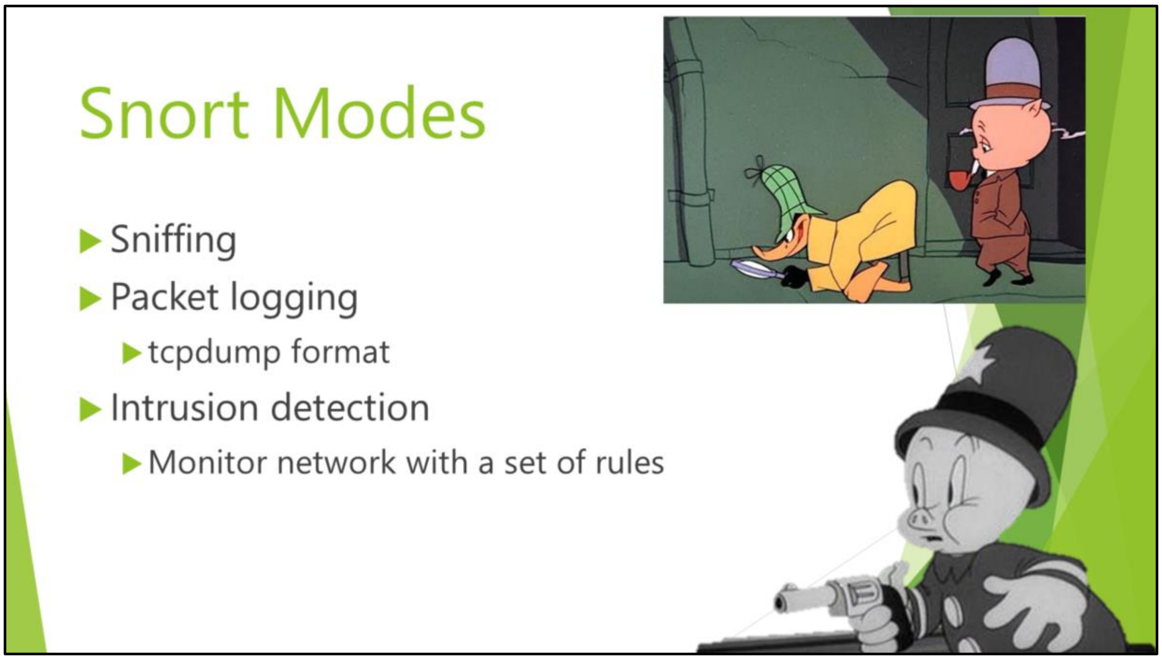
SNORT®

Abegail Jakop, Minh Le Hoang

What is Snort?

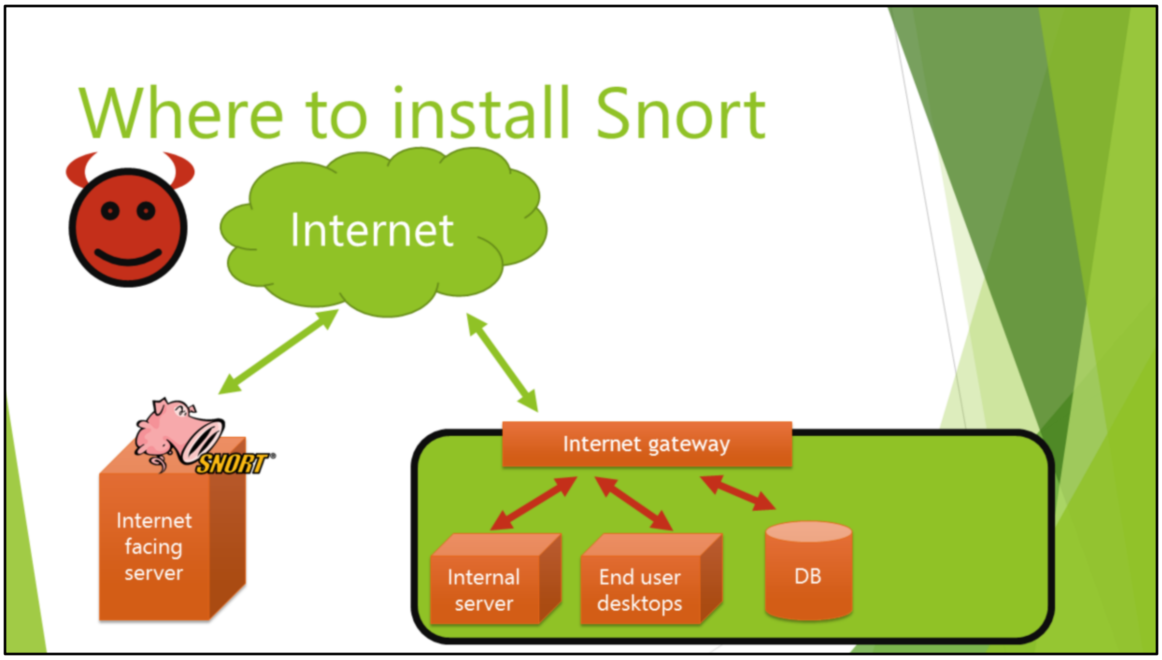
- Network intrusion detection and prevention
  - Attempts to detect intrusions
    - Protocol analysis
    - Content matching



Snort Modes

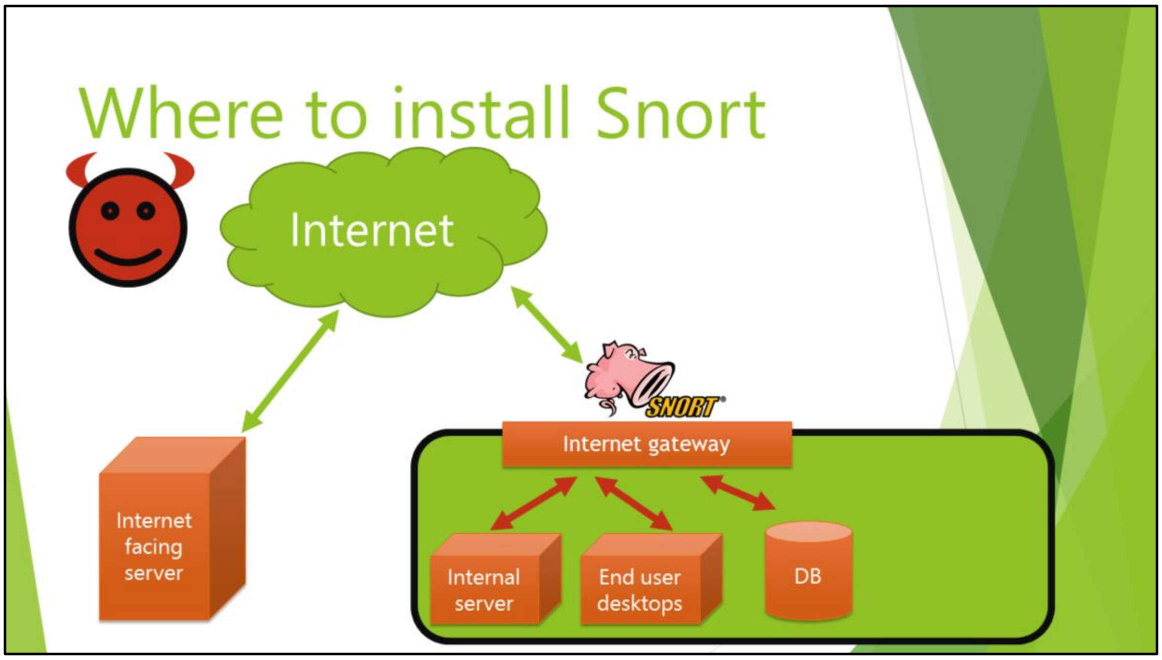
- Sniffing
- Packet logging
  - tcpdump format
- Intrusion detection
  - Monitor network with a set of rules

Sniffing mode outputs the seen packets onto the console. Not super useful.
Logging mode with do the same, just output it to a log file instead. It's also in the same format as tcpdump, so whatever tool you use to read tcpdump logs can be used to read snort logs.
Intrusion detection, probably most useful for the everyday user, since no one wants to read through massive logs containing every packet ever seen.



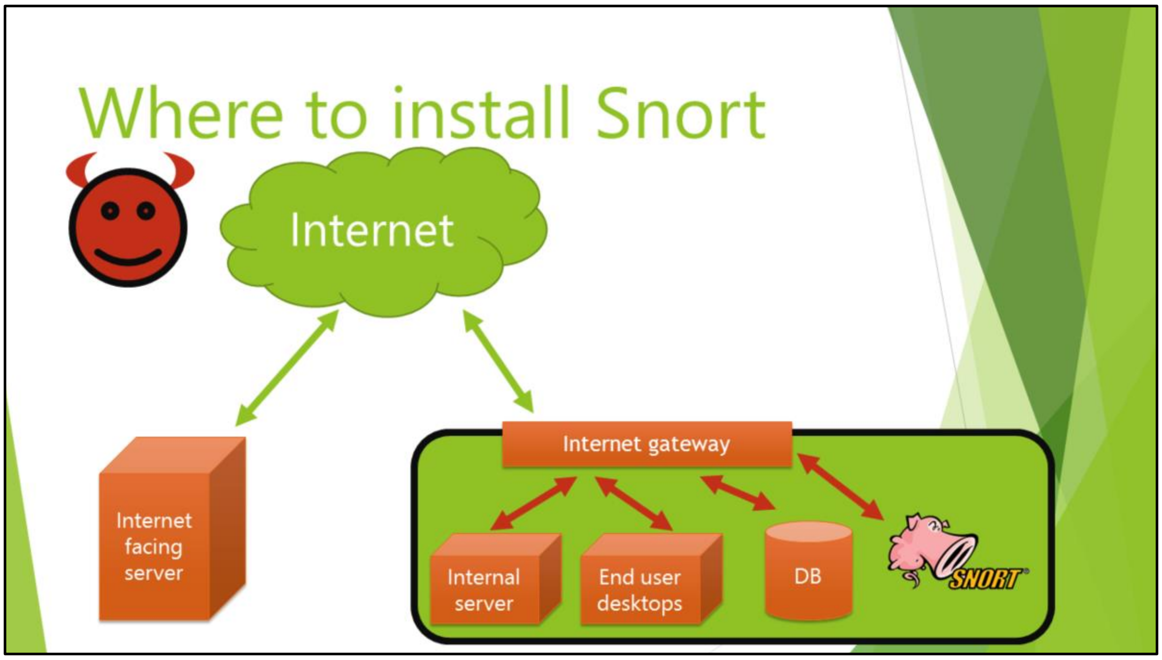
Where to install Snort

3 places to install snort. Where you install snort matters as it determines what packets snort is privy to.
This slide shows Snort on an internet facing server. Therefore, snort will monitor the incoming and outgoing traffic between the internet and the internet facing server.

