

Information Assurance: Homework 4 Answers and Comments

Due October 10, 2007

1. In class we discussed a number of different types of separation to protect different entities from one another.
 - a. Describe one advantage and one disadvantage to physical separation.

With true physical separation you are guaranteed that there is no information leakage between programs. This comes at the cost of extra hardware and general awkwardness in managing entities across multiple physical devices.

- b. Describe one advantage and one disadvantage to temporal separation.

With temporal separation, you do not have to pay for multiple copies of physical hardware. In addition, you do not have to program using the hardware at the same time, so you get some of the benefits of physical separation. However, with temporal separation, the same hardware will eventually be used for multiple entities. If data is not adequately removed between program slices, there may be inadvertent information leakage.

2. For each of the memory protection frameworks below answer the following questions:
 - a. Does it protect the operating system from errors in the user process? How?
 - b. Does it protect one user process from another? How?
 - c. What is the limitation of this approach, if any?

The memory protection frameworks are:

- Fence register

- a) *Yes, the fence register stores the address above the top of the operating system. The user program will use relative addresses starting at 0 to refer to its own address space. At run time, the contents of the fence register will be added in, so there is no way the user program can directly address OS memory space unless the user program can change the value of the fence register.*
- b) *No, with a single fence register, there is no protection between multiple user processes.*
- c) *In general fence registers are good for single user systems. The fence register does not protect against the user program addressing too high, beyond its own address space.*

- Base bounds registers

- a) *Yes, the lower base register acts much as the fence register did. The user program refers to addresses starting at 0 and the base is added in at run time.*
- b) *Yes, this can protect between user programs. Each user program space is given a pair (or multiple pairs of base bounds registers). The system adds in the base to the address and verifies that the address is less than the value in the bounds. So if two user programs are given two separate address ranges, they will have separate base bounds registers with separate values. Assuming these registers are set correctly, there is no way the first process can access addresses in the second process's range unless the user program can change the values of the base bound registers.*
- c) *The bounds check can add significant overhead. If the user programs space requirements change during execution, there may not be space to augment the existing range.*

- **Segmentation**

- a) *Yes, segmentation is an evolution of the base bounds registers. Rather than a single range, the user's address space can be broken into a series of segments. Each program address reference is with respect to a segment and offset pair. The segment table maps the segment ranges to physical memory much as the base/bounds registers did. The OS memory is not mapped to user level segments.*
- b) *Yes, segmentation should also protect user programs from each other. However, if programs did need to share data, they could chose to share segments. Both programs could refer to addresses in the same segment. This implies that there is an access control mechanism to prevent unwanted sharing. If program 1 did not want to share his bob segment, the architecture would need to provide a means to prevent program 2 from accessing the bob segment even if it had a legitimate offset into the segment.*
- c) *Segments have many of the same drawbacks as base-bounds registers. Doing the upper bounds checking is a performance hit. If the segment needs to grow beyond its currently mapped location, it will need to be remapped elsewhere causing another performance hit and potential fragmentation problems.*

- **Paging**

- a) *Yes, all program addresses are expressed in terms of page number and offset. A page table is used to map from a logical page to a physical location. If the OS memory is not paged, there is no way from a user program to directly access OS memory.*

- b) *Each program has a separate page table, so there is no way for one program to access another program's pages. So the page table does protect user programs from each other.*
- c) *With page tables, you lose the logical groups of areas of user memory that you had with segments. You also lose the ability to naturally share memory address space which can be beneficial for cooperating processes.*

- Paged Segments

- a) *Yes, as with segments, the user programs refer to logical addresses in terms of segments and offsets. The OS memory is not segmented, so there is no way for the user program to access this data directly.*
- b) *Yes, as with segments, the user programs refer to separate logical addresses. In theory programs could chose to share segments, so there should be another access control mechanism to prevent undesired sharing.*
- c) *The paged segments provide two levels of table which can add to performance overhead.*

3. Consider the following system. There are the following users

- Alice and Bob are Engineers
- Carol is in Finance
- Dave and Bob are system administrators
- Ellen is the CEO

There are the following files:

- System designs which should be read and written by the Engineers and read by the CEO.
- Financial statements which should read and written by the Financial department and read by the CEO.
- System config files which should be read and written by the system administrators
- In an emergency, the CEO should be able to read and write all files and delegate access to others.

a. Write an Access Control Matrix for this system

You could have written this in terms of individuals or groups. There are multiple ways to address the emergency requirement. At the most basic, you could give the CEO read/write right up front and add a delegate right, but this would violate least privilege. You could have two ACMs, and switch to the other ACM in the event of emergency. You could add an emergency right, that enables the holder to give himself any privilege and delegate. Many people made two logins for Ellen: one normal and one emergency with more rights.

An number of people attempted use what appeared to be an ownership right 'O' which would have worked for delegation. Some other letters appeared too. You only got full credit if you explained when the new right did.

In the ACM below, I modeled the emergency requirement with a Take Ownership right (T), that enables the subject to give himself all rights, and a delegate right (D), that enables the subject to pass a right to a file to another entity. Implementing delegation safely can be quite tricky and is often seen as a benefit of the capability model. On second look at my solution, my non-emergency Ellen is not running at least privilege either because she has the 'D' right in the non-emergency case which she should not.

