Queen Mary
University of London

Friday 23rd February 2018 9:00 to 11:00 AM

**ECS655U-ECS775P** **Security Engineering** **Midterm (20%)** **Duration:** 120 minutes

# SOLUTIONS AND MARKING SCHEME

**YOU ARE NOT PERMITTED TO READ THE CONTENTS OF THIS QUESTION PAPER UNTIL INSTRUCTED TO DO SO BY AN INVIGILATOR.**

**Instructions:** Enter your name and ID# on this page right away. This paper contains FOUR questions. **Answer ALL questions**. Cross out any answers that you do not wish to be marked.

Non-programmable calculators are permitted. Please state on your answer book the name and type of machine used.

Possession of unauthorised material at any time when under examination conditions is an assessment offence and can lead to expulsion from QMUL. Check now to ensure you do not have any notes, mobile phones, smartwatches or unauthorised electronic devices on your person. If you do, raise your hand and give them to an invigilator immediately.

**Exam papers must not be removed from the exam room.**

Full Name: .....

Student ID Number: .....

| Question | Points | Score |
|---|---|---|
| 1 | 25 | |
| 2 | 30 | |
| 3 | 30 | |
| 4 | 30 | |
| Total: | 115 | |

Leave this table blank.

Examiners: Dr Arman Khouzani, Dr Na Yao

**Question 1**

(a) Provide a short definition for each of the following security services:

**[5 marks — basic]**

- ▷ Confidentiality:
- ▷ Data integrity:

**Solution:**

- ▷ Confidentiality: Ascertaining that only authorised entities can view a data.
- ▷ Data integrity: Ascertaining that a piece of data is not modified (accidentally or maliciously) after its creation by a specific entity.

---

*Marking scheme:*

- ▷ 3 points for each, capped at 5 in total.

(b) Does "data integrity" require "confidentiality"? Support your answer with a brief explanation.

**[5 marks — medium]**

**Solution:** No. The adversary may be able to get the message, but as long as modifications are detected, we will have data integrity. For instance Message Authentication Codes (MACs) can provide data integrity but no confidentiality.

---

*Marking scheme:*

- ▷ 2 points for a correct identification, 3 points for a clear explanation (either providing an example like MAC or a logical explanation.)

(c) For each of the following security service, determine whether or not a "digital signature" scheme can provide it. Support your claim with a brief explanation.

**[5 marks — basic]**

- ▷ Confidentiality:
- ▷ Detection of Accidental Changes (e.g. due to channel noise):

**Solution:**

- ▷ Confidentiality: No; the main security service of digital signatures is non-repudiation. That is, a guarantee that a message is indeed created by an entity (at some point in the past) and that entity cannot deny it (because no-one else could have created it). In particular, the fact that the message can be seen or not by adversaries is irrelevant. For instance, in the "digital signature schemes with appendix", the digital signature is added to the message and sent.

▷ Detection of Accidental Changes (e.g. due to channel noise): Yes; if we have non-repudiation, it means we also have data origin authentication, which in turn implies we also have data integrity, which in turn implies we can detect any changes (accidental or malicious).

---

*Marking scheme:* 1 point for each correct identification (yes/no). 2 points for each correct explanation. Capped at 5 points in total.

(d) Provide two similarities and two differences between "Message Authentication Codes (MAC)" and "digital signatures".

**[10 marks — medium]**

▷ Two similarities:

▷ Two differences:

**Solution:**

▷ Two similarities: many possibilities, e.g.

- Both provide data integrity and data origin authentication;
- In combination with a freshness mechanism, both can be used for entity authentication;
- both create get a message of arbitrary size, plus a key, and create a fixed length output; . . .

▷ Two differences: again, many possible answers, e.g.

- a MAC scheme cannot directly provide non-repudiation, while digital signatures provide non-repudiation directly;
- in MAC, the key to create a digest from a message and the key to verify it is the same, while in a typical digital signatures, the signing key and the verification keys are different;
- in digital signatures typically anyone can do the verification (because it used the public key for verification), while in MAC, only the recipient can do the verification (because it uses symmetric key cryptosystem);. . .

---

*Marking scheme:* 3 points per each correct entry, capped at two for similarities and two for differences. Capped at 10 points in total.

**Question 2**

(a) Consider the following symmetric encryption scheme:

- ▷ Consider a secret key $K$ of 16 bytes, expressed in binary format (1 byte = 8 bits).
- ▷ break up the message $m$ into blocks of 16 bytes each, applying necessary padding if need be: $m = (m_1, m_2, m_3, ...)$
- ▷ bitwise XOR each block of the message with the key block to get the ciphertext: $c = (m_1 \oplus K, m_2 \oplus K, m_3 \oplus K, ...)$

Recall: the truth table for the XOR operation is as follows (on each bit):

| $x$ | $y$ | $x \oplus y$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

(i) What is the key-space of this cryptosystem? Also what is the size of the key-space? Provide a brief description of how you arrived at your answer.

**[5 marks — basic]**

**Solution:** 16 bytes means $16 \times 8 = 128$ bits. The key space (the set of all possible keys) is the set of all possible realization of a 128 bit-long binary representations (that is, the first element of the key space is $(0, ..., 0)$, i.e., 128 zeros, and the last element of the key space is $(1, ..., 1)$, i.e., 128 ones). The key space "size" (like the size of any set), is number of distinct elements in it. All the possible realizations of a 128-bit key is $2^{128}$: each bit can be independently zero or 1, so each bit has two possibilities, and there are 128 bits. So the total possible keys are $\underbrace{2 \times ... \times 2}_{128 \text{ times}} = 2^{128}$.

*Marking scheme:*

- ▷ 2 points for correct description of the key space. 2 points for correct value of the key space size. 2 points for clear computation steps. Capped at 5 points in total.

(ii) Provide the corresponding decryption algorithm.

**[5 marks — basic]**

**Solution:** Recall that XORing a binary message $m$ with another binary $k$ twice, cancels the effect out and gives back the original binary. Hence, the decryption is as follows: chop the ciphertexts into ciphertext blocks $(c_1, c_2, c_3, ...)$, then XOR the key to each block: $(c_1 \oplus K, c_2 \oplus K, c_3 \oplus K, ...)$, which will give us $(m_1, m_2, m_3, ...)$. What remains is removing any padding if it was added.

*Marking scheme:*

- ▷ 5 points for a complete answer. 3 points if only mentioning XORing with the same key.

(iii) Describe a method to "break" this cipher that is better than exhaustive key-search, i.e.,

is better than brute-force attack. (Amazingly, a similar "simple XOR cipher" was the cryptosystem used in early wireless keyboards!)

**[5 marks — medium]**

**Solution:** Since this cryptosystem does not provide any positional dependency (that is, for a given key, a plaintext gets mapped to the same ciphertext no matter where in the message it appears), it is susceptible to frequency analysis, as well as dictionary attack. The frequency analysis attack is as follows: the attacker collects a lot of ciphertext blocks, and construct a histogram of their frequency. Then the attacker knows the most frequent block in the cipher text is coming from the most frequent block in the plaintext, which is public knowledge. Now, having a plaintext-ciphertext block pair, the attacker can derive the key easily (by XORing them together: because $m_1 \oplus K \oplus m_1 = K$). Having the key, the attacker can do anything (decipher all ciphertexts, create any ciphertext, etc.).
[Note that if the attacker guesses a plaintext block for a ciphertext block, even without using frequency analysis, she can just XOR them together and get a candidate key. Then she can create a dictionary of candidate keys this way and try each of them.]

---

*Marking scheme:*

▷ 2 points for just mentioning frequency analysis + 3 points for a correct description.

(b) Consider the following diagram describing the encryption algorithm in a particular mode of operation of a block cipher like AES, and answer the subsequent questions.

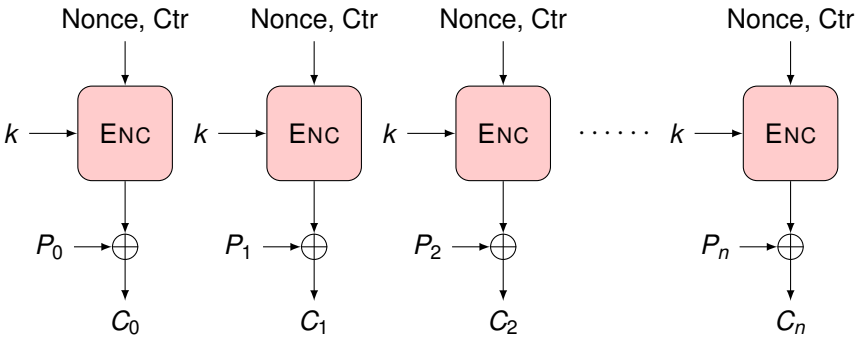



Figure 1: Q. 2-(b). Notations: $P_0,\ldots,P_n$: blocks of plaintext. "Nonce, Ctr": A counter that keeps incrementing starting from an initial value of "Nonce". $C_0,\ldots,C_n$: blocks of ciphertext. $k$: cryptographic key. ENC: an encryption block. $\oplus$: XOR (eXclusive OR) operator.

(i) Provide one advantage and one disadvantage of this mode of operation compared with the Electronic Code Book (ECB) mode.

**[5 marks — medium]**

▷ Advantage:

▷ Disadvantage:

**Solution:**

- ▷ Advantage: many possible answers, e.g.
  - unlike in ECB mode, there is positional dependency (this is because even if the same plaintext block is entered, the value of the counter has changed, which results in a completely different encryption of the counter, which is then XORed with the plaintext block, resulting in a completely different ciphertext block).
  - Unlike the ECB mode, the encryption and decryption can use the same implementation.
  - one bit error in ciphertext results in only one bit error in recovered plaintext, while in ECB, it leads to one erroneous block (on average, half of the bits in the recovered plaintext block will be wrong).
  - This mode does not require plaintext padding, unlike ECB, because what is passed to the encryption block is the counter not the plaintext.
- ▷ The main disadvantage is that it requires having a synchronised counter in the sender and receiver. It can also be said that ECB is easier to implement (in part, because of the lack of need to have a synchronised counter).

---

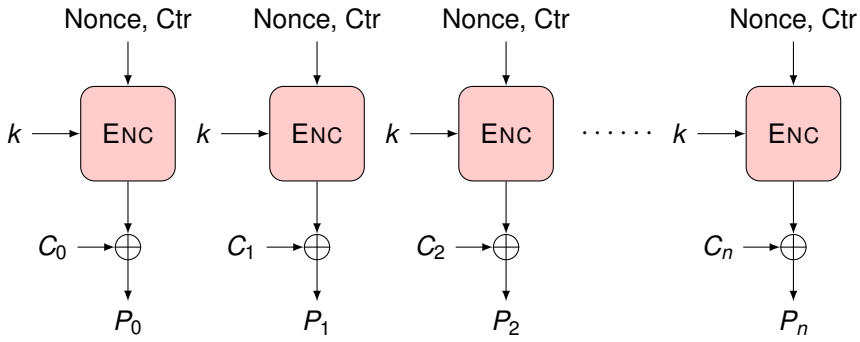
*Marking scheme:*

▷ 3 points for a correct advantage, 3 points for a correct disadvantage, capped at 5 points in total.

(ii) Draw the diagram of the corresponding decryption process.

**[5 marks — medium]**

**Solution:** Note that the hint for this was Question 2-a! The fact that if you XOR a binary with another binary twice, you will get the original binary back. A cipher block $C_0$ was produced by XORing the plaintext block $P_0$ with the encryption of the initial value of the counter. Hence, to get $P_0$ back, we need to XOR $C_0$ with exactly the same thing, which is the encryption of the same counter. So we arrive at the following diagram:




---

*Marking scheme:*

▷ 3 points if all components are correct. 2 points if the over structure is correct but some components are not marked properly. 1 point if the inputs are ciphertexts and outputs are plaintext (so a proper decryption) and some correct components (e.g. the key is there, and is the same, etc.).

(iii) If there is an error in a single bit of the cipher text (e.g. due to noise in the channel), what would be the effect on the recovered plaintext at the receiver (using this mode of operation).

**[5 marks — advanced]**

**Solution:** only one bit: this is because the value of the counter is not affected by the error in the cipher-block, so what comes out of the encryption block is correct. This correct block then gets XORed with a cipher-text that has only one bit error.

---

*Marking scheme:*

▷ 5 points for correctly identifying 1 bit. 2 points if only mentioned one block is affected.

**Question 3**

(a) Consider the following cryptographic protocol:

- Alice (the sender) and Bob (receiver) have established a symmetric key $K$. That is, $K$ is "only" known by (both) Alice and Bob.
- Both parties have also already agreed on the choice of a strong cryptographic hash function $H$.
- Alice (the sender) construct the following message $y$ from her plaintext message $m$, and sends $y$ to Bob over a public channel:

$$y = m||H(m)||H(K)$$

Recall that the operator $||$ represents concatenation (appending) of data together.

(i) For each of the following security services, determine whether our protocol provides it. Each of your answers should be supported by a brief but clear justification.

**[5 marks — basic]**

▷ Confidentiality:

▷ Data Integrity:

▷ Entity authentication:

**Solution:**

▷ confidentiality: No, the message is transmitted "in the clear" (as is, without any encryption).

▷ data integrity: No, since an attacker can do the following: replace the message $m$ with another message $m'$, then computes its hash $H(m')$, and just copies the

last part from the intercepted message $H(K)$. So the attacker can construct the message $m'||H(m')||H(K)$. The recipient will not have any way to tell whether this message is real or not.

▷ entity authentication: since there is no data integrity as described above, we do not have entity authentication either.

---

*Marking scheme:*

▷ 1 point per each correct identification, 1 point for each correct explanation, capped at 5 points in total.

(ii) Suppose the following modification is proposed: Alice first generates a random nonce $n_A$, and then constructs the message $y$ from her message $m$ as follows, and sends $y$ to Bob:

$$y = m \parallel n_A \parallel H\big(m||K||n_A\big)$$

Describe the corresponding process that takes place in the receiver; i.e., what does Bob do to $y$ upon receiving it?

**[5 marks — medium]**

**Solution:** The receiver (Bob) first identifies the three part of the received message. Say we label them as $a \parallel b \parallel c$. Bob knows $K$ as well, so he constructs $a||K||b$ and passes it to the hash function $H$, i.e., computes $H(a||K||b)$. He then compares this value with $c$ and see if it matches. So if $H(a||K||b) = c$, then he concludes the message is created (at some point in the past) by some entity who knew secret $K$ and has not been modified since its creation.

---

*Marking scheme:*

▷ if the computation of hash and comparing with the given hash is mentioned, it secures a 3 points. A complete description gets a 5.

(iii) Now, answer part (i) for the modified protocol, that is, for each of the following security services, determine whether the modified protocol provides it. Again, each of your answers should be supported by a brief but clear justification.

**[5 marks — advanced]**

▷ Confidentiality:

▷ Data Integrity:

▷ Entity authentication:

**Solution:**

▷ Confidentiality: No confidentiality still, as the message is still in the clear.

▷ Data Integrity: Yes, this time, no adversary (or accidental change) can match the keyed hash. In particular, if the message is modified by the adversary, in order

to create a correct digest, the key $K$ would be needed, which is only known to Alice and Bob.

▷ Entity authentication: No, because the protocol is still susceptible to replay attack despite having data origin authentication. This is because there is no way for the recipient (Bob) to trust that the received nonce is really fresh (he has to just rely on the word of Alice! which could be the adversary replaying an old message!) Note that if the nonce was created by Bob instead and Alice had used it in her digest, then we would have had entity authentication.

---

*Marking scheme:*

▷ 1 point per each correct identification, 1 point for each correct explanation, capped at 5 points in total.

(b) Here is a summary of the Diffie-Hellman key exchange algorithm:

▷ Both Alice and Bob (and anyone else!) know the values of two numbers: $q$ a large prime number and $\alpha$, where $\alpha$ has a special relation with $q$ (it is a "primitive root" of $q$)

▷ Alice secretly selects a random number $X_A$ less than $q$. She then computes $y_A = (\alpha^{X_A} \bmod q)$, and sends $y_A$ to Bob.

▷ Bob also secretly selects a random number $X_B$ less than $q$, computes $y_B = (\alpha^{X_B} \bmod q)$, and sends it to Alice.

