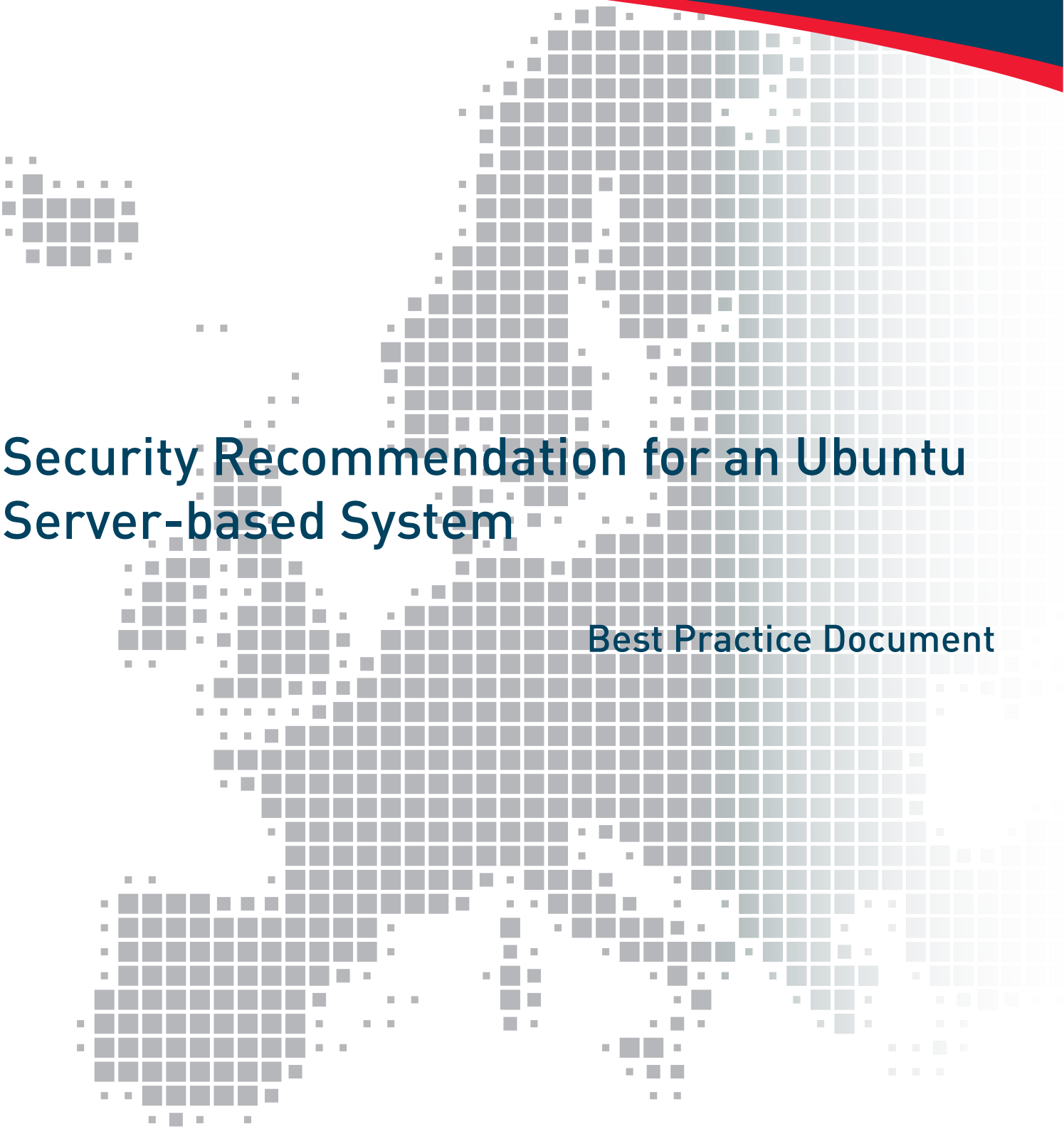
GÉANT
Networks • Services • People



Security Recommendation for an Ubuntu Server-based System

Best Practice Document

Produced by the MREN-led Campus Networking working group

Authors: Milan Čabak (MREN), Vladimir Gazivoda (MREN), Božo Krstajić (MREN)

March 2016

Document No: GN4P1-NA3-T2-MREN003
Version / date: June 2015
Original language : Montenegrin
Original title: “Sigurnosne preporuke za Ubuntu server”
Original version / date: Version 1 / 21 June 2015
Contact: Milan Čabak, milan@ac.me; Vladimir Gazivoda, vladg@ac.me; Božo Krstajić, bozok@ac.me

MREN is responsible for the contents of this document. The document was developed by the MREN-led working group on security with the purpose of implementing joint activities on the development and dissemination of documents encompassing technical guidelines and recommendations for network services in higher education and research institutions in Montenegro.

The research leading to these results has received funding from the European Union’s Horizon 2020 research and innovation programme under Grant Agreement No. 691567 (GN4-1).

Table of Contents

Table of Figures

Summary

This document describes the basic and advanced tools that provide security for the Ubuntu server from a variety of attacks and threats from the Internet.
Recommendations for increasing security during the actual installation of the Ubuntu server, as well as recommendations for creating user accounts, are contained in the document.
Furthermore, the document will present some of the most important security tools such as:
firewall, fail2ban, psad, tripwire.

Emphasis is placed on increasing the security of web and mail servers.

1 Introduction

Permanent server connection to the Internet significantly increases security risk from a variety of attacks. Attackers can cause a range of interference with the service; disabling normal functioning, collecting confidential data, or exploiting the system without the knowledge of the administrator.

The need for security tools has increased with the development of technology and the Internet. A server without basic security mechanisms can, in a very short time from the moment of connecting to the Internet, move over into the hands of attackers. All servers connected to the Internet are exposed to constant *"brute-force"* attacks in the form of scanning ports or attempts to login to the system.

