Windows Security

CSE497b - Spring 2007

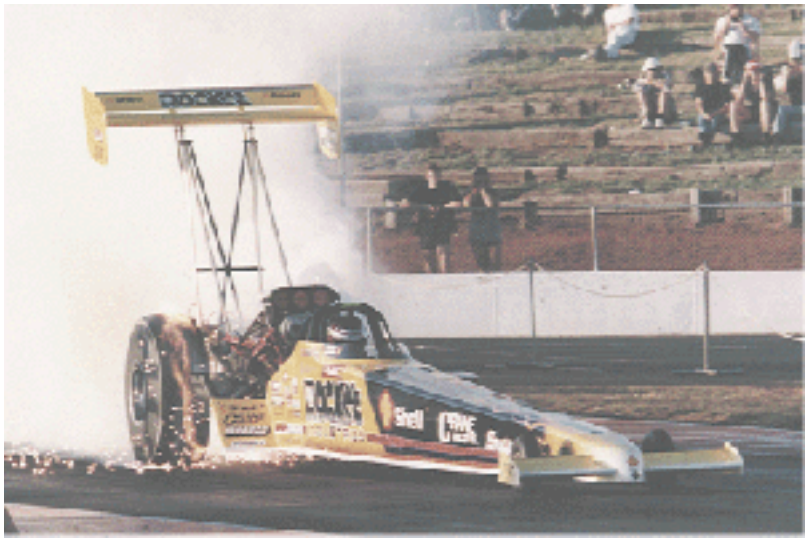
Introduction Computer and Network Security

Professor Jaeger

www.cse.psu.edu/~tjaeger/cse497b-s07/

Windows Security

- 0 to full speed
  - No protection system in early versions
- Advantage
  - Know the limits of the UNIX security model
    - What are these?
- Disadvantage
  - Legacy approaches from insecure environment
    - Will they conflict with new protection system?

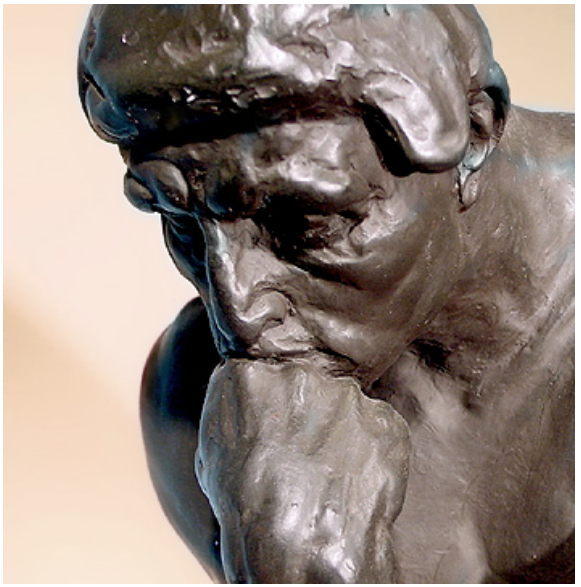


Windows Protection System

- What we will discuss was designed for Windows 2000
- Protection State
  - Fine-grained access control model
    - Flexible, but complex
  - Flexible definition of subjects and objects
  - Extensible set of operations
- Enforcement Mechanism
  - Reference Monitor
  - Does it meet guarantees?
- Transitions
  - Discretionary Access Control

Subjects

- How would you define subjects?
- UNIX has users and groups
  - Keep these?
- Should users have multiple subjects that they can use?
  - Per program
  - Per ...?
- How broadly should subjects be recognized?
  - UNIX subjects applied to one machine
  - Should subjects be global?



Windows Subjects (Access Tokens)

- User SID (subject identifier)
  - Authenticated SID
- Group and Alias SIDs
  - Groups and Aliases that apply to this user
- Privileges
  - Ad hoc rights
    - E.g., Take ownership of files
    - Like POSIX capabilities in UNIX
- Defaults for New Objects
  - Access rights for new objects created (like umask)
- Miscellaneous
  - login session ID
  - token ID

Windows Services -- Domains

- An organization of machines
  - For single sign-on and centralized security administration
- Domain is a collection of machines sharing
  - common user accounts
  - security policy
- Designate one or more *domain controllers*
  - A trusted third party
  - Stores users and groups in a domain, including passwords
  - Centralized authentication

User Authentication

- Secure attention sequence
  - CRTL-ALT-DEL
- Trusted path to login process
  - winlogon
- User name and password
  - Passed to the Local Security Authority
    - local: LSA calls SAM which authenticates and returns user SID and group SIDs
    - domain: uses Kerberos where LSA on a DC does authentication
  - LSA obtains user SIDs, group SIDs and privileges of subject
- Start a shell for user
  - new *logon session* with
  - subject access tokens are attached to process