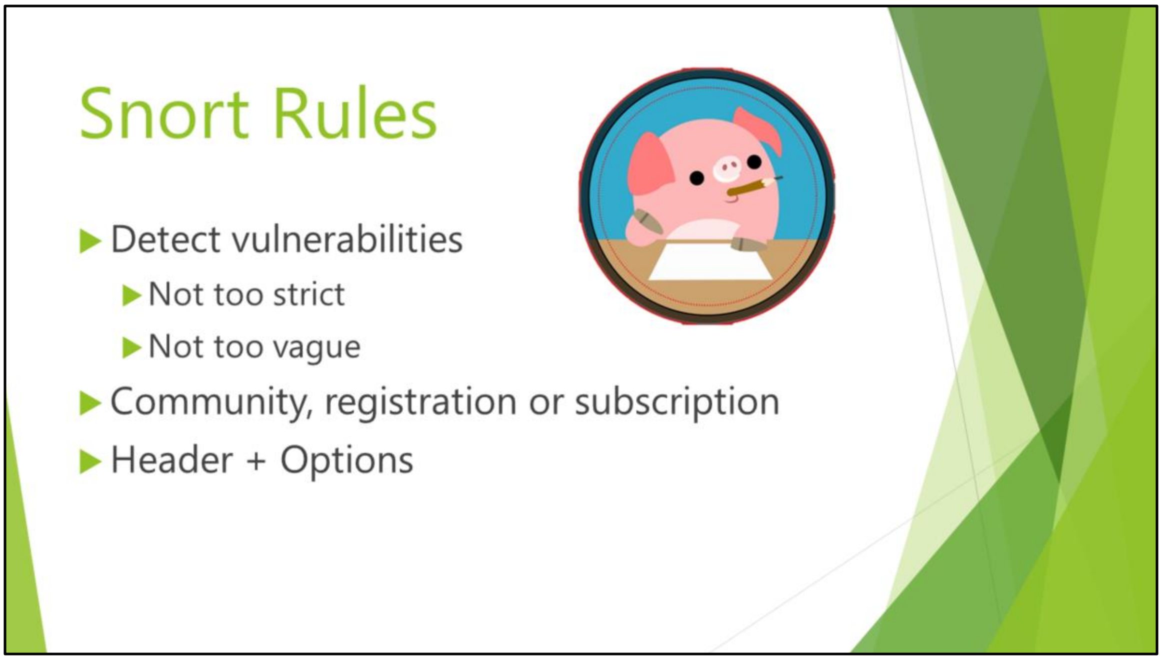


Where to install Snort

Another place to install Snort is on the boundary of an internet gateway. So now Snort monitors the traffic between the internet and a LAN, or you could see it as monitoring what passes between the gateway and the internet.



Lastly, Snort can be installed within a LAN, so it'll monitor the traffic between the different services in the LAN. Eg, monitor the traffic between the server and the DB to make sure nothing malicious is inputted into the DB.

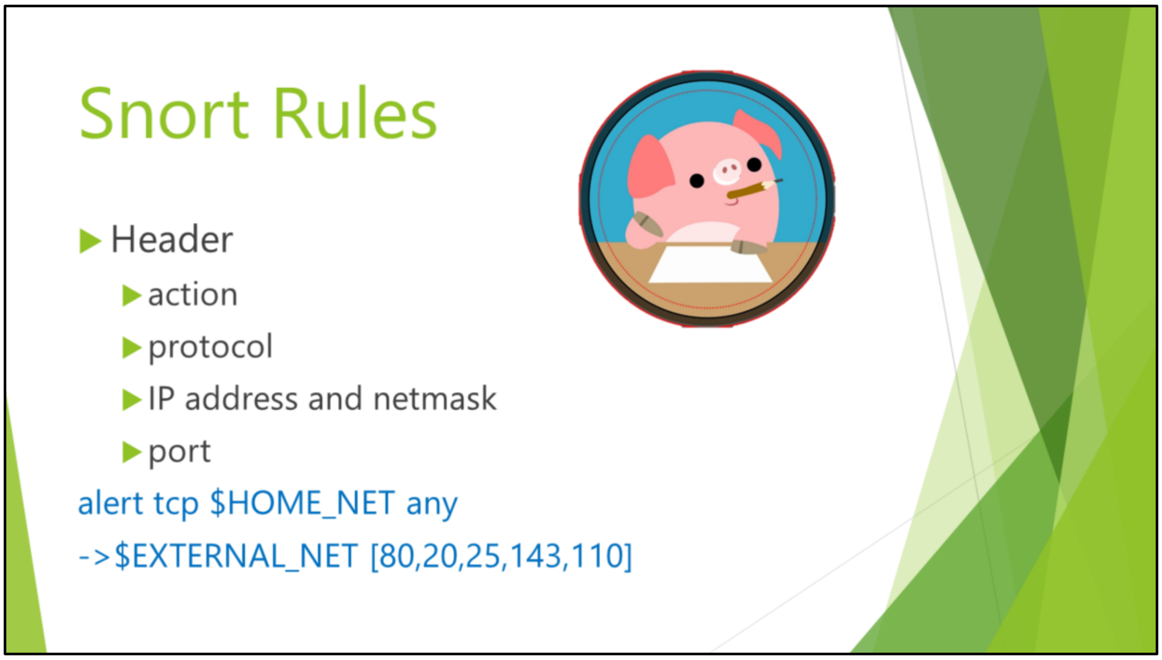


Snort Rules

- Detect vulnerabilities
  - Not too strict
  - Not too vague
- Community, registration or subscription
- Header + Options

Don't want the rules to be too strict, otherwise you might be missing some cases where a detail that was specified doesn't actually matter Don't wan the rules to be too vague otherwise you'll end up with false-positives, and if you've got add-ons, this can lead to negative consequences.

The
way you get your rules determines how early you get them. subscription is the earliest, then registration. from that, maybe the community will adapt a version?



Snort Rules

- Header
  - action
  - protocol
  - IP address and netmask
  - port

alert tcp $HOME_NET any
->$EXTERNAL_NET [80,20,25,143,110]

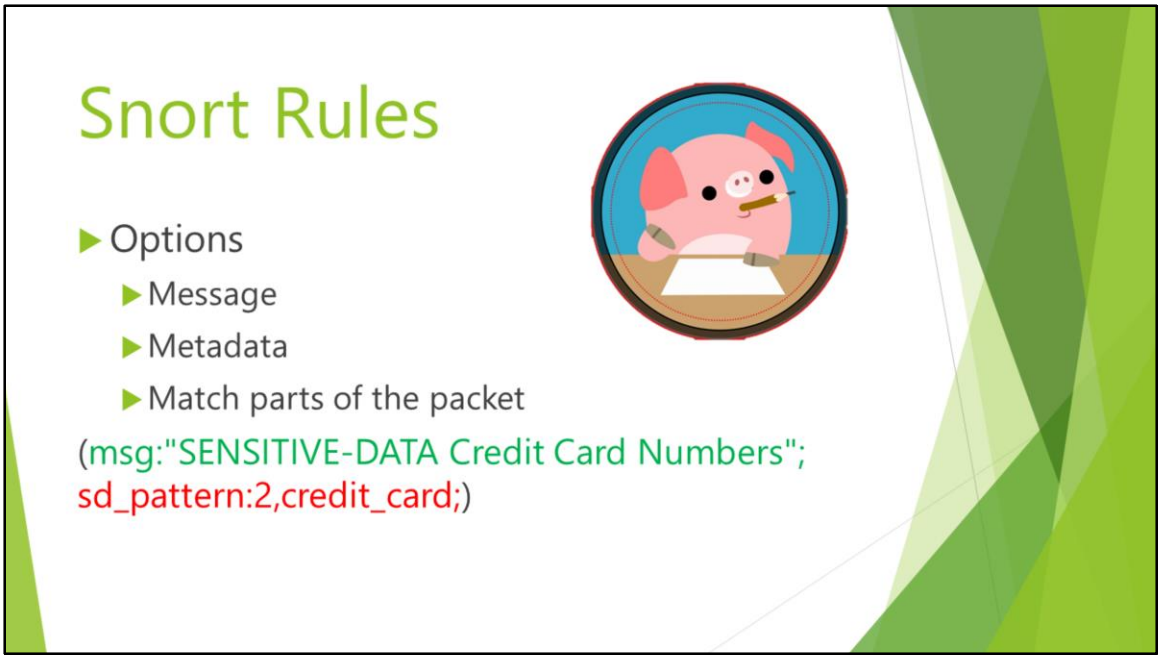
A basic primer on the structure of snort rules.

In the example header,
action: alert
protocol: tcp
source IP addr + netmask: $HOME_NET
source port: any
destination IP + netmask: $EXTERNAL_NET
destination port: [80,20,25,143,110] (side note: I don't recall what the services these ports correspond to. sorry)

the arrow (->) points from the source to the destination. (so you can also have it the other way around (<-), it just means that the source is now on the left of the arrow and the destination is on the right of the arrow) variables $HOME_NET and $EXTERNAL_NET are defined in a configuration file.

Cheat sheet: http://detroitdavesraves.blogspot.ca/2012/09/snort-rules-cheat-sheet.html

One thing not covered is the action:dynamic. This action allows you to hold off on logging/alerting/dropping a packet until after another rule is triggered. We'll go over this in tutorial.



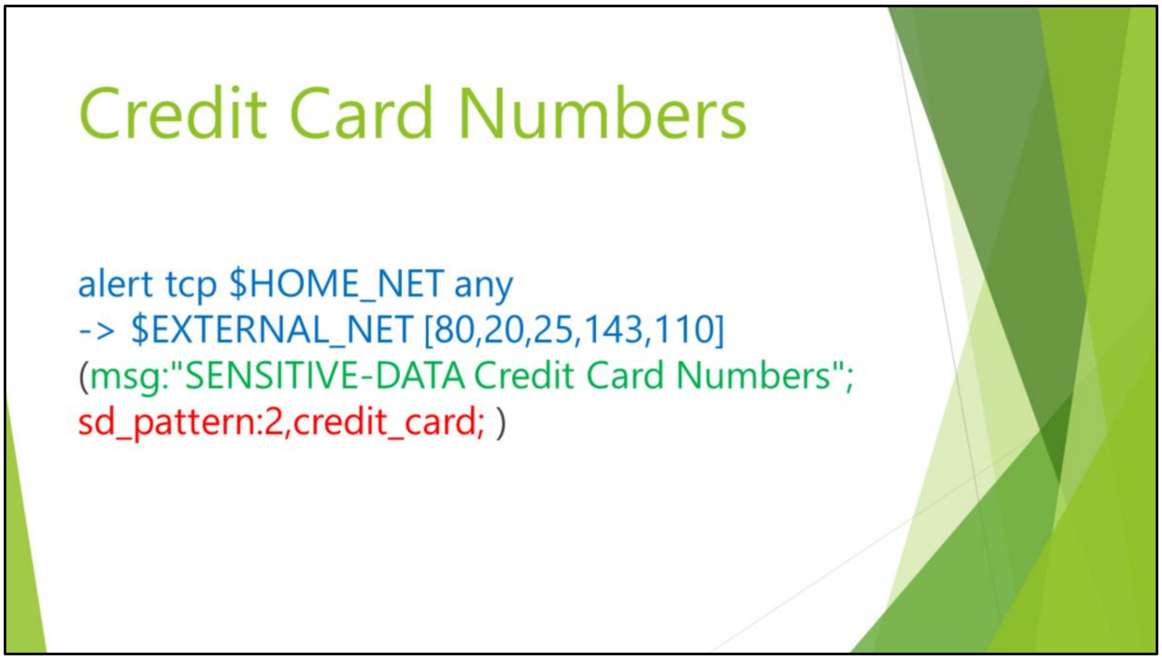
Snort Rules

- Options
  - Message
  - Metadata
  - Match parts of the packet

(msg:"SENSITIVE-DATA Credit Card Numbers";
sd_pattern:2,credit_card;)

options consist of a variety of different pattern matches, a message and maybe metadata. (actually, I think a message is optional as well, but typically you want want to supplement what gets dumped in the log)

