SSL PINNING

AND BYPASSES

(ANDROID & IOS)

BY

ANANT SHRIVASTAVA

ANANT SHRIVASTAVA

- Information Security Consultant
- Admin - Dev - Security
- null + OWASP + G4H
- http://anantshri.info and @anantshri
- Trainer : Blackhat, RuxCon, NullCon, g0s, c0c0n
- Speaker : Nullcon, c0c0n, ClubHack, RootConf

Android Tamer Code Vigilant

SSL PINNING

Another layer to achieve secure communication specially protection against MiTM

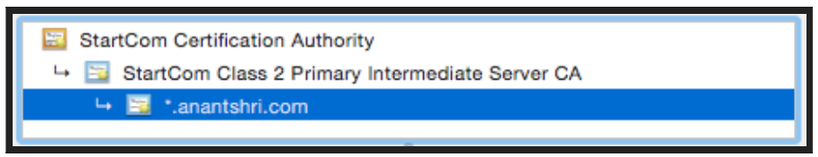
HOW MITM WORKS

1. Add Root CA of interception proxy in Browser.
2. Divert traffic via interception proxy, proxy handles SSL Connection

Client **<--HTTPS-->** Interception Proxy **<--HTTPS-->** Server

1. **Browser validates** that **certificate** is **issued** by **Trusted CA** and **allows** connection

BEFORE



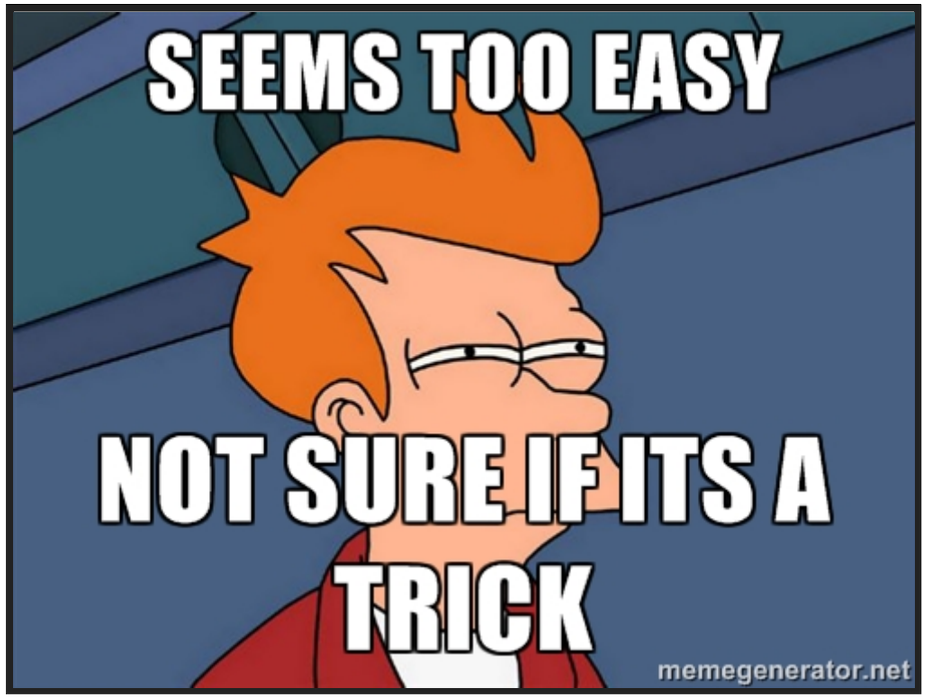
AFTER

PKI IS BROKEN

1. System Trust all CA in Trust Store (PortSwigger CA)
2. System Trust's ROOT CA not certification chain
3. Any CA can issue certificate to any website (Diginotar, Trustwave, NIC and many more)
4. Certificate Stolen: Welcome to Revocation hell and CRL Nightmare
5. OCSP to the rescue **over port 80**
6. and many more

SO WHAT SHOULD WE DO

Pin Trust on our own certification chain and validate it at **Client Side**


SEEMS TOO EASY
NOT SURE IF ITS A TRICK
memegenerator.net

WHAT'S THE CATCH

1. What if you get new certificate from a different service provider
2. What if your certificate chain changes
3. What if certificate is revoked
4. What if certificate is stolen
5. What is Client is malacious
6. What if .....

