Prometheus, Jaeger, and Istio:

Observability for Today’s Enterprises
About Me

Priyanka Sharma
Director of Cloud Native Alliances,
GitLab
Contributor, OpenTracing
@pritianka
The **open source** ecosystem is teaming with projects related to **observability**. It is imperative to understand the top projects and **develop a framework** for utilizing them together for a cohesive observability strategy **that meets business goals**.
Special thanks to

Cindy Sridharan, Observability Badass (Apple)
Richard Hartman, OpenMetrics
Ted Young, OpenTracing
What is Observability?
OH - "Observability - because devs don't like to do "monitoring" we need to package it in new nomenclature to make it palatable and trendy."
Observability is a superset of monitoring and consists of four pillars that should help us reason about and respond to modern software systems.

Pillars of Observability:
- **Metrics**
- **Tracing**
- **Logging**
- **Alerts**

Build your own definition: [https://medium.com/@copyconstruct/monitoring-and-observability-8417d1952e1c](https://medium.com/@copyconstruct/monitoring-and-observability-8417d1952e1c)
Why Observability Matters
Speeding Up Release Cycle Time is Critical

Cycle time compression may be the most underestimated force in determining **winners & losers** in tech.

— Marc Andreessen
Business survival depends upon a radically faster DevOps lifecycle
We are not at peak performance today
DevOps and Observability have not done as well as expected

- $3.9B spent on DevOps software
- >50% DevOps time wasted on logistics & repeatable tasks
- 87% organizations are disappointed with results of DevOps
Today’s toolchain limits a faster DevOps lifecycle
Integration complexity of toolchains slows down teams

Plan
- Portfolio mgmt
- Issue tracking

Create
- Version control
- Code review

Verify
- Continuous integration
- Security testing

Package
- Container registry

Release
- CD/Release

Configure
- Configuration Management

Monitor
- Monitoring

Secure
- Security testing

Plan
- Portfolio mgmt
- Issue tracking

Create
- Version control
- Code review

Verify
- Continuous integration
- Security testing

Package
- Container registry

Release
- CD/Release

Configure
- Configuration Management

Monitor
- Monitoring

Secure
- Security testing

https://about.gitlab.com/sdlc/#interfaces
The Toolchain Crisis is costing $$$
The Price We Pay

Time and cost to **acquire** all of these tools

Time and cost to **integrate** all of these tools

User context **switching** between all of these tools
And it’s getting worse

The shift to microservices creates an explosion of projects today’s toolchain is not built to handle.
Observability Challenges for microservices

- Cloud computing and orchestration enables complexity
- This complexity manifests itself as:
  - Fragmentation
  - Data explosion
  - Siloed teams
  - Operational issues
  - Terrifying MTTR

Observability contains this complexity by providing ways for us to reason about our systems
Pillars of Observability

Artificial trichotomy caused by cost issues
Pillars of Observability - Metrics
### Pillars of Observability - Logging

![Splunk Search Console](image)

**Example Splunk Search Query**
```
source="/var/log/containers/counter*" | rex field=source "file:/var/log/containers/(/?pod=[a-zA-Z0-9-]*_)?(?!namespace=[a-zA-Z0-9-]*_)?(?!container=[a-zA-Z0-9-]*_)?(?!conatinerid=[a-zA-Z0-9-]*_)? table time, host, namespace, pod, container, log
```

**Table Example**

<table>
<thead>
<tr>
<th>time</th>
<th>host</th>
<th>namespace</th>
<th>pod</th>
<th>container</th>
<th>log</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-10-17T02:15:50.059931183Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>779: Tue Oct 17 02:15:50 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:49.058977192Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>778: Tue Oct 17 02:15:49 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:48.05839533Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>777: Tue Oct 17 02:15:48 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:47.057115148Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>776: Tue Oct 17 02:15:47 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:46.056241377Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>775: Tue Oct 17 02:15:46 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:45.055318588Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>774: Tue Oct 17 02:15:45 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:44.054317664Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>773: Tue Oct 17 02:15:44 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:43.053397666Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>772: Tue Oct 17 02:15:43 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:42.052420032Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>771: Tue Oct 17 02:15:42 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:41.051545289Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>770: Tue Oct 17 02:15:41 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:40.050389907Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>769: Tue Oct 17 02:15:40 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:39.049403142Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>768: Tue Oct 17 02:15:39 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17T02:15:38.048332062Z</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>767: Tue Oct 17 02:15:38 UTC 2017</td>
</tr>
</tbody>
</table>
Pillars of Observability - Tracing
The Holy Grail of System Analysis

- **Alerts**
  - p99 5% example trace 1
  - p95 10% example trace 2

- **Metrics**
- **Tracing**
- **Logging**

Relevant logs
Some Top Tools in the Ecosystem

- GitLab
- Prometheus
- Istio
- fluentd
- Grafana
- Jaeger
- OpenTracing
An open-source monitoring system with a dimensional data model, flexible query language, efficient time series database and modern alerting approach.
Prometheus

