HONEYPOTS & IDS 101

THE HONEYNET PROJECT ANNUAL WORKSHOP IN AUSTRIA 2019

$ whoami

- Miguel Raúl Bautista Soria
  - Miguel Bautista
    - Mike

- Security analyst with Cisco TALOS (formerly SourceFire VRT)
  - Analysis and research of malicious activity and vulnerabilities
  - Development of detection content for Snort and ClamAV

- Chief Workshop Officer and Full Member (2+ years) at The Honeynet Project

- InfoSec, music, video games and technology fan

AGENDA

- First half of the day - Honeypots
  - What is a Honeypot?
  - Honeypots and their classification
  - Honeypots today
  - Deploying and maintaining a Honeypot
  - Create Your Own Honeypot (CYOH)

- Second half of the day – Intrusion Detection Systems (IDS)
  - What is an IDS?
  - Types of IDS
  - IDS or IPS
  - Deploying an IDS in our network: Snort
  - Creating detection content for Snort

HONEYPOTS

A drop of honey catches more flies than a gallon of gall.

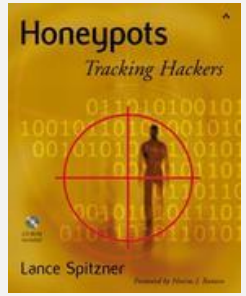
Abraham Lincoln
United States
1809 // 1865

www.wordsandquotes.com

http://www.wordsandquotes.com/quote/a-drop-of-honey-catches-more-flies-than-a-gallon-abraham-lincoln-5158

WHAT IS A HONEYPOT?

Is a security resource whose value lies in being probed, attacked, or compromised[1].



[1] Honeypots: Tracking Hackers. Lance Spitzner, September 2002.

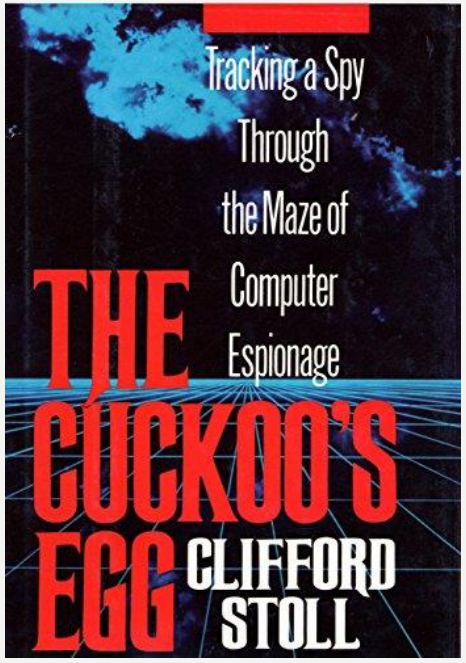
WHAT IS A HONEYPOT? (IN OTHER WORDS)

Is a decoy device deployed in a network location to receive and, therefore, detect, malicious traffic or related patterns.

WHEN HONEYPOTS STARTED?

1986 – Clifford Stoll discovers a security breach in one of the computers he used to manage at the Lawrence Berkeley National Lab.

- Book: *The Cuckoo´s Egg.* Published in 1989 by Clifford Stoll.



ATTENTION: Mrs. Barbara Sherwin Document Secretary

SUBJECT: SDI Network Project

Dear Mrs. Sherwin:

I am interested in the following documents. Please send me a price list and an update on the SDI Network Project. Thank you for your cooperation.

Very truly yours,
Laszlo J. Balogh

#37.6 SDI Network Overview Description Document, 19 pages, December 1986

#41.7 SDI Network Functional Requirement Document, 227 pages, Revised September 1985

#45.2 Strategic Defense Initiations and Computer Network Plans and Implementations of Conference Notes, 300 Pages, June 1986

#47.3 SDI Network Connectivity Requirements, 65 pages, Revised April 1986

#48.8 How to Link to SDI Network, 25 pages, July 1986

#49.1 X.25 and X.75 Connection to SDI Network (includes Japanese, European, Hawaiian, 8 pages, December 1986)

#55.2 SDI Network Management Plan for 1986 to 1988, 47 pages, November 1986)

#62.7 Membership list (includes major connections), 24 pages, November 1986)

#65.3 List, 9 Pages, November 1986

"Hey Mike, remember those carrots I left out for bait in January?"
"You mean those SDI* files you concocted?"
"Yeah," I said. "Well, my dear, sweet, **nonexistent secretary** just received a letter."

*SDI: Strategic Defense Initiative

WHEN HONEYPOTS STARTED?

- 1986 – Clifford Stoll discovers a security breach in one of the computers he used to manage at the Lawrence Berkeley National Lab.
  - Book: *The Cuckoo´s Egg.* Published in 1989 by Clifford Stoll.

- 1991 – Bill Cheswick discovers an attacker attempting to exfiltrate private information through a supposed security hole affecting an email server at the Bell Labs of AT&T.
  - Article: *An Evening with Berferd.* http://www.cheswick.com/ches/papers/berferd.pdf.

An Evening with Berferd
In Which a Cracker is Lured, Endured, and Studied

Bill Cheswick

AT&T Bell Laboratories

Abstract

On 7 January 1991 a cracker, believing he had discovered the famous sendmail DEBUG hole in our Internet gateway machine, attempted to obtain a copy of our password file. I sent him one.

