Who am I?

Abdirahman Hish Mohamed

Cyber Security Consultant

Sentinel Africa Consulting

Abdirahman[dot]Mohamed[at]sentinelafrica [dot]co[dot]ke

@abdihakx

NetSec-Pi:
Deploying a Defensive Raspberry Pi in Your Home Network/ IoT Environment

United States International University-Africa
Sentinel Africa
Risk Management Consultants

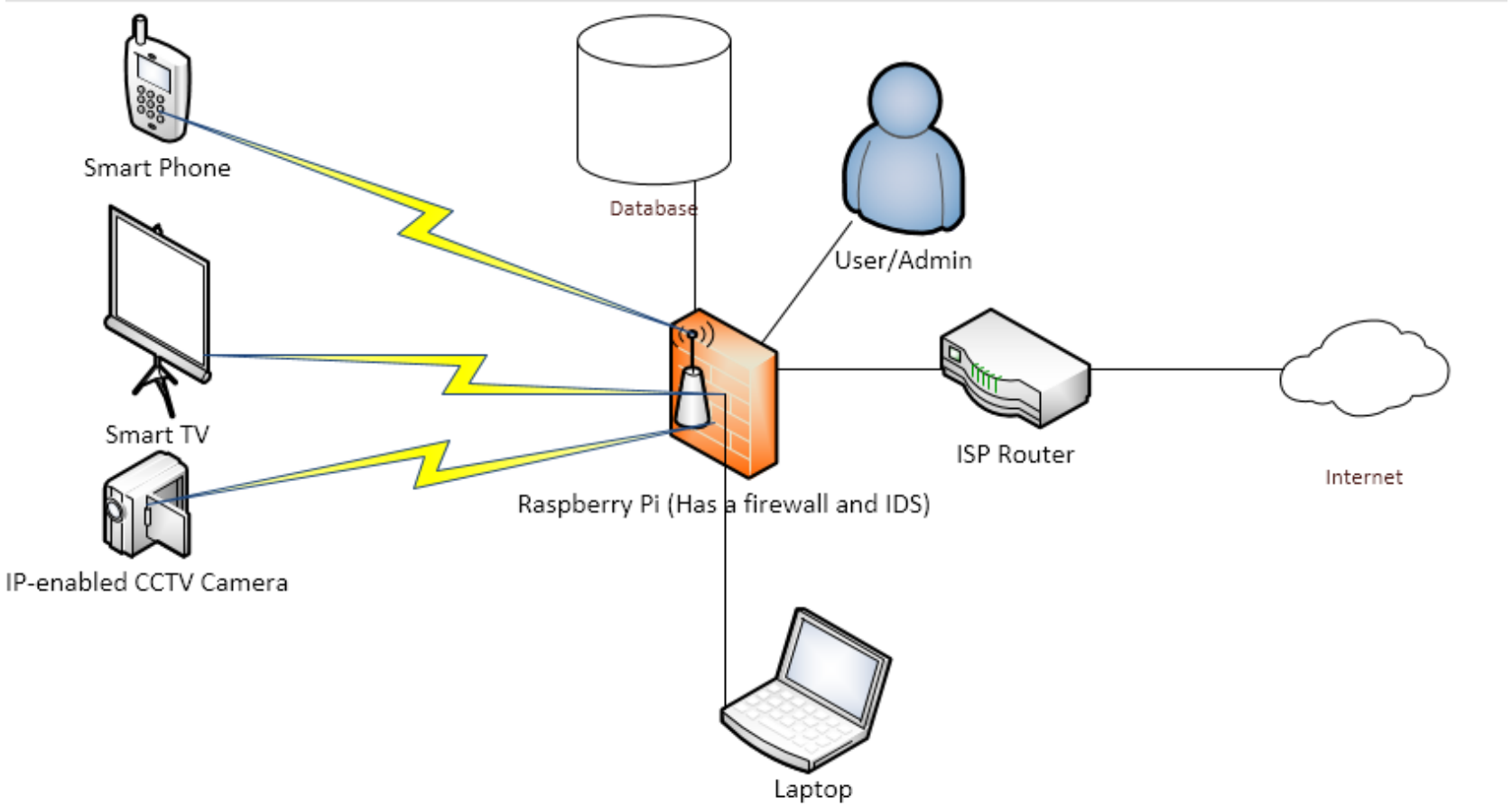
Why

- Fascinated by the Raspberry Pi
- Insecure Default Configurations on ISP Routers
- Proliferation of Internet Connected Devices in Home Networks (IoT)
- IoT Devices being used as bots – Mirai Botnet
- Inability to deploy security software on IoT Devices
- Expensive Home Internet Security Solutions
- I couldn't change my Zuku Router admin Password – Lol! 😇

The Set Up

- Raspberry Pi 1 Model B+
- 16GB+ Class 10 Micro SD Card
- Case
- Micro USB Power Cord
- A Ralink RT5370 USB Wireless Adaptor
- An Ethernet Cable

