Global Information Assurance Certification Paper

A Step-by-Step: How-To Build a Bridge Firewall
GSEC Practical Assignment
Version 1.2f
Arpandi Kasim
October 1, 2001

Introduction

History

Last time, in my company, all the machines was connected to the Internet with a public IP, some machines running as server, the rest is just a workstation. No perimeters in between, you can imagine how big is the risk! Until one day we decided to put a firewall, we moved all the workstation PCs behind firewall, but for the servers, we can't move it. Those servers provide very important data. Some of the machines are encoding machines that assigned with a fix IP [Diagram 1]. Until finally we found that one of the solutions is to use Transparent Firewall, or Bridge Firewall.

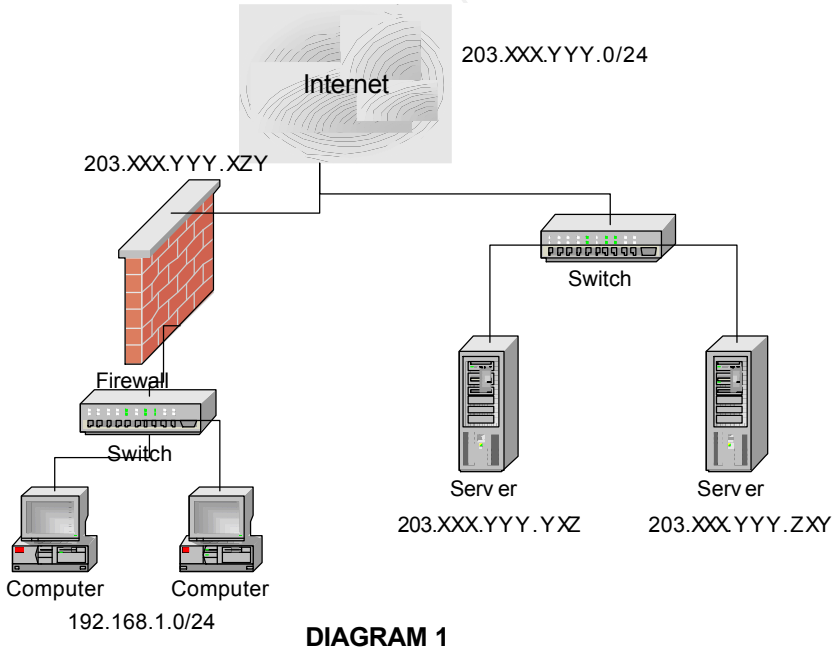


DIAGRAM 1

Overview

Bridge is a device that operates at layer 2, the data-link layer, of the OSI model for the purpose of reducing the amount of traffic that Ethernet hosts have to deal with on a particular segment. Traffic reduction is achieved by segmenting a large Ethernet segment into smaller segments[1]. This device will separate two or more network segments within one logical network (e.g. a single IP-subnet)[2].

Firewall is a collection of components, interposed between two networks that filter traffic between them according to some security policy [3].
A Bridge Firewall is a combination of a Bridge and a Firewall. With a Bridge Firewall, we can separate a single IP-subnet into two or more different network segments, and we can filter the traffics between those networks.

In Diagram 1, as you can see, our servers are running in our public network (203.XXX.YYY.ZZZ), with Bridge Firewall or Transparent Firewall, we can separate it into two different networks but still using the same IP, and the Bridge Firewall filter the traffics in between [Diagram 2].

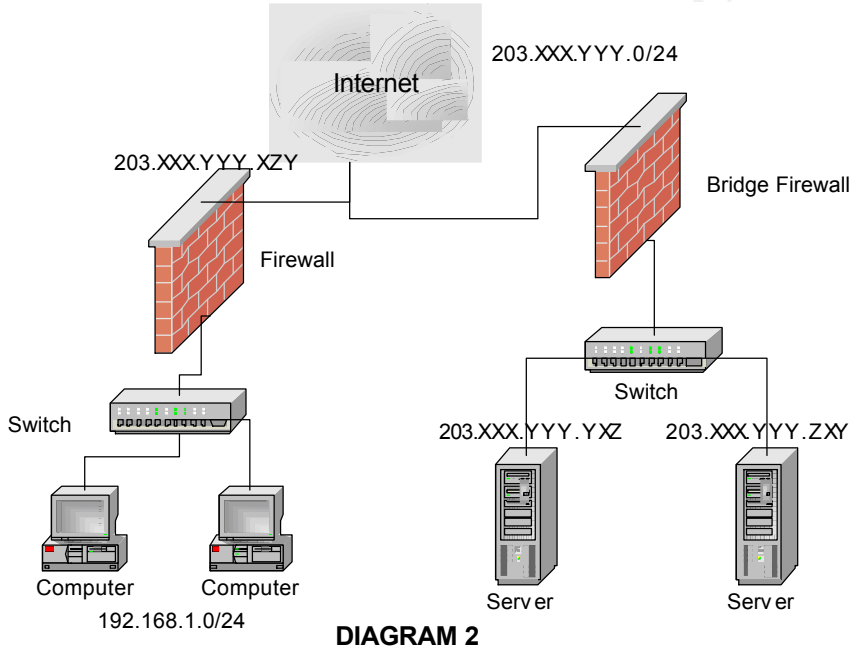


DIAGRAM 2

Setting up Bridge Firewall

In this paper, we will talk about bridging in Linux kernel 2.4.9, bridging module is a part of the mainstream for kernel 2.3.47 above, but it is not support the firewalling, to do the firewalling you will need to patch the kernel.

