How we went from being astronauts to being mission control

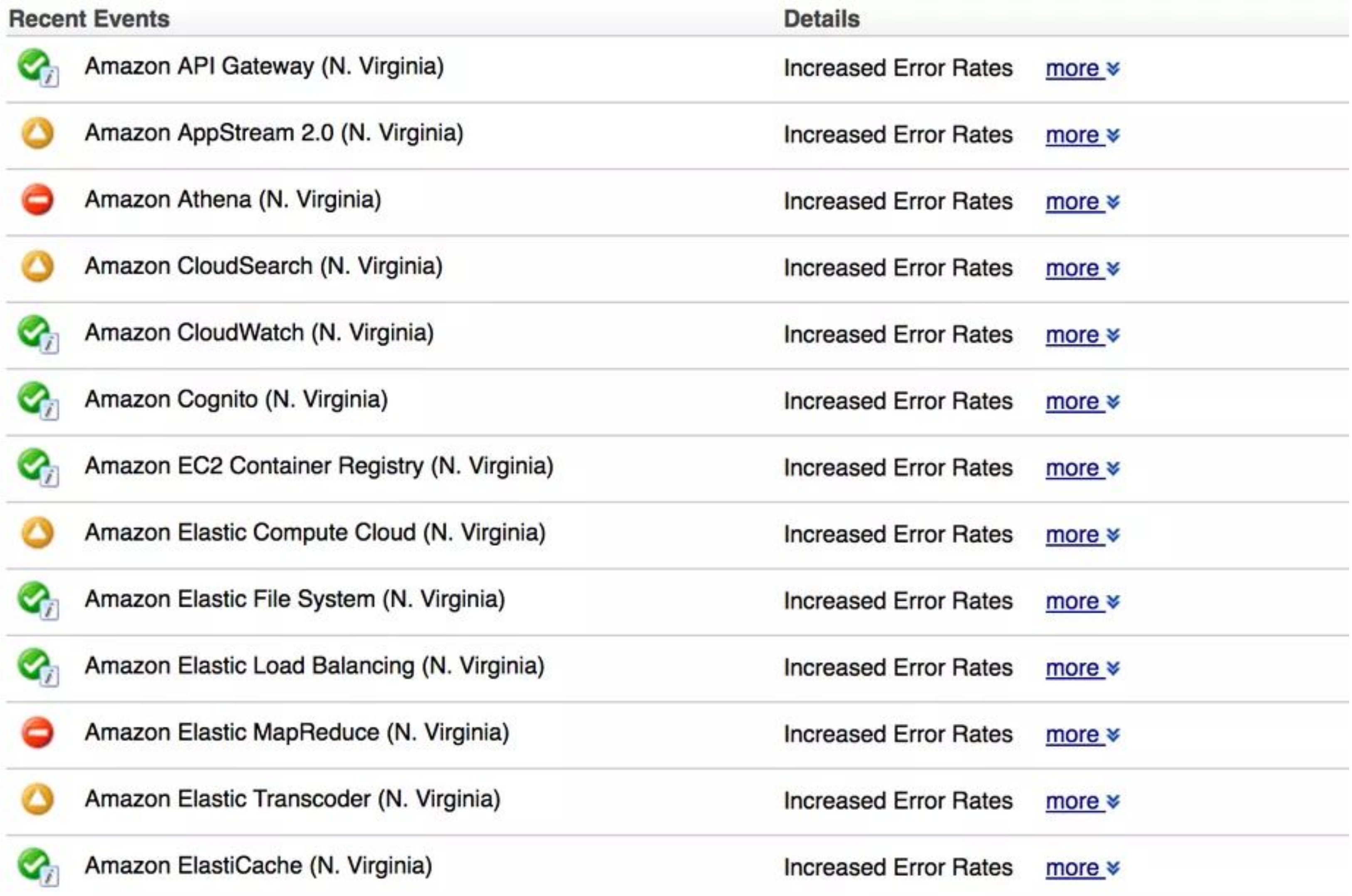
Managing systems in an age of dynamic complexity

Laura Nolan, Stanza

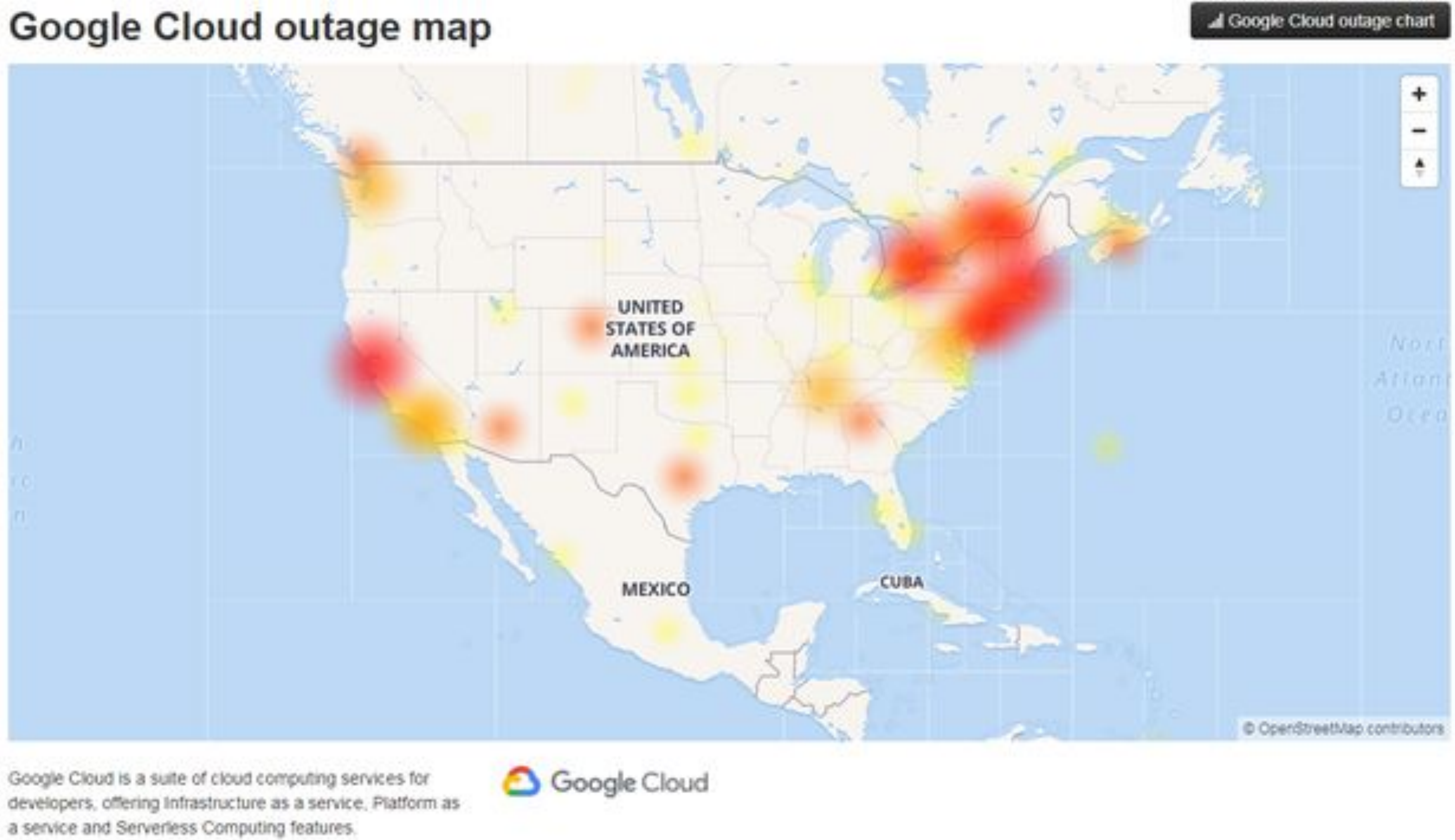
About Laura Nolan

- Not a real astronaut (sorry)
- Principal Software Engineer at Stanza
- Contributor to *Site Reliability Engineering* ('the SRE book'), *Seeking SRE*, *97 Things Every SRE Should Know*, InfoQ, and (most importantly) USENIX ;login:
- Campaigner for a ban treaty against Lethal Autonomous Weapons: stopkillerrobots.org
- @lauralifts on Twitter

