Computer Science 425
Distributed Systems

CS 425 / ECE 428

Fall 2015

Indranil Gupta (Indy)

Dec 3, 2015

Datacenter Disasters – Four Case Studies

Lecture 28

Quick Quiz

Which of the following is the leading cause of datacenter outages?

1. **Power outage**
2. **Over-heating**
3. **Human error**
4. **Fire**
5. **DOS attacks**

Quick Quiz

Which of the following is the leading cause of datacenter outages?

1. **Power outage**
2. **Over-heating**
3. **Human error (70%)**
4. **Fire**
5. **DOS attacks**

Human Error Examples

- A State of Virginia technician pulled the wrong controller and crashed a redundant SAN that already had suffered a controller failure.
- A technician with DBS Bank made an unauthorized repair on a redundant SAN and took down both sides.
- A system operator mistakenly deleted the $38 billion Alaska Permanent Fund database and then deleted its backup.
- A maintenance contractor's mistake shut down the Oakland Air Traffic Control Center.
- Thirteen million German web sites went dark when an operator mistakenly uploaded an empty zone file.
- A test technician failed to disable a fire alarm actuator prior to testing the fire suppression system.
- Siren noise damaged several disks, including the virtual backup disks.
  - https://www.youtube.com/watch?v=tDacjrSCeq4
- A system administrator closed all applications on one server in an active/active pair to upgrade it and then shut down the operating server.
- (hosting.com) Incorrect breaker operation sequence executed by servicing vendor caused a shutdown of UPS and offline time to websites of 1-5 hours

Source:
http://www.availabilitydigest.com/public_articles/0704/data_center_outages-lessons.pdf

Why Study Outages?

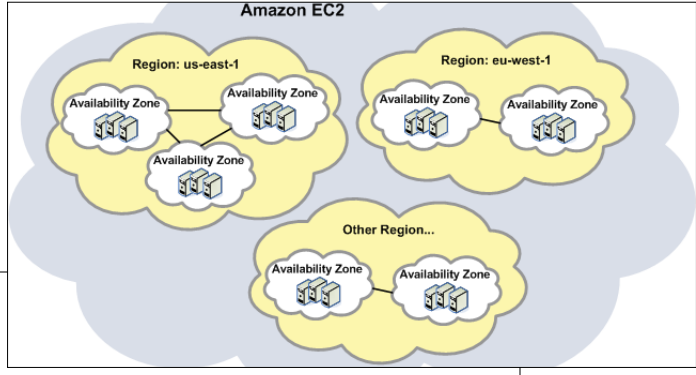
- They're fun! (Schadenfreude!)
- But really – so that we can learn lessons
- Learn more about the actual behavior of systems in the real world
- Design better systems in the future

Case Study 1: AWS – Apr 21 2011

- **History**
  - Several companies using AWS EC2 went down – e.g., Reddit, FourSquare
  - AWS dashboard showed problems with EC2, and other storage
  - Lasted 3.5 days (at least)
  - Led to some data loss
  - Amazon released post-mortem analysis

Source: http://aws.amazon.com/message/65648/

AWS – Apr 21 2011



Background:

- AWS Regions: Separate from each other
  - Consist of availability zones: can have automatic data replication across zones within a region
- AWS Elastic Block Storage (EBS) – mountable storage "devices", accessible from EC2 instances
- 1 EBS volume runs inside an Availability Zone
  - Two networks: primary n/w used for EC2 and EBS control plane; secondary n/w used for overflow – has lower capacity
  - Control information replicated across zones (for availability)
- EBS volumes replicated for durability
  - Each volume has a primary replica
  - If out of sync or node failure, aggressive re-mirroring of data

Internal Timeline

- ***12.47 AM: Routine primary n/w capacity upgrade in an av. zone in US East Region***
- **Traffic shifted off several primary n/w routers to other primary n/w routers**
  - **Critical Error: someone shifted traffic for one such router to a secondary n/w router**
- **=> Several EBS volumes now had no/bad primary n/w**
  - **Primary n/w disconnected**
  - **Second n/w has low capacity and thus overwhelmed**
  - **Many primary replicas had no backup**
- **Team discovered critical error and rolled it back**

(Is it over yet?)

