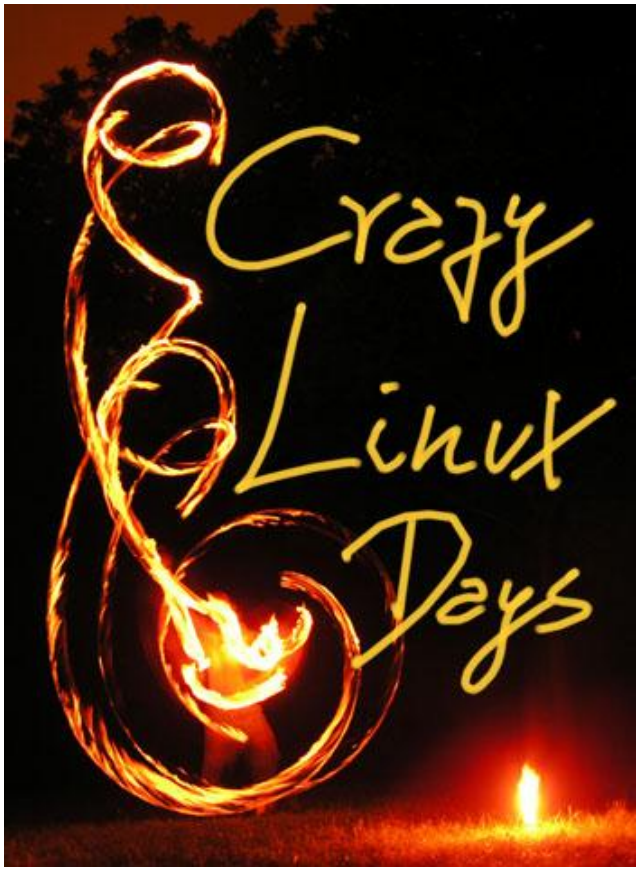



## Kerberos Administration

- Kerberos Grundlagen
- Komponenten und Begriffe
- Kommunikationsablauf
- Principals verwalten
- Kerberos Clients

Peter Jahn, MSc
Jahn@LinuxCampus.net

# Was ist Kerberos

- Antwort 1
  - Dreiköpfiger Höllenhund aus der griechischen Mythologie der den Eingang zur Unterwelt bewacht

- Antwort 2
  - Ein in 1980 durch MIT entwickeltes Authentication System um in einem nicht sicheren TCP/IP Netzwerk sicher auf Ressourcen zugreifen zu können
  - Authentication, Authorization, Audit

# Was ist Kerberos

- Eigenschaften von Kerberos
  - Basiert auf symmetrischen Schlüsseln (DES, 3DES,AES...)
  - Keine Übertragung von Passwörtern übers Netz
  - Erlaubt *Single Sign On* Implementierungen

| Begriff | Bedeutung |
|---|---|
| KDC | Key Distribution Center |
| AS | Authentication Server/Service |
| TGS | Ticket Granting Server/Service |
| Ticket Cache | Gespeicherte Tickets ermöglichen Single Sign On |
| Keytab | Gespeicherter Anwendungsschlüssel |
| Kerberized Applikation | Ein Programm welches Kerberos unterstützt |

# Kerberos MIT

- MIT
  - Referenzimplementierung
  - v5 für Linux und Windows erhältlich
  - weit verbreitet
  - wird in den USA entwickelt
  - Verschlüsselungsverfahren sind DES, 3DES, AES und RC4
  - Prüfsummenverfahren sind MD5, SHA-1, HMAC und CRC32
  - wird von vielen Anwendungen unterstützt

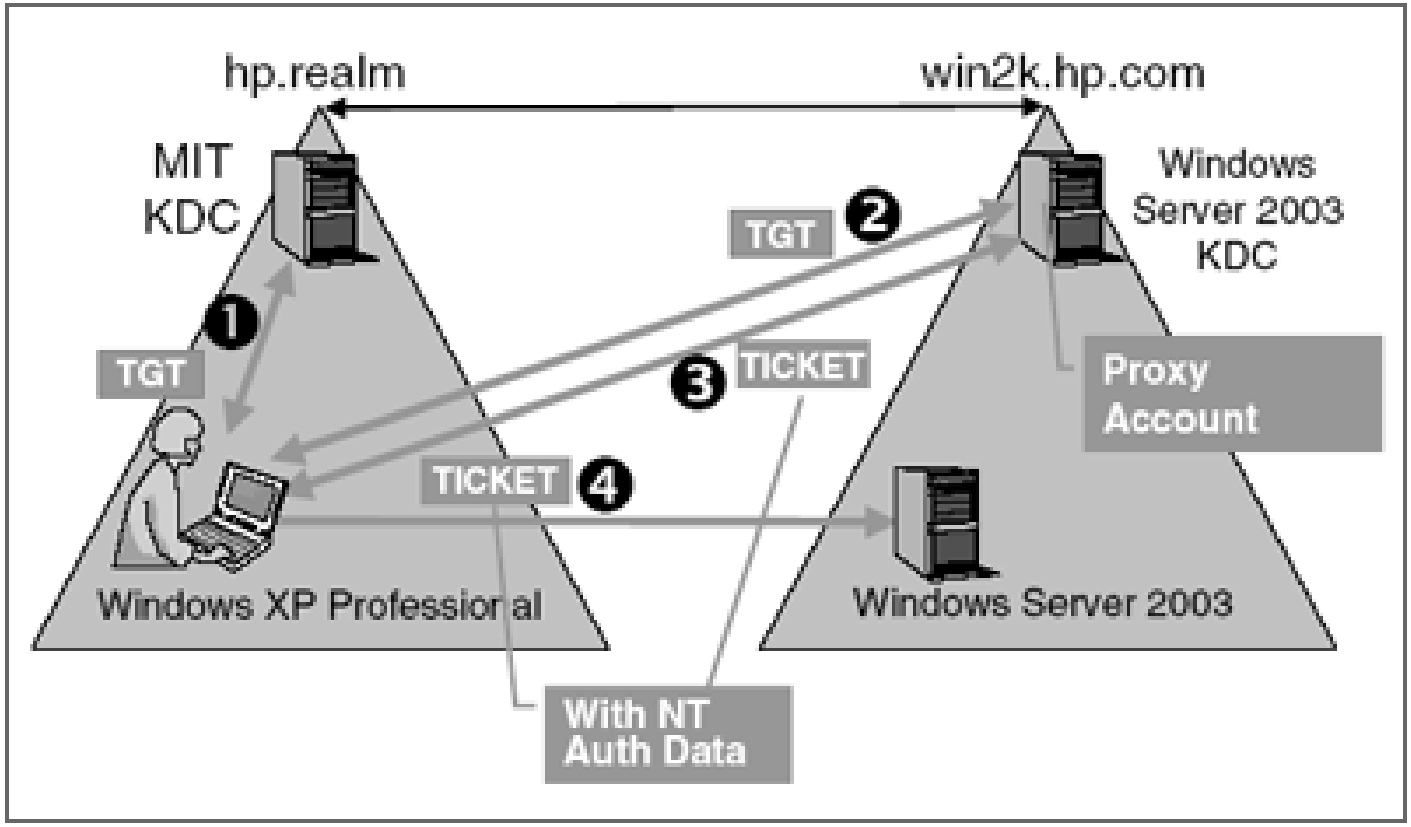
# Kerberos Heimdal

- Heimdal
  - neuere Implementierung
  - nur für Linux erhältlich
  - wird außerhalb der USA entwickelt
  - bessere Master/Slave Synchronisierung
  - Unterstützung von (3)DES, AES und RC4
  - Kerberos API und GSSAPI unterscheiden sich jedoch von den MIT APIs
  - Standard bei Samba 4
    - MIT Unterstützung ist in Entwicklung

# Kerberos Microsoft

- Microsoft
  - neueste Implementierung (seit Windows 2000)
  - unterstützt nur Kerberos 5
  - unterstützt RC4, DES, AES, aber kein 3DES
  - Unix Clients mit Windows KDC ist möglich
  - Windows Clients mit Unix KDC ist nicht ohne weiteres möglich. (benötigt eigenen Kerberos Client)
  - kein GSSAPI (dafür SSPI)

# Cross Realm Trust MS/Linux



# Kerberos Komponenten

# Kerberos Komponenten

Ressource

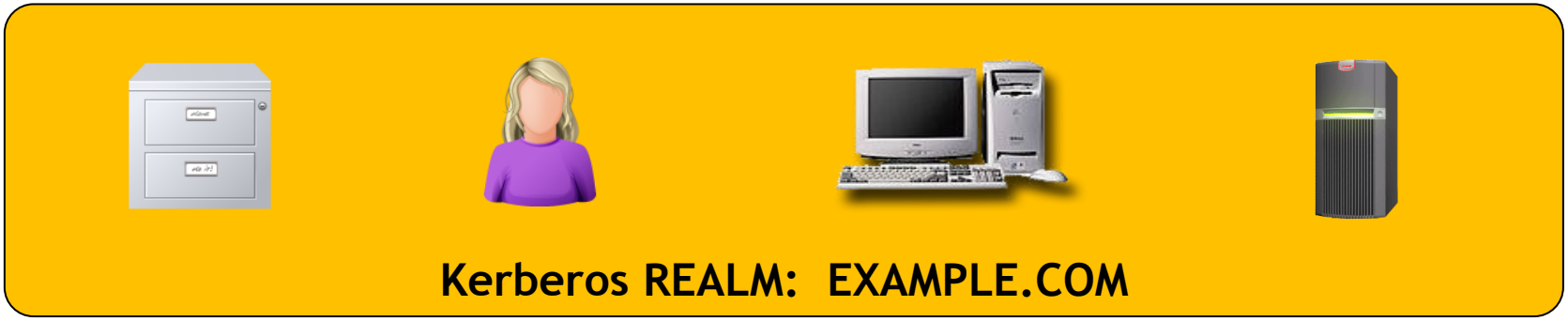
Kerberos Server

Maria

Maria's PC

Eine typische Kerberos Umgebung

# Kerberos Realm

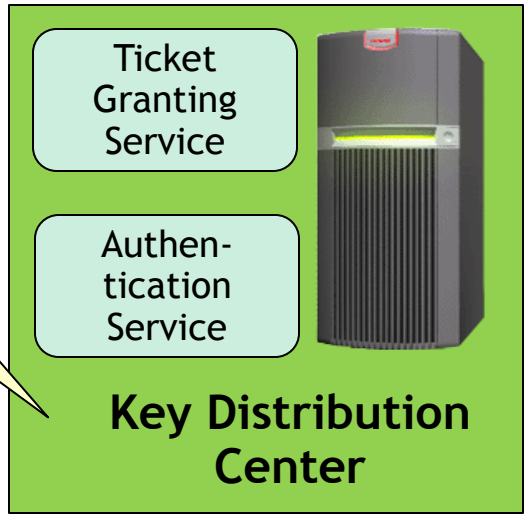



- Kerberos Realm
  - Ein administrativer Bereich zur Verwaltung von
    - Benutzerkennungen, Rechnern und Services
  - Eine Realm kann hierarchisch strukturiert sein
  - Jede REALM hat einen eindeutigen Namen, angelehnt an DNS Domainnamen wobei der Name der Realm in Großbuchstaben geschrieben wird
  - DNS: example.com -> REALM: EXAMPLE.COM

