CSE 430 Homework 9

Ryan Dougherty

Question 14.12

The access-control matrix can be used to determine whether a process can switch from, say, domain A to domain B and enjoy the access privileges of domain B. Is this approach equivalent to including the access privileges of domain B in those of domain A? Yes, this approach is equivalent (given that the switch privileges in domain B are copied over to domain A).

Question 14.15

Discuss the strengths and weaknesses of implementing an access matrix using access lists that are associated with objects.

- Strength: the control from storing the access privileges along with each object (i.e. the object can revoke or expand the access privileges locally).
- Weakness: the overhead of checking whether the requesting domain appears on the access list (the check is expensive and done every time the object is accessed).

Question 14.16

Discuss the strengths and weaknesses of implementing an access matrix using capabilities that are associated with domains.

- Strengths: these capabilities have flexibility and allow faster access to objects in the access matrix (what is only needed is to check the capability's authenticity). Another strength is that these capabilities can be passed between domains.
- Weaknesses: less control from revoking any capabilities. Also, restricting their flow is difficult.

Question 14.19

What is the need-to-know principle? Why is it important for a protection system to adhere to this principle? The need-to-know principle is when a process may access at any time those resources that it has been authorized to access and are required currently to complete its task. It is important because it limits the amount of damage a faulty process can cause in a system.

Question 15.1

Buffer-overflow attacks can be avoided by adopting a better programming methodology or by using special hardware support. Discuss these solutions. One way is to prevent code execution inside the stack segment of a processs address space. When this attack happens, the buffer overflows on the stack frame, and overwrites the function's return address. At this point, the control jumps to a different part of the stack frame that has maliciously written code because of the buffer overflow. If we prevent execution of code from the stack segment, we eliminate the problem.

Ways using a better programming methodology are made for the concept of checking bounds, guarding against any buffer overflow; they do not require hardware support but have runtime cost in checking bounds.

Question 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. We can detect this event in that if a user does a login, the associated system prints to the user the last time he/she logged in to the system. That way, if the user sees an access time that he/she was not a part of, he/she can know if a different user logged in with the same password.

Question 15.9

Make a list of six security concerns for a banks computer system. For each item on your list, state whether this concern relates to physical, human, or operating-system security. Six security concerns (and their associated securities) are:

- Programmers/Data personnel are trusted: human.
- The network cannot be tampered: physical, human, operating system.
- Protect/Guard backups/drives: physical, human.
- Log/Prevent unauthorized data transfers: human, operating system.
- Limit/Eliminate modem access: physical, human.
- System is in a well-protected and well-guarded location: physical, human.

Question 15.12

Compare symmetric and asymmetric encryption schemes, and discuss the circumstances under which a distributed system would use one or the other.

- Symmetric encryption allows the same key to be used for encryption and decryption. This encryption is less expensive than asymmetric, but does not have theoretical guarantees. A distributed system may use symmetric encryption if performance is an issue.
- Asymmetric encryption requires the use of two different keys for encryption and decryption. This encryption is theoretically secure, but is much more expensive than symmetric. It is also better than symmetric because it can be used for authentication, confidentiality, and key distribution, useful in a distributed system.

Question 15.13

Why doesn't $D_{kd,N}(E_{ke,N}(m))$ provide authentication of the sender? To what uses can such an encryption be put? $D_{kd,N}(E_{ke,N}(m))$ is equivalent to: the message is encrypted using the public key and decrypted using the private key. It cannot guarantee authentication because anyone can obtain public keys. Therefore, any attacker can make up the message. However, the only person who can perform decryption is the one who owns the private key. This guarantees that the message is secret from the sender to the person owning the private key.