

HTTPS REMOVAL TECHNIQUES FOR NON-BROWSER APPLICATIONS

Jacob Thompson

October 6, 2016



About ISE

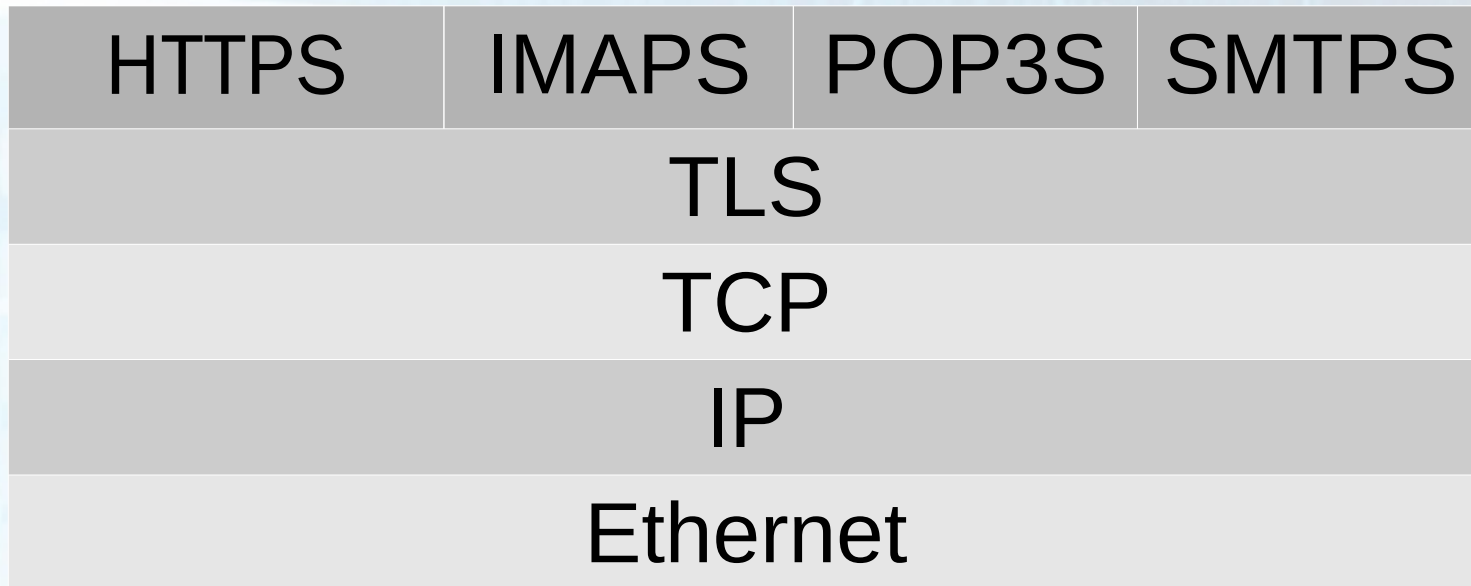
- We are:
 - Computer Scientists
 - Academics
 - Ethical Hackers
- Our customers are:
 - Fortune 500 enterprises
 - Entertainment, software security, healthcare
- Our perspective is:
 - White box

HTTPS vs. TLS

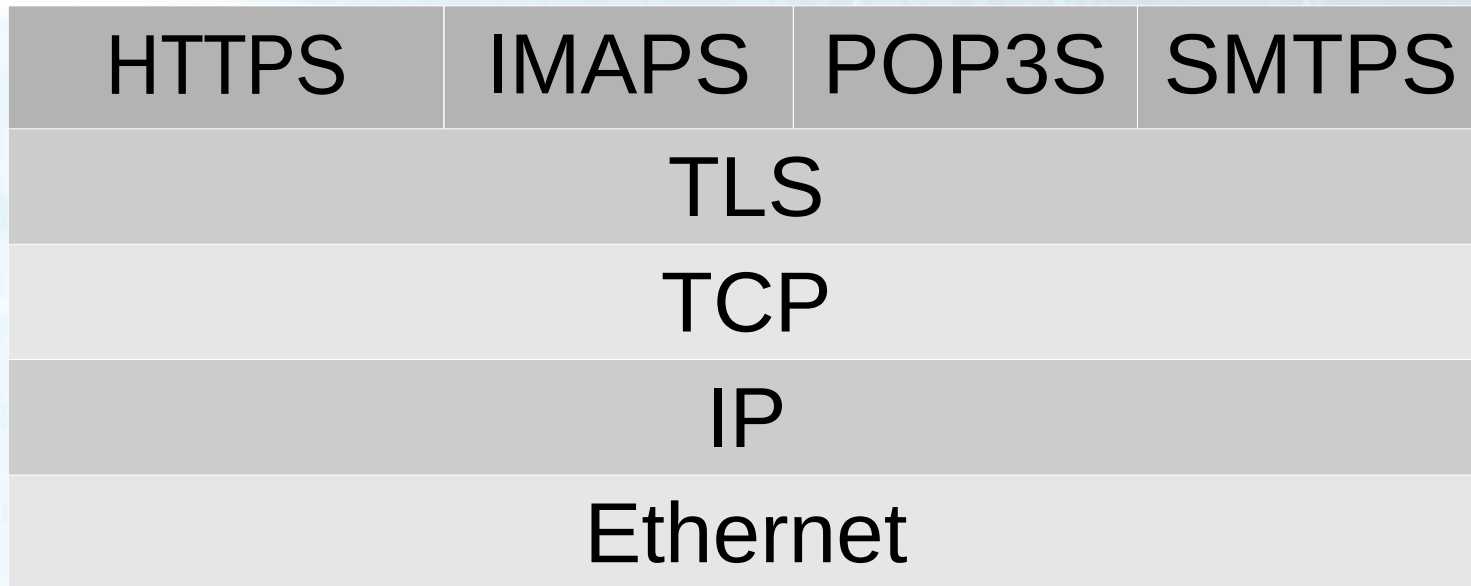
HTTPS/TLS in the Networking Stack

HTTPS	IMAPS	POP3S	SMTPS
TLS			
TCP			
IP			
Ethernet			

HTTPS/TLS in the Networking Stack



HTTPS/TLS in the Networking Stack



HTTPS/TLS in the Networking Stack

HTTPS	IMAPS	POP3S	SMTPS
TLS			
TCP			
IP			
Ethernet			

HTTPS/TLS in the Networking Stack

HTTPS	IMAPS	POP3S	SMTPS
TLS			
TCP			
IP			
Ethernet			

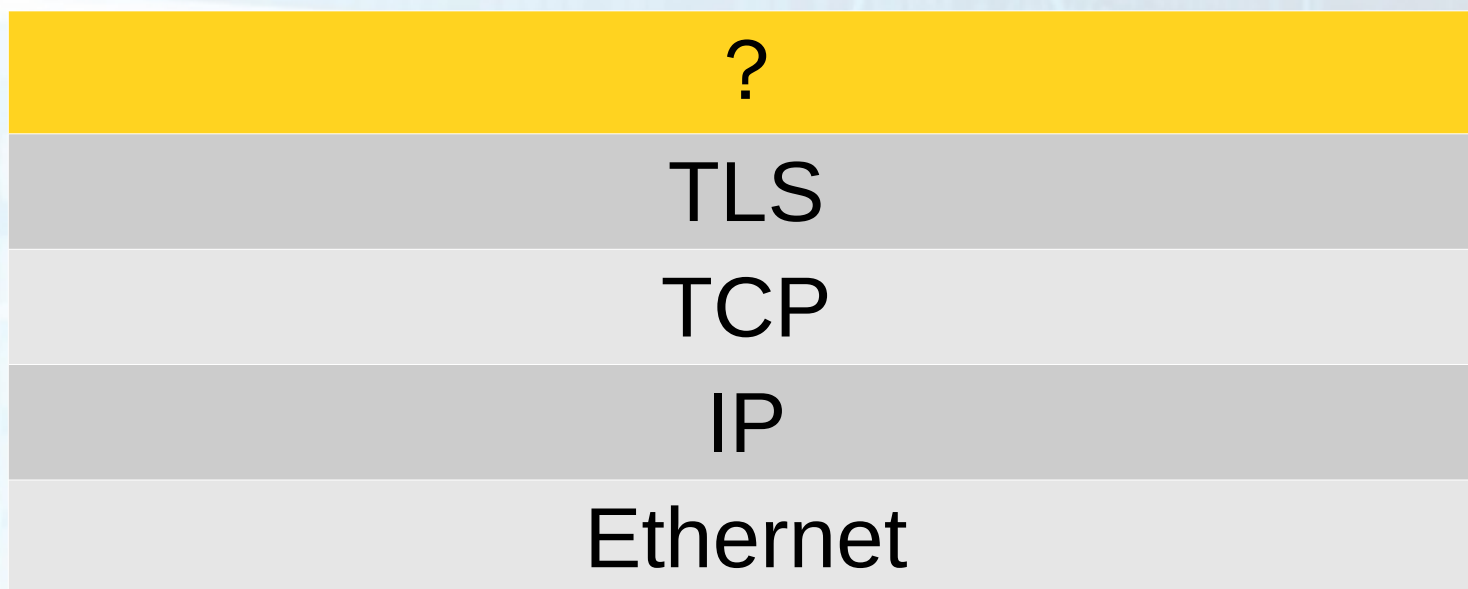
HTTPS/TLS in the Networking Stack

HTTPS	IMAPS	POP3S	SMTPTS
TLS			
TCP			
IP			
Ethernet			

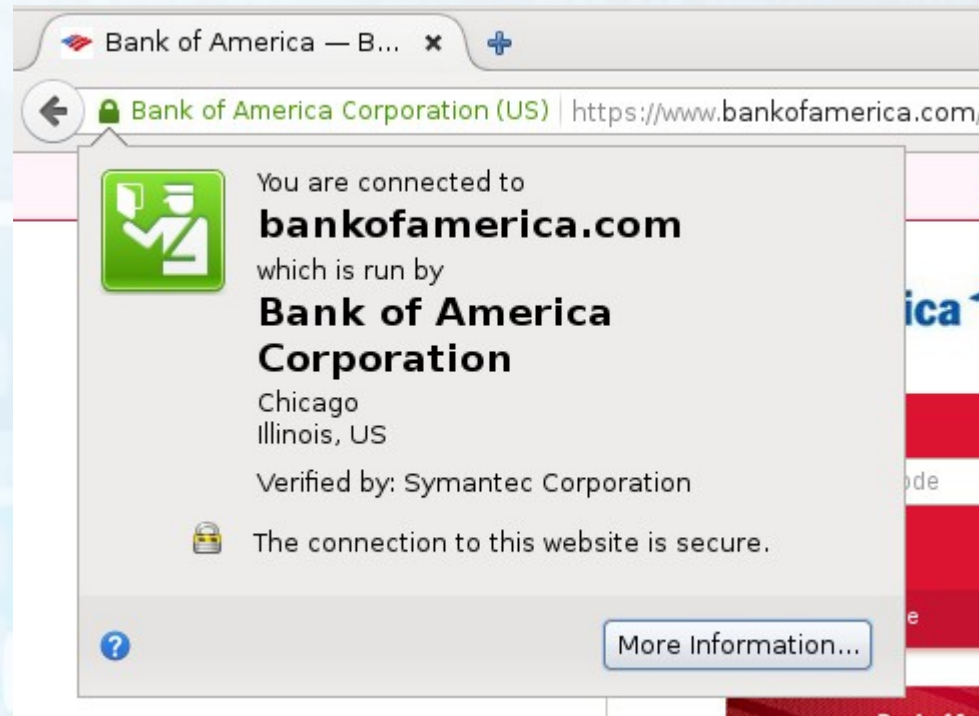
HTTPS/TLS in the Networking Stack

HTTPS	IMAPS	POP3S	SMTPS
TLS			
TCP			
IP			
Ethernet			

HTTPS/TLS in the Networking Stack



HTTPS



HTTPS = HTTP over TLS
CA/Browser Forum vs. IETF

Fundamentals of TLS

Fundamentals of TLS

1. Message Confidentiality

Fundamentals of TLS

1. Message Confidentiality
2. Message Integrity

Fundamentals of TLS

1. Message Confidentiality
2. Message Integrity
3. Server Authentication

Fundamentals of TLS

1. Message Confidentiality
2. Message Integrity
3. Server Authentication

*Preview: Defeat server authentication
and we defeat TLS*

TLS and Security Testing

TLS and Security Testing



TLS and Security Testing

 **BURPSUITE**
FREE EDITION



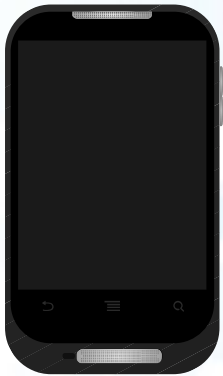
TLS and Security Testing



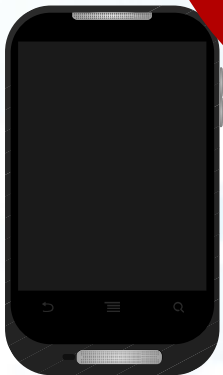
TLS and Security Testing



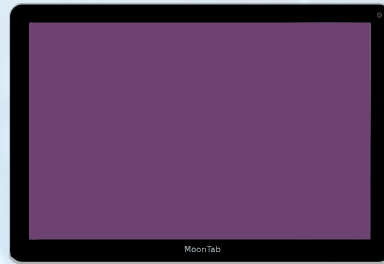
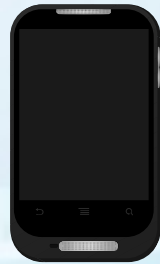
TLS and Security Testing



