

There's a Bug in My  
Service Mesh! What Do  
You Do When the Mesh  
is At Fault?



LINKERD + PAYBASE<sup>®</sup>

# Ana Calin

Systems Engineer @ Paybase

 @AnaMariaCalin

 @calinah



"White dog smiles at the camera" by Ashley Coates is licensed under CC BY-SA 2.0

# Risha Mars

Software Engineer @ Buoyant

 @marzipan

 @rmars



# Mmm...what's cooking?

Mushrooms & trees



Meshes - what is Linkerd?

Mysteries - symptoms of the bug

Microscopes - finding the bug

Mastery! - understanding how to find bugs!







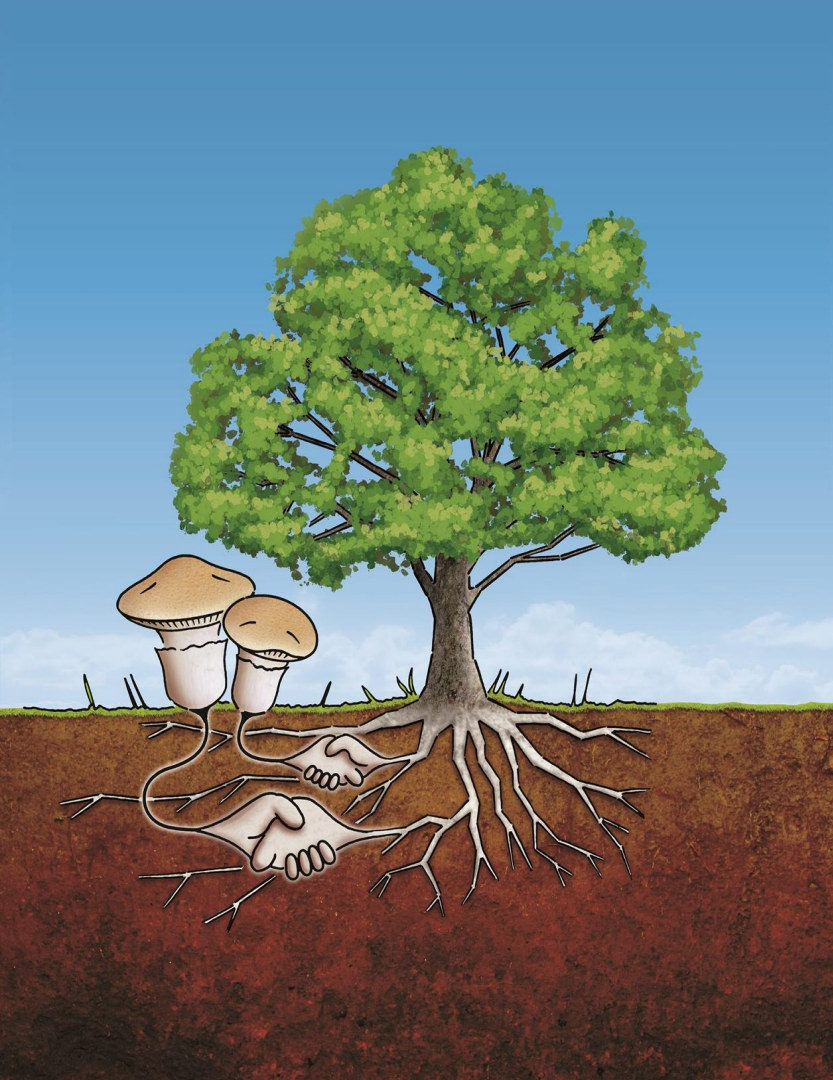
# Interacting with an OSS project

## Users ~~vs~~ Maintainers

**and**



# Users & Maintainers of OSS



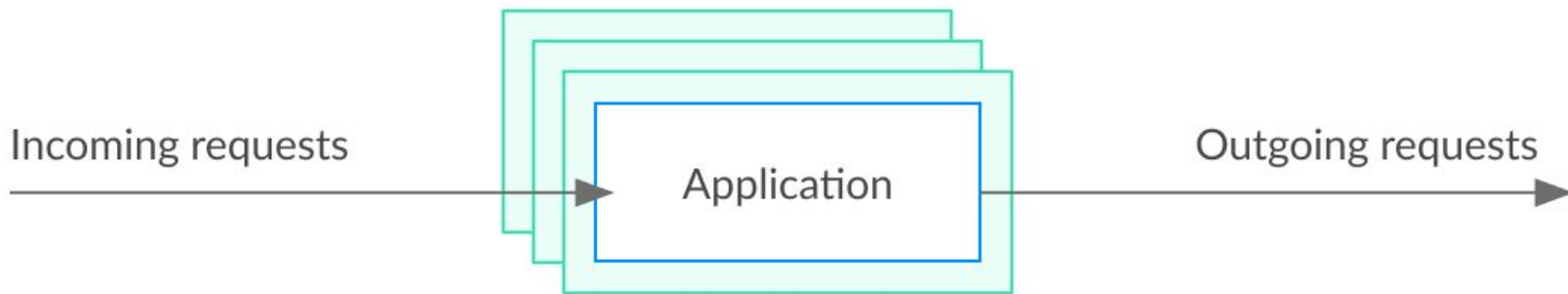
- Users and maintainers should be in a symbiotic relationship
- An OSS project grows based on user feedback, user testing and user contributions
- Some OSS projects are not funded and the work is not paid for so be nice - approach everything in a blame-free way
- Effective communication can open opportunities to learn and fix bugs fast and painlessly
- Being nice is nice

The background of the slide is a complex geometric pattern composed of numerous triangles of varying sizes. The triangles are colored in shades of light blue, medium blue, and a muted green. They are arranged in a way that creates a sense of depth and movement, with some triangles pointing towards the center and others pointing outwards. The overall effect is a modern, abstract design.

