Implementing NGINX Microservice Architectures with OpenShift

December 15, 2016

NGINX

Christopher Stetson

Chief Architect,
NGINX

MORE INFORMATION AT
NGINX.COM

Agenda

- A Bit of History
- The Big Shift
- The Networking Problem
  - Service Discovery
  - Load Balancing
  - Secure & Fast Intercommunication
- Architectures
- Issues

A Bit of History

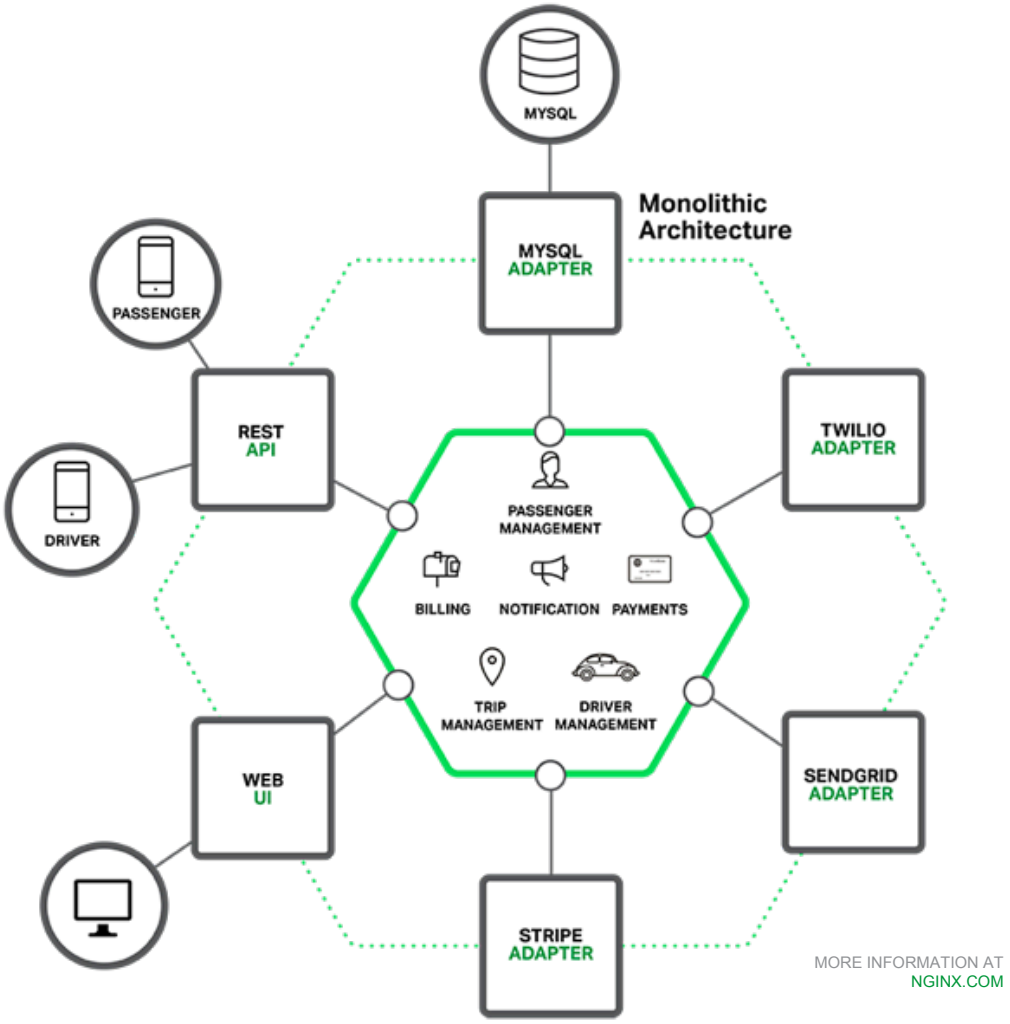
Red Hat ❤ Microservices

MORE INFORMATION AT
NGINX.COM

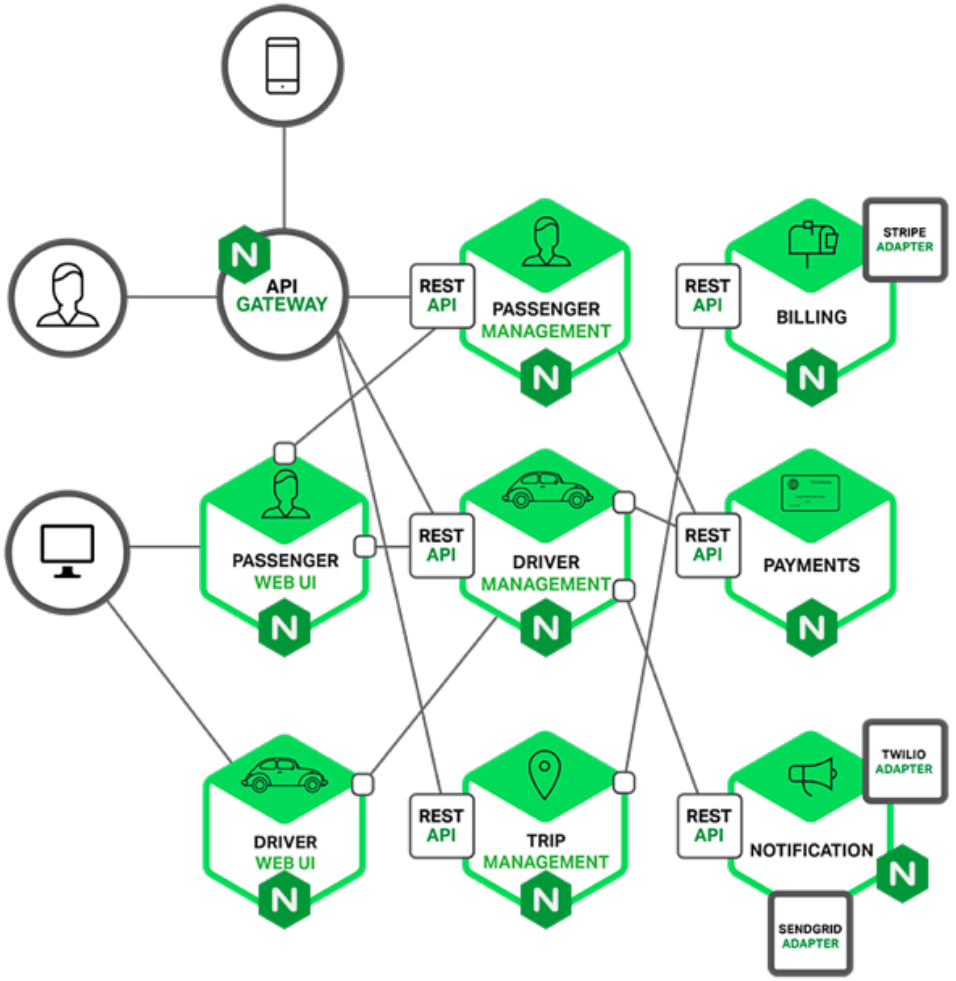
OpenShift 3.3 Delivers on the Vision

The Big Shift

Architectural Changes: Monolith to Microservices



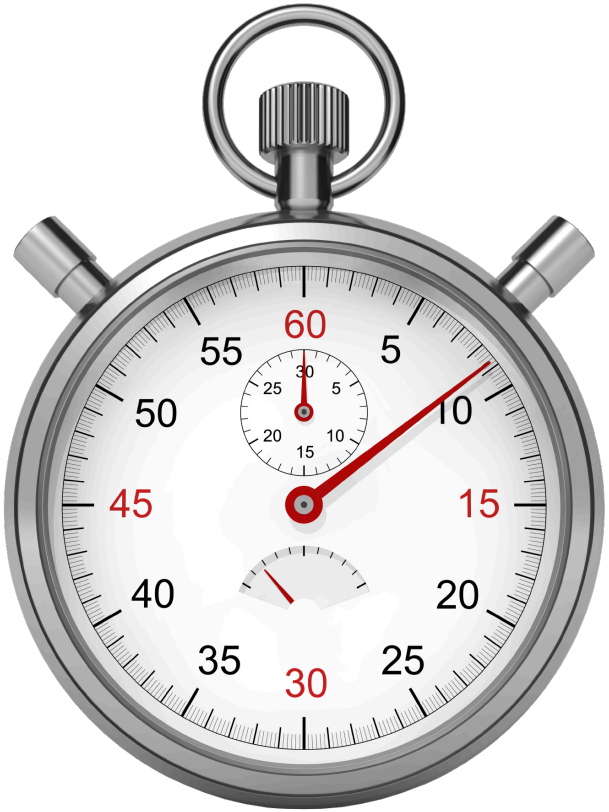
Architectural Changes: Monolith to Microservices



An Anecdote

The tight loop problem

- Rest calls
- 1000’s of requests
- Looped data



Mitigation

- Group requests
- Cache data
- Optimize the network

