



Unikraft Performance Monitoring with Prometheus

Cezar Crăciunoiu <cezar.craciunoiu@upb.ro>

The Unikraft Unikernel

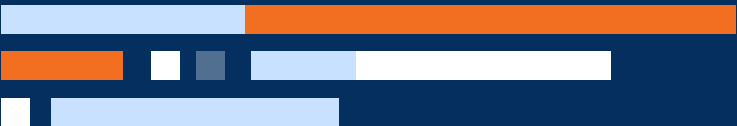


The Unikraft Unikernel



OPEN-SOURCE PROJECT

 <https://unikraft.org>



The Unikraft Unikernel



OPEN-SOURCE PROJECT

 <https://unikraft.org>

LIBRARY-OS

Complete kernel specialization



The Unikraft Unikernel



OPEN-SOURCE PROJECT

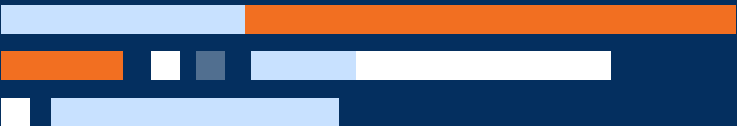
 <https://unikraft.org>

LIBRARY-OS

Complete kernel specialization

MORE & MORE APPS

Growing number of use cases



The Unikraft Unikernel



OPEN-SOURCE PROJECT

 <https://unikraft.org>

LIBRARY-OS

Complete kernel specialization

MORE & MORE APPS

Growing number of use cases

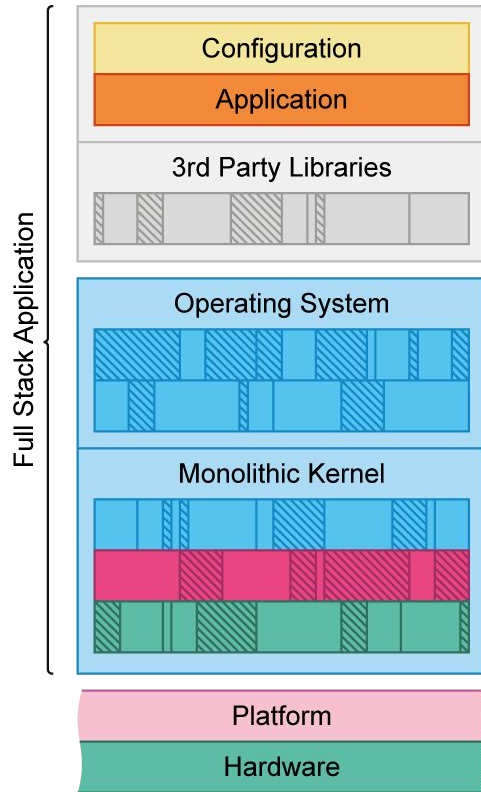
ACTIVE COMMUNITY

Releases every 2 months

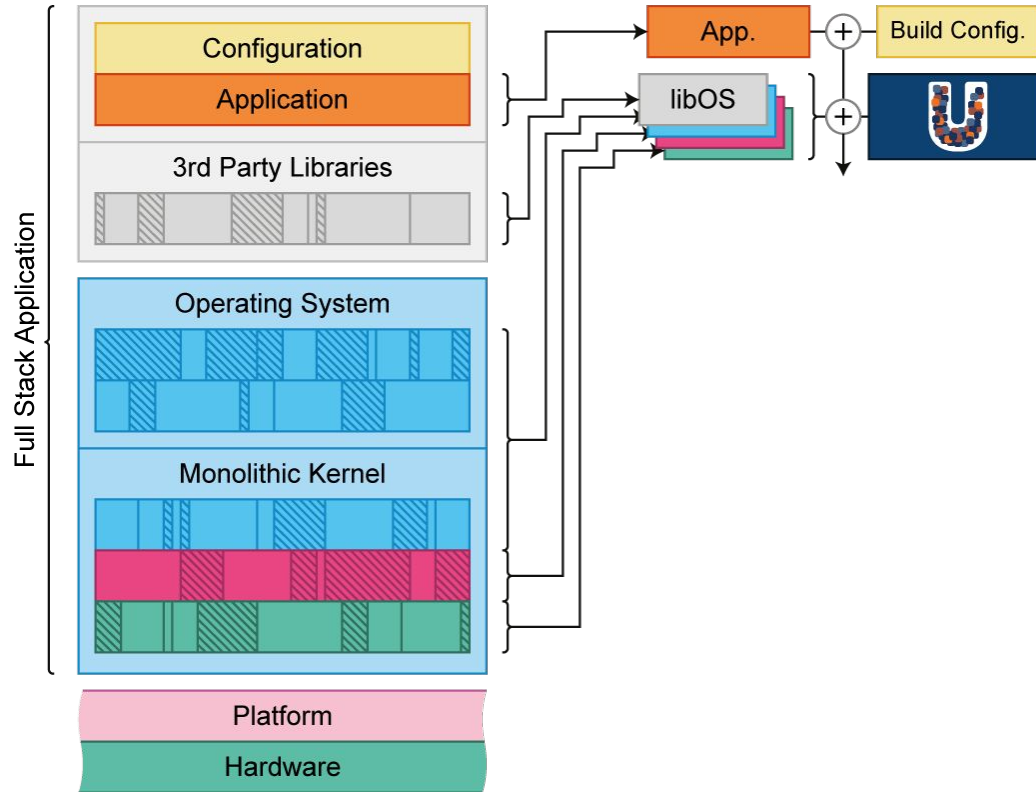


So what's a Unikernel?

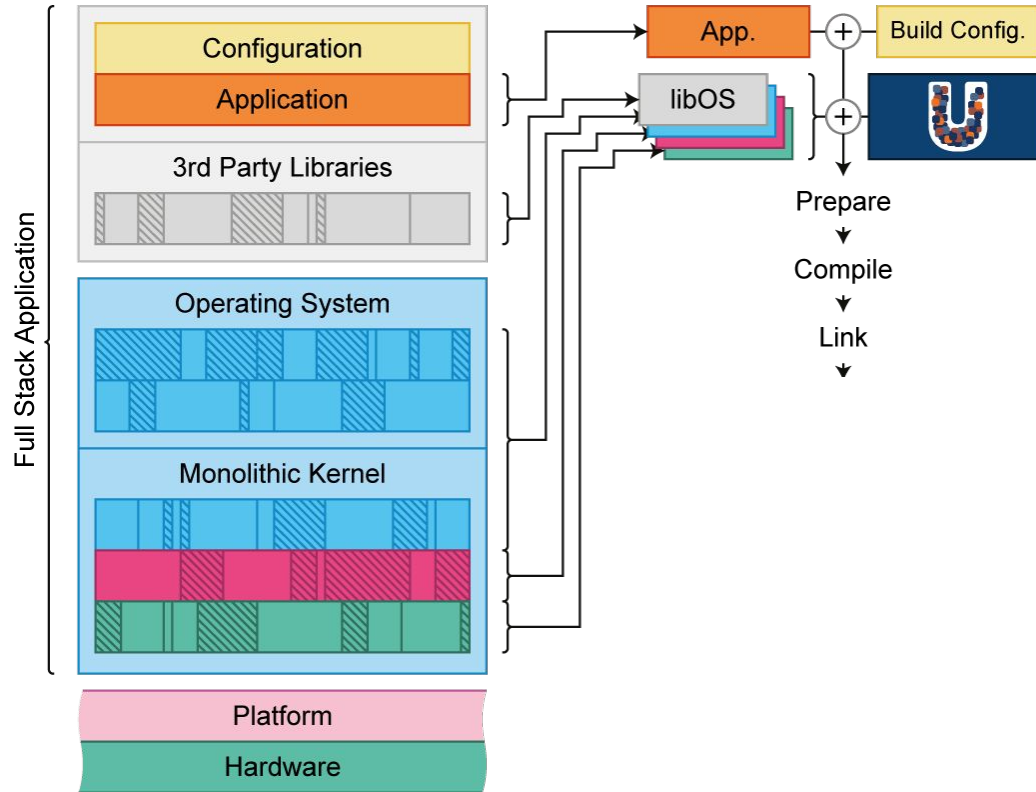
So what's a Unikernel?



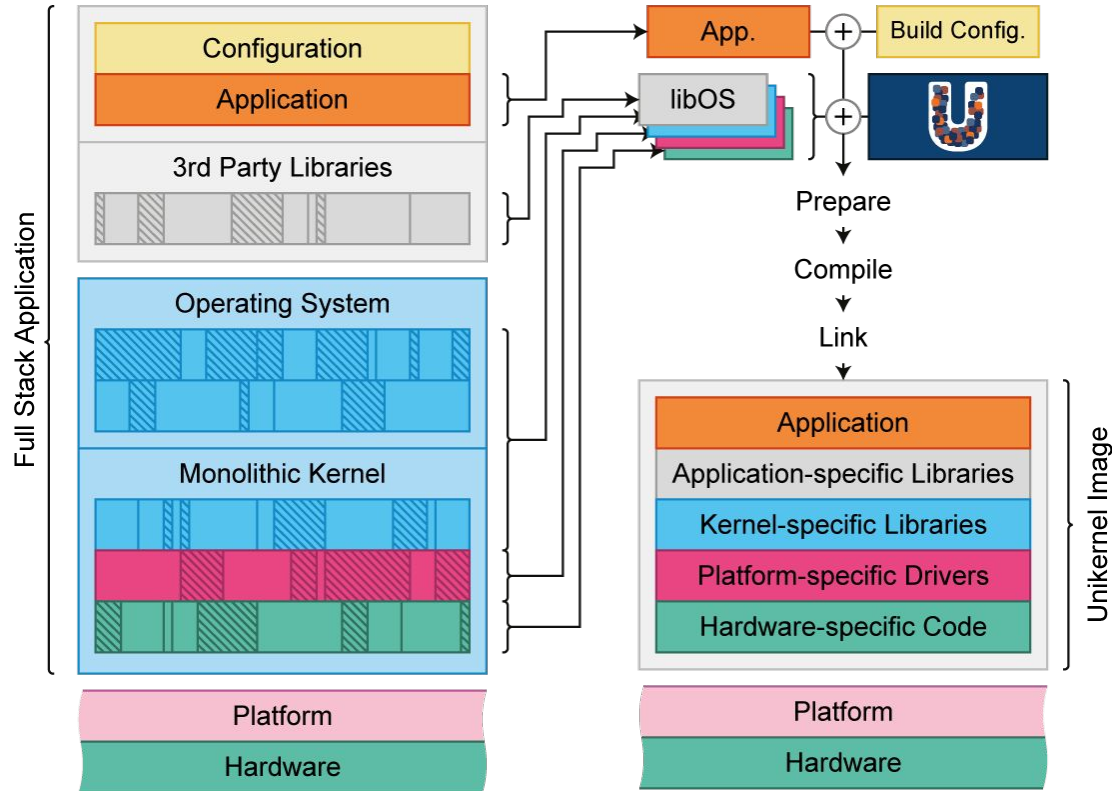
So what's a Unikernel?



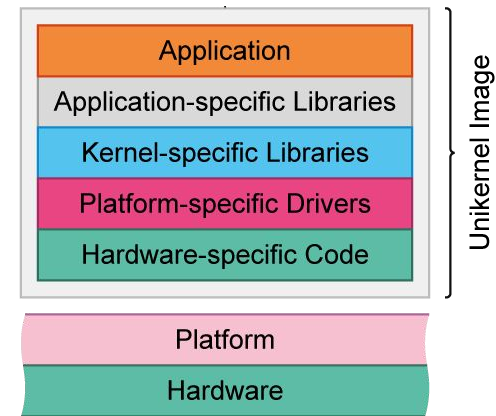
So what's a Unikernel?



So what's a Unikernel?

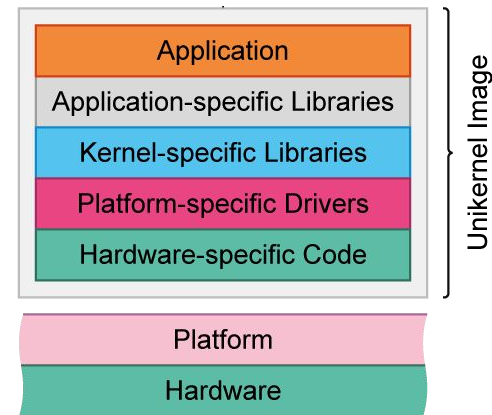


The Unikernel Model



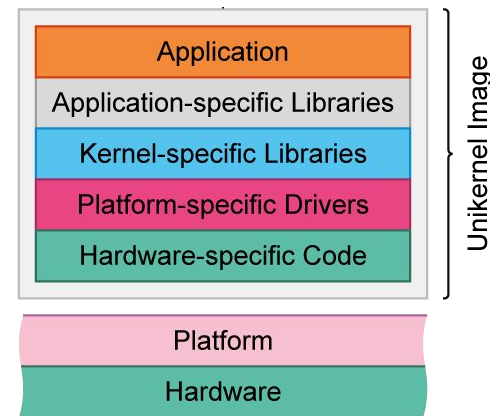
The Unikernel Model

→ Compile-time specialization strategy



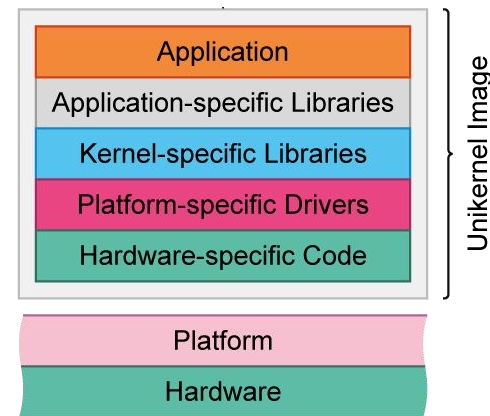
The Unikernel Model

- Compile-time specialization strategy
- Lightweight virtual machines



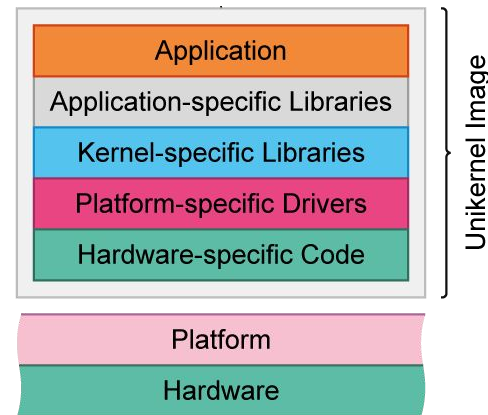
The Unikernel Model

- Compile-time specialization strategy
- Lightweight virtual machines
- Single, sealed address-space



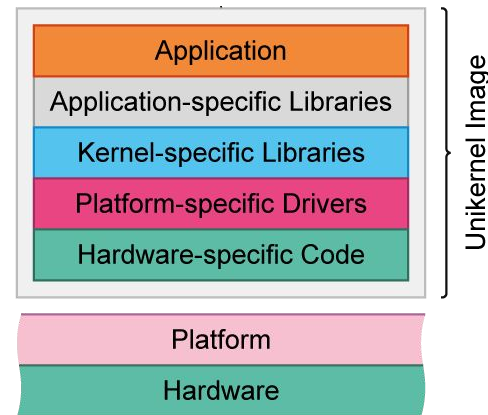
The Unikernel Model

- Compile-time specialization strategy
- Lightweight virtual machines
- Single, sealed address-space
- No costly syscalls



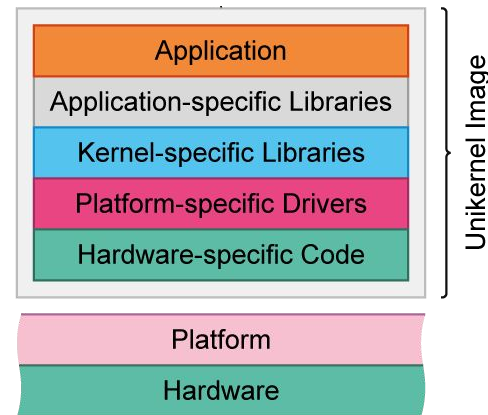
The Unikernel Model

- Compile-time specialization strategy
- Lightweight virtual machines
- Single, sealed address-space
- No costly syscalls
- Only necessary functionality for applications



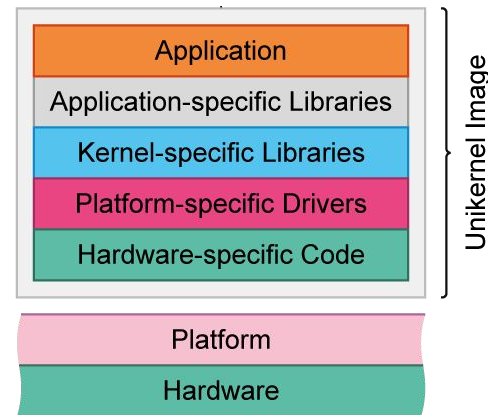
The Unikernel Model

- Compile-time specialization strategy
- Lightweight virtual machines
- Single, sealed address-space
- No costly syscalls
- Only necessary functionality for applications
- No daemons/system libs/SSH



The Unikernel Model

- Compile-time specialization strategy
- Lightweight virtual machines
- Single, sealed address-space
- No costly syscalls
- Only necessary functionality for applications
- No daemons/system libs/SSH
- Platform/Hardware specific



What about Prometheus?



What about Prometheus?



What about Prometheus?



What about Prometheus?



What about Prometheus?



Anything

Anytime

Anywhere

What about Prometheus?



Anything



Anytime

Anywhere

What about Prometheus?



Anything



Anytime



Anywhere

What about Prometheus?



