IT-Security (ITS) B1

DIKU, E2020

Today’s agenda

Part 1+2: Intrusion detection

Next time: Forensics

Overall IT-security goals

Prevent as much as possible with *best practices* such as secure coding, whitelisting, patching, secure configurations and more

Anticipate breaches and **build to contain** with segmentation, diversity, least privilege, defense in depth and more

Detect and **respond** when things go wrong

Learn and **repeat**

If or when?

“There are two kinds of companies.

There are those who've been hacked and those who don't know they've been hacked.”

FBI Director James Comey, 2014

The Cyber Kill Chain

MITRE ATT@CK

ATT&CK Matrix for Enterprise

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Drive-by Compromise	AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	AppleScript	Audio Capture	Commonly Used Port	Automated Exfiltration	Data Destruction
Exploit Public-Facing Application	CMSTP	Accessibility Features	Accessibility Features	Binary Padding	Bash History	Application Window Discovery	Application Deployment Software	Automated Collection	Communication Through Removable Media	Data Compressed	Data Encrypted for Impact
External Remote Services	Command-Line Interface	Account Manipulation	AppCert DLLs	BITS Jobs	Brute Force	Browser Bookmark Discovery	Distributed Component Object Model	Clipboard Data	Connection Proxy	Data Encrypted	Defacement
Hardware Additions	Compiled HTML File	AppCert DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping	Domain Trust Discovery	Exploitation of Remote Services	Data from Information Repositories	Custom Command and Control Protocol	Data Transfer Size Limits	Disk Content Wipe
Replication Through Removable Media	Control Panel Items	AppInit DLLs	Application Shimming	Clear Command History	Credentials in Files	File and Directory Discovery	Logon Scripts	Data from Local System	Custom Cryptographic Protocol	Exfiltration Over Alternative Protocol	Disk Structure Wipe
Spearphishing Attachment	Dynamic Data Exchange	Application Shimming	Bypass User Account Control	CMSTP	Credentials in Registry	Network Service Scanning	Pass the Hash	Data from Network Shared Drive	Data Encoding	Exfiltration Over Command and Control Channel	Endpoint Denial of Service
Spearphishing Link	Execution through API	Authentication Package	DLL Search Order Hijacking	Code Signing	Exploitation for Credential Access	Network Share Discovery	Pass the Ticket	Data from Removable Media	Data Obfuscation	Exfiltration Over Other Network Medium	Firmware Corruption
Spearphishing via Service	Execution through Module Load	BITS Jobs	Dylib Hijacking	Compile After Delivery	Forced Authentication	Network Sniffing	Remote Desktop Protocol	Data Staged	Domain Fronting	Exfiltration Over Physical Medium	Inhibit System Recovery
Supply Chain Compromise	Exploitation for Client Execution	Bootkit	Exploitation for Privilege Escalation	Compiled HTML File	Hooking	Password Policy Discovery	Remote File Copy	Email Collection	Domain Generation Algorithms	Scheduled Transfer	Network Denial of Service
Trusted Relationship	Graphical User Interface	Browser Extensions	Extra Window Memory Injection	Component Firmware	Input Capture	Peripheral Device Discovery	Remote Services	Input Capture	Fallback Channels		Resource Hijacking
Valid Accounts	InstallUtil	Change Default File Association	File System Permissions Weakness	Component Object Model Hijacking	Input Prompt	Permission Groups Discovery	Replication Through Removable Media	Man in the Browser	Multi-hop Proxy		Runtime Data Manipulation
	Launchctl	Component Firmware	Hooking	Control Panel Items	Kerberoasting	Process Discovery	Shared Webroot	Screen Capture	Multi-Stage Channels		Service Stop
	Local Job Scheduling	Component Object Model Hijacking	Image File Execution Options Injection	DCShadow	Keychain	Query Registry	SSH Hijacking	Video Capture	Multiband Communication		Stored Data Manipulation
	LSASS Driver	Create Account	Launch Daemon	Deobfuscate/Decode Files or Information	LLMNR/NBT-NS Poisoning and Relay	Remote System Discovery	Taint Shared Content		Multilayer Encryption		Transmitted Data Manipulation
	Mshta	DLL Search Order Hijacking	New Service	Disabling Security Tools	Network Sniffing	Security Software Discovery	Third-party Software		Port Knocking		

Good news

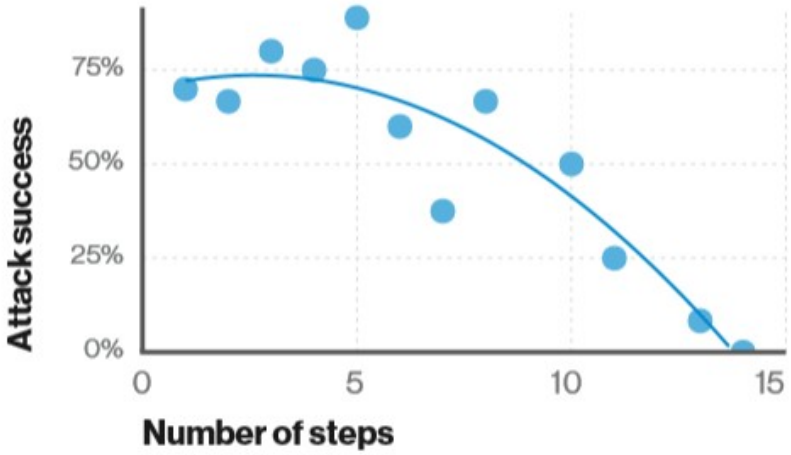


Figure 34. Attack success by chain length in simulated incidents (n=87)