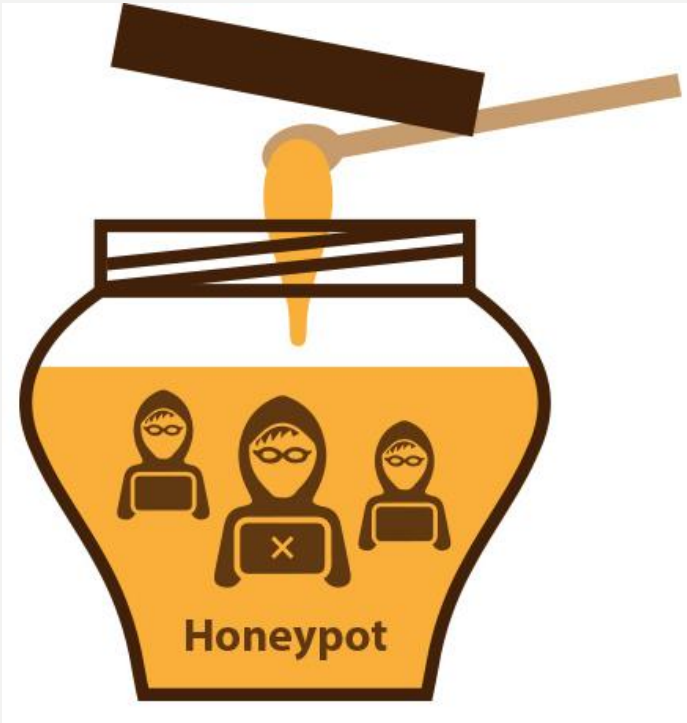
For several months we led this cracker on a merry chase in order to trace his location and learn his techniques. This paper is a chronicle of the cracker's "successes" and disappointments, the bait and traps used to lure and detect him, and the chroot "Jail" we built to watch his activities.

We concluded that our cracker had a lot of time and persistence, and a good list of security holes to use once he obtained a login on a machine. With these holes he could often subvert the *uucp* and *bin* accounts in short order, and then *root*. Our cracker was interested in military targets and new machines to help launder his connections.

WHEN HONEYPOTS STARTED?

- 1999 – The Honeynet Project was born as a non-profit organization established in the USA and founded by Lance Spitzner.
  - The main objective of the Project is the research related to Honeypots and potential threats to the network, devices and assets.
  - Lance also introduced for the first time the term *honeynet*, which can be understood as a network of high interaction honeypots, simulating a production network and that is configured in a way that all the activity is monitored, logged and discretely regulated.
    https://www.honeynet.org/about

CLASSIFYING HONEYPOTS



CLASSIFYING HONEYPOTS

- By its mode of operation:
  - Low interaction Honeypots
  - Medium interaction Honeypots
  - High interaction Honeypots
- By their environment:
  - Production Honeypots
  - Honeypots for research

LOW INTERACTION HONEYPOTS

- Devices whose main goal is emulate the services of a real system in order to have enough interaction with threats or bad actors and gather as much data as they can.
- They are very efficient for detection of malicious patterns, malware capture and generate statistics of network traffic, in general.
- Easy to detect.
- Examples: Telnet, FTP, HTTP, SSH.

MEDIUM INTERACTION HONEYPOTS

- Can expect certain activity and are designed to give certain responses beyond what a low interaction honeypot would provide.
- In case an specific exploit or threat is received, it will never get full access to the operating system, limiting risk for being only an “specific” emulated application.
- Are usually more time consuming to install and configure than low interaction honeypots however, it rewards us with a large amount of information.
- Examples: An specific vulnerable version or feature of Microsoft IIS web server.

HIGH INTERACTION HONEYPOTS

- Real devices and the interaction is directly with the software installed in the operating system.
- Allows the attacker to have a higher control of the system, which implies a higher risk and therefore the need of having an external control system continuosly working to monitor, store and process the captured information.
- Suitable for research environments that studies the techniques and trends of security attacks in a device, corporate network or even the Internet.
- Very complex to deploy, however, once implemented correctly, the high-interaction honeypot can give insight into attackers that no other honeypot can.

PRODUCTION HONEYPOTS

- Devices located in a real network along with other productive servers and devices.

- Commonly they are low interaction and work as a complement to the detection of network threats, which in some way mitigates the risks in the network where they are located.

HONEYPOTS FOR RESEARCH

- Devices deployed with the only objective of study the behavior and new trends of threats affecting networks and other devices in the Internet, caused by either thrat actors or massive automated techniques.
- They are deployed in controlled environments and for academic purposes.
- Their main characteristic is capture and analyze all the collected data, so its configuration and management can be more complex.

CLASSIFYING HONEYPOTS

- By its mode of operation:
  - Low interaction Honeypots
  - Medium interaction Honeypots
  - High interaction Honeypots
- By their environment:
  - Production Honeypots (low interaction)
  - Honeypots for research (high interaction)

COMPARISON CHART

	Low Interaction	Medium Interaction	High Interaction
Installation	Easy	Involved	Difficult
Maintenance	Easy	Involved	Difficult
Level of risk	Low	Medium	High
Requires control?	No	Variable	Yes
Information gathering	Limited	Variable	Extensive
Type of interaction	Emulated services	As required	Total control

ADVANTAGES OF HONEYPOTS

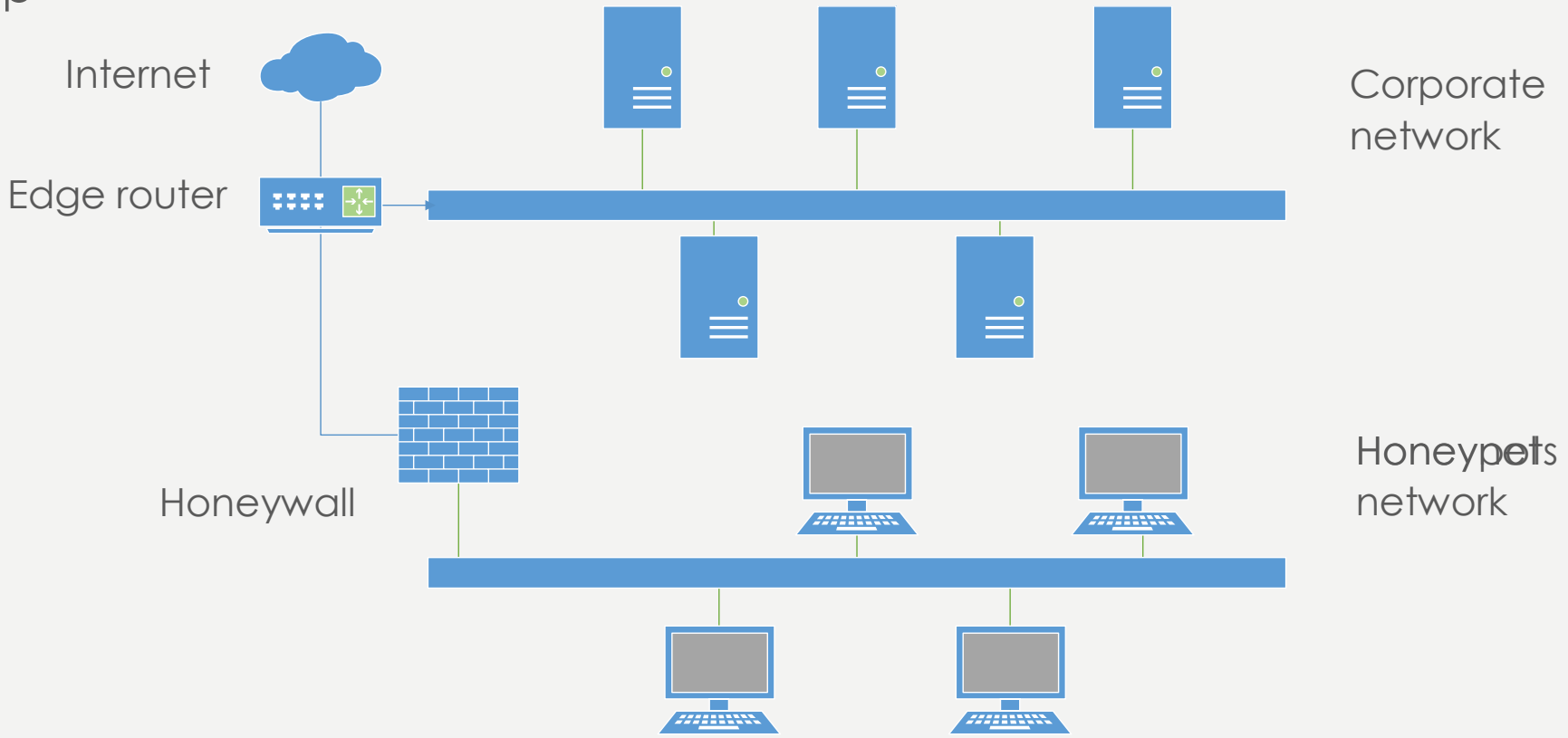
- High value of the data they collect
- Resources saving
- Detection of false positives and false negatives
- Support of IPv4 and IPv6
- Simplicity
- Allow to discover new attacks, malware and trends in the Internet

DISADVANTAGES OF HONEYPOTS

- Limited scope and field of view
- The malicious activity observed in the honeypot may not be the whole picture
- Can be identified easily using fingerprinting techniques
- Risk of losing control of the device
- Collected data could be lost if something goes wrong

HONEYNET

- A Honeynet is a set of more than one high interaction honeypot
- All the honeypots are deployed under a network traffic monitor and control setup

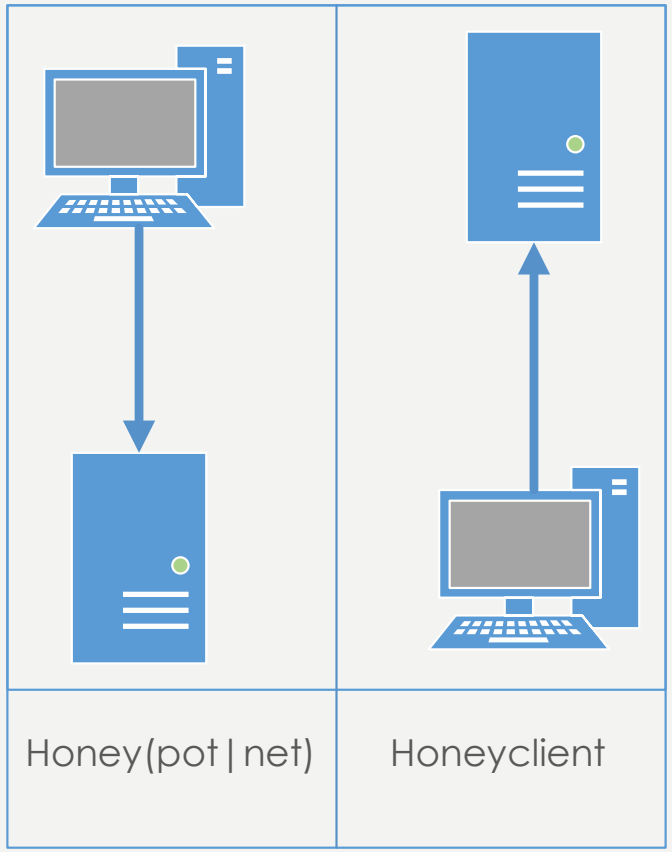


OTHER HONEYPOTS: HONEYCLIENT

- Is a tool that allows to identify malicious servers on the Internet
- Interact and analyze the server content looking for malware redirects, code injections, shellcodes, etc.
- They exist for the most used services on the Internet such as SMTP, HTTP or FTP however, HTTP is where you can see the most effectiveness.

OTHER HONEYPOTS: HONEYCLIENT

COMPARISON



OTHER HONEYPOTS: HONEYCLIENT

OBJECTIVES:

- Evaluation / characterization of websites
- Security tests
- Detection of potential 0-day attacks
- Locate problematic neighbors
- Obtaining malware and exploits

OTHER HONEYPOTS: HONEYCLIENT

OBSOLETE AGAINST:

- Black lists
- Dialog boxes
- Anti-crawler techniques
- Dynamic events embedded in pages
- Behavior in navigation

HONEYPOTS TODAY

HONEYPOTS TODAY

LOW INTERACTION

- Honeyd
- Dionaea*
- Honeytrap
- Glastopf*
- Conpot*
- Kippo
- Cowrie*
- Snare
- Glutton
- T-Pot**

HIGH INTERACTION

- CaptureHPC (honeyclient)
- High Interacion Honeypot Analysis Toolkit
- SMB Honeypot
- Qebek
- Sebek

