# CSS 322 – ASSIGNMENT 2

First name: _____ Last name: _____

ID: _____ Total Marks: _____

out of 80

Due Date: Wednesday 21 February 2007, 9am (you can hand in before the start of the lecture)

*I certify that, unless otherwise acknowledged, all work carried out in this assignment is my own.*

Sign Name: _____ Date: _____

**Instructions**

- This is an individual assignment. You are not allowed to work in groups on any part of the assignment. Plagiarism will be penalised. (You must sign the statement at the top of this cover sheet).
- The assignment must be handed in by the due date. Late assignments will receive 0 marks.
- The assignment should be neatly handwritten and/or computer generated (for example, Word).
- You must give your calculations, working out, discussion and design decisions. That is, if you only give a correct answer, but no calculations, then you will not receive full marks (in fact, you will receive very few, maybe 0 marks).
- The assignment should be on A4 sheets, with a single staple in the top-left corner. Please do not use plastic sleeves, folders etc.
- You must attach this Cover Sheet (including name, ID and signature) to the front of your assignment.
- Some of these questions, and the text explaining them are taken from the course textbook by Stallings.

**Question 1** [10 marks + 10 bonus]

The toy *tetragraph hash* (*tth*) is a hash function similar in nature to SHA, but operates on letters instead of binary data. The function can be described as follows:

Given a message consisting of a sequence of letters, *tth* produces a hash value consisting of four letters. First, *tth* divides the message into blocks of 16 letters, ignoring spaces, punctuation and capitalisation. If the message length is not divisible by 16, it is padded out with nulls. A four-number running total is maintained that start out with the value (0,0,0,0); this is input to the compression function for processing the first block.

The compression function consists of two rounds:

1. Round 1: Get the next block of text and arrange it as a row-wise 4 x 4 block of text and convert it to numbers (A = 0, B = 1, etc.). For example, for the block ABCDEFGHIJKLMNOP, we have:

| | | | |
|---|---|---|---|
| A | B | C | D |
| E | F | G | H |
| I | J | K | L |
| M | N | O | P |

| | | | |
|---|---|---|---|
| 0 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 |
| 12 | 13 | 14 | 15 |

   Then, add each column mod 26 and add the result to the running total, mod 26. In this example, the running total is (24,2,6,10).

2. Round 2: Using the matrix from round 1, rotate the first row left by 1, second row left by 2, third row left by 3, and reverse the order of the fourth row. In our example:

| | | | |
|---|---|---|---|
| B | C | D | A |
| G | H | E | F |
| L | I | J | K |
| P | O | N | M |

| | | | |
|---|---|---|---|
| 1 | 2 | 3 | 0 |
| 6 | 7 | 4 | 5 |
| 11 | 8 | 9 | 10 |
| 15 | 14 | 13 | 12 |

Now, add each column mod 26 and add the result to the running total. The new running total is (5,7,9,11).

This running total is now the input into the first round of the compression function for the next block of text. After the final block is processed, convert the final running total to letters. For example, if the message is ABCDEFGHIJKLMNOP, then the hash is FHJL.

a) Calculate the hash function for the 48-letter message: "I leave twenty million dollars to my friendly cousin Bill." Hint: Use a Excel spreadsheet or similar to perform the calculation. [10 marks]

b) To demonstrate a weakness of *tth* (that it is easy to find collisions), find a 48-letter block that produces the same hash as that just derived. Hint: start with all A's and then look at the influence of changing the first 4 or 5 letters. [**Bonus 10 marks**]

**Question 2** [25 marks]

A direct digital signature scheme involves only the source and destination users. The examples of signatures covered in lectures were using direct digital signatures. A problem with these schemes is if the source/sender later wants to deny signing a message by claiming their private key was lost or stolen.

The problems associated with direct digital signatures can be address by using an arbiter. Every signed message from sender X to receiver Y actually goes from sender X to arbiter A and then from arbiter A to receiver Y. That is, the arbiter A also signs the message and checks the origin and content. This can solve the problem of sender X disowning a message (e.g. saying their private key was stolen).