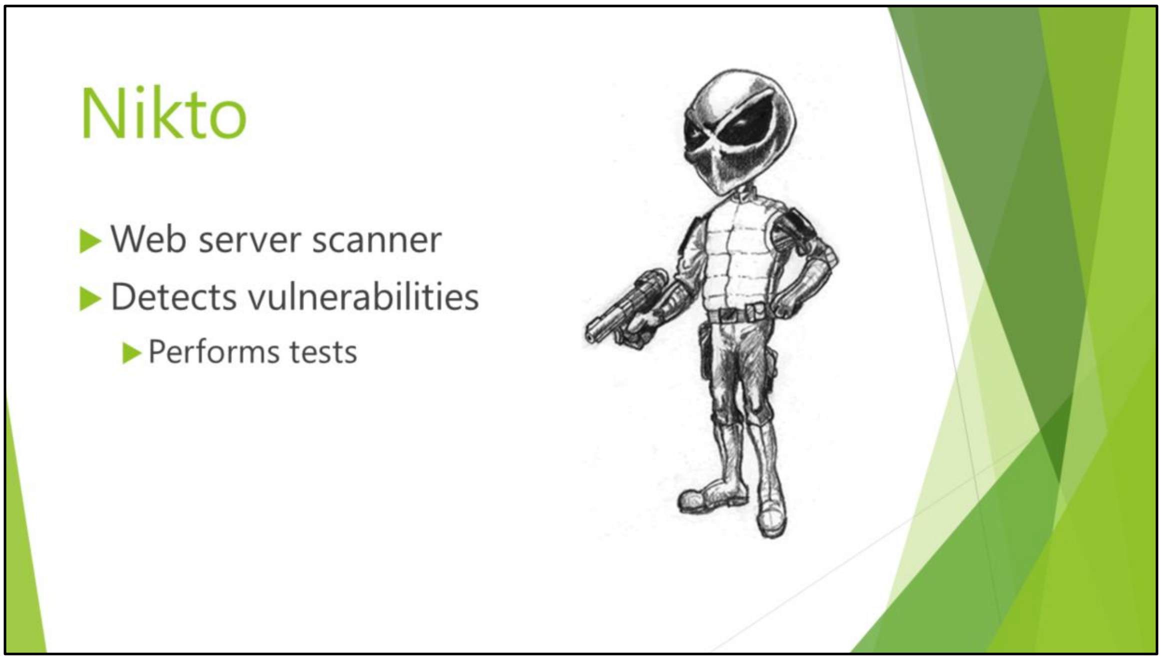
in the example,
green text is the message, red text refers to a pattern matching option trying to match 2 occurences in the payload to something that resembles a credit card number.



Credit Card Numbers

```
alert tcp $HOME_NET any
-> $EXTERNAL_NET [80,20,25,143,110]
(msg:"SENSITIVE-DATA Credit Card Numbers";
sd_pattern:2,credit_card; )
```

Putting the example together, snort will alert the use if there are any packets that contain two occurences of a credit card in the payload, where the packet is going from the home network to an external network of the listed ports.
So, trying to alert the user if it thinks that sensitive data, in this case unencrypted credit card numbers, are being transferred to an external network.

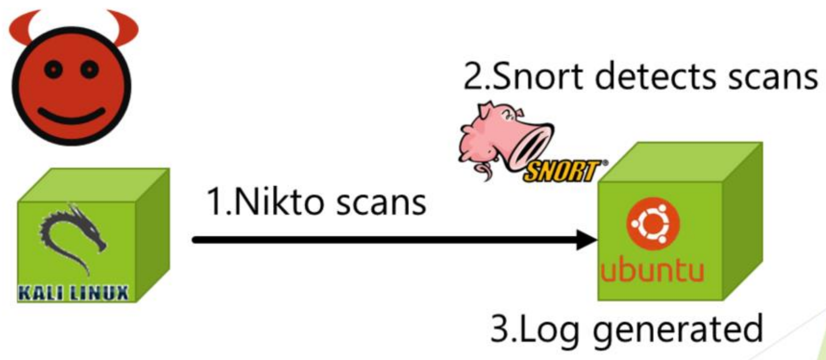


Nikto

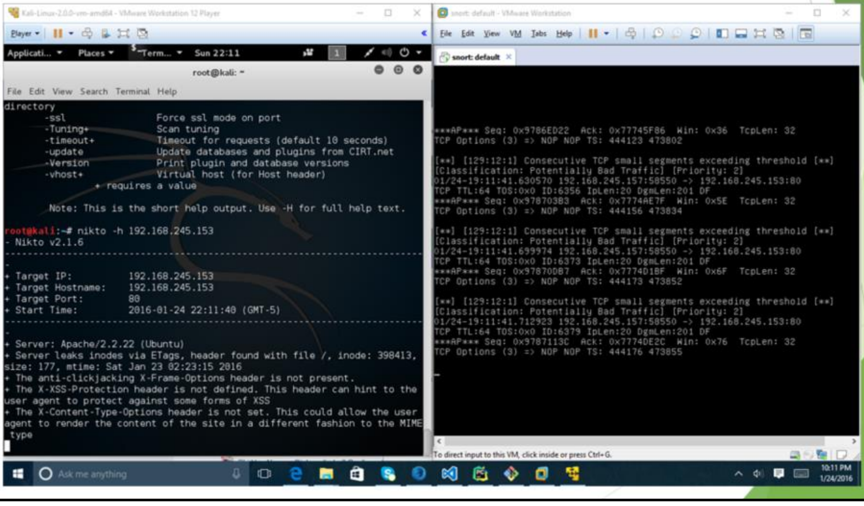
- Web server scanner
- Detects vulnerabilities
  - Performs tests

Can be used for malicious purposes . you can test target/victim's servers for vulnerabilities

Snort vs Nikto:Topology



Snort vs Nikto

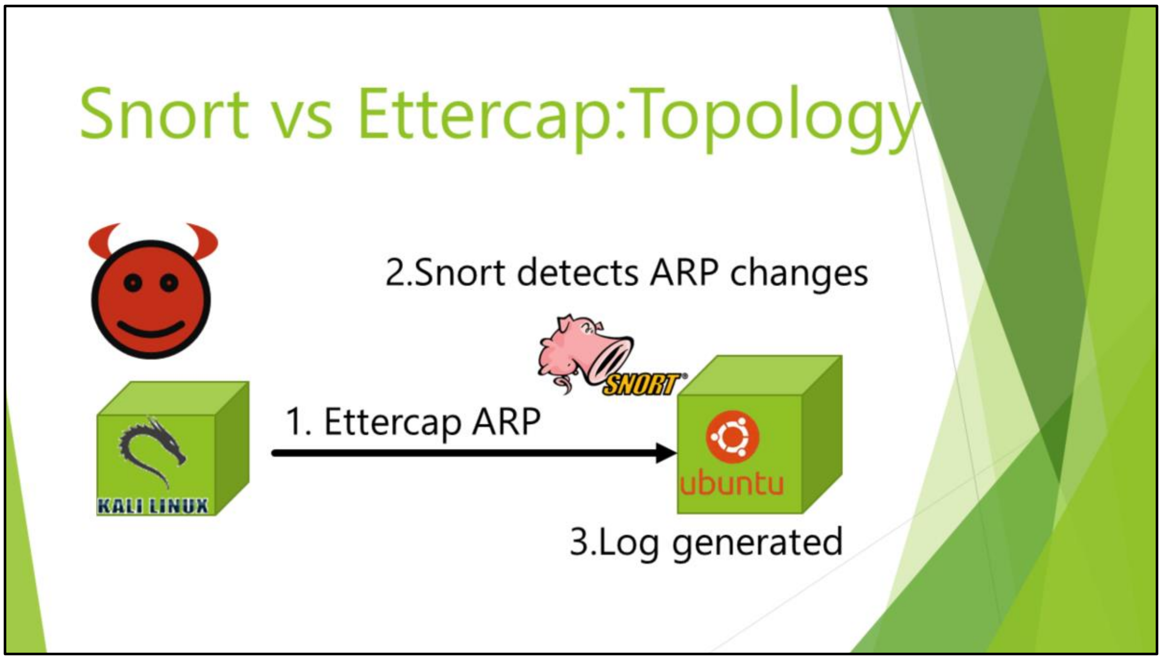

Kali-Linux-2.0.0-vm-amd64 - VMware Workstation 12 Player
Player
Applicati...
Places
Term...
Sun 22:11
root@kali: ~
File Edit View Search Terminal Help
snort: default - VMware Workstation
File Edit View VM Tabs Help
snort: default
To direct input to this VM, click inside or press Ctrl+G.
Ask me anything
10:11 PM
1/24/2016

```
directory
       -ssl                 Force ssl mode on port
       -Tuning+             Scan tuning
       -timeout+            Timeout for requests (default 10 seconds)
       -update              Update databases and plugins from CIRT.net
       -Version             Print plugin and database versions
       -vhost+              Virtual host (for Host header)
                + requires a value

        Note: This is the short help output. Use -H for full help text.

root@kali:~# nikto -h 192.168.245.153
- Nikto v2.1.6
---------------------------------------------------------------------------
-
+ Target IP:          192.168.245.153
+ Target Hostname:    192.168.245.153
+ Target Port:        80
+ Start Time:         2016-01-24 22:11:40 (GMT-5)
---------------------------------------------------------------------------
-
+ Server: Apache/2.2.22 (Ubuntu)
+ Server leaks inodes via ETags, header found with file /, inode: 398413,
size: 177, mtime: Sat Jan 23 02:23:15 2016
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the
user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user
agent to render the content of the site in a different fashion to the MIME
 type
```

```
***AP*** Seq: 0x9786ED22  Ack: 0x77745F86  Win: 0x36  TcpLen: 32
TCP Options (3) => NOP NOP TS: 444123 473802

[**] [129:12:1] Consecutive TCP small segments exceeding threshold [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
01/24-19:11:41.630570 192.168.245.157:58550 -> 192.168.245.153:80
TCP TTL:64 TOS:0x0 ID:6356 IpLen:20 DgmLen:201 DF
***AP*** Seq: 0x97870383  Ack: 0x7774AE7F  Win: 0x5E  TcpLen: 32
TCP Options (3) => NOP NOP TS: 444156 473834

[**] [129:12:1] Consecutive TCP small segments exceeding threshold [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
01/24-19:11:41.699974 192.168.245.157:58550 -> 192.168.245.153:80
TCP TTL:64 TOS:0x0 ID:6373 IpLen:20 DgmLen:201 DF
***AP*** Seq: 0x97870DB7  Ack: 0x7774D1BF  Win: 0x6F  TcpLen: 32
TCP Options (3) => NOP NOP TS: 444173 473852

[**] [129:12:1] Consecutive TCP small segments exceeding threshold [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
01/24-19:11:41.712923 192.168.245.157:58550 -> 192.168.245.153:80
TCP TTL:64 TOS:0x0 ID:6379 IpLen:20 DgmLen:201 DF
***AP*** Seq: 0x9787113C  Ack: 0x7774DE2C  Win: 0x76  TcpLen: 32
TCP Options (3) => NOP NOP TS: 444176 473855

_
```