Consider cloud reliability…

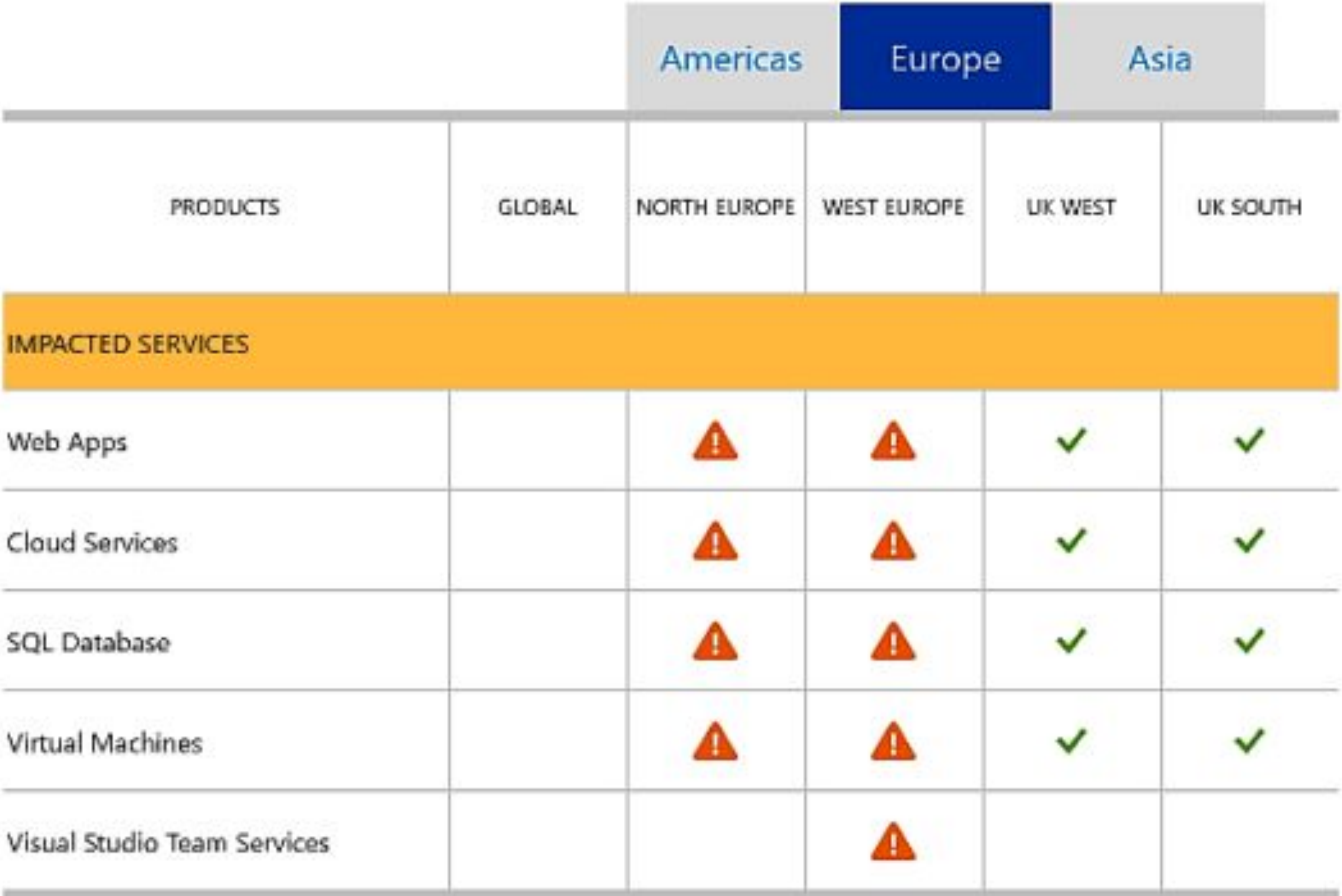


Recent Events		Details	
✓	Amazon API Gateway (N. Virginia)	Increased Error Rates	more ⌄
⚠	Amazon AppStream 2.0 (N. Virginia)	Increased Error Rates	more ⌄
⛔	Amazon Athena (N. Virginia)	Increased Error Rates	more ⌄
⚠	Amazon CloudSearch (N. Virginia)	Increased Error Rates	more ⌄
✓	Amazon CloudWatch (N. Virginia)	Increased Error Rates	more ⌄
✓	Amazon Cognito (N. Virginia)	Increased Error Rates	more ⌄
✓	Amazon EC2 Container Registry (N. Virginia)	Increased Error Rates	more ⌄
⚠	Amazon Elastic Compute Cloud (N. Virginia)	Increased Error Rates	more ⌄
✓	Amazon Elastic File System (N. Virginia)	Increased Error Rates	more ⌄
✓	Amazon Elastic Load Balancing (N. Virginia)	Increased Error Rates	more ⌄
⛔	Amazon Elastic MapReduce (N. Virginia)	Increased Error Rates	more ⌄
⚠	Amazon Elastic Transcoder (N. Virginia)	Increased Error Rates	more ⌄
✓	Amazon ElastiCache (N. Virginia)	Increased Error Rates	more ⌄



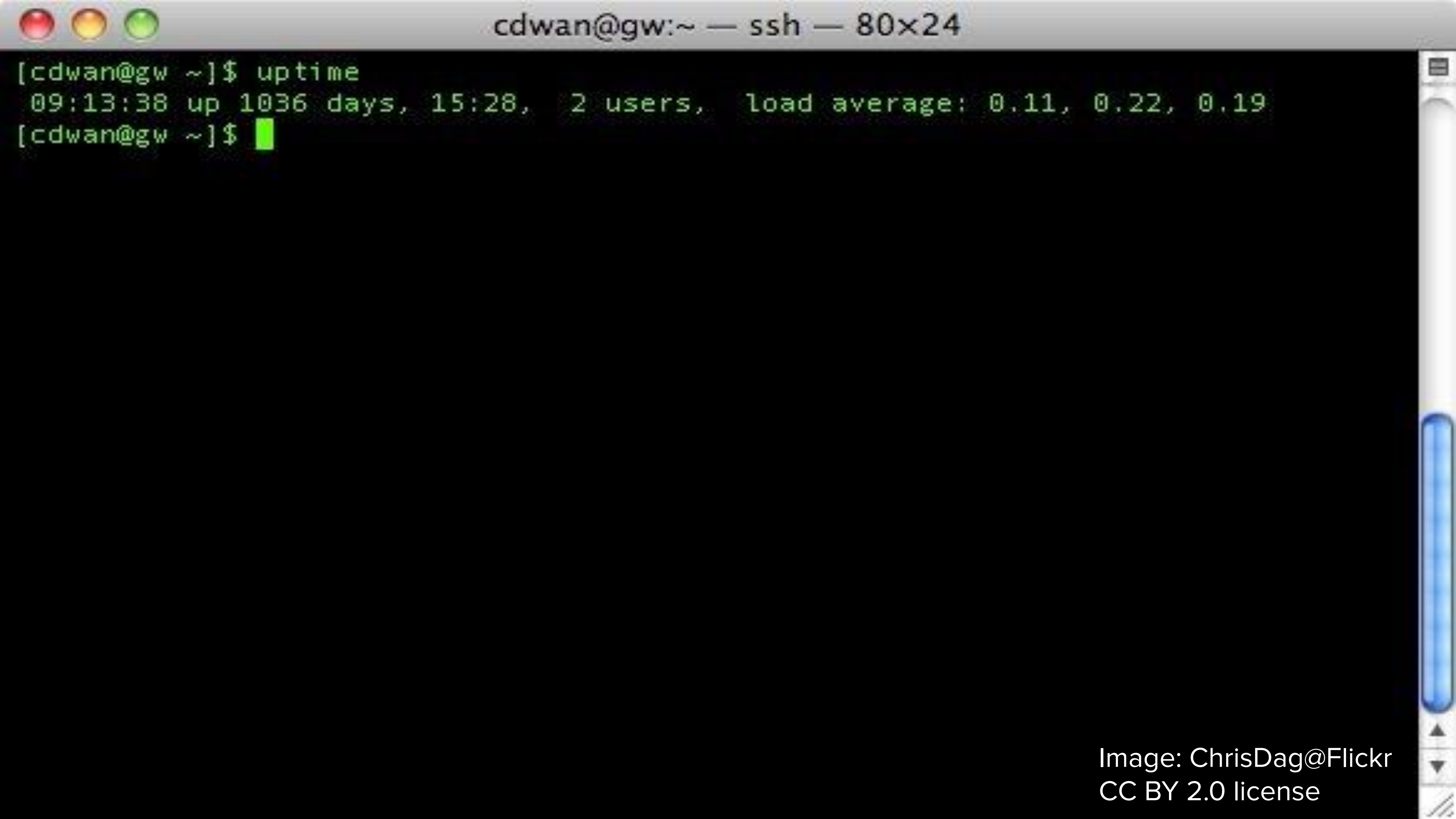
Google Cloud outage map

Google Cloud is a suite of cloud computing services for developers, offering Infrastructure as a service, Platform as a service and Serverless Computing features.