Bad news

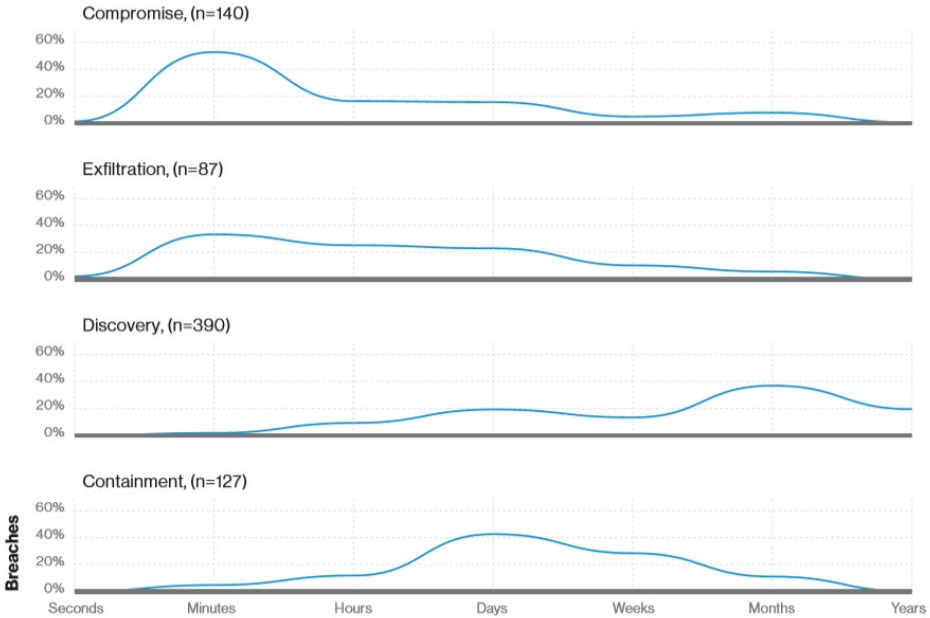
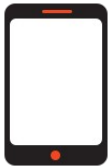


Figure 28. Breach timelines

Where should we focus?



What does the evidence say?

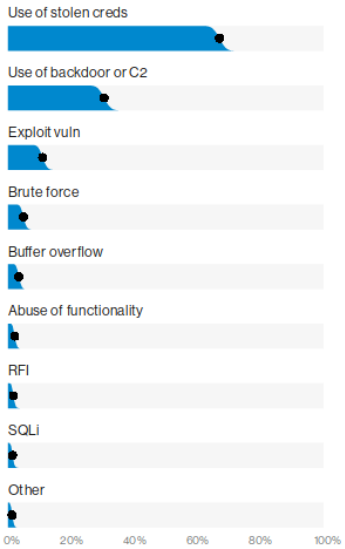


Figure 13. Top hacking action varieties in breaches (n=755)

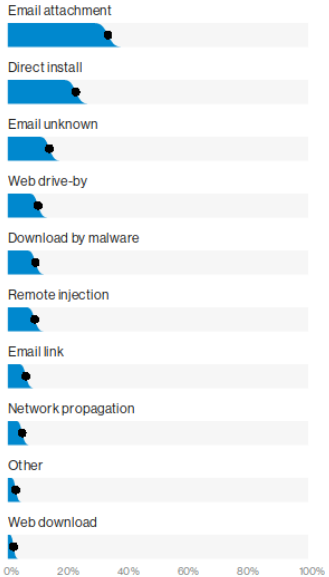


Figure 18. Top malware action vectors in incidents (n=795)

Warm-up 1

Is this an incident?

[**] IIS vti_inf access attempt [**]

05/31-19-09:16:13 63.209.91.31:4791 -> 10.0.0.13:80

TCP TTL:116 TOS:0x0 ID:6075 DF

***PA* Seq:0x1CB4699 Ack:0x2AE6F9 Win:0x217C

← **Snort**

[Tue May 31 09:16:13 2019] [error] [client 63.209.91.31]

File does not exist: /usr/local/apache/htdocs/_vti_inf.html

[Tue May 31 09:16:14 2019] [error] [client 63.209.91.31]

