From Finger-Defined Networks to Streaming Telemetry – Getting Visibility on Your Network

7th SIG-NOC Meeting - Barcelona

Carles Batalla
Solutions Architect (Cisco)

April 2018

Why

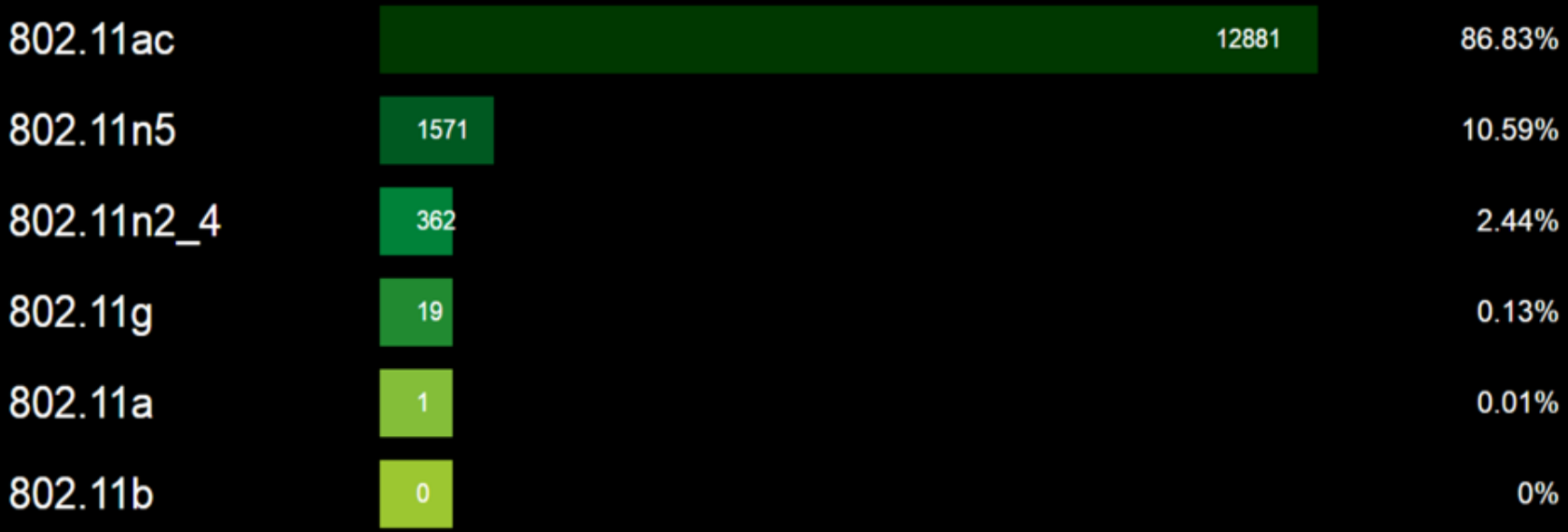
If a ROUTER falls in the NETWORK , and nobody is around to MONITOR it, does it make an impact ?

Wireless Client Distribution

	802.11.ac	802.11n5	802.11n2_4	802.11g	802.11a	802.11b
#clus	12838	1436	0	0	1	N.A.
#clus_2.4GHz	0	0	362	19	0	N.A.
[NOC personnel]	34	1	0	0	0	N.A.
WirelessSAC	7	0	0	0	0	N.A.

Wireless Radio Distribution

Total Wireless Clients: 14834



How Then

How do you ‘monitor’ your network?

If this is a picture of your NOC personnel…

You may have a Finger-Defined Network.

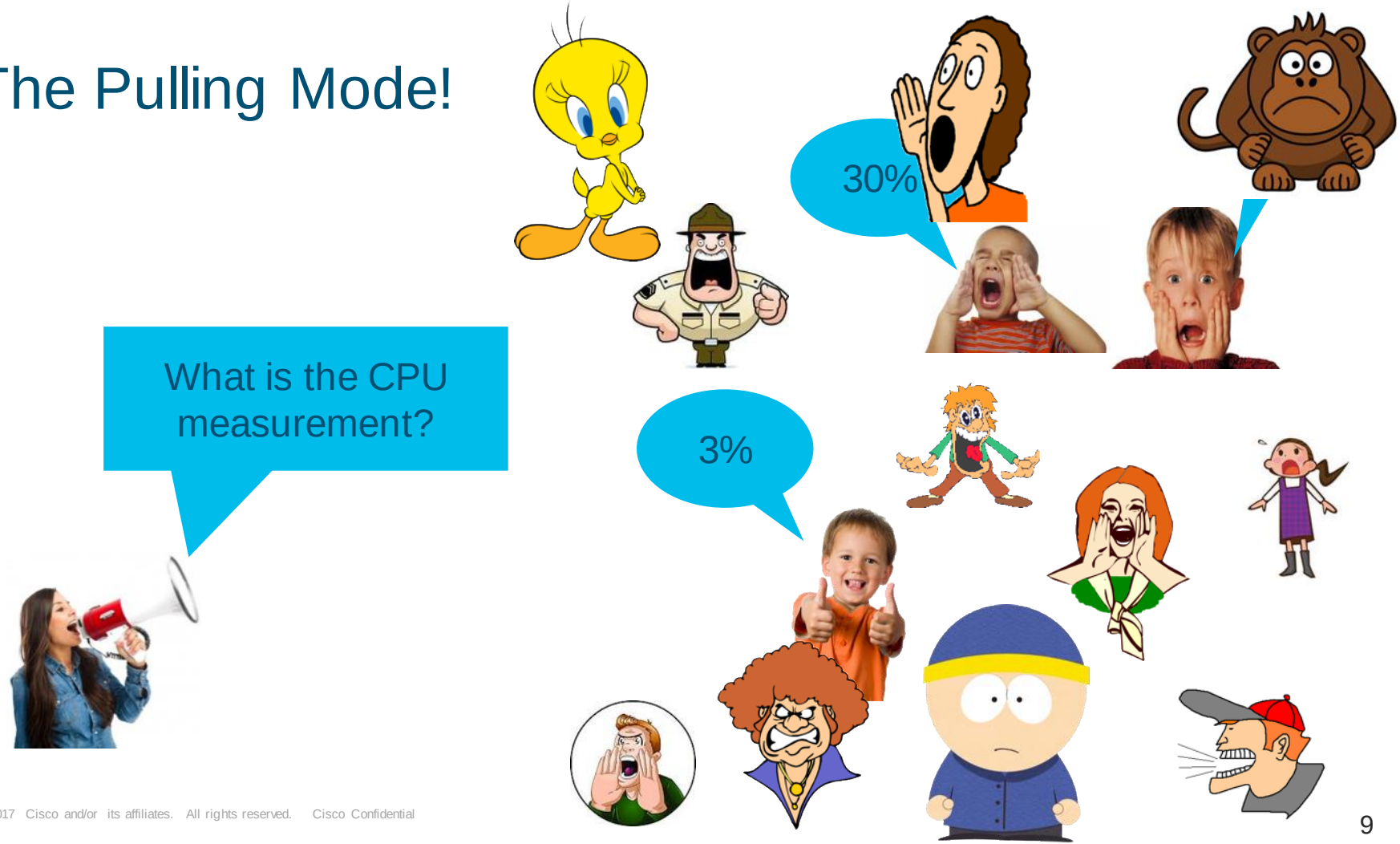
Why Change?

- Familiar Manual, CLI-driven, device-by-device approach is inefficient
- Increased need for programmatic interfaces which allow faster and automated execution of processes and workflows with reduced errors
- Need for a 'central source of truth' and touch-point

```
davisnet-3560CG>sh int
Vlan1 is up, line protocol is up
  Hardware is EtherSVI, address is 381c.1a72.1bc0 (bia 381c.1a72.1bc0)
  Internet address is 192.168.1.10/24
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive not supported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 1/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 4000 bits/sec, 2 packets/sec
  5 minute output rate 1000 bits/sec, 1 packets/sec
     569740 packets input, 41609077 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     16520 packets output, 1406421 bytes, 0 underruns
     0 output errors, 2 interface resets
     0 output buffer failures, 0 output buffers swapped out
GigabitEthernet0/1 is up, line protocol is up (connected)
  Hardware is Gigabit Ethernet, address is 381c.1a72.1b81 (bia 381c.1a72.1b81)
  Description: downlink to Papa8ear2
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="4" xmlns="urn:ietf:params:netconf:base:1.0">
    <data>
        <xml-config-data> Building configuration...
            <Device-Configuration>
                <interface>
                    <Param>GigabitEthernet0/0</Param>
                    <ConfigIf-Configuration>
                        <ip>
                            <address><dhcp/></address>
                        </ip>
                        <duplex><auto/></duplex>
                        <speed><auto/></speed>
                    </ConfigIf-Configuration>
                </interface>
                <end></end>
            </Device-Configuration>
        </xml-config-data>
    </data>
</rpc-reply>]]>]]>
```

The Pulling Mode!



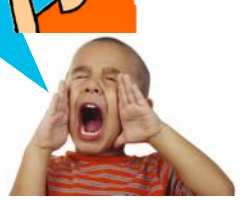
What is your IPv6 traffic?

What is the last fault generated?

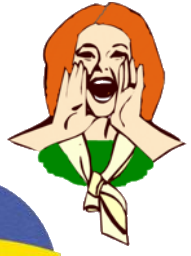
What is the CPU measurement?

Can I get CPU also?