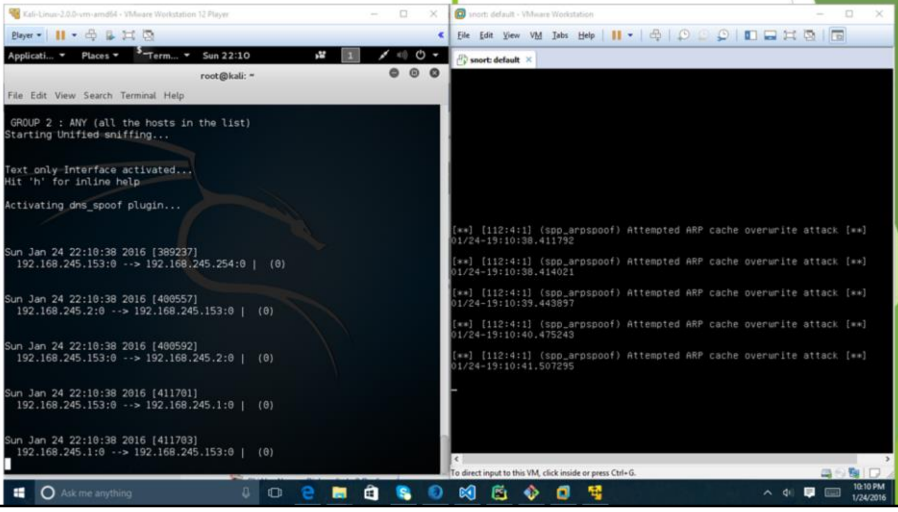
Ettercap

- Man-in-the-Middle attacks
- ARP poisoning the target



Snort vs Ettercap:Topology

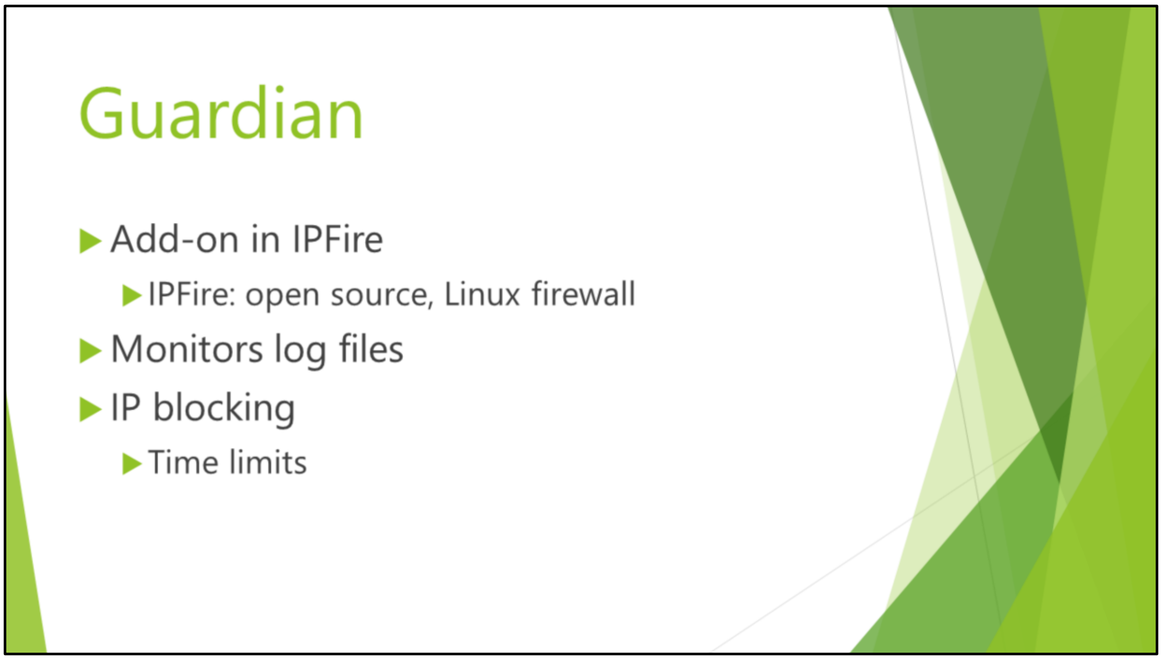
Snort vs Ettercap



```
 GROUP 2 : ANY (all the hosts in the list)
Starting Unified sniffing...


Text only Interface activated...
Hit 'h' for inline help

Activating dns_spoof plugin...



Sun Jan 24 22:10:38 2016 [389237]
  192.168.245.153:0 --> 192.168.245.254:0 |  (0)


Sun Jan 24 22:10:38 2016 [400557]
  192.168.245.2:0 --> 192.168.245.153:0 |  (0)


Sun Jan 24 22:10:38 2016 [400592]
  192.168.245.153:0 --> 192.168.245.2:0 |  (0)


Sun Jan 24 22:10:38 2016 [411701]
  192.168.245.153:0 --> 192.168.245.1:0 |  (0)


Sun Jan 24 22:10:38 2016 [411703]
  192.168.245.1:0 --> 192.168.245.153:0 |  (0)
```

```
[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
01/24-19:10:38.411792

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
01/24-19:10:38.414021

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
01/24-19:10:39.443897

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
01/24-19:10:40.475243

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
01/24-19:10:41.507295
```

quick reminder what ettercap is
How can snort detect ettercap?

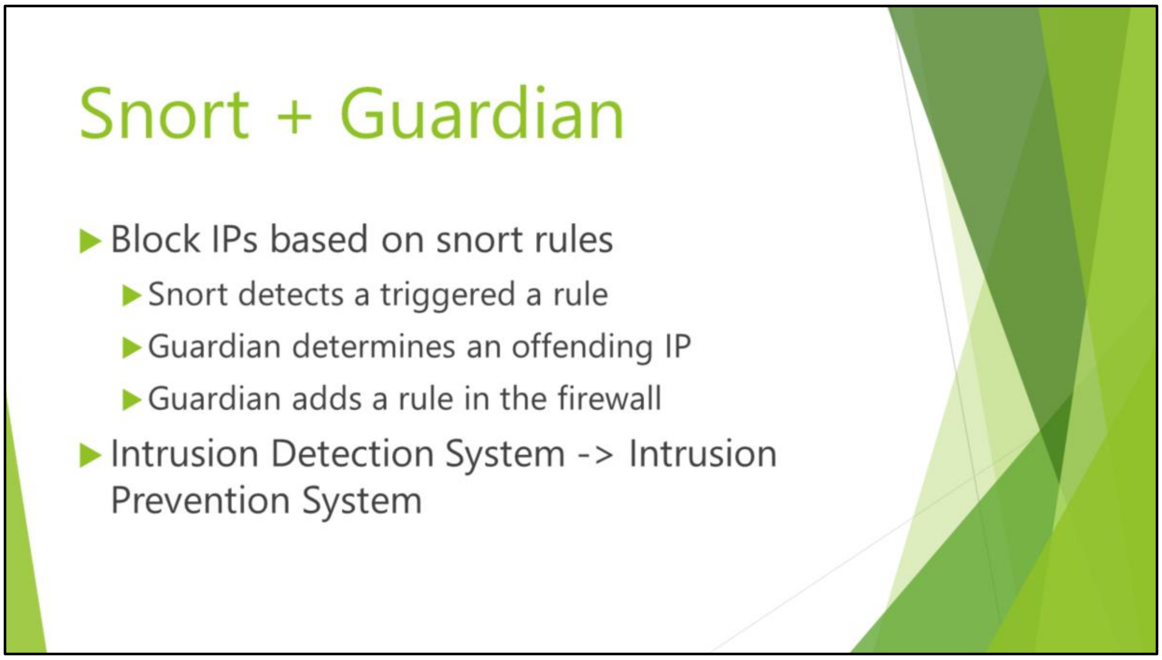


Guardian

- Add-on in IPFire
  - IPFire: open source, Linux firewall
- Monitors log files
- IP blocking
  - Time limits

Guardian's main job is to read though (specified) log files and from that, determine if an IP address should be blocked since it's being malicious/intruding on the network. Blocks are blocked for a day, and then it needs to reoffend.

