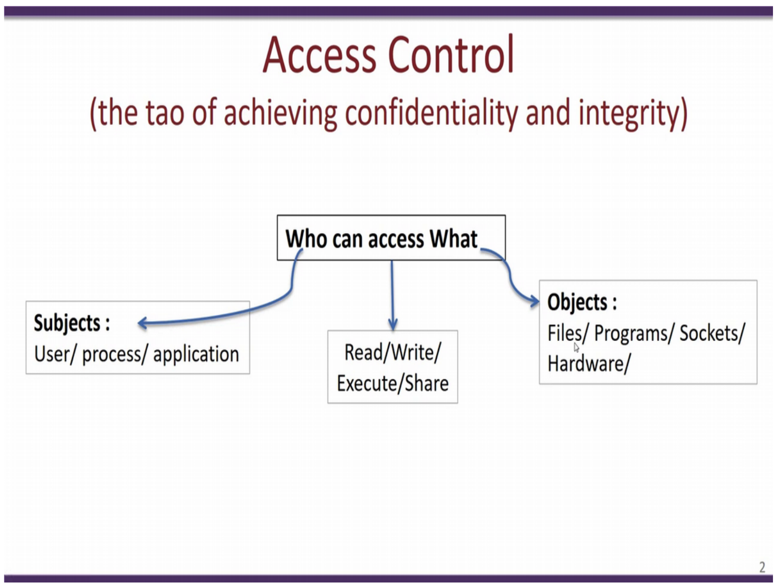
Information Security 5 Secure System Engineering
Professor Chester Rebeiro
Indian Institute of Technology Madras
Access Control
Mod05_Lec28

Hello and welcome to this lecture in the course for secure system engineering, in the previous lecture and the lecture that we have seen so far we were considering two executables, we had looked at how vulnerabilities in the executable can be used to create different exploits, in this lecture and the lectures that follow we have been looking at access control mechanisms, so for any system to be, design to be secure we would have to ensure some access control policies, essentially for any secure systems there are three specific goals it is known as the CINA, confidentiality, integrity and availability.

So the access control is used to obtained the confidentiality and the integrity aspects of a secure system, so in this particular lecture we have been looking at fundamentals about what access control is and how it can be implemented at different levels.

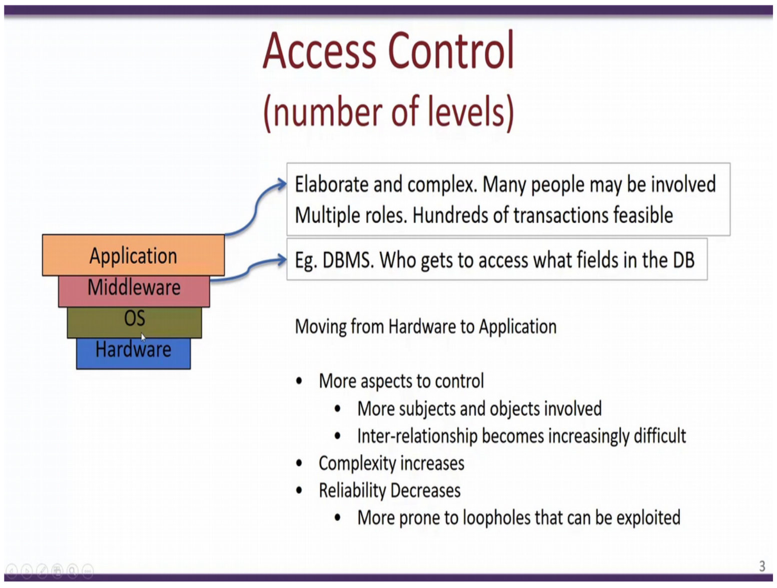
(Refer Slide Time: 1:23)


Access Control
(the tao of achieving confidentiality and integrity)

Who can access What
Subjects : User/ process/ application
Read/Write/ Execute/Share
Objects : Files/ Programs/ Sockets/ Hardware/
2

At a very high level access control defines who can access what, the who over here are the subjects and what are the objects and axis is define as various operations these subjects can perform on these objects, for example subjects can be user or process or an application, while objects can be files, programs, sockets or hardware and access would be read, write, execute or share, so for example access control can define if a user can read, write or execute a particular file, program or a socket in this system.