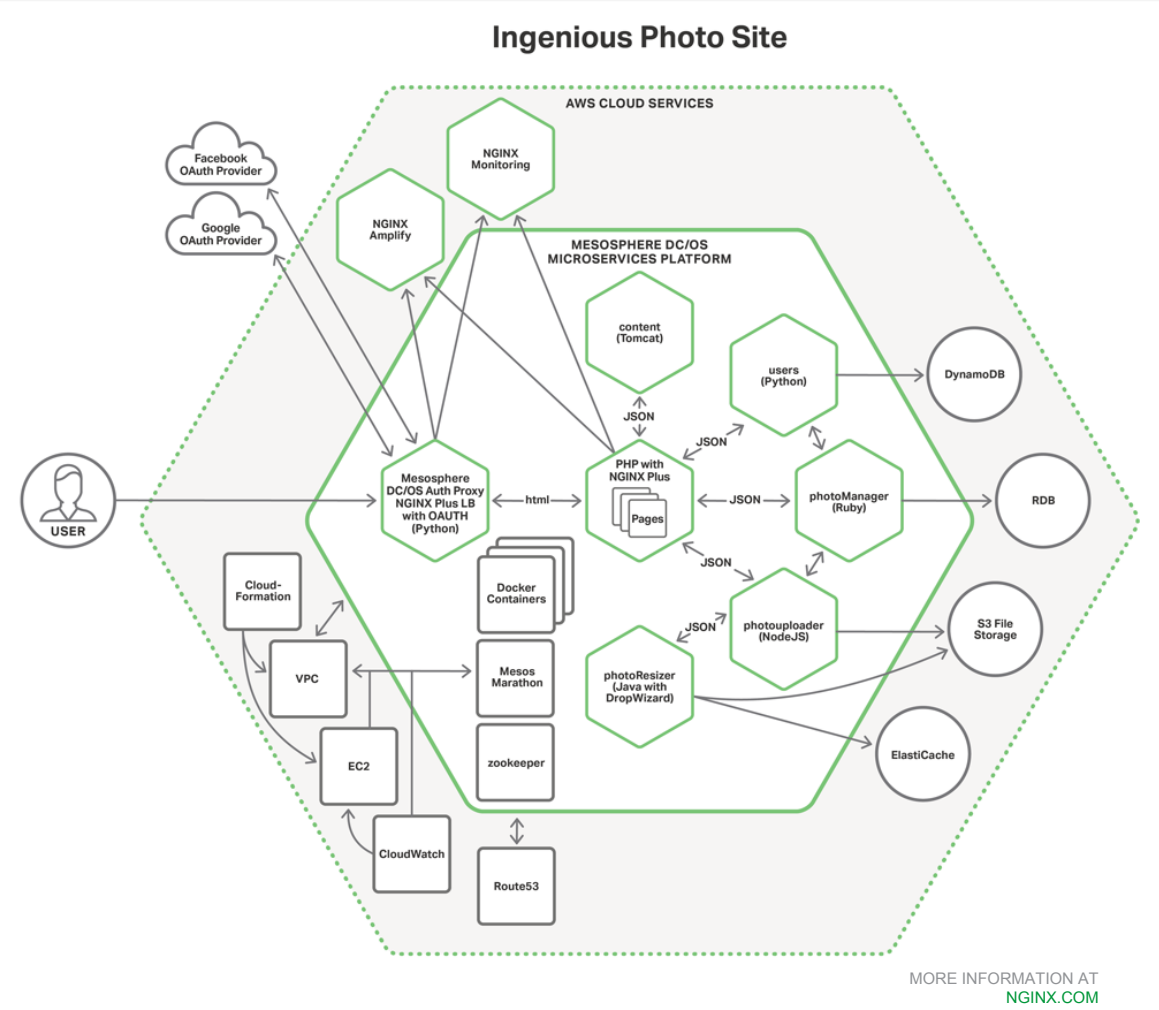

```
{
  "email": "luca.comellini@nginx.com",
  "google_id": "109398466340798000000",
  "id": "c8bbfa44-f1f2-4182-86ff-5c6acdc1faac",
  "name": "luca.comellini@nginx.com"
}
{
  "banner_album_id": "552",
  "baner_url": "https://ngra-images.s3-us-west-1.amazonaws.com/uploads/photos/4ee95887-6fc0-4fd5-a67e-347e749068
  "email": "facebook@cstetson.metadogs.com",
  "facebook_id": "10153778371209200",
  "id": "fe2f6f03-03df-46a4-9e4e-c919225ede8b",
  "name": "facebook@cstetson.metadogs.com"
}
{
  "email": "rick@nginx.com",
  "google_id": "110707406787155000000",
  "id": "94df6c24-71a6-4ead-a9f7-06c42b977abd",
  "name": "rick@nginx.com"
}
{
  "email": "shannon@nginx.com",
  "google_id": "103187003852211000000",
  "id": "5d8141d6-9211-4e76-9f63-a3ec26a18837",
  "name": "shannon@nginx.com"
}
```

