Introduction

Prepared for CocoaHeads by
Jeff Meador
January 19, 2017

Overview

01 CocoaHeads

02 Jeff Meador

03 Vectorform

CocoaHeads

What it is. What it is not.


Map
Satellite
ASIA
ASIA
NORTH AMERICA
AFRICA
SOUTH AMERICA
OCEANIA
Indian Ocean
Pacific Ocean
Indian Ocean
Pacific Ocean
Google
Map data ©2017
Terms of Use
Report a map error

CocoaHeads is a group devoted to discussion of Apple Computer's Cocoa and CocoaTouch Frameworks for programming on MacOS X and iOS. During monthly meetings, members present on their projects and offer tutorials on various programming topics.

cocoaheads.org

What it is not

- Jeff Meador's CocoaHeads
- Vectorform's CocoaHeads

What it is

- Our CocoaHeads.
- A place to spread knowledge.
- A way to meet like-minded people.
- Everyone gets something out of it and everyone can contribute.

2.0

Jeff Meador

Your Host and Organizer

03

Vectorform

My Home & Our Host

Vectorform is a digital invention company.

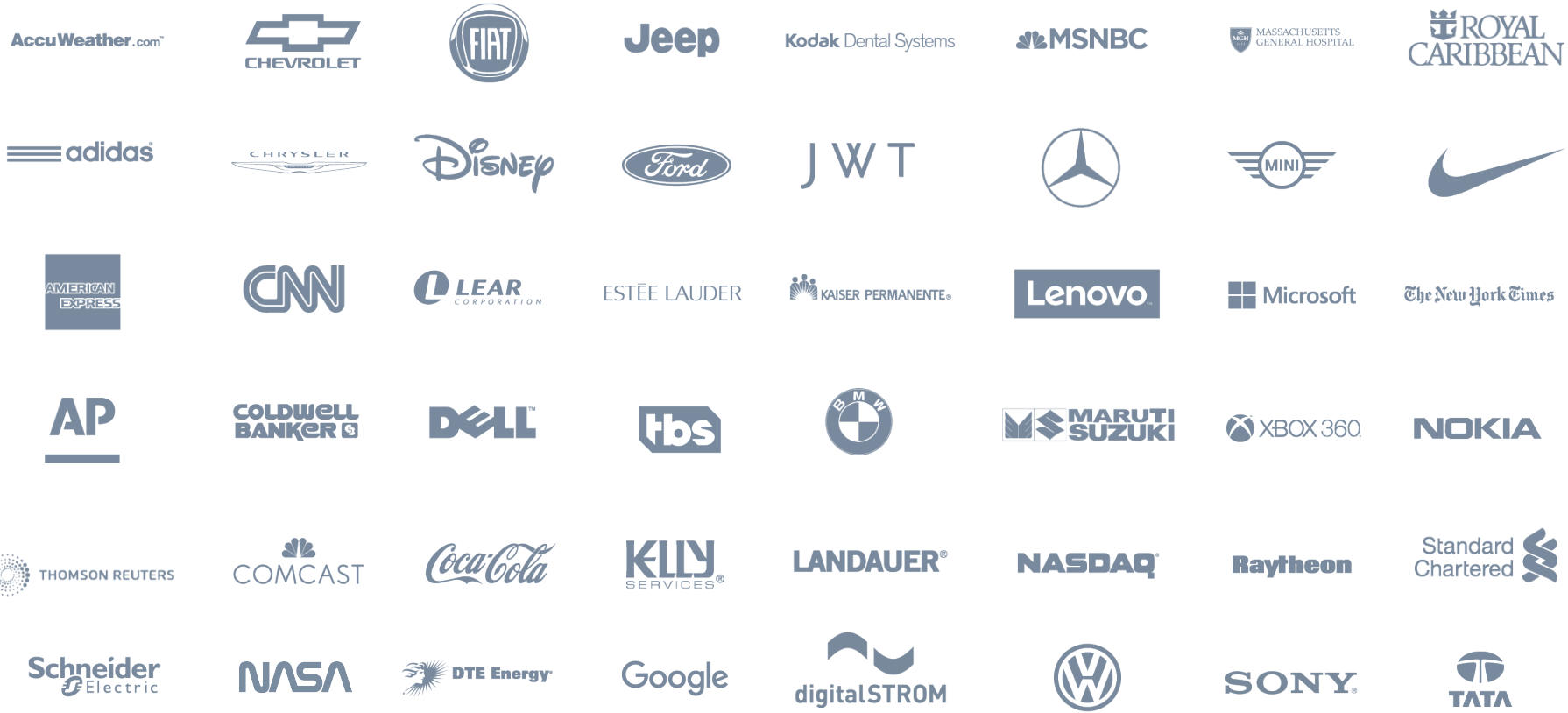
We invent digital products and experiences.
Our work helps companies define the future and solve complex problems.

