

Application monitoring in Checkmk

What's new?

Community call, Oct 21st 2025

Niklas Pulina

Product Manager

Checkmk GmbH

checkmk.com

Agenda

01 Recap

02 Demo time: latest improvements

03 What's next?

04 Special guest

05 Q&A





Recap

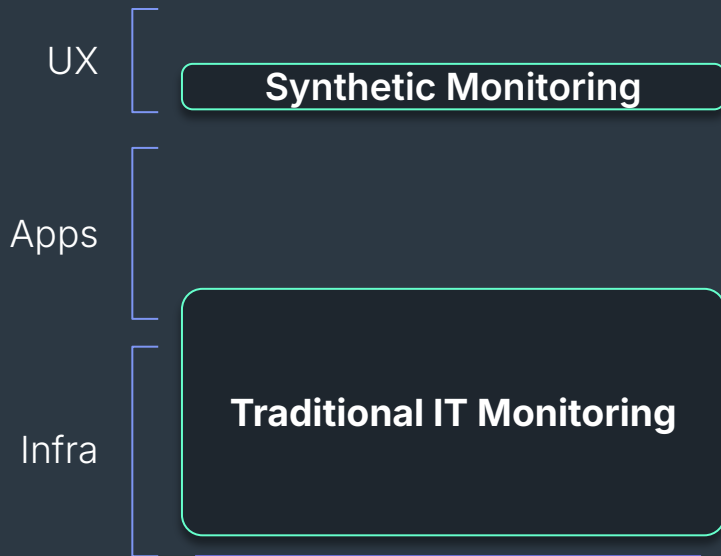


The challenge

What problem do we want to solve?

The IT monitoring market is evolving due to the shift towards cloud-native technologies and DevOps operating models. There's a growing move towards hybrid environments that combine traditional and cloud-native infrastructure.

Checkmk has historically excelled in monitoring traditional infrastructure. However, modern cloud-native applications are more dynamic and complex, requiring different monitoring approaches, often referred to as "observability."

