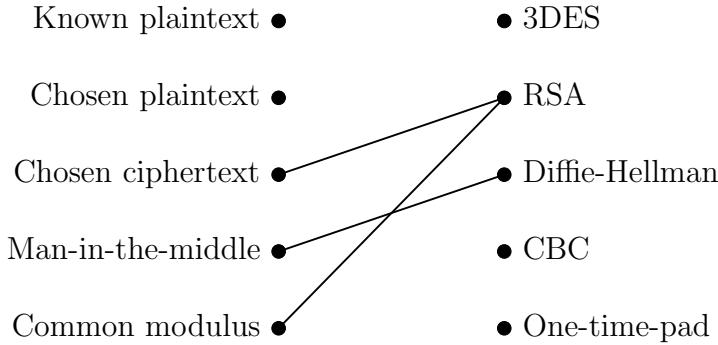


Midterm Exam

Question 1: Attacks (15) - Nodes on the left represent attacks. Nodes on the right represent protocols. Connect any node on the left with a node on the right if the protocol is vulnerable to the attack.



Known plaintext: attacker has samples of plaintext *and* its ciphertext; can use them to reveal secret information such as a secret key.

Chosen plaintext: attacker can choose arbitrary plaintexts to be encrypted and obtain the corresponding ciphertexts; goal of the attack is to gain information which reduces the security of the encryption scheme; for example, the scheme's secret key.

Chosen ciphertext: attacker chooses a ciphertext and causes it to be decrypted with an unknown key.

Man-in-the-middle: attacker can read, insert and modify at will, messages between two parties without either party knowing that the link between them has been compromised.

Common modulus: RSA public keys of two people share n , have relatively prime e_1 and e_2 . Same message sent encrypted as c_1 and c_2 to both. Attacker solves $re_1 + se_2 = 1$ and computes $c_1^{-1} \bmod n$. Then $(c_1^{-1})^{-r} * c_2^s = m^{e_1*r + e_2*s} \bmod n = m \bmod n$.

Question 2: Message Integrity (15) - Assume a Server and Client share a secret key, K , and a Message Digest algorithm. Denote by $\text{SHA}(X)$ the output of the Message Digest algorithm applied to input X . Use $A|B$ to denote updating the Message Digest with A then B . Suppose the Client sends a message M to the Server in plaintext. The Client also sends $C = \text{SHA}(M|K)$ to the Server. To make sure the message is unmodified in transit, the Server computes $\text{SHA}(M|K)$, which is easy because it possesses both K and M , and checks against C . If they match, the message is declared unmodified.

1. *What can go wrong with this idea?* This is much better than putting the secret at the front. However, it is vulnerable to an off-line guessing attack.
2. *Why?* An attacker knows M and therefore $\text{SHA}(M)$ which is at the end of the digest $\text{SHA}(M|K)$ except for K . So, an attacker might be able to generate a message that produces the same digest as $\text{SHA}(M)$. In this case the MAC of the original message can be used to authenticate the forged message.
3. *Propose a fix.* Use $\text{SHA}(K \oplus \text{opad}|\text{SHA}(K \oplus \text{ipad}|M))$ where $\text{opad} = 0x363636\dots$ and $\text{ipad} = 0x5C5C5C\dots$

Question 3: Zero-Knowledge Proofs (15) - Choose one from (a) State what happens on a round of a zero-knowledge interactive protocol based on Graph Isomorphism and explain why this is safe; (b) There is a very long but narrow and single-ended tunnel and at the closed end there is a stargate with access to a variety of worlds. Anybody can request an item from another world using the stargate - whatever is requested will be delivered in whatever package the asker wants. However, only a person with a special key can use the stargate to travel to another world and return. Devise a zero-knowledge interactive protocol that will allow a Prover to convince a Verifier that it has the special key without showing it.

(a) Graph Isomorphism: The Prover has a graph G_0 and a permutation π of the vertices of G_0 . Say $G_1 = \pi(G_0)$. Both G_0 and G_1 are public. Permutation π is secret.

commit: Prover generates permutation ρ and chooses $e \in \{0, 1\}$. Prover sends $H = \rho(G_e)$ to the Verifier.

challenge: Verifier chooses $e' \in \{0, 1\}$. Verifier asks Prover to show H is isomorphic to $G_{e'}$.

reveal: If $e = e'$ then set $\phi = \rho$. If $e = 0$ ($e' = 1$) then set $\phi = \rho\pi^{-1}$. Otherwise set $\phi = \rho\pi$. Prover sends ϕ to verifier. Verifier applies ϕ to $G_{e'}$. If it matches H the round succeeds.

