Windows Security

Roadmap for This Lecture

- Windows Security Features
- Components of the Security System
- Protecting Objects
- Security Descriptors and Access Control Lists
- Auditing and Impersonation
- Privileges
- Windows Logon
- Kerberos Protocol Principles / Active Directory

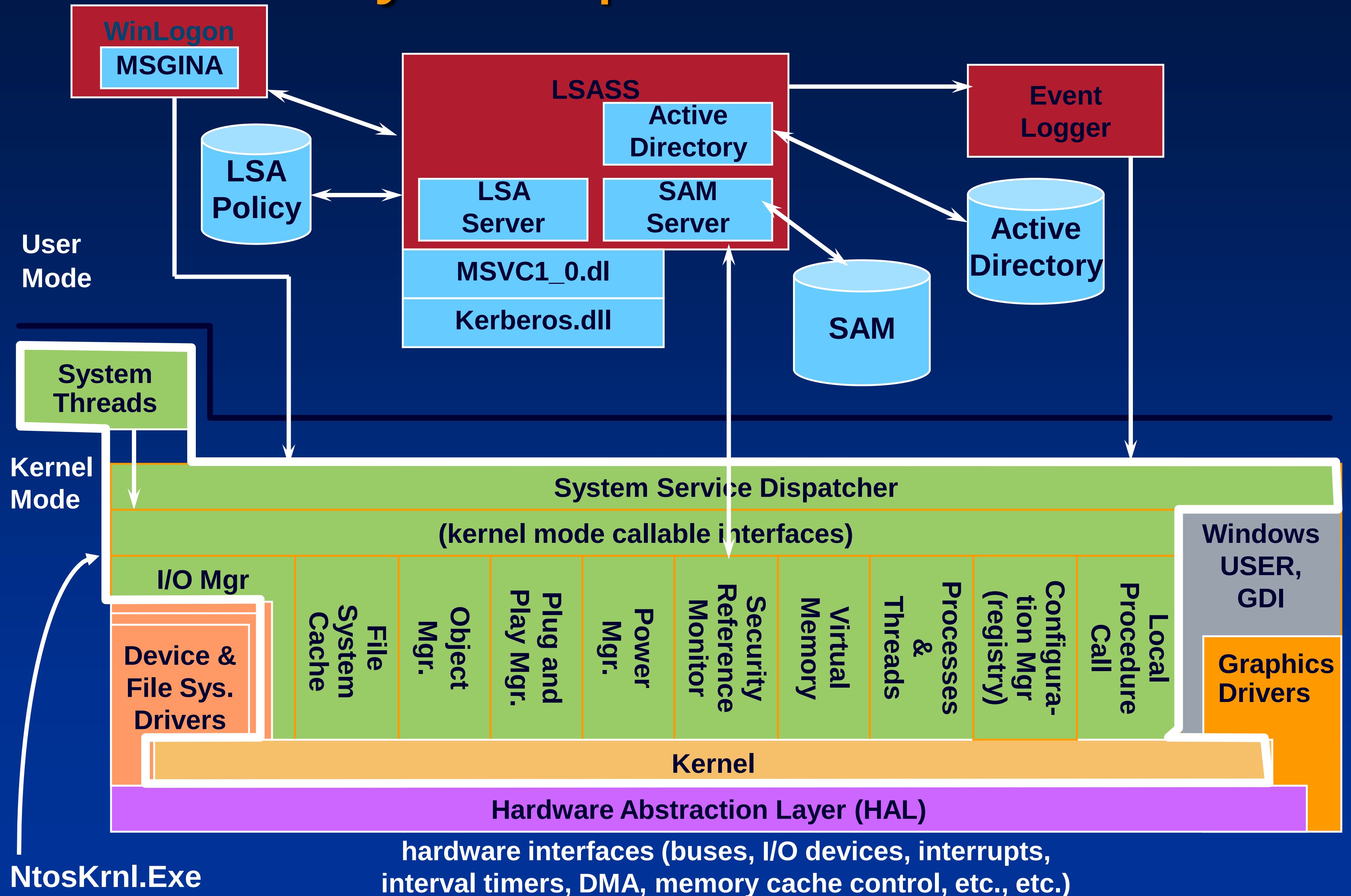
Windows Security Mechanisms

- Permissions can be applied to all shareable resources
  - Including the NTFS file system
  - …but not the FAT file system
- Encrypted File System protects data while OS is offline
  - Un-authorized physical access
- Native support for Kerberos authentication
- Public Key infrastructure to pass digital certificates
- IP Security to protect sensitive data traveling across the wire
- Crypto-APIs built into Windows
  - Hashing and encryption

The three hearts of Windows Security

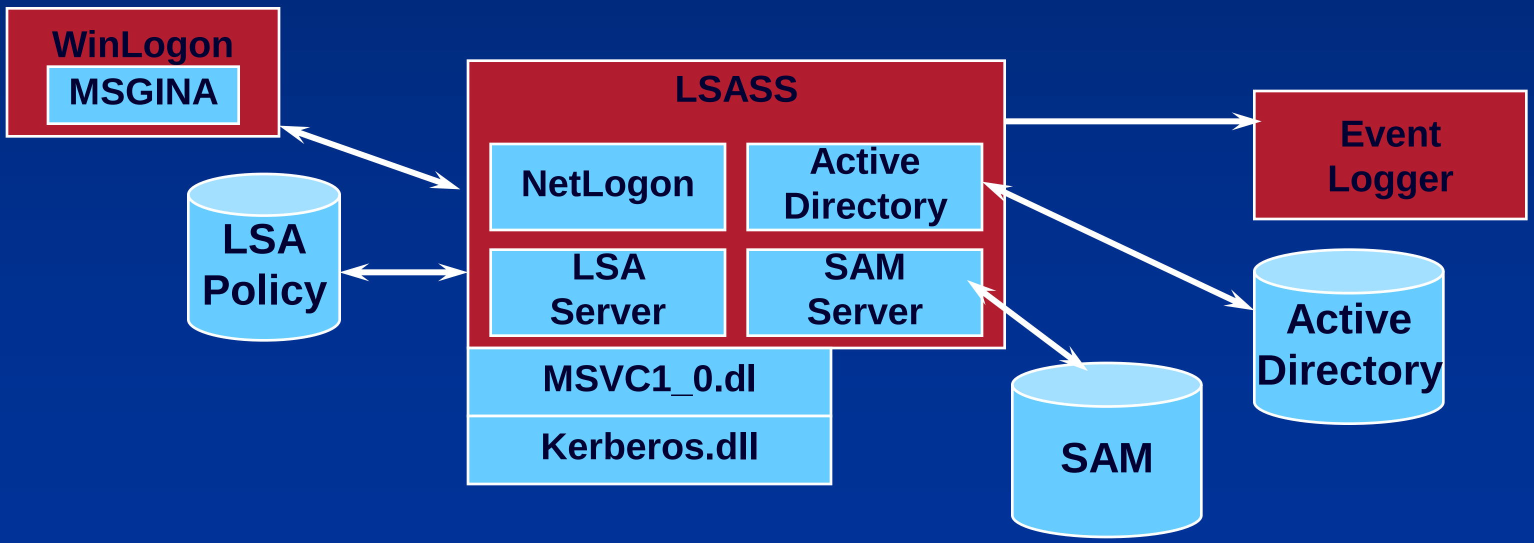
- Local Security Authority (LSA) -
  as local user-mode process
  - Heart of user authentication on local machine
- LSA - on domain controller
  - Heart of user authentication on networked machines
- Security Reference Monitor
  - Heart of object access protection

Security Components



Security Components

- Local Security Authority
  - User-mode process (\Windows\System32\Lsass.exe) that implements policies (e.g. password, logon), authentication, and sending audit records to the security event log
  - LSASS policy database: registry key HKLM\SECURITY