File does not exist: /usr/local/apache/htdocs/_vti_bin/shtml.exe/_vti_rpc

← **Web server**

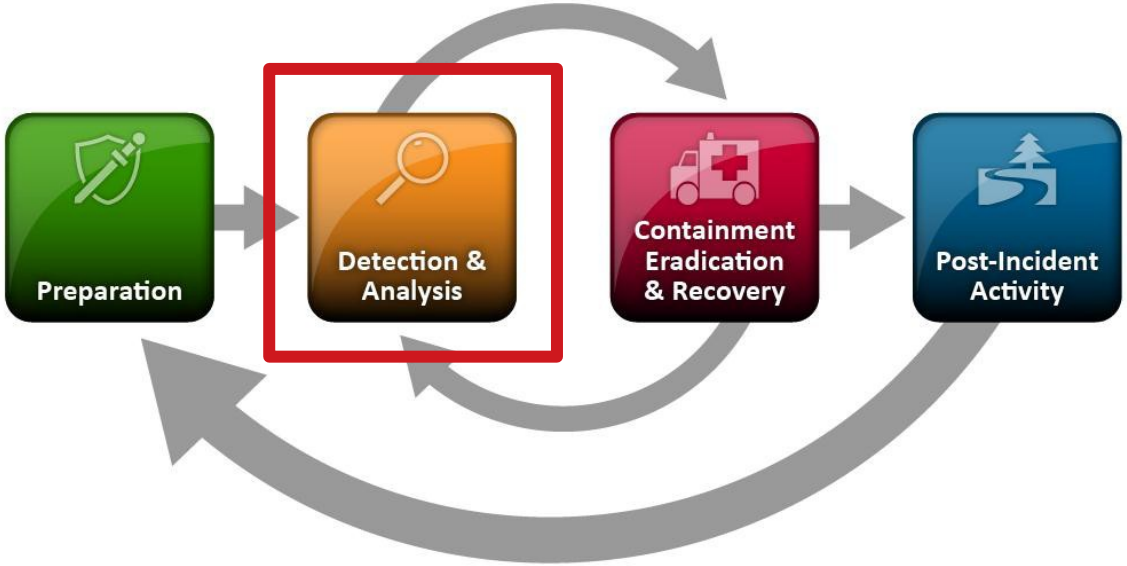
Warm-up 2

#1 attack vector: spearphishing

Where can detection occur?

Intrusion detection

Intrusion detection process



What is in our intrusion detection toolbox?

Network detection

Host detection

Indicators of compromise

Anomaly detection

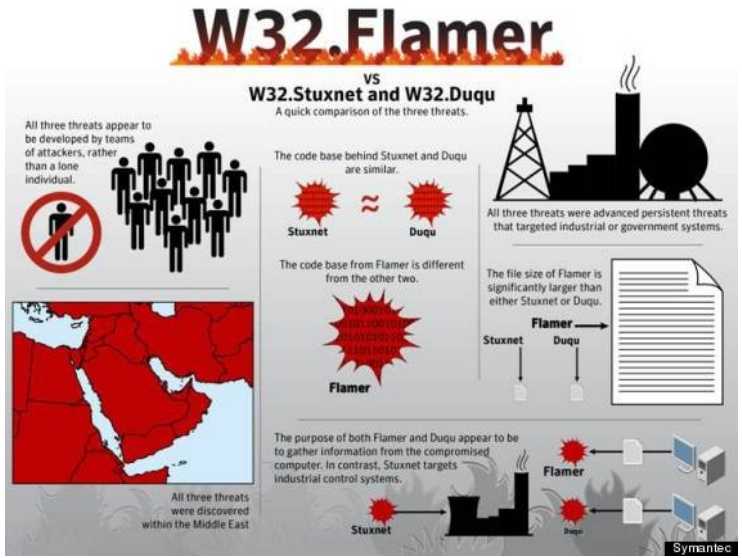
Human ingenuity

Indicators of compromise (IOCs)

Technical characteristics that identify a known threat, attacker methodology, or other evidence of compromise, e.g.:

- C2 domains
- IPs used in attack
- Special GET requests
- Malware file system locations
- Malware hashes
- Memory artifacts

IOCs on Duqu



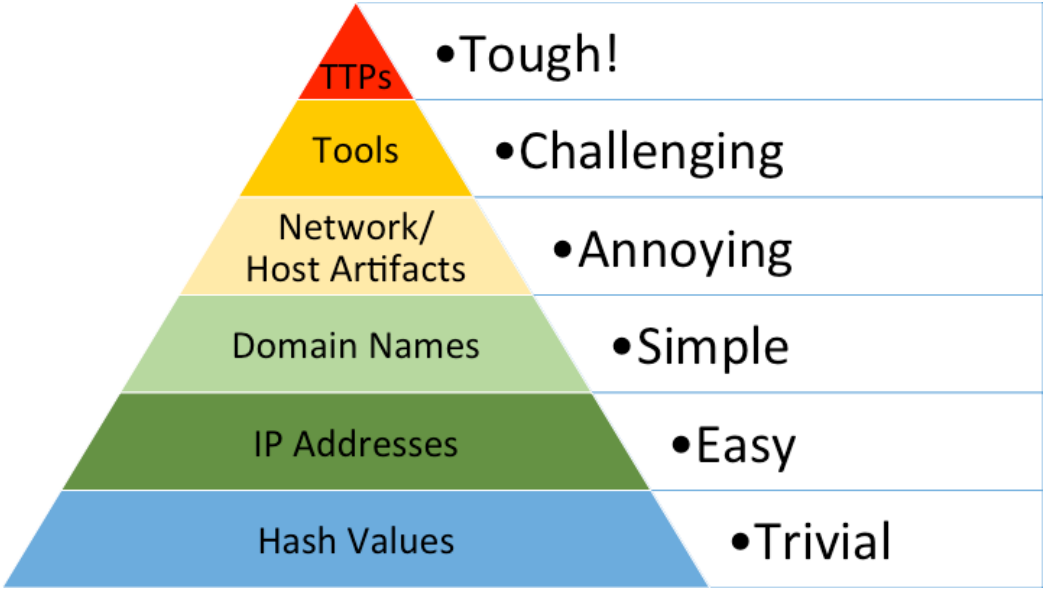
https://securelist.com/files/2015/06/7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f.ioc - Opera

