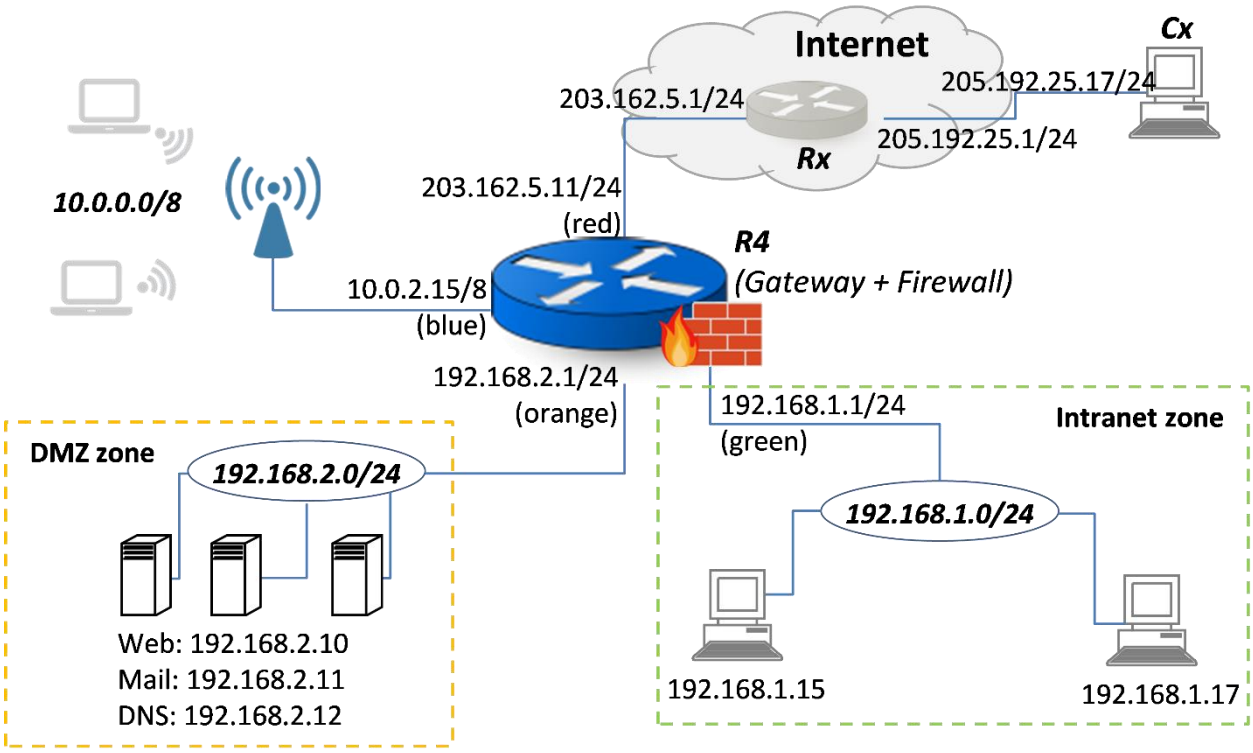
Thiết kế & triển khai mạng IP

Bài thực hành số 5: Bảo mật trên mạng

Bài 1: Cài đặt và thiết lập cấu hình Firewall



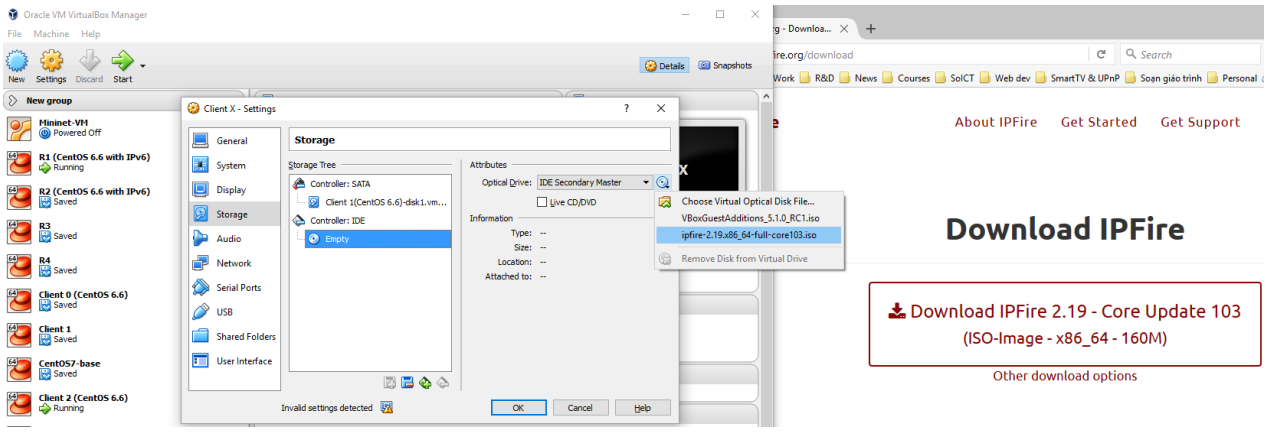
Bài này yêu cầu sử dụng firewall IP Fire trên Linux để thiết lập một tường lửa cho mạng Intranet của công ty theo sơ đồ bên trên. Vùng DMZ (orange) chứa các máy chủ cần có thể truy nhập từ ngoài Internet, vùng Wifi (blue) cho phép các trạm di động kết nối vạo hệ thống mạng. Vùng Intranet (green) là cùng làm việc nội bộ của công ty. Cuối cùng, firewall có một kết nối với Internet (red). Các bước thực hiện như sau:

- Chuẩn bị cài đặt firewall
- Cài đặt và thiết lập cấu hình IPFire
- Bắt đầu làm việc với IPFire qua giao diện Web
- Mở cổng *ssh* để kết nối từ một trạm vùng green

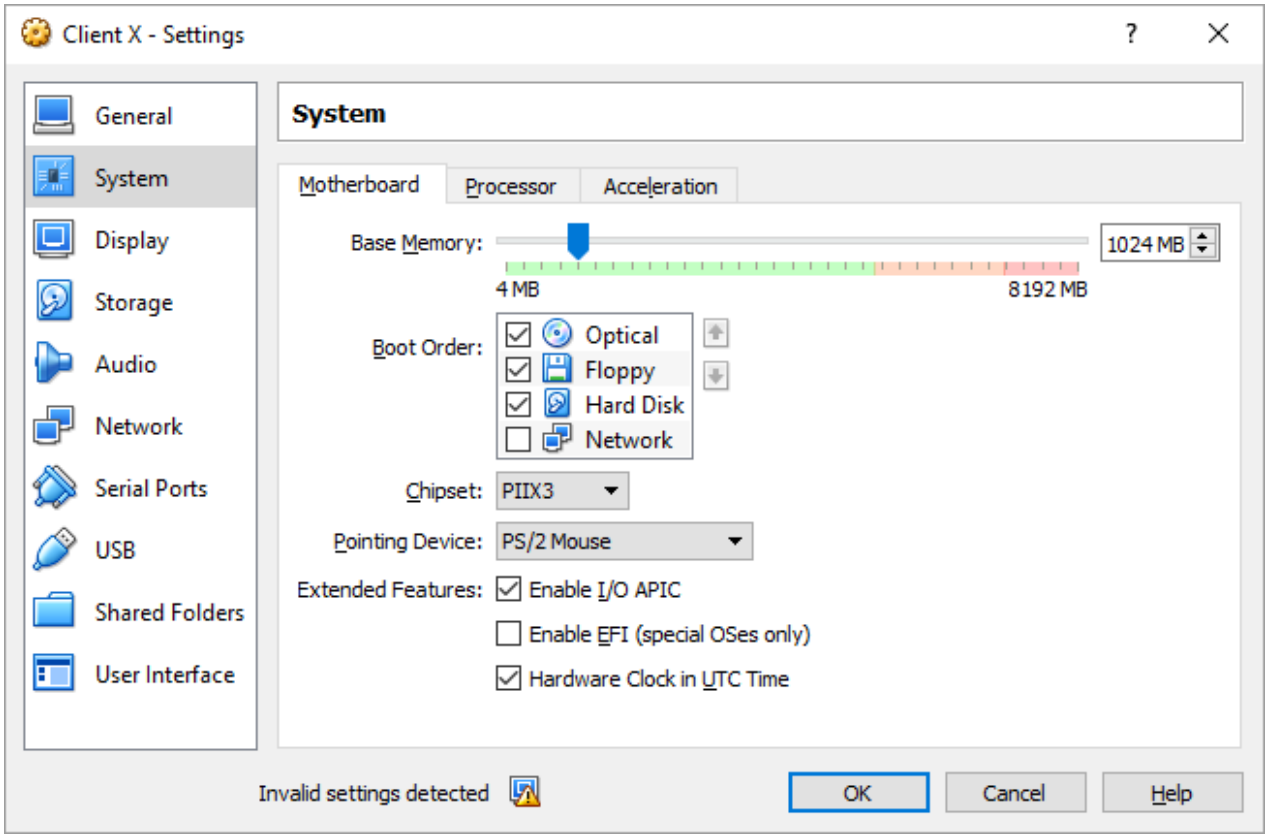
Bước 1: Chuẩn bị cài đặt firewall

Để cài đặt firewall IPFire lên một máy Linux làm Gateway cho mạng Intranet, phương pháp đơn giản nhất là download file ISO-image trên trang web IPFire và cài đặt nó như một hệ điều hành của máy Gateway. Phương pháp này thực hiện cài đặt và cấu hình

firewall IPFire trước, sau đó tùy theo nhu cầu quản trị mà admin có thể bổ sung thêm các công cụ phần mềm cần thiết vào firewall để phục vụ công việc quản trị mạng hoặc cấu hình các tính năng của router. Trường hợp Gateway đã tồn tại và cần bổ sung thêm firewall IPFire, cần download bản cài đặt *rpm* phù hợp với hệ điều hành Gateway hoặc build IPFire từ các file nguồn IPFire. Ở đây ra sử dụng cách thứ nhất. File ISO-image sau khi được download về cần được gán vào đĩa CDROM của máy ảo (thiết lập trong mục Storage):



Tiếp theo, thiết lập Optical là thứ tự khởi động đầu tiên cho máy ảo (mục System). Điều này cho phép khi khởi động máy ảo, nó sẽ tìm đến đĩa ISO-image IPFire để khởi động và từ đó bắt đầu các bước cài đặt IPFire vào hệ điều hành máy Gateway.



Theo thiết kế kiến trúc mạng, firewall IPFire sử dụng 4 kết nối mạng để kết nối 4 vùng khác nhau (gọi là các vùng red, orange, blue và green). Cần thiết lập 4 card mạng cho máy ảo Gateway. Sau đó khởi động máy ảo để bắt đầu cài đặt và thiết lập cấu hình IPFire.

Bước 2: Cài đặt và thiết lập cấu hình IPFire

Khi khởi động máy Gateway lần đầu tiên từ đĩa ISO-image IPFire, các bước cài đặt và cấu hình được lần lượt được hiển thị. Thực hiện đúng theo các chỉ dẫn trên màn hình và tham khảo các bước cài đặt ở đây: *http://wiki.ipfire.org/en/installation/step5*. Lưu ý khi lựa chọn cấu hình mạng cần lựa chọn đủ 4 kết nối (red, orange, blue và green) và thiết lập địa chỉ IP cho các kết nối đúng theo sơ đồ thiết kế mạng. Kết quả thiết lập cấu hình mạng như sau:

```
[root@ipfire ~]# ifconfig -a
blue0     Link encap:Ethernet  HWaddr 08:00:27:65:41:33
inet addr:10.0.2.15  Bcast:10.255.255.255  Mask:255.0.0.0
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
green0    Link encap:Ethernet  HWaddr 08:00:27:D5:B5:32
inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:419 errors:0 dropped:0 overruns:0 frame:0
TX packets:243 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:80590 (78.7 Kb)  TX bytes:62663 (61.1 Kb)
orange0   Link encap:Ethernet  HWaddr 08:00:27:FC:E4:6C
inet addr:192.168.2.1  Bcast:192.168.2.255  Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
red0      Link encap:Ethernet  HWaddr 08:00:27:A8:BE:90
inet addr:203.162.5.11  Bcast:203.162.5.255  Mask:255.255.255.0
UP BROADCAST RUNNING  MTU:1500  Metric:1
RX packets:4 errors:0 dropped:0 overruns:0 frame:0
TX packets:84 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:1368 (1.3 Kb)  TX bytes:5040 (4.9 Kb)
```

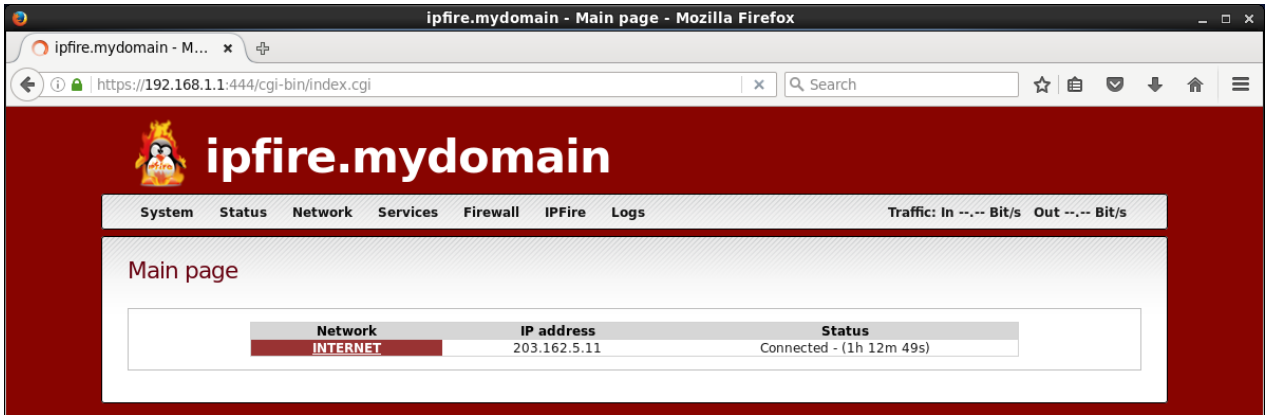
Ngoài ra, firewall IPFire kết nối ra Internet thông qua một router *Rx* có địa chỉ 203.162.5.1 nên default gateway của firewall này là 203.162.5.1:

```
[root@ipfire ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         203.162.5.1     0.0.0.0         UG    0      0        0 red0
10.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 blue0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 green0
```

