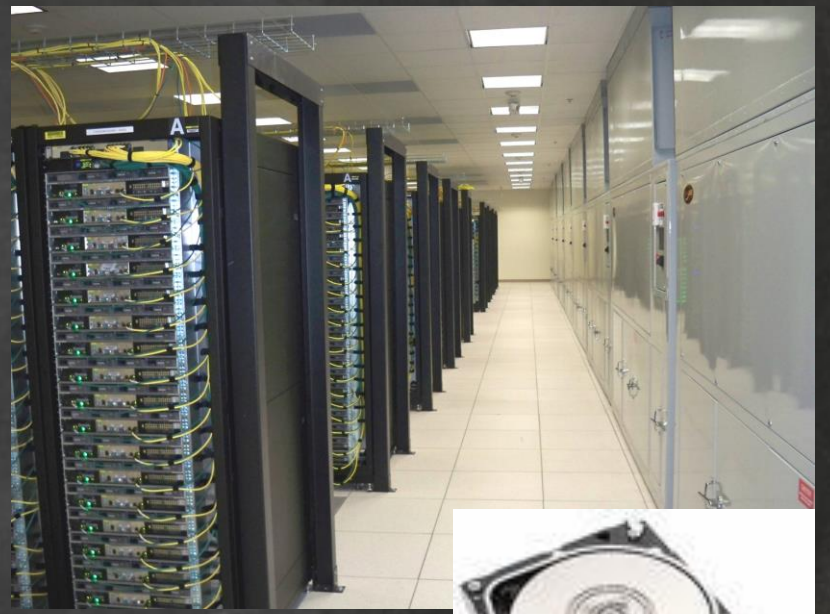
AWS re:Invent

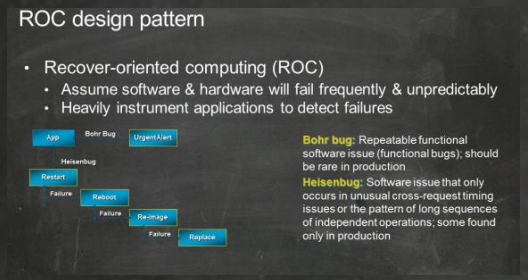
Failures at Scale & How to Ride Through Them

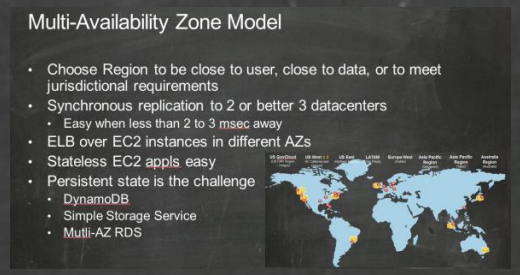
James Hamilton, VP & Distinguished Engineer

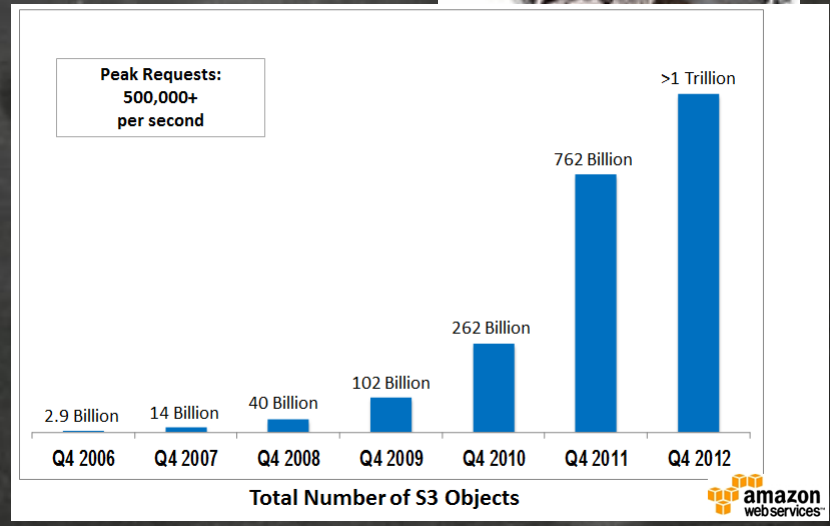
Agenda

- At scale the incredibly rare is commonplace
- Availability through application redundancy
  - Inter-region replication
  - AWS Regions & Availability Zones
  - Recovery Oriented Computing
  - Avoiding capacity meltdown
- Example rare events from industry & AWS
- Infrastructure & application lessons





