Twitch Plays Pokémon: Twitch's Chat Architecture

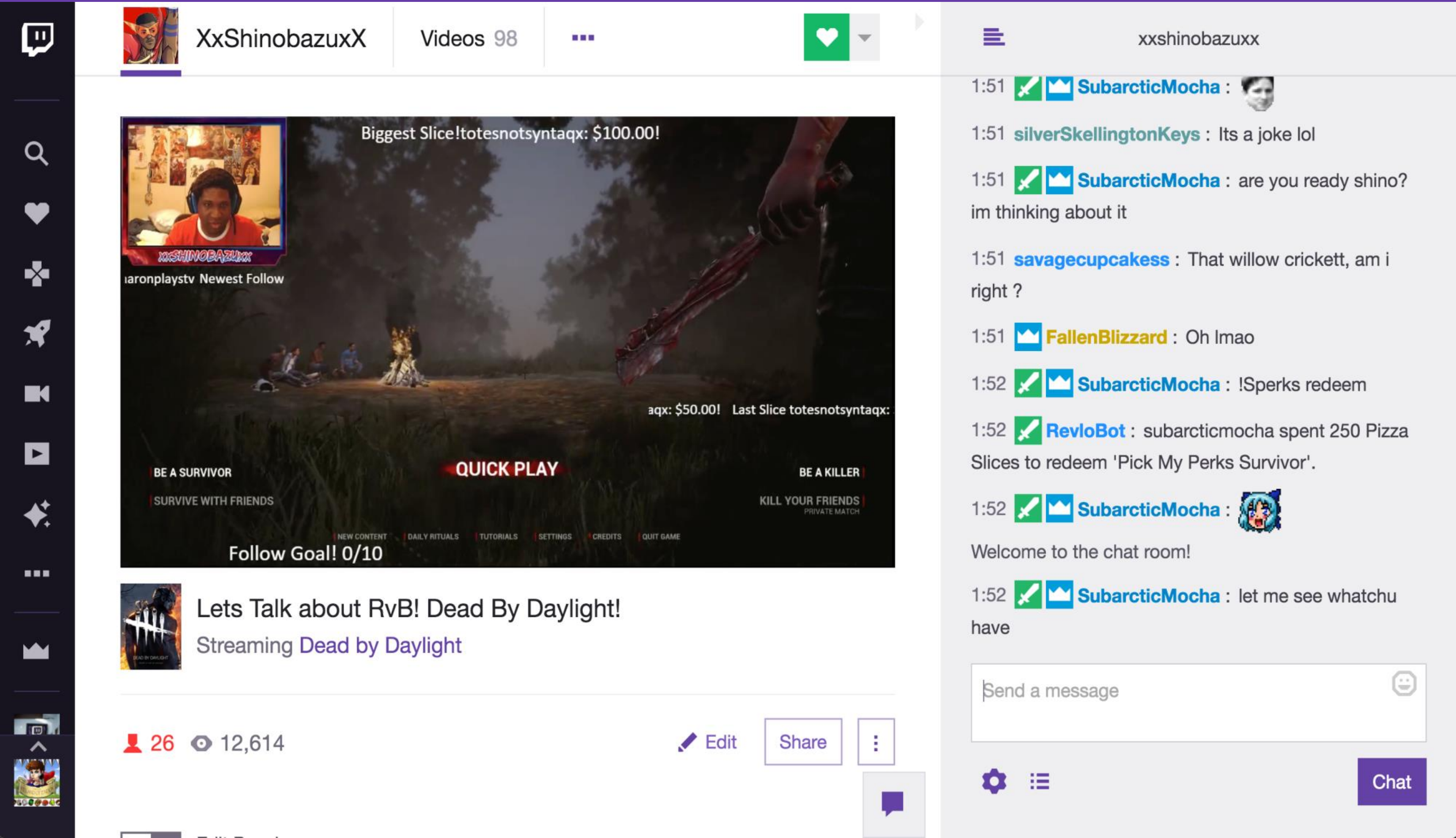
John Rizzo
Sr Software Engineer

www.twitch.tv

About Me

www.twitch.tv

Twitch Introduction



Lets Talk about RvB! Dead By Daylight!
Streaming Dead by Daylight

1:51 SubarcticMocha :

1:51 silverSkellingtonKeys : Its a joke lol

1:51 SubarcticMocha : are you ready shino? im thinking about it

1:51 savagecupcakess : That willow crickett, am i right ?

1:51 FallenBlizzard : Oh lmao

1:52 SubarcticMocha : !Sperks redeem

1:52 RevloBot : subarcticmocha spent 250 Pizza Slices to redeem 'Pick My Perks Survivor'.

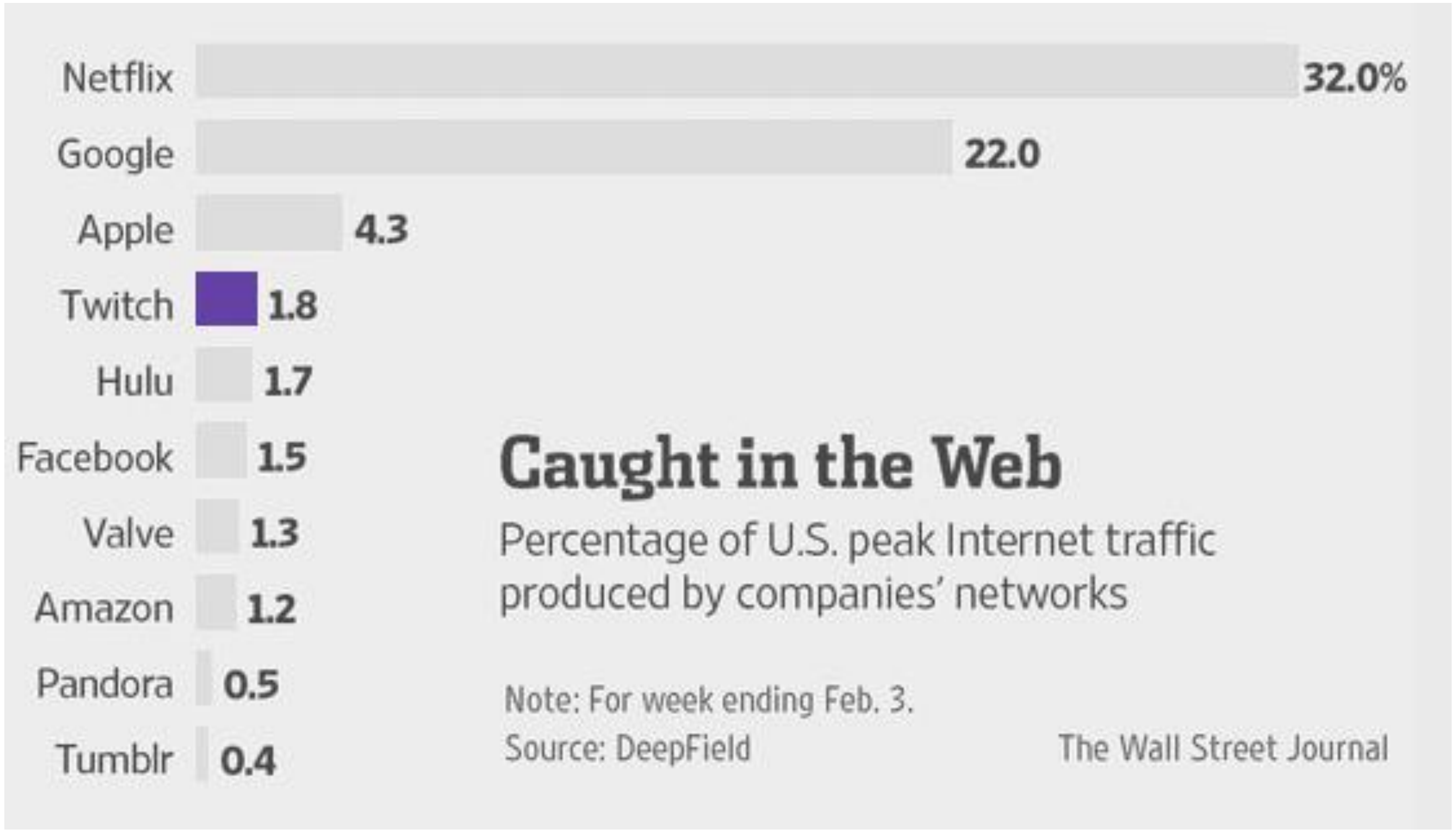
1:52 SubarcticMocha :

Welcome to the chat room!

1:52 SubarcticMocha : let me see whatchu have

www.twitch.tv

Twitch Introduction



Caught in the Web

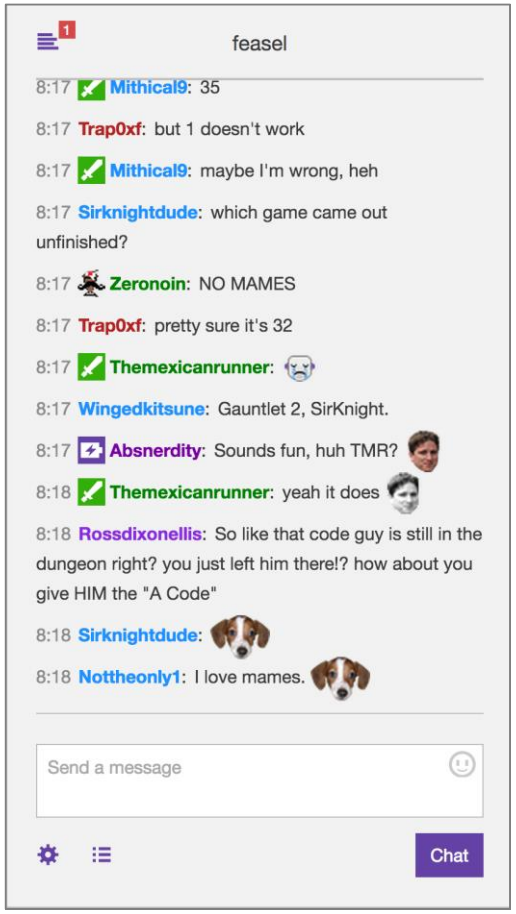
Percentage of U.S. peak Internet traffic produced by companies' networks

Note: For week ending Feb. 3.
Source: DeepField

The Wall Street Journal

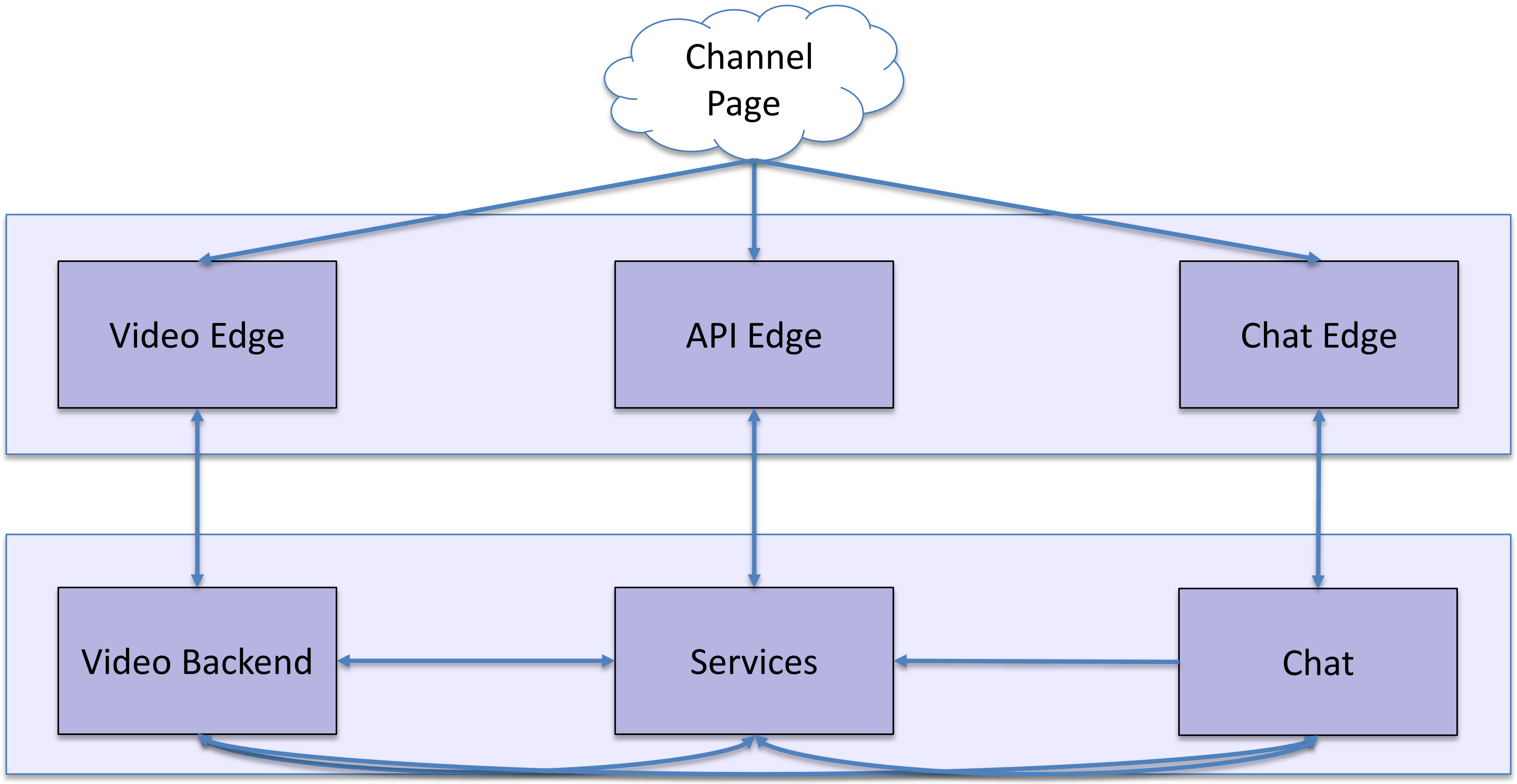
Twitch Introduction

- Over 800k concurrent users
- Tens of BILLIONS of daily messages
- ~10 Servers
- 2 Engineers
- 19 Amp Energy drinks per day

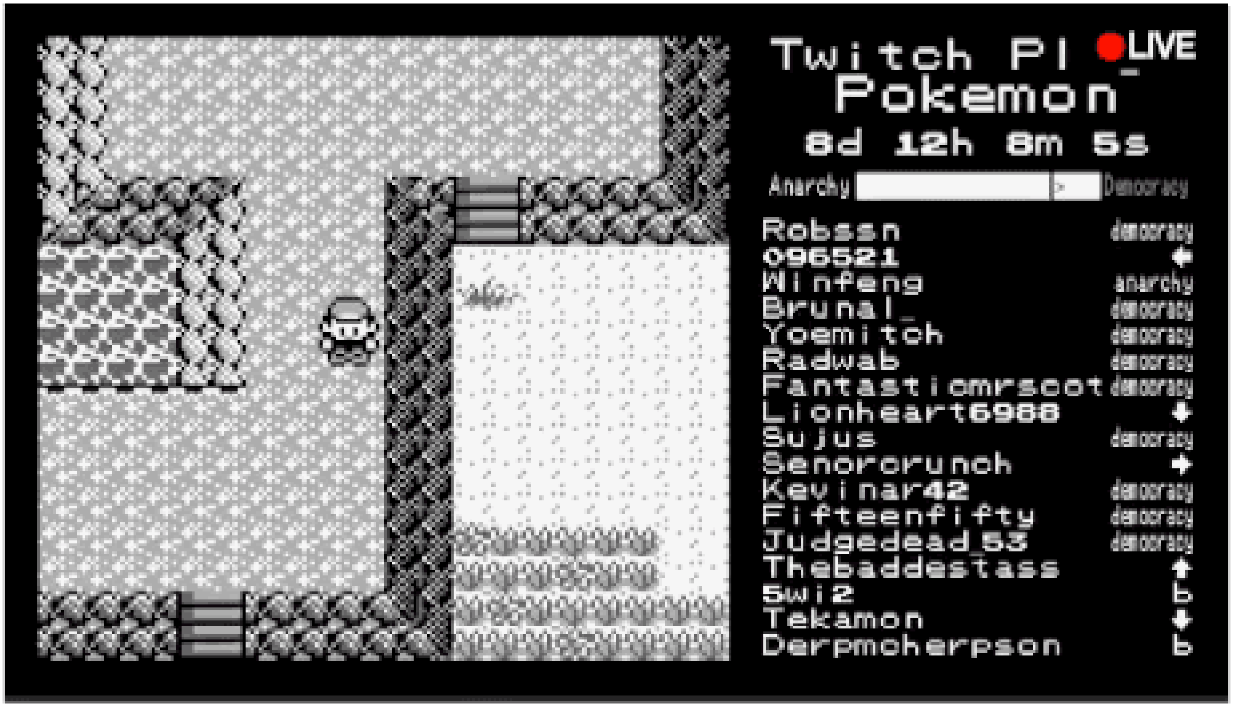

feasel

8:17 Mithical9: 35
8:17 Trap0xf: but 1 doesn't work
8:17 Mithical9: maybe I'm wrong, heh
8:17 Sirknightdude: which game came out unfinished?
8:17 Zeronoin: NO MAMES
8:17 Trap0xf: pretty sure it's 32
8:17 Themexicanrunner:
8:17 Wingedkitsune: Gauntlet 2, SirKnight.
8:17 Absnerdity: Sounds fun, huh TMR?
8:18 Themexicanrunner: yeah it does
8:18 Rossdixonellis: So like that code guy is still in the dungeon right? you just left him there!? how about you give HIM the "A Code"
8:18 Sirknightdude:
8:18 Nottheonly1: I love mames.

