Leibniz
Universität
Hannover

Kerberos

Protokoll, Implementation und Anwendungen

Protokoll
Implementation
Anwendungen

Designziele

Hintergrund: Es ist Anfang der 1980er Jahre, es gibt noch kein ssh, kein (etabliertes) X.509.

- Gegenseitige (mutual) Authentifizierung über einen unsicheren Kanal
- Single Sign On: nur einmal (z.B. pro Tag) anmelden, viele Services nutzen

Technische Vorraussetzungen

- Symmetrische Verschlüsselung
  - Public-Key-Verfahren waren zwar schon erfunden, aber nicht verbreitet
- UDP, TCP
- gleichlaufende Uhren auf allen beteiligten Systemen

Begriffe

Prinzipal: Eine Benutzer- oder Service-Identität

Realm: Gesamtheit von Prinzipalen, die gemeinsam verwaltet werden

- user@EXAMPLE.COM
- host/pc1.example.com@EXAMPLE.COM

Begriffe

Secret Key (K): Geheimnis, das nur dem Prinzipal und der Realm-Verwaltung bekannt ist; langlebig

Session Key (S): Temporärer Schlüssel für die abgesicherte Kommunikation von Prinzipalen untereinander

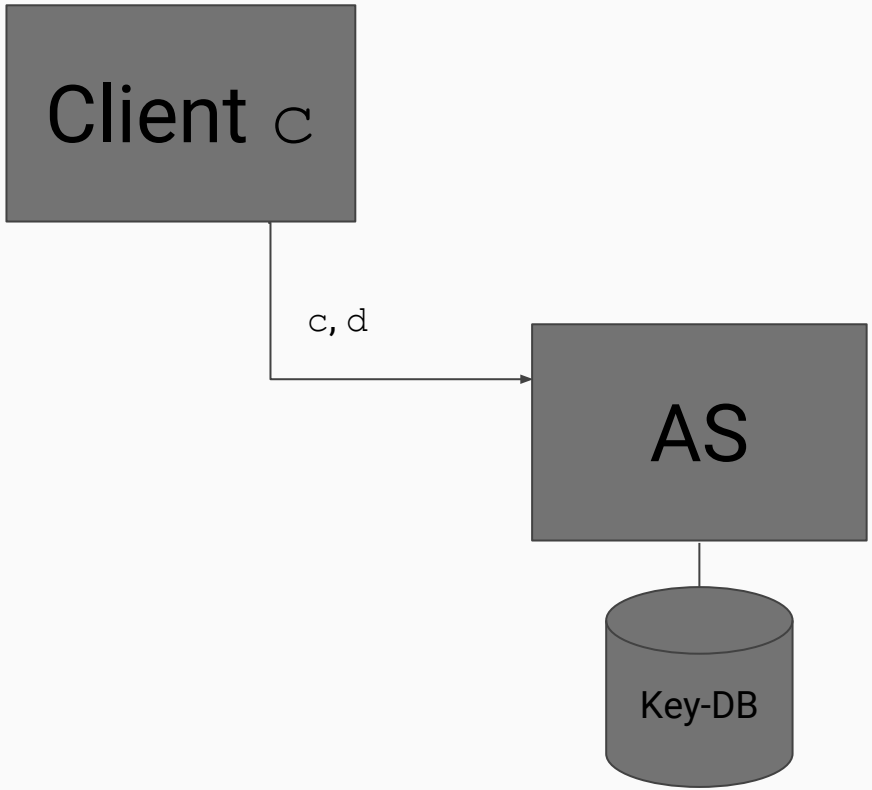
Anmeldung am Authentication Service (AS)

Die Anmeldung eines Clients c am AS ist immer bezogen auf einen Dienst d, sie besteht aus zwei Nachrichtentypen:

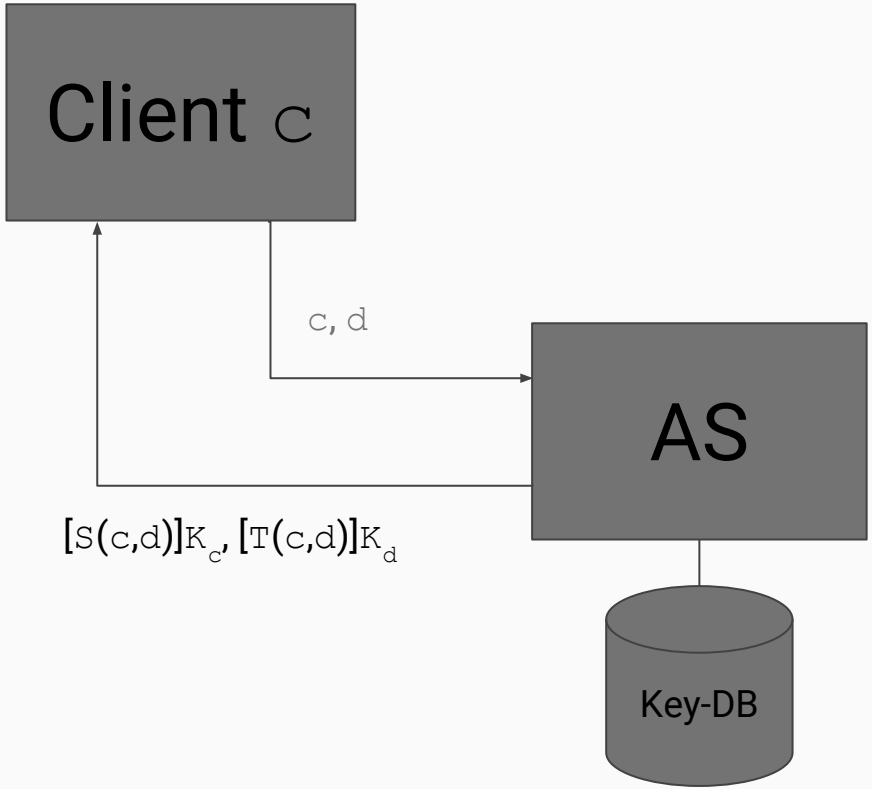
- KRB_AS_REQ: Client c bittet um Ticket für Zugriff auf d
- KRB_AS_REP: AS generiert Session Key $S(c,d)$ und schickt ihn verschlüsselt mit K_c zurück. Das Ticket $T(c,d)$ besteht aus c, d, $S(c,d)$ und weiteren Attributen und Flags und wird mit K_d verschlüsselt.

(es gibt noch einen dritten Typ, KRB_ERROR)

KRB_AS_REQ



KRB_AS_REP

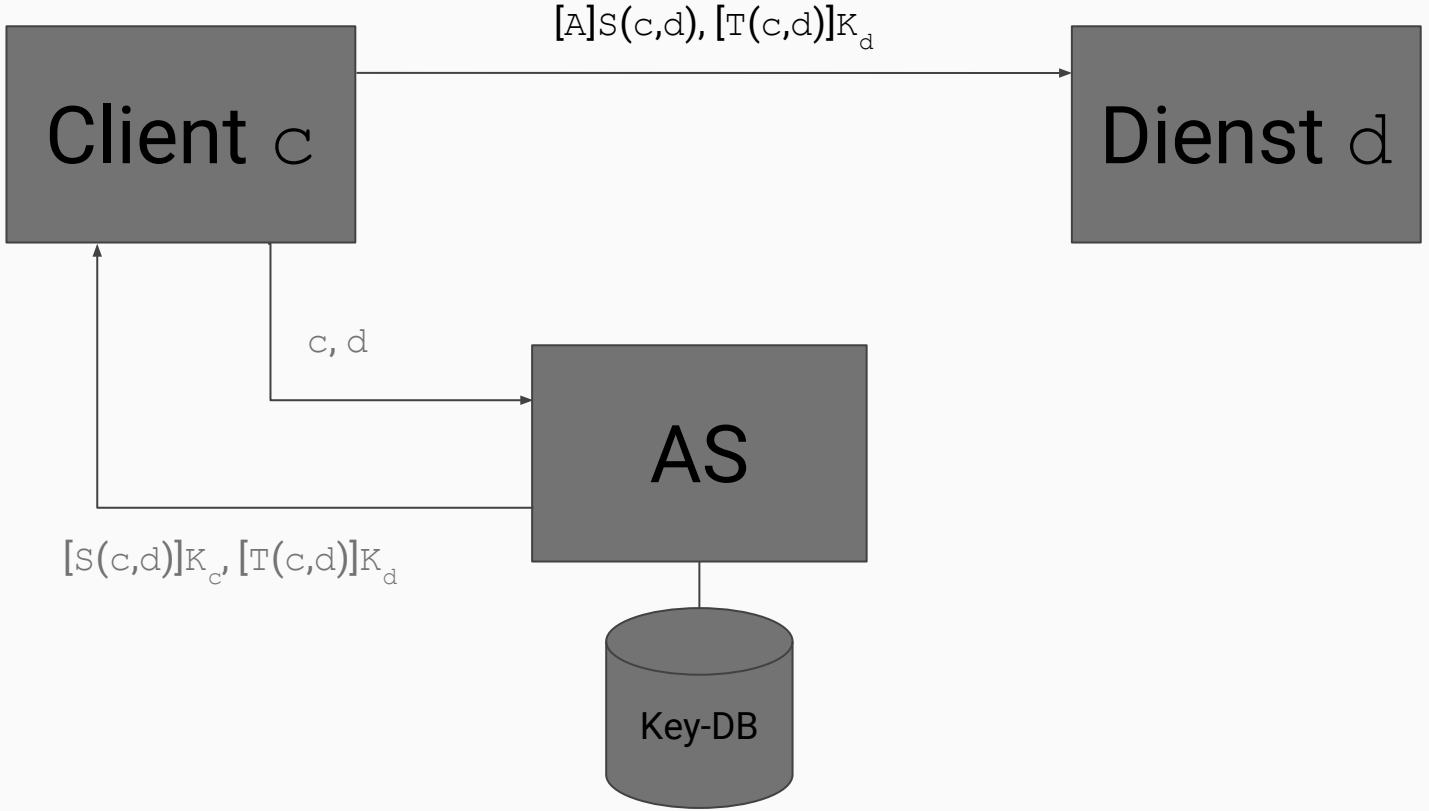


Anmeldung am Application Server (AP)