NGINX ♥ Microservices

MORE INFORMATION AT
NGINX.COM

Microservices Reference Architecture

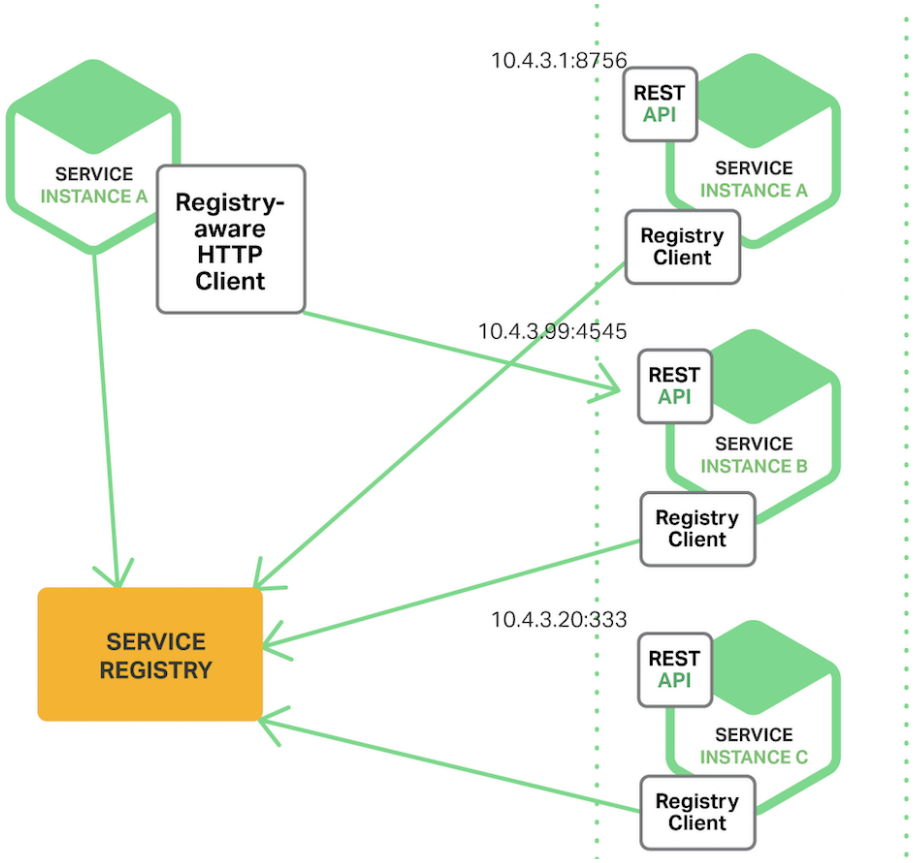
- Docker containers
- Polyglot services
- 12-Factor App(-esque) design



The Networking Problem

Service Discovery

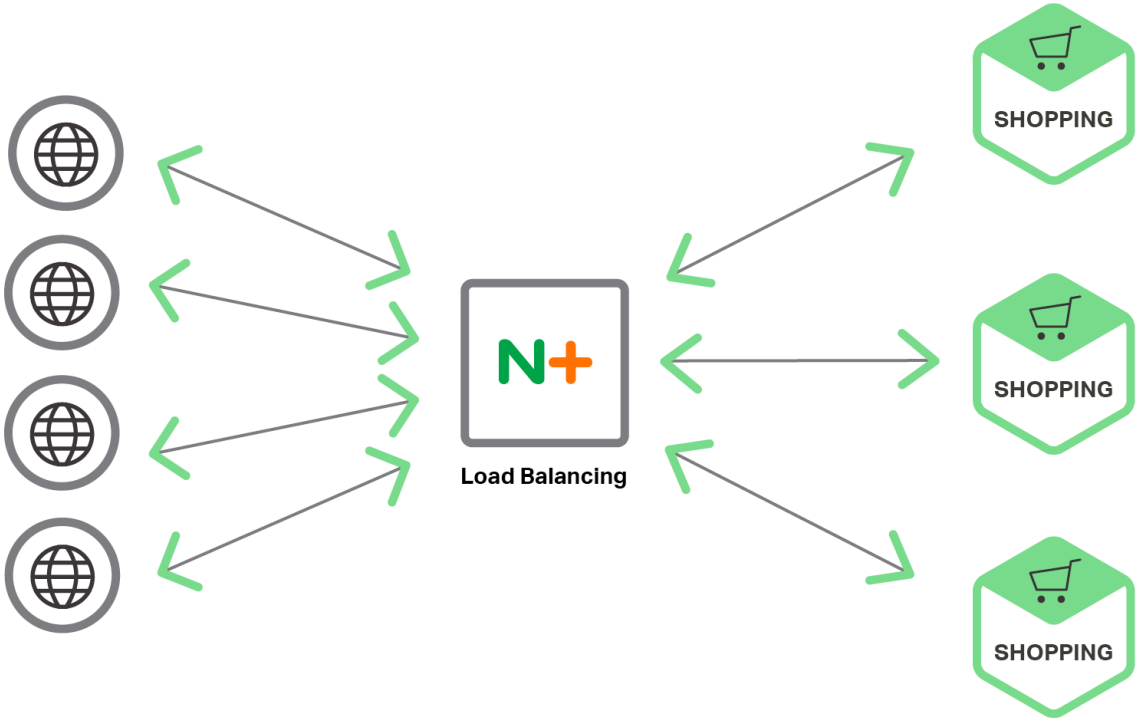
- Services needs to know where other services are
- Service registries work in many different ways
- Register and read service information



