What breaks our systems:

A taxonomy of black swans
Laura Nolan

Contributor to the Site Reliability Engineering book and to Seeking SRE. Brand new Production Engineer@Slack.

Member of the International Committee for Robot Arms Control (ICRAC) and the Campaign to Stop Killer Robots.

@lauralifts on Twitter (DMs open) or find me on the SREcon Slack
What is a Black Swan?

- Outlier event
- Hard to predict
- Severe in impact
Every black swan is unique

But there are patterns, and sometimes we can use those to create defences
Black swans can become routine non-incidents

Example: the class of incidents caused by change can be mostly defeated with canarying
On sharing postmortems
Some subspecies of black swan

Hitting limits  Spreading slowness  Thundering herds

Automation interactions  Cyberattacks  Dependency problems
1. Hitting Limits
Instapaper,
February 2017

- Prod DB on Amazon MySQL RDS
- Hit a 2TB limit because filesystem ext3 - nobody knew this would happen
- Had to dump data and import into a DB backed by ext4
- Down for over a day, limited for 5 days

[Link to incident report]
Sentry, July 2015

- Down for most of the US working day
- Maxed out Postgres transaction IDs, fixing this with vacuum process
- Had to truncate a DB table to get back up and running

Link to incident report
SparkPost May 2017

- Unable to send mail for multiple hours
- High DNS workload
- Recently expanded their cluster
- Hit undocumented per-cluster AWS connection limits

[Link to incident report]
Foursquare, October 2010

- Total site outage for 11 hours
- One of several MongoDB shards outgrew its RAM, hitting a performance cliff
- Backlog of queries
- Resharding while at full capacity is hard

[Link to incident report]
EU region down for 4 hours
Orchestration software wouldn’t start
Library problem: queried all Zookeeper nodes via pipe with 64K buffer
Buffer filled, exception, fail

Link to incident report
Hitting Limits

- Limits problems can strike in many ways
- System resources like RAM, logical resources like buffer sizes and IDs, limits imposed by providers and many others
Defence: load and capacity testing

- Including cloud services (warn your provider first)
- Include write loads
  - Use a replica of prod
  - Grow past your current size
- Don’t forget ancillary datastores
- Also test startup and any other operations (backups, resharding etc) with larger sized datasets
Defence: monitoring

- The best documentation of known limits is a monitoring alert
- Include a link that explains the nature of the limit and what to do about it
- The more involved the response, the more lead time responders will need
- Lines on your monitoring graphs that show limits are really useful
2. Spreading Slowness
AWS problems, HostedGraphite goes down

BUT! They’re not on AWS

Their LB connections were being saturated due to slow connections coming from customers inside AWS

[Link to incident report]
Spotify, April 2013

- Playlist service overloaded because another service started using it
- Rolled that back, but huge outgoing request queues and verbose logging broke a critical service
- Needed to be restarted behind firewall to recover

[Link to incident report]
Auth system slowed to a crawl
- Cause: Redis had gotten overloaded
- Clients were retrying Redis transactions up to 500 times with no backoff
Defence: fail fast

- Failing fast is better than slow
- Enforce deadlines for all requests - in and out
- Limit retries, exponential backoff and jitter
- Consider circuit breaker pattern
  - Limits retries from a client, sharing state across multiple requests
Defence: USE dashboards

- Utilisation, saturation, errors
  - Utilisation: average time working
  - Saturation: degree of queueing
  - Errors: count of events
- Quick way to identify bottlenecks
- Consider physical resources and also software resources - connections, threads, locks, file descriptors etc
3. Thundering Herds
The world is much more correlated than we give credit to. And so we see more of what Nassim Taleb calls "black swan events" - rare events happen more often than they should because the world is more correlated.”

-- Richard Thaler
Where does coordinated demand come from?

