



MANICCODE
SECURE CODING EDUCATION

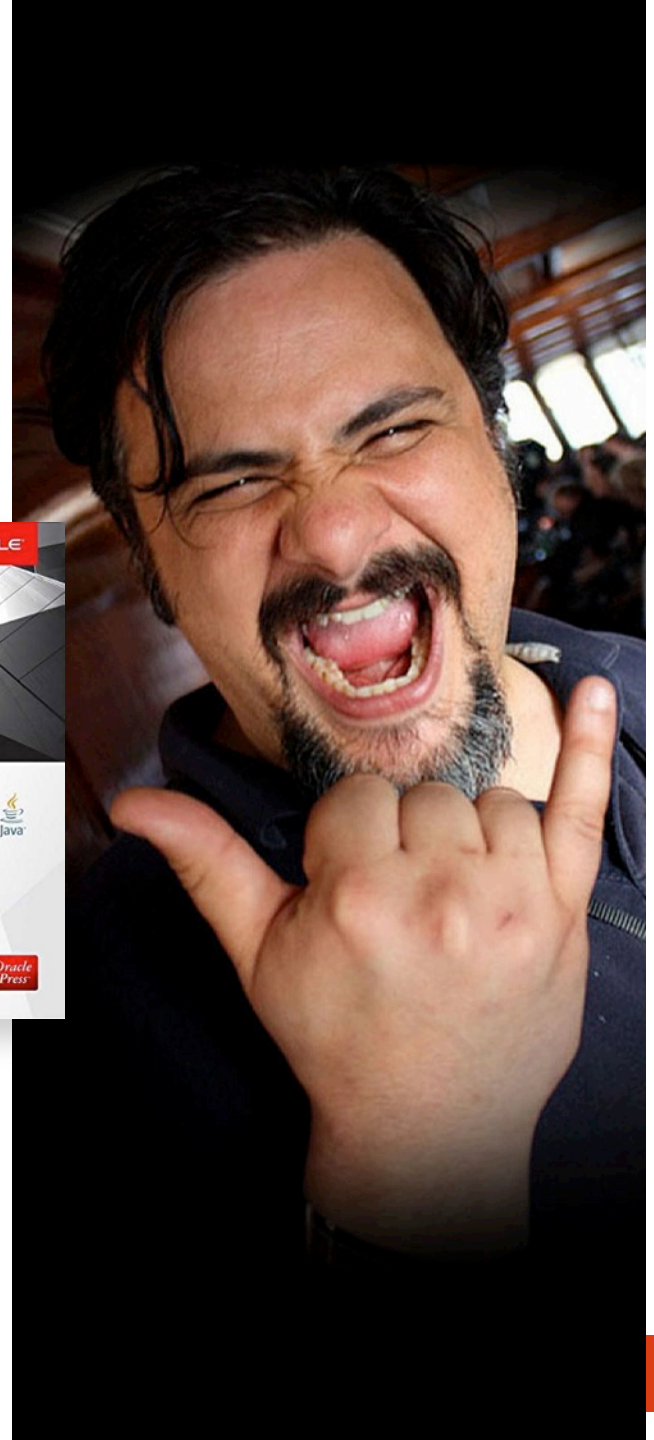
New Security Control Enhancements Java 8 and 9

A little background dirt...

jim@manico.net

 [@manicode](https://twitter.com/manicode)

- Author of "Iron-Clad Java, Building Secure Web Applications" from McGraw-Hill/Oracle-Press
- 20+ years of software development experience
- OWASP Volunteer and Former OWASP Global Board Member
- Kauai, Hawaii Resident



Java Enhancement Proposals

'*ohana*

(oh-ha-na)

MEANING:

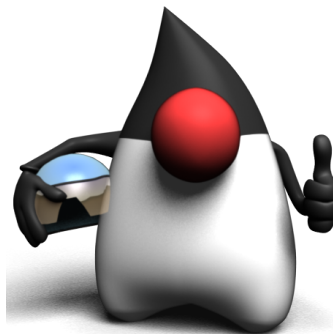
Family.

MOST COMMON USE:

In referring to the **WHOLE** family.

- JEP stands for a **JDK Enhancement Proposal**
- JEP's are how you **drive change** in the Java ecosystem.
- **Involvement** is actually a lot of work.
- Attention is given to people that **put in the work**.

- The way to make improvements or **get ideas seriously considered** is to do them via the JEP proposal process.
- **Mike Ernst and Werner Dietl** are good examples. They are the duo that **built type annotations** which we we will talk about soon.



