The Path to a Self Assembling Architecture :: A Consul <3 Story

Zachary Schneider

Senior Product Manager
SaaS Operations

Who?

Where have I been, what have I done?

Lead Operations Engineer

- Cloud Sites
- Cloud DNS
- Cloud Databases

Operations Architect

- Boundary Enterprise (Flow)
- Boundary Premium (Metrics)

Senior Product Manager :: SaaS Ops

- TrueSight Pulse (Boundary)
- TrueSight Intelligence (New Offering)

Special Thanks!

Jesse Hodges @gjesse Design support and implementation of the Consul integration at the service level

Agenda

1. An Ideal State
2. The Path <3
3. The Benefits
4. What's Next?
5. Resources

01

An Ideal State

Do You Remember When?

- Installing software often did not require configuration
- The Package Manager
- One command to install any piece of software

Demo

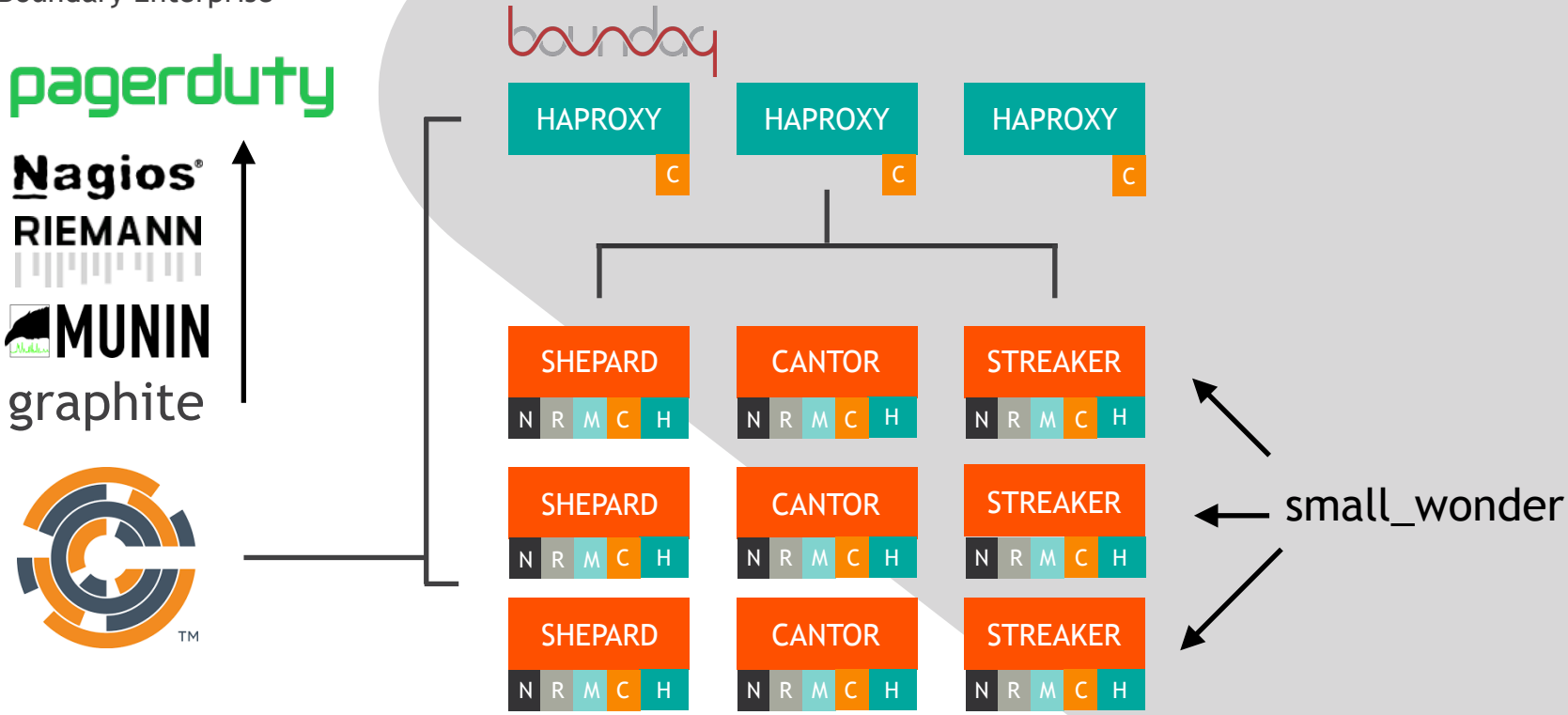
- When a service is installed it:
  - Starts
  - Registers its name, port, tags, and health checks with Consul
- When a service is stopped/uninstalled it:
  - Deregisters itself from Consul
- Discovery
  - In process load balancing HTTP client seeded via Consul
- Health
  - Each service exposes an HTTP health endpoint
  - Consul checks these endpoints