	System Design	Finance Stmtns	System Config	Alice	Bob	Carol	Dave	Ellen
Alice	RW							
Bob	RW		RW					
Carol		RW						
Dave			RW					
Ellen	RTD	RTD	TD					

- b. Write access control lists for the representative types of objects that encodes these constraints.

System Design ACLs:

- Alice: RW
- Bob : RW
- Ellen: RTD

Finance Stmtns ACLs:

- Carol: RW
- Ellen: RTD

System Config ACLs:

- Dave: RW
- Bob: RW
- Ellen: TD

To activate Ellen's delegate right, you could simply give Ellen the right to change the file's ACL. Though this loses the sense that the other subject is doing an operation on Ellen's behalf.

A number of people added a directory here which wasn't necessary. I don't know if that was an artifact of the example shown in the slides or confusion with the discussion of directories as an access control tracking mechanism in the text.

c. Write capabilities for the users that encode these constraints.

Alice's capabilities:

- *System Design: RW*

Bob's capabilities:

- *System Design: RW*
- *System Config: RW*

Carol's capabilities:

- *Finance Stmts: RW*

Dave's capabilities:

- *System config: RW*

Ellen's capabilities:

- *System Design: RTD*
- *Finance Stmts: RTD*
- *System Config: TD*

In the capability system, Ellen could delegate by giving another subject a copy of her capability. Say she activated her Taken Over right and now has the following capability:

- *System Design: RWTD*

Alice and Bob are away, so she wants Carol to fix something in the System Design area. The D right could give her the right to create a copy of the capability to pass onto Carol, e.g.

- *System Design: RW*

With a capability, Ellen could keep hold of the copy and revoke it if she feels that Carol is misusing the delegated rights. This would be more awkward to do with ACLs.

4. A system allows the user to choose a password with a length of one to ten characters inclusive. Assume that 15,000 passwords can be tested per second. The system administrators want to expire passwords once they have a probability of 0.10 of being guessed. Determine the expected time to meet this probability under each of the following conditions.
 - a. Password characters must be digits (“0” through “9”).

$1/10 = \text{Number of Passwords explored} / \text{Total number of passwords}$

*Number of passwords explored = 15,000*T, where T is the number of seconds running*

Total number of passwords = $N = \sum_{i=1}^{10} 10^i = 11,111,111,110$

$$0.1 = 15,000 * T / N$$

$$N * 0.1 / 15,000 = T = 74074 \text{ seconds} = 20.5 \text{ hours}$$

- b. Password characters may be capital letters (“A” through “Z”) and numerics (“0” through “9”).

$N = \sum_{i=1}^{36} 36^i = 3760620109779060$ passwords

$$T = 25070800731.8604 \text{ seconds} = 794 \text{ to } 795 \text{ years}$$

- c. 12 bits of salt are added for both a and b.

Multiple the results of a and b by 4096. With the salt, the attacker should try all 4096 hash algorithms to see if any of the variants are represented in the password file. The salts are stored in the password file, so the attacker can note that only k salts are used in this particular file and only test those hash variants, but at worst it will increase his work by a factor of 4096.

Some of you noted that salt does not deter the attacker when attacking a specific user offline or attacking the system online.

5. Try running the John the Ripper password cracking program <http://www.openwall.com/john/>. You should be able to install it local to your environment for an unprivileged account. Obtain a password file from <http://www.cs.uiuc.edu/class/fa07/cs461/class07-passwd> This file contains nine accounts with passwords from a linux system. At least one password should be cracked very easily. If you have access to a private system, try running the program for a while longer to see if you get more passwords cracked. Submit the account names and passwords that you crack. As long as you get the quickly cracked passwords, you will get full credit.

Alice: Alice.

Bob: bobbob

Carol: carol77

dave: 10-10-72

ellen: qghrlz

grace: maclusbar

fred: class-test

helga: cs461

ivan: xylophone

Everyone got Alice and Carol. Many people got Helga's password. Six of you got passwords for Alice, Carol, Helga, and Ivan.

6. An organization implements a biometric authentication system. All employees register their fingerprints, and the organization stores the resulting templates on a central server. Eve hacks the server and gains access to the template. What harm can occur from this breach? How does it compare to hacked passwords?

The template is not an exact copy of the fingerprint. Rather it is a measure of some key features of the registered fingerprint. Eve could use this information to generate a fingerprint facsimile. Eve knows exactly what the reader is looking for. In theory she could manufacture a fingerprint measurement sequence that matches what the reader is looking for.

Of course, we leave fingerprints all over the place. So if Eve wants to create a copy of someones fingerprint, she could get that information without hacking into the server. See "Impact of Artificial "Gummy" Fingers on Fingerprint Systems" <http://www.lfca.net/Fingerprint-System-Security-Issues.pdf> for more information on fooling fingerprint readers. This is not necessarily the case with other biometric measurements like iris scans.

In general stealing the biometric template is worse than having a password hacked. You can change your password. It is more difficult if not impossible to change a physical characteristic.

Some of you noted that once you have a person's fingerprint data, you can correlate every other account owned by this person, even accounts at other organizations as long as they use the same biometric. Another one of you noted that you need to be physically present to use the hacked biometric, whereas a hacked password can be used over a network connection-- and on the other side of the coin, it can gain you physical access where a password may not (depending on the policy, of course).

There is ongoing work in the biometric area to verify biometric characteristics while keeping the specifics of the registered biometric data hidden.