# Microservice Forensics

**Honest Status Page**
@honest_update

Follow

We replaced our monolith with micro services so that every outage could be more like a murder mystery.

4:10 PM - 7 Oct 2015

# Thomas Rampelberg

**Software Engineer @ Buoyant**

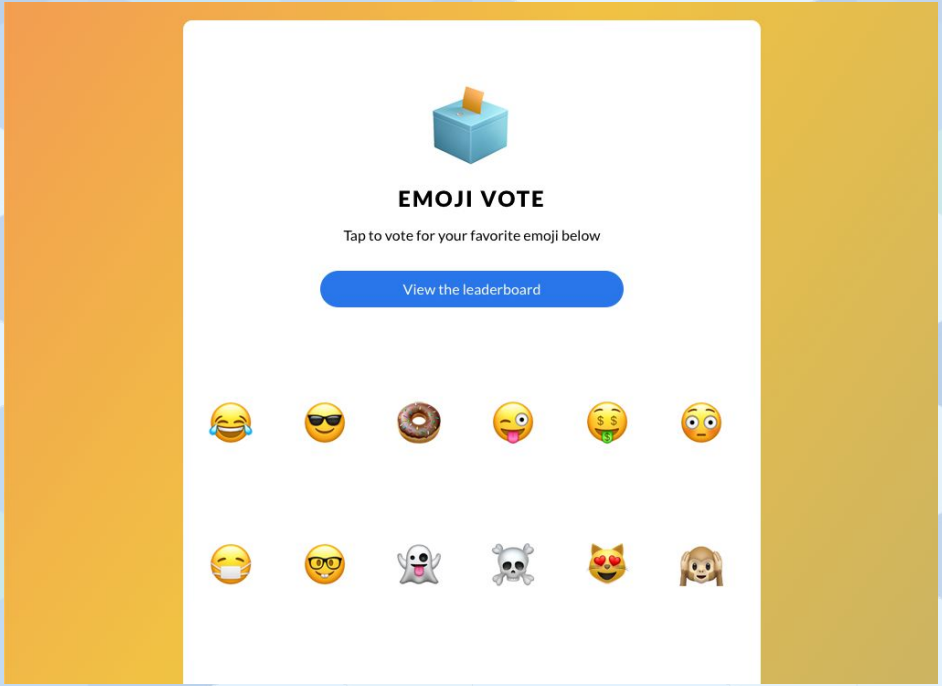

@grampelberg

@grampelberg

@grampelberg

# kubernetes

# The Scene of the Crime




https://emoji.l5d.io

HELLO? CAN ANYBODY HEAR ME?

# Downtime

- External monitoring
- livenessProbe/readinessProbe
- kube-state-metrics

Pingdom: https://www.pingdom.com/
Configuring livenessProbe: https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-probes/
Kube-state-metrics: https://github.com/helm/charts/tree/master/stable/kube-state-metrics

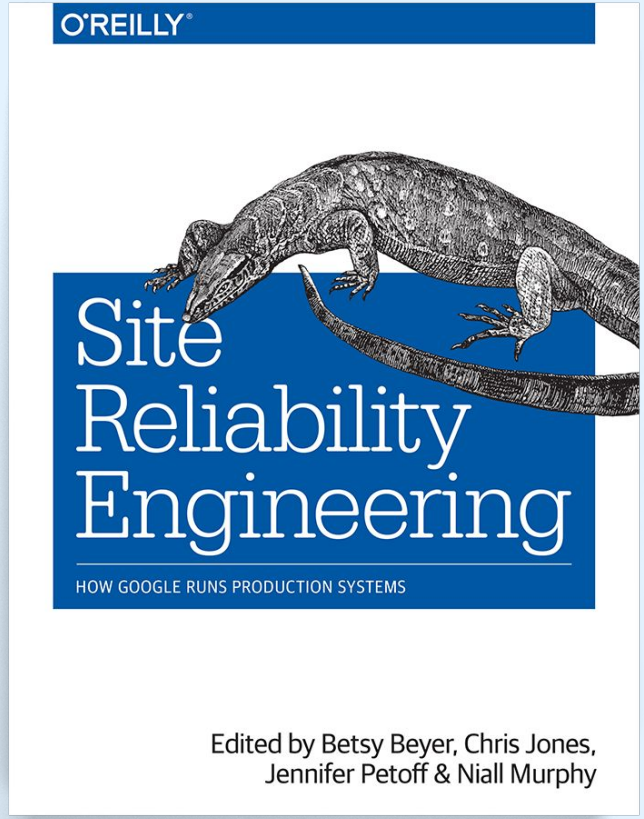
# Exceptions

```go
func readIntoBytes(filename string) ([]byte, error) {
  file, err := static.Templates.Open(filename)
  if err != nil {
    return nil, err
  }
  defer file.Close()

  buf := new(bytes.Buffer)
  buf.ReadFrom(file)

  return buf.Bytes(), nil
}
```