The use of tools to detect and stop these attacks is of great importance for the smooth functioning of the system.

2 Increasing security when installing Ubuntu Server

When installing the Ubuntu server, with the aim of increasing its security, it is necessary to select the option *"Install security updates automatically"*. Security updates will be automatically downloaded and installed on the day of their publication by the server.
Timely installation of security updates is of great importance for increasing Ubuntu server security.

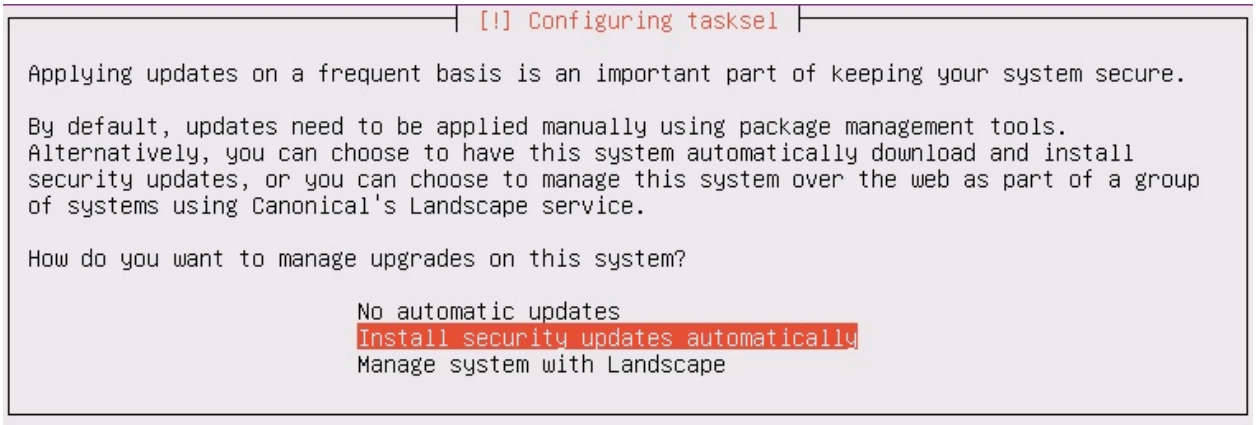


Figure 1 Automatic installation of security updates

If during the installation the default option *"No automatic updates"* is chosen, security updates will not be installed automatically by the server when they are available. However, it is possible to subsequently include the automatic installation of security updates using the following commands:
Command to install the package:
 # apt-get install unattended-upgrades
Configuration and subsequent activation of security updates:
 # dpkg-reconfigure unattended-upgrades

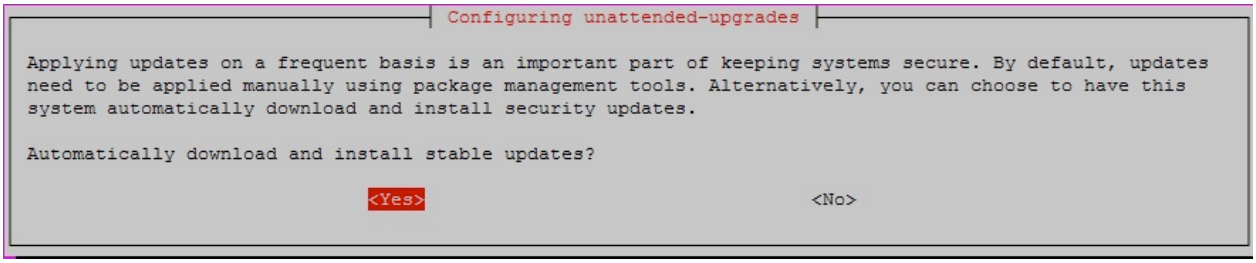


Figure 2 Activating security updates

Security updates can be installed manually, with the command:
 # unattended-upgrade

When installing the Ubuntu server, for security reasons, do not activate the *"root"* account, but use a user account that is created in one of the installation steps. If the need arises it is possible to temporarily activate the *"root"* account with the command:
 # sudo -i

It is recommended that the *"root"* account, if necessary, should be activated with the given command, i.e. that the user account should be used whenever possible. Activating and using only the *"root"* account can be a big security risk.

3 Increasing security after installing Ubuntu Server

The basic steps for increasing the security of the Ubuntu server after its installation refer to the setting of shared memory, SSH security protocol, limitations of using *'su'* command, Network Security, *RKHunter* and *CHKRootKit* tools, *AppArmor* module, and *LogWatch* system.

3.1 Shared memory

Shared memory can be used for the purpose of attacks on running services. The default tmpfs is mounted with 'rw' rights and allows the launching of the service, which can lead to exploiting flaws in web presentations. It is therefore necessary to set rights on tmpfs temporary file system to enhance its protection from attacks.

Edit the configuration file:
 # nano /etc/fstab

Add the following line at the end of the .cfg file:
Tmpfs /dev/shm tmpfs defaults,noexec,nosuid 0 0

Restart the server to make the changes accepted [7].

3.2 Setting the security of SSH protocol

The simplest way to increase the protection of SSH protocol is to disable the *root* account via SSH connection, as well as to change the standard port.

The change of the standard port is performed by modifying SSH .cfg file on the site:
 # nano /etc/ssh/sshd_config
Port < the port number is less than 1024>

After making changes, run the command:
 # /etc/init.d/ssh restart

3.3 The limitation of *su* command use

It is necessary to carry out actions which include: creating an account for administration and joining the admin group, in order to limit the processing of the *'su'* command (switching users or obtaining *'super user'* privileges) only to user accounts from *admin* group.

When installing the Ubuntu server the user account is created. It is necessary to assign the created user account to the *admin* group and to enable it to use the *'su'* command.
 # groupadd admin
 # usermod -a -G admin < user _ account >
 # dpkg-statoverride --update --add root admin 4750 /bin/su

If a user who is not a member of the admin group tries to force a change on the user or to get 'super user' privileges by using the *'su'* command, the server will considered it as an incident and such an event will be logged in the/var/mail/root file.

user@server:/root$ sudo su
[sudo] password for user:
user is not in the sudoers file. This incident will be reported.