Architecture Overview



Twitch Plays Pokémon Strikes!



8d 12h 8m 5s

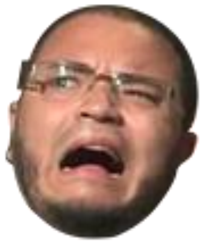
```
Robssn               democracy
096521               ←
Winfeng              anarchy
Brunal_              democracy
Yoemitch             democracy
Radwab               democracy
Fantastiomrscoot     democracy
Lionheart6988        ↓
Sujus                democracy
Senororunch          →
Kevinar42            democracy
Fifteenfifty         democracy
Judgedead_53         democracy
Thebaddestass        ↑
Swi2                 b
Tekamon              ↓
Derpmoherpson        b
```

www.twitch.tv

Public Reaction

Media outlets have described the proceedings of the game as being "mesmerizing", "miraculous" and "beautiful chaos", with one viewer comparing it to "watching a car crash in slow motion" - Wikipedia

“Some users are hailing the development as part of internet history…” – BBC



– Twitch Chat Team

Twitch Plays Pokémon Strikes!

10:33pm, February 14: TPP hits 5k concurrent viewers

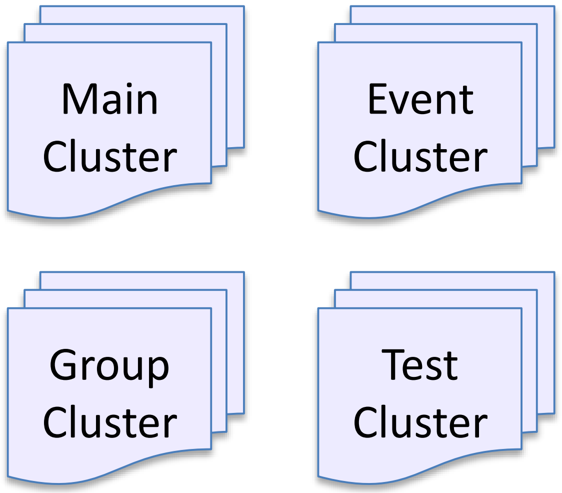
9:42am, February 15: TPP hits 20k

8:21pm, February 16: Time to take preemptive action

Chat Server Organization

We separate servers into several clusters

- Robust in the face of failure
- Can tune each cluster for its specific purpose



Twitch Plays Pokémon Strikes!

8:21pm, February 16: Move TPP onto the event cluster

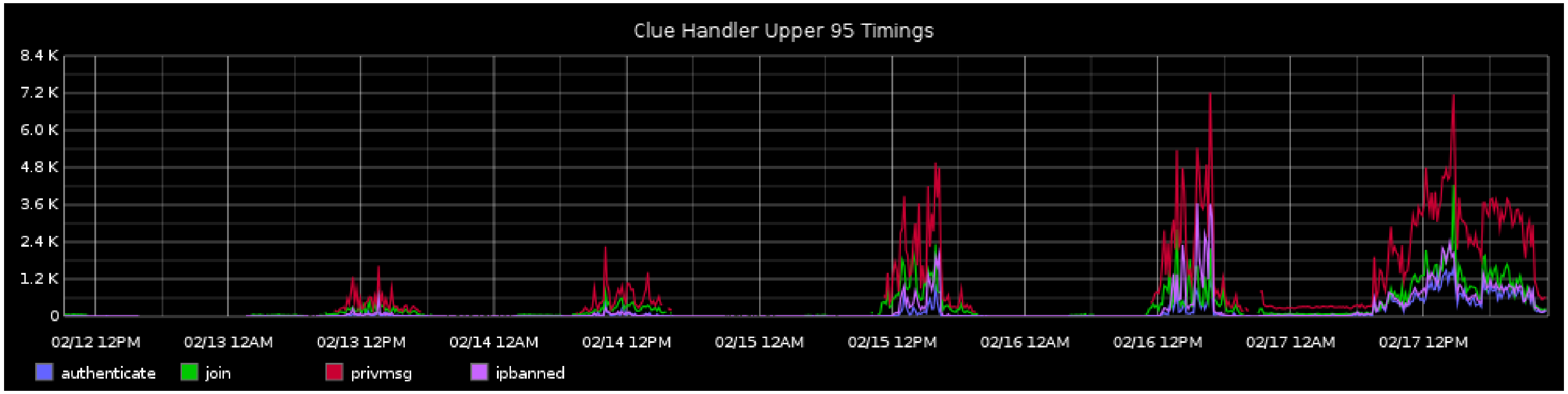
www.twitch.tv

Twitch Plays Pokémon Strikes!

5:43pm, February 16: TPP hits 50k users, chat system starts to show signs of stress (but…it's on the event cluster!)

8:01am, February 17: Twitch chat engineers panic, rush to the office

9:35am, February 17: Investigation begins

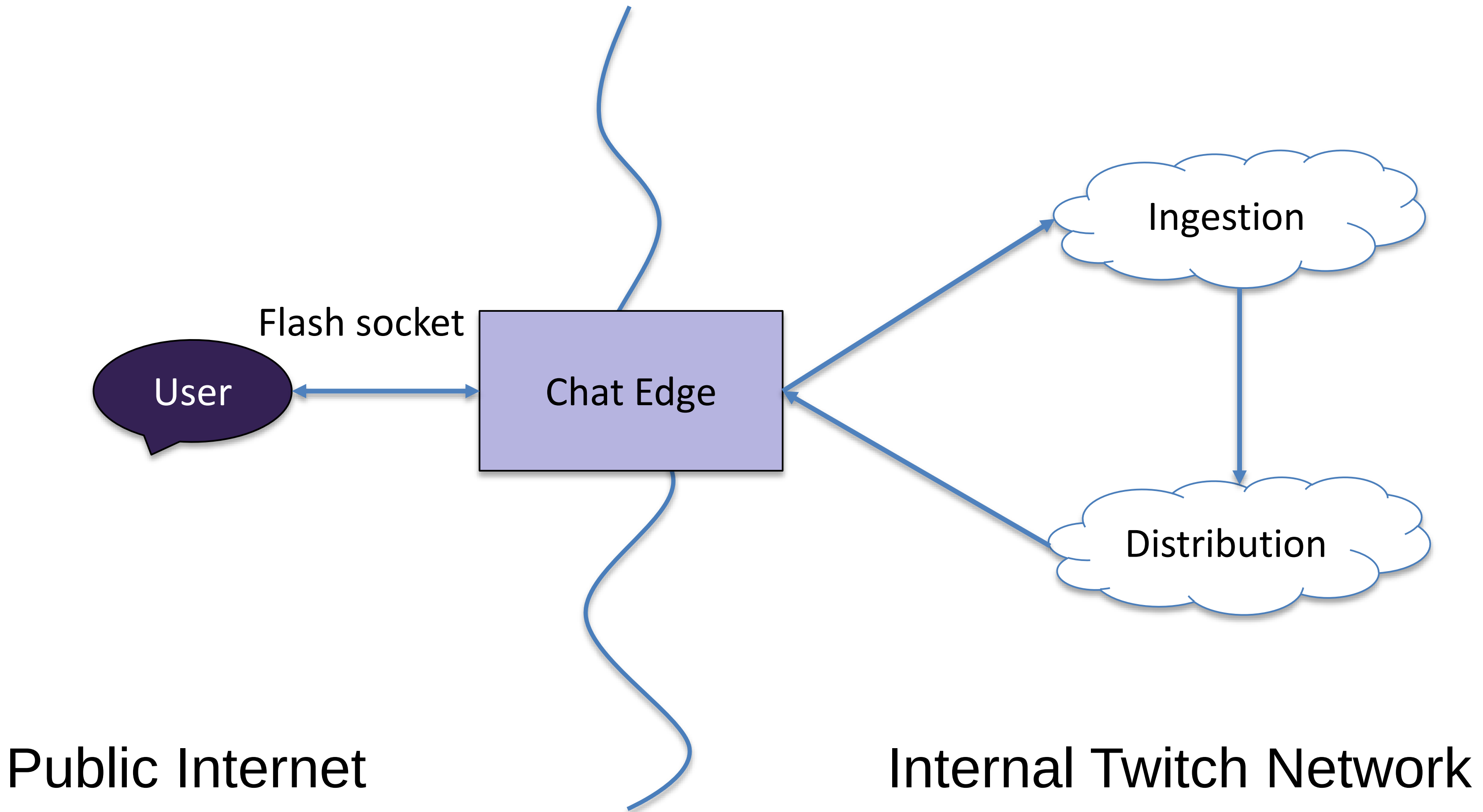


Solving Twitch Plays Pokémon

Debugging principle: Start investigating upstream, move downstream as necessary

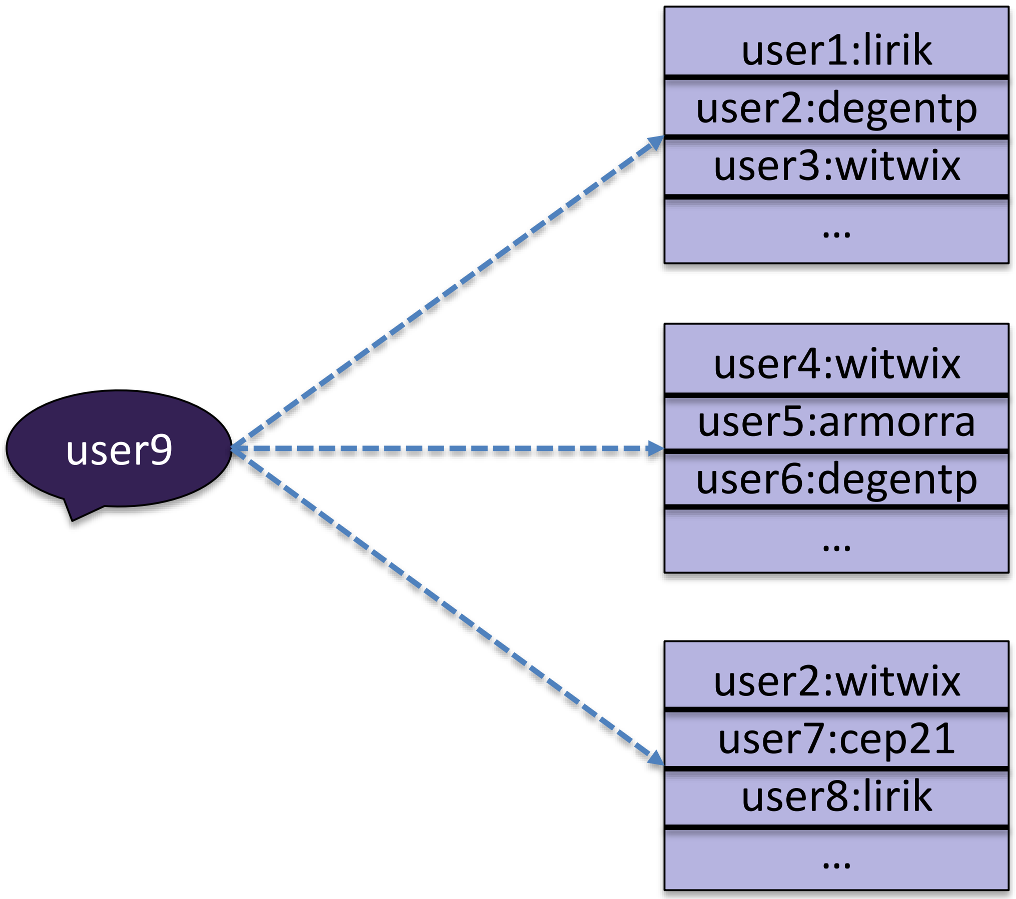
Chat Architecture Overview

Edge Server: Sits at the "edge" of the public internet and Twitch's internal network

