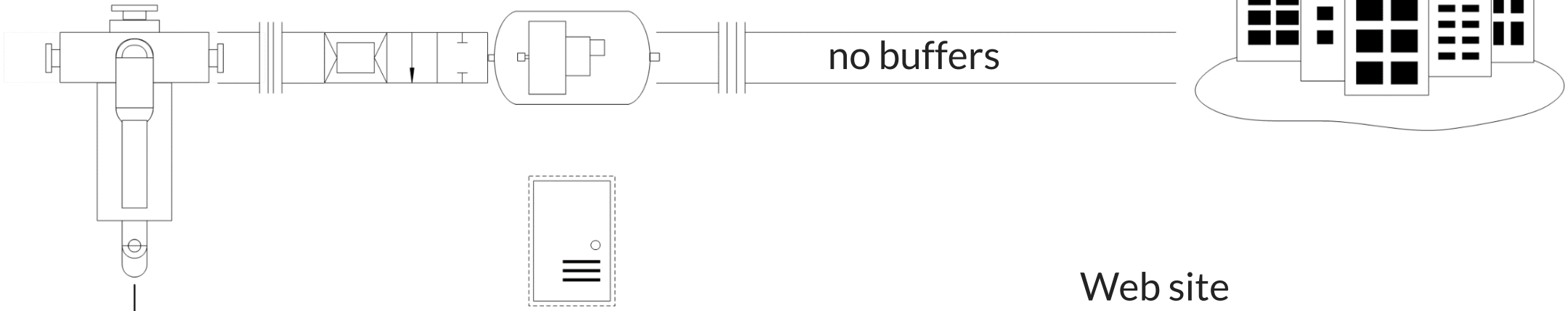
Running a real-world mission-critical system on Azure