Anything



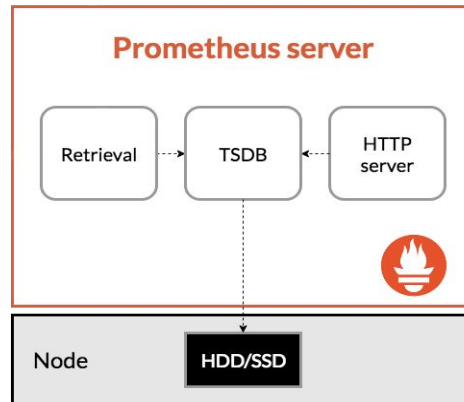
Anytime



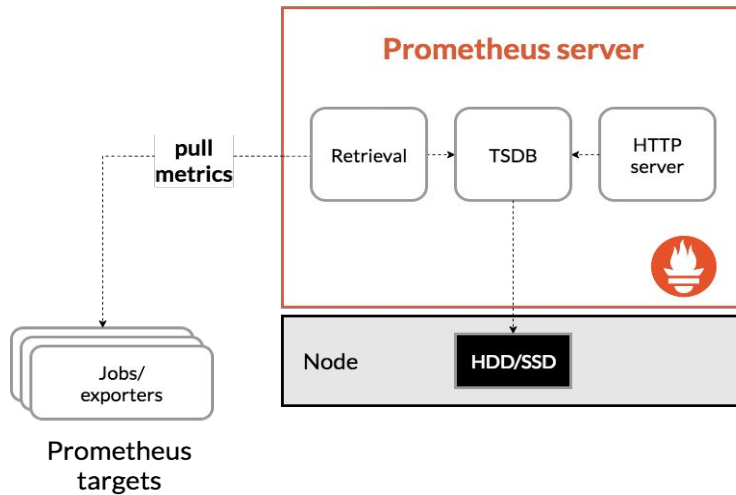
Anywhere



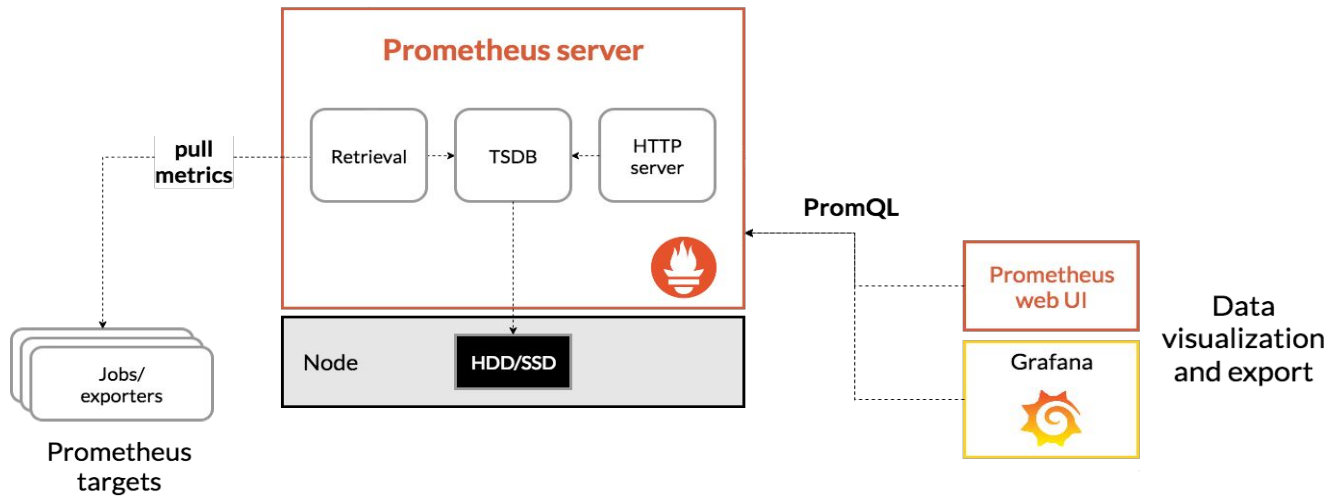
Prometheus in Detail



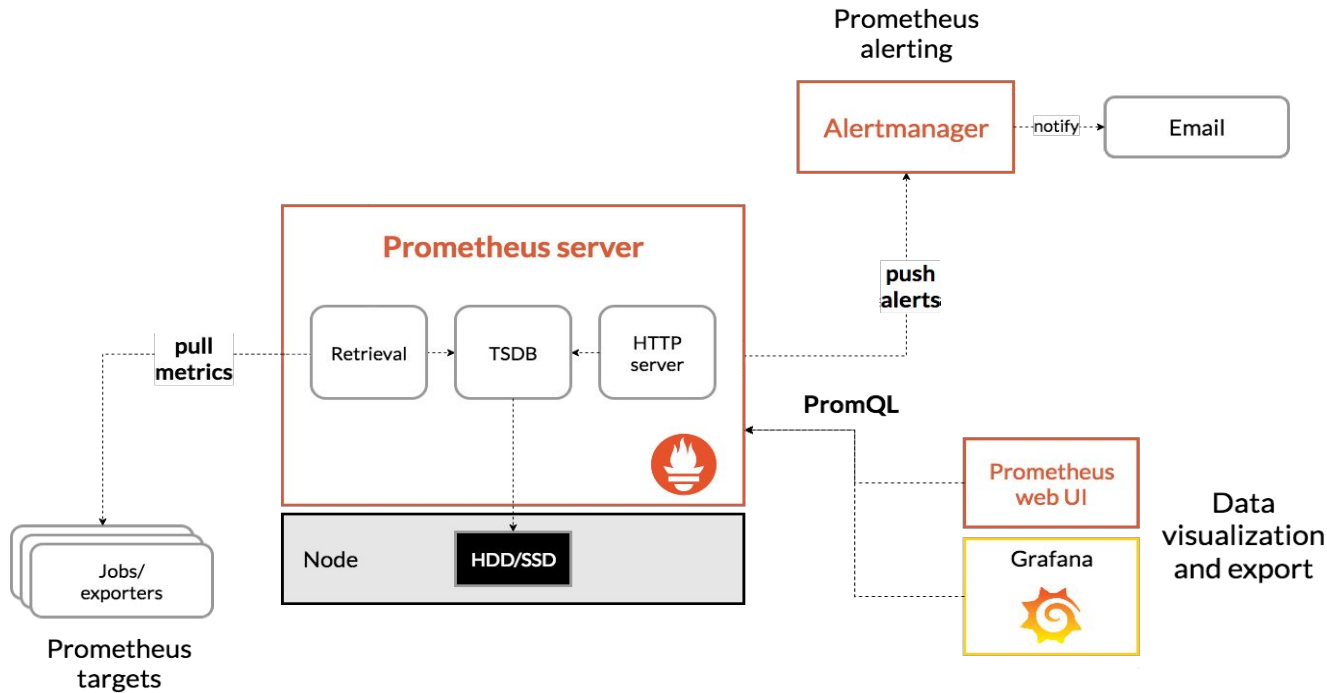
Prometheus in Detail



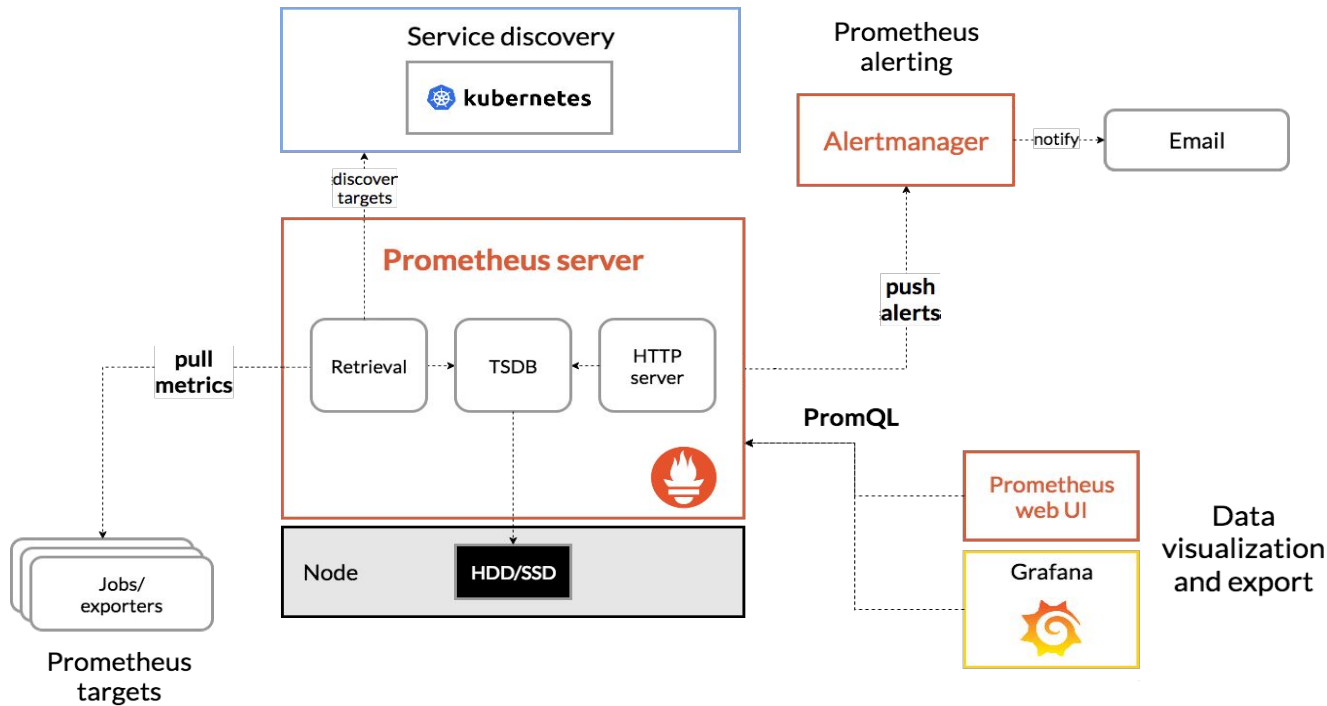
Prometheus in Detail



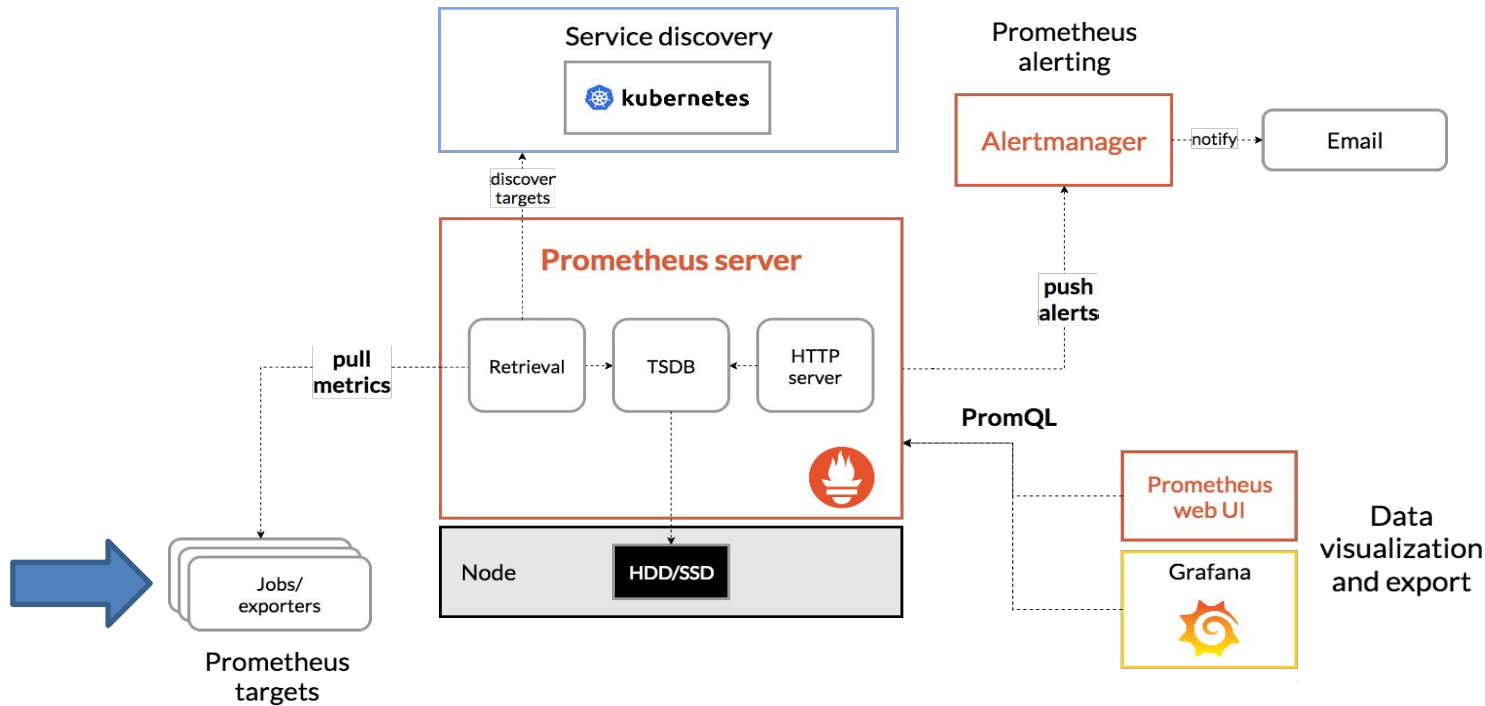
Prometheus in Detail



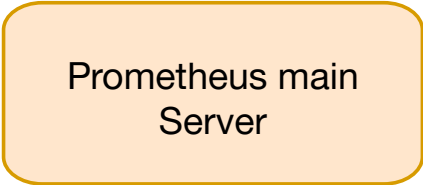
Prometheus in Detail



Prometheus in Detail



Prometheus and Unikraft



Prometheus main
Server

A diagram consisting of a single light orange rounded rectangle with a thin orange border, centered on a white background. The rectangle contains the text 'Prometheus main' on the top line and 'Server' on the bottom line, both in black font.

Prometheus and Unikraft

Unikraft instance

Prometheus main
Server

Prometheus and Unikraft

Unikraft instance

Unikraft instance

Prometheus main
Server

Prometheus and Unikraft

Unikraft instance

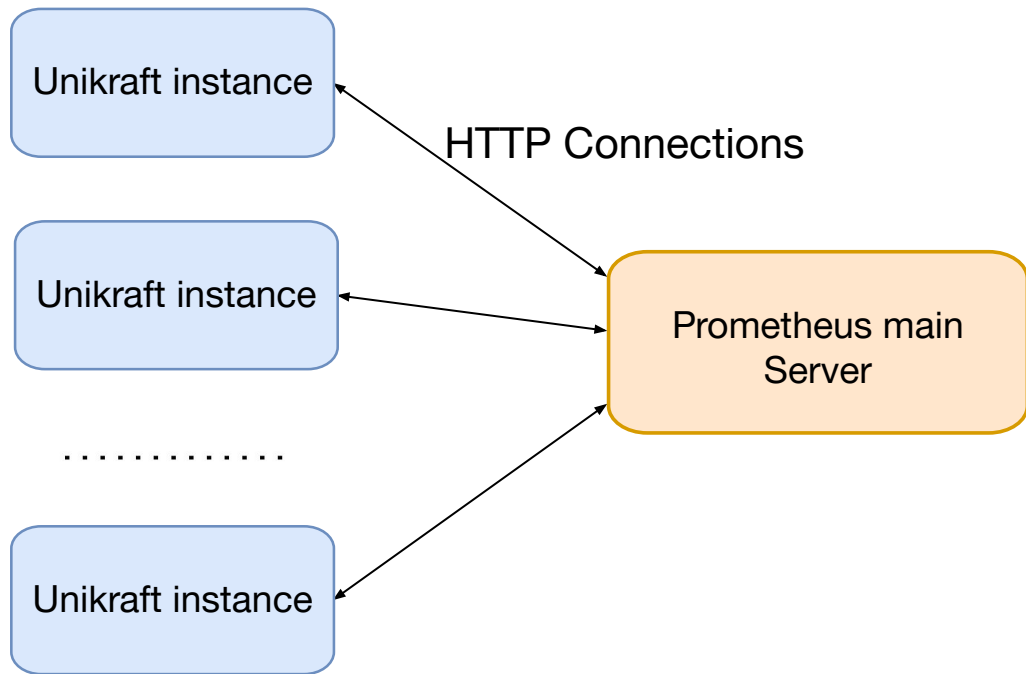
Unikraft instance

.....

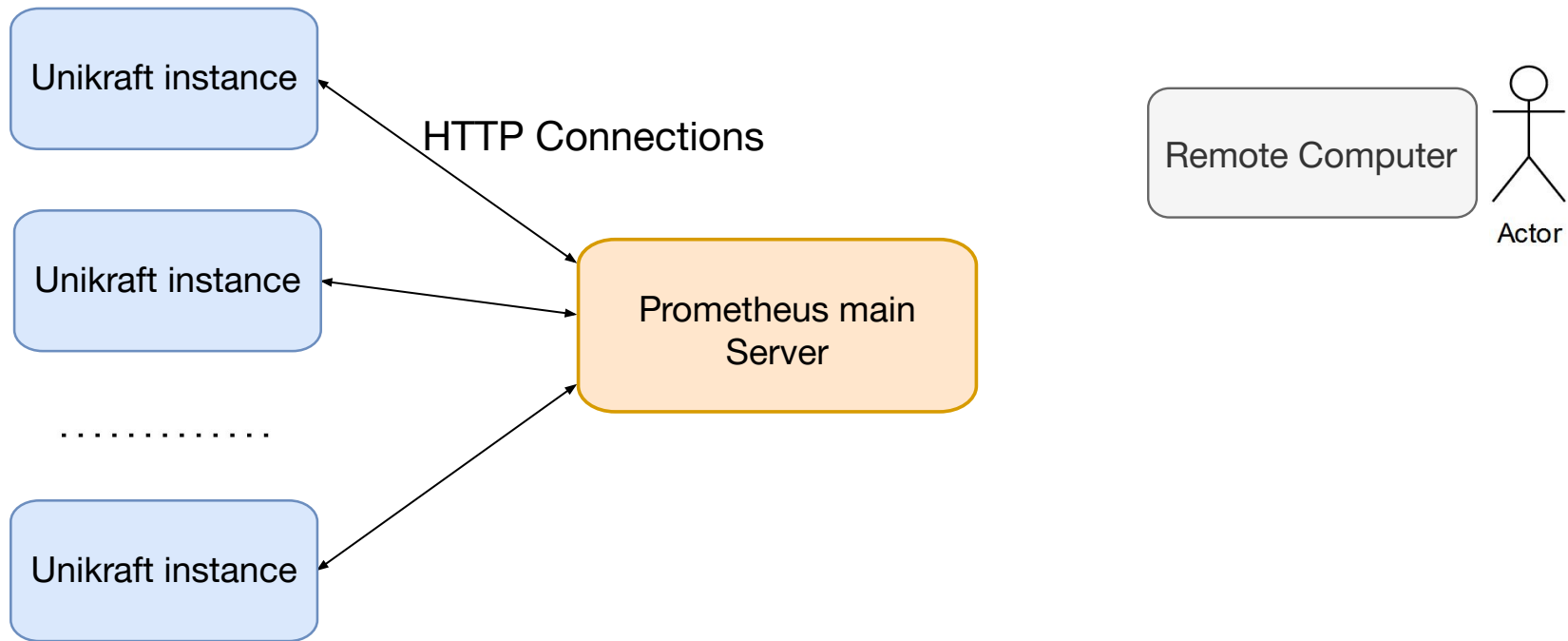
Unikraft instance

Prometheus main
Server

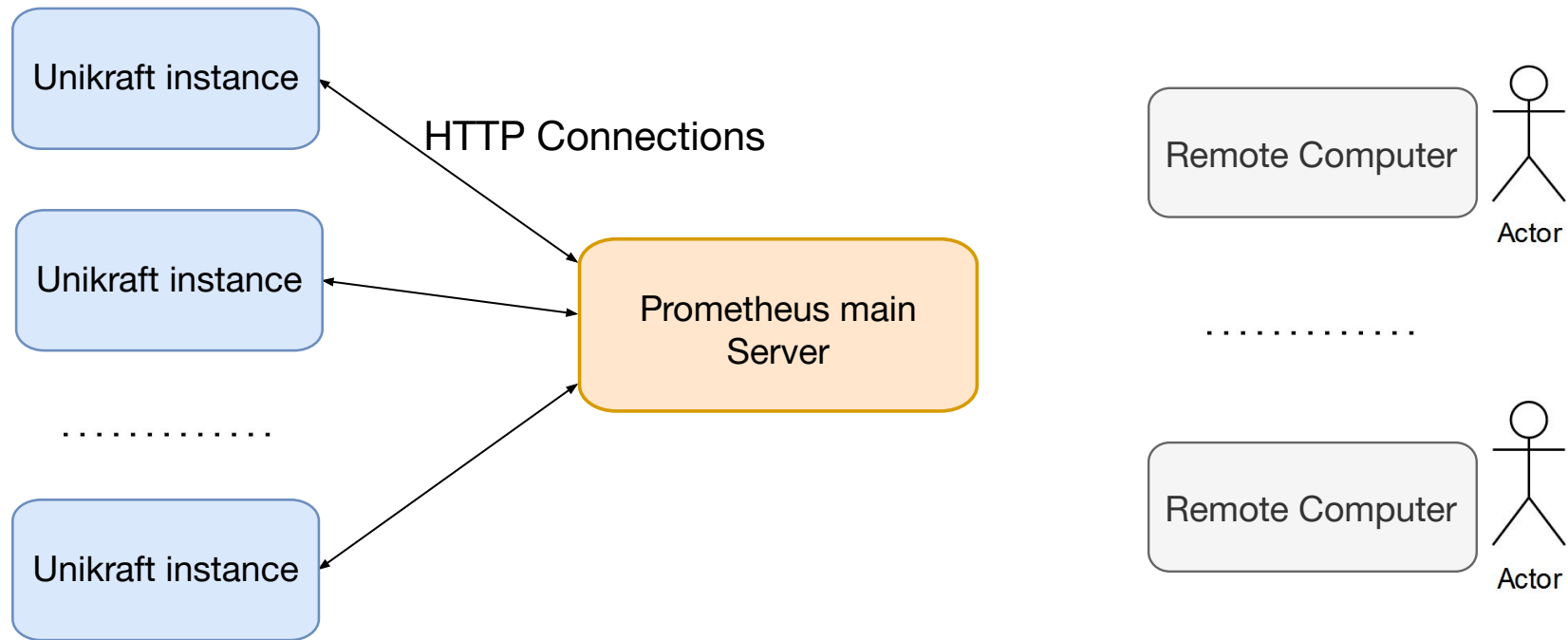
Prometheus and Unikraft



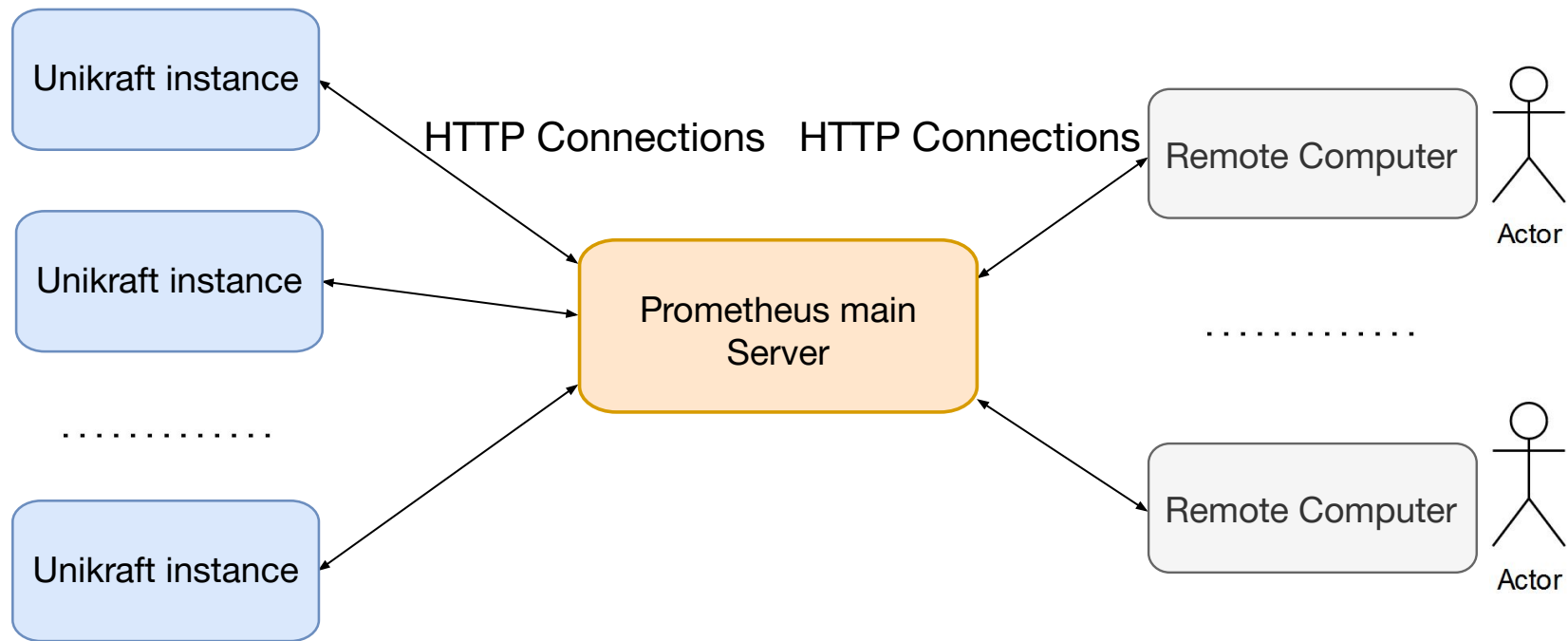
Prometheus and Unikraft



Prometheus and Unikraft



Prometheus and Unikraft





Unikraft Exporter Overview

Unikraft Core



No Implementation

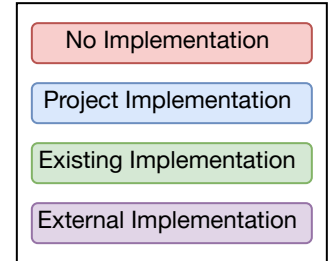
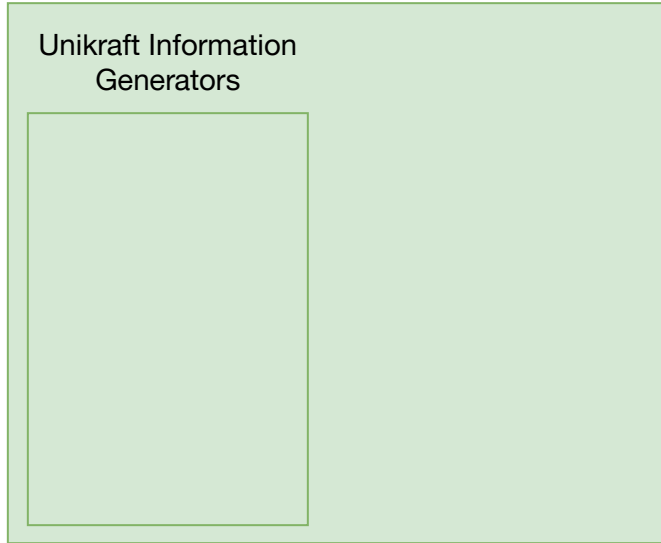
Project Implementation

