



HACKING VS ENGINEERING

**NGINX UPSTREAMS & CONSUL**



# HACKING VS ENGINEERING

The background of the slide is a dark, atmospheric photograph of an offshore oil rig engulfed in a massive fire. A large, billowing plume of black smoke rises from the center of the inferno. Several fireboats are positioned around the burning rig, directing powerful jets of water onto the flames from multiple angles. The scene is set at night or in low light, emphasizing the intensity of the fire.

- **HACKING**

- The act of producing an inelegant solution to a problem

- **ENGINEERING**

- Skillful or artful contrivance; maneuvering

# HACKING VS ENGINEERING

The background image shows a large offshore oil rig engulfed in a massive fire, with thick black smoke billowing into the sky. Several fireboats are positioned around the burning rig, directing high-pressure water jets onto the flames from multiple angles. The scene is set on a body of water under a hazy sky.

- **Systems are products of the constraints enforced during the design/production process**
- **HACKING**
  - Time
  - Functional (does it work)
- **ENGINEERING**
  - Time
  - Functional
  - Lifecycle
  - Maintenance
  - Operational Characteristics



# HACKING VS ENGINEERING

The background of the slide is a photograph of an offshore oil rig engulfed in a massive fire. Thick black smoke billows from the top of the flames. Several fireboats are positioned around the burning rig, directing high-pressure water jets onto the base of the fire. The scene is set on the ocean under a hazy sky.

- **WHICH** of these sets of constraints can guarantee success for the future (sustainability)?
- **ENGINEERING**
- **WHY** must we constantly bring this up?
  - The **JUST MAKE IT WORK** mentality is pervasive in our industry
  - Continuously presented with examples of work where there is no evidence that **ANY** prior art was considered

# HACKING VS ENGINEERING

The background image shows a large offshore oil rig engulfed in a massive fire, with thick black smoke billowing into the sky. Several fireboats are positioned around the burning rig, directing high-pressure water jets onto the flames. The scene is set in a body of water under a hazy sky.

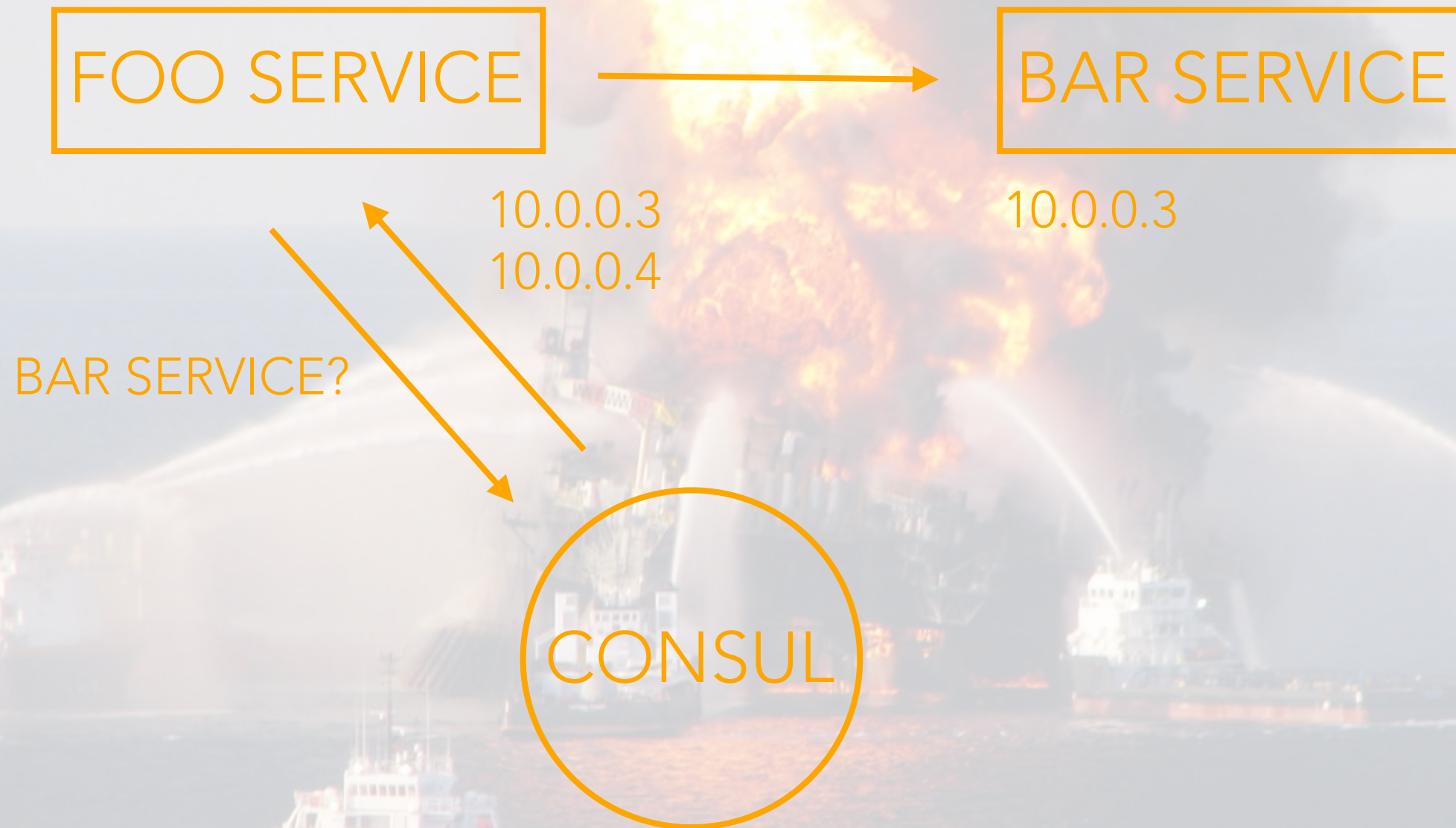
- The **PROBLEM** NGINX upstreams via Consul
- **HACKING** pre-existing solutions
- **ENGINEERING** a solution utilizing ngx-lua