More here: https://github.com/paralax/awesome-honeypots

DIONAEA

- A low interaction honeypot
- A very simple tool to start capturing malware
- Emulates vulnerabilities in the most common network services (including Microsoft Windows propietary ones) in order to collect information for potential attacks
- http://dionaea.carnivore.it/

HANDS ON: DIONAEA

- Getting dionaea
- Installing/Compiling
- Running dionaea
- Catching malicious activity

GLASTOPF

- A low interaction honeypot that emulates an HTTP server with “several” pages and with some vulnerabilities.
- Allows us to catch attacks like SQL injection, XSS (Cross-Site-Scripting), Remote File Inclusion and Local File Inclusion.
- Written by Lukas Rist, former CRO of The Honeynet Project
- https://github.com/mushorg/glastopf

HANDS ON: GLASTOPF

- Getting glastopf
- Installing/Compiling
- Running glastopf
- Catching web attacks

CONPOT

- A low interaction honeypot for emulation of Industrial Control Systems (SCADA).
- Capable of emulate real production infrastructures and includes an interface that allows to “manage” the systems.
- Pictures the threat landscape for these kind of devices.
- Written by Lukas Rist and Johnny Vestergaard, both members of the HoneyNor chapter.
- https://github.com/mushorg/conpot

HANDS ON: CONPOT

- Getting Conpot
- Installing/Compiling
- Running Conpot
- Catching attacks

KIPPO

- Low interaction honeypot that simulates a Secure Shell (SSH) server.
- Useful to detect password-based brute force attacks.
- Most used commands at the post-exploitation phase.
- Record all the actions of an attacker inside a compromised server.
- https://github.com/desaster/kippo
- No longer in maintenance!

COWRIE

- A new medium interaction honeypot for SSH: AKA NG Kippo.
- Forked from Kippo and remastered with increased support for more commands, user activity and new features.
- Full fake filesystem based in Debian 5 with a lot of features that aims to provide an undetectable Secure Shell server.
- Written by Michel Oosterhof, member of The Honeynet Project in the HoneyNED chapter.
- https://github.com/cowrie/cowrie

HANDS ON: COWRIE

- Getting Cowrie
- Installing/Compiling
- Running Cowrie
- Catching attackers

DEPLOYING AND MAINTAINING A HONEYPOT

DEPLOYMENT

- First define a goal
  - Malware capturing.
  - Detection of new vulnerabilities or activity of existing ones.
  - Detection of attacks.
  - Profiling and network analysis.
  - Researching attackers' tools, tactics, and motives

- Select the honeypot(s) and the proper infrastructure and environment.
- Include also the amount of resources needed (services, devices, network topology, other security products, etc).

MAINTENANCE

- Alert detection
  - Some of the best ways to configure and receive alerts. Without proper alert mechanisms, a honeypot has very little, if any, value.
- Response policies
  - What types of honeypot activity you will react to and how. Without defined policies, critical information can be lost or destroyed.
- Data analysis
  - A process for turning collected data into valuable information.
- Updates
  - How to ensure that your honeypot stays current to new threats and attacks.

MAINTENANCE

- Maintaining our honeypot deployments is critical to their success.
- We can deploy the greatest honeypot solutions ever, but they will be worthless if we do not do anything with them.
- To effectively sustain our honeypot deployments, we must develop effective policies and practices for alert detection, reaction, and data analysis.
- We also **must** keep our honeypots up to date.