# Key Distribution Center

**Ressource**

Ein KDC
besteht aus 2 Services




**Maria**

**Maria's PC**

Ein Kerberos Server wird als Key Distribution Center bezeichnet

# Key Distribution Center

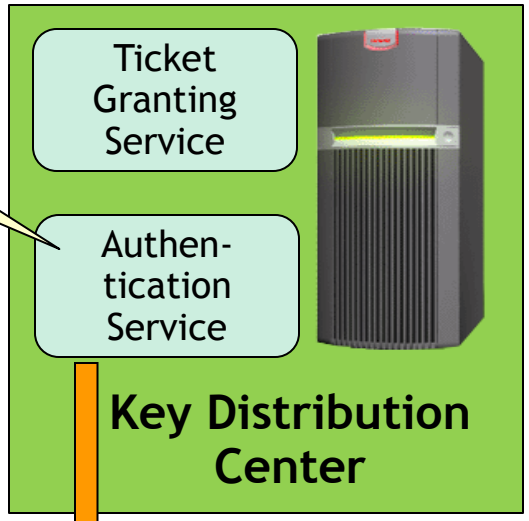
- Key Distribution Center (KDC)
  - Ein KDC besteht aus dem Authentication Server (AS) und dem Ticket Granting Server (TGS)
  - Der KDC muss besonders geschützt werden da es die Datenbank mit den User- und Serverprincipals der Realm enthält
  - Aus Redundanzgründen sind auch weitere KDC-Server (Slaves) möglich welche im RO-Modus betrieben werden
  - Die Slaves gleichen die Datenbank regelmäßig über ein Protokoll mit dem Master KDC ab.

# Key Distribution Center

Ressource

Ich überprüfe ob ein Benutzer/Client auch wirklich der ist für den er sich ausgibt

Ich kenne die folgenden Principals:
- Benutzer Maria
- PC Maria
- Ressource X




Maria

Maria's PC

Ein Authentication Service muss den Benutzer in (s)einer Datenbank finden

# Terminologie

- Principals
  - sind Teilnehmer des Kerberos Authentisierungssystems
  - Eindeutige Identifikation eines Benutzers oder Dienstes innerhalb einer REALM
  - Jeder Principal muss vor der Nutzung in der REALM-Datenbank angelegt werden
  - Besitzt eine ID und ein "Geheimnis"
    - "Geheimnis" ist nur dem Principal und dem KDC bekannt
  - Principals werden durch kadmin erzeugt und in der KDC Datenbank gespeichert

# Userprincipals

- Userprincipals
  - Die ID besteht aus einem Namen, einem @ und einer REALM
    - user@REALM -> jahn@EXAMPLE.COM
  - Als "Geheimnis" wird ein Passwort verwendet welches der Benutzer mit kpasswd ändern kann
  - BenutzerID kann eine Instance beinhalten
    - user/instance@realm -> jahn/admin@EXAMPLE.COM
    - Instanzen weisen Benutzer unterschiedliche Rollen zu
      - über Rollen können unterschiedliche Rechte definiert werden

# Ressource

Ressource

Repräsentiert eine Ressource welche eine Kerberos Authentifizierung benötigt

Ich kenne diese Ressource und habe einen passenden Service-Principal dafür


Maria

Maria's PC

Ressource kann sein: Web Server, FTP-Server, SSH-Server, Datei-Freigabe

# Serviceprincipals

- Serviceprincipals
  - Die ID besteht aus einer Serviceangabe, gefolgt von einem Slash '/' sowie dem vollqualifizierten Rechnernamen des Servers (FQDN)
  - Als Serviceangabe sind u.a. host, ftp oder z.B. pop erlaubt.
    - Syntax: service/dns.domain.name@REALM
      - host/sv1.example.com@EXAMPLE.COM
      - ftp/sv2.example.com@EXAMPLE.COM
  - Als "Geheimnis" wird ein Zufallswert verwendet welches auf dem Server in der Datei /etc/keytab hinterlegt wird
    - Der Schlüssel eines Servers kann nicht geändert, sondern lediglich neu erzeugt werden.

# Key Distribution Center

Ressource

Ich erstelle Tickets damit User/Clients auf Kerberos geschützte Ressourcen zugreifen können


Maria

Maria's PC

Ein spezielles Ticket wird pro geschützter Ressource benötigt

# Ticket

- Authentisierung von Benutzer/Dienst via Ticket
  - Ein Ticket wird vom Client beim KDC für einen bestimmten Dienst angefordert und in einem Ticketcache gespeichert
  - Gegenseitige Authentisierung von Benutzer und Dienst (Mutual Authentication)
  - Das Ticket wird mit dem Serviceschlüssel des Servers verschlüsselt und enthält:

# Ticket Cache

- Ticket Cache im Detail
  - Ein Ticket ist normalerweise 10-24 Stunden gültig und kann innerhalb dieser Zeit beliebig oft verwendet werden
  - Ausgestellt Tickets werden am Client gespeichert

- file based credential cache (MIT)
  - alle Tickets werden in einer Datei unter /tmp gespeichert
    - /tmp/krb5cc_UID
- memory based credential cache (Microsoft, Apple)
  - alle Tickets werden im Arbeitsspeicher hinterlegt und beim Abmelden gelöscht

# Ticket Granting Ticket

- Ticket Granting Ticket (TGT)
  - wird beim Authentication Server (AS) angefordert, wenn sich der Benutzer mit seinem Passwort authentisiert
  - ist ein Service-Ticket für den Ticket Granting Server (TGS)
    - Automatische Beantragung beim Anmelden oder manuell durch kinit
  - TGT hat ein default Lifetime von 8 Stunden (änderbar) und wird im Ticketcache des Client gespeichert
  - Für die Authentisierung bei Anforderung von Servicetickets am TGT
  - Der Benutzer muss sein Passwort nicht noch einmal eingeben
  - Beim Abmelden wird der Ticketcache automatisch gelöscht.

# Protokolle

- Generic Security Services API (GSSAPI):
  - generisches Interface zur Unterstützung von "strong Authentication", wie z.B. Kerberos wird oft von Services verwendet um Kerberos zu unterstützen
- Security Support Provider Interface (SSPI):
  - Microsoft Pendant zu GSSAPI
- Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO):
  - Übermittlung eines Kerberos-Tickets per HTTP

# Kerberos Vorraussetzungen

- Vorraussetzungen
  - Zeitsynchronisierte Systeme (max. 5 Minuten Differenz)
  - Client und Server müssen korrekt im DNS eingetragen sein
  - KDC's müssen speziell gesicherte Systeme sein
  - Mindestens zwei KDC's (Master, Slave).
  - Backupkonzept für REALM Datenbank

# Kerberos Kommunikation im Detail

# Ticket Granting Service

Ressource

Hallo AS, kann ich bitte ein Ticket haben damit ich damit Service Tickets beantragen kann? 3

Zuerst benötige ich aber ein Ticket vom Ticket Granting Service. 2

Ich würde gerne auf die Ressource zugreifen 1

Maria

Maria's PC

Ticket Granting Service

Authen-tication Service

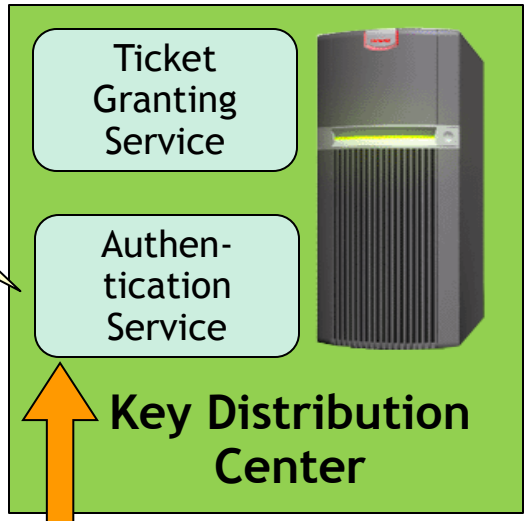
**Key Distribution Center**

Maria möchte auf eine Kerberos geschützte Ressource zugreifen

# Ticket Granting Ticket

Ressource

Okay, Ich habe das Kuvert mit deinem geheimen Passwort gesperrt. Wenn du es öffnen kannst findest du darin mein Ticket Granting Ticket (TGT)



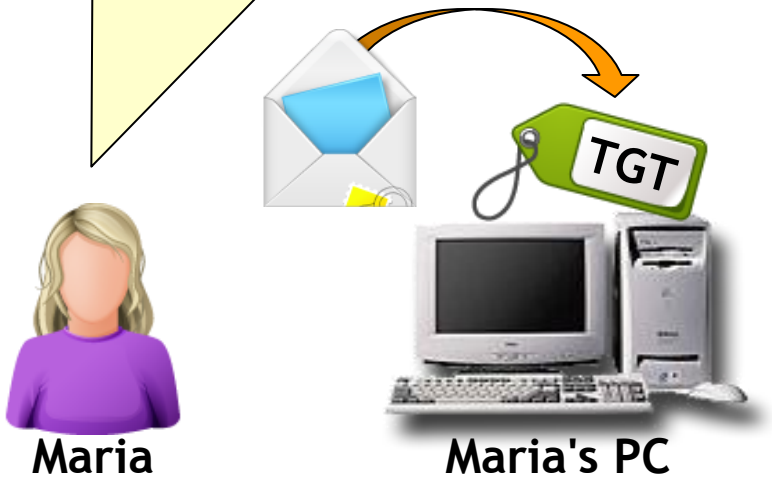

Maria

Maria's PC


Maria musste nicht ihr Passwort an das AS schicken. Das AS hat Zugriff auf die Datenbank und die Passwörter der User

# Ticket Granting Ticket

Super, ich konnte das Kuvert öffnen (Nachricht entschlüsseln) und bin jetzt Besitzer eines Ticket Granting Ticket (TGT)
Damit habe ich bewiesen dass ich Maria bin :-)




Das TGT wird benutzt um Service Tickets zu beantragen
Das TGT beinhaltet kein Passwort

# Ticket Granting Service

Ressource

Lass mich bitte der Ressource beweisen das ich Maria bin!
Hier ist eine Kopie meines TGT das beweist das ich wirklich Maria bin

TGT

Ticket Granting Service

Authen-tication Service

Key Distribution Center

Maria

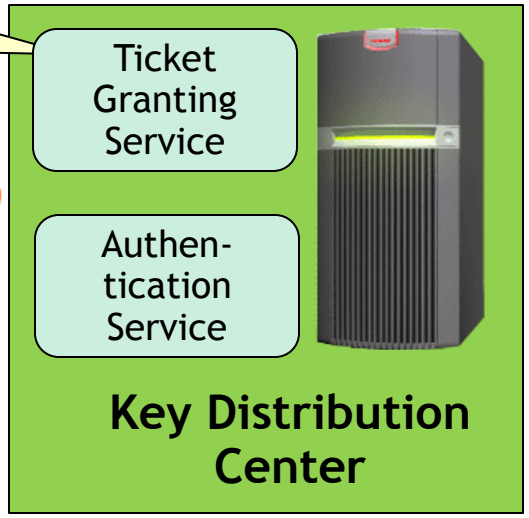

Maria's PC

Tickets für Ressourcen werden vom TGS ausgestellt

# Service Ticket

Ressource

Ja stimmt, du bist wirklich Maria!
Hier hast du ein bestätigtes TGS
für die gewünschte Ressource