Existing Implementation

External Implementation

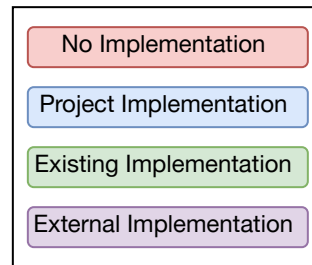
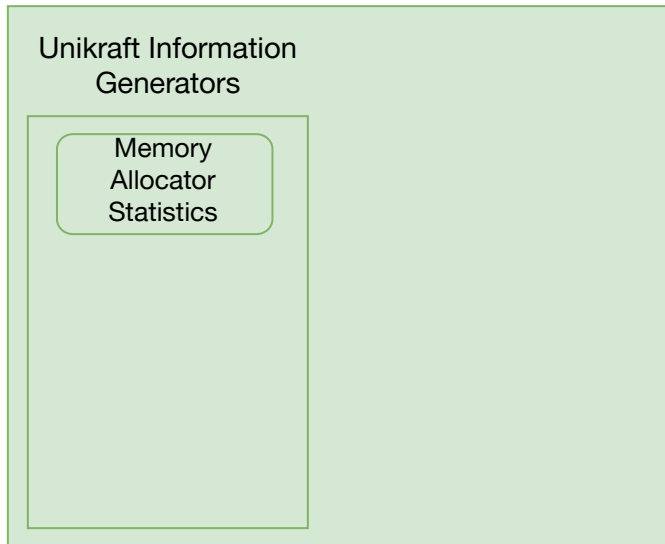
Unikraft Exporter Overview

Unikraft Core



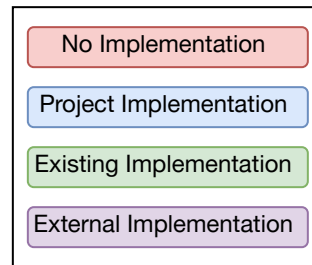
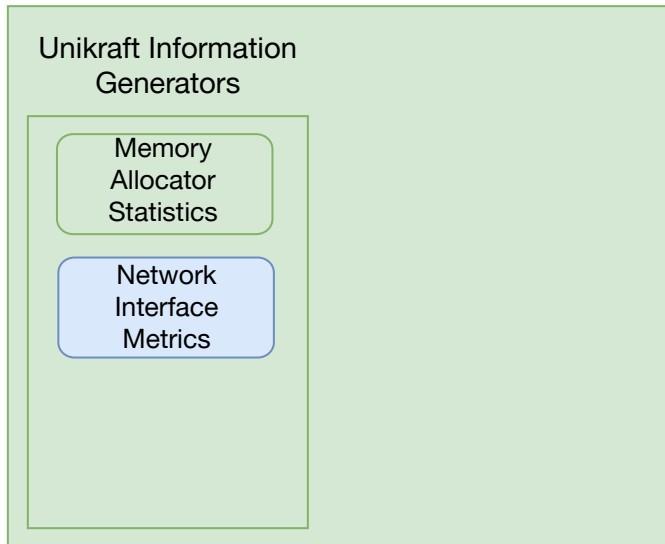
Unikraft Exporter Overview

Unikraft Core



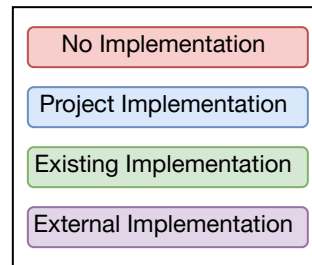
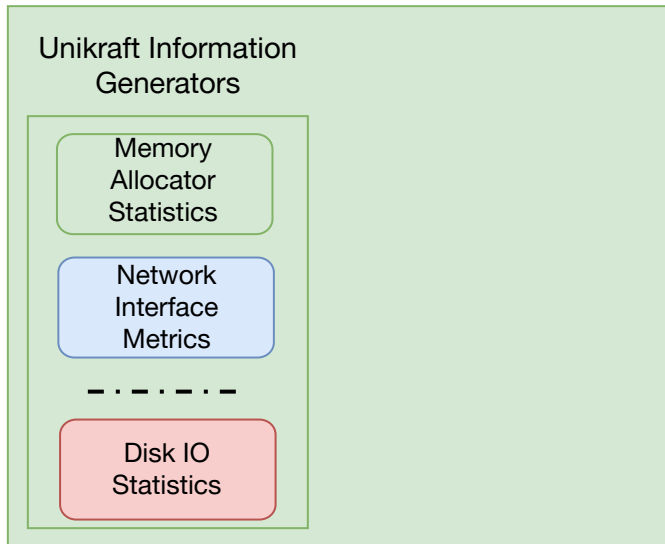
Unikraft Exporter Overview

Unikraft Core



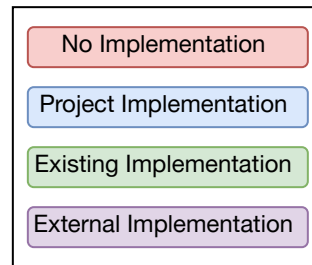
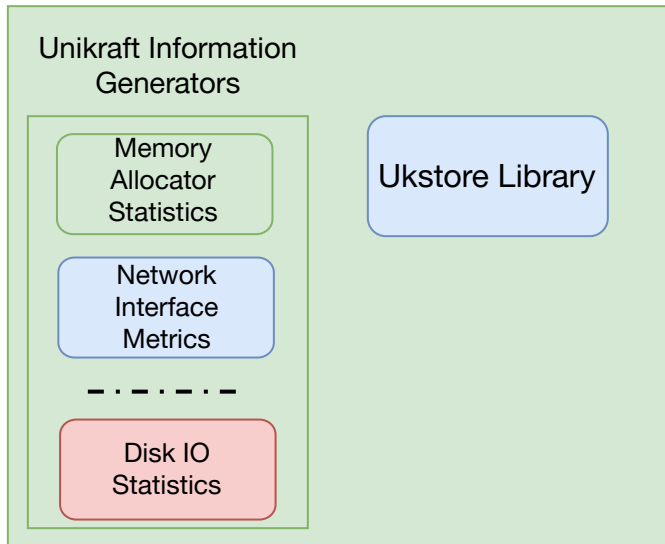
Unikraft Exporter Overview

Unikraft Core

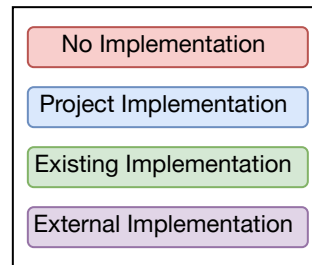
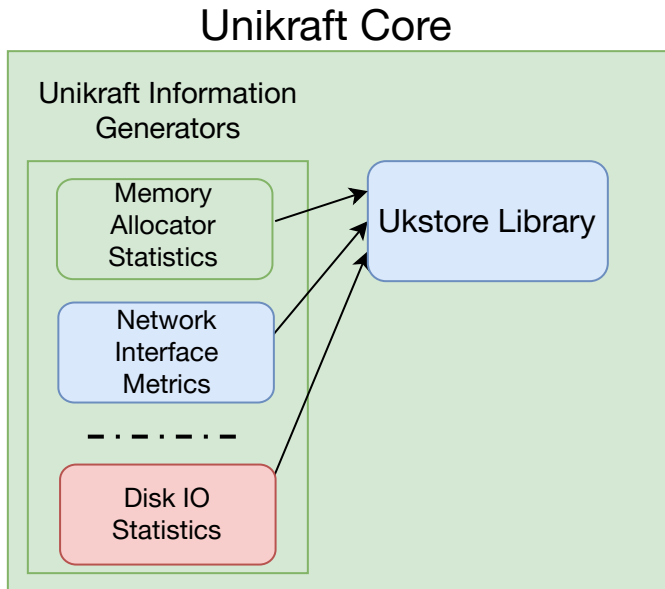


Unikraft Exporter Overview

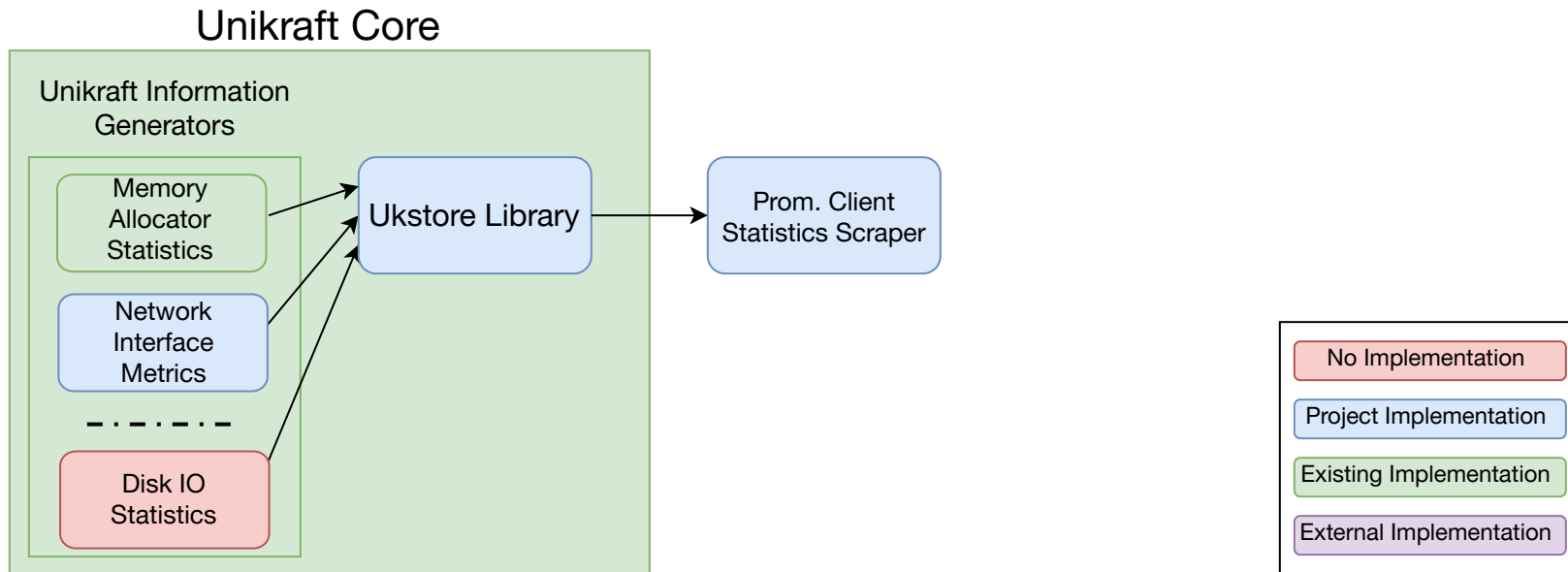
Unikraft Core



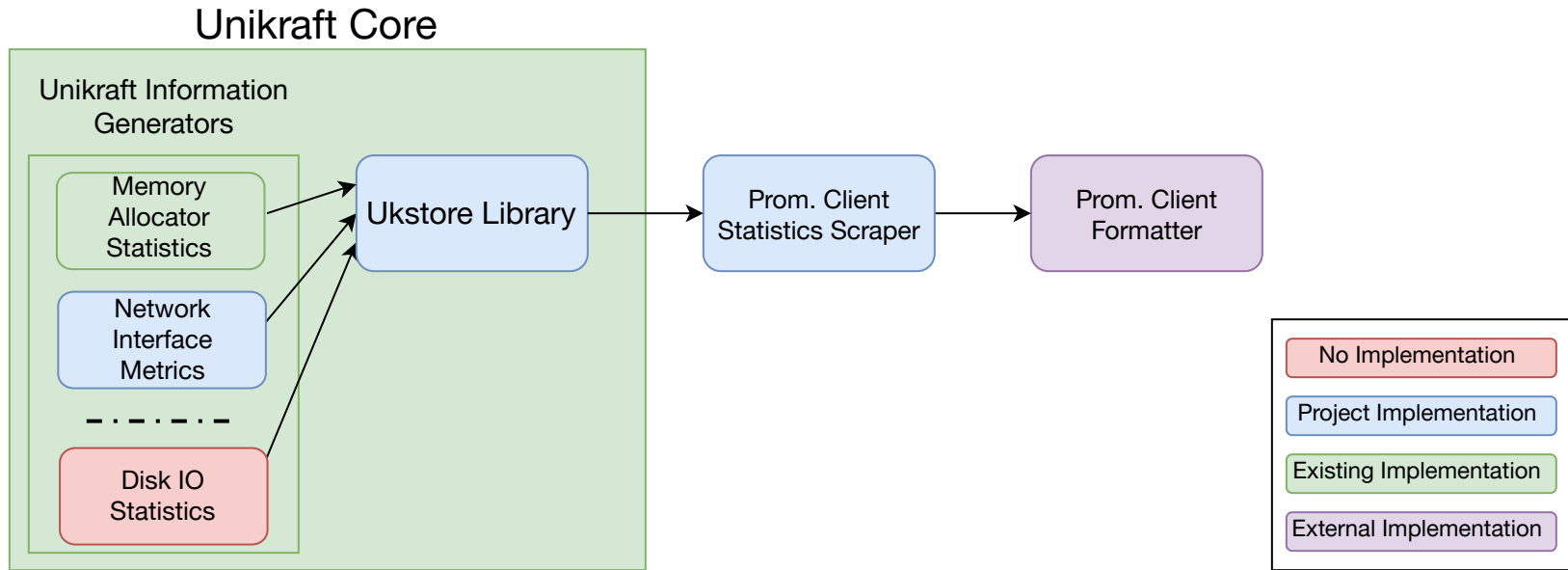
Unikraft Exporter Overview



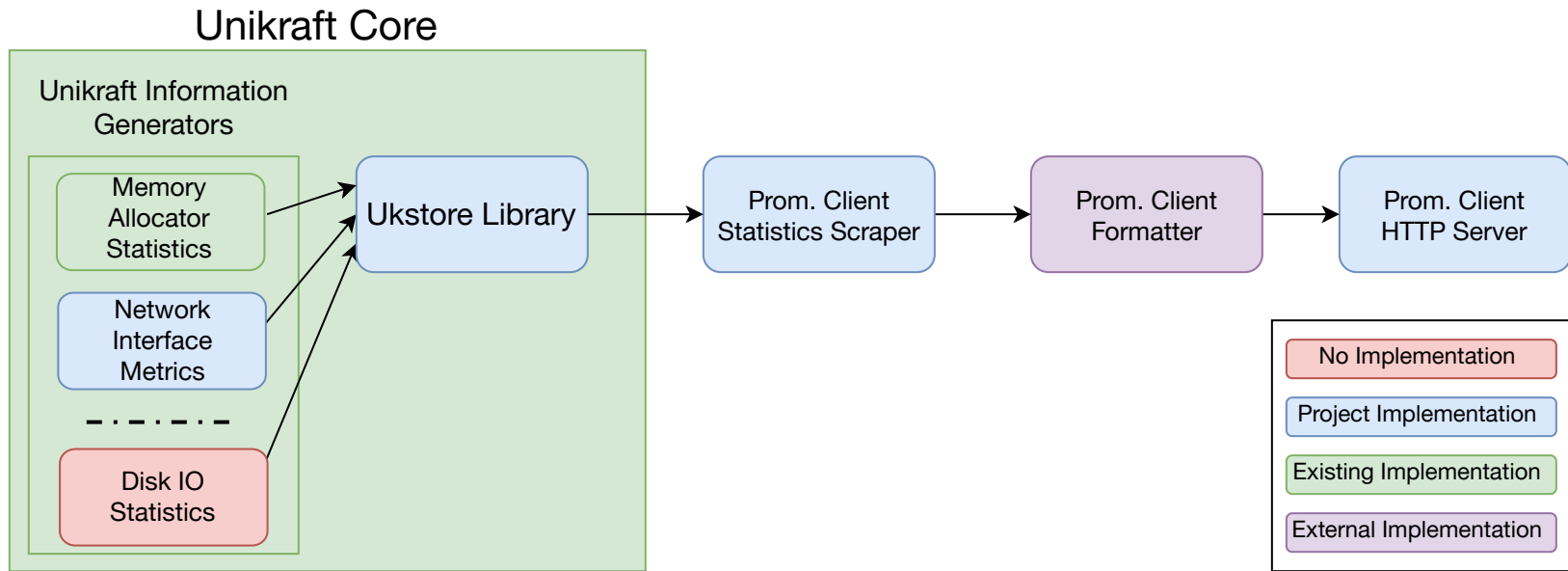
Unikraft Exporter Overview



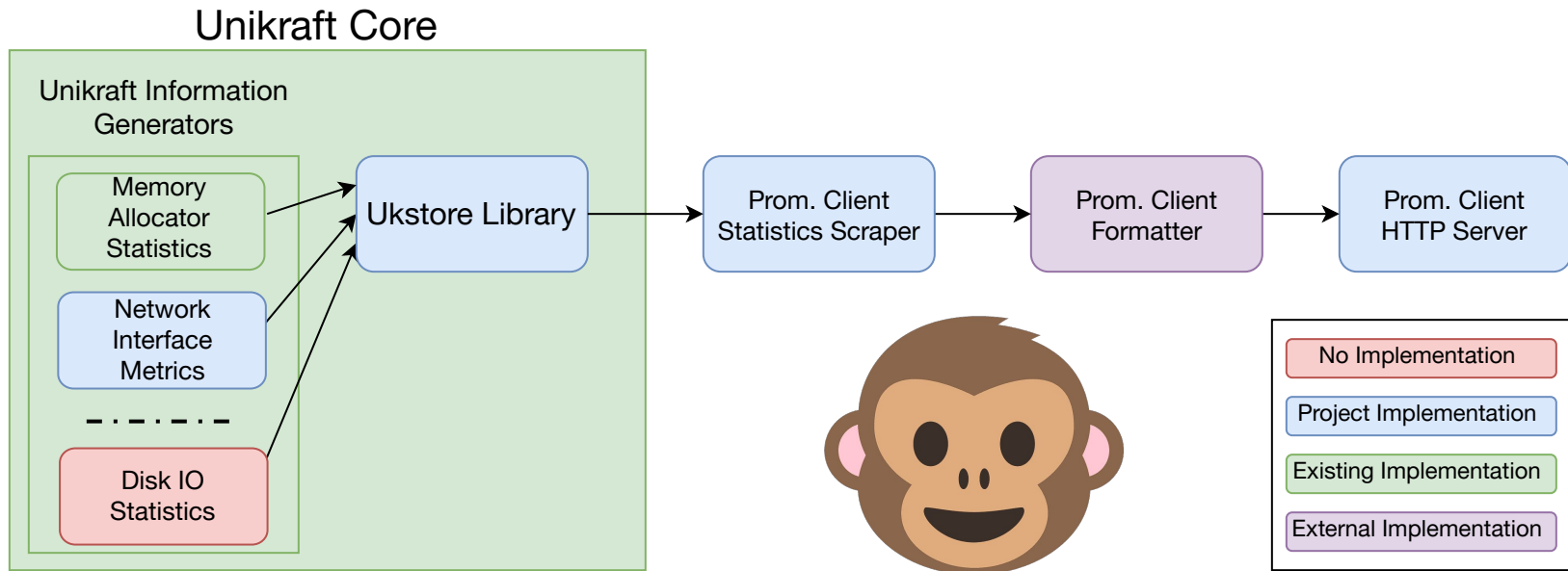
Unikraft Exporter Overview



Unikraft Exporter Overview

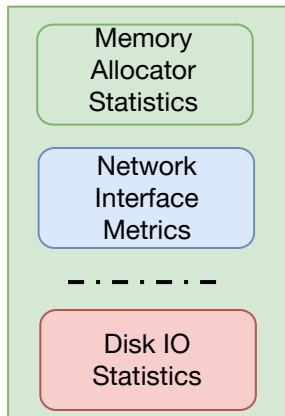


Unikraft Exporter Overview



Solution – Info Generators

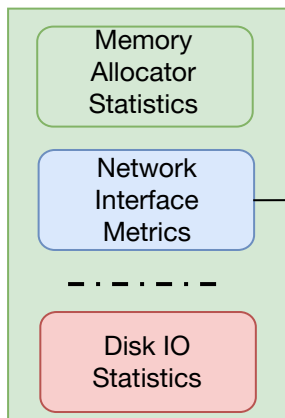
Unikraft Information
Generators



A top down view

Solution – Info Generators

Unikraft Information
Generators



```
struct uk_netdev_metrics {  
    struct uk_netdev_tx_metrics tx_m;  
    struct uk_netdev_rx_metrics rx_m;  
};  
  
int uk_netdev_metrics_get(struct uk_netdev *dev,  
    struct uk_netdev_metrics *dev_metrics)
```