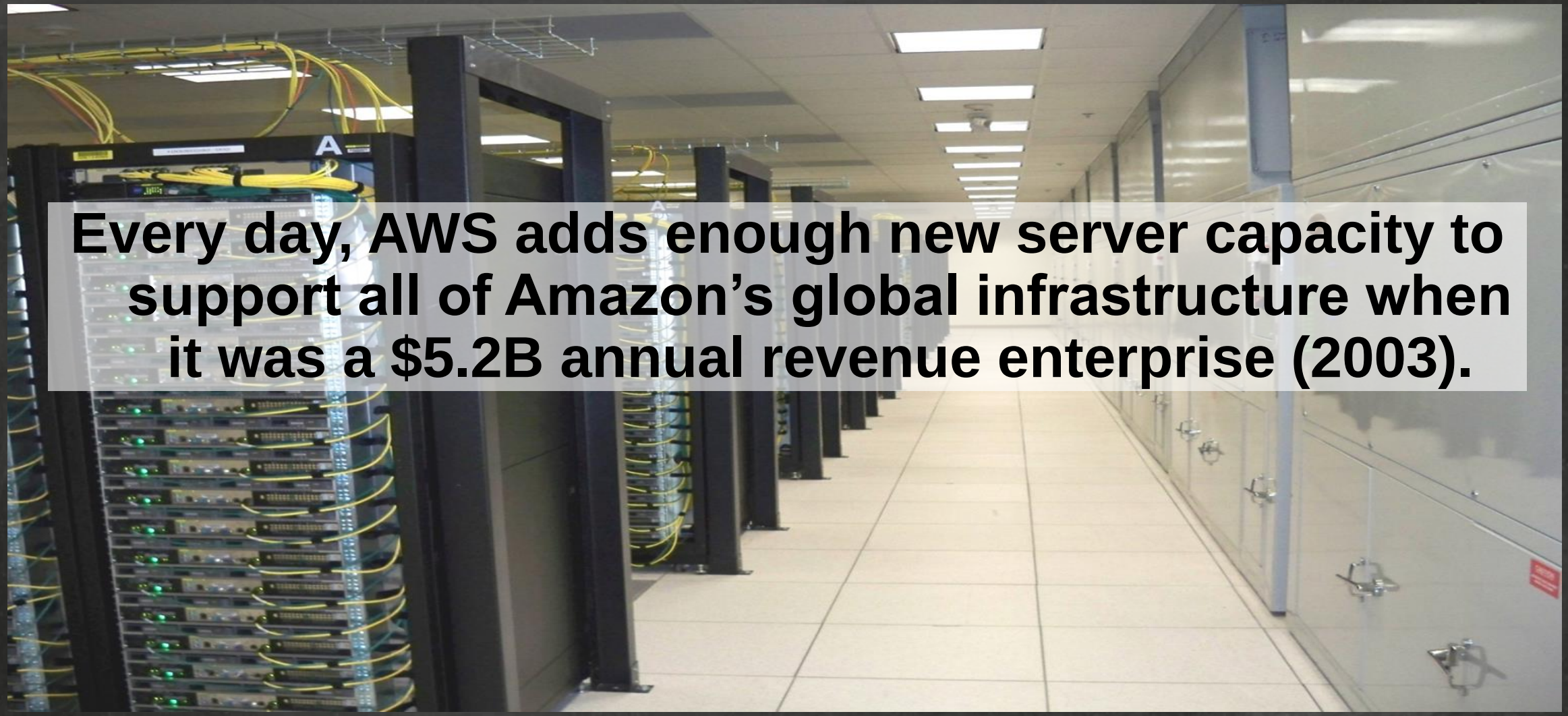




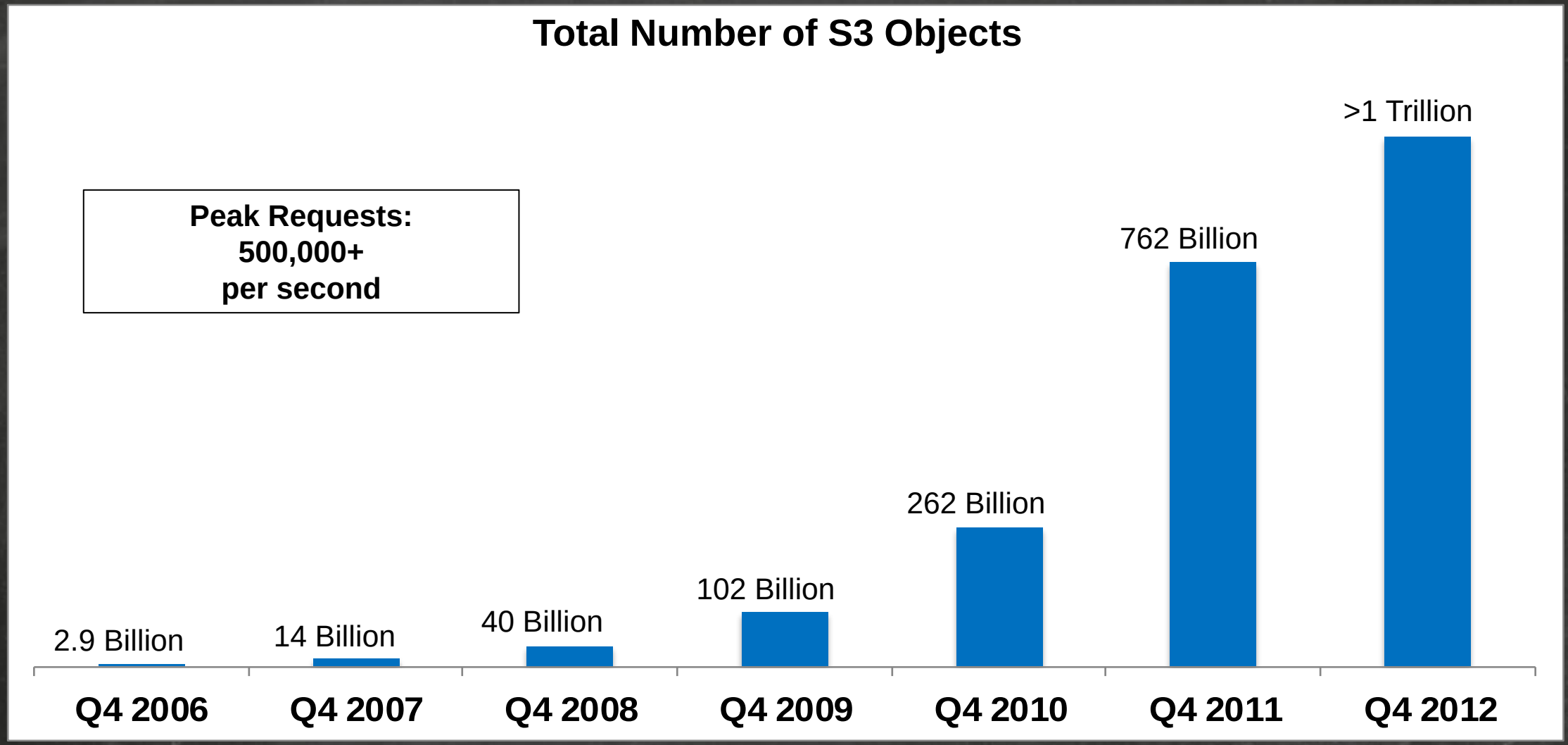
At Scale, the Incredibly Rare is Commonplace

- Server & disk failure rates:
  - Disk drives: 4% to 6% annual failure rate
  - Servers: 2% to 4% annual failure rate (AFR)
- 3% server AFR yields MTBF of 292,000 hours
  - More than 33 years
- But, at scale, in DC with 64,000 servers with 2 disks each:
  - On average, more than 5 servers & 17 disks fail each day
- Failure both inevitable & common
- Applies to all infrastructure at all levels
  - Switchgear, cooling plants, transformers, PDUs, servers, disks,…