Menu https://securelist.com/file

securelist.com/files/2015/06/7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f.ioc


```
▼<ioc xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.mandiant.com/2010/ioc"
 id="7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f" last-modified="2015-06-10T11:48:29">
  <short_description>TheDuqu 2.0 IOCs</short_description>
  ▼<description>
     Indicators of compromise for the Duqu 2.0 https://securelist.com/blog/research/70504/the-mystery-
     of-duqu-2-0-a-sophisticated-cyberespionage-actor-returns/
   </description>
   <authored_by>Kaspersky Lab</authored_by>
   <authored_date>2015-06-09T21:47:32</authored_date>
   <links/>
  ▼<definition>
    ▼<Indicator operator="OR" id="ad9e4858-9a36-4bf3-822f-04aad37e4887">
      ▼<IndicatorItem id="aa142b0a-c795-4a01-ad86-a938910091ea" condition="is">
         <Context document="FileItem" search="FileItem/Md5sum" type="mir"/>
         <Content type="md5">089a14f69a31ea5e9a5b375dc0c46e45</Content>
       </IndicatorItem>
      ▼<IndicatorItem id="87853206-5a78-4260-a4ac-2a9b1e82c1f3" condition="is">
         <Context document="FileItem" search="FileItem/Md5sum" type="mir"/>
         <Content type="md5">16ed790940a701c813e0943b5a27c6c1</Content>
       </IndicatorItem>
      ▼<IndicatorItem id="7fe336c9-c70c-43c1-a6c9-dce88dae9c40" condition="is">
         <Context document="FileItem" search="FileItem/Md5sum" type="mir"/>
         <Content type="md5">26c48a03a5f3218b4a10f2d3d9420b97</Content>
       </IndicatorItem>
```

Intrusion Detection “Pyramid of Pain”



IOC (hash) strategy

Collect IOC hashes

For each host in my network:

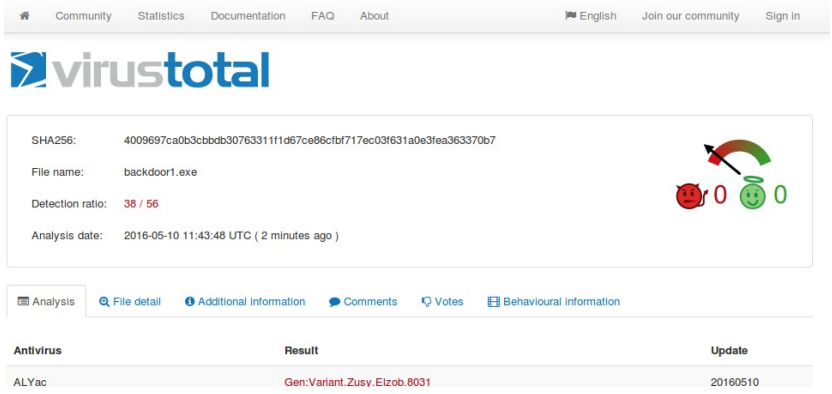
    Calculate file hashes and match against IOC list

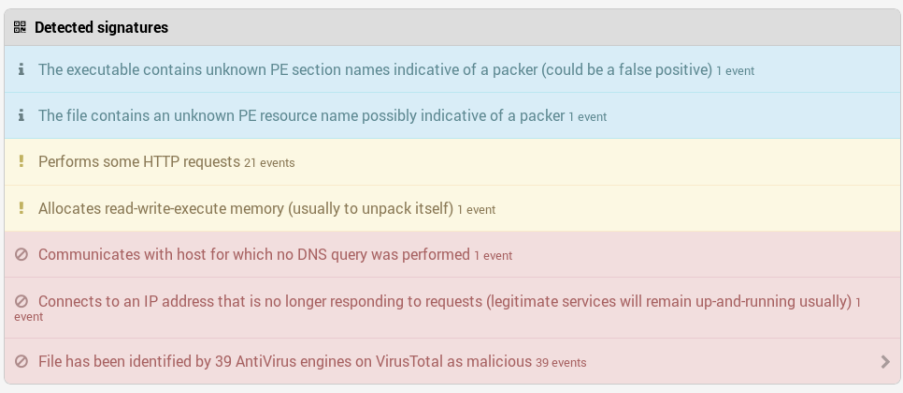
Question: What if something matches? Where did the IOC come from?

Problem: What if attacker updates the malware?