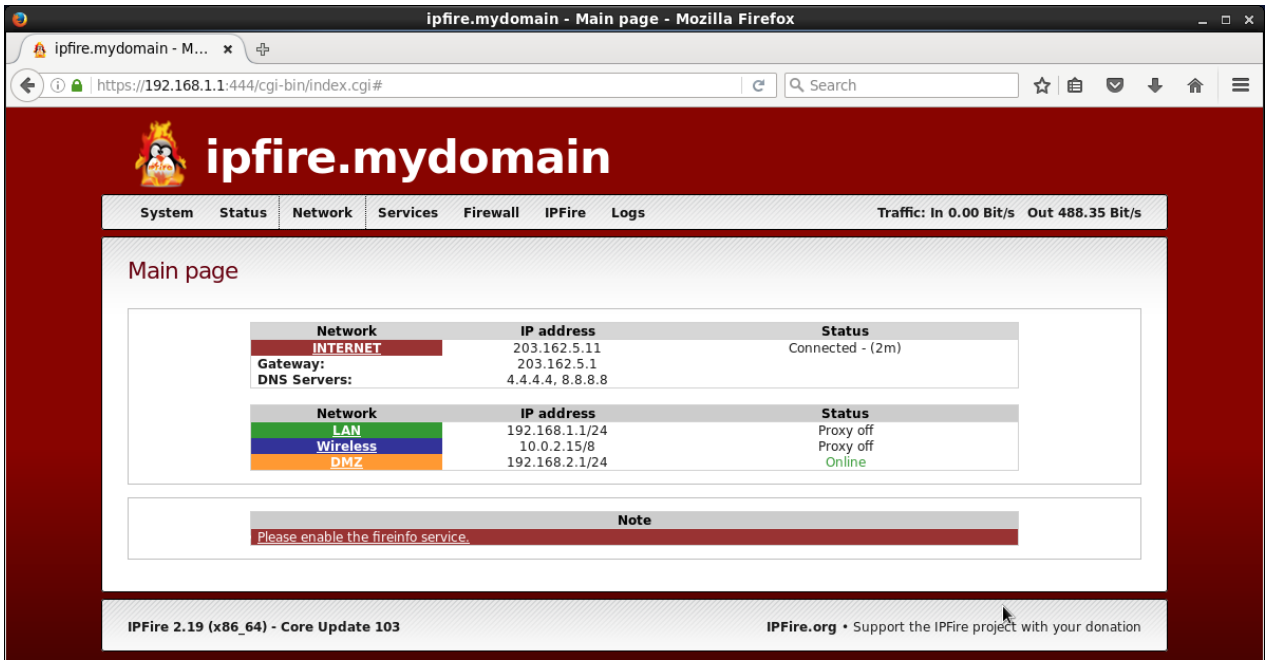
```
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 orange0
203.162.5.0     0.0.0.0         255.255.255.0   U     0      0        0 red0
```

Bước 3: Bắt đầu làm việc với IPFire qua giao diện Web

IPFire sau khi khởi động mặc định sẽ cho phép kết nối từ một trạm trong vùng green để thực hiện thiết lập các thông số cấu hình. Đứng trên trạm 192.168.1.5, mở web browser và kết nối vào IPFire theo địa chỉ *https://192.168.1.1:444*. Sau khi xác thực user admin (password được thiết lập khi cài đặt IPFire), giao diện web của IPFire như sau:

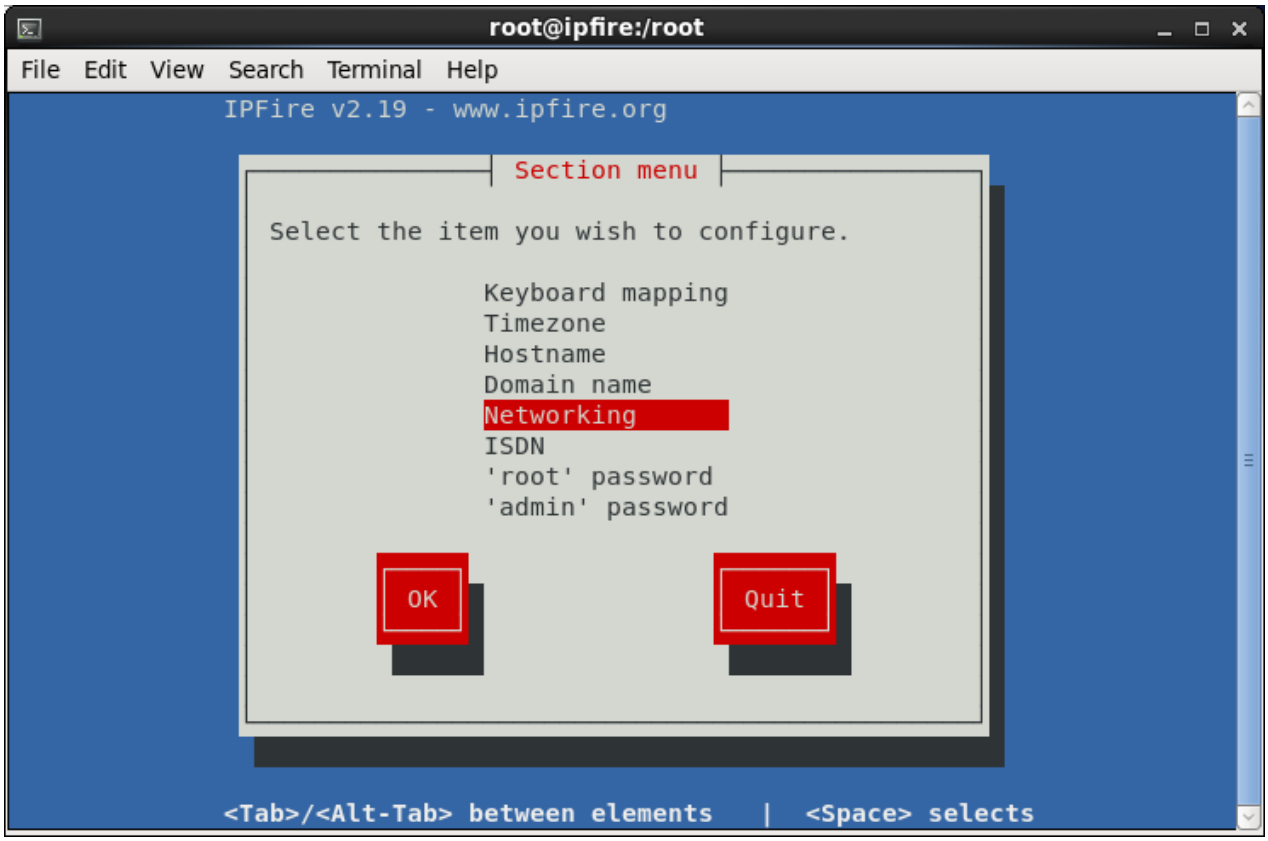


Có thể kiểm tra các thông số kết nối với 4 vùng red, orange, blue và green bằng cách chọn mục Network trên giao diện web:



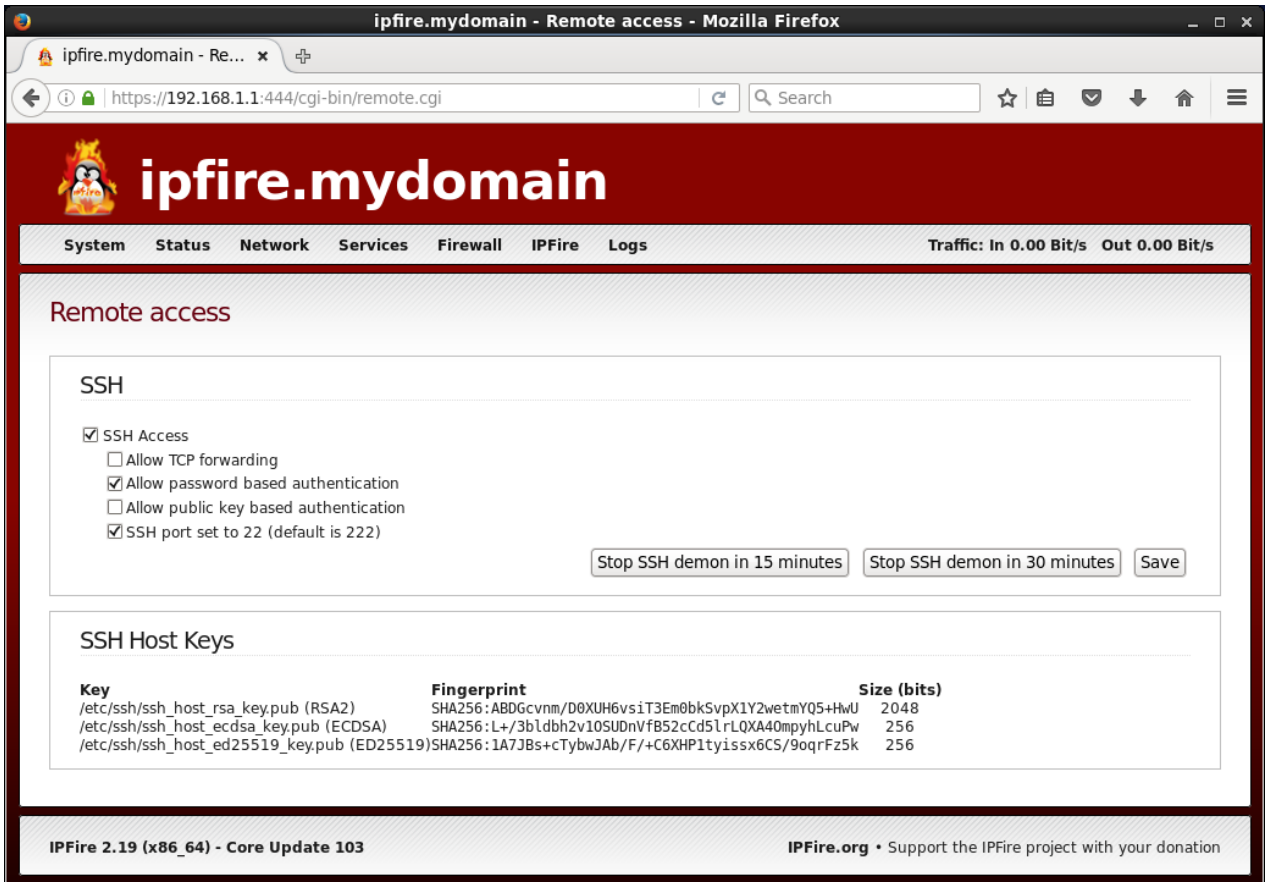
Các thông số này có thể được thiết lập lại bằng cách thực hiện lệnh *setup* trong *console* của IPFire:

[root@ipfire ~]# **setup**



Bước 4: Mở cổng ssh để kết nối từ một trạm vùng green

Mặc định, IPFire cấm kết nối *ssh* và chỉ cho phép một kết nối duy nhất là giao diện Web. Trong nhiều trường hợp, cần thực hiện các lệnh trực tiếp trên console của IPFire thay vì qua giao diện web (ví dụ như khi thay đổi địa chỉ IP cho các kết nối mạng red, orangem blue hay green). Vào giao diện web, mục System/SSH Access, và thiết lập cho phép SSH Access. Lưu ý rằng vì lý do an ninh, cổng truy nhập ssh của IPFire được thiết lập mặc định là 222. Có thể đổi *ssh* về cổng chuẩn 22 bằng cách lựa chọn “SSH port set to 22”.



Sau khi thiết lập cho phép *ssh*, kiểm tra lại hệ thống *ssh* đã thông bằng một kết nối *ssh* từ máy trạm 192.168.1.15.

Bài 2: Thiết lập rule cho các vùng orange, green và blue

Mặc định, IPFire thiết lập các qui tắc cho phép/cấm kết nối giữa các mạng như sau:

	Direction		Status
Red	→	Firewall	Closed, Use external access
Red	→	Orange	Closed. Use port forwarding
Red	→	Blue	Closed. Use port forwarding or VPN
Red	→	Green	Closed. Use port forwarding or VPN
Orange	→	Firewall	Closed, No DNS nor DHCP for Orange
Orange	→	Red	Open
Orange	→	Blue	Closed, use DMZ pinholes
Orange	→	Green	Closed, use DMZ pinholes
Blue	→	Firewall	Closed, no access for Blue
Blue	→	Red	Closed, no access for Blue
Blue	→	Orange	Closed, no access for Blue
Blue	→	Green	Closed, use DMZ pinholes or VPN
Green	→	Firewall	Open
Green	→	Red	Open
Green	→	Orange	Open
Green	→	Blue	Open

Có thể thấy rằng tất cả các truy nhập từ Internet (vùng red) vào các vùng phía trong firewall (green, blue và orange) đều bị cấm (Closed). Để cho phép một truy nhập, cần thiết lập luật "port forwarding" tương ứng. Ngược lại, các truy nhập từ cùng nội bộ (green) hoặc DMZ (orange) ra Internet đều được phép (Open). Dựa trên bảng qui tắc trên, cần tạo ra các luật (firewall rule) để cấm hoặc cho phép các kết nối mạng.

Bước 1: Kiểm tra kết nối giữa các vùng

Có thể sử dụng lệnh *ping* hoặc *iperf3* để kiểm tra kết nối giữa các vùng. Ví dụ, có thể ping từ một trạm cùng green hoặc orange đến một trạm ngoài Interent và thấy thành công.

```
[root@C2 ~]# ping 205.192.25.17
PING 205.192.25.17 (205.192.25.17) 56(84) bytes of data.
64 bytes from 205.192.25.17: icmp_seq=1 ttl=62 time=3.17 ms
64 bytes from 205.192.25.17: icmp_seq=2 ttl=62 time=1.91 ms
64 bytes from 205.192.25.17: icmp_seq=3 ttl=62 time=2.16 ms
^C
--- 205.192.25.17 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2181ms
rtt min/avg/max/mdev = 1.918/2.419/3.177/0.546 ms
```

Nếu *ping* từ green sang orange hoặc ngược lại, từ orange sang blue cũng thành công:

```
[root@C2 ~]# ping 192.168.2.3
PING 192.168.2.3 (192.168.2.3) 56(84) bytes of data.
64 bytes from 192.168.2.3: icmp_seq=1 ttl=63 time=1.49 ms
64 bytes from 192.168.2.3: icmp_seq=2 ttl=63 time=1.44 ms
64 bytes from 192.168.2.3: icmp_seq=3 ttl=63 time=1.14 ms
^C
--- 192.168.2.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2178ms
rtt min/avg/max/mdev = 1.142/1.360/1.494/0.161 ms
[root@C1 ~]# ping 192.168.1.15
PING 192.168.1.15 (192.168.1.15) 56(84) bytes of data.
64 bytes from 192.168.1.15: icmp_seq=1 ttl=63 time=1.26 ms
64 bytes from 192.168.1.15: icmp_seq=2 ttl=63 time=1.28 ms
64 bytes from 192.168.1.15: icmp_seq=3 ttl=63 time=1.47 ms
^C
--- 192.168.1.15 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2511ms
rtt min/avg/max/mdev = 1.261/1.339/1.472/0.103 ms
```

Ngược lại, nếu *ping* từ một trạm ngoài Internet vào trạm orange hoặc green thì thấy thất bại:

```
[root@Cx ~]# ping 192.168.1.15
PING 192.168.1.15 (192.168.1.15) 56(84) bytes of data.
^C
--- 192.168.1.15 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2760ms
```

Bước 2: Kiểm tra kết nối đi ra Internet với cơ chế NAT

Mặc định IPFire cho phép các kết nối từ bên trong firewall (vùng green, orange và blue) ra Internet. Hơn nữa, cơ chế để một trạm nội bộ kết nối ra Internet là sử dụng NAT. Điều này có thể được kiểm tra như sau. Trên một trạm Internet (giả sử có địa chỉ 205.192.25.17), chạy *iperf3* chế độ server ở cổng TCP 5100:

```
[root@Cx ~]# iperf3 -s -p 5100
-----------------------------------------------------------
Server listening on 5100
-----------------------------------------------------------
```

