

ECS 235B Module 4

Access Control Matrix

Description

objects (entities)

	O_1	...	O_m	S_1	...	S_n
s_1						
s_2						
...						
s_n						

subjects

- Subjects $S = \{ s_1, \dots, s_n \}$
- Objects $O = \{ o_1, \dots, o_m \}$
- Rights $R = \{ r_1, \dots, r_k \}$
- Entries $A[s_i, o_j] \subseteq R$
- $A[s_i, o_j] = \{ r_x, \dots, r_y \}$ means subject s_i has rights r_x, \dots, r_y over object o_j

Example 1

- Processes p, q
- Files f, g
- Rights r, w, x, a, o

	f	g	p	q
p	rwo	r	$rwxo$	w
q	a	ro	r	$rwxo$

Example 2

- Host names *telegraph*, *nob*, *toadflax*
- Rights *own*, *ftp*, *nfs*, *mail*

	<i>telegraph</i>	<i>nob</i>	<i>toadflax</i>
<i>telegraph</i>	<i>own</i>	<i>ftp</i>	<i>ftp</i>
<i>nob</i>		<i>ftp, mail, nfs, own</i>	<i>ftp, nfs, mail</i>
<i>toadflax</i>		<i>ftp, mail</i>	<i>ftp, mail, nfs, own</i>

Example 3

- Procedures *inc_ctr*, *dec_ctr*, *manage*
- Variable *counter*
- Rights *+*, *-*, *call*

	<i>counter</i>	<i>inc_ctr</i>	<i>dec_ctr</i>	<i>manage</i>
<i>inc_ctr</i>	<i>+</i>			
<i>dec_ctr</i>	<i>-</i>			
<i>manager</i>		<i>call</i>	<i>call</i>	<i>call</i>

UNIX/Linux Access Controls

- Files: A is `~bishop/a.out` (0755), B is `/etc/passwd` (0644), H is `/home/bishop` (0711), S is `/bin/su` (4711)

	<i>A</i>	<i>B</i>	<i>S</i>	<i>H</i>
<i>bishop</i>	<i>rwxo</i>	<i>r</i>	<i>x</i>	<i>rwxo</i>
<i>zheng</i>	<i>rx</i>	<i>r</i>	<i>x</i>	<i>x</i>
<i>root</i>	<i>rwX</i>	<i>rwo</i>	<i>rwXO</i>	<i>rwX</i>

UNIX/Linux Access Controls

- Access control matrices are dynamic:
- After bishop executes `chmod 700 /home/bishop`:

	<i>A</i>	<i>B</i>	<i>S</i>	<i>H</i>
<i>bishop</i>	<i>rwxo</i>	<i>r</i>	<i>x</i>	<i>rwxo</i>
<i>muwei</i>		<i>rx</i>		
<i>root</i>	<i>rwX</i>	<i>rwo</i>	<i>rwXO</i>	<i>rwX</i>

Boolean Expression Evaluation

- ACM controls access to database fields
 - Subjects have attributes
 - Verbs define type of access
 - Rules associated with objects, verb pair
- Subject attempts to access object
 - Rule for object, verb evaluated, grants or denies access

Example

- Subject annie
 - Attributes *role* (artist), *group* (creative)
- Verb paint
 - Default 0 (deny unless explicitly granted)
- Object picture
 - Rule:
paint: 'artist' in subject.role and
'creative' in subject.groups and
time.hour ≥ 0 and time.hour ≤ 4

ACM at 3AM and 10AM

At 3AM, time condition met
ACM is:

... picture ...

... annie ...			
	paint		

At 10AM, time condition not met
ACM is:

... picture ...

... annie ...			

History

- Problem: what a process has accessed may affect what it can access now
- Example: procedure in a web applet can access other procedures depending on what procedures it has already accessed
 - S set of *static rights* associated with procedure
 - C set of current rights associated with each executing process
 - When process calls procedure, rights are $S \cap C$

Example Program

```
// This routine has no filesystem access rights  
// beyond those in a limited, temporary area
```

```
procedure helper_proc()  
    return sys_kernel_file
```

```
// But this has the right to delete files
```

```
program main()  
    sys_load_file(helper_proc)  
    tmp_file = helper_proc()  
    sys_delete_file(tmp_file)
```

- *sys_kernel_file* contains system kernel
- *tmp_file* is in limited area that *helper_proc()* can access

Before *helper_proc* Called

- Static rights of program

	<i>sys_kernel_file</i>	<i>tmp_file</i>
<i>main</i>	delete	delete
<i>helper_proc</i>		delete

- When program starts, current rights:

	<i>sys_kernel_file</i>	<i>tmp_file</i>
<i>main</i>	delete	delete
<i>helper_proc</i>		delete
<i>process</i>	delete	delete

After *helper_proc* Called

- Process rights are intersection of static, previous “current” rights:

	<i>sys_kernel_file</i>	<i>tmp_file</i>
<i>main</i>	delete	delete
<i>helper_proc</i>		delete
<i>process</i>		delete

Quiz

In an access control matrix, do the rights “r”, “w”, and “x” represent “read”, “write”, and “execute” permissions, respectively?

- Yes, because the permission symbols are tied to those permissions (“r” for “read”, “w” for “write”, “x” for “execute”).
- Possibly; the meanings of the permission symbols depends upon the instantiation.
- No, because the meanings of the permission symbols changes as the matrix evolves, so you cannot say what the symbols mean; you can only manipulate them based on the given commands.