LSASS Components

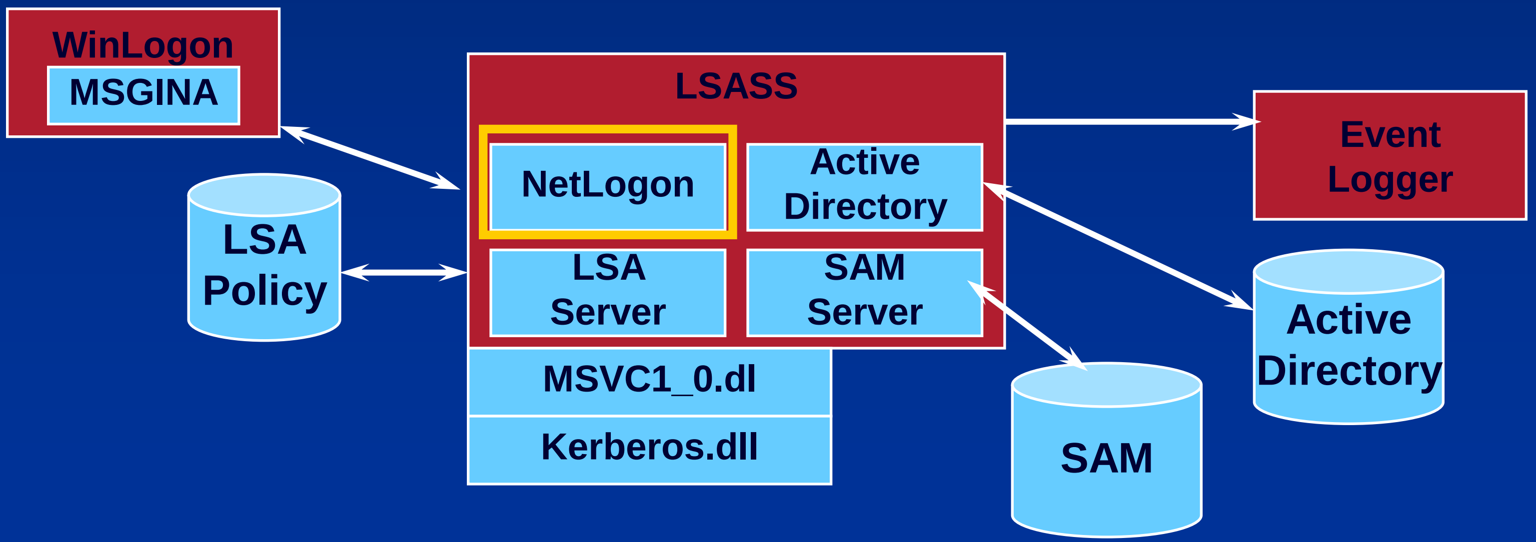
- SAM Service
  - A set of subroutines (\Windows\System32\Samsrv.dll) responsible for managing the database that contains the usernames and groups defined on the local machine
  - SAM database: A database that contains the defined local users and groups, along with their passwords and other attributes. This database is stored in the registry under HKLM\SAM.
  - Password crackers attack the local user account password hashes stored in the SAM

LSASS Components

- Active Directory
  - A directory service that contains a database that stores information about objects in a domain
  - A *domain* is a collection of computers and their associated security groups that are managed as a single entity
  - The Active Directory server, implemented as a service, \Windows\System32\Ntdsa.dll, that runs in the Lsass process
- Authentication packages
  - DLLs that run in the context of the Lsass process and that implement Windows authentication policy:
    - LanMan: \Windows\System32\Msvc1_0.dll
    - Kerberos: \Windows\System32\Kerberos.dll
    - Negotiate: uses LanMan or Kerberos, depending on which is most appropriate

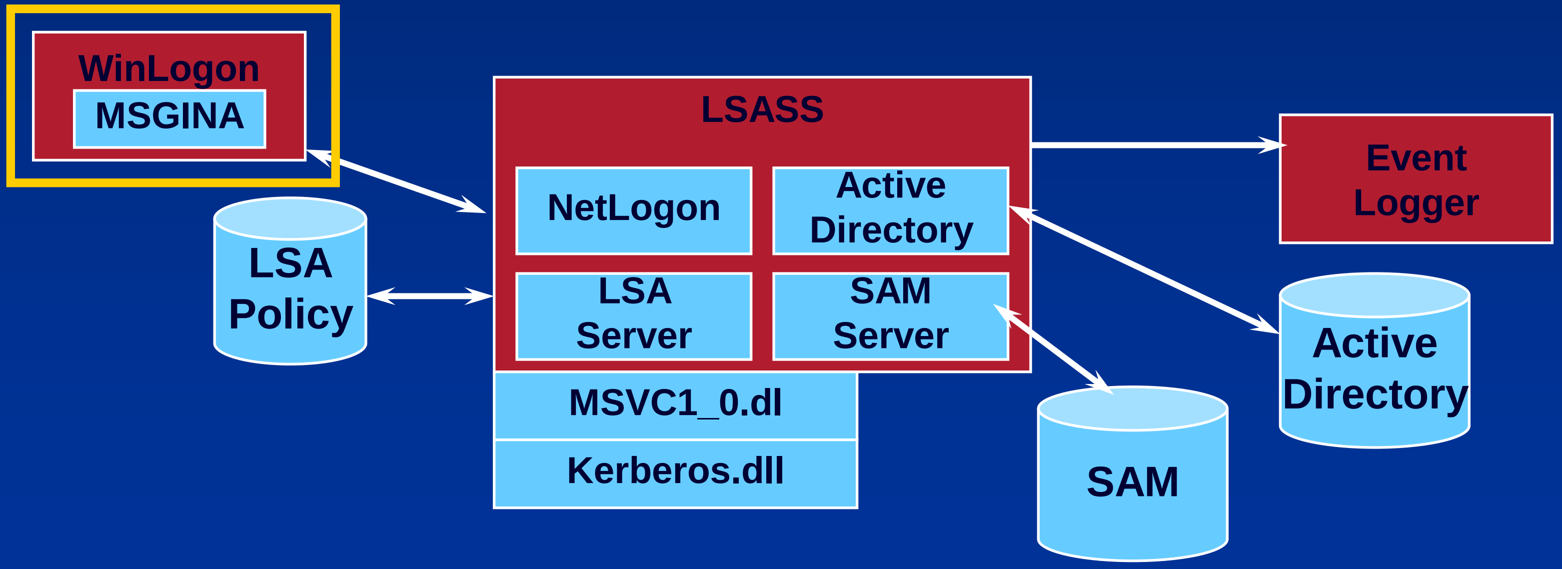
LSASS Components

- Net Logon service (Netlogon)
  - A Windows service (\Windows\System32\Netlogon.dll) that runs inside Lsass and responds to Microsoft LAN Manager 2 Windows NT (pre-Windows 2000) network logon requests
  - Authentication is handled as local logons are, by sending them to Lsass for verification
  - Netlogon also has a locator service built into it for locating domain controllers



Security Components

- Logon process (Winlogon)
  - A user-mode process running \Windows\System32\Winlogon.exe that is responsible for responding to the Secure Attention Sequence (SAS) and for managing interactive logon sessions
- Graphical Identification and Authentication (GINA)
  - A user-mode DLL that runs in the Winlogon process and that Winlogon uses to obtain a user's name and password or smart card PIN
    - Default is \Windows\System32\Msgina.dll

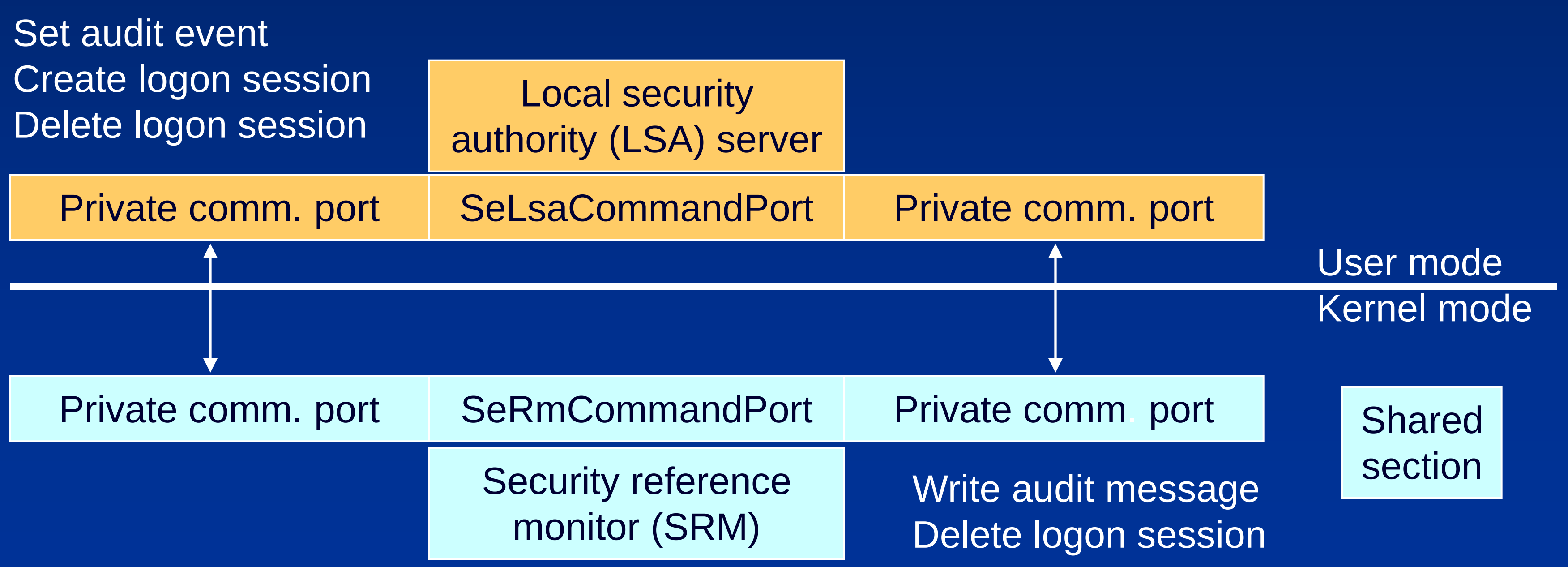


Security Reference Monitor

- Performs object access checks, manipulates privileges, and generates audit messages
- Group of functions in Ntoskrnl.exe
  - Some documented in DDK
  - Exposed to user mode by Windows API calls

Communication between SRM and LSA

- Communication via local procedure call (ALPC)
  - SeLsaCommandPort/SeRmCommand port for initialization
  - Usage of private ports/shared memory when initialization is completed



Protecting Objects

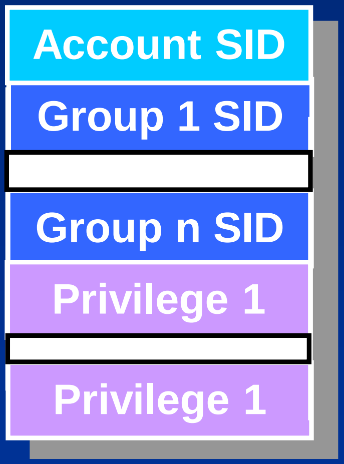
- Access to an object is gated by the Security Reference Monitor (SRM),
  - performs access validation at the time that an object is opened by a process
- Access validation is a security equation that consists of the following components:
  - Desired Access: the type of access that is being requested.
    - 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.