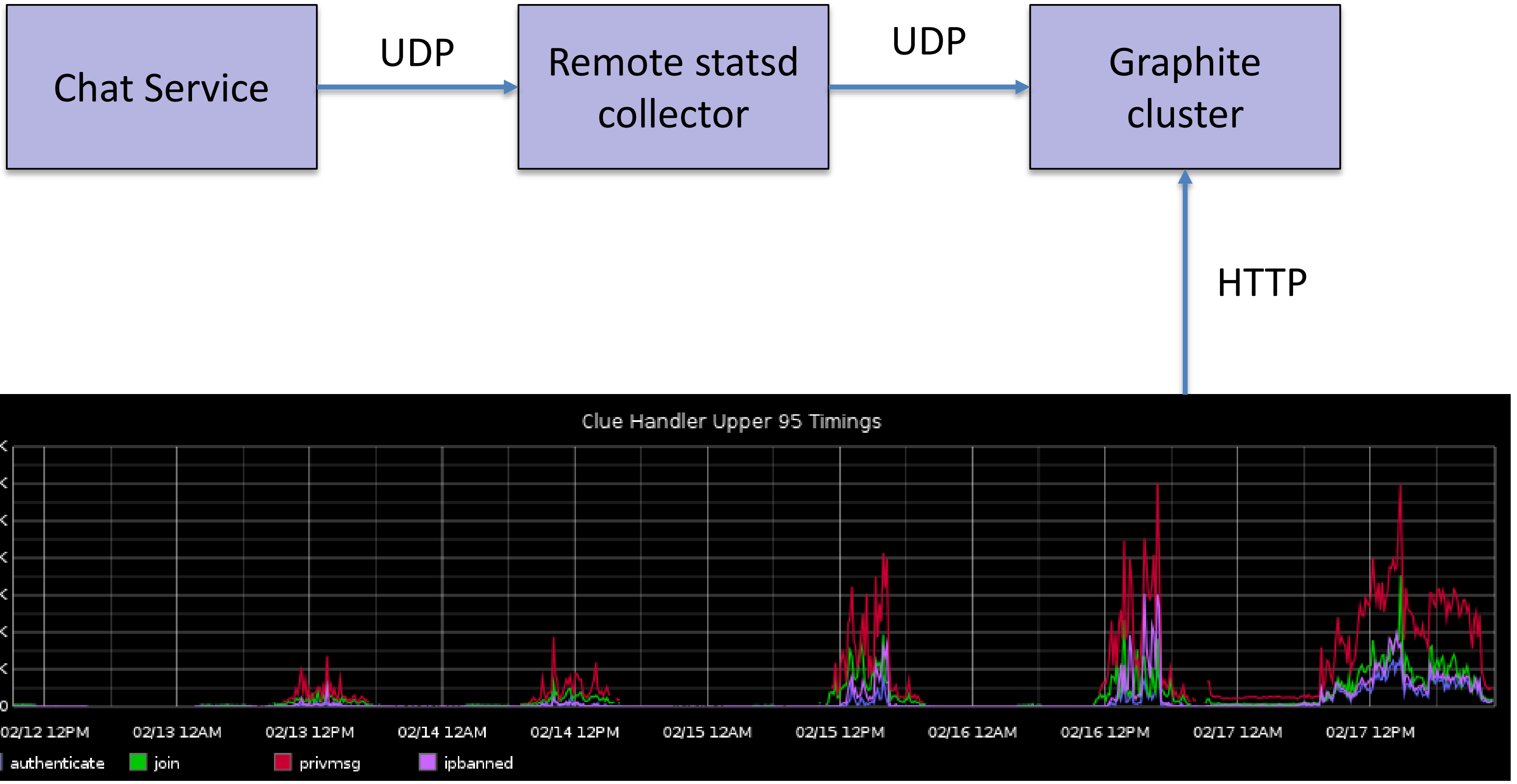


Solving Twitch Plays Pokémon

Any user:room:edge tuple is valid



A Note on Instrumentation



www.twitch.tv

Solving Twitch Plays Pokémon

So let’s take a look at our edge server logs…

```
Feb 18 06:54:04 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
Feb 18 06:54:04 tmi_edge: [clue] Message successfully sent to #degentp
Feb 18 06:54:04 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
Feb 18 06:54:04 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
Feb 18 06:54:04 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
Feb 18 06:54:04 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
Feb 18 06:54:04 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
Feb 18 06:54:05 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
Feb 18 06:54:04 tmi_edge: [clue] Message successfully sent to #paragusrants
```

Solving Twitch Plays Pokémon

Let’s dissect one of these log lines

```
Feb 18 06:54:04 chat_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
```

Solving Twitch Plays Pokémon

Let’s dissect one of these log lines

```
Feb 18 06:54:04 chat_edge: [clue] Timed out (after 5s) while writing request for:
                 privmsg
```

Timestamp - when this action was recorded on the server

Solving Twitch Plays Pokémon

Let’s dissect one of these log lines

```
Feb 18 06:54:04 chat_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
```

Server - the name of the server that is generating this log file

Solving Twitch Plays Pokémon

Let’s dissect one of these log lines

```
Feb 18 06:54:04 chat_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
```

Remote service - the name of the service that edge is connecting to

Solving Twitch Plays Pokémon

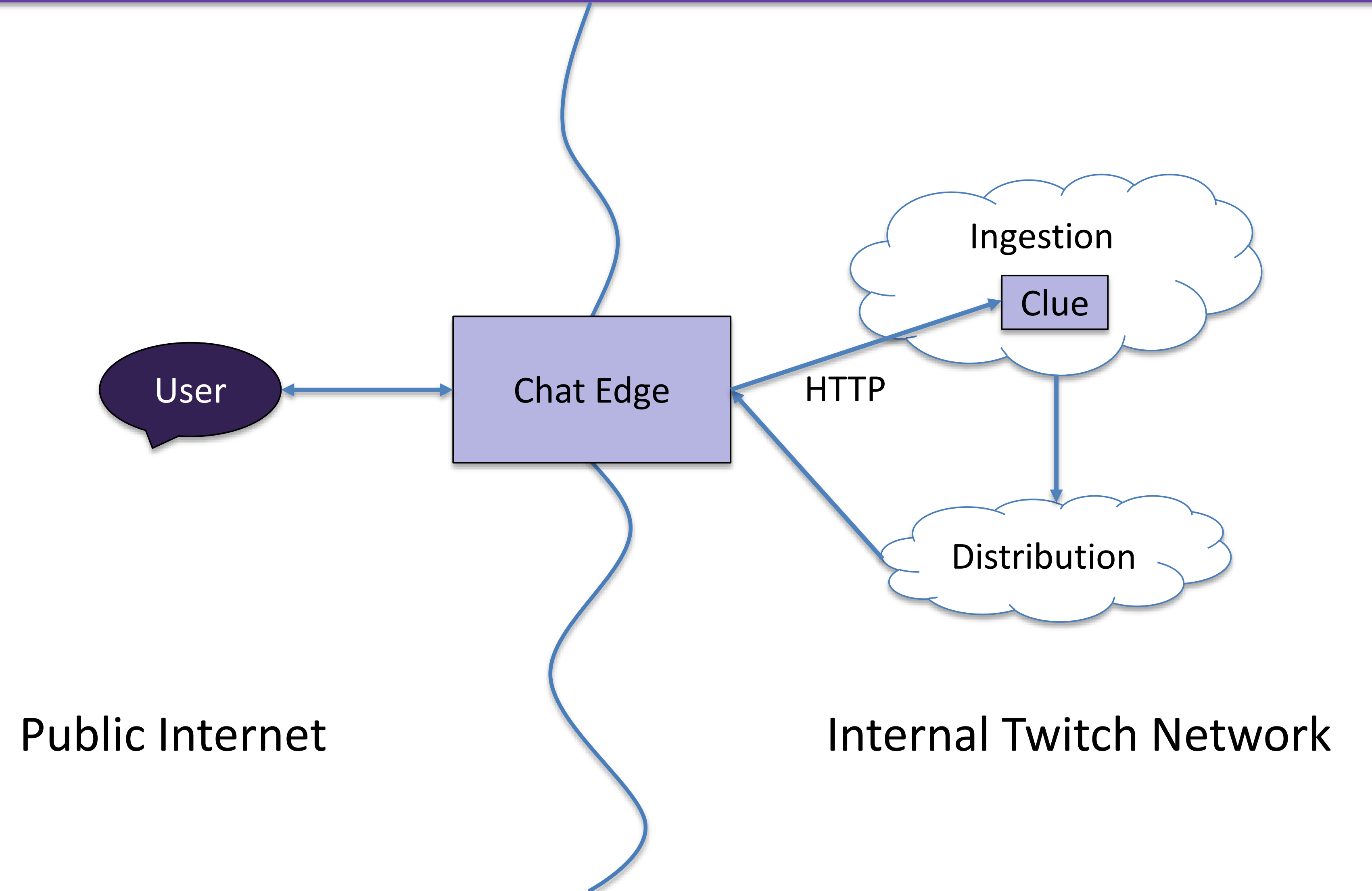
Let’s dissect one of these log lines

```
Feb 18 06:54:04 chat_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
```

Event detail

Why did the clue service take so long to respond? Also, what is the clue service?

Message Ingestion



Solving Twitch Plays Pokémon

Let’s dissect one of these log lines

```
Feb 18 06:54:04 tmi_edge: [clue] Timed out (after 5s) while writing request for:
                          privmsg
```

Clue server took longer than 5 seconds to process this message.

Why?

Solving Twitch Plays Pokémon

Clue logs…

```
Feb 18 06:54:04 chat_clue: WARNING:tornado.general:Connect error on fd 10:
                      ECONNREFUSED
Feb 18 06:54:04 chat_clue: WARNING:tornado.general:Connect error on fd 15:
                      ECONNREFUSED
Feb 18 06:54:05 chat_clue: WARNING:tornado.general:Connect error on fd 9:
                 ECONNREFUSED
Feb 18 06:54:05 chat_clue: WARNING:tornado.general:Connect error on fd 18:
                      ECONNREFUSED
```

Not very useful…but we get some info. Clue's connections are being refused.

Which machine is clue failing to connect to? Why?

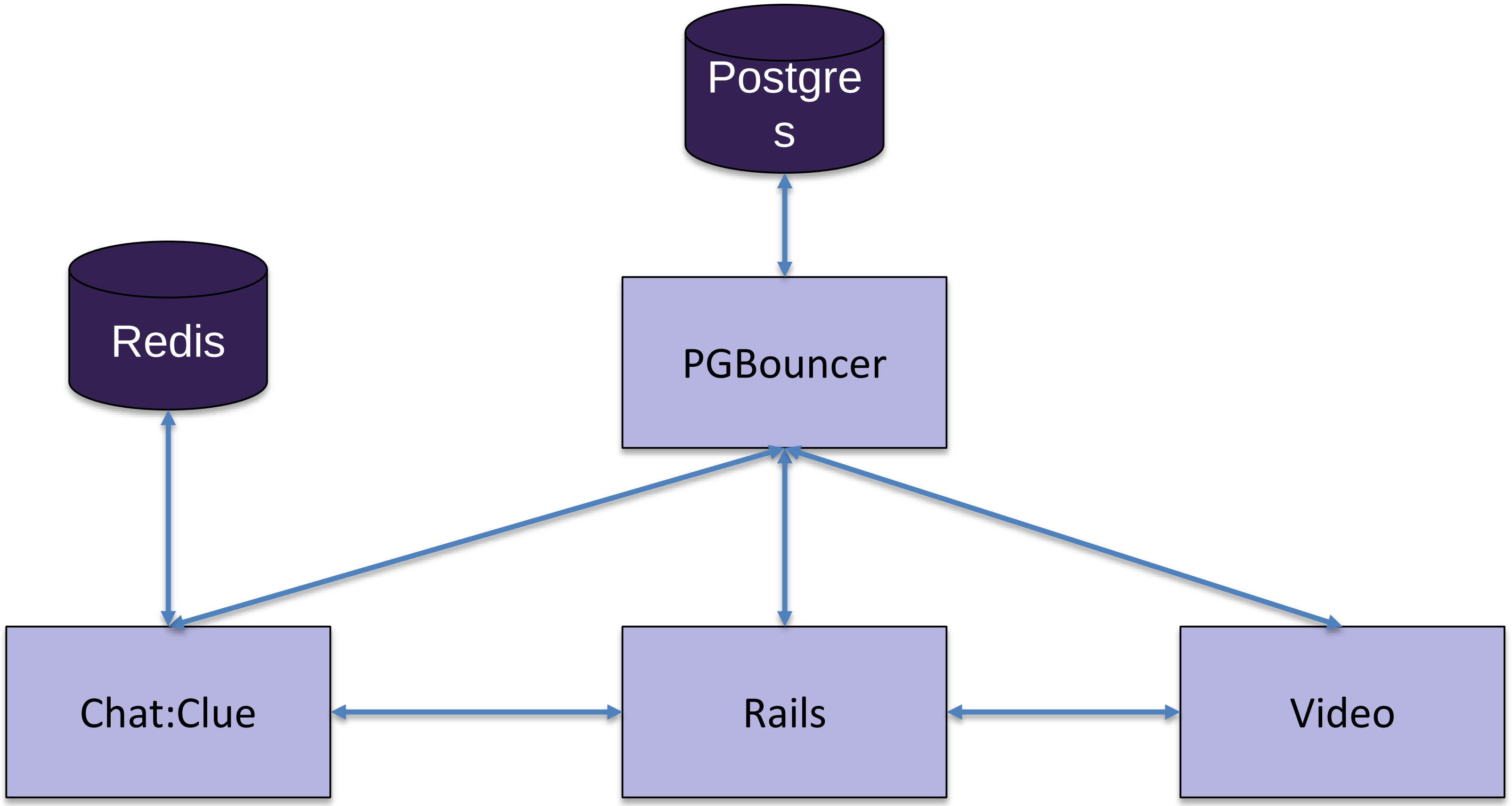
Investigating Clue

Let’s take a step back…these errors are happening on both main AND the event clusters. Why?

Are there common services or dependencies?

- Databases (store chat color, badges, bans, etc)
- Cache (to speed up database access)
- Services (authentication, user data, etc)

Investigating Clue



Investigating Clue

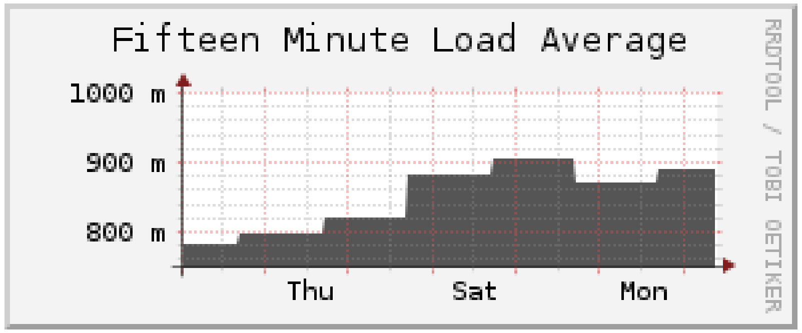
Can rule out databases and services – rest of site is functional

Let’s look closer at our cache – this one is specific to chat servers

Investigating Redis

Redis: where do we start investigating?

Strategy: start high-level then drill down



Redis server isn’t being stressed very hard

Investigating Redis

Let’s look at how Clue is **using** Redis…

Clue Configuration

```
DB_SERVER=db.internal.twitch.tv
DB_NAME=twitch_db
DB_TIMEOUT=1s
CACHE_SERVER=localhost
CACHE_PORT=2000
CACHE_MAX_CONNS=10
CACHE_TIMEOUT=100ms
…
…
```

Clue Configuration

Looks like our whole cache contains only one local instance?

```
DB_SERVER=db.internal.twitch.tv
DB_NAME=twitch_db
DB_TIMEOUT=1s
CACHE_SERVER=localhost
CACHE_PORT=2000
CACHE_MAX_CONNS=10
CACHE_TIMEOUT=100ms
…
…
```

Redis is single-process and single-threaded!

Redis Configuration

```
$ ps aux | grep redis
13909   0.0  0.0  2434840    796 s000  S+ grep redis
```

Redis doesn’t seem to be running locally - what listens on port 2000?