System Architecture

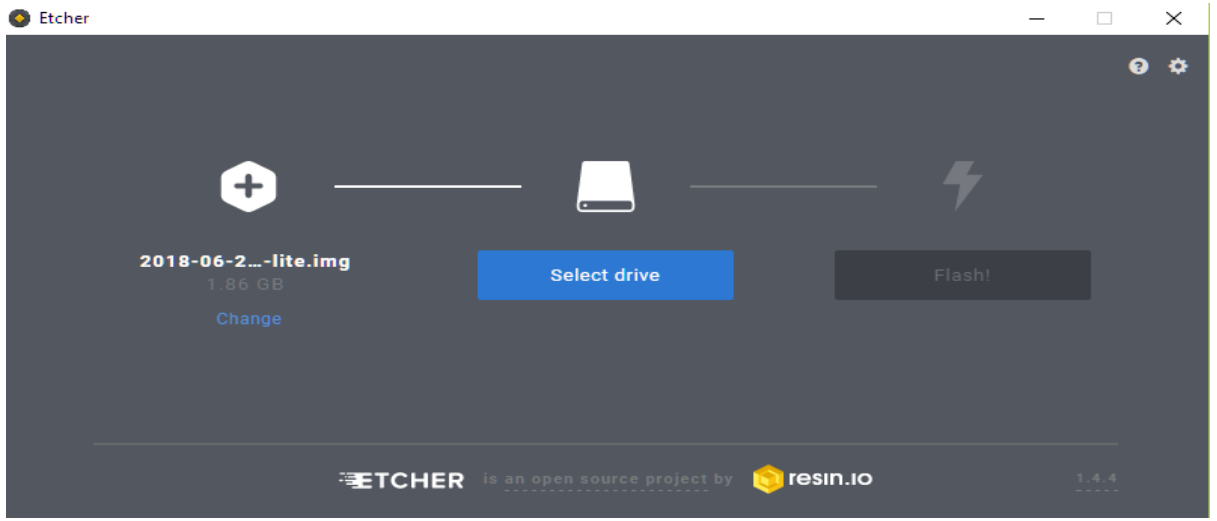


Install the OS

- Raspbian Stretch Lite

Alternatives:

- NOOBS
- Raspbian Stretch with desktop and recommended software
- Raspbian Stretch with desktop
- Windows 10 IoT Core – Good luck with this
- Ubuntu Mate
- Kali Linux ARM Image for RPi



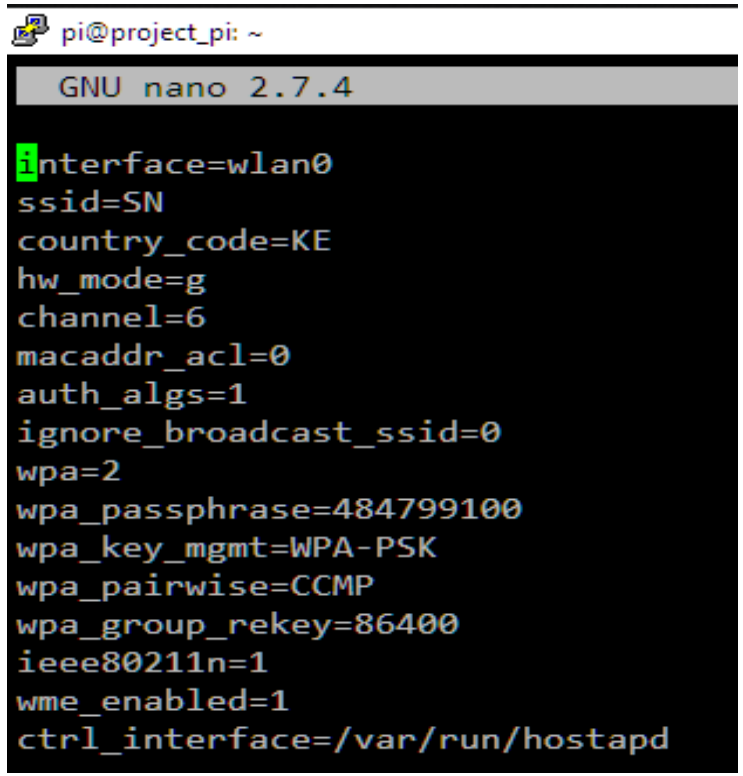
Wireless Gateway

- Host Access Point Service (hostapd) - *$Sudo apt-get install hostapd*
- isc-dhcp server - *$sudo apt-get install isc-dhcp-server*
- Network Interfaces

```
pi@project_pi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.18  netmask 255.255.255.0  broadcast 192.168.0.255
        ether b8:27:eb:67:ff:85  txqueuelen 1000  (Ethernet)
        RX packets 859  bytes 59361 (57.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 454  bytes 44519 (43.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1122  bytes 356903 (348.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1122  bytes 356903 (348.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.42.1  netmask 255.255.255.0  broadcast 192.168.42.255
        ether 00:13:ef:b0:00:37  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2  bytes 284 (284.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Wireless Gateway

- Wireless Access Point Configuration - `$sudo nano /etc/hostapd/hostapd.conf`



NAT Configuration

$Sudo sh -c "echo net.ipv4.ip_forward=1 >> /etc/sysctl.conf"

$sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"

$sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

*$sudo iptables -A FORWARD -i **eth0** -o **wlan0** -m state -state RELATED,ESTABLISHED -j **ACCEPT***

*$sudo iptables -A FORWARD -i **wlan0** -o **eth0** -j **ACCEPT***

$sudo sh -c "iptables-save > /etc/iptables/rules.v4"

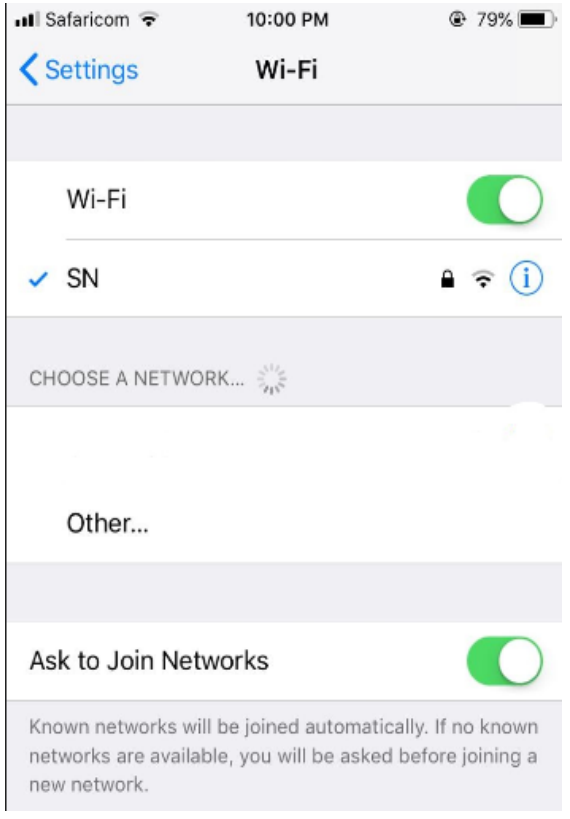
Wireless Gateway

- DHCP Server Configuration

