

# Operating System Security

Susan Hinrichs

Cyber Security Spring 2006

# Outline

- Unix/Linux Access Control
  - Users and groups
  - File system controls
- Windows NT/XP Security Executive
  - Access tokens
  - Security descriptors
  - ACLs
- Windows Vista
  - Security additions

# Unix Reading Material

- Man pages
  - Groups, newgroup
  - Chmod, chown, chgrp
- Unix and Security: The Influences of History
  - <ftp://coast.cs.purdue.edu/pub/doc/misc/spaf-influences-of-history.ps.Z>

# Basic Unix Security Model

- User authenticated on logon
  - User ID associated with process
  - Default Group ID associated with process
  - Default Process listed in passwd file
- Groups defined in /etc/groups
  - Set of users listed with each group definition
  - User can be member of multiple groups

# Shadow Files

- /etc/passwords and /etc/group must be readable by everyone
- Both files contain crypt'ed passwords
  - Access enable offline attacks
- Add shadow versions of each file
  - Password obscured in passwords and group
  - Stored in more restricted shadow versions of these files

# Unix Access Control

- Three permission octets associated with each file and directory
  - Owner, group, and other
  - Read, write, execute
- For each file/directory
  - Can specify RWX permissions for one owner, one group, and one other

# Unix Access Check

- First test effective user ID against owner
  - If match, then use owner rights
- Then test all groups user is a member of against group
  - If match, then use group rights
- Otherwise, use other rights
- Can view as rwx, or a value from 0-7
  - E.g. rx = 5 and rw = 6

# Constraining Control of New Objects

- Umask can be set to constrain allowed access on new objects created by user
- Expressed as a 3 octet mask
  - E.g. 0022
- Inverse of umask anded by requested access for new object
  - E.g. open requests 0666 (read and write for all)
  - $0666 \& \sim 0022 = 0666 \& 755 = 644$

# Other Bits

- **Set UID and Set GUID bits**
  - When set, the process created by executing file takes on user ID or group ID associated with file
- **Sticky bit**
  - On directories, prevents anyone but owner of file removing file in directory

# File System Extensions

- Ext2 extra attributes
  - a – append only
  - c – compressed
  - s – secure deletion
  - u – undeletable
  - i - immutable

# Unix Security Problems

- Created as a subset of more complete Multics model
  - Expedient at the time
  - Limits modern expressibility
- Security evolved over 30 years
  - Inconsistencies
- Early evolution occurred in open university environments
  - Encourages bad habits

# Windows Reading Material

- Inside Windows NT, Helen Custer
  - Chapter 3 section 3
- [Windows NT Security in Theory and Practice](#)
- Vista Security Features
  - <http://www.microsoft.com/technet/windowsvista/evaluate/feat/secfeat.mspx>

# NT Security Model

- Ultimately NT security controls access and auditing
- Implements the standard subject/object security model
  - Designed into NT. Implemented a security constrained executive
- Controls applied to core OS objects like processes and sockets in addition to the more traditional file system elements (NTFS)
  - Everything that can be named is an object
  - All objects can have same security controls applied

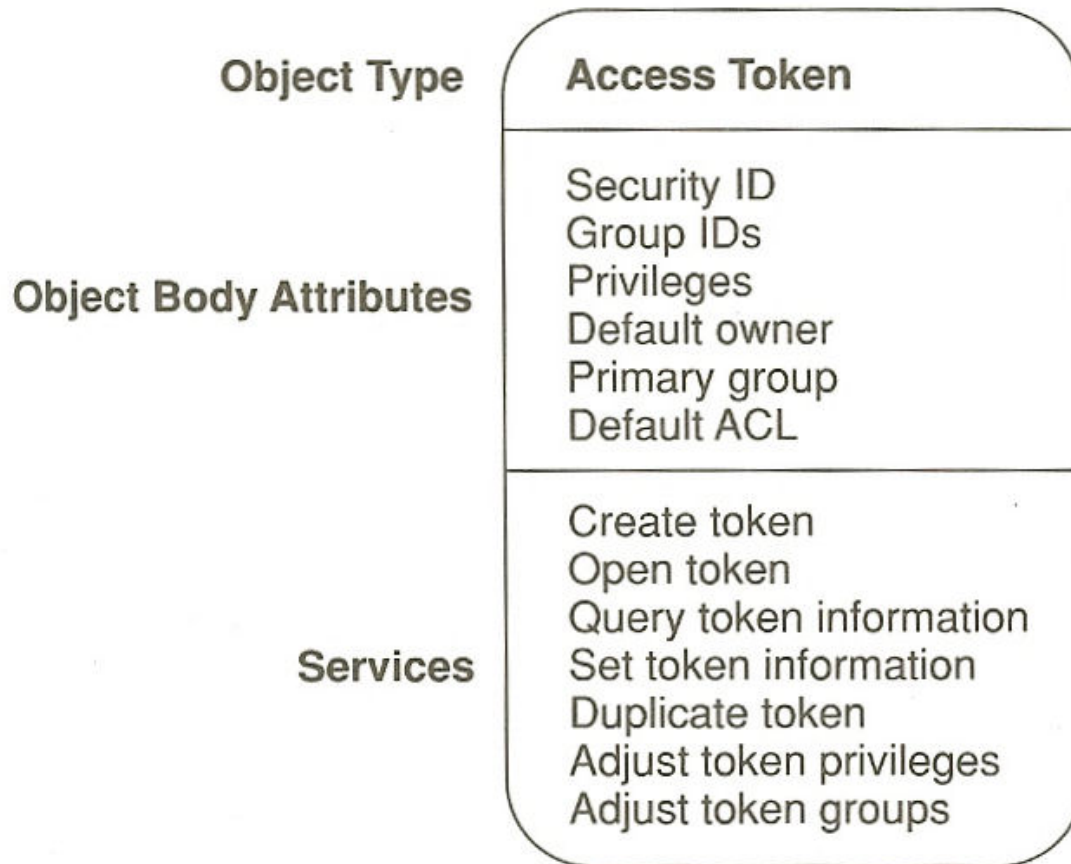
# NT Security Elements

- Subject – Process or thread running on behalf of the system or an authenticated user
- Security ID (SID) – A globally unique ID that refers to the subject (user or group)
- Access token – the runtime credentials of the subject
- Privilege – ability held by the subject to perform “system” operations. Usually breaks the standard security model
  - Associated with the access token
  - Generally disabled by default.
  - Can be enabled and disabled to run at least privilege
  - Example powerful privileges
    - **SeAssignPrimaryTokenPrivilege** – Replace process token
    - **SeBackupPrivilege** – Ignore file system restrictions to backup and restore
    - **SeIncreaseQuotaPrivilege** - Add to the memory quota for a process
    - **SeTcbPrivilege** – Run as part of the OS

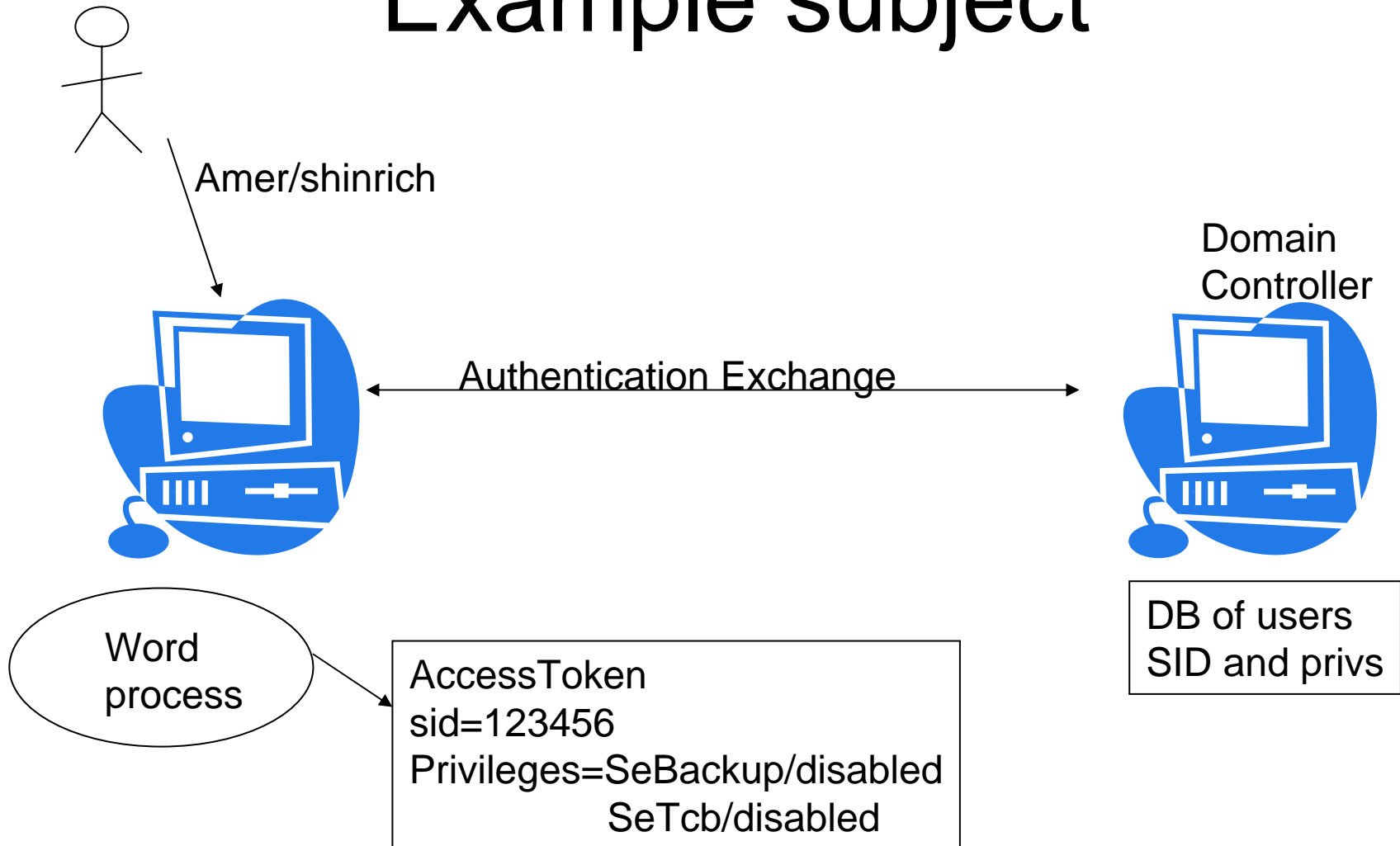
# Windows User/Group Definitions

- Control Panel/Computer Management
  - Contains the User/Group definition
- Control Panel/Local Security Settings
  - Under user rights
  - Lets the user associate users and groups with privileges

# Access Token



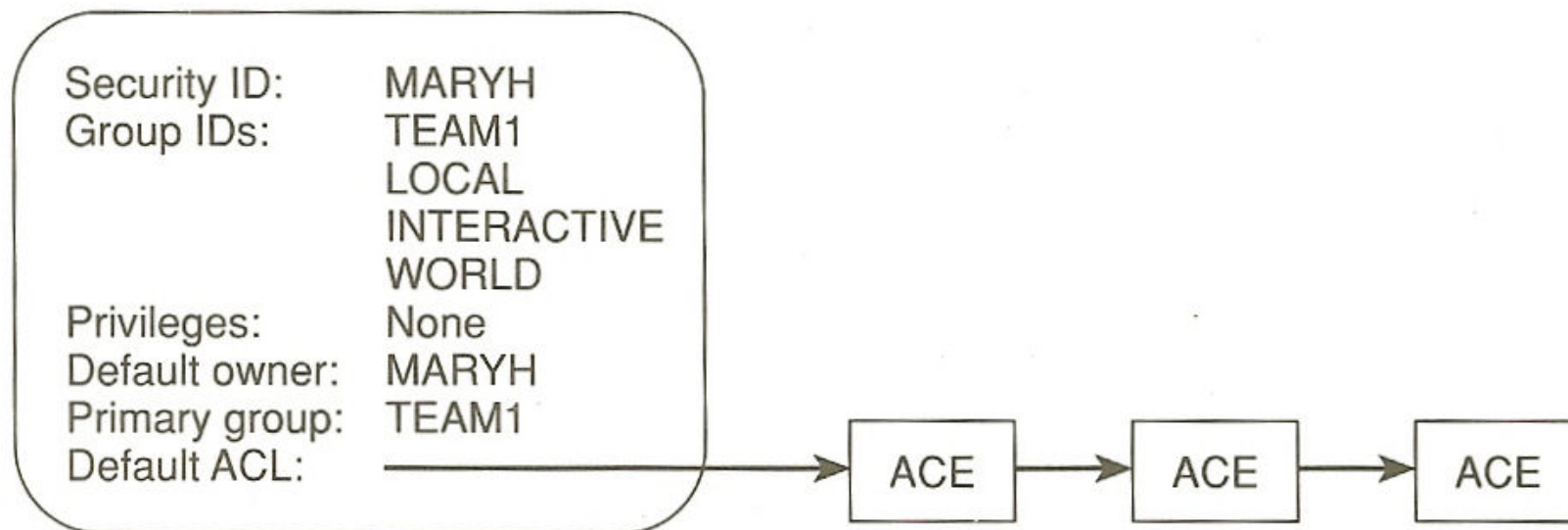
# Example subject



# More security elements

- Object – Individually secured entity such as a file, pipe, or even a process
- Rights – actions associated between object and subject.
  - Read, write, execute, audit
- Access control list (ACL)
  - Associated with an object
  - Ordered list
  - Each access control entry (ACE) contains a subject and a right
  - Evaluated by the security subsystem to determine access to protected objects.
  - Discretionary ACLs control access
  - System ACLs control audit

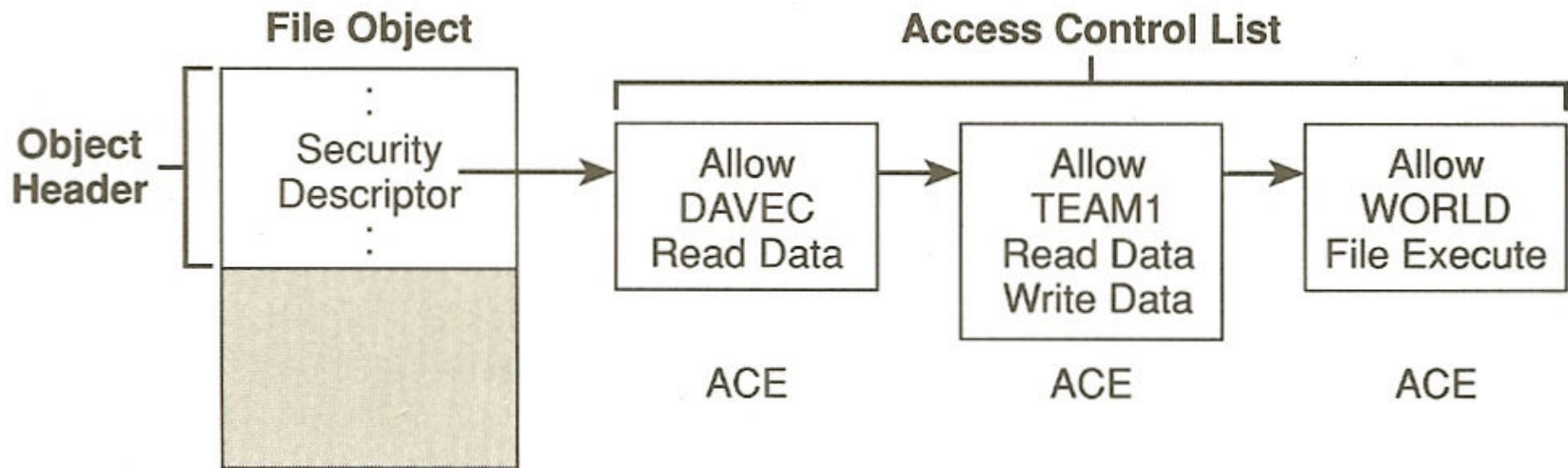
# Access Control List



# Still more security elements

- Security Descriptor – represents an object in the system. Contains the following information:
  - Object's owner
  - Object's group
  - Object's DACL
  - Object's SACL
- **AccessCheck** evaluates an ACL, subject, object triple
  - Called by many system calls
  - Can be called from user code too

# Security Descriptor



# Example ACL

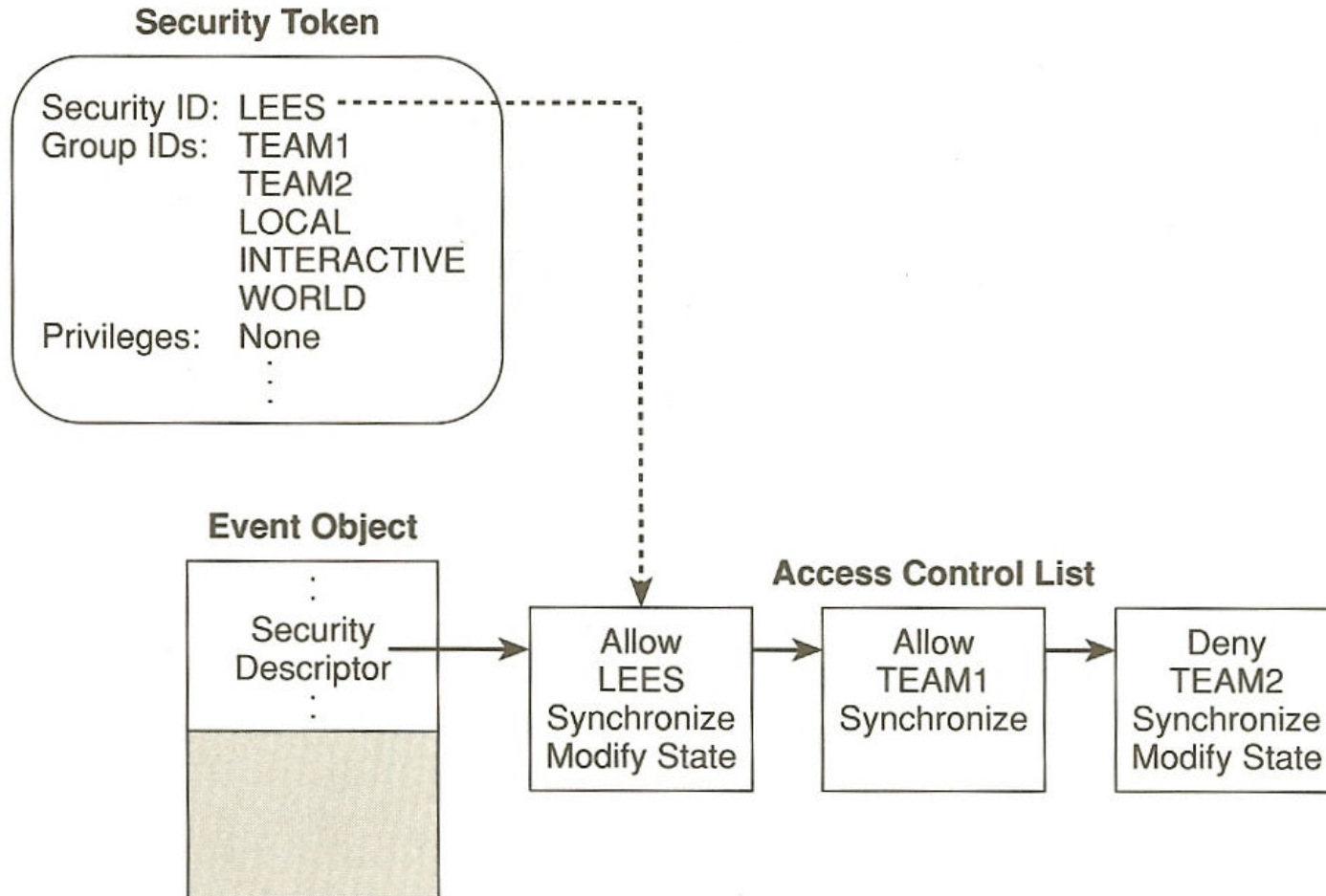
\mydocs\hw1.doc

Security Descriptor:  
sid=123456  
gid=78910  
DACL= —————→  
SACL=null

SID=Everyone:read  
SID=123456:read,write

SID=22222:deny  
SID=Everyone:read  
SID=123456:read,write

# Example Evaluation



# Working with ACLs

- Accessed via FileExplorer. Right-click file/directory and select sharing and security.
- Can programmatically create and traverse ACL's
  - See [MSDN](#) for details

# SACL controls auditing

- In addition to DACL that controls access, each object has a SACL to control auditing
  - Process access token is compared to SACL to determine whether to log
  - Also enabled by local policy

# Windows Security Problems

- Kernel level security model is reasonable
  - More consistent and complete than Unix
- So why do Windows installations have so many security problems?
  - Unix evolved from a multi-user environment
  - Windows came from a single user, stand alone environment
    - Kernel provides least privilege and fine granularity control, Windows users and app writers did not know how to use

# Vista Security Additions

- As far as I can tell, the core security mechanisms are unchanged
- Important changes in user and service mode
  - Make it easier to run at low privilege
- Additional features
  - Host intrusion detection
  - Firewall improvements
  - Network quarantine

# User Account Protection

- Enable non-privileged users to perform many operations that require privilege today
  - Add printer, update WEP keys
- Prompt user to activity privileged account if privilege is needed
- Registry and file virtualization
  - Sandboxes unprivileged users

# Windows Service Hardening

- In XP, most services are run as high privilege LOCAL SYSTEM
  - Can run as other user
  - Awkward to install because must create unprivileged user and prompt user to create password etc.
- This create a SID for each service
  - Like an unprivileged user that cannot login

# Data Protection

- Uses secure co-processor, Trusted Platform Module, that is included with many of today's laptops
- Use to implement Secure Startup
  - Detects changes to system on reboot
  - Protects from making changes to system made by mounting system from other OS

# Network Access Protection

- Network quarantine
  - Places restrictions on the characteristics of a computer that can connect to the network
  - For example can connect to the network only if the patches are up to date

# Summary

- Standard operating systems security elements
  - Unix shows security has been available for many decades
  - Windows shows security underpinnings exist in widely used OS perceived to be insecure
  - Vista security changes make it easier to use existing security mechanisms
- Security is continuing to evolve