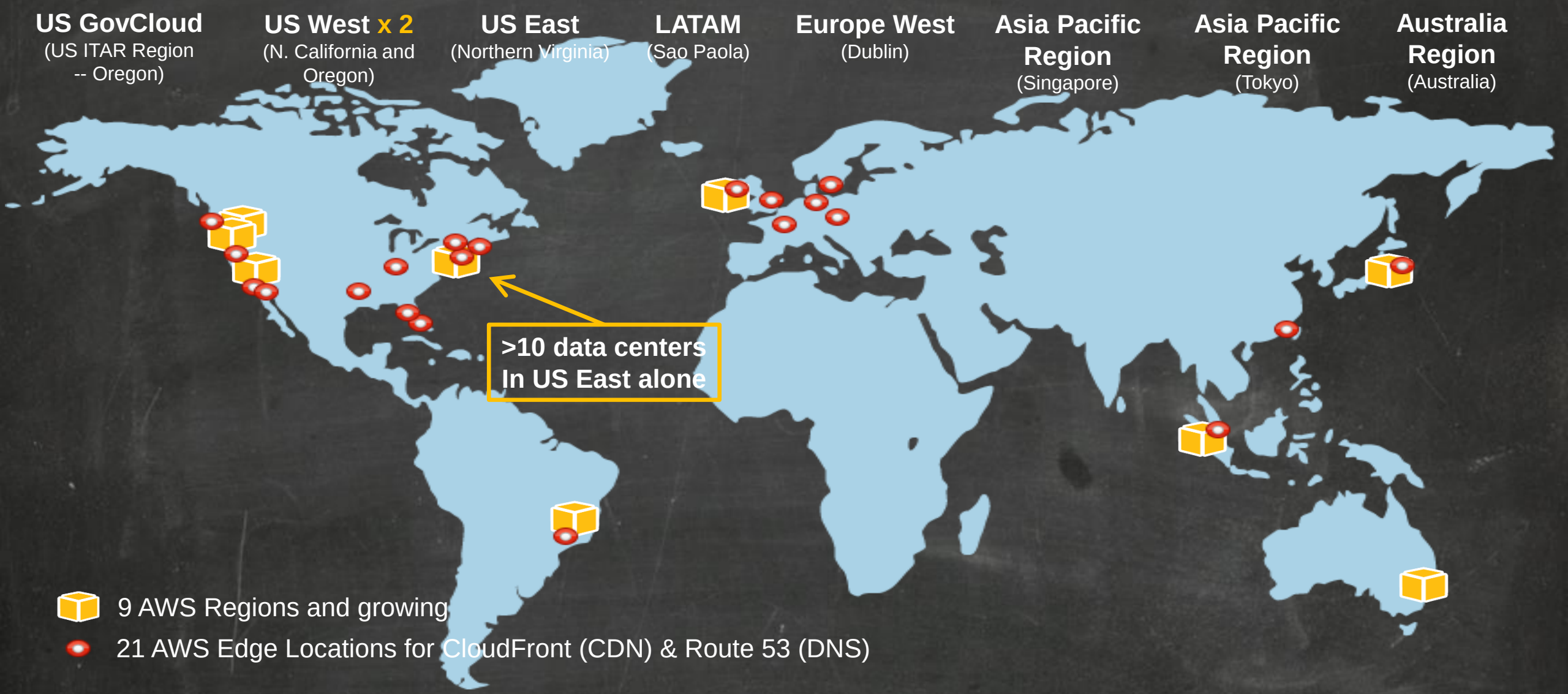
Perspective on Scaling



The Cloud Scales: Amazon S3 Growth



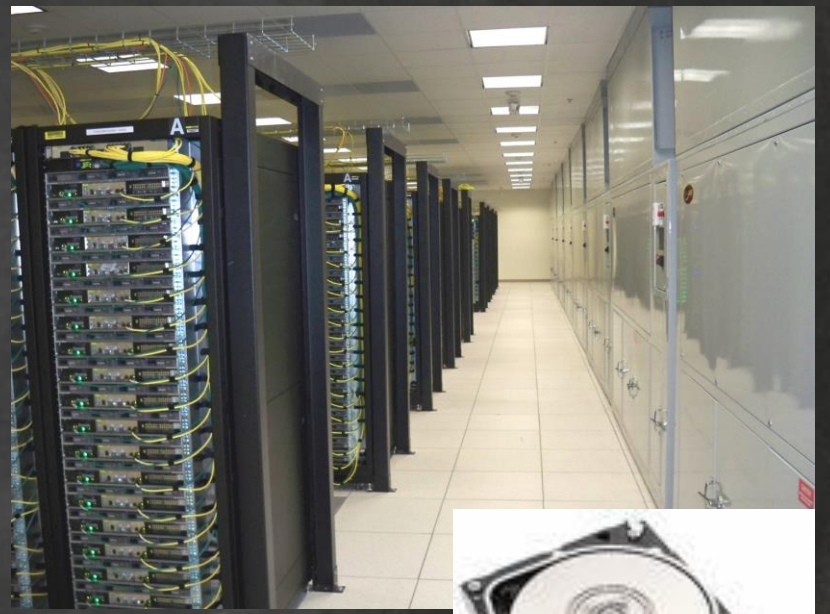
AWS Datacenters in 9 Regions

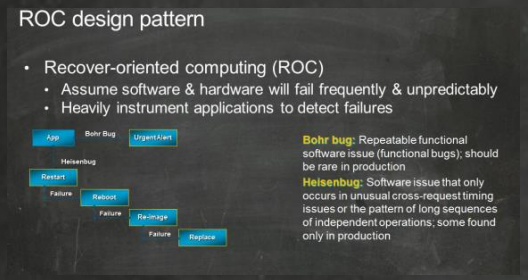


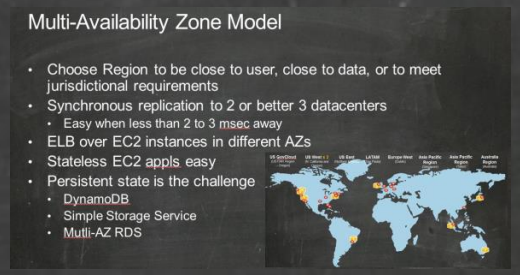
- 9 AWS Regions and growing
- 21 AWS Edge Locations for CloudFront (CDN) & Route 53 (DNS)

