Exploring Services Architecture at Bungie

Michael Williams

BUNGiE

Hi Everyone!

- I'm Michael Williams
- This was built by a ton of talented folks
- I'll be in the chat!

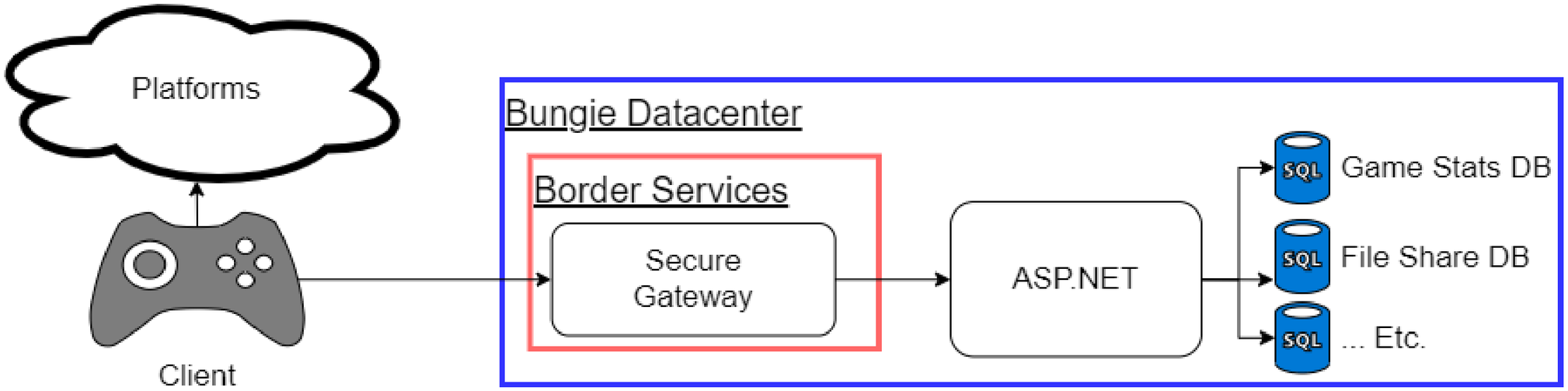
BUNGiE

A Brief History of Services

- **Halo Services**
  - Xbox Live handled Matchmaking, Friends, Session state, and (most) Presence
  - Bungie Services
    - Bungie.net Stats
    - File Share (screenshots, saved films, edited maps)
    - Cheat detection
    - Playlist settings
    - Player server-to-client messaging

BUNGiE

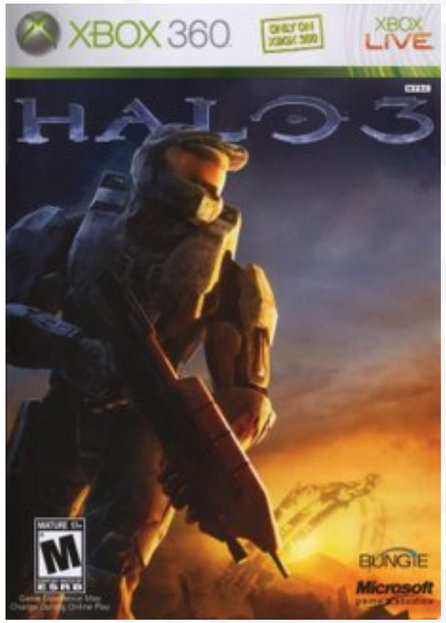
Halo Service Technology



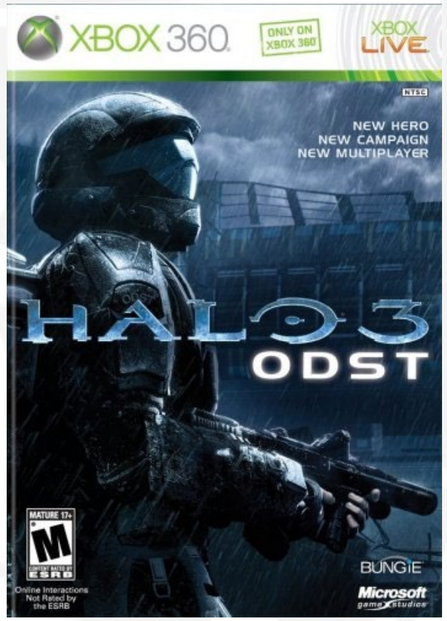
BUNGIE

Service Engineers per-game

.5



2



3

4

Note: Excludes Bungie.net Engineers and Gameplay Networking Engineers

BUNGiE

Pivot to Destiny

- Always Online game
- Loot-based gameplay means larger player data
- Multi-platform means high concurrency
- A complex hybrid hosting plan
- A “single shard” universe

BUNGiE

Pragmatic Choices

- Stick with Windows servers to host Destiny
- Continue using C# to write services
- Microsoft SQL as the Backing store

BUNGIE

Additional Choices

- **Destiny services aren't highly latency sensitive**
  - We can build out a single Datacenter to control complexity!
- **Grow the team to meet the challenge**
  - 4->16 Service Engineers at the time of Destiny 1 Launch

BUNGiE

Success!!


Home › Originals › A Thank You to Sony, Microsoft and Bungie for a smooth Destiny...

Originals

A Thank You to Sony, Microsoft and Bungie for a smooth Destiny launch

By **Matt Liebl** - September 11, 2014


Destiny 2 enjoys a fairly smooth launch

There are some issues being reported, but for the most part, it's good news.

2017-09-07 11:49 • Bengt Lemne •

BUNGiE

…Mostly

Kicks and queues: Destiny's Rise of Iron expansion is having a bumpy rollout

Posted **5 years ago** by Chris Carter

33

EDITORS' PICK | Jan 29, 2020, 08:27am EST | 6,141 views

'Destiny 2' Restores Missing Currency In Its First Ever Rollback, Losing An Hour Of Game Progress

BUNGIE

Destiny Today

- **Nearly 40+ Services**
- **18 unique SQL Databases**
  - Not counting shards!
- **9 Redis caches**
- **Plus plenty of other tech**
  - Kibana,
  - Elastic Search / Graphana,
  - Redis,
  - Etc

BUNGiE

Why can't I hold all these Microservices?

SignOn

SignOn Queue

Accounts

Commercialization

STUN

Client Config

Service Config

(Claims)

BAP (Front door)

World Server

(Queue Pub-Sub)

Relay Conversation Manager

Nat Relay

Clans

Activity Host

Activity Host Proxy

Bubble Host

Bubble Host Proxy

IPM (Intra Process Monitor)

IPM Agent

IPM Locator

Mission Control

(Log Processor)

(Website Gateway)

(Mission Control Gateway)

(Perf Counter Gatherer)

AWS Elastic Relay

GraphiteS3Relay

SMS Verification

Friends

Invites

Chat

(Matchmaking)

