Seminar IT-Sicherheit

Das Authentifikationsverfahren

Kerberos

Autor: Tibor Dekany

Leitung: PD Dr. Rolf Oppliger

7.12.2001, Universität Zürich, Institut für Informatik

Inhaltsverzeichnis

1 Einführung

1.1 Was heisst Kerberos?

In der Griechischen Mythologie ist Kerberos das Ungeheuer in Form eines dreiköpfigen Hundes, welcher die Tore zur Unterwelt, dem Reich von Hades, bewacht und die Toten zwar hinein, nicht aber wieder heraus lässt.

Hier geht es aber um das Sicherheitsprotokoll gleichen Namens, einem Authentifizirungsprotokol für Netzwerke. Es wurde im Rahmen des Projektes "Athena" für den internen Einsatz am MIT von Steve Miller und Clifford Neumann entwickelt (Version 1−4). Erst die Version 4 wurde 1987 veröffentlicht [KRBAS] [KRBAS2]. Die Spezifikation der zur Zeit aktuelle Version 5 wurde 1993 im RFC 1510 [RFC1510] festgelegt und gilt somit als Internetstandard. Am IETF ist für Kerberos die Kerberos Working Group [KRBWG] zuständig. Neben freie verfügbaren Open Source Implementationen gibt es auch eine Reihe kommerzieller Produkte.

Das Ziel von Kerberos ist, in einem offenen, ungeschützten Netzwerk, Kommunikationspartner in einer Client/Server Umgebung, also User und Server[1], eindeutig und sicher identifizieren zu können. Diesen Prozess nennt man Authentifizierung, was nicht mit Autorisierung zu verwechseln ist. Autorisierung gibt einer (zuvor identifizierten) Entität die Berechtigung auf den Zugriff einer bestimmte (eventuell ebenfalls zu authentifizierenden) Ressource.

Ein Beispiel zur Verdeutlichung: Der Staat spielt die Rolle des unabhängigen Authentifizierungsservers. Er gibt Identitätskarten an die jeweiligen "Benutzer" ab, und gibt ebenfalls z.B. Banken eine Bewilligung. Wenn nun jemand an einem Bankschalter Geld von seinem Konto abheben möchte, so muss er sich mit der Identitätskarte (Bankkarte reicht nicht) ausweisen: die Bank vertraut somit betreffend Identität einer dritten Instanz, dem Staat. Dies ist die Authentifikation, damit wurde er identifiziert, damit ist aber erst sichergestellt, dass er derjenige ist, den er zu sein vorgibt, und nicht, ob er wirklich Geld von einem Konto abheben darf. Dies hat die Bank zu bestimmen, diesen Schritt nennt man

1 Es ist schwierig nur von "Client" und "Server" zu reden. In der Regel möchte ein User etwas von einem Server haben (z.B. Mail abholen). Der User (Person) veranlasst einen Client (Emailprogramm auf seiner Maschine) mit einem (Email−) Server zu kommunizieren. User, Client und Server werden nun Entitäten bzw. Kommunikatinospartner genannt (englisch: principal). Diese Entitäten treten selber als Client in Kontakt mit dem Kerberosserver.

Autorisation.

1.2 Warum Kerberos?

Auf einer einzelnen, nicht vernetzten Maschine regelt das Betriebssystem (die Rede ist von echten Multiusersystemen, nicht etwa DOS oder Windows 9x) die Zugriffsrechte auf die vorhandene Ressourcen und sorgt dafür, dass die Benutzer beim Login korrekt identifiziert werden, indem es das eingegebene Passwort mit seiner internen, von den anderen Benutzern nicht zugänglichen Passworttabelle vergleicht. Bei Übereinstimmung kann es so davon ausgehen, dass es sich wirklich um den angegebenen User handelt, solange ausschliesslich nur der Benutzer und das Betriebssystem im Besitz des Passwortes sind. Das Passwort ist also ein nicht öffentlicher Schlüssel[2].