Trên một trạm nội bộ thuộc vùng green (giả sử có địa chỉ 192.168.1.15), chạy *iperf3* chế độ client để kết nối đến server *iperf3* bên ngoài Internet. Kết quả kết nối thành công:

```
[root@C2 ~]# iperf3 -c 205.192.25.17 -p 5100
Connecting to host 205.192.25.17, port 5100
[  4] local 192.168.1.15 port 36854 connected to 205.192.25.17 port 5100
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec  20.9 MBytes   175 Mbits/sec    1   682 KBytes
[  4]   1.00-2.00   sec  27.1 MBytes   227 Mbits/sec   25   615 KBytes
[  4]   2.00-3.00   sec  30.0 MBytes   251 Mbits/sec    0   677 KBytes
[  4]   3.00-4.00   sec  28.3 MBytes   238 Mbits/sec    5   519 KBytes
[  4]   4.00-5.00   sec  27.0 MBytes   226 Mbits/sec    0   554 KBytes
[  4]   5.00-6.00   sec  24.7 MBytes   207 Mbits/sec    0   574 KBytes
[  4]   6.00-7.00   sec  24.6 MBytes   207 Mbits/sec    0   584 KBytes
```

```
[ 4]   7.00-8.00   sec  19.6 MBytes   164 Mbits/sec    0    588 KBytes
[ 4]   8.00-9.00   sec  23.7 MBytes   199 Mbits/sec    0    588 KBytes
[ 4]   9.00-10.00  sec  23.2 MBytes   194 Mbits/sec    0    590 KBytes
- - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[ 4]   0.00-10.00  sec   249 MBytes   209 Mbits/sec   31          sender
[ 4]   0.00-10.00  sec   247 MBytes   208 Mbits/sec               receiver
iperf Done.
```

Về phía server *iperf3*, kết quả kết nối từ client cũng được thể hiện:

```
[root@Cx ~]# iperf3 -s -p 5100
-----------------------------------------------------------
Server listening on 5100
-----------------------------------------------------------
Accepted connection from 203.162.5.11, port 36852
[  5] local 205.192.25.17 port 5100 connected to 203.162.5.11 port 36854
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-1.00   sec  19.0 MBytes   159 Mbits/sec
[  5]   1.00-2.00   sec  26.9 MBytes   225 Mbits/sec
[  5]   2.00-3.00   sec  29.4 MBytes   246 Mbits/sec
[  5]   3.00-4.00   sec  28.6 MBytes   240 Mbits/sec
[  5]   4.00-5.00   sec  27.4 MBytes   230 Mbits/sec
[  5]   5.00-6.00   sec  24.4 MBytes   204 Mbits/sec
[  5]   6.00-7.00   sec  24.4 MBytes   204 Mbits/sec
[  5]   7.00-8.00   sec  19.6 MBytes   164 Mbits/sec
[  5]   8.00-9.00   sec  23.7 MBytes   198 Mbits/sec
[  5]   9.00-10.00  sec  23.1 MBytes   194 Mbits/sec
[  5]  10.00-10.05  sec  1.19 MBytes   203 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-10.05  sec  0.00 Bytes  0.00 bits/sec                  sender
[  5]   0.00-10.05  sec   247 MBytes   207 Mbits/sec                receiver
-----------------------------------------------------------
Server listening on 5100
-----------------------------------------------------------
```

So sánh thông tin log của *iperf3* phía client và server thấy cơ chế NAT được thể hiện như sau:

- Client chạy trên máy 192.168.1.15, cổng TCP 36854 để kết nối đến server 205.192.25.17 tại cổng TCP 5100.
- Server nhận được kết nối của client từ địa chỉ 203.162.5.11. Như vậy, server nhận kết nối từ client theo địa chỉ IP mặt ngoài (kết nối red) của IPFire, không phải từ địa chỉ IP nội bộ của client (là 192.168.1.15).
- Thông tin các kết nối trên có thể được xem trong IPFire (menu Status/Connections). Nhìn vào danh sách các kết nối này, có thể thấy cơ chế NAT được thể hiện trong cột "Source IP: Port" theo dạng 192.168.1.15 > 203.162.5.11. Tương ứng với kết nối này, cột "Dest. IP; Port" cho thấy thông tin trên *iperf3* server (203.192.25.17: 5100).

Connections

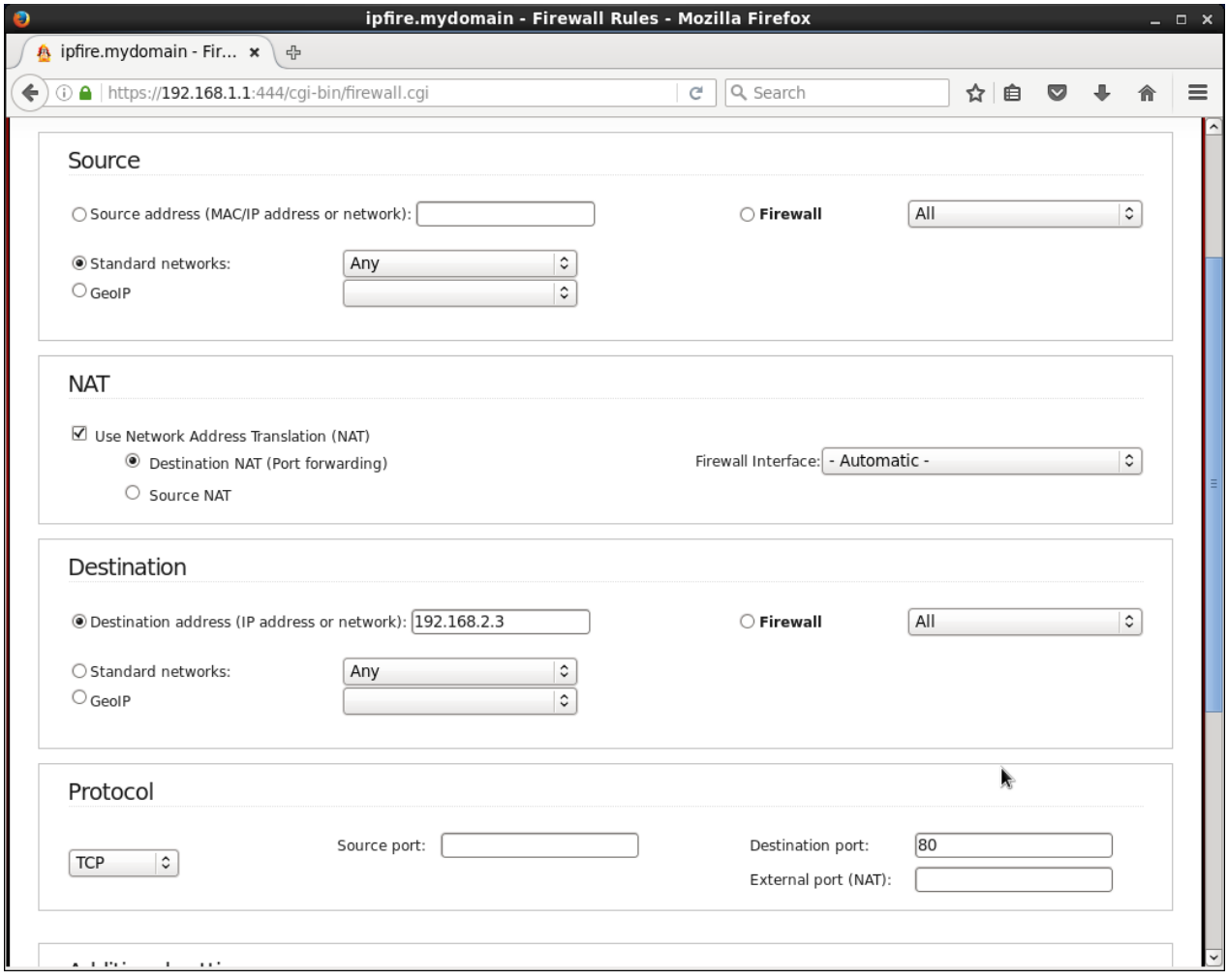
iptables Connection Tracking

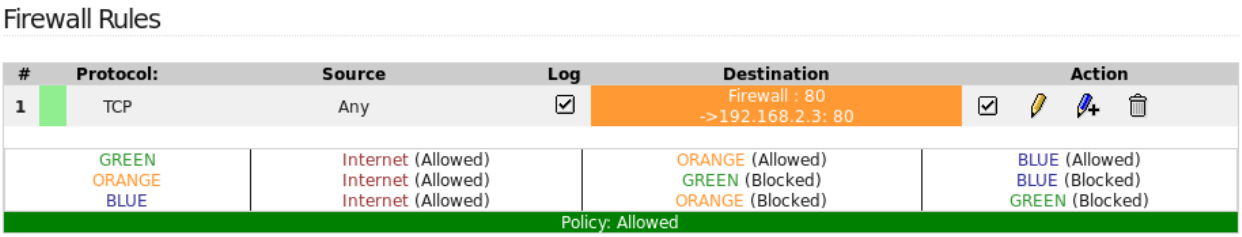
Legend : LAN | INTERNET | DMZ | Wireless | IPFire | VPN | OpenVPN | Multicast

Protocol:	Source IP: Port		Dest. IP: Port		download / Upload	Connection Status	Expires (Secs)
TCP	192.168.1.15 > 203.162.5.11	48860	205.192.25.17	22	23k / 27k	ESTABLISHED	119:59:59
TCP	192.168.1.15	50254	192.168.1.1	444	1k / 1k	ESTABLISHED	119:59:59
TCP	192.168.1.15	50252	192.168.1.1	444	1k / 1k	ESTABLISHED	119:59:59
TCP	192.168.1.15	50244	192.168.1.1	444	34k / 91k	ESTABLISHED	119:59:56
TCP	192.168.1.15 > 203.162.5.11	37720	203.162.5.1	22	1M / 74M	ESTABLISHED	119:58:35
TCP	192.168.1.15	48540	192.168.1.1	22	93k / 212k	ESTABLISHED	119:26:00
TCP	192.168.1.15	36698	192.168.2.3	22	46k / 61k	ESTABLISHED	118:50:52
TCP	192.168.1.15 > 203.162.5.11	36852	205.192.25.17	5100	1k / 891B	TIME_WAIT	0:01:59
TCP	192.168.1.15	50246	192.168.1.1	444	2k / 2k	TIME_WAIT	0:01:30
TCP	192.168.1.15 > 203.162.5.11	54984	184.28.218.96	80	360B / 0B	SYN_SENT	0:00:20
TCP	192.168.1.15 > 203.162.5.11	36854	205.192.25.17	5100	265M / 596k	CLOSE	0:00:09

Bước 3: Thiết lập cho phép kết nối từ Internet vào vùng DMZ

DMZ là vùng chứa các máy chủ dịch vụ như Web, FTP, Mail, v.v.. mà cho phép các trạm ngoài Internet kết nối đến. Giả sử máy chủ Web của công ty được cài đặt tại địa chỉ 192.168.2.3 (là một địa chỉ thuộc vùng orange), luật port forwarding được thiết lập như sau:





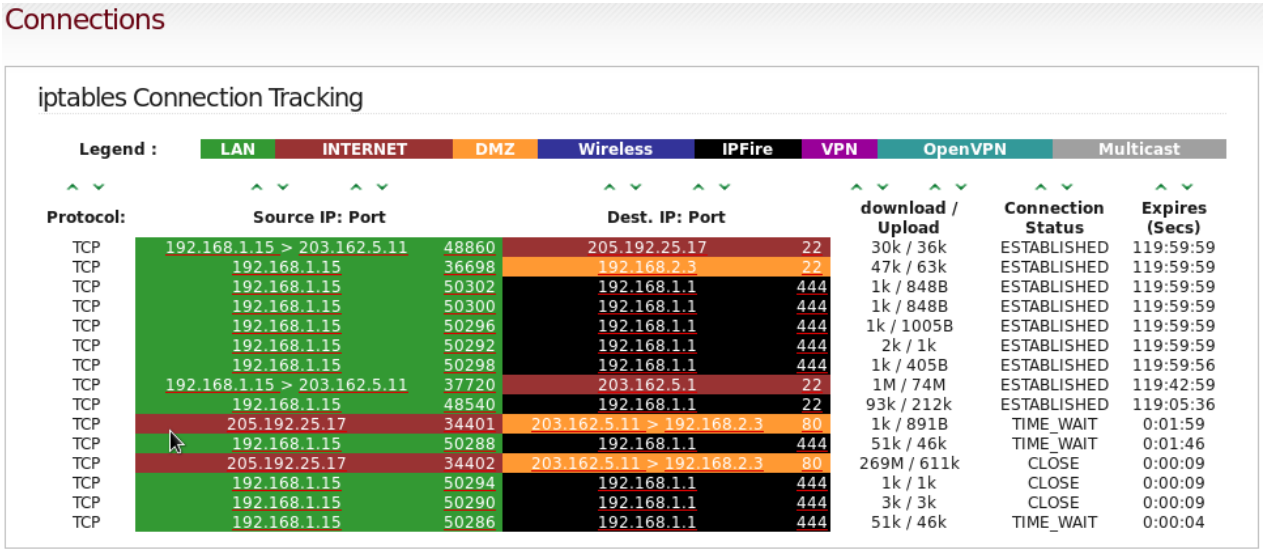
Sử dụng *iperf3* để giả lập Web server trên máy 192.168.2.3 và cũng cùng iperf3 giả lập web client trên máy ngoài Internet (địa chỉ 205.192.25.17). Kết quả là sau khi có luật port forwarding, máy ngoài Internet đã truy nhập được đến cổng 80 (web) của máy trong DMZ zone (orange) thông qua địa chỉ mặt ngoài của firewall.

```
[root@web ~]# iperf3 -s -p 80
-----------------------------------------------------------
Server listening on 80
-----------------------------------------------------------
Accepted connection from 205.192.25.17, port 34392
[  5] local 192.168.2.3 port 80 connected to 205.192.25.17 port 34393
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-1.00   sec  25.7 MBytes   215 Mbits/sec
[  5]   1.00-2.00   sec  30.1 MBytes   254 Mbits/sec
[  5]   2.00-3.00   sec  28.6 MBytes   240 Mbits/sec
[  5]   3.00-4.00   sec  26.8 MBytes   225 Mbits/sec
[  5]   4.00-5.00   sec  28.7 MBytes   241 Mbits/sec
[  5]   5.00-6.00   sec  28.1 MBytes   236 Mbits/sec
[  5]   6.00-7.00   sec  29.5 MBytes   248 Mbits/sec
[  5]   7.00-8.00   sec  29.9 MBytes   251 Mbits/sec
[  5]   8.00-9.00   sec  30.3 MBytes   254 Mbits/sec
[  5]   9.00-10.00  sec  30.1 MBytes   253 Mbits/sec
[  5]  10.00-10.05  sec  1.10 MBytes   200 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-10.05  sec  0.00 Bytes  0.00 bits/sec                  sender
[  5]   0.00-10.05  sec   289 MBytes   241 Mbits/sec                  receiver
[root@Cx ~]# iperf3 -c 203.162.5.11 -p 80
Connecting to host 203.162.5.11, port 80
[  4] local 205.192.25.17 port 34393 connected to 203.162.5.11 port 80
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec  29.4 MBytes   246 Mbits/sec   33    382 KBytes
[  4]   1.00-2.00   sec  28.8 MBytes   242 Mbits/sec    0    420 KBytes
[  4]   2.00-3.00   sec  28.4 MBytes   238 Mbits/sec    0    443 KBytes
[  4]   3.00-4.00   sec  27.0 MBytes   227 Mbits/sec    0    455 KBytes
[  4]   4.00-5.00   sec  29.3 MBytes   246 Mbits/sec    0    461 KBytes
[  4]   5.00-6.00   sec  27.8 MBytes   233 Mbits/sec    0    461 KBytes
[  4]   6.00-7.00   sec  29.8 MBytes   250 Mbits/sec   19    373 KBytes
[  4]   7.00-8.00   sec  29.7 MBytes   249 Mbits/sec    0    414 KBytes
[  4]   8.00-9.00   sec  30.5 MBytes   256 Mbits/sec    0    440 KBytes
[  4]   9.00-10.00  sec  29.6 MBytes   248 Mbits/sec    0    455 KBytes
- - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-10.00  sec   290 MBytes   243 Mbits/sec   52             sender
[  4]   0.00-10.00  sec   289 MBytes   243 Mbits/sec                  receiver
iperf Done.
```