(Refer Slide Time: 2:09)



Access Control
(number of levels)

Application
Middleware
OS
Hardware

Elaborate and complex. Many people may be involved Multiple roles. Hundreds of transactions feasible

Eg. DBMS. Who gets to access what fields in the DB

Moving from Hardware to Application

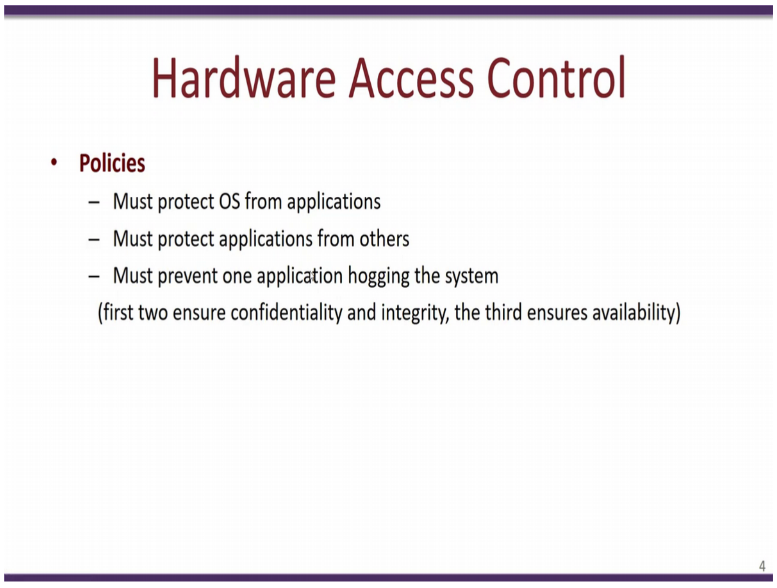
- More aspects to control
  - More subjects and objects involved
  - Inter-relationship becomes increasingly difficult
- Complexity increases
- Reliability Decreases
  - More prone to loopholes that can be exploited

3

So access controls are available in a number of levels in the system, so you have access control policies which are present at the hardware, operating system, middleware like DBMS and also in various applications, so as we go upwards from the hardware towards the application, the access control mechanisms become more and more complex, so in hardware for example a simple access control mechanism to prevent say reading or writing from one memory location would require far less amounts of overheads and far more simple compare to the access control mechanisms that are implemented in the hardware.

So as we move from the hardware to the application there are more aspects and more finer expects to be control, so the complexity increases the same way and also the reliability and the guarantees that can be provided by the access control is more difficult to achieve, so essentially access control mechanisms provided in the application or middleware are more prone prove loopholes and can be exploited, on the other hand finding a loophole or a problem or a vulnerability in the hardware is far more difficult, so what will be seeing in this particular lecture is some of the hardware level access control mechanisms and also the OS level access control mechanisms.

(Refer Slide Time: 3:47)


Hardware Access Control

- **Policies**
  - Must protect OS from applications
  - Must protect applications from others
  - Must prevent one application hogging the system

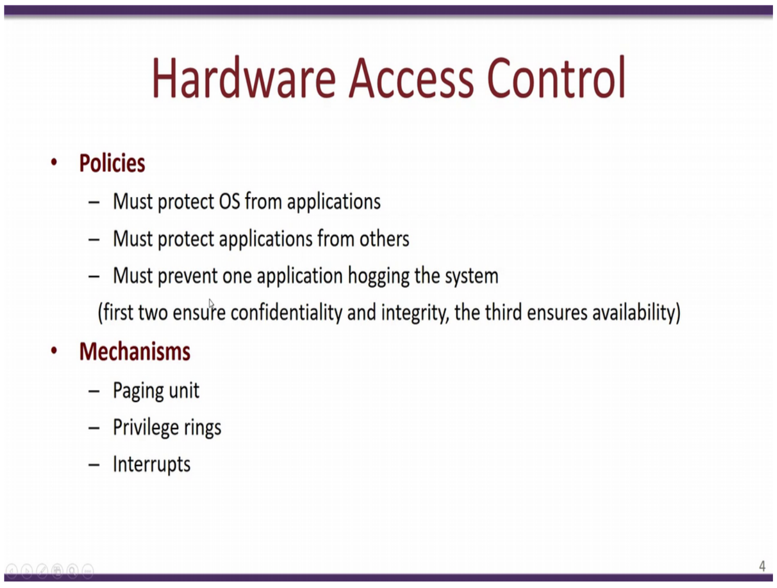
  (first two ensure confidentiality and integrity, the third ensures availability)

4

So we will start with the hardware access control, so the policies with which hardware access control are built are these, first the hardware should ensure that the operating system is protected from the applications, second it should have techniques and policies to ensure that one application cannot interfere with an order application, third there should be policies to ensure that one application does not hogg the entire system, so these two things the first and second would ensure confidentiality and the integrity.

