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

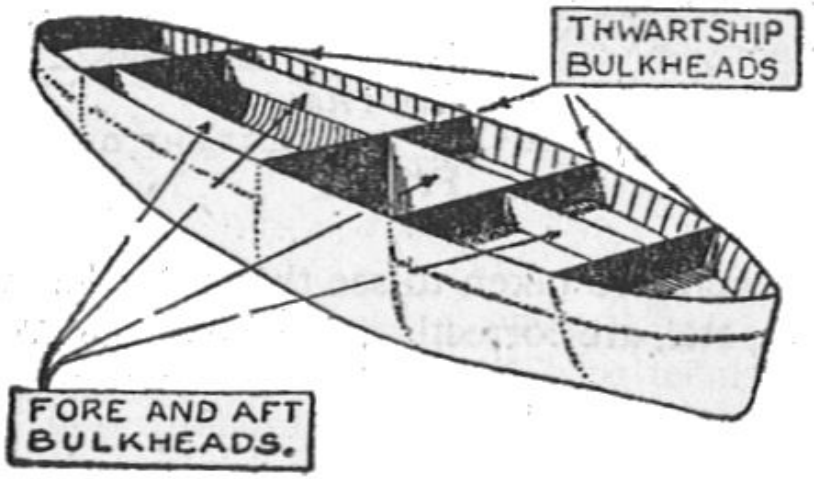
Keep processing in face of impulses, stresses, or component failures

Capacity

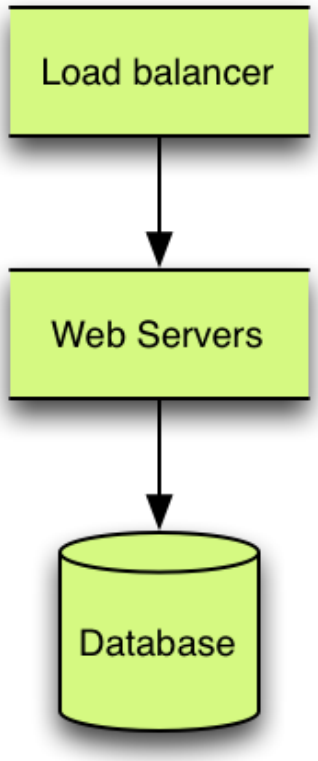
Throughput a system can sustain with acceptable response time

Bulkheads

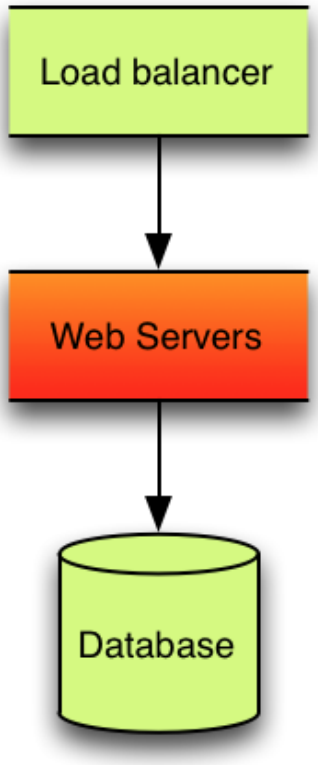
Partitioning systems to prevent cascading failures