Maria

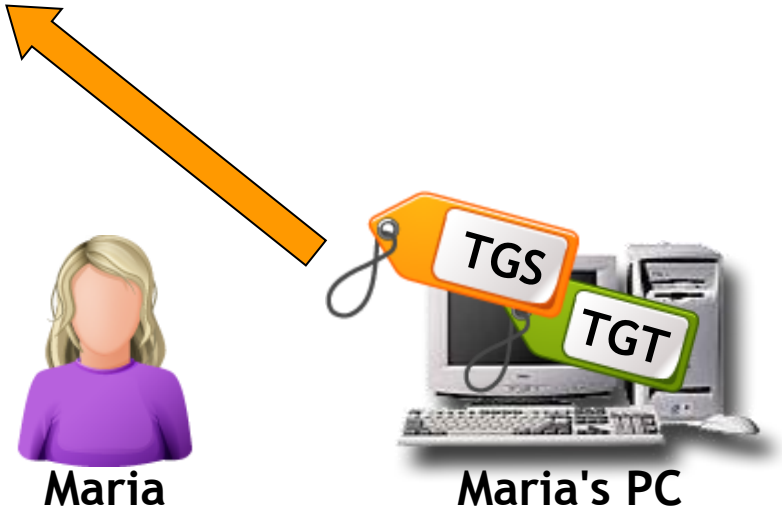

Maria's PC

TGS erstellt ein zeitlich beschränktes Service Ticket

# Zugriff auf die Ressource


Ressource

Hallo Ressource! Ich bin Maria und als Beweis bekommst du mein gültiges TGS


Maria

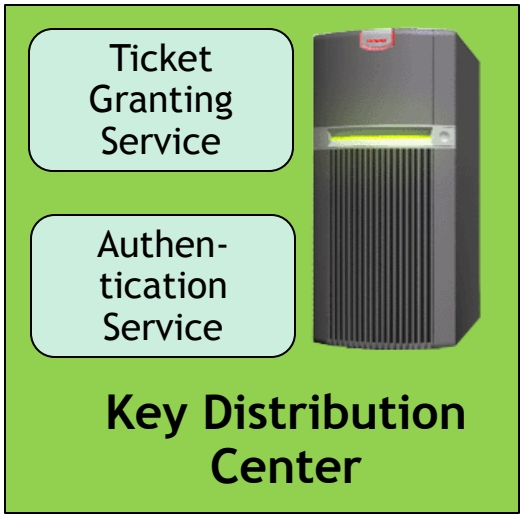
Maria's PC

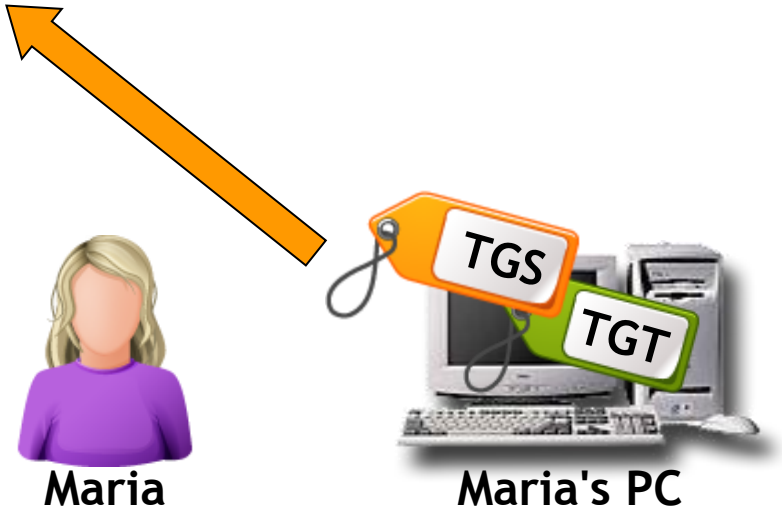
Maria möchte der Ressource beweisen, dass sie wirklich Maria ist

# Ressource Authorization


Ressource

Hallo Maria! Lass mich kurz überprüfen ob der Benutzer Maria überhaupt auf mich zugreifen darf!




Maria

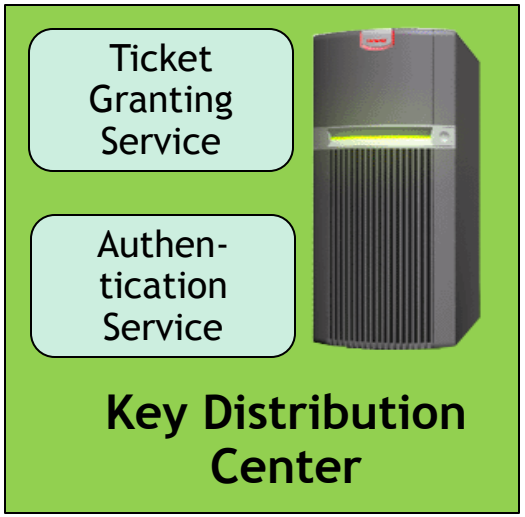
Maria's PC

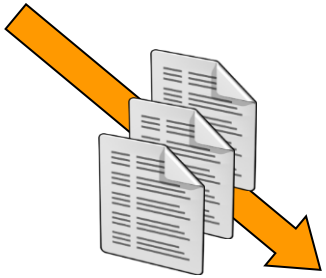
Nur weil Maria sich authentifiziert hat, bedeutet das aber nicht dass sie auch das Rechte hat auf die gewünschte Ressource zu zugreifen

# Ressourcen Freigabe

Ressource

Ja du darfst auf die Ressourcen zugreifen. Hier sind die gewünschten Daten



Maria


Maria's PC

Maria darf auf die Ressource zugreifen

# Erneuter Zugriff auf die Ressource


Ressource

Hallo ich bin es schon wieder, und ich möchte wieder auf die Ressource zugreifen. Anbei eine Kopie meines TGS für die Ressource

Nochmal auf die Ressource zugreifen

Maria


Maria's PC


Jedes Ticket hat ein Ablaufdatum. Solange es nicht abgelaufen ist kann es immer wieder verwendet werden

# Erneute Ressource Authorization


Ressource

Aha schon wieder Maria...mal schauen ob sie noch immer auf die Ressource zugreifen darf


Maria

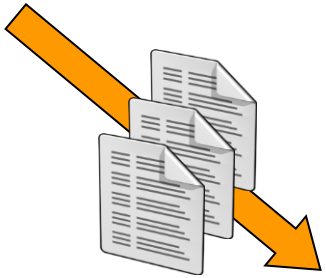

Maria's PC

In der Zwischenzeit könnte sich ja die Authorisierung verändert haben

# Ressourcen Freigabe

Ressource

Ja du darfst noch immer auf die Ressourcen zugreifen. Hier sind die gewünschten Daten



Maria


Maria's PC


Maria darf noch immer auf die Ressource zugreifen

# Vorbereiten der Konfigurationsumgebung

# virtuelle Maschine KDC01

- Infrastruktur
  - Betriebssystem: Debian 6.0.x
  - Nebenrolle: DNS-Server
  - Hostname: kdc01.example.com
  - IP-Adresse: 192.168.100.100 / 24
    - Konfigurationsdateien aus Übungsordner übernehmen

- Ubuntu Administration
  - Am besten Root User ein Kennwort geben und als Root arbeiten

# Tipp

- Netzwerkmanager entfernen
  - apt-get purge network-manager network-manager-gnome
- AppArmor Framework entfernen (Ubuntu)
  - AppArmor würde per Standardeinstellung nicht erlauben das Dienste wie nslcd und slapd Zugriff auf den Credential Cache unter /var/cache/krb5cc haben. Wenn AppArmor nicht benötigt wird dann ist die einfachste Lösung die Deinstallation
  - apt-get purge remove apparmor apparmor-utils libapparmor-perl libapparmor1

# Optionale Tätigkeiten

- Für die Administration sollten folgende Pakete installiert werden
  - apt-get install ssh vim nmap

- Netzwerkkarten in der VMWARE
  - eth0 ...Host only
  - eth1 ...Bridged

# Kurssetup: IP-Adresse

- Ändern der IP-Adressen Konfiguration
  - Wenn die IP-Adresse nicht mit dem physischen LAN übereinstimmt (z.B. Testumgebung) kann eine zweite Netzwerkkarte hinzugefügt werden die auf DHCP steht

```
# /etc/network/interfaces
...
auto eth0
iface eth0 inet static
        address 192.168.100.100
        netmask 255.255.255.0
        #gateway 192.168.100.1      #deaktivieren bei eth1+dhcp
auto eth1
iface eth1 inet dhcp
```

# DHCP Client Konfiguration

- DHCP Client Konfiguration
  - Da eth1 auf DHCP steht würde der DHCP Client immer die manuelle DNS-Client Konfiguration überschreiben
  - Lösung entfernen von domain-name & domain-name-servers

```
# /etc/dhcp/dhclient.conf
...
request subnet-mask, broadcast-address, time-offset, routers,
  domain-name, domain-name-servers, domain-search, host-
  name, netbios-name-servers, netbios-scope, interface-mtu,
  rfc3442-classless-static-routes, ntp-servers;
```

# Zeitsynchronsisation

- Zeit und Kerberos
  - für Kerberos ist es extrem wichtig das alle Server und Clients Zeitsynchron sind.
  - Auf physischen Servern ist NTP die erste Wahl
  - Auf virtuellen Servern ist es abhängig von der Virtualisierungs-lösung welche Variante am besten funktioniert
  - VMWARE: vmware-toolbox&

# NTP-Konfiguration KDC01

- apt-get install ntp  ntpdate
  - NTP Software Installation
- /etc/ntp.conf
  - server 0.pool.ntp.org  #Standardeinstellung bei Debian
- /etc/init.d/ntp stop
  - NTP Client stoppen
- ntpd -q -g  ODER  ntpdate 0.pool.ntp.org
  - Jetzt vom NTP Server die Zeit holen
- /etc/init.d/ntp start
  - NTP Client starten

# Hostname

- Ändern des Hostnamens
  - Der Name muss in den folgenden Dateien korrigiert werden
    - /etc/hostname
    - /etc/hosts

```
# /etc/hostname
kdc01
```

```
# /etc/hosts
127.0.0.1   localhost          #hier darf nicht der Hostname stehen
```

# DNS Server vorbereiten

- DNS Server installieren
  - apt-get update
  - apt-get install bind9
- DNS Konfiguration einrichten
  - Bind Konfigurationsdateien in /etc/bind/ via "gksu nautilus" oder ssh austauschen
    - 192.168.100, example, named.conf.{local,options}
    - Besitzer der neuen Dateien: root.bind

# DNS Server vorbereiten II

- DNS Name Forwarder für lokales Netz einrichten

```
# /etc/bind/named.conf.options
forwarders { 172.16.0.11;  }; #DNS Server des lokalen Netzes!
```

