Load Balancing Microservices at ELEME

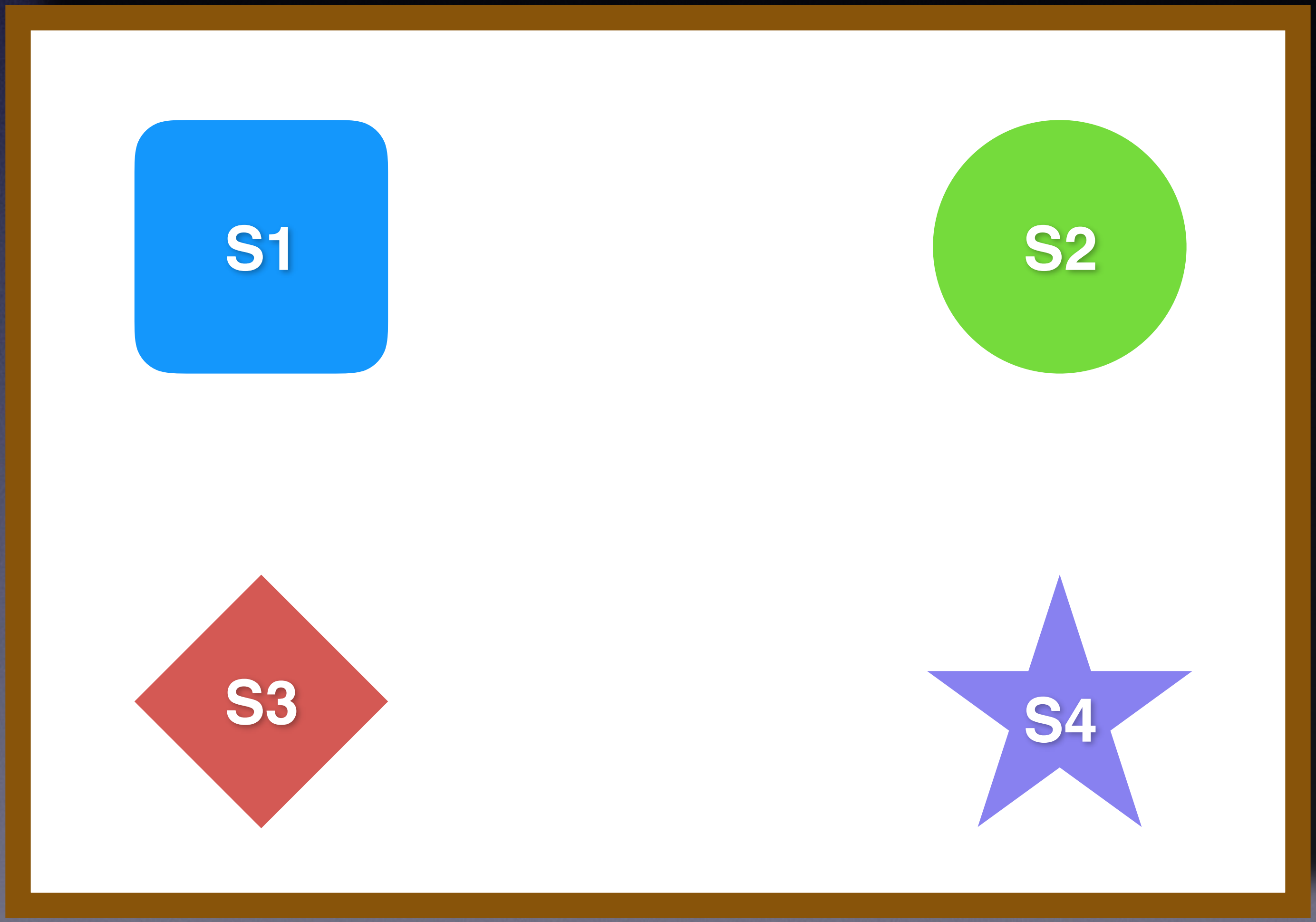
HAProxy to Samaritan

Tevin Zhang

Agenda

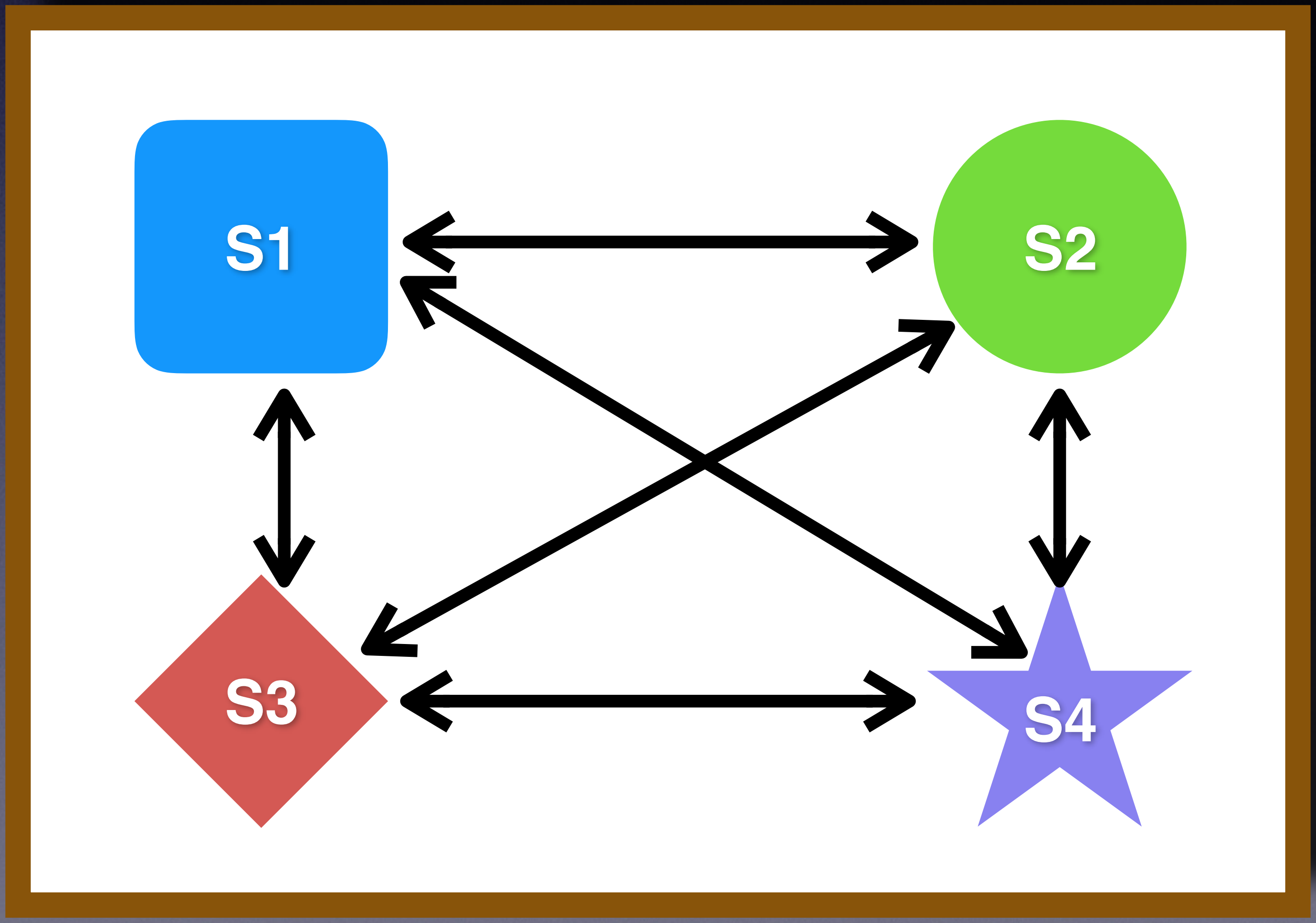
- **The problem**
- **Solutions**
- **How HAProxy is used**
- **Samaritan overview**

The problem



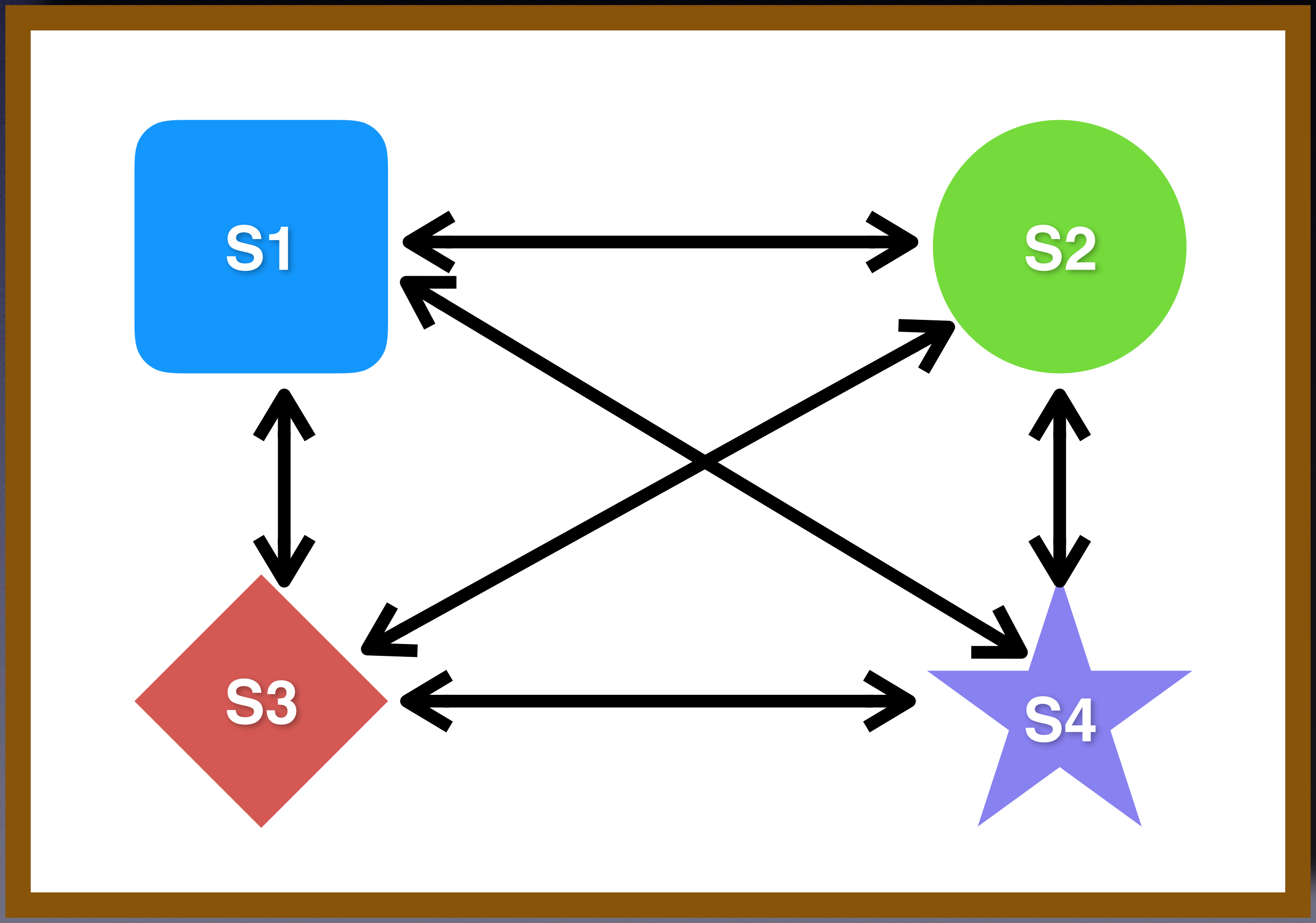
The problem

Service registration & discovery



The problem

Service registration & discovery



Solutions

Solutions

- **Static config (or worse: hard coding)**

Solutions

- **Static config (or worse: hard coding)**
- **Smart DNS**

Solutions

- **Static config (or worse: hard coding)**
- **Smart DNS**
- **Central load balancing**