PlayerPrivacy

MonitoringProxy

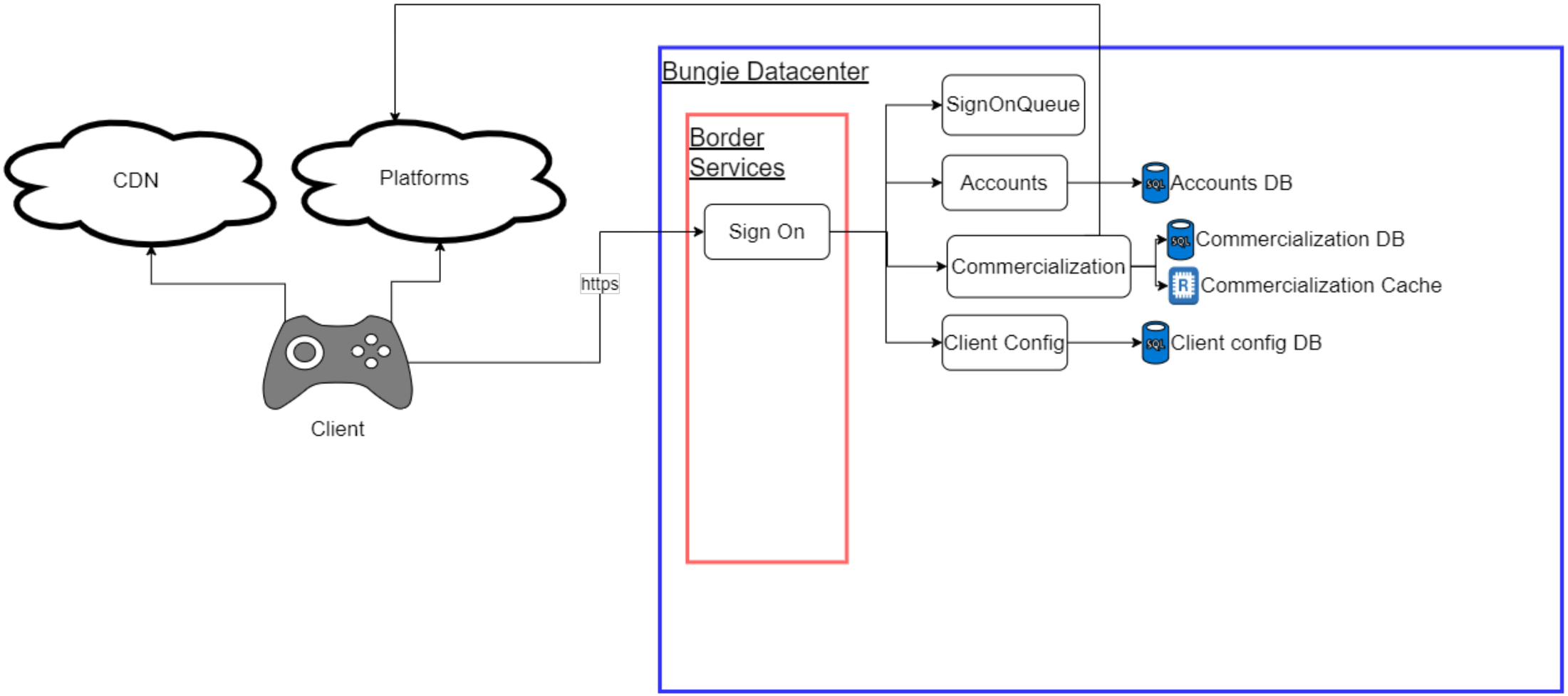
Leaderboards

Presence

Prototype Service

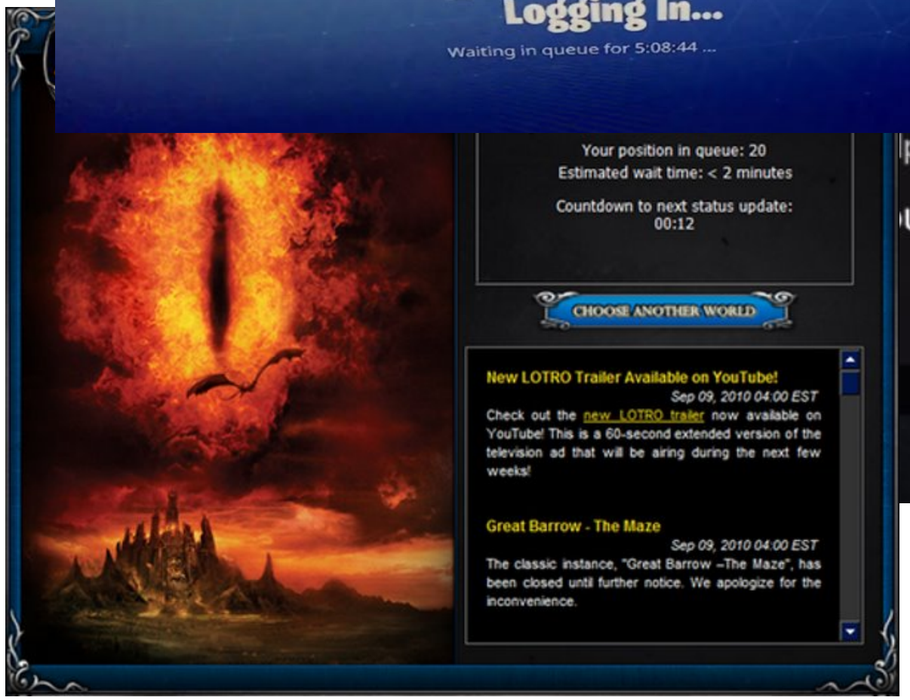
BUNGiE

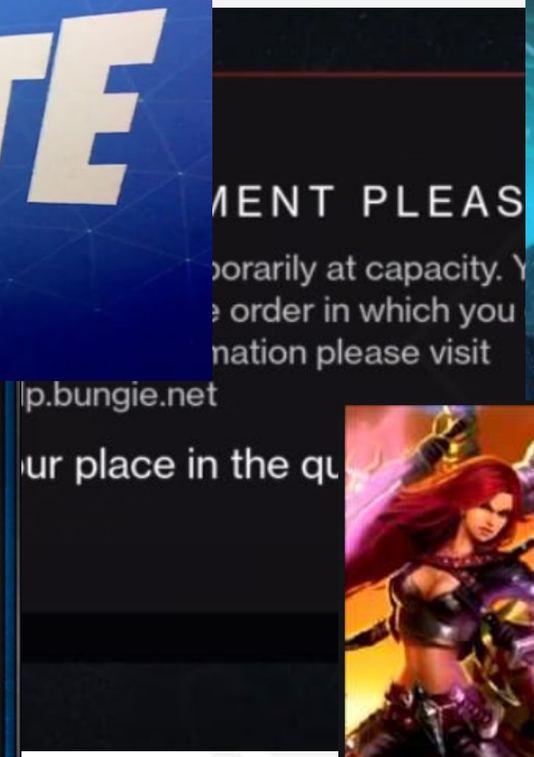
Bootflow

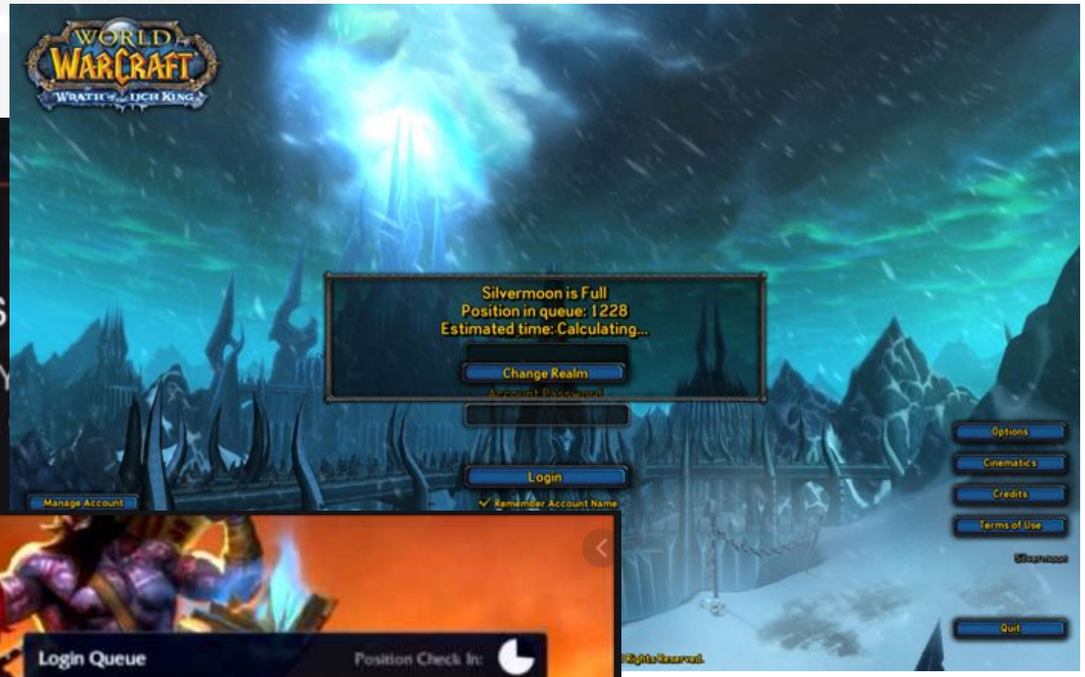


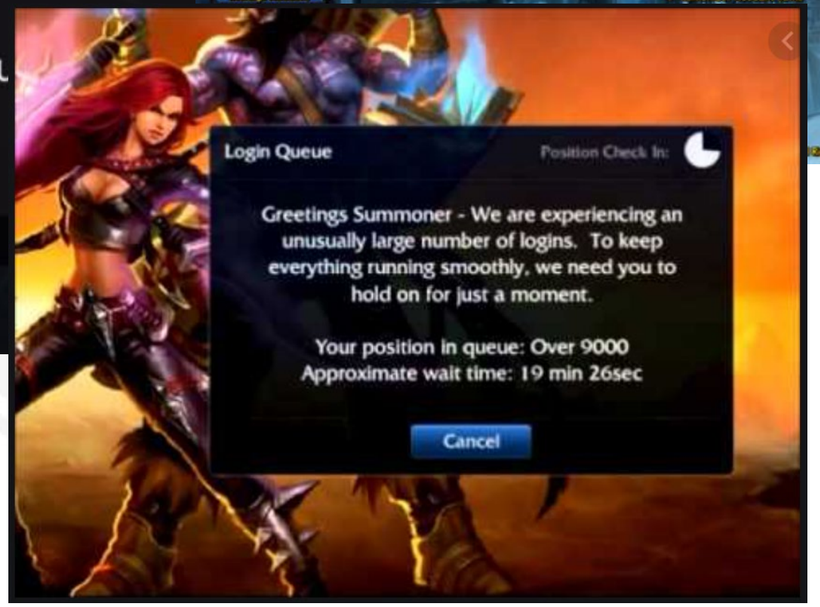
BUNGiE

Sign On Queues and Throttles









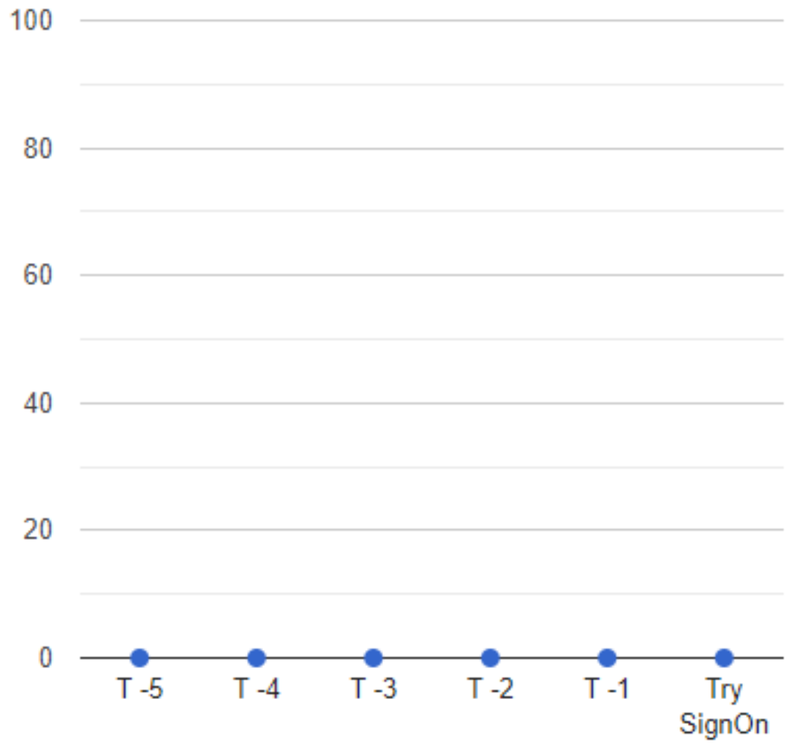
BUNGiE

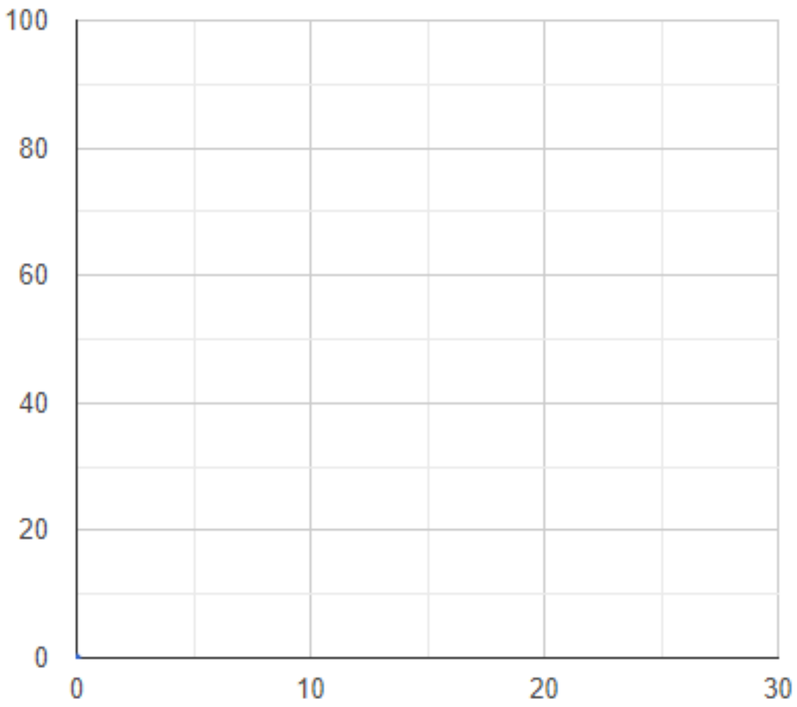
Risks to Online Service Uptime

- You have a maximum capacity
- "Thundering Herds" of players
- Hardware failures
- You will likely want planned downtime