Americas | Europe | Asia

PRODUCTS	GLOBAL	NORTH EUROPE	WEST EUROPE	UK WEST	UK SOUTH
IMPACTED SERVICES					
Web Apps		⚠	⚠	✓	✓
Cloud Services		⚠	⚠	✓	✓
SQL Database		⚠	⚠	✓	✓
Virtual Machines		⚠	⚠	✓	✓
Visual Studio Team Services			⚠		



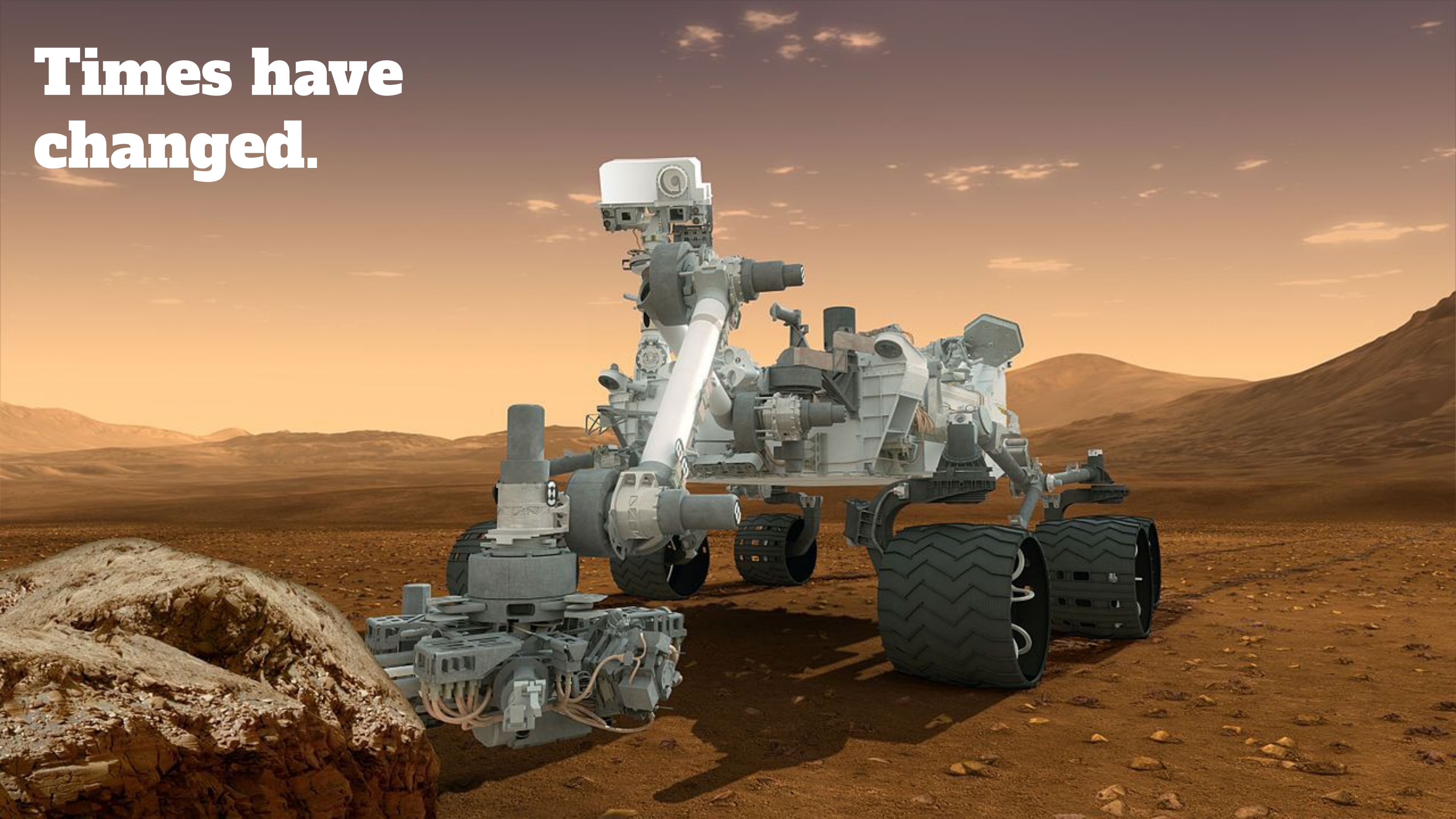
```
[cdwan@gw ~]$ uptime
 09:13:38 up 1036 days, 15:28,  2 users,  load average: 0.11, 0.22, 0.19
[cdwan@gw ~]$ 
```

Image: ChrisDag@Flickr
CC BY 2.0 license

The old ways

- Configuring servers done by hand, or semi-automated
- Humans managing loadbalancer backend pools
- No autoscaling - things already sized for peak
- No job orchestration
- Everything was pretty static

Times have changed.



Automate everything:

- Job orchestration
- Autoscaling number of instances
- Routing, failover and balancing traffic

MERCURY
GEMINI
APOLLO
2019
W
IN GOD WE TRUST
GC
JFM
LIBERTY
UNITED STATES OF AMERICA
FIVE DOLLARS
E PLURIBUS UNUM

Other pressures

- Better performance and latency, especially tail latency
- Reduce repetitive toil of managing a large fleet
- React faster to routine hardware failures
- More consistency in production
- Avoid compliance risks related to engineers touching production

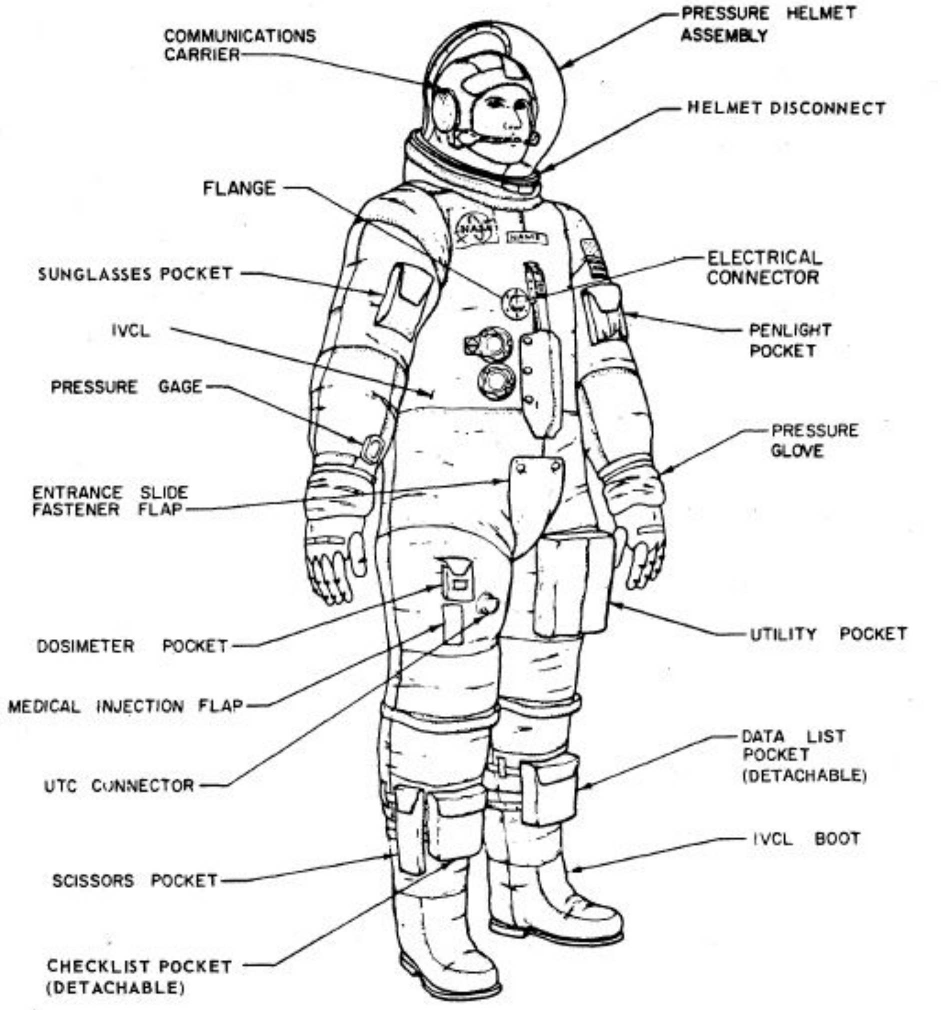
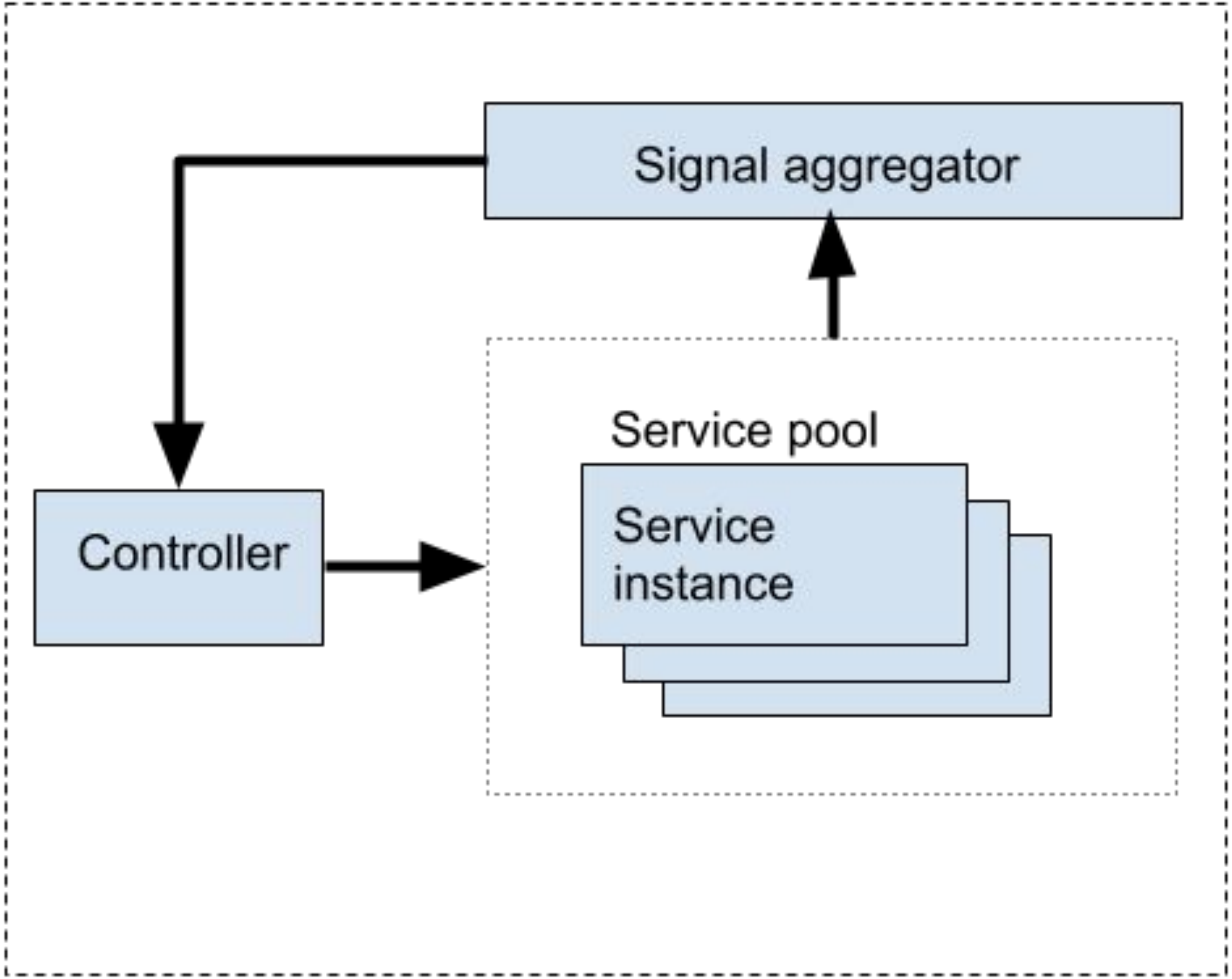


