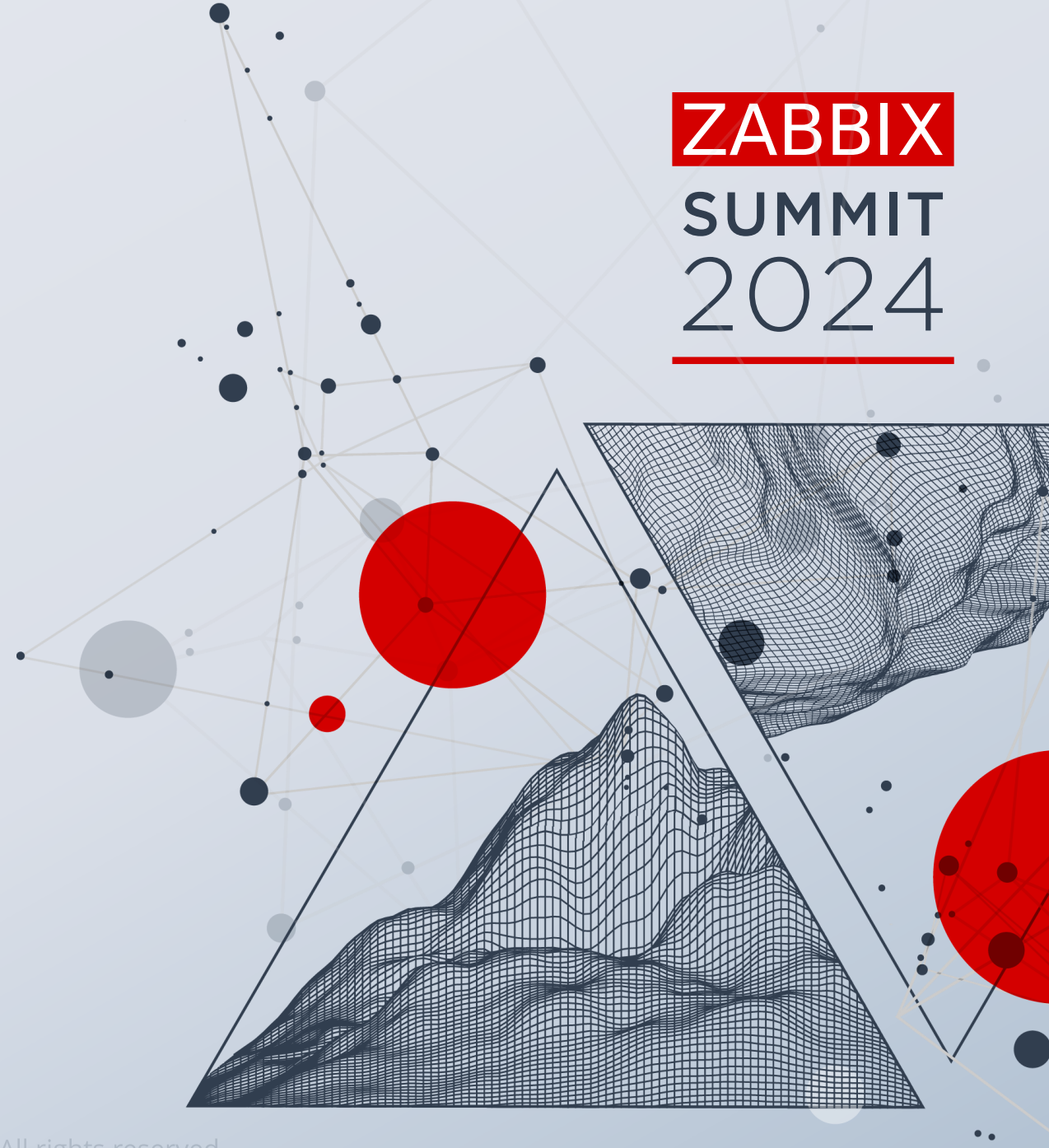
ZABBIX SUMMIT 2024

Model Driven Telemetry and Zabbix



Roberto Suárez Soto

Team Lead System Administrator

About us

ZABBIX SUMMIT 2024

We are a boutique IT services company specialized in the design and implementation of integrated scalable solutions, with a strong focus on content acceleration, monitoring systems, observability, and security.