BUNGiE

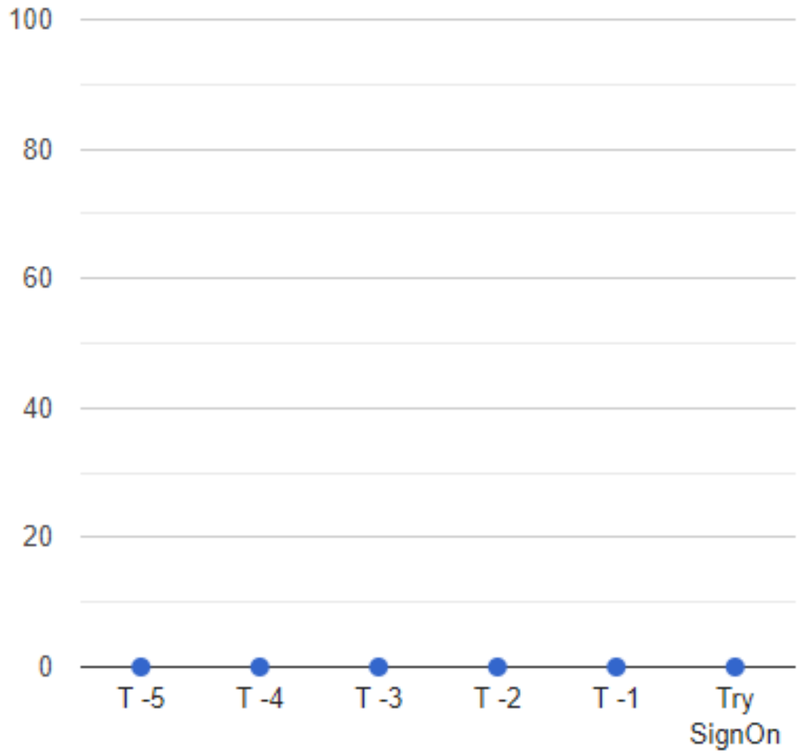
Sign On Queue

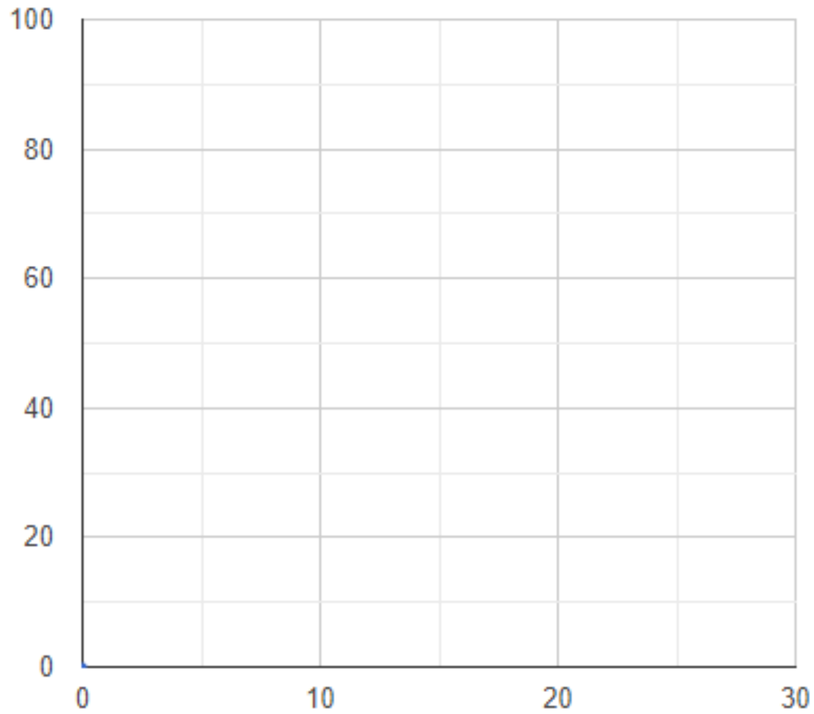




BUNGiE

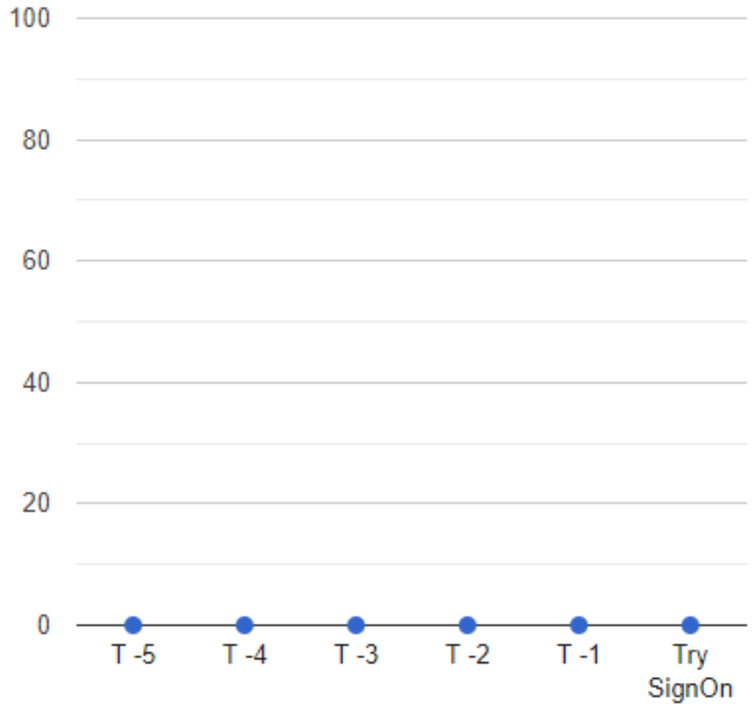
Sign On Throttle

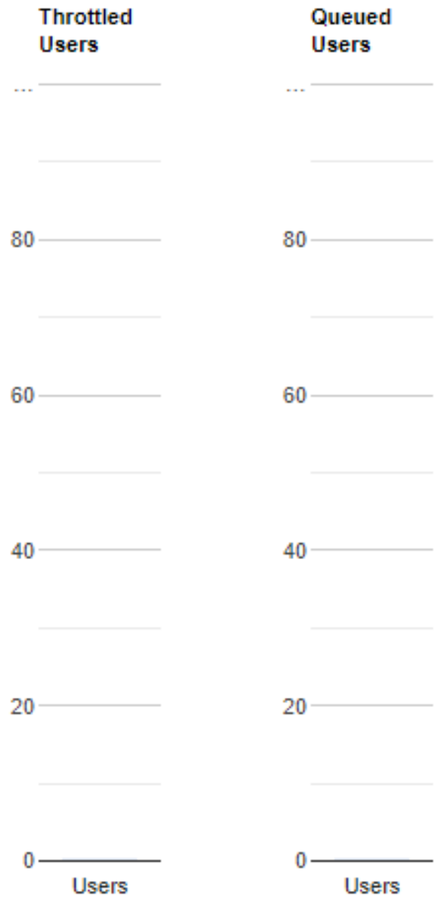


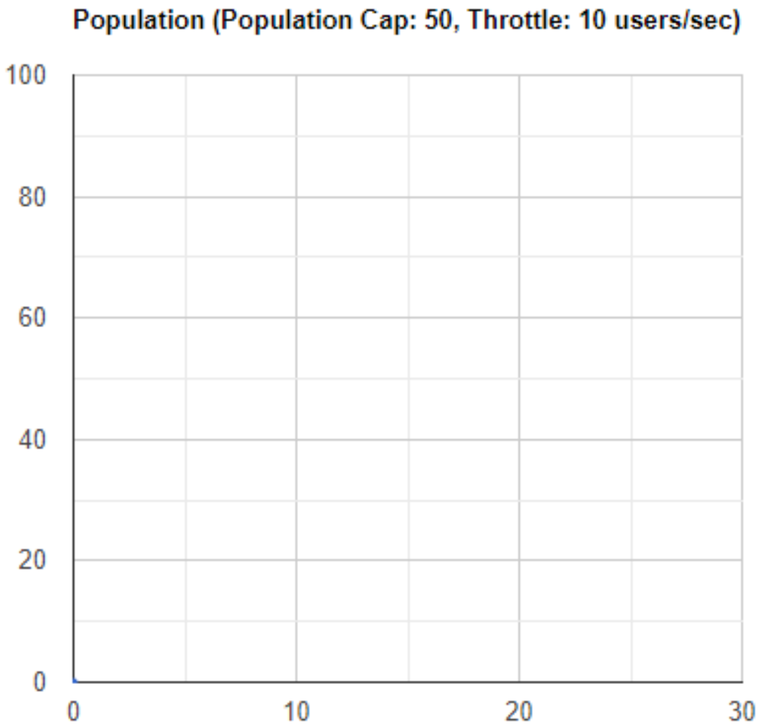


BUNGiE

Combined!







BUNGiE

Queue & Throttle Response Opportunities

- **Responses can control client behavior!**
  - Give the client a token to provide on retry
  - Can send custom messages for the client to display

BUNGiE

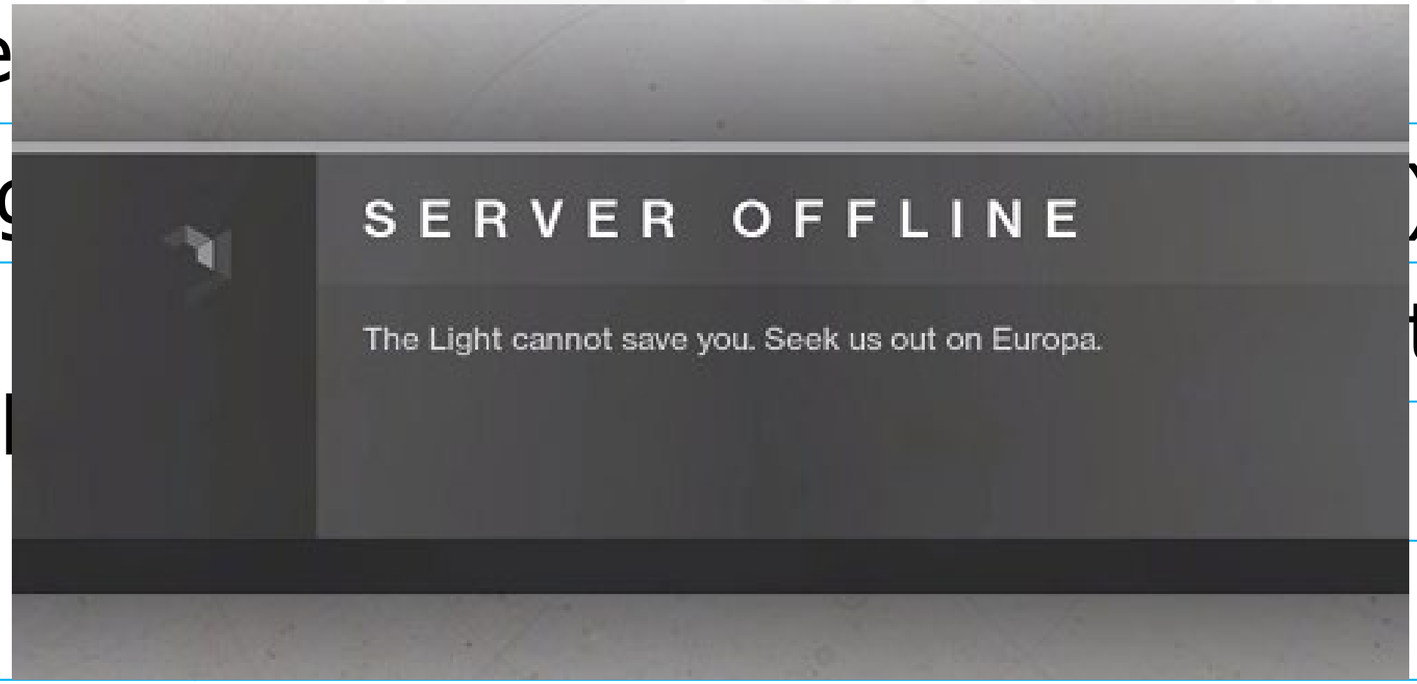
Destiny Queue Response

Current Position in Queue	
Message to display to user	
Next time to retry	
Remaining Time Estimate (currently unused)	
Queue Token	Ticket ID ("real" queue position)
	Original entry time
	Last known queue state.

BUNGiE

Destiny Queue Response

Current Position in Queue
Message to display to user
Next time
Remaining
Queue Tol



BUNGiE

Destiny Queue Response

Current Position in Queue	
Message to display to user	
Next time to retry	
Remaining Time Estimate (currently unused)	
Queue Token	Ticket ID ("real" queue position)
	Original entry time
	Last known queue state.

BUNGiE

Best Practices

- Limit your queue & throttle dependencies