- Specialized - only does metrics and does them well
- Operationally easy
- Low barrier to entry because it is very easy to emit data to create metrics
- Extremely scalable for the Enterprise: Designed to handle 10x of the scale that the enterprise users currently have. Fortune 500 companies are happy with single instances. They can do hundreds of thousands of single values per second for time series per instance. **1.5M/sec has been done.**
Vendor-neutral APIs and instrumentation for distributed tracing
OpenTracing

● “A vendor-neutral open standard for distributed tracing”
  ○ Open Source API
  ○ Enables instrumenting existing code with distributed tracing functionality
  ○ Does not commit to any implementation (Jaeger, Zipkin, LightStep, etc.)
  ○ Bit of History
    ■ Google’s Dapper ~2000
    ■ Twitter’s Zipkin ~2009
    ■ OpenTracing in 2016
      ● Built by industry experts to solve common problem
GitLab is the first **single application for the entire DevOps lifecycle** and it accelerates the software lifecycle by over 200%.
An open platform to **connect, monitor, and secure** microservices.
Istio features

Istio Security

Ease the burden of security, freeing your developers to focus on other critical tasks.

Istio Monitoring

Detect and fix issues quickly and effectively with robust, easy-to-use monitoring.

Istio Connect

Istio simplifies traffic management as your deployment scales.
Envoy: Baked into Istio

OUT OF PROCESS ARCHITECTURE
Envoy is a self-contained, high performance server with a small memory footprint. It runs alongside any application language or framework.

HTTP/2 AND GRPC SUPPORT
Envoy has first-class support for HTTP/2 and gRPC for both incoming and outgoing connections. It is a transparent HTTP/1.1 to HTTP/2 proxy.

ADVANCED LOAD BALANCING
Envoy supports advanced load balancing features including automatic retries, circuit breaking, global rate limiting, request shadowing, zone local load balancing, etc.

APIS FOR CONFIGURATION MANAGEMENT
Envoy provides robust APIs for dynamically managing its configuration.

OBSERVABILITY
Deep observability of L7 traffic, native support for distributed tracing, and wire-level observability of MongoDB, DynamoDB, and more.
Best Practices for Engineering Leaders
Essential Question

What is your company-wide workflow?
You need something that is flexible and yet ...

**VISIBLE**
Real time view across the entire lifecycle

**EFFICIENT**
Collaborate without waiting

**GOVERNED**
Develop and operate with confidence
Traditional DevOps toolchain

Microsoft Word

- One person edits at a time
- Multiple copies (hand offs)
- Version conflicts
- Waiting for feedback
- Sequential
What it could be like

**Microsoft Word**
- One person edits at a time
- Multiple copies (hand offs)
- Version conflicts
- Waiting for feedback
- Sequential

**Google Docs**
- Many people edit at the same time
- One copy
- No conflicts
- Real time feedback
- Concurrent
Start with the external objective in mind

What have you promised to deliver your customer?
Defining your observability stack and culture

Waterfall the external objective internally

*What is the objective for each company, organization, department, team, engineer?*
Define what data you need

*Use the objectives gathered to *identify the data* you need*
Instrument for depth, not breadth

Go for the home run, not the test play
Some opinions

Logs should not be your first stop

Focus on **metrics first**. If you want to improve or fix that's where traces come in. Logs come in for compliance and when things really go south.
Don’t overburden humans

Codify alert states and don’t randomly set them up based on (well-meaning) human whims and desires. Humans are expensive.
A service mesh isn’t for observability

Getting a **service mesh is not a substitute** for instrumenting your system and connecting the dots between processes. **BUT**, the more you use a service mesh, the more you need to observe said mesh.
Create a hub model that is flexible and yet ...

VISIBLE
Real time view across the entire lifecycle

EFFICIENT
Collaborate without waiting

GOVERNED
Develop and operate with confidence
Resources

... because 20 minutes is not enough
Resources

- Prometheus: [https://prometheus.io/](https://prometheus.io/)
- OpenTracing: [http://opentracing.io/](http://opentracing.io/)
- Envoy: [https://www.envoyproxy.io/](https://www.envoyproxy.io/)
- Istio: [https://cloud.google.com/istio](https://cloud.google.com/istio)
- Grafana: [https://grafana.com/](https://grafana.com/)
- GitLab: about.gitlab.com
- Blog post: [Monitoring and Observability](https://www.cindy-sridharan.com/blog/monitoring-and-observability) by Cindy Sridharan
- Blog post: [Bryan Cantrill’s blog](https://bryancantrill.com/blog)
Thank you!