30%

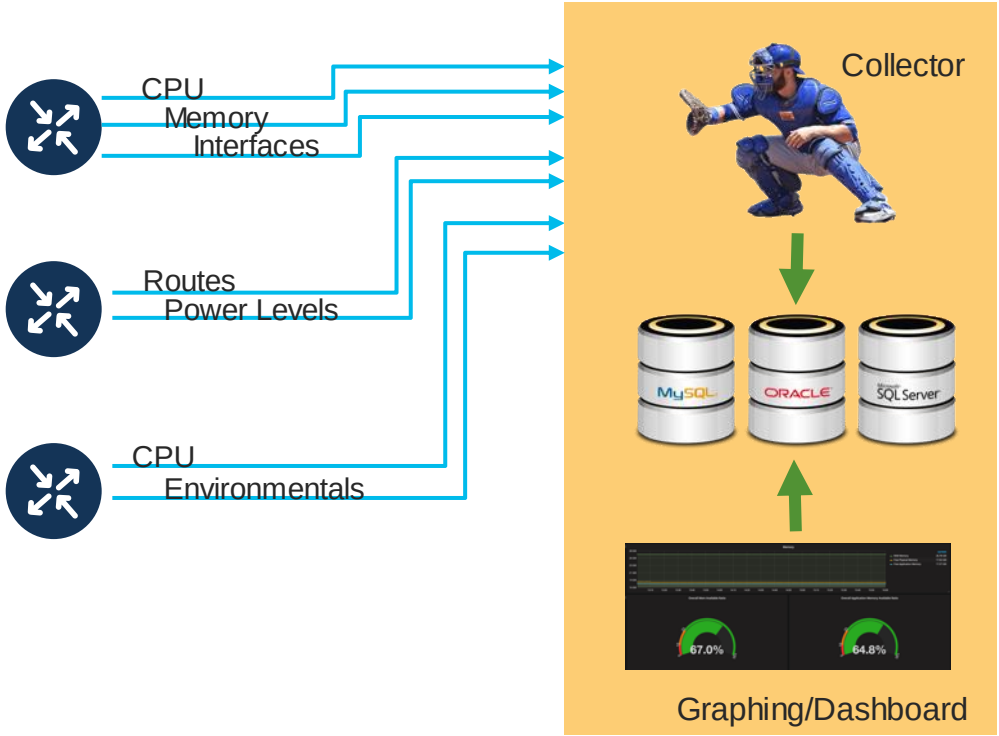


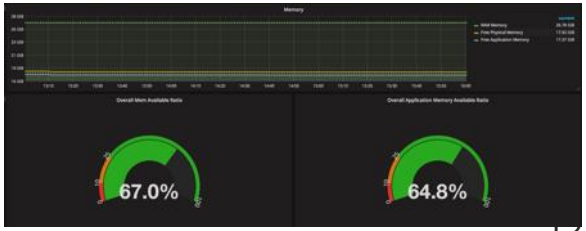
3%



How Now

Push Model

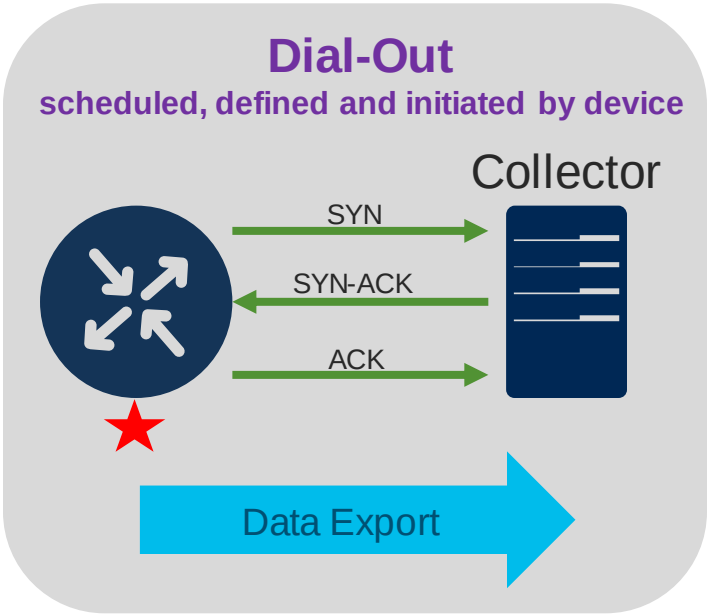




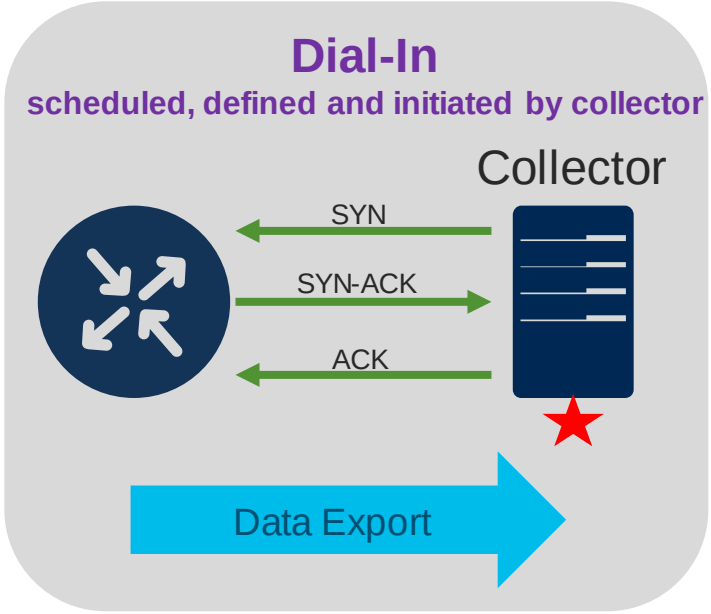
Steps to Configuring Model-driven Streaming Telemetry

1) Define destination(s)/target(s)
2) Define Sensor Groups and sensor paths
3) Define Subscriptions (sensor groups sent to destinations with a frequency)

Transport Options



gRPC & TCP



gRPC

Transport – Google Remote Procedure Call (gRPC)

- https://grpc.io/
- A modern open source high performance RPC framework
- Efficiently connect services with pluggable support for load balancing, tracing, health checking and authentication
- Highly efficient on wire and with a simple service definition framework
- Bi-directional streaming with http/2 based transport – also providing TLS support

Encoding – Google Protocol Buffers (GPB)

From https://developers.google.com/protocol-buffers/

"Protocol buffers are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler."

```
{ "node-name: "0/RP0/CPU0",
   "process-cpu": {
     { "total-cpu-fifteen-minute":  5,
       "total-cpu-five-minute":  6,
       "total-cpu-one-minute":  12
     }
   }
}
```

"Self-describing"
More data to transfer

```
1: 0/RP0/CPU0
10: 5
11: 6
12: 12
```

"compact"
Faster to transfer – less data
More complex to correlate

1) Define destinations(s)/target(s)

Dial-Out Example IOS-XR

```
telemetry model-driven
  destination-group Destination01
    vrf Management
    address-family ipv4 1.2.3.4 port 5432
      encoding self-describing-gpb
      protocol tcp
```

User-defined destination name

vrf path, if needed

Collector target IPv4/IPv6 address and port

Type of encoding
- self-describing-gpb
- gpb

Type of protocol
- grpc
- tcp

1) Define destinations(s)/target(s)

Dial-In Example IOS-XR

```
grpc
      port 57500
```

Dial-In does not have a user-defined destination, per se, but it does require enabling the gRPC server feature (global)

Dial-In is the least config-impacting

Dial-Out requires robust config management tools and processes

2) Define Sensor-Group

Dial-Out Example IOS-XR

```
telemetry model-driven
 sensor-group SG001-CPU
  sensor-path Cisco-IOS-XR-wdsysmon-fd-oper:system-monitoring/cpu-utilization
 !
 sensor-group SG002-Memory
  sensor-path Cisco-IOS-XR-nto-misc-oper:memory-summary/nodes/node/summary
 !
 sensor-group SG003-Ints-10G
  sensor-path Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface[interface-name='TenGigE0/2/0/*']/latest/generic-counters
```

YANG Model

subtree path

YANG – Yet Another Next Generation

- Defined 2010 in RFC 6020 by Tail-f's Martin Bjorklund (now at Cisco) to provide a modeling language for NETCONF – expanded beyond
- Human readable, easy to learn representation – compact C and Java-like syntax
- Hierarchical models with reusable types and groupings
- Supports definition of operations (RPCs)
- Constraints and configuration validation
- Well-defined version rules

self-contained top-level hierarchy of nodes

import or define data types

Containers group related nodes

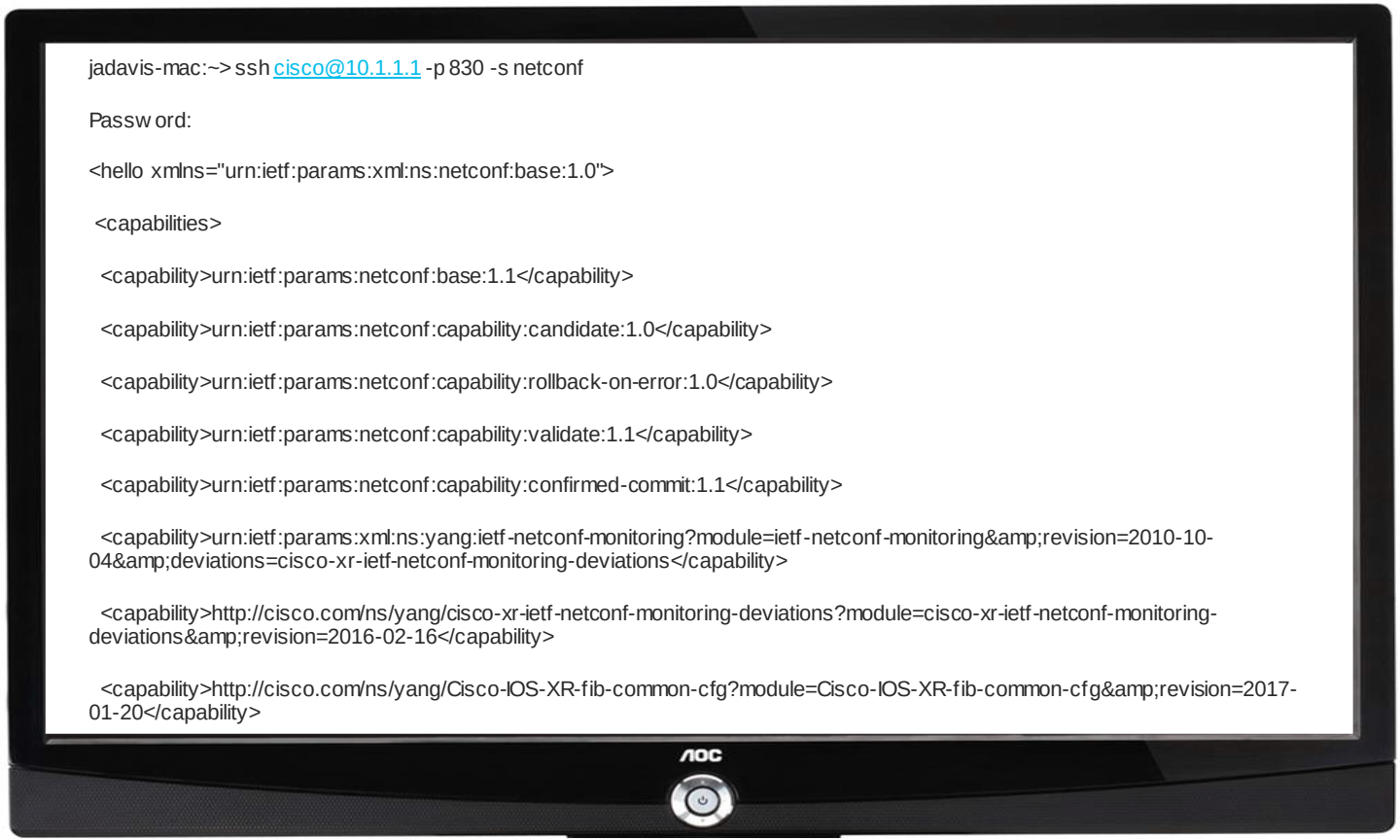
Lists for sequence of entries

Leaf nodes for simple data

```
module acme-system {
    namespace "http://acme.example.com/system";
    prefix "acme";

    organization "ACME Inc.";
    contact "joe@acme.example.com";
    description
        "The module for entities implementing the ACME system.";

    revision 2007-11-05 {
        description "Initial revision.";
    }

    container system {
        leaf host-name {
            type string;
            description "Hostname for this system";
        }

        leaf-list domain-search {
            type string;
            description "List of domain names to search";
        }

        list interface {
            key "name";
            description "List of interfaces in the system";
            leaf name {
                type string;
            }
            leaf type {
                type string;
            }
            leaf mtu {
                type int32;
            }
        }
    }
}
```

