Cybersecurity at ntop: Present and Future

Luca Deri <deri@ntop.org>
@lucaderi

ntop

Part I:
Past and Present

20+ Years of Network Monitoring

- Increased speed:
  - 10 Gbit is now commodity for many companies.
  - 100 Gbit is standard for ISPs.
- Monitoring Protocols
  - Still NetFlow and sFlow, just at higher speed.
  - Packet/Flow sampling prevents full visibility.
- Monitoring Metrics
  - Bytes and packets are still the main metrics for many network vendors.

IDSs and ML [1/2]

- Traditional IDS, often based on signatures and rule-based approaches shown their limitations in detection capability, especially when attackers heavily rely on encryption to obfuscate communications.
- While we do believe that ML (machine learning) technologies are playing (and will play in the future) an important role in cybersecurity, we strongly believe that domain knowledge and feature engineering have tremendous value for any detection problem.

IDSs and ML [2/2]

- Increasing adoption of encryption technologies, DPI (deep packet inspection) can be used to extract very strong signals from the raw traffic.
- While one could feed those signals to ML-based detectors, we highlight that when strong signals are available, one can greatly profit from them even with less sophisticated data processing technologies.
- This presentation shows how real-time, DPI-based cyber threat detection is feasible and effective using the concepts that will be explained later.

Signature-based IDSs (1998-Today)

alert tcp any any -> any [443,465] (msg:"Detected non-TLS on TLS port"; flow:to_server; app-layer-protocol:!tls; threshold: type limit, track by_src, seconds 90, count 1; sid: 210003; rev:1;)

alert tcp any any <> any 443 (msg:"APT.Backdoor.MSIL.SUNBURST"; content:"|16 03|"; depth:2; content:"|55 04 03|"; distance:0; content:"digitalcollege.org"; within:50; sid:77600846; rev:1;)

- Techniques easy to circumvent.
- No application protocol visibility (packet header only, byte-based payload analysis).
- Outdated and error-prone format (“proto=TLS and SNI=digitalcollege.org”).

Cybersecurity and Network Edge [1/2]

- Today most traffic is encrypted (80%+) and traditional clear-text protocols are moving to encryption (e.g. DNS vs DNS-over-HTTPS).
- As edge network speed is increasing, security threats on customer networks can propagate the issue to the core.
- Insecure devices (e.g. simple IoT devices) are placed in privileged network segments, thus requiring accurate supervision as they can cause severe troubles in case of breach.

Cybersecurity and Network Edge [2/2]

- Data centers with unhealthy customer traffic can affect neighbours and decrease the whole network reputation score.
- Limiting traffic observability to bandwidth usage is no longer wise: it is time to monitor customer traffic in an unobtrusive way in order to report users all threats they have not detected, mitigate issues and thus implement a healthier Internet.
- In essence we need to implement a lightweight (Raspberry an up, no GPU or GB of RAM) and scalable system able to model and analyse network traffic on a per-device basis, and being able to track device changes in behaviour.

Welcome to nDPI

- In 2012 we decided to develop our own GNU LGPL DPI toolkit order to build an open source DPI layer.
- Protocols supported exceed 250+ and include:
  - P2P (BitTorrent)
  - Messaging (Viber, Whatsapp, Telegram, Facebook)
  - Multimedia (YouTube, Last.gm, iTunes)
  - Conferencing (Skype, Webex, Teams, Meet, Zoom)
  - Streaming (Zattoo, Disney, Netflix)
  - Business (VNC, RDP, Citrix)
  - Gaming

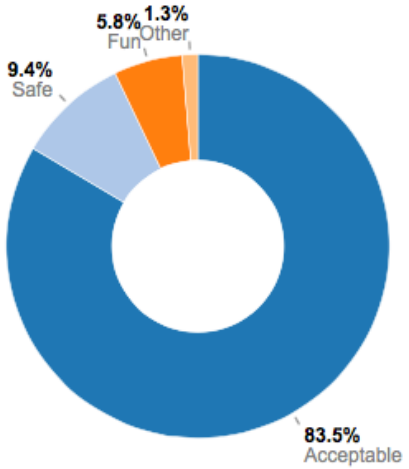
GitHub https://github.com/ntop/nDPI

nDPI Traffic Analysis

Layer 4 Protocol

Good or Bad?

TCP / HTTP



Layer 7 Protocol

nDPI in Cybersecurity

- Analyses encrypted traffic to detect issues un-inspectable due to encrypted payload content.
- Extracts metadata from selected protocols (e.g. DNS, HTTP, TLS..) and matches it against known algorithms for detecting selected threats (e.g. DGA hosts, Domain Generated Algorithm).
- Associates a “flow risk” with specific flows to identify communications that are affected by security issues.

nDPI: Flow Risks

- HTTP suspicious user-agent
- HTTP numeric IP host contacted
- HTTP suspicious URL
- HTTP suspicious protocol header
- TLS connections not carrying HTTPS (e.g. a VPN over TLS)
- Suspicious DGA domain contacted
- Malformed packet
- SSH/SMB obsolete protocol or application version
- TLS suspicious ESNI usage
- Unsafe Protocol used
- Suspicious DNS traffic
- TLS with no SNI
- XSS (Cross Site Scripting)
- SQL Injection
- Arbitrary Code Injection/Execution
- Binary/.exe application transfer (e.g. in HTTP)
- Known protocol on non standard port
- TLS self-signed certificate
- TLS obsolete version
- TLS weak cipher
- TLS certificate expired
- TLS certificate mismatch
- DNS suspicious traffic
- HTTP suspicious content
- Risky ASN
- Risky Domain Name
- Malicious JA3 Fingerprint
- Malicious SHA1 Certificate
- Desktop of File Sharing Session
- TLS Uncommon ALPN

Legenda: Clear Text Only, Encrypted/Plain Text, Encrypted Only

- TLS Certificate Validity Too Long
- Suspicious TLS Extension
- TLS Fatal Alert
- Suspicious Protocol traffic Entropy
- Clear-text Credentials Exchanged
- DNS Large Packet
- DNS Fragmented Traffic
- Invalid Characters Detected

Implemented Recently
(nDPI is a live project !)

ntop

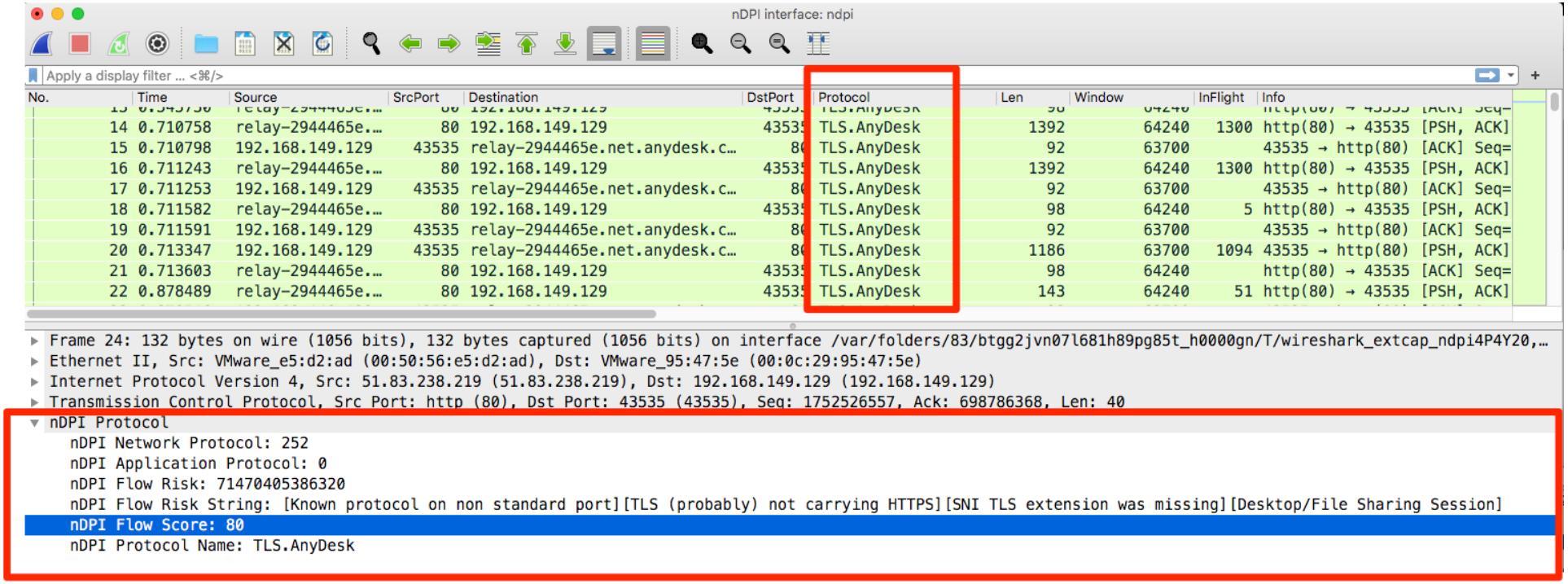
nDPI Encrypted Traffic Analysis

```
TCP 10.9.25.101:49184 <-> 187.58.56.26:449 [byte_dist_mean: 124.148883][byte_dist_std:
58.169660][entropy: 5.892724][total_entropy: 7124.302784][score: 0.9973][proto: 91/TLS]
[cat: Web/5][97 pkts/36053 bytes <-> 159 pkts/149429 bytes][Goodput ratio: 85/94][111.31
sec][bytes ratio: -0.611 (Download)][IAT c2s/s2c min/avg/max/stddev: 0/0 1129/662
19127/19233 2990/2294][Pkt Len c2s/s2c min/avg/max/stddev: 54/54 372/940 1514/1514
530/631][Risk: ** Self-signed Certificate **** Obsolete TLS version (< 1.1) **][TLSv1]
[JA3S: 623de93db17d313345d7ea481e7443cf][Issuer: C=AU, ST=Some-State, O=Internet Widgits
Pty Ltd][Subject: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd][Certificate SHA-1:
DD:EB:4A:36:6A:2B:50:DA:5F:B5:DB:07:55:9A:92:B0:A3:52:5C:AD][Validity: 2019-07-23 10:32:39
- 2020-07-22 10:32:39][Cipher: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA]

TCP 10.9.25.101:49165 <-> 144.91.69.195:80 [byte_dist_mean: 95.694525][byte_dist_std:
25.418150][entropy: 0.000000][total_entropy: 0.000000][score: 0.9943][proto: 7/HTTP][cat:
Web/5][203 pkts/11127 bytes <-> 500 pkts/706336 bytes][Goodput ratio: 1/96][5.18 sec]
[Host: 144.91.69.195][bytes ratio: -0.969 (Download)][IAT c2s/s2c min/avg/max/stddev: 0/0
23/9 319/365 49/37][Pkt Len c2s/s2c min/avg/max/stddev: 54/54 55/1413 207/1514 11/134]
[URL: 144.91.69.195/solar.php][StatusCode: 200][ContentType: application/octet-stream]
[UserAgent: pwtyyEKzNtGatwnJjmCcBLbOveCVpc][Risk: ** Binary application transfer **][PLAIN
TEXT (GET /solar.php HTTP/1.1)]
```

Trickbot Traffic

nDPI in Wireshark



From Flow Risk To Score [1/2]

nDPI supported risks:

Id	Risk	Severity	Score	CliScore	SrvScore
1	XSS attack	Severe	250	225	25
2	SQL injection	Severe	250	225	25
3	RCE injection	Severe	250	225	25
4	Binary application transfer	Severe	250	125	125
5	Known protocol on non standard port	Medium	50	25	25
6	Self-signed Certificate	High	100	90	10
7	Obsolete TLS version (older than 1.2)	High	100	90	10
8	Weak TLS cipher	High	100	90	10
9	TLS Expired Certificate	High	100	50	50
10	TLS Certificate Mismatch	High	100	50	50
11	HTTP Suspicious User-Agent	High	100	90	10
12	HTTP Numeric IP Address	Low	10	5	5
13	HTTP Suspicious URL	High	100	90	10
14	HTTP Suspicious Header	High	100	90	10
15	TLS (probably) not carrying HTTPS	Low	10	5	5
16	Suspicious DGA domain name	High	100	90	10
17	Malformed packet	Low	10	5	5
18	SSH Obsolete Client Version/Cipher	High	100	90	10
19	SSH Obsolete Server Version/Cipher	Medium	50	5	45
20	SMB Insecure Version	High	100	90	10
21	TLS Suspicious ESNI Usage	Medium	50	25	25
22	Unsafe Protocol	Low	10	5	5
23	Suspicious DNS traffic	High	100	90	10
24	SNI TLS extension was missing	Medium	50	25	25
25	HTTP suspicious content	High	100	90	10
26	Risky ASN	Medium	50	25	25
27	Risky domain name	Medium	50	25	25
28	Possibly Malicious JA3 Fingerprint	Medium	50	25	25
29	Possibly Malicious SSL Cert. SHA1 Fingerprint	Medium	50	25	25
30	Desktop/File Sharing Session	Low	10	5	5
31	Uncommon TLS ALPN	Medium	50	25	25
32	TLS certificate validity longer than 13 months	Medium	50	25	25
33	TLS suspicious extension	High	100	90	10
34	TLS fatal alert	Low	10	5	5
35	Suspicious entropy	Medium	50	25	25
36	Clear-text credentials	High	100	90	10
37	DNS packet larger than 512 bytes	Medium	50	25	25
38	Fragmented DNS message	Medium	50	25	25
39	Text contains non-printable characters	High	100	90	10

