

# Dynatrace

---

Innovation & Strategy



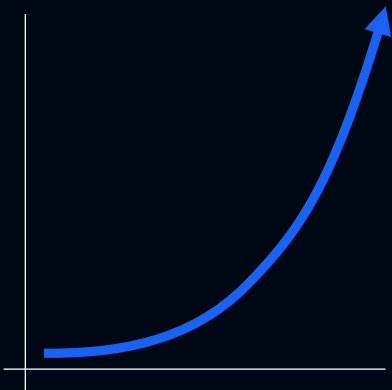
PRESENTER

Roman Spitzbart

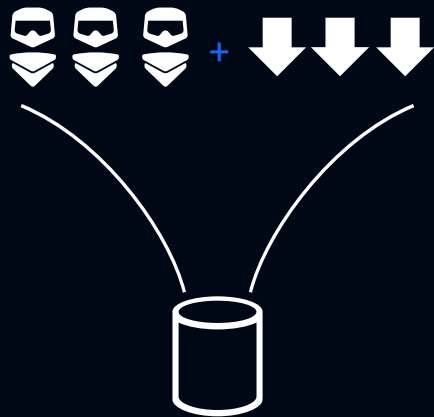
VP Solutions Engineering EMEA

# MACRO TRENDS SHOW TOOLS ARE BREAKING

Data collection  
explosion



Data ingest grows  
significantly



Observability + security,  
automation convergence



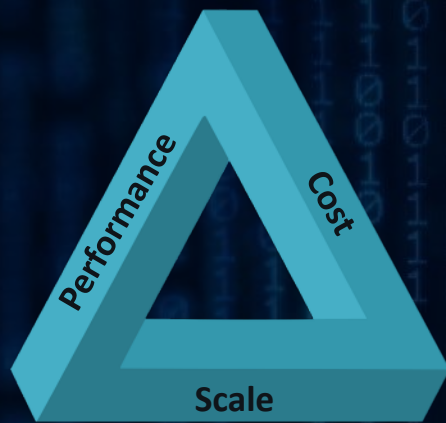
Enterprise SaaSification  
continues to rise

**51%** → **86%**  
2018 2023





# Cost of existing data platforms are outpacing the value provided



Today's data stores lack full context, require expensive infrastructure and/or cloud resources, and consume the time of expensive staff in reactive mode

expensive

slow

cumbersome

scale challenges

rigid

cross purpose



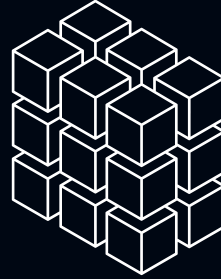
**WE'VE REACHED A TIPPING POINT IN  
TODAY'S DATA PLATFORMS**

**WE NEED INNOVATION**

---



# FROM DISPERSED DATABASES to a CAUSATIONAL MPP DATA LAKEHOUSE



**Siloed, incompatible**

**Data lake**

**Data warehouse**

**Indexed**  
🔍 wait, admins,  
expensive

**Generic**



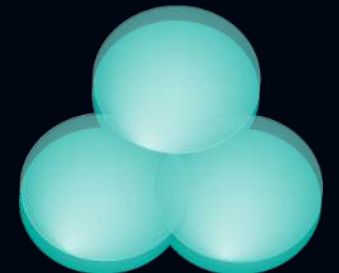
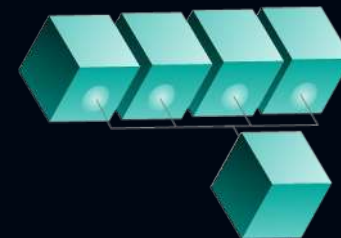
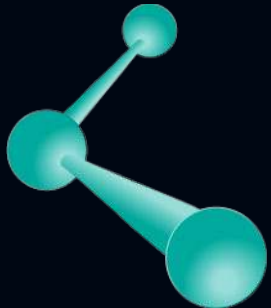
**Causational Context**

