Step-by-step guide to Linux security

Clément Levallois

2017-04-03

Table of Contents

Ordering the server . . . 1
Disabling password authentication, enabling SSH . . . 1
  How to generate a SSH key? . . . 1
  How to disable password auth and enable SSH? . . . 1
  Changing the SSH port . . . 1
Setting up a firewall . . . 2
  ip tables . . . 2
  better: uncomplicated firewall . . . 2
Creating users and disabling SSH connections for root . . . 2
the end . . . 3

last modified: 2017-04-04

Ordering the server

- Server ordered on Hetzner.de
- I use Debian, version 8.7
- Vi is used as a text editor in the following
- we are logged as root first

Disabling password authentication, enabling SSH

Password authentication is less secure than SSH public key. A password transits through the Internet for the auhtentication, it can be hacked at this step.

A SSH private key is not transmitted on the wire. So, it can't be hacked this way.

A detailed explanation is available here.

How to generate a SSH key?

- On Windows, use Puttygen.
- On Mac, use the Terminal
- On Linux, use the ssh-keygen command

How to disable password auth and enable SSH?

→ the server I rented on Hetzner asked it from the console right when renting it. → otherwise, this can be set up manually this way.

(see also the "all commands in order" tutorial for precise instructions)

Changing the SSH port

By default, loggging to the server via SSH is done on the port 22. Knowing that, attackers scan the port 22. Changing the port to a different one makes the attacker's job more difficult. To do that:

```
vi /etc/ssh/sshd_config
```

Text to change in the file: change port SSH 22 by a new port (**let's say 1234**), write the new port down somewhere

```
service sshd restart
```

Setting up a firewall

A firewall gives you control on what can enter and leave your server.

ip tables

The rules for setting up ip tables are logical but quite complicated. Using an ip tables generator could help.

But there is an even easier alternative.

better: uncomplicated firewall

Following @mgilbir's advice, I'll use ufw: a linux package for "uncomplicated firewall". To install it:

```
apt-get install ufw
```

The firewall is now installed, but is is not active yet.

We add a rule to block all incoming traffic, except for SSH connections through the port we defined:

```
ufw default deny incoming
ufw allow 1234/tcp
```

```
//ST: !
```

Now, we can activate the firewall

```
ufw enable
```

Creating users and disabling SSH connections for root

We should now disable root login via SSH. Why? Because attackers would know that a "root" user is available to log in, and it just remains to attack its password.

With the root user disabled at the SSH login step, the attackers must guess **both** the username and its password to access the connection, and that's much harder.

Of course, an attacker who aims at you or your server specifically (a "targeted" attack) would expect a series of usernames (in my case "seinecle", the name I use on all social media), so don’t use it either.

the end

Author of this tutorial: Clement Levallois

All resources on linux security: https://seinecle.github.io/linux-security-tutorials/