- Put the queue and throttle in front of everything, including authentication
- Have an allow-list for test accounts

BUNGiE

Best Practices (Continued!)

- **Use the queue as the main gating for your game**
- **Default population cap to 80% of your known capacity**
- **If an issue is happening:**
  1. Clamp the population cap to 0
  2. Let people drain from the game (or kick them)
  3. Then slowly ratchet the population cap back up

BUNGIE

Key Takeaway #1

- **Sign on Queues and Throttles are one of your first and best tools to handle and prevent Service issues**

BUNGiE

Into Orbit!

BUNGiE

BAP Server (Bungie Access Protocol)

- Stateful gateway to Destiny's service layer
- Communication on TCP
- Securely encrypted using the token provided from the Sign On service.

BUNGIE

WorldServer

- Character data is worked on in-memory
- One of our few stateful services
- A given server can handle ~5000 accounts
- Hosts C++ game logic DLLs that run on player data

BUNGiE

Claims Service ("ClaimZ")

- Acts a simple routing service to the stateful Worldserver
- Redis Backed

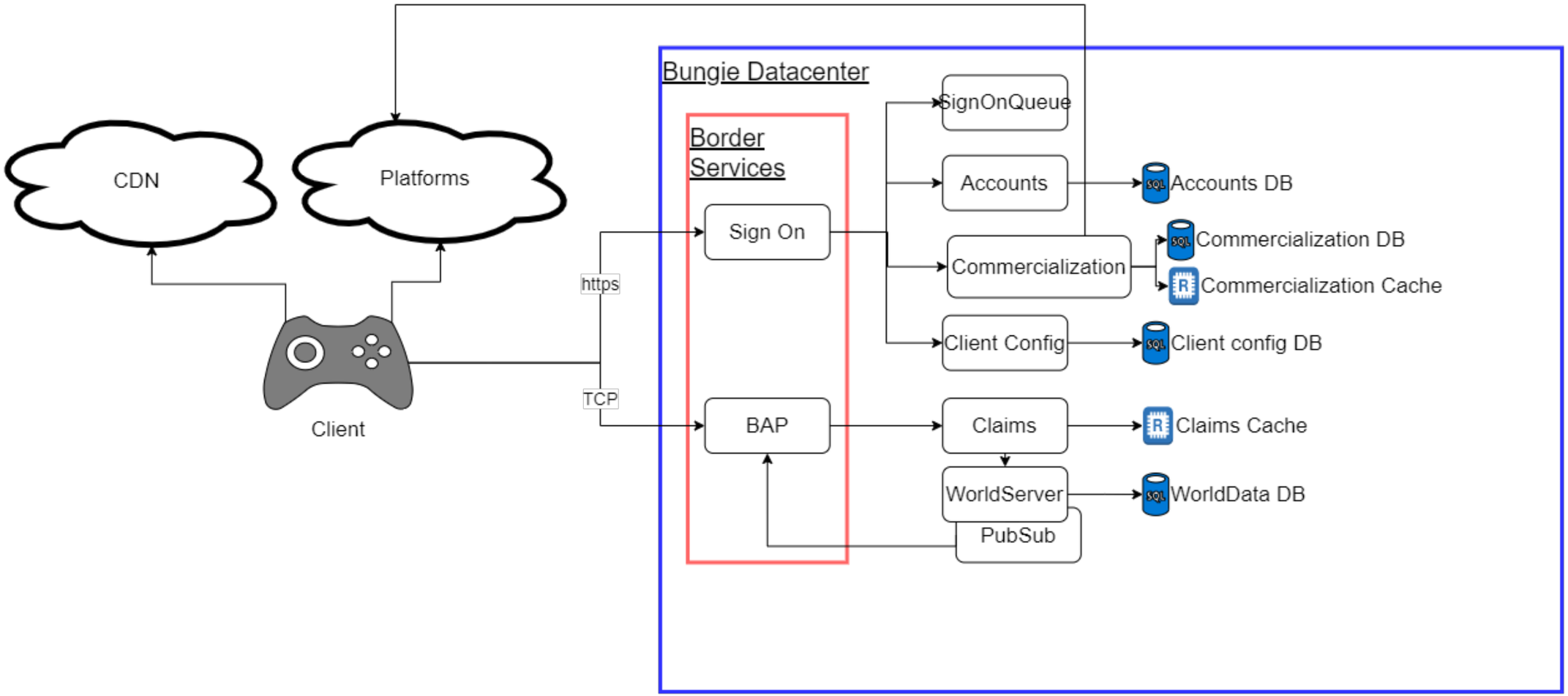
BUNGiE

Character PubSub Service ("QueueZ")

