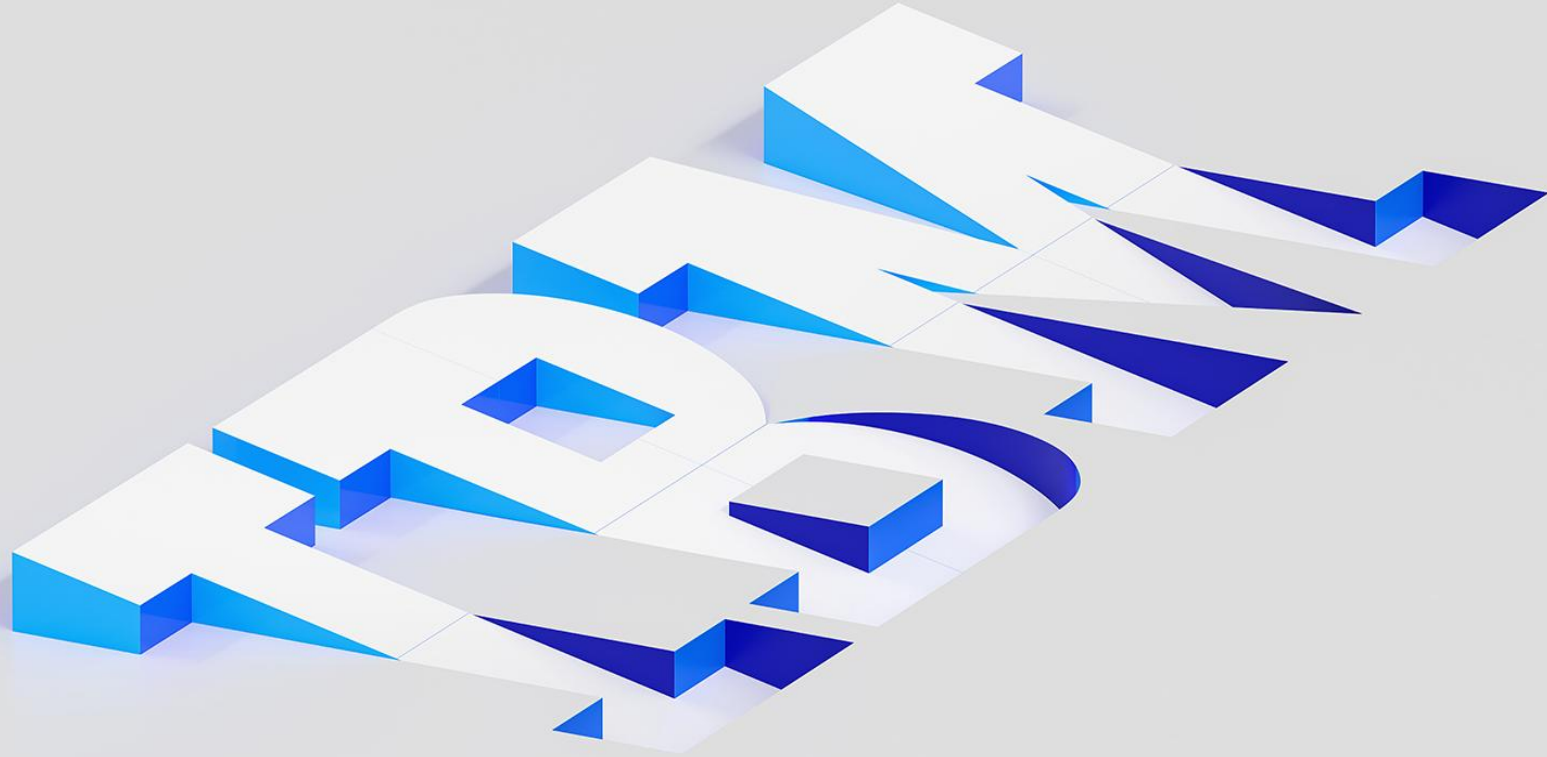


# Project Bob ahora en Sala 1



## Manzan and Friends: OSS Tools for System Management

**Jesse Gorzinski**  
Senior Business Architect, IBM

@OSSJesseG

© Copyright IBM Corporation 2025

**IBM**i

# About your speaker

- I work for IBM i Development (Rochester, MN, USA)
- Architect of Open Source and AI with IBM i
- Brags:
  - First person in the world to run Node.js on IBM i
  - First person in the world to integrate IBM i with a quantum computer
  - First person in the world to run deep learning algorithms on IBM i
  - If you plant one of my toenail clippings, a new one of me will grow



# Agenda

- Some of Manzan's friends
- Manzan overview
- Manzan + AI

# **Current State of System Management on IBM i**

# What tools are your clients using to monitor your systems?

Choose up to 3:

1. Dynatrace
2. Nagios
3. Instana
4. DataDog
5. Control4i
6. Syslog Reporting Manager (SRM)
7. Created your own
8. Other

# Is specialty a collective disadvantage?

1. Dynatrace
2. Nagios
3. Instana
4. DataDog
5. Control4i
6. Syslog Reporting Manager (SRM)
7. Created your own
8. Other

Each solutions may have their own...

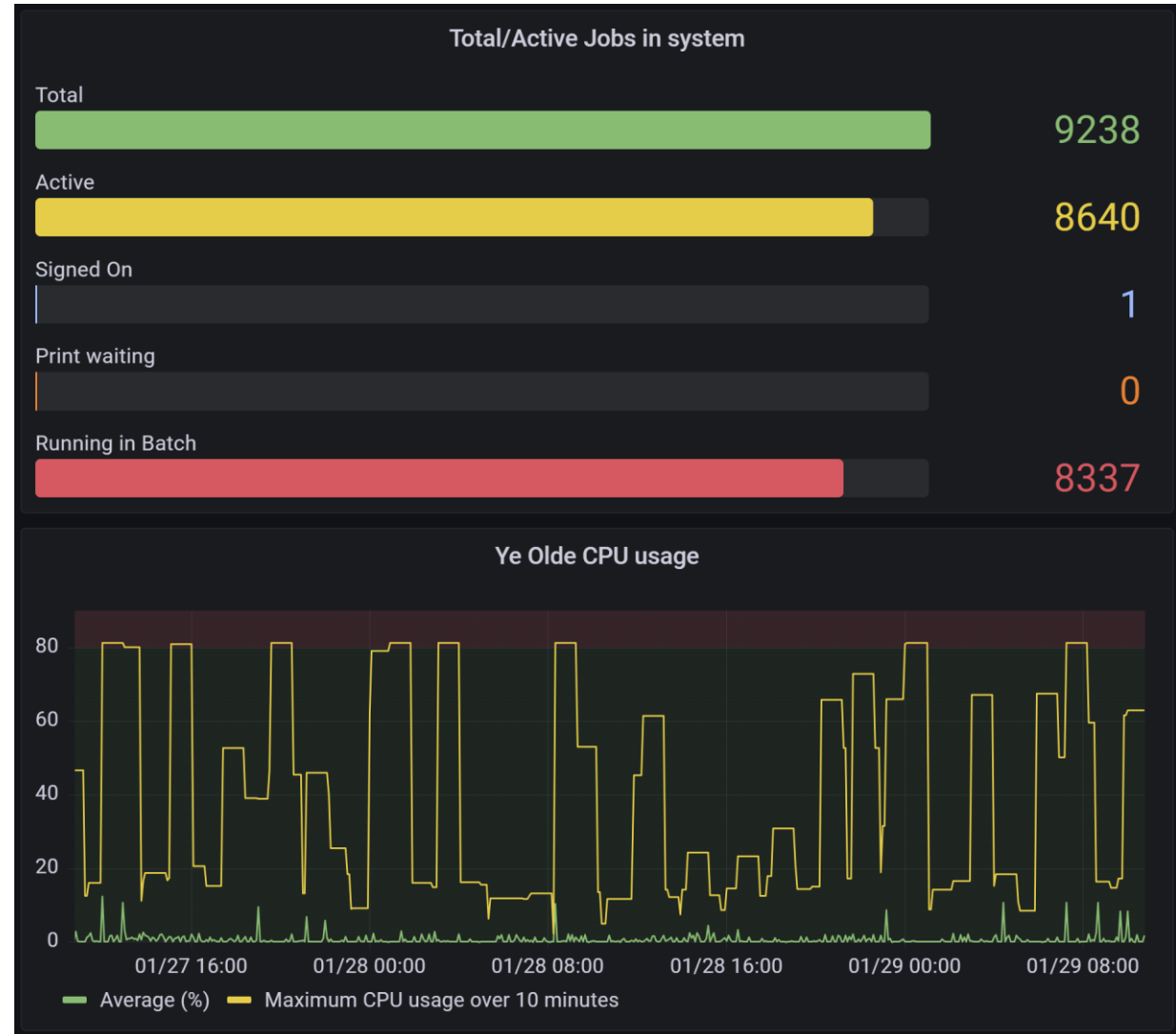
- Configuration
- Host installation requirements
- Monitoring capabilities
  - Collect system metrics (active jobs, ASP consumption)
  - View sub system information
  - Identify long-running SQL
  - View job queue

Can you integrate with other tools?

