App Transport Security

App Transport Security

- New in iOS 9 and OS X 10.11
- Mandates secure network connections
- Applies automatically

Enabling ATS

1. Link against iOS 9 or OS X 10.11
2. Use NSURLSession
3. Profit!

Why Bother with ATS?

- It will probably make Apple happy
- Mandates use of secure connections
  - ...most commonly HTTPS over HTTP

Why Bother with HTTPS?

- Make sure you know who the other person is
  - …and that you trust who they say they are
- Avoid having data sniffed in transmit
- Stop someone masquerading as you
- Keep old messages secret, even if your key is lost

Using ATS in Your App

Frameworks Using ATS

- NSURLSession and NSURLConnection
  - Anything built on these, such as Alamofire or AFNetworking
- CFURL APIs (according to the tech note)
- *Exempt*: other CFNetwork APIs

Satisfying ATS Requirements

- The best way: **support modern HTTPS!**
  - TLS 1.2
  - Ciphers providing forward secrecy
  - SHA256 signature hash
  - 2048-bit RSA key or 256-bit ECC key
- This will be almost entirely server-side

Temporarily Disabling ATS

- The key word: **temporarily**
- Add exceptions to ATS requirements
  - Either global or domain-specific
  - Either requirement-specific or wholesale

ATS Exceptions: Global

```
<!-- Try very hard not to do this -->

<key>NSAppTransportSecurity</key>
<dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
</dict>
```

ATS Exceptions: Domain-Specific

```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSExceptionDomains</key>
    <dict>
        <key>example.com</key>
        <dict>
            <!-- Turns off all ATS requirements for example.com -->
            <key>NSExceptionAllowsInsecureHTTPLoads</key>
            <true/>
        </dict>
    </dict>
</dict>
```

ATS Exceptions: Requirement-Specific

```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSExceptionDomains</key>
    <dict>
        <key>example.com</key>
        <dict>
            <!-- Weakens only TLS requirement for example.com -->
            <key>NSExceptionMinimumTLSVersion</key>
            <string>TLSv1.0</string>
        </dict>
    </dict>
</dict>
```