02

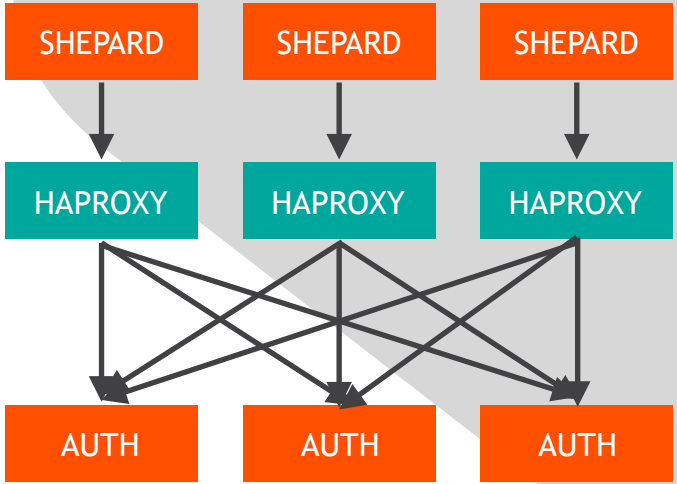
The Path <3

The Beginning

Boundary Enterprise



HAProxy Fan Out



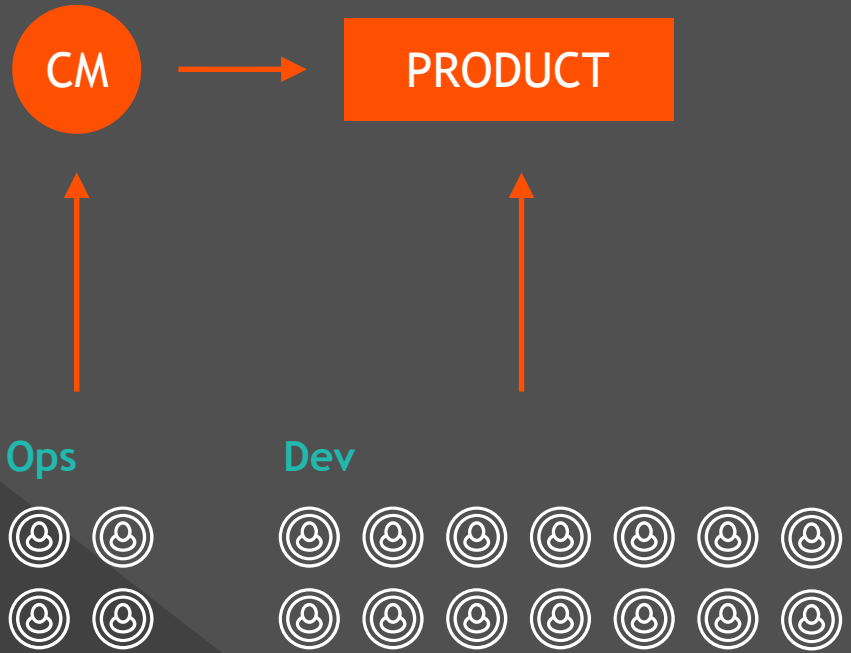
Customers derive value from product not config management

Configuration Management code base was

75%

the size of product code base.

