

Manolis (Emmanouil Vasilomanolakis)

network security: threat detection

Course plan

- Lecture 1: Intro (**Manolis, Carsten**) 30.01
- Lecture 2: **Crypto** essentials (**Carsten**) 06.02
- Lecture 3: **Authentication** (**Manolis**), lab bootcamp (TAs) 13.02
- Lecture 4: **TLS** (**Manolis**) 20.02
- Lecture 5: **Threat detection** (**Manolis**) 27.02
- Lecture 6: Hacking Lab day (**TAs**) – blue team 05.03
- Lecture 7: **IoT security** (**Manolis**) 12.03
- Lecture 8: **WIFI security** (**Manolis**) 19.03
- Lecture 9: **Private communication** (**Carsten**) 02.04
- Lecture 10: **When everything fails** (**Manolis**) 09.04
- Lecture 11: Hacking Lab day (**TAs**) – red team 16.04
- Lecture 12: Guest lecture (OT security, **Ludwig**) 23.04
- Lecture 13: **Exam preps** (**Carsten, Manolis**) 30.04

Outline

- **Introduction**
- **Firewalls**
- **Intrusion detection**
- **Cyber-deception**
- **Lab exercises**

Threat detection lecture overview

- **Block traffic** going in:
 - firewalls
- **Detecting malicious traffic/entities** that bypassed the firewall:
 - Intrusion detection systems (IDS)
- **Blocking malicious traffic/entities** that bypassed the firewall:
 - Intrusion prevention systems (IPS)
- **Detecting malicious traffic/entities that bypassed everything:**
 - Cyber-deception
 - Most common practical tool: honeypots

Firewalls

Firewalls

- What is a **firewall**? (before we invented computers and computer networks)

Dictionary



firewall

/ˈfaɪəwɔːl/ 

noun

1. a wall or partition designed to inhibit or prevent the spread of fire.



Firewalls in a Computer Network

- What is a **firewall** in a computer network?
 - A **networking software/device** that protects a network by **monitoring** the packets that are coming **in and out** of the network and **filtering** unwanted traffic
 - Firewalls can be stand-alone computers, integrated in routers, end devices, etc
 - Firewalls implement a **security policy**
 - They enforce a set of **predetermined rules** that specify what traffic is allowed



Default Behaviour

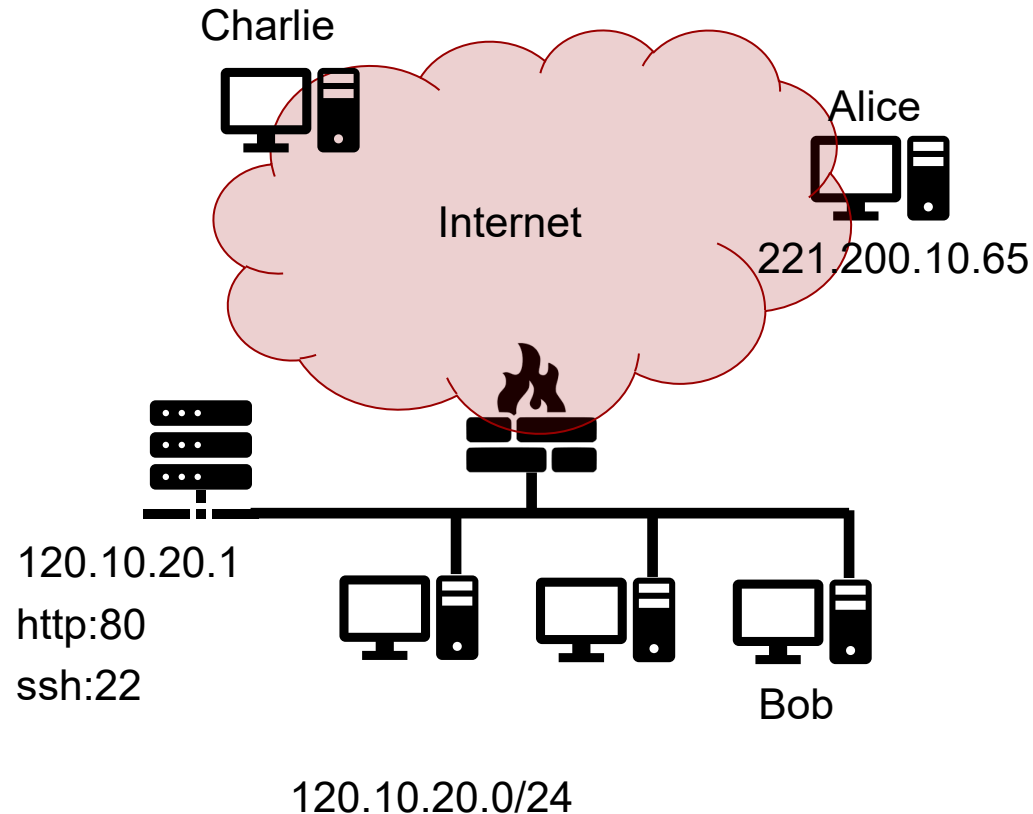
- **Default allow**
 - Allow traffic unless otherwise specified
 - More **user friendly**: users do not like restrictions
- **Default deny**
 - Deny traffic unless otherwise specified
 - More **secure**, it is easier to imagine all the possible things that are acceptable than the opposite

The 2 steps for configuring a firewall

- Step 1: Determine a particular **security policy**
 - The ideal policy never allows a single unauthorised packet but is **invisible** to legitimate users
- Step 2: Express this security policy in terms of **firewall rules**
 - Firewall rules are typically examined from top to bottom:
 - If rule matches, the specified action is performed
 - Otherwise, next rule is evaluated
 - If no rule matches, the default action is performed (e.g. default deny)

Example of a Packet Filtering Firewall

- External users can access the **web server** but they should not be able to log in using **SSH**
- Alice is the **administrator** of the server and needs **SSH access** from home



Source IP	Source Port	Destination IP	Destination Port	Action
*	*	120.10.20.1	80	Accept
221.200.10.65	*	120.10.20.1	22	Accept
*	*	*	*	Deny

Quiz:

- Can Charlie reach the **SSH** server of 120.10.20.1?
- Can Alice reach the **SSH** server of 120.10.20.1?
- Can Charlie reach the web server of 120.10.20.1?

IPTABLES

- Program to edit/**configure IP packet filter rules of the Linux kernel firewall**
- Written in C
- 1998, by Netfilter



Stateless vs Stateful Filtering

- **Stateless** packet filtering
 - Every packet is evaluated **individually**
 - If header information match rules, then specified policy applies
 - Fast and **efficient**, **no memory** requirements
 - All examples so far are examples of stateless filtering
- **Stateful** packet filtering
 - Examine packets in relation to previous packets
 - Firewall must keep memory state information from one packet to another
 - **Higher overhead**

Stateful Filtering Examples

- **DoS attacks**

- One failed HTTP request from the same IP is OK (SYN, SYN/ACK, No ACK)
- 1000 HTTP requests per second from the same IP is a DoS attack (SYN Flooding)
- Stateful firewalls can enforce **rate limit policies** (no more X packets per second)
- DDOS?

- **Port Scanning**

- A probe on few ports is OK (might be legitimate or an honest mistake)
- A probe on 1000 ports is port scanning by somebody that is potentially dangerous!
- Stateful firewalls can keep track and block unacceptable behaviour
- nmap has a **paranoid mode** (waits 5 minutes between probes!)

- **Port Knocking**

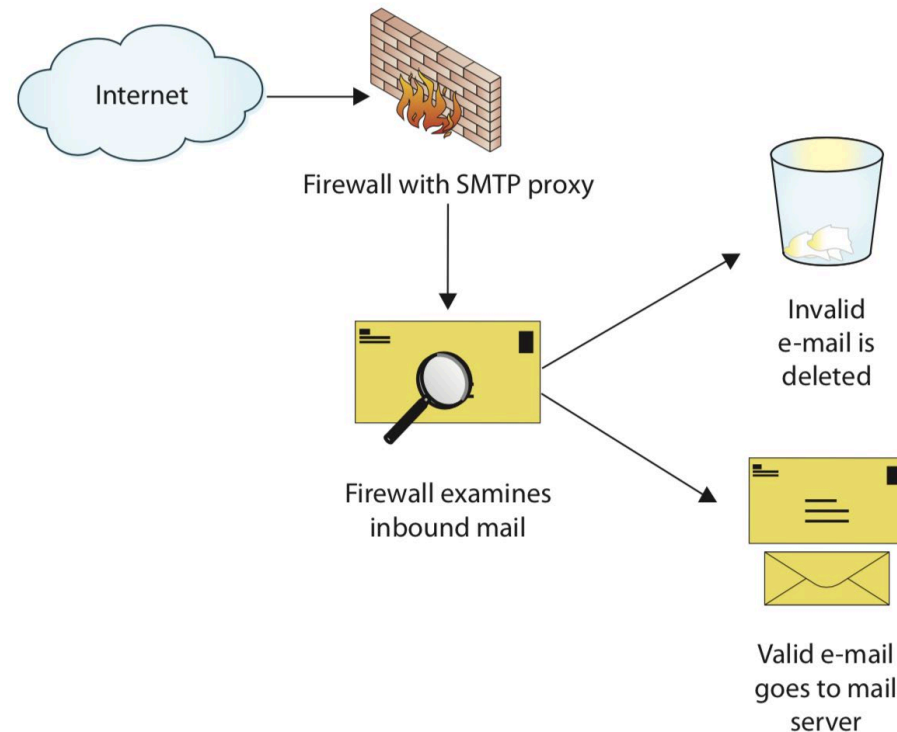
- Use TCP SYN on a series of ports as “password”
 - Accept packets after a series of 3 failed TCP SYN attempts on ports 1000, 2000, 3000

