server or local system and using the present vulnerabilities in that specific version to intrude into the system for doing malicious activities.

**Solution:** Make sure to update OS to the latest available version.

**sudo apt update && sudo apt upgrade -y**



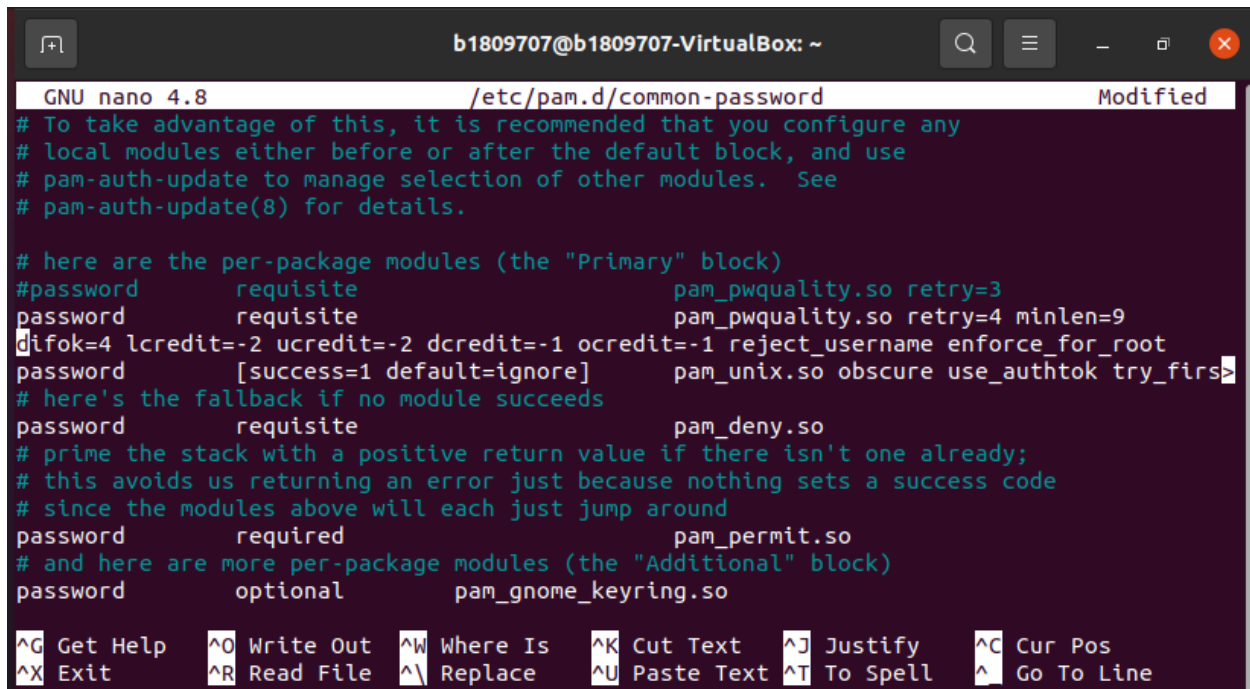
```
b1809707@b1809707-VirtualBox: ~  
b1809707@b1809707-VirtualBox:~$ sudo apt update && sudo apt upgrade -y  
[sudo] password for b1809707:  
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease  
Hit:2 http://id.archive.ubuntu.com/ubuntu focal InRelease  
Hit:3 http://id.archive.ubuntu.com/ubuntu focal-updates InRelease  
Hit:4 http://id.archive.ubuntu.com/ubuntu focal-backports InRelease  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
27 packages can be upgraded. Run 'apt list --upgradable' to see them.  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
Calculating upgrade... Done  
The following packages will be upgraded:  
  alsa-ucm-conf gnome-control-center gnome-control-center-data  
  gnome-control-center-faces libllvm12 libmbim-glib4 libmbim-proxy libmm-glib0  
  libnss-systemd libpam-systemd libqmi-glib5 libqmi-proxy libsmbclient  
  libsystemd0 libudev1 libudisks2-0 libwbclient0 modemmanager  
  python-apt-common python3-apt samba-lsfs systemd systemd-sysv  
  systemd-timesyncd thermald udev udisks2  
27 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

## 5.2 Enable and enforce secure password policies

**Problem:** As computing continued to evolve, it became even easier to guess or manipulate passwords.

**Solution:** Install and use the pwquality module of PAM.

Linux Pluggable Authentication Modules (PAM) is a suite of libraries that allows a Linux system administrator to configure methods to authenticate users.