- A subcomponent of WorldServer
- PubSub system with in-order differential updates
- Can subscribe to different levels of detail on any character
- A given client is subscribed to itself, party members, friends, clanmates
- Uses ZeroMQ

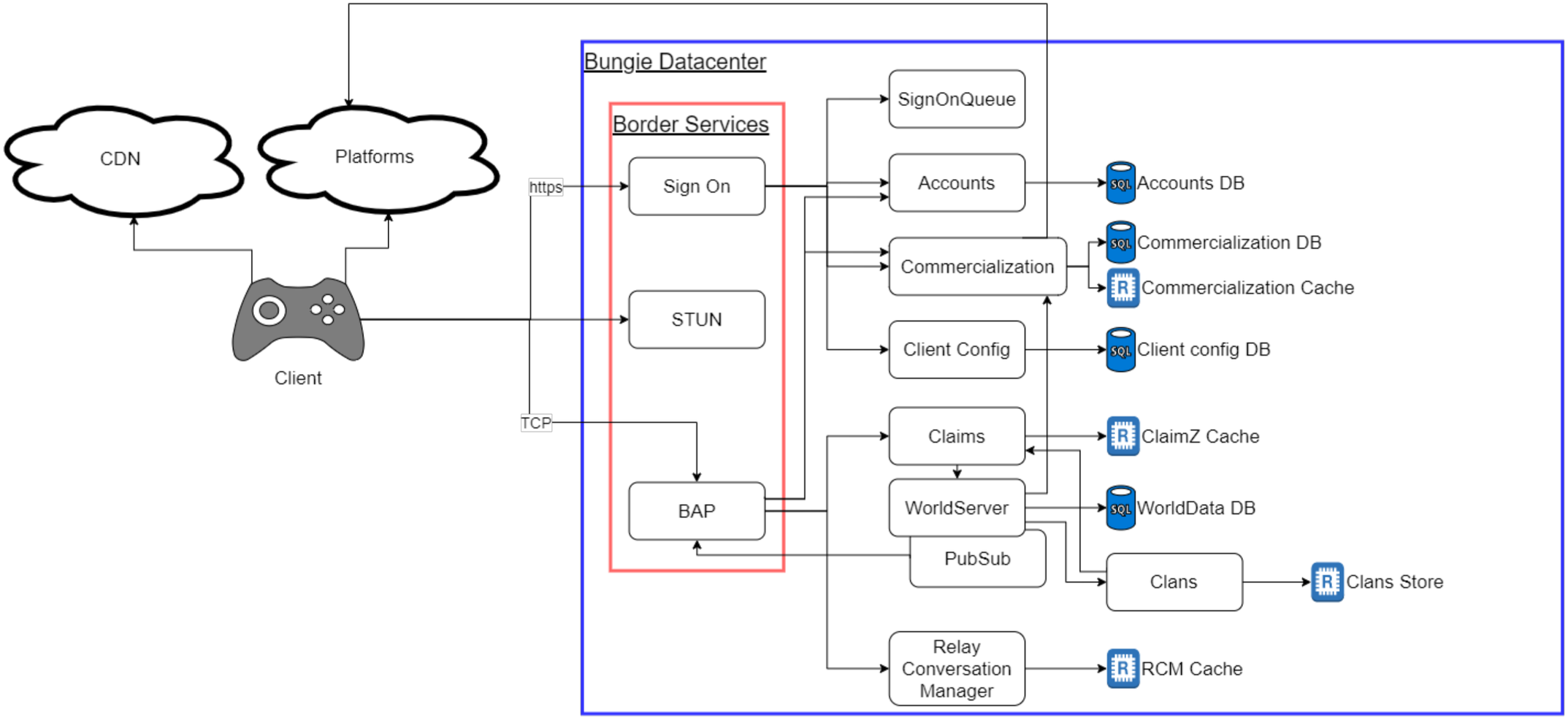
BUNGiE

Into Orbit!



BUNGiE

Into Orbit!



BUNGiE

Character Storage

BUNGIE

Character Storage - Schema

AccountCharacters	
AccountId	CharacterId
1	10
1	11

AccountCharacterAttributes		
CharacterId	CharacterAttributeId	CharacterAttributeValue
10	101	1111
10	102	2222
11	101	1111
11	102	4444

CharacterItems	
CharacterId	ItemId
1	10
1	10

CharacterItemAttributes		
ItemId	ItemAttributeId	itemAttributeValue
1000	201	1234
…	…	…

BUNGiE

Character Storage - Issues

- **Heavy duty loads**
  - 6000 account attributes
  - 4000 character attributes per-character
  - 25 attributes per item
  - 3 characters + 500 account items = 30,000+ rows!

BUNGiE

Character Storage – Issues (Continued)

- Analytics queries were difficult
- SQL fixups were terrifying
- Low savings from incremental updates

BUNGIE

Character Storage V2

- **A New blob-store model**
  - Accounts blob
  - Per-Character blobs
  - One blob for all items
- **Binary blobs encoded using protocol buffers**
- **Still stored in SQL**

BUNGiE

Character Storage V2 - Results

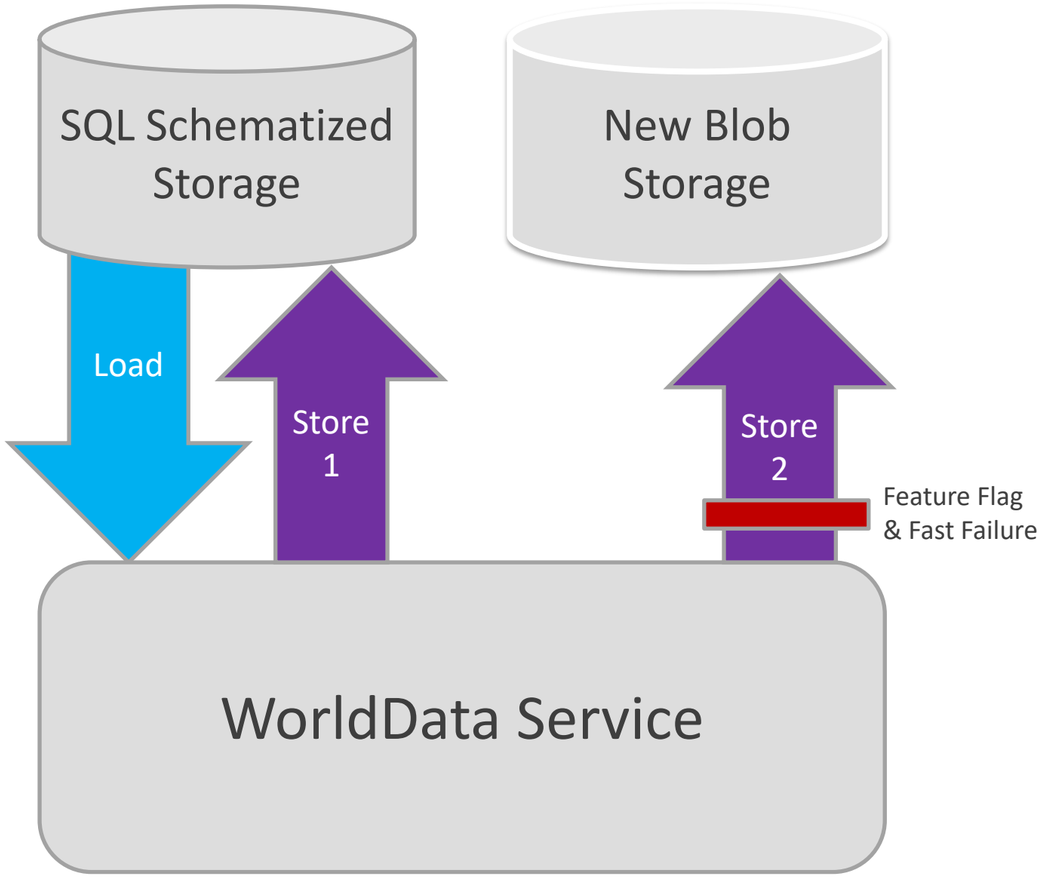
- 90% savings in space, and improved load times
- Easily stores 200,000+ “rows” for an account
- New debugging functionality made possible
- No major conversion downtime impact for users!
  - ?!?