Java 9 Security JEP's

Java 9 Security Enhancements

- There are 8 main security related JEPs for JDK 9:

219: Datagram Transport Layer Security (DTLS)

229: Create PKCS12 Keystores by Default

232: Improve Secure Application Performance

244: TLS Application-Layer Protocol Negotiation Extension

246: Leverage CPU Instructions for GHASH and RSA

249: OCSP Stapling for TLS

287: Support SHA-3 Hash Algorithms

288: Disable SHA-1 Certificates

akamai

(ah-ka-my)

MEANING:

Smart or Clever.

MOST COMMON USE:

Smart.

JEP 219

JEP 219

Datagram Transport Layer Security (DTLS)

<http://openjdk.java.net/jeps/219>

- **TLS is not sufficient** for a variety of datagram applications.
- Datagram applications still **need transport security!**
- JEP 219 defines an API for Datagram Transport Layer Security (DTLS) version 1.0 (**RFC 4347**) and 1.2 (**RFC 6347**).
- Small implementation that is **transport-independent** and similar to `javax.net.ssl.SSLEngine`.

JEP 244

JEP 244

TLS *Application-Layer Protocol Negotiation Extension*

<http://openjdk.java.net/jeps/244>

- JEP 244 extends javax.net.ssl package to support the TLS [Application Layer Protocol Negotiation \(ALPN\) Extension](#).
- When different protocols are supported on the same TCP or UDP port, ALPN allows a negotiation to determine **which protocol will be used with a TLS connection**.
- An important consumer of this enhancement is the **HTTP/2 client** (JEP 110).

JEP 249

JEP 249

OCSP Stapling for TLS

<http://openjdk.java.net/jeps/249>

- JEP 249 implements **OCSP Stapling for TLS clients**.
- Revocation (in general) does not work well.

It takes at least **10 days** for the revocation Information to fully *propagate*

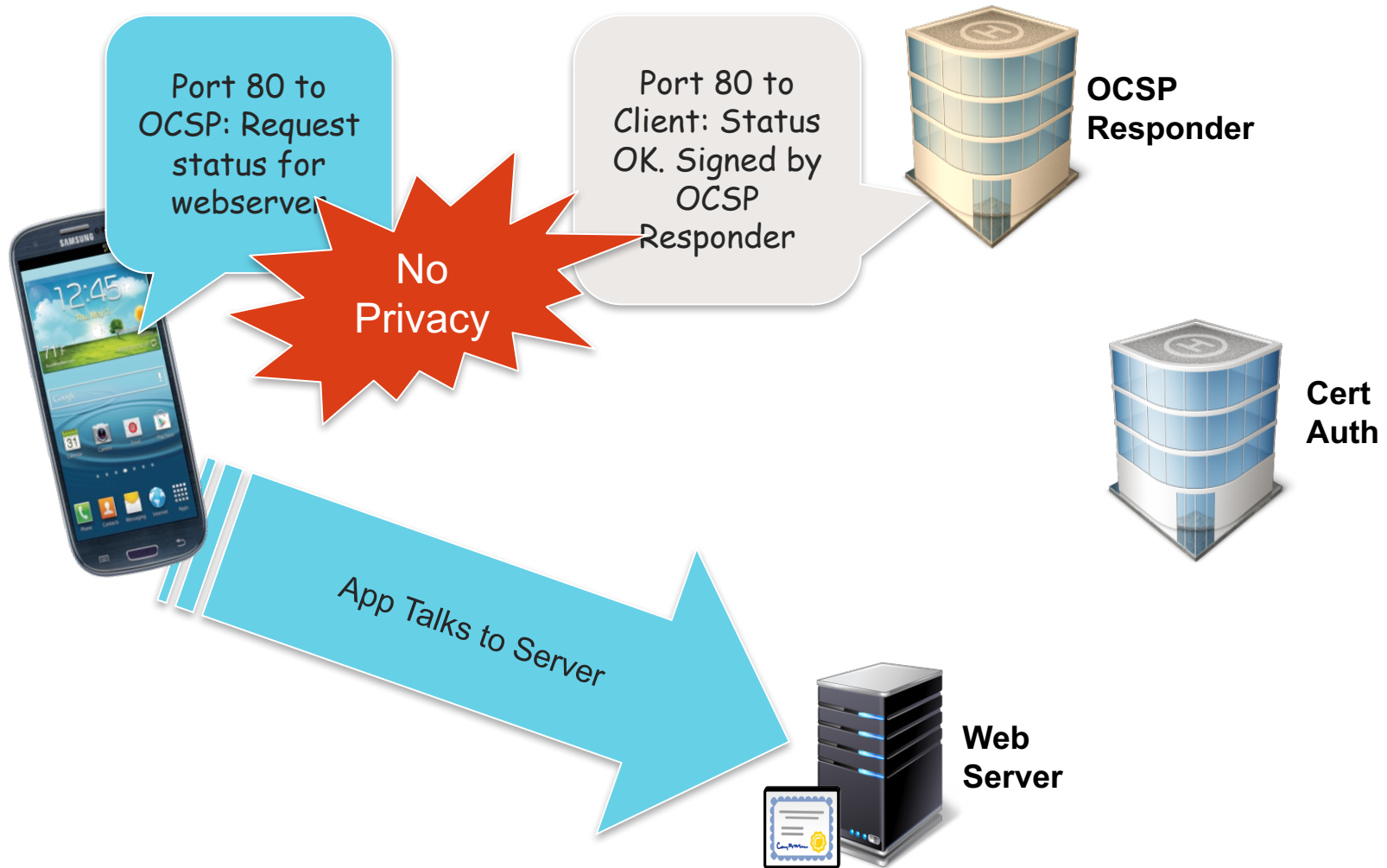
Browser *soft fail policy* makes revocation *ineffective*