# Do you use Grafana?

Choose 1:

1. Yes, with IBM i
2. Yes, but not with IBM i
3. No, but we want to
4. No, we don't want to
5. No, don't know what it is



# **Operational Monitoring with Prometheus**



# Prometheus Overview

## What is it?

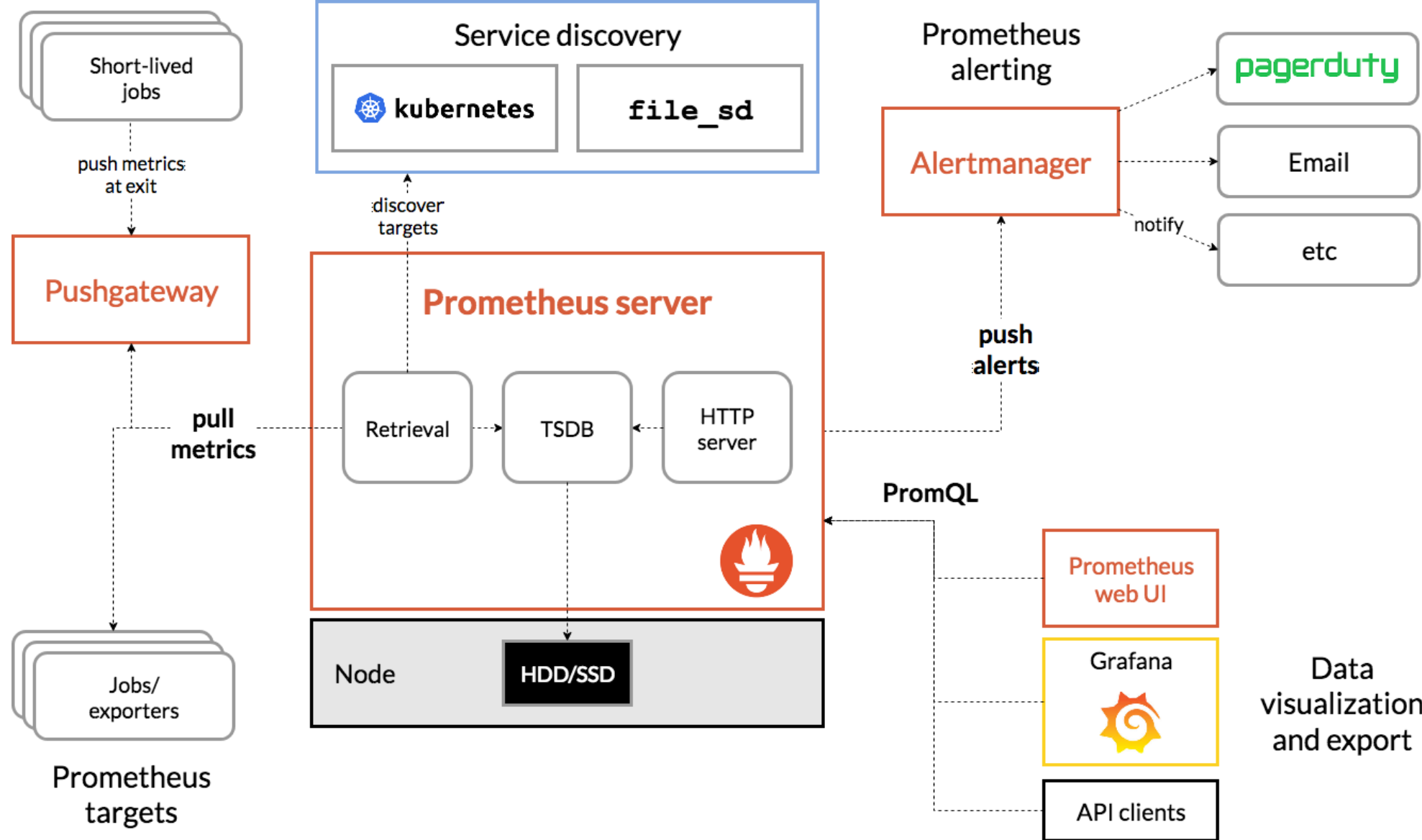
- Leading open-source systems monitoring and alerting toolkit
- Collects and stores metrics as timeseries data

## Features

- Multi-dimensional data model with time series data
- Provides a functional query language called PromQL
- No reliance on distributed storage
- Has an alert manager built-in
- Easily paired with Grafana and other monitoring solutions



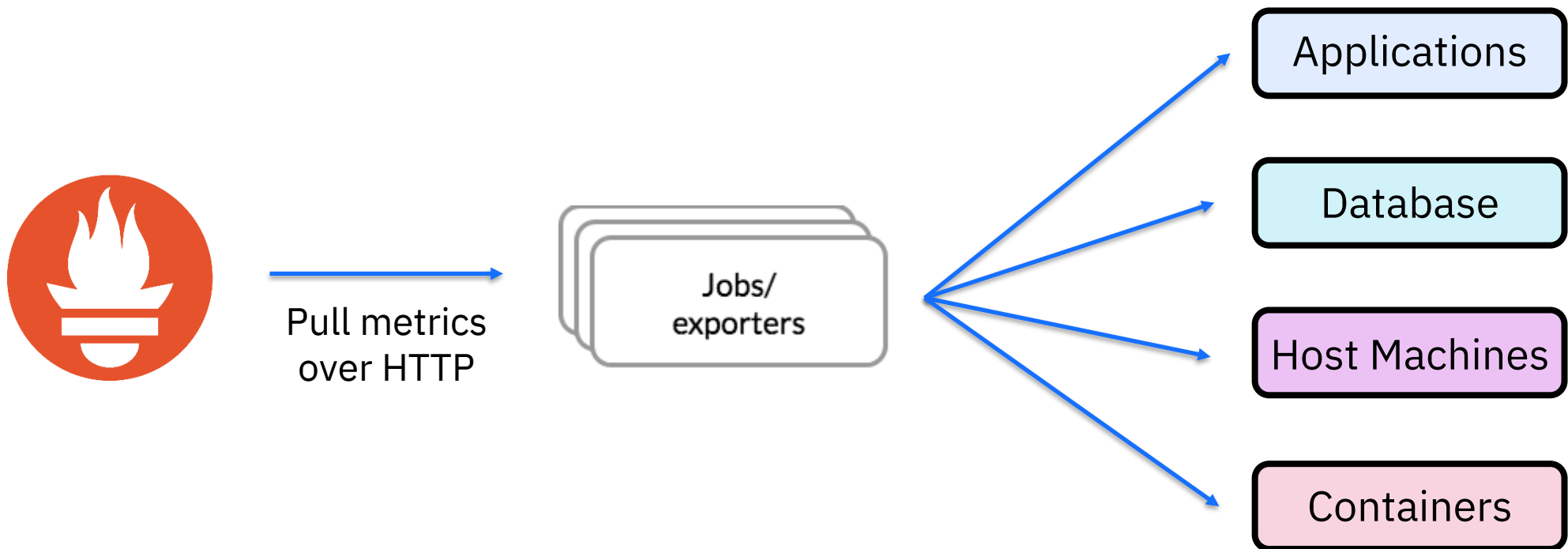
# Prometheus Architecture



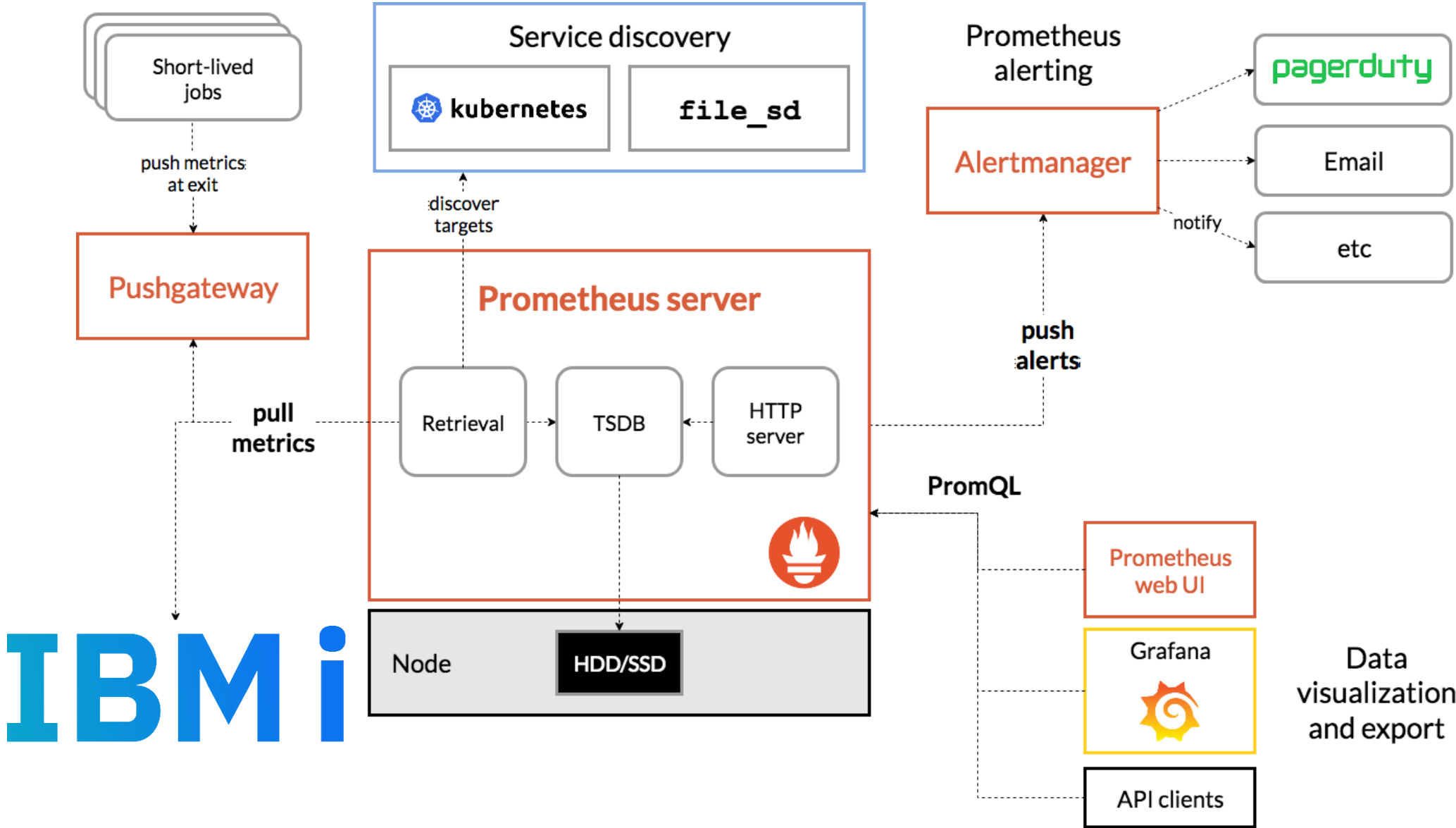
# What do exporters do?

