

Networking with NSURLSession

Session 711

Luke Case Software Engineer

Andreas Garkuscha Software Engineer

Dan Vinegrad Software Engineer

App Transport Security

New protocol support in NSURLSession

NSURLSession on watchOS

NSURLSession API changes

NSURLSession

API for downloading HTTP content

Rich set of delegate methods

Background download/upload capability

HTTP

(cleartext)



HTTP

(cleartext)



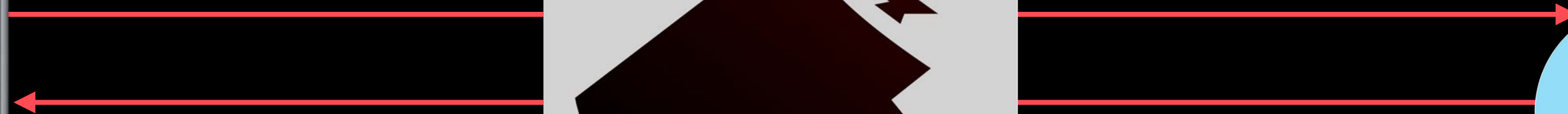
GET

HTTP/1.1 200 OK



HTTP

(cleartext)



HTTP
(cleartext)

HTTP

(cleartext)

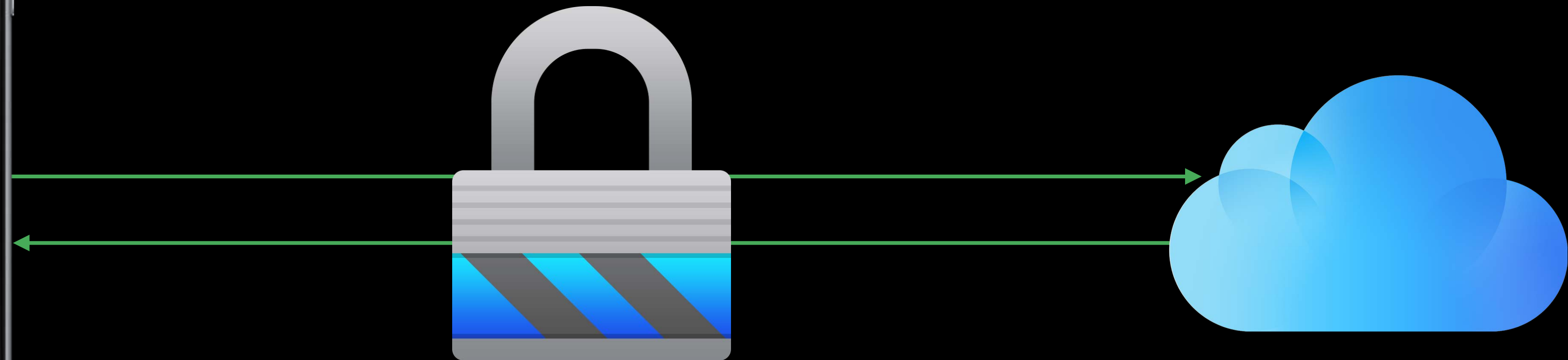




HTTPS



HTTPS



HTTPS

HTTP over TLS



Encryption

Integrity

Authentication

Protecting Customer Data

HTTPS: securing HTTP with TLS



"http://" to "https://"

Server support required

Widespread support available

All data is sensitive

App Transport Security

Protecting Customer Data

NEW

What is App Transport Security (ATS)?

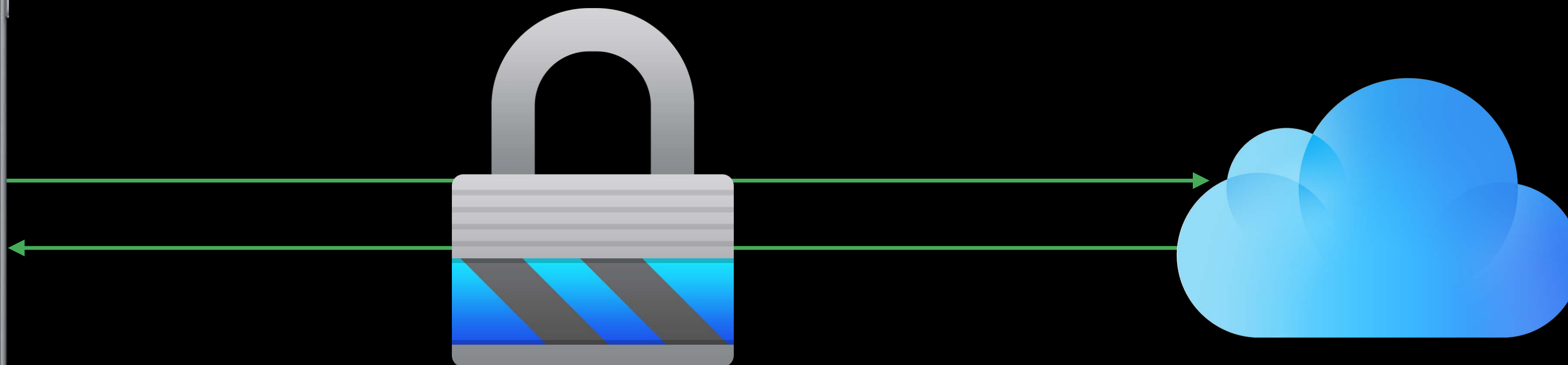
Disallows cleartext HTTP URL loads

Encourages best practice secure connections

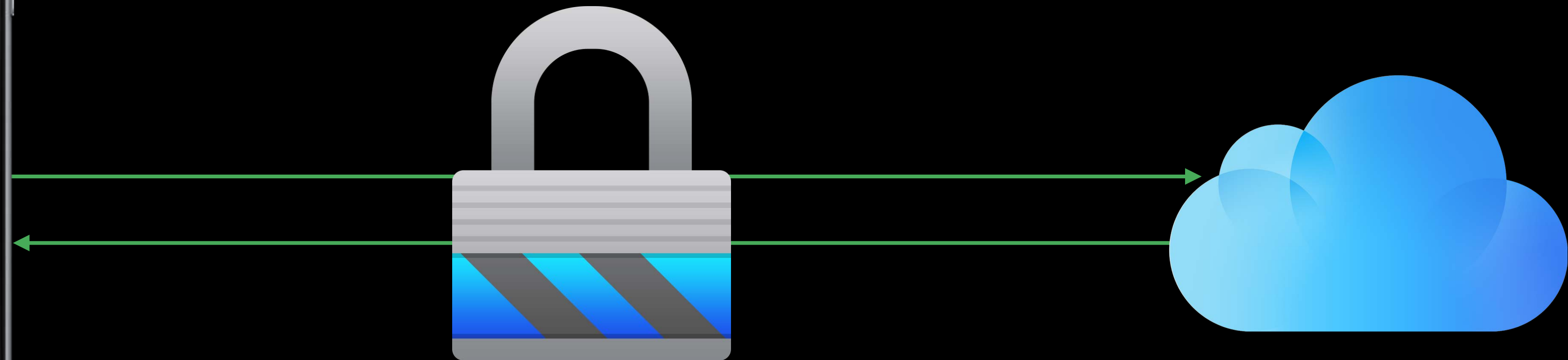
Defaults to stronger security

Allows configuration via app's Info.plist

▼ NSAppTransportSecurity	⬆	Dictionary	(1 item)
▼ NSExceptionDomains		Dictionary	(2 items)
▼ media.example.com		Dictionary	(2 items)
NSIncludesSubdomains		Boolean	YES
NSExceptionAllowsInsecureHTTPLoads		Boolean	YES
▼ other.example.com		Dictionary	(2 items)
NSExceptionRequiresForwardSecrecy		Boolean	NO
NSExceptionMinimumTLSVersion		String	TLSv1.1



