Question 1 (1 mark)

What are the two main drawbacks with one-time pad?

Suggested answer:

1. There is the practical problem of making large quantities of random keys. Supplying truly random characters in large quantities is a significant task.
2. The second problem is the distribution of this huge key. For every message to be sent, a key of equal length is needed by both sender and receiver. Hence, a mammoth key distribution problem exists.

Question 2 (1 mark)

Describe a pratical use for a MAC.

Suggested answer:

A MAC can be used to authenticate the sender of a message and verify that the data integrity of a message is maintained. Given a symmetric cryptosystem, a sender (E.g. Alice) computes the MAC $\mu = C_k(m)$ and sends (m, μ) to an intended reciptient (e.g. Bob.) Upon receipt of the pair, Bob checks whether $\mu = C_k(m)$. If so, the authentication succeeds; otherwise, the authentication fails. In a case when a message is altered, the check will fail as well because the authentication tag will be different. By the properties of a MAC, an attacker without knowing the key k, it is very unlikely that the attacker is able to produce a valid pair (m, μ) that will be accepted by Bob.

Question 3 (1 mark)

In cryptography, in particular digital signature context, explain the term nonrepudiation.

Suggested answer:

Nonrepudiation provides protection against denial by one of the entities involved in communication of having participated in all or part of the communication. It is a service in digital signature that provides proof that the message was sent by the specified party as well as proof that the message was received by the specified party.

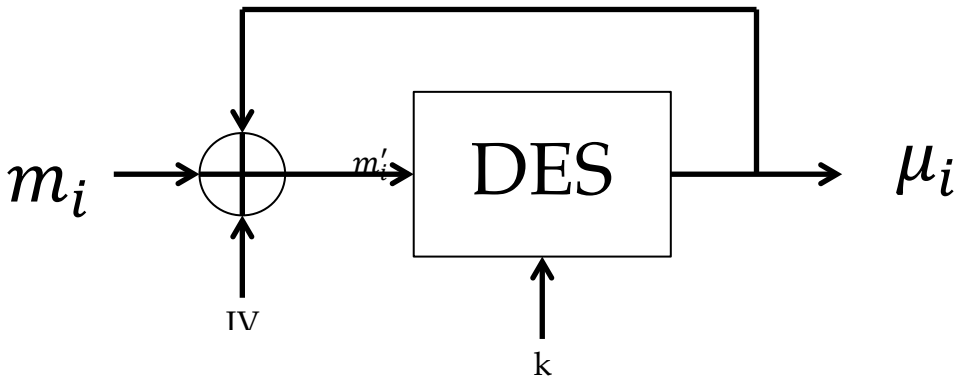
Question 4 (2 marks)

Describe how to build a MAC system using a block cipher in CBC (chain-block cipher) or CFB (cipher-feedback Block) chain mode.

Suggested answer:

A MAC can be built by using a block cipher in CBC or CFB chaining mode as follow:

- Consider a 64-bit MAC
- Let $m = m_1 \cdots m_n$ and $\mu = \mu_1 \cdots \mu_n = E_k(m_1 \cdots m_n)$



$$m'_1 = m_1 \oplus IV$$
$$m'_i = m_i \oplus \mu_{i-1} \quad for\ i >$$
$$MAC_k(m)$$

In CBC or CFB mode, each ciphertext block μ_i depends on both the current message block m_i and on the previous ciphertext block μ_{i-1}. Hence, the last ciphertext block μ_n depends on the entire message $m_1 \cdots m_n$, as desired. We can then define the $MAC_k(m) = \mu_n$.

m	μ_n

Question 5 (4 marks)

Compute the following by demonstrating the step-by-step calculation correctly.

(i) Compute $21^{221} mod\ 123$ using fast exponentiation algorithm discussed in lecture and/or tutorial. Show all steps.
(ii) Compute $3037^{-1} mod\ 4051$
(iii) Using the extended Euclidean algorithm, find the multiplicative inverse of 1234 and 4321.
(iv) Find x such that the equation 18x = 11 mod 19 is satisfied?

Suggested answer:

(i) 21^{221} mod 123 = 21 mod 123 = 21.

The student is expected to use fast exponentiation to compute.

221 (in decimal) = 11011101 (in binary)

	$2^7=128$	$2^6=64$	$2^5=32$	$2^4=16$	$2^3=8$	$2^2=4$	$2^1=2$	$2^0=1$
221 =	1	1	0	1	1	1	0	1
21^{221} mod 123	21^{128} mod 123	21^{64} mod 123		21^{16} mod 123	21^{8} mod 123	21^{4} mod 123		21^{1} mod 123
	78 mod 123	**18 mod 123**	51 mod 123	**57 mod 123**	**78 mod 123**	**18 mod 123**	72 mod 123	**21 mod 123**
	19^{221} mod 123 = 78 × 18 × 57 × 78 × 18 × 21 mod 123 = **21 mod 123**							

$Thus\ 21^{221} mod\ 123 = 21\ mod\ 123 = 21.$

(ii) Compute 3221^{-1} mod 4019.