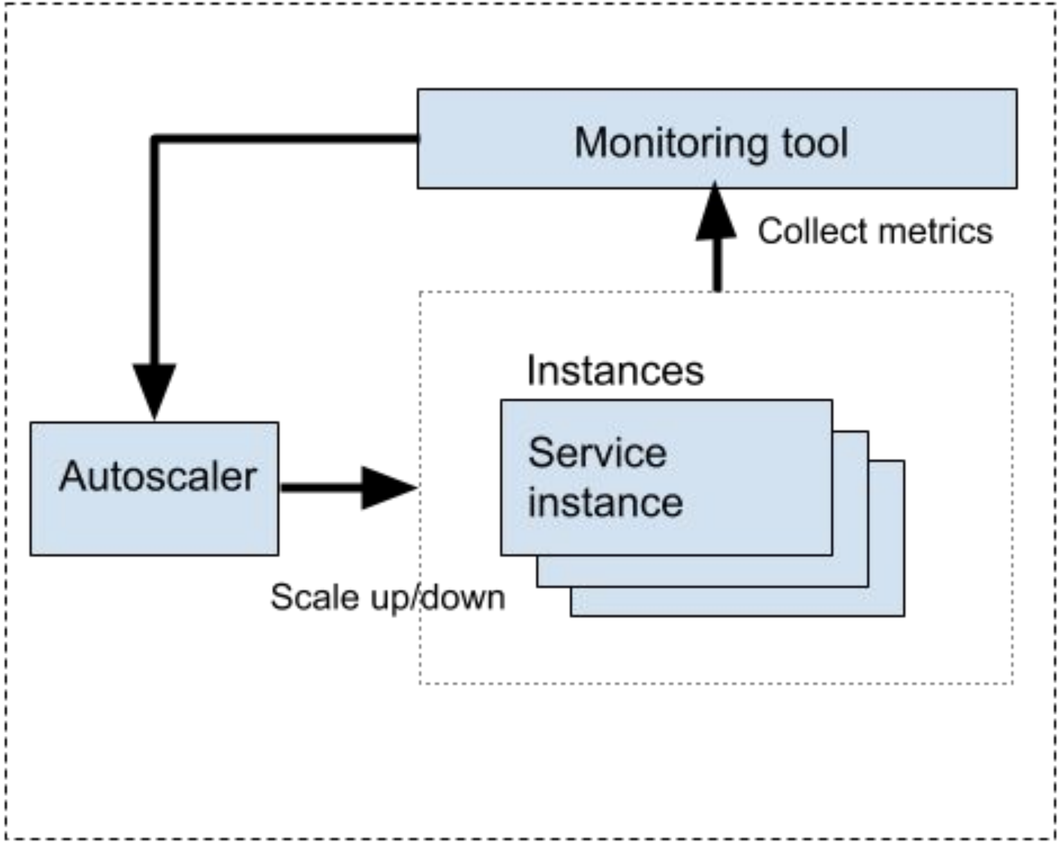
Figure I-4. - Intravehicular pressure garment assembly.

The Dynamic Control Plane Architecture Pattern

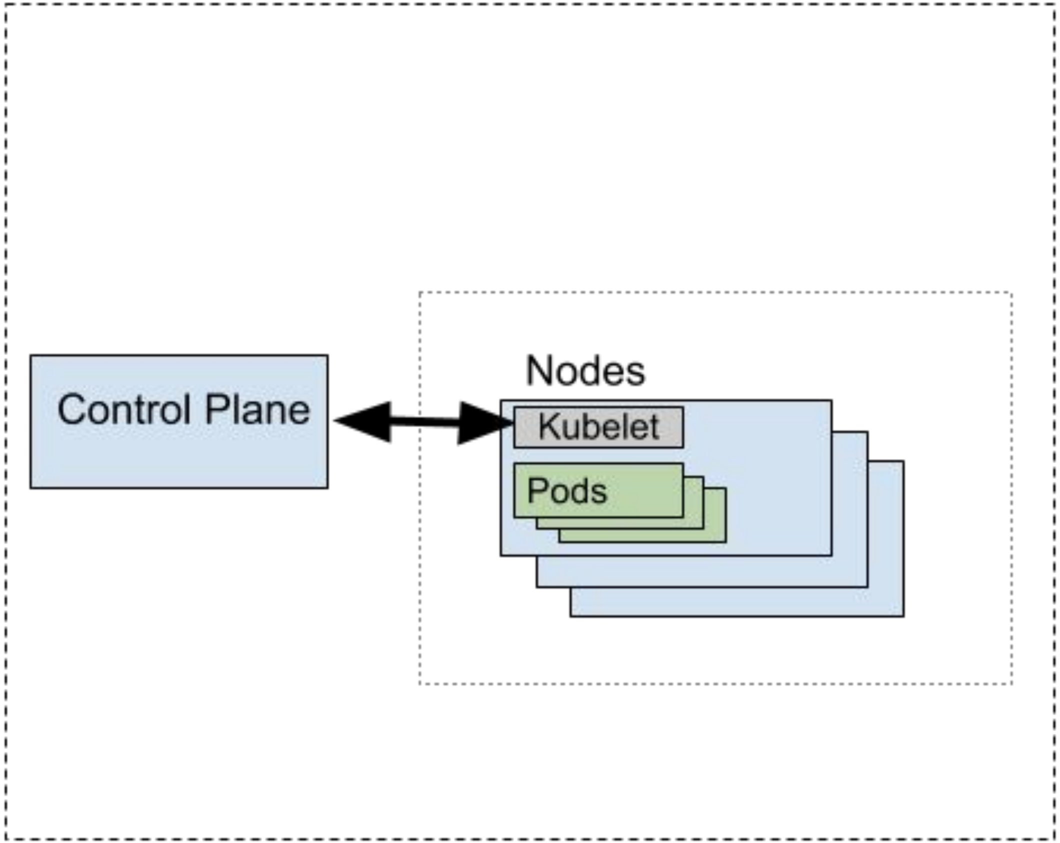
A common architectural pattern in software (and network) operations that arises in order to address global optimisation problems.