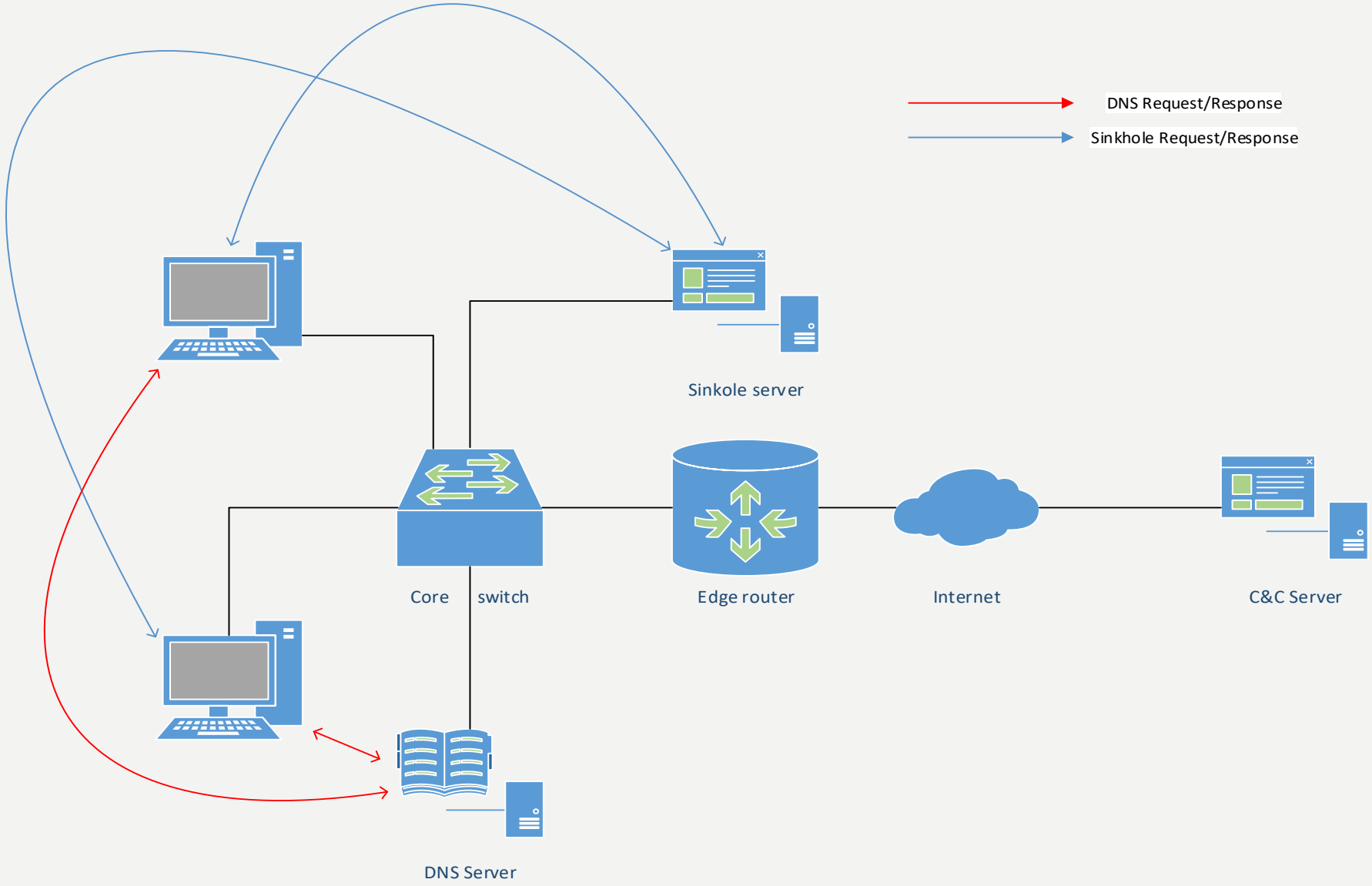
EXTRA RESOURCE: SINKHOLE

- Device that receives, process and serves any kind of malicious request.
- Allows to perform reverse engineering to malicious communication protocols, requests and network traffic.
- Identification of malicious domain names:
  - Botnets
  - Spyware
  - Phishing
  - Adware, etc

EXTRA RESOURCE: SINKHOLE

- A DNS server is required to redirect the malicious requests to the Sinkhole.
- The Sinkhole will receive all the network activity generated by hard-coded domain names.
- It will reply with a fake file with the requested name.
- Logs all the activity received for further analysis.
- Manual configuration must be required to ensure effectiveness.

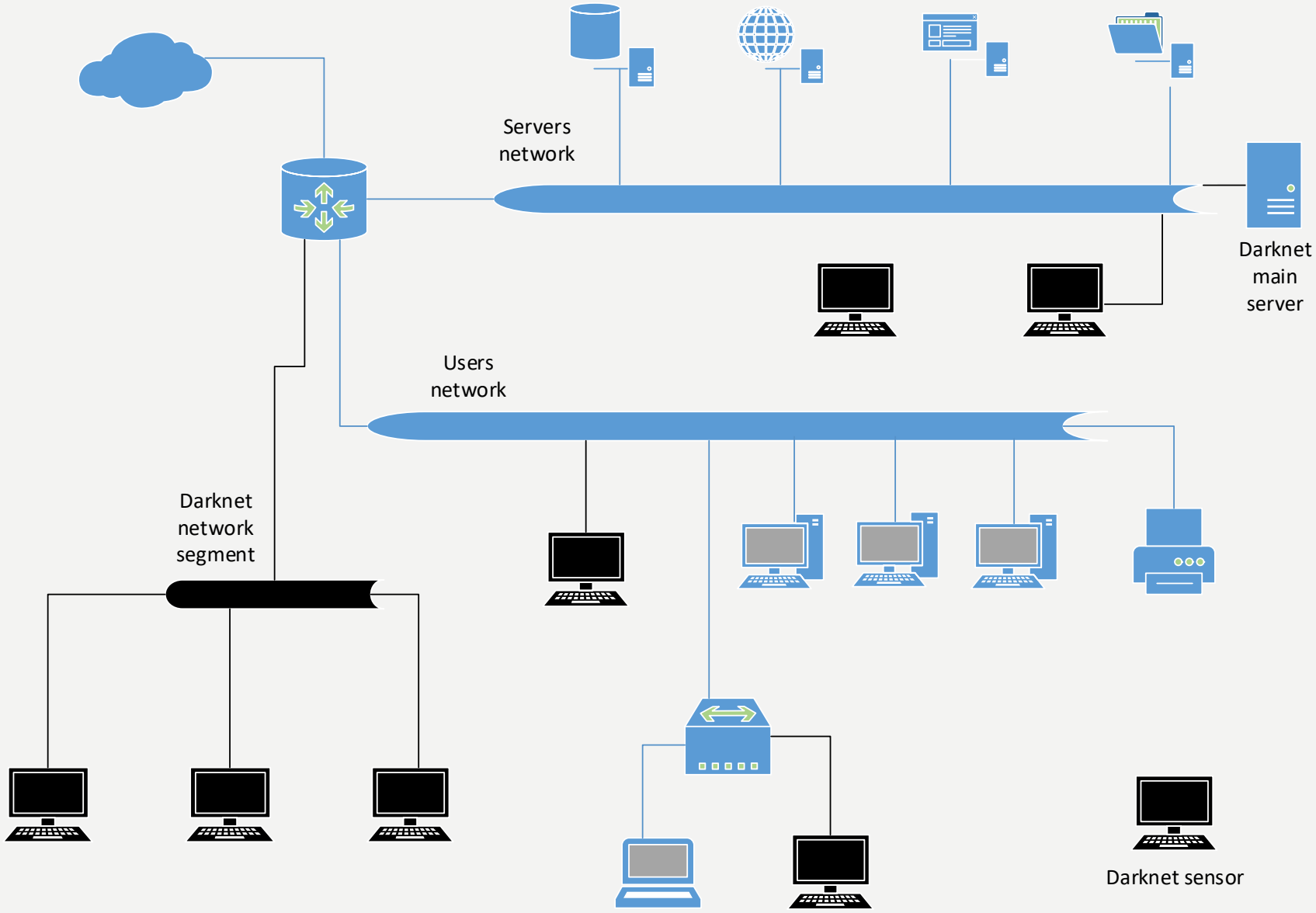
EXTRA RESOURCE: SINKHOLE



EXTRA RESOURCE: DARKNET

- A device or cluster of several devices that allocates network segments or single IP addresses that are not yet assigned to a productive service or device in a network.
- All these IP addresses are explicitly reserved for the Darknet.
- The purpose of this reservation is to ensure that, if there is no service or device assigned to an IP address, then no traffic should be seen.
- If the Darknet detects network activity then there is something likely malicious or there is a misconfiguration in the network.
- The effectivity can be directly proportional to the amount of unassigned IP addresses.