(Disclaimer) Some of this code also managing real hardware, and systems software



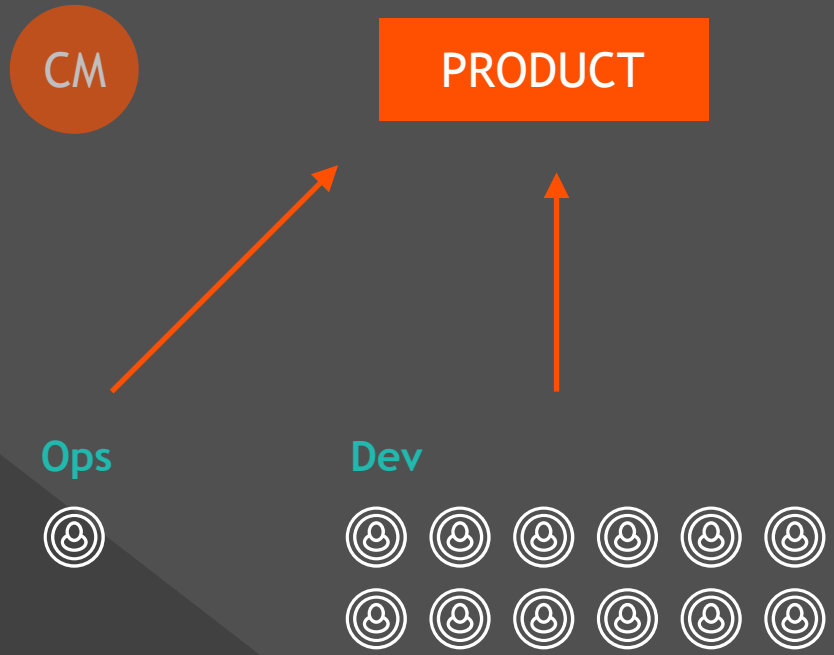
Configuration management is a delivery vehicle for unfinished work!

That sucks but is sustainable until…

- The pivot, must merge two software stacks
- Down to one operator :(
- We need to be able to deploy on premise
  - Product to be managed by on site customer personnel