Redis Configuration

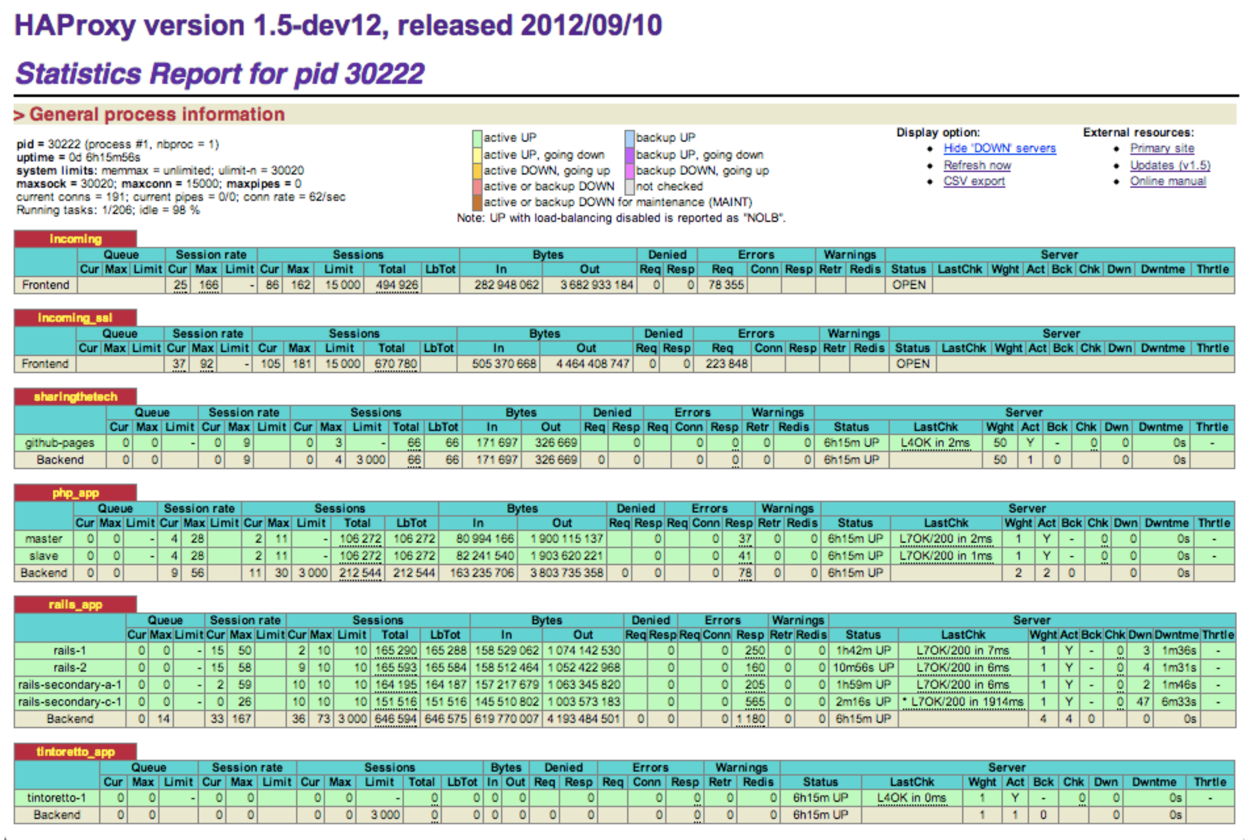
```
$ netstat -lp | grep 2000
tcp 0 0 localhost:2000 *:*  LISTEN   2109/haproxy
```

HAProxy!

www.twitch.tv

HAProxy

- Limits for #connections, requests, etc
- Robust instrumentation



HAProxy version 1.5-dev12, released 2012/09/10

Statistics Report for pid 30222

> General process information

pid = 30222 (process #1, nbproc = 1)
uptime = 0d 6h15m56s
system limits: memmax = unlimited; ulimit-n = 30020
maxsock = 30020; **maxconn** = 15000; **maxpipes** = 0
current conns = 191; current pipes = 0/0; conn rate = 62/sec
Running tasks: 1/206; idle = 98 %

🟩	active UP	🟦	backup UP
🟨	active UP, going down	🟪	backup UP, going down
🟧	active DOWN, going up	🟪	backup DOWN, going up
🟥	active or backup DOWN	⬜	not checked
🟫	active or backup DOWN for maintenance (MAINT)		

Note: UP with load-balancing disabled is reported as "NOLB".

Display option:
- Hide 'DOWN' servers
- Refresh now
- CSV export

External resources:
- Primary site
- Updates (v1.5)
- Online manual

Incoming

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				25	166	-	86	162	15 000	494 926		282 948 062	3 682 933 184	0	0	78 355					OPEN								

incoming_ssl

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				37	92	-	105	181	15 000	670 780		505 370 668	4 464 408 747	0	0	223 848					OPEN								

sharingthetech

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
github-pages	0	0	-	0	9		0	3	-	66	66	171 697	326 669		0		0	0	0	0	6h15m UP	L4OK in 2ms	50	Y	-	0	0	0s	-
Backend	0	0		0	9		0	4	3 000	66	66	171 697	326 669	0	0		0	0	0	0	6h15m UP		50	1	0		0	0s	

php_app

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
master	0	0	-	4	28		2	11	-	106 272	106 272	80 994 166	1 900 115 137		0		0	37	0	0	6h15m UP	L7OK/200 in 2ms	1	Y	-	0	0	0s	-
slave	0	0	-	4	28		2	11	-	106 272	106 272	82 241 540	1 903 620 221		0		0	41	0	0	6h15m UP	L7OK/200 in 1ms	1	Y	-	0	0	0s	-
Backend	0	0		9	56		11	30	3 000	212 544	212 544	163 235 706	3 803 735 358	0	0		0	78	0	0	6h15m UP		2	2	0		0	0s	

rails_app

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
rails-1	0	0	-	15	50		2	10	10	165 290	165 288	158 529 062	1 074 142 530		0		0	250	0	0	1h42m UP	L7OK/200 in 7ms	1	Y	-	0	3	1m36s	-
rails-2	0	0	-	15	58		9	10	10	165 593	165 584	158 512 464	1 052 422 968		0		0	160	0	0	10m56s UP	L7OK/200 in 6ms	1	Y	-	0	4	1m31s	-
rails-secondary-a-1	0	0	-	2	59		10	10	10	164 195	164 187	157 217 679	1 063 345 820		0		0	205	0	0	1h59m UP	L7OK/200 in 6ms	1	Y	-	0	2	1m46s	-
rails-secondary-c-1	0	0	-	0	26		10	10	10	151 516	151 516	145 510 802	1 003 573 183		0		0	565	0	0	2m16s UP	* L7OK/200 in 1914ms	1	Y	-	0	47	6m33s	-
Backend	0	14		33	167		36	73	3 000	646 594	646 575	619 770 007	4 193 484 501	0	0		0	1 180	0	0	6h15m UP		4	4	0		0	0s	

tintoretto_app

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
tintoretto-1	0	0	-	0	0		0	0	-	0	0	0	0		0		0	0	0	0	6h15m UP	L4OK in 0ms	1	Y	-	0	0	0s	-
Backend	0	0		0	0		0	0	3 000	0	0	0	0	0	0		0	0	0	0	6h15m UP		1	1	0		0	0s	

Attribution: Shareholic.com

Redis Configuration

Are we load balancing across many Redis instances?

```
DB_SERVER=db.internal.twitch.tv
DB_NAME=twitch_db
DB_TIMEOUT=1s
CACHE_SERVER=localhost
CACHE_PORT=2000
CACHE_MAX_CONNS=10
CACHE_TIMEOUT=100ms
…
…
```

Redis Configurtaion

Are we load balancing across many Redis instances?

```
class twitch::haproxy::listeners::chat_redis (
    $haproxy_instance = ‘chat-backend',
    $proxy_name     = 'chat-redis',
    $servers        = [
        'redis2.internal.twitch.tv:6379',
    ],
    …
    …
    …
```

www.twitch.tv

Redis Configurtaion

Are we load balancing across many Redis instances?

```
class twitch::haproxy::listeners::chat_redis (
  $haproxy_instance = ’chat-backend',
  $proxy_name     = 'chat-redis',
  $servers       = [
    'redis2.internal.twitch.tv:6379',
  ],
  …
  …
  …
```

We are **not** load balancing across several instances

Investigating Redis

Let’s take a look at the Redis box…

```
$ top
Tasks: 281 total,   1 running, 311 sleeping,   0 stopped,   0 zombie
Cpu(s):  10.3%us,  10.5%sy,  0.0%ni, 95.4%id,  0.0%wa,  0.0%hi,
Mem:  24682328k total,6962336k used, 17719992k free,    13644k buffers
Swap:  7999484k total,       0k used,  7999484k free,  4151420k cached

  PID  PR  NI  VIRT  RES   SHR   S  %CPU %MEM   TIME+   COMMAND
26109  20   0 76048  128m  1340  S    99  0.2   6133:28 redis-server
 3342  20   0  9040  1320   844  R     2  0.0   0:00.01 top
    1  20   0 90412  3920  2576  S     0  0.0 103:45.82 init
    2  20   0     0     0     0  S     0  0.0   0:05.17 kthreadd
```

Investigating Redis

Redis is unsprisingly maxing out the CPU

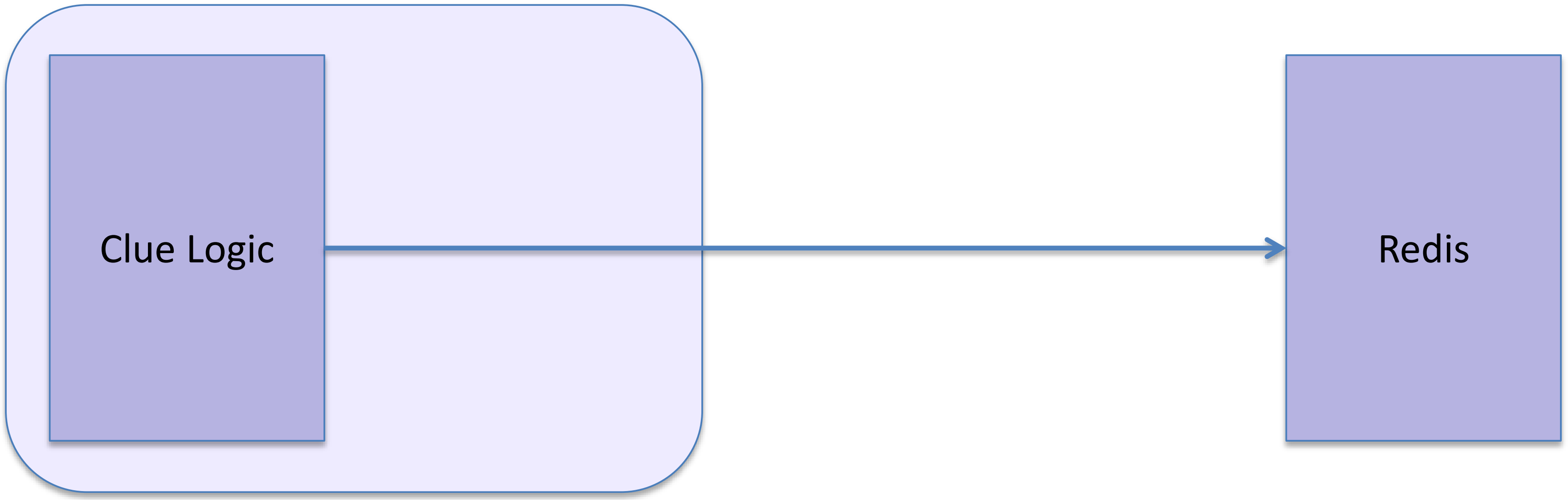
```
$ top
Tasks: 281 total,   1 running, 311 sleeping,   0 stopped,   0 zombie
Cpu(s):  10.3%us,  10.5%sy,  0.0%ni, 95.4%id,  0.0%wa,  0.0%hi
Mem:  24682328k total,6962336k used, 17719992k free,    13644k buffers
Swap:  7999484k total,       0k used,  7999484k free,  4151420k cached

  PID  PR  NI  VIRT  RES   SHR   S  %CPU %MEM   TIME+   COMMAND
26109  20   0 76048  128m  1340  S    99  0.2   6133:28 redis-server
 3342  20   0  9040  1320   844  R     2  0.0   0:00.01 top
    1  20   0 90412  3920  2576  S     0  0.0 103:45.82 init
    2  20   0     0     0     0  S     0  0.0   0:05.17 kthreadd
```

Redis Optimization Options?

- Optimize Redis at the application-level
- Distribute Redis

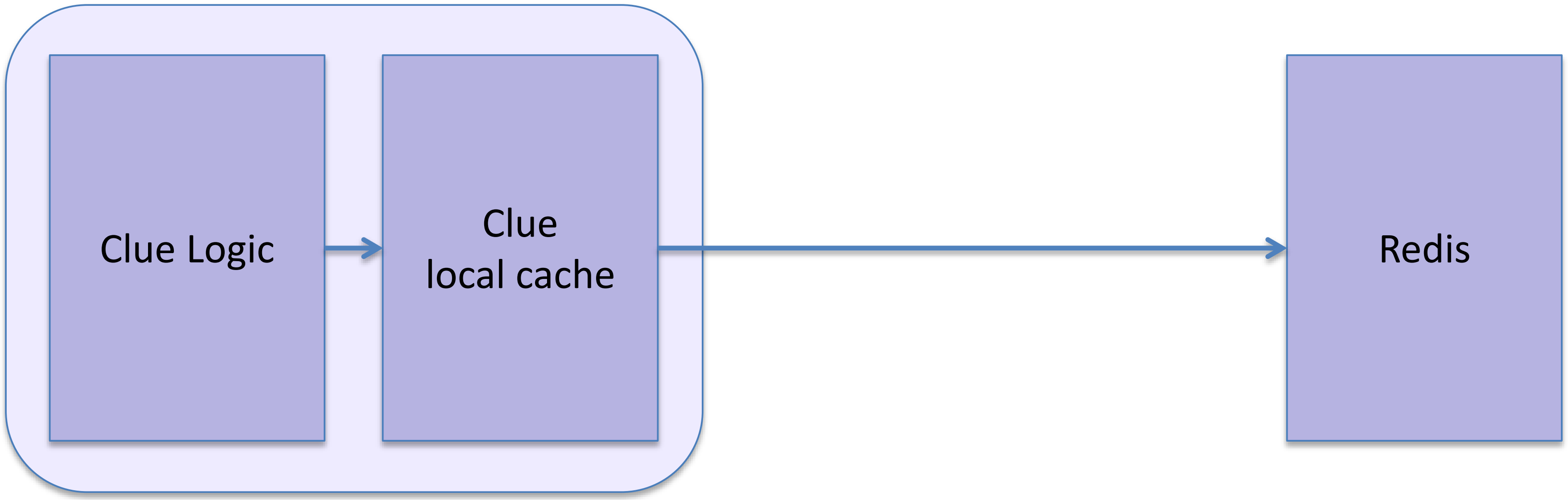
Redis Optimization Options?

2:00pm: Smarter caching?



Redis Optimization Options?

2:00pm: Smarter caching?



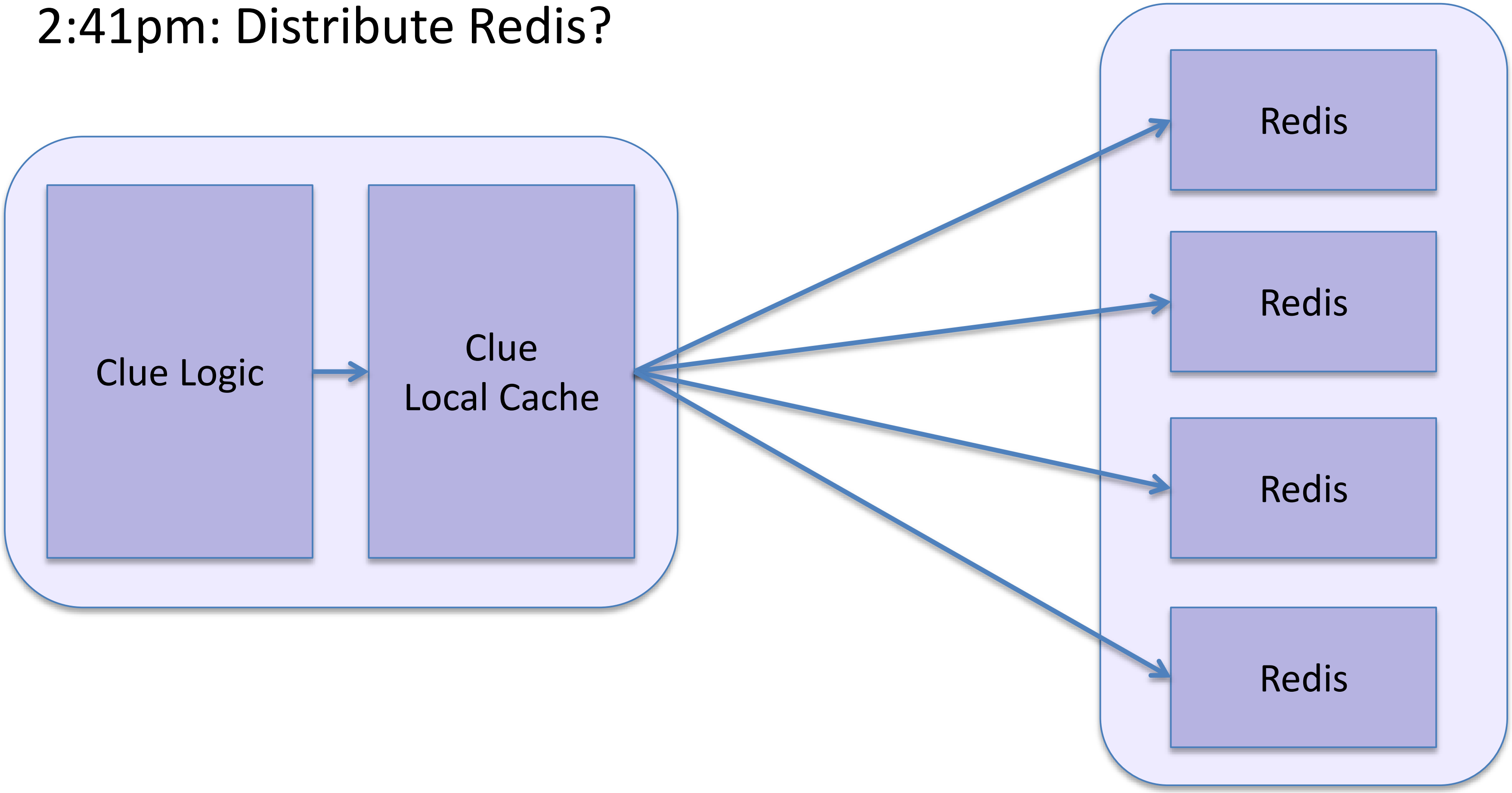
www.twitch.tv

Redis Optimization Options?