BUNGiE

Migration best practices

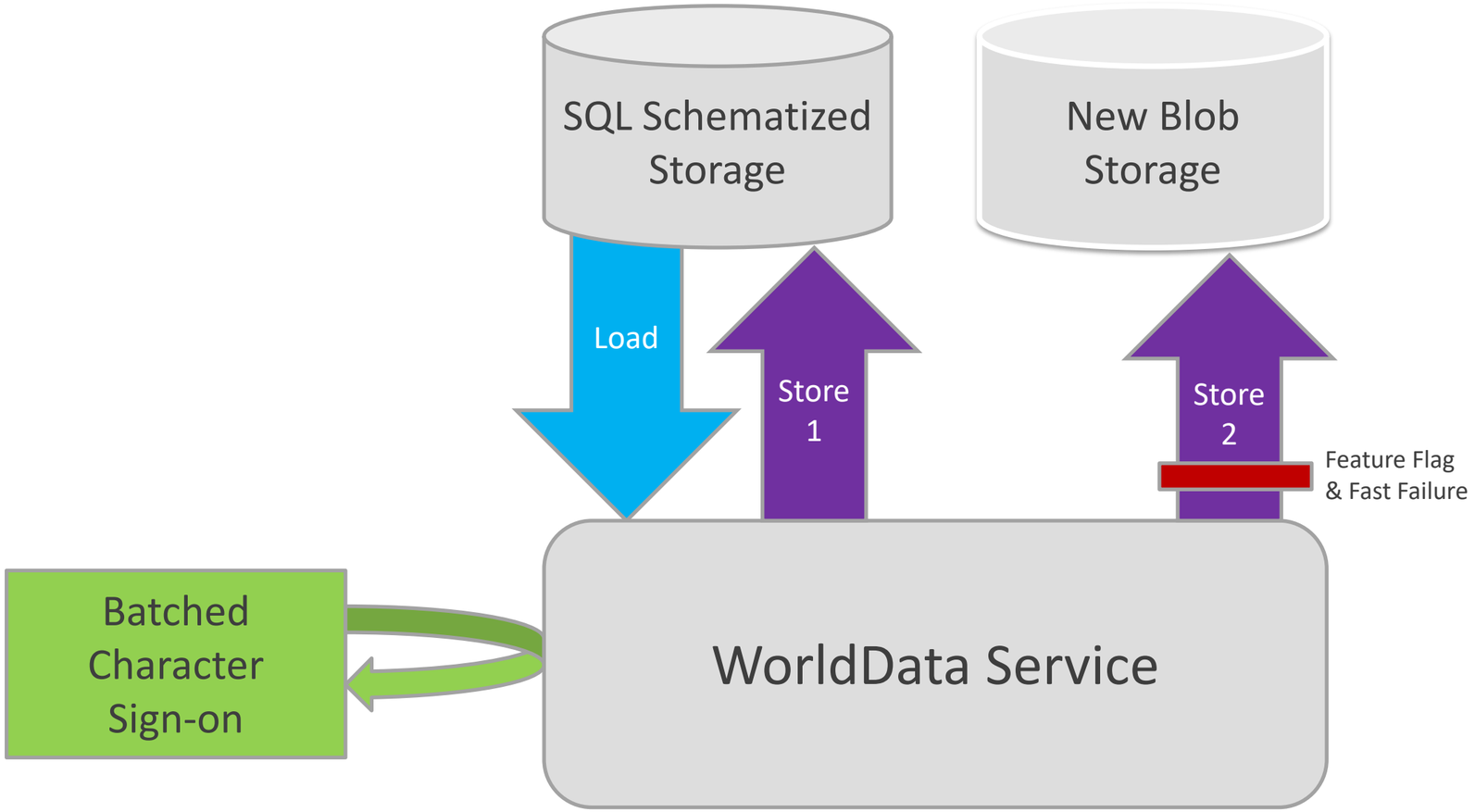
BUNGIE

Migration best practices – Step 1



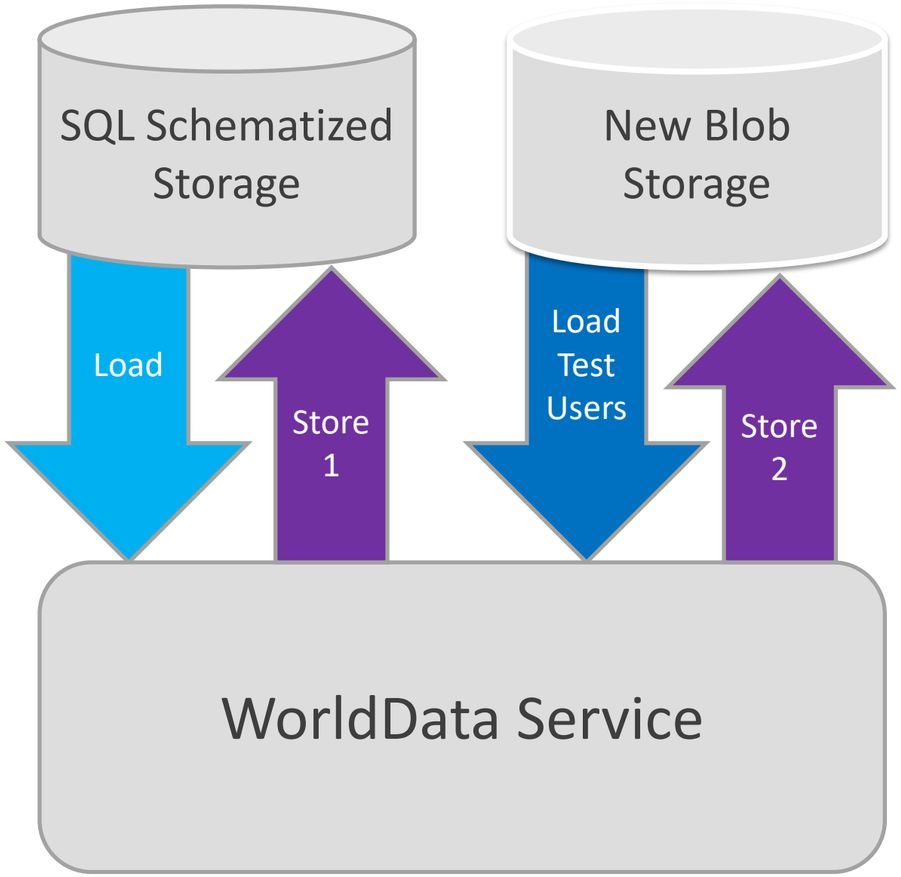
BUNGIE

Migration best practices – Step 2



BUNGiE

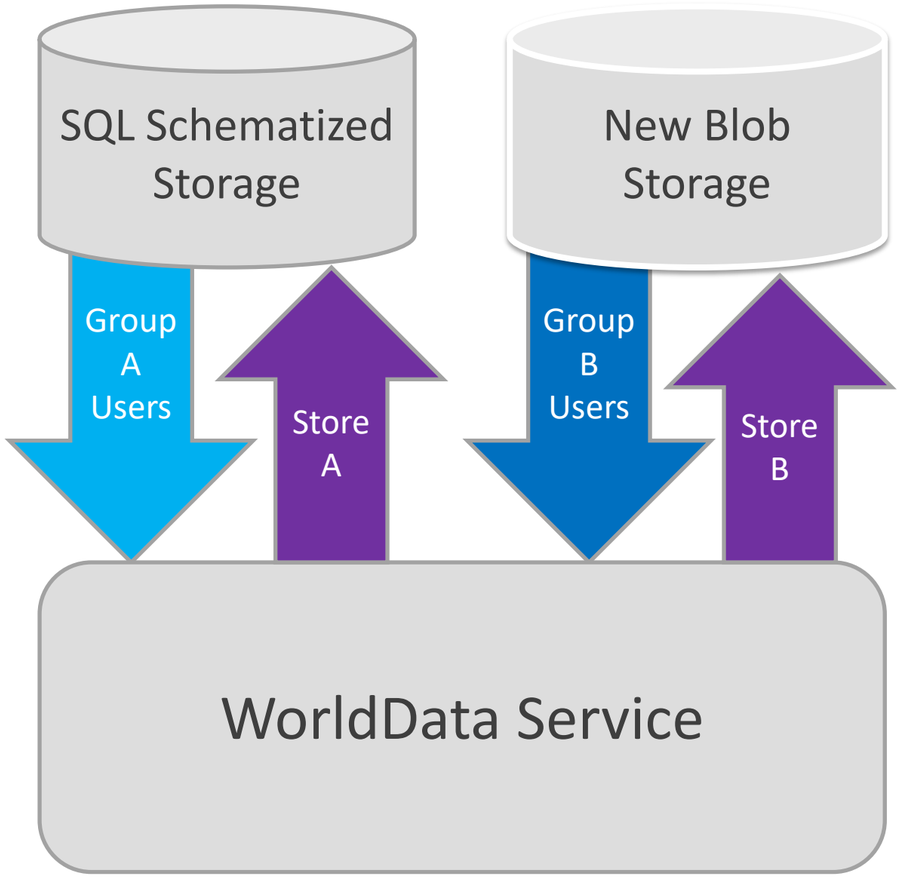
Migration best practices – Step 3



BUNGiE

Migration best practices – Step 4

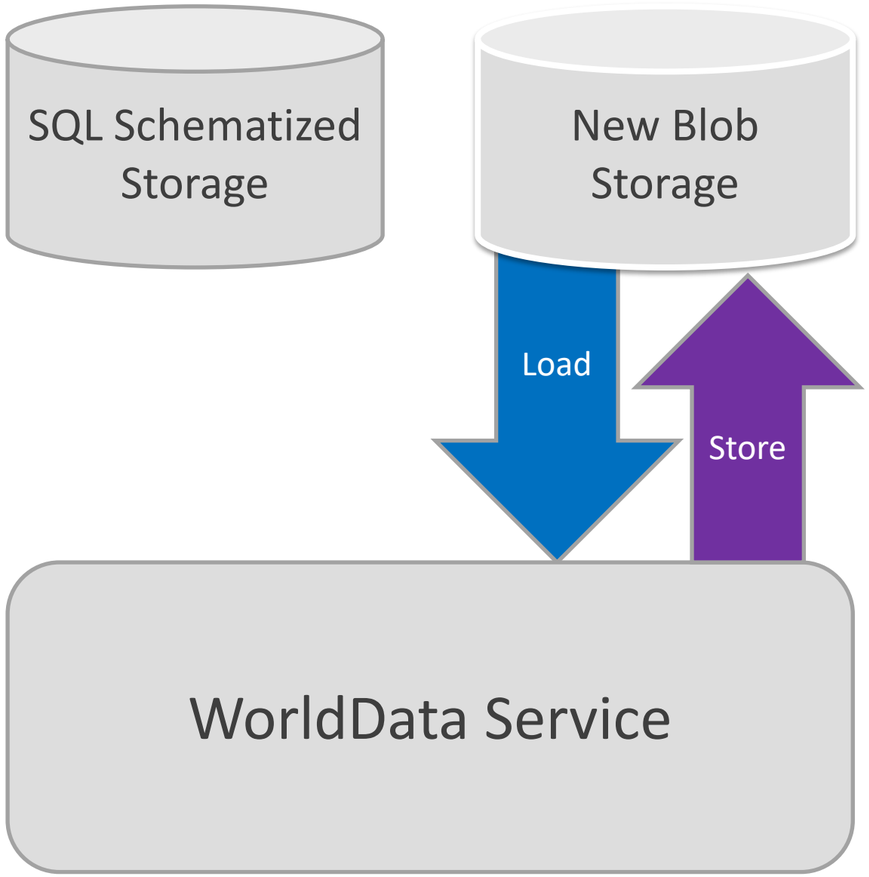
Group A: 100% ➔ 0%



Group B: 0% ➔ 100%

BUNGiE

Migration best practices – Step 5



BUNGiE

Migration best practices - Generalized

- Write to old and new systems
- Make sure all accounts have written to both
- Set test accounts to read from new
- Slowly ramp the population from 1% to 100%
- Turn off the old version

BUNGiE

Key Takeaway #2

- **Whenever possible, launch new systems side-by-side with old systems, and slowly cut over**

BUNGiE

Data Reliability

BUNGiE

Data Reliability

- Character data is very important to persist
- If a player disconnects, we persist their data
- But what about if the WorldServer crashes?
- Constant write-through super-expensive
- Change logging is super-complex

BUNGiE

Data Reliability - Solution

- **Persist character data every 5 minutes.**
- **Do a bonus persist on "important" events**
  - Like getting an exotic item
- **If a WorldServer dies, player loses at most 5 minutes of progress.**

BUNGIE

Key Takeaway #3

- **Understand the real reliability requirements for your systems and be skeptical of 100% targets**

BUNGIE

Clans System

- Clan data can have many simultaneous writes