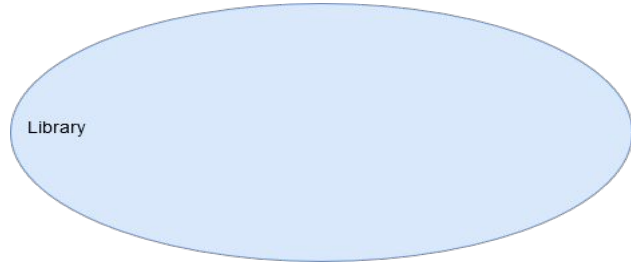
Solution – Ukstore (I)

Solution – Ukstore (I)

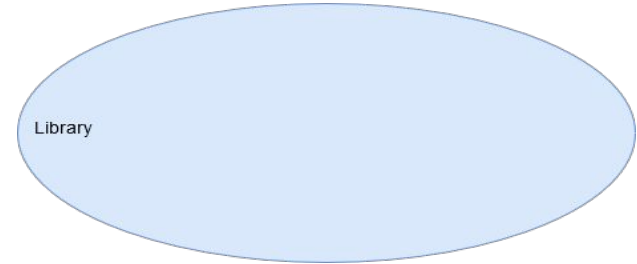
Static Structure

Dynamic Structure

Solution – Ukstore (I)

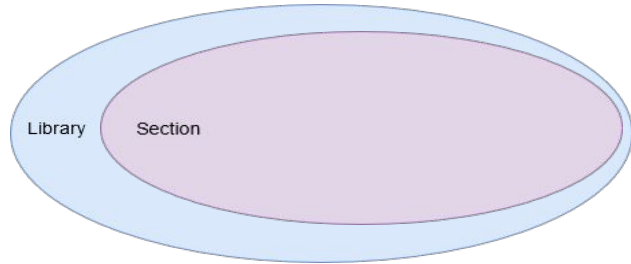


Static Structure

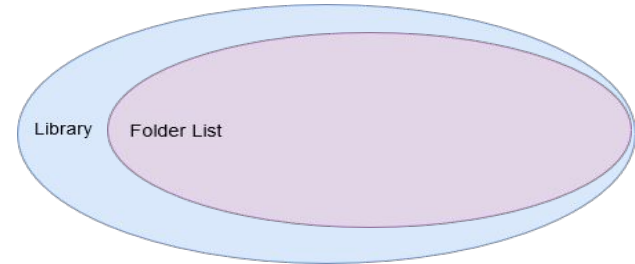


Dynamic Structure

Solution – Ukstore (I)

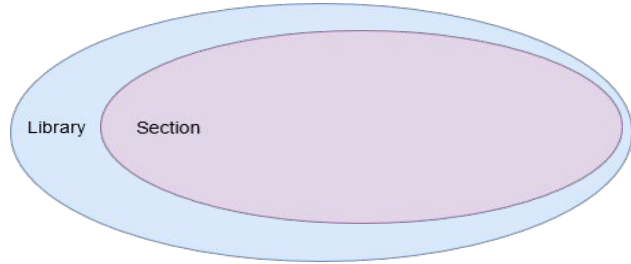


Static Structure

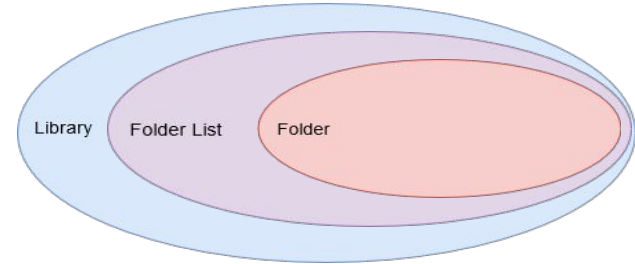


Dynamic Structure

Solution – Ukstore (I)

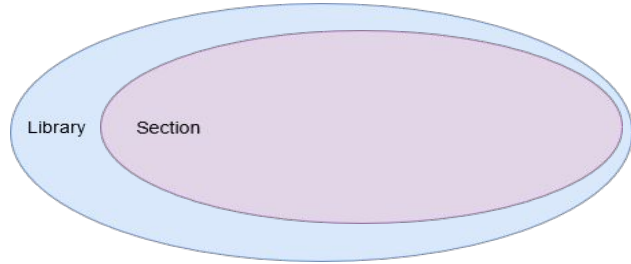


Static Structure

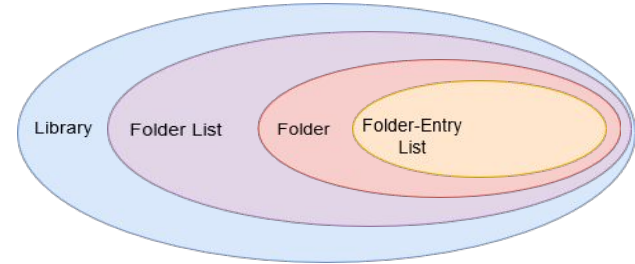


Dynamic Structure

Solution – Ukstore (I)

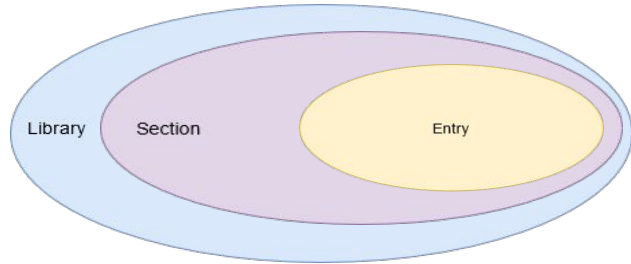


Static Structure

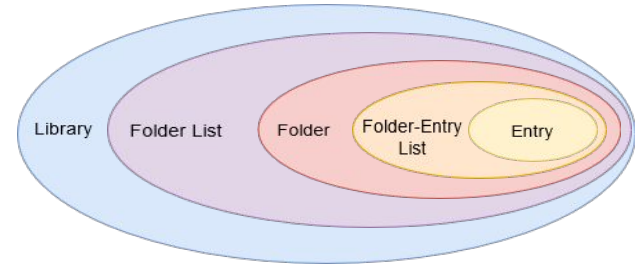


Dynamic Structure

Solution – Ukstore (I)

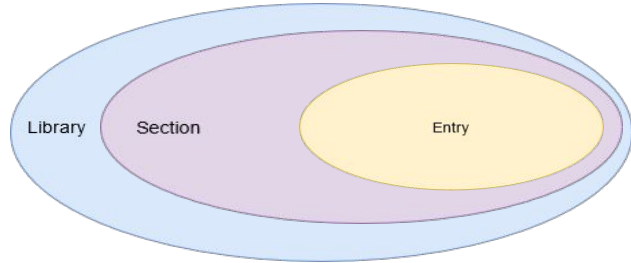


Static Structure

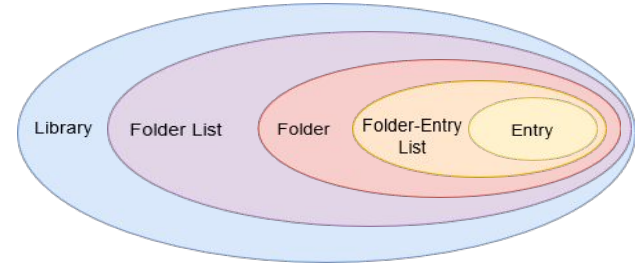


Dynamic Structure

Solution – Ukstore (I)



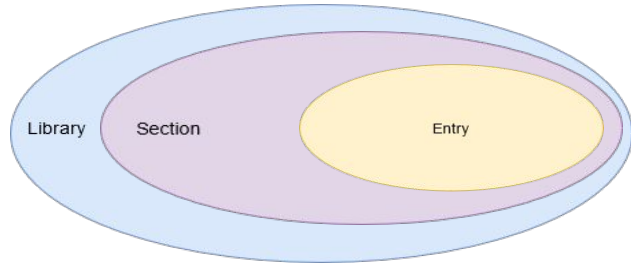
Static Structure



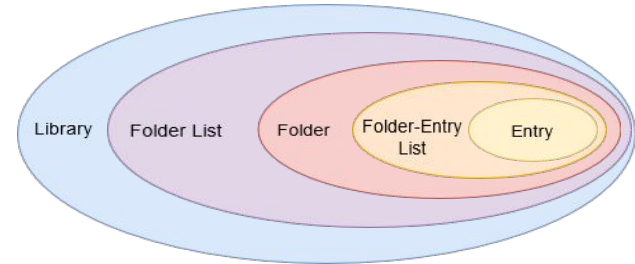
Dynamic Structure

- Identical functionality

Solution – Ukstore (I)



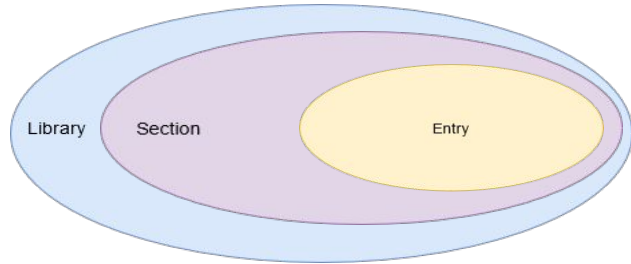
Static Structure



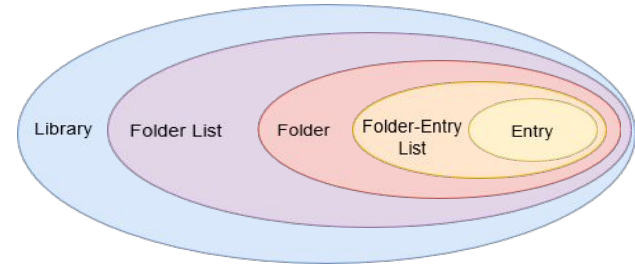
Dynamic Structure

- Identical functionality
- Different use cases

Solution – Ukstore (I)



Static Structure



Dynamic Structure

- Identical functionality
- Different use cases
- Different overhead

Solution – Ukstore (II)

How to link generators to Ukstore?

Solution – Ukstore (II)

How to link generators to Ukstore?

```
int uk_netdev_metrics_get_all(__uptr *dst)
```

Solution – Ukstore (II)

How to link generators to Ukstore?

```
int uk_netdev_metrics_get_all(__uptr *dst)
```

```
#include <uk/store.h>
```


```
UK_STORE_STATIC_ENTRY(en0_metrics_all, uptr, uk_netdev_metrics_get_all, NULL);
```

Solution – Ukstore (II)

How to link generators to Ukstore?

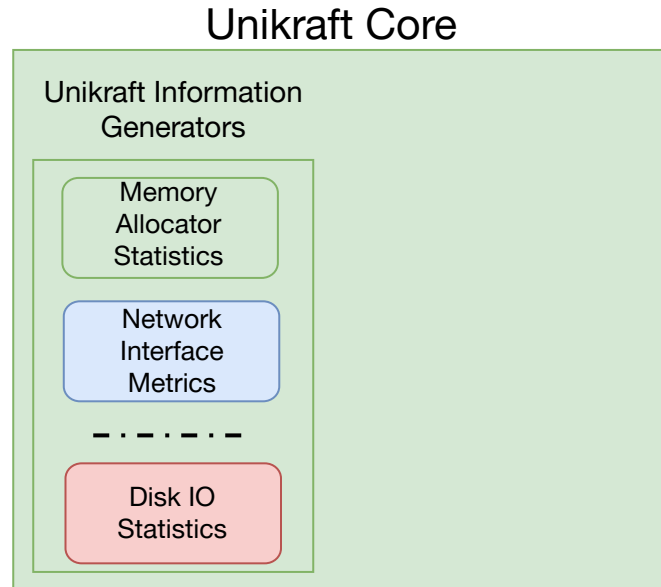
```
int uk_netdev_metrics_get_all(__uptr *dst)
```

```
#include <uk/store.h>
```

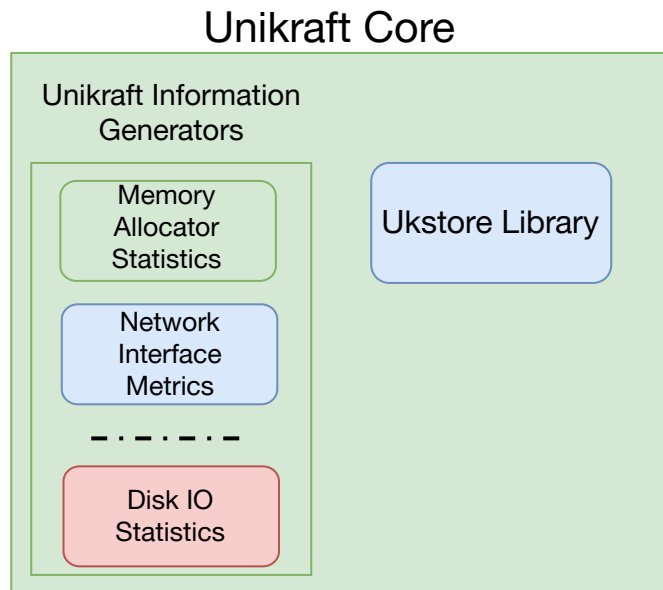
```
UK_STORE_STATIC_ENTRY(en0_metrics_all,  uptr, uk_netdev_metrics_get_all, NULL);
```

uptr, u64, s64, u32, s32, ...

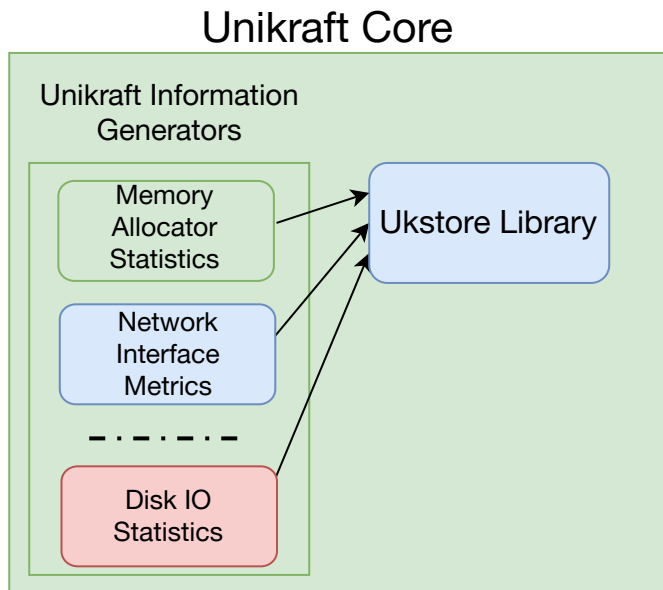
Solution – Ukstore (II)



Solution – Ukstore (II)



Solution – Ukstore (II)

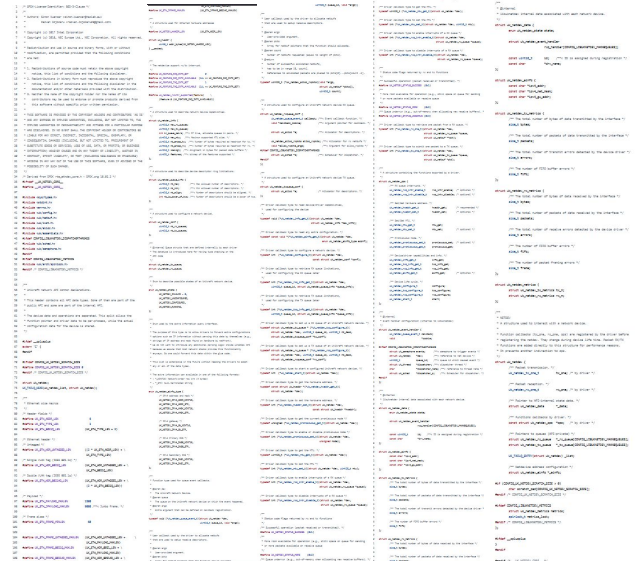


Ukstore – Basic Usage

→ *Why go through all this hassle just to call a function?*

Ukstore – Basic Usage

- *Why go through all this hassle just to call a function?*
- Sure, go ahead, here's one file



Ukstore – Basic Usage

- *Why go through all this hassle just to call a function?*
- Or you can just use Ukstore

```
struct uk_store_entry *netdev_entry;
```

Ukstore – Basic Usage

- *Why go through all this hassle just to call a function?*
- Or you can just use Ukstore

```
struct uk_store_entry *netdev_entry;  
.....  
netdev_entry = uk_store_get_entry(libuknetdev, NULL, "en0_metrics_all");
```

Ukstore – Basic Usage

- *Why go through all this hassle just to call a function?*
- Or you can just use Ukstore

```
struct uk_store_entry *netdev_entry;  
.....  
netdev_entry = uk_store_get_entry(libuknetdev, NULL, "en0_metrics_all");  
.....  
struct uk_netdev_metrics *net_stats = NULL;  
uk_store_get_value(netdev_entry, uptr, (__uptr *) &net_stats);
```

Ukstore – Basic Usage

- *Why go through all this hassle just to call a function?*
- Or you can just use Ukstore

```
struct uk_store_entry *netdev_entry;  
.....  
netdev_entry = uk_store_get_entry(libuknetdev, NULL, "en0_metrics_all");  
.....  
struct uk_netdev_metrics *net_stats = NULL;  
uk_store_get_value(netdev_entry, uptr, (__uptr *) &net_stats);  
.....  
uk_store_free_entry(&netdev_entry);
```

