Closed book. Closed notes. No electronic device.

Conventions

$[sk, pk]$	public-key pair
$\mathsf{E_P}(pk, x)$	public-key encryption of x with public key pk
$\mathsf{Sgn}(sk, x)$	public-key signature of x with secret key sk
$\mathsf{E_{AES}}(s, x)$	symmetric-key AES block encryption with key s
$\mathsf{D_{AES}}(s, x)$	symmetric-key AES block decryption with key s
$\mathsf{E}(s, x)$	symmetric-key CBC-AES (CBC mode using AES) encryption of x with key s
$\mathsf{D}(s, x)$	symmetric-key CBC-AES (CBC mode using AES) decryption of x with key s
$\mathsf{MAC}(s, x)$	symmetric-key MAC (ECBC) of x with key s
$\mathsf{H}(x)$	SHA-256 hash function of x
$\mathsf{HMAC}(k, x)$	HMAC of x using key k and H
$\mathsf{H}^k(x)$	k-fold hash: $\mathsf{H}(\mathsf{H}(\ldots\mathsf{H}(x)\ldots))$

1.

AES
Authenticity
Block cipher
Collision resistant
CA
Certificate
Certificate chain
CRL
Confidentiality
DES
Diffie-Helman
Dictionary attack
Euclid's algorithm
Eucler's theorem
Existential forgery
HMAC
Integrity
KDC
MAC
Mode
CBC
CTR
OFB
Non-repudiation
OCSP
OCSP stapling
One-time pad
Pre-image resistant
PGP
PKCS
PKI
RSA
Session key
Signing
Ticket
Totient function
Verification

For each description below, give **one** term that *best* describes it. In many cases, but not all, the term will be in the table at left. In each case, give only **one** answer of **at most** 4 words (otherwise you get zero).

1. This ensures that the data can be read only by the intended receiver.
2. This ensures that any modification to the data is detected by the intended receiver.
3. This ensures that data received was sent by the specified sender.
4. This ensures that a third party can verify that the data was sent by the specified sender.
5. This kind of crypto uses different keys for encryption and decryption.
6. The attack model in which the attacker has access to an encryption oracle but not a decryption oracle.
7. This symmetric block cipher supports only one key size.
8. This symmetric block cipher supports multiple key sizes.
9. This defines how to use a block cipher on arbitrary-size data.
10. This ensures that encrypting the same message more than once results in different ciphertext.
11. This mode allows the block cipher encryption function calls to be made before the data is available.
12. This mode allows encryption to be done in parallel.
13. This mode allows a hash function to be used for encryption of arbitrary-size data.
14. The property of a hash function that makes it hard to find a message m that hashes to a given number.
15. This is the standard method for generating MACs from block ciphers.
16. This is the standard method for generating MACs from hash functions.
17. This is the set of integers in $1, \cdots, n-1$ that are relatively prime to n.
18. This is the number of integers in $1, \cdots, n-1$ that are relatively prime to n.
19. What do we need to efficiently compute the number of integers in $1, \cdots, n-1$ that are relatively prime to n.
20. What allows us to efficiently compute $\phi(n)$.
21. An attack that goes through a set of candidate passwords.
22. This means that after a session-key is forgotten by the principals that used it, no one can decrypt data encrypted with that key.
23. A public-key infrastructure that is not hierchical.

2. Alice has an account with a server. The server makes her change her password every few months, to which Alice just increments a number in her password, e.g., `pwd1`, `pwd2`, $\cdots$.

Why does the server not complain that the new password is very much like her old one?

3. Let $[e, n]$ be the RSA public key of a server. Suppose someone gives you the prime factors of n, say p and q. Can you obtain the private key $[d, n]$? If not, explain briefly. If yes, briefly give the steps.

4. A hash function $H()$ generates a 256-bit hash. How many random messages on average would one have to hash before finding two distinct messages that hash to the same value.

5. Is a strong password significantly better than a weak password against an online dictionary attack.

6. Is a strong password significantly better than a weak password against an offline dictionary attack.

7. A server has N users and stores hashes of the users' passwords in a map P indexed by user id. Specifically, for user u, the entry $P(u)$ is $\mathsf{H}^4(p)$, where p is u's password.

a. What would the entry be if you "salted" the entries.
b. If N is 20, does salting improve security significantly? Explain briefly.

8. Let x be an element of Z_n and y denote its multiplicative-inverse-mod-n.

a. When does y exist?
b. Give the equation that x and y satisfy.

9. Let message msg consist of blocks $[m_1, \cdots, m_n]$. Let $[c_0, c_1, \cdots, c_n]$ be the ciphertext resulting from encrypting msg using AES with key k in some mode.

a. Assume CBC mode. Give c_i as a function of the plaintext, key, $\mathsf{E}_{\mathsf{AES}}()$ and/or $\mathsf{D}_{\mathsf{AES}}()$, for $i = 1, \cdots, n$.
b. Repeat part a for CTR mode.

10. For an arbitrary-size message msg, let $M(k, msg)$ denote the last block of CBC-AES encryption using key k and IV$= 0$. Is $M(k, msg)$ a secure MAC. If you answer yes, explain briefly. If you answer no, give a counter example.

11. Why is a random pad needed for RSA encryption of a msg.

12. Server B has a well-known fixed IP address and TCP port, and no other service can use that address and port. User A shares a password, say pwd with B. A connects as follows:

1. Establish a shared key s with a standard (not authenticated) Diffie-Helman.
2. Send ["A", pwd] encrypted with key s to the server.
3. B authenticates the user if the password matches.

a. Assume an attacker that can only eavesdrop on messages. Does the above ensure that key s is securely shared between A and B. If you answer "no", give an attack. If you answer "yes", explain.
b. Repeat part a for an attacker that can eavesdrop and tamper with messages (intercept and change them).

