Homework Assignment 7
Sample Solution

*1. Consider the **Kerberos Authentication System**, given in Lecture 13, Slides 7 and 8.*
(a) Why is the nonce (timestamp) t used in message 6?
(b)Why is the nonce (timestamp) t encrypted?

(a): To make sure that each session between A and B uses a different key. Additionally if a third computer C snoops message 6 and sends it again to the TSG impersonating A, the TSG will detect the duplicated message (same t) and reject the request. A yet different but related reason is that if a computer C discovers Alice's $K_{A,AS}$, it can impersonates Alice and convince the TSG that it is Alice to be able to talk to B.
(b): This is to provide an extra layer of security. If a computer C snoops the message 6 it can extract the timestamp and send another request to the TGS with a newer timestamp trying to impersonate Alice. If the timestamp is encrypted, it is a lot more difficult for C to do so.

*2. Consider the **Diffie Hellman Key Exchange Algorithm**, given in Lecture 13, Slide as corrected, with the values n = 19, g = 4, x = 3 and y = 6.*
(a) Show the specific numbers that Alice and Bob send to each other in Steps (1) and (2) of the algorithm.
(b) Show that Alice and Bob construct the same shared secret key in Steps (3) and (4) of the algorithm.

(a)
In step **1** Alice sends (n, g, g^x mod n) to Bob:
(4^3 mod 19) = (19, 4, 64 mod 19) = **(19, 4, 7)**
In step **2** Bob sends (g^y mod n) to Alice:
(4^6 mod 19) = (4096 mod 19) = **(11)**
(b)
In step **3** Alice computes $K_{A,B}$ = (g^y mod n)^x mod n:
(4^6 mod 19)^3 mod 19 = (4096 mod 19)^3 mod 19 = 11^3 mod 19 = 1331 mod 19 = **1**
In step **4** Bob computes $K_{A,B}$ = (g^x mod n)^y mod n:
(4^3 mod 19)^6 mod 19 = (64 mod 19)^6 mod 19 = 7^6 mod 19 = 117649 mod 19 = **1**
Which is consistent with $K_{A,B}$ = g^{xy} mod n:
4^{3*6} mod 19 = 68719476736 mod 19 = **1**

*3. Consider an **access control matrix** M and the corresponding **access control list** A and **capability list** C.*
(a) Explain the relationship between M and A, and the relationship between M and C.
(b) Explain how a server uses an access control list for an object O to allow/disallow access to the object O by a subject (client).
(c) Explain how a server uses a capability list for a subject (client) S to allow/disallow access to an object by the subject S.

(a): An access control list ***A*** is list that each object maintains of the access rights of a subjects. ***A*** for a object ***O*** It can be thought as the non-zero entries of column ***M***[*,***O***]. Conversely, a capability list ***C*** is given to each subject that contains list of the capabilities that subject has for each object. It can be thought as the non-zero entries of row ***M***[***S***,*].

(b): When a client sends a request to a server to access an object, the server will first check if it knows the client, if so, it checks if the client is allowed to access that particular object with its **access control list**, if so, it allows the access, else it disallows it.

(c): When using **capability lists**, clients sends their request for access an object, as well as its **capability**. The server checks if the capability is valid and if the operation is listed in the capability list. If so, it allows the access, else it disallows it.

*4. (a) In a **distributed object-based system**, describe the purpose of the **stub** on the client-side and the **skeleton** on the server-side.*
*(b) Explain what is meant by **marshalling** and **unmarshalling** of messages.*

(a): An **stub** is essentially a proxy that is used to provide a transparent interface of a remote object as if it was local, as such, the stub needs to provide the same interface as the remote object. A stub takes requests from clients, prepares them (by marshalling) and sends them over the network to the remote server. The **skeleton** is quite similar but in the server side. It receives method calls, translated them (by unmarshalling) to operations to be run on the object. The skeleton then returns the result back to the caller in a similar fashion.

(b): Marshalling refers to converting requests or data in general into a format that is suitable for transmission over a network, typically as a byte stream. Unmarshalling is the reverse operation: transform the transmitted data into the original data or request.