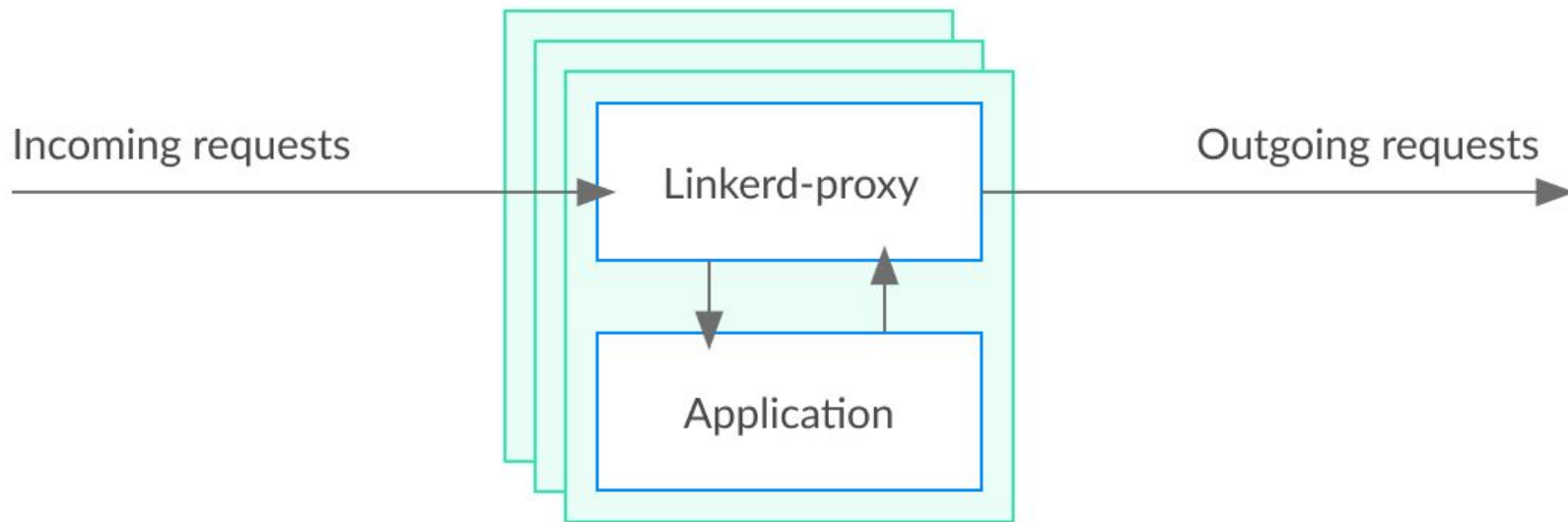
# What is Linkerd?