Dutch Azure Meetup

Loek Duys

@LDuys

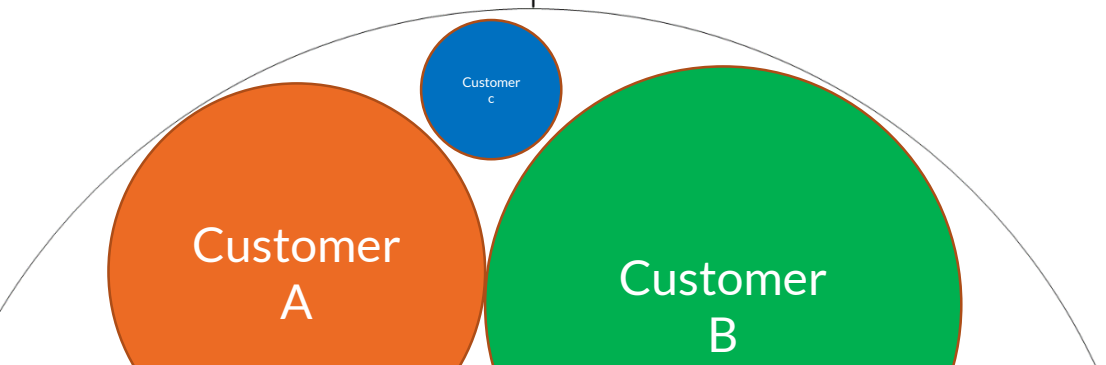
A ‘bank’ for natural gas



Web site
API’s
Message Gateway

inject /
withdraw

Time critical



How we deal with troubles and keep the system alive…

Disaster

Fire drills

Fail-over

Backup

Geo-replication

Continuous Delivery

Infra as Code

SecDevOps

Containers

Commodity hardware

Compute Clusters

Monitoring and Alerting

Scaling

Alternative routes

Transient error handling

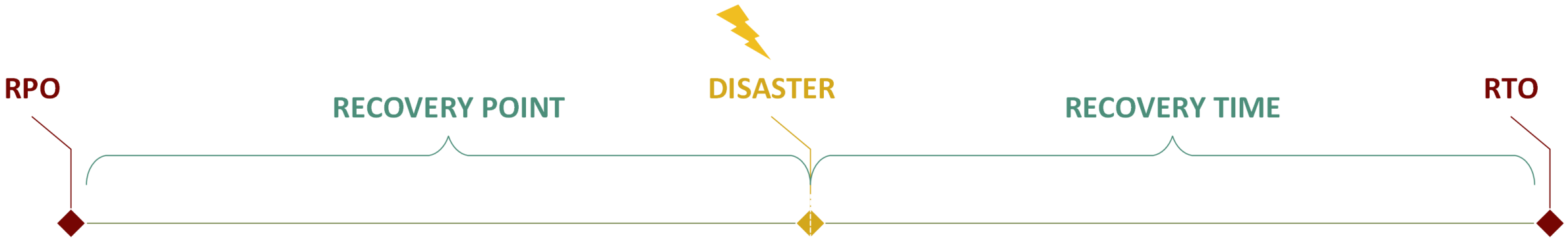
Outbox

Cut dependencies

Retry / Circuit breaker

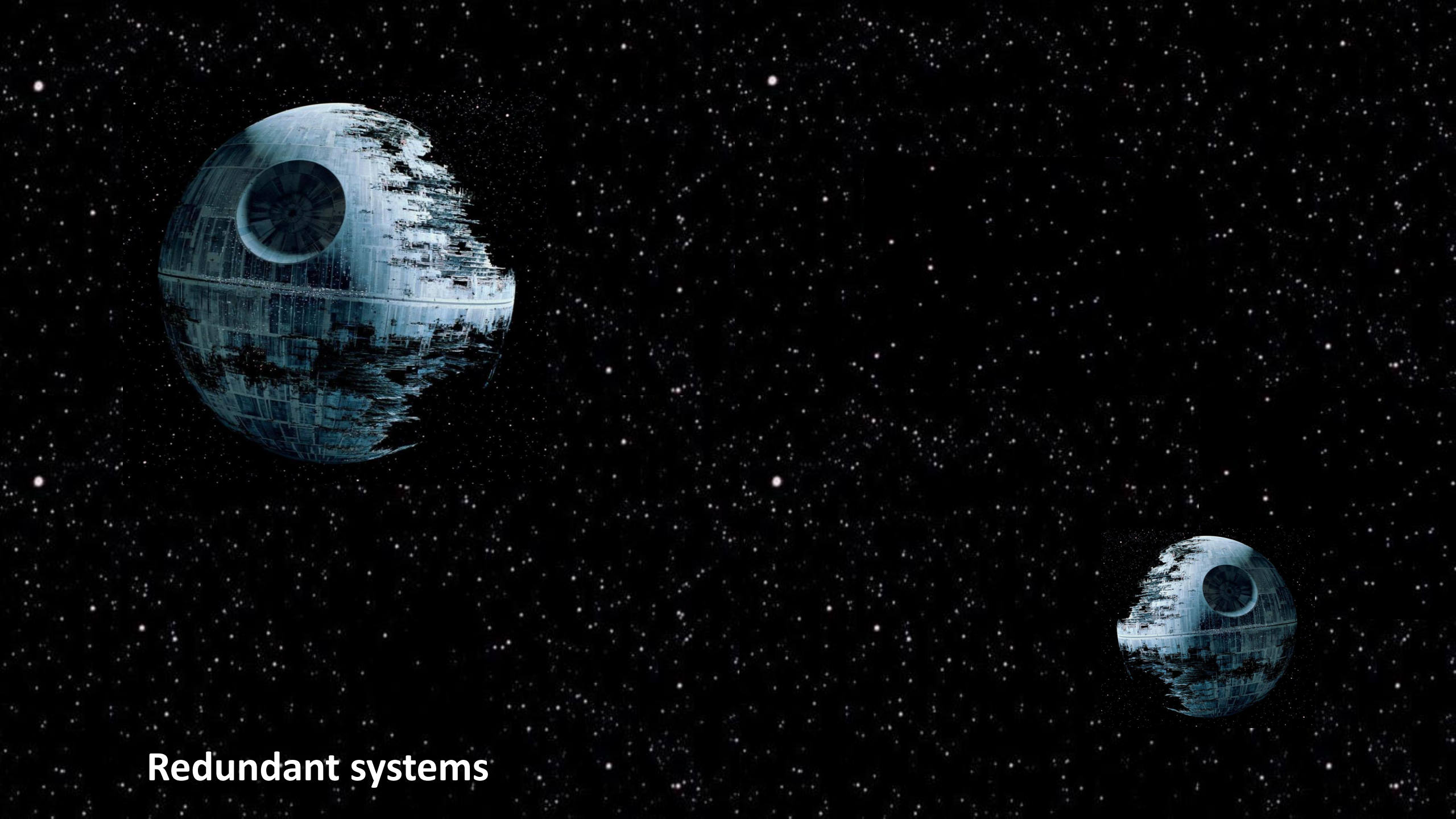
Leader election

Outage



What is acceptable in your situation?