OCSP request can be *intercepted* (more on this than later)

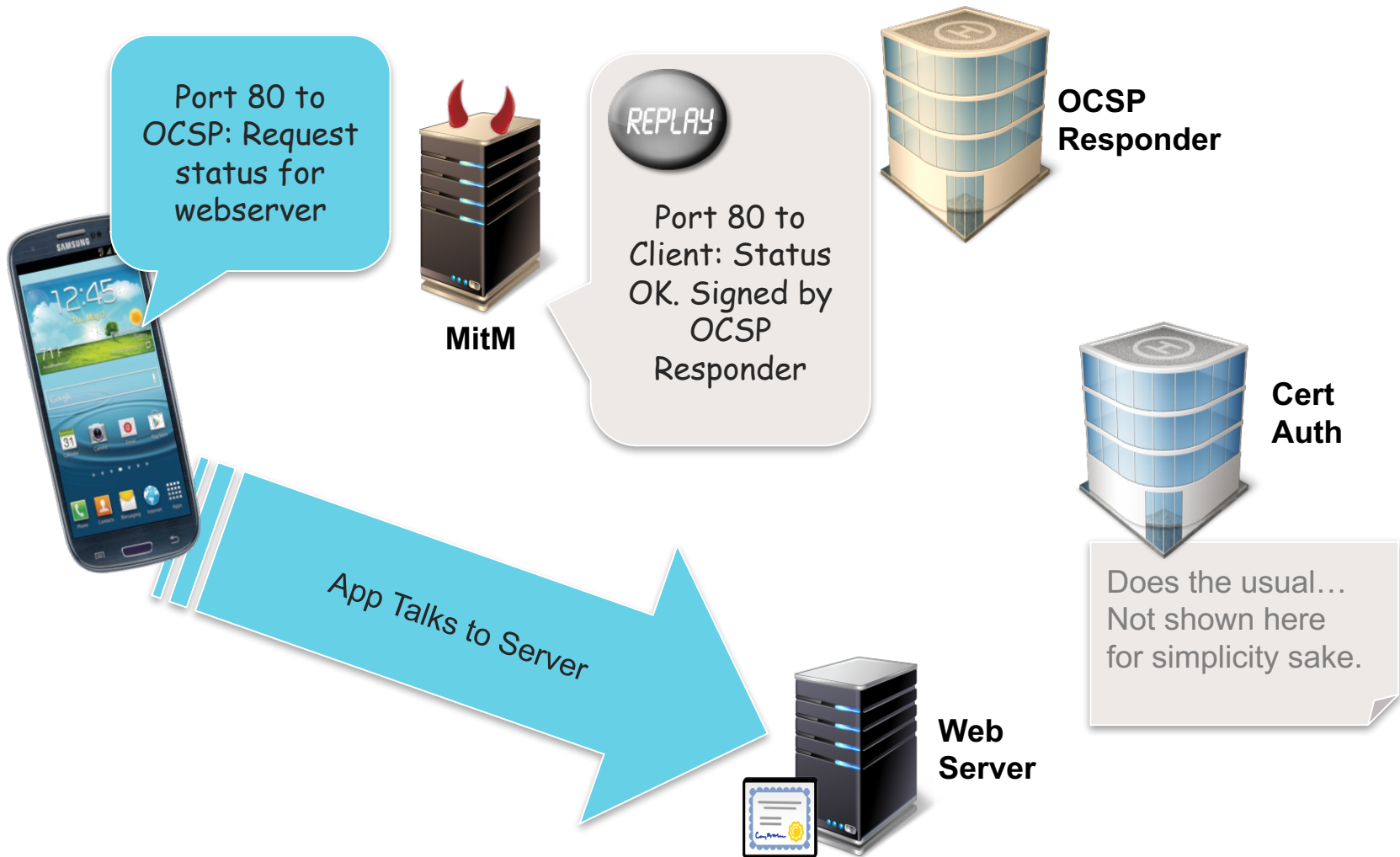
Most browsers ignore revocation for all certificates but EV certificates

