Capacity and Stability Patterns
Me

- Brian Pitts
- Systems Engineer at Eventbrite
- @sciurus on twitter
- https://www.polibyte.com
Systems
Eventbrite

- Global Marketplace for Live Events
- 600,000 event organizers
- 150 million tickets sold
- $2+ Billion in ticket sales
Obligatory hockey stick graph
Patterns we find helpful

Examples of how we apply them
Stability

Keep processing in face of impulses, stresses, or component failures
Capacity

Throughput a system can sustain with acceptable response time
Bulkheads

Partitioning systems to prevent cascading failures
Bulkheads

Load balancer

Web Servers

Database
Bulkheads

Load balancer

Web Servers

Database
Bulkheads

Load balancer

Web Bulkhead A

Web Bulkhead B

Database
Canary testing

Gradual rollout of new code
from gargoyle import gargoyle

def my_function(request):
    if gargoyle.is_active('cool_feature', request):
        do_cool_new_thing()
    else:
        do_old_boring_thing()
new proposal for `cool_feature` by alice@eventbrite.com (admin page)

bob@eventbrite.com approved alice@eventbrite.com's proposal for `cool_feature`

prod: `cool_feature` set to *active for IP Address Internal IPs OR User Percent: 50% (0-50)*
Graceful degradation

Turning functionality off in response to failures or load
Load shedding

Purposefully not handling some requests in order to reserve resources for others

You're currently in line in the waiting room.

No need to refresh your browser!

This event is quite popular, and a lot of customers are attempting to get tickets right now. Please sit tight as our system helps the event organizer process orders on a first-come, first-serve basis. Your spot in this waiting room does not guarantee tickets. If tickets become available, you'll be prompted to complete your order. If tickets are unavailable, we'll send you back to the event page.
Rate limiting

Controlling the amount of work you accept
Timeouts

Limiting time you wait for a request to complete
Caching

Saving and re-serving results to reduce expensive requests
Invalidation strategies

TTL: Keep it short, stupid

For service calls, centralized invalidation logic

Wrapper strategy for dynamic TTL
Wrapper example

Request from user


Response from app to varnish

<html>
  <!-- URL to inner page, details obfuscated to protect the guilty -->
  <esi:include src="/esi/event/35662110332?lang=en&timestamp=1509732000">
</html>
Capacity Planning

Getting the resources you need in place, before you need them
Recap

- Bulkheads
- Canary testing
- Graceful degradation
- Rate limiting
- Timeouts
- Load shedding
- Caching
- Planning
Further resources

- Release it! - Michael Nygard
- The Art of Scalability - Abbott and Fisher
- The Practice of Cloud System Administration - Limonocell, Chalup, and Hogan
- Site Reliability Engineering - Beyer, Jones, Petoff, and Murphy
- Production-Ready Microservices - Susan Fowler
Thanks!

Questions?

@sciurus / https://www.polibyte.com