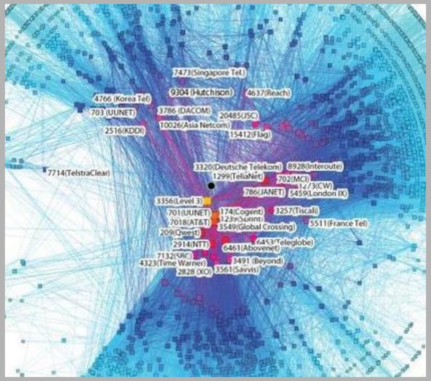
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Prof. Dr. Mathias Fischer
Winter term 2021/2022



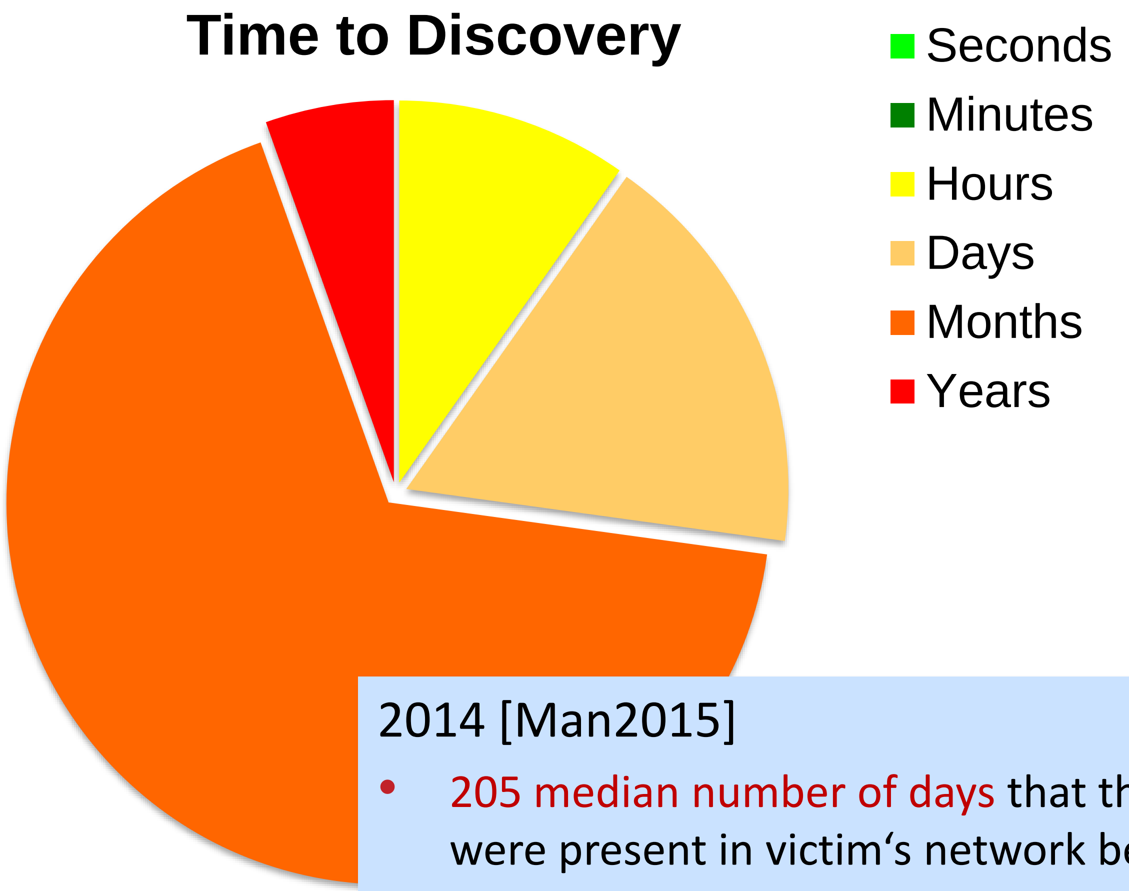
Resilient Networks

Network Monitoring and Intrusion Detection

Outline

- Goals of IDS
- Requirements to an IDS
- Classification of IDS
- Problems of IDS
- Alert Correlation
- Cyber-Killchain und MITRE ATT&CK
- IDS Evasion
- Summary

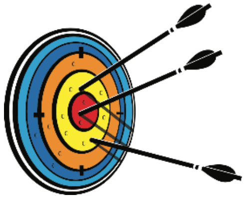
Time to Discovery of Attacks



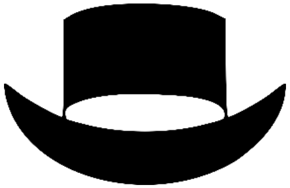
2014 [Man2015]

- 205 median number of days that threat groups were present in victim's network before detection
- Longest Presence: 2982 days
- 31% victims discovered breach internally
- 69% victims notified by external entity

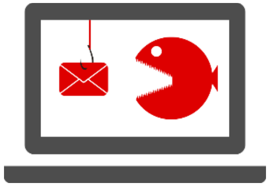
Goal of Intrusion Detection/Prevention Systems



- Overall goal:
  - Intrusion Detection Systems (IDS)
    Supervision of computer systems and communication infrastructures to detect intrusions and misuse
  - Intrusion Prevention Systems (IPS)
    Detect and stop intrusion/misuse



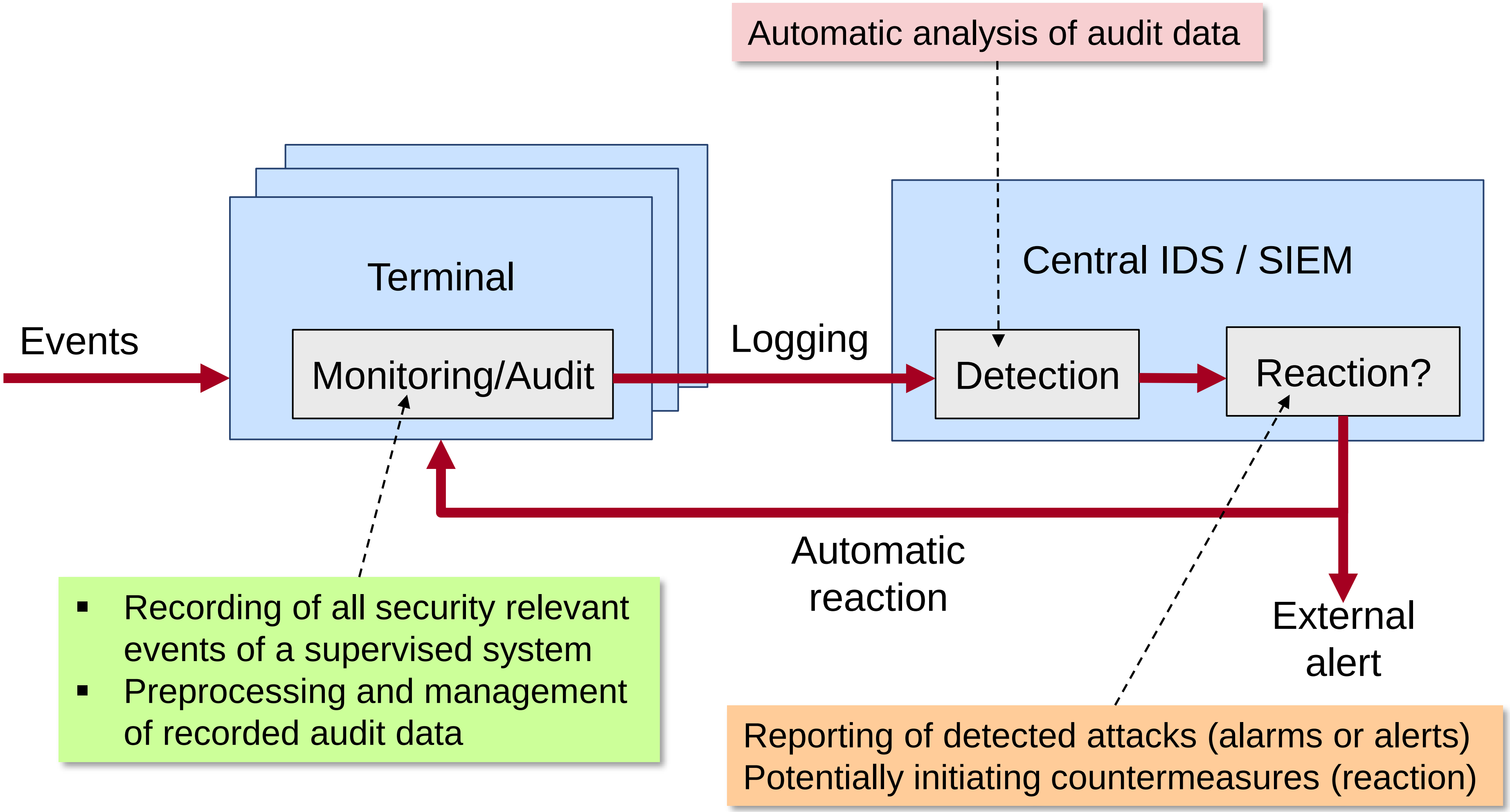
- Why detection of attackers?
  - Full protection not possible!
  - Security measures too expensive or too inflexible
  - Wrong postulates about attacker capabilities (NSA!?)
  - Unpatched systems for compliance reasons
  - …



- What can be attained with intrusion detection?
  - Detection of attacks and attackers + detection of system misuse
  - Limitation of damage if (automated) response mechanisms exist
  - Gain of experience to recover from attack, improve preventive measures
  - Deterrence of other potential attackers (if police is able to arrest them!)

