Security and Cryptography
Item 1 - Security Protocol Report (CW)

Module Coordinator	Dr David Williams <david.m.williams@port.ac.uk>
Issued	March 2024
Code	M22359/M33122 Security and Cryptography
Purpose	This document specifies the security protocol analysis to be completed for the security protocol component of the module to be submitted as an assignment worth 50% of the overall module mark.
Version	1.0 — Initial Release

Schedule & Deliverables

Item	Value	Format	Deadline
Item 1 - Report (CW)	50%	PDF	2024-05-08 16:00 **BST**

Notes

- Submit your answers to the following exercises in one single PDF file to the submission dropbox on moodle.
- Use the accompanying examples to help you complete the security protocol analysis; together.
- Remember: we're here to support you as you build, develop and refine your knowledge and skills and to help you rise to the challenges set.
- This is individual work. You *will* learn extensively from each other, but any work you submit must be your own.

SECRYPT - Security and Cryptography - Exercises

Use the accompanying examples to help you complete the following exercises; together, the examples and exercises help develop a technical understanding of a broad range of ciphers.

Make a copy of the Item 1 Template file and populate it with your student number and answers to each of the following exercises; do not change the formatting.

Print to PDF and submit this single PDF file to the submission dropbox on moodle.

Exercise 1 - Diffie Hellman Key Exchange

For this exercise you will calculate shared secrets established between protocol participants based on values generated using the functions `generate_DH_values` and `generate_ECDH_values` in `protocol_generator.py`

This function takes a three digit number `num` as input and returns a set of parameters.

To generate your parameters enter the last three digits of your student number.

Here's an example of using the python function to generate the plaintext:

```
>>> generate_DH_values('123')
```

a. Calculate the secret that Alice and Bob upon via Diffie Hellman Key Exchange using the values generated using `generate_DH_values`. Also provide the values sent over the public channel by Alice and Bob in establishing the shared secret.

[10 marks]
4 marks for correct value of shared secret
and 3 marks for each correct calculation of a sent value

b. Calculate the secret that Alice and Bob agree upon via Elliptic Curve Diffie Hellman Key Exchange using the values generated using `generate_ECDH_values`. Also provide the coordinates sent over the public channel by Alice and Bob in establishing the shared secret.

[10 marks]
4 marks for correct coordinates of shared secret
and 3 marks for each correct calculation of a sent value

[20 marks total for exercise]

Exercise 2 - Authentication Protocol Specification

For this exercise you will analyse a security protocol generated using the function `generate_protocol_two(num)` in `protocol_generator.py`

This function takes a three digit number `num` as input and returns a protocol specification.

To generate your protocol for analysis, enter the last three digits of your student number.

Here's an example of using the python function to generate the plaintext:

```
>>> generate_protocol_two('123')
```

We shall call this protocol `PROTOCOL_TWO`, i.e., `PROTOCOL_TWO` is the protocol generated by inputting the last three digits of your student number into `generate_protocol_two(num)`

a. In your own words, describe the message exchange of `PROTOCOL_TWO`.
[10 marks]
2 marks for each accurate point made that aids the description (up to 10 marks max)

b. Construct an SPDL specification of `PROTOCOL_TWO` for verification in Scyther.
[10 marks]
10 marks for a correct construction without errors
else 2 marks for each accurate element (up to 8 marks max)

c. To what extent does `PROTOCOL_TWO` mutually authenticate the protocol participant? Justify your answer through analysis.
[10 marks]
2 marks for each accurate point made (up to 10 marks max)

[30 marks total for exercise]

Exercise 3 - Authentication Protocol Analysis

Consider the following authentication protocol given in common syntax:

```
A, B, S:     principal
Na, Nb, Ns:  nonce
pk, sk:      principal -> key (keypair)

1. A->S: {A,B,Na}pk(S)
2. S->B: {S,A,Ns}pk(B)
3. B->S: {A,B,Ns,Nb}pk(S)
4. S->A: {B,Na,Nb}pk(A)
5. A->B: {S,Nb}pk(B)
```

We shall call this protocol `PROTOCOL_THREE`

The SPDL specification of `PROTOCOL_THREE` has been written for you, it is to be found in the file `PROTOCOL_THREE.spdl`.

Including the responder's identity in message 3 of `PROTOCOL_THREE` is necessary for the protocol to ensure Aliveness from the perspective of the initiator `I`.

a. In your own words, describe a counter-example that illustrates that the removal of the responder's identity `R` from message 3 causes the protocol to fail to exhibit Aliveness from the perspective of the initiator `I`.

[10 marks]

2 marks for each accurate point that aids the description (up to 10 marks max)

b. In your own words, describe why it is necessary to identify the initiator `I` in message 1. Justify its importance through security protocol analysis.

[5 marks]

1 mark for each accurate point that aids the description/justification (up to 5 marks max)

c. In your own words, describe why it is necessary to include the number used once `Ns` (generated by the authentication server) in messages 2 and 3. Justify its importance through security protocol analysis.

[5 marks]

1 mark for each accurate point that aids the description/justification (up to 5 marks max)

d. In your own words, describe why it is necessary to identify the server `S` in message 5. Justify its importance through security protocol analysis.

[5 marks]

1 mark for each accurate point that aids the description/justification (up to 5 marks max)

[25 marks total for exercise]

Exercise 4 - Authentication Properties

Consider the following authentication protocol given in common syntax:

```
A, B:        Principal
Na:          Nonce
sk:          Principal -> Key

1. A->B: {A,B,Na}sk(A)
2. B->A: {Na}sk(B), {A}sk(B)
```

We shall call this protocol `PROTOCOL_FOUR`.

The SPDL specification of `PROTOCOL_FOUR` has been written for you, it is to be found in the file `PROTOCOL_FOUR.spdl`.

a. With reference to `PROTOCOL_FOUR`, explain (in your own words) how Weak Agreement and Non-Injective Agreement differ from each other.

[10 marks]
2 marks for each accurate point made (up to 10 marks max)

[10 marks total for exercise]

Exercise 5 - Authentication Using Tickets

For this exercise you will analyse a security protocol generated using the function `generate_protocol_five(num)` in `protocol_generator.py`

This function takes a three digit integer `num` as input and returns a protocol specification.

To generate your protocol for analysis, enter the last three digits of your student number.

Here's an example of using the python function to generate the plaintext:

```
>>> generate_protocol_five('123')
```

We shall call this protocol `PROTOCOL_FIVE`; it is the protocol generated by inputting the last three digits of your student number into `generate_protocol_five(num)`.

a. In your own words, describe the message exchange of `PROTOCOL_FIVE`.

[5 marks]
1 mark for each accurate point made that aids the description (up to 5 marks max)

b. Construct an SPDL specification of `PROTOCOL_FIVE` for verification in Scyther.

[5 marks]
5 marks for a correct construction without errors
else 1 marks for each accurate element (up to 4 marks max)

c. To what extent does `PROTOCOL_FIVE` mutually authenticate the protocol participant? Justify your answer through analysis.

[5 marks]
1 marks for each accurate point made (up to 5 marks max)

[15 marks total for exercise]

[100 marks total for assignment]