How Sites Can Manage HTTPS When Users Don’t

Andy Ozment
MIT Lincoln Laboratory

Stuart Schechter
MIT Lincoln Laboratory

Rachna Dhamija
Harvard University

W3C Workshop on Transparency & Usability of Web Authentication

16 March 2006
Simplifying Users’ Tasks & Decisions

Users currently burdened with tasks related to:

1. Addressing (place)
2. HTTPS (security)

Our proposal removes users’ HTTPS tasks
Goals

• Reduce users’ cognitive overhead

• Free users to focus on location (place)

• Complement ‘anti-phishing’ research
Two methods by which to ensure a secure connection

§ Request a secure connection to a site

§ Verify security after the connection is established
Current process:
Method A: Request

Start

Task A1
Determine that intended site supports HTTPS.

Task A2
Enter https:// and domain name from trusted source (e.g. type it).

Connection Established?

Yes
Secure Connection

No

Event A1
Perhaps this site doesn't support HTTPS. Perhaps the secure site is temporarily unavailable.

No

Use plaintext HTTP?

Yes
Potential Man in the Middle Attack in Progress

Site unavailable
**Current process**

**Method B: Verify**

1. **Enter address from a trusted source (e.g. bookmark) or untrusted source (e.g. email)**
   - **Connection established?**
     - Yes: **Task B1**
     - No: **Can't reach site**

2. **Task B1**
   - **Verify that the domain name in the browser address bar belongs to the service I requested.**
   - **Can domain name be verified?**
     - Yes: **Task B2**
     - No: **Event Bi**

3. **Task B2**
   - **Verify presence of lock icon, HTTPS scheme, or address bar color.**
   - **Do indicators imply channel is secure?**
     - Yes: **Secure Connection**
     - No: **Event Bii**

4. **Event Bi**
   - **Do I trust the address provided to my browser?**
     - Yes: **Perhaps the secure site sent me here, was moved here, or is outsourced to this site.**
     - No: **Trust anyway?**
       - Yes: **Potentially Redirected by Man in the Middle Attack**
       - No: **Can't reach site**

5. **Event Bii**
   - **Perhaps this site doesn't support HTTPS. Perhaps the secure site is temporarily unavailable.**
   - **Use plaintext HTTP?**
     - Yes: **Potential Man in the Middle Attack in Progress**
     - No: **Can't reach site**
Determine that intended site supports HTTPS.

Enter https: // and domain name from trusted source (e.g. type it).

Use plaintext HTTP?

Start

Yes

Secure Connection

No

Site unavailable

Potential Man-in-the-Middle Attack in Progress

Connection Established?

Yes

No

Start

Enter address from a trusted source (e.g. bookmark) or untrusted source (e.g. email)

Can't reach site

Connection established?

Yes

No

Can domain name be verified?

Yes

Do indicators imply channel is secure?

No

Event A i

No

Yes

Secure Connection

Task A1

Determine that intended site supports HTTPS.

Task A2

Enter https: // and domain name from trusted source (e.g. type it).

Task B1

Verify that the domain name in the browser address bar belongs to the service I requested.

Task B2

Verify presence of lock icon, HTTPS scheme, or address bar color.

Can domain name be verified?

No

Do indicators imply channel is secure?

Event A ii

No

Yes

Secure Connection

Potential Man-in-the-Middle Attack in Progress

Do I trust the address provided to my browser?

Yes

Secure Connection

No

Event B i

Can't reach site

Potential Man-in-the-Middle Attack in Progress

Perhaps this site doesn't support HTTPS. Perhaps the secure site is temporarily unavailable.

Perhaps the secure site sent me here, was moved here, or is outsourced to this site.
Using our SSR Proposal:
Users’ Tasks & Decisions

Method 1: Trusted Source

Start

Enter address from a trusted source

Connection Established?

Yes

Site unavailable

Secure Connection

Method 2: Untrusted Source

Start

Enter address from an untrusted source (e.g. web link, email)

Connection Established?

Yes

Verify that the domain name in the browser address bar belongs to the service I requested.

Can domain name be verified?

Yes

Secure Connection

No

Site unavailable

No
Why Don’t We Have the Simpler Model?