HTTPS



NSAppTransportSecurity

NSExceptionDomains

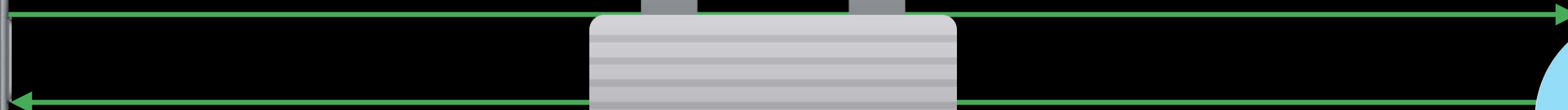
"example.com"

NSIncludesSubdomains = YES

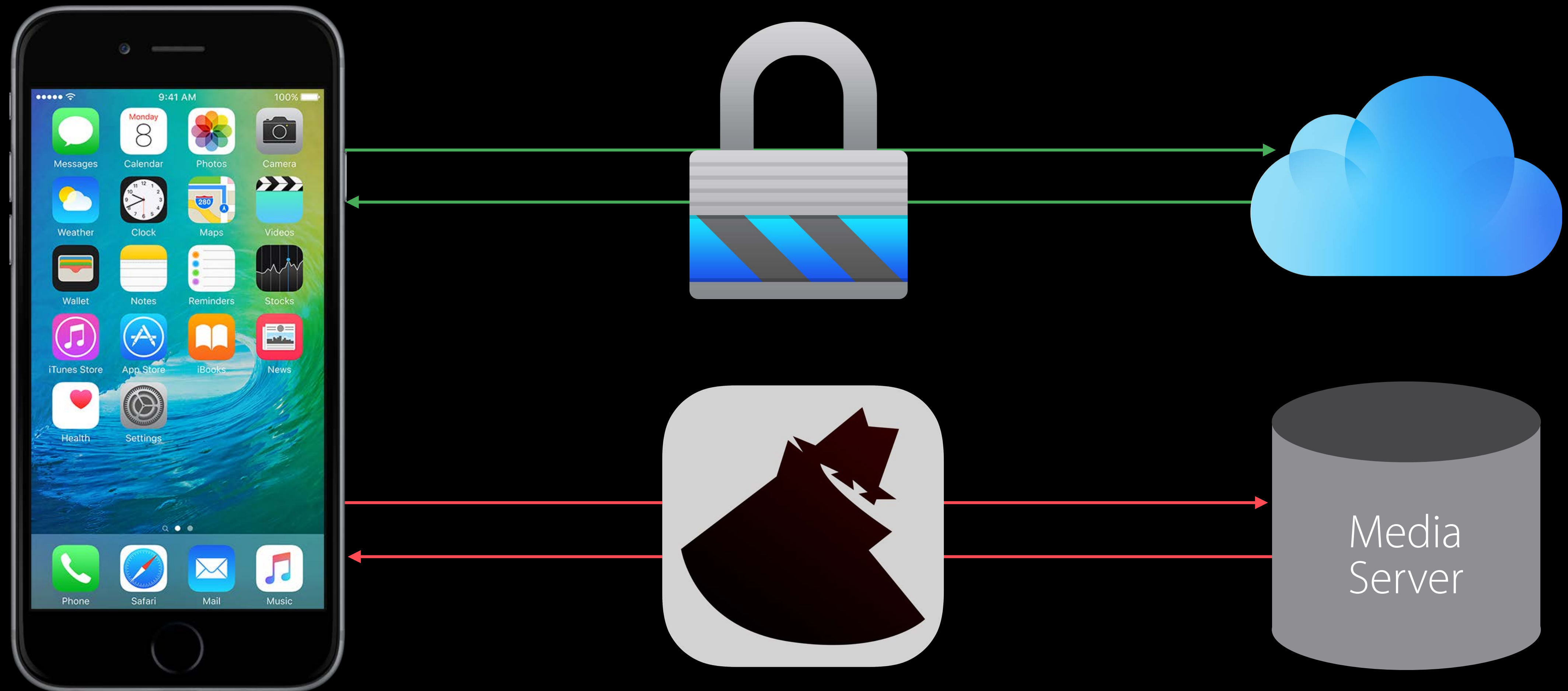
NSExceptionRequiresForwardSecrecy = NO

NSExceptionMinimumTLSVersion = "TLSv1.1"