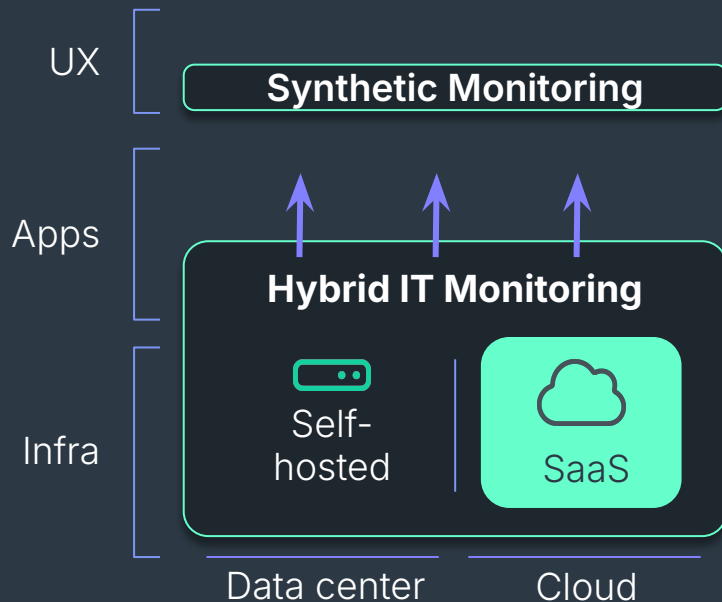


The solution

Application monitoring in Checkmk

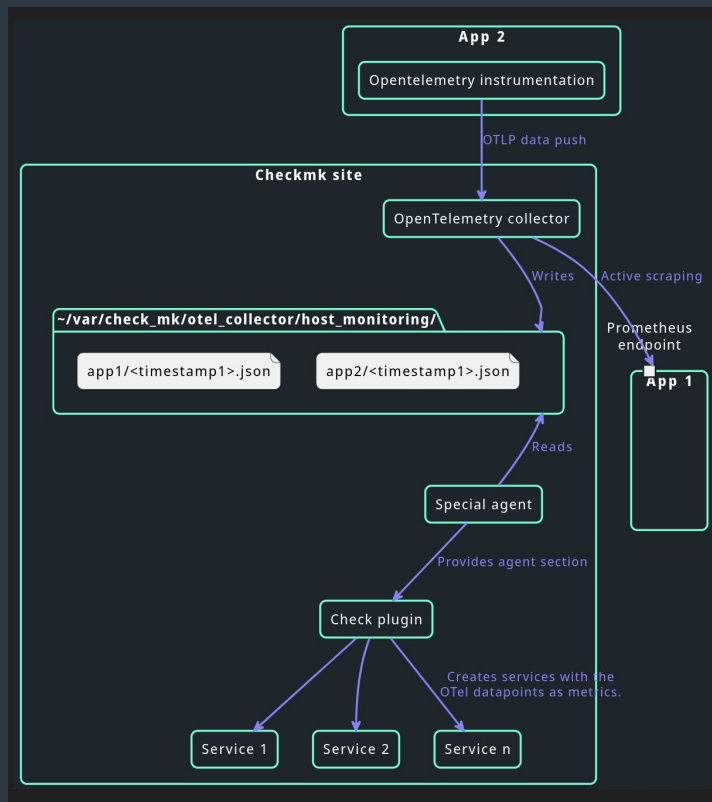
To provide a more comprehensive platform that covers both traditional and cloud-native environments, Checkmk expanded its focus on application monitoring in 2.4.

This strategic shift allows Checkmk to support customers throughout their transformation journeys and cater to the changing requirements of IT monitoring in a hybrid world.



The architecture in Checkmk 2.4

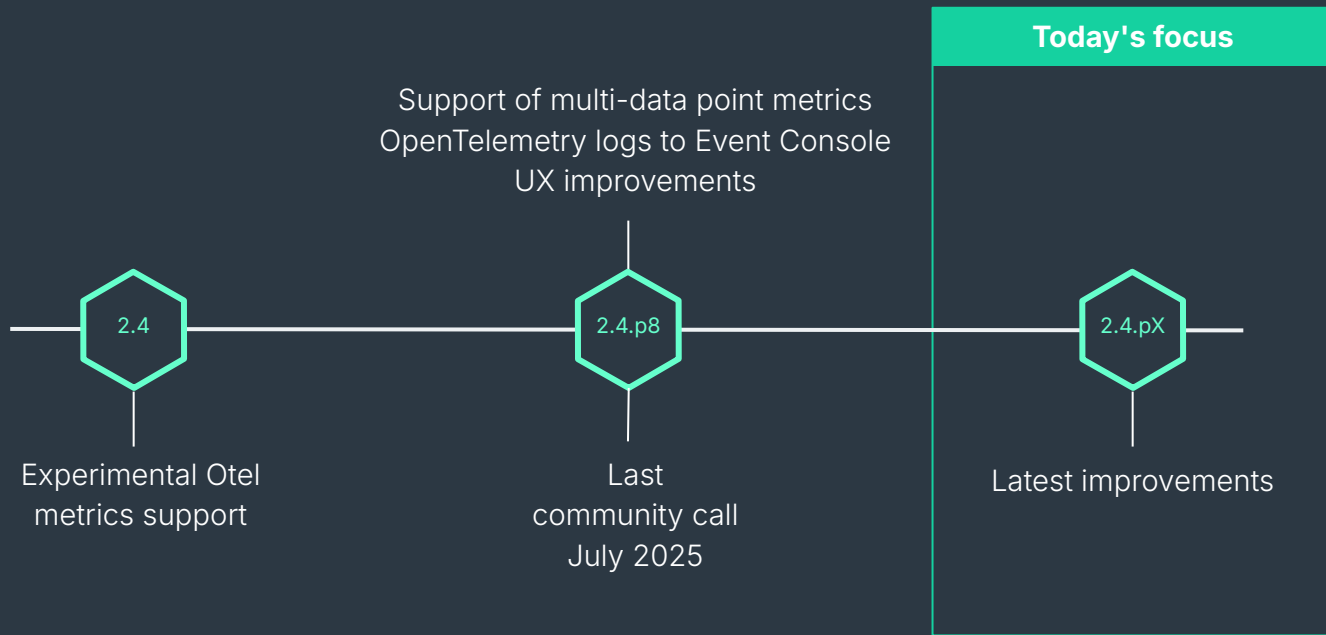
(A simplified version of it)





The delivery

What has happened since the release of Checkmk 2.4.0?



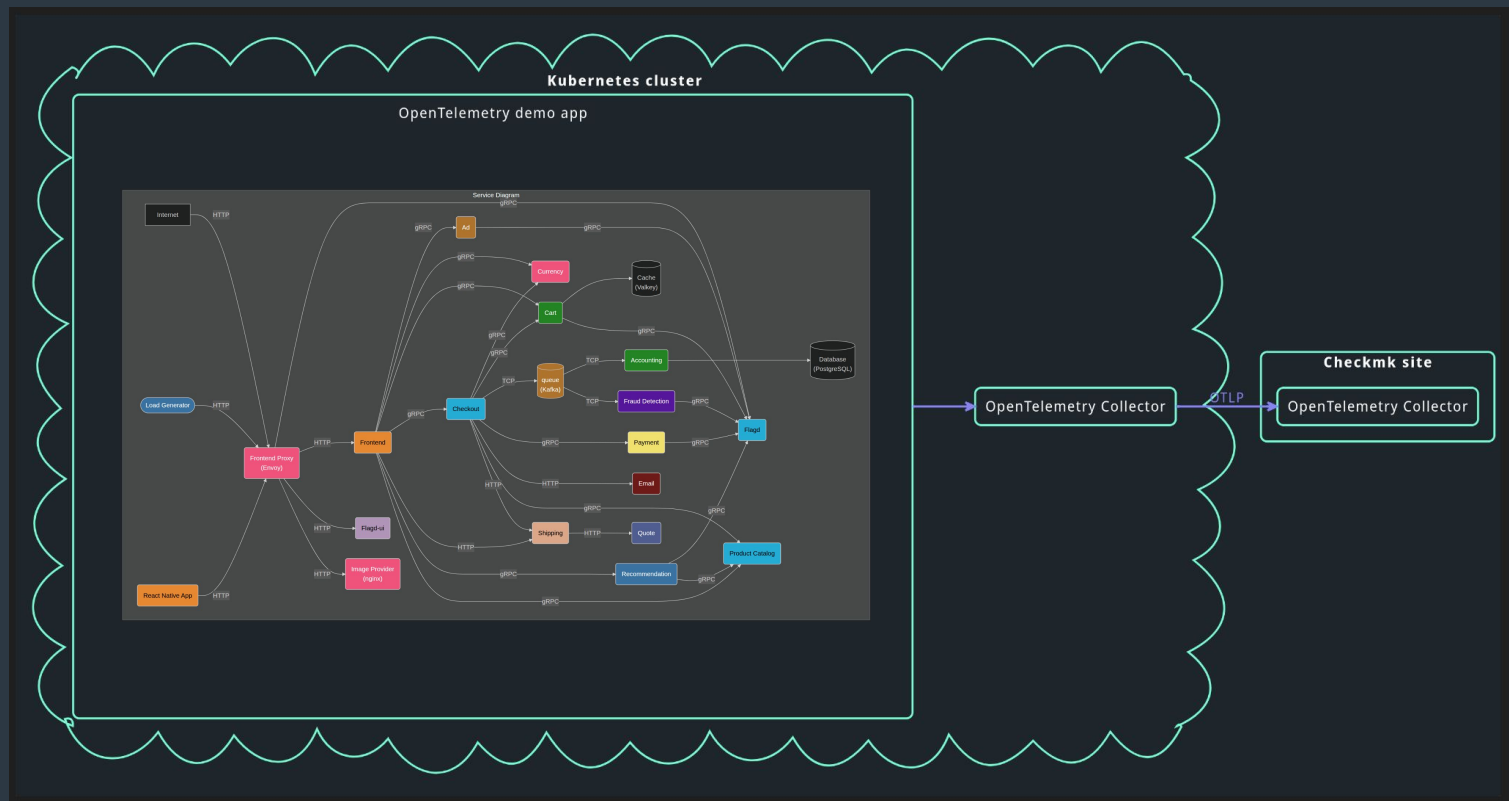


Demo time: latest improvements

Interrupt and ask whenever anything is unclear!

Today's demo setup

The OpenTelemetry demo app, deployed in a Kubernetes cluster



Start the OpenTelemetry Collector from the UI



You can now start the OpenTelemetry Collector from Checkmk's UI, and do no longer need to use OMD.

The screenshot shows the 'Edit global setting' interface in Checkmk. The left sidebar contains the Checkmk logo and navigation icons for Monitor, Customize, and Setup. The main content area is titled 'Edit global setting' with a breadcrumb 'Setup > General > Global settings > Edit global setting'. Below the title are tabs for 'Setting', 'Display', and 'Help'. Three action buttons are visible: 'Save' (with a green checkmark), 'Remove explicit setting' (with a blue double arrow), and 'Global settings' (with a blue up arrow). A yellow warning banner states: 'Changing this setting triggers a full restart of all affected sites during activate changes.' The main setting is 'Enable OpenTelemetry collector (experimental)', which is currently set to 'on'. The 'Current setting' is shown with a blue 'x' icon, the 'Factory setting' is 'on', and the 'Current state' is 'Your setting and factory settings are identical.'

checkmk

Monitor

Customize

Setup

Edit global setting

Setup > General > Global settings > Edit global setting

Setting Display Help

Save Remove explicit setting Global settings

Changing this setting triggers a full restart of all affected sites during activate changes.

▼ Enable OpenTelemetry collector (experimental)

Current setting	✖
Factory setting	on
Current state	Your setting and factory settings are identical.

Monitor the Collector's memory consumption



There is a new service named **OMD <site> OTel collector** that monitors the memory usage of the collector using the self-monitoring data.

Service OMD stable OTel collector, stable

Monitor > Overview > All hosts > stable > Services of host > Service

Commands Service Host Export Display Help ⌵

✖ Acknowledge problems 🚨 Schedule downtimes 🔍 Filter ⌵ Show checkboxes 🔗 Services of host

Site alias	The central site
Host name	stable
Service name	OMD stable OTel collector
Service labels	
Service icons	☰ 📁
Service state	<div>OK</div>
Summary	Bytes of allocated heap objects (see 'go doc runtime.MemStats.HeapAlloc') [alpha]: 12.1 MiB
Details	Bytes of allocated heap objects (see 'go doc runtime.MemStats.HeapAlloc') [alpha]: 12.1 MiB
Service Perf-O-Meter	

Limit collector memory usage

Checkmk 2.5



To ensure the OTel collector does not consume too much memory, you can now set custom memory limits.

By default, the “hard” memory limit is equal to 80% of the available memory on startup, the “soft” memory limit is set to 60%.

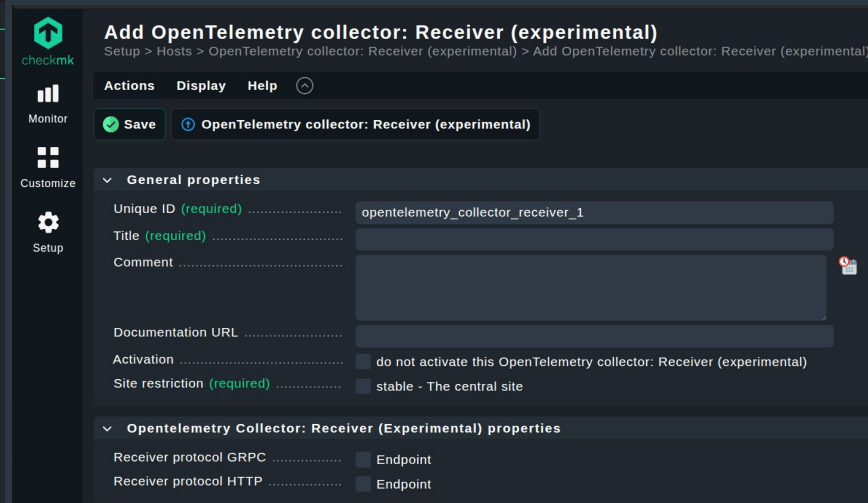
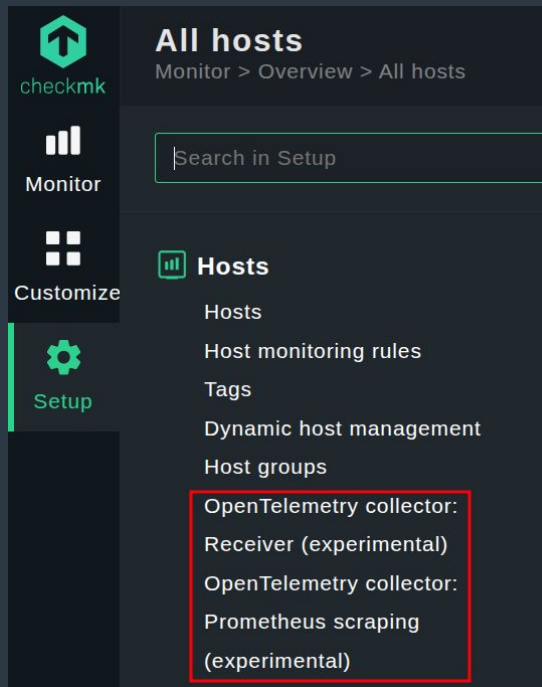
The screenshot shows the 'Memory Limiter' configuration panel in the Checkmk web interface. The panel is titled 'Memory Limiter' with a close button (X). It contains the following settings:

- Check Interval:** A text input field showing '1 s' and '0 ms'.
- Memory Limit:** A section with a dropdown menu set to 'Relative Limit in %'. Below it, there are two input fields:
 - Limit:** A text input field showing '80'.
 - Spike Limit:** A text input field showing '20'.

Split collector configuration



You now configure the OTEL receivers and the prometheus scraping using individual rule sets.



Better handling of histograms



OpenTelemetry histograms consist of three main elements:

- **Count:** The total number of measurements recorded
- **Sum:** The sum of all recorded values
- **Buckets** (optional): Predefined ranges that track how many values fall within each range

Previously, only the count value of histograms was emitted as metric.

Now, histograms get normalized to their delta between two check cycles.

Eventually, the metric is calculated as follows ...

$$\Delta \text{ sum} / \Delta \text{ count}$$

... to give a more meaningful representation of the values behind.

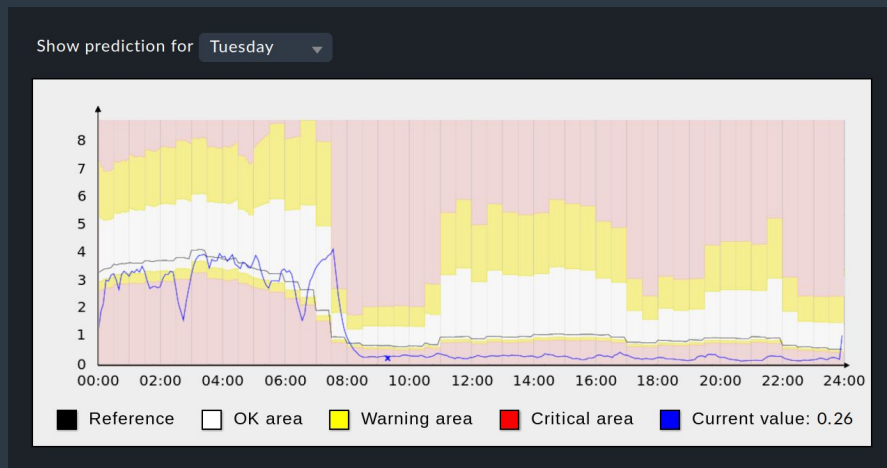
Support for predictive levels

Checkmk 2.5



You can now configure predictive levels when configuring specific OTEL metrics and use it for alerting.

Previously, predictive levels were not available for OpenTelemetry metrics.



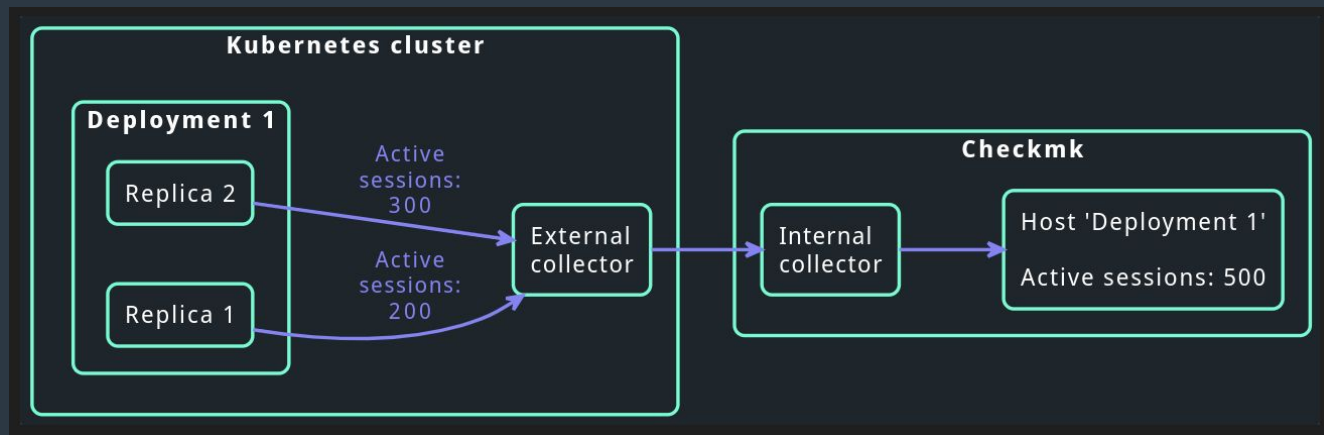
Multi-emitter metrics aggregation



You can configure how the OTel metrics emitted by multiple entities get aggregated when mapped to a single host in Checkmk.