ZABBIX

SIEM

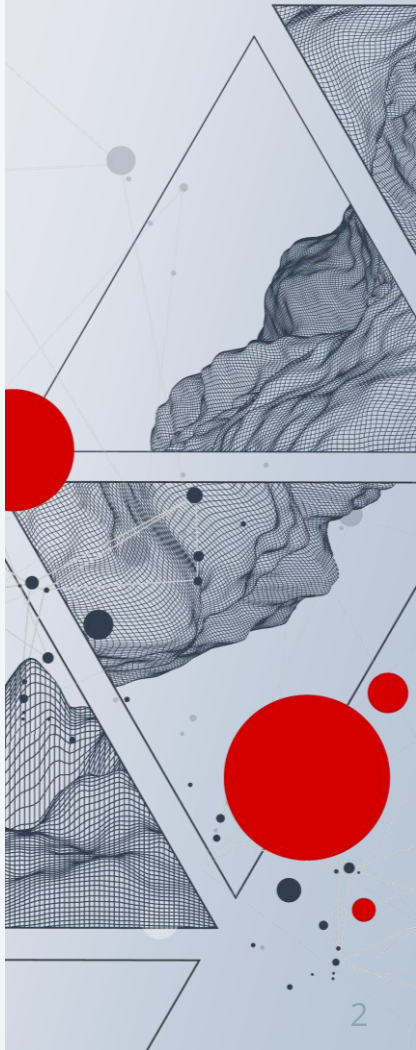
Content Acceleration

Observability & Monitoring

DevOps and Systems Integration

Systems Administration

Software Development



ZABBIX SUMMIT 2024

Scenario

Scenario

ZABBIX SUMMIT 2024

- Our client produces the live television, radio and digital coverage for one of the largest worldwide sports event which takes place every 4 years
- Scenario formed (mostly) by video streaming devices connected to Cisco switches (Catalyst and Nexus) and routers (NCS)
- Client wanted to use Model Driven Telemetry both because they needed real time monitoring of network traffic and because they wanted to try this technology
- Internet bandwidth usage and device availability were major concerns
- Zabbix was used to gather the monitoring data and generate alerts, Grafana to display the information for the IT team
- Some numbers:
  - Zabbix 6.0 (ok, not strictly a number)
  - 90 devices
  - ~600GB database
  - ~2k NVPS on average, peak 3k

ZABBIX SUMMIT 2024

What is Model Driven Telemetry

What is MDT: in a nutshell

ZABBIX SUMMIT 2024

- Model Driven Telemetry ("MDT") provides a mechanism to stream monitoring data from an MDT-capable device to a destination receiver
- Applications can subscribe to specific data items they need by using standards-based YANG data models over open protocols
    - YANG models define all the possible telemetry data points that can be collected from a device
    - Publicly available (Github)
    - XPath-like syntax
- Structured data is published at a defined cadence or on-change
- Dial-out: pushes the data to a destination group (defined by adress, port, encoding and transport)
    - Uses raw TCP or gRPC
    - Needs predefined subscriptions
- Dial-in:
    - Also uses gRPC; bidirectional once established connection
    - Subscriptions are created on the fly