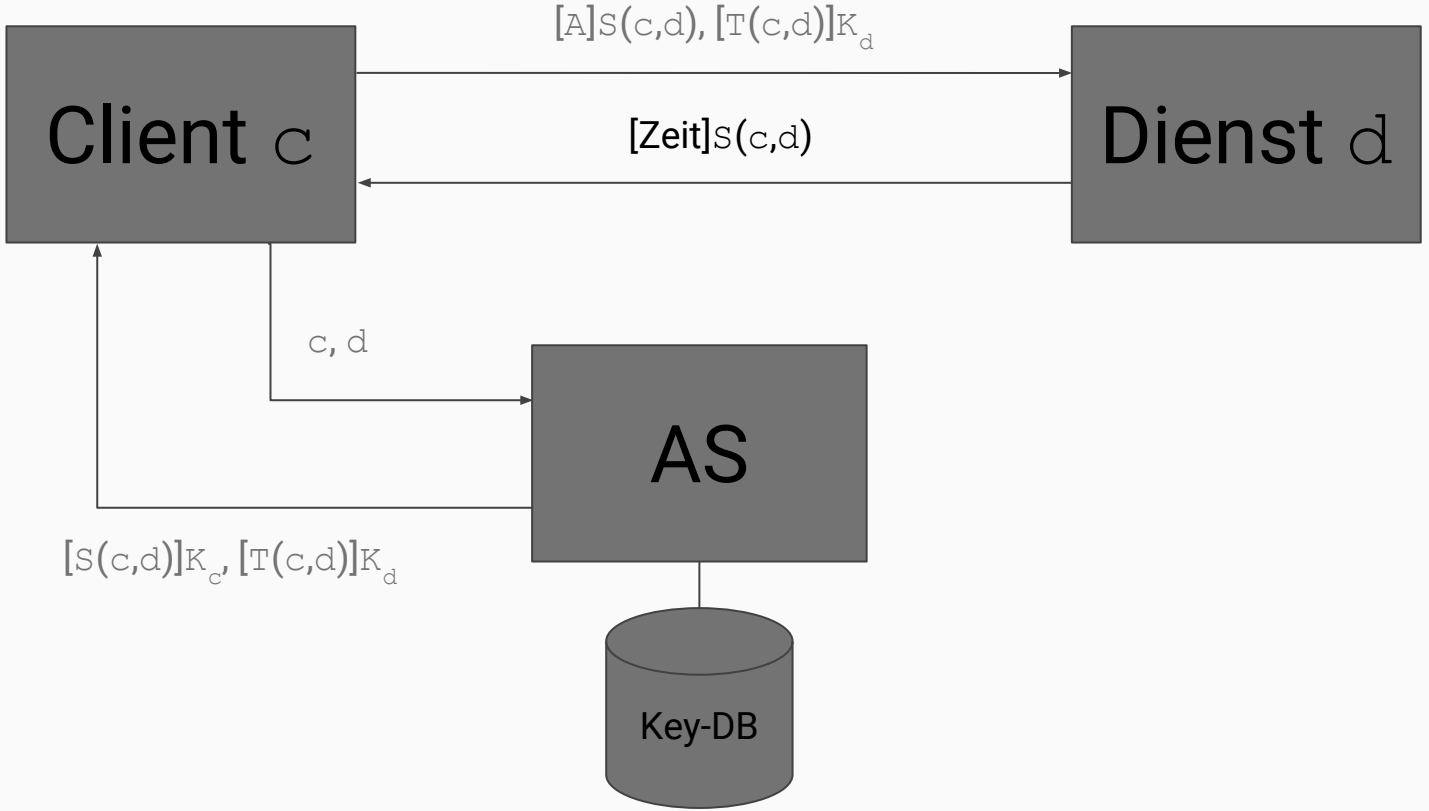
Mit dem Ticket $T(c,d)$, das mit dem Key des Service verschlüsselt ist, kann sich der Client an diesem Service anmelden. Es gibt wieder zwei Nachrichtentypen:

- KRB_AP_REQ: Der Client sendet $[T(c,d)]K_d$ und einen mit dem Session Key verschlüsselten Authenticator A an den Service. A enthält c, einen Zeitstempel und evtl. weitere Daten.
- KRB_AP_REP: Der Service sendet den Zeitstempel, verschlüsselt mit dem Session Key, zurück.