Solution – Prometheus Scraper

Solution – Prometheus Scraper

Prometheus Library
Sources

Solution – Prometheus Scraper

Prometheus Library
Sources

HTTP Server

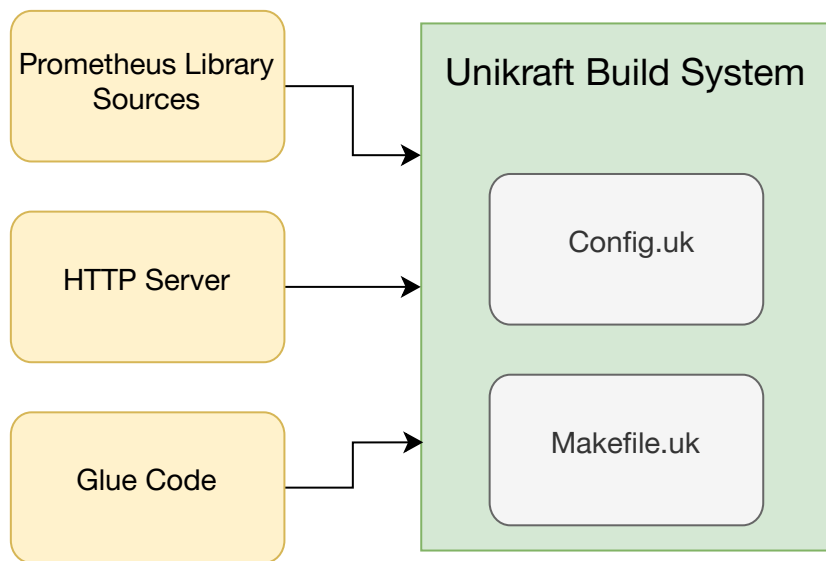
Solution – Prometheus Scraper

Prometheus Library
Sources

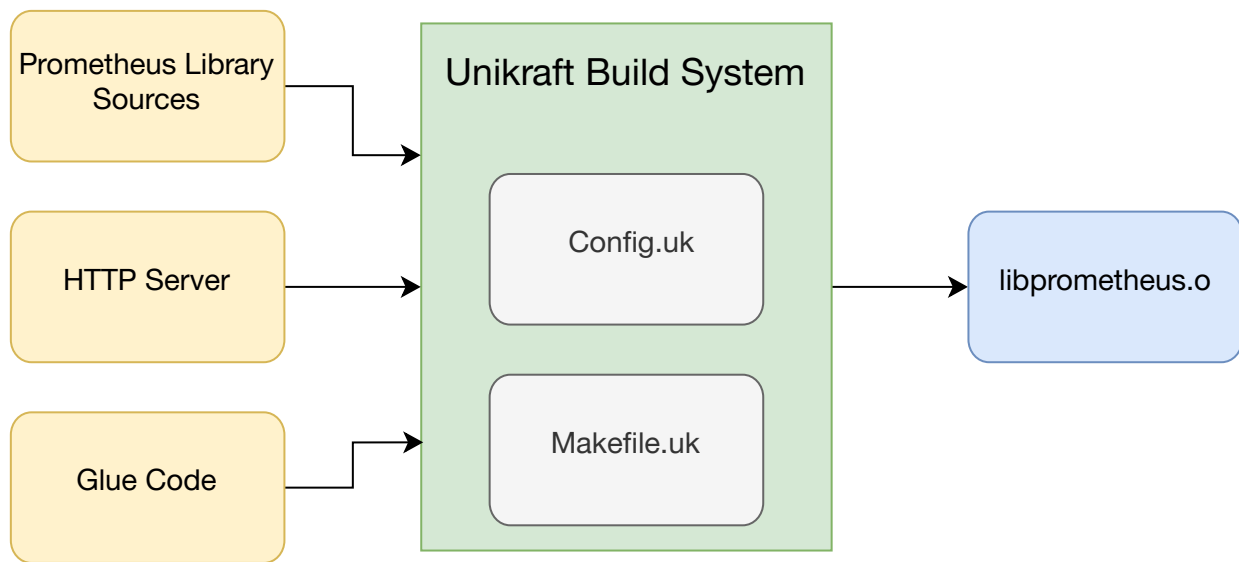
HTTP Server

Glue Code

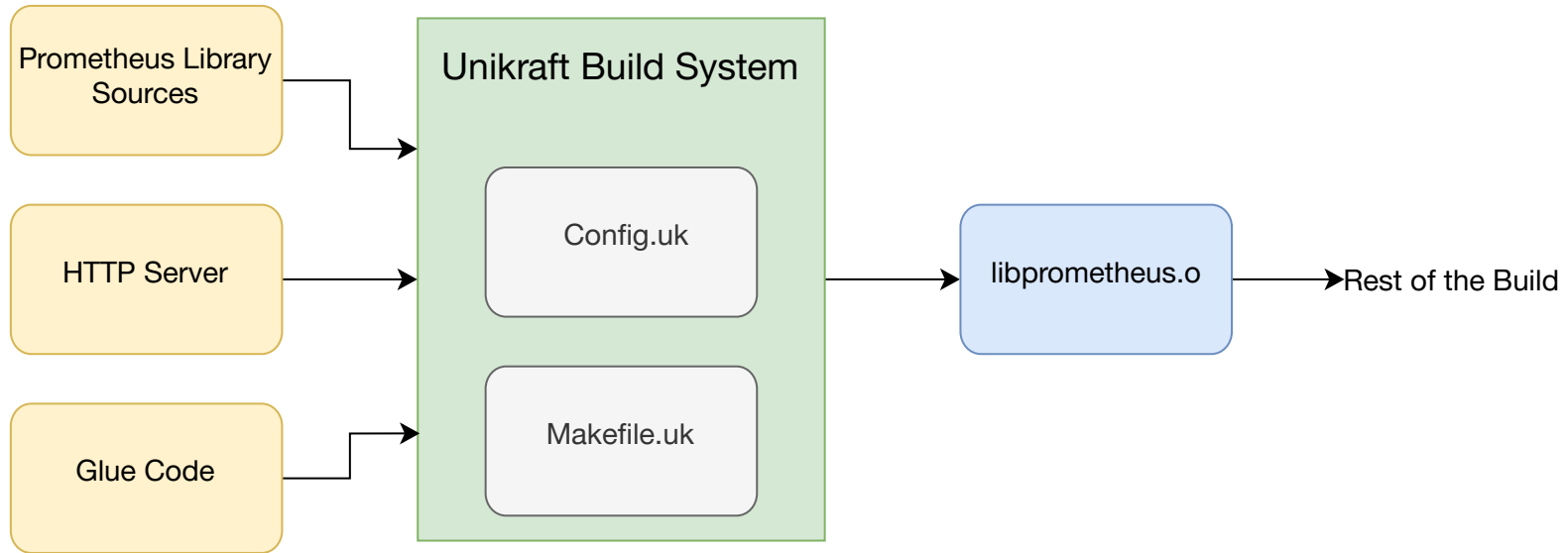
Solution – Prometheus Scraper



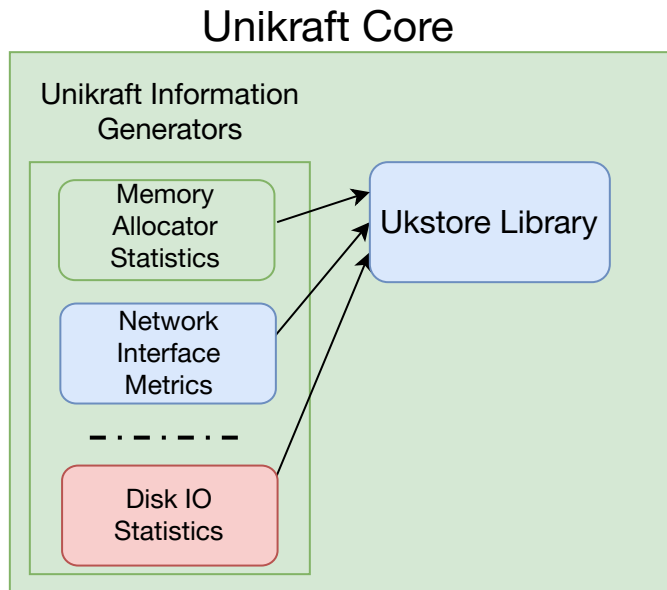
Solution – Prometheus Scraper



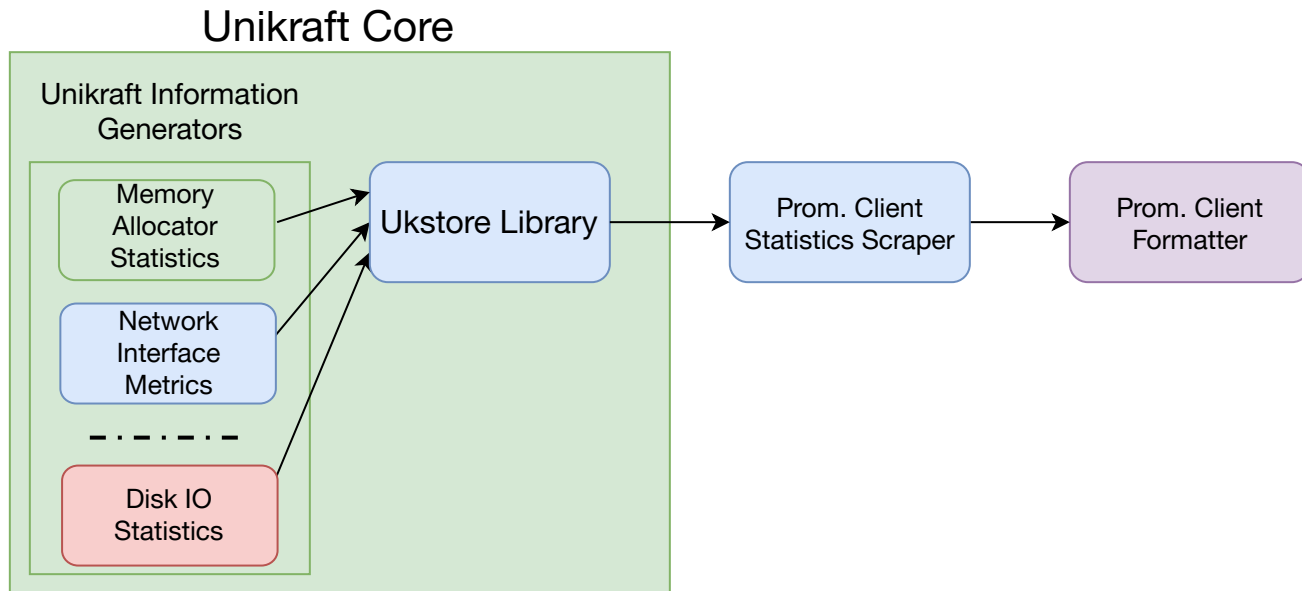
Solution – Prometheus Scraper



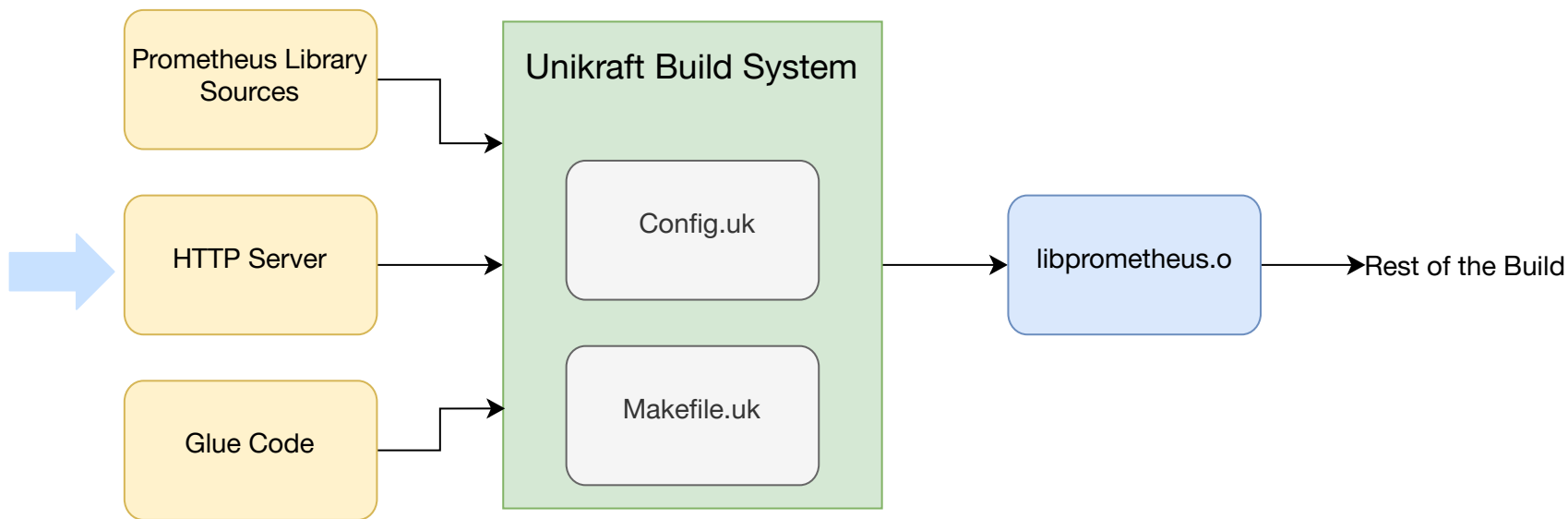
Solution – Prometheus Scraper



Solution – Prometheus Scraper

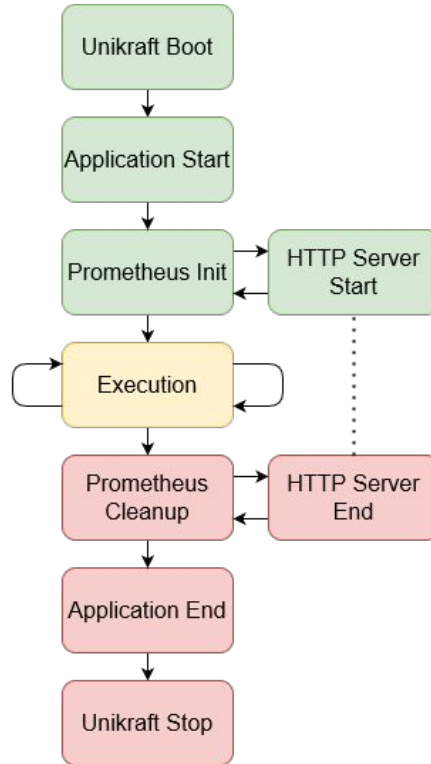


Solution – Prometheus Scraper

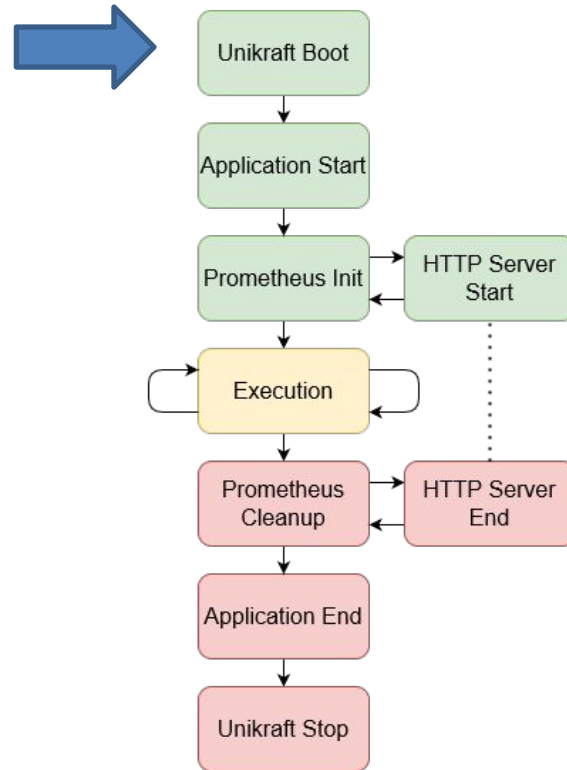


Solution – HTTP Server

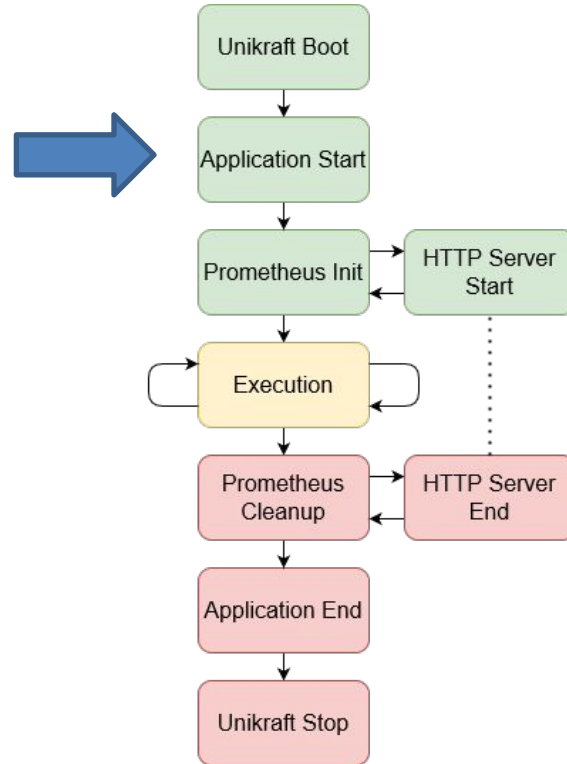
Solution – HTTP Server



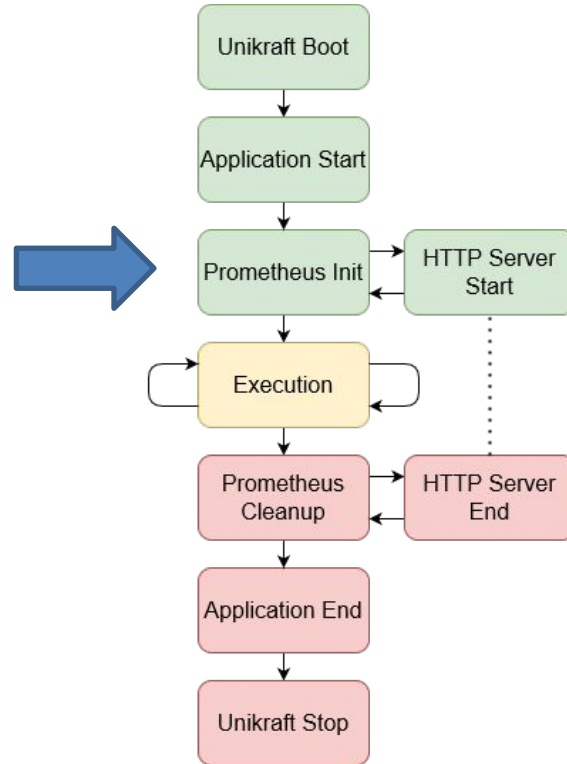
Solution – HTTP Server



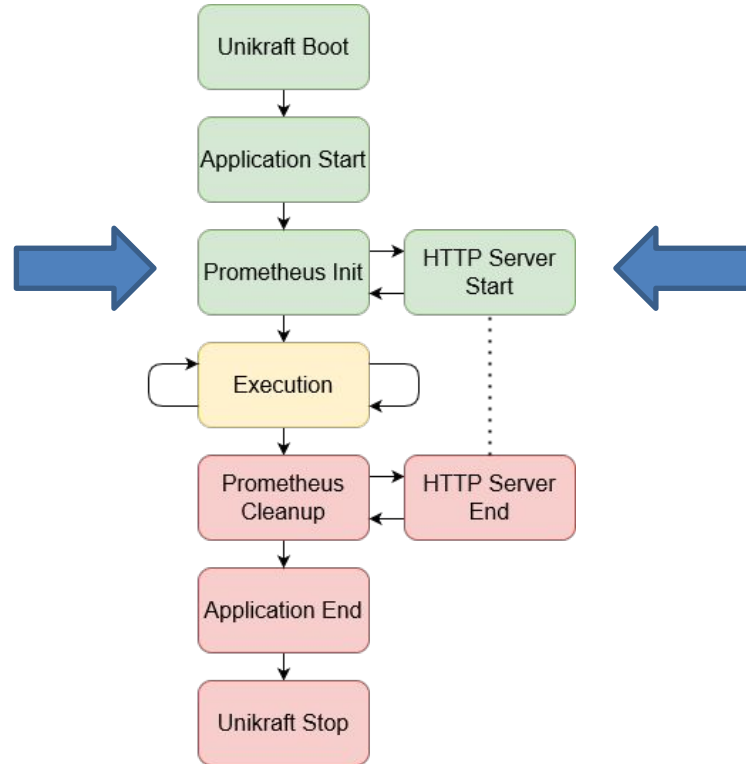
Solution – HTTP Server



Solution – HTTP Server

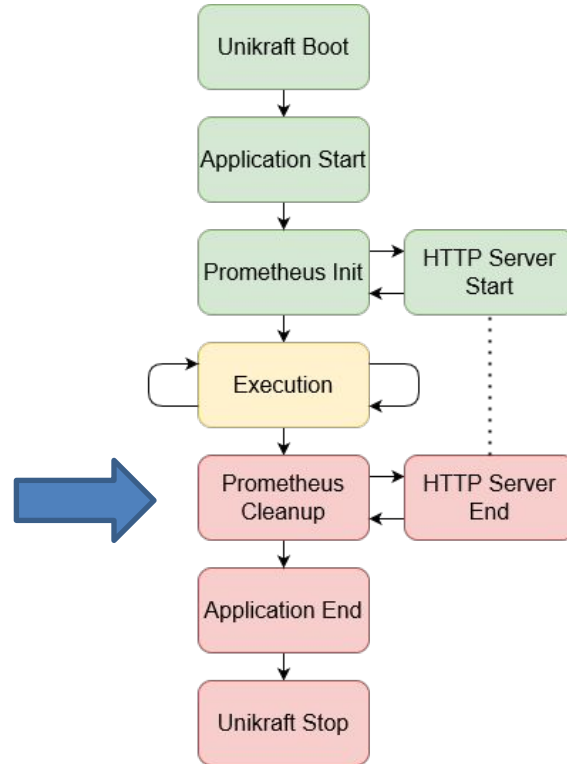


Solution – HTTP Server

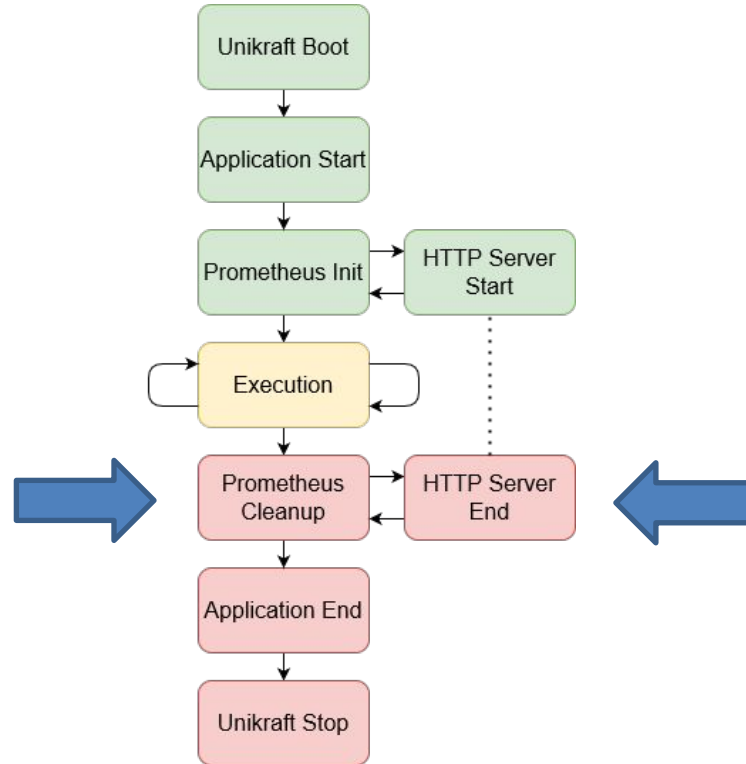




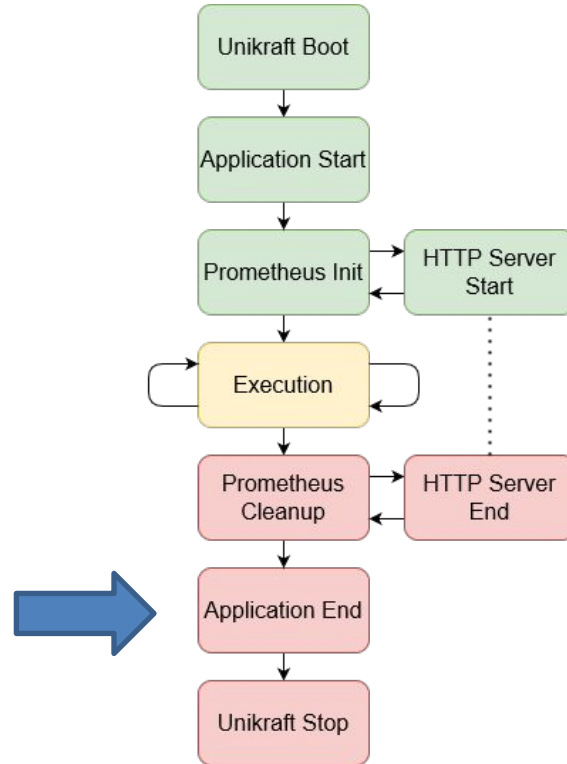
Solution – HTTP Server



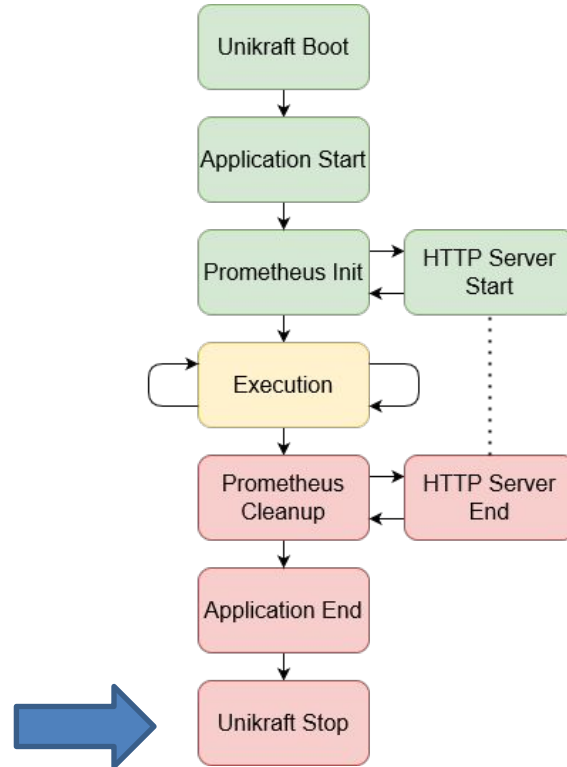
Solution – HTTP Server



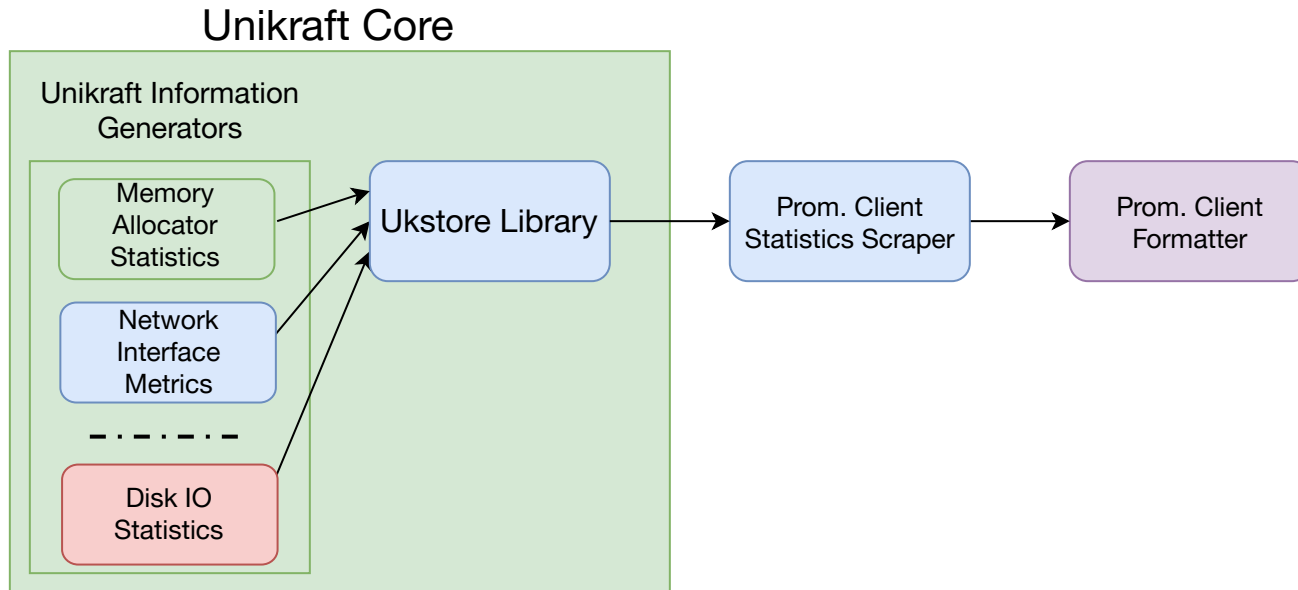
Solution – HTTP Server



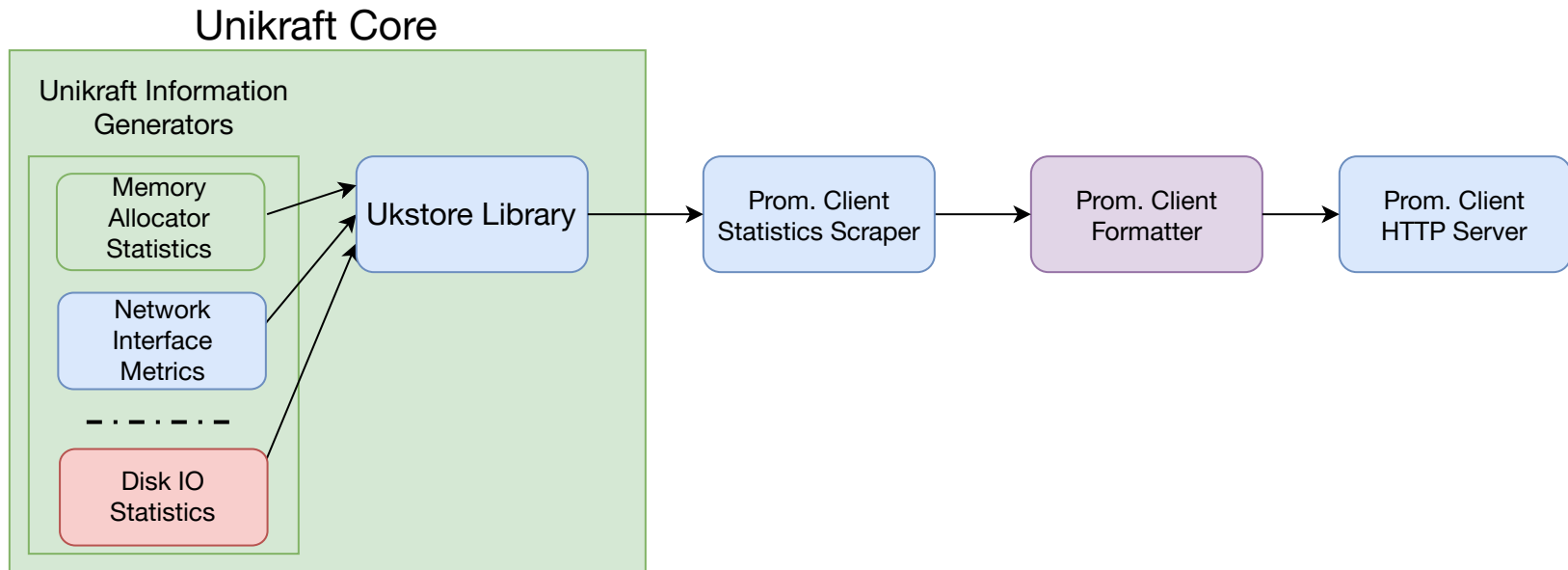
Solution – HTTP Server



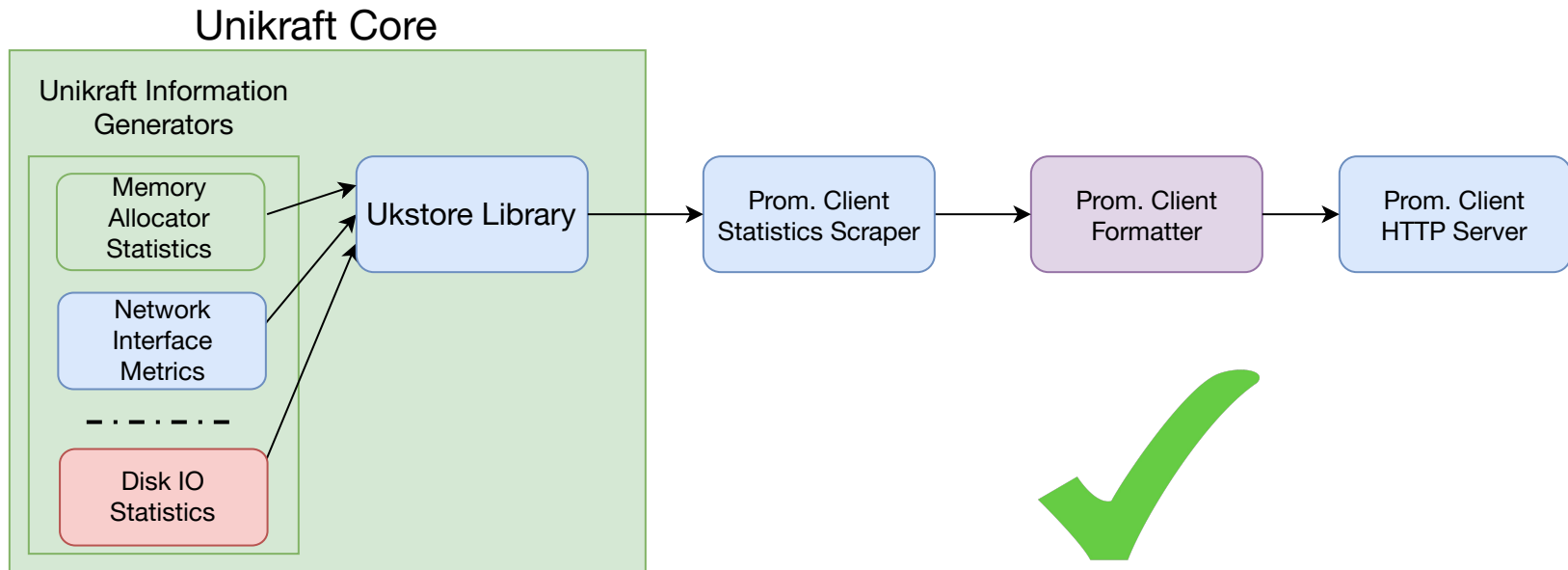
Solution – HTTP Server



Solution – HTTP Server



Solution – HTTP Server



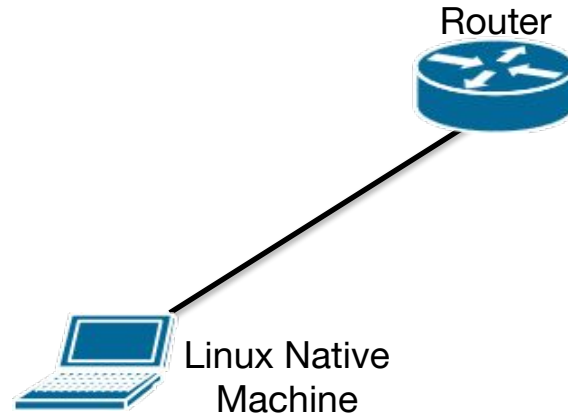