Port knocking demo video

- <https://www.youtube.com/watch?v=IBR3oLqGBj4>

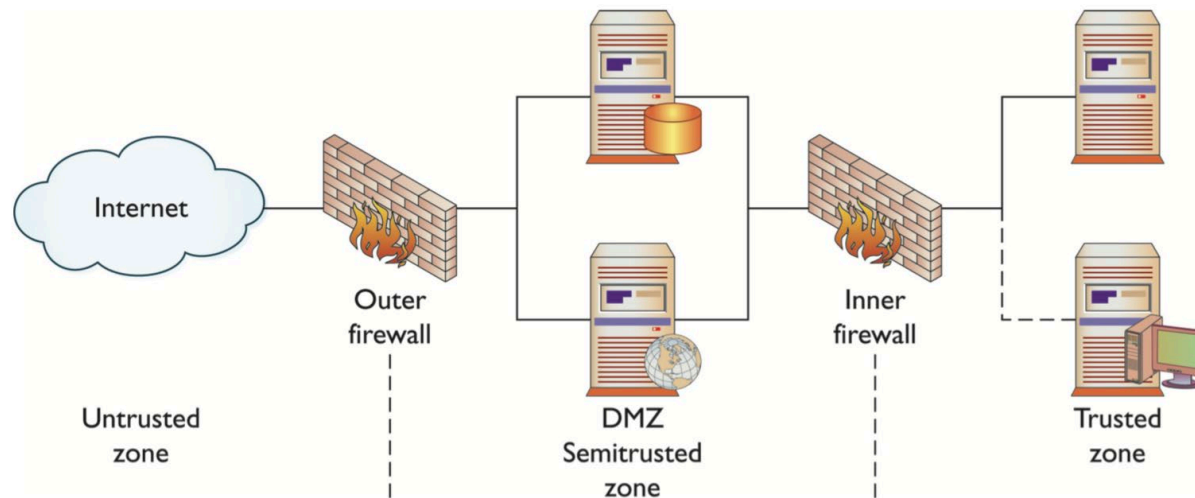
Packet Filtering vs Application Layer Firewalls

- **Packet Filtering Firewalls** enforce rules upon analysing the **headers** of the packets
- **Application Layer Firewalls** inspect both the headers and the **content**



Security Zones

- Remember: Firewalls are also useful to prevent the spread of a fire!
- DMZ (Demilitarised Zone)
 - Buffer zone between the wild Internet and the internal network
 - Public servers (e.g. organisation's web-server) are placed in the DMZ



Outline

- Introduction
- Firewalls
- **Intrusion detection**
- **Cyber-deception**
- **Lab exercises**

Intrusion detection

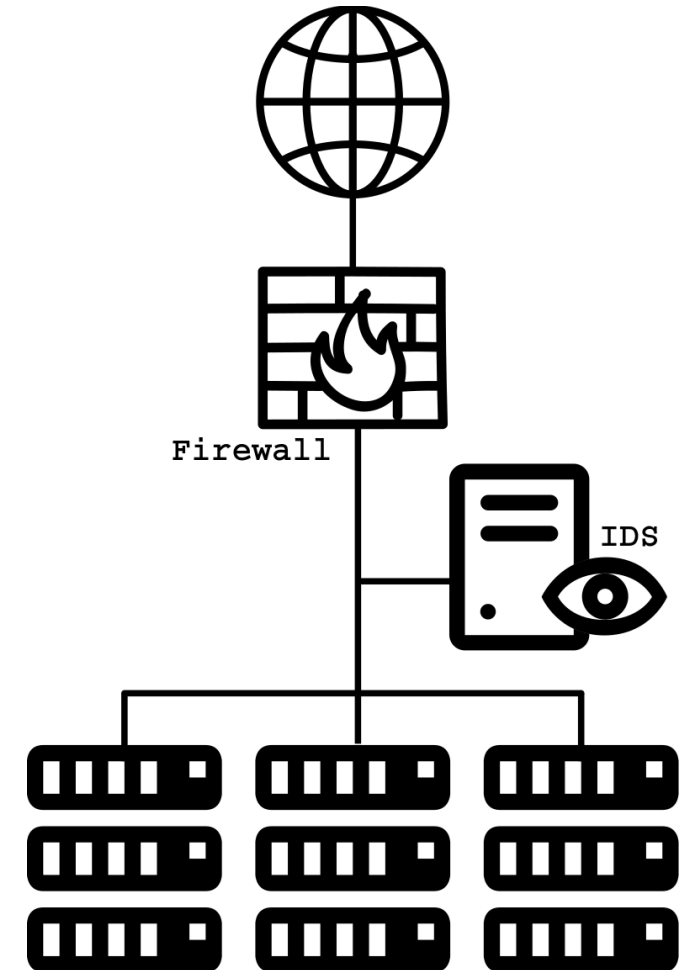
Intrusion Detection Systems (IDS)

Firewalls:

- **Filter** network traffic based on a **security policy**
- Sit **between** private networks and the Internet

Intrusion Detection Systems:

- **Monitor** activity in a private environment
- **Spectate**; they do not interfere with network traffic
- **Notify** administrators upon detection of an intrusion



IDSs are like a smoke detector:

if a “fire” spreads past the firewall, the smoke detector will kick in and alert you

Intrusion Prevention System (IPS)

If we can detect malicious activity, why not act on it?

Intrusion Prevention Systems **take action** when detecting a threat

- IPS is an **extension** of IDS
- Distinction is blurry (many IDSs are IPSs to some extent)

Intrusion Detection System (IDS)

- Definition: a system (software or hardware) that *monitors a host or a network for signs of intrusions, manifested by malicious behavior or security policy violations.*



Classifications of IDSs

- Based on the placement (location) of the IDS:
 - Host-based
 - Network-based
 - Other: wireless, SDN (software-defined network), etc.
- Based on the detection method:
 - Signature-based
 - Anomaly-based
 - Hybrid

Host-based IDSs [1/2]

- OS/Application-level monitoring
 - Log all relevant system events
 - Monitoring of system calls
 - Audit information may include
 - File R/W operations (e.g., file modifications: /etc/passwd)
 - Authentication mechanisms (e.g., login attempts: brute force attacks)

Host-based IDSs [2/2]

- Advantages
 - Very detailed/in-depth monitoring/analysis
 - Ideally, majority of attacks can be identified
 - No need for additional hardware
- Disadvantages
 - Complete isolation (“see the forest for the trees”)
 - OS/Application specific
 - Expensive to maintain

Example: Lynis

- Lynis is something like a host-based (static) IDS/scanner
 - Actually, more of a system hardening and auditing tool
 - **But** very similar to the logic of traditional host-based IDSs

```
[+] Users, Groups and Authentication
-----
- Search administrator accounts... [ OK ]
- Checking UIDs... [ OK ]
- Checking chkgrp tool... [ FOUND ]
- Consistency check /etc/group file... [ OK ]
- Test group files (grpck)... [ OK ]
- Checking login shells... [ WARNING ]
- Checking non unique group ID's... [ OK ]
- Checking non unique group names... [ OK ]
- Checking LDAP authentication support [ NOT ENABLED ]
- Check /etc/sudoers file [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]

[+] Shells
-----
- Checking console TTYs... [ WARNING ]
- Checking shells from /etc/shells...
  Result: found 6 shells (valid shells: 6).

[ Press [ENTER] to continue, or [CTRL]+C to stop ]

[+] File systems
-----
- [FreeBSD] Querying UFS mount points (fstab)... [ OK ]
- Query swap partitions (fstab)... [ OK ]
- Testing swap partitions... [ OK ]
- Checking for old files in /tmp... [ WARNING ]
- Checking /tmp sticky bit... [ OK ]
```

Classifications of IDSs

- Based on the placement (location) of the IDS:
 - Host-based
 - **Network-based**
 - Other: wireless, SDN (software-defined network), etc.
- Based on the detection method:
 - Signature-based
 - Anomaly-based
 - Hybrid

Network-based IDSs

- Network-level monitoring
 - Placement in strategic points (usually behind a firewall)
 - Monitor all traffic (promiscuous interface)
- Advantages
 - “Big picture” can be observed
 - Independent of the operating system
- Disadvantages
 - Analysis might not be as in-depth as the host-based IDSs
 - Scalability: what happens when the monitored network is very large?

Classifications of IDSs

- Based on the placement (location) of the IDS:
 - Host-based
 - Network-based
 - Other: wireless, SDN (software-defined network), etc.
- Based on the detection method:
 - **Signature-based**
 - Anomaly-based
 - Hybrid

Signature-based (or misuse detection) IDSs [1/5]

- Similar to an antivirus program
- Search for known patterns of malicious activity

```
alert tcp any any <> any  
[443,465,563,636,695,898,989,990,992,993,994,995,2083,2087,2096,2484,8443,8883,9091]  
(content:"|18 03 00|"; depth: 3; content:"|01|"; distance: 2; within: 1;  
content:"|00|"; within: 1; msg: "SSLv3 Malicious Heartbleed Request V2";  
sid: 1;)
```

```
alert tcp any any <> any  
[443,465,563,636,695,898,989,990,992,993,994,995,2083,2087,2096,2484,8443,8883,9091]  
(content:"|18 03 01|"; depth: 3; content:"|01|"; distance: 2; within: 1;  
content:"|00|"; within: 1; msg: "TLSv1 Malicious Heartbleed Request V2";  
sid: 2;)
```

```
alert tcp any any <> any  
[443,465,563,636,695,898,989,990,992,993,994,995,2083,2087,2096,2484,8443,8883,9091]  
(content:"|18 03 02|"; depth: 3; content:"|01|"; distance: 2; within: 1;  
content:"|00|"; within: 1; msg: "TLSv1.1 Malicious Heartbleed Request V2";  
sid: 3;)
```



Signature-based IDSs [2/5]

- Advantages:
 - Easy deployment
 - Low overhead (low alarm rates, low maintenance)
 - Accuracy
 - Mature systems
 - The majority of real-world deployed IDSs are in this class
 - Many different systems exist

Signature-based IDSs [3/5]

- Disadvantages:
 - Who generates the signatures?
 - Detection techniques are sometimes simplistic
 - Cannot detect unknown attacks (e.g., 0-day exploits)
 - Packet analysis a major bottleneck

WannaCry Snort coverage

Lots of news out there this evening about a new Ransomware with auto-propagation ability. Please see our Talos blog post here:
<http://blog.talosintelligence.com/2017/05/wannacry.html>

We have Snort coverage available in the form of rules:
42329-42332, 42340, 41978

This coverage is available in our Snort Subscriber Rule Set.

In order to subscribe now to Talos's newest rule detection functionality, you can subscribe for as low as \$29 US dollars a year for personal users, be sure and see our business pricing as well at https://snort.org/products#rule_subscriptions.