KRB_AP_REQ



KRB_AP_REP



Noch nicht ganz fertig!

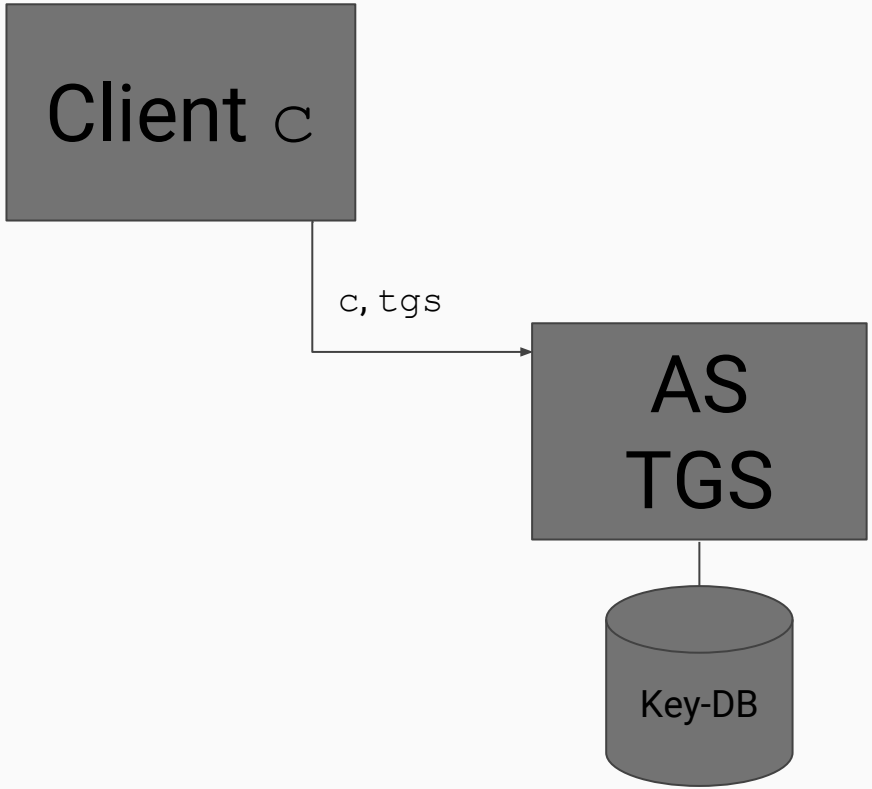
Bis auf Details erlaubt das Verfahren die wechselseitige Authentifizierung, aber:

- Es müsste für jeden Dienst wiederholt werden, der Benutzer müsste also jedesmal sein Passwort eingeben: kein Single Sign On!

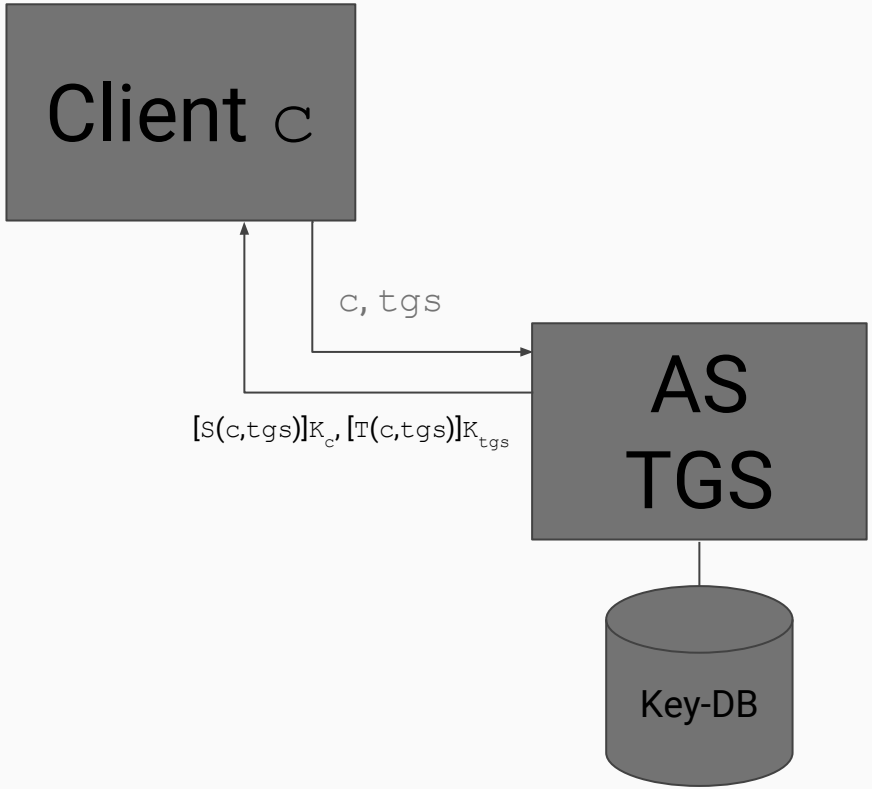
Zwischen-Service TGS (Ticket Granting Service):

- Ein Service mit Zugriff auf die Key-DB, der Service-Tickets ausstellt.