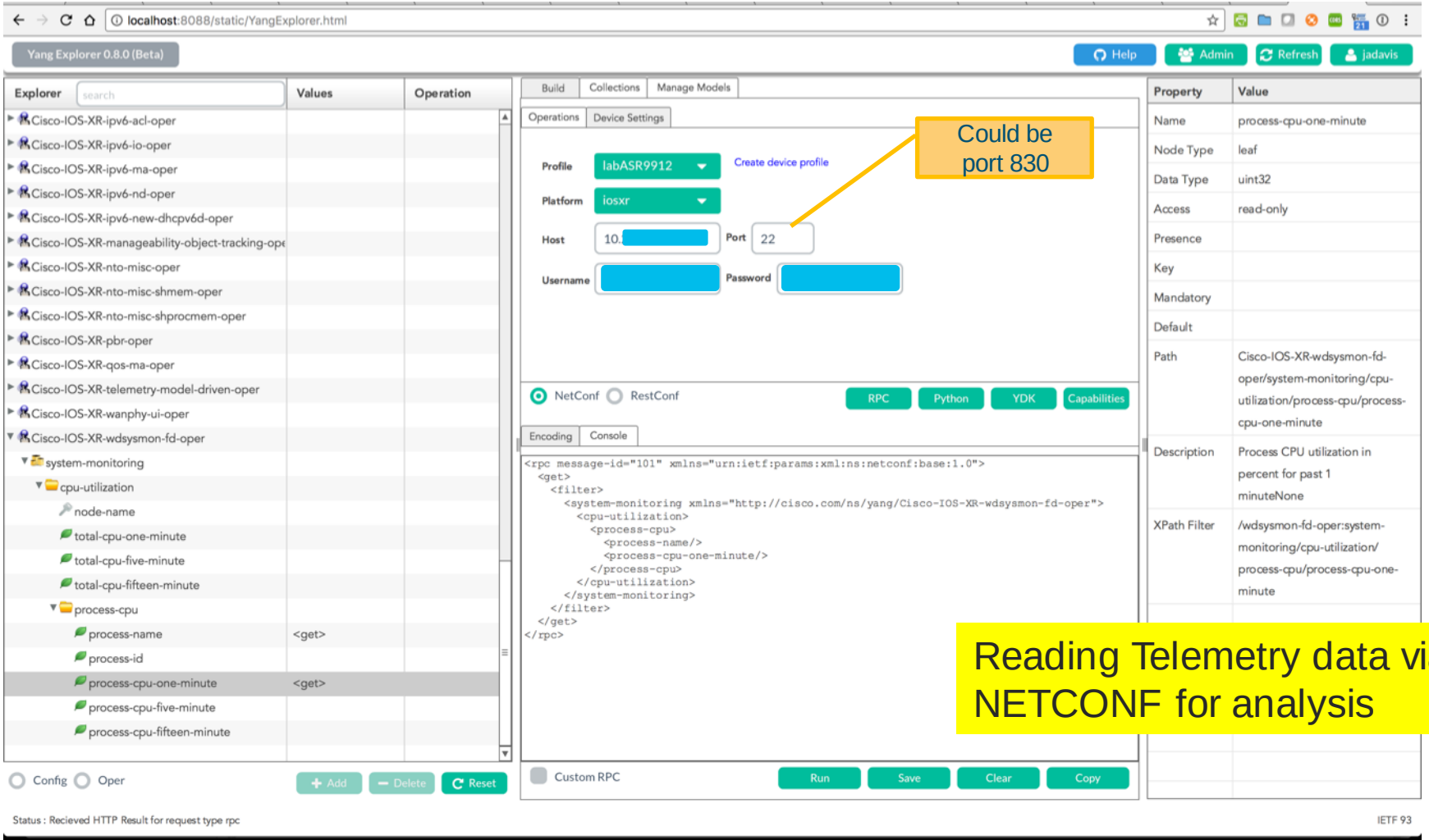
How to Tell What Your Device & SW Supports

- Ensure the device has NETCONF enabled
- SSH to the NETCONF port and subsystem as a local user [eg. admin or cisco] with proper password
- The system will push back a capabilities exchange listing the supported YANG models


jadavis-mac:~> ssh cisco@10.1.1.1 -p 830 -s netconf
Password:
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<capabilities>
<capability>urn:ietf:params:netconf:base:1.1</capability>
<capability>urn:ietf:params:netconf:capability:candidate:1.0</capability>
<capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</capability>
<capability>urn:ietf:params:netconf:capability:validate:1.1</capability>
<capability>urn:ietf:params:netconf:capability:confirmed-commit:1.1</capability>
<capability>urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring?module=ietf-netconf-monitoring&revision=2010-10-04&deviations=cisco-xr-ietf-netconf-monitoring-deviations</capability>
<capability>http://cisco.com/ns/yang/cisco-xr-ietf-netconf-monitoring-deviations?module=cisco-xr-ietf-netconf-monitoring-deviations&revision=2016-02-16</capability>
<capability>http://cisco.com/ns/yang/Cisco-IOS-XR-fib-common-cfg?module=Cisco-IOS-XR-fib-common-cfg&revision=2017-01-20</capability>
AOC

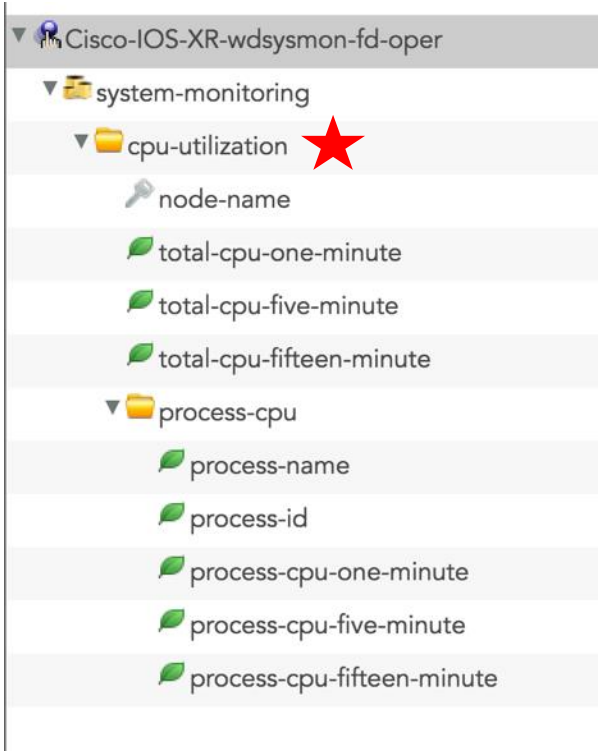
Options for Reading/Analyzing YANG Models

- ***PYANG*** - CLI-driven tool – popular
  https://github.com/mbj4668/pyang
- ***Yang Explorer*** – graphical – my preference because of features
  https://github.com/CiscoDevNet/yang-explorer
- ***Yang Suite*** – early, active development –
  released on Cisco internal GitHub
  https://wwwin-github.cisco.com/yang-suite/yang-suite-dev


localhost:8088/static/YangExplorer.html
Yang Explorer 0.8.0 (Beta)
Help
Admin
Refresh
jadavis
Explorer
search
Values
Operation
Cisco-IOS-XR-ipv6-acl-oper
Cisco-IOS-XR-ipv6-io-oper
Cisco-IOS-XR-ipv6-ma-oper
Cisco-IOS-XR-ipv6-nd-oper
Cisco-IOS-XR-ipv6-new-dhcpv6d-oper
Cisco-IOS-XR-manageability-object-tracking-ope
Cisco-IOS-XR-nto-misc-oper
Cisco-IOS-XR-nto-misc-shmem-oper
Cisco-IOS-XR-nto-misc-shprocmem-oper
Cisco-IOS-XR-pbr-oper
Cisco-IOS-XR-qos-ma-oper
Cisco-IOS-XR-telemetry-model-driven-oper
Cisco-IOS-XR-wanphy-ui-oper
Cisco-IOS-XR-wdsysmon-fd-oper
system-monitoring
cpu-utilization
node-name
total-cpu-one-minute
total-cpu-five-minute
total-cpu-fifteen-minute
process-cpu
process-name
<get>
process-id
process-cpu-one-minute
<get>
process-cpu-five-minute
process-cpu-fifteen-minute
Config
Oper
Add
Delete
Reset
Build
Collections
Manage Models
Operations
Device Settings
Profile
labASR9912
Create device profile
Platform
iosxr
Host
10.
Port
22
Username
Password
Could be port 830
NetConf
RestConf
RPC
Python
YDK
Capabilities
Encoding
Console
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
<filter>
<system-monitoring xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-wdsysmon-fd-oper">
<cpu-utilization>
<process-cpu>
<process-name/>
<process-cpu-one-minute/>
</process-cpu>
</cpu-utilization>
</system-monitoring>
</filter>
</get>
</rpc>
Custom RPC
Run
Save
Clear
Copy
Property
Value
Name
process-cpu-one-minute
Node Type
leaf
Data Type
uint32
Access
read-only
Presence
Key
Mandatory
Default
Path
Cisco-IOS-XR-wdsysmon-fd-oper/system-monitoring/cpu-utilization/process-cpu/process-cpu-one-minute
Description
Process CPU utilization in percent for past 1 minuteNone
XPath Filter
/wdsysmon-fd-oper:system-monitoring/cpu-utilization/process-cpu/process-cpu-one-minute
Reading Telemetry data via NETCONF for analysis
Status : Recieved HTTP Result for request type rpc
IETF 93