From Flow Risk To Score [2/2]

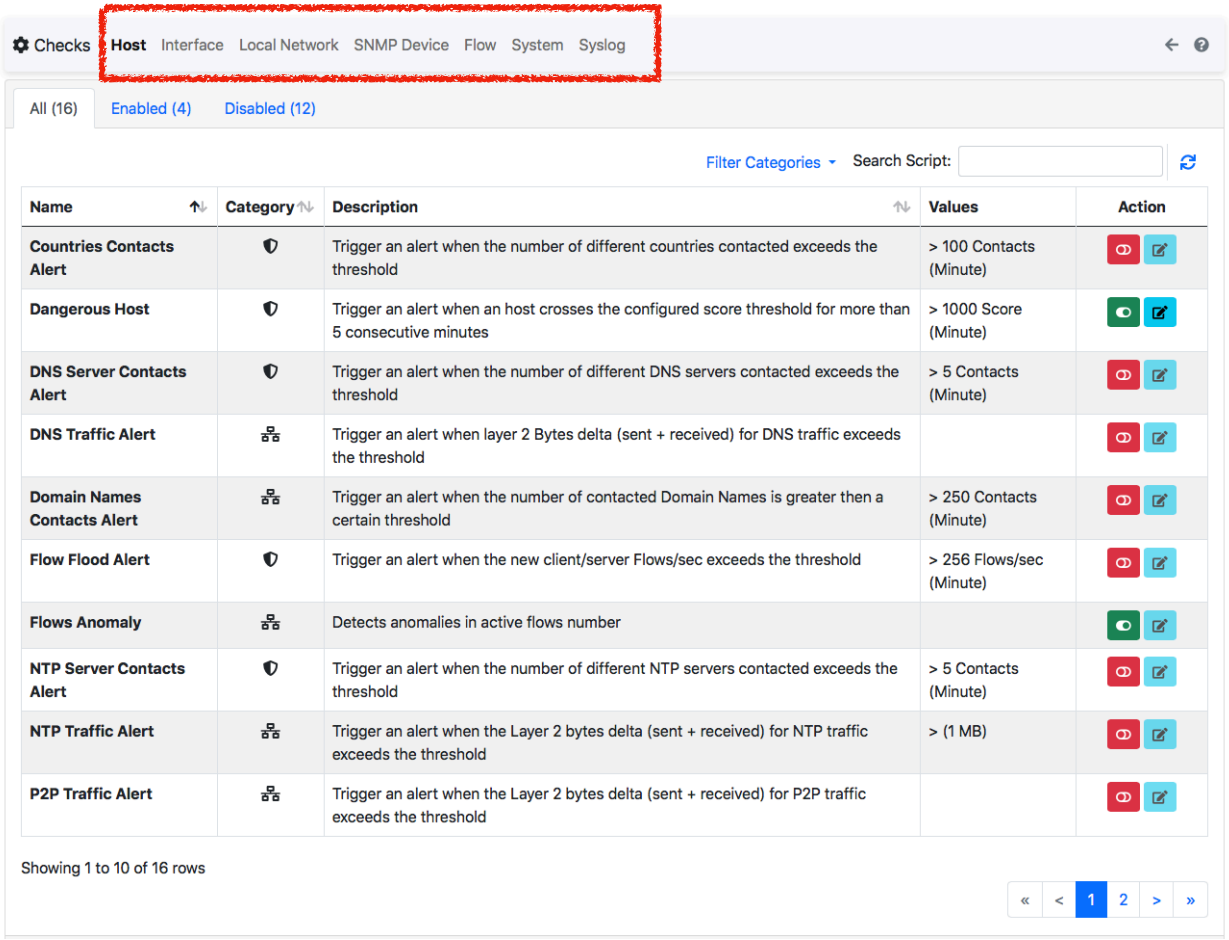
TCP 192.168.149.129:43535 <-> 51.83.238.219:80 [proto: 91.252/**TLS.AnyDesk**][cat: RemoteAccess/12][2942 pkts/175103 bytes <-> 4001 pkts/2618640 bytes][Goodput ratio: 9/92][55.97 sec][bytes ratio: -0.875 (Download)][IAT c2s/s2c min/avg/max/stddev: 0/0 19/14 7028/7028 153/126][Pkt Len c2s/s2c min/avg/max/stddev: 54/60 60/654 1514/1514 50/618][Risk: ** Known protocol on non standard port **** TLS (probably) not carrying HTTPS **** SNI TLS extension was missing **** Desktop/File Sharing Session **][**Risk Score: 80**][TLSv1.2][JA3C: 201999283915cc31cee6b15472ef3332][JA3S: 107030a763c7224285717ff1569a17f3][Issuer: CN=AnyNet Root CA, O=philandro Software GmbH, C=DE][Subject: C=DE, O=philandro Software GmbH, CN=AnyNet Relay][Certificate SHA-1: 9E:08:D2:58:A9:02:CD:4F:E2:4A:26:B8:48:5C:43:0B:81:29:99:E3][Firefox][Validity: 2018-11-18 02:14:23 – 2028-11-15 02:14:23][Cipher: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384][Plen Bins: 0,7,17,2,1,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,3,0,0,33,0,0,0,0,29,0,0]

Detected Risk	**Risk Score Value**
Known protocol on non standard port	10
TLS (probably) not carrying HTTPS	10
SNI TLS extension was missing	50
Desktop/File Sharing Session	10
Flow Score Total	80

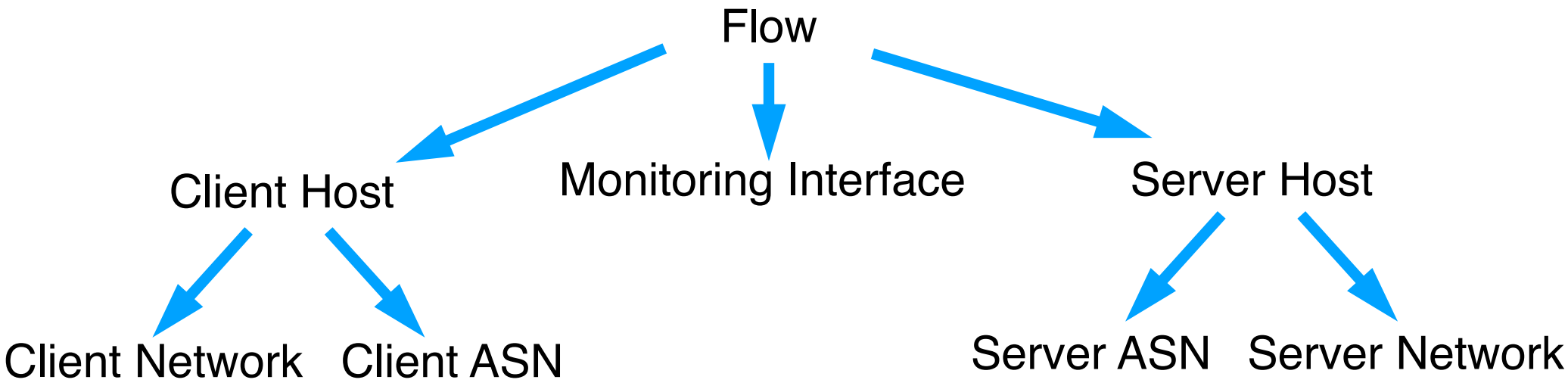
Consolidating Score [1/3]

- Flow traffic analysis is too granular and it needs to be consolidated into:
  - Network Interface
  - Host/Network/Customer.
  - ASN/Country
- In essence that is the pillar for creating a (client/server) numerical score that can be quickly used to spot issues (network, security…).

Consolidating Score [2/3]



Consolidating Score [3/3]



- Flow score is computed in realtime (flow lifetime)
- (Host/Interface/….) Checks are performed every minute

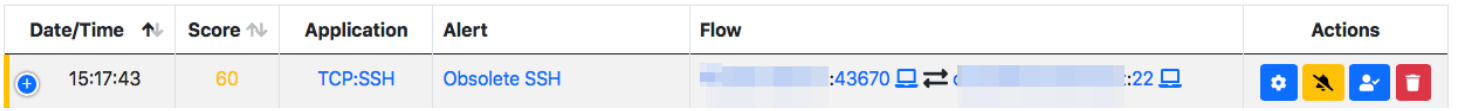
What about Risk Exceptions ? [1/3]

- Many cybersecurity products are very strict with policies and they divide the world in good and bad.
- Unfortunately reality is a bit more complicated (indeed grey exists), and “modern” needs to coexist with “ancient” that in computing terms can be just a few years old.
- The score principle is effective only if there are no false positives as otherwise they can deceive detection algorithms by generating false alerts.

What about Risk Exceptions ? [2/3]

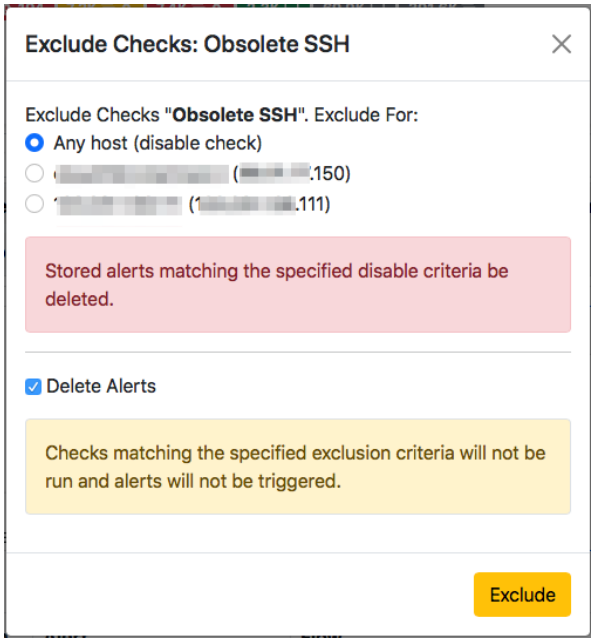
- A few typical exception examples:
  - Private IPs with self-signed TLS certificates.
  - Insecure protocols/hosts that cannot be upgraded but that provide a specific service to a few clients.
  - Applications running on non standard ports (e.g. SSH server on port 2222).
  - TLS towards numeric IP address (no symbolic hostname).

What about Risk Exceptions ? [3/3]

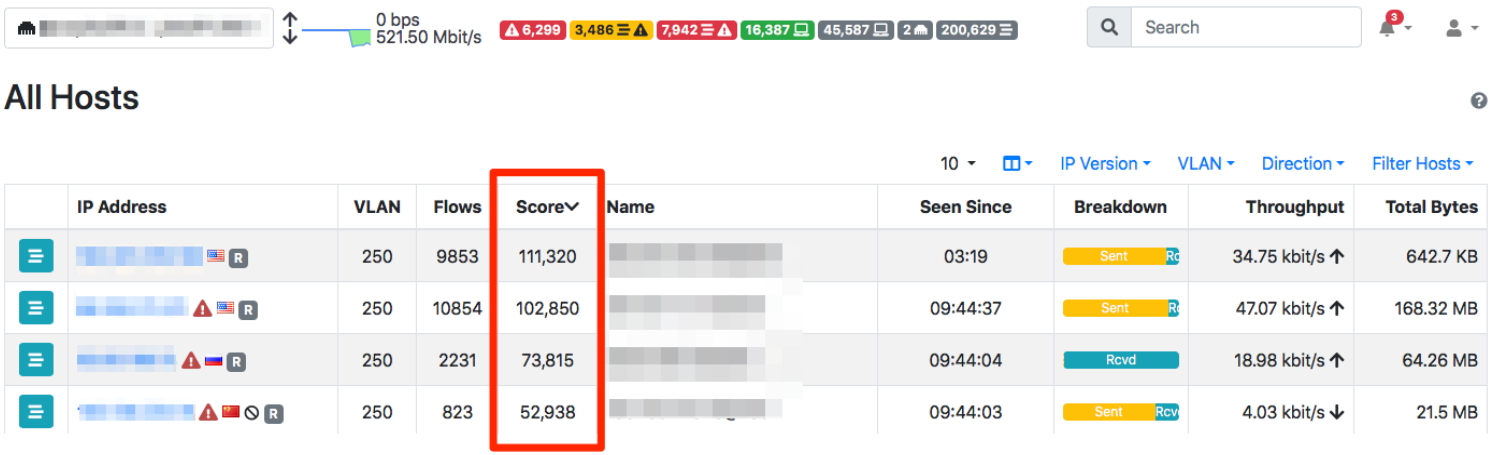


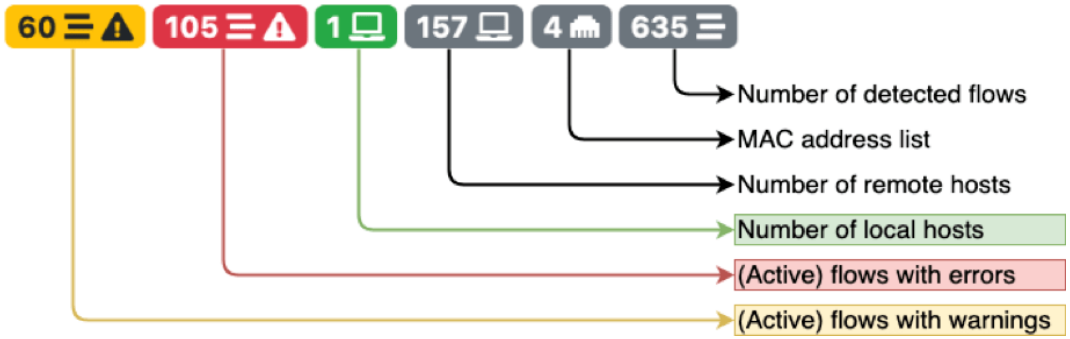
Available options:

- Disable Check (for everybody).
- Exclude the check for a specific host.
- Acknowledge the alert

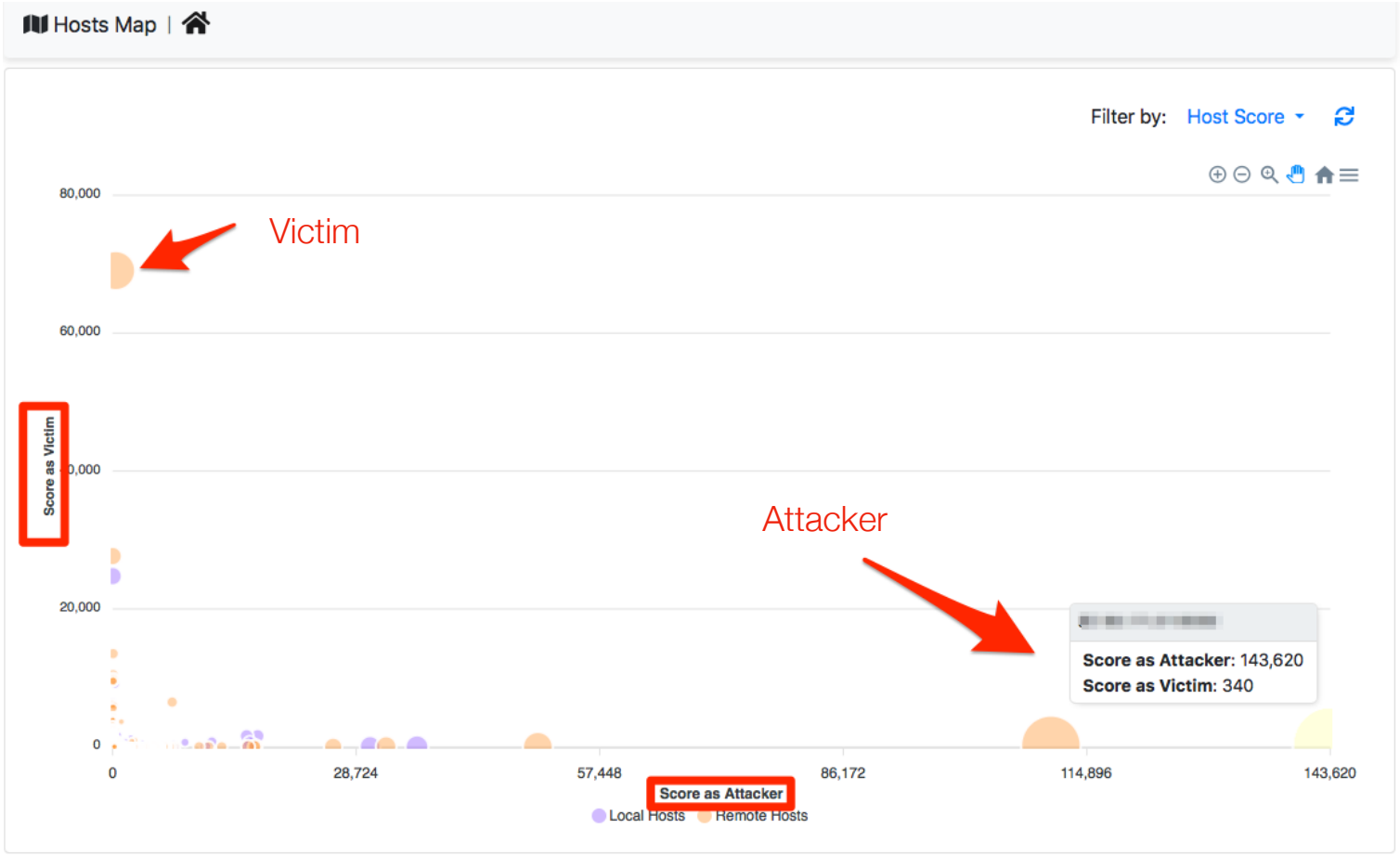


Score At Work



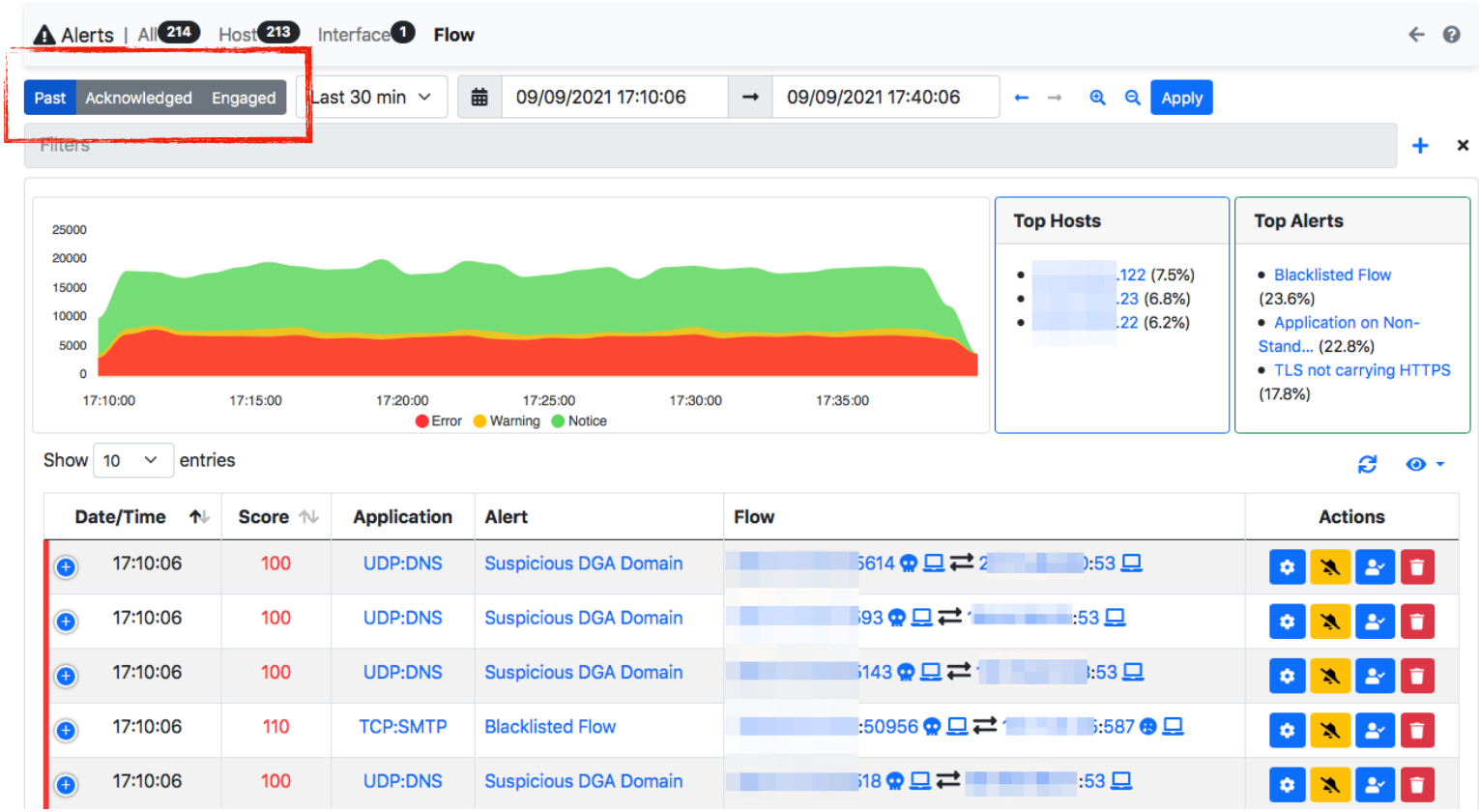


Visualising Cybersecurity: Bubbles



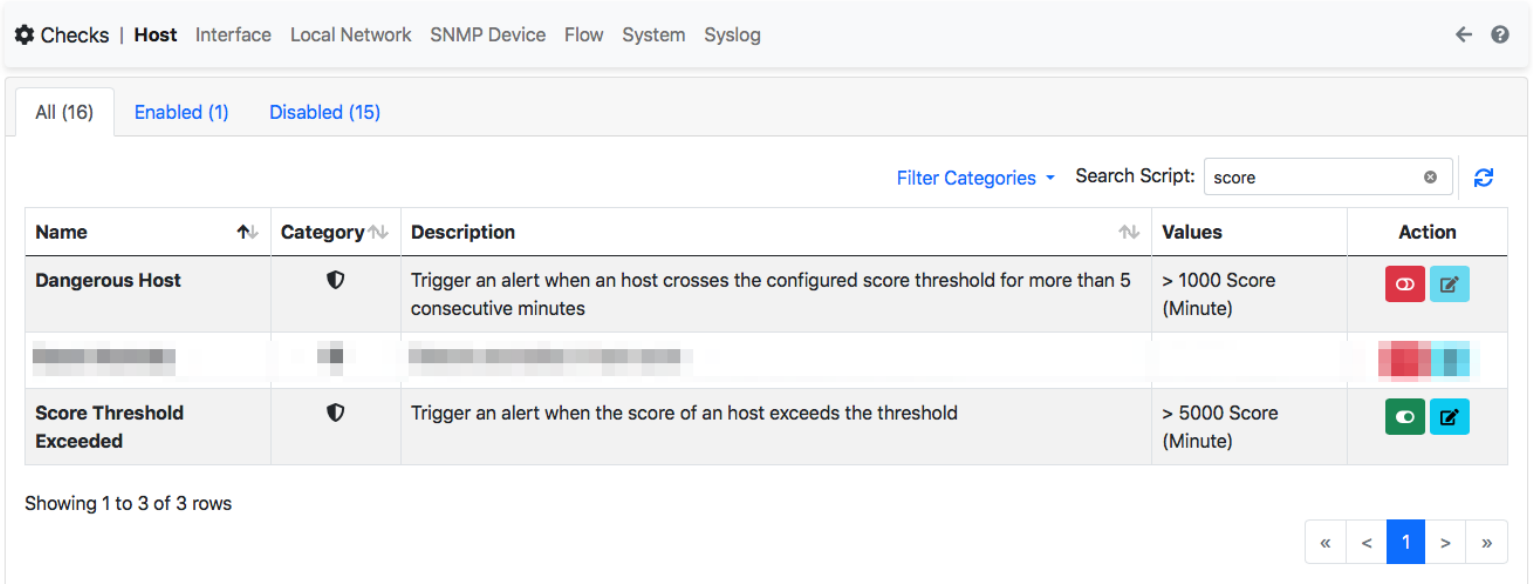
Score-based Alerts [1/2]

Alarm Lifecycle



Score-based Alerts [2/2]

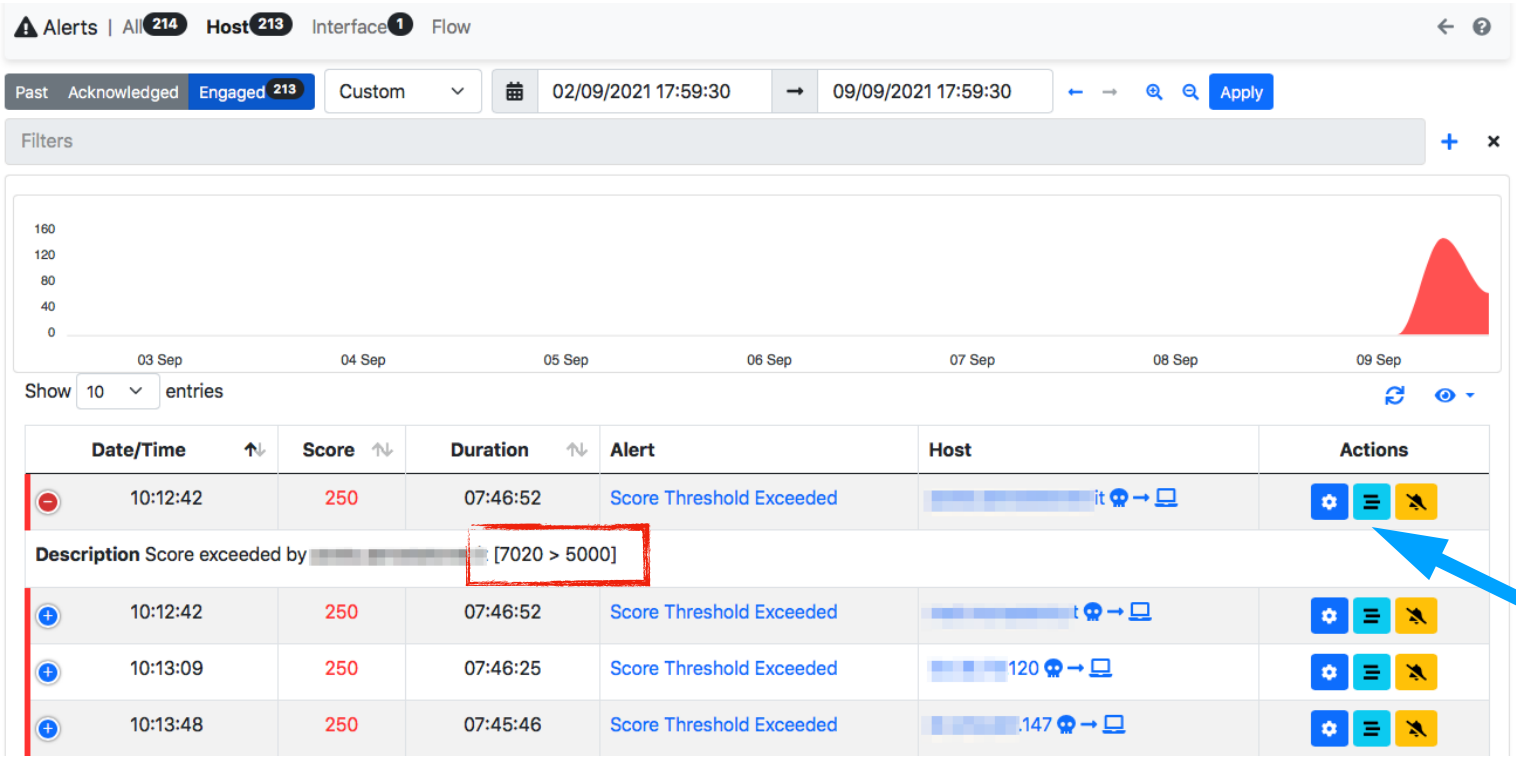
Threshold-based Score Alerts [1/2]