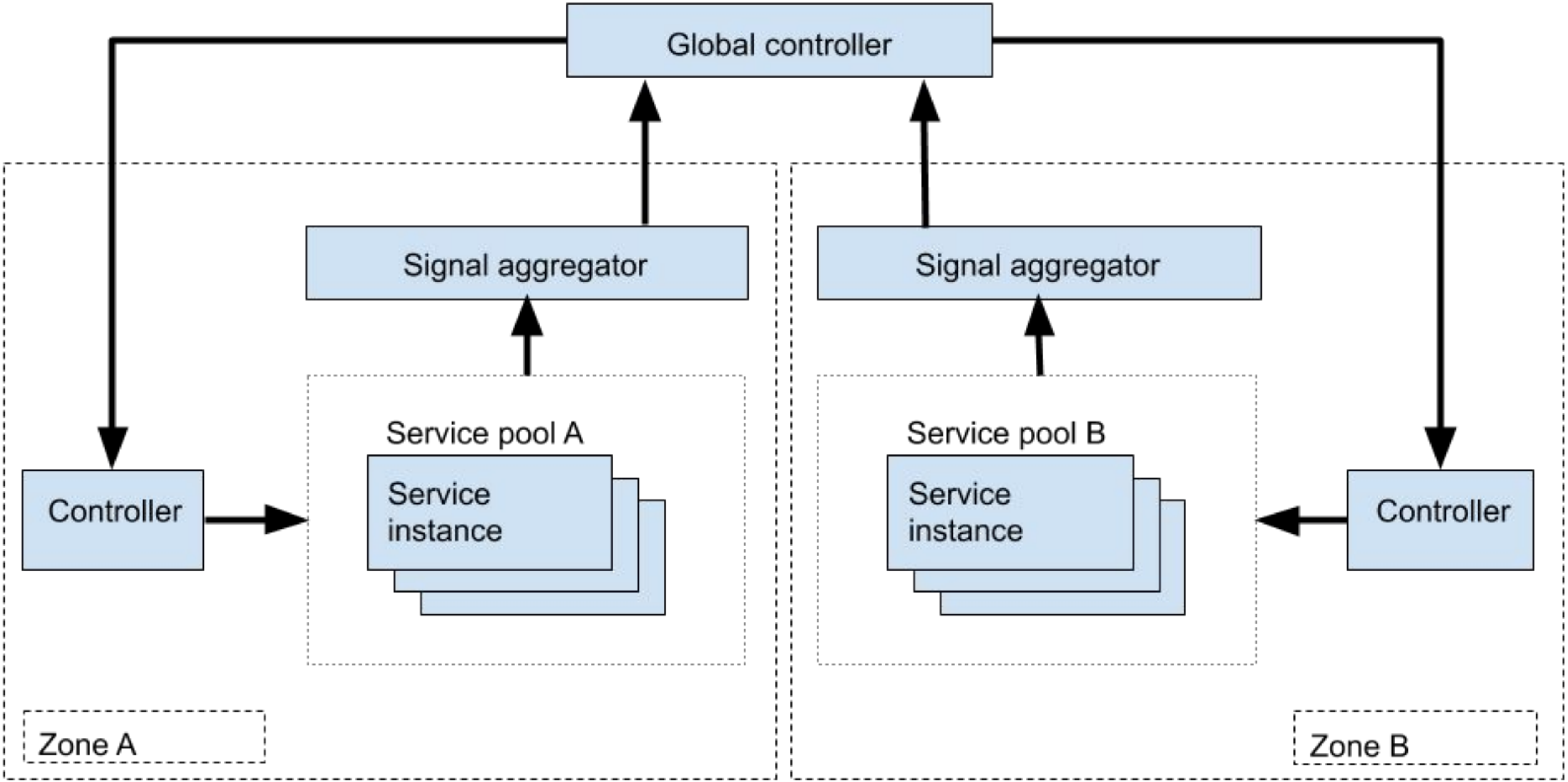

Signal aggregator
Service pool
Service
instance
Controller

Autoscaling group

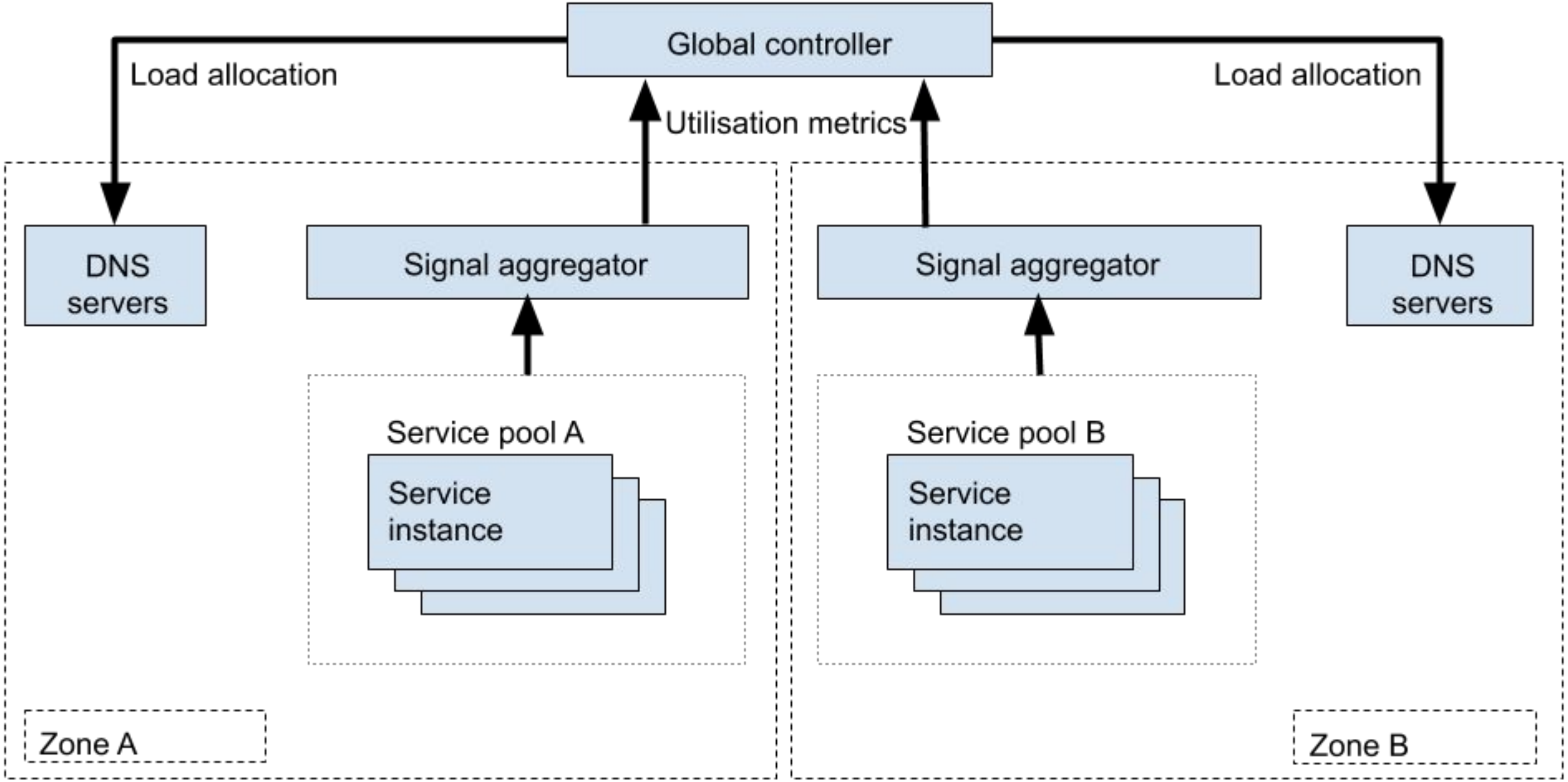


Kubernetes cluster

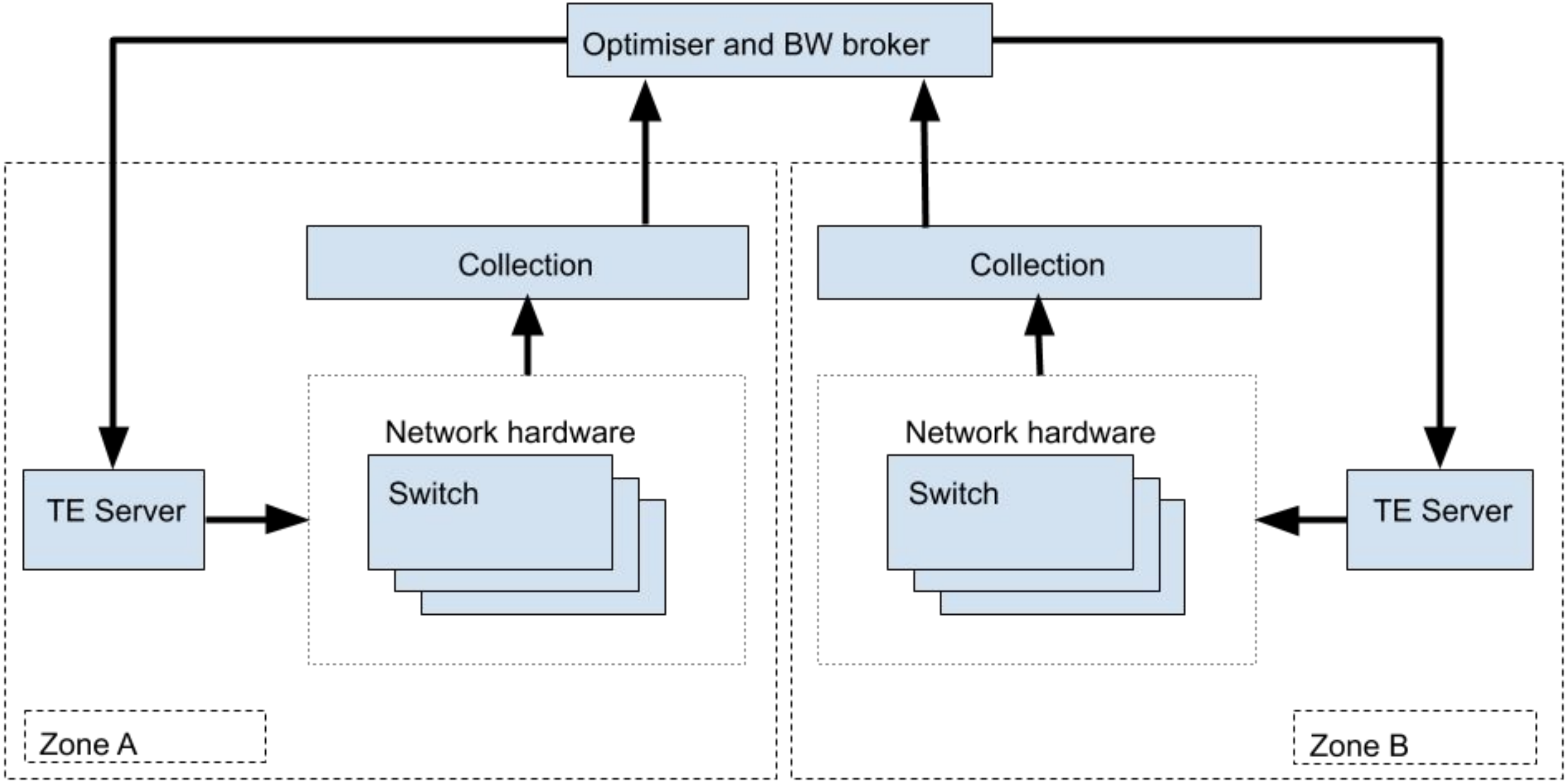



Global controller
Signal aggregator
Signal aggregator
Service pool A
Service instance
Service pool B