Ngoài cổng 80 này, các truy nhập khác đều bị cấm:

```
[root@Cx ~]# iperf3 -c 203.162.5.11 -p 21
iperf3: error - unable to connect to server: No route to host
```

Có thể thấy cơ chế Port Forwarding được thể hiện trong các connections trên IPFire:



Connections

iptables Connection Tracking

Legend : LAN | INTERNET | DMZ | Wireless | IPFire | VPN | OpenVPN | Multicast

Protocol:	Source IP: Port		Dest. IP: Port		download / Upload	Connection Status	Expires (Secs)
TCP	192.168.1.15 > 203.162.5.11	48860	205.192.25.17	22	30k / 36k	ESTABLISHED	119:59:59
TCP	192.168.1.15	36698	192.168.2.3	22	47k / 63k	ESTABLISHED	119:59:59
TCP	192.168.1.15	50302	192.168.1.1	444	1k / 848B	ESTABLISHED	119:59:59
TCP	192.168.1.15	50300	192.168.1.1	444	1k / 848B	ESTABLISHED	119:59:59
TCP	192.168.1.15	50296	192.168.1.1	444	1k / 1005B	ESTABLISHED	119:59:59
TCP	192.168.1.15	50292	192.168.1.1	444	2k / 1k	ESTABLISHED	119:59:59
TCP	192.168.1.15	50298	192.168.1.1	444	1k / 405B	ESTABLISHED	119:59:56
TCP	192.168.1.15 > 203.162.5.11	37720	203.162.5.1	22	1M / 74M	ESTABLISHED	119:42:59
TCP	192.168.1.15	48540	192.168.1.1	22	93k / 212k	ESTABLISHED	119:05:36
TCP	205.192.25.17	34401	203.162.5.11 > 192.168.2.3	80	1k / 891B	TIME_WAIT	0:01:59
TCP	192.168.1.15	50288	192.168.1.1	444	51k / 46k	TIME_WAIT	0:01:46
TCP	205.192.25.17	34402	203.162.5.11 > 192.168.2.3	80	269M / 611k	CLOSE	0:00:09
TCP	192.168.1.15	50294	192.168.1.1	444	1k / 1k	CLOSE	0:00:09
TCP	192.168.1.15	50290	192.168.1.1	444	3k / 3k	CLOSE	0:00:09
TCP	192.168.1.15	50286	192.168.1.1	444	51k / 46k	TIME_WAIT	0:00:04

Bài 3: Xây dựng hệ thống IDS với Snort

Snort là một ứng dụng IDS mã nguồn mở, hoạt động trên nhiều hệ điều hành trong đó có Linux và Windows. Bài này thực hiện cài đặt và vận hành Snort như một IDS chế độ network-based và host-based. Các bước thực hiện như sau:

- Bước 1: Cài đặt Snort.
- Bước 2: Vận hành Snort chế độ IDS.
- Bước 3: Tạo luật đơn giản cảnh báo truy nhập từ bên ngoài.
- Bước 4: Cảnh báo truy nhập từ bên trong đến một nội dung nhạy cảm.
- Bước 5: Cảnh báo quét cổng (port scan) với preprocessor sfportscan.

Bước 1: Cài đặt Snort

Cài đặt các thư viện phần mềm cần thiết:

```
[root@C2 ~]# yum install -y wget gcc flex bison zlib zlib-devel libpcap libpcap-devel libdnet libdnet-devel
pcre pcre-devel tcpdump git libtool curl man
```

Thực hiện các bước download mã nguồn, dịch mã nguồn và cài đặt tiếp theo như hướng dẫn trên trang chủ Snort:

```
[root@C2 ~]# mkdir snort_src
[root@C2 ~]# cd snort_src
[root@C2 ~]# wget https://www.snort.org/downloads/snort/daq-2.0.6.tar.gz
```

```
--2016-10-19 02:11:30-- https://www.snort.org/downloads/snort/daq-2.0.6.tar.gz
...
[root@C2 ~]# wget https://www.snort.org/downloads/snort/snort-2.9.8.3.tar.gz
--2016-10-19 02:11:51-- https://www.snort.org/downloads/snort/snort-2.9.8.3.tar.gz
...
[root@C2 ~]# tar xvfz ./daq-2.0.6.tar.gz
...
[root@C2 ~]# cd daq-2.0.6
[root@C2 daq-2.0.6]# ./configure
...
[root@C2 daq-2.0.6]# make
...
[root@C2 daq-2.0.6]# make install
...
[root@C2 ~]# cd ..
[root@C2 ~]# tar xvfz ./snort-2.9.8.3.tar.gz
...
[root@C2 ~]# cd snort-2.9.8.3
[root@C2 daq-2.0.6]# ./configure
...
[root@C2 daq-2.0.6]# make
...
[root@C2 daq-2.0.6]# make install
...
```

Sau khi cài đặt thành công, Snort có thể hoạt động ở 3 chế độ:

- Package sniffer: hiển thị thông tin header các gói tin
- Package log: ghi lại các thông tin vào file log để xử lý sau này
- IDS: phân tích các gói tin hoặc các luồng TCP, thực hiện các chức năng IDS theo cơ chế signature-based.

Kiểm tra Snort hoạt động ở chế độ package sniffer với tham số *-dev*. Trong khi chạy Snort ở chế độ này, thực hiện *ping* đến Google, ta sẽ thấy Snort bắt được các gói tin ICMP Echo và Reply rồi hiển thị thông tin các gói tin này lên màn hình.

```
[root@C2 snort-2.9.8.3]# snort -dev
Running in packet dump mode

      --== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to passive.
Acquiring network traffic from "eth1".
Decoding Ethernet

      --== Initialization Complete ==--

  ,,_     -*> Snort! <*-
 o"  )~   Version 2.9.8.3 GRE (Build 383)
  ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
          Copyright (C) 2014-2015 Cisco and/or its affiliates. All rights reserved.
          Copyright (C) 1998-2013 Sourcefire, Inc., et al.
          Using libpcap version 1.4.0
          Using PCRE version: 7.8 2008-09-05
          Using ZLIB version: 1.2.3
```

```
Commencing packet processing (pid=20207)
WARNING: No preprocessors configured for policy 0.
10/19-02:35:02.723245 08:00:27:F7:C2:01 -> 52:54:00:12:35:02 type:0x800 len:0x62
10.0.2.15 -> 203.113.129.120 ICMP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:60238   Seq:14  ECHO
66 79 06 58 00 00 00 00 0E 09 0B 00 00 00 00 00  fy.X............
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F  ................
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F   !"#$%&'()*+,-./
30 31 32 33 34 35 36 37                          01234567

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
10/19-02:35:02.758950 52:54:00:12:35:02 -> 08:00:27:F7:C2:01 type:0x800 len:0x62
203.113.129.120 -> 10.0.2.15 ICMP TTL:56 TOS:0xA ID:59395 IpLen:20 DgmLen:84
Type:0  Code:0  ID:60238  Seq:14  ECHO REPLY
66 79 06 58 00 00 00 00 0E 09 0B 00 00 00 00 00  fy.X............
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F  ................
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F   !"#$%&'()*+,-./
30 31 32 33 34 35 36 37                          01234567

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Tạo thư mục log */var/log/snort* và chạy Snort vận hành ở chế độ package log:

```
[root@C2 snort-2.9.8.3]# mkdir /var/log/snort
[root@C2 snort-2.9.8.3]# snort -dev -l /var/log/snort
Running in packet logging mode

      --== Initializing Snort ==--
Initializing Output Plugins!
Log directory = /var/log/snort
pcap DAQ configured to passive.
Acquiring network traffic from "eth1".
Decoding Ethernet

      --== Initialization Complete ==--

  ,,_     -*> Snort! <*-
 o"  )~   Version 2.9.8.3 GRE (Build 383)
  ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
          Copyright (C) 2014-2015 Cisco and/or its affiliates. All rights reserved.
          Copyright (C) 1998-2013 Sourcefire, Inc., et al.
          Using libpcap version 1.4.0
          Using PCRE version: 7.8 2008-09-05
          Using ZLIB version: 1.2.3

Commencing packet processing (pid=20212)
^C*** Caught Int-Signal
WARNING: No preprocessors configured for policy 0.
===============================================================================
Run time for packet processing was 18.484920 seconds
Snort processed 15 packets.
Snort ran for 0 days 0 hours 0 minutes 18 seconds
   Pkts/sec:         0
===============================================================================
Memory usage summary:
```

```
  Total non-mmapped bytes (arena):       811008
  Bytes in mapped regions (hblkhd):      21590016
  Total allocated space (uordblks):      670688
  Total free space (fordblks):           140320
  Topmost releasable block (keepcost):   135008
================================================================================
Packet I/O Totals:
   Received:          15
   Analyzed:          15 (100.000%)
    Dropped:           0 (  0.000%)
   Filtered:          0 (  0.000%)
Outstanding:           0 (  0.000%)
   Injected:          0
================================================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:         15 (100.000%)
       VLAN:          0 (  0.000%)
        IP4:         13 ( 86.667%)
       Frag:          0 (  0.000%)
       ICMP:          7 ( 46.667%)
        UDP:          6 ( 40.000%)
        TCP:          0 (  0.000%)
        IP6:          0 (  0.000%)
    IP6 Ext:          0 (  0.000%)
   IP6 Opts:          0 (  0.000%)
      Frag6:          0 (  0.000%)
     ICMP6:           0 (  0.000%)
      UDP6:           0 (  0.000%)
      TCP6:           0 (  0.000%)
     Teredo:          0 (  0.000%)
    ICMP-IP:          0 (  0.000%)
    IP4/IP4:          0 (  0.000%)
    IP4/IP6:          0 (  0.000%)
    IP6/IP4:          0 (  0.000%)
    IP6/IP6:          0 (  0.000%)
        GRE:          0 (  0.000%)
    GRE Eth:          0 (  0.000%)
   GRE VLAN:          0 (  0.000%)
    GRE IP4:          0 (  0.000%)
    GRE IP6:          0 (  0.000%)
GRE IP6 Ext:          0 (  0.000%)
   GRE PPTP:          0 (  0.000%)
    GRE ARP:          0 (  0.000%)
    GRE IPX:          0 (  0.000%)
   GRE Loop:          0 (  0.000%)
       MPLS:          0 (  0.000%)
        ARP:          2 ( 13.333%)
        IPX:          0 (  0.000%)
   Eth Loop:          0 (  0.000%)
   Eth Disc:          0 (  0.000%)
   IP4 Disc:          0 (  0.000%)
   IP6 Disc:          0 (  0.000%)
   TCP Disc:          0 (  0.000%)
   UDP Disc:          0 (  0.000%)
  ICMP Disc:          0 (  0.000%)
All Discard:          0 (  0.000%)
      Other:          0 (  0.000%)
Bad Chk Sum:          4 ( 26.667%)
    Bad TTL:          0 (  0.000%)
     S5 G 1:          0 (  0.000%)
```

```
   S5 G 2:         0 (  0.000%)
    Total:        15
===============================================================================
Snort exiting
```

Trong khi chạy Snort chế độ package log, cũng thực hiện *ping* đến Google. Kết quả là trong thư mục */var/log/snort* xuất hiện file log của Snort ghi lại các gói tin đã bắt được:

```
[root@C2 snort-2.9.8.3]# ls /var/log/snort/
snort.log.1476819559
```

Lưu ý rằng file log được ghi lại ở dạng nhị phân, theo cấu trúc các gói tin. Có thể dụng lệnh *file* để kiểm tra cấu trúc của file nhị phân này:

```
[root@C2 snort-2.9.8.3]# file /var/log/snort/snort.log.1476819559
/var/log/snort/snort.log.1476819559: tcpdump capture file (little-endian) - version 2.4 (Ethernet, capture length
1514)
```

Để đọc file này, cần sử dụng phần mềm *tcpdump*. Có thể thấy kết quả hiển thị các gói tin ban đầu là dịch vụ ARP để xác định địa chỉ MAC từ địa chỉ IP khi trạm làm việc cần gửi gói tin ra Gateway. Tiếp theo là các gói tin dịch vụ DNS để xác định địa chỉ IP của Google. Cuối cùng là các gói tin ICMP Echo và Reply của lệnh *ping*:

```
[root@C2 snort-2.9.8.3]# tcpdump -r /var/log/snort/snort.log.1476819559
reading from file /var/log/snort/snort.log.1476819559, link-type EN10MB (Ethernet)
02:39:25.449298 IP 10.0.2.15.49648 > alu7750testscr.xyz1.gblx.mgmt.Level3.net.domain: 16399+ A?
www.google.com. (32)
02:39:30.448623 ARP, Request who-has 10.0.2.2 tell 10.0.2.15, length 28
02:39:30.449542 ARP, Reply 10.0.2.2 is-at 52:54:00:12:35:02 (oui Unknown), length 46
02:39:30.455255 IP 10.0.2.15.42932 > google-public-dns-a.google.com.domain: 16399+ A? www.google.com. (32)
02:39:30.547000 IP google-public-dns-a.google.com.domain > 10.0.2.15.42932: 16399 8/0/0 A 203.113.129.184, A
203.113.129.187, A 203.113.129.181, A 203.113.129.185, A 203.113.129.186, A 203.113.129.182, A
203.113.129.180, A 203.113.129.183 (160)
02:39:30.547521 IP 10.0.2.15 > 203.113.129.184: ICMP echo request, id 62798, seq 1, length 64
02:39:30.602657 IP 203.113.129.184 > 10.0.2.15: ICMP echo reply, id 62798, seq 1, length 64
02:39:30.603036 IP 10.0.2.15.38095 > alu7750testscr.xyz1.gblx.mgmt.Level3.net.domain: 45677+ PTR?
184.129.113.203.in-addr.arpa. (46)
02:39:35.608456 IP 10.0.2.15.57237 > google-public-dns-a.google.com.domain: 45677+ PTR? 184.129.113.203.in-
addr.arpa. (46)
02:39:35.685650 IP google-public-dns-a.google.com.domain > 10.0.2.15.57237: 45677 NXDomain 0/1/0 (116)
02:39:35.686025 IP 10.0.2.15 > 203.113.129.184: ICMP echo request, id 62798, seq 2, length 64
02:39:35.726270 IP 203.113.129.184 > 10.0.2.15: ICMP echo reply, id 62798, seq 2, length 64
02:39:36.687875 IP 10.0.2.15 > 203.113.129.184: ICMP echo request, id 62798, seq 3, length 64
02:39:36.725882 IP 203.113.129.184 > 10.0.2.15: ICMP echo reply, id 62798, seq 3, length 64
02:39:37.689496 IP 10.0.2.15 > 203.113.129.184: ICMP echo request, id 62798, seq 4, length 64
```

Bước 2: Vận hành Snort chế độ IDS

Trong chế độ này, Snort thực hiện kiểm soát các gói tin và thực hiện các phản ứng theo các luật được khai báo trước. Các file cấu hình của Snort được đặt trong thư

mục */etc/snort*. Cần copy tất cả các file cấu hình mặc định trong thư mục cài đặt (*snort-2.9.8.3/etc*) vào thư mục này:

```
[root@C2 ~]# cd ~/snort_src/snort-2.9.8.3
[root@C2 snort-2.9.8.3]# cp ./etc/* /etc/snort
[root@C2 snort]# ls -l /etc/snort/
total 328
-rw-r--r--. 1 root root   1281 Oct 19 10:53 attribute_table.dtd
-rw-r--r--. 1 root root   3757 Oct 19 10:53 classification.config
-rw-r--r--. 1 root root  23058 Oct 19 10:53 file_magic.conf
-rw-r--r--. 1 root root  31971 Oct 19 10:53 gen-msg.map
-rw-r--r--. 1 root root  13257 Oct 19 10:53 Makefile
-rw-r--r--. 1 root root    190 Oct 19 10:53 Makefile.am
-rw-r--r--. 1 root root  12306 Oct 19 10:53 Makefile.in
-rw-r--r--. 1 root root    687 Oct 19 10:53 reference.config
-rw-r--r--. 1 root root  26804 Oct 19 10:53 snort.conf
-rw-r--r--. 1 root root   2335 Oct 19 10:53 threshold.conf
-rw-r--r--. 1 root root 160606 Oct 19 10:53 unicode.map
```

File cấu hình để vận hành Snort chế độ IDS là */etc/snort/snort.conf*. Cần sửa đổi một số thông số như sau:

- HOME_NET: đây là mạng hoặc máy trạm mà Snort sẽ bảo vệ (tùy theo chế độ hoạt động là network-based hay host-based). Trong bước này, ta sử dụng Snort để bảo vệ một máy chủ Web thuộc vùng DMZ (xem hình vẽ trong bài thực hành số 1) có địa chỉ IP là 192.168.2.10.
- EXTERNAL_NET: là vùng bên ngoài, không cần được giám sát. Nó sẽ là tất cả các mạng & các máy trạm mà không thuộc HOME_NET.
- RULE_PATH: thư mục chứa các luật xử lý gói tin. Thiết lập giá trị để thư mục này là */etc/snort/rules*.
- Tạm thời ta chưa sử dụng các chức năng nhúng modul động (dynamic) trong Snort nên comment tham sô *dynamicdetection*.
- Tạm thời ta sẽ tự thiết lập các rule cho Snort mà không dùng các rule có sẵn. Vì vậy khai báo include file chứa rule (*/etc/snort/rules/myrules.rules*) và comment tất cả các lênh include các rule có sẵn.
- Khai báo thư mục chứa các danh sách white (các trạm an toàn) và black (các trạm nghi vấn) cũng là thư mục RULE_PATH.

Sau khi thiết lập các thông số, file cấu hình *snort.conf* có dạng sau:

```
[root@C2 ~]# cat /etc/snort/snort.conf
. . .
# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.2.10
# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET !$HOME_NET
. . .
# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
```

```
var RULE_PATH ./rules
. . .
# path to dynamic rules libraries
# dynamicdetection directory /usr/local/lib/snort_dynamicrules
. . .
# my rules
include $RULE_PATH/myrules.rules
# site specific rules
# include $RULE_PATH/local.rules
# include $RULE_PATH/app-detect.rules
. . .
var WHITE_LIST_PATH ./rules
var BLACK_LIST_PATH ./rules
```

Khi khởi động Snort, mặc định nó cần đọc các file *white_list.rules* và *black_list.rules* để xử lý các trạm thuộc danh sách white và black. Tạm thời ta chưa xử lý gì đặc biệt đối với các trạm white và black nên tạo 2 file rỗng tương ứng. Tương tự, file myrules.rules cũng chưa được định nghĩa luật nào:

```
[root@C2 rules]# touch /etc/snort/rules/black_list.rules
[root@C2 rules]# touch /etc/snort/rules/white_list.rules
[root@C2 rules]# touch /etc/snort/rules/myrules.rules
[root@C2 rules]# ls -l /etc/snort/rules/
total 4
-rw-r--r--. 1 root root  0 Oct 19 10:25 black_list.rules
-rw-r--r--. 1 root root  0 Oct 19 10:24 white_list.rules
-rw-r--r--. 1 root root  0 Oct 19 10:26 myrules.rules
```

Sau khi chuẩn bị các file cấu hình làm việc, chạy Snort chế độ IDS với yêu cầu bắt gói tin trên kết nối mạng *eth2*. Nếu thành công, Snort bắt đầu thực hiện bắt gói tin bằng dòng thông báo "Commencing packet processing (pid=23464)":

```
[root@C2 ~]# snort -i eth2 -l /var/log/snort -c /etc/snort/snort.conf
Running in IDS mode
      --== Initializing Snort ==--
. . .
Acquiring network traffic from "eth2".
Reload thread starting...
Reload thread started, thread 0x7f3ee96d0700 (23465)
Decoding Ethernet
      --== Initialization Complete ==--
  ,,_     -*> Snort! <*-
o"  )~   Version 2.9.8.3 GRE (Build 383)
''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2015 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.4.0
Using PCRE version: 7.8 2008-09-05
Using ZLIB version: 1.2.3
        Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.6  <Build 1>
Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
Preprocessor Object: SF_POP  Version 1.0  <Build 1>
Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
```

```
Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
Commencing packet processing (pid=23464)
```

Bước 3: Luật đơn giản cảnh báo truy nhập từ bên ngoài

Snort đã vận hành ở chế độ IDS thành công. Bước tiếp theo là khai báo các luật xử lý cho Snort. Ta bắt đầu với luật đơn giản nhất - phát hiện các truy cập từ bên ngoài với *ssh* hoặc *ping*.

a) Phát hiện truy cập ssh

Giả sử cần cảnh báo khi xuất hiện truy nhập *ssh* từ một máy ở xa vào HOME_NET. Luật được thiết lập trong file *myrules.rules* như sau:

```
[root@C2 rules]# cat /etc/snort/rules/myrules.rules
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"incoming SSH connection!"; flags:S; sid:10000;)
```

Cú pháp của luật này như sau:

- *alert:* thực hiện cảnh báo khi xuất hiện hoạt động khớp với khai báo của luật.
- *tcp:* luật được thiết lập dựa trên thông số của gói tin TCP. Có thể thay thông số này bằng các loại gói tin khác như UDP, IP, ICMP, v.v..
- *$EXTERNAL_NET any -> $HOME_NET 22:* điều kiện kiểm tra là gói tin đến từ bất cứ trạm nào trong EXTERNAL_NET và từ bất cứ cổng nào, gửi đến HOME_NET cổng 22 (là cổng mà dịch vụ SSH đang hoạt động).
- msg:"incoming SSH connection!": hiển thị thông báo cảnh báo.
- *flags:S*: điều kiện hạn chế lọc gói tin. Khi thực hiện một kết nối TCP đến cổng 22, có rất nhiều gói tin được gửi đến. Điều kiện lọc dựa trên nguồn và đích (địa chỉ & cổng) sẽ tạo ra nhiều cảnh báo cho cùng một hành động *ssh*. Thủ thuật ở đây là dựa vào thông điệp SYN. Ta biết rằng các kết nối TCP luôn phải bắt đầu bằng quá trình bắt tay 3 bước với các thông điệp SYN, ACK SYN, ACK. Vậy nên nếu lọc bổ sung thêm các thông điệp này (*flags:S* tương ứng với thông điệp SYN) sẽ chỉ tạo ra 1 cảnh bảo cho 1 hành động SSH.
- *sid*:*10000*: mã số để khớp giữa cảnh báo với luật. Ví dụ khi cần liệt kê các cảnh báo theo từng luật thì có thể căn cứ vào sid của luật để lọc các cảnh báo.

Trong khi Snort đang vận hành ở chế độ IDS, từ một trạm nào đó thực hiện kết nối *ssh* vào máy chủ 192.168.2.10, một thông điệp cảnh báo sẽ được gửi đến file log:

```
[root@C2 rules]# tail -f /var/log/snort/alert
```

```
 [**] [1:10000:0] incoming SSH connection! [**]
[Priority: 0]
10/19-11:41:35.096890 192.168.2.25:38559 -> 192.168.2.10:22
TCP TTL:64 TOS:0x0 ID:49678 IpLen:20 DgmLen:60 DF
******S* Seq: 0x3467110D  Ack: 0x0  Win: 0x3908  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 3724035 0 NOP WS: 6
```

b) Phát hiện kết nối ping

Có thể bổ sung luật mới để phát hiện ai đó đang *ping* vào hệ thống:

```
[root@C2 rules]# cat /etc/snort/rules/myrules.rules
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"incoming SSH connection!"; flags:S; sid:10000;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"soomeone ping me!"; sid:10001;)
```

Khởi động lại Snort và dùng một trạm khác *ping* đến địa chỉ 192.168.2.10. Kết quả cảnh báo như sau:

```
[root@C2 rules]# tail -f /var/log/snort/alert
[**] [1:10001:0] soomeone ping me! [**]
[Priority: 0]
10/19-11:57:21.361893 192.168.2.25 -> 192.168.2.10
ICMP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:54020   Seq:1  ECHO
[**] [1:10001:0] soomeone ping me! [**]
[Priority: 0]
10/19-11:57:22.364432 192.168.2.25 -> 192.168.2.10
ICMP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:54020   Seq:2  ECHO
```

Bước 4: Cảnh báo truy nhập từ bên trong đến một nội dung nhạy cảm

Cần kiểm soát các trạm thuộc HOME_NET và phát hiện trạm nào truy nhập đến một nội dung nhạy cảm không được phép. Lấy ví dụ giả định nội dung nhạy cảm là "terrorism" (khủng bố). Có thể thiết lập luật kiểm tra các truy nhập từ bên trong đến nội dung này và cảnh báo như sau:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg:"terrorism contact!"; content:"terrorism"; nocase;
sid:10003;)
```

Khởi động lại Snort chế độ IDS và kiểm tra file cảnh báo */var/log/snort/alert*. Tiếp theo, dung web browser tìm kiếm một trang web nào đó có tên chứa từ khóa *terrorism*, giả sử là *http://www.merriam-webster.com/dictionary/terrorism*. Click vào link để truy nhập đến trang web này. Cảnh báo sẽ được phát ra:

```
[root@C2 ~]# tail -f /var/log/snort/alert
[**] [1:10003:0] terrorism contact! [**]
[Priority: 0]
10/21-12:47:40.222111 10.0.2.15:45708 -> 118.69.16.14:80
TCP TTL:64 TOS:0x0 ID:21096 IpLen:20 DgmLen:1294 DF
***AP*** Seq: 0x81B33DB6  Ack: 0x2AB3C524  Win: 0x9C4E  TcpLen: 20
```

Trường hợp luật cảnh báo không phát hiện được khi truy nhập đến trang web, một trong những lý do là Snort không bắt được gói tin gửi đi trong khi tực tế trang web vẫn được truy nhập và hiển thị trong browser. Nguyên nhân phổ biến là vấn đề offload checksum. Trước khi gửi gói tin, bên truyền cần tính toán checksum và đưa vào trường checksum của gói tin. Nếu card mạng có chức năng tính toán checksum, phần mềm ứng dụng sẽ không tính checksum nữa mà chuyển gói tin có trường checksum bằng 0 xuống cho card mạng xử lý. Thư viện *libpcap* (thư viện phần mềm bắt gói tin trên Linux) hoạt động ở giữa phần ứng dụng và card mạng do vậy khi tiếp nhận gói tin sẽ thấy checksum không hợp lệ và không xử lý gói tin này. Dẫn đến luật alert không được thực thi. Có thể chạy Snort với thám số -k none để thông báo Snort không kiểm tra trường checksum:

```
[root@C2 ~]# snort -d -c /etc/snort/snort.conf -i eth1 -k none
. . .
```

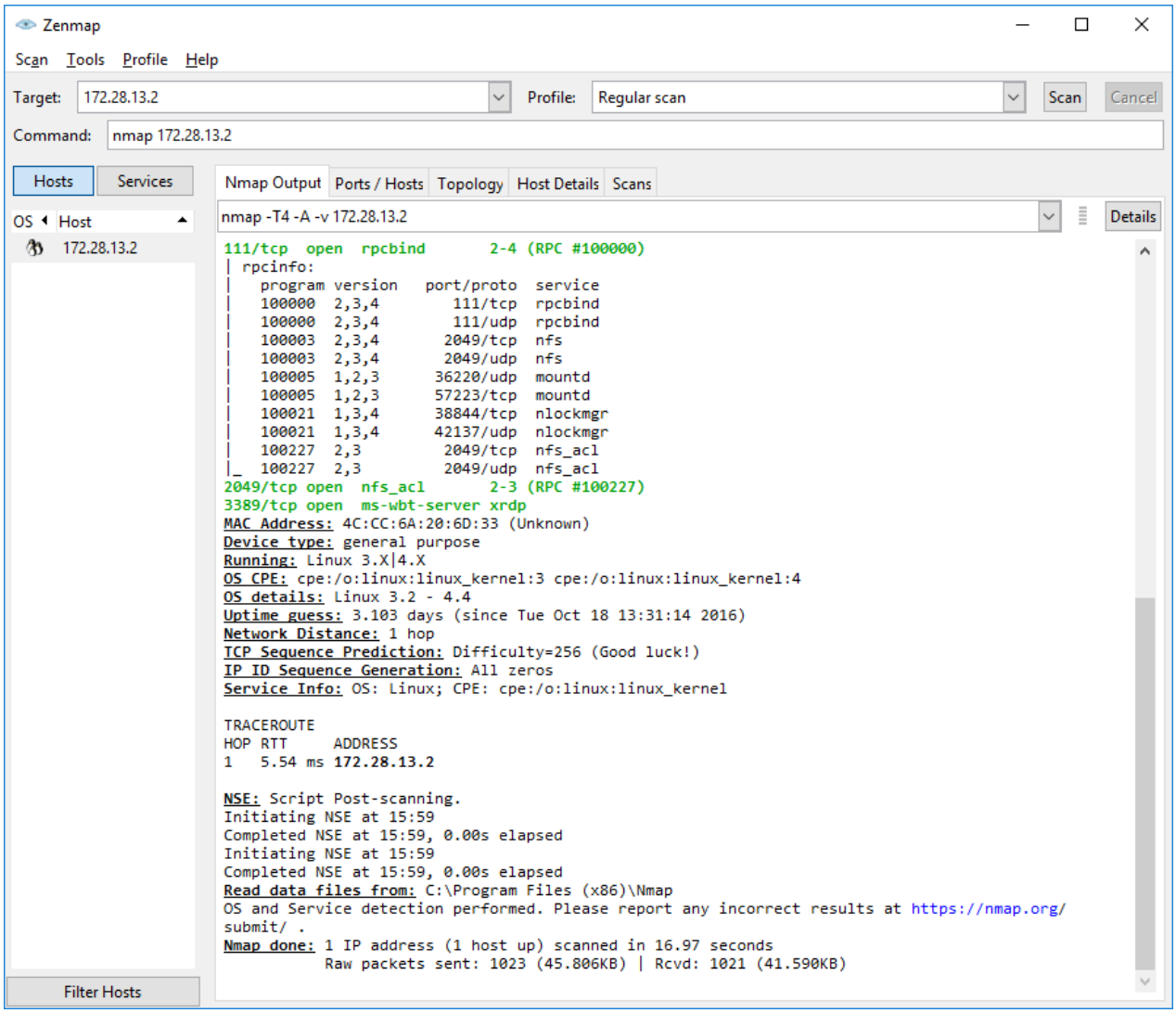
Bước 5: Cảnh báo quét cổng (port scan) với preprocessor sfportscan