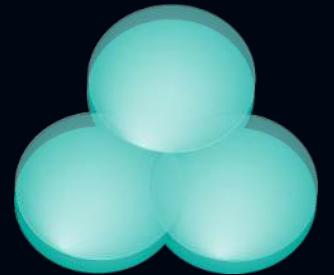
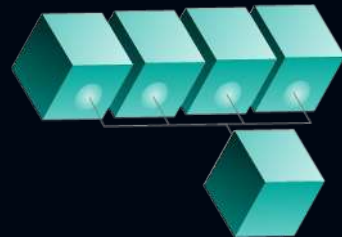
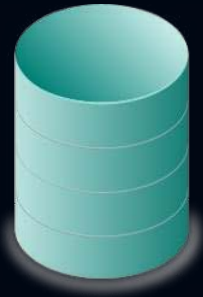
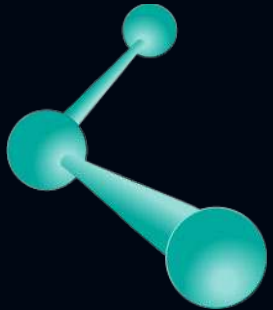
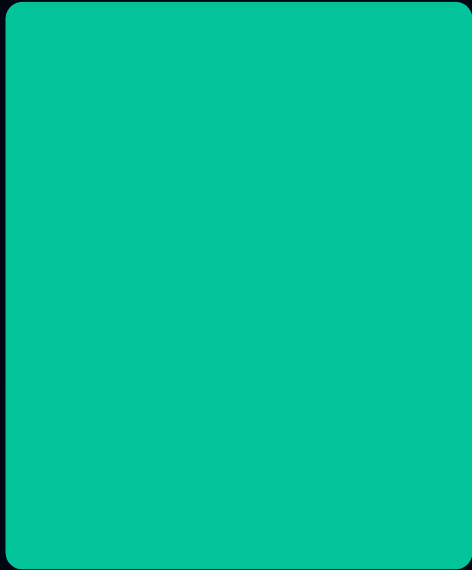
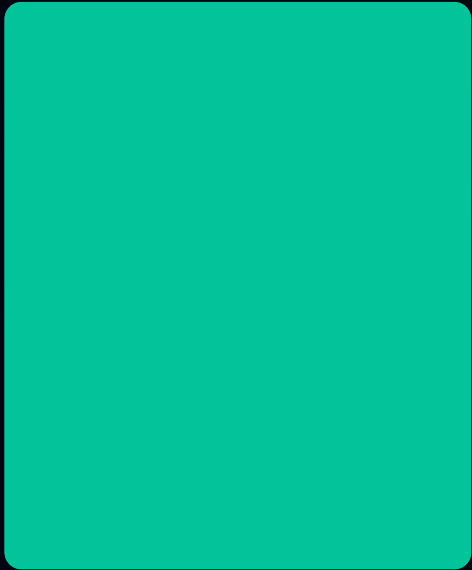
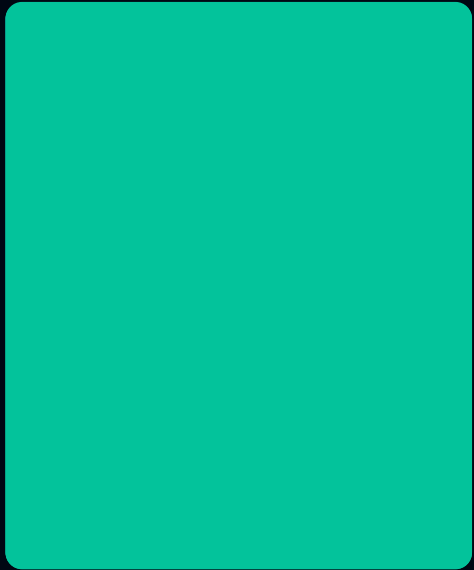
**1000 petabytes**  
Cost effective

**Instant**  
Schema-on read

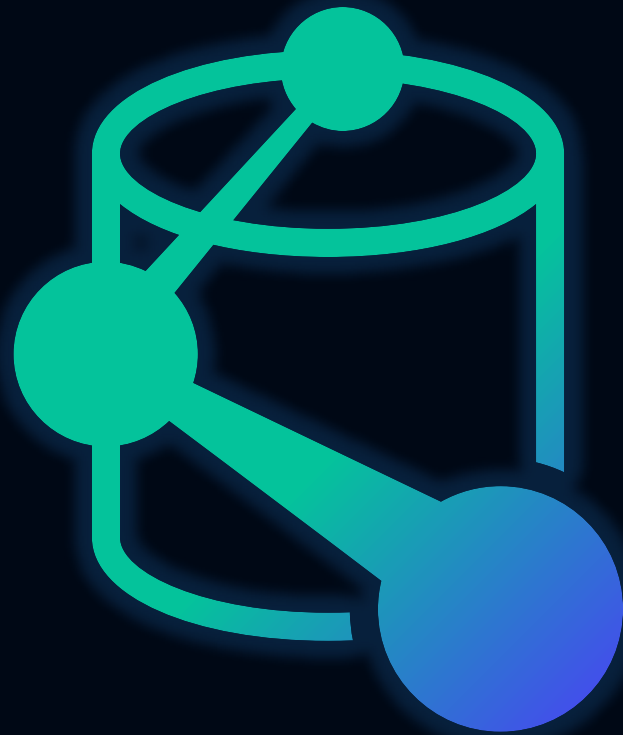
**Query at scale**  
Massively Parallel  
Processing

**Observability and**  
Security analytics









# Grail™

**Grail is a causal data lakehouse with a massively parallel processing (MPP) analytics engine. It leverages the new Dynatrace Query Language (DQL) for context-rich analytics**



