Ryan Young

CSE 3400

Midterm #2

4/16/2020

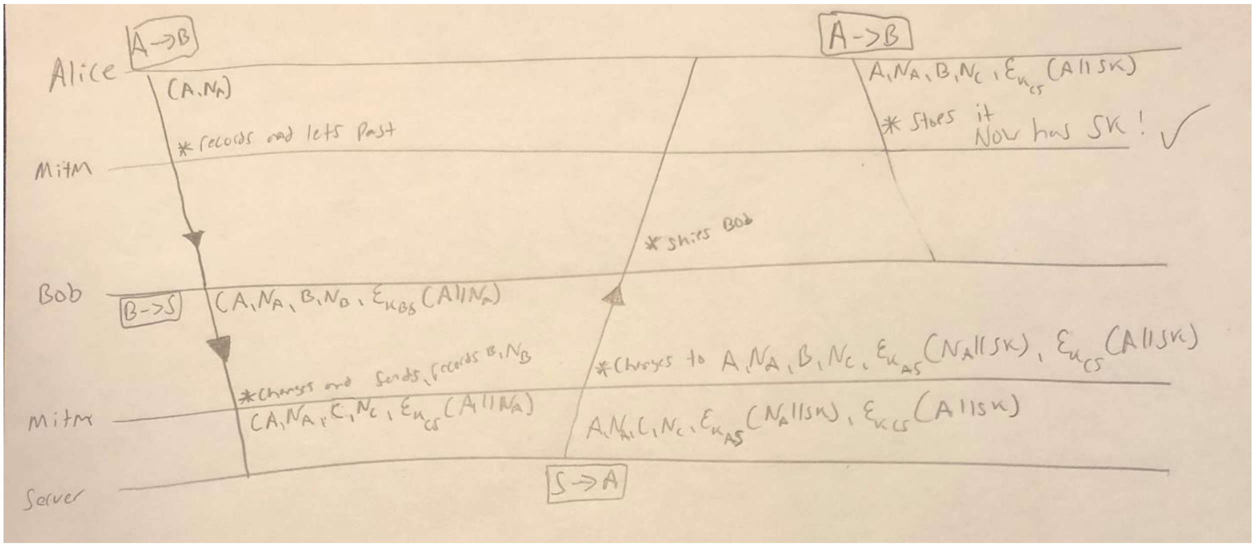
1.) *Consider the following key establishment protocol between any two users A and B with an assistance of a server S. Each user U shares a secret key K_{US} with S. For exmaple, A and S share K_{AS}, and B and S share K_{BS}. In the following, N_A is a nonce randomly generated by A, N_B is a nonce randomly generated by B, and sk is the key randomly generated by S. A->B indicates a message from A to B. E is an authenticated encryption.*

$$\begin{aligned}
&A \rightarrow B : (A, N_A)\\
&B \rightarrow S : (A, N_A, B, N_B, \mathcal{E}_{K_{BS}}(A\|N_A))\\
&S \rightarrow A : (A, N_A, B, N_B, \mathcal{E}_{K_{AS}}(N_A\|sk), \mathcal{E}_{K_{BS}}(A\|sk))\\
&A \rightarrow B : (A, N_A, B, N_B, \mathcal{E}_{K_{BS}}(A\|sk))
\end{aligned}$$

Show an attack which allows an attacker, who has the MitM capability, to impersonate one of the parties to the other. At the end of the authentication, the attacker should have a shared key establish with the other party.

*Important Note: k_{CS} is the shared key between the man in the middle and the server

Sequence Diagram:



Alice
A->B
(A, N_A)
* records and lets past
MitM
Bob
B->S
(A, N_A, B, N_B, $\mathcal{E}_{K_{BS}}$(A||N_A)
*Changes and sends, records B, N_B
(A, N_A, C, N_C, $\mathcal{E}_{K_{CS}}$(A||N_A)
mitm
Server
S->A
A, N_A, C, N_C, $\mathcal{E}_{K_{AS}}$(N_A||sk), $\mathcal{E}_{K_{CS}}$(A||sk)
*Changes to A, N_A, B, N_C, $\mathcal{E}_{K_{AS}}$(N_A||sk), $\mathcal{E}_{K_{CS}}$(A||sk)
* skips Bob
A->B
A, N_A, B, N_C, $\mathcal{E}_{K_{CS}}$(A||sk)
* Stores it
Now has SK!

Assuming the man in the middle classifies as a user and has their own shared key with the server, then he or she can impersonate Bob. Alice will send: A, N_A to Bob and the man in the middle will record it and let it move to Bob. Bob will now as a result go to send A, N_A, B, N_B, $E_{k_{BS}}(A||N_A)$ to the server however the man in the middle will record B and N_B stop Bob's message and instead send the server: A, N_A, C, N_C, $E_{k\,CS}$(A||N_A). The server will then send Alice: A, N_A, C, N_C $E_{k\,AS}$(N_A||sk), $E_{k\,CS}$(A||sk), the man in the middle will stop it and instead send: A, N_A, B, N_C, $E_{k\,AS}$(N_A||sk), $E_{k\,CS}$(A||sk). Alice will finally send Bob (A, N_A, B, N_C, $E_{K\,CS}$(A||SK). The man in the middle will intercept this message and record it not allowing it to go to Bob. Now the man in the middle has successful obtained the SK and now has a shared key with Alice without her knowing she has shared in with the attacker rather than Bob.