30+
FORTUNE 500 CLIENTS

05
GLOBAL OFFICES

200+
PRODUCTS & EXPERIENCES

Clients


AccuWeather.com
CHEVROLET
FIAT
Jeep
Kodak Dental Systems
MSNBC
MASSACHUSETTS GENERAL HOSPITAL
ROYAL CARIBBEAN
adidas
CHRYSLER
Disney
Ford
JWT
MINI
AMERICAN EXPRESS
CNN
LEAR CORPORATION
ESTĒE LAUDER
KAISER PERMANENTE
Lenovo
Microsoft
The New York Times
AP
COLDWELL BANKER
DELL
fbs
MARUTI SUZUKI
XBOX 360
NOKIA
THOMSON REUTERS
COMCAST
Coca-Cola
KELLY SERVICES
LANDAUER
NASDAQ
Raytheon
Standard Chartered
Schneider Electric
NASA
DTE Energy
Google
digitalSTROM
SONY
TATA

VECTORFORM OVERVIEW

USA

1 Detroit

Vectorform llc
123 W Fifth st
Royal Oak, MI 48067

T+ 1248 777 7777 T+ 800 475 6279

USA

2 Seattle

Vectorform llc
2107 Elliot Ave, Suite 303
Seattle, WA 98121

T+ 1800 475 6279 F+ 1248 616 1980

USA

3 New York

Vectorform llc
190 N 10th Street, Suite 303
Brooklyn, NY 11211

T+1800 475 6279 F+ 1248 616 1980

GERMANY

4 Munich

Vectorform GmbH
Arnulfstrasse 37
80636 Munich

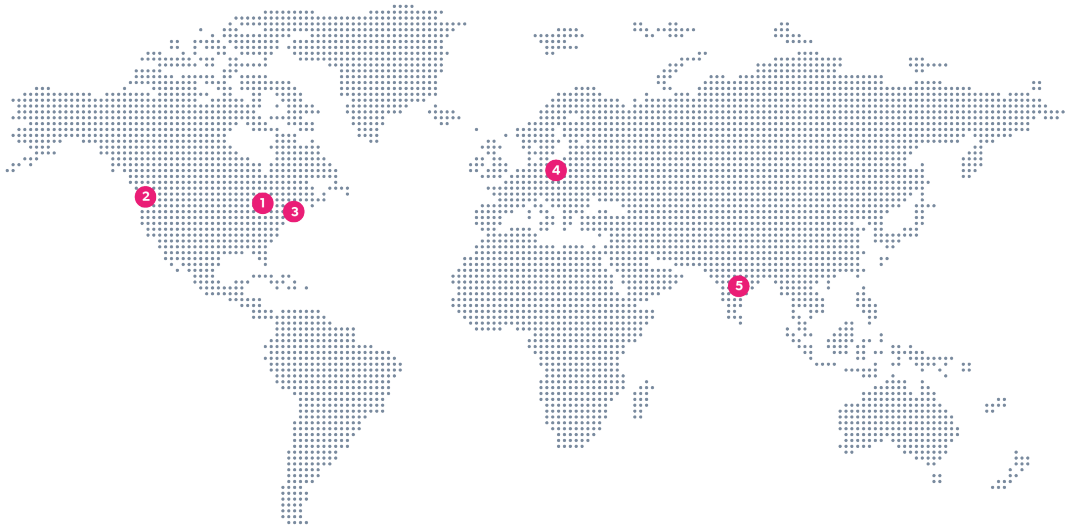
T+ 49 89 856 33 90T+ 49 89 856 33 99

INDIA

5 Hyderabad

Vectorform Software Pvt Ltd
301, 3rd Floor, Babukan Estate
Basheerbagh, Hyderabad - 500001
Andhra Pradesh

T+ 91 4031000007 F+ 91 4031000008



Vectorform's diverse portfolio spans mobile, gaming, desktop, online, in-vehicle, and next-generation platforms. Our award-winning expertise in creating cross-platform interactive experiences and our reputation as a tenacious, outcome-oriented partner is unmatched.

- Founded in 1999 in Detroit, MI
- Owned and operated by managing partners Jason Vazzano, Kurt Steckling, and Karl Steckling
- Global offices in Seattle, Detroit, New York, Munich, and Hyderabad
- 120+ employees
- Unique office atmosphere also includes dedicated hardware workshop, UX lab, video and photography studio

App Transport Security

Prepared for CocoaHeads by

January 19, 2017

APP TRANSPORT SECURITY

Overview

01 What is ATS?

02 The Deadline

03 Determining Compliance

04 Becoming Compliant

05 Other Keys

06 Bonus

What is ATS?

Is this a new API? Do I need to change my app? Do I need to change my server?

App Transport Security

Your app's network traffic needs to be encrypted.

Encryption needs to be done with modern algorithms.

If that's not possible, you need to tell Apple why during review.

This only applies to NSURLConnection and NSURLSession.

What is ATS?