Service instance
Controller
Controller
Zone A
Zone B

Global DNS Loadbalancer



SDN WAN


Optimiser and BW broker
Collection
Collection
Network hardware
Switch
Network hardware
Switch
TE Server
TE Server
Zone A
Zone B

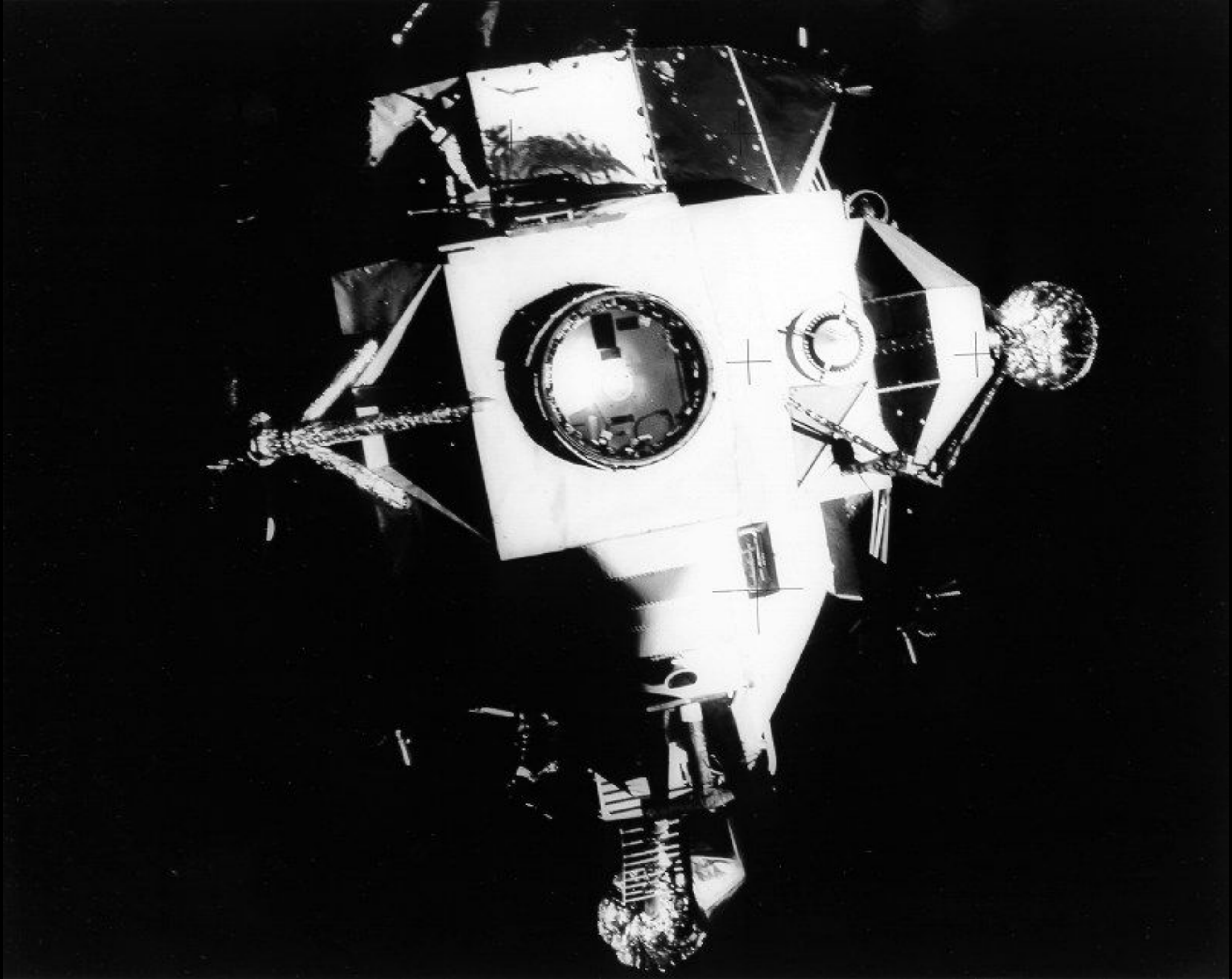
The Dynamic Control Plane: not just any old automation