```
GNU nano 4.8 /etc/pam.d/common-password Modified
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
password      requisite                       pam_pwquality.so retry=3
password      requisite                       pam_pwquality.so retry=4 minlen=9
difok=4 lcredit=-2 ucredit=-2 dcredit=-1 ocredit=-1 reject_username enforce_for_root
password      [success=1 default=ignore]      pam_unix.so obscure use_authtok try_first_pass
# here's the fallback if no module succeeds
password      requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
password      required                       pam_permit.so
# and here are more per-package modules (the "Additional" block)
password      optional                       pam_gnome_keyring.so

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

**password requisite pam\_pwquality.so retry=4 minlen=9 difok=4 lcredit=-2  
ucredit=-2 dcredit=-1**

**ocredit=-1 reject\_username enforce\_for\_root**

The parameters in above command mean:

1. retry: No. of consecutive times a user can enter an incorrect password.
2. minlen: Minimum length of password
3. difok: No. of character that can be similar to the old password
4. lcredit: Min No. of lowercase letters
5. ucredit: Min No. of uppercase letters
6. dcredit: Min No. of digits
7. ocredit: Min No. of symbols
8. reject\_username: Rejects the password containing the user name
9. enforce\_for\_root: Also enforce the policy for the root user



```
b1809707@b1809707-VirtualBox: ~  
b1809707@b1809707-VirtualBox:~$ sudo passwd testuser  
New password:  
BAD PASSWORD: The password contains less than 1 digits  
New password:  
BAD PASSWORD: The password contains less than 2 uppercase letters  
New password:  
BAD PASSWORD: The password contains less than 2 lowercase letters  
New password:  
BAD PASSWORD: The password contains less than 1 non-alphanumeric characters  
passwd: Have exhausted maximum number of retries for service  
passwd: password unchanged  
b1809707@b1809707-VirtualBox:~$
```

### 5.3 Securing Web Server

**Problem:** Web servers are one of the most targeted parts of an organization's network, because of the sensitive data that they typically host. As a result, it's important that as well as securing web applications and server wider network, admins take through measures to secure the web servers themselves.

The majority of web application attacks are through XSS, Info Leakage, Session Management, DDoS, DoS and SQL Injection attacks which are due to weak programming code and failure to sanitize web application infrastructure.

**Solution:** Configuring Apache server for enhanced security and Integrate "Mod Security" for Apache Server

#### *Configuring Apache server for enhanced security*

```
b1809707@b1809707-VirtualBox: /var/www/html  
GNU nano 4.8 /etc/apache2/apache2.conf Modified  
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet  
ServerSignature Off  
ServerTokens Prod
```

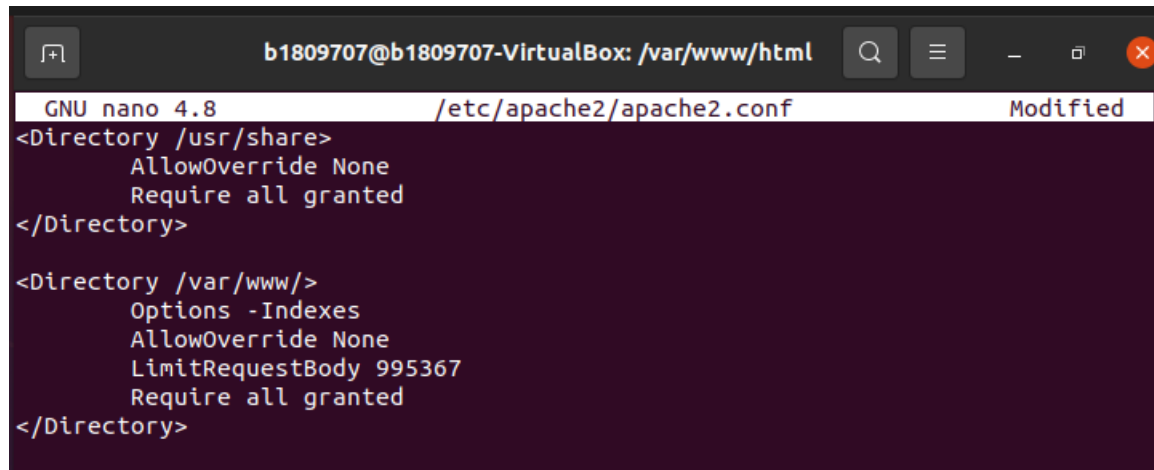
**ServerSignature Off**

**ServerTokens Prod**

The parameters in above command mean:

The default configuration of an Apache Server exposes a lot of details about the server and its settings. For example, enabled ServerSignature and ServerTokens directives in the /etc/apache2/apache2.conf file add an additional header to the HTTP Response that exposes potentially sensitive information. This information includes server setting details,

such as server version and hosting OS, that can help the attacker with the reconnaissance process. These parameters disable these directives.



```
b1809707@b1809707-VirtualBox: /var/www/html
GNU nano 4.8 /etc/apache2/apache2.conf Modified
<Directory /usr/share>
    AllowOverride None
    Require all granted
</Directory>

<Directory /var/www/>
    Options -Indexes
    AllowOverride None
    LimitRequestBody 995367
    Require all granted
</Directory>
```

## Options -Indexes

### AllowOverride None

### LimitRequestBody 995367

The parameters in above command mean:

1. Options -Indexes: The Directory listings display all content saved in the root folder or sub-directories. The directory files can include sensitive information not intended for public display, such as PHP scripts, configuration files, files containing passwords, logs, etc. This parameter will disallow directory listings.
2. AllowOverride None: The .htaccess file is a convenient and powerful feature that allows configuration outside the main apache2.conf file. However, in cases where a user can upload files to the server, this can be exploited by an attacker to upload his or her own “.htaccess” file with malicious configurations. So, this parameter disable the .htaccess directive.
3. LimitRequestBody 995367: Unlimited HTTP/HTTPS requests can also lead to low server performance or a DoS attack. This parameter limit receiving HTTP requests by using LimitRequestBody to less than 100K.

## ***Integrate “Mod Security” for Apache Server***

Host-based firewalls like iptables, UFW, and FirewallD, etc. They work on layer 3 and 4 of the OSI model and take actions based on IP address and port number. ModSecurity, or web application firewalls in general, is specialized to focus on HTTP traffic (layer 7 of the OSI model) and takes action based on the content of HTTP request and response.

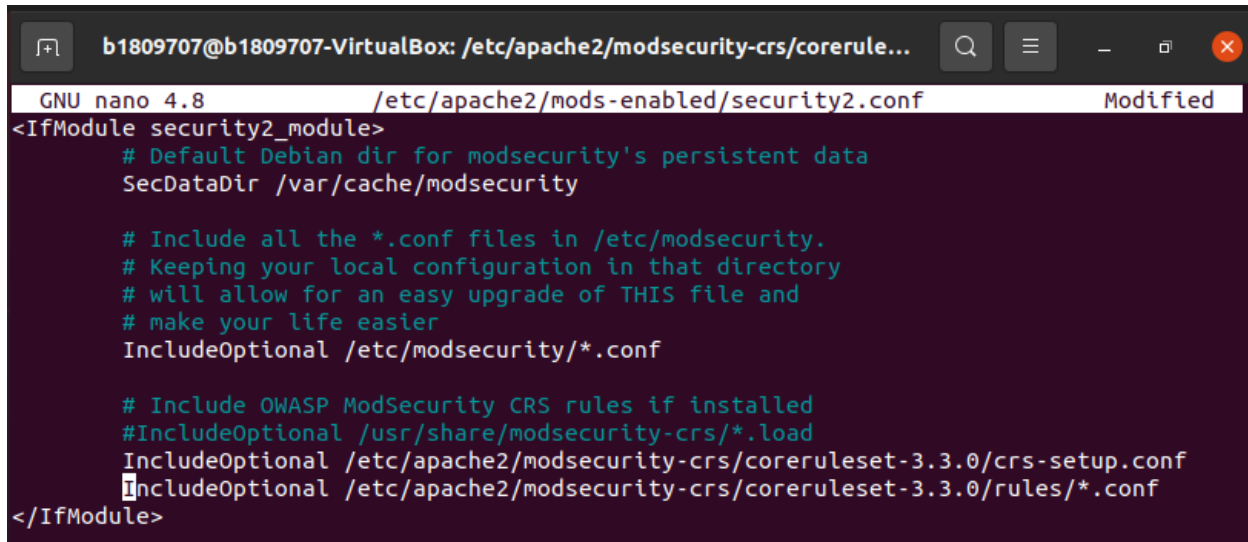
Visit <https://github.com/coreruleset/coreruleset/archive/v3.3.0.tar.gz> and download the corresponding mod\_security.