- 2:23pm: There are some challenges (cache coherence, implementation difficulty)
- Is there any low-hanging fruit?
- Yes! Rate limiting code!
- 2:33pm: Change has little effect...

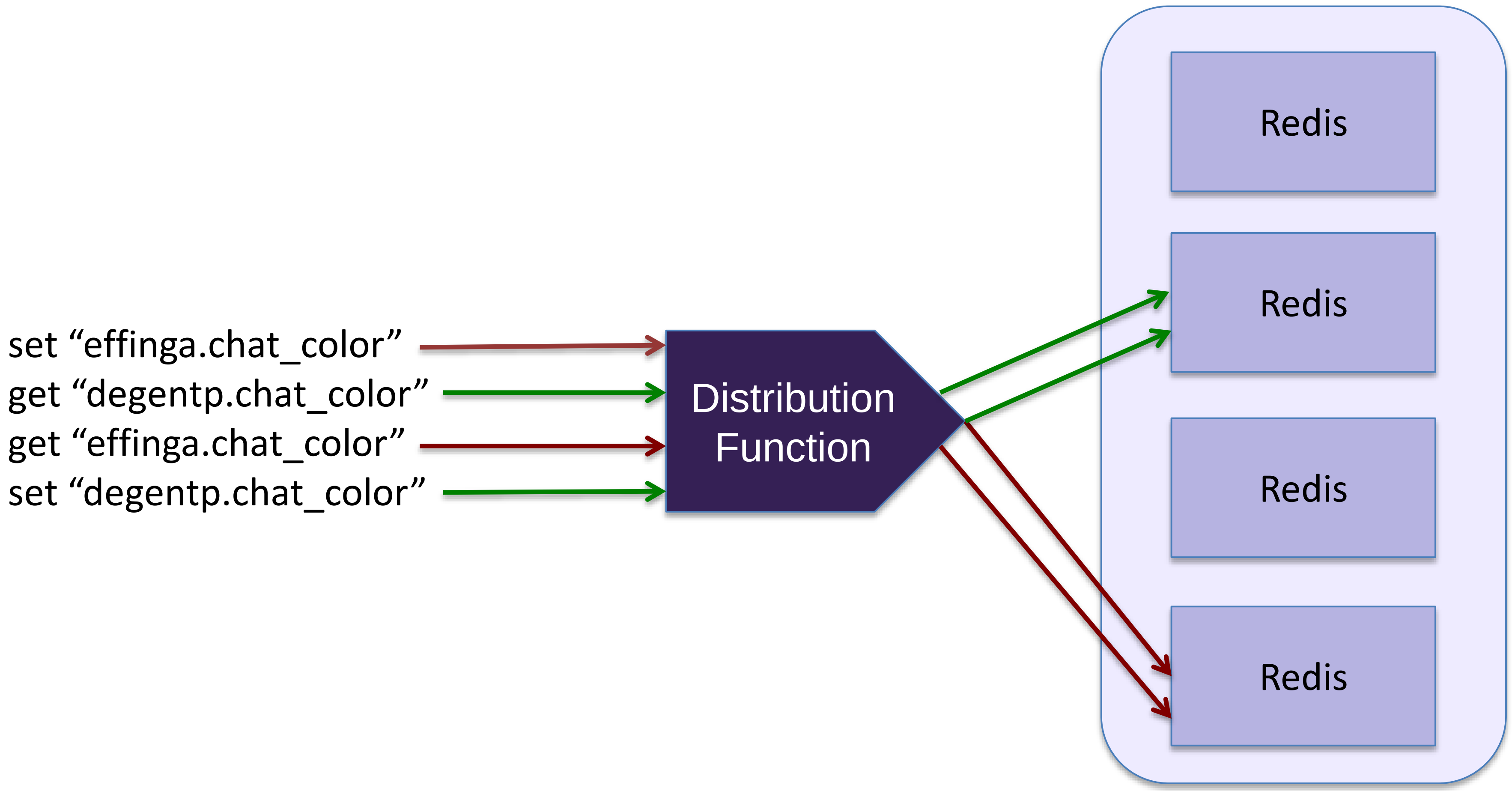
Redis Optimization Options?

2:41pm: Distribute Redis?



Redis Optimization Options?

2:56pm: Yes! Sharding by key!



Distributing Redis

3:03pm: How do we implement this?

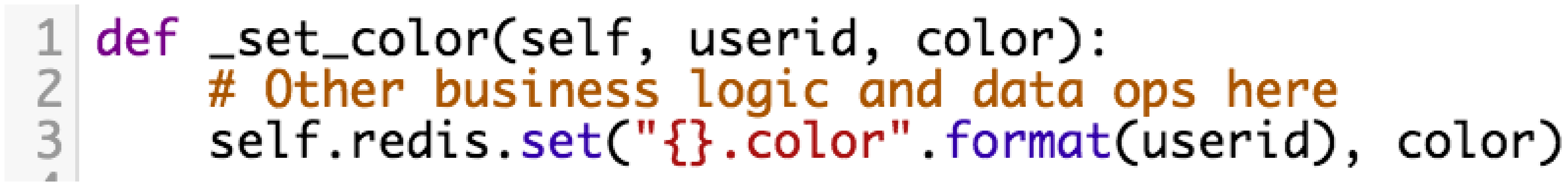
- Puppet configuration changes
  - HAProxy changes
  - Redis deploy changes (copy/paste!)
- Do we need to modify any code?
  - Can we let HAProxy load balance for us?
  - No – LB needs to be aware of Redis protocol
  - Changes required at the application level

Distributing Redis

3:52pm: Code surveyed – Two problematic patterns

Distributing Redis

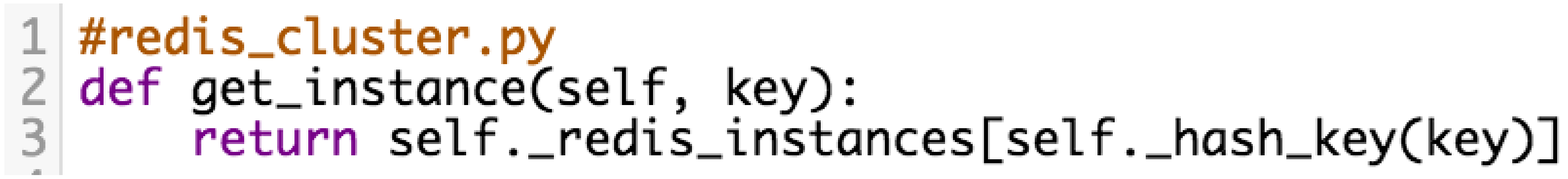
Problematic pattern #1:



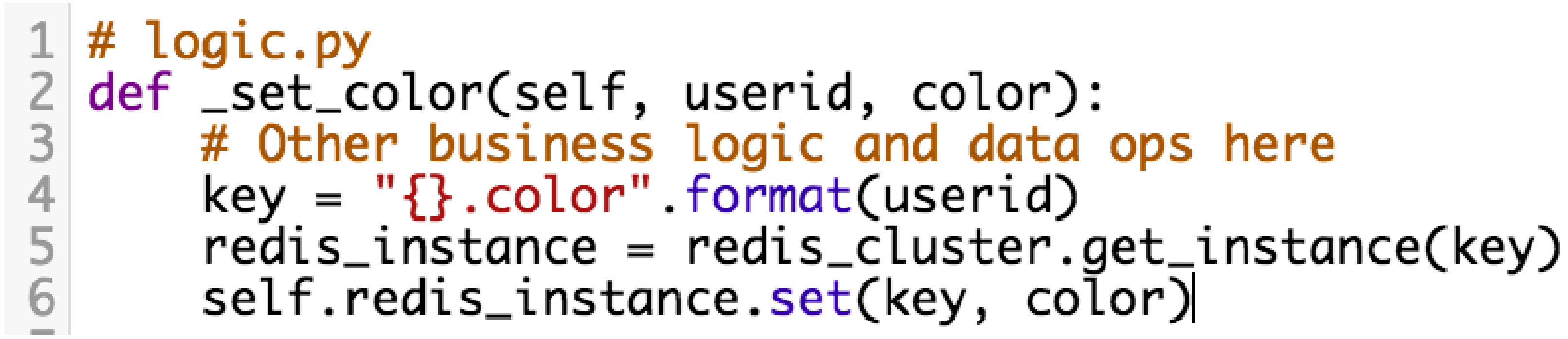
```python
def _set_color(self, userid, color):
    # Other business logic and data ops here
    self.redis.set("{}.color".format(userid), color)
```

Distributing Redis

Problematic pattern #1 solution:



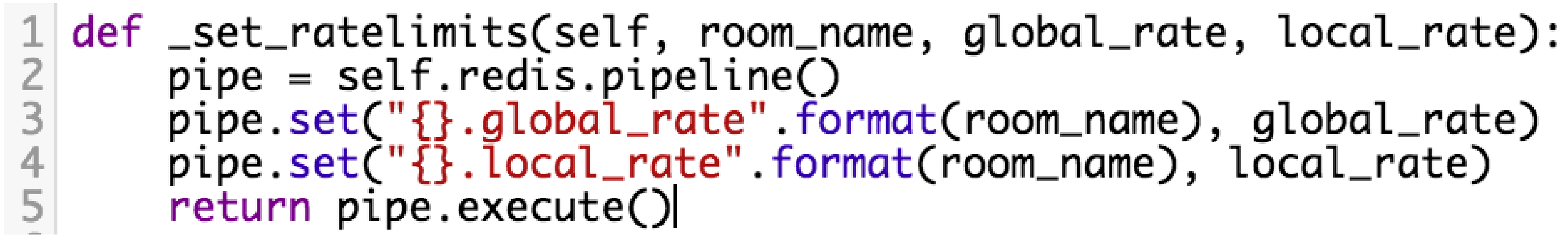
```python
#redis_cluster.py
def get_instance(self, key):
    return self._redis_instances[self._hash_key(key)]
```



```python
# logic.py
def _set_color(self, userid, color):
    # Other business logic and data ops here
    key = "{}.color".format(userid)
    redis_instance = redis_cluster.get_instance(key)
    self.redis_instance.set(key, color)
```

Distributing Redis

Problematic pattern #2:



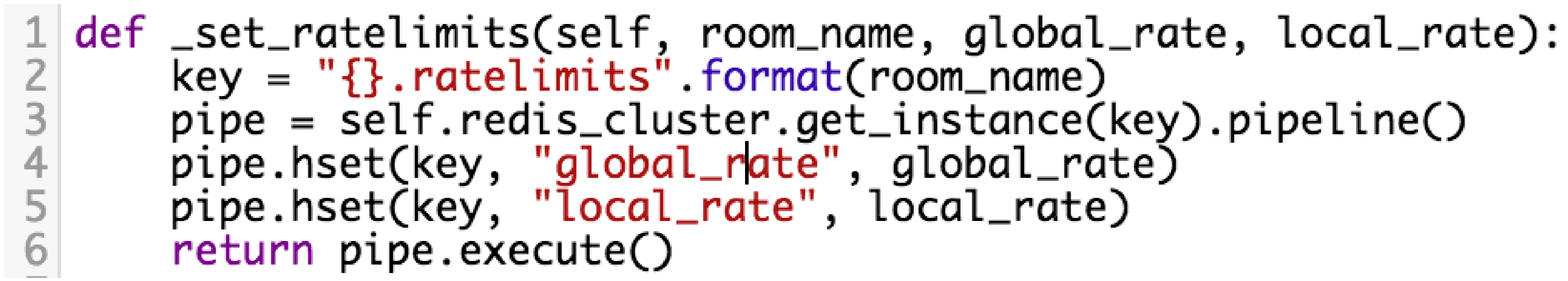
```python
def _set_ratelimits(self, room_name, global_rate, local_rate):
    pipe = self.redis.pipeline()
    pipe.set("{}.global_rate".format(room_name), global_rate)
    pipe.set("{}.local_rate".format(room_name), local_rate)
    return pipe.execute()
```

What if we need these keys in different contexts?

www.twitch.tv

Distributing Redis

Problematic pattern #2 solution:



```python
def _set_ratelimits(self, room_name, global_rate, local_rate):
    key = "{}.ratelimits".format(room_name)
    pipe = self.redis_cluster.get_instance(key).pipeline()
    pipe.hset(key, "global_rate", global_rate)
    pipe.hset(key, "local_rate", local_rate)
    return pipe.execute()
```

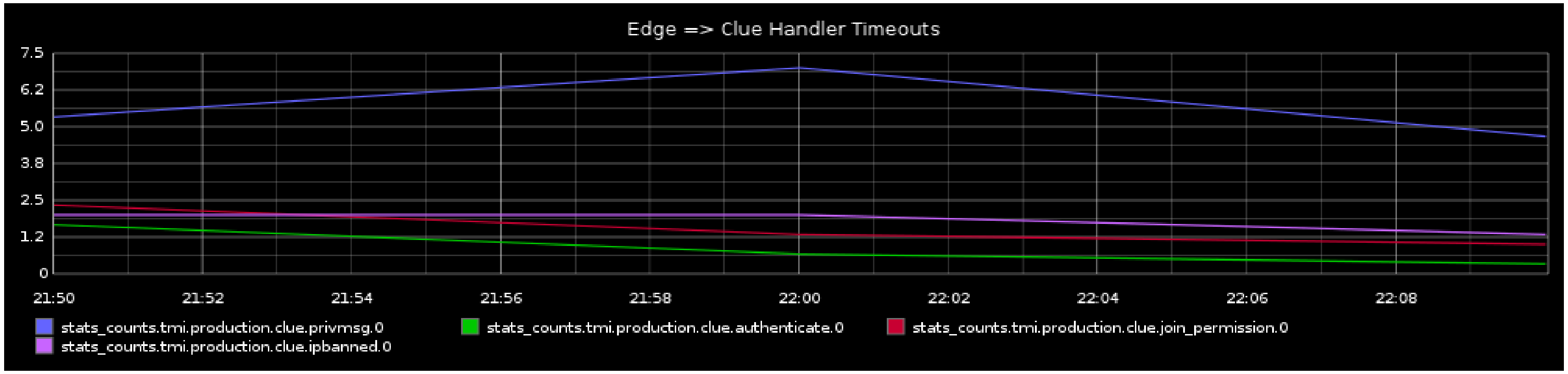
Distributing Redis

7:21pm: Test (fingers crossed)
8:11pm: Deploy cache changes
9:29pm: Deploy chat code changes

www.twitch.tv

Solving Twitch Plays Pokémon

10:10pm: Better, but still bad…



Solving Twitch Plays Pokémon

Let’s use some tools to dig deeper…

```
$ redis-cli -h redis2.internal.twitch.tv -p 6379 INFO
# Clients
connected_clients:3021
client_longest_output_list:0
client_biggest_input_buf:0
blocked_clients:0

$ redis-cli -h redis2.internal.twitch.tv -p 6379 CLIENT LIST |
grep idle | wc -l
     2311
```