(b) Narrow tunnel:

commit: Prover sends plane ticket to Ali Baba who arrives at the end of the tunnel in plain view of the Verifier. The Prover also carries 20 feet of heavy rope.

challenge: Verifier asks the Prover to take Ali Baba to the other end of the tunnel. The Prover also takes the rope. When they are far enough into the tunnel the Verifier yells to the Prover asking either to get a Martian from Mars or to send Ali Baba to Mars.

reveal: The Prover complies. If the Verifier asked the Prover to send Ali Baba to Mars, the Prover ties him up, takes him to Mars via the stargate, and returns. Then the prover invites the Verifier to the end of the tunnel. If Ali Baba is not found by the Verifier and the Verifier asked that he be sent to Mars, or if a Martian is in the tunnel, then the round succeeds. The tunnel is long and narrow and the Verifier sees every inch of it after the challenge so Ali Baba cannot be hiding in it. Hence, in the case Ali Baba was supposed to be sent to Mars, the only possibility is that Ali Baba has left the tunnel via the stargate.

Question 4: Third Party Authentication (15)

1. Describe the Needham-Schroeder protocol - state the givens, as needed

Client A wishes to establish an encrypted communication with Client B using a third party KDC which supplies a one-time session key. Clients A and B share secrets K_A and K_B , respectively, with KDC.

- (a) Client A sends a nonce N_A and request to communicate with B to KDC.
- (b) KDC makes a session key K_{AB} and a ticket $T = K_B\{K_{AB}, A\}$.
- (c) KDC sends $K_A\{N_A, B, K_{AB}, T\}$ to A.
- (d) A decrypts using K_A and finds the ticket T and session key K_{AB} . The nonce is checked to authenticate KDC. It also prevents a replay after K_B is stolen: wait for A to contact KDC. Play back a previous response to A, get ticket encrypted with old K_B . Session key established and attacker impersonates B.
- (e) A makes a nonce N_B and sends T and $K_{AB}\{N_B\}$ to B.
- (f) B decrypts T with K_B and finds K_{AB} . Then B decrypts the nonce N_B with the session key.
- (g) B sends $K_{AB}\{N_B - 1, N_{AB}\}$ to A.
- (h) A decrypts using K_{AB} and finds $N_B - 1$ which it uses to authenticate B. A finds N_{AB} and sends $K_{AB}\{N_{AB} - 1\}$ to B.
- (i) B decrypts using K_{AB} and finds $N_{AB} - 1$ which is used to authenticate A.

2. State what can go wrong.

T can be used even after K_A is changed. Attacker can save $K_A\{N_A, B, K_{AB}, T\}$ until K_A is known and then decrypt to find $T = K_B\{K_{AB}, A\}$. Attacker may send T to B (with the nonce) which B will assume came from A. Thus, the attacker can impersonate A.

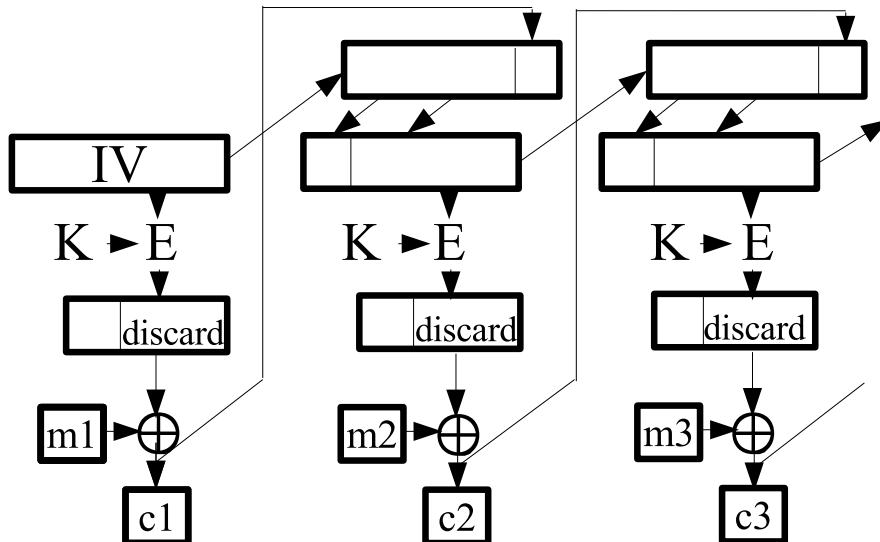
3. State the Ottway-Reese protocol

- (a) A sends $N_{AB}, A, B, K_A\{N_A, N_{AB}, A, B\}$ to B.
- (b) B sends $K_A\{N_A, N_{AB}, A, B\}, K_B\{N_B, N_{AB}, A, B\}$ to KDC.
- (c) KDC makes K_{AB} and sends $N_{AB}, K_A\{K_{AB}, N_A\}, K_B\{K_{AB}, N_B\}$ to B.
- (d) B sends $K_A\{K_{AB}, N_A\}$ to A.
- (e) A decrypts and sends $K_{AB}\{anything\ recognizable\}$ to B.