- DNS Client auf KDC01 konfigurieren

```
# /etc/resolv.conf
domain example.com
nameserver 192.168.100.100
```

# DNS Server starten

- DNS Server starten
  - /etc/init.d/bind9 start
    - /var/log/daemon.log ...auf Fehler prüfen
- Testen von DNS nach einem Reboot
  - hostname -f ...sollte 'kdc01.example.com' zeigen
  - ping www.linuxcampus.net
  - host kdc01.example.com
  - host 192.168.100.100

# Aufbau einer Kerberos Realm mit MIT Kerberos

# Aufbau einer MIT Realm

- Vorgehensweise
  - Nötige Software auf kdc01 installieren
  - Kerberos-Dienste stoppen
  - Eigene KDC-Konfiguration erzeugen
  - Initiale Principal-Datenbank erzeugen
  - Zusätzliche Principals anlegen
  - Kerberos-Dienste starten
  - Testen

# Kerberos Software installieren

- Kerberos Software installieren
  - apt-get install krb5-user krb5-doc krb5-kdc krb5-admin-server
    - Vorschläge des Wizards mit Enter bestätigen
- Dienste stoppen
  - /etc/init.d/krb5-kdc stop
  - /etc/init.d/krb5-admin-server stop

# Der Master Key

- Aufgabe des Masterkeys
  - In einer REALM werden die kryptografischen Schlüssel von allen Principals gespeichert und daher müssen sie vor Angreifern besonders geschützt werden.
  - Die KDC Datenbank ist durch Dateirechte vor Zugriff geschützt und nur die Prozesse krb5kdc und kadmin haben darauf Zugriff
  - Backup Administratoren oder der Diebstahl der Festplatte könnten dazu führen das jemand Zugriff auf alle Daten bekommen
  - Darum wird der Masterkey der Datenbank noch zusätzlich durch ein Master Keywort geschützt

# Master Passwort

- Master Passwort
  - Das Master Passwort verhindert das Angreifer den Inhalt der Datenbank lesen können
  - Nachteil des Passwortes ist aber das auch die Services beim starten das Passwort benötigen und ein interaktives Starten der Dienste nicht mehr möglich
- Stash Datei
  - Das Master Kennwort kann auch in einer Datei abgespeichert werden und Dienste könnten beim Start darauf zugreifen
  - Natürlich wird dabei aber eine wichtige Sicherheitsfunktion deaktiviert

# KDC Konfiguration

- KDC-Konfiguration
  - Das KDC wird in der Datei /etc/krb5kdc/kdc.conf konfiguriert
  - Diese Datei existiert bereits durch die Installation
  - Öffnen Sie diese Datei in einem Editor und ersetzen Sie den kompletten Inhalt mit der Konfiguration von der nächsten Seite

# Anpassen der KDC.CONF Konfiguration

```
[kdcdefaults]
   kdc_ports = 88
   kdc_tcp_ports = 88
   v4_mode = disable
[realms]
   EXAMPLE.COM = {
   database_name = /var/lib/krb5kdc/principal
   acl_file = /etc/krb5kdc/kadm5.acl
   key_stash_file = /etc/krb5kdc/stash
   max_life = 10h 0m 0s
   max_renewable_life = 7d 0h 0m 0s
   master_key_type = aes256-cts
   supported_enctypes = aes256-cts:normal arcfour-hmac:normal des3-hmac-sha1:normal
   default_principal_flags = +preauth
}
[logging]
   kdc = SYSLOG:INFO:AUTH
   admin_server = SYSLOG:INFO:AUTH
```

# Struktur der kdc.conf

```
[kdcdefaults]
  Default-Parameter-1 = Wert-1
  ...
[realms]
  REALM_A = {
     Realm_A_Parameter-1 = Wert-1
     ...
  }
 REALM_B = {
     Realm_B_Parameter-1 = Wert-1
     ...
  }
[logging]
  kdc = Log-Datei
  admin_server = Log-Datei
```

Standardparameter für alle Realms

Realm Definition(en) und Realm spezifische Parameter

Log Konfiguration der einzelnen KDC Dienste

# Abschnitt: kdcdefaults

```
# /etc/krb5kdc/kdc.conf
[kdcdefaults]
    kdc_ports = 88
    kdc_tcp_ports = 88
    v4_mode = disable
```

- kdc_ports
  - auf welchen UDP Ports der KDC Prozess hören soll
    - Kerberos v5 = 88, Kerberos v4 = 750
- kdc_tcp_ports
  - zusätzlich werden auch TCP Anfragen aktiviert (optional)
- v4_mode
  - keine Kerberos v4 Unterstützung aktivieren

# Abschnitt: realms I

```
# /etc/krb5kdc/kdc.conf
[realms]
  EXAMPLE.COM = { ... }
```

- database_name
  - Kryptografischen Langzeitschlüssel aller Principals
- acl_file
  - Ablageort der Access Control Lists
- supported_enctypes
  - Eine List von Verschlüsselungstypen und Salts welche für jeden neuen Principal erstellt werden
    - verschlüsselungs_typ:salt_typ

# Abschnitt: realms II

```
# /etc/krb5kdc/kdc.conf
[realms]
  EXAMPLE.COM = { ... }
```

- max_life
  - Maximale Lebensdauer für Tickets die der KDC erstellt
- max_renewable_life
  - Maximale Zeitspanne innerhalb der Tickets erneuert werden können
- key_stash_file
  - Definiert ob der Masterkey aus einer Stash Datei gelesen werden soll

# Abschnitt: realms III

```
# /etc/krb5kdc/kdc.conf
[realms]
  EXAMPLE.COM = { ... }
```

- default_principal_flags
  - definiert welche Standardflags bei jedem neu erstellten Principal konfiguriert werden
    - Als Wert kann mein eine kommaseparierte Liste von Flags angeben, die jeweils durch ein + oder - angeführt werden
    - +preauth sorgt dafür, dass alle Principals eine Pre-Authentication machen müssen bevor sie ein Service Ticket bekommen

# Abschnitt: Logging

```
# /etc/krb5kdc/kdc.conf
[logging]
  kdc = SYSLOG:INFO:AUTH
  kdc = FILE:/var/log/kdc.log
  admin_server = SYSLOG:INFO:AUTH
  admin_server = CONSOLE
```

- [logging]
  - bestimmt welche Logging Arten verwendet werden sollen
    - FILE= das Log wird bei jedem Start überschrieben
    - FILE: Inhalte werden angehängt
    - CONSOLE Ausgabe erfolgt auf der Konsole des KDC
    - SYSLOG:severity:facility

# KDC Datenbank erstellen

- kdb5_util
  - Mit diesem Tool können KDC Datenbanken erstellt werden

| Optionen | Bedeutung |
|---|---|
| create | Eine neue KDC Datenbank erstellen |
| destroy | Löschen einer KDC Datenbank |
| create -s | zusätzlich zum Create eine Stash Datei erstellen |
| stash | nachträglich eine Stash Datei erstellen |
| -r REALM | Speziellen REALM Namen definieren |
| dump | KDC Datenbank in Textfile speichern, wichtig für Backup und Replikation |
| load | Datenbank aus einem Textfile wiederherstellen |

# Datenbank erstellen

- KDC Datenbank initialisieren
  - kdb5_util create -s -r EXAMPLE.COM
- Erklärung
  - Dieses Kommando erzeugt eine neue Datenbank in /var/lib/krb5kdc/principal
  - Das abgefragte Master Passwort das die KDC Datenbank schützt, wird durch die Option -s in einer Stash Datei /etc/krb5kdc/stash abgelegt

# Datenbank erstellen

```
# KDC Datenbank initialisieren
root@kdc01:~#  kdb5_util create -s -r EXAMPLE.COM
Loading random data
Initializing database '/var/lib/krb5kdc/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
        You will be prompted for the database Master Password.
        It is important that you NOT FORGET this password.
        Enter KDC database master key:
        Re-enter KDC database master key to verify:
root@kdc01:~#
```

# Datenbank verwalten

- kadmin und kadmin.local
  - Mit diesen Tools können KDC Datenbanken verwaltet werden
  - kadmin.local funktioniert nur auf dem KDC und verlangt beim Aufruf kein Passwort
    - funktioniert auch wenn der Dienst nicht gestartet ist

| Optionen | Bedeutung |
|---|---|
| -r REALM | Definiert die Standard REALM |
| -p principal | Spezieller User für Verbindungsaufbau |
| -k | Keytab anstatt Kennwort verwenden |
| -k -t keytab | Spezielle Keytab definieren |
| -m | Keine Keytab verwenden und anstatt nach einem PW fragen |
| -s server[:Port] | zu speziellen Kerberos Admin Server verbinden |

# Initiale Datenbank prüfen

- kadmin.local -r EXAMPLE.COM
  - Verbindung zur Realm aufbauen
- listprincs
  - Alle vordefinierten Principals auflisten

```
# KDC Datenbank Administrationstool starten
root@kdc01:~#  kadmin.local  -r EXAMPLE.COM
Authenticating as principal root/admin@EXAMPLE.COM with password.
kadmin.local: # Anzeigen der definierten Principals
kadmin.local: listprincs
K/M@EXAMPLE.COM
kadmin/admin@EXAMPLE.COM
kadmin/changepw@EXAMPLE.COM
kadmin/history@EXAMPLE.COM
kadmin/kdc01@EXAMPLE.COM
krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

# Wichtige Principals

- krbtgt/EXAMPLE.COM@EXAMPLE.COM
  - Das Ticket Granting Service (TGS)
- kadmin/kdc01.example.com@EXAMPLE.COM
  - Principal für die Authentifizierung des kadmin Dienstes
- kadmin/kchangepw@EXAMPLE.COM
  - Principal für Passwortänderungen des kpasswd Dienstes
- K/M@EXAMPLE.COM
  - Dieser Principal enthält den Master Key
- kadmin/history@EXAMPLE.COM
  - Gehört zum kadmin Dienst und wird für die interne Verschlüsselung und Passwort History verwendet

# Befehle in kadmin

| Optionen | Bedeutung |
|---|---|
| ? | Auflisten der möglichen Befehle |
| addprinc | Principal erstellen |
| delprinc | Principal löschen |
| listprincs | Principal anzeigen |
| modprinc | Principal modifizieren |
| addpol | Policy erstellen |
| delpol | Policy löschen |
| listpols | Alle Policies anzeigen |
| cpw | Passwort ändern |
| ktadd | Einträge zur Keytab hinzufügen |
| ktremove | Einträge aus der Keytab löschen |
| ... | |

# Aufgabe: Erstellen von 2 Principals

- addprinc user@EXAMPLE.COM
  - Ein normaler Principal für den Benutzer "user"
  - kann verwendet werden um Tickets anzufragen
- addprinc user/admin@EXAMPLE.COM
  - Ein User mit Administratorrechten um die Datenbank verwalten zu dürfen. z.B erstellen von neuen Principals

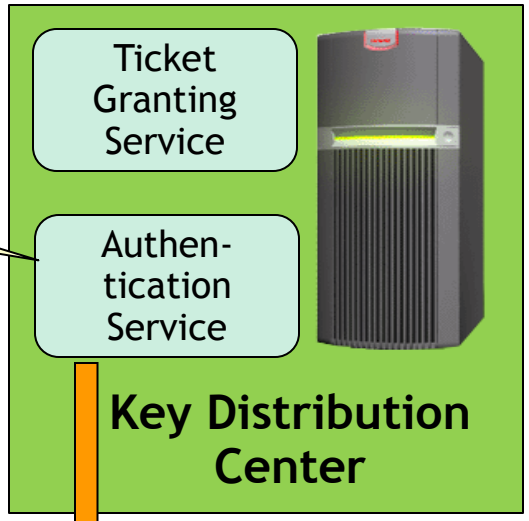
```
# Principal erstellen
kadmin.local:  addprinc user/admin@EXAMPLE.COM
WARNING: no policy specified for user/admin@EXAMPLE.COM; defaulting to no p.
Enter password for principal "user/admin@EXAMPLE.COM":
Re-enter password for principal "user/admin@EXAMPLE.COM":
Principal "user/admin@EXAMPLE.COM" created.
```

# Principals in der Kerberos Realm

Ich kenne jetzt die folgenden User-Principals:
- user@EXAMPLE.COM
- user/admin@EXAMPLE.COM

user

user/admin