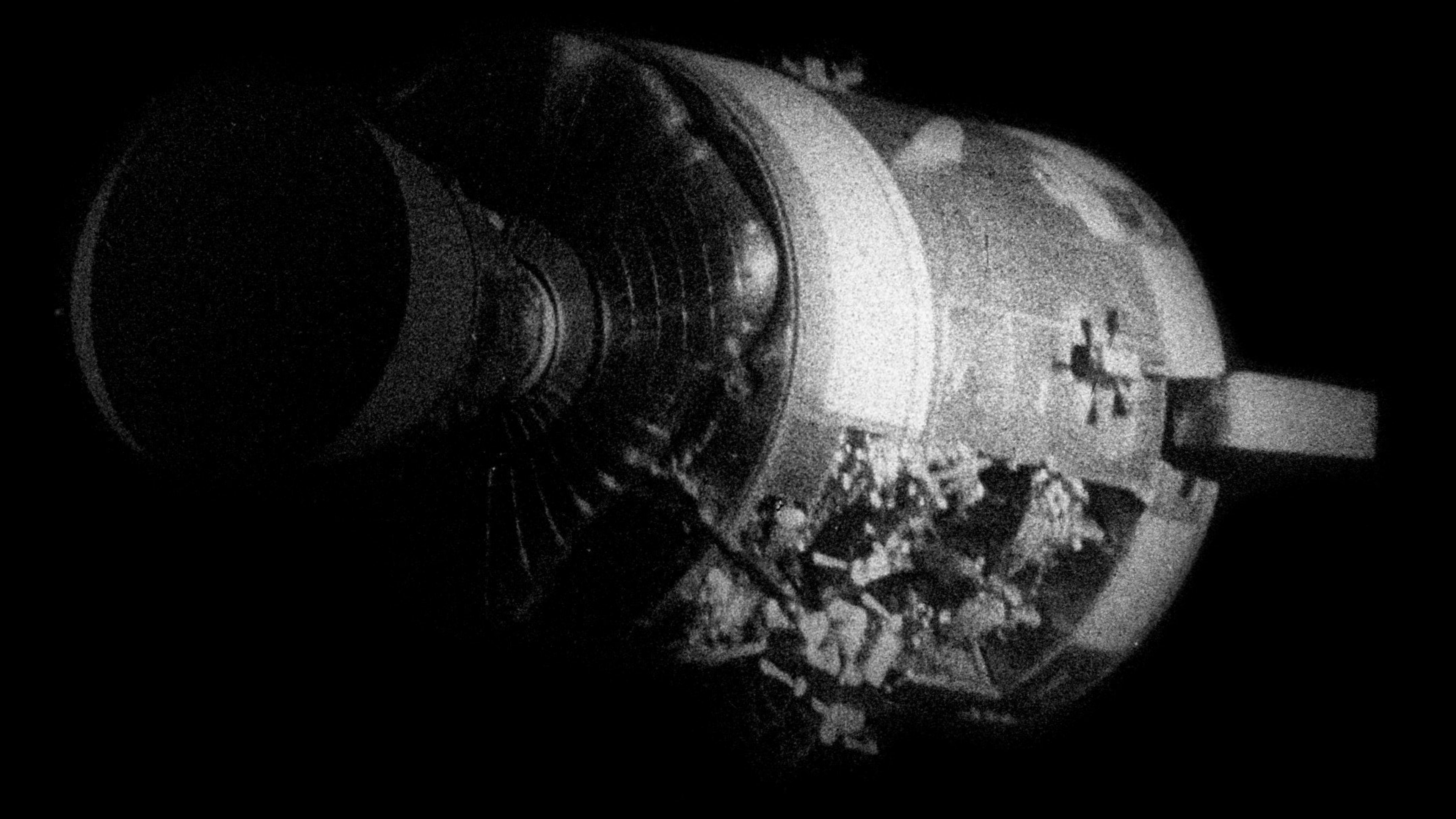
This pattern tends to arise specifically in systems that control critical parts of production and are doing zonal or global configuration, optimisation and balancing.

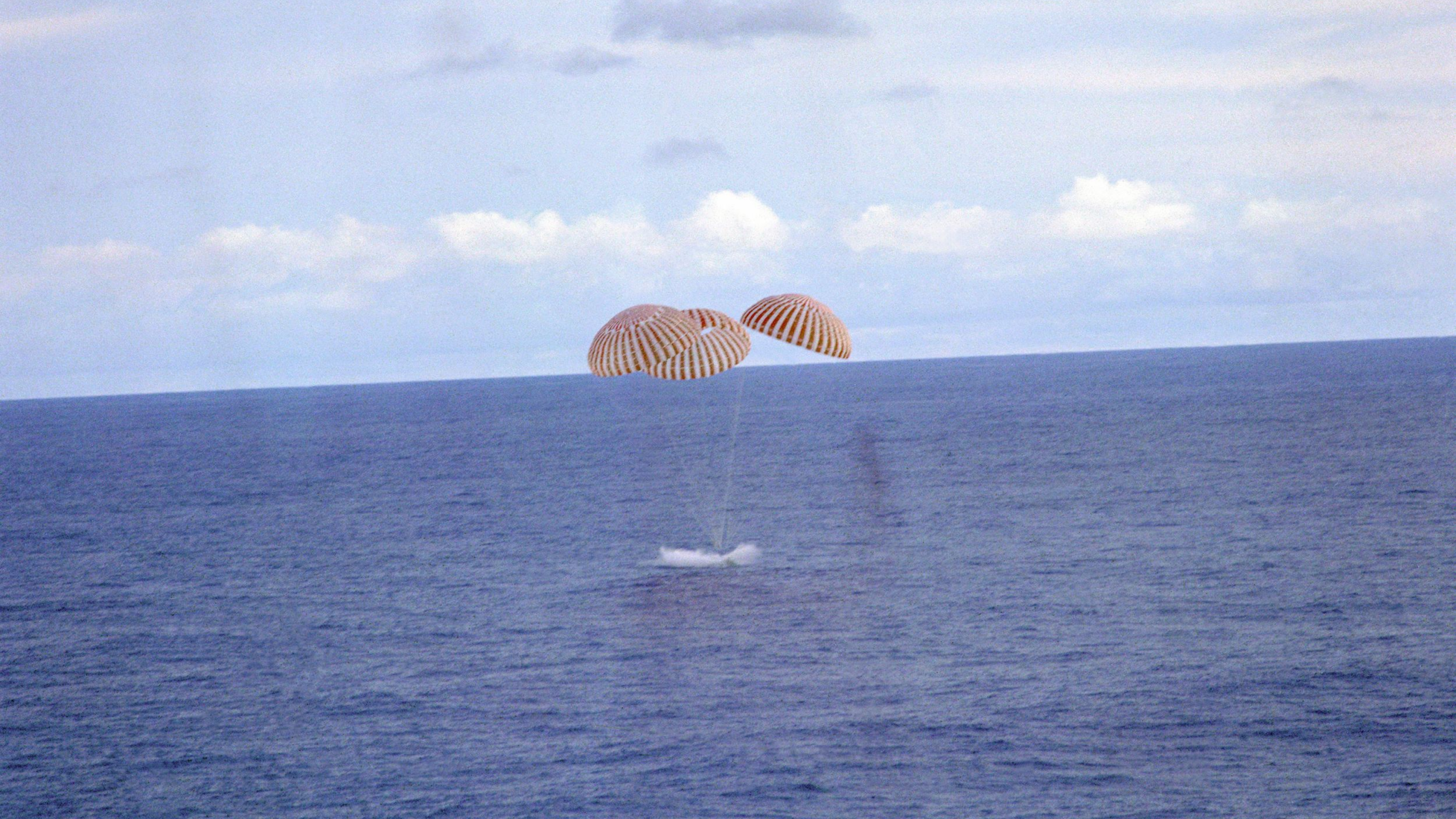
Now we are mission control.

We don’t run the systems anymore.

We build and run the systems that run the systems.







Now we are mission control.

It is much harder for us to fully understand our systems in production..

Dynamic control plane incidents.

No judgments.

December 24 2012: AWS Elastic LB

- Twas the night before Christmas, and API calls related to managing new or existing LBs started to throw mysterious errors
- Running ELBs seemed to be OK
- *“The team was puzzled as many APIs were succeeding (customers were able to create and manage new load balancers but not manage existing load balancers) and others were failing.”*

See: https://aws.amazon.com/message/680587/

December 24 2012: AWS Elastic LB

- After more than four hours, they noticed that running LBs were OK, unless someone tried to make a config update, or they scaled up or down
- Scaling workflows were disabled once they figured that out
- *“It was when the ELB technical team started digging deeply into these degraded load balancers that the team identified the missing ELB state data as the root cause of the service disruption.”*

See: https://aws.amazon.com/message/680587/

December 24 2012: AWS Elastic LB

- The ultimate fix was a data recovery process to restore the lost data and merge in changes since the data loss occurred. Full recovery from the incident took around 24 hours.
- Post incident action item was to lock down write access to the ELB control plane state.
- This incident showcases the difficulty of debugging problems in control plane software. We trust them to be stewards of critical system state and it can be very painful when that fails.

See: https://aws.amazon.com/message/680587/

Operators need mental models of both the system and the automation.

11 April 2016: GCE

- Google Compute Engine (GCE) lost external network connectivity for 18 minutes.
- An unused IP block is removed from a network configuration and the control system that propagates network configurations begins to process it.
- A race condition triggers a bug which removes all GCE IP blocks.

See: https://status.cloud.google.com/incident/compute/16007

11 April 2016: GCE

- The configuration was sent to a canary system (a second dynamic control system).
- The canary system correctly identified that there was a problem.
- But the signal that the canary system sent back to the network configuration propagation system wasn’t correctly processed.