Operation of Intrusion Detection/Prevention Systems

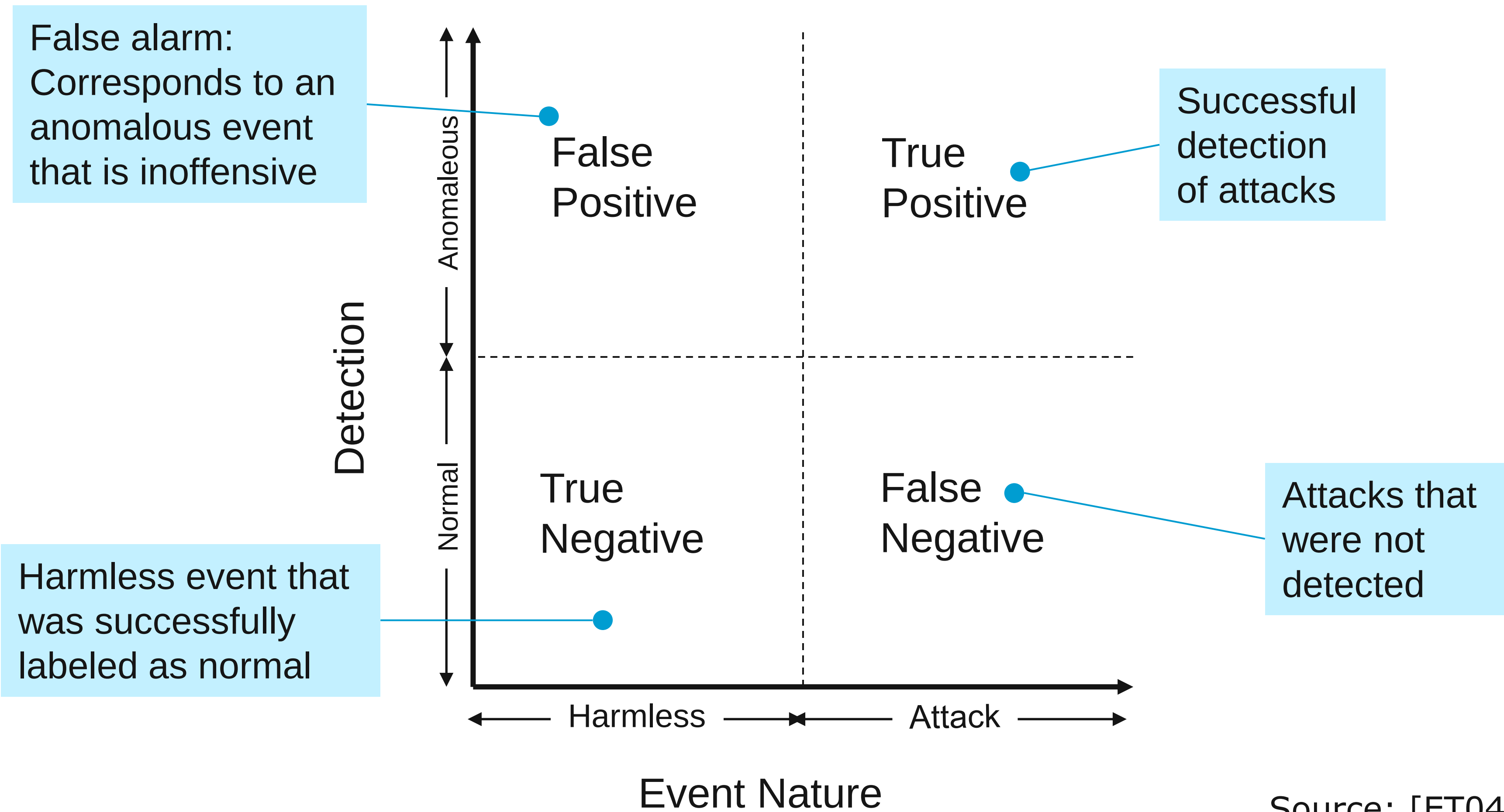
SIEM = Security Information and Event Management



Requirements to Intrusion Detection Systems

- Easy to integrate into a system / network
- Easy to configure & maintain
- Autonomous and fault tolerant operation
- Low resource requirements
- Self-protection, so that IDS cannot be deactivated by deliberate attack (to conceal subsequent attacks)
- High accuracy
  (= low rate of false positives and false negatives)

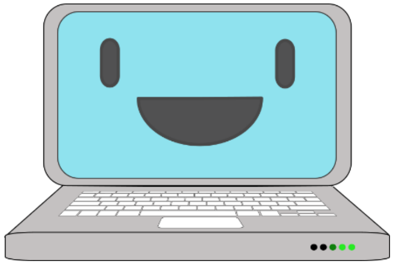
Detection Quality



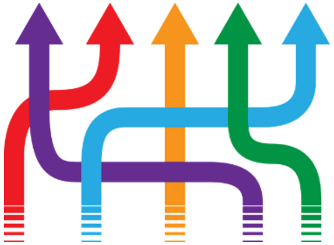
Source: [ET04]

Types of Audit Data

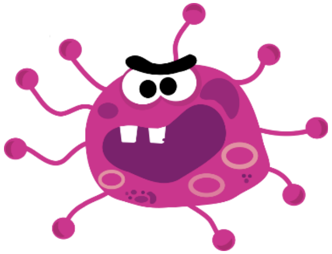
- Events recorded in a computer system:
  - Opening of files
  - Execution of programs
  - Detected access violation
  - Failed password verification
  - etc.



- Events recorded in a network:
  - Connection establishment and release
  - Packets transferred from / to specific systems / ports
  - Specific signaling events,
    e.g. ICMP network unreachable message, etc.



- Application specific events:
  - Have to be programmed for a specific application
  - Events are application specific and indicate security relevant activities



Classification of IDS

- Scope
  - Host-based: analysis of system events
  - Network-based: analysis of exchanged information (IP packets)
  - Hybrid: combined analysis of system events and network traffic

- Time of analysis
  - Post mortem analysis
  - Online analysis

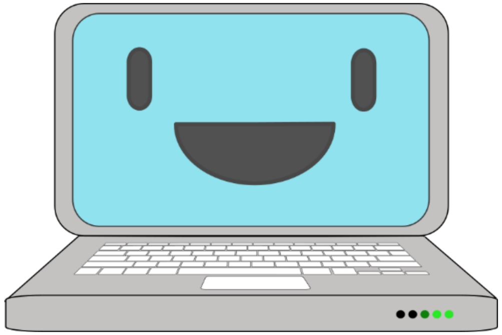
- Detection mechanism
  - Signature-based
  - Policy-based / Misuse-based / Anomaly-based

Types of IDS (1) – Host IDS

Host Intrusion Detection Systems (HIDS)

- Works on information available on a system,
  e.g., OS-Logs, application-logs, timestamps
- Can easily detect attacks by insiders, as modification of files, illegal access to files, installation of Trojans or rootkits

- Problems:
  - must be installed on every system
  - produces lots of information
  - often no real-time-analysis but predefined time intervals
  - hard to manage large number of systems



```
osquery> SELECT uid, name FROM listening_ports l, processes p WHERE l.pid=p.pid;
```

Example of a Host-Monitor – Osquery (1)

- Allows to use OS as high-performance relational database
  - SQL tables representing abstract concepts
- Power of complete SQL language on top of dozens of useful tables

processes

All running processes on the host system.

Column	Type	Description
pid	BIGINT_TYPE	Process (or thread) ID
name	TEXT_TYPE	The process path or shorthand argv[0]
path	TEXT_TYPE	Path to executed binary
cmdline	TEXT_TYPE	Complete argv
state	TEXT_TYPE	Process state
cwd	TEXT_TYPE	Process current working directory
root	TEXT_TYPE	Process virtual root directory
uid	BIGINT_TYPE	Unsigned user ID
gid	BIGINT_TYPE	Unsigned group ID
euid	BIGINT_TYPE	Unsigned effective user ID
egid	BIGINT_TYPE	Unsigned effective group ID
suid	BIGINT_TYPE	Unsigned saved user ID
sgid	BIGINT_TYPE	Unsigned saved group ID
on_disk	INTEGER_TYPE	The process path exists yes=1, no=0, unknown=-1
wired_size	BIGINT_TYPE	Bytes of unpagable memory used by process
resident_size	BIGINT_TYPE	Bytes of private memory used by process
total_size	BIGINT_TYPE	Total virtual memory size
user_time	BIGINT_TYPE	CPU time spent in user space
system_time	BIGINT_TYPE	CPU time spent in kernel space
start_time	BIGINT_TYPE	Process start in seconds since boot (non-sleeping)
parent	BIGINT_TYPE	Process parent's PID
pgroup	BIGINT_TYPE	Process group
threads	INTEGER_TYPE	Number of threads used by process
nice	INTEGER_TYPE	Process nice level (-20 to 20, default 0)

```
select * from processes where pid = 1
```

Tables

All Platforms
- carbon_black_info
- chrome_extensions
- cpuid
- etc_hosts
- etc_protocols
- etc_services
- interface_addresses
- interface_details
- kernel_info
- listening_ports
- os_version
- platform_info
- process_open_sockets
- processes
- system_info
- uptime
- users

- running processes
- logged in users
- password changes
- USB devices
- firewall exceptions
- listening ports
- ....

usb_devices

USB devices that are actively plugged into the host system.

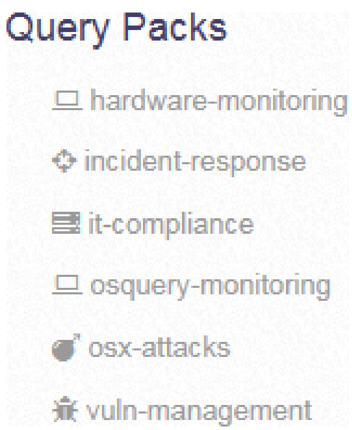
Column	Type	Description
usb_address	INTEGER_TYPE	USB Device used address
usb_port	INTEGER_TYPE	USB Device used port
vendor	TEXT_TYPE	USB Device vendor string
vendor_id	TEXT_TYPE	Hex encoded USB Device vendor identifier
model	TEXT_TYPE	USB Device model string
model_id	TEXT_TYPE	Hex encoded USB Device model identifier
serial	TEXT_TYPE	USB Device serial connection
removable	INTEGER_TYPE	1 If USB device is removable else 0

https://osquery.io

Example of an Host-Sensor - Osquery (2)

- High-performance and low-footprint distributed host monitoring
  - To query the system in an abstract way
  - Independent of OS, software or hardware configuration

- Host monitoring daemon
  - allows to schedule queries to be executed across entire infrastructure
  - takes care of aggregating query results over time and generates logs which indicate state changes in the infrastructure



https://osquery.io

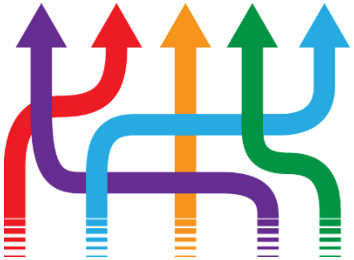
- Cross platform operating system instrumentation framework for
  - intrusion detection,
  - infrastructure reliability
  - or compliance monitoring