▷ Alice, who has received $y_B$ and knew $X_A$, computes $(y_B^{X_A} \bmod q)$.

▷ Bob, who has received $y_A$ and knew $X_B$, computes $(y_A^{X_B} \bmod q)$.

(i) What is the equation that needs to hold in order for Alice and Bob to indeed reached the same value (which is their shared key).
*Hint: the shared key is* $(y_B^{X_A} \bmod q)$ *which is at the same time equal to* $(y_A^{X_B} \bmod q)$*, now just replace for* $y_A$ *and* $y_B$*. You do not need to "prove" the equation!*

**[5 marks — basic]**

**Solution:** We need $(y_B^{X_A} \bmod q) = (y_A^{X_B} \bmod q)$, which means we need:

$$\left((\alpha^{X_B} \bmod q)^{X_A} \bmod q\right) = \left((\alpha^{X_A} \bmod q)^{X_B} \bmod q\right)$$

This would be sufficient as an answer. But following the properties of modular arithmetic, this can be simplified to:

$$\left((\alpha^{X_B})^{X_A} \bmod q\right) = \left((\alpha^{X_A})^{X_B} \bmod q\right) \quad \text{which is equivalent to:}$$

$$(\alpha^{X_B X_A} \bmod q) = (\alpha^{X_A X_B} \bmod q)$$

Again, this later equations are not necessary for the answer of this question.

---

*Marking scheme:*

▷ 2 points if just stating ($y_B^{X_A} \mod q$) = ($y_A^{X_B} \mod q$) and 5 points if $y_A$ and $y_B$ are replaced for correctly.

(ii) Considering that transmission of the messages $y_A$ and $y_B$ happens over a public channel, and that both $q$ and $\alpha$ are publicly known, why cannot an eavesdropper find out the value of the established key? (in other words, difficulty of which specific computation prevents an eavesdropper to get to the established key?)

**[5 marks — medium]**

**Solution:** The eavesdropper sees $y_A$ and $y_B$. If she can extract $X_A$ or $X_B$ from them, then she can compute the shared key. Consider computing $X_A$ from $y_A$. So the difficult problem is finding $X_A$ from the equation $y_A = (\alpha^{X_A} \mod q)$ where $y_A$, $\alpha$ and $q$ are known. This problem, known as the "discrete logarithm" problem is believed to be computationally difficult (when $q$ is a large prime and $\alpha$ is a primitive root of it). This is why the eavesdropper, despite knowing $\alpha$ and $q$ and observing $y_A$ and $y_B$ will not be able to work out the shared key.

---

*Marking scheme:*

▷ 4 points if only mentioning discrete logarithm. 5 points if correctly describe the problem to be solved by the eavesdropper (even without mentioning the name of it to be discrete logarithm).

(iii) The Diffie-Hellman described as above is vulnerable to Man-in-the-Middle (MITM) attack. Describe exactly how to enhance the algorithm using public key certificates and digital signatures to be safe against MITM.

**[5 marks — advanced]**

**Solution:** There are many possible solutions. Here is an example:

▷ Alice computes $y_A$ (as $\alpha^{X_A} \mod q$) along with her certificate `CertA` for her verification key.

▷ Bob verifies `CertA`, then computes $y_B$ (as $\alpha^{X_B} \mod q$). Then, he signs the message $(Alice \| y_A \| y_B)$. Bob then sends the message $y_B \| CertB \| sig_B(Alice \| y_A \| y_B)$ to Alice.

▷ Now Alice verifies `CertB`, and uses the public key (verification key) of Bob to verify his signature on the message $(Alice \| y_A \| y_B)$. Next, she signs the following message $(Alice \| y_A \| y_B)$ and sends it to Bob, i.e., $sig_A(Alice \| y_A \| y_B)$

▷ Finally, Bob uses Alice's public key (verification key) to verify her signature on the message $(Alice \| y_A \| y_B)$.

---

*Marking scheme:* This is probably the most difficult question of the exam, although the answer is in the book (section 9, e.g. as depicted in Figure 9.12). Even if the solution does not provide defence against reflection attack, it will get the 5 points.