Patching the kernel

Some patches are needed for the bridge to support the netfilter or iptables firewalling. Remember that these patches are used to patch kernel 2.4.9, if you don't have kernel 2.4.9, you should download it. You can download the kernel from http://www.kernel.org
After you download the kernel, you will need to unzip/untar it.

```
root@bridgefw:~ # cd /usr/src
root@bridgefw:~ # rm linux
root@bridgefw:~ # tar zxf linux-2.4.9.tar.gz
root@bridgefw:~ # mv linux linux-2.4.9
root@bridgefw:~ # ln –s linux-2.4.9 linux
```

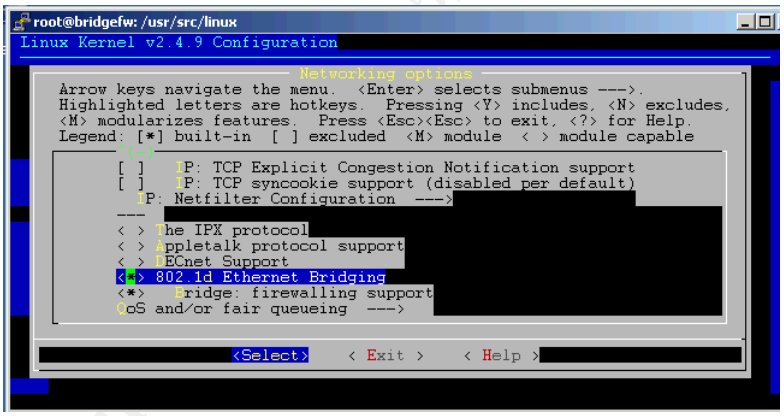
Example 1 (Unzip and create a new directory for kernel 2.4.9 in Redhat)

The patches can be downloaded from http://bridge.sourceforge.net/devel/bridge-nf/20010907-2/, save all the patches to the Linux source directory (usually all the patch file will have .diff extension), and then do the patching.

```
root@bridgefw:~ # patch –p1<patchesfilename.diff
```

After you patched the kernel, download *br_passthrough.c* from the URL above and then copy *br_passthrough.c* file to *net/bridge/netfilter/br_passthrough.c* in your Linux source directory.

Once you're done this part, compile the kernel with the new patches



Picture 1

In Picture 1, you can see a new choice in the kernel menu configuration, if you don't have this choice, that's mean you have done something wrong when you do the patching. The new choice is *Bridge: firewalling support*. Make sure you choose this module, before you compile the kernel.

```
root@bridgefw:~ # make dep clean bzImage modules modules_install
```

Example 2. (Compiling the kernel)

Configuring the bridge

After you done the compiling part, download and install the bridge utilities, you can download it from http://bridge.sourceforge.net/bridge-utils.html.

Now is the time to configure the bridge, Standard configuration should consist of:

1. Create the bridge interface
   root@bridgefw:~ # ***brctl addbr br0***

2. Add the interfaces to the bridge
   root@bridgefw:~ # ***brctl addif br0 eth0***
   root@bridgefw:~ # ***brctl addif br0 eth1***

3. Zero IP the interfaces
   root@bridgefw:~ # ***ifconfig eth0 0***
   root@bridgefw:~ # ***ifconfig eth1 0***

   Bridges are completely transparent, you don't need to assign IP addresses to the Ethernet interfaces.

4. Bring up the bridge
   root@bridgefw:~ # ***ifconfig br0 up***

   Or you can also set an IP to the bridge (not to the Ethernet interfaces),
   root@bridgefw:~ # ***ifconfig br0 203.XYZ.XYZ.XYZ up***

   With this command, the bridge will have an IP, this will allow you to communicate with the machine.

If the kernel panics after you bring up the bridge please check your network card, some network card like HP J2585A is not working properly for bridging.

Configuring the Firewall

The bridge is up! the next thing to do is setting the firewall. For kernel 2.4.9, we are using iptables for the firewalling part. You can download iptables from http://netfilter.samba.org/.

Basically any packet that entering your PC goes through INPUT chain, any packet that send out from your PC goes through OUTPUT chain, and any packet that your PC picks up on one network and sends to another network goes through the FORWARD chain[4].

Definition of those 3 chains in Bridge Firewall is like this:

1. INPUT: any packet that comes from outside to the Bridge machine (or machine that

running Bridge Firewall) will goes through this chain.
2. OUTPUT: any packet that goes out from the Bridge machine to outside will goes through this chain.
3. FORWARD: any packet that comes from one network in the bridge to another network in the bridge will goes through this chain.