- Only monitoring, no intrusion detection capabilities on its own

Types of IDS (2) – Network IDS

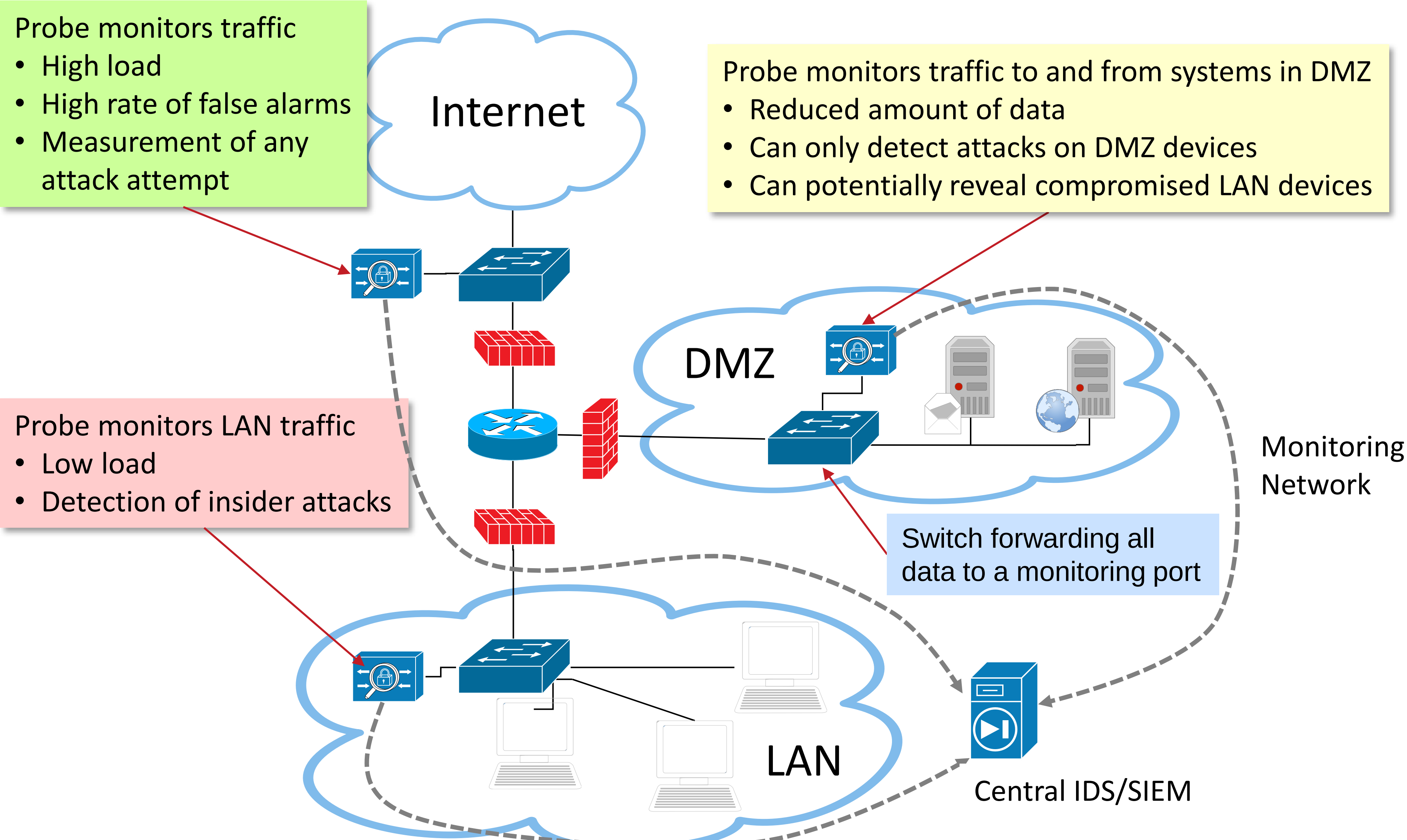
Network Intrusion Detection System (NIDS)

- Works on information provided by network, mainly packets sniffed from network layer.
- Existing systems use combination of
  - signature detection,
  - protocol decoding,
  - statistical anomaly analysis



- Can detect DoS with buffer overflow attacks, invalid packets, attacks on application layer, DDoS, spoofing attacks, port scans
- Often used on network hubs to monitor a segment of the network

Network IDSs



SIEM = Security Information and Event Management

Signature Detection

- Basic idea
  - Some attack patterns can be described with sufficient detail
    → specification of "attack signatures"
  - The event audit analyzed if it contains known attack signatures
- Identifying attack signatures
  - Analyzing vulnerabilities
  - Analyzing past attacks that have been recorded in the audit
- Specifying attack signatures
  - Based on identified knowledge so-called rules describing attacks are specified
  - Most IDS offer specification techniques for describing rules
- Drawbacks of signature-based detection
  - Requires prior knowledge of potential attacks
  - Signature database requires continuous updating
  - High rate of false negatives if signature database is not up to date

Signature Detection – Example: Snort (1)

Network IDS and intrusion prevention system

- Analysis of IP packets in real time
- Mainly signature based, each intrusion needs a predefined rule

```
alert tcp $HOME_NET any -> any 9996 \
 (msg:"Sasser ftp script to transfer up.exe"; \
content:"|5F75702E657865|"; depth:250; flags:A+; classtype: misc-activity; \
sid:1000000; rev:3)
```

- Three step processing of captured information (capturing is done by libpcap):
  - Preprocessing (normalizing and reassembling packets)
  - Detection Engine works on data and decides what action should be taken
  - Action (log, alert, pass)

Policy-based Detection

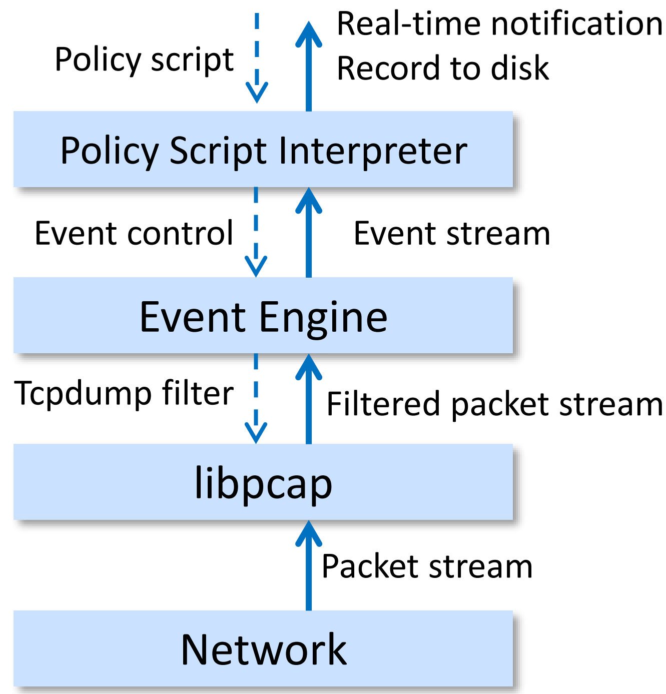
- Also called misuse-based detection

- Basic Idea
  - Specify what is allowed in a network and/or what is forbidden
  - Violations create alerts
  - In that sense, similar to a Firewall

- Drawbacks
  - You can only detect what you configured / what deviates from what you have configured
  - Needs expert knowledge of the system to be protected

Policy-based Detection – Example: Zeek (1)

- Real-time network analysis framework
  - Primary a network monitoring tool
  - Can be used for pure traffic analysis
  - Powerful IDS
- Focus on
  - Application-level semantic analysis
  - Policy-based detection in protocols
  - Tracking information over time
- Zeek comes with >10,000 lines of script code
  - Prewritten functionality that's just loaded
  - Extensive customization and extension possible
  - Growing community writing 3rd party scripts
- Intrusion prevention
  - Zeek can act as dynamic and intelligent firewall



[Pa99]

Zeek Logs (1)

zeek

```
> zeek -i eth0
[ … wait … ]
> ls *.log
app_stats.log            irc.log                  socks.log
communication.log        known_certs.log          software.log
conn.log                 known_hosts.log          ssh.log
dhcp.log                 known_services.log       ssl.log
dns.log                  modbus.log               syslog.log
dpd.log                  notice.log               traceroute.log
files.log                reporter.log             tunnel.log
ftp.log                  signatures.log           weird.log
http.log                 smtp.log
```

Zeek Logs (2)

```
> zeek -i eth0
[ … wait … ]
> cat conn.log
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   conn
#open   2013-04-28-23-47-26
#fields ts                 uid           id.orig_h       id.orig_p   id.resp_h       […]
#types  time               string        addr            port        addr            […]
1258531221.486539  arKYeMETxOg   192.168.1.102   68          192.168.1.1     […]
1258531680.237254  nQcgTWjvg4c   192.168.1.103   37          192.168.1.255   […]
1258531693.816224  j4u32Pc5bif   192.168.1.102   37          192.168.1.255   […]
1258531635.800933  k6kgXLOoSKl   192.168.1.103   138         192.168.1.255   […]
1258531693.825212  TEfuqmmG4bh   192.168.1.102   138         192.168.1.255   […]
1258531803.872834  5OKnoww6xl4   192.168.1.104   137         192.168.1.255   […]
1258531747.077012  FrJExwHcSal   192.168.1.104   138         192.168.1.255   […]
1258531924.321413  3PKsZ2Uye21   192.168.1.103   68          192.168.1.1     […]
[…]
```

Zeek Logs (3) – conn.log

ts	1393099191.817686	Timestamp
uid	Cy3S2U2sbarorQgmw6a	Unique ID
id.orig_h	177.22.211.144	Originator IP
id.orig_p	43618	Originator Port
id.resp_h	115.25.19.26	Responder IP
id.resp_p	25	Responder Port
proto	tcp	IP Protocol
service	smtp	App-layer Protocol
duration	1.414936	Duration
orig_bytes	9068	Bytes by Originator
resp_bytes	4450	Bytes by Responder
conn_state	SF	TCP state
local_orig	T	Local Originator?
missed_bytes	0	Gaps
history	ShAdDaFf	State History
tunnel_parents	(empty)	Outer Tunnels

Zeek Logs (4) - http.log

ts	1393099291.589208
uid	CKFUW73bIADw0r9pl
id.orig_h	17.22.7.4
id.orig_p	54352
id.resp_h	24.26.13.36
id.resp_p	80
method	POST
host	com-services.pandonetworks.com
uri	/soapservices/services/SessionStart
referrer	-
user_agent	Mozilla/4.0 (Windows; U) Pando/2.6.0.8
status_code	200
username	anonymous
password	-
orig_mime_types	application/xml
resp_mime_types	application/xml

Zeek Logs (5) – ssl.log

ts	1392805957.927087
uid	CEA05l2D7k0BD9Dda2
id.orig_h	2a07:f2c0:90:402:41e:c13:6cb:99c
id.orig_p	40475
id.resp_h	2406:fe60:f47::aaeb:98c
id.resp_p	443
version	TLSv10
cipher	TLS_DHE_RSA_WITH_AES_256_CBC_SHA
server_name	www.netflix.com
subject	CN=www.netflix.com,OU=Operations, O=Netflix, Inc.,L=Los Gatos, ST=CALIFORNIA,C=US
issuer_subject	CN=VeriSign Class 3 Secure Server CA, OU=VeriSign Trust Network,O=VeriSign, C=US
not_valid_before	1389859200.000000
not_valid_after	1452931199.000000
client_subject	-
client_issuer_subject	-
cert_hash	197cab7c6c92a0b9ac5f37cfb0699268
validation_status	ok

