103011227 周延儒 CS18

1. [20 points] **15.1** Buffer-overflow attacks can be avoided by adopting a better programming methodology or by using special hardware support. Discuss these solutions.

   One form of hardware support that ensure a buffer overflow attack won't take place is to **prevent the execution of code that is located in the stack segment of a process's address space.**
   To be more specific,buffer-overflow attacks are usually performed by **overflowing the buffer on a stack frame**, **overwriting the return address** of the function, thereby jumping to another portion of the stack frame that contains **malicious executable code,** which had been placed there due to the buffer overflow.
   By **preventing the execution of code from the stack segment**, this problem is eliminated. Approaches that use a better programming methodology are typically built around the use of **bounds-checking to guard against buffer overflows**. Buffer overflows do not occur in languages like Java since every array access is checked to be within bounds through a software check.
   Such approaches require no hardware support but result in runtime costs associated with performing bounds-checking.

2. [20 points] **15.2** A password may become known to other users in a variety of ways. Is there a simple method for detecting that such an event has occurred? Explain your answer.

   In my opinion, we could detect such an event by whenever a user logs in, the system **prints the last time that user was logged on the system.**

3. [20 points] **15.4** The list of all passwords is kept within the operating system. Thus, if a user manages to read this list, password protection is no longer provided. Suggest a scheme that will avoid this problem. (Hint: Use different internal and external representations.)

   We could just **encrypt the passwords internally** so that they can only be accessed in coded form. The only person **with access or knowledge of decoding** should be the **system operator.**

4. [20 points] **15.11** What commonly used computer programs are prone to man-in-the- middle attacks? Discuss solutions for preventing this form of attack.

   In general, any protocol that requires a sender and a receiver to **agree on a session key before they start communicating** is prone to the man-in-the-middle attack.For example , if one were to implement on a secure shell protocol by having the two communicating machines to **identify a common session key,** and if the **protocol messages for exchanging the session key is not protected** by the appropriate authentication mechanism, it is possible for an **attacker to manufacture a separate session key and get access to the data** being communicated between the two parties.

   In particular, if the server is supposed to manufacture the session key, the attacker could obtain the session key from the server, communicate its locally manufactured session key to the client, and thereby **convince the client to use the fake session key.**
   When the **attacker receives the data from the client, it can decrypt the data, reencrypt it with the original key from the server**, and transmit the encrypted data to the server without alerting either the client or the server about the attacker 's presence.
   The attacks could be **avoided by using digital signatures to authenticate messages from the server.** If the server could communicate the session key and its identity in a message that is guarded by a **digital signature granted by a certifying authority,** then the attacker would not be able to forge a session key, and therefore the man-in-the-middle attack could be avoided.

5. [20 points] **15.12** Compare symmetric and asymmetric encryption schemes, and discuss the circumstances under which a distributed system would use one or the other.

   A symmetric encryption scheme **allows the same key to be used for encrypting and decrypting messages.** On the contrary, an asymmetric scheme requires the use of **two different keys for performing the encryption and the corresponding decryption**.

   Moreover, Asymmetric key cryptographic schemes **are based on mathematical foundations and formula that guarantee the intractability of reverse-engineering the encryption scheme**, but they are typically much

more costly than symmetric schemes, which do not provide any such theoretical guarantees.

Plus, asymmetric schemes are usually **superior to symmetric schemes** because they could be used for other purposes such as **authentication, confidentiality, and key distribution.**