**Question 4**

(a) Describe the "collision resistance" property of a cryptographic hash function.

**[5 marks — medium]**

**Solution:** The property that it should be computationally difficult to find two distinct inputs that have the same hash value. That is, finding $x$, $y$, such that $x \neq y$ and $H(x) = H(y)$ should be computationally a difficult task.

---

*Marking scheme:* 5 points for a correct explanation. Note that if pre-image resistance or second pre-image resistance is explained instead, that will only get 1 point. Also, if instead of "difficulty" of finding, it is wrongly stated that no two distinct inputs should hash to the same value, that will only get 1 point.

(b) The following is proposed as an improvement to public key certificates as a way to enable easier entity authentication: besides the digital signature of the Certificate Authority, the owner of the public key also digitally signs the certificate, as a proof that she is in possession of the private key corresponding to the public key included in the certificate. What is the problem with this proposition? **[5 marks — advanced]**

**Solution:** This will not help because for entity authentication we need "liveness" (aka "freshness", or "timeliness") they need to sign a fresh piece of data (for instance a random nonce that is presented to them by the other entity). Signing the certificate does not provide freshness. So in particular, anyone can present this signed certificate and claim to be the owner. (Note that the certificate only creates an association between a public key and an owner's ID, but another mechanism is required to prove that whoever presents the certificate is in possession of its private key).

---

*Marking scheme:* 5 points if lack of freshness is stated. Partial points possible.

(c) Explain why the following simple attack does not work on digital signatures: "although the attacker does not have access to the signing key, she can just copy/paste the digital signature of a message $m_1$, and append it to a message of her own $m_2$ as its digital signature."

**[5 marks — basic]**

**Solution:** Unlike human signatures, digital signatures are tied to their message. That is, although the signing key is the same, the signature value will be different for different message inputs. You should know this as a basic property of digital signatures: digital signatures provide data integrity and origin authentication as well! So in short: the signing key is the same, but the digital signature of two documents that are different even in a single bit will be completely different (different in about half of the bits on average).

Therefore, a digital signature for a document will fail to verify for a different document even if the signing key (and verification keys) are the same.

---

*Marking scheme:*

▷ 5 points. 3 points if it is just mentioned that the signature will not verify. 5 points if just stating that the signature value will be different based on the message being signed.

(d) In the SSL/TLS protocol, after the "handshake protocol" is done and a master key is established, a key derivation function is used to create many symmetric keys. List four of these symmetric keys (based on what they are used for during the "record" protocol). **[5 marks — medium]**

**Solution:** If you remember the principle of "key separation", then the answer is immediate:

- A key for (symmetric) encryption of client's data sent to server (client's confidentiality);
- A key for (symmetric) encryption of server's data to client (server's confidentiality);
- A key for creating MAC for client's data sent to server (for client's data integrity and origin authentication);
- A key for (symmetric) encryption of server's data sent to client (for server's data integrity and data origin authentication).

---

*Marking scheme:*

▷ 2 points for one correct key, 3 for two keys, 4 for 3 keys, 5 points for 4 correct keys.

(e) Recall that HTTPS is just running HTTP over TLS. It is often argued that HTTPS is not required for static websites used for streaming, or holding some public information. The argument is that since there is no sensitive data being transmitted from a client to the server, and the secrecy of the information is not needed, then a HTTP suffices. What is your analysis of this argument? **[10 marks — medium]**

**Solution:** This is to test how you can translate your theoretical knowledge to a practical scenario. What does a a "secure" channel mean? It is indeed not just providing "confidentiality". TLS provides other security services: "data integrity" and "entity authentication". These other services are no less important than confidentiality. D.I. and E.A. means we are sure at the other side of the channel is the true web-server, not a man-in-the-middle, and the traffic has not been tampered with after it leaves the web-server. Indeed, some shoddy ISPs inject adware into http webpages! More seriously, adversarial man-in-the-middles can inject malicious code into the data and the recipients will be none-the-wiser.

---

*Marking scheme:*

▷ 5 points if the provision of "data integrity", "data origin authentication" or "entity authentication" is just mentioned. 5 more points if it is accompanied by a clear explanation.

**End of questions**