- Can arise from users
- Very often from systems
  - Cron jobs at midnight
  - Mobile clients all updating at a specific time
  - Large batch jobs starting
Two separate incidents caused significant numbers of users to be disconnected:

- WebSockets based API - long running sessions
- Simultaneous reconnect caused saturation in their databases
GitHub was down for a while
When it came back traffic surged
Requests are queued into their DB
  • Complex scheduling logic
Load resulted in huge DB contention

CircleCI, July 2015
Defence: plan and test

- Almost any Internet facing service can potentially face a thundering herd
- Explicitly plan for this
  - Degraded modes
  - What requests can be dropped?
  - Queuing input that can be processed asynchronously
- Test and iterate
4. Automation interactions
Google erases its CDN

- Engineer tries to send 1 rack of machines to disk erase process
- Accidentally the entire Google CDN
- Slower queries and network congestion for 2 days until system restored

[Link to incident report]
Performing a Zookeeper migration

Turned off their autoscaler so it wouldn't read from Zookeeper during migration process

Automation turns Autoscaler back on

Autoscaler gets confused and turns off most of the site

Link to incident report
Complex systems are inherently hazardous systems.

-- Richard Cook, MD
Defence: control

- Create a constraints service to limit automation operations
  - Example: limit how many operations per unit time
  - Example: set lower bounds for remaining resources
  - Example: don’t reduce capacity when a service has received alerts/isn’t in SLO
- Provide easy ways to disable automation - and use them
- All automation should log to one searchable place
5. Cyberattacks
Maersk, June 2017

- Infected by NotPetya malware - one of their office machines ran vulnerable accounting software
- Maersk turned off its entire global network
- They couldn't unload ships, take bookings for days - 20% hit to global shipping
- Cost billions overall

[Link to incident report]
Defence: smaller blast radius

- Separate prod from non-prod as much as possible
- Break production systems into multiple zones, limit and control communication between them
- Validate and control what runs in production
- Minimize worst possible blast radius for incidents
6. Dependency problems
Dependency loops

- Can you start up your entire service from scratch, with none of your infrastructure running?
- Simultaneous reboots happen.
- This is a bad time to notice that your storage infra depends on your monitoring to start, which depends on your storage being up...
Github, January 2018

- 2 hour outage
- Power disruption led to 25% of their main DC rebooting
- Some machines didn’t come back
- Redis clusters unhealthy
- Main application backends wouldn’t start due to unintentional hard Redis dependency

[Link to incident report]
Trello, March 2017

- AWS S3 outage brought down their frontend webapp
- Trello API should have been fine but wasn’t
  - It was checking for the web client being up, even though it didn’t otherwise depend on it

[Link to incident report]
Defence: layer and test

- Layer your infrastructure
  - Only allow each service to have dependencies on lower layers
- Regularly test the process of starting your infrastructure up
  - How long does that take with a full set of data?
- Beware of soft dependencies - can easily become hard dependencies
This was not an exhaustive list

But it's a set of problems that we can do something useful about
Further general defensive strategies

Disaster testing drills  Fuzztesting  Chaos engineering
Defence: incident management process

- FEMA’s incident management system
- Practice using it for any nontrivial incident
- Any oncaller should be able to easily summon help
  - Pager alias for a higher-level cross-functional incident response team
Defence: communication

- Shouldn't rely on your infrastructure
  - Or its dependencies
- Phone bridge, IRC etc are good backups
- Make sure people (key technical staff, executives) know how to use it
  - Laminated wallet cards work
- Practice using it
Defence: priorities and budgets
Psychology of battling the black swans
Further reading:
- Michael T. Nygard’s ‘Release It!’, 2nd edition
- Other people’s postmortems:
  - github.com/danluu/post-mortems
  - sreweekly.com/
We’re hiring!

Slack is used by millions of people every day. We need engineers who want to make that experience as reliable and enjoyable as possible.

https://slack.com/careers
Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Pixabay
- And all the authors of the postmortems, articles and talks referenced throughout
Questions?

Or you can find me at @lauralifts
Links

- **Safety constraints:**
  [https://www.usenix.org/conference/srecon18americas/presentation/schulman](https://www.usenix.org/conference/srecon18americas/presentation/schulman)
- **USE method:**
- **Load shedding:**
  [https://www.youtube.com/watch?v=XNEIkivvaV4](https://www.youtube.com/watch?v=XNEIkivvaV4)
- **Layering:**
  [https://www.youtube.com/watch?v=XNEIkivvaV4](https://www.youtube.com/watch?v=XNEIkivvaV4)
- **Incident management:**