EXTRA RESOURCE: DARKNET



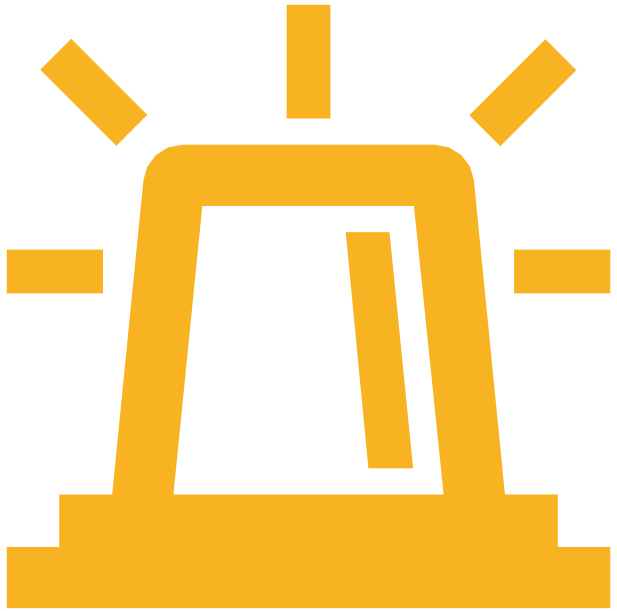
CREATE YOUR OWN HONEYPOT

CREATE YOUR OWN HONEYPOT

- Requirements:
  - Programming skills.
  - Identify what kind of Honeypot we want to create.
  - Define the scope of the Honeypot.
  - Gather the technical requirements to make it work: RFCs or public documentation.
  - Define the methods for results delivery. Could be an easy output format that can be chained to other systems.
  - Let's go.

- Let's write an SMTP Honeypot
  - https://tools.ietf.org/html/rfc821

INTRUSION DETECTION SYSTEMS



WHAT IS AN IDS

WHAT IS AN IDS?

- Intrusion Detection System

- Is a system designed for the capture and analyisis of network traffic, so we can identify:
  - External threats
  - Internal threats

- Adds another protection layer to a security architecture
  - Firewall
  - Router
  - Proxy, etc.

WHAT IS AN IDS?

- Helps to identify the events related to each phase of an attack:
  - Reconnaissance
  - Scanning
  - Privilege escalation
  - Gaining access
  - Covering tracks

TYPES OF IDS

ANOMALIES DETECTION

- They require an initial basis of behavior:
  - Finite time
  - Detection of devations
  - Learning phase
  - Identification of protocol violations:
    - Based on the RFCs
    - Requires a knowledge base

- They can not detect most attacks
  - Generally, they complement other solutions.

ANOMALIES DETECTION

- They can not determine the context of the attack
  - They identify anomalies, not attacks.
  - The impact is determined by the analyst.

SIGNATURE-BASED DETECTION

- It is based on the search of patterns in network traffic
  - Alerts are generated, as the presence of malicious traffic is identified.
  - Event traffic may not be stored.
- Most signatures look for patterns in the payload
  - Exploits, information leakage, character strings, etc.
- Some signatures are available on each vendor's site.

SIGNATURE-BASED DETECTION

- Signatures are also known as rules.
  - They are updated constantly
  - They use parameters specific to the detection engine:
    - Reserved words
    - Detection and analysis techniques
  - Some use something more generic, like regular expressions.
- Depending on the vendor, some can be open source

IDS OR IPS

INTRUSION DETECTION SYSTEM

INTRUSION DETECTION SYSTEM

- It was the security solution of the turn of the century
- Firewall light version: less security, less risk
- Firewall advanced version: deeper analysis
- Vulnerable to … being ignored
- Counterproductive to too many false positives

INTRUSION PREVENTION SYSTEM

INTRUSION PREVENTION SYSTEM

- The security tool of nowadays
- The logical union of detection and blocking
- Some variants are vulnerable to DoS, but with greater resistance than a firewall
- Detection and correction of problems in the network
- Adaptability against new threats
- It may be too much for some needs

ISSUES WITH THE IDS/IPS

- Complex patterns:
  - Increase in the use of resources
  - The complex is prone to errors

- The use of simple or general signatures.
  - False positives:
  - Identification of benign traffic as malicious

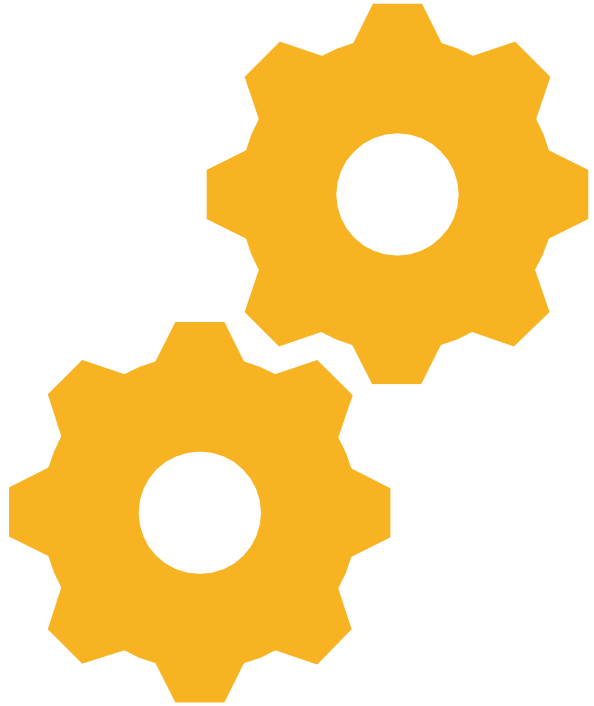
- False negatives:
  - A signature fails to throw an alert during an attack
  - There is no way to know if they have happened unless the attack has been successful

IDS AND IPS SOLUTIONS

- There are several existing solutions out there, including commercial and open source:
  - Bro (Open source)
  - Suricata (Open source)
  - Security Onion (Open source Linux distribution)
  - Snort (Open source)