```
# Anzeigen der bekannten Principals
root@kdc01:~# kadmin.local
kadmin.local:  listprincs
user@EXAMPLE.COM
user/admin@EXAMPLE.COM
```

Ein Authentication Service muss alle Benutzer in (s)einer Datenbank finden

# Starten des KDC

- Starten des KDC als Dienst
  - Damit der KDC-Dienst beim starten weiß für welchen Realm er zuständig ist wird der Dienst um den Parameter -r REALM erweitert

```
# /etc/default/krb5-kdc
DAEMON_ARGS="-r EXAMPLE.COM"
```

  - /etc/init.d/krb5-kdc start
    - nmap localhost sollte Port 88 zeigen

# Client Konfiguration

- Kerberos Client Konfiguration
  - Die Kerberos Konfiguration eines Clients erfolgt in der Datei /etc/krb5.conf welche auf allen Rechnern gleich ist
  - Wenn die Datei die richtigen Einträge beinhaltet können alle Kerberos Tools ihre Standardwerte wie REALM, Administrations Server, etc. aus dieser Datei entnehmen
  - Wenn die Datei nicht existiert oder falsche Informationen enthält müssen die einzelnen Kerberos Werkzeuge mit detaillierten Informationen gestartet werden

# Client Konfiguration /etc/krb5.conf

```
[libdefaults]
  default_realm = EXAMPLE.COM

[realms]
  EXAMPLE.COM = {
  kdc = kdc01.example.com
  admin_server = kdc01.example.com
}

[domain_realm]
  example.com = EXAMPLE.COM
  .example.com = EXAMPLE.COM
  ads.example.com = ADS.EXAMPLE.COM     # Vorbereitung für
 .ads.example.com = ADS.EXAMPLE.COM     # spätere Übungen
```

# Aufgabe: Erstellen eines Service P.

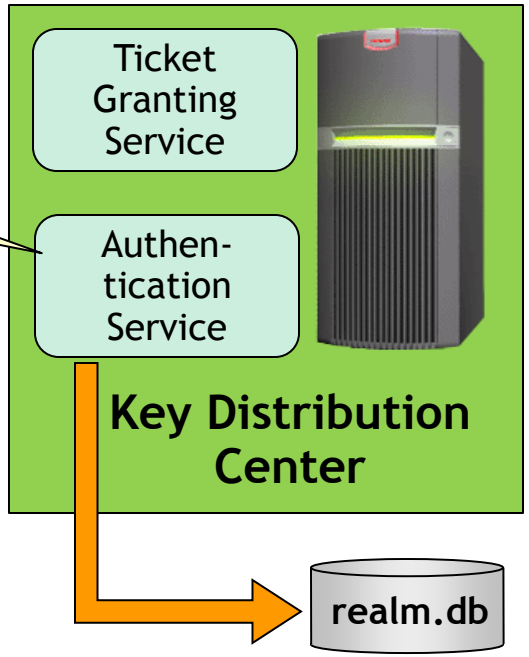
- addprinc -randkey host/kdc01.example.com@EXAMPLE.COM
  - Ein Service Principal ist gleichwertig mit einem User Principal mit dem Unterschied das sie ihr Passwort nicht kennen müssen da sie direkt mit den Schlüsseln arbeiten können
  - -randkey ...erzeugt automatisch ein sehr gutes Passwort

```
# Verbindung aufbauen. -r EXAMPLE.COM ist nicht mehr notwendig
root@kdc01:~# kadmin.local
Authenticating as principal root/admin@EXAMPLE.COM with password.
# Service Principal erstellen
kadmin.local:  addprinc -randkey  host/kdc01.example.com@EXAMPLE.COM
WARNING: no policy specified for host/kdc01.example.com@EXAMPLE.COM; ...
Principal "host/kdc01.example.com@EXAMPLE.COM" created.
kadmin.local:  listprincs
host/kdc01.example.com@EXAMPLE.COM
```

# Principals in der Kerberos Realm

Ich kenne jetzt die folgenden Service-Principals:
- host/kdc01.example.com@EXAMPLE.COM

**Ressource**
**host/kdc01**




```
# Anzeigen der bekannten Principals
root@kdc01:~# kadmin.local
kadmin.local:  listprincs
host/kdc01.example.com@EXAMPLE.COM
```

Ein Authentication Service muss alle Services in (s)einer Datenbank finden

# Starten und Testen

- KDC starten
  - /etc/init.d/krb5-kdc start
  - Alternativ:  service krb5-kdc start
- Tickets beziehen
  - kinit  user@EXAMPLE.COM
  - kvno  host/kdc01.example.com@EXAMPLE.COM
- Ticketcache überprüfen
  - klist

# User Tickets beziehen

user

kinit: Hallo ich heisse user@example.com
(ohne Passwort)
1

Verschlüsseltes TGT
2

Ticket Granting Service

Authen-tication Service

**Key Distribution Center**

3

Mithilfe des Passwortes kann der User das Kuvert öffnen und das TGT im lokalen Ticket Cache abspeichern

TGT

**user's PC**

```
root@kdc01:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: user@EXAMPLE.COM

...    Service principal
...    krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

# User Ticket beziehen

- Ticket für ein User Principal beziehen
  - Das MIT-Kommando kinit dient dazu den Clients mit Tickets zu versorgen. Der Benutzer benötigt dazu den Principal Name und dessen Passwort
  - Das Passwort wird dabei nie über das Netz geschickt

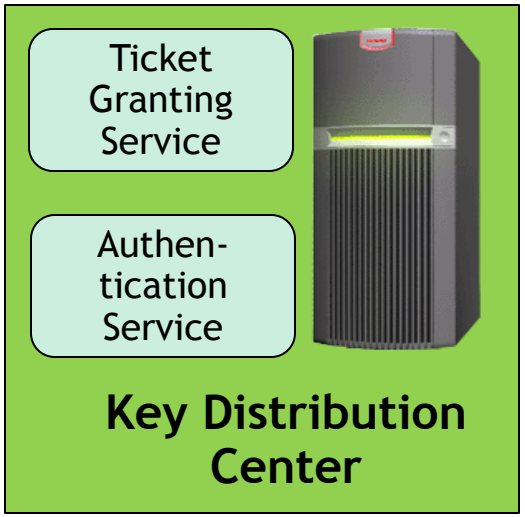
```
root@kdc01:~#  kinit user@EXAMPLE.COM
Password for user@EXAMPLE.COM:
root@kdc01:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: user@EXAMPLE.COM

Valid starting     Expires            Service principal
10/02/11 13:51:02  10/02/11 23:51:02  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 10/03/11 13:50:58
```

# Ticket für ein Service beziehen

user

1 kvno: Hallo hier ist mein TGT, bitte gib mir ein Ticket für die Ressource-X (host/kdc01.example.com)

TGT




TGS

Hier ist das TGS für Ressource-X 2

3

TGS

Das TGS für die Ressource-X wird im lokalen Ticket Cache gespeichert

user's PC

TGT TGS

```
root@kdc01:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: user@EXAMPLE.COM

...   Service principal
...   krbtgt/EXAMPLE.COM@EXAMPLE.COM
...   host/kdc01.example.com@EXAMPLE.COM
```

# Service Ticket beziehen

- Ticket für ein Service Principal beziehen
  - Befindet sich bereits ein TGT im Ticket Cache kann mit kvno ein Ticket für eine kerberisierte Anwendung geholt werden
  - dient nur zum testen, passiert normalerweise automatisch

```
root@kdc01:~#  kvno host/kdc01.example.com@EXAMPLE.COM
host/kdc01.example.com@EXAMPLE.COM: kvno = 1
root@kdc01:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: user@EXAMPLE.COM

Valid starting     Expires            Service principal
10/02/11 13:51:02  10/02/11 23:51:02 krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 10/03/11 13:50:58
10/02/11 13:51:18  10/02/11 23:51:02 host/kdc01.example.com@EXAMPLE.COM
        renew until 10/03/11 13:50:58
```

# Ticket Cache anzeigen

- Ticket Cache Anzeigen
  - Mit klist läßt sich der Inhalt des Ticket Cache anzeigen
    - -f ...zusätzlich die Ticket Flags anzeigen
    - -e ...zusätzlich die Verschlüsselungstypen anzeigen

```
root@kdc01:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: user@EXAMPLE.COM

Valid starting     Expires            Service principal
10/02/11 13:51:02  10/02/11 23:51:02 krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 10/03/11 13:50:58
10/02/11 13:51:18  10/02/11 23:51:02 host/kdc01.example.com@EXAMPLE.COM
        renew until 10/03/11 13:50:58
```

# Ticket Flags

```
root@kdc01:~# klist -f
Valid starting     Expires          Service principal
10/02/11 13:51:02  10/02/11 23:51:02  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 10/03/11 13:50:58, Flags: RIA
10/02/11 13:51:18  10/02/11 23:51:02  host/kdc01.example.com@EXAMPLE.COM
        renew until 10/03/11 13:50:58, Flags: RAT
```

| Flag | Bedeutung | Flag | Bedeutung |
|---|---|---|---|
| R | Renewable | f | Forwarded |
| A | Allow-Postdate | P | Proxiable |
| d | Postdated | p | Proxy |
| i | Invalid | 0 | Ok-As-Delegate |
| F | Forwardable | I | Inital |
| A | Pre-Authenticated | T | Transited Policy Checked |
| H | HW-Authent | | |

# Löschen des Ticket Cache Inhaltes

- kdestroy
  - Damit können alle im Cache befindlichen Tickets gelöscht werden. Idealerweise wird der Befehl beim abmelden des User automatisch ausgeführt z.b via `~/.bash_logout`

```
root@kdc01:~# klist
Valid starting     Expires          Service principal
10/02/11 13:51:02  10/02/11 23:51:02 krbtgt/EXAMPLE.COM@EXAMPLE.COM
      renew until 10/03/11 13:50:58
10/02/11 13:51:18  10/02/11 23:51:02 host/kdc01.example.com@EXAMPLE.COM
      renew until 10/03/11 13:50:58
# Löschen des Ticket Cache Inhaltes
root@kdc01:~# kdestroy
root@kdc01:~# klist
```

