Firewall

Caution

This guide is only for people who have a good level in dedicated server administration with linux environment. the handling of a firewall can be very dangerous. Indeed, you can block your server what will force you to start it again in HARD mode. If you make a mistake on the final script and you set it in auto startup, you won't have any access on your own server any more! Be very careful and if you don't feel comfortable at all with this guide, don't set up your firewall!

What's the firewall?

It's a software which blocks some ports on your own server and can open other ones. Imagine your house for example: You have a front door and a back door. You never use the one at the back so you better block it up.why? Because it's potential risk for a thief to come in. With a firewall it's the same, we close every port, we don't need.

Which ports do you use in general? Warning!

First of all, be very careful that you are going to do. Indeed, you risk to close a wrong port. Imagine if you close the SSH port therefore you will have either to restart the server via telnet, or via webmin or reboot.

By default opened ports on OVH server are :

21 − ftp (the FTP server according to your use). 22 − ssh (the crypted shell access according to your use). 23 − telnet (the non−crypted shell access in case of troubleshooting service). 25 − smtp (the SMTP server according to your use). 53 − dns (the DNS server according to your use). 80 − http (the web server according to your use). 110 − pop3 (the mail access according to your use). 143 − imap (the mail access if you don't use Pop3). 443 − http (the crypted web access according to your use). 1000 − webmin (Server configuration panel according to your use).
These ports are open by default, but you might have running software which open other ones.It's up to you to know which one you have to keep or not. Ones you have made your choice, you're ready to start.

Iptables

Iptables is a powerfull firewall, installed on all OVH servers. The process is as follows: We will open some ports and close the rest. In this example, only 22 port (SSH) and 80 (HTTP) will remain open. This is only an example, in the future, adapt it according to your need.

Connect you with SSH in root. Firstly, you must check the iptables version: $ /sbin/iptables −V iptables v1.2.4 The version is too old. We'll install the 1.2.9 version: $ cd /root $ wget http://www.netfilter.org/files/iptables−1.2.9.tar.bz2 $ tar xvfj iptables−1.2.9.tar.bz2 $ cd iptables−1.2.9 $ make KERNEL_DIR=/usr/src/linux $ make install KERNEL_DIR=/usr/src/linux $ cd /sbin $ mv iptables iptables.old $ mv iptables−restore iptables−restore.old $ mv iptables−save iptables−save.old $ ln −s /usr/local/sbin/iptables iptables $ ln −s /usr/local/sbin/iptables−restore iptables−restore $ ln −s /usr/local/sbin/iptables−save iptables−save $ /sbin/iptables −V iptables v1.2.9
It's done, iptables is update, we can continue.

We list the folowing rules:

$ /sbin/iptables −L Chain INPUT (policy ACCEPT) target prot opt source destination Chain FORWARD (policy ACCEPT) target prot opt source destination Chain OUTPUT (policy ACCEPT) target prot opt source destination
We can see 3 chains: Input, Forward and Output. Firstly, we'll work on Input chain (for the input traffic). We authorize 22 and 88 ports:

$ /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 22 −j ACCEPT $ /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 80 −j ACCEPT
−A INPUT : We set our rule on input. −i eth0 : Here, we are only concerned by the ethernet interface. −p tcp : The concerned protocol is the TCP protocol (We only work on this one for the moment). −−dport 22 : The rule will be applied on the SSH port (number 22). −j ACCEPT : We accept this traffic.

We display all rules:

$ /sbin/iptables −L Chain INPUT (policy ACCEPT) target prot opt source destination ACCEPT tcp ? anywhere anywhere tcp dpt:ssh ACCEPT tcp ? anywhere anywhere tcp dpt:www Chain FORWARD (policy ACCEPT) target prot opt source destination Chain OUTPUT (policy ACCEPT) target prot opt source destination
The input section is filled slowly, it's a good sign ;).