Solving Twitch Plays Pokémon

Lots of bad connections

```
$ redis-cli -h redis2.internal.twitch.tv -p 6379 INFO
# Clients
connected_clients:3021
client_longest_output_list:0
client_biggest_input_buf:0
blocked_clients:0

$ redis-cli -h redis2.internal.twitch.tv -p 6379 CLIENT LIST |
grep idle | wc -l
     2311
```

Solving Twitch Plays Pokémon

Let’s grab the pid of one Redis instance

```
$ sudo svstat /etc/service/redis_*
/etc/service/redis_6379: up (pid 26109) 3543 seconds
/etc/service/redis_6380: up (pid 26111) 3543 seconds
/etc/service/redis_6381: up (pid 26113) 3543 seconds
/etc/service/redis_6382: up (pid 26114) 3544 seconds
```

Solving Twitch Plays Pokémon

Let’s grab the pid of one Redis instance

```
$ sudo svstat /etc/service/redis_*
/etc/service/redis_6379: up (pid 26109) 3543 seconds
/etc/service/redis_6380: up (pid 26111) 3543 seconds
/etc/service/redis_6381: up (pid 26113) 3543 seconds
/etc/service/redis_6382: up (pid 26114) 3544 seconds
```

Solving Twitch Plays Pokémon

```
$ sudo lsof -p 26109 | grep chat | cut -d ' ' -f 32 | cut -d ':' -f
2 | sort | uniq -c

   2012 6421->chat-testing.internal.twitch.tv
    121 6421->chat1.internal.twitch.tv
    101 6421->chat3.internal.twitch.tv
   ...
```

Solving Twitch Plays Pokémon

```
$ sudo lsof -p 26109 | grep tmi | cut -d ' ' -f 32 | cut -d ':' -f
2 | sort | uniq -c

   2012 6421->chat-testing.internal.twitch.tv
    121 6421->chat1.internal.twitch.tv
    101 6421->chat3.internal.twitch.tv
   ...
```

Solving Twitch Plays Pokémon

Lesson learned: Testing is bad

Solving Twitch Plays Pokémon

11:12pm: Shut off testing cluster completely

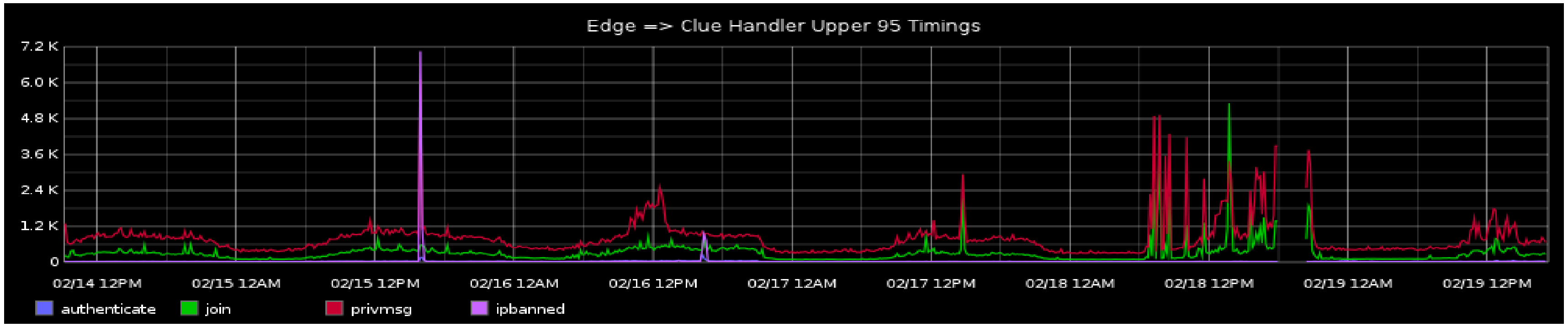
www.twitch.tv

Twitch Plays Pokémon is Solved(?)

Users can connect to chat again!

Users can send messages again!

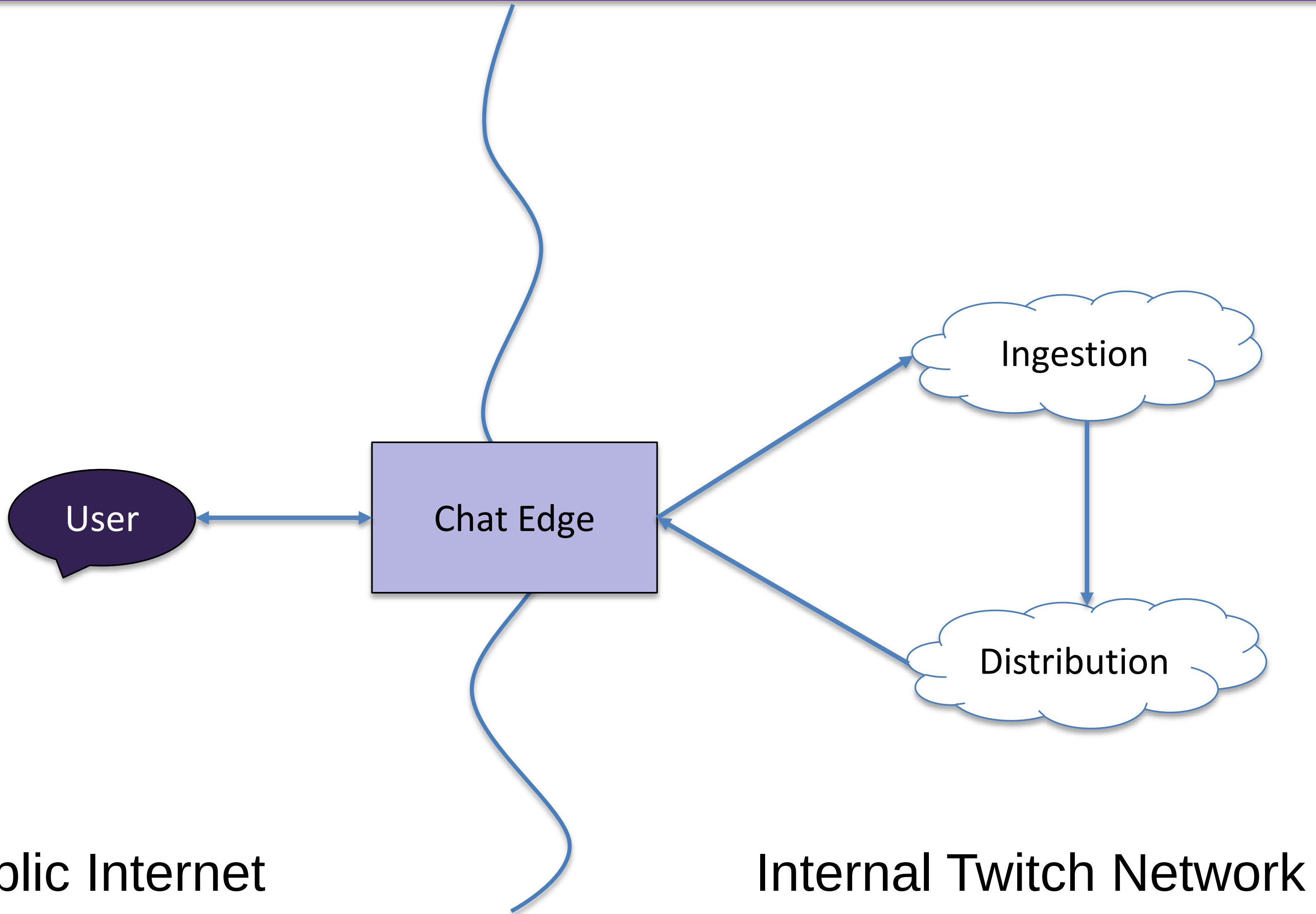
Chat team leaves the office at 11:31pm



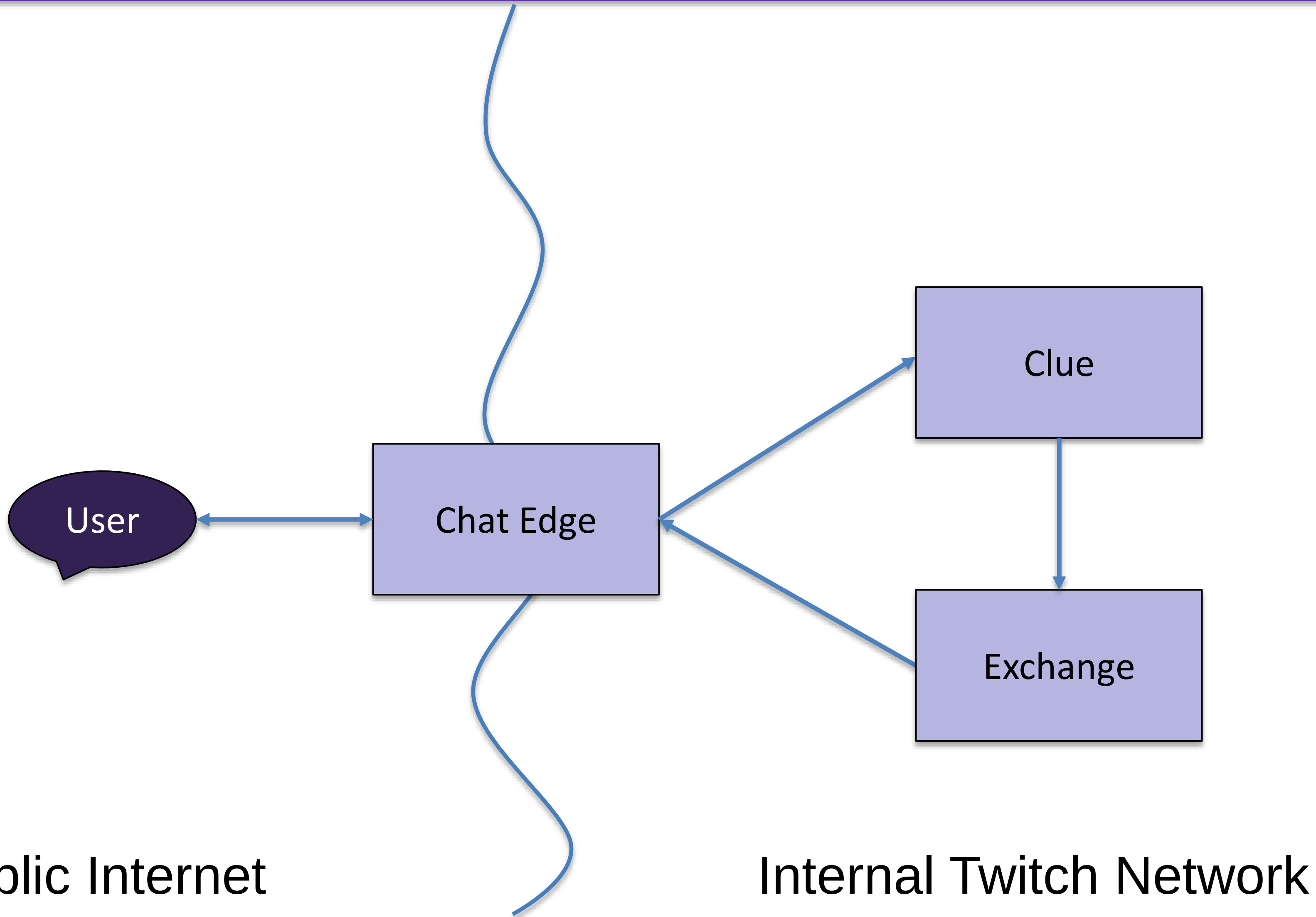
A New Bug Arises

Complaints that users don’t see the same messages

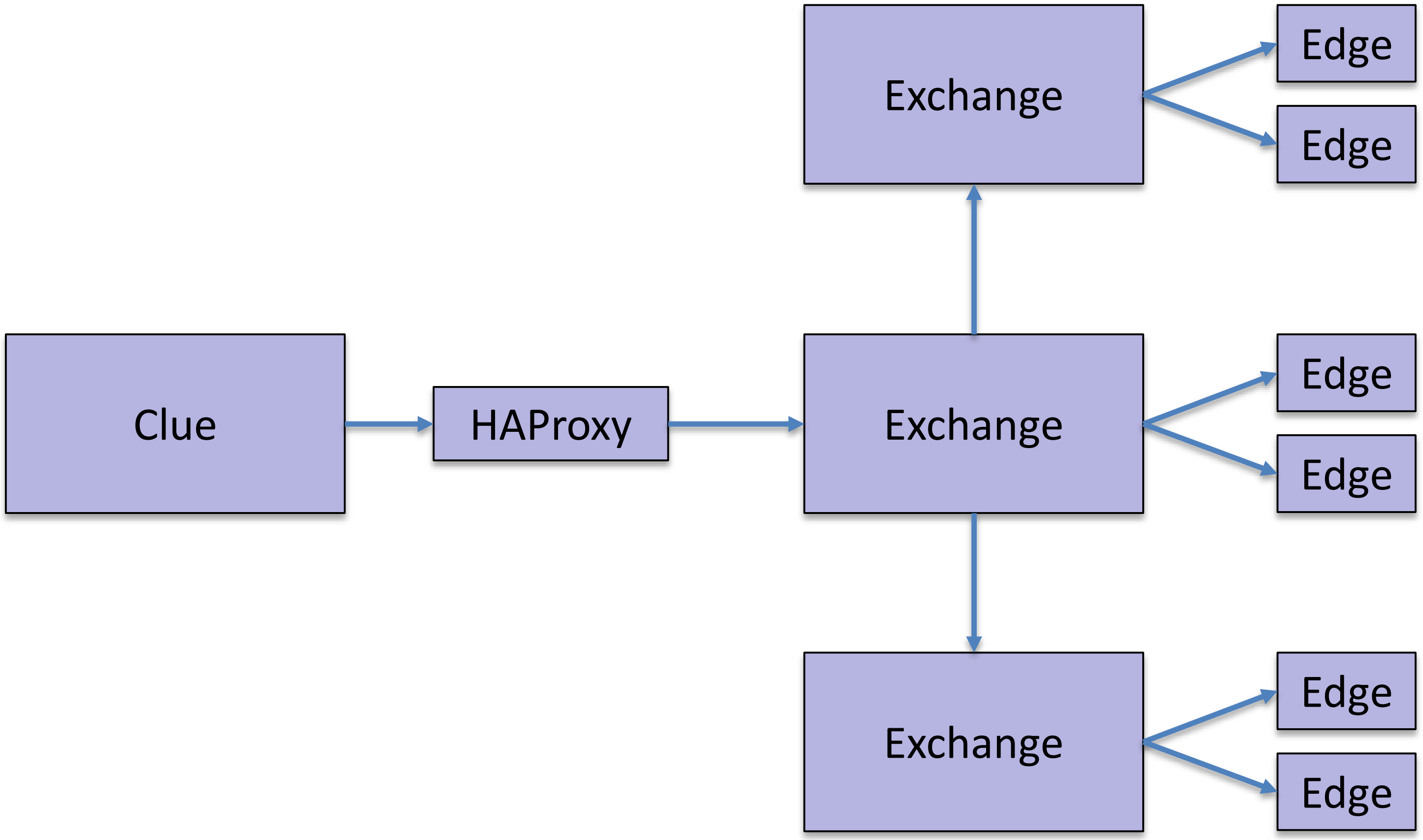
Message Distribution



Message Distribution



Message Distribution



Solving the Distribution Problem

- Edge/Clue instrumentation show no errors
- Exchange isn’t even instrumented!
- Let’s fix that…