# WITH GAME CHANGING VALUE ATTRIBUTES



**BILLIONS**

of dependencies can  
be analyzed in  
context

**CONTEXT**

**1000**  
petabytes/day

all retained for more  
than 12 months

**INGEST**

**5-100x**

faster to query and  
parse  
executed in massive  
parallel on 1000s of  
nodes

**SPEED**

**0 secs**

instant query,  
no index,  
no rehydration,  
no schema creation, no  
extra parsing step

**SCALE**

**1<sup>st</sup>**

All-in-one platform  
that brings  
observability,  
security and  
business together

**PLATFORM**

\* Grail™ architecture is enabling to scale to that dimension for a multi-tenant cluster, we reserve right to deliver based on future market needs

\*\* potential combined performance gain from massive-parallel processing query engine, and faster parsing compared to regular expressions, for many advanced query use-cases on large datasets

# WOVEN INTO THE ARCHITECTURE TO DELIVER ANSWERS AND INTELLIGENT AUTOMATION FROM DATA



Deep context rich, full stack beyond observability sources

Automatically captured in context & pre-processed

**NOW**  
Stored contextually with massive processing and retrieval capabilities

Accessed by our causal AI for analysis & answers

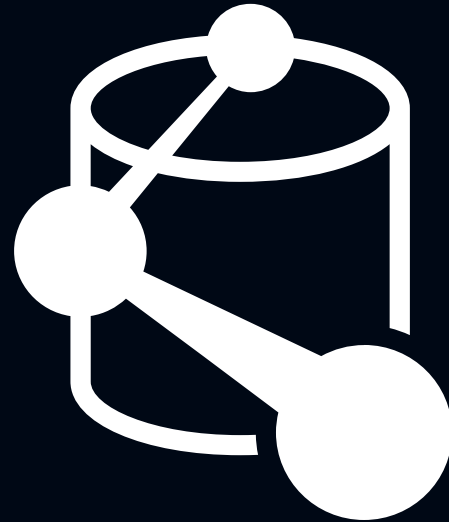
Powering automation, orchestration, prevention and protection

# GRAIL™ WILL POWER ALL PLATFORM MODULES



# LOG (R)EVOLUTION

**Log Management  
and Analytics**  
*powered by*



**Graiil™**



# LOG MANAGEMENT AND ANALYTICS

powered by



Grail™

**100TB**/day per tenant  
ingest performance towards Q1 CY23

**MPP**

**Instant!**

No rehydration  
No index generation  
No schema admin

up to... **100x faster\***  
Blazing Performance  
~1TB in ~1s @ 1000 cores

**Cost  
Efficient**

**>100x**  
more scalable \*\*

\* for many advanced queries on large datavolumes

\*\* compared to executing queries on open-source index based databases, are limited by expensive disks, compared to the built-in automatic multi-tiered approach of Grail

# EXCERPT / IN-DEPTH INDUSTRY STANDARD (INDEXED DATABASE)

1 Decide relevant indexes before ingest

2 Limited query operators

3 Simple keyword search

3 Unlimited transforming, restructuring of all raw logs

4 Wait for new logs to arrive

5 Statistical approximate (sampled) summaries

4 Full historical analysis

...vs Grail™

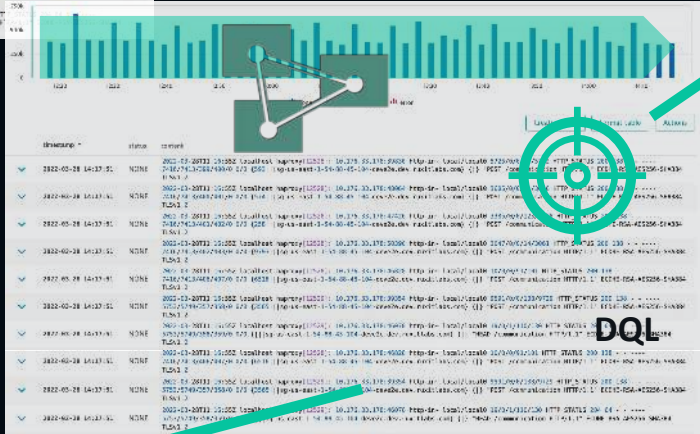
5 100% precision insight into all data stored

2 Powerful data exploration & transformation with DQL



OR, AND

1 Decide at any time what is relevant in logs



# AUTOMATIC EVERYTHING, NOW WITH A LANGUAGE

Time consuming  
writing and maintaining queries



Too complicated  
for everyday users



Lack of instant access



DQL

Continued automatic everything  
Out of the box, for common use-cases

Easy visual query building  
for simple customizations

Powerful query language  
for custom and extended use cases, instantaneously

# DYNATRACE QUERY LANGUAGE (DQL)

implicit use of DQL

The screenshot shows the 'Simple Mode' interface. At the top, there are filter tags: 'log.source: atterwind.info', 'content ~ "Received SCK data"', and 'loglevel: INFO'. Below the filters is a search bar and buttons for 'Create processing rule', 'Create metric', 'Format table', and 'Actions'. A table of search results is displayed with columns: timestamp, status, content, and stage. The content column shows JSON-like data for 'Received SCK data'.