So for example the hardware provides mechanisms by which the applications cannot modify or view the OS executable and the OS code, on the other hand the third mechanism which is provided by the hardware ensures availability, so it will ensure that one application does not utilise the entire hardware all the time.

(Refer Slide Time: 5:03)



Hardware Access Control

- **Policies**
  - Must protect OS from applications
  - Must protect applications from others
  - Must prevent one application hogging the system

  (first two ensure confidentiality and integrity, the third ensures availability)
- **Mechanisms**
  - Paging unit
  - Privilege rings
  - Interrupts

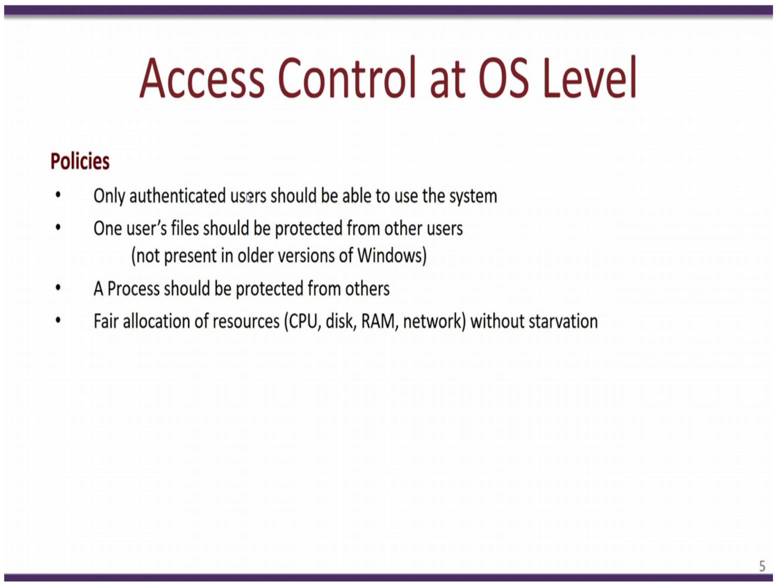
4

So in order to achieve this three objectives, so in order to achieve this three policies what is implemented in hardware or the mechanisms, so we have paging units, privilege rings and interrupts, so the paging unit would be able to isolate one application address space from another application thus using the paging unit we would be able to protect one application on the other, there is also the privilege rings, so privilege rings in Intel x86 there are privilege rings 0, 1, 2 and 3, so these rings would achieve protection of the operating system from applications.

So what for example in x86 the operating system runs in ring zero and the applications run in ring three, the hardware takes care of the fact that applications in ring three cannot directly access, cannot directly read or write to any application running in ring zero, since the OS runs in ring zero, therefore it is completely isolated from the user level applications which are running in ring three.

So the third mechanism is known as interrupts, so interrupts are used to achieve this particular policy where it can use be used to prevent one application from hogging the system, so the interrupts can be used by schedulers to obtain contact switching, so when at interrupt occurs the scheduler can pick a process and preventing the existing process, thus achieving the fact that one application is not hogging the entire system.

(Refer Slide Time: 7:05)



Access Control at OS Level

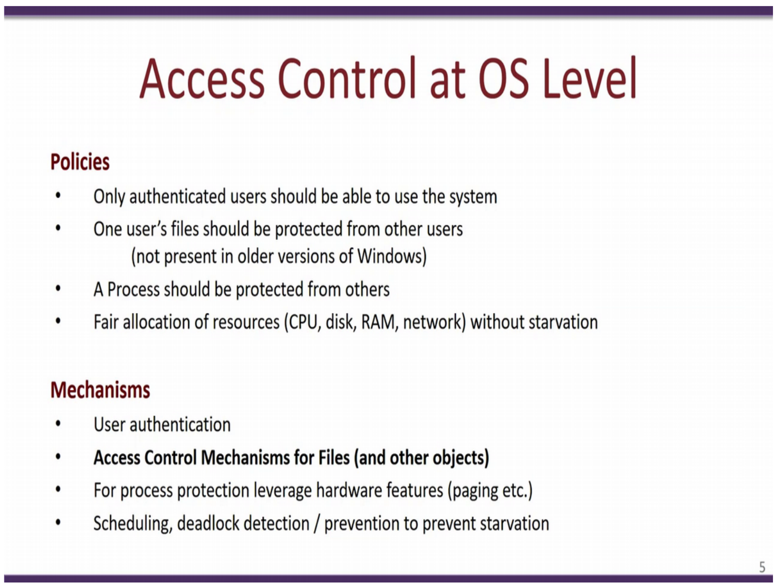
Policies

- Only authenticated users should be able to use the system
- One user's files should be protected from other users
  (not present in older versions of Windows)
- A Process should be protected from others
- Fair allocation of resources (CPU, disk, RAM, network) without starvation

5

Now at the OS level there are several more different types of policies, some of these policies are implemented with the help of the hardware, while some are very specific to the operating system, so some of the access control policies supported by a typical operating system are as follows, one particular policy is to ensure that only authenticated users should be able to access the system, so as we know in order to implement this we have mechanisms such as the login and password checking and only if users have a valid password and then would they be able to access the system.