The trouble with single points of failure

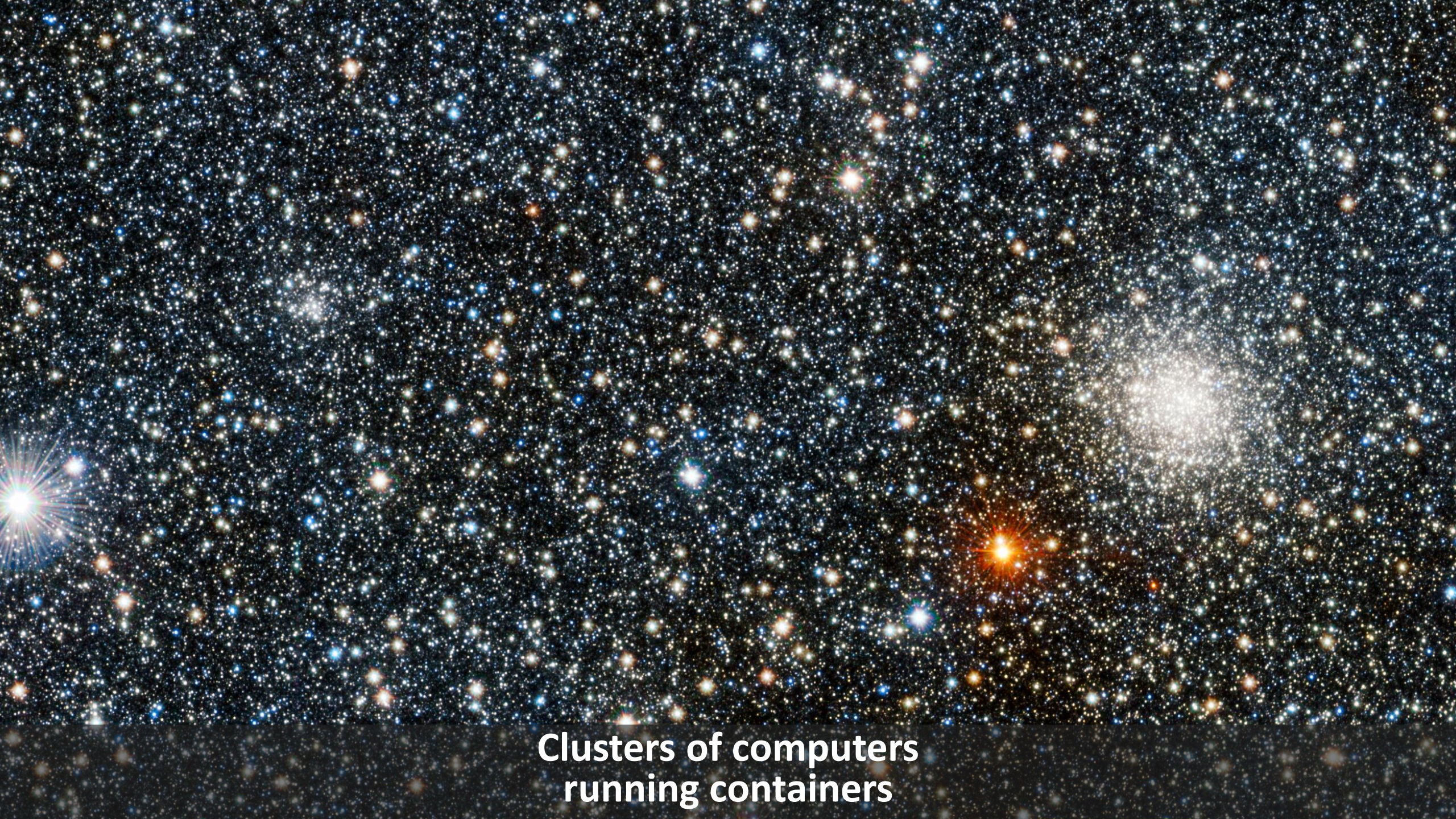

Redundant systems

Smaller version of the original


Autonomous Business Capabilities


Fault tolerance
Degraded performance
BRIDGESTONE


Commodity hardware

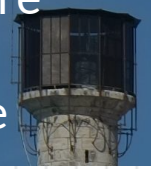

Clusters of computers
running containers

Automated Software and infrastructure deployment

Test software and infrastructure

Spoofing of user identity
Tampering
Repudiation
Information disclosure
Denial of service
Elevation of privilege

Authentication
ACL, TLS
Logging
Cryptography, signing
Scaling, no multipliers
Sandbox, don't trust input



