Paxson
Spring 2017

CS 161
Computer Security

Discussion 7

Week of March 13, 2017

Question 1 ***Diffie–Hellman key exchange*** **(15 min)**

Recall that in a Diffie-Hellman key exchange, there are values a, b, g and p. Alice computes $g^a \bmod p$ and Bob computes $g^b \bmod p$.

(a) Which of these values are publicly known and which must be kept private?

Solution:

g, p, $g^a \bmod p$, and $g^b \bmod p$ are publicly known. Implementations of Diffie-Hellman often have carefully picked values of g and p which are known to everyone. Alice and Bob must keep a and b secret respectively. However they compute $g^a \bmod p$ and $g^b \bmod p$, respectively, and send these values to each other so each person may compute the secret key $g^{ab} \bmod p$.

(b) Eve can eavesdrop on everything sent between Alice and Bob, but can't change anything. Alice and Bob run Diffie-Hellman and have agreed on a shared symmetric key K. However, Bob accidentally sent his b to Alice in plain text. If Eve viewed all traffic since the beginning of the exchange, can she figure out what K is?

Solution:

It depends on when Bob sent b. If he sent it during the DH exchange *instead* of sending $g^b \bmod p$, then he and Alice will compute different key values (Alice will compute $b^a \bmod p$, while Bob will compute $(g^a)^b \bmod p$) and be unable to communicate. Thus, there's no well-defined K for Eve to figure out.

However, if Bob sent b in the clear after the exchange, then it will be very easy for Eve to figure out K: she can use the value $A = g^a \bmod p$ which Alice sent to calculate $K = A^b \bmod p$.

(c) Mallory can not only view all Alice↔Bob communications but also intercept and modify it. Alice and Bob perform Diffie-Hellman to agree on a shared symmetric key K. After the exchange, Bob gets the feeling something went wrong and calls Alice. He compares his value of K to Alice's and realizes that they are different. Explain what Mallory has done and what she can now do.

Solution:

Mallory is performing a **man-in-the-middle attack**. Mallory pretends to be Bob when she talks to Alice, and Mallory also pretends to be Alice when she talks to Bob. In this way, both Alice and Bob are unknowingly talking to Mallory. Mallory can then decrypt/re-encrypt the traffic in both directions and modify it however she wishes to.

More technically, when Alice sends $A = g^a \bmod p$ to Bob, Mallory intercepts this (preventing it from going to Bob), and sends back to Alice: $M = g^c \bmod p$. Now when Alice sends a message to Bob, she uses $K_{bad} = M^a \bmod p$ which Mallory knows as $K_{bad} = A^c \bmod p$. Mallory can then decrypt all messages sent from Alice. She can also send messages to Alice which Alice thinks are from Bob. Mallory then does the same trick to Bob.

Question 2 *Perfect Forward Secrecy* (15 min)

Alice (A) and Bob (B) want to communicate using some shared symmetric key encryption scheme. Consider the following key exchange protocols which can be used by A and B to agree upon a shared key, K_{ab}.

RSA-Based Key Exchange Protocol

Message 1	$A \rightarrow B$:	$\{K_{ab}\}_{K_B^{pub}}$
	Key exchanged	
Message 2	$A \leftarrow B$:	$\{secret1\}_{K_{ab}}$
Message 3	$A \rightarrow B$:	$\{secret2\}_{K_{ab}}$

Diffe-Hellman Key Exchange

Message 1	$A \rightarrow B$:	$g^a \mod p$
Message 2	$A \leftarrow B$:	$g^b \mod p$
	Key exchanged $K_{ab} = g^{ab} \mod p$	
Message 3	$A \leftarrow B$:	$\{secret1\}_{K_{ab}}$
Message 4	$A \rightarrow B$:	$\{secret2\}_{K_{ab}}$

Some additional details:

- K_B^{pub} is Bob's long-lived public key.
- All messages are destroyed immediately after reading them.
- K_{ab} and DH exponents a and b are destroyed once all messages are sent.

Eavesdropper Eve records all communications between Alice and Bob, but is unable to decrypt them. At some point in the future, Eve is lucky and manages to compromise Bob's computer.

(a) Is the confidentiality of Alice and Bob's prior RSA-based communication in jeopardy?

Solution: Yes. The compromise of Bob's computer gives Eve access to Bob's private key, allowing Eve to decrypt the traffic she previously recorded that was encrypted using Bob's public key. Once decrypted, she obtains K_{ab}, and can then apply it to decrypt the traffic encrypted using symmetric key encryption.

(b) What about Alice and Bob's Diffe-Hellman-based communication?

Solution: No. Since Alice and Bob destroy the DH exponents a and b after use, and since the key computed from them itself is never transmitted, there is no information present on Bob's computer that Eve can leverage to recover K_{ab}. This means that with Diffie-Hellman key exchanges, later compromises in no way harm the confidentiality of previous communication, even if the ciphertext for that communication was recorded in full. This property is called *Perfect Forward Secrecy.*

Question 3 ***Introduction to Networking*** **(10 min)**

(a) **Protocol Layers.** At which network layer does each of the following operate (physical, link, network, transport, or application)?

Solution:

- Ethernet – **Physical (1), Link (2)**
- SMTP (email) – **Application (7)**
- SYN packet – **Transport (4)**
- UDP – **Transport (4)**
- Fiber optics – **Physical (1)**
- FTP – **Application (7)**
- DNS request – **Application (7)**
- BitTorrent – **Application (7)**
- IP address – **Network (3)**
- 127.0.0.1 – **Network (3)**
- 802.11n WiFi – **Physical, Link (1, 2)**

(b) **TCP and UDP.** The transmission control protocol (TCP) and user datagram protocol (UDP) are two of the primary protocols of the Internet protocol suite.

i. How do TCP and UDP relate to IP (Internet protocol)? Which of these protocols are encapsulated within (or layered atop) one another? Could all three be used simultaneously?

ii. What are the differences between TCP and UDP? Which is considered "best effort"? What does that mean?

Solution:

i. TCP and UDP both exist within the transport layer, which is one layer above IP (network layer). Either can be encapsulated in IP, referred to as TCP/IP and UDP/IP. TCP and UDP are alternatives; neither would normally be encapsulated within the other.

ii. TCP provides a *connection-oriented*, *reliable*, *bytestream* service. It includes sophisticated rate-control enabling it to achieve high performance but also respond to changes in network capacity. UDP provides a *datagram-oriented*, *unreliable* service. (Datagrams are essentially individual packets.) The main benefit of UDP is that it is lightweight.

"Best effort" refers to a delivery service that simply makes a single attempt to deliver a packet, but with no guarantees. IP provides such a service, and because UDP simply encapsulates its datagrams directly into IP packets with very little additional delivery properties, it, too, provides "best effort" service.