timestamp	status	content	stage
2022-09-27 09:40:34	INFO	Received SCK data: { temp: '10.9', humidity: '83', press: '999.8', seapress: '1058.1', windspeed: '3.1', windspeedavg:...	production
2022-09-27 09:40:24	INFO	Received SCK data: { temp: '10.9', humidity: '83', press: '999.8', seapress: '1058.1', windspeed: '3.1', windspeedavg:...	production

explicit use of DQL

The screenshot shows the 'Advanced mode' interface. A code editor contains the following DQL query:

```
1 fetch logs  
2 | filter log.source == "atterwind.info" and matchesPhrase(content, "Received SCK data") and loglevel == "INFO"
```

Below the code editor is a 'Run query' button. The search results are shown in a table with columns: timestamp, status, content, loglevel, log.source, event.type, and stage.

timestamp	status	content	loglevel	log.source	event.type	stage
2022-09-18 17:04:34	INFO	Received SCK data: { temp: '10.8', humidity: '83', press: '1004.7', seapress: '1063.3', windspee...	INFO	atterwind.info	LOG	production
2022-09-18 17:06:54	INFO	Received SCK data: { temp: '10.8', humidity: '83', press: '1004.3', seapress: '1062.9', windspee...	INFO	atterwind.info	LOG	production

**Do more with less**

- | Simple visual query builder powered by DQL
- | Ability to do advanced custom queries
- | Instant access to your unknown unknowns

**All in one**

- | Ubiquitous, analytics and power



# BUILT WITH POWER AND PURPOSE

## Effortless migration

- | Minimal learning curve for new users – it just makes sense
- | Familiar to power users – i.e. Splunk

## Purpose build for observability and security

- | 50% reduced time to query
- | 5-10x faster parsing with DPL and (DPL = Dynatrace Pattern Language)
- | 90% reduced trial-and-error compared to Regular Expressions

## Readable

- | 2-100x increase in productivity on query building and collaboration
- | Step by step data processing

```
fetch logs, from:now()-10m
| filter event.type == "K8S"
| filter dt.event.group_label == "Failed"
  or k8s.container.name == "contr"
  and not contains(log.source, "c")
| fields timestamp, namespace = k8s.name
  workload = dt.kubernetes.workload
Parse HERE E HRRERERE HTERER HERHRE
| filter namespace == "dps-ingest"
| sort timestamp desc
| fields last_mountfail = timestamp, workload
| limit 1
```



# GRAIL™ WILL ENABLE MANY USE-CASES



# USE-CASES

Hybrid cloud and Kubernetes analytics	Hybrid cloud distributed tracing	Attack blocking and protection	Mobile, web browser and API	Real-time business insights	Closed-loop remediation	Ecosystem integrations
Log and event management	Automatic code-level root-cause and profiling	Vulnerability runtime analytics	Feature adoption analysis	Impact and conversion	Quality gate, service level and delivery	Custom solutions
Automatic enterprise-grade observability	Front- & back-end availability and performance	Risk-based remediation	Visual session replay	BizDevOps integration and automation	DevOps, SRE lifecycle	API programmability

Infrastructure Monitoring | Applications & Microservices | Application Security | Digital Experience | Business Analytics | Cloud Automation | Dynatrace Hub

**dynatrace** Software Intelligence Platform

OneAgent® | PurePath® | Smartscape® | Grail™ | Davis® AI

- Traces
- Metrics
- Logs
- +
- Topology
- Behaviour
- Code
- Metadata
- Network
- +
- API
- OpenTelemetry
- keptn

600+

Supported technologies

- Kubernetes
- OpenShift
- AWS
- Azure
- GCP
- Tanzu
- Enterprise
- Hybrid cloud

Automatic and intelligent observability

Broadest multicloud and technology support

# Exciting initial use cases



## Log to Metrics



## Troubleshooting



## Application Optimization



## Audit and Forensics

Logs

### Log & Events Viewer

Explore your log data using facets, or use [Dynatrace Query Language \(DQL\)](#) in advanced mode for a deeper analysis.