In stead of IOCs, submit file for analysis

Not really anomaly detection – but may be used as data points for machine learning





Virustotal has an API

```
$ cat hashes
DC1E56092CC57FB4605B088D3DCCBF7A
6F8842584D868174E24CACFD18B366B5
0DA1C970D9AA3CCCCFBA7EE90876CBAB

$ cat vt.py
import requests
import sys
import time
with open(sys.argv[1], 'r') as fd:
	for line in fd.readlines():
		params = {'apikey': 'key', 'resource': line.rstrip()}
		response = requests.get('https://www.virustotal.com/vtapi/v2/file/report', params=params)
		response_json = response.json()
		print line.rstrip(), response_json['positives'], response_json['total']

$ python vt.py hashes
DC1E56092CC57FB4605B088D3DCCBF7A   0   66
6F8842584D868174E24CACFD18B366B5   0   68
0DA1C970D9AA3CCCCFBA7EE90876CBAB   26 57
```

Refined approach: In stead of all files, calculate executables that autostart

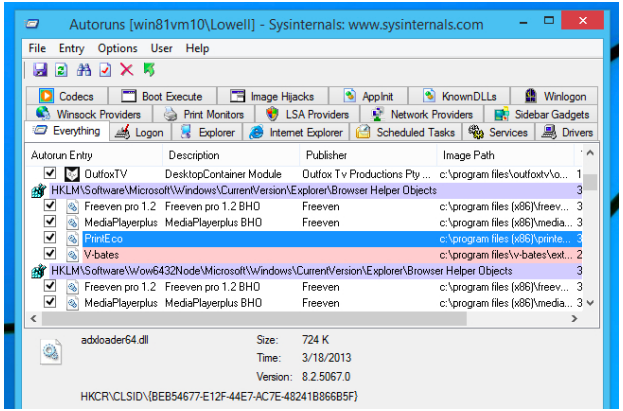
Same as before

IOCs, Virustotal, Sandboxing

Plus

Look for new entries

Look hosts with entries unlike most?



