Experts Are Not Infallible
The Need for Usable System Security

Matthew Smith
Usable Security and Privacy Lab, Universität Bonn
Security Fails – a lot

Online banking users get their credentials phished

Comodo Hack: 37,000 Legitimate Certificates Issued by CAs for Unqualified Names

Stuxnet Virus sets back Iran’s Nuclear Program by 2 Years. Physical damage to facilities

Sony Hack 2011: Personal Information from Approximately 24.6 Million Sony OE Accounts stolen
Online banking users get their credentials phished

Comodo Hack: 37,000 Legitimate Certificates Issued by CAs for Unqualified Names

Stuxnet Virus sets back Iran’s Nuclear Program by 2 Years. Physical damage to facilities

Sony Hack 2014: Over 100 TB stolen without anybody noticing. Including emails, medical records and unreleased scripts and films
Online banking users get their credentials phished

Comodo Hack: 37,000 Legitimate Certificates Issued by CAs for Unqualified Names

Stuxnet Virus sets back Iran’s Nuclear Program by 2 Years. Physical damage to facilities

Sony Hack 2014 No. 2: Hacker Group Lizard Squad Takes Down Sony's PlayStation Network for a couple of days
Security is hard!
Our goal is to make it easy
Solution: Usable Security and Privacy

- Three seminal papers are seen as the origin of Usable Security and Privacy research:
  - Zurko and Simon’s: “User-Centered Security”
  - Adams and Sasse’s: “Users Are Not the Enemy”
  - Whitten and Tygar’s “Why Johnny Can’t Encrypt: A Usability Evaluation of PGP 5.0”
- All argued that users should not be seen as the problem to be dealt with,
  - but that security experts need to communicate more with users, and adopt user-centered design approaches.
Usable Security Research
Example: HTTPS
HTTPS Part 1: Security Indicators
HTTPS Indicators (old)

- Microsoft IE
- Mozilla
- Firefox
- Safari
The Emperor’s New Security Indicators
An evaluation of website authentication
and the effect of role playing on usability studies
(2007)

Stuart E. Schechter
MIT Lincoln Laboratory

Rachna Dhamija
Harvard University & CommerceNet

Andy Ozment
MIT Lincoln Laboratory & University of Cambridge

Ian Fischer
Harvard University
All participants entered their passwords after HTTPS indicators were removed, including all 27 who were using their own account credentials.
HTTPS Indicators (newer)

- Made more visible
- Security “signals”
  - Green = all is well
- But things still change on a regular basis
- Effectiveness still isn’t great
Would you trust…?

Place your cursor on the SSL certificate details to view more information about the website's security.
Would you trust...?
HTTPS Part 2: Security Warnings
Firefox 2 Warning

You are being redirected to Cameo.

Please click here if you continue.

Website Certified by an Unknown Authority

Unable to verify the identity of cameo.library.cmu.edu as a trusted site.

Possible reasons for this error:
- Your browser does not recognize the Certificate Authority that issued the site's certificate.
- The site's certificate is incomplete due to a server misconfiguration.
- You are connected to a site pretending to be cameo.library.cmu.edu, possibly to obtain your confidential information.

Please notify the site's webmaster about this problem.

Before accepting this certificate, you should examine this site's certificate carefully. Are you willing to to accept this certificate for the purpose of identifying the Web site cameo.library.cmu.edu?

Examine Certificate...

- Accept this certificate permanently
- Accept this certificate temporarily for this session
- Do not accept this certificate and do not connect to this Web site
What users actually see

You are being redirected to Cameo.

Please click here if you wish to proceed.

Something happened and you need to click OK to get on with things.

Certificate mismatch security identification administration communication intercept liliputian snotweasel foxtrot omegafon.

- Technical Crap...
- More technical crap
- Hoyvin-Glavin!
- Launch photon torpedos

Adapted from Jonathan Nightingale

Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and Lorrie Faith Cranor
Carnegie Mellon University
{sunshine, egelman, hazim}@cs.cmu.edu, natri@andrew.cmu.edu, lorraine@cs.cmu.edu
In native warning conditions, no significant difference in reactions at library and bank

In new warning conditions, users more likely to heed warnings at bank than at library
On the Challenges in Usable Security Lab Studies: Lessons Learned from Replicating a Study on SSL Warnings (2011)

Andreas Sotirakopoulos
University of British Columbia
Vancouver, BC, Canada
andreass@ece.ubc.ca

Kirstie Hawkey
Dalhousie University
Halifax, NS, Canada
hawkey@cs.dal.ca

Konstantin Beznosov
University of British Columbia
Vancouver, BC, Canada
beznosov@ece.ubc.ca
Effect of Browser/Warning on Behavior