Query Saved Recent Sample queries

Advanced query mode

```
1 fetch logs, timeframe:"2022-04-17T00:00:00Z/2022-04-19T02:00:00Z"
2 | filter `event.type` = "LOG"
3 | summarize count(), alias:counter, by:{`host.name`, `dt.process.name`}
4 | sort `counter`, direction:"descending"
```

Run query Save query Add event

Table Chart

29,303 results

Showing latest 1K entries

timestamp	content	dt.entity.host
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 157.25.19.100:38440 http-in- individual_servers/apmng6 217/0/0/1/218 HTTP_STATUS 200 284 - - --NN 5749/5745/0/1/0 0/0 {   umsaywsjuo.dev.dynatracelabs.com:443} {} "POST /communication7...	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:44208 http-in- local/local0 5432/0/0/103/5535 HTTP_STATUS 200 138 - - ---- 7416/7413/407/408/0 0/0 {686   sg-us-east-1-54-88-45- 104-deve2e.dev.ruxitlabs.com} {} "POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 10.176.33.178:46070 http-in- local/local0 19/0/1/110/130 HTTP_STATUS 204 64 - - ---- 5753/5749/358/359/0 0/0 {   sg-us-east-1-54-88-45-104- deve2e.dev.ruxitlabs.com} {} "HEAD /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 10.176.33.178:39354 http-in- local/local0 9591/0/0/138/9729 HTTP_STATUS 200 138 - - ---- 5753/5749/357/358/0 0/0 {3565   sg-us-east-1-54-88-45- 104-deve2e.dev.ruxitlabs.com} {} "POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:46828 http-in- local/local0 10/0/0/91/101 HTTP_STATUS 200 138 - - ---- 7416/7413/406/407/0 0/0 {6516   sg-us-east-1-54-88-45-104- deve2e.dev.ruxitlabs.com} {} "POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:53612 http-in- local/local0 18317/0/0/123/18440 HTTP_STATUS 200 138 - - ---- 7416/7413/405/406/0 0/0 {6553   sg-us-east-1-54-88- 45-104-deve2e.dev.ruxitlabs.com} {} "POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:45988 http-in- local/local0 3243/0/0/37/3280 HTTP_STATUS 200 138 - - ---- 7416/7413/404/405/0 0/0 {2283   sg-us-east-1-54-88-45- 104-deve2e.dev.ruxitlabs.com} {} "POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:54274 http-in- local/local0 22918/0/0/42/22960 HTTP_STATUS 200 138 - - ---- 7416/7413/403/404/0 0/0 {582   sg-us-east-1-54-88-45- 104-deve2e.dev.ruxitlabs.com} {} "POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 10.176.33.178:44112 http-in- local/local0	



Simple mode:  
quick search

Advanced mode:  
DQL queries

Actionable  
results:  
- create charts  
- pin to  
dashboard

## Log & Events Viewer

Explore your log data using facets, or use [Dynatrace Query Language \(DQL\)](#) in advanced mode for a deeper analysis.

Query Saved Recent Sample queries

Advanced query mode Clear query

```

1 fetch logs, timeframe:"2022-04-17T00:00:00Z/2022-04-19T02:00:00Z"
2 | filter `event.type` = "LOG"
3 | summarize count(), alias:counter, by:({`host.name`, `dt.process.name`})
4 | sort `counter`, direction:'descending'

```

Run query Save query Add event

Table Chart

29,303 results Show latest 10 entries Add metric Format table Download Share

timestamp	content	dt.entity.host
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 157.25.19.100:38440 http-in- individual_servers/apmg6 217/0/0/1/218 HTTP_STATUS 200 284 - - - - - NN 5749/5745/0/1/0 0/0 {}    unsays uo.dev.dynatrace.com:443} {} *POST /communication?...	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:44200 http-in- local/local0 5432/0/0/103/5535 HTTP_STATUS 200 138 - - - - - 7416/7413/407/408/0 0/0 {686}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 10.176.33.178:46070 http-in- local/local0 19/0/1/110/130 HTTP_STATUS 204 64 - - - - - 5753/5749/358/359/0 0/0 {}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *HEAD /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 10.176.33.178:39354 http-in- local/local0 9591/0/0/138/9729 HTTP_STATUS 200 138 - - - - - 5753/5749/357/358/0 0/0 {3565}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:46828 http-in- local/local0 10/0/0/91/101 HTTP_STATUS 200 138 - - - - - 7416/7413/406/407/0 0/0 {6516}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:53612 http-in- local/local0 10317/0/0/123/18440 HTTP_STATUS 200 138 - - - - - 7416/7413/405/406/0 0/0 {6553}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:45988 http-in- local/local0 3243/0/0/37/3280 HTTP_STATUS 200 138 - - - - - 7416/7413/404/405/0 0/0 {2283}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:54274 http-in- local/local0 22918/0/0/42/22960 HTTP_STATUS 200 138 - - - - - 7416/7413/403/404/0 0/0 {5821}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12529]: 10.176.33.178:44112 http-in- local/local0 3147/0/0/138/3285 HTTP_STATUS 200 138 - - - - - 5753/5749/357/358/0 0/0 {9261}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:50390 http-in- local/local0 3047/0/0/14/3061 HTTP_STATUS 200 138 - - - - - 7416/7413/402/403/0 0/0 {9793}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:47426 http-in- local/local0 3385/0/0/123/3508 HTTP_STATUS 200 138 - - - - - 7416/7413/401/402/0 0/0 {258}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:48064 http-in- local/local0 3665/0/0/61/3666 HTTP_STATUS 200 138 - - - - - 7416/7413/400/401/0 0/0 {5741}    sg-us-east-1-54-88-45-104-deve2e.dev.ruxitlabs.com} {} *POST /communication HTTP/1.1" ECDHE-RSA-AES256-SHA384 TLSv1.2	HOST-IG-73-50729
2022-04-19 21:43:22	2022-04-19T19:42:35Z localhost haproxy[12528]: 10.176.33.178:39830 http-in- local/local0 5725/0/0/67/5792 HTTP_STATUS 200 138 - - - - - 7416/7413/399/400/0 0/0 {5931}    sg-us-east-1-54-88-45-	HOST-IG-73-50729

