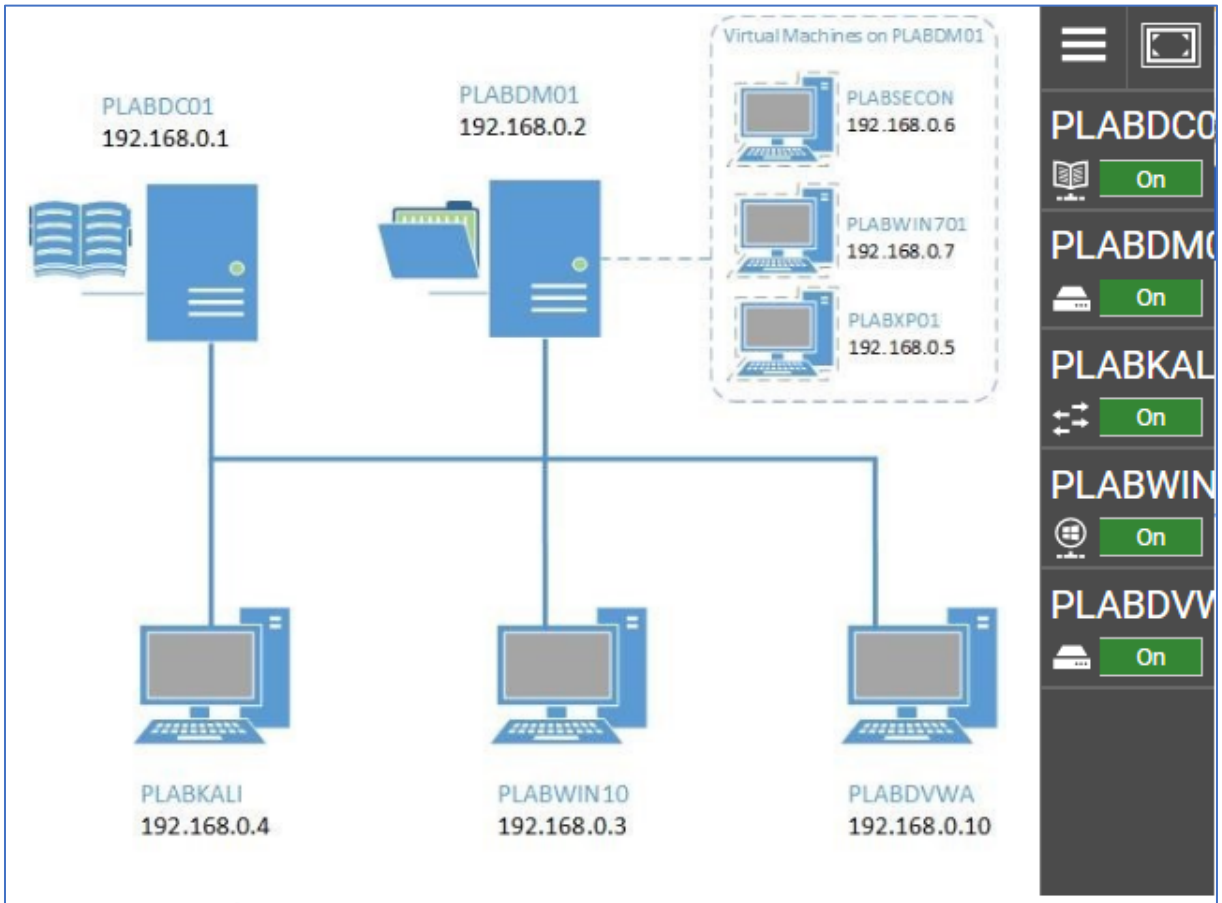
Snort : Windows Version

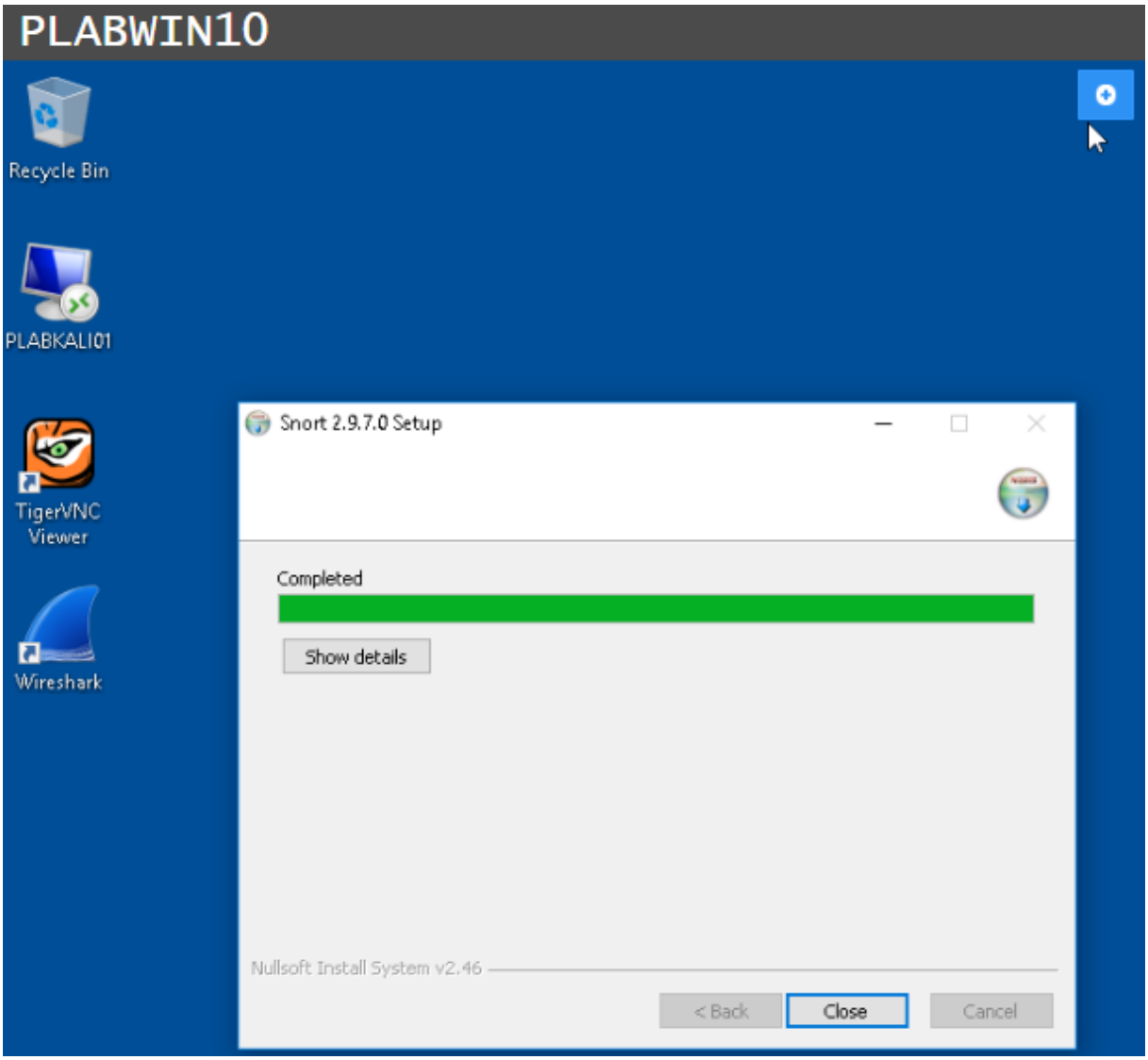


Connecting to your lab

In this module, you will be working on the following equipment to carr

- **PLABDC01** (Windows Server 2016 - Domain Controller)
- **PLABDM01** (Windows Server 2016 - Member Server)
- **PLABWIN10** (Windows 10 - Domain Member)
- **PLABKALI01** (Kali 2016.2)
- **PLABDVWA** (Fedora)
- **PLABSECON** (Ubuntu-Security Onion)
- **PLABWIN701** (Windows 7)
- **PLABXP01** (Windows XP)