```
subnet 192.168.42.0 netmask 255.255.255.0 {
        range 192.168.42.10 192.168.42.50;
        option broadcast-address 192.168.42.255;
        option routers 192.168.42.1;
        default-lease-time 600;
        max-lease-time 7200;
        option domain-name "local";
        option domain-name-servers 208.67.222.222, 208.67.220.220;
}
```

- Interface Configuration

```
pi@project_pi: ~
  GNU nano 2.7.4

# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd
# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto lo
iface lo inet loopback
up route add -net 10.0.0.0/8 gw 127.0.0.1 metric 200
up route add -net 172.16.0.0/12 gw 127.0.0.1 metric 200
up route add -net 192.168.0.0/12 gw 127.0.0.1 metric 200
up route add -net 224.0.0.0/4 gw 127.0.0.1 metric 200

auto eth0
iface eth0 inet dhcp

#wlan0 - internal network
auto wlan0
allow-hotplug wlan0
iface wlan0 inet static
  address 192.168.42.1
  network 192.168.42.0
  netmask 255.255.255.0
```



Iptables Firewall

NAT Section

- `$sudo touch iptables.rules`

pi@project_pi: ~/final_year_project

```
  GNU nano 2.7.4                                  File: iptables.test.rules

##############################################################################
# Author: Abdirahman Hish Mohamed
# Linux Firewall Ruleset (IPTABLES)
# (C) October 2018
##############################################################################

*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]

#Translate all packets using inside interface address (wlan0) to eth0 interface address
-A POSTROUTING -o eth0 -j MASQUERADE

COMMIT
```

Iptables Firewall

Filtering Section

pi@project_pi: ~/final_year_project

```
 GNU nano 2.7.4                                        File: iptables.test.rules

*filter

:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]
:LOG_DROPS - [0:0]

#############################################
## INPUT CHAIN
#############################################

# DROP RFC 1918 addresses on eth0 to prevent against SPOOFING
# Uncomment the below rules ONLY if your eth0 interface has a public IP address
#-A INPUT -i eth0 -s 0.0.0.0/8 -j LOG_DROPS
#-A INPUT -i eth0 -s 127.0.0.0/8 -j LOG_DROPS
#-A INPUT -i eth0 -s 10.0.0.0/8 -j LOG_DROPS
#-A INPUT -i eth0 -s 192.168.0.0/16 -j LOG_DROPS
#-A INPUT -i eth0 -s 172.16.0.0/12 -j LOG_DROPS
#-A INPUT -i eth0 -s 169.254.0.0/16 -j LOG_DROPS
#-A INPUT -i eth0 -s 224.0.0.0/24 -j LOG_DROPS

# DROP IP FRAGMENTS
-A INPUT -f -j LOG_DROPS

#DROP BAD TCP/UDP COMBINATIONS
-A INPUT -p tcp --dport 0 -j LOG_DROPS
-A INPUT -p udp --dport 0 -j LOG_DROPS
-A INPUT -p tcp --tcp-flags ALL NONE -j LOG_DROPS
-A INPUT -p tcp --tcp-flags ALL ALL -j LOG_DROPS
```

Iptables Firewall

Filtering Section – INPUT Chain

```
#PASS EVERYTHING ON THE LOOPBACK INTERFACE
-A INPUT -i lo -j ACCEPT

#ALLOWED SERVICES *UNCOMMENT BELOW ONLY IF YOUR DEFAULT POLICY CHAINS IS SET TO DROP*

#ALLOW ACCESS TO THE R-PI Web Server
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT

#ALLOW ACCESS TO THE NETWORK DASHBOARD
-A INPUT -p tcp --dport 19999 -j ACCEPT

#DNS
-A INPUT -p udp --dport 53 -j ACCEPT
-A INPUT -p tcp --dport 53 -j ACCEPT

#SSH
-A INPUT -p tcp --dport 22 -j ACCEPT
-A INPUT -p icmp -j ACCEPT

#DNS, DHCP, SSH, ntp, icmp-echo, squid web proxy etc
-A INPUT ! -i eth0 -p udp --dport 53 -j ACCEPT
-A INPUT ! -i eth0 -p tcp --dport 53 -j ACCEPT
-A INPUT ! -i eth0 -p udp --dport 67 -j ACCEPT
-A INPUT ! -i eth0 -p tcp --dport 22 -j ACCEPT
-A INPUT ! -i eth0 -p udp --dport 123 -j ACCEPT
-A INPUT ! -i eth0 -p tcp --dport 3128 -j ACCEPT
-A INPUT ! -i eth0 -p icmp -j ACCEPT

#Final Input - Realted and Drop
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

-A INPUT -j LOG_DROPS
```

Iptables Firewall

Filtering Section – OUTPUT Section

pi@project_pi: ~/final_year_project

```
  GNU nano 2.7.4                                  File: iptables.test.rules

##Outbound initiated by the Raspberry Pi to the internet
-A OUTPUT -o l0 -j ACCEPT

-A OUTPUT -p tcp --dport 22 -j ACCEPT
-A OUTPUT -p tcp --dport 25 -j ACCEPT
-A OUTPUT -p tcp --dport 53 -j ACCEPT
-A OUTPUT -p udp --dport 53 -j ACCEPT
-A OUTPUT -o eth0 -p tcp --dport 443 -j ACCEPT
-A OUTPUT -o eth0 -p tcp --dport 80 -j ACCEPT
-A OUTPUT -o eth0 -p udp --dport 123 -j ACCEPT
-A OUTPUT -o eth0 -p tcp --dport 43 -j ACCEPT
-A OUTPUT -o eth0 -p udp --dport 500 -j ACCEPT
-A OUTPUT -o eth0 -p udp --dport 4500 -j ACCEPT
-A OUTPUT -o eth0 -p tcp --dport 7547 -j ACCEPT
-A OUTPUT -o eth0 -p tcp -j ACCEPT

# Any outbound tcp, udp, and icmp-echo to the inside network
-A OUTPUT ! -o eth0 -p udp --dport 68 -j ACCEPT
-A OUTPUT -p icmp -j ACCEPT

#Final Output - Related and Drop
-A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -j LOG_DROPS
```

Iptables Firewall

Filtering Section – FORWARD Chain

```
##########################################
##FORWARD CHAIN
##########################################

# Block access to mobile.twitter.com for test purposes
-I FORWARD -s mobile.twitter.com -j LOG_DROPS
-I FORWARD -d mobile.twitter.com -j LOG_DROPS
-I FORWARD -d www.facebook.com -j LOG_DROPS

# Forward all other traffic from wlan0 to eth0 and vice versa
-A FORWARD -i wlan0 -o eth0 -j ACCEPT
-A FORWARD -i eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT

-A FORWARD ! -i eth0 -s 10.0.0.0/8 -p tcp -j ACCEPT
-A FORWARD ! -i eth0 -s 10.0.0.0/8 -p udp -j ACCEPT
-A FORWARD ! -i eth0 -s 10.0.0.0/8 -p icmp -j ACCEPT
-A FORWARD ! -i eth0 -s 192.168.0.0/16 -p tcp -j ACCEPT
-A FORWARD ! -i eth0 -s 192.168.0.0/16 -p tcp -j ACCEPT
-A FORWARD ! -i eth0 -s 192.168.0.0/16 -p tcp -j ACCEPT
-A FORWARD ! -i eth0 -s 172.16.0.0/12 -p tcp -j ACCEPT
-A FORWARD ! -i eth0 -s 172.16.0.0/12 -p tcp -j ACCEPT
-A FORWARD ! -i eth0 -s 172.16.0.0/12 -p tcp -j ACCEPT

-A FORWARD -j LOG_DROPS
```

Snort IDS/IPS

Overview

- Open source IDS/IPS
- Created by Martin Roesch in 1998 – Source Fire
- Recently acquired by Cisco
- Real-time traffic analysis and packet logging on IP networks
- Can detect buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, etc.

Snort IDS

Installation

Step 1: Install dependencies

```
$sudo apt-get install libpcap-dev libpcre3-dev
libdumbnet-dev bison flex -y
```

Step 2: Install Data Acquisition (DAQ) Library

```
$wget https://www.snort.org/downloads/snort/daq-
2.0.6.tar.gz
$tar xvzf daq-2.0.6.tar.gz
$cd daq-2.0.6
$./configure && make && sudo make install
```

Snort IDS

Installation

Step 3: Install Snort from Source

```
$wget https://www.snort.org/downloads/snort/snort-2.9.12.tar.gz

$tar xvzf snort-2.9.12.tar.gz

$cd snort-2.9.12

$./configure --enable-sourcefire

$make

$sudo make install
```

This takes a while…….

Snort IDS

Installation

Step 3: Create Snort User/Group, Directories & Files

```
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /etc/snort
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /etc/snort/rules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /etc/snort/rules/iplists
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /etc/snort/preproc_rules
```

