SAT 2.0 – Security Analysis Tool 2.0

Helml Thomas

a member of the SAT-team at FIM

FIM – Institut für Informationsverarbeitung und Mikroprozessortechnik
Johannes Kepler Universität Linz, A-4040 Linz, Austria
Email: sat@fim.uni-linz.ac.at, Web: http://www.fim.uni-linz.ac.at/sat

Dank

Dieses Projekt wird von
Microsoft Research Cambridge UK
(http://www.research.microsoft.com/labs/cam.asp)
finanziell unterstützt.

Inhalt

- I. SAT
- II. Security im Active Directory
- III. Implementierung

I. SAT

- SAT-Team
- Motivation – Projekt SAT
- Historie
- SAT Architektur

SAT-Team

- Hörmanseder: Projektinitiator, -leiter
- Achleitner: NTFS, Registry
- Helml: ADS, DB
- Zarda: MMC (Visualisierung), Group/User Membership
- Hanner: SAT-"Pionier"

Motivation - Projekt SAT

- Rechte werden auf Objekte vergeben – können auch per Objekt betrachtet werden
- Fragestellung: „Wo hat User X Rechte?"
  - kann mit Windows 2000 nicht befriedigend beantwortet werden
- Notwendigkeit eines Tools für Administratoren => SAT

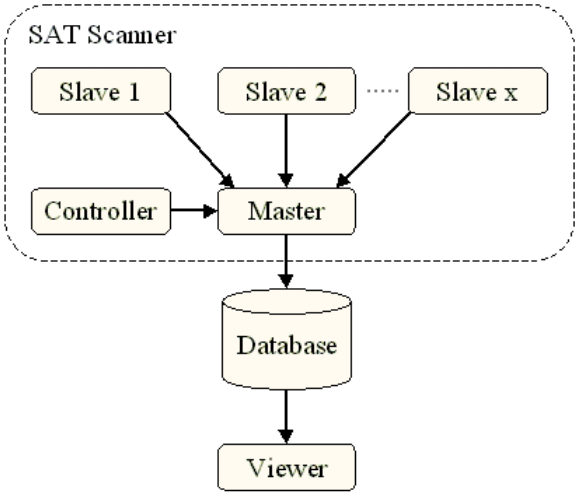
Historie 1/2

- SAT 1.0 (alt)
  - Version 1.0B wurde im Nov. 1999 fertiggestellt
  - für NT 4.0 Server
  - NTFS wird gescannt
  - keine Registry Information
  - Speicherung in Access-DB

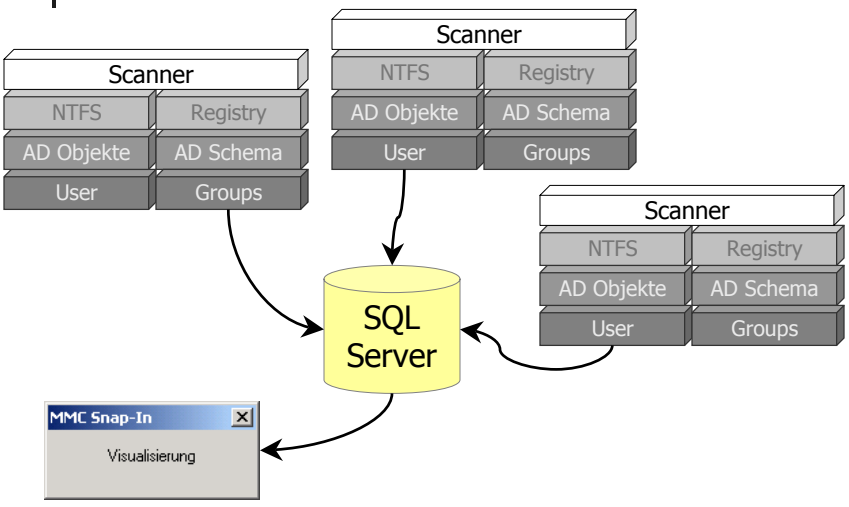
Historie 2/2

- SAT 2.0 (XSAT)
  - Projektbeginn: WS 2000/2001, geplantes Projektende: Sept. 2001
  - Windows 2000 Netzwerke
  - NTFS 5
  - Registry
  - Active Directory
  - SQL-Server zur Speicherung
  - Visualisierung mittels MMC Snap-In

SAT 1.0 Architektur



SAT 2.0 Architektur

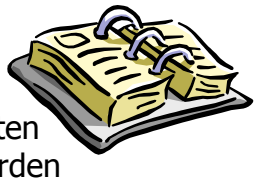


II. Security im Active Directory

- Active Directory Services
- Active Directory Schema
- Administration
- Security (ACE, ACL, Security Descriptor)
- Vererbung von Security
- „Standardrechte"

Active Directory Services 1

- Was ist ein Directory?
  - Ansammlung von Informationen (Objekten), die in einer bestimmten Art gruppiert bzw. aufgelistet werden



- Beispiel: Telefonbuch vs. Gelbe Seiten
  - Telefonbuch: Namen und dazugehörige Nummern alphabetisch geordnet
  - Gelbe Seiten: Namen und dazugehörige Nummern nach Kategorien aufgelistet

Active Directory Services 2

- Was ist ein Directory Service?
  - Ein spezieller Dienst (Service) für
    - den Zugriff auf Informationen in einem Verzeichnis
    - die Speicherung von Informationen in ein Verzeichnis

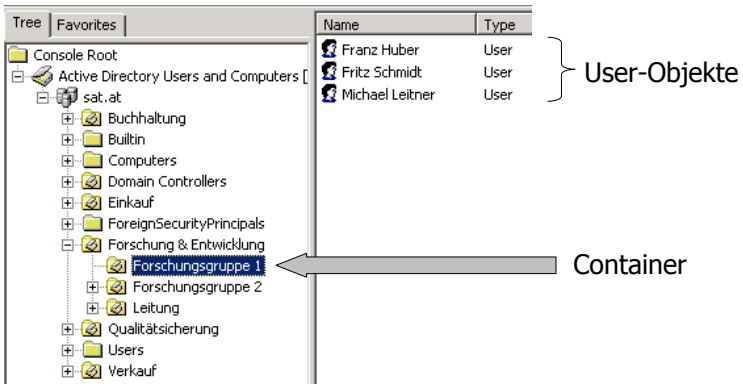
- Beispiel: „Telefonzentrale"
  - Mitarbeiter bietet seine Dienste an:
    - für Zugriff auf das Telefonregister
    - für das Eintragen von neuen Nummern

Active Directory Services 3

- Active Directory
  - Verzeichnisdienst von Microsoft
  - Speicherung von Informationen über Firmenstruktur (Firma, Abteilungen, Mitarbeiter, Computer, ...)
    - Zugriff auf diese Informationen
  - notwendigerweise: Sicherheitsmechanismen
  - Erweiterbar!
  - Usermanagement für Domains => AD
  - andere Anwendungen speichern ihre Userdaten im AD ab (z.B. Exchange)

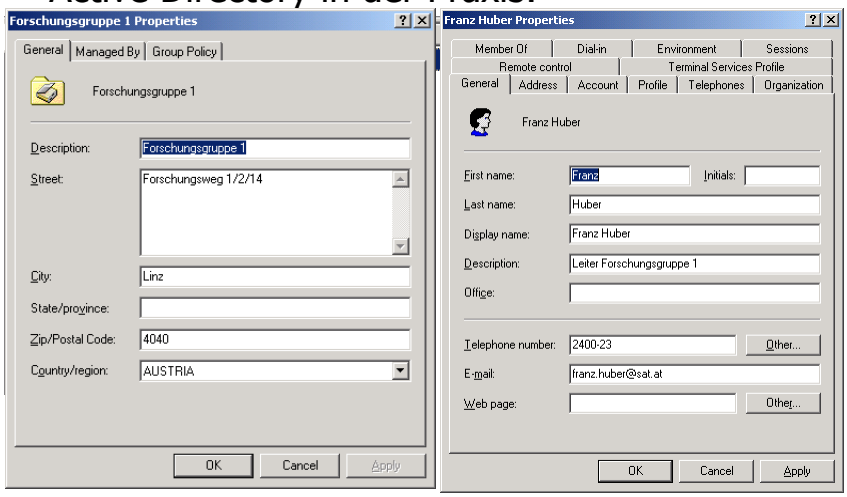
Active Directory Services 4

- Active Directory in der Praxis:



Active Directory Services 4

- Active Directory in der Praxis:



Active Directory Services 5/5

- Zusammenfassung:
  - Hierarchische Speicherung
  - Container sind eine spezielle Art von Objekten
  - Container beinhalten Container und/oder andere Objekte
- Jedes Objekt muss eindeutig identifiziert werden können
  => Distinguished Name