Installing Snort

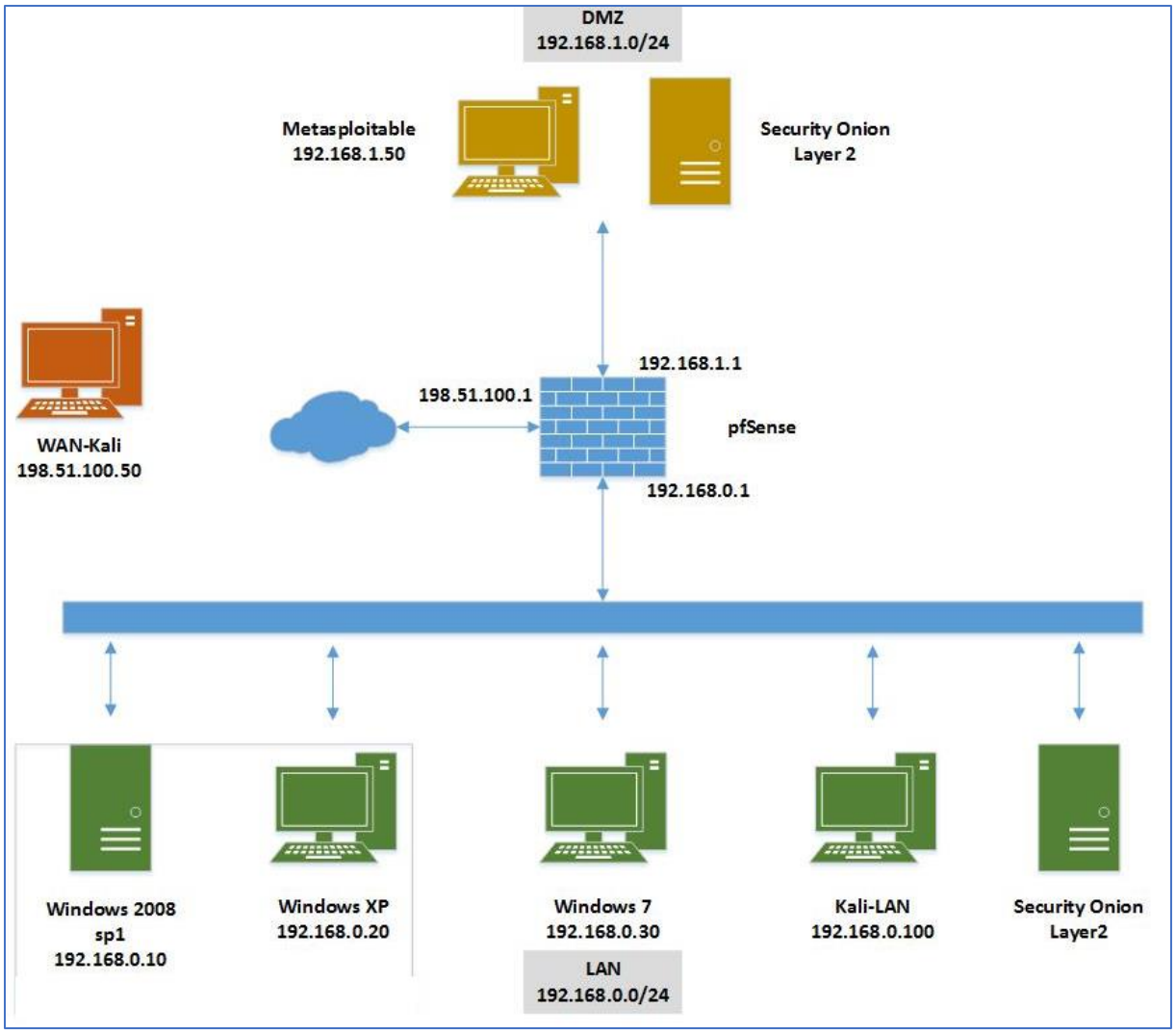


Launching Snort with command lines

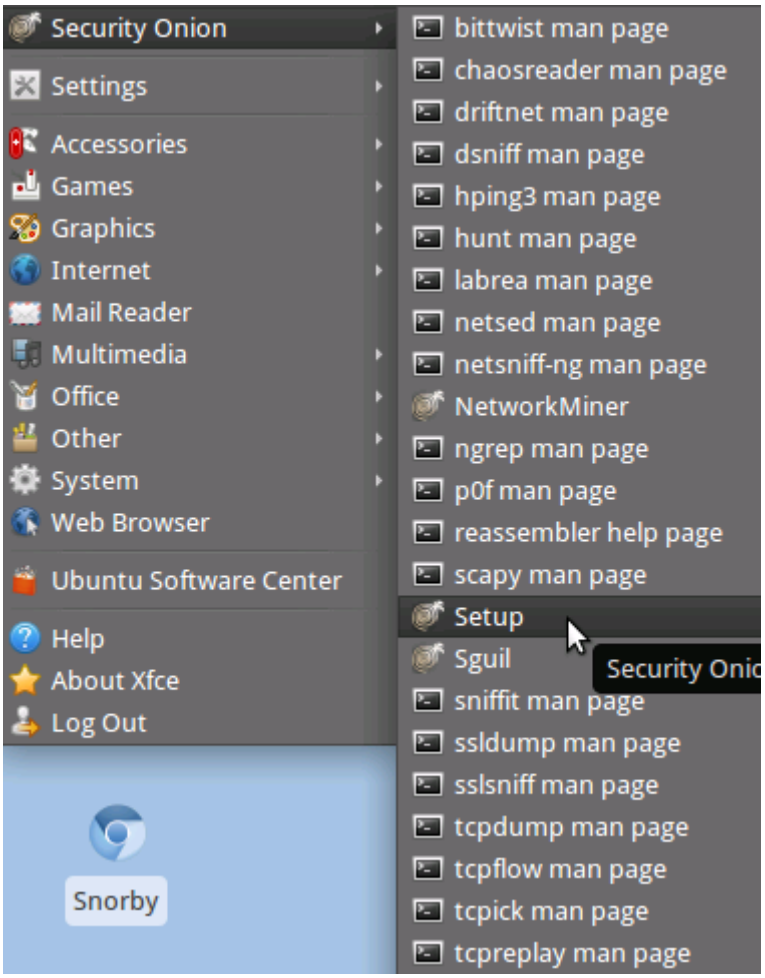
```
C:\Snort>dir
 Volume in drive C has no label.
 Volume Serial Number is BCA5-9EBD

 Directory of C:\Snort

04/10/2020  03:00 PM    <DIR>          .
04/10/2020  03:00 PM    <DIR>          ..
04/10/2020  03:00 PM    <DIR>          bin
04/10/2020  03:00 PM    <DIR>          doc
04/10/2020  03:00 PM    <DIR>          etc
04/10/2020  03:00 PM    <DIR>          lib
04/10/2020  03:00 PM    <DIR>          log
04/10/2020  03:00 PM    <DIR>          preproc_rules
04/10/2020  03:00 PM    <DIR>          rules
04/10/2020  03:00 PM            50,103 Uninstall.exe
               1 File(s)         50,103 bytes
               9 Dir(s)  16,832,606,208 bytes free

C:\Snort>cd bin

C:\Snort\bin>snort -W

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.8.3-WIN32 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 PCRE version: 8.10 2010-06-25
           Using ZLIB version: 1.2.3

Index   Physical Address        IP Address      Device Name     Description
-----   ----------------        ----------      -----------     -----------
```

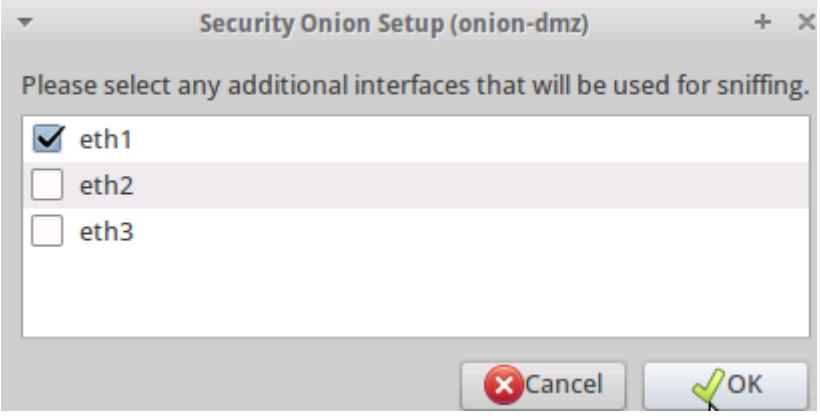
Snort : Linux version using Security Onion

