4. Firewalls

[Plano Técnico]

Windows

- Zone Alarm http://www.zonealarm.com
- Kaspersky http://www.kaspersky.com/
- Comodo http://www.personalfirewall.comodo.com/
- Sunbelt Personal Firewall http://www.sunbelt-software.com/Home-Home-Office/Sunbelt-Personal-Firewall/
- Kerio http://www.kerio.com/
- Nativo del SO
- Windows Server ISA Server (ISA: Internet Security and Acceleration)

Unix

- Firestarter http://www.fs-security.com/
- Fwbuilder http://www.fwbuilder.org/
- nmap http://insecure.org/nmap/

Hack 69. Protect Your Network with a Firewall

Protect your network with a firewall managed from your Ubuntu desktop.

Linux has an excellent kernel-based network packet-management system called iptables that can be configured either directly from the command line or through a variety of GUI administration interfaces. One of the most powerful firewall-management interfaces is called Firewall Builder, a system designed to separate policy from implementation and allow you to concentrate on what you want your firewall to do, rather than how you want it to do it.

The Firewall Builder interface presents hosts, routers, firewalls, networks, and protocols as objects, and allows you to drag and drop those objects to define your firewall policy. Firewall Builder then compiles your policy into the actual rules needed to enforce it, with multiple policy compilers available to suit different types of firewall. You can define your policy using Firewall Builder running on an Ubuntu desktop and then have it compiled for a firewall running iptables on Linux, ipfilter on BSD, or any of about half a dozen other firewall technologies. The policy can be defined exactly the same way, regardless of the technology deployed on the target firewall. And because Firewall Builder can support multiple firewalls simultaneously, you can use it as a central management console to configure a variety of firewalls and individual hosts throughout your network, all from a single, unified interface.

You can run Firewall Builder directly on your firewall if you choose, but as a general policy, it's a good idea to have your firewall running the absolute minimum system possible, so a better approach is to have a dedicated machine as your firewall and run Firewall Builder on a desktop or laptop management machine. Then whenever you want to update your firewall policy, you can run Firewall Builder on your management machine to generate new rules and push them out to the firewall.

Initial Firewall Setup

Start by setting up your firewall machine with a minimal Ubuntu installation: run the installer in server mode [Hack #93] so that it installs only basic packages, and preferably install at least one extra Ethernet card so that you can keep untrusted Internet traffic away from your internal network. A standard approach is to run three network

interfaces on a firewall: one for your internal network (downstream), one to connect to the Internet (upstream), and one to a separate local network called the De-Militarized Zone (DMZ), where you can put servers that you want to expose to the Internet. Configure the network interfaces to suit the networks they are connecting to and make sure that your firewall can connect to each one of them individually by using ping to check whether you can see hosts on each network.

Your firewall machine is now sitting at the crossroads between the Internet, your internal network, and any servers that you want to run, but it doesn't yet know how to pass data from one to another so everything will be effectively isolated. To enable your firewall to pass packets from one network interface to another and perform packet filtering and network/port address translation, you will need to install iptables, and to allow the firewall to be managed remotely, you will need to install an SSH server:

```
$ sudo apt-get install iptables ssh
```

Initial Management-Machine Setup

Install Firewall Builder on your management machine along with RCS and the Firewall Builder documentation package:

```
$ sudo apt-get install fwbuilder rcs fwbuilder-doc
```

Now you're ready to perform the rest of the steps.

Create a firewall project

First, launch Firewall Builder:

```
$ fwbuilder
```

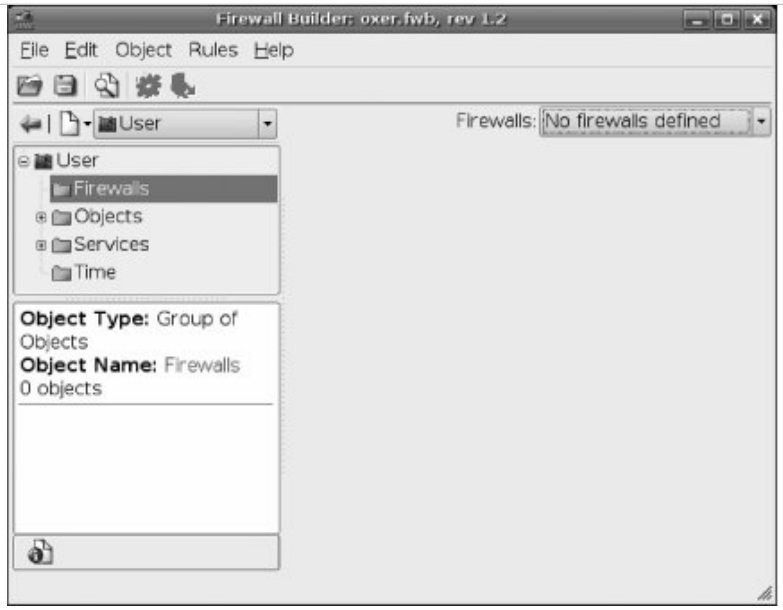
Select Create New Project File and specify a location to save it. Ideally, you should create a special directory to hold the project file because Firewall Builder will also generate other files in the same directory, and having them all in one place makes it easier to back up your firewall configuration.