Simple to use for detecting hosts with high score:

- Continuously
- Score spikes

Threshold-based Score Alerts [2/2]



Flows
(Drill Down)

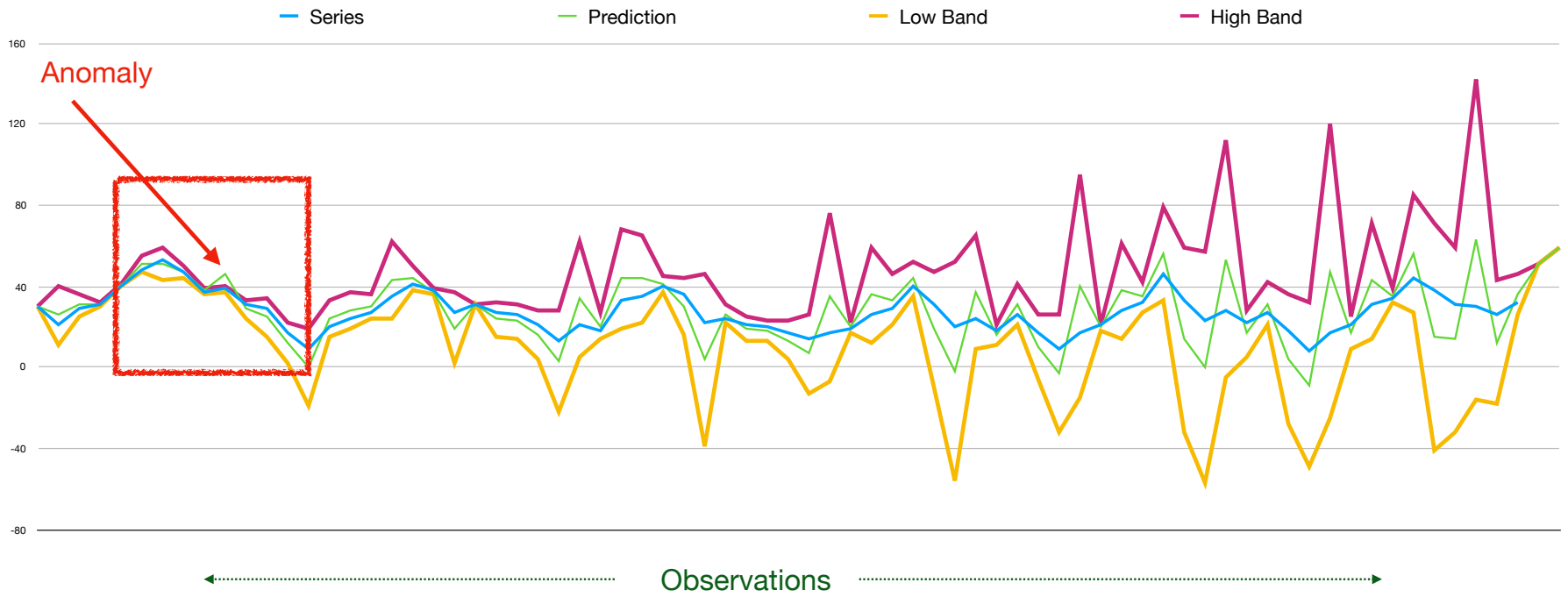
Score-based Behaviour Analysis [1/5]

- Thresholds are useful to spot issues that can be identified with boundaries.
- However
  - How do you define a typical host threshold? Not all hosts behave the same way.
  - How can I detect changes in behaviour? A host can double its score and still be unalarmed, but the network operator needs to be informed that something has changed.

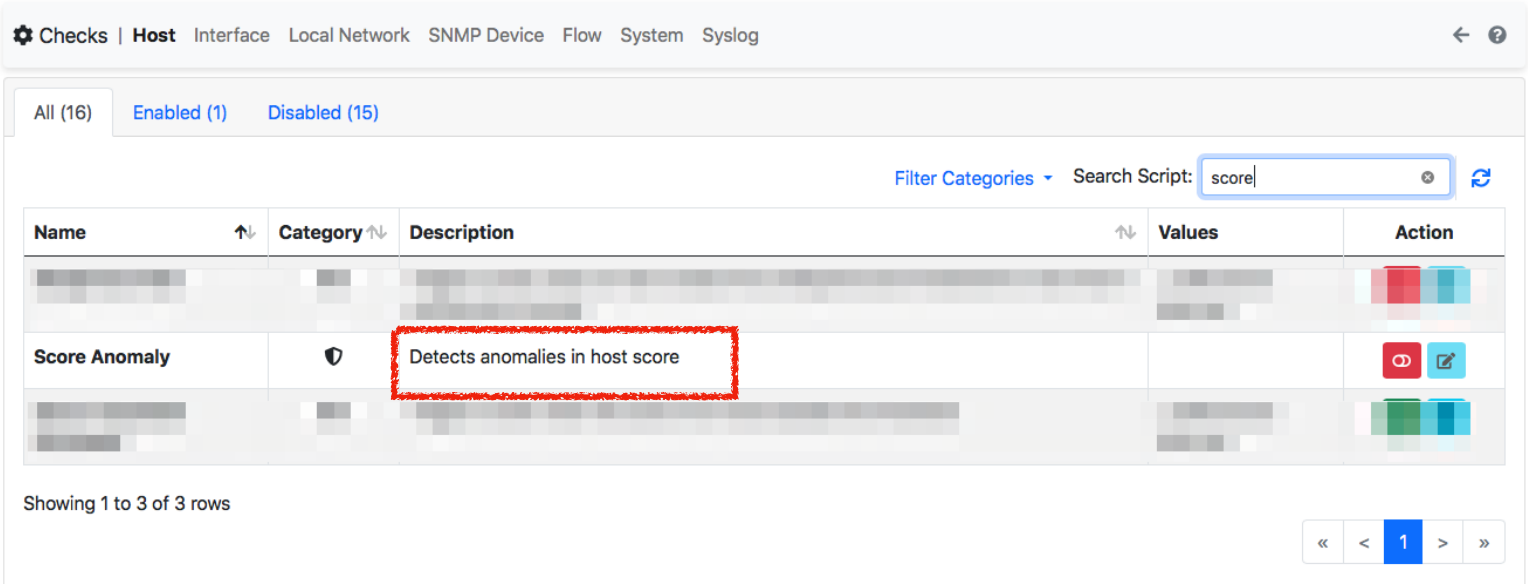
Score-based Behaviour Analysis [2/5]

- Without having to disturb ML that can be heavy for many users, we have decided to use (mature) statistical methods for spotting these changes.
- The advantage of statistical methods is that we can create a lightweight model per metric (hosts have tent of metrics) that uses little memory and CPU.
- For the record, we have used DES (Double Exponential Smoothing) that implements data forecasting and high/lower band for detecting changes in behaviour.

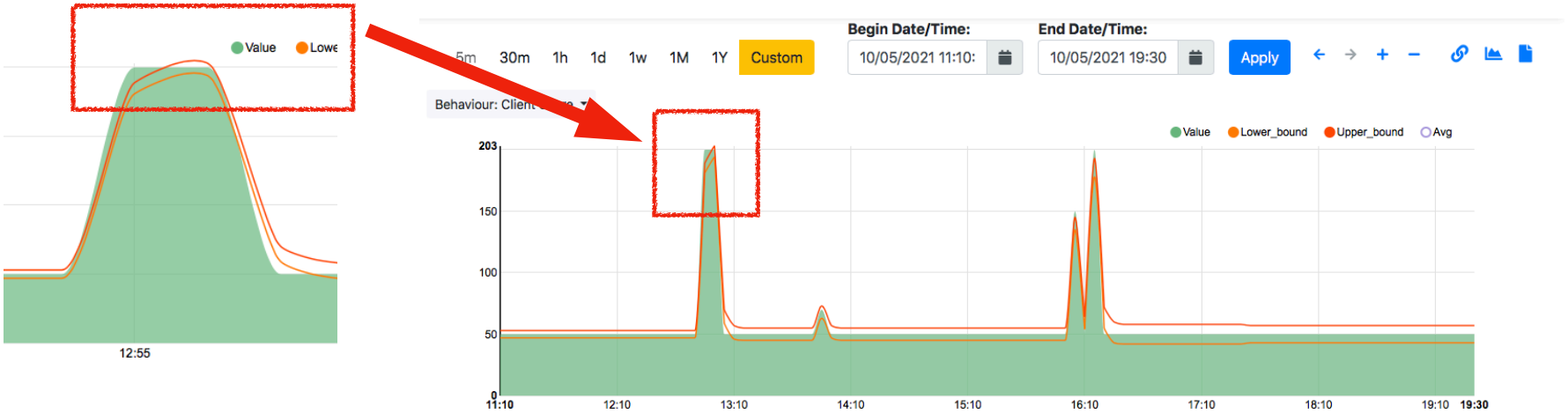
Score-based Behaviour Analysis [3/5]



Score-based Behaviour Analysis [4/5]



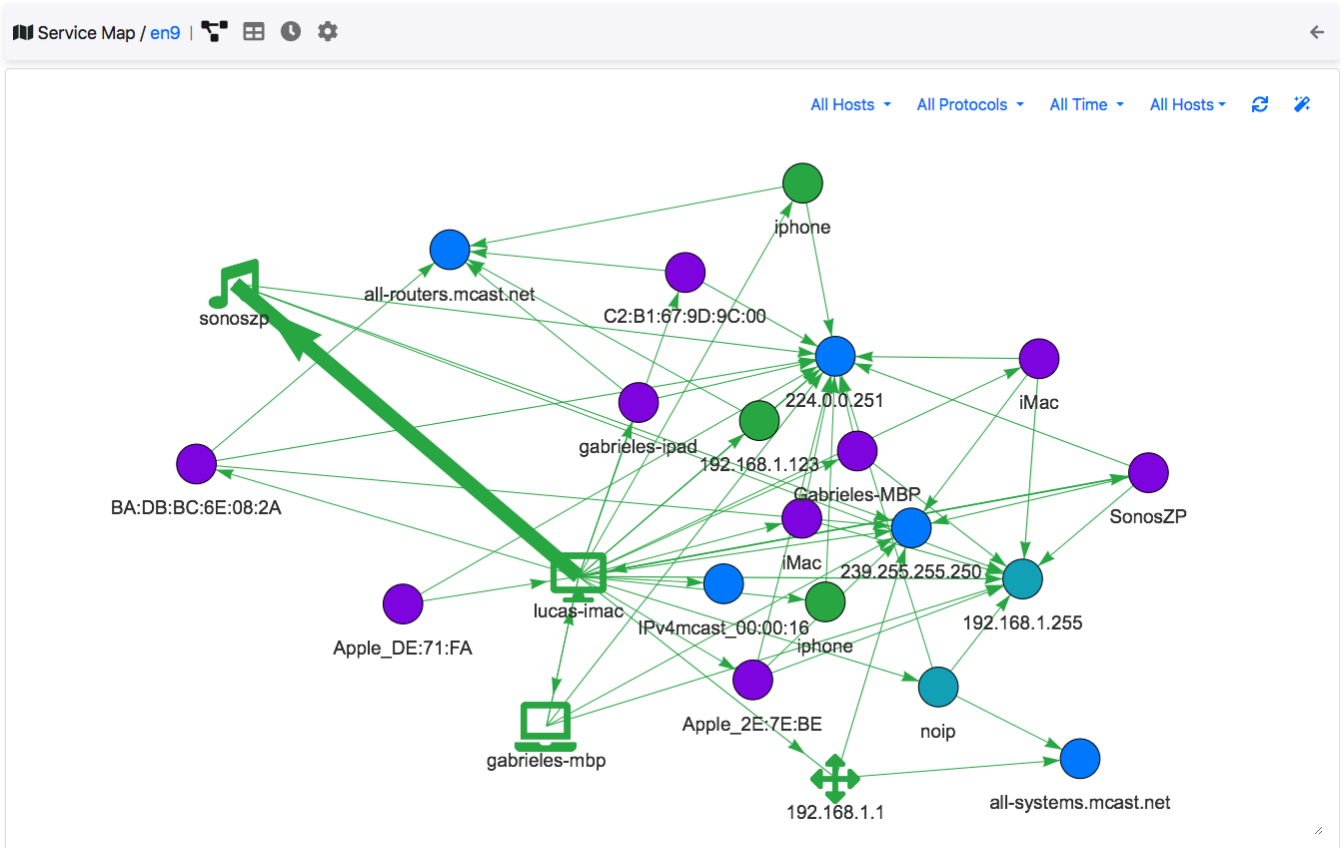
Score-based Behaviour Analysis [5/5]