Solving the Distribution Problem

Let’s look at our new exchange logs

```
Feb 19 14:11:06 exchange: [exchange] i/o timeout
Feb 19 14:11:06 exchange: [exchange] Exchange success
Feb 19 14:11:06 exchange: [exchange] i/o timeout
Feb 19 14:11:06 exchange: [exchange] i/o timeout
Feb 19 14:11:06 exchange: [exchange] i/o timeout
Feb 19 14:11:06 exchange: [exchange] i/o timeout
Feb 19 14:11:06 exchange: [exchange] i/o timeout
Feb 19 14:11:06 exchange: [exchange] i/o timeout
Feb 19 14:11:06 exchange: [exchange] Exchange success
```

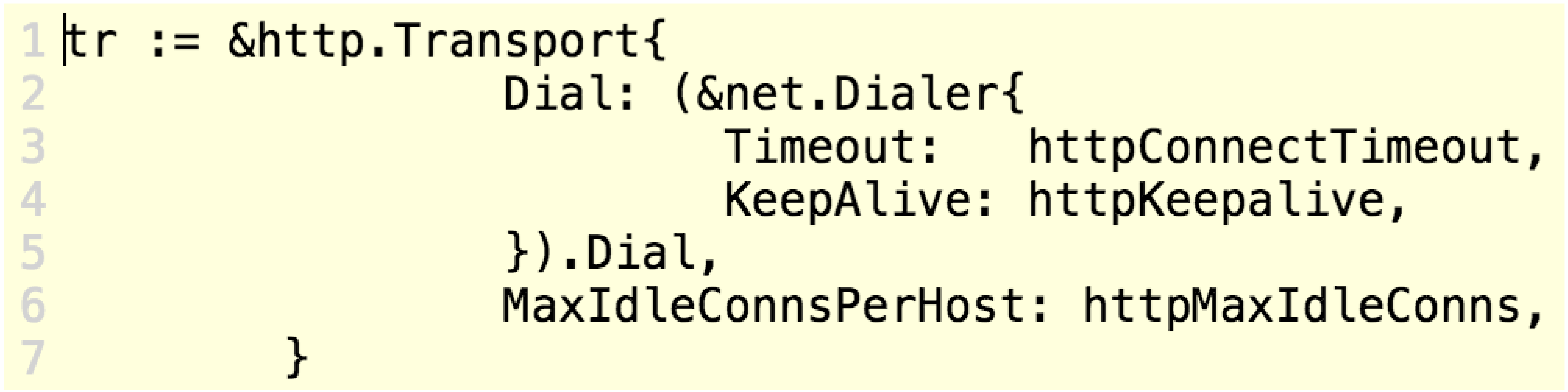
Ideas?

Solving the Distribution Problem

- These are extremely short and simple requests, but there are many of them
- We aren’t using HTTP keepalives

Solving the Distribution Problem

Go makes this extremely simple



```
tr := &http.Transport{
                Dial: (&net.Dialer{
                        Timeout:   httpConnectTimeout,
                        KeepAlive: httpKeepalive,
                }).Dial,
                MaxIdleConnsPerHost: httpMaxIdleConns,
        }
```

Lessons Learned

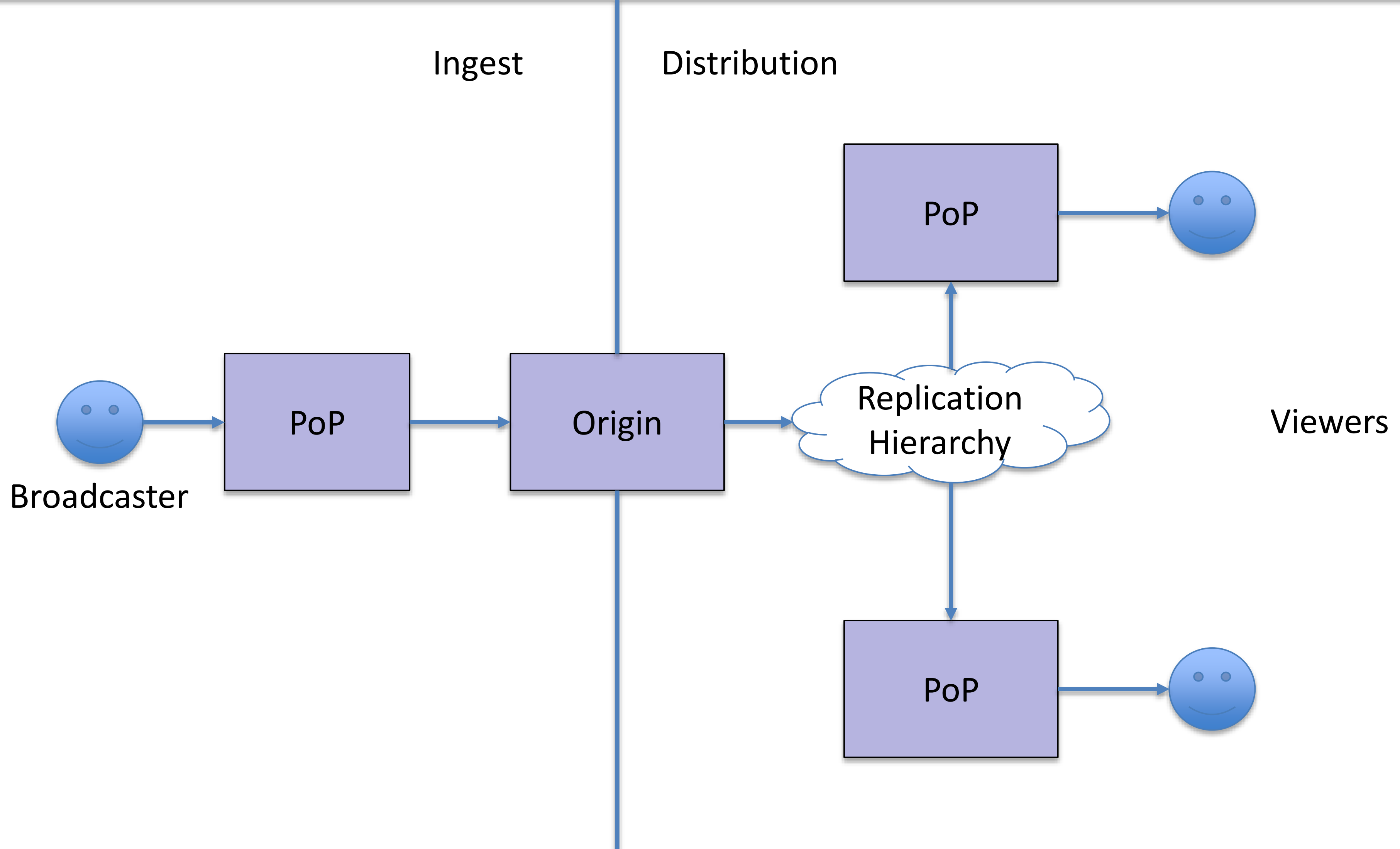
- Better logs/instrumentation to make debugging easier
- Generate fake traffic to force us to handle more load than we need
- Make sure we use and configure our infrastructure wisely!

What have we done since?

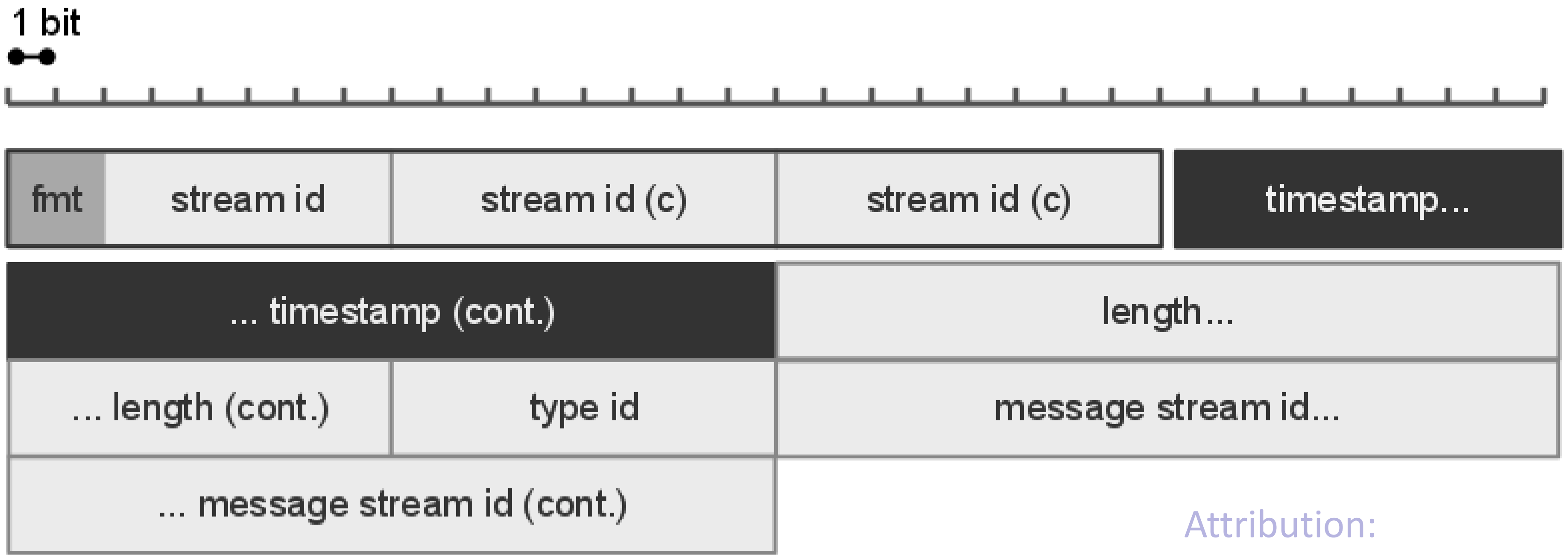
- We now use AWS servers exclusively
- Better Redis infrastructure
- Python -> Go
- Lots of other big and small changes to support new products and better QoS

Thank you.
Questions?