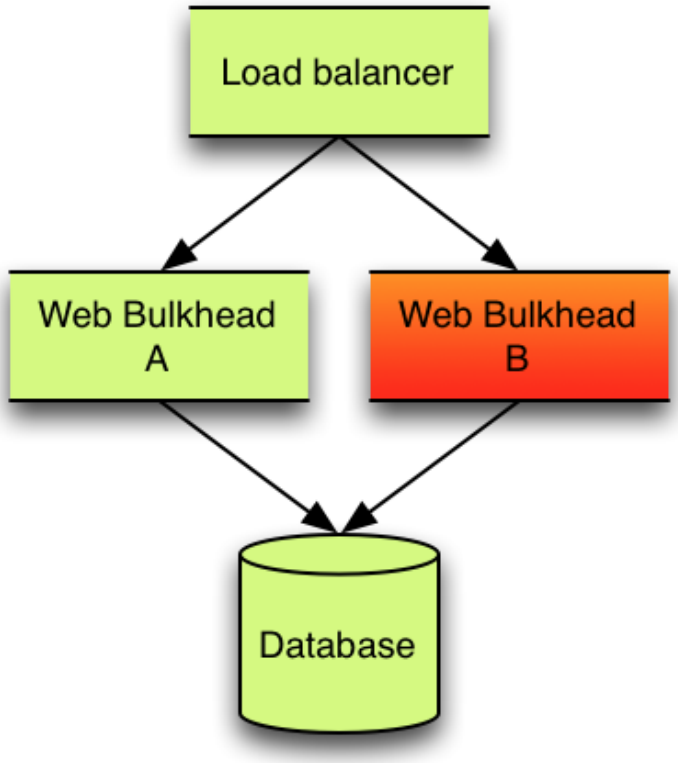
Bulkheads



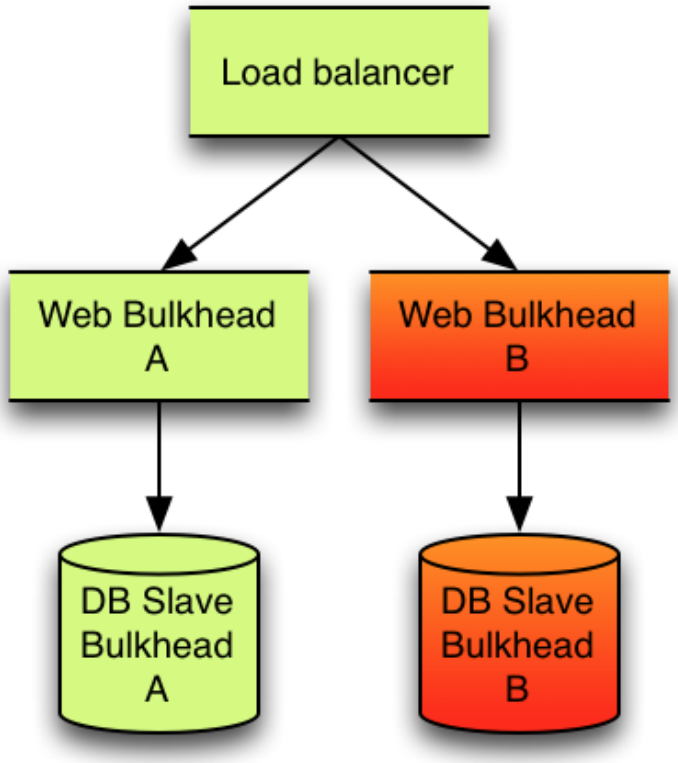
Bulkheads



Bulkheads



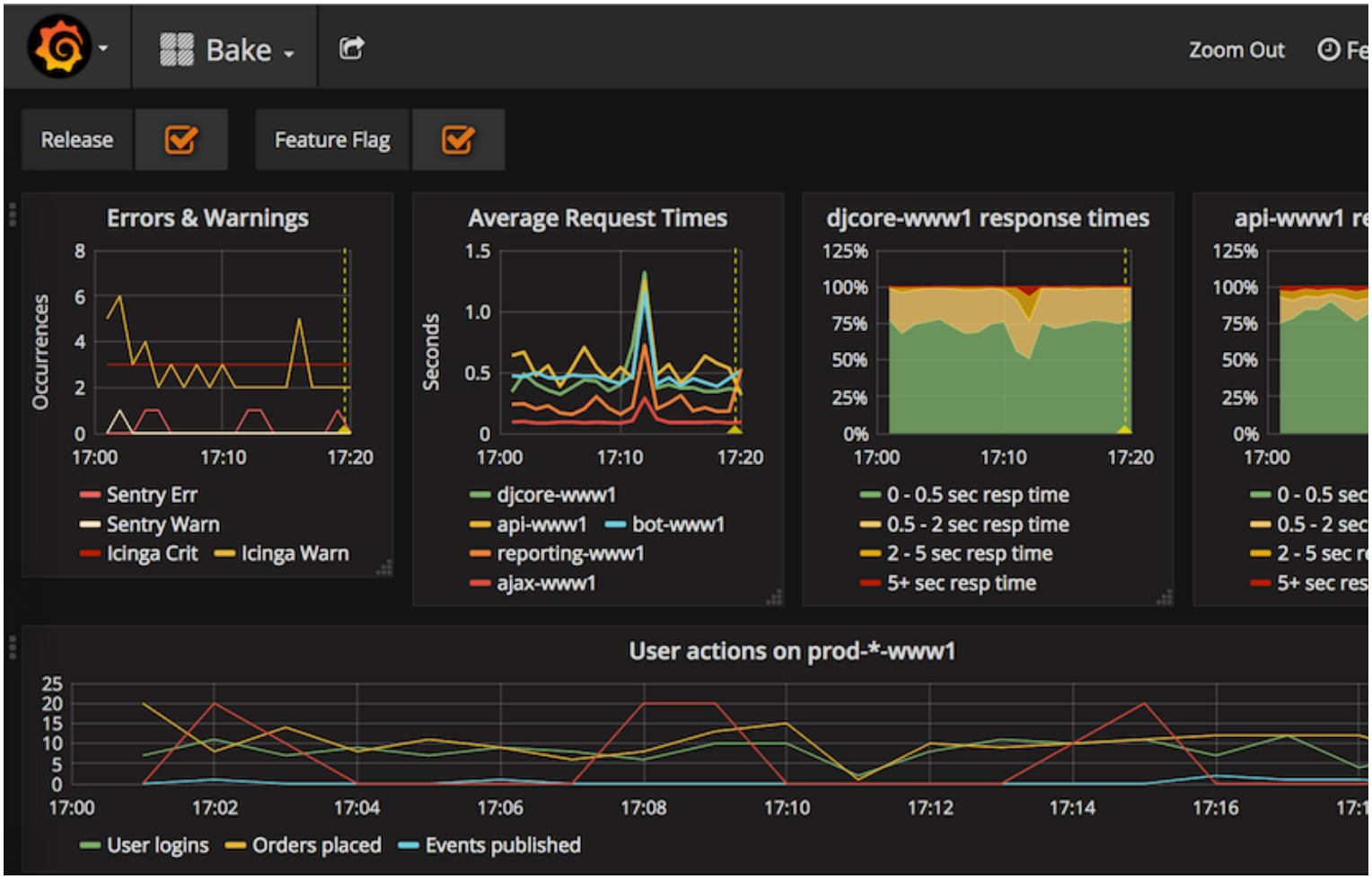
Bulkheads


Load balancer
Web Bulkhead A
Web Bulkhead B
DB Slave Bulkhead A
DB Slave Bulkhead B

Canary testing

Gradual rollout of new code

Baking releases



Feature flags

```python
from gargoyle import gargoyle

def my_function(request):
    if gargoyle.is_active('cool_feature', request):
        do_cool_new_thing()
    else:
        do_old_boring_thing()