Testing Environment Setup

Testing Environment Setup

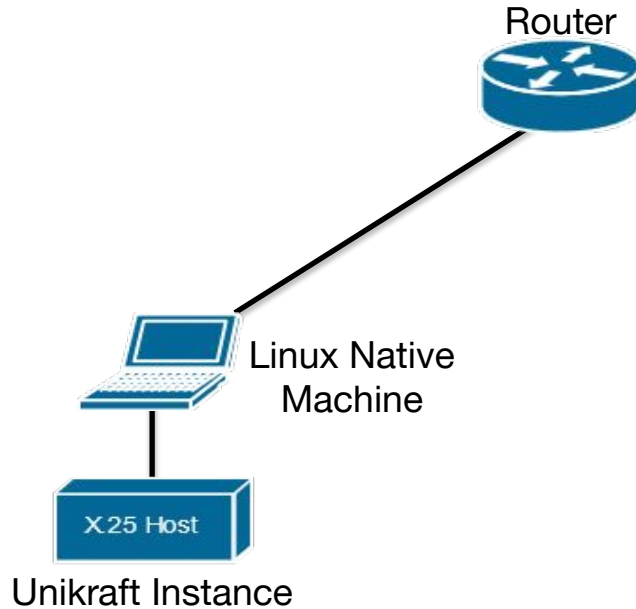
Router



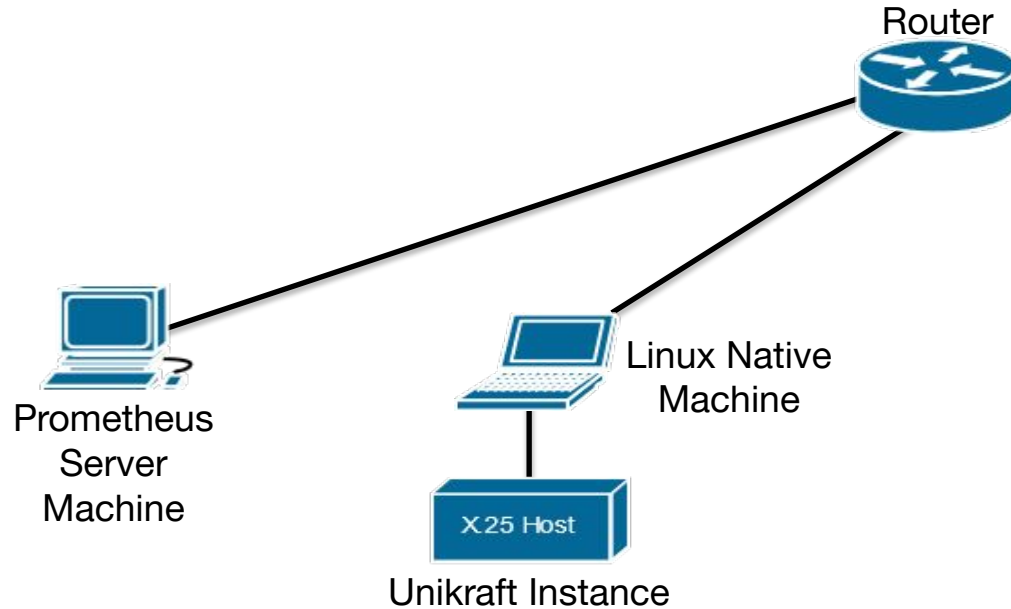
Testing Environment Setup



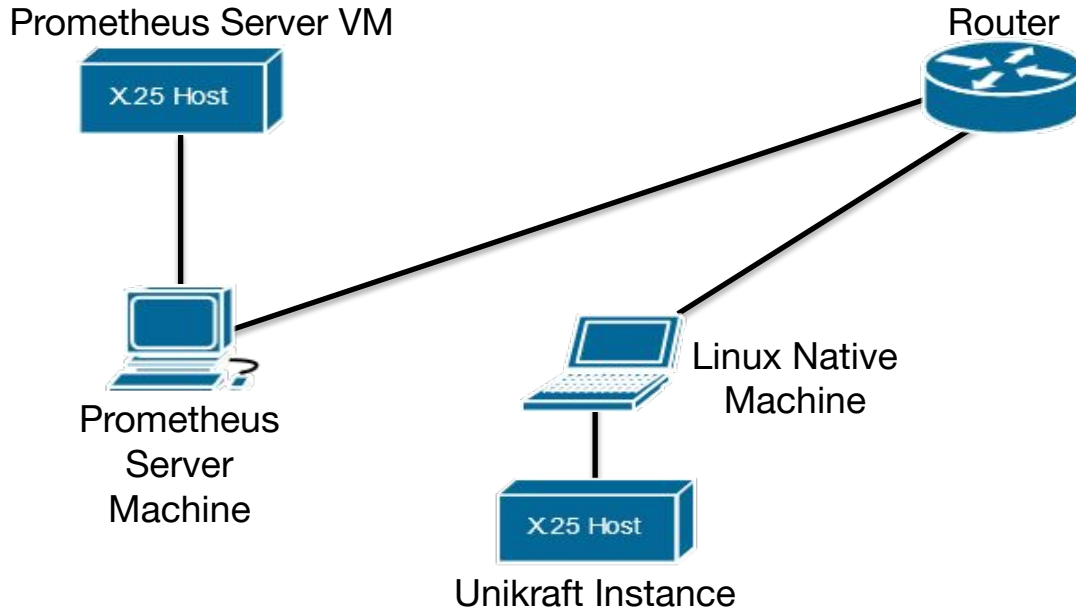
Testing Environment Setup



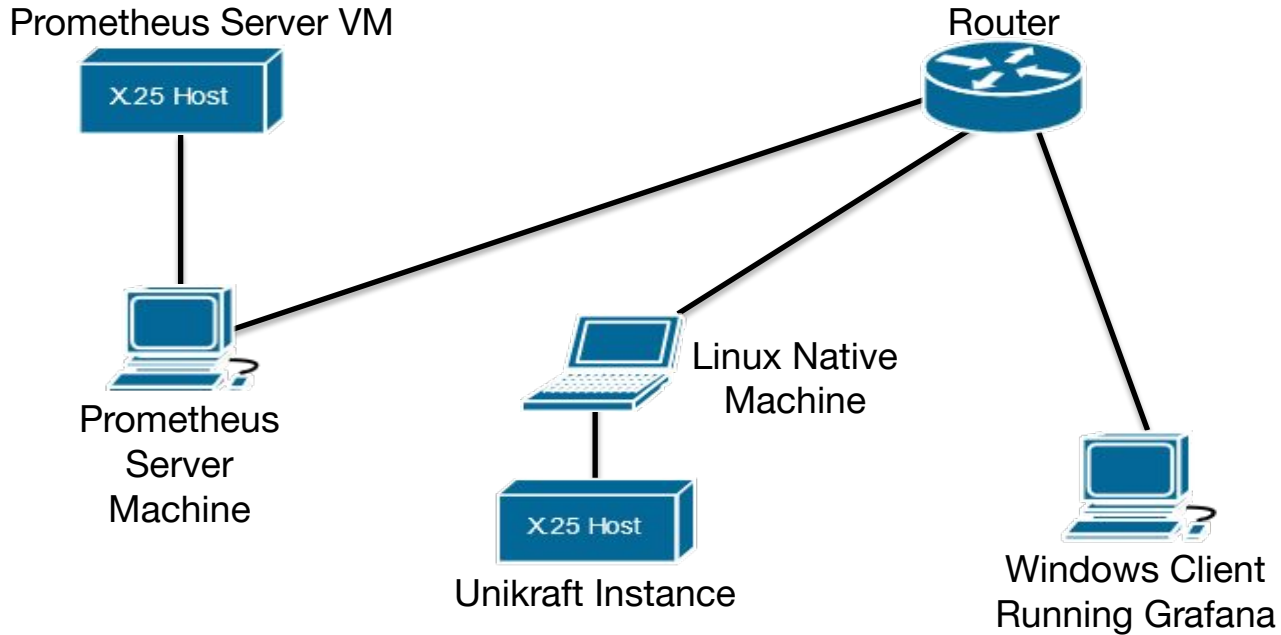
Testing Environment Setup



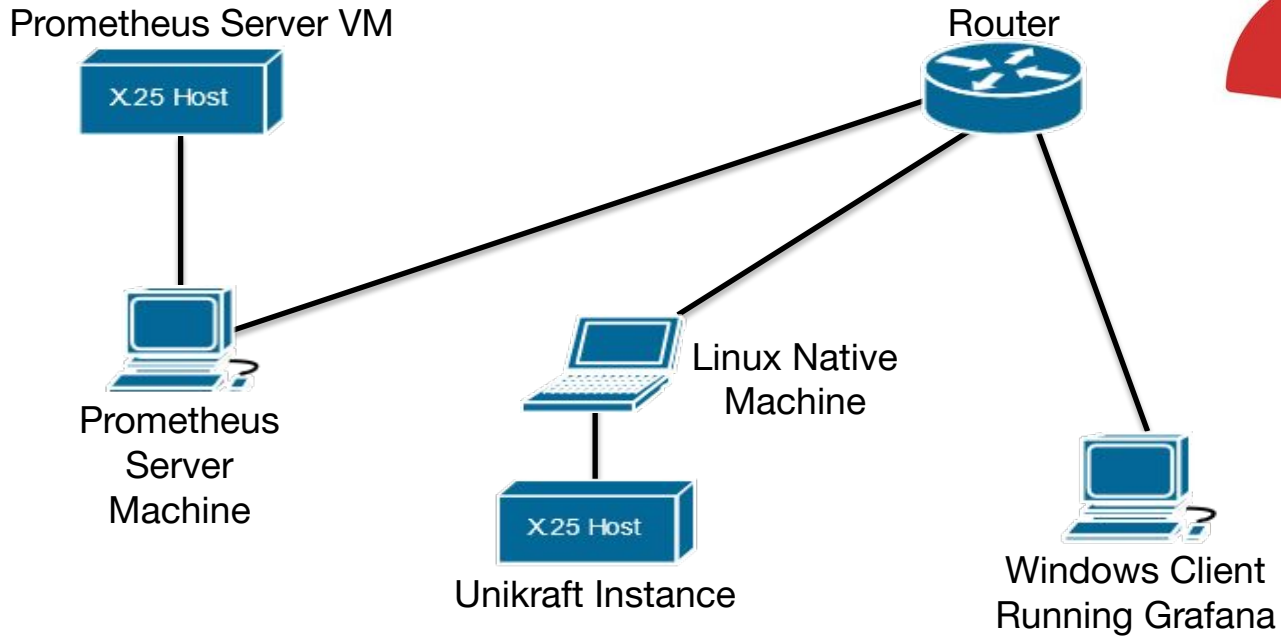
Testing Environment Setup



Testing Environment Setup



Testing Environment Setup



Evaluation Results (I)

```
← → ↺ 🏠 192.168.0.188:9100/metrics

# HELP uk_netdev_tx_bytes numbers all transmit bytes
# TYPE uk_netdev_tx_bytes gauge
uk_netdev_tx_bytes(iface="en0") 250919

# HELP uk_netdev_rx_packets numbers all received packets
# TYPE uk_netdev_rx_packets gauge
uk_netdev_rx_packets(iface="en0") 496

# HELP uk_netdev_tx_packets numbers all transmit packets
# TYPE uk_netdev_tx_packets gauge
uk_netdev_tx_packets(iface="en0") 335

# HELP uk_alloc_max_alloc_size holds the biggest satisfied allocation size
# TYPE uk_alloc_max_alloc_size gauge
uk_alloc_max_alloc_size(alloc="default") 135168

# HELP uk_alloc_last_alloc_size holds the size of the last allocation
# TYPE uk_alloc_last_alloc_size gauge
uk_alloc_last_alloc_size(alloc="default") 4096

# HELP uk_netdev_rx_bytes numbers all received bytes
# TYPE uk_netdev_rx_bytes gauge
uk_netdev_rx_bytes(iface="en0") 65219

# HELP prom_http_requests numbers all prom requests
# TYPE prom_http_requests counter
prom_http_requests 81

# HELP uk_netdev_rx_frame numbers all received frame errors
# TYPE uk_netdev_rx_frame gauge
uk_netdev_rx_frame(iface="en0") 0