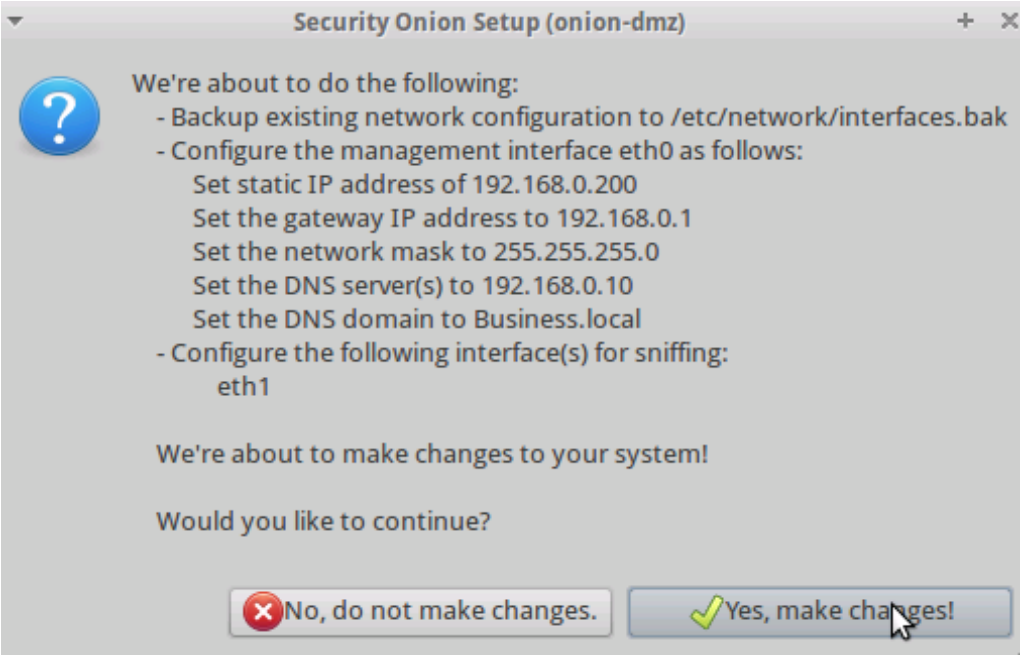
Lab Topology


DMZ
192.168.1.0/24
Metasploitable
192.168.1.50
Security Onion
Layer 2
192.168.1.1
198.51.100.1
pfSense
WAN-Kali
198.51.100.50
192.168.0.1
Windows 2008
sp1
192.168.0.10
Windows XP
192.168.0.20
Windows 7
192.168.0.30
Kali-LAN
192.168.0.100
Security Onion
Layer2
LAN
192.168.0.0/24

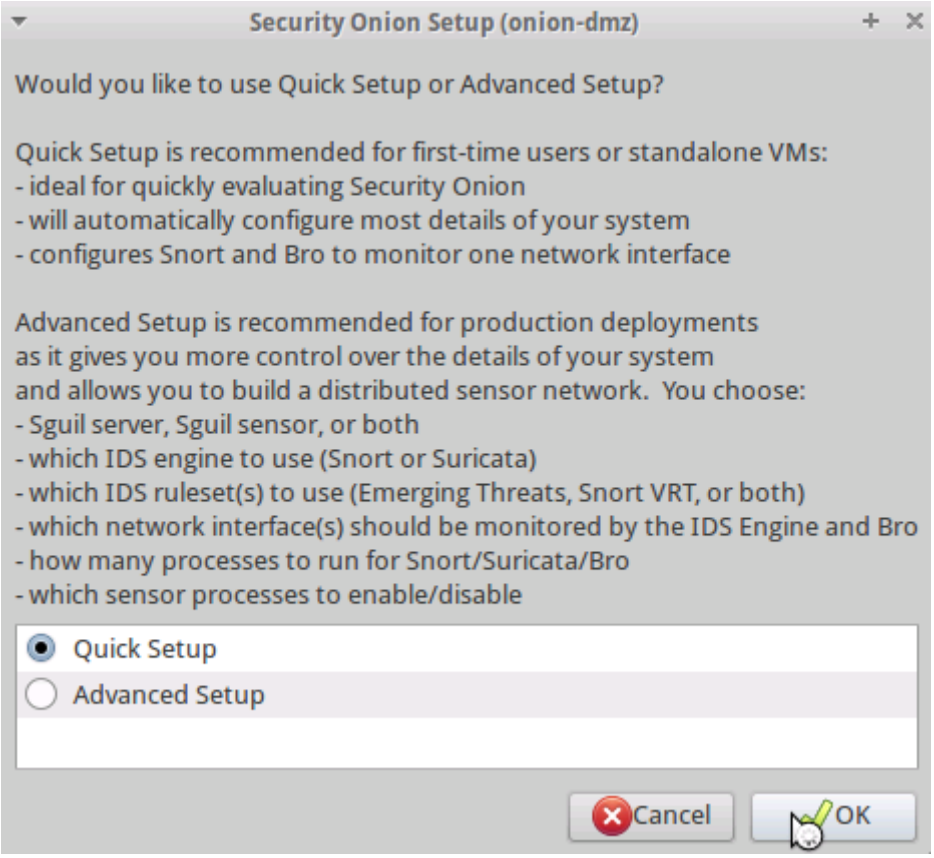
Setup icon to begin setting up Security Onion to function as a Network based IDS.

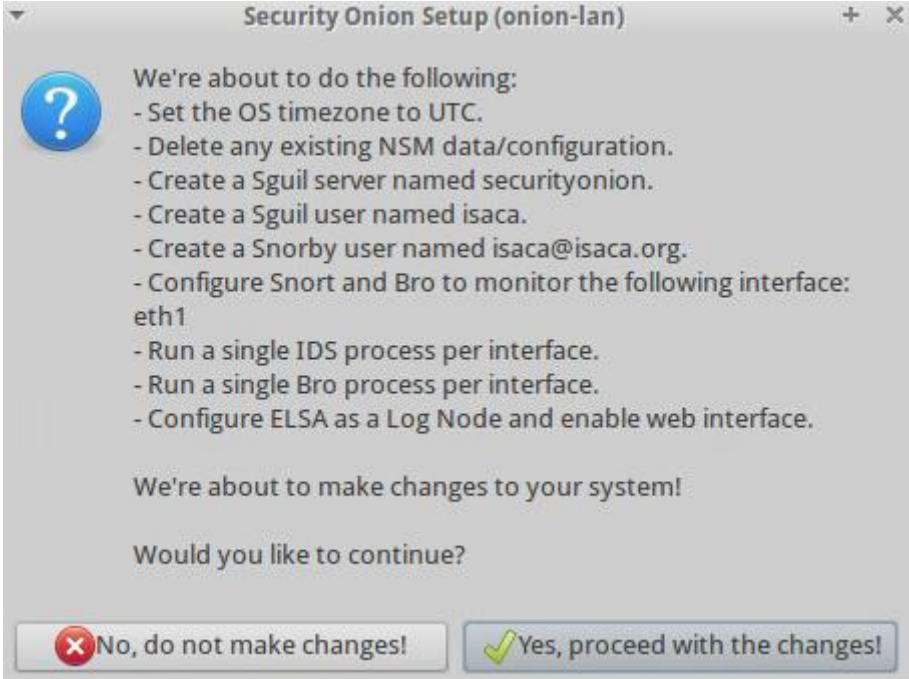

Security Onion
Settings
Accessories
Games
Graphics
Internet
Mail Reader
Multimedia
Office
Other
System
Web Browser
Ubuntu Software Center
Help
About Xfce
Log Out
Snorby
bittwist man page
chaosreader man page
driftnet man page
dsniff man page
hping3 man page
hunt man page
labrea man page
netsed man page
netsniff-ng man page
NetworkMiner
ngrep man page
p0f man page
reassembler help page
scapy man page
Setup
Sguil
Security Onic
sniffit man page
ssldump man page
sslsniff man page
tcpdump man page
tcpflow man page
tcpick man page
tcpreplay man page


