Building a Billion User Load Balancer

Patrick Shuff
Production Engineer, Traffic Team
We’ll be talking about

- Serving Dynamic Facebook Requests
- L4/L7 Load Balancing
- Edge PoP and Reducing Latency
- Global DNS Load Balancing
- Q&A
Facebook scale
as of March 2015

1.25 billion mobile monthly active people
936 million daily active people on average
1.44 billion monthly active people
798 million mobile daily active people on average

Facebook scale
as of March 2015

1.25 billion mobile monthly active people
936 million daily active people on average
1.44 billion monthly active people
798 million mobile daily active people on average

Approximately 82.4% of our daily active users are outside the US and Canada
What is Facebook?
(from traffic's perspective)

Dynamic Requests
- Newsfeed
- Likes
- Messaging
- Status Updates

Static Requests
- Photos
- Videos
- Javascript/CSS
What is Facebook?
(from traffic's perspective)

Dynamic Requests
Static Requests

Newsfeed
Photos
Likes
Videos
Messaging
Javascript/CSS

Status Updates

Terabits of egress (outgoing bits per second)
What is facebook?
(from traffic's perspective)

Dynamic Requests

Newsfeed
Likes
Messaging
Status Updates

Static Requests

Photos
Videos
Javascript/CSS
What are we talking about?

HHVM
What are we not talking about?

- HHVM
- MySQL
- Cache
- Msgs
- Feed
- Ads
Weekly egress cycle

Egress

Ingress

7 Days
Diurnal egress Cycle

Time zone == Pacific (-0800 GMT)
Sum of timezones

United Kingdom
Canada
Indonesia

Time zone == Pacific (-0800 GMT)
TCP/IP Review
<table>
<thead>
<tr>
<th>Layer</th>
<th>Purpose</th>
<th>Ex</th>
</tr>
</thead>
<tbody>
<tr>
<td>7: Application</td>
<td>High-Level API</td>
<td>HTTP, SPDY, MQTT</td>
</tr>
<tr>
<td>6: Presentation</td>
<td>Data Translation</td>
<td>ASCII, JPEG</td>
</tr>
<tr>
<td>5: Session</td>
<td>Communication Session</td>
<td>RPC</td>
</tr>
<tr>
<td>4: Transport</td>
<td>Transmission</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>3: Network</td>
<td>Address, Routing, Flow</td>
<td>IPv6, IPv4</td>
</tr>
<tr>
<td>2: Data Link</td>
<td>Reliable Physical Comm.</td>
<td>IEEE, 802.2</td>
</tr>
<tr>
<td>1: Physical</td>
<td>Raw bit transmission</td>
<td>DSL, USB</td>
</tr>
</tbody>
</table>
## IP Header (OSI Layer 3)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version number of the IP protocol</td>
</tr>
<tr>
<td>DSCP</td>
<td>Differentiated Services Code Point</td>
</tr>
<tr>
<td>ECN</td>
<td>Explicit Congestion Notification</td>
</tr>
<tr>
<td>Flow Label</td>
<td>Flow Label</td>
</tr>
<tr>
<td>Payload Length</td>
<td>Payload length</td>
</tr>
<tr>
<td>Next Header</td>
<td>Next header</td>
</tr>
<tr>
<td>Hop limit</td>
<td>Hop limit</td>
</tr>
<tr>
<td>Source Address</td>
<td>Source address</td>
</tr>
<tr>
<td>Destination Address</td>
<td>Destination address</td>
</tr>
<tr>
<td>Data</td>
<td>Data</td>
</tr>
</tbody>
</table>
# TCP Header (OSI Layer 4)

<table>
<thead>
<tr>
<th>Source Port</th>
<th>Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence Number</td>
<td>Acknowledgement Number</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>Application Payload</td>
<td></td>
</tr>
</tbody>
</table>

---

© 2015 Facebook | Menlo Park | Credit:
HTTP Request (OSI Layer 7)