What is MDT: device configuration

ZABBIX SUMMIT 2024

- NETCONF, gNMI or SSH can be used to configure the devices
- "Sensor groups" define groups of monitoring data, using YANG models
  - For example, you can define a "cpu" sensor group with all the CPU-related data points
  - Not used in all devices
- "Subscriptions" define how to send the data
  - What data to send
  - Frequency of data collection (data is collected only once per period)
  - Destination (IP address of the collector)
  - Protocol to be used
  - Can be predefined (for dial-out mode) or defined on the fly (for dial-in mode)

```
telemetry model-driven
destination-group ALLENTA-DESTINATION
  address-family ipv4 172.16.99.99 port 57000
  encoding self-describing-gpb
  protocol grpc no-tls

sensor-group ALLENTA-HARD-REAL-TIME-SENSOR-GROUP
  sensor-path openconfig-
interfaces:interfaces/interface[name="HundredGigE0/0/1/2
"]/state/counters
  sensor-path openconfig-
interfaces:interfaces/interface[name="HundredGigE0/0/1/3
"]/state/counters

subscription ALLENTA-SUBSCRIPTION
  sensor-group-id ALLENTA-HARD-REAL-TIME-SENSOR-GROUP
strict-timer
  sensor-group-id ALLENTA-HARD-REAL-TIME-SENSOR-GROUP
sample-interval 2000

  destination-id ALLENTA-DESTINATION
```

ZABBIX SUMMIT 2024

Why use MDT instead of SNMP

Why use MDT instead of SNMP

ZABBIX SUMMIT 2024

- "Push" vs "pull" model
  - Dial-out makes for easier network configuration: only configure ACLs in the receiving end
  - No polling required, decreases load on monitoring systems
- TCP and HTTP/2-based transports
  - More robust connections, traffic is not lost so easily
  - Compression and encryption available
- JSON or XML formats available
  - Structured format makes it easier to parse the data
  - It also allows for more complex data to be sent
- Lower CPU and memory cost
  - Direct access to required data
  - Monitoring data is already prepared when asked for, instead of being collected on every request
  - Once generated, the same data can be send to multiple monitoring systems