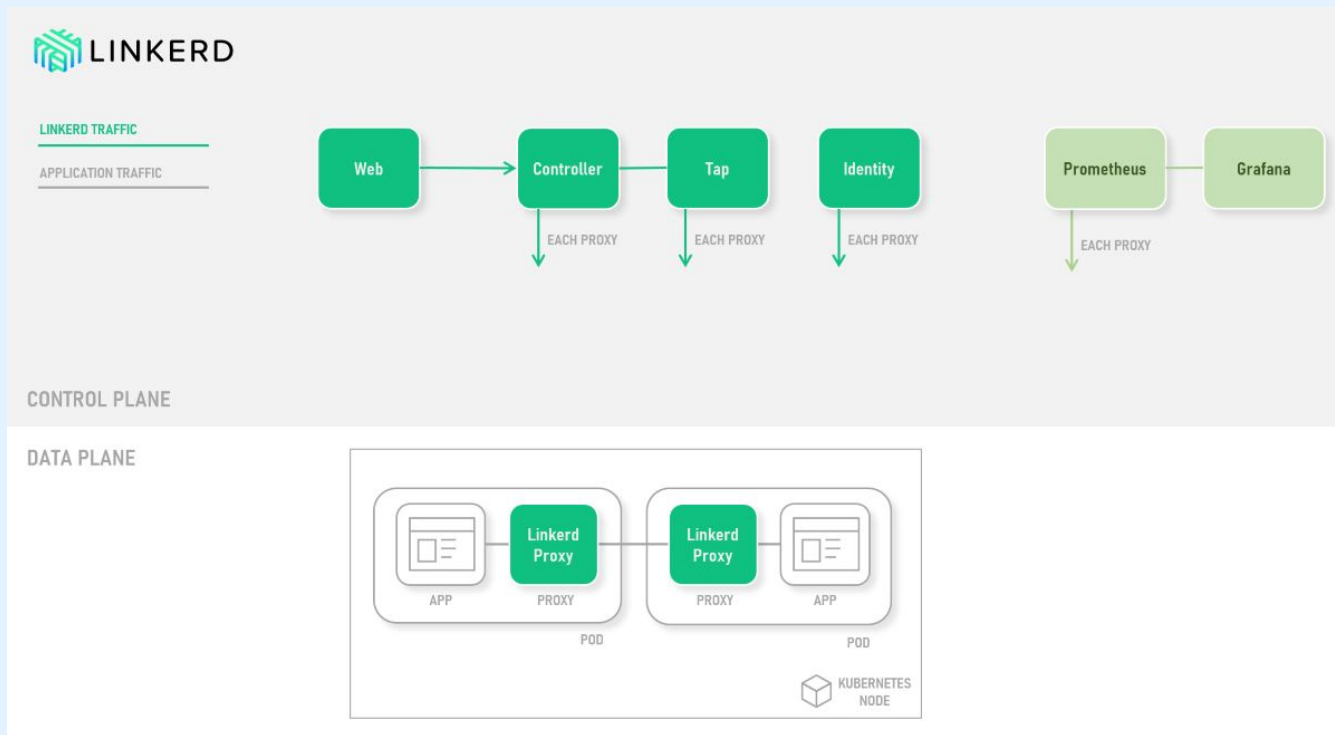
# Your application



# Your application + data plane



# Traffic flow in a meshed application

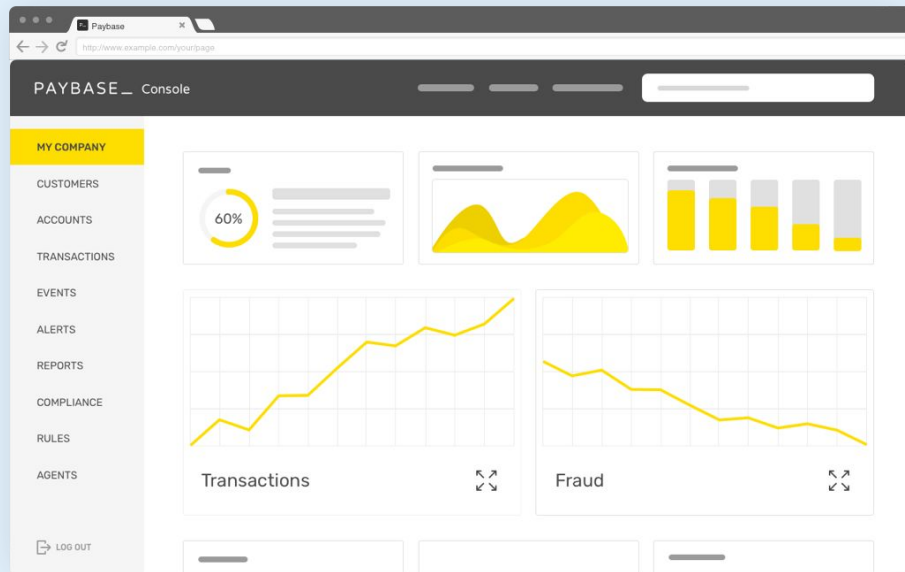


# Service meshes are awesome


- Automatic mTLS between your meshed services
- Telemetry and Monitoring
- Distributed Tracing
- HTTP, HTTP/2, and gRPC Proxying
- Latency-aware load balancing
- Retries and Timeouts
- TCP Proxying and Protocol Detection

# PAYBASE®

- API driven payments platform
- B2B – marketplace, gig/sharing economies, crypto, sophisticated payment flows
- Highly regulated





- 

LINKERD

Namespace > linkerd

CLUSTER

Namespaces

Control Plane

LINKERD ▾

WORKLOADS

Daemon Sets

Deployments

Jobs










Pods

Replication Controllers

Stateful Sets

CONFIGURATION

Deployments

Deployment ↑	↑ Meshed	↑ Success Rate	↑ RPS	↑ P50 Latency	↑ P95 Latency	↑ P99 Latency	Grafana
<a href="#">linkerd-controller</a>	1/1	100.00% <span>●</span>	1.03	750 ms	1.89 s	1.98 s	
<a href="#">linkerd-destination</a>	1/1	100.00% <span>●</span>	0.3	8 ms	18 ms	20 ms	
<a href="#">linkerd-grafana</a>	1/1	100.00% <span>●</span>	0.3	15 ms	925 ms	985 ms	
<a href="#">linkerd-identity</a>	1/1	100.00% <span>●</span>	0.3	25 ms	385 ms	397 ms	
<a href="#">linkerd-prometheus</a>	1/1	100.00% <span>●</span>	6.63	238 ms	883 ms	988 ms	
<a href="#">linkerd-proxy-injector</a>	1/1	100.00% <span>●</span>	0.28	3 ms	950 ms	990 ms	
<a href="#">linkerd-sp-validator</a>	1/1	100.00% <span>●</span>	0.3	4 ms	49 ms	50 ms	
<a href="#">linkerd-tap</a>	1/1	100.00% <span>●</span>	0.3	8 ms	19 ms	20 ms	
<a href="#">linkerd-web</a>	1/1	100.00% <span>●</span>	0.3	25 ms	39 ms	40 ms	

## Deployments

Deployment	Meshed	Success Rate	RPS	P50 Latency	P95 Latency	P99 Latency	Grafana
linkerd-controller	1/1	100.00% <span></span>	1.03	750 ms	1.89 s	1.98 s	<span></span>
linkerd-destination	1/1	100.00% <span></span>	0.3	8 ms	18 ms	20 ms	<span></span>
linkerd-grafana	1/1	100.00% <span></span>	0.3	15 ms	925 ms	985 ms	<span></span>
linkerd-identity	1/1	100.00% <span></span>	0.3	25 ms	385 ms	397 ms	<span></span>
linkerd-prometheus	1/1	100.00% <span></span>	6.63	238 ms	883 ms	988 ms	<span></span>
linkerd-proxy-injector	1/1	100.00% <span></span>	0.28	3 ms	950 ms	990 ms	<span></span>
linkerd-sp-validator	1/1	100.00% <span></span>	0.3	4 ms	49 ms	50 ms	<span></span>
linkerd-tap	1/1	100.00% <span></span>	0.3	8 ms	19 ms	20 ms	<span></span>
linkerd-web	1/1	100.00% <span></span>	0.3	25 ms	39 ms	40 ms	<span></span>

# What we expected vs what we got

```
{
  "x-token": "",
  "x-trace-id": "94b5687f-abcd3-efge5",
  "actorid": "agent/ef69e4de-be2d-4778",
  "actorrole": "role/c3cad570-b112-4273",
  "actorscopes": "api",
  "actorscopes": "actor1",
  "actorscopes": "actor2",
  "originatorid": "agent/ef69e4de-be2d-4778",
  "originatorrole": "role/c3cad570-b112-4273",
  "originatorsscopes": "api.foo.bar/075dd0a2-87fc-4406",
  "originatorsscopes": "api",
  "originatorsscopes": "originator1",
  "originatorsscopes": "originator2",
  "supplierid": "supplier/bbbbbbb-aaaa-1aaa",
  "roles": "role/c3cad570-b112-4273",
  "roles": "role/c3cad570-b112-4273",
  "te": "trailers",
  "content-type": "application/grpc",
  "user-agent": "grpc-node/1.20.3 grpc-c/7.0.0",
  "grpc-accept-encoding": "identity,deflate,gzip",
  "accept-encoding": "identity,gzip",
  "grpc-timeout": "30S",
  "l5d-dst-canonical": "service.default.svc.cluster.local:9090"
}
```

```
{
  "x-token": "",
  "x-trace-id": "94b5687f-abcd3-efge5",
  "actorid": "agent/ef69e4de-be2d-4778",
  "actorrole": "role/c3cad570-b112-4273",
  "actorscopes": [
    "api",
    "actor1",
    "actor2"
  ],
  "originatorid": "agent/ef69e4de-be2d-4778",
  "originatorrole": "role/c3cad570-b112-4273",
  "originatorsscopes": "api.foo.bar/075dd0a2-87fc-4406",
  "supplierid": [
    "supplier/bbbbbbb-aaaa-1aaa"
  ],
  "roles": [
    "role/c3cad570-b112-4273",
    "role/c3cad570-b112-4273"
  ],
  "user-agent": "grpc-node/1.20.3 grpc-c/7.0.0",
  "l5d-dst-canonical": "service.default.svc.cluster.local:9090"
}
```