Extract the file and move the extracted directory to /etc/apache2/modsecurity-crs/

Edit the /etc/apache2/mods-enabled/security2.conf file.

**IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-3.3.0/crs-setup.conf**

**IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-3.3.0/rules/\*.conf**



```
b1809707@b1809707-VirtualBox: /etc/apache2/modsecurity-crs/corerule...
GNU nano 4.8 /etc/apache2/mods-enabled/security2.conf Modified
<IfModule security2_module>
# Default Debian dir for modsecurity's persistent data
SecDataDir /var/cache/modsecurity

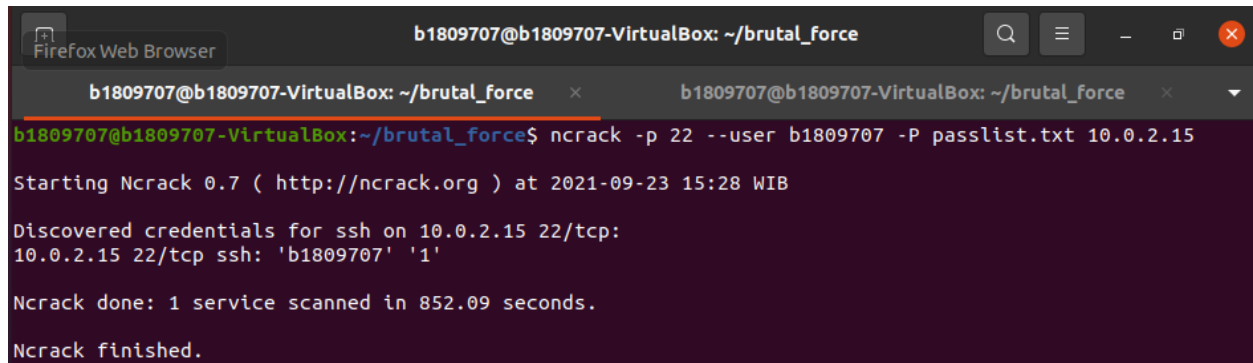
# Include all the *.conf files in /etc/modsecurity.
# Keeping your local configuration in that directory
# will allow for an easy upgrade of THIS file and
# make your life easier
IncludeOptional /etc/modsecurity/*.conf

# Include OWASP ModSecurity CRS rules if installed
#IncludeOptional /usr/share/modsecurity-crs/*.load
IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-3.3.0/crs-setup.conf
IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-3.3.0/rules/*.conf
</IfModule>
```