Security Onion Setup (onion-dmz)
Please select any additional interfaces that will be used for sniffing.
eth1
eth2
eth3
Cancel
OK

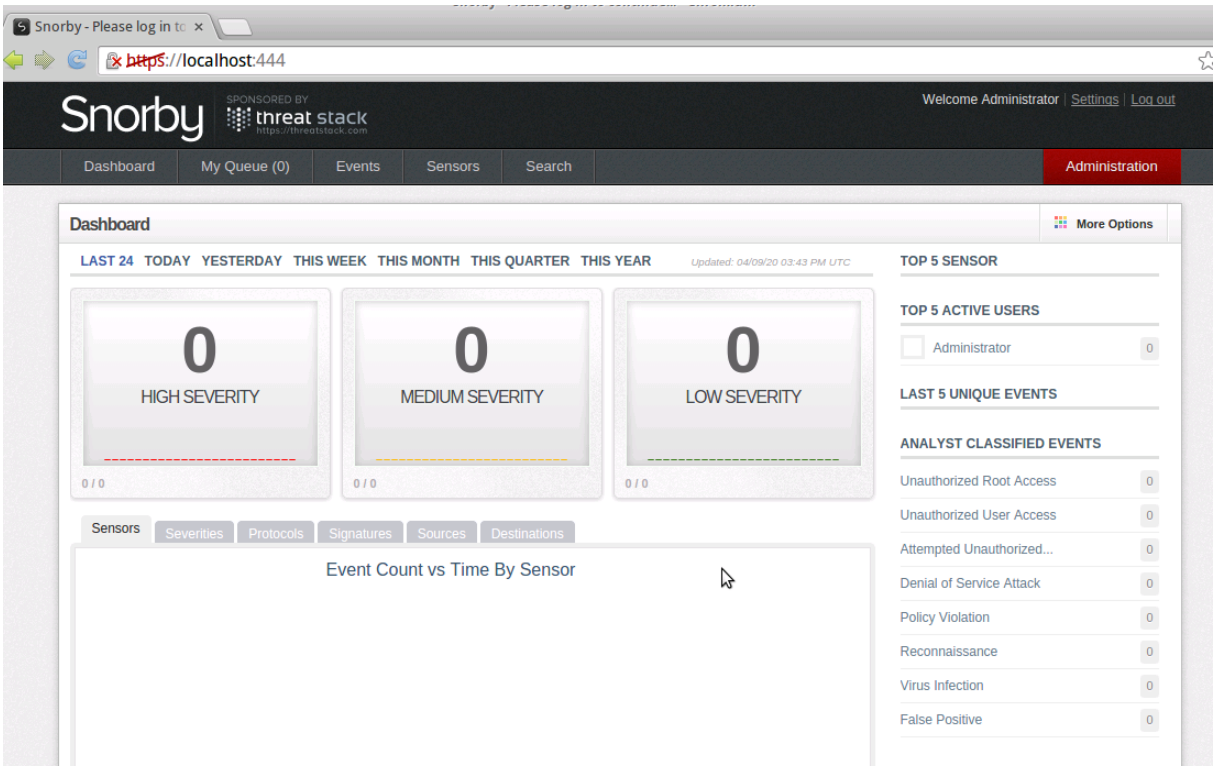


After the sytem reboot, I continued the setup





I am now logged in to the Snorby (a ruby-on-rails app using Snort, Suricata and Sagan) interface of the IDS and ready to analyze any new alerts.



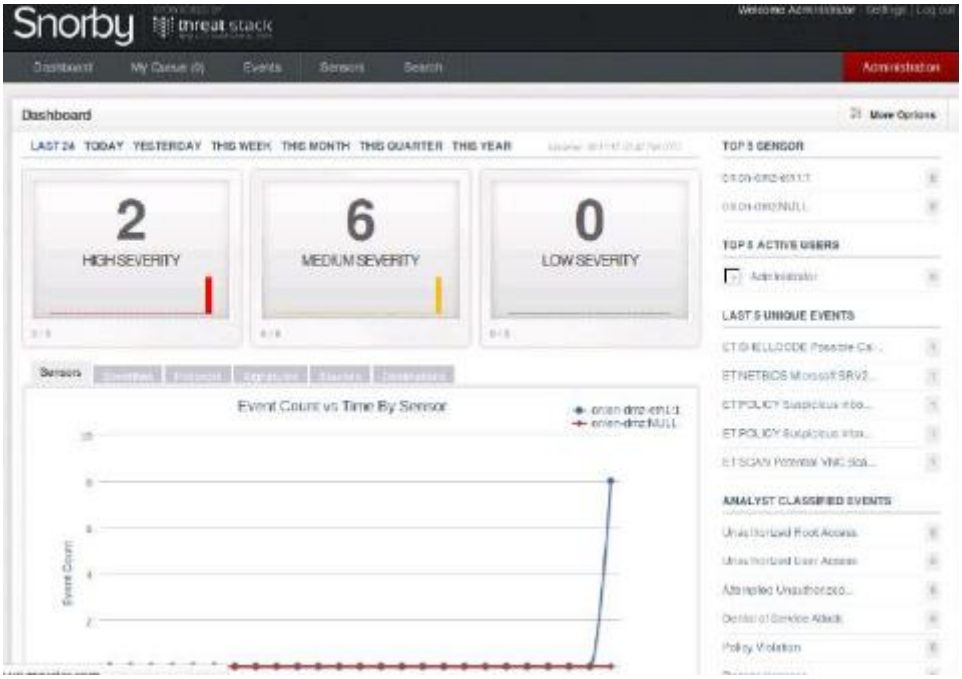
On the kali machine, I launch a nmap scan, easy to detect on our IDS machine

```
Terminal
File  Edit  View  Search  Terminal  Help
root-kali$nmap 192.168.0.30

Starting Nmap 7.25BETA1 ( https://nmap.org ) at 2020-04-09 11:46 EDT
mass_dns: warning: Unable to open /etc/resolv.conf. Try using --system-dns or sp
ecify valid servers with --dns-servers
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
 Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.0.30
Host is up (0.00045s latency).
Not shown: 992 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
8000/tcp  open  http-alt
8089/tcp  open  unknown
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
MAC Address: 00:15:5D:01:80:03 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 1.87 seconds
```

Then with Metasploit using a smb exploit, I simulate an attack aimed at the Windows 192.168.0.30

```
msf exploit(ms09_050_smb2_negotiate_func_index) > set lhost 5555
lhost => 5555
msf exploit(ms09_050_smb2_negotiate_func_index) > run

[*] Started reverse TCP handler on 0.0.21.179:4444
[*] 192.168.0.30:445 - Connecting to the target (192.168.0.30:445)...
[*] 192.168.0.30:445 - Sending the exploit packet (930 bytes)...
[*] 192.168.0.30:445 - Waiting up to 180 seconds for exploit to trigger...
```