We can see that the default policy is to accept everything => Chain INPUT(policy ACCEPT). We want to block all the traffic, which we didn't authorize previously. Therefore We'll add a rule which will block the others ports. But we encounter a problem:

When a connection will be established from our server to the kernel.org server to download the new kernel for example, it will establish a connection to the site and will wait for its response. The request will reach the kernel.org correctly and but How will it come back to the server, as we blocked everything?

Fortunately, iptables is powerful and can sort packets according to their states. We will then add a rule:

$ /sbin/iptables −A INPUT −i eth0 −m state −−state ESTABLISHED,RELATED −j ACCEPT
Now, we can block the rest (Warning, it's where the firewall will be fully in action, check that you have correctly configured your rules otherwise you will block your server!):

$ /sbin/iptables −A INPUT −i eth0 −j DROP
For this rule, we have 2 choices. The first solution, we drop packets, i.e if a packet arrives and it isn't accepted, we delete it. The client will wait for a response until a timeout. The second solution is to reject packets (REJECT instead of DROP). If a unsolicited packet arrives, we send back to the client an error and he won't wait as he has a negative response.

Reject packets is cleaner but to throw it is more secure. Indeed, imagine someone who sends you packets in loop to a wrong port, your server won't process them, whereas with the reject rule, it will take time to answer.

It's up to you ;)

To reset your firewall, type:

$ /sbin/iptables −F INPUT
This command will delete all the rules of Input part. If you want to add a rule between the first and the second, type this:

$ /sbin/iptables −I INPUT 2 ... the following of your rule
To delete the rule number 3, type this:

$ /sbin/iptables −D INPUT 3
To block totally an Ip address:

$ /sbin/iptables −I INPUT 1 −s −j DROP
Now, the firewall is in action. Try to scan your server, you will be able to see only 22 and 80 ports open. If the scan is very slow, it's because of the DROP rule.

Exclude and Authorize an Ip address

For your dedicated server:

If you want to block the ICMP protocol (Ping requests), you have to let at least ping.ovh.net, proxy.p19.ovh.net, proxy.rbx.ovh.net and proxy.ovh.net to ping your server. It enables to the OVH team to check the status of your server.

In addition, you have to let the Ip address as the following example:

The Ip address of your server is aaa.bbb.ccc.ddd You have to pass: aaa.bbb.ccc.250 Example: 213.186.57.153 must pass 213.186.57.250 for the SLA server and 213.186.57.251 for the MRTG server in order to benefit from the RTM.

If you are an owner of HG server, pass Ip adress aaa.bbb.ccc.249 (temporary rule).

If you block all ping requests as well as Ovh's requests, we won't be able to check the state of your server and if a problem occurs, we won't be informed. To authorize ping from our servers, type the following rules:

/sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.p19.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.rbx.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.rbx2.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source ping.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source IP.250 −j ACCEPT # IP = aaa.bbb.ccc according to the previous rule /sbin/iptables −A INPUT −i eth0 −p icmp −−source IP.249 −j ACCEPT # temporary, only for HG server /sbin/iptables −A INPUT −i eth0 −p icmp −−source IP.251 −j ACCEPT # IP for the monitoring system
Concerning SSH, if you want to restrict the access from your Ip only, we advise you to keep cache.ovh.net. Indeed, in case of problem on your server, we'll be able to intervene and fix it. If you close the 22 port for Ovh technicians, we won't be able to help you if your server is blocked.

To authorize SSH from our server, type the following rule:

/sbin/iptables −A INPUT −i eth0 −p tcp −−dport 22 −−source cache.ovh.net −j ACCEPT
f you have a RAID filer, don't forget to authorize the NFS connections. We can authorize everything that comes from intern network 192.168.0.0/16:

/sbin/iptables −A INPUT −i eth0 −p tcp −−source 192.168.0.0/16 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p udp −−source 192.168.0.0/16 −j ACCEPT
If you have a cluster configuration, you must authorize the 79 port in order OCO to communicate with the distributor of load.