- No statistically significant differences were observed between the various conditions in the study.
- There was also no significant differences between participants who were randomly assigned IE7 and native IE7 users

<table>
<thead>
<tr>
<th></th>
<th>FF3</th>
<th>FF3 custom</th>
<th>IE7</th>
<th>IE7 custom</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMU</td>
<td>11 (55%)</td>
<td>N/A</td>
<td>18 (90%)</td>
<td>9 (45%)</td>
</tr>
<tr>
<td>UBC</td>
<td>16 (80%)</td>
<td>17 (85%)</td>
<td>N: 14 (70%)</td>
<td>R: 15 (75%)</td>
</tr>
</tbody>
</table>

Table 1: Comparison of results between the two studies for participants who chose to ignore the SSL warning at the bank sign-in web site. (N: Participants using their normal browser, R: participants are assigned to browsers randomly)

<table>
<thead>
<tr>
<th></th>
<th>FF3</th>
<th>FF3 custom</th>
<th>IE7</th>
<th>IE7 custom</th>
</tr>
</thead>
<tbody>
<tr>
<td>UBC</td>
<td>14 (70%)</td>
<td>16 (80%)</td>
<td>N: 16 (80%)</td>
<td>R: 17 (85%)</td>
</tr>
</tbody>
</table>

Table 2: Results for participants who chose to ignore the SSL warning at the hotmail sign up web site. (N: Participants using their normal browser, R: participants are assigned to browsers randomly)
Secure Connection Failed

www.vedetta.com uses an invalid security certificate.

The certificate is not trusted because it is self-signed.

(Error code: sec_error_ca_cert_invalid)

- This could be a problem with the server's configuration trying to impersonate the server.
- If you have connected to this server successfully in the past, you can try again later.

Or you can add an exception.

There is a problem with this website's security certificate.

The security certificate presented by this website was not issued by a trusted certificate authority.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

The site's security certificate is not trusted!

You attempted to reach vps1234.inmotionhosting.com, but the server presented a certificate issued by an entity that is not trusted by your computer's operating system. This may mean that the server has generated its own security credentials, which Google Chrome cannot rely on for identity information, or an attacker may be trying to intercept your communications.

You should not proceed, especially if you have never seen this warning before for this site.

Proceed anyway Back to safety

Help me understand
Participatory Design for Security-Related User Interfaces
(2015)

Susanne Weber, Marian Harbach
Usable Security and Privacy Lab
Gottfried Wilhelm Leibniz Universität Hannover, Germany
{weber,harbach}@usecap.uni-hannover.de

Matthew Smith
Usable Security and Privacy Lab
Rheinische Friedrich-Wilhelms-Universität Bonn, Germany
smith@cs.uni-bonn.de
- Use participatory design
  - to have users design their own warnings messages
Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness (2013)

Devdatta Akhawe  
University of California, Berkeley*  
devdatta@cs.berkeley.edu

Adrienne Porter Felt  
Google, Inc.  
felt@google.com
Real World Analysis

- Studied the click-through rate for malware and HTTPS warnings
  - **Malware**
    - Firefox 7.2%
    - Chrome 23.2%
  - **Phishing**
    - Firefox 9.1%
    - Chrome 18.0%
  - **HTTPS**
    - Firefox 33.0%
    - Chrome 70.2%
Here's My Cert, So Trust Me, Maybe?  
Understanding TLS Errors on the Web  

(2013)

Devdatta Akhawe  
University of California, Berkeley  
devdatta@cs.berkeley.edu

Bernhard Amann  
International Computer Science Institute, Berkeley  
bernhard@icir.org

Matthias Vallentin  
University of California, Berkeley  
vallentin@cs.berkeley.edu

Robin Sommer  
International Computer Science Institute, Berkeley  
robin@icir.org
Real World Analysis

- Studied TLS activity of more than 300,000 users
  - collected certificates passively at egress points of ten network sites
  - over a nine-month period
  - validated certificate chains using browser logic locally
  - 98.46% of the filtered connections validate correctly, implying a false warning rate of 1.54%

- In a scenario with a hypothetical MITMA chance of 1 in 1,000,000
  - 1,000,000 connections would produce 15,401 warnings
  - out of which 15,400 would be false warnings
There’s more…

- End-users are only a small part of the HTTPS ecosystem.
- Administrators are responsible for (mis)configuration web-servers.
- Developers are responsible for (mis)using HTTPS in their applications.
- Alternative PKI designs might make things better – they might also make them worse...
Chapter 1: Administrators
Scope of the problem

- We used HTTPS certificates collected by Google's web-crawler
  - Period of 12 months
  - ~55.7 million different hosts
  - ~4.49 million different X.509 certificates
  - We extracted all certificates that did not validate correctly based on the Firefox browser logic