TLS and Security Testing



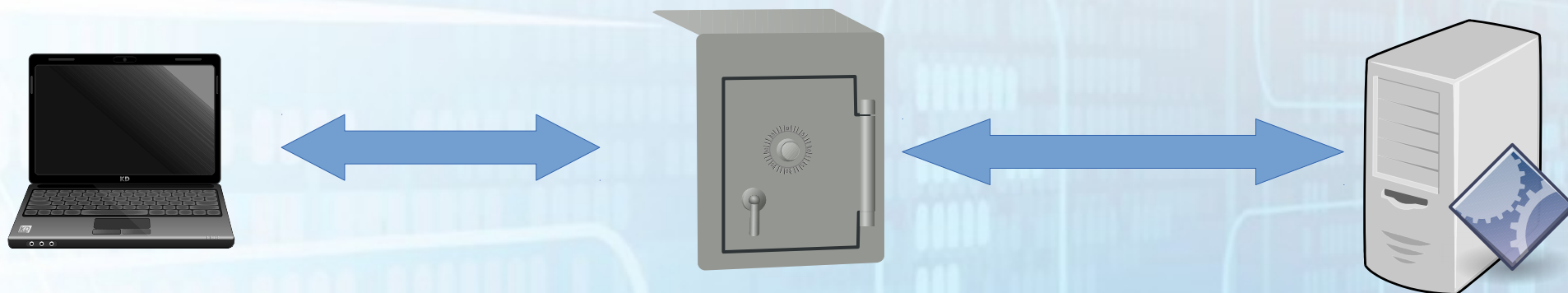
Bypassing TLS



Defeating certificate verification
is the focus of the remainder of
the talk

How TLS Works

How TLS Works



Asymmetric Cryptography



Asymmetric Cryptography



Asymmetric Cryptography



Asymmetric Cryptography



Certificates

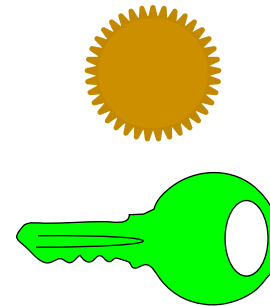
Certificate

**I hereby certify that the public key attached
hereto belongs to:**

www.example.com


Certificate Authority

Valid from:
Jan. 1, 2016
To:
Jan. 1, 2017



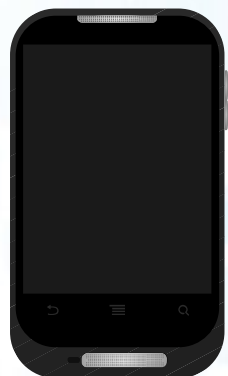
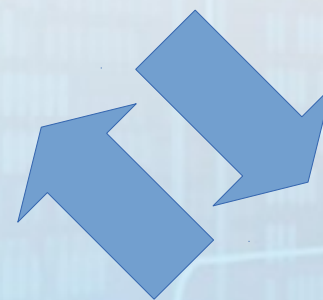
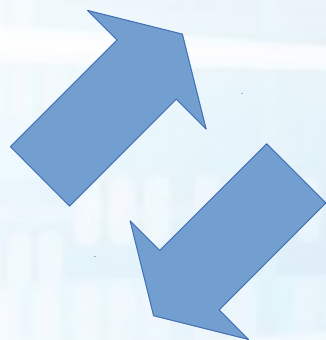
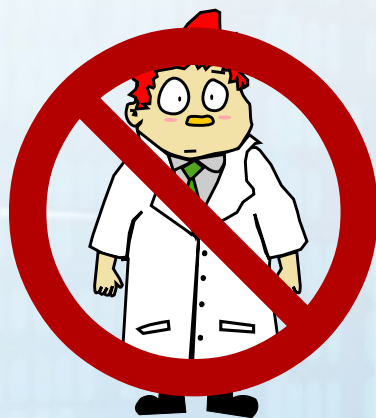
Certificate Chain



TLS and Real World Attack



TLS and Security Research

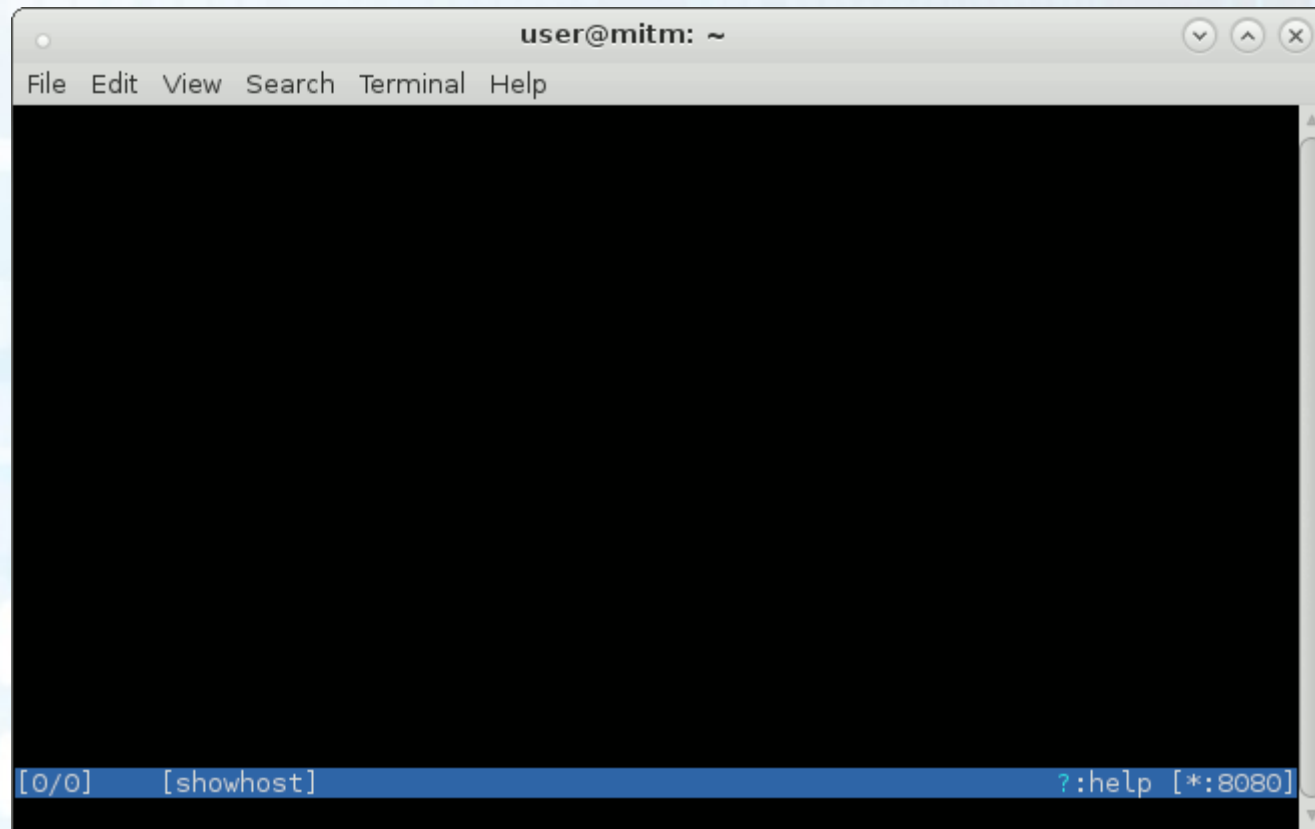


Custom Certificate Authority



Mitmproxy

Mitmproxy



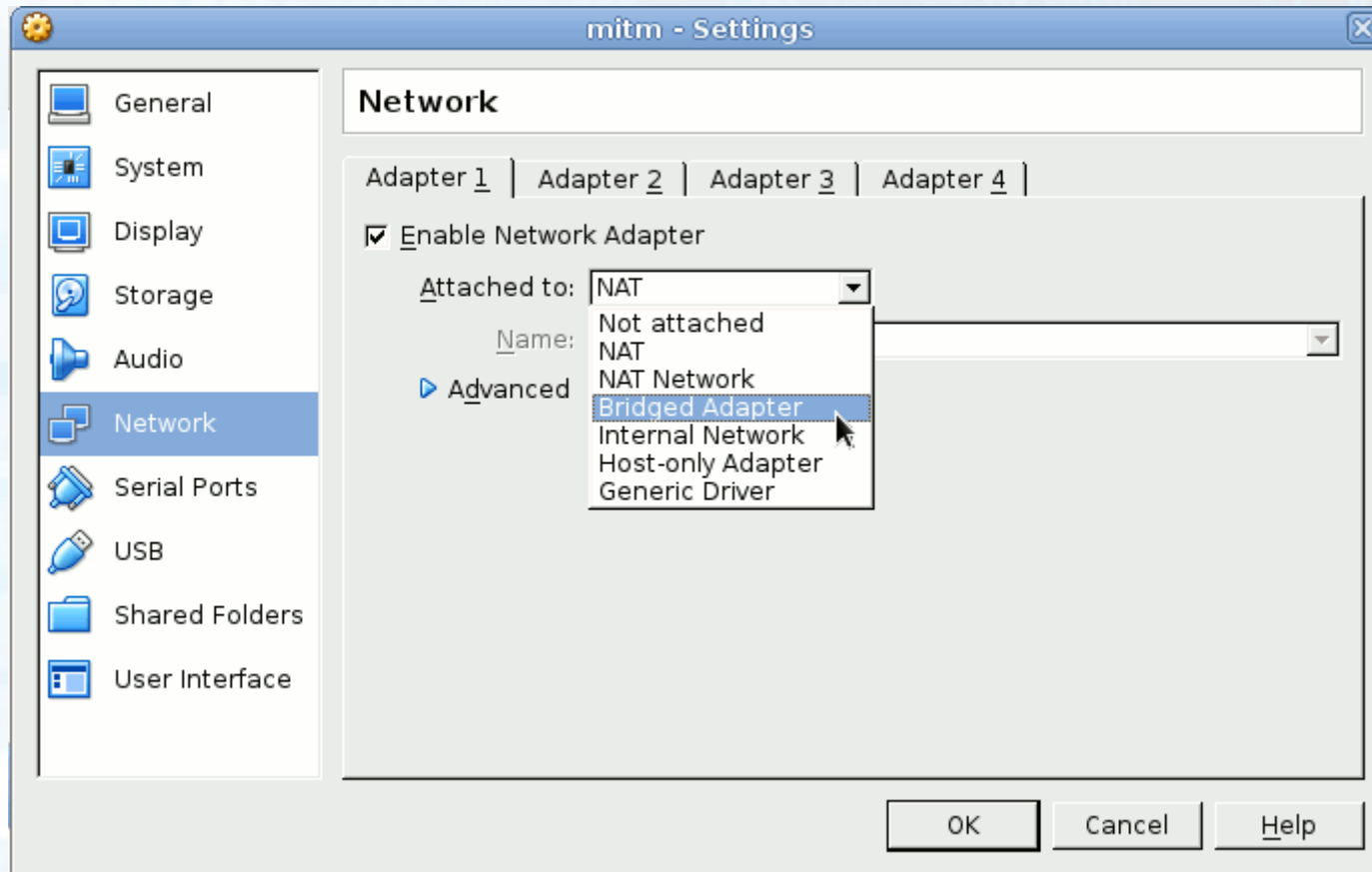
Virtual Machine Setup

VirtualBox



debian

Virtual Machine Bridged Mode



Virtual Machine Firewall

```
SUBNET=10.42.16.0/24
```

```
for i in /proc/sys/net/ipv4/conf/*/send_redirects; do echo 0 > $i; done
```

```
iptables -A INPUT -i lo -j ACCEPT
```

```
iptables -A INPUT -p icmp -j ACCEPT
```

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
iptables -A INPUT -s $SUBNET -j ACCEPT
```

```
iptables -A INPUT -d $SUBNET -j ACCEPT
```

```
iptables -A INPUT -p tcp -j REJECT --reject-with tcp-reset
```

```
iptables -P INPUT DROP
```

```
iptables -A FORWARD -s $SUBNET -j ACCEPT
```

```
iptables -A FORWARD -d $SUBNET -j ACCEPT
```

```
iptables -P FORWARD DROP
```

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080
```

```
iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-ports 8080
```

```
iptables -t nat -A POSTROUTING -j MASQUERADE
```

```
sysctl -w net.ipv4.ip_forward=1
```

Virtual Machine Firewall

```
SUBNET=10.42.16.0/24
```

```
for i in /proc/sys/net/ipv4/conf/*/send_redirects; do echo 0 > $i; done
```

```
iptables -A INPUT -i lo -j ACCEPT
```

```
iptables -A INPUT -p icmp -j ACCEPT
```

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
iptables -A INPUT -s $SUBNET -j ACCEPT
```

```
iptables -A INPUT -d $SUBNET -j ACCEPT
```

```
iptables -A INPUT -p tcp -j REJECT --reject-with tcp-reset
```

```
iptables -P INPUT DROP
```

```
iptables -A FORWARD -s $SUBNET -j ACCEPT
```

```
iptables -A FORWARD -d $SUBNET -j ACCEPT
```

```
iptables -P FORWARD DROP
```

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080
```

```
iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-ports 8080
```

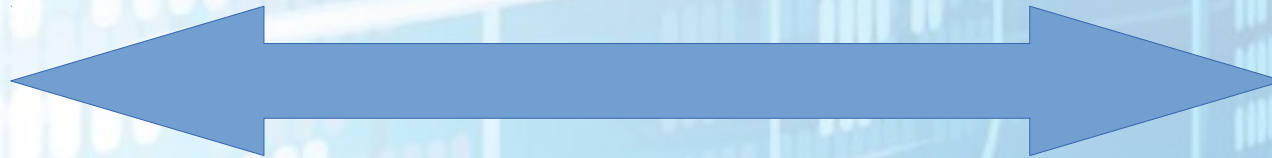
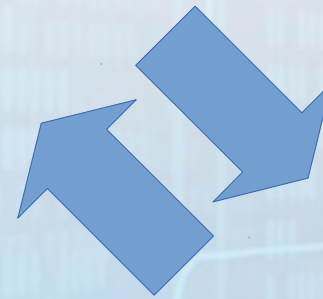
```
iptables -t nat -A POSTROUTING -j MASQUERADE
```

```
sysctl -w net.ipv4.ip_forward=1
```

iptables rules



80,443



Launching Mitmproxy

`mitmproxy --host` (HTTP proxy mode)

`mitmproxy -T --host` (Transparent mode)

Mitmproxy's ability to run as a transparent proxy is what allows us to study the traffic of non-proxy-enabled devices or software—so long as we can control certificate verification!