Online Certificate Status Protocol (OCSP)

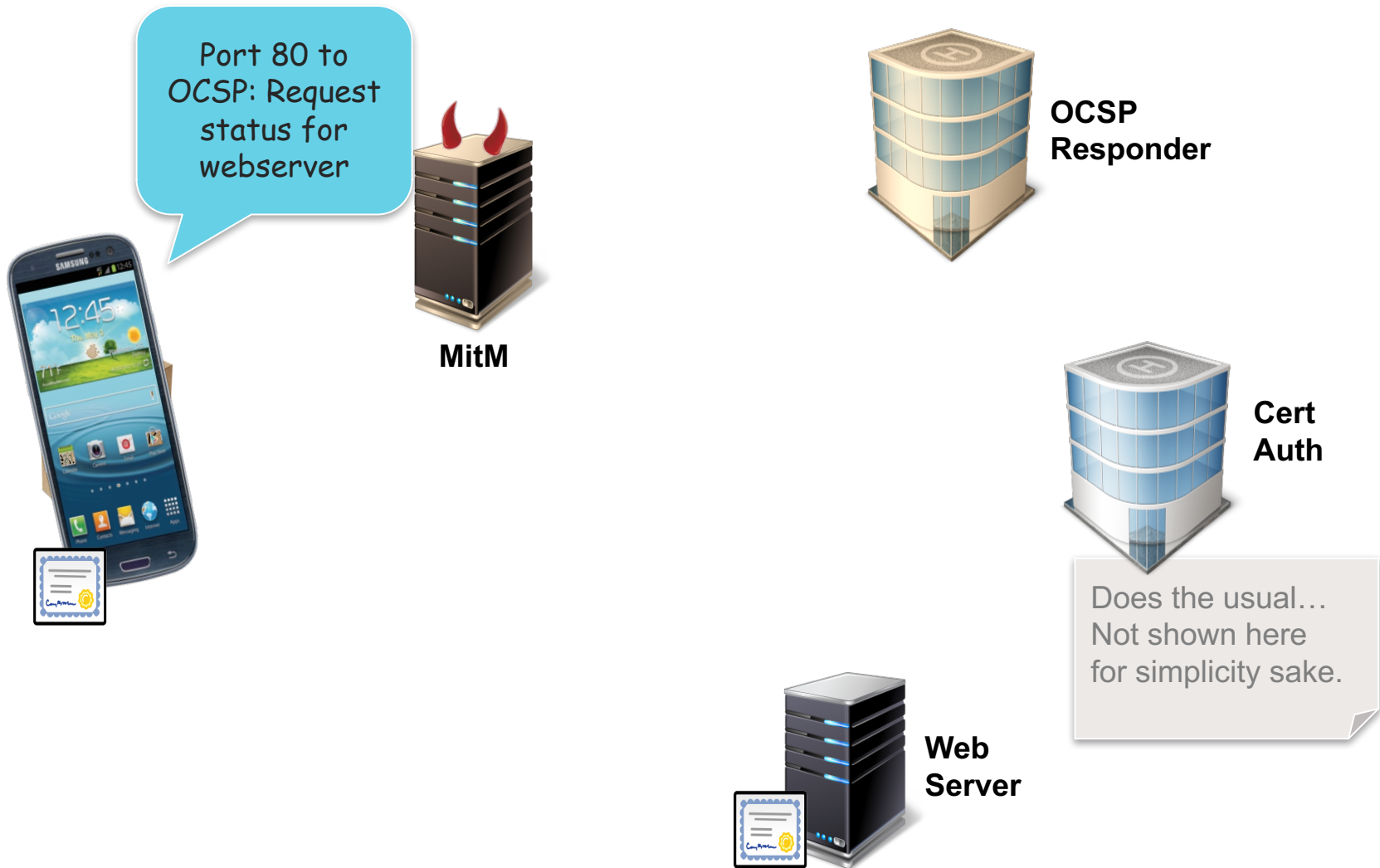
An alternative to certificate lists (CRL)