Lateral Movement [1/4]

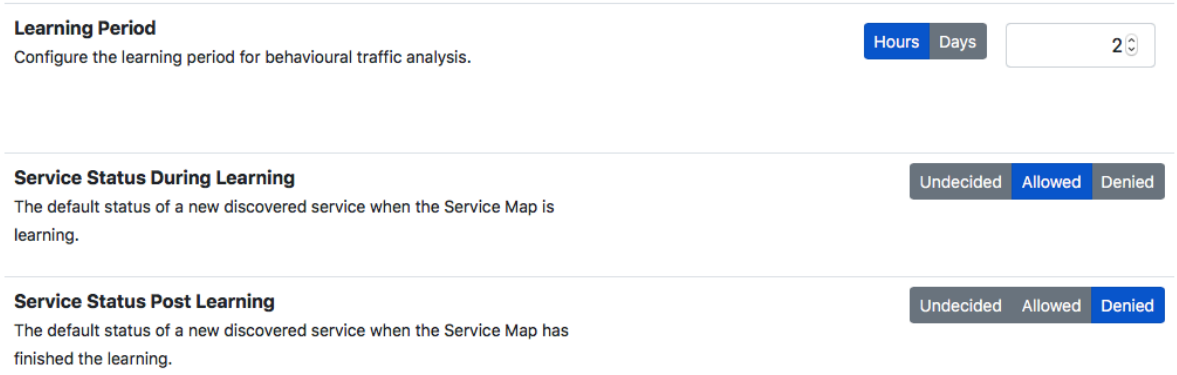
- What happens if a malware is roaming in our network? How can we spot it?
- In addition to the checks just presented, it can help to create a model of the network traffic and to continuously match it against live communications.
- Communications not matching the model are probably an indication of mistakes or new traffic patterns worth to be analysed.

Lateral Movement [2/4]



Lateral Movement [3/4]

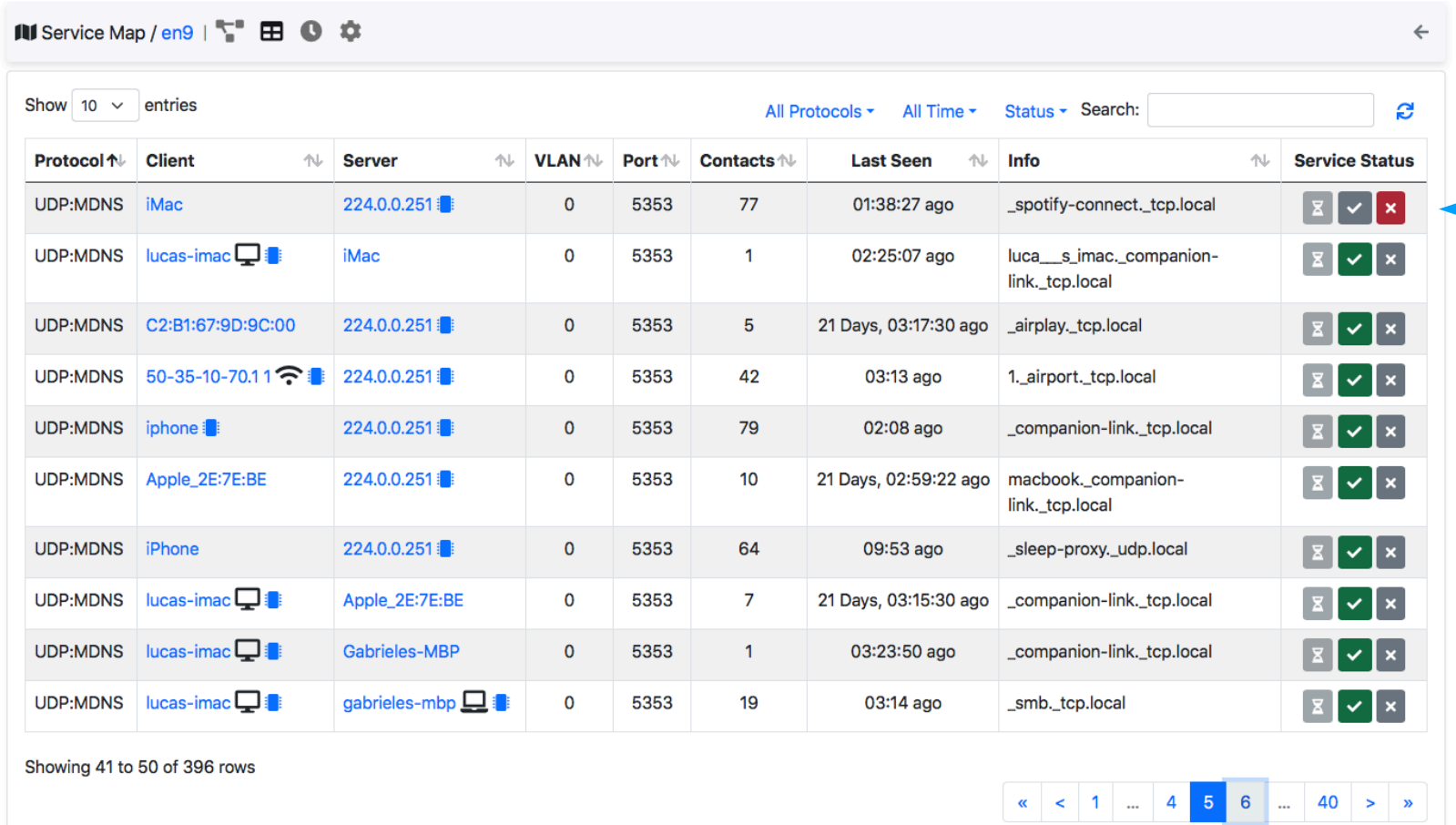
- Learning Period
  - Discover new services and assign a default policy to them.
  - No alert is generated during learning.



Post Learning

Alerts Enabled

Lateral Movement [4/4]



Forbidden

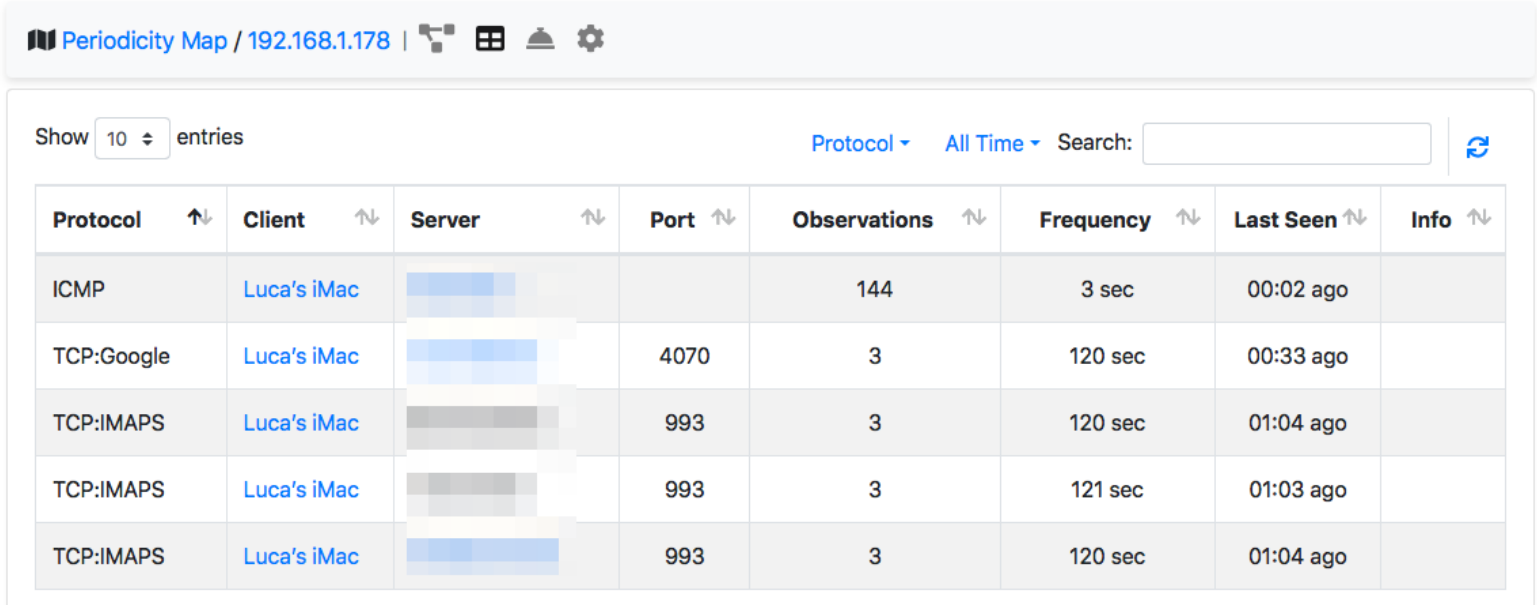
Beaconing Detection [1/3]

- Beacons are periodic low-volume communications that can be easily hidden inside the overall traffic.
- They are:
  - Often used by malware to talk back with the master.
  - An indication of failures (e.g. periodic connection to a service that is unavailable).
  - Used to identify monitoring activities (e.g. scans etc) or periodic checks (e.g. email download).
- In essence beaconing is not just for cybersecurity but also for spotting activities worth to be analysed.

Beaconing Detection [2/3]

- Instead of using AI or complex algorithms for beaconing detection we use a simple method:
  - Keep track of quadruplets <source/destination IP, destination port, layer 4 protocol>.
  - As soon as a new flow is detected a quadruplet is created (if not already present) or updated (if already created).
  - Idle quadruplets or quadruplets whose periodicity isn't too constant (of course we take into account time drifts) are discarded.

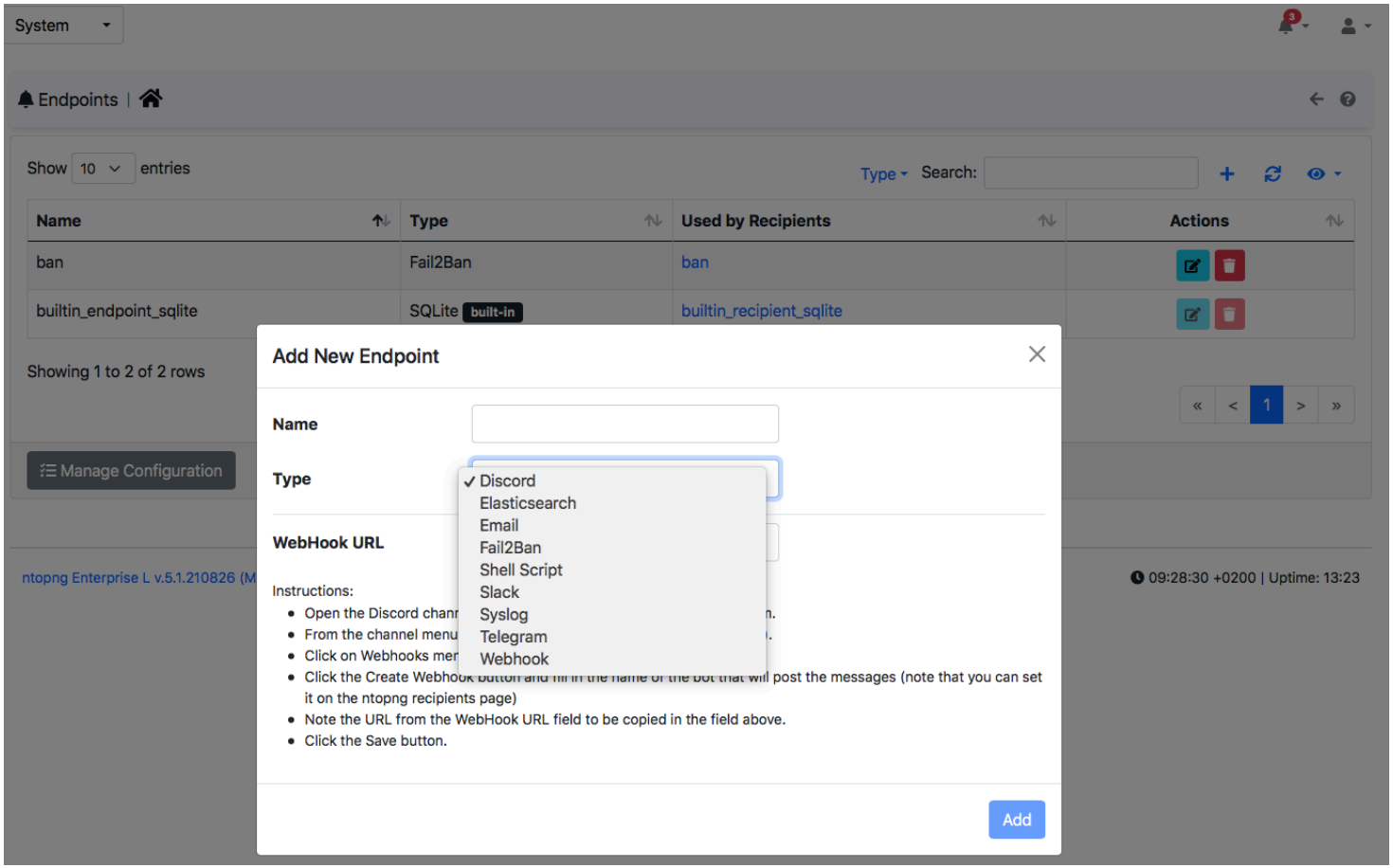
Beaconing Detection [3/3]



Protocol	Client	Server	Port	Observations	Frequency	Last Seen	Info
ICMP	Luca's iMac			144	3 sec	00:02 ago	
TCP:Google	Luca's iMac		4070	3	120 sec	00:33 ago	
TCP:IMAPS	Luca's iMac		993	3	120 sec	01:04 ago	
TCP:IMAPS	Luca's iMac		993	3	121 sec	01:03 ago	
TCP:IMAPS	Luca's iMac		993	3	120 sec	01:04 ago	

- Beaconing with Unknown or "unpleasant" (e.g. IRC) protocols are an indicator of suspicious communications.
- Beaconing begin/end is reported as informative alert.