<table>
<thead>
<tr>
<th>Error Type</th>
<th>#Certificates</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid</td>
<td>3,876,497</td>
<td>86.38%</td>
</tr>
<tr>
<td>Self-Signed</td>
<td>89,981</td>
<td>2.0%</td>
</tr>
<tr>
<td>Expired</td>
<td>309,350</td>
<td>6.89%</td>
</tr>
<tr>
<td>Hostname Mismatch</td>
<td>146,941</td>
<td>3.27%</td>
</tr>
<tr>
<td>Unknown Issuer</td>
<td>64,694</td>
<td>1.44%</td>
</tr>
</tbody>
</table>
Solutions?

- So what should we do to help the administrators?
  - Create better configuration tools?
  - Reduce the complexity of the entire system?
Find out where the problems lie

- ~4.49 million “bad” certificates
  - We picked a random sample of 50,000
  - Pruned non-current certs down to 46,145
  - And contacted the admins
- We sent 40,473 emails to webmaster@domain.com
- and 5,672 to addresses embedded in the certs.
- Of the 46,145 emails we sent
  - 37,596 could not be delivered to the intended recipient,
  - leaving us with 8,549 successfully delivered surveys
  - 755 complete responses to our survey (~8%)
Find out where the problems lie

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Deliberate</th>
<th>Misconfiguration</th>
<th>Not Actively Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Signed</td>
<td>90</td>
<td>45</td>
<td>20</td>
</tr>
<tr>
<td>Expired</td>
<td>74</td>
<td>38</td>
<td>16</td>
</tr>
<tr>
<td>Hostname Mismatch</td>
<td>82</td>
<td>50</td>
<td>51</td>
</tr>
<tr>
<td>Unknown Issuer</td>
<td>84</td>
<td>32</td>
<td>14</td>
</tr>
<tr>
<td>Total</td>
<td>330</td>
<td>165</td>
<td>101</td>
</tr>
</tbody>
</table>

- **Reasons given in survey**
  - ~21% sub-domains/virtual hosts/redirects
  - ~16% to difficult
  - ~16% for a small group of users
  - ~7% NSA, PRISM & co.
  - ~5% untrusted CA
  - ~3% default configuration
  - ~2% mistake
  - ...

- **Risk perception**
  - ~70% very small
  - ~3% very high
  - ~11% didn’t know there were warnings
Administrators’ wish list

- Lower Price for CA-signed certificates
  - Price is perceived too high for little effort on the CA’s side
  - Free CA-signed certificates
  - Cheaper wildcard certificates

- Allow CACert
  - More trust in CACert’s web of trust model

- Better Support for Non-Validating Certificates
  - Support for trust-on-first-use, Pinning, TACK

- Better Tool Support
  - OpenSSL command line tool too complicated
  - Server configuration cumbersome, especially for v-hosts
  - Auto-Update Reminder
  - Notification of problems
Chapter 2: Developers
Trust me I’m an Engineer
The default Android HTTPS API implements correct certificate validation.

What could possibly go wrong?
HTTPS Usage on Android and iOS

- A server needs a certificate that was signed by a trusted Certificate Authority
  - (~130 pre-installed CAs)
- For non-trusted certificates a custom workaround is needed
- Error handling requires custom code
- Additional security measures such as pinning or Certificate Transparency require custom code
Q: I am getting an error of „javax.net.ssl.SSLException: Not trusted server certificate“.

[...]

I have spent 40 hours researching and trying to figure out a workaround for this issue.

A: Look at this tutorial
SSL Static Code Analysis

- Analysis of 13,500 popular, free apps from Google’s Play Market
  - 92.8% of the apps use the Internet permission
  - 91.7% of networking API calls are HTTP(S) related
  - 0.8% exclusively HTTPS URLs
  - 46.2% mix HTTP and HTTPS

- 17.28% of all apps that use HTTPS include code that fails in SSL certificate validation
  - 1070 include critical code
  - 790 accept all certificates
  - 284 accept all hostnames
Manual App Testing Results

- Cherry-picked 100 apps
  - 21 apps trust all certificates
  - 20 apps accept all hostnames

- Captured credentials for:
  - American Express, Diners Club, Paypal, bank accounts, Facebook, Twitter, Google, Yahoo, Microsoft Live ID, Box, WordPress, remote control servers, arbitrary email accounts, and IBM Sametime, among others.
Tusting all Certificates