Sentry: https://docs.sentry.io/clients/

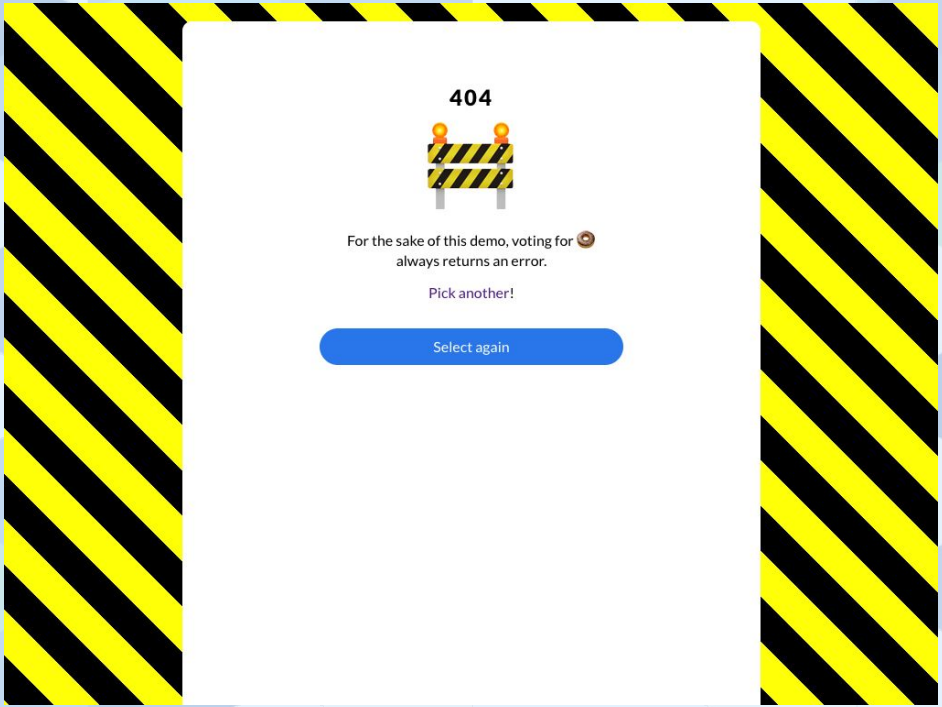
# Traffic Monitoring

- Latency
- Traffic
- Errors

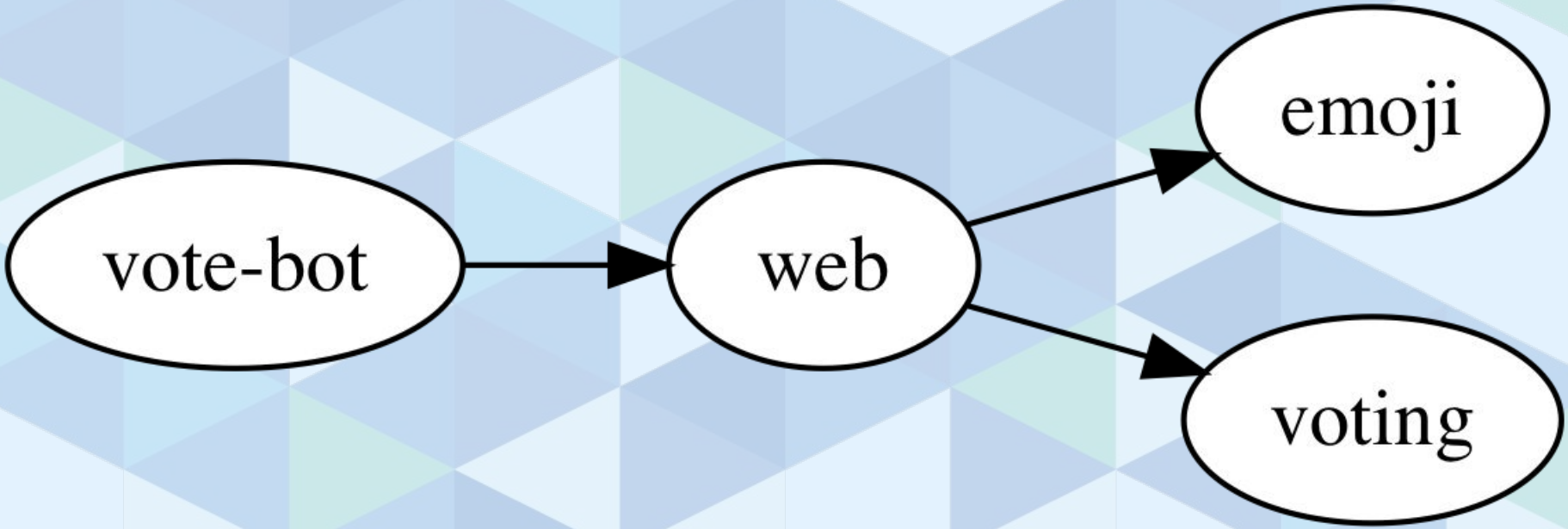



http://bit.ly/golden-signals

# The Murder


404
For the sake of this demo, voting for 🍩 always returns an error.
Pick another!
Select again

# The Suspects

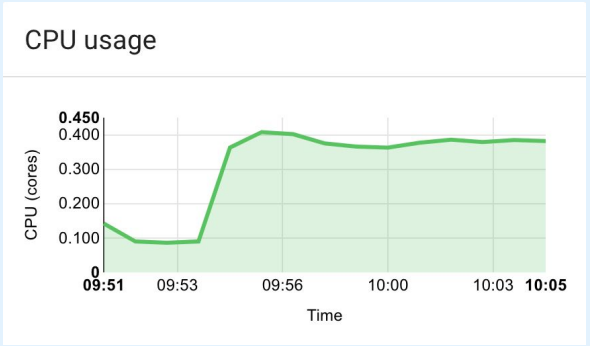


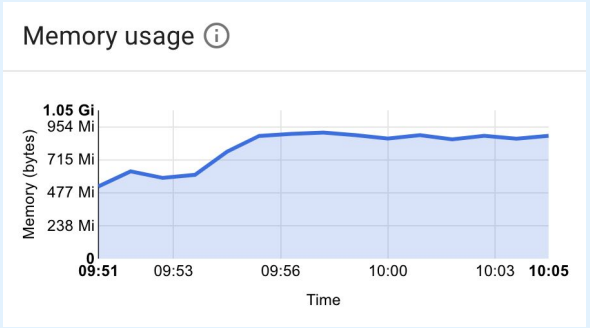
# Forensics

Created by Daouna Jeong
from Noun Project

# System Monitoring

- CPU
- Memory
- Load Average
- Open Files
- Network



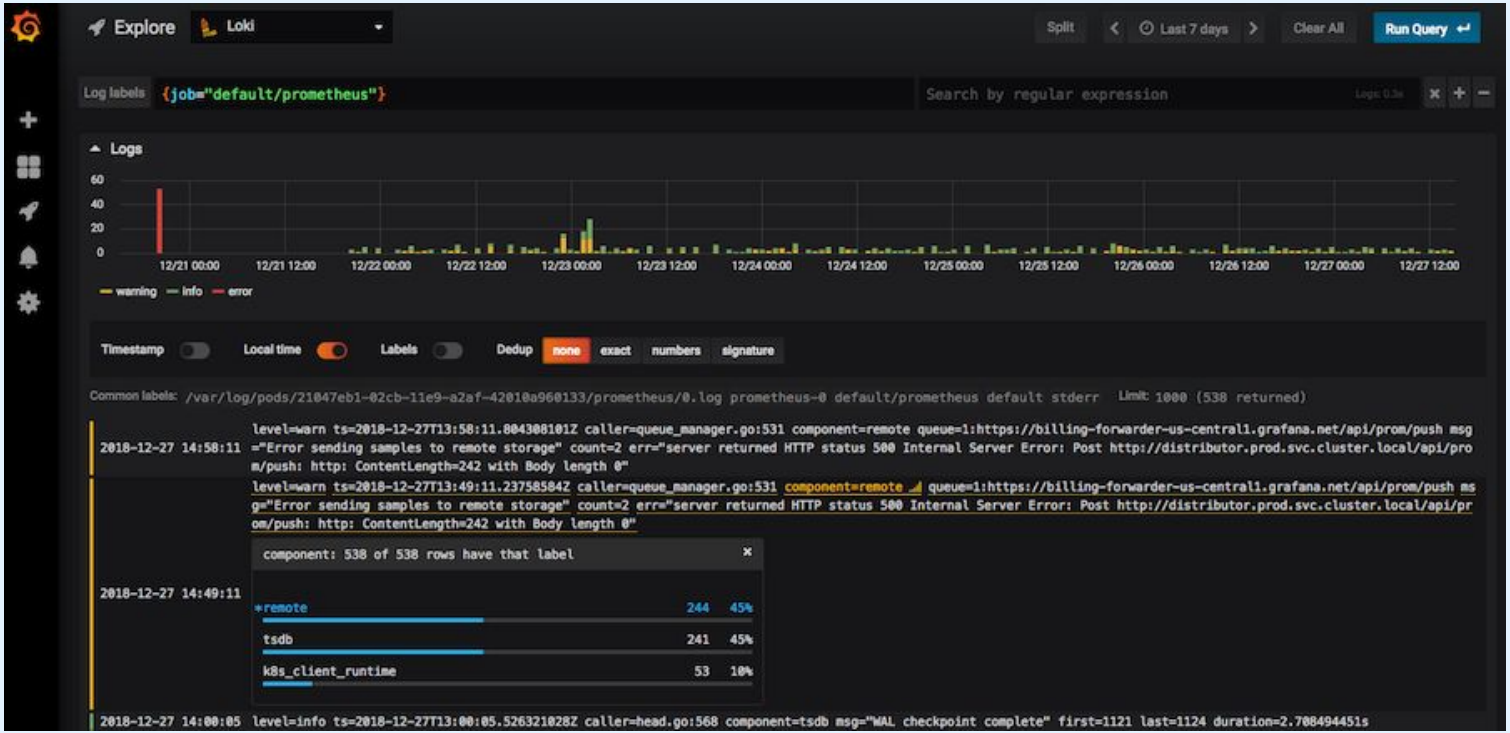
# Application Instrumentation

```
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 1.9864e-05
go_gc_duration_seconds{quantile="0.25"} 5.2299e-05
go_gc_duration_seconds{quantile="0.5"} 7.1593e-05
go_gc_duration_seconds{quantile="0.75"} 0.000151335
go_gc_duration_seconds{quantile="1"} 0.003560545
go_gc_duration_seconds_sum 0.058749902
go_gc_duration_seconds_count 254
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 59
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.10.3"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 6.619704e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if fr
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 1.465679856e+09
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling buc
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.50257e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 6.977458e+06
# HELP go_memstats_gc_cpu_fraction The fraction of this program's available CPU
e the program started.
# TYPE go_memstats_gc_cpu_fraction gauge
go_memstats_gc_cpu_fraction 3.924477433282826e-05
```

```
func main() {
        recordMetrics()

        http.Handle("/metrics", promhttp.Handler())
        http.ListenAndServe(":2112", nil)
}
```

