BI-SSB - Lab Manual 1
Different Firewalling Techniques for Routers in GNS3 use router firmware 3725

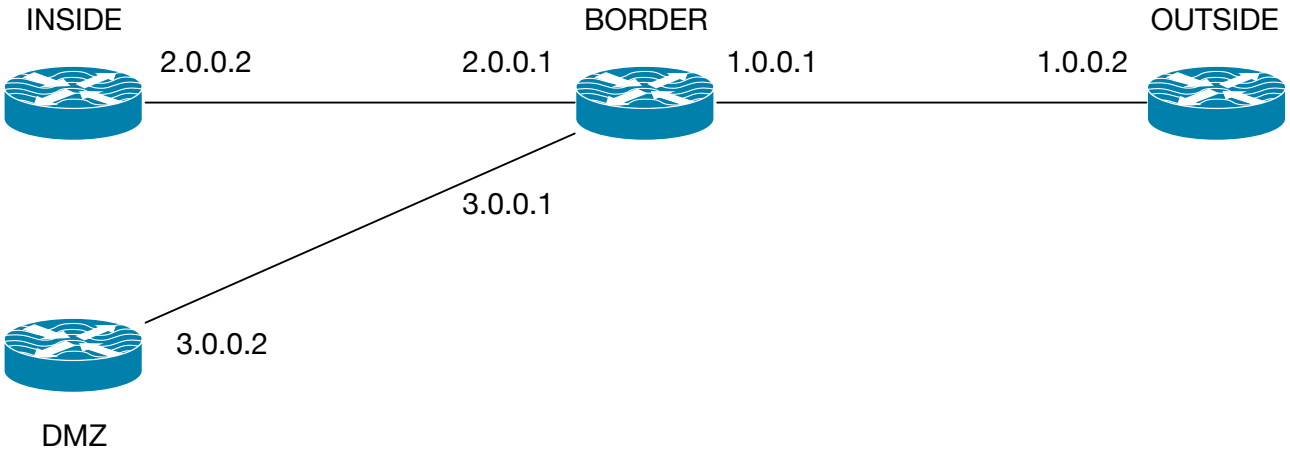
1. Introduction

In the first lab we had an introduction to the GNS3 simulator. As a test we implemented a simple expansion to classic accesslists which is reflexive accesslists. Keep in mind that there are many types of accesslists which can be used in different combinations: standard, extended, names, numbered, ip based, reflexive, established, timed, security, etc.

The network topology which we shall use during the semester simulates the zones in an enterprise: inside, outside and DMZ (demilitarised zone) - their purpose will be given during lectures. For simplicity all of these are connected to one router which represents the enterprise edge or the border router or the gateway - as presented in the BI-PSI subject.

The diagram is presented in Figure 1 and we suppose for the purpose of BI-SSB the fact that all the IP addresses are public. Of course, the only thing necessary for this topology to work with private IP addresses is activating PAT (or NAT overload - the PAT in the vision of Cisco).

Figure 1 - The diagram of the network



In this diagram all the masks are supposed to be /8, thus 255.0.0.0 and thus any needed wildcard (inverted mask) will be 0.255.255.255. The routers BORDER, INSIDE and DMZ are supposed to belong to out company while the OUTSIDE router is supposed to belong to the ISP (Internet Service Provider).

The terms “inbound” and “outbound” are referred from the perspective of our company, thus inbound traffic is traffic from OUTSIDE to any other router. Outbound traffic is traffic from any router to OUTSIDE.

The terms “incoming” and “outgoing” are referred to a router, representing traffic coming to and respectively going from that particular router. Thus, from the perspective of router BORDER, incoming traffic may be from any other router in the topology and outgoing traffic is towards any other router of the topology.

2. Configuring the basic settings

Before we apply security we need connectivity first. So, let us configure the devices for full connectivity.

2.1. Analysis:

On router INSIDE we need: ip address on interface and default gateway.

On router DMZ, the same as on router INSIDE, we need: ip address on interface and default gateway.