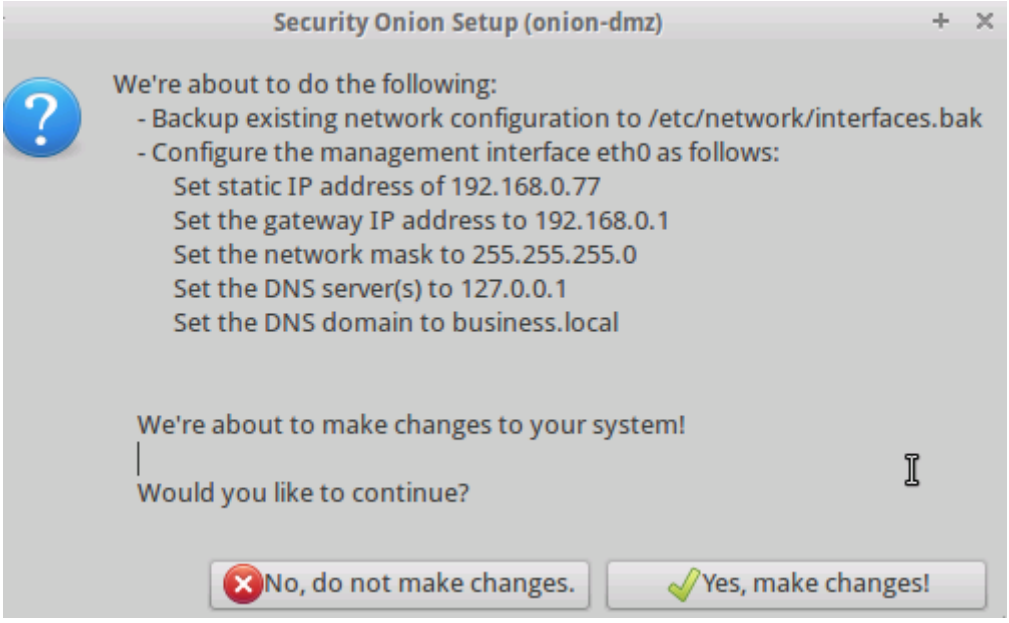
The attack are detected on the IDS machine



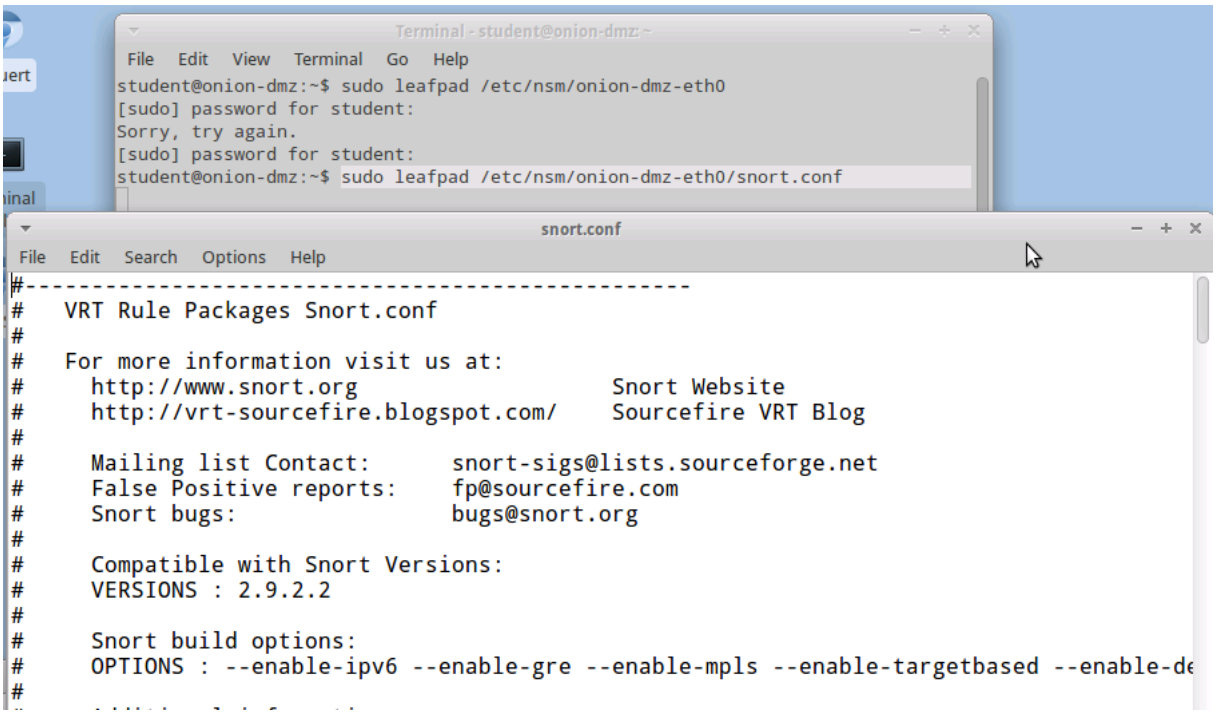
Now let's say that I and have received intelligence that suspected attacker have set up a C2 controller (Server) on our network and are communicating with it. I want to create snort rules that alert you to traffic to 172.16.200.2 and DNS lookups on Port 53 to monitor what DNS servers are being queried as it is suspected that DNS is being used as a covert channel to create outbound connections.

I have also noticed that the suspected attackers download nc.exe from ftp servers and deploy them on suspected systems. You want to see which servers in your enterprise are downloading this tool to be away of which systems have been possibly compromised.

Following the previous steps, I used the following setup:



Once the system rebooted I look at some of the existing rules within snort



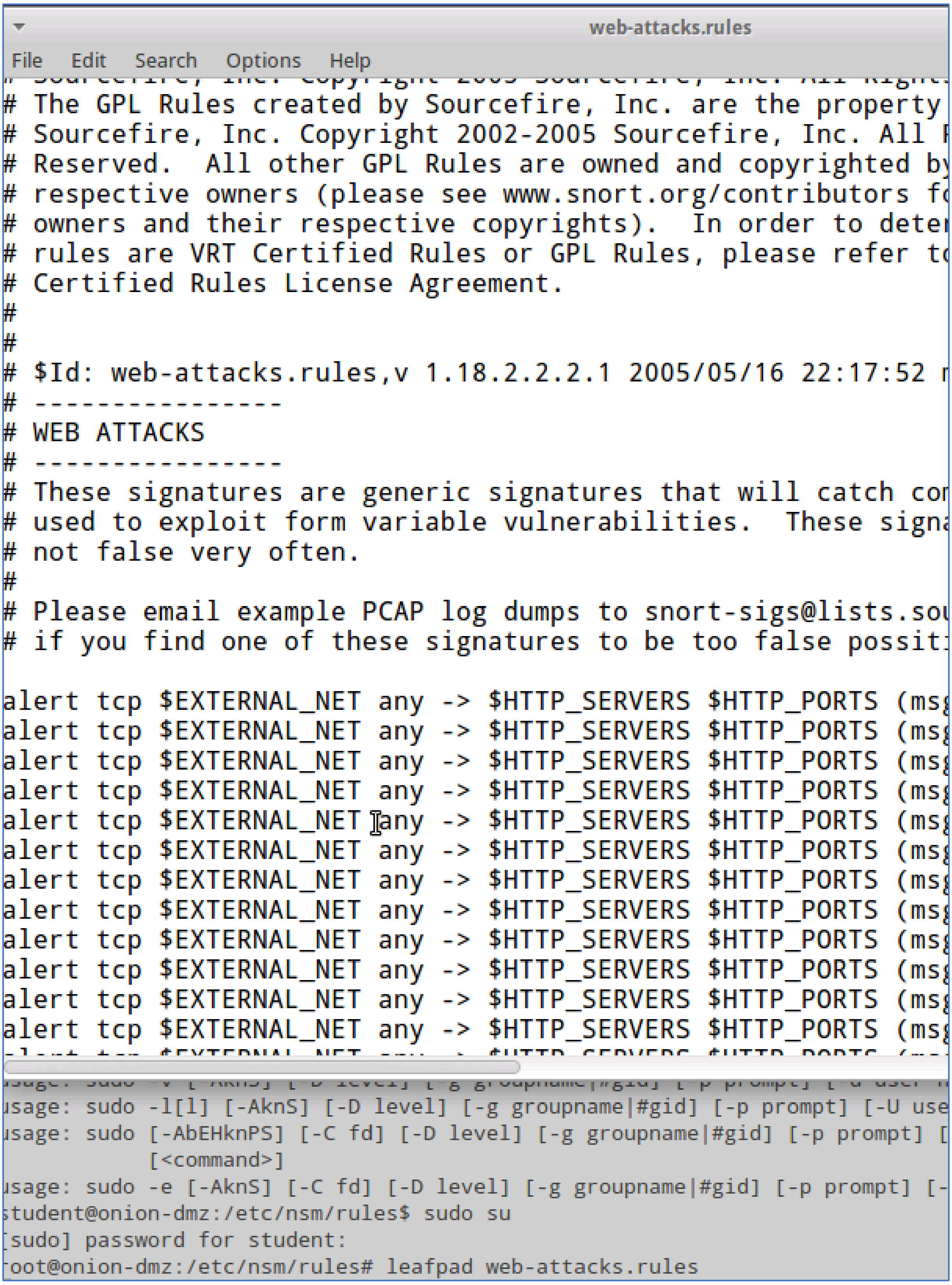
```
# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
var RULE_PATH /etc/nsm/rules
var SO_RULE_PATH /etc/nsm/so_rules
var PREPROC_RULE_PATH /etc/nsm/preproc_rules
```

The var is set to /etc/nsm/rules.