HTTPS



HTTPS

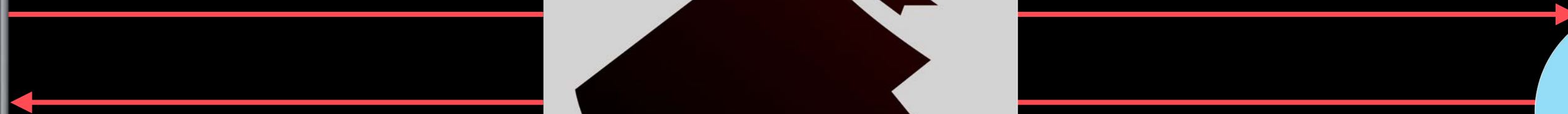


```
NSAppTransportSecurity
```

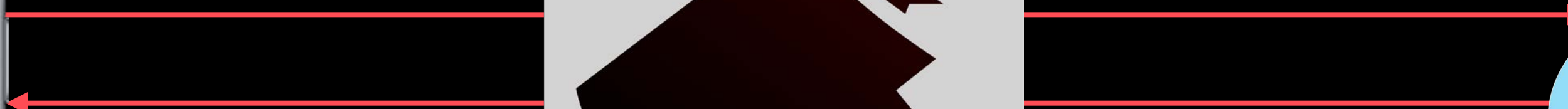
```
    NSExceptionDomains
```

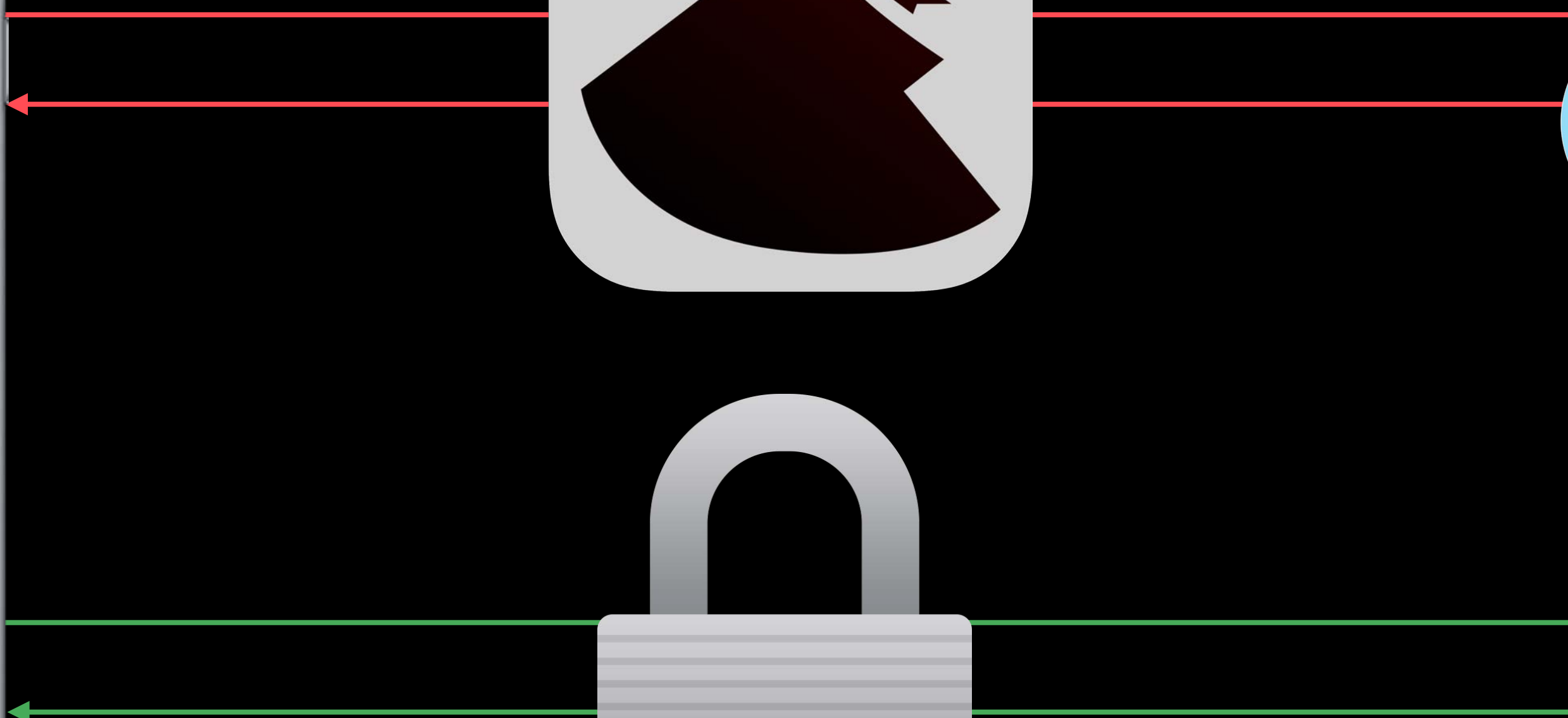
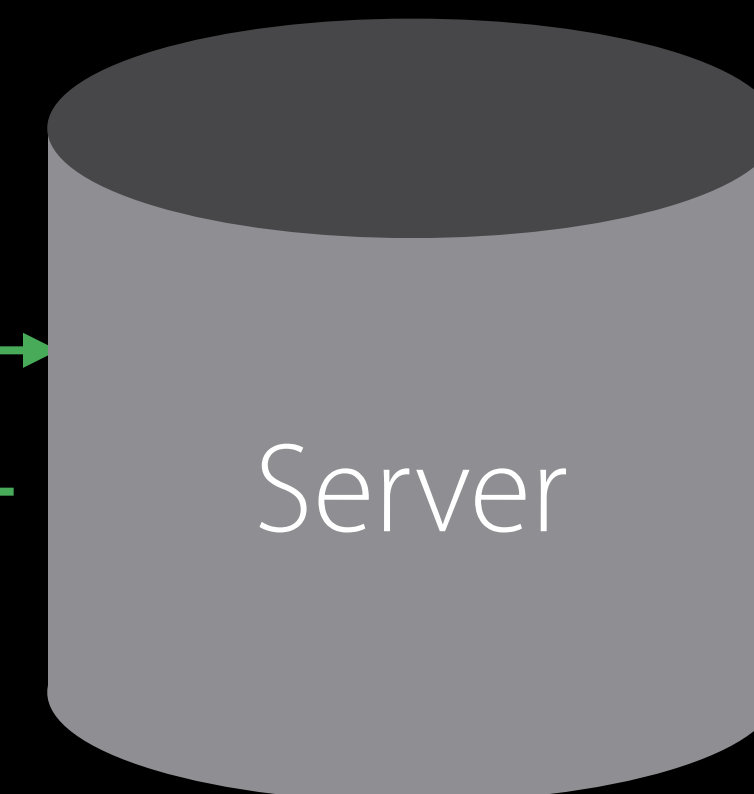
```
        "media.example.com"
```

```
            NSExceptionAllowsInsecureHTTPLoads = YES
```

NSAppTransportSecurity
NSAllowsArbitraryLoads = YES






```
NSAppTransportSecurity  
  NSAllowsArbitraryLoads
```

```
  NSExceptionDomains
```

```
    "secure.example.com"
```

```
    NSExceptionAllowsInsecureHTTPLoads = NO
```

Configuring ATS

Diagnosing issues

Active when you build your app for iOS 9 and OS X El Capitan

"http://" to "https://" is automatic

Use **NSAllowsArbitraryLoads** for quick triage

Log NSURLSession errors

CFNETWORK_DIAGNOSTICS=1

Protecting Customer Data

Next steps

Use HTTPS for new projects

Transition existing apps from HTTP to HTTPS

Use exceptions where needed

New Protocol Support in NSURLSession

Andreas Garkuscha
Software Engineer

New in NSURLSession

NEW

New in URLSession

NEW

HTTP/2

RFC 7540

New in URLSession

NEW

HTTP/2

RFC 7540

New in NSURLSession

NEW

New in NSURLSession

NEW

New in NSURLSession

NEW

Future of the Web

HTTP/2

Major milestone
in the evolution
of the Web

Solves HTTP/1.1
problems

RFC 7540

Makes the Web
faster

HTTP/2 in NSURLSession

Why a new protocol?

HTTP/2 in depth

Adoption using NSURLSession

Why a New Protocol?

Why a New Protocol?

Why a New Protocol?

FTP HTTP POP3 LDAP RTP

SMTP NTP TCP IMAP UDP

ARP SNMP PPP SOAP ICMP

IP APX Telnet RADIUS VNC

SSH VoIP AFP smb nfs

DHCP SKYPE RDP WebSocket RIP

AIM Bonjour ICQ QUAKE3 ...

Why a New Protocol?

FTP HTTP POP3 LDAP RTP

SMTP NTP TCP IMAP UDP

ARP SNMP PPP SOAP ICMP

IP APX Telnet RADIUS VNC

SSH VoIP AFP smb nfs

DHCP SKYPE RDP WebSocket RIP

AIM Bonjour ICQ QUAKE3 ...

The screenshot shows the Apple website homepage from July 14, 1997. The header features the Apple logo and the text "Welcome to Apple 1997". A red navigation bar on the left contains links: "Find It", "Product Information", "Customer Support", "Technology & Research", "Developer World", "Groups & Interests", "Resources Online", and "About Apple". Below this is a section for "Apple Sites Worldwide" with a list of countries (Asia, Australia, Belgium, Canada, Chile) and a "Go" button. The main content area includes a "What's Hot" section with three articles: "Preorder Mac OS 8" (describing the new Mac OS version), "Be the First to Know" (announcing new Macintosh software releases), and "Want a PowerBook?" (a contest to win a PowerBook 3400/200). A sidebar on the right promotes "EMATE 300" (a mobile device) and "MOVIES FROM MARS" (a QuickTime VR experience). At the bottom, there are links for "Where to Buy", "Register to Win", "Software Updates", and "Home Page Archives".

Find It

Product Information

Customer Support

Technology & Research

Developer World

Groups & Interests

Resources Online

About Apple

Apple Sites Worldwide

Asia
Australia
Belgium
Canada
Chile

Go

Where to Buy

Register to Win

Software Updates

Home Page Archives

What's Hot

Preorder Mac OS 8
Now you can [preorder Mac OS 8](#), described by Macworld as "the most comprehensive update to the Mac OS in years, sporting a bold new look, a speedier Finder, more shortcuts and integrated Internet functions."

Want a PowerBook?
Qualify to win a [PowerBook 3400/200](#) by [entering](#) this month's Apple Registration Sweepstakes.

Be the First to Know
Learn about new Macintosh software releases the moment they become available. Check [Hot Mac Products](#) to hear about programs like Speed Demon, ReBirth RB-338 and QuickCRC.