Configuring the Client Device

Configuring the Client Device

- Proxy-Enabled
 - Browsers
 - Most desktop applications
 - Most iOS applications
- Non-Proxy-Enabled
 - Most Android applications
 - Many standalone devices

General Technique

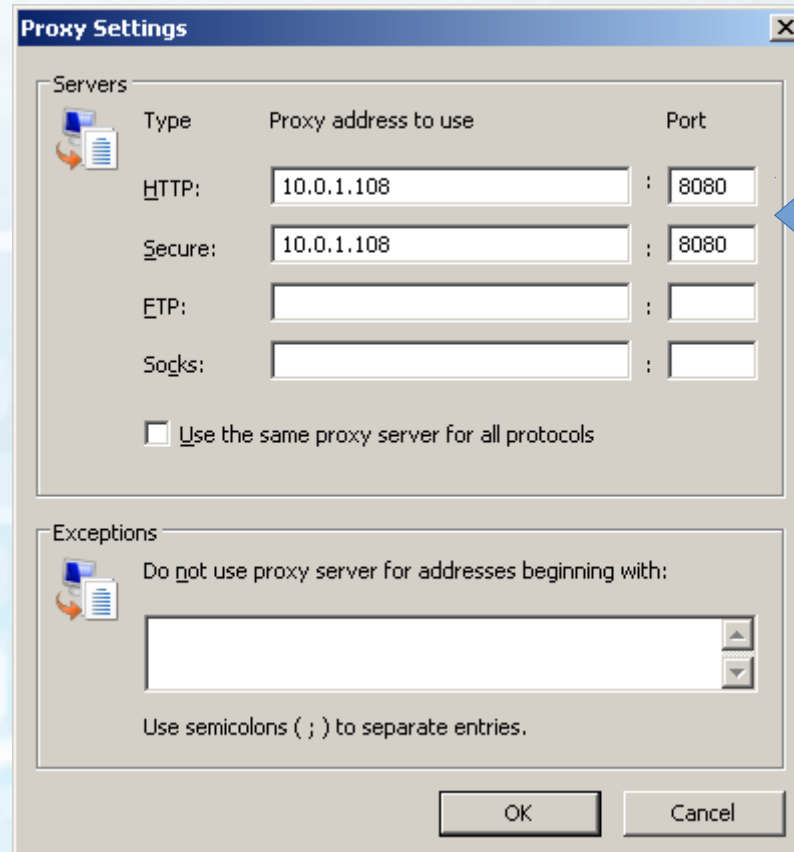
- Route traffic through Mitmproxy VM
 - Explicit proxy if available
 - Default gateway configuration if not
- Add Mitmproxy CA to trusted CAs
- View traffic

Windows Applications

Windows Applications

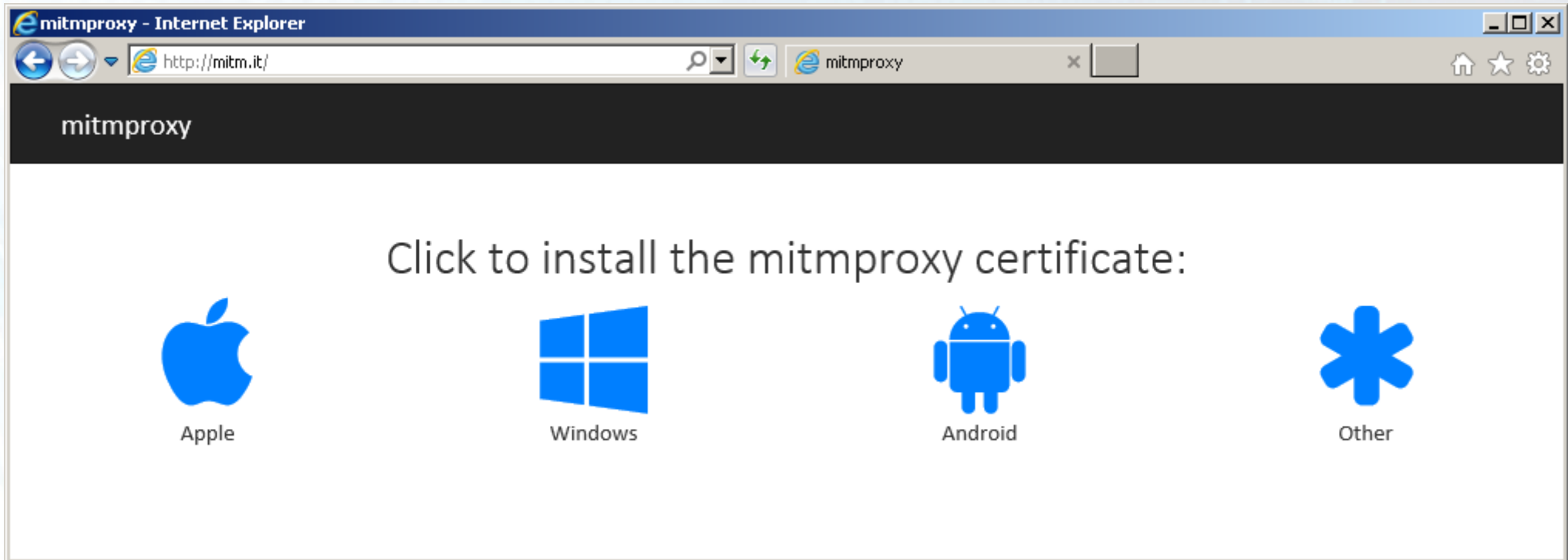


IE Proxy Configuration



Tools → Internet Options → Connections → LAN settings → Advanced
Verify that Mitmproxy is in HTTP proxy mode!

Mitmproxy CA Download Page

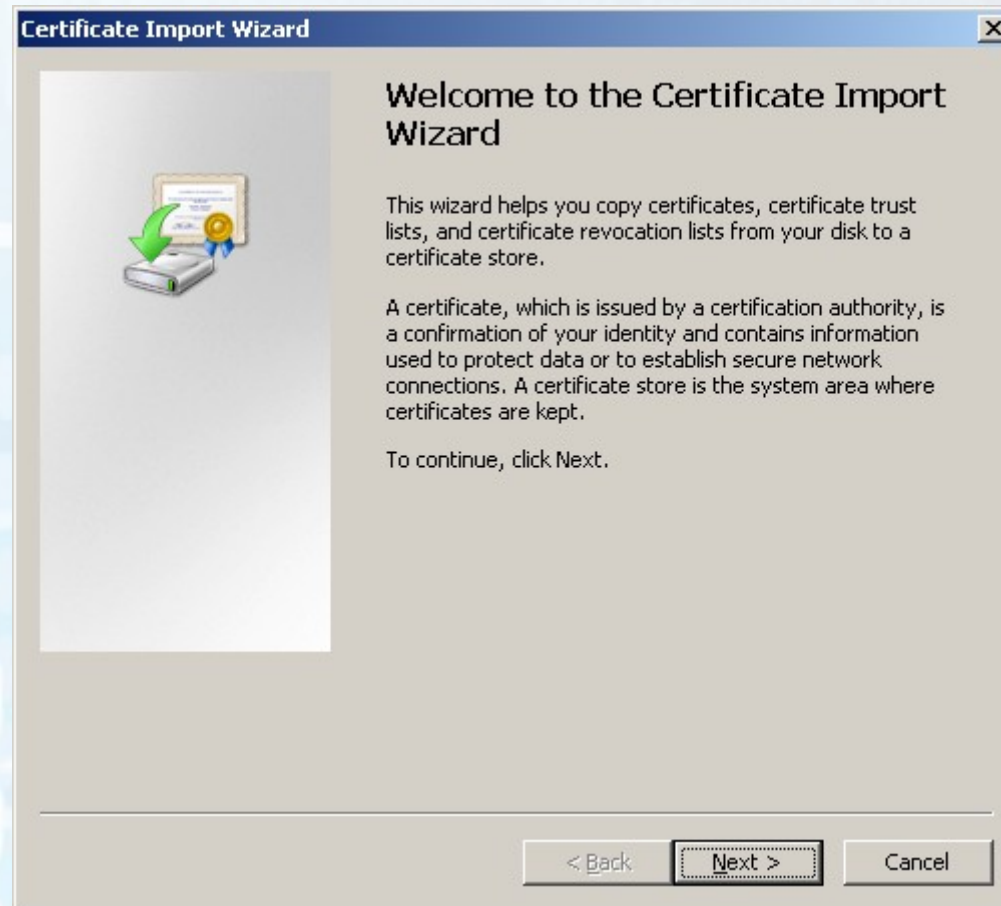


<http://mitm.it/>

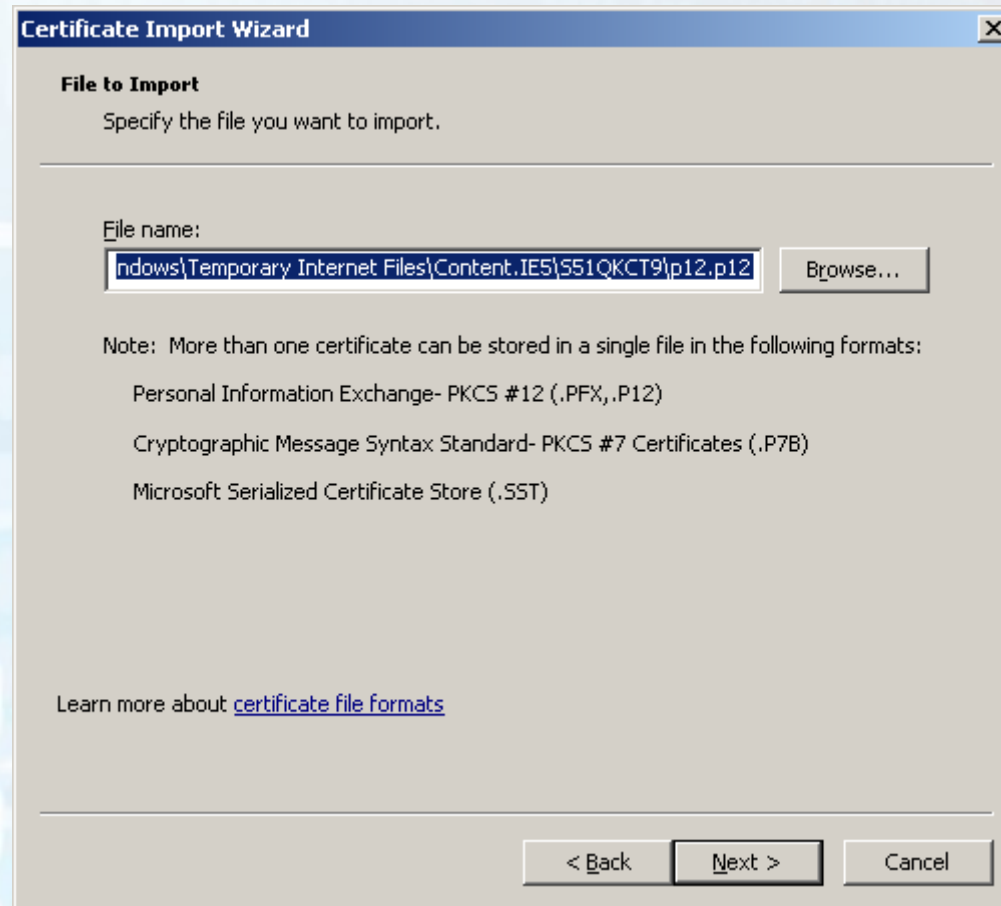
Mitmproxy CA Download

The screenshot shows an Internet Explorer browser window titled "mitmproxy - Internet Explorer". The address bar contains "http://mitm.it/". The page content includes the text "mitmproxy" and "Click to install the mitmproxy certificate:". Below this text are four blue icons: an Apple logo, a Windows logo, an Android robot, and a six-pointed star icon labeled "Other". At the bottom of the browser window, a yellow-bordered dialog box asks: "Do you want to open or save **p12.p12** (1.62 KB) from **mitm.it**?" with buttons for "Open", "Save", and "Cancel".

Importing the Mitmproxy CA



Importing the Mitmproxy CA



Importing the Mitmproxy CA

Certificate Import Wizard [X]

Password

To maintain security, the private key was protected with a password.

Type the password for the private key.

Password:

 Enable strong private key protection. You will be prompted every time the private key is used by an application if you enable this option.

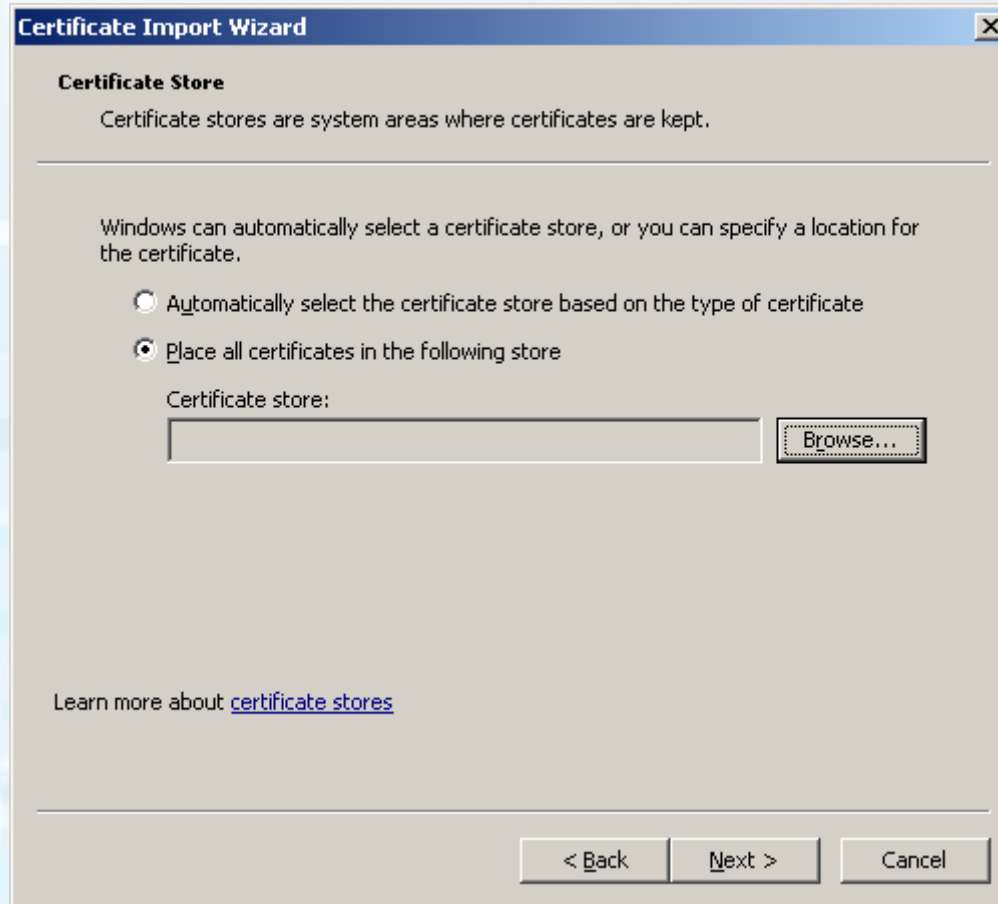
Mark this key as exportable. This will allow you to back up or transport your keys at a later time.