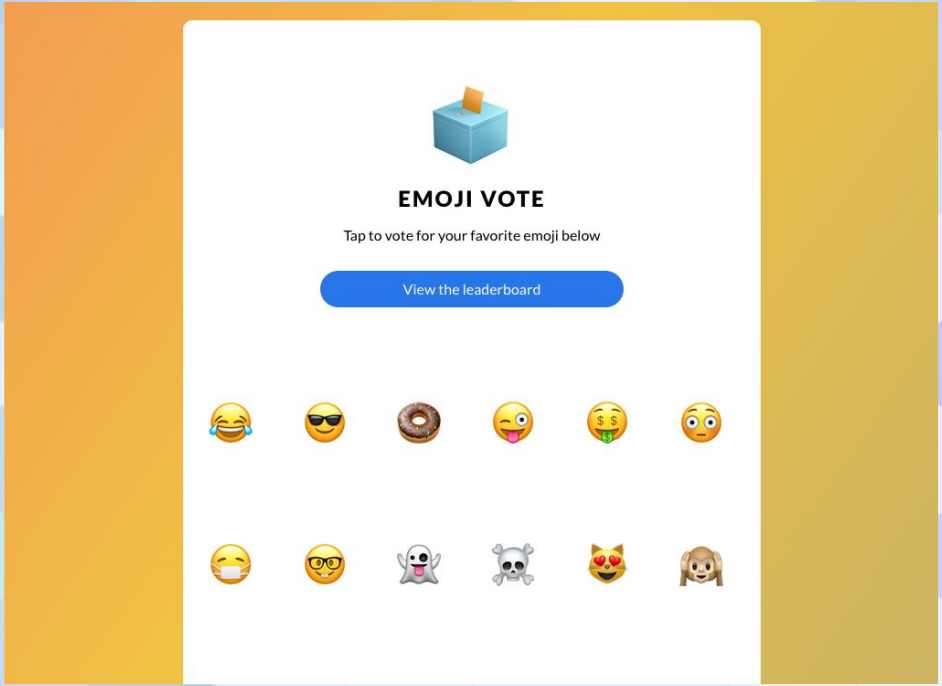
client_golang: https://prometheus.io/docs/guides/go-application/#adding-your-own-metrics

# Log Aggregation




Loki: https://grafana.com/loki

# Get your votes in!



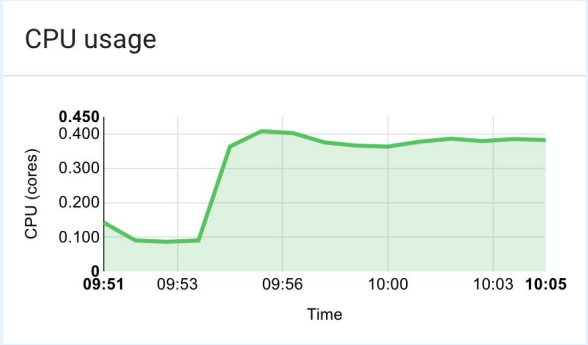

https://emoji.l5d.io

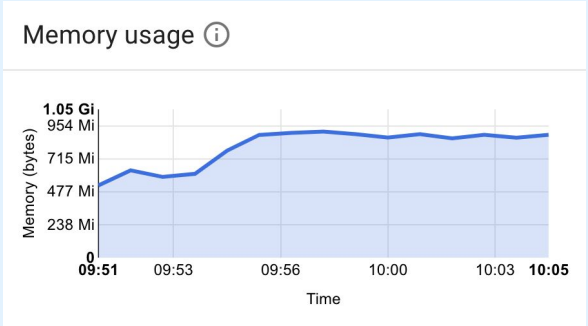
# 🙋 Question the Suspects!

- ☐ Service running?
- ☐ Health checks passing?
- ☐ Logs look okay?

# Nothing out of the ordinary, constable

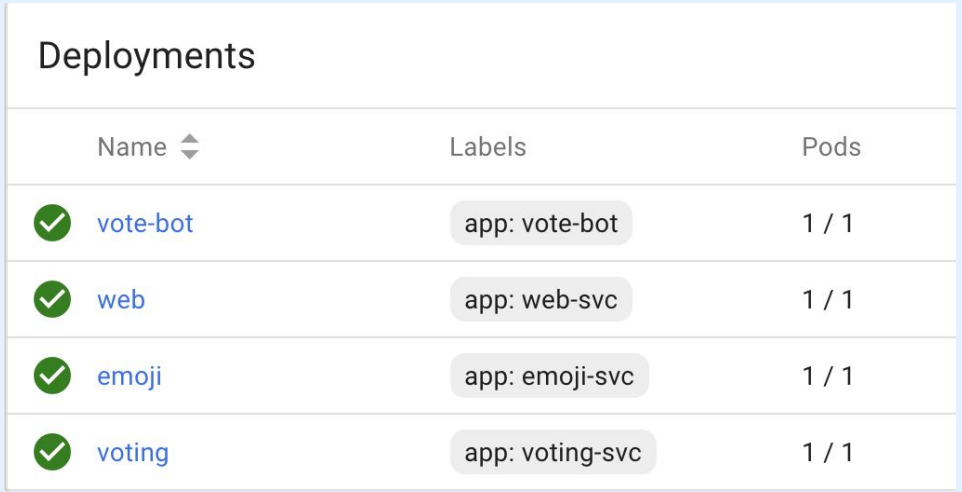
- [x] Service running!
- [ ] Health checks passing?
- [ ] Logs look okay?