3.4 Setting network security

As for the network, it is necessary to amend the *'sysctl'* rules that are located in the */etc/sysctl.conf* configuration file, which will help to prevent security incidents. Among other things, these rules prevent *smurf* attacks, *SYN flood* attacks, *source routed packets* problems.

In */etc/sysctl.conf* configuration file you need to apply the following rules:
 # nano /etc/sysctl.conf

Remove the comment with the following two lines to ensure *"spoof"* protection:
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1

Add the following line to protect the server from *'icmp echo' broadcast* and *multicast* requests:
net.ipv4.icmp_echo_ignore_broadcasts = 1

Remove the comment from the following two lines and disable the routing of network traffic:
net.ipv4.conf.all.accept_source_route = 0
net.ipv6.conf.all.accept_source_route = 0

In the same section add the following two lines:
net.ipv4.conf.default.accept_source_route = 0
net.ipv6.conf.default.accept_source_route = 0

Remove the comment from the following line and disable forwarding ICMP package:
net.ipv4.conf.all.send_redirects = 0

For blocking SYN attacks remove the comment from the following line:
net.ipv4.tcp_syncookies = 1

In the same section add the following configuration:
net.ipv4.tcp_max_syn_backlog = 2048
net.ipv4.tcp_synack_retries = 2
net.ipv4.tcp_syn_retries = 5

Remove the comment from the following line, and add the following to login *'martian'* package [8]:
net.ipv4.conf.all.log_martians = 1
net.ipv4.icmp_ignore_bogus_error_responses = 1

Prevent redirections of ICMP package and MITM (*man in the middle*) attacks. Remove the comment from the following two lines:
net.ipv4.conf.all.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0

To ban *ping* requests and replies, add the following line:
net.ipv4.icmp_echo_ignore_all = 1

After saving the changes to the /etc/sysctl.conf configuration file, you need to run the following command to make the change accepted.
 # sysctl -p

3.5 RKHunter and CHKRootKit tools

These two tools have the same purpose, and that is to test the system for the existence of the *rootkit* software. *Rootkit* is software designed to hide processes from the usual methods of detection, with the aim of providing privileged access to the machine.
RKHunter is a tool that compares SHA-1 *hash* values of the known files with *online* databases and detects the modified access rights, suspected strings in kernel modules, and other types of exploits. *Chkrootkit* is a shell script that uses *Linux/Unix* commands for searching the signatures of systematic applications and comparing /proc file system with the launched status of the process.

The installation of the tools is performed with the command:
 # apt-get install rkhunter chkrootkit

Launching CHKRootkit script is executed as follows:
 # chkrootkit

The result of chkrootkit command, or script, is immediately visible on the screen in the following form:

Searching for Romanian rootkit...	nothing found
Searching for Suckit rootkit...	Warning: /sbin/init INFECTED

Launching and updating the RKHunter tool is done with the following commands:
 # rkhunter --update
 # rkhunter --propupd
 # rkhunter --check

The result of the rkhunter-check command, or tool, is immediately visible on the screen in the following form:

/usr/bin/unhide.rb	[Warning]
/usr/bin/gawk	[OK]

3.6 AppArmor module

AppArmor is a MAC (*Mandatory Access Control*) system, which is an addition to the kernel with the aim of limiting the application access to resources [9]. When booting the system, AppArmor initialises profiles together with the kernel. Profiles can be either limiting or warning. Limiting profiles apply restrictive rules with notifications when detecting the actions opposing rules, while warning profiles notify of the actions opposing rules, without limiting resources.
AppArmor module is installed and launched on the Ubuntu Server. It uses profiles to determine which files and permissions are required by the application.

Additional AppArmor profiles are installed with the command:
apt-get install apparmor-profiles

Checking the status of the profile is executed with the following command:
apparmor_status

3.7 LogWatch

LogWatch is a system which analyses system logs and creates reports that account for the system administration. The installation is simple and it is necessary to type the following command:
apt-get install logwatch libdate-manip-perl

An overview of logwatch results is performed with the following command:
logwatch | less

To send logwatch reports via messages it is necessary to use the following command:
logwatch --mailto mail@doman.com --output mail --format html --range 'between -7 days and today'

4 Setting and configuration of *firewall* server

One of the fundamental steps ensuring the Ubuntu server security is firewall activation and configuration on the server. Ubuntu server has a UFW (*Uncomplicated Firewall*) firewall that simplifies working with iptables rules. Working with iptables rules can be complicated, so it is recommended to use the UFW firewall, which simplifies the procedure of defining firewall rules on the Ubuntu server.

All commands in the document will be performed with a temporary activation of the "root" account. The activation of UFW firewall is performed with the following command [1]:
 # ufw enable

The initial firewall activation is followed by adding new rules. The new rules are simply defined, and the document will specify the basic rules that are meant to raise the security of the server. The rules should be defined so that they pass only the ports necessary for the operation of the server.
Basic firewall configuration means that all communication to the server (*incoming traffic*) is prohibited until otherwise defined, and all communications from the server (*outgoing traffic*) is allowed.

If it is a Web server where you need to open port 80, this is performed with the command:
 # ufw allow 80/tcp

The range of ports that needs to be accessible from the Internet is opened with the command:
 # ufw allow proto tcp to any port 33400:33444

In a situation when the address accessing the server and the port on which the communication is received are known, the rule is activated with the following command:
 # ufw allow from 192.168.1.11 to any port 5666 proto tcp

If you need to allow the server access to a particular network on a known port, the command is as follows:
 # ufw allow from 192.168.1.0/24 to any port 22 proto tcp

Firewall rule can be inserted at an exactly defined position with the command:
 # ufw insert 1 allow from 192.168.1.0/24 to any port 22 proto tcp

Removal of the rules is performed with the command:
 # ufw delete 1

To view the entered firewall rules use the commands:
 # ufw status verbose
 # ufw status numbered

The "verbose" command shows whether the firewall is active or not, as well as the default rules for incoming and outgoing traffic, and it lists active rules. The other "numbered" command sets the serial number in front of active rules.