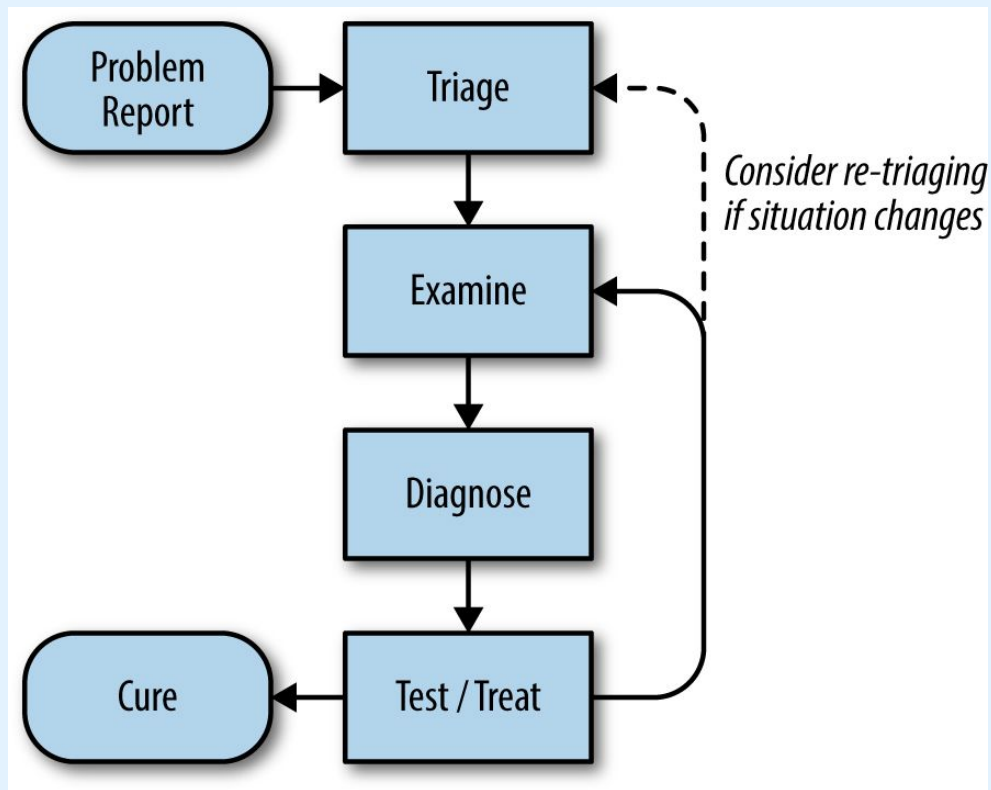


# A framework for troubleshooting

**"Be warned that being an expert is more than understanding how  
a system is supposed to work. Expertise is gained by  
investigating why a system doesn't work."**