Yang Explorer

Getting telemetry data via NETCONF is different than via streaming gRPC/TCP – a base collection point is enforced



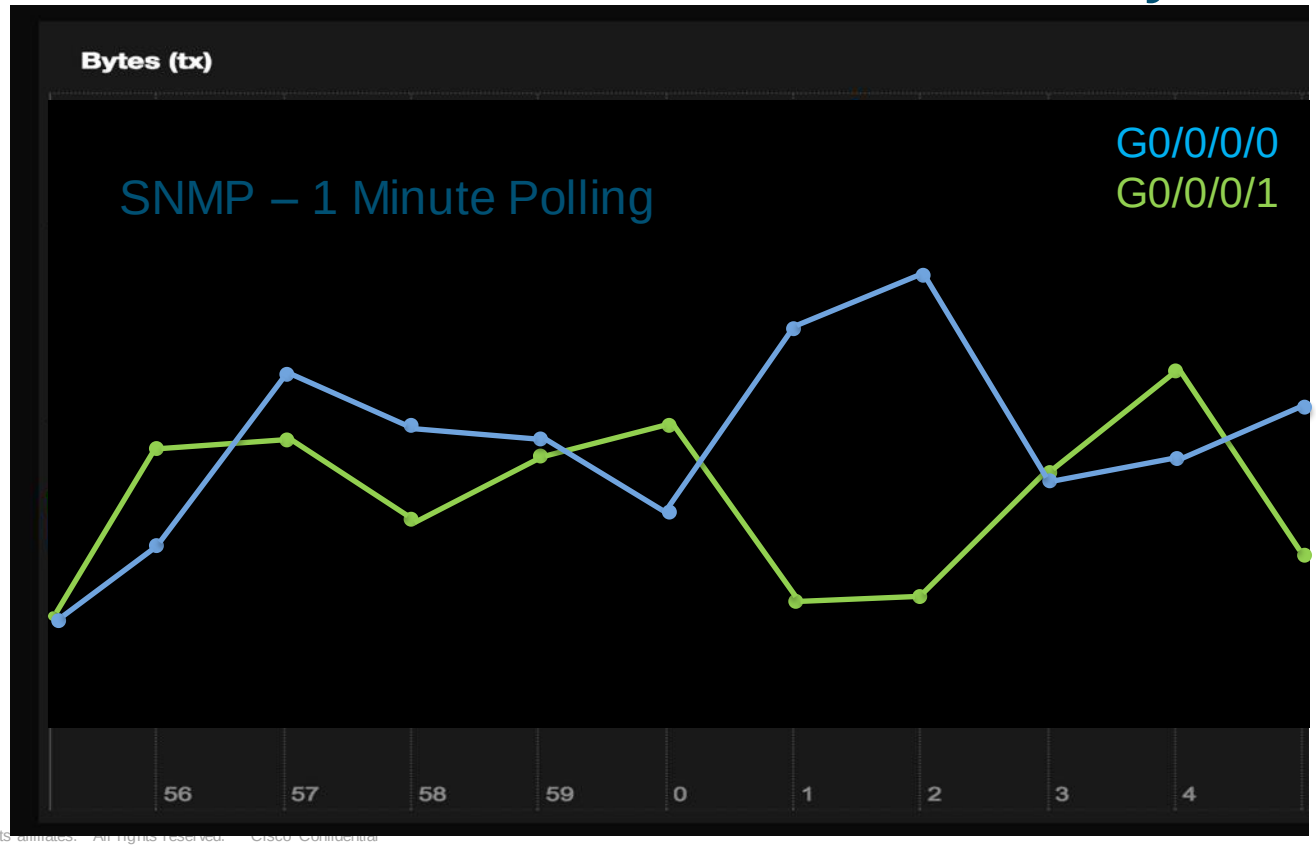
Some Recommended Sensors

Feature/Function	Yang Model:Sensor Path	Minimum Polling Frequency
CPU	Cisco-IOS-XR-wdsysmon-fd-oper:system-monitoring/cpu-utilization	60 sec
Memory	Cisco-IOS-XR-nto-misc-oper:memory-summary/nodes/node/summary	10 sec
Interface	Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/generic-counters OR… Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/cache/generic-counters Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/data-rate	5 sec 30 sec 60 sec
Optical Power Levels	Cisco-IOS-XR-dwdm-ui-oper:dwdm/ports/port/info/optics-info	15 sec
Routing Table counts	Cisco-IOS-XR-ip-rib-ipv4-oper:rib/rib-table-ids/rib-table-id/summary-protos/summary-proto	120 sec

Note: Min Polling Frequency are MY recommendations based on use/need/capability – some sensors can export more quickly, but data accuracy may become suspect

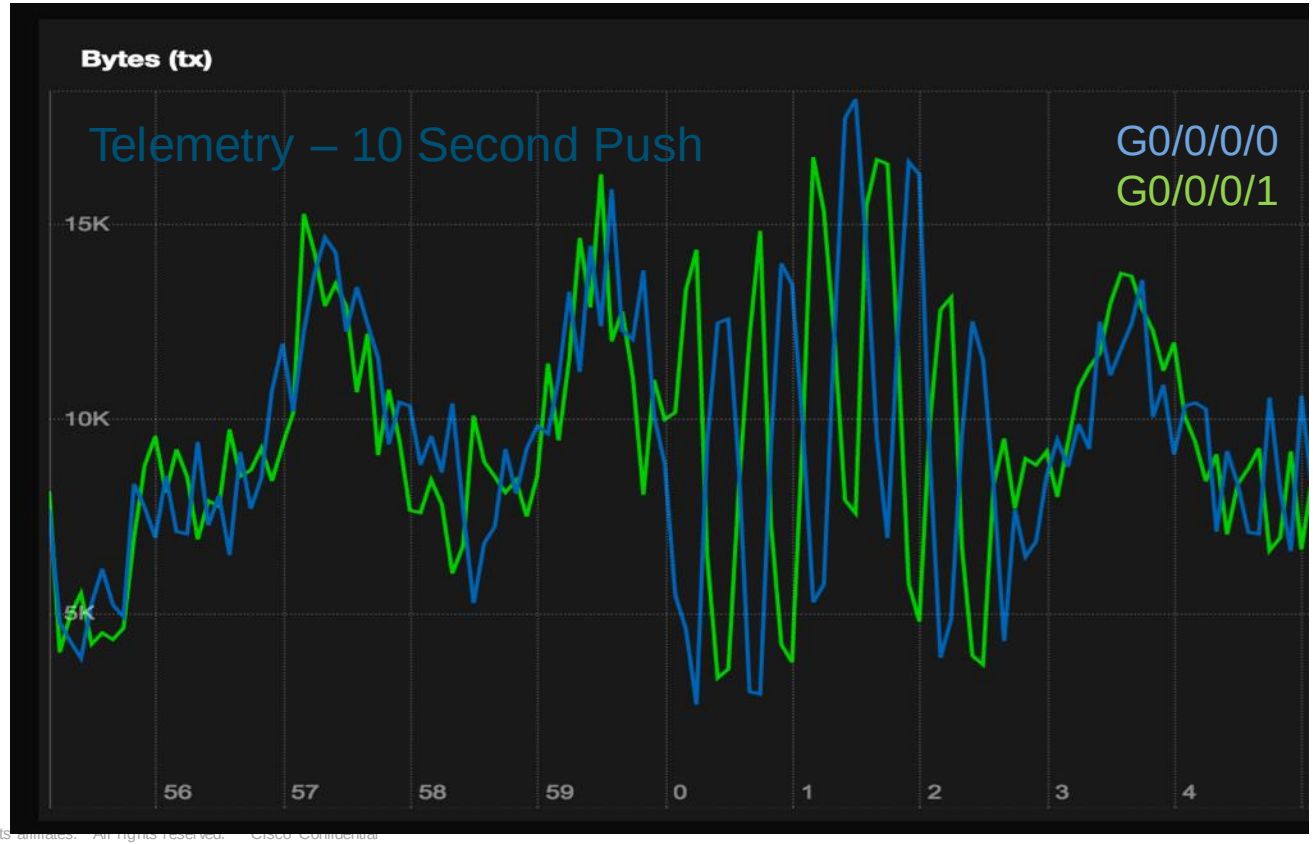
Bundle Polarization: SNMP vs. Telemetry

Is the bundle polarized?



Use Case: Bundle Polarization

Is the bundle polarized?



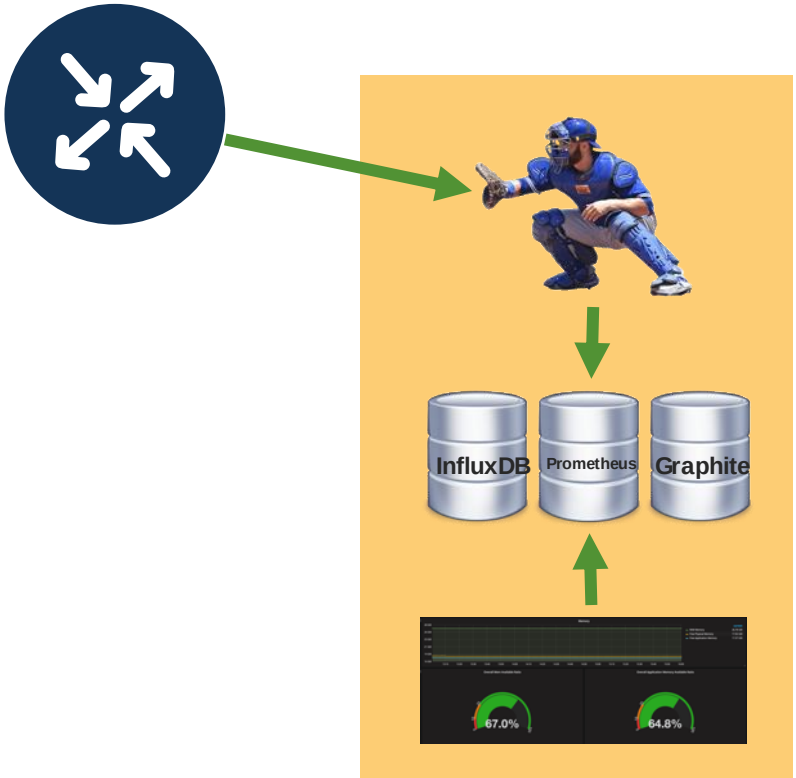
3) Define Subscription

Dial-Out Example
IOS-XR

```
telemetry model-driven
 subscription MDT-Sub01
  sensor-group-id SG001-CPU sample-interval 60000
  sensor-group-id SG002-Memory sample-interval 60000
  sensor-group-id SG003-Ints-10G sample-interval 10000
  destination-id Destination01
```

