Developers are not the Enemy!

Matthew Smith

Usable Security and Privacy Lab, Universität Bonn, Fraunhofer FKIE
I’d like to thank

- Alena Naiakshina
- Anastasia Danilova
- Christian Tiefenau
- Dr. Emanuel von Zezschwitz
- Dr. Henning Perl
- Karoline Busse

- Dr. Khaled Yakdan
- Dr. Marian Harbach
- Dr. Michael Brenner
- Dr. Sascha Fahl
- Sergej Dechand
- Yasemin Acar
Security is hard!
The goal of Usable Security is to make it easy!
Users are not the enemy!
Adams & Sasse’99
Overview

- Story 1: HTTPS
- Story 2: Passwords
- Story 3: Malware Analysis
- or Frontiers of Usable Security Methodology
Story 1

HTTPS/TLS
HTTPS Part 1: Security Indicators
HTTPS Indicators (old)

- Microsoft IE
- Mozilla
- Firefox
- Safari
User Study

- Schechter et al. conducted a lab study with 67 participants*
  - Complete an online banking task
  - Three groups
    - Role playing
    - Role playing with hint to behave securely
    - Users’ real online banking account

- Removed HTTPS security indicator
  - 100% entered their credentials
  - Even those using their real online banking credentials

* Schechter et al., The Emperor’s New Security Indicators. An evaluation of website authentication and the effect of role playing on usability studies, IEEE Security and Privacy 2007
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
HTTPS Part 2: Security Warnings
Firefox 2 Warning

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 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

OK Cancel
What users actually see

Adapted from Jonathan Nightingale

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

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

Technical Crap …

- More technical crap
- Hoyvin-Glayvin!
- Launch photon torpedos

OK Cancel
Idea: Ask users a question

Multi-page warning

Secure Connection Failed

The website responding to your request failed to provide verifiable identification.

What type of website are you trying to reach?
- Bank or other financial institution
- Online store or other e-commerce website
- Other
- I don't know

Continue

You are seeing this warning because the response contained a self-signed certificate.

Sunshine et. al. Crying Wolf, Usenix Security 2009
Newer HTTPS Warnings

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, and you can try again later.

Or you can add an exception.

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
HTTPS: Administrator Mistakes

Akhawe et al: Server misconfigurations lead to

15.400 false positive per 1 true positive certificate warnings

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, or it could be someone trying to impersonate the server.
- If you have connected to this server successfully in the past, the error may be temporary, and you can try again later.

Or you can add an exception.

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, or it could be someone trying to impersonate the server.
- If you have connected to this server successfully in the past, the error may be temporary, and you can try again later.

Or you can add an exception.

\(^1\) Akhawe et al. (WWW ’13)
HTTPS: Administrator Mistakes

Akhawe et al: Server misconfigurations lead to

15.400 false positive per 1 true positive certificate warnings

15.400 to 1 odds shouldn’t be dealt with on the end-user level but on the system level

1 Akhawe et al. (WWW ’13)
Developers are not the enemy!

Green & Smith IEEE S&P Magazine’16
End-users are only a small part of the HTTPS ecosystem.

Administrators are responsible for (mis)configuration of 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...
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>%</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>
USEC Studies with Administrators

- ~610k 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><strong>Total</strong></td>
<td><strong>330</strong></td>
<td><strong>165</strong></td>
<td><strong>101</strong></td>
</tr>
</tbody>
</table>

- **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, etc.

- Better Tool Support
  - OpenSSL command line tool too complicated
  - Server configuration cumbersome, especially for v-hosts
  - Auto-Update Reminder
  - Notification of problems

Published at ACM AsiaCCS’14
Let's Encrypt
Study with 32 computer science students

<table>
<thead>
<tr>
<th>CA</th>
<th>Success</th>
<th>Fail</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-C</td>
<td>28</td>
<td>4</td>
</tr>
<tr>
<td>CA-T</td>
<td>16</td>
<td>16</td>
</tr>
</tbody>
</table>
Developers
The default Android HTTPS API implements correct certificate validation.

What could possibly go wrong?
Trust me! I know what I’m doing!

Usable Security and Privacy Lab – Universität Bonn
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