ZABBIX SUMMIT 2024

MDT and Zabbix

MDT and Zabbix: not supported by default

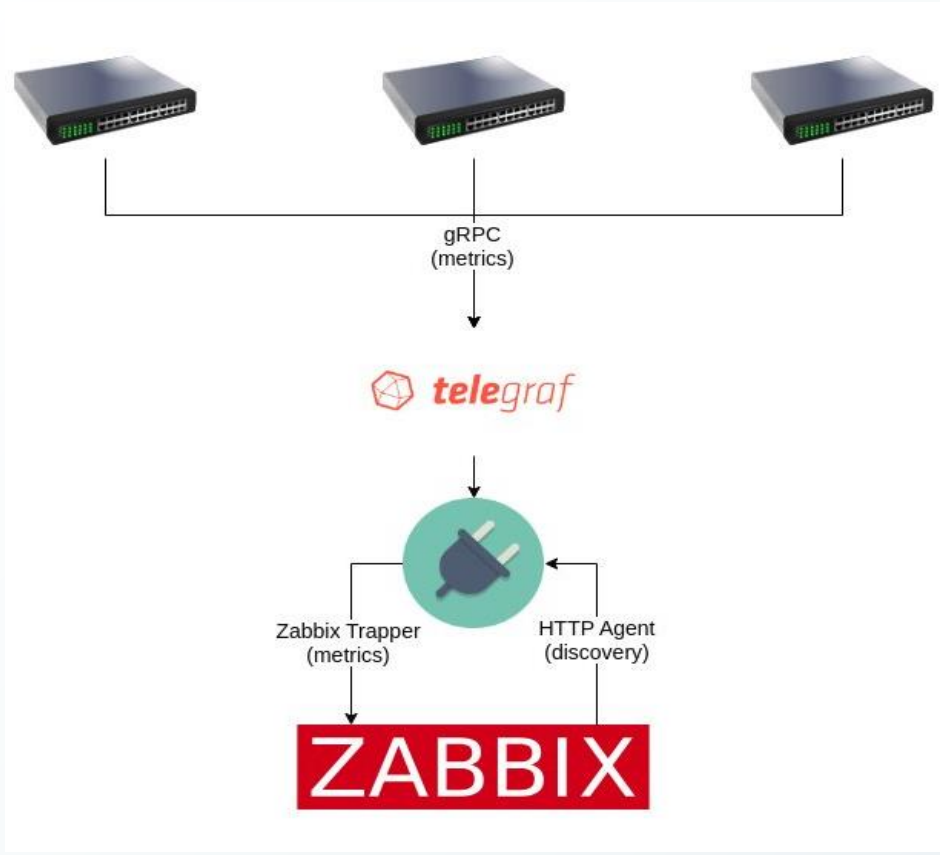
ZABBIX SUMMIT 2024

- Zabbix does not have native support for MDT
  - How do we receive data from the devices?
  - How do we send MDT data to Zabbix?
  - How do we turn MDT entities into Zabbix entities?
- We chose Telegraf to receive the data from the devices, using a Cisco Telemetry plugin
- We bridged the gap from Telegraf to Zabbix with a custom plugin

MDT and Zabbix: Architecture

ZABBIX SUMMIT 2024

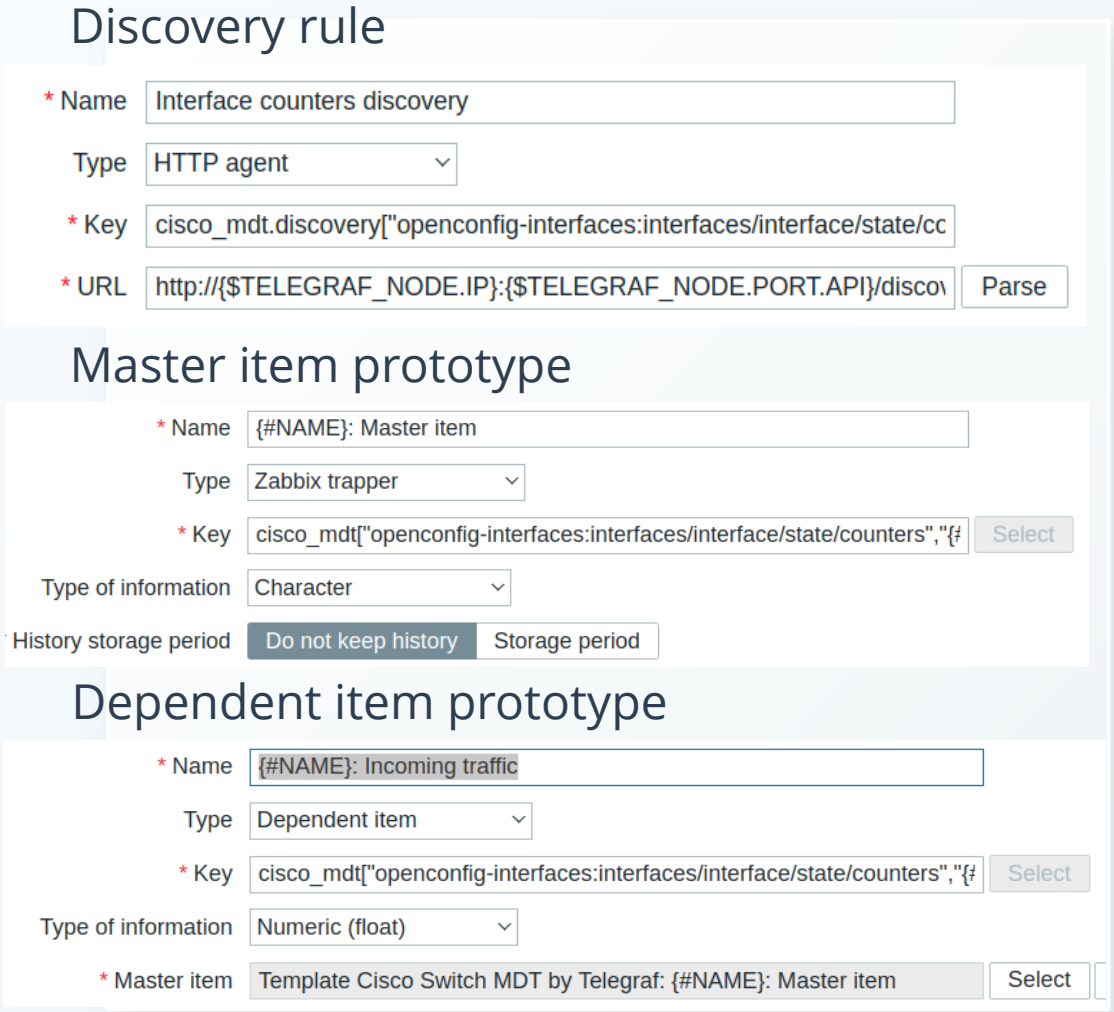
- "Dial out" subscriptions are configured on the devices
- We gather "hard" and "soft" real time statistics for network interfaces:
  - "Hard" real time subscriptions gather data every 2 seconds
  - "Soft" real time subscriptions gather data every 10 seconds
- Devices send MDT data to Telegraf using gRPC
- Telegraf feeds received data to a custom plugin
- The custom plugin parses MDT data and sends it to Zabbix
- First prototype in Python, production version in Go Lang because of performance issues



MDT and Zabbix: Zabbix config

ZABBIX SUMMIT 2024

- The connector exposes an API that Zabbix uses to discover the items to be monitored
- It also sends metrics to Zabbix, using Zabbix sender
- There are two kinds of metrics: normal items and "throttled" items, copies of the previous ones but sent every 60s instead of "real time" (useful when showing big time windows)
- Templates are configured in Zabbix to define the discovery rules, item prototypes and trigger prototypes that can be used with MDT data



MDT and Zabbix: item prototypes

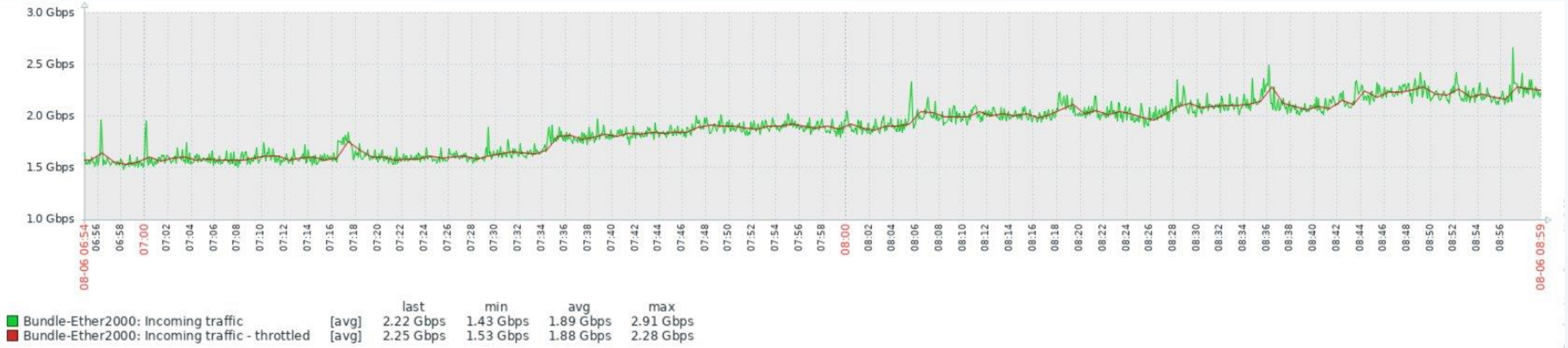
ZABBIX SUMMIT 2024

All templates / Template Cisco Switch MDT ... Discovery list / Interface counters discovery Item prototypes 14 Trigger prototypes 1 Graph prototypes Host prototypes