Collector

Points of Consideration

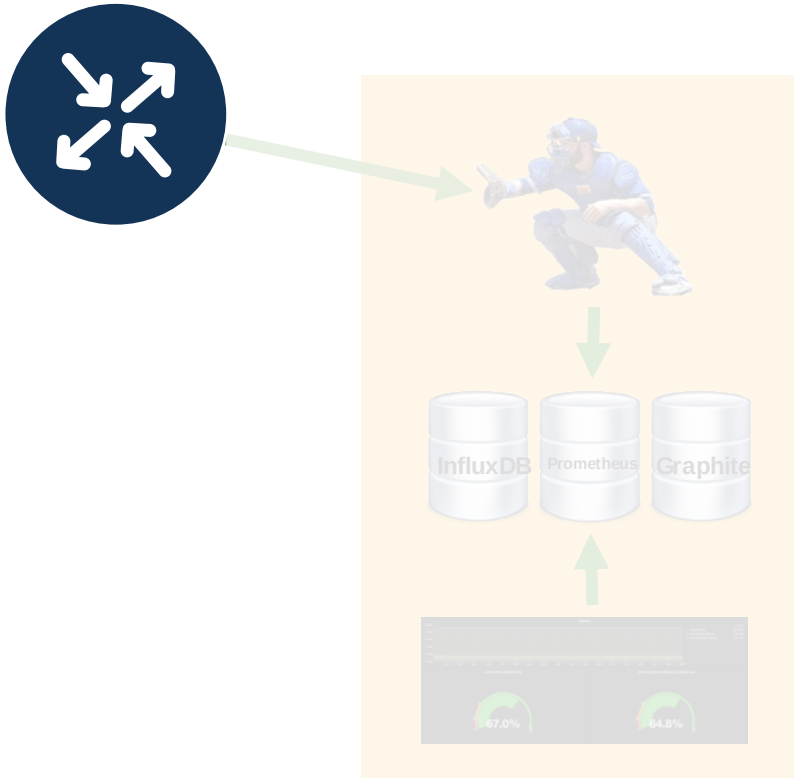


Collector

Database

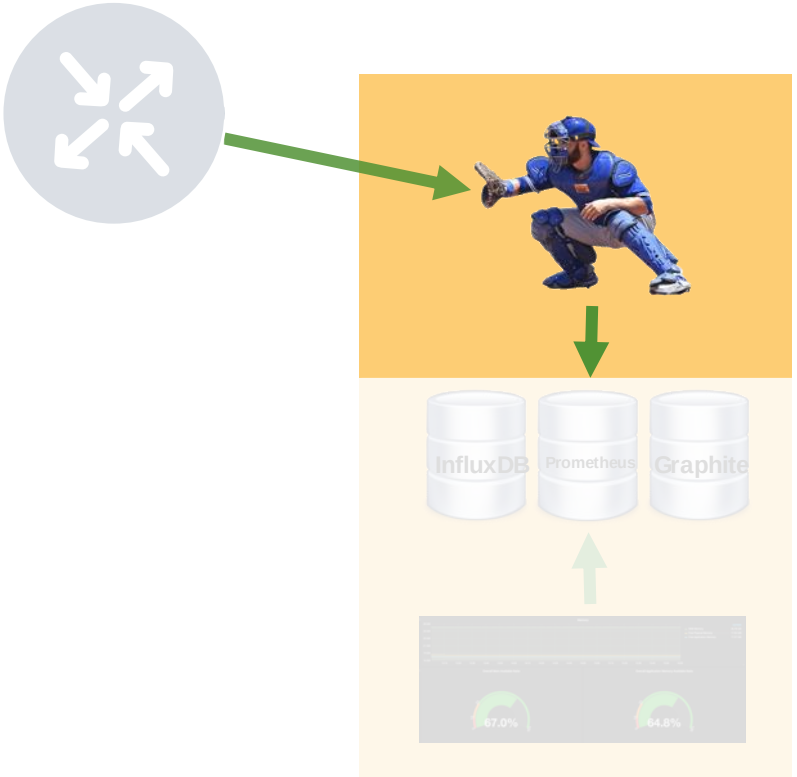
Graphing/Dashboard

Points of Consideration



- What sensors/instrumentation are supported?
- What is max/min frequency of export?
- Can it be exported by ’event’ (on-change)?
- What is the most-specific ‘branch’ needed to export?

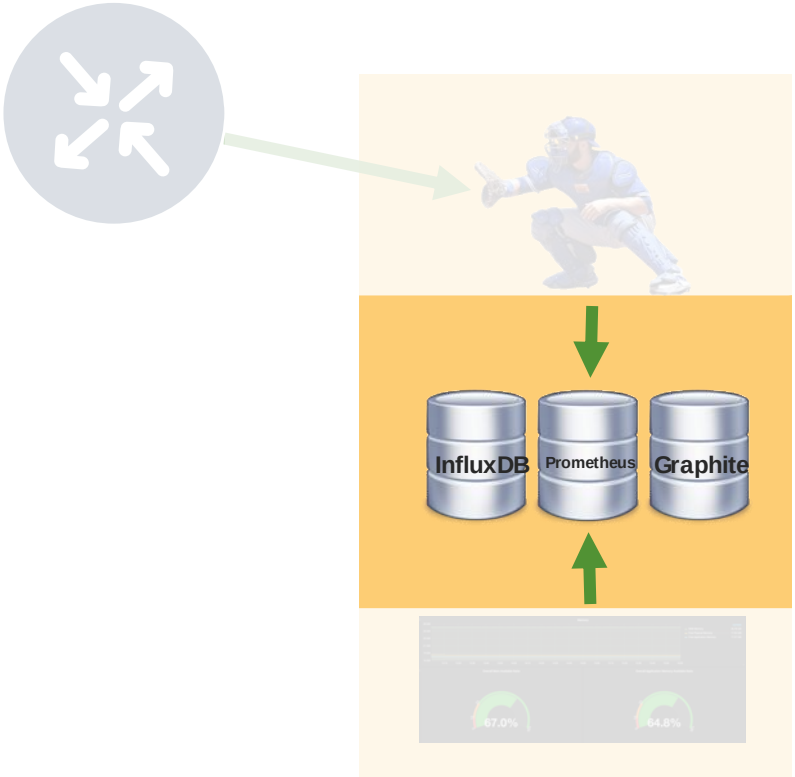
Points of Consideration



Collector

- What is the max rate of raw data ingestion into the collector?
- What is the max rate of transformed data export from the collector?
- Can I apply rules/policies against the telemetry and do alerting?
- Are those rules/policies dynamic/ML or manually created?

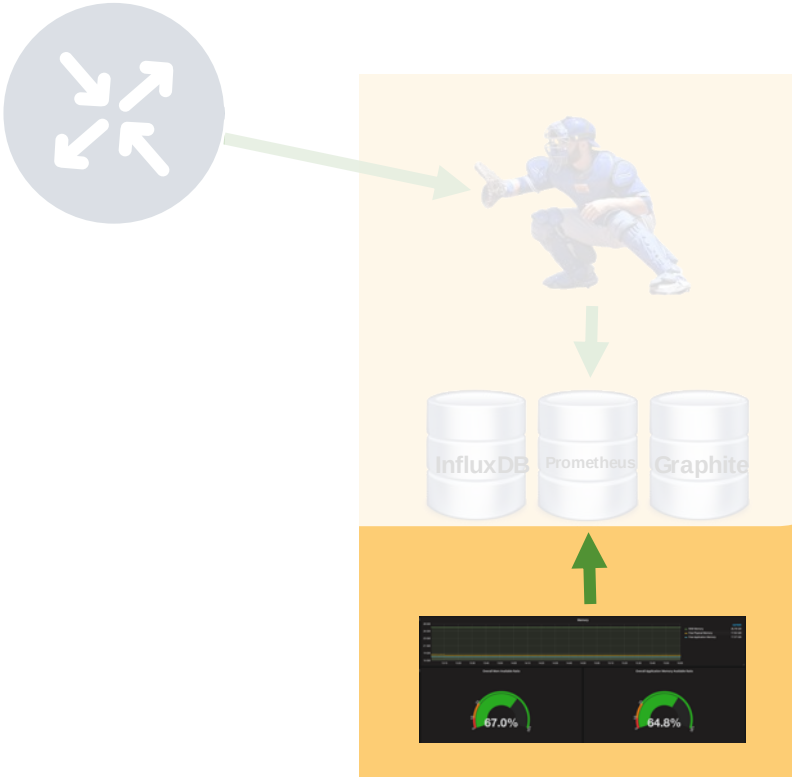
Points of Consideration



Database

- What is the maximum rate of data ingestion from all collector sources?
- What is the maximum rate of db polling from all querying sources?
- What is the desired minimum polling latency?
- How do I handle missing data?
- How do I handle data retention and growth?
- What is the policy on data roll-up/ aggregation?

Points of Consideration

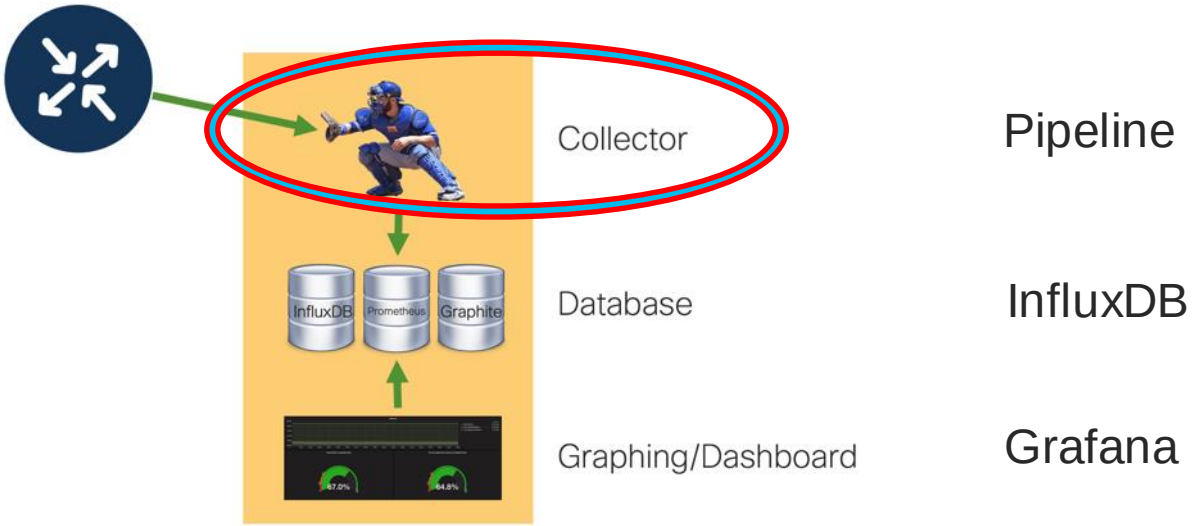


- What is the maximum rate I can poll the db?
- How do I handle missing data?
- Who builds the queries/policies?
- Can I perform fault/alerts from the data?
- How do I want to represent the data?