```
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /usr/local/lib/snort_dynamicrules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /etc/snort/so_rules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /var/log/snort
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo mkdir /var/log/snort/archived_logs
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo touch /etc/snort/rules/iplists/black_list.rules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo touch /etc/snort/rules/iplists/white_list.rules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo touch /etc/snort/rules/local.rules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo touch /etc/snort/sid-msg.map
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chmod -R 5775 /etc/snort
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chmod -R 5775 /var/log/snort
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chmod -R 5775 /var/log/snort/archived_logs
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chmod -R 5775 /etc/snort/so_rules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chown -R snort:snort /etc/snort
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chown -R snort:snort /var/log/snort
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $ sudo chown -R snort:snort /usr/local/lib/snort_dynamicrules
pi@project_pi:~/sourcecode/snort_src/snort-2.9.12 $
```

Snort IDS

Installation

Step 3: Copy configuration files from Snort source directory to the newly created /etc/snort directory

```
$sudo cp *.conf* /etc/snort
$sudo cp *.map /etc/snort
$sudo cp *.dtd /etc/snort
$cd ~/source/snort-2.9.1.2/src/dynamic-preprocessors/build/usr/local/lib/snort_dynamicpreprocessor/
$sudo cp * /usr/local/lib/snort_dynamicpreprocessor
```

Snort IDS

Installation

Step 4: Customize Snort Configuration

- Comment out all rulesets in `snort.conf`

```
$sudo sed -i "s/include \$RULE\_PATH/#include \$RULE\_PATH/"
/etc/snort/snort.conf
```

```
$sudo nano -c /etc/snort/snort.conf
```

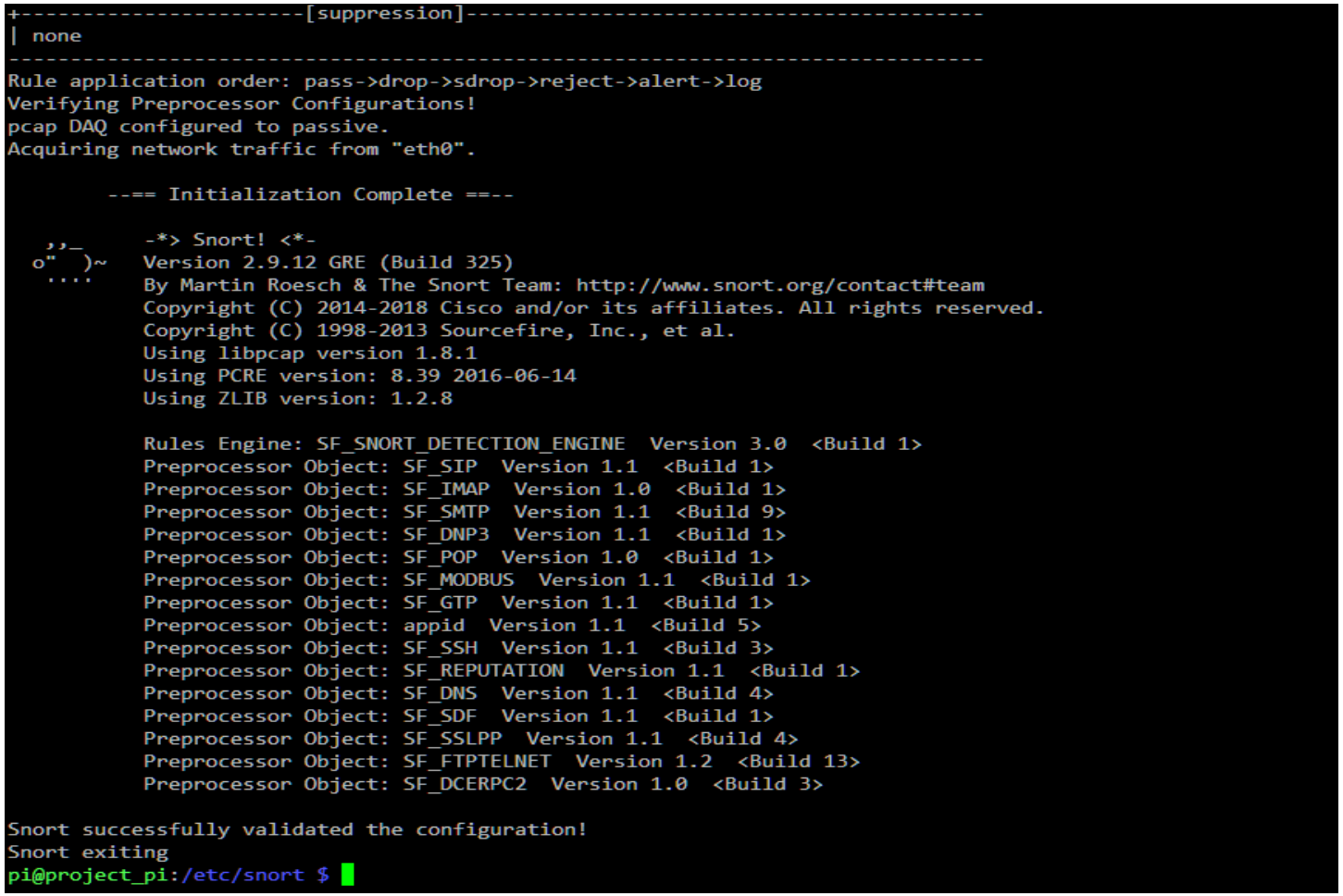
- Line 45 – `ipvar HOME_NET 192.168.1.0/24`
- Line 104-106
  - `var RULE_PATH /etc/snort/rules`
  - `var SO_RULE_PATH /etc/snort/so_rules`
  - `var PREPROC_RULE_PATH /etc/snort/preproc_rules`
- Line 113-114
  - `var WHITE_LIST_PATH /etc/snort/rules/iplists`
  - `var BLACK_LIST_PATH /etc/snort/rules/iplists`
- Uncomment Line 546 - `include $RULE_PATH/local.rules`

Snort IDS

Installation

Step 4: Test for Successful Installation

