ACME-29
Assignment 4 Report

Edoardo Gabrielli (1693726), Davide Quaranta (1715742), Alessio Tullio (1809077)

Initial brainstorming

The ACME network now requires to send logs generated in the servers, to the Log Server in the `Internal Servers network`. The firewall rules to allow it are already configured in the first assignment, so we only need to configure logging services.

We opted to use **rsyslog**, since it is already installed on all the hosts, and the configuration only requires specifying what to forward and where.

Implementation details

Generally it is needed to enable **log reception** on the Log Server, and set the hosts to **forward** their logs to the Log Server.

Server setup

We opted for receiving logs on the **standard UDP port 514**.
It can be done by uncommenting the following lines in `/etc/rsyslog.conf`:

```
module(load="imudp")
input(type="imudp" port="514")
```

Then we created a file `/etc/rsyslog.d/remote.conf`, with contents:

```
$template RemoteLogs,"/var/log/%HOSTNAME%/%PROGRAMNAME%.log"
*.* ?RemoteLogs
& ~
```

This will organize receiving logs in a **directory for each sending hostname**.
For example, if the logs come from the `webserver` hostname, the structure would be `/var/log/webserver/logname.log`.

Clients setup

The configuration is similar for each host, adapted to which log to forward.
In our model, we configured clients to

- Forward all the logs defined in `/etc/rsyslog.conf` (e.g. syslog, mail).
- Forward extra logs for other services (e.g. apache2).

Generally, in each client we added a file `/etc/rsyslog.d/00-common.conf` with content:

```
module(load="imfile")
```

This allows us to define personalized file inputs, by creating a file `/etc/rsyslog.d/<service>.conf`, with content:

```
input(type="imfile"
      File="/var/log/<service>.log"
      PersistStateInterval="10"
      Tag="<service>"
      Severity="notice"
      Facility="local0")
local0.notice    @100.100.1.3:514
```

Where **`<service>`** is the name of a service that we want to log.

For example, this is the configuration to forward `apache2` logs from the `Web Server`: (`/etc/rsyslog.d/apache2.conf`):

```
input(type="imfile"
      File="/var/log/apache2/error.log"
      PersistStateInterval="10"
      Tag="apache2"
      Severity="error"
      Facility="local0")
local0.error    @100.100.1.3:514

input(type="imfile"
      File="/var/log/apache2/access.log"
      PersistStateInterval="10"
      Tag="apache2"
      Severity="notice"
      Facility="local0")
local0.notice    @100.100.1.3:514
```

This will create the following structure on the `Log Server`: `/var/log/webserver/<tag>.log`, where `<tag>` in this case is `apache2`.

Please refer to the **attached `.conf` files** for the forwarding of other services.

Tests

We can verify that the logs flow to the Log Server, by watching the directory structure inside `/var/log`, and making sure that the files are the expected ones:

```
root@logserver:/var/log# tree dc logserver webserver proxyserver
dc
|-- CRON.log
[... cut ...]
`-- zentyal.log
logserver
|-- CRON.log
[... cut ...]
|-- fail2ban-client.log
|-- fail2ban-server.log
`-- userdel.log
webserver
|-- CRON.log
[... cut ...]
|-- apache2.log
|-- fail2ban.log
`-- systemd.log
proxyserver
|-- CRON.log
|-- clamd.log
[... cut ...]
`-- zentyal.log

0 directories, 57 files
```

Final remarks

After satisfying all the ACME requests, we offered to the corporation the installation of **fail2ban**, a log files scanner (e.g. `/var/log/apache/error_log`), able to block IPs that try to brute force the **webserver** and the **logserver** via ssh.
The following guide applies to both the webserver and logserver machine.

Installing and configuring fail2ban

We installed fail2ban from APT, and made sure that it runs on system startup, with:

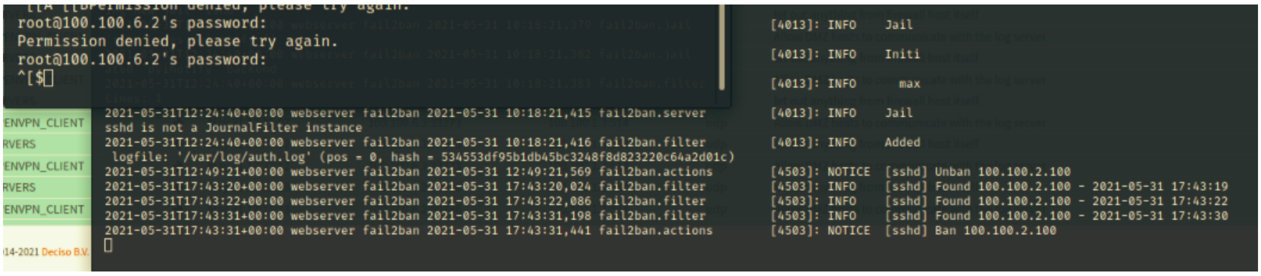
```
$ sudo systemctl enable fail2ban.service
```

Creating SSH jails with fail2ban

We created a file, called `jail.local`, containing the configuration to block brute force attempts via ssh; `/etc/fail2ban/jail.local`:

```
[sshd]
enabled = true
port = ssh
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
findtime = 300
bantime = 3600
ignoreip = 127.0.0.1
```

Now, all the hosts that try to access the Web Server and Log Server via ssh, will be **banned** at the fourth wrong credentials attempt.



```
 [[A [[BPermission denied, please try again.
root@100.100.6.2's password:                                                          [4013]: INFO    Jail
Permission denied, please try again.
root@100.100.6.2's password:                                                          [4013]: INFO    Initi
^[$
                                                                                      [4013]: INFO      max
2021-05-31T12:24:40+00:00 webserver fail2ban 2021-05-31 10:18:21,415 fail2ban.server   [4013]: INFO    Jail
sshd is not a JournalFilter instance
2021-05-31T12:24:40+00:00 webserver fail2ban 2021-05-31 10:18:21,416 fail2ban.filter   [4013]: INFO    Added
 logfile: '/var/log/auth.log' (pos = 0, hash = 534553df95b1db45bc3248f8d823220c64a2d01c)
2021-05-31T12:49:21+00:00 webserver fail2ban 2021-05-31 12:49:21,569 fail2ban.actions  [4503]: NOTICE  [sshd] Unban 100.100.2.100
2021-05-31T17:43:20+00:00 webserver fail2ban 2021-05-31 17:43:20,024 fail2ban.filter   [4503]: INFO    [sshd] Found 100.100.2.100 - 2021-05-31 17:43:19
2021-05-31T17:43:22+00:00 webserver fail2ban 2021-05-31 17:43:22,086 fail2ban.filter   [4503]: INFO    [sshd] Found 100.100.2.100 - 2021-05-31 17:43:22
2021-05-31T17:43:31+00:00 webserver fail2ban 2021-05-31 17:43:31,198 fail2ban.filter   [4503]: INFO    [sshd] Found 100.100.2.100 - 2021-05-31 17:43:30
2021-05-31T17:43:31+00:00 webserver fail2ban 2021-05-31 17:43:31,441 fail2ban.actions  [4503]: NOTICE  [sshd] Ban 100.100.2.100
```