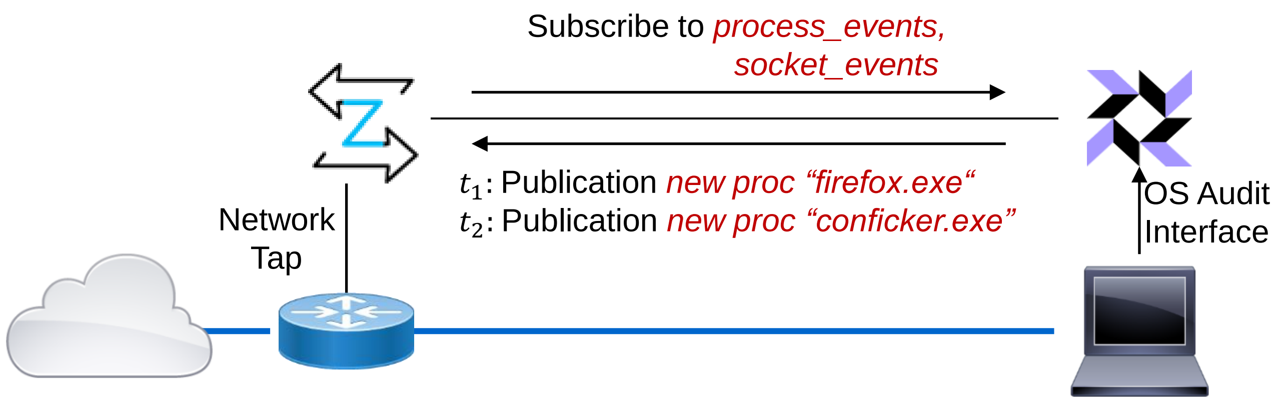
Our Work: zeek-osquery (1)

Zeek

- Flexible network monitoring and IDS
- Integrated scripting language

osquery

- Host monitor
- Information from OS audit interface



Osquery Example Tables	
acpi_tables	firefox_addons
apt_sources	iptables
arp_cache	kernel_modules
certificates	known_hosts
cpu_info	memory_map
cpu_time	process_events
device_partitions	processes
disk_events	socket_events
docker_container_labels	usb_devices
docker_container_mounts	user_ssh_keys
...	

zeek-osquery framework

- Zeek framework that connects to Zeek-enhanced osquery instances
- Attributes network to host activity
- Joint processing of host-events and network data in Zeek scripts

How effective is zeek-osquery in the attribution of connections to processes?

Is zeek-osquery scalable with an increasing amount of osquery hosts?

Our Work: zeek-osquery (2) - Evaluation

Test run on 11 office machines during three days:

- Attribution of network flows to processes

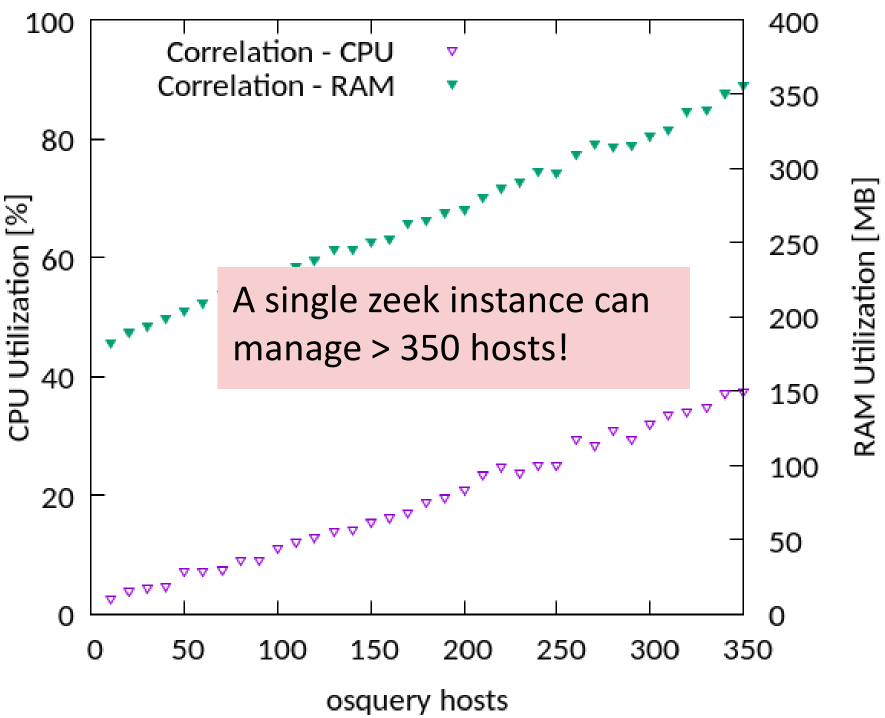
Prot.	# Flows	Zeek	zeek-osquery
All	334.366	0,06%	86,61%
TCP	273.241	0,07%	96,05%
UDP	70.929	0%	50,43%

zeek-osquery		
1	Firefox	23,17%
2	Thunderbird	12,30%
3	Spotify	6,11%
4	Opera	5,41%
5	Syncthing	5,39%
6	Chromium	4,55%
7	Skype	3,86%
8	Seafile	3,80%
9	Chrome	3,56%

zeek-osquery enhances the visibility of Zeek and can attribute connections to processes and users!

Scalability: CPU and RAM utilization at Zeek host

- One Zeek instance, varying number hosts
- 2 events per second per host



A single zeek instance can manage > 350 hosts!

Scales via distributed Zeek deployment

Our Work: zeek-osquery (3)

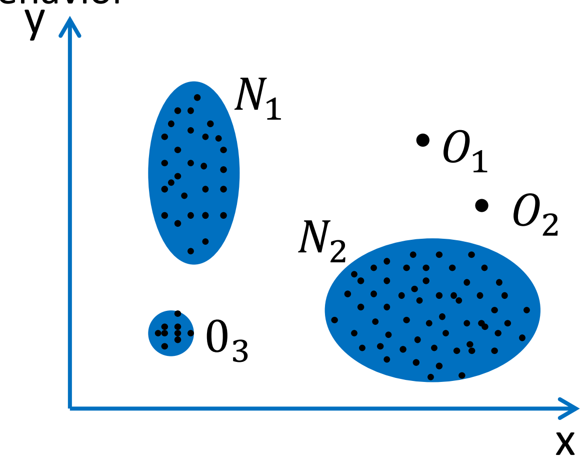
https://github.com/zeek/zeek-osquery
https://github.com/zeek/zeek-agent

- Further application scenarios of zeek-osquery/zeek-agent
  - Transparent decryption of TLS connections [WiHa+21]
  - Detection of malicious file attachments in Emails + information if user opened the attachment
  - Detection of SSH chain logins
  - …

Anomaly Detection (1)

Basic idea – detect behavior that differs significantly from normal use:

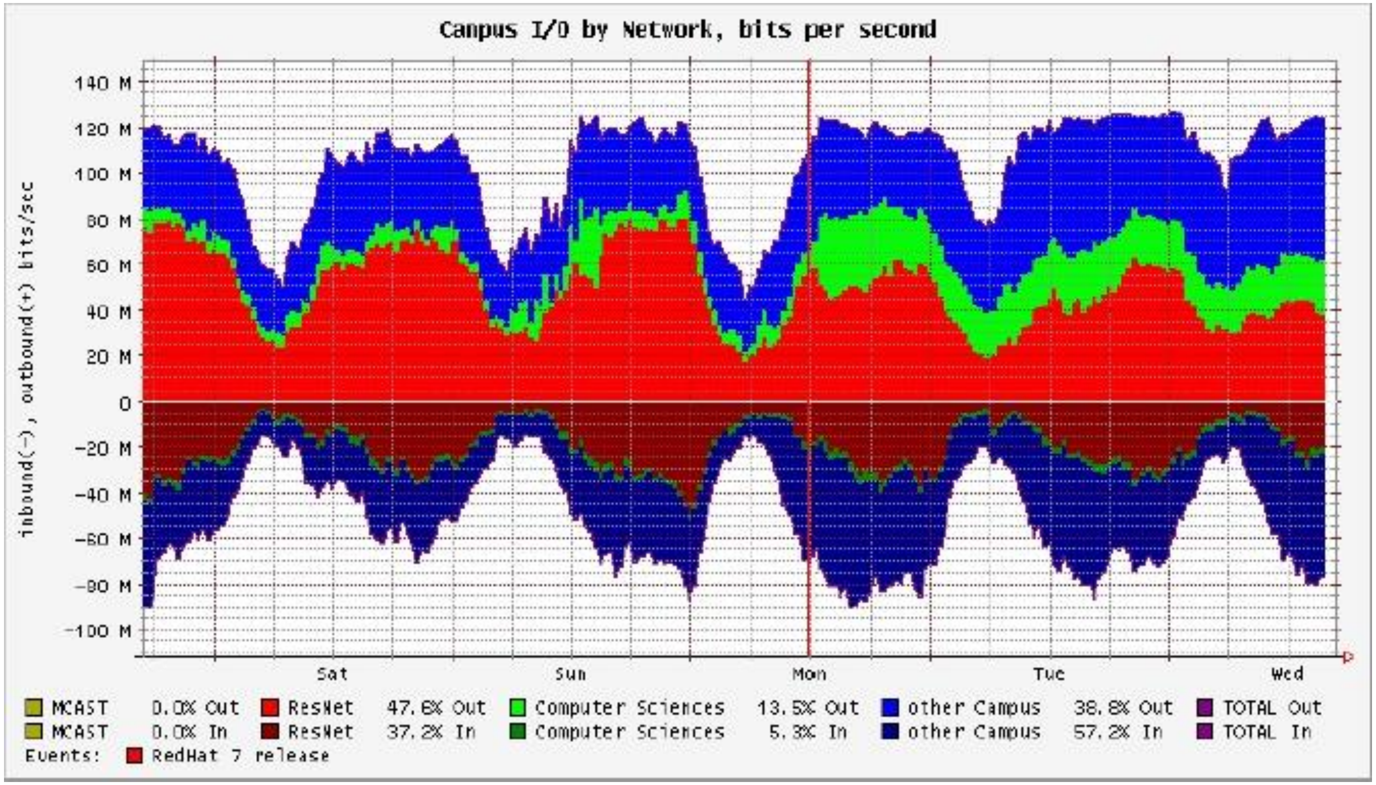
- Users have certain habits in their system usage:
  - Duration of usage
  - Login times
  - Amount of file system usage
  - Executed programs, accessed files, …
- Assumption: "normal user behavior" can be described statistically
  - Requires a learning phase / specification of normal behavior
  - Most approaches require labeled data



- Analysis:
  - compares recorded events with reference profile of normal behavior
- Advantage:
  - An attack scenario needs not to be defined a priori
  - This approach can, in principle, detect unknown attacks

Anomaly Detection (2)

