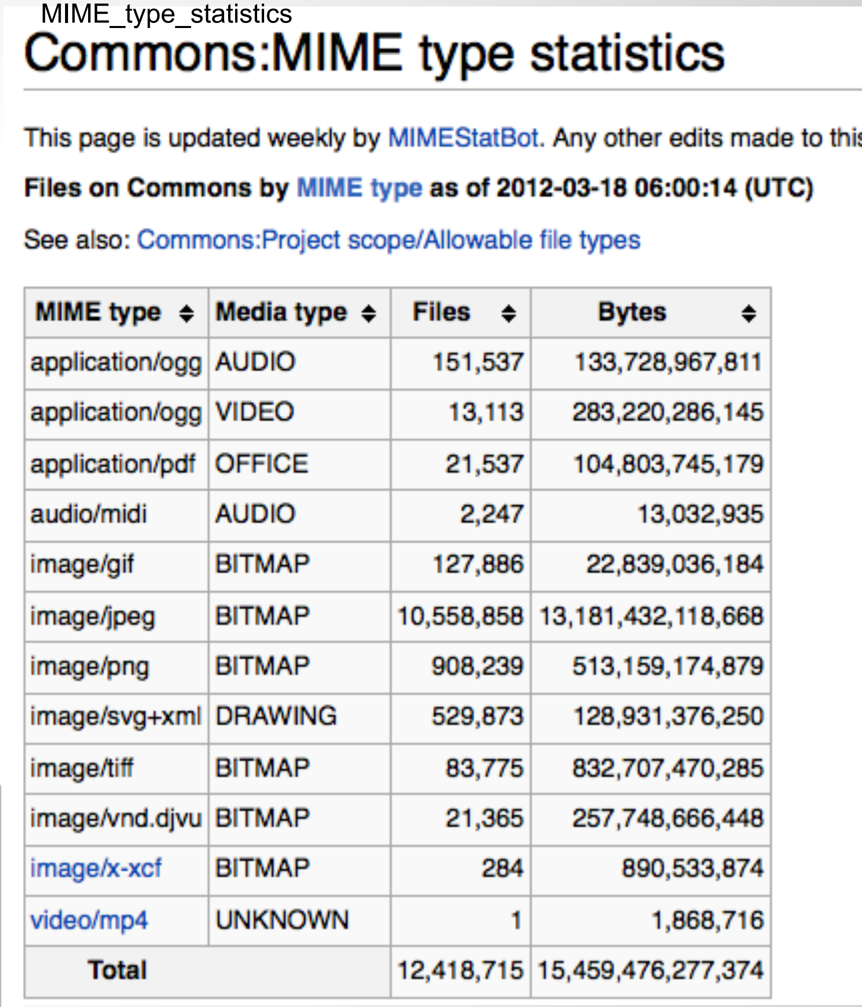
Swift at Wikimedia

Ben Hartshorne
Operations Engineer
<bhartshorne@wikimedia.org>

Media Storage

- All images, sounds, and videos on all wikis
- All scaled versions of all those images
- Currently One Big Box.
- It just keeps growing..

http://commons.wikimedia.org/wiki/Commons:MIME_type_statistics



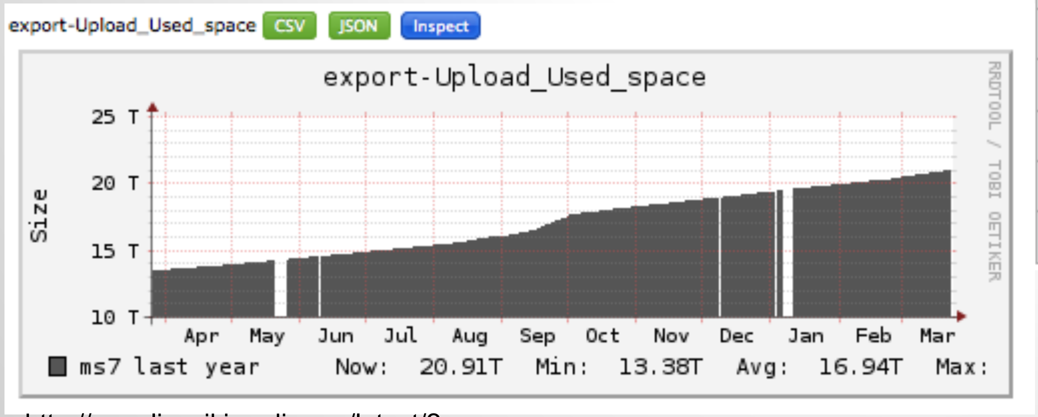
Commons:MIME type statistics

This page is updated weekly by MIMEStatBot. Any other edits made to this

Files on Commons by MIME type as of 2012-03-18 06:00:14 (UTC)