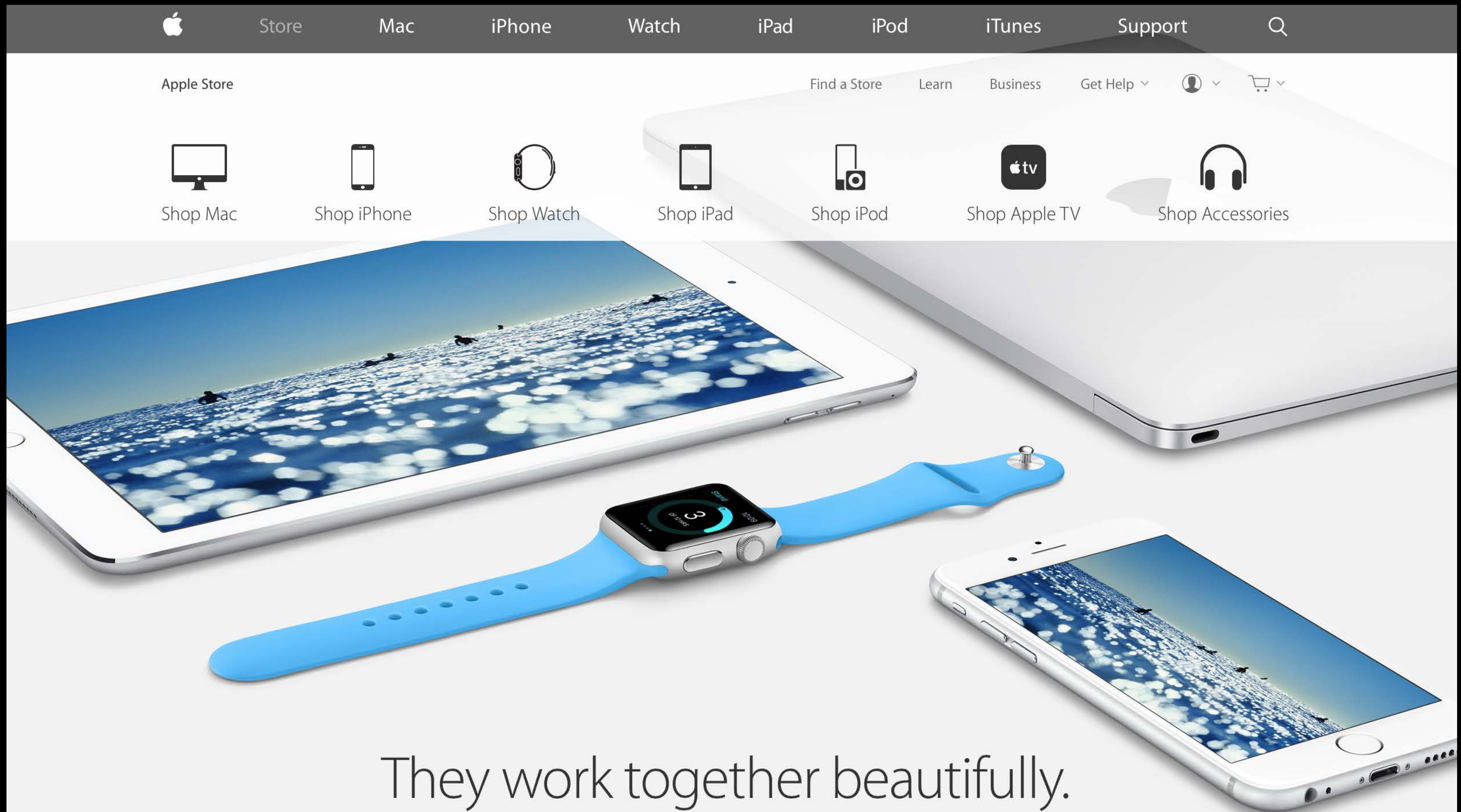
Newton Connects
Newton, Inc., will enhance network connectivity for Newton-based devices this fall via [Newton Internet Enabler 2.0](#). Ethernet capability can connect devices to Local Area Networks.

EMATE 300
Mobile, Affordable, & Smart

MOVIES FROM MARS
QuickTime VR Takes You Out of this World

Introducing CyberDrive
Register today for a free CD-ROM.

Why a New Protocol?



HTTP/1.1 Common Issues

HTTP/1.1

Common issues

One outstanding request

HTTP/1.1

Common issues

One outstanding request

HTTP pipelining

HTTP/1.1

Common issues

One outstanding request

HTTP pipelining

Multiple connections

HTTP/1.1

Common issues

One outstanding request

HTTP pipelining

Multiple connections

Textual protocol overhead

HTTP/1.1

Common issues

One outstanding request

HTTP pipelining

Multiple connections

Textual protocol overhead

Header compression

The Path to HTTP/2

Experimental protocols

Specification for HTTP/2

IETF standardization, RFC

New in NSURLSession

NEW

New in URLSession

NEW

HTTP/2

RFC 7540

HTTP/1.1

HTTP/2

HTTP/1.1

Multiple TCP connections to a host

HTTP/2

Only one TCP connection



HTTP/1.1

HTTP/2

Multiple TCP connections to a host

Only one TCP connection



Head-of-line blocking

Fully multiplexed



HTTP/1.1

Multiple TCP connections to a host

Head-of-line blocking

FIFO restrictions

HTTP/2

Only one TCP connection

Fully multiplexed

Requests have priorities



HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time 

HTTP/1.1: Head-of-Line Blocking (without Pipelining)

HTTP/2: Multiplexing

Priority Legend

Low

Medium

High

HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time 

HTTP/1.1: Head-of-Line Blocking (without Pipelining)

image.jpg

styles.css

data.xml

HTTP/2: Multiplexing

Priority Legend

Low

Medium

High

HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time 

HTTP/1.1: Head-of-Line Blocking (without Pipelining)

image.jpg GET

styles.css

data.xml

HTTP/2: Multiplexing

Priority Legend

Low

Medium

High

HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time 

HTTP/1.1: Head-of-Line Blocking (without Pipelining)

image.jpg GET 200 OK ✓

styles.css

data.xml

HTTP/2: Multiplexing

Priority Legend

Low

Medium

High

HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time →

HTTP/1.1: Head-of-Line Blocking (without Pipelining)