```

Feature flags

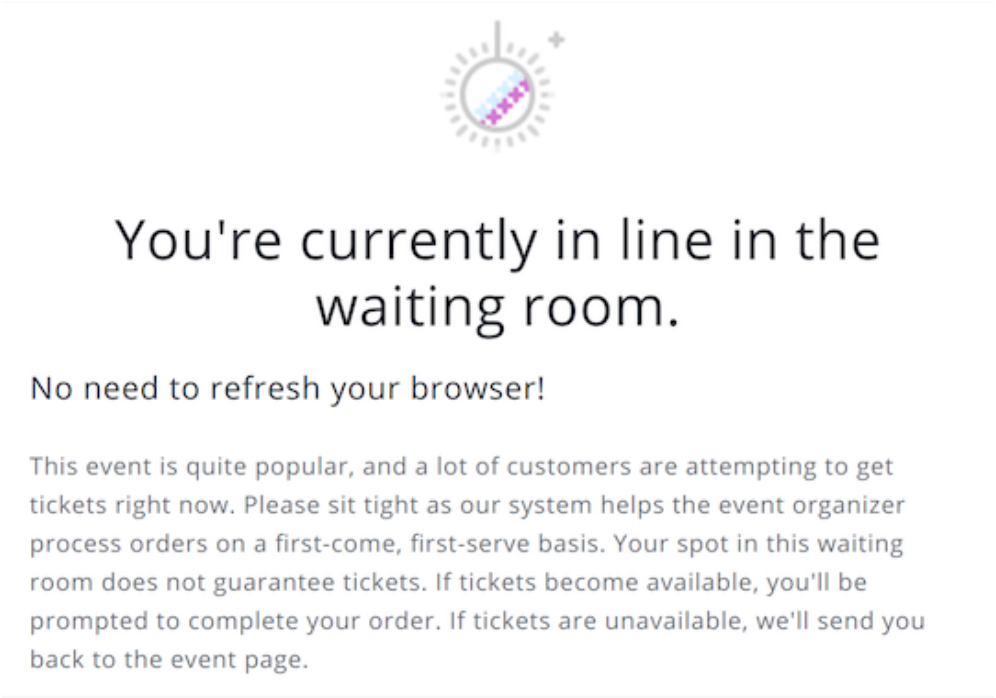
```
bot BOT [15:49]
new proposal for `cool_feature` by alice@eventbrite.com (admin page)

bot BOT [16:15]
bob@eventbrite.com approved alice@eventbrite.com's proposal for `cool_feature`

Rho BotBOT [16:15]
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



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

https://www.eventbrite.com/e/pytennessee-2018-tickets-35662110332?aff=ehomesaved

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