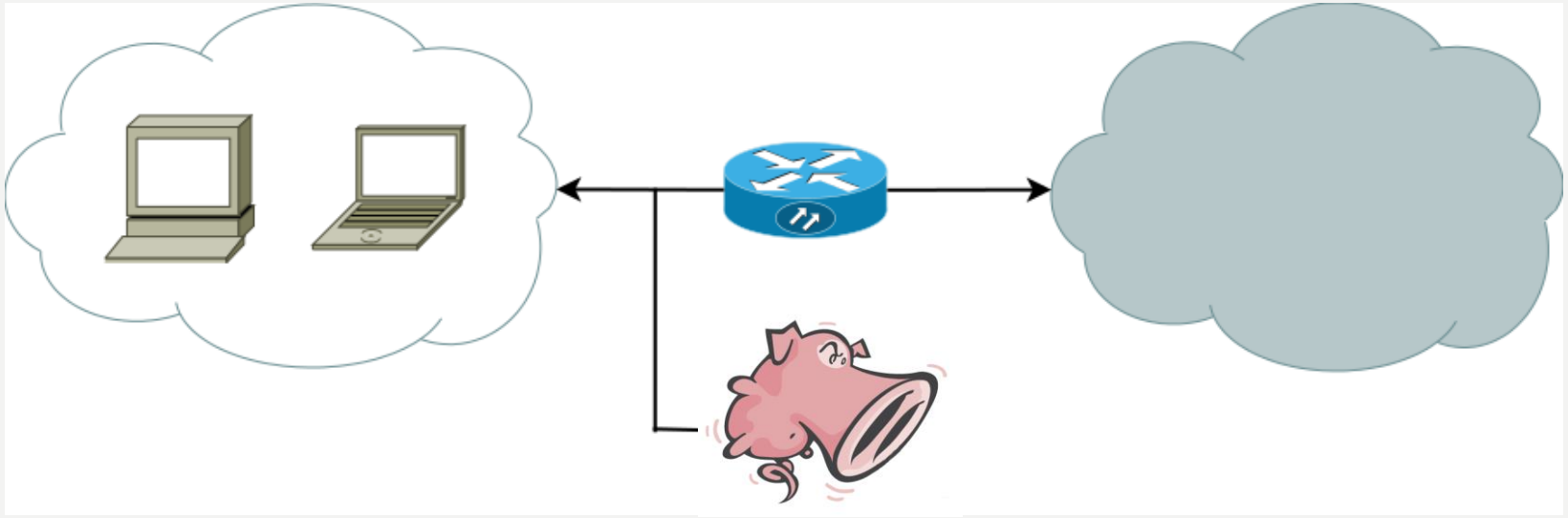
  - Cisco FirePower (commercial)
  - McAfee's Network Security Platform (commercial)
  - TippingPoint Threat Protection System (commercial)
  - FireEye Network Security (commercial)
  - Vectra's Cognito (Commercial)

DEPLOYING AN IDS: SNORT

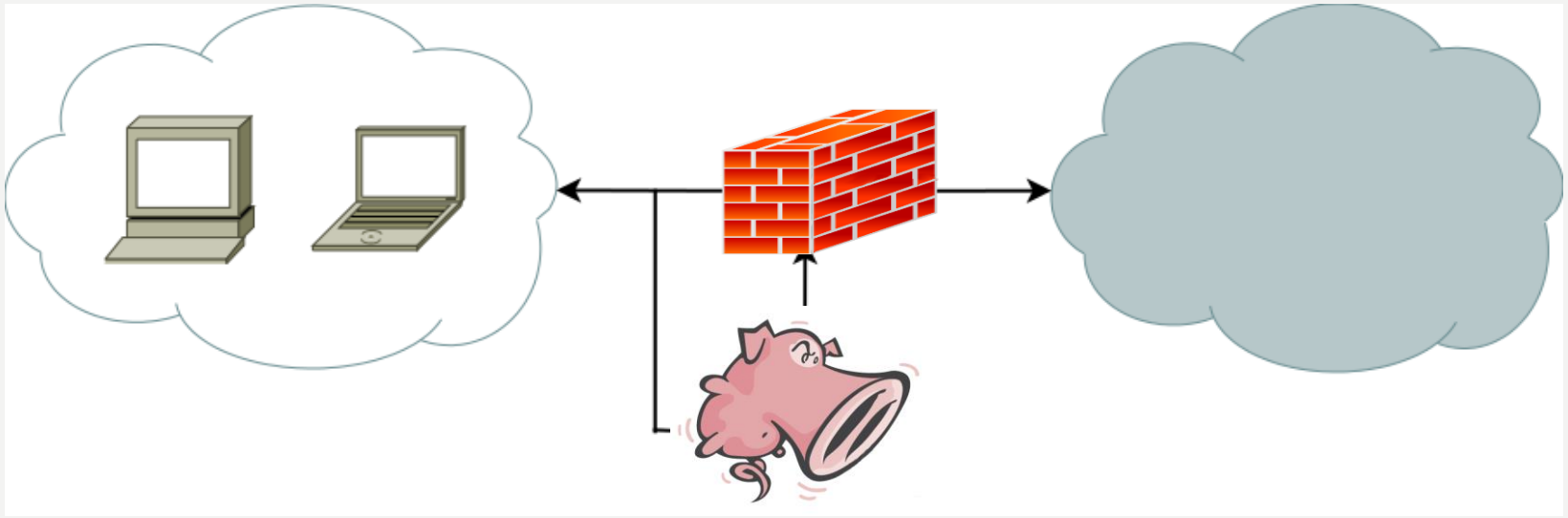


INTRO TO SNORT

SNORT AS AN IDS



SNORT AS AN IPS



STARTING WITH SNORT

- Snort is available for free download in: https://snort.org
- Can be installed from source code, (Linux and UNIX), using software package managers (Debian, Read Hat, etc) or with a Windows executable file.
- The latest stable version of Snort, until late June 2019 is 2.9.13
  - Additional required libraries are also available at the website
- The ruleset is also available for free download, however there are different types of rulesets:
  - Community
  - Registered
  - Subscriber

STARTING WITH SNORT

- Snort installation guides are available in the website along with other documents with setups and deployments made by the community of users.