MORE INFORMATION AT
NGINX.COM

Load-balancing

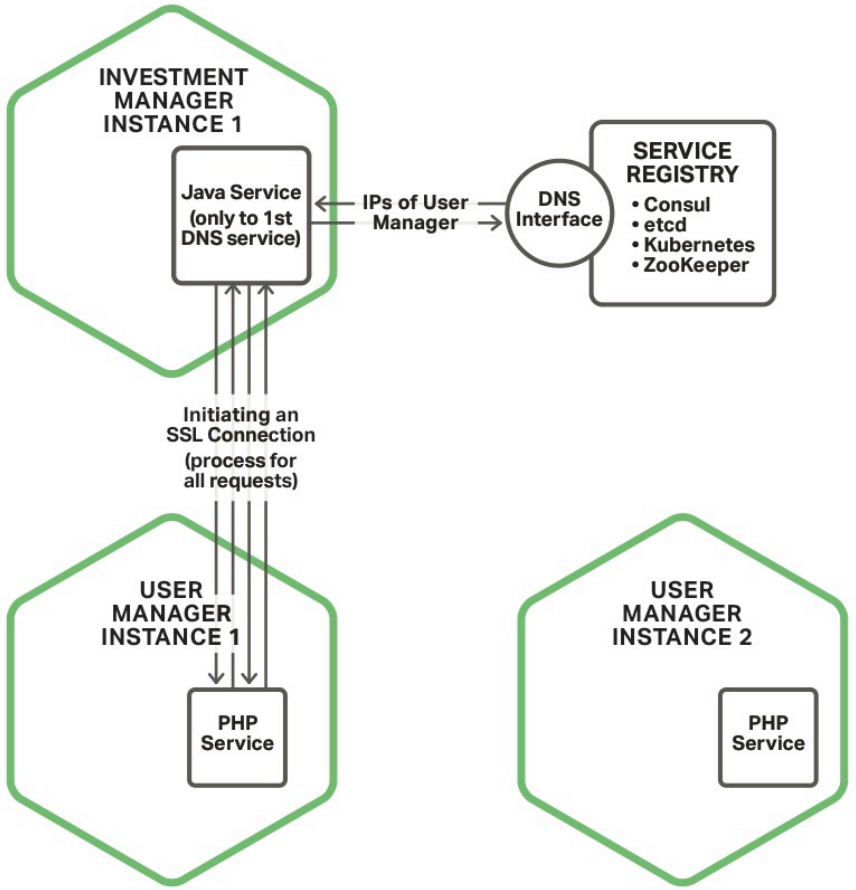
- High Quality Load Balancing
- Developer Configurable



MORE INFORMATION AT
NGINX.COM

Secure & Fast Communication

- Encryption at the transmission layer is becoming standard
- SSL communication is slow
- Encryption is CPU intensive



MORE INFORMATION AT
NGINX.COM

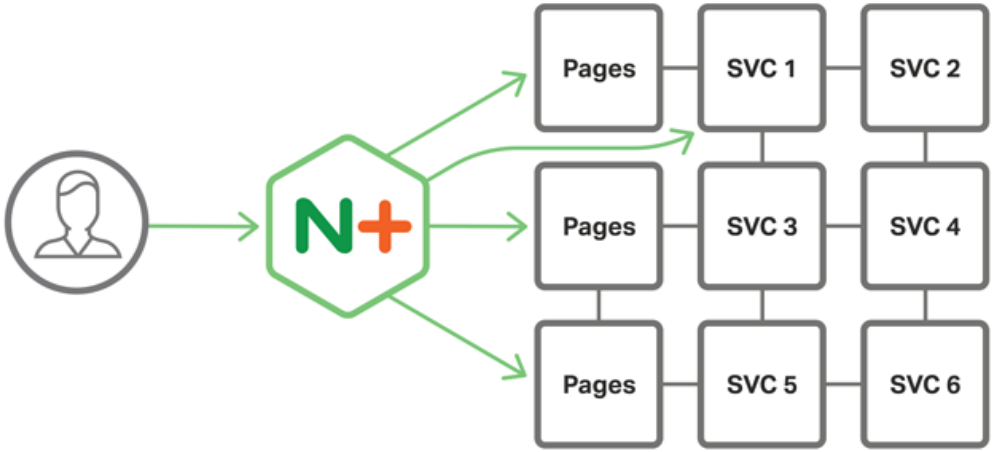
Solution

- Service discovery
- Robust load balancing
- Fast encryption

Network Architectures

Proxy Model

- In bound traffic is managed through a reverse proxy/load balancer
- Services are left to themselves to connect to each other.
- Often through round-robin DNS



Proxy Model

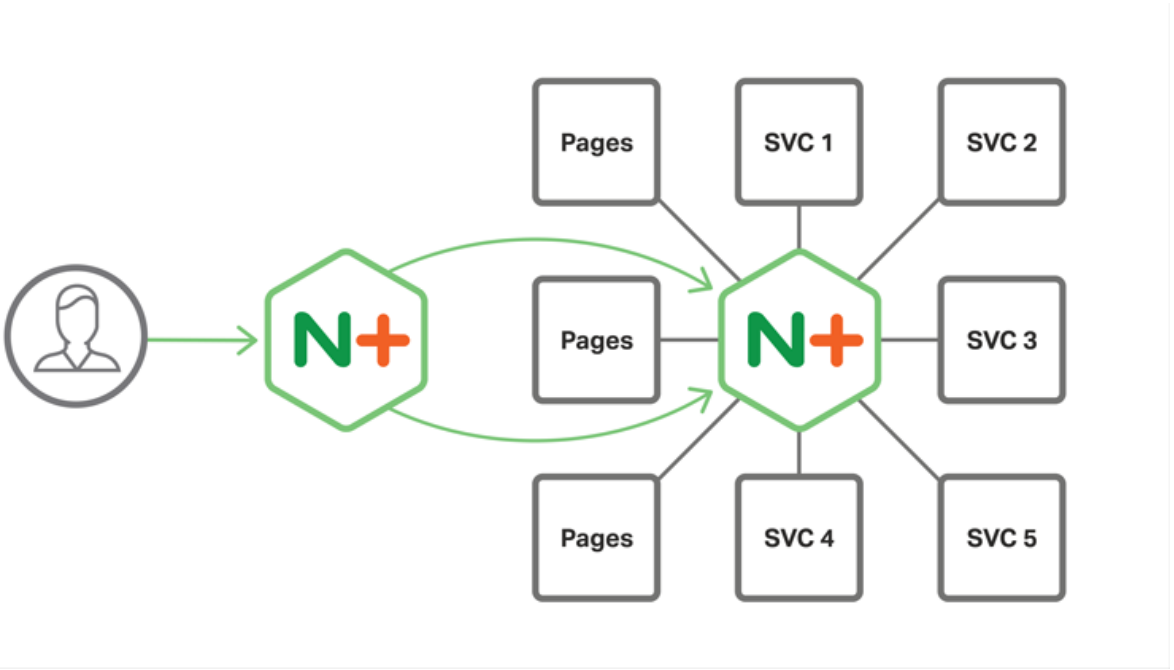
- Focus on internet traffic
- A shock absorber for your app
- Dynamic connectivity