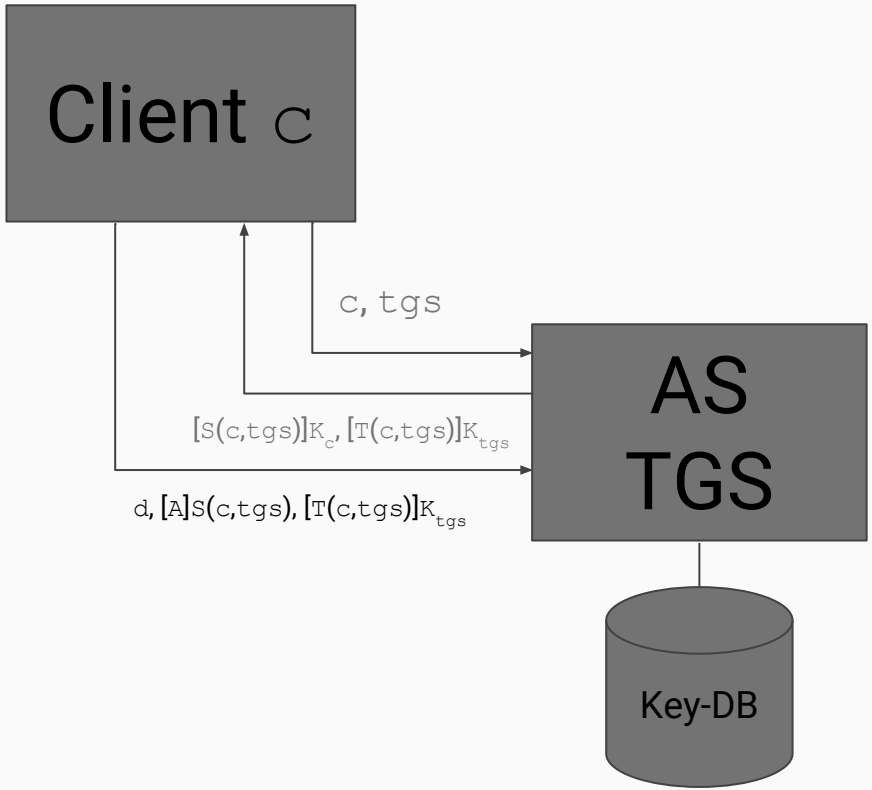
KRB_AS_REQ (TGS)



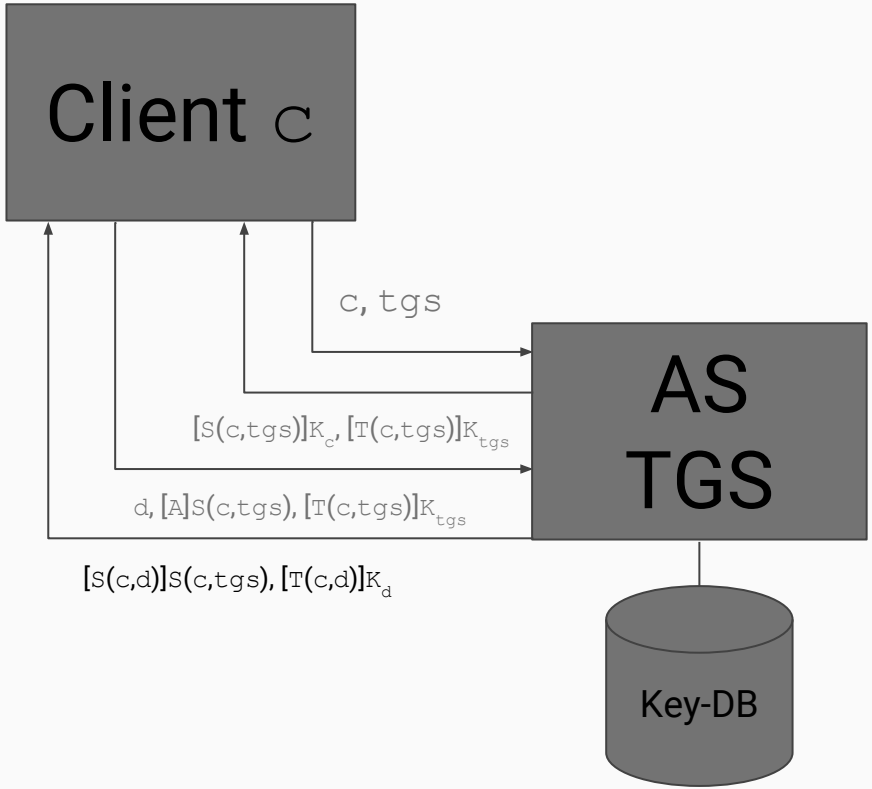
KRB_AS_REP (TGS)