Alerts: Actionable Items

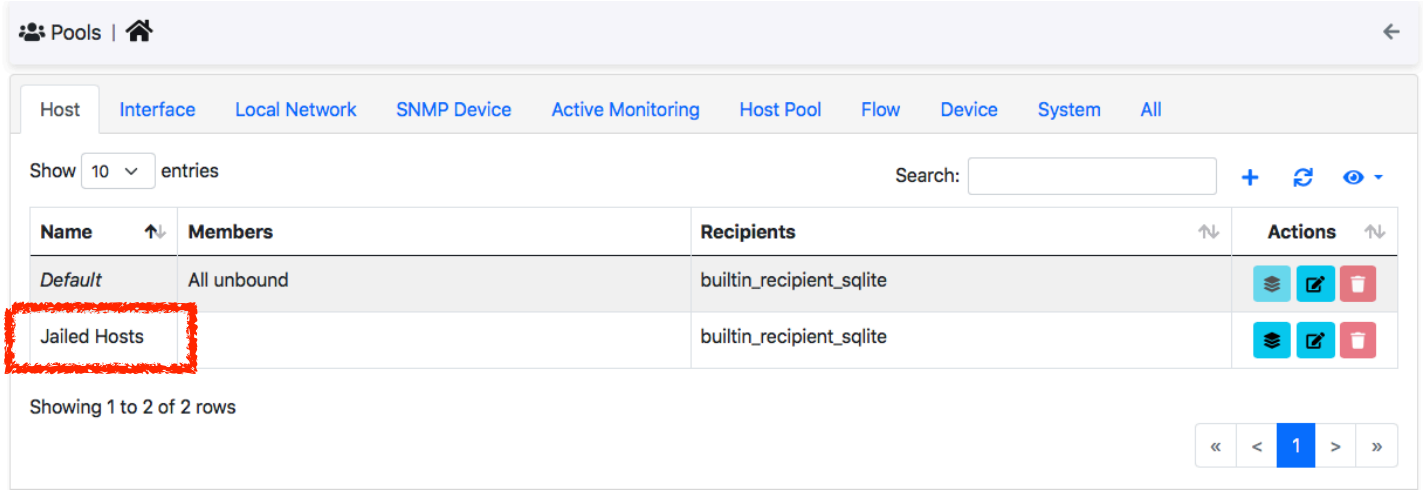


Using Score to Enforce Policies [1/7]

- A host pool is a logical group of hosts that for some reason (i.e. they do not need to belong to the same IP network or VLAN) can be grouped together.
- Pools can have alert actions defined: this allows hosts to perform different actions when an alert is triggered.
- Example:
  - Send a slack message to XYZ when there is alert for pool ABC.
  - Just log the alert for hosts other than XYZ.

Using Score to Enforce Policies [2/7]

- All pools are alike with the exception of the “Jailed Hosts” Pool.

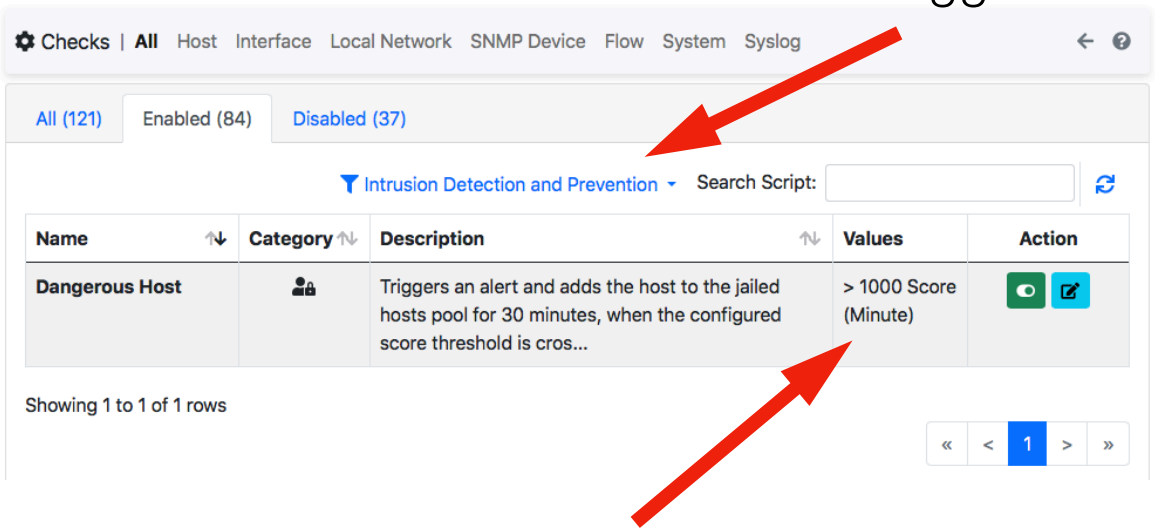


- Dangerous hosts are added/removed to/from this pool as they are detected or come back to normal.

Using Score to Enforce Policies [3/7]

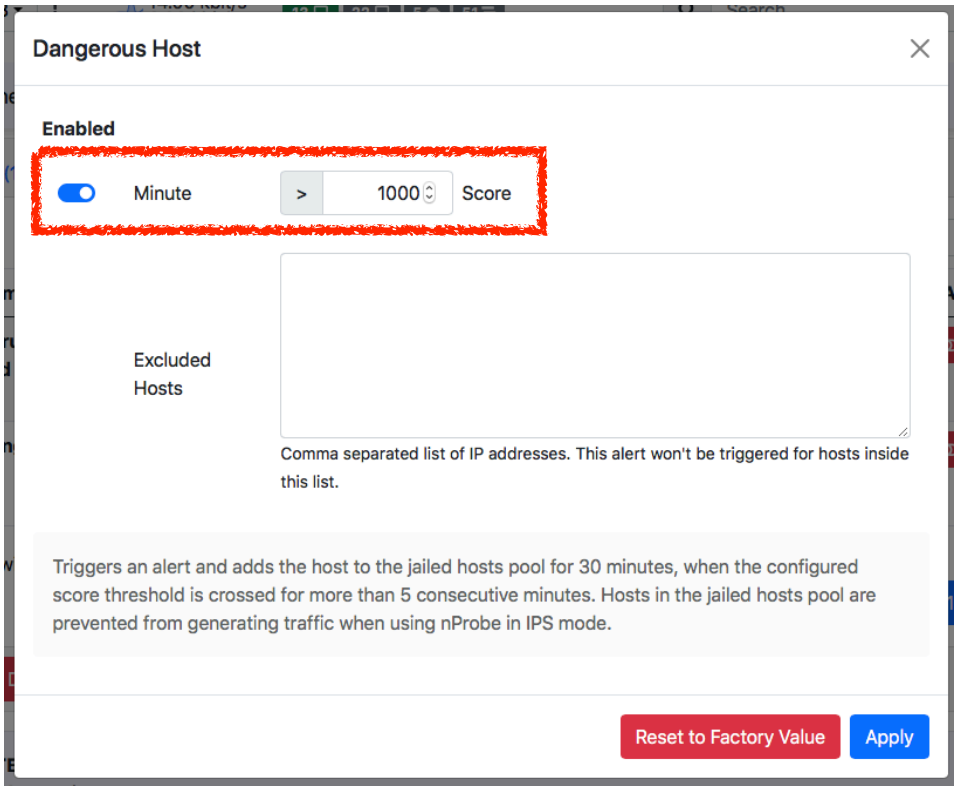
- It is possible to use the host score in order to prevent hosts from “infecting” the rest of the network.

Trigger Family



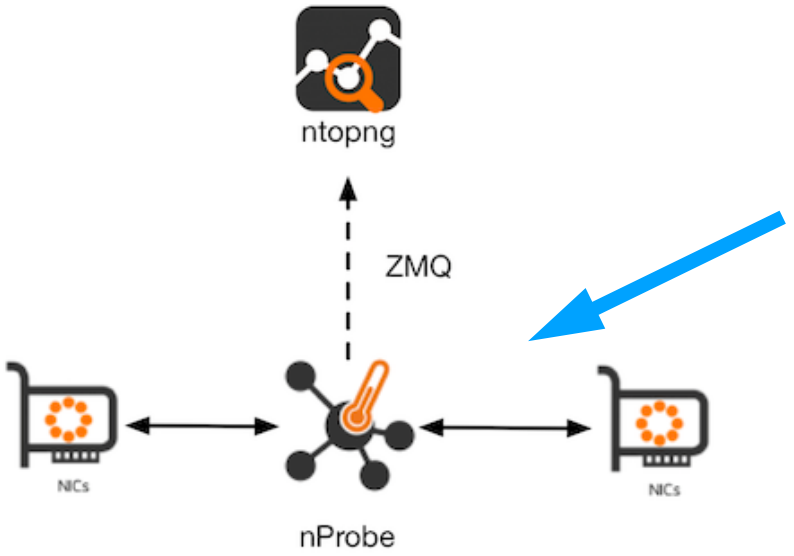
Stateful Alert (no permanent block)

Using Score to Enforce Policies [4/7]



Using Score to Enforce Policies [5/7]

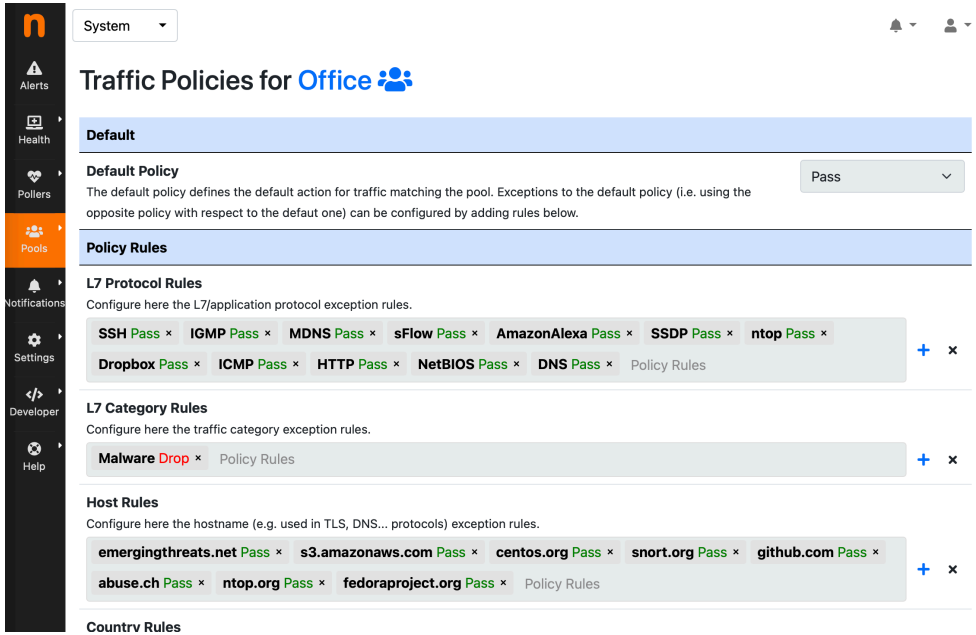
- It is possible to combine ntop tools to enforce policies using nProbe in IPS mode:



Typical deployment is close to the gateway (nord/sud traffic)

Using Score to Enforce Policies [6/7]

- With nProbe IPS, each host pool can have custom traffic policies configured by ntopng and enforced by nProbe IPS.