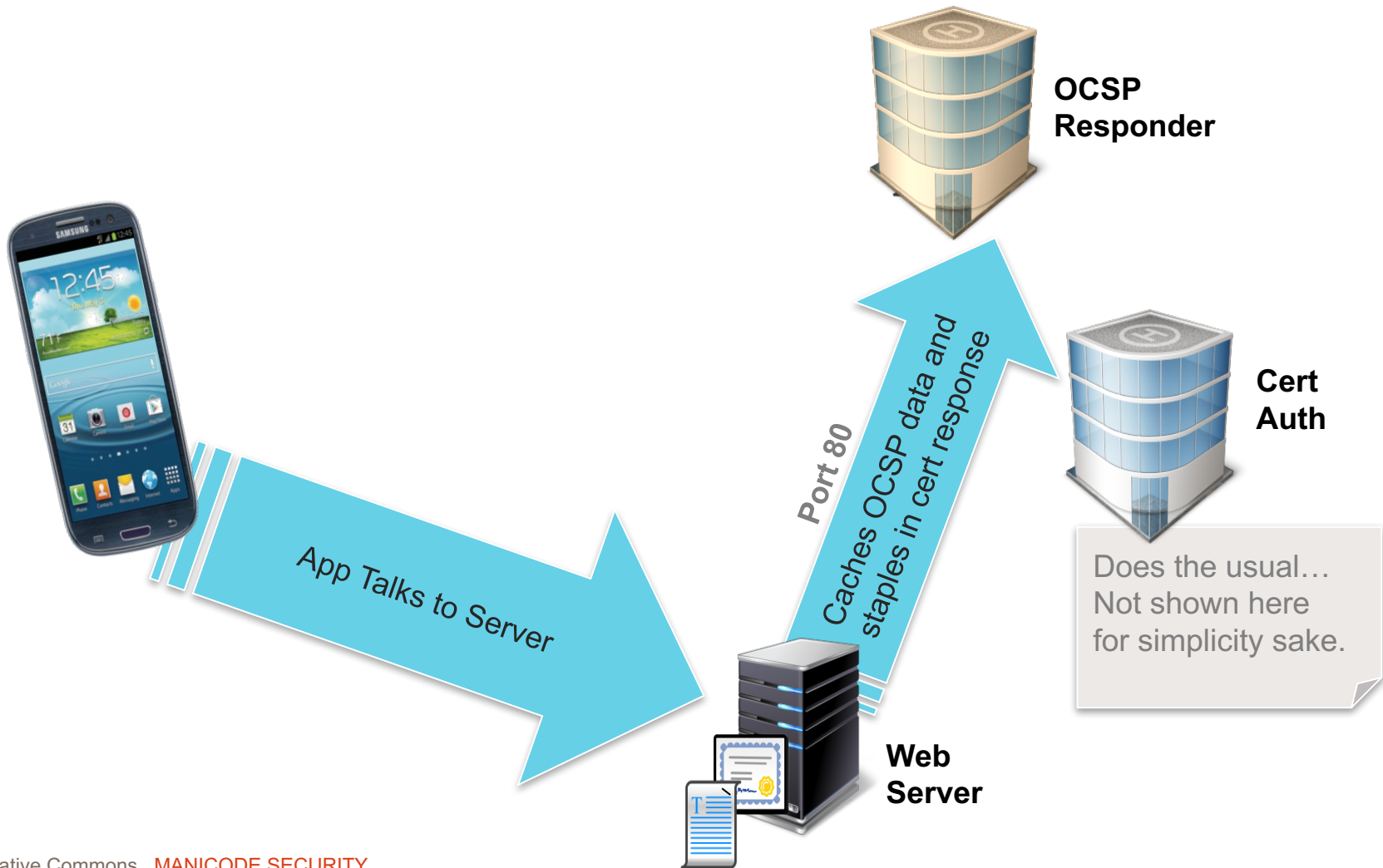
Attacks against OCSP



Attacks against OCSP



OCSP Stapling Faster, safer and more private



JEP 287

JEP 287

SHA-3 Implementation

<http://openjdk.java.net/jeps/287>

- FIPS 202 defines four new hash functions: **SHA3-224, SHA3-256, SHA3-384, and SHA3-512**. These can be implemented as new algorithms of the java.security.MessageDigest API under the standard names "SHA3-224", "SHA3-256", "SHA3-384", and "SHA3-512".
- **No new APIs are necessary**, since there are no new parameters required.
- Here is the **list of providers and the corresponding algorithm** enhancements:
 - "SUN" provider: SHA3-224, SHA3-256, SHA3-384, and SHA3-512
 - "OracleUcrypto" provider: SHA-3 digests supported by Solaris 12.0