Collection of exporters: <https://prometheus.io/docs/instrumenting/exporters/>

1. Fetch metrics
2. Convert them to Prometheus format
3. Expose data via an HTTP endpoint for it to be scraped



# How to use Prometheus with IBM i



# IBM i Prometheus Exporter

- JDBC-based exporter that leverages SQL to export IBM i metrics
- Exports hundreds of metrics including:
  - System activity levels (such as CPU usage)
  - Memory pool information
  - Apache HTTP server metrics
  - Number of remote TCP connections
- User customizable metrics with SQL

<https://github.com/ThePrez/prometheus-exporter-jdbc>

```
config.json X
home > SANJULA > jar > prometheus > {} config.json > ...
1  {
2    "port": 9853,
3
4    "queries": [{
5      "name": "System Statistics",
6      "interval": 60,
7      "enabled": true,
8      "prefix": "STATS",
9      "sql": "SELECT * FROM TABLE(QSYS2.SYSTEM_STATUS(RESET_STATISTICS=>'YES',DETAILED_INFO=>'ALL')) X"
10     },
11     {
12       "name": "System Activity",
13       "interval": 20,
14       "prefix": "SYSACT",
15       "enabled": true,
16       "sql": "SELECT * FROM TABLE(QSYS2.SYSTEM_ACTIVITY_INFO())"
17     },
18     {
19       "name": "number of remote connections",
20       "interval": 30,
21       "enabled": true,
22       "sql": "select COUNT(REMOTE_ADDRESS) as REMOTE_CONNECTIONS from qsys2.netstat_info where TCP_STATE = 'ESTABLISHED'
                AND REMOTE_ADDRESS != ':::1' AND REMOTE_ADDRESS != '127.0.0.1'"
23     },
24     {
25       "name": "Memory Pool Info",
26       "interval": 100,
27       "enabled": true,
28       "multi_row": true,
29       "prefix": "MEMPOOL",
30       "sql": "SELECT POOL_NAME,CURRENT_SIZE,DEFINED_SIZE,MAXIMUM_ACTIVE_THREADS,CURRENT_THREADS,RESERVED_SIZE FROM TABLE
                (QSYS2.MEMORY_POOL(RESET_STATISTICS=>'YES')) X"
31     }
32   ]
33 }
```

# Deploying the IBM i Prometheus Exporter

**Video Demo:** <https://www.youtube.com/watch?v=qIcAx4lJZIA>

1. Download latest .jar file from release page and store in the IFS:

```
wget https://github.com/ThePrez/prometheus-exporter-jdbc/releases/latest/download/prom-client-ibmi.jar
```

2. If this is your first time, create a default configuration file ([config.json](#)):

```
java -jar prom-client-ibmi.jar
```

3. Respond with **y** for the following message:

```
Configuration file config.json not found. Would you like to initialize one with defaults? [y]
```

4. Confirm the Prometheus client has started:

```
=====
Successfully started Prometheus client on port 9853
=====
```

5. Create a service commander configuration file ([prometheus.yml](#)) and start it:

```
java -jar prom-client-ibmi.jar sc
sc start prometheus.yml
```

# Deploying the Prometheus Instance

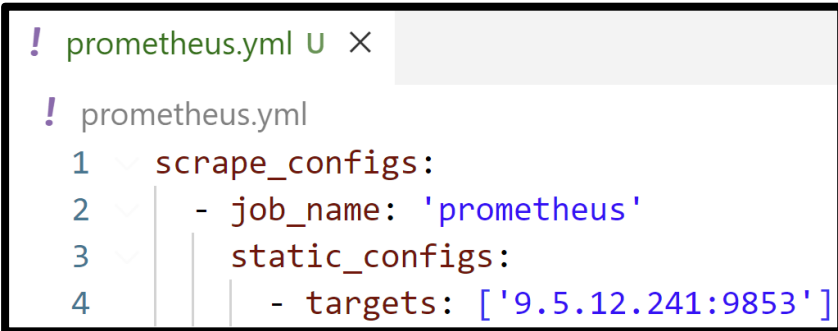
1. Download the Prometheus-provided image from Docker Hub to some central location:

```
docker pull prom/prometheus
```

or

```
podman pull docker.io/prom/prometheus
```

2. Create a Prometheus configuration file ([prometheus.yml](#)):

A screenshot of a code editor window titled 'prometheus.yml'. The editor shows a YAML configuration for Prometheus. The content is as follows:

```
! prometheus.yml
! prometheus.yml
1 scrape_configs:
2   - job_name: 'prometheus'
3     static_configs:
4       - targets: ['9.5.12.241:9853']
```

3. Launch the image in a container named “[prometheus](#)”:

```
docker run -d --name=prometheus -v /path/to/prometheus.yml:/etc/prometheus/prometheus.yml -p 9090:9090
prom/prometheus
```

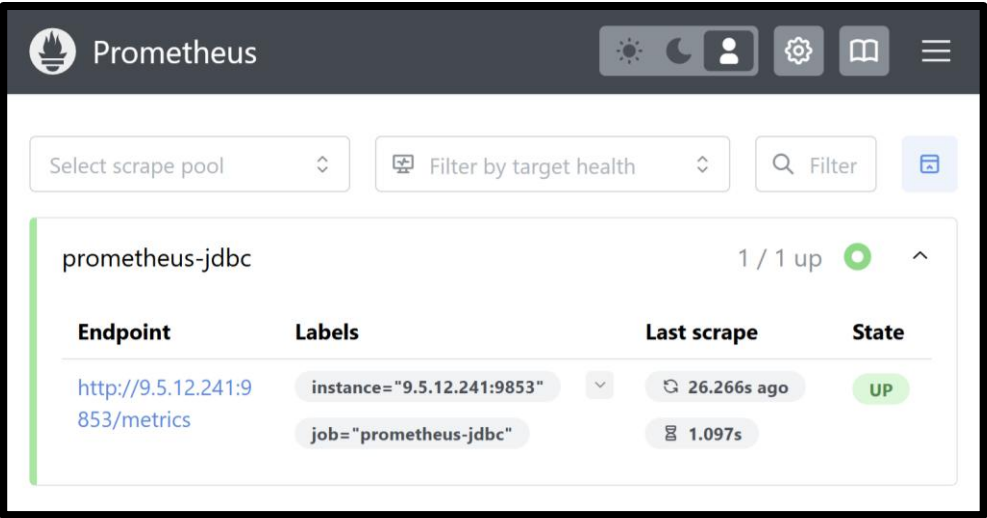
or

```
podman run -d --name=prometheus -v /path/to/prometheus.yml:/etc/prometheus/prometheus.yml -p 9090:9090
prom/prometheus
```

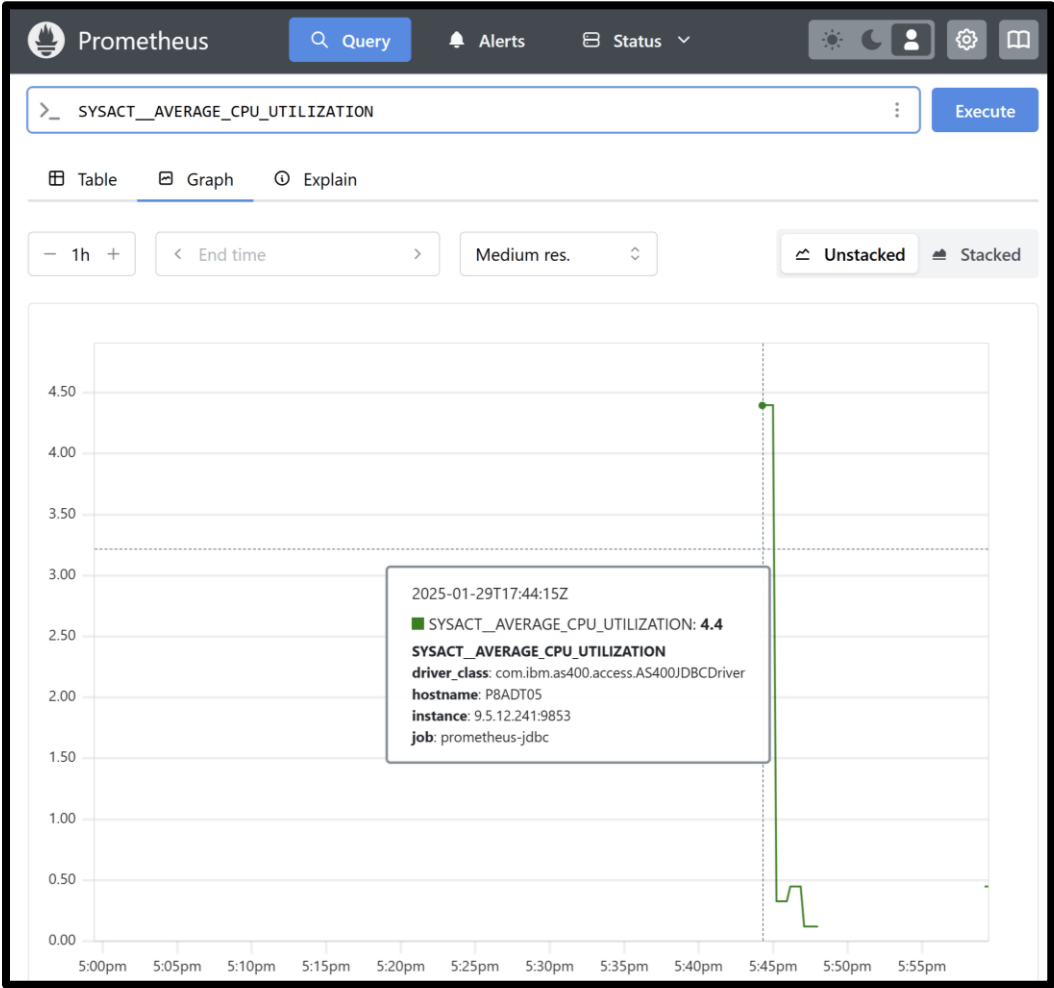
4. Navigate to Prometheus web UI at [http://<container\\_ip>:9090](http://<container_ip>:9090)