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

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)

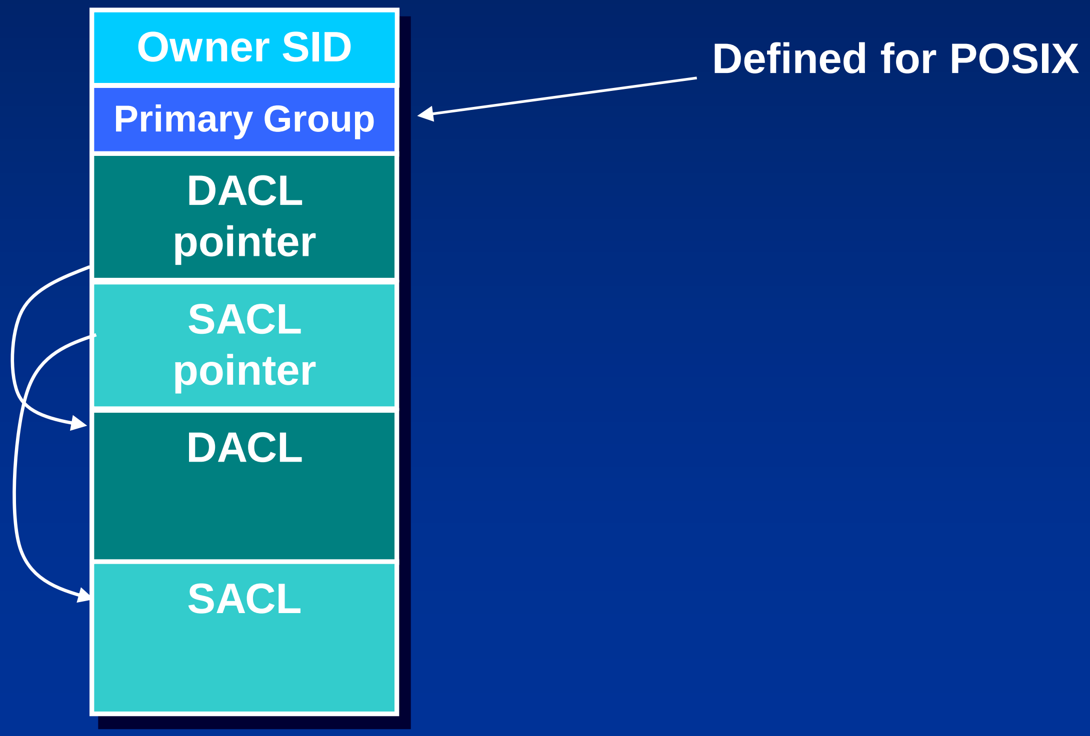


Security Identifiers - SIDs

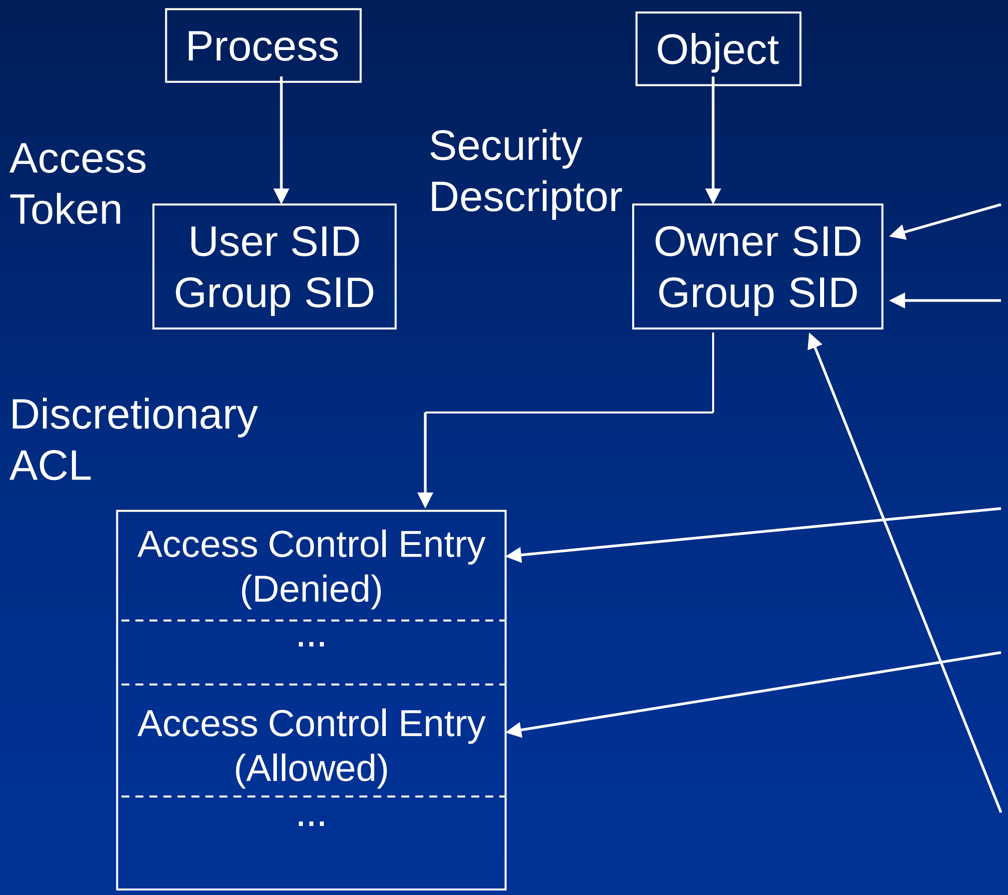
- Windows uses Security Identifers (SIDs) to identify security principles:
  - Users, Groups of users, Computers, Domains
- SIDs consist of:
  - A revision level e.g. 1
  - An identifier-authority value e.g. 5 (SECURITY_NT_AUTHORITY)
  - One or more subauthority values
- SIDs are generally long enough 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)

Security Descriptors

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



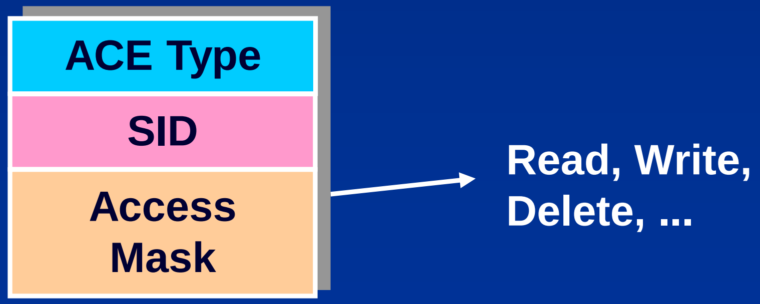
Constructing a Security Descriptor



1) InitializeSecurityDescriptor()
2) SetSecurityDescriptorOwner()
3) SetSecurityDescriptorGroup()
4) InitializeAcl()
5) AddAccessDeniedAce()
...
6) AddAccessAllowedAce()
...
7) SetSecurityDescriptorDacl()