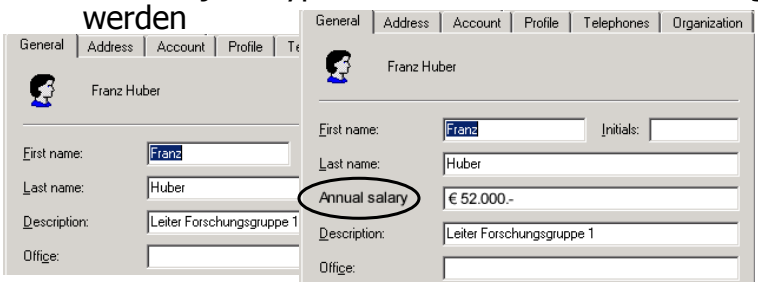
DOS 8.3	Distinguished Name (DN)
\sat.at\Forsch~1\Forsch~1	OU=Forschungsgruppe 1, OU=Forschung& Entwicklung, DC=sat, DC=at

Active Directory Schema 1/2

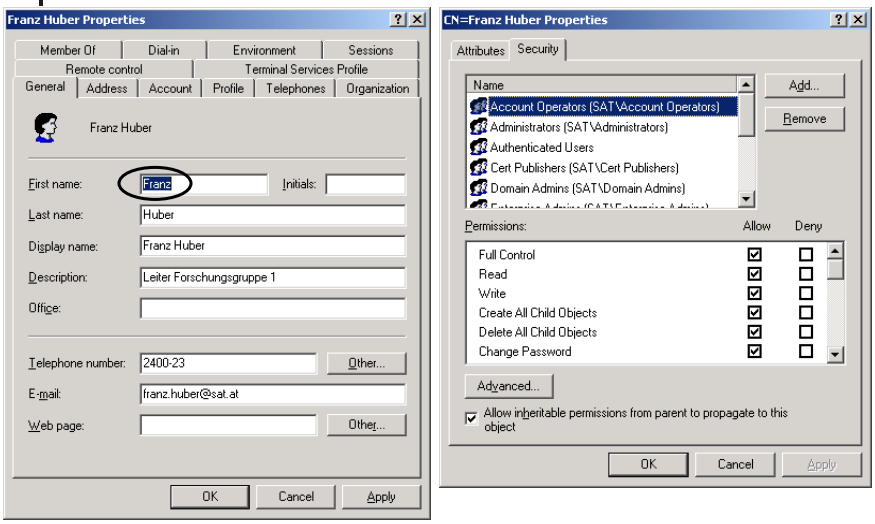
- Objekte haben unterschiedliche Eigenschaften
  - eine Eigenschaft eines Objekts wird im AD als Attribut bezeichnet
  - jedes Objekt hat eine bestimmte Anzahl von Attributen
- es muss also verschiedene Objektarten (=Typen / Klassen) geben
  - der Typ eines Objekts wird von seiner Basisklasse bestimmt => ein Objekt ist eine Instanz seiner Basisklasse
  - die Basisklasse bestimmt Art und Anzahl der Attribute

Active Directory Schema 2/2

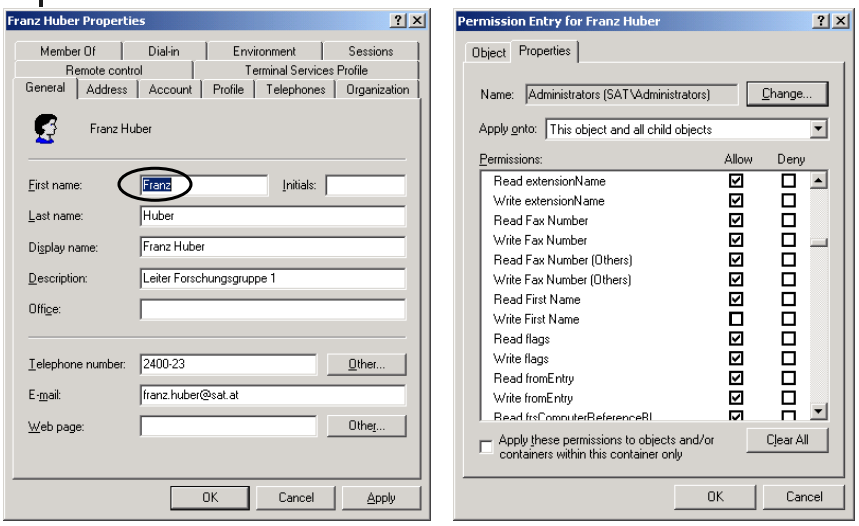
- Im Schema („Meta-Directory") werden Klassen- und Attributdefinitionen gespeichert
  - jedes Objekt im AD wird von einer Klasse aus dem Schema abgeleitet
  - das Schema ist erweiterbar
  - neue Objekt-Typen und Attribute können erzeugt werden