Threat modeling

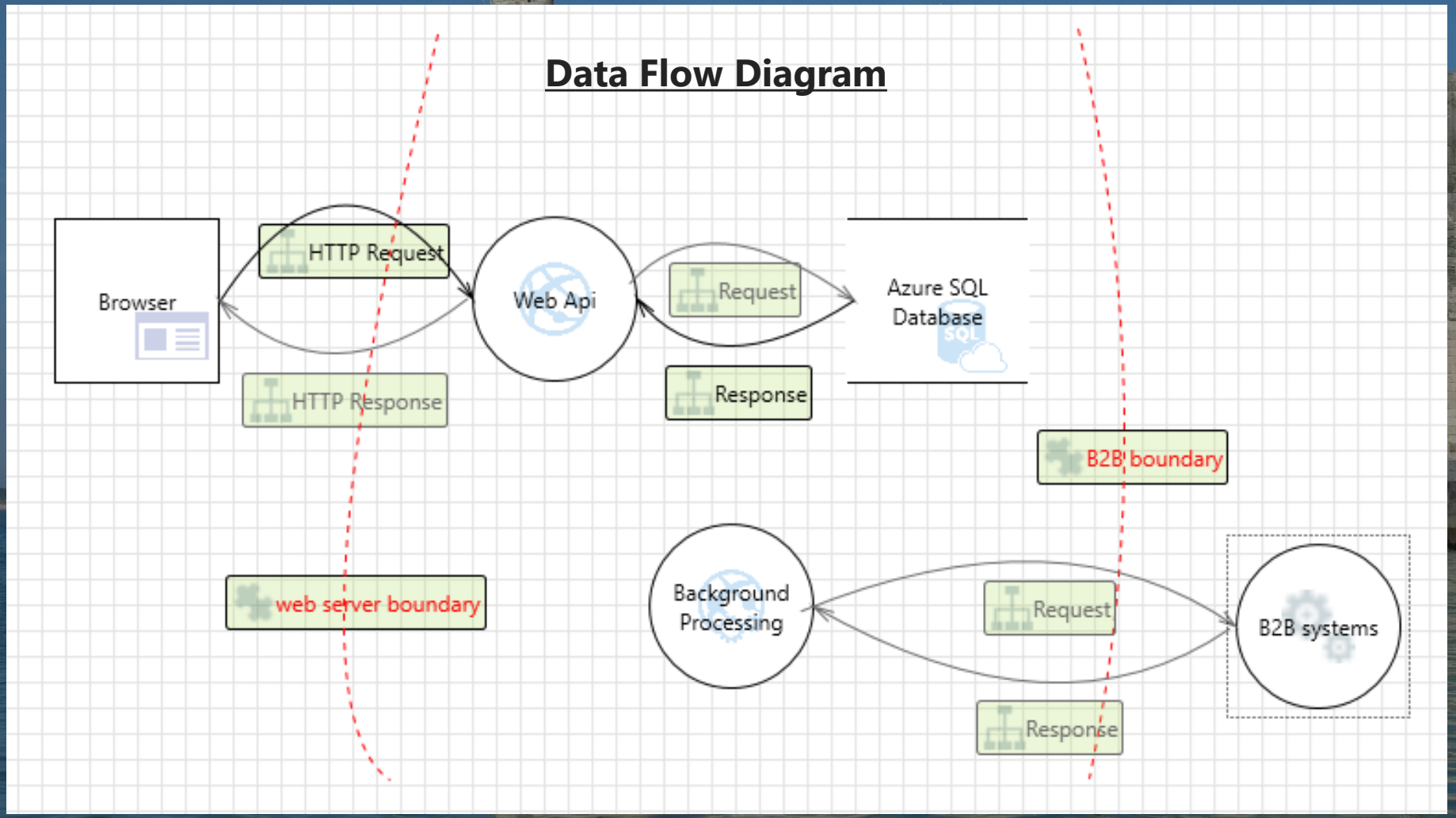
Scanning in pipeline

External audits

Four eyes

Audit trails

Auditable infra




Monitoring – alerting – paging
App Insights
VictorOps
Billing
SQL Server
Slack Bot
Infrastructure
Resource consumption
App health
Security
Performance