Discretionary Access Control Lists DACLs

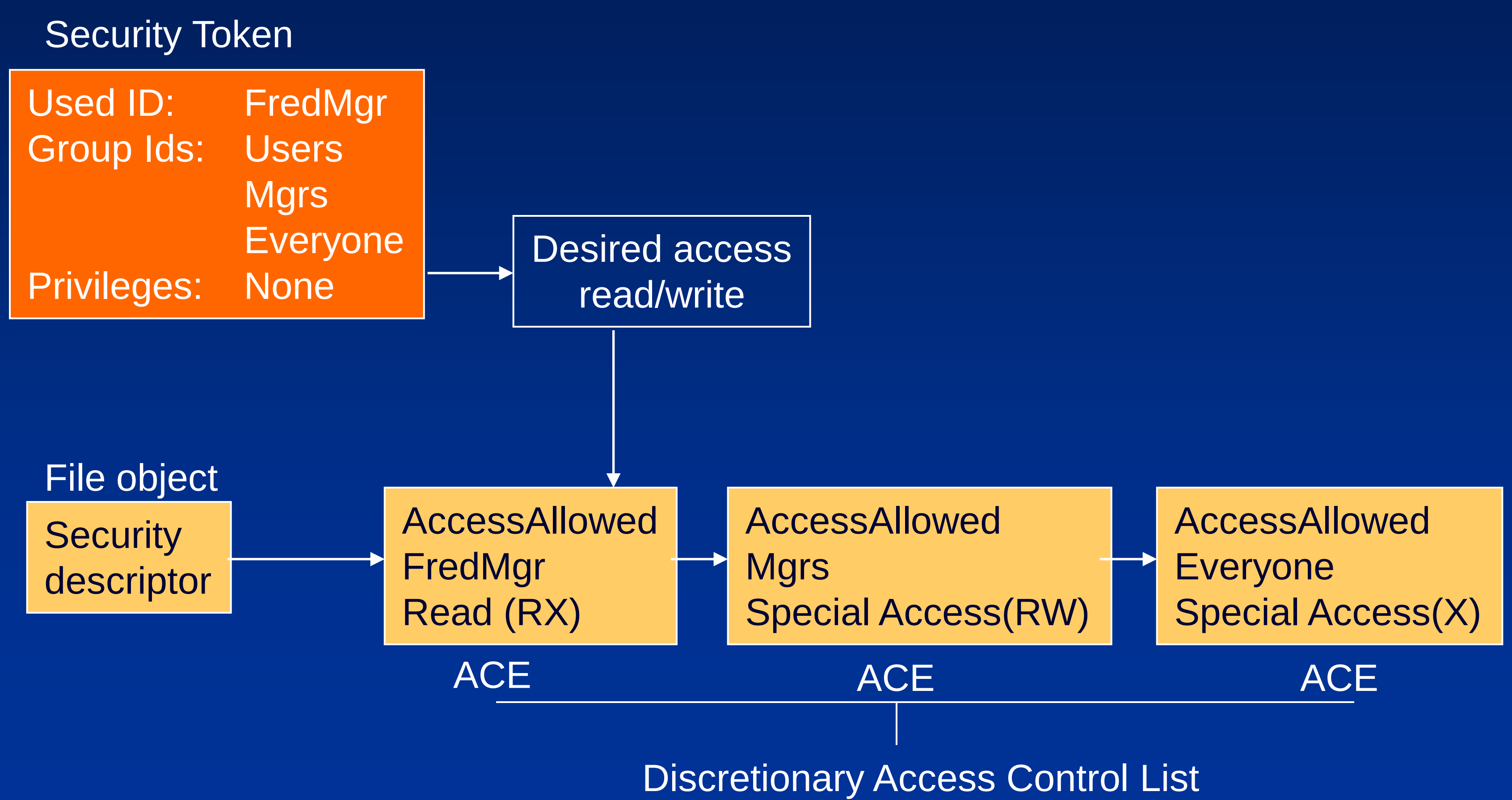
- DACLs consist of zero or more Access Control Entries
  - A security descriptor with no DACL allows all access
  - A security descriptor with an empty (0-entry) DACL denies everybody all access
- An ACE is either “allow” or “deny”



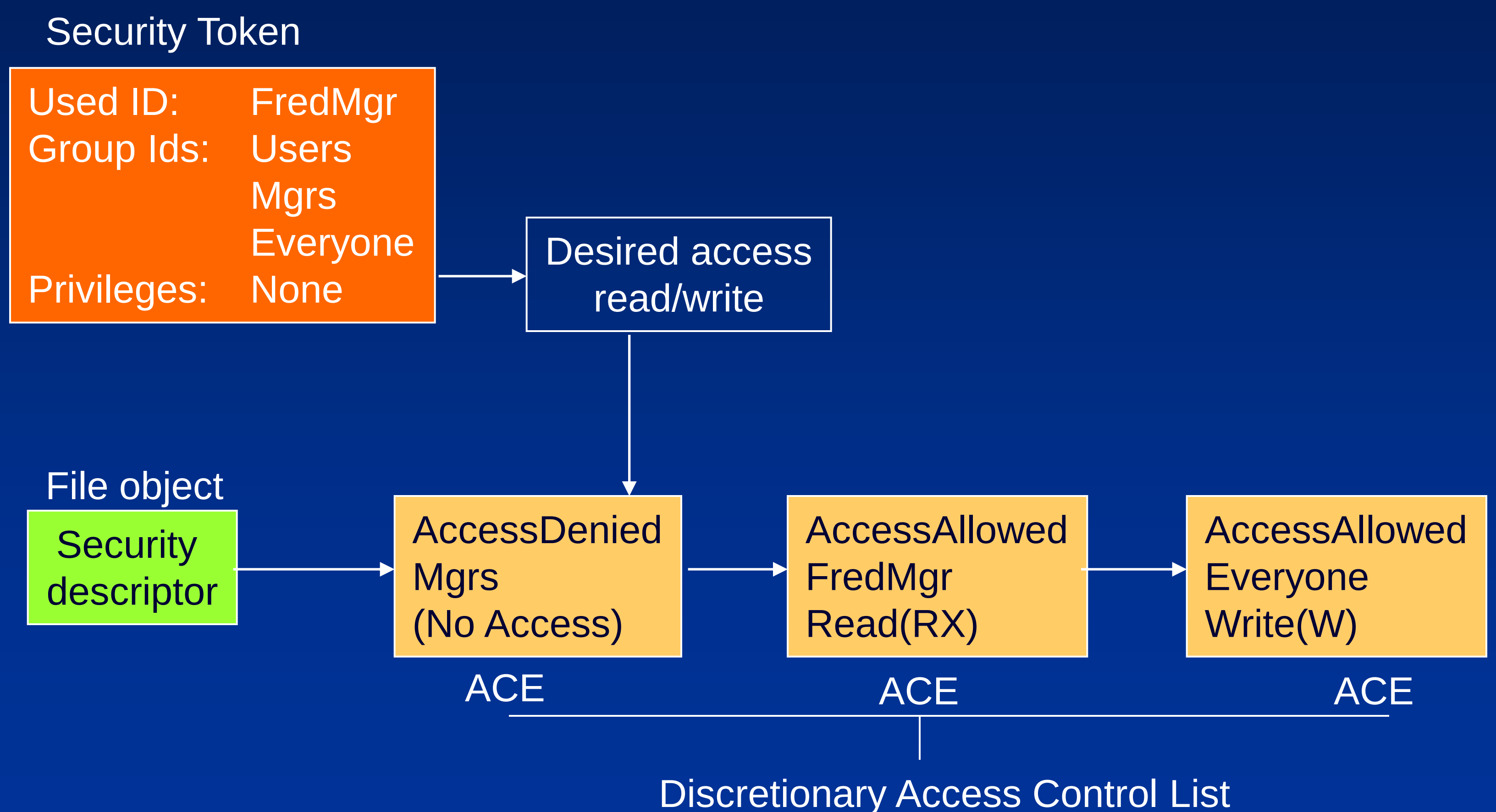
Access Check

- 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())