Graphing/Dashboard

Pipeline

- https://github.com/cisco/bigmuddy-network-telemetry-pipeline



Grafana

Graph | General | Metrics | Axes | Legend | Display | Alert | Time range

A | FROM | default | Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/generic-counters | WHERE | +

SELECT | field (bytes-received) | mean () | non_negative_derivative (1s) | math (*8) | +

GROUP BY | time ($__interval) | tag (interface-name) | fill (null) | +

FORMAT AS | Time series

ALIAS BY | $col.[[tag_interface-name]]

B | Add Query

Panel Data Source | default

Graph | General | Metrics | Axes | Legend | Display | Alert | Time range

A | SELECT non_negative_derivative(mean("bytes-received"), 1s) *8 FROM "Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/generic-counters" WHERE $timeFilter GROUP BY time($__interval), "interface-name

FORMAT AS | Time series | ALIAS BY | $col.[[tag_interface-name]]

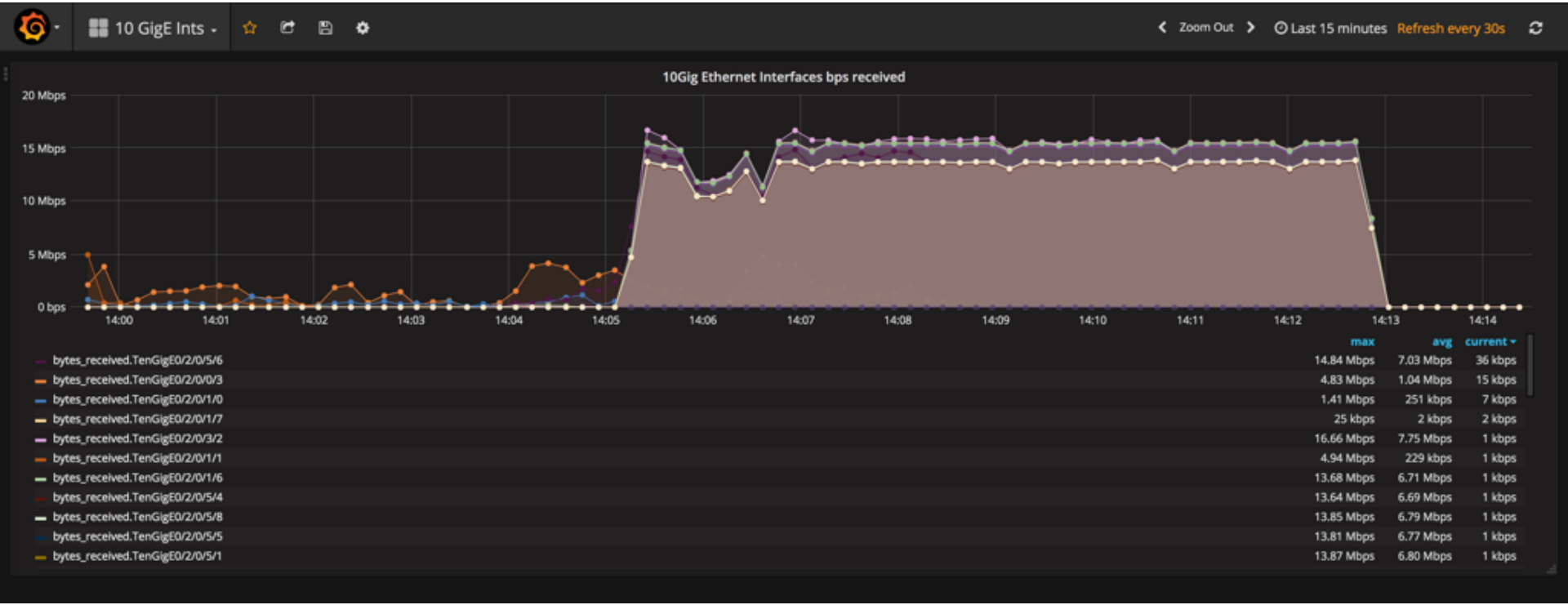
B | Add Query

Panel Data Source | default

Group by time interval | example: >10s

alias patterns | stacking & fill | group by time

Grafana

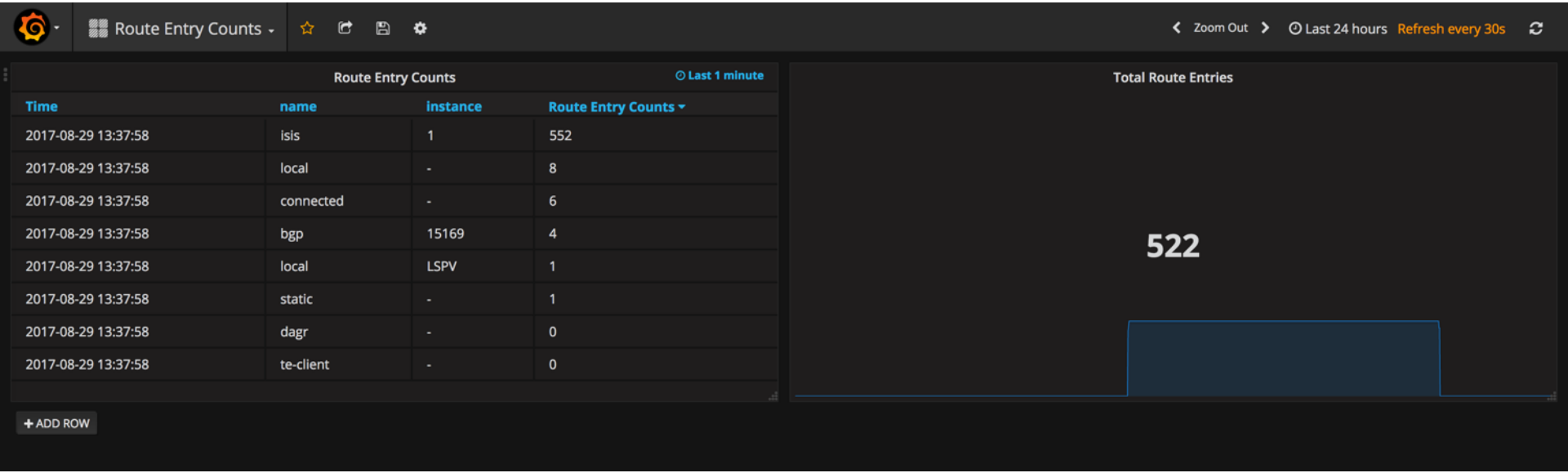



Grafana
ASR 9912 CPU & Processes
Zoom Out
Last 3 hours
CPU
Percent
70
60
50
40
30
20
10
0
11:20
11:30
11:40
11:50
12:00
12:10
12:20
12:30
12:40
12:50
13:00
13:10
13:20
13:30
13:40
13:50
14:00
14:10
max avg current
0/2/CPU0 1min Avg 18 6 9
0/RP0/CPU0 1min Avg 63 13 5
Top Non-Zero Utilization Processes
Percent
30
25
20
15
10
5
0
11:20
11:30
11:40
11:50
12:00
12:10
12:20
12:30
12:40
12:50
13:00
13:10
13:20
13:30
13:40
13:50
14:00
14:10
max avg current
1min-avg Process: alarm-logger 1 0 0
1min-avg Process: bgp 1 0 0
1min-avg Process: emsd 3 3 3
1min-avg Process: fib_mgr 1 0 0
1min-avg Process: netio 1 0 0
1min-avg Process: parser_server 1 0 0
1min-avg Process: perl 1 0 0

Grafana

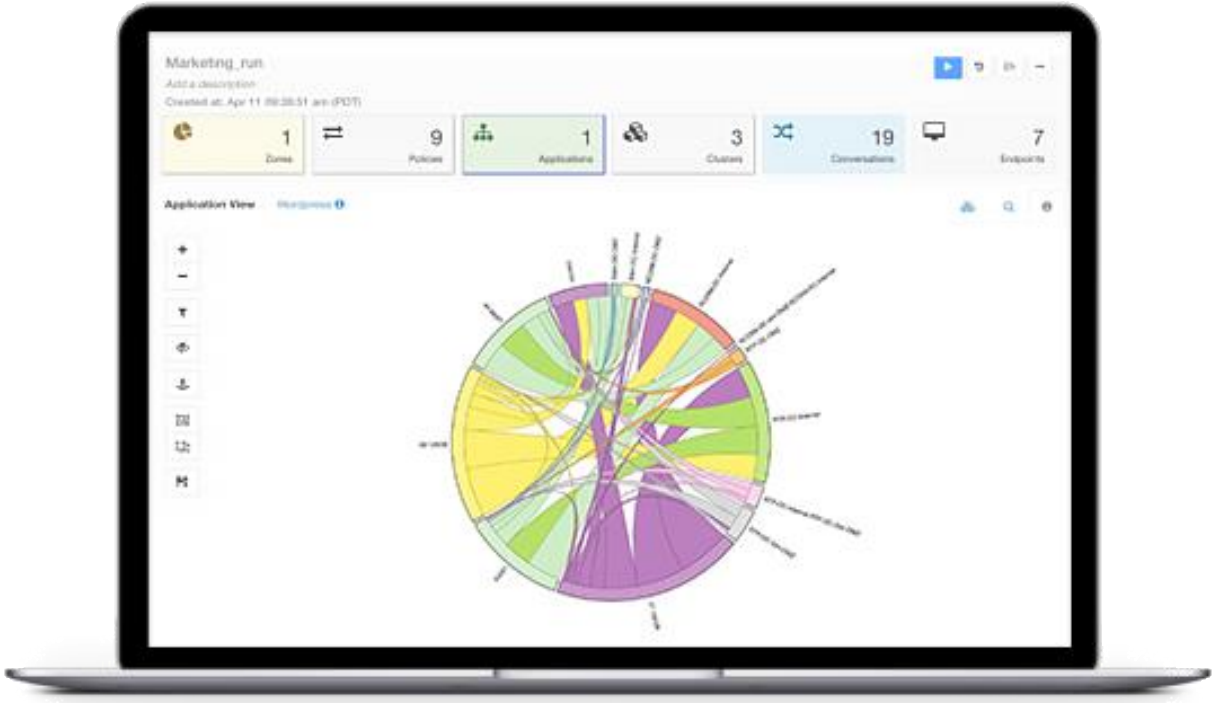
Grafana

Grafana



Other Collectors
Tetration, DNA Center

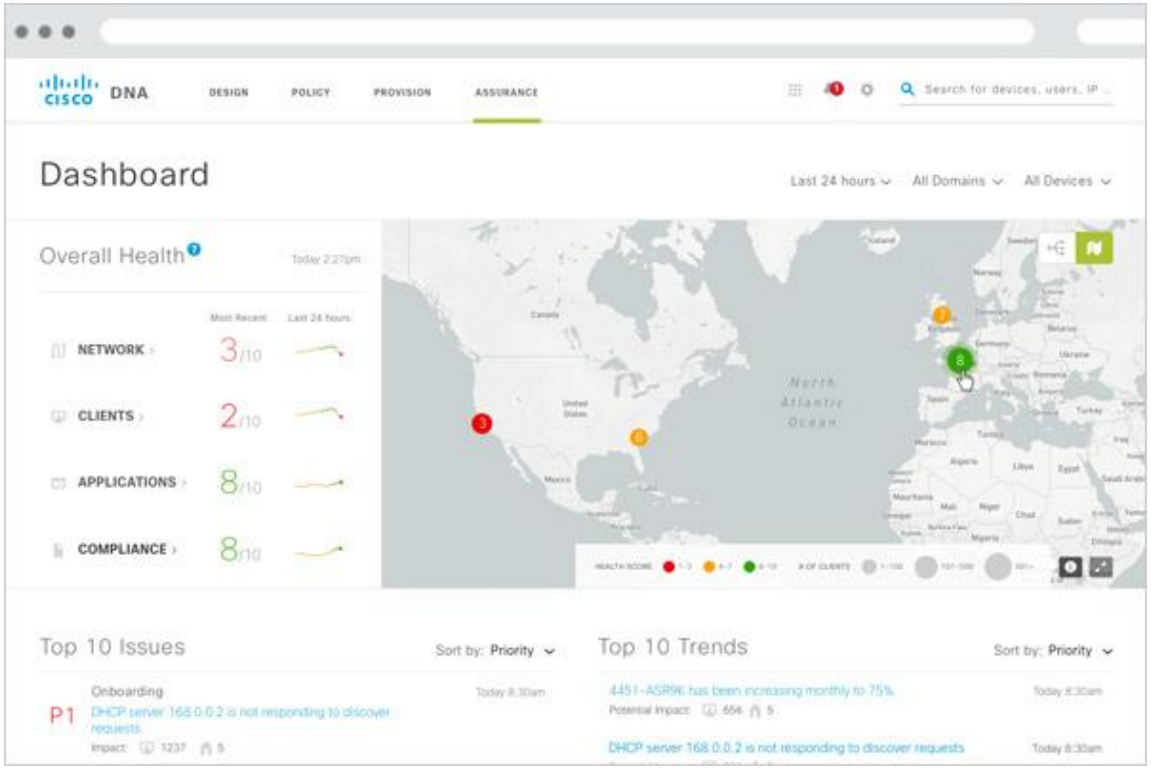
Tetration



Focused on
- ACI DC
- Application/Path

Leverages agents and ASIC technology for instrumentation

DNA Center / Assurance



Focused on Campus/Branch

Leverages SNMP polling, Netflow and Syslog for instrumentation

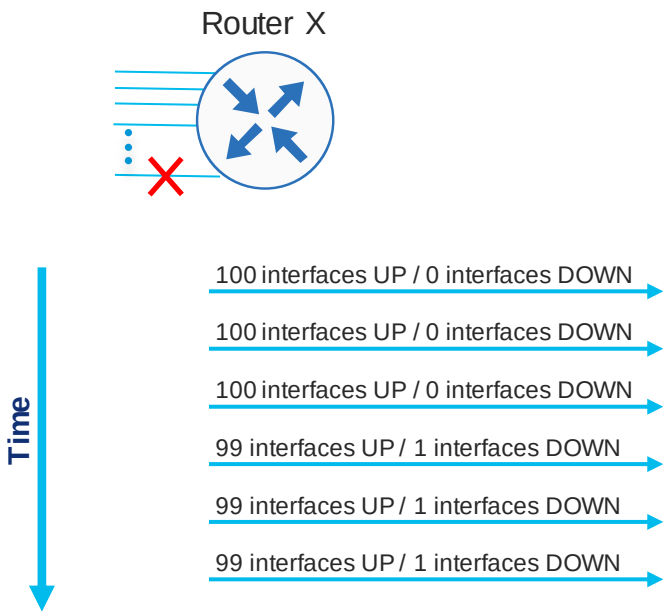
Built with specific 'dashboard' KPIs in mind – APIs coming soon for external expansion

Optimization

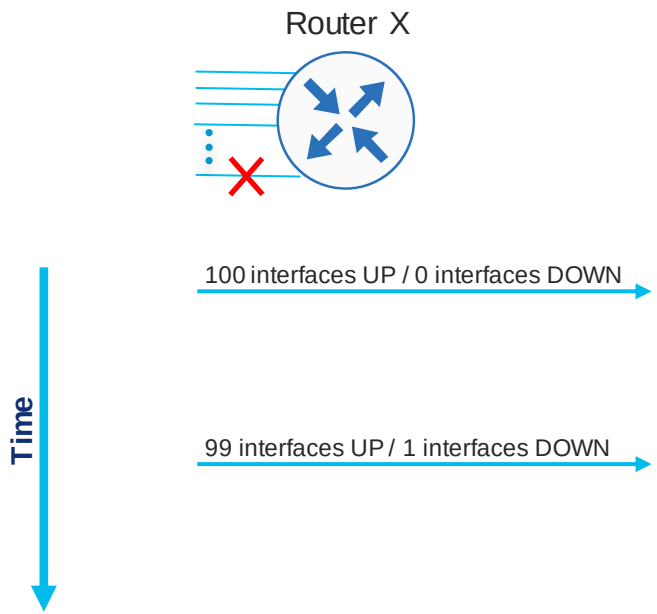
Event-Driven Telemetry overview

Event Driven Telemetry (starts with XR 6.3.1) is complimentary to MDT. The main goal is to send updates whenever they happen. At the start, EDT sends the dump of all the data for the configured sensor-path and only updates are streamed after.

Model-Driven Telemetry



Event-Driven Telemetry



Use Presenter View for this slide

- Recognize export frequency limits on devices/functions
- Gauge export requirements with frequency limits
- Determine which metrics makes sense for periodic exports versus event-based [eg. export only when value changes]
- Remember Telemetry isn't JUST about the network devices being monitored – apply it also to the collectors and databases so you know when you're over-capacity on your tools!

1st Use Case: Cisco StealthWatch

Discovery - Cisco Encryption Traffic Analytics (ETA)

Malware in encrypted traffic

Is the payload within the TLS session malicious?

- End to end confidentiality
- Channel integrity during inspection
- Adapts with encryption standards

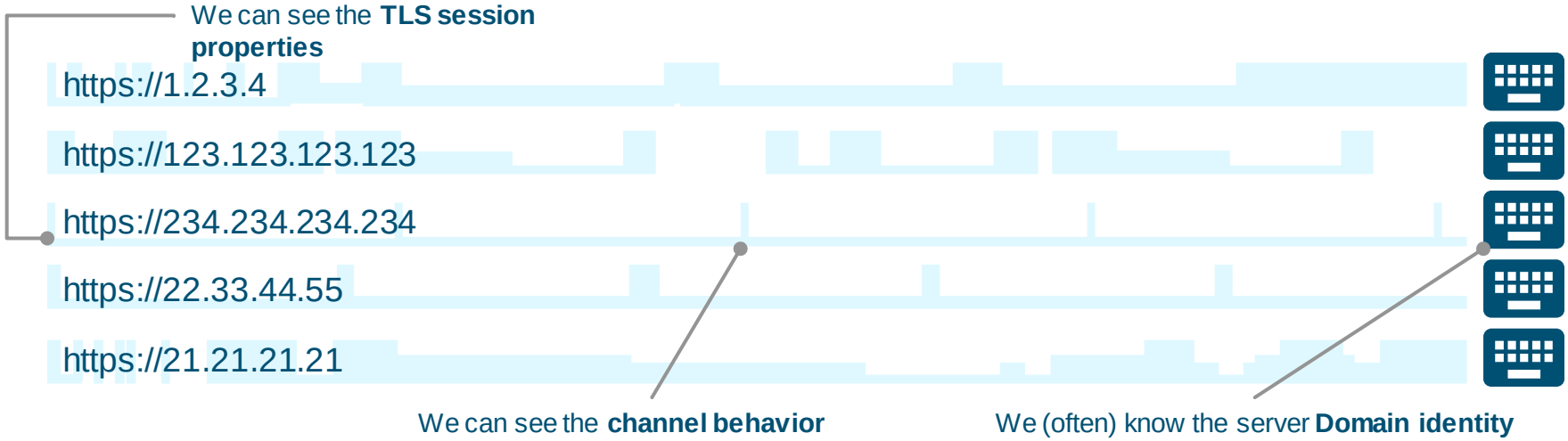
Cryptographic compliance

How much of my digital business uses strong encryption?

- Audit for TLS policy violations
- Passive detection of Ciphersuite vulnerabilities
- Continuous monitoring of network opacity

Cisco StealthWatch

Discovery - Cisco Encryption Traffic Analytics (ETA)



- TLS session properties - Crypto information educates us on client/server behavior and application identity
- Channel behavior - Size/timing of the packets allow us to estimate type of data inside the encrypted channel
- Domain identity - Models +20 features of 150M risky endpoints on the internet (Domain, Whois, TLS Cert…)

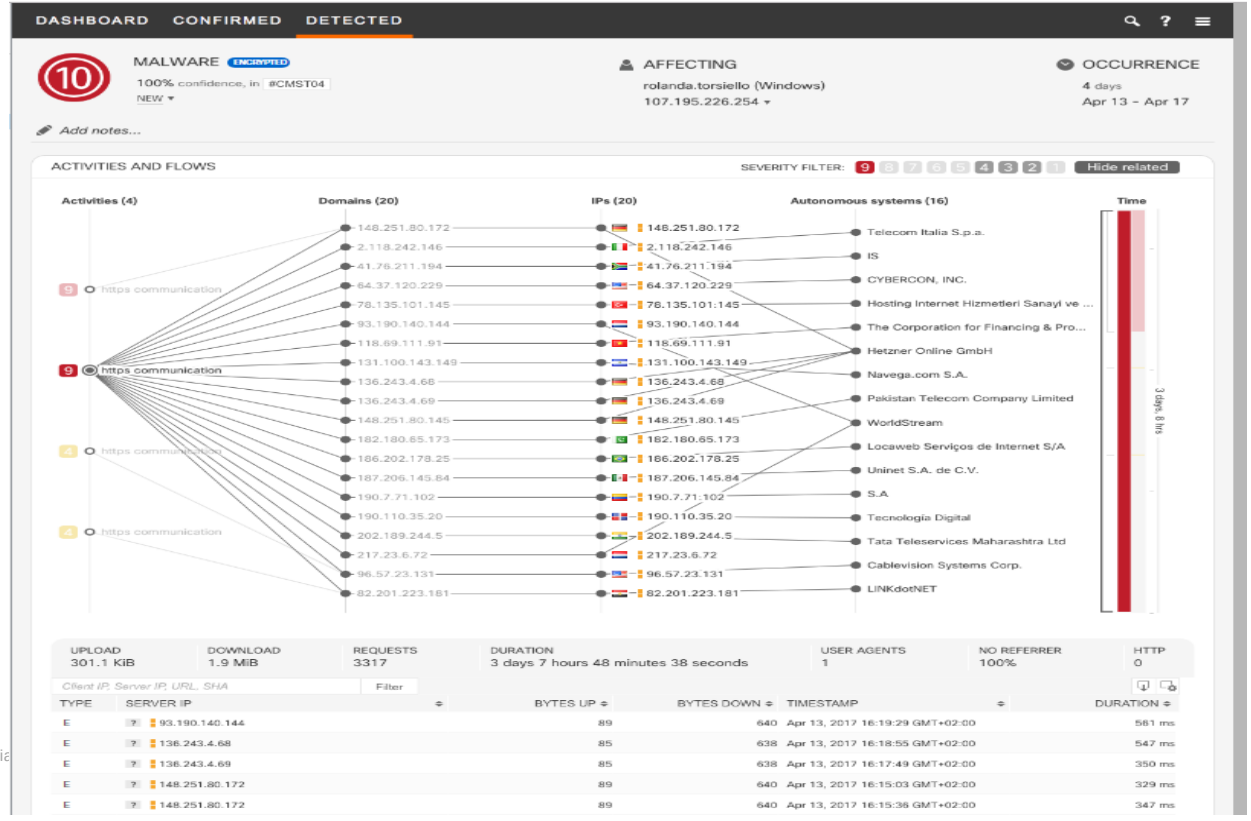
Encrypted Traffic Analytics: Foundation

Encrypted Traffic Analytics extracts four main data elements:

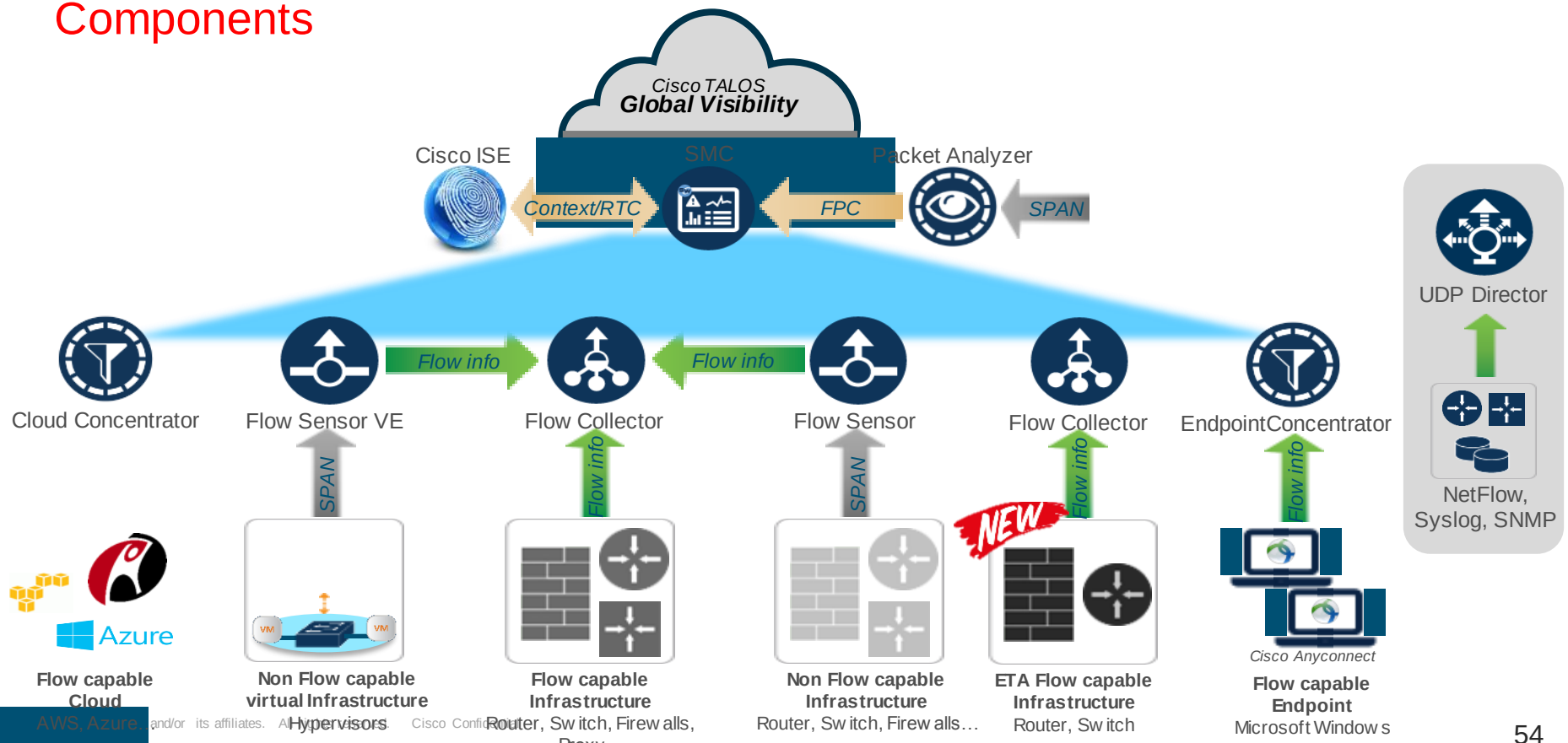
1. **Sequence of Packet Lengths and Times** (SPLT): SPLT conveys the length (number of bytes) of each packet's application payload for the first several packets of a flow, along with the interarrival times of those packets.
2. **Initial Data Packet** (IDP): IDP is used to obtain packet data from the first packet of a flow. It allows extraction of interesting data such as an HTTP URL, DNS hostname and address, and other data elements. 3. 3.
3. **Byte distribution**: The byte distribution represents the probability that a specific byte value appears in the payload of a packet within a flow. 4. 4.
4. **TLS-specific features**: The TLS handshake is composed of several messages that contain interesting, unencrypted metadata used to extract data elements, such as cipher suite, TLS version, and the client's public key length.

Cisco StealthWatch

IOC identification – Encrypted Traffic Analytics

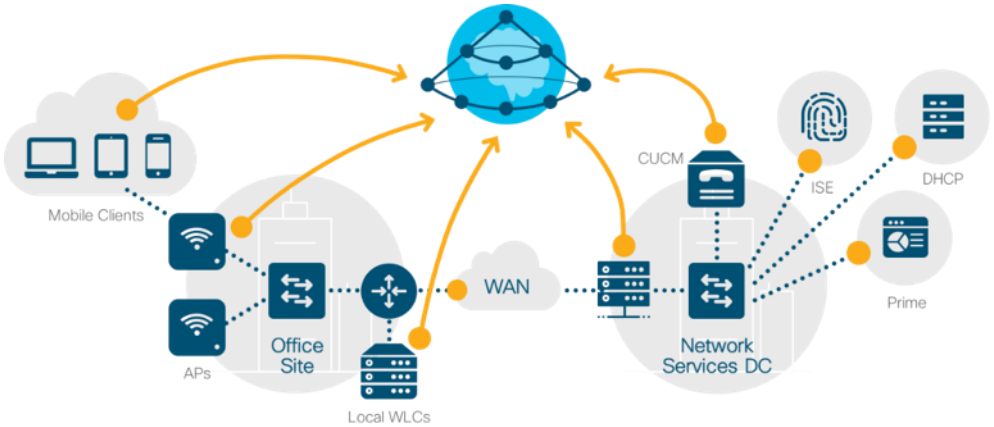


Cisco StealthWatch
Components



2nd Use case:
Machine Learning
For Wireless, Wired Networks and IOT

End to end visibility

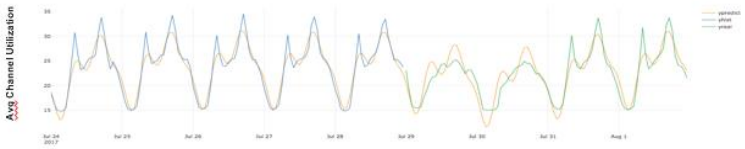


Predictive Analytics

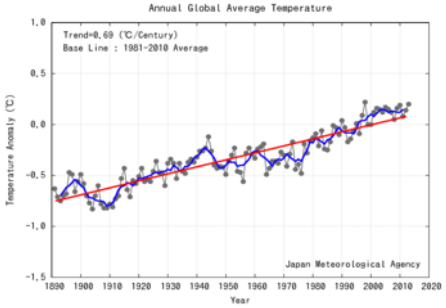
DNA Analytics detects (complex) early signs of an an issue, in the future, with confidence levels

What is Predictive Analytics ?

- Seasonality & Trends:



Model (complex) seasonality



Long term trending

- True Predictive is more ambitious: learn complex/subtle early signs that an event of interest will happen
- *Why not a simple rule ?* If CPU > 80% & Memory < 15% then Router crashes => if the event were so deterministic we would avoid it in the first place
- Such combination of events leading to a target is simply unknown a priori: we want the machine to find such a rule and learn

Thanks!