Graduate Program in Information Science and Telecommunications and Networking
School of Information Sciences
University of Pittsburgh

# TEL2821/IS2150: INTRODUCTION TO SECURITY
# Lab: Operating Systems and Access Control

Version 3.2, Last Edited 9/11/2009

Students Name: _____

_____

_____

Date of Experiment: _____

***Read the following guidelines before working in the lab***

## General Guidelines

There are four (4) computers for use for this lab. They are located in groups of two machines in two separate tables in the lab and are labeled for use in this course.

The machines you'll use have two operating systems installed on them: Windows Vista Ultimate Edition and Unix/Linux variant called Fedora Core 7. By default the system will boot to the Windows Vista operating system, when you need to work on Unix/Linux you should reboot (restart) the system and press the Enter key when the system reports that it will try to load the operating system (a few seconds after the system is booting up). Then, select Fedora as the operating system to use for you Unix/Linux work.

A written report is required for this assignment. You should turn in this printout as part of your written report. Space has been provided to answer some of the questions of the exercises in this lab. However, you should attach extra sheets of paper with your answers for some of the questions. Please label your extra sheets with your name and indicate precisely which questions you are answering.

For the Windows Vista assignment you will need to log on and log off from different user accounts quite frequently. The log off option is not very visible in the operation system. So, to log off from an account select *Start* (the Windows Icon in the lower left corner) and then select the right arrow next to the lock icon presented in the window. The *Log Off* option should become visible and you can now select it.

## I: Objective

The objective of the exercises presented here are to familiarize the student with the access control features available in the Microsoft Windows and UNIX-based systems and to compare them.

## II: Equipment/Software

- A PC with Microsoft Windows 2000/XP or Windows Vista installed on it.
- A PC with Linux installed on it.

## III. Access control in Microsoft Windows OS

Access control refers to the ability of a user to access a particular object and possibly modify it. In terms of operating systems, access control refers to the ability of a user to read, write or execute a certain file or folder. In this laboratory, we shall study the access control framework for Microsoft Windows and UNIX-based platforms, by taking Microsoft Windows Vista and Linux as respective examples.

The Microsoft Windows 2000/XP/2003 and Vista series of OSs introduced access control for files, directories and devices. Before we introduce access control for these operating systems, let us take a look at how objects are arranged in these systems.

The Active Directory service was introduced in the Windows NT family of operating systems as a means of arranging all users, devices and objects at a centralized location and allowing these networked entities to find each other through this service. Entities are known as objects and they are arranged into a hierarchical structure by the administrators, known as the logical structure. A collection of objects that share the same security policies is known as a domain (a container object) and multiple domains can be arranged hierarchically into a tree. A forest is a complete instance of the Active Directory that consists of a set of domains that trust each other through a two-way transitive trust. This arrangement of objects into logical structures enables easy management of the objects and allows for more flexible access control. The place Active Directory has in the network is shown in Figure 1.

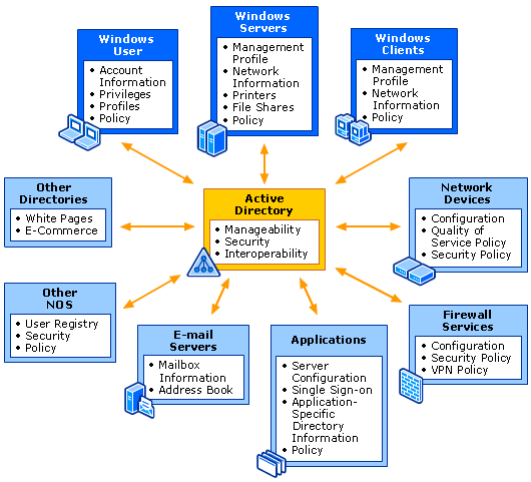



**Figure 1: Active Directory**

Every object (all users, groups, domains, processes) has a Security Identifier (*SID*) associated with them that is a unique identification associated with it. The SID is very similar to the UID in UNIX. Objects that have some operations associated with them and

whose access must be controlled are called *securable objects*. Securable objects have *security descriptors* associated with them that consist of *Discretionary Access Control lists* (DACLs), describing which users or groups have what access rights over them and *System Access Control Lists* (SACLs) that describe how auditing is done and the SID of the owner of the object. Every time an object is created, a security descriptor can be assigned to it, but if it is not assigned, it will inherit it from its parent object.