Tokens

- Like the UID/GID in a UNIX process
- Subsequent processes inherit access tokens
  - Different processes may have different rights
- To obtain access to remote services
  - Processes create
    - network logon sessions (Kerberos tickets)
  - No remote caching
- What about setuid equivalent?
  - Services

Windows Objects

- Many types
  - Executive (processes and threads)
  - Filesystem (files and directories)
  - Others (Registry keys and devices)
- Securable objects have a *security descriptor*
  - Owner SID
    - READ_CONTROL: read access to security descriptor
    - WRITE_DAC: write access to DACL
  - Primary group
    - Compliance
  - Discretionary ACL
    - Permissions
  - System ACL
    - Audit policy

Windows Objects -- Active Directory

- Tree of typed objects
  - Extensible set of object types
- Object Types
  - A set of “properties” (attributes)
  - A globally unique ID for each type
  - Even properties have GUIDs
- “Directories” are containers of objects
  - May contain objects of different types
- Access expressed on containers or objects
  - Objects inherit access rights of containers
  - Amazingly complex combinations!

Windows Permissions

- Permissions
  - To display permissions for a file
    - Select file, properties, security
- Standard access rights
  - Apply to most objects
  - Delete, write owner, synchronize, read control, and write dac
- Otherwise, specific access rights for each type (2000)
  - Some generic rights to build on (e.g., read, write, all)
- Access rights are stored in an access mask form
  - 32-bit consisting of
    - type-specific rights
    - standard rights (above)
    - generic rights (read, write, etc)

Access Checking

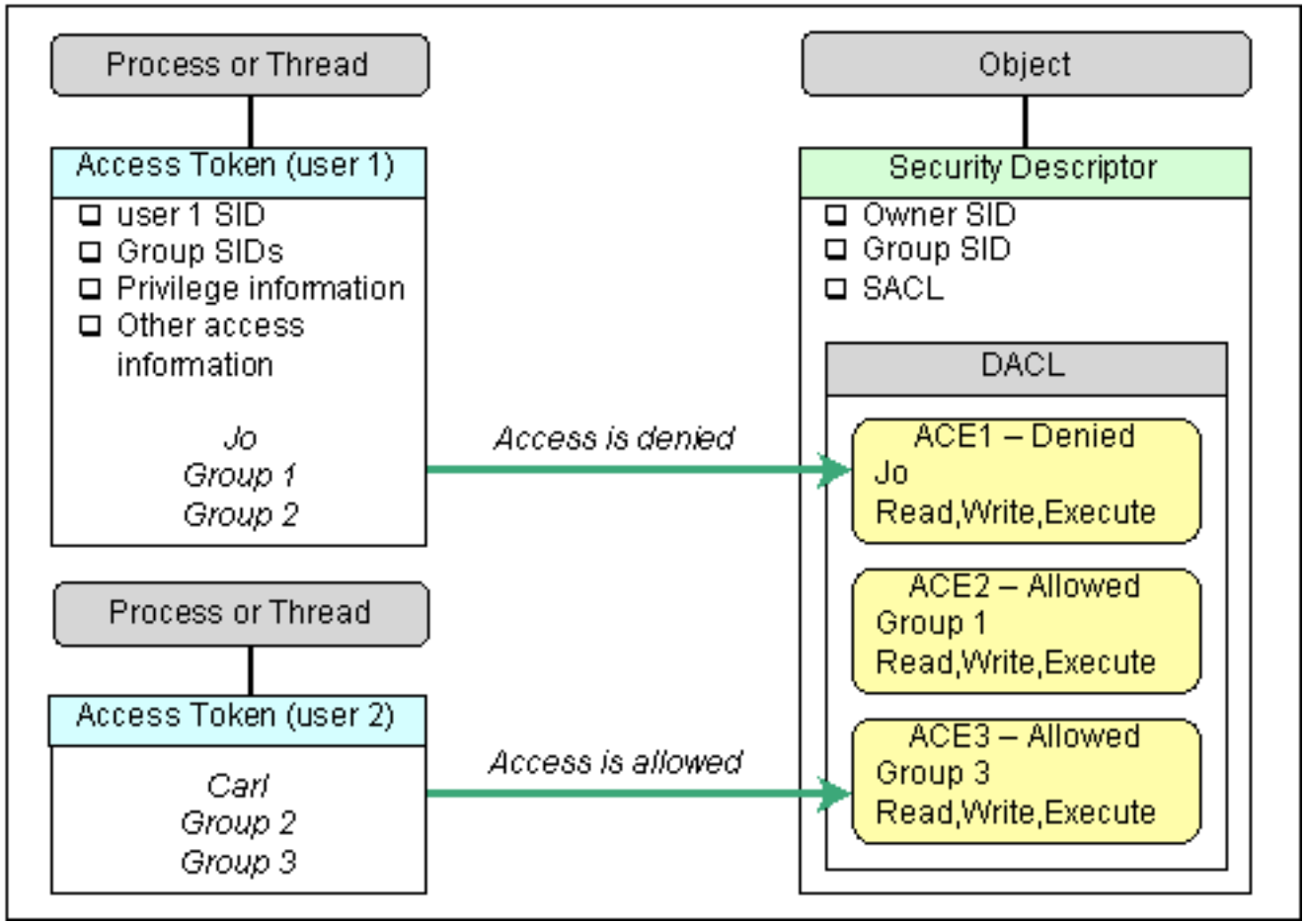
- Object types have *object managers*
  - create, store, control access
- Authorization
  - Object manager calls Security Reference Monitor
  - SRM returns policy decision
  - Object manager enforces decision
- Normally
  - Subject, object, operation, but not all are always required
- Reference monitor
  - Where is complete mediation defined?
  - Where is reference monitor implemented?
  - Which code do we depend upon for access control?

