Informationssicherheit

7. Autorisierung

Prof. Dr. Christoph Skornia
christoph.skornia@oth-regensburg.de

Autorisierung

Typische Einsatzbereiche:

- Zugriff auf Ressourcen in einem Computernetzwerk
- Installation oder Benutzung von Software

Anmerkung:

- Umsetzung der Autorisierung durch den Referenzmonitor
- Basis ist die Policy

Generelle Struktur: Zugriffsmatrix

Implementierungskonzepte:

- **Zugriffskontrolllisten (Access Lists)**
- **Capability-Listen**
- Domain-Type-Enforcement
- Lock-Key-Konzept

Subjekte	Objekte		
	Datei 1	Datei 2	
Bill	owner, r, w	w	
Joe	r,w	Joe	Joe
Alice		owner,r,x	

Autorisierung

Access List (ACL)

Vorteile:

- vergebene Rechte sind effizient für Objekte bestimmbar
- Rechterücknahme ist meist effizient realisierbar
- dezentrale Kontrolle möglich, denn über ACL sind Rechte idR direkt mit Objekt verknüpft

Nachteile:

- Bestimmen der Subjekt-Rechte idR sehr aufwändig
- schlechte Skalierbarkeit bei dynamisch wechselnder Menge von Subjekten

Capability Lists

Vorteile:

- Einfache Bestimmung der Rechte eines Subjekts
- Einfache Zugriffskontrolle: nur noch Ticketkontrolle!
- festlegen von Protection Domains mit Capabilities möglich

Nachteile:

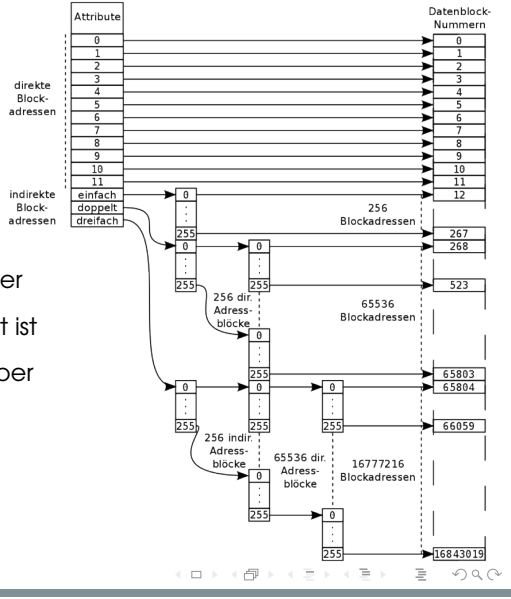
- Rechterücknahme schwierig, Kopien müssen gesucht werden!
- keine Subjekt-Ticket-Kopplung, Besitz berechtigt automatisch zur Wahrnehmung der Rechte
- Objekt-Sicht auf Rechte schwierig: wer darf was!

Subjekte	Objekte		
	Datei 1	Datei 2	
Bill	owner, r, w	w	
Joe	r,w	Joe	Joe
Alice		owner,r,x	

Autorisierung: Beispiel Unix

Zugriffskontrolle unter Unix/Linux

1. Open-System-Call: Angabe des Zwecks `r, w, x`
2. Aktionen des Unix Kerns (**ACL**)
   1. Laden der i-node der zu öffnenden Datei `XYZ`
   2. Prüfen, ob zugreifender Prozess gemäß der ACL der Datei zum gewünschten Zugriff `r, w, x` berechtigt ist
   3. Falls o.k., return File-Handle: enthält Information über zulässige Zugriffsrecht `r, w, x`
3. Erstellung eines File-Handle (**Capability**)

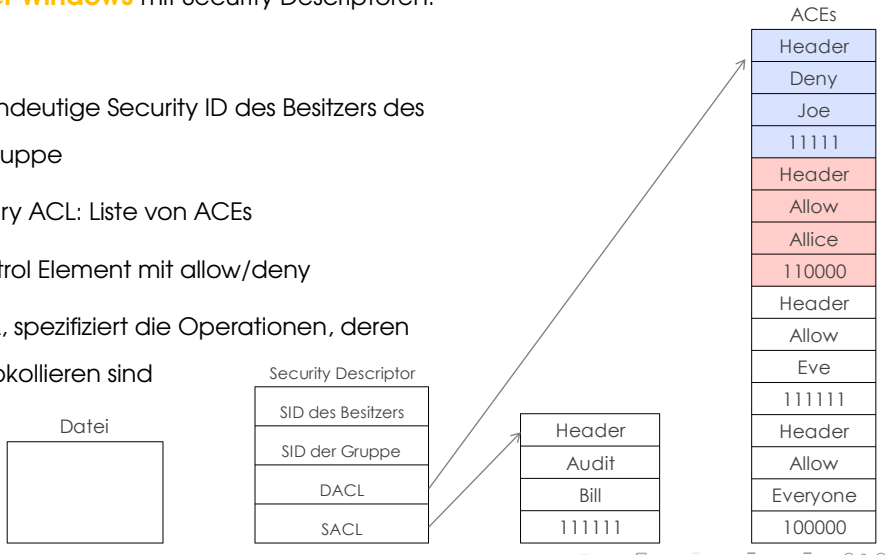


Autorisierung: Beispiel Windows

Zugriffskontrolle unter Windows mit Security Descriptoren:

Diese enthalten:

- SID = systemweit eindeutige Security ID des Besitzers des Objekts und der Gruppe
- DACL = Discretionary ACL: Liste von ACEs
- ACE = Access Control Element mit allow/deny
- SACL = System ACL, spezifiziert die Operationen, deren Nutzungen zu protokollieren sind



Autorisierung: Beispiel Windows

Ablauf der Autorisierung

1. Falls keine ACL festgelegt, dann ist Zugriff erlaubt
2. Falls Subjekt der Objekt-Owner ist, dann
   1. besitzt es automatisch read und write-DACL-Rechte,
   2. keine DACL-Prüfung, falls keine weiteren Rechte angefordert,
   3. sonst: DACL-Prüfung
3. DACL-Prüfung: ACEs werden nach FIFO durchlaufen falls die SID in der ACE mit der SID oder einer aktivierten Gruppen-SID im AT übereinstimmt
   1. falls Allow-ACE: Rechte der ACE werden gewährt

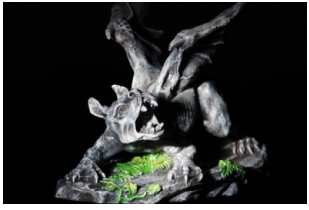
      falls alle angeforderten Rechte gewährt:

      Algorithmus terminiert mit Access allowed
4. falls es eine Deny-ACE ist : Zugriffsverbot, falls in ACE mindestens eines der angeforderten Rechte verboten ist, Algorithmus terminiert mit Access denied

Authenfizierung & Autorisierung in Kombination

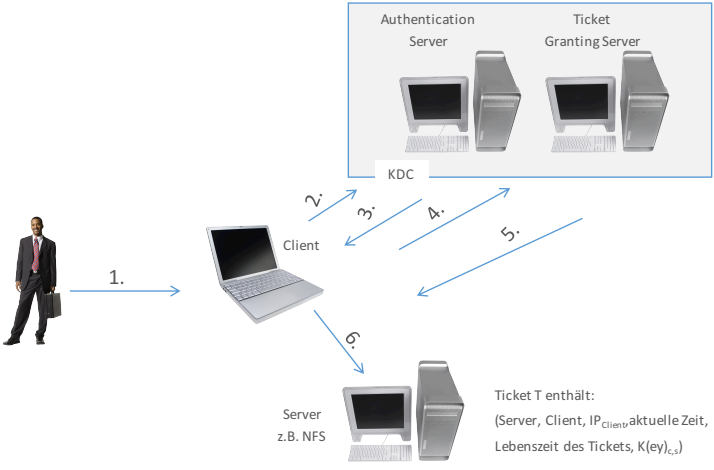
Kerberos-Protokoll

- Name: gr. Mythologie: 3-köpfiger Hund, der den Eingang zum Hades bewacht.
- 1983 im Athena Projekt am MIT entwickelt
- z.Zt. im Einsatz: Version 5 (RFC 4120)
- Ziele von Kerberos:
  - Authentifizierung von Subjekten, genannt Principals:
  - u.a. Benutzer, PC/Laptop, Server
  - Austausch von Sitzungs-Schlüsseln für Principals
  - Single-Sign-on für Dienste/Personen in einer administrativen Domäne (realm) (bzw. auch Inter-realm)



Authenfizierung & Autorisierung in Kombination

Kerberos-Ablauf



$$A_c = (c, addr, timestamp)$$

Schritt	Inhalt
1	Joe, K_{Joe} (Passwort, etc)
2	Joe, TGS, $Nonce_1$, $\{Joe,Time\}_{K_{Joe}}$
3	$\{K_{Joe,TGS}, Nonce_1\}_{K_{Joe}}$, $\{T_{Joe,TGS}\}_{K_{TGS}}$
4	$\{A_{Joe}\}_{K_{Joe,TGS}}$, $\{T_{Joe,TGS}\}_{K_{TGS}}$, NFS, $Nonce_2$
5	$\{K_{Joe,NFS}, Nonce_2\}_{K_{Joe,TGS}}$, $\{T_{joe,NFS}\}_{K_{NFS}}$
6	$\{A_{Joe}\}_{K_{Joe,NFS}}$, $\{T_{Joe,NFS}\}_{K_{NFS}}$

Danke für Ihr Interesse

Fortsetzung folgt