# Troubleshooting

```
root@kdc01:~#  kinit user@EXAMPLE.COM
kinit: Cannot resolve network address for KDC in realm "EXAMPLE.COM"
while getting initial credentials
```

- Error: Cannot resolve network address for KDC
  - Ursache 1: falscher Eintrag in /etc/krb5.conf
  - Ursache 2: Namensauflösung funktioniert nicht
    - Läuft der DNS-Server?
    - Verwendet der Client den DNS-Server?
    - Kann der KDC-Name mit host aufgelöst werden?
      - host kdc01.example.com
      - host kdc01
      - host 192.168.100.100

# Kerberos Remote Administration

# Kerberos Administration

- kadmin.local
  - Der Nachteil dieser Verwaltung ist das man am KDC als Root angemeldet sein muss

- kadmin
  - Mit diesem Tool kann man remote auf einen Kerberos Administrationsserver zugreifen
  - Es können mehrere Administratoren mit unterschiedlichen Rechten definiert werden
    - admin1: Nur Passwort Verwaltung
    - admin2: Principal Verwaltung

# kadmin

- kadmin
  - entspricht dem kadmin.local mit dem Unterschied das der Zugriff bereits über das Kadmin-Protokoll erfolgt
  - Der Netzwerkverkehr wird über Kerberos authentisiert und gesichert also unsere erster kerberosierter Dienst

| Optionen | Bedeutung |
|---|---|
| -r REALM | Definiert die Standard REALM |
| -p principal | Spezieller User für Verbindungsaufbau |
| -k | Keytab anstatt Kennwort verwenden |
| -k -t keytab | Spezielle Keytab definieren |
| -m | Keine Keytab verwenden und anstatt nach PW fragen |
| -s server[:Port] | zu speziellen Kerberos Admin Server verbinden |

# kadmin

- Kadmin Protokoll
  - ist MIT-spezifisch und nicht kompatibel zu Heimdal oder Active Directory
- Principal Name
  - als kerborisierter Dienst besitzt Kadmin auch einen eigenen Principal und eigene kryptografische Schlüssel welche bereits bei der Installation erzeugt werden
    - kadmin/kdc01.example.com@EXAMPLE.COM
- Keytab Datei
  - kadmin hat eine Sonderstellung und benötigt keine keytab

# Admin Server Konfiguration in kdc.conf

```
[kdcdefaults]
  admin_server = kdc01.example.com:749
[realms]
  EXAMPLE.COM = {
    acl_file = /etc/krb5kdc/kadm5.acl
  }
[logging]
  admin_server = SYSLOG:INFO:AUTH
```

- admin_server
  - bestimmt den Administration Server (Optional da default)
- acl_file
  - Zugriffsrechte für die KDC Datenbank

# /etc/krb5kdc/kadm5.acl

```
Principal  Zugriffsmaske  [optionales Zugriffsziel]
-------------------------------------------------------
*/admin@ATHENA.MIT.EDU  *
joeadmin@ATHENA.MIT.EDU ADMCIL
joeadmin/*@ATHENA.MIT.EDU il  */root@ATHENA.MIT.EDU
*@ATHENA.MIT.EDU cil  *1/admin@ATHENA.MIT.EDU
```

- ACL Konfiguration
  - Über Access Control Lists kann gesteuert werden welcher Principal welche Rechte auf dem KDC hat
  - Die Reihenfolge der Einträge ist wichtig da nicht weiter geprüft wird, wenn ein passender Eintrag gefunden wurde

http://web.mit.edu/kerberos/krb5-1.5/krb5-1.5.4/doc/krb5-install/Add-Administrators-to-the-Acl-File.html

# ACL-Zugriffsmaske

| Recht | | Bedeutung |
|---|---|---|
| a | *add* | Hinzufügen von Objekten gestatten |
| A | *add* | - " - verbieten |
| d | *delete* | Löschen von Objekten gestatten |
| D | *delete* | - " - verbieten |
| m | *modify* | Änderungen an Objekten gestatten |
| M | *modify* | - " - verbieten |
| c | *change* | Passwortänderungen gestatten |
| C | *change* | - " - verbieten |
| i | *inquiry* | Anfragen über Objektdetails gestatten |
| I | *inquiry* | - " - verbieten |
| l | *list* | Auflisten der Objekte in der KDC Datenbank gestatten |
| L | *list* | - " - verbieten |
| x | | Kurzform für admcil |
| * | | Gleiche Bedeutung wie x |

# Admin Berechtigungen setzen

**Principal Zugriffsmaske [optionales Zugriffsziel]**

- Vergeben von Administrator Rechten
  - Alle Benutzer deren zweite Komponente admin lautet sollen volle Administrationsrechte erhalten
  - Mit dieser Konfiguration machen wir aus einem normalen Benutzer einen Benutzer mit speziellen Administrations-rechten in der Realm
  - Nur notwendig für den Zugriff über kadmin

```
# /etc/krb5kdc/kadm5.acl
*/admin@EXAMPLE.COM  *
```

user/admin

# Starten des kadmin Dienstes

- Starten des kadmin als Dienst
  - Damit der kadmin-Dienst beim starten weiß für welchen Realm er zuständig ist wird der Dienst um den Parameter -r REALM erweitert

```
# /etc/default/krb5-admin-server
RUN_KADMIND=true
DAEMON_ARGS="-r EXAMPLE.COM"
```

  - /etc/init.d/krb5-admin-server start
    - nmap localhost sollte zusätzlich Port 749 zeigen
  - kadmin -p user/admin@EXAMPLE.COM

# Troubleshooting

- Probleme mit kadmin
  - falls zwar der Verbindungsaufbau klappt aber danach keine Rechte wirken gibt können mehrere Ursachen verantwortlich sein
    - fehlende ACL-Datei auf kdc01
    - falscher Dateiname für die ACL-Datei in kdc.conf
    - falsche Rechte in der ACL-Datei

```
root@kdc01:~#  kadmin -p user/admin@EXAMPLE.COM
Authenticating as principal user/admin@EXAMPLE.COM with password.
Password for user/admin@EXAMPLE.COM:
kadmin:  listprincs
get_principals: Operation requires ``list'' privilege while retrieving list.
```

# Linux Host lx01 via Kerberos einbinden

# virtuelle Maschine lx01

- Infrastruktur
  - Betriebssystem: SUSE Linux Enterprise Server 11 SP1
  - Hostname: lx01.example.com
  - IP-Adresse: 192.168.100.101 / 24
  - DNS-Server: 192.168.100.100 (=KDC01)

  - Netzwerkkarte in VMWARE auf Host Only stellen damit einen Kommunikation mit dem KDC01 möglich ist

# Vorbereitung lx01 SUSE

- Ändern der IP-Adresse, DNS-Server
  - am besten via Yast
    - IP-Adresse: 192.168.100.101 / 24
    - DNS-Server: 192.168.100.100 (=KDC01)

- Ändern des Hostnamens
  - Der Name muss in den folgenden Dateien korrigiert werden
    - /etc/HOSTNAME
    - /etc/hosts

# Vorbereiten von lx01

- Debian/Ubuntu Kerberos Client Software installieren
  - apt-get install krb5-user  krb5-doc
- SUSE Kerberos Client Software installieren
  - zypper install  krb5  krb5-client  krb5-doc
- Kerberos Client Konfiguration
  - Die Kerberos Konfigurationsdatei /etc/krb5.conf von KDC01 übernehmen

# Vorbereitung auf lx01

- Zeitsynchronisation
  - zwischen dem Client und dem Server ist extrem wichtig
- /etc/ntp.conf
  - server kdc01.example.com
- rcntp ntptimeset
- rcntp start
- chkconfig -a ntp
- ntpq -p kdc01 lx01
  - Offset sollte bei beiden Servern annährend gleich sein

# kadmin remote von lx01 verwenden

- Verbindung zu Admin Server herstellen
  - kadmin -p  user/admin@EXAMPLE.COM
- Neuen Service Principal erstellen
  - addprinc -randkey host/lx01.example.com@EXAMPLE.COM

```
root@kdc01:~#  kadmin -p user/admin@EXAMPLE.COM
Authenticating as principal user/admin@EXAMPLE.COM with password.
Password for user/admin@EXAMPLE.COM:
kadmin:  addprinc -randkey host/lx01.example.com@EXAMPLE.COM
WARNING: no policy specified for host/lx01.example.com@EXAMPLE.COM; ...
Principal "host/lx01.example.com@EXAMPLE.COM" created.
```

# Troubleshooting

- Probleme mit kadmin
  - falls zwar der Verbindungsaufbau klappt aber danach keine Rechte wirken gibt können mehrere Ursachen verantwortlich sein
    - fehlende ACL-Datei auf kdc01
    - falscher Dateiname für ACL-Datei in kdc.conf
    - falsche Rechte in ACL-Datei

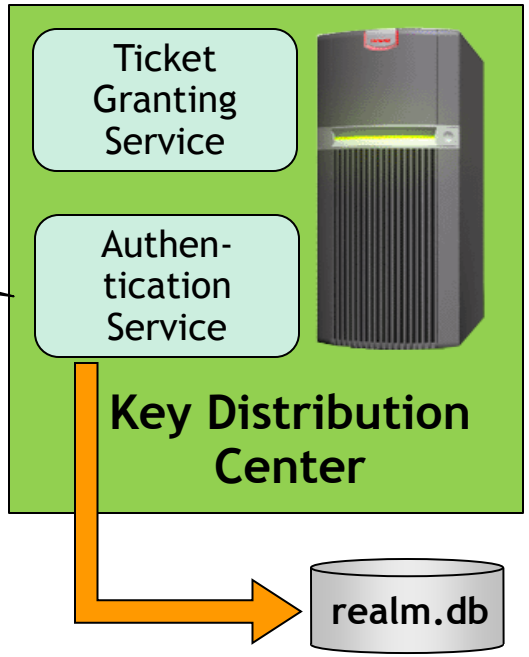
```
root@lx01:~#  kadmin -p user/admin@EXAMPLE.COM
Authenticating as principal user/admin@EXAMPLE.COM with password.
Password for user/admin@EXAMPLE.COM:
kadmin:  addprinc -randkey host/lx01.example.com@EXAMPLE.COM
WARNING: no policy specified for host/lx01.example.com@EXAMPLE.COM; ...
add_principal: Operation requires ``add" privilege while creating "host/
lx01.example.com@EXAMPLE.COM".
```

# Key Distribution Center

Ich kenne jetzt die folgenden Service Principals:
- host/kdc01.example.com@EXAMPLE.COM
- host/lx01.example.com@EXAMPLE.COM

**Ressource host/kdc01**

**Ressource host/lx01**




Ein Authentication Service muss alle Principals in (s)einer Datenbank finden

# Keytab

- Was sind Keytab Dateien
  - Jeder Principal muss seine Schlüssel kennen.
  - Der Client gelangt durch sein Anwender Passwort und die string2key Funktion an seine Schlüssel
  - Ein Dienst braucht aber direkten Zugriff auf seine Schlüssel und daher werden die Langzeitschlüssel in lokalen Keytab Datei(en) gespeichert. Jede Service muss wissen welche Keytab es zu verwenden hat. z.B:
    - /etc/krb5.keytab (rw- --- --- root:root)
      - für Services die als Root laufen
    - /etc/apache/krb5.keytab (rw- --- --- www:www)
      - für spezielle Services die nicht als Root laufen