Run fire drills

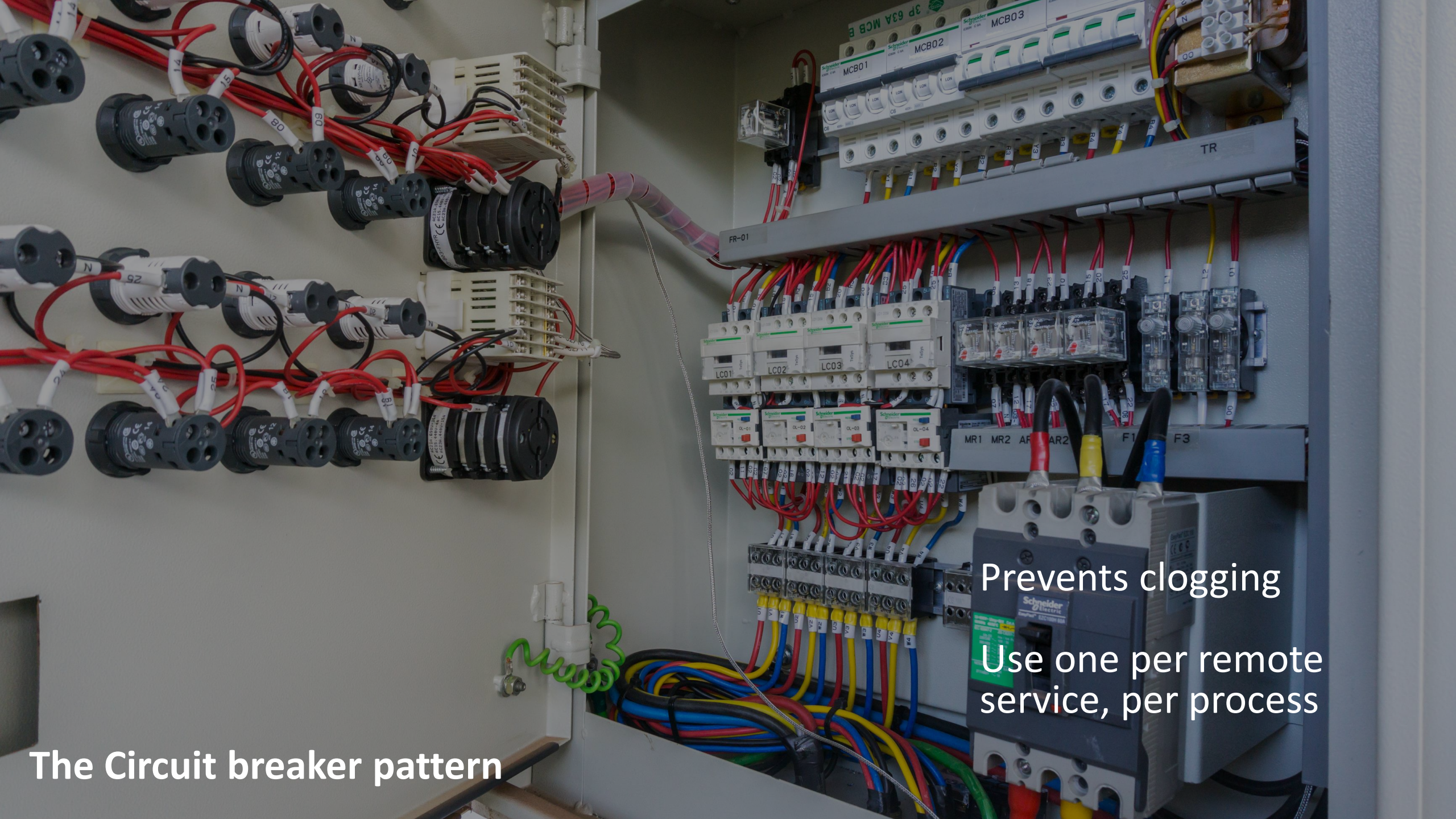

Cutting dependencies

The Retry pattern

Retry

Distinguish transient from permanent errors

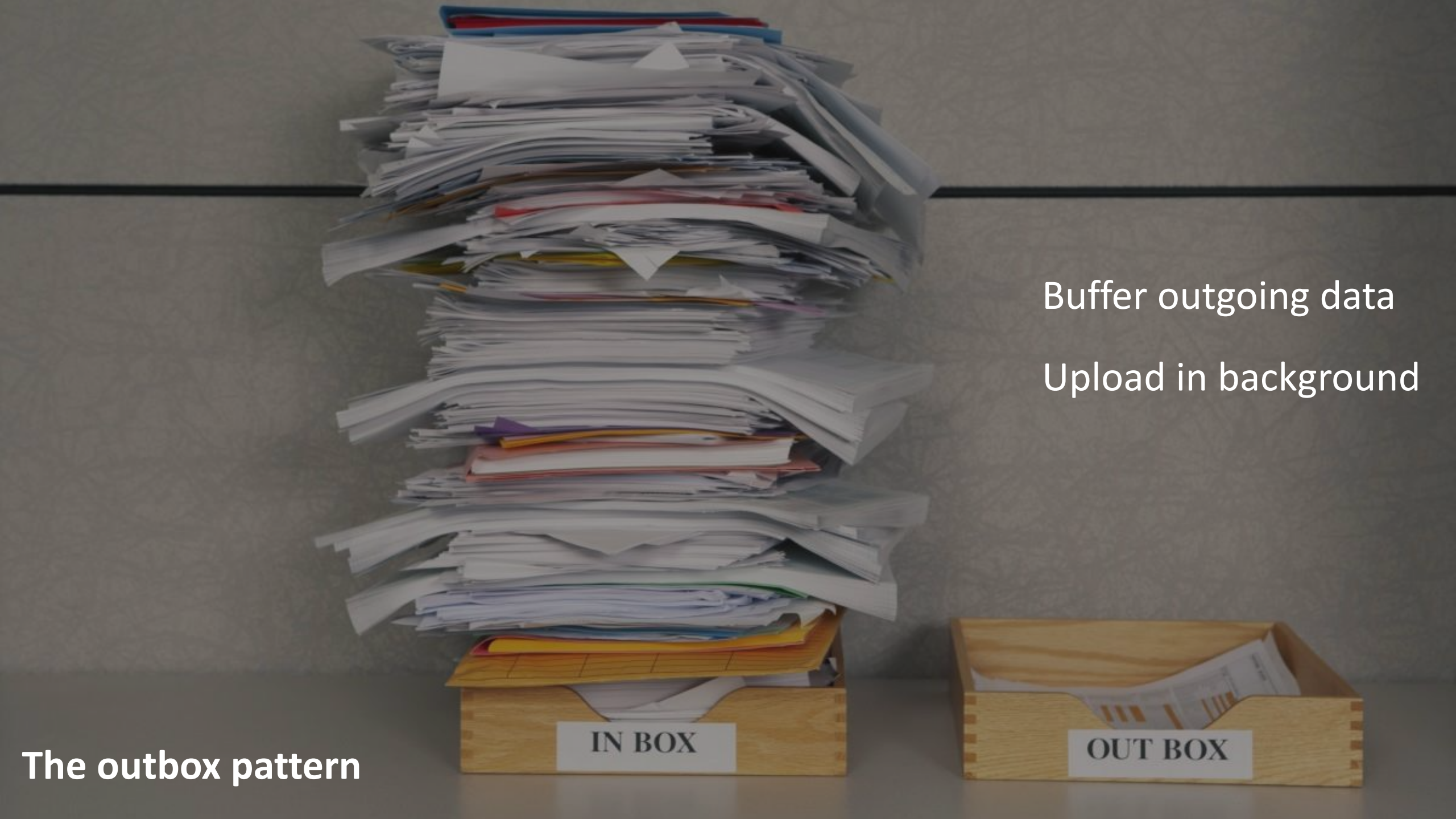
Polly



The Circuit breaker pattern