image.jpg



styles.css



data.xml

HTTP/2: Multiplexing

Priority Legend

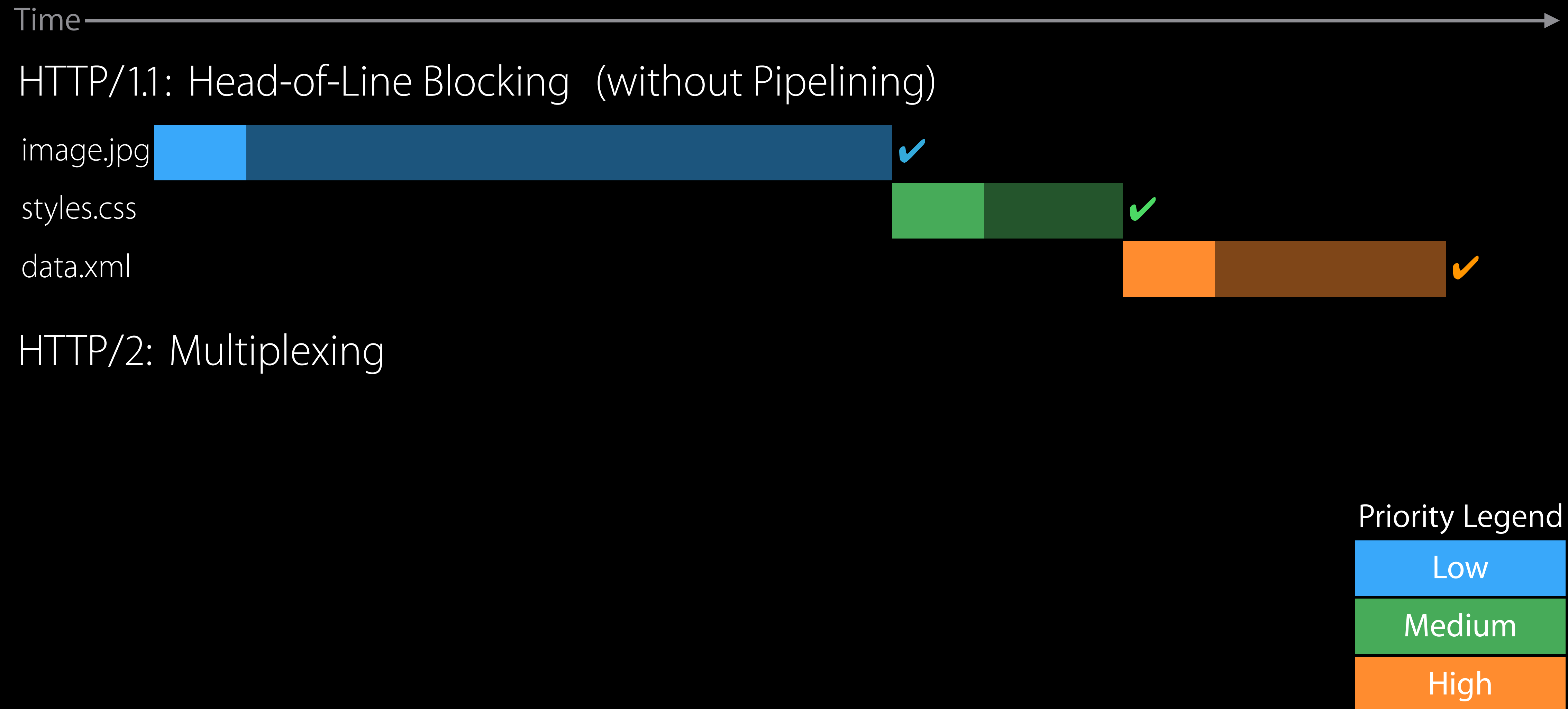
Low

Medium

High

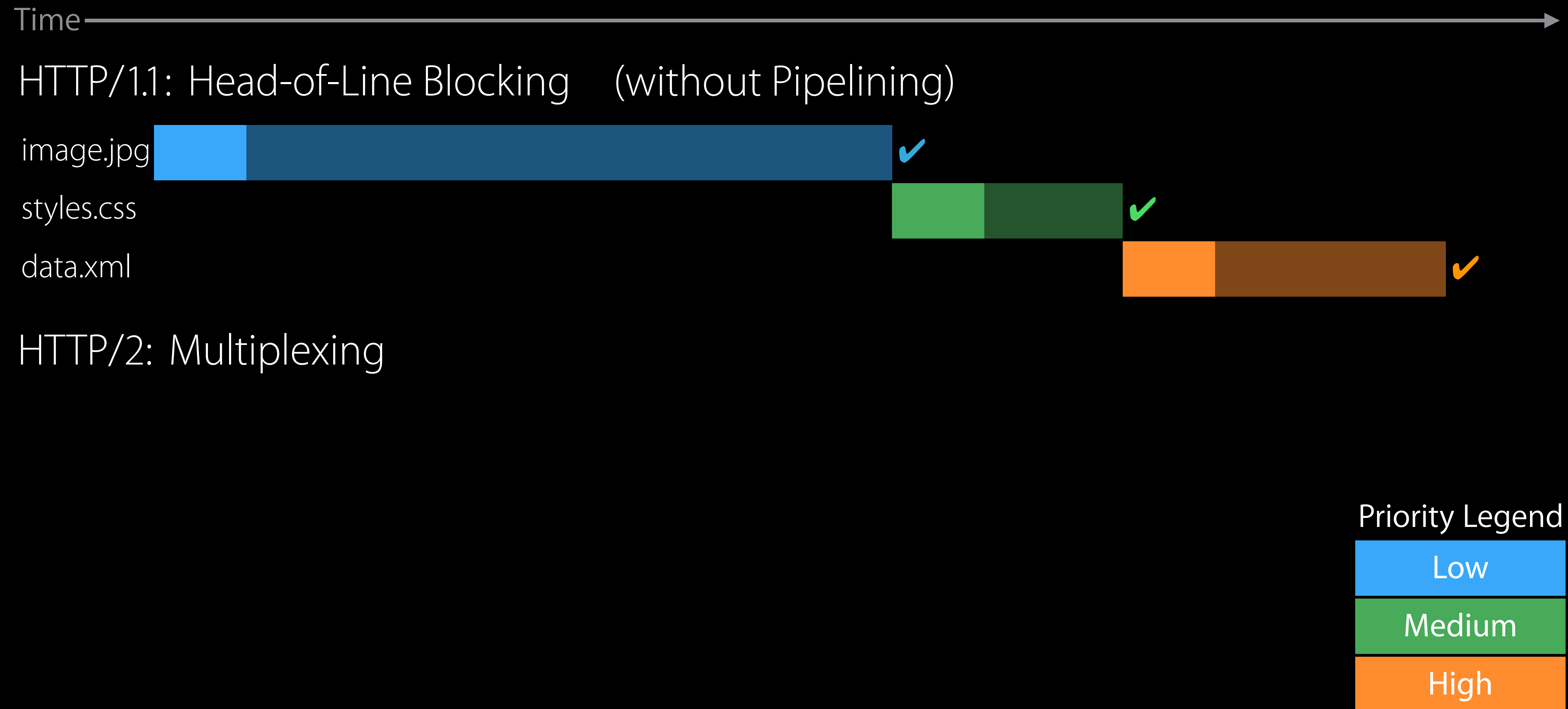
HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem



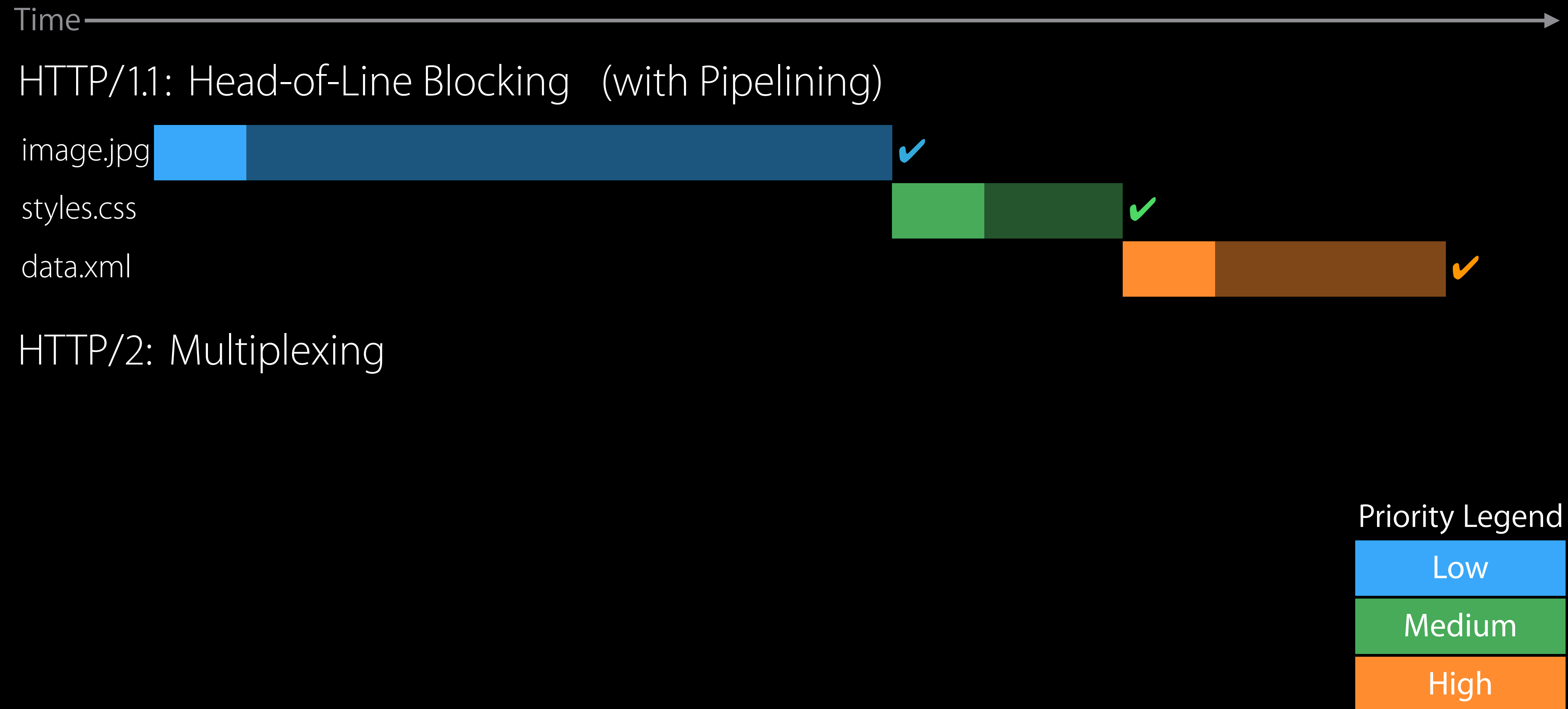
HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem



HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem



HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

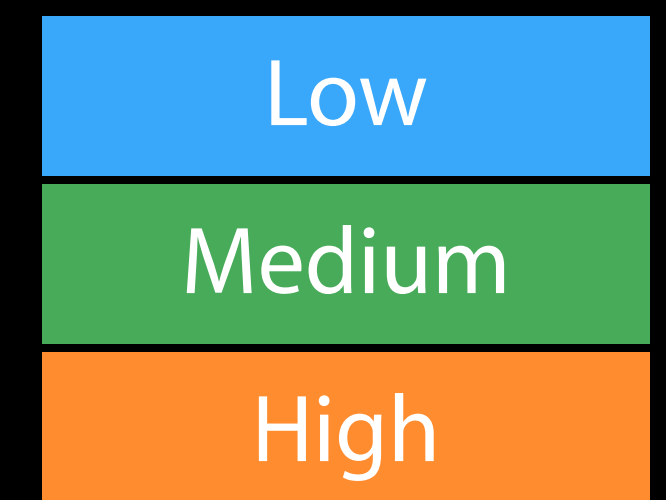
Time →

HTTP/1.1: Head-of-Line Blocking (with Pipelining)



HTTP/2: Multiplexing

Priority Legend



HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time →

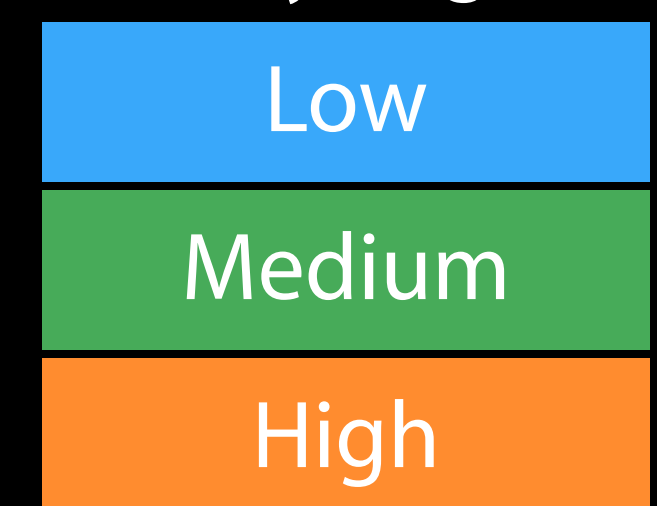
HTTP/1.1: Head-of-Line Blocking (with Pipelining)



HTTP/2: Multiplexing

image.jpg
styles.css
data.xml

Priority Legend



HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time →

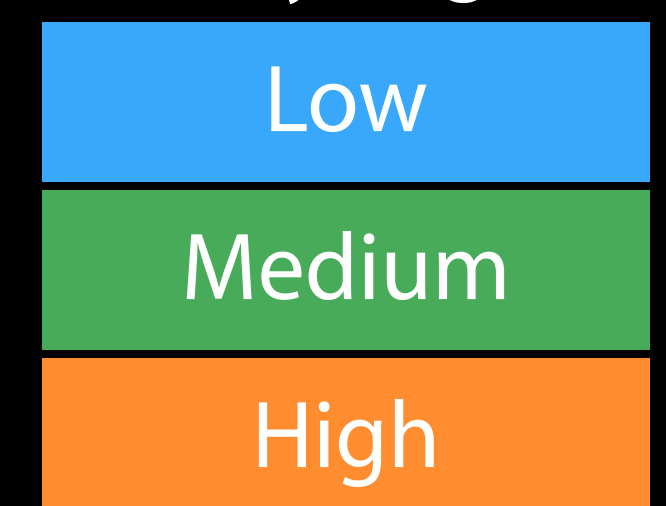
HTTP/1.1: Head-of-Line Blocking (with Pipelining)



HTTP/2: Multiplexing



Priority Legend



HTTP/2 - Multiplexing

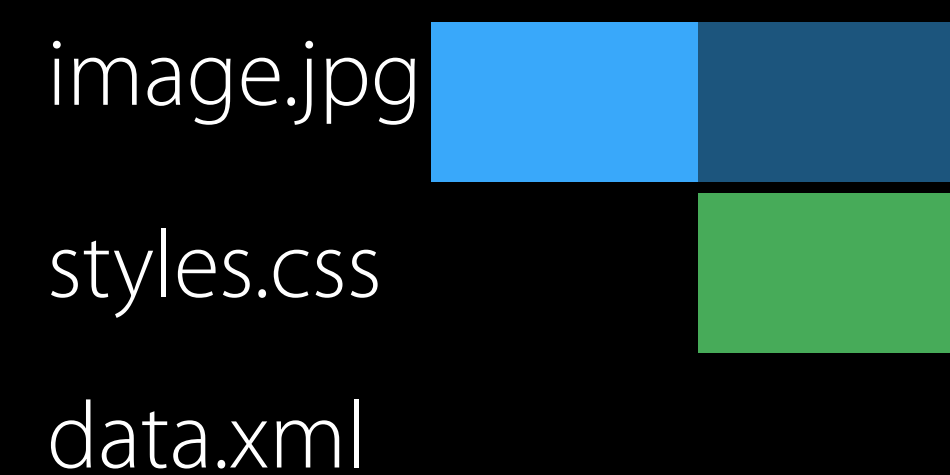
Solution for Head-of-Line Blocking problem