Three arbitrated digital signature techniques available are (when X wants to send a signed message to Y):

Option 1 – Conventional encryption is used; the arbiter sees the message

Message 1 from X to A: $M \,||\, E(K_{xa},[ID_X \,||\, H(M)])$

Message 2 from A to Y: $E(K_{ay}, [ID_X \,||\, M \,||\, E(K_{xa},[ID_X \,||\, H(M)]) \,||\, T])$

Option 2 – Conventional encryption is used; the arbiter does not see the message

Message 1 from X to A: $ID_X \,||\, E(K_{xy}, M) \,||\, E(K_{xa}, [ID_X \,||\, H(E(K_{xy}, M))])$

Message 2 from A to Y: $E(K_{ay}, [ID_X \,||\, E(K_{xy}, M)]) \,||\, E(K_{xa}, [ID_X \,||\, H(E(K_{xy}, M)) \,||\, T])$

Option 3 – Public key encryption; arbiter does not see the message

Message 1 from X to A: $ID_X \,||\, E(PR_X, [ID_X \,||\, E(PU_y, E(PR_x, M))])$

Message 2 from A to Y: $E(PR_a, [ID_X \,||\, E(PU_Y, E(PR_X, M)) \,||\, T])$

Notation:

| | | |
|---|---|---|
| $X$ = sender | $M$ = message | $T$ = timestamp |
| $Y$ = recipient | $A$ = arbiter | $K_{ab}$ = Shared secret key between a and b |

$PU_a$ = Public key of A $PR_a$ = Private key of A

In the following questions, it is very important that you clearly describe the procedures (e.g. when you refer to a key, you must say what type and possible whose key). It is recommended that you use the same notation as used above in your answers.

a) Explain how a man-in-the-middle attack on the message between X and A in Option 1 can be detected. State your assumptions about the key and the hash function H(). [5 marks]

b) Explain how the man-in-the-middle attack in part (a) can be successful (that is, undetected by A) if the same assumption hold about the key as in part (a), but the hash function used is that used in Question 1 (*tth* – make note of *tth*'s weakness in Q1(b), even if you do not solve Q1(b)). [5 marks]

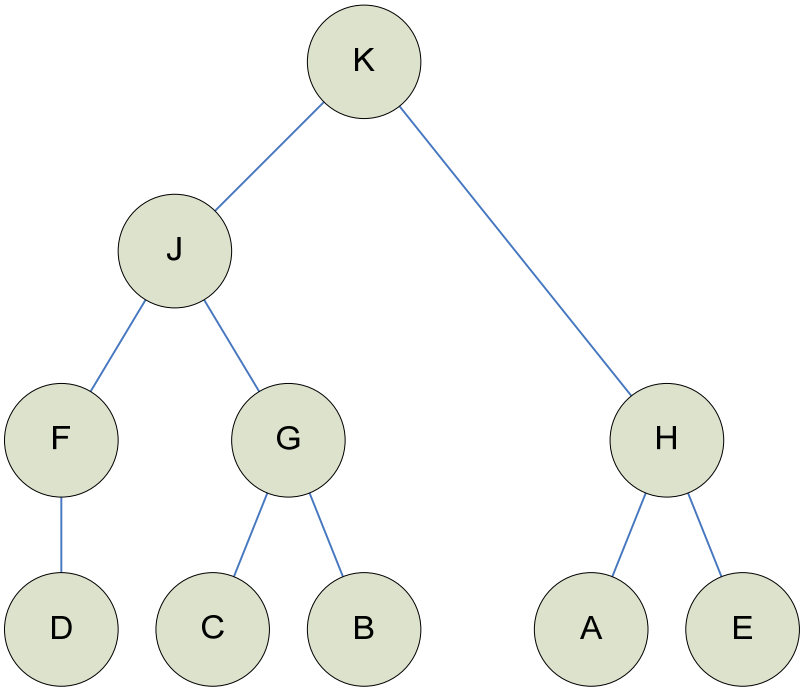
c) If a malicious node intercepts the message from A to Y in Option 3, can the malicious node perform a replay attack. If yes, explain how the attack is performed. If no, explain how it is detected. [5 marks]

d) Modify the technique in Option 3 to avoid triple encryption of the entire message. [5 marks]

e) A limitation of direct digital signatures is that a node can "cheat" by saying someone has stolen the key. With the arbitrated schemes above, explain a way that two of the nodes could collude (work together) to cheat the system. Also explain why this is a very unlikely scenario. [5 marks]