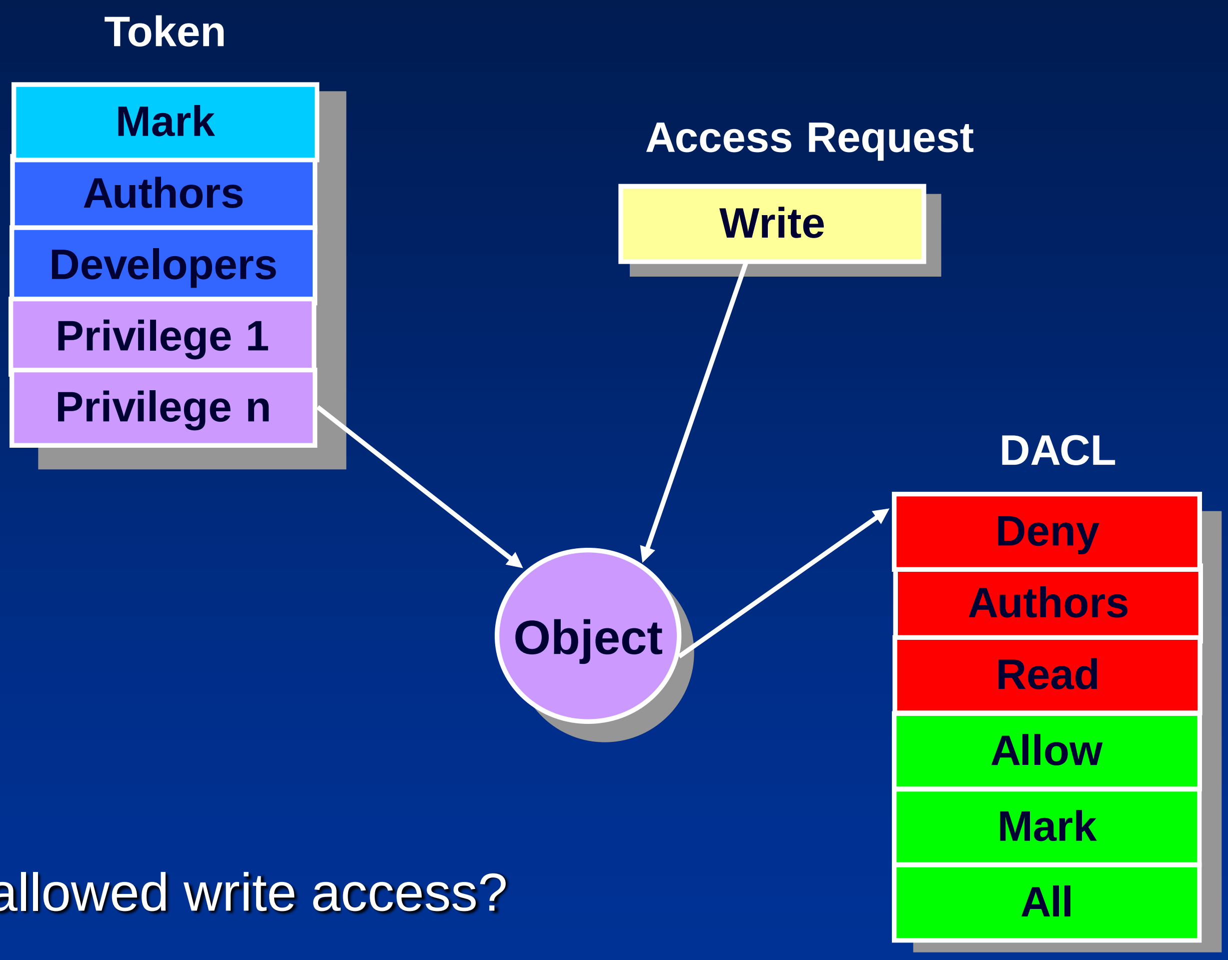
Example: Access granted



Example: Access denied



Access Check Quiz

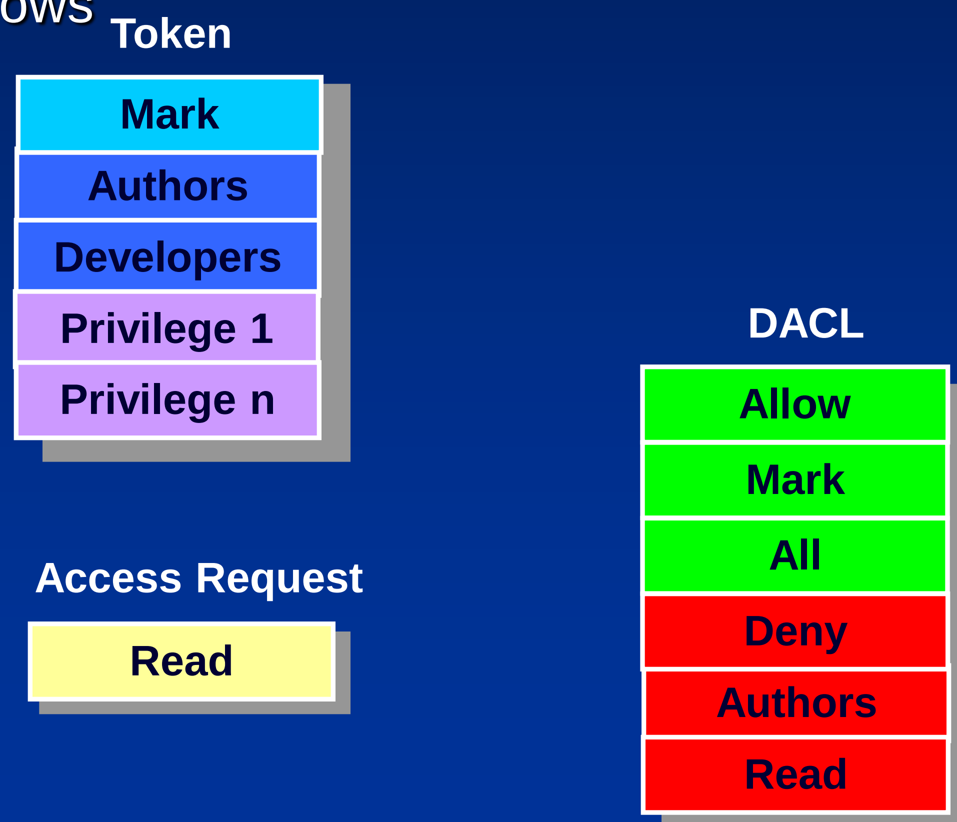


Is Mark allowed write access?

yes

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:



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

Object-specific ACEs

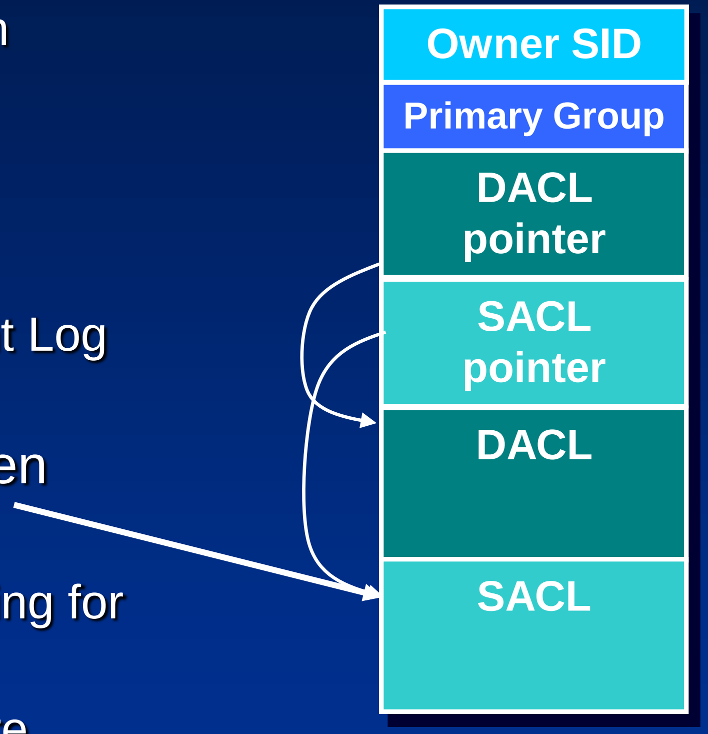
- Object-specific ACEs can be applied to Directory Services (DS) objects
  - They are just like ACES, but have two GUID fields
- The GUIDs allow the ACE to:
  - Control access to a property sheet or set on the object
  - Specify the type of child object that can inherit the ACE
  - Specify the type of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