In einem Netzwerk entstehen hier zwangsläufig Probleme, wenn ein Benutzer, welcher an verschiedenen Arbeitsstationen arbeiten kann, auf verteilte Dienste zugreifen möchte. Ein Server kann zwar verlangen, dass das Betriebssystem auf dem Client den Benutzer identifiziert (nach diesem Verfahren arbeiten die berüchtigten Unix R−Befehle), kann aber der Antwort vom Client nur dann trauen, wenn wir uns in einem abgeschotteten Netzwerk befinden, in der alle Maschinen von einer zentralen Stelle verwaltet werden und sich daher gegenseitig vertrauen können. Dies war der Hauptgrund, warum am MIT, wo die einzelnen Institute selber für ihre Computer zuständig sind, Kerberos entwickelt wurde.

Daher führt kein Weg daran vorbei, dass zur Authentifizierung sich alle Clients bei allen Services ausweisen müssen. Dies ist aber auf herkömmlicher Weise umständlich, da so alle Services jeweils alle Benutzer und deren Passwörter verwalten müssten. Ausserdem müsste der Benutzer bei jedem Zugriff auf einen Service sein Passwort eingeben, was nicht nur unbequem ist, sondern zudem das Passwort jedesmal ungeschützt über das Netzwerk wandert.

1.3 Die Anforderungen an Kerberos

Kerberos wurde nun entwickelt, um folgenden Anforderungen zu genügen:

1. Es muss sicher sein. Ein potentieller Feind, der z.B. das Netzwerk abhört, darf aus den

2 Häufig werden die Begriffe Passwort und Schlüssel synonym verwendet, was im Grunde falsch ist. In Wahrheit wird der Schlüssel mit Hilfe einer Einwegfunktion aus dem Passwort generiert, aus dem Schlüssel kann man somit nicht auf das Passwort schliessen.

vorbeifliessenden Daten keine Informationen gewinnen können oder sich mit "geklauten" Daten als einen anderen User ausgeben können.

2. Es muss zuverlässig sein: viele Services und User sind vom Authentifikationsservice von Kerberos abhängig.

3. Es muss transparent sein: idealerweise sollte der Benutzer von der Authentifizierung nichts merken und wie gewohnt arbeiten können, ohne sich an andere oder sogar mehrere neue auszuführende Schritte gewöhnen müssen.

4. Es muss skalierbar sein.

1.4 Was ist Kerberos?

Kerberos ist ein "trusted third−party authentication service", also ein unabhängiger Authentifizierungsservice, dem alle Kommunikationspartner eines bestimmten Benutzerkreises uneingeschränkt vertrauen. Aber es ist nicht mehr. Es kann also einer Entität nur sagen, ob die andere Entität auch die ist, für die sie sich ausgibt. Diesem Urteil müssen dann aber auch alle Beteiligte glauben. D.h. bei einer Client/Server Applikationen muss Kerberos alle Clients, aber auch alle Server identifizieren können, muss somit alle privaten Schlüssel aller Entitäten kennen und hat diese darum in Klartext[3] gespeichert. Bei den Usern wird der Schlüssel mit Hilfe einer Einwegfunktion aus dem Passwort generiert. Bei den Servern hingegen wird der Schlüssel selbst, wie auch beim Kerberosserver, in Klartext gespeichert (da auch der Server seinen eigenen privaten Schlüssel kennen muss). Dies ist eine der grössten Schwächen von Kerberos bzw. aller anderen Verfahren auch, wo symmetrische Algorithmen vorkommen: den privaten Schlüssel müssen mindestens 2 Parteien kennen. Daher nennt man dieses System auch BigBrother System, da dem Kerberosserver alle vertrauen müssen.

Für die Autorisierung selbst ist nach wie vor der Server selber zuständig.