Brian Redman



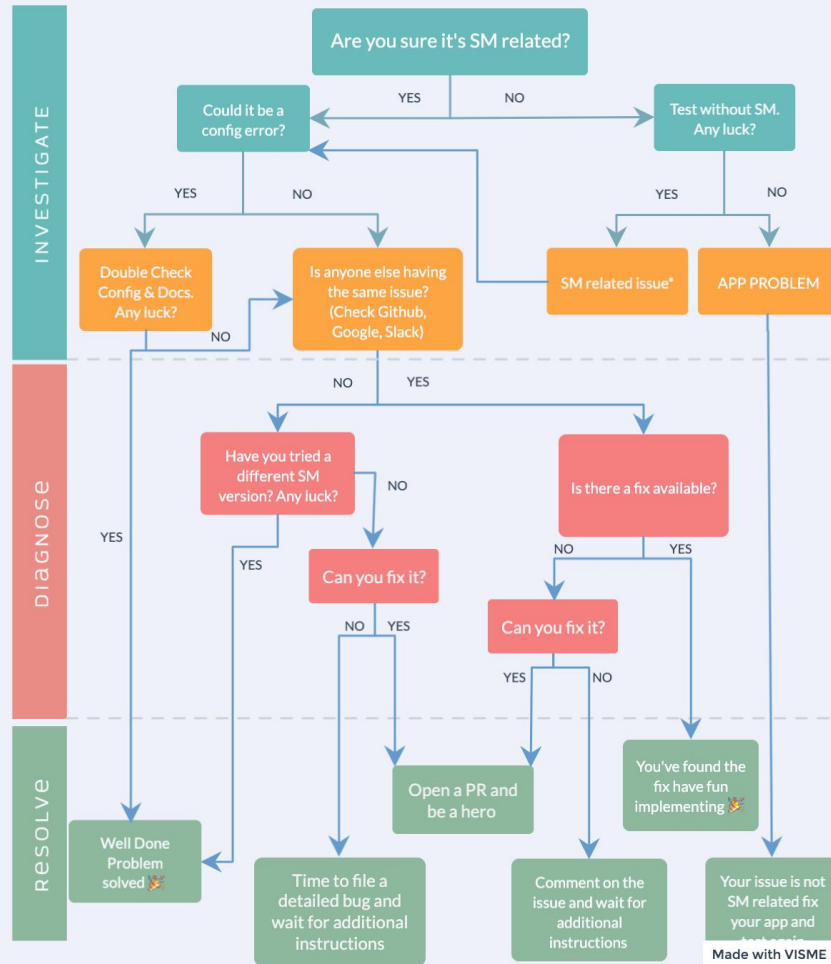


*Site Reliability Engineering: How Google Runs Production Systems, Chapter 12*

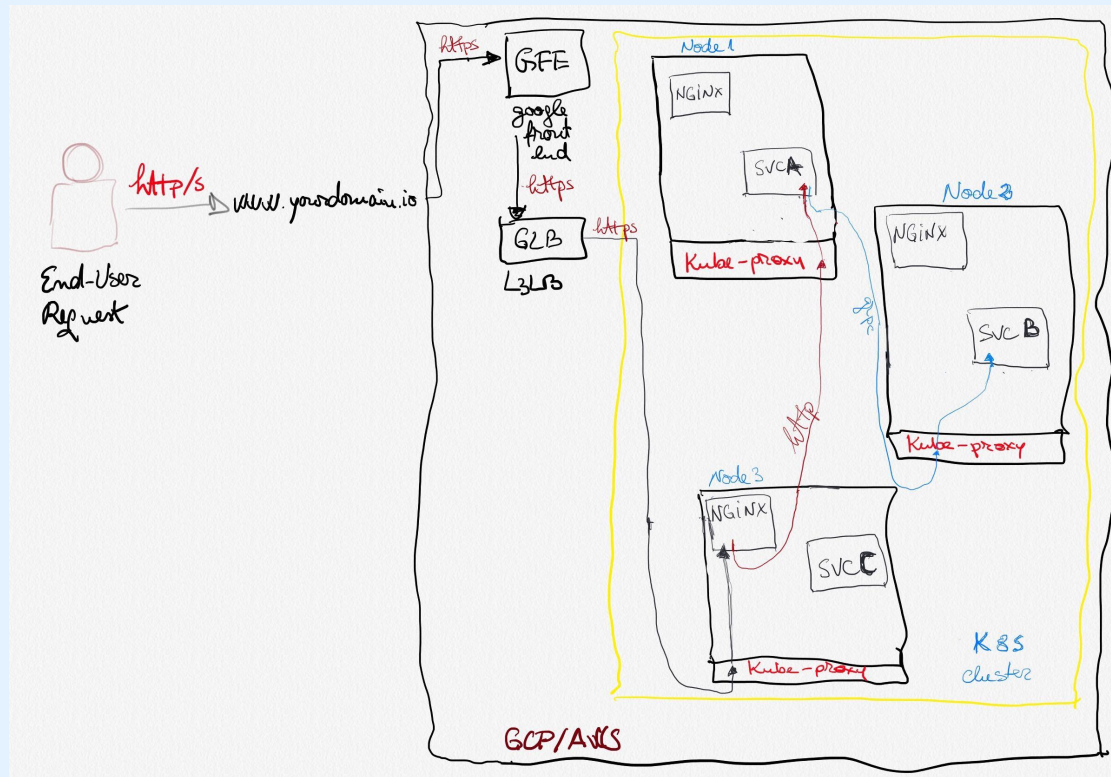


# SERVICE MESH ERRORS?

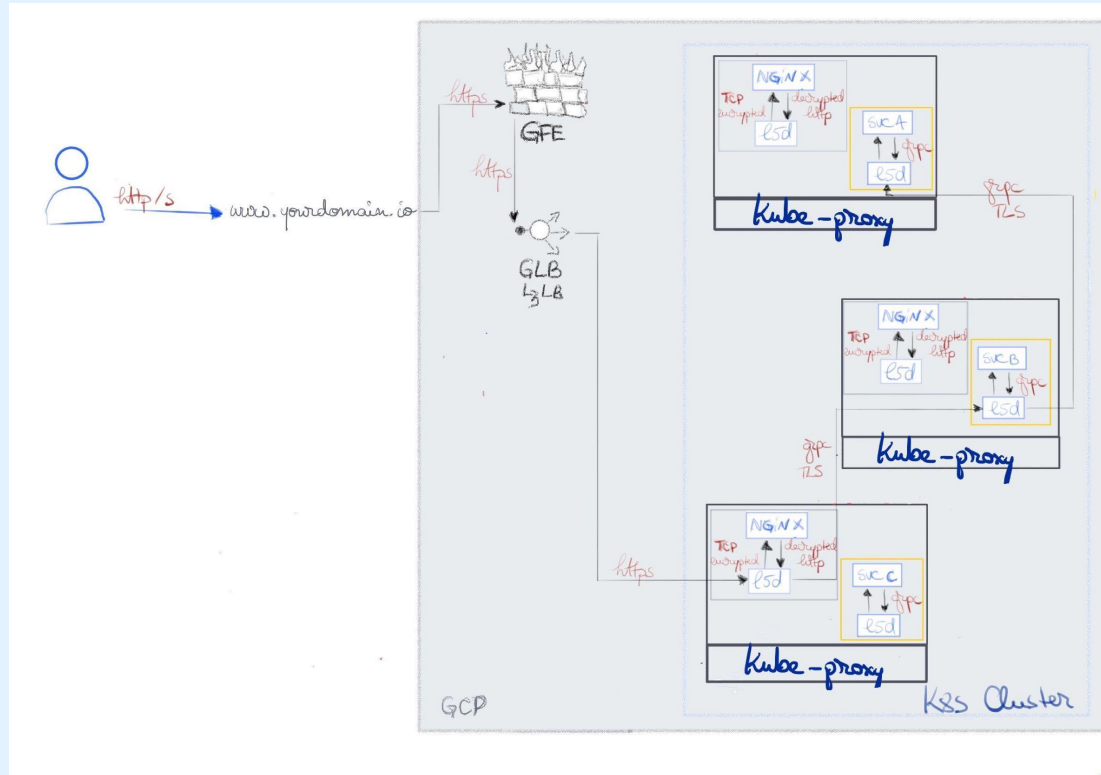
This is the Flow Chart You're Looking For



# Diagnose: System without Linkerd



# Diagnose: System with Linkerd




# Problem Report: Filing an Issue

- Follow the recommendations
- Attach log outputs
- Don't assume people know your system – be as detailed as possible
- Include what you've already tried
- This is universal advice any time to ask For external help (regardless of medium)

**Issue:** 🐛 Bug Report

If something isn't working as expected 🤔. If this doesn't look right, [choose a different type](#).



