Information Integrity and Authentication

IMPORTANT NOTES:
Study lecture materials at least 1 hour and prepare Q1-6 prior to the tutorial session. Prepared questions will be discussed in the tutorial session.

1. Briefly describe three main use of digital signature.

   Digital signature is used for proving authenticity of the origin of documents as well as for non-repudiation as it can prove the authenticity of the signer.

   Another use of digital signature is to ensure the integrity of the messages against unauthorised modification.

2. Describe the stages of generating and verifying the RSA digital signature for long documents.

   Signature Generation

   (a) Generate the message digest $h = H(M)$ of the document M using a (secure) one-way hash function H.

   (b) Exponentiate the message digest with private key (d, n) of the signer by computing $s = h^d \bmod n$

   (c) Attach the signature s to the message

   Signature Verification

   (a) Generate the message digest $h = H(M)$ for received document M using the same (secure) one-way hash function

   (b) Exponentiate the attached signature s using public key (e, n) of the signer to compute $h' = s^e \bmod n$

   (c) Compare the h' with h, if a match happens then verification is successful.

3. Discuss features of a good one-way hash function.

   (a) Can be applied to any size of data: files and messages with arbitrary sizes can be signed

   (b) Produces a short fixed-length output: communication overhead of hash for transmission or storage is low and independent of the hashed message length

   (c) Fast and efficient algorithm: hashing algorithm should be fast and efficient (low overhead in terms of processing time)

   (d) One-wayness: infeasible to get back message from the hash value

   (e) Hard to find collisions (collision-resistance): it should be infeasible to find two inputs that will be hashed into the same value

4. Discuss digital signature requirements.

   - Must be a bit pattern based on the message being signed
   - Must use some information unique to the sender (prevent forgery and denial)
   - Relatively easy to produce
   - Relatively easy to recognize and verify
   - Computationally infeasible to forge by either constructing a new document with the same digital signature or forge a signature for a document
   - Must be practical to retain a copy of the digital signature

5. What is Message authentication?

   Message authentication is a mechanism or service used to verify the integrity of a message. Message authentication assures that data received are exactly as sent by (i.e., contain no modification, insertion, deletion, or replay) and that the purported identity of the sender is valid.

6. What are three requirements for MAC?

   (a) Knowing a message and MAC, it is computationally infeasible to find another message with same MAC

   (b) MACs should be uniformly distributed across the messages

   (c) MACs should depend equally on all bits of the message

7. For n=77, e=13 and d=37 what is the value of a RSA digital signature of message M=15? ($15^5 \bmod 77 = 1$). Assume the basic (textbook) RSA signature where no hash function is used.

   Signature = $M^d \bmod n = 15^{37} \bmod 77 = (15^5 \bmod 77)^7 \times (15 \times 15 \bmod 77) = 225 \bmod 77 = 71$

8. For n=77 e=17 the value of a RSA digital signature for message M=12 is 45. Show the verification process ($9^{15} \bmod 77 = 1$, $5^{15} \bmod 77 = 34$). Assume the basic (textbook) RSA signature where no hash function is used.

   Message = $(Sig)^e \bmod n = 45^{17} \bmod 77 = [(5 \times 9)^{15} \bmod 77] \times [(5 \times 9)^2 \bmod 77] =$
   $(5^{15} \bmod 77) \times (9^{15} \bmod 77) \times (81 \bmod 77) \times (25 \bmod 77) = (34 \times 1 \times 4 \times 25 \bmod 77) =$
   $34 \times 100 \bmod 77 = 34 \times 23 \bmod 77 = 782 \bmod 77 = 12 = M$ (Signature is verified)

9. List two disputes that can arise in the context of using Message Authentication Codes (MACs).

   Suppose that John sends an authenticated message to Mary. The following disputes could arise:

   (a) Mary may forge a different message and claim that it came from John. Mary would simply have to create a message and append an authentication code using the key that John and Mary share.

   (b) John can deny sending the message. Because it is possible for Mary to forge a message, there is no way to prove that John did in fact send the message.

10. Suppose $H(m)$ is a collision resistant hash function that maps a message of arbitrary bit length into an n-bit hash value. Is it true that, for all messages x, x' with $x \neq x'$, we have $H(x) \neq H(x')$? Explain your answer.

    The statement is false. Such a function cannot be one-to-one because the number of inputs to the function is of arbitrary, but the number of unique outputs is 2^n. Thus, there *exist* multiple inputs that map into the same output. However, if the hash function is collision-resistant, it is computationally infeasible to actually *compute* two distinct inputs that map into the same output.

Optional Questions for Further Exploration

1. What is the difference between a message authentication code and a one-way hash?

   A hash function, by itself, does not provide message authentication. A secret key must be used in some fashion with the hash function to produce authentication. A MAC, by definition, uses a secret key to calculate a code used for authentication.

