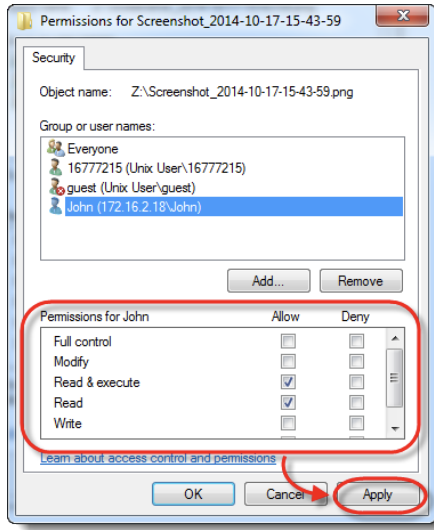
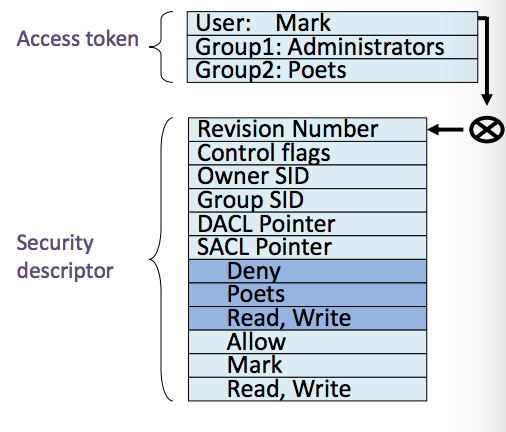
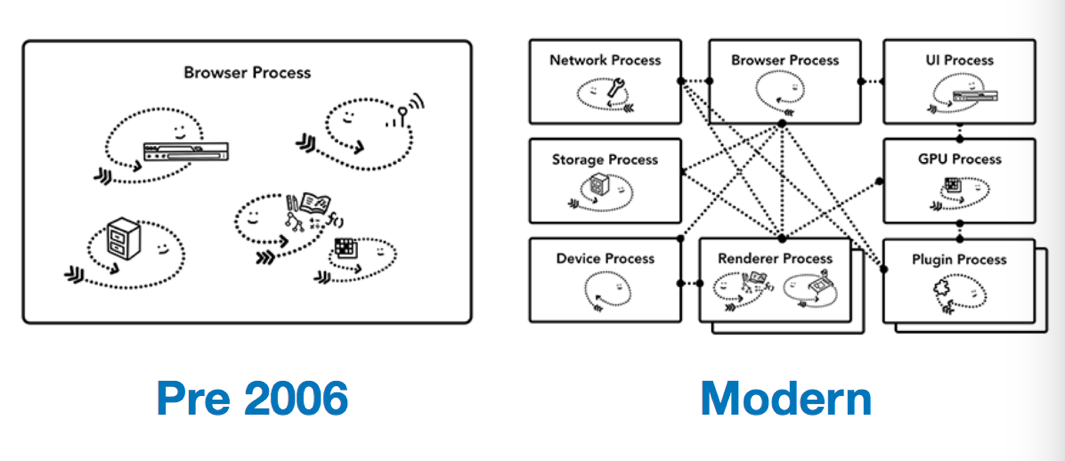
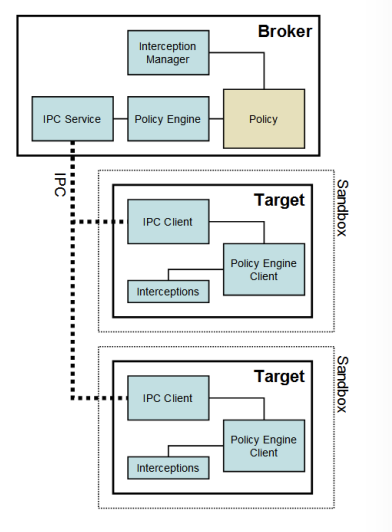
CAS CS 357

