

Unit OS7: Security

7.3. Windows Security Descriptors

Windows Operating System Internals - by David A. Solomon and Mark E. Russinovich with Andreas Polze

Roadmap for Section 7.3.

- Protecting Objects
- Security Descriptors and Access Control Lists
- Auditing and Impersonation
- Privileges

3

Protecting Objects

- Access to an object is gated by the Security Reference Monitor (SRM),
 - performs access validation at the time that an object is opened by a process
- Access validation is a security equation that consists of the following components:
 - Desired Access: the type of access that is being requested.
 - must be specified up front,
 - include all accesses that will be performed on the object as a result of the validation.
 - Token: identifies the user that owns the process, as well as the privileges of the user.
 - Threads can adopt a special type of token called an "impersonation token" that contains the identify of another account.
 - The object's Security Descriptor
 - contains a Discretionary Access Control List (DACL),
 - describes the types of access to the object users are allowed.

4

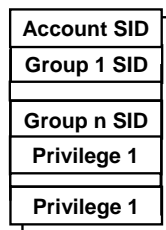
Handles and Security

- If the validation succeeds, a handle is created in the process requesting access and through which the process accesses the resource
- Changing security on an object only affects subsequent opens
 - Processes that have existing handles can continue to access objects with the accesses they were granted
 - E.g. changing permissions on a share won't affect currently connected users
- Lab: View process handles and corresponding granted accesses with Process Explorer

5

Tokens

- The main components of a token are:
 - SID of the user
 - SIDs of groups the user account belongs to
 - Privileges assigned to the user (described in next section)



6

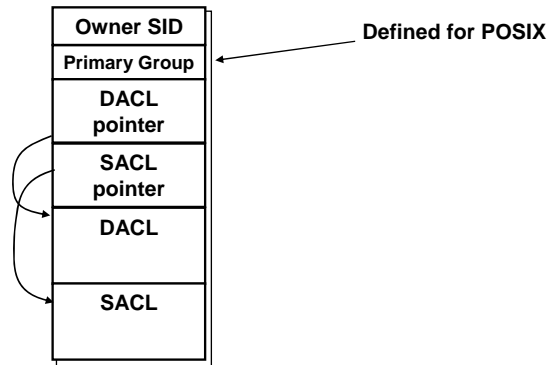
Security Identifiers - SIDs

- Windows uses Security Identifiers (SIDs) to identify security principles:
 - Users, Groups of users, Computers, Domains
- SIDs consist of:
 - A revision level e.g. 1
 - An identifier-authority value e.g. 5 (SECURITY_NT_AUTHORITY)
 - One or more subauthority values
- SIDs are generally long enough to be globally statistically unique
- Setup assigns a computer a SID
- Users and groups on the local machine are assigned SIDs that are rooted with the computer SID, with a Relative Identifier (RID) at the end
 - Some local users and groups have pre-defined SIDs (eg. World = S-1-1-0)
 - RIDs start at 1000 (built-in account RIDs are pre-defined)

7

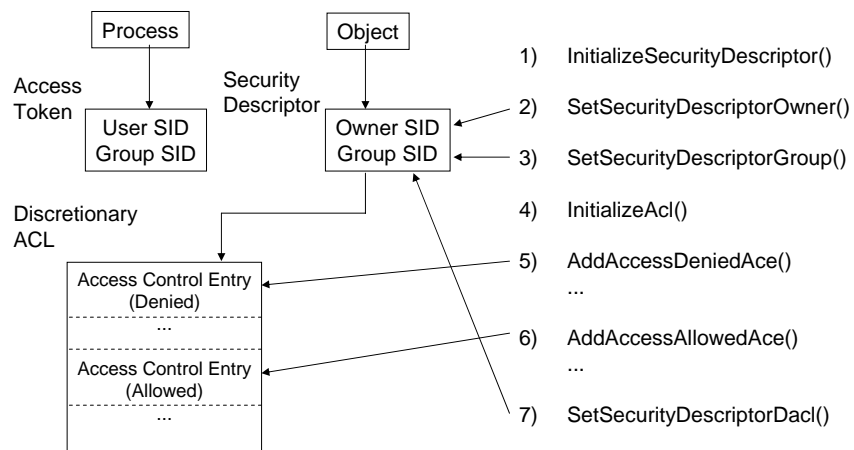
Security Descriptors

- Descriptors are associated with objects: e.g. files, Registry keys, application-defined
- Descriptors are variable length