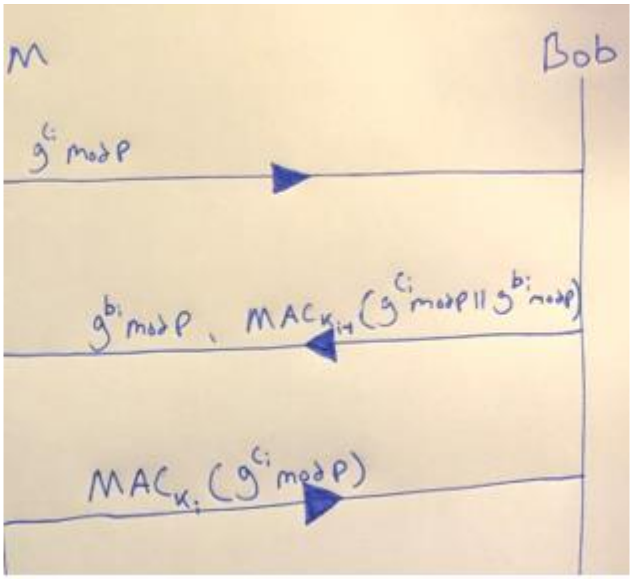
2.) *The following is a slightly different protocol for using authenticated DH to ensure resilient key exchange. A is Alice and B is Bob.*

$$\begin{aligned} &A \rightarrow B : (g^{a_i} \bmod p) \\ &B \rightarrow A : \left(g^{b_i} \bmod p, \mathtt{MAC}_{k_{i-1}}\left(g^{a_i} \bmod p \| g^{b_i} \bmod p\right)\right) \\ &A \rightarrow B : (\mathtt{MAC}_{k_i}\left(g^{a_i} \bmod p\right)) \end{aligned}$$

After the first two messages, both A and B can compute session key:

$$k_i = h(g^{a_i b_i} \bmod p)$$

Where h is a secure key derivation function. Present a sequence diagram showing that this protocol is not secure. Hint: Show how an attacker is able to impersonate as Alice, without knowing any of Alice's previous keys. At the end of the handshake, Bob will believe it has exchange key k_i with Alice, but the key was actually exchanged with the attacker.


M
Bob
g^{c_i} mod P
g^{b_i} mod P, $MAC_{K_{i-1}}(g^{c_i}$ mod P || g^{b_i} mod P)
$MAC_{K_i}(g^{c_i}$ mod P)

*M in this diagram is the attacker

First the attacker sends g^{ci}mod p to Bob (g and p are known public information and c_I in this case is an attacker chosen number so he or she can create the session key with Bob), Bob replies by sending g^{bi}mod p, MAC_{k-1}(g^{ci}mod p || g^{bi}mod p). Next the attacker sends MAC_{ki}(g^{ci}mod p) by deriving k_i using k_i = h(g^{cibi}mod p). This can be done because after the first two messages (one attacker -> Bob and one Bob -> Attacker) the key k_i can be generated.

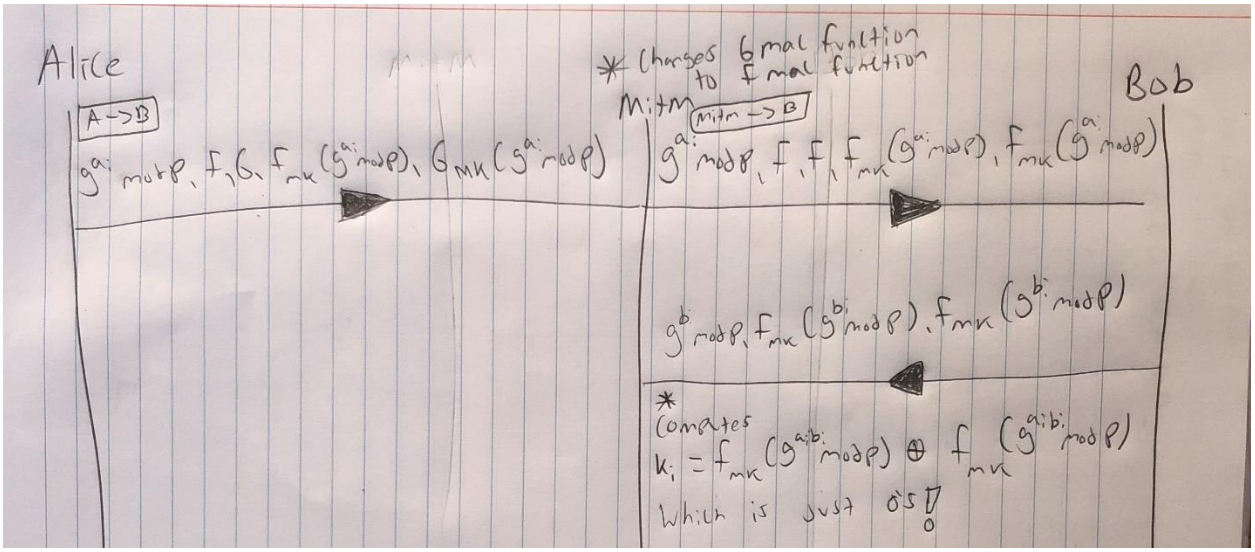
3.) *The following protocol is an (incorrect) attempt at a robust combiner authenticated DH protocol. A is Alice and B is Bob. A and B share a master key MK. F and G are two MAC functions randomly selected from a set of MAC functions by A. B checks if both F and G are in the set.*