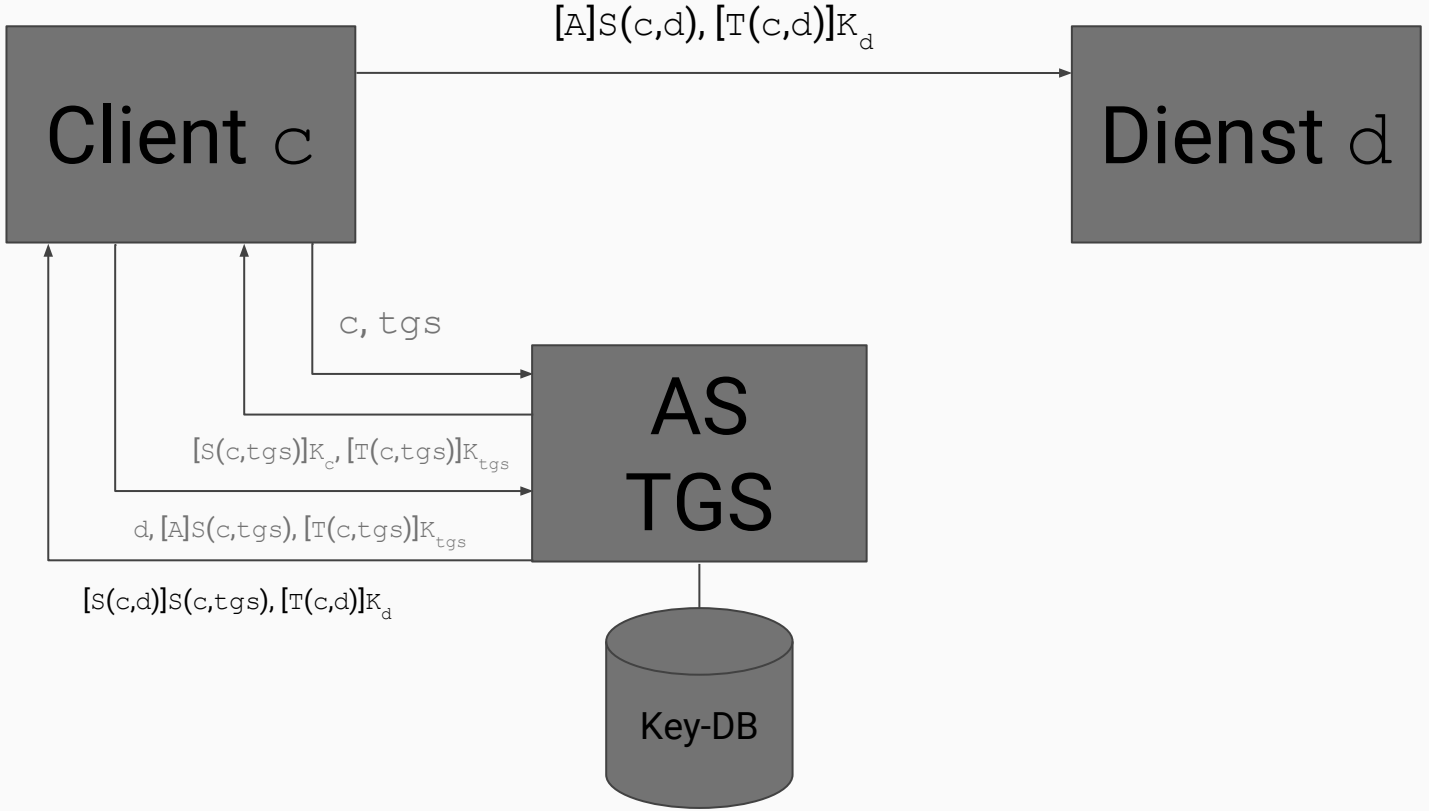
KRB_TGS_REQ


Client C
c, tgs
AS
TGS
[S(c,tgs)]K_c, [T(c,tgs)]K_tgs
d, [A]S(c,tgs), [T(c,tgs)]K_tgs
Key-DB