Internal Timeline (2)

- **Team discovered critical error and rolled it back**
  - **Due to network partitioning, many primary replicas thought they had no backup: these automatically, started re-mirroring aggressively**
  - ***All at once*: free n/w cap quickly used, replicas stuck in loop**
  - **Re-mirroring storm: 13% of EBS volumes**
- **N/w unavailable for Control Plane**
  - **Unable to serve "create volume" API requests**
  - **Control plane ops have long time-out; began to back up**
  - **When thread pool filled up, control plane started to reject create volume requests**
- ***2.40 AM: Team disabled all such new requests***
- ***2.50 AM: all error rates and latencies for EBS APIs recover***

(Is it over yet?)

Internal Timeline (3)

- **Two issues made things worse**
  - **Primaries searching for potential replicas did not back off**
  - **Race condition in EBS code that was triggered by high request rates: caused node failure**
- ***5.30 AM: Error rates and latencies increase again***
- **Re-mirroring is negotiation b/w EC2 node, EBS node, and EBS control plane (to ensure 1 primary)**
  - **Due to race condition, EBS nodes started to fail**
  - **Rate of negotiations increased**
  - **Caused more node failures (via race), and rinse-n-repeat**
  - **"Brown-out" of EBS API functionalities**
- ***8.20 AM: Team starts disabling all communication b/w EBS cluster in affected av. zone and EBS control plane***
  - **Av. zone still down, but control plane recovering slowly**

Internal Timeline (4)

- ***11.30 am: Team figures out how to prevent EBS servers in av. zone from futile re-mirroring***
  - **Affected av. zone slowly recovers**
- **Customers still continued to face high error rates for new EBS-backed EC2 instances until noon**
  - **Another new EBS control plane API had recently been launched (for attaching new EC2 instances to volumes)**
  - **Its error rates were being shadowed by new errors**
- **Noon: No more volumes getting stuck**
- **But 13% volumes still in stuck state**

Internal Timeline (5)

- **Long tail of recovery**
  - **Read more on the post-mortem to find out how team addressed this**
  - **By noon April 24th, all but 1.04 % of volumes had been restored**
  - **Eventually, 0.07% volumes could not be recovered**
- **This outage also affected relational database service (RDS) that were single – av. zone.**

Lessons

Generic: large outages/failures

- **Often start from human error**
- **But balloon due to cascading sub-failures**

Specific to this Outage:

- **Audit n/w configuration change processes**
- **Higher capacity in secondary n/w**
- **Prevent re-mirroring storm: backing off rather than aggressively retry**
- **Fixing race condition**
- **Users who wrote code to take advantage of multiple av. zones within region not affected**
- **Better tools for communication, health (AWS Dashboard), service credit for customers (10 day credit)**

Case Study 2: Facebook Outage Sep 23, 2010

- Unreachable for 2.5 hours (worst in past 4 years)
- Background
  - Data stored in a persistent store, and cache
  - Includes configuration data
  - FB has automated system for verifying configuration values in the cache, and replace invalid values with updated values from the store
- Source: https://www.facebook.com/notes/facebook-engineering/more-details-on-todays-outage/431441338919

Timeline

- On Sep 23, FB made a change to the persistent copy of a configuration
  - that was invalid
- All clients (FB servers) saw invalid value
  - All attempted to fix it
  - All queried cluster of databases
  - Databases overwhelmed quickly by 100K's qps
- Team fixed the invalid configuration

(Is it over yet?)

Timeline

- When client received error from DB, it interpreted it as invalid and deleted cache entry
  - When DB failed to respond => created more queries
  - No back off
  - Rinse-n-repeat
  - (Cascading failures)
- FB's Solution
  - Turn off entire FB website
  - Stopped all traffic to DB cluster
  - DB recovers
  - Slowly allow users back on: allowed clients to slowly update caches
  - Took until later in day for entire site to be back up
- Lessons
  - New configuration system design
  - Back off