(Refer Slide Time: 7:53)



Access Control at OS Level

Policies

- Only authenticated users should be able to use the system
- One user's files should be protected from other users
  (not present in older versions of Windows)
- A Process should be protected from others
- Fair allocation of resources (CPU, disk, RAM, network) without starvation

Mechanisms

- User authentication
- **Access Control Mechanisms for Files (and other objects)**
- For process protection leverage hardware features (paging etc.)
- Scheduling, deadlock detection / prevention to prevent starvation

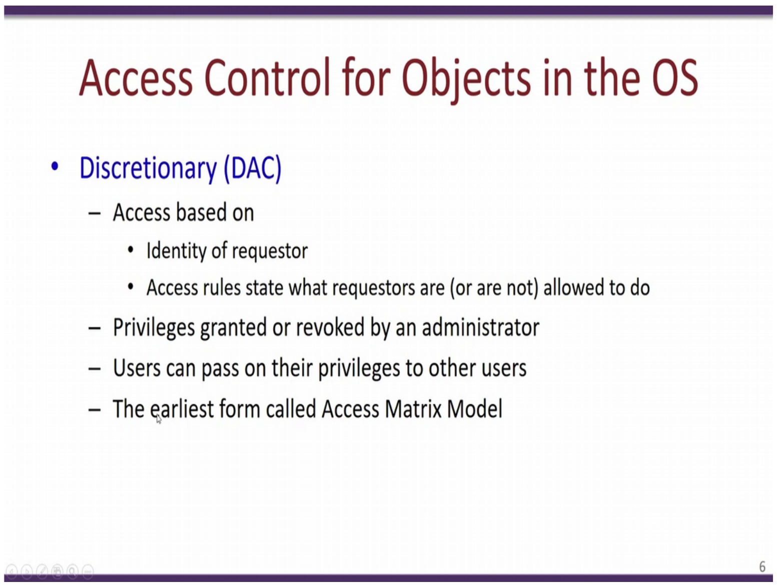
5

So this particular policy is achieved by user authentication, another aspect which many operating system support is to ensure that one users file should be protected from the other users, so the prior operating systems such as MS-DOS and older versions of Windows do not support is particular features, however in most modern operating systems we have access control mechanisms for files which would permit users to actually set privileges and various access control policies like read, write and execute to ensure who can read files, who can write or modified their files and who can execute and corresponding programmes.

In addition to the file system and user or authentication type of access control mechanisms, operating systems along with the hardware also plays a role in protecting one processes memory from other processes, so essentially this is done by the page table mechanisms, while the operating system creates this page tables and configures the hardware to manage this table page tables, it is the hardware which actually ensures the isolation between the various pages and also ensures that the access control policies such as read, write and execute to a particular page is very fight at runtime.

Another access control policy which typical OS is implemented is the fair allocation of resources such as CPU, disk, RAM and network, so that one process is not starved of a particular resource, in order to implement this particular access control policy various mechanisms such as scheduling deadlock detection and prevention are used in order to prevent starvation of a particular process and ensure that all resources are optimally shared among the various processes in the system.

(Refer Slide Time: 10:03)



Access Control for Objects in the OS

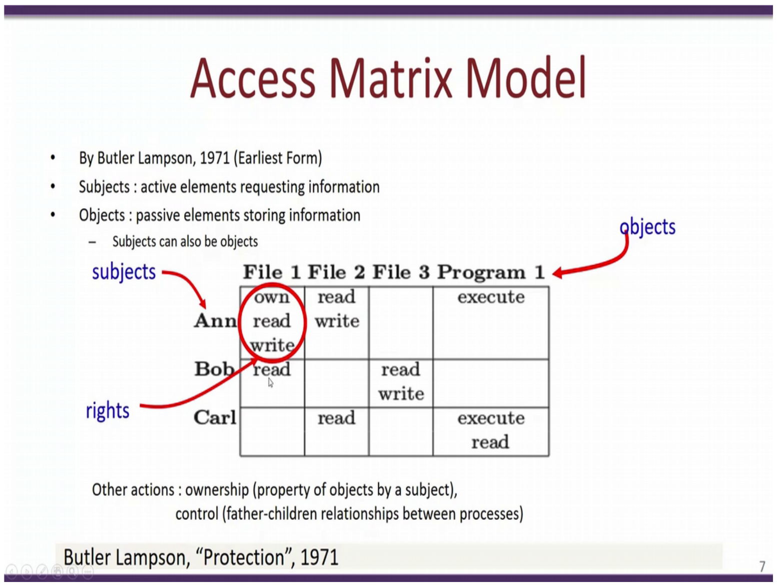
- Discretionary (DAC)
  - Access based on
    - Identity of requestor
    - Access rules state what requestors are (or are not) allowed to do
  - Privileges granted or revoked by an administrator
  - Users can pass on their privileges to other users
  - The earliest form called Access Matrix Model