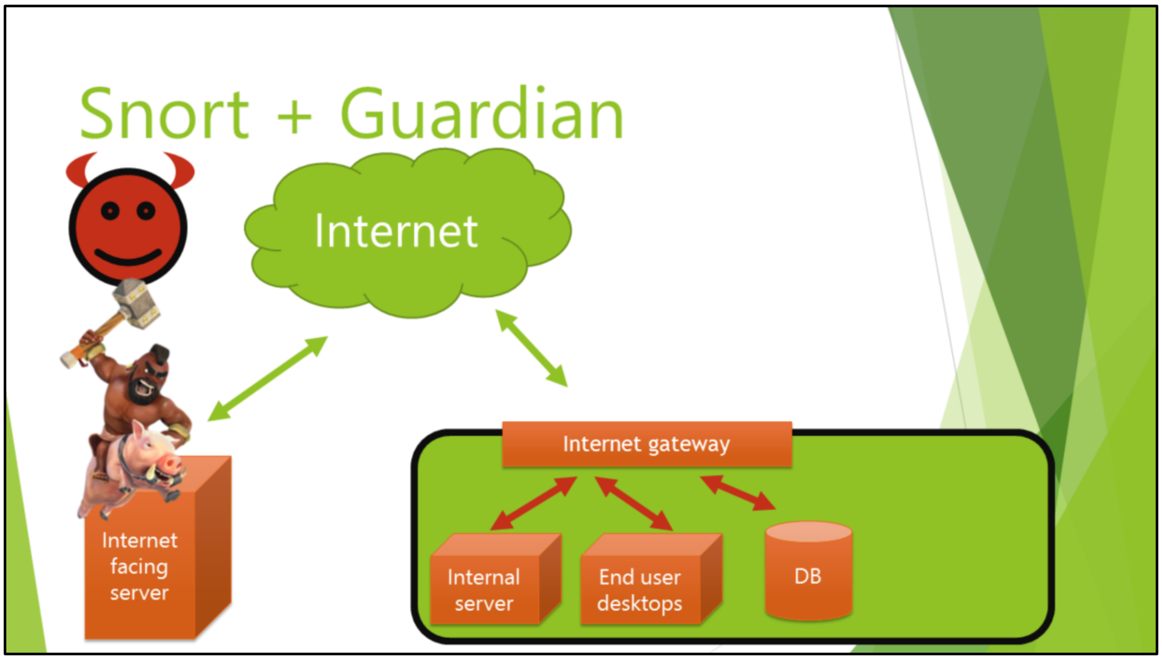
Snort + Guardian



Snort + Guardian

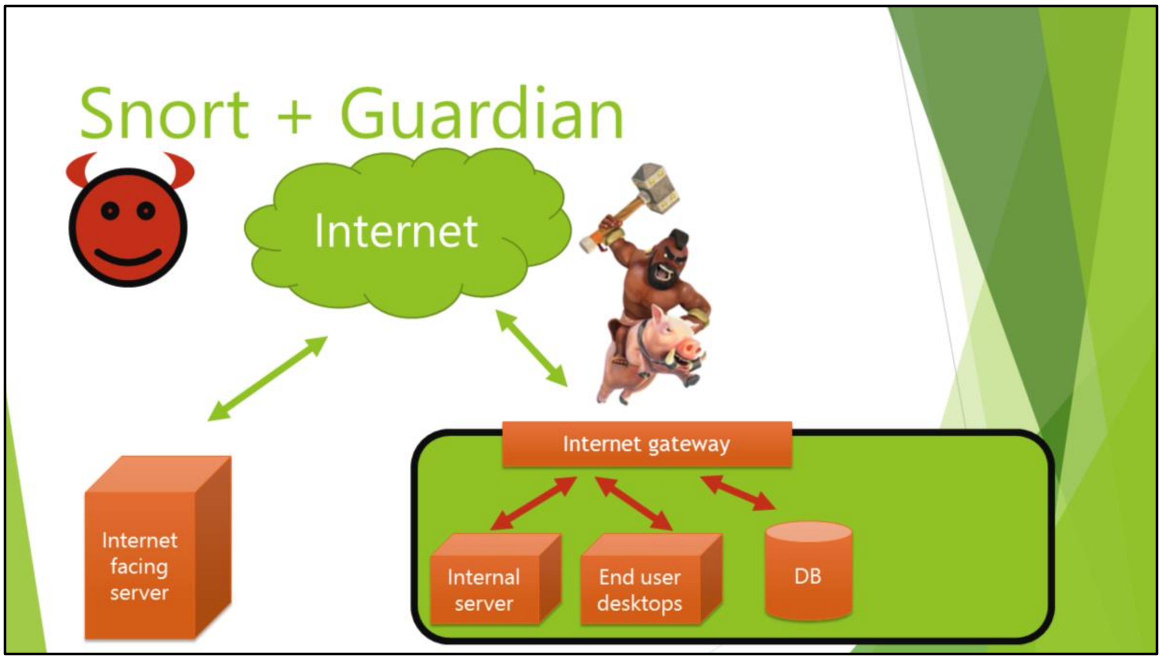
- Block IPs based on snort rules
  - Snort detects a triggered a rule
  - Guardian determines an offending IP
  - Guardian adds a rule in the firewall
- Intrusion Detection System -> Intrusion Prevention System

From the wiki:
For example; If you have one of Snort's “Scan” rules enabled and a system on the internet attempts to perform a portscan which defined by that rule, Guardian will automatically add a firewall rule to block the system's IP address. This prevents them from any future communication with your IPFire system for a period of time (by default, one day).



Snort + Guardian

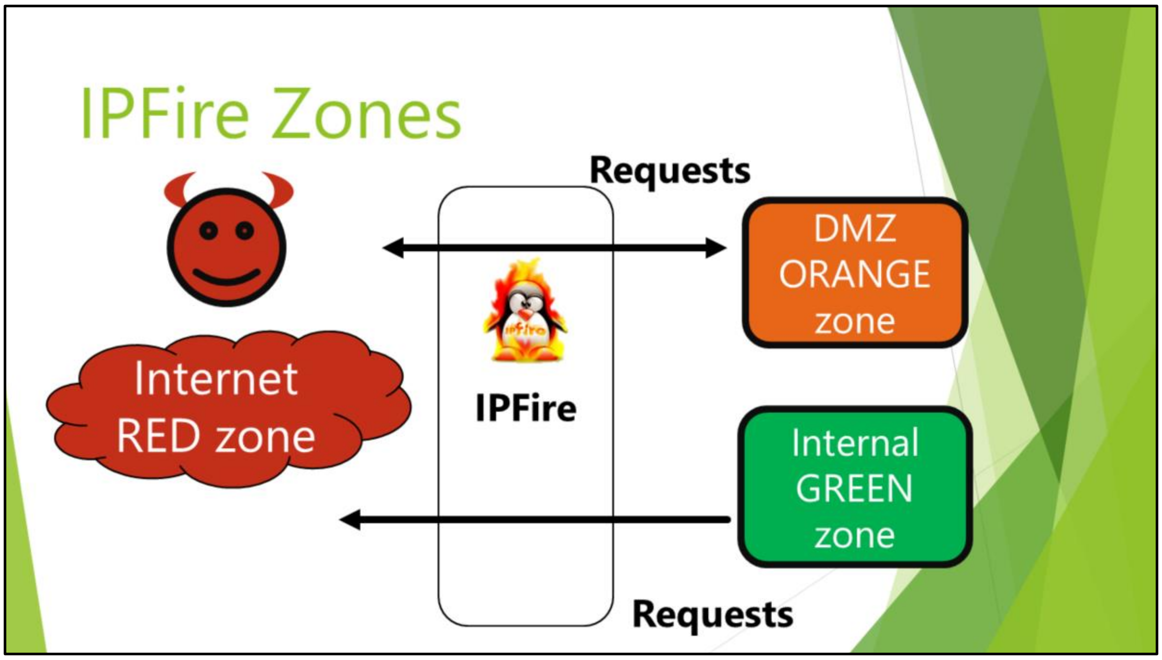
2 places to install Snort. The first we'll cover is on an internet facing server. Ideally, guardian would prevent an malicious IP's from attacking the server.



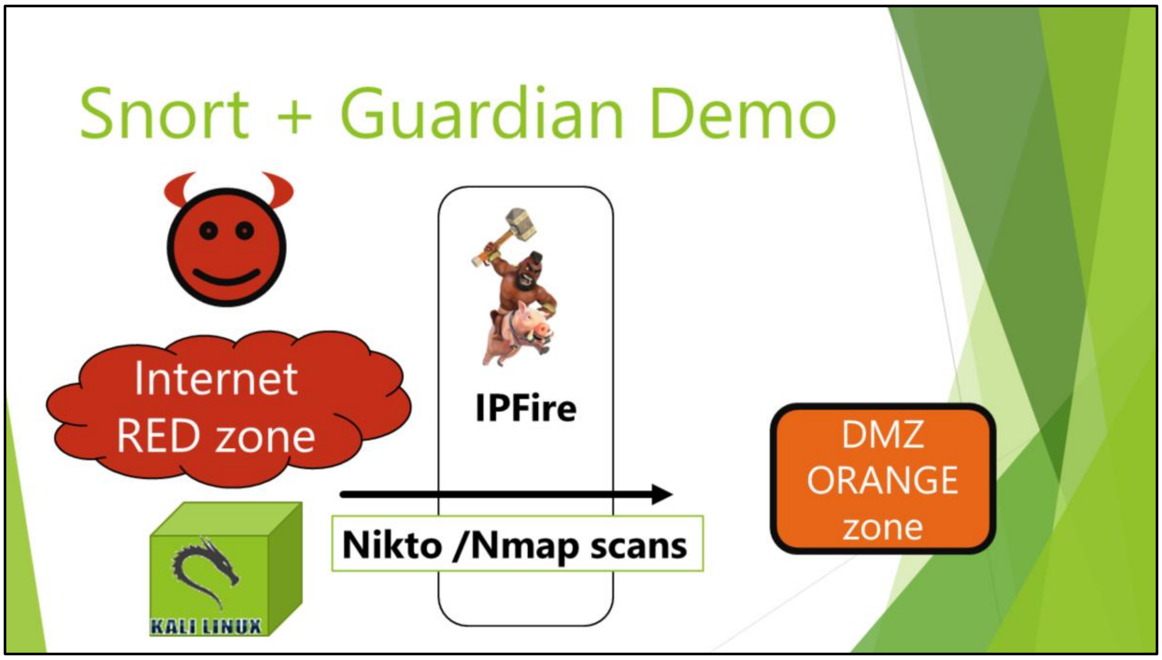
Snort + Guardian

second, on the internet gateway. same idea as on the internet facing server, prevent the internet from harming the LAN.

Side note: it didn't really make sense to us to have guardian running inside the LAN. I mean, it's possible, but it seemed like if you have a device that's being malicious, it shouldn't just be blocked, it should be disconnected since it's probably compromised. Just our opinion, and that's why we didn't include it.

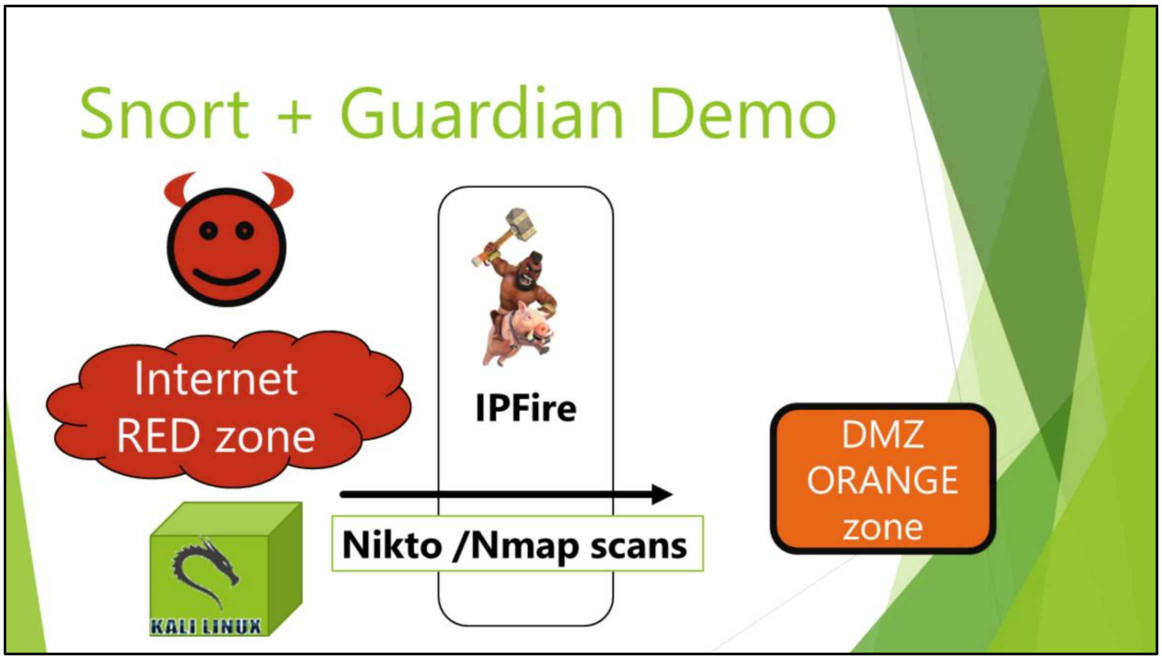


IPFire Zones



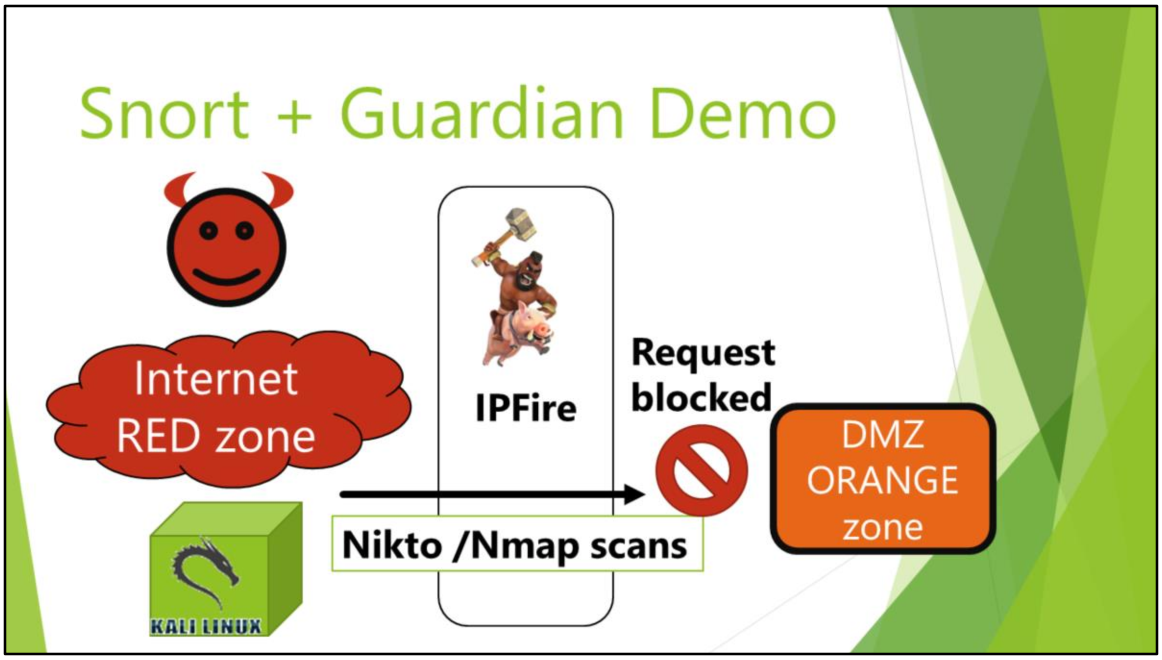
Snort + Guardian Demo

Maybe worth it?