Include all extended properties.

Learn more about [protecting private keys](#)

< Back Next > Cancel

Importing the Mitmproxy CA



Certificate Import Wizard

Certificate Store
Certificate stores are system areas where certificates are kept.

Windows can automatically select a certificate store, or you can specify a location for the certificate.

Automatically select the certificate store based on the type of certificate

Place all certificates in the following store

Certificate store:

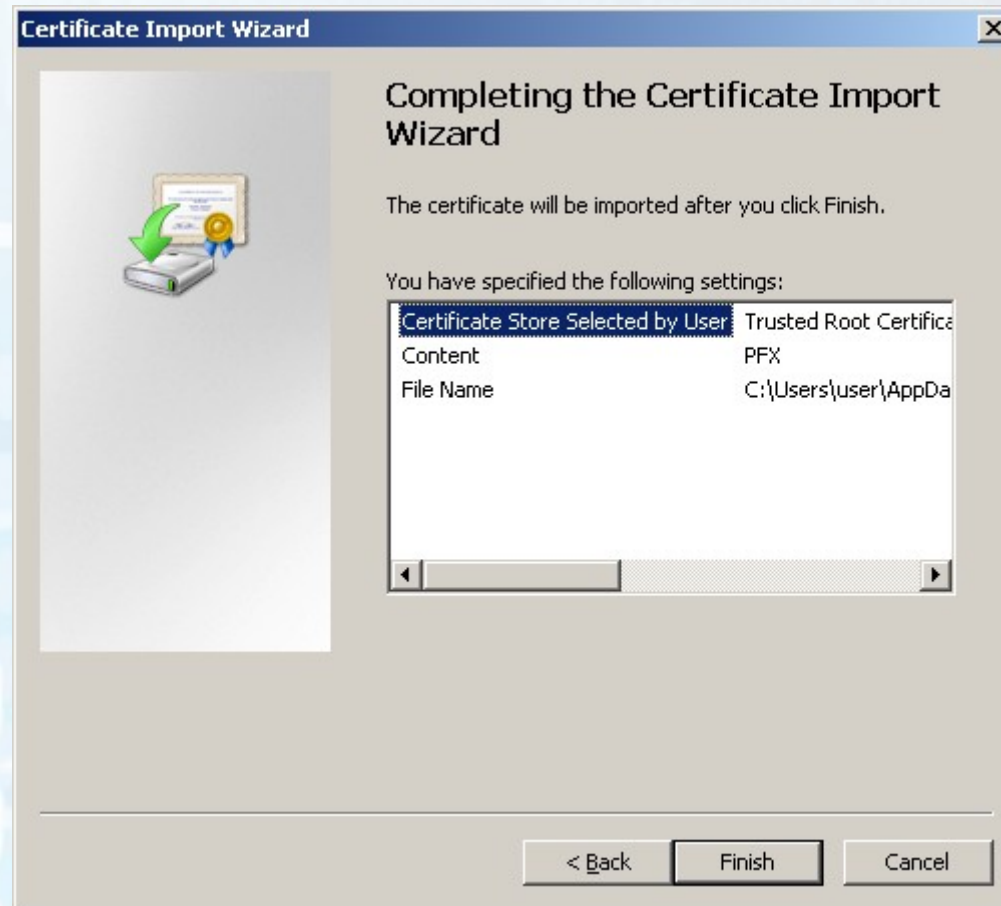
Learn more about [certificate stores](#)

< Back Next > Cancel

Importing the Mitmproxy CA



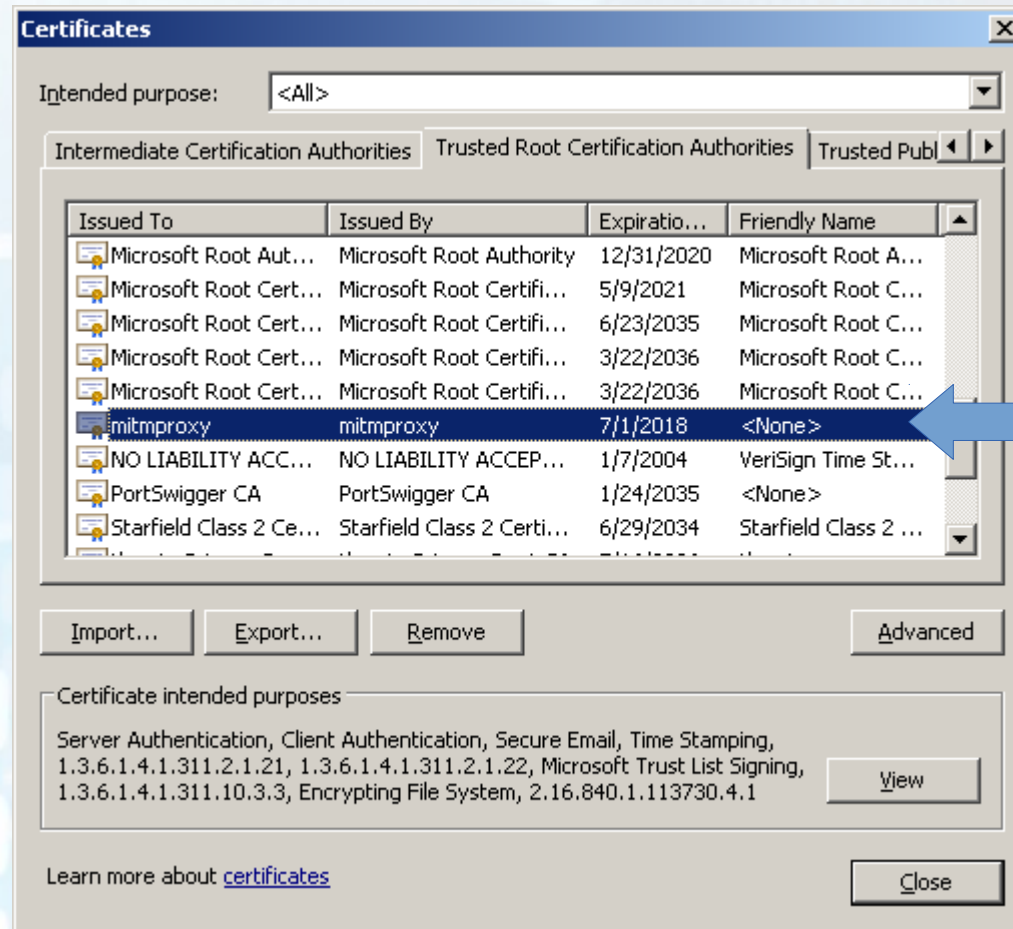
Importing the Mitmproxy CA



Importing the Mitmproxy CA

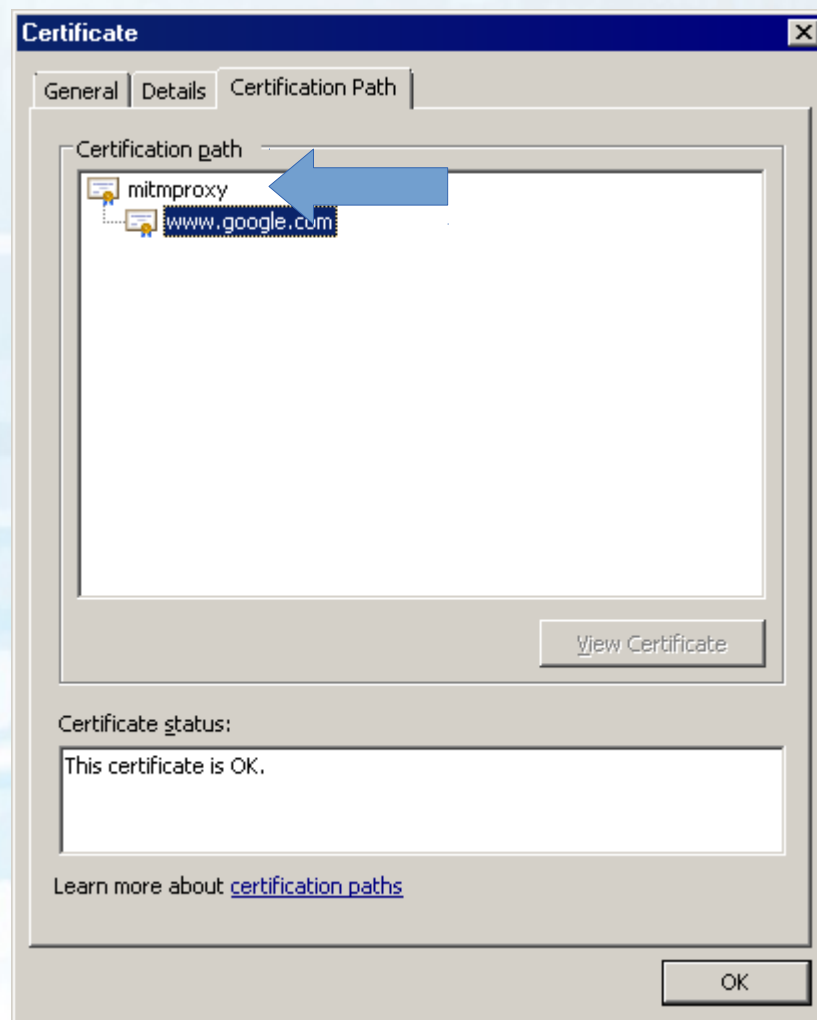


Verify CA Installation



Tools → Internet Options → Content → Certificates → Trusted Root

Visiting an HTTPS Site



Visiting an HTTPS Site

```
user@mitm: ~  
File Edit View Search Terminal Help  
>> GET https://www.google.com/  
  ← 200 text/html 53.29kB 1.18MB/s  
GET https://www.google.com/images/hpp/ic_wahlberg_product_core_48.png8.png  
  ← 304 [no content] 50.65kB/s  
GET https://www.google.com/images/nav_logo242.png  
  ← 304 [no content] 82.93kB/s  
GET https://www.gstatic.com/og/_/js/k=og.og2.en_US.Q76M4N9eFug.0/rt=j/m=def/exm=in,fot/d=1/ed=1/rs=AA2YrTtwUfn6ZjmIMkOSiVIiefBPpPtMlw  
  ← 304 [no content] 38.77kB/s  
GET https://www.google.com/xjs/_/js/k=xjs.s.en_US.VGbNrZB1_Fg.0/m=sx,c,sb,cdo  
s,cr,elog,jsa,r,hsm,qsm,j,p,d,csi/am=AJQkAYRE_H8ICLcQLEgFGAwC/rt=j/d=1/t=  
zcms/rs=ACT90oHgdR9e0sCj4qppAUn92XxocqHfQ  
  ← 304 [no content] 89.81kB/s  
GET https://www.google.com/images/branding/googlelogo/1x/googlelogo_color_272  
x92dp.png  
  ← 304 [no content] 88.05kB/s  
GET https://ssl.gstatic.com/gb/images/il_1967ca6a.png  
  ← 304 [no content] 86.09kB/s  
POST https://www.google.com/_/og/promos/z  
  ← 200 text/html 22B 13.15kB/s  
GET https://www.google.com/textinputassistant/tia.png  
  ← 304 [no content] 72.06kB/s  
[3/14] [showhost] ? :help [*:8080]
```

Troubleshooting

- Applications that don't follow IE proxy settings
 - Use default gateway technique
 - As on Android (up next)

Troubleshooting

- Applications that don't follow IE CA settings
 - Mozilla Firefox
 - Most applications with embedded TLS libraries
 - OpenSSL
 - NSS
 - Java applications
 - More to come later

Android Applications

Android Network Settings

ISE LAB

Signal strength **Excellent**

Security **WPA2 PSK**

Password

Show password

Show advanced options

Proxy settings **None**

IP settings **Static**

IP address **10.42.16.105**

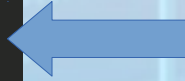
Gateway **10.42.16.102**

Network prefix length **24**

DNS 1 **8.8.8.8**

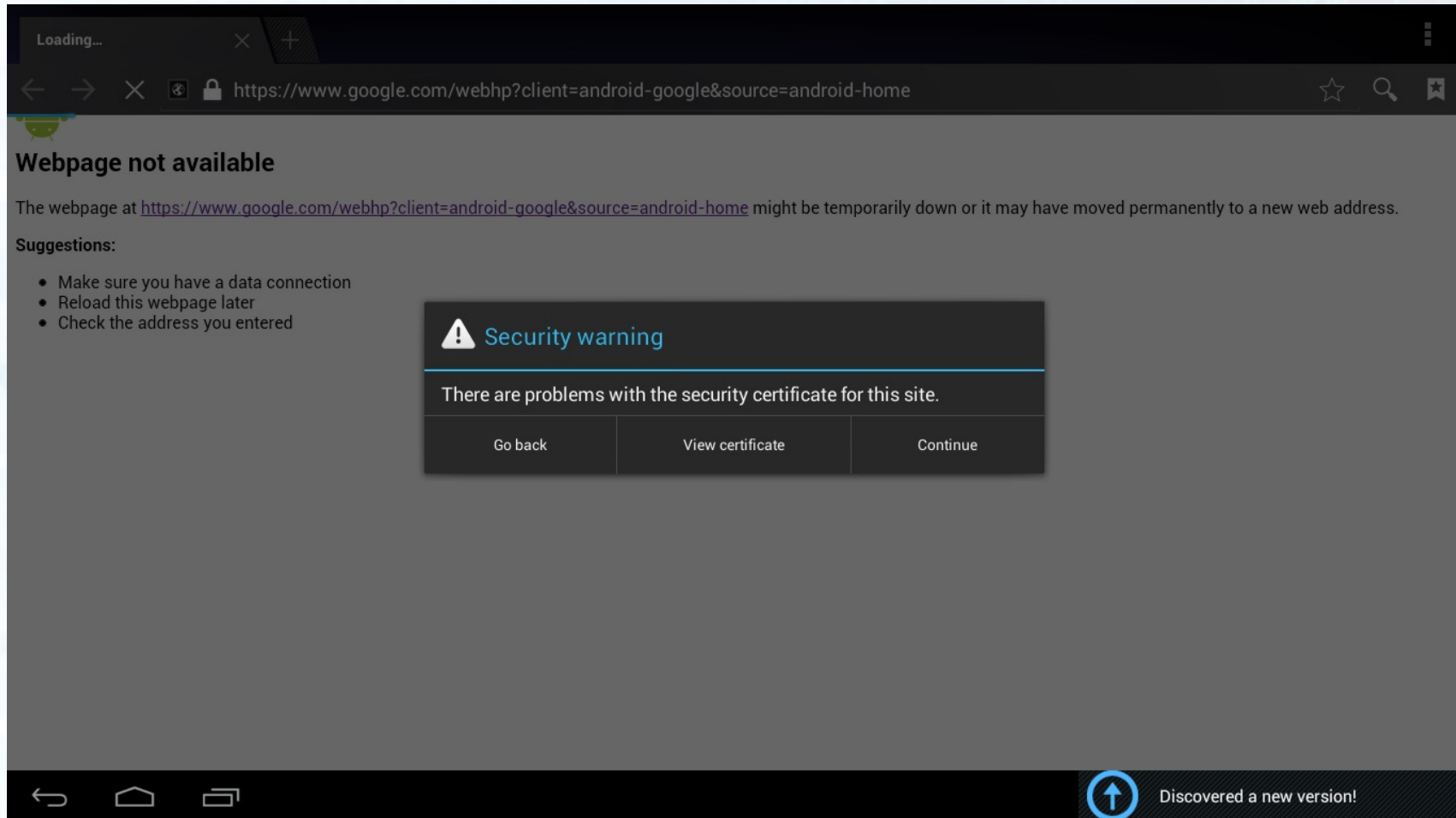
DNS 2 **8.8.4.4**

Cancel Connect

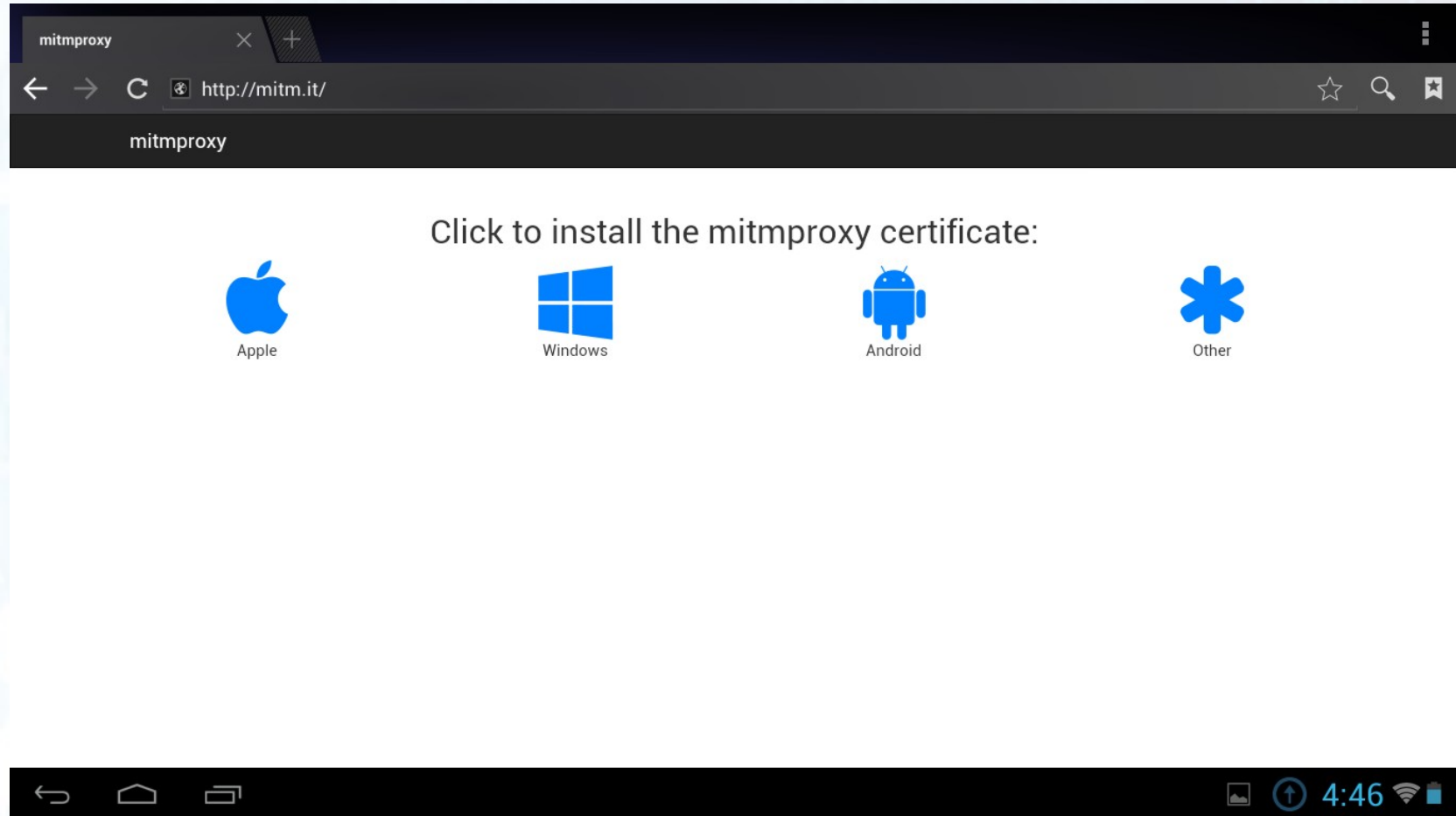


Ensure Mitmproxy is in transparent mode!

Android Browser Certificate Warning

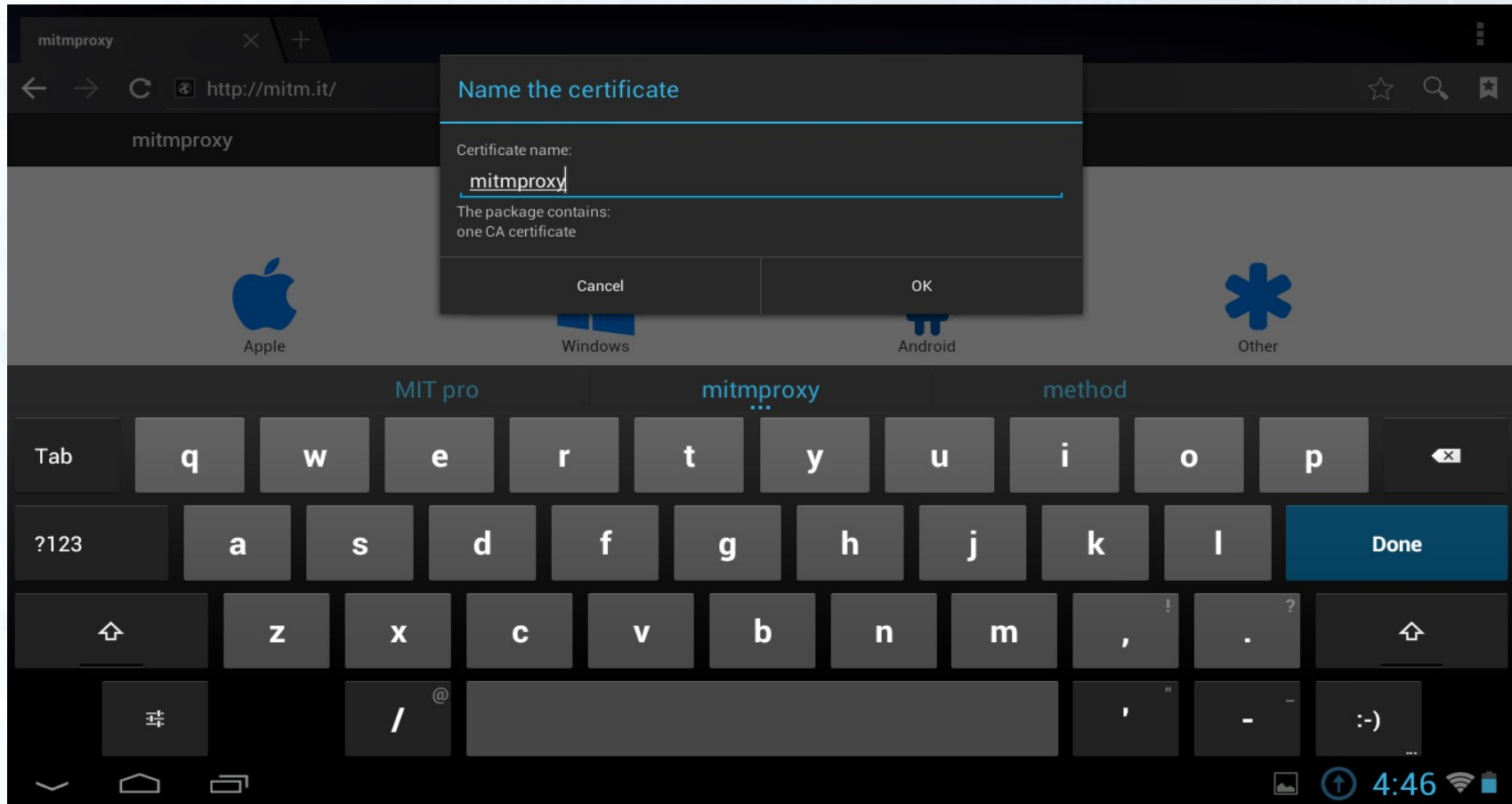


Mitmproxy CA Download Page

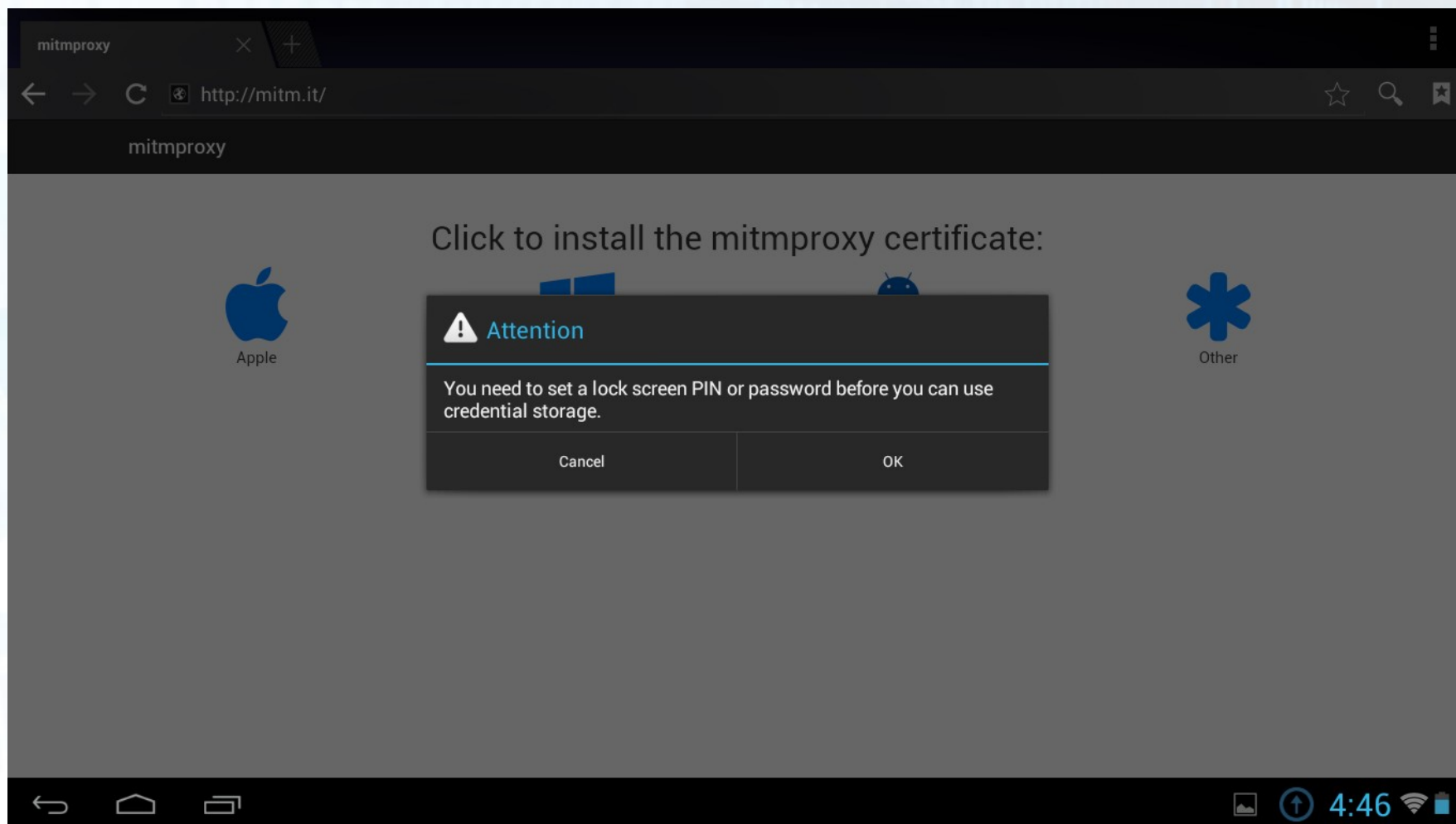


<http://mitm.it/>

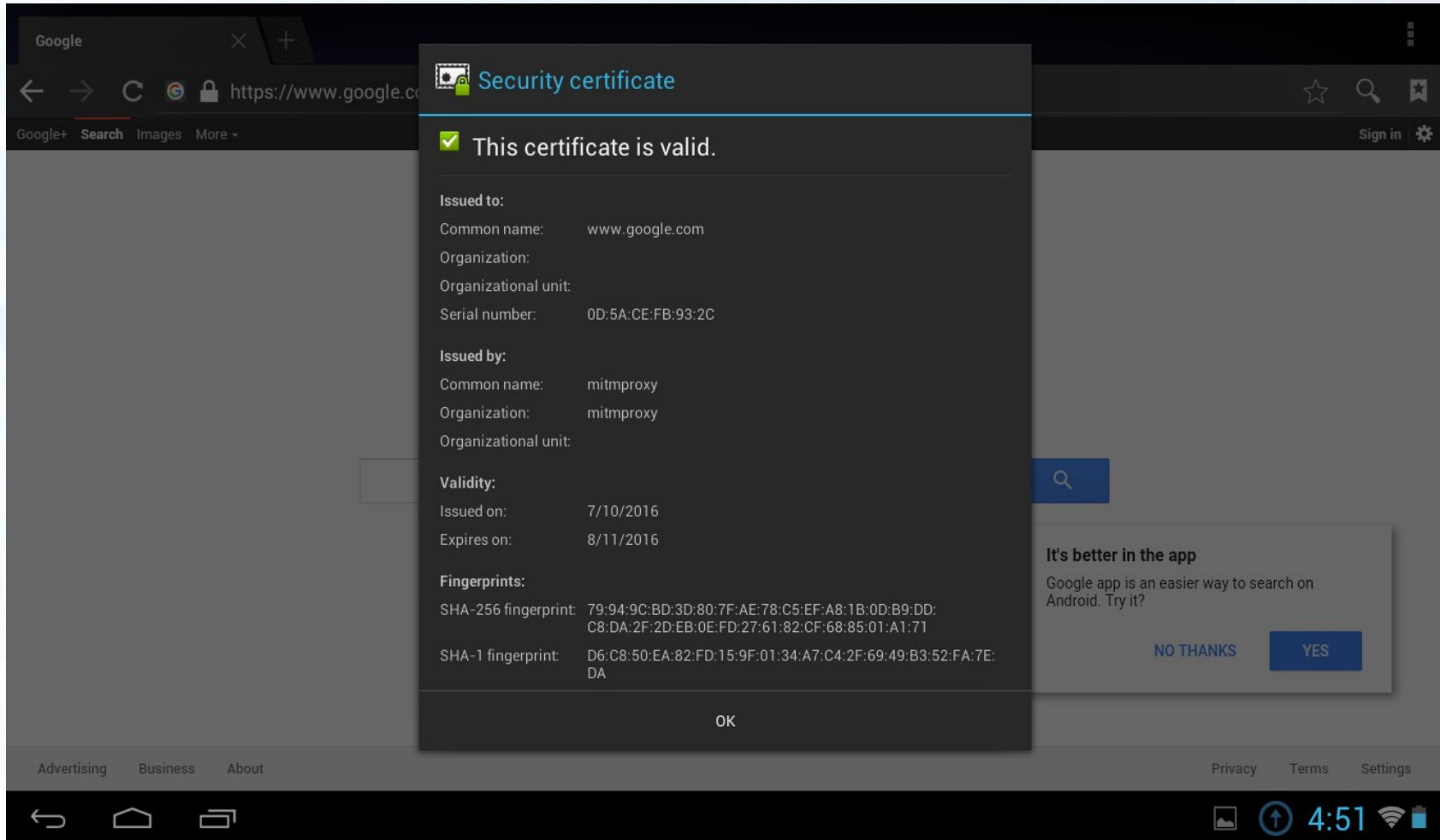
Mitmproxy CA Installation



Android Credential Storage PIN



Android Browser



Visiting an HTTPS Site

```
user@mitm: ~  
File Edit View Search Terminal Help  
>> GET https://www.google.com/  
  ← 200 text/html 53.29kB 1.18MB/s  
GET https://www.google.com/images/hpp/ic_wahlberg_product_core_48.png8.png  
  ← 304 [no content] 50.65kB/s  
GET https://www.google.com/images/nav_logo242.png  
  ← 304 [no content] 82.93kB/s  
GET https://www.gstatic.com/og/_/js/k=og.og2.en_US.Q76M4N9eFug.0/rt=j/m=def/exm=in,fot/d=1/ed=1/rs=AA2YrTtwUfn6ZjmIMkOSiVIiefBPaPtMlw  
  ← 304 [no content] 38.77kB/s  
GET https://www.google.com/xjs/_/js/k=xjs.s.en_US.VGbNrZB1_Fg.0/m=sx,c,sb,cdo  
s,cr,eelog,jsa,r,hsm,qsm,j,p,d,csi/am=AJQkAYRE_H8ICLcQLEgFGAwC/rt=j/d=1/t=  
zcms/rs=ACT90oHgdR9e0sCj4qppAUn92XxocqHfQ  
  ← 304 [no content] 89.81kB/s  
GET https://www.google.com/images/branding/googlelogo/1x/googlelogo_color_272  
x92dp.png  
  ← 304 [no content] 88.05kB/s  
GET https://ssl.gstatic.com/gb/images/il_1967ca6a.png  
  ← 304 [no content] 86.09kB/s  
POST https://www.google.com/_/og/promos/z  
  ← 200 text/html 22B 13.15kB/s  
GET https://www.google.com/textinputassistant/tia.png  
  ← 304 [no content] 72.06kB/s  
[3/14] [showhost] ? :help [*:8080]
```


Troubleshooting

- Proxy setting on Android is unreliable
 - Notice we used default gateway
- Android Nougat (7) trusted CA changes
- Certificate pinning
- JNI

iOS Applications

iOS Network Settings

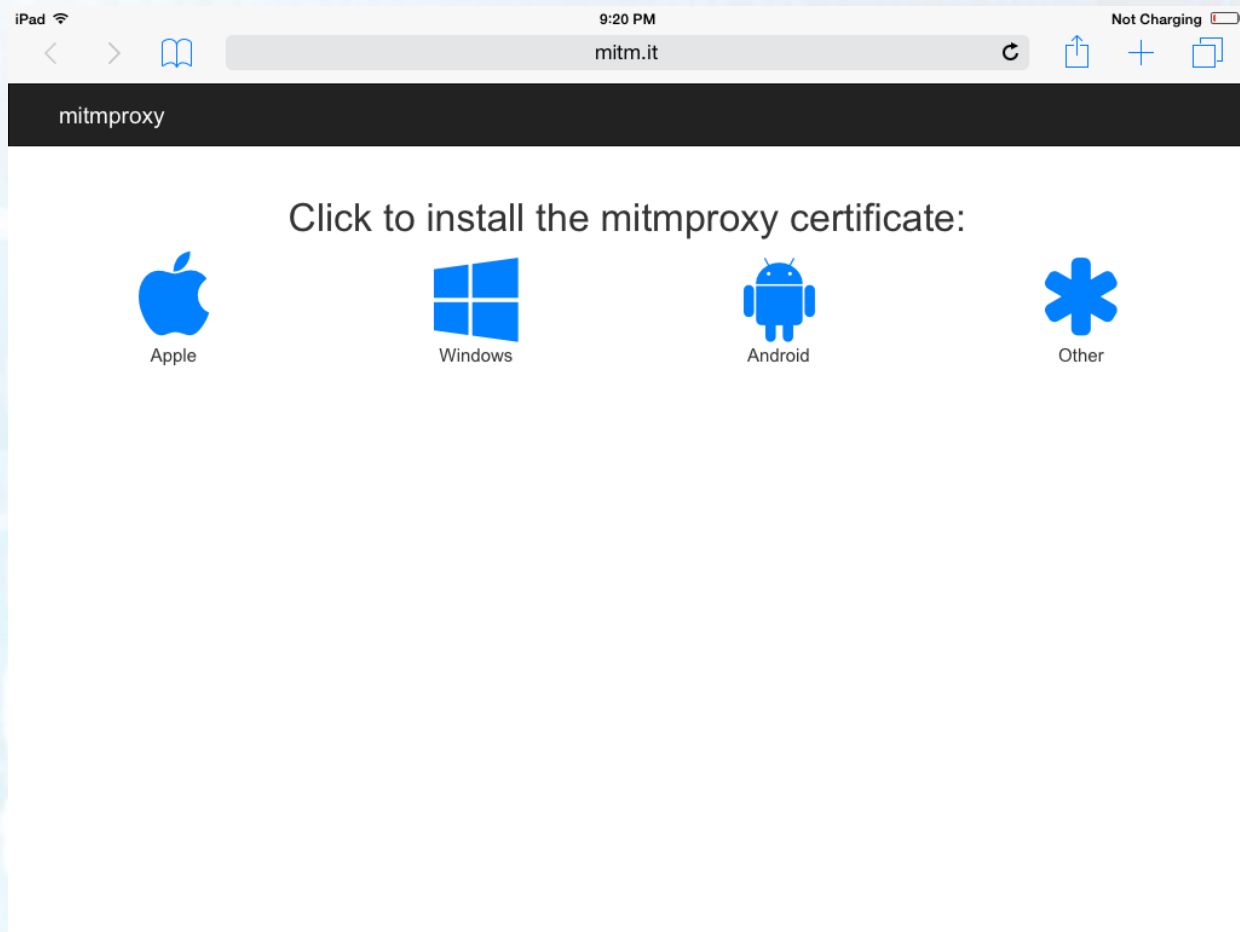
The screenshot shows the iPad's Settings app. The left sidebar lists various settings categories: Airplane Mode, Wi-Fi (selected), Bluetooth, Notifications, Control Center, Do Not Disturb, General, Display & Brightness, Wallpaper, Sounds, Passcode, Privacy, and iCloud. The main screen displays the Wi-Fi settings for the 'ISE LAB' network. A blue arrow points to the 'Router' field, which is set to '10.42.16.101'. Other fields include IP Address (10.42.16.102), Subnet Mask (255.255.255.0), and DNS (8.8.8.8). The IP Address section has three tabs: DHCP, BootP, and Static (selected). The HTTP PROXY section has three tabs: Off (selected), Manual, and Auto.

Field	Value
IP Address	10.42.16.102
Subnet Mask	255.255.255.0
Router	10.42.16.101
DNS	8.8.8.8
Search Domains	

HTTP PROXY: Off (selected), Manual, Auto

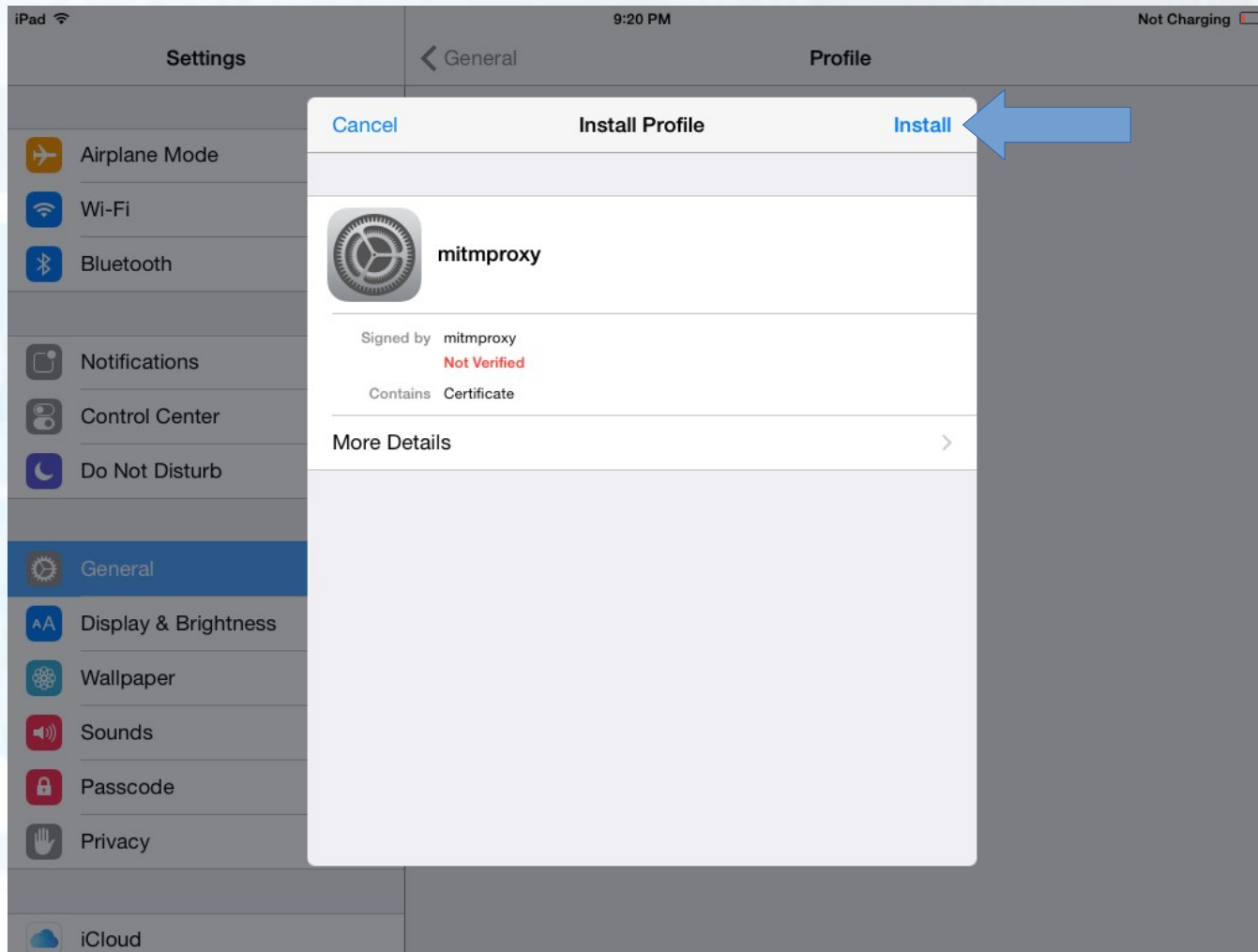
HTTP Proxy usually works too, on iOS.
Ensure Mitmproxy is in the correct mode!

Mitmproxy CA Download Page



<http://mitm.it/>

Mitmproxy CA Installation



Visiting an HTTPS Site

```
user@mitm: ~  
File Edit View Search Terminal Help  
>> GET https://www.google.com/  
  ← 200 text/html 53.29kB 1.18MB/s  
GET https://www.google.com/images/hpp/ic_wahlberg_product_core_48.png8.png  
  ← 304 [no content] 50.65kB/s  
GET https://www.google.com/images/nav_logo242.png  
  ← 304 [no content] 82.93kB/s  
GET https://www.gstatic.com/og/_/js/k=og.og2.en_US.Q76M4N9eFug.0/rt=j/m=def/exm=in,fot/d=1/ed=1/rs=AA2YrTtwUfn6ZjmIMkOSiVIiefBPpPtMlw  
  ← 304 [no content] 38.77kB/s  
GET https://www.google.com/xjs/_/js/k=xjs.s.en_US.VGbNrZB1_Fg.0/m=sx,c,sb,cdo  
s,cr,elog,jsa,r,hsm,qsm,j,p,d,csi/am=AJQkAYRE_H8ICLcQLEgFGAwC/rt=j/d=1/t=zcms/rs=ACT90oHgdR9e0sCj4qppAUn92XxocqHfQ  
  ← 304 [no content] 89.81kB/s  
GET https://www.google.com/images/branding/googlelogo/1x/googlelogo_color_272  
x92dp.png  
  ← 304 [no content] 88.05kB/s  
GET https://ssl.gstatic.com/gb/images/il_1967ca6a.png  
  ← 304 [no content] 86.09kB/s  
POST https://www.google.com/_/og/promos/z  
  ← 200 text/html 22B 13.15kB/s  
GET https://www.google.com/textinputassistant/tia.png  
  ← 304 [no content] 72.06kB/s  
[3/14] [showhost] ? :help [*:8080]
```

Troubleshooting

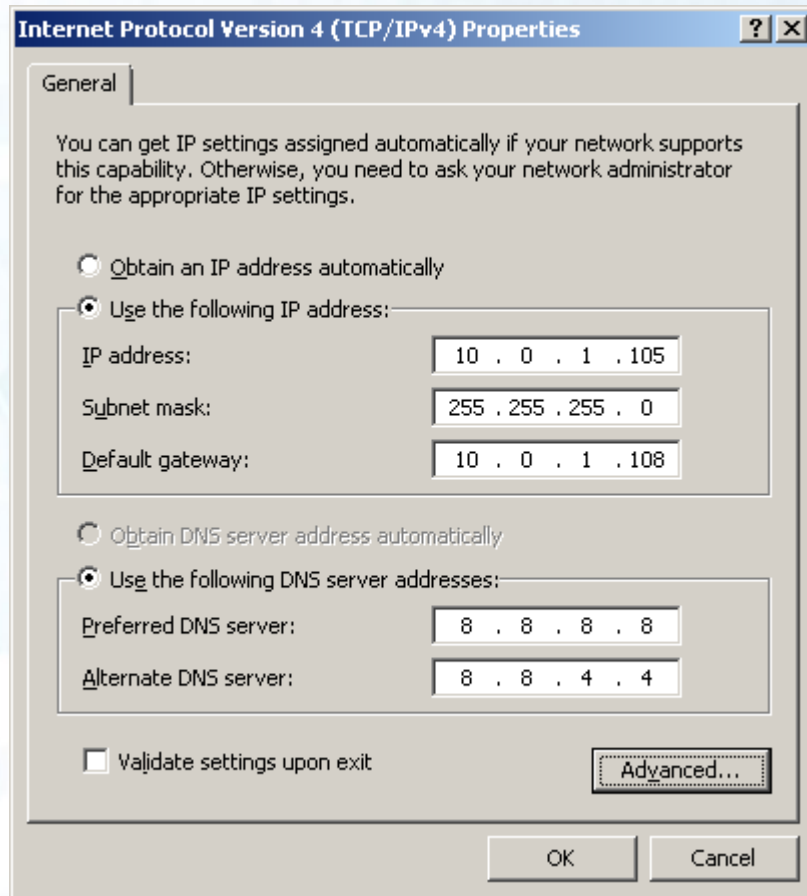
- Certificate pinning
 - See SSL Kill Switch and similar tools

Java Applications

Java Applications

- Java does *not* always use Internet Explorer or other system-wide proxy and certificate settings

Configure Default Gateway



Could also configure Java proxy settings.

Ensure Mitmproxy is in transparent mode!

Sample Java Application (jcurl)

```
C:\>java -jar jcurl-all.jar -e url https://www.google.com/
```

Sample Java Application (jcurl)

```
C:\>java -jar jcurl-all.jar -e url https://www.google.com/  
Starting jCurl in C:\
```

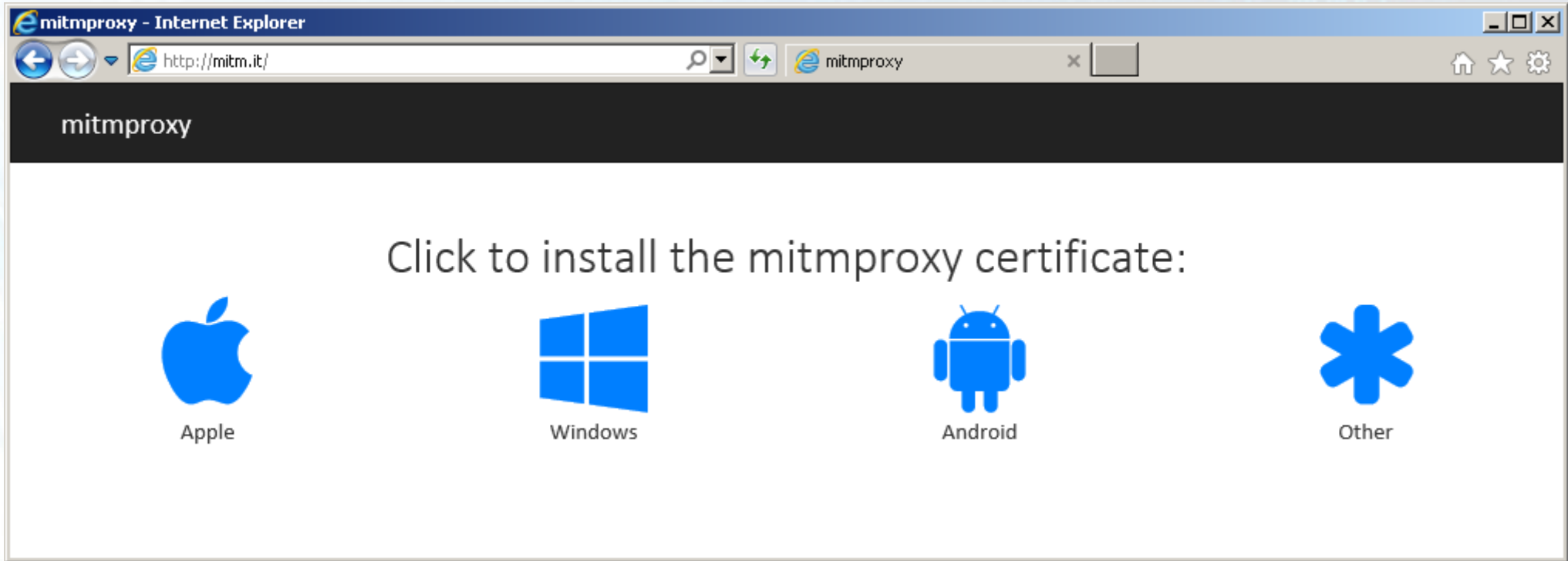
```
Sending 'GET' request to URL : https://www.google.com/  
Exception in thread "main" javax.net.ssl.SSLHandshakeException:  
sun.security.validator.ValidatorException: PKIX path building failed:  
sun.security.provider.certpath.SunCertPathBuilderException: unable to  
find valid certification path to requested target  
    at sun.security.ssl.Alerts.getSSLException(Unknown Source)  
    at sun.security.ssl.SSLSocketImpl.fatal(Unknown Source)  
    at sun.security.ssl.Handshaker.fatalSE(Unknown Source)  
    at sun.security.ssl.Handshaker.fatalSE(Unknown Source)  
    at sun.security.ssl.ClientHandshaker.serverCertificate(Unknown  
Source)  
    at sun.security.ssl.ClientHandshaker.processMessage(Unknown  
Source)  
    at sun.security.ssl.Handshaker.processLoop(Unknown Source)  
    at sun.security.ssl.Handshaker.process_record(Unknown Source)  
    ...
```

Java Certificate Authority Store

```
$JAVA_HOME/lib/security/cacerts
```

Different paths for 32-bit and 64-bit JREs
JDK vs. JRE location varies

Downloading Mitmproxy CA



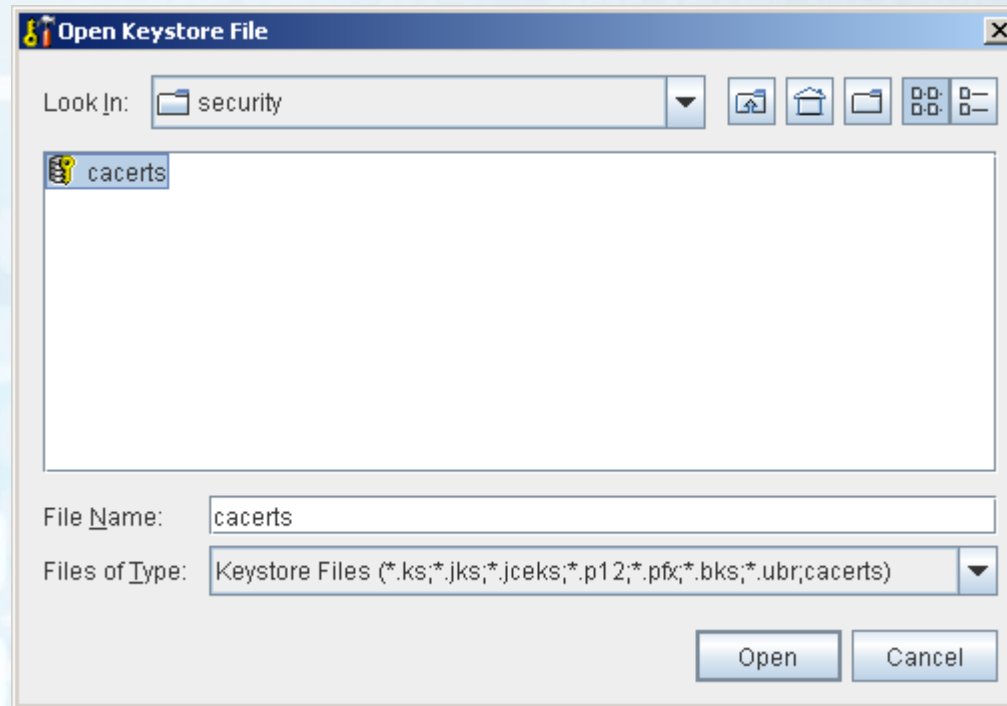
<http://mitm.it/>

Portecle



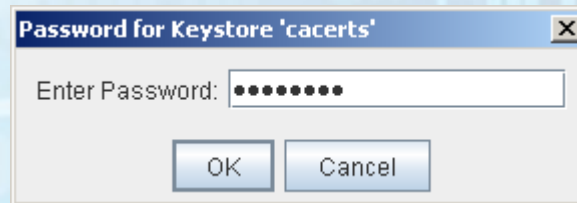
```
java -jar portecle.jar
```

Open cacerts File



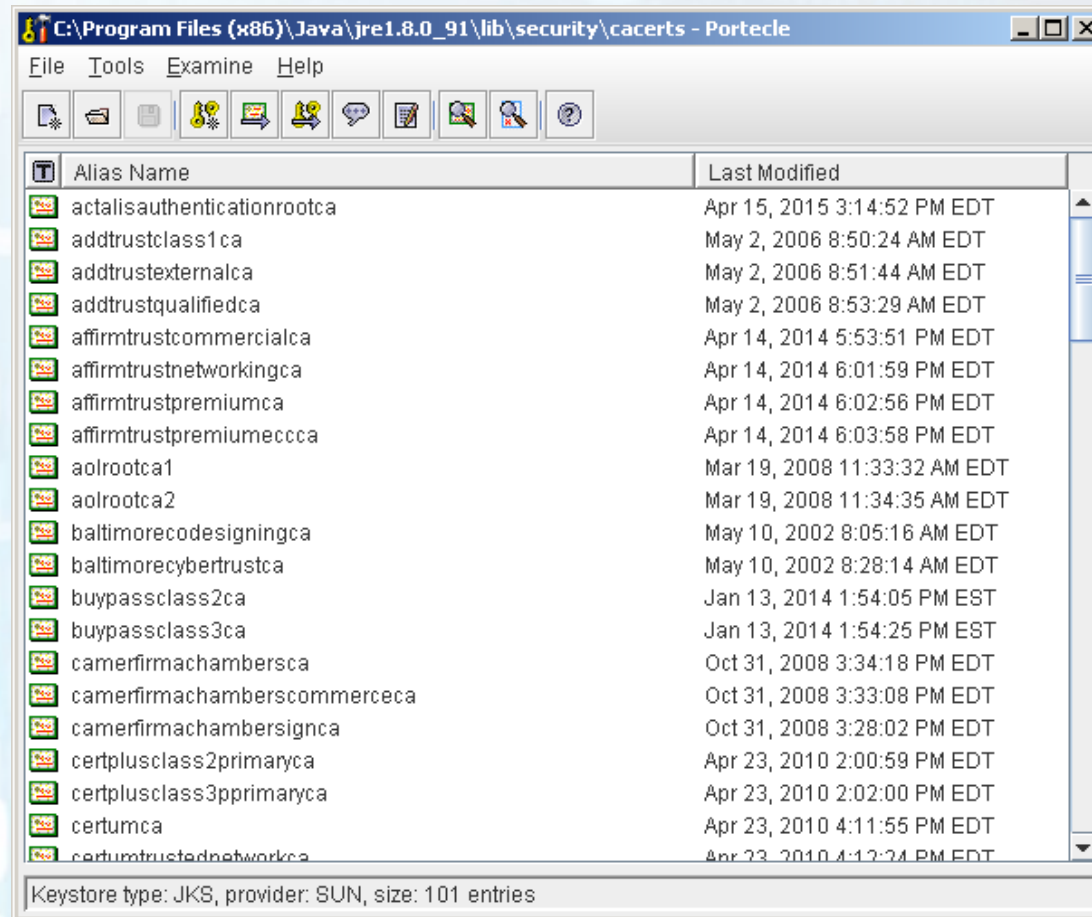
Must have sufficient privileges to edit the file!

CA Store Password



“changeit”

Viewing Trusted CAs



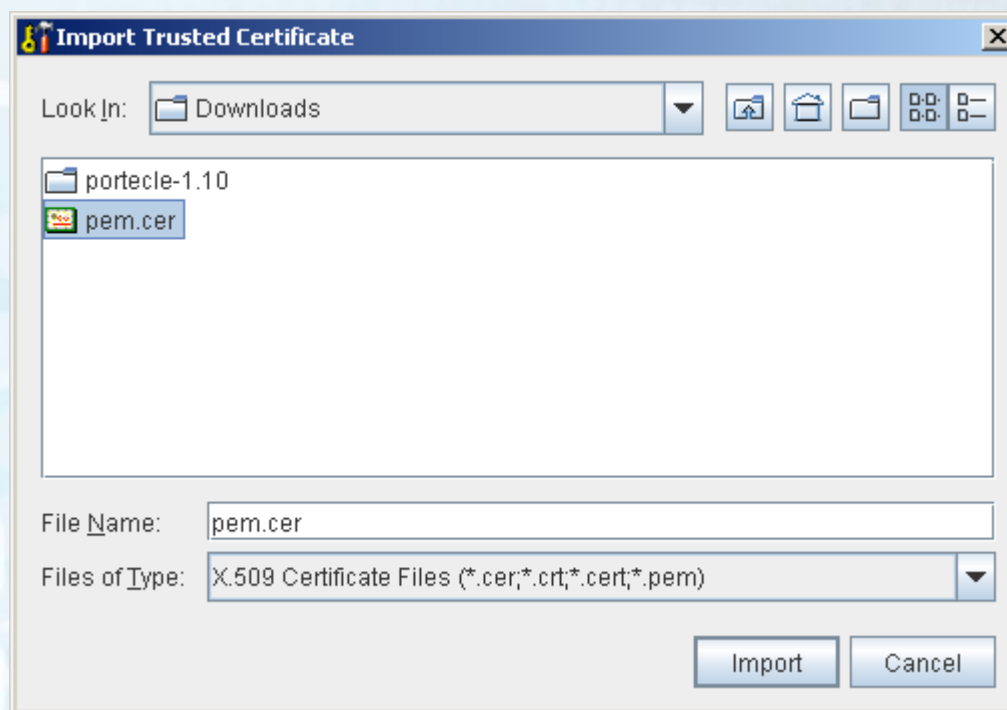
The screenshot shows a Java application window titled "C:\Program Files (x86)\Java\jre1.8.0_91\lib\security\cacerts - Portecle". The window contains a table of trusted certificates with columns for "Alias Name" and "Last Modified". The table lists various certificates such as "actalisauthenticationrootca", "aolrootca1", and "certumca". At the bottom of the window, it states "Keystore type: JKS, provider: SUN, size: 101 entries".

Alias Name	Last Modified
actalisauthenticationrootca	Apr 15, 2015 3:14:52 PM EDT
addtrustclass1ca	May 2, 2006 8:50:24 AM EDT
addtrustexternalca	May 2, 2006 8:51:44 AM EDT
addtrustqualifiedca	May 2, 2006 8:53:29 AM EDT
affirmtrustcommercialca	Apr 14, 2014 5:53:51 PM EDT
affirmtrustnetworkingca	Apr 14, 2014 6:01:59 PM EDT
affirmtrustpremiumca	Apr 14, 2014 6:02:56 PM EDT
affirmtrustpremiumecccac	Apr 14, 2014 6:03:58 PM EDT
aolrootca1	Mar 19, 2008 11:33:32 AM EDT
aolrootca2	Mar 19, 2008 11:34:35 AM EDT
baltimorecodesigningca	May 10, 2002 8:05:16 AM EDT
baltimorecybertrustca	May 10, 2002 8:28:14 AM EDT
buypassclass2ca	Jan 13, 2014 1:54:05 PM EST
buypassclass3ca	Jan 13, 2014 1:54:25 PM EST
camerfirmachambersca	Oct 31, 2008 3:34:18 PM EDT
camerfirmachamberscommerceca	Oct 31, 2008 3:33:08 PM EDT
camerfirmachamberssignca	Oct 31, 2008 3:28:02 PM EDT
certplusclass2primaryca	Apr 23, 2010 2:00:59 PM EDT
certplusclass3pprimaryca	Apr 23, 2010 2:02:00 PM EDT
certumca	Apr 23, 2010 4:11:55 PM EDT
certumtrustednetworkca	Apr 23, 2010 4:12:24 PM EDT

Keystore type: JKS, provider: SUN, size: 101 entries

Tools → Import Trusted Certificate

Select Mitmproxy CA PEM File



Accept Trust Warning



Accept Trust Warning

Certificate Details for 'pem.cer' ✕

← Certificate 1 of 1 →

Version: 3

Subject: O=mitmproxy, CN=mitmproxy

Issuer: O=mitmproxy, CN=mitmproxy

Serial Number: 0D5A 8EBC 740B

Valid From: Jul 11, 2016 1:48:13 PM EDT

Valid Until: Jul 1, 2018 1:48:13 PM EDT

Public Key: RSA (1,024 bits)

Signature Algorithm: SHA1withRSA

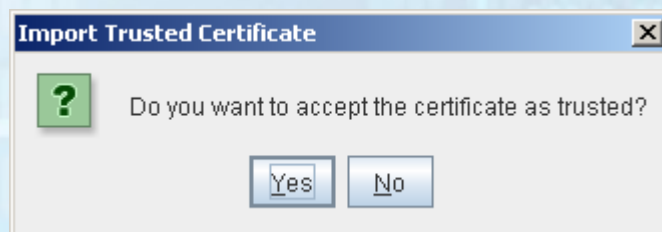
SHA-1 Fingerprint: 3C:B7:01:41:EE:CC:1F:D7:5B:69:A2:18:2B:27:8C:41:F3:48:C6:6C

MD5 Fingerprint: 1C:FD:3D:2A:28:B3:BB:F7:C3:A1:51:9C:6B:48:5A:6E

Extensions PEM Encoding

OK

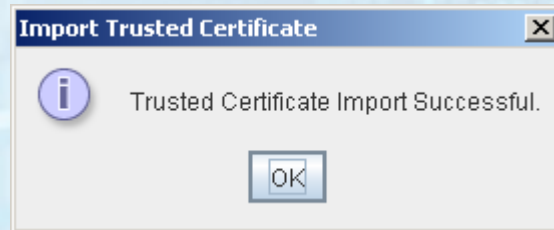
Accept Trust Warning



Name CA Certificate



Import Certificate



Retry jCurl HTTPS Download

```
C:\>java -jar jcurl-all.jar -e url https://www.google.com/  
Starting jCurl in C:\
```

```
Sending 'GET' request to URL : https://www.google.com/  
Response Code : 200
```

```
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"  
lang="en">
```

```
...
```

Advanced Techniques

Certificate Pinning

Certificate Pinning

- Or “certificate authority pinning”
- Overcoming certificate pinning
 - Debugging
 - Patching

Java: Custom TrustManager

Java: Custom TrustManager

The screenshot shows a Mozilla Firefox browser window with the title "X509TrustManager (Java Platform SE 8) - Mozilla Firefox". The address bar shows the URL "http://docs.oracle.com/javase/8/docs/api/javax/net/ssl/X509TrustManager.html". The page content is as follows:

Interface X509TrustManager

All SuperInterfaces:
TrustManager

All Known Implementing Classes:
X509ExtendedTrustManager

public interface **X509TrustManager**
extends TrustManager

Instance of this interface manage which X509 certificates may be used to authenticate the remote side of a secure socket. Decisions may be based on trusted certificate authorities, certificate revocation lists, online status checking or other means.

Since:
1.4

Method Summary

All Methods	Instance Methods	Abstract Methods
Modifier and Type	Method and Description	
void	checkClientTrusted (X509Certificate[] chain, String authType) Given the partial or complete certificate chain provided by the peer, build a certificate path to a trusted root and return if it can be validated and is trusted for client SSL authentication based on the authentication type.	
void	checkServerTrusted (X509Certificate[] chain, String authType) Given the partial or complete certificate chain provided by the peer, build a certificate path to a trusted root and return if it can be validated and is trusted for server SSL authentication based on the authentication type.	

trustmanager ^ v Highlight All Match Case 2 of 6 matches

SSLContext.init

init

```
public final void init(KeyManager[] km,  
                      TrustManager[] tm, ←  
                      SecureRandom random)  
    throws KeyManagementException
```

Initializes this context. Either of the first two parameters may be null in which case the installed security providers will be searched for the highest priority implementation of the appropriate factory. Likewise, the secure random parameter may be null in which case the default implementation will be used.

Only the first instance of a particular key and/or trust manager implementation type in the array is used. (For example, only the first `javax.net.ssl.X509KeyManager` in the array will be used.)

Parameters:

`km` - the sources of authentication keys or null

`tm` - the sources of peer authentication trust decisions or null

`random` - the source of randomness for this generator or null

Throws:

`KeyManagementException` - if this operation fails

null TrustManager = default

Suppressing Custom TrustManager

```
$ jdb CustomCA  
Initializing jdb ...  
>
```


Suppressing Custom TrustManager

```
$ jdb CustomCA  
Initializing jdb ...  
> stop in javax.net.ssl.SSLContext.getInstance(java.lang.String)  
Deferring breakpoint  
javax.net.ssl.SSLContext.getInstance(java.lang.String).  
It will be set after the class is loaded.
```

Suppressing Custom TrustManager

```
$ jdb CustomCA
Initializing jdb ...
> stop in javax.net.ssl.SSLContext.getInstance(java.lang.String)
Deferring breakpoint
javax.net.ssl.SSLContext.getInstance(java.lang.String).
It will be set after the class is loaded.
> run
run CustomCA
Set uncaught java.lang.Throwable
Set deferred uncaught java.lang.Throwable
>
```

Suppressing Custom TrustManager

```
$ jdb CustomCA
Initializing jdb ...
> stop in javax.net.ssl.SSLContext.getInstance(java.lang.String)
Deferring breakpoint
javax.net.ssl.SSLContext.getInstance(java.lang.String).
It will be set after the class is loaded.
> run
run CustomCA
Set uncaught java.lang.Throwable
Set deferred uncaught java.lang.Throwable
>
VM Started: Set deferred breakpoint
javax.net.ssl.SSLContext.getInstance(java.lang.String)

Breakpoint hit: "thread=main",
javax.net.ssl.SSLContext.getInstance(), line=155 bci=0

main[1]
```

Suppressing Custom TrustManager

```
main[1] step up
```

```
>
```

```
Step completed: "thread=main", CustomCA.main(), line=29 bci=19  
29          SSLContext sctx = SSLContext.getInstance("TLS");
```

```
main[1] locals
```

```
Method arguments:
```

```
args = instance of java.lang.String[0] (id=940)
```

```
Local variables:
```

```
tmf = instance of javax.net.ssl.TrustManagerFactory(id=941)
```

```
trustManagers = instance of javax.net.ssl.TrustManager[1]  
(id=942)
```

```
main[1]
```

Suppressing Custom TrustManager

```
main[1] step up
```

```
>
```

```
Step completed: "thread=main", CustomCA.main(), line=29 bci=19  
29          SSLContext sctx = SSLContext.getInstance("TLS");
```

```
main[1] locals
```

```
Method arguments:
```

```
args = instance of java.lang.String[0] (id=940)
```

```
Local variables:
```

```
tmf = instance of javax.net.ssl.TrustManagerFactory(id=941)
```

```
trustManagers = instance of javax.net.ssl.TrustManager[1]  
(id=942)
```

```
main[1] set trustManagers=null
```

```
trustManagers=null = null
```

```
main[1] cont
```

```
> <!doctype html><html itemscope=""
```

```
itemtype="http://schema.org/WebPage" lang="en">
```

C: OpenSSL

C: OpenSSL



```
long SSL_get_verify_result(const SSL *ssl);
```

Returns X509_V_OK (0) on successful validation
Non-zero otherwise

wget Call to OpenSSL

```
430e84:      e8 a7 31 fd ff      callq  404030
<SSL_get_verify_result@plt>
430e89:      48 85 c0            test   %rax,%rax
430e8c:      48 89 c3            mov    %rax,%rbx
430e8f:      c6 44 24 0f 01     movb  $0x1,0xf(%rsp)
430e94:      0f 85 a6 02 00 00   jne   431140
<SSLv3_client_method@plt+0x2c270>
```


wget Call to OpenSSL

```
430e84:      90                nop
430e85:      90                nop
430e86:      90                nop
430e87:      31 c0            xor     %eax,%eax
430e89:      48 85 c0        test   %rax,%rax
430e8c:      48 89 c3        mov    %rax,%rbx
430e8f:      c6 44 24 0f 01  movb  $0x1,0xf(%rsp)
430e94:      0f 85 a6 02 00 00  jne   431140
<SSLv3_client_method@plt+0x2c270>
```

Certificate Magic Numbers

Certificate Magic Numbers

- PEM format
 - —BEGIN CERTIFICATE—
 - —END CERTIFICATE—
- DER format
 - 0x30 0x82 0x01 (1024-bit RSA certificate)
 - 0x30 0x82 0x02 (2048-bit RSA certificate)

Mistakes in Implementing HTTPS

Mistakes in Implementing HTTPS

- Failing to Verify Certificates
 - No verification at all
 - CA, but not hostname
 - OpenSSL limitations
 - Hostname, but not CA

Mistakes in Implementing HTTPS

- Allowing anonymous cipher suites
 - Confusing OpenSSL APIs

Mistakes in Implementing HTTPS

- Allowing NULL encryption cipher suites

Mistakes in Implementing HTTPS

- Outdated Libraries
 - Apple “goto fail”
 - ChangeCipherSpec Injection
 - DROWN
 - Logjam

Conclusion

Bypassing TLS

- Modify trusted CA list
 - Windows
 - Android
 - iOS
 - Java
- Overcome certificate pinning
 - Debugging (e.g., Java)
 - Patching (e.g., C + OpenSSL, certificate magic numbers)
- Leverage implementation mistakes

Bypassing TLS

- Questions?