Related Issues [Beta](#) [Try it.](#)

---

### Bug Report

**What is the issue?**

**How can it be reproduced?**

**Logs, error output, etc**

(If the output is long, please create a [gist](#) and paste the link here.)

`linkerd check output`


`your output here ...`

---

### Environment

- Kubernetes Version:
- Cluster Environment: (GKE, AKS, kops, ...)
- Host OS:
- Linkerd version:

# Triage

 [linkerd](#) / [linkerd2](#)

👁 Unwatch ▾ 175

★ Star 4.8k

🍴 Fork 448

<> Code

🔔 Issues 279

🔗 Pull requests 21

🚀 Actions

📁 Projects 0

📖 Wiki

🛡 Security

📊 Insights

Filters ▾

🔍 is:issue is:open label:bug

🏷 Labels 56

📅 Milestones 0

New issue

✕ Clear current search query, filters, and sorts

☐ 📌 24 Open ✓ 181 Closed

Author ▾


Labels ▾

Projects ▾

Milestones ▾

Assignee ▾


Sort ▾

☐ 📌 🚨 **Weird behavior of the `Run Linkerd Check` button** [area/web](#) [bug](#) 


#3703 opened 23 hours ago by alpeb

☐ 📌 🚨 **Mutatingwebhookconfiguration does not retry after all worker nodes terminate.** [bug](#)


#3688 opened 6 days ago by danpalmer-niceincontact

☐ 📌 🚨 **Problems with HA on lost node** [bug](#) 


#3606 opened 23 days ago by falcoriss

☐ 📌 🚨 **HTTP/1.1 outbound external call loses response -- response never received by sending container** [area/proxy](#) [bug](#) [needs/repro](#) [priority/P0](#) 

#3605 opened 25 days ago by rocketrman

☐ 📌 🚨 **Linkerd controller becomes inaccessible from the proxies** [bug](#) [needs/repro](#) 

#3599 opened 27 days ago by cpretzer

☐ 📌 🚨 **502 Bad Gateway from proxy on retry after timeout** [bug](#) [needs/repro](#) 

#3596 opened 27 days ago by rocketrman

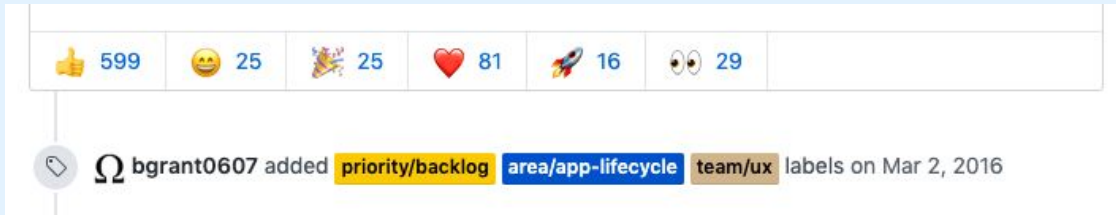


# Triage: Reproducibility

- Is the problem clearly stated?
- Are there enough details to reproduce the bug?
- What is the smallest reproducible test case?
- Can we reproduce it without having your whole architecture in play?

# Triage: Impact

- How bad is the bug?
  - Does it impact security? Usability?
- Are multiple users experiencing this bug?
  - If a user files a good bug report, other users can comment if they are also experiencing it!



# Diagnosing the Bug

Where could the bug be?

Is it...

- in the application?
- in the application's dependencies?
- in the Linkerd control plane? (Golang)
- in the Golang dependencies?
- in the Linkerd-proxy? (Rust)
- in the Rust dependencies?
- in Kubernetes?

# Diagnosing the Bug

- The initial bug report contained proxy and application logs
  - Application logs (`kubectl logs -f deploy/foo -n bar`)
  - Proxy logs (`linkerd logs`)
  - There were protocol errors on requests that had gRPC metadata in the headers
- We asked for further detail: `linkerd tap`
  - Examined the requests between services in the application
- We dived even deeper: `tcpdump`
  - Looked at the raw TCP packets
  - Saw that headers were being split across two frames
  - This is unusual, as headers typically only take up one frame