stackoverflow.com
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.
T outing 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
    }
} };
```
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!
Common: Blaming Developers

“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 ✓
A New Approach to TLS on Android

Central TLS service for Android
- Force TLS validation
- Supports self-signed certificates
- Certificate Pinning
- Standardised user interaction
- Alternate Cert validation strategies

More details can be found in our CSS paper: Rethinking ssl development in an appified world
CA Infrastructure
Problems with the CA 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 (Google)
- ARPKI/SCION (ETH Zürich)

All offer better security
- All are more complex
- How will developers cope?
- How will administrators cope?
- How will users cope?
Story 2

Passwords
Password Advice

- Passwords are still a mainstay of modern security
  - and a very common cause of security problems

- Common password advice
  - make it long and random
  - use special characters
  - don’t write it down
  - change it often
  - don’t re-use across services

  **good technical advice**

  **bad usability advice**

- Password problems lead to
  - lost productivity
  - recovery cost
  - frustrated users who try and circumvent system
Ur et al. How Does Your Password Measure Up? The Effect of Strength Meters on Password Creation, USENIX Security‘12
Shay et al. Correct horse battery staple: Exploring the usability of system-assigned passphrases, SOUPS‘12
Yahoo says all 3 billion user accounts were impacted by 2013 security breach
Study Design

Role-playing scenario
- Social networking platform of the University of Bonn.
- Code for user registration and user authentication.

Two Groups
- Secure password storage
- API usability.
Frameworks

JSF
- Manual level of support
- No built-in functions for hashing

Spring
- Opt-in support
- Built-in functions for hashing
### 4 Conditions

<table>
<thead>
<tr>
<th>Framework</th>
<th>Level of Support</th>
<th>(Non-)Priming</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 JSF</td>
<td>manual</td>
<td>Priming</td>
<td>JP</td>
</tr>
<tr>
<td>2 JSF</td>
<td>manual</td>
<td>Non-Priming</td>
<td>JN</td>
</tr>
<tr>
<td>3 Spring</td>
<td>opt-in</td>
<td>Priming</td>
<td>SP</td>
</tr>
<tr>
<td>4 Spring</td>
<td>opt-in</td>
<td>Non-Priming</td>
<td>SN</td>
</tr>
</tbody>
</table>
Participant Demographics

- 20 participants
  - 3 female, 17 male
- Students: 18 Computer Science, 2 Media Informatics
  - 7 BSc, 13 MSc Students
- Mean age 24 years
  - Range: 19-27 years
- 8 hours to complete study
- Post study interview
- The end-user password is salted (+1) and hashed (+1).

- The derived length of the hash is at least 160 bits long (+1).

- The iteration count for key stretching is
  - at least 1 000 (+0.5) or 10 000 (+1) for PBKDF2 and
  - at least $2^{10} = 1 024$ for bcrypt (+1).

- A memory-hard hashing function is used (+1).

- The salt value is generated randomly (+1).

- The salt is at least 32 bits in length (+1).
Results
Non-Primed Group
How many participants had a basic background knowledge of hashing?
How many participants have managed to store the user passwords securely?
Umm, actually literally when I was in the project
I didn’t feel much like that
it was related to security. (JN5)
I would ask my supervisor about it. [...] There is definitely another person that understood these kinds of things. (JN3)
I assumed that the connection will be a secure connection like with an HTTPS connection, so everything should come encrypted. (JN1)
Developers are not the enemy!
Primed Group
How many participants had a basic background knowledge of hashing?
9/10
How many participants have managed to store the user passwords securely?
### Primed Group

- **7/10** included at least some security
- **4/10** participants received 6 points.
- **3/4** were in the Spring group

<table>
<thead>
<tr>
<th>Hash Function</th>
<th>Sec</th>
<th>Func</th>
</tr>
</thead>
<tbody>
<tr>
<td>JP1</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>JP2</td>
<td>PBKDF2(SHA256)</td>
<td>5.5</td>
</tr>
<tr>
<td>JP3</td>
<td>SHA256</td>
<td>2</td>
</tr>
<tr>
<td>JP4</td>
<td>PBKDF2(SHA1)</td>
<td>6</td>
</tr>
<tr>
<td>JP5</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>SP1</td>
<td>BCrypt</td>
<td>6</td>
</tr>
<tr>
<td>SP2</td>
<td>MD5</td>
<td>1</td>
</tr>
<tr>
<td>SP3</td>
<td>BCrypt</td>
<td>6</td>
</tr>
<tr>
<td>SP4</td>
<td>BCrypt</td>
<td>6</td>
</tr>
<tr>
<td>SP5</td>
<td>-</td>
<td>0</td>
</tr>
</tbody>
</table>
Story 3
Security Analysis
Malware Analysis

Source code

```c
int f(int a){
    int i = 0;
    for(; i < a ; i++)
        ...
}
```

Decompiled code

