Paxson
Spring 2017

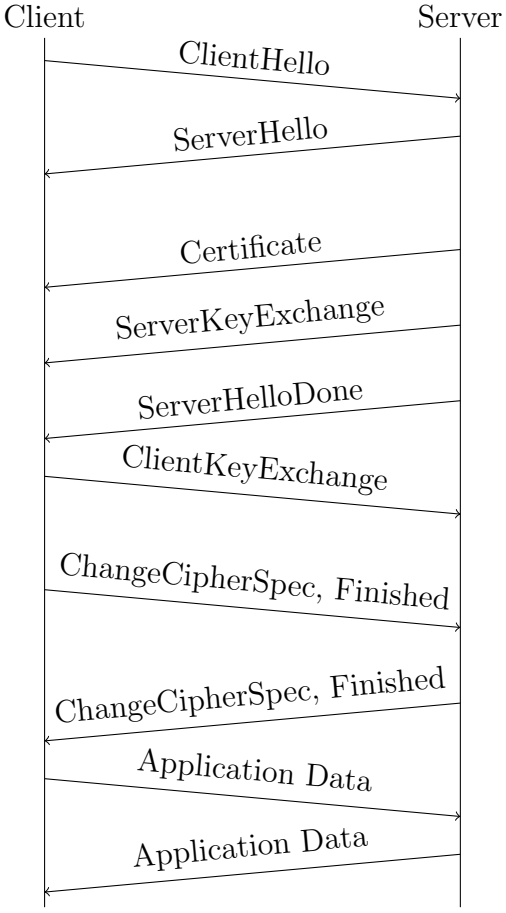
CS 161
Computer Security

Discussion 10

Week of April 10, 2017

Question 1 ***TLS protocol details*** **(25 min)**

Depicted below is a typical instance of a TLS handshake. Use the image to help answer the questions.



1. Client sends 256-bit random number R_b and supported ciphers
2. Server sends 256-bit random number R_s and chosen cipher
3. Server sends certificate
4. DH: Server sends $\{g, p, g^a \bmod p\}_{K_{\text{server}}^{-1}}$
5. Server signals end of handshake
6. DH: Client sends $g^b \bmod p$
   RSA: Client sends $\{PS\}_{K_{\text{server}}}$

   Client and server derive cipher keys C_b, C_s and integrity keys I_b, I_s from R_b, R_s, PS
7. Client sends MAC$(dialog, I_b)$
8. Server sends MAC$(dialog, I_s)$
9. Client data takes the form $\{M_1, \text{MAC}(M_1, I_b)\}_{C_b}$
10. Server data takes the form $\{M_2, \text{MAC}(M_2, I_s)\}_{C_s}$

Figure 1: TLS 1.2 Key Exchange

(a) What is the purpose of the *client random* and *server random* fields?

(b) ClientHello and ServerHello are not encrypted or authenticated. What would happen if somebody eavesdropped on them? What about an active man-in-the-middle?

(c) Unlike the ServerKeyExchange, the ClientKeyExchange is sent without a signature. Why is this the case, and what are the implications?

(d) TLS provides end-to-end authentication, integrity, and confidentiality guarantees. Is that enough to make online commerce safe and secure? Why or why not?

Question 2 ***TLS threats*** **(15 min)**

An attacker is trying to attack the company Wahoo and its users. Assume that users always visit Wahoo's website with an HTTPS connection, using RSA and AES encryption (no Diffie-Hellman). You should also assume that Wahoo does not use certificate pinning. The attacker may have one of three possible goals:

1. Impersonate the Wahoo web server to a user
2. Discover some of the plaintext of data sent during a past connection between a user and Wahoo's website a user and Wahoo's website
3. Replay data that a user previously sent to the Wahoo server over a prior HTTPS connection

For each of the following scenarios, describe if and how the attacker can achieve each goal.

(a) The attacker obtains a copy of Wahoo's certificate.

(b) The attacker obtains the private key of a certificate authority trusted by users of Wahoo.

(c) The attacker obtains the private key that was used by Wahoo's server during a past connection between a victim and Wahoo's server, but not the current private key. Also, assume that the certificate corresponding to the old private key has been revoked and is no longer valid.

Question 3 ***Denial-of-service on the web*** **(10 min)**

Your friend has just launched `SiteTester.com`, a cool web service that helps website developers see how their web site will look when rendered with Chrome vs. how it will look in Internet Explorer.

The service is pretty simple. If you visit a URL like `http://sitetester.com/?u=http://berkeley.edu/`, the SiteTester server launches a

process running the Chrome browser, loads `http://berkeley.edu/` in Chrome, and takes a screenshot of the Chrome window after the site finishes loading. In parallel, SiteTester starts up Internet Explorer, loads `http://berkeley.edu/` in Internet Explorer, and takes a screenshot of Internet Explorer after the page loads. After both screenshots are available, the SiteTester server serves you a dynamically-generated HTML document that shows both screenshots side-by-side. (Note that, while the SiteTester is handling this HTTP request from the user, it will to issue two separate HTTP requests to `berkeley.edu`, one for each browser.) The SiteTester service can be used on any web page you specify; everything after the `?u=` is treated as a URL and loaded into both browsers. This makes SiteTester very useful to web developers for testing how portable their website is.

How could an attacker mount a denial-of-service attack against the SiteTester server, simply by visiting a single URL? Show in your answer the malicious URL that causes so much trouble. You can assume the SiteTester developers haven't taken any special precautions against denial of service.