Browsers have needed user input to activate HTTPS

- Browsers must default to HTTP
- Security-activation agreement problem
  - No secure means of discovering whether a site offers HTTPS
Our Proposal:
Sites Publish their Security Requirements

The Service Security Requirements (SSR) record

Example requirement:
All web connections must use HTTPS, minimum SSLv3

• Securely published and universally accessible

• SSR is a record stored in the DNS

• DNSSEC provides the security for the record itself
DNS + DNSSEC + SSR:
Query root zone DNS server

Q: What is the address of:

www.w3.org

A: I don’t know www.w3.org, but…

| .org    | = 134.58.14.3          |
| key.org | = 0xA51                |

Signed with key_root

Alice
Browser
key_root = 0x23D

Root
key_root = 0x23D
.org
key.org = 0xA51

Recursive Resolver

.(root) zone DNS server

.org zone DNS server

w3.org zone DNS server
Q: What is the address of:

www.w3.org

A: The w3.org DNS server is:

w3.org = 134.58.14.3
key_w3.org = 0xA51

Signed with key.org
**DNS + DNSSEC + SSR:**

**Query w3.org zone DNS server**

**Q:** What is the address of:  

```plaintext
www.w3.org
```

**A:**

```plaintext
www.w3.org = 134.58.14.3  
www.w3.org has an SSR record  
Signed with key w3.org
```
DNS + DNSSEC + SSR:
Return responses to client

Q: What is the address of:

www.w3.org

A: Signed with key_root

w3.org = 134.58.14.3
key_w3.org = 0xA51

Signed with key.org

www.w3.org = 134.58.14.3
www.w3.org has an SSR record

Signed with key_w3.org

www.w3.org = 134.58.14.3
www.w3.org has an SSR record

Alice

Browser

key_root = 0x23D

Root
key_root = 123.42.12.1
= 0x23D
.org
key.org = 234.238.53.142
= 0xA51
w3.org DNS server
key_w3.org = 134.58.14.1
= 0x38F
www.w3.org
= 134.58.14.3
has SSR record

.org zone
DNS server

w3.org zone
DNS server

(root) zone
DNS server
DNS + DNSSEC + SSR:
Query w3.org zone DNS server for SSR

A: The www.w3.org SSR record:
HTTPS required: SSLv3 or TLS v1
HTTP forbidden
Signed with key w3.org

Q: What are the security requirements for:
www.w3.org

A: The www.w3.org SSR record:
HTTPS required: SSLv3 or TLS v1
HTTP forbidden
Signed with key w3.org

Q: What are the security requirements for:
www.w3.org
**DNS + DNSSEC + SSR:**

Initiate HTTPS connection

**Recursive Resolver**

- . (root) zone DNS server
- .org zone DNS server
- w3.org zone DNS server
- www.w3.org

**Alice**

**Browser**

SSLv3 HTTPS (secure)
The SSR Record’s Capabilities

Enables sites to raise security level of users’ configurations
(exactly what Chuck Wade requested)

• Require protocols & protocol options
  – Cipher, keylength, etc.
  – E.g. HTTPS using SSLv3 and AES-256

• Forbid protocols & protocol options
  – E.g. no HTTP, no SHA1

• Securely redirect
  – E.g. etrade.com → secure.us.etrade.com

• Restrict subdomains
  – E.g. acceptable subdomains are login.w3.org, www.w3.org
Value of Simpler Model

Requirements presented yesterday by Ian Fette (CMU)

• Concepts at users’ level of understanding
  ✓ Removes HTTPS tasks (which are not at user level)

• Minimal interaction with user
  ✓ Removes interaction

• Should be hard to make mistakes
  ✓ SSR records prevent users’ mistakes managing HTTPS

• Works wherever the user is
  ✓ Yes, if we can seduce those browser folks (wink, wink)

• Consider disabilities
  ✓ Disabled users no longer need to see locks
SSR Design Questions

Open questions:

• Secure redirection: use S-NAPTR or build into SSR?

• How should we define a service?
  – Yes! It can be used for services other than the web.

• What other abilities could/should SSR provide?

• How to best meet the needs of browser developers?
Questions & Discussion

Questions?

Comments?

Ideas?