### Kubernetes event

Context

**Warning** 2021-12-16 18:04:23

[View user session](#) [View trace](#)

Application namespace: oep-load-generator-ni

Kubernetes cluster: <https://k8s-api.lab.dynatrace.org>

Source entity: [k8s-oneagent-linux-x86-docker-mm9t](#)

Message [View pipeline](#)

0/31 nodes are available: 15 Insufficient cpu, 2 node(s) were not ready, 2 node(s) were out of disk space, 2 node(s) were unschedulable, 27 node(s) didn't match node selector.

[Attributes](#) [Original message](#)

Search for key or value

Core

log.source: <https://k8s-api.lab.dynatrace.org>

Dynatrace

dt.entity.cloud\_application: CLOUD\_APPLICATION-BAFD088FE5C/BC18 (not monitored)

dt.event.group.label: FailedScheduling

dt.events.root\_cause\_relevant: true

dt.kubernetes.cluster\_name: <https://k8s-api.lab.dynatrace.org>

dt.kubernetes.event\_involved\_object\_kind: Pod

dt.kubernetes.event\_involved\_object\_name: [k8s-oneagent-linux-x86-docker-mm9t](#)

dt.kubernetes.event\_reason: FailedScheduling

dt.kubernetes.workload\_kind:

Meaningful  
context:  
- linked traces  
- sessions

Drill down  
to details



# Deployment Verification example

Questions are asked quickly....

Find **all apps** with **failure** trace logged **more than 10x** within a **5-minute interval**.

... answers are **even faster**:

Logs powered by Grail

```
1 fetch logs
2 | filter contains(content,"failure")
3 | summarize failurecount=count(), by:{bin(timestamp, 5m), app}
4 | filter failurecount > 10
5 | summarize by:app
```

Query with DQL

Use entity model

Compare logs with metrics

Generate reports

Pin on dashboard

# Troubleshooting - Fault isolation example

Questions are asked **quickly...**

Are our **app error logs** correlated with **too many requests from the same IP?**  
(example where IP address is not extracted/indexed on ingest)

... answers are **even faster:**

Logs powered by Grail

```
1 fetch logs
2 | filter dt.process.name=="myApp" and status=="ERROR"
3 | parse content, "LD IPADDR:ip"
4 | summarize errorcount=count(), by:{ip}
5 | sort errorcount
```

Query with DQL

Use entity model

Compare logs with metrics

Generate reports

Pin on dashboard

# Audit & Forensics example

Questions are asked **quickly...**

Is that a **malicious user**? Check the logs for **similar user input patterns** for **last 12 months**.

... answers are **even faster:**

Logs powered by Grail

```
1 fetch logs, from:now()-1y
2 | filter endsWith(log.source,"audit.log") and
3 (action.type == "change_password" or action.type=="reset_password")
4 | parse content, "LD:userId ',' DATA:oldPW (' !>>SPACE) LD:to EOL"
5 | summarize attempts=count(), by:{bin(timestamp,1d),userId}
6 | sort attempts desc
```

Query with DQL

Use entity model

Compare logs with metrics

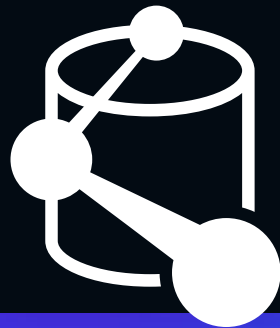
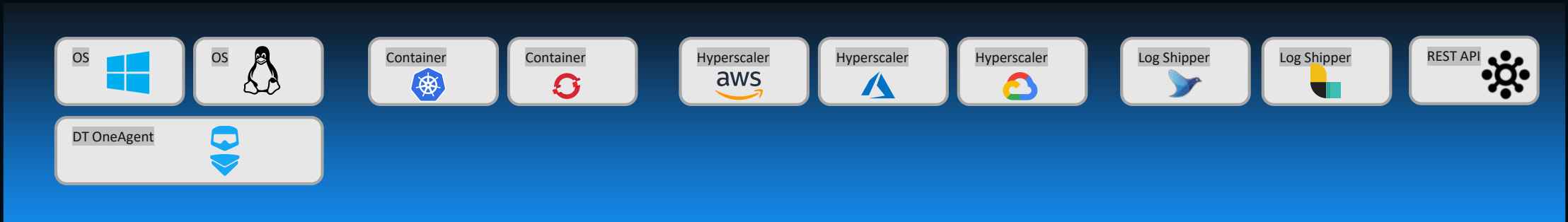
Generate reports