OpenShift Implementation

- Primary host route
- Pass Through
- Ingress Controller

Router Mesh Model

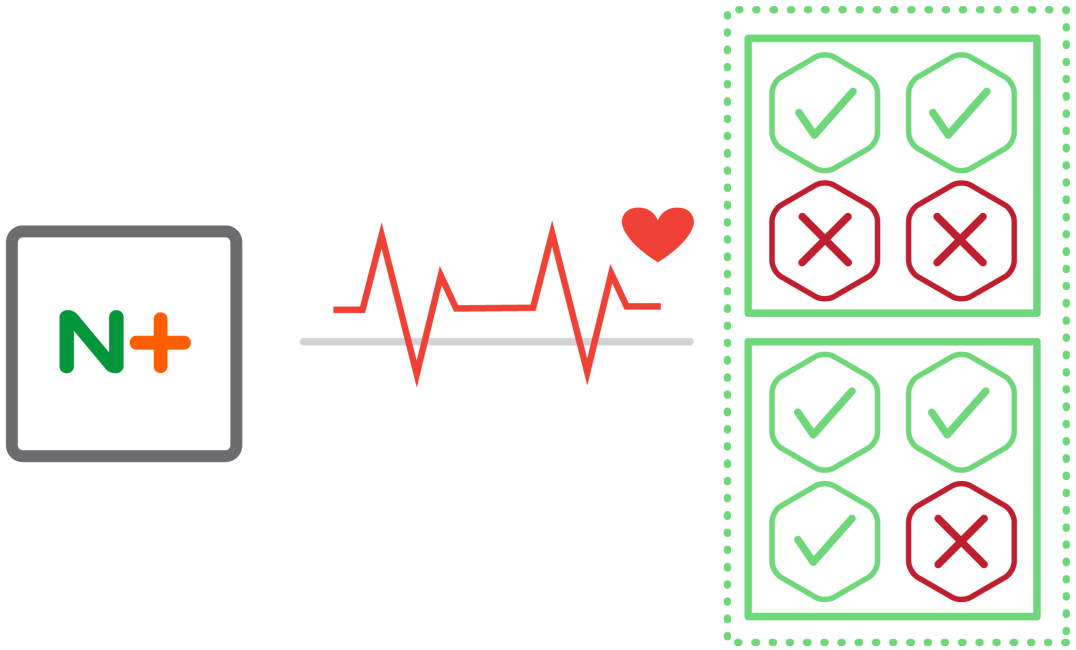
- In-bound routing through reverse proxy
- Centralized load balancing through a separate load balancing service
- Deis Router work like this.



MORE INFORMATION AT
NGINX.COM

Circuit Breakers

- Active health checks
- Retry
- Caching



MORE INFORMATION AT
NGINX.COM

Router Mesh

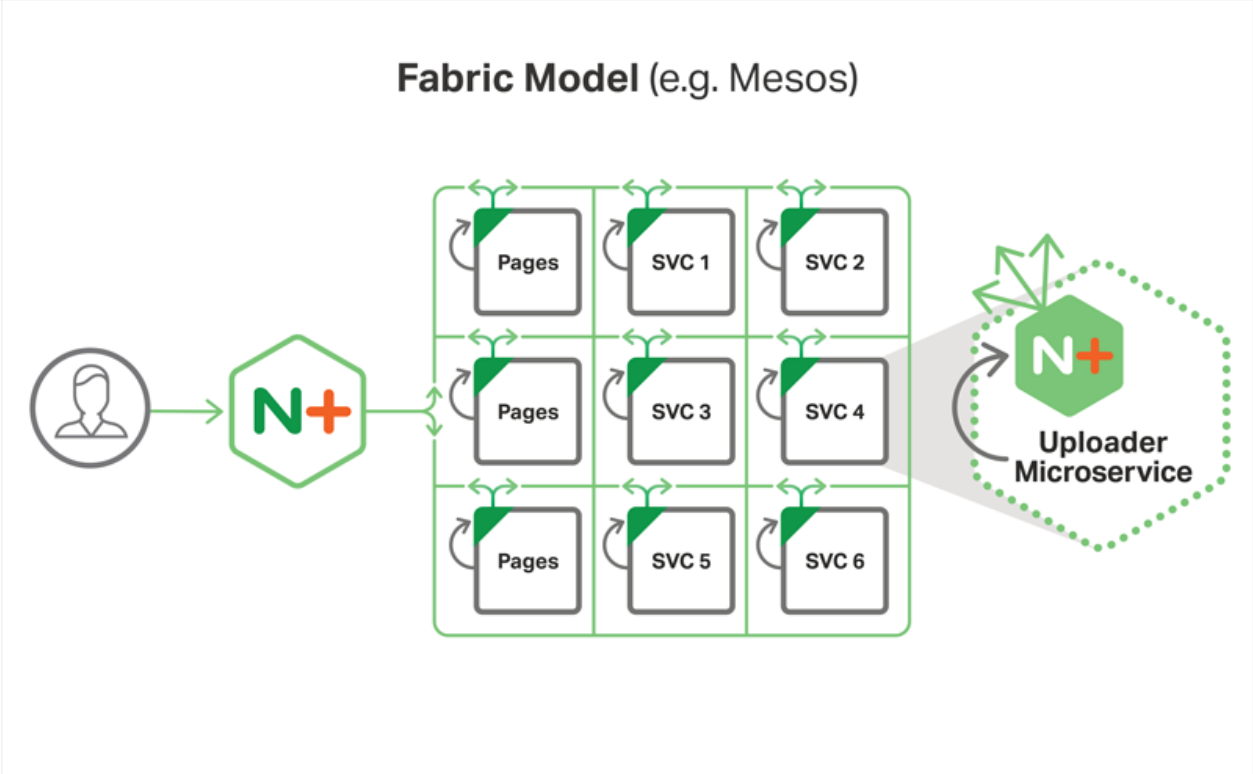
- Robust service discovery
- Advanced load balancing
- Circuit breaker pattern

OpenShift Implementation

- Kubernetes event listener
- LB_Service env vars
- Each service implemented as a Kubernetes service
- Privileged user