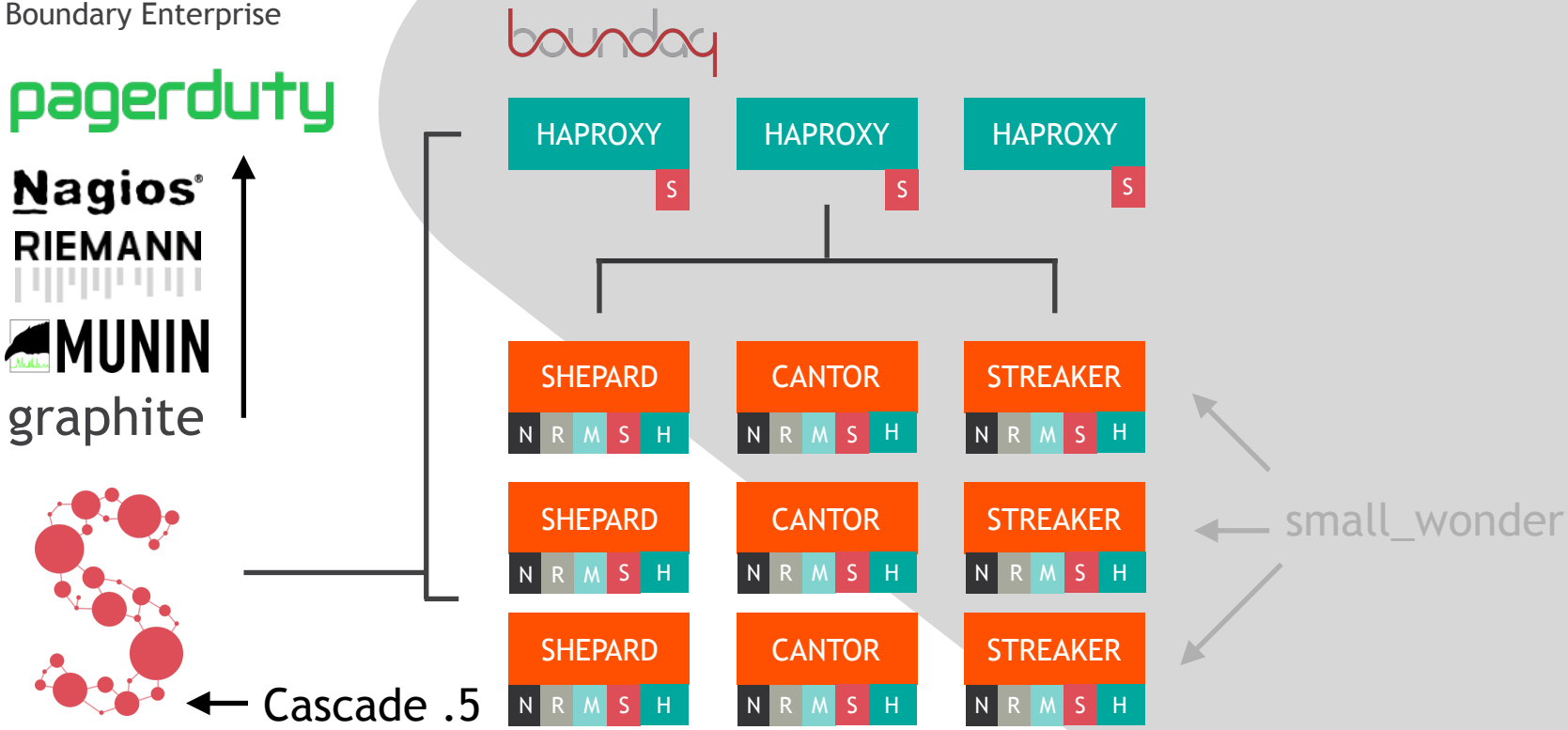
Status quo exit strategy

Configuration management must be minimized or eliminated. Only do work that adds value for the customer.



Deployment :: Serf + Cascade .5

Boundary Enterprise



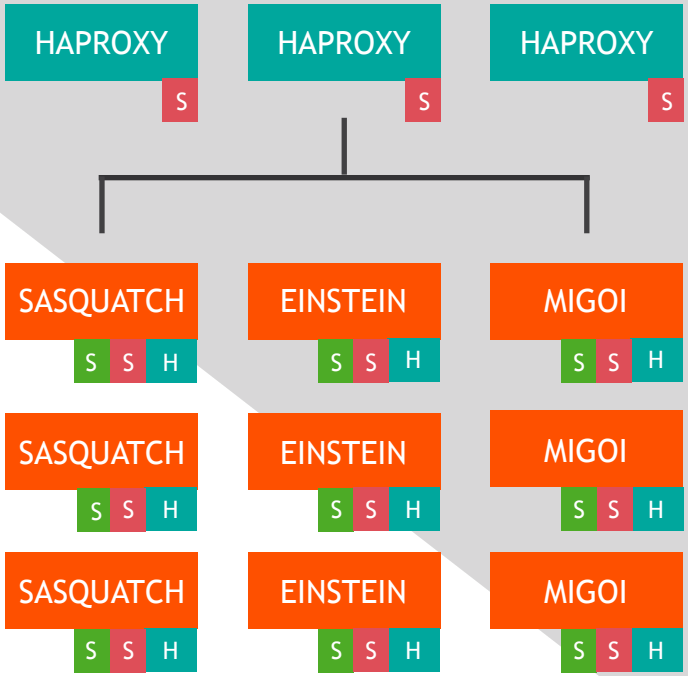
This is Better … However

- Monitoring
  - Utilizing three separate monitoring systems
  - There is no single authority for systems and application health
- HAProxy Fanout
  - A lot of load balancer stubs
  - A lot of config management driving these
- Deployment
  - Serf does not provide the ability to ensure that only a single roll out is occurring at a given time