## 5.4 Securing SSH

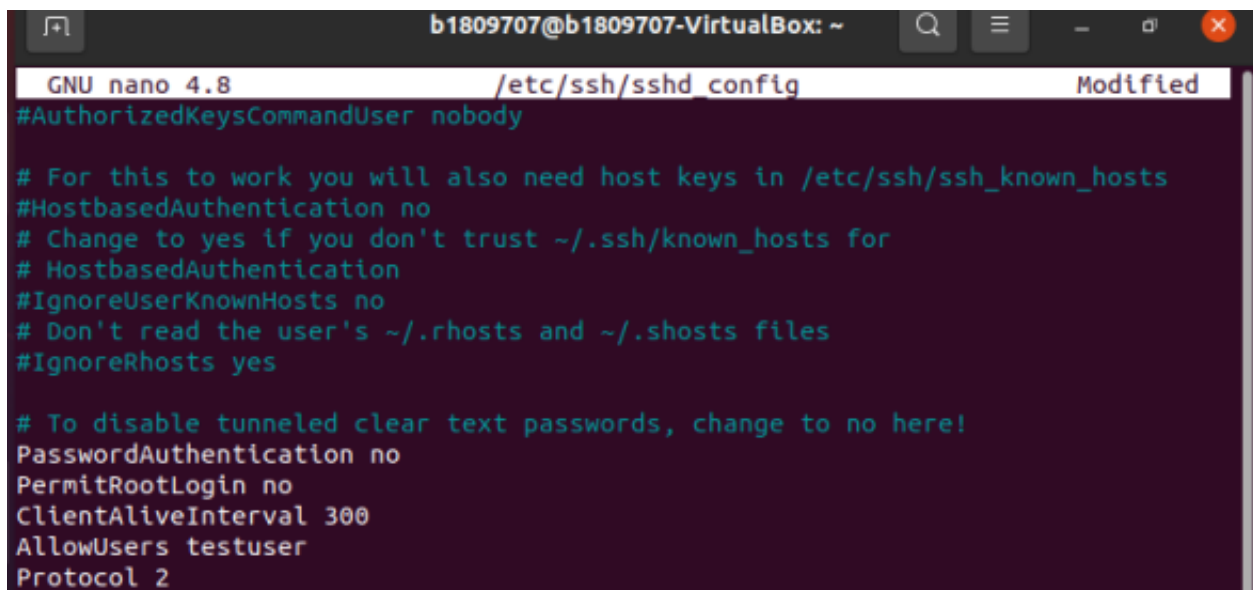
**Problem:** SSH is a valuable attack vector for hackers. One of the most reliable ways to gain SSH access to servers is by brute-forcing credentials.

Passlist source: <http://zeldor.biz/other/bruteforce/passlist.txt>



```
b1809707@b1809707-VirtualBox: ~/brutal_force
b1809707@b1809707-VirtualBox: ~/brutal_force$ ncrack -p 22 --user b1809707 -P passlist.txt 10.0.2.15
Starting Ncrack 0.7 ( http://ncrack.org ) at 2021-09-23 15:28 WIB
Discovered credentials for ssh on 10.0.2.15 22/tcp:
10.0.2.15 22/tcp ssh: 'b1809707' '1'
Ncrack done: 1 service scanned in 852.09 seconds.
Ncrack finished.
```

**Solution:** Configuring SSH service for enhanced security



```
GNU nano 4.8 /etc/ssh/sshd_config Modified
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
PermitRootLogin no
ClientAliveInterval 300
AllowUsers testuser
Protocol 2
```

**PasswordAuthentication no**

**PermitRootLogin no**

**ClientAliveInterval 300**

**AllowUsers testuser**

**Protocol 2**

The parameters in above command mean:

1. PasswordAuthentication: Instead of using passwords for authentication, SSH keys are considered more secure. Therefore, if you have generated the SSH keys for authentication, then you must disable password-based authentication.
2. PermitRootLogin: You should strictly forbid root logins for protecting any intruder from gaining root-level access to your server.
3. ClientAliveInterval: This feature is used to logout a user if he stays inactive for a long time so that no other user can gain access to his system.
4. AllowUsers: The SSH server is not a server whose access is required by every other user. Therefore, its access must be restricted only to those users who actually need it.
5. Protocol 2: implements more advanced security features, which is why it is preferred over Protocol 1.

## 5.5 Configuring Firewall

**Problem:** Without a firewall, Linux server can accepting every connection into server network from anyone. Administrators wouldn't have any way to detect incoming threats. That could leave your Linux server vulnerable to malicious users.

Not having a firewall could leave server services exposed, which could allow someone to gain control over your computer or network. Cybercriminals could delete your data. Or they could shut down server network.

**Solution:** Install IPTABLES and use IPTABLES for Linux Firewall.

Linux operating systems come equipped with a very powerful, stateful packet filtering application or a firewall known as IPTABLES.

When a packet matches a rule, it is given a target, which can be another chain or one of these special values:

ACCEPT – will allow the packet to pass through.

DROP – will not let the packet pass through.

RETURN – stops the packet from traversing through a chain and tell it to go back to the previous chain.

Meaning of three chains:

INPUT – controls incoming packets to the server.

FORWARD – filters incoming packets that will be forwarded somewhere else.

OUTPUT – filter packets that are going out from your server.