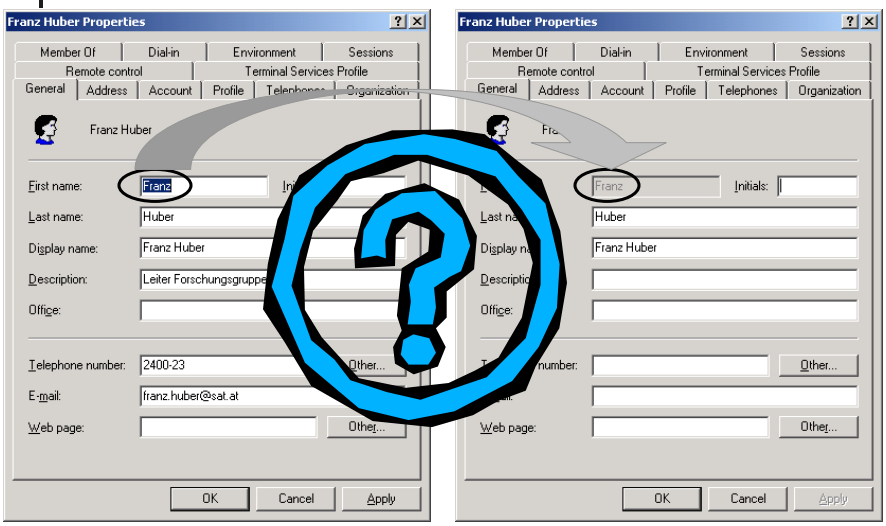
AD aus Administratorsicht



AD aus Administratorsicht

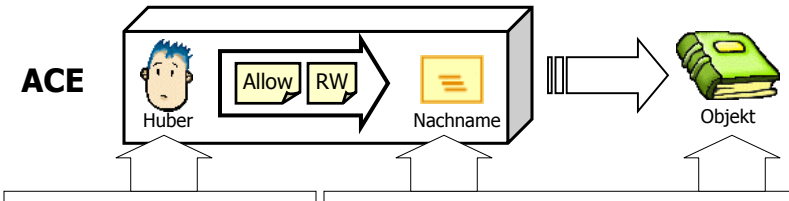


AD aus Administratorsicht



Access Control Entry (ACE)

- ACE legt fest:
  WER hat **WELCHE** Rechte auf ein Objekt?



SID (Security Identifier) identifiziert u.a. User im System eindeutig

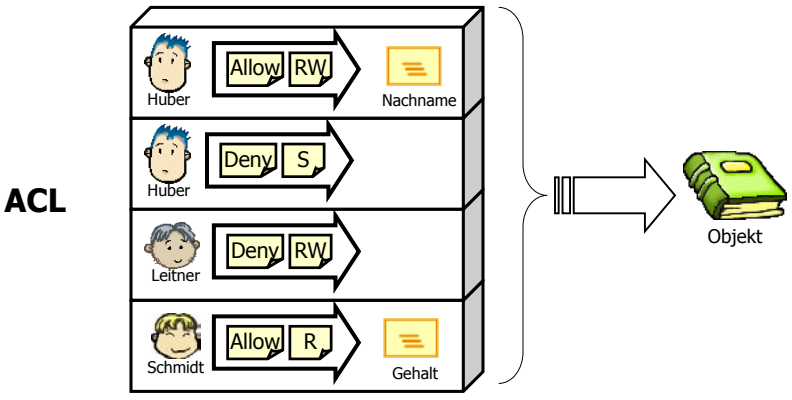
Bsp. SID: S-1-25-3-1

GUID (Global Unique Identifier) identifiziert Objekte im System eindeutig

Bsp. GUID:
{AB12E123ABFE15- AB12-E123-8325-ABFE15EA}

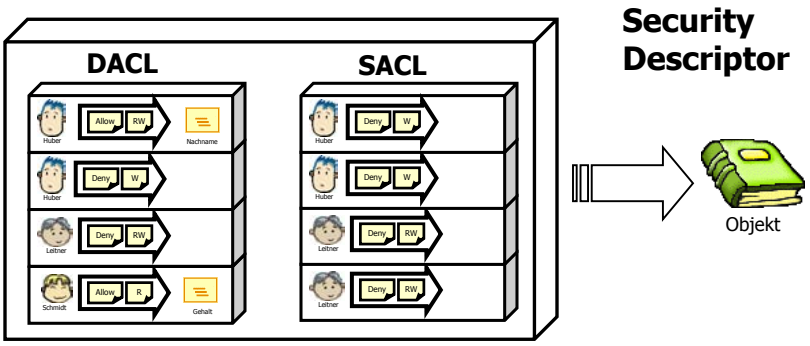
Access Control List (ACL)

