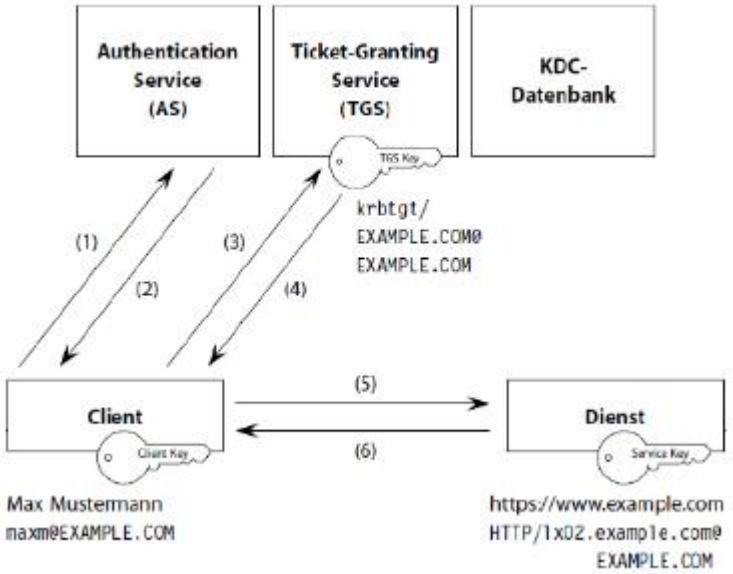
NR	Message Name		Ticketname/KeyName	verschlüsselt mit	Schlüssel ist bekannt				Erklärungen
					Clien	KDC		Server	
						AS	TGS	Dienst	
1	**AS_REQ (1)** **Authenticati** Client → KDC **Service Request**		username. Das Passwort wird in (2) benötigt						Der AS_REQ wird dem AS geschickt. Dieser Request enthält den Principal-Namen des Clients und den Prinzipal-Namen des TGS
2	**AS_REP (2)** **Authentication Service Reply** Client ← KDC	**AS_REP Client-Teil**	TGS_$S_{A,KDC}$ (Session Key) +exp.Time +TGS Service Name	K_A -priv. Langzeitschlüssel Client	X	X			Mit dem TGS-Session-Key kann der Client seine Identität dem KDC beweisen, weil nur der Client und der KDC diesen Schlüssel kennen {TGS_$S_{A,KDC}$, expiration time, TGS Service Name,.....}. K_A Dieser Client-Teil des AS_REP kann der Client mit seinem Passwort entschlüsseln und kann somit den TGS_Session-Key aus der Meldung extrahieren.