2. With regards to a n-bit output hash function H with $n \geq 32$:

   (a) How long does it take on average for a brute force attack to find a message M such that $H(M)$ has 32 **zero** leading (leftmost) bits? Note that in this case, the $n - 32$ trailing (rightmost) bits of $H(M)$ can be arbitrary. Assume that the output of the hash is evenly distributed and each input is independent.

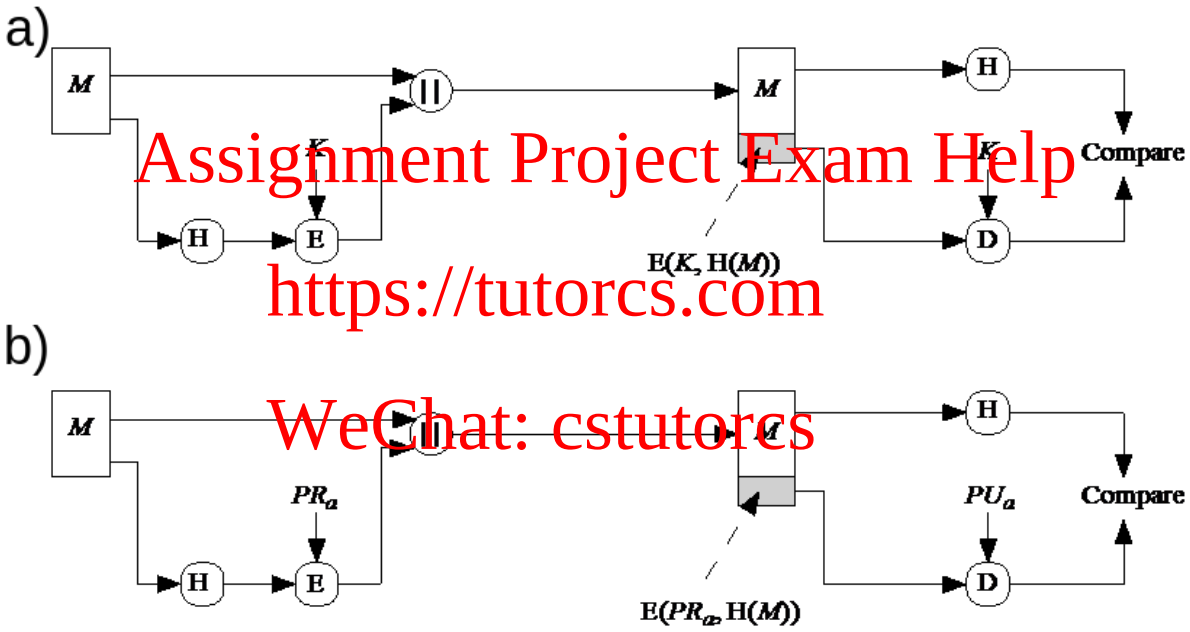
(b) How long does it take on average for a brute force attack to find two messages M_1 and M_2 such that $H(M_1)$ and $H(M_2)$ collide on the 32 leading bits? Note that in this case the string of 32 leading bits of $H(M_1)$ must equal the string of 32 leading bits of $H(M_2)$ though the value of that 32 bit string can be arbitrary, while the trailing $n - 32$ bits of $H(M_1)$ and $H(M_2)$ can be arbitrary and unequal bit strings.

If we only consider the 32 bits at the leftmost (leading) side of $H(M)$ as the hash value, the size of the hash value space is 2^{32}.

(a) It will take about 2^{32} trials on average to find a hash value equal to a specific value (in this case the specific value is 32 zero bits), the same as the size of the hash value space.

(b) Due to the birthday paradox, it will only take about $\sqrt{2^{32}} = 2^{16} = 65536$ trials on average to find a collision on the 32 leftmost bits of two messages, i.e. the square root of the size of the hash value space.

3. The following figure illustrates two methods in which a hash code can be used to provide message authentication. Explain both methods.



(a) Only the hash code is encrypted, using symmetric encryption. No other entity without having access to the key K can modify the message ($M \rightarrow M'$) and produce a valid $E(K, H(M'))$ or produce a fraudulent message (M'') and attach a valid $E(K, H(M''))$ hence the destination B can verify the authenticity and integrity of the message by decrypting the $E(K, H(M))$ and comparing it with locally generated $H(M)$.

(b) Only the hash code is encrypted, using public-key encryption and using the sender's private key. As only the sender has access to PR_a no other entity can modify the message ($M \rightarrow M'$) and produce a valid $E(PR_a, H(M'))$ or produce a fraudulent message (M'') and attach a valid $E(PR_a, H(M''))$ and since the public has access to PU_a the destination B can verify the authenticity and integrity of the message by decrypting the $E(PR_a, H(M))$ (using PU_a) and comparing it with locally generated $H(M)$.