- ACL: Sammlung mehrerer ACE's
  - mehrere User haben Rechte auf **EIN** Objekt



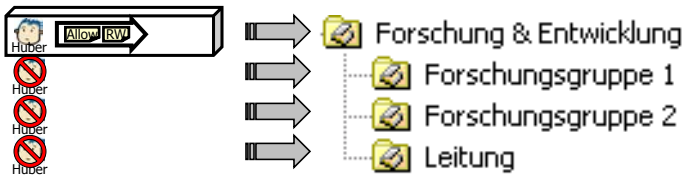
Security Descriptor (SD)

- SD besteht
  - DACL (Discretionary ACL)
  - SACL (System Audit ACL)



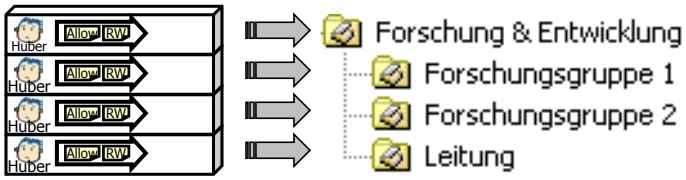
Vererbung von Rechten 1/2

- zusätzl. Komplexität durch Vererbung!
- vererbt werden können einzelne ACE's
  - Beispiel: ohne Vererbung



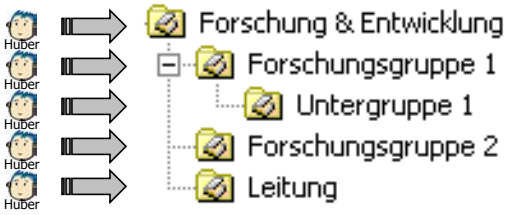
Vererbung von Rechten 1/2

- zusätzl. Komplexität durch Vererbung!
- vererbt werden können einzelne ACE's
  - Beispiel: mit Vererbung



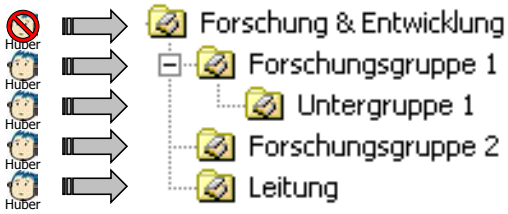
Vererbung von Rechten 2/2

- die „Vererbungs-Tiefe" lässt sich genauer spezifizieren:
  - auf das Objekt und seine Söhne



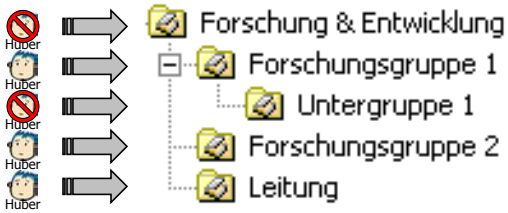
Vererbung von Rechten 2/2

- die „Vererbungs-Tiefe" lässt sich genauer spezifizieren:
  - nur auf die Söhne



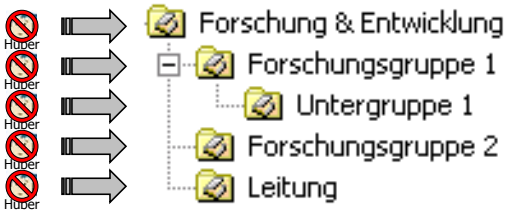
Vererbung von Rechten 2/2

- die „Vererbungs-Tiefe" lässt sich genauer spezifizieren:
  - nur auf die direkten Söhne



Vererbung von Rechten 2/2

- die „Vererbungs-Tiefe" lässt sich genauer spezifizieren:
  - zusätzlich können Ausnahmen definiert werden!



„Standardrechte" 1/2

- Wie bekommt ein neu angelegtes Objekt Rechte?
  - **Explizit:**
    - beim Erzeugen eines neuen Objekts wird SD als Parameter übergeben
    - vererbte Rechte werden dazuaddiert
  - **Schema:** falls nichts explizit angegeben
    - im Schema ist für Objekttypen ein „Standard-SD" definiert (optional)
    - vererbte Rechte werden dazuaddiert

„Standardrechte“ 2/2

- Wie bekommt ein neu angelegtes Objekt Rechte?
  - **Access Token**: falls kein „Standard-SD“ im Schema
    - Standard-DACL aus dem Access Token verwendet
  - **Voller Zugriff:** falls nichts explizit angegeben, vererbt oder im Schema gefunden wurde
    - voller Zugriff für JEDEN auf das Objekt

III. Implementierung

- Zugriffsprotokolle: LDAP vs. ADSI
- Datenbank
  - Optimierungen
- „Komprimierung“
- Ausnahmen
- Aktueller Stand

