TA: George Iordache
CC all reviews to: georgei+509r@cs.sunysb.edu

ACM
- undecidable
- ACL lists, capabilities

Bell-Lapudula
- read
- write
- MAC vs DAC

Bell-Lapudula
ex. (Top Secret, {Area 51})
(Confidential, {})
- ordering on labels

"Read down, write up"

For read:
Process with ℓ can read process with ℓ' if $\ell' \leq \ell$

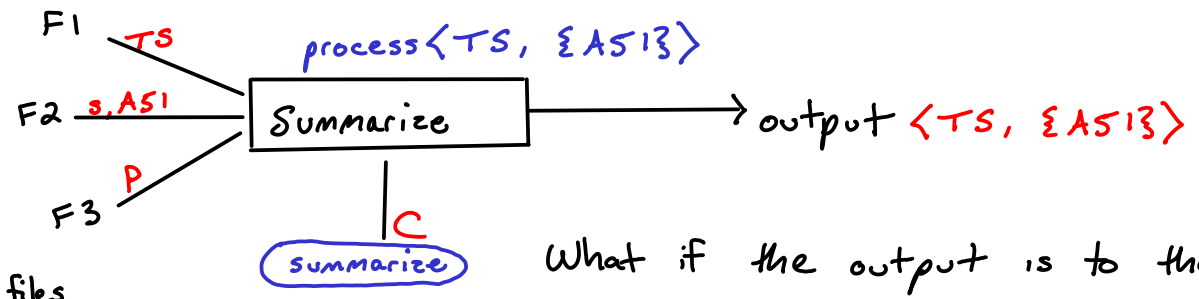
For write:
Process with ℓ can read process with ℓ' if $\ell \leq \ell'$

Basic Security Theorem
If Σ starts in secure state and we use read down / write up, then the system can never reach an insecure state

Two strategies:

① fixed process label

② process label = lab(labels of files read)



What if the output is to the screen, and the user doesn't have clearance?

MAC- Mandatory Access Control (system tells you)

DAC- Discretionary Access Control (you tell system)

Declassification

- special, trusted declassification process that is allowed to break the read down/write up rule

Combine policy and privilege

ex: salaries (secret) → averager (declassifier) → average salary (public)

ex: separation of privilege

- declassifier shouldn't have bugs
- no theorems :(

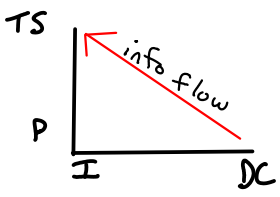
Biba (Bell-Lapudula upside down)- Read up, write down

Double Checked (DC)

Reliable Witness

Anonymous Tips

Internet (I)



Over time, info is

- less reliable
- more secret

Role-Based Access Control (RBAC)

- rights with roles in organization
  - each user is assigned set of roles

University

role	rights
student	submit hw, view grade
professor	read hw, write grade
dean	change grade

hierarchial role:

prof ≤ dean

	submit HW	read HW	write grade	change grade
student				
professor				
dean				

Static separation of duties

No user is both professor and student

Capabilities & the Confused Deputy

- Process possesses a set of rights
- OS automatically allows access if any one of the process' rights allows it
- Process can't decide which write is used for access

Ex: Compiler that stores billing info in special file /var/cc/txns
writable only by processes in CC group

is setgid cc

% cc hello.c -o hello.exe ← should only write to the file if user can

% cc hello.c -o /var/cc/txns

Fixes:

- Check args for "/var/cc/txns"
  - doesn't scale as number of "special" files grows
  - can get out of sync
  - symlink } canonicalization
  - /var/../

- Separate priviledge management from priviledge enforcement.
  - App manages privileges
  - OS checks them
    - App must provide proof that access is ok to the OS

Compiler source:

```
// write billing info
open ("/var/cc/txns" cctoken)
  ...

open (outfile, user token)
```

↑ fails if outfile = /var/cc/txns

- Compiler Holds
  - two tokens: cc token, user token

Capability Implementations

- Cryptography
- Unforgable Pointers - unix file handles