Let's browse to that directory

```
^Cstudent@onion-dmz:~$ cd /etc/nsm/rules
student@onion-dmz:/etc/nsm/rules$ ls -la
total 1608
drwxr-xr-x 2 root root   4096 Jun  4  2015 .
drwxr-xr-x 8 root root   4096 Jun  4  2015 ..
-r-------- 1 root root   5520 Jun  4  2015 attack-responses.rules
-r-------- 1 root root  17898 Jun  4  2015 backdoor.rules
-r-------- 1 root root   3862 Jun  4  2015 bad-traffic.rules
-r-------- 1 root root   7994 Jun  4  2015 chat.rules
-r-------- 1 root root  12759 Jun  4  2015 community-bot.rules
-r-------- 1 root root   1223 Jun  4  2015 community-deleted.rules
```

Checking the webrules file



```
# The GPL Rules created by Sourcefire, Inc. are the property
# Sourcefire, Inc. Copyright 2002-2005 Sourcefire, Inc. All
# Reserved.  All other GPL Rules are owned and copyrighted by
# respective owners (please see www.snort.org/contributors fo
# owners and their respective copyrights).  In order to dete
# rules are VRT Certified Rules or GPL Rules, please refer to
# Certified Rules License Agreement.
#
#
# $Id: web-attacks.rules,v 1.18.2.2.2.1 2005/05/16 22:17:52
# ----------------
# WEB ATTACKS
# ----------------
# These signatures are generic signatures that will catch co
# used to exploit form variable vulnerabilities.  These sign
# not false very often.
#
# Please email example PCAP log dumps to snort-sigs@lists.so
# if you find one of these signatures to be too false possit

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (ms
```

```
usage: sudo -l[l] [-AknS] [-D level] [-g groupname|#gid] [-p prompt] [-U use
usage: sudo [-AbEHknPS] [-C fd] [-D level] [-g groupname|#gid] [-p prompt] [
            [<command>]
usage: sudo -e [-AknS] [-C fd] [-D level] [-g groupname|#gid] [-p prompt] [-
student@onion-dmz:/etc/nsm/rules$ sudo su
[sudo] password for student:
root@onion-dmz:/etc/nsm/rules# leafpad web-attacks.rules
```

Now let’s create custom rules

```
root@onion-dmz:/etc/nsm/onion-dmz-eth0# cd /etc/nsm/rules
root@onion-dmz:/etc/nsm/rules# sudo leafpad custom.rules
```

```
# These are rules in response to todays incident
alert tcp any any -> any 21 (msg:"Established FTP Connections"; flags:S; sid:10000;)
alert ip any any <> 172.16.200.2 any (msg:" KNOWN C2 Server "; sid:100001;)
alert tcp any any <> any any (msg: "Looking for nc.exe "; content:"nc.exe"; nocase; sid:10002;)
alert udp any any <> any any (msg: "DNS Requests ";sid:10004;)
```

Snort rules consist of the following;

alert tcp any any -> any 21 (msg: "Establshed FTP connections "; flags:S; sid:10000;)

alert - what we want this rule to do (options include log, activate,reject)

tcp - the protocol we are monitoring (options include tcp,udp,icmp,ip)

any - the source ip address(s) or network (CIDR allowed)

any - The source port(s) we wish to monitor

-> the direction we wish to monitor -> outbound <- inbound <> both directions

any - The desination IP address (CIDR allowed)

21 - The destination port we wish to monitor

msg: - What message we wish to be included in the message

flags:S - This is the sync flag in a tcp connection.

sid:10001 - This is the security ID . Newer versions of snort required a security identifier with each log and will complain with they are duplicates. These are user created and must contain only numbers.

On the **snort.conf** we also add include **$RULE_PATH/custom.rules**

```
*snort.conf
File  Edit  Search  Options  Help
# ensure smoother upgrades to future versions of this package.
include database.conf
#

# prelude
# output alert_prelude

# metadata reference data.  do not modify these lines
include classification.config
include reference.config


##################################################
# Step #7: Customize your rule set
# For more information, see Snort Manual, Writing Snort Rules
#
# NOTE: All categories are enabled in this conf file
##################################################

# Note for Debian users: The rules preinstalled in the system
# can be *very* out of date. For more information please read
# the /usr/share/doc/snort-rules-default/README.Debian file

# site specific rules
include $RULE_PATH/local.rules
include $RULE_PATH/custom.rules
```

Now I test the custom rules by running it against a packet capture that contains data your alerts SHOULD fire on. I should always test my rules AGAINST a sample data set to ensure desired results. From a terminal emulator type the following:

sudo snort -l . -c /etc/nsm/onion-dmz-eth0/snort.conf -r /home/student/capture131.pcap

-l - Where to log the alerts to (here we use "." to signify current working directory)
-c - Which configuration file to use
-r - Which capture file to read into snort

```
student@onion-dmz:/etc/nsm/rules$ cat alert | grep FTP
[**] [1:10000:0] Established FTP Connections  [**]
[**] [1:553:7] POLICY FTP anonymous login attempt [**]
[**] [1:489:7] INFO FTP no password [**]
[**] [1:10000:0] Established FTP Connections  [**]
[**] [1:553:7] POLICY FTP anonymous login attempt [**]
[**] [1:489:7] INFO FTP no password [**]
```

Now let's say that there are credentials going across the wire in plain text. I will use Wireshark and Snort to examine a pcap file that the network team handed over to you

the syntax of snort:
the -l specifies where to place the log file, in this case it will be a single alert file.
the -c will specify the location on the hard disk for the snort.conf file
the -r tells snort to read the following capture file

```
      Total sessions: 7
      Packet stats
        Packets: 100
        Ignored bytes: 1159
        Maximum outstanding requests: 1
        SMB command requests/responses processed
          Close (0x04) : 1/1
          Transaction (0x25) : 6/2
            TRANS_TRANSACT_NMPIPE (0x0026) : 2/2
          Transaction2 (0x32) : 3/0
            TRANS2_FIND_FIRST2 (0x0001) : 1/0
            TRANS2_QUERY_FS_INFORMATION (0x0003) : 1/0
            TRANS2_QUERY_PATH_INFORMATION (0x0005) : 1/0
          Tree Disconnect (0x71) : 5/5
          Negotiate (0x72) : 7/7
          Session Setup AndX (0x73) : 14/14
          Logoff AndX (0x74) : 2/2
          Tree Connect AndX (0x75) : 6/6
          Nt Create AndX (0xA2) : 1/1

  DCE/RPC
    Connection oriented
      Packet stats
        PDUs: 4
          Bind: 1
          Bind Ack: 1
          Request: 1
          Response: 1
        Request fragments: 0
        Response fragments: 0
        Client PDU segmented reassembled: 0
        Server PDU segmented reassembled: 0
===============================================================================
===============================================================================
SIP Preprocessor Statistics
  Total sessions: 0
===============================================================================
+-----------------------[filtered events]-------------------------------------
| gen-id=1      sig-id=2923       type=Threshold tracking=dst count=10  seconds=
60  filtered=1
| gen-id=1      sig-id=2924       type=Threshold tracking=dst count=10  seconds=
60  filtered=1
Snort exiting
student-onion-dmz$sudo snort -l /home/student -c /etc/nsm/onion-dmz-eth0/snort.c
```

sudo snort -l /home/student -c /etc/nsm/onion-dmz-eth0/snort.conf -r capture131.pcap