Suggested answer:

$3221^{-1} mod\ 4019 = -277\ mod\ 4019 = 3742$

The student is expected to use Extended Euclidean Algorithm to compute as follow:

n1	n2	r	q	a1	b1	a2	b2
4019	3221	798	1	1	0	0	1
3221	798	29	4	0	1	1	-1
798	29	15	27	1	-1	-4	5
29	15	14	1	-4	5	109	-136
15	14	1	1	109	-136	-113	141

(iii) gcd(4321, 1234) = 1 = (309) (4321) + (-1082) (1234)

The student is expected to use Extended Euclidean Algorithm to compute.

n1	n2	r	q	a1	b1	a2	b2
4321	1234	619	3	1	0	0	1
1234	619	615	1	0	1	1	-3
619	615	4	1	1	-3	-1	4
615	4	3	153	-1	4	2	-7
4	3	1	1	2	-7	-307	1075
3	1	0	3	-307	1075	309	-1082

(309)(4321) + (-1082)(1234) = 1

Hence, the multiplicative inverse of 1234 mod 4321 is -1082.

(iv) Suggested solution:

$18x = 11 \; mod \; 19$
$x = \frac{11}{18} \; mod \; 19$
$x = 11 \; (18)^{-1} \; mod \; 19$
$x = (11)(-1) mod \; 19$
$x = (11)(18) mod \; 19 = 198 \; mod \; 19 = 8$

The verification is not mandatory.
$Verify$:
$18(8) = 11 \; mod \; 19$
$144 = 11 \; mod \; 19$
$11 = 11$

Question 6 (2 marks)

DSA is a public key signature algorithm and is specified by the NIST's Digital Signature Standard. DSA is used to create a small, publicly verifable signature σ for a given message M. Describe how a message M is signed and verified using DSA. Show that the signing and verification algorithms are correct.

Suggested answer:

In Digital Signature Algorithm (DSA), the private signing key is $u \in Z_q$, and the verification key is $y = g^u \bmod p$.

The message m is signed as followed:
$r = (g^k \bmod p) \bmod q$
$s = k^{-1}(h(m) + ur) \bmod q,$
$where\ u \in Z_q\ is\ the\ private\ signing\ key.$

The signed message is (h(m), (r,s))

To verify the message, the recepient computes $\frac{r \times h(m)}{s}$ in the following three steps:

$w = s^{-1} mod\ q$
$u_1 = h(m)w \bmod q$
$u_2 = rw \bmod q$

and verify that

$r =?\ (g^{u_1} y^{u_2} \bmod p) \bmod q$

Correctness of the algorithm:

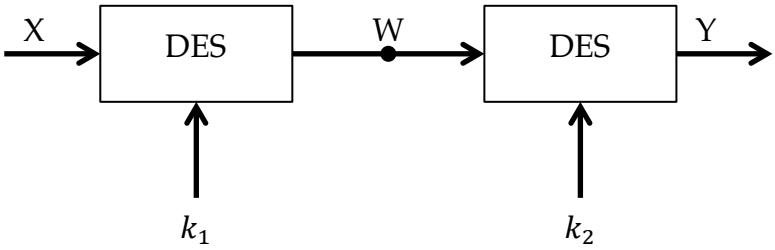
$r =?\ (g^{u_1} y^{u_2} \bmod p) \bmod q$
$=?\ \left(g^{h(m)w}\ y^{rw} \bmod p\right) \bmod q, \quad since\ y = g^u \bmod p$
$=?\ \left(g^{h(m)w}\ (g^u)^{rw} \bmod p\right) \bmod q$
$=?\ \left(g^{h(m)w}\ g^{urw} \bmod p\right) \bmod q$
$=?\ \left(g^{h(m)w+urw} \bmod p\right) \bmod q$
$=?\ \left(g^{w(h(m)+ur)} \bmod p\right) \bmod q$
$=?\ \left(g^{s^{-1}(h(m)+ur)} \bmod p\right) \bmod q$
$=?\ \left(g^{\left(k(h(m)+ur)^{-1}\right)(h(m)+ur)} \bmod p\right) \bmod q$
$=?\ (g^k \bmod p) \bmod q \quad since\ \ r = (g^k mod\ p) mod\ q$
$r = r \bmod q$

Question 7 (2 marks)

One issue with DES is the key size of 56-bit too short. To increase the key size, one approach is to double encrypt, and hence effectively increase the key size to 112 bits. However, this approach is not really the same as if there were a single DES of 112-bit. Explain why is it much less secure to impplement a double DES.