On router BORDER we need: ip addresses on all interfaces and here we shall apply security mechanisms, this being the security device in our network. We do not need routing because it is directly connected to all other networks.

On router OUTSIDE we need: ip address and static routes to the networks 2.0.0.0 and 3.0.0.0. From this moment on, instead on network 2.0.0.0 I shall say network “2” - the same notation will be applied to other networks.

2.2. Cisco IOS modes and commands

Actual commands are given in *italic font*.

> = normal mode (you can view some parameters but you cannot view advanced things nor you can configure)

= privileged mode. You enter into this by typing at > the command *enable*. From # you type *disable* to return to >

config# = this is the configuration mode. You enter here from # by typing *configure terminal*. In this mode you cannot view, unless you trigger the command using *do*, like for example: *do show ip route*.

Any command can be cancelled by prefixing it with *no*. Thus, to cancel *ip route 0.0.0.0 0.0.0.0 1.0.0.1* you type *no ip route 0.0.0.0 0.0.0.0 1.0.0.1*.

Save your configuration: router# *write memory*

2.3. The configuration

At the end of each command press enter.

2.3.1. INSIDE- full configuration, commented

```
router> enable                                  // normal mode, we enter privileged
router#                                         // we are in privileged mode
router# configure terminal                      // we enter config mode
router-config#                                  // we are in the config mode
router-config# hostname INSIDE                  // we change the name of the router to INSIDE
INSIDE-config#
INSIDE-config# interface fastEthernet 0/0       //we then enter to configure the interface
                                                // connected to BORDER. WARNING: it may
                                                // be a different interface on your
                                                // configuration!!! Use yours!!!
INSIDE-config-if#
INSIDE-config-if# ip address 2.0.0.2 255.0.0.0  // the set of ip address
```

INSIDE-config-if# *no shutdown* // turns on the interface
INSIDE-config-if# *exit* // we return to the global configuration mode
INSIDE-config#
INSIDE-config# *ip route 0.0.0.0 0.0.0.0 2.0.0.1* // we set up the default gateway
INSIDE-config# *end* // we exit config mode
INSIDE#
INSIDE# *write memory* // we save the configuration

2.3.2. DMZ - only important commands

For the exact commands use the example given in 2.3.1.
Set up the hostname as DMZ.
On interface: *ip address 3.0.0.2 255.0.0.0* and *no shutdown*
Set default gateway to 3.0.0.1
Save the configuration.

2.3.3. OUTSIDE - only important commands

For the exact commands use the example given in 2.3.1.
Set up the hostname as OUTSIDE.
On interface: *ip address 1.0.0.2 255.0.0.0* and *no shutdown*
Set gateways to networks 2 and 3:

OUTSIDE-config# ip route 2.0.0.0 255.0.0.0 1.0.0.1
OUTSIDE-config# ip route 3.0.0.0 255.0.0.0 1.0.0.1

Save the configuration.

2.3.4. Testing

From each router in # mode test connectivity by using *ping* followed by ip address. You should see !!! which means OK.

You can also view the content of the routing table by issuing: # *show ip route*

The configuration of the interfaces can be seen at: # *show ip interface brief*

The configuration at this step is named a “clean configuration”.

3. Accesslist - ACL

Purpose: if applied to an interface it blocks the traffic. If not applied, it just matches the traffic without doing anything. At the end of an ACL (Access Control List) there is an implicit denial for all ip traffic (deny ip any any), which is invisible.

An ACL should be applied as close as possible to the source of the traffic, on the interface closest to the source of the traffic, on the direction in / out from the perspective of the router on which is applied, if it is incoming or outgoing traffic.

We start with a clean configuration.

Example: let us block the traffic from OUTBOUND to INSIDE but not DMZ.

All the settings are done on BORDER router, as it is our security device.