8

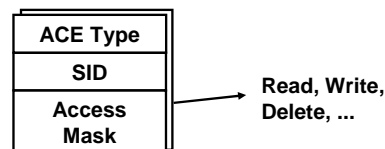
Constructing a Security Descriptor



9

Discretionary Access Control Lists DACLs

- DACLs consist of zero or more Access Control Entries
 - A security descriptor with no DACL allows all access
 - A security descriptor with an empty (0-entry) DACL denies everybody all access
- An ACE is either “allow” or “deny”



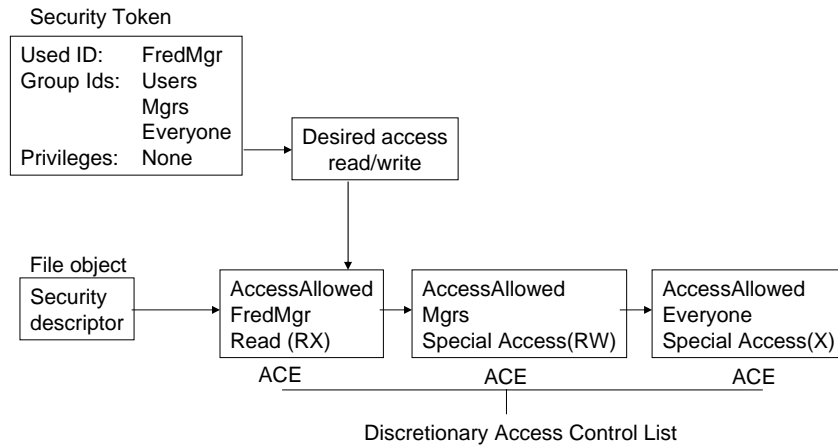
10

Access Check

- ACEs in the DACL are examined in order
 - Does the ACE have a SID matching a SID in the token?
 - If so, do any of the access bits match any remaining desired accesses?
 - If so, what type of ACE is it?
 - Deny: return ACCESS_DENIED
 - Allow: grant the specified accesses and if there are no remaining accesses to grant, return ACCESS_ALLOWED
 - If we get to the end of the DACL and there are remaining desired accesses, return ACCESS_DENIED
- The Security Reference Monitor (SRM) implements an *explicit allow* model
 - Exposed to apps through Windows API AccessCheck(), AccessCheckByType(), TrusteeAccessToObject()