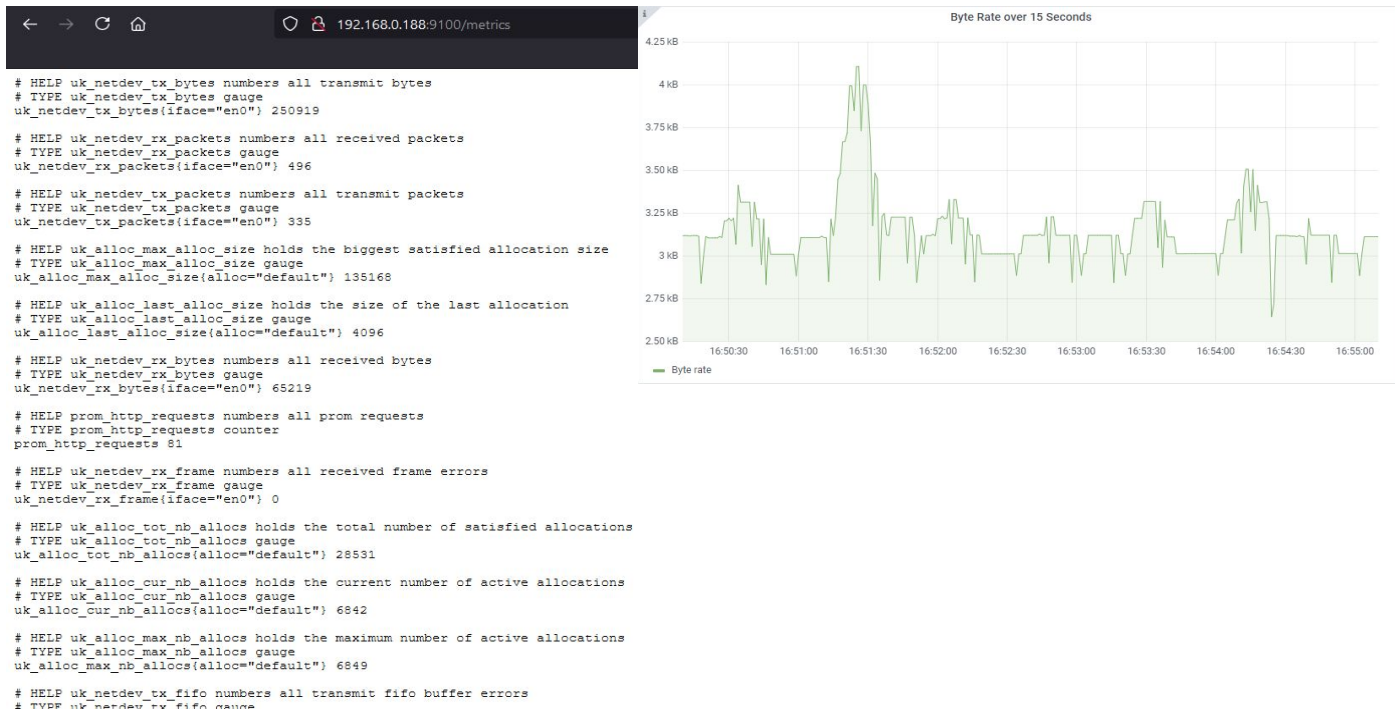
# HELP uk_alloc_tot_nb_allocs holds the total number of satisfied allocations
# TYPE uk_alloc_tot_nb_allocs gauge
uk_alloc_tot_nb_allocs(alloc="default") 28531

# HELP uk_alloc_cur_nb_allocs holds the current number of active allocations
# TYPE uk_alloc_cur_nb_allocs gauge
uk_alloc_cur_nb_allocs(alloc="default") 6842

# HELP uk_alloc_max_nb_allocs holds the maximum number of active allocations
# TYPE uk_alloc_max_nb_allocs gauge
uk_alloc_max_nb_allocs(alloc="default") 6849

# HELP uk_netdev_tx_fifo numbers all transmit fifo buffer errors
# TYPE uk_netdev_tx_fifo gauge
```

Evaluation Results (I)



Evaluation Results (I)



The data arrives
successfully

Evaluation Results (II)



Plots

Evaluation Results (II)



Many plots

Evaluation Results (II)



So many plots

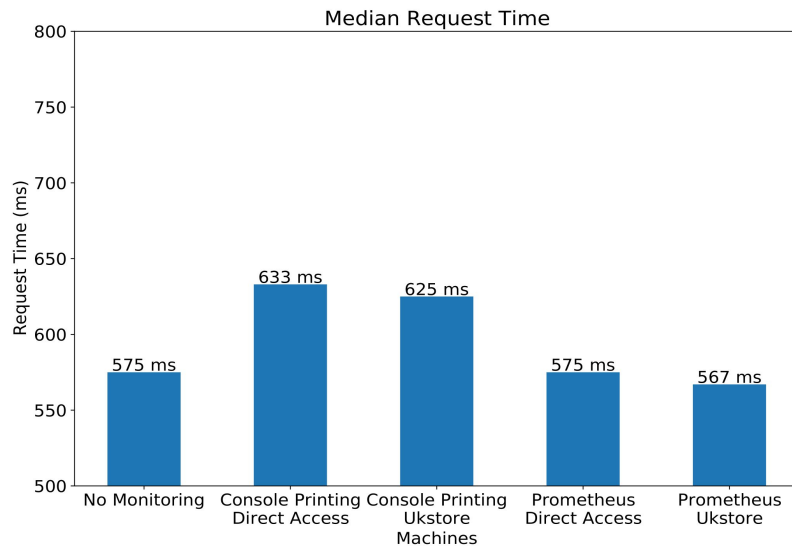
Evaluation Results (II)



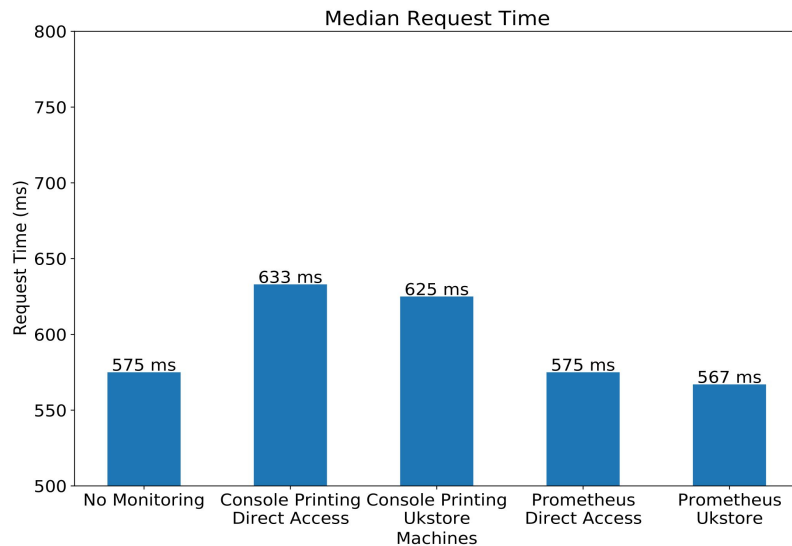
The dream: achieved

Performance Results (I)

Performance Results (I)

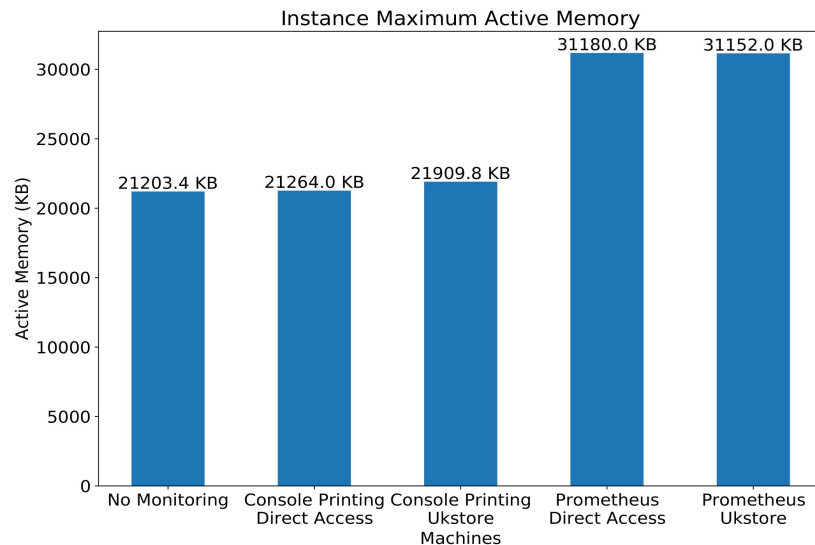
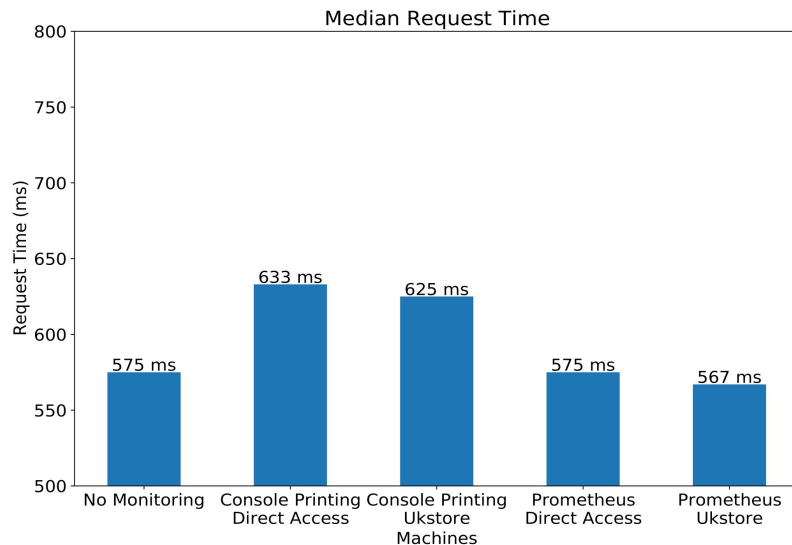


Performance Results (I)



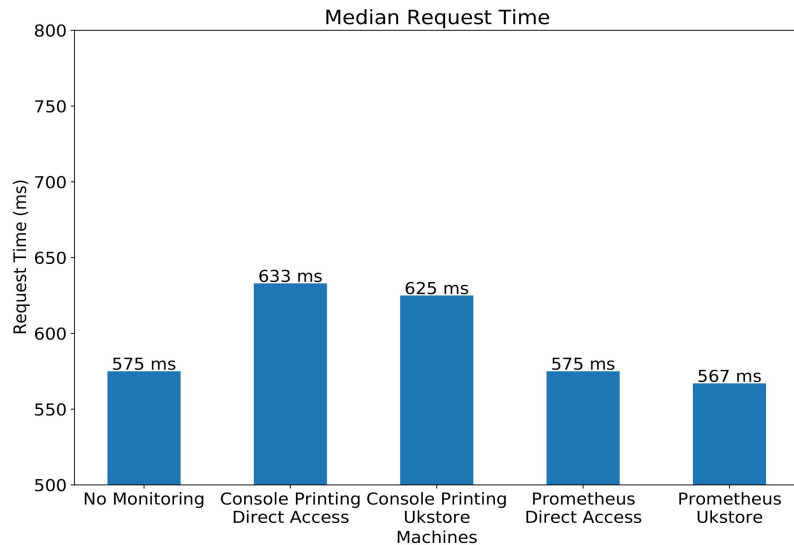
Speed - ideal

Performance Results (I)

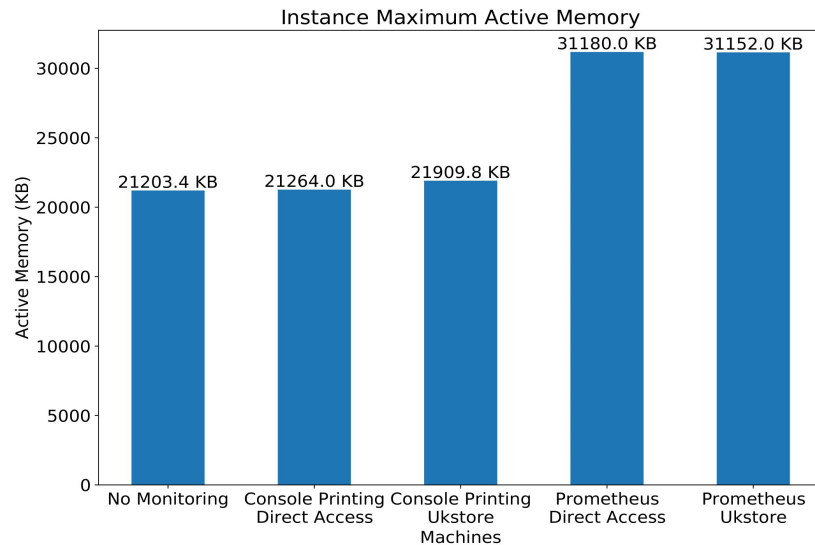


Speed - ideal

Performance Results (I)



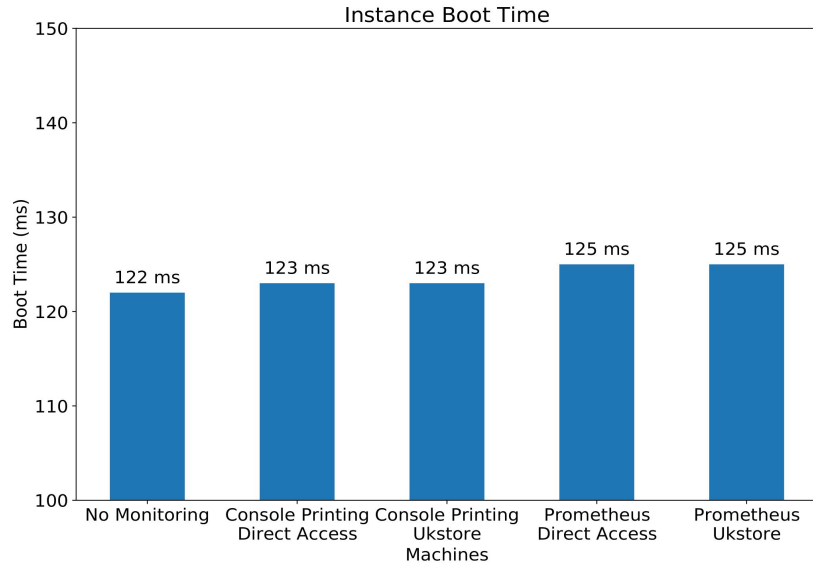
Speed - ideal



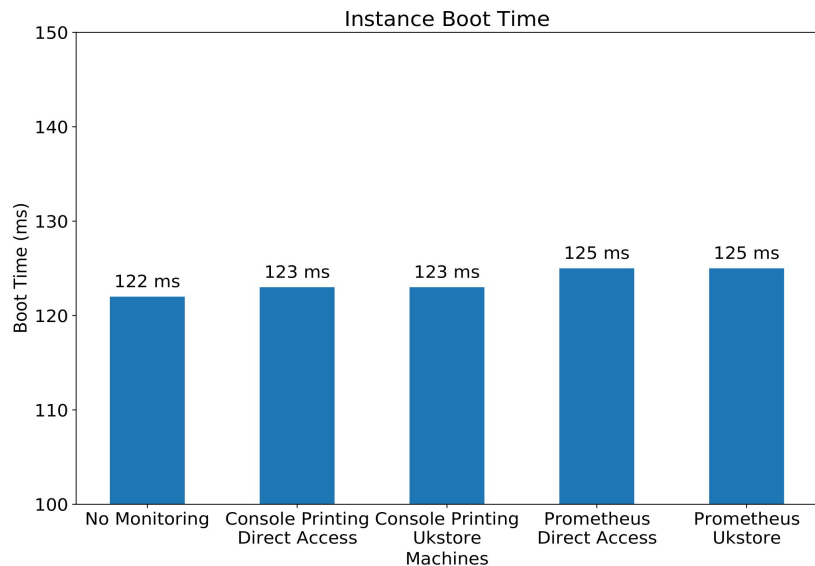
Memory - decent

Performance Results (II)

Performance Results (II)

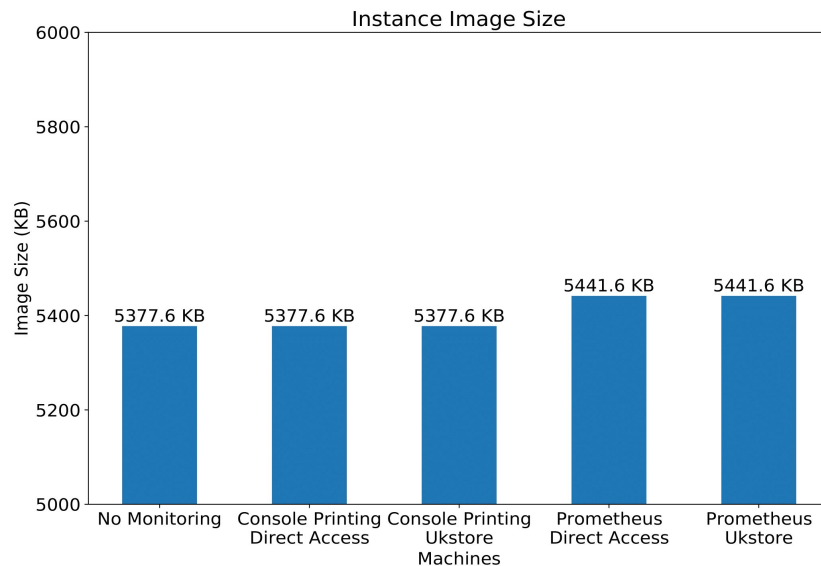
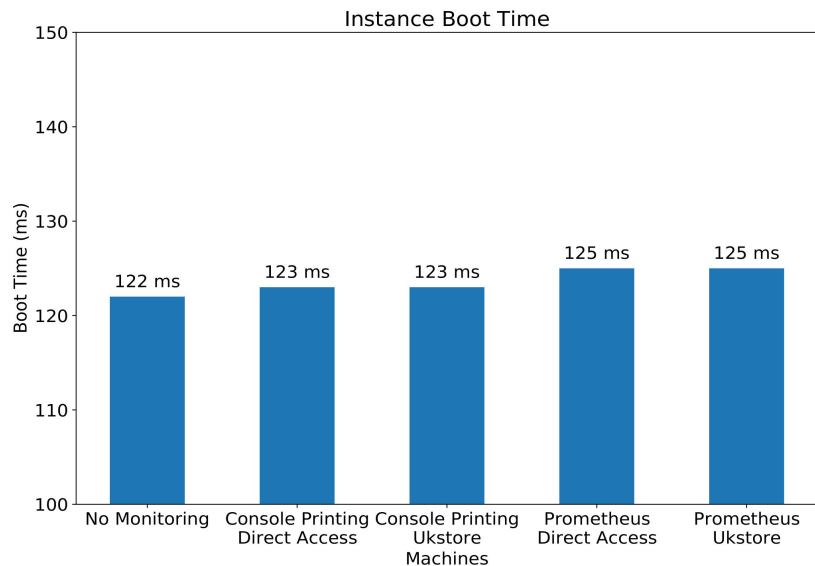


Performance Results (II)



