Exercises will not be graded. Try to work out your answer and then check with the one provided below. Note that this may not be the only correct answer. If you have questions, you can ask them during one of the exercise sessions on Wed 15-18 in **HG F 26.5** and NO E 39, or on Thu 15-18 in **LFW E 13** and LFW E 15. The rooms in bold are the ones we will fill up first.

1 Internationalised Domain Names

Internationalised domain names (IDNs) allow domain names to be expressed with non-ASCII characters. Under this system, zürich.ch, безопасность.рф, and even mixed domain names such as google.рф would be considered valid IDNs.

1. Suppose that when your browser initiates an SSL connection to an HTTPS site, it does not verify that the domain name listed in the certificate matches the domain name the browser is connected to. How might an adversary fool you into accepting the certificate, even if you visually verify that the domain names match?
2. Browsers can prevent the above attack by rendering URLs in what is called Punycode, which uniquely maps a Unicode string to an ASCII string. Using this scheme, paypal.com would map to xn–pypal-4ve.com, which is easily distinguishable from the true PayPal site. However, this defense has an unexpected side effect that still allows the adversary to carry out a man-in-the-middle attack. What is this side effect? (Hint: Suppose that you work at a company in Russia, and your salary is deposited into a local Russian bank.)

Solution

1. The adversary can use a mixed domain name such as ethz.ch, whose first character is actually the Cyrillic letter e. Because this is extremely difficult to visually distinguish from the ASCII letter e, most users would be fooled into thinking that the certificate is valid.
2. If URLs are rendered in Punycode, it becomes much more difficult for users to distinguish URLs. For example, безопасность.ru maps to xn–80abmi5aecftcl4j.ru while безопасный.ru maps to xn–80abmilxfit5h.ru. Users cannot easily determine which site is which, allowing an adversary to easily redirect the user to a different site in order to mount a man-in-the-middle attack.

2 SSL/TLS Attacks

In this problem we will look at three subtle attacks on the SSL/TLS protocol.

1. Some banks display "security phrases/pictures" to their users when allowing them to log in. Users first type in their username to the bank's webpage. The bank then displays their security phrase or picture so that the user knows they are connected to the correct site. The user can then type in their password and log into their account. Suppose the initial bank webpage does not use SSL, allowing an attacker to obtain a username entered in the webpage. What attack is possible?
2. Most users do not type https into their browser when accessing an SSL-enabled site. Rather, they reach such a site via links (through a Google search, for instance) or via redirects (for example, their bank might automatically redirect them to the HTTPS-enabled version of the site). How mght an adversary take advantage of this tendency to mount a man-in-the-middle attack? (Assume that the user does not check the URL too carefully.)

3. Suppose a client tries to connect to PayPal. The client's browser checks that the name in the certificate matches the URL by reading both strings character by character from the beginning and making sure that each pair of characters is equal. If the equality check succeeds, the client proceeds to validate the signature on the certificate and continue with the SSL handshake. How might an adversary carry out a man-in-the-middle attack in this situation?

Solution

1. The adversary can simply mount a man-in-the-middle attack to spoof the user's security phrase and picture. As soon as the user has input his username, the adversary can obtain his security phrase and picture and display them to him in a normal HTTP site. Since the user sees the security phrase and picture prominently, the attack is likely to succeed.
2. If for example the page containing the link to the HTTPS-enabled site is not itself an HTTPS site, then the adversary could intercept that page and replace all links to HTTPS sites with HTTP addresses. As soon as the adversary sees the user request such an address, it sets up an SSL connection with the HTTPS version of the server and proxies the user's traffic, enabling a man-in-the-middle attack. Since the page will display with the correct URL (only with http rather than https), the attack will be successful.
3. The adversary can obtain a legitimate wildcard certificate for *.evil.com (or whatever his own site is called). When a client tries to connect to PayPal, for example, the adversary sends a certificate for paypal.com\0evil.com. The browser will read the name to the null character and see that the URL (paypal.com) matches the name on the certificate. Since the certificate has a valid signature on it, the client will proceed with the SSL handshake.