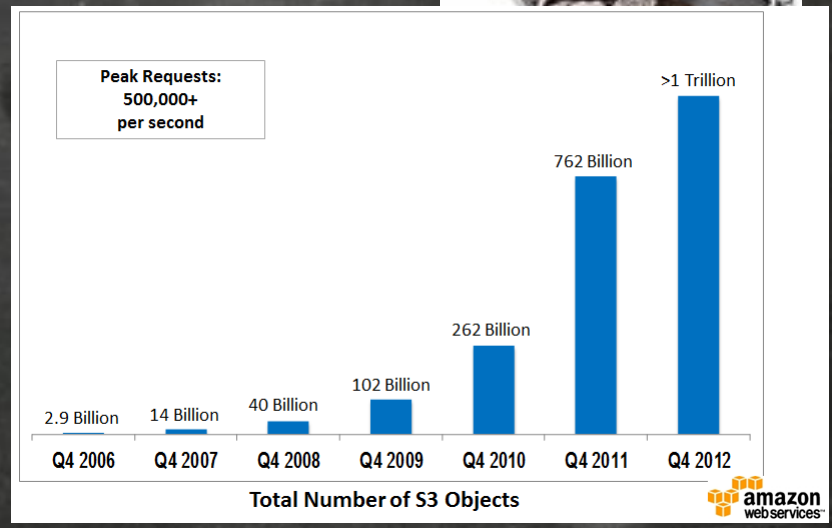
Agenda

- At scale the incredibly rare is commonplace
- Availability through application redundancy
  - Inter-region replication
  - AWS Regions & Availability Zones
  - Recovery Oriented Computing
  - Avoiding capacity meltdown
- Example rare events from industry & AWS
- Infrastructure & application lessons









Conventional Availability : Cross-Regional Replication

- 5th app availability “9” only via multi-datacenter replication
- Conventional approach:
  - Two datacenters in distant locations
  - Replicate all data to both datacenters

99.999%

- The industry-wide dominant multi-DC availability approach
  - Looks rock solid but performs remarkably poorly in practice
- Acid Test:
  - Are you willing to pull the plug on the primary server?

What is wrong with inter-regional replication?

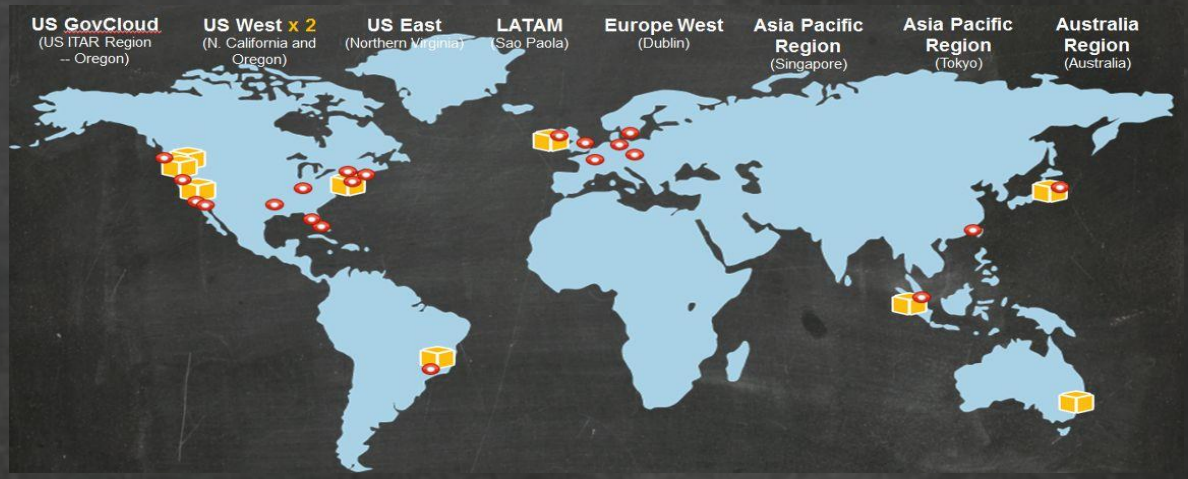
- Asynchronous replication between datacenters
  - Committing to an SSD order 1 to 2 msec
  - LA to New York 74msec round trip
  - You can't wait 74 msec to commit a transaction
- On failure, a difficult & high skill decision:
  - Fail-over & lose transactions, or
  - Don't fail-over & lose availability
- I've been in these calls in past roles
  - No win situation
  - Very hard to get right

What else is wrong with inter-regional replication?

- Fragile: Active/Passive doesn't work
  - Failover to a system that hasn't been taking operational load
  - Passive secondary not recently tested
  - Secondary config or S/W version different, incorrect load balancer config, incorrect network ACLs, latent hardware problem, router problem, resource shortage under load, …
  - Can't test without negative customer impact
  - If you don't test it, it won't work
- 2-way redundancy expensive:
  - More than ½ capacity reserved to handle failure
  - 3 datacenters much less expensive but impractical w/o high scale

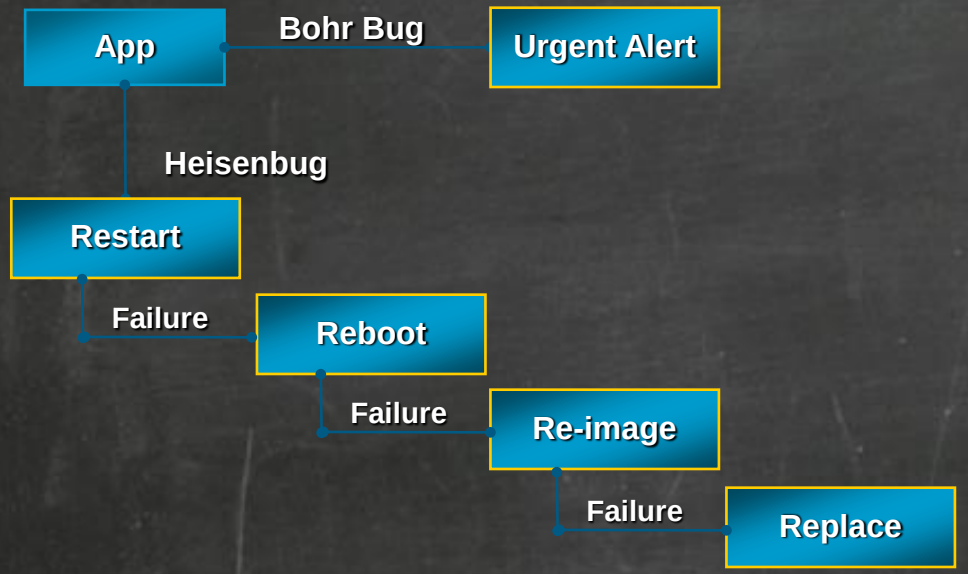
AWS Multi-Availability Zone Model

- Choose Region to be close to user, close to data, or meeting jurisdictional requirements
- Synchronous replication to 2 (or better 3) Availability Zones
  - Easy when less than 2 to 3 msec away
  - Can failover w/o customer impact
- ELB over EC2 instances in different AZs
- Stateless EC2 apps easy
- Persistent state is the challenge
  - DynamoDB
  - Simple Storage Service
  - Mutli-AZ RDS



ROC design pattern

- Recovery Oriented Computing (ROC)
  - Assume software & hardware will fail frequently & unpredictably
  - Heavily instrument applications to detect failures



Bohr bug: Repeatable functional software issue (functional bugs); should be rare in production

Heisenbug: Software issue that only occurs in unusual cross-request timing issues or the pattern of long sequences of independent operations; some found only in production

Canary in the data center

- All systems produce non-linear latencies and/or failures beyond application-specific load level
- Load limit is software release dependent
  - Changes as the application changes
- Canary in the data center
  - Route increased load to one server in the fleet
  - When starts showing non-linear delay or failure, immediately reduce load or take out of load balancer rotation
  - Result: limit is known before full fleet melts down

Graceful degradation & admission control

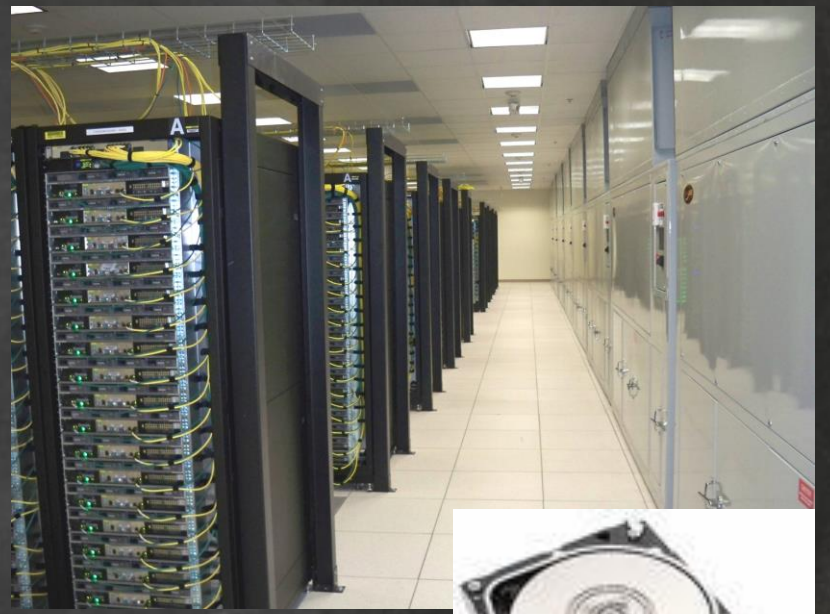
- No amount of capacity head room is sufficient
- Graceful degradation prior to admission control
  - First: shed non-critical workload
  - Then: degraded operations mode
  - Finally: admission control
- Related concept: Metered rate-of-service admission
  - Allow in small increments of users when restarting after failure
- Best practice: do not acquire new resources when failing away
  - No new EC2 instances, no new EBS volumes, ….
  - Minimize new AWS control plane resource requests
  - Run active/active & just stop using failed instances

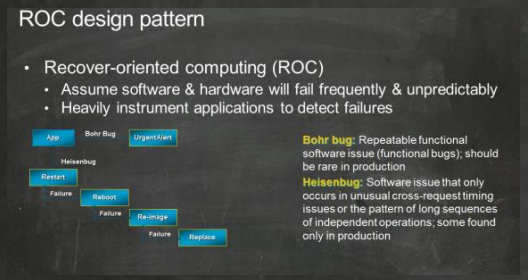
The Last Vital Step: Continuously Test in Production

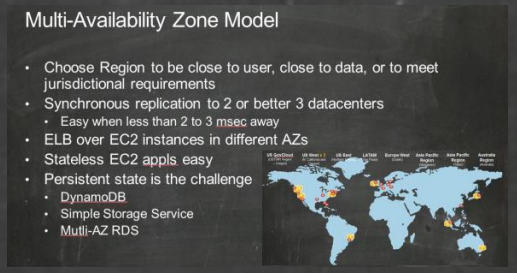
- Run active/active & test in production
  - Without constant live load, it won't work when needed
  - Test in production or it won't work when needed
- Amazon.com: Game Days
  - Disable all or part of amazon.com production capacity in entire datacenter
  - With warning & planning to avoid customer impact
- Netflix: Chaos Monkey
  - .NET Application that can be run from command line
  - Can be pointed at set of resources in a region
  - Mon to Thurs 9am to 3pm random instance kill
  - Application configuration options (including opt out)

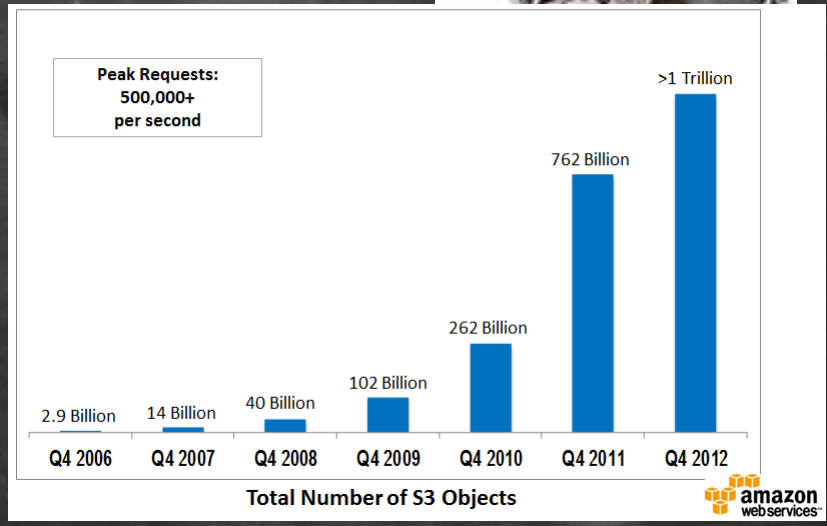
Agenda

- At scale the incredibly rare is commonplace
- Availability through application redundancy
  - Inter-region replication
  - AWS Regions & Availability Zones
  - Recovery Oriented Computing
  - Avoiding capacity meltdown
- Example rare events from industry & AWS
- Infrastructure & application lessons