JEP 288

TLS Benefits

Confidentiality

Spy cannot view your data

Integrity

Spy cannot change your data

Authenticity

*Server you are visiting is the right one,
backed up by the Certificate Authority System*



TLS Certificates



TLS uses X.509 Certificates

Used to authenticate the other party

NOT used to help negotiate a symmetric key (beyond authentication)

TLS certificates from certificate authorities help websites prove their authenticity. These certificates contain:

- The certificate holder
- The domain that the certificate was issued to
- ***The signature of the Certificate Authority who verified the certificate***

JEP 288

Disable SHA-1 Certificates

<http://openjdk.java.net/jeps/288>


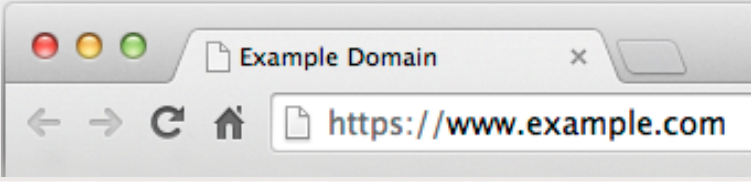
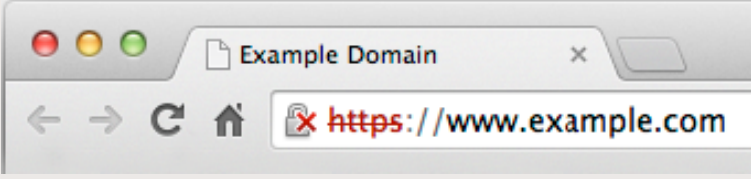
- **Disable** X.509 certificate chains with **SHA-1 based digital signatures**.
- *"SHA-1 should no longer be used to apply digital signatures to data"*
 - <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf>
 - NIST Recommendation for Key Management, Part 1: General
- *"CAs MAY continue to sign certificates to verify OCSP responses using SHA1 until January 1, 2017"*
 - v1.3 of the CA/Browser Forum Baseline Requirements (Section 7.1.3)
 - <https://cabforum.org/documents/#Baseline-Requirements>

SHA-1 and the Urgency to Move On

- In 2005, cryptographers proved that SHA-1 could be **cracked 2,000 times** faster than predicted.
- At one point **90% of websites** used SHA-1 to protect themselves from being impersonated. That number is now below 4%.
- **As long as browsers need to support SHA-1** for someone, anyone's **certificate can be forged** because browsers will not know there is a good cert that uses SHA-2

Year	Cost (In US\$)	Cost Within Reach For
2012	2,770,000	Government, large corporations
2016	700,000	Medium size institutions
2018	173,000	Organized crime
2021	43,000	University research

The Death of SHA-1 according to Google

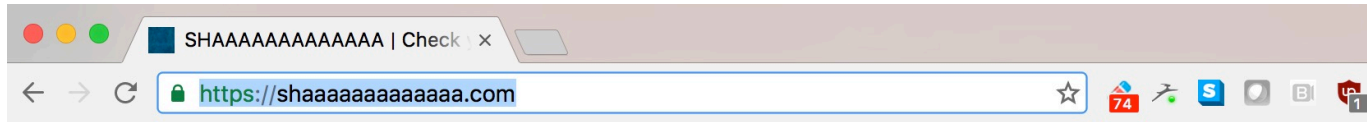
Chrome Version	Date	UI Changes	Behavior
39	Sept 2014		Certs that expire in Jan 2017 using SHA-1 or mixed content
40	Nov 2014		Certs that expire between 1 June 2016 to 31 Dec 2016 using SHA-1 in the chain
41	Q1 2016		Certs that expire on or after 1 Jan 2017

Source: <http://googleonlinesecurity.blogspot.com/2014/09/gradually-sunset-sha-1.html>

The Death of SHA-1 according to Mozilla

- Show the “Untrusted Connection” error **whenever a publically issued SHA-1 certificate issued after January 1, 2016**, is encountered in Firefox.
 - Locally installed authorities (like MITM proxy tools) are NOT subject to this rule.
- Firefox will show the "Untrusted Connection" error message for all **SHA-1-based certificates after January 2017**.
- <https://www.fxsitecompat.com/en-CA/docs/2015/sha-1-based-certificates-with-validity-period-from-2016-will-not-be-validated/>

<https://shaaaaaaaaaaaaa.com/>



SHAAAAAAAAAAAAA

Check your site for weak SHA-1 certificates. [Open source](#), by [@konklone](#).

Check above to see if a site is still using certificates that were issued using the [dangerously outdated](#) SHA-1 signature algorithm.