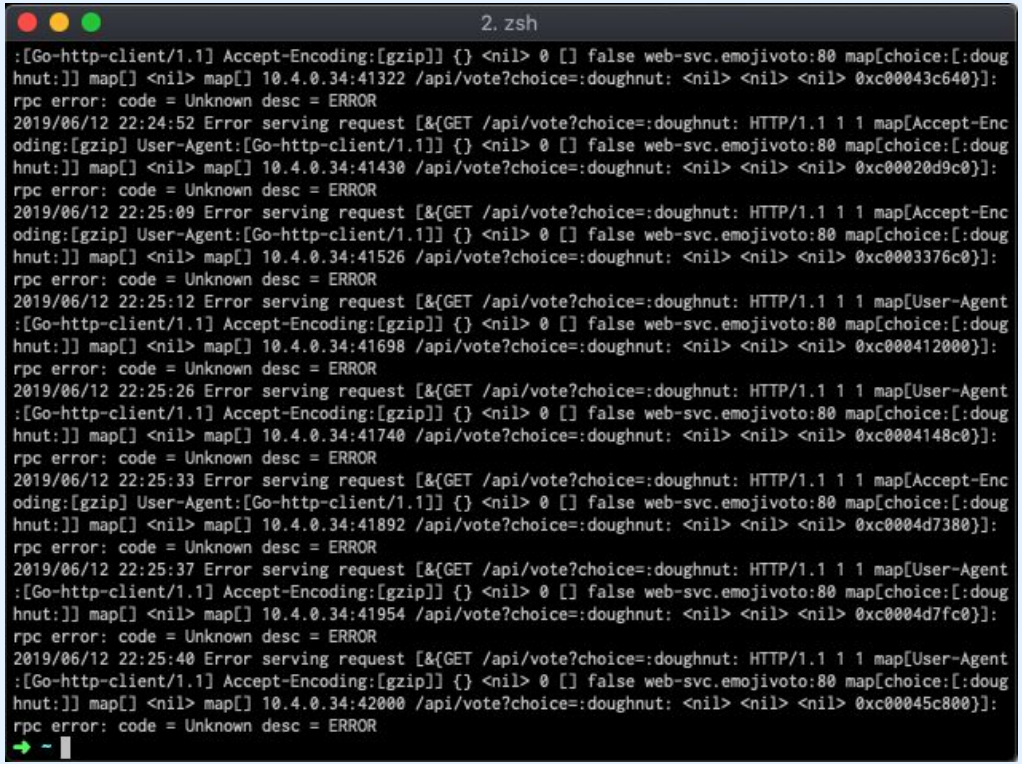
# Everything is fine

- [x] Service running!
- [x] Health checks passing!
- [ ] Logs look okay?



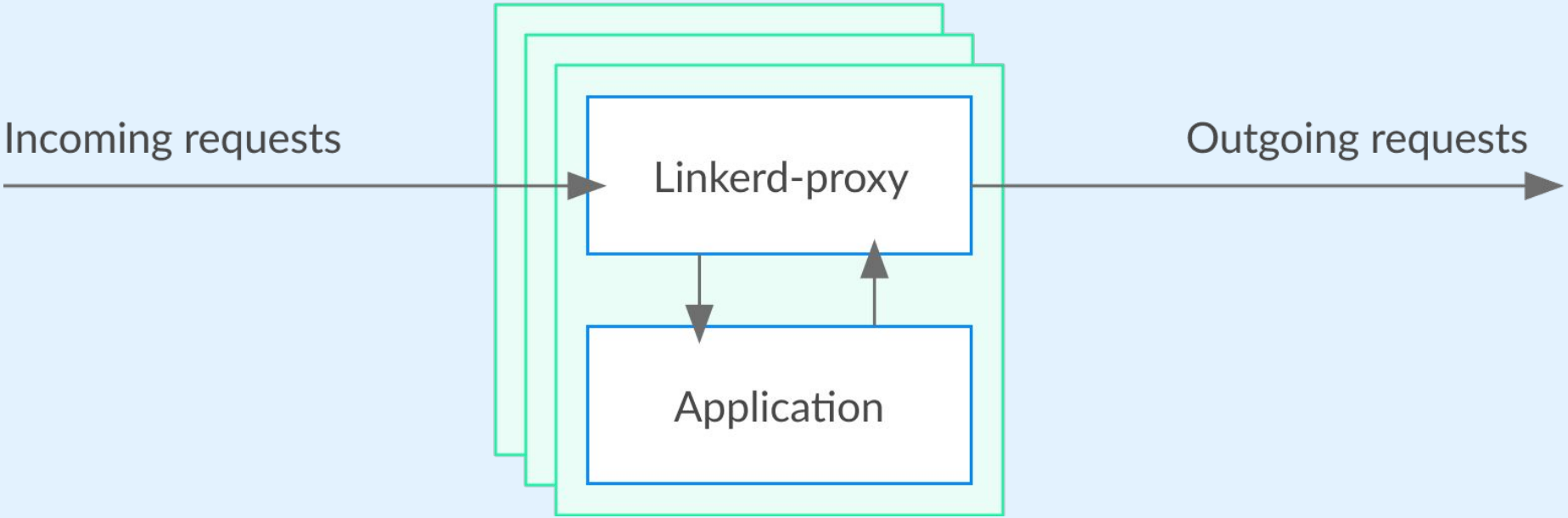
# 🤷 Okay, *something* is fishy

- ✅ Service running!
- ✅ Health checks passing!
- ⁉️ Logs look … something?