- Used a stateless model with optimistic concurrency

BUNGIE

Clans Optimistic Concurrency Model

1. Receive clan action from WorldServer
2. Load clan data and clan data version
3. Run action against clan data
4. Attempt to persist.
5. If the stored version is different from the persisted version, go back to step 2 (Retry N times)

BUNGIE

Clans System Results

- **In practice, this worked really well**
- **Used Redis for the persistent store due to read/write rate capabilities**
- **Extra consideration – Different member content**
  - When do you version the clan data?

BUNGIE

Activity Hosts & Bubble Hosts

BUNGiE

Activity Hosts & Bubble Hosts

- **“Shared World Shooter: Destiny's Networked Mission Architecture”**
  - Justin Truman, GDC 2015
- **Act as the game script & physics hosts for the game**

BUNGiE

Activity Hosts & Bubble Hosts

- **4 services**
  - Activity Host (AH),
  - Activity Host Proxy (AHP)
  - Bubble Host (BH),
  - Bubble Host Proxy (BHP)
- **1 Proxy per machine, many Activity & Bubble hosts**

BUNGiE

Activity Hosts & Bubble Hosts

- AH & BH are cut-down versions of the game client
- AHP & BHP are C# routing and management services
- 1000s of Activity Hosts and Bubble Hosts per server
- AHP/BHP start up "Zombie" instances in advance

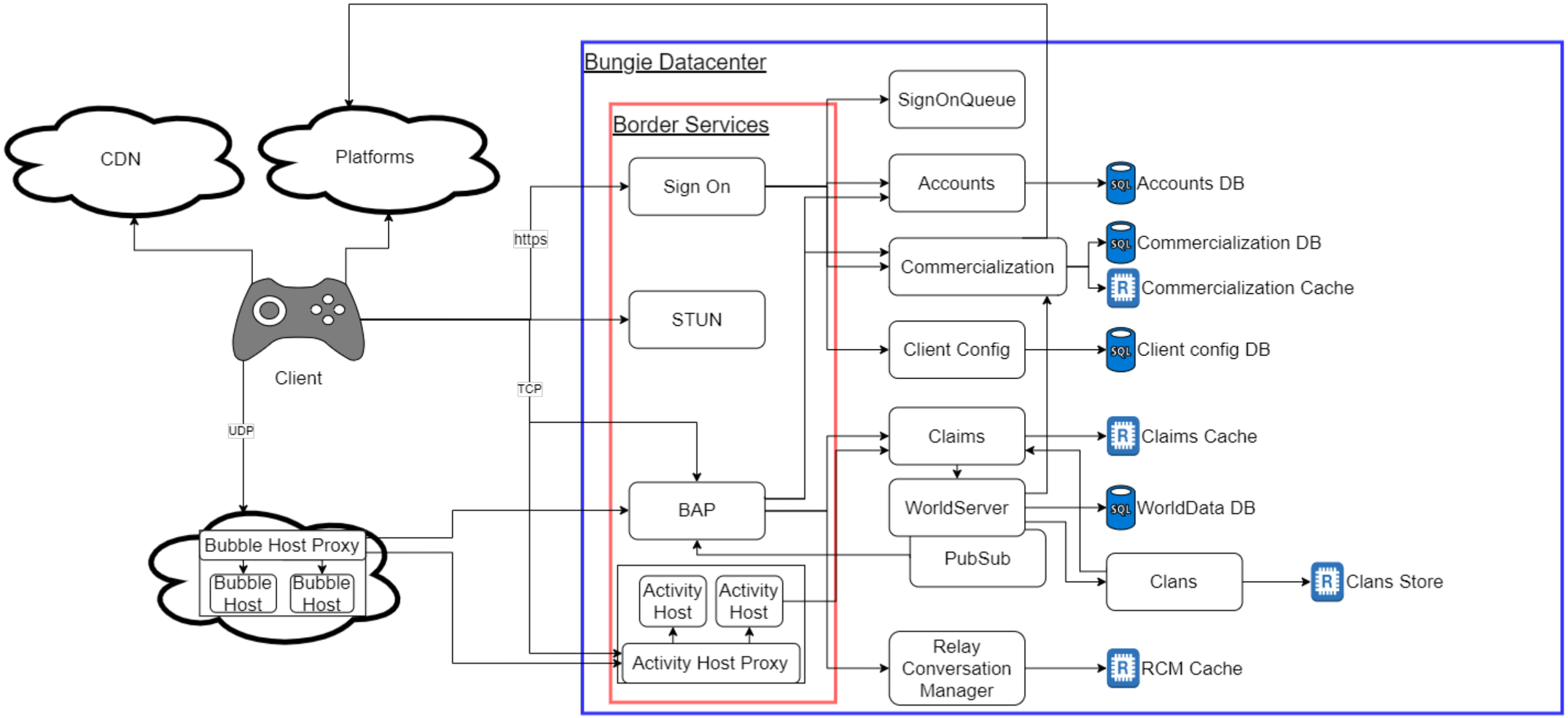
BUNGiE

Activity Hosts & Bubble Hosts

- Activity Host uses TCP.
- Bubble Host uses UDP
- Bonus Host Type: Group Activity Host

BUNGiE

Destiny's Service Infrastructure



BUNGiE

Topic Grab Bag!