# Understanding the Bug(s)!



HTTP/2 in the Linkerd service mesh



HTTP/2 frame types and header compression



# Understanding the Bug(s)!

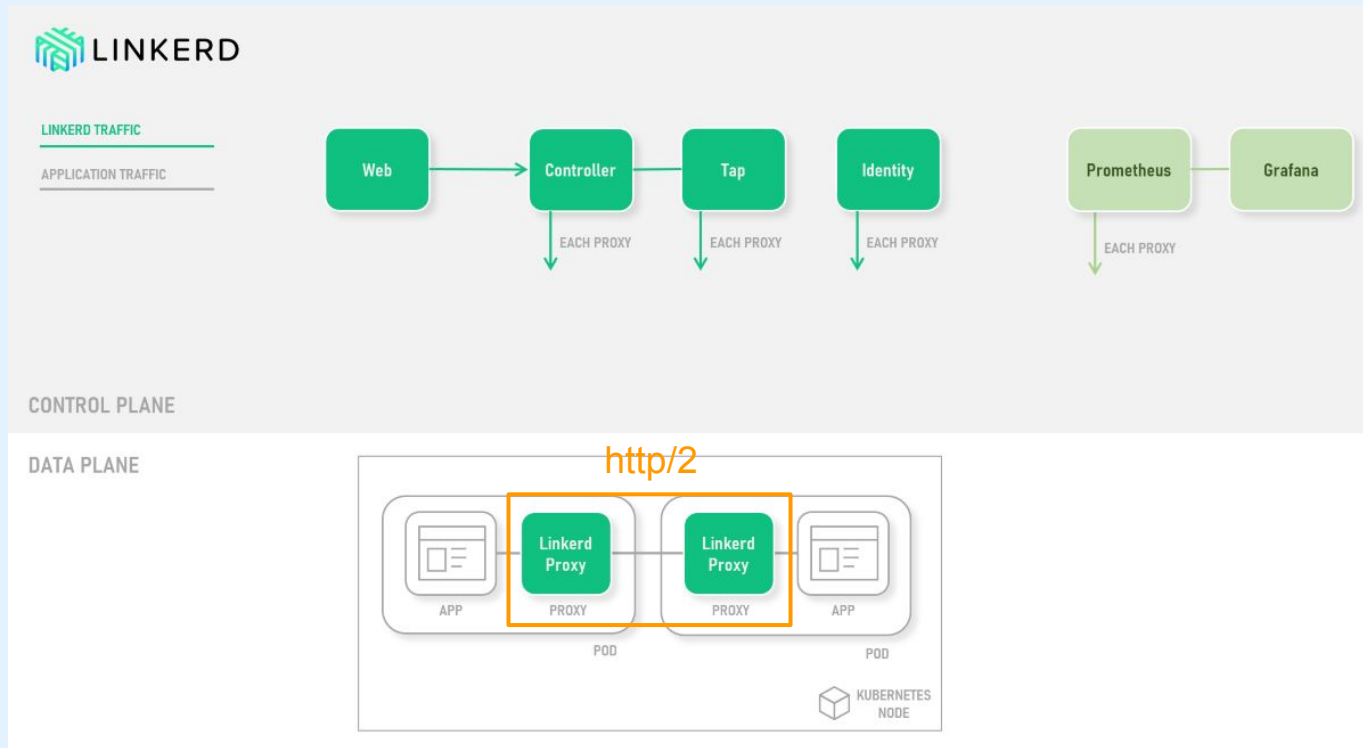


HTTP/2 in the Linkerd service mesh



HTTP/2 frame types and header compression

# HTTP/2 in the Linkerd service mesh



# Understanding the Bug(s)!

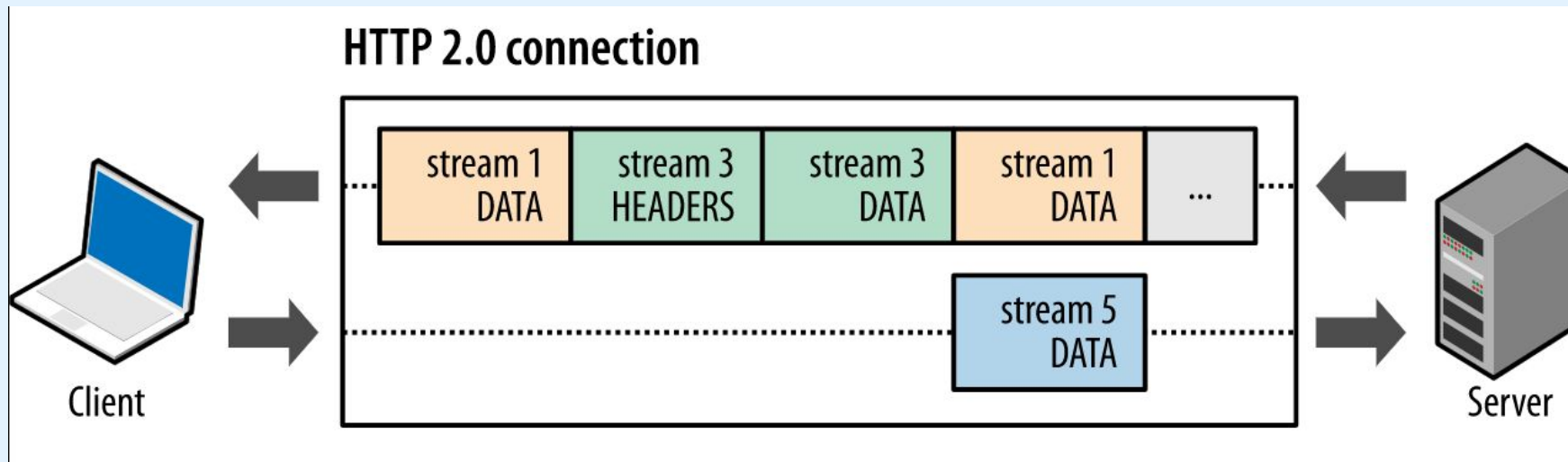


HTTP/2 in the Linkerd service mesh

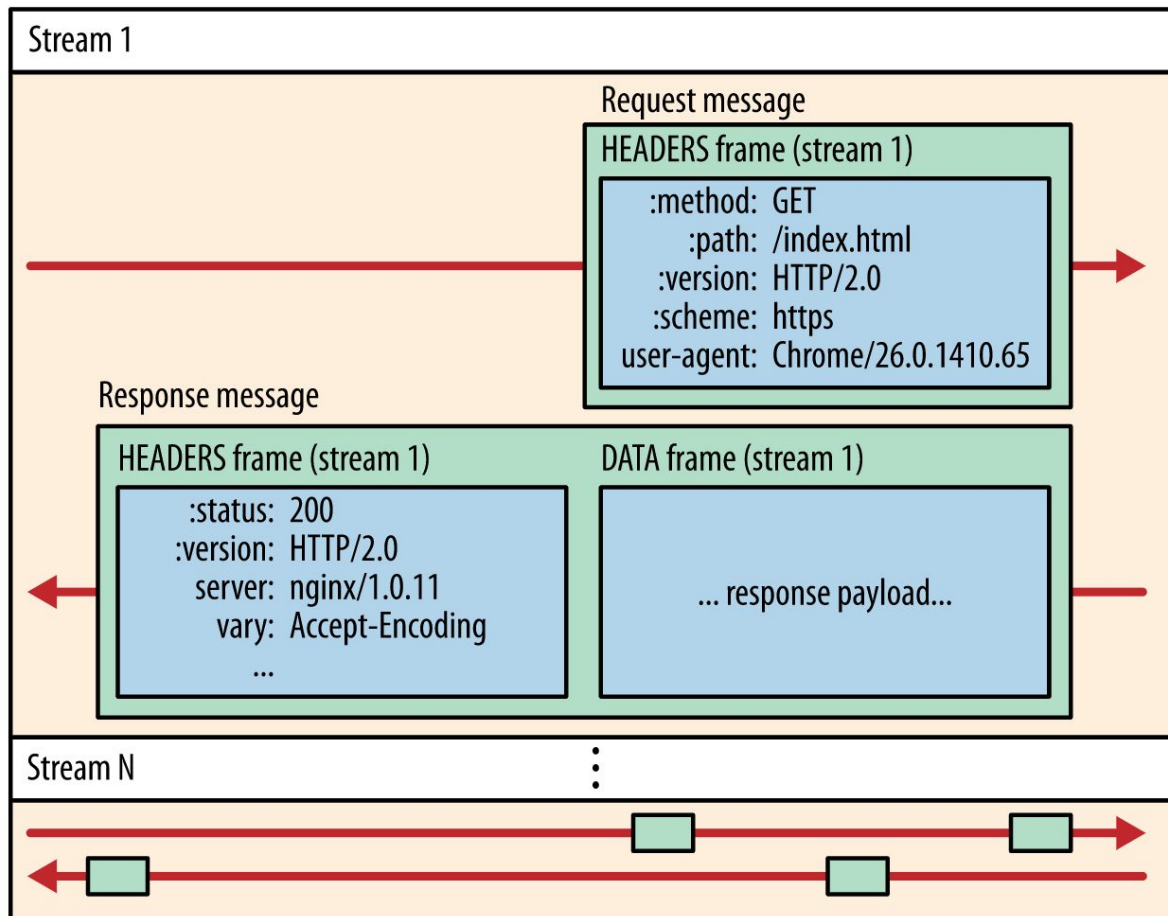


**HTTP/2 frame types and header compression**

# HTTP/2: multiplexing, frame types



## Connection



Request #1

:method	GET
:scheme	https
:host	example.com
:path	/resource
accept	image/jpeg
user-agent	Mozilla/5.0 ...

HEADERS frame (Stream 1)

```
:method: GET
:scheme: https
:host: example.com
:path: /resource
accept: image/jpeg
user-agent: Mozilla/5.0 ...
```

Request #2

:method	GET
:scheme	https
:host	example.com
:path	/new_resource
accept	image/jpeg
user-agent	Mozilla/5.0 ...

HEADERS frame (Stream 3)

```
:path: /new_resource
```

# Bug #1: Continuation Frame Panic

The code panicked when a CONTINUATION frame contained a repeated header

src/hpack/encoder.rs

```
...    @@ -103,10 +105,9 @@ impl Encoder {  
103    105                dst.truncate(len);  
104    106                return Encode::Partial(resume);  
105    107            }  
108    +                last_index = Some(resume.index);
```



**seanmonstar** on May 14

Author

Member



This fixes the first issue.



Reply...



## Bug #2: Evicted Table Header Index

We were looking up repeated headers using the wrong index

src/hpack/table.rs

```
...    @@ -125,7 +125,7 @@ impl Table {
125    125        Indexed(idx, ..) => idx,
126    126        Name(idx, ..) => idx,
127    127        Inserted(idx) => idx + DYN_OFFSET,
128    -        InsertedValue(idx, _) => idx,
128    +        InsertedValue(_name_idx, slot_idx) => slot_idx + DYN_OFFSET,
```



seanmonstar on May 14

Author

Member

+ 😊 ...

This makes the next header (that has no name, so using this index) point at the new index, not at the old name index, which may have been evicted.



Reply...

# The bugs are fixed!



**calinah** commented on Jul 9

Author



@siggy @seanmonstar @olix0r I am super pleased to confirm that the changes you guys have rolled out have fixed the 502 issues. I've solved our internal issue and have managed to test multiple times with `edge-19.7.1` and once with `edge-19.7.2` . Thank you for your assistance 🙏