Signature-based IDSs [4/5]: Snort

Snort IDS

- Open source
- Network-based
- Signature-based
- Packet sniffer
- Packet logger



Signature-based IDSs [5/5]: Snort rule example

Rule Header	<code>alert tcp \$EXTERNAL_NET \$HTTP_PORTS -> \$HOME_NET any</code>
Message	<code>msg: "BROWSER-IE Microsoft Internet Explorer CacheSize exploit attempt";</code>
Flow	<code>flow: to_client,established;</code>
Detection	<code>file_data; content:"recordset"; offset:14; depth:9; content:".CacheSize"; distance:0; within:100; pcree:"/CacheSize\s*=\s*/"; byte_test:10,>,0x3fffffff,0,relative,string;</code>
Metadata	<code>policy max-detect-ips drop, service http;</code>
References	<code>reference:cve,2016-8077;</code>
Classification	<code>classtype: attempted-user;</code>
Signature ID	<code>sid:65535;rev:1;</code>

Other network/signature-based IDSs

- **Zeek** (formally known as Bro)



- Extremely **powerful**
- Signatures **but** also statistics/analytics/anomaly
- Uses its own scripting language (you can play with it @:
<http://try.zeek.org/#/?example=hello>)
- Rather complicated to learn/deploy

- **Suricata**

- Signature-based
- Closer to Snort than Zeek
- High performance
- Can make use of Snort rules
- (relatively) easy to use



Classifications of IDSs

- Based on the placement (location) of the IDS:
 - Host-based
 - Network-based
 - Other: wireless, SDN (software-defined network), etc.
- Based on the detection method:
 - Signature-based
 - **Anomaly-based**
 - Hybrid

Anomaly-based IDSs [1/2]

- Analyze network/system
- Generate a normality model
 - Obvious usage of ML
- Examine network for anomalies
 - Patterns that do not conform to the expected behavior
 - Examples:
 - Credit card fraud
 - Heavy unexpected network traffic

Anomaly-based IDSs [2/2]

- Advantages:
 - Can detect unknown attacks (basically anything that is considered an anomaly)
- Disadvantages:
 - An anomaly is not always an attack
 - Requires the IDS to learn the “normal” state of the network
 - Sometimes more interesting for research than actual deployment
 - Might introduce large numbers of false positives!

Classifications of IDSs

- Based on the placement (location) of the IDS:
 - Host-based
 - Network-based
 - Other: wireless, SDN (software-defined network), etc.
- Based on the detection method:
 - Signature-based
 - Anomaly-based
 - **Hybrid**

Hybrid IDSs

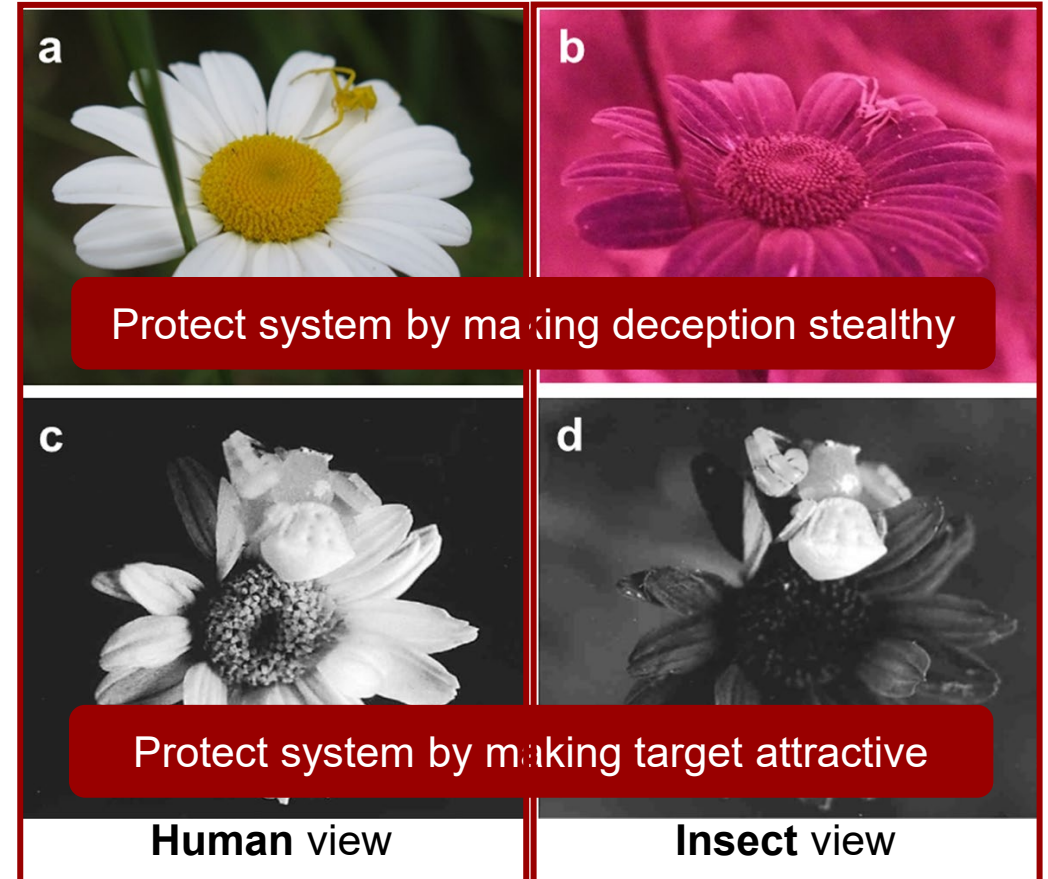
- Combine both signature-based and anomaly-based detection algorithms
- Advantages:
 - Combination of advantages from both systems
- Disadvantages/challenges:
 - If not completely automated: can introduce overhead for the administrators
 - How to combine the detectors?
 - Which anomaly-detection algorithms should be used?
 - How to train them?

Outline

- Introduction
- Firewalls
- Intrusion detection
- **Cyber-deception**
- **Lab exercises**

Deception: from nature to the military

- **Nature:**
 - **Camouflage**
 - Parasitic signalling
 - **Mimicry**
 - Defense/predators
- **Military:**
 - Camouflage
 - Sabotage
 - Etc.



Mimicry phenomena share two core characteristics

“resemblance to a **model** & **received deception**”

[Enrique Font, *Mimicry, Camouflage & Perceptual Exploitation:*

the Evolution of Deception in Nature, 2018]

What is cyber-deception?

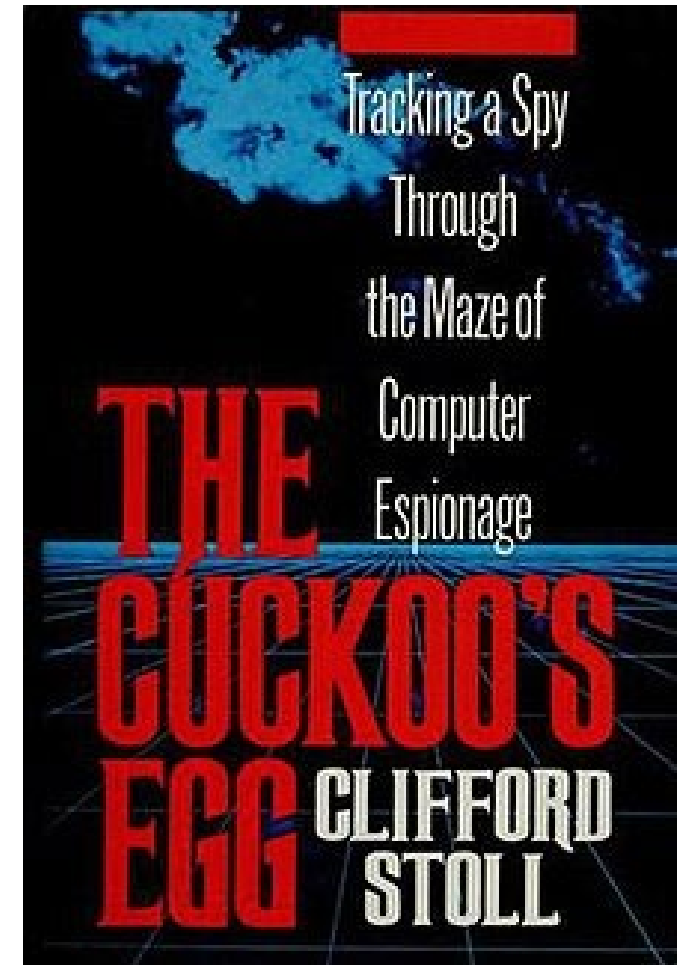
- **Umbrella term** that:
 - Describes some sort of deception
 - Includes technologies such as:
 - Honeypots
 - Honeytokens (aka honeywords)
 - Decoys
 - Breadcrumbs
 - Moving target defense
 - Etc.

Deception Taxonomies (Bell and Whaley 1991 & Qassrawi and Hongli 2010)

- **Masking**
 - hiding things in the background
 - e.g., concealing the monitoring of users by the honeypot by modifying the operating system to hide its traces
- **Repackaging**
 - hiding something as something else
 - e.g., embedding of attack-thwarting software within innocent utilities of a computer's operating system
- **Dazzling**
 - hiding something by having it overshadowed by something else
 - e.g., sending many error messages to an attacker when they try to do something malicious
- **Mimicking**
 - imitating aspects of something else
 - e.g., construction of a fake file directory for a honeypot that looks like the file system of a busy user, with the goal of helping convince the attacker it is not a honeypot
- **Inventing**
 - creating new, often “fake”, objects to interest the deceive
 - e.g., a piece of software left in a honeypot for attackers to download that reports their personal data to authorities when run
- **Decoying**
 - using diversions unrelated to the object of interest
 - e.g., planting passwords of honeypot websites to encourage attackers to log in there

Honeypots and cyber deception

- First known case of a honeypot in 1986(!)
 - Clifford Stoll
 - 9 seconds of unpaid computer time at the Lawrence Berkeley National Laboratory
 - Created a honeypot by putting together terminals, and inserting fake accounts and documents
 - (successfully found the hacker, who happened to be a KGB spy)



Motivation

- Traditional cyber-defense mechanisms are okay but...
 - Attackers are expecting all these mechanisms.
 - What about more aggressive/active techniques?
 - How can we learn about novel attack trends?
 - How can we observe attackers' behavior?

