

sicurezza dei sistemi Windows

fonte

Windows Internals 8th ed. Part 1

P. Yosifovich, A. Ionescu, M. E. Russinovich, D.A. Solomn,

Windows Internals 8th ed. Part 2

A. Allievi, A. Ionescu, M. E. Russinovich, D.A. Solomn,

Microsoft Press

summary

- Architettura
- Oggetti e loro attributi di sicurezza
- Soggetti e loro attributi di sicurezza
- Security reference monitor
 - algoritmo di controllo di accesso
- Integrità in windows
- Virtualization based security
- Autenticazione e User Access Control

architettura di windows

- hal

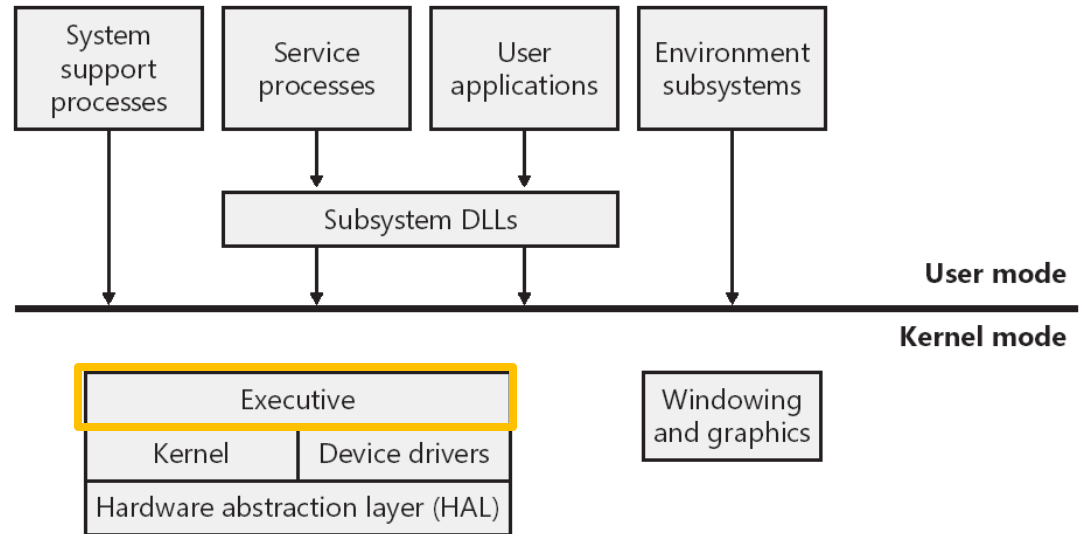
- gestisce differenza tra motherboard

- kernel

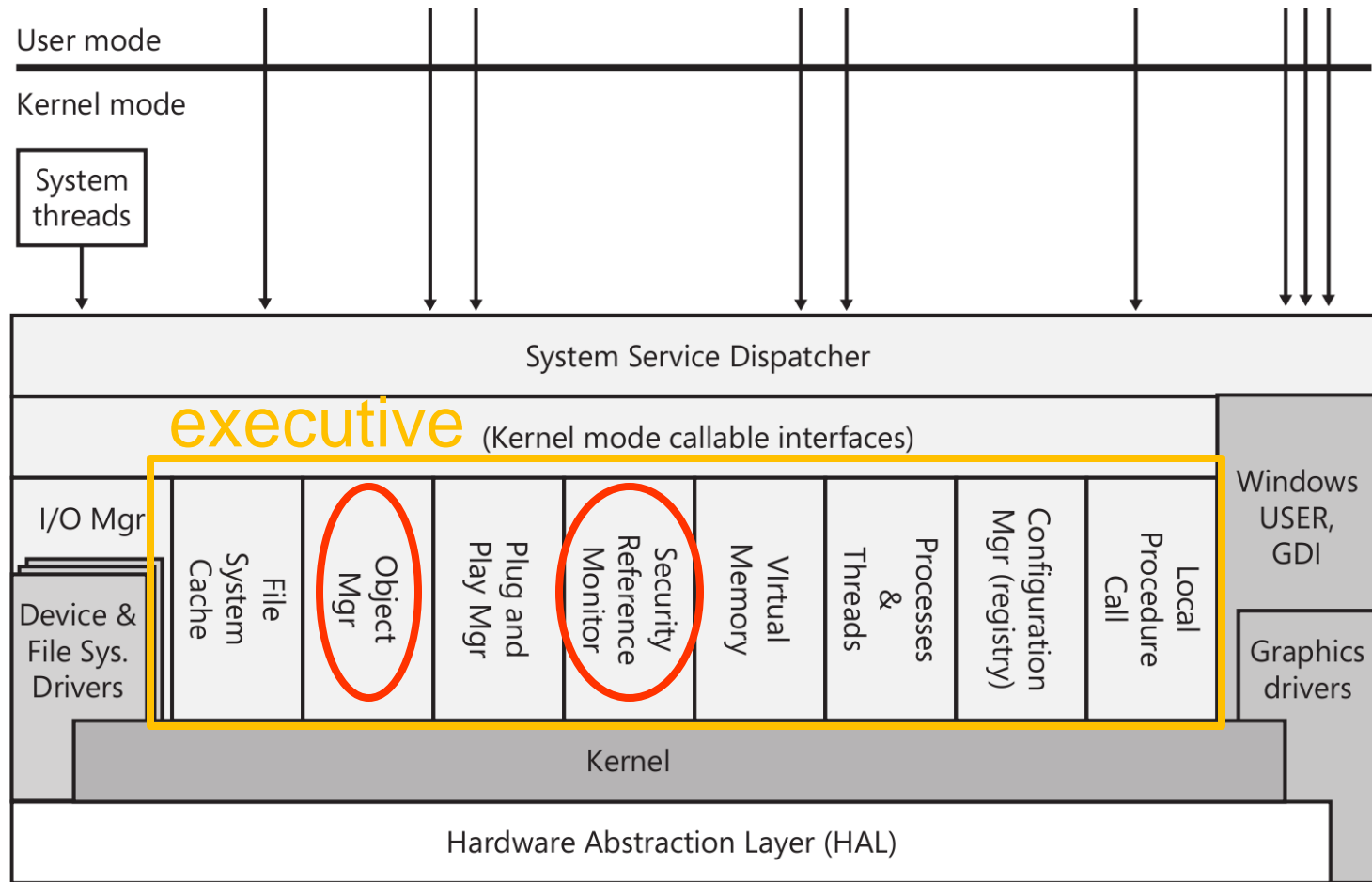
- schedulazione processi e thread, sincronizzazione per sistemi multi-processori, gestione interrupt
- non si occupa di I/O se non per il minimo indispensabile

- executive

- basato su kernel
- executive **objects**
- memory management, process/thread management, **security**, I/O, networking, inter-process communication

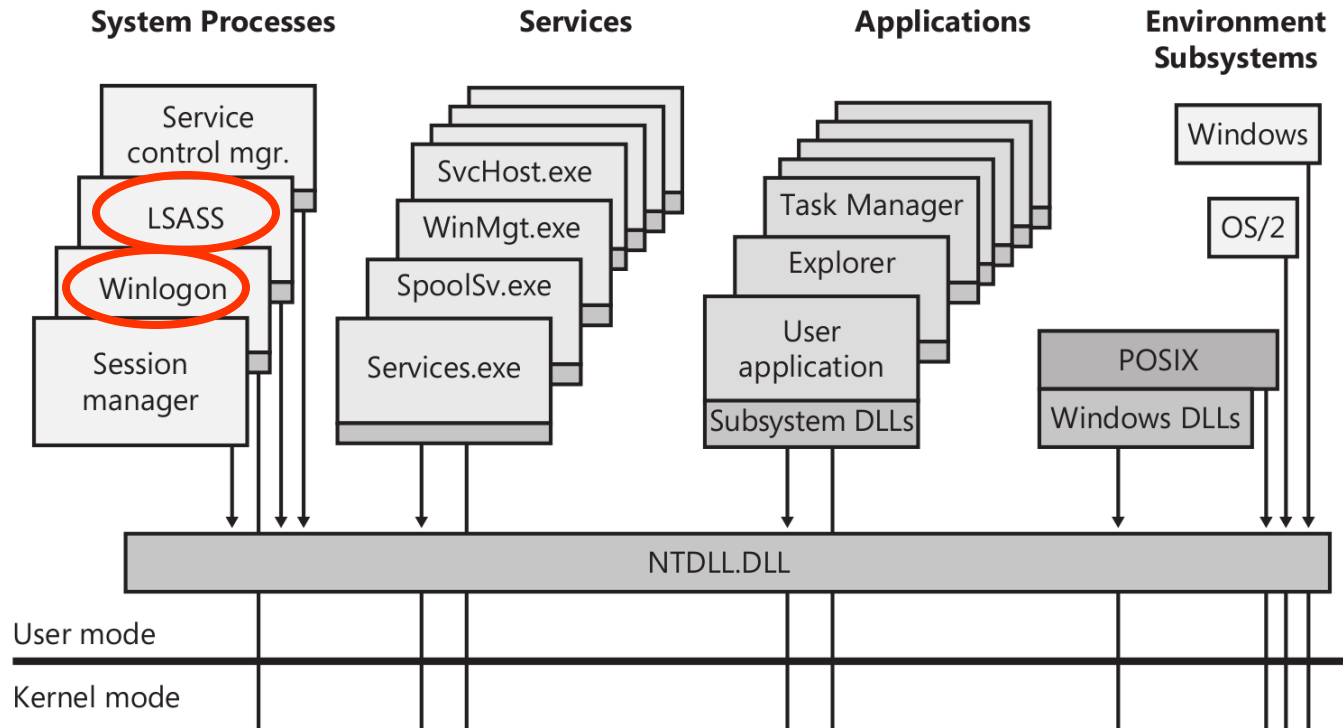


architettura di dettaglio



Hardware interfaces (buses, I/O devices, interrupts, interval timers, DMA, memory cache control, etc.)

architettura di dettaglio



risorse, oggetti e “maniglie”

- qualsiasi risorsa viene vista dai processi sotto forma di **executive object**
 - ad esempio un file aperto
- un executive object esiste in kernel space
- in user space tali oggetti sono rappresentati da **handles**
 - i processi usano gli executive object tramite le handle
- object manager
 - parte dell’executive
 - gestisce per ciascun processo una **process handle table**
 - cioè quelli che il processo può usare legittimamente
 - cioè, per ciascun soggetto, l’insieme degli oggetti su cui può agire

uso degli oggetti

- un processo ottiene un'handle tramite...
 - api di creazione di oggetti
 - api di richiesta di “apertura” di un oggetto che già esiste
- un processo usa handles come parametri di winapi

```
HFILE WINAPI OpenFile(  
    __in    LPCSTR lpFileName,  
    __out   LPOFSTRUCT lpReOpenBuff,  
    __in    UINT uStyle  
);  
  
BOOL WINAPI ReadFile(  
    __in          HANDLE hFile,  
    __out         LPVOID lpBuffer,  
    __in          DWORD nNumberOfBytesToRead,  
    __out_opt     LPDWORD lpNumberOfBytesRead,  
    __inout_opt   LPOVERLAPPED lpOverlapped  
);  
  
BOOL WINAPI CloseHandle(  
    __in  HANDLE hObject  
);
```


tipi di executive objects

type	Description
Process	A collection of executable threads along with virtual addressing and control information.
Thread	An entity containing code in execution, inside a process.
Job	A collection of processes.
File	An open file or an I/O device.
File mapping object	A region of memory mapped to a file.
Access token	The access rights for a process
Event	An object which encapsulates some information, to be used for notifying processes of something.
Semaphore/Mutex	Objects which serialize access to other resources.
Timer	An objects which notifies processes at fixed intervals.
Key	A registry key.
Desktop	A logical display surface to contain GUI elements.
Clipboard	A temporary repository for other objects.
WindowStation	An object containing a group of Desktop objects, one Clipboard and other user objects.
Symbolic link	A reference to other objects, via which the referred object can be used.

generalità sui controlli di accesso

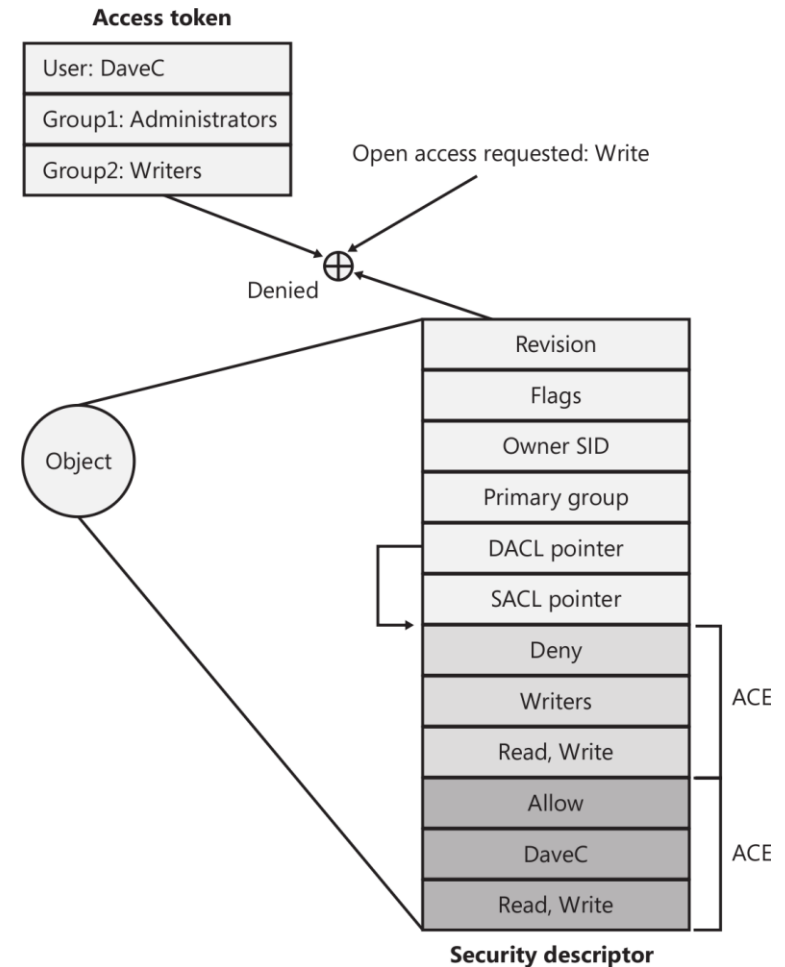
- discretionary (DAC)
 - identificatori di utenti, access list, ecc.
 - modificabili dagli utenti
- mandatory (MAC)
 - livelli di integrità
- i controlli sono effettuati alla richiesta dell'handle
 - i privilegi ottenuti sono associati alla handle
 - verifica molto più efficiente

Security Reference Monitor

- security reference monitor (SRM)
 - parte dell'executive
 - autorizza o nega l'accesso
- funzione principale di SRM: seAccessCheck
 - richiamata dall'object manager durante l'apertura o la creazione di un oggetto
- se l'accesso è accordato il risultato viene memorizzato nella process handle table dall'object manager associato alla handle
 - successive alterazioni dei diritti degli oggetti non hanno effetto sulle handle già aperte

Security Reference Monitor

- input
 - soggetto: AccessToken del processo
 - in particolare i SID
 - oggetto: oggetto da accedere
 - cioè il suo security descriptor
 - DACL
 - integrity level
 - operazione: la AccessMaks richiesta
- output
 - esito: successo o fallimento



strutture dati rilevanti per la sicurezza in windows

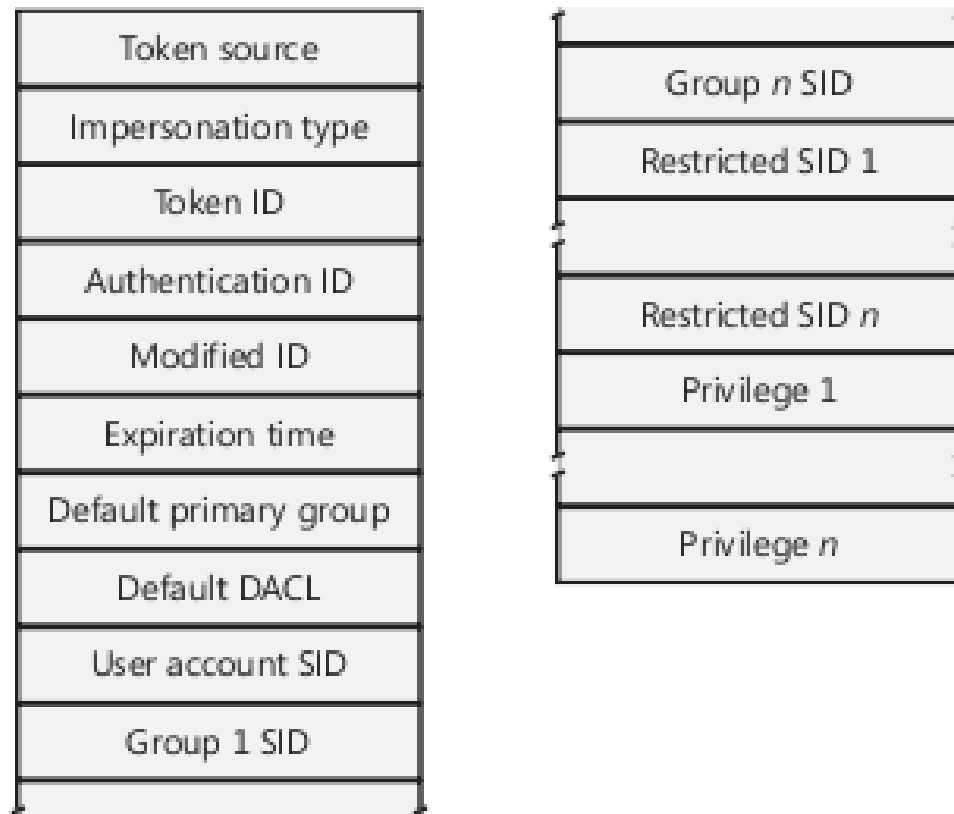
- Security ID (SID)
 - codici univoci che identificano i “soggetti” (o insiemi di soggetti) in windows
 - ad esempio gli utenti e i gruppi, ma non solo
 - numerici di lunghezza variabile
 - concatenazione di vari codici e sottocodici numerici
 - codifica come stringa: S-1-5-21-1463437245-1224812800-863842198-1128
- access mask
 - parola di 32 bit che identifica un insieme di diritti
 - la semantica dei bit cambia a seconda del tipo dell’oggetto a cui fa riferimento
 - una access mask viene associata a ciascuna handle al momento della creazione se il controllo di accesso è andato a buon fine

strutture dati rilevanti per la sicurezza in windows

- security descriptor
 - è parte di ciascun executive object
 - contiene l'integrity level dell'oggetto (usato per MAC)
 - contiene due access control list (una è usata per DAC)
 - sono sequenze di **access control entry (ACE)**
 - DACL (Discretionary ACL)
 - autorizzano o limitano accessi
 - SACL (System ACL)
 - identificano quali accessi vanno loggati
- ACE
 - semplificando può essere considerata una tripla: deny o allow, SID, Access Mask
 - contiene anche dei flag

strutture dati rilevanti per la sicurezza in windows

- access token (a.k.a. security context)
 - è un executive object associato a ciascun processo o thread
 - rappresenta le credenziali associate ad un processo



strutture dati rilevanti per la sicurezza in windows

- access token
 - il livello di integrità è codificato mediante l'inserimento tra i gruppi di un SID particolare
 - low: S-1-16-0x1000
 - medium: S-1-16-0x2000
 - high: S-1-16-0x3000

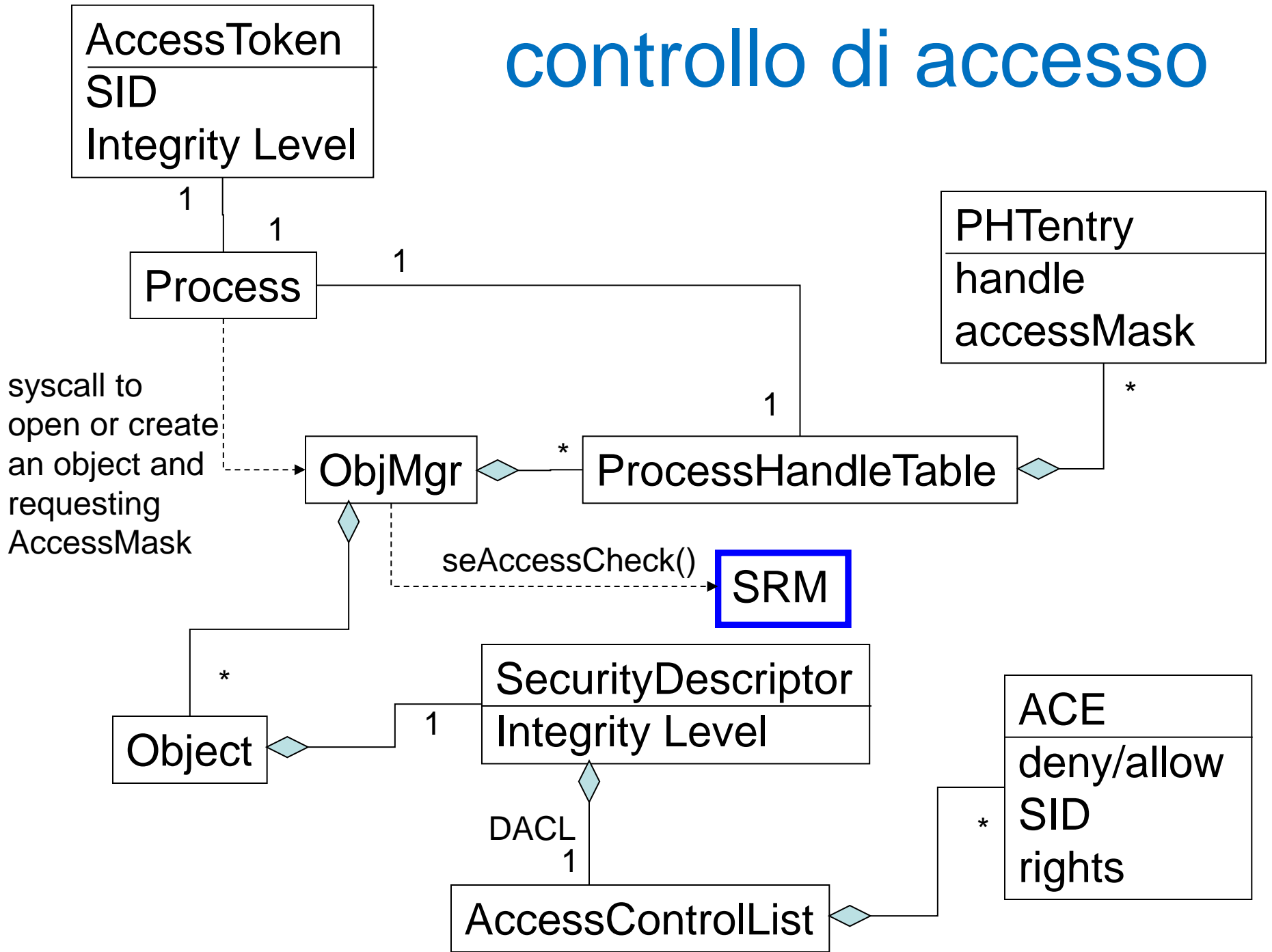
DAC: algoritmo (semplificato)

- input: am (access mask richiesta), at (access token), o (l'oggetto)
 - `o.securityDescriptor.DACL` contiene le ACE
- ACE esaminate nell'ordine in cui appaiono in DACL
- solo le ACE relative ai SID nominati nell'at sono efficaci (gli altri sono ignorati)
- per ciascuna ACE
 - se è un ACE deny e $(ACE.mask \ \&\& \ am) \neq 0$ allora fallisce
 - se è un ACE allow e $(ACE.mask \ \&\& \ am) \neq 0$ allora “accumula” i bit di maschera “allowed”
 - se tutti i bit di AccessMask sono “allowed” allora esce subito con successo
- se si arriva alla fine della lista e ciò non è vero si ha un fallimento (default deny)

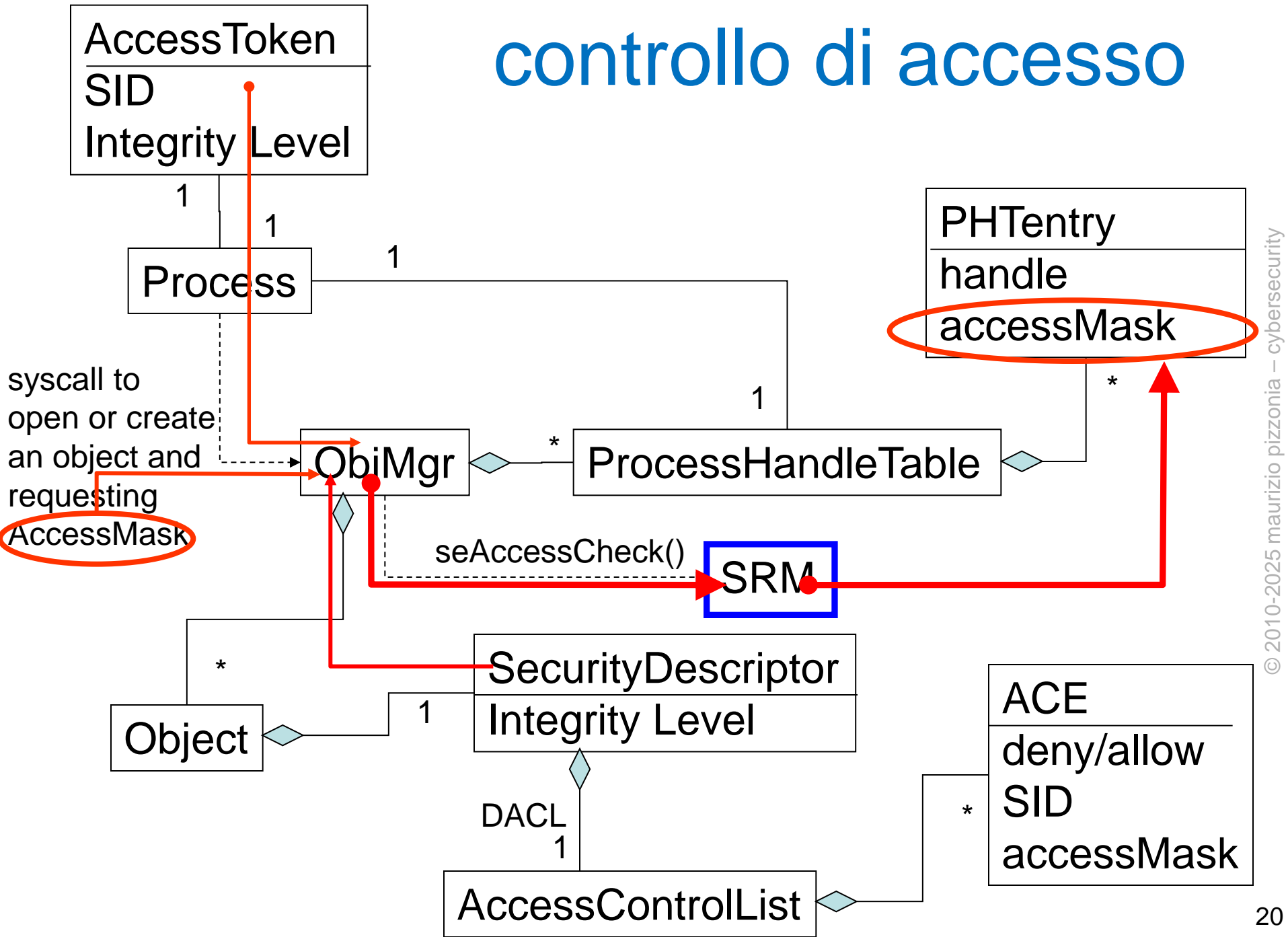
DAC: cosa non abbiamo considerato

- i SID del token possono essere abilitati, disabilitati, deny-only o restricted
- ereditarietà e gerarchie di oggetti (es. NTFS)
 - una ACE può essere marcata *inherit-only*
- owner rights
 - impedisce all'owner di modificare i permessi

controllo di accesso



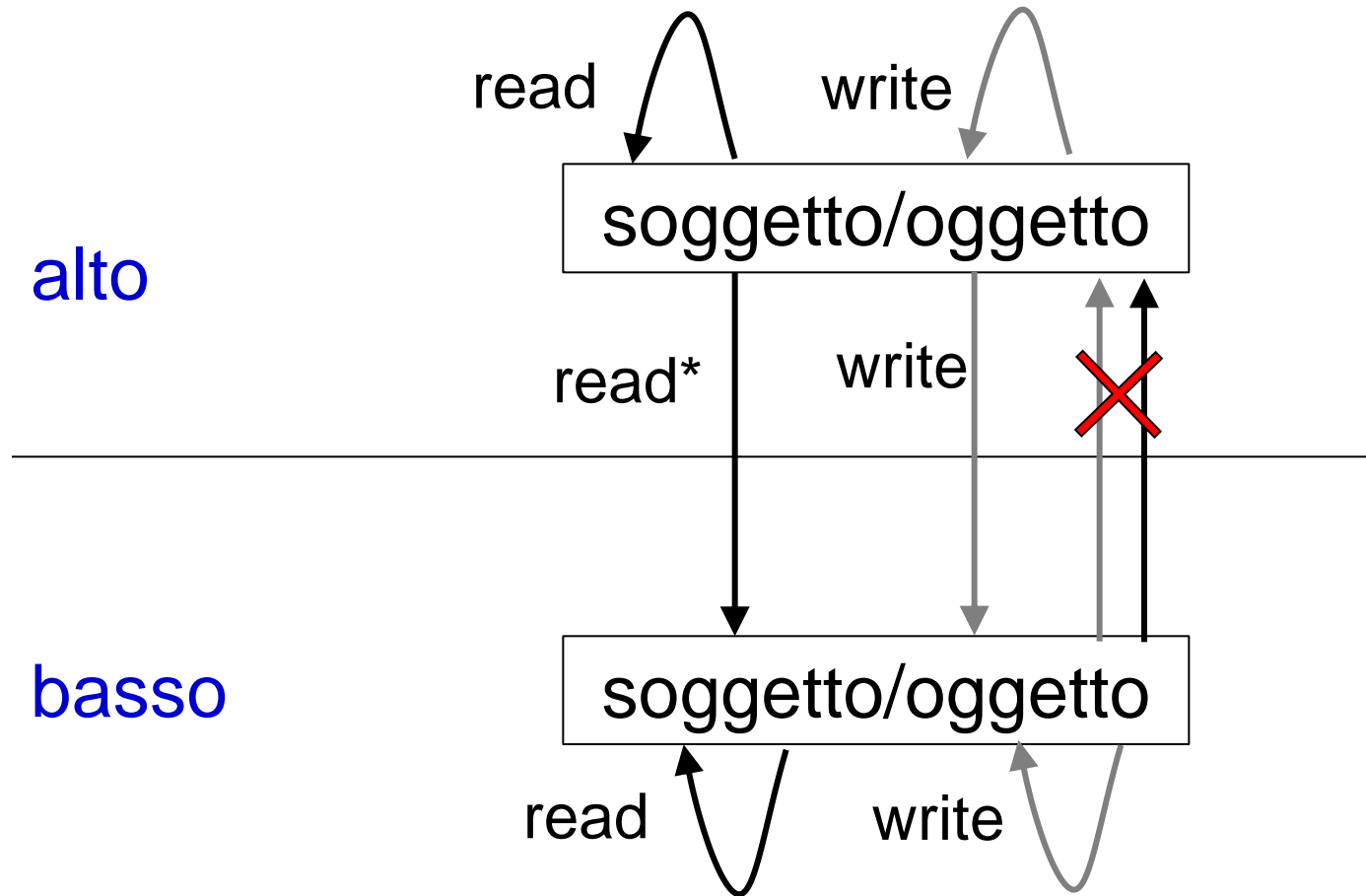
controllo di accesso



parentesi: il modello Biba

- un MAC usato per garantire integrità dei dati (non confidenzialità)
- due (o più) livelli di integrità
 - livelli bassi considerati meno sicuri: contengono malware con più probabilità
- vincolo sui flussi di dati: solo verso il basso
 - “*no read down, no write up*”
- blocca due modi di diffusione
 - attiva: scrittura verso l'alto
 - passiva: lettura eseguita dall'alto su oggetti bassi

flussi di dati ammessi dal modello Biba



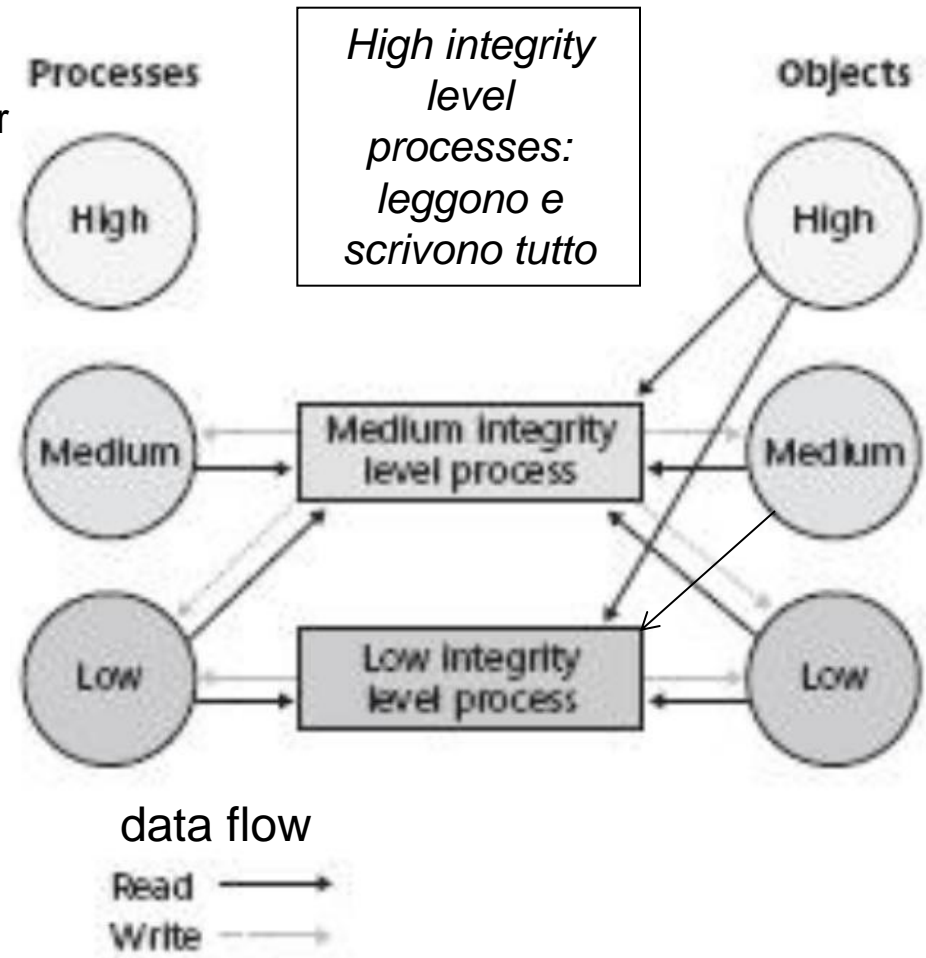
Soggetto: processo, Oggetto: file

* la read viene eseguita da un processo a livello basso su un oggetto a livello alto, la freccia indica il flusso di dati

MAC e integrity levels in windows

mandatory integrity controls o windows integrity mechanism
variante del modello Biba

- 3 livelli
 - low
 - IE protected mode
 - recently: Chrome, Acrobat Reader
 - medium
 - processi degli utenti o degli amministratori
 - high
 - processi di amministratore “elevati”
- Biba con limitazione solo per le scritture
- ammessa comunicazione tra processi tramite files, pipes, LPC
- no-read-up (un po' di confidenzialità)
 - da processi e threads

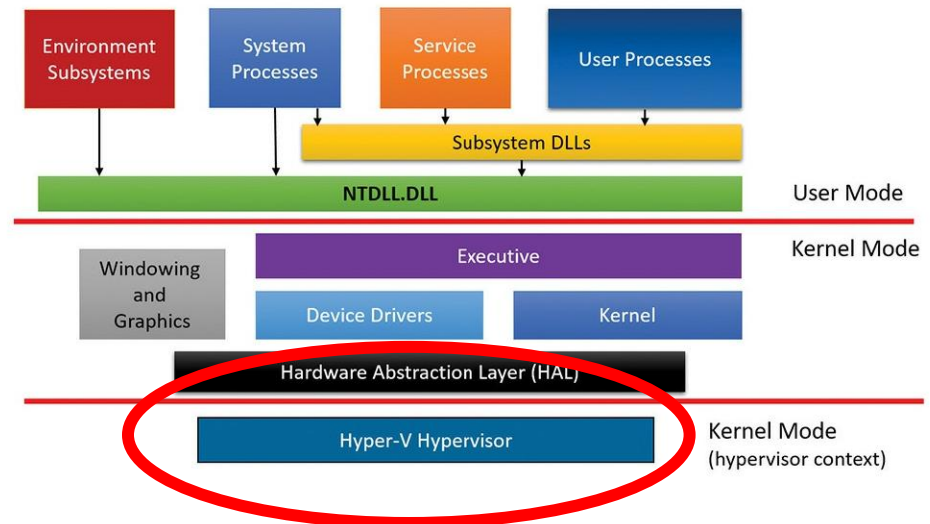


privilegi

- i privilegi sono memorizzati nell'AccessToken
- permettono di effettuare operazioni che non sono relativi ad alcun oggetto
 - es. shutdown della macchina, cambia il system time, ecc.
- non esiste un reference monitor per queste, le syscall fanno il check indipendentemente
- corrispondono ad alcuni “account rights” del local security policy

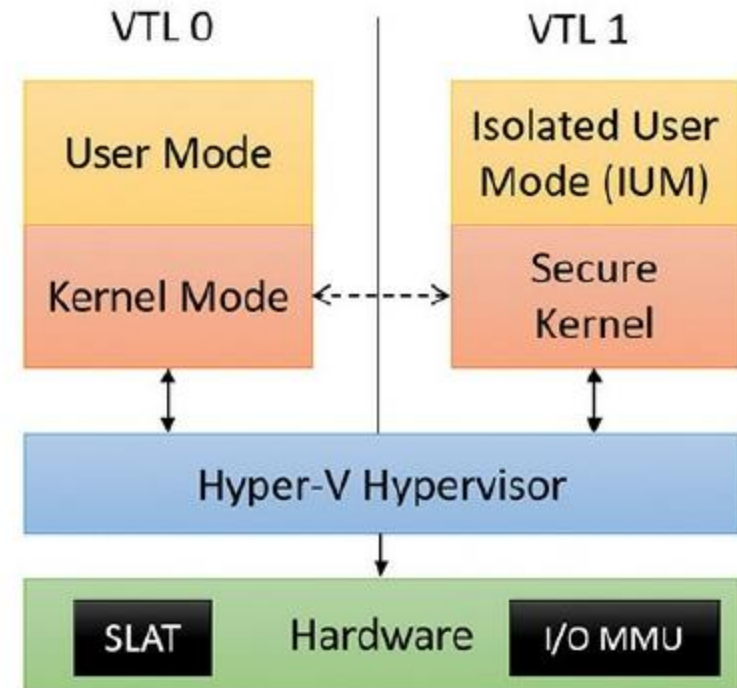
virtualizzazione come strumento per la sicurezza

- tutti i sistemi windows recenti (\geq win10) possono girare su hyper-V hypervisor (bare metal – tipo 1)
 - attivo di default nelle version “enterprise/educational” e comunque attivabile
- la virtualizzazione è usata come strumento per la sicurezza (Virtualization Base Security (VBS))
 - necessità di supporto hardware alla virtualizzazione per avere tutte le garanzie di isolamento previste



Virtualization Based Security e Virtual Trust Levels

- VTL0 installazione regolare di window
- VTL1 un “mini-windows” indipendente, isolato e molto sicuro
 - Secure kernel
 - Isolated User Mode
- la compromissione di VTL0 non permette accesso a VTL1



VTL1 e *Credential Guard*

- l'uso tipico di VTL è di mantenere “credenziali” (password e segreti crittografici) in un posto sicuro
- questa funzionalità è nota come Credential Guard

Isolated User Mode (IUM)

- i processi che girano in IUM non sono accessibili anche se il kernel windows è compromesso
 - questi processi sono anche chiamati “trustlets”
 - le trustlets hanno accesso a VTL0
 - una vulnerabilità in una trustlet è molto grave
 - sono state trovate alcune vulnerabilità di questo tipo (e mettono a rischio le credenziali di Credential Guard)

HyperVisor-protected Code Integrity (HVCI)

- si occupa di validare ogni pezzo di codice prima che venga eseguito nel kernel in VTL0
- prima dell'esecuzione si verifica
 - firma
 - integrità
- caching per efficienza, sicuro perché...
 - o marcatura read-only
 - o cache invalida se pagina modificata
- protezione contro attacco tramite Direct Memory Access
- è un esempio di *runtime attestation*

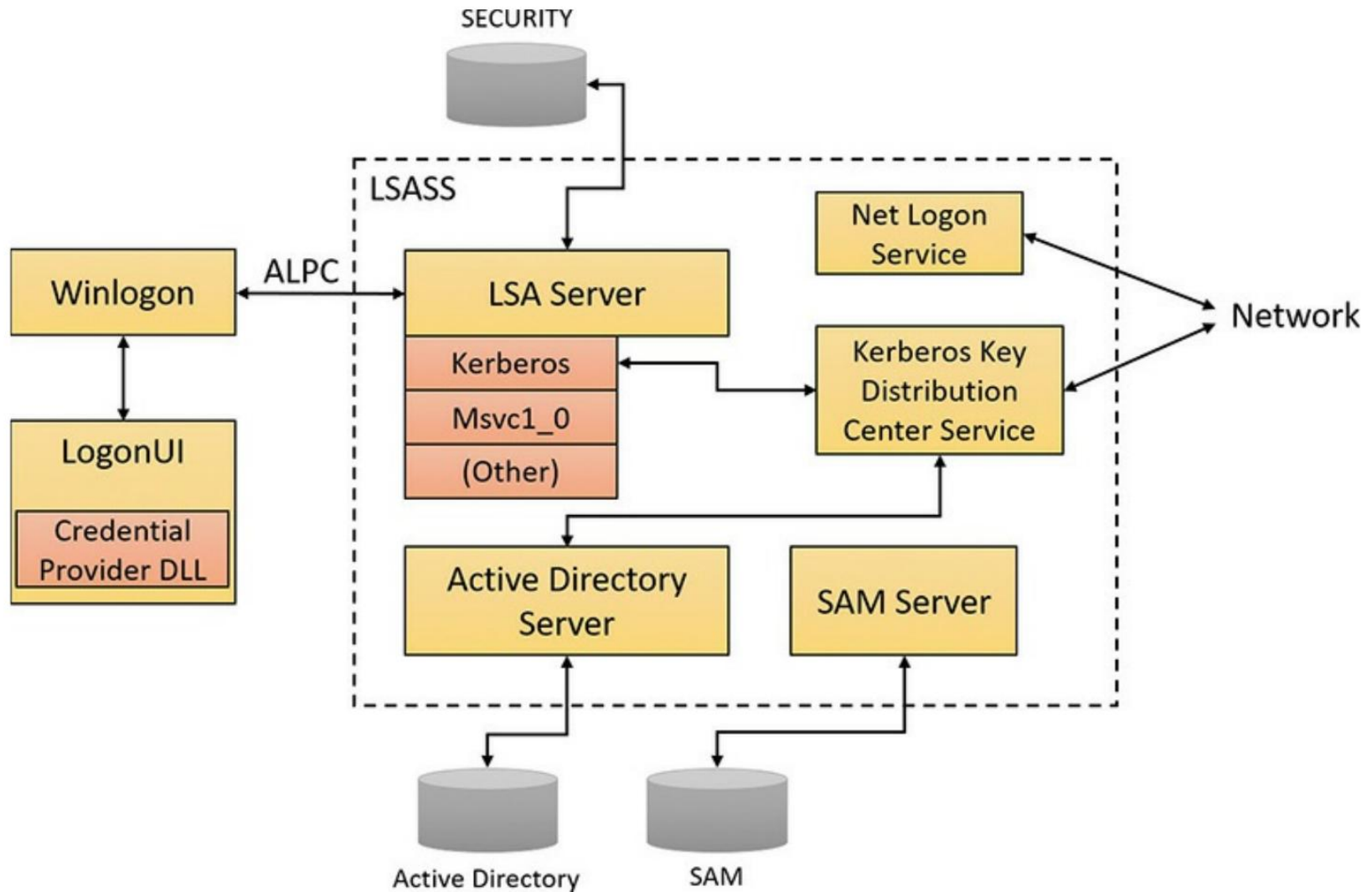
autenticazione

- componenti coinvolti

- Winlogon

- coordina il logon interattivo (cioè non via rete)
 - intercetta la Security Attention Sequence (ctrl-alt-del)
 - gestisce un desktop “sicuro”
 - lancia LogonUI per ottenere la password
 - contatta Lsass per verificare le credenziali dell'utente e ottenere l'access token
 - lancia il primo processo dell'utente con l'access token appena ottenuto

autenticazione



autenticazione

- componenti coinvolti
 - LogonUI
 - chiede la password
 - usa dei credential providers (sono delle .dll, stessi obiettivi di PAM sotto UNIX ma solo per l'input)
 - Lsass (Local Security Authority subsystem)
 - Authentication Packages
 - stessi obiettivi di PAM in unix ma solo per la verifica della password
 - LSAiso è una trustlet
 - tiene le credenziali in VTL1
 - per protocolli che richiedono password in chiaro, Token Granting Ticket (come in active directory), hash NTLM poco sicuri
 - realizza la funzionalità chiamata *Credential Guard*

autenticazione

- componenti coinvolti
 - SAM (Security Account Manager)
 - db utenti locale, nel registro HKLM\SAM
 - Active Directory
 - via rete

livelli standard ed elevation

- i processi normali hanno integrity level medio
- la procedura di **elevation** permette di lanciare un processo con integrity level high
 - è una winapi particolare che richiede la password
 - “run as ...”

User Account Control

- mettere sempre la password è tedioso
- ciò spinge ad essere sempre amministratori
- UAC: anche l'amministratore ha integrity level medium (Filtered Access Token)
- Admin Approval Mode
 - il salto ad integrità high e privilegi di amministratore richiede solo il consenso (e non la password)

Admin Approval Mode

- la procedura di richiesta del consenso viene avviata in varie situazioni
 - nel framework .NET una particolare opzione in un file dell'applicazione fa chiedere il consenso
 - euristiche che riconoscono i programmi di installazione

nuovi approcci: MSIX e UWP

- MSIX: è il formato di pacchettizzazione moderno per app Windows
- UWP (Universal Windows Platform): applicazioni che girano in container
 - compatibili con molti dispositivi (es. pc, tablet, xbox, ecc.)
- UAC non interviene quasi mai poiché...
 - Firma digitale obbligatoria: se la fonte è fidata per molte installazioni l'elevation richiesta
 - Riduce la necessità di modificare il sistema (registry, file system), quindi non richiede elevation
 - Le UWP hanno privilegi minimi

altre caratteristiche

- virtualizzazione
 - del registro e del filesystem
- impersonation
 - per ottenere token alternativi (con meno privilegi)