# Prometheus Web UI

## Target System Status



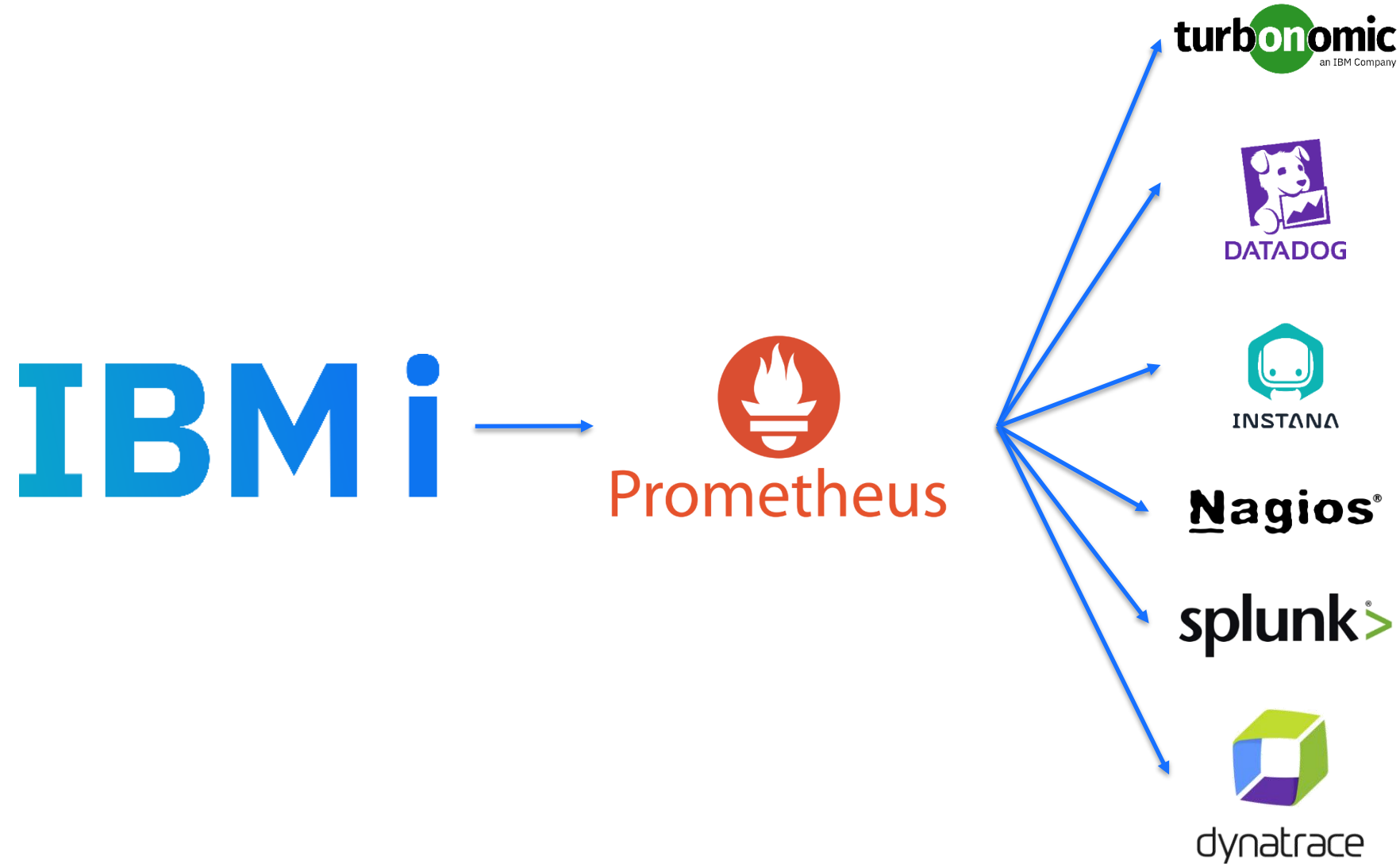
## Graphing Historical Metric Data



Blog post by Jesse Gorzinski: "Monitoring IBM i with Prometheus"  
<https://techchannel.com/open-source-on-ibm-i/monitoring-ibm-i-with-prometheus/>















# Prometheus can be the bridge to other solutions



# **Data Visualization with Grafana**


# Grafana Overview

- Open-source analytics, visualization, and monitoring solution
- Driven by Grafana Labs. Plenty of open-source components. See <https://grafana.com/>
- Over 300 plugins available


 Grafana Labs			Products	Open source	Solutions	Learn	Company
			<b>Grafana Loki</b> Multi-tenant log aggregation system				<b>Grafana k6</b> Load testing for engineering teams
			<b>Grafana</b> Query, visualize, and alert on data				 <b>Prometheus</b> Monitor Kubernetes and cloud native
			<b>Grafana Tempo</b> High-scale distributed tracing backend				 <b>OpenTelemetry</b> Instrument and collect telemetry data
			<b>Grafana Mimir</b> Scalable and performant metrics backend				 <b>Graphite</b> Scalable monitoring for time series data