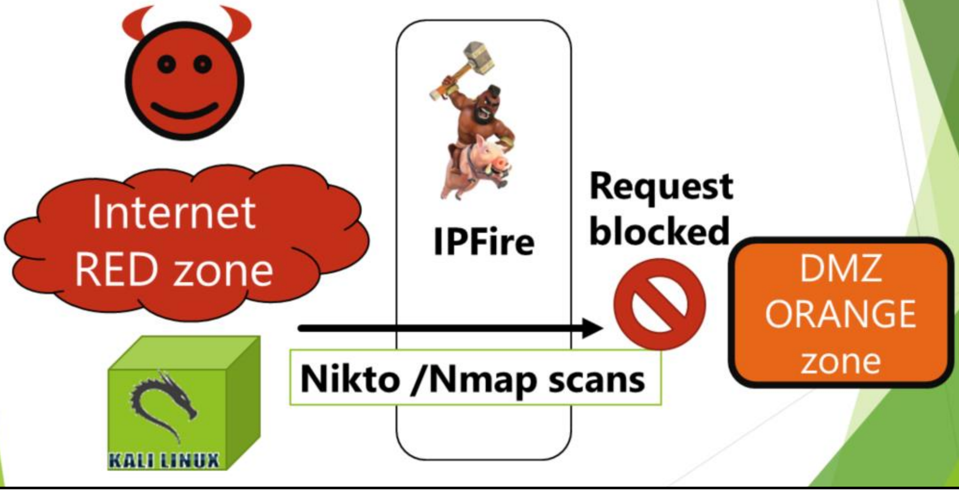


Snort + Guardian Demo

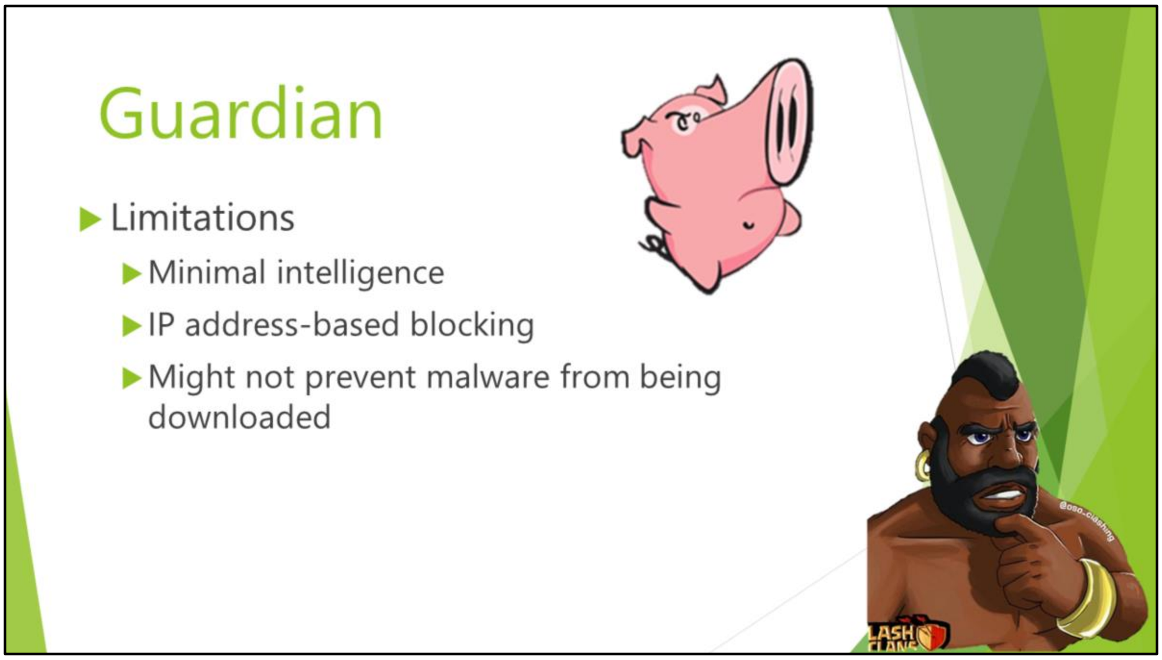
Maybe worth it?



Snort + Guardian Demo



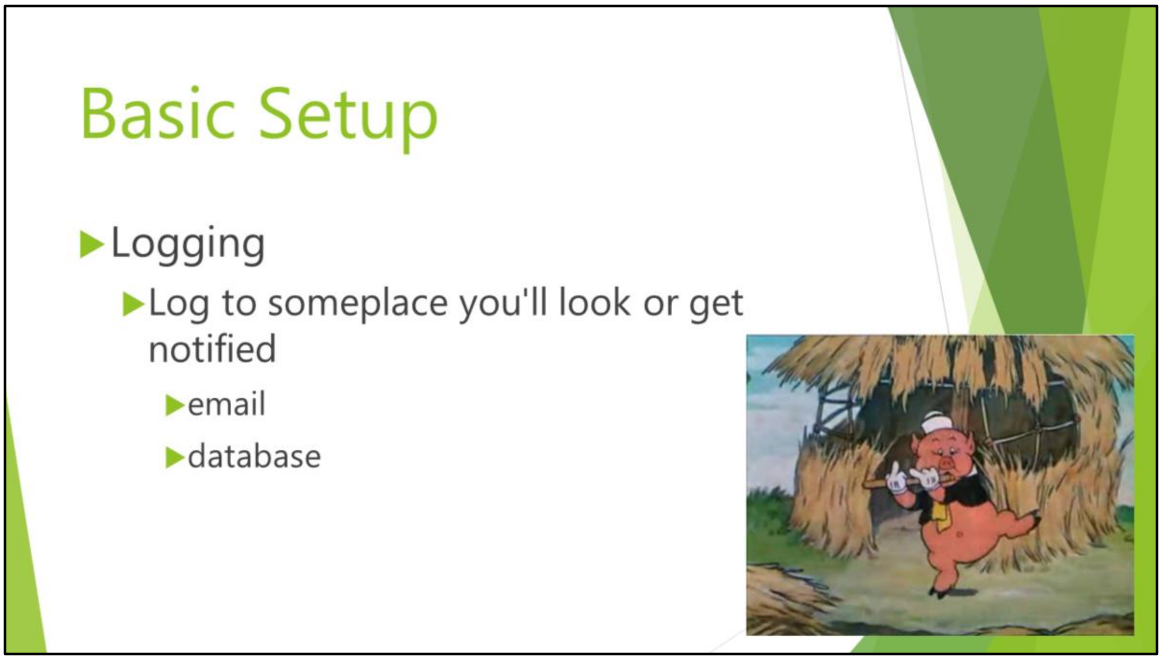
Maybe worth it?



Guardian

- Limitations
  - Minimal intelligence
  - IP address-based blocking
  - Might not prevent malware from being downloaded

Guardian is a nice way to beef up Snort. It's not perfect though.
Limitations
if you have crappy snort rules, you have crappy prevention
blocks by IP addrs. -> false positives, doesn't help if the ip addr keeps changing
depending on the size of malware and internet speed chances are that the delay between a snort rule being hit/run into and guardian implementing a block on ip, the malware would probably have already downloaded. (Guardian requires an IP addr to offend a certain # of times before it blocks it.)

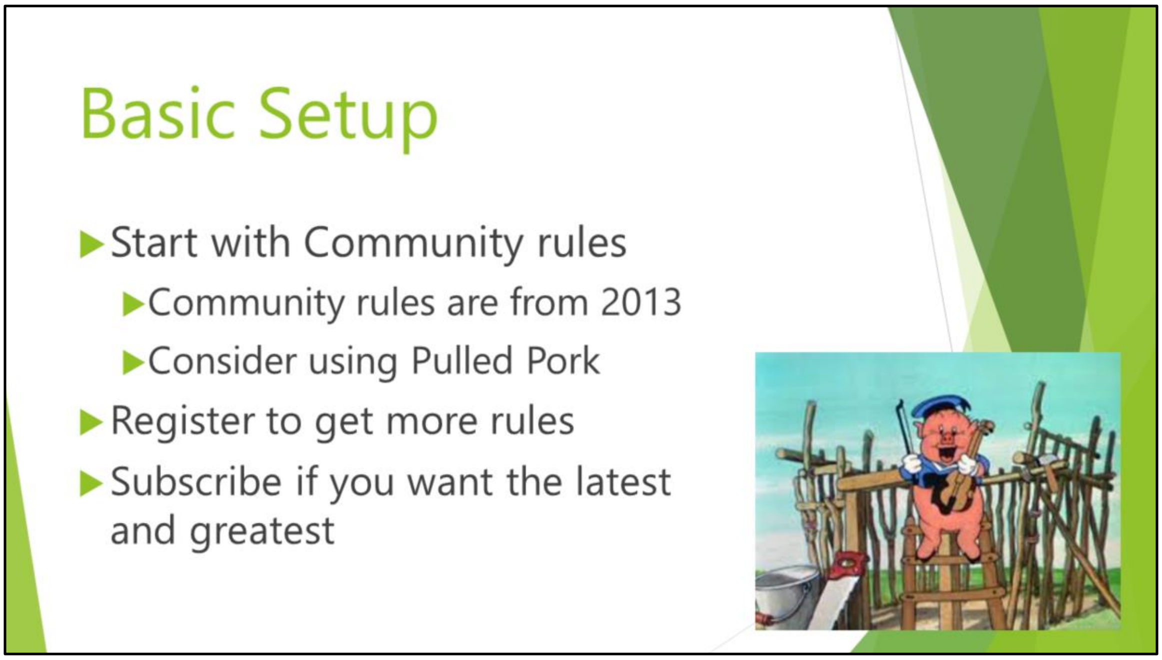


Basic Setup

- Logging
  - Log to someplace you'll look or get notified
    - email
    - database

You really just want logs. If you want logs for historical data, maybe like analysing stuff, then generic logging is best. Otherwise, for the everyday user, go with Network intrusion detection mode as that will reduce the size of the logs.