# Service Mesh

# What is a service mesh?



# Metrics everywhere!

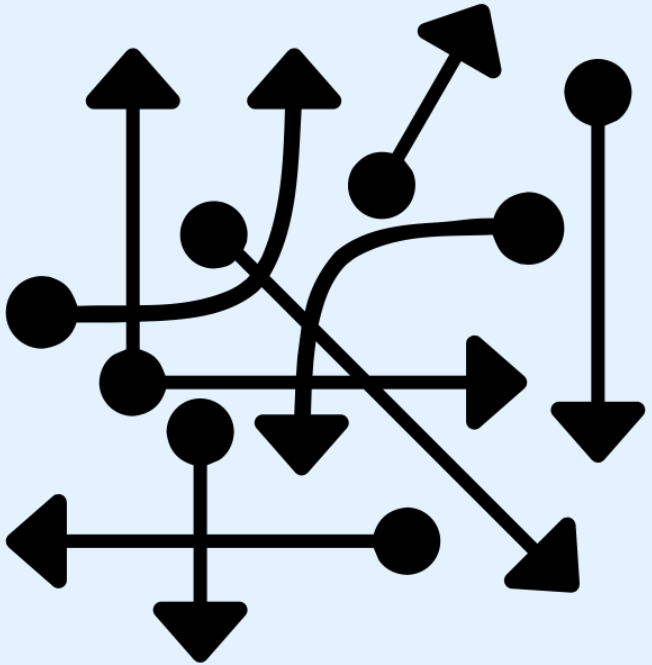
2. zsh

```
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="10"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="20"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="30"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="40"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="50"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="100"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="200"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="300"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="400"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="500"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="1000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="2000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="3000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="4000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="5000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="10000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="20000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="30000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="40000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="50000"} 5
control_response_latency_ms_bucket{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200",le="+Inf"} 5
control_response_latency_ms_count{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200"} 5
control_response_latency_ms_sum{addr="linkerd-destination.linkerd.svc.cluster.local:8086",tls="true",status_code="200"} 7
# HELP control_response_total Total count of HTTP responses.
# TYPE control_response_total counter
control_response_total{addr="linkerd-identity.linkerd.svc.cluster.local:8080",tls="true",status_code="200",classification="success"} 1
# HELP control_retry_skipped_total Total count of retryable HTTP responses that were not retried.
# TYPE control_retry_skipped_total counter
# HELP process_start_time_seconds Time that the process started (in seconds since the UNIX epoch)
# TYPE process_start_time_seconds gauge
process_start_time_seconds 1560378885
# HELP process_cpu_seconds_total Total user and system CPU time spent in seconds.
# TYPE process_cpu_seconds_total counter
process_cpu_seconds_total 0
# HELP process_open_fds Number of open file descriptors.
# TYPE process_open_fds gauge
process_open_fds 26
# HELP process_max_fds Maximum number of open file descriptors.
# TYPE process_max_fds gauge
process_max_fds 1048576
# HELP process_virtual_memory_bytes Virtual memory size in bytes.
# TYPE process_virtual_memory_bytes gauge
process_virtual_memory_bytes 97001472
# HELP process_resident_memory_bytes Resident memory size in bytes.
# TYPE process_resident_memory_bytes gauge
process_resident_memory_bytes 12263424
➜ ~
```

# Yesterday's developer

| Business Logic |
| --- |
| Testing |
| Monitoring |
| Security |

Developer Responsibility



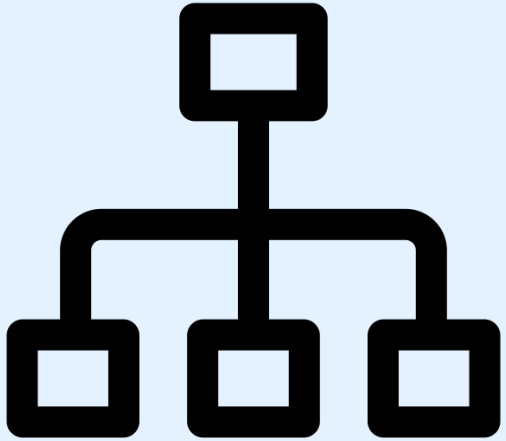
Created by Anna Sophie from Noun Project