Monitoring :: Boundary + Sensu

Boundary Premium

← Cascade .5



METRICS

Consul <3

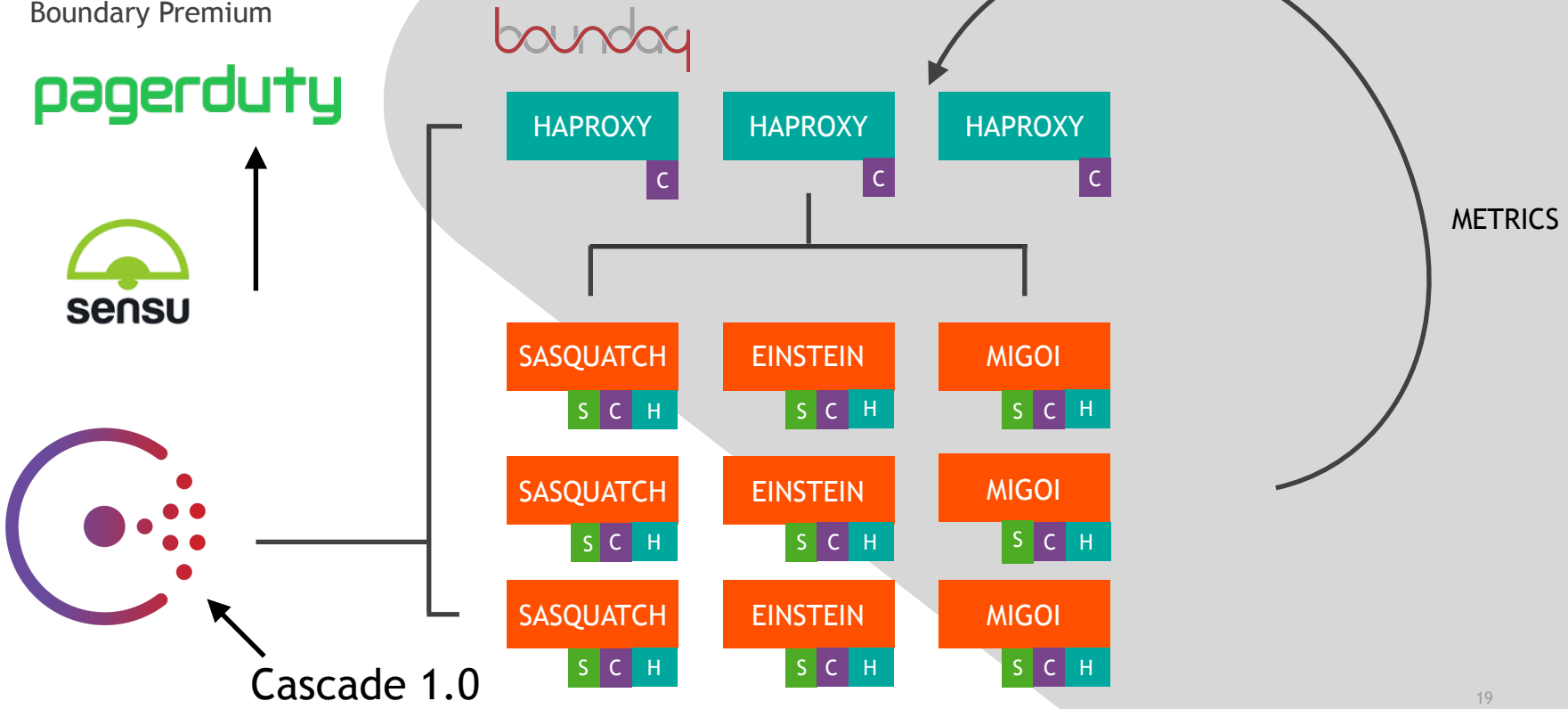
- Reservations
  - Already invested in Zookeeper
    - Ingest load sharing coordination, and ephemeral state
  - Serf event/query API arguably better than Consul watches for orchestration
  - No node (system) tags

- Selling Points
  - ZK is moot due to Kafka broker requirements
  - I can work with the watch API
  - Batteries included service discovery
  - Batteries included health checks
  - HTTP API, huge for accessibility
  - Can register Cascade as a service and tag service with node roles

- SOLD!