Zugriffsprotokolle 1/4

- LDAP (Lightweight Directory Access Protocol)
  - aktuelle Version: LDAPv3 (siehe RFC2251, Dez.1997)
  - ermöglicht Zugriff auf LDAP-Verzeichnisdienste
  - Plattformen:
    - Win32/Win16
    - Unix
    - für andere Plattformen: freie LDAP Client Implementation für GNU C: „Open LDAP Project"
  - Unterstützte Sprachen:
    - C (C++)
  - „Low-Level"

Zugriffsprotokolle 2/4

- ADSI (Active Directory Service Interface)
  - aktuelle Version: ADSI 2.5
  - bietet Zugriff zu verschiedenen Verzeichnisdiensten: Security Accounts Manager (SAM), Novell Directory Services (NDS), LDAPv3-konforme (z.B. AD)
  - Plattformen:
    - Win32
    - Unix (Mainsoft, Bristol Technologies)

Zugriffsprotokolle 3/4

- ADSI (Active Directory Service Interface)
  - Unterstützte Sprachen:
    - Visual Basic, VBScript
    - Java (wenn COM unterstützt wird)
    - C/C++
  - „High-Level"
    - ADSI ist ein Set von COM-Interfaces
    - darunterliegendes Protokoll: LDAP

Zugriffsprotokolle 4/4

- Entscheidung: ADSI
  - Vorteile ADSI:
    - wird von Microsoft stark forciert (zukunftssicher?)
    - Onlinedoku (MSDN)
    - in zukünftigen SAT-Versionen könnten auch andere Verzeichnisdienste (Bsp. NDS) gescannt werden
  - Vorteile LDAP:
    - schneller (Overhead bei ADSI durch COM-Kapselung)
    - kein COM
    - „Plattform-unabhängiger"

ADSI-Beispiel

- Auslesen des Namen eines Userobjekts

```
IADs *piIADs=NULL;
BSTR bstrName;
HRESULT hr;
// Bind to user object
hr=ADsGetObject(
     L"LDAP://server.sat.at/CN=Huber,OU=Forschung,
       OU=Users, DC=sat,DC=at",
     IID_IADs, (void**)&piIADs);
if (SUCCEEDED(hr))
{    // Get property
     hr=piIADs->get_Name(&bstrName);
     if (SUCCEEDED(hr)) printf("%S\n", bstrName);
     SysFreeString(bstrName)
}
piIADs->Release();
```


LDAP-Beispiel 1/2

- DN aller User der Abteilung „Forschung“

```
LDAP* psLdap = ldap_init( NULL, LDAP_PORT );
if( psLdap != NULL )
{  ULONG uErr = ldap_bind_s( psLdap, NULL, NULL,
                             LDAP_AUTH_NEGOTIATE );
   if( uErr == LDAP_SUCCESS )
   {  LDAPMessage* psResult = NULL;
      // Issue the search
      uErr = ldap_search_s(
        psLdap,                          // LDAP session
        "OU=Forschung,OU=Users,DC=sat,DC=at", // base container
        LDAP_SCOPE_ONELEVEL,            // scope of search
        "objectClass=User",              // search filter
        NULL,                           // get all attributes
        false,                          // attribute names only?
        &psResult);                     // result set
```

LDAP–Beispiel 2/2

```
      if( uErr== LDAP_SUCCESS )
      {
        // Count the entries returned
        int nEntries = ldap_count_entries( psLdap, psResult );
        cout << nEntries << " entries found" << endl;
        // Retrieve each of the returned entries
        for (LDAPMessage* psEntry =
             ldap_first_entry(psLdap, psResult);
            Entry != NULL;
            psEntry = ldap_next_entry( psLdap, psEntry ))
           cout << ldap_get_dn( psLdap, psEntry ) << endl;
      }
      ldap_msgfree( psResult );
   }
   ldap_unbind( psLdap );
}
```


Datenbank 1/3

- Microsoft SQL Server 2000
  - Zugriff über ActiveX Data Objects (ADO) via C++
  - Hauptproblem: Dokumentation sehr gut, leider nur für Visual Basic
- kein „Datensammler" mehr:
  - Scanner (Clients) schreiben alle direkt in Datenbank
- es muss (wahrscheinlich) noch in die Optimierung der DB investiert werden!