GET / HTTP/1.1
host: www.facebook.com
## Putting it all together

<table>
<thead>
<tr>
<th>Version</th>
<th>DSCP</th>
<th>ECN</th>
<th>Flow Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Payload Length</strong></td>
<td><strong>Next Header</strong></td>
<td><strong>Hop limit</strong></td>
<td></td>
</tr>
</tbody>
</table>

### L3

#### Source Address

#### Destination Address

<table>
<thead>
<tr>
<th>Source Port</th>
<th>Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Sequence Number</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Acknowledgement Number</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>....</strong></td>
</tr>
</tbody>
</table>

```
GET / HTTP/1.1
host: www.facebook.com
```
Putting it all together

IP Packet

TCP Segment

HTTP Request
Serving Dynamic Facebook Requests
FB Request -- one web server

rps = requests per second

low rps

how do we get more rps?!
Add a load balancer!

DNS

L7LB (proxyogen)

lots more rps

low rps

www ?

rps = requests per second

how do we get more rps?!
Add another load balancer!

network bound

lots more rps

low rps

how do we get more rps?!
Add another load balancer!

AAAA www.facebook.com
AAAA 2a03:2880:2130:cf05:face:b00c::1

DNS

ECMP

GET /<html>... L4LB (ipvs) network bound

GET /<html>... L7LB (proxygen) lots more rps

HHVM

HHVM

low rps

lots more rps

network bound

GET /<html>... L4LB (ipvs)
Front end Web Cluster

~10

~100

Thousands
More RPS? Add another cluster!

DNS

AAAA www.facebook.com
AAAA 2a03:2880:2130:cf05:face:b00c::1

GET /<html>...
L4LB (ipvs)
L7LB (proxygen)
HHVM

GET /<html>...
L4LB (ipvs)
L7LB (proxygen)
HHVM

how do we get more rps?!
Add another datacenter!

AAAA www.facebook.com
AAAA 2a03:2880:2130:cf05:face:b00c::1
Not really top down

1.2.3.4
L4LB
L7LB
HHVM
Load Balancing: L4/L7
Let's break it down

www?
A 1.2.3.4

DNS

L4LB (ipvs)

GET / <html>

L4LB (ipvs)

GET / <html>

L7LB (proxygen)

GET / <html>

HHVM

L4 = IP/port
L7 = HTTP
### OSI Model: What is L4/L7?

<table>
<thead>
<tr>
<th>Layer</th>
<th>Purpose</th>
<th>Ex</th>
</tr>
</thead>
<tbody>
<tr>
<td>7: Application</td>
<td>High-Level API</td>
<td>HTTP, SPDY, MQTT</td>
</tr>
<tr>
<td>6: Presentation</td>
<td>Data Translation</td>
<td>ASCII, JPEG</td>
</tr>
<tr>
<td>5: Session</td>
<td>Communication Session</td>
<td>RPC</td>
</tr>
<tr>
<td>4: Transport</td>
<td>Transmission</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>3: Network</td>
<td>Address, Routing, Flow</td>
<td>IPv6, IPv4</td>
</tr>
<tr>
<td>2: Data Link</td>
<td>Reliable Physical Comm.</td>
<td>IEEE, 802.2</td>
</tr>
<tr>
<td>1: Physical</td>
<td>Raw bit transmission</td>
<td>DSL, USB</td>
</tr>
</tbody>
</table>
NOTE: L7 (proxygen) Listens to the VIP on loopback (lo) interface. Not eth0.
NOTE: L7 (proxygen) Listens to the VIP on loopback (lo) interface. Not eth0.
L4LB Example

ECMP Hash

State Table + Hash

Example

State Table

Hash

ECMP

face:b00c::1 (lo)

face:b00c::1 (lo)

face:b00c::1 (lo)

face:b00c::1 (lo)
L4LB Routing

ECMP Hash

State Table + Hash

B

face:b00c::1 (lo)

L4LB

face:b00c::1 (lo)

L4LB

face:b00c::1 (lo)

L4LB

face:b00c::1 (lo)

L4LB

face:b00c::1 (lo)

L7LB

face:b00c::1 (lo)

L7LB

face:b00c::1 (lo)

L7LB

face:b00c::1 (lo)

L7LB

face:b00c::1 (lo)

L7LB
Remember this?