This is a simple example rules

Iptables –A INPUT –s 203.XXX.YYY.0/24 –j ACCEPT
Iptables –A INPUT –j DROP

These rules will allow any connection from 203.XXX.YYY.0 – 203.XXX.YYY.255 (Subnet Class-C) to the bridge machine. The rest will be dropped.
These rules will not block any connection made to the other machines behind the Bridge Firewall. To filter the packets that go to any other machines behind Bridge Firewall, we'll need to set the rules in FORWARD chain.

Iptables –A FORWARD –s 203.XXX.YYY.0/24 –j ACCEPT
-s 203.XXX.YYY.0/24 is to tell the source
This will allow any connection from 203.XXX.YYY.0 - 203.XXX.YYY.255 to access the machines behind the Bridge Firewall.

Iptables –A FORWARD –p tcp –d 203.XXX.YYY.ZXY --dport 80 –j ACCEPT
-p tcp means the protocol is a TCP protocol
-d 203.XXX.YYY.ZXY is to tell the destination
--dport 80 means, any connection to port 80
This will allow outside accessing WWW (TCP port 80) on server 203.XXX.YYY.ZXY.

Iptables –A FORWARD –p tcp –d 203.XXX.YYY.ZXY --sport 80 –j ACCEPT
--sport 80 means, any connection that comes from port 80
This rule will allow any WWW connection from 203.XXX.YYY.ZXY to outside, or in other words, we can surf the web from 203.XXX.YYY.ZXY

Iptables –A FORWARD –j DROP
This rule will drop all packets that coming or going to or from the bridge, which not match to all the rules above.

To check how many packets that goes through your firewall, type ***iptables –L –v***

Visit this site http://www.linuxguruz.org/iptables/howto/iptables-HOWTO.html for more information about firewalling with iptables.

Testing the Bridge Firewall

For testing you can use Nmap port scanner, Nmap will show all opened/filtered port.
This is the result of the port scanning to one of the machine behind the Bridge Firewall from trusted network (203.XXX.YYY.0/24)

```
Starting nmap V. 2.54BETA29 ( www.insecure.org/nmap/ )
Interesting ports on  (203.XXX.YYY.ZXY):
(The 1536 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
25/tcp     open        smtp
80/tcp     open        http
135/tcp    open        loc-srv
139/tcp    open        netbios-ssn
443/tcp    open        https
445/tcp    open        microsoft-ds
1031/tcp   open        iad2
1433/tcp   open        ms-sql-s
3389/tcp   open        msrdp
5000/tcp   open        fics

Nmap run completed -- 1 IP address (1 host up) scanned in 11 seconds
```

At here you can see, all the port is open, this is happen because in the rules, we accepted all packets that comes from trusted (203.XXX.YYY.0/24).

Now we do a scanning from outside (not from trusted), and the result is

```
Starting nmap V. 2.54BETA29 ( www.insecure.org/nmap/ )
Interesting ports on  (203.XXX.YYY.ZXY):
(The 1547 ports scanned but not shown below are in state: filtered)
Port       State       Service
80/tcp     open        http

Nmap run completed -- 1 IP address (1 host up) scanned in 590 seconds
```

This result shown that the only port that opens is port 80 (WWW), which is the same as what we set on our rules, we only allow any connections from outside to the server's port 80.

Conclusion

Bridge Firewall in kernel 2.4 above still in experiment stuff, it's not stable yet, there are few bugs founded, one of the bugs that well known is, if netfilter ip_conntrack is loaded then the bridge will trash fragmented packets (the MAC level header gets replaced by garbage). But, from what we experienced, the Bridge Firewall is working properly for our needs, to filter out all packets that comes from or to the servers reside behind the Bridge Firewall.

References:

1. Parkhurst. R. William, Ph.D., CCIE #2969
   Cisco Router OSPF McGraw-Hill, 1998

2. Böhme, Uwe and Buytenhenk, Lennert
   Linux BRIDGE-STP-HOWTO Release v.0.04
   http://www.bnhof.de/~uwe/bridge-stp-howto/BRIDGE-STP-HOWTO Jan 11, 2001

3. W. R. Cheswick and S. M. Bellovin.
   Firewalls and Internet Security: Repelling the Wily Hacker Addison-Wesley, 1994

4. Prince_Kenshi
   Iptables Basics NHF
   http://www.linuxnewbie.org/nhf/intel/security/iptables_basics.html

5. W. Daniel
   Distributed Firewall
   http://www.sans.org/infosecFAQ/firewall/dist.htm May 28, 2001

6. **Bridge Project**
   http://bridge.sourceforge.net Nov 27, 2000

7. Ward, Brian
   The Linux Kernel HOW-TO v3.0
   http://www.linuxdoc.org/HOWTO/Kernel-HOWTO.html July 15, 2001

8. **The Netfilter Project**
   http://netfilter.samba.org/

9. **The Bridge Mailing list Archives**
   http://www.math.leidenuniv.nl/pipermail/bridge/