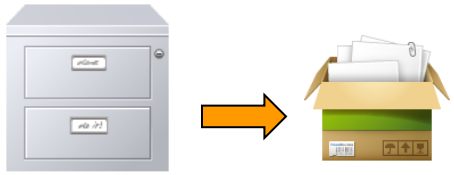
# Keytab für lx01 erstellen

- Keytab auf lx01 in kadmin erstellen
  - ktadd -k /etc/krb5.keytab host/lx01.example.com
    - -k ...der obige Wert ist der Standardwert und die Angabe dieses Parameters kann auch entfallen

```
root@kdc01:~#  kadmin -p user/admin@EXAMPLE.COM
Authenticating as principal user/admin@EXAMPLE.COM with password.
Password for user/admin@EXAMPLE.COM:
kadmin:  ktadd -k /etc/krb5.keytab host/lx01.example.com
Entry for principal host/lx01.example.com with kvno 2, encryption type
  AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.k.
Entry for principal host/lx01.example.com with kvno 2, encryption type
  ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/lx01.example.com with kvno 2, encryption type
  Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
```

# lokale Keytab Datei

Passwort für meine Service Principals:
Ich habe mein Geheimnis in einer lokalen Datei mit dem Namen /etc/keytab abgespeichert



**Ressource host/lx01**


Falls ein Service das Geheimnis (Passwort) benötigt um etwas zu entschlüsseln kann das Service direkt auf diese Datei zugreifen. Optional kann jedes Service seine eigene Keytab haben

Jeder Principal benötigt Zugriff aus sein Geheimnis (Passwort)

# Ticket für ein Service beziehen

user

1 kvno: Hallo hier ist mein TGT, bitte gib mir ein Ticket für die Ressource-X (host/kdc01.example.com)

TGT


TGS

Hier ist das TGS für Ressource-X 2

3

TGS

Das TGS für die Ressource-X wird im lokalen Ticket Cache gespeichert

user's PC

TGT TGS

```
root@kdc01:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: user@EXAMPLE.COM

...   Service principal
...   krbtgt/EXAMPLE.COM@EXAMPLE.COM
...   host/kdc01.example.com@EXAMPLE.COM
```

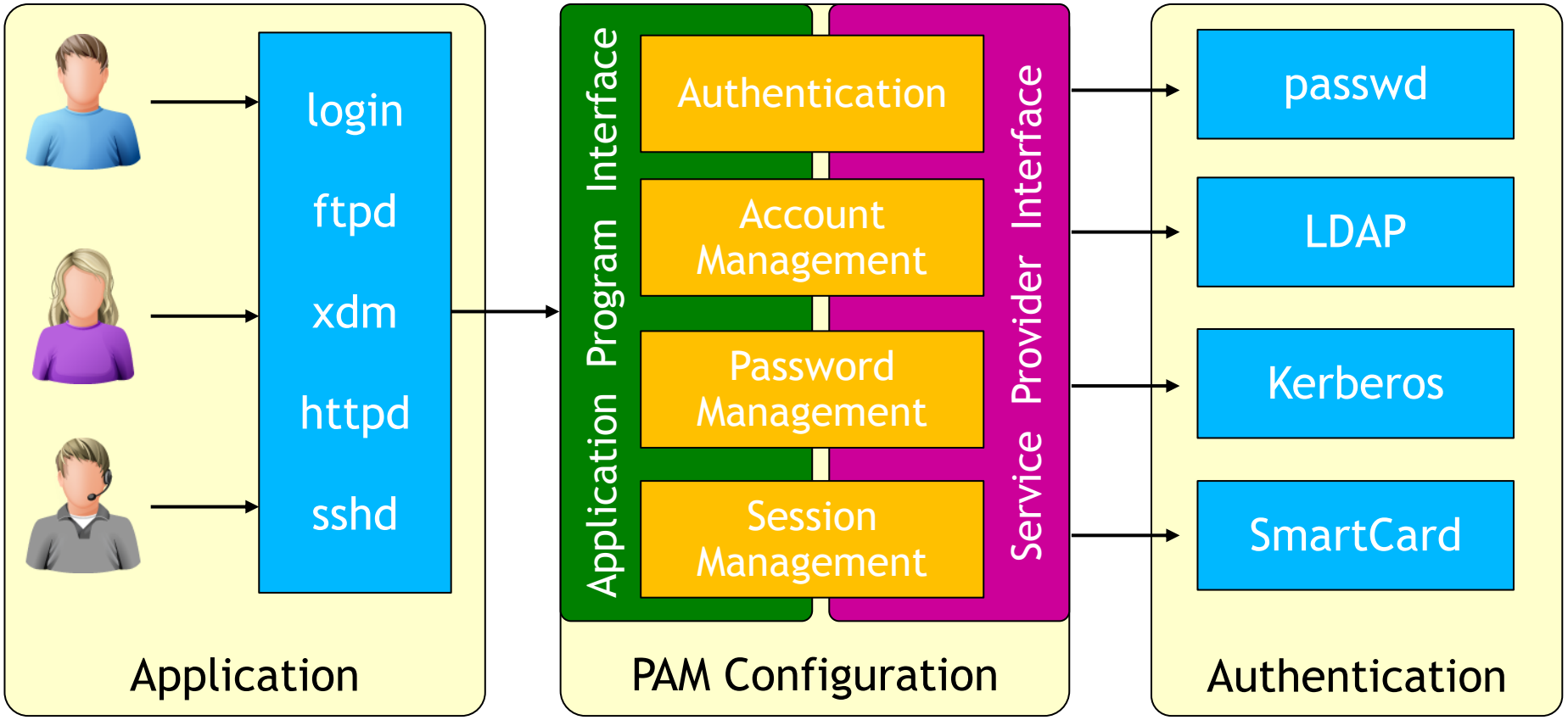
# Keytab für lx01 erstellen

- Testen der Keytab Funktion
  - kinit -k -t /etc/krb5.keytab host/lx01.example.com
    - kinit ...Testet die Client Funktionalität
    - -k ...Keytab benutzen, -t ...Pfad zur Keytab Datei
  - kvno -k /etc/krb5.keytab host/lx01.example.com
    - kvno ...Testet die Client & Dienste Funktionalität
    - Funktioniert erst wenn kinit durchgeführt wurde

```
root@kdc01:~# klist
Valid starting     Expires          Service principal
10/03/11 09:55:29  10/03/11 19:55:29  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 10/04/11 09:55:29
10/03/11 09:55:44  10/03/11 19:55:29  host/lx01.example.com@EXAMPLE.COM
        renew until 10/04/11 09:55:29
```

# Kerberisieren des Linux Logins

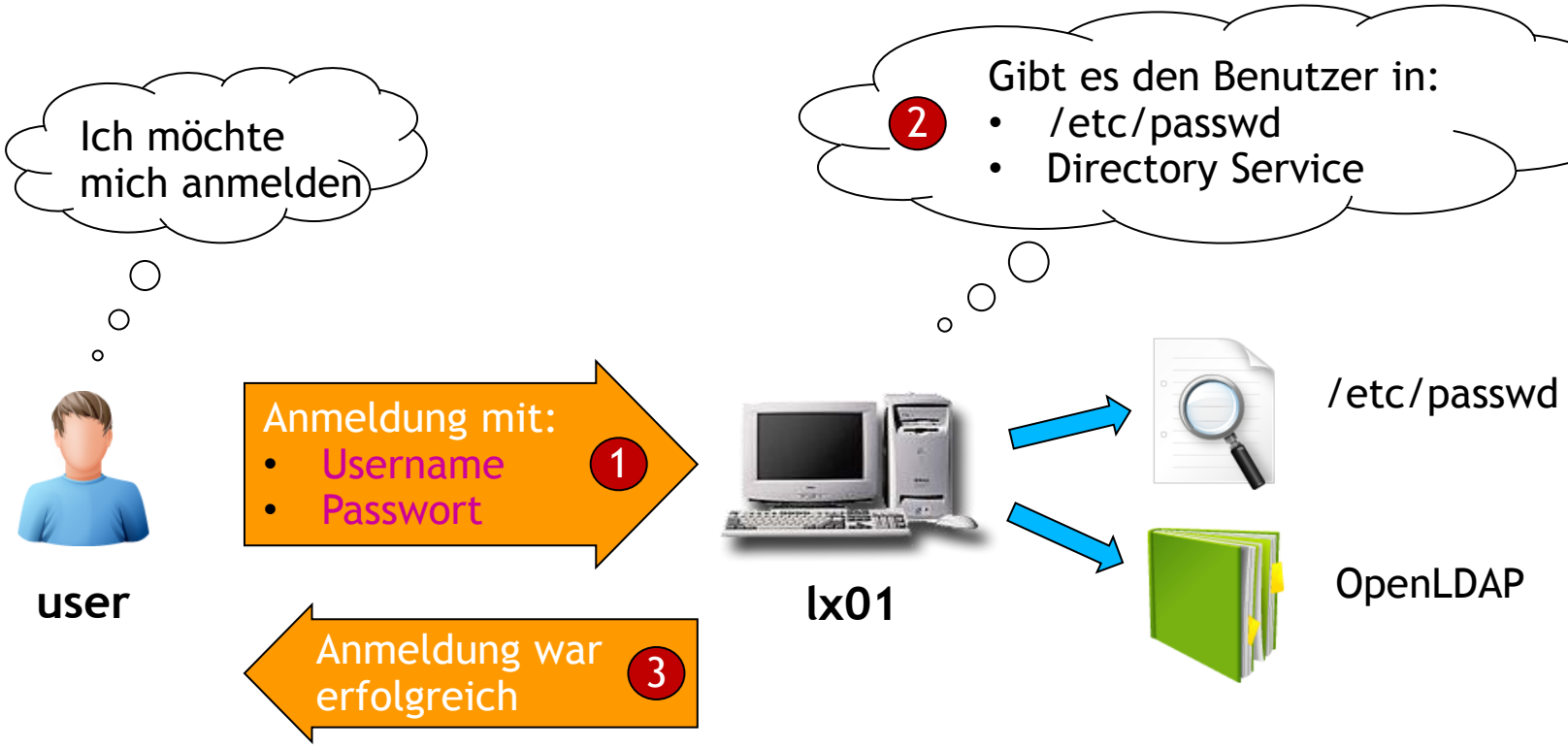
# Pluggable Authentication Modules



# Ablauf des kerberosierten Logins

- Ablauf der Benutzeranmeldung
  1. Der Benutzer meldet sich mit Username und Passwort an
  2. Der Client überprüft die Login Credentials
     - Lokale Benutzerdatenbank, Directory Service
  3. Die PAM Konfiguration erzwingt die zusätzliche Überprüfung ob es diesen Principal auch in der Realm gibt
     1. Wenn ja sendet der KDC ein passendes TGT für diesen User Principal
     2. Das TGT wird im Ticket Cache abgelegt
  4. Der Login Vorgang des Benutzers ist abgeschlossen

# Kerberosierter Linux Login 1



# Kerberosierter Linux Login 2

Ich habe nichts zu tun :-)