Quét cổng thường là bước chuẩn bị cho các cuộc tấn công mạng. Nó thực hiện do thám hệ thống của nạn nhân để xác định các dịch vụ nào đang hoạt động và các thông tin liên quan đến các dịch vụ này. Có nhiều công cụ để thực hiện quét cổng trong đó *nmap* là công cụ phổ biến nhất, hỗ trợ nhiều hệ điều hành và có cả phiên bản hỗ trợ giao diện đồ họa. Sử dụng *yum* để cài đặt *nmap* và thực hiện quét cổng đến máy nạn nhân (địa chỉ 192.168.2.10):

```
[root@C1 ~]# yum install nmap
. . .
Package 2:nmap-5.51-4.el6.x86_64 already installed and latest version
Nothing to do
[root@C1 ~]# nmap -v -A 192.168.2.10
Starting Nmap 5.51 ( http://nmap.org ) at 2016-10-21 04:57 EDT
NSE: Loaded 57 scripts for scanning.
Initiating ARP Ping Scan at 04:57
Scanning 192.168.2.10 [1 port]
Completed ARP Ping Scan at 04:57, 0.00s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 04:57
Completed Parallel DNS resolution of 1 host. at 04:57, 0.03s elapsed
Initiating SYN Stealth Scan at 04:57
Scanning 192.168.2.10 [1000 ports]
Discovered open port 22/tcp on 192.168.2.10
Discovered open port 6000/tcp on 192.168.2.10
Completed SYN Stealth Scan at 04:57, 0.15s elapsed (1000 total ports)
Initiating Service scan at 04:57
Scanning 2 services on 192.168.2.10
Completed Service scan at 04:57, 6.01s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against 192.168.2.10
Retrying OS detection (try #2) against 192.168.2.10
Retrying OS detection (try #3) against 192.168.2.10
Retrying OS detection (try #4) against 192.168.2.10
Retrying OS detection (try #5) against 192.168.2.10
NSE: Script scanning 192.168.2.10.
Initiating NSE at 04:57
Completed NSE at 04:57, 0.13s elapsed
Nmap scan report for 192.168.2.10
```

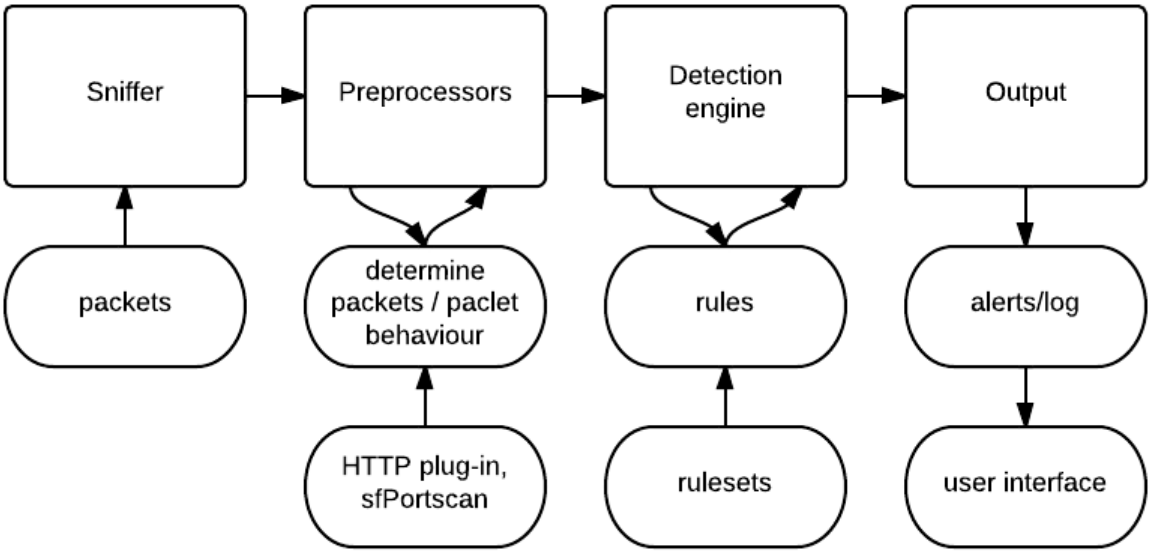
```
Host is up (0.00047s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 5.3 (protocol 2.0)
| ssh-hostkey: 1024 24:59:54:fe:47:1e:63:ce:61:b3:5b:f6:b5:83:58:ec (DSA)
|_2048 42:95:fe:74:4c:17:65:58:72:75:2c:8d:34:8e:c7:7b (RSA)
6000/tcp open  X11     (access denied)
MAC Address: 08:00:27:1B:C2:02 (Cadmus Computer Systems)
No exact OS matches for host (If you know what OS is running on it, see http://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=5.51%D=10/21%OT=22%CT=1%CU=35217%PV=Y%DS=1%DC=D%G=Y%M=080027%TM=5
OS:809D884%P=x86_64-redhat-linux-gnu)SEQ(SP=101%GCD=1%ISR=107%TI=Z%CI=Z%II=
OS:I%TS=A)OPS(O1=M5B4ST11NW7%O2=M5B4ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7%
OS:O5=M5B4ST11NW7%O6=M5B4ST11)WIN(W1=3890%W2=3890%W3=3890%W4=3890%W5=3890%W
OS:6=3890)ECN(R=Y%DF=Y%T=40%W=3908%O=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=
OS:O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD
OS:=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0
OS:%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1
OS:(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI
OS:=N%T=40%CD=S)
Uptime guess: 0.171 days (since Fri Oct 21 00:51:40 2016)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=257 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Unix
TRACEROUTE
HOP RTT     ADDRESS
1   0.47 ms 192.168.2.10
Read data files from: /usr/share/nmap
OS and Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.26 seconds
Raw packets sent: 1111 (52.918KB) | Rcvd: 1079 (46.702KB)
```

Trên Windows, phiên bản đồ họa của *nmap* có tên *zenmap* cho phép quét cổng của máy nạn nhân và hiển thị tất cả các thông số liên quan đến các dịch vụ đang chạy trên máy nạn nhân.



Nguyên lý hoạt động của các công cụ quét cổng là lần lượt gửi gói tin đặc biệt (chẳng hạn TCP SYN) đến tất cả các cổng của máy nạn nhân và đợi gói tin trả về. Một dịch vụ đang hoạt động tại cổng tương ứng không phân biệt được đây là một kết nối từ trạm sử dụng thông thường hay là một hành động trong cuộc tấn công quét cổng, do vậy sẽ trả lời bằng một gói tin phù hợp (ví dụ, với TCP SYN sẽ là SYN ACK). Dựa vào kết quả trả về này, công cụ quét cổng xác định được dịch vụ đang "nghe" trên máy nạn nhân và tiếp tục gửi các gói tin phù hợp khác để do thám thêm thông tin về dịch vụ tương ứng. Để phát hiện và cảnh báo một cuộc tấn công quét cổng như vậy, các luật đơn giản kiểm tra thông tin của từng gói tin riêng rẽ sẽ không xử lý được. Snort cung cấp cơ chế preprocessor và các luật prepresessor để phát hiện các dạng tấn công "tổ hợp" như vậy. Một cách khái quát, preprocessor là cách thức Snort chấp nhận thêm các modul (plugin) tiền xử lý gói tin. Các modul này được người dùng cài đặt thêm (bằng cách sử dụng thư viện hàm Snort API) và gắn vào Snort để nhận được các gói tin và xử lý chúng trước khi gói tin được chuyển cho máy xử lý trung tâm của Snort. Tùy vào từng mục đích riêng mà mỗi modul có thể kiểm tra các thông tin trong gói tin, thay đổi các thông tin này, rồi chuyển cho Snort xử lý. Một ví dụ là modul xử lý các gói tin tấn công dịch vụ web. Giả sử vùng chứa các script web admin (cần được bảo về) được đặt tại URL http://example.com/admin và Snort được thiết lập cảnh báo hoặc cấm truy nhập từ

bên ngoài đến URL này. Tuy nhiên, một số biến thể của URL sẽ cho kết quả tương tự, ví dụ như http://example.com/./admin. Modul chuẩn hóa các gói tin HTTP cài đặt dạng preprocessor sẽ kiểm tra và chuyển tất cả các biến thể URL này trở thành URL đúng và chuyển cho Snort xử lý. Sơ đồ bên dưới mô tả vai trò và vị trí của các preprocessor trong kiến trúc Snort.



Với tấn công quét cổng, preprocessor *sfportscan* đã được phát triển để xử lý. Nó thu thập nhiều gói tin và phân tích sự liên quan giữa các gói tin này để quyết định chúng có phải là chuỗi các gói tin quét cổng hay không. Để kích hoạt preprocessor này, cần thiết lập tham số cấu hình *sfportscan* và include tập luật *preprocessor.rules* trong /etc/snort/snort.conf:

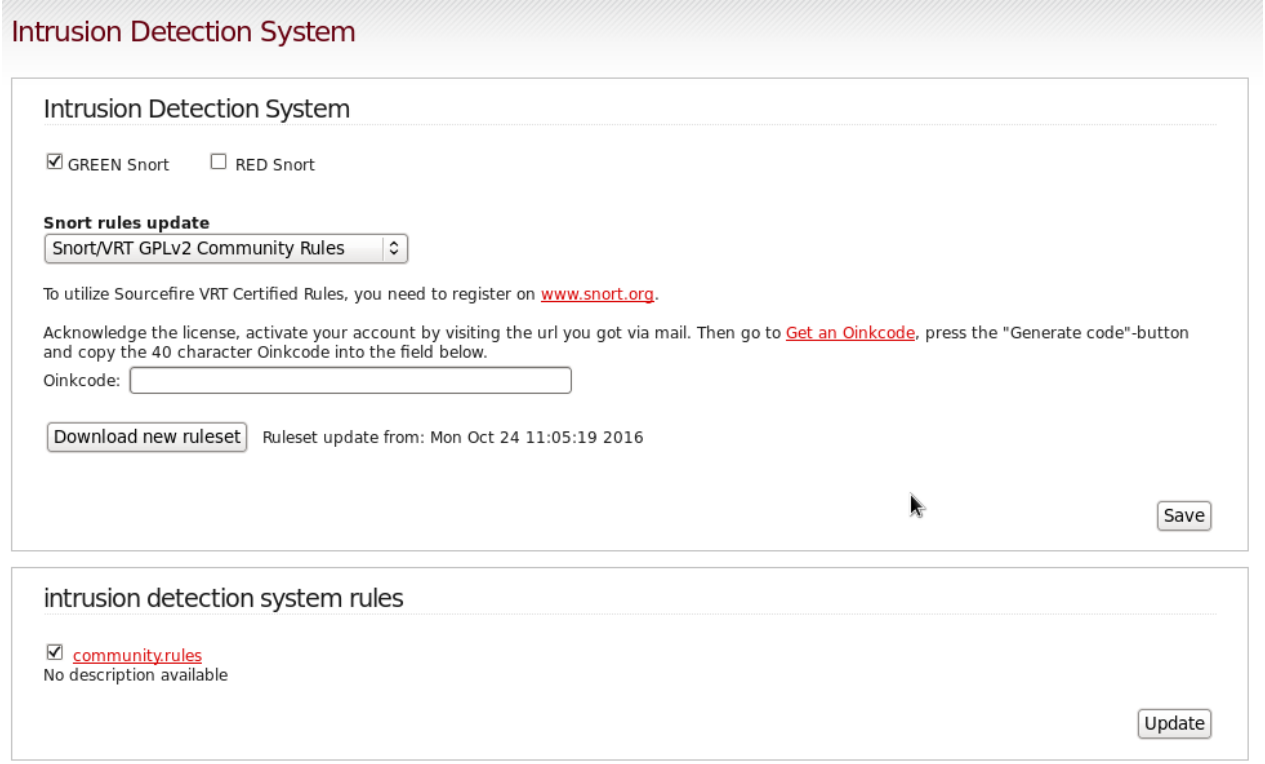
```
[root@C2 tmp]# cat /etc/snort/snort.conf | grep sfportscan
# Portscan detection.  For more information, see README.sfportscan
preprocessor sfportscan: proto  { all } scan_type { all } sense_level { low }
[root@C2 tmp]# cat /etc/snort/snort.conf | grep preprocessor.rules
include $PREPROC_RULE_PATH/preprocessor.rules
```

Khởi động lại Snort và thực hiện quét cổng từ một máy khác bằng *nmap* như mô tả bên trên. Cuộc tấn công quét cổng sẽ được Snort phát hiển và cảnh báo:

```
[root@C2 ~]# tail -f /var/log/snort/alert
[**] [122:1:1] (portscan) TCP Portscan [**]
[Classification: Attempted Information Leak] [Priority: 2]
10/21-16:22:17.259813 192.168.2.25 -> 192.168.2.10
PROTO:255 TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:159 DF
```

Bài 4: Intrusion Prevention System (IPS) = IDS + Firewall

IPFire được tích hợp sẵn với Snort để bổ sung chức năng IDS cho tường lửa. Để truy nhập đến chức năng cấu hình cho IDS này, sử dụng menu *Services/Instrusion Detection*. Trang cấu hình này cho phép thiết lập một số thông tin cơ bản của Snort như là kết nối mạng mà Snort sẽ giám sát (có thể chọn 1 hoặc nhiều kết nối), hoặc cho phép download các tập luật cảnh báo mới nhất (*Snort rules update*) để cập nhật vào thành phần IDS của Firewall.