Original IP Packet

TCP Segment

HTTP Request
IP in IP encapsulation

IP Packet from IPVS

Original IP Packet

TCP Segment

HTTP Request
Direct Server Return

- Original IP Packet
  - TCP Segment
  - HTTP Request

- IP Packet from IPVS
  - L4LB
    - TCP Segment
    - HTTP Request
  - L7LB
    - TCP/SSL Term
    - Forward HTTP Request

- Response IP Pckt
  - L4LB
    - TCP Segment
    - HTTP Response
  - L7LB
    - TCP Segment
    - HTTP Response

- Original IP Packet
  - TCP Segment
  - HTTP Request

- IP Packet from IPVS
  - face:b00c::1 (lo)

- TCP Segment
  - HTTP Request

- Response IP Pckt
  - face:b00c::1 (lo)

- TCP Segment
  - HTTP Response
Edge PoP’s & Reducing Latency
Seoul -> Oregon

TCP Connect: 150ms
HTTPS Seoul -> Oregon

TCP conn established: 150 ms
SSL session established: 450 ms
Response Received 600 ms

SYN
ACK
ClientHello
ServerHello
ChangeCipherSpec
ChangeCipherSpec
GET
HTTP 1.1
Seoul -> Tokyo -> Oregon

TCP Connect: 30ms
SSL Session: ??
HTTP Response: ??
HTTPS Seoul->Tokyo->Oregon

Sessions established: 90 ms (vs 450 ms)

Response Received: 240 ms

GET

15ms

60ms

GET

Request Received

HTTP 1.1 200

HTTP 1.1
Seoul -> Oregon

TCP Connect: 150ms - 30ms
SSL Session: 450ms - 90ms
HTTP Response: 600ms - 240ms
How do the LB’s in PoP’s work?

TCP Routing (ip/port)

TCP/SSL

HTTP

Facebook

L4LB

L7LB

face:b00c::1

face:b00c::1 (lo)

face:b00c::1

face:b00c::1 (lo)

HHVM

HHVM


DNS LB: Cartographer
Edge POP Locations

*POP = points of presence.
DNS LB Decision

Considerations:
- Closest Edge** to user
- Capacity
- Health
- Geo data
Sonar: Measuring “Closeness”
Cartographer Architecture

Internet

ISP DNS resolver

Facebook Infrastructure

cartographer

DNS map

DNS Server

capacity
resolvers/latency
routing
health

offline processing
Cartographer in action

- **User Demand**
- **Capacity Limit**
- **Cartographer Controlled Utilization**
- **Morning**
- **Mid-Day**
- **Evening**
- **Sleep**
Regional Load Shedding
Global Load Shedding
Monitoring
Monitoring (Not enough time)
Open Source
Open Source Components

- Proxygen Libs
  https://github.com/facebook/proxygen
- HHVM
  https://hhvm.com
- TinyDNS
  https://cr.yp.to/djbdns/tinydns.html
- IPVS (IP Virtual Server)
  http://www.linuxvirtualserver.org/software/ipvs.html
- ExaBGP
  https://github.com/Exa-Networks/exabgp
- Python
  https://python.org
- Zookeeper
  https://zookeeper.apache.org
Photo Credits

http://www.flickr.com/photos/27587002@N07/5170590074
http://www.flickr.com/photos/yaketyyakyak/7001664846
http://www.flickr.com/photos/hinnosaar/3778903507
http://www.flickr.com/photos/eamoncurry/8698726494
http://www.flickr.com/photos/43158397@N02/4514113429
http://www.flickr.com/photos/nobusue/6876280595
http://www.flickr.com/photos/29487672@N07/14760573314
http://www.flickr.com/photos/joyosity/3595242078
http://www.flickr.com/photos/kyntharyn74/3262089319
http://www.flickr.com/photos/rexipe/826987087
http://www.flickr.com/photos/lablasco/6815671096
https://thenounproject.com/term/iphone-profile/54906/
https://thenounproject.com/term/browser/59091/