Introduction [1/2]

–Definition: **“A security resource whose value lies in being probed, attacked or compromised”**

- Doesn't have to be a (real) system
- We actually want the resource to get compromised!
- Certainly **not** a standalone security mechanism
 - A honeypot cannot take the place of an IDS or firewall

Introduction [2/2]

But why honeypots?

- (Almost) no false-positives
- Understand how attackers work
- Research/industry purposes:
 - » Malware collection and analysis
 - » Botnet mitigation
- Reducing the available attack surface/early warning system
- Because they are **FUN!**
- Identifying attack trends (see next slide)

Honeypots can highlight trends



Port	Protocol/Service	Number of Attacks
135	RPC	24,667
139	NetBIOS	20,249
23	Telnet	11,058
80	HTTP	10,735
445	SMB	9,294

"Lastly, we identified that a number of the attacks that were targeting Telnet were conducted by insecure infected embedded devices, e.g., IP web-cams."

**A honeypot-driven cyber incident monitor:
lessons learned and steps ahead,**

Vasilomanolakis et al., SIN 2015

Honeypot Classifications

- **Interaction-level classification**

- Low-interaction
- Medium-interaction
- High-interaction

- **Purpose**

- Generic
- Malware collectors
- Protocol-specific
- Technology-specific

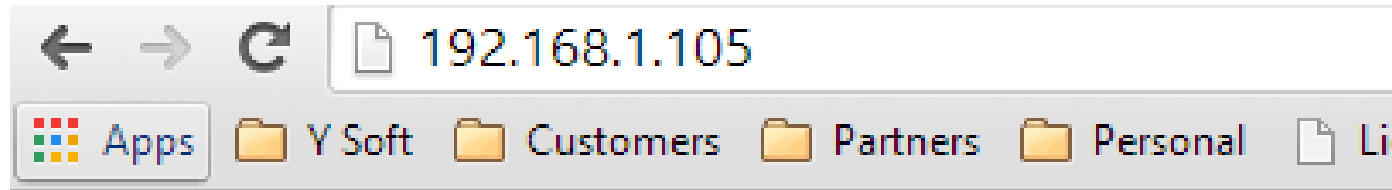
- **Placement**

- Production
- Research



Interaction-level classification

- **Low** interaction: simulate network operations (usually at the TCP/IP stack)



Interaction-level classification

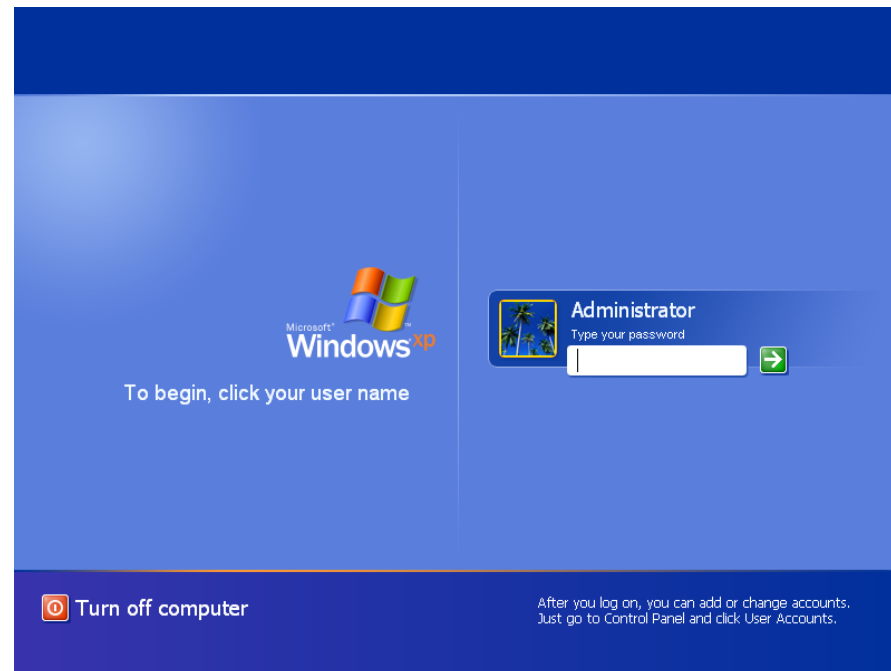
- **Medium** interaction: simulate network operations with more “sophisticated” ways

```
db2:~# w
 06:42:55 up 6 days,  8:04,  1 user,  load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
root      pts/0    95.141.37.19   06:42    0.00s  0.00s  0.00s w
db2:~# unam e-a
bash: unam: command not found
db2:~# uname -a
Linux db2 2.6.26-2-686 #1 SMP Wed Nov 4 20:45:37 UTC 2009 i686 GNU/Linux
db2:~#
```

- Questionable class:
 - sometimes synonym to low interaction
 - sometimes not (e.g. container-based emulation)

Interaction-level classification

- **High** interaction: real systems or VMs
 - Full functionality/interaction
 - Very expensive to maintain



Purpose-based classification [1/2]

- **Generic honeypots**

- All-around, general purpose style honeypots
- Not easy to develop such (realistic-looking) honeypots
- Example: honeyd



- **Malware collectors**

- Main purpose is the collection of malware binary files
- Honeypot provides enough interaction to receive the binary
- Very useful for botnet monitoring
- Example: Dionaea



Purpose-based classification [2/2]

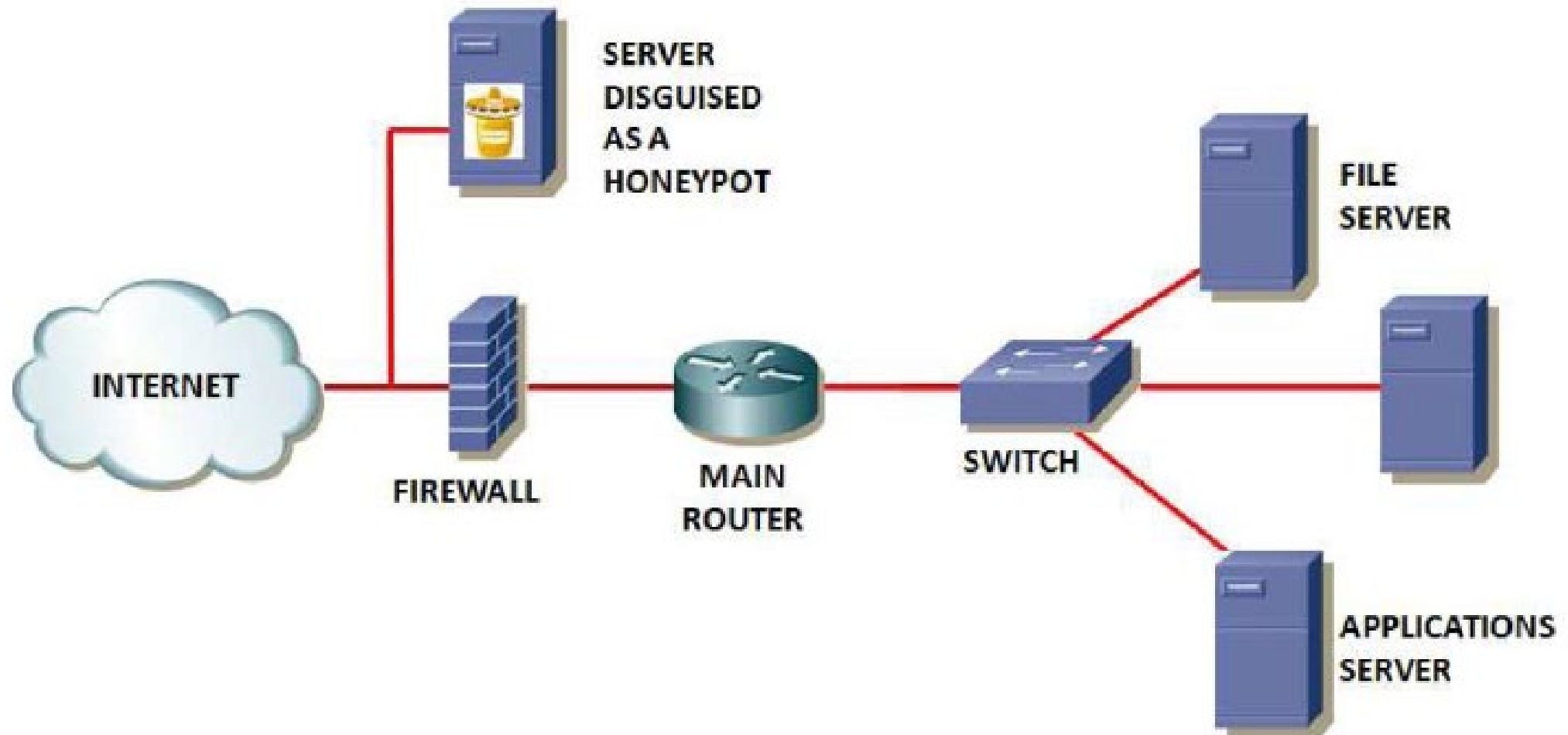
- **Protocol specific honeypot**

- Emulation of a specific protocol of interest
- Easier to develop
- Higher interaction level can be provided
- Example: Kippo SSH honeypot