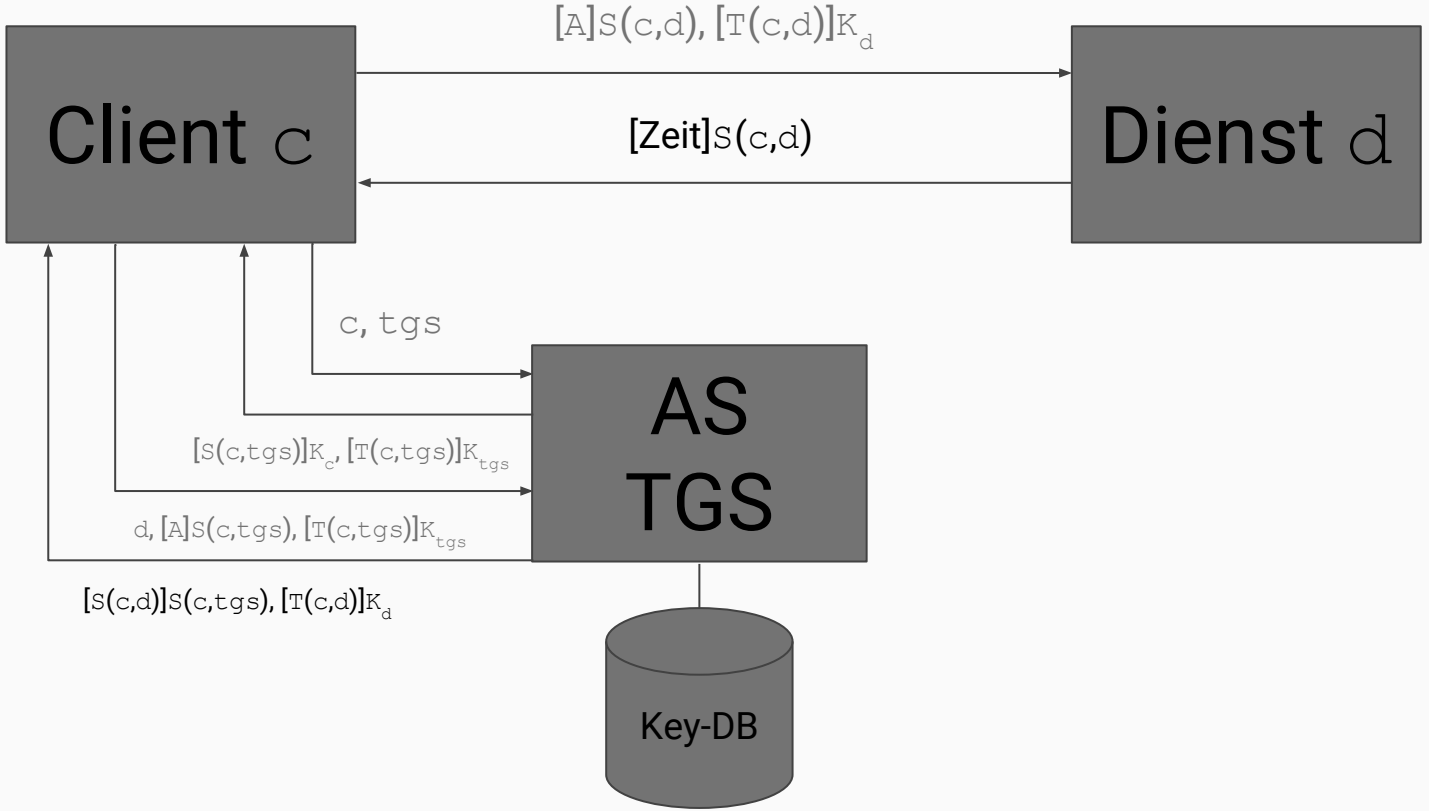
KRB_TGS_REP


Client C
c, tgs
AS
TGS
[S(c,tgs)]K_c, [T(c,tgs)]K_tgs
d, [A]S(c,tgs), [T(c,tgs)]K_tgs
[S(c,d)]S(c,tgs), [T(c,d)]K_d
Key-DB

KRB_AP_REQ (TGS)



KRB_AP_REP (TGS)



Das waren die wesentlichen Punkte des Protokolls. Es fehlen u.a.:

- Pre-Authentication: Der Client beweist durch das Verschlüsseln eines Zeitstempels mit seinem Key vorab seine Authentizität.
- KRB_SAFE und KRB_PRIV: signierte bzw. verschlüsselte Nachrichten
- Cross-Realm-Authentifizierung: Anmeldung an „fremden” Diensten in einem anderen Realm; wird ermöglicht durch „inter-realm” Keys in den beteiligten Kerberos-Servern.

Protokoll
Implementation
Anwendungen

MIT Kerberos

- Ur-Implementierung
  - es gibt weitere, z.B. Heimdal, Shishi
- ist in den üblichen Paketverwaltungen vorhanden
- AS und TGS sind integriert, heißen hier KDC (Key Distribution Center)
- Außerdem gibt es den admin-server, er erlaubt administrative Arbeiten am Realm (Prinzipale anlegen, löschen; Passwortänderungen)

Relevante Pakete in debian installieren

Die benötigten Pakete holt man mit

```
apt-get install krb5-kdc krb5-admin-server krb5-config krb5-user krb5-doc
```

Es werden ein paar Parameter abgefragt (z.B. gewünschter Realm-Name, Servernamen). Nach der Installation wird zwar der KDC gestartet, aber es muss noch explizit ein Realm angelegt werden:

```
krb5_newrealm
```

Das Skript sammelt ein wenig Zufall und fragt nach einem Master-Passwort für den Realm. Nach einem Neustart der Dienste (`krb5-kdc` und `krb5-admin-server`) kann man mit

```
kadmin.local
```

den Realm administrieren.