Once snort exits, an alert file is generated.

```
student-onion-dmz$ls
alert           Desktop    manifest.txt  Public
capture131f.cap Documents  Music         tcpdump.log.1586451423
capture131.pcap Downloads  Pictures      Templates
```

```
<alert>
File  Edit  Search  Options  Help
bid/9752]

[**] [1:538:15] NETBIOS SMB IPC$ unicode share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
03/03-21:00:16.051222 198.51.100.50:35271 -> 192.168.1.50:139
TCP TTL:63 TOS:0x0 ID:2916 IpLen:20 DgmLen:146 DF
***AP*** Seq: 0x2B1302C7  Ack: 0x9BFB099E  Win: 0xF5  TcpLen: 32
TCP Options (3) => NOP NOP TS: 62556 159291

[**] [1:2404:5] NETBIOS SMB-DS Session Setup AndX request unicode username overflow attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
03/03-21:03:47.047745 198.51.100.50:37902 -> 192.168.1.50:445
TCP TTL:63 TOS:0x0 ID:3837 IpLen:20 DgmLen:494 DF
***AP*** Seq: 0xEAE5CE06  Ack: 0x5FDDBF59  Win: 0xF5  TcpLen: 32
TCP Options (3) => NOP NOP TS: 115305 180189
[Xref => http://www.eeye.com/html/Research/Advisories/AD20040226.html][Xref => http://www.securit
bid/9752]
```

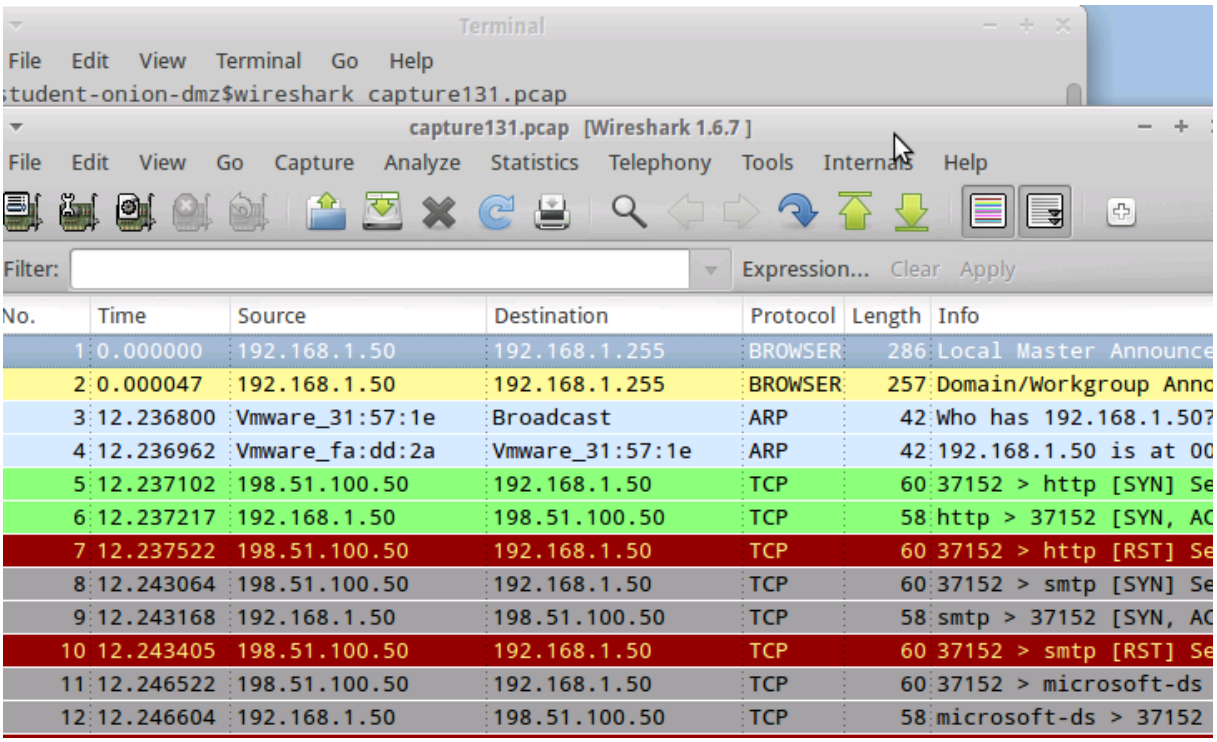
They both deal with FTP (File Transfer Protocol), which is sent over the network in clear text. Specifically, we see anonymous logins, which means a user logged on to the system without credentials.

```
[**] [1:553:7] POLICY FTP anonymous login attempt [**]
[Classification: Misc activity] [Priority: 3]
03/03-20:56:48.694625 198.51.100.50:34305 -> 192.168.1.50:21
TCP TTL:128 TOS:0x0 ID:233 IpLen:20 DgmLen:50 DF
***AP*** Seq: 0xCA47FBE3  Ack: 0xD97C3E4B  Win: 0xFFEB  TcpLen: 20

[**] [1:489:7] INFO FTP no password [**]
[Classification: Unknown Traffic] [Priority: 3]
03/03-20:56:49.670791 198.51.100.50:34305 -> 192.168.1.50:21
TCP TTL:128 TOS:0x0 ID:236 IpLen:20 DgmLen:47 DF
***AP*** Seq: 0xCA47FBED  Ack: 0xD97C3E6D  Win: 0xFFC9  TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS322]
```

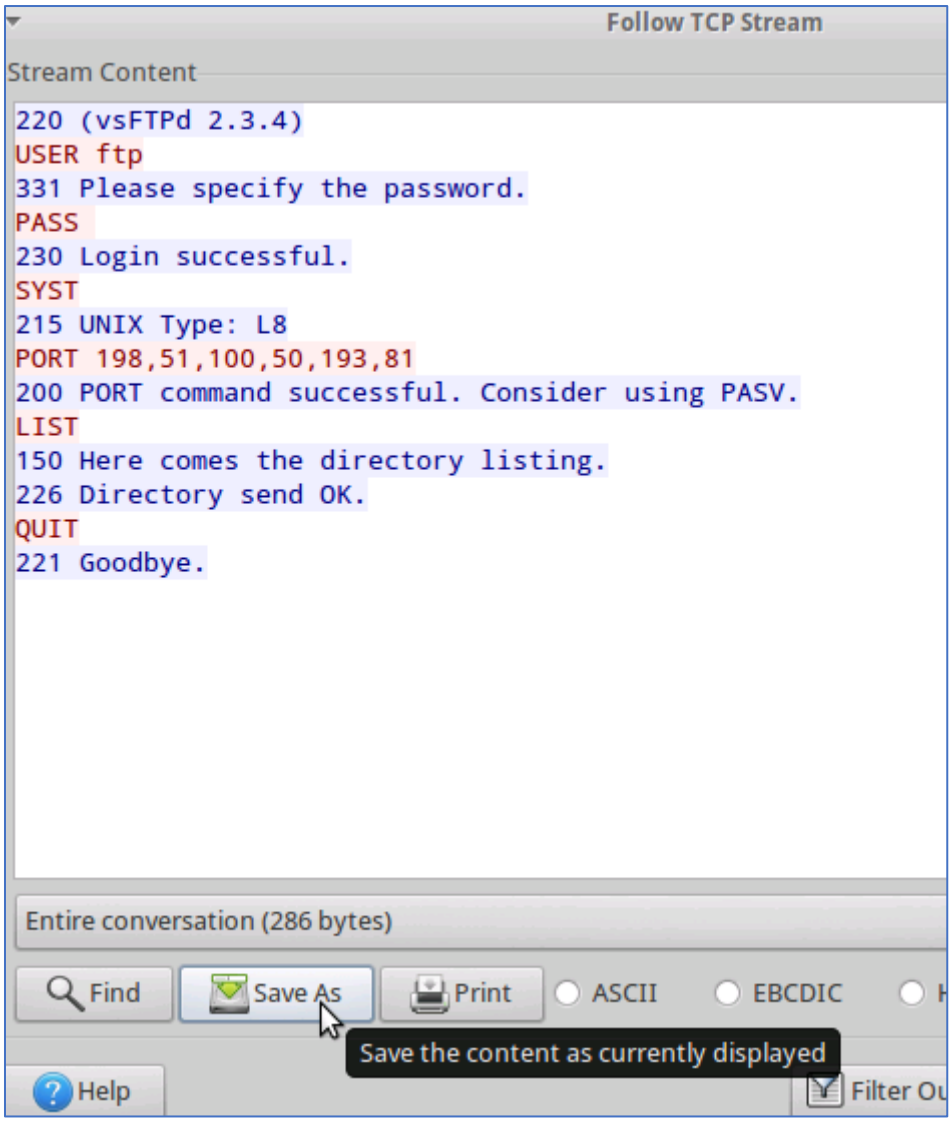
Next, let's view the next alert (telnet connection). Since data is also sent over plain-text with Telnet, you can see the traffic details. Also notice that this traffic is coming from an IP external to the network.

```
[**] [1:716:13] INFO TELNET access [**]
[Classification: Not Suspicious Traffic] [Priority: 3]
03/03-20:57:13.970065 192.168.1.50:23 -> 198.51.100.50:60809
TCP TTL:64 TOS:0x10 ID:20903 IpLen:20 DgmLen:64 DF
***AP*** Seq: 0xE9AA7247  Ack: 0xE1D8658B  Win: 0xB5  TcpLen: 32
TCP Options (3) => NOP NOP TS: 141258 14511
[Xref => http://cgi.nessus.org/plugins/dump.php3?id=10280][Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?
name=1999-0619][Xref => http://www.whitehats.com/info/IDS08]
```