A *security context* is associated with every process (or user) which describes which groups it belongs to, what privileges it has and what accounts are associated with it. The security context is maintained in an *access token*. Access Control Lists (ACLs) for an object contain the SID of the intended *trustee* and an access mask for the various access rights. When access is requested, the access token of the accessing object requesting, is checked with the security descriptor of the accessed object, to see if the access should be permitted or not. An example is given in figure 2.

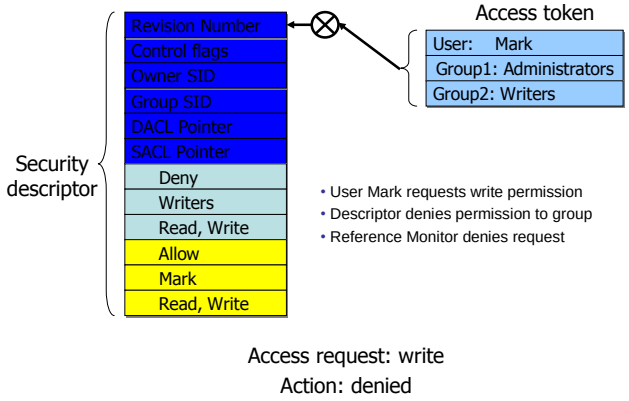



**Figure 2: Example of Access Request**

For added security, to protect sensitive data, the Encrypted File System (EFS) was introduced in the Windows NT family. The EFS allows users to encrypt objects created by them so that no other object can access them. The encryption is done using an EFS certificate that the user gets and multiple users can be added to allow access, with the help of their EFS certificates.

## Lab Procedures

***Read the lab procedures before starting work in the lab. If you have any questions, clarify them with the GSA or the instructor. This will make your work in the lab more efficient.***

**Exercise 1.1: Adding users to the system**

1. Log in as *StudentAdmin* (password: *security*)
2. Go to Start -> Control Panel -> User Accounts and Family Safety -> Add or remove user accounts
3. Click on "Create a new account" to add a new user. Enter the username as *group<gr#>A* where *<gr#>* is the identification number that has been assigned to your group by the instructor. Select "Standard User" as the account type.
   **Note:** You should have been assigned a group number by the instructor. Otherwise select a number between 1 and 99 as your group number and mention it in your document to the GSA or instructor.
   With your group number, the accounts you create for this lab will have the form (assuming a group number 8) *group8A* and *group8B.*
4. Click "Create Account…"

**Exercise 1.2: Studying the effects of using the *Read-Only* and *Hidden* attributes of a file.**

1. Log in as *StudentAdmin*
2. Open the *Notepad* application to create a text document (Start->All Programs -> Accessories -> Notepad)
3. Type some text in the document and save it in the *C:\doctemp* folder with the name *file1_group<gr#>.txt*
4. Open the *C:\doctemp* folder and open the *Properties* window for the file you just created by doing a clicking with the right button of the mouse over the file and selecting *Properties* from the menu.
5. In the Attributes section, mark the **Read-Only** attribute and then click *OK*
6. Open the file again, add some text and try to save the changes.
   a. What happens? Explain why it happens.

      _____
      _____
      _____
      _____
      _____
      _____

7. Log off from the *StudentAdmin* account and log on as *group<gr#>A*.
8. Open the *C:\doctemp* folder and open the *Properties* window for the *file1<gr#>.txt* file.

9. Mark the **Hidden** property of the file. Click OK. Close the window and then open the *C:\doctemp* folder again.
    a. Do you see the file now? _____
    b. If you cannot see the file, try to find ways to show the hidden file. Which one worked?

       _____
       _____

**Exercise 1.3: Explicitly assigning permissions to different users for a given file.**

1. Log in as *StudentAdmin* .
2. Use *Notepad* to create a .txt document and save it in the *C:/doctemp* folder with *file2_group<gr#>.txt* as the filename.
3. Open the *C:\doctemp* folder and open the *Properties* window for the file you just created.
4. Select the *Security* tab. Click on the *Edit…* button
5. Click on the *Add...* button and add the user *group<gr#>A* to the file by typing *group<gr#>A* in the text box field. Click on the *Check Names* button. If no errors are reported, click on OK. Click the other OK buttons until you return to the *Properties* window and *Security* tab is active.
6. Select the user you just added by clicking on its username. What are the default permissions given to this user? While still viewing the *Security* tab options for this file see the permissions assigned to the group called *Users*. Are the default permissions for this group different than those of the user you added to have access to the file?