### ***Sample IPTABLES Firewall Configuration File***

***Drop all Input and Forward requests which are not allowed in this configuration file.***

```
*filter
```

```
:INPUT DROP [0:0]
```

```
:FORWARD DROP [0:0]
```

```
:OUTPUT ACCEPT [0:0]
```

```
:LogAndDrop - [0:0]
```

***Drop all incoming traffic from private networks on the public interface because they must be spoofed to arrive in public interface***

-A INPUT -i eth0 -s 10.0.0.0/8 -j DROP

-A INPUT -i eth0 -s 169.254.0.0/16 -j DROP

-A INPUT -i eth0 -s 172.16.0.0/12 -j DROP

-A INPUT -i eth0 -s 127.0.0.0/8 -j DROP

***Drop all traffic from and to multicast addresses***

-A INPUT -s 224.0.0.0/4 -j DROP

-A INPUT -d 224.0.0.0/4 -j DROP

-A INPUT -s 240.0.0.0/5 -j DROP

-A INPUT -d 240.0.0.0/5 -j DROP

-A INPUT -s 0.0.0.0/8 -j DROP

-A INPUT -d 0.0.0.0/8 -j DROP

-A INPUT -d 239.255.255.0/24 -j DROP

-A INPUT -d 255.255.255.255 -j DROP

***Drop bogus and packets with invalid states***

-A INPUT -m state --state INVALID -j DROP

-A FORWARD -m state --state INVALID -j DROP

-A OUTPUT -m state --state INVALID -j DROP

-A INPUT -p tcp -m tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

-A INPUT -p tcp -m tcp --tcp-flags SYN,RST SYN,RST -j DROP

***Accept traffic that already has established or related connection***

-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

***Accept all traffic from loopback interface***

-A INPUT -i lo -j ACCEPT

***Disable ICMP messages with additional requests***

-A INPUT -p icmp -m icmp --icmp-type address-mask-request -j DROP

-A INPUT -p icmp -m icmp --icmp-type timestamp-request -j DROP

-A INPUT -p icmp -m icmp -m limit --limit 1/second -j ACCEPT

***Accept Traffic for public Services ( Web & FTP )***

-A INPUT -p tcp -m multiport --dport 80,443 -j ACCEPT

***Allow ftp only from private 10.x network***

-A INPUT -p tcp -m multiport --dport 20,21 -s 10.0.0.0/8 -j ACCEPT

***Limit SSH attempts to 3 per minute, 4th attempt will be send to LogAndDrop Chain to log and then Drop Traffic***

-A INPUT -p tcp --dport 2222 -i eth0 -m state --state NEW -m recent --set

-A INPUT -p tcp --dport 2222 -i eth0 -m state --state NEW -m recent --update --seconds 60 --hitcount 4 -j LogAndDrop

***Any Traffic routed to this chain will be logged and then dropped***

-A LogAndDrop -m limit --limit 1/min --limit-burst 1 -j LOG --log-prefix "bruteforce"