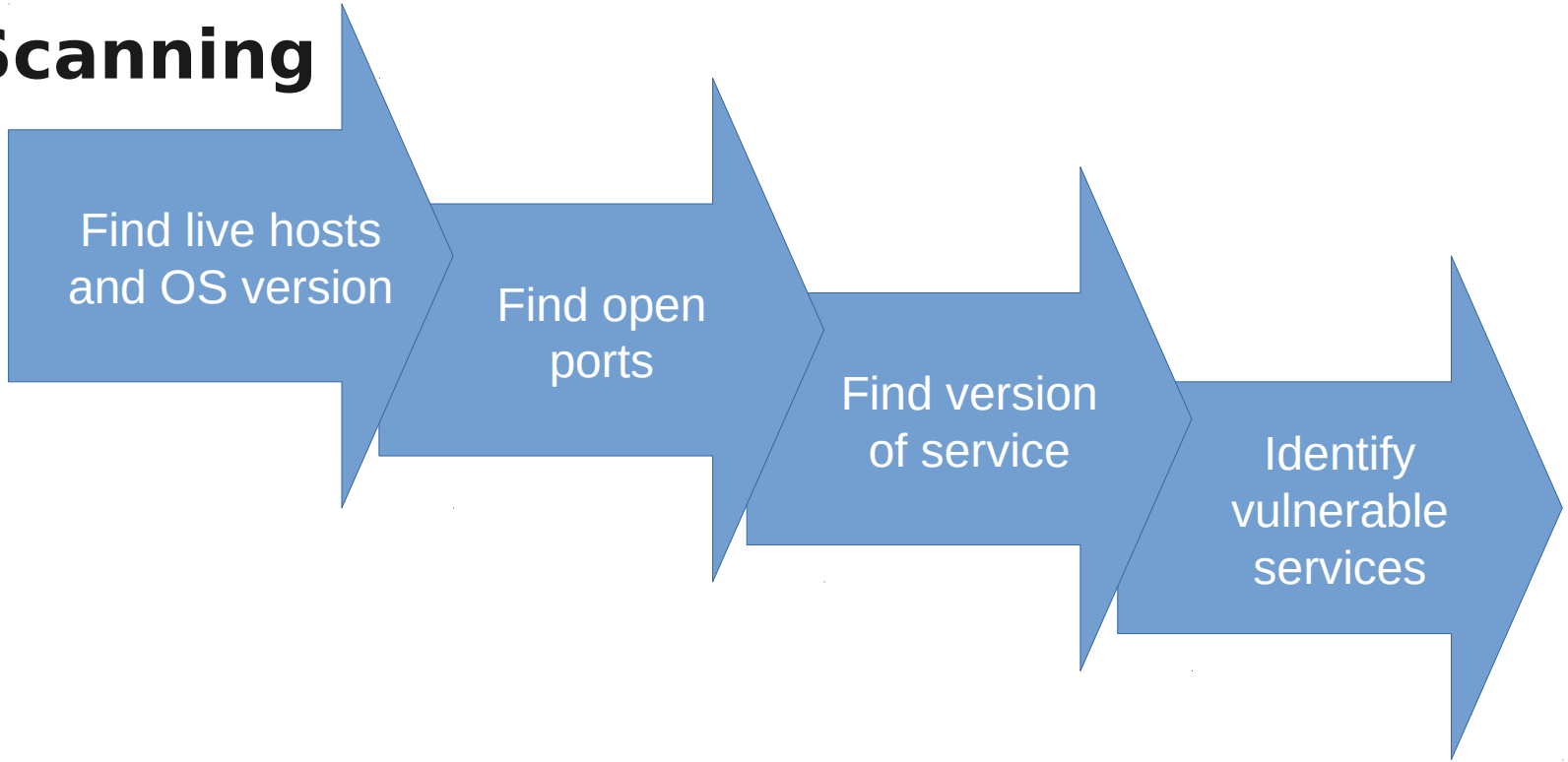
Detection example – Log analysis

Log analysis – Impossible Travel

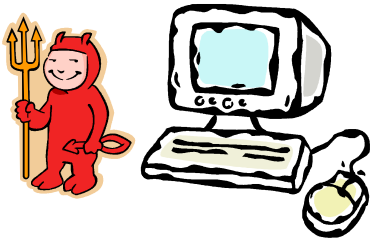
2018-01-21T08:29:49	ceo@NON_DISCLOSED_COMPANY.dk	UserLoggedIn	IP Address
2018-01-21T08:31:34	ceo@NON_DISCLOSED_COMPANY.dk	UserLoggedIn	IP Address
2018-01-21T08:31:45	ceo@NON_DISCLOSED_COMPANY.dk	UserLoggedIn	IP Address
2018-01-21T08:31:47	ceo@NON_DISCLOSED_COMPANY.dk	UserLoggedIn	IP Address
2018-01-21T08:31:48	ceo@NON_DISCLOSED_COMPANY.dk	UserLoggedIn	IP Address
2018-01-21T08:31:54	ceo@NON_DISCLOSED_COMPANY.dk	UserLoggedIn	IP Address
2018-01-21T08:32:54	ceo@NON_DISCLOSED_COMPANY.dk	UserLoggedIn	IP Address
2018-01-21T08:42:30	ceo@NON_DISCLOSED_COMPANY.dk	Set-Mailbox	IP Address

Detection example - Scanning

Scanning

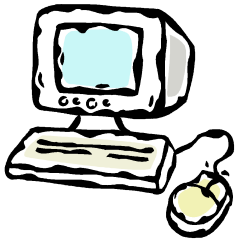

Find live hosts and OS version
Find open ports
Find version of service
Identify vulnerable services

Scanning



Echo request

Echo reply



Scapy / Ping sweep

```
$ sudo python
>>> from scapy.all import *
>>> conf.verb = 0
>>> for i in range(1, 256):
...     packet = IP(dst="192.168.184." + str(i), ttl=20)/ICMP()
...     reply = sr1(packet, timeout=1)
...     if not (reply is None):
...         print reply.src
...     else:
...         print "timeout " + str(i)
...
192.168.184.140
192.168.184.148
```

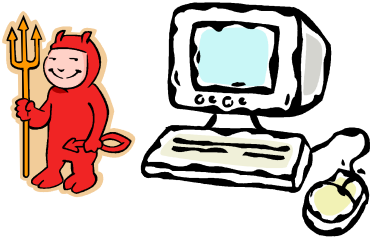
Scapy / ARP sweep

```
$ cat arpscanner.py
from scapy.all import *
conf.verb=0
ans, unans = srp( Ether(dst='ff:ff:ff:ff:ff:ff') \
                /ARP(pdst='192.168.184.139-141'),  \
                timeout = 0.1, iface='vmnet8', inter=0.1)

for a in ans:
  print a[1].psrc, ' ', a[1].hwsrc

$ sudo python arpscanner.py
192.168.184.140 00:50:56:e9:42:d2
192.168.184.140 00:50:56:e9:43:d3
```