See also: Commons:Project scope/Allowable file types

MIME type	Media type	Files	Bytes
application/ogg	AUDIO	151,537	133,728,967,811
application/ogg	VIDEO	13,113	283,220,286,145
application/pdf	OFFICE	21,537	104,803,745,179
audio/midi	AUDIO	2,247	13,032,935
image/gif	BITMAP	127,886	22,839,036,184
image/jpeg	BITMAP	10,558,858	13,181,432,118,668
image/png	BITMAP	908,239	513,159,174,879
image/svg+xml	DRAWING	529,873	128,931,376,250
image/tiff	BITMAP	83,775	832,707,470,285
image/vnd.djvu	BITMAP	21,365	257,748,666,448
image/x-xcf	BITMAP	284	890,533,874
video/mp4	UNKNOWN	1	1,868,716
Total		12,418,715	15,459,476,277,374



http://ganglia.wikimedia.org/latest/?r=year&cs=&ce=&m=&c=Miscellaneous+pmtpa&h=ms7.pmtpa.wmnet&tab=m&vn=&mc=2&z=medium&metric_group=ALLGROUPS

Alternatives to One Big Box(tm)

We considered a number of clustered storage technologies*, but that was before my time.

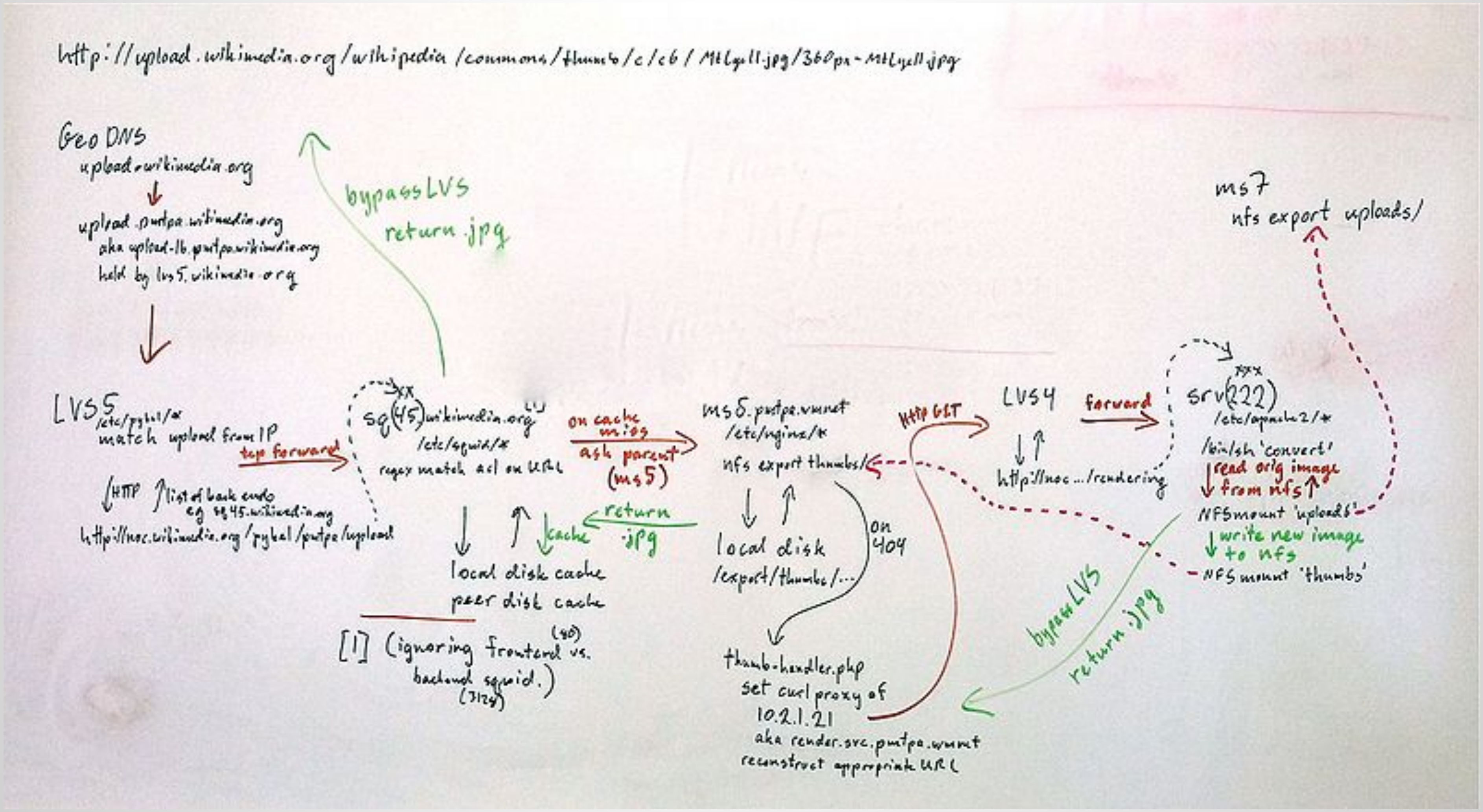
Reasons to use Swift:

- We're using openstack for labs; sticking with the same project is beneficial
- HTTP-accessible object store is a good choice for media storage

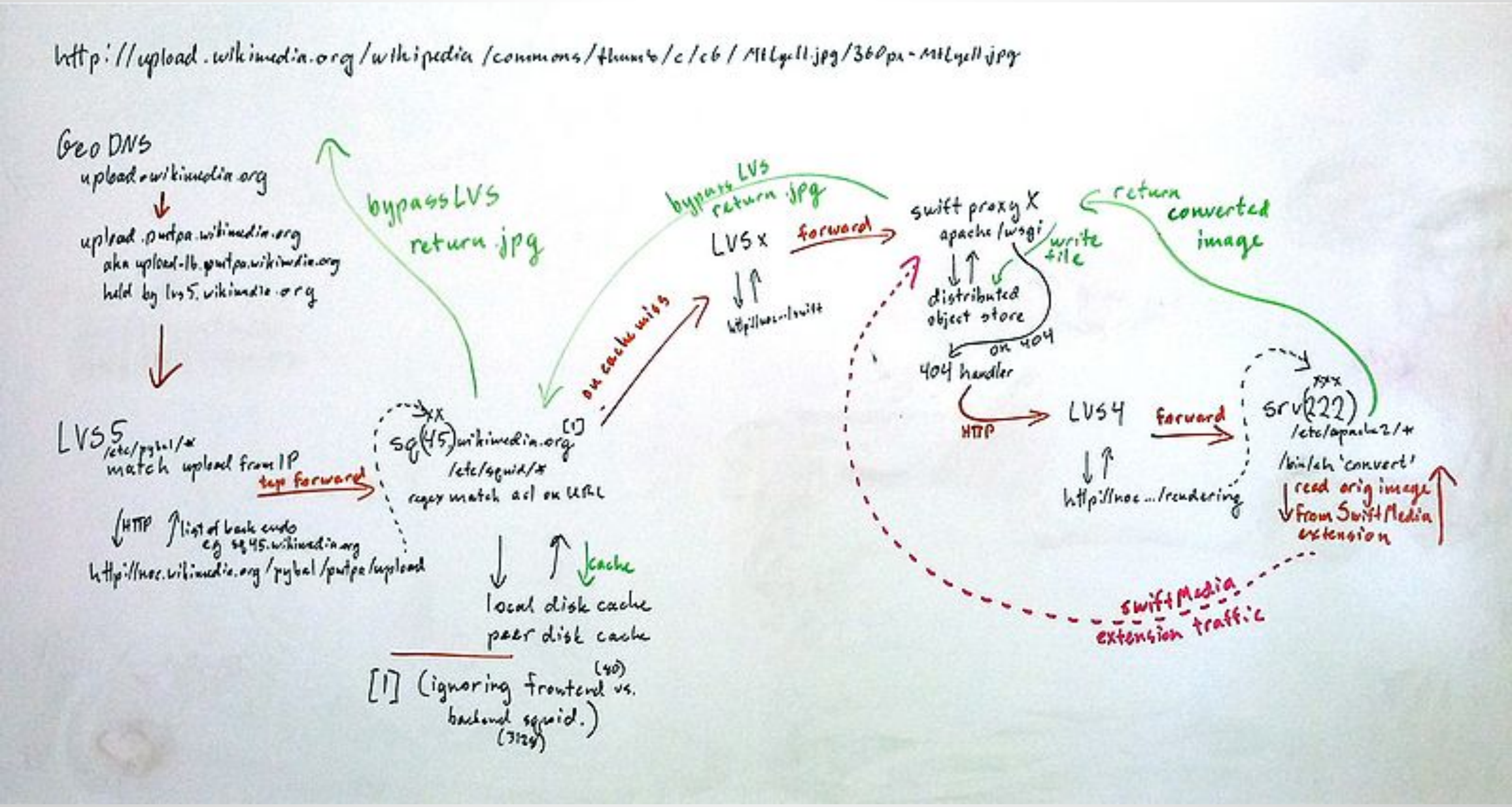
* gluster, mogile, swift, etc. http://wikitech.wikimedia.org/view/Media_server/Distributed_File_Storage_choices

Implementation

How it used to work (thumbnails)



How it will work (not all that different)