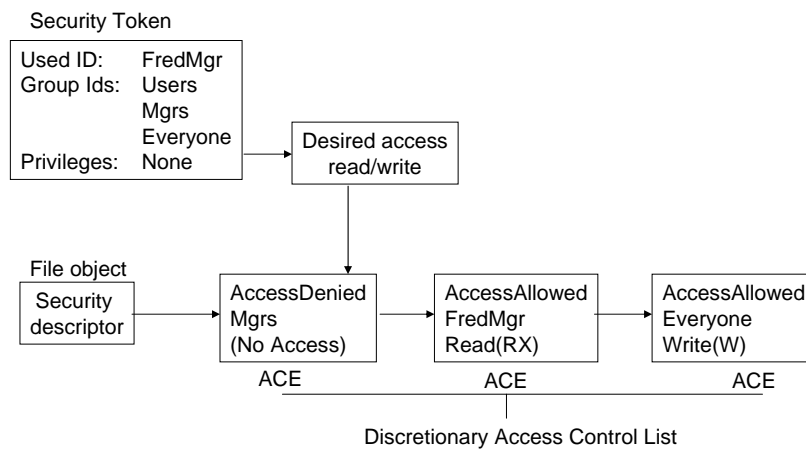
11

Example: Access granted



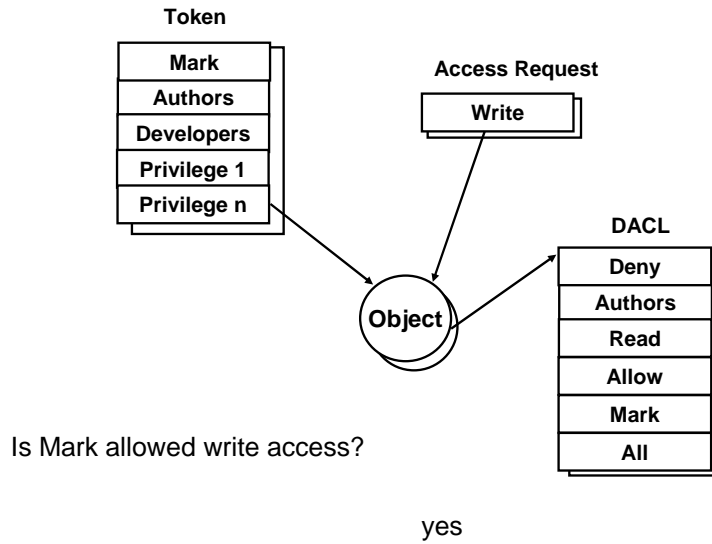
12

Example: Access denied



13

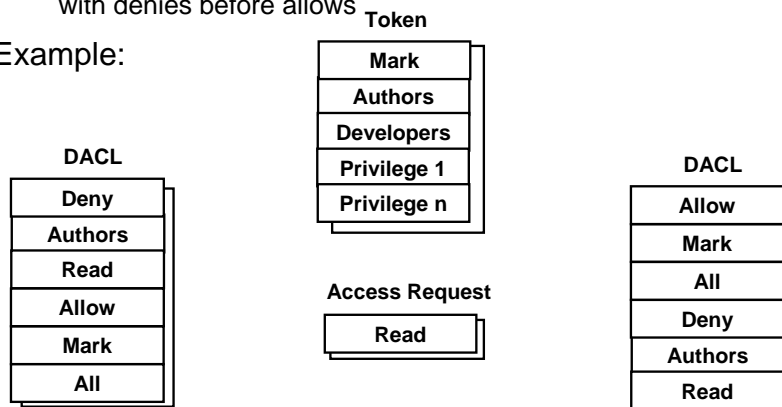
Access Check Quiz



14

ACE Ordering

- The order of ACEs is important!
 - Low-level security APIs allow the creation of DACLs with ACEs in any order
 - All security editor interfaces and higher-level APIs order ACEs with denies before allows
- Example:



15

Access Special Cases

- An object's owner can always open an object with WRITE_DACL and READ_DACL permission
- An account with "take ownership" privilege can claim ownership of any object
- An account with backup privilege can open any file for reading
- An account with restore privilege can open any file for write access

16

Object-specific ACEs

- Object-specific ACEs can be applied to Directory Services (DS) objects
 - They are just like ACES, but have two GUID fields
- The GUIDs allow the ACE to:
 - Control access to a property sheet or set on the object
 - Specify the type of child object that can inherit the ACE
 - Specify the type of child object for which the ACE grants or denies creation rights

17

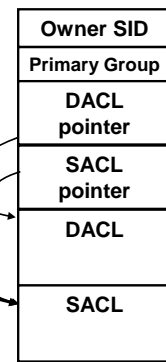
Controllable Inheritance

- In NT 4.0, objects only inherit ACEs from a parent container (e.g. Registry key or directory) when they are created
 - No distinction made between inherited and non-inherited ACEs
 - No prevention of inheritance
- In Windows 2000 and higher inheritance is controllable
 - SetNamedSecurityInfoEx and SetSecurityInfoEx
 - Will apply new inheritable ACEs to all child objects (subkeys, files)
 - Directly applied ACEs take precedence over inherited ACEs