Suggested answer:

With double DES, the plaintext *Y* is encrypted two times, with different keys; $Y \mapsto E_{k2}\big(E_{k1}(X)\big)$.



Since the plaintext *X* is encrypted two times with k_1 and k_2, the key-space size of double DES is bascially $2^{56}\times 2^{56} = 2^{112}$. However, double DES can be attacked by "meet-in-the-middle", which takes not much more time than 2^{57}. With the "meet-in-the-middle" attack, the attacker knows a pair of plaintext and ciphertext (X, Y). The attacker tries to **decrypt** all 2^{56} values of permutations (identified as z'') and produces a list (w_i'', z_i'') where $w_i'' = DES_{z_i'}^{-1}(Y)$, and z_i'' is all 2^{56} values of permutation. The attacker then tries all 2^{56} values of permutation (identified as z') to **encrypt** X and produces a list (w_i', z_i') where $w_i' = DES_{z_i'}(X)$. The attacker then sorts the lists (w_i', z_i') in w_i' assending order, and (w_i'', z_i'') *in* w_i'' ascending order. The attacker is able to compare these two lists and determine the keys. If a match $(w_i' = w_i'')$ is found, then z_i'' is k_2 and z_i' is k_1. With this attack, the number of operations is equal to the sum of the number of decryption $\left(w_i'' = DES_{z_i'}^{-1}(Y)\right)$ and the number of encryption $\left(w_i = DES_{z_i'}^{-1}(Y)\right)$. Hence the total number of operation required equals $2^{56} + 2^{56} = 2^{57}$. In order words, the effective key space is only 2^{57} bits and not 2^{112} bits.

Question 8 (2 marks)

(i) Consider the RSA algorithm where p = 7 and q = 19. What are the private and public keys?
(ii) Suppose Adam generates an RSA key pair with modulus $n = p \times q$, but his private key is stolen (compromised). Rather than generating a new modulus, Adam decides to create a new key pair using the same modulus. Show to Adam that this is not safe.

Suggested solution:

(i) Generating the private and public key.

- Compute the modulus $n = p \times q = 7 \times 19 = 133$.
- Compute $\Phi(n) = (p-1)(q-1) = 6 \times 18 = 108$.
- Randomly choose a value e = 37 such that $\gcd(e, \Phi(n)) = 1$.

Check if gcd(37, 108) = 1?

n1	n2	r	q	a1	b1	a2	b2
108	37	34	2	1	0	0	1
37	34	3	1	0	1	1	-2
34	3	1	11	1	-2	-1	3
3	1	0	3	-1	3	12	-35

Thus gcd(37,108) = 1

- Thus d can be derived from the above extended GCD computation as follow:

$d = -35 \ mod \ 108 = 73.$

We can verify the correctness of e and d by computing $e \times d \ mod \ 108 = 37 \times 73 \ mod \ 108 = 2701 \ mod \ 108 = 1$. *Hence, e = 37 and d = 73.*

The public key pair is (e, n) = (37, 133), and the private key pair is (d, n) = (73, 133).

(ii)

The problem Adam will encounter is known common modulus attack. Even though Adam generates a new key, let's say (e'_A), but since the new key is generated using the same modulus, if the attacker now captures Adam's ciphertext that is generated using the new key, the attacker can use the old ciphertext and new ciphertext to find the new message *m* as follow:

The attacker knows both the ciphertexts – ciphertext that was generated using the stolen key, $C_A \equiv m^{e_A} \ mod \ n$, and the new ciphertext $C'_A \equiv m^{e'_A} \ mod \ n$. The attacker also know that $\gcd(e_A, e'_A) = 1$. Thus the attacker can compute the inverse multiplication of $e_A \ and \ e'_A$ using extended Euclidean Algorithm to get $(e_A)(a) + (e'_A)(b) = 1$.

The attacker then computes $(C_A)^a \times (C'_A)^b mod\ n$, which the attacker can obtain the message m as follow:

$$= (m^{e_A})^a \cdot \left(m^{e'_A}\right)^b \ mod\ n$$
$$= (m^{e_A \cdot a}) \cdot \left(m^{e'_A \cdot b}\right) mod\ n$$
$$= m^{(e_A)(a)+(e'_A)(b)} \ mod\ n$$
$$= m\ mod\ n$$
$$= m$$

END OF TEST