$$A \to B : (g^{a_i} \bmod p, F, G, F_{MK}(g^{a_i} \bmod p), G_{MK}(g^{a_i} \bmod p))$$
$$B \to A : (g^{b_i} \bmod p, F_{MK}(g^{b_i} \bmod p), G_{MK}(g^{b_i} \bmod p))$$

Then, both A and B can compute session key:

$$k_i = F_{MK}(g^{a_i b_i} \bmod p) \oplus G_{MK}(g^{a_i b_i} \bmod p)$$

Show a sequence diagram for an attack where a MitM attacker, without knowing MK, can impersonate A and establish a session key with B. Explain your diagram.


Alice
A -> B
g^{a_i} mod p, F, G, $F_{mk}(g^{a_i}$ mod p), $G_{mk}(g^{a_i}$ mod p)
* Changes G mac function to F mac function
MitM
MitM -> B
g^{a_i} mod p, F, F, $F_{mk}(g^{a_i}$ mod p), $F_{mk}(g^{a_i}$ mod p)
Bob
g^{b_i} mod p, $F_{mk}(g^{b_i}$ mod p), $F_{mk}(g^{b_i}$ mod p)
*
Computes
$k_i = f_{mk}(g^{a_i b_i}$ mod p) $\oplus$ $f_{mk}(g^{a_i b_i}$ mod p)
Which is just 0's !

To start Alice sends:

$$A \to B : (g^{a_i} \bmod p, F, G, F_{MK}(g^{a_i} \bmod p), G_{MK}(g^{a_i} \bmod p))$$

To Bob but the man in the middle makes a key change instead of using two different MACs the man in the middle makes Bob use the F MAC for both. He or she does this by sending Bob:

g^{ai}mod p, F, F, F_{MK}(g^{ai}mod p), F_{MK}(g^{ai}mod p)

Bob now sends back:

$$g^{bi}\text{mod } p, F_{MK}(g^{bi}\text{mod } p), F_{MK}(g^{bi}\text{mod } p)$$

Now the man in the middle knows the session key because k_i is computed by xoring the outputs of both MAC functions and since they output the same the key will now be just a string of all 0s.

4.) *Consider the use of the textbook RSA for encryption (no padding). Present an algorithm for an attacker to win the IND-CCA game (Show that the textbook RSA is insecure against a chosen-ciphertext attack).*

Textbook RSA encryption (with no padding) is indeed vulnerable to Chosen Ciphertext Attacks and thus an attacker can win the IND-CCA game against this encryption scheme. Say the attacker wants to decrypt the ciphertext $c = m^e$ mod n, the attacker will simply make the query $c^l = c * (m^l)^e$ mod n (c and c^l are different ciphertexts, as well as m and m^l are different messages with m^l being known to the attacker and m not being known). This will return $m * m^l$ and since the attacker knows m^l they can find m and thus win the IND-CCA game against textbook RSA encryption (with no padding). The textbook RSA encryption scheme was also shown to be vulnerable to a practical feedback-only CCA attack by Bleichenbacher.

5.) *Assume a programmer is a contributor to the Linux kernel. To test their changes they push them to a powerful remote server that complies the kernel. The server encrypts the result of the compilation using the EL-Gamal cryptosystem (the server has the programmer's public key). A message m = 1 corresponds to a successful compilation, any other message m != 1 corresponds to an error. Explain how a MitM attacker that has the programmer's public key can make the programmer think their changes cause arbitrary errors during the kernel compilation process.*

Given that the attacker is a man in the middle, he or she will have the ability to change or manipulate messages sent on the line of communication. So, a man in the middle attack could simply watch the line of communication, wait for a message to be sent from the remote server and xor the last half of the tuple ($M*e_A{}^B$ mod p) ciphertext with a string of 1's. The programmer will now assume that their changes caused arbitrary errors during the kernel compilation process. This works because this xor will not be checked in decryption process and if will flip all the bits meaning that it would change the encrypted 1 to something else.