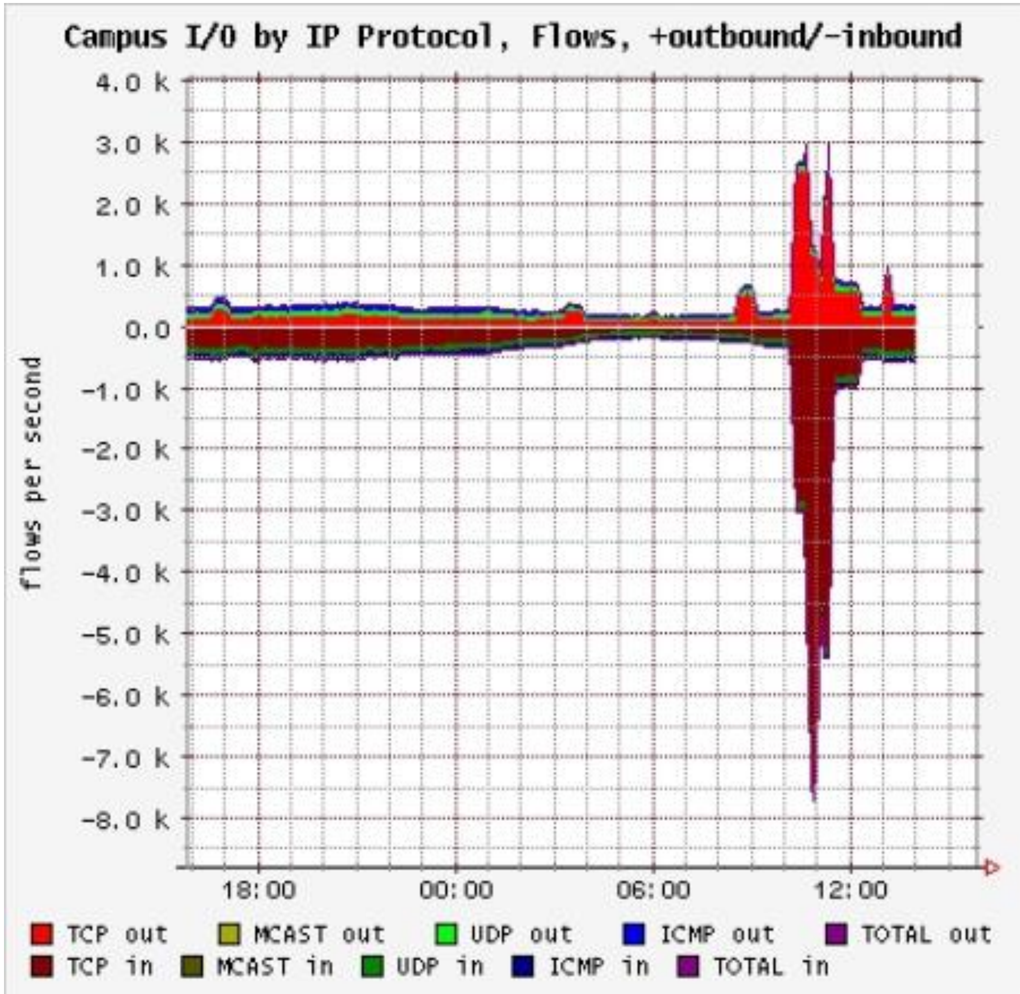
- “Flash crowd anomalies”
  - Caused by software releases or special interest in a web site



[Bar01]

Anomaly Detection (3)

- Network abuse anomalies
  - DoS flood attacks
  - Port scans

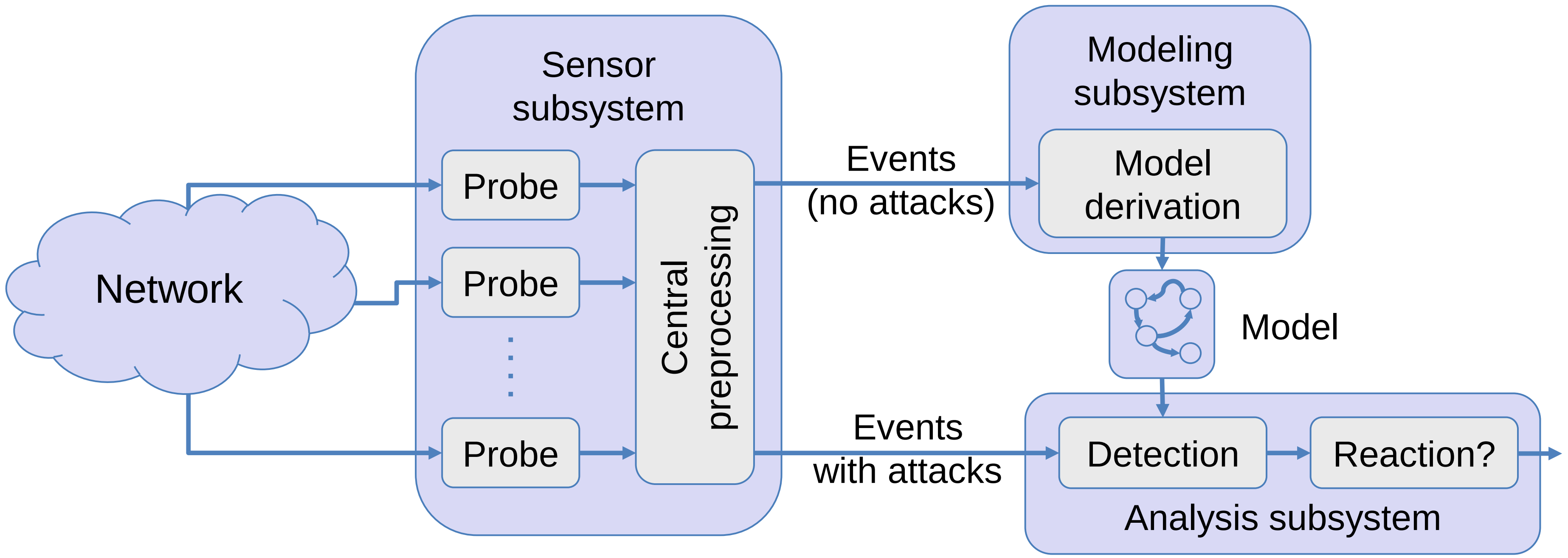


[Bar01]

Automatic Anomaly Detection – System model

[ET04]

Generic anomaly detection system



Problems of IDS – Audit Data

- Amount of log data
  - Auditing often generates a rather high data volume
  - Significant storage capacities are required
  - Processing of audit data should be automated as much as possible
- Location of audit data storage
  - Alternatives: on specific “log server” or the system to be supervised
  - If stored on log server, data must be transferred to this server
  - If stored on system to be supervised, the log uses significant amounts of resources of the system
- Protection of audit data
  - If system gets compromised, audit data stored on it might get compromised either
- Expressiveness of audit data
  - Which information is relevant?
  - Audits often contain rather low percentage of useful information

Problems of IDS – Privacy *(data protection)*

- User identifying data elements are logged, e.g.,
  - Directly identifying elements*:* user IDs
  - Indirectly / partly identifying elements*:* names of directories and subdirectories (home directory), file names, program names
  - Minimally identifying elements*:* host type + time + action, access rights + time + action

- IDS audits may violate the privacy of users
  - Violation of the user's right to determine himself which data is collected regarding his person
  - Collected information might be abused if not secured properly
  - Recording of events puts a psychological burden on users (→ "big brother is watching you")

- Potential (but not sufficient) solution
  - Pseudonymous audit*:* log activities with user pseudonyms and ensure, that they can only be mapped to user IDs upon incident detection

Problems of IDS - Analysis

- Limited efficiency of analysis
  - Most IDS follow a centralist approach for analysis: so-called agents collect audit data and one central evaluation unit analyzes this data

  ⇒ No (partial) evaluation in agents

  ⇒ Performance bottleneck
  - Insufficient efficiency, especially concerning attack variants and attacks with parallel actions

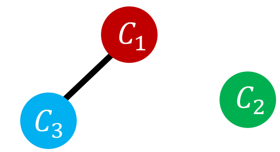
- High number of false positives
  - In practice, many IDS report too many false alarms (some publications report up to 10.000 per month)
  - Potential countermeasure: alert correlation

Alert Level – Alert Correlation Process

[HaFi+19]

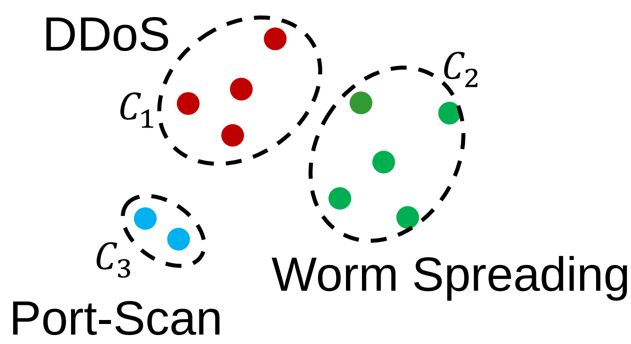
Attack Interconnection

Multi-step Attacks
e.g., Port-Scan -> Targeted attack



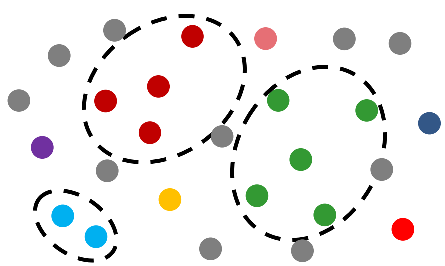
Context Supplementation

Distributed Attacks
DDoS,
Distributed Port-Scans,
Worm spreadings …



Alert Clustering

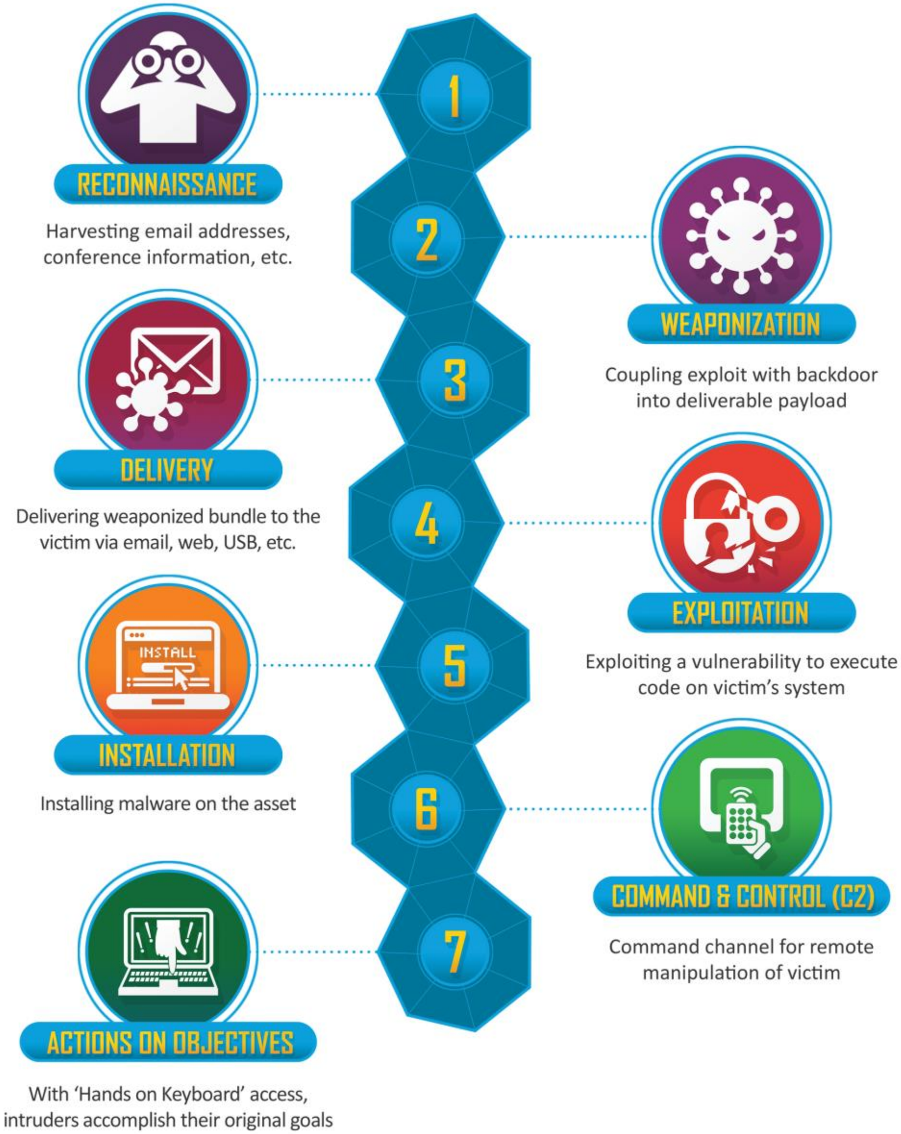
Alert attributes
Src IP, Dst IP
Src Port, Dst Port
…



Alerts from Network- or Host IDS .