It is recommended to pass only the necessary ports on the server firewall, and, if known, to open communication only for networks and hosts that need to communicate with the server. Of course, firewall rules depend on the type of server. Public servers (web, mail) need to have open ports that are visible from the Internet for all networks. Public ports that are used for unobstructed legal communication might pose a security risk and an opportunity for the attacker to directly, via publicly available ports, abuse the vulnerabilities of the server.

To prevent the above mentioned type of attack, it is necessary to install additional security tools that will be described in the following chapters.

5 Installation and configuration of *Fail2Ban* tools

The server can be protected by defining access rules on the firewall; however, there are always public ports that must be accessible from the Internet. Ports that are accessible from the Internet are most likely to be attacked. Fail2ban is a tool for defence against "brute-force" attacks [2]. In case of discovering "*brute-force*" attacks Fail2Ban will block the attacker's IP address by changing firewall rules.

The attacker's IP address is added to the specially defined rules that prohibit further communication with the server.

Command for installing the Fail2Ban tool:
 # apt-get install fail2ban

Before the configuration of the Fail2Ban tool it is necessary to copy and change the name of the configuration file. This needs to be done for future updates of the Fail2Ban tool, to keep configuration changes saved.
 # cp -p /etc/fail2ban/jail.conf /etc/fail2ban/jail.local

When changing the jail.local configuration file, attention should be paid to some of the most important configuration parameters that directly affect the operation of the Fail2Ban tool.
 # nano /etc/fail2ban/jail.local

The configuration parameter, which is located in the jail.local file, ignoreip, contains the list of IP addresses or networks that will not be blocked by the Fail2ban tool, separated by spaces.
ignoreip = 127.0.0.1/8 192.168.1.10

The following configuration parameter is "bantime" which defines the time of IP address blockage in seconds. At the end of the specified period of time the IP address will be automatically removed from firewall rules on the server.
bantime = 600

Parameter findtime determines the time period monitored by Fail2ban. It is necessary to record the event within the defined period of time, in order for Fail2ban to activate the blockage.
findtime = 600

Maxretry parameter defines the number of attempts after which the host that meets the criteria will be blocked by the Fail2ban tool.
maxretry = 3

In order to send the notifications on the blockage of the Fail2ban tool to the mail, it is necessary to install the "sendmail" package.
 # apt-get install sendmail

Then it is necessary to change at least two parameters in the jail.local file. The first parameter is the address to which all future alerts will be sent:
destemail = root@localhost

The other parameter is the activation of sending warnings to the previously defined address.
action = %(action_mwl)s

The next most important configuration is the configuration of "jails" parameters. In most cases it is just enough to activate the jail policy with option "enabled = true". Each of the "jails" parameters consists of many lines that define it.

```
[ssh]
enabled = true
port    = ssh
filter  = sshd
logpath = /var/log/auth.log
maxretry = 6
```

The choice of jail functionality you need to activate will depend on the purpose of the server. The configuration file contains predefined jail configurations for http, ftp, mail, and dns servers.
For Mail server you need to include a rule that will, after six failed attempts to sign in, within five minutes block the user.

```
[dovecot]
enabled = true
port= pop3,pop3s,imap,imaps
filter = dovecot-pop3imap
logpath = /var/log/syslog
maxretry = 6
findtime = 300
```

For FTP server it is possible to activate a similar rule which will block the brute-force attempts to log on the server.

```
[proftpd]
enabled = true
port = ftp,ftp-data,ftps,ftps-data
filter = proftpd
logpath = /var/log/proftpd.log
maxretry = 6
findtime = 300
```

Checking the activity of Fail2ban filters and jail parameters is done with the command:
 # fail2ban-client status

By adding the name of the jail parameter to the previous command you get the number of events that, in this case, was registered by ssh, jail, as well as the number of successful blockades.
 # fail2ban-client status ssh

The validity of each activated jail rule is possible to check with the fail2ban-regex command which specifies the path to the log file (logpath) and the cfg file of the selected jail filter (filter).
 # fail2ban-regex /var/log/auth.log /etc/fail2ban/filter.d/sshd.conf

Successfully activated jail rule performs the modification of firewall on the server by adding the IP address of the attackers when the specified parameters (findtime and maxretry) are fulfilled. The list of blocked IP addresses is listed with the command:
 # iptables -L -n

All blocked IP addresses are listed in a separate section for activated jail rules.
Chain fail2ban-ssh (1 references)

Removing IP address from the blockade is performed with the command:
 # fail2ban-client set ssh unbanip 192.168.1.10

After any modification of the jail.local cfg file it is necessary to restart Fail2ban service, with the command:
 # service fail2ban restart

The Fail2ban tool provides adequate protection from "brute-force" attacks for famous services and ports. The next chapter will describe the tool that detects and prevents attacks on ports that are not open on the server and that are blocked by firewall.