1/20/2012: Power Distribution Failure “Without Warning”

Blogs

January 23, 2012 | Posted By : [redacted]

Our Friday Outage and Actions We Are Taking

Tweet 38 Share

On Friday, January 20th, we experienced a widespread outage that affected all [redacted] services. The outage started around 8:13 am Pacific Time. [redacted] services started coming back online for customer use at 3:49 pm, and all services were fully restored at 6:22 pm PST. We absolutely realize how important our services are for businesses and users who rely on us; we let you down on Friday. Please accept our humblest apologies.

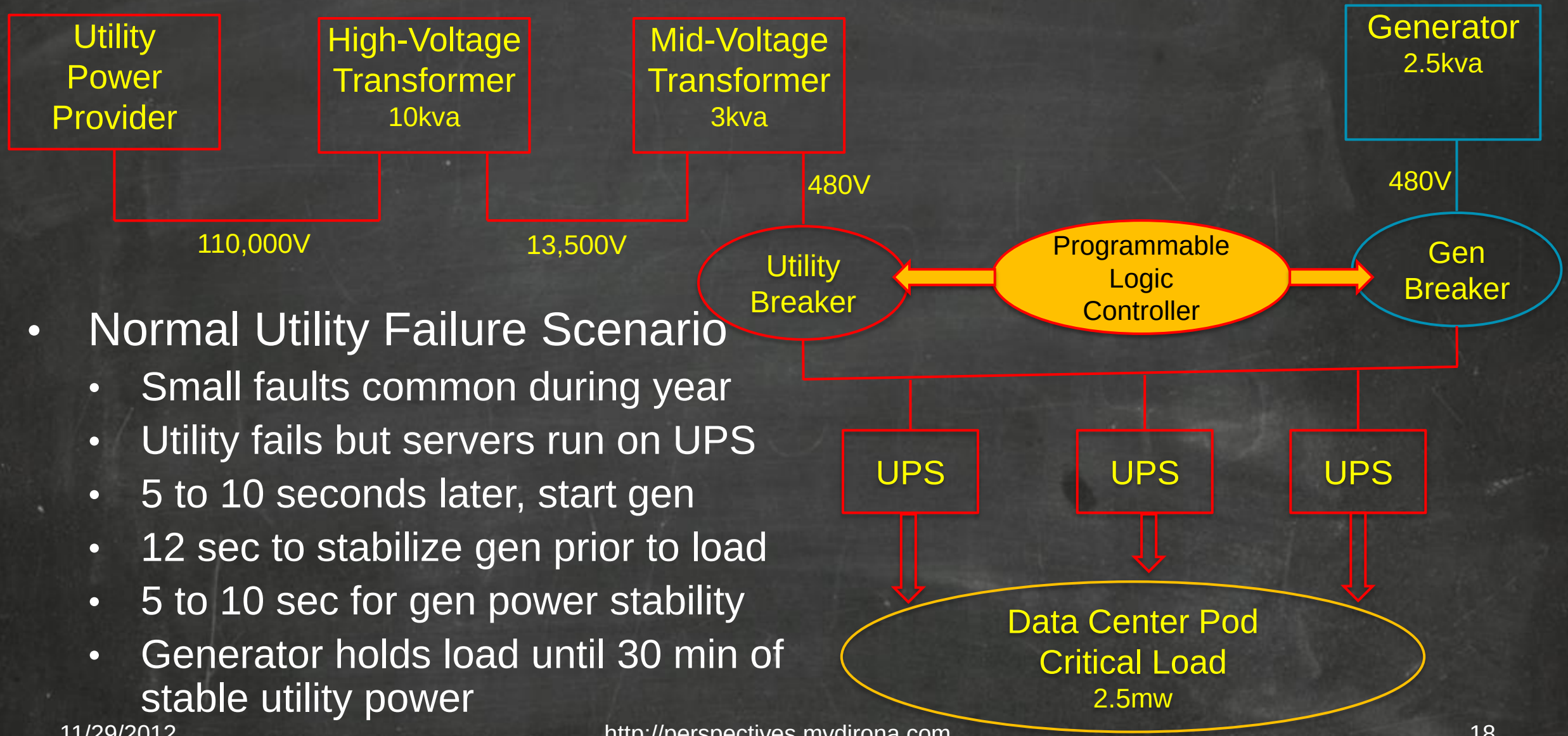
The cause of the outage was an abrupt power failure in our state-of-the-art collocated data center facility (owned and operated by [redacted] in the Silicon Valley area, California. [redacted] provides us physically secure space, highly redundant power and cooling. We get our internet connectivity from separate service providers. We own, maintain and operate the servers and the network equipment and the software. The problem was not just that the power failure happened, the problem was that it happened *abruptly*, with no warning whatsoever, and all our equipment went down all at once. Data centers, certainly this one, have triple, and even quadruple, redundancy in their power systems just to prevent such an abrupt power outage. The intent is that any power failure would have sufficient warning so that equipment, databases most importantly, can be shut down gracefully. In fact, the main function such data centers perform is to provide extreme redundancy in power systems, provide cooling for the equipment and provide physical security. Absolutely no warning happened prior to this incident, which is what we have asked our vendor to explain, and we hope they would be transparent with us. I do want to say that [redacted] has served us well, they are a leader in this field, we have never suffered an abrupt power outage like this in 5+ years. But they do owe us and other customers in that data center an explanation for what happened on Friday. They restored power quickly, but the damage was done because of the abruptness of the outage.

…cause of the outage was an abrupt power failure in our state-of-the-art data center facility … The problem was not just that the power failure happened, the problem was that it happened *abruptly*, with no warning whatsoever, and all our equipment went down all at once… Data centers, certainly this one, have triple, and even quadruple, redundancy in their power systems just to prevent such an abrupt power outage.

Observations

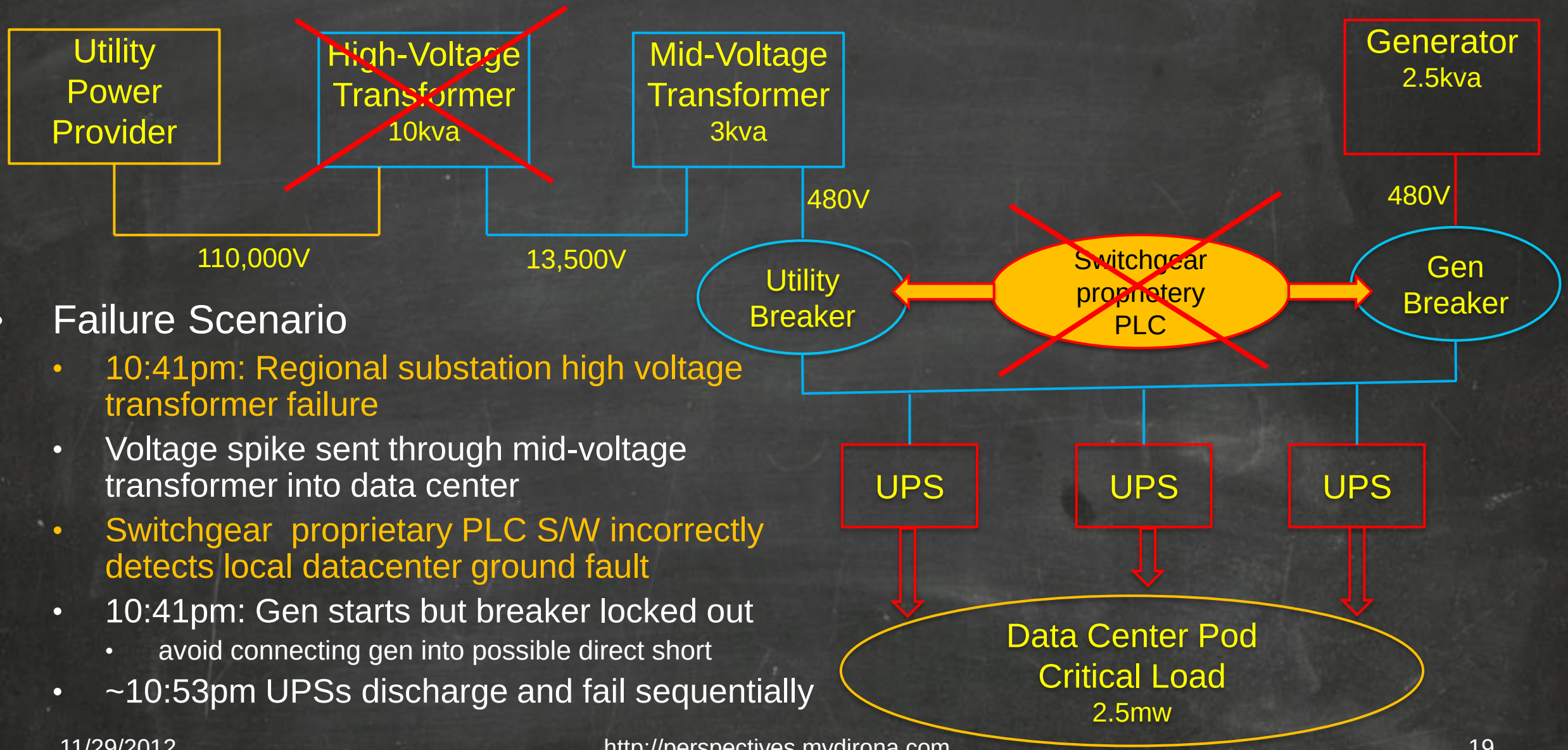
- Single datacenter availability model doesn’t work
- Costs scale with facility redundancy levels
- Decreasing or inverse payback as single facility redundancy increases

Managing Utility Faults: Correct Operation



- Normal Utility Failure Scenario
  - Small faults common during year
  - Utility fails but servers run on UPS
  - 5 to 10 seconds later, start gen
  - 12 sec to stabilize gen prior to load
  - 5 to 10 sec for gen power stability
  - Generator holds load until 30 min of stable utility power

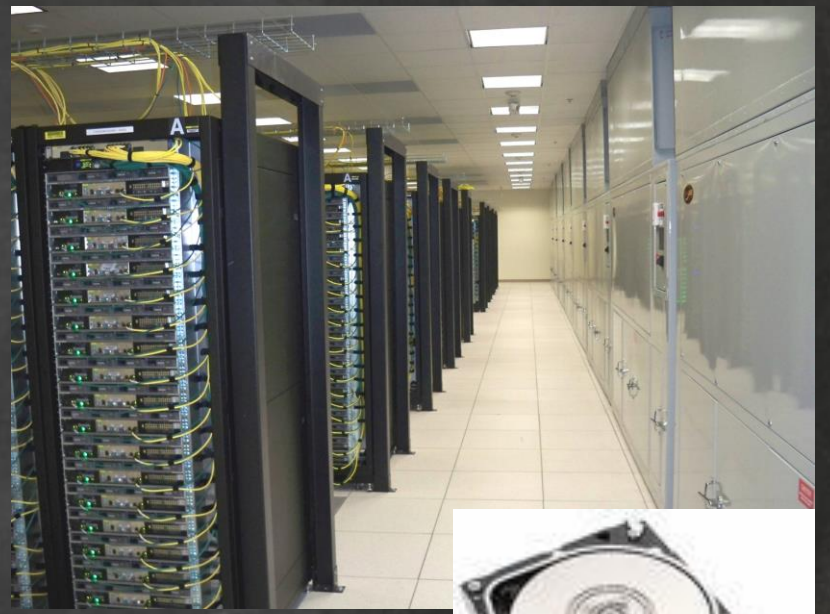
8/7/2011: Utility Switchgear “Safety” Lockout