- Correct HTTPS certificate validation is easy
  - Only a (costly) trusted CA signed certificate required
- What some Apps do:

```java
// Create a trust manager that does not validate certificate chains
TrustManager[] trustAllCerts = new TrustManager[]{
  new X509TrustManager()
}

public java.security.cert.X509Certificate[] getAcceptedIssuers() {
  return null;
}

public void checkClientTrusted(X509Certificate[] chain, String authType) throws CertificateException {
  // do nothing
}

public void checkServerTrusted(X509Certificate[] chain, String authType) throws CertificateException {
  // do nothing
}
```
Allowing all Hostnames

- What other Apps do:
  - Check CA signature, but allow mallory.com for google.com

```
KeyStore trustStore = KeyStore.getInstance(KeyStore.getDefaultType());
trustStore.load(null, null);

SSLSocketFactory sf = new MySSLSocketFactory(trustStore);
sf.setHostnameVerifier(SSLSocketFactory.ALLOW_ALL_HOSTNAME_VERIFIER);
```
Anti-Virus Example

- **ZonerAV**
  - Anti-Virus app for Android
  - Awarded best free anti-virus app for Android by av-test.org

- Virus signature updates via HTTPS GET
  - The good thing: It uses SSL
  - Unfortunately: The wrong way

```java
static final HostnameVerifier DO_NOT_VERIFY = new HostnameVerifier()
{
    public boolean verify(String paramString, SSLSession paramSSLSession)
    {
        return true;
    }
};
```

- Zoner fixed the bug immediately!
- 22 different TrustManager implementations

- and all turn effective certificate validation off
How Do (Good) Apps React to MITMAs?

- Technically they do not endanger the user
- However they suffer from serious usability problems

Flickr

Facebook

Login Failed

Sorry, login Failed to reach Facebook servers. Please check your network connection or try again later.
( hostname in certificate didn’t match: <api.facebook.com> != <*
.mallory.com> [javax.net.ssl.SSLEexception])
“It’s all the developers’ fault!”
Solutions?

So what should we do to help the developers?

Security experts need to communicate more with developers, and adopt developer-centered design approaches.
Talking To Developers

- Finding broken HTTPS in Android and iOS apps is good...
  ...knowing what the root causes are is even better

- We contacted 80 developers of broken apps
  - informed them ✓
  - offered further assistance ✓
  - asked them for an interview ?

- 15 developers agreed ✓
“This app was one of our first mobile apps and when we noticed that there were problems with the SSL certificate, we just implemented the first working solution we found on the Internet.”
“We use self-signed certificates for testing purposes and the easiest way to make them working is to remove certificate validation. Somehow we must have forgotten to remove that code again when we released our app.”
“ [...] When I used Wireshark to look at the traffic, Wireshark said that this is a proper SSL protected data stream and I could not see any cleartext information when I manually inspected the packets. So I really cannot see what the problem is here.”
“The app accepts all SSL certificates because some users wanted to connect to their blogs with self-signed certs and […] because Android does not provide an easy-to-use SSL certificate warning message, it was a lot easier to simply accept all self-signed certificates.”
Developer Survey Summary

- **Self-Signed Certificates – Development.**
  - Developers commonly wish to use self-signed certificates for testing purposes and hence want to turn off certificate validation during testing.

- **Self-Signed Certificates – Production.**
  - A few developers wanted to use self-signed certificates in their production app for cost and effort reasons.

- **Code Complexity.**
  - Developers described the code-level customization features of HTTPS as too complex and requiring too much effort.

- **Certificate Pinning / Trusted Roots.**
  - Developers liked the idea of having an easy way to limit the number of trusted certificates and/or certificate authorities.

- **Global Warning Message.**
  - Developers requested global HTTPS warning messages since they described building their own warning messages as too challenging.
Chapter 3: System Design
Problems with the infrastructure

- Approximately 100-200 trusted root CAs in
  - Firefox, Chrome, IE Explorer, Windows, Mac OS, Linux
  - Extended to ~650 via CA hierarchies
  - EFF Map of these organizations
- SSL / HTTPS only as strong as the weakest link
  - Weak (email-based) authentication with many CAs
  - Targeted attacks against CAs - a real world threat
  - No CA scopes

https://www.eff.org/observatory
Up-and-coming PKIs

- DANE
- Certificate Transparency
- ARPKI (Perrig et. al – next door at SENT)

All promise better security
- All are more complex
- How will developers cope?
- How will administrators cope?
- How will users cope?
So what do we do now?
Administrators and developers are humans too
  - We should be supporting them just as much – if not more – than end-users
  - Especially during systems design

Short term goals:
  - Talk with administrators and developers
  - Find out where the problems lie
  - Extract and implement wish-lists

Long term goal: Usable Systems Security
  - Design entire IT-Ecosystem with administrators and developers in mind
Users Are Not The Enemy

Experts Are Not The Enemy (either)

Let’s give them our support