As of **January 1, 2016**, no publicly trusted CA is allowed to issue a SHA-1 certificate. So any new certificate you get should automatically use a SHA-2 algorithm for its signature.

However, existing SHA-1 certificates are still trusted by modern browsers and operating systems. Generally, they will be removing support for SHA-1 entirely by January 1, 2017.

Legacy clients will continue to accept SHA-1 certificates, and it is possible to have requested a certificate on December 31, 2015 valid for 39 months. So, it is possible to see SHA-1 certificates in the wild that expire in 2019.

Credits

This website is an [open source](#) project, and includes a [command line tool](#) — please [lend a hand!](#)

Thanks to [Mathias Bynens](#), [Justin Mayer](#), and [Jonny Barnes](#) for inspiration and assistance.

SHA1 Collisions No Longer Theoretical (As of 5 Days Ago)

Secure <https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html>

Google Security Blog

The latest news and insights from Google on security and safety on the Internet

Announcing the first SHA1 collision

February 23, 2017

Posted by Marc Stevens (CWI Amsterdam), Elie Bursztein (Google), Pierre Karpman (CWI Amsterdam), Ange Albertini (Google), Yarik Markov (Google), Alex Petit Bianco (Google), Clement Baisse (Google)

Cryptographic hash functions like SHA-1 are a cryptographer's swiss army knife. You'll find that hashes play a role in browser security, managing code repositories, or even just detecting duplicate files in storage. Hash functions compress large amounts of data into a small message digest. As a cryptographic requirement for wide-spread use, finding two messages that lead to the same digest should be computationally

Crypto Note

- If Crypto API's are important to you over time, please keep an eye on the **Java Cryptographic Roadmap**
- This will help you **keep an eye on planned changes** in the Oracle JRE/JDK
- <http://www.java.com/cryptoroadmap/>
- Well maintained, **last updated 48 hours ago.**

- Also, take a look at all of the **major crypto changes** that happened in **Java 8**. This is one of the largest moves in supporting modern crypto
 - NSA Suite B Cryptography
 - Strong Ephemeral Cipher Suites
 - Better Support for High Entropy Random Number Support
 - Weak algorithms Disabled by Default (Where is Dr. Depreciator?)

Please Do Not Use
Java Crypto API's

Encryption Library: Libsodium

<https://www.gitbook.com/book/jedisct1/libsodium/details>

- A high-security, cross-platform & easy-to-use crypto library.
- Modern, easy-to-use software library for encryption, decryption, signatures, password hashing and more.
- It is a portable, cross-compilable, installable & packageable fork of [NaCl](#), with a compatible API, and an extended API to improve usability even further
- Provides all of the core operations needed to build higher-level cryptographic tools.
- Sodium supports a variety of compilers and operating systems, including Windows (with MinGW or Visual Studio, x86 and x86_64), iOS and Android.
- The design choices emphasize security, and "magic constants" have clear rationales.

Moving Towards a Plugin Free Web



Web Plugin Deprecated in Java 9

- The Java Web Plugin will be deprecated in Java 9.
 - https://blogs.oracle.com/java-platform-group/entry/moving_to_a_plugin_free
- It will still be there but will go away fully in a later version.
- Start moving to Java Web Start or javapackager.

- The tech press ran with this more than what was actually said.
- There was already was a big drop of Java plugin attacks due to extensive work fixing those issues.
 - <http://blog.cobaltstrike.com/2014/01/21/obituary-java-self-signed-applet-age-1-7u51>
- FUN FACT: Stuart Marks aka *Dr. Depreciator* is a member of the JDK team.
 - <https://twitter.com/DrDeprecator>

Here's a few for the
Java Security Analysis Tool Geeks

Make Analyzing Java for Security Great Again

- A variety of JEP's In Java 9 will **help security tool vendors analyze Java** for security in more effective ways.
- **JEP-236**: Parser API for Nashorn (ECMAScript abstract syntax tree)
 - https://blogs.oracle.com/java-platform-group/entry/nashorn_the_rhino_in_the
- **JEP-190**: Pluggable Static Analyzers
 - <http://openjdk.java.net/jeps/190>
- **JEP-243**: Java-Level JVM Compiler Interface
 - <http://openjdk.java.net/jeps/243>

Java Modularity

Why Reduce JRE Attack Surface?

- **Asset:** My application / runtime.
- **Threat:** Unknown future risk.
- **Mitigation:** Remove unused pieces.
- **How:** Compact Profiles (JDK 8) and Jigsaw (JDK 9)
- **Difficulty:** Removing things that you really do need will break your program.