-A LogAndDrop -j DROP

COMMIT



```
b1809707@b1809707-VirtualBox: ~/Documents
b1809707@b1809707-VirtualBox:~/Documents$ sudo iptables-restore < iptables.dump
b1809707@b1809707-VirtualBox:~/Documents$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
DROP       all  --  10.0.0.0/8             anywhere
DROP       all  --  169.254.0.0/16         anywhere
DROP       all  --  172.16.0.0/12          anywhere
DROP       all  --  localhost/8            anywhere
DROP       all  --  base-address.mcast.net/4 anywhere
DROP       all  --  anywhere               base-address.mcast.net/4
DROP       all  --  240.0.0.0/5            anywhere
DROP       all  --  anywhere               240.0.0.0/5
DROP       all  --  0.0.0.0/8              anywhere
DROP       all  --  anywhere               0.0.0.0/8
DROP       all  --  anywhere               239.255.255.0/24
DROP       all  --  anywhere               255.255.255.255
DROP       all  --  anywhere               anywhere             state INVALID
DROP       tcp  --  anywhere               anywhere               tcp flags:FIN,SYN
/FIN,SYN
DROP       tcp  --  anywhere               anywhere               tcp flags:SYN,RST
/SYN,RST
ACCEPT     tcp  --  10.0.0.0/8             anywhere               multiport dports
ftp-data,ftp
tcp  --  anywhere               anywhere               tcp dpt:2222 state
NEW recent: SET name: DEFAULT side: source mask: 255.255.255.255
LogAndDrop tcp  --  anywhere               anywhere               tcp dpt:2222 state
NEW recent: UPDATE seconds: 60 hit_count: 4 name: DEFAULT side: source mask:
255.255.255.255
```

```
s Terminal Sep 23 19:45
b1809707@b1809707-VirtualBox: ~/Documents
Chain FORWARD (policy DROP)
target     prot opt source                destination
DROP       all  --  anywhere               anywhere             state INVALID

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
DROP       all  --  anywhere               anywhere             state INVALID

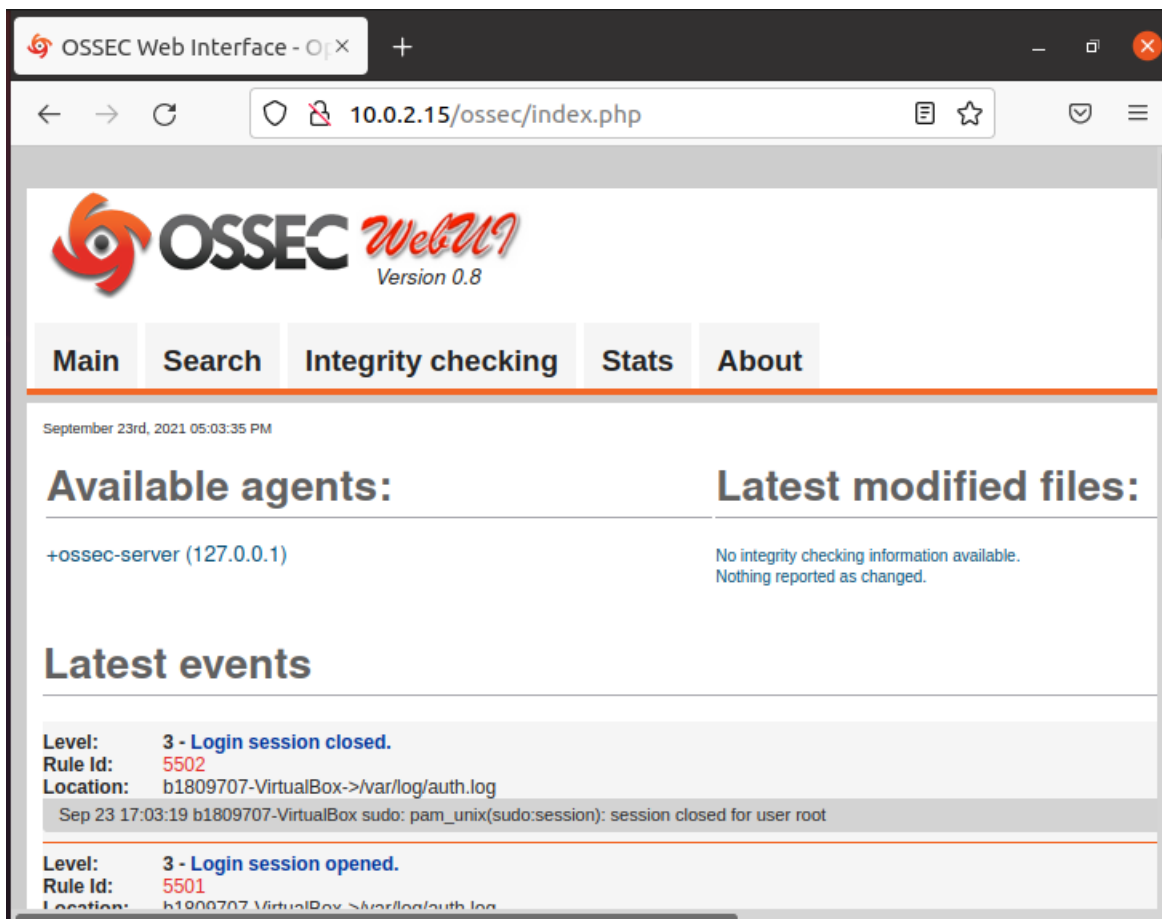
Chain LogAndDrop (1 references)
target     prot opt source                destination
LOG        all  --  anywhere               anywhere             limit: avg 1/min
burst 1 LOG level warning prefix "bruteforce"
DROP       all  --  anywhere               anywhere
```

## 5.6 Installing HIDPS

**Problem:** It's common knowledge that most applications that are running on devices and networks can and will create log messages of the activities and threats while a session is active. Administrators can collect and organize all the data created by themselves, but this will quickly become expensive from a time management perspective – that is just because of the big volume of data that admins need to keep track of.

