Web Security in the Real World

Steven M. Bellovin
Chief Technologist, FTC
Security is Built on Trust

- We trust the OS
- We trust the browser
- We trust the cryptography
- We trust the destination website
“We trust the cryptography”?

- “These practices and controls include… encrypting the transfer of personal information … via … (SSL)”
- Similar statements appear in many websites’ security and privacy policies
- People equate security with privacy and trustworthiness, and conflate both with encryption
Verifying encryption, per one web site:

- A key or a lock at the bottom
  - “Unbroken” or “locked”
  - Not true on my browsers…

- Or the “color of the address bar”
  - Green—or blue—or yellow
  - Also not true for me…

What’s a user to do?
What’s a Certificate?

- Very few users know what a certificate is
- Fewer care
- Virtually none know or care about trust anchors
- Can ordinary users employ advanced techniques?
What’s a Certificate?

- A certificate is a digitally-signed binding between a name and a public key.
- Who can sign such things? Anyone—but generally, it’s done by a Certificate Authority.
- All modern browsers trust hundreds of CAs—browser vendors have their own policies for deciding whom to list.
Who Checks Certificates?

- Who here knows what a certificate is?
- Who has ever checked a web site’s certificate?
- Who always checks?
- Who verifies that the certificate authority is “reasonable”?
- Who knows *anything* about most CAs that your browser happens to trust?
Things Change (Example)

- CyberTrust was originally part of GTE
- In 2000, it was bought by Baltimore Technologies, an Irish company
- In 2003, CA operation sold to Betrusted
- In 2004, Betrusted merged with TruSecure; the new company moved to Virginia and named itself CyberTrust.
- In 2007, CyberTrust was bought by Verizon, which had earlier bought GTE—they’re back together…
What’s the Issue?

- From a security perspective, too many root CAs in every browser
- Any one of these can issue certificates to any web site
- A compromise of (or misbehavior by) any one of these allows for spoofing of any site
- This has already happened
The DigiNotar Case

- A Dutch CA
- Their CA site was hacked in July, 2011
- Fraudulent certificates were issued for Google, Yahoo, Mozilla, Tor, Wordpress, and hundreds of others
- The Google certificate, at least, was used in Iran
- DigiNotar was delisted by all major browser vendors...
Who issues certificates to government web sites?

- NIST?
- The NSA?
- Nope…
But why does Verisign have to attest to its identity?
Is the FBI Feuding with the CIA?

<table>
<thead>
<tr>
<th>General</th>
<th>Details</th>
</tr>
</thead>
</table>

This certificate has been verified for the following uses:

**SSL Server Certificate**

<table>
<thead>
<tr>
<th>Issued To</th>
<th>*fbi.gov</th>
<th>*fbi.gov</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Name (CN)</td>
<td>*fbi.gov</td>
<td>*fbi.gov</td>
</tr>
<tr>
<td>Organization (O)</td>
<td>GlobalSign</td>
<td>GlobalSign</td>
</tr>
<tr>
<td>Organizational Unit (OU)</td>
<td>Domain Control Val</td>
<td>Domain Control Val</td>
</tr>
</tbody>
</table>

**Issued By**

<table>
<thead>
<tr>
<th>Common Name (CN)</th>
<th>Go Daddy Secure Certification Authority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization (O)</td>
<td>GoDaddy.com, Inc.</td>
</tr>
<tr>
<td>Organizational Unit (OU)</td>
<td><a href="http://certificates.godaddy.com/repository">http://certificates.godaddy.com/repository</a></td>
</tr>
</tbody>
</table>

**Validity**

<table>
<thead>
<tr>
<th>Issued On</th>
<th>08/20/2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expires On</td>
<td>08/19/2013</td>
</tr>
</tbody>
</table>

**Fingerprints**

|------------------|--------------------------------------------------------|
An Intelligence Agency Playing Games?
The Labor Department Gets It

OSHA is part of the Labor Department; its address rightly appears in the dol.gov certificate (but how did the CA verify the relationship?)
Congress Divided
Using CertPatrol

Remember that famous comment line from early Unix source code? “You are not expected to understand this.”
/*
 * If the new process paused because it was
 * swapped out, set the stack level to the last call
 * to savu(u_ssav). This means that the return
 * which is executed immediately after the call to aretu
 * actually returns from the last routine which did
 * the savu.
 *
 * You are not expected to understand this.
 */