**Question 3** [30 marks]

The figure below shows a X.509 hierarchy of users and Certificate Authorities (CAs), where the users are leaf nodes and each parent node is a CA of its children nodes.



a) Which node issues A's certificate? [2 marks]

b) Explain the steps that must be taken for user A to verify the identity and public key of user B. In your steps you must include all necessary exchanges with CAs (e.g. users' obtaining certificates). State any assumptions you make. [10 marks]

c) If you followed your steps described in part (a), for each of the 10 nodes, list the public keys it now has. [5 marks]

d) List and describe an advantage and disadvantage of using such a certificate hierarchy. [5 marks]

e) An important aspect of public key distribution is determining who to trust. Design a scheme for establishing trust amongst users where there is no hierarchy (e.g. no CA's) and different levels of trust are assigned to users. For example, one user may assign a trust level to another user of 0.8, meaning they "trust them 80%". You can assume there are some users (but not all) that a user will trust 100%.

In your design, consider ways to obtain and calculate the trust level, and when to know that you can trust (and hence use) information from a user or not. As much as possible, these methods should be automatic, that is, without any human intervention (although you may need some initial assignments of trust by human users).

Your design does not have to be long, but should be detailed enough so that I could use it to implement a simple application that could be used in a real computer network.

In your design make sure you state any assumptions and you identify and comment on any security or performance limitations of your scheme. [8 marks]

**Question 4** [15 marks]

You must submit your answers in two forms:

1. Written (including this assignment sheet)
2. Electronic (via email to steve@siit.tu.ac.th)

The steps below outline what you must do to complete the electronic submission. You must use CrypTool to create the certificates and perform encryptions.

1. Generate your own certificate using RSA/1024-bit. This will automatically be signed by CrypTool, which acts as a Certificate Authority. You can use any PIN for PSE (I do not have to know it). Include a printout of your certificate data in your report (you can copy and paste it from CrypTool into MS Word).
2. Export your certificate and save it using the default file name (e.g. mine is similar to "[Gordon][Steven][RSA-1024][12345678].p12"). Note that when exporting you must enter a PKCS#12 PIN – this is different than the PSE PIN. Use the value 1234 for the PKCS#12 PIN, as I need to know this value.
3. Import my certificate into CrypTool (I also used the PKCS#12 PIN of 1234). An electronic copy (in the Personal Information Exchange format, which can be directly imported into CrypTool) is available at: http://it.siit.tu.ac.th/~sgordon/cs322/protected/GordonStevenRSA-1024-certificate.p12

4. Create a text file with your answers to question 1 only (not calculations) in the following format:

   ID=12345689

   Answer1a=value1

   Answer1b=value2

   where you replace 12345689 with your ID, and replace value1 with the value you calculated for Answer 1(a) and so on. If you did not calculate an answer, use 0 as the answer.

5. Sign your answer text file using your certificate and a SHA1 160-bit hash function.
6. Encrypt the signed answer text file using RSA and my certificate. Save the resulting file as a binary (.hex) file using your ID as the file name, e.g. 123456789.hex (if your ID was 123456789).
7. Send me via email your certificate file (.p12) and the encrypted/signed answer text file (.hex). Your subject of the email must be: CSS322 Assignment 2 ID (where your replace ID with your actual ID), and the two files should be attached. There is no need to archive the two files using ZIP or RAR – simply attach them as is.

When marking your assignments, I will perform the following steps to check:

1. Import your certificate into CrypTool
2. Decrypt the answers file using RSA and my certificate
3. Verify your signature.
4. Check the answers.

I will not be checking your certificate and answers before the assignment is due (in other words, please do not ask me if your certificate worked). However, I will try to reply to your email confirming its receipt as soon as I receive the email.