Case Study 3: Explosion at The Planet – May 31, 2008

- **Not Star Wars**
- **The Planet – 4th largest web hosting company, supported 22K websites**
  - **6 datacenters: Houston (2), Dallas (4)**
- **Took down 9K servers and 7.5K businesses**
- **Source: http://www.availabilitydigest.com/public_articles/0309/planet_explosion.pdf**

Timeline

- 5.55 pm: Explosion in H1 Houston DC
  - Short circuit in transformer set it on fire
  - Caused an explosion of battery-acid fumes from UPS backup
  - (Cascading failures)
  - Blew out 3 walls of first floor
- No servers were damaged, but 9K servers brought down
- Fire department evacuated building
  - Directed that backup generators could not be turned on
  - Due to fire hazard, no staff allowed back in until 10 pm
- The Planet staff had to physically ship some critical servers to their other DCs (on pickups)
  - But limited power and cooling at other DCs

Timeline (2)

- 5 pm Jun 2: Power restored to second floor
- Jun 4: First floor servers were being restored one rack at a time
- Frequent updates to customers (15 min to 1 hour)

- Lessons
  - Backup across DCs, perhaps across different providers
    - » Whose responsibility would this be?
    - » Provider?
    - » Customer? More difficult due to extra work and data lock-in across providers.
  - May cost customers more (but we pay insurance don't we?)

Case Study 4: Rackspace Outage – Nov 11, 2007

- Provides hosting for 1000s of websites
- Has several DCs
- Claimed zero downtime until this outage
- 4 am
  - Mechanical failure caused outage in Dallas DC
  - Service restored by crack team

(Is it over yet?)

True Story

- 6.30 pm Nov 12
  - On the road outside the DC, a trucker passed out and his truck rammed into main transformer of DC
  - Transformer exploded
- Emergency generator kicked in
- Ops switched to secondary power utility line
- 15 mins later, emergency personnel trying to extricate driver shut down secondary power source (for safety of all, including first responders)
- Things happening too fast for any coordination
- Emergency generator restarted

(Is it over yet?)

Timeline (2)

- Each power interruption caused AC chillers to recycle
  - Takes them 30 mins (one reset factored into DC design)
  - Chillers out for 15 mins after first power outage
  - Emergency restart reset chillers again (total time: 45 mins > threshold for which DC was built)
- Temperatures rose quickly
  - Servers would have overheated and melted hardware
- Team decided to shut down entire DC
- After power and cooling were restored, most of the sites were up by Nov 13

Lessons

- Keeping customers updated helps, e.g., via dashboards
- Data redundancy across DCs (Business Continuity plans)

Many other disasters

Not all companies as open as those discussed

- **RIM Apr 2007 – day-long outage; no details**
- **Hostway Jul 2007 – informed customers that it would move its DC Miami → Tampa, and that outage would be 12 hours**
  - **Outage was 3-7 days**

Overall Lessons Learnt

- **Datacenter fault-tolerance akin to diseases/ medicine today**
  - **Most common illnesses (crash failures) addressed**
  - **Uncommon cases can be horrible**
- **Testing is important**
  - **American Eagle, during a disaster discovered that they could not fail over to backup DC**
- **Failed upgrades common cause**
  - **Fallback plan**
  - **IRS decided to decommission their hardware before starting up new hardware: no audits next year ($300 M loss to US govt.)**
- **Source: http://www.availabilitydigest.com/public_articles/0704/data_center_outages-lessons.pdf**

Overall Lessons Learnt

- Data availability and recovery
  - Cross-DC replication, either by provider or by customer
- Consistent Documentation
  - A Google AppEngine outage prolonged because ops didn't know which version of docs to use for recovery
- Outages always a cascading series of failures
  - Need more ways to break the chain avoid outages
- DOS-resistance
- Internet outages
  - Under-sea cable cut
  - DNS failures
  - Government turning "Internet off" (mostly DNS)
- Planning: many failures are unexpected
- There are also planned outages, and they need to be planned well

Announcements

- **Final exam time and date posted on website**
  - **See Course schedule**
  - **2 separate rooms: Please go to your classroom according to your last name**
  - **1 page (2 sided) cheatsheet allowed: need to submit it along with exam**
- **Next Tuesday's lecture: mandatory**
  - **We will review how far we've come since the beginning of the semester**