You are then given the option of activating revision control for the project as well as setting it to be opened automatically when Firewall Builder starts up. Turn on both options.

The revision-control option tells Firewall Builder to store the configuration file in RCS, allowing you to see the entire history of the file, including all changes that have ever been made to it. This feature can be extremely handy if you manage to break your firewall and need to roll back to a known-good working configuration.

Firewall Builder will initially start with an empty configuration containing only a number of predefined services in two libraries (see Figure 7-1). The libraries are called User and Standard, and you can switch between them using the drop-down menu near the upper left. The Standard library is a read-only library that ships with Firewall Builder and contains predefined services for almost every TCP and UDP service in common use, along with predefined network ranges and time ranges. The User library is where objects you define will be stored, including firewalls, custom TCP and UDP services, and custom network ranges.

Figure 7-1. Firewall Builder policy management

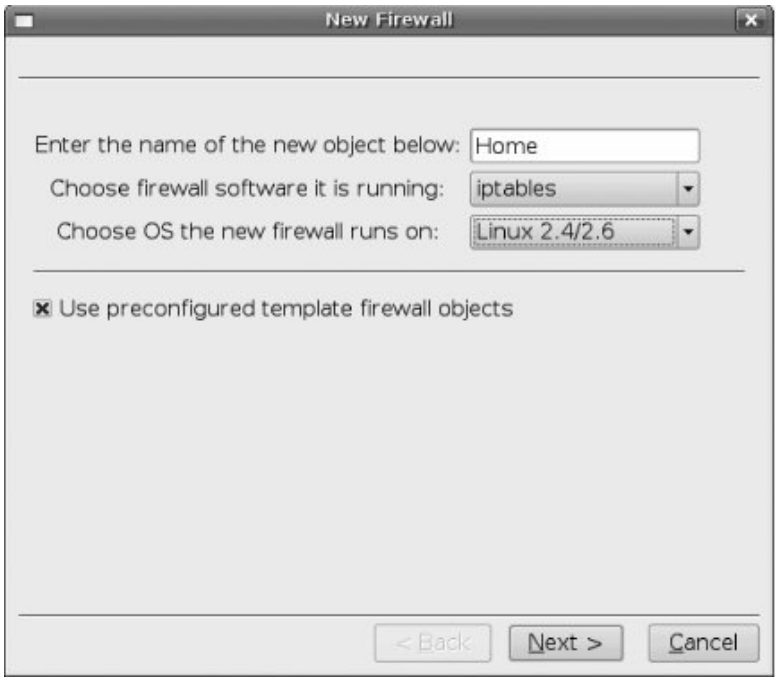


Define a new firewall

Make sure the User library is selected, right-click on the Firewalls folder, and select New Firewall. The New Firewall dialog appears, as shown in Figure 7-2. Give your firewall a name, select the firewall software (typically "iptables" if you run Linux on your firewall), and select the firewall operating system (Linux 2.4/2.6). This is where you start to see the flexibility of Firewall Builder and its support for multiple firewall types.

You also have the option of using preconfigured template firewall objects, which is a good idea if you're just getting started with Firewall Builder. The templates make it very easy to get started with a typical firewall scenario, rather than starting from scratch with a totally blank configuration. After you've made your selection, click Next.