   _____
   _____
   _____
   _____

7. View the advanced set of permissions for access to this file for user *group<gr#>A*. To do this, open the *Security* tab in the *Properties* window for this file, select the user, press the *Advanced* button, select the user again in the new window and press *Edit* , again press the *Edit* button
8. What do you see? How is it different than the permissions in the parent *security* tab? Is there any relation?

   _____
   _____
   _____
   _____
   _____
   _____
   _____
   _____

**Exercise 1.4: Understanding MS Windows access control mechanism**

1. Log in as *StudentAdmin*
2. Create an additional user in the system. The user name will be *group<gr#>B* (Refer to Exercise 1.1).
3. Open the *C:\doctemp* folder and click the *Organize* button, select the *New Folder* option and create a new folder with the name *folder<gr#>*
4. Select the folder you just created and open its *Properties* window by right clicking on it and selecting the *Properties* option.
5. Select the *Security* tab and click on the *Edit..*
6. Click the *Add...* button and add the *group<gr#>B* user. Be sure to press the *Check Names* button before pressing OK.
7. Select the *group<gr#>B* user in the *Permissions* window. What permissions are currently assigned to the user?

   _____
   _____
   _____

8. Mark the *List folder contents* permission under the *Deny* column. Click OK and approve all changes.
9. Logoff from the *StudentAdmin* account and log in as *group<gr#>A*
10. Open the *Notepad* application to create a text document (Start->All Programs -> Accessories -> Notepad)
11. Type some text in the document and save it in the *C:\doctemp\folder<gr#>* folder with the name *file3_group<gr#>.txt*
12. Open the *C:\doctemp\folder<gr#>* folder and verify that you can open the document you just created.
13. Logoff from the *group<gr#>A* account and log in as the *group<gr#>B* user
14. Open the *C:\doctemp\folder<gr#>*. Explain what happens when you access the folder and why?

    _____
    _____
    _____
    _____
    _____
    _____

## Auditing in Windows

The family of Windows OS products from Windows 2000 onwards provides audit functions that include the following basic functions:

- audit collection
- audit review
- audit log overflow protection
- audit log restricted access protection

By default, at installation, only application logs and error logs are collected and stored by the Audit function. The server administrator must enable security auditing on the machine.

**Exercise 1.5: Audit Log Review**

The administrator is able to view all the security log records using an event viewer administrator tool and differentiate security logs from application and system logs. The objective of this exercise is to view the security log records with administrative privileges.

1. Log in as *StudentAdmin*
2. Open the event viewer. (Start->Control Panel->System and Maintenance ->Administrative tools -> View event logs)
3. Once in the Event Viewer, select *Windows Logs* category (left side of the window)
4. Study the entries for security logs, application logs and system logs.
5. Log off from the *StudentAdmin* account and log on as *group<gr#>A*
6. Repeat steps 2 to 4

**Questions**

1. Can you view security logs as normal users? Should normal users be able to view security logs?

   _____
   _____
   _____
   _____

2. What is the difference between security logs and system logs?

   _____
   _____
   _____
   _____

**Exercise 1.6: Object Access Auditing**

Windows allows the auditing of *object access* thus allowing the audit of successful as well as failed accesses. To enable object access auditing, do the following:

1. Log in as *StudentAdmin*
2. Open the event Microsoft Management Console. (Start->Control Panel ->System and Maintenance ->Administrative tools -> Local Security Policy)
3. Select *Local Policies* and then *Audit Policy*
4. Get properties of *Audit Object Access*
5. Checkmark both “Success” and “Failure” in the properties window
6. Click OK to apply the changes.

**Questions**

1. Why are audit policies disabled by default?

   _____
   _____
   _____
   _____

**Exercise 1.7: Audit Collection**

Now that object access auditing has been enabled on the system. You can define which objects should be audited. The objective of this exercise is to audit the successful *create* and *delete* activities over a set of files and folders for all authenticated users on the machine.