Let’s stay in touch:
@pritianka
psharma@gitlab.com
Step 1: Metrics monitored
Step 2: Alerts drill down to Tracing
Step 3: Tracing drill down to Logs

```
source="/var/log/containers/counter** | rex field=source ".*/var/log/containers/\([^pod=[a-zA-Z0-9]*\)\](?=[^container=[a-zA-Z0-9]*]) | table time, host, namespace, pod, container, log
```

<table>
<thead>
<tr>
<th>time</th>
<th>host</th>
<th>namespace</th>
<th>pod</th>
<th>container</th>
<th>log</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-10-17 02:15:00</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>777: Tue Oct 17 02:15:00 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:49</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>778: Tue Oct 17 02:15:49 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:48</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>777: Tue Oct 17 02:15:48 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:47</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>776: Tue Oct 17 02:15:47 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:46</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>775: Tue Oct 17 02:15:46 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:45</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>774: Tue Oct 17 02:15:45 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:44</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>773: Tue Oct 17 02:15:44 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:43</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>772: Tue Oct 17 02:15:43 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:42</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>771: Tue Oct 17 02:15:42 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:41</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>770: Tue Oct 17 02:15:41 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:40</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>769: Tue Oct 17 02:15:40 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:39</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>768: Tue Oct 17 02:15:39 UTC 2017</td>
</tr>
<tr>
<td>2017-10-17 02:15:38</td>
<td>splunk-forwarder-bqfbq</td>
<td>sample</td>
<td>counter-1-smzfit</td>
<td>counter</td>
<td>767: Tue Oct 17 02:15:38 UTC 2017</td>
</tr>
</tbody>
</table>
The Holy Grail of System Analysis - One Unified Experience

- Metrics
- Tracing
- Logging
Prometheus: Cool features

**Dimensional data**
Prometheus implements a highly dimensional data model. Time series are identified by a metric name and a set of key-value pairs.

**Powerful queries**
A flexible query language allows slicing and dicing of collected time series data in order to generate ad-hoc graphs, tables, and alerts.

**Great visualization**
Prometheus has multiple modes for visualizing data: a built-in expression browser, Grafana integration, and a console template language.

**Efficient storage**
Prometheus stores time series in memory and on local disk in an efficient custom format. Scaling is achieved by functional sharding and federation.

**Simple operation**
Each server is independent for reliability, relying only on local storage. Written in Go, all binaries are statically linked and easy to deploy.

**Precise alerting**
Alerts are defined based on Prometheus’s flexible query language and maintain dimensional information. An alertmanager handles notifications and silencing.

**Many client libraries**
Client libraries allow easy instrumentation of services. Over ten languages are supported already and custom libraries are easy to implement.

**Many integrations**
Existing exporters allow bridging of third-party data into Prometheus. Examples: system statistics, as well as Docker, HAProxy, StatsD, and JMX metrics.
Digital Ocean, Erikkson, Improbable, SoundCloud, CloudFlare, Red Hat, ING, Fastly, Fitbit, and many more
Grafana, GitLab, Pager Duty, Slack

OpenMetrics: Cleaned up, tighter format for Prometheus data with support from industry players … or so I hear. Jury is out currently.
OpenTracing: Complementary

Jaeger, LightStep, Instana, New Relic, DataDog, Grafana, Pager Duty, Slack
GitLab: Complementary

Prometheus, Grafana, Jaeger (coming soon), logging systems of choice, service meshes
Envoy: Adoption
Envoy: Complementary

OpenTracing, Grafana, GitLab, Pager Duty, Slack
Example trace

<table>
<thead>
<tr>
<th>Operation</th>
<th>Component</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_donuts</td>
<td>donut-browser</td>
<td>10.2s</td>
</tr>
<tr>
<td>ingress</td>
<td>front-proxy</td>
<td>2.77s</td>
</tr>
<tr>
<td>ingress</td>
<td>donutsalon-1</td>
<td>2.77s</td>
</tr>
<tr>
<td>order_donut(chocolate)</td>
<td>donut-webserver</td>
<td>2.77s</td>
</tr>
<tr>
<td>make_donut</td>
<td>donut-webserver</td>
<td>2.77s</td>
</tr>
<tr>
<td>process_payment</td>
<td>charge-hard</td>
<td>250ms</td>
</tr>
<tr>
<td>ingress</td>
<td>front-proxy</td>
<td>1.82s</td>
</tr>
<tr>
<td>ingress</td>
<td>donutsalon-1</td>
<td>1.82s</td>
</tr>
</tbody>
</table>
Envoy - baked into Istio

- Addresses the core issues of distributed systems:
  - Networking
  - Observability
- Battle-tested at Lyft
- Built upon the learnings from Nginx, HAProxy, hardware load balancers, and cloud load balancers
The Ideal Workflow
OpenTracing Architecture

- microservice process
  - application logic
  - web frameworks
  - control-flow packages
  - RPC libraries
  - Logging and metrics

OpenTracing API

Main() function

Tracing infrastructure

- Zipkin
- Jaeger
- Lightstep
- Appdash
- Trace
GitLab: End-User adoption