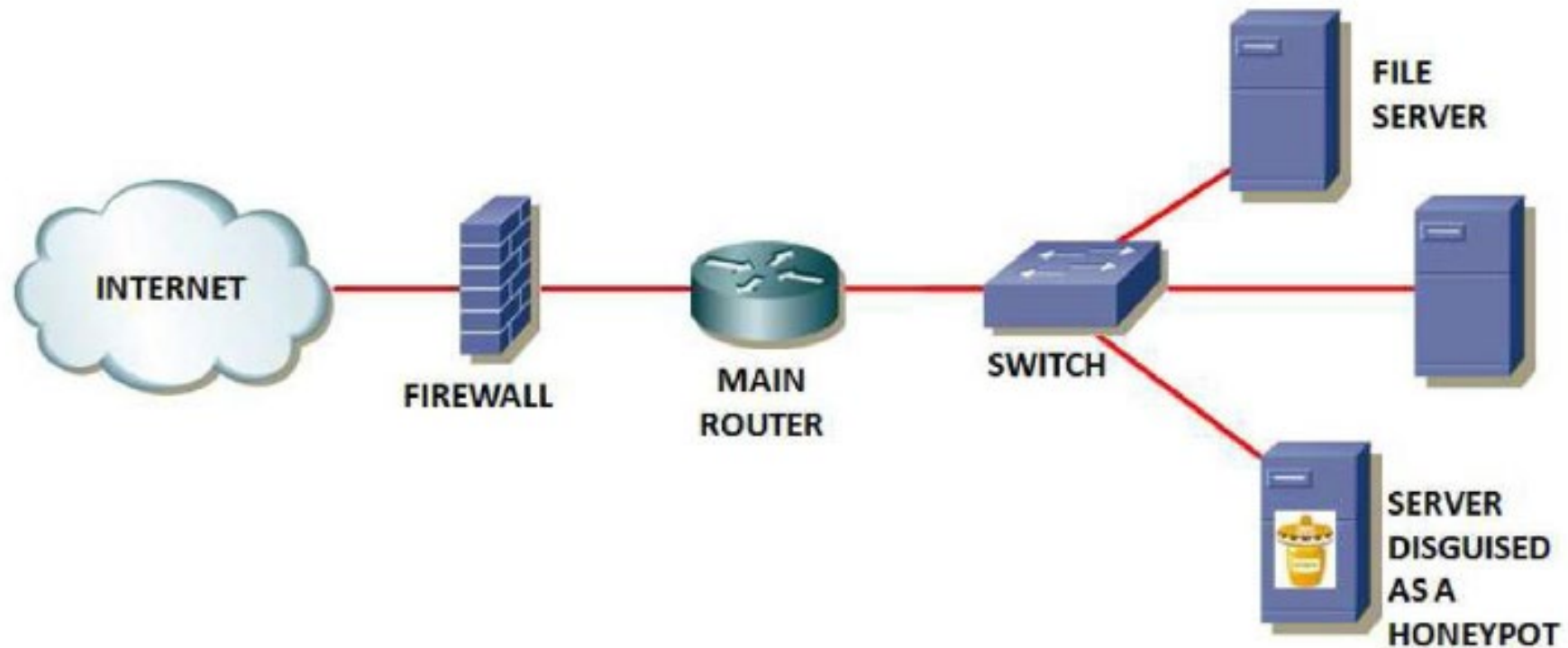
- **Technology-specific honeypot**

- Emulation of a specific technology
- A “technology” is usually realized as a collection of protocols
- Example: IoTPot – Internet of Things honeypot

Placement Classification Architectures [1/2]



Placement Classification Architectures [2/2]



Many honeypots...

- A Survey on Honeypot Software and Data Analysis
 - Nawrocki et al., 2016
 - Lists approximately **50 low and high interaction honeypots**
- **However:**
 - **Many** projects **abandoned**
 - This creates vulnerabilities (we will return to this later)
- We will discuss some of the most used/interesting honeypots next!

An all-around malware collector

DIONAEA HONEYPOT

Dionaea

- Low Interaction honeypot for collecting malware
- Uses Libemu a "*library written in C offering basic x86 emulation and shellcode detection using GetPC heuristics*"
- Basic protocol simulated: SMB (port 445)
- Others: HTTP, HTTPS, FTP, TFTP, MSSQL and SIP (VOIP)
- Also supports IPv6



Dionaea muscipula

Dionaea

- **Great** honeypot for capturing automatically spreading malware
- Malware files: stored locally or/and sent to 3rd party entities
(CWSandbox, Norman Sandbox, Anubis, VirusTotal)

Dionaea

- Disadvantage: getting old
- Most versions can be easily detected via Nmap:

```
C:\Users\Mert>nmap [redacted] -sV ←
Starting Nmap 7.12 ( https://nmap.org ) at 2017-07-12 19:00 Turkey Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
servers with --dns-servers
Nmap scan report for [redacted]
Host is up (0.019s latency).
Not shown: 990 closed ports
PORT      STATE    SERVICE    VERSION
21/tcp    open    ftp        Dionaea honeypot ftpd
25/tcp    filtered smtp
42/tcp    open    nameserver?
135/tcp   open    msrpc?
445/tcp   open    microsoft-ds Dionaea honeypot smb
1433/tcp  open    ms-sql-s   Dionaea honeypot MS-SQL server
1720/tcp  filtered h323q931
3306/tcp  open    mysql      MySQL 5.0.54
5060/tcp  open    sip        (SIP end point; Status: 200 OK)
5061/tcp  open    ssl/sip    (SIP end point; Status: 200 OK)
```

The ultimate SSH honeypot

~~KIPPO~~/COWRIE HONEYPOT

Kippo (and its successor “*Cowrie*”)

- Low interaction SSH/Telnet honeypot
- Features:
 - Presenting a fake (but “functional”) system to the attacker (resembling a Debian installation)
 - Attacker can download his tools through *wget*, and they are saved for later inspection
 - Session logs are stored in an UML- compatible format for easy replay with original timings
- Easy to deploy
- Harder to attract attackers!

HosTaGe - Overview

- Lightweight, low-interaction honeypot for (rooted) mobile devices
- 15,000++ lines of code (JAVA)
- Open source
- Available in Play Store
- Emulates several protocols
 - HTTPS, FTP, MySQL, SIP, SSH,...
- Additionally
 - Support for **many major ICS/IoT** protocols
 - Detection of multi-stage attacks
 - Signature generation

HosTaGe – Protocol Emulation

- **AMQP**
- **COAP**
- **FTP**
- **HTTP/HTTPS**
- **MySQL**
- **MQTT**
- **Modbus**
- **S7comm**
 - Propriety protocol utilized in PLCs of the Siemens S7-300/400 family
- **SNMP**
- **SIP**
- **SMB**
- **SSH**
- **SMTP**
- **Telnet**



ICS/SCADA

CONPOT HONEYPOT

Conpot

- Low interaction **Industrial Control System (ICS)** honeypot
- Released in May 2013
- Multiple **protocols**:
 - HTTP, MODBUS, SNMP, BACnet, IPMI, S7comm
- Many **configurations** offered:
 - Siemens SIMATIC S7-200 PLC, Guardian AST tankmonitoring system, Kamstrup 382, etc.
- Usage of **XML** templates to define hardware devices

```
# conpot --template default
```

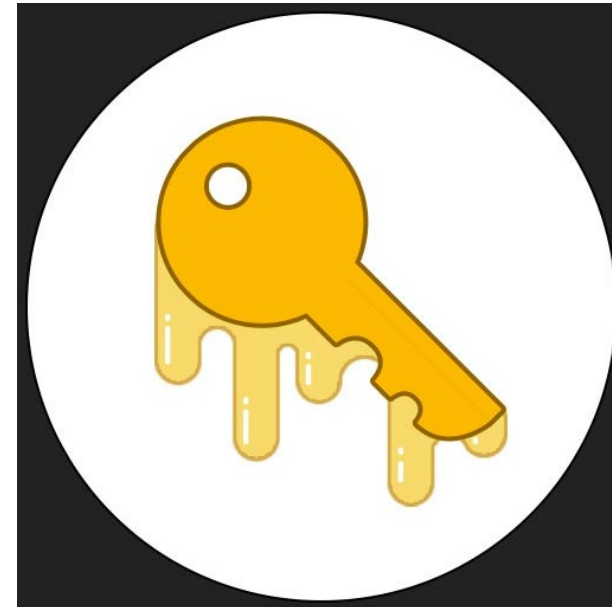


```
Version 0.6.0  
MushMush Foundation
```

```
2018-08-09 19:13:15,085 Initializing Virtual File System at ConpotTempFS/___conpot__ootc_k3j. Source specified : tar:
2018-08-09 19:13:15,100 Please wait while the system copies all specified files
2018-08-09 19:13:15,172 Fetched x.x.x.x as external ip.
2018-08-09 19:13:15,175 Found and enabled ('modbus', <conpot.protocols.modbus.modbus_server.ModbusServer object at 0x7f1af5a
2018-08-09 19:13:15,177 Found and enabled ('s7comm', <conpot.protocols.s7comm.s7_server.S7Server object at 0x7f1af5a
2018-08-09 19:13:15,178 Found and enabled ('http', <conpot.protocols.http.web_server.HTTPServer object at 0x7f1af4fc
2018-08-09 19:13:15,179 Found and enabled ('snmp', <conpot.protocols.snmp.snmp_server.SNMPServer object at 0x7f1af4f
2018-08-09 19:13:15,181 Found and enabled ('bacnet', <conpot.protocols.bacnet.bacnet_server.BacnetServer object at 0
2018-08-09 19:13:15,182 Found and enabled ('ipmi', <conpot.protocols.ipmi.ipmi_server.IpmiServer object at 0x7f1af5a
2018-08-09 19:13:15,185 Found and enabled ('enip', <conpot.protocols.enip.enip_server.EnipServer object at 0x7f1af5a
2018-08-09 19:13:15,199 Found and enabled ('ftp', <conpot.protocols.ftp.ftp_server.FTPServer object at 0x7f1af4fcec1
2018-08-09 19:13:15,206 Found and enabled ('tftp', <conpot.protocols.tftp.tftp_server.TftpServer object at 0x7f1af4f
2018-08-09 19:13:15,206 No proxy template found. Service will remain unconfigured/stopped.
2018-08-09 19:13:15,206 Modbus server started on: ('0.0.0.0', 5020)
2018-08-09 19:13:15,206 S7Comm server started on: ('0.0.0.0', 10201)
2018-08-09 19:13:15,207 HTTP server started on: ('0.0.0.0', 8800)
2018-08-09 19:13:15,402 SNMP server started on: ('0.0.0.0', 16100)
2018-08-09 19:13:15,403 Bacnet server started on: ('0.0.0.0', 47808)
2018-08-09 19:13:15,403 IPMI server started on: ('0.0.0.0', 6230)
2018-08-09 19:13:15,403 handle server PID [23183] running on ('0.0.0.0', 44818)
2018-08-09 19:13:15,404 handle server PID [23183] responding to external done/disable signal in object 1397536723090
2018-08-09 19:13:15,404 FTP server started on: ('0.0.0.0', 2121)
2018-08-09 19:13:15,404 Starting TFTP server at ('0.0.0.0', 6969)
```

Honeytokens

- A honeypot is not always a (fully functional) system
- Honeytokens (aka *honeywords*) are such an example
- Honeytokens can be:
 - A URL
 - A username/password
 - An email account
 - A file/folder
 - ...