Solutions

- **Static config (or worse: hard coding)**
- **Smart DNS**
- **Central load balancing**
- **In-app registration & discovery**

Solutions

- **Static config (or worse: hard coding)**
- **Smart DNS**
- **Central load balancing**
- **In-app registration & discovery**
- **Client-side load balancer**

Static config / hard coding

Static config / hard coding

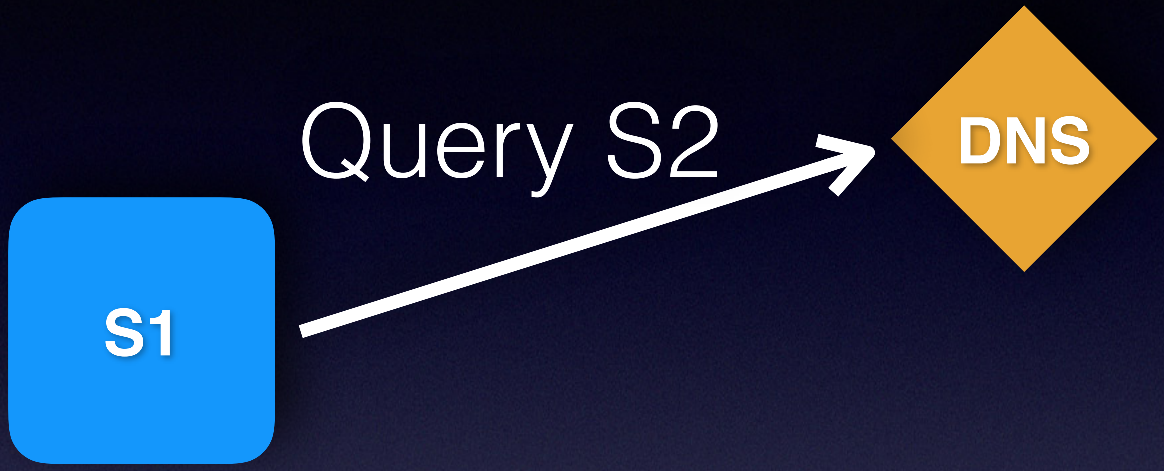
- Hard/Unable to update

Static config / hard coding

- Hard/Unable to update
- Too many cons

