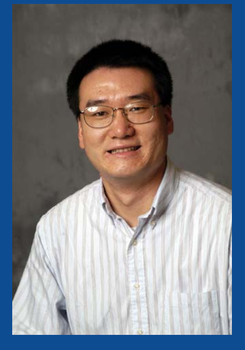
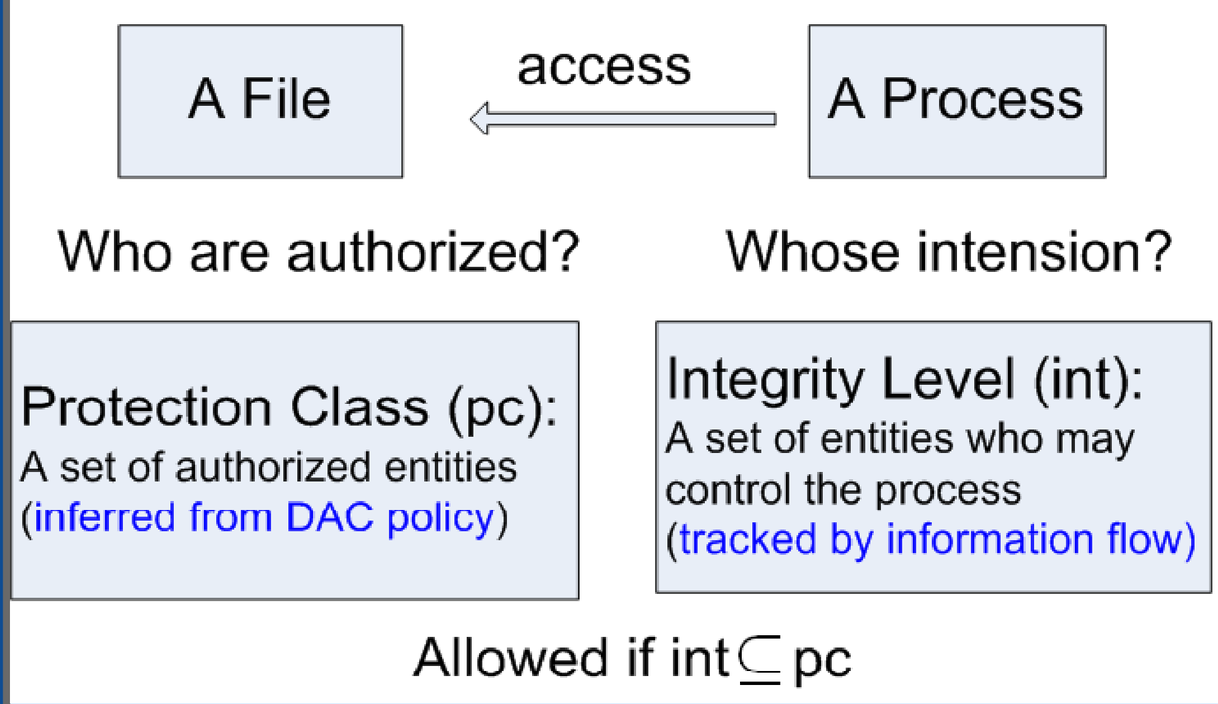
Trojan Horse Resistant Discretionary Access Control

Ninghui Li, Ziqing Mao, Hong Chen (Purdue University) Xuxian Jiang (GMU)



- DAC: the dominant access control approach in modern OS
  - Vulnerable to Trojan horse and vulnerability exploitation
  - But easy and intuitive to use; users are familiar with it
- MAC: more secure, but difficult to configure and often break existing applications
- Can we have the best of both worlds? **Yes!**
- Consider UNIX DAC, which has two components
  - discretionary policy specification: (e.g., rwx permission bits)
  - enforcement: (e.g., effective user id):
    - **Source of UNIX DAC's weakness: unable to tell the true origin(s) of a request**
    - Because UNIX DAC enforcement tries to identify a single principle behind any request, whereas in practice multiple principals can affect a request.
- Our Solution: IFEDAC: Information Flow Enhanced DAC
  - Keep DAC's discretionary policy specification
  - Fix DAC's enforcement
    - maintains a set of principals for each request
    - Introduces an additional entity, net, representing remote attacker



Integrity Tracking Rules:

Subject Integrity Tracking[a]	
After creating the first subject s_0	$int(s_0) \leftarrow \top$
After s creates s'	$int(s') \leftarrow int(s)$
After s executes o	$int(s) \leftarrow int(s) \cup int(o)$
After s reads from the network	$int(s) \leftarrow int(s) \cup \{\text{net}\}$
After s reads from o	$int(s) \leftarrow int(s) \cup int(o)$
After s logs in a non-administrator u	$int(s) \leftarrow int(s) \cup \{u\}$
After s_1 receives IPC data from s_2	$int(s_1) \leftarrow int(s_1) \cup int(s_2)$
Object Integrity Tracking	
When o is created by s	$int(o) \leftarrow int(s)$
When $int(o)$ is not previously assigned	$int(o) \leftarrow wpc(o)$
After o is written by s	$int(o) \leftarrow int(s) \cup int(o)$

- Protection class may be different from that inferred from DAC
  - can be explicitly set by users
- File integrity level can be manually upgraded by users
  - must satisfy certain conditions
- Program Exceptions
  - RAP: maintain IL when receiving network traffic
    - trusted to process network inputs correctly
  - LSP: maintain IL when reading file and receiving IPC data
    - trusted to process file and IPC inputs correctly
  - SP: access files without satisfying the protection rules
    - trusted to process inputs correctly, or
    - attacker is unable to inject all malicious code into the AS
- Security Assumptions
  - Programs that are explicitly identified as benign are benign
    - By specifying initial integrity level and integrity upgrading
  - Programs that have exceptions are correct
- Implementation
  - Implemented using LSM
  - Use extended attributes to store file's int and pc

- A Usage Case:
  - John launches ThunderBird (T); T.IL = {john}
  - T communicates with remote mail server; T.IL = {john, net}
  - T needs to access the working directory whose wpc = {john}
    - Grant a special privilege (~/.thunderbird, write) to T's binary
  - John wants to save an email attachment to local file system
    - John has an Internet Directory ~/download
    - The directory's wpc is explicit assigned as {john, net}
  - John saves the email attachment A to ~/download
    - A.IL = {john, net}
  - John opens A using a pdf viewer V
    - V.IL = {john, net}, after V reads A
  - A is a mal-formed file and exploits a vulnerability in V
    - V cannot access system files and john's normal files
  - John saves another email attachment B ~/download
    - B.IL = {john, net}
  - John wants to install B to the system, so executes B as BP
    - BP.IL = {john, net}
    - BP cannot touch the system files, installation failed
    - BP cannot launch damage if B is a Trojan horse
  - John really trusts B and wants to install it
    - John login as an administrator
    - John explicitly upgrades B.IL to top
  - John executes B as BP'
    - BP'.IL = top, installation succeed

National Science Foundation
WHERE DISCOVERIES BEGIN

NSF Cyber Trust Principal Investigators Meeting
March 16-18, 2008
New Haven, CT

PURDUE UNIVERSITY