Example case: canarytokens



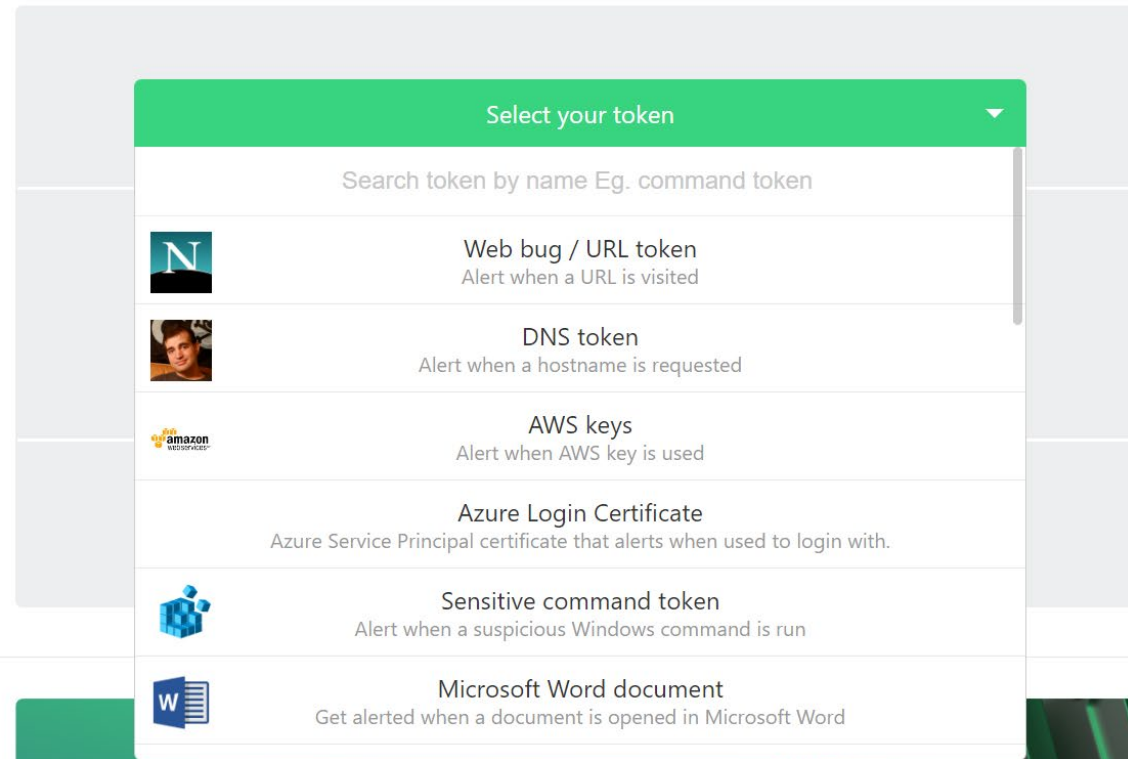
Select your token

Provide an email address or webhook URL (or both space separated)

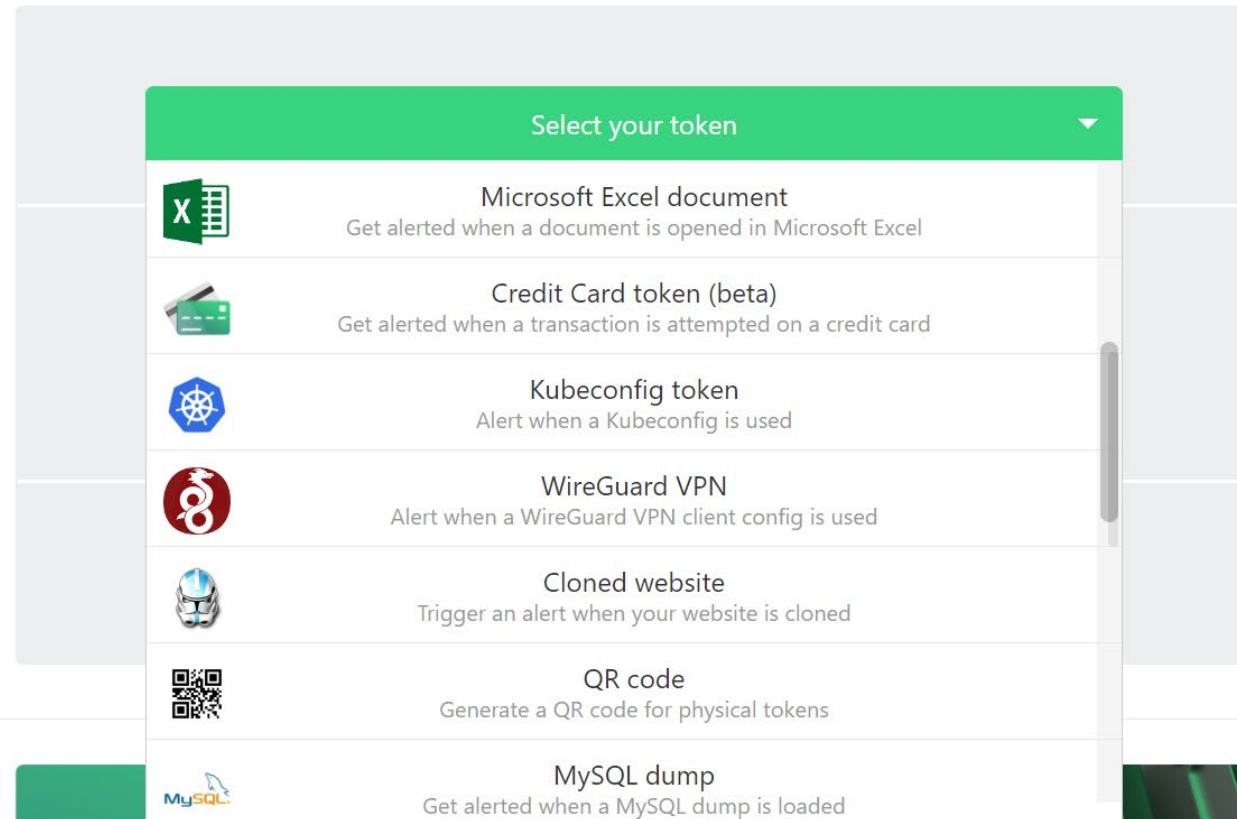
Reminder note when this token is triggered.

Fill in the fields above

Example case: canarytokens



Example case: canarytokens



Example case: canarytokens



Microsoft Word document ▼

emmva@dtu.dk

Current topics exams secret

Create my Canarytoken

Example case: canarytokens






Your MS Word token is active!

Download your MS Word file

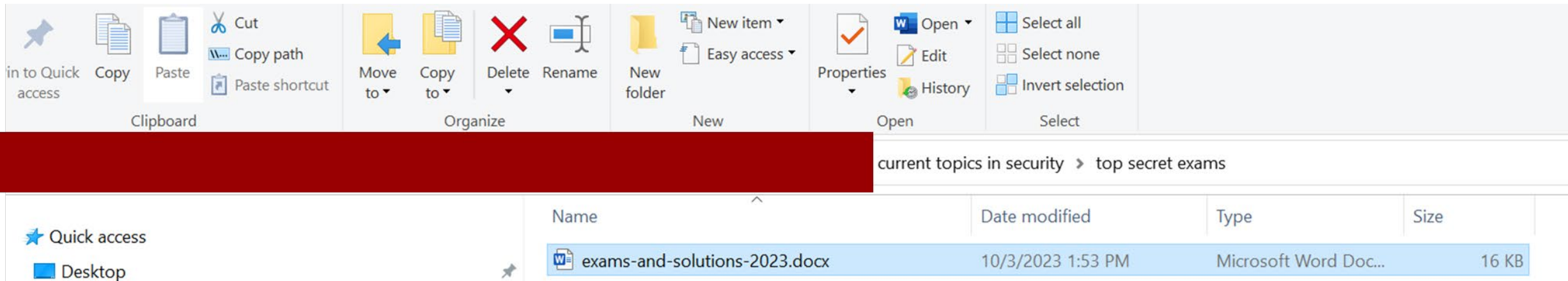
You'll get an alert whenever this document is opened in Microsoft Office, on Windows or Mac OS.

You can rename the document without affecting its operation.

Ideas for use:

-  Drop the file on a Windows network share.
-  Leave the file on a web server in an inaccessible directory, to detect webserver breaches.
-  Attach to an email with a tempting Subject line.

Example case: canarytokens




The screenshot displays the Windows File Explorer interface. The ribbon at the top includes the following tabs and options:

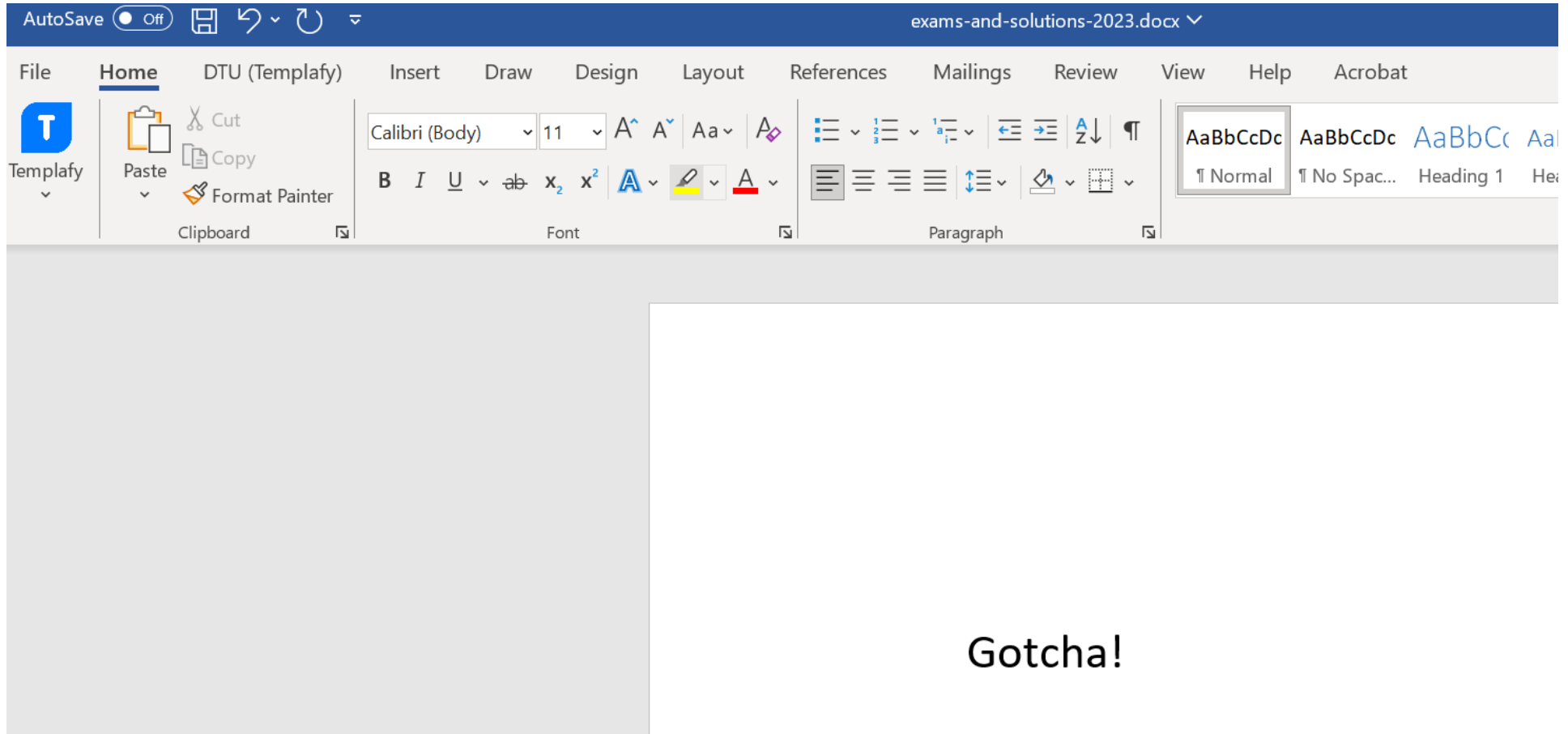
- Clipboard:** Pin to Quick access, Copy, Paste, Cut, Copy path, Paste shortcut.
- Organize:** Move to, Copy to, Delete, Rename.
- New:** New folder, New item, Easy access.
- Open:** Properties, Open, Edit, History.
- Select:** Select all, Select none, Invert selection.

A red bar obscures the ribbon area. Below the ribbon, the address bar shows the current location: [current topics in security](#) > [top secret exams](#).

The file list shows the following file selected:

Name	Date modified	Type	Size
 exams-and-solutions-2023.docx	10/3/2023 1:53 PM	Microsoft Word Doc...	16 KB

Example case: canarytokens



Example case: canarytokens

Canarytoken triggered

ALERT

An MS Word Canarytoken has been triggered by the Source IP 19[REDACTED]

Basic Details:

Channel	HTTP
Time	2023-10-03 11:56:11.094930
Canarytoken	h7qyk7u9tnwi8vf[REDACTED]
Token reminder	Current topics exams secret
Token type	MS Word
Source IP	19[REDACTED]
User-agent	Mozilla/4.0 (compatible; ms-office; MSOffice 16)

Canarytoken Management Details:

Manage this Canarytoken here
More info on this token here

Powered by: [Thinkst Canary](#)

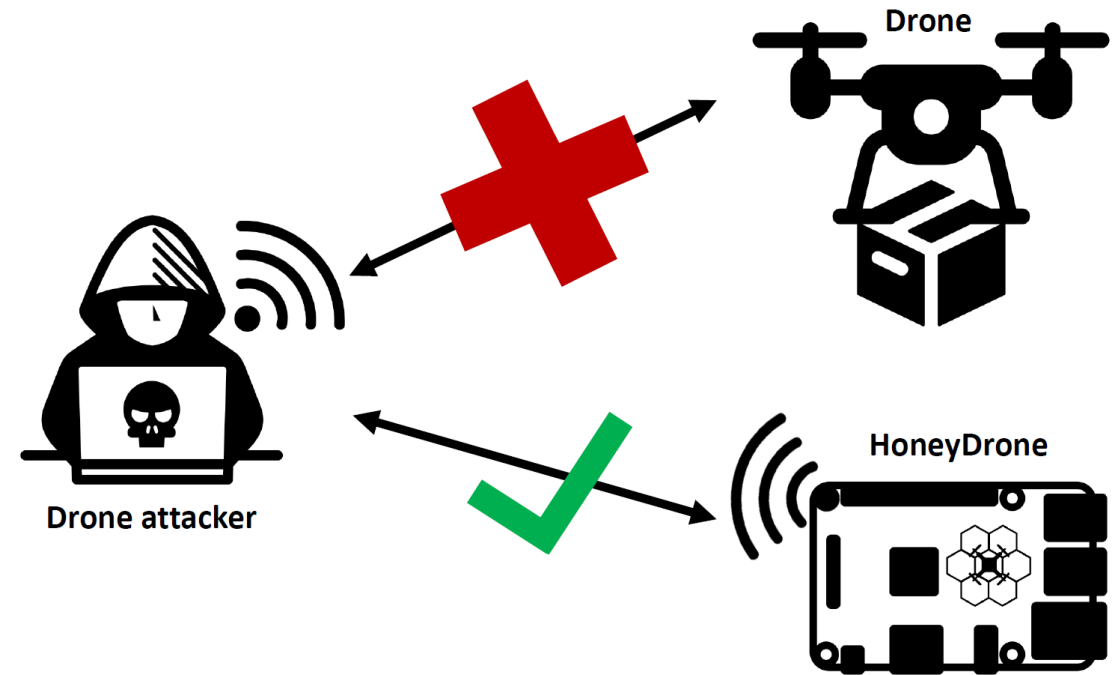
More crazy ideas on honeypots

- Honeypots can emulate **ANYTHING**
- and they can be used in out-of-the-box scenarios



Don't Steal my Drone: Catching Attackers with an Unmanned Aerial Vehicle Honeypot,

Vasilomanolakis et al., IEEE/IFIP NOMS 2018

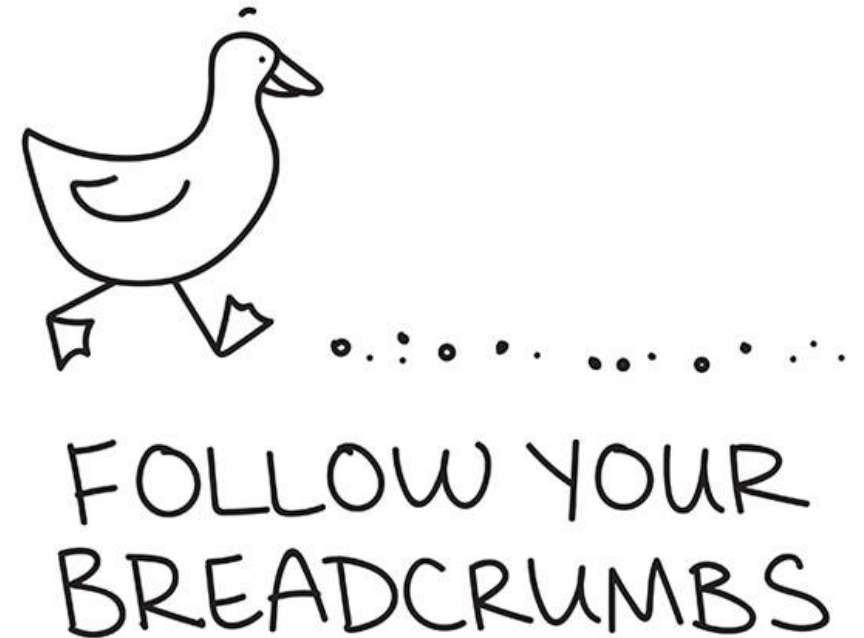


Other deception methods

- **Decoy** systems:
 - Fake windows/Linux boxes (**inventing**),
 - login always fail (**dazzling**)
 - All attempts are logged
 - touch,
 - scan,
 - probe,
 - login attempt
- **Tarpit**: delay connection

Breadcrumbs?

- Endpoint **lures**, **breadcrumbs**, and **baits**: fake artifacts including registry entries, credentials, shared drives
- AD specific: Deception Decoys and Breadcrumbs Obfuscate AD infrastructure and expose attempts to attack it, using fake domain controllers, AD forests, and baits



detecting fake systems

HONEYPOT DETECTION

Attacks on honeypots

- **Detection of honeypots**
 - Honeypots are fake systems and hence can always be detected; it's only a question of effort
- **Evasion**
- **Availability attacks**
 - DDoS attack
- **Hack a honeypot**
 - Pivot attacks

Honeypot detection

- **Low/medium** interaction honeypots can be identified by:
 - Artifacts as a result of their fixed implementation
 - Signatures (e.g., via Nmap)
 - By executing non-expected commands into them
 - Manually by carefully examining the compromised system
- **High** interaction honeypots are normally invisible:
 - Real systems
 - The attacker might attempt to detect the virtual environment
 - The attacker might attempt to detect the monitoring tools

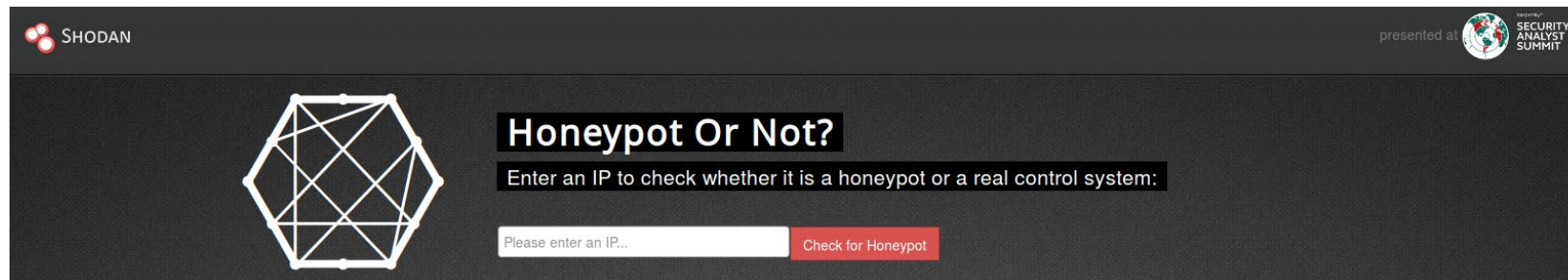
Nmap scanning

- Nmap can, by default, detect many honeypots:
 - Dionaea
 - Honeyd
 - ...

```
Starting Nmap 6.40 ( http://nmap.org ) at 2016-11-07 12:18 CET
Nmap scan report for test1.tk.informatik.tu-darmstadt.de (130.83.163.16)
Host is up (0.00027s latency).
Not shown: 988 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          Dionaea honeypot ftpd
22/tcp    open  ssh          (protocol 2.0)
42/tcp    open  nameserver?
80/tcp    open  http?
135/tcp   open  msrpc?
443/tcp   open  ssl/https?
445/tcp   open  microsoft-ds Dionaea honeypot smbd
1433/tcp  open  ms-sql-s     Dionaea honeypot MS-SQL server
1723/tcp  open  pptp?
3306/tcp  open  mysql        MySQL 5.0.54
5060/tcp  open  sip          (SIP end point; Status: 200 OK)
5061/tcp  open  ssl/sip?     (SIP end point; Status: 200 OK)
5 services unrecognized despite returning data. If you know the service/version,
please submit the following fingerprints at http://www.insecure.org/cgi-bin/ser
vicefp-submit.cgi :
```

Shodan automatic scanning

- Shodan recently introduced a “honeypot or not” service
 - Crawls the Internet
 - Performs various checks on the detected systems
 - Binary result: “yes or no”




Manual Shodan checks: Conpot artifact example

Shodan Scanhub Developers View All...