# Grafana Data Sources


## Time series databases




**Prometheus**  
Open source time series database & alerting  
[Core](#)



**Graphite**  
Open source time series database  
[Core](#)




**InfluxDB**  
Open source time series database  
[Core](#)



**OpenTSDB**  
Open source time series database  
[Core](#)


## SQL



**MySQL**  
Data source for MySQL databases  
[Core](#)




**Microsoft SQL Server**  
Data source for Microsoft SQL Server compatible databases  
[Core](#)




**PostgreSQL**  
Data source for PostgreSQL and compatible databases  
[Core](#)

## Cloud




**Azure Monitor**  
Data source for Microsoft Azure Monitor & Application Insights  
[Core](#)




**CloudWatch**  
Data source for Amazon AWS monitoring service  
[Core](#)

## Logging & document databases

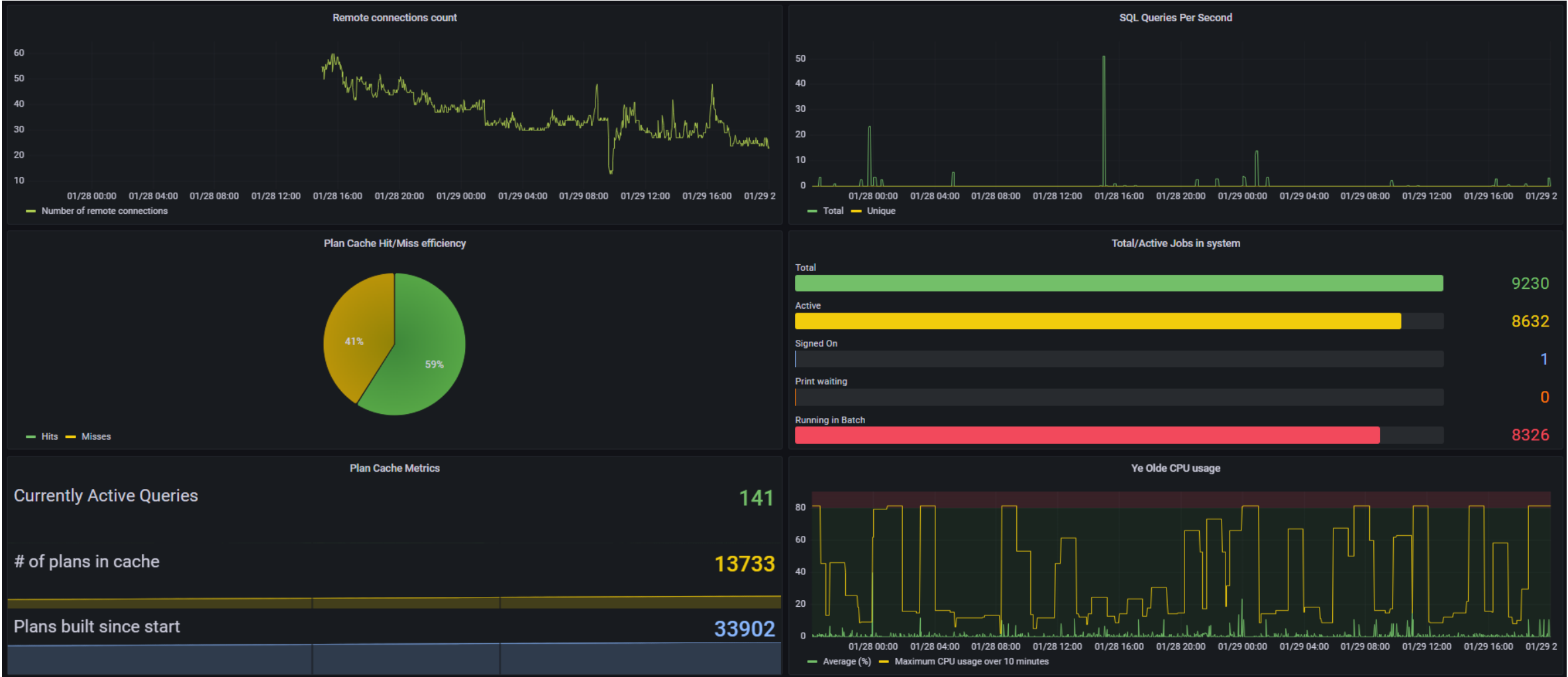


**Loki**  
Like Prometheus but for logs. OSS logging solution from Grafana Labs  
[Core](#)



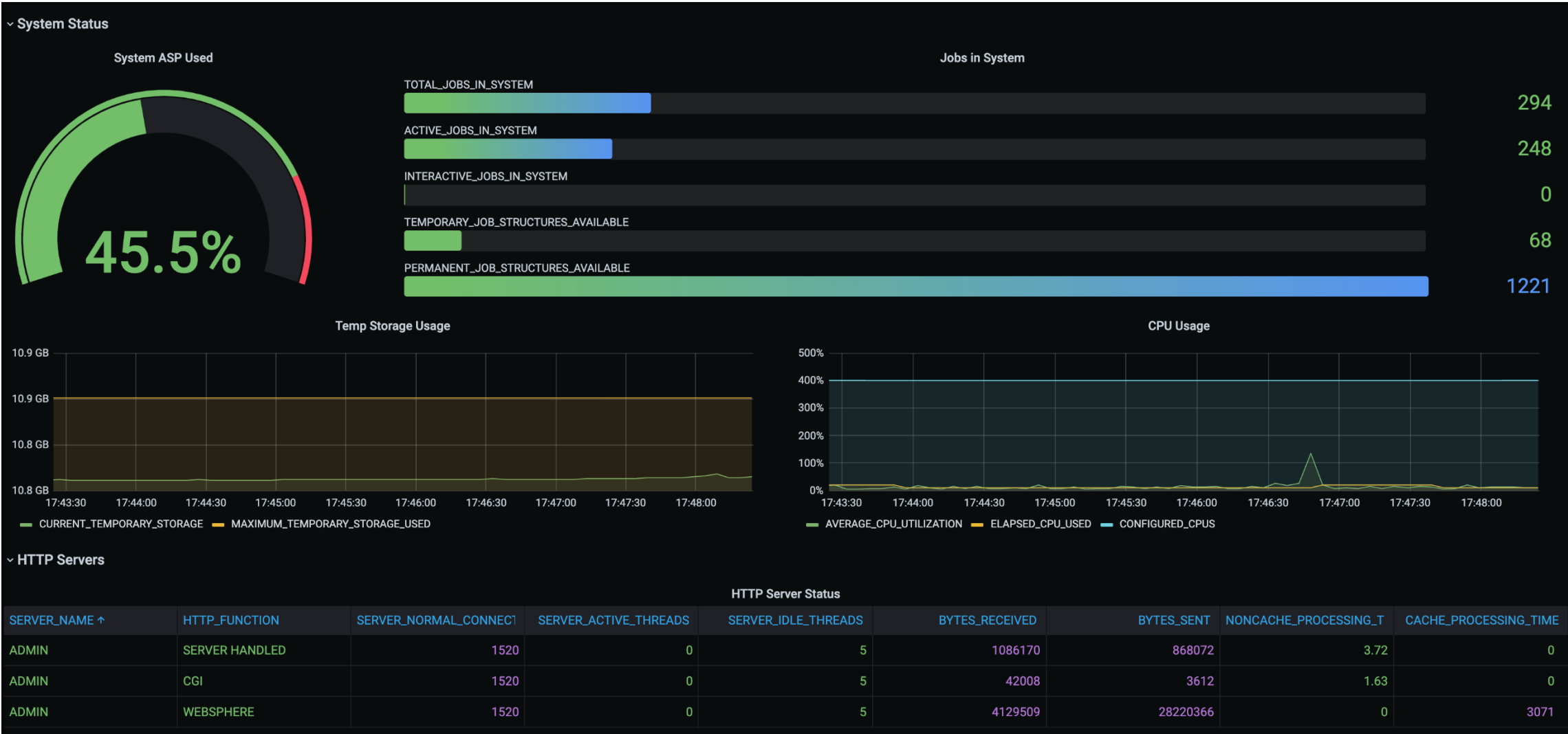
**Elasticsearch**  
Open source logging & analytics database  
[Core](#)

# Prometheus Visualization with Grafana



Public COMMON1 Grafana Dashboard: <http://ibm.biz/ibmi-prometheus>

# Grafana Backend (Without Prometheus)



Blog post by Jesse Gorzinski: "How to Deploy a Simple Grafana Dashboard"  
<https://techchannel.com/performance/how-to-deploy-a-simple-grafana-dashboard/>

# Grafana with or without Prometheus

	With Prometheus	Straight to Grafana
Persistent storage	Prometheus	Grafana
Persistent storage of unused metrics	Prometheus	--
Metric type	Numerics	Numerics/strings/other
Ecosystem	Extremely Broad	There
Scalability	Excellent	Good
IBM i requisites	None	Node.js
Initial setup	Easier	Easy

# **Event Monitoring with Manzan**

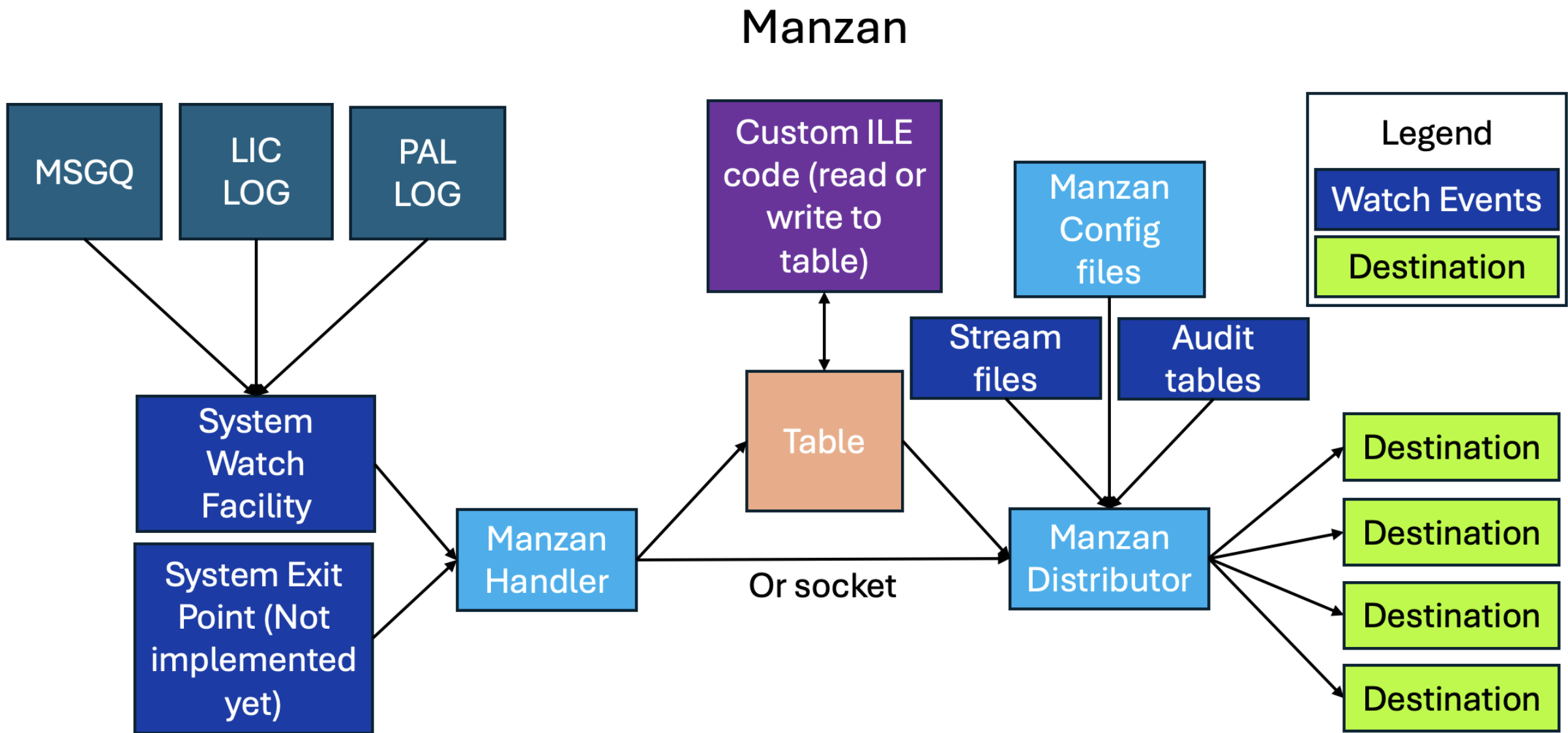


# Manzan Overview

- Open-source event handling tool designed to simplify handling of system events
- Serves as a gateway for publishing IBM i events to a variety of endpoints:
  - User applications
  - External resources
  - Open-source technologies
- Example use cases:
  - Monitor system events with a third-party open source or proprietary tool
  - Consolidate logs for auditing/reporting + query for system events
  - Feed real-time system event data into stream processing systems (such as Apache Kafka)
  - Send alerts for system events