# THE PROBLEM :: ARCHITECTURE

- **PULSE** is a service architecture
- Some services depend on other services
  - Relatively straight forward to implement service connectivity within each service
  - Consul service discovery via http
  - Load balancing http client seeded with IPs discovered via consul

# THE PROBLEM :: ARCHITECTURE





# THE PROBLEM :: ARCHITECTURE

- We want to present a unified external surface to the user, in this case <https://api.truesight.bmc.com>
- URIs terminated at the edge loadbalancer (NGINX)
- Multiple instances of each service
- Each service owns a namespace (uri path)
  - Example /measurements -> sasquatch
- Should be able to dynamically add service nodes without running configuration management



# HACKING :: EXISTING SOLUTIONS

- Rebuild existing configuration every so often with config management
  - Lot's of issues, though mostly with local development
- Rebuild upstreams with a script via Consul watch and reload NGINX configuration
  - As it turns out consul watches on service membership are a bit noisy
  - Requires configuration management to manage to Consul watches and scripts
  - Scripted reloads of a service typically not the best idea
- Consul-Template
  - Watch for changes in consul, update an upstream template, reload nginx



# HACKING :: CONSUL-TEMPLATE

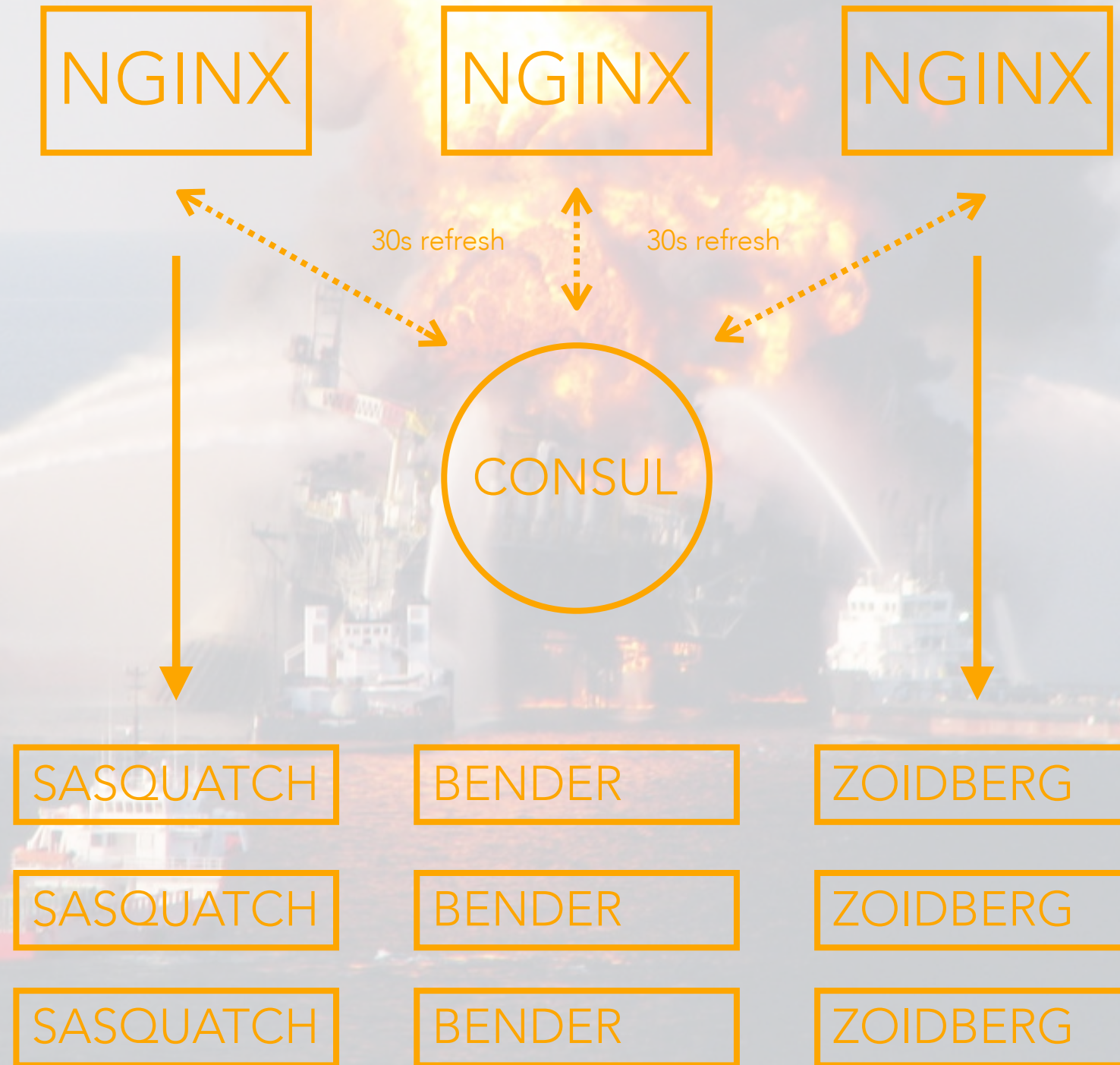
- The best of all the **HACKING** solutions, ... or is it
  - Need to run **ANOTHER** service on each loadbalancer
  - I need more configuration management to manage consul-template, the templates, the consul agent, and the NGINX reload
  - I am interacting with NGINX via a file interface and a possibly brittle reload
  - Don't get me wrong: consul-template is great for certain use cases



# ENGINEERING :: DESIGN GOAL

/v1/measurements  
(POST)

<https://api.truesight.bmc.com>





# ENGINEERING :: RESEARCH

- What do we know about **NGINX** now?
  - Has a module system, can be extended in **C**
  - Some portion of the C API is surfaced in **lua**
- What do we do first? **RESEARCH PRIOR ART**
  - Found some examples of similar work in **C**
    - By doing this research we gained a lot of insight into NGINX internals
  - We can do this in **C**, what can we do in **lua**?
    - Doing this research we learned about
      - OpenResty
      - lua-nginx-module
        - lua-nginx-module api and what can be accomplished on top of NGINX internals



# ENGINEERING :: IMPLEMENTATION

- Code Configuration Walkthrough