- Other important resources in the website are:
  - The Snort manual
  - Snort rule infographic
  - Snort FAQ
  - Other ways of interaction and communication with the Snort user's community

HOW TO USE SNORT

- Snort can be used in three modes of execution:
  - Sniffer mode, which simply reads the packets off of the network and displays them for you in a continuous stream on the console (screen).
  - Packet Logger mode, which logs the packets to disk.
  - Network Intrusion Detection System (NIDS) mode, which performs detection and analysis on network traffic. This is the most complex and configurable mode.
  - IPS (extra)
  - The mode of execution depends on the options of the command line and the hardware.

SNIFFER MODE

- Capture, analyze and display the packages on the screen.

- Some parameters to use in the command line:
  - v: Enable verbose mode (shows more information)
  - i: Specify the network interface
  - d: Dumps the contents of the packets
  - e: Print the MAC addresses of the packets

SNIFFER MODE OUTPUT

10/25-17:36:20.895739 09:FF:00:9C:61:13 -> AA:00:04:00:0A:04 type:0x800 len:0x72

192.168.0.3:51006 -> 192.168.0.31:22 TCP TTL:64 TOS:0x10 ID:12471 IpLen:20 DgmLen:100 DF

AP Seq: 0xEAD4B262 Ack: 0x4EA52EE2 Win: 0x2000 TcpLen: 32

TCP Options (3) => NOP NOP TS: 272761746 4060028

```
B2 7E F8 20 36 E7 1B C6 B2 05 B9 A9 3F 0D 49 37        .~. 6.......?.I7
14 7C FC 02 61 4C 17 11 92 F7 CE 25 49 D1 F2 90        .|..aL.....%I...
6E E4 1B FF 9F 1B A9 42 31 9E 05 CC C7 A5 7B 33        n.......B1.....{3
…
```

PACKET LOGGER MODE

- Tells Snort to store the received packets into a file.

- The write options are:
  - l: Specify the path where the files will be stored
  - b:Snort stores the files in binary format (libpcap)

- Snort can also read PCAP files and compare the captured traffic against the detection rules (NIDS).

- You can use the switch r:
  - snort –r snort.log.##########
  - snort –r file.pcap

NIDS MODE

- It is the most important mode of operation of Snort.
- It uses rules for detecting malicious traffic.
- You can use additional tools to improve the detection capability.
- Monitor the network at all times to detect malicious activity.
- Use a configuration file:
  - snort -c /usr/snort/etc/snort.conf
  - snort -c /etc/snort/snort.conf
- It can be complex yet at the same time very powerful.

NIDS MODE: CONFIGURATION

- Two ways to do it:
  - Using the CLI and the available switches of Snort (Ninja CLI)
  - The Snort configuration file **snort.conf** (Easy recommended method)

- Outcomes
  - Custom analysis settings
  - Add the decoders, plugins and rules you need
  - Get the output you want

- It will depend on you, how you want to use it

- We will use the configuration file in this training

NIDS MODE: CONFIG FILE

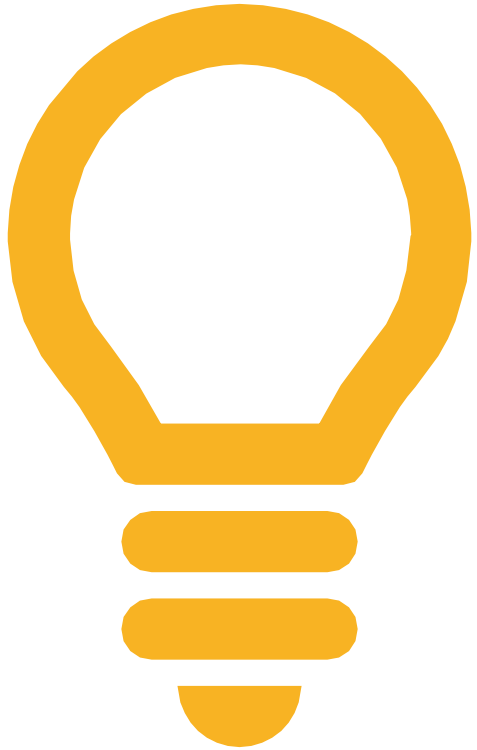
- You can configure:
  - Includes: Adds a a file with Snort configuration that will override the settings of the main file
    - include <path_to_file>
  - Variables: Define any value that suits your needs (syntax later)
    - var “name” “value”
    - ipvar “name” “value”
    - portvar “name” “value”
  - Configs: Allows to specify the values to the parameters of Snort’s CLI
    - For example: **config logdir: <dir>** is equivalent to **snort –l <dir>**

NIDS MODE: CONFIG FILE

- Preprocessors: Code that is run before the detection engine is called, but after the packet has been decoded
  - preprocessor <name>: <options>
- Output: They allow Snort to be much more flexible in the formatting and presentation of output to its users
  - output <name>: <options>

- Everything gets executed in the order of declaration in the file, keep this in mind if you want something to happen first.

CREATING DETECTION CONTENT FOR SNORT



CREATING DETECTION CONTENT FOR SNORT

- Rules are the heart of Snort's detection
- Define what Snort will search for in the network packets
- Define who and what is considered a network threat or suspicious activity
- Inspect the header, protocols and content of each packet
- They have an easy and simple to read and write syntax
- Let's refer to Snortology 101…

CREATING DETECTION CONTENT FOR SNORT

- Let's write rules for:
  - Malicious shell access and dropper
  - Ransomware C&C traffic
  - A false positive
  - Catching a vulnerability in a DHCP Server
  - A file that exploits a vulnerability

REFERENCES

- https://honeynet.org
- https://github.com/paralax/awesome-honeypots
- https://hub.docker.com/u/honeynet/
- http://manual.snort.org
- https://snort.org

THANK YOU!

MIGUEL ´MIKE´ BAUTISTA

@MIGUELRAULB
MIGUELRAULB@GMAIL.COM