```c
int f(int arg){
    int var = 0;
    while(var < arg)
        ...
    var = var + 1;
}
```

High-level abstractions are lost

Binary code

0101010101010101010100
0101010101010101010100
0101010101010101010100
0101010101010101010100
0101010101010101010100
0101010101010101010100
0101010101010101010100
0101010101010101010100

Recovered abstractions
Decompiling a P2P Zeus sample with Hex-Rays

- 1,571 goto for 49,514 LoC
- 1 goto for each 32 LoC
- DREAM Decompiler
  - Pattern independent CFG structuring
  - No more gotos!
  - Most compact code
Usability Problems

- Complex expressions
  - (too) Many variables
  - Code in loop statements
  - Pointer expressions

- Control Flow
  - Duplicate/inlined code
  - Complex loop structure

- No Semantics
  - Special API function
  - Magic number of file types
Simda Malware

DREAM++

dwSeed = 0x45AE94B2

results in?
User Study

- We recruited 21 students who successfully took part in our malware bootcamp over the last 5 years and
- 9 malware analysis professionals

- $3 \times 2 \times 2$ mixed-subjects design
- 3 decompilers (within-subjects)
  - Hex-Rays
  - DREAM
  - DREAM++
- 2 levels of experience (between-subject)
  - Students and Professionals
- 2 groups of malware analysis tasks (split-plot)
  - 3 medium and 3 hard task (within-subjects)
Simda Malware

DREAM++

dwSeed = 0x45AE94B2
results in?
v17 = "cihunemyror"
v3 = "cihunemyror"
“The code mostly looks like a straightforward C translation of machine code; besides a general sense about what is going on, I think I'd rather just see the assembly.” - DREAM

“This code looks like it was written by a human, even if many of the variable names are quite generic. But just the named index variable makes the code much easier to read!” – DREAM++
## Results

<table>
<thead>
<tr>
<th>Decompiler</th>
<th>Avg. Score</th>
<th>p</th>
<th>Pass</th>
<th>Fail</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Students</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DREAM++</td>
<td>70.24</td>
<td></td>
<td>30</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>DREAM</td>
<td>50.83</td>
<td>0.002</td>
<td>16</td>
<td>26</td>
<td>0.002</td>
</tr>
<tr>
<td>Hex-Rays</td>
<td>37.86</td>
<td>&lt;0.001</td>
<td>11</td>
<td>31</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td><strong>Experts</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DREAM++</td>
<td>84.72</td>
<td></td>
<td>15</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>DREAM</td>
<td>79.17</td>
<td>0.234</td>
<td>15</td>
<td>3</td>
<td>0.570</td>
</tr>
<tr>
<td>Hex-Rays</td>
<td>61.39</td>
<td>0.086</td>
<td>9</td>
<td>9</td>
<td>0.076</td>
</tr>
</tbody>
</table>
Ranking Results

[Diagram showing box plots for expert and student categories, comparing Hex-Rays, DREAM, and DREAM++.]
Frontiers of Usable Security

ERC

USECFrontiers
Multi-dimensional Problem

Fundamental Objectives

- F1.1 Incentives
- F1.2 Task Design
- F1.3 Type of Participant
- F1.4 Priming/Deception
- F1.5 Self-reporting
- F1.6 Type of Study

- F2 Security APIs
- F3 Risk Perception & Mental Models
Multi-dimensional Problem

Fundamental Objectives
- F1.1 Incentives
- F1.2 Task Design
- **F1.3 Type of Participant**
  - Students
  - Online Freelancers
  - Developers/Admins
- F1.4 Priming/Deception
- F1.5 Self-reporting
- F1.6 Type of Study
Multi-dimensional Problem

Fundamental Objectives

- F1.1 Incentives
- F1.2 Task Design
- F1.3 Type of Participant
- F1.4 Priming/Deception
  - Priming/No Deception
  - Non-priming/Deception
- F1.5 Self-reporting
- F1.6 Type of Study

Priming

Type of Participant

Priming/No Deception
Non-priming/Deception
Multi-dimensional Problem

Fundamental Objectives

- F1.1 Incentives
- F1.2 Task Design
- F1.3 Type of Participant
- F1.4 Priming/Deception
- F1.5 Self-reporting
- F1.6 Type of Study
  - Qual/Quant
  - Lab
  - Online
  - Field
  - Within/Between
  - Interviews
  - Focus Groups

Research Goal

Type of Participant

Priming

Online, Lab, Interviews, Focus Groups
Novel Approach

Primary Study

Randomized control trial:
Create a backend service including user accounts

Control condition
- JCA

Treatment condition
- Spring

Meta-Study

Frontiers of Usable Security – Matthew Smith, University of Bonn & Research Center L3S
Novel Approach

Primary Study

Randomized control trial:
Create a backend service including user accounts

Control condition
- JCA

Treatment condition
- Spring

Meta-Study

F1.3 Participants
F1.6 Type of Study

Frontiers of Usable Security – Matthew Smith, University of Bonn & Research Center L3S
Novel Approach

Primary Study

Randomized control trial:
Create a backend service including user accounts

Control condition
- JCA

Treatment condition
- Spring

Meta-Study

F1.4 Priming/Deception

- Priming
- Non-Priming
- Lab
- Online
- Students
- Online Freelancers
- Developers/Admins
We are here

Usable Security for Administrators and Developers holds huge potential!
More Interesting Work on DevUSEC

Yasemin Acar & Sascha Fahl @ Uni-Hannover

Developers are not the enemy!

Green & Smith IEEE S&P Magazine’16