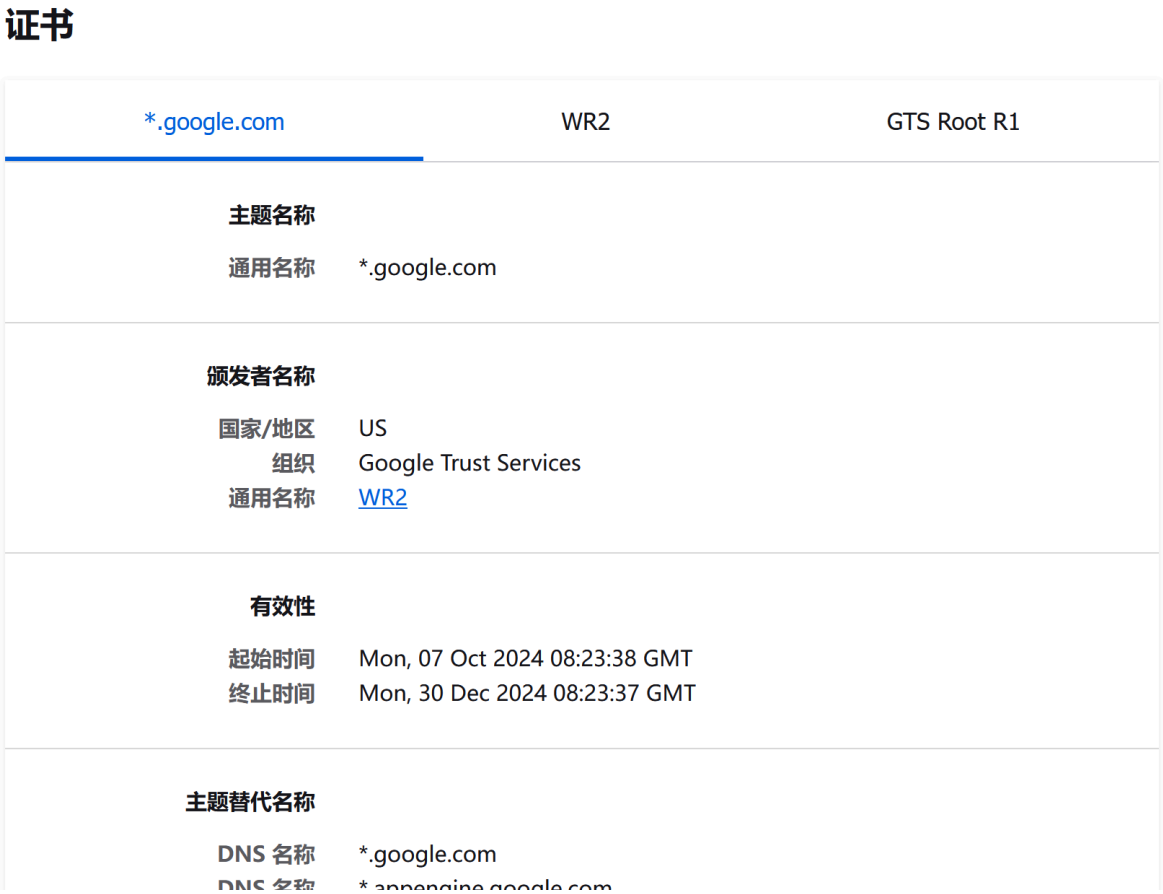
COMP7906 Introduction to Cyber Security

Assignment 3

ZHANG Hongyi
h1kari@connect.hku.hk

Q1

PKC Screenshot

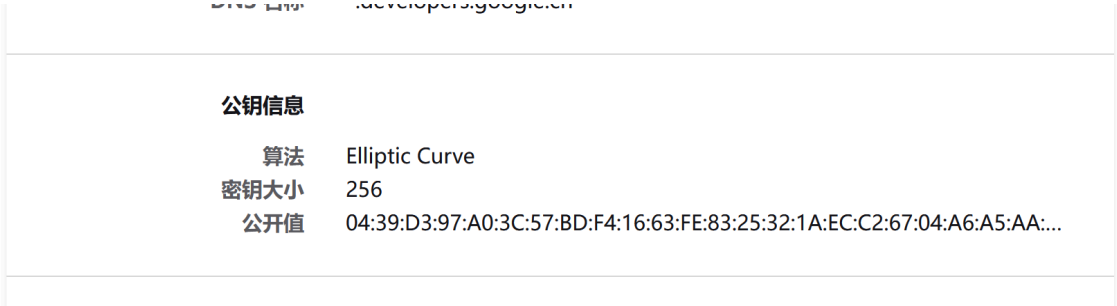


The screenshot of the PKC of www.google.com in Firefox is shown above.