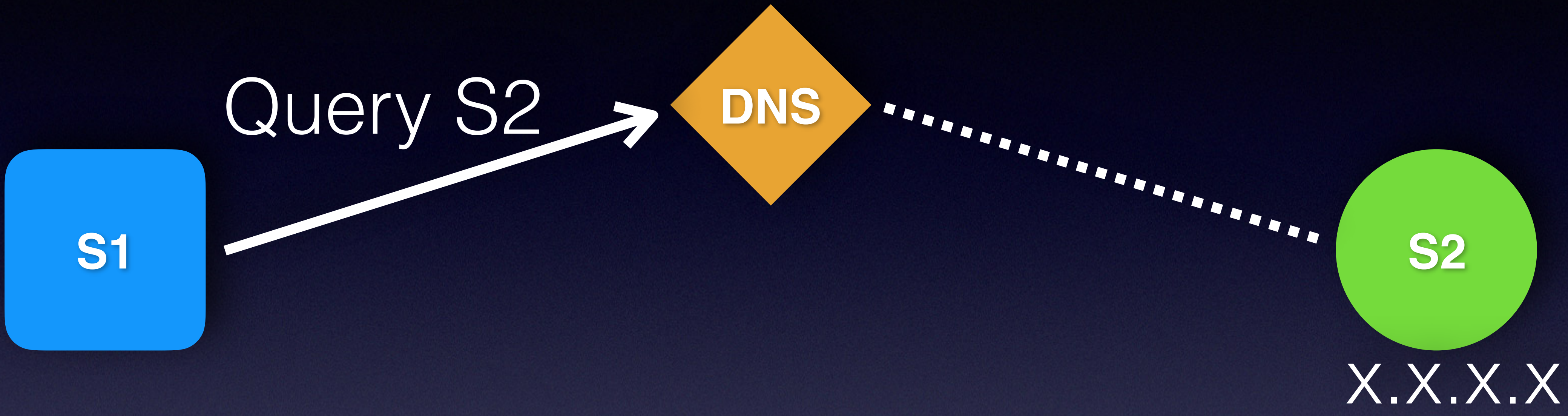
Smart DNS

Smart DNS

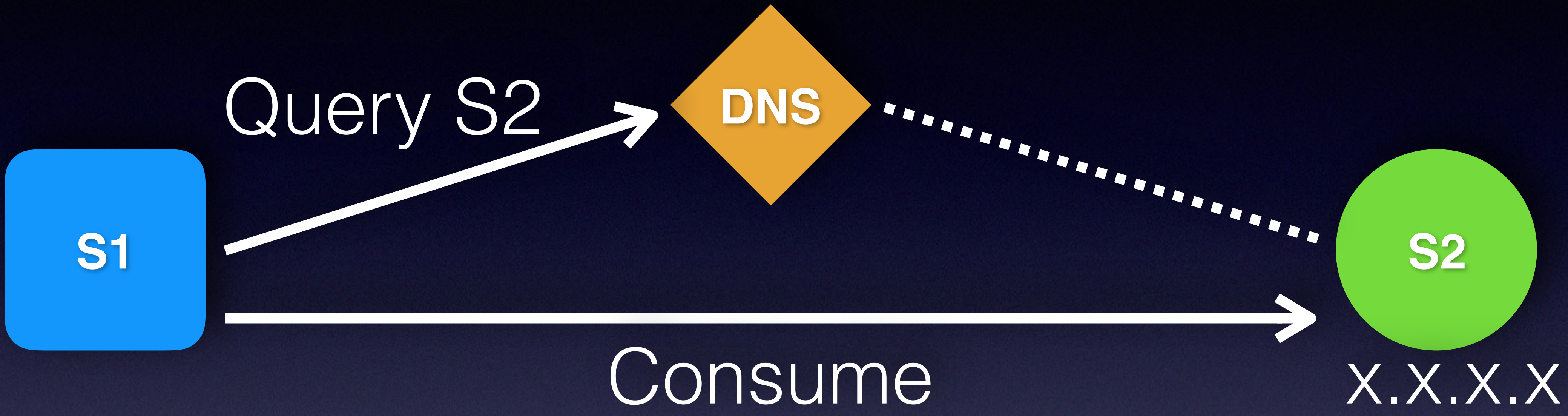


Query S2

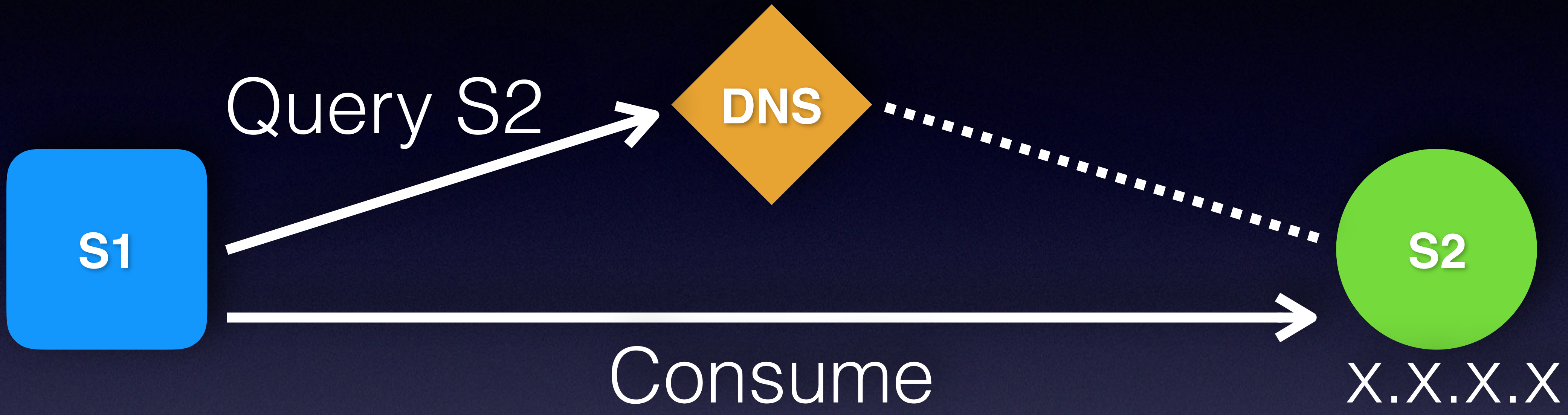
Smart DNS



Smart DNS

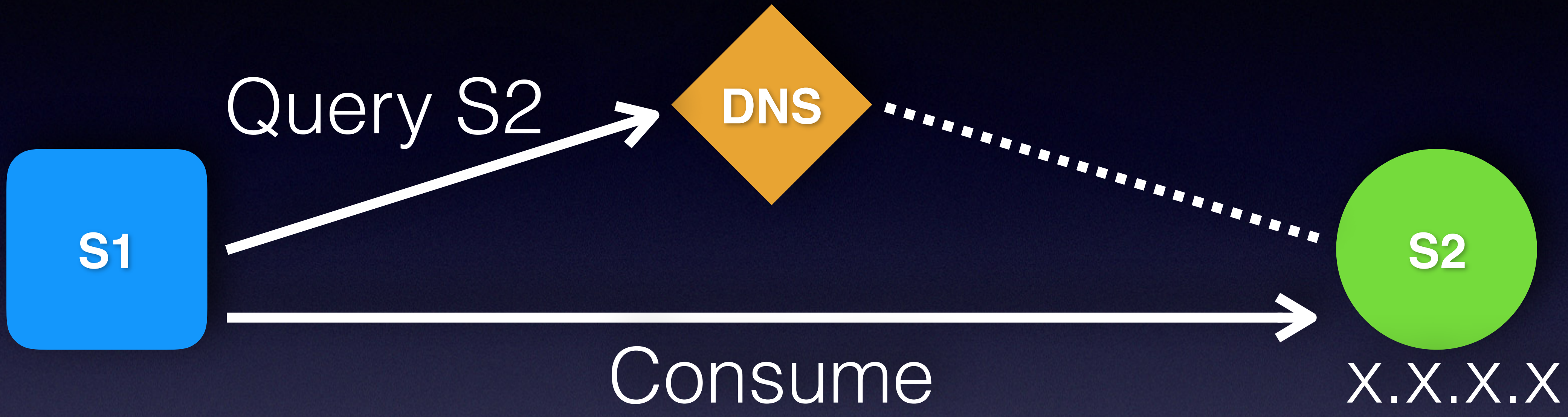


Smart DNS



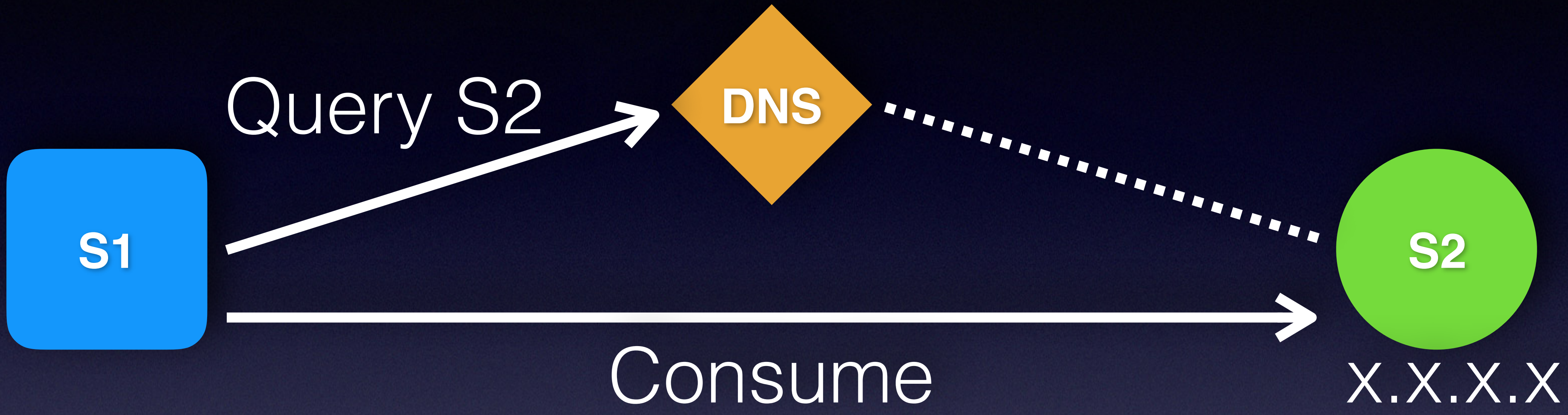
- **Propagation delay**

Smart DNS



- **Propagation delay**
- **Layers of caching**

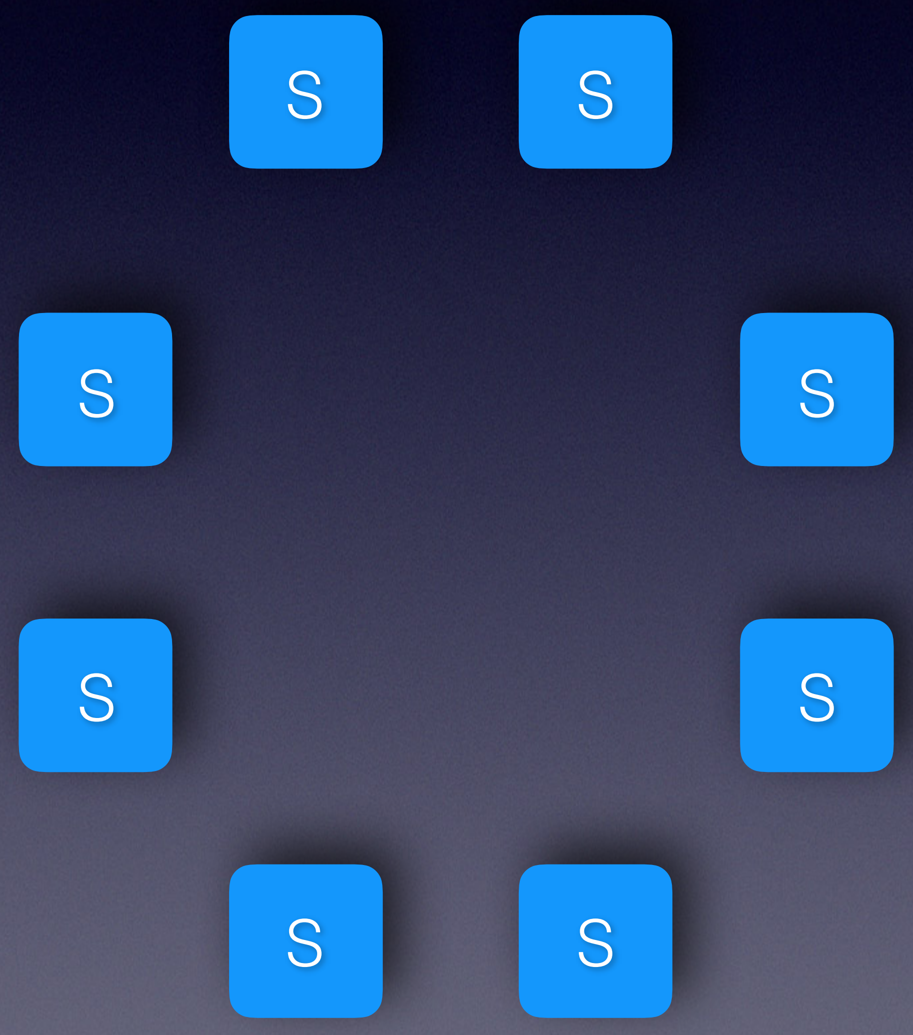
Smart DNS



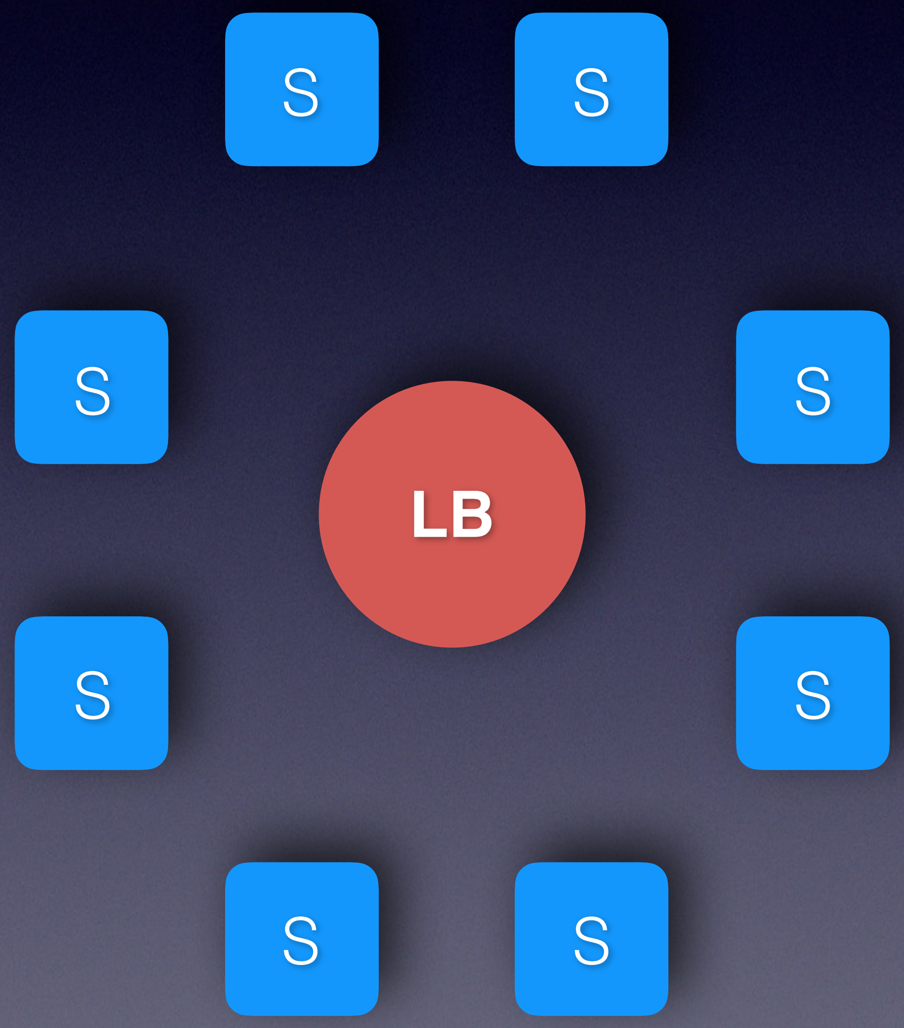
- **Propagation delay**
- **Layers of caching**
- **Unable to push state**

Central load balancing

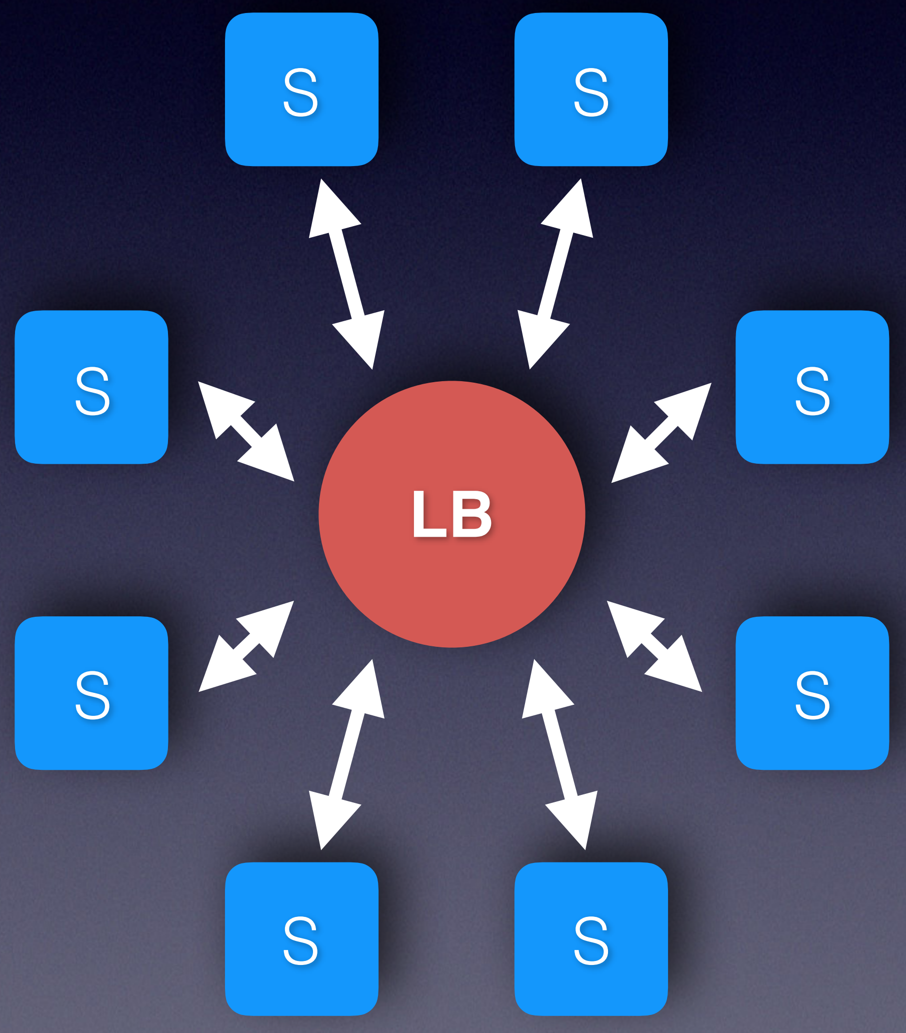
Central load balancing



Central load balancing

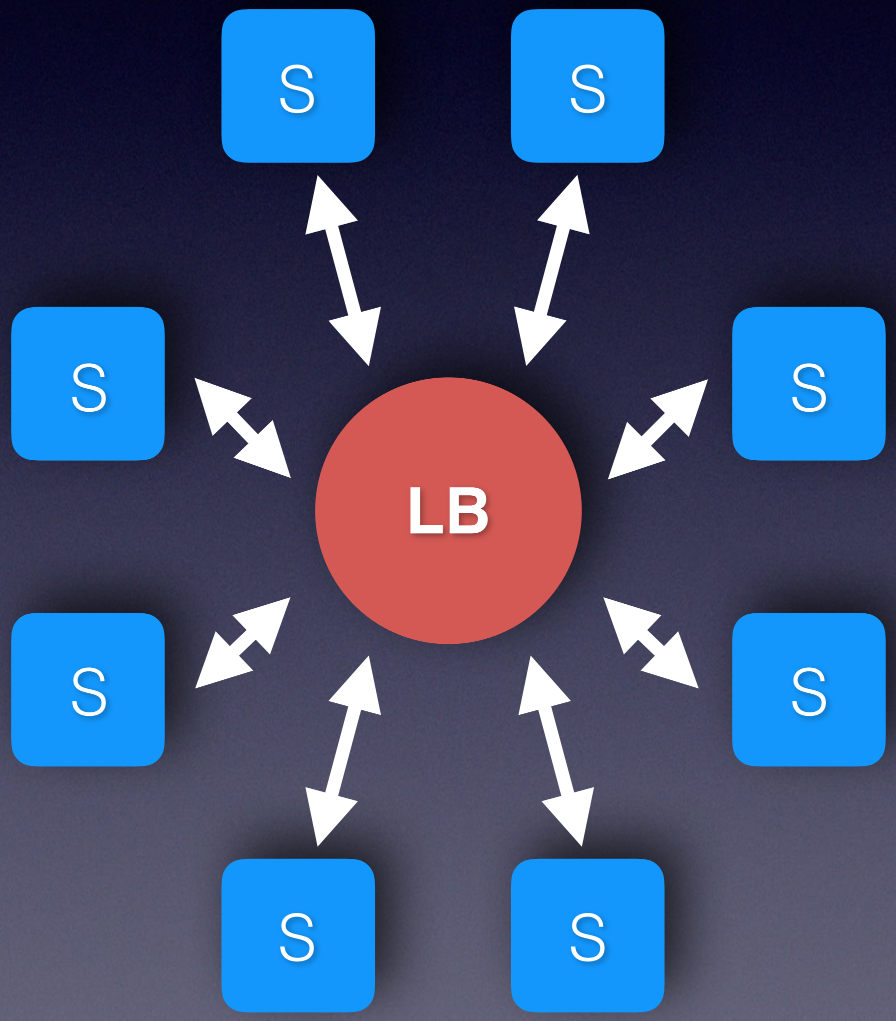


Central load balancing



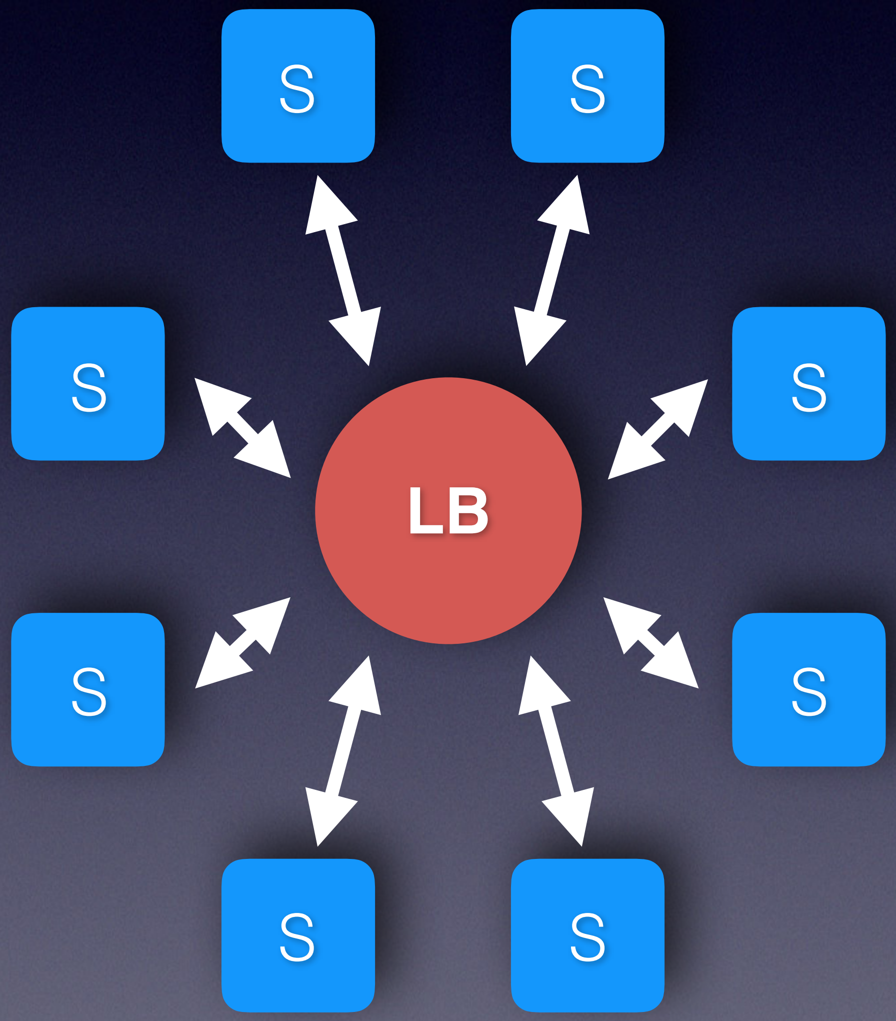
Central load balancing

- **Single point of failure**



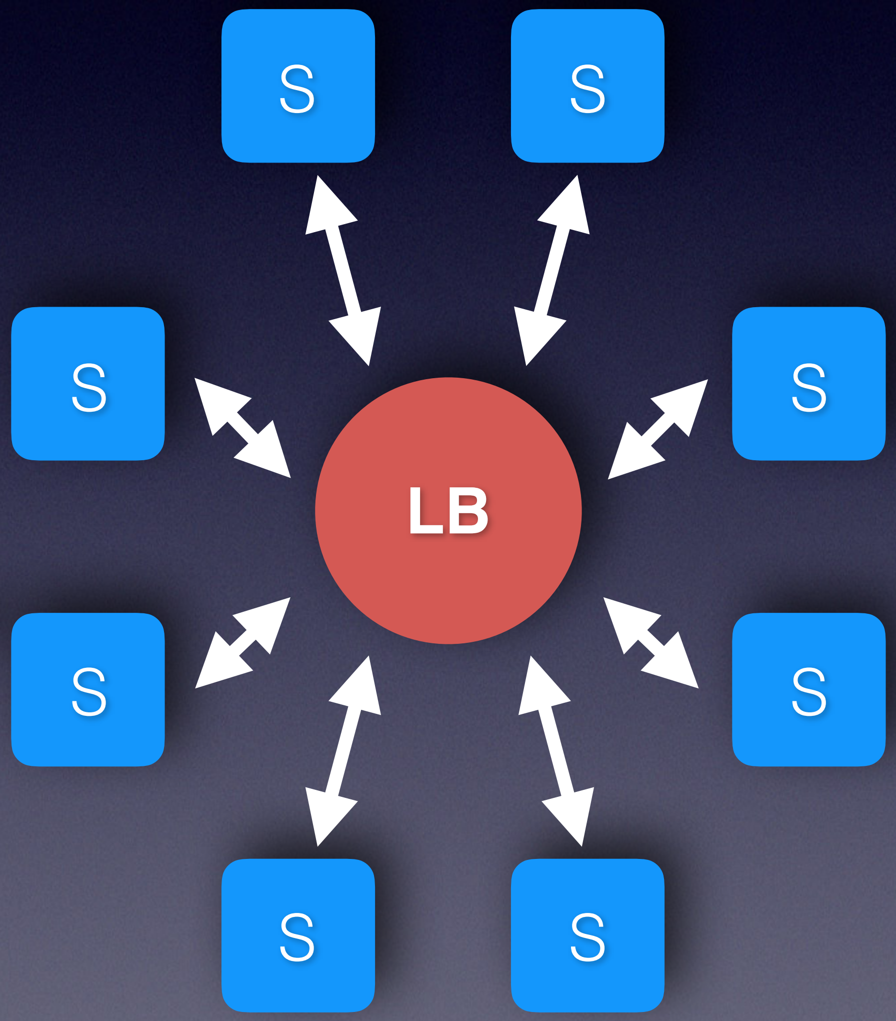
Central load balancing

- Single point of failure
- Performance bottleneck



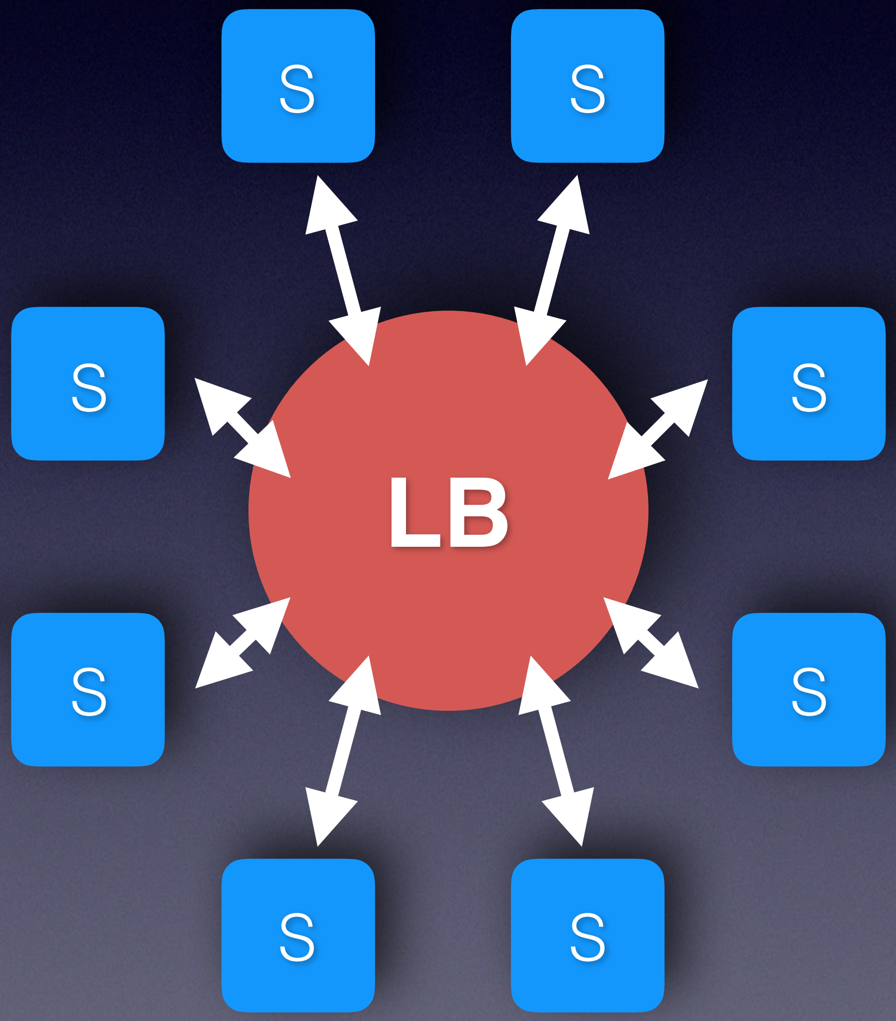
Central load balancing

- **Single point of failure**
- **Performance bottleneck**
- **Hard to deploy & maintain**



Central load balancing

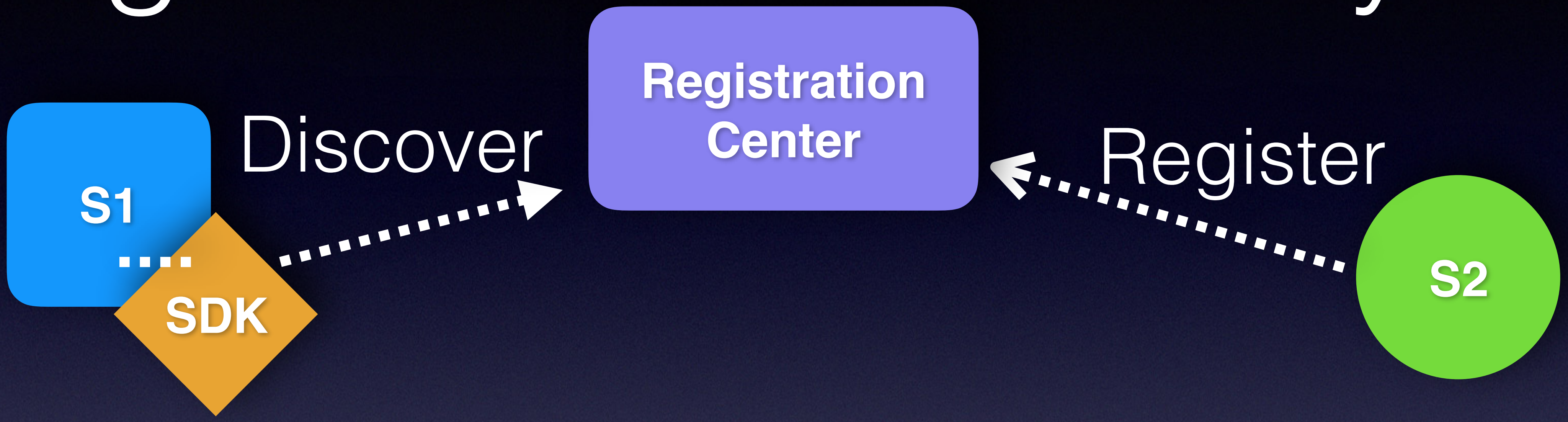
- Single point of failure
- Performance bottleneck
- Hard to deploy & maintain
- Expensive hardware