Name	Key	Interval	History	Trends	Type	Create enabled	Discover	Tags
{#NAME}: Master item: {#NAME}: Incoming discards	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters","{#NAME}","in_discards"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: RT
{#NAME}: Master item - throttled: {#NAME}: Incoming discards - throttled	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters/throttled","{#NAME}","in_discards"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: throttled
{#NAME}: Master item: {#NAME}: Incoming errors	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters","{#NAME}","in_errors"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: RT
{#NAME}: Master item - throttled: {#NAME}: Incoming errors - throttled	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters/throttled","{#NAME}","in_errors"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: throttled
{#NAME}: Master item: {#NAME}: Incoming traffic	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters","{#NAME}","in_octets"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: RT
{#NAME}: Master item - throttled: {#NAME}: Incoming traffic - throttled	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters/throttled","{#NAME}","in_octets"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: throttled
{#NAME}: Master item	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters","{#NAME}"]		0		Zabbix trapper	Yes	Yes	component: raw
{#NAME}: Master item - throttled	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters/throttled","{#NAME}"]		0		Zabbix trapper	Yes	Yes	component: raw
{#NAME}: Master item: {#NAME}: Outgoing discards	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters","{#NAME}","out_discards"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: RT
{#NAME}: Master item - throttled: {#NAME}: Outgoing discards - throttled	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters/throttled","{#NAME}","out_discards"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: throttled
{#NAME}: Master item: {#NAME}: Outgoing errors	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters","{#NAME}","out_errors"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: RT
{#NAME}: Master item - throttled: {#NAME}: Outgoing errors - throttled	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters/throttled","{#NAME}","out_errors"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: throttled
{#NAME}: Master item: {#NAME}: Outgoing traffic	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters","{#NAME}","out_octets"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: RT
{#NAME}: Master item - throttled: {#NAME}: Outgoing traffic - throttled	cisco_mdt["openconfig-interfaces:interfaces/interface/state/counters/throttled","{#NAME}","out_octets"]		90d	365d	Dependent item	Yes	Yes	component: network, interface: {#NAME}, type: throttled

Displaying 14 of 14 found

MDT and Zabbix: "real time" vs "throttled"



- "Real time" data used a lot of datapoints
  - It didn't look good when showing big time windows
  - It caused unnecessary load on the DB
- Throttled data provides a smoother graph and is easier on the DB

ZABBIX SUMMIT 2024

Lessons learned

Lessons learned: works as advertised

ZABBIX SUMMIT 2024

- MDT delivered as promised
  - We were able to gather data in almost real time fashion
  - Once we were aware of some quirks, configuration of the devices was straightforward
- A different point of view when debugging problems
  - You no longer control when devices send data

Lessons learned: some rough edges

ZABBIX SUMMIT 2024

- "Real time" not so much "real time"
  - Performance issues (data not sent at expected intervals) when there is a lot to monitor in less powerful devices
  - CPU usage still considerably high when low interval monitoring is configured for a lot of interfaces
  - Some interface counters not updated in real time in the devices
- "In theory, theory and practice are the same; in practice, they aren't"
  - YANG models and behaviours can change from device to device (even within the same brand), including different subscription formats
  - Eg: sending data points one by one versus sending them all in the same message
  - Eg: slightly different models for "onchanges" and periodic sending
  - Eg: Xpath support more or less complete depending on the device
  - Some of these issues are not a problem if you store data in InfluxDB, but make Zabbix templates more complex

Lessons learned: some workarounds needed

ZABBIX SUMMIT 2024

- Too many datapoints due to RT are tough on the DB
  - Some DB queries become noticeable slower
  - In complex dashboards (especially those with repeats), Zabbix plugin for Grafana not performant enough when using 2s refresh time
- SQL queries used instead of Zabbix plugin for two reasons:
  - A requisite was to "archive" the DB afterwards and use the dashboards without Zabbix
  - Better performance, at the cost of more complex configuration

ZABBIX SUMMIT 2024

Questions?

ZABBIX SUMMIT 2024

Thank you!

Roberto Suárez Soto

Team Lead System Administrator