Full Documentation: <https://theprez.github.io/Manzan/#/>

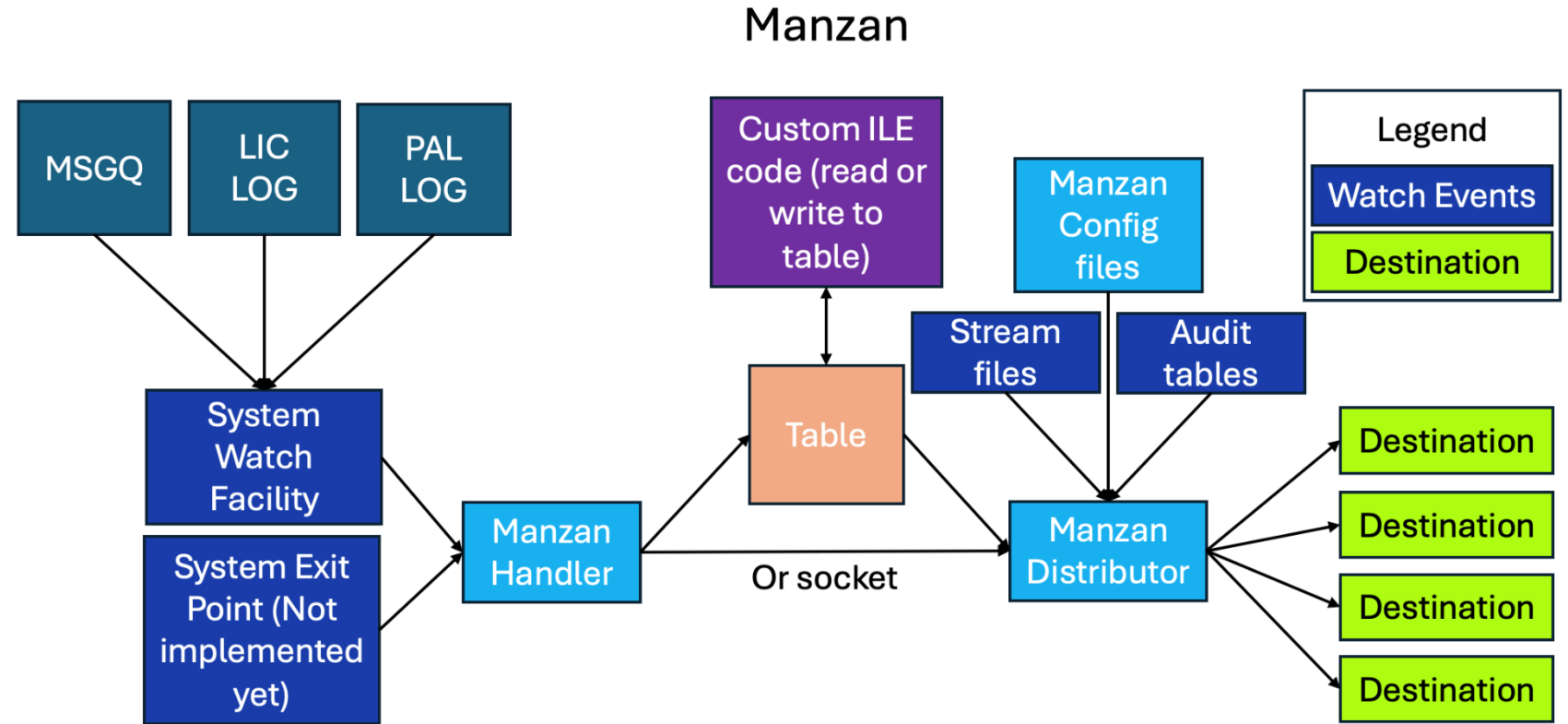
# Understanding the Architecture



# What are inputs?

## Watch events: Sources of your data

- Stream file
- System watch facility (STRWCH)
  - MSGQ
  - LIC logs
  - PAL logs
- SQL Tables
- Audit journals
- System exit points (*coming soon*)



# So practically what can Manzan monitor?

Application  
crashes

Log data

Custom SQL

System  
Limits alerts

History Log  
entries

Problem log  
entries

\*SYSOPR  
messages

Specific job  
log messages

Audit journal  
events

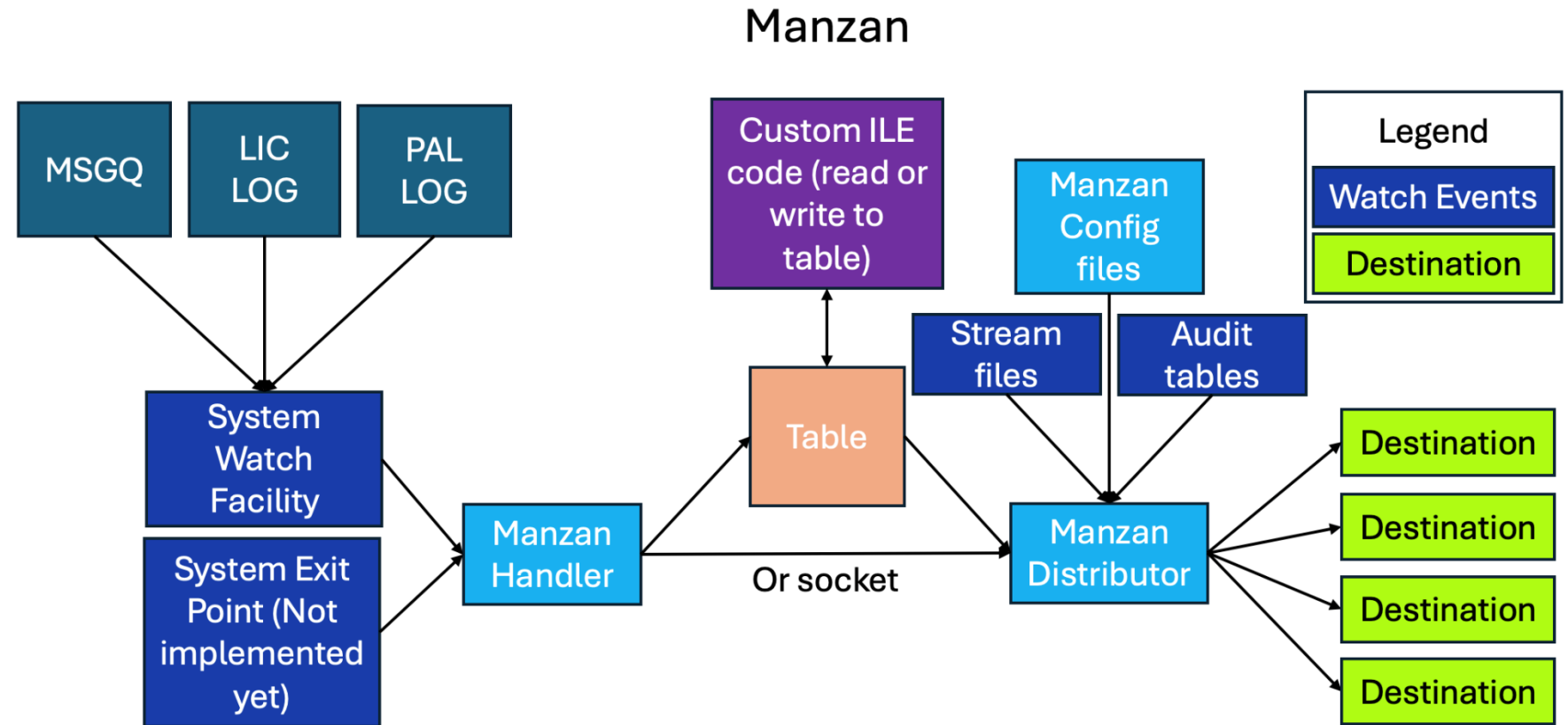
PAL entries

VLOGs

Fishy TCP  
connections  
(future)

## Destinations: Locations to send data

- Built-in supported destinations:
  - HTTP/HTTPS endpoints (REST, etc.)
  - Email (SMTP/SMTPS)
  - SMS (via Twilio)
  - Slack
  - FluentD
  - Kafka
  - Sentry
  - Grafana Loki
  - Google Pub/Sub
  - ActiveMQ
- Custom ILE Code



# Can Manzan send data anywhere?

- Many destinations are already working, but there are more to come
- Track supported destinations:  
<https://theprez.github.io/Manzan/#/?id=where-can-i-send-these-events>
- Desired target not on the list? Please open an issue to the repository and let us know!

- [ActiveMQ](#) ✓
- [AWS Simple Email Service \(SES\)](#) ⌚
- [AWS Simple Notification System \(SNS\)](#) ⌚
- [Elasticsearch](#) ✓
- Email (SMTP/SMTPS) ✓
- [FluentD](#) ✓
- [Google Drive](#) ⌚
- [Google Pub/Sub](#) ✓
- [Grafana Loki](#) ✓
- HTTP endpoints (REST, etc) ✓
- HTTPS endpoints (REST, etc) ✓
- [Internet of Things \(mqtt\)](#) ⌚
- [Kafka](#) ✓
- [Mezmo](#) ✓
- [Microsoft Teams](#) ⌚
- [PagerDuty](#) ✓
- [Sentry](#) ✓
- [Slack](#) ✓
- SMS (via [Twilio](#)) ✓
- [Splunk](#) ✓

✓ = implemented    🟡 = partially implemented    ⌚ = future

# Understanding the Architecture: Handler and Distributor

## Handler

- Start system watches based on config files ([data.ini](#))
- Transform system event data into structured format
- Add structured data to table or send over socket

## Distributor

- Fetch data off table or via socket
- Send data to destinations using Apache Camel based on config files ([dests.ini](#))

## Data From Message Queue

```
{
  "EVENT_TYPE" : "message",
  "SESSION_ID" : "SANJULA",
  "JOB" : "286747/QSECOFR/QP0ZSPWP",
  "MESSAGE_ID" : "*IMMED",
  "MESSAGE_TYPE" : "*INFO",
  "SEVERITY" : 80,
  "MESSAGE_TIMESTAMP" : "2025-01-29 22:17:23.199",
  "SENDING_USRPRF" : "SANJULA",
  "MESSAGE" : "Hello World!",
  "SENDING_PROGRAM_NAME" : "QCAPCMD",
  "SENDING_MODULE_NAME" : "      ",
  "SENDING_PROCEDURE_NAME" : ""
}
```

## Data From Stream File

```
{
  "FILE_NAME" : "my_app_logs.txt",
  "FILE_PATH" : "/home/SANJULA/my_app_logs.txt",
  "FILE_DATA" : "[ERROR] Application crashed due to memory leak"
}
```

# Configuring Inputs and Destinations

Configuration files are stored in [/QOpenSys/etc/manzan](#)

## app.ini

Used for general  
Manzan configuration  
(leave the default contents)

```
≡ app.ini  ×
config > ≡ app.ini
1  [install]
2  library=MANZAN
```

## data.ini

Used for configuring  
different inputs

```
≡ data.ini  ×
≡ data.ini
1  [<id>]
2  # The type of input (file or watch):
3  type=<type>
4  # As defined in dests.ini:
5  destinations=<destinations>
6  # Optional properties:
7  format=<format>
8  enabled=<enabled>
9  # Other <type> specific properties
10 # (ie. file, strwch)...
```

## dests.ini

Used for configuring  
different destinations

```
≡ dests.ini  ×
≡ dests.ini
1  [<id>]
2  # The type of destination
3  # (ie. smtp, slack, twilio, sentry)...
4  type=<type>
5  # Optional properties:
6  format=<format>
7  # Other <type> specific properties
8  # (ie. server, webhook, from, dsn)...
```