Figure 7-2. New firewall

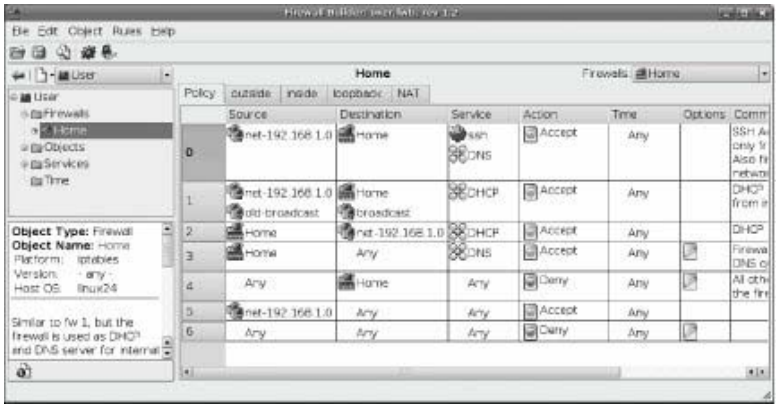


Click on each of the firewall template names to see a diagram and a brief explanation of how it works. For most small networks, "fw template 1" is a good choice, giving you a typical firewall with a dynamic external address, static internal addresses on the 192.168.1.0/24 network, unrestricted outbound access from the network, and access to

the firewall itself only via SSH from inside the network. "fw template 2" is similar but also allows the firewall to operate as a DHCP and DNS server for the internal network.

Once you have selected a template, you will be returned to the main Firewall Builder policy-management screen, but you will now have a default policy defined for your new firewall. Firewall Builder displays policy rules in a list, with the rules applied in order, starting at the top and working down until a match is found, as shown in Figure 7-3.

Figure 7-3. Policy management



There are multiple rules lists that can be accessed using tabs across the top of the list. The primary list is Policy, which are the rules that control allowed and disallowed activity throughout the firewall. Then there is an individual rule list for each interface on the firewall, and finally a NAT list that allows you to configure Network Address Translation rules. You don't usually need to worry about the individual interfaces; most changes will be to Policy and NAT.

Add a host-specific policy

To understand how Policy and NAT are managed, try applying a specific scenario, such as providing external access to an internal web server.

Start by adding a Host object for the server. Go through the User object tree to User⟶Objects⟶Hosts, right-click on Hosts, and select New Host. Enter a name such as *www.example.com* for your host, select the checkbox labeled Use Preconfigured Template Host Objects, and then click Next.

Select the "PC with 1 interface" template and click Finish. An object will be added to your Hosts list with an interface predefined, so click through to User⟶Objects⟶Hosts⟶<www.example.com>⟶eth0⟶<www.example.com>, then double-click on it to edit the interface values. Change the IP address to match the actual internal address of your server, apply the changes, and close the dialog.

Assuming you have a range of public, static IP addresses assigned to the external interface of your network, right-click on "eth0" and select Add IP Address. Enter the public IP address and network mask, apply changes, and close.

Your server now has two IP addresses defined: the real address assigned to its interface and the public address that you want people to use to access it.

Now click the NAT tab at the top of the rules list. Add a new, blank NAT rule by either right-clicking on the number of an existing rule or selecting Rules⟶Add Rule Below from the menu.

You can now build your NAT rule by dragging and dropping object icons into appropriate places. The columns are:

The original source address of packets before translation

Original Dst

The original destination of packets before translation

Original Srv

The original service (port) the packets arrived on

Translated Src

The new source address to report that packets came from

Translated Dst

The new destination address to apply to packets

Translated Srv

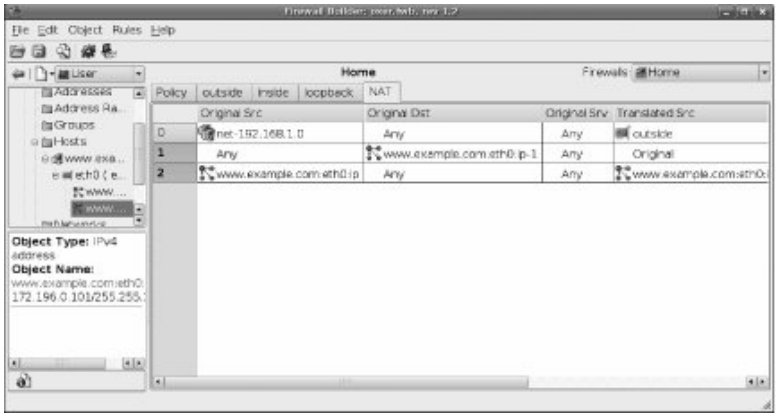
The new port to direct packets to

The first rule to set up is the translation to apply to packets directed toward the server from outside the firewall. Click the icon representing the external IP address of the server and drag it into the Original Dst box, and then click the internal IP address icon and drag it into the Translated Dst box.