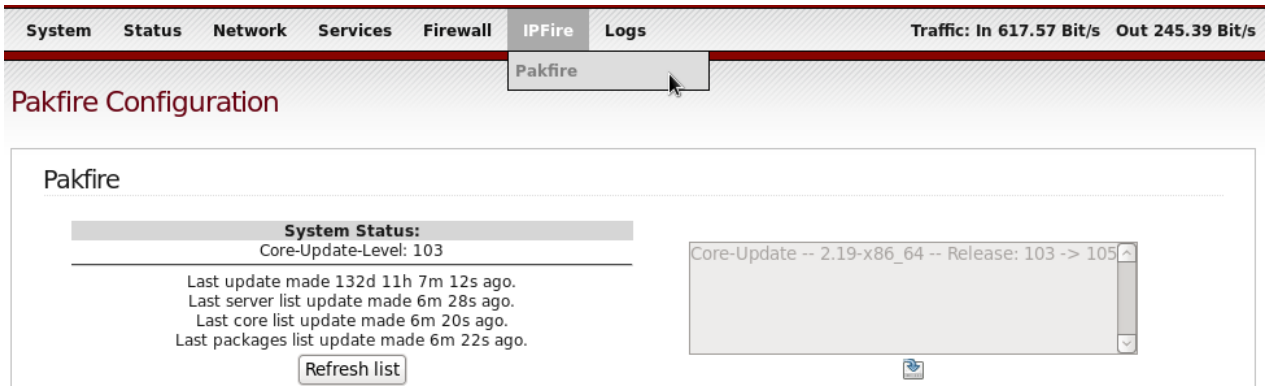
Bên cạnh chức năng Snort IDS của IPFire, Guardian là một thành phần Addon của IPFire, cho phép tự động ngăn chặn các cuộc tấn công mạng ngay khi Snort phát hiện ra. Như vậy, Snort kết hợp với Guardian sẽ biến IPFire trở thành một hệ thống chống xâm nhật (IPS). Tư tưởng của Guardian là đặt ngưỡng cảnh báo số lần Snort phát hiện tấn công (*Strike Threshold*, mặc định là 3) từ một địa chỉ IP nào đó và khi một trạm tấn công đạt đến ngưỡng này, IP của trạm tấn công sẽ bị IPFire tự động đưa vào danh sách khóa (block) trong một khoảng thời gian (mặc định là 86400 giây, tức là 24h). Các thông số này có thể được thiết lập lại trong trang cấu hình Guardian. Các bước thực hiện như sau:

- Bước 1: Cập nhật phiên bản mới cho IPFire và cài đặt Addon Guardian.
- Bước 2: Lựa chọn luật (rule) và kích hoạt IDS.
- Bước 3: Thiết lập ngưỡng cảnh báo và kích hoạt IPS.

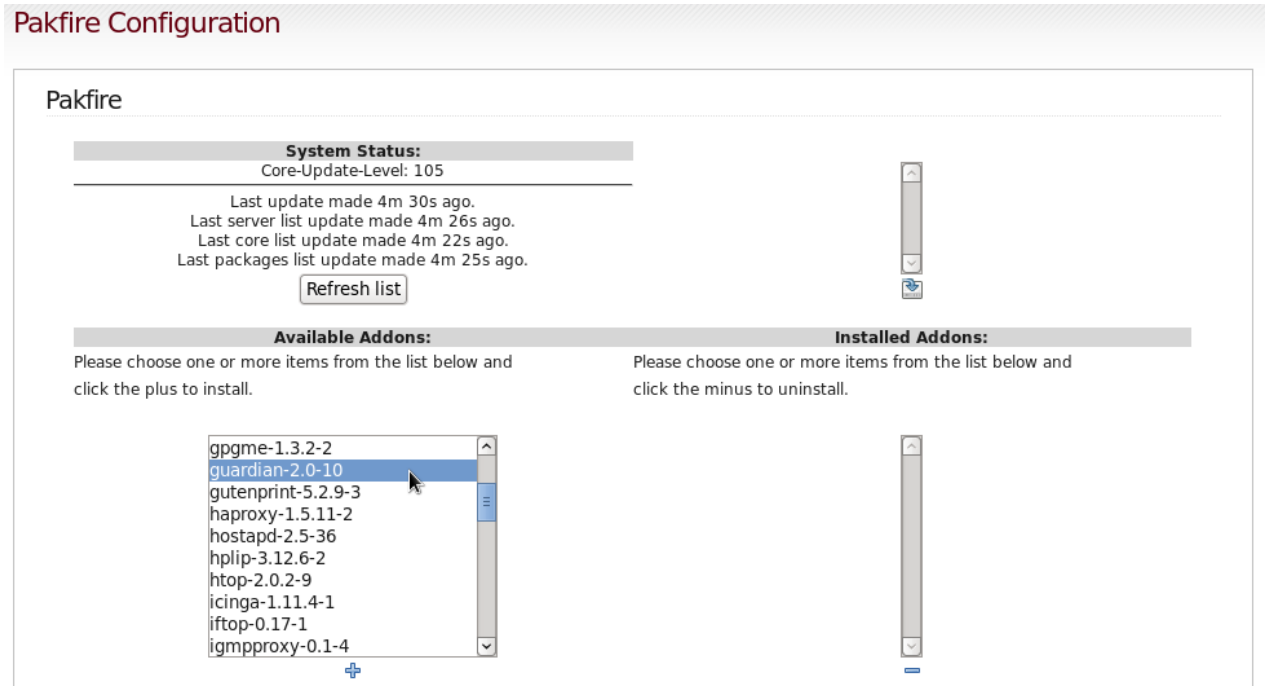
Bước 1: Cập nhật phiên bản mới cho IPFire và cài đặt Addon Guardian

Một số phiên bản IPFire cũ gặp vấn đề hiển thị khi cài đặt thêm thành phần Addon Guardian. Cần thực hiện cập nhật phiên bản mới nhất cho IPFire bằng công cụ Pakfire

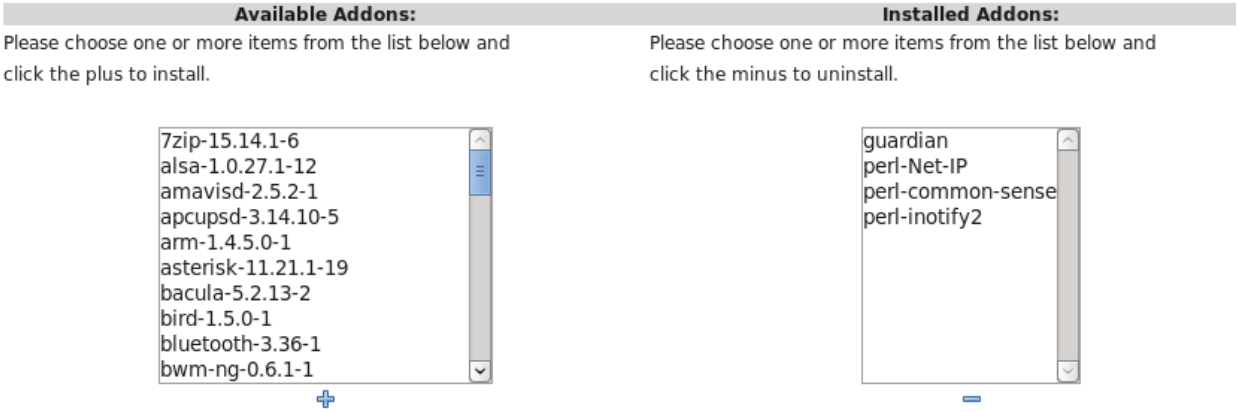
(menu *IPFire/Pakfire*). Kiểm tra kết nối Internet thông suốt, hệ thống sẽ thông báo có phiên bản mới hơn so với phiên bản đang dùng hay không (trong ví dụ bên dưới, phiên bản IPFire đang dùng là *Core-Update-Level:103*, phiên bản mới nhất là *Core-Update-Level:105*). Thực hiện cập nhật và khởi động lại IPFire.



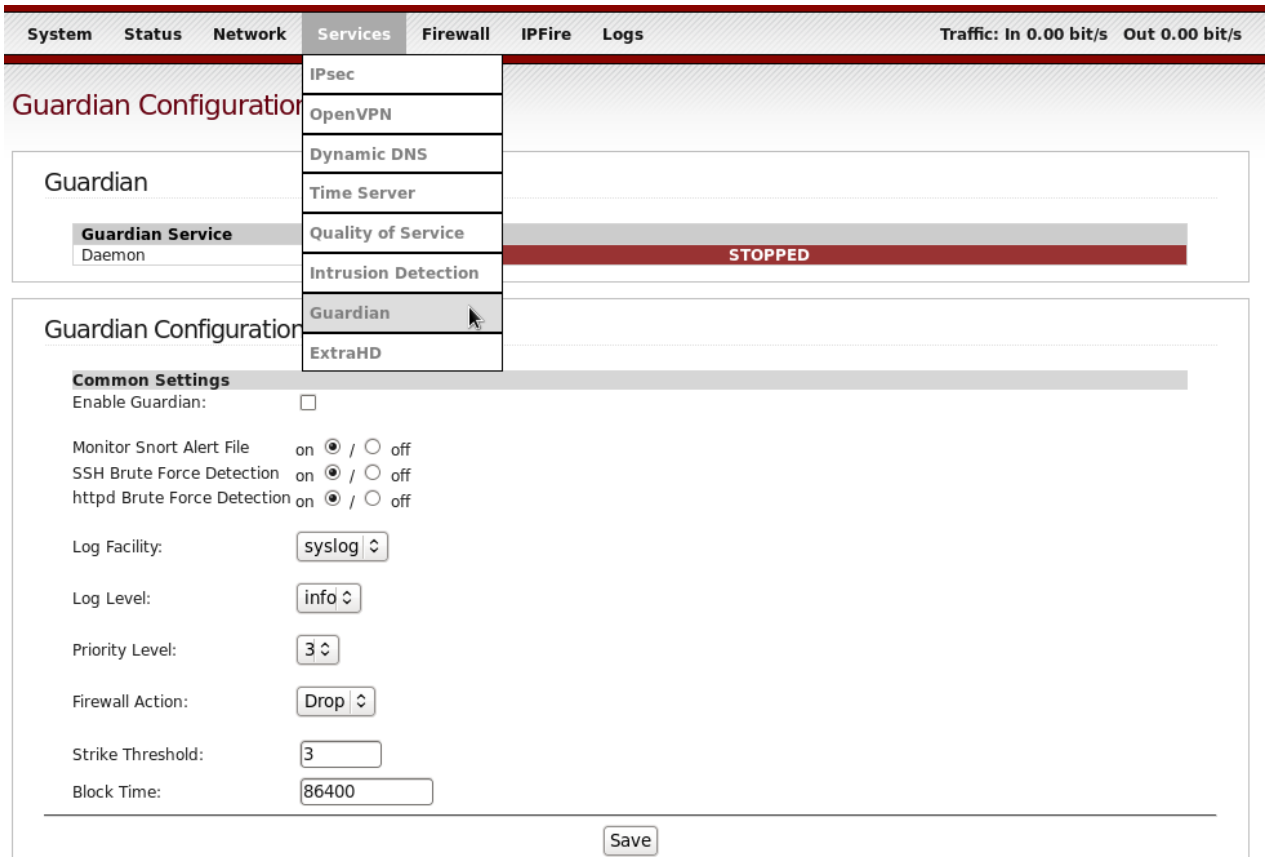
Sau khi cập nhật và khởi động lại IPFire, kiểm tra phiên bản đã cập nhật là *Core-Update-Level:105* và không có phiên bản nào mới hơn. Bước tiếp theo là cài đặt thành phần Addon Guardian vào IPFire với công cụ Pakfire. Trong ví dụ bên dưới, phiên bản Guardian được tìm thấy và có thể cài đặt là *guardian-2.0-10*.



Khi chọn cài đặt thành phần Addon Guardian, một số thư viện cần thiết cũng được tự động cài đặt cùng, bao gồm *perl-Net-IP*, *perl-common-sense* và *perl-inotify2*.



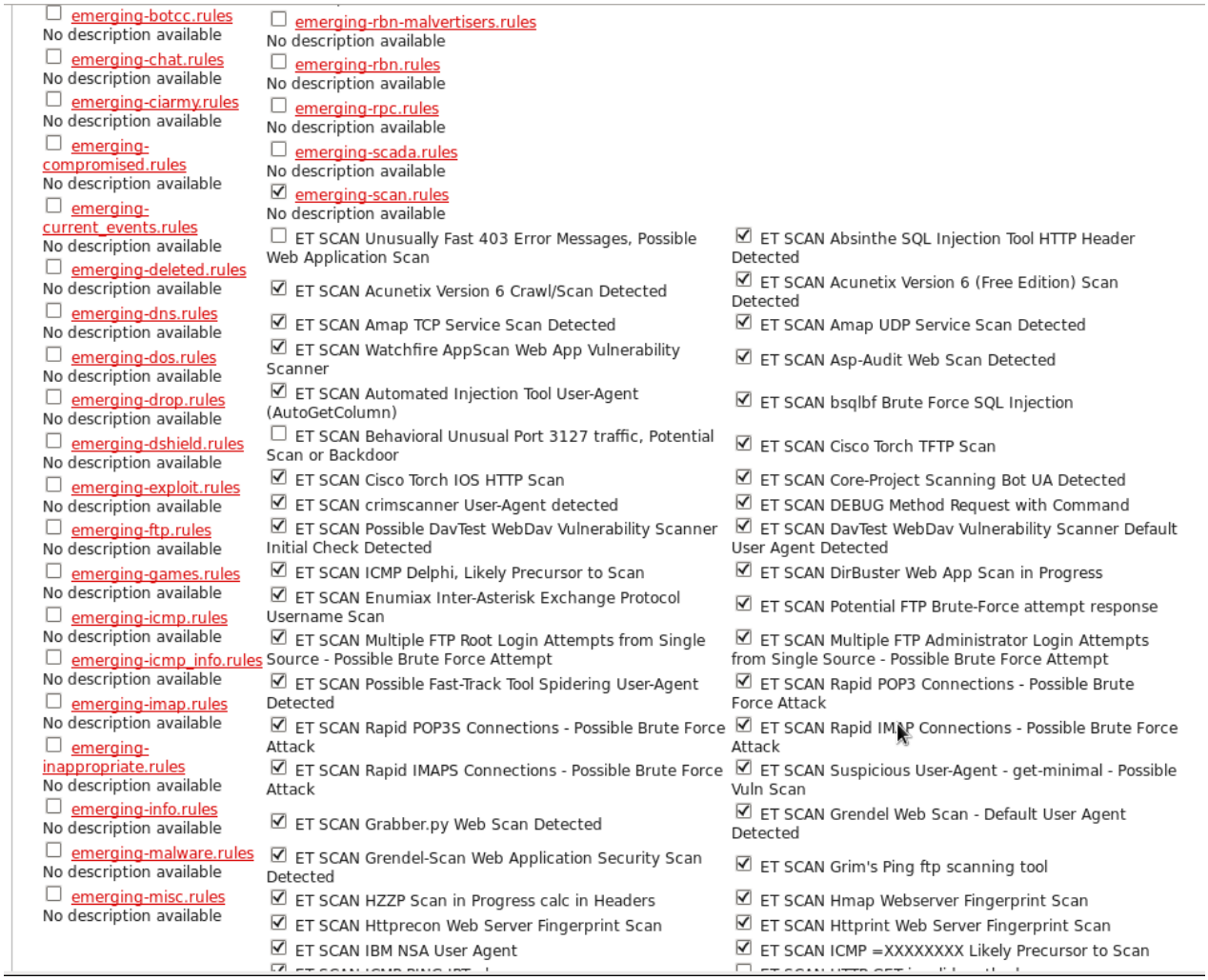
Sau khi thành phần Addon Guardian được cài đặt vào IPFire, menu Services xuất hiện thêm thành phần Guardian để cho phép thiết lập các thông số cấu hình:



Bước 2: Lựa chọn luật (rule) và kích hoạt IDS

Như đã môt tả bên trên, có thể vào menu *Services/Instrusion Detection* để tải các luật giúp Snort phát hiện tấn công. Giả sử ta cần một hệ thống chống tấn công quét cổng, tập luật Snort hỗ trợ phát hiện tấn công quét cổng có tên *emerging-scan.rules* nằm trong bộ luật *Snort/VRT GPLv2 Community Rules*. Vào trang *Services/Instrusion Detection*, chọn *Snort rules update* là *Snort/VRT GPLv2 Community Rules* và click "Download new ruleset" để tải bộ luật này về. Sau khi tải xong, danh sách các luật được hiển thị trong phần "instrusion detection system rules", tìm đến tập luật *emerging-scan.rules* và click

vào luật này để xem chi tiết các luật. Click chọn áp dụng tập luật *emerging-scan.rules* và click "Update" để lưu lại cấu hình cho Snort.