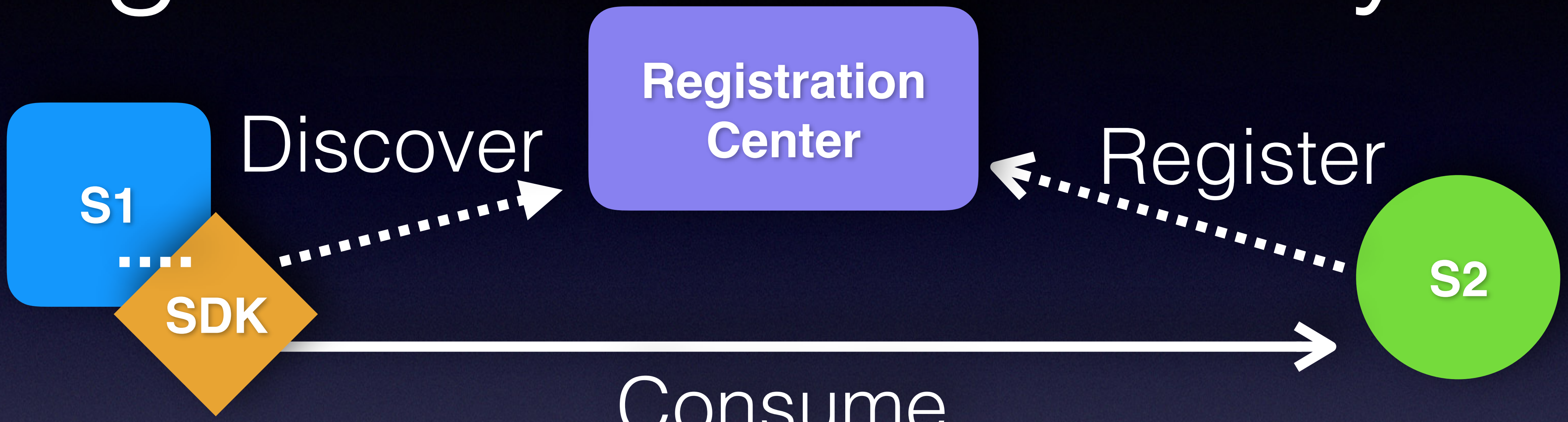
In-app registration & discovery

In-app registration & discovery

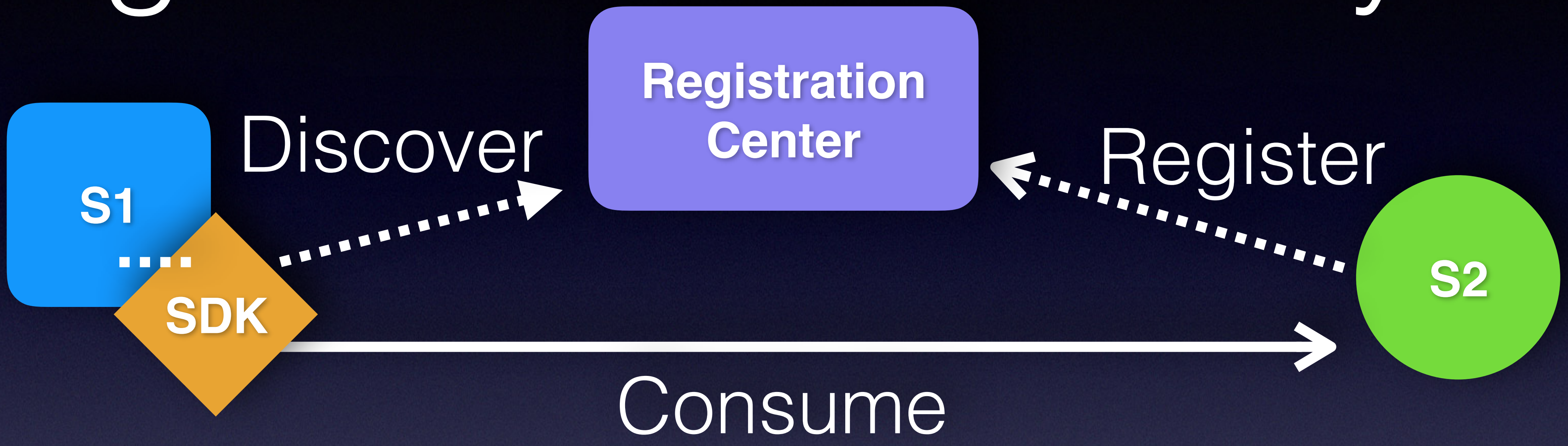
In-app registration & discovery



In-app registration & discovery

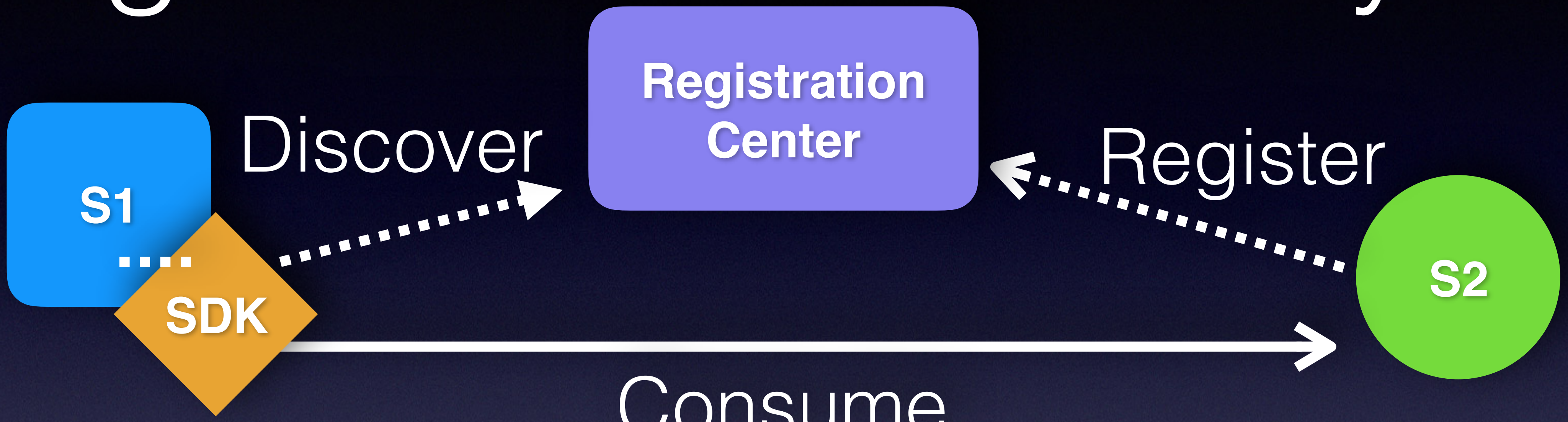


In-app registration & discovery



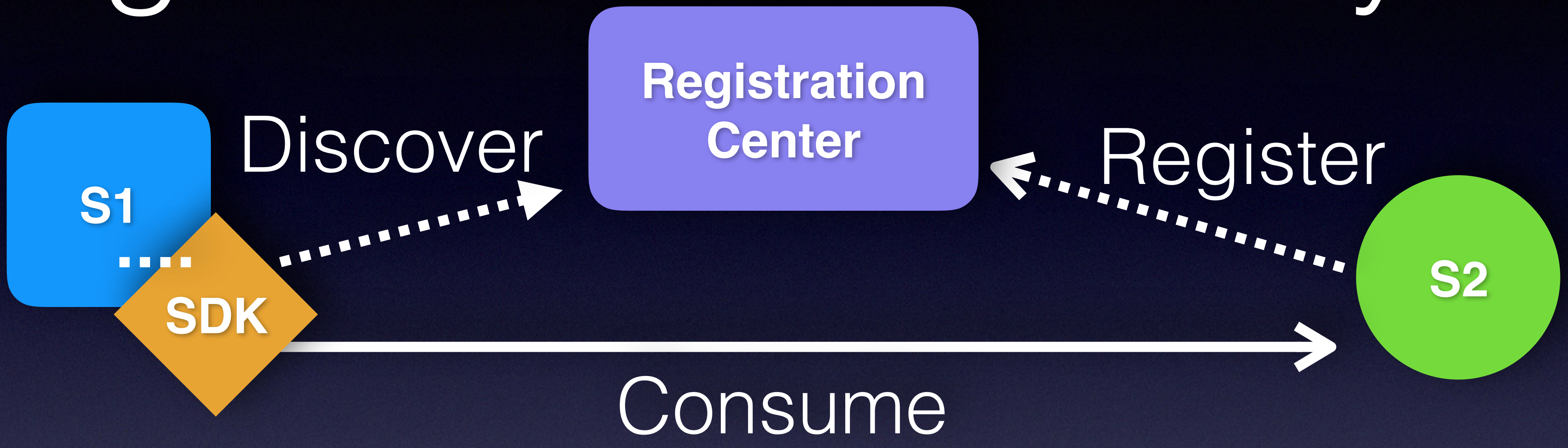
- **Same logic in different languages**

In-app registration & discovery



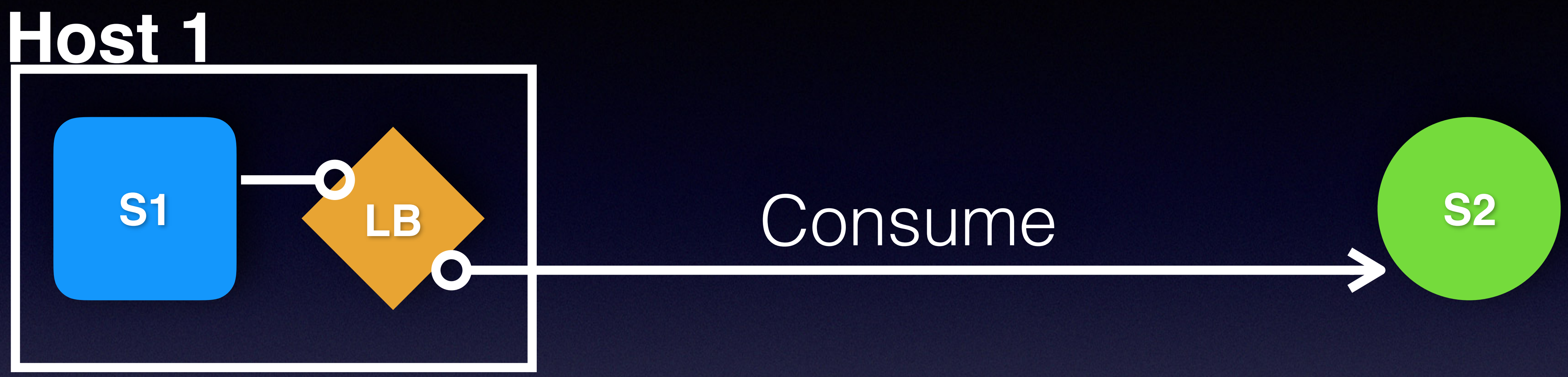
- Same logic in different languages
- May have too many connections to registration centre (e.g. Zookeeper)

In-app registration & discovery

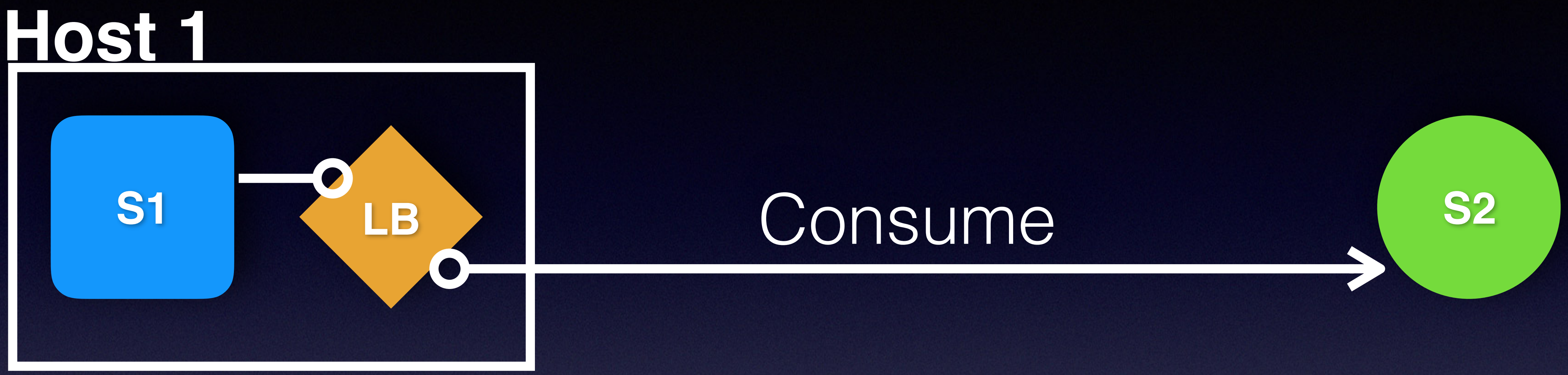


- Same logic in different languages
- May have too many connections to registration centre (e.g. Zookeeper)
- Unable to use for things you don’t wrote (e.g. Nginx, Jenkins)