# Tomorrow’s developer

| |
|---|
| Business Logic |
| Testing |
| Monitoring |
| Security |

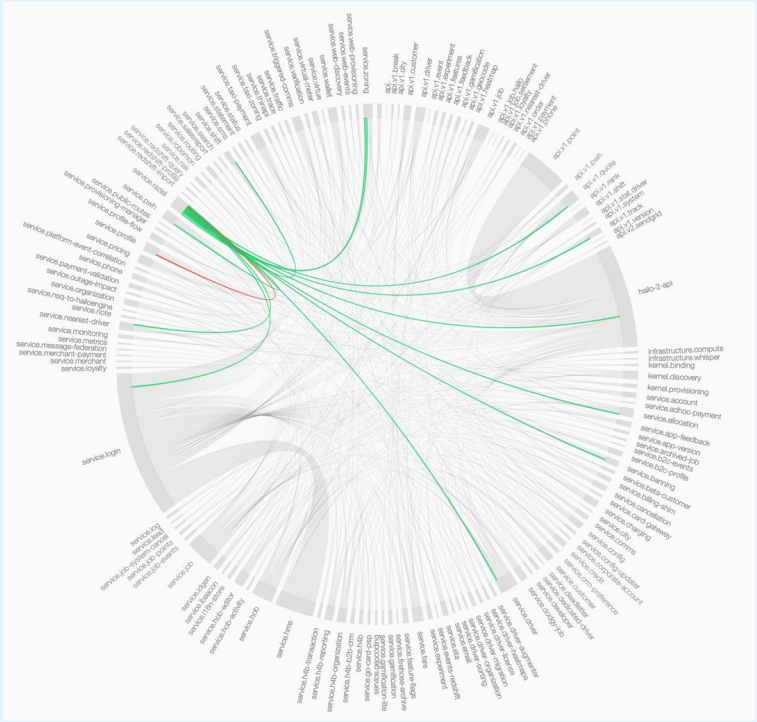
Developer Responsibility

SRE Responsibility

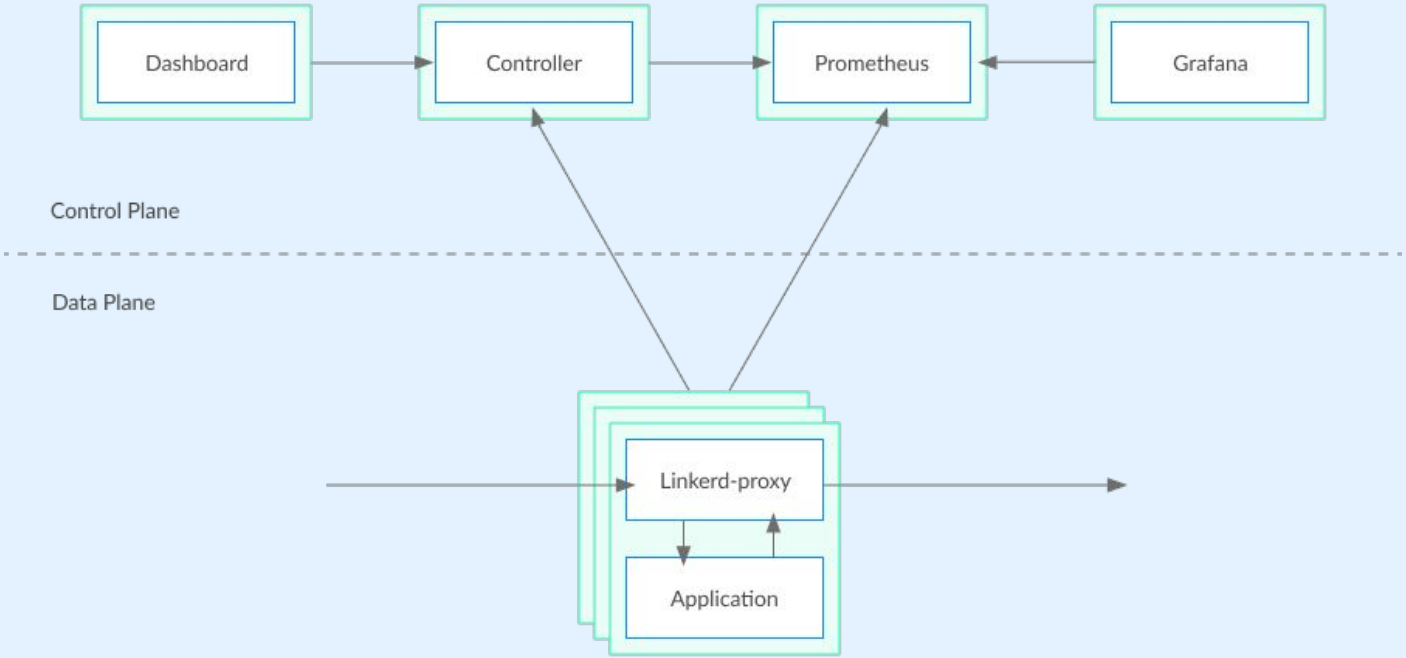


Created by kinsley from Noun Project

# Traffic Monitoring



# Architecture



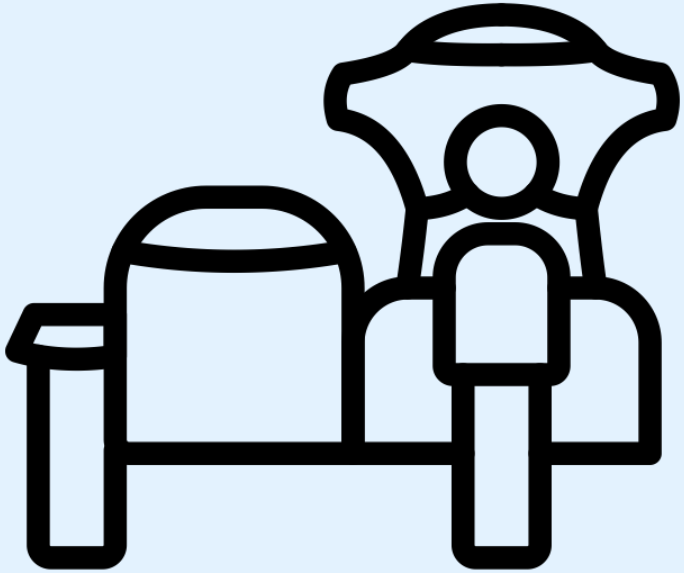
# It is easier than you think ...

- [ ] Install Linkerd
- [ ] Add the sidecar
- [ ] Find the culprit

LINKERD

# Watch the traffic

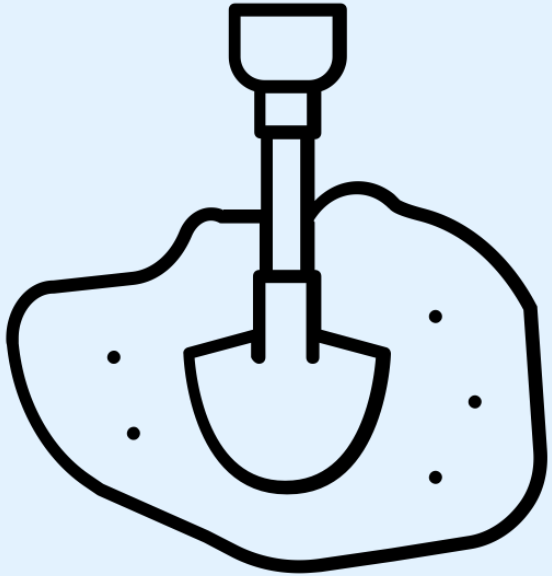