REQUIREMENTS

With App Transport Security (ATS) fully enabled, the system requires that your app's HTTP connections use HTTPS and that they satisfy the following security requirements:

- The X.509 digital server certificate must meet at least one of the following trust requirements:
    - **Issued by a certificate authority (CA) whose root certificate is incorporated into the operating system**
    - **Issued by a trusted root CA and installed by the user or a system administrator**
- The negotiated Transport Layer Security (TLS) version must be TLS 1.2. Attempts to connect without TLS/SSL protection, or with an older version of TLS/SSL, are denied by default.
- The connection must use either the AES-128 or AES-256 symmetric cipher. The negotiated TLS connection cipher suite must support perfect forward secrecy (PFS) through Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) key exchange, and must be one of the following:
    - **TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384**
    - **TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256**
    - **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384**
    - **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA**
    - **TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256**
    - **TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA**
    - **TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384**
    - **TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256**
    - **TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384**
    - **TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256**
    - **TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA**
- The leaf server certificate must be signed with one of the following types of keys:
    - **Rivest-Shamir-Adleman (RSA) key with a length of at least 2048 bits**
    - **Elliptic-Curve Cryptography (ECC) key with a size of at least 256 bits**
- In addition, the leaf server certificate hashing algorithm must be Secure Hash Algorithm 2 (SHA-2) with a digest length, sometimes called a "fingerprint," of at least 256 (that is, SHA-256 or greater).

The requirements listed in this section are current as of this document's publication date, with stricter requirements possible in the future. Changes to these requirements will not break app binary compatibility.

What is ATS?

REQUIREMENT - HTTPS

REQUIREMENT

With App Transport Security (ATS) fully enabled, the system requires that your app's HTTP connections use HTTPS

DEFINITION

Hypertext Transport Protocol (also HTTP) — Really? You're looking this up in the glossary!?! Anyway, see RFC 2616. (Apple, Technical Note TN2232)

HTTPS – HTTP with encryption (using TLS) to protect user data.

What is ATS?

REQUIREMENT - TLS

REQUIREMENT

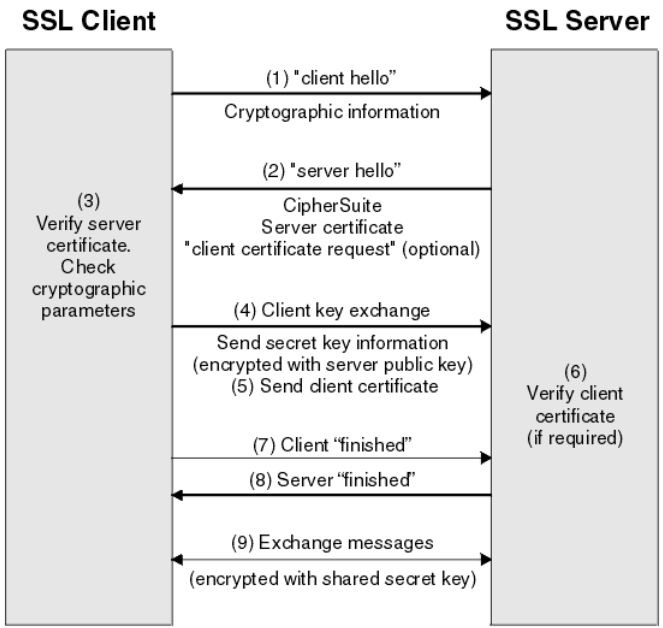
The negotiated Transport Layer Security (TLS) version must be TLS 1.2. Attempts to connect without TLS/SSL protection, or with an older version of TLS/SSL, are denied by default.

DEFINITION

TLS - Transport Layer Security is a protocol for encrypting network data. Anything less than 1.2 is no longer secure.

What is ATS?

REQUIREMENT - TLS



1. The SSL or TLS client sends a "client hello" message that lists cryptographic information such as the SSL or TLS version and, in the client's order of preference, the CipherSuites supported by the client. The message also contains a random byte string that is used in subsequent computations. The protocol allows for the "client hello" to include the data compression methods supported by the client.
2. The SSL or TLS server responds with a "server hello" message that contains the CipherSuite chosen by the server from the list provided by the client, the session ID, and another random byte string. The server also sends its digital certificate. If the server requires a digital certificate for client authentication, the server sends a "client certificate request" that includes a list of the types of certificates supported and the Distinguished Names of acceptable Certification Authorities (CAs).
3. The SSL or TLS client verifies the server's digital certificate. For more information, see How SSL and TLS provide identification, authentication, confidentiality, and integrity.
4. The SSL or TLS client sends the random byte string that enables both the client and the server to compute the secret key to be used for encrypting subsequent message data. The random byte string itself is encrypted with the server's public key.
5. If the SSL or TLS server sent a "client certificate request", the client sends a random byte string encrypted with the client's private key, together with the client's digital certificate, or a "no digital certificate alert". This alert is only a warning, but with some implementations the handshake fails if client authentication is mandatory.
6. The SSL or TLS server verifies the client's certificate. For more information, see How SSL and TLS provide identification, authentication, confidentiality, and integrity.
7. The SSL or TLS client sends the server a "finished" message, which is encrypted with the secret key, indicating that the client part of the handshake is complete.
8. The SSL or TLS server sends the client a "finished" message, which is encrypted with the secret key, indicating that the server part of the handshake is complete.
9. For the duration of the SSL or TLS session, the server and client can now exchange messages that are symmetrically encrypted with the shared secret key.

http://www.ibm.com/support/knowledgecenter/SSFKSJ_7.1.0/com.ibm.mq.doc/sy10660_.htm

What is ATS?

REQUIREMENT - CERTIFICATE TRUST

REQUIREMENT

The X.509 digital server certificate must meet at least one of the following trust requirements:

- Issued by a certificate authority (CA) whose root certificate is incorporated into the operating system
- Issued by a trusted root CA and installed by the user or a system administrator

DEFINITIONS

Digital Certificates – A data structure that ensures a server is who it says it is.

X.509 – The only type of digital certificate that's relevant to TLS.

Certificate Authority – Trusted source for issuing certificates.

The requirements above are trying to say ATS won't support servers with no certificate, or a self-signed, expired, or host-name-mismatched certificates.

What is ATS?

REQUIREMENT – CERTIFICATE SIGNING

REQUIREMENT

The leaf server certificate must be signed with one of the following types of keys:

- Rivest-Shamir-Adleman (RSA) key with a length of at least 2048 bits
- Elliptic-Curve Cryptography (ECC) key with a size of at least 256 bits

In addition, the leaf server certificate hashing algorithm must be Secure Hash Algorithm 2 (SHA-2) with a digest length, sometimes called a "fingerprint," of at least 256 (that is, SHA-256 or greater).

DESCRIPTION

Signature – Used to verify authenticity of Certificiate. Signature is verified using the key + signature algorithm.

RSA or ECC – These are different types of keys. Each one has it's own required key size for ATS.

SHA-2 – Hashing algorithm for generating certificate signature.

What is ATS?

REQUIREMENT - CIPHER SUITE

REQUIREMENT

The connection must use either the AES-128 or AES-256 symmetric cipher. The negotiated TLS connection cipher suite must support perfect forward secrecy (PFS) through Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) key exchange, and must be one of the following:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA

What is ATS?

REQUIREMENT - CIPHER SUITE

DESCRIPTION

Cipher Suite – An identifier for the collection of cryptographic algorithms that will be used to encrypt the data.
AES-128 or AES-256 – Advanced Encryption Standard with different key sizes (128 or 256).
PFS – Perfect Forward Secrecy is a protocol that ensures the keys used for encryption are only valid for the current session. Historical and future data cannot be decrypted with the key for this particular session.
ECDHE - Elliptic curve Diffie–Hellman is a key-agreement protocol that allows for perfect forward secrecy.

TLS_ECDHE_RSA_**WITH_AES_256**_GCM_SHA384

RSA or ECDSA – The authentication method.
GCM or CBC – The mode of operation that describes how we encrypt larges amounts of data using AES.
SHA, SHA384, or SHA256 – Hashing algorithm for data used for integrity checks.

The Deadline

Will ATS ever be enforced?

Does it matter?

"Beginning January 1, 2017 any updates or new apps submitted will need to be ATS compliant. More information on this requirements will be available soon. Once this information is available developers will be contacted directly."

Cedric
App Store Review – November 2016

The Deadline

Supporting App Transport Security

December 21, 2016

App Transport Security (ATS), introduced in iOS 9 and OS X v10.11, improves user security and privacy by requiring apps to use secure network connections over HTTPS. At WWDC 2016 we announced that apps submitted to the App Store will be required to support ATS at the end of the year. To give you additional time to prepare, this deadline has been extended and we will provide another update when a new deadline is confirmed. Learn more about ATS.

The Deadline

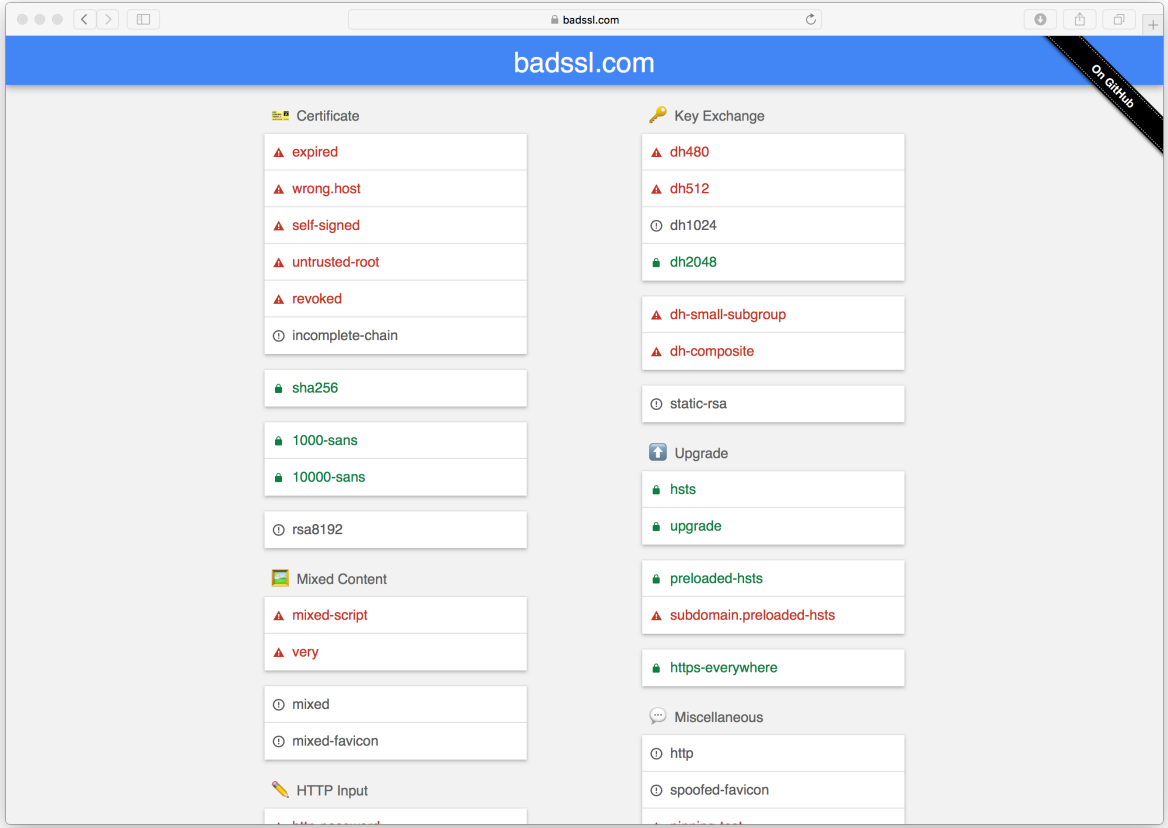
Does it matter?