Prevents clogging

Use one per remote service, per process



The outbox pattern

Buffer outgoing data

Upload in background

Prevents competing consumers

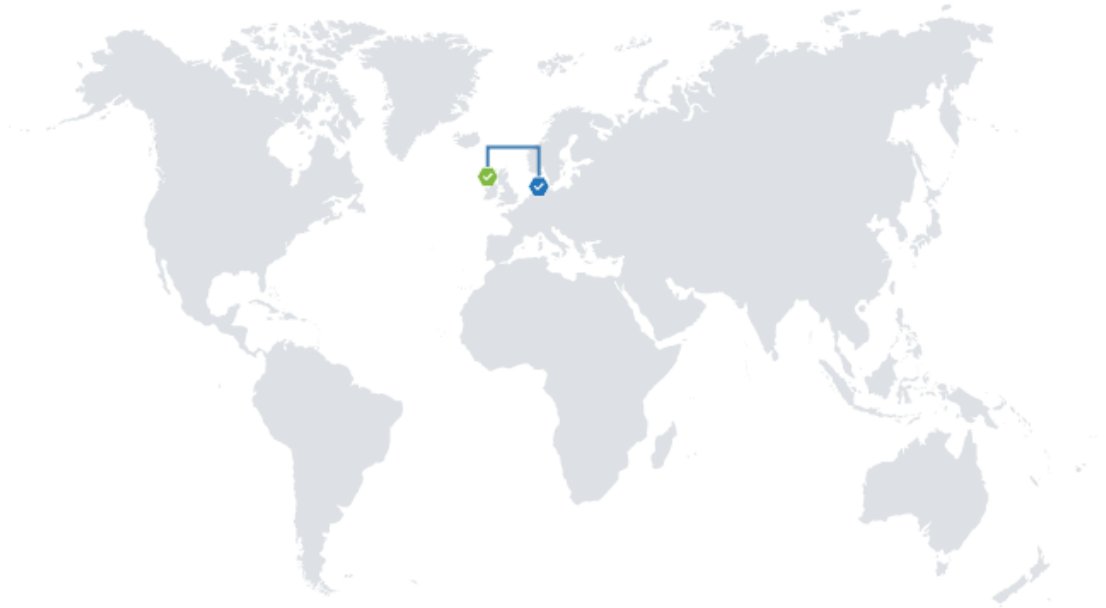
Continuously race to acquire lock and refresh it