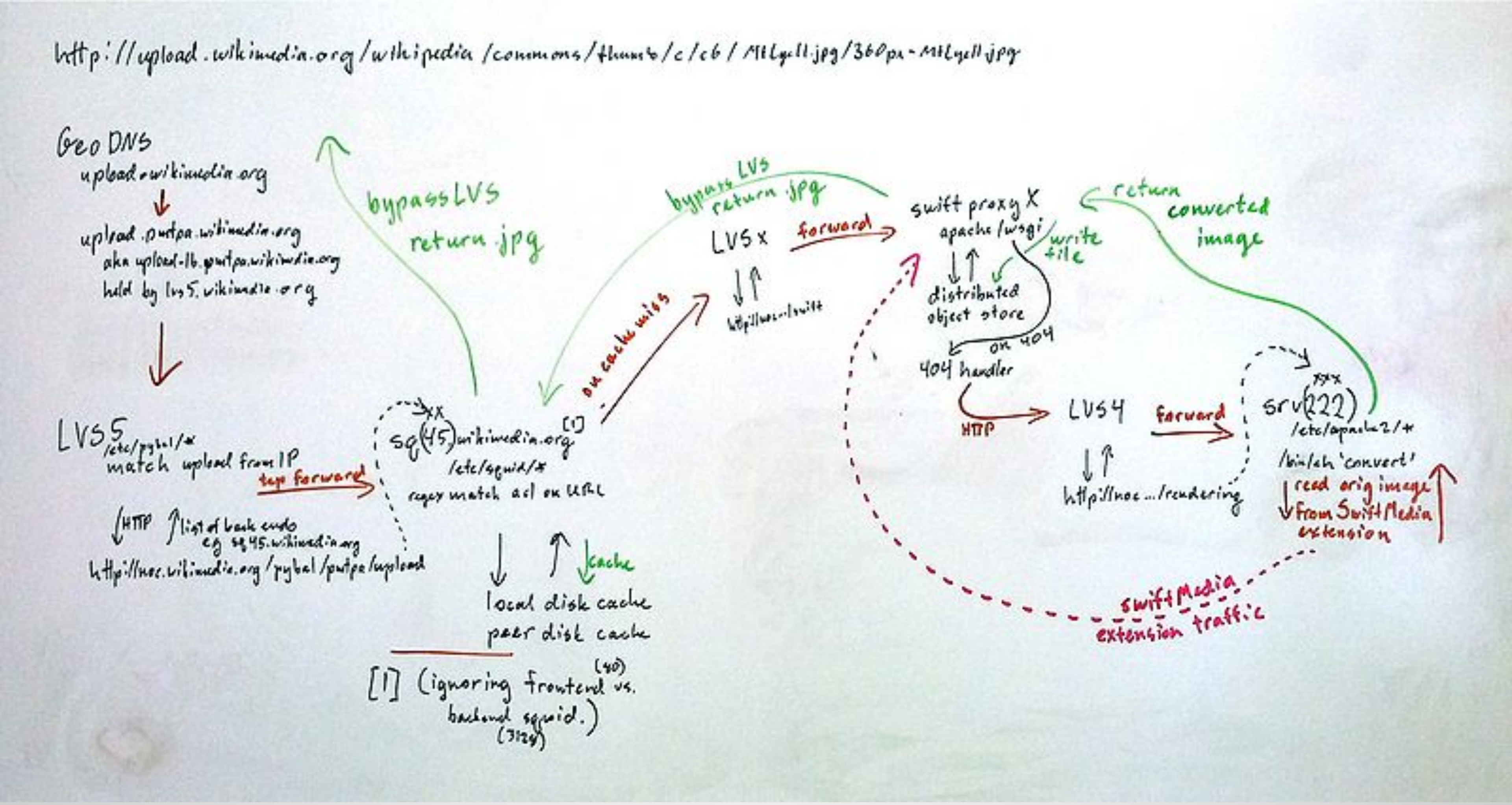
Rewrite middleware

- New thumbnails are scaled on demand
- 404 handler tries to scale images
  that don't exist
- swift-proxy is built for this
  - in /etc/swift/proxy-server.conf:

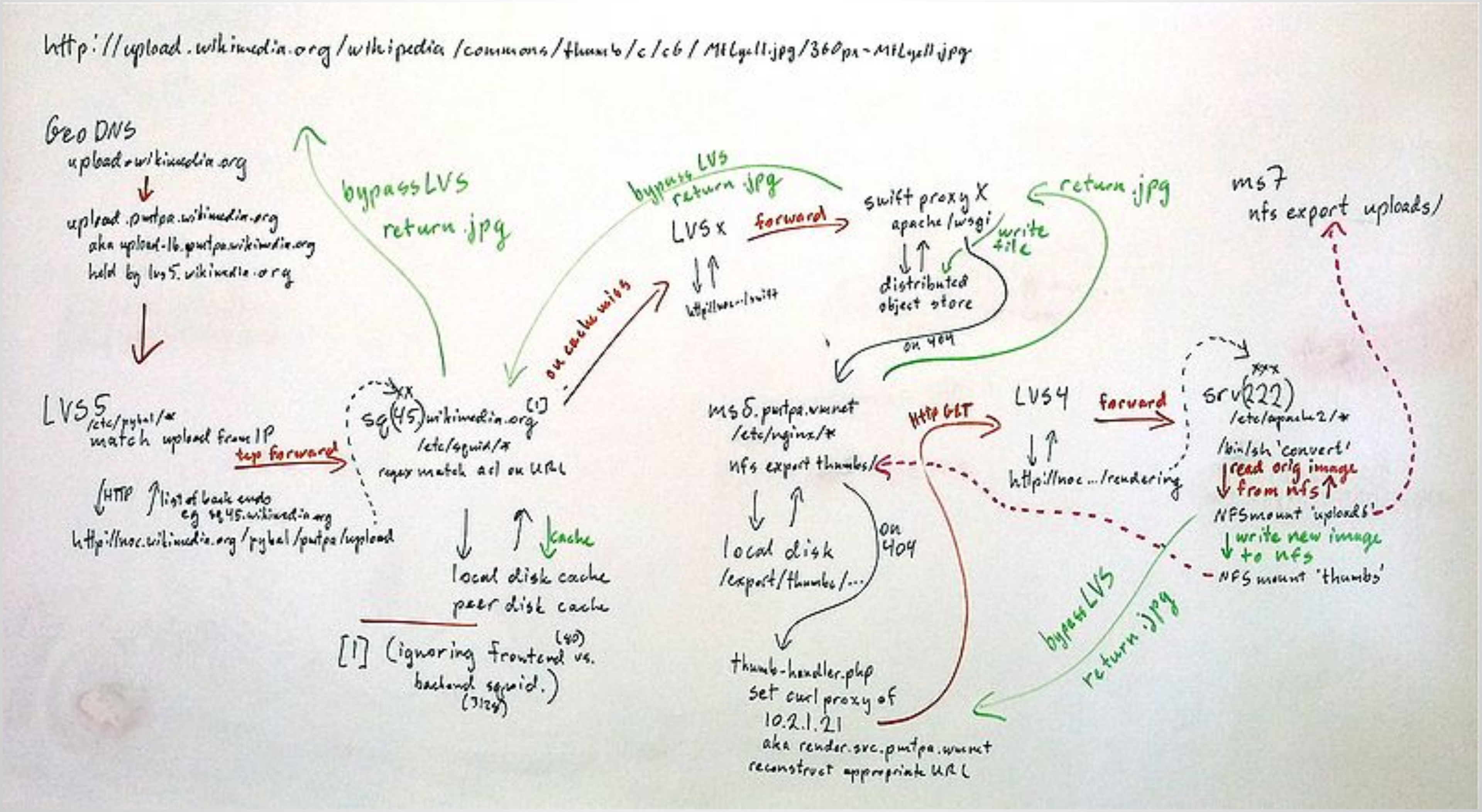
```
 [pipeline:main]
pipeline = rewrite healthcheck cache swauth proxy-server
```

- rewrite does two things
  - call back to get the scaled version of the image
  - write that scaled version into swift

Rewrite middleware



We're only half way through...



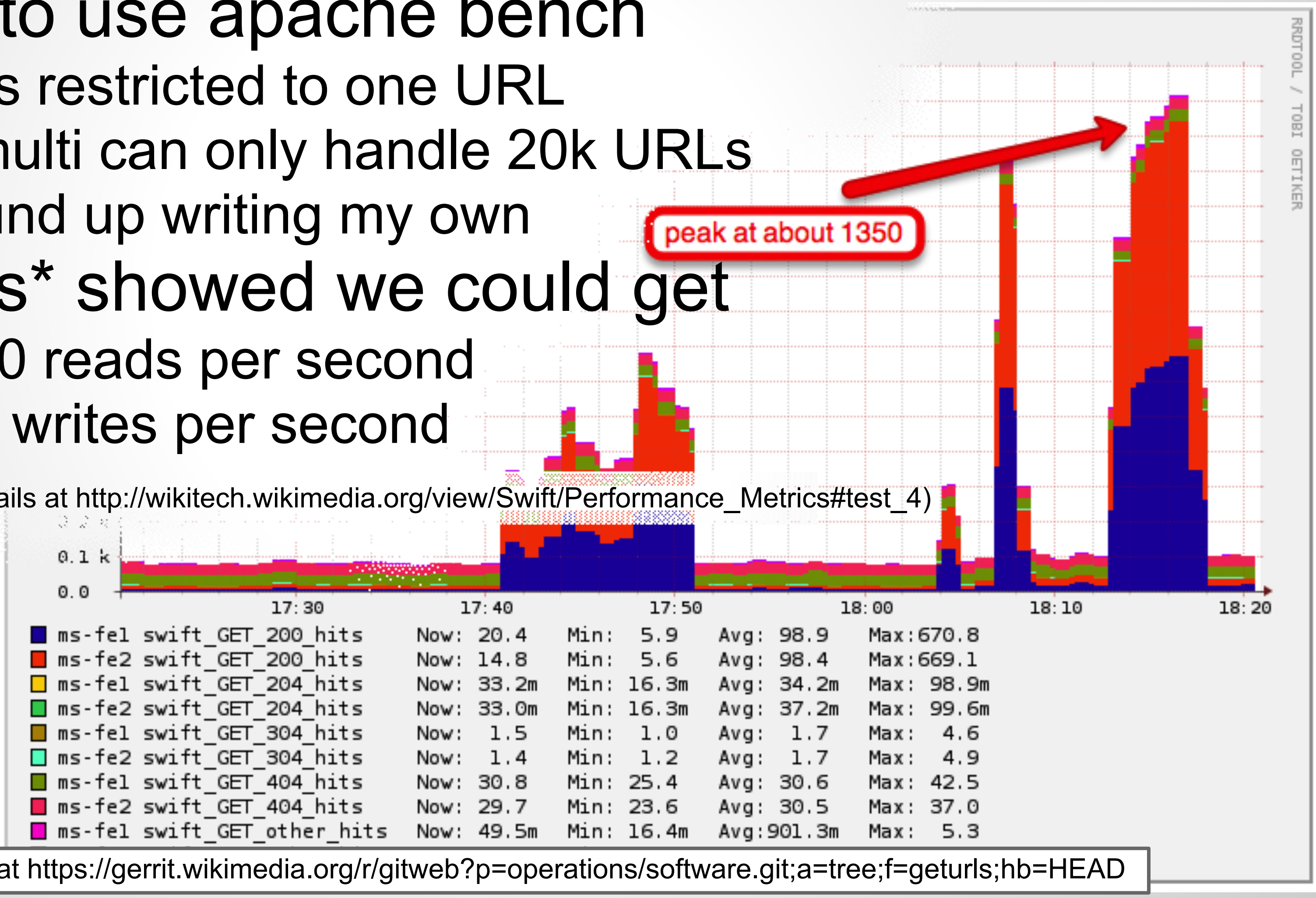
Integration with Mediawiki

- MW storage mechanisms abstracted to a FileBackend class with multiple subclasses
    - local filesystem, swift, azure, S3, etc.
- All interactions with the FileBackend implemented as appropriate for each backend storage module
- Swift storage implemented using CloudFiles
    - https://github.com/rackspace/php-cloudfiles
- More detail on this part: Aaron Schulz

Throughput and Latency Performance

Initial tests

- Tried to use apache bench
  - ab is restricted to one URL
  - abmulti can only handle 20k URLs
  - wound up writing my own
- geturls* showed we could get
  - 1300 reads per second
  - 120 writes per second
  - (full details at http://wikitech.wikimedia.org/view/Swift/Performance_Metrics#test_4)