# Simple data.ini configurations

## Stream File

- Watch application log file ([test.txt](#)) and only take action when a line written to the file contains the string “[error](#)”

```
[logfile1]
type=file
file=test.txt
destinations=email_it, test_out
filter=error
format=$FILE_DATA$
```

## System Watch

- Manage information from watch session with id “[jesse](#)”

```
[watcher1]
type=watch
id=jesse
destinations=test_out, slackme
enabled=false
```

# Advanced data.ini configuration

## System Watch

- Manage information from watch session with id “jesse” which watches all messages in the IBM i history log
- Format the message into a human-sensible format

```
[watchout]
type=watch
id=jesse
destinations=test_out, slackme
format=$MESSAGE_ID$ (severity $SEVERITY$): $MESSAGE$
strwch=WCHMSG((*ALL)) WCHMSGQ((*HSTLOG))
```

# Example dests.ini configurations

Send to Manzan's standard output

```
[test_out]  
type=stdout
```

Send an email

```
[email_it]  
type=smtps  
format=Hey, check out this information!! \n\n$FILE_DATA$  
server = my.smtpserver.com  
subject = Testemail  
from=me@mycompany.com  
to=me@mycompany.com
```

# More example dests.ini configurations

Send message in Slack channel:

```
[slackme]  
type=slack  
channel=open-source-system-status  
webhook=https://hooks.slack.com/services/TA3EF58G4...
```

Send event to Sentry

```
[sentry_out]  
type=sentry  
dsn=https://k23de06d7bff24f18a86f33178e67291@o1963...
```

Send SMS message with Twilio

```
[twilio_sms]  
type=twilio  
componentOptions.sid=AC1234567890abcdef1234567890a...  
componentOptions.token=8aebaa92e9a4f6d85cf5c0e6ba3...  
to=+12345678901  
from=+10987654321
```

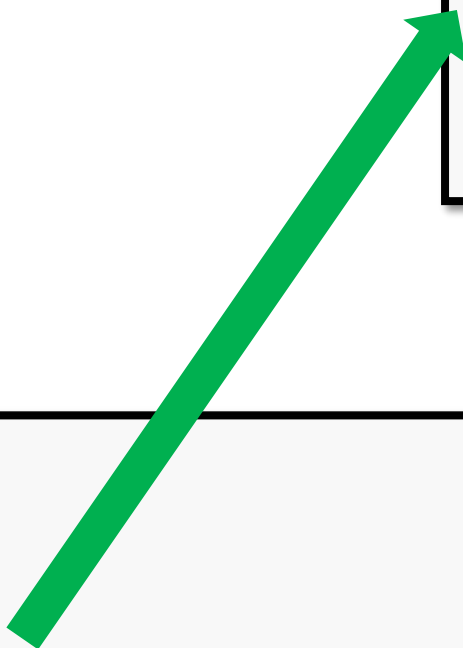
# Let's put it together...

dests.ini

```
[slackme]  
type=slack  
channel=open-source-system-status  
webhook=https://hooks.slack.com/services/TA3EF58G4...
```

data.ini

```
[watchout]  
type=watch  
id=jesse  
destinations=slackme  
format=$MESSAGE_ID$ (severity $SEVERITY$): $MESSAGE$  
strwch=WCHMSGQ(*ALL) WCHMSGQ(*HSTLOG)
```



# What does it look like in Slack?

# open-source-system-status – Dec 20th, 2022



**ibm i system monitor** APP 9:01 PM

CPIAD09 (severity 0): User LINUX from client 9.163.40.192 connected to job 492988/QUSER/QZDASOINIT in subsystem QUSRWRK in QSYS on 12/20/22 20:17:06.

# More examples...Sentry!

IBM  
Liam

Projects

Issues

Performance

Profiling beta

Releases

Crons beta

User Feedback

Alerts

Discover

Dashboards

5

Quick Start

5 Remaining tasks

Upgrade Now

Help

What's new

Collapse

Issues

All Unresolved 12 For Review Ignored

manzan All Envs 14D is:unresolved

Resolve

Ignore

Last Seen

GRAPH: 24h 14d EVENTS USERS ASSIGNEE

MCH3601: Pointer not set for location referenced.Cause . . . . : A pointer v  
24946  
MANZAN-C 1wk ago | 2wk old

385

1

MCH4421: Invalid operation for program.Cause . . . . : The operation requ  
26669  
MANZAN-7 1wk ago | 2wk old

47

1

MCH1401: Receiving context contains duplicate object.Cause . . . . : An at  
25195  
MANZAN-5 1wk ago | 2wk old

233

2

MCH3601: Pointer not set for location referenced.Cause . . . . : A pointe  
26742  
MANZAN-G 1wk ago | 1wk old

1

1

MCH3401: Cannot resolve to object QTMSTRCSV. Type and Subtype X'19  
26987  
MANZAN-A 1wk ago | 2wk old

29

1

MCH5801: Dequeue operation not satisfied in 15 seconds for queue &1C

39

© Copyright IBM Corporation 2024

# More examples...Sentry!

IBM  
Liam

Projects

Issues

Performance

Profiling beta

Releases

Crons beta

User Feedback

Alerts

Discover

Dashboards

5

Quick Start

5 Remaining tasks

⚡

Upgrade Now

?

Help

🗨

What's new

< Collapse

sentry.io

FormattedRaw

Additional Data

HANDLED_TIMESTAMP	2022-12-12T17:21:17.996Z
JOB	293151/QUSER/QZDAS0INIT
MESSAGE	Invalid operation for program.Cause . . . . . : The operation requested is not supported for program &1. The system supports distinct "models" of program objects. Not all operations are supported for all models of programs. For example, some program objects cannot be deactivated.
MESSAGE_ID	MCH4421
MESSAGE_TIMESTAMP	2022-12-12T17:21:17.936Z
MESSAGE_TYPE	*ESCAPE
ORDINAL_POSITION	25197
SENDING_MODULE_NAME	QSQRINZ
SENDING_PROCEDURE_NAME	QSQRINZ
SENDING_PROGRAM_NAME	QSQRINZ
SENDING_USRPRF	QUSER
SESSION_ID	JESSE
SEVERITY	40



# More examples...Google Pub/Sub!

my-sub - Pub/Sub - my-project

console.cloud.google.com/cloudpubsub/subscription/detail/my-sub?project=my-project-438217&tab=messages&pli=1&inv=AbeaQg

Google Cloud my-project Search (/) for resources, docs, products, and more Search