6 Installation and configuration of psad "*intrusion detection*" system

Fail2ban, as described in the previous chapter, provides protection from attacks on known ports. Protecting the server from the "*intrusion*" attacks can be improved by using psad tools. Psad tool is an "*intrusion detection system*" that detects a variety of known attacks on the basis of "*signature based*" detection.
Installing psad tools on the Ubuntu server is simple and performed with the command [3]:
 # apt-get install psad

Psad functions by analysing firewall logs on the server. Earlier in the document, for the sake of simplicity, it was decided to use the ufw firewall. With UFW firewall there are a number of levels to log on: full, high, medium, low, off and on. For the purposes of psad tools it is best to use full level of logging activated with the command:
 # ufw logging full

With the activation of full logging UFW firewall records a large number of logs in three *locations:*
/var/log/kern.log
/var/log/syslog
/var/log/ufw.log

To avoid duplication of log records it is necessary for ufw firewall to save logs only on /var/log/ufw.log site and this is achieved by modifying the following cfg file:
 # nano /etc/rsyslog.d/20-ufw.conf

You must remove the comment in front of the line below:
& ~

Log file ufw.log can contain a large number of records, so you need to change the logrotate configuration file for the ufw firewall, with the command:
 # nano /etc/logrotate.d/ufw

The change means change in the number of records of the log files and the period for which the new log file will be created.
rotate 7
daily

After the configuration and setting up logging it is necessary to adjust psad configuration, at the following location.
 # nano /etc/psad/psad.conf

Basic configuration includes changes such as input of parameters to send notifications in the form of email messages.
EMAIL_ADDRESSES test@domain.com;

Change in the server name.
HOSTNAME psad.domain.com;

You must specify the path of the log file from which the psad tool will analyse the logs. IPT_SYSLOG_FILE/var/log/ufw.log;

Sending email notifications only for critical messages is set by modifying: EMAIL_ALERT_DANGER_LEVEL 5;

Currently psad tool acts as a detection. The function of prevention, i.e. blocking attacks, can be activated by changing the basic configuration:
ENABLE_AUTO_IDS Y;

Set up the blockade of certain attacks with high levels.
AUTO_IDS_DANGER_LEVEL 5;

The duration of the blockade, in seconds, of the IP address from which the attack was discovered, is configured by modifying:
AUTO_BLOCK_TIMEOUT 60;

To avoid accidental blockage due to false-positive notifications, you can add the IP addresses that psad will not block by modifying auto_dl file:
nano/etc/psad/auto_dl
Then add new record on file with 0 level.
127.0.0.1 0;

Working with psad tool requires using a few basic commands.

Update rules of psad tool.
psad--sig update

Restart psad service.
service psad restart

Review of psad service.
service psad status

7 Installation and configuration of tripwire server IDS package

The previous tools described in the document firewall, fail2ban, psad, serve to prevent known attacks. Tripwire tool aims to detect attacks that have managed to bypass all defence mechanisms described in the document.
Tripwire monitors changes to system files, changes in their structure, and indicates potential malicious activities that occur on the system. Each unauthorised change of configuration file will be documented.

Installation of tripwire tool is fairly simple, as with previous tools, because it is in Ubuntu repository [4]:
apt-get install tripwire

The installation process requires interaction, and in a few steps you need to confirm option <Yes> to create the site and the local keys that are used to provide the security of tripwire configuration files.

The first step after installation is to produce policies, with the command:
twadmin --create-polfile /etc/tripwire/twpol.txt

Initialising database with newly created default policy file will report as a result a large number of errors:
tripwire-init

Errors indicate a discrepancy between the created policy file and the system on which the tripwire tool is installed. Troubleshooting requires modification of the created policy file:
nano/etc/tripwire/twpol.txt

It is necessary to comment on all lines shown in the section "Error Report", which are the result of the command:
tripwire-check

if the installed psad tool is described in the document, it is necessary to comment on "/var/log" because of the constant records of various log files of the tool, and to define directories that need to be monitored. Any modification of twpol.txt file requires updating of the policy:
# *twadmin -m P /etc/tripwire/twpol.txt*

After the change of policy it is necessary to initialise tripwire tool:
tripwire –init

After initialising the tripwire tool it is necessary to change the policy and to repeat the previous steps in case of certain disagreements between the policy and the system on which it is applied.

After initialising the tripwire tool the following command checks to see if there are certain errors or changes to the system:
tripwire-check
Command for an interactive check and policy update:

tripwire-check-interactive

With the configuration of the Fail2ban tool a sendmail package is installed to send notifications in the form of email messages. With the help of sendmail package the following command sends tripwire tool report about the changes in the system to the desired address:
 # tripwire --check | mail -s "Tripwire report for `uname -n`" email@domain.com

8 Increasing the security of Mail server

A significant number of attacks on computer network takes place via e-mail. It is necessary to devote great attention to the security of mail server, and therefore of the entire network, as well as of the end users. Users often fall victim to various phishing attacks, virus and malware infections that they come into contact via infected electronic messages.

8.1 Filtering electronic messages on the basis of contents

Filtering electronic messages based on the header or the text of the message can be very effective in protecting mail server from spam and phishing messages [6].

When you check the headers and the content of electronic mail it is necessary to add the following lines to '/etc/postfix/main.cf' configuration file:
header_checks= regexp:/etc/postfix/header_checks
body_checks=regexp:/etc/postfix/body_checks

For checking headers a new file named header_checks is created in the following location:
nano/etc/postfix/header_checks

The contents of the file should be in the following format:
/^HEADER:.*text/ACTION

The header can be any option located in the header of the message, e.g.: "Received", "Subject", "From", "Date", and other fields. ". *" in front of the text indicates any combination of characters in front of the text. Action is the action that will be performed when the specified text in the message header is found.

Checking the contents of a message requires you to create a new file body_checks in the selected location:
nano/etc/postfix/body_checks

The contents of the file should be in the following format:
/text contained in the message/ACTION

Each message that contains the specified text will be prosecuted, depending on the action that is selected. The text that is specified as the object of filtering in the message is not sensitive to upper and lower case letters.

When the filter finds the specified text in the header or message content, it will apply one of the actions:

- *REJECT* - message will be rejected by postfix agent;
- *IGNORE* – the specified text will be removed, and the message will be delivered to the recipient;
- *WARN* - action will produce a notification when detecting the specified text;

- *HOLD* - message will not be delivered to the recipient and the administrator's action will be required;
- *DISCARD* - postfix agent will delete the incoming message without sender's or recipient's knowledge;
- *FILTER* - It is possible to select another server which would take action on the incoming message.