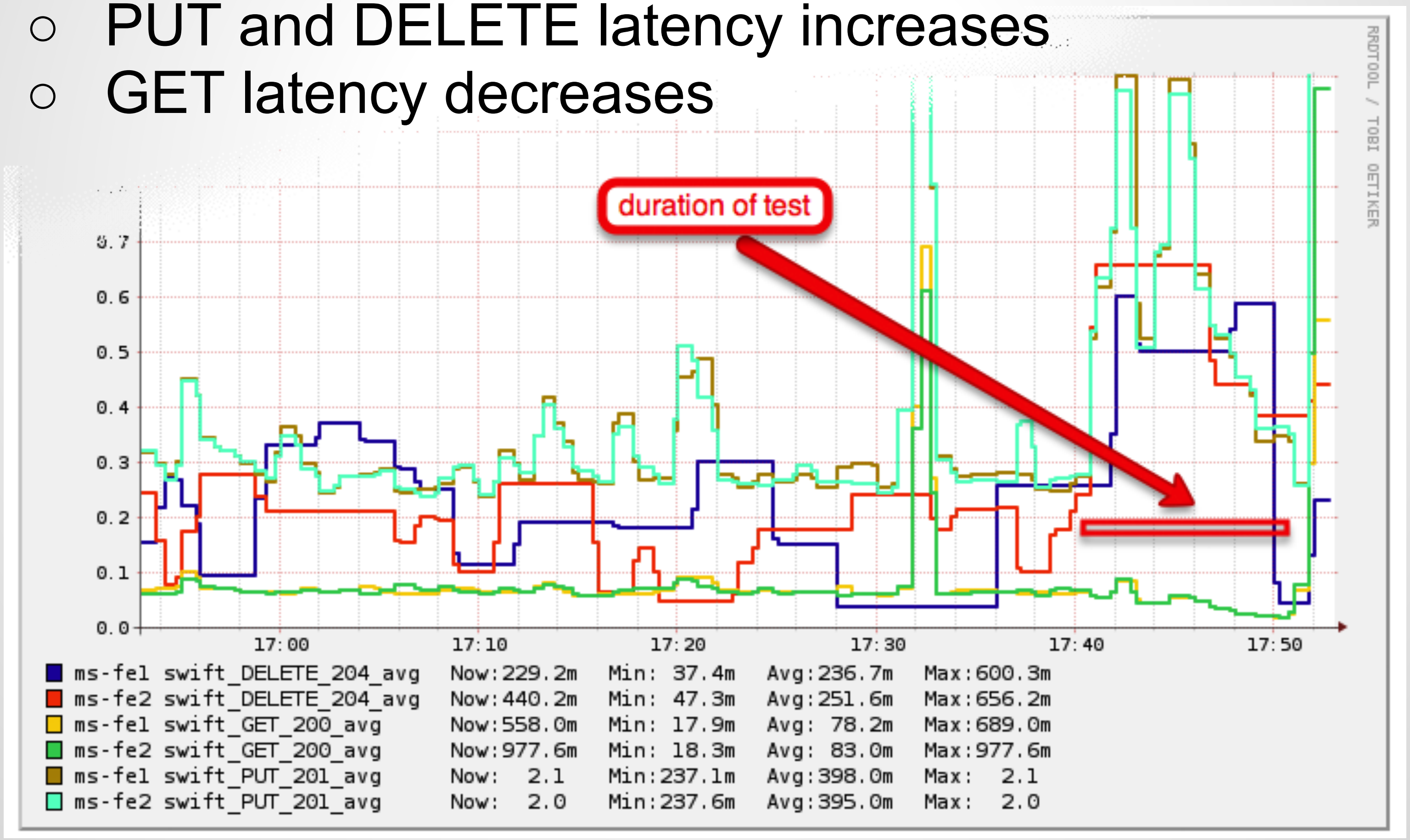


```
🟦 ms-fe1 swift_GET_200_hits     Now: 20.4    Min:  5.9    Avg: 98.9    Max:670.8
🟥 ms-fe2 swift_GET_200_hits     Now: 14.8    Min:  5.6    Avg: 98.4    Max:669.1
🟨 ms-fe1 swift_GET_204_hits     Now: 33.2m   Min: 16.3m   Avg: 34.2m   Max: 98.9m
🟩 ms-fe2 swift_GET_204_hits     Now: 33.0m   Min: 16.3m   Avg: 37.2m   Max: 99.6m
🟫 ms-fe1 swift_GET_304_hits     Now:  1.5    Min:  1.0    Avg:  1.7    Max:  4.6
🟩 ms-fe2 swift_GET_304_hits     Now:  1.4    Min:  1.2    Avg:  1.7    Max:  4.9
🟩 ms-fe1 swift_GET_404_hits     Now: 30.8    Min: 25.4    Avg: 30.6    Max: 42.5
🟥 ms-fe2 swift_GET_404_hits     Now: 29.7    Min: 23.6    Avg: 30.5    Max: 37.0
🟪 ms-fe1 swift_GET_other_hits   Now: 49.5m   Min: 16.4m   Avg:901.3m   Max:  5.3
```

* geturls code available at https://gerrit.wikimedia.org/r/gitweb?p=operations/software.git;a=tree;f=geturls;hb=HEAD