4. Say how this fixes the problem of Needham-Schroeder

Attacker can save $K_A\{N_A, N_{AB}, A, B\}$ and N_{AB} but this does not contain the session key! Since the session key is required for authentication, the attacker must try to get $K_A\{K_{AB}, N_A\}$ or $K_B\{K_{AB}, N_B\}$. But these are encrypted with the new keys. Hence the attacker cannot decrypt either with old keys. It avails him nothing.

Question 5: Message Encryption (15) - explain, especially with diagrams, how a message can be encrypted so that, if one block is corrupted in transit, at most some number k of blocks will be undecipherable at the receiving end. Your solution should provide for a simple way to insure that no message is encrypted the same way twice with very little bandwidth needed to support this feature. In other words, the length of any transmission (handshake) needed to allow differing encryptions should be independent of message length.



The above depicts a one-time pad started with a random initialization vector (IV). The idea is to encrypt messages by exclusive oring each block with some block of random looking bits. Some bits of the pad block are shifted out on every message block encryption, being replaced cipher bits. If a block is damaged in transit, the pad at that block will be corrupted, but only those bits from the received cipher block. Eventually all those bits will be shifted out.

Question 6: Chinese Remainder Theorem (15) - Devise a way for a person to uniquely represent his or her age as a sequence of numbers and for easy translation of the numbers back to the correct age. The scheme you design should be based on the Chinese Remainder Theorem.

The idea is to choose some prime numbers that multiply to at least the age of the person whose age is being sent. If this is a general scheme, we do not want those numbers (the key) to change. So, to be on the safe side, choose the first four primes 3, 5, 7, 11. The product of these is 1155 so we are safe for centuries! The scheme is to send the four numbers:

$$\langle \text{age mod } 3, \text{ age mod } 5, \text{ age mod } 7, \text{ age mod } 11 \rangle$$

For me, this is $\langle 1, 3, 2, 3 \rangle$.

Question 7: Authentication (10) - Explain how to sign a plaintext message.

Use a public key encryption scheme such as DSS or RSA. This example uses RSA. Like all public key schemes, an RSA user has a private key K_s and a public key K_p . Let m be the plaintext message to sign and assume all parties have a hash algorithm such as SHA. A simple signature involves sending $m, K_s\{\text{SHA}(m)\}$. At least this is good enough for this question. The receiver finds $M_1 = K_p\{K_s\{\text{SHA}(m)\}\}$ and $M_2 = \text{SHA}(m)$. If $M_1 = M_2$ the signature is verified.

Question 8: Bonus Mystery Question - Note that, if n_1, n_2, n_k are prime numbers then, for any $1 \leq i \leq k$, n_i and $n_{\bar{i}} = \prod_{j \neq i} n_j$ are relative prime. That means there exists r and s such that $r * n_i + s * n_{\bar{i}} = 1$. We know that r and s can be found using the GCD algorithm. We also know that this equation can be rewritten $s * n_{\bar{i}} = 1 \pmod{n_i}$. But, since $n_{\bar{i}}$ contains all n_j , $j \neq i$, as factors, it is evenly divisible by all but n_i . Then $s * n_{\bar{i}} = 0 \pmod{n_j}, j \neq i$. From this information show how to find x given a system of equations

$$\begin{aligned} x \pmod{n_1} &= a_1 \\ x \pmod{n_2} &= a_2 \\ &\dots \\ x \pmod{n_k} &= a_k \end{aligned}$$

Answer: Write

$$x = \sum_{i=1}^k a_i * s * n_{\bar{i}} \pmod{n_1 * n_2 * \dots * n_k}$$

Then

$$\begin{aligned} x \pmod{n_1} &= a_1 * 1 \pmod{n_1} + a_2 * 0 + a_3 * 0 + \dots + a_k * 0 = a_1 \\ x \pmod{n_2} &= a_1 * 0 + a_2 * 1 \pmod{n_2} + a_3 * 0 + \dots + a_k * 0 = a_2 \\ &\dots \\ x \pmod{n_k} &= a_1 * 0 + a_2 * 0 + a_3 * 0 + \dots + a_k * 1 \pmod{n_k} = a_k \end{aligned}$$