Define an ACL:

config# *ip access-list extended BLOCK-OUTSIDE-IN*
deny ip 1.0.0.0 0.255.255.255 2.0.0.0 0.255.255.255
permit ip 1.0.0.0 0.255.255.255 3.0.0.0 0.255.255.255

We apply the ACL on the interface towards router OUTSIDE, but on the router BORDER:

config-if# ip access-group BLOCK-OUTSIDE-IN in

Test from any router connectivity to any router. INSIDE should be able to ping BORDER and DMZ but not OUTSIDE (the ping actually goes but the reply does not), DMZ should ping successfully INSIDE, BORDER and OUTSIDE. OUTSIDE should be able to ping DMZ and BORDER but not INSIDE.

You can view the triggered ACL by typing # *show access-lists*

Save the configuration on all devices.

4. Reflexive accesslists

Purpose: to allow inbound traffic only for the requests generated outbound, from within the company not anything else.

We start with a clean configuration.

Example: we shall allow outbound ping but not inbound ping.

All the settings are done on BORDER router, as it is our security device.

Define a reflexive ACL which keeps track of outbound traffic in the record LOG:

config# *ip access-list extended PERMIT-OUTBOUND* //last param is the name of ACL
permit ip 2.0.0.0 0.255.255.255 1.0.0.0 0.255.255.255 reflect LOG
permit ip 3.0.0.0 0.255.255.255 1.0.0.0 0.255.255.255 reflect LOG
permit ip 2.0.0.0 0.255.255.255 3.0.0.0 0.255.255.255
permit ip 3.0.0.0 0.255.255.255 2.0.0.0 0.255.255.255
deny ip any any

Keep in mind that the last line you may skip as it is anyway implicit. Also note that the previous two lines are the default communication allowed within the company between the INSIDE and DMZ zones, thus no restriction will be applied.

We apply this ACL on the interfaces towards INSIDE and DMZ using the same command (executed once on each interface):

config-if# *ip access-group PERMIT-OUTBOUND in*

We now define a matching reflexive ACL which allows only inbound traffic matching an outbound connection:

config# *ip access-list extended BLOCK-INBOUND*
evaluate LOG
deny ip any any

And we apply it on the interface towards OUTSIDE:

config-if# *ip access-group BLOCK-INBOUND in*

Test connectivity: from INSIDE and DMZ you should successfully ping OUTSIDE. From OUTSIDE you should not be able to ping INSIDE ow DMZ. Firewall.

You can view the triggered ACL by typing # *show access-lists*

Save the configuration on all devices.

Conclusion: it works but only with static configuration parameters. No analysis of ports, TCP or UDP, anything. In chapter 5 we add one element: transport layer identification.

5. CBAC - Context Based Access Control

Purpose: more advanced than reflexive ACL, it allows for the identification of traffic. Can we have traffic through an interface on which a deny ip any any is applied? Sure, by using CBAC.

We start with a clean configuration.

All the settings are done on BORDER router, as it is our security device.

Let us build an ACL denying all traffic:

config# *ip access-list extended DENY-ALL*
deny ip any any

You realise that this could have been an empty ACL (as the implicit denial is there anyway).

Let us apply it on the interface towards OUTSIDE:

config-if# *ip access-group DENY-ALL in*

In this moment you cannot ping OUTSIDE from INSIDE, DMZ or BORDER. The ping goes there but the reply will be dropped.

Now in the global config we ask for the inspection of the interesting traffic. Let us say TCP and ICMP for today.

config# *ip inspect name LOG tcp*
ip inspect name LOG icmp

The traffic is now inspected.

On the interface towards OUTSIDE we allow the inspection to take precedence over the blocking ACL:

config-if# *ip inspect LOG out*

Testing: try ping from BORDER, INSIDE, DMZ to the OUTSIDE.

You can also test tcp communication by implementing some telnet servers on the 4 routers and try to connect to them.

Save the configuration on all devices.

6. Control Plane Management

In the lectures I talked about control plane as being one of the three planes: data plane, management plane and control plane. It allows us to impose rules on the data traffic bypassing all the mechanisms. In this example I shall show you that we can slow down ping, essentially allowing it but with a considerably slower speed.

For this example we start with a clean configuration and we use only routers BORDER and OUTSIDE.

All the settings are done on BORDER router, as it is our security device.

Let us match ICMP (ping) traffic using a simple, standard ACL called 100:

config# *access-list 100 permit icmp any any*

We define next a class map which will be triggered by the ACL. A class map allows very fine matching of data traffic but for this example we focus only on protocols. Class maps are part of the QoS mechanism. Class maps can match ip addresses, port numbers as well as do deep packet inspection for as long as the data field of the ip packets is not encrypted (as it happens in VPN).

config# *class-map ICMPCM*
match access-group 100

We then define a policy for the protocol and we police the data rate at 8000 bps for the ICMP traffic. The policy is triggered by the class map.

config# *policy-map ICMPPM*
class ICMPCM // the class triggers the effect of policing
police 8000 conform-action transmit exceed-action drop

// this means: for as long as your ping rate is within the rate of 8000 bps the traffic is allowed to be transmitted but any exceeding traffic will be dropped

And now we apply this policy map on the interface towards OUTSIDE:

config# *control-plane*
service policy input ICMPPM

You see the control plane command :) Now let us flood from the router OUTSIDE our router BORDER. Type in # *ping* and use the ? Commands to generate 1000 ping packets. You will see what happens, how the router pulls the break on your traffic using the control plane.

Save the configuration on all devices.

7. ZBFW - Zone Based Firewall - use 3725 Router

The ZBFW is based on the fact that inspection up to this moment was defined on a per interface basis. This becomes complicated when we have routers with a lot of interfaces. However, the concept of ZBFW - as its name says - is baed on zones. The inspection takes place between zones. Let us focus on two simple protocols: TELNET and ICMP.

Let us configure TELNET on the routers OUTSIDE (simulating a server in the Internet), on DMZ (simulating a public server of our company) on INSIDE (simulating a private internal server).

The configuration of TELNET is:

```
config#      usename mama privilege 15 secret tata     //a user mama with password tata and
                                                       // privilege level 15 - root equivalent

config#      line vty 0 15             //virtual terminal lines allowing 16 simultaneous connections
config-line# transport input telnet    //enable telnet as protocol
             login local               //use for authentication the local username database
```

Do this on all the routers: OUTSIDE, INSIDE, DMZ. Test TELNET connectivity by trying to remotely connect between devices. Take care that in reality TELNET is NOT ENCRYPTED and unless it is encapsulated as encrypted payload over the Internet, all passwords can be intercepted. I shall show you this during the lab, using Wireshark.

The test is done simply by typing: # *telnet* and IP of the target device.

Now let us define ZBFW. ZBFW has the following theoretical rules:

- Traffic between zones which is specified will be inspected and a decision of drop, permit or classify is taken based on the policy maps;
- Traffic to and from zone SELF is by default permitted (self zone is the router itself);
- All other traffic is by default denied / dropped.
- All traffic within one zone is by default permitted.

On router BORDER we are going to define 3 zones for this example: INSIDE, OUTSIDE and DMZ.

```
config#   zone security INSIDE
          zone security OUTSIDE
          zone security DMZ
```

After the configuration of each zone you need to exit in order to define the next zone.

Now we associate each interface to its proper zone:

```
Towards INSIDE router config-if#    zone-member security INSIDE
Towards OUTSIDE router config-if#   zone-member security OUTSIDE
Towards DMZ router config-if#       zone-member security DMZ
```

Then we define a class map which will match our desired protocols:

```
config#   class-map type inspect match-any CM
          match protocol telnet
          match protocol icmp
```

Then we define a policy map which will inspect what is defined in the class map:

```
config#   policy-map type inspect PM
          class type inspect CM      // it will be triggered by this class map
          inspect                    // this is what we do: we inspect the traffic
```

Now we need to define what happens at the border of two zones and apply the policy:

```
config#   zone-pair security OUTBOUND source DMZ destination OUTSIDE
          service policy type inspect PM
```

Test the telnet and ping and watch what happens.

You can also have a more detailed view using the commands:

show policy-map type inspect zone-pair

show policy-map type inspect zone-pair session // for TCP sessions like TELNET