- Stress Testing
- Cloud Usage
- User Error Reporting
- Service Settings
- Logging

BUNGIE

Stress Testing

- Major run each annual release
- Allocate a portion of our datacenter as a stress cluster
- Cloud hosted virtual clients
- Finds key ship-stopping issues every time

BUNGiE

Stress Testing - Drawbacks

- Expensive
- Labor intensive
- Heavy maintenance burden
- An Imperfect simulation

BUNGIE

Stress Testing - Alternatives

- **Stress Test individual services as much as possible**
  - Doesn't catch multiple services contending for the same resource
- **Rely on the Queue to protect you**
- **Prefer soft-launching new features and systems**

BUNGiE

Cloud Usage

- **Destiny 1**
  - Peer-to-Peer networking allowed a single datacenter
  - NAT Relay had to be geolocated
- **Destiny 2**
  - Allowed Bubble Hosts to scale into the Cloud
  - Still not geolocated!
  - Used to handle population spikes for large launches

BUNGiE

User Error Reporting

BUNGiE

Service Settings

- **Definition**

```
public readonly bool CrossSaveWarningEnabled = Setting(
    true,                                                    ⇦ Default Setting
    "Whether or not to give a one-time warning to users about cross save.",   ⇦ Text description of setting behavior
    Behavior(Retrieving.PerRequest))                         ⇦ Describing how soon the setting will "take" after being changed (per call? after service restart?)
    .Override(false, Service.Any, Environment.Onebox)
    .Override(false, Service.Any, Environment.Shared);       ⇦ Optionally overrides the default setting per-environment
```

- **Usage**

```
if (SignOnServiceSettings.CurrentValue.CrossSaveWarningEnabled)
{
    // Send the Cross save warning!!
}
```

BUNGiE

Service Settings

Edit Setting: Search:

CrossSaveWarningEnabled

— Whether or not to give a one-time warning to users about cross save.

Not available on servers operated by public cloud providers.

Member of Bungie.Server.SignOnService.SignOnServiceSettings

from ...\Bungie.Server.SignOnService\Settings\SignOnServiceSettings.cs, line 923

» Protect CrossSaveWarningEnabled from deployment changes.

Service	Environment	Tag	Server	Value	
*	*	*	*	True	e +
*	Onebox	*	*	False	e + x
*	Shared	*	*	False	e + x

Generate Export XML » Display Inline
Generate C# Code

Services will pick up changes to this setting PerRequest

Published Change History

Expand All

- Thu Sep 24 2020 11:09:45 GMT-0700 (Pacific Daylight Time) by DeployTool(UserName)

BUNGiE

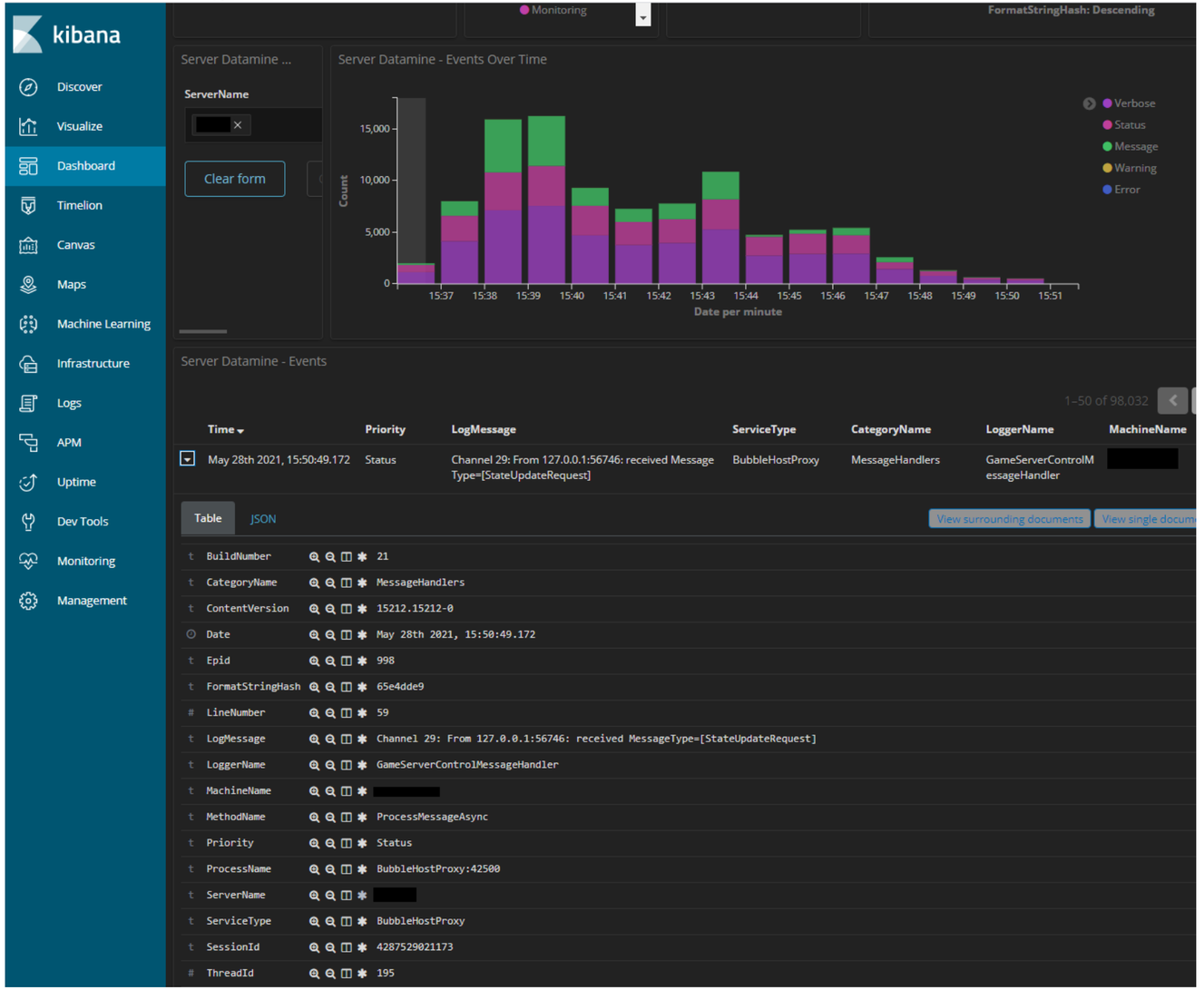
Logging

```
_logger.Warning(sessionContext,
    m => m("Invalid machine identifier: {0}",
    Hexer.ToHex(extendedMachineIdentifier)));
```

- Session Context has caller information for filtering
- String Format only called if error level active
- Calculates a hash of the format string

BUNGiE

Logging



BUNGIE

Key Takeaway #4

- **Invest in making it as easy as possible for your engineers to do the right thing**

BUNGiE

Not everything goes right

BUNGiE

Time for cowboy hats

BUNGiE

Game-Logic based data corruption

- **Incredibly dangerous, but also hard to catch**
- **Logic change that corrupts player or economy**

- **Example: Currency cap drastically reduced**
  - Responsible for Destiny's longest-ever unplanned downtime, as we executed a full game rollback

BUNGiE

Corruption Mitigations

- Test with real player data
- Offline the game immediately!
- Have a rollback runbook
- Investigate options for fast-recovery

BUNGiE

Data Store performance degradation

- Data growth can cause big performance shifts
- New call patterns create new pressures
- One slow query can cause others to slow too
- Finding the real culprit can be nontrivial

BUNGIE

Performance Degradation Mitigations

- Sign On Queue and Throttle
- Disable optional systems
- Stress testing can catch many issues in advance
- Have instrumentation on the timing of every call
- Alert at dangerous thresholds

BUNGIE

Retry-based death spiral

- Systems will tend to retry when failures occur
- This can be automatic, or player driven
- Unexpected extra load from retries can cause systems to topple over... causing more retries
- Requests queuing can timeout before reaching the front of the queue... causing more retries

BUNGiE

Death Spiral mitigations

- Simulate failures during stress testing
- If Death-spiraling – Offline, and slow ramp population
- Where possible, avoid retries
- Prefer retry systems with backoff and Jitter
- Prefer short queues with rapid-rejection

BUNGIE

Key Takeaway #5

- **Understand your failure space and have response runbooks ready**

BUNGiE

The Future of Services?

BUNGiE

The Future

- Bungie Services not Destiny Services
- Support higher concurrencies
- Further leverage the cloud
- Increase iteration and deployment speeds
- Scale the team to meet the challenge

BUNGIE

In summary

1. Sign on Queues and Throttles are one of your first and best tools to handle and prevent Service issues
2. Whenever possible, launch new systems side-by-side with old systems, and slowly cut over
3. Understand the real reliability requirements for your systems and be skeptical of 100% targets
4. Invest in making it as easy as possible for your engineers to do the right thing
5. Understand your failure space and have response runbooks ready

BUNGiE

Thank You!!

http://bungie.net/careers

http://www.linkedin.com/in/michael-williams-engineer

Bonus Slides - Ignore

BUNGiE

Hardware failure

- **This can happen at any layer of your stack at any time**
  - Network gear
  - Hard drives
  - Key servers
  - ISPs
- **Can cause very unexpected states**
  - Network partitions
  - Significant latency increases
  - Intermittent request failures
- **Soft failures are often much worse than full hardware failures**
  - A slow hard drive is significantly worse than one that stops completely

BUNGiE

Hardware failure mitigations

- **Build out hardware redundantly where possible**
  - And regularly test the failover systems
- **Track metrics on hardware performance, not just simple health**
- **When possible, disable optional systems while working around a hardware failure.**

BUNGiE

Certificate issues

- **Your services will likely involve a huge number of certificates**
  - Cloud certs, signing certs, publisher certs, encryption certs, etc.
- **Any one of these expiring can immediately cause whole critic Atal elements of your service to fail**
  - And certificates are rarely tied to optional elements of your services
- **Certificates expire all the time!**

BUNGiE

Certificate issue mitigations

- **Track your certificates**
- **Use multiple alert methods (including nag-mails) when certs are a few months out from expiration**
- **Update certificates early**
  - Because renewing certificates often requires partner conversations, a renewal cycle can become a big risk if you let it run down to the wire
- **Invest in certificate tracking solutions.**
- **Consider shorter renewals for certificates!**
  - A certificate that expires once every 5 years likely means the folks who installed it last time aren't in the same role today!