Pin on dashboard

# Easy enablement, broad tech coverage

✓ Install **OneAgent** or use the **REST API**  
→ start ingesting data

✗ *No index creation*  
*No data schema decisions*  
*No ballooning of retention costs*

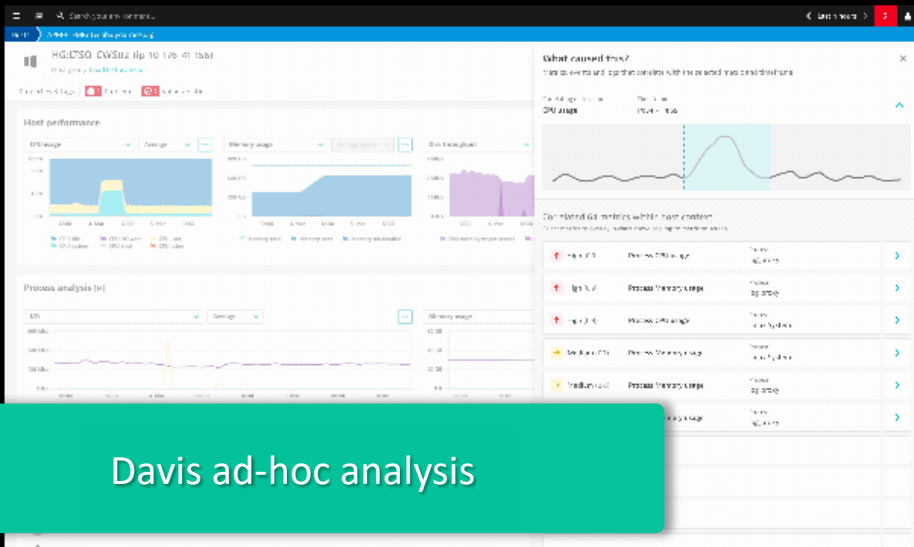


Grail™

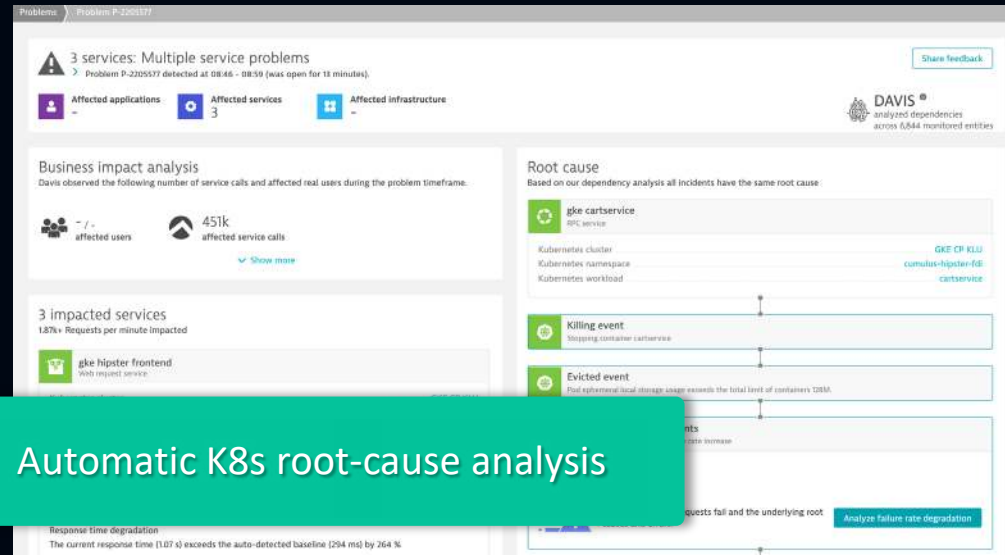


# NOTHING BEATS DAVIS

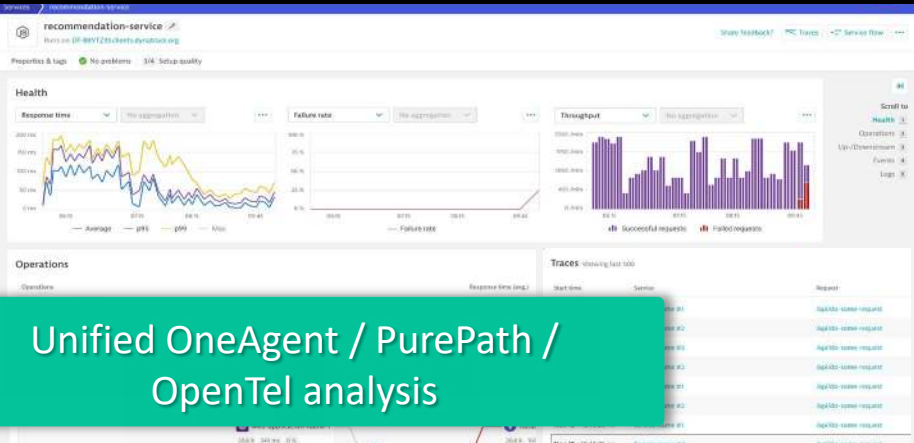
## WE CONTINUE TO ENHANCE OUR CAUSAL AI ENGINE