The Leader Election pattern

Command Query Responsibility Segregation Event Sourcing

Save | Discard | Add databases | Edit configuration | Remove databases | Failover | Forced Failover | Delete

Configuration details | Databases within group | Databases selected to be added (0) | Databases selected for removal (0)



SERVER	ROLE	READ/WRITE FAILOVER POLICY	GRACE PERIOD
dbp (West Europe)	Primary	Automatic	1 hours
dbfop (North Europe)	Secondary		

Read/write listener endpoint

dbgrp.database.windows.net

Read-only listener endpoint

dbgrp.secondary.database.windows.net

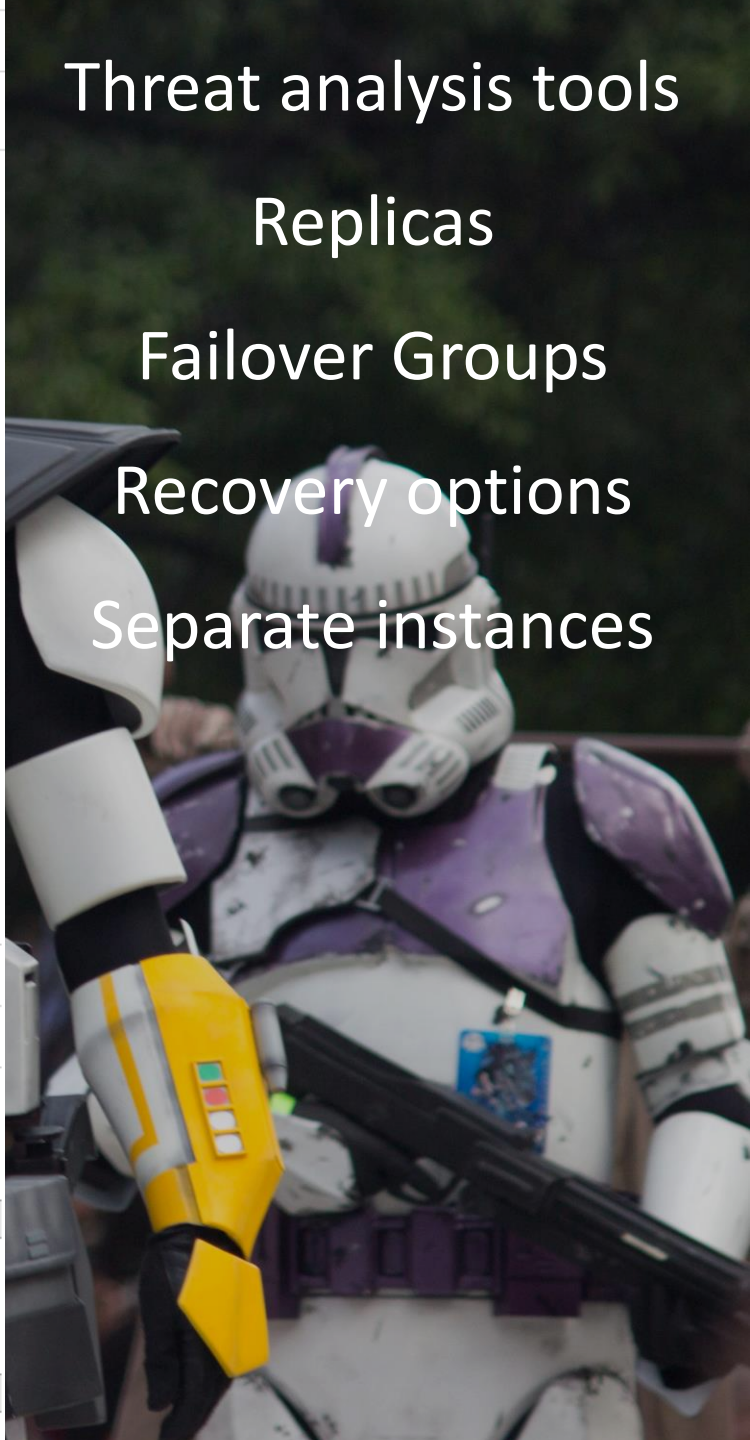
Threat analysis tools

Replicas

Failover Groups

Recovery options

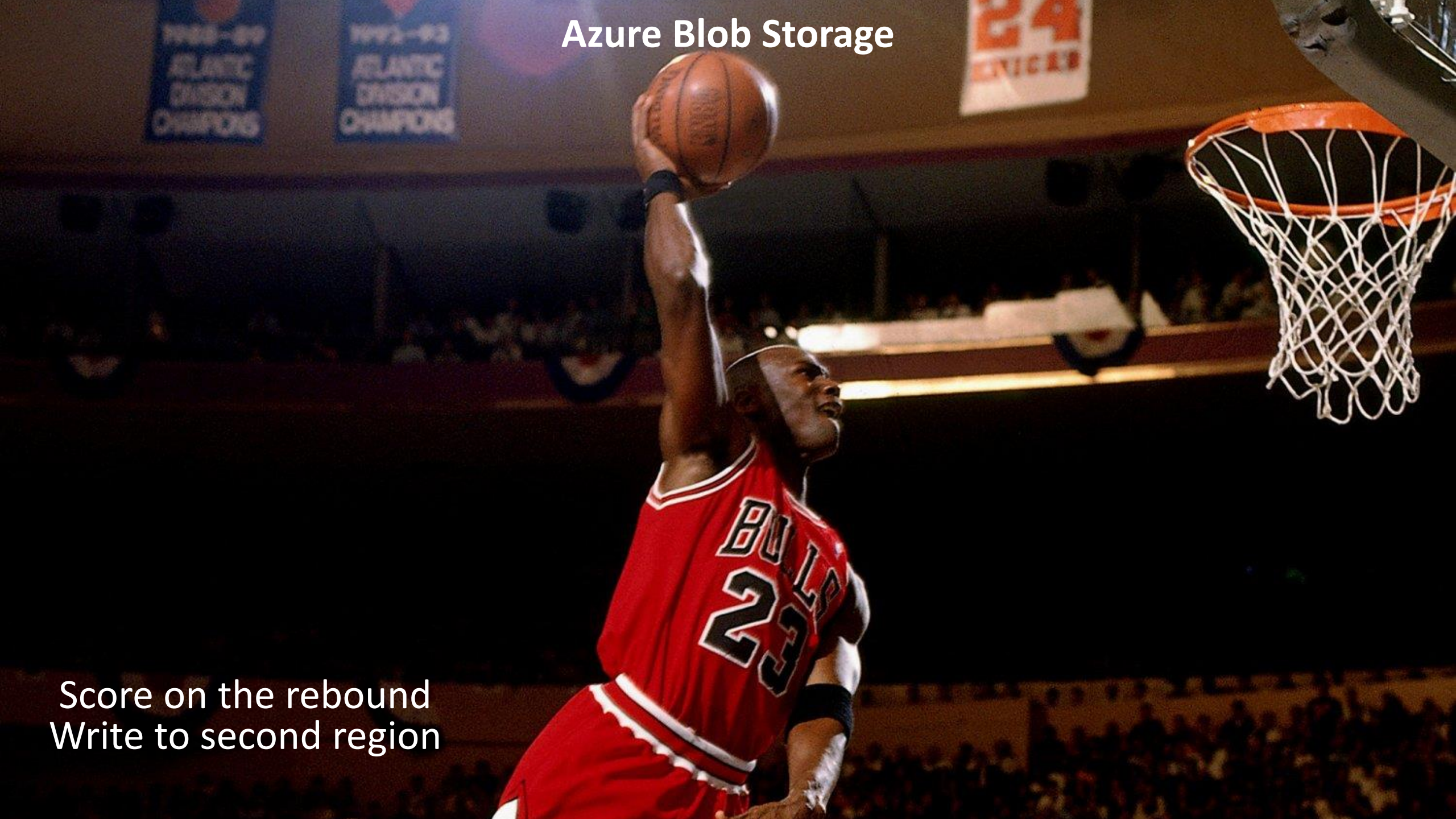
Separate instances



Azure Service Bus

Use multiple buses

Azure Blob Storage



Score on the rebound
Write to second region

Use the Premium tiers

Use Availability Zones

Cloud platforms

Azure SQL Database
- Geo replication
- Failover groups

Storage
- RA GRS
- Soft delete
- Failover (preview)
- AzCopy

Azure AD
- Global
- MFA
- Federation

Azure DevOps
- CI/CD
- Automate everything

Availability Zones
- Co-location in region
- SQL/IP/VM/LB/SB/ST

Service Bus
- Geo redundancy
- Partitioning

Some of our mistakes

Disaster

Fire drills

Fail-over

Backup

Geo-replication

Continuous Delivery

Infra as Code

SecDevOps

Containers

Commodity hardware

Compute Clusters

Monitoring and Alerting

Scaling

Alternative routes

Transient error handling

Outbox

Cut dependencies

Retry / Circuit breaker

Leader election

Outage

https://github.com/loekd/MisCritAz

Thank you!