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

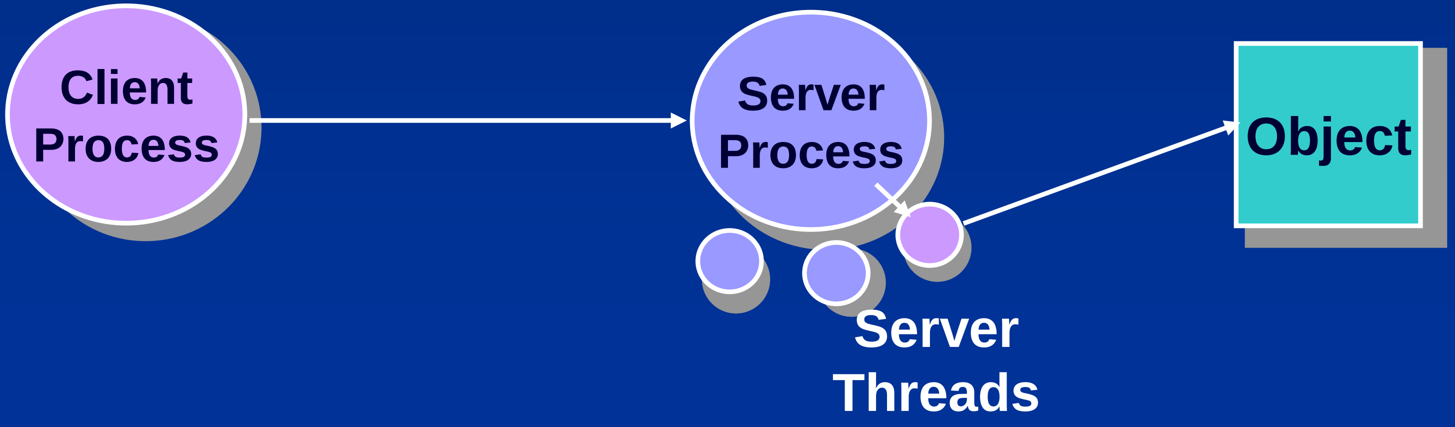
Auditing

- 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

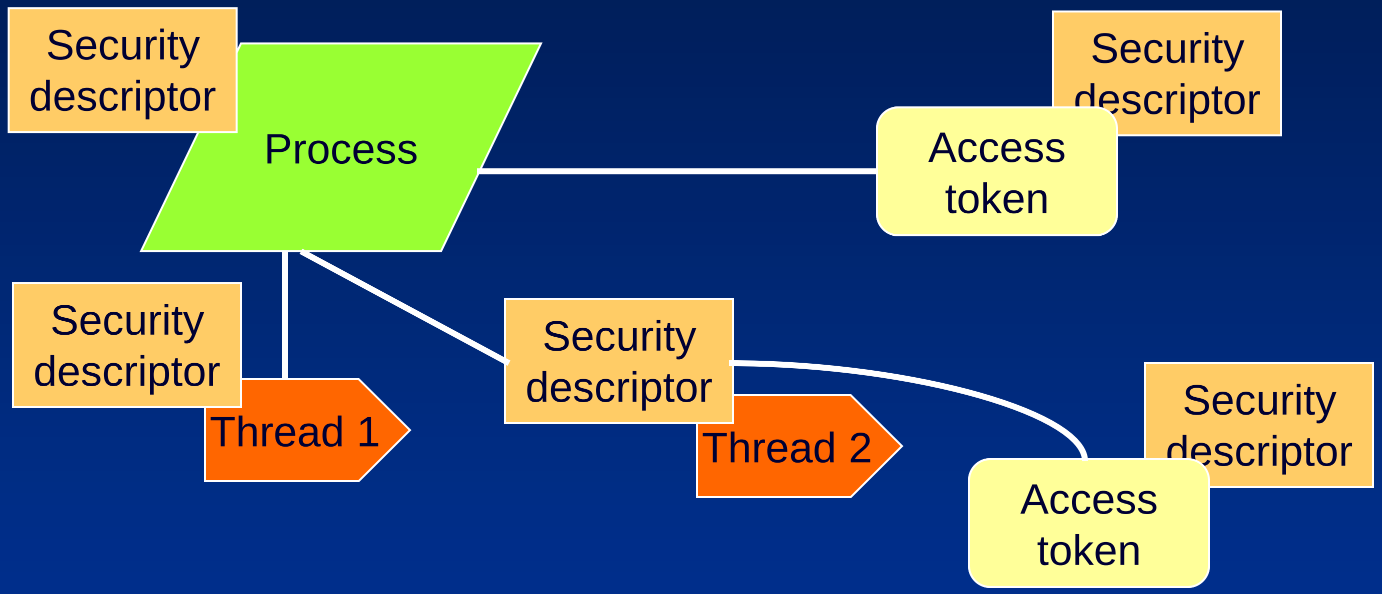


Impersonation

- Lets an application adopt the security profile of 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



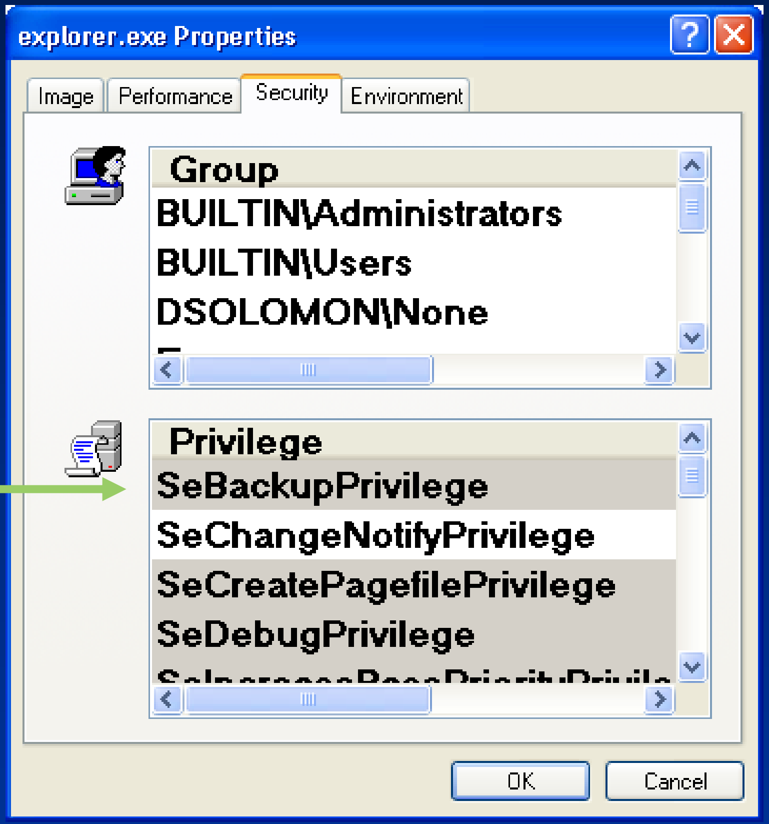
Process and Thread Security Structures



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

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



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

What Makes Logon Secure?

- Before anyone logs on, the visible desktop is Winlogon’s
- Winlogon registers CTRL+ALT+DEL, the Secure Attention Sequence (SAS), as a standard hotkey sequence
- SAS takes you to the Winlogon desktop
- No application can deregister it because only the thread that registers a hotkey can deregister it
- When Windows’ keyboard input processing code sees SAS it disables keyboard hooks so that no one can intercept it

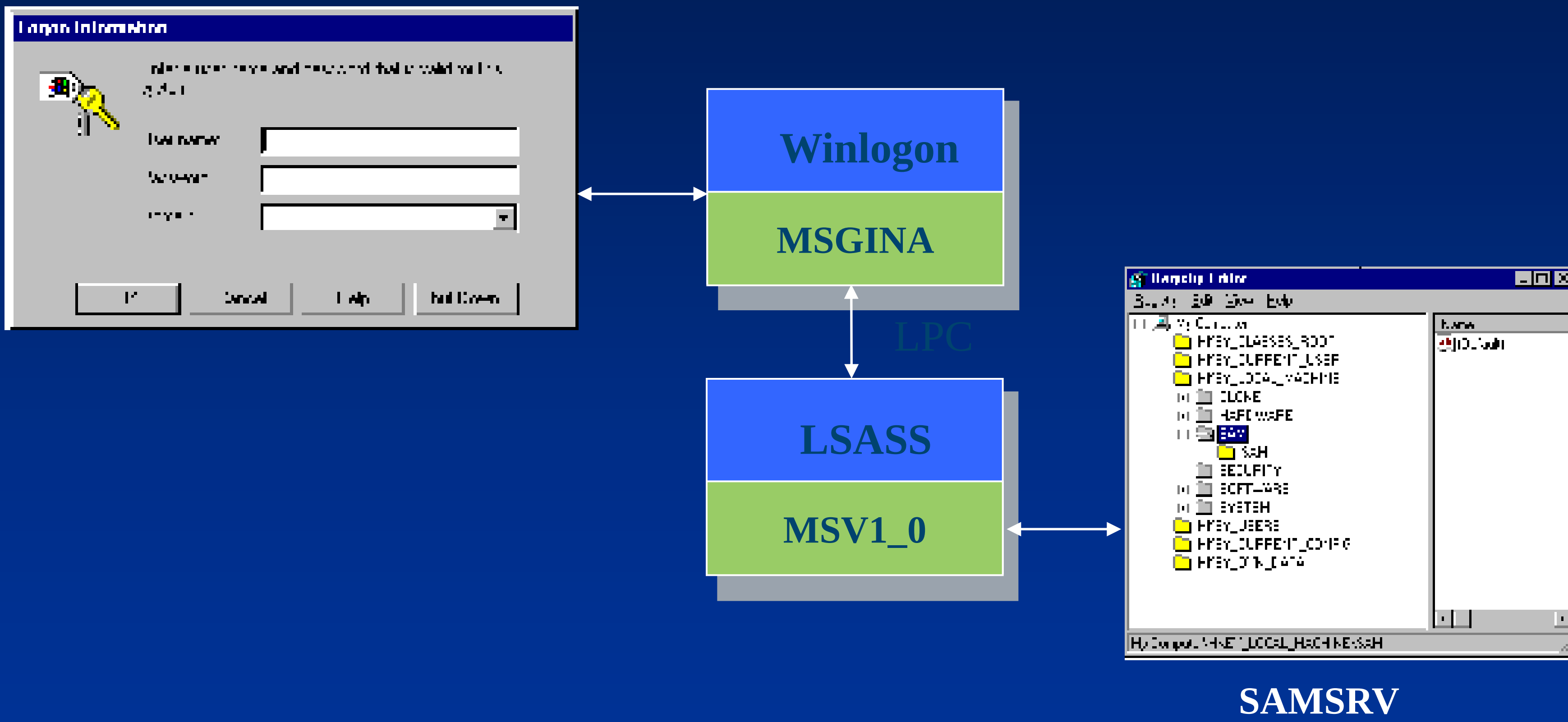
Logon

- After getting security identification (account name, password), the GINA sends it to the Local Security Authority Sub System (LSASS)
- LSASS calls an authentication package to verify the logon
  - If the logon is local or to a legacy domain, MSV1_0 is the authenticator. User name and password are encrypted and compared against the Security Accounts Manager (SAM) database
  - If the logon is to a AD domain the authenticator is Kerberos, which communicates with the AD service on a domain controller
- If there is a match, the SIDs of the corresponding user account and its groups are retrieved
- Finally, LSASS retrieves account privileges from the Security database or from AD