Cyber Kill Chain (1)

[HuCl10]



- Proposed by Lockheed Martin in 2011
- Targets Advanced Persistent Threats (APTs)
- 7 consecutive stages that describe the attack campaign
- Inflexible and oversimplified when compared to known attacks

Cyber Kill Chain (2) - Variations

- Several adaptions of the original kill chain
  - By domain (industrial systems, insider attacks..)
  - For increased flexibility (new and optional stages)
- More focus on zone breaching and lateral movement and host activity
- Unified Kill Chain (UKC) as most comprehensive model
  - 18 (partially optional) stages
  - Based on literature review and case studies

#	*Unified Kill Chain*
1	*Reconnaissance*
2	*Weaponization*
3	*Delivery*
4	*Social Engineering*
5	*Exploitation*
6	*Persistence*
7	*Defense Evasion*
8	*Command & Control*
9	*Pivoting*
10	*Discovery*
11	*Privilege Escalation*
12	*Execution*
13	*Credential Access*
14	*Lateral Movement*
15	*Collection*
16	*Exfiltration*
17	*Target Manipulation*
18	*Objectives*

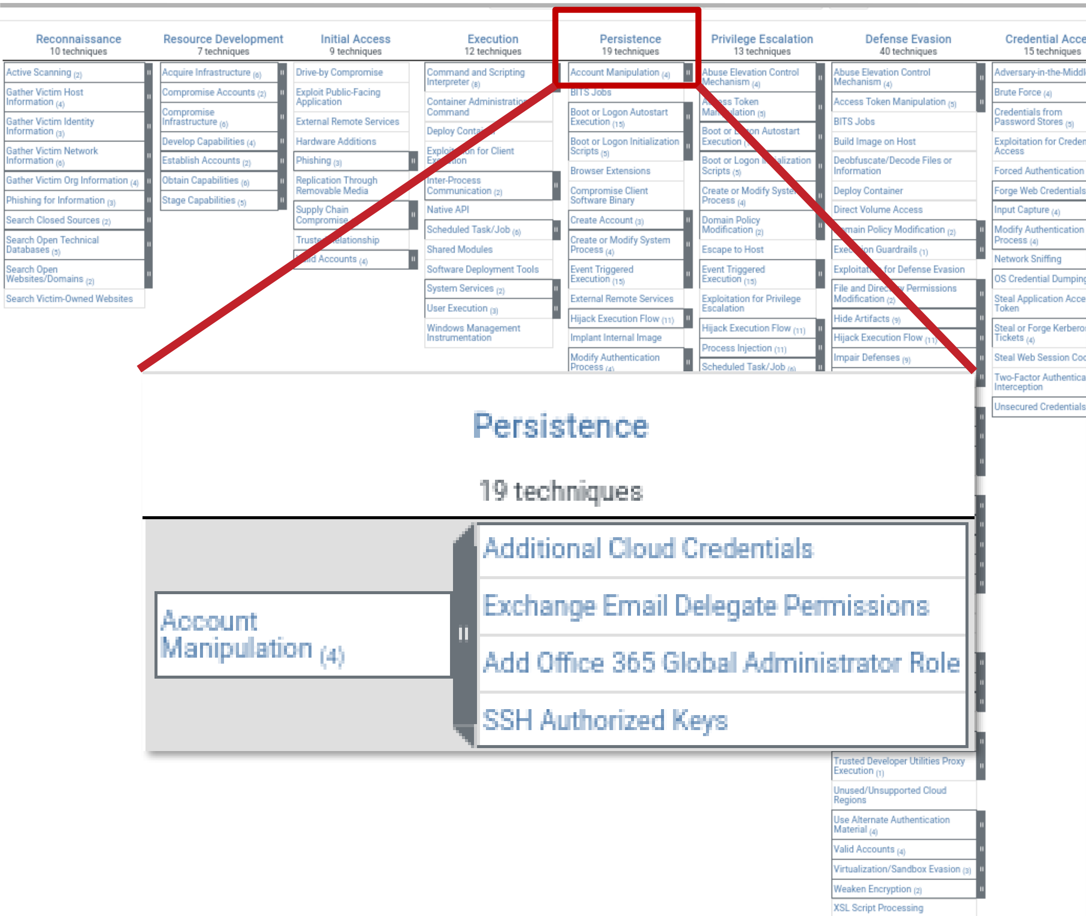
MITRE ATT&CK Framework (1)

- Knowledge base of adversarial tactics and techniques as well as potential mitigations and how to detect attacks
- Based on real-world observations
- Summarized in so-called matrices

- Different matrices for different application scenarios, e.g.,
  - Enterprise security
  - Mobile security
  - Industrial control security

https://attack.mitre.org/

MITRE ATT&CK Framework (2) - Enterprise Matrix



14 categories in this matrix

- Reconnaissance
- Resource Development
- Initial Access
- Execution
- Persistence
- Privilege Escalation
- Defense Evasion
- Credential Access
- Discovery
- Lateral Movement
- Collection
- Command and Control
- Exfiltration
- Impact

MITRE ATT&CK Framework (3) - Account Manipulation

Home > Techniques > Enterprise > Account Manipulation

Account Manipulation

Sub-techniques (4)

Adversaries may manipulate accounts to maintain access to victim systems. Account manipulation may consist of any action that preserves adversary access to a compromised account, such as modifying credentials or permission groups. These actions could also include account activity designed to subvert security policies, such as performing iterative password updates to bypass password duration policies and preserve the life of compromised credentials. In order to create or manipulate accounts, the adversary must already have sufficient permissions on systems or the domain.

ID: T1098

Sub-techniques: T1098.001, T1098.002, T1098.003, T1098.004

Tactic: Persistence

Platforms: Azure AD, Google Workspace, IaaS, Linux, Office 365, Windows, macOS

Contributors: Jannie Li, Microsoft Threat Intelligence Center (MSTIC); Praetorian; Tim MalcomVetter

Version: 2.2

Created: 31 May 2017

Last Modified: 18 October 2021

Version Permalink

Procedure Examples

ID	Name	Description
G0022	APT3	APT3 has been known to add created accounts to local admin groups to maintain elevated access.[1]
S0274	Calisto	Calisto adds permissions and remote logins to all users.[2]
G0074	Dragonfly 2.0	Dragonfly 2.0 added newly created accounts to the administrators group to maintain elevated access.[3][4]
G0032	Lazarus Group	Lazarus Group malware WhiskeyDelta-Two contains a function that attempts to rename the administrator's account.[5][6]
S0002	Mimikatz	The Mimikatz credential dumper has been extended to include Skeleton Key domain controller authentication bypass functionality. The `LSADUMP::ChangeNTLM` and `LSADUMP::SetNTLM` modules can also manipulate the password hash of an account without knowing the clear text value.[7][8]
G0034	Sandworm Team	Sandworm Team used the `sp_addlinkedsrvlogin` command in MS-SQL to create a link between a created account and other servers in the network.[9]
S0649	SMOKEDHAM	SMOKEDHAM has added created user accounts to local Admin groups.[10]

Procedure Examples

ID	Name	Description
G0022	APT3	APT3 has been known to add created accounts to local admin groups to maintain elevated access.[1]
S0274	Calisto	Calisto adds permissions and remote logins to all users.[2]
G0074	Dragonfly 2.0	Dragonfly 2.0 added newly created accounts to the administrators group to maintain elevated access.[3][4]
G0032	Lazarus Group	Lazarus Group malware WhiskeyDelta-Two contains a function that attempts to rename the administrator's account.[5][6]
S0002	Mimikatz	The Mimikatz credential dumper has been extended to include Skeleton Key domain controller authentication bypass functionality. The `LSADUMP::ChangeNTLM` and `LSADUMP::SetNTLM` modules can also manipulate the password hash of an account without knowing the clear text value.[7][8]
G0034	Sandworm Team	Sandworm Team used the `sp_addlinkedsrvlogin` command in MS-SQL to create a link between a created account and other servers in the network.[9]
S0649	SMOKEDHAM	SMOKEDHAM has added created user accounts to local Admin groups.[10]