```
$ sudo snort -T -i eth0 -c /etc/snort/snort.conf
```



Snort IDS

Additional Components Installation

Barnyard2

- Takes care of Snort binary output files and converts them into human readable data and saves it to a database
- https://github.com/firnsy/barnyard2

PulledPork

- Pulledpork is a script that manages the Snort rules.
- It can be used for automated downloads, parsing and modifications for all the Snort rulesets.
- https://github.com/shirkdog/pulledpork

Deployment

Options

- Gateway
- SPAN/Mirror Port
- In-line Mode

Network Visibility

United States International University-Africa

-List of Connected Devices on the Network-

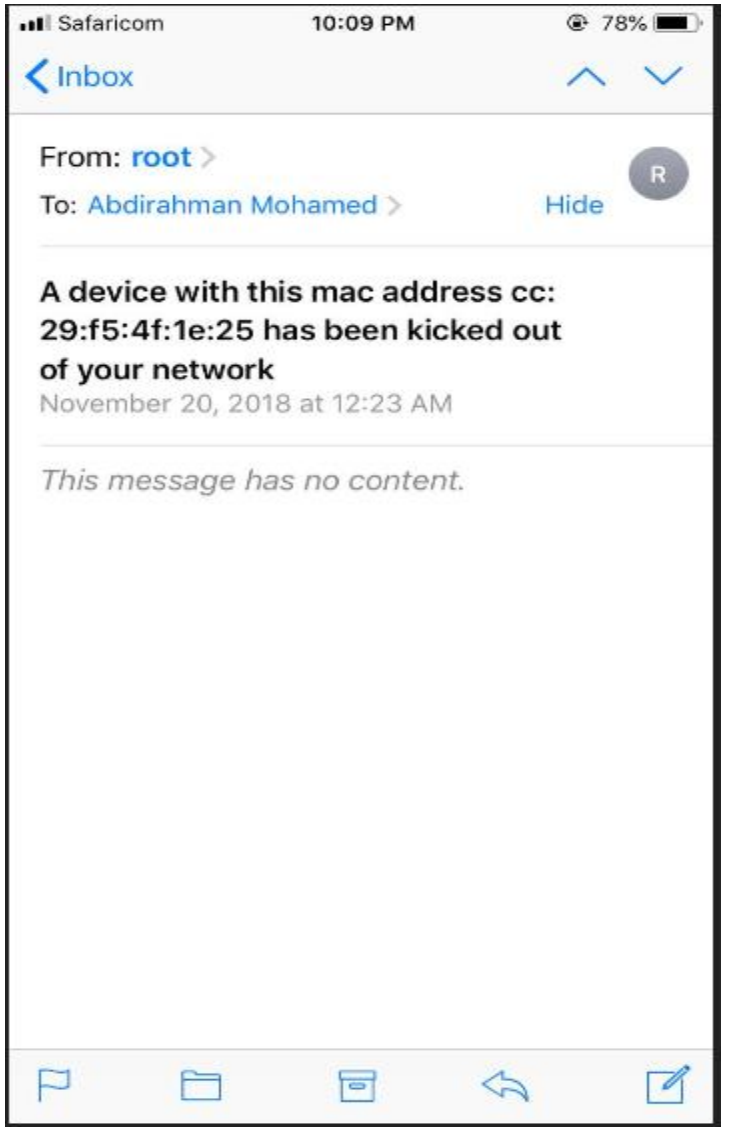
```
                                Reading leases from /var/lib/dhcp/dhcpd.leases
                                                Processing: 33% complete
  MAC               IP              hostname       valid until         manufacturer