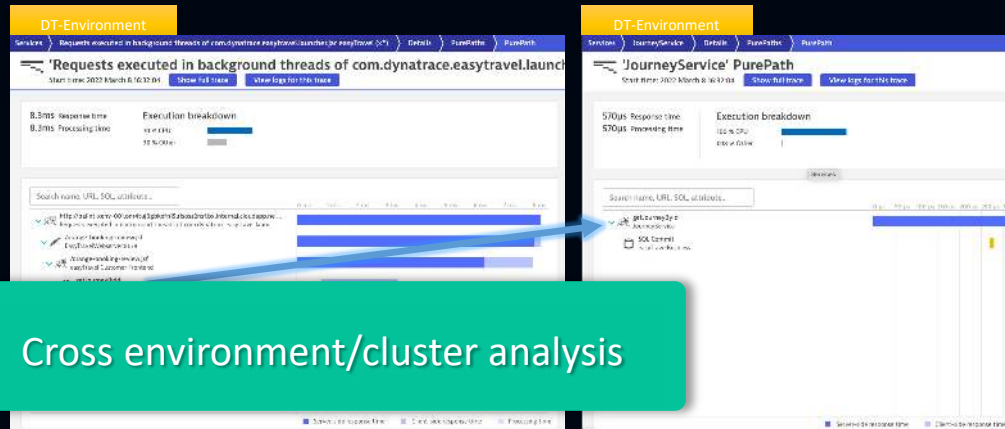
Davis ad-hoc analysis



Automatic K8s root-cause analysis



Unified OneAgent / PurePath / OpenTel analysis



Cross environment/cluster analysis

# AND ULTIMATELY WE AUTOMATE, NOT-JUST-DATA ON GLASS



Cloud Automation / dynatrace

2 Services

- carts-db**  
Last processed artifact: mongo:4.2.2
- carts**  
Last processed artifact: carts:0.11.3  
Last time fetched: Today at 10:17:41

**Problem resolved** 2021-01-16 17:38  
Response time degradation (production)

**Configuration changed** 2021-01-16 14:59  
carts:0.11.3 (staging production)

**Configuration changed** 2021-01-16 14:29  
carts:0.11.2 (staging)

**Service created** 2021-01-16 14:27

**Problem opened** 2021-01-16 17:38  
Source: dynatrace-service  
Problem: Response time degradation on Web service sockshop.carts.production  
Labels: Problem URL

**Remediation triggered** 2021-01-16 17:38  
Source: remediation-service  
Labels: Problem URL

**Remediation status changed** 2021-01-16 17:38  
Source: remediation-service  
Labels: Problem URL

**Action triggered** 2021-01-16 17:38  
Source: remediation-service

**Action started** 2021-01-16 17:38  
Source: helm-service

**Action finished** 2021-01-16 17:39  
Source: helm-service

**Problem resolved** 2021-01-16 17:47  
Source: dynatrace-service  
Problem: Response time degradation on Web service sockshop.carts.production  
Labels: Problem URL

# THE POWER OF THE PLATFORM IS EVOLVING

## A WORLD WHERE SOFTWARE WORKS PERFECTLY

**LOG**

Hybrid cloud and Kubernetes analytics | Hybrid cloud distributed tracing | Attack blocking and protection | Mobile, web browser API | Real-time business insights | Closed-loop remediation | Ecosystem integrations

Log and event management | Automatic code level and profiling | Vulnerability runtime analytics | Feature adoption analysis | Impact and conversion | Quality level analysis | Cloud solution | Automatic enterprise grade observability | Scalable cloud availability and performance | Risk-based remediation | BizDevOps integration and automation | DevOps lifecycle | API programmability

Infrastructure Monitoring | Applications & Microservices | Application Security | Digital Experience | Business Analytics | Cloud Automation | Dynatrace Hub

**dynatrace** Software Intelligence Platform

OneAgent® | PurePath® | Smartscape® | Grail™ | Davis® AI

Traces | Metrics | Logs | + | Topology | Behaviour | Code | Metadata | Network | + | API | OpenTelemetry | k8s | **600+** Supported technologies | Kubernetes | OpenShift | AWS | Azure | GCP | Tanzu | Enterprise | Hybrid cloud

Automatic and intelligent observability | Broadest multicloud and technology support

**THANK YOU**

---