Datenbank (Optimierungen) 2/3

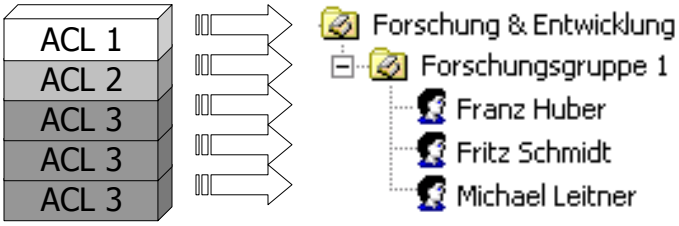
- ACLs werden nicht einfach für jedes Objekt abgespeichert
  - keine doppelte Speicherung von ACLs
  - jedes Objekt speichert lediglich eine ACL-ID
  - Stichprobe im AD (direkt nach Installation)
    - Repräsentativer Teilbaum im AD (DomainNC)
    - ~ 1100 Objekte gespeichert / ~ 55 verschiedene ACLs
    - bei direkter Speicherung: 1100 ACLs in DB
    - Optimierung: 55 ACLs in DB => 1/20 Speicherplatz

Datenbank (Optimierungen) 3/3

- mehrere Scanner arbeiten parallel und schreiben somit parallel in die DB
- um DB-Zugriffe zu minimieren => Caching
  - ACLs werden beim Scanner lokal im Cache gehalten
  - implementiert mittels einer Move-To-Front-List
  - Hoffentlich sehr effizient, da sonst für jedes (!!) Objekt in der DB geprüft werden müßte, ob ACL bereits gespeichert ist.

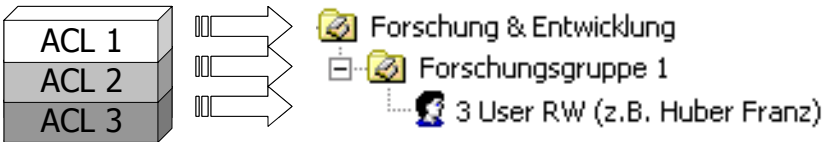
„Komprimierung“ 1/4

- Komprimierung: Zusammenfassen von Objekten
  - Betrachtung für Bsp: „Systemsicht“
  - Komprimierungsstufe 0: keine Komprimierung



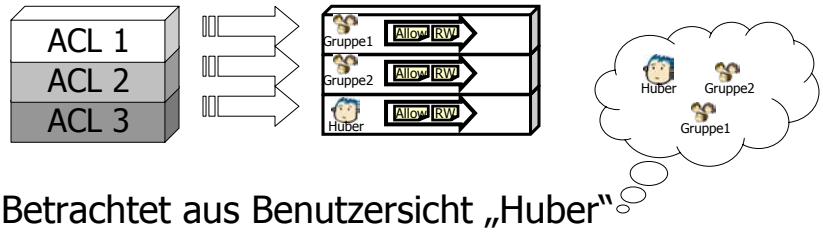
„Komprimierung“ 1/4

- Komprimierung: Zusammenfassen von Objekten
  - Betrachtung für Bsp: „Systemsicht“:
  - Komprimierungsstufe 1: Zusammenfassen von Objekten mit gleichem Typ und gleicher Security

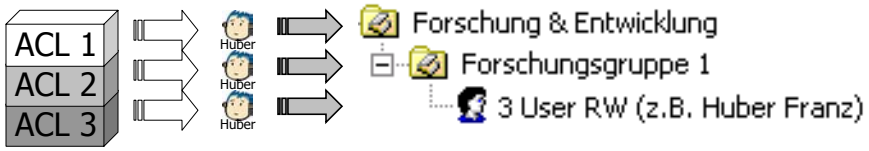


„Komprimierung“ 2/4

- In DB steht z.B.:

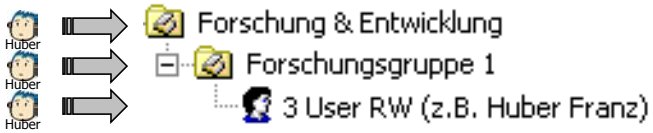


- Betrachtet aus Benutzersicht „Huber“



„Komprimierung“ 3/4

- für uns sind nur Ausnahmen interessant
- daher wird so etwas weiter komprimiert:



„Komprimierung“ 3/4

- für uns sind nur Ausnahmen interessant
- daher wird so etwas weiter komprimiert:

Komprimierung 4/4

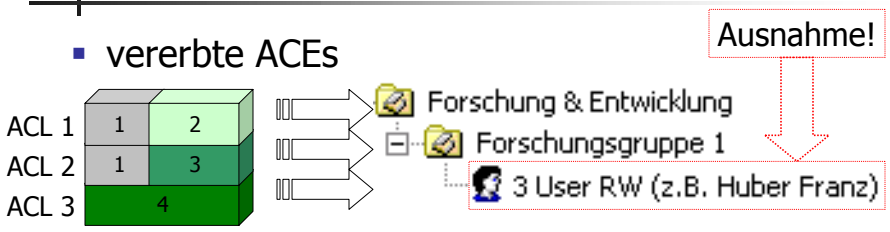
- ein User kann zusätzliche Rechte durch Gruppenmitgliedschaft bekommen
- User bekommt neuen SID durch:
  - Update von NT 4 auf Windows 2000: neue SIDs
  - User-Objekt verschieben von Domain zu Domain
- die alten SIDs sollten aber noch gültig sein
  - jedes Objekt (insb. User und Gruppen) hat ein Attribut „SID-History“, worin alle alten SIDs gespeichert werden
- muss bei der Benutzersicht berücksichtigt werden!!

Ausnahmen 1/2

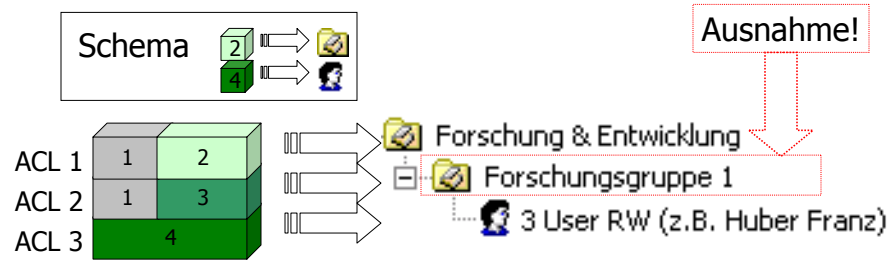
- für den Administrator sind **Ausnahmen** vom Normalfall entscheidend
  - Unterscheide:
    - vererbte ACEs (dynamisch)
    - direkt auf das Objekt vergebenen ACEs (statisch)
  - vererbte ACEs
    - Ausnahmen: dort, wo Vererbung unterbrochen wird!
  - statische ACEs
    - Ausnahmen: Abweichungen von der Standard-Security aus dem Schema!

Ausnahmen 2/2

- vererbte ACEs



- statische ACEs



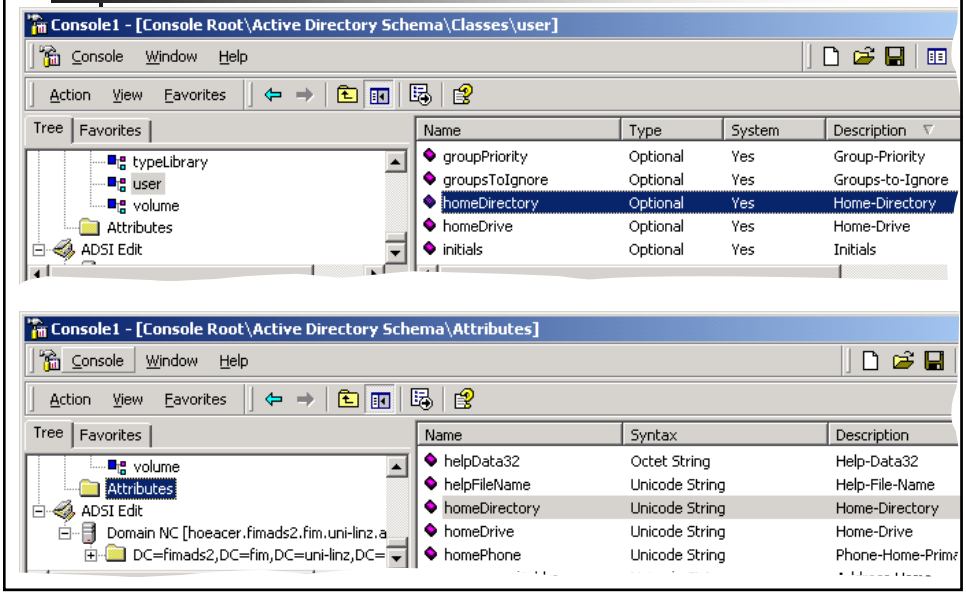
Aktueller Stand der Implementierung

- DB-Schnittstelle fertig
- Caching implementiert
- fast alle Daten in DB
- Zerlegung statischer bzw. dynamischer Security noch in Arbeit
- Testläufe unter „realen" Bedingungen werden Performance und auch Flaschenhälse zeigen
- Auswertungen mittels Access auf SQL-Server bis Visualisierung fertig

The End

Vielen Dank für Ihre Aufmerksamkeit!

Schema – Edit (Class - Attribute)



Schema-Edit (Class - Attribute)