2 Funktionsweise

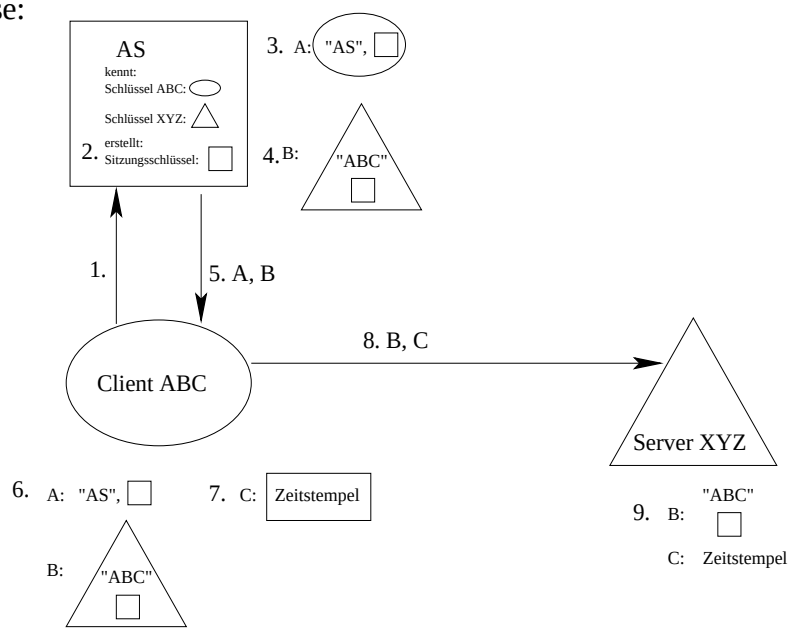
2.1 Überblick

Kerberos basiert auf das 1978 von Needham und Schroeder veröffentlichte "Key distribution protocol" [NEED]. Dies ist das erste Protokoll, welches auf einen zentralen

3 Selbstverständlich sind die Schlüssel mit einem Masterkey verschlüsselt in der Datenbank abgelegt.

"Schlüsselvergabeserver" basiert und private key Algorithmen einsetzt. Dieser Key Distribution Server (KDC), dem sowohl Client wie auch Server vertrauen, liefert nach erfolgreicher Authentifikation beiden Kommunikationspartnern einen Sitzungsschlüssels (session key).

Die Authentifizierung bei Kerberos funktioniert vereinfacht [DIALOG][MORON] folgender Weise:



1. Der User schickt dem Authentifizierungs−Server (AS) folgende Nachricht: "Ich, User ABC, möchte mit dem Mailserver XYZ kommunizieren.".

2. Der AS generiert nun einen Sitzungsschlüssel, den er sowohl dem User ABC, als auch dem Server XYZ zukommen lassen wird, damit diese mit Hilfe dieses Schlüssels sicher kommunizieren können.

3. AS erstellt nun eine erste Nachricht (für den User ABC) bestehend aus seinem eigenen Namen "AS" und dem Sitzungsschlüssel mit dem Schlüssel des Users. Diese Nachricht A verschlüsselt er mit dem Schlüssel des Users.

4. Der AS erstellt danach eine zweite Nachricht (für den Server XYZ bestimmt) bestehend aus dem Namen des Users und dem Sitzungsschlüssel. Diese Nachricht B (Ticket genannt) wird nun mit dem Schlüssel des Servers XYZ verschlüsselt.

5. Der AS schickt nun beide Nachrichten an den User.

6. Der User ABC kann nun nur die Nachricht A entschlüsseln, und entnimmt den Sitzungsschlüssel. Dass er die Nachricht richtig entpacken konnte, sieht er daran, dass die Nachricht den Namen des AS enthält.

7. Die Nachricht B kann der User nicht entschlüsseln, da diese mit dem Schlüssel des Servers XYZ verschlüsselt ist. Er erstellt dafür eine Nachricht C, welche einfach die aktuelle Zeit enthält. Diese Nachricht wird mit dem Sitzungsschlüssel verschlüsselt.

8. Der User schickt nun die Nachricht B (Ticket) und C (Authentifikator) dem Server XYZ.

9. Zunächst kann der Server nur die Nachricht B mit Hilfe seines Schlüssels öffnen. Daraus erhält er nun den Namen des Users ABC (und kann jetzt entscheiden, ob dieser überhaupt berechtigt ist, diesen Service zu nutzen: dies ist die Autorisation). Ausserdem findet er in der Nachricht den Sitzungsschlüssel und kann damit die Nachricht C entschlüsseln. Mit diesen Nachrichten bzw. Ausweisen wurde dem Server die Identität des Users bewiesen. Falls gewünscht ist, dass sich der Server beim User ebenfalls ausweisen soll, so schickt der Server eine mit dem Sessionkey verschlüsselte Nachricht dem User zurück, diese beinhaltet den Namen des Servers und dem aus Nachricht C erhaltenen Zeitstempel. Somit können sich User und Server gegenseitig identifizieren und mit Hilfe der Sitzungsschlüssel miteinander kommunizieren.