Client-side load balancer

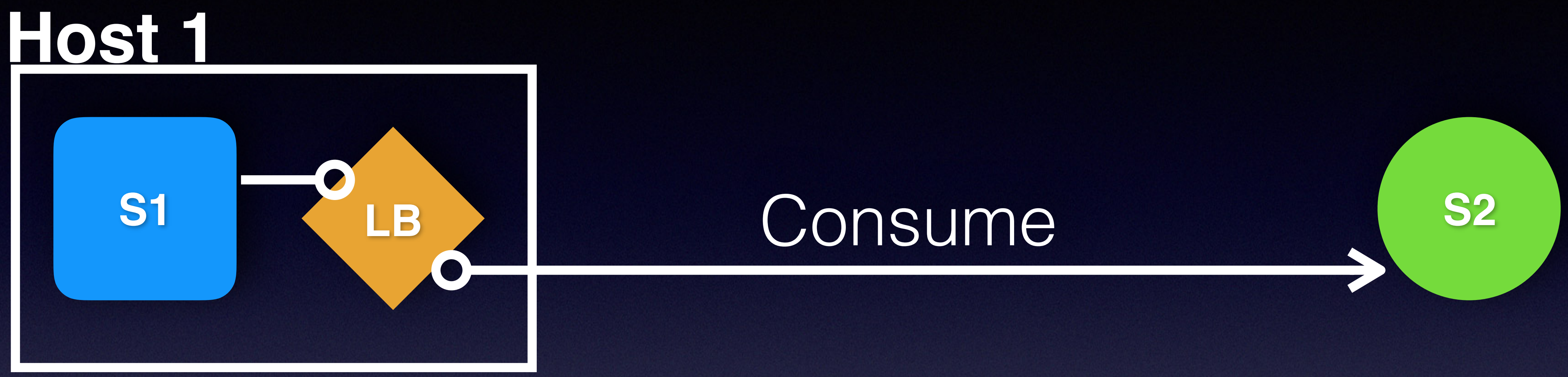


Client-side load balancer



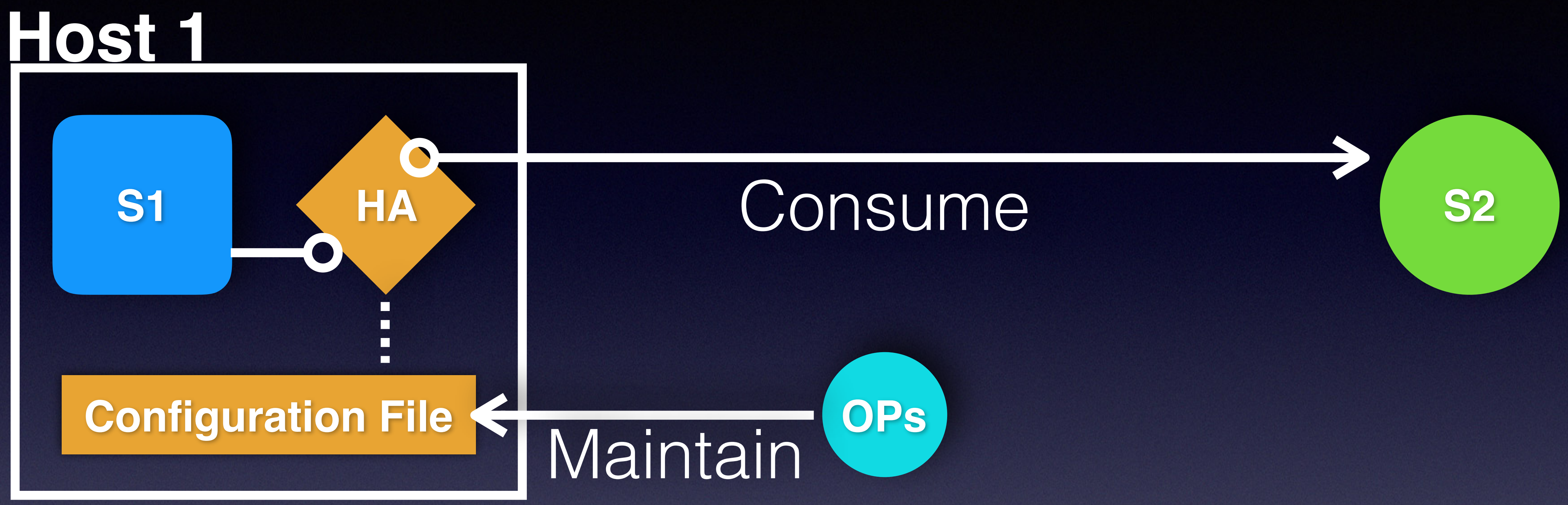
- **Very scalable**

Client-side load balancer

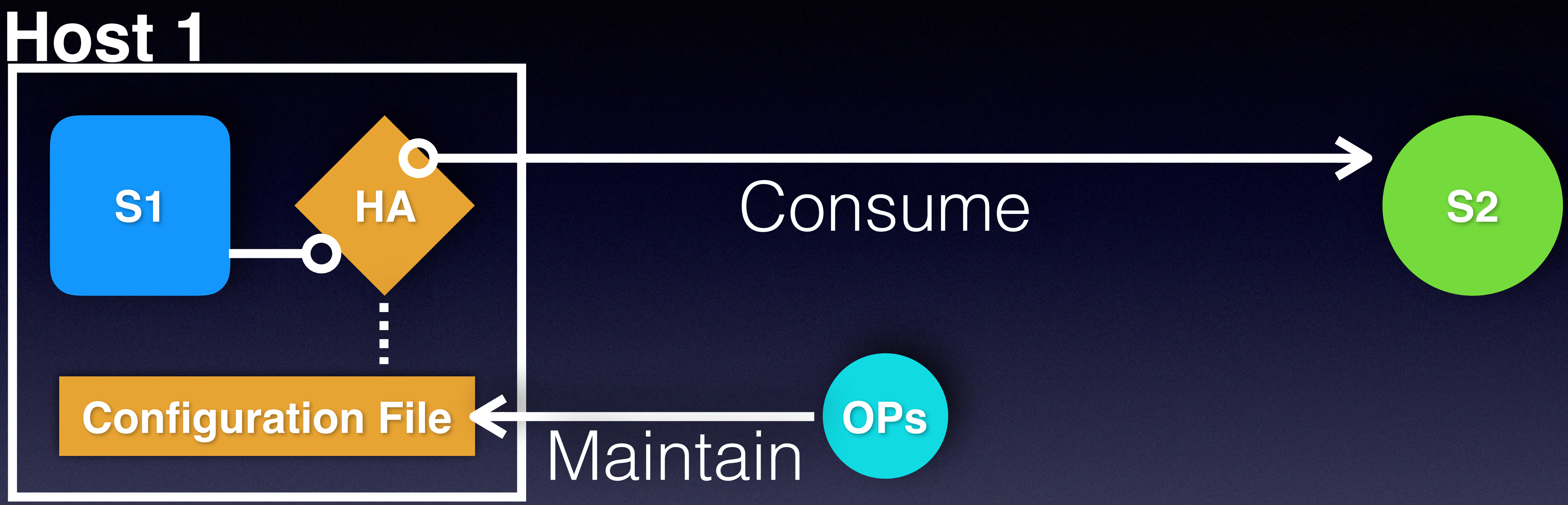


- Very scalable
- Cons may differ according to how you do it

How HAProxy is used

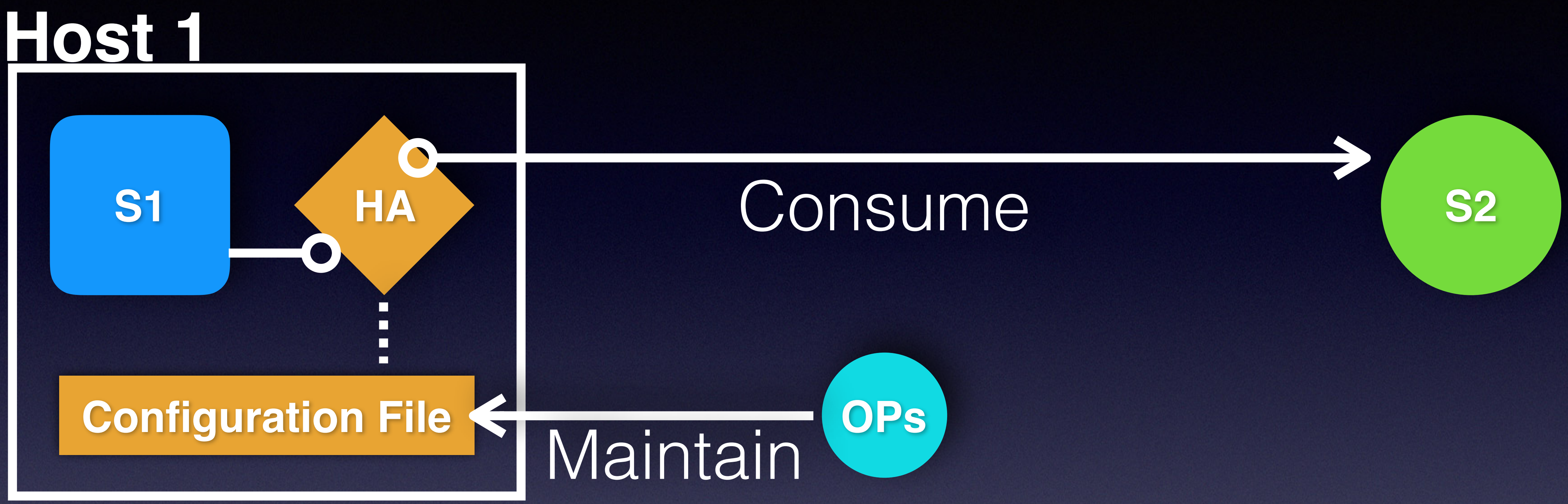


How HAProxy is used



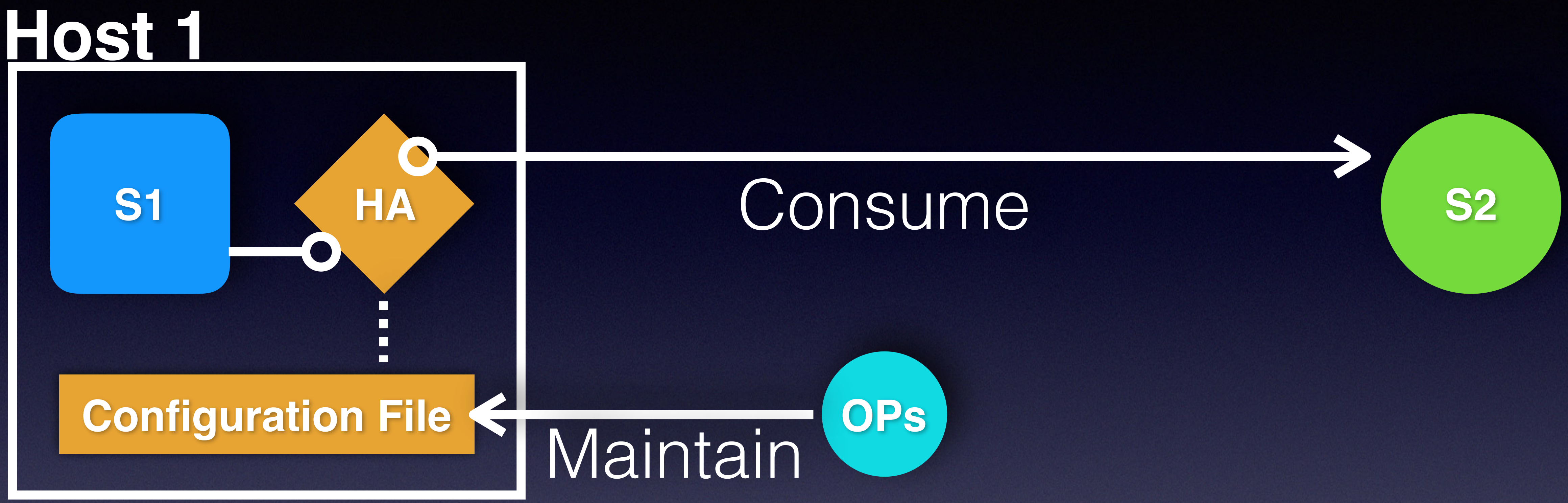
- Restart is required after configuration change

How HAProxy is used



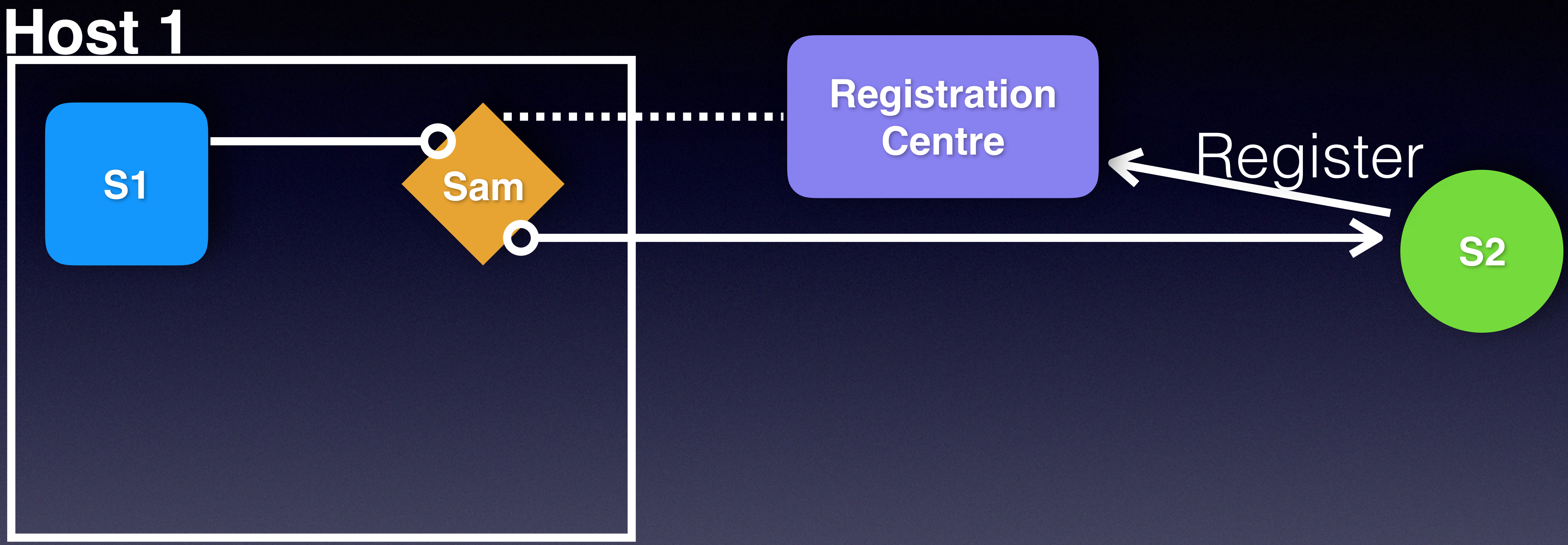
- Restart is required after configuration change
- Configuration files are hard to maintain

How HAProxy is used

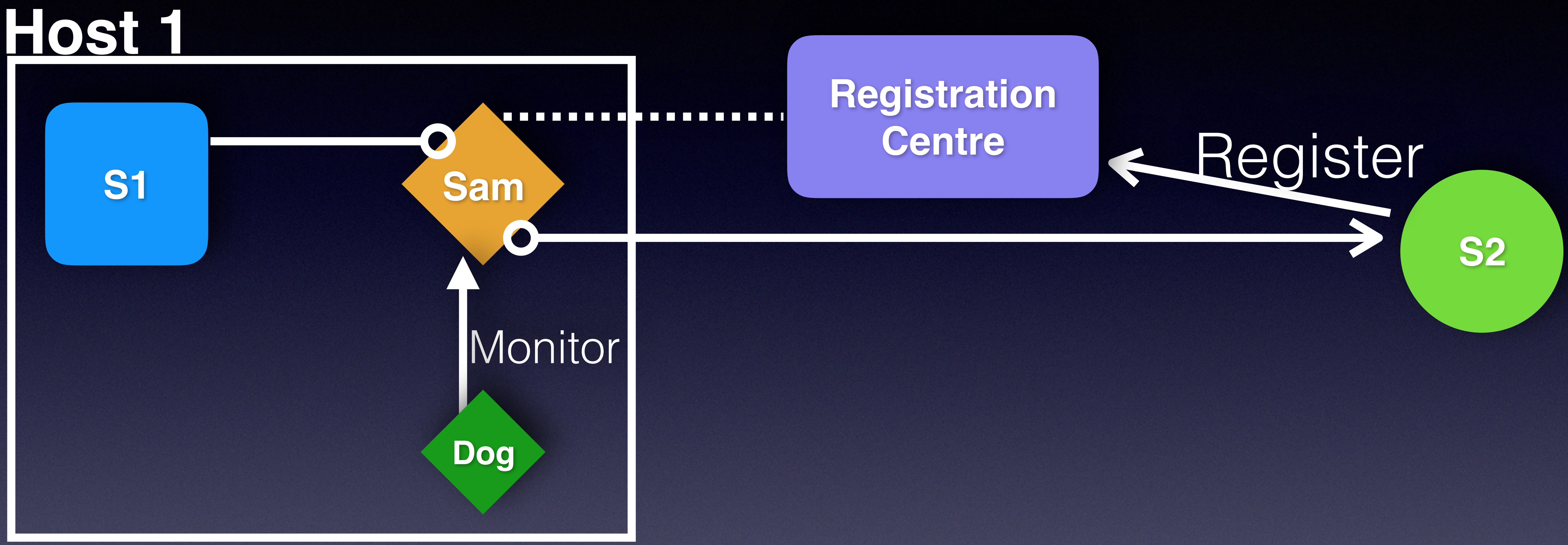


- Restart is required after configuration change
- Configuration files are hard to maintain
- Human brings failures

