

20 September 2006

By: Mihai Marinof, Linux Editor



Iptables Basic Guide

Learning the basics about iptables

People always complain about how hard iptables are to understand and configure. I'm not saying it's an easy process but once you get the hang of it, it should seem a little less difficult. Iptables is a user space tool which is used to create rules for packet filtering and NAT modules. Basically, iptables create the firewall for your Internet connection in Linux. If you are new to Linux but will use it on a regular basis, then you must learn how to use iptables as your whole system security is based on it.

Kernel support check Before you configure iptables, you first have to check if your system has been configured properly. First of all, you must check if the kernel was compiled with iptables support. This can be done with the 'grep' command on the kernel config file. The command: `# cat /boot/config-your.kernel.version.here | grep -i "CONFIG_IP_NF"` should print some lines ending with '=y' or '=m'. Also, "CONFIG_IP_NF_IPTABLES" line must end with '=m' (that means iptables was compiled as a module).

Iptables check/installing You must check if you have iptables installed by executing the command: `# rpm -qa | grep iptables`. This should print iptables-*your.installed.version* and eventually iptables-devel-*your.installed.version* (this is optional). If you don't have it on your system, you can easily install it either by downloading the latest rpm package from iptables homepage and executing the command: `# rpm -Uvh iptables-downloaded.version.rpm`. Or by using yum: `# yum install iptables`

Where are the main iptables files stored? `/etc/init.d/iptables` is the INIT script which is used to start, stop the service and/or to save the rulesets. `/etc/sysconfig/iptables` this is the file that holds the saved rulesets. `/sbin/iptables` and this is the iptables binary.

Before you actually start configuring the rules, let's take a look at the current configuration: `# iptables -L`

By default, there are only three chains which hold exclusive rules: INPUT, OUTPUT and FORWARD. The INPUT chain contains rules for traffic incoming to your server, OUTPUT contains the rules for traffic outgoing from your server to the Internet and FORWARD contains the rules for traffic that will be forwarded to other computers behind yours (if your server is a firewall for a LAN). Iptables are mostly configured to deal with traffic incoming from the Internet to the Linux server, so the chain INPUT is used. When traffic passes through your Linux kernel, a TARGET is determined based on whether the packet matches a rule in the rulesets or not. The mainly used targets are: **ACCEPT**: traffic is allowed to pass through to its destination; **REJECT**: traffic is blocked from reaching its destination and a packet is sent back to the sending host with a quick explanation; **DROP**: traffic is blocked with no explanation sent back whatsoever.

Configuring iptables rulesets Before the actual configuration, I insist on giving you a few tips I've learned the hard way so you don't have to. You must keep in mind that the order in which you add rules is everything. For example, if you make the mistake of adding the first rule to deny everything, then, no matter what you set to allow after, it will still be denied. The second thing you must keep in mind is that the rulesets are in memory and are not saved on disk automatically, so if you reboot your computer without first running the INIT script (or iptables-save) to save them, everything you've worked for will be gone. Another important thing that applies if you work on your server from a remote PC, through ssh: it's important not to block yourself out so make this your first rule: `# iptables -A INPUT -s 213.10.10.13 -d 192.168.1.1 -p TCP -dport 22 -j ACCEPT`

Explanation: **-A**: appends the rule to the INPUT chain; **-s**: source IP, in this case, the IP you are ssh'ing from; **-d**: destination IP, in this case, the server IP; **-p**: communication protocol; **--dport**: destination port, in our case it's the SSHd port; **-j**: stands for 'Jump'. If everything matches, the packet is accepted.

Next, let's set some basic rules for general traffic. One of iptables' features is the ability to determine the state a packet is in. This is the

packets' state on a new connection:**NEW**: 1st server sends 2nd a SYN packet that it wants to create a new connection.**RELATED**: 2nd server receives the SYN packet and sends to 1st server a SYN-ACK packet which tells that everything is alright.**ESTABLISHED**: 1st server receives the SYN-ACK packet and sends 2nd server an ACK packet which is the final acknowledgment, the connection finishes establishing and the traffic start between the two servers. In order for your server to be able to establish TCP connections with other servers, iptables must be configured with the following rules:

```
# iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT# iptables -A FORWARD -i eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT# iptables -A OUTPUT -m state --state RELATED,ESTABLISHED
```

Custom rulesTo block IPs:**# iptables -A INPUT -s 213.10.10.13 -j DROP**This rule blocks any incoming traffic from 192.168.1.13.**# iptables -A INPUT -d 192.168.1.15 -j REJECT**This rule blocks any incoming traffic for LAN computer with IP 192.168.1.15 (behind your server). To allow IPs:**# iptables -A INPUT -s 213.10.10.13 -d 192.168.1.4 -p tcp --dport 21**This rule accepts incoming traffic from source IP to destination IP which is a FTP server. After you've configured all the rules for every port on your local or network service you want to allow (or block) traffic to or from, it's time to block the rest:**# iptables -A INPUT -j REJECT# iptables -A FORWARD -j REJECT**These block rules MUST be added last. To delete a rule, simply write it again but replace the '-A' before the chain with '-D' (Delete).Saving rulesetsTo save your iptables configuration, simply execute the command:**# /etc/init.d/iptables save**To stop iptables and flush all rules: (careful, use the save INIT script before stopping iptables and flushing rules)**# /etc/init.d/iptables stop**To start the iptables again and loading the rulesets from /etc/sysconfig/iptables:**# /etc/init.d/iptables start**

The EndThis is a very, very basic guide. Iptables can be used way beyond these basics, in extremely numerous combinations; therefore, it's not humanly possible to write a complete guide about iptables. However, for a basic security and for one to make an idea about how iptables work, I hope this guide will help someone, someday...