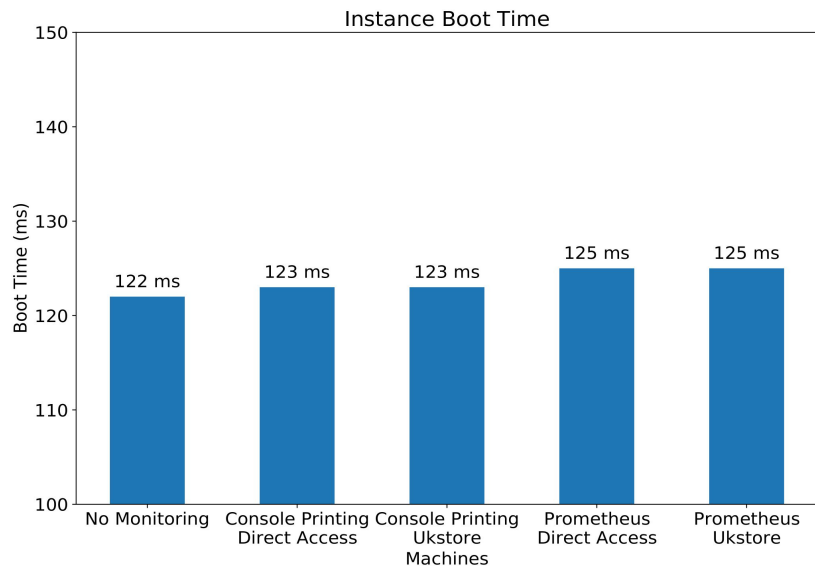
Boot - ideal

Performance Results (II)

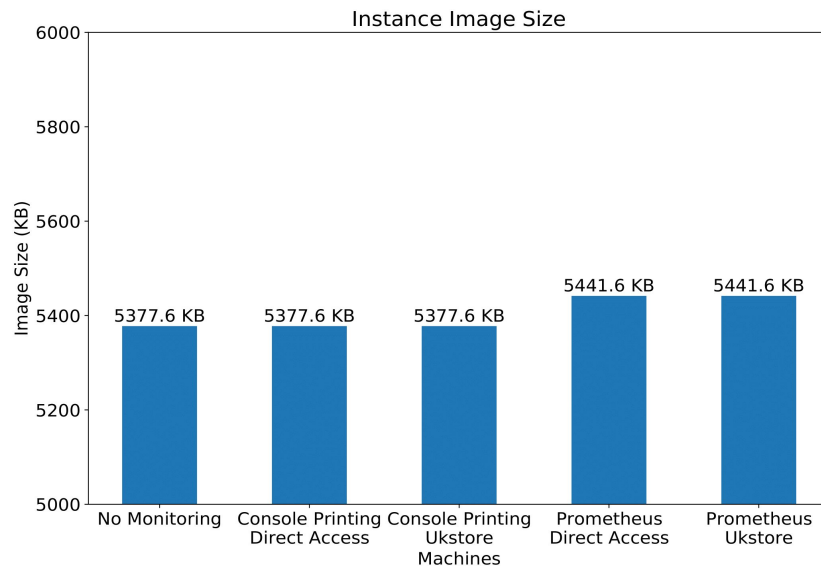


Boot - ideal

Performance Results (II)



Boot - ideal



Size - ideal

Conclusions

Conclusions

→ Prometheus & Unikraft, a good match

Conclusions

- Prometheus & Unikraft, a good match
- Small impact with any settings

Conclusions

- Prometheus & Unikraft, a good match
- Small impact with any settings
- Decoupling maintained

Further Work – Overview

Further Work – Overview

- Instrument the rest of Unikraft
- Export more metrics

Further Work – Overview

- Instrument the rest of Unikraft
- Export more metrics
- Integration with ProcFS

Further Work – Overview

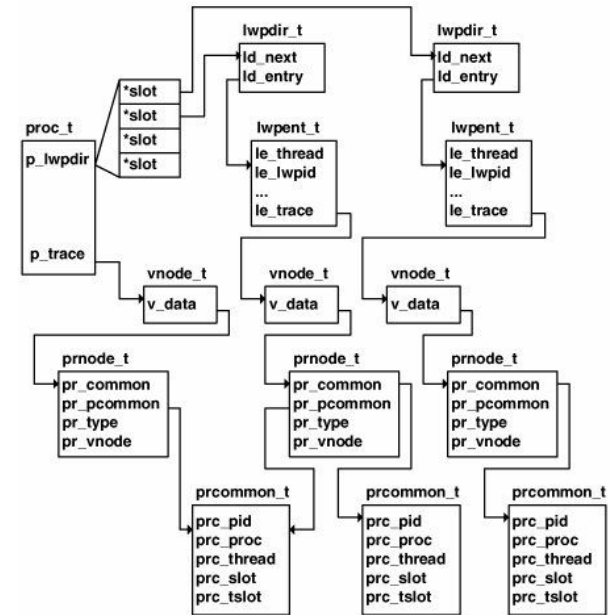
- Instrument the rest of Unikraft
- Export more metrics
- Integration with ProcFS
- Integration with Kubernetes

Further Work – Overview

- Instrument the rest of Unikraft
- Export more metrics
- Integration with ProcFS
- Integration with Kubernetes
- Integration with the Unikraft CI/CD

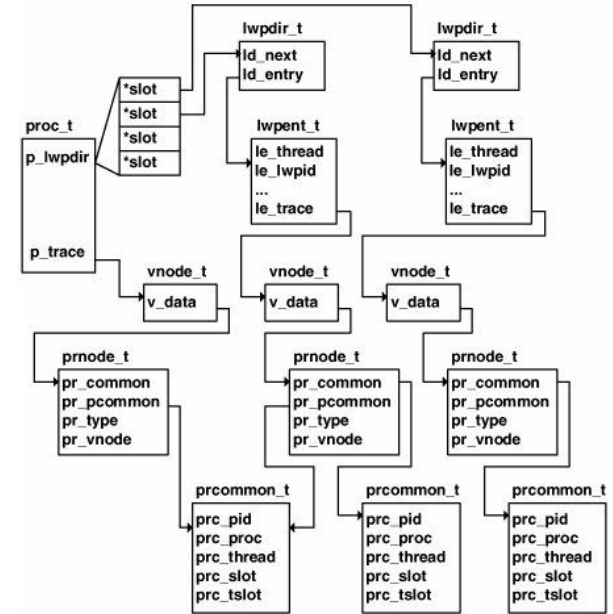
Further Work – ProcFS

Further Work – ProcFS



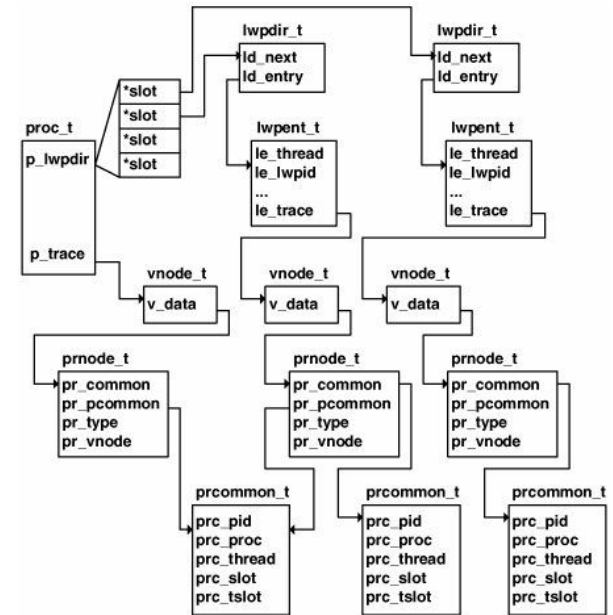
Further Work – ProcFS

→ Bring in the full functionality of ProcFS



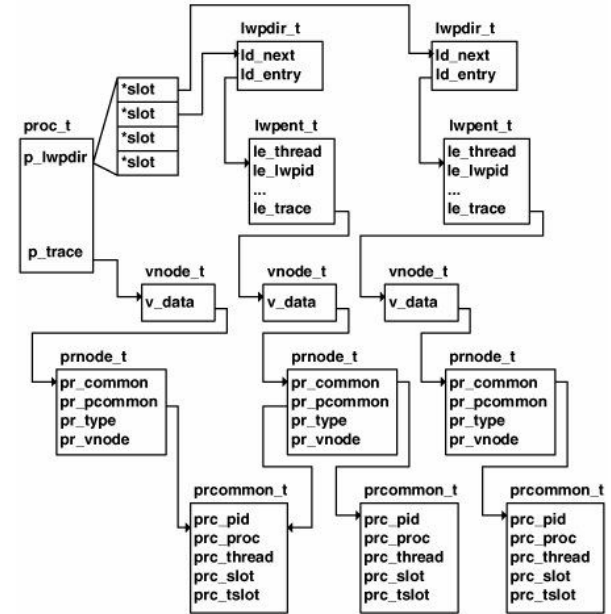
Further Work – ProcFS

- Bring in the full functionality of ProcFS
- Seamless integration with other exporters



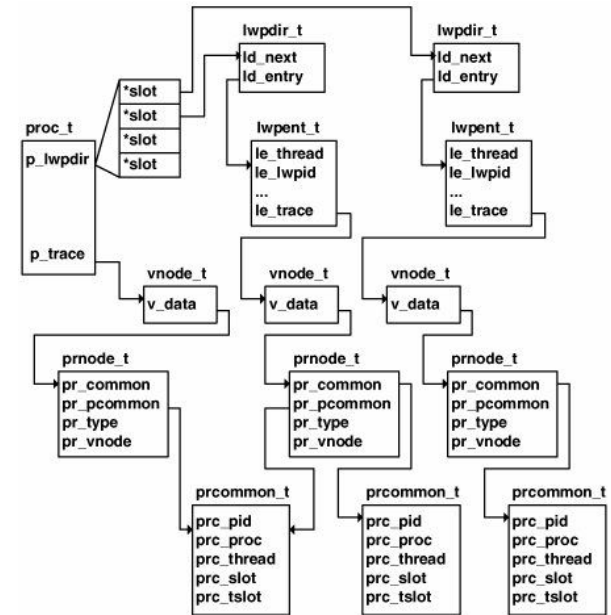
Further Work – ProcFS

- Bring in the full functionality of ProcFS
- Seamless integration with other exporters
- Familiar to use



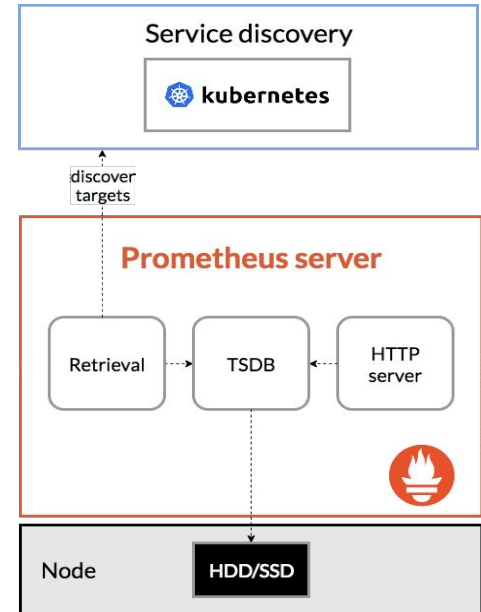
Further Work – ProcFS

- Bring in the full functionality of ProcFS
- Seamless integration with other exporters
- Familiar to use
- Compatibility with other applications



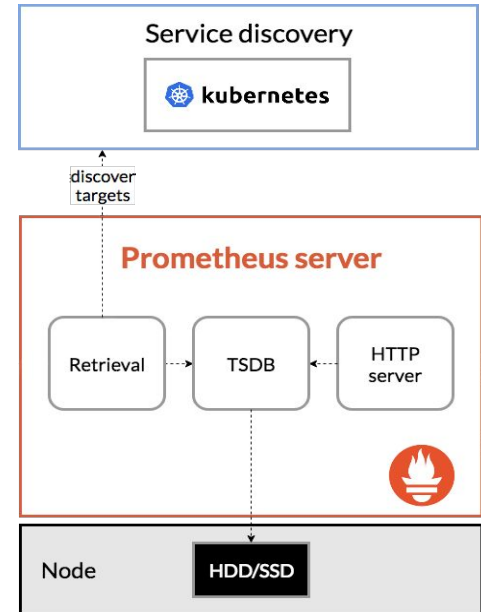
Further Work – Kubernetes

Further Work – Kubernetes



Further Work – Kubernetes

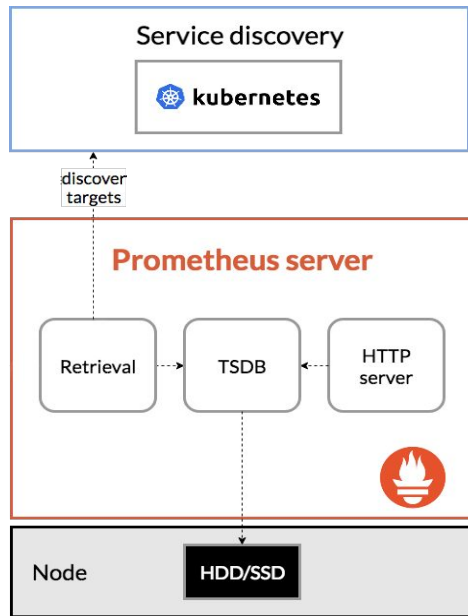
→ Unikraft inside Kubernetes, already underway



Further Work – Kubernetes

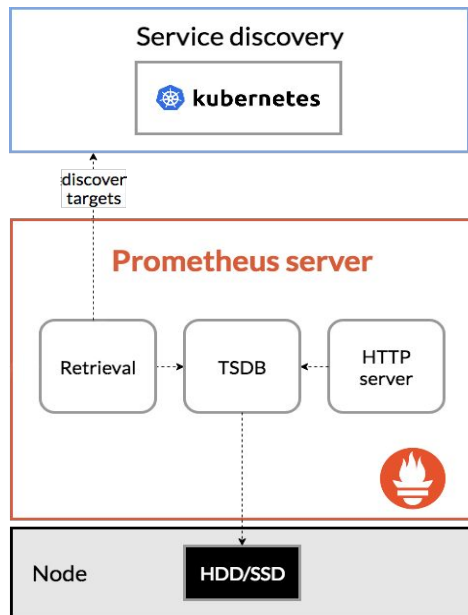
- Unikraft inside Kubernetes, already underway
- Prometheus + Kubernetes = ❤️

```
$ helm upgrade prometheus stable/prometheus \  
  --install \  
  --namespace monitoring \  
  --set server.service.type=NodePort \  
  --set server.service.nodePort=30090 \  
  --set server.persistentVolume.enabled=false
```



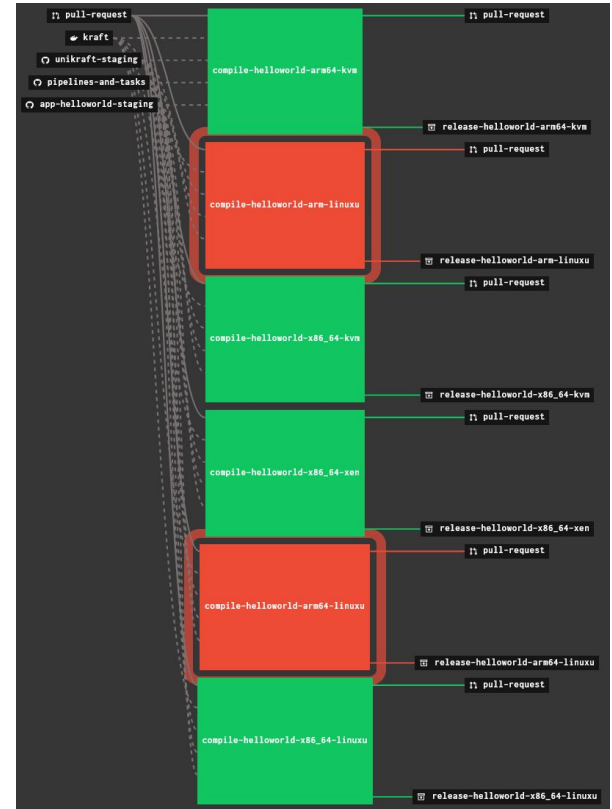
Further Work – Kubernetes

- Unikraft inside Kubernetes, already underway
- Prometheus + Kubernetes = ❤️
- Autoscaling based on Prometheus



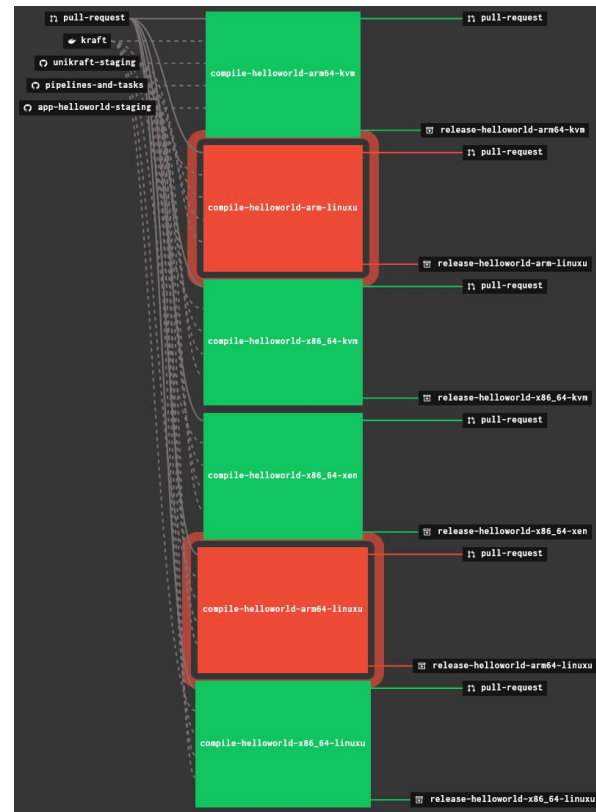
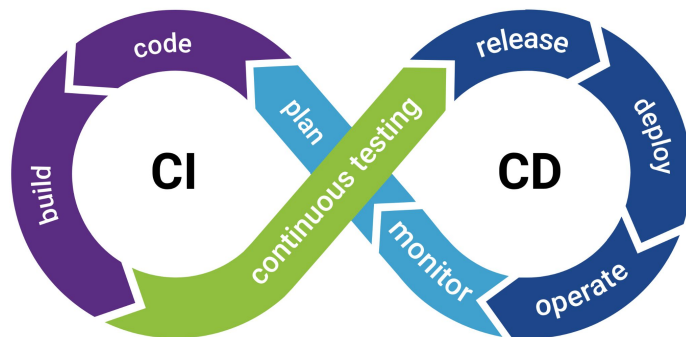
Further Work – Unikraft CI/CD

Further Work – Unikraft CI/CD



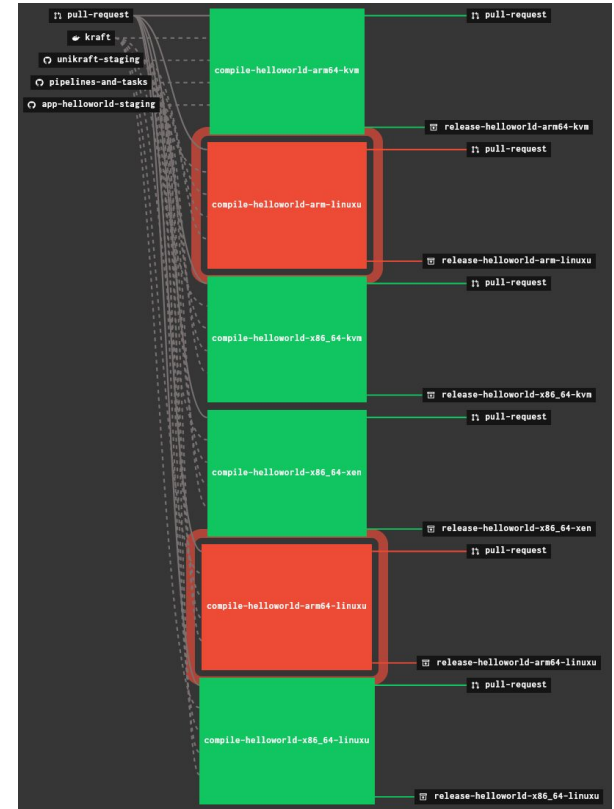
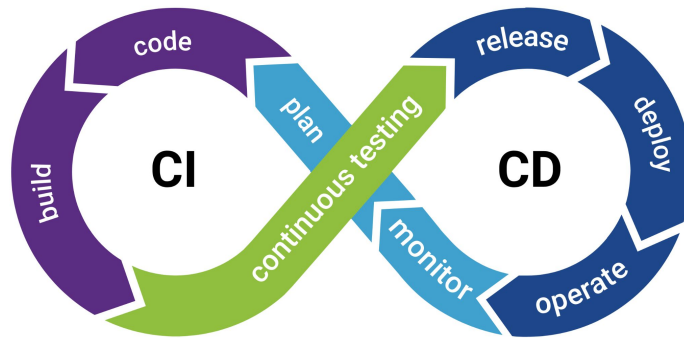
Further Work – Unikraft CI/CD

→ Verify performance of builds after commits



Further Work – Unikraft CI/CD

- Verify performance of builds after commits
- Have a reliable way of testing instances for output



THANKS !

Do you have any questions?



<https://unikraft.org>

Resources

https://www.pinclipart.com/pindetail/iTmobmo_monkey-face-emoji-clipart-monkey-emoji-png-download/

<https://flylib.com/books/en/2.830.1.31/1/>

https://en.wikipedia.org/wiki/File:Green_tick.png

<https://www.synopsys.com/glossary/what-is-cicd.html>

<https://prometheus.io/docs/introduction/overview/>

<https://www.vecteezy.com/vector-art/376574-question-mark-vector-icon>

<https://app.diagrams.net/>