Duration : 90 minutes
Class : INT3307E
*Open books and notes, no notebooks, no mobile phones*
*No discussion or exchange of documents between students during the exam*

# Final Exam Solutions
# Network Security

*(3 problems, 3 pages, point values given in parentheses, 10 maximum)*

## 1. Secret key distribution and user authentication (4 points)

*a. (2 points)*

A believes that she shares $K'_{AB}$ with B since her nonce came back in message 2 encrypted with a key known only to B (and A). *(0,75 point)*

B believes that he shares $K'_{AB}$ with A since $N_A$ was encrypted with $K'_{AB}$, which could only be retrieved from message 2 by someone who knows $K'_{AB}$ (and this is known only by A and B). *(0.75 point)*

A believes that $K'_{AB}$ is fresh since it is included in message 2 together with $N_A$ (and hence message 2 must have been constructed after message 1 was sent). *(0.25 point)*

B believes (indeed, knows) that $K'_{AB}$ is fresh since he chose it himself. *(0.25 point)*

*b. (1.5 point)*

We consider the following interleaved runs of the protocol:

1. A → C(B) : $ID_A \| N_A$
1`. C(B) →A : $ID_B \| N_A$
2`. A →C(B) : E($K_{AB}$, [$N_A \| K'_{AB}$])
2. C(B) →A : E($K_{AB}$, [$N_A \| K'_{AB}$])
3. A →C(B) : E($K'_{AB}$, $N_A$) *(1 point)*

C cannot encrypt A's nonce, so he needs to get help with message 2. He therefore starts a new run with A, letting A do the encryption and reflecting the reply back. A will accept the unprimed protocol run and believe that B is present. *(0.5 point)*

*c. (0.5 point)*

To prevent the attack, we need to be more explicit in the messages, e.g. by changing message 2 to include the sender and receiver (in this order), i.e. to be E($K_{AB}$, [$ID_A \| ID_B \| N_A \| K'_{AB}$]).

## 2. X.509 certificates (2 points)

The chain of certificates that enables A to verify B's public key in B's X.509 certificate, issued by U, is as follows:

T<<Q>>Q<<P>>P<<W>>W<<X>>X<<R>>R<<U>>U<<B>> *(1 point)*

The verification proceeds as follows:

- A uses T's valid public key, which can be obtained securely when A requests T to issue a certificate for A, to verify Q's certificate issued by T (T<<Q>>).

- If the certificate T<<Q>> is valid then A uses Q's valid public key contained in it to verify P's certificate issued by Q (Q<<P>>).
- If the certificate Q<<P>> is valid then A uses P's valid public key contained in it to verify W's certificate issued by P (P<<W>>).
- The process continues similarly for subsequent certificates.
- If the certificate R<<U>> is valid then A uses U's valid public key contained in it to verify B's certificate issued by U (U<<B>>).
- If the certificate U<<B>> is valid then B's public key contained within it is considered valid. *(1 point)*

## 3. Transport-level security (4 points)

*a. (3 points)*

Draw all the messages, except the *certificate* and *certificate_request* messages in phase 2, and the *certificate* and *certificate_verify* messages in phase 3. *(1.5 point)*

The *server_key_exchange* message in phase 2 contains another RSA *(0.25 point)* public key *(0.25 point)*, generated by the server instantly *(0.25 point)* and used for encryption *(0.25 point)*.

The *client_key_exchange* message in phase 3 contains the *pre_master_secret* generated by the client on the fly *(0.25 point)* and encrypted using the instant RSA public key received earlier from the server *(0.25 point)*.

*b. (1 point)*

- The attacker obtains a copy of Boogle's certificate
  None of the above. *(0.25 point)*
  The certificate is public. Anyone can obtain a copy simply by connecting to Boogle's webserver. So learning the certificate doesn't help the attacker. *(0.25 point)*
- The attacker obtains the private key of a certificate authority trusted by users of Boogle
  The attacker can impersonate the Boogle web server to a user. The attacker can't decrypt past data. First, Boogle's private key is used in the protocol, not the CA's. Second, Diffie–Hellman provides "forward-secrecy", and so the attacker could not decrypt it regardless. The CA's private key can be used for creating bogus certificates, which can be used to fool the client into thinking it is talking to Boogle. Replays aren't possible, due to the nonces in the TLS handshake. *(0.5 point)*