Fundamental Requirements

- We need a solution that is (considerably more) secure than what we have today
- It must not involve new user interactions
- It must not generate (yet more) incomprehensible error messages
- It MUST NOT present new dialog boxes with text that will be ignored but still have a button to click labeled “OK”
Other Issues

- Existing businesses and business models
- Existing operational practices
- Self-signed certificates
- Enterprise firewalls and IDSs that do certificate-spoofing, to monitor employee conversations
- Non-web (e.g., app) uses of certificates by systems
Let’s look at DNSSEC, an element of one proposed solution (DANE)

- DANE: “DNS Authentication of Named Entities”—certificates in the DNS
- I’ll also mention “certificate transparency”; see http://www.certificate-transparency.org/

Comcast and Google do DNSSEC evaluation

What does the user see?
The DNSSEC Validator Extension

- The primary indicator is a key icon
- You only get the box if you click on the key
- Who will notice or understand?
Transparent Checking

- It protects the user from bad stuff
- It gives no hint what the real issue is
- Who will help the helpdesk?
It’s Not Just the User Interface

- Most ISPs don’t check DNSSEC
- Neither do most hosts
- Note that DNSSEC depends on upstream DNS registrars
- We’re missing APIs and secure, deployable over-the-wire signaling protocols
- RFC 6698 (TLSA): “DNSSEC validation is best performed on-host, even when a secure path to an external validator is available.”
Whose Behavior Must Change?

- **Users?**
  - Billions of users; no opportunity for training

- **Web sites?**
  - Tens (hundreds?) of millions; retraining sysadmins and webmasters is slow

- **ISPs?**
  - What’s in it for them? N.B.: see the comment on helpdesks

- **Browsers and operating systems?**
  - What is the upgrade rate?
DNSSEC Trust Models

- Sites have to trust a chain of registrars and registries up to the DNS root
- Many of these organizations are not accustomed to handling keys
- Effectively, though, they’re CAs; however, there’s only one root rather than many
  - Better for security, but no opportunity for vendors to compete on basis of security
What About Business Models?

- Some proposals (e.g., certificate transparency) require new parties
  - Who pays them?
  - Who pays them to *scale*?
    - What if there’s a DDoS attack on such parties?
  - Who pays them to respect privacy?
What About Governments?

- Many governments, hence many policies about trust, liability, content, privacy, etc.
- What ability should governments have to intervene in the trust model?
- Which governments should have that right, and under what conditions?
Who should vouch for whom?

• In theory, an organization should vouch for its own subunits

• This is rarely done—is it that hard for an organization to run its own sub-CA?

• On the other hand, how do outside CAs verify not just the real organizational structure of the parent enterprise, but the authorization?
What Changes Rapidly?

- (Desktop) operating systems rarely change; machines are replaced instead
  - Requirements for better hardware cap the upgrade rate
- OS-linked browsers (IE, Safari) are upgraded somewhat slowly (stats source: clicky.com)
  - Probably faster than the underlying OS
- Independent browsers (Firefox, Chrome) are upgraded more rapidly
Are the Browsers the Leverage Point?

- They’re effectively the trust anchor anyway; users trust what their browser tells them to trust
- They change more rapidly than operating systems
- There are many more ISPs than browser vendors, and the vendors are probably more agile
What schemes give us benefits before we have large-scale deployment?

Conversely, what schemes cause user confusion during the transition, especially when it’s mostly complete?

Can we increase confidence in a result, even for scenarios where we (should) have less than full trust?
Who is the Enemy?

To do certificate-spoofing, you must:

- Subvert a CA
- Be on-path with the victim or lure the traffic to or through you

Who can do this?

- Governments
- Sophisticated criminal hackers

Certificate-spoofing is a two-part attack, aimed at particular victims
Major Issues

- Usability
- Deployability
- Business model
- Trust model
- Threat model
- Delegation to CDNs

- Organizational structure
- Enterprise needs
- Government needs
- Existing hotspot practices
- More?
Where Are We?

- The current setup can’t last; it’s too fragile against serious enemies
- The enemy is generally very sophisticated
- The problem is overconstrained
- We have to find a good path nevertheless