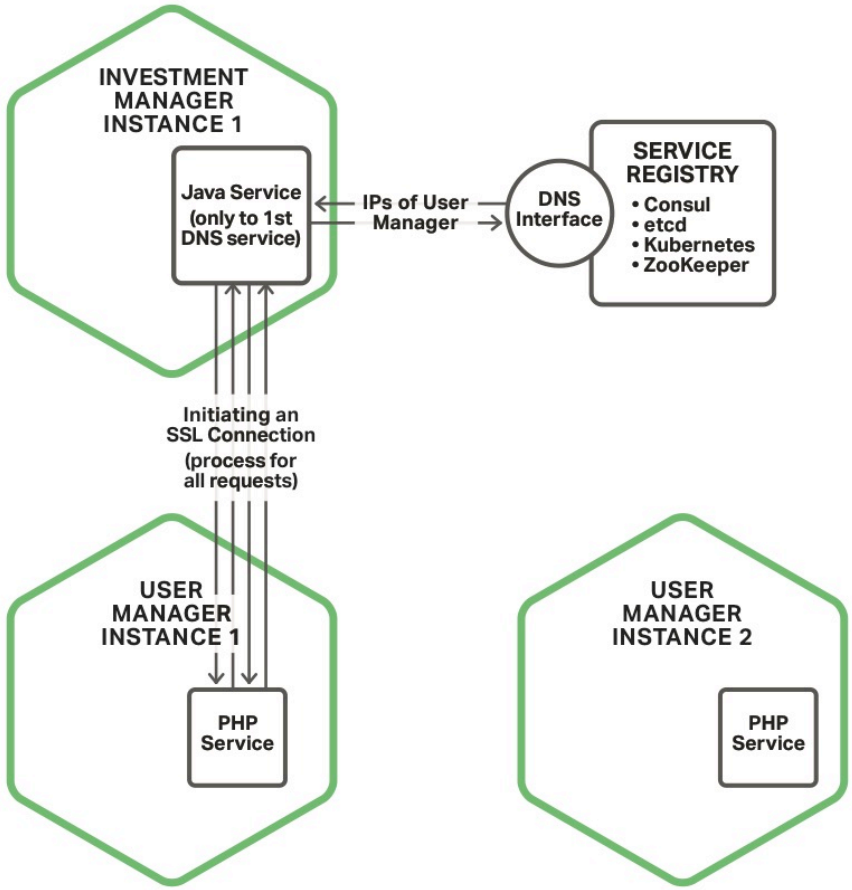
Inter-Process Communication

- Routing is done at the container level
- Services connect to each other as needed
- NGINX Plus acts as the forward and reverse proxy for all requests



Normal Process

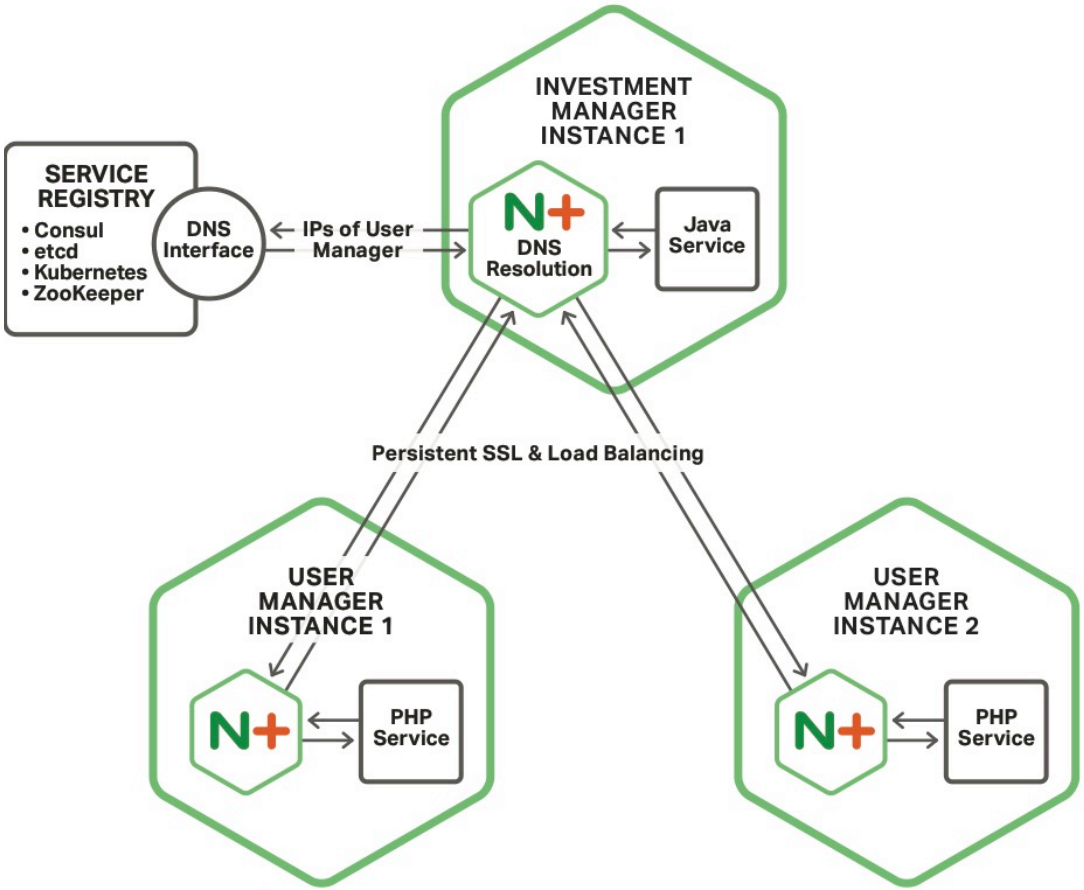
- DNS service discovery
- Relies on round robin DNS
- Each request creates a new SSL connection which fully implemented is 9 requests