Filters that will be mentioned as an example served for defence against phishing attacks on the Academic mail at the University of Montenegro. Academic mail was the target of constant phishing attacks with different email addresses, where the users were requested to send their credentials to log in to the mail server. MailScanner that will be described in the next section did not recognise this type of phishing attacks and a number of users sent their data to the attackers, causing the stolen user accounts to be used for sending spam messages. The sender constantly changed addresses from which the phishing messages were sent, however, the weakness of these attacks was in the unchanged contents of the message.

On the basis of the contents, filters were created which refused further reception of phishing messages from the attackers, disabling the message to reach the end user that would become possible victim of the attacks and allow the attacker to use the provided account for malicious purposes.

Examples of filtering message header and content:
*/^Subject: .*update your web mail account/ REJECT Message is rejected. Header Rule #1*

The example given is intended to automatically reject all messages that have the specified text in their header.
/Your web mail quota has exceeded/ REJECT Phishing Attack. Body Rule #1

Another example is filtering the content of the message. Each message that contains the specified text will be automatically rejected by postfix agent.

This specified filtering can prevent attacks from the examples or various other attacks that aim to exploit vulnerabilities in the server and the system through end users.

The next chapter will describe the MailScanner tool with advanced message filtering mechanisms and defence of both mail server and end users from viruses and spam attacks.

8.2 Setting and configuration of MailScanner tool

MailScanner is an open-source tool for mail server protection from viruses, malware, spam, phishing and other types of attacks. It uses the popular open-source tools such as Clam AV antivirus and Spamassassin for detecting spam messages.

Academic Network mail server is fully based on open-source tools Postfix and Dovecot, and MailScanner filters messages and detects spam and virus contents.

From the official MailScanner tool page you need to download the installation package [5]:
 # wget https://s3.amazonaws.com/mailscanner/release/v4/deb/MailScanner-4.85.2-3.deb.tar.gz

The command tar-zxvf uncompresses the downloaded installation package:
 # tar -zxvf MailScanner-4.85.2-3.deb.tar.gz

Moving into the newly created installation folder and running the install.sh script starts the installation:
 # cd MailScanner-4.85.2-3/
 # ./install.sh

The installation process is simple and requires answers to a few installation steps. The first question should be answered with "N" assuming MailScanner install is done on previously configured mail server with Postfix MTA agent.

- *Do you want to install MTA?*
  When installing Spamassassin tool you need to give an affirmative answer with *"Y"*.
- *Do you want to install or update Spamassassin?*
  As with installing Spamassassin tool you need to give an affirmative answer with "Y" and install Clam AV tool.
- *Do you want to install or update Clam AV during this installation process?*
  Then it is necessary to install additional packages that are essential for the functioning of MailScanner tool.
- *Do you want to install missing perl modules via CPAN?*
  The following question should be negatively answered with *"N"*.
- *Do you want to ignore MailScanner dependencies?*

Assuming that Postfix MTA agent is used, it is necessary to make modifications in /etc/postfix/main.cf configuration file, and to add the line below:
header_checks = regexp:/etc/postfix/header_checks

Then create a file/etc/postfix/header_checks with the following line, if it hasn't already been done in the previous chapter:
/^Received:/ HOLD

It is necessary to modify the MailScanner configuration file /etc/MailScanner/MailScanner.conf and to set up Postfix for MTA agent. It is necessary to modify the following configuration lines:
Run As User = postfix
Run As Group = postfix
Incoming Queue Dir = /var/spool/postfix/hold
Outgoing Queue Dir = /var/spool/postfix/incoming
MTA = postfix

Enable the right of registration to *postfix* users in */var/spool/MailScanner/incoming* and */var/spool/MailScanner/quarantine* folders:
 # chown postfix.postfix /var/spool/MailScanner/incoming
 # chown postfix.postfix /var/spool/MailScanner/quarantine

After the configuration of MailScanner tool for using postfix agent it is necessary to make further changes to */etc/MailScanner/MailScanner.conf* configuration file:
nano /etc/MailScanner/MailScanner.conf

Changes include the entry of data which define organisational name, full name, Web address, and possibly changing of the language of MailScanner reports.

%org-name% = yoursite
%org-long-name% = Your Organisation Name Here
%web-site% = www.your-organisation.com
%report-dir% = /etc/MailScanner/reports/en

According to the recommendations it is necessary to make changes in the configuration file, and to enable the use of *Calm AV* antivirus protection:
Incoming Work Group = clamav
Incoming Work Permissions = 0640
Virus Scanners = clamav

In addition to antivirus protection, it is necessary to enable the spam detection to protect the mail server:
Use SpamAssassin = yes

Activating spam detection requires creating a folder and changing the rights of access to the created folder:
 # mkdir /var/spool/MailScanner/spamassassin
 # chown postfix.postfix /var/spool/MailScanner/spamassassin

Then it is necessary to specify the path to the created folder in MailScanner configuration file:
SpamAssassin User State Dir = /var/spool/MailScanner/spamassassin

The last configuration step requires defining spam lists that will be used for detecting spam messages:
Spam List = spamhaus-ZEN

In the end, it is necessary to modify /etc/default/mailscanner option:
 # nano /etc/default/mailscanner
run_mailscanner=1

The previous change enables MailScanner, after all configuration changes, to be run with the command:
/etc/init.d/mailscanner start

If MailScanner runs without errors this means that the configuration is completed successfully and that MailScanner tool manages the security of the mail server.

9 Security of the web server (Apache)

Apache is the most widely used HTTP service. When you install the Apache package, it is necessary to apply certain configuration changes, as well as additional modules, in order to increase the protection of the service. In this section we are going to mention the most significant changes and additions. Of course, the overall protection of the apache service requires a comprehensive approach to the complete installed system (depending on the selected package), which goes beyond the purpose and scope of this document.