Now add another blank rule and set it to translate packets directed from the server to the outside world. Click the internal IP address icon and drag it into the Original Src box, and the external IP address icon and drag it into the Translated Src box.

You now have rules that will cause packets traveling to and from the server to be modified as they pass through the firewall, with external machines seeing only the external IP address (see Figure 7-4).

Figure 7-4. NAT rules



In these examples, only the IP addresses were translated. However, you can also apply translations to services. For

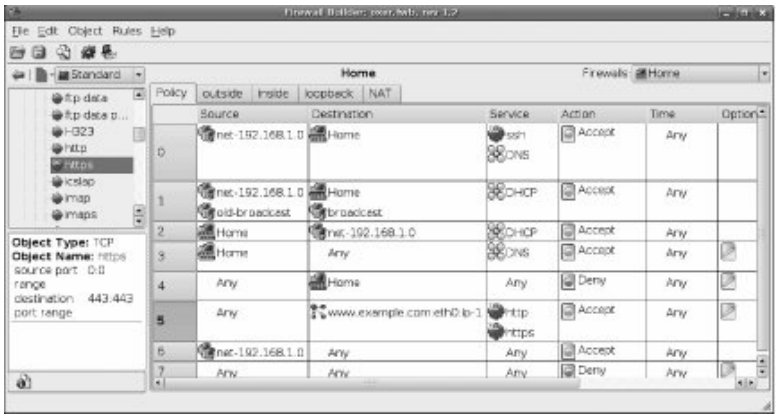
example, you may have a reverse-proxy cache running as a web-server accelerator internally on port 8080, and you want external users to be able to access it using the standard HTTP port, port 80. By dragging and dropping services into the Original Srv and Translated Srv boxes from the Standard library (and creating services in the User library as necessary), you can apply port translation in the same way as address translation. Firewall Builder gives you the flexibility to convert the source and destination addresses and port of packets at will, so adjust the NAT rules until they represent the transformations you want to apply to network traffic.

However, NAT rules alone are not enough. Without a matching policy rule, no packets will be allowed through the firewall, even if they match NAT rules. While the NAT rules define what can happen, policy rules define what is allowed to happen. Policy rules have the final say.

Click the Policy tab and go down through the list of existing rules to find an appropriate place to add rules for your host, most likely right near the end, before the general network rule and the fall-through Deny All rule. Right-click in the rule-number column and add a rule. Drag the external IP address icon of your web server into the Destination box, and then right-click the Deny icon in the Action column and change it to Allow.

If you want to allow full access to every port on your web server from outside your network, that's all you need to do. However, it's safest to make the rule more specific and allow only certain services through, so click the User/Standard drop-down near the top left and switch to the Standard library. Browse down through the library to Standard⟶Services⟶TCP⟶"http" and drag its icon into the Service box of your new rule. If you want to allow SSL connections, you can also drag the "https" icon into the same box. You now have a rule that explicitly allows connections from any host to your web server but only on ports 80 and 443 (see rule 5 in Figure 7-5).

Figure 7-5. Policy rules



Compile and Install the Policy

Once you are happy with the policy you have created, you need to apply it to the firewall. Firewall Builder does this in two steps. First, the policy is compiled into a script to suit the software on your firewall; then, it's pushed out to the firewall and loaded.

Select Rules⟶Compile or click the gear icon to have Firewall Builder compile your rules. You will see a dialog reporting progress, and when it finishes, you will have a new script placed in Firewall Builder's working directory alongside the project file. You can now copy the script to the target firewall by SSH and execute it to have the rules applied. To keep everything neat, it's a good idea to create a directory on the firewall to store the script:

```
$ sudo mkdir /etc/firewall
```

Then copy your script into that directory and execute it manually to test it. Even though it will have a .fw extension, the file is actually just a shell script that you can run in the normal way:

```
$ sudo /etc/firewall/                                    firewallname.fw
```

Firewall Builder provides a large number of configuration options to control the way the script is generated for each firewall, so if the script didn't work properly on your firewall, you may need to right-click the firewall icon in the object tree, select Edit, and then click the Firewall Settings button. The tabs at the top give you access to a lot of options, so go through them carefully and change any settings that may apply to your particular firewall; then compile the rules and test them again.