Port scanning

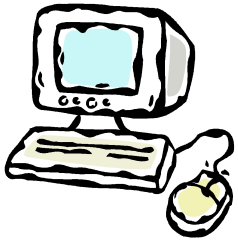


SYN →

← SYN,ACK

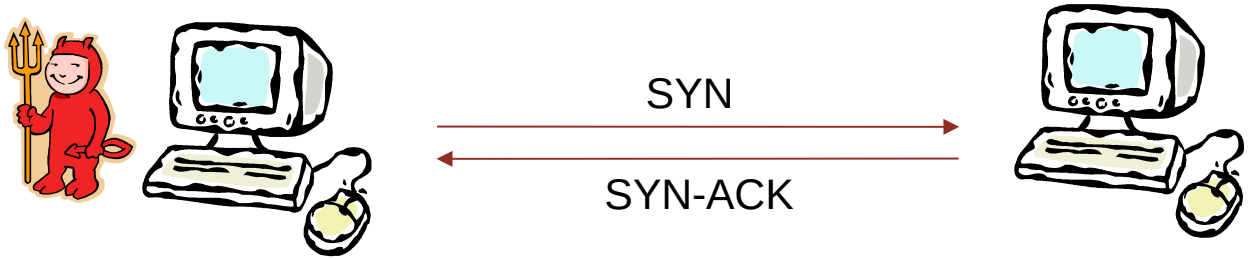
ACK →

DATA ↔

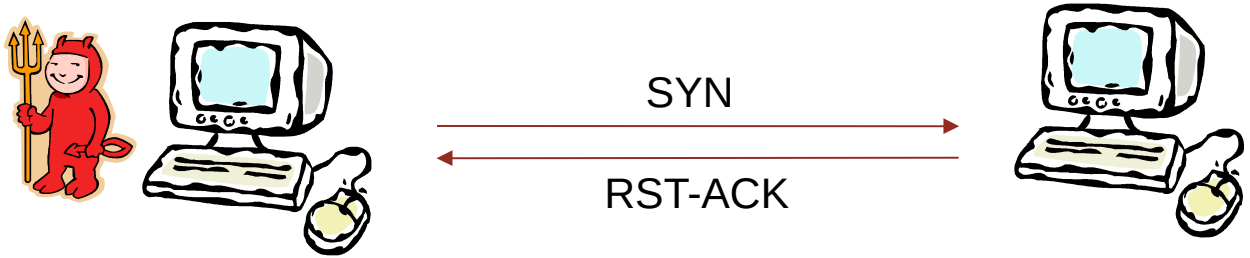


TCP

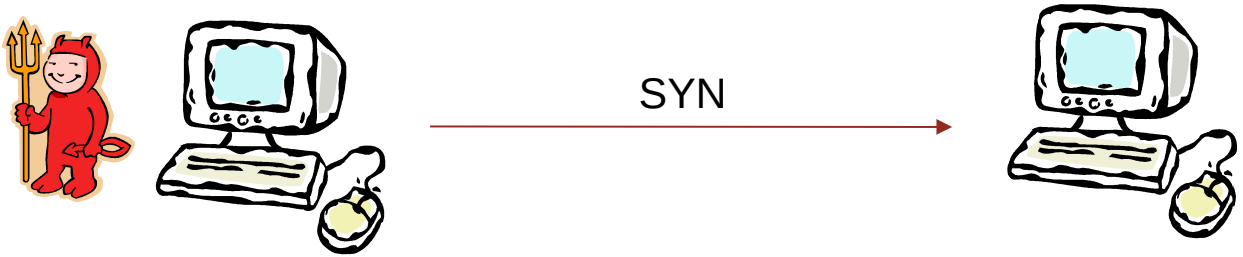
Port open!



Port closed!

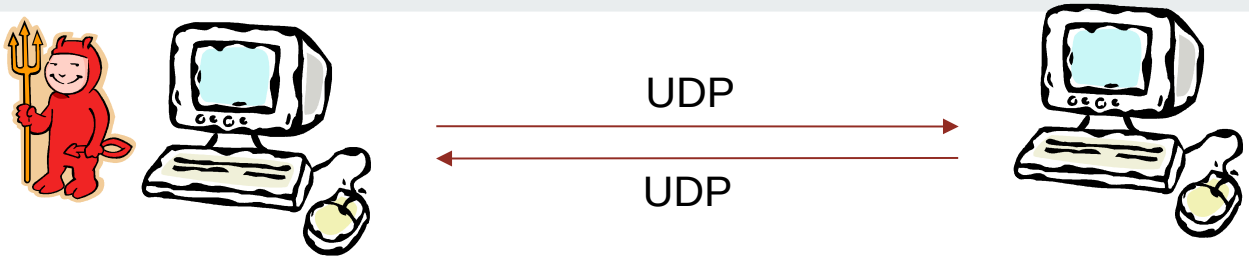


Blocked by firewall?