Time →

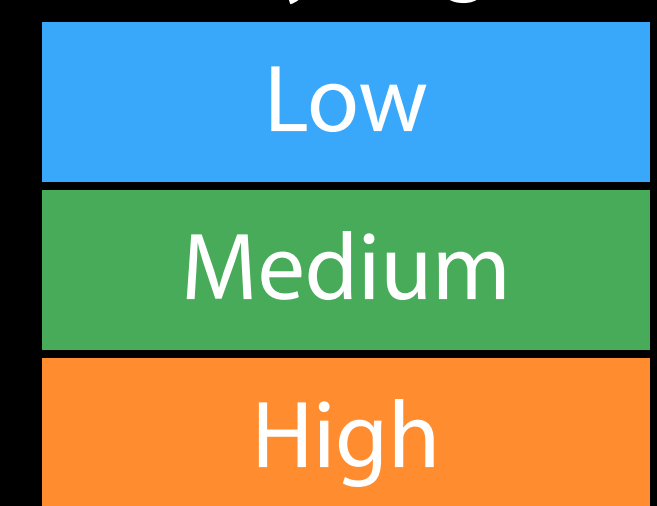
HTTP/1.1: Head-of-Line Blocking (with Pipelining)



HTTP/2: Multiplexing



Priority Legend



HTTP/2 - Multiplexing

Solution for Head-of-Line Blocking problem

Time →

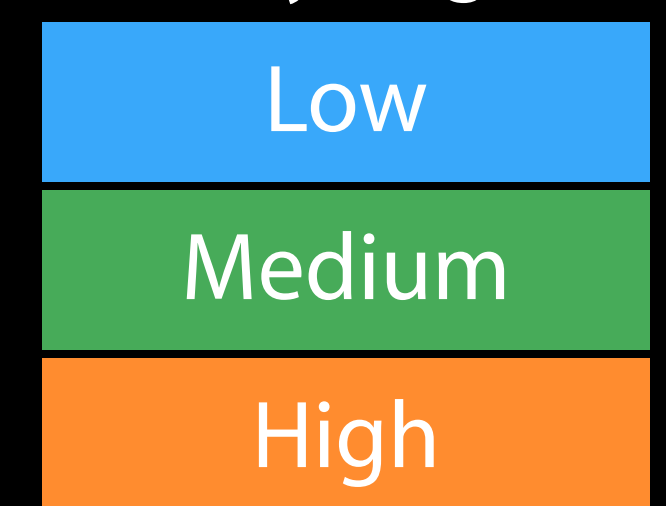
HTTP/1.1: Head-of-Line Blocking (with Pipelining)



HTTP/2: Multiplexing



Priority Legend



HTTP/2 - Multiplexing

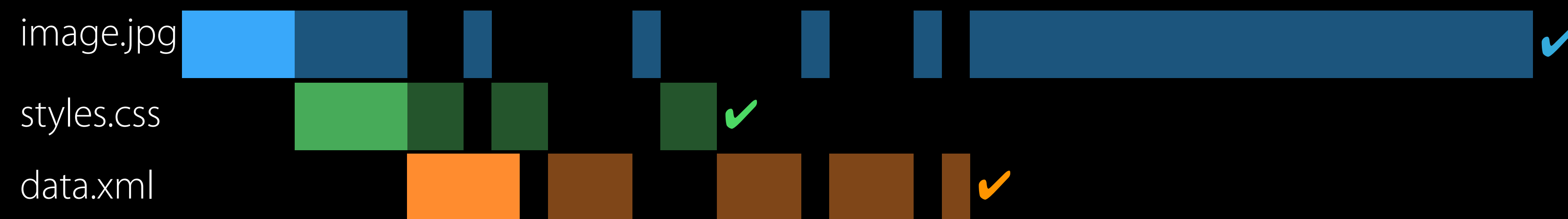
Solution for Head-of-Line Blocking problem

Time →

HTTP/1.1: Head-of-Line Blocking (with Pipelining)



HTTP/2: Multiplexing



HTTP/1.1

HTTP/2

Multiple TCP connections to a host

Only one TCP connection



Head-of-line blocking





Fully multiplexed



FIFO restrictions

Requests have priorities



HTTP/1.1	HTTP/2
Multiple TCP connections to a host	Only one TCP connection 
Head-of-line blocking	Fully multiplexed 
FIFO restrictions	Requests have priorities 
Textual protocol overhead	Binary 

HTTP/1.1

Multiple TCP connections to a host

Head-of-line blocking

FIFO restrictions

Textual protocol overhead

No header compression

HTTP/2

Only one TCP connection

Fully multiplexed

Requests have priorities

Binary

Header compression (HPACK)



HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Representation:

:method: GET
:scheme: https
:path: /
:authority: www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Representation:

:method:	GET
:scheme:	https
:path:	/
:authority:	www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Representation:

:method: GET
:scheme: https
:path: /
:authority: www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Representation:

:method: GET
:scheme: https
:path: /
:authority: www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Representation:

:method: GET
:scheme: https
:path: /
:authority: www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Representation:

:method: GET
:scheme: https
:path: /
:authority: www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	


Dynamic Table

Index	Header Name	Header Value
-		

HTTP/1.1 Request:

GET / HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Representation:

:method: GET
:scheme: https
:path: /
:authority: www.example.com
www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
1	:authority	www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
1	:authority	www.example.com

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
1	:authority	www.example.com

HTTP/1.1 Second request:

GET /index.html HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
1	:authority	www.example.com

HTTP/1.1 Second request:

```
GET /index.html HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n
```

HTTP/2 Second request:

```
:method: GET
:scheme: https
:path: /index.html
:authority: www.example.com
```


HTTP/2 Header Compression

HPACK Static Table

Index	Header Name	Header Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
...	...	
14	:status	500
15	accept-charset	
16	accept-encoding	gzip, deflate
17	accept-language	
...	...	
57	transfer-encoding	
58	user-agent	
59	vary	
60	via	
61	www-authenticate	

Dynamic Table

Index	Header Name	Header Value
1	:authority	www.example.com

HTTP/1.1 Second request:

GET /index.html HTTP/1.1\r\n
Host: www.example.com\r\n
Connection: Keep-Alive\r\n\r\n

HTTP/2 Second request:

□□□□

HTTP/2

Adoption on the client

HTTP/2

Adoption on the client

NSURLSession supports HTTP/2 protocol automatically

HTTP/2

Adoption on the client

NSURLSession supports HTTP/2 protocol automatically

No source code changes needed

HTTP/2

Adoption on the client

NSURLSession supports HTTP/2 protocol automatically

No source code changes needed

```
let sessionConfig = NSURLSessionConfiguration.defaultSessionConfiguration()
let session = NSURLSession(configuration: sessionConfig)

let url = NSURL(string: "https://www.example.com")!
let task = session.dataTaskWithURL(url) {
    (data: NSData?, response: NSURLResponse?, error: NSError?) in
        ...
}

task?.resume()
```

You Only Need an HTTP/2 Server

HTTP/2 in NSURLSession

Compatibility

Backward compatible with HTTP/1.1 servers

Encrypted connection only

HTTP/2 server requires ALPN or NPN support

HTTP/2 Today

Current adoption

HomeKit Remote Access via iCloud

Services provided by Google and Twitter

Open source web servers

Content Delivery Network providers

HTTP/2

At a glance

HTTP/2 is available in WWDC seed today

Available in NSURLSession

Enabled in Safari on OS X 10.11 and iOS 9

NSURLSession on watchOS

Dan Vinegrad
Software Engineer

NSURLSession on watchOS

NEW

Fully supported for HTTP(S)

Uses best available connectivity (iPhone or WiFi)

NSURLSession on watchOS

NEW

Fully supported for HTTP(S)

Uses best available connectivity (iPhone or WiFi)

It's Magic

NSURLSession on watchOS

Best practices

NSURLSession on watchOS

Best practices

Download the minimal assets required

- Small screen
- Limited bandwidth

NSURLSession on watchOS

Best practices

Download the minimal assets required

- Small screen
- Limited bandwidth

Apps will only run for a very short time

- Send small amounts of data
- Use background uploads/downloads for larger content