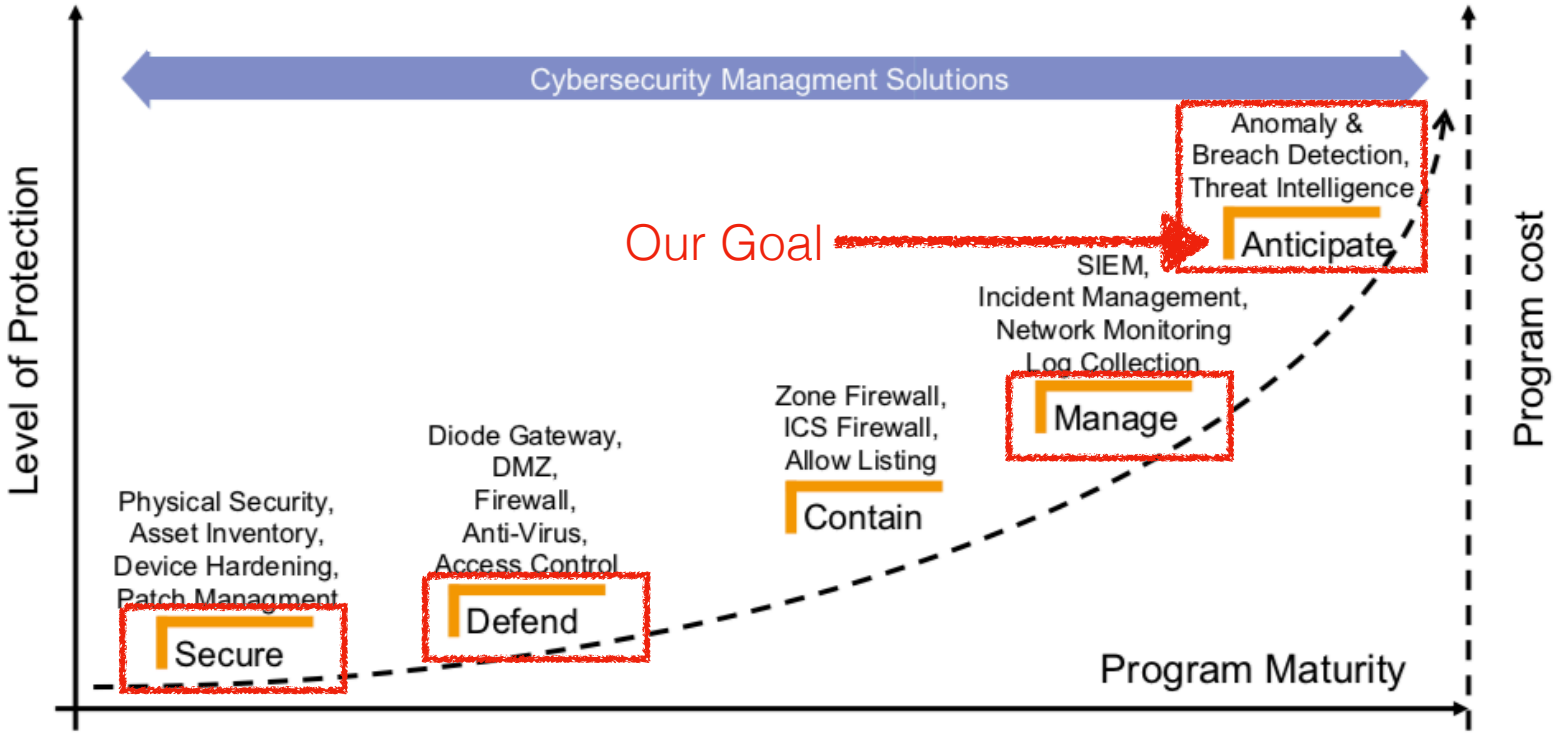


Using Score to Enforce Policies [7/7]

- The only exception to this policy is the Jailed Hosts pool:
  - Hosts added to this pool are blocked.
  - When a host is removed from this pool (after having been blocked), such host is moved back to the original pool (or the default pool).
- Whenever a policy is changed or a host is added/removed from this pool, ntopng informs all nProbes in IPS mode (yes, you can have more than one) automatically, with no user action whatsoever. All actions performed are logged, and "dry run" mode is available for simulating the actions before moving inline.

Part II:
Ongoing Developments

2021 Monitoring Goals



Picture courtesy of switch.ch

How Can we Anticipate a Problem?

- Monitoring can show you when a problem is happening or (better) what are suspicious flows that can be an indication of a future problem.
- Can we do anything better than this? What if I could detect the user and application that generated a traffic flow?
- Goal: extend current monitoring capabilities with system analysis in order to report richer information and build new, more powerful checks.

Cybersecurity and Networking

- In a way, cybersecurity would not be that important without the Internet as networks propagate threats.
- Using DPI and traffic analysis techniques so far presented it is possible to have a great level of visibility and protection but…
- East-west traffic monitoring is not so simple and available techniques (e.g. sFlow) are sampled.
- Threats do their best to hide themselves: volumetric attacks are “nice” as they can be easily spotted.
- More packets, more ML and more checks are the only viable solution to this problem ?

nProbe Agent

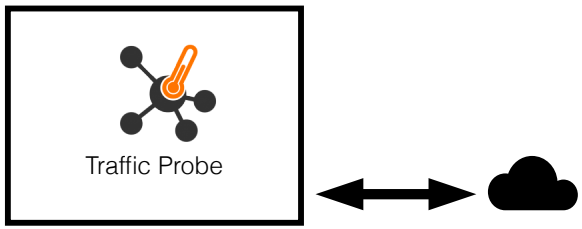
- In 2018 we have released a Linux-only event based (i.e. packet-less) agent named nProbe Agent.


```
28/Apr/2019 23:46:29 [Netlink.cpp:1159] [Netlink] [counters] { "timestamp":
"1556487989.626174", "ifName": "veth40297a6", "ifIndex": 21, "LOCAL_CONTAINER": { "DOCKER":
{ "NAME": "tecmint-web3" } }, "ifInOctets": 32477, "ifInPackets": 328, "ifInErrors": 0,
"ifInDrops": 0, "ifOutOctets": 13110951, "ifOutPackets": 40902, "EXPORTER_IPV4_ADDRESS":
"x.x.x.x" }

9/Apr/2019 12:09:54 [EBPF.cpp:178] [eBPF] { "timestamp": "1556532594.175074", "LOCAL_PROCESS":
{ "PID": 17932, "UID": 135, "GID": 145, "PROCESS_PATH": "/usr/bin/influxd" },
"LOCAL_FATHER_PROCESS": { "PID": 1, "UID": 0, "GID": 0, "PROCESS_PATH": "/lib/systemd/
systemd" }, "EVENT_TYPE": "ACCEPT", "IP_PROTOCOL_VERSION": 4, "PROTOCOL": 6, "L4_LOCAL_PORT":
51176, "L4_REMOTE_PORT": 8086, "IPV4_LOCAL_ADDR": "127.0.0.1", "IPV4_REMOTE_ADDR": "127.0.0.1",
"EXPORTER_IPV4_ADDRESS": “x.x.x.x” }
```


- The idea was to merge network with system visibility. It turned out to confuse people using nProbe. For this reason we have decided to revamp this idea:
  - Merging this code with nProbe and discontinue nProbe Agent.
  - Adding Windows support

Merging Network and System IEs [1/4]



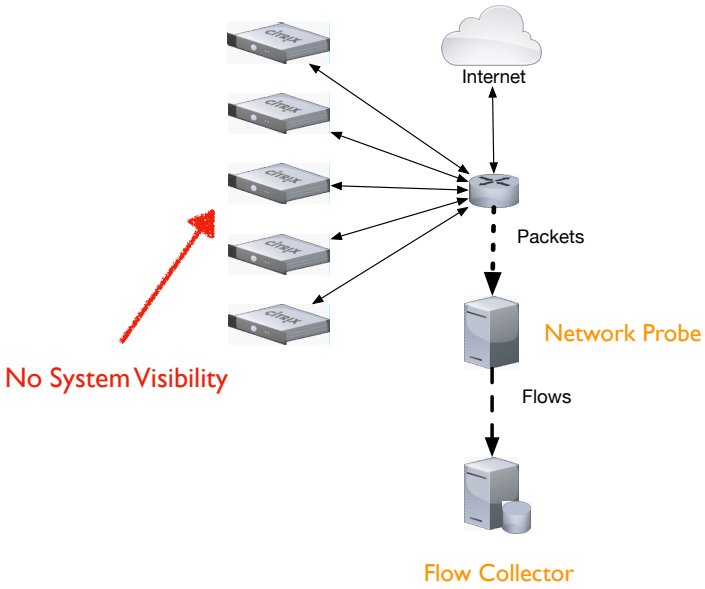
```
[57640][Len 4]  %SRC_PROC_PID                Flow source process PID
[57641][Len 16] %SRC_PROC_NAME               Flow source process name
[57897][Len 4]  %SRC_PROC_UID                Flow source process userId
[57844][Len 16] %SRC_PROC_USER_NAME          Flow source process user name
[57845][Len 4]  %SRC_FATHER_PROC_PID         Flow source father process PID
[57846][Len 16] %SRC_FATHER_PROC_NAME        Flow source father process name
[57847][Len 4]  %DST_PROC_PID                Flow dest process PID
[57848][Len 16] %DST_PROC_NAME               Flow dest process name
[57898][Len 4]  %DST_PROC_UID                Flow dest process userId
[57849][Len 16] %DST_PROC_USER_NAME          Flow dest process user name
[57850][Len 4]  %DST_FATHER_PROC_PID         Flow dest father process PID
[57851][Len 16] %DST_FATHER_PROC_NAME        Flow dest father process name
```

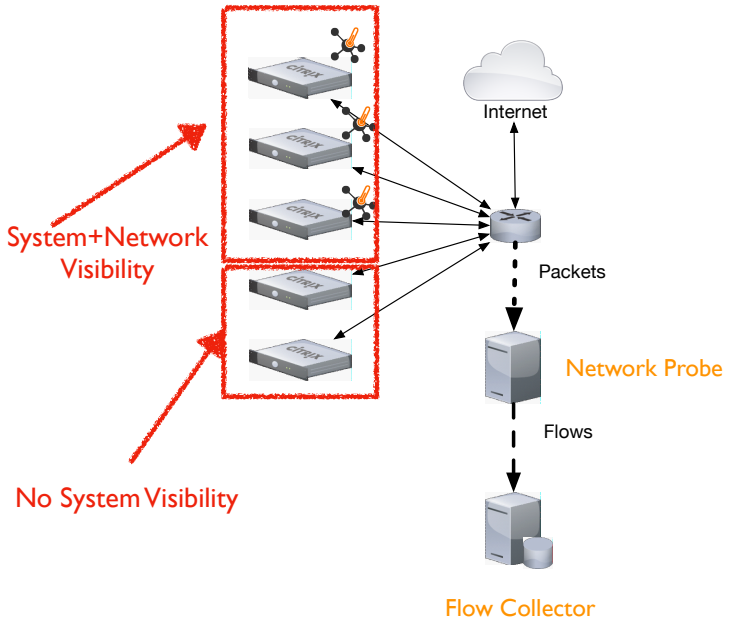
Merging Network and System IEs [2/4]

Flow: 192.168.1.187:50837 ⇄ 104.244.42.136:443 | **Overview**

Flow Peers [Client / Server]	192.168.1.187 R :50837 ⇄ 104.244.42.136 R :443 [Twitter Inc.]	
Protocol / Application	TCP / TLS.Twitter (SocialNetwork)	
First / Last Seen	22/10/2021 16:16:06 [00:42 sec ago]	22/10/2021 16:16:06 [00:42 sec ago]
Total Traffic	Total: 93 Bytes	
	Client → Server: 1 Pkts / 41 Bytes	Client ← Server: 1 Pkts / 52 Bytes
	192.168.1.187:50837	104.244.42.136:443
DSCP / ECN [Client / Server]	Best Effort [CS0] / Disabled (0)	Best Effort [CS0] / Disabled (0)
CommunityId	1:/Z0VhpynbI/B2TsC/c47Ei9ZAlg=	
Actual / Peak Throughput	0 bit/s / 0 bit/s	
ASN [Client / Server]		13414
Flow Verdict	0	
Additional Flow Elements		
IPv4 address of the host were nProbe runs	192.168.1.187	
Client process name	C:\Program Files (x86)\Google\Chrome\Application\chrome.exe	
Total number of exported flows	373	

Merging Network and System IEs [3/4]





Merging Network and System IEs [4/4]

- Advantages
  - Map traffic to processes/users: finally we know “who is doing what”.
  - Detect unexpected processes making traffic.
  - Simplified troubleshooting and incident analysis with contextual data.
- Limitations
  - Still a passive tool: the collector has the knowledge.
  - It is unable to detect “changes” but only “facts” (i.e. annotated flows with limited system metadata).

Towards a nProbe-based EDR

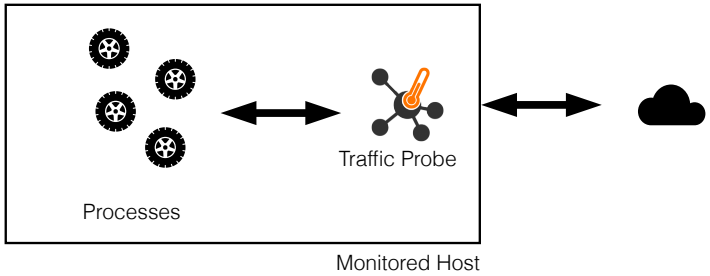
- What if nProbe could:
  - Detect changes in configuration invisible to the network.
  - Use process and user information to properly evaluate risks in communications.
  - Use contextual information (e.g. process) not just for enriching flow data but also for preventing threats from spreading in the network?
- What about a nProbe-based EDR (Endpoint Detection and Response) ?