Logon

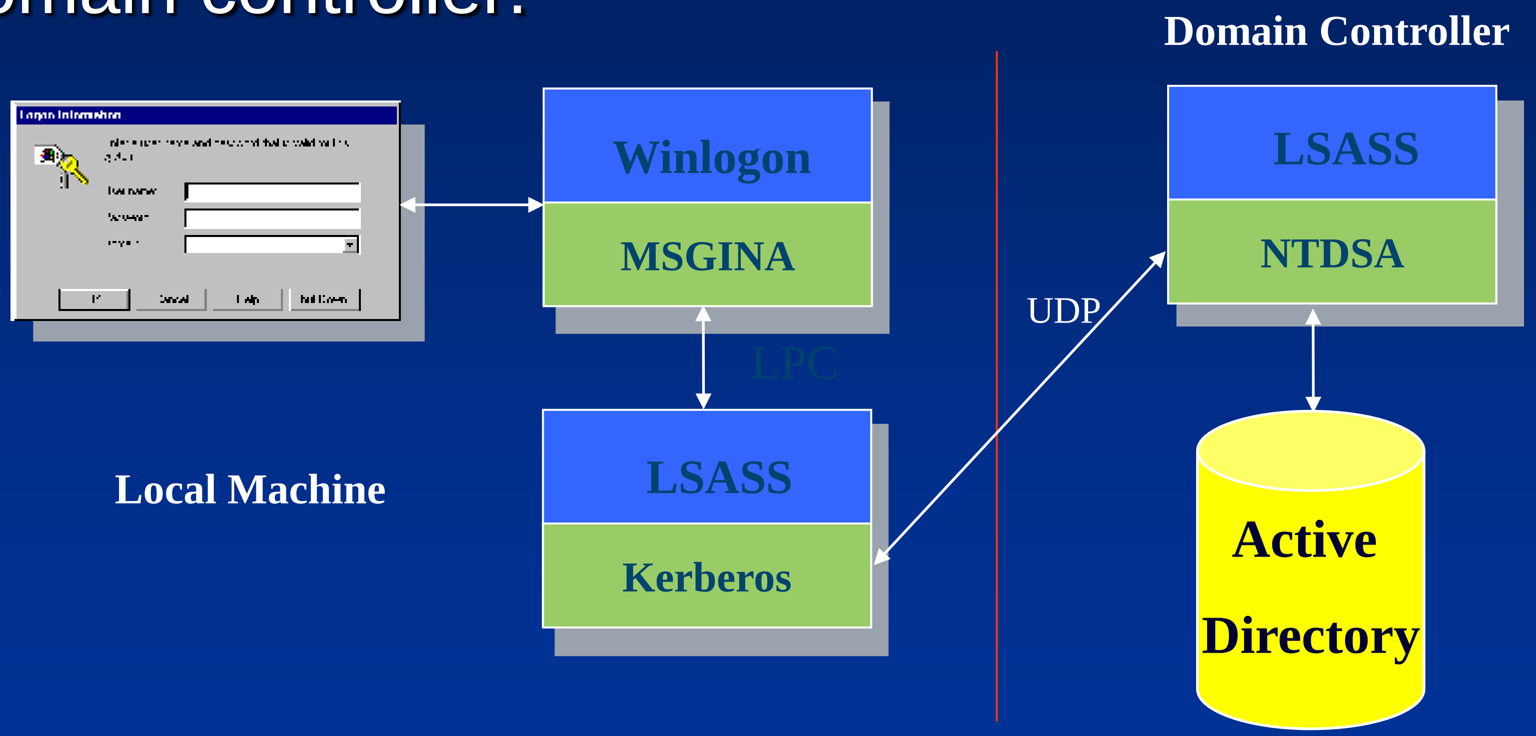
- LSASS creates a token for your logon session and Winlogon attaches it to the first process of your session
  - Tokens are created with the NtCreateToken API
  - Every process gets a <u>copy</u> of its parent's token
- SIDs and privileges can be added to a token
- A logon session is active as long as there is at least one token associated with the session

Local Logon



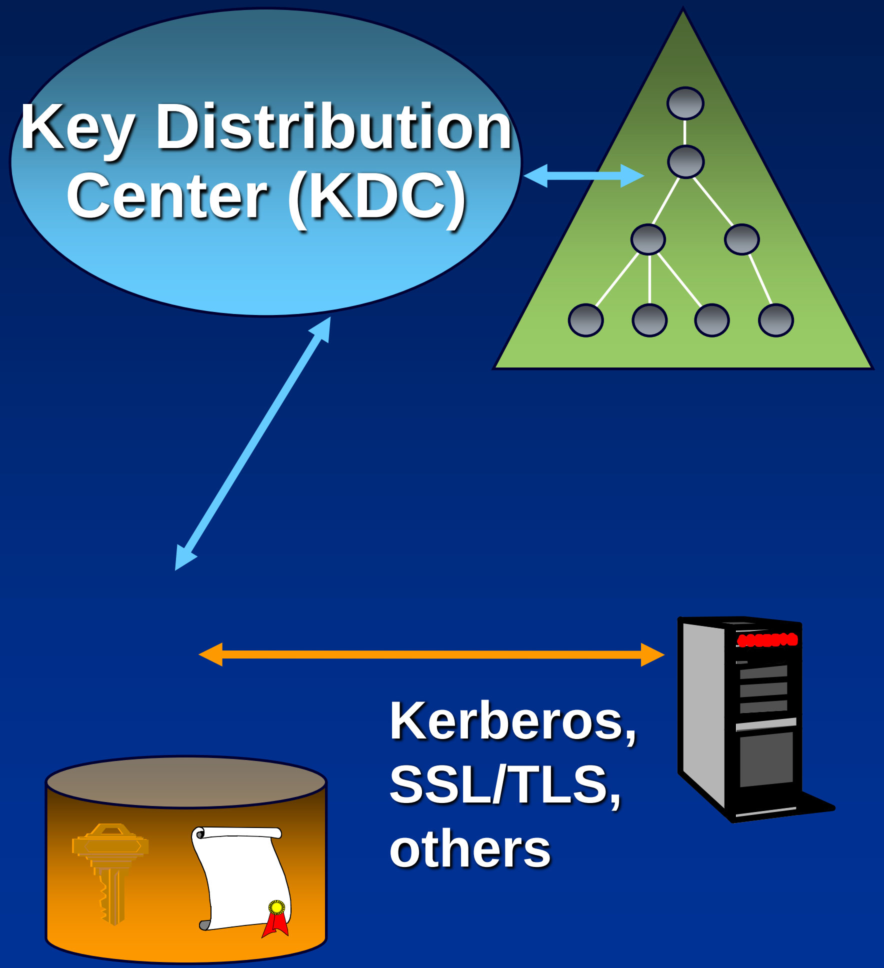
Remote Logon - Active Directory

- If the logon is for a domain account, the encrypted credentials are sent to LSASS on the domain controller:



Kerberos Authentication

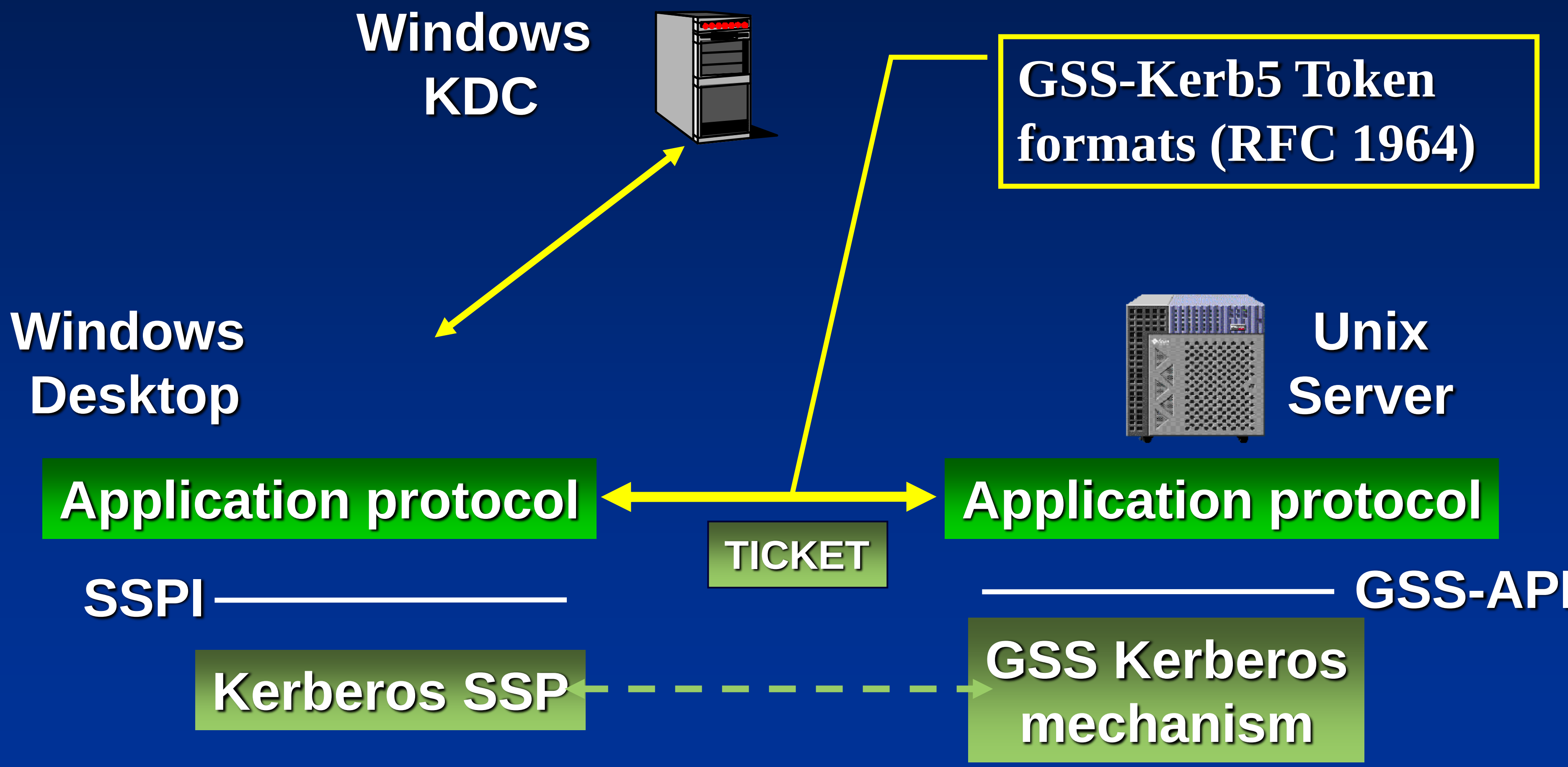
- Single account store in Active Directory
- Integrated Kerberos v5 logon
- Protected store for public key credentials
- Industry standard network security protocols



(SSL - Secure Socket Layer, TLS - Transport Layer Security)

Cross-platform Strategy

- Common Kerberos domain



(SSPI - Security Service Provider Interface, GSS - Global Security Service)

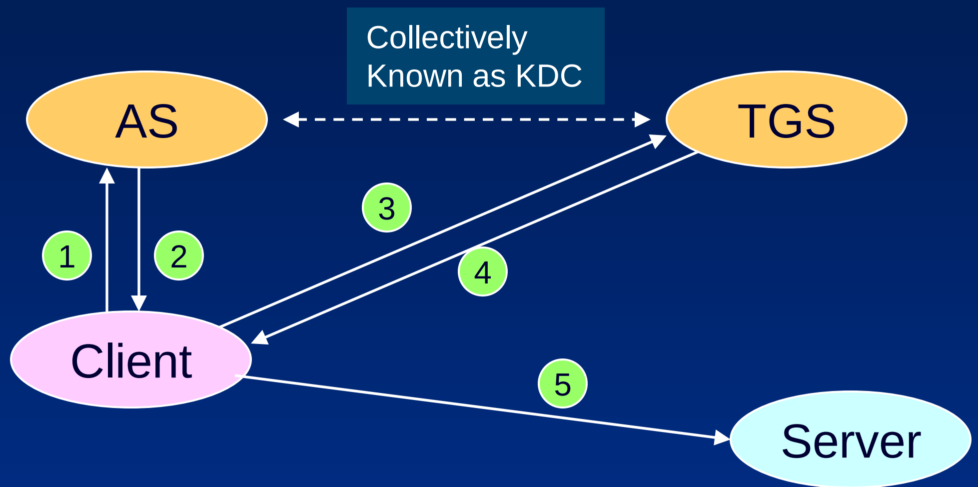
Kerberos Authentication Service

- Developed as part of MIT project Athena
- Kerberos implements an authentication procedure which verifies identity of communication partners
  - DES algorithm, symmetric key encryption
  - Authentication server (Kerberos Server)
  - TGS (Ticket Granting Service)
  - Client proves his identity by presenting an encrypted, service-specific ticket ($T_{c,s}$) when issuing a request
- Kerberos server and Ticket Granting Service (TGS) are assumed to be secure (trusted hosts)

Kerberos principles

- Kerberos requires three main steps:
  1. Client identifies himself against Kerberos Server (Active Directory), it receives a master ticket (the Ticket Granting Ticket - TGT)
  2. Client requests service-specific tickets and prove his identity with the TGT
  3. Client uses service-specific ticket to contact server
- Authentication is transparent from user‘s point of view
  - Windows login program acquires TGT
  - (Client) Applications transparently acquire service-specific tickets
  - TGS-issued tickets and TGT have a default lifetime of eight hours

Kerberos principles (contd.)



$K_{c:}$ client's secret key
$K_{c,tgs:}$ key for comm. between client and TGS
$\{T_{c,tgs}\}K_{tgs:}$ encrypted ticket for TGS
$K_{c,s:}$ key for client/service communication
$\{T_{c,s}\}K_{s:}$ encrypted ticket for service
$A_{c:}$ authentication info

1. Client -> AS: c, tgs, n
2. AS -> Client: $\{K_{c,tgs},n\}K_c$, $\{T_{c,tgs}\}K_{tgs}$
3. Client -> TGS: $\{A_c\}K_{c,tgs}$, $\{T_{c,tgs}\}K_{tgs}$, s, n
4. TGS -> Client: $\{K_{c,s}, n\}K_{c,tgs}$, $\{T_{c,s}\}K_s$
5. Client-> Server: $\{A_c\}K_{c,s}$, $\{T_{c,s}\}K_s$

Tickets and Authentication info

- Kerberos tickets contain the following data:
  - User name
  - Address of workstation
  - Time stamp
  - Lifetime of the ticket
  - Address of the host running the requested service
  - Session key for client/server communication
- Tickets are encrypted with the server‘s private key (K_s)
- Authentication info (A_c) contains the following data:
  - User name
  - Address of workstation
  - Time stamp
- Authentication info is encrypted with the session key $K_{c,s}$

Kerberos Version 5 - Windows

- Multiple supported encryption algorithms through Crypto-API foundation
- Keys carry info about encryption algorithm used
  - Can be re-used for different encryption algorithms
- Network addresses may have arbitrary formats
  - Server may specify all supported protocols/addresses in ticket
- Network data format and encryption are standardized
  - ASN.1 format (ISO 8824), no special format for multi-byte data
  - Encryption based on (ISO 8825)
- Tickets contain plaintext section
  - Server may support multiple personalities, actual role is chosen on plaintext info
- Tickets carry starting time and expiration time

Ticket Characteristics

- KDC returns special tickets on initial ticket exchange
  - Password can only be changed with those special tickets
- Renewable tickets may carry two expiration dates
  - Only valid after first but before second date
- Tickets may be postdated
  - Interesting for batch processing
- Authorization data field
  - KDC copies authorization info from TGT into every newly generated ticket
  - Windows Kerberos supports public/private key for initial authentication (to obtain TGT via user-supplied private key)

Lab Demos

- Inspecting SAM service
  - Open Lsass.exe process properties – click on services tab
  - Click Find DLL – search for Samsrv.dll
- Open Ntoskrnl.exe with Dependency Walker and view functions starting with “Se”
- Run “LogonSessions –p” (from Sysinternals) to view the active logon sessions on your system
  - Run pview.exe (in \sysint\reskit folder)
  - Select process (explorer.exe)
  - Watch process/thread/p-token/t-token security descriptors
  - Watch process/thread access token (gray button – no thread specific token existent)
- View process handles and corresponding granted accesses with Process Explorer
- Explorer file auditing settings

Further Reading

- Mark E. Russinovich and David A. Solomon,
  Microsoft Windows Internals, 5th Edition, Microsoft Press, 2009.
  - Chapter 6, Security
- Wikipedia: Kerberos (Protocol)
  http://en.wikipedia.org/wiki/Kerberos_(protocol)

- John T.Kohl, B.Clifford Neumann, Theodore Y.Ts'o, The Evolution of the Kerberos Authentication Service, Proceedings of Spring 1991 EurOpen Conference, Tromsø, Norway.

- 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)

Source Code References

- Windows Research Kernel sources
- Windows Research Kernel sources
  - \base\ntos\se – Security Reference Monitor
  - \base\ntos\inc\se.h – additional structure definitions
- Note: WRK does not include sources for security processes or network security components