   Kerberos unterstützt 3 Arten der Kommunikation zwischen Client und Server:

1. Einmalige Authentifikation: danach werden die Netzwerkadressen freigeschaltet und weitere Nachrichten fliessen ohne weitere Überprüfung.

2. Überprüfung der Authentizität pro Nachricht: die Nachrichten selbst fliesst dann immer noch im Klartext über das Netzwerk ("safe message").

3. Zu 2. zusätzliche Verschlüsselung der Nachricht: die Nachricht wird mit dem Sitzungsschlüssel verschlüsselt ("private message").

2.2 Voraussetzungen eine sichere Umgebung

- Kerberos bietet keinen Schutz gegen DoS (Denial of Service) Attacken, diese müssen anderweitig abgewehrt werden. Da alle Benutzer beim Login und bei Benützung von Services mit dem AS kommunizieren müssen, kann ein Ausfall des AS fatale folgen

haben.

- Die Kommunikationspartner müssen ihre eigenen privaten Schlüssel geheim halten. Mit dem Passwort des Users kann sich ein Angreifer als diesen User ausgeben. Die Maschinen, auf denen Services laufen, und natürlich auch der Kerberosserver, müssen gesichert sein (und zwar nicht nur Softwareseitig z.B. durch Firewalls, sondern auch der Zutritt zur Maschinenhardware / Konsole), damit niemand auf die Schlüssel Zugriff hat.
- Passwortattacken können von Kerberos nicht verhindert werden. Daher ist es wichtig, dass die User keine einfache Passworte wählen. Ein Angreifer kann mit einer abgefangenen Nachricht vom AS (Nachricht A) sogar offline das Passwort des Users herausbekommen, und dies geht bei einfachen Passworten und guten Dictionaries relativ schnell. Desshalb soll auch bei Kerberos der User ab und zu sein Passwort ersetzen.
- Die Uhren in allen Computern einer Kerberosdomäne müssen mehr oder weniger synchronisiert sein. Denn der Authentifikator ist in der Regel nur 5 Minuten lang gültig. Dies impliziert aber, wenn die Uhren via Netzwerk (z.B. via NTP) eingestellt werden, dass diese Synchronisation ebenfalls vor Angreifern gesichert werden muss.
- Vorsicht ist auch bei der Wiederverwendung von Benutzernamen angebracht. Sollte ein User den Benutzernamen seines Vorgängers erhalten (z.B. da beide gleich heissen), ist es möglich, dass der neue User auf gewisse (noch nicht angepasste) Services seines Vorgängers zugreifen kann, da die Autorisierung vom Server und nicht von Kerberos vorgenommen wird.

2.3 Erfüllung der Anforderungen

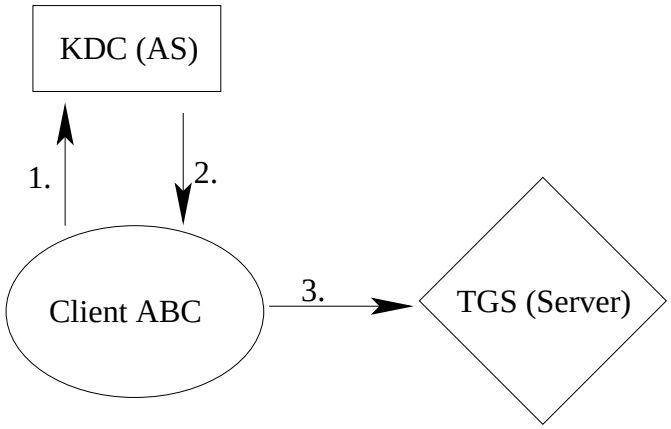
Unter Berücksichtigung dieser o.g. Punkte kann Kerberos die in Kapitel 1.3 genannten Anforderungen 1. und 2. erfüllen: es werden keine Passwortdaten klartext übermittelt.

2.3.1 Transparenz

Wie kann aber Kerberos transparent in ein bestehendes System eingeführen werden? Auf der Serverseite müssen alle Applikationen, welche Kerberos unterstützen sollen, gegen solche ausgetauscht werden. Wird ein Programm um die Unterstützung von Kerberos erweitert, so nennt man dieses Programm "kerberized". Hierbei existieren bereits viele kerberisierte standard Serverprogramme, z.B. POP3 und IMAP Server, telnetd, ftpd, etc.

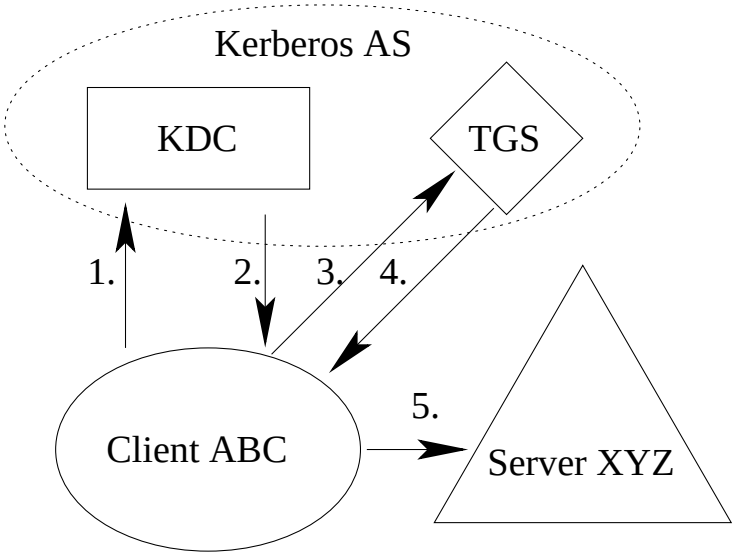
Aber auch an den Workstations der User müssen Änderungen vorgenommen werden, d.h. auch diese Maschinen müssen kerberisiert werden, zu den Servern müssen auch die entsprechenden Clientprogramme aufgespielt werden. Es müssen aber auch neue Programme installiert werden. Das wichtigste ist wohl "kinit", mit diesem Programm kann der User seine Ausweise und sein Ticket vom Kerberosserver anfordern. Des weitern braucht er neue Administrationsprogramme, z.B. eines um sein Passwort zu ändern (dieses muss ja dann nicht nur lokal, sondern auch auf dem AS geändert werden). Glücklicherweise muss hier aber der User nicht viel neues dazulernen, denn die kerberisierten Clientprogramme funktionieren völlig transparent, und das standard Passwortänderungsprogramm des Betriebssystems wird einfach gegen das von Kerberos ausgetauscht.

Was aber ist mit "kinit"? Muss der User vor jedem Zugriff auf einen Server dieses Programm aufrufen? Die Lösung dieses Problemes sieht folgendermassen aus: kinit wird das erste Mal aufgerufen, wenn sich der Benutzer einloggt. Denn das Loginprogramm wurde ebenfalls durch eine kerberisierte Version ersetzt, damit sich auch nur der wahre User einloggen kann. Damit nun der User nicht jedesmal für jede Verbindung zu einem Server kinit aufrufen muss (und jedesmal sein Passwort eingeben müsste), wurde der AS in zwei logische Teile aufgeteilt [KRBEVO]: in einen Schlüsselverteilservice (Key Distribution Center KDC) und in einen Ticketverteilservice (Ticket Granting Server, TGS).



Der TGS ist nun als normaler Server zu verstehen. Die Idee ist, dass der KDC nur ein einziges mal, beim Login (1.) mit "kinit", dem User ein Ticket für die Benutzung des TGS gibt (das sogenannte Ticket Granting Ticket TGT (2.)). Somit erhält der User und der TGS

einen Sitzungsschlüssel und können mit Hilfe von diesem sicher kommunizieren.



KDC und TGS sind nur logisch getrennt, sie befinden sich physisch auf der selben Maschine. Im Gegensatz zu einem normalen, entfernten Server, haben hier aber sowohl KDC als auch TGS Zugriff auf die Schlüsseldatenbank. Die Idee ist nun, dass der Client, möchte er mit einem anderen Server kommunizieren, sein Ticket (4.) nicht vom KDC, sondern vom TGS anfordert, und zwar über die bereits bestehende, sichere Verbindung! Damit muss der User nicht "kinit" starten und muss somit auch nicht sein Passwort wieder eingeben. Der Vorteil hierbei ist, dass die angeforderten Tickets hier nicht mehr mit dem Schlüssel des Users verschlüsselt sind, sondern mit dem Sitzungsschlüssel, welchen der User im TGT erhalten hat. Dieser verfällt, sobald sich der User vom System abmeldet, spätestens aber nach acht Stunden (danach ist ein erneuter Login nötig, oder die manuelle Ausführung von "kinit").

Somit ist das System transparent, denn der User muss sich nicht mehr selber um die Ticketanforderungen kümmern, dies geschieht automatisch im Hintergrund.

2.3.2 Skalierbarkeit

Kerberosserver haben einen bestimmten Zuständigkeitsbereich, eine Domäne (englisch: realm). Es wird sogar empfohlen, eine Kerberosdomäne gleich wie die entsprechende Internetdomain zu wählen. Um Internetdomain und Kerberosdomäne auseinanderhalten zu können, sollen die Kerberosdomänen in Grossbuchstaben geschrieben werden (z.B. Kerberosdomäne für das ifi: "IFI.UNIZH.CH".

Ein normaler User wird somit folgendermassen identifiziert: username@DOMAIN, also z.B. tibor@IFI.UNIZH.CH. Für User an Domänen können auch mehrere Instanzen definiert werden, dies wird z.B. benötigt, wenn für gewisse Aufgaben bestimmte Privilegien erforderlich sind, z.B. als Systemadministrator: tibor/admin@IFI.UNIZH.CH. Der Name besteht also aus "Basisname/Instanz@DOMÄNE".

Alle Server müssen ebenfalls eindeutig identifiziert werden können, diese erhalten einen Namen in folgender Form: host/domain@DOMAIN, z.B. ftp/ifi.unizh.ch@IFI.UNIZH.CH

2.3.2.1 Master / Slave Server

Kerberos bietet die Möglichkeit, bei grossen Domänen, mehrere gespiegelte Kerberosserver aufzustellen. Hierzu muss einer als Master, die anderen als Slave konfiguriert werden. Die Schlüsseldatenbank wird hierbei sporadisch gespiegelt. Diese Methode hat aber den Nachteil, dass nicht alle Server immer präzise auf dem gleichen Stand sind. Ändert z.B. der User sein Passwort, so wird die Änderung auf dem einen Slave sofort vorgenommen. Dieser sorgt auch dafür, dass der Master ebenfalls den neuen Schlüssel erhält. Ein anderen Slave wird aber erst viel später, nämlich bei der nächsten Spiegelung, den neuen Schlüssel erhalten.

2.3.2.2 Inter Domain Authentifizierung

Kerberos erlaubt auch die Authentifizierung über verschiedene Domänen hinweg. Jede Domäne hat einen zuständigen Kerberosserver mit KDC und TGS. Es wird nun ein weiterer Zwischenschritt benötigt, ähnlich wie in Kapitel 2.3.1 der Zwischenschritt "TGS" eingeführt wurde. Möchte ein User einer Domäne mit einem Server einer anderen Domäne kommunizieren, so muss der Kerberosserver der Userdomäne den TGS der entfernten Domäne kennen, dem "remote TGS", oder kurz: RTGS. Jetzt muss der User zuerst, wie gehabt, mit Hilfe des TGT Zugang zum eigenen TGS erhalten, dann vom TGS ein Ticket für den RTGS anfordern, und schlussendlich kann der User vom RTGS ein Ticket für den gewünschten Server der anderen Domäne anfordern. Dieses System kann hierarchisch aufgebaut werden, ähnlich wie das Internet Domain Name System (DNS). So muss nicht jeder TGS jeden RTGS direkt kennen, sondern nur die der direkt über− und untergeordneten Domänen.

3 Gebrauch von Kerberos

3.1 Administration (Unix)

3.1.1 Kerberosserver

Der Kerberosserver muss, wie schon unter Kapitel 2.2 erwähnt, gut geschützt sein. Das Aufsetzen des Servers selbst ist relativ einfach, bei den meisten Linuxdistributionen ist Kerberos direkt als fertiges Binarypaket vorbereitet, ist daher schnell zu installieren und muss nur noch konfiguriert werden. Es müssen die Domänenzuständigkeiten definiert werden, die Datenbank für die User, Server und deren Schlüsseln erstellt werden und schlussendlich die beiden Serverprozesse "krb5kdc" und "kadmind" gestartet werden. Mit "kadmin" können dann die User, Server und Schlüssel eingegeben werden.

3.1.2 Kerberisierung

Es müssen nun alle Server− und Clientprogramme, welche mit Kerberos laufen sollen, angepasst werden. Glücklicherweise sind viele Standardprogramme bereits vorhanden, wie z.B. die Login−applikation, die Unix r−Befehle (rlogin, rsh, ...), POP, FTP Server etc. Nicht kerberisierten Programme müssen an dieser Stelle angepasst und somit umprogrammiert werden. In der Kerberosdistribution sind einfache Programmbeispiele enthalten, wie die Kerberos API eingebunden werden kann.

3.2 User (Unix)

Für den User ist Kerberos ziemlich transparent. Das TGT wird in der Regel gleich beim Login übermittelt. Manuell kann der User ein TGT mit dem Befehl "kinit" anfordern, hier muss er sein Passwort eingeben. Wenn er jetzt auf einen Server zugreifen möchte (z.B. rlogin), so führt er einfach die kerberisierte Version von rlogin aus. Mit dem Befehl "klist" können die im Hintergrund erhaltenen Tickets im Ticketcache ausgegeben werden. "kdestroy" wir automatisch beim Logout aufgerufen und löscht den Ticketcache. Dieser Befehl kann bei Bedarf auch manuell ausgeführt werden.

3.3 Kerberos unter Windows 2000

Standardmässig wird bei Windows NT (insbesondere bei den alten Versionen) das NT Lan Manager Protokoll verwendet, welches heute als Sicherheitsrisiko zu betrachten ist. Sobald aber eine ganze NT Domäne mit Hilfe von Active Directory (AD) verwaltet werden

soll, und somit keine Kompatibilität zu den alten NT Versionen mehr gewahrt werden muss, wird standardmässig Kerberos [MSKRB] zur Benutzerauthentifizierung verwendet. Somit sind sowohl Kerberos als auch AD privilegierte Dienste, da beide mit geheimen Informationen (z.B. den privaten Schlüsseln) umgehen können müssen, denn Kerberos ist für die Authentifikation zuständig, während AD für die (u.a. Benutzer−) Administration verantwortlich ist.

3.3.1 Erweiterungen des Protokolls

Microsoft hat nun eigenmächtig das standard Kerberos Protokoll erweitert, indem es Autorisierungsdaten in einem nach dem Standard reservierten, (noch) nicht verwendetem Feld verschickt. Leider hält es Microsoft nicht für nötig, seine Implementation offenzulegen und unterminiert somit den offenen Standard [HEISE].

Es ist natürlich schön, dass damit nun eine Authentifizierungs− und Autorisierungs Infrastruktur entstanden ist (AAI), aber dies bedeutet zugleich, dass in heterogenen Netzen zwangsweise Microsoft Kerberosserver verwendet werden müssen. Z.B. wird so die Integration des frei erhältlichen NT−Servers Samba in eine AD Struktur erschwert.

Literaturverzeichnis

NEED: Needham, R., and M. Schroeder, Using Encryption for Authentication in Large Networks of Computers, Communications of the ACM, 1978, 993−999
MSKRB: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/prodtech/kerbtech.asp
MORON: http://www.isi.edu/gost/brian/security/kerberos.html
DIALOG: http://web.mit.edu/kerberos/www/dialogue.html
KRBEVO: * John T. Kohl, B. Clifford Neuman, and Theodore Y. T'so, The Evolution of the Kerberos Authentication System. In Distributed Open Systems, IEEE Computer Society Press, 1994, 78−94
HEISE: http://www.heise.de/newsticker/data/odi−29.02.00−000/
KRBAS2: J. G. Steiner, B. Clifford Neuman, J.I. Schiller, Kerberos: An Authentication Service for Open Network Systems, 1988
KRBAS: Miller, S., Neuman, C., Schiller, J., and J. Saltzer, E.2.1: Kerberos Authentication and Authorization System, 1987
KRBWG: http://www.ietf.org/html.charters/krb−wg−charter.html
RFC1510: http://www.ietf.org/rfc/rfc1510.txt