18

Auditing

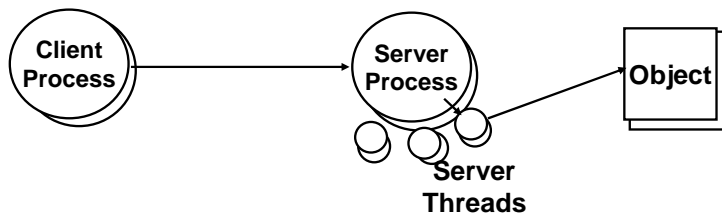
- Provides for monitoring of accesses to objects
 - Even if you specify auditing information for an object, it won't result in audit records unless Auditing is enabled
 - An administrator can enable it with the Local Security Policy Editor (secpol.msc)
 - The security log can be viewed with the Event Log Viewer
- Like for DACLs, SACL check is made on open after access check
 - Audit check is performed only if system auditing for access check result is on
 - Only ACEs that match access check result are processed
 - Test is similar to DACL test, but a record is written if there is any match
- Demo: Explorer file auditing settings



19

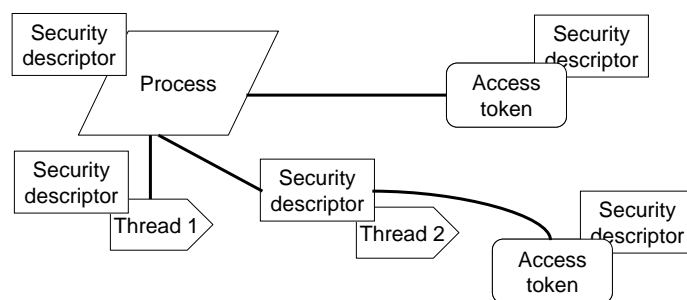
Impersonation

- Lets an application adopt the security profile another user
 - Used by server applications
 - Impersonation is implemented at the thread level
 - The process token is the “primary token” and is always accessible
 - Each thread can be impersonating a different client
- Can impersonate with a number of client/server networking APIs – named pipes, RPC, DCOM



20

Process and Thread Security Structures

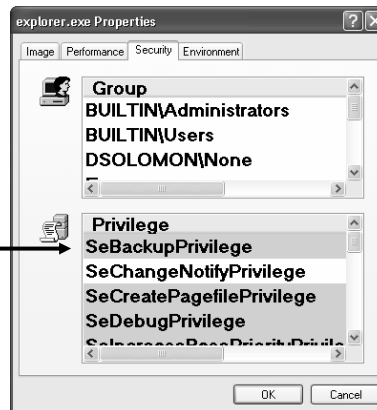


- Process/thread/access token objects have security descriptors
- Thread 2 has an impersonation token
- Thread 1 defaults to process access token

21

Privileges

- Specify which system actions a process (or thread) can perform
- Privileges are associated with groups and user accounts
 - There are sets of pre-defined privileges associated with built-in groups (e.g. System, Administrators)
- Examples include:
 - Backup/Restore
 - Shutdown
 - Debug
 - Take ownership
- Privileges are disabled by default and must be programmatically turned on with a system call



22

Powerful Privileges

- There are several privileges that gives an account that has them full control of a computer:
 - Debug: can open any process, including System processes to
 - Inject code
 - Modify code
 - Read sensitive data
 - Take Ownership: can access any object on the system
 - Replace system files
 - Change security
 - Restore: can replace any file
 - Load Driver
 - Drivers bypass all security
 - Create Token
 - Can spoof any user (locally)
 - Requires use of undocumented Windows API
 - Trusted Computer Base (Act as Part of Operating System)
 - Can create a new logon session with arbitrary SIDs in the token

23

Further Reading

- Mark E. Russinovich and David A. Solomon, Microsoft Windows Internals, 4th Edition, Microsoft Press, 2004.
 - Chapter 8, Security
 - Security Descriptors and Access Control (from pp. 506)
 - Account Rights and Privileges (from pp. 516)
- Johnson M. Hart, Win32 System Programming: A Windows® 2000 Application Developer's Guide, 2nd Edition, Addison-Wesley, 2000.
 - Chapter 5, Securing Win32 objects (from pp. 111)

24

Source Code References

- Windows Research Kernel sources
 - `\base\ntos\se` – Security Reference Monitor
 - `Accessck.c` – central access check function
 - `Token*.*`, `Seclient.c` – Token support
 - `Seaudit.c`, `Adt*.*` – security auditing
 - `Privileg.c`, `Seastate.c` – User right/privilege management
 - `Sep.h` – private structure/type definitions
 - `\base\ntos\inc\se.h` – additional structure/type definitions

25