13. A domain has a CA X, which is the trust anchor for the domain's users.

a. What steps does a new user, say A, take upon joining the domain.
b. What steps are taken when a user, say A, leaves the domain.
c. What steps are taken when X's secret key is exposed.

14. The users in domain `x.com` has a CA X as trust anchor. The users in domain `y.com` has a CA Y as trust anchor. One day, `x.com` and `y.com` are acquired by `z.com`, which has a CA Z as trust anchor.

List the steps that will allow users in all three domains to talk to each other. Minimize the number of new certificates that are issued.

15. A domain's authentication is handled by KDC X.

a. What steps does a new user, say A, take upon joining the domain.
b. What steps are taken when a user, say A, leaves the domain.
c. What steps are taken when X's key (used to encrypt its user-key table) is exposed.

16. A domain's authentication is handled by KDC X. Tickets can have long expiry times. Consider the following:

1. A gets a post-dated ticket T from the KDC to interact with server B.
2. B changes its master key with the KDC.
3. A presents T to B.

What is the problem here? What is a solution?

17. Authentication protocols

This problem has independent parts. Each part describes an authentication protocol that Alice (A) initiates to send a message m to Bob (B), and then asks one or more questions. Answer each question by circling *YES* or *NO*.

The first question asks whether the protocol is ***broken***, i.e., does it require Alice or Bob to do something they cannot (e.g., decrypt a message without the key).

- If you answer *YES* to the first question, *skip the remaining questions in that part*. You will get full marks if your answer is correct, and zero otherwise.
- If you answer *NO* to the first question, *attempt to answer all the remaining questions in that part*. For each question, you will get 1 point if your is answer correct, zero points if you do not answer, and **–1 point if your answer is wrong**.

Alice has public-key pair $[sk_A, pk_A]$. Bob has pk_A.
Bob has public-key pair $[sk_B, pk_B]$. Alice has pk_B.
Alice and Bob also share a symmetric key S_{AB}

Unless otherwise stated, symmetric keys are strong, and the attacker can eavesdrop and tamper with messages.

17.1.

A: generate a new symmetric key s
send $[\mathsf{E_P}(pk_B, s), \mathsf{E}(s, m), \mathsf{Sgn}(sk_A, \mathsf{H}(m))]$ B: receive message; extract m

- Is the protocol broken? *yes* *no*
- Is confidentiality satisfied for m? *yes* *no*
- Is integrity satisfied for m? *yes* *no*
- Is non-repudiation satisfied for m? *yes* *no*
- Is perfect forward secrecy satisfied for m? *yes* *no*
- If s comes from a password, is m vulnerable to an offline dictionary attack? *yes* *no*

17.2.

A: generate random c_A
  $n_A \leftarrow \mathsf{E}(S_{AB}, c_A)$
  send $[n_A]$

B: receive message
  $c_A \leftarrow \mathsf{D}(S_{AB}, n_A)$
  $r_A \leftarrow \mathsf{E}(S_{AB}, c_A + 1)$
  generate random c_B
  $n_B \leftarrow \mathsf{E}(S_{AB}, c_B)$
  send $[r_A, n_B]$

A: receive message
  if $(\mathsf{D}(S_{AB}, r_A) \neq c_A + 1)$ "FAIL"
  $c_B \leftarrow \mathsf{D}(S_{AB}, n_B)$
  $r_B \leftarrow \mathsf{E}(S_{AB}, c_B + 1)$
  session key $x \leftarrow c_A \oplus c_B$
  send $[r_B, \mathsf{E}(x, m), \mathsf{MAC}(x, m)]$

B: receive message
  if $(\mathsf{D}(S_{AB}, r_B) \neq c_B + 1)$ "FAIL"
  extract msg m

- Is the protocol broken? *yes* *no*
- Is confidentiality satisfied for m? *yes* *no*
- Is integrity satisfied for m? *yes* *no*
- Is non-repudiation satisfied for m? *yes* *no*
- Is perfect forward secrecy satisfied for m? *yes* *no*
- If S_{AB} comes from a password, is m vulnerable to an offline dictionary attack: *yes* *no*

17.3.

A: generate Diffie-Helman parameters p and g
  generate random x
  $T_A \leftarrow g^x \text{mod-}p$
  send $[T_A]$

B: receive message
  generate random y
  $T_B \leftarrow g^y \text{mod-}p$
  $s \leftarrow {T_A}^y$
  send $[T_B]$

A: receive message
  $s \leftarrow {T_B}^x$
  send $[E(s, m)]$

B: receive message
  extract msg m

- Is the protocol broken? *yes* *no*
- Is confidentiality satisfied for m? *yes* *no*
- Is integrity satisfied for m? *yes* *no*
- Is non-repudiation satisfied for m? *yes* *no*
- Is perfect forward secrecy satisfied for m? *yes* *no*

17.4. Repeat 17.3 assuming the attacker can only eavesdrop (but not tamper).

- Is the protocol broken? *yes* *no*
- Is confidentiality satisfied for m? *yes* *no*
- Is integrity satisfied for m? *yes* *no*
- Is non-repudiation satisfied for m? *yes* *no*
- Is perfect forward secrecy satisfied for m? *yes* *no*