Access Control Entries

- DACL in the security descriptor of an object
  - List of *access control entries* (ACEs)
- ACE structure (proposed by Swift et al)
  - Type (grant or deny)
  - Flags
  - Object Type: global UID for type (limit ACEs checked)
  - InheritedObjectType: complex inheritance
  - Access rights: access mask
  - Principal SID: principal the ACE applies to
- Checking algorithm
  - ACE matches SID (user, group, alias, etc)
  - ACE denies access for specified right -- deny
  - ACE grants access for some rights -- need full coverage

Access Checking with ACEs

- Example



Windows vs. UNIX

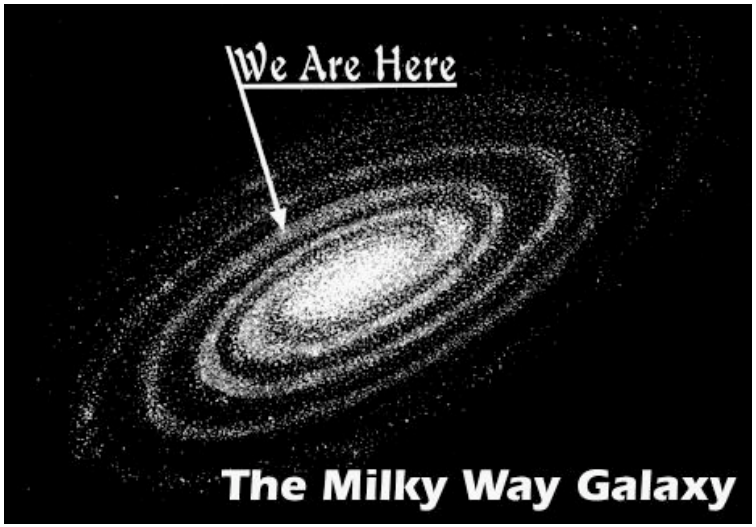
- Let's write some policies
- Is the additional expressive power of Windows worth it?
  - Who is supposed to use it?

Other Features

- Inheritance
  - InheritedObjectType of ACE
    - Only ACEs with a matching InheritedObjectType are copied
  - Inheritance Flags
    - E.g., ACE is only for inheritance
- Restricted Context
  - Implement a form of least privilege
  - Restricted tokens are used to remove privilege from process's access token
  - Access only allowed if the two access tokens grant access
- Prevent the "Confused Deputy Problem"
- "Runas"

Windows In Total

- Subject: User SID and group/alias SID
  - Multiple groups active
  - Attributes: can turn some off
- Files: ACL
  - Access Control Entry: SID, operations
  - Negative Access Tokens
  - First matching ACE is selected, may deny or grant
- Privileged users
  - Administrator
  - Anyone with Administrator group
  - Admin privileges on domain controller
    - Access throughout the domain



Windows Vulnerabilities

- Things that existed/evolved independently from security
  - E.g., Registry
- Some odd search semantics
  - Where should we find libraries and executables
- System Flexibility
  - Every application can execute remote code
- Administration model
  - Everything runs as user
  - User needs to install programs
  - Programs must run
  - User has admin privileges often

Windows Services -- Registry

- Maps “Keys” to Values (not a crypto key)
- Example Keys:
  - File extension associations: extension to application
  - Current user info: user to configuration info
  - Local machine: local machine configuration
- Access to keys
  - Determines who can edit
  - Specified in terms of keys: Below for remote registry access
  - HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurePipeServers\Winreg
  - “Not specified” means no check
- Attacking the registry is a common problem (Spyware)