================================================================================================
  cc:29:f5:4f:1e:25  192.168.42.10   AbdirahnsiPhon 2018-11-26 19:09:59 Apple, Inc.
```

-Network Access-

To revoke network access, enter MAC address of the device and click the revoke button.

Enter MAC Address of the Device

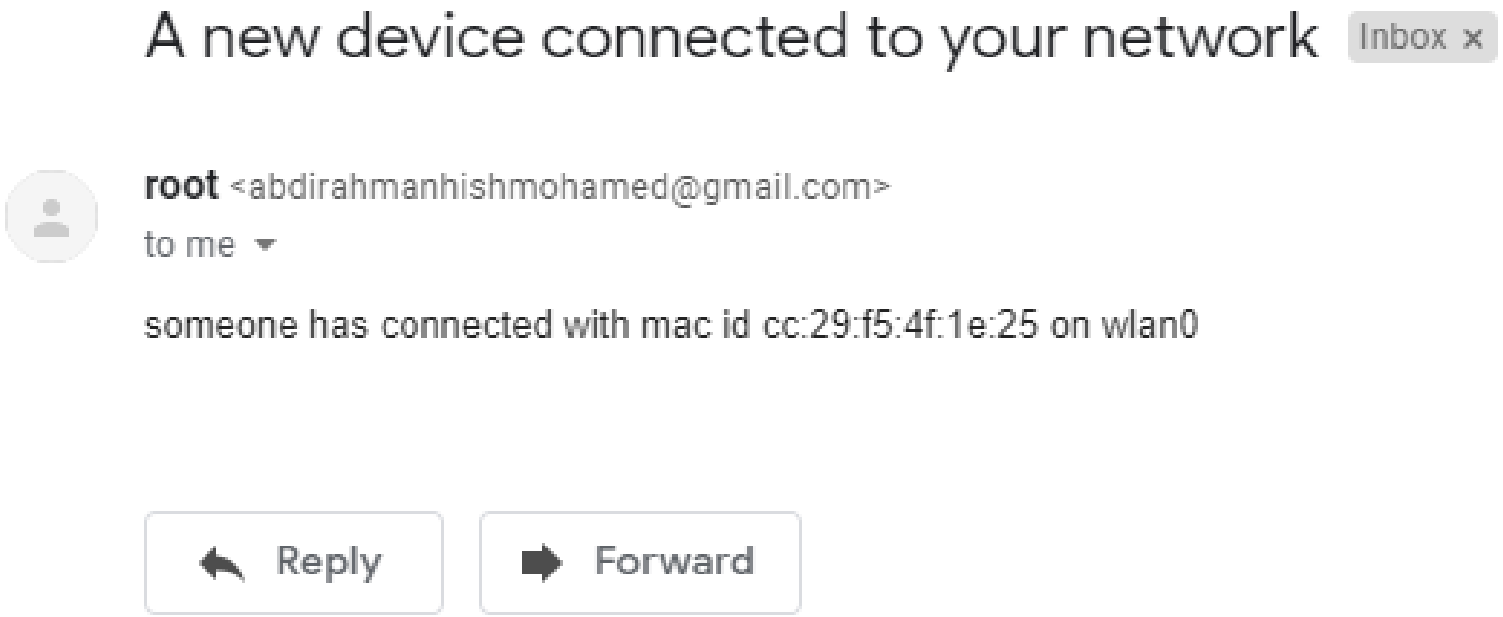
Revoke Access



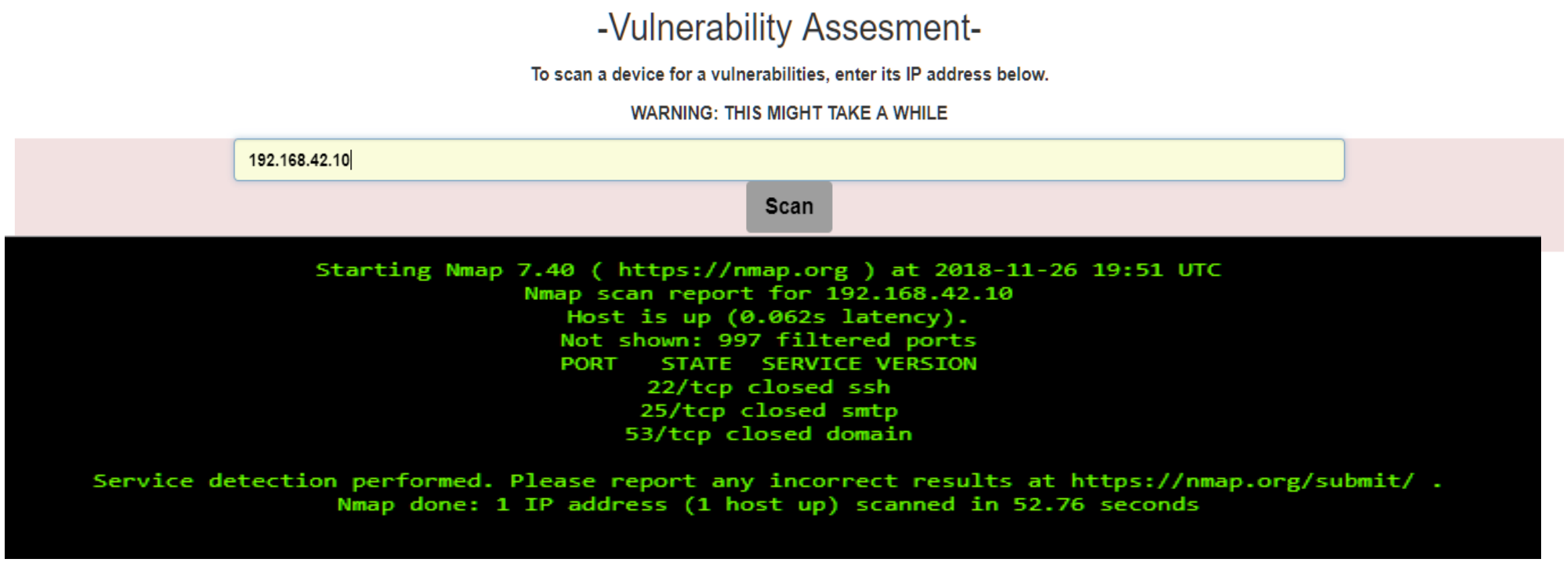
Sentinel Africa
Risk Management Consultants

ISO/IEC 27001 CERTIFIED PECB

Network Visibility



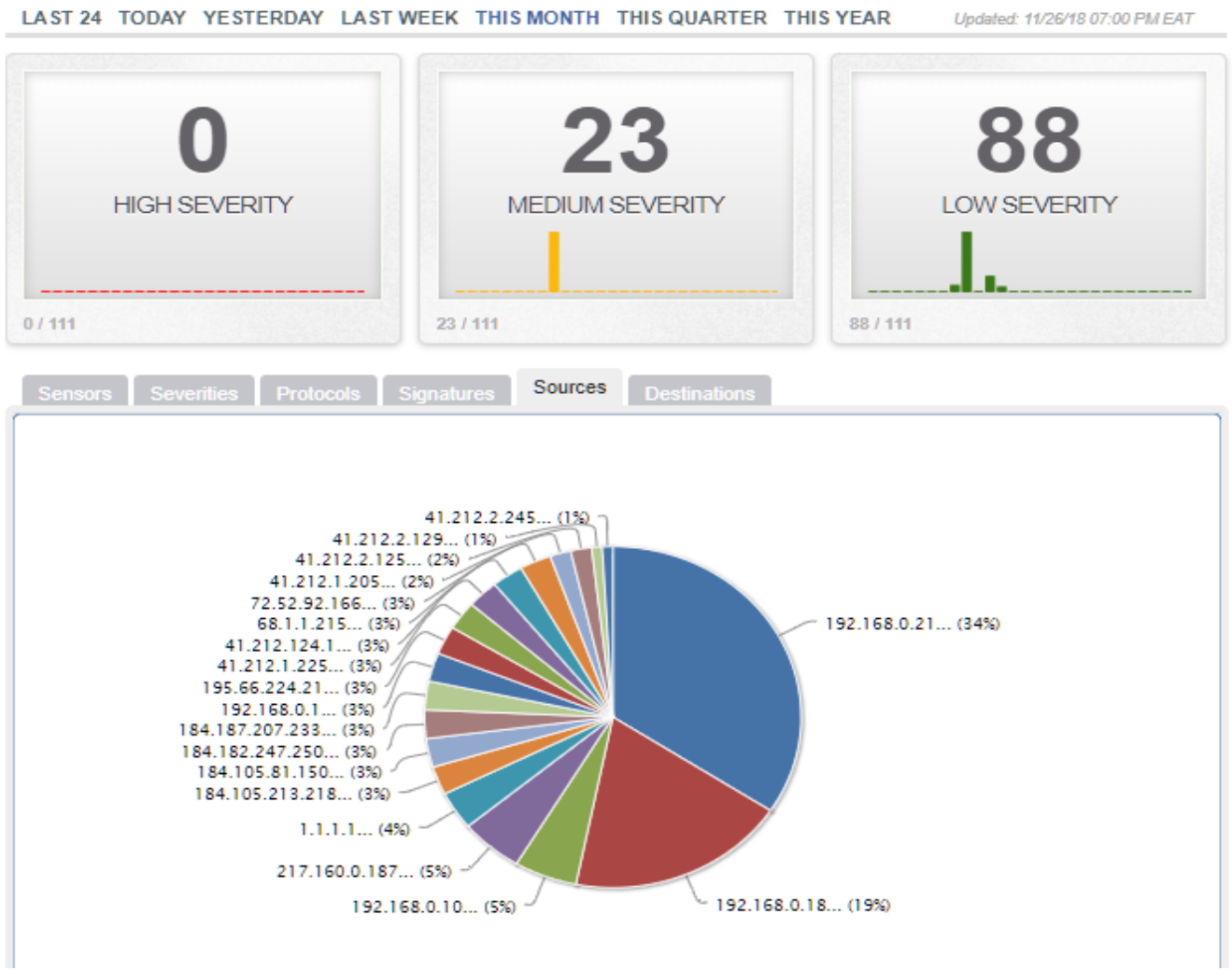
NMAP Vulnerability Scanner



Marc Ruef

https://github.com/scipag/vulscan

Snorby



TOP 5 SENSOR

project_pi:NULL	111

TOP 5 ACTIVE USERS

Administrator	0

LAST 5 UNIQUE EVENTS

Snort Alert [1:10000002:1]	88
Snort Alert [1:10000002:1]	88
Snort Alert [1:10000002:1]	88
Snort Alert [1:10000002:1]	88
Snort Alert [1:10000002:1]	88

ANALYST CLASSIFIED EVENTS

Unauthorized Root Access	0
Unauthorized User Access	0
Attempted Unauthorized...	0
Denial of Service Attack	0
Policy Violation	0
Reconnaissance	0
Virus Infection	0
False Positive	0

United States International University-Africa

Sentinel Africa
Risk Management Consultants

ISO/IEC 27001 CERTIFIED PECB

Similar Commercial Products

United States International University-Africa

BitDefender Box

$179.99
VAT included

Trend Micro SHN

$109.95

ASUS RT- Series Routers (AiProtection)

RT-AC5300 – $399.99
RT-AC88U – $299.99
RT-AC3100 – $299.99
RT-AC3200 – $249.99
RT-AC68P – $199.99
RT-AC87U – $219.99
RT-AC68U – $199.99
RT-AC56U – $109.99

NetSec-Pi

~$50

Sentinel Africa
Risk Management Consultants

ISO/IEC 27001 CERTIFIED PECB

So What's Next Now?

- Raspberry Pi 3 Model B+
- Going offensive with Kali Linux for R-Pi
- Squid Web proxy for parental control
- Deploy the IDS in-line
- Security Onion

United States International University-Africa

Thank You

Sentinel Africa
Risk Management Consultants

ISO/IEC 27001 CERTIFIED PECB