MORE INFORMATION AT
NGINX.COM

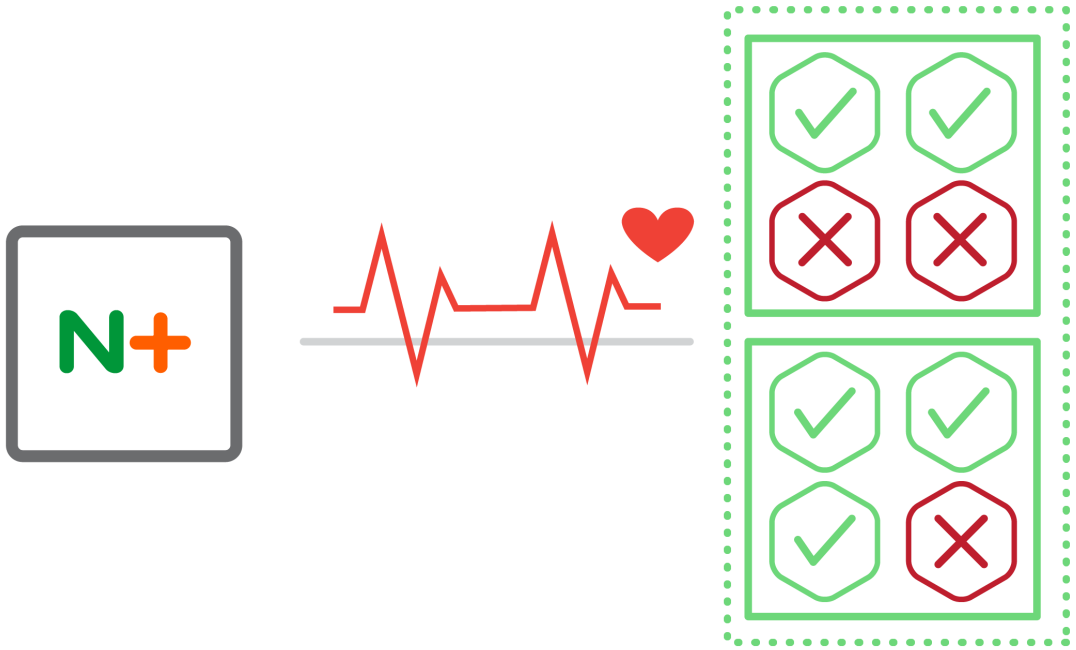
Detail

- NGINX Plus runs in each container
- Application code talks to NGINX locally
- NGINX talks to NGINX
- NGINX queries the service registry



Circuit Breaker Plus

- Active health checks
- Retry
- Caching



MORE INFORMATION AT
NGINX.COM

Fabric Model

- Robust service discovery
- Advanced load balancing
- Circuit breaker pattern
- High-performance SSL

OpenShift Implementation

- Each app is a Kubernetes service
- Name the ports (e.g. https)

Issues

Docker Recommendation: 1 service per container

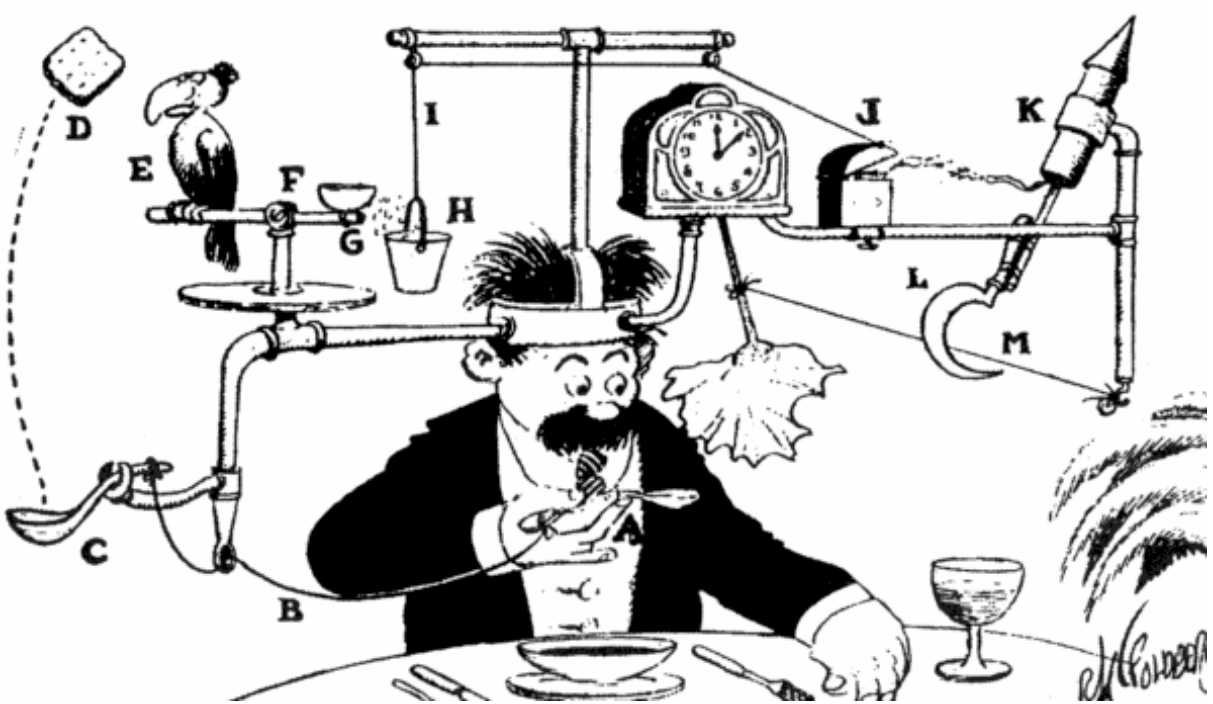
- Keeps docker images simple
- Process failure means container failure
- Only a recommendation