Issuer (CA)

The issuer of the PKC is **Google Trust Services LLC** according to the detailed information.

Signing algorithm and the key length



In Firefox, the signing algorithm is shown as **Elliptic Curve**. And the key length of it is **256 bits**.

However, for its superior certificate, which are shown as *WR2* and *GTS Root R1* above, the algorithm is **RSA**, and the key lengths are **2048** and **4096**.

Q2

Alice's private key

$$\begin{aligned} n_A &= 77 \\ &= 7 \times 11 \end{aligned}$$

So we can choose $p = 7, q = 11$.

Since $e_A = 23$, and $e_A \times d_A \equiv 1 \bmod \varphi(n)$, $d_A = \frac{k\varphi(n)+1}{e_A} = \frac{60k+1}{23}, k \in \mathbb{N}$, we can calculate the value of d_A:

$$d_A = 47$$

So, Alice's private key $(d_A, n_A) = (47, 77)$.

The value of the plaintext m

According to RSA, $m = c^{d_B} \bmod 91 = 82$.

So the plaintext m is **82**.

Q3

How can B show evidence to prove A had sent message?

B can use his own private key B_{pri} to decrypt the transmitted message, which includes A's signature and PKC.

Then B can obtain the public key A_{pub} corresponding to A's private key A_{prv} by the PKC to verify the aforementioned signature to ensure it belongs to A, which can prove the message was sent by A.

How can B be sure that another person D cannot get the plaintext?

The message was sent between A and B using asymmetric encryption method. According to the schema, the message was encrypted using the public key B_{pub}.

Hence, it is computationally infeasible for D to derive B_{priv} from B_{pub} or to break the encryption to get the plaintext, as long as the private key B_{pri} is well kept as secret.

Which properties of the cryptographic hash function prevent an attacker from froging the signature?

- **One-way Property**: Given any value y, it is infeasible to find a value x such that $h(x) = y$. This property makes sure that attackers cannot come up with a fake message and keep the same signature as the original one.
- **Second-preimage Resistance**: Given $h(x)$ and x, it is infeasible to find $y \neq x$ such that $h(x) = h(y)$. This property ensures that attackers cannot find another message producing the same hash result as the original message.

Apart from these properties mentioned in the slides, the cryptographic hash function can compute the hash value quickly and the result of the same plaintext is always the same.

The length of the output of the hash function is fixed as well.

Q4

Which of the above approach is wrong?

The third approach are wrong.

B has the public key A_{pub} corresponding to A's secret key A_{prv}. So what B can obtain from the ciphertext is two copies of $H(M)$. Due to the properties of cryptographic hash function, B cannot know the plaintext M even in the situation of knowing the hash function. In this approach, B cannot get the message M.

However, the first approach is also problematic because it cannot guarantee the integrity of the message.

Which one is most efficient? And which one is least efficient?

The fourth one is the most efficient approach, and the first one is least efficient.

The fourth approach use asymmetric encryption to encrypt the symmetric encryption key K, which is much shorter than the megabytes-long message, and then use block cipher encryption to encrypt and decrypt the message. Compared to the other two approach, the fourth approach avoids using the asymmetric encryption method on long messages.

The first approach applys the asymmetric encryption method on long messages twice, due to the asymmetric encryption method will not reduce the length of the text. So it is the least efficient approach compared with the second approach which only applys the public key operation once every time.

Q5

For 1024-bit RSA, the number of operations is roughly 1024^3. And for 2048-bit RSA, the number of operations is roughly 2048^3.

The ratio of operations required for 2048-bit integers to that for 1024-bit integers:

$$\text{Ratio} = \frac{2048^3}{1024^3} = 8$$

Thus, the performance deteriorates by a factor of 8 when moving from 1024-bit to 2048-bit RSA.