6

So we will look at one of the most common access control mechanism for the operating system, this is known as the discretionary access control or DAC, in DAC access control policies, the access is based on the identity of a requester, there are some access rules that are defined and based on the identity of the requester, these rules are verified and only if the verification passes only then will this requester which is the subject would get access to the corresponding object.

So in a typical UNIX like operating system all of this privileges of who cannot access what is typically managed by the administrator, so the administrator such as the root user of the particular Linux or UNIX system can decide what privileges can be granted or revoked, users would be able to pass on privileges from one user to another, so we will look at one of the earliest forms which is known as the access matrix model.

(Refer Slide Time: 11:16)



Access Matrix Model

- By Butler Lampson, 1971 (Earliest Form)
- Subjects : active elements requesting information
- Objects : passive elements storing information
  - Subjects can also be objects

	File 1	File 2	File 3	Program 1
Ann	own read write	read write		execute
Bob	read		read write	
Carl		read		execute read

subjects
objects
rights

Other actions : ownership (property of objects by a subject), control (father-children relationships between processes)

Butler Lampson, "Protection", 1971

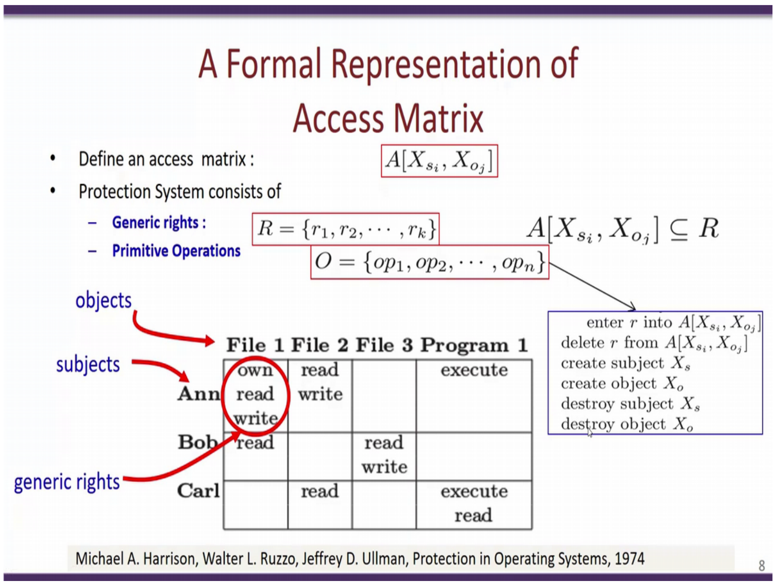
7

So the access matrix model looks like this, so it was proposed by Butler Lampson in 1971, where he had define subjects and objects, so essentially subjects are the active elements in the system which request to have access to specific objects, objects on the other hand are passive elements and used to store informations, so they could be like files or programs or any other thing, so for example a program can be a object and you could have access control policies to determine who can run that particular program.

At the same time a subject can be a process and you could have access control mechanisms for that particular process to determine what files that process can access, what other programs that file can execute and so on, so the entire access matrix model is represented as a matrix like this, on each row corresponds to a different subject for example over here we have three subjects Ann, Bob and Carl, while the columns on the other hand represent objects.

So we have objects file 1, file 2, file 3 and a particular program, now corresponding to each cell in the matrix is the various rights for what that subject has on that particular object, so for example over here Ann is the subject and a file 1 is the object, Ann can read, so Ann is the owner of this file 1 and therefore Ann can read and write to file 1, on the other hand Bob can only read to file 1, Bob is not the owner and Bob cannot right to the file 1.