Answer:

You need to update the code everytime certificate changes

ITS EASY PUSH AN UPDATE

SO WHY SHOULD I BOTHER

1. **Developers** : This hinders attacker from traffic interception. Adds another layer for Attacker to look for. Without Rooting devices its nearly imposible to bypass it so far.
2. **Pentesters** : This hinders you from inspecting application (be ready for bypasses section)

SSL PINNING IN ANDROID & BYPASS

HOW TO IMPLEMENT SSL PINNING

Multiple ways

1. Store Certificate in sqlite and use it directly
2. Store sha1 hashes and compare
3. Store sha1 hash of one element in chain and compare

DEMO DETAILS

1. We have used a helper library called okhttp by square
2. Pins sha1 hashes of entire chain or set of elements in chain

https://github.com/square/okhttp/blob/master/okhttp/src/main/java/com/squareup/okhttp/CertificatePinner.java

```
131   */
132  public final class CertificatePinner {
133    public static final CertificatePinner DEFAULT = new Builder().build();
134
135    private final Map<String, Set<ByteString>> hostnameToPins;
136
137    private CertificatePinner(Builder builder) {
138      hostnameToPins = Util.immutableMap(builder.hostnameToPins);
139    }
140
141    /**
142     * Confirms that at least one of the certificates pinned for {@code hostname}
143     * is in {@code peerCertificates}. Does nothing if there are no certificates
144     * pinned for {@code hostname}. OkHttp calls this after a successful TLS
145     * handshake, but before the connection is used.
146     *
147     * @throws SSLPeerUnverifiedException if {@code peerCertificates} don't match
148     *     the certificates pinned for {@code hostname}.
149     */
150    public void check(String hostname, List<Certificate> peerCertificates)
151        throws SSLPeerUnverifiedException {
152
153      Set<ByteString> pins = findMatchingPins(hostname);
154
155      if (pins == null) return;
```

DEMO SSL PINNING

BYPASS DEMO

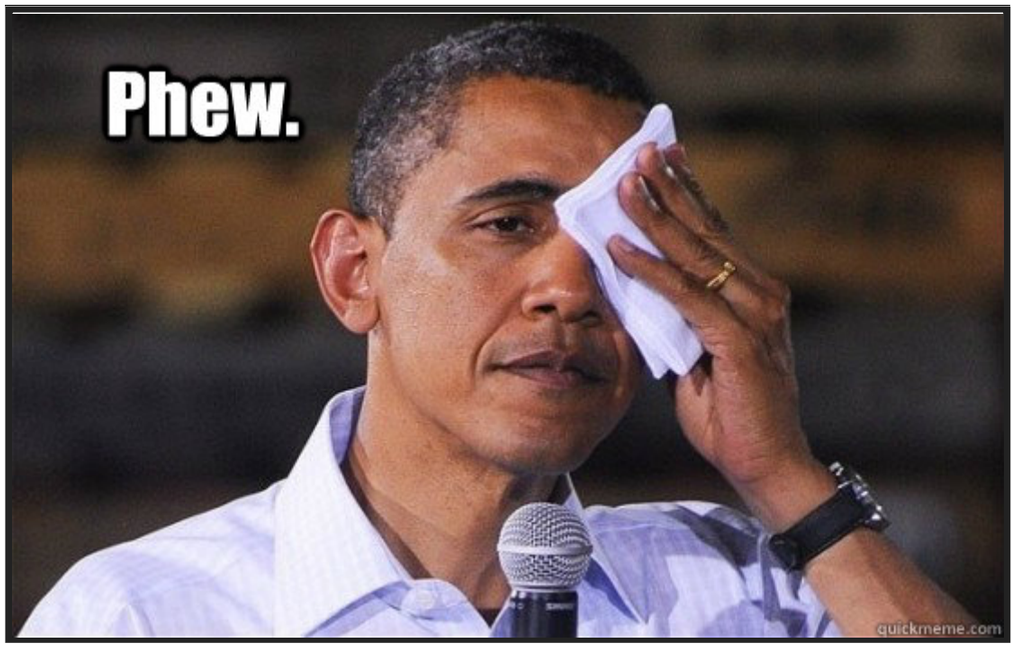
HOW BYPASS WORKED

1. Xposed Framework Hooks into all Function Calls
2. Whenever request is made for **check** function inside **com.squareup.okhttp.CertificatePinner** class, return true

```
  /* hooking OKHTTP by SQUAREUP */
  /* com/squareup/okhttp/CertificatePinner.java available online @ https://github.com/square/okhttp/blob/master/okhttp/src/main/java/com/squareup/
  /*public void check(String hostname, List<Certificate> peerCertificates)
throws SSLPeerUnverifiedException{}*/
/* Either returns true or a exception so blanket return true */
/* Tested against version 2.5 */
  findAndHookMethod("com.squareup.okhttp.CertificatePinner", lpparam.classLoader, "check", String.class, List.class, new XC_MethodReplacement() {
      @Override
      protected Object replaceHookedMethod(MethodHookParam methodHookParam) throws Throwable {
          return true;
      }
  });
```