4

# Aftermath:

## Linkerd diagnostic improvements!

- A debug sidecar container
  - Deploy the container into a failing pod to diagnose problems
  - The debug image contains `tshark`, `tcpdump`, `lsof`, and `iproute2`
  - Once installed, it automatically logs all traffic with `tshark`
- More visibility into application traffic with `linkerd tap`
  - Can now also view request bodies
- Tracing in the Rust libraries
  - Increased visibility into the libraries we depend on

# Useful commands for troubleshooting

```
# increase log_level &/or skip inbound/outbound ports

$ kubectl get deploy -o yaml | linkerd inject --proxy-log-level=debug,linkerd2_proxy=debug --skip-outbound-ports 8529,5432,8015 --skip-inbound-ports 8529,5432,8015 - | kubectl apply -f -

# enable linkerd debug container

$ kubectl get deploy $deploy_name -o yaml | linkerd inject --enable-debug-sidecar --manual - | kubectl apply -f -

# run tcpdump in debug container

$ kubectl exec -it pod_name linkerd-debug -- tcpdump -i any -s 65535 -w out.pcap

# listen to traffic stream for a resource

$ linkerd tap deploy/web

# display pods (CPU/Memory/Storage) usage

$ kubectl top pods
```

# Summary

There was more than one bug! The bugs were deep in the stack!

All got fixed fairly quick due to:

- Detailed bug reports
- Space to test with/without linkerd , different versions
- Used those log suggestions
- Looked through code / other bugs



+

PAYBASE<sup>®</sup>\_

Linkerd has a  
**friendly, welcoming**  
community! Join us!

Slides at  
**[bit.ly/bug-in-my-mesh](https://bit.ly/bug-in-my-mesh)**

 [github.com/linkerd](https://github.com/linkerd)

 [slack.linkerd.io](https://slack.linkerd.io)

 [@linkerd](https://twitter.com/linkerd)

FROM YOUR FRIENDS AT



[bit.ly/bug-in-my-mesh](https://bit.ly/bug-in-my-mesh)



# References

<https://developers.google.com/web/fundamentals/performance/http2>

<https://http2.github.io/http2-spec/#FrameHeader>

<https://tools.ietf.org/html/rfc7541>

<https://linkerd.io/2/tasks/using-the-debug-container>

<https://buoyant.io/2017/04/25/whats-a-service-mesh-and-why-do-i-need-one>

<https://linkerd.io/2/reference/architecture>

<https://docs.microsoft.com/en-us/azure/aks/servicemesh-linkerd-about>