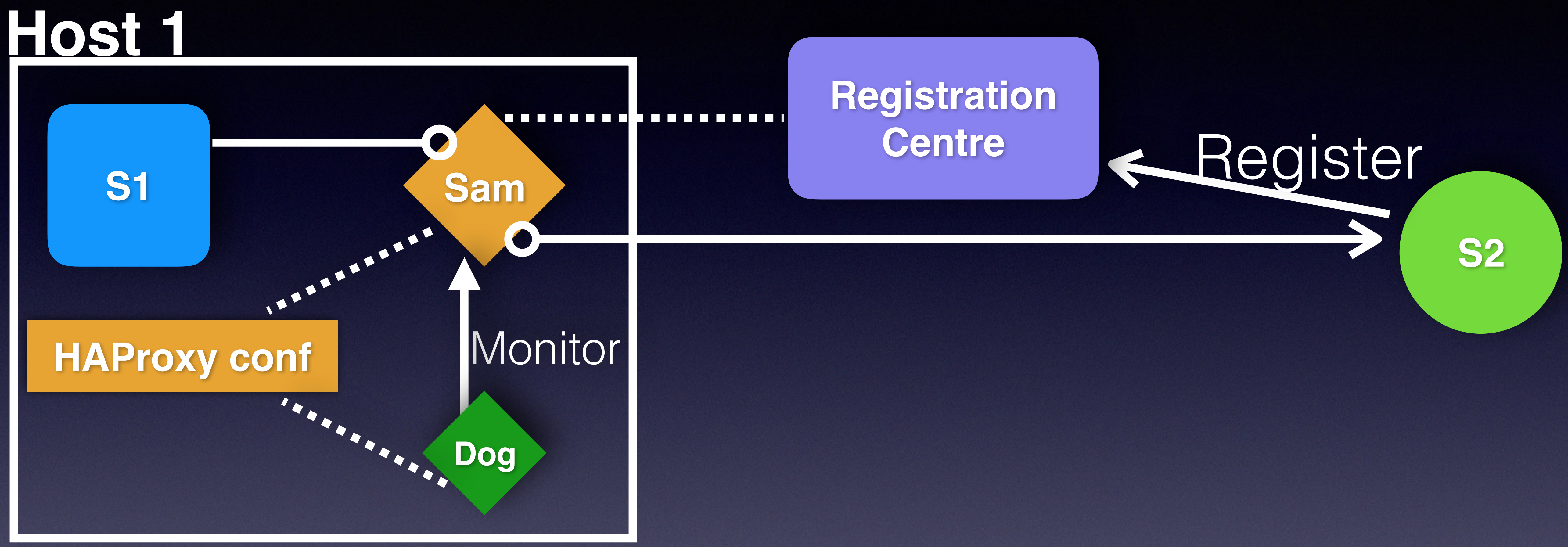
Samaritan overview



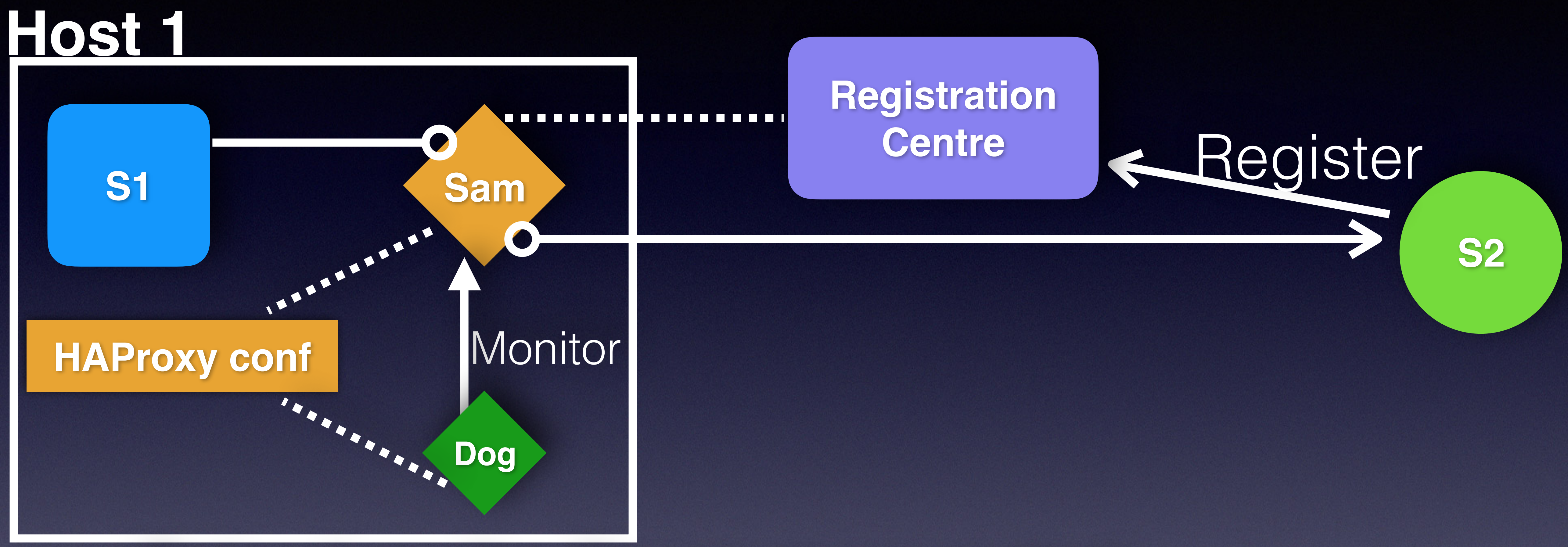
Samaritan overview



Samaritan overview

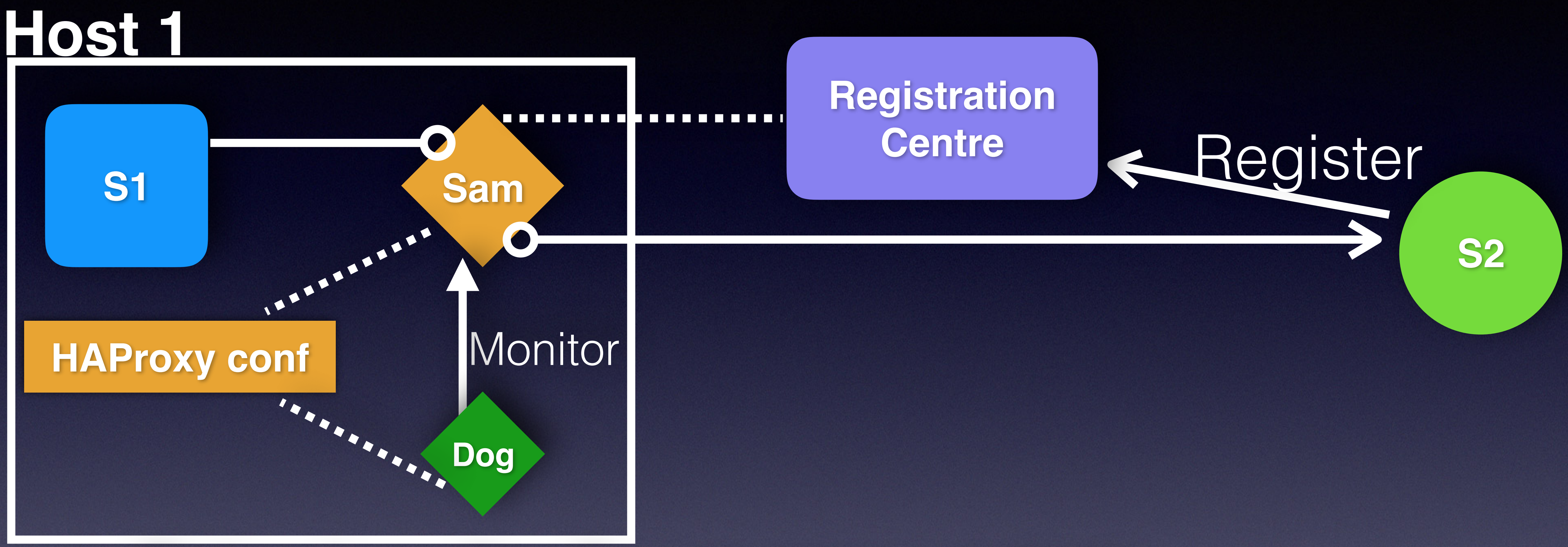


Samaritan overview



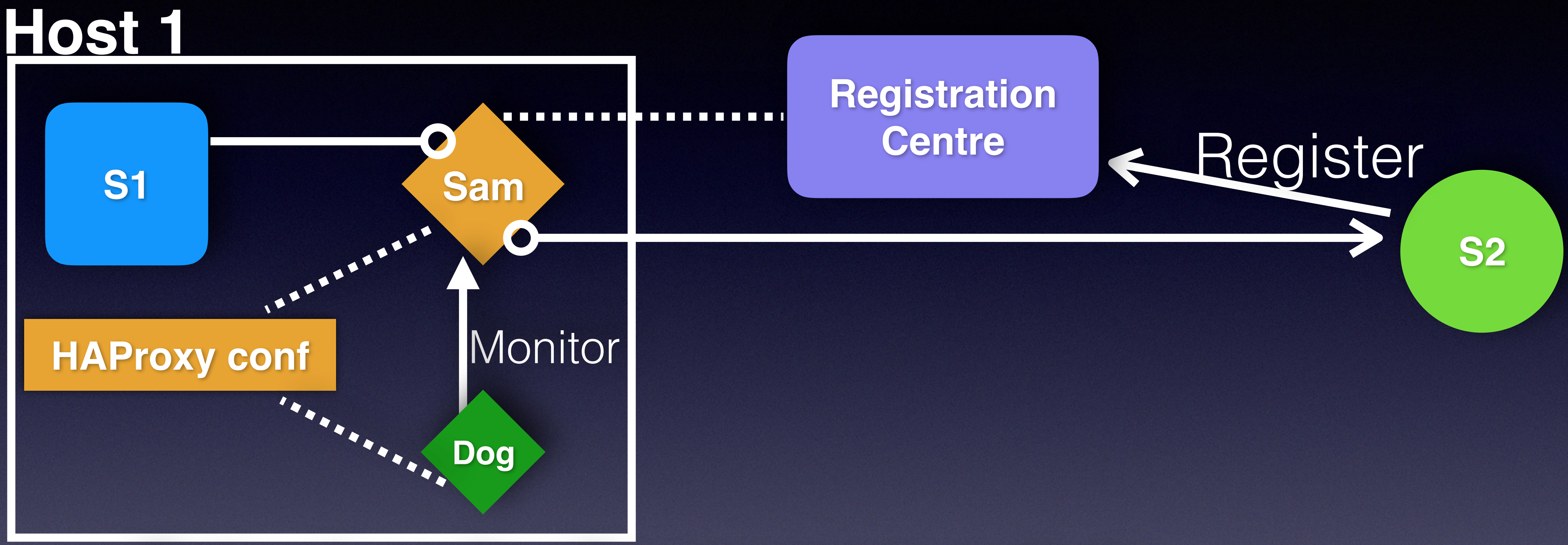
- Reload on-the-fly, no more restart

Samaritan overview



- Reload on-the-fly, no more restart
- Zero configurations at client-side, no more configurations files

Samaritan overview



- Reload on-the-fly, no more restart
- Zero configurations at client-side, no more configurations files
- No human involved, OPs are relieved

Why Samaritan?

Why Samaritan?

- **Reload on-the-fly**

Why Samaritan?

- **Reload on-the-fly**
- **Easy to deploy & maintain**

Why Samaritan?

- **Reload on-the-fly**
- **Easy to deploy & maintain**
- **Simple & Solid**
  (Compared to managed HAProxy)

Thanks