3 Brute-Forcing SSL

A client (without certificate) and a server (with certificate) set up a secure SSL connection. The server supports the following SSL3 ciphersuites:

a) RC4 encryption with a 128-bit key and an MD-5 MAC

b) Triple DES encryption with a 168-bit key and a SHA-1 MAC

c) FIPS 140-1 compliant triple DES encryption and SHA-1 MAC

d) DES encryption with a 56-bit key and a SHA-1 MAC

e) DES encryption with a 40-bit key and a SHA-1 MAC

The client supports the following SSL3 ciphersuites:

a) FIPS 140-1 compliant DES encryption and SHA-1 MAC

b) RC4 encryption with a 128-bit key and an MD5 MAC

c) RC4 encryption with a 40-bit key and an MD5 MAC

Assume that the time taken to brute force a key of length L is 2^L seconds.

1. Assuming the client and server use ephemeral Diffie-Hellman to set up the key, what is the minimum amount of time in which the attacker can brute force the encryption key for traffic from client to server? Explain why.
2. Assuming the client and server use anonymous Diffie-Hellman to set up the key, what is the minimum amount of time in which the attacker can brute force the key used by the client and the server? Explain why.

Solution

1. In SSL, the client first tells the server the list of the ciphersuites that it supports, and the server picks one from the list for the communication after negotiation. Since the list is in the decreasing order of preference, the server is supposed to choose "RC4 encryption with a 128-bit key." Assuming that ephemeral Diffie-Hellman is a secure key exchange protocol, the attacker should brute force the 128-bit encryption key. Therefore, it takes 2^{128} seconds.

2. No time needed for brute forcing. Anonymous Diffie-Hellman is vulnerable to man-in-the-middle attacks, so the attacker can easily get the keys exchanged by the anonymous Diffie-Hellman protocol.

4 Diffie-Hellman

Consider Alice and Bob that have two cell phones that communicate over a wireless network and wish to set up a secret key between each other. They do not share any secrets, nor do they have any PKI certificates. To evade the man-in-the-middle attack, they establish a Diffie-Hellman key K_{AB} and compare the hash output as follows. Alice and Bob's phones will both display the least significant 32 bits of the hash of key K_{AB}, e.g., $[\text{MD5}(K_{AB})]_{32}$, and Alice and Bob compare their hashes to ensure that they share a key.

Does this approach really prevent a man-in-the-middle attack? Argue why or why not. If not, how you would fix the protocol to prevent the attack?

Solution

This scheme attempts to prevent the MITM attack by comparing the key resulting from a DH key exchange, attempting to make sure the protocol yielded the same key for both Alice and Bob. If a MITM attack takes place, $[MD5(K_{AB})]_{32}^{Alice} \neq [MD5(K'_{AB})]_{32}^{Bob}$ because $MD5(g^{am_a})$ will not equal $MD5(g^{bm_b})$. This approach provides very little protection against a MITM attack because of the small size of the search space for an attacker. Since only 32 bits are compared, the attacker must generate m_a, m_b such that $[MD5(g^{am_a})]_{32} = [MD5(g^{bm_b})]_{32}$. There are 2^{32} possible hashes, but since the attacker has control over both inputs (m_a and m_b), he must generate on average $\sqrt{2^{32}} = 2^{16}$ values. This is a relatively small number and can easily be performed on commodity hardware. To fix the protocol, one needs to compare more of the MD5 output so the space the attacker must search is computationally infeasible.

An alternative approach to securing this protocol is to have Alice and Bob commit to their respective half-keys, so that the attacker cannot do any pre-computation.

5 SSL Covert Channels

In the Ephemeral Diffie-Hellman key exchange in SSL, the server sends its Diffie-Hellman public key to the client first, then the client sends its Diffie-Hellman public key to the server. Please briefly describe a covert channel attack in which malicious code running on the client machine sends 3 bits of information with each SSL session establishment without being detected.

Solution

The client selects its private key c, computes its public key $g^c mod p$, then computes pre-master key $K = g^{cs} mod p$. If the last three bits of K provide the information, the client continues, otherwise, it picks a new private key c. On average, this requires four trials for each protocol run.