Search Issues

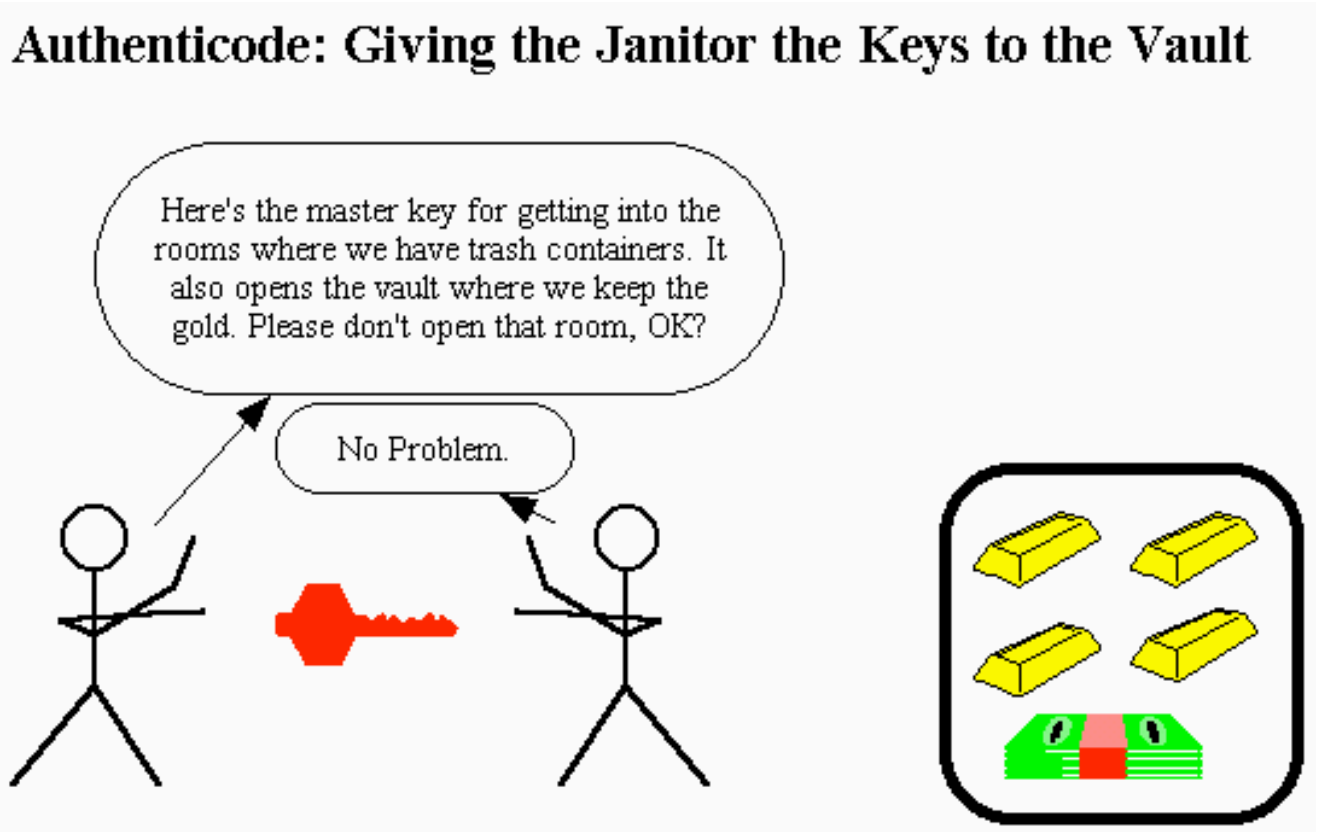
- The "." issue
  - Windows searches for executables in the working directory before using the PATH environment
  - Attackers could get files written in the user's directory tree
- PATH and other environment variables
  - Users can modify directories searched for executables
  - Attackers could place malicious logic in the user's PATH
- Shortcut spoofing
  - Assign a different destination to a shortcut
  - Attacker can get another program to run
- Standard Extensions
  - Mapping of standard extensions to executables
  - Serves all users

Windows: Library Loads

- Search in directories for DLL of specified name
  - Program Directory: directory of executable
  - System Directory: “presumably protected” directory
  - Working Directory: directory of process (where user exec’d from)
- Problem: Attacker may get file in working directory
  - User likely does not even know the working directory of a process
  - Program Directory is always first
- SafeDllSearchMode
  - Load from working before system directory if 0
  - System before working if 1
  - Default value is 1 in Windows2003 and 0 in XP

Windows Execution

- Applications that can execute programs
  - Email clients
    - All kinds
  - Web browsers
    - Scripts
  - Java virtual machine
    - Applets, servlets
  - Microsoft Word
    - Macros
- Authenticode model



Windows 2000/3 vs. UNIX

- Least Privilege
  - Which can achieve more restrictive controls?
- Fail-Safe Defaults
  - How fail-safe are each?
- Economy of Mechanism
  - Complexity of mechanisms?
- Psychological Acceptability
  - Ease of use?