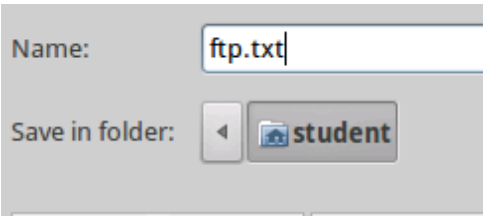
I open the capture file with Wireshark

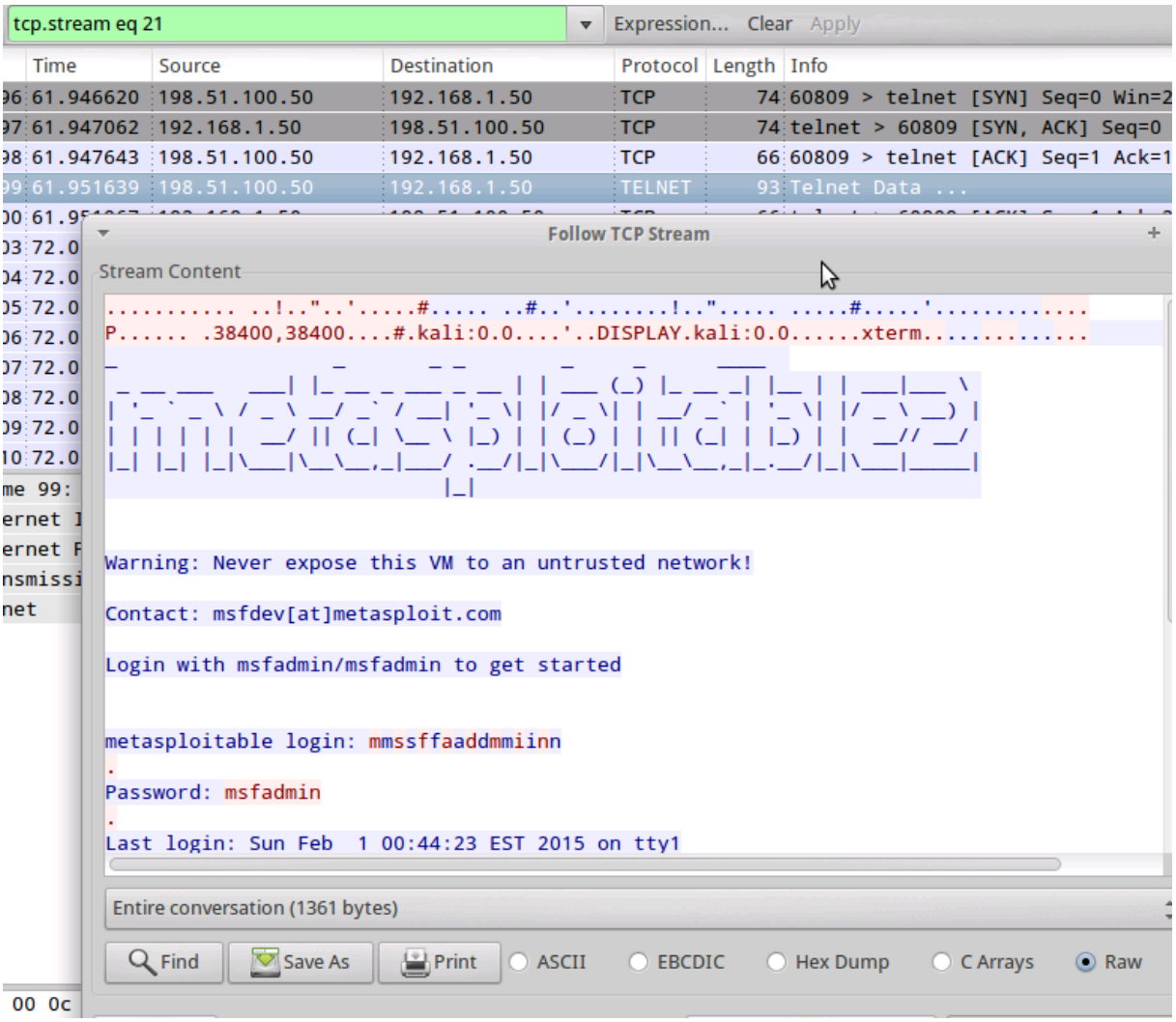


The FTP login successfully worked without any password

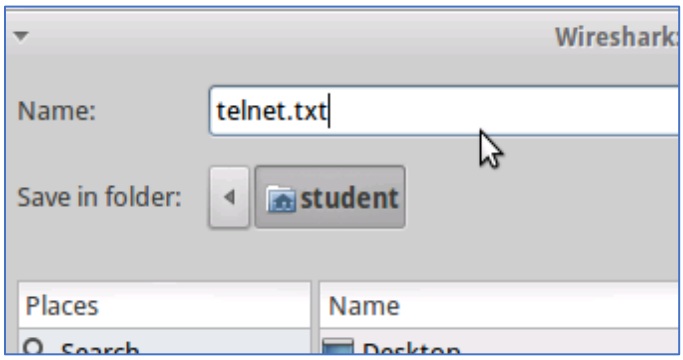


Saving the file as **ftp.txt**





Saving the plain text in a file called **telnet.txt**



er: smb and tcp and ip.addr == 198.51.100.50 Expression... Clear Apply

	Time	Source	Destination	Protocol	Length	Info
262	254.040705	198.51.100.50	192.168.1.50			
264	254.041774	192.168.1.50	198.51.100.50			
266	254.046138	198.51.100.50	192.168.1.50			
267	254.047178	192.168.1.50	198.51.100.50			
268	254.050774	198.51.100.50	192.168.1.50			
270	254.096577	192.168.1.50	198.51.100.50			
271	254.096996	198.51.100.50	192.168.1.50			
273	254.097207	192.168.1.50	198.51.100.50			
274	254.097701	198.51.100.50	192.168.1.50			
275	254.097885	192.168.1.50	198.51.100.50			
276	254.099470	198.51.100.50	192.168.1.50			
277	254.099765	192.168.1.50	198.51.100.50			

Mark Packet (toggle)
Ignore Packet (toggle)
Set Time Reference (tog
Manually Resolve Addre
Apply as Filter
Prepare a Filter
Conversation Filter
Colorize Conversation
SCTP
Follow TCP Stream