Q6

1. *Customer (C) selects items (I) and confirm the order (O), and send the request purchase(C, I, O) to the shop.*
2. Shop (S) calculates the total cost of the order (TotalCost), and send a message (S, TotalCost, C, O) to the customer.
3. Customer sends payment request to the bank with the message $\text{Enc}_{K_{\text{cust}}}(C, \text{TotalCost}, S)$. Here, C represents the bank account of the customer. Since we can assume communications between them are secure, it's not necessary to use generated random numbers to prevent attacks.
4. The bank receive the payment request and process the payment. After the transaction is done, the bank sends back the receipt: $\text{Enc}_{K_{\text{cust}}}\Big(\text{Enc}_{K_{\text{shop}}}(C, \text{TotalCost}, S)\Big)$.
5. The customer receives the receipt and decrypts it with the shared key between the bank and the customer K_{cust}, and sends the message to the shop: $\Big(\text{Enc}_{K_{\text{shop}}}(C, \text{TotalCost}, S), C, O\Big)$.

6. The shop receives the message and decrypts it with the shared key between the bank and the shop K_{shop}, and get the proof of the customer C has paid TotalCost through the bank. After checking the corresponding C and O, the shop sends the confirmation to the customer.

For the four requirements:

1. **The bank does not know what items are purchased by the customer**: The bank is only made aware of the TotalCost and the order number, but it does not know the individual items being purchased. The item list is never sent to the bank.
2. **The shop does not know how the customer makes the payment**: The shop receives a Receipt from the customer, but it does not have any details about the customer's bank account or payment method. The shop only knows that the payment was processed by the bank.
3. **The customer has proof of payment**: The bank sends a Receipt to the customer, which acts as cryptographic proof that the payment was made. The customer can present this receipt to the shop.
4. **No direct communication between the bank and the shop during the payment process**: The bank and the shop do not need to communicate during the transaction. The customer forwards the Receipt to the shop. The shop can verify the receipt with the shared key K_{shop} after receiving it from the customer.

Q7

In this project, we can use both asymmetric encryption method (e.g. RSA) and symmetric encryption method (e.g. AES) to ensure the security of the discussion forum.

For the users, everyone should generate different pairs of asymmetric keys for signing the documents they post, which declares the ownership of the document or the comments.

Let's assume that there is a remote server, which stores the documents and their information (e.g. their classes, owners and comments) as well as the secrets including public keys and AES keys of documents. We also assume that the communications between servers and users are secured by shared symmetric encryption.

If a user want to post a document

The user sends a request to upload a document to the server, which includes the process identifier and the user identifier. If the document is of class "G", the user sends the document directly combined with the signature calculated using the asymmetric encryption method. However, if the document is class "C" or class "E", the server responds with a generated AES key that is used to encrypt the content of the document. The user encrypts the document with the received AES key and sends it to the server along with the signature calculated using the asymmetric encryption method. The server checks the availability of these messages and stores the encrypted document and its signature. The AES keys should be transported using the user's public key in encrypted form.

Specifically, for class "E" documents, the user should also send the list of members who can view the document.

If a user want to comment on a document/comment

The user sends a request of commenting on a specific document/comment that includes the operation identifier, the object's identifier and the user identifier. After receiving the request, the server checks user's class first to determine whether the user can comment on the specific document/comment. If the comment is class "G", the user sends the comment directly to the server combined with the signature. For class "C" and "E", like posting documents, the server comes up

with a generated AES key which will be used to encrypt the content of the comment, and sends it using user's public key. The user encrypts the document with the received AES key, and sends it to the server combined with the signature calculated using asymmetric encryption method. After server confirms the availability of the message, it will store the comment.

Specifically, for class "E" comments, the user should send the list of members who can view the document as well.

If a user want to access other's document/comment

The user sends a request of accessing a specific document/comment that includes the operation identifier, the object's identifier and the user identifier. After receiving the request, the server checks user's class first to determine whether the user can access the specific document/comment.

If the object is of class "G", the server will transport the content directly to the user.

Otherwise, if the user can access the document/comment, the server transports the AES key encrypted using user's public key and encrypted content to the user. Then the user can view the content. However, due to the lack of the correct signature which signed using the owner's private key, the user cannot modify the content.

If a user want to modify a document

The user sends a request of accessing a specific document/comment that includes the operation identifier, the object's identifier and the user identifier. After receiving the requst, the server should verify if the user is the document's owner. So the server sends a newly-generated AES key to the user encrypted using owner's public key. In order to modify the document, the user should respond with a message including modified document with the signature (encrypted using new AES key). So that the server can make sure the identity of the user and apply the modification.