Selectable methods are:

- Latest
- Average
- Max
- Min
- Sum



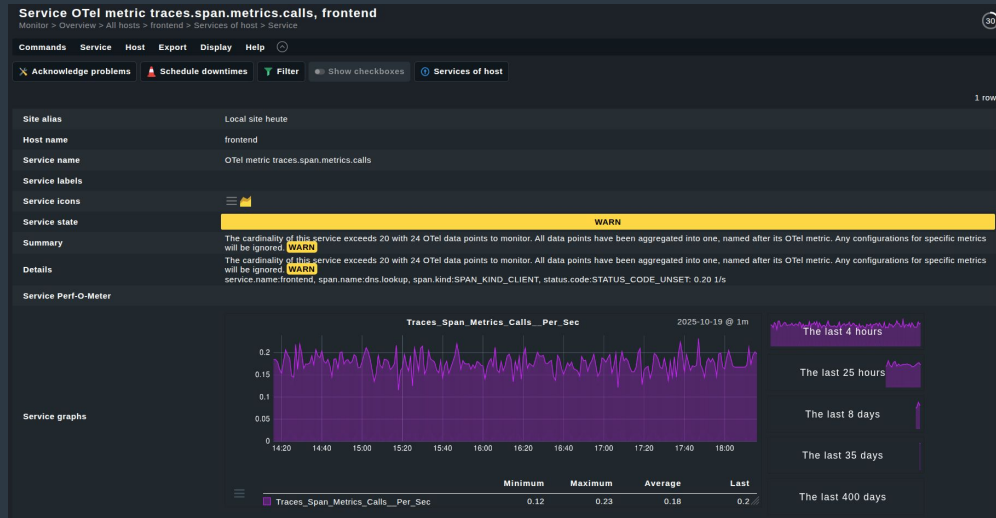
High-cardinality metrics aggregation



If an OpenTelemetry metric which is mapped to one Checkmk service has more than 20 unique combinations of datapoint attributes, the resulting time series get aggregated into one Checkmk metric.

Selectable aggregation methods are:

- Latest
- Average
- Max
- Min
- Sum

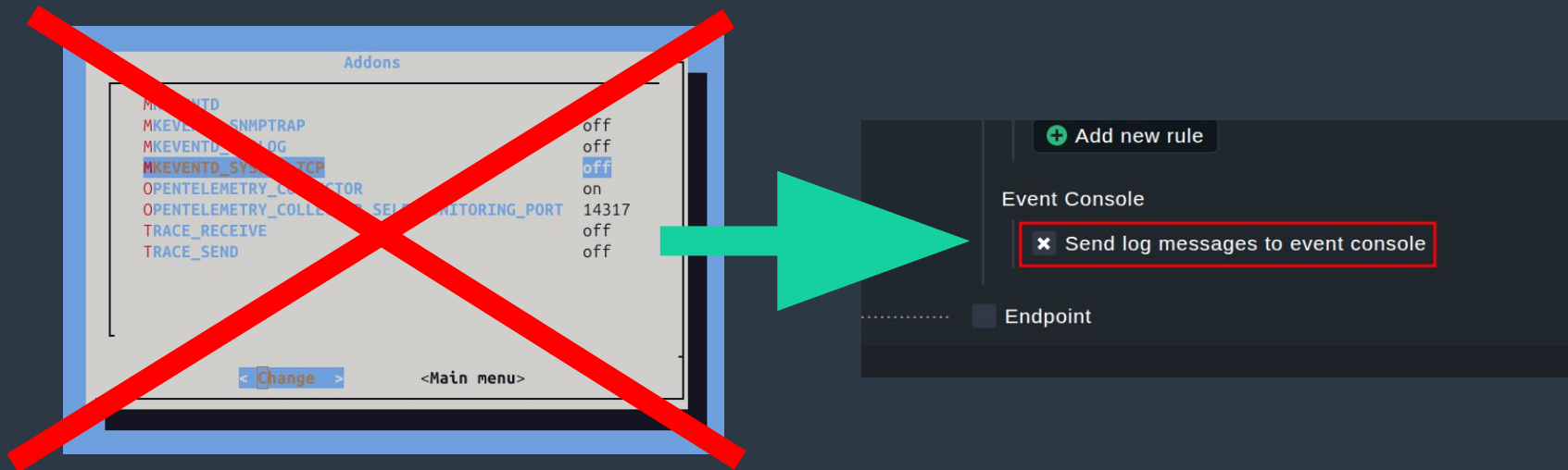


Syslog logs via Unix socket



Previously, you had to make the Event Console listen on port 514 to make it receive syslog logs converted from OpenTelemetry logs.

Now you can conveniently pass them to the Event Console via Unix socket.



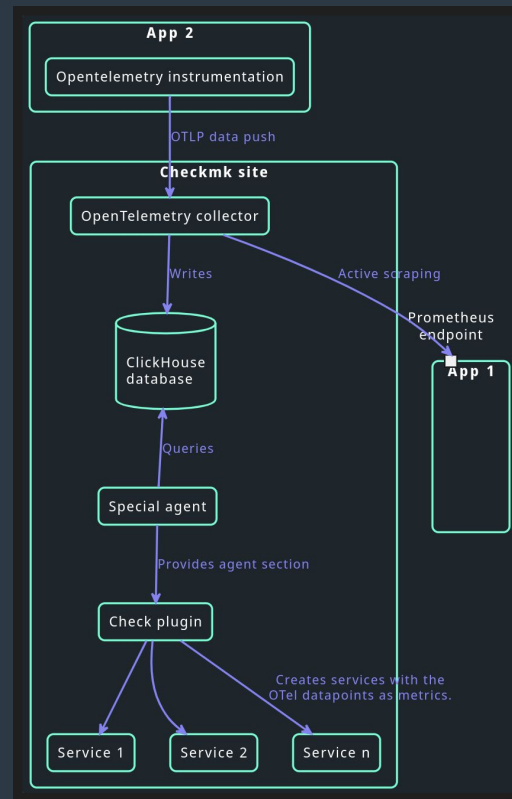
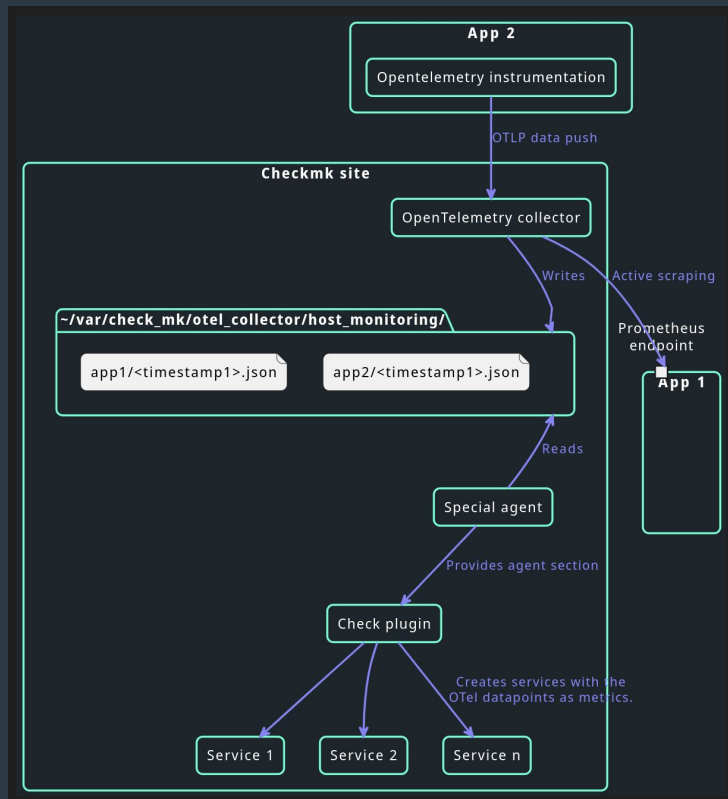
What's next?



A new metric backend

Enabling a wide range of new use cases

Checkmk 2.5



Next steps

Let's think about Checkmk 2.6