1*

MORE INFORMATION AT
NGINX.COM

Complexity

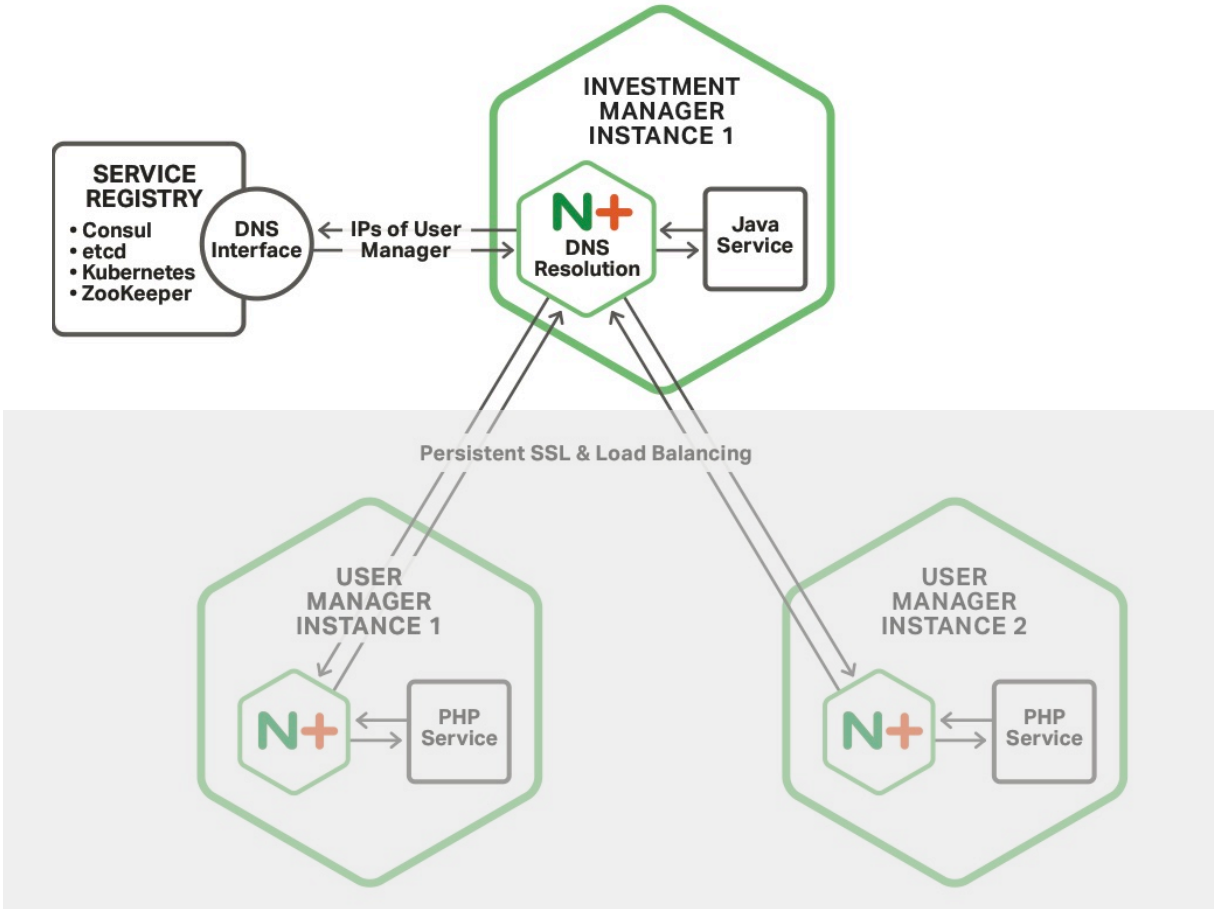
- Adding another layer to the stack
- Lots of power to give to dev team
- Tooling to make the Fabric Model simple to create and deploy



Conclusion

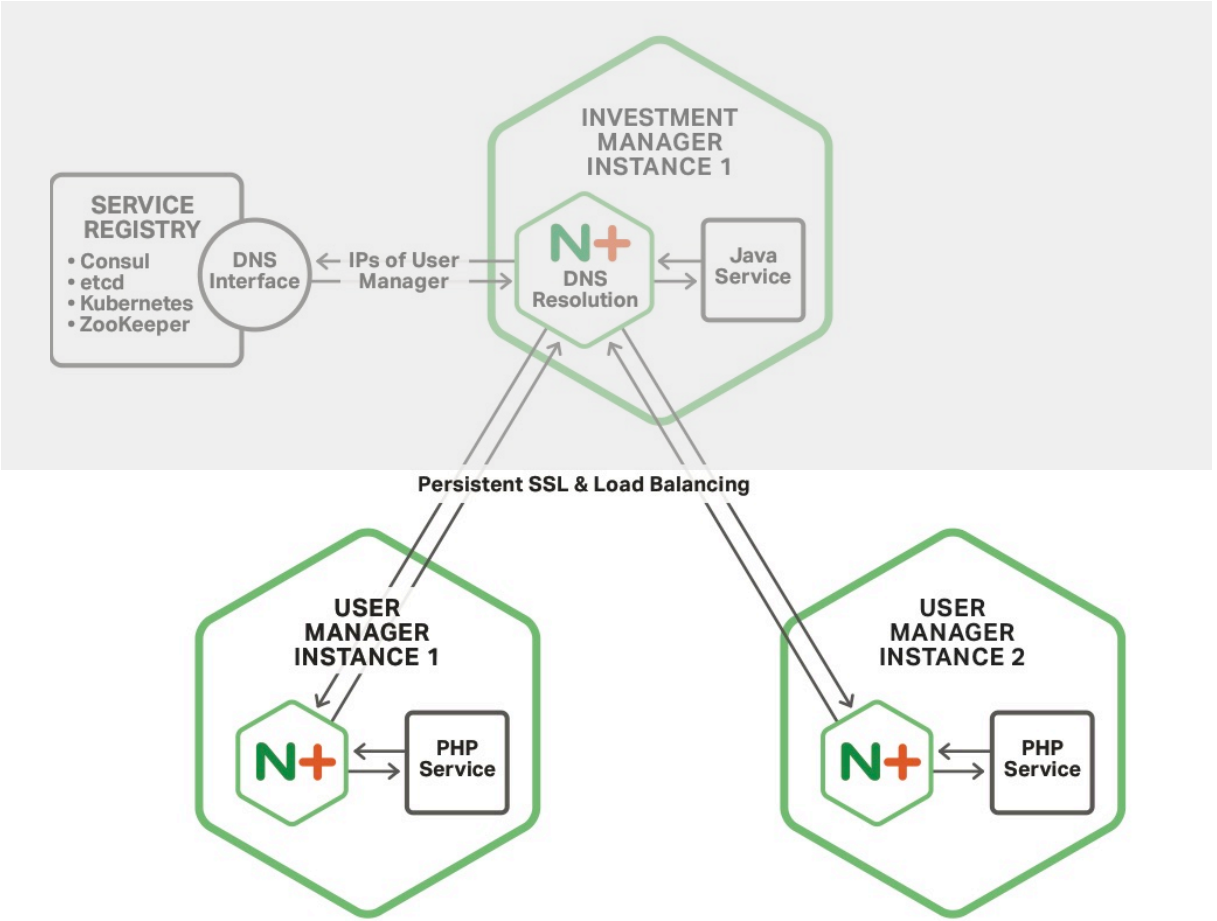
Service Discovery

- DNS is a clear way to manage service discovery
- NGINX Plus Asynchronous Resolver
- SRV records allow you to effectively use your resources



Load-balancing

- Proper request distribution
- Flexibility based on the backing service
- Different load-balancing schemes



Persistent SSL Connections

- Applications generate thousands of connections
- 9 steps in SSL negotiation
- Persistent SSL upstream keepalive