Effect of load on performance

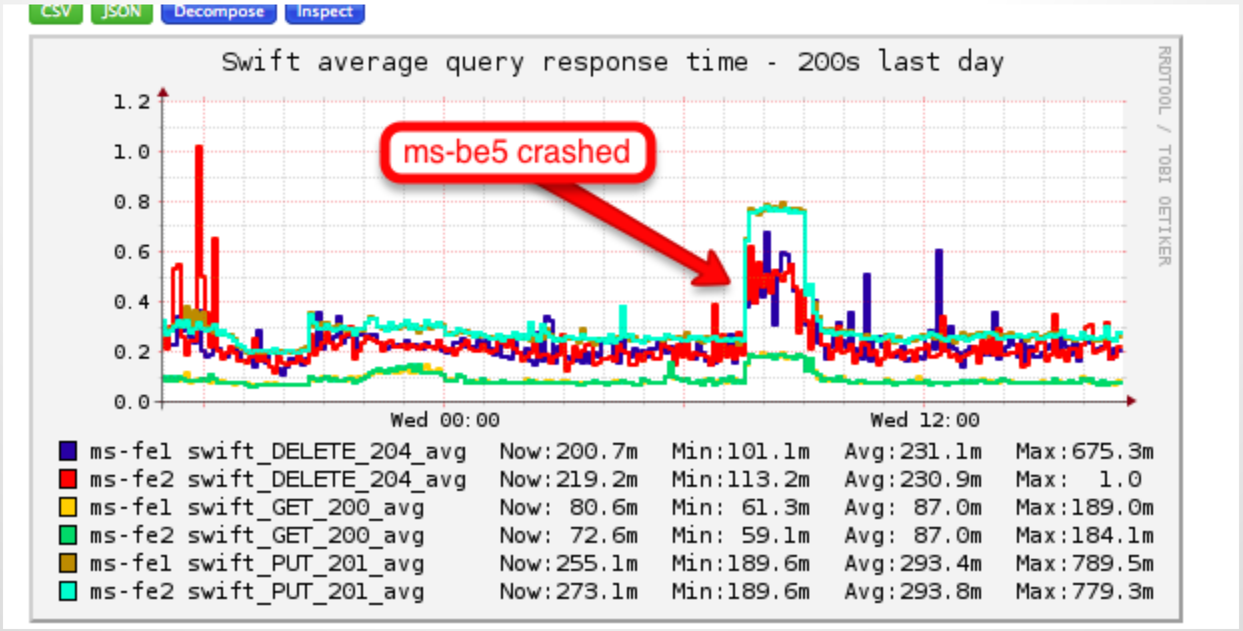
- Under heavy read load
  - PUT and DELETE latency increases
  - GET latency decreases



```
■ ms-fe1 swift_DELETE_204_avg  Now:229.2m  Min: 37.4m  Avg:236.7m  Max:600.3m
■ ms-fe2 swift_DELETE_204_avg  Now:440.2m  Min: 47.3m  Avg:251.6m  Max:656.2m
■ ms-fe1 swift_GET_200_avg     Now:558.0m  Min: 17.9m  Avg: 78.2m  Max:689.0m
■ ms-fe2 swift_GET_200_avg     Now:977.6m  Min: 18.3m  Avg: 83.0m  Max:977.6m
■ ms-fe1 swift_PUT_201_avg     Now:   2.1  Min:237.1m  Avg:398.0m  Max:   2.1
■ ms-fe2 swift_PUT_201_avg     Now:   2.0  Min:237.6m  Avg:395.0m  Max:   2.0
```

Effect of node failure

- One (out of 5) storage nodes crashing
  - 2x read latency (from 100ms to 200ms)
  - 3x write latency (250ms to 750ms)
  - 2.5x delete latency (200ms to 500ms)
- No data (yet) on proxy nodes crashing



🟦	ms-fe1 swift_DELETE_204_avg	Now:200.7m	Min:101.1m	Avg:231.1m	Max:675.3m
🟥	ms-fe2 swift_DELETE_204_avg	Now:219.2m	Min:113.2m	Avg:230.9m	Max: 1.0
🟨	ms-fe1 swift_GET_200_avg	Now: 80.6m	Min: 61.3m	Avg: 87.0m	Max:189.0m
🟩	ms-fe2 swift_GET_200_avg	Now: 72.6m	Min: 59.1m	Avg: 87.0m	Max:184.1m
🟫	ms-fe1 swift_PUT_201_avg	Now:255.1m	Min:189.6m	Avg:293.4m	Max:789.5m
🟦	ms-fe2 swift_PUT_201_avg	Now:273.1m	Min:189.6m	Avg:293.8m	Max:779.3m

Open Performance Questions

- It's not clear where the bottleneck exists
  - Are we bound on CPU, memory or some configuration parameter?
- how does scaling the number of proxies vs. storage nodes affect performance?
- what are the impacts of various configuration choices on performance?
  - eg. number of auditing and replication processes
- what is the effect of rebalancing the rings on performance?

Open Performance Questions

- how long does it take before a newly added node no longer affects performance? (~1wk)
- how do we measure container listing latency?

Open problems

- Effect of one storage node crashing on performance is too large
- Container listing latency is sometimes too high
- Consistency problems with the rewrite middleware
  - ETags help
  - Still have issues sometimes (cleaner script)
- It's difficult diagnosing problems with rewrite
  - natural effect of asynchronous code (eventlet)
  - eg. stack trace in proxy logs

WIKIMEDIA
FOUNDATION

Thanks!

Ben Hartshorne
Operations Engineer
<bhartshorne@wikimedia.org>