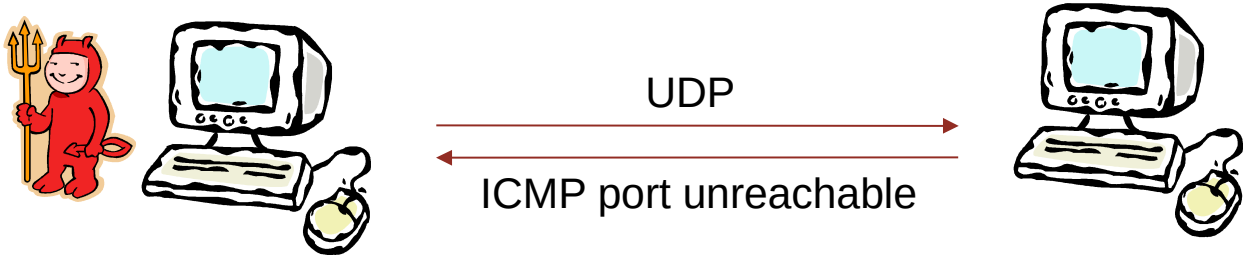


UDP

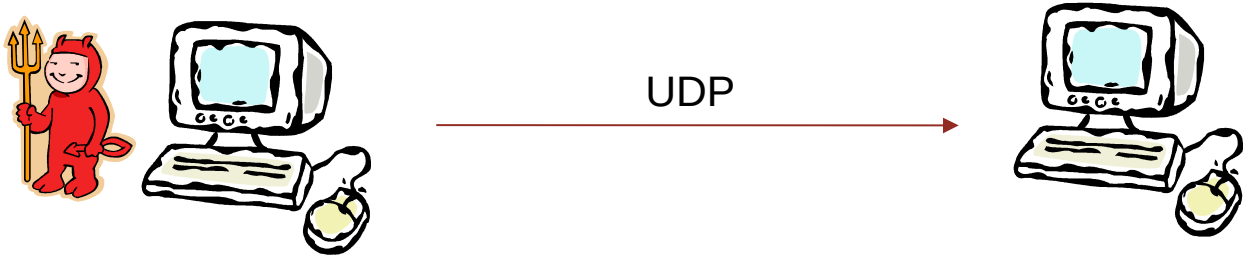
Port open!



Port closed (blocked by firewall?)!



Port closed or blocked by firewall or port open but expecting specific data?



Scanning

nmap
-sS (TCP SYN)
-sT (TCP connect)

nmap 127.0.0.1
nmap –sT 127.0.0.1
nmap –sT –O 127.0.0.1
nmap –sV –p 80,443 127.0.0.1
nmap –sV –script=vulnscan 127.0.0.1

Compare between scans

```
$ nmap -iL targets.txt > scan1

$ nmap -iL targets.txt > scan2

$ diff scan1 scan2
```

Detection example – Files sent over the network

Catch it before it persists

Detection main objective repeated

Catch them as early as possible

Detect before it persists

Terminal

File Edit View Search Terminal Help

```
[spear-demo]$ file smtp.pcap
smtp.pcap: tcpdump capture file (little-endian) - version 2.4 (Ethernet, capture length 65535)
[spear-demo]$ cat getfiles.bro
event file_over_new_connection(f: fa_file, c: connection, is_orig: bool)
{
    Files::add_analyzer(f, Files::ANALYZER_MD5);
    Files::add_analyzer(f, Files::ANALYZER_SHA1);

    local fname = fmt("%s-%s.%s", f$source, f$id, "");
    Files::add_analyzer(f, Files::ANALYZER_EXTRACT, [$extract_filename=fname]);
}
[spear-demo]$ bro -r smtp.pcap getfiles.bro
[spear-demo]$ file extract_files/*
extract_files/SMTP-F0Ruuz4XKclJkPLBTc.: ASCII text, with CRLF line terminators
extract_files/SMTP-F4jmsQ3mEzw9pvd4Ai.: PDF document, version 1.5
extract_files/SMTP-FCjckV1feY1WL7eN33.: ASCII text, with CRLF line terminators
extract_files/SMTP-FdwEMuqwejiv9Si1i.:  ASCII text, with CRLF line terminators
extract_files/SMTP-FG5rTinVh9RHUsri3.:  PDF document, version 1.2
extract_files/SMTP-FMRBBd2EPnXVUZ5a94.: PNG image data, 300 x 300, 8-bit/color RGBA, non-interlaced
extract_files/SMTP-FR99RDrcZPCeMAg9b.:  PE32 executable (GUI) Intel 80386, for MS Windows
extract_files/SMTP-FUtgMs2lTfs1UBaof6.: ASCII text, with CRLF line terminators
extract_files/SMTP-FVnEFCIKs18DeJqbf.:  ASCII text, with CRLF line terminators
[spear-demo]$ md5sum extract_files/*
9f582f5736650a5d2ad9e451e7e56cb2  extract_files/SMTP-F0Ruuz4XKclJkPLBTc.
1ee0b74de96f15c54ef80ac2ee7ea6bc  extract_files/SMTP-F4jmsQ3mEzw9pvd4Ai.
9f582f5736650a5d2ad9e451e7e56cb2  extract_files/SMTP-FCjckV1feY1WL7eN33.
9f582f5736650a5d2ad9e451e7e56cb2  extract_files/SMTP-FdwEMuqwejiv9Si1i.
e1a79647295c4c116ca57df3244ff473  extract_files/SMTP-FG5rTinVh9RHUsri3.
ff2342dfa13ada586cf72d8d83662f1c  extract_files/SMTP-FMRBBd2EPnXVUZ5a94.
73191fc401e30a188fd7bafeda3e6068  extract_files/SMTP-FR99RDrcZPCeMAg9b.
9f582f5736650a5d2ad9e451e7e56cb2  extract_files/SMTP-FUtgMs2lTfs1UBaof6.
9f582f5736650a5d2ad9e451e7e56cb2  extract_files/SMTP-FVnEFCIKs18DeJqbf.
[spear-demo]$ for each $hash; do look_up_virustotal; done
```

Wrap-up

Lecture plan

Mandag d. 28. september
- kl. 10-12 Cryptography

Fredag d. 2. oktober
- kl. 09-10 Internet security protocols (bemærk ekstra time fra kl. 9 allerede)
- kl. 10-12 Intrusion detection

Mandag d. 5. oktober
- kl. 9-11 Forensics (bemærk flyttet fra kl. 10-12 til kl. 09-11)