Automatic Policy Startup

The firewall script needs to be run each time your firewall boots up, so use your favorite editor to open */etc/rc.local* and add the path to the script just before the `exit 0` line. The end of */etc/rc.local* should look something like this:

```
/etc/firewall/firewallname.fw  exit 0
```

The *rc.local* file is executed after all the other startup scripts whenever Ubuntu switches to a new multiuser runlevel. Referencing your script in there ensures it will be executed after other services such as networking have started up, and that it won't be started if you boot up in single-user mode. This is handy if you need to fix configuration problems by booting into single-user mode.

Automatic Policy Installation

Once you are happy that the rules are being generated correctly for your firewall, you can save yourself some effort on subsequent updates by configuring Firewall Builder to manage installing and activating the script on your behalf. Select the firewall icon (located in User⟶Firewalls), right-click, select Edit, and the Firewall dialog will appear. Click Firewall Settings to re-enter the firewall options dialog. Click the Installer tab to be presented with options to execute a script to install and activate the firewall rules. The lower section of the dialog provides two text-entry fields that you can use to invoke any external script or command you like; so, for example, you can write a script that copies the script to the firewall by SCP and then executes it using SSH. Firewall Builder even comes with a sample script to do exactly what you will find installed in */usr/bin/fwb_install*. Full information on how to use fwb_install is available in its manpage:

```
$ man fwb_install
```

Firewall Builder also has an internal policy-installation mechanism, which is perfectly adequate for most environments.

To set up automatic policy installation, first create a group such as fwadmin on the firewall, and then create a user and make it a member of the group:

```
$ sudo addgroup fwadmin  $ sudo adduser fwadmin -G fwadmin
```

Set up a directory on the firewall to store the firewall configuration:

```
$ sudo mkdir -m 0770 /etc/firewall  $ sudo chown fwadmin:fwadmin /etc/firewall
```

Configure sudo to allow this user to execute the firewall script without entering a password by running:

```
$ sudo visudo
```

and adding a line similar to the following to the end of the /etc/sudoers file:

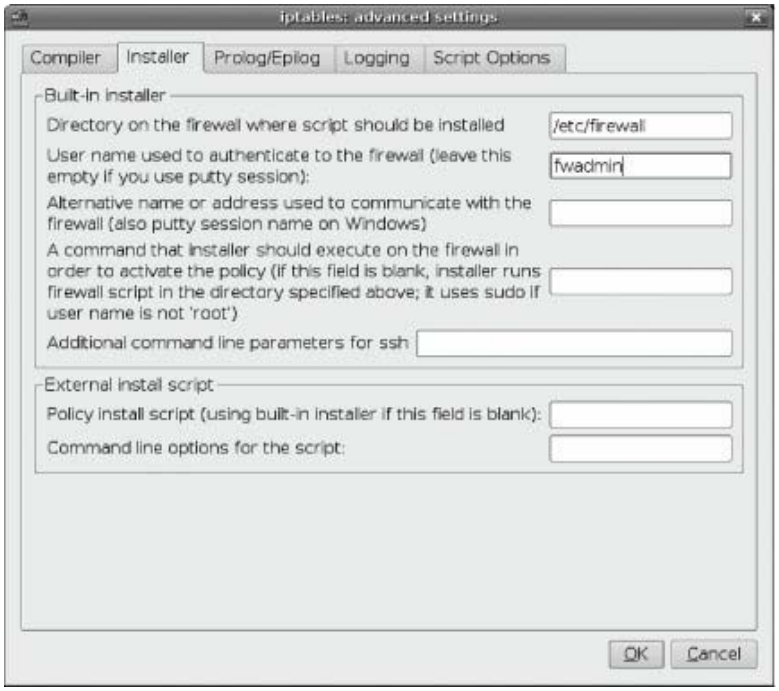
```
%fwadmin = NOPASSWD:/etc/firewall/firewallname.fw
```

The *firewallname.fw* string should be replaced by the actual name of the script generated by Firewall Builder for this firewall.

You can optionally set up public-key encryption for access to the fwadmin account on the firewall.

In the Installer tab of the Firewall Settings dialog, put in the path to the directory you created on the firewall and the username you created, as shown in Figure 7-6.

Figure 7-6. Policy installation options



You may not need to specify an alternate name or address for the firewall: Firewall Builder will try to automatically determine the IP address to use to communicate with the firewall, but it will use an address if you put one in.

Save and close the Firewall Settings dialog, and you should now be able to install and activate your policy simply by clicking the Install icon or by selecting Rules⟶Install from the menu. From now on, any updates you make can be applied simply by clicking Compile and then Install.

Extensive documentation and tutorials are available on the official Firewall Builder site at http://www.fwbuilder.org if you want to learn more.