CS 494/594
Computer and Network Security

Dr. Jinyuan (Stella) Sun
Dept. of Electrical Engineering and Computer Science
University of Tennessee
Fall 2010
Exercise: Chapters 13, 15-18
1. [Kaufman] 13.1
Design a variant of Kerberos in which the workstation generates a TGT. The TGT will be encrypted with the user’s master key rather than the KDC’s master key. How does this compare with standard Kerberos in terms of efficiency, security, etc.? What happens in each scheme if the user changes her password during a login session?
Let Alice be the user. In this variant, Alice’s TGT is just $K_{\text{Alice}}\{\text{"Alice"}, S_A\}$, where Alice (actually, her workstation) has invented the session key $S_A$. This causes a problem for the KDC when it wants to get the session key from the TGT. The KDC has to try user keys until it finds the correct key to decrypt the TGT. To avoid this problem, Alice’s name/instance/realm (unencrypted) should always be paired with the TGT when transmitted (it is already included in the encrypted TGT).

Since the only purpose of the TGT is to allow use of the short-term session key $S_A$ instead of the long-term master key $K_{\text{Alice}}$ when Alice and the KDC talk, and since knowledge of $K_{\text{Alice}}$ is all that is required for mutual authentication of Alice and the KDC, there is no difference in security between the two schemes except when Alice changes her password (and thus her key $K_{\text{Alice}}$) during a session. If Alice believes someone has learned her password and therefore changes it, with normal Kerberos the TGT obtained by the impostor remains valid until expiration. With the modified scheme, the TGT would be immediately invalidated, though tickets could continue to be used until expiration.

In normal Kerberos, Alice can change her password (and thus her key $K_{\text{Alice}}$) during a session, because $S_A$ is used for encryption, and $S_A$ is in the TGT encrypted with $K_{\text{KDC}}$ (so the TGT remains valid). In the variant, changing $K_{\text{Alice}}$ invalidates the TGT, so to change her password, Alice would have to send a message containing the old TGT, and the new $K_{\text{Alice}}$ encrypted with the unchanged $S_A$. She would then compute a new TGT for herself.
2. [Kaufman] 15.4

Compare the following schemes for obtaining Bob's public key, in terms of bandwidth and computation efficiency, security, flexibility, and any other criteria you can think of: downloading Bob's key from the node located at a particular IP address (via an unauthenticated interaction), looking up Bob's key in a directory via an unauthenticated interaction, having an authenticated conversation to the directory, having the directory sign the information you request, storing and retrieving certificates from the directory, having no directory but having each principal responsible for keeping its own certificate and sending it to someone who needs to talk to it.
2. [Kaufman] 15.4

Downloading Bob's key from whatever responds to what is believed to be his IP address. If someone can intercept traffic sent to Bob's IP address, they can give Alice (the requester) the wrong public key. However, the advantage of this scheme is that in order for Alice and Bob to communicate, only Alice and Bob need to be available—no directory, no KDC, etc. This is computationally cheap since there is no authentication required to obtain Bob's public key.

Looking up Bob's key in a directory via an unauthenticated interaction. If someone breaks into the directory, they can install incorrect information. They can also impersonate the directory through various means, and give out faulty information, even if they can't actually break into the directory. This is computationally cheap since there is no authentication required to obtain Bob's public key.

Authenticated interaction with the directory. Now nobody can impersonate the directory, but the directory is an on-line trusted thing, so if someone were to compromise the directory, they could subvert security.

Having the directory sign the information you request. Similar security properties to above (authenticated interaction with the directory), but it means that Bob can obtain signed information from the directory and send it to Alice, rather than having everyone that wants to talk to Bob have to carry on an authenticated exchange with the directory.

Storing and retrieving certs from the directory. This is more secure because the trusted party can be off-line. Compromising the directory can be at worst a denial of service.

Having each principal keep its own certificate. This might make chain-building difficult, unless there were a globally agreed-upon root, in which case everyone could keep the complete chain from the root to themselves. It is less flexible, since Bob has to be available for Alice to obtain his cert before she can do something like compose an encrypted message for him.
3. [Kaufman] 16.7
Devise a protocol based on a pre-shared secret key that hides identities and gives PFS for identity hiding. Make two variants, one in which an active attacker can learn only the initiator's identity, and one in which an active attacker can learn only the target's identity.
3. [Kaufman] 16.7

Messages 1 and 2: Diffie-Hellman exchange.

- variant in which active attacker can learn initiator’s identity: in message 3, initiator sends identity, and proof of knowledge of the shared key, encrypted with the Diffie-Hellman key. In message 4, the target sends its identity and proof of knowledge of the shared key encrypted with the Diffie-Hellman key.