Mitigations

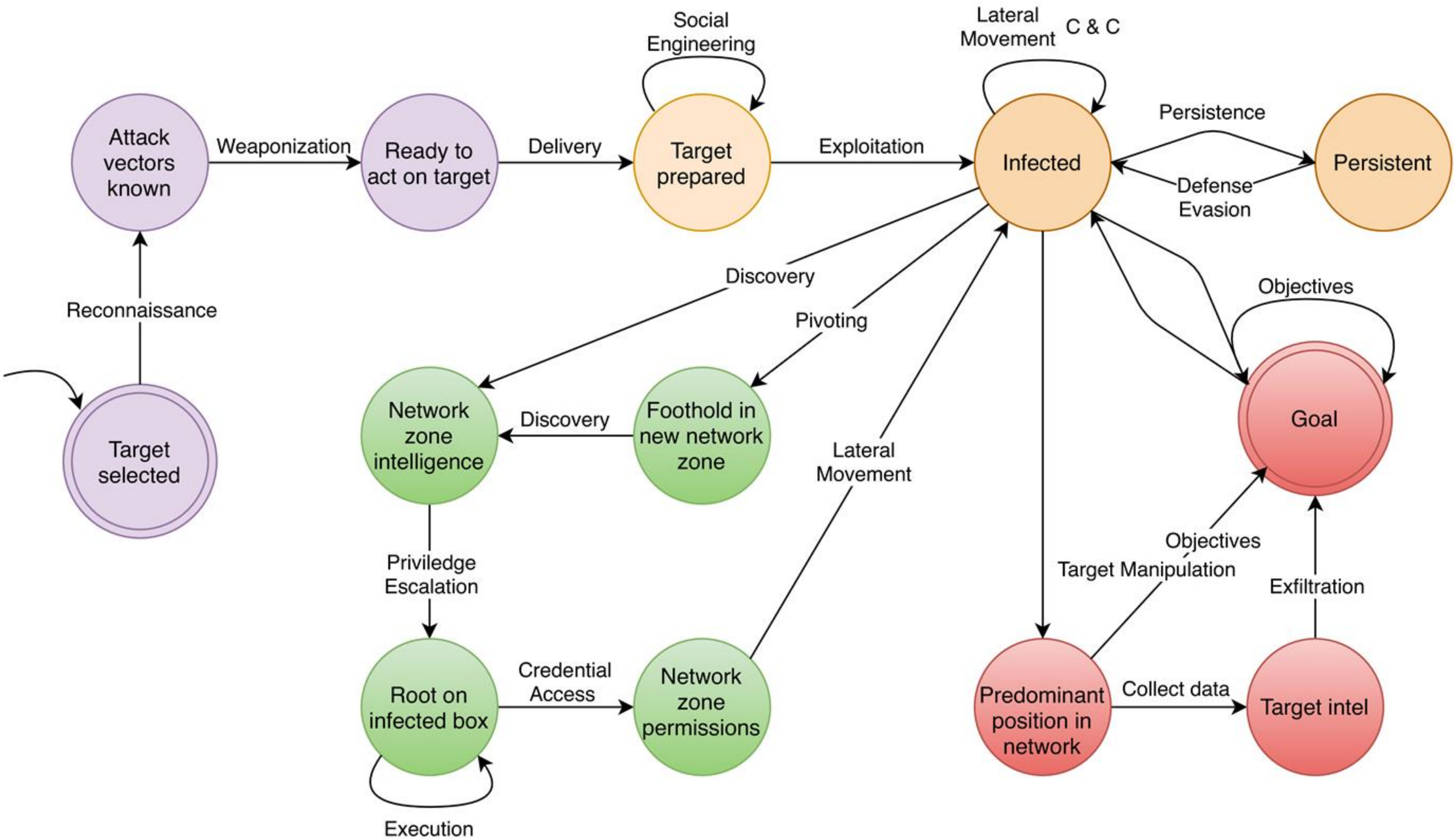
ID	Mitigation	Description
M1032	Multi-factor Authentication	Use multi-factor authentication for user and privileged accounts.
M1030	Network Segmentation	Configure access controls and firewalls to limit access to critical systems and domain controllers. Most cloud environments support separate virtual private cloud (VPC) instances that enable further segmentation of cloud systems.
M1028	Operating System Configuration	Protect domain controllers by ensuring proper security configuration for critical servers to limit access by potentially unnecessary protocols and services, such as SMB file sharing.
M1026	Privileged Account Management	Do not allow domain administrator accounts to be used for day-to-day operations that may expose them to potential adversaries on unprivileged systems.

Detection

ID	Data Source	Data Component
DS0026	Active Directory	Active Directory Object Modification
DS0017	Command	Command Execution
DS0022	File	File Modification
DS0036	Group	Group Modification

Our Work: Kill Chain State Machine (1)

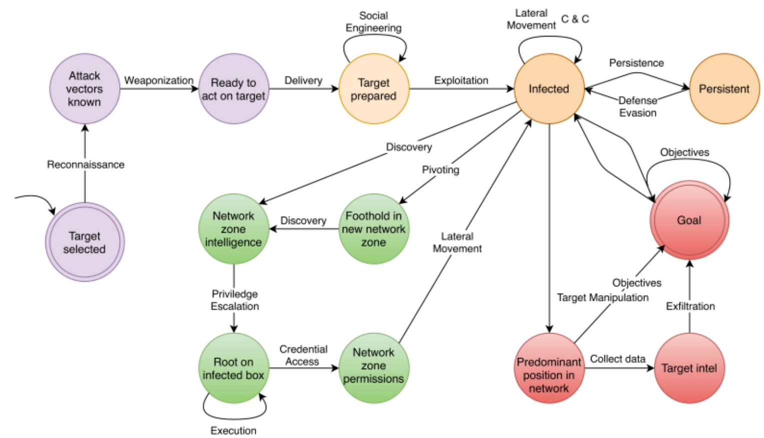
[WiOr+21]



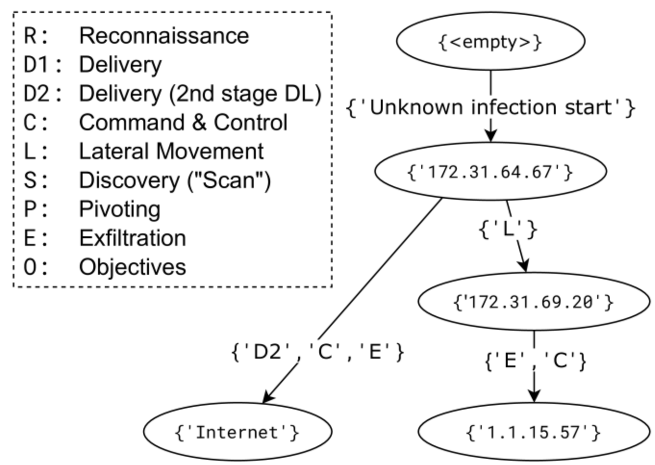
Our Work: Kill Chain State Machine (2)

[WiOr+21]

- State machine derived from UKC
  - Alerts → Transitions
  - Stages: Campaign progress



- Detection algorithm
  1. Maps alerts to transitions
  2. Connect transitions based on SM
  3. Deduplicate and optimize chains
  4. Prioritize scenarios based on length/complexity



- Currently network only, but extensible to other alert types

Our Work: Kill Chain State Machine (3) - Evaluation

- CSE-CIC-IDS2018 Dataset
- Realistically embedded (artificial) APT campaign

Table III. CSE-CIC-IDS2018: Overview

Property	Value
# Subnets/Zones	6 + *Internet*
# Target Hosts	450
# Attacker Hosts	50
# Connections	63 973 325
# (unrelated) attacks	7
Duration	10 days
Size in GB	559

Table IV. IDS2018-APT: Campaign overview

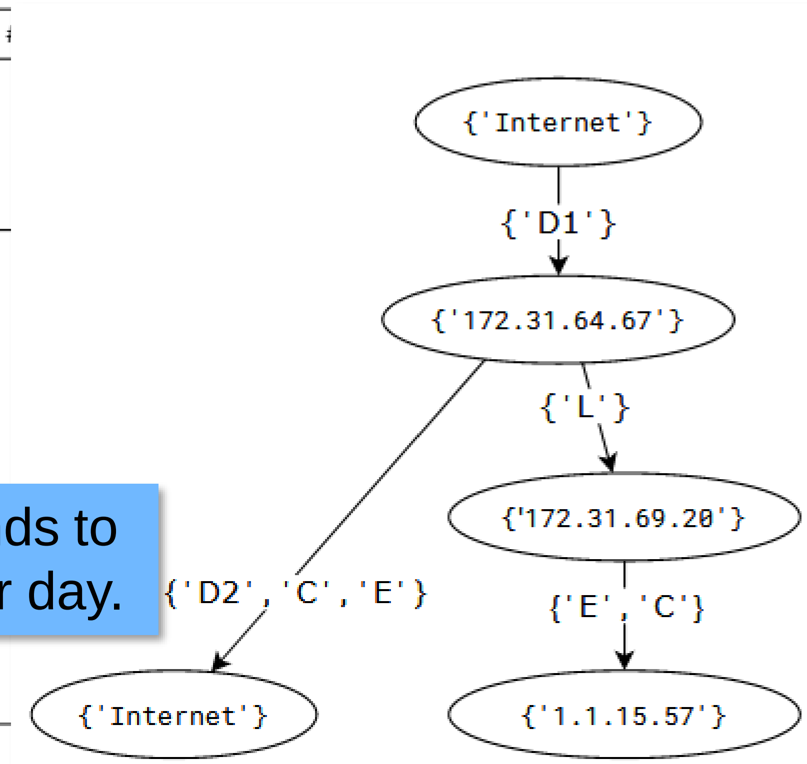
Day	Attack	Source	Target
1	*EternalRomance* RCE	1.1.13.37	172.31.64.67
1	2nd stage trojan download	172.31.64.67	12.34.12.34
4	Cosmic Duke C&C	172.31.64.67	1.1.14.47
8	PS-EXEC via SMB	172.31.64.67	172.31.69.20
10	Data exfiltration via HTTPS	172.31.69.20	1.1.15.57

Our Work: Kill Chain State Machine (4) - Evaluation

		IDS2018-APT-MIN			IDS2018-APT-FULL		
Source	Alert Type	# Alerts	APT related	Ratio	# Alerts	APT related	Ratio
Zeek Default	Conn::Retransmission_Inconsistency	1 171	0	0	1 171	0	0
	Scan::Address_Scan	1 555	0	0	1 555	0	0
	Scan::Port_Scan	38	0	0	38	0	0
	SSH::Password_Guessing	5	0	0	5	0	0
	SSL::Weak_Key	120	0	0	120	0	0
Custom Scenario Scripts	Custom::Stalled_HTTP_Connection	4 976	0	0	4 976	0	0
	Custom::HTTP_Windows_Executable_Download	13	0	0	13	0	0
	Custom::NON_HTTP_Windows_Executable_Download	8	2	0.25	8	2	0.25
	Custom::SMB_Executable_File_Transfer	1	1	1.00	1	1	1.00
	Custom::Javascript_Web_Injection_URI	5 934	0	0	5 934	0	0
	Custom::SQL_Web_Injection_URI	79	0	0	79	0	0
	Custom::Web_Login_Guessing	14	0	0	14	0	0
	Custom::Large_Outgoing_Tx	5 772	0	0	5 772	0	0
	Custom::Multiple_Large_Outgoing_Tx	187	0	0	187	0	0
	Custom::Very_Large_Outgoing_Tx	10	1	0.10	10	1	0.10
Mitre BZAR	ATTACK::Execution	—	—	—	2	2	1.00
	ATTACK::Lateral_Movement	—	—	—	4	4	1.00
	ATTACK::Lateral_Movement_and_Execution	—	—	—	1	0	0
	ATTACK::Lateral_Movement_Extracted_File	—	—	—	1	1	1.00
	ATTACK::Lateral_Movement_Multiple_Attempts	—	—	—	245	0	0
EternalSynergy	EternalSafety::DoublePulsar	—	—	—	1	1	1.00
	EternalSafety::EternalBlue	—	—	—	53	0	0
	EternalSafety::EternalSynergy	—	—	—	1	1	1.00
	EternalSafety::ViolationCmd	—	—	—	1 389	0	0
	EternalSafety::ViolationNtRename	—	—	—	8 731	0	0
	EternalSafety::ViolationPidMid	—	—	—	6 133	0	0
	EternalSafety::ViolationTx2Cmd	—	—	—	408 686	1	0.000002
	Total	19 883	3	0.000151	445 130	13	0.000029