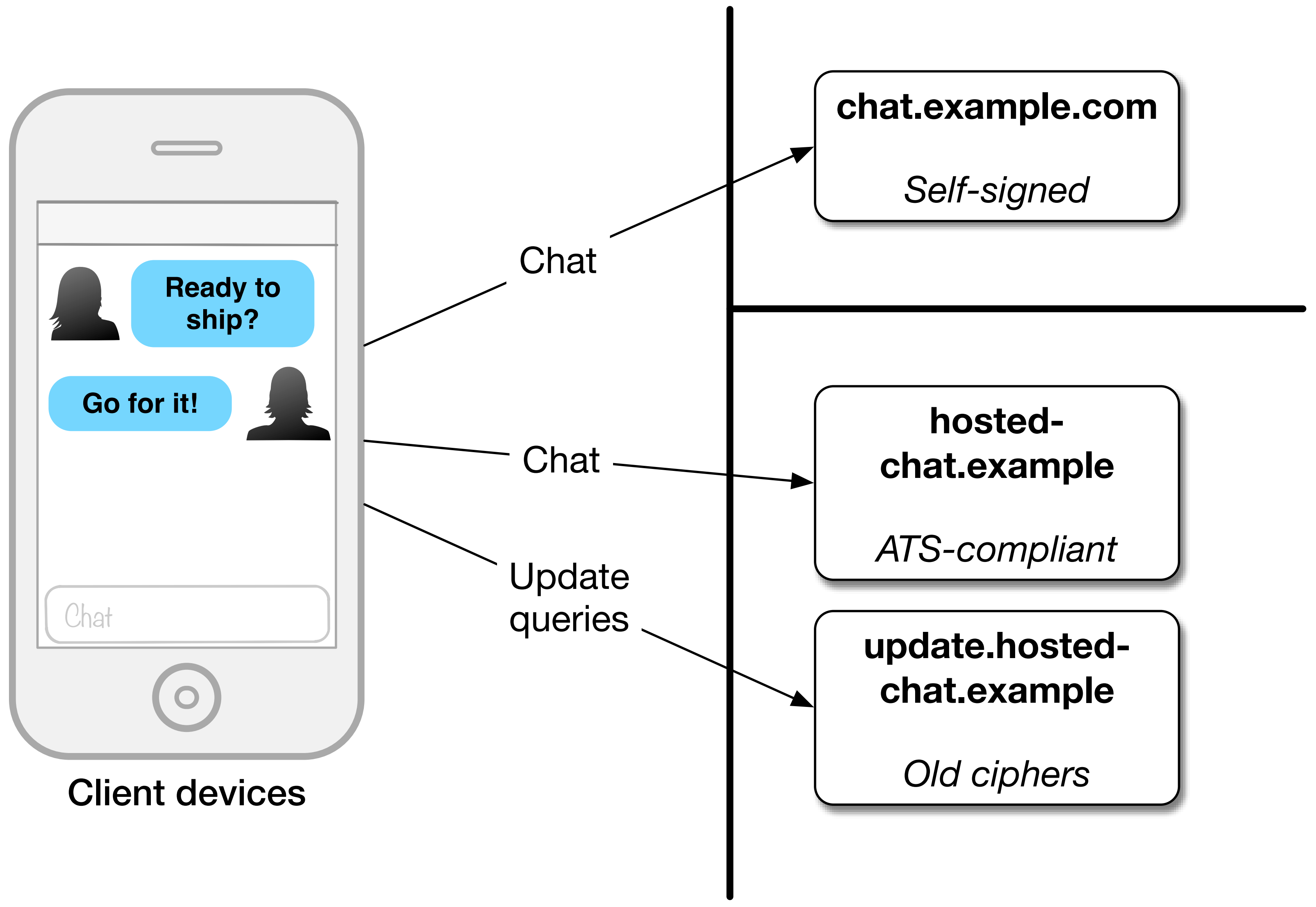
Subdomain Exceptions

- `NSIncludesSubdomains` key makes rules recursive
- Exceptions *without* subdomains given higher priority
  - Make an "exception to the exception"

When to Use an Exception

- Domain-specific for individual Web services
  - Have a plan for getting these upgraded
- Requirement-specific for older services
  - Might be semi-permanent, depending on needs
- Global only in rare cases
  - Connecting to customer-defined servers
  - Not because a dependency asked you to

Example: ATS and Chat



Client devices

Example: ATS and Chat

```
<key>NSAppTransportSecurity</key>
<dict>

    <!-- Let customers have bad security -->
    <key>NSAllowsArbitraryLoads</key>
    <true/>

    ...
```

Example: ATS and Chat

```
...
<key>NSExceptionDomains</key>
<dict>
    <!-- But make sure we're compliant everywhere -->
    <key>hosted-chat.example</key>
    <dict>

        <key>NSExceptionAllowsInsecureHTTPLoads</key>
        <false/>

        <key>NSIncludesSubdomains</key>
        <true/>

    </dict>
    ...
```

Example: ATS and Chat

```
        …

        <!-- Except this one server that we're upgrading -->
        <key>update.hosted-chat.example</key>
        <dict>

            <key>NSExceptionRequiresForwardSecrecy</key>
            <false/>

        </dict>

    </dict>
</dict>
```

ATS Tidbits

- Requirements apply at every step of a redirect
  - Each step must be secure, or have an exception
- Installed certificates require careful handling
  - Still need to manage cert in delegate methods
- Cannot add dynamic exceptions
  - Everything's specified up front in Info.plist

ATS-Exempt Technologies

- Playgrounds don't apply ATS restrictions
  - There's no Info.plist to control its behavior
- SFSafariViewController can load arbitrary content
  - Great for writing social media apps
  - Handy for help content, release notes, etc.

Debugging ATS Failures

CFNetwork Diagnostics

- Setting `CFNETWORK_DIAGNOSTICS` environment variable turns on extra logging
  - Do this in the app's scheme for easy toggling
  - Integer values range from 1 to 3
- On first request, app prints path to log file

CFNetwork Diagnostics

```
Oct  5 20:36:01  SafariViewControllerATSTest[99228]
  <Notice>: CFNetwork Diagnostics [1:1] 20:36:01.119 {
    LoaderWhatToDo
        Request: <CFURL 0x7fbbaa448810 [0x1061f37b0]>{string = http://example.com/,
                                                      encoding = 134217984,
                                                      base = (null)}
    CachePolicy: 0
       WhatToDo: originload
    CreateToNow: 0.00093s
    } [1:1]
Oct  5 20:36:01  SafariViewControllerATSTest[99228]
  <Notice>: CFNetwork Diagnostics [1:2] 20:36:01.122 {
    Response Error
    Request: <CFURLRequest 0x7fbbaa448950 [0x1061f37b0]> {url = http://example.com/,
                                                          cs = 0x0}
      Error: Error Domain=kCFErrorDomainCFNetwork Code=-1022 "(null)"
    } [1:2]
```

nscurl

- CFNetwork diagnostics are helpful, but cryptic
- In 10.11, OS X includes the `nscurl` utility
  - Like `curl`, but with Foundation networking?
- Flag `--ats-diagnostics` checks ATS requirements

nscurl

```
================================================================

Default ATS Secure Connection
---
ATS Default Connection
2015-10-05 20:43:09.410 nscurl[21060:613707]
    CFNetwork SSLHandshake failed (-9824)
2015-10-05 20:43:09.411 nscurl[21060:613707]
    NSURLSession/NSURLConnection HTTP load failed (kCFStreamErrorDomainSSL, -9824)
Result : FAIL
---

================================================================
```

nscurl

- Just like Info.plist, flags available for exceptions
  - `--verbose`: explain more about failures
  - `--ats-tls-version`: specify TLS requirement
  - `--ats-disable-pfs`: don't require Perfect Forward Secrecy (i.e. allow weaker TLS ciphers)

See Also

- The official App Transport Security Technote
- Early blog posts: Neglected Potential, Use Your Loaf
- WWDC 2015 session 706: Security and Your Apps
- @timothyekl

Thank You