- variant in which active attacker can learn target’s identity: message 2 consists of Diffie-Hellman number and, encrypted with the Diffie-Hellman key, the target’s identity and proof of knowledge of shared key. Message 3 is as in previous bullet.
4. [Kaufman] 16.11

In the Protocol 16-6, explain why Bob knows that Alice is the real Alice, and not someone replaying Alice's messages. How does Alice know that it's the real Bob if she uses a different a each time? Modify the protocol to allow both Alice and Bob to reuse their a and b values, and yet have both sides be able to know they are talking to a live partner.
4. [Kaufman] 16.11

Bob knows it’s the real Alice because she knows the key $K$, which is a function of the nonce as well as a Diffie-Hellman value signed by Alice. If someone had stolen a previous “a” and the signed $g^a \mod p$, then they can impersonate Alice to Bob, however. Hopefully it is as unlikely someone can steal an $a$ as stealing Alice’s private key. If Alice uses a different $a$ each time, then she knows it’s the real Bob, and not someone replaying Bob’s messages. But if she reuses $a$, then she can’t tell.

To modify the protocol so she can tell, and allow both Alice and Bob to reuse their Diffie-Hellman values (to save computation), Alice sends a nonce in message 1, and have $K$ be a function of both nonces as well as the Diffie-Hellman value.
5. [Kaufman] 17.1
Suppose Alice is sending packets to Bob using IPsec. Suppose Bob's TCP acknowledgment gets lost, and Alice's TCP, assuming the packet was lost, retransmits the packet. Will Bob's IPsec implementation notice that the packet is a duplicate and discard it?
5. [Kaufman] 17.1

No. IPsec treats a retransmitted TCP packet as a new IPsec packet. It is up to TCP to notice the packet is a duplicate.
6. [Kaufman] 17.4
Would it be possible for the SA to be defined only by the destination address and the SPI (i.e., leave out whether it's ESP or AH)? Would this require any changes to the IPsec protocol? Would an implementation of a receiver that defined the SA based solely on destination address and SPI interwork with one that did what the IPsec specification says?
6. [Kaufman] 17.4

Yes to both questions. Since it’s the receiver that defines the SPI, it can assign different SPIS to ESP SAs vs. AH SAs. This will interwork with implementations that allow the same SPI to be assigned to both, and distinguish which SA it belongs to based on whether it’s AH or ESP.
7. [Kaufman] 17.5
When sending encrypted traffic from firewall to firewall, why does there need to be an extra IP header? Why can't the firewall simply encrypt the packet, leaving the source and destination as the original source and destination?
7. [Kaufman] 17.5

Suppose one portion of your Intranet is connected to the Internet with firewall F1, and another portion of your Intranet is connected with firewalls F2 and F3. All addresses inside that portion are reachable equally well through F2 and F3. Since SAs are pairwise, F1 will have two SA's: one to F2, and one to F3. When F1 forwards a packet for destination D, it has to choose which SA to send it on, encrypting the packet with the key for the F1-F2 SA or with the key for the F1-F3 SA. Internet routing can route packets for D via either F2 or F3. If it chooses a different F than F1 assumed, then it will not work. So F1 has to specify which of the F's the Internet should deliver the packet to.
8. [Stallings] 16.4
The IPsec architecture document states that when two transport mode SA's are bundled to allow both AH and ESP protocols on the same end-to-end flow, only one ordering of security protocols seems appropriate: performing the ESP protocol before performing the AH protocol. Why is this approach recommended rather than authentication before encryption?
8. [Stallings] 16.4

This order of processing facilitates rapid detection and rejection of replayed or bogus packets by the receiver, prior to decrypting the packet, hence potentially reducing the impact of denial of service attacks. It also allows for the possibility of parallel processing of packets at the receiver, i.e., decryption can take place in parallel with authentication.
9. [Kaufman] 18.1
Suppose if Alice's aggressive-mode IKE connection initiate is refused, Alice starts up another aggressive-mode connection initiate with her next (and weaker) choice of Diffie-Hellman group, rather than starting a main-mode exchange telling Bob all her supported Diffie-Hellman groups. What is the vulnerability, given an active attacker?
9. [Kaufman] 18.1

The active attacker can trick Alice into using weak crypto. The bad guy can impersonate Bob and refuse Alice’s strong crypto proposal. Then Alice will retry with a weaker proposal.
10. [Kaufman] 18.5
How can you modify aggressive mode with public signature keys (see § 18.5.10.2 Public Signature Keys, Aggressive Mode) to hide the endpoint identifiers from eavesdroppers? Which identity will be hidden from an active attacker? Give a disadvantage of this.
10. [Kaufman] 18.5

Remove Alice's name from message 1. Encrypt Bob's name, certificate, and proof with $g^{ab} \mod p$ in message 2. Encrypt Alice's name, proof and certificate with $g^{ab} \mod p$ in message 3.

The disadvantage is that Bob has to compute $g^{ab} \mod p$ before he can send message 2 and Alice can't start computing it until she receives message 2, so there is no opportunity for parallel computation.
11. [Kaufman] 18.8
Design a protocol in which authentication is one-way since only one side has a public key. Do the protocol with a public signature key. Now do it with a public encryption key.
11. [Kaufman] 18.8

Bob has signature key. Client has no key. Client knows it's talking to Bob, but not vice versa:

Bob has encryption key. Client has no key. Client knows it's talking to Bob, but not vice versa: