**Technische Universität Chemnitz**

Fachbereich Informatik

---

# AFS-Kerberos-5-Migration

Studienarbeit

eingereicht von:

Norman Schulze
Matrikelnr. 14662

Hauptstraße 38
09390 Meinersdorf

Prüfer:

Prof. Dr. Uwe Hübner

Meinersdorf, d. 14.10.1998

## Aufgabenstellung

Es sollen Untersuchungen und Tests zum Übergang von AFS-kaserver zu Kerberos 5 ausgehend vom "AFS-Kerberos 5 migration kit" durchgefuehrt werden.
(/afs/world/transarc.com/public/afs-contrib/tools/afs-krb5-1.1.tar.Z).
Dabei ist die in Europa verfügbare Kerberos5-Implementation
(http://www.pdc.kth.se/heimdal/) einzubeziehen, ferner die Absichten von Microsoft zur Integration von Kerberos 5 in NT 5.0 (bis hin zum KDC). Es soll untersucht werden, ob und wie eine solche Migration unter unseren Bedingungen machbar ist (ggf. findet man auch Aussagen zur Verfügbarkeit, Nützlichkeit usw. der angekündigten Implementation von Microsoft in NT5). Neben der Betrachtung der Server-Seite sollten nach Möglichkeit auch Voraussetzungen zur Integration in Clienten untersucht und geschaffen werden (Linux-PAM).

# INHALTSVERZEICHNIS


**0. EINLEITENDE BEMERKUNGEN ..... 1**

**1. KERBEROS ..... 2**

1.1. EINFÜHRUNG ..... 2
1.2. DAS KERBEROS-MODELL ..... 2
1.3. DER TICKET-NACHRICHTENAUSTAUSCH ..... 4
1.4. DER TICKET-GRANTING-SERVICE TGS ..... 5
1.5. INTER-REALM-AUTHENTIFIZIERUNG ..... 6
1.6. ÄNDERUNGEN IN KERBEROS VERSION 5 ..... 6
1.7. VERSCHLÜSSELUNG IN KERBEROS ..... 9
*1.7.1. Kryptosysteme in Kerberos ..... 9*
*1.7.2. Die String-to-Key – Funktion zur Schlüsselgenerierung ..... 9*
*1.7.3. Keytabs zur Speicherung von Server-Schlüsseln ..... 10*
*1.7.4. Key version numbers ..... 11*

**2. MIGRATION VON AFS-KASERVER ZU KERBEROS V5-KDC ..... 12**

2.1. UNTERSCHIEDE ZWISCHEN AFS-KERBEROS UND KERBEROS V4 ..... 12
2.2. AUTHENTIFIZIERUNG AN AFS BEI NUTZUNG VON AFS-KASERVER ..... 13
2.3. UNTERSCHIEDE ZWISCHEN AFS-KASERVER UND KERBEROSV5-KDC ..... 14
2.4. AUTHENTIFIZIERUNG AN AFS BEI NUTZUNG VON KERBEROS V5-KDC ..... 16
2.5. VORAUSSETZUNGEN ZUR INTEGRATION VON AFS IN KERBEROS V5 ..... 18
*2.5.1. Konvertierung der AFS-kaserver-Datenbank ..... 18*
*2.5.2. Abwärtskompatibilität zu AFS und Kerberos V4 ..... 19*
*2.5.3. Abstimmung des AFS-KeyFile mit der KerberosV5-Datenbank ..... 20*
2.6. SCHRITTE ZUR MIGRATION VON AFS-KASERVER ZU KERBEROS V5-KDC ..... 21

**3. KERBEROS VERSION 5 – „HEIMDAL“ ..... 22**

3.1. ALLGEMEINES ..... 22
3.2. INSTALLATION VON „HEIMDAL“ UND KONFIGURATION EINER REALM ..... 22
*3.2.1. Die Testumgebung im URZ ..... 22*
*3.2.2. Installation von „Heimdal“ ..... 22*
*3.2.3. Die Bestandteile von „Heimdal“ ..... 23*
*3.2.4. Das KDC von „Heimdal“ ..... 26*
*3.2.5. Die Erstellung einer neuen Realm ..... 27*
*3.2.6. Erzeugen von Keytabs ..... 28*
*3.2.7. Test der Grundinstallation ..... 28*

**4. INTEGRATION VON „HEIMDAL“ IN AFS ..... 30**

4.1. ZIELE ..... 30
4.2. BESTEHENDE INFRASTRUKTUR IM URZ ..... 30
4.3. VORGEHENSWEISE BEI DER MIGRATION ..... 32
4.4. EINRICHTUNG DER KERBEROS-DATENBANKSERVER ..... 32
*4.4.1. Installation von „Heimdal“ ..... 32*
*4.4.2. Die Initialisierung der Realm ..... 33*
*4.4.3. Die Erstellung des AFS-Service-Principals ..... 33*
*4.4.4. Die Konvertierung der kaserver-Datenbank ..... 35*
*4.4.5. Die Verteilung der Principal-Datenbank an die Slave-Server ..... 38*
*4.4.6. Der Paßwort-Änderungs-Server kpasswdd ..... 39*
*4.4.7. Der Administrationsserver kadmind ..... 41*
*4.4.8. Inbetriebnahme der Datenbankserver ..... 41*
4.5. EINRICHTUNG DER CLIENT-HOSTS ..... 43
*4.5.1. Konfiguration der Hosts ..... 43*
*4.5.2. Installation der Kerberos-Client-Programme ..... 43*
*4.5.3. Installation der Kerberos-kompatiblen Server ..... 44*
*4.5.4. Integration der Secure-Shell (SSH) ..... 45*
*4.5.5. Authentifizierung von Nutzern an Client-Hosts ..... 46*
4.6. CROSS-CELL-AUTHENTIFIZIERUNG ..... 47

**5. DIE KERBEROSV5-IMPLEMENTATION VON WINDOWS NT 5 ..... 49**

**6. ABSCHLIEßENDE BEMERKUNGEN ..... 52**

**ANHANG – DIE KONFIGURATIONSDATEI „KRB5.CONF“ ..... 54**

**LITERATURVERZEICHNIS ..... 55**

# 0. Einleitende Bemerkungen

Aufgabe der vorliegenden Studienarbeit war die Untersuchung der Möglichkeit der Migration von AFS-kaserver zu Kerberos Version 5 im Rechenzentrum der TU Chemnitz unter Einbeziehung des „AFS-Migration-Kit“ der Firma Transarc.
Die Version 5 des Kerberos-Authentifizierungs-Protokolls wurde bisher in einer kleinen Anzahl von Produkten implementiert, dabei ist die Implementation des Massachusetts Institute of Technology (MIT-Kerberos5) die am weitesten verbreitete. Allen diesen Produkten ist gemeinsam, daß sie von Institutionen bzw. Firmen in den Vereinigten Staaten von Amerika entwickelt wurden. Aufgrund des Verbotes des Exports starker Kryptographie aus den USA, welches auch Kerberos betrifft, sind die Produkte in Deutschland nicht erhältlich.
Das „AFS-Migration-Kit“ von Transarc ist als Erweiterung des MIT-KerberosV5 entwickelt worden. Insbesondere wird der MIT-Quellcode durch einen im Kit enthaltenen Patch verändert. Zusätzliche Anwendungen des Kits benötigen zur Übersetzung Include-Dateien und Bibliotheken des MIT-KerberosV5. Da MIT-Kerberos nicht aus den USA exportiert werden darf, kann das „AFS-Migration-Kit“ im URZ der TU Chemnitz nicht genutzt werden.
An der Kungliga Tekniska Högskolan (KTH) Stockholm wird seit 1996 eine frei verfügbare Implementation von Kerberos V5 namens „Heimdal“ entwickelt. Die Software ist zur Zeit unangekündigt, der Zeitpunkt der Veröffentlichung einer offiziellen Beta-Testversion ist nicht bekannt. Das Paket ist in der aktuellen Version „0.0t“ mit einer ausreichenden Anzahl von Netzdiensten und Administrationswerkzeugen ausgestattet, so daß der Funktionsumfang der Software einen Test im Praxisbetrieb erlaubt und sinnvoll erscheinen läßt.
Bei der Analyse des Quellcodes von „Heimdal“ wurde schnell erkennbar, daß eine gewisse Unterstützung von AFS in „Heimdal“ enthalten ist. Insbesondere werden Teile der Funktionalität des AFS-kaserver emuliert. Aufgrund dieser Unterstützung und des erwähnten zufriedenstellenden Funktionsumfangs von „Heimdal“ wurde diese Studienarbeit begonnen, ohne das „AFS-Migration-Kit“ zu verwenden.
Es war weiterhin Aufgabe dieser Studienarbeit, die Absichten der Firma Microsoft zur Integration von Kerberos V5 in Windows NT 5 zu untersuchen. Da Microsoft bisher die Beta2-Version von Windows NT 5 nur einem beschränkten Kreis von Entwicklern zugänglich gemacht hat, waren praktische Tests mit Windows NT 5 nicht möglich. Allerdings konnten Aussagen von Microsoft und Kerberos-Entwicklern analysiert werden, die einige wichtige Aspekte der KerberosV5-Implementation von Windows NT 5 beleuchten.

# 1. Kerberos

## *1.1. Einführung*

Kerberos ist ein Service zur Bereitstellung von Authentifizierung in Netzwerken unter Benutzung symmetrischer Kryptographieverfahren. Das Kerberos-Protokoll wurde am Massachusetts Institute of Technology (MIT) entwickelt, um die Netzdienste des Projektes „Athena“ zu schützen[1]. Es basiert auf der Annahme eines unsicheren Netzwerkes, d.h. Daten können vom Netz abgehört und verändert werden, Adressen können gefälscht werden. Zur Authentifizierung sind deshalb Daten und Adressen nicht geeignet, sondern es werden kryptographische Verfahren benutzt, um die Authentizität von Kommunikationspartnern sicherzustellen.

Kerberos Version 1 bis 3 wurde nur intern am MIT verwendet. Obwohl in erster Linie für das Projekt „Athena“ konzipiert, hat die Version 4 von Kerberos weite Verbreitung außerhalb des MIT gefunden. Die Netzwerk-Umfeld und die Nutzung von Netzdiensten von Organisation zu Organisation verschieden sind, wurde in manchen Umgebungen Funktionalität benötigt, die in Kerberos Version 4 nicht enthalten ist. Auf der Basis von Vorschlägen und Erfahrungen, die aus der Nutzung von Kerberos V4 entstammten, wurde die Version 5 des Protokolls entwickelt. Im Jahre 1993 wurde KerberosV5 in RFC 1510 standardisiert[2].

## *1.2. Das Kerberos-Modell*

Das Kerberos-Protokoll ermöglicht Netzdiensten und -nutzern die sichere Identifikation ihrer Kommunikationspartner in einem unsicheren Netzwerk. Dazu initiiert der **Client** (die kontaktaufnehmende Partei) einen 3-Parteien-Nachrichtenaustausch, um dem **Server** (der kontaktierten Partei) seine Identität zu beweisen. Der Client authentifiziert sich dem Server gegenüber durch ein **Ticket**, das die Namen der beteiligten **Principals** (Nutzer und Dienste), einen **session-key** (einen für die Dauer der Kommunikationssitzung gültigen Schlüssel) sowie weitere Informationen, wie Gültigkeitsdaten und Adressen, enthält. Das Ticket erhält der Client von der dritten beteiligten Partei, dem **Key Distribution Center (KDC)**.

[1] Miller, S., Neuman, C., Schiller, J., and J. Saltzer, "Section E.2.1: Kerberos Authentication and Authorization System", M.I.T. Project Athena, Cambridge, Massachusetts, December 21, 1987.
[2] Kohl, J., Neuman, C., „The Kerberos Network Authentication Service (V5)“, RFC 1510, September 1993

Die Nutzung eines KDC zur Authentifizierung in Netzwerken wurde von NEEDHAM und SCHROEDER[3] vorgeschlagen. Kerberos benutzt das NEEDHAM-SCHROEDER-Protokoll unter Verwendung einiger Änderungen von DENNING und SACCO[4]. Das KDC besitzt das Vertrauen aller Principals des Netzwerks und speichert die geheimen Schlüssel aller Clients und Server in der **Principal-Datenbank**. Diese Schlüssel werden zur Kodierung der Tickets benutzt. Kerberos benutzt symmetrische Kryptographieverfahren zur Verschlüsselung von Daten, insbesondere wird in KerberosV4 ausschließlich der Data Encryption Standard (DES)[5] eingesetzt. Kerberos Version 5 bietet die Möglichkeit, weitere symmetrische Kryptosysteme, wie die sicherere DES-Variante Triple-DES, zu benutzen. Bedingt durch die Funktionsweise symmetrischer Kryptoalgorithmen ist die Entschlüsselung der Tickets nur mit dem zur Kodierung benutzten Schlüssel möglich. Die Kenntnis der geheimen Schlüssel bildet somit die Basis, auf der Clients und Server die Authentizität der erhaltenen Tickets annehmen.

In einem großen Netzwerk, das mehrere Organisationen verbindet, treten mit nur einem KDC Probleme auf. Mit zunehmender Anzahl von Principals wird es immer schwieriger, eine für alle Principals vertrauenswürdige Instanz zu finden, die das KDC beherbergt und administriert, da diese Instanz Zugriff auf alle geheimen Schlüssel und somit Zugriff auf alle geschützten Ressourcen hätte. Selbst bei Vorhandensein einer solchen vertrauenswürdigen Instanz muß auch das Vertrauen in die physische Sicherheit des Hosts, auf dem das KDC ausgeführt wird, gewährleistet sein. Bei der Kompromittierung der Principal-Datenbank wären alle geheimen Schlüssel betroffen. Aus diesen Gründen werden die Principals in einem Netzwerk in **Realms** unterteilt. Jede Realm unterhält ein eigenes KDC, das nur die Daten und Schlüssel der ihm zugeordneten Principals verwaltet. Es ist üblich, die Realm wie die Internet-Domain zu benennen und dabei nur Großbuchstaben zu verwenden.

Die Bezeichnung eines Principals besteht in KerberosV4 aus drei Komponenten, dem Namen, der Instanz und dem Realmnamen. Die Schreibweise für KerberosV4-Principals ist:

**NAME.INSTANZ@REALM**

---

[3] Needham, R., and M. Schroeder, "Using Encryption for Authentication in Large Networks of Computers", Communications of the ACM, Vol. 21 (12), pp. 993-999, December 1978.
[4] Denning, D., and G. Sacco, "Time stamps in Key Distribution Protocols", Communications of the ACM, Vol. 24 (8), pp. 533-536, August 1981.
[5] National Bureau of Standards, U.S. Department of Commerce, "Data Encryption Standard", Federal Information Processing Standards Publication 46, Washington, DC (1977).

Dabei sind alle drei Bestandteile auf eine Länge von 40 Zeichen beschränkt. Weiterhin ist die Verwendung des Zeichens „.“ nicht gestattet. KerberosV5 benutzt ein verändertes Namensschema, auf das später eingegangen wird.

### *1.3. Der Ticket-Nachrichtenaustausch*

Wenn ein Client C einen Server S nutzen möchte, sendet er eine Ticket-Anfrage an das KDC. Diese enthält die Namen von C und S. Das KDC überprüft das Vorhandensein der beiden Principals C und S in der Principal-Datenbank. Wenn beide Namen gültig sind, erzeugt das KDC ein Ticket mit C's und S's Namen, C's Netzwerk-Adresse ($C_{ADDR}$), der aktuellen Zeit T, der Lebensdauer des Tickets L und einem zufällig erzeugten session-key $K_{C,S}$:

$$T_{C,S} = ( C, S, C_{ADDR}, T, L, K_{C,S} )$$

Dieses Ticket wird dem geheimem Schlüssel von S ($K_S$) chiffriert, so daß nur S das Ticket entschlüsseln kann.

Die Antwort des KDC an C besteht aus dem verschlüsselten Ticket $\{T_{C,S}\}K_S$, S' Name, der aktuellen Zeit T, der Lebensdauer des Tickets L, und dem session-key $K_{C,S}$, alles mit C's geheimem Schlüssel $K_C$ kodiert:

$$\{S, T, L, K_{C,S}, \{T_{C,S}\}K_S \}K_C$$

*Bemerkung:* Mit der Schreibweise {T}K wird die Verschlüsselung des Klartextes T mit dem Schlüssel K bezeichnet.

C dechiffriert die Antwort unter Verwendung seines geheimen Schlüssels und erhält T, L, $K_{C,S}$ und $\{T_{C,S}\}K_S$. Durch die Verschlüsselung mit $K_S$ hat C keine Möglichkeit, daß Ticket $\{T_{C,S}\}K_S$ zu dechiffrieren. Das erhaltene Ticket wird für die spätere Nutzung im Ticket-Cache gespeichert.

Bevor C eine Nachricht an S sendet, erzeugt C einen **Authenticator** $A_C$, der aus C's Name, C's Adresse, der aktuellen Zeit T und einer von C gewählten Prüfsumme PS besteht.

Der Authenticator wird mit $K_{C,S}$ verschlüsselt:

$$A_C = \{C, C_{ADDR}, T, PS\}\ K_{C,S}$$

$A_C$ wird zusammen mit dem verschlüsselten Ticket $\{T_{C,S}\}K_S$ an S gesendet. Nach Empfang dieser Nachricht dechiffriert S das Ticket unter Benutzung von $K_S$. Dabei erhält S den session-key $K_{C,S}$. Mit diesem session key kann S nun den Authenticator entschlüsseln. Um die Authentizität von C festzustellen, vergleicht S den Inhalt des Tickets mit dem des Authenticators. Wenn die Inhalte übereinstimmen und die Zeiten T nicht mehr als einen definierten (geringen) Unterschied aufweisen, sieht S den Principal C als authentifiziert an.

Die folgende Abbildung zeigt den Ablauf der Authentifizierung:

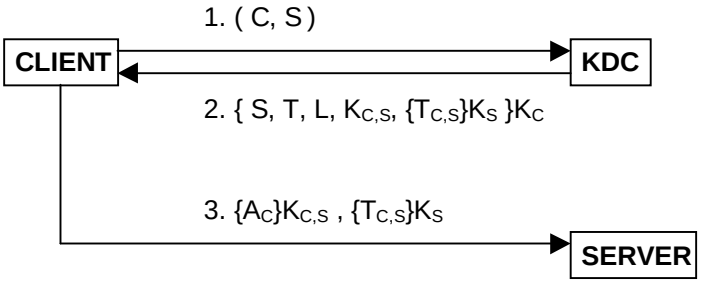



**Diagramm 1: Nachrichtenaustausch bei der Authentifizierung mit Kerberos**

### *1.4. Der Ticket-Granting-Service TGS*

Der unter 1.3. beschriebene Nachrichtenaustausch wird primär benutzt, um ein spezielles Ticket für den **Ticket-Granting-Server (TGS)** zu erhalten. Nach Erhalt dieses **Ticket-Granting-Ticket (TGT)** löscht der Client seine Kopie des geheimen Schlüssels $K_C$, wodurch das Risiko des Diebstahls des Schlüssels reduziert wird.

Das TGT wird von einem Teilprozeß des KDC, dem **Authentication Server (AS)** angefordert. Weitere Tickets für andere Services werden dann unter Benutzung des TGT vom Ticket-Granting-Server ausgestellt. Der verschlüsselte Teil dieser neuen **Service-Tickets** ist mit dem session-key aus dem TGT chiffriert, so daß der Client seinen geheimen Schlüssel $K_C$ nicht speichern muß. Mit dem vom TGT erhaltenen Service-Ticket authentifiziert sich der Client nun bei dem Server, mit dem er kommunizieren möchte.

TGS und AS sind Teile des KDC. Beide werden auf demselben Host ausgeführt und greifen auf dieselbe Datenbank von Principals und Schlüsseln zu.
Die vollständige Authentifizierung eines Principals verläuft nach folgendem Schema:

**Diagramm 2: Nachrichtenaustausch zum Erhalt eines TGT und eines Service-Tickets**

### *1.5. Inter-Realm-Authentifizierung*

Kerberos unterstützt die Authentifizierung zwischen Principals in verschiedenen Realms, d.h. ein Client in Realm A kann sich gegenüber einem Server in Realm B authentifizieren. Dies wird erreicht, indem beide KDC das jeweils andere KDC als Principal registriert haben und beiden Principals ein gemeinsamer geheimer Schlüssel zugeordnet wird. Dieser Schlüssel wird als **Inter-Realm Key** bezeichnet.

### *1.6. Änderungen in Kerberos Version 5*

Kerberos V5 ist eine beträchtliche Erweiterung der Version 4. Obwohl Kerberos V4 weite Verbreitung gefunden hat, wurde in einigen Organisationen erweiterte Funktionalität benötigt oder zumindest gewünscht. Weiterhin unterschieden sich die Netzumgebungen von der des Projektes „Athena“, zu dessen Sicherung Kerberos ursprünglich entwickelt wurde. Aus diesen Gründen begann man 1989 die Arbeit an der Version 5 des Kerberos-Protokolls. Im Jahre 1993 wurde Kerberos V5 im RFC 1510 standardisiert.

Die wichtigsten Änderungen betreffen die Bereiche:

**Verschlüsselung**

Um die Nutzung anderer symmetrischer Kryptoalgorithmen neben DES zu ermöglichen, werden verschlüsselte Abschnitte in Protokollnachrichten mit einem Typfeld versehen, das den Verschlüsselungsalgorithmus kennzeichnet. Somit kann der Empfänger der Nachricht die notwendige Entschlüsselungsroutine auswählen. Außerdem werden die geheimen Schlüssel mit einer Typ- und Längenkennzeichnung versehen, um unterschiedliche Variationen von Kryptosystemen (z.B. unterschiedliche DES-Modi) verwenden zu können.

**Nachrichtenkodierung**

KerberosV5 benutzt die Datenrepräsentationssprache Abstract Syntax Notation One (ASN.1)[6] in Verbindung mit den Basic Encoding Rules (BER)[7] zur Nachrichtenkodierung. Diese Kodierung wurde eingeführt, um Nachrichten flexibler gestalten zu können (keine feste Nachrichtenlänge) und Schwierigkeiten beim Byte-Ordering zu überwinden.

**Namenskonventionen**

Die Namenskonvention von Kerberos V4 wurde aufgrund der Einschränkungen dieser Konvention nicht in Kerberos Version 5 übernommen. Ein Principal besteht in Kerberos V5 aus zwei Komponenten, dem Namen und der Realm. Der Name enthält eine Typbezeichnung und zusätzlich eine beliebige Anzahl von frei wählbaren Textstrings. Das Instanzfeld der Version 4 wird durch einen Extra-Eintrag im Namensfeld ersetzt. Ein Principalname hat also folgenden allgemeinen Aufbau:

**TYP/STRING1/STRING2/.../STRINGN@REALMNAME**

[6] International Organization for Standardization, „Information Processing Systems – Open Systems Interconnection – Specification of Abstract Syntax Notation One (ASN.1)“, IS 8824 (Dec. 1987). First Edition.
[7] International Organization for Standardization, „Information Processing Systems – Open Systems Interconnection – Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)“, IS 8825 (November 1987). First Edition.

**Ticketformat**

Kerberos V5-Tickets haben ein erweitertes Format, das notwendige Änderungen zu Kerberos Version 4 ermöglicht. Das Ticket ist in einen unverschlüsselten und einen verschlüsselten Abschnitt unterteilt. Die Lebensdauer wird als Start- und Endezeit kodiert, was nahezu unbegrenzte Lebenszeiten für Tickets ermöglicht.

**Inter-Realm Authentifizierung**

Bereits Kerberos V4 unterstützt die Authentifizierung zwischen verschiedenen Realms (s.1.5). Dazu wird ein Inter-Realm-Key benötigt, der beiden KDC bekannt ist. Bei n verschiedenen Realms beträgt die Anzahl der notwendigen Schlüsselaustausche für eine vollständige gegenseitige Authentifizierung $O(n^2)$.
Kerberos V5 unterstützt die hierarchische Anordnung von Realms basierend auf den Bestandteilen des Realmnamens. Es entsteht ein **Realm-Baum**. Jedes KDC hält nur noch einen gemeinsamen Schlüssel mit dem KDC der übergeordneten Realm und allen KDCs von untergeordneten Realms im Baum. Die Anzahl der benötigten Schlüsselaustausche vermindert sich somit auf O(log n). Der Kontakt zu einer entfernten Realm wird hergestellt, indem der Realm-Baum traversiert wird, bis die Ziel-Realm erreicht ist.

**Neue Ticket-Optionen**

In Kerberos V5-Tickets ist ein Flag-Feld enthalten, das flexibler anwendbare Tickets ermöglicht. Tickets können als **erneuerbar (renewable)** mit zwei Ablaufzeiten gekennzeichnet werden. Das Ticket verliert seine Gültigkeit zum ersten Zeitpunkt, kann aber durch Präsentation an das KDC vor dem zweiten Zeitpunkt erneuert werden. Weiterhin können Tickets erst zu einem zukünftigen Zeitpunkt gültig werden (**postdated ticket**). Vor dem Zeitpunkt des Beginns der Gültigkeit ist das Ticket nicht verwendbar. In Kerberos V5 können weiterleitbare Tickets (**forwardable tickets**) beantragt werden. Der Client kann ein solches Ticket verwenden, um ein Ticket mit einer neuen Adresse zu beantragen, so daß das Ticket auf einem anderen Host genutzt werden kann.

## *1.7. Verschlüsselung in Kerberos*

### 1.7.1. Kryptosysteme in Kerberos

Zur Verschlüsselung werden in Kerberos symmetrische Kryptosysteme benutzt, d.h. die zu chiffrierenden Daten werden mit demselben Schlüssel verschlüsselt, der auch zur Dechiffrierung verwendet wird. KerberosV4 verwendet ausschließlich den Data Encryption Standard (DES), der mit einer Schlüssellänge von 64 Bit arbeitet, von denen 56 Bit zur Verschlüsselung benutzt werden. Kerberos V5 kann neben DES weitere symmetrische Kryptosysteme, wie Triple-DES (Schlüssellänge 168 Bit), benutzen. In jüngster Zeit gibt es auch Ansätze zur Integration von Public-Key-Kryptosystemen in Kerberos V5[8] [9]. Durch seine Flexibilität und Erweiterbarkeit wird Kerberos V5 auch in Zukunft in der Lage sein, ausreichende Sicherheit gegen Angriffe zu bieten.

### 1.7.2. Die String-to-Key – Funktion zur Schlüsselgenerierung

Bei einer Ticket-Anforderung eines Clients an das KDC verschlüsselt dieses die Antwort mit dem geheimen Schlüssel des Clients aus der Principal-Datenbank. Zur Dekodierung der Antwort benötigt der Client nun genau diesen Schlüssel. Da sich ein Client an wechselnden Hosts anmelden wird, ist eine permanente Speicherung des Schlüssels nicht praktikabel und sicher, da auf allen Hosts die Schlüssel aller Clients gespeichert werden müßten. Somit muß der Client den geheimen Schlüssel aus Wissen, Besitz oder biometrischen Eigenschaften generieren können. Die Nutzung von Besitz oder biometrischen Eigenschaften zur Schlüsselgenerierung wäre theoretisch möglich, allerdings unterstützt Kerberos V5 keine derartigen Systeme. Insbesondere der Einsatz von Chipkarten und entsprechenden Lesegeräten zusammen mit einem Paßwort wäre ein interessanter Ansatz zur Erhöhung der Sicherheit in einer Kerberos-Umgebung. Derzeit wird von Kerberos nur die Erzeugung des Schlüssels aus Wissen unterstützt. Da ein 8 Byte langer DES-Schlüssel für Menschen schwer zu merken ist, werden zur Gewinnung der geheimen Schlüssel Paßworte benutzt, die einfacher zu merken sind. Diese Paßworte werden durch eine sogenannte **String-to-Key-Funktion** in einen Schlüssel umgewandelt.

[8] Neumann, C., Tung, B., Wray, J., Medvinsky, A., Hur, M., Trostle, J., „Public Key Cryptography for Initial Authentication in Kerberos“, draft-ietf-cat-kerberos-pkinit-03.txt
[9] Neumann, C., Tung, B., Wray, J., Medvinsky, A., Hur, M., Trostle, J., „Public Key Cryptography for Initial Authentication in Kerberos“, draft-ietf-cat-kerberos-pkinit-03.txt

Die String-to-Key-Funktion ist eine Einwegfunktion, d.h. die Ermittlung des Paßwortes aus einem gegebenen Schlüssel ist nicht möglich.

In Kerberos V4 wird zur Generierung des geheimen Schlüssels nur das Paßwort benutzt:

$$K_{V4} = \text{stringtokey(Paßwort)}$$

Dieses Verfahren hat den Nachteil, daß bei einer Kompromittierung des Schlüssels eines Clients in einer Realm auch die Schlüssel des Clients in anderen Realms betroffen sind, wenn das gleiche Paßwort benutzt wurde. Da Nutzer dazu neigen, nur wenige verschiedene Paßworte zu verwenden, ist dies häufig der Fall.

Kerberos V5 benutzt deshalb außer dem Paßwort den gesamten Namen des Principals (der auch den Realmnamen enthält) als Argument für die String-to-Key-Funktion:

$$K_{V5} = \text{stringtokey(Paßwort, Principal)}$$

AFS verwendet eine dritte Version der String-to-Key Funktion. Diese nutzt das Paßwort und den AFS-Zellnamen als Funktionsparameter:

$$K_{AFS} = \text{stringtokey(Paßwort, Zellname)}$$

Aus dem gleichen Paßwort werden demnach in KerberosV4, KerberosV5 und AFS verschiedene geheime Schlüssel erzeugt.

### 1.7.3. Keytabs zur Speicherung von Server-Schlüsseln

Der kontaktierten Server benötigt den geheimen Schlüssel $K_S$, um das vom Client gesendete verschlüsselte Ticket $\{T_{C,S}\}K_S$ zu dekodieren. Der Schlüssel wird dazu auf dem Host, auf dem der Server ausgeführt wird, in einer Datei gespeichert, der sogenannten **Keytab**. Diese Datei muß durch geeignete Maßnahmen vor unbefugtem Zugriff geschützt werden (z.B. durch Nutzung von Zugriffsbeschränkungen des Dateisystems). Keytabs werden erzeugt, indem die Schlüssel der Server aus der Principal-Datenbank extrahiert werden. Für jeden Host, auf dem Serverprozesse ausgeführt werden, muß eine eigene Keytab generiert und lokal gespeichert werden.

### 1.7.4. Key version numbers

In Kerberos ist es selbstverständlich möglich, den Schlüssel eines Principals zu ändern. Dazu wird der in der Principal-Datenbank gespeicherte Schlüssel durch einen neuen Schlüssel ersetzt. Dieser kann aus einem Paßwort oder zufällig erzeugt werden. Bei der Änderung des Schlüssels kann es nun passieren, daß ein Client C ein Ticket für einen Server S im Ticket-Cache hält und S wärend der Gültigkeitsdauer des Tickets einen neuen Schlüssel benutzt. Bei der Authentifizierung von C am Server S könnte S das Ticket $\{T_{C,S}\}K_S$ nicht mehr entschlüsseln, da sich $K_S$ geändert hat.

Um dieses Problem zu lösen, werden in Kerberos V5 Schlüssel mit einer Versionsnummer versehen, der **Key-version number (kvno)**. Bei der Änderung eines Schlüssels wird ein neuer Schlüssel mit einer höheren kvno erzeugt und in der Principal-Datenbank abgelegt. Der veraltete Schlüssel wird nicht aus der Datenbank entfernt. Die kvno wird in allen Nachrichten zusammen mit dem Schlüssel übertragen, um die Verwendung der richtigen Schlüsselversion zu ermöglichen. Bei der Ticketgenerierung im KDC wird die jeweils höchste kvno zur Chiffrierung benutzt.

# 2. Migration von AFS-kaserver zu Kerberos V5-KDC

## *2.1. Unterschiede zwischen AFS-Kerberos und Kerberos V4*

Das Andrew File System (AFS), welches im Rechenzentrum der TU Chemnitz eingesetzt wird, benutzt Kerberos Version 4 zur Authentifizierung von Nutzern gegenüber den Fileservern. Die Aufgaben des KDC übernimmt dabei der AFS-kaserver. Die Implementation des Kerberos-Protokolls, das in AFS benutzt wird, erfolgte vor der Standardisierung von Kerberos V4. Daraus resultieren einige Unterschiede zwischen AFS-Kerberos V4 und Standard-Kerberos V4:

- AFS-Dienstprogramme (auch AFS-Clients genannt) benutzen das RX-Protokoll zur Kommunikation mit dem AFS-kaserver, Kerberos V4 benutzt das Kerberos-Protokoll.

- Der AFS-kaserver verwendet eine andere String-to-Key-Funktion zur Ermittlung von Schlüsseln aus Paßworten als Kerberos V4 (s.a. 1.7.2.).

- Bei der Authentifizierung an AFS verwirft der AFS-Client das TGT, nachdem er ein AFS-Service-Ticket erhalten hat. Deshalb können keine weiteren Service-Tickets mit diesem TGT beschafft werden. In Kerberos V4 wird ein TGT für eine gewisse Lebensdauer aufbewahrt und zur Anforderung mehrerer Service-Tickets genutzt.

- Das AFS-Service-Ticket wird nicht direkt zur Authentifizierung gegenüber dem Fileserver benutzt, sondern aus dem Service-Ticket und weiteren Daten wird eine Datenstruktur, das **AFS-Token,** erstellt, die der AFS-Cache-Manager zur Authentifizierung benutzt.

- Der Schlüssel zur Dekodierung des AFS-Tokens wird in der Datei „KeyFile“ gespeichert, nicht in der lokalen Keytab.

Aufgrund dieser Unterschiede läßt sich ein AFS-kaserver nicht als universelles Kerberos V4-KDC einsetzen. Besonders die Benutzung des RX-Protokolls und die veränderte String-to-Key-Funktion verhindern dies.

### *2.2. Authentifizierung an AFS bei Nutzung von AFS-kaserver*

Bei der Authentifizierung eines Nutzers gegenüber dem AFS-Service (durch „klog“ oder „AFS-login“) werden folgende Schritte durchlaufen:

1. Ein KerberosV4-TGT wird durch einen AFS-Client (klog) vom AFS-kaserver angefordert

2. Der AFS-kaserver überprüft, ob der anfordernde Principal (der Nutzer) in der Datenbank vorhanden ist und sendet das TGT, kodiert mit dem geheimen Schlüssel des Principals aus der AFS-kaserver-Datenbank.

3. Das Paßwort des Nutzers wird abgefragt. Aus dem eingegebenen Paßwort und dem AFS-Zellenname wird durch die AFS-String-to-key-Funktion ein Schlüssel berechnet.

4. Das Authentifizierungsprogramm („klog“, „AFS-login“) dekodiert das TGT unter Verwendung des unter 3. erzeugten Schlüssels. Nur bei Verwendung des richtigen Schlüssels liegt nach der Dechiffrierung ein gültiges Kerberos-Ticket vor.

5. Ein Service-Ticket für den AFS-Service (Principal **afs@<Zellenname>**) wird unter Benutzung des TGT vom AFS-kaserver angefordert.

6. Die Antwort des kaserver, die u.a. das AFS-Service-Ticket enthält, wird an den AFS-Client gesendet. Das AFS-Service-Ticket wird mit dem Schlüssel des AFS-Service aus der Principal-Datenbank verschlüsselt.

7. Das AFS-Dienstprogramm entschlüsselt die Antwort des AFS-kaserver mit Hilfe des session-key aus dem TGT. Das TGT wird danach verworfen und das AFS-Service-Ticket wird aus der Nachricht extrahiert.

8. Aus dem AFS-Service-Ticket und weiteren Daten, wie der Benutzer-ID, wird ein AFS-Token erzeugt und im Kernel installiert. Dieses Token benutzt der AFS-Cache-Manager zur Authentifizierung gegenüber dem AFS-Fileserver.

Zur Entschlüsselung des im Token enthaltenen AFS-Service-Ticket benutzt der Fileserver den im AFS-KeyFile gespeicherten Schlüssel, der deshalb mit dem in der Principal-Datenbank gespeicherten Schlüssel übereinstimmen muß.

Das folgende Diagramm veranschaulicht den Ablauf der Authentifizierung:

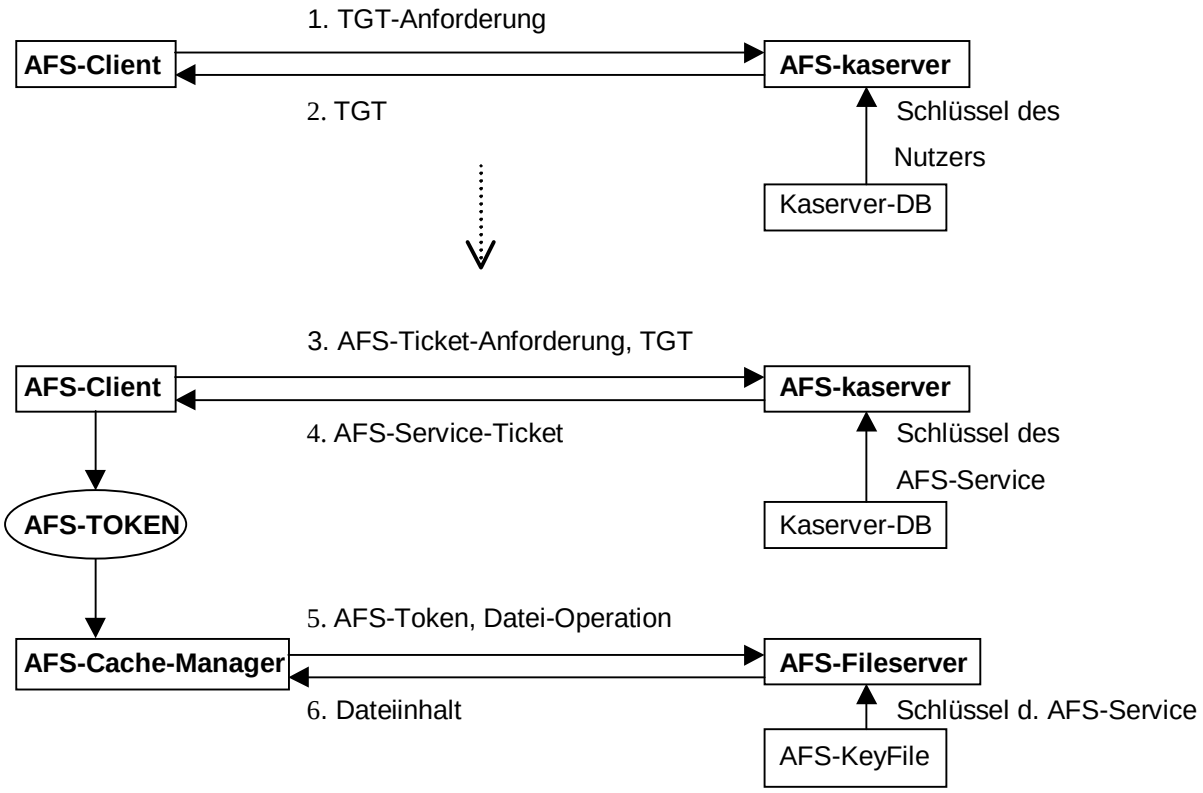



**Diagramm 3: Authentifizierung im AFS bei Nutzung von AFS-kaserver**

## *2.3. Unterschiede zwischen AFS-kaserver und KerberosV5-KDC*

Wenn die Authentifizierung in einer AFS-Zelle anstelle von AFS-kaserver durch Standard-Kerberos durchgeführt werden soll, ist es sinnvoll, die Version 5 einzusetzen. Obwohl AFS-kaserver und Kerberos V5 parallel betrieben werden können, sollte der AFS-kaserver durch das Kerberos V5-KDC ersetzt werden.

Für diese Vorgehensweise sprechen einige Gründe:

- Bei mehreren Principal-Datenbanken wird ein Mechanismus zum Abgleich des Inhalts der Datenbanken benötigt. Dieser Aufwand entfällt bei nur einer Principal-Datenbank.

- Kerberos V5 ist sicherer und flexibler einsetzbar als Kerberos V4 und bietet mehr Funktionalität, u.a forwardable und postdateable tickets.

- Nachrichten von Kerberos V4-Clients können von Kerberos V5-KDC‘s verstanden und verarbeitet werden, während Kerberos V4-KDC nur Nachrichten von Kerberos V4-Clients verstehen. Alle erhältlichen Kerberos V5-Implementationen bieten außerdem die Möglichkeit, Nachrichten von AFS-Clients, wie „klog“ oder „AFS-login“, zu verarbeiten, so daß diese Dienstprogramme weiterhin einsetzbar bleiben.

Um ein Kerberos V5-KDC als Ersatz für den AFS-kaserver einsetzen zu können, müssen die Unterschiede der beiden verwendeten Protokolle beachtet werden, da diese Unterschiede Maßnahmen zur Integration von AFS in Kerberos V5 erforderlich machen. Der AFS-kaserver und das von ihm verwendete modifizierte Kerberos V4-Protokoll unterscheidet sich in wichtigen Punkten von den von Kerberos V5 benutzten Gegenstücken.

Die AFS ka_* Bibliotheksfunktionen, die von AFS-Dienstprogrammen genutzt werden, kommunizieren mit dem AFS-kaserver unter Benutzung des RX-Protokolls. Kerberos V5-Clients nutzen hingegen das Kerberos V5-Protokoll, um mit dem KDC in Verbindung zu treten. Eine Implementation eines Kerberos V5-KDC muß daher das RX-Protokoll verstehen, um mit AFS-Clients kommunizieren zu können. Ist diese Unterstützung nicht implementiert, muß für die AFS-Dienstprogramme ein entsprechender Ersatz bereitgestellt werden.

Kerberos V5 verwendet verschiedene Programme zur Anforderung des TGT und des AFS-Service-Tickets. Das TGT wird im Gegensatz zu AFS nicht verworfen, sondern im Ticket-Cache gespeichert, um weitere Service-Tickets anfordern zu können. Außerdem haben AFS-TGT und AFS-Service-Ticket das Kerberos V4-Format, das sich vom Kerberos V5-Ticketformat unterscheidet. Da AFS-Clients nur Kerberos V4-Tickets auswerten können, müssen diese vom Kerberos V5- in das V4-Format umgewandelt werden, wenn sie von AFS-Dienstprogrammen verarbeitet werden sollen.

Wie schon in 1.7.2. erwähnt, wird in Kerberos V5 eine andere String-to-Key-Funktion als in AFS benutzt. Da AFS-Clients ein Paßwort nur mit der AFS-String-to-Key-Funktion in einen Schlüssel umwandeln können, muß genau dieser Schlüssel vom KDC zur Kodierung von Tickets benutzt werden. Kerberos V5-Clients sind hingegen in der Lage, den verwendeten Schlüsseltyp aus den Ticket-Informationen herauszulesen und so die angemessene String-to-Key-Funktion zu benutzen.

Ein weiterer Unterschied ist der Aufbau der Principal-Datenbank. Die AFS-kaserver-Datenbank hat ein anderes Format als die Kerberos V5-Datenbank. Daher ist das Vorhandensein eines Werkzeuges zur Konvertierung der AFS-kaserver-Datenbank in eine Kerberos V5-Datenbank notwendig, um bestehende AFS-Nutzer übernehmen zu können.

### *2.4. Authentifizierung an AFS bei Nutzung von Kerberos V5-KDC*

Wird der AFS-kaserver durch ein Kerberos V5-KDC ersetzt, so unterscheidet sich der Ablauf der Authentifizierung aufgrund der unter 2.3. aufgezeigten Unterschiede von Kerberos V5 und AFS deutlich gegenüber dem Ablauf in 2.2.:

1. Ein TGT wird vom KerberosV5-KDC angefordert, entweder von einem Kerberos V5-Client („kinit“), von einem Kerberos V4-Client oder von einem AFS-Client („klog“).

2. Das KDC überprüft das Vorhandensein des anfragenden (Nutzer-)Principals in der Kerberos V5-Datenbank und generiert das TGT. Dieses wird mit einem der geheimen Schlüssel des Principals aus der Datenbank verschlüsselt. Das Format des gesendeten TGT und der benutzte Schlüsseltyp hängt vom Typ der Anfrage ab. Kam diese von einem Kerberos V4- oder AFS-Dienstprogramm, wird das TGT vom Kerberos V5-Format in das Kerberos V4-Format umgewandelt es wird ein DES-Schlüssel benutzt. Bei Anfragen von Kerberos V5-Clients wird ein Kerberos V5-TGT versandt, das mit dem als am sichersten eingestuften Schlüssel in der Datenbank verschlüsselt wird.

3. Das Paßwort des Nutzers wird abgefragt. Das eingegebene Paßwort wird mit einer vom Client abhängigen String-to-Key-Funktion in einen Schlüssel umgewandelt. Kerberos V5-Dienstprogramme werten die in der Nachricht enthaltenen Informationen über den Schlüsseltyp aus und wählen die entsprechende String-to-Key-Funktion aus. AFS- und KerberosV4-Dienstprogramme benutzen immer die AFS- bzw. KerberosV4-String-to-Key-Funktion (s. 1.7.2). Zusätzlich benötigte Daten, wie der AFS-Zellenname, werden vom KDC zusammen mit dem Schlüssel versandt.

4. Das TGT wird unter Verwendung des unter 3. erzeugten Schlüssels dechiffriert. Bei Verwendung des richtigen Schlüssels liegt ein gültiges TGT vor und dieses wird für die weitere Benutzung im Ticket-Cache gespeichert.

5. Eine Anforderung für ein AFS-Service-Ticket wird unter Benutzung des TGT an den Kerberos V5-TGS gesandt. Die Anforderung kann von AFS-Clients („klog“), von Kerberos V4-Clients, als auch von Kerberos V5-Clients („afslog“) gesendet werden.

6. Der TGS dekodiert die Anforderung und generiert ein AFS-Service-Ticket, das mit dem geheimen Schlüssel des AFS-Service-Principals verschlüsselt wird. Das AFS-Service-Ticket wird in das Kerberos V4-Format umgewandelt, wenn die Anfrage von einem AFS- oder Kerberos V4-Client kam.

7. Die Antwort des TGS wird entschlüsselt und das AFS-Service-Ticket wird zur Generierung eines AFS-Tokens benutzt (das AFS-Service-Ticket bildet den verschlüsselten Teil des AFS-Tokens). Das Token vom AFS-Cache-Manager zur Authentifizierung gegenüber dem AFS-Fileserver benutzt. Der AFS-Fileserver benutzt zur Entschlüsselung des AFS-Tokens den Schlüssel aus dem AFS-KeyFile. Dieser muß deshalb mit dem Schlüssel in der KerberosV5-Datenbank übereinstimmen.

Der geschilderte Ablauf wird in Diagramm 4 dargestellt:

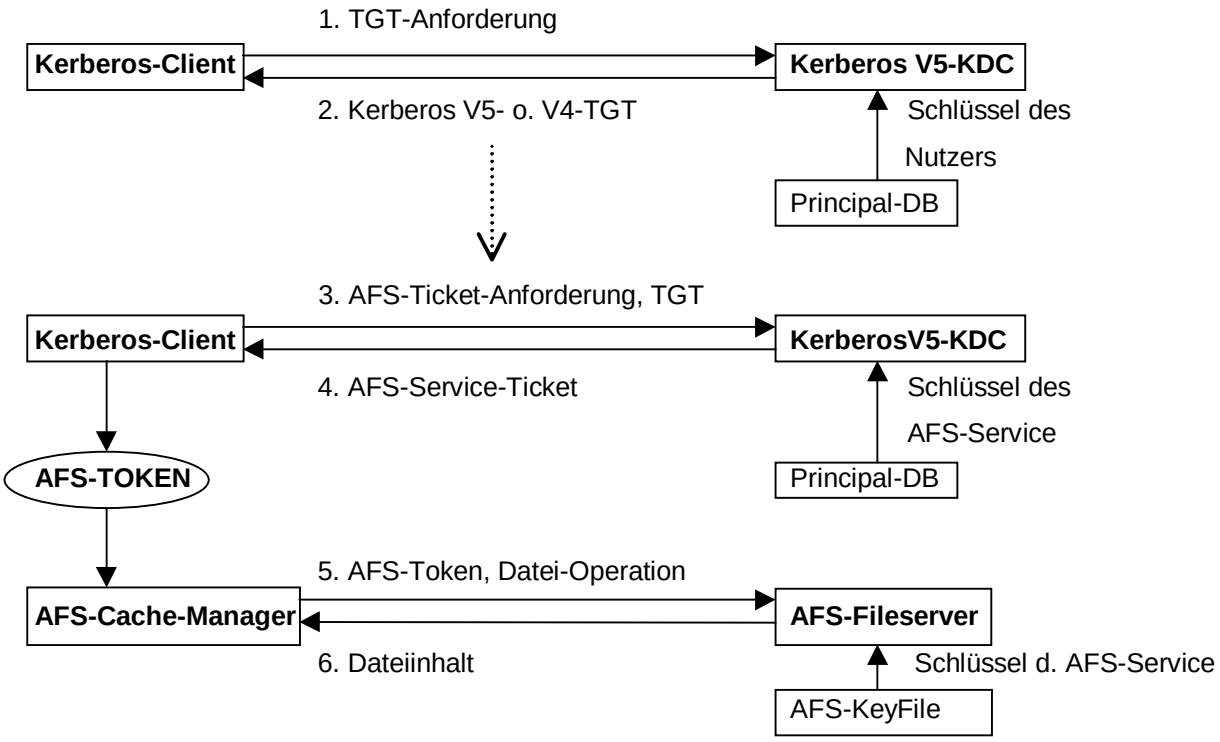



**Diagramm 4: Authentifizierung an AFS mit KerberosV5-KDC**

## *2.5. Voraussetzungen zur Integration von AFS in Kerberos V5*

### 2.5.1. Konvertierung der AFS-kaserver-Datenbank

Um die bestehenden AFS-Nutzer nicht neu in der Principal-Datenbank anlegen zu müssen, ist es notwendig, die AFS-kaserver-Datenbank in ein vom Kerberos V5-KDC lesbares Format zu überführen. Da die Neuanlage der Nutzer-Principals in großen Umgebungen, wie dem Rechenzentrum der TU Chemnitz, nicht oder nur mit erheblichem Aufwand machbar ist, stellt das Vorhandensein eines Konvertierungswerkzeuges eine wesentliche Voraussetzung für die Migration von AFS-kaserver zu Kerberos V5-KDC im URZ dar. Sowohl in „Heimdal“ als auch im „AFS-Migration-Kit“ sind Werkzeuge enthalten, die die Konvertierung der AFS-Datenbank ermöglichen. Dabei ist das im „AFS-Migration-Kit“ enthaltene Werkzeug „afs2k5db“ erheblich komfortabler und flexibler als die in „Heimdal“ enthaltenen. Es erlaubt die Auswahl der zu konvertierenden Principals und generiert eine vom Datenbank-Administrationstool lesbare Dump-Datei. „Heimdal“ bietet hingegen kein einzelnes Werkzeug für die Konvertierung von kaserver-Datenbanken, allerdings lassen sich Dienstprogramme für andere Aufgaben zur Konvertierung „zweckentfremden“. Ein Problem bei der Nutzung dieser Werk-

zeuge ist die vollständige Konvertierung der kaserver-Datenbank, d.h. alle Principals werden konvertiert, eine Auswahl einzelner Principals ist nicht möglich. Da es nicht sinnvoll ist, andere als die Nutzer-Principals zu migrieren, muß ein Weg gefunden werden, nur einen Teil der konvertierten Datenbank in die bestehende Datenbank zu integrieren. Dies ist in „Heimdal“ möglich, allerdings mit zusätzlichem Aufwand verbunden (s. 4.4.4.).

Die in der AFS-kaserver-Datenbank gespeicherten Schlüssel wurden mit der AFS-String-to-Key-Funktion erzeugt. Das Konvertierungswerkzeug ist nur in der Lage, vorhandene Schlüssel die neue Datenbank zu kopieren, es kann jedoch die Schlüssel nicht neu erzeugen, da die verwendeten Paßworte nicht bekannt sind. Deshalb werden in der konvertierten Datenbank AFS-Schlüssel (DES-Schlüssel von AFS-String-to-Key-Funktion) angelegt. Die Kerberos V5-Dienstprogramme müssen also in der Lage sein, den zur Ticketchiffrierung benutzten Schlüsseltyp zu erkennen, um die richtige String-to-Key-Funktion bei der Dechiffrierung auszuwählen. Das ist in Kerberos V5 standardmäßig der Fall, da das KDC Schlüsseltyp und -länge sowie zusätzliche zur Schlüsselgenerierung notwendige Daten (wie den AFS-Zellenname) zusammen mit dem Schlüssel speichert. Bei einem Nachrichtenaustausch Client ←→ KDC werden diese Informationen zum Client übertragen, der somit in der Lage ist, die richtige String-to-Key-Funktion mit den notwendigen Daten zu benutzen. AFS-Clients benutzen immer die AFS-String-to-Key-Funktion zur Schlüsselgenerierung. Sowohl Kerberos V5- als auch AFS-Clients sind also fähig, die Tickets für konvertierte Nutzer zu entschlüsseln.

Probleme treten bei der Änderung von Paßworten auf. Das Dienstprogramm zur Paßwortänderung muß sicherstellen, daß für den AFS-Nutzer ein neuer AFS-Schlüssel erzeugt wird, da AFS-Dienstprogramme sonst keine Tickets mehr dekodieren können.

### 2.5.2. Abwärtskompatibilität zu AFS und Kerberos V4

Um vorhandene AFS-Dienstprogramme, wie „klog“ oder „AFS-login“, weiterbenutzen zu können, muß das Kerberos V5-KDC auf Ticketanforderungen über das RX-Protokoll reagieren können. Ist diese Unterstützung nicht implementiert, müssen alle AFS-Clients durch KerberosV5-Dienstprogramme ersetzt werden. Im „Heimdal“-KDC ist eine AFS-kaserver-Emulation implementiert, so daß das KDC Authentifizierungsanforderungen über das RX-Protokoll verarbeitet. Weiterhin ist als Ersatz für „klog“ das Dienstprogramm „afslog“ vorhanden, welches unter Benutzung eines KerberosV5-TGT ein AFS-Service-Ticket anfordert und daraus ein AFS-Token erzeugt.

Da sowohl das TGT als auch das AFS-Service-Ticket bei einem Kerberos V5-KDC angefordert werden, generiert das KDC Tickets im Kerberos V5-Format. Kam die Ticketanfrage von Kerberos V4- oder AFS-Clients, müssen die Tickets in das Format von Kerberos V4 umgewandelt werden, da diese Clients nur das V4-Format verarbeiten können. Die Umwandlung kann entweder direkt im KDC erfolgen (bei „Heimdal“) oder von einem Umwandlungsserver vollzogen werden („AFS-Migration-Kit“). Eine Konsequenz der Umwandlung ist, daß nach einer Authentifizierungsanfrage eines AFS-Clients kein gültiges TGT im Kerberos V5-Format vorliegt, das zu weiteren Ticketanforderungen benutzt werden kann. Bei der Nutzung von Dienstprogrammen des KerberosV5-Pakets (z.B. „afslog“ in „Heimdal“) besteht diese Einschränkung nicht.

### 2.5.3. Abstimmung des AFS-KeyFile mit der KerberosV5-Datenbank

Das AFS-Service-Ticket wird vom TGS mit dem geheimen Schlüssel des AFS-Server-Principals (afs@<ZELLENNAME>) verschlüsselt. Bei der Nutzung des AFS-Service-Tickets als Bestandteil des AFS-Tokens durch den AFS-Fileserver dekodiert dieser das Ticket mit dem im AFS-KeyFile gespeicherten Schlüssel. Der Schlüssel des AFS-Service in der Principal-Datenbank muß deshalb mit dem im AFS-KeyFile gespeicherten Schlüssel übereinstimmen. Es sind zwei Möglichkeiten vorstellbar, diese Gleichheit zu erreichen:

(a) Im AFS-KeyFile wird ein zunächst neuer Schlüssel angelegt (durch „bos“). Dieser Schlüssel wird dann in die KerberosV5-Datenbank überführt.

(b) In der Kerberos V5-Datenbank wird ein zufälliger Schlüssel für den AFS-Service erzeugt. Dieser Schlüssel wird dann mit einem geeigneten Werkzeug in das AFS-KeyFile kopiert.

Der erste Ansatz erfordert etwas mehr Aufwand als der zweite, da die Datenbank manipuliert werden muß. In „Heimdal“ kann allerdings nur (a) umgesetzt werden, da das für (b) benötigte Werkzeug nicht vorhanden ist. Das „AFS-Migration-Kit“ bietet ein solches Werkzeug an, so daß mit ihm beide Möglichkeiten benutzt werden könnten. Dieses Werkzeug „asetkey“ kann Einträge des AFS-Keyfile anzeigen und Schlüssel aus der Kerberos-Keytab in das AFS-KeyFile kopieren. In der „Heimdal“-Umgebung funktioniert „afs2k5db“ nicht, da sich die Formate der Keytabs von MIT-Kerberos und „Heimdal“ unterscheiden.

### *2.6. Schritte zur Migration von AFS-kaserver zu Kerberos V5-KDC*

Um den AFS-kaserver durch ein Kerberos V5-KDC zu ersetzen, sind unabhängig von der verwendeten Implementation folgende Schritte durchzuführen:

1. Installation von Kerberos V5 auf einem oder mehreren Hosts, auf denen das KDC ausgeführt wird (**Datenbankserver)**. Einer der Datenbankserver wird **Master-Server**, alle Änderungen an der Datenbank werden auf ihm ausgeführt. Vom Master-Server wird die Datenbank an die untergeordneten **Slave-Server** verteilt, so daß alle Server die gleiche Datenbank verwenden.

2. Hinzufügen des AFS-Principals **afs@<ZELLENNAME>** zur Principal-Datenbank und Sicherstellung der Gleichheit des Schlüssels dieses Principals mit dem Schlüssel im AFS-KeyFile (auch die kvno muß übereinstimmen, s. 2.5.3.).

3. Konvertierung der AFS-kaserver-Datenbank und Integration der konvertierten Datenbank in die bestehende Kerberos V5-Datenbank (s. 2.5.1.).

4. Installation der im Kerberos-Paket enthaltenen Kerberos V5-kompatiblen Server (ftpd, telnetd, ...) auf den entsprechenden Netzknoten. Die Schlüssel der installierten Server müssen in der Keytab des Hosts gespeichert werden. Diese Server arbeiten sowohl mit Clients, die Kerberos zur Authentifizierung benutzen, als auch mit „Nicht-Kerberos“-Clients zusammen.

5. Installation der inm Kerberos-Paket enthaltenen Kerberos V5-kompatiblen Clients (kinit, ftp, telnet, ...) und der Konfigurationsdateien auf allen Netzknoten der Kerberos-Realm.

6. Anpassung der Werkzeuge zur Einrichtung, Löschung und Modifikation von Nutzern. Die AFS-Werkzeuge „kas“ und „uss“ können nicht mit dem „Heimdal“-KDC benutzt werden, da die von ihnen gesendeten RX-Nachrichten nicht verarbeitet werden. Für diese Dienste muß ein Ersatz bereitgestellt werden. Die Funktionen von „kas“ und „uss“ können in „Heimdal“ von ***kadmin*** und übernommen werden.

Die aufgeführten Schritte und die dabei notwendigen Arbeiten werden am Beispiel von „Heimdal“-Kerberos V5 in den Kapiteln 3. und 4. detailliert beschrieben.

# 3. Kerberos Version 5 – „Heimdal“

## *3.1. Allgemeines*

Das Softwarepaket „Heimdal“[10] [11] wird seit 1996 an der Kungliga Tekniska Högskolan (KTH) Stockholm, Schweden entwickelt. „Heimdal“ ist eine Implementation des Kerberos-Protokolls Version 5 unter Zugrundelegung der Standardisierung von Kerberos V5 in RFC 1510. Das Paket ist frei verfügbar und unterliegt keinen Lizenz- und Nutzungsbeschränkungen. Insbesondere ist auch der gesamte Quellcode frei verfügbar. An der KTH wurde auch eine frei verfügbare Implementation von Kerberos V4[12] entwickelt. Die Include-Dateien und Bibliotheken dieser Kerberos V4-Implementation werden zur Übersetzung von „Heimdal“ benötigt, wenn KerberosV4- und AFS-Clients unterstützt werden sollen.

Bei den Untersuchungen im Rahmen dieser Studienarbeit wurde „Heimdal“ verwendet, da es zur Zeit die einzige außerhalb der USA legal einsetzbare Kerberos V5-Implementation ist.

## *3.2. Installation von „Heimdal“ und Konfiguration einer Realm*

### 3.2.1. Die Testumgebung im URZ

Zum Test von „Heimdal“ und der Integration von AFS in Kerberos V5 stellte das URZ den Rechner „otroc.hrz.tu-chemnitz.de“ zur Verfügung. Auf diesem Rechner wurde eine AFS-Zelle mit dem Zellenname „kerberos5.tu-chemnitz.de“ eingerichtet. Anschließend wurde „Heimdal“ installiert und eine Realm mit dem Namen „KERBEROS5.TU-CHEMNITZ.DE“ eingerichtet.

### 3.2.2. Installation von „Heimdal“

„Heimdal“ kann als Quellcode-Archiv heimdal-*x*.tar.gz bezogen werden, wobei x für die jeweils aktuelle Versionsnummer steht (z. Zt. „0.0t“ ). Zur Übersetzung werden einige spezielle Programme und Bibliotheken benötigt:

- lex oder flex
- awk

[10] http://www.pdc.kth.se/heimdal/
[11] ftp://ftp.pdc.kth.se/heimdal/src/
[12] http://www.pdc.kth.se/kth-krb/

- yacc oder bison
- NDBM oder Berkeley DB

Die Anpassung an die vorhandene Systemumgebung wird durch GNU autoconf und GNU automake vorgenommen. Zu Beginn der Installation wird das Script **configure** gestartet, das eine Reihe von systemabhängigen Variablen belegt. Um die Unterstützung für AFS- und KerberosV4-Clients einzuschalten, muß eine Kerberos V4-Implementation der KTH zur Verfügung stehen, da Include- und Bibliotheksdateien dieses Pakets benötigt werden. Weiterhin muß **configure** mit einigen Optionen gestartet werden:

-- *with-krb4 = Pfad zu KTH-KerberosV4 (/usr/athena/)*

-- *enable-kaserver*

Diese Option schaltet die AFS-kaserver-Emulation im KDC ein.

-- *enable-kaserver-db*

Ermöglicht das Einlesen von AFS-kaserver-Datenbanken in **hprop**.

Nach der Ausführung von **configure** sind in allen Unterverzeichnissen die für **make** notwendigen Makefiles generiert. Unter Solaris 2.x kommt es dabei zu einem Fehler im Makefile des Verzeichnisses /kdc, der durch **automake** verursacht wird. Es werden dabei einige Sonderzeichen im Makefile angelegt, die **make** nicht interpretieren kann. Diese Zeichen können manuell entfernt werden. Unter Linux tritt dieser Fehler nicht auf. Mit **make** wird das gesamte Paket übersetzt und durch **make install** im Pfad */usr/heimdal/* installiert.

### 3.2.3. Die Bestandteile von „Heimdal“

Das Paket „Heimdal“ besteht aus den Teilen:

**1. KDC**

Das KDC wird auf allen Datenbank-Servern ausgeführt und wickelt alle Ticket- und Authorisierungsanfragen ab.

## 2. KerberosV5-kompatible Server und Server zur Administration

- der Adminserver ***kadmind***

  Dieser Server dient im Zusammenspiel mit ***kadmin*** zum Zugriff auf die Principal-Datenbank von entfernten Hosts aus. Er wird nur auf dem Master-Server ausgeführt.
- der Paßwort-Änderungsserver ***kpasswdd***

  kpasswdd wird zusammen mit kpasswd zur Änderung von Nutzer-Paßworten verwendet. Dieser Server wird nur auf dem Master-Server ausgeführt.
- der Datenbank-Propagierungsserver ***hpropd***

  Dieser Server empfängt die Principal-Datenbank vom Master-Server und legt diese auf dem Slave-Server an. Er muß deshalb auf allen Slave-Servern ausgeführt werden.
- Pop3-Server ***popper***

  ein POP3-Server mit Kerberos V5-Unterstützung (KPOP).
- der FTP-Server ***ftpd***

  ein FTP-Server mit Kerberos V5-Unterstützung. Dieser FTP-Server arbeitet auch mit FTP-Clients zusammen, die nicht Kerberos zur Authentifizierung benutzen.
- der Telnet-Server ***telnetd***

  ein Telnet-Server mit KerberosV5-Unterstützung. Dieser Telnet-Server arbeitet auch mit Telnet-Clients zusammen, die nicht Kerberos zur Authentifizierung benutzen.

Die Server werden bei der Installation im Pfad */usr/heimdal/libexec* angelegt.

## 3. Administrationswerkzeuge

- das Datenbankverwaltungs-Werkzeug ***kadmin***

  Dieses Programm dient zur Administration der Principal-Datenbank. Mit ihm werden alle Änderungen an der Datenbank, wie Anlage, Modifikation und Löschung von Principals, durchgeführt.
- das Keytab-Verwaltungsprogramm ***ktutil***

  Mit ***ktutil*** werden Einträge in Keytabs verwaltet. Diese können z.B. angezeigt oder gelöscht werden.

- ***kstash*** zur Generierung eines Master-Key zur Datenbankverschlüsselung

  Dieses Werkzeug erzeugt einen Master-Key für die Verschlüsselung der Principal-Datenbank und legt ihn in der Datei */var/heimdal/m-key* ab

Die Administrationswerkzeuge werden im Pfad */usr/heimdal/sbin* installiert.

**4. KerberosV5-Clients**

- ***kinit*** zur Anforderung eines TGT vom KDC

  Mit ***kinit*** wird ein TGT angefordert und im Ticket-Cache gespeichert.
- ***klist*** zur Anzeige vorhandener Tickets

  ***klist*** zeigt alle Tickets des Principals an (TGT und Service-Tickets).
- ***afslog*** zur Generierung eines AFS-Tokens

  Dieses Programm fordert unter Benutzung eines vorhandenen TGT ein AFS-Ticket an und generiert daraus ein AFS-Token.
- ***kdestroy*** zur Zerstörung aller vorhandenen Tickets
- ***string2key*** zur Generierung von Schlüsseln aus Paßworten (zum Test).

  Mit string2key kann die String-to-Key-Funktion von Kerberos V5, Kerberos V4 und AFS zur Generierung von Schlüsseln ausgeführt werden.
- ein FTP-Client ***ftp***

  Dieser FTP-Client unterstützt KerberosV5-Authentifizierung im Zusammenspiel mit dem FTP-Server aus dem „Heimdal“-Paket.
- ein Telnet-Client ***telnet***

  Der Telnet-Client unterstützt Kerberos-Authentifizierung, wenn er mit dem Telnet-Server von „Heimdal“ kommuniziert.

Die Clientdienstprogramme werden im Pfad */usr/heimdal/bin* installiert.

**5. Bibliotheken** - im Verzeichnis */usr/heimdal/lib*.

**6. Includedateien** - *im Verzeichnis /usr/heimdal/include*.

**7. Dokumentation** - */usr/heimdal/doc* und */usr/heimdal/man*.

### 3.2.4. Das KDC von „Heimdal“

Das KDC spielt die zentrale Rolle in jeder Kerberosinstallation. Es wickelt alle eingehenden Authentifizierungsanfragen in der jeweiligen Realm ab und greift dabei direkt auf die Principal-Datenbank zu. Das Verhalten des KDC von „Heimdal“ wird über eine Konfigurationsdatei gesteuert, */etc/krb5.conf*. Beim Start des KDC werden aus dieser Datei Steuerungsinformationen und –optionen ausgelesen. *Krb5.conf* ist hierarchisch strukturiert und in Sektionen unterteilt. In jeder Sektion werden eine oder mehrere Variablenbindungen festgelegt. Eine umfassende Erläuterung der Konfigurationsoptionen und die auf *otroc* und die Testumgebung angepaßte Konfigurationsdatei als Anhang enthalten.

Das KDC wird auf den Datenbankservern der Realm als Hintergrundprozeß ausgeführt und sollte beim Systemstart automatisch geladen werden. Der KDC-Prozeß bindet sich nach dem Start an mehrere TCP- und UDP-Ports:

- Port 88, TCP und UDP – Well-known-Port für KerberosV5-Authentifizierung.
- Port 750, TCP und UDP – Well-known-Port für Kerberos4-Authentifizierung.
- Port 7004, nur UDP – Port für AFS-kaserver-Kommunikation.

Auf diesen Ports erwartet das KDC Authentifizierungsanfragen von Clients. Das KDC enthält als Bestandteile den Authentication Server (AS) und den Ticket-Granting-Server (TGS). Die Funktionen dieser Server wurden in 1.4. beschrieben. Das KDC protokolliert alle Ticket-Anfragen in der Datei */var/heimdal/kdc.log*. Der Speicherort dieser Datei kann in krb5.conf angepaßt werden. Die Log-Datei bietet wichtige Informationen zur Fehlersuche und –behebung.

Das „Heimdal“-KDC kann sowohl Nachrichten von Kerberos V5- und Kerberos V4-Clients verarbeiten, als auch Nachrichten von AFS-Dienstprogrammen. Dabei werden nur zwei der drei Nachrichtentypen von AFS-Clients verarbeitet, **KA_AUTHENTICATION_SERVICE** und **KA_TICKET_GRANTING_SERVICE**. Der Nachrichtentyp **KA_MAINTENANCE_SERVICE** wird ignoriert.

Weiterhin werden nur die Operationen **AUTHENTICATE** und **GETTICKET** ausgeführt, andere Operationen, wie **CHANGEPASSWORD**, **CREATEUSER, GETRANDOMKEY**

wurden nicht implementiert. Diese teilweise Unterstützung für AFS-Clients reicht aus, um mit AFS-Dienstprogrammen zur Authentifikation, wie „klog“ oder „AFS-Login“ zu interagieren. Andere AFS-Dienstprogramme zur Datenbankverwaltung, wie „kas“ oder „uss“, können jeoch nicht mit einem „Heimdal“-KDC zusammenarbeiten. Werden diese Dienstprogramme benutzt, muß ein Ersatz bereitgestellt werden bzw. muß deren Funktionalität von „Heimdal“-Diensten übernommen werden.

### 3.2.5. Die Erstellung einer neuen Realm

Bei der Installation einer neuen Realm mit „Heimdal“ wird zuerst die Principal-Datenbank im Pfad */var/heimdal/heimdal.db* erzeugt. Diese Datenbank kann insgesamt mit einem **Master-Key** verschlüsselt werden. Zum Anlegen des Master-Key dient das Programm ***kstash***, das den Schlüssel in der Datei *var/heimdal/m-key* ablegt. Die Verschlüsselung der Datenbank ist sinnvoll, da bei einem Diebstahl der Principal-Datenbank, z.B. während einer Übertragung über das Netzwerk, das Auslesen geheimer Schlüssel aus der Datenbank verhindert wird.

Zur Administration der Datenbank dient das Programm ***kadmin***. Nach Start von ***kadmin*** kann mit dem Kommando **init REALMNAME** eine neue Realm initialisiert werden. Dabei werden einige für Kerberos notwendige Principals zur Datenbank hinzugefügt und geheime Schlüssel für diese erzeugt:

| | |
|---|---|
| **krbtgt/<REALMNAME>@<REALMNAME>** | Der Ticket-granting-Server TGS |
| **kadmin/admin@<REALMNAME>** | Der Admin-Server (kadmind) |
| **kadmin/changepw@<REALMNAME>** | Der Paßwort-Änderungs-Server |
| **default@<REALMNAME>** | ein default-Principal |

Zu beachten ist, daß ***kadmin*** für jeden Principal einen DES- und einen Triple-DES-Schlüssel in der Datenbank anlegt. Dieses Verhalten kann im derzeitigen Entwicklungsstand der Software nicht beeinflußt werden. Es kann weder das Anlegen eines weiteren Schlüsseltyps erzwungen werden, noch kann ein Schlüsseltyp von der Anlage ausgeschlossen werden.

Für jeden Server, der Kerberos zur Authentifizierung nutzt, muß ein Service-Principal in der Datenbank angelegt werden. Der Name des Principals setzt sich aus der Bezeichnung des Servers und dem Hostnamen zusammen, beispielsweise wird für den FTP-Server auf dem Host „otroc“ der Principal **ftp/otroc.hrz.tu-chemnitz.de@<REALMNAME>** angelegt.

Für Server, die einen Zugriff auf den Host bieten, wie telnet, rlogin oder rsh, wird statt der Bezeichnung des Servers der Name **host** verwendet. Es ist zu beachten, daß gleiche Server auf verschiedenen Hosts als verschiedene Principals betrachtet werden, d.h. es muß für jeden Host und jeden Server ein neuer Principal angelegt werden.

Das Anlegen eines Service-Principals wird in ***kadmin*** durch das Kommando **add –-random <Principalname>** bewirkt. Durch die Option „-- random“ werden zufällige Schlüssel erzeugt und es wird kein Paßwort abgefragt. Der Schlüssel muß dem zugehörigen Server bei der Authentifizierung eines Clients lokal zur Verfügung stehen, um das Ticket des Clients entschlüsseln zu können. Deshalb wird der Schlüssel des Servers aus der Datenbank in eine Keytab „extrahiert“ und diese Keytab auf dem Host, auf dem der Server ausgeführt wird, installiert.

### 3.2.6. Erzeugen von Keytabs

Um einen Schlüssel eines KerberosV5-Servers in eine Keytab zu speichern, dient in ***kadmin*** der Befehl **ext <Principalname>**. Voraussetzung für die Extraktion ist die Existenz des Principals in der Datenbank. Bei der Anwendung von **ext** werden alle für den Principal in der Datenbank gespeicherten Schlüssel in die Keytab extrahiert. Diese steht im Pfad */etc/krb5.keytab*. Die Keytab kann mit dem Dienstprogramm ***ktutil*** angezeigt und bearbeitet werden.

### 3.2.7. Test der Grundinstallation

Zum Test der Installation ist es notwendig, mit ***kadmin* add** einen Test-Principal zu erzeugen. Für diesen kann mit ***kinit* <*Testnutzer*>** ein TGT beantragt werden. Der Inhalt des Ticket-Cache, in dem alle erhaltenen Tickets gespeichert werden, kann mit dem Programm ***klist*** angezeigt werden. War ***kinit*** erfolgreich, erzeugt ***klist*** folgende Ausgabe:

*Credentials cache: krb5cc_xxx*
*Principal: testnutzer@<REALMNAME>*

| *Issued* | *Expires* | *Principal* |
|---|---|---|
| *Sep 17 13:02:01* | *Sep 17 23:02:01* | *krbtgt/<REALMNAME>@<REALMNAME>* |

Das erhaltene TGT kann nun zur Authentifizierung gegenüber Kerberos-kompatiblen Servern, wie „Heimdal“-FTP oder -Telnet benutzt werden. Dazu startet der Testnutzer einen Kerberos-Client, beispielsweise ***ftp***, und meldet sich an einem entfernten Host an, auf dem der entsprechende Kerberos-Service ausgeführt wird. Bei erfolgreicher Authentifizierung wird das Paßwort des Nutzers nicht abgefragt, ist kein TGT vorhanden, wird die herkömmliche Paßwort-Authentifikation benutzt.

# 4. Integration von „Heimdal“ in AFS

## *4.1. Ziele*

Ziel der Integration von AFS in Kerberos V5 sollte es sein, den Übergang von AFS-kaserver zu Kerberos V5-KDC so transparent wie möglich zu gestalten und gleichzeitig die zusätzlichen Möglichkeiten von Kerberos V5 nutzbar zu machen. Für die Nutzer des URZ bedeutet Transparenz, daß ihre Paßworte gültig bleiben und die bisher benutzten Dienstprogramme zur Authentifizierung weiterhin nutzbar bleiben. Für das URZ als Betreiber der AFS-Zelle steht ein kleinstmöglicher Aufwand zur Umstellung, geringe Ausfallzeit des AFS-kaserver bei der Umstellung, die Nutzung erweiterter Funktionen von Kerberos V5 und die Sicherheit der AFS-Umgebung im Vordergrund.

## *4.2. Bestehende Infrastruktur im URZ*

Vom URZ der TU-Chemnitz wird eine einzelne AFS-Zelle „tu-chemnitz.de“ betrieben. In dieser Zelle werden drei Datenbankserver benutzt:

**zuse.hrz.tu-chemnitz.de**
**aetius.hrz.tu-chemnitz.de**
**phoenix.hrz.tu-chemnitz.de**

Mehrere Datenbankserver bieten Vorteile im Hinblick auf Ausfallsicherheit und Leistung. Allerdings muß die Kerberos-Datenbank zwischen den Datenbankservern kohärent gehalten werden, was einen Mechanismus zur Datenbankverteilung erforderlich macht.

Die Authentifizierung an AFS, also die Beschaffung eines AFS-Tokens, wird im URZ durch verschiedene Mechanismen erreicht:

- Ein Login-Dienst mit AFS-Unterstützung („AFS-Login“).
- Das AFS-Dienstprogramm „klog“.
- Eine modifizierte Version der „Secure Shell“ (SSH), die AFS-Token anfordern und weiterleiten kann.
- PAM-Module für die Authentifizierung an AFS und Kerberos V4

Ein Ziel der Migration zu Kerberos V5 sollte die Fähigkeit zur Weiterbenutzung aller dieser Mechanismen sein. Wo dies machbar ist, sollte aufgrund der erweiterten Funktionalität von Kerberos V5 ein Austausch der Authentifizierungsdienste gegen Kerberos V5-kompatible Dienste mit AFS-Unterstützung angestrebt werden.

Am einfachsten ist die Erweiterung auf Kerberos V5 bei der Nutzung von PAM-Modulen zu erreichen, da ein PAM-Modul zur Authentifizierung an Kerberos V5 mit den bisher eingesetzten Modulen zusammen eingesetzt werden könnte.

Die „Secure Shell“ bietet zwar eine Unterstützung für Kerberos V5, diese wurde aufgrund des Fehlens einer KerberosV5-Realm im URZ bisher jedoch nicht eingesetzt. Die gleichzeitige Nutzung der KerberosV4- und der KerberosV5-Unterstützung ist nicht möglich. Es ist nicht bekannt, ob die AFS-Unterstützung der Secure Shell noch funktioniert, wenn anstatt der KerberosV4- die KerberosV5-Unterstützung eingeschalten wird. Sollte dies nicht der Fall sein, so bleibt die Secure Shell in ihrer zur Zeit eingesetzten Version mit „Heimdal“ benutzbar, kann aber keine Kerberos V5-TGT anfordern und weiterleiten.

Die Benutzung von „klog“ und „AFS-Login“ ist weiterhin möglich, da das KDC von „Heimdal“ den AFS-kaserver teilweise emuliert. Das „AFS-Login“ sollte durch ein Kerberos V5-kompatibles Login-Programm ausgestauscht werden. Ein solches Login-Programm ist in in einer frühen Version in „Heimdal“ enthalten, wird derzeit aber noch nicht installiert. Dieses Login-Dienstprogramm fordert ein Kerberos V5-TGT an, das vom „Heimdal“-Dienstprogramm „afslog“ benutzt werden kann, um ein AFS-Token zu beschaffen. Dabei ist keine Neueingabe des Nutzerpaßworts erforderlich, so daß für den Nutzer kein Unterschied zwischen „AFS-Login“ und „Heimdal“-Login+„afslog“ bemerkbar wäre. Der AFS-Client „klog“ kann durch den „Heimdal“-Client „kinit“ ersetzt werden, da dieser bei Aufruf mit der Option „—afslog“ ein AFS-Token anfordert und installiert.

Der Vorteil der Ersetzung von „AFS-Login“ und „klog“ ist das Vorhandensein eines KerberosV5-TGT nach der Authentifizierung. Dieses TGT kann zur Anforderung weiterer Service-Tickets genutzt werden, so daß der Nutzer im Idealfall sein Paßwort nur einmal bei der Anmeldung an der Kerberos-Realm eingeben muß.

## *4.3. Vorgehensweise bei der Migration*

Die Migration von AFS-kaserver zu „Heimdal“-Kerberos V5 umfaßt eine Reihe von Teilaufgaben. Im ersten Schritt ist die Installation von „Heimdal“ auf den Datenbank-Servern der AFS-Zelle sinnvoll. Bei der Installation sind folgende Teilarbeiten notwendig:

1. Installation von „Heimdal“ auf den Datenbankservern.
2. Initialisierung der Realm „tu-chemnitz.de“ auf dem Master-Server.
3. Erstellung des Principals für den AFS-Service „afs@TU-CHEMNITZ.DE“.
4. Konvertierung der kaserver-Datenbank (kaserver.DB0) in eine „Heimdal“-Datenbank und Einfügen der konvertierten AFS-Nutzer in die Datenbank der Realm.
5. Einrichtung des Verteilungsmechanismus der Datenbank an die Slave-Server.
6. Installation des Paßwortänderungs- und Administrations-Servers auf dem Master-Server.
7. Inbetriebnahme des KDC und Test von Kerberos V5 und AFS.
8. Abschaltung des AFS-kaserver und Integration des KDC in den Basic Overseer Server.

Diese Schritte werden nachfolgend im Abschnitt 4.4. detailliert beschrieben.

Nachdem die Datenbankserver ihren Betrieb aufgenommen haben, gilt es, die restlichen Hosts der Realm (Kerberos-Clients) für die Nutzung von Kerberos V5 zu konfigurieren. Dazu ist auf allen Clients eine Konfigurationsdatei */etc/krb5.conf* zu installieren. Sollen KerberosV5-kompatible Server, wie Kerberos-FTP und –Telnet benutzt werden, muß zusätzlich eine Keytab auf dem entsprechenden Host gespeichert werden. Die Arbeiten für die Konfiguration der Clients werden in Abschnitt 4.5. erläutert.

## *4.4. Einrichtung der Kerberos-Datenbankserver*

### 4.4.1. Installation von „Heimdal“

Die bestehende Struktur der AFS-Zelle mit drei Datenbankservern sollte beibehalten werden, da sie den Vorteil der Ausfallsicherheit bietet und sie sich im Betrieb bewährt hat. „Heimdal“ enthält Unterstützung für den Betrieb mehrerer Datenbankserver. Einer der drei Datenbank-server wird zum Master-Server bestimmt (im Beispiel **zuse**), die beiden anderen werden als Slave-Server betrieben. Alle Änderungen an der Principal-Datenbank werden nur auf dem

Master-Server vorgenommen und die geänderte Datenbank wird anschließend an die Slave-Server verteilt.

Die Installation von „Heimdal“ erfolgt wie in 3.2.2. beschrieben auf allen Datenbankservern. Danach sollte „Heimdal“ im Pfad */usr/heimdal/* installiert sein. Nun wird die Konfigurationsdatei */etc/krb5.conf* angelegt, die Informationen über Realm und die zugehörigen Datenbankserver enthält. Die Konfigurationsdatei ist im Anhang aufgelistet.

### 4.4.2. Die Initialisierung der Realm

Die Realm **„TU-CHEMNITZ.DE“** wird auf dem Master-Server neu initialisiert. Das dazu notwendige Vorgehen wurde bereits in Abschnitt 3.2.5. beschrieben. Die Principal-Datenbank enthält nach der Initialisierung folgende Einträge:

1. **krbtgt/TU-CHEMNITZ.DE@TU-CHEMNITZ.DE**
   Der Eintrag für den Ticket-granting-Server (TGS) der Realm.

2. **kadmin/admin@TU-CHEMNITZ.DE**
   Der Eintrag für den Administrationsserver ***kadmind***.

3. **kadmin/changepw@TU-CHEMNITZ.DE**
   Der Eintrag für den Paßwortänderungs-Server ***kpasswdd***.

4. **default@TU-CHEMNITZ.DE**
   Der Default-Principal.

### 4.4.3. Die Erstellung des AFS-Service-Principals

Um Tickets für den AFS-Service erhalten zu können, ist es notwendig, einen AFS-Service-Principal in der Datenbank anzulegen. Dies geschieht unter Benutzung von ***kadmin*.** Der Principal erhält die Bezeichnung **afs@TU-CHEMNITZ.DE**.

Für diesen Service wird von ***kadmin*** ein zufälliger DES- und DES3-Schlüssel in der Datenbank angelegt. Wenn vom KDC ein Ticket für den AFS-Service angefordert wird, verschlüsselt das KDC dieses Ticket standardmäßig mit dem Schlüssel, der die größte Sicherheit bietet, in diesem Fall mit dem DES3-Schlüssel.

Da AFS-Fileserver nur mit DES-Schlüsseln, aber nicht mit DES3-Schlüsseln umgehen können, müssen letztere aus der Principal-Datenbank entfernt werden, so daß nur noch der DES-Schlüssel in der Datenbank steht. Weiterhin muß der DES-Schlüssel und seine kvno für den erstellten AFS-Service-Principal mit dem Eintrag im AFS-KeyFile übereinstimmen. Wie in Abschnitt 2.5.3. bereits dargelegt, bietet „Heimdal“ keine Dienstprogramme für diese Abgleichung von Datenbank und KeyFile, sondern die Änderungen in der Datenbank müssen „von Hand“ vorgenommen werden.

Mit ***kadmin*** und dem Kommando **dump** kann die Principal-Datenbank in einem lesbaren Format in eine Dump-Datei gespeichert werden. Für jeden Principal in der Datenbank wird dabei ein Eintrag mit folgendem Aufbau angelegt:

Principalname
kvno:Schlüsseltyp1:Schlüssel1:Schlüsseldaten1:Schlüsseltyp2:Schlüssel2...
Erzeugungsdatum und –Principal : Änderungsdatum und –Principal
Principal gültig ab : Principal gültig bis : Principal-Schlüssel verfällt am :
Maximale Ticket-Lebensdauer : Maximale Ticket-erneuerbar-Zeit :
Flags : unterstützte Verschlüsselungstypen

Für den neu angelegten Principal **afs@TU-CHEMNITZ.DE** ergibt sich folgender Eintrag:

afs@TU-CHEMNITZ.DE 1:0:*1:aabbccddeeff1122:-*
**:0:7:11223344556677889900aabbccddeeff1122334455667788:-**
19980913120102:kadmin/admin@TU-CEHMNITZ.DE - - - - 360000 -126 -

Man kann erkennen, daß zwei Schlüssel mit der kvno 1 für diesen Principal generiert wurden, ein Schlüssel des Typs 1 (DES, kursiv dargestellt) und ein Schlüssel des Typs 7 (DES3, fett dargestellt). Der DES3-Schlüssel wird nun entfernt, indem der fett dargestellte Abschnitt in der Dump-Datei gelöscht wird. Der DES-Schlüssel bleibt erhalten, da er aber zufällig erzeugt wurde, stimmt er noch nicht mit dem Schlüssel im AFS-KeyFile überein. Mit dem Befehl

**bos listkeys –host zuse**

werden die derzeit im AFS-KeyFile abgelegten Schlüssel angezeigt. Dabei werden allerdings nur die kvno der Schlüssel und eine Prüfsumme aufgelistet. Nun kann durch

**bos addkey –host zuse –key XXXX –kvno Y**

ein neuer DES-Schlüssel mit der kvno Y im AFS-KeyFile angelegt werden. Dabei muß Y höher als alle bisher im AFS-KeyFile gespeicherten kvno sein, um vom Fileserver benutzt zu werden. Der neue Schlüssel wird von ***bos*** mit der AFS-String-to-Key-Funktion aus dem Paßwort „XXXX“ und dem AFS-Zellenname erzeugt.

Der Administrator wird jetzt mit dem „Heimdal“-Werkzeug ***string2key –a*** diesen Schlüssel nocheinmal erzeugen und anstelle des bisherigen DES-Schlüssels in die Dump-Datei eintragen. Es ergibt sich nun folgender Eintrag in der Dump-Datei:

```
afs@TU-CHEMNITZ.DE
5:0:1:1a2b3c4e5f6a7b8c:- 19980913120102:kadmin/admin@TU-CHEMNITZ.DE - - - - 360000 - 126 -
```

kvno — DES-Key — Erstellungszeit — Erstellender Principal — Lebensdauer — Flags

wobei im Beispiel ***1a2b3c4e5f6a7b8c*** der 8-Byte lange DES-Schlüssel und die kvno ***5*** wäre.

Die geänderte Dump-Datei wird mit ***kadmin*** und dem Kommando **load** eingelesen und die „Heimdal“-Datenbank wird entsprechend neu erzeugt. Das geänderte AFS-KeyFile muß auf allen AFS-Fileservern der Zelle neu installiert werden. Damit sind alle Voraussetzungen erfüllt, um ein gültiges AFS-Token für die Zelle „tu-chemnitz.de“ vom „Heimdal“-KDC anfordern zu können.

### 4.4.4. Die Konvertierung der kaserver-Datenbank

Die Anforderung von AFS-Service-Tickets bzw. AFS-Token wird üblicherweise von Nutzern des AFS vorgenommen. Diese Nutzer müssen aus der bestehenden AFS-kaserver-Datenbank in die Principal-Datenbank von „Heimdal“ übernommen werden.

Die Datenbank des AFS-kaservers ist im Pfad */usr/afs/db/kaserver.DB0* gespeichert. Da diese Datenbank ein Format hat, das sich vom „Heimdal“-Format unterscheidet, können weder ***kadmin*** noch das KDC von „Heimdal“ auf sie zugreifen. Das einzige „Heimdal“-Dienstprogramm, das AFS-kaserver-Datenbanken einlesen kann, ist das Datenbank-Verteilungsprogramm ***hprop***. Es dient in erster Linie zur Verteilung der Principal-Datenbank vom Master- an die Slave-Server, kann jedoch auch zur Konvertierung der AFS-kaserver-Datenbank genutzt werden.

Vor der Nutzung von ***hprop*** und ***hpropd*** muß mittels ***kadmin*** je ein Principal für diese Programme in der Datenbank angelegt werden:

| | |
|---|---|
| **kadmin/hprop@TU-CHEMNITZ.DE** | Service-Principal für ***hpropd*** |
| **hprop/zuse.hrz.tu-chemnitz.de@TU-CHEMNITZ.DE** | Service-Principal für ***hprop*** |

Diese beiden Einträge müssen mittels ***kadmin*** **ext** in die Keytab des Master-Servers extrahiert werden.

Mit dem Kommando:

**hprop –d /usr/afs/db/kaserver.DB0 –K –c tu-chemnitz.de –n > /var/heimdal/kaserver.txt**

wird die gesamte AFS-kaserver-Datenbank im hprop-Format in */var/heimdal/kaserver.txt* gespeichert. Die geheimen Schlüssel der Principals werden zusammen mit dem Zellenname „tu-chemnitz.de“ in die neue Datei kopiert. Der Zellenname wird in allen Nachrichten mit dem kopierten Schlüssel zusammen versandt, da er von AFS- und Kerberos-Clients als Argument für die AFS-String-to-Key-Funktion benötigt wird.

Die Datei */var/heimdal/kaserver.txt* kann nun von ***hpropd*** eingelesen und in eine KerberosV5-Datenbank */var/heimdal/kaserver.db* überführt werden:

**cat /var/heimdal/kaserver.txt | hpropd –n –D /var/heimdal/kaserver**

Aus der konvertierten Datenbank werden nur die Nutzer-Principals benötigt. Alle anderen Principals, wie

**AuthServer/Admin@TU-CHEMNITZ.DE**
**krbtgt/TU-CHEMNITZ.DE@TU-CHEMNITZ.DE**
**afs@TU-CHEMNITZ.DE**

werden nicht konvertiert, da sie entweder nicht benötigt werden oder neu angelegt wurden. Weiterhin müssen die bei der Initialisierung der „Heimdal“-Realm erzeugten Principals beibehalten werden.

Aus diesen Gründen kann die konvertierte Datenbank nicht einfach die „Heimdal“-Datenbank ersetzen, sondern es muß ein Teil der migrierten AFS-kaserver-Datenbank in die schon bestehende Datenbank eingefügt werden.

Das ***kadmin***-Programm ist in der Lage, mit dem Befehl **merge** einen Datenbank-Dump in eine bestehende Datenbank einzufügen. Es muß also ein Dump der AFS-kaserver-Datenbank erzeugt werden und dieser dann in die bei der Initialisierung erzeugte „Heimdal“-Datenbank eingefügt werden. Da wiederum nur ***kadmin*** Dump-Dateien erzeugen kann und ***kadmin*** auf */var/heimdal/heimdal.db* zugreift, wird die bestehende Datenbank */var/heimdal/heimdal.db* umbenannt und */var/heimdal/kaserver.db* nach */var/heimdal/heimdal.db* kopiert:

| | | |
|---|---|---|
| /var/heimdal/heimdal.db | → umbennen → | /var/heimdal/heimdal_old.db |
| /var/heimdal/kaserver.db | → kopieren → | /var/heimdal/heimdal.db |

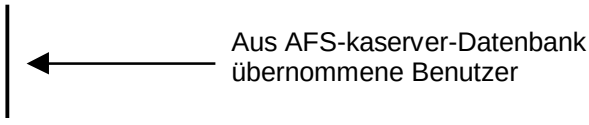
Jetzt kann die konvertierte AFS-kaserver-Datenbank mit ***kadmin*** bearbeitet werden. Mit dem Kommando **del** werden alle Principals, die keine Benutzer repräsentieren, gelöscht. Mittels **dump afs.dump** wird eine Dump-Datei *afs.dump* angelegt. Im nächsten Schritt wird die originale „Heimdal“-Datenbank wiederhergestellt:

| | | |
|---|---|---|
| /var/heimdal/heimdal_old.db | → umbenennen → | /var/heimdal/heimdal.db |

Nach dem erneuten Start von ***kadmin*** kann diese bearbeitet werden. Mit dem Befehl **merge afs.dump** wird die im vorigen Schritt erzeugte Dump-Datei mit den Einträgen der AFS-Benutzer aus der AFS-kaserver-Datenbank in die „Heimdal“-Datenbank übernommen. Die so erzeugte Datenbank enthält folgende Einträge:

krbtgt/TU-CHEMNITZ.DE@TU-CHEMNITZ.DE
default@TU-CHEMNITZ.DE
kadmin/admin@TU-CHEMNITZ.DE
kadmin/hprop@TU-CHEMNITZ.DE
kadmin/changepw@TU-CHEMNITZ.DE
afs/tu-chemnitz.de@TU-CHEMNITZ.DE
hprop/zuse.hrz.tu-chemnitz@TU-CHEMNITZ.DE
host/localhost@TU-CHEMNITZ.DE

user1@TU-CHEMNITZ.DE
user2@TU-CHEMNITZ.DE
user3@TU-CHEMNITZ.DE
admin@TU-CHEMNITZ.DE



Nachdem nun die Konvertierung der AFS-kaserver-Datenbank beendet ist, sollte eine Sicherheitskopie derselben angelegt werden.

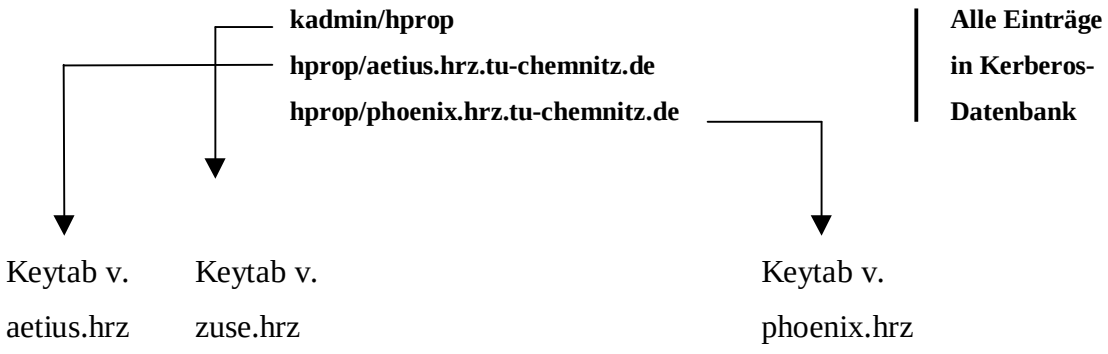
Das beschriebene Verfahren zur Konvertierung ist relativ umständlich und kompliziert. Deshalb wurde vom Verfasser ein Programm ***afsconv*** erstellt, welches die beschriebenen Schritte bis zur Erzeugung der Dump-Datei der AFS-kaserver-Datenbank ausführt. Dazu wurden Codebestandteile aus ***hprop***, ***hpropd*** und ***kadmin*** verwendet.

### 4.4.5. Die Verteilung der Principal-Datenbank an die Slave-Server

Auf den Slave-Servern (im Beispiel aetius und phoenix) wird „Heimdal“ wie auf dem Master-Server installiert. Die Principal-Datenbank muß nun vom Master-Server auf die Slave-Server überführt werden. Zu diesem Zweck enthält „Heimdal“ den Verteilungsserver ***hpropd*** und den Client ***hprop.***

Der Server ***hpropd*** wird auf den Slave-Servern ständig ausgeführt und wartet auf eine Verbindung vom Master-Server auf dem Port 754 bzw. dem in der Datei */etc/services* unter „hprop/tcp“ angegebenen Port. Auf dem Master-Server wir der Client ***hprop*** immer dann gestartet, wenn die Datenbank an einen Slave-Server gesendet werden soll. Da die Verteilung der Datenbank sicherheitsrelevant ist, authentifiziert sich ***hprop*** beim Start gegenüber dem KDC als Principal kadmin/hprop. Dieser Principal muß daher in der Principal-Datenbank und in der Keytab des Master-Servers vorhanden sein.

Für den Principal kadmin/hprop wird zunächst ein TGT beantragt. Mit dem erhaltenen TGT wird ein Ticket für den Service **hprop/<Slaveserver>** vom KDC angefordert. Die Principals hprop/<Slaveserver> müssen also in der Datenbank angelegt werden und in die Keytab des jeweiligen Slave-Servers extrahiert werden:



Beim Start von ***hprop*** müssen der Name und Pfad der Datenbank, der Name und Pfad der lokalen Keytab und der Hostname des Slave-Servers angegeben werden:

**hprop –d /var/heimdal/heimdal.db –k /etc/krb5.keytab aetius.hrz**

Nach erfolgreicher Authentifizierung baut ***hprop*** eine Verbindung zu ***hpropd*** auf dem jeweiligen Slave-Server auf und sendet die Datenbank. Diese wird von ***hpropd*** automatisch im Pfad */var/heimdal/heimdal.db* gespeichert. Ist die Datenbank mit einem Master-Key verschlüsselt, so muß dieser Master-Key auf allen Datenbankservern identisch sein, da sonst die Principal-Datenbank auf den Slave-Servern nicht richtig dekodiert wird. Die Benutzung eines Master-Key zur Verschlüsselung ist bei der Verteilung der Principal-Datenbank unbedingt zu empfehlen, da die Datenbank vom Netz abgelauscht werden kann. Ist diese unverschlüsselt, ist die Extraktion der geheimen Schlüssel leicht.

Der Start von ***hprop*** kann entweder von Hand oder durch ein periodisch ablaufendes Script durchgeführt werden. Ein automatischer Start nach Änderungen in der Datenbank ist in „Heimdal“ nicht vorgesehen, wäre allerdings wünschenswert.

Bei den Tests mit ***hprop*** und ***hpropd*** wurden Funktionssschwierigkeiten unter Solaris 2.x festgestellt, die von einer Fehlfunktion in ***hpropd*** herrühren. Bei der Erstellung der neuen Datenbank wird nur eine leere Datei angelegt. Vor der Speicherung von Principals in der neuen Datenbank stürzt ***hpropd*** ab. Unter Linux 2.x wurden keine derartigen Funktionsstörungen festgestellt.

### 4.4.6. Der Paßwort-Änderungs-Server kpasswdd

Den Nutzern des KerberosV5- und AFS-Service muß es natürlich möglich sein, ihre Paßworte und somit ihre geheimen Schlüssel zu ändern. Dazu muß direkt auf die Principal-Datenbank zugegriffen werden. Es gibt zwei Möglichkeiten zum Zugriff auf die Datenbank, die Benutzung von ***kadmin*** oder von ***kpasswd***.

Das Dienstprogramm ***kadmin*** kann unter anderem auch zur Änderung von Paßworten benutzt werden. Da nur der Administrator der Kerberos-Realm die notwendigen Berechtigungen zur Ausführung von ***kadmin*** besitzt, müßte der betroffene Nutzer also den Administrator

beauftragen, das Paßwort zu ändern. Dieser Weg ist nicht akzeptabel, da er zum einen für den Administrator unnötigen Aufwand generiert und zum anderen für den Benutzer die Nennung des Paßwortes bedeutet.

Um einen einfacheren Weg zur Änderung von Nutzerpaßworten zur Verfügung zu stellen, enthält „Heimdal“ den Paßwort-Änderungs-Server ***kpasswdd***, der direkt auf die Principal-Datenbank zugreift. Er muß deshalb auf dem Master-Server ausgeführt werden. Der Server wartet auf Port 464 auf Verbindungen von Paßwort-Änderungs-Clients ***kpasswd***. Möchte ein Benutzer sein Paßwort ändern, startet er den Client. Dieser fragt zuerst das bisherige Paßwort des Nutzers ab und authentifiziert den Nutzer damit. Im Gegensatz zur Anforderung eines TGT fordert ***kpasswd*** ein Ticket für den Service **kadmin/changepw@TU-CHEMNITZ.DE** an. Der Schlüssel dieses Principals muß in der Keytab des Master-Servers enthalten sein, damit ***kpasswdd*** das Ticket entschlüsseln kann.

Nach der Authenitfizierung des Nutzers wird das neue Paßwort abgefragt. Dieses Paßwort wird jetzt verschlüsselt zum Server übertragen, der in der Principal-Datenbank neue Schlüssel mit einer um 1 erhöhten kvno erzeugt. Die bisher benutzten Schlüssel werden gelöscht, da sie nicht mehr benötigt werden. Selbst wenn der Benutzer im Besitz eines gültigen TGT ist, wird der zum Erhalt dieses TGT einstmals benötigte geheime Schlüssel zum Beantragen von weiteren Service-Tickets nicht mehr gebraucht, da der session-key aus dem TGT genutzt wird (s.a. 1.4.).

In der Originalversion von „Heimdal“ führt eine Änderung eines AFS-Schlüssels, d.h. eines DES-Schlüssels, der unter Benutzung der AFS-String-to-Key-Funktion berechnet wurde, zu fehlerhaften Ergebnissen. Dieser Fehler beruht auf der Verwendung der falschen String-to-Key-Funktion bei der Berechnung des Schlüssels. Nach Rücksprache mit dem Entwicklerteam von „Heimdal“ wurde von diesem ein Patch zur Fehlerbereinigung erstellt und vom Verfasser integriert. Dieser Patch ermöglicht die Änderung von AFS-Nutzer-Paßworten.

Nach einer Paßwortänderung muß die geänderte Principal-Datenbank vom Master-Server auf die Slave-Server verteilt werden. Dies wirft Probleme auf, da der Administrator den Zeitpunkt der Änderung nicht kennt. Er kann daher die Verteilung nicht manuell anstoßen. Erfolgt die Verteilung periodisch, kann der Benutzer bis zum Zeitpunkt der nächsten Abgleichung der

Datenbank unter Umständen kein TGT anfordern, nämlich dann, wenn er einen der Slave-Server kontaktiert. Die beste Methode zur Umgehung dieses Problems ist die Anforderung eines TGT durch den Nutzer vor der Änderung des Paßwortes.

### 4.4.7. Der Administrationsserver kadmind

Um die Administration der Kerberos-Realm zu erleichtern, gibt es in „Heimdal“ die Möglichkeit, Änderungen an der Datenbank durch ***kadmin*** von allen Hosts im Netzwerk durchführen zu können. Dazu wird auf dem Master-Server der Administrationsserver ***kadmind*** gestartet. Dieser wartet auf Anfragen von ***kadmin*** auf Port 465. Wird ***kadmin*** mit dem Hostnamen des Master-Servers als Argument gestartet, wird zuerst ein Ticket für den Service **kadmin/admin@TU-CHEMNITZ.DE** angefordert. Dieser Principal muß deshalb in der Keytab des entfernten Rechners enthalten sein. Nach der Authentifizierung wird ***kadmin*** gestartet. Um unbefugte Änderungen an der Principal-Datenbank auszuschließen, muß der Benutzer, der ***kadmin*** startet, bestimmte Privilegien besitzen. Diese werden durch einen Eintrag des Principals in eine „Access Control List“ (ACL) erteilt. Die ACL wird in der Datei */var/heimdal/kadmind.acl* gespeichert. Für jeden Benutzer, dem Rechte zur Änderung der Datenbank eingeräumt werden sollen, ist ein Eintrag der Form

**Nutzer Flags**

in der ACL erforderlich, wobei das Flag-Feld eine Kombination der Rechte „change-password“, „add“, „modify“, „delete“, „list“ oder „all“ (durch Kommata getrennt) enthält.

### 4.4.8. Inbetriebnahme der Datenbankserver

Nachdem alle Schritte der Installation der Datenbankserver abgeschlossen wurden, können diese in Betrieb genommen werden. Das bedeutet das Abschalten des AFS-kaserver und den Start des „Heimdal“-KDC:

**bos stop zuse kaserver**
**/usr/heimdal/libexec/kdc&**

Der AFS-kaserver wird vom AFS-Basic-Overseer-Server (BOS) automatisch zusammen mit den anderen AFS-Servern gestartet. Da in Zukunft anstelle des kaservers das KDC gestartet werden soll, wird die BOS-Konfiguration entsprechend abgeändert. Mit

**bos create zuse simple kdc /usr/heimdal/libexec/kdc**

wird eine KDC-Instanz erzeugt, die zusammen mit den AFS-Servern gestartet wird. Zum Test sollte der KDC-Prozeß mit „kill“ beendet werden. Wird er von „bos“ danach automatisch neu gestartet, war die Erstellung der KDC-Instanz erfolgreich.

Da die Installation der Datenbankserver abgeschlossen ist, sollte jetzt überprüft werden, ob für die konvertierten AFS-Nutzer ein TGT beschafft werden kann. Mit ***kinit*** **<*Nutzer*>** wird ein TGT vom KDC angefordert. Mit dem Programm ***klist*** wird der Inhalt des Ticket-Cache des aktuell angemeldeten Benutzers angezeigt, er sollte ein TGT enthalten (s. a. 3.2.7.). Das KDC erzeugt im Pfad */var/heimdal/kdc.log* eine Log-Datei, die im Fehlerfall eine wertvolle Informationsquelle darstellt.

Ist ***kinit*** erfolgreich gewesen und ein gültiges TGT ist beschafft, wird mit ***afslog*** ein AFS-Token für den Benutzer angefordert. Mit dem Programm ***tokens*** kann der Erhalt des AFS-Tokens überprüft werden. Ist ein solches verfügbar, sollten einige Files im AFS gelesen und geschrieben werden. Wenn bei der Nutzung des Tokens AFS-„rxkad“-Fehlermeldungen auftreten, ist eine Ungleichheit der kvno in „Heimdal“-Datenbank und AFS-KeyFile der wahrscheinlichste Grund. Mit den Kommandos:

**bos listkeys zuse** und
**kadmin get –long afs**

können die kvno’s des AFS-Service in AFS-KeyFile und Kerberos-Datenbank angezeigt werden. Diese müssen übereinstimmen.

Im nächsten Schritt sollte der Verteilungsmechanismus getestet werde. Dazu wird auf dem Master-Server das Kommando

**hprop –d /var/heimdal/heimdal.db –k /etc/krb5.keytab <Slave-Server>**

ausgeführt. Auf dem entsprechenden Slave-Server sollte danach die Principal-Datenbank im Pfad */var/heimdal/heimdal.db* zur Verfügung stehen. Ist auch der Verteilungsmechanismus ordnungsgemäß eingerichtet, kann die Installation der Datenbankserver als abgeschlossen erachtet werden.

## *4.5. Einrichtung der Client-Hosts*

### 4.5.1. Konfiguration der Hosts

Die Hosts der Kerberos-Realm, die nicht als Datenbankserver fungieren, werden über die Datei */etc/krb5.conf* konfiguriert. Diese Datei enthält Informationen über die Namen der Realm und der Datenbankserver. Weiterhin werden Informationen für die Umwandlung von Kerberos V4-Principals in Kerberos V5-Principals in krb5.conf gespeichert. Die vollständige krb5.conf für die Realm „TU-CHEMNITZ.DE“ ist als Anhang beigefügt.

### 4.5.2. Installation der Kerberos-Client-Programme

Im Gegensatz zu den Datenbankservern werden auf den Client-Rechnern nur die Programme installiert, die im Pfad */usr/heimdal/bin* stehen. Außerdem werden die Man-Pages (/*usr/heimdal/man*) und die KerberosV5-Bibliotheken (*/usr/heimdal/lib)* mitinstalliert.
Die wichtigsten Client-Programme des „Heimdal“-Pakets sind:

- ***kinit*** dient zur Anforderung eines TGT vom KDC.
- ***klist*** listet den Inhalt des Ticket-Cache (/tmp/krb5cc_xxxx) auf.
- ***kdestroy*** entfernt alle Tickets des Benutzers aus dem Ticket-Cache
- ***afslog*** beschafft unter Benutzung eines TGT ein AFS-Token.
- ***ftp*** Ein FTP-Client mit Kerberos-Integration, d.h. keine Paßwort-Eingabe notwendig, wenn TGT vorhanden. Dieser Client kann auch mit FTP-Servern Verbindung aufnehmen, die Kerberos nicht unterstützen.
- ***telnet*** Ein Telnet-Client mit Kerberos-Integration, d.h. keine Paßwort-Eingabe notwendig, wenn TGT vorhanden. Dieser Client kann auch mit Telnet-Servern Verbindung aufnehmen, die Kerberos nicht unterstützen.
- ***kpasswd*** dient zur Änderung von Paßworten. Dieses Programm kommuniziert mit dem Server kpasswdd auf dem Master-Server.

Zur Benutzung dieser Programme sind keine weiteren Voraussetzungen außer der Konfigurationsdatei notwendig.

### 4.5.3. Installation der Kerberos-kompatiblen Server

Sollen auf bestimmten Rechnern Dienste zur Verfügung gestellt werden, die Kerberos V5 zur Authentifizierung benutzen, müssen neben den Client-Programmen auch die unter dem Pfad */usr/heimdal/libexec* stehenden Server-Programme installiert werden. Dabei werden nicht alle in „Heimdal“ enthaltenen Server installiert. Insbesondere werden die Server ***kpasswdd, hpropd, kdc, hprop*** und ***kadmind*** nur auf Datenbankservern benötigt.

Für den Einsatz im Netzwerk sind vor allem der FTP-, der Telnet- und der POP3-Server sinnvoll. Diese bieten gegenüber den bislang benutzten Servern den Vorteil der Nutzung von Kerberos V5, die diese Dienste sicherer und komfortabler macht. Nutzung von Kerberos bedeutet insbesondere, daß ein Nutzer, der bereits ein KerberosV5-TGT besitzt, nicht nocheinmal sein Paßwort eingeben muß („single sign-on“). Das Paßwort wird so nicht über das Netz übertragen und kann demzufolge auch nicht abgehört werden.

Wie bereits in 1.4. erläutert, fordert ein Client mit seinem gültigen TGT ein Service-Ticket für einen bestimmten Service an, den er benutzen möchte. Dieses Service-Ticket wird mit dem geheimen Schlüssel des jeweiligen Service verschlüsselt. Damit der Server das Ticket entschlüsseln kann, benötigt er diesen geheimen Schlüssel. Dazu werden die Schlüssel der Services in der Keytab des Hosts gespeichert, üblicherweise in */etc/krb5.keytab*. Die Einträge in dieser Keytab sind hostspezifisch, so daß für jeden Rechner eine eigene Keytab erzeugt werden muß. Diese sollte auf Rechnern, die nicht Datenbank-Server sind, folgende Einträge enthalten:

1. **host/<HOSTNAME>@TU-CHEMNITZ.DE** für den Telnet-Server.
   (wird auf allen Hosts benötigt, auf die mit telnet, ssh oder rsh zugegriffen werden soll)

2. **ftp/<HOSTNAME>@TU-CHEMNITZ.DE** für den FTP-Server.
   (wird auf allen Hosts benötigt, auf denen ein FTP-Server ausgeführt werden soll)

3. **pop3/<HOSTNAME>@TU-CHEMNITZ.DE** für den POP3-Server.
   (wird auf Hosts benötigt, auf denen der POP3-Server v. „Heimdal“ läuft)

Diese Principals müssen natürlich in der Principal-Datenbank vorhanden sein, bevor sie in eine Keytab extrahiert werden können.

Um die Services vom Internet-Daemon ***inetd*** bei Bedarf automatisch starten zu lassen, wird in der Datei */etc/inetd.conf* der Pfad, unter dem der FTP-, Telnet- bzw. POP3-Server gespeichert sind, auf */usr/heimdal/libexec* angepaßt. Auf dem Testrechner „otroc“ sind sowohl FTP- als auch Telnet-Dienst durch Kerberos-Server ersetzt worden. Diese haben bislang problemlos sowohl mit „Heimdal“-Clients als auch mit anderen Clients zusammengearbeitet.

Der Telnet-Server von „Heimdal“ stellt die Weiterleitung von TGT‘s zur Verfügung, d.h. ein Nutzer mit einem gültigen TGT, der über Telnet an einem anderen Host arbeiten möchte, bekommt automatisch ein gültiges TGT für den neuen Host. Allerdings muß dazu im TGT das Flag „forwardable“ gesetzt werden. Dies wird durch Aufruf von ***kinit –forwardable*** erreicht.

Der in „Heimdal“ enthaltene POP3-Server ***popper*** unterstützt neben der Authentifizierung über POP3, bei der Paßworte im Klartext über das Netz gesendet werden, das Kerberos-POP-Protokoll (KPOP). Bei der Nutzung von KPOP durch einen Email-Client wird Kerberos zur Authentifizierung benutzt. Es existieren mehrere Clients mit KPOP-Unterstützung, u.a. „fetchmail“ und „pine“.

Der in einem Kerberos-Service-Ticket enthaltene session-key $K_{C,S}$ ermöglicht theoretisch eine Verschlüsselung der gesamten Kommunikation zwischen Client und Server. Inwieweit diese Fähigkeit in den in „Heimdal“ enthaltenen Servern und Clients eingebaut wurde, konnte mangels Dokumentation nicht festgestellt werden.

### 4.5.4. Integration der Secure-Shell (SSH)

Die Secure-Shell ssh wird im Rechenzentrum vor allem als Ersatz für Telnet eingesetzt, da sie viele Vorteile gegenüber Telnet bietet. Insbesondere werden bei ssh keine Paßworte im Klartext über das Netz gesendet, was den Diebstahl von Paßworten erschwert.

Der Secure-Shell-Server ***sshd***, der im Rechenzentrum eingesetzt wird, bietet Kerberos V4- und AFS-Unterstützung. Der ***sshd*** ist in der Lage, ein vorhandenes AFS-Token an den entfernten Rechner weiterzuleiten und neue AFS-Token anzufordern, falls der entfernte Host Mitglied einer anderen AFS-Zelle ist.

Weiterhin ist eine Authentifizierung an einem Kerberos V4-KDC möglich. Da „Heimdal“ abwärtskompatibel ist, sind diese Funktionen beim Einsatz von ***sshd*** in Verbindung mit einem Kerberos V5-KDC weiterhin nutzbar.

Das ssh-Paket bietet neben AFS- und Kerberos V4-Authentifikation eine Unterstützung für Kerberos V5. Diese Unterstützung wurde im URZ bei der Übersetzung des Pakets bislang nicht aktiviert, da kein Kerberos V5-KDC betrieben wurde. Eine Aktivierung der Kerberos V5-Unterstützung des ssh-Paketes würde die Weiterleitung und Neuanforderung von Kerberos V5-TGT ermöglichen. Die Secure-Shell könnte somit vollständig in die Kerberos V5-Realm integriert werden.

### 4.5.5. Authentifizierung von Nutzern an Client-Hosts

Die Authentifizierung von Nutzern wird unter Unix durch die Überprüfung eines Paßwortes erreicht. Nach erfolgter Authentifizierung wird dem Nutzer eine ID-Nummer zugewiesen, die zur Authorisation im System dient. Diese Vorgänge vollzieht das „Login“-Dienstprogramm. Bei der Nutzung von Kerberos V5 wäre es nun wünschenswert, durch das „Login“-Programm ein TGT zu beschaffen, so daß nur zum Zeitpunkt der Anmeldung an das System ein Paßwort eingegeben werden muß („single sign-on“). Ein solches „Login“-Programm ist zwar in einer frühen Version in „Heimdal“ enthalten, es wird allerdings noch nicht übersetzt. Es steht zu erwarten, daß dieses Programm in den nächsten Versionen von „Heimdal“ funktionsfähig sein wird. Ob es auch in der Lage ist, ein AFS-Token zu beschaffen, bleibt abzuwarten. Allerdings benötigt das „Heimdal“-Programm ***afslog*** keine Eingaben zur Beschaffung eines AFS-Tokens (es benutzt ein vorhandenes TGT), so daß dieses Programm in einem Login-Script integrierbar ist und auf diesem Wege ein AFS-Token bei der Anmeldung beschafft werden kann.

Verschiedene Linux-Distributionen und auch Solaris 2.5 und 2.6 benutzen zur Authentifizierung die Pluggable Authentication Modules (PAM). Dabei werden verschiedene Module benutzt, um den Benutzer zu authentifizieren. Der Vorteil dieser Methode ist die Flexibilität. Die Module können bei Bedarf ausgetauscht werden und miteinander kombiniert werden. Es ist möglich, mehrere Module hintereinander auszuführen, um so verschiedene Authentifizierungen gleichzeitig durchzuführen. Im URZ wird ein PAM-Modul zur Authentifizierung an AFS benutzt. Dieses Modul ruft das AFS-Dienstprogramm „klog“ auf und beschafft so ein AFS-Token. Da „klog“ mit „Heimdal“ zusammenarbeitet, wird auch das PAM-Modul mit

„Heimdal“ funktionieren. Auch die Änderung des Benutzerpaßwortes ist mit dem PAM-Modul möglich, wenn AFS-kpasswd durch „Heimdal“-kpasswd ersetzt wird.

NAOMARO ITOI hat ein PAM-Modul[13] geschrieben, das ein Kerberos V5-TGT anfordert. Dieses Modul wurde für den Betrieb mit MIT-KerberosV5 programmiert, insbesondere werden Include-Dateien und Bibliotheken von MIT-KerberosV5 zur Übersetzung benötigt. Da die Kerberos-Bibliothek von „Heimdal“ mit der MIT-KerberosV5-Bibliothek im wesentlichen kompatibel ist, konnte das PAM-Modul für den Betrieb mit „Heimdal“ angepaßt werden. Wird das Kerberos V5-PAM-Modul zusammen mit dem AFS-PAM-Modul benutzt, ist eine vollständige Authentifizierung des Benutzers sowohl gegenüber Kerberos V5 als auch AFS möglich, so daß ein Kerberos V5-TGT und ein AFS-Token zur Verfügung stehen.

### *4.6. Cross-Cell-Authentifizierung*

Kerberos V5 bietet die Möglichkeit, Authentifizierung über Realmgrenzen hinweg durchzuführen. Das erlaubt in Verbindung mit AFS die Anforderung von AFS-Token für andere Zellen außerhalb der eigenen. Um diese Cross-Cell-Authentifikation zu ermöglichen, wird im ersten Schritt die Inter-Realm-Authentifikation zwischen der eigenen und der fremden Kerberos-Realm eingerichtet. Eine gravierende Einschränkung bei der Nutzung der Inter-Realm-Authentifizierung ist, daß diese nur zwischen Kerberos V5-Realms funktioniert.

Wenn zwischen den Realms REALM_A und REALM_B eine Inter-Realm-Authentifizierung installiert werden soll, müssen zuerst in beiden Realms die Principals

**krbtgt/REALM_A@REALM_B**
**krbtgt/REALM_B@REALM_A**

angelegt werden. Dabei wird den korrespondierenden Principals in beiden Realms der gleiche Schlüssel mit der gleichen kvno zugeordnet. Nun ist es möglich, Tickets für die jeweils andere Realm zu erhalten. Zu beachten ist, daß ein Principal aus REALM_A, der ein Ticket für einen Service in REALM_B bekommt, diesem Service als user@REALM_A erscheint.

[13] www-personal.engin.umich.edu/~itoi/pam_krb5/pam_krb5-1.0.1.tar.gz

Im nächsten Schritt wird die AFS-Komponente eingerichtet. Wenn Benutzer <*user*> der Zelle realm_a ein AFS-Token für die Zelle realm_b anfordert, erscheint er dem Fileserver und der Protection Database in dieser Zelle als <*user*>*@realm_a*. Dieses Verhalten ist durch die Arbeitsweise der Inter-Realm-Authentifikation von Kerberos V5 bedingt. Benutzer aus „fremden“ Zellen werden jedoch nicht automatisch der AFS-Gruppe *system:authuser* zugeordnet. Es ist also notwendig, einen PTS-Eintrag für <*user*>*@realm_a* zu erstellen, da sonst keinerlei Ressourcen der AFS-Zelle realm_b genutzt werden können. Ist <*user*>*@realm_a* in der Protection Database angelegt, kann er in die entsprechenden ACL aufgenommen werden und erhält damit Zugriff auf AFS-Ressourcen in der entfernten Zelle.