See: https://status.cloud.google.com/incident/compute/16007

11 April 2016: GCE

- The network configuration is rolled out to other sites in turn. GCE IP blocks are advertised (over BGP) from multiple sites via IP Anycast.
- This means that probes to these IPs continued to work until the last site was withdrawn.
- The rollout process therefore lacked critical signal on the effect of its actions on the health of GCE.
- This is a classic complex systems failure involving multiple bugs and latent problems.

See: https://status.cloud.google.com/incident/compute/16007

Testing is a real challenge.

June 2, 2019: Google network outage

- Google Cloud projects running services in multiple US regions experienced elevated packet loss as a result of network congestion for a duration of up to 4 hours 25 minutes.

- Google's machines are segregated into multiple logical clusters, each with their own dedicated cluster management software.

- A maintenance event began in a single physical location and was the trigger for the outage.

See: https://status.cloud.google.com/incident/cloud-networking/19009

June 2, 2019: Google network outage

- Maintenances are common and automated.
- In the case of this specific kind of maintenance, the software control plane for the network was incorrectly configured to be turned off.
- The misconfiguration extended to the network control plane in the entire region, not just one physical location.

See: https://status.cloud.google.com/incident/cloud-networking/19009

June 2, 2019: Google network outage

- Without the control jobs, the network will ‘fail static’, meaning that it’ll continue to use its current configuration and work for a period of time.
- However after several minutes the network capacity was withdrawn.
- The incident was root-caused relatively quickly.
- However, because all instances of the network control plane had been descheduled, data had been lost and needed to be rebuilt.

See: https://status.cloud.google.com/incident/cloud-networking/19009

June 2, 2019: Google network outage

- This event required multiple misconfigurations, bugs and permissions problems in order to occur.
- It involved one dynamic control plane (the automation software) operating on at least two others (the network control plane itself and the cluster management control plane).
- Again - very hard to predict these kinds of sequences of events.
- Like the first AWS incident it illustrates the pain that data loss can cause.

See: https://status.cloud.google.com/incident/cloud-networking/19009

Blast radius may be large.

Testing failsafe/fail static behaviour is scary, and easy to neglect.

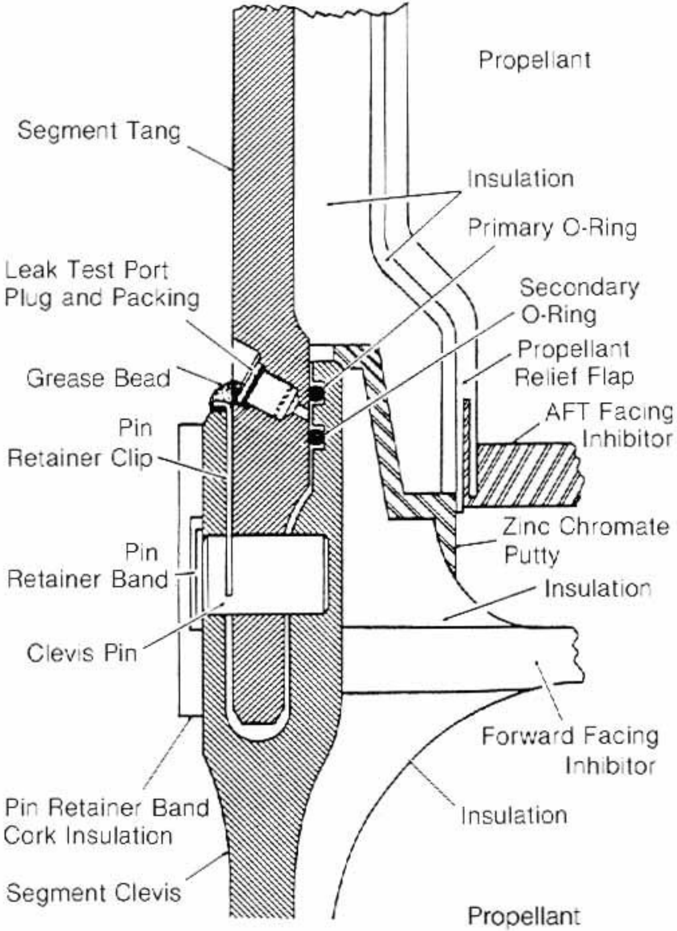


Figure 14
Solid Rocket Motor cross section shows positions of tang, clevis and O-rings. Putty lines the joint on the side toward the propellant.

What can we do?

NOAA/NASA

Use regional or zonal control systems where feasible

Test them at least as carefully as your main production systems

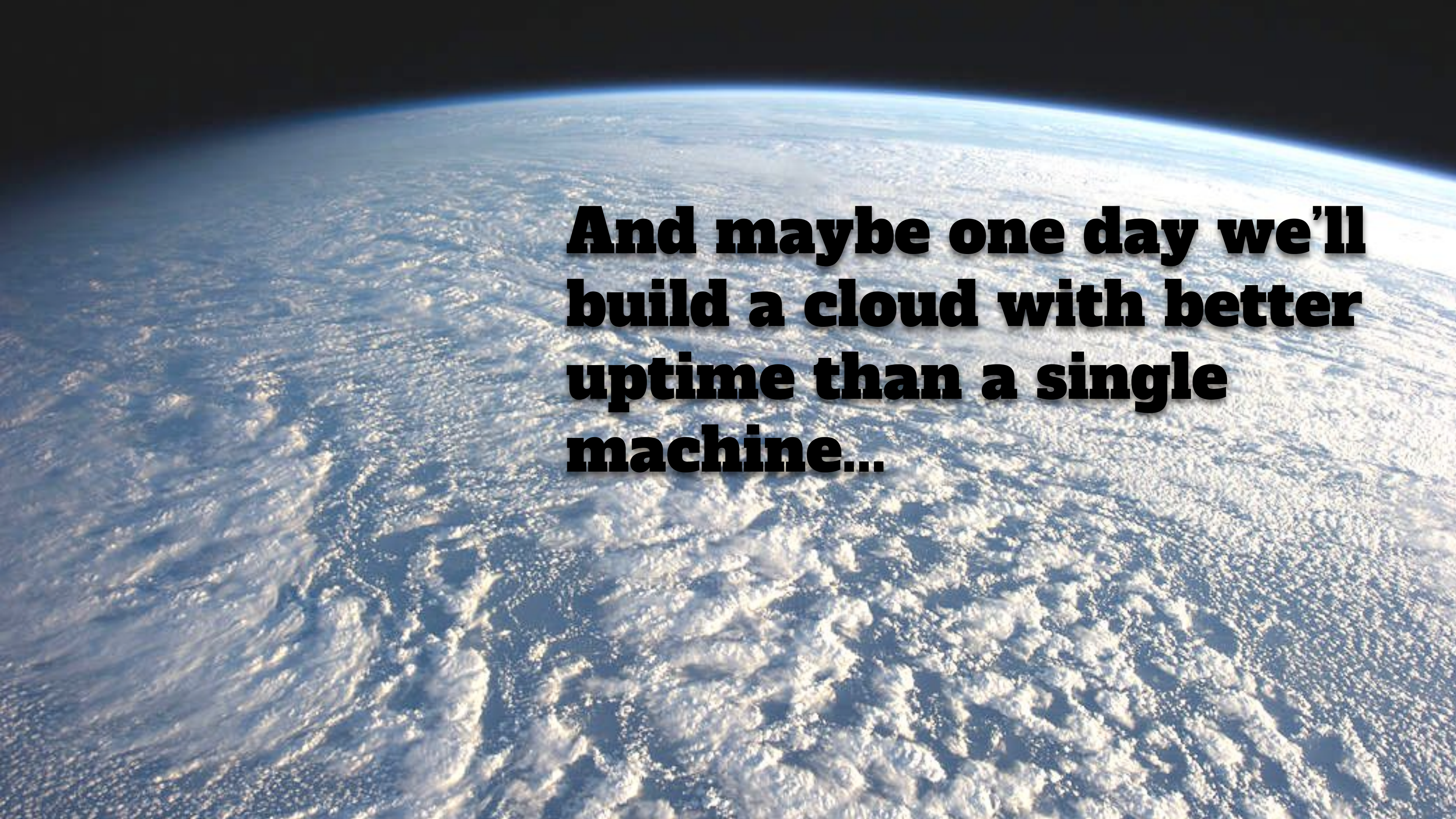
Plan for time needed for operators to stay familiar with the underlying operations.

Put guardrails around your control systems

Sometimes humans are better.

Weigh up the use of each dynamic control plane with care

Make your control systems easily observable and overridable by humans



And maybe one day we'll build a cloud with better uptime than a single machine…

Questions?

Twitter: @lauralifts