Security in Computer Networks

Annalise Tarhan

January 11, 2021

1 Using the monoalphabetic cipher in figure 8.3, encode the message "This is an easy problem." Decode the message "rmij'u uamu xyj."

"This is an easy problem." $\rightarrow$ "Uasi si mj cmiw lokngch."
"rmij'u uamu xyj." $\rightarrow$ "Wasn't that fun."

2 Show that Trudy's known-plaintext attack, in which she knows the (ciphertext, plaintext) translation pairs for seven letters, reduces the number of possible substitutions to be checked in the example in section 8.2.1 by approximately 10^9.

This is incorrect. The difference between 26! and 19! is approximately $4.03 * 10^{26}$. This question appears to assume that the difference between 26! and 19! is $26 * 25 * 24 * 23 * 22 * 21 * 20$, which is approximately $3.3 * 10^9$, but $26 * ... * 20 * 19! \neq 26 * ... * 20 + 19!$.

3 Consider the polyalphabetic system shown in figure 8.4. Will a chosen-plaintext attack that is able to get the plaintext encoding of the message "The quick brown fox jumps over the lazy dog." be sufficient to decode all messages? Why or why not?

No, it will not. Although the sentence includes every letter, it does not include every letter in both even and odd positions. For example, the encoded version of the sentence will reveal that when q is in an odd position (where the first letter is in position 0), it will be replaced with a j. It will not reveal that if q is in an even position, it will instead be replaced by a v.

4 Consider the block cipher in figure 8.5. Suppose that each block cipher T_i simply reverses the order of the eight input bits. Further suppose that the 64-bit scrambler does not modify any bits.

4.1 With $n = 3$ and the original 64-bit input equal to 10100000 repeated eight times, what is the value of the output?

00000101 x8

4.2 Repeat, but change the last bit of the original 64-bit input from a 0 to a 1.

00000101 x7, 10000101

4.3 Repeat each of the previous problems but now suppose that the 64-bit scrambler inverses the order of the 64 bits.

10100000 x8
10100001, 10100000 x7

5 Consider the block cipher in figure 8.5. For a given "key" Alice and Bob would need to keep eight tables, each 8 bits by 8 bits. For Alice or Bob to store all eight tables, how many bits of storage are necessary? How does this number compare with the number of bits required for a full-table 64-bit block cipher?

For this problem, assume that the tables are designed so the inputs are implicit and only the outputs need to be stored. For example, the first table entry might correspond to 00000000, the second to 00000001, and so on. This cuts the storage requirement in half.

In the first scenario, eight tables would need to be stored, each with 2^8 entries of 8 bits each. $8 * 2^8 * 8 = 16384$ bits, or 2,048 bytes.

A full-table block cipher would require 2^{64} entries, one for each of the possible 64-bit inputs. $2^{64} * 64 = 1.18 * 10^{21}$ bits, or approximately $1.48 * 10^{11}$ gigabytes.

6 Consider the 3-bit block cipher in table 8.1. Suppose the plaintext is 100100100.

6.1 Initially assume that CBC is not used. What is the resulting ciphertext?

011011011

6.2 Suppose that Trudy sniffs the ciphertext. Assuming she knows that a 3-bit cipher without CBC is being employed, what can she surmise?

That the text consists of three repeated 3-bit elements.

6.3 Now suppose that CBC is used with IV=111. What is the resulting ciphertext?

100110101

7 Using RSA to encode the word "dog" and then apply the decryption algorithm to recover the original plaintext message.

7.1 Choose p=3 and q=11 and encode the word by encrypting each letter separately.

$p = 3 \qquad q = 11 \qquad n = pq = 33 \qquad z = (p-1)(q-1) = 20$
Choose e such that $e < n$ and e is relatively prime to z. Let $e = 7$. Choose d such that $ed \bmod z = 1$. Let $d = 3$. Encode the letters separately, where 'd o g' is represented by '4 15 7' using $c = m^e \bmod n$:
$c_1 = 4^7 \bmod 33 = 16$
$c_2 = 15^7 \bmod 33 = 27$
$c_3 = 7^7 \bmod 33 = 28$

Decrypt the message ‘16 27 28’ using $m = c^d \bmod n$:
$m_1 = 16^3 \bmod 33 = 4$
$m_2 = 27^3 \bmod 33 = 15$
$m_3 = 28^3 \bmod 33 = 7$

7.2 Now, encode the word as one message m.

To combine the letters, represent each in binary and concatenate, then convert to decimal. (Five bits is sufficient for all possible values):
00100 01111 00111 = 4583