Pub/Sub

Topics

Subscriptions

Snapshots

Schemas

Pub/Sub Lite

Lite Reservations

Lite Topics

Lite Subscriptions

Release Notes

my-sub

EDIT

CREATE SNAPSHOT

REPLAY MESSAGES

PURGE MESSAGES

DETACH

DELETE

LEARN

SHOW INFO PANEL

Subscription name

projects/my-project-438217/subscriptions/my-sub

Subscription state

active

Topic name

projects/my-project-438217/topics/my-topic

METRICS

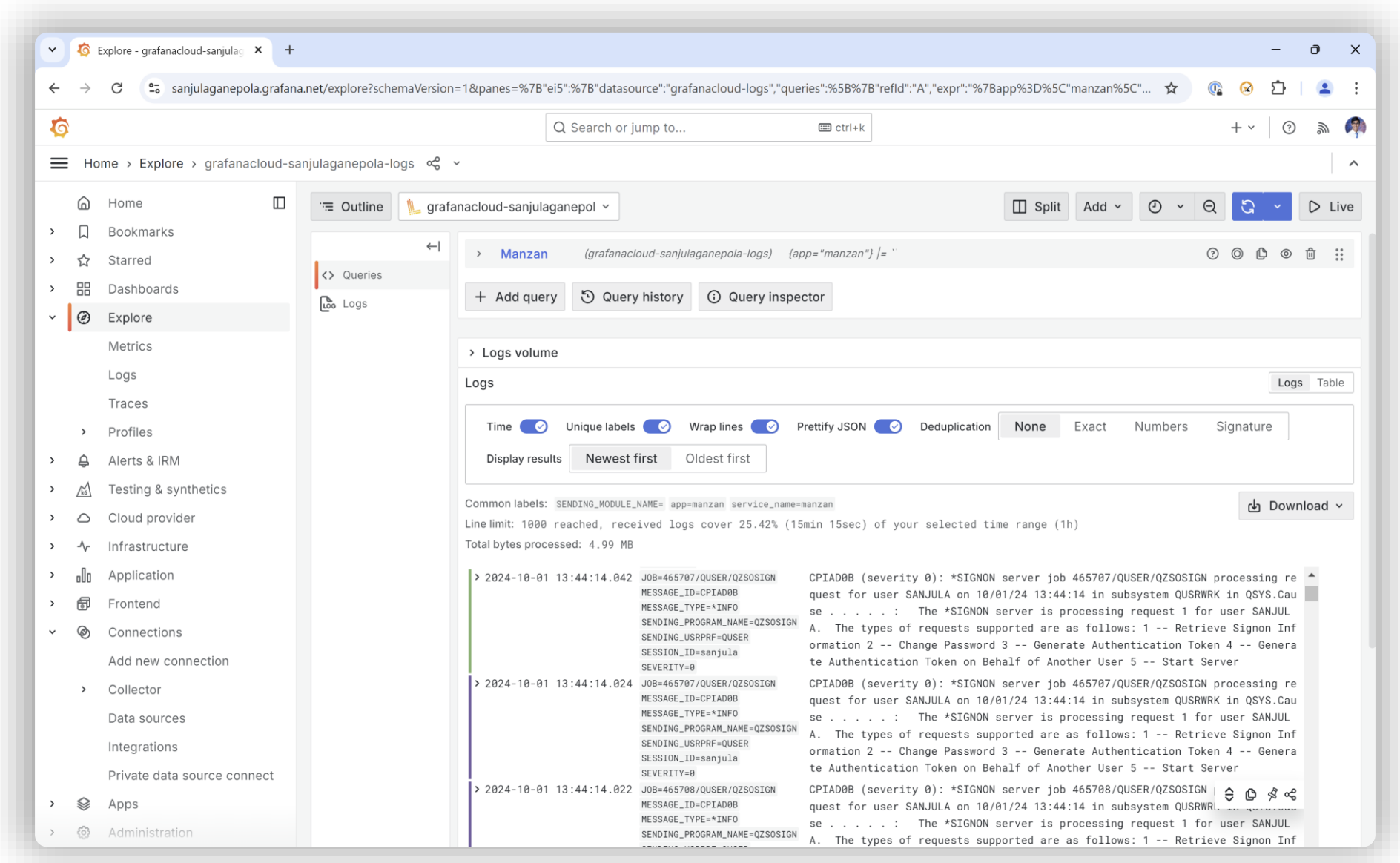
DETAILS

MESSAGES

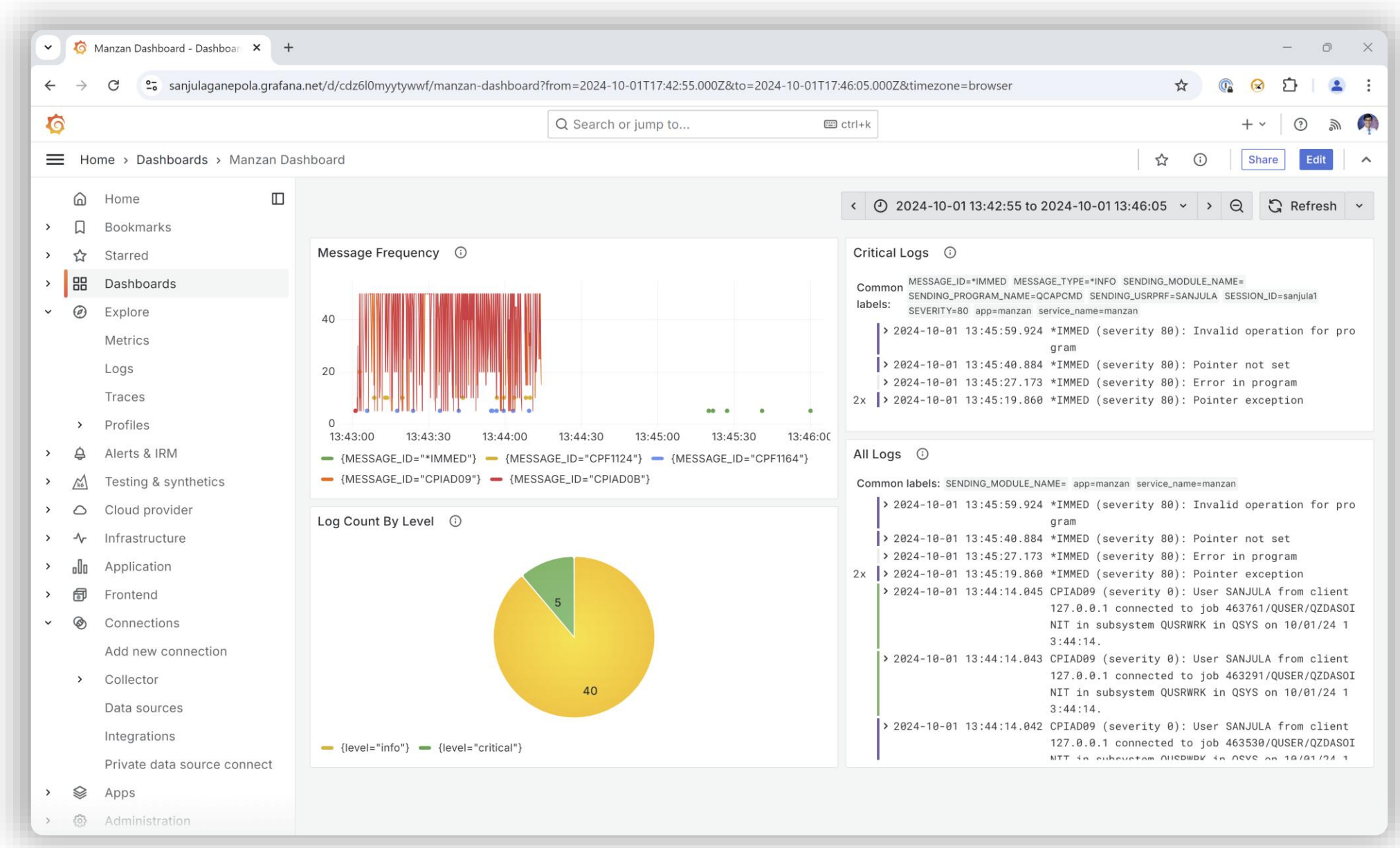
Publish time	Attribute keys	Message body	Ordering key	Ack
Oct 10, 2024, 3:14:09 PM	<div>HANDLED_TIMESTAMP</div> <div>JOB MESSAGE</div> <div>MESSAGE_ID</div> <div>MESSAGE_TIMESTAMP</div> <div>MESSAGE_TYPE</div> <div>ORDINAL_POSITION</div> <div>SENDING_MODULE_NAME</div> <div>SENDING_PROCEDURE_NAME</div> <div>SENDING_PROGRAM_NAME</div> <div>SENDING_USRPRF</div> <div>SESSION_ID SEVERITY</div>	CPIAD0B (severity 0): *SIGNON server job 540281/QUSER/QZSOSIGN processing request for user SANJULA on 10/10/24 19:10:11 in subsystem QUSRWRK in QSYS.Cause . . . . . : The *SIGNON server is processing request 1 for user SANJULA. The types of requests supported are as follows: 1 - Retrieve Signon Information 2 - Change Password 3 - Generate Authentication Token 4 - Generate Authentication Token on Behalf of Another User 5 - Start Server	-	ACK ^
Oct 10, 2024, 3:14:09 PM	<div>HANDLED_TIMESTAMP</div> ...	CPIAD09 (severity 0): User SANJULA from client 10.234.201.160 connected to job ...	-	ACK v
Oct 10, 2024, 3:14:09 PM	<div>HANDLED_TIMESTAMP</div> ...	CPIAD0B (severity 0): *SIGNON server job 540179/QUSER/QZSOSIGN processing ...	-	ACK v
Oct 10, 2024, 3:14:09 PM	<div>HANDLED_TIMESTAMP</div> ...	CPIAD0B (severity 0): *SIGNON server job 540329/QUSER/QZSOSIGN processing ...	-	ACK v
Oct 10, 2024, 3:14:09 PM	<div>HANDLED_TIMESTAMP</div> ...	CPIAD09 (severity 0): User SANJULA from client 10.234.201.160 connected to job ...	-	ACK v
Oct 10, 2024, 3:14:09 PM	<div>HANDLED_TIMESTAMP</div> ...	CPIAD09 (severity 0): User SANJULA from client 10.234.201.160 connected to job ...	-	ACK v
Oct 10, 2024, 3:14:09 PM	<div>HANDLED_TIMESTAMP</div> ...	CPIAD09 (severity 0): User SANJULA from client 10.234.201.160 connected to job ...	-	ACK v
Oct 10, 2024, 3:14:10 PM	<div>HANDLED_TIMESTAMP</div> ...	CPIAD09 (severity 0): User SANJULA from client 10.234.201.160 connected to job ...	-	ACK v

# More examples...Grafana Loki!

Video Demo: <https://www.youtube.com/watch?v=3XXjvkchCWc>



# Visualize Logs in Grafana Dashboard



# Monitor jobs with high CPU or I/O

- Possible with our new SQL support, which allows for custom queries

```
[sql1]
type=sql
query=SELECT JOB_NAME, JOB_USER, SUBSYSTEM, JOB_STATUS, CPU_TIME,
ELAPSED_CPU_PERCENTAGE, TOTAL_DISK_IO_COUNT, ELAPSED_TIME, TEMPORARY_STORAGE,
MEMORY_POOL, FUNCTION, THREAD_COUNT FROM TABLE(QSYS2.ACTIVE_JOB_INFO()) AS X
WHERE ELAPSED_CPU_PERCENTAGE > 20 OR TOTAL_DISK_IO_COUNT > 100000 ORDER BY
ELAPSED_CPU_PERCENTAGE DESC FETCH FIRST 20 ROWS ONLY
destinations=stdout
interval=300000
```

# Capturing the resource intensive jobs

```
{
  "JOB_NAME" : "748012/QSYS/QSPLMAINT",
  "JOB_USER" : "QSYS",
  "SUBSYSTEM" : null,
  "JOB_STATUS" : "EVTW",
  "CPU_TIME" : 2661,
  "ELAPSED_CPU_PERCENTAGE" : 0.00,
  "TOTAL_DISK_IO_COUNT" : 184535,
  "ELAPSED_TIME" : 0.000,
  "TEMPORARY_STORAGE" : 7,
  "MEMORY_POOL" : "BASE",
  "FUNCTION" : null,
  "THREAD_COUNT" : 1
}
{
  "JOB_NAME" : "748034/QSYS/QWCPJOBS",
  "JOB_USER" : "QSYS",
  "SUBSYSTEM" : null,
  "JOB_STATUS" : "DEQW",
  "CPU_TIME" : 13483,
  "ELAPSED_CPU_PERCENTAGE" : 0.00,
  "TOTAL_DISK_IO_COUNT" : 1032667,
  "ELAPSED_TIME" : 0.000
```

# Future Enhancements to Manzan

- System exit point processing
- Metrics exported to Prometheus, for instance:
  - How many MCH exceptions have happened in production applications?
  - How many severity 40+ history log entries?
  - How many errors showing in web server logs?
  - How many PASE VLOGs have been created?
- Ability to trigger events from Prometheus exporter, for instance:
  - Memory pool usage anomalies
  - Increased HTTP server traffic
  - Unexpected amount of remote connections
  - Operational data beyond thresholds

# **Any Questions?**

**IBM i**