Optimizing Your App for Today’s Internet

Session 714

Stuart Cheshire, Apple DEST
Jiten Mehta, Apple CFNetwork Engineer
Internet growth and Internet Protocol version 6 (IPv6)
Explicit Congestion Notification (ECN)
Multipath TCP (MPTCP)
TCP Fast Open (TFO)
Quick UDP Internet Connections (QUIC)
DNS performance
Avoid “preflight checks”
Transport Layer Security (TLS) 1.3
Certificate Transparency
Bonjour Conformance Test
API choices and guidance
Details on URLSession
Mobile Data Around the World

Four billion people now use the Internet
• Over half the world’s population
• Worldwide, growth rate of human users is slowing

But... Internet growth continues nonetheless
• Machine-to-Machine, Internet of Things, Smart Homes
• China, India
• Smartphones, mobile data
Pay Attention to Less-Than-Ideal Networks

2G mobile networks are still common

Network Link Conditioner (NLC) is your friend
- Approximates real-world network conditions

Use tools like Wireshark and tcptrace to analyze
## IPv6 Availability

Percentage of connections made on a network that offered IPv6 connectivity

<table>
<thead>
<tr>
<th></th>
<th>Wi-Fi</th>
<th>Mobile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>29%</td>
<td>44%</td>
</tr>
<tr>
<td>US</td>
<td>39%</td>
<td>87%</td>
</tr>
<tr>
<td>UK</td>
<td>32%</td>
<td>0.12%</td>
</tr>
<tr>
<td>France</td>
<td>35%</td>
<td>0.03%</td>
</tr>
<tr>
<td>Germany</td>
<td>50%</td>
<td>34%</td>
</tr>
<tr>
<td>Belgium</td>
<td>60%</td>
<td>0.05%</td>
</tr>
<tr>
<td>India</td>
<td>34%</td>
<td>29%</td>
</tr>
</tbody>
</table>
India Cellular RTT Values

Cumulative Distribution Function

- IPv6 TCP Handshake
- IPv4 TCP Handshake
- IPv6 Connections
- IPv4 Connections

- <25 ms
- <50 ms
- <75 ms
- <100 ms
- <125 ms
- <150 ms
- <175 ms
- <200 ms
- <225 ms
- <250 ms
- <275 ms
- <300 ms
- <325 ms
- <350 ms
- <375 ms
- <400 ms
- <425 ms
- <450 ms
- <475 ms
- <500 ms

IPv6 Connections: 11%
IPv4 Connections: 89%
Explicit Congestion Notification (ECN)
Survey of Alexa top million web sites

35% 56% 70% 74% 77%

2013 2014 2015 2016 2017 2018

Your App and Next Generation Networks

WWDC 2015
Multipath TCP (MPTCP)

Faster connection fail-over

Used with proxies today

Used end-to-end with services like Siri since 2013

Mobile Carriers

- Multipath TCP works on 78 percent of carrier networks worldwide
- 22 percent of carrier networks block Multipath TCP
TCP Fast Open (TFO)

Avoids TCP three-way handshake connection setup time
Quick UDP Internet Connections (QUIC)

New transport protocol
• Potential successor to TCP
• Runs over unreliable datagram layer
• Provides reliable, congestion-controlled streams

IETF Standardization in progress

Apple engineers at meeting in Sweden right now
DNS Performance

Many web sites use short lifetimes on the DNS records
  • Sixty seconds or less
  • Enables fast fail-over

But data centers rarely go down
  • Use of fast fail-over capability is rare
  • IP addresses often don’t change from one lookup to the next
  • Adds 250 ms delay for no reason
  • Some ISPs cache DNS answers for much longer than sixty seconds anyway
Optimistic DNS

Currently used by CloudKit

In DNSServiceQueryRecord or DNSServiceGetAddrInfo, opt in by setting:
• kDNSServiceFlagsAllowExpiredAnswers

Use in conjunction with Happy Eyeballs (RFC 6555)
• Start connection attempt(s) with the address(es) you have now
• If DNS reports additional addresses later, try those too
Avoid SCNetworkReachability
Poor design pattern

Check SCNetworkReachability

Attempt connection

If failure, go to step one
Avoid SCNetworkReachability
Good design pattern

Just connect, using `waitsForConnectivity`
Transport Layer Security (TLS) 1.3

Improved security

Reduced connection setup time

Draft 28 approved by IESG on 21 March 2018

In RFC Editor queue to be published as an IETF RFC this summer
Transport Layer Security (TLS) 1.3

This final version is already in your WWDC seed of iOS 12 for testing


```
sudo defaults write /Library/Preferences/com.apple.networkd tcp_connect_enable_tls13 1
```
Certificate Transparency

Background

Public verifiable logs of issued certificates
Anyone can submit a certificate to a log
Client checks for proof that certificate has been logged
Certificate Transparency

How it works

Certificate Authority → Certificate → Log → Certificate → Server → Certificate → Client
Certificate Transparency

How it works

Certificate Authority

Server

Attacker’s Server

Certificate rejected by client

Client

Certificate rejected by client
Certificate Transparency
New policy

Starting in late 2018:
• All newly issued TLS certificates from publicly trusted CAs must be CT-validated