Since m must be less than n, p and q must be larger. Let $p = 101$ and $q = 61$. Then $n = 6161$ and $z = 6000$. Let $e = 199$ and $d = 3799$.
$c = 4583^{199} \bmod 6161 = 816$

To verify, decode using n:
$m = c^d \bmod n = 816^{3799} \bmod 6161 = 4583$

8 Consider RSA with $p = 5$ and $q = 11$.

8.1 What are n and z?

$n = pq = 55 \qquad z = (p-1)(q-1) = 40$

8.2 Let $e = 3$. Why is this an acceptable choice?

$3 < n = 55$ and 3 is coprime with $z = 40$

8.3 Find d such that $de = 1 \bmod z$ and $d < 160$.

$d = 27$

8.4 Encrypt the message $m = 8$ using the key (n, e). Let c denote the corresponding ciphertext.

$c = m^e \bmod n = 8^3 \bmod 55 = 17$

9 Diffie-Hellman

9.1 Prove that, in general, Alice and Bob obtain the same symmetric key. That is, prove S=S’.

$S = T_B^{S_A} mod\, p \quad S' = T_A^{S_B} mod\, p \quad T_A = g^{S_A} mod\, p \quad T_B = g^{S_B} mod\, p$

Repeated application of the fact that $[(a \; mod \; n)(b \; mod \; n)] \; mod \; n = (ab) \; mod \; n$ gives the modular exponentiation property: $(a^b) \; mod \; n = [(a \; mod \; n)^b] \; mod \; n$.
$S = T_B^{S_A} mod \; p = (g^{S_B} mod \; p)^{S_A} mod \; p = (g^{S_B})^{S_A} mod \; p = g^{S_A S_B} \; mod \; p$
$S' = T_A^{S_B} mod \; p = (g^{S_A} mod \; p)^{S_B} mod \; p = (g^{S_A})^{S_B} mod \; p = g^{S_A S_B} mod \; p$

9.2 With $p = 11$ and $g = 2$, suppose Alice and Bob choose private keys $S_A = 5$ and $S_B = 12$, respectively. Calucu-late Alice's and Bob's public keys, T_A and T_B.

$T_A = g^{S_A} \; mod \; p = 2^5 \; mod \; 11 = 32 \; mod \; 11 = 10$
$T_B = g^{S_B} \; mod \; p = 2^{12} \; mod \; 11 = 4096 \; mod \; 11 = 4$

9.3 Now calculate S as the shared symmetric key.

$S = T_A^{S_B} mod \; p = 10^{12} \; mod \; 11 = 1$
$S = T_B^{S_A} mod \; p = 4^5 \; mod \; 11 = 1$

9.4 Show how Diffie-Hellman can be attacked by a man-in-the-middle.

An attacker could establish themselves between Alice and Bob, masquerading as each to the other. They would establish two key exchanges, one with Alice and one with Bob. Each message intercepted would be decrypted, re-encrypted, and forwarded.

10 Design a scheme that uses a Key Distribution Center, a server that shares a unique secret symmetric key with each user, to distribute session keys to Alice and Bob within three messages.

This answer assumes Alice generates the session key she will share with Bob. It might be more plausible that Alice simply tells the KDC who she wants to communicate with, and the KDC generates the session key.

10.1 What is the second message?

The KDC sends the session key back to Alice, encrypted with the symmetric key the KDC shares with Bob. This message is encrypted with the symmetric key the KDC shares with Alice.

10.2 What is the third message?

Alice sends the encrypted (with Bob's KDC key) session key to Bob.

11 Compute a third message, different from the two messages in figure 8.8, that has the same checksum as the messages in figure 8.8.

```
I O U 5
5 0 . 5
5 B O B
```

12 Suppose Alice and Bob share two secret keys: an authentication key S_1 and a symmetric encryption key S_2. Augment figure 8.9 so that both integrity and confidentiality are provided.

After a hash of the message+S_1 is appended to the message, the whole thing should be encrypted with S_2 before being sent. Obviously, Bob would need to decrypt it before checking the hash.

13 In the BitTorrent protocol, it is critical for peers to have a way to verify the integrity of a block. Assume that when a peer joins a torrent, it initially gets a .torrent file from a fully trusted source. Describe a simple scheme that allows peers to verify the integrity of the blocks.

The .torrent file should include a list of all the blocks as well as their hashes. This way, when a peer receives a block, it can hash the received block and compare the result to the hash in the .torrent file.

14 The OSPF routing protocol uses a MAC rather than digital signatures to provide message integrity. Why do you think a MAC was chosen over digital signatures?

Digital signatures, created by encrypting messages with a private key, are computationally expensive. A MAC guarantees message integrity and is appropriate when confidentiality isn't critical.