Our Work: Kill Chain State Machine (5) - Evaluation

Source	Alert Type	IDS2018-APT-MIN			IDS2018-APT-FULL		
		#					Ratio
Zeek Default	Conn::Retransmission_Inconsistency						0
	Scan::Address_Scan						0
	Scan::Port_Scan						0
	SSH::Password_Guessing						0
	SSL::Weak_Key						0
om Scenario Scripts	Custom::Stalled_HTTP_Connection						0
	Custom::HTTP_Windows_Executable_Download						0
	Custom::NON_HTTP_Windows_Executable_Download						0.25
	Custom::SMB_Executable_File_Transfer						1.00
	Custom::Javascript_Web_Injection_URI						0
	Custom::SQL_Web_Injection_URI						0
	Custom::Web_Login_Guessing						0
	Custom::Large_Outgoing_Tx						0
							0
							0.10
Mitre							1.00
							1.00
	ATTACK::Lateral_Movement_and_Execution						0
	ATTACK::Lateral_Movement_Extracted_File						1.00
	ATTACK::Lateral_Movement_Multiple_Attempts						0
EternalSynergy	EternalSafety::DoublePulsar						1.00
	EternalSafety::EternalBlue	—	—	—	53	0	0
	EternalSafety::EternalSynergy						1.00
	EternalSafety::ViolationCmd						0
	EternalSafety::ViolationNtRename						0
	EternalSafety::ViolationPidMid	—	—	—	6 133	0	0
	EternalSafety::ViolationTx2Cmd	—	—	—	408 686	1	0.000002
	Total	19 883	3	0.000151	445 130	13	0.000029



686 scenario graphs corresponds to investigating 68,6 scenarios per day.

Reconstructed APT campaign

Evasion Techniques to Bypass IDS

- Signature Evasion
  - Attack Obfuscation
  - Duplicate Insertion
  - Packet Splitting
  - Packet Overlapping

- Anomaly Evasion
  - Training Data Injection
  - Mimicry Attacks
  - Covert Channel Attacks

Signature Evasion - Attack Obfuscation

- Transformation of malicious code into semantically equivalent one
- As the signature will defer from the original it will not be detected

Depending on the level of mutation

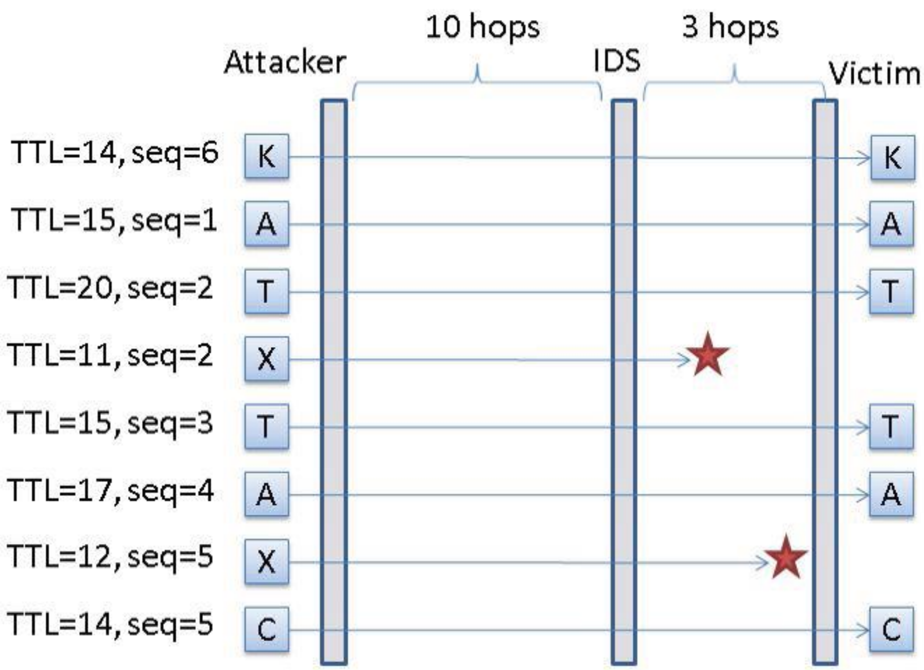
- Payload mutation
  - Change the signature of the payload of the packet

- Shellcode mutation
  - Obfuscate the shellcode with polymorphic techniques
  - Easily done via popular penetration testing tools like Metasploit Framework

Signature Evasion – Duplicate Insertion

- IDS and possible victim may handle duplicated or fragments differently
- IDS lacks information about network topology and operating system of victim

- Duplicate Insertion
  - Attacker inserts some segments with small TTL, so that they will be dropped before victim
  - If IDS cannot predict whether segments reach victim, it will not be able to reassemble segments and see same content as victim



Anomaly Evasion – Mimicry Attacks

- Attack transformation by imitating normal activity
- Usually achieved with the insertion of "dummy" system calls
- Final system sequence looks normal

```
setreuid(0,0), chroot("pub"),
 chdir("../../../../../../../../../"), chroot("/"),
 open("/etc/passwd", O_APPEND|O_WRONLY),
 write(fd, "toor:AAaaaaaaaaaaa:0:0::/:/bin/sh", 33),
 close(fd), exit(0)
```

```
read() write() close() munmap() sigprocmask() wait4()
sigprocmask() sigaction() alarm() time() stat() read()
alarm() sigprocmask() setreuid() fstat() getpid()
time() write() time() getpid() sigaction() socketcall()
sigaction() close() flock() getpid() lseek() read()
kill() lseek() flock() sigaction() alarm() time()
stat() write() open() fstat() mmap() read() open()
fstat() mmap() read() close() munmap() brk() fcntl()
setregid() open() fcntl() chroot() chdir() setreuid()
lstat() lstat() lstat() lstat() open() fcntl() fstat()
lseek() getdents() fcntl() fstat() lseek() getdents()
close() write() time() open() fstat() mmap() read()
close() munmap() brk() fcntl() setregid() open() fcntl()
chroot() chdir() setreuid() lstat() lstat() lstat()
lstat() open() fcntl() brk() fstat() lseek() getdents()
lseek() getdents() time() stat() write() time() open()
getpid() sigaction() socketcall() sigaction() umask()
sigaction() alarm() time() stat() read() alarm()
getrlimit() pipe() fork() fcntl() fstat() mmap() lseek()
close() brk() time() getpid() sigaction() socketcall()
sigaction() chdir() sigaction() sigaction() write()
munmap() munmap() munmap() exit()
```

Summary

- IDS
  - Signature-based vs. policy-based vs. anomaly-based IDS
  - In combination with Firewalls: IPS
  - Classification according to kind of sensors deployed, level of distribution

- IDS problems
  - Huge amounts of data to process
  - Limited accuracy and large number of false positives
  - Privacy
  - IDS evasion techniques

- Alert correlation to obtain the bigger picture of attacks
  - Alert correlation process
  - Cyber Kill Chain and MITRE ATT&CK
  - Alert Correlation

Additional References

[Zwi00a] E. Zwicky, S. Cooper, B. Chapman. *Building Internet Firewalls.* Second Edition, O'Reilly, 2000.

[Sem96a] C. Semeria. *Internet Firewalls and Security.* 3Com Technical Paper, 1996.

[Wack95a] J. P. Wack, L.J. Carnahan. *Keeping Your Site Comfortably Secure: An Introduction to Internet Firewalls.* NIST Special Publication 800-10, 1995.

[StaBro08] William Stallings and Lawrie Brown. Computer Security: Principles and Practice. Prentice Hall Press, Upper Saddle River, NJ, USA, 2nd edition, 2008.

[Bar01] P. Barford, D. Plonka, Characteristics of Network Traffic Flow Anomalies. Proceedings of ACM SIGCOMM Internet Measurement Workshop, October 2001.

[Chan09] Chandola, Varun, Arindam Banerjee, and Vipin Kumar. 2009. "Anomaly Detection: A Survey." ACM Computing Surveys 41 (3) (July 1): 1–58. doi:10.1145/1541880.1541882.

[Man2015] Mandiant, M-Trends® 2015: A View from the Front Lines https://www2.fireeye.com/WEB-2015-MNDT-RPT-M-Trends-2015_LP.html

[Sym2016] Symantec, Internet Security Threat Report, 2016, https://www.symantec.com/security-center/threat-report

[Yo2012] Yves Younan, 25 Years of Vulnerabilities: 1988-2012, RESEARCH REPORT, Sourcefire Vulnerability Research Team (VRT)

[Pa99] V. Paxson, "Bro: a system for detecting network intruders in real-time," in USENIX Security Symposium, 1999, vol. 7, pp. 1–22.

[VaKa+15] E. Vasilomanolakis, S. Karuppayah, M. Muehlhaeuser, and M. Fischer, "Taxonomy and Survey of Collaborative Intrusion Detection," ACM Computing Surveys, pp. 1–35, 2015.

Additional References

[Ju03] Julisch, K. (2003). Clustering intrusion detection alarms to support root cause analysis. ACM Transactions on Information and System Security (TISSEC), 6(4), 443–471. https://doi.org/10.1145/950191.950192

[Han92] HAN, J., CAI,Y., AND CERCONE, N. 1992. Knowledge discovery in databases: An attribute-oriented approach. In 18th International Conference on Very Large Databases, 547–559

[ZhoLe+10] C. V. Zhou, C. Leckie, and S. Karunasekera, “A survey of coordinated attacks and collaborative intrusion detection,” Comput. Secur., vol. 29, no. 1, pp. 124–140, 2010.

[HuCl10] E.M. Hutchins, M.J. Cloppert and R.M Amin PH.D., "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains," Proc. 6th Int'l Conf. Information Warfare and Security(ICIW 11), Academic Conferences Ltd., 2010, pp. 113–125; http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/LM-White-Paper-Intel-Driven-Defense.pdf

[WiOr+21] F. Wilkens, F. Ortmann, S. Haas, M. Vallentin, and M. Fischer, “Multi-Stage Attack Detection via Kill Chain State Machines,” Submitted to: ACM Conference on Computer and Communications Security (CCS), 2021.

[HaFi18] S. Haas and M. Fischer, “GAC: Graph-Based Alert Correlation for the Detection of Distributed Multi-Step Attacks”, ACM Symposium on Applied Computing (SAC), 2018.

[HaFi+19] S. Haas and M. Fischer, “On the alert correlation process for the detection of multi-step attacks and a graph-based realization”, ACM SIGAPP Applied Computing Review 19, 1 (April 2019), 5-19

[WiHa+21] F. Wilkens, S. Haas, J. Amann, M. Fischer, ”Passive, Transparent, and Selective TLS Decryption for Network Security Monitoring”, arXiv.org, 2021 https://arxiv.org/abs/2104.09828