**Solution:** Install and use OSSEC for HIDPS

OSSEC is an Open Source Host-based Intrusion Detection System that performs log analysis, file integrity checking, policy monitoring, rootkit detection, real-time alerting and active response



OSSEC Web Interface - x

10.0.2.15/ossec/index.php

**Rule Id:** 5501  
**Location:** b1809707-VirtualBox->/var/log/auth.log  
Sep 23 17:03:19 b1809707-VirtualBox sudo: pam\_unix(sudo:session): session opened for user root by (uid=0)

**Level:** 3 - Successful sudo to ROOT executed  
**Rule Id:** 5402  
**Location:** b1809707-VirtualBox->/var/log/auth.log  
**User:** b1809707  
Sep 23 17:03:19 b1809707-VirtualBox sudo: b1809707 : TTY=pts/0 ; PWD=/var/www/html/ossec ; USER=root ; COMMAND=/usr/bin/systemd

**Level:** 3 - Login session closed.  
**Rule Id:** 5502  
**Location:** b1809707-VirtualBox->/var/log/auth.log  
Sep 23 17:03:12 b1809707-VirtualBox sudo: pam\_unix(sudo:session): session closed for user root

**Level:** 3 - Login session opened.  
**Rule Id:** 5501  
**Location:** b1809707-VirtualBox->/var/log/auth.log  
Sep 23 17:03:03 b1809707-VirtualBox sudo: pam\_unix(sudo:session): session opened for user root by (uid=0)

**Level:** 3 - Successful sudo to ROOT executed  
**Rule Id:** 5402  
**Location:** b1809707-VirtualBox->/var/log/auth.log  
**User:** b1809707  
Sep 23 17:03:03 b1809707-VirtualBox sudo: b1809707 : TTY=pts/0 ; PWD=/var/www/html/ossec ; USER=root ; COMMAND=./setup.sh

**Level:** 3 - Login session closed.  
**Rule Id:** 5502  
**Location:** b1809707-VirtualBox->/var/log/auth.log  
Sep 23 16:59:53 b1809707-VirtualBox sudo: pam\_unix(sudo:session): session closed for user root

**Level:** 3 - Login session opened.

## **6. Feelings and Opinions**

In summary this project was a great exploration of a Linux Operating system, its default installation and security posture at the install time. This project covered a lot of basic installation defaults and operating system configuration changes that should be made in order to make the server production ready. The project walked through various security controls, securing common services and applications like Apache, and also provided an insight to firewall configuration and Intrusion Detection and Prevention system installation and Operation.

Even though the project is not a complete security solution to a Linux Server, it is a good starting place towards a secure server. The security issues covered by the project are the basic weaknesses which when left vulnerable makes the server an easy target and the attackers may enjoy the vulnerabilities and compromise the server there by impacting business operations, confidentiality, integrity and availability of data and information contained in the server as well as pose a threat to consumer's personal information.

Thus in summary, this project presents a good starting point for security of Linux Server and a complete real-time monitoring and intrusion detection system as well as firewall configuration to protect the server and ensure availability of services and information served and contained by the server.

## 7. Reference

- [1] <https://www.imperva.com/learn/application-security/brute-force-attack/>
- [2] <https://www.ucl.ac.uk/isd/what-ssh-and-how-do-i-use-it>
- [3] <https://logz.io/blog/open-source-hids/>
- [4] <https://cybersecuritynews.com/linux-firewall-iptables/#:~:text=A%20Linux%20firewall%20is%20a,rules%20on%20a%20Linux%20machine.>
- [5] <https://whatis.techtarget.com/definition/Web-server>
- [6] <https://www.kaspersky.com/resource-center/definitions/sql-injection>
- [7] <https://owasp.org/www-community/attacks/xss/>
- [8] <https://www.kaspersky.com/resource-center/threats/ddos-attacks>
- [9] <https://blog.trendmicro.com/linux-is-secureright/>
- [10] <https://1thegioi.vn/canh-bao-lo-hong-bao-mat-de-khai-thac-tren-he-dieu-hanh-linux-167154.html>