15 Consider the authentication protocol in figure 8.18. Give a scenario by which Trudy, pretending to be Alice, can now authenticate herself as Bob to Alice.

Trudy initiates with an "I am Alice" message, to which Bob responds with a nonce, R. Trudy then waits for Bob to send an "I am Bob" message, to which she replies with the same R. She waits for Bob to return the nonce encrypted with the symmetric key and sends it back as a response to to Bob's original message, 'authenticating' herself as Alice.

16 Consider whether a nonce and public key cryptography can be used to solve the end-point authentication problem with the following protocol. Suppose certificates are not used. Describe how Trudy can become a "woman-in-the-middle" by intercepting Alice's messages and then pretending to be Alice to Bob.

This protocol is only as strong as Bob's confidence that the public key he is applying is actually Alice's. Anyone could call themselves Alice and encrypt the nonce using a private key, and if they manage to convince Bob that the corresponding public key belongs to Alice, he won't know the difference.

17 Figure 8.19 shows the operations that Alice must perform with PGP to provide confidentiality, authentication, and integrity. Diagram the corresponding operations that Bob must perform on the package received from Alice.

When Bob receives the email, his first step is to separate out the symmetric key, which will be encrypted with his public key. After he decrypts it using his private key, he can use the key to decrypt the rest of the message, which will have two parts: the message and an encrypted hash of the message. He will use Alice's public key to decrypt the hash and compare it to his own calculated hash of the received message.

18 Suppose Alice wants to send an email to Bob, who has a public, private key pair that Alice has verified using his certificate. But Alice does not have a public, private key pair. Alice and Bob (and the entire world) share the same hash function.

18.1 In this situation, is it possible to design a scheme so that Bob can verify that Alice created the message?

No. There is nothing uniquely identifying Alice. Anything she could do, an attacker could as well, short of them meeting face-to-face.

18.2 Is it possible to design a scheme that provides confidentiality for sending the message from Alice to Bob?

Yes, all Alice would have to do is encrypt the message using Bob's public key.

19 Consider the Wireshark output below for a portion of an SSL session.

19.1 Is Wireshark packet 112 sent by the client or server?

Client

19.2 What is the server's IP address and port number?

216.75.194.220
443

19.3 Assuming no loss and no retransmissions, what will be the sequence number of the next TCP segment sent by the client?

79+204=283

19.4 How many SSL records does Wireshark packet 112 contain?

Three

19.5 Does packet 112 contain a Master Secret or an Encrypted Master Secret or neither?

It contains an encrypted Pre-Master Secret

19.6 Assuming that the handshake type field is 1 byte and each length field is 3 bytes, what are the values of the first and last bytes of the Master Secret?

bc 29

19.7 The client encrypted handshake message takes into account how many SSL records?

All cryptographic algorithms it supports

19.8 The server encrypted handshake message takes into account how many SSL records?

Three

20 We showed that without sequence numbers, a man-in-the-middle can wreak havoc in an SSL session by interchanging TCP segments. Can they do something similar by deleting a TCP segment? What do they need to do to succeed at the deletion attack?

Deleting a TCP segment will cause the receiver to indicate to the sender that it has not received an expected segment. The sender will re-send until it receives one with the missing sequence number. The only way to avoid detection would be to alter all of the sequence numbers for the duration of the TCP connection.

21 Suppose Alice and Bob are communicating over an SSL session. Suppose an attacker, who does not have any of the shared keys, inserts a bogus TCP segment into a packet stream with correct TCP checksum and sequence numbers (and correct IP addresses and port numbers.) Will SSL at the receiving side accept the bogus packet and pass the payload to the receiving application? Why or why not?

No. Its MAC will still be incorrect.

22 True/False

22.1 When a host in 172.16.1/24 sends a datagram to an amazon.com server, the router R1 will encrypt the datagram using IPsec.

False. (Unless the host is part of Amazon's network)

22.2 When a host in 172.16.1/24 sends a datagram to a host in 172.16.2/24, the router R1 will change the source and destination address of the IP datagram.

True/False. The original IP datagram will not be changed, only encrypted. The new datagram will have different IP addresses - those of the external router interfaces.

22.3 Suppose a host in 172.16.1/24 initiates a TCP connection to a web server in 172.16.2/24. As part of this connection, all datagrams sent by R1 will have protocol number 50 in the left-most IPv4 header field.

True. The encrypted datagram's version number will indicate it is an IPsec datagram using the ESP protocol.