Cybersecurity Simplified [1/2]

- Challenge: can we allow administrators to block threats before the problem shows up?
- Options: block traffic of applications that
  - Are not installed as package or that are started from non-standard locations (e.g. /tmp).
  - Have not been running previously.
  - Communicate with blacklisted IPs.
  - Have a periodicity and are not monitoring tools.
  - …(cont).

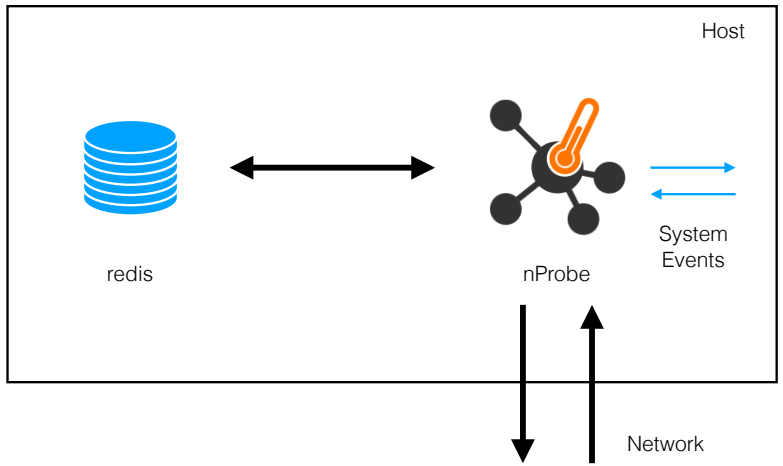
Cybersecurity Simplified [2/2]

- Combining system visibility with network monitoring, enabled us to create an active probe able to block specific application traffic and that can very well fit with the zero-trust principle that is becoming increasingly popular.



Introducing System Visibility in nProbe [1/5]

- nProbe:
  - Sits on top of the network stack (including containers) in order to receive traffic and inspect/block it.
  - Listen to system events in order to bind local traffic to processes and users.



Introducing System Visibility in nProbe [2/5]

- nProbe uses redis as local policy cache for storing learnt information and as inter-process communication in case of high traffic rates that need to be handled by multiple nProbe processes.
- During the learning period, nProbe stores on redis observed <user>:<process> associations.
- Past learning, redis is used to retrieve known policies to be used for enforcement.

Introducing System Visibility in nProbe [3/5]

- It is possible to query redis for users who sent data out, and for each process (that transmitted/received data) run by each user.

```
$ redis-cli keys "process.*"
 1) "process.root"
 2) "process.www-data"
 3) "process.influxdb"
 4) "process._apt"
 5) "process.postgres"
 6) "process.avahi"
 7) "process.clickhouse"
 8) "process.chronograf"
 9) "process.deri"
10) "process.grafana"
```

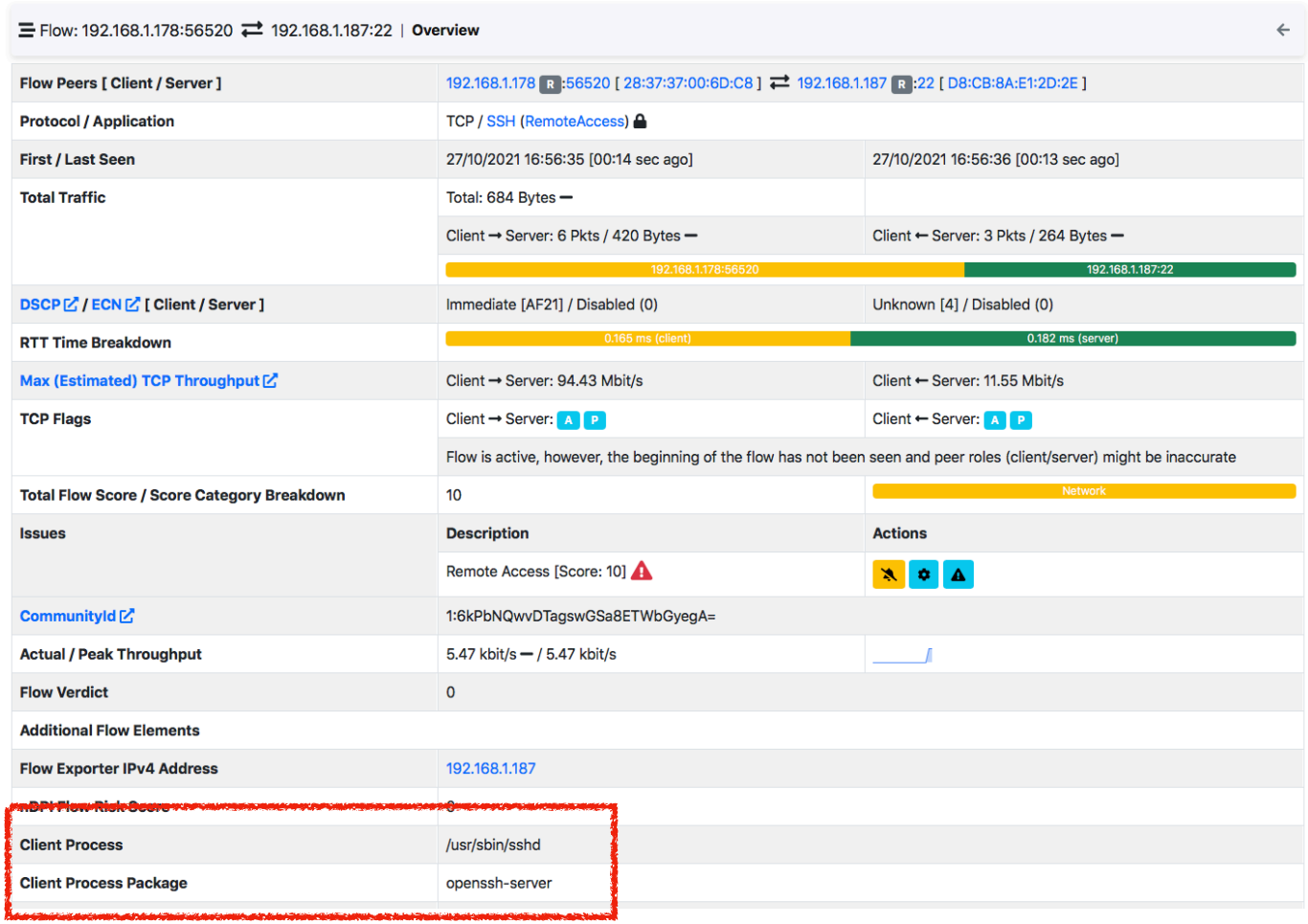
```
$ redis-cli hkeys "process.root"
1) "/usr/sbin/NetworkManager"
2) "/usr/lib/sm.bin/sendmail"
3) "/usr/sbin/ntpdate"
4) "/sbin/dhclient"
5) "/usr/sbin/cups-browsed"
6) "/snap/core/11606/usr/lib/snapd/snapd"
7) "/home/deri/nprobe"
8) "sendmail-mta"
```

- Is an unknown process allowed to do networking ? Probably not.

Introducing System Visibility in nProbe [4/5]

- Unless you are developing software, applications need to be installed with packages.
- Malware applications are (usually) not packaged, so this can be a good indicator of compromise.
- Currently we support Linux packaging: both .deb and .rpm families are supported.
- Windows is not yet supported. We believe that osquery.io might be an option to consider in the future.

Introducing System Visibility in nProbe [5/5]



Further Visibility: Server Side [1/3]

- As said before, a good strategy for detecting issues/ reconfigurations/malware is to track changes.
- When a malware speaks with remote peers, nProbe can detect the flow and report contextual information (process and package name).
- What if the malware isn't making any traffic (so it's in essence invisible to flows) but it's ready to accept connections from applications? Or if the traffic is so little that hides itself in background noise?

Further Visibility: Server Side [2/3]

- nProbe has been enhanced with local host port monitoring for:
  - Binding a port with an application and a package.
  - Detecting changes in port allocation: a new port is open, an existing port is closed, or a different process is listening to an existing open port.
  - Reporting this information to flow collectors for increased visibility.
- This feature is implemented on both Windows and Linux nProbe versions.

Further Visibility: Server Side [3/3]


```
{
    "ip-addresses": ["10.3.240.28", "192.168.1.187"],
    "listening-ports": {
        "tcp4": [{
            "port": 22,
            "proc": "/usr/sbin/sshd",
            "pkg": "openssh-server"
        }, {
            "port": 53,
            "proc": "/usr/sbin/dnsmasq",
            "pkg": "dnsmasq-base"
        }, {
            "port": 1234,
            "proc": "/home/deri/nProbe/nprobe",
            "pkg": ""
        }],
        "tcp6": [{
            "port": 9000,
            "proc": "/usr/bin/clickhouse",
            "pkg": “clickhouse-common-static"
...
```


No Package !

Exporting System Information

- Process information can be combined with DPI and flow risks to determine the flow “verdict”.

```
# cat /tmp/2021/09/22/22/49.flows
IPV4_SRC_ADDR|IPV4_DST_ADDR|INPUT_SNMP|OUTPUT_SNMP|IN_PKTS|IN_BYTES|FIRST_SWITCHED|LAST_SWITCHED|L4_SRC_PORT|
L4_DST_PORT|TCP_FLAGS|PROTOCOL|SRC_PROC_NAME|SRC_PROC_PID|DST_PROC_NAME|DST_PROC_PID|FLOW_VERDICT
192.168.1.187|192.168.1.178|0|0|17|6564|1632343764|1632343765|22|56218|24|6||0|/usr/sbin/sshd|2910|0
192.168.1.178|192.168.1.187|0|0|17|884|1632343764|1632343765|56218|22|16|6|/usr/sbin/sshd|2910||0|0
192.168.1.178|192.168.1.187|0|0|9|612|1632343767|1632343768|49372|22|24|6|/usr/sbin/sshd|2910||0|0
192.168.1.187|192.168.1.178|0|0|5|504|1632343767|1632343768|22|49372|24|6||0|/usr/sbin/sshd|2910|0
192.168.1.187|192.168.1.178|0|0|11|3648|1632343767|1632343768|22|56218|24|6||0|/usr/sbin/sshd|2910|0
192.168.1.178|192.168.1.187|0|0|11|572|1632343767|1632343768|56218|22|16|6|/usr/sbin/sshd|2910||0|0
192.168.1.187|192.168.1.1|0|0|2|116|1632343768|1632343768|44199|53|0|17||0|/usr/bin/traceroute.db|4909|2
192.168.1.1|192.168.1.187|0|0|1|106|1632343768|1632343768|53|44199|0|17|/usr/bin/traceroute.db|4909||0|2
192.168.1.187|192.168.1.178|0|0|9|3264|1632343771|1632343771|22|56218|24|6||0|/usr/sbin/sshd|2910|0
192.168.1.178|192.168.1.187|0|0|9|468|1632343771|1632343771|56218|22|16|6|/usr/sbin/sshd|2910||0|0
192.168.1.178|192.168.1.187|0|0|4|244|1632343772|1632343772|49372|22|24|6|/usr/sbin/sshd|2910||0|0
192.168.1.187|192.168.1.178|0|0|3|296|1632343772|1632343772|22|49372|24|6||0|/usr/sbin/sshd|2910|0
```

Drop

- **2** means drop as traceroute was either unknown during learning phase, or not part of an installed package (this culprit can be solved if SRC_PROC_PACKAGE_NAME is also exported).

Enforcement vs Monitoring [1/2]

- nProbe can both enforce traffic policies (i.e. pass/drop) or passively monitor traffic.
- The difference is just on how the tool is started:
  - Monitoring
    - Capture traffic from an interface.
  - Enforcement
    - nProbe is started on top of netfilter (Linux firewall architecture) for blocking traffic if necessary.

Enforcement vs Monitoring [2/2]

- Passive Monitoring

```
nprobe -i enp5s0 -T "%IPV4_SRC_ADDR %IPV4_DST_ADDR %INPUT_SNMP %OUTPUT_SNMP
%IN_PKTS %IN_BYTES %FIRST_SWITCHED %LAST_SWITCHED %L4_SRC_PORT %L4_DST_PORT
%TCP_FLAGS %PROTOCOL %SRC_PROC_NAME  %DST_PROC_NAME %FLOW_VERDICT" --redis
localhost --process-learning-duration 86400:0
```

- Enforcement

```
nprobe -i nf:0 -T "%IPV4_SRC_ADDR %IPV4_DST_ADDR %INPUT_SNMP %OUTPUT_SNMP %IN_PKTS
%IN_BYTES %FIRST_SWITCHED %LAST_SWITCHED %L4_SRC_PORT %L4_DST_PORT %TCP_FLAGS
%PROTOCOL %SRC_PROC_NAME  %DST_PROC_NAME %FLOW_VERDICT" --redis localhost --
process-learning-duration 86400:0
```

- This nProbe pre-release is currently available for Ubuntu 18.04 and 20.04.
- Windows version of nProbe is (so far) monitoring only.
- Final release is expected in December/January timeframe.
- Note: all items discussed so far are container friendly.

Open Discussion

- https://blog.ntop.org
- https://github.com/ntop/
- https://www.ntop.org/community/