- [x] Install Linkerd
- [ ] Add the sidecar
- [ ] Find the culprit



Created by Grégory Montigny
from Noun Project

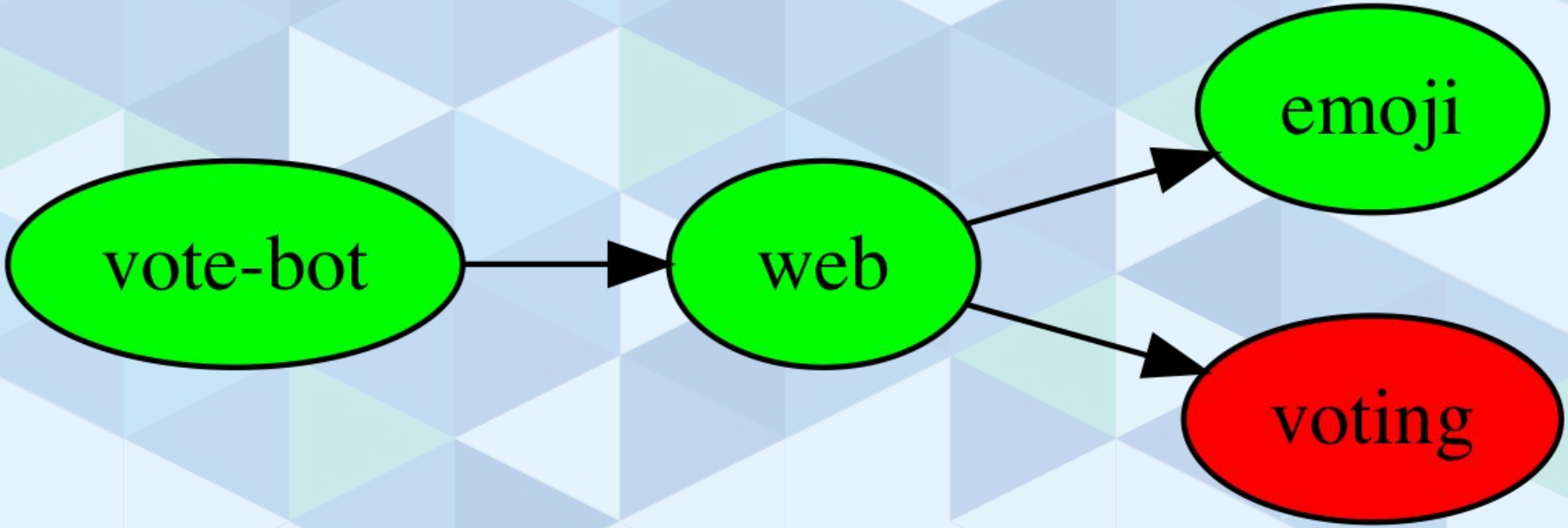
# 🔍 Dig into the details

- [x] Install Linkerd
- [x] Add the sidecar
- [ ] Find the culprit



Created by darwis
from Noun Project

# The Culprit



# Join our community!

github.com/linkerd

slack.linkerd.io

@linkerd

FROM YOUR FRIENDS AT

| | |
|---|---|
| Slides | http://bit.ly/thomas-talks |
| Get Started! | https://bit.ly/linkerd-get-started |