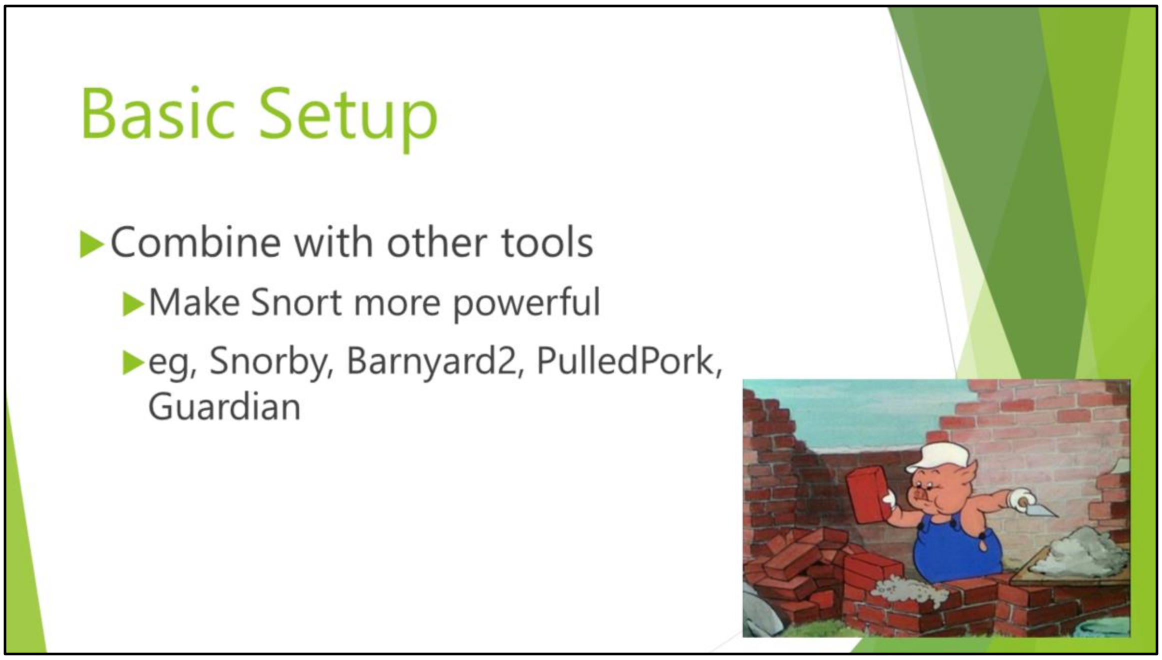
Ideally you want the logs to end up in someplace you'll look. Or, get the alerts emailed to you. Maybe even put them into a DB.
Basically, don't just forget about the logs.



Basic Setup

- Start with Community rules
  - Community rules are from 2013
  - Consider using Pulled Pork
- Register to get more rules
- Subscribe if you want the latest and greatest

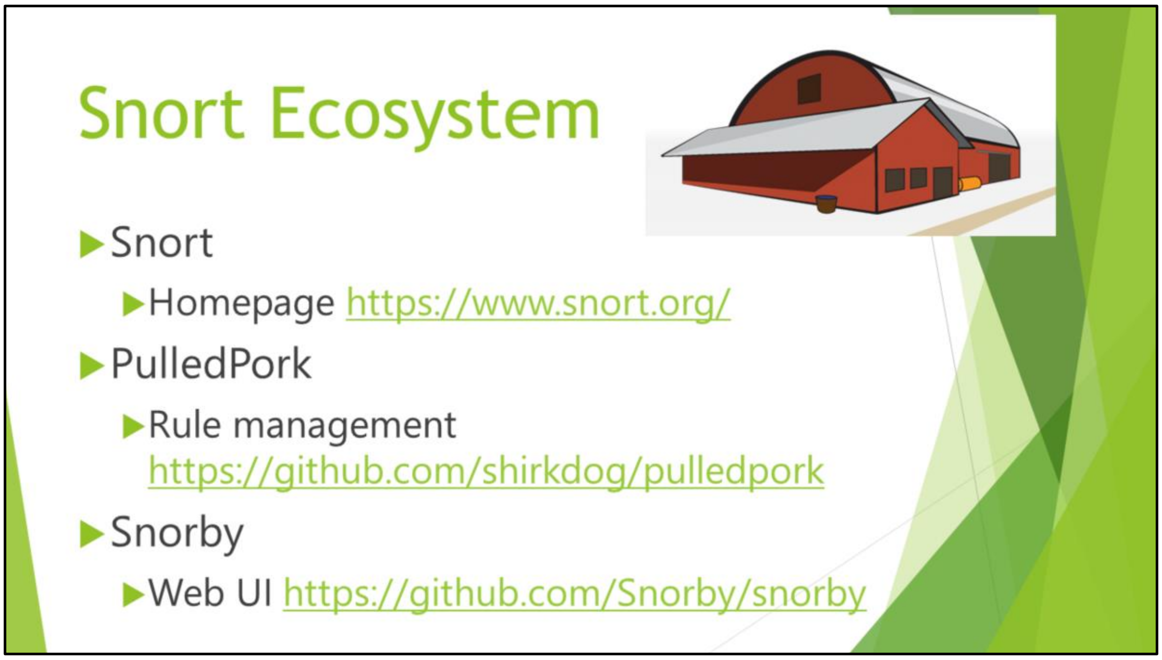
Build a solid (stick) foundation with rules. Start with the community rules which are kind of old if you just get them from the website. Use Pulled Pork to update/manage your rules.
It's beneficial to register for more rules. the registered rules consist of rules that have been part of the subscription rules for at least 60 days, along with possibly other rules.
The subscription rules, although they cost, they're the latest and greatest released by Snort/Cisco. <- I guess this can be seen as max protection snort can offer?



Basic Setup

- Combine with other tools
  - Make Snort more powerful
  - eg, Snorby, Barnyard2, PulledPork, Guardian

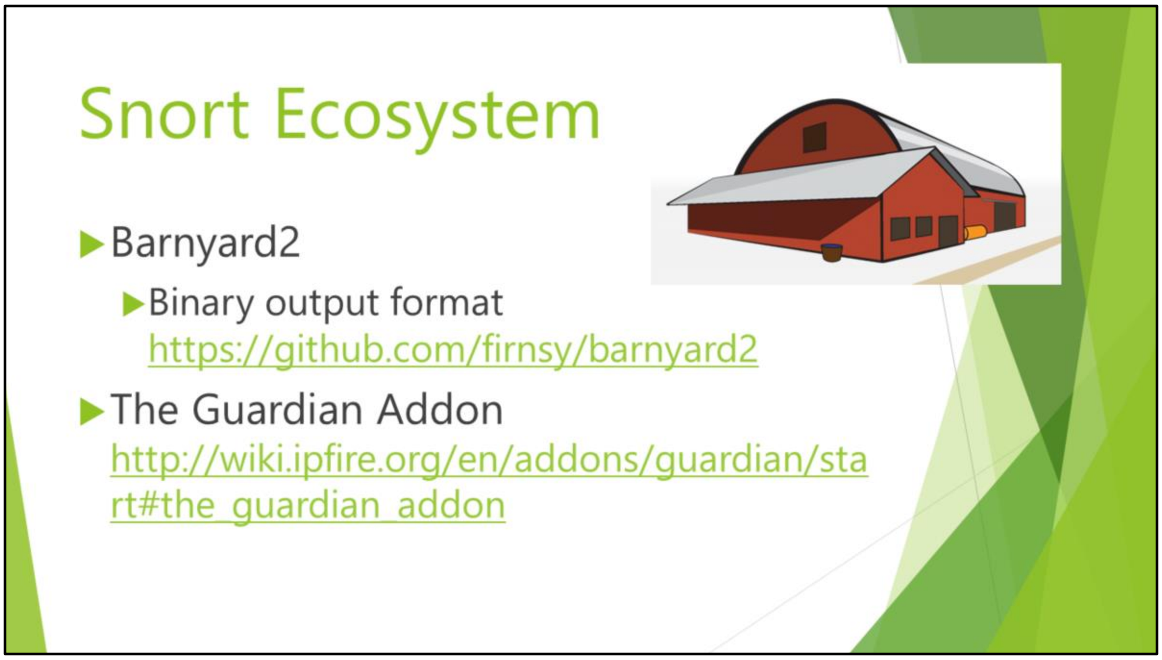
Bulk up Snort by combining it with other tools.
Snorby is a Web UI for viewing snort logs. Makes the logs look pretty. And UIs are nice.
Barnyard2 allows for Snort to process packets faster by logging in binary format instead of plain text.
Pulled pork is the rule updater/manager.
Guardian is a tool to block offending IPs by reading snort logs.



Snort Ecosystem

- Snort
  - Homepage https://www.snort.org/
- PulledPork
  - Rule management https://github.com/shirkdog/pulledpork
- Snorby
  - Web UI https://github.com/Snorby/snorby

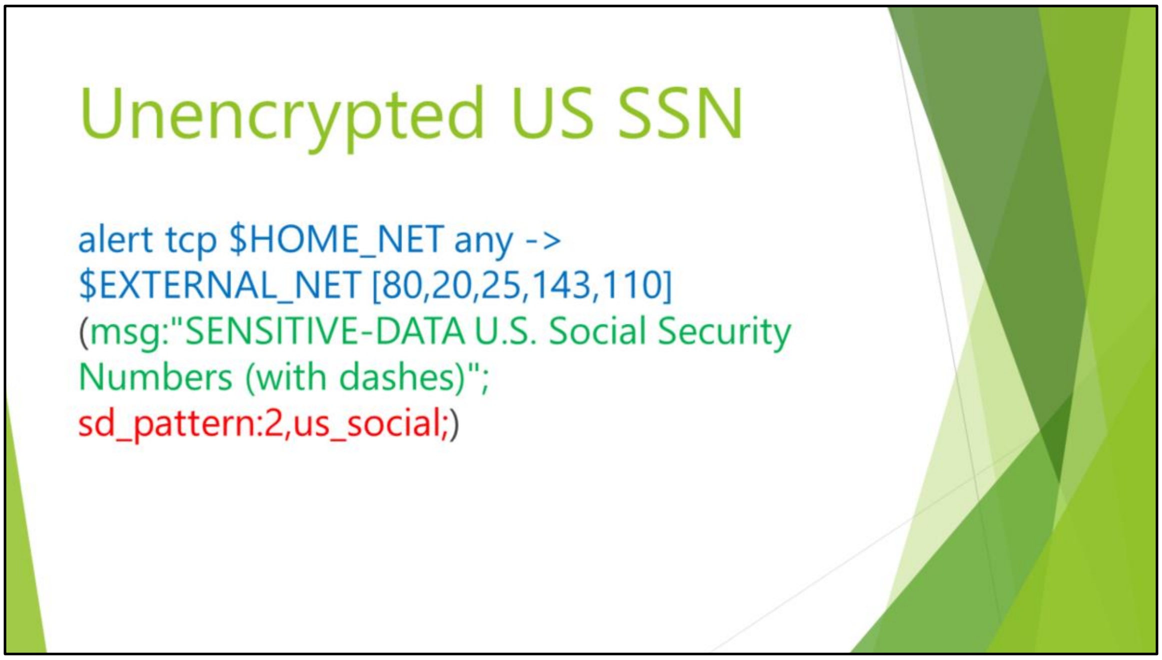
Useful links according to us.
Meant to introduce you to the Snort ecosystem.