Consul Adoption <3

Boundary Premium

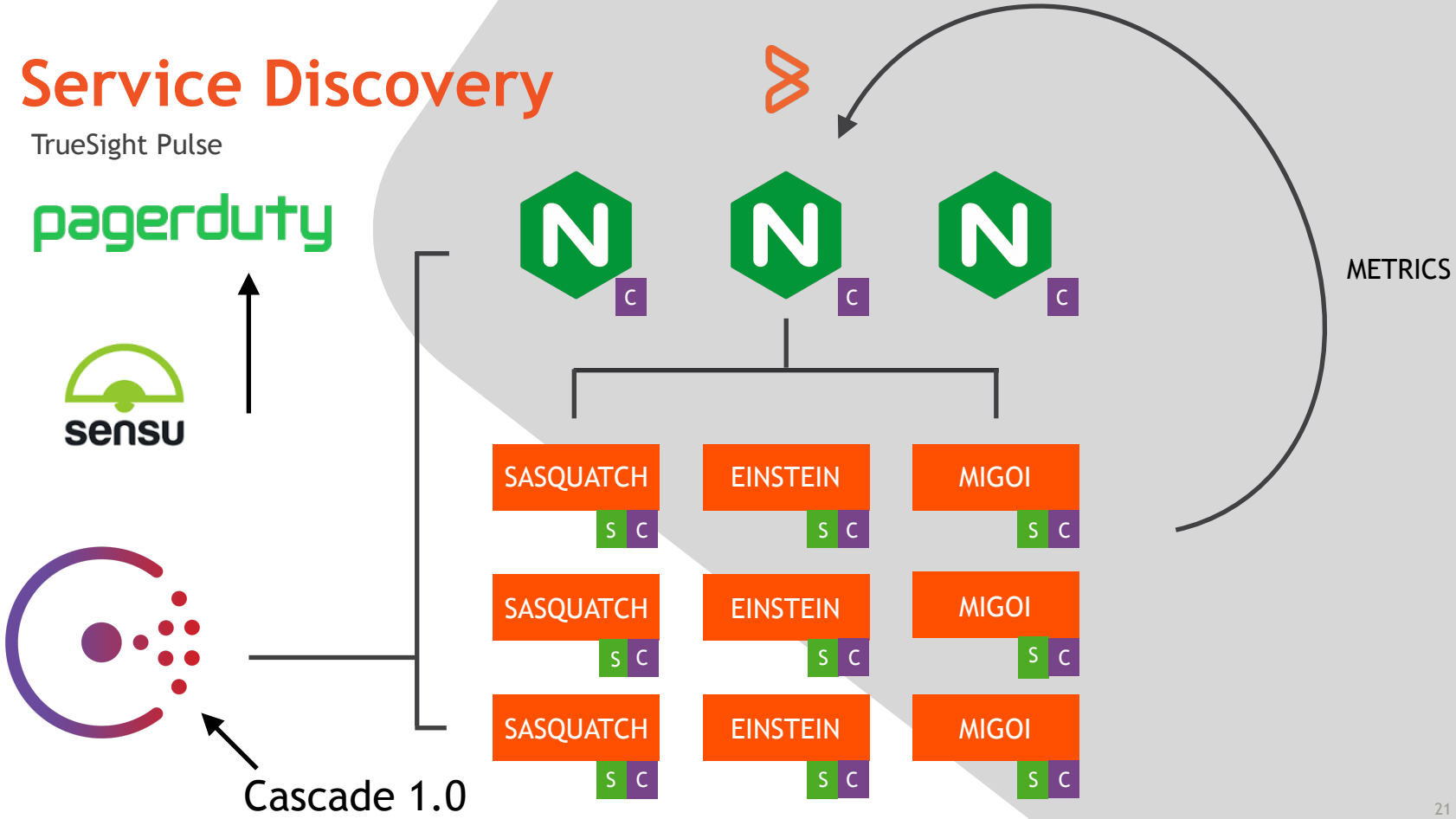


Service Discovery

- All services must register ports, tags, names, ids and health checks with Consul on startup
- All services must deregister on shutdown
- Services that communicate with each other via HTTP must do so with an in process load balancing HTTP client
  - The node:port list must be populated from Consul and changes monitored for via a service health watch