1. Logon as *StudentAdmin*
2. Open the *C:\doctemp* folder and click the *Organize* button, select the *New Folder* option and create a new folder with the name *folder2_<gr#>*
3. Select the folder you just created and open its *Properties* window by right clicking on it and selecting the *Properties* option.
4. Select the *Security* tab and click on the *Advanced*
5. Select the *Auditing* tab and *Continue*.
6. Click the *Add*.. button and add the *group<gr#>B* user. Be sure to press the *Check Names* button before pressing OK.
7. You should see a list of the events that can be audited when the user accesses the folder you are working on. Which events would you consider of interest to audit? Explain your reasoning for at least 3 events. Please consider that auditing events generates a lot of information which tends to generate large log files that an administrator will later have to review, so try to be specific in your choices of events to audit.

   _____
   _____
   _____
   _____

8. For this exercise mark the *List folder/Read Data* and *Create files/write data* under the Successful column. Click on OK and approve changes on all windows.
9. Log off from *StudentAdmin* and log in as the *group<gr#>B* user.
10. Use *Notepad* to create a .txt document and save it in the *C:\doctemp\folder2_<gr#>* folder with *file5_group<gr#>.txt* as the filename.
11. Open the *C:\doctemp\folder2_<gr#>* folder and verify that you can access the file you just created
12. Log off *group<gr#>B* and log in as the *StudentAdmin* user.
13. Open the event viewer. (Start->Control Panel->System and Maintenance ->Administrative tools -> View event logs)
14. Once in the Event Viewer, select *Windows Logs* category (left side of the window) and select the *Security* category

15. We will search the Security log for the audit entries related to the *group<gr#>B* user. To do this, in the *Actions* panel of the window (right side of the window) select the *Find…* command and write *group<gr#>B* in the find box.
16. Study the entries in the security log that don't belong to Logoff or Log on attempts. In particular study the ones that belong to File System access and that register in the Access Request Information a Read Data or ListDirectory event. You might have to go through several log entries until you find the ones that meet this criteria. What is the event ID registered for these entries?

    _____
    _____
    _____
    _____

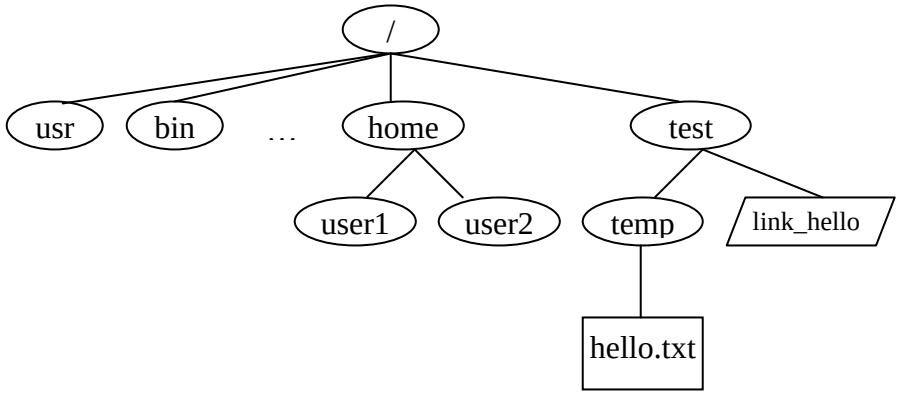
17. To finish this part of the lab, do the steps mentioned in Exercise 1.6 but **disable** object access auditing.

**Restoring the system**

Log in as *StudentAdmin* and delete the user accounts that you created for this lab. Be careful to not delete any user accounts that you did not create; otherwise you will be held responsible for any information loss.

# IV. UNIX

## File Hierarchy




The Unix file system is organized as a hierarchy with the root (/) at the highest level from which directories and files may exist. Typically, some of the directories that may occur under the root are *usr*, *bin*, *sbin*, *home*, *var*, *boot*, *dev*, *etc*, etc. In the figure shown below, *user1* and *user2* are sub-directories under home. *hello.txt* is a plain-text file and *link_hello* is a *link*ing file that points to *hello.txt*. In order to access the file */test/temp/hello.txt*, the system begins its search from the root(/) folder and then to *test* and *temp* folders consecutively and then finally the file *hello.txt*.