SHODAN "module: 88111222" Q Explore Membership Contact Us Blog Enterprise Access

Exploits Maps Download Results Create Report

TOP COUNTRIES



United States	11
Taiwan, Province of China	6
Netherlands	3
United Kingdom	2
Germany	2

TOP ORGANIZATIONS

Amazon.com	3
UNE	2
ServerStack	2
DigitalOcean	2
Three	1

Showing results 1 - 10 of 35

188.29.54.93

188.29.54.93.threembb.co.uk
Three
Added on 2015-06-14 08:51:49 GMT
United Kingdom
Details

Location designation of a **module:**
Copyright: Original Siemens Equipment
Module type: IM151-8 PN/DP CPU
PLC name: Technodrome
Module: v.0.0
Plant identification: Mouser Factory
OEM ID of a **module:**
Module name: Siemens, SIMATIC, S7-200
Serial number of **module: 88111222**

46.101.9.153

DigitalOcean
Added on 2015-06-08 10:47:50 GMT
United Kingdom, London
Details

Location designation of a **module:**
Copyright: Original Siemens Equipment
Module type: IM151-8 PN/DP CPU
PLC name: Technodrome
Module: v.0.0
Plant identification: Mouser Factory
OEM ID of a **module:**
Module name: Siemens, SIMATIC, S7-200
Serial number of **module: 88111222**

Bitter Harvest: Systematically Fingerprinting Low- and Medium-interaction Honeypots at Internet Scale (University of Cambridge, 2018)

- Transport layer fingerprinting probes

- Managed to detect **~7600 instances of honeypots!**

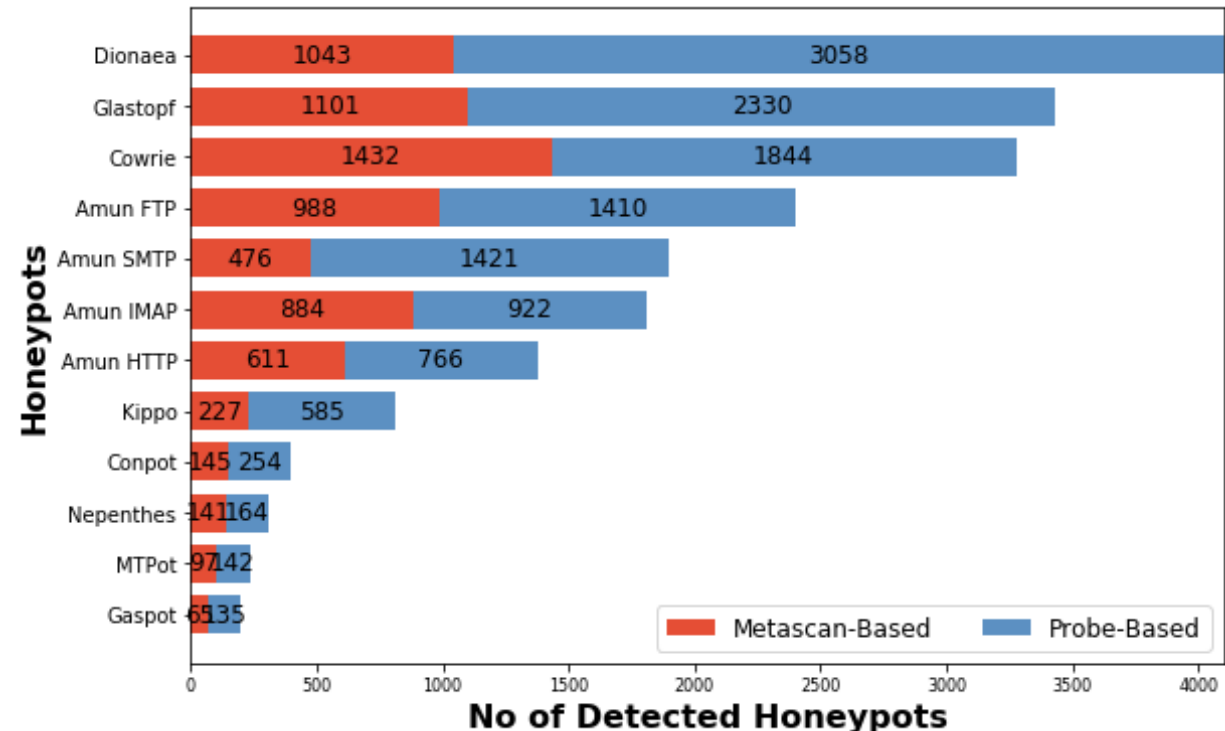
	Updated	Language	Library
SSH			
Kippo	May 15	Python	TwistedConch
Cowrie	May 18	Python	TwistedConch
Telnet			
TPwd	Feb 16	C	custom
MTPot	Mar 17	Python	telnetrv
TIoT	May 17	Python	custom
Cowrie	May 18	Python	TwistedConch
HTTP/Web			
Dionaea	Sep 16	Python	custom
Glastopf	Oct 16	Python	BaseHTTPServer
Conpot	Mar 18	Python	BaseHTTPServer

Table 4: Top 10 ASNs used to host Honeypots (latest scans)

CO	ASN	Organisation	Telnet	SSH	HTTP	Total
US	16509	Amazon.com	140	520	506	1166
JP	2500	WIDE Project	–	–	490	490
US	14061	Digital Ocean	162	189	139	490
FR	16276	OVH SAS	117	202	122	441
TW	4662	GCNet	15	2	254	271
TW	18182	Sony Network	2	–	256	258
US	15169	Google LLC	45	139	46	230
TW	9924	Taiwan Fixed	1	74	146	221
US	14618	Amazon.com	12	70	110	192
RO	43443	DDNET Sol.	30	–	155	185

Gotta catch 'em all: a Multistage Framework for honeypot fingerprinting

- Managed to detect **~20,000 instances of honeypots!**



Honeyscanner

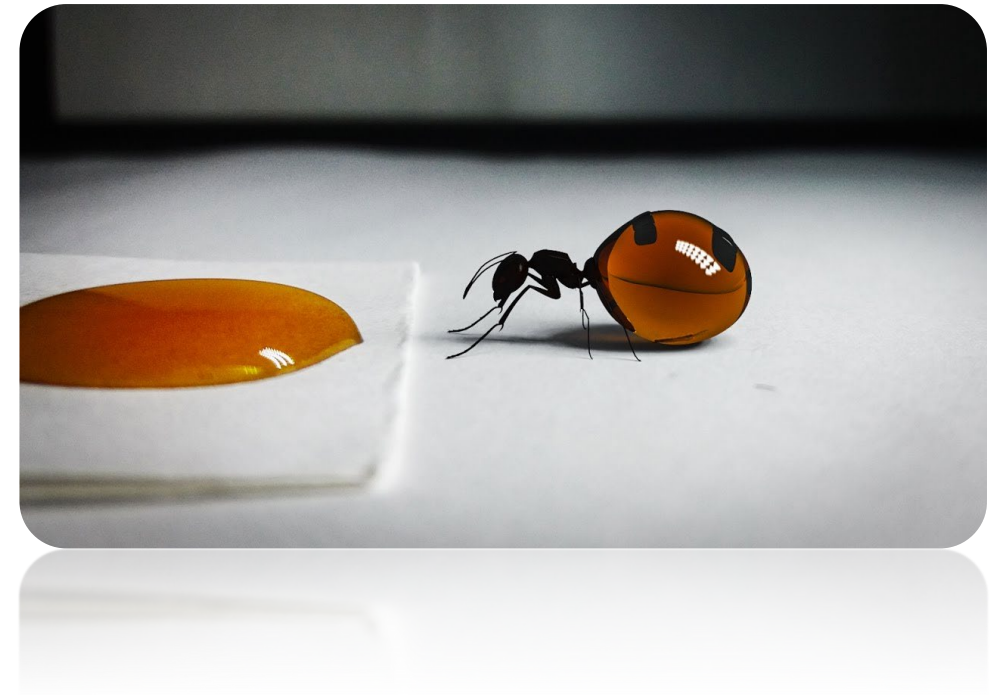
- [Open-source](#) vulnerability analysis tool for honeypots
- Supports Cowrie, Kippo, Dionaea
- Performs:
 - Passive attacks
 - Active attacks
 - E.g., a DoS attack
 - Fuzzing
 - Software library exploitation
 - Tar bomb attacks



Google
Summer of Code

Honeypots

- **Core takeaway message:**
 - Honeypots are useful and fun
 - (but still they need to be handled with care)
 - Not a stand-alone security mechanism
 - Avoid deprecated honeypots and default configurations
- Outside your firewall
 - See what kind of attacks are hitting your network
- Inside your firewall
 - Find old/infected internal devices
 - Insiders



Overview

- Introduction
- Firewalls
- Intrusion detection
- Cyber-deception
- **Lab exercises**

Lab exercises

- Play, edit, and understand **IPTABLES** and firewall rules
- Install, setup, and play with a **honeypot**
 - Cowrie SSH honeypot