Service Discovery

TrueSight Pulse

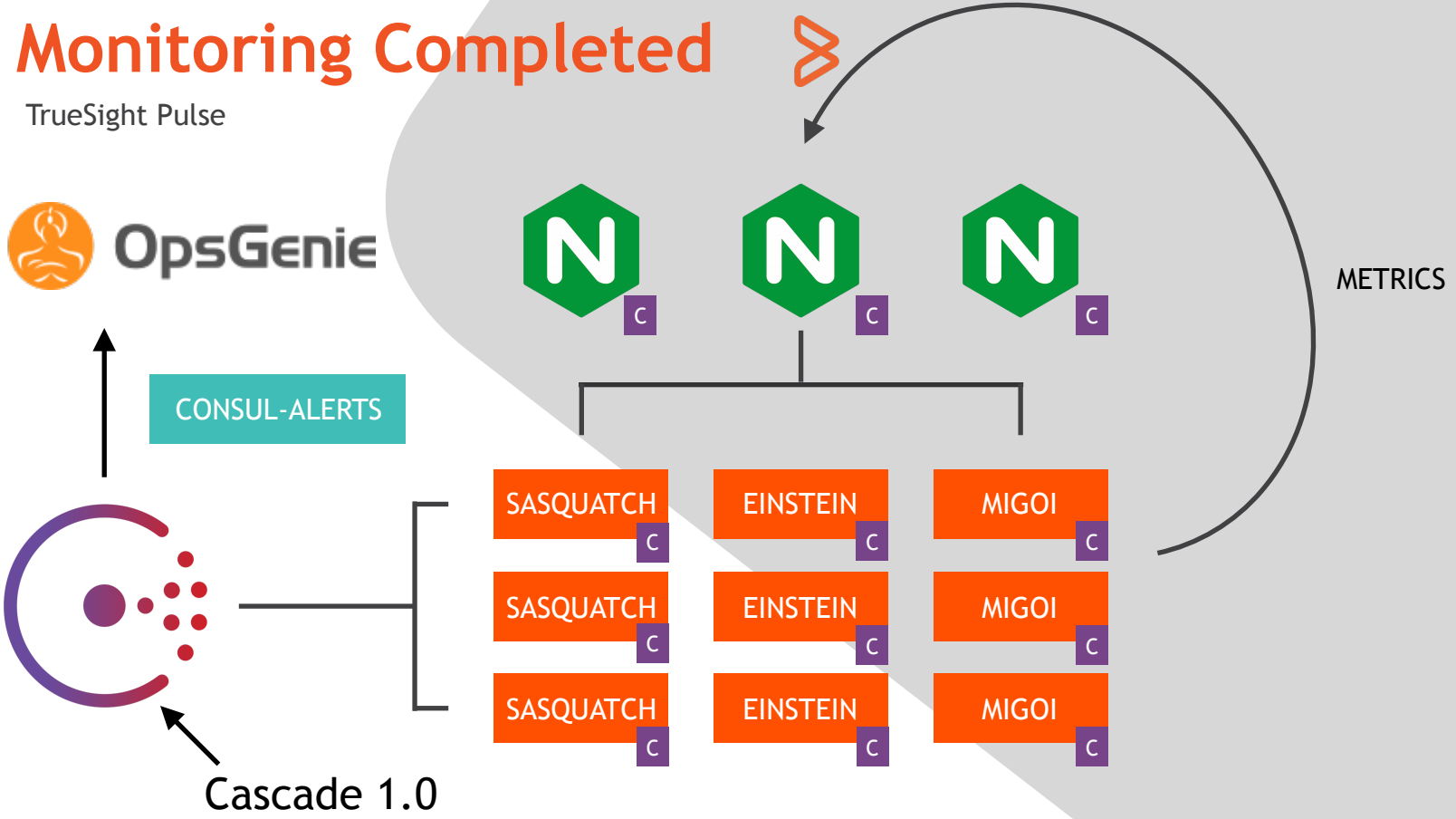


Monitoring

- Services have already registered their health checks with Consul
- That means …
  - We just have to shuttle the open and close alerts somewhere!
- Consul-alerts to the rescue!
  - A small open source multi-node service that shuttles Consul health faults to your favorite pager service!
  - We chose OpsGenie
    - For an important reason: policy matching on content for delivery to various teams and escalation policies

Monitoring Completed

TrueSight Pulse



03

The Benefits

Finished Work

- With service supervision architecture is fault tolerant
- Adding capacity: bootstrap node, install software
- Consul is our infrastructure API
  - Same in AWS and On Prem
- Architecture is sustainable
- Config management is minimal

Chef Service DSL

```
platform_service 'festoon' do
  action :enable
  ports 'admin' => 11601, 'jmx' => 11602, 'service' => 11600
  keys 'cerberus_public' => 'cerberus.public.pem'
  svwait 'cerberus-api', 'zookeeper-server', 'consul'
end
```

04

What’s Next

What's Next?

- Configuration
  - Static values (only values that change per environment)
    - Store in Consul
    - Loaded by service on startup merged with local config
  - Dynamic values
    - Store in Consul
    - Service watches for changes, reconfigure on change
  - Secrets
    - Store in Vault
    - Service integration with Vault
- Deployment
  - We are now scheduler ready
  - Nomad?

05

Resources

Resources

- Service Integration
  - https://github.com/boundary/dw-consul
    - Tools for integrating with drop wizard applications
  - https://github.com/OrbitzWorldwide/consul-client
    - Java Consul Client
- Load Balancer Integration
  - https://medium.com/@sigil66/dynamic-nginx-upstreams-from-consul-via-lua-nginx-module-2bebc935989b
- Deployment and Orchestration
  - https://github.com/boundary/cascade
    - Transitional orchestration shell
  - https://github.com/boundary/chef-cascade
    - Custom chef client
    - Integrates cascade roles, chef code delivery via packages
- Health
  - Dependency Mapped Health Checks https://github.com/sigil66/Presentations/blob/master/State%20of%20the%20Industry%20in%20Application%20Health.pdf

Thank You

Bring IT to Life.™