Currently issued certificates are unaffected

Clients are unaffected

What’s New in Security

<table>
<thead>
<tr>
<th>WWDC 2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>What’s New in Security</td>
</tr>
<tr>
<td>Your Apps and Evolving Network Security Standards</td>
</tr>
<tr>
<td>WWDC 2017</td>
</tr>
</tbody>
</table>
Bonjour Conformance Test

Bonjour Conformance Test certification is required for:
• Using the Bonjour name and logo with your hardware product (no charge)
• Bundling the Bonjour for Windows installer with your Windows application
• AirPrint, AirPlay, CarPlay, HomeKit devices

Passing the Bonjour Conformance Test:
• Helps improve your product reliability
• Makes your customers happy
API Choices

Application

Third-Party Library

URLSession

BSD Sockets
API Choices

Introducing Network.framework: A modern alternative to Sockets

Hall 3
Thursday 11:00AM
Avoid BSD Sockets

Avoid third-party libraries that use BSD Sockets

If you are the author of a library that uses BSD Sockets, CFSocketStream, or SecureTransport

• Please look at switching to Network.framework
• Contact Developer Relations with feedback
URLSession

Optimizing your app for best performance

Jiten Mehta, Apple CFNetwork Engineer
Latency
Throughput
Responsiveness
System resources
URLSession

High-level Foundation networking API
• HTTP/2, HTTP/1.1
• In-process and out-of-process transfers
• Handles cookies, cache, authentication, and proxies
• URLSessionStreamTask for TCP connections

Recommended API for all Apple platforms
Latency
Throughput
Responsiveness
System resources
Issues with HTTP/1.1
HTTP/2

Time

HTTP/1.1 Connection

HTTP/2 Connection

info.json
logo.png
data.xml

Idle  Setup  Request  Response
Upgrade to HTTP/2 Today

No head-of-line blocking at HTTP layer
Better bandwidth utilization
No client-side changes
Server-side savings
HTTP/2 Connection Coalescing

Reuses connections for multiple hosts when the following conditions are met:
• IP addresses match
• Hostnames covered by same TLS certificate

New URLSession behavior
HTTP/2 Connection Coalescing

Old behavior

menu.example.com

*.example.com

delivery.example.com

*.example.com

2001:db8:1984::1
HTTP/2 Connection Coalescing

New behavior

menu.example.com
delivery.example.com
*.example.com

2001:db8::1
Fewer URLSession Objects

Connection reuse

Especially important for HTTP/2

Creating URLSession objects can be expensive
Latency

Throughput

Responsiveness

System resources
Reduce Request Size

HTTP cookies
- Specify domain and path
- Smaller cookies
- Delete unwanted cookies or set expiry
- Save state on server

HTTP/2 header compression
Use HTTP Compression

Gzip
• HTTP and HTTPS

Brotli
• HTTPS only
• Optimized for text and HTML
Latency
Throughput
Responsiveness
System resources
Quality of Service (QoS) Classes

- User-Interactive
- User-Initiated
- Default
- Utility
- Background

URLSession is QoS-aware

Dispatch queue QoS is captured at `task.resume()`

Delegate messages respect your QoS
Network Service Type

Use default and background for majority

Hint for network switches on Cisco Fast Lane

voice
callsSignaling
video
responseData
default
background
URLSession Adaptable Connectivity

Requests are sent as soon as we have connectivity

Enable `waitsForConnectivity` for every URLSession

// Optionally implement the delegate method
func URLSession(_ session: URLSession, taskIsWaitingForConnectivity task: URLSessionTask) {
    // Present fallback UI or error message
}

Call `task.cancel()` if resource is no longer needed

Advances in Networking, Part 2
Latency
Throughput
Responsiveness
System resources
Use Background Sessions

Utilizes system intelligence

Transfers continue when your app is not running

// Optionally set isDiscretionary for non-urgent downloads
let configuration = URLSessionConfiguration.background(withIdentifier: "com.apple.example")
configuration.isDiscretionary = true
Cache Wisely

Ephemeral sessions
Don’t cache unique content

// Implement delegate method to decide when to cache content
func URLSession(_ session: URLSession, dataTask: URLSessionDataTask,
    willCacheResponse proposedResponse: CachedURLResponse,
    completionHandler: @escaping (CachedURLResponse?) -> Void) {
    // If you don't want to cache
    completionHandler(nil)
}

Server-side Cache-Control: no-store header
## Recap

<table>
<thead>
<tr>
<th>Advice</th>
<th>Latency</th>
<th>Throughput</th>
<th>Responsiveness</th>
<th>System Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move to HTTP/2</td>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fewer URLSessions</td>
<td>✔️</td>
<td></td>
<td></td>
<td>✔️</td>
</tr>
<tr>
<td>Reduce request size</td>
<td></td>
<td>✔️</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QoS</td>
<td></td>
<td></td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>Background sessions</td>
<td></td>
<td></td>
<td></td>
<td>✔️</td>
</tr>
<tr>
<td>Event</td>
<td>Location</td>
<td>Date and Time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------</td>
<td>---------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Introducing Network.framework: A modern alternative to Sockets</td>
<td>Hall 3</td>
<td>Thursday 11:00AM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Networking Lab</td>
<td></td>
<td>Thursday 2:00PM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Networking Lab</td>
<td></td>
<td>Friday 9:00AM</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

More Information

https://developer.apple.com/wwdc18/714