Video Architecture

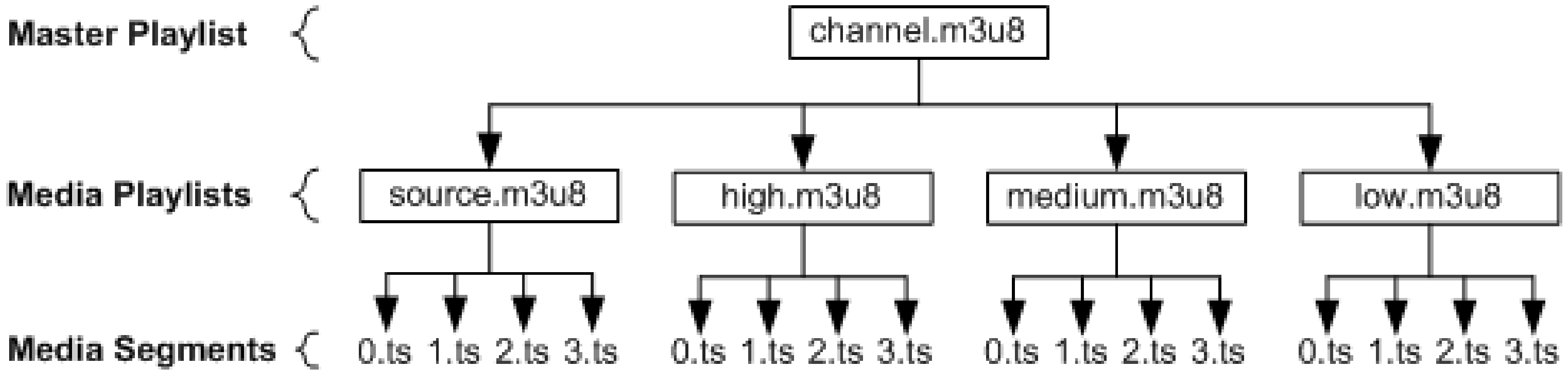


RTMP Protocol



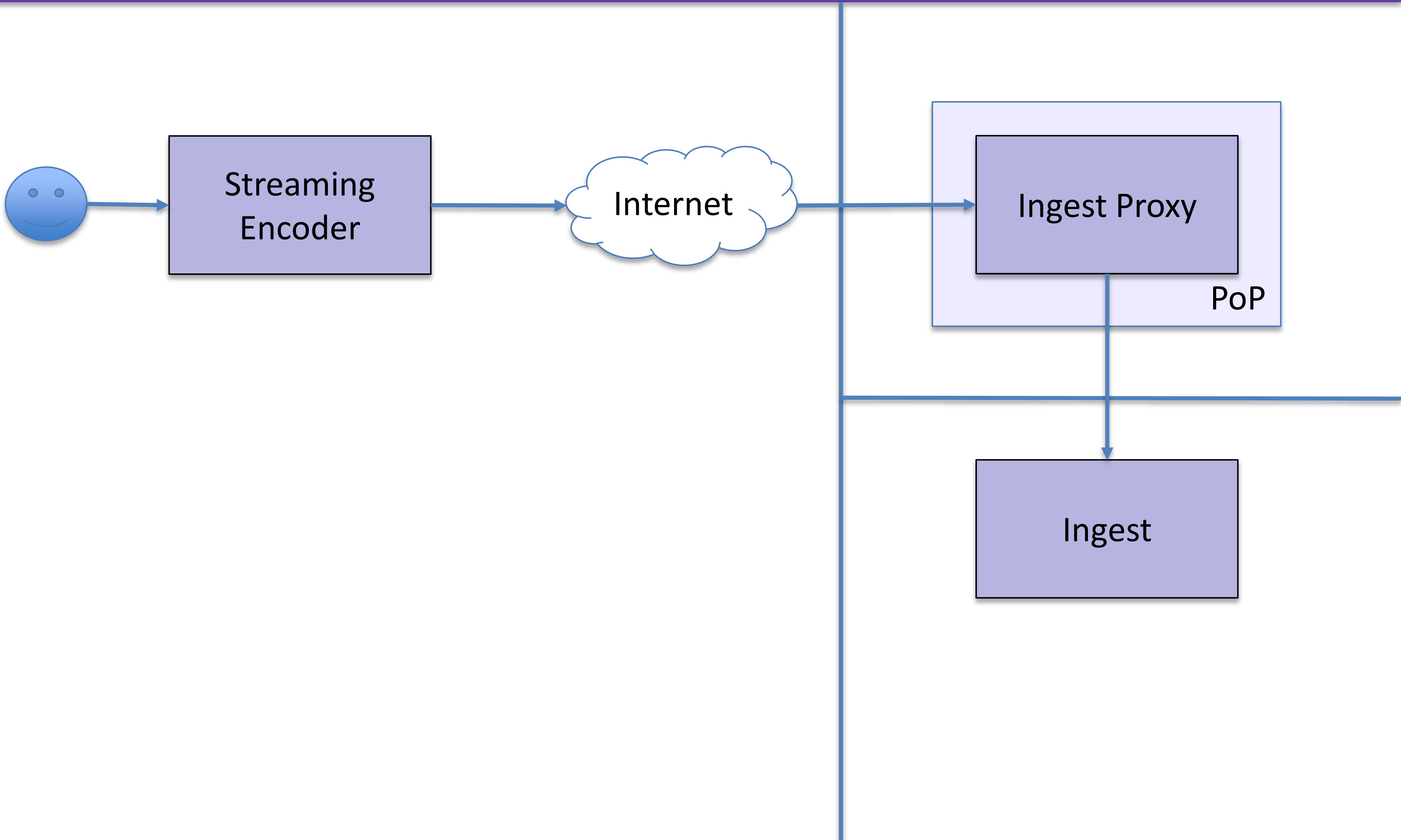
Attribution:
MMick66 - Wikipedia

HLS Protocol



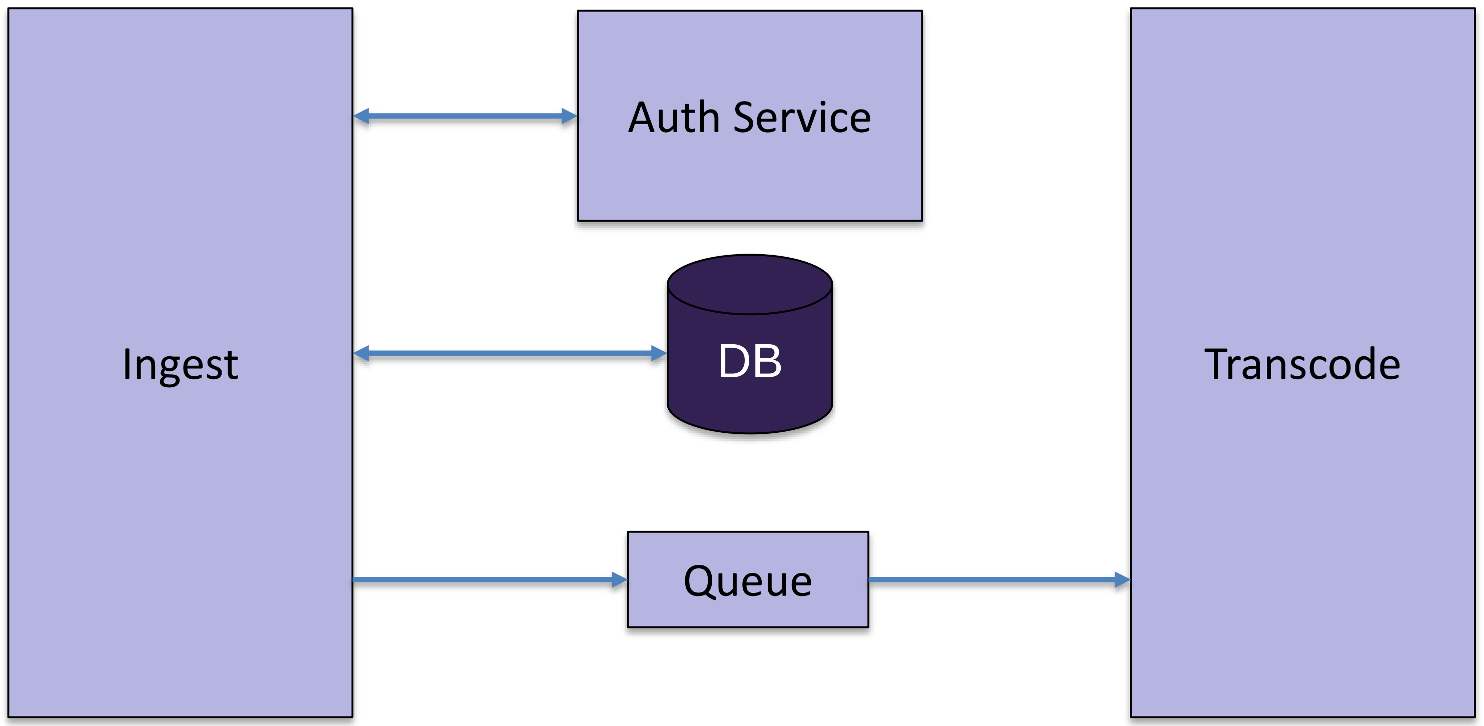
www.twitch.tv

Video Ingest

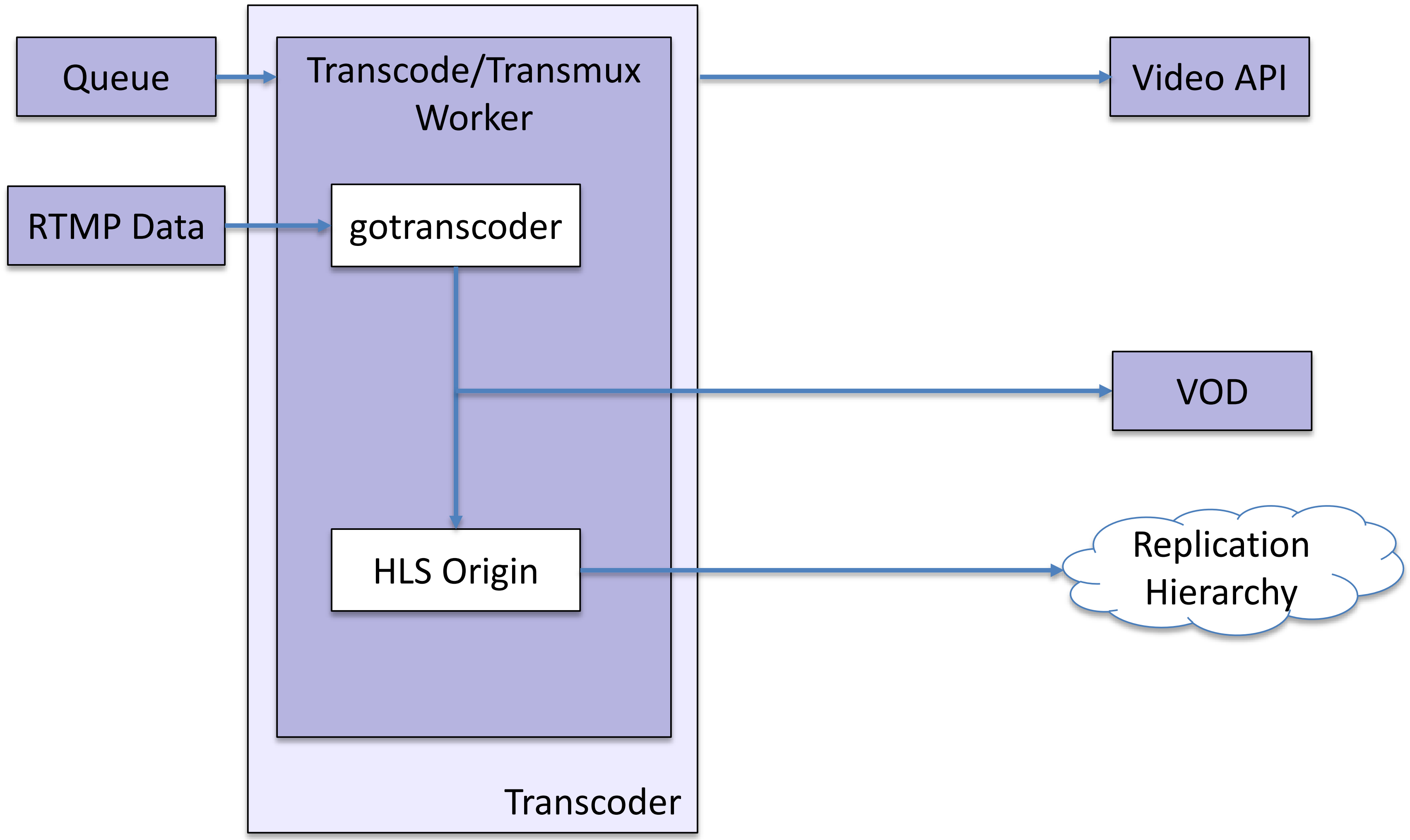


www.twitch.tv

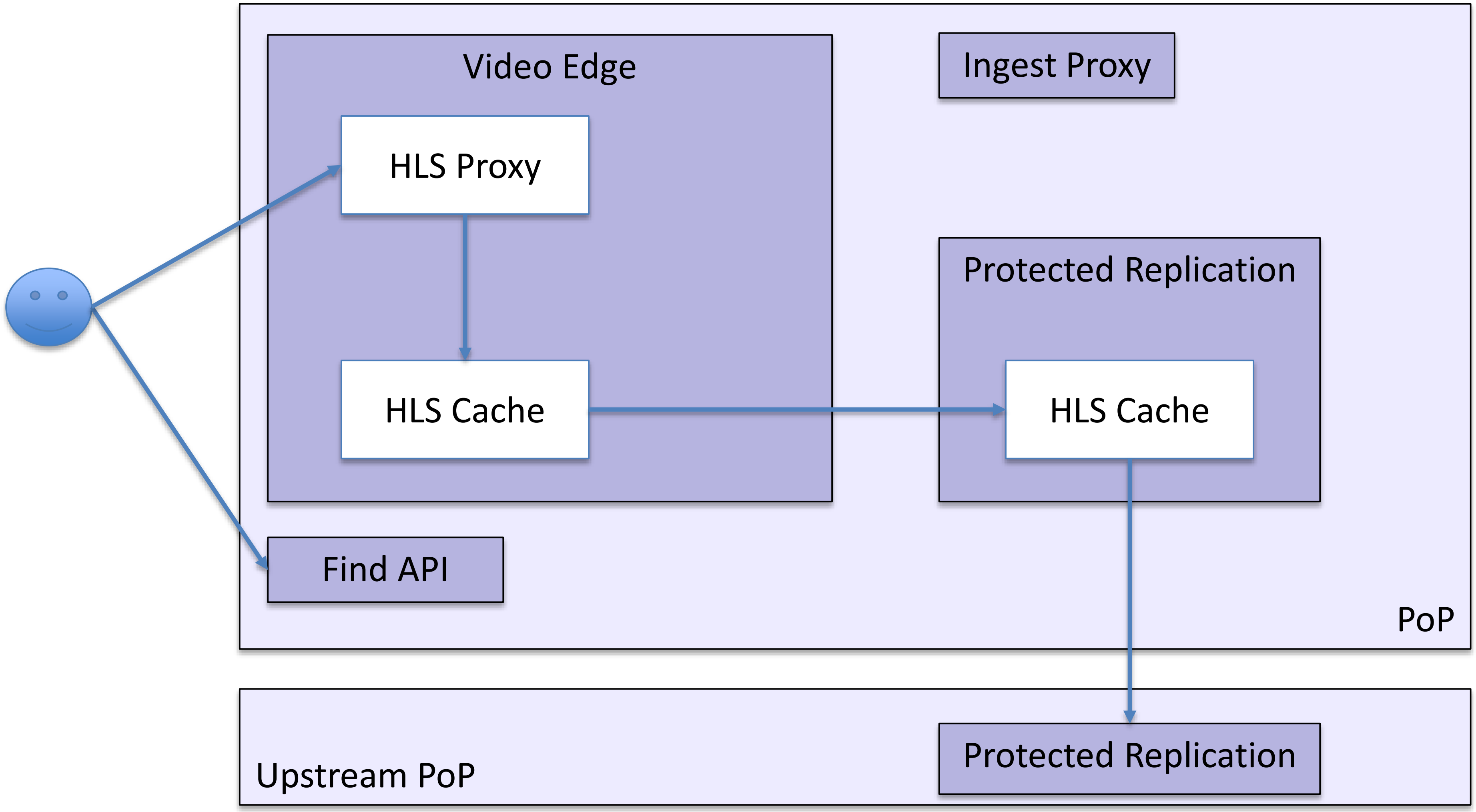
Video Ingest



Video Ingest

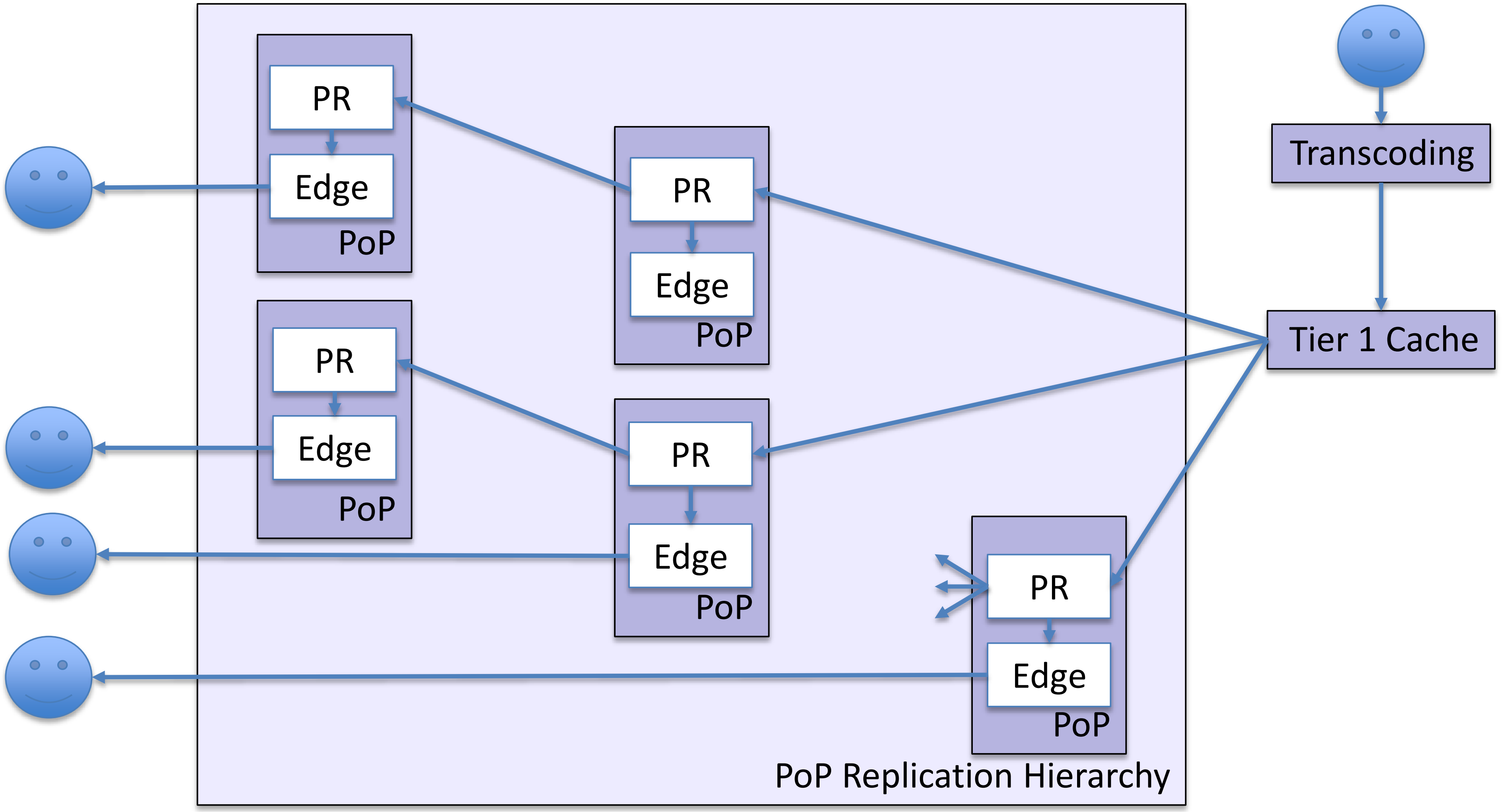


Video Distribution



www.twitch.tv

Video Distribution



Thank you.
Questions?