InClass Note 24

1. Flexible ACLs
2. Windows has complex access control options
3. Objects have full ACLs – possibility for fine grained permissions
4. Users can be member of multiple groups, groups can be nested
5. ACLs support Allow and Deny rules
6. 
7. Object Security Descriptors
8. Every object has a security descriptor: specifies who can perform what and audit rules
9. Contains
10. Security identifiers (SIDs) for the owner and primary group of an object
11. Discretionary ACL (DACL): access rights allowed users or groups
12. System ACL (SACL): types of attempts that generate audit records
13. Tokens
14. Every process has a set of tokens – its “security context:

* ID of user account
* Id of groups
* ID of login session
* List of OS privileges held by user/groups
* List of restrictions

1. Impersonation token can be used temporarily to adopt a different context
2. Access Request
3. When a process wants to access an object, it presents its set of security tokens (security context)
4. Windows check whether the security context has access to the object based on the object’s security descriptor
5. 
6. Capabilities vs ACLs
7. Capabilities: subject presents an unforgeable ticket that grants access to an object
8. Systems don’t care who subject is, just that they have access
9. ACL: system checks where subject is on list of users with access to the object
10. Weak protection on Desktops
11. Relying on user permission provides user with little protection against malicious applications
12. Malicious application running as you has access to all of your files
13. Adobe Acrobat can edit, delete, and encrypt/ransom all of your data
14. MAC OS App Sandbox
15. Mac OS now sandboxes many applications and mediates access to:
16. Hardware
17. Network connections
18. App data
19. User files
20. Access to any resource not explicitly requested in the project definition is rejected by the system at run time
21. Android process Isolation
22. Android uses Linux and its own kernel application sandbox for isolation
23. Each application runs with its own UID in its own VM
24. Reference monitor checks permissions on intercomponent communication
25. Modern Chrome Architecture
26. 
27. Chrome Processes
28. Browser process: controls “chrome” part of the application like address bar and bookmarks, also handles the invisible, privileged parts of a web browser like network requests
29. Renderer process: control anything inside of the tab where a website is displayed
30. Plugin process: controls any plugins used by the website, for example, flash
31. GPU process: handles GPU tasks in isolation from other processes. It is separated into different process because GPUs handles requests from multiple apps and draw them in the same surface
32. Chrome Architecture
33. Broker (Main Browser): privileged controller/supervisor of the activities of the sandboxed processes
34. Renderer’s only access to the network is via its parent browser process and file system access can be restricted
35. 
36. Restricted Security Context
37. Chrome calls CreatRestrctionToken to create a token that has a subset of the user’s privileges
38. Assigns the token the user and group S-1-0-0 Nobody. Removes access to nearly every system resource
39. As long as the disk root directories have non-null security, not files (even with null ACLs) can be accessed
40. No network access (on Vista and later)
41. Windows Job object
42. Renderer runs as a “job” object rather than an interactive process
43. Eliminate access to:
44. Desktop and display settings
45. Clipboard
46. Creating subprocesses
47. Access to global atoms table
48. Alternate Windows Desktop
49. Windows on the same desktop are effectively in the same security context because the sending and receiving of window messages is not subject to any security checks
50. Sending messages across desktops is not allowed
51. Chrome creates an additional desktop for target processes
52. Isolates the sandboxed processes from snooping in the user’s interactions
53. Windows Integrity Levels
54. Windows Vista introduced concept of integrity levels to ease development

* Untrusted, low, medium, high, system

1. Most processes run at medium level
2. Low-integrity level has limited scope, e.g., can red but cannot write files
3. Open Design
4. “The security of a mechanism should not depend on the secrecy of its design or implementation”
5. If the details of the mechanism leaks (through reverse engineering, dumpster diving or social engineering), then it is a catastrophic failure for all users at once
6. If the secrets are abstracted from the mechanism, e.g., inside a key, then leakage of a key only affects one user
7. Kerckhoff’s principle
8. “a crypto system should be secure even if everything about the system, except the key, is public knowledge” – Auguste Kerckhoff