Sau khi tập luật *emerging-scan.rules* đã được lựa chọn kích hoạt, sử dụng nmap trên một trạm nào đó và thực hiện tấn công dò xét các cổng của máy IPFire:

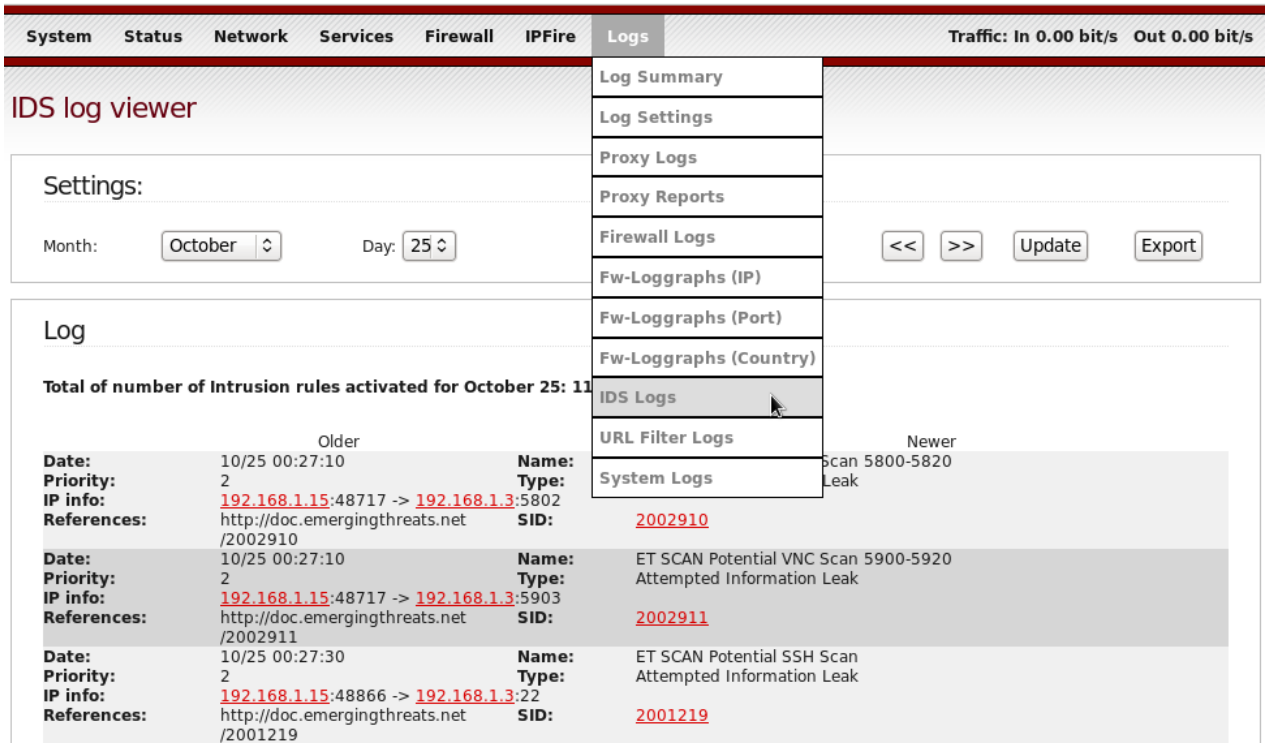
```
[root@C2 ~]# nmap -v -A 192.168.1.3
Starting Nmap 5.51 ( http://nmap.org ) at 2016-10-25 00:07 ICT
NSE: Loaded 57 scripts for scanning.
Initiating ARP Ping Scan at 00:07
Scanning 192.168.1.3 [1 port]
. . .
```

Các hành động tấn công quét cổng này sẽ bị Snort phát hiện và hiển thị cảnh báo trong file *alert*:

```
[root@ipfire ~]# tail -f /var/log/snort/alert
[**] [1:2002910:5] ET SCAN Potential VNC Scan 5800-5820 [**]
[Classification: Attempted Information Leak] [Priority: 2]
10/25-00:27:10.707470 192.168.1.15:48717 -> 192.168.1.3:5802
TCP TTL:46 TOS:0x0 ID:17553 IpLen:20 DgmLen:44
******S* Seq: 0xE9A32109  Ack: 0x0  Win: 0xC00  TcpLen: 24
```

```
TCP Options (1) => MSS: 1460
[Xref => http://doc.emergingthreats.net/2002910]
[**] [1:2002911:5] ET SCAN Potential VNC Scan 5900-5920 [**]
[Classification: Attempted Information Leak] [Priority: 2]
10/25-00:27:10.859326 192.168.1.15:48717 -> 192.168.1.3:5903
TCP TTL:55 TOS:0x0 ID:21634 IpLen:20 DgmLen:44
******S* Seq: 0xE9A32109  Ack: 0x0  Win: 0x1000  TcpLen: 24
TCP Options (1) => MSS: 1460
[Xref => http://doc.emergingthreats.net/2002911]
[**] [1:2001219:19] ET SCAN Potential SSH Scan [**]
[Classification: Attempted Information Leak] [Priority: 2]
10/25-00:27:30.308375 192.168.1.15:48866 -> 192.168.1.3:22
TCP TTL:41 TOS:0x0 ID:39400 IpLen:20 DgmLen:60
******S* Seq: 0x87730F4F  Ack: 0x9E743CCD  Win: 0x3F  TcpLen: 40
TCP Options (5) => MSS: 1400 WS: 0 SackOK TS: 4294967295 0 EOL
[Xref => http://doc.emergingthreats.net/2001219][Xref => http://en.wikipedia.org/wiki/Brute_force_attack]
[**] [1:2018489:3] ET SCAN NMAP OS Detection Probe [**]
[Classification: Attempted Information Leak] [Priority: 2]
10/25-00:27:30.789909 192.168.1.15:48719 -> 192.168.1.3:34005
UDP TTL:63 TOS:0x0 ID:4162 IpLen:20 DgmLen:328
Len: 300
```

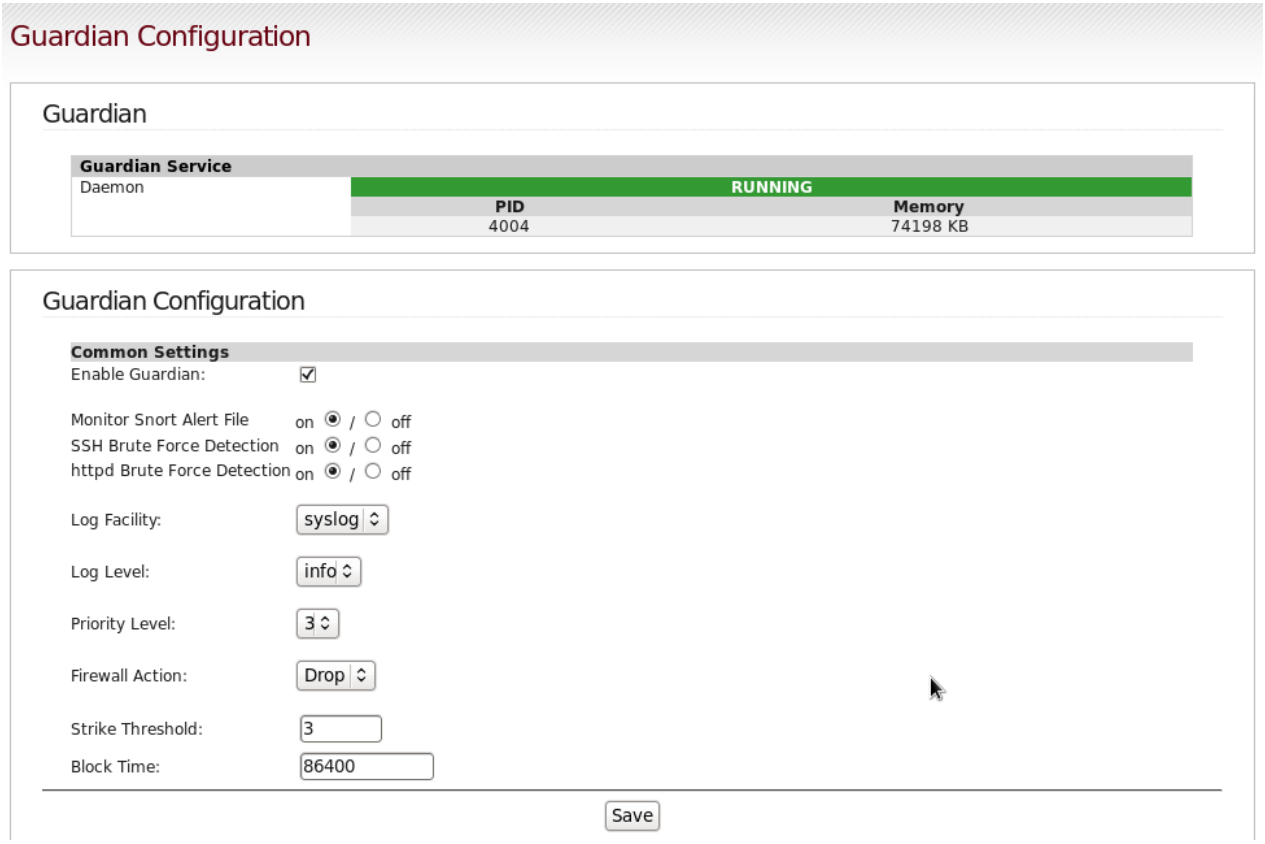
Cảnh báo này cũng có thể được xem trong mục *IDS Logs* (menu *Log/IDS Logs*):



Bước 3: Thiết lập ngưỡng cảnh báo và kích hoạt IPS

Sau khi hệ thống IDS của IPFire đã phát hiện được các hành động tấn công quét cổng, bước cuối cùng là tự động ngăn chặn các cuộc tấn công này. Vào menu *Services/Guardian* để thiết lập các thông số cần thiết (có thể để tất cả các thông số theo giá trị mặc định) và kích hoạt modul Guardian (chọn checkbox *Enable Guardian* và

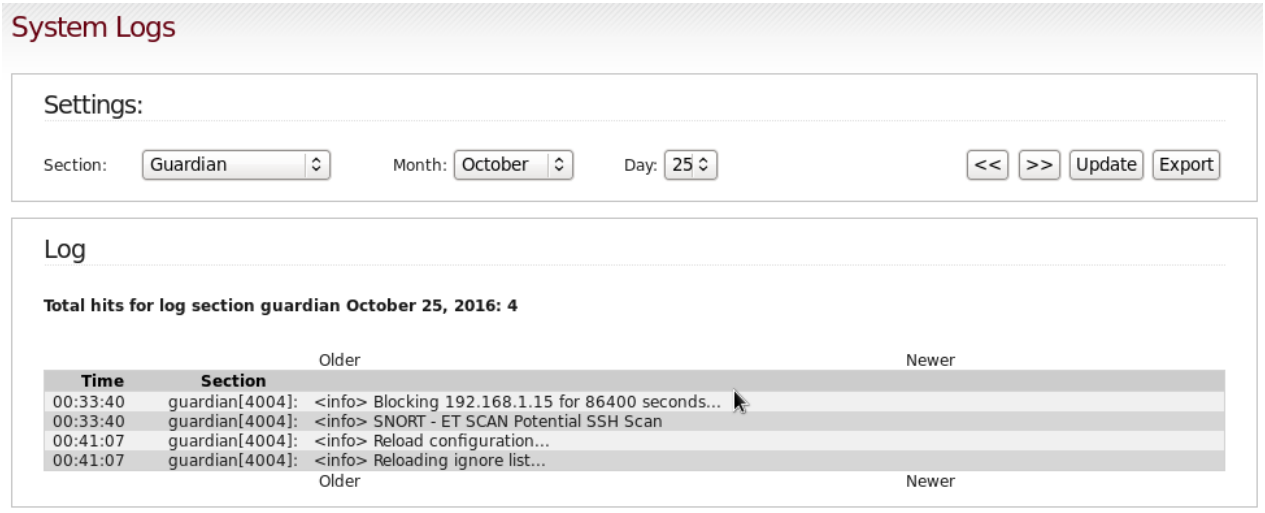
click “Save”). Sau khi modul Guardian được kích hoạt thành công, trạng thái của nó sẽ được hiển thị trong phần *Guardian Service: Deamon* là RUNNING cùng với PID của nó:



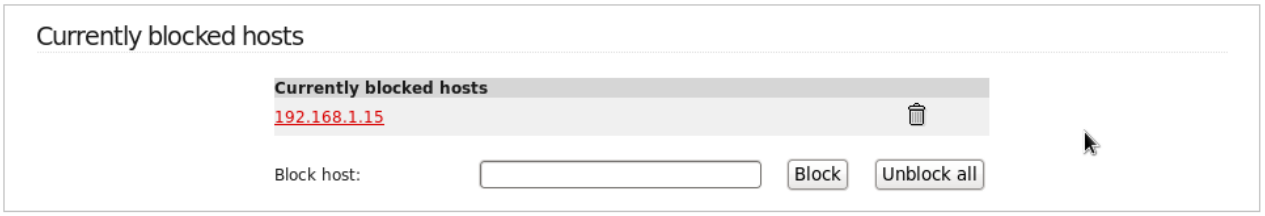
Sử dụng *nmap* trên một trạm có địa chỉ 192.168.1.15 và quét cổng trên máy IPFire:

```
[root@C2 ~]# nmap -v -A 192.168.1.3
Starting Nmap 5.51 ( http://nmap.org ) at 2016-10-25 00:07 ICT
NSE: Loaded 57 scripts for scanning.
Initiating ARP Ping Scan at 00:07
Scanning 192.168.1.3 [1 port]
. . .
```

Kiểm tra các thông tin log trên IPFire sẽ thấy khi cuộc tấn công quét cổng đạt đến ngưỡng giới hạn đã thiết lập (3), Guardian tự động block địa chỉ IP của máy đang thực hiện tấn công với thời gian là 86400 giây:



Trang Guardian (menu Services/Guardian) cho thấy danh sách *Curently blocked hosts* chứa địa chỉ IP của máy vừa thực hiện tấn công quét cổng:



Mọi hành động truy nhập đến máy IPFire từ địa chỉ này đều bị ngăn chặn, ví dụ lệnh *ping*:

```
[root@C2 ~]# ping 192.168.1.3
PING 192.168.1.3 (192.168.1.3) 56(84) bytes of data.
^C
--- 192.168.1.3 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2531ms
```

Quá trình block địa chỉ IP này sẽ kéo dài 24h hoặc cho đến khi người quản trị hủy bỏ địa chỉ IP này trong danh sách *blocked hosts*.