API Changes

NSURLConnection

NSURLConnection



Deprecated in OS X 10.11 and iOS 9.0

- Not going away, apps using it will still work
- New features will be added to NSURLSession

Not supported on watchOS

Switching to NSURLSession is easy

Switching to NSURLSession

```
let queue: NSOperationQueue = ...
let url = NSURL(string: "https://www.example.com")!
let request = NSURLRequest(URL: url)

NSURLSession.sendAsynchronousRequest(request, queue: queue) {
    (response: NSURLResponse?, data: NSData?, error: NSError?) in
        ...
}
```

Switching to NSURLSession

```
let queue: NSOperationQueue = ...
let url = NSURL(string: "https://www.example.com")!
let request = NSURLRequest(URL: url)

NSURLConnection.sendAsynchronousRequest(request, queue: queue) {
    (response: NSURLResponse?, data: NSData?, error: NSError?) in
        ...
}

let task = NSURLSession.sharedSession().dataTaskWithRequest(request) {
    (data: NSData?, response: NSURLResponse?, error: NSError?) in
        ...
}
task?.resume()
```

New NSURLSession API

Sharing Cookies

NEW

Between Apps and Extensions

Apps and their extensions have different cookie storages by default

Can use application groups to get a shared data container

OS X 10.11 and iOS 9 introduce new API to create a shared cookie storage

New NSHTTPCookieStorage API

NEW

```
let ident = "group.mycompany.mygroupname"
let cookieStorage =
    NSHTTPCookieStorage.sharedCookieStorageForGroupContainerIdentifier(
        identifier: ident)

let config = NSURLSessionConfiguration.defaultSessionConfiguration()
config.HTTPCookieStorage = cookieStorage
let session = NSURLSession(configuration: config)
```

NSURLSessionStreamTask

TCP/IP networking

NEW

NSURLSessionStreamTask

TCP/IP networking

Sometimes need a protocol other than HTTP(S)

NEW

NSURLSessionStreamTask

NEW

TCP/IP networking

Sometimes need a protocol other than HTTP(S)

Advantages over NSInputStream / NSOutputStream

- Asynchronous read/write interface
- Can automatically get through HTTP proxies
- New API goodies

NSURLSessionStreamTask

NEW

TCP/IP networking

Sometimes need a protocol other than HTTP(S)

Advantages over NSInputStream / NSOutputStream

- Asynchronous read/write interface
- Can automatically get through HTTP proxies
- New API goodies

Legacy NSStream support

NSURLSessionStreamTask

TCP/IP networking

NEW

NSURLSessionStreamTask

NEW

TCP/IP networking

Supports TCP/IP connections:

- Hostname and port
- NSNetService resolution

NSURLSessionStreamTask

NEW

TCP/IP networking

Supports TCP/IP connections:

- Hostname and port
- NSNetService resolution

Uses existing NSURLSession configuration and session delegates

NSURLSessionStreamTask

NEW

TCP/IP networking

Supports TCP/IP connections:

- Hostname and port
- NSNetService resolution

Uses existing NSURLSession configuration and session delegates

Secure TLS connections supported

- Even after connected!

NSURLSessionStreamTask

NEW

Asynchronous read

```
let task =
    URLSession.sharedSession().streamTaskWithHostName("chat.example.com",
                                                         port: 5555)!

task.resume()

task.readDataOfMinLength(16384, maxLength:65536, timeout: 30.0) {
    (data: NSData?, eof: Bool, error: NSError?) in
        ...
}
```

NSURLSessionStreamTask

NEW

Asynchronous write

```
let task =
    URLSession.sharedSession().streamTaskWithHostName("chat.example.com",
                                                        port: 5555)!

task.resume()

let data: NSData = ...
task.writeData(data, timeout: 30.0) { (error: NSError?) in
    ...
}
```

NSURLSessionStreamTask

NEW

Enabling TLS

```
let task =
    URLSession.sharedSession().streamTaskWithHostName("chat.example.com",
                                                        port: 5555)!

task.startSecureConnection()
task.resume()
let data: NSData = ...
task.writeData(data, timeout: 30.0) { (error: NSError?) in
    ...
}
```

NSURLSessionStreamTask

Conversion to NSStreams

NEW

NSURLSessionStreamTask

Conversion to NSStreams

NEW

Some existing APIs consume NSStream objects

NSURLSessionStreamTask

NEW

Conversion to NSStreams

Some existing APIs consume NSStream objects

Pending reads/writes complete before NSStreams are produced

NSURLSessionStreamTask

NEW

Conversion to NSStreams

Some existing APIs consume NSStream objects

Pending reads/writes complete before NSStreams are produced

Detaches the task from the session

NSURLSessionStreamTask

NSStream support

NEW

```
let streamTask: NSURLSessionStreamTask = ...
```

```
streamTask.captureStreams()
```

NSURLSessionStreamTask

NEW

NSStream support

```
func URLSession(session: NSURLSession, streamTask: NSURLSessionStreamTask,  
                didBecomeInputStream inputStream: NSInputStream,  
                outputStream: NSOutputStream) {  
  
    // Can pass inputStream / outputStream to APIs that consume NSStreams  
  
}
```

NSURLSessionStreamTask

NEW

Other (informational) delegate methods

```
optional func URLSession(session: NSURLSession, betterRouteDiscoveredForStreamTask  
streamTask: NSURLSessionStreamTask)
```

NSURLSessionStreamTask

NEW

Other (informational) delegate methods

```
optional func URLSession(session: NSURLSession, betterRouteDiscoveredForStreamTask  
streamTask: NSURLSessionStreamTask)
```

```
optional func URLSession(session: NSURLSession, readClosedForStreamTask streamTask:  
NSURLSessionStreamTask)
```

```
optional func URLSession(session: NSURLSession, writeClosedForStreamTask streamTask:  
NSURLSessionStreamTask)
```

NSURLSessionStreamTask

DataTask conversion

NEW

NSURLSessionStreamTask

NEW

DataTask conversion

NSURLSessionDataTask may be converted to a stream task

- Use NSURLSession to get through HTTP proxies

NSURLSessionStreamTask

NEW

DataTask conversion

NSURLSessionDataTask may be converted to a stream task

- Use NSURLSession to get through HTTP proxies

Conversion can occur when response is received

NSURLSessionStreamTask

NEW

DataTask conversion

```
func URLSession(session: NSURLSession, dataTask: NSURLSessionDataTask,  
                didReceiveResponse response: NSURLResponse,  
                completionHandler: (NSURLSessionResponseDisposition) -> Void) {  
  
    completionHandler(.BecomeStream)  
}  
  
func URLSession(session: NSURLSession, dataTask: NSURLSessionDataTask,  
                didBecomeStreamTask streamTask: NSURLSessionStreamTask) {  
  
}
```

Summary

App Transport Security

HTTP/2 support in NSURLSession

NSURLSession on watchOS

NSURLConnection deprecation

New API

- Group container cookie storages
- NSURLSessionStreamTask

More Information

Documentation

Developer Resources

<https://developer.apple.com>

Technical Support

Apple Developer Forums

<http://devforums.apple.com>

Developer Technical Support

<http://developer.apple.com/support/technical>

General Inquiries

Paul Danbold, Core OS Evangelist

danbold@apple.com

Related Sessions

Security and Your Apps	Mission	Tuesday 4:30PM
Introducing Watch Connectivity	Pacific Heights	Thursday 11:00AM
Your App and Next Generation Networks	Mission	Friday 11:00AM

Related Labs

Networking Lab	Frameworks Lab E	Thursday 10:00AM
Networking Lab	Frameworks Lab B	Friday 1:30PM