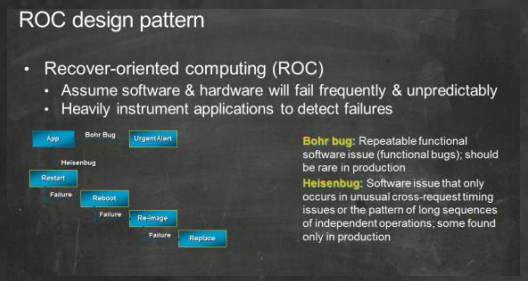


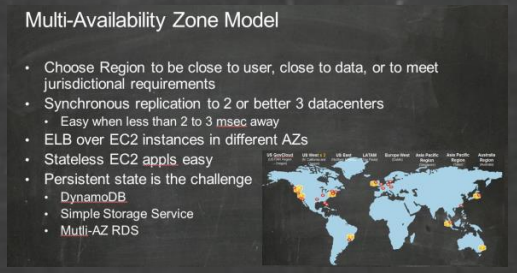
- Failure Scenario
  - 10:41pm: Regional substation high voltage transformer failure
  - Voltage spike sent through mid-voltage transformer into data center
  - Switchgear proprietary PLC S/W incorrectly detects local datacenter ground fault
  - 10:41pm: Gen starts but breaker locked out
    - avoid connecting gen into possible direct short
  - ~10:53pm UPSs discharge and fail sequentially

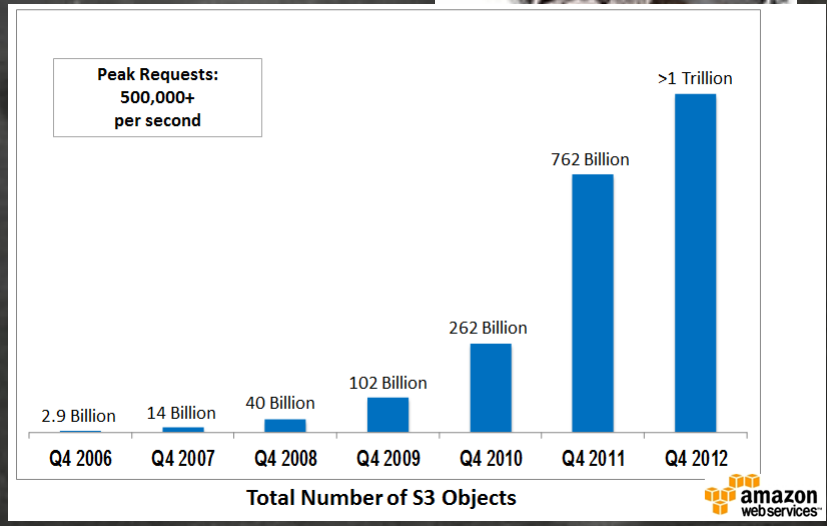
Agenda

- At scale the incredibly rare is commonplace
- Availability through application redundancy
  - Inter-region replication
  - AWS Regions & Availability Zones
  - Recovery Oriented Computing
  - Avoiding capacity meltdown
- Example rare events from industry & AWS
- Infrastructure & application lessons



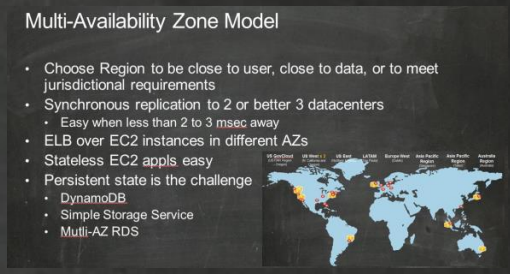






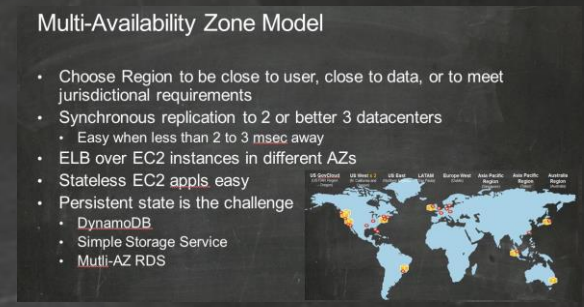
Infrastructure Observations & Updates

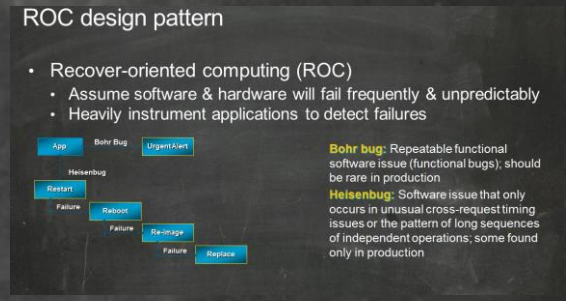
- Multi-AZ is AWS unique & a powerful app redundancy model
- Full power distribution redundancy & concurrent maintainability
  - Power redundant even during maintenance operations
- Switchgear now custom programmed to AWS specs
  - “hold the load” prioritized above capital equipment protection
- All configuration settings with maximum engineering headroom
- Network redundancy & resiliency:
  - Systematically replace 2-way redundancy with N-way (N>>2)
  - Custom monitoring to pinpoint faulty router or link fault before app impact
- Full production testing of all power distribution systems



Generalizing the Lessons for Applications

- Even incredibly rare events will happen at scale
- Multi-AZ & ROC protect against infrastructure, app, & admin issues
- Design applications using ROC principals
  - When application health is unknown, assume failure
  - Trust no single instance, server, router, or data center
  - Only use small number of simple, tested application failure paths
  - Test in production
- No new resources when failing away
- More high-scale application best practices at:
  - http://mvdirona.com/jrh/talksAndPapers/JamesRH_Lisa.pdf





Conclusion

- Black Swan events will happen at scale
- Multi-data center required for last 9
  - App & admin errors dominate infrastructure faults
- Multi-AZ redundancy will operate through unexpected failures in all levels in app & infrastructure stack
  - Multi-AZ is more reliable & easier to administer
  - Use a small number of simple app failure paths
  - Test failure paths frequently in production
- Reap reward of sleeping all night & riding through failure

We are sincerely eager to hear your **feedback** on this presentation and on re:Invent.

Please fill out an evaluation form when you have a chance.