9.1 Protection of Apache service by restricting data display

After installing Apache, it displays information about the installed system in the "error" pages (Apache signature), that should not be displayed in public because it can lead to compromising the system. In order to avoid this, it is necessary to change the following details in the main configuration file:

- Edit *apache2 security* configuration file:

 # nano /etc/apache2/conf.d/security

- Modify the following items:

ServerTokens Prod
ServerSignature Off
TraceEnable Off
Header unset ETag
FileETag None

In this way we also disable "trace HTTP" method, which can retrieve data that can be used to compromise the system.

Also, it is necessary to prohibit the host directory automatic "isting" web. Configuration files are in the “mods-enabled” directory, and here it is necessary to delete "autoindex" configuration files if they have been previously installed.
 # rm /etc/apache2/mods-enabled/autoindex.conf
 # rm /etc/apache2/mods-enabled/autoindex.load

9.2 Protection from DDOS attacks, Modevasive service

DDOS stands for “distributed denial-of-service attack". One way of DDOS attacks is to send a large number of queries to the server, which results in taking up all resources and the server’s inability to respond to valid server requirements, i.e. overloading the server resources and its inability to work in normal mode.

Modevasive is an Apache module that provides an ability to avoid HTTP DOS and DDOS, as well as brute force attacks. Also, modevasive communicates with firewall, and has the ability to communicate with routers. The detection is performed by modevasive's maintaining a database of IP addresses that access the server and block all the addresses that have more than the allowed number of accesses in a second, or have a large number of requests for the apache instance. Also, modevasive has a blacklist in which it temporarily or permanently blocks the IP addresses of possible attackers.

To activate this module you must first install a module from the package repository, as well as create a folder and apply adequate property rights:
apt-get install, libapache2-mod-evasive

Then it is necessary to create the log directory in which modevasive module log files will be placed:
 # mkdir -p /var/log/apache2/evasive

After that, change the access rights to the apache process:
 # chown -R www-data:www-data /var/log/apache2/evasive

The configuration file is located at:
 # nano /etc/apache2/mods-available/mod-evasive.conf
Add or modify the following options in it:
<ifmodule mod_evasive20.c>
DOSHashTableSize 3097
DOSPageCount 2
DOSSiteCount 50
DOSPageInterval 1
DOSSiteInterval 1
DOSBlockingPeriod 10
DOSLogDir /var/log/mod_evasive
DOSEmailNotify email@domain.com
DOSWhitelist 127.0.0.1
</ifmodule>

In module "mod-evasive.load" configuration file it is necessary to add the following lines:
 # nano /etc/apache2/mods-available/mod-evasive.load
DOSHashTableSize 2048
DOSPageCount 20 # maximum number of requests for the same page
DOSSiteCount 300 # total number of requests for any object by the same client IP on the same listener
DOSPageInterval 1.0 # interval for the page count threshold
DOSSiteInterval 1.0 # interval for the site count threshold
DOSBlockingPeriod 10.0# time that a client IP will be blocked for
DOSLogDir "/var/log/apache2/evasive"
DOSEmailNotify admin@domain.com

Before starting the service, verify that the module is loaded:

 # a2enmod mod-evasive
 # service apache2 restart

9.3 Protection from Slowloris attacks

Slowloris is a type of DOS attack which holds open HTTP sessions and sends headers in pre-planned intervals that keep open the opened sessions. To prevent this type of attacks there is also an apache module, which is installed with the command:
 # apt-get -y install libapache2-mod-qos

Verify the configuration in the following location:
 # nano /etc/apache2/mods-available/qos.conf
QoS Settings
<IfModule mod_qos.c>
 # handles connections from up to 100000 different IPs
QS_ClientEntries 100000
 # will allow only 50 connections per IP
QS_SrvMaxConnPerIP 50
 # maximum number of active TCP connections is limited to 256
MaxClients 256
 # disables keep-alive when 70% of the TCP connections are occupied:
QS_SrvMaxConnClose 180
 # minimum request/response speed (deny slow clients blocking the server,
 # ie. slowloris keeping connections open without requesting anything):
QS_SrvMinDataRate 150 1200
 # and limit request header and body (carefull, that limits uploads and
</IfModule>

9.4 Spamhaus modul

Spamhaus module uses DNSBL to prevent "cc injection" attacks, "spam relay" attacks through web forms, and generally to protect the server from the known malicious IP addresses. For its installation it is necessary to run the following commands and to set up the configuration file:
 # apt-get -y install libapache2-mod-spamhaus
 # touch /etc/spamhaus.wl

Add the following lines in 'apache2.conf':
 # nano /etc/apache2/apache2.conf
<IfModule mod_spamhaus.c>
MS_METHODS POST,PUT,OPTIONS,CONNECT
MS_WhiteList /etc/spamhaus.wl
MS_CacheSize 256
</IfModule>

9.5 Protection of PHP module

PHP is an essential part of a web page so it is necessary to protect the PHP service on your system. Some of the possible attacks:

- XSS-scripts that work among multiple hosted Web sites can pose a security risk and enable the attacker to retrieve the user information. Its protection means the appropriate configuration of the apache service and creating more secure PHP scripts.
- SQL injection – sql command execution via PHP page. This attack is handled by limiting and validating user inputs.
- Uploading files by which attackers get access to the system and opportunities to access parts of the file system.
- Remote execution of files allows attackers to set backdoor scripts.
- CSRF "cross-site request forgery" – this attack allows the end user to perform unwanted activities on the web application for which they are currently authenticated.

In order to avoid successfully completed attacks it is necessary:

- to disable the option of advertising information about PHP installed on your system;
- to enable log for errors and disable the option of public display of errors.