Das „AFS-Migration-Kit“ der Firma Transarc ermöglicht die automatische Kreierung der PTS-Einträge. Dazu wird vom Administrator in der Zelle realm_b eine PTS-Gruppe system:authuser@realm_a angelegt und mit einer Gruppen-Quote größer Null versehen. Das Dienstprogramm „aklog“ des „AFS-Migration-Kit“ generiert nun automatisch die PTS-Einträge für <user>@realm_a und fügt diese der Gruppe system:authuser@realm_a zu. „Heimdal“ ist bisher nicht in der Lage, diese automatische Generierung vorzunehmen. Werden die PTS-Einträge manuell erstellt, sollte auch eine Gruppe system:authuser@realm_a in der Protection Database angelegt werden, in die diese Nutzer aufgenommen werden.

Wie schon erwähnt, ist die Cross-Cell-Authentifizierung nur zwischen Kerberos V5-Realms möglich. Sobald das „Heimdal“-KDC die Funktion des AFS-kaserver übernommen hat, ist daher keine Cross-Cell-Authentifizierung mit AFS-Zellen möglich, die AFS-kaserver benutzen.

# 5. Die KerberosV5-Implementation von Windows NT 5

Bisher hat die Firma Microsoft die zweite Beta-Version des Produkts an Entwickler und Mitglieder des „Microsoft Developer Network“ ausgeliefert. Der Zeitpunkt des Erscheinens der fertigen Version von Windows NT 5 ist nicht bekannt. Dem Verfasser war es aufgrund der Beschränkung der Vergabe der Beta2-Version auf Entwickler nicht möglich, Zugriff auf Windows NT 5 zu erlangen. Daher konnten keine Untersuchungen der KerberosV5-Implementation von Windows NT 5 erfolgen. Es wurden jedoch Aussagen der Firma Microsoft und von Entwicklern des MIT-KerberosV5 gefunden, die einige Aspekte der Implementation von Microsoft und der Zusammenarbeit mit bestehenden KerberosV5-Installationen beleuchten.

Microsoft plant, KerberosV5 als primäres Sicherheitsprotokoll in Windows NT 5 zu benutzen. Jeder Domain-Controller von Windows NT 5 wird auch ein KDC sein, wobei es eine 1:1-Abbildung zwischen Domain und Kerberos-Realm gibt. Im Die Principal-Datenbank wird im neuen „Active Directory“, einer verteilten Datenbank, gespeichert. Im Unterschied zu anderen KerberosV5-Implementationen gibt es keine Trennung zwischen Master- und Slave-Servern, sondern alle Domain-Controller sind gleichberechtigt und gleichen das „Active Directory“ untereinander ab (Multi-Master-Modell). Microsoft legt die Standardisierung von KerberosV5 in RFC 1510 für ihre Implementation zu Grunde, möchte allerdings einige Erweiterungen einbeziehen, vor allem die Unterstützung für Public-Key-Kryptographie[14].

In Windows NT 5 wird Kerberos nicht nur zur Authentifizierung, sondern auch zur Authorisierung von Benutzern verwendet. Dazu nutzt Windows NT 5 das „Authorization Data“–Feld im Kerberos-Ticket. In diesem Feld speichert Windows NT 5 die Security Identification (SID) und die Gruppenzugehörigkeiten des Nutzers. Diese werden vom KDC bei der Anforderung eines TGT im Feld „Authorization Data“ abgelegt. Die Benutzung solcher Authorisierungsdaten ist in RFC 1510 ausdrücklich gestattet, auch OSF-DCE benutzt Daten zur Authorisierungssteuerung in Kerberos V5-Tickets. Die Authorisierungsdaten werden vom KDC verschlüsselt, um die unzulässige Aneignung von Berechtigungen zu verhindern. Das KerberosV5-Protokoll transportiert die Authorisierungsinformationen nur, die Auswertung der Daten bleibt Anwendungen vorbehalten. Wird mit dem TGT ein Service-Ticket ange-

[14] Phil Cox, „Windows NT 5.0: Integration Friendly?“, ;login:, Official Magazine of the Usenix Association, Dezember 1997

fordert, kopiert der TGS die Authorisierungsdaten in das Service-Ticket. Diese Daten werden vom kontaktierten Service benutzt, um beim Start die Rechte des Nutzer anzunehmen.

„Heimdal"-KDC und andere Implementationen von KerberosV5, die nicht von Microsoft stammen, haben keinen Zugriff auf Windows-Authorisierungsdaten. Ob ein UNIX-KDC zur Authentifizierung von Windows NT 5-Workstations genutzt werden kann, ist umstritten. Es existiert eine Aussage von THEODORE TS'O, daß Microsoft plant, das KDC und andere Teile des Domain-Controllers von Windows NT 5 so zu implementieren, daß eine Aufsplittung eines Domain-Controllers auf mehrere Hosts unmöglich ist[15]. Somit könnte ein bestehendes KerberosV5-KDC nicht zur Authentifizierung von Windows-Clients benutzt werden. Andererseits behauptet PETER BRUNDRETT, ein Entwickler von Microsoft, daß die Nutzung eines UNIX-KDC zur Authentifizierung möglich ist, wenn keine Authorisierungsdaten benötigt werden[16]. Sind jedoch Authorisierungsdaten erforderlich, was in den allermeisten Fällen zutrifft, kann das UNIX-KDC definitiv nicht zur Anmeldung genutzt werden.

Sollen bestehende Kerberos V5-Installationen mit Windows NT 5 zusammenarbeiten, bietet sich die Inter-Realm-Authentifizierung an. Dazu muß eine Vertrauensstellung zwischen der bestehenden Realm und der Windows NT 5-Realm geschaffen werden. Microsoft möchte die Inter-Realm-Authentifizierung benutzen, um mehrere Windows NT-Domänen untereinander zu verbinden. Es steht deshalb zu erwarten, daß Inter-Realm-Authentifizierung unterstützt wird und auch mit Nicht-Windows NT-Realms zusammenarbeitet[16]. Wird mit einem TGT einer Nicht-Windows-Realm ein Service-Ticket vom Windows NT 5-KDC angefordert, bemerkt dieses das Fehlen der Authorisierungsdaten und wird versuchen, diese Daten zu generieren. Dazu wird der Principal-Name ausgewertet. Ist dieser Principal nicht in der Datenbank von Windows NT 5 vorhanden, werden Standard-Authorisierungsdaten verwendet.

Inwieweit ein Windows NT 5-KDC zur Interoperabilität mit Kerberos V5-Dienstprogrammen aus anderen Implementationen fähig ist, kann mangels praktischer Tests nicht beantwortet werden. BRUNDRETT behauptet, daß Windows NT 5 in der Lage ist, mit anderen Kerberos V5-Implementationen, wie MIT-KerberosV5, zusammenzuarbeiten.

[15] Theodore Ts'o, „Microsoft ‚embraces and extends' Kerberos V5", ;login:, Official Magazine of the Usenix Association, November 1997
[16] Brundrett, P.: Kerberos Authentication in Windows NT 5.0 Domains, ;login:, Official Magazine of the Usenix Association, Special Issue on Security, May 1998

Dazu wird die Einhaltung der „Interoperability Requirements“ des RFC 1510 vorausgesetzt, in denen bestimmte Verschlüsselungs-Algorithmen, gegenseitige Authentifizierung und andere Ticket-Optionen festgelegt werden, die jede KerberosV5-Implementation einsetzen sollte. Diese Aussage ist glaubhaft, da die Interoperabilität eine Möglichkeit bieten würde, bestehende Kerberos V5-Installationen durch Windows NT 5 abzulösen.

Wenn ein TGT von einem UNIX-Host angefordert wird (z.B. durch „Heimdal“-kinit), sollte das KDC von Windows NT 5 in der Lage sein, die Anforderung zu bearbeiten und ein TGT im Standard-Kerberos V5-Format auszustellen. Das Windows NT 5-KDC wird im TGT Authorisierungsinformationen für den Principal ablegen, die seine Gruppenzugehörigkeiten kennzeichnen. Mit dem TGT können nun weitere Service-Tickets angefordert werden. Dabei werden die Authorisierungsdaten vom Windows-KDC geprüft und in neue Service-Tickets übernommen. Der Service, für den das Ticket ausgestellt wurde, benutzt diese Authorisierungsinformationen zur Zugriffssteuerung. Wichtig ist in diesem Zusammenhang, daß Kerberos V5-Dienstprogramme aus „Heimdal“ oder anderen Implementationen, die nicht von Microsoft stammen, die Authorisierungsdaten von Windows NT 5 im Kerberos V5-Ticket ignorieren und weiterleiten. Ist dieses Verhalten gegeben, was dem Inhalt von RFC 1510 entspricht, steht dem Einsatz von Windows NT 5 als Campus-weites KDC für heterogene Netze nichts im Wege.

Probleme sind dann zu erwarten, wenn Abwärtskompatibilität mit Kerberos V4 und AFS benötigt wird. Es ist sehr unwahrscheinlich, daß Microsoft in seiner KDC-Implementation Unterstützung für AFS-Clients bietet. Ohne diese Unterstützung können AFS-Dienstprogramme, die das RX-Protokoll zur Ticketanforderung benutzen, nicht mehr eingesetzt werden. Ob und wie es möglich sein wird, eine Integration von AFS in Windows NT 5 in der Art, wie sie in der vorliegenden Arbeit in Kapitel 4 beschrieben wurde, vorzunehmen, kann mangels einer Windows NT 5-Installation nicht beantwortet werden.

Es bleibt festzustellen, daß die Möglichkeit gegeben sein könnte, Windows NT 5 für die gesamte Authentifizierung im URZ einzusetzen, wenn die Versprechungen von Microsoft bezüglich Interoperabilität mit bestehenden Realms wahr werden. Kann die Einrichtung eines AFS-Service-Principals erreicht werden, sollte eine Windows NT 5-Domäne „TU-CHEMNITZ.DE“ eingerichtet werden. An dieser Domäne bzw. Realm könnten dann sowohl Windows NT-Benutzer als auch Nutzer von Linux- und Unix-Hosts angemeldet werden.

# 6. Abschließende Bemerkungen

Die Ergebnisse dieser Studienarbeit zeigen, daß es prinzipell möglich ist, den AFS-kaserver durch die „Heimdal“-Implementation von Kerberos V5 zu ersetzen. Die Entwickler der KTH Stockholm haben eine enorme Leistung bei der Implementation von Kerberos V5 erbracht. Die derzeit aktuelle Version ist größtenteils stabil und bietet eine weitreichende Unterstützung für AFS-Clients. Es steht zu erwarten, daß in der fertigen Version von „Heimdal“ eine umfassende AFS-Unterstützung sowie ausgereifte Werkzeuge zur Datenbankverwaltung und –konvertierung zur Verfügung stehen. Trotz der umfangreichen Funktionalitität von „Heimdal“ sind in der aktuellen Version einige Kritikpunkte enthalten:

- Kein dediziertes Werkzeug zur Datenbankkonvertierung
- Stabilitätsprobleme; Abstürze mit „segmentation fault“ sind öfters aufgetreten
- Kein „Login“-Dienstprogramm im Paket enthalten
- nur rudimentäre Werkzeuge zur Administration von Keytabs und Principal-Datenbank
- fehlende Dokumentation

Das Rechenzentrum der TU Chemnitz ist eine zentrale Einrichtung der Universität. Das Funktionieren des AFS und seine Sicherheit sind daher von großer Bedeutung für die gesamte TU Chemnitz. Aufgrund der aufgeführten Mängel ist es aus meiner Sicht nicht empfehlenswert, das „Heimdal“-Paket in seiner jetzigen frühen Implementierungsphase als Ersatz für AFS-kaserver in der Zelle „tu-chemnitz.de“ zu installieren. Sollte „Heimdal“ als fertige Version verfügbar sein, kann es jedoch durchaus zur Ablösung des AFS-kaserver benutzt werden. Alle Zielvorgaben, wie Weiterbenutzung der bestehenden Authentifizierungsmechanismen oder die Möglichkeit der Übernahme bestehender AFS-Nutzer, werden von „Heimdal“ erfüllt. Zur Vorbereitung der Migration wäre es sinnvoll, bis zur Ankündigung der Final-Version von „Heimdal“ eine Test-Realm mit AFS-Service im URZ zu betreiben. Damit wäre die TU Chemnitz die erste Einrichtung in Deutschland, die eine AFS-Zelle mit „Heimdal“ betreibt.

Eine weitere Möglichkeit zur Migration würde die Lockerung oder Aufhebung des Kryptographie-Exportverbotes durch die USA bieten. Die Anzeichen für diesen Schritt mehren sich. In diesem Fall könnte das ausgereifte MIT-KerberosV5 zusammen mit dem „AFS-Migration-Kit“ von Transarc zum Übergang zu Kerberos V5 benutzt werden. Diese Kombination bietet mehr Funktionalität und bessere AFS-Unterstützung als „Heimdal“.

Die neue Version 5 des Betriebssystems Windows NT könnte einen weiteren Ansatz für die Ablösung des AFS-kaserver bieten. Allerdings stehen noch Fragen bezüglich Interoperabilität und Abwärtskompatibilität offen, die erst nach Erscheinen der fertigen Version des Produkts beantwortbar sind. Der Vorteil dieser Lösung wäre eine URZ-weit einheitliche Authentifizierungslösung, die den Administrationsaufwand senken könnte. Als nachteilig kann der hohe Preis und die Abhängigkeit von Microsoft angesehen werden.

Mein besonderer Dank gilt Herrn Thomas Müller, der als Betreuer dieser Arbeit immer ein offenes Ohr für Fragen und Probleme hatte. Ohne seine Hilfe wäre diese Arbeit nicht erfolgreich abgeschlossen worden.

## Anhang – Die Konfigurationsdatei „krb5.conf“

Die Konfigurationsdatei „krb5.conf“ wird auf jedem Host in der „Heimdal“-Realm benötigt, um Informationen über die Namen der Datenbankserver der verschiedenen Realms zu liefern. In der Realm „tu-chemnitz“ hätte diese Datei folgenden Inhalt:

```
[libdefaults]
        default_realm = TU-CHEMNITZ.DE          enthält den Namen der lokalen Realm
        v4_instance_resolve = yes               gibt an, ob V4-Instanznamen mit DNS
                                                aufgelöst werden sollen
[realms]
        TU-CHEMNITZ.DE = {
                kdc = zuse.hrz.tu-chemnitz.de   Namen der KDC, auch IP-Adressen
                kdc = aetius.hrz.tu-chemnitz.de zulässig.
                kdc = phoenix.hrz.tu-chemnitz.de
                v4_name_convert = {             Parameter zur Abbildung v. V4-Name
                        host = {                nach V5-Name
                                ftp = ftp
                                rcmd = host
                                pop = pop
                        }
                }
                default_domain = hrz.tu-chemnitz.de
        }
[domain_realm]
        .hrz.tu-chemnitz.de = TU-CHEMNITZ.DE    gibt an, welche Realm f. welche
                                                Domain benutzt wird
[logging]
        kdc = FILE:/var/heimdal/kdc.log         Position des Logfiles d. KDC
        kdc = SYSLOG:INFO
        default = FILE:/var/heimdal/kdc.log     Default-Logfile f. alle Kerberos-
                                                Dienste (kadmind, kpasswdd,...)
```

# LITERATURVERZEICHNIS


Brundrett, P.: Kerberos Authentication in Windows NT 5.0 Domains, in: ;login:, Official Magazine of the Usenix Association, Special Issue on Security, May 1998

Coulouris, G., Dollimore, J., Kindberg, T.: Distributed Systems – Concepts and Design, ISBN 0-201-62433-8, Addison-Wesley, 1996

Cox, Phil: Windows NT 5.0: Integration Friendly?, in: ;login:, Official Magazine of the Usenix Association, Dezember 1997

Denning, D., and G. Sacco: Time stamps in Key Distribution Protocols, Communications of the ACM, Vol. 24 (8), pp. 533-536, August 1981.

Kaufmann, C., Perlman, R., Spencer, M.: Network Security: Private Communication in a Public World, Prentice Hall, Englewood Cliffs, New Jersey 1995

Kohl, J., Neuman, C.: The Kerberos Network Authentication Service (V5), RFC 1510, September 1993

Miller, S., Neuman, C., Schiller, J., and J. Saltzer: Section E.2.1: Kerberos Authentication and Authorization System, M.I.T. Project Athena, Cambridge, Massachusetts, December 21, 1987.

Needham, R., and M. Schroeder: Using Encryption for Authentication in Large Networks of Computers, Communications of the ACM, Vol. 21 (12), pp. 993-999, December 1978.

Neumann, C., Tung, B., Wray, J., Medvinsky, A., Hur, M., Trostle, J.: Public Key Cryptography for Initial Authentication in Kerberos, draft-ietf-cat-kerberos-pkinit-03.txt

Neumann, C., Tung, B., Wray, J., Medvinsky, A., Hur, M., Trostle, J.: Public Key Cryptography for Initial Authentication in Kerberos, draft-ietf-cat-kerberos-pkinit-03.txt

Ts'o, Theodore: Microsoft „embraces and extends" Kerberos V5, in: ;login:, Official Magazine of the Usenix Association, November 1997