Server-Konfig: /etc/krb5kdc/kdc.conf

Die Install-Skripte haben schon lauffähige Konfigurationen erstellt. Im Detail:

/etc/krb5kdc/kdc.conf :

```
[kdcdefaults]
    kdc_ports = 88, 750
…
```

Standardmäßig lauscht des KDC nur auf UDP, es wird auch empfohlen, TCP nicht zu verwenden. Die entsprechende Option heißt `kdc_tcp_ports`, wird sie weggelassen, wird kein TCP verwendet.

Port `750` kann auch weggenommen werden, ist veraltet.

Server-Konfig: /etc/krb5kdc/kdc.conf

Es folgt ein Konfigurationsblock für jeden Realm:

```
…
[realms]
    EXAMPLE.COM = {
        database_name = /var/lib/krb5kdc/principal
        admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
        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 = …
        supported_enctypes = …
        default_principal_flags = +preauth
    }
```

Der KDC kann (laut Doku) Server für bis zu 32 Realms sein.

Server-Konfig: /etc/krb5kdc/kadm5.acl

Auf dem Admin-Server selbst kann mit `kadmin.local` ohne Anmeldung administriert werden. Sollen echte Nutzer Administrationsrechte erhalten, kann eine ACL-Datei angelegt werden:

/etc/krb5kdc/kadm5.acl :

```
*/admin *
```

Diese Regel besagt, dass jedes Prinzipal, das `admin` als zweite Komponente hat, volle Administrationsrechte erhält. Eine genaue Beschreibung der Syntax ist in der Man-Page zu `kadmind` enthalten.

Danach kann `kadmin` benutzt werden, um den Realm zu administrieren, auch von fremden Hosts aus.

Server-Konfig: /etc/krb5.conf

Die Datei /etc/krb5.conf dient hauptsächlich der Konfiguration der Client-Anwendungen, wird aber auch von der Server-Diensten gelesen. Hier kann auch ein Logging eingestellt werden:

```
[logging]
        kdc = FILE:/var/log/kerberos/krb5kdc.log
        admin_server = FILE:/var/log/kerberos/kadmin.log
        default = FILE:/var/log/kerberos/krb5lib.log
```

Protokoll
Implementation
Anwendungen

Konfiguration der libkrb5: /etc/krb5.conf

In der Datei /etc/krb5.conf stehen Einstellungen für Clientanwendungen:

Der `default_realm` wird automatisch ergänzt bei Prinzipalen ohne Realm-Angabe.

```
[libdefaults]
        default_realm = EXAMPLE.COM
```

Konfiguration der libkrb5: /etc/krb5.conf

In den Abschnitten `domain_realm` und `realms` wird das Mapping Domain->Realm sowie Realm->Server eingestellt:

```
[domain_realm]
        example.com = EXAMPLE.COM
        .example.com = EXAMPLE.COM

[realms]
        EXAMPLE.COM = {
                kdc = krb1.example.com
                kdc = krb2.example.com
                admin_server = krb1.example.com
        }
```

Konfiguration via DNS

Die Informationen der vorherigen Folie können auch in speziellen DNS-Einträgen hinterlegt werden:

```
$ORIGIN example.com.
_kerberos              TXT "EXAMPLE.COM"

_kerberos._udp         SRV 0 0  88 krb1
                       SRV 0 0  88 krb2
_kerberos-adm._tcp     SRV 0 0 749 krb1
_kpasswd._udp          SRV 0 0 464 krb1
```

Einrichten von libpam-krb5

In einer üblichen debian-Installation liefert ein

```
apt-get install libpam-krb5
```

ein (fast) funktionierendes Beispiel. Was fehlt, ist der host-Key auf dem Client-Rechner. Dazu muss, am besten von diesem Client aus, `kadmin` benutzt werden, um ein entsprechendes Prinzipal anzulegen und den Key zu exportieren:

```
klopp@client1:~$ kadmin -p klopp/admin
kadmin:  addprinc -randkey host/client1
kadmin:  ktadd host/client1
```

Jetzt kann man sich auf diesem Rechner mit Hilfe seines Kerberos-Passworts anmelden. Außerdem wird in der Session das TGT erzeugt, so dass weitere kerberisierte Dienste benutzt werden können.

Kerberisieren von ssh

In /etc/ssh/sshd_config und (/etc/ssh/ssh_config oder ~/.ssh/config):

```
GSSAPIAuthentication yes
```

Damit wird ein Kerberos-Ticket zum Anmelden am System erzeugt und akzeptiert (auch dazu ist ein gültiger Host-Key nötig).

Wenn man das Ticket in die neue Sitzung weiterleiten möchte, ist noch folgende Einstellung nötig:

In /etc/ssh/ssh_config (oder ~/.ssh/config):

```
GSSAPIDelegateCredentials yes
```