It is necessary to edit php.ini file:
nano /etc/php5/apache2/php.ini

Then, modify the following lines:
disable_functions = exec,system,shell_exec,passthru
register_globals = Off
expose_php = Off
display_errors = Off
track_errors = Off
html_errors = Off
magic_quotes_gpc = Off

In "security.ini" file it is necessary to add or modify the following lines:
#Control of the maximum size of the upload file
file_uploads=On
upload_max_filesize=1M

# Control of the size of the request
post_max_size=1K

# Disabling the remote execution and picking up information from a remote location
allow_url_fopen=Off
allow_url_include=Off

#Protection of *mysql* – ignoring any additional arguments
sql.safe_mode=On
magic_quotes_gpc=Off

#DOS control
max_execution_time = 30
max_input_time = 30
memory_limit = 40M

Disabling dangerous PHP functions

disable_functions=exec,passthru,shell_exec,system,proc_open,popen,curl_exec,curl_multi_exec,parse_ini_file,show_source

After modifying the "security.ini" file you need to restart the apache service:
 # service apache2 restart

9.6 Modsecurity service

ModSecurity is an apache module that provides detection of attacks and protection of web applications. The module itself does not provide protection, the policies that are implemented are important. Basic rules of CRS (Core Rule Set) are available from OWASP (Open Web Application Security Project) website. CRS rules are also contained in Debian repository.

ModSecurity provides protection against several types of attacks:

- *SQL injection*
- *XSS cross-site scripting*
- *path traversal*
- *file inclusion*
- *Denial of service (DOS)*
- *Brute force*

The installation is simple:
 # apt-get install libapache2-modsecurity

Then it is necessary to copy OWASP rules to the configuration directory.
 # cp -rp /usr/share/modsecurity-crs/ /etc/modsecurity*

With the following command you link the rules from "base_rules" and "optional_rules" folders to active rules:
for f in `ls base_rules/` ; do ln -s /etc/modsecurity/base_rules/$f activated_rules/$f ; done
for f in `ls optional_rules/` ; do ln -s /etc/modsecurity/optional_rules/$f activated_rules/$f ; done

Then it is necessary to modify the configuration file. The configuration file is located at "*/etc/modsecurity/modsecurity.conf*". The following lines need to be changed:
 # nano /etc/modsecurity/modsecurity.conf

```
<IfModule security2_module>
 Include /etc/modsecurity/*.conf
 Include /etc/modsecurity/activated_rules/*.conf

 # enable suspicious URL blocking
 SecRuleEngine On
 SecResponseBodyAccess Off
 SecRequestBodyLimit 16384000
 SecRequestBodyNoFilesLimit 16384000
 SecRequestBodyInMemoryLimit 16384000

 # uncomment on production
 #SecAuditEngine Off
```

</IfModule>

Initially, modsecurity enabled uploading files of up to 120kb. The above values raise the limit to 16 MB.

"headers" is an apache module that is required for modsecurity rules, and is launched as follows:
a2enmod headers

Then it is necessary to activate the modsecurity module and to restart the apache service:
a2enmod modsecurity
service apache2 restart

10 Conclusion

The implementation of the described security tools has improved the security of the Ubuntu server of the University of Montenegro Academic network (AMUCG). Most servers in AMUCG are based on Ubuntu Linux server distribution. Ubuntu server distribution was selected because of its stability, long-term support (LTS), ease of administration, and the availability of numerous free tools.

Timely detection and prevention of attacks are key to the security of each system. In order to raise the security of the server to a satisfactory level it is necessary to follow some basic security recommendations that are described in this document. The recommendations include a mandatory configuration of the firewall on the server, installation and configuration of Fail2ban tools, with the aim of preventing "brute-force" attacks, as well as the described "intrusion detection" systems.

The variety of attacks also requires the implementation of various security mechanisms. For a number of attacks, which generally do not have a clearly defined target, implementation of basic security recommendations represents a satisfactory level of protection. The Implementation and selection of security mechanisms largely depends on the purpose of the server. The document describes methods for protecting the Mail server and its defence against spam, phishing, and other attacks. It also describes and gives recommendations for Web server protection against various attacks, by using and configuring security tools modevasive, spamhouse, modsecurity, as well as other methods.

References

[1] Ubuntu, "*UFW - Uncomplicated Firewall*",

https://help.ubuntu.com/community/UFW

[2] Ubuntu, "*Fail2ban*",

https://help.ubuntu.com/community/Fail2ban

[3] Justin Ellingwood, "*How To Use psad to Detect Network Intrusion Attempts on an Ubuntu VPS*",

https://www.digitalocean.com/community/tutorials/how-to-use-psad-to-detect-network-intrusion-attempts-on-an-ubuntu-vps

[4] Justin Ellingwood, "How To Use Tripwire to Detect Server Intrusions on an Ubuntu VPS",

https://www.digitalocean.com/community/tutorials/how-to-use-tripwire-to-detect-server-intrusions-on-an-ubuntu-vps

[5] MailScanner,

https://www.mailscanner.info/

[6] Postfix, "Header and Body checks",

https://posluns.com/guides/

[7] How to secure an Ubuntu 12.04 LTS server - Part 1 The Basics,

https://www.thefanclub.co.za/how-to/how-secure-ubuntu-1204-lts-server-part-1-basics

[8] Martian packet, Wikipedia,

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

[9] AppArmor, Ubuntu,

https://help.ubuntu.com/12.04/serverguide/apparmor.html

List of abbreviations

AMUCG	Akademska mreža Univerziteta Crne Gore
CSRF	Cross Site Request Forgery
DDOS	Distributed Denial of Service Attack
DNS	Domain Name System
DNSBL	DNS Based Blackhole List
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
LTS	Long Term Support
MAC	Mandatory Access Control
MITM	Man In The Middle
MTA	Mail Transfer Agent
OWASP	Open Web Application Security Project
SQL	Structured Query Language
SSH	Secure Shell
TMPFS	Temporary File Storage Facility
UFW	Uncomplicated Firewall
XSS	Cross Site Scripting