/sbin/iptables −A INPUT −i eth0 −p tcp −−dport 79 −j ACCEPT
For your RPS server:

The interface monitored by our services is the eth0 with the firewall rules applied to it.

If you block all ping requests, even those with OVH, we will not be able to monitor the correct working of your server and if it falls, we will not warned. To allow ping from our servers, enter the following rules:

/sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.p19.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.rbx.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.rbx2.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source ping.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 22 −−source cache.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source IP.251 −j ACCEPT # IP for the RTM monitoring system RTM /sbin/iptables −A INPUT −i eth0 −p icmp −−source IP.251 −j ACCEPT # IP for the SLA monitoring system
It will also allow your filer, to find it use the command:

r12xxx ~ # netstat −tanpu | grep iscsi tcp 0 0 91.121.xx.xx:38632 91.121.191.16:3260 ESTABLISHED 3097/iscsid
The IP of your filer: 91.121.191.16 The rule must be added:

/sbin/iptables −A INPUT −i eth0 −p tcp −−source 91.121.191.16 −j ACCEPT

Example of complete configuration

Here's an example of complete script to protect your server via iptables. It is permissive, because all services present on your server, are reachable but it can be used for your own configuration:

/sbin/iptables −A INPUT −i eth0 −m state −−state ESTABLISHED,RELATED −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 25 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 53 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p udp −−dport 53 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 80 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 110 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 443 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 10000 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 21 −−source xx.xx.xx.xx −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 22 −−source cache.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 22 −−source xx.xx.xx.xx −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.p19.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source proxy.rbx.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p icmp −−source ping.ovh.net −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−source 192.168.0.0/16 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p udp −−source 192.168.0.0/16 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −p tcp −−dport 79 −j ACCEPT /sbin/iptables −A INPUT −i eth0 −j REJECT
In these rules, you must replace xx.xx.xx.xx by the Ip address of server, which you are able to connect to your server in FTP and SSH.

Automate the firewall

Once your server is perfectly configured, you have to create a script which will execute at the beginning of each boot of your server. Here's an example to put in a file named "firewall" for example in the directory /etc/init.d/:

#!/bin/sh 1. chkconfig: 3 21 91 2. description: Firewall IPT=/sbin/iptables case "$1" in start) $IPT −A INPUT

−i eth0 −m state −−state ESTABLISHED,RELATED −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 25 −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 53 −j ACCEPT $IPT −A INPUT −i eth0 −p udp −−dport 53 −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 80 −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 110 −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 443 −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 10000 −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 21 −−source xx.xx.xx.xx −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 22 −−source cache.ovh.net −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 22 −−source xx.xx.xx.xx −j ACCEPT $IPT −A INPUT −i eth0 −p icmp −−source proxy.ovh.net −j ACCEPT $IPT −A INPUT −i eth0 −p icmp −−source proxy.p19.ovh.net −j ACCEPT $IPT −A INPUT −i eth0 −p icmp −−source proxy.rbx.ovh.net −j ACCEPT $IPT −A INPUT −i eth0 −p icmp −−source ping.ovh.net −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−source 192.168.0.0/16 −j ACCEPT $IPT −A INPUT −i eth0 −p udp −−source 192.168.0.0/16 −j ACCEPT $IPT −A INPUT −i eth0 −p tcp −−dport 79 −j ACCEPT $IPT −A INPUT −i eth0 −j REJECT exit 0 ;; stop) $IPT −F INPUT exit 0 ;; *) echo "Usage: /etc/init.d/firewall {start|stop}" exit 1 ;; esac
Give it the 700 rights and type "/etc/init.d/firewall start" to start it and "/etc/init.d/firewall stop" to stop it. To launch it automatically at the startup:

$ /sbin/chkconfig −−level 3 firewall on $ /sbin/chkconfig −−level 06 firewall off
Before launching the script at each startup, check that is correct otherwise your server will be absolutely blocked!