(Refer Slide Time: 13:17)



A Formal Representation of Access Matrix

- Define an access matrix : $A[X_{s_i}, X_{o_j}]$
- Protection System consists of
  - **Generic rights :** $R = \{r_1, r_2, \cdots, r_k\}$
  - **Primitive Operations** $O = \{op_1, op_2, \cdots, op_n\}$

$$A[X_{s_i}, X_{o_j}] \subseteq R$$

enter r into $A[X_{s_i}, X_{o_j}]$
delete r from $A[X_{s_i}, X_{o_j}]$
create subject X_s
create object X_o
destroy subject X_s
destroy object X_o

	File 1	File 2	File 3	Program 1
Ann	own read write	read write		execute
Bob	read		read write	
Carl		read		execute read

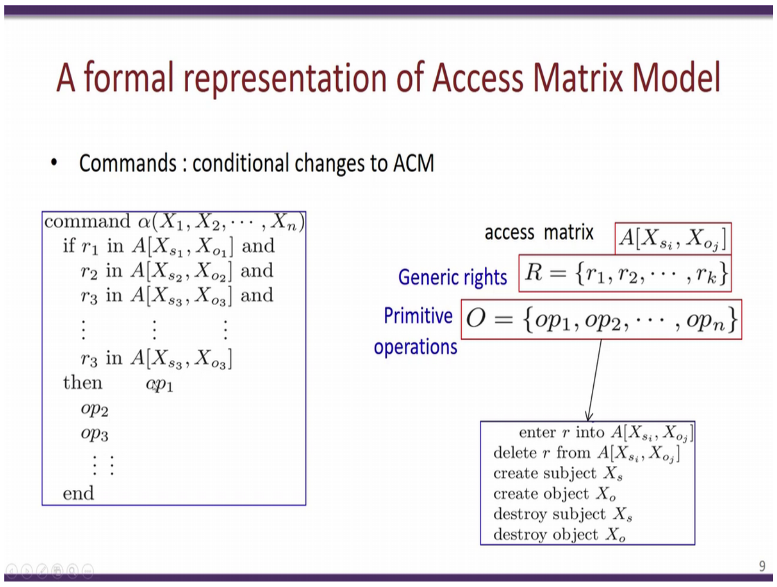
Michael A. Harrison, Walter L. Ruzzo, Jeffrey D. Ullman, Protection in Operating Systems, 1974

8

A more former representation of this access matrix is shown over here where we define the access matrix as a matrix A comprising of subjects and objects, further we have generic rights which are present, so this is define by the set R comprising of various different rights, so here we show K different rights, R1 to RK and we have some primitive operations which is defined by the set O, so these operations there are six of them which would defined what operations can be performed on this particular matrix.

For example you can enter a right R into a particular cell A of X SI, A of X OJ, for example I can enter a right to read in a particular cell such as this, similarly you can delete a right from a particular cell, create subject, create object, destroy subject and destroy object, now based on this mechanism.

(Refer Slide Time: 14:23)



A formal representation of Access Matrix Model

- Commands : conditional changes to ACM

```
command $\alpha(X_1, X_2, \cdots, X_n)$
  if $r_1$ in $A[X_{s_1}, X_{o_1}]$ and
     $r_2$ in $A[X_{s_2}, X_{o_2}]$ and
     $r_3$ in $A[X_{s_3}, X_{o_3}]$ and
     $\vdots$        $\vdots$        $\vdots$
     $r_3$ in $A[X_{s_3}, X_{o_3}]$
  then    $op_1$
     $op_2$
     $op_3$
      $\vdots$ $\vdots$
  end
```

access matrix $A[X_{s_i}, X_{o_j}]$

Generic rights $R = \{r_1, r_2, \cdots, r_k\}$

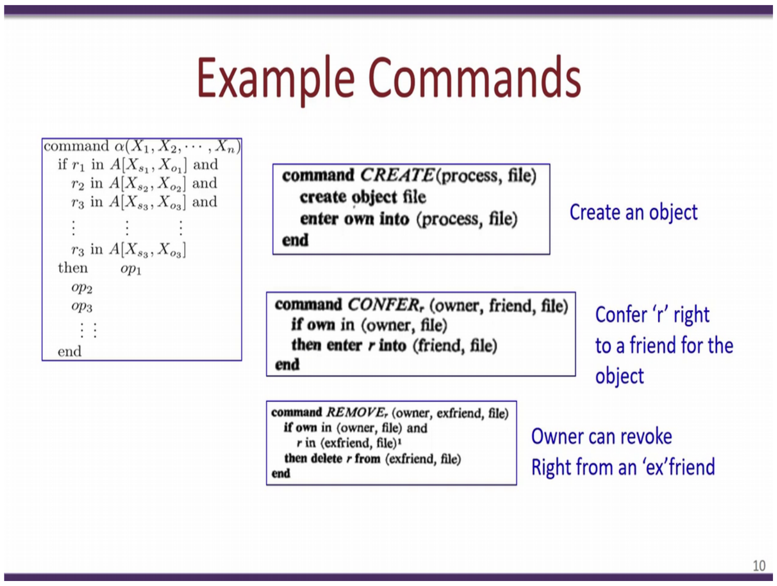
Primitive operations $O = \{op_1, op_2, \cdots, op_n\}$

```
enter $r$ into $A[X_{s_i}, X_{o_j}]$
delete $r$ from $A[X_{s_i}, X_{o_j}]$
create subject $X_s$
create object $X_o$
destroy subject $X_s$
destroy object $X_o$
```