22.4 Consider sending a TCP segment from a host in 172.16.1/24 to a host in 172.16.2/24. Suppose the acknowledgement for this segment gets lost, so that TCP resends the segment. Because IPsec uses sequence numbers, R1 will not resend the TCP segment.

False. A resent TCP segment will have a different IPsec sequence number.

23 Consider the example in figure 8.28. Suppose Trudy is a woman-in-the-middle, who can insert datagrams into the stream of datagrams going from R1 to R2. As part of a replay attack, Trudy sends a duplicate copy of one of the datagrams sent from R1 to R2. Will R2 decrypt the duplicate datagram and forward it into the branch-office network? If not, describe in detail how R2 detects the duplicate datagram.

No, IPsec uses sequence numbers to prevent replay attacks. It will see that the sequence number is lower than what it expects and drop it. If the sequence number is altered, it will fail the integrity check and will also be dropped.

24 Consider the following pseudo-WEP protocol.

24.1 We want to send the message m=10100000 using the IV=11 and using WEP. What will be the values in the three WEP fields?

IV: 11   Message: 0101 1010   ICV: 1010

24.2 Show that when the receiver decrypts the WEP packet, it recovers the message and the ICV.

The received message is 11010110101010. The receiver knows the first two digits are the IV and the last four are the ICV, so it can isolate the message bits and identify the keystream associated with IV 11, 11111010.
11111010 XOR 01011010 = 10100000
The receiver verifies the message's integrity by comparing 1010 XOR 0000 = 1010 with the ICV, which is the same.

24.3 Suppose Trudy intercepts a WEP packet (not necessarily with the IV=11) and wants to modify it before forwarding it to the receiver. Suppose Trudy flips the first ICV bit. Assuming that Trudy does not know the keystreams for any of the IVs, what other bit(s) must Trudy also flip so that the received packet passes the ICV check?

The first ICV bit is based on the first and fifth encrypted message bits, so she would have to flip one of those.

24.4 Justify your answer by modifying the bits in the WEP packet in the first problem, decrypting the resulting packet, and verifying the integrity check.

Flip the first message bit and the first ICV bit. 11010110101010
11X1011010X010
11110110100010
IV: 11   Message: 11011010   ICV: 0010

First, XOR the message with the keystream.
11111010 XOR 11011010 = 00100000
Then, XOR the first four and last four message bits and compare to ICV bits.
0010 XOR 0000 = 0010 = ICV bits

25 Provide a filter table and a connection table for a stateful firewall that is as restrictive as possible but accomplishes the following goals.

Action	Source Address	Dest Address	Protocol	Source Port	Dest Port
allow	222.22/16	outside 222.22/16	TCP	any	23
allow	outside 222.22/16	222.22.0.12	TCP	any	any
deny	any	any	any	any	any

Source Address	Dest Address	Source Port	Dest Port
37.96.87.123	222.22.1.7	12699	80
199.1.205.23	222.22.93.2	37654	80
203.77.240.43	222.22.65.143	48712	80

26 Suppose Alice wants to visit the Web site activist.com using a TOR-like service. This service uses two non-colluding proxy servers, Proxy1 and Proxy2. Alice first obtains the certificates (each containing a public key) for Proxy1 and Proxy2 from some central server. Denote K1+(), K2+(), K1-(), and K2-() for the encryption/decryption with public and private RSA keys.

26.1 Provide a protocol (as simple as possible) that enables Alice to establish a shared session key S_1 with Proxy1. Denote $S_1(m)$ for encryption/decryption of data m with the shared key S_1.

Alice sends $K1+(S_1)$ to Proxy1, which decrypts it using its private key K1-. Proxy1 sends $S_1(ACK)$.

26.2 Provide a protocol (as simple as possible) that allows Alice to establish a shared session key S_2 with Proxy2 without revealing her IP address to Proxy2.

Alice sends $S_1(K2+(S_2))$ to Proxy1.
Proxy1 forwards $K2+(S_2)$ to Proxy2, which decrypts it using its private key.
Proxy2 sends $S_2(ACK)$ to Proxy1, which forwards it to Alice.

26.3 Assume now that shared keys S_1 and S_2 are established. Provide a protocol (as simple as possible and not using public-key cryptography) that allows Alice to request an html page from activist.com without revealing her IP address to Proxy2 and without revealing to Proxy1 which site she is visiting. Your protocol should end with an HTTP request arriving at activist.com.

Alice sends $S_1(S_2(m))$ to Proxy1, where m is the HTTP request.
Proxy1 sends $S_2(m)$ to Proxy2, after decrypting $S_1(S_2(m))$ using S_1.
Proxy2 decrypts $S_2(m)$ using S_2 and sends the plaintext HTTP request to activist.com.