**user**

4 PAM-Modul: Gibt es den Benutzer in der Kerberos Realm?

**lx01**

Anmeldung mit: user@EXAMPLE.COM 5

**KDC**

Verschlüsseltes TGT  6

7


Ticket Cache

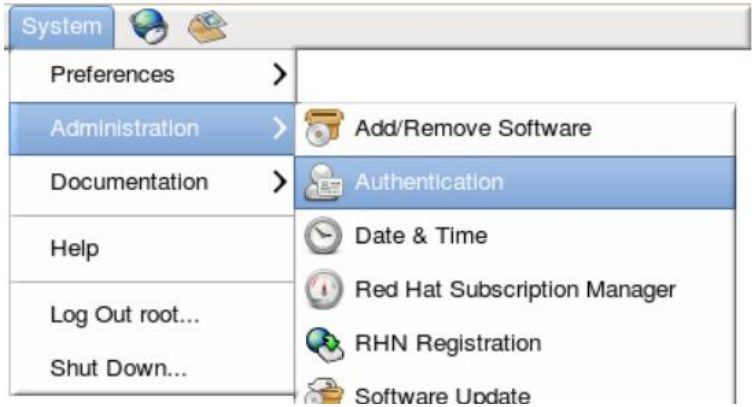
# Kerberos und PAM

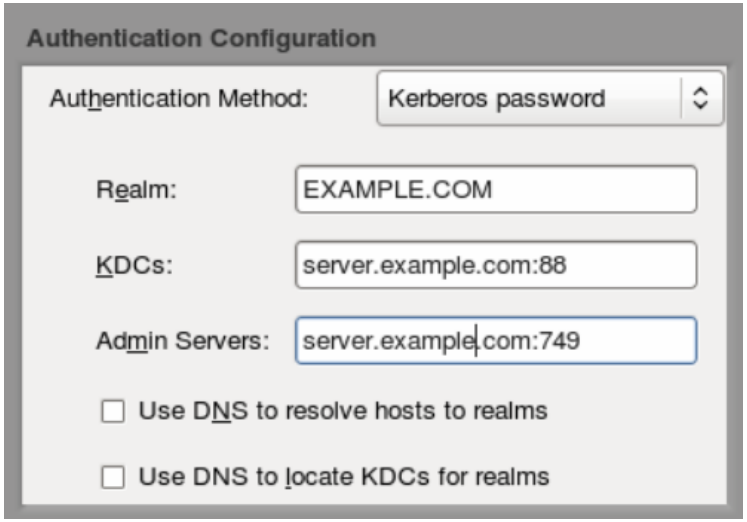
- Debian, Ubuntu
  - apt-get install  libpam-krb5
  - Alle notwendigen Änderungen für die Funktionsweise des Modules werden automatisch vorgenommen

- SUSE
  - zypper install  pam_krb5
  - Automatisch werden keine Änderungen vorgenommen aber es gibt einen tollen Yast Wizard

# Kerberos und PAM

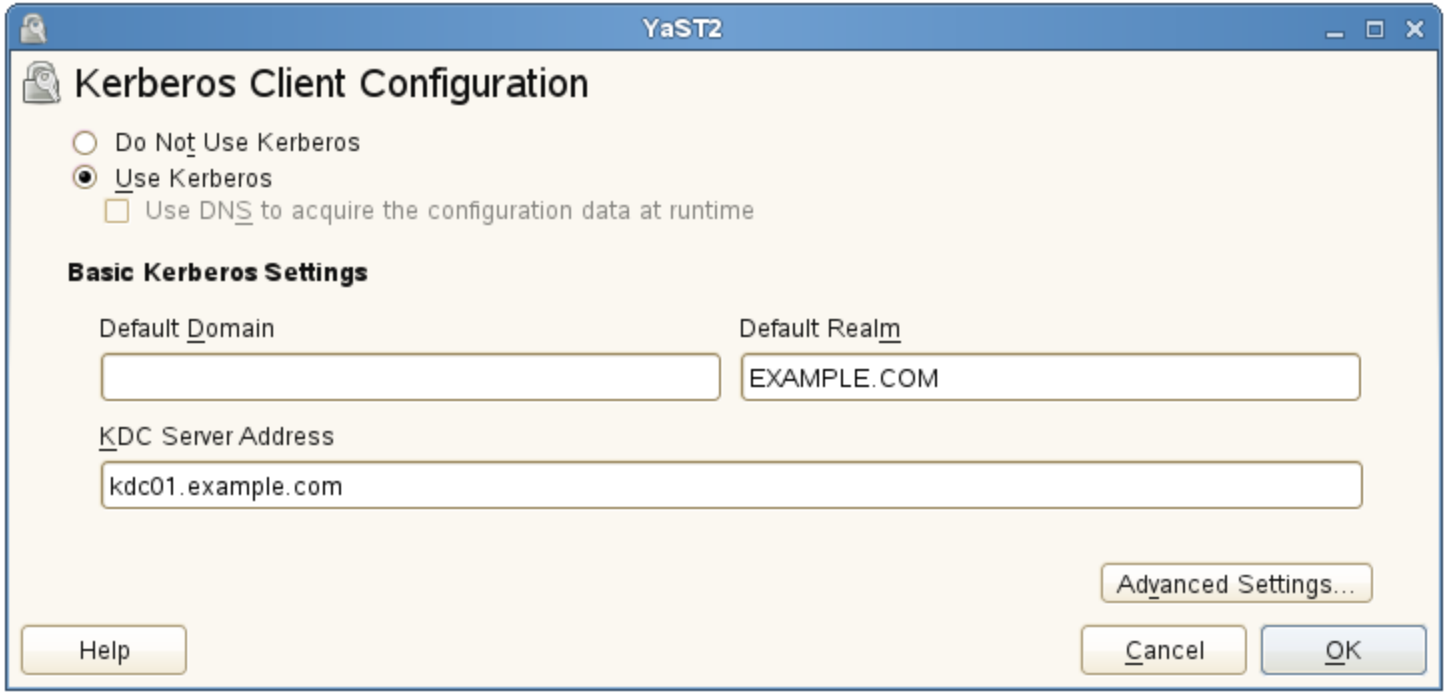
- Red Hat
  - yum install krb5-libs  krb5-workstation
  - Die Konfiguration erfolgt über system-config-authentication



# Yast Kerberos Client Konfiguration

- Yast - Netzwerk Services - Kerberos Client
  - Default Realm und KDC-Adresse ausfüllen und OK klicken



# Kerberos und pam auf DEBIAN

```
# /etc/pam.d/common-auth
auth    [success=1 default=ignore]      pam_winbind.so krb5_auth
    krb5_ccache_type=FILE cached_login try_first_pass
```

```
# /etc/pam.d/common-account
account required        pam_krb5.so minimum_uid=1000
```

```
# /etc/pam.d/common-password
password    requisite     pam_krb5.so minimum_uid=1000
```

```
# /etc/pam.d/common-session
session optional    pam_krb5.so minimum_uid=1000
```

- minimum_uid=1000
  - stellt sicher das Service Account wie auch Root sich anmelden können auch wenn kein KDC erreichbar ist

# Passwort Änderung

```
#Ändern eines Benutzer Kennwortes welcher auch in Kerberos existiert
 root@kdc01:~#  passwd user
Current Kerberos password:
Enter new Kerberos password:
Retype new Kerberos password:
passwd: Passwort erfolgreich geändert
```

```
#Ändern eines Benutzer Kennwortes welches in Kerberos nicht existiert
 root@kdc01:~#  useradd -m karl
root@kdc01:~#   passwd karl
Current Kerberos password:
passwd: Benutzer bei zu Grunde liegendem Authentifizierungsmodul nicht bekannt
passwd: Passwort nicht geändert
# Aufgrund der PAM-Einstellung Requisite wird sofort abgebrochen
```

# Kerborisierter Login

```
# Kerborisierter Login funktioniert nicht da der Principal nicht bekannt ist
root@kdc01:~#  su - karl
karl@kdc01:~#  klist
klist: No credentials cache found (ticket cache FILE:/tmp/krb5cc_1001)
# Kerborisierter Login funktioniert da der Principal bekannt ist
karl@kdc01:~$ su - user
Password:
user@kdc01:~$ klist
Ticket cache: FILE:/tmp/krb5cc_1000_Vhz6e0
Default principal: user@EXAMPLE.COM

Valid starting     Expires            Service principal
01/07/12 18:10:43  01/08/12 04:10:43  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 01/08/12 18:10:43
```

- Tipp
  - für eine professionelle Lösung für lx01 benötigt man einen zentralen Verzeichnisdienst wie OpenLDAP, ADS, eDirectory

# Troubleshooting

- su - user
  - Wenn der Root mit SU zu dem Benutzer user wechselt wird er nicht nach einem Kennwort gefragt
    - Aus diesem Grunde kann auch kein passendes Kerberos Ticket geholt werden da ja das passende Geheimnis zum Principal nicht vorhanden ist!
  - Nur wenn beim SU nach einem Passwort gefragt wurde kann das passende Ticket angefordert werden

# Kerberisieren des SSH-Servers auf KDC01

# Vorbereiten von SSH auf KDC01

- Vorbeitende Maßnahmen
  - Lokalen User mit Heimatverzeichnis erstellen
    - useradd -m user
    - passwd user ...PW=linux
  - In kadmin Keytab für KDC01 erstellen
    - ktadd -k /etc/krb5.keytab host/kdc01.example.com
    - Optional Keytab Inhalt prüfen: klist -k -e -K
  - SSH Software installieren
    - apt-get install ssh

# Vorbereiten von SSH auf KDC01

- Kerberos Authentifizierung im SSH-Server aktivieren

```
# /etc/ssh/sshd_config
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
```

- SSH Daemon neu starten
  - /etc/init.d/ssh restart
  - update-rc.d ssh enable
- SSH Zugriff testen (ohne Passwort Abfrage)
  - kinit user
  - ssh user@kdc01.example.com

# Vorbereiten von SSH auf lx01

- Kerberos Authentifizierung im SSH-Client aktivieren
  - Achtung nicht Client und Server Konfiguration verwechseln

```
# /etc/ssh/ssh_config
GSSAPIAuthentication yes
GSSAPIDelegateCredentials yes
```

- Aufbauen einer SSH Verbindung von lx01
  - kinit user ...Kerberos Ticket für User holen
  - ssh user@kdc01.example.com ...Passwortfreier Zugriff sollte gehen

# Troubleshooting

- Falls der Verbindungsaufbau nicht klappt
  - Starten des ssh clients im Verbose Modus -vv
  - Optional können auch die Client Parameter -o option explizit übergeben werden

```
#Starten von SSH im Diagnose Modus
 root@kdc01:~#  ssh -vv  -o GSSAPIDelegateCredentials=yes \
 -o GSSAPIKeyExchange=yes -o GSSAPIAuthentication=yes user@kdc01
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Applying options for *
debug2: ssh_connect: needpriv 0
debug1: Connecting to kdc01 [127.0.1.1] port 22.
debug1: Connection established.
debug1: Unspecified GSS failure.  Minor code may provide more information
Server host/kdc01@EXAMPLE.COM not found in Kerberos database
...
```


LinuxCampus.net
trainings from the experts