## Ownership and Permissions

Ownership of files in UNIX can be viewed in one of three ways: *owner* (creator), *group* or *others*. Using this simple notion of ownership access to files can be controlled associating unique user ID (UID) and group ID (GID) with twelve permission bits for each file. Typically these bits are divided into three sets of three bits and three extra bits as shown in table below.

| Permission Bits | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Extra | | | owner | | | group | | | others | | |
| su | sg | t | r | w | x | r | w | x | r | w | x |

r, *w* and *x* bits stand for read, write and execute bits for each of the owner, group and others permissions. *su*, *sg* and *t* stand for set_user_id, set_group_id and sticky bits. These 4 sets of bits are often represented in their octal digits. For example, "100 111 101 101" is represented as "4755." When the *su* bit is set, whosoever executes the file, the UID of the process will be the owner of the file. Similarly, if *sg* is set, the GID of the process will be the owner of the file.

## Lab procedures

*Note: To enter the commands required for this assignment you will need to activate a text terminal application to enter the required commands. If you want to work inside Fedora's graphical environment look for and activate the Terminal application in one of Fedora's main menus. Otherwise, to exit the graphical environment and enter directly to a text terminal, press Ctrl-Alt-F3. To return to graphical mode press Ctrl-Alt-F7.*

### Exercise 2.1: Setting up File Structure and User space

The objective of this exercise is to setup the file hierarchy structure and the user accounts that are required for the exercises in this section. The **su** command is used to switch users.

NOTE: Ideally, after the purpose of switching to a user is accomplished, it is better to "exit" from that user. However, for the purposes of the lab, su has been used repeatedly to indicate switching of users.

1. Login as root (password = "security")
2. Use **useradd** command to create two new users ***user1*** and ***user2*** as follows:
   a. `useradd user1 -g users`
   b. `useradd user2 -g users`
3. Use **passwd** command to set the password for the users you created (required in the case you want to log in). For convenience, set the passwords the same as the usernames. You need to retype the passwords and ignore password warnings:
   a. `passwd user1`
   b. `passwd user2`
4. Check user information with the **id** command. Note the uid, gid for each output.
   a. `id user1`
   b. `id user2`
5. Create directory structure
   a. `mkdir /test`
   b. `mkdir /test/temp`
6. Switch user roles as *user1* and then back to *root* using the su command
   a. `whoami`
   b. `su user1`
   c. `su` **OR** `su root`
7. Create a new file as root user
   a. `touch /home/user2/HelloWorld`
   b. `ls –l /home/user2/HelloWorld` (observe owner and group)
8. Change group ownership as well as user ownership of the file
   a. `chgrp users /home/user2/HelloWorld`
   b. `chown user2:users /home/user2/HelloWorld`
   c. `ls –l /home/user2/HelloWorld` (observe owner and group)

**Exercise 2.2: Differences in File and Folder Permissions**

The objective of the following exercises is to see the differences between file and folder (directory) permissions. The **chmod** command will be used to change file and directory permission to demonstrate the slight differences in permissions for files and directories.

1. Observe the result of **ls** and **cd** commands
   a. `cd /`
   b. `ls –l`
   c. `ls -al /home`
   d. What are the directory permissions for ***user1***, ***user2*** and ***test*** directories?

      _____
      _____
      _____

   e. Switch to ***user1*** using su user1
   f. `ls -al /home/user2` (Can you list directory?)
   g. `cd /home/user2` (Can you change directory?)

2. Change directory permissions of ***user2*** directory and try again as ***user1***.
   a. `su root`
   b. `chmod 740 /home/user2`
   c. Repeat steps *1e* to *1g* (Can you list or change directory?)
   d. `su root`
   e. `chmod 750 /home/user2`
   f. Repeat steps *1e* to *1g* (Can you list or change directory?)
   g. `touch /home/user2/hello12.txt`
      (Can you create new file?)
   h. `su root`
   i. `chmod 770 /home/user2`
   j. `su user1`
   k. Repeat step *2g*. (Can you create new file?)
   l. `ls –l /home/user2`

***Alternative syntax for chmod command***

The access permissions for the file *hello.txt* is to set the *su* bit only, allow all access permissions to owner, read and execute rights to the group and only read rights to others. In other works the 12 bit permission required on the file *hello.txt* is as follows: "100 111 101 100." This can be achieved in several ways using **chmod** command:

1. `chmod 4754 hello.txt`
2. `chmod u+srwx g+rx o+r hello.txt`
3. `chmod u=srwx, g=rx, o=r hello.txt`

**Exercise 2.3: New text files and linking files**

Unix supports two kinds of link files--a *hard link* and a *symbolic link*. A *hard* link is a file with the actual address space of some ordinary file's data blocks. A *symbolic* link is just a reference to another file. It contains the pathname to some other file. (It's a sort of shortcut to access a file)

1. In the /test/temp/ directory, as root user, create a new text file ("hello.txt") and fill it with some text.
   a. `echo something > /test/temp/hello.txt`
2. Create a link ***link_hello*** in the ***test*** folder pointing to ***hello.txt*** in the ***temp*** folder (refer to file structure in introduction)
   a. `cd /`
   b. `ln -s /test/temp/hello.txt /test/link_hello`
   c. Is there any difference in file permissions of *link_hello* and *hello.txt*?

   _____
   _____
   _____

   d. `cat /test/link_hello` --what is the output?

   _____
   _____

**Exercise 2.4: Default file permissions and Group access control**

Whenever a new file is created a default set of permissions is assigned to it. Whatever the permissions are, the UNIX system allows the user to filter out unwanted permissions set by default. This default setting can be set by the user using the **umask** command. The command takes the permissions set during creation of file and performs a bitwise AND to the bitwise negation of mask value. Some common umask values are 077 (only user has permissions), 022 (only owner can write), 002 (only owner and group members can write), etc.

1. In a terminal window, make sure you are a root user. If not the root user, then switch back to root user.
2. Use *umask* command to check the current mask permission and assign a new mask.
   a. `umask`
   b. What is the current mask? How is it interpreted? (try umask –S or the man pages)
   c. `cd /test`
   d. `touch testmask1`
   e. `ls -al`
   f. What are the permissions of the file *testmask1*
   g. `umask 0077`
   h. `touch testmask2`
   i. Now what are the permissions of the file *testmask2*

3. What is the effect of setting mask value to 0000?

   _____
   _____
   _____

4. The risks of setting the extra bits will be covered in Exercise 2.5 which shows that extra bits should not be set, in general. What should be the umask value to ensure that the "extra bits" can never be set (i.e. assigned '1' value)?

   _____
   _____
   _____

**Exercise 2.5: *setuid* bit, *setgid* bit and *sticky* bit**

As explained in the *ownership and permission* section, the highest three bits of the permission value of a file represent the *setuid* bit, *setgid* bit and the *sticky* bit. If the *setuid* bit is set then the uid will always be set to the owner of the file during execution. If the *setuid* bit is not set then the uid will be the user who executes the process. Similarly, if the *setgid* bit is set then the gid will be set to the group that owns the file during execution. If the *setgid* bit is not set then the gid will be the group that executes the process. The sticky bit is set to keep processes in the main memory.

In the following exercise, the objective is to demonstrate how processes are affected when the *setuid* bit is set. The exercise must be begun with root privileges.

a. `which touch`
b. `ls –l /bin/touch`
c. `chmod 4755 /bin/touch`
d. `ls –l /bin/touch`
e. `ls –l /home/user2`
f. `chmod 700 /home/user2/HelloWorld`
g. `ls –l /home/user2` (observe timestamp and permissions)
h. `su user1`
i. `touch /home/user2/HelloWorld`
j. `ls –l /home/user2` (observe timestamp, is it updated?)
k. `su root`
l. `chmod 0755 /bin/touch`
m. `su user1`
n. `touch /home/user2/HelloWorld`

Why permission denied, while previously allowed?

_____
_____
_____
_____
_____
_____

**RESTORE THE SYSTEM**

After the series of exercises, it is important that the system is restored to its normal state so that other students may undertake the exercises again. Below are the set of commands that you should issue to restore the system to its original form (commands No. 4 and 5 could take some time).

```
1. su root
2. umask 0022
3. chmod 0755 /bin/touch
4. userdel user1
5. userdel user2
6. rm –rf /home/user1
7. rm –rf /home/user2
8. rm –rf /test
9. rm –rf /home/test/
```

Questions

1. How different is the UNIX access control architecture from that of Windows based platforms?
2. How different is access control management in Windows, compared to UNIX? Which would you say is easier? Which is more efficient?