Snort Ecosystem

- Barnyard2
  - Binary output format https://github.com/firnsy/barnyard2
- The Guardian Addon http://wiki.ipfire.org/en/addons/guardian/start#the_guardian_addon

Useful links according to us.

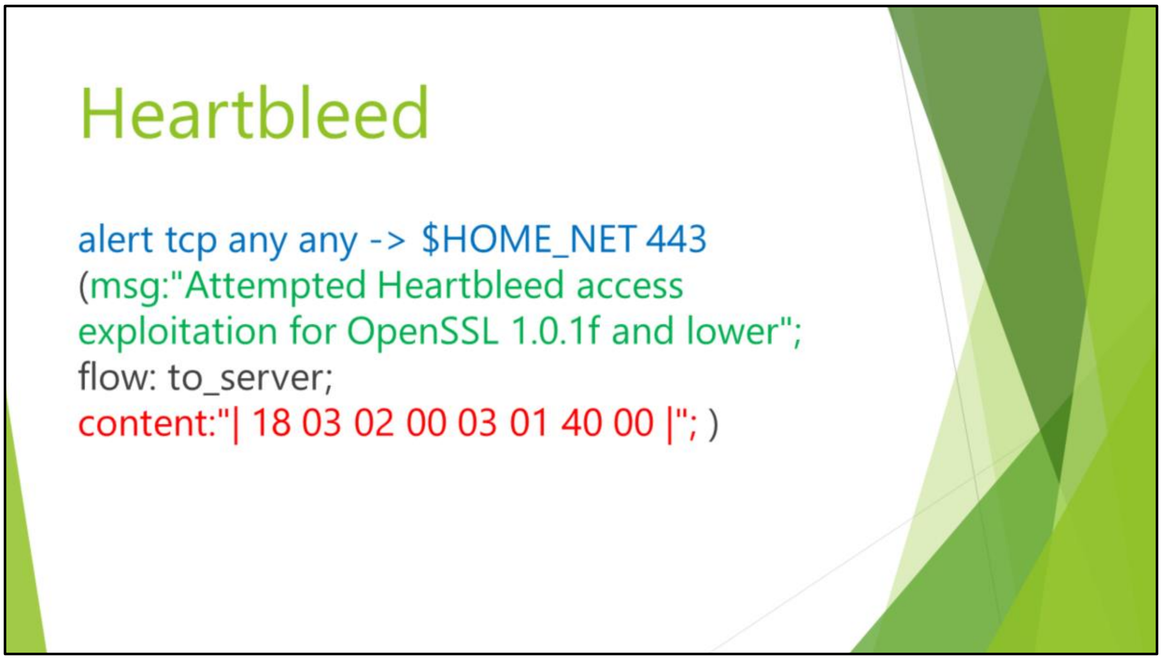


Unencrypted US SSN

```
alert tcp $HOME_NET any ->
$EXTERNAL_NET [80,20,25,143,110]
(msg:"SENSITIVE-DATA U.S. Social Security
Numbers (with dashes)";
sd_pattern:2,us_social;)
```

Additional content!

Similar to the Credit card example. This time, the pattern that's being matched it a US SSN.



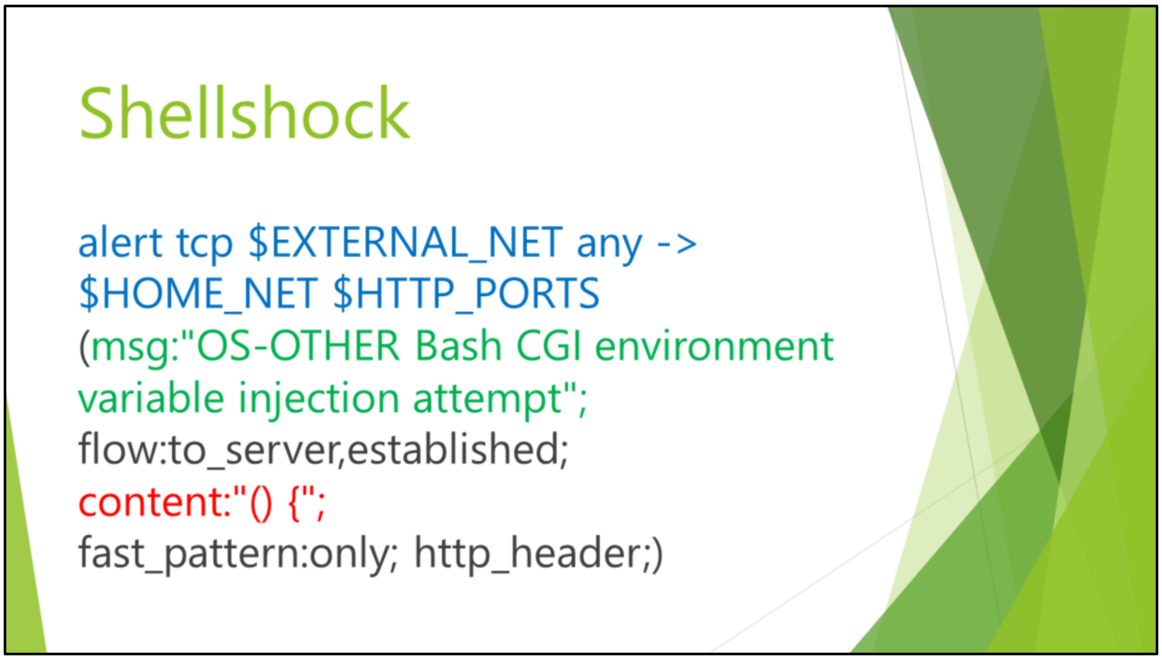
Heartbleed

```
alert tcp any any -> $HOME_NET 443
(msg:"Attempted Heartbleed access
exploitation for OpenSSL 1.0.1f and lower";
flow: to_server;
content:"| 18 03 02 00 03 01 40 00 |"; )
```

A tcp packet from anywhere (don't know why it's not limited to an external network) is sent to your home network on port 443 (default port for https).

black/grey text: it's checking that the flow is from the client to the server
red text: not entirely sure what the hex is getting at, but it's looking to match the hex characters to some part of the packet's payload.

Note: there are multiple rules related to and attempted heartbleed access. If you look in the registered rules, and grep for "Heartbleed", you'll find a bunch. Another useful tip that will be covered in tutorial is that there are plenty of the rules in the registered set that reference a CVE. so that's one way of looking for rules.



Shellshock

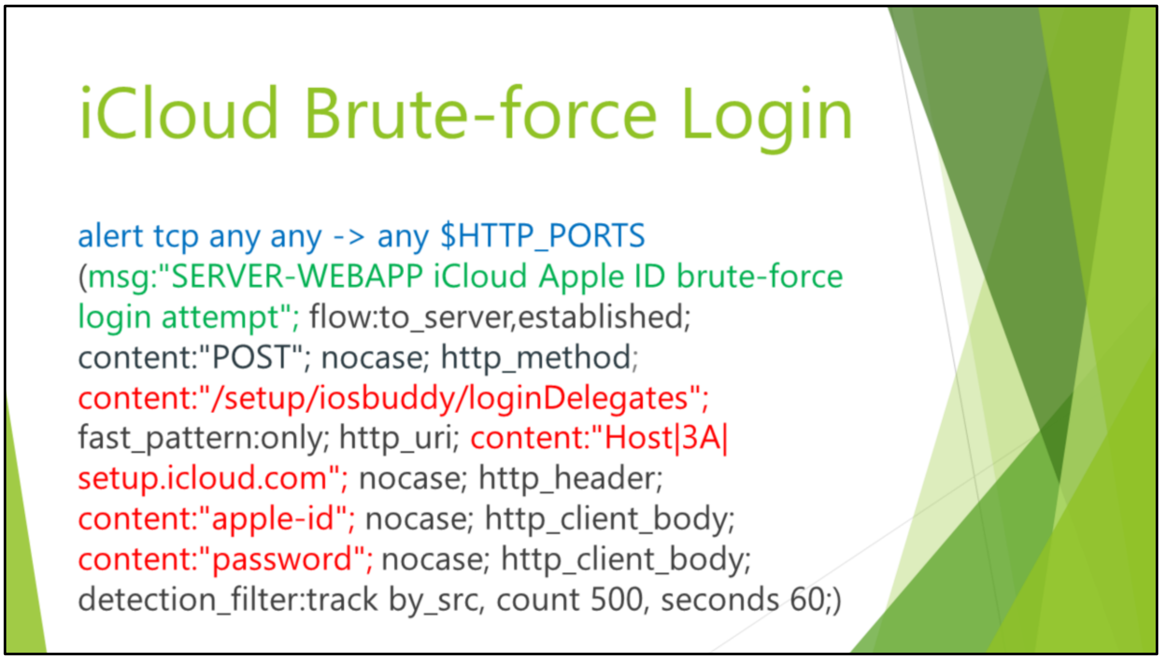
```
alert tcp $EXTERNAL_NET any ->
$HOME_NET $HTTP_PORTS
(msg:"OS-OTHER Bash CGI environment
variable injection attempt";
flow:to_server,established;
content:"() {";
fast_pattern:only; http_header;)
```

again, multiple versions of rules for this vulnerability.

Alert on a TCP packet from an external network on any port to a home network on the HTTP ports (most likely just port 80, I'd think).

Checking that the flow is from the client to the server and that the connection is listed as established.

fast_pattern: just a type of searching. more details at:
http://manual.snort.org/node382.html
http_header: the content search (so looking for "() {" is limitted to looking into any http headers within the packet payload. more details:
http://manual.snort.org/node358.html



iCloud Brute-force Login

```
alert tcp any any -> any $HTTP_PORTS
(msg:"SERVER-WEBAPP iCloud Apple ID brute-force
login attempt"; flow:to_server,established;
content:"POST"; nocase; http_method;
content:"/setup/iosbuddy/loginDelegates";
fast_pattern:only; http_uri; content:"Host|3A|
setup.icloud.com"; nocase; http_header;
content:"apple-id"; nocase; http_client_body;
content:"password"; nocase; http_client_body;
detection_filter:track by_src, count 500, seconds 60;)
```

Vulnerabilty from last year: https://github.com/Pr0x13/iBrutr
If I recall, it is, or is related to, the vulnerabilty that allowed someone to access a whole bunch of celebrities iClouds' and leak private photos of them

content: multiple of these, so in order of appearance.
look for "POST" (don't care about case) in the http method of the packet payload.
look for "/setup/iosbuddy/loginDelegates" in the request URI of the packet payload
look for "HOST|3a|setup.icloud.com" without caring about case in the http header of the packet payload. (not sure why the 3A. also, I guess it's good to not that any content within | | indicated that it's treated as a hex character.)
look for "apple-id" in the body of the http client request in the packet payload
look for "password" in the body of the http client request in the packet payload
detection_filter: detect the rate of the packet by course. it must eceed a count of 500 packets within 60 seconds.

flow : from the client to the server, on an established connection.