Ref: https://github.com/Fuzion24/JustTrustMe/pull/12

ANDROID DEMO END



SSL PINNING IN IOS & BYPASS

HOW TO IMPLEMENT SSL PINNING

1. Use Third Party helper like
    1. SwiftHTTP
    2. TrustKit
2. Or Use **SecTrustEvaluate** via **NSURLConnectionDelegate** (third party helper basicly are wrapper to do this)

DEMO SSL PINNING

HOW TO BYPASS

1. https://github.com/iSECPartners/ios-ssl-kill-switch
2. https://github.com/nabla-c0d3/ssl-kill-switch2 (superseeds ios-ssl-kill-switch works on 9.0.2 also, doesn't work with itunes/appstore by default)

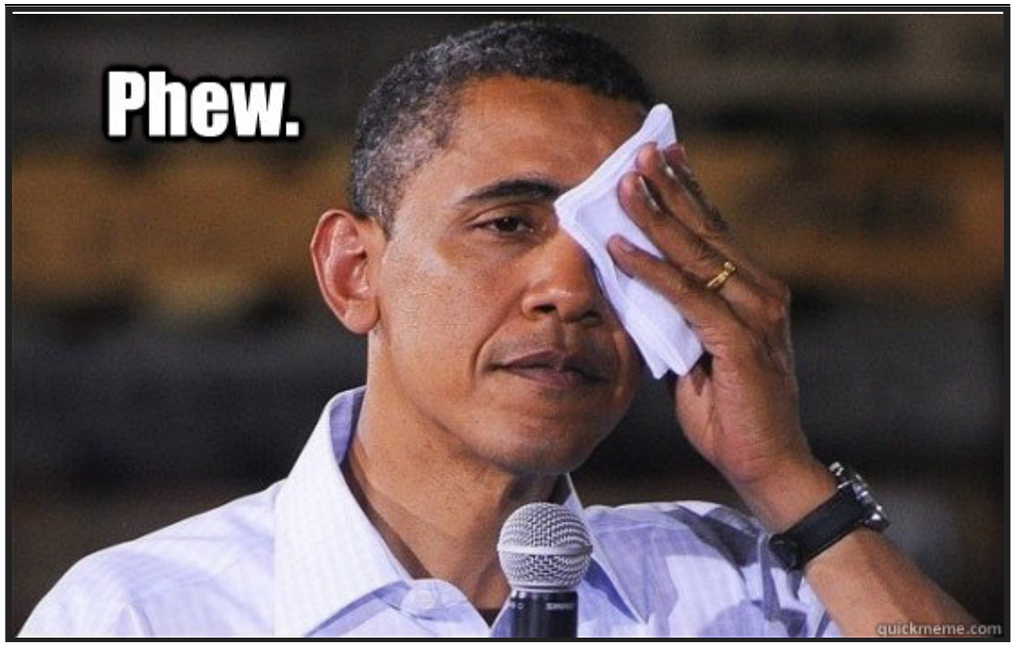
BYPASS DEMO

HOW IT WORKS

1. Leverages Cydia substrate
2. Uses MobileSubstrate to inject on process
3. Hooks on Secure Trasport API instead of SecTrustEvaluate or NSURL* as The Secure Transport API is "the lowest-level TLS implementation on iOS"
    1. Patch SSLCreateContext(): Disable the built-in certificate validation in all SSL contexts
    2. Patch SSLSetSessionOption(): Remove the ability to re-enable the built-in certificate validation
    3. Patch SSLHandshake(): Force a trust-all custom certificate validation

Reference: https://nabla-c0d3.github.io/blog/2013/08/20/ios-ssl-kill-switch-v0-dot-5-released/

IOS DEMO END



ANY QUESTIONS

ANANT SHRIVASTAVA

- Information Security Consultant
- Admin - Dev - Security
- null + OWASP + G4H
- http://anantshri.info and @anantshri
- Trainer : Blackhat, RuxCon, NullCon, g0s, c0c0n
- Speaker : Nullcon, c0c0n, ClubHack, RootConf

	
Android Tamer	Code Vigilant

REFERENCES

Generic

1. https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning

Android

1. https://github.com/square/okhttp
2. https://github.com/Fuzion24/JustTrustMe

iOS

1. https://github.com/daltoniam/SwiftHTTP
2. https://github.com/datatheorem/TrustKit
3. https://github.com/iSECPartners/ios-ssl-kill-switch
4. https://github.com/nabla-c0d3/ssl-kill-switch2/