Server JRE Attack Surface Reduction

- **Server JRE decreases attack surface** by not including applets since **2013!**

Savings from modularization in Java 8

- JDK 8 (embedded):
- Regular JRE: About 163MB.
 - Compact 3: Remove graphics, CORBA, and sound. About 21MB.
 - Compact 2: No Kerberos and JMX monitoring. About 15MB.
 - Compact 1: No JDBC and XML. About 11MB.

JEP 200

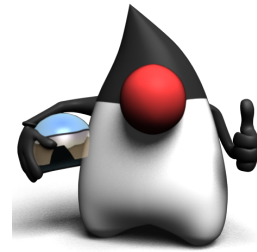
Java 9 Modularity (Jigsaw)

<http://openjdk.java.net/jeps/200>

- **Reliable configuration** to replace the classpath mechanism with a means for program components to declare dependences (**SPEAKER NOTE: WAIT FOR APPLAUSE**)
- **Strong encapsulation** to allow a component to declare which of its public types are accessible to other components
- Decreases program size
- Reduces the attack surface by removing unused features
- <http://cr.openjdk.java.net/~mr/jigsaw/ea/module-summary.html> describes 72 **epic** levels of modularity.

Annotation Enhancements (Java 8)

Annotation Security Enhancements (Java 8)



- Java 8 introduced an important changes to Annotations
 - **Type Annotations** are designed to help developers produce better code and improve the accuracy of automated code analysis
 - **Checker Framework** from University of Washington allows for Type Annotations as early as Java 4.
 - <http://types.cs.washington.edu/jsr308/>

Type Annotations

- Java 7 and earlier **only allowed annotations on definitions**.
- Java 8+ allows annotations anywhere that a **type** is used
- More info https://blogs.oracle.com/java-platform-group/entry/java_8_s_new_type

Annotation Example	Meaning
<code>@NonNull List<String></code>	A non-null list of Strings.
<code>List<@NonNull String></code>	A list of non-null Strings.
<code>@Regex String validation = "(Java JDK)[7,8]"</code>	Check at compile time that this String is a valid regular expression.
<pre>private String getInput(String parameterName){ final String retval = @Tainted request.getParameter(parameterName); return retval; }</pre>	The object assigned to retval is tainted and not for use in sensitive operations.
<pre>private void runCommand(@Untainted String... commands){ ProcessBuilder processBuilder = new ProcessBuilder(command); Process process = processBuilder.start(); }</pre>	Each command must be untainted. For example, the previously tainted String must be validated before being passed in here.

```
final String retval =  
@Tainted request.getParameter(name);  
  
@SuppressWarnings("tainting")
```

```
private void runCommand(@Untainted  
String... commands)
```

It's all about the little things



“You need to let the little things that would ordinarily bore you suddenly thrill you.”

— Andy Warhol

The Little Things are Bigger Than They Appear



- Much of the **real work that has a big impact** is seeming boring stuff
- Take a look at the work from Joe Darcy
 - Cleaned up **thousands of bad lint warnings in the JDK**
 - Done by @SuppressWarnings or actually fixing the code
 - https://blogs.oracle.com/darcy/entry/warnings_removal_a_dvice
- A little **irrational exuberance** goes a long way
- "When you spend time to remove the things that aren't worth looking at, it's easier to **see the things that are important.**" – Erik Costlow

Conclusion

pau

(pow)

(OW as in COW)

MEANING:

Finished, completed, done.

MOST COMMON USE:

Finished; all done.



A BIG DUKE THANK YOU TO...

Sean Mullan

- Technical lead and architect of Oracle's Java Security Libraries team.
- Lead of OpenJDK Security Group



What can **you** do?



- **Create JEP's** for enhancements you would like to see in the JDK.
- **Support existing JEP's** you wish to see pushed live with comments, support and code!
- Please play with the **JDK 9 early access builds** at <https://jdk9.java.net/>
- Check out the **Quality Outreach Campaign** which helps open source groups handle feedback
 - <https://wiki.openjdk.java.net/display/quality/Quality+Outreach>



A BIG DUKE THANK YOU TO...

- Output Encoding Library for XSS defense
 - Hey, if we have a JS AST class since Java 6 why not an encoding library?
 - https://www.owasp.org/index.php/OWASP_Java_Encoder_Project
- Content Security Policy library abstractions
 - The future of web security, perhaps JEE?
- Better data-driven access control, like the OACC project
 - <http://oaccframework.org/index.html>
- More automatic built-in framework security!
- SafeString class to force removal of sensitive data.

a hui ho

(ah who-EE ho)

MEANING:

Until meeting again; til later.

MOST COMMON USE:

See you later.



A hui ho, Java Ohana!

`jim@manicode.com`