The Deadline

- iOS 10
  - RC4 disabled by default
- iOS 11?
  - SHA-1
  - 3DES

The Deadline

BADSSL.COM



The Deadline

RC4 TEST

```
UILabel * label = [[UILabel alloc] initWithFrame:self.view.bounds];
[label setNumberOfLines:0];
[label setTextAlignment:NSTextAlignmentCenter];
[label setText:@"loading"];
[self.view addSubview:label];

[[[NSURLSession sharedSession] downloadTaskWithURL:[NSURL URLWithString:@"https://rc4.badssl.com"]
                                 completionHandler:^(NSURL * _Nullable location,
                                                     NSURLResponse * _Nullable response,
                                                     NSError * _Nullable error) {
                        dispatch_async(dispatch_get_main_queue(), ^{
                            [label setText:[error localizedDescription] ?: @"That seemed to work ok"];
                        });
}] resume];
```

The Deadline

RC4 TEST

	iOS 9	iOS 10
Xcode 7 (iOS 9 SDK)	PASS	FAIL
Xcode 8 (iOS 10 SDK)	PASS	FAIL

Determining Compliance

My networking code is broken. I'm guessing it's ATS.

Determining Compliance

ATS TEST

```
[[[NSURLSession sharedSession] downloadTaskWithURL:[NSURL URLWithString:@"https://3des.badssl.com"]
                                 completionHandler:^(NSURL * _Nullable location,
                                                     NSURLResponse * _Nullable response,
                                                     NSError * _Nullable error) {
    NSLog(@"%@",error);
}] resume];
```

Determining Compliance

ATS TEST

```
Error Domain=NSURLErrorDomain Code=-1200 "An SSL error has occurred and a secure connection to the
server cannot be made." UserInfo={_kCFStreamErrorCodeKey=-9824, NSLocalizedRecoverySuggestion=Would you
like to connect to the server anyway?, NSUnderlyingError=0x608000053b30 {Error
Domain=kCFErrorDomainCFNetwork Code=-1200 "(null)"
UserInfo={_kCFStreamPropertySSLClientCertificateState=0, _kCFNetworkCFStreamSSLErrorOriginalValue=-
9824, _kCFStreamErrorDomainKey=3, _kCFStreamErrorCodeKey=-9824}}, NSLocalizedDescription=An SSL error
has occurred and a secure connection to the server cannot be made.,
NSErrorFailingURLKey=https://3des.badssl.com/, NSErrorFailingURLStringKey=https://3des.badssl.com/,
_kCFStreamErrorDomainKey=3}
```

Determining Compliance

LOGGING

Xcode environment variable enables logging for networking code.

```
CFNETWORK_DIAGNOSTICS=1
```

Determining Compliance

NSCurl - Built into macOS

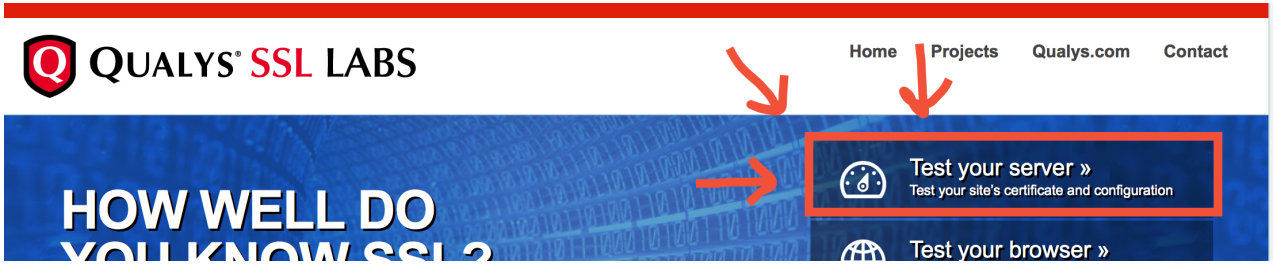
Determinig Compliance

TLSTool - https://developer.apple.com/library/content/samplecode/sc1236/Introduction/Intro.html

```
r-jmeador-mbp:Debug jmeador$ ./TLSTool s_client -connect 3des.badssl.com:443 -noverify
*  input stream did open
* output stream did open
* output stream has space
* protocol: TLS 1.2
* cipher: ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
* trust result: unspecified
* certificate info:
*   0 + rsaEncryption 2048 sha256-with-rsa-signature '*.badssl.com'
*   1 + rsaEncryption 2048 sha384-with-rsa-signature 'COMODO RSA Domain Validation Secure
Server CA'
*   2 + rsaEncryption 4096 sha384-with-rsa-signature 'COMODO RSA Certification Authority'
*  input stream has bytes
*  input stream end
* close
* bytes sent 0, bytes received 0
```

Determining Compliance

SSL Labs - www.sslabs.com



Safari 9 / OS X 10.11 R	RSA 2048 (SHA256)	TLS 1.2 > http/1.1	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA ECDH secp256r1 FS
Safari 10 / iOS 10 R	RSA 2048 (SHA256)	TLS 1.2 > http/1.1	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA ECDH secp256r1 FS
Safari 10 / OS X 10.12 R	RSA 2048 (SHA256)	TLS 1.2 > http/1.1	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA ECDH secp256r1 FS
Apple ATS 9 / iOS 9 R	Server sent fatal alert: handshake_failure		
Yahoo Slurp Jan 2015	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA ECDH secp256r1 FS
YandexBot Jan 2015	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA ECDH secp256r1 FS

(1) Clients that do not support Forward Secrecy (FS) are excluded when determining support for it.

Becoming Compliant

Hey ATS, let's be friends.

Becoming Compliant

REQUIREMENTS

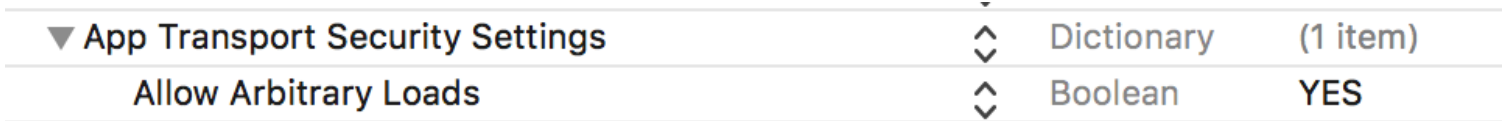
With App Transport Security (ATS) fully enabled, the system requires that your app's HTTP connections use HTTPS and that they satisfy the following security requirements:

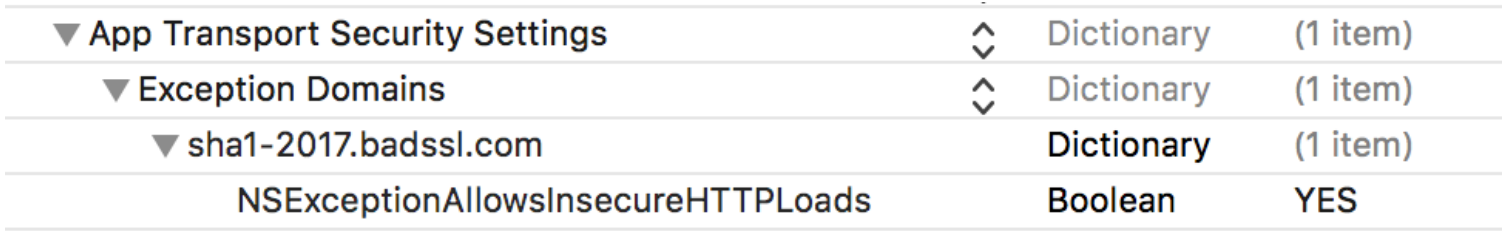
- The X.509 digital server certificate must meet at least one of the following trust requirements:
    - **Issued by a certificate authority (CA) whose root certificate is incorporated into the operating system**
    - **Issued by a trusted root CA and installed by the user or a system administrator**
- The negotiated Transport Layer Security (TLS) version must be TLS 1.2. Attempts to connect without TLS/SSL protection, or with an older version of TLS/SSL, are denied by default.
- The connection must use either the AES-128 or AES-256 symmetric cipher. The negotiated TLS connection cipher suite must support perfect forward secrecy (PFS) through Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) key exchange, and must be one of the following:
    - **TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384**
    - **TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256**
    - **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384**
    - **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA**
    - **TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256**
    - **TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA**
    - **TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384**
    - **TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256**
    - **TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384**
    - **TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256**
    - **TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA**
- The leaf server certificate must be signed with one of the following types of keys:
    - **Rivest-Shamir-Adleman (RSA) key with a length of at least 2048 bits**
    - **Elliptic-Curve Cryptography (ECC) key with a size of at least 256 bits**
- In addition, the leaf server certificate hashing algorithm must be Secure Hash Algorithm 2 (SHA-2) with a digest length, sometimes called a "fingerprint," of at least 256 (that is, SHA-256 or greater).

The requirements listed in this section are current as of this document's publication date, with stricter requirements possible in the future. Changes to these requirements will not break app binary compatibility.

Becoming Compliant

EXCEPTIONS





Becoming Compliant

REQUIREMENT - HTTPS

REQUIREMENT

With App Transport Security (ATS) fully enabled, the system requires that your app's HTTP connections use HTTPS

SYMPTOMS

Search for "http:" URLs in code

Your console will say: "App Transport Security has blocked a cleartext HTTP (http://) resource load since it is insecure. Temporary exceptions can be configured via your app's "

RESOLUTION

Add the following exception to your info.plist. This will trigger an App Store review, so be prepared.

This example is for http://http.badssl.com

```
<key>http.badssl.com</key>
<dict>
        <key>NSExceptionAllowsInsecureHTTPLoads</key>
         <true/>
</dict>
```

Becoming Compliant

REQUIREMENT - TLS

REQUIREMENT

The negotiated Transport Layer Security (TLS) version must be TLS 1.2. Attempts to connect without TLS/SSL protection, or with an older version of TLS/SSL, are denied by default.

SYMPTOMS

Console shows error kCFStreamErrorDomainSSL, -9801 (errSSLNegotiation)

TLSTool shows protocol TLS 1.1 or 1.0. If it shows SSL, this won't work.

RESOLUTION

Add the following exception to your info.plist. This will trigger an App Store review, so be prepared.

This example is for https://tls-v1-1.badssl.com:1011

```
<key>tls-v1-1.badssl.com</key>
<dict>
        <key>NSExceptionMinimumTLSVersion</key>
        <string>TLSv1.1</string>
</dict>
```

Becoming Compliant

REQUIREMENT – CERTIFICATE TRUST

REQUIREMENT

The X.509 digital server certificate must meet at least one of the following trust requirements:

- Issued by a certificate authority (CA) whose root certificate is incorporated into the operating system
- Issued by a trusted root CA and installed by the user or a system administrator

SYMPTOMS

Console shows error kCFStreamErrorDomainSSL, -9802 (errSSLFatalAlert)

Console: BadSSL Untrusted Root Certificate Authority or An SSL error has occurred and a secure connection to the server cannot be made.

TLSTool shows “trust result: recoverable trust failure”

Becoming Compliant

REQUIREMENT - CERTIFICATE

RESOLUTION

Add the following exception to your info.plist. This will trigger an App Store review, so be prepared. This is a special case for key NSExceptionAllowsInsecureHTTPLoads that allows you to verify a cert yourself.

This example is for http://self-signed.badssl.com

```
<key>self-signed.badssl.com</key>
<dict>
        <key>NSExceptionAllowsInsecureHTTPLoads</key>
         <true/>
</dict>
```

Becoming Compliant

REQUIREMENT - CERTIFICATE

```
- (void)connect{

    NSURLSession * session = [NSURLSession sessionWithConfiguration:[NSURLSessionConfiguration defaultSessionConfiguration]
                              delegate:self
                         delegateQueue:nil];

    [[session downloadTaskWithURL:[NSURL URLWithString:@"https://self-signed.badssl.com"]
            completionHandler:^(NSURL * _Nullable location,
                  NSURLResponse * _Nullable response,
                  NSError * _Nullable error){
              NSLog(@"%@",error);
            }] resume];
}

- (void)URLSession:(NSURLSession *)session
didReceiveChallenge:(NSURLAuthenticationChallenge *)challenge
 completionHandler:(void (^)(NSURLSessionAuthChallengeDisposition, NSURLCredential *))completionHandler{

        if([challenge.protectionSpace.host isEqualToString:@"self-signed.badssl.com"]){
            NSURLCredential *credential = [NSURLCredential credentialForTrust:challenge.protectionSpace.serverTrust];
            completionHandler(NSURLSessionAuthChallengeUseCredential,credential);
    }
}
```

Becoming Compliant

REQUIREMENT - CERTIFICATE SIGNING

REQUIREMENT

The leaf server certificate must be signed with one of the following types of keys:

- Rivest-Shamir-Adleman (RSA) key with a length of at least 2048 bits
- Elliptic-Curve Cryptography (ECC) key with a size of at least 256 bits

In addition, the leaf server certificate hashing algorithm must be Secure Hash Algorithm 2 (SHA-2) with a digest length, sometimes called a "fingerprint," of at least 256 (that is, SHA-256 or greater).

SYMPTOMS

Console shows error kCFStreamErrorDomainSSL, -9802 (errSSLFatalAlert)

TLSTool shows certificate info that doesn't match the requirements

Bad Hashing Example

* trust result: recoverable trust failure
* certificate info:
    * 0 + rsaEncryption 2048 sha1-with-rsa-signature

Bad Key Example

* trust result: recoverable trust failure
* certificate info:
* 0 + rsaEncryption 1024 sha256-with-rsa-signature

Becoming Compliant

REQUIREMENT - CERTIFICATE SIGNING

RESOLUTION

Add the following exception to your info.plist. This will trigger an App Store review, so be prepared. This is a special case for key NSExceptionAllowsInsecureHTTPLoads that allows you to verify a cert yourself.

This example is for https://sha1-2017.badssl.com

```
<key>sha1-2017.badssl.com</key>
<dict>
        <key>NSExceptionAllowsInsecureHTTPLoads</key>
         <true/>
</dict>
```

Becoming Compliant

REQUIREMENT - CIPHER SUITE

REQUIREMENT

The connection must use either the AES-128 or AES-256 symmetric cipher. The negotiated TLS connection cipher suite must support perfect forward secrecy (PFS) through Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) key exchange, and must be one of the following:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA

Becoming Compliant

REQUIREMENT - CIPHER SUITE

PERFECT FORWARD SECRECY SYMPTOMS

Console shows error kCFStreamErrorDomainSSL, -9824 (errSSLPeerHandshakeFail)

TLSTool shows cipher suite that doesn't match our list, more importantly the key exchange portion doesn't match.

Example:

- cipher: TLS_**RSA**_WITH_AES_256_GCM_SHA384

RESOLUTION

Is the cipher in this list?

- TLS_RSA_WITH_AES_256_GCM_SHA384
- TLS_RSA_WITH_AES_128_GCM_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_128_CBC_SHA

Becoming Compliant

REQUIREMENT - CIPHER SUITE

RESOLUTION

Add the following exception to your info.plist. This will **NOT** trigger an App Store review.

This example is for http://www.example.com

```
<key>www.example.com</key>
<dict>
        <key>NSExceptionRequiresForwardSecrecy</key>
         <false/>
</dict>
```

Becoming Compliant

REQUIREMENT - CIPHER SUITE

ENCRYPTION/HASHING SYMPTOMS

Console shows error kCFStreamErrorDomainSSL, -9824 (errSSLPeerHandshakeFail)

TLSTool shows cipher suite that doesn't match our list, more importantly the encryption/encoding portion doesn't match.

Example:

- cipher: ECDHE_RSA_WITH_**3DES_EDE_CBC_SHA**
- cipher: ECDHE_RSA_WITH_**RC4_128_SHA**

RESOLUTION

If you want to use ATS, revert this endpoint to HTTP and add the following exception to your info.plist. This will trigger an App Store review, so be prepared.

This example is for https://3des.badssl.com

```
<key>3des.badssl.com</key>
<dict>
        <key>NSExceptionAllowsInsecureHTTPLoads</key>
         <true/>
</dict>
```

Other Keys

I know there's more than that

Other Keys

SUB DOMAINS

NSIncludesSubdomains (true/false)

Lets you put **badssl.com** in your info.plist instead of **3des.badssl.com** and **rc4.badssl.com**.

Other Keys

MEDIA

NSAllowsArbitraryLoadsForMedia (true/false)

Added in iOS 10. If you need this in 9 you need to completely disable ATS.

Lets you disable ATS for AVFoundation.

This triggers an app store review.

Other Keys

WEB CONTENT

NSAllowsArbitraryLoadsInWebContent (true/false)

Added in iOS 10. If you need this in 9 you need to completely disable ATS.

Lets you disable ATS for WKWebView and UIWebView.

This triggers an app store review.

SFSafariViewController is not affected by your app's ATS settings.

Other Keys

LOCAL NETWORKING

NSAllowsLocalNetworking (true/false)

Added in iOS 10. If you need this in 9 you need to completely disable ATS.

Lets you perform local networking without completely disabling ATS.

This triggers an app store review.

Other Keys

CERTIFICATE TRANSPARENCY

NSRequiresCertificateTransparency (true/false)

Added in iOS 10. If you need this in 9 you need to completely disable ATS.

Certificate Transparency is not an ATS requirement. This is just here if you want to enable it. Requires valid, signed Certificate Transparency timestamps for server certificates for the named domain

This avoids communication with a Certificate Authority. That means increased speed, reliability, and the CA can't see what you're up to.

6.0

Creating Tools

Too much work?

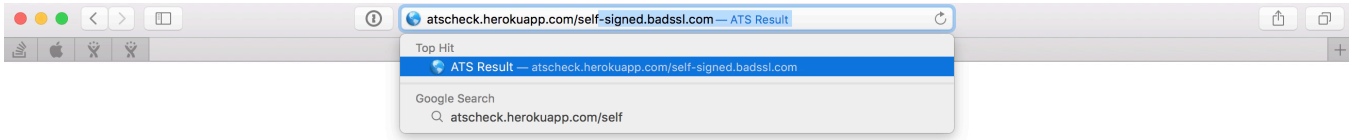
Creating Tools

ATS Check

https://atscheck.herokuapps.com/www.domain.com

Creating Tools

ATS Check



Usage

https://atscheck.herokuapps.com/www.domain.com

Built by Vectorform using Vapor and SSL Labs

Thank You

Vectorform  Invent with us.