9

We can define commands, so this commands is what is used to specify the access control for that particular system, a typical command would look something like this, so you have a command alpha and unless certain rights are available only then can certain operations be performed.

(Refer Slide Time: 14:46)



Example Commands

```
command $\alpha(X_1, X_2, \cdots, X_n)$
  if $r_1$ in $A[X_{s_1}, X_{o_1}]$ and
     $r_2$ in $A[X_{s_2}, X_{o_2}]$ and
     $r_3$ in $A[X_{s_3}, X_{o_3}]$ and
     $\vdots$        $\vdots$        $\vdots$
     $r_3$ in $A[X_{s_3}, X_{o_3}]$
  then    $op_1$
     $op_2$
     $op_3$
      $\vdots$ $\vdots$
  end
```

```
command CREATE(process, file)
  create object file
  enter own into (process, file)
end
```

Create an object

```
command CONFER_r (owner, friend, file)
  if own in (owner, file)
  then enter r into (friend, file)
end
```

Confer 'r' right to a friend for the object

```
command REMOVE_r (owner, exfriend, file)
  if own in (owner, file) and
    r in (exfriend, file)¹
  then delete r from (exfriend, file)
end
```

Owner can revoke Right from an 'ex'friend

10

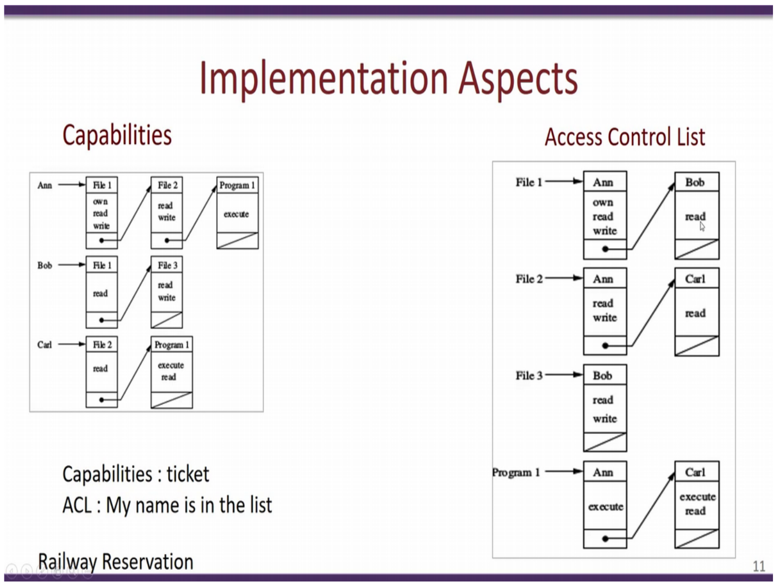
So we will take a few examples of this, for example you have this create command, so this create command takes a process end file and the command is as follows, so create object file and enter own into process, file, in this way what we see is that a process can create a file which would mean that in the cell corresponding to this process and this file the ownership attribute is added on, similarly this is another example of confer right, so in order to run this command we first check if the owner is in fact the owner of the file, this is checked by looking into the access matrix corresponding to owner and file and determining whether own is present in that particular cell.

If the own is present then you have the right to add R into that friends entry, so another example is this confer right, where you can confer right R to a friend or a particular object, so in this particular example we shown how someone can confer the a particular right R for a file to a friend, so the first thing to do is to check whether the owner of the file is the right owner, we do this by looking at the access matrix and noting whether own is present in that particular cell corresponding to owner and file and if this is true then we can add the right R into friend, file into the cell corresponding to friend, file.

This is another example of a command which is to remove a right from an ex-friend from a particular object in this case a file, so what we check over here is that only the owner can remove the right from a particular subject for a particular file, so only if the owner is the rightful owner of the file only then the this particular command will succeed, so what we see over here is based on this definition of the subject, object or the access matrix and the various primitive operations, one could create various different types of commands.

So these commands define the access control policies for that corresponding operating systems, this access matrix would be the basic building block for creating your access control policies for your operating systems, therefore we could have two operating systems using the same mechanisms based on these commands what we see is that the operating system can define how various objects can have access to the various objects resonate that system.

(Refer Slide Time: 17:42)



Implementation Aspects

Capabilities

Access Control List

Capabilities : ticket

ACL : My name is in the list

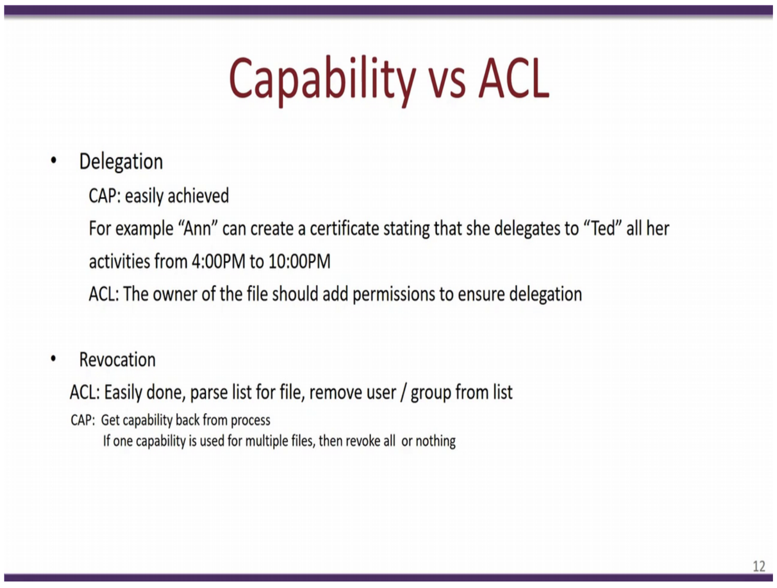
Railway Reservation

11

So in practice it is not a very optimal way to implement this access metrics model, this is because quite often the number of subjects and objects are quite large and the matrix that we obtain is highly sparse and therefore a more practical implementation is by using capabilities or access control list, in a capability based system what we have is that we have one user and we have the capabilities of the users.

So we have a user and this user owns a list of all the different capabilities that he or she processes, so for example over here and has a list of all the capabilities and has the read and write capability for file 1, she has the read and write capability for file 2 and only the execute capability for program 1 right, on the other hand the access control list is exactly the opposite, here we maintain a link list for each object, so for example file 1 can be read and written to by Ann and Ann is also the owner of this file 1 and Bob can only read the file, he does not have any other permission for this particular file.

(Refer Slide Time: 19:09)



Capability vs ACL

- Delegation
  CAP: easily achieved
  For example "Ann" can create a certificate stating that she delegates to "Ted" all her activities from 4:00PM to 10:00PM
  ACL: The owner of the file should add permissions to ensure delegation

- Revocation
  ACL: Easily done, parse list for file, remove user / group from list
  CAP: Get capability back from process
  If one capability is used for multiple files, then revoke all or nothing

12

Capabilities versus access control list have their own advantages and disadvantages, for example if you want to delegate your rights somebody else and a capability based model is much more easy, so let us say for example in an office environment and is going on vacation and she wants to delegate all her task to somebody else call Ted for some time, for example she wants to delegate all her jobs to Ted from 4 PM to 10 PM, so what she does is that she can create a certificate stating that the delegates on her jobs to Ted, the refer from 4 PM to 10 PM Ted is complete access for of all her capabilities.

So what is required is Ted having just this particular ticket which gives in right to all of Anns files, if we want to achieve the same kind of dedication with the ACL that is the access control list and it is far more difficult, the reason being doing this case Ann has to pass through every object that is present the system and just for a period of 4PM to 10 PM she has to add permissions for Ted to ensure delegation.

So this is far more difficult and just creating a certificate which gives Ted a permission to assess all of Anns files from a particular period to a particular period, another widely used functionality is that of revocation, in revocation access control list are much more efficient than the capability base model, in during revocation it is easily done in ACL because all that is required is to parse the ACL list for a particular object and remove the user or the subject of that particular list.

However if you want to actually do this using the capability base model, this could get a little bit tricky specially if you have one capability that is use to service multiple files and what we want is for that particular subject to access all the files except one, so this is very difficult to do with the capability base model.

So in this lecture we had actually looked at access control policies, we had seen how access control policies can be, are implemented in the hardware and a very primitive form of discretionary access control which is implemented in many of the operating systems, in the next lecture we will look at how this access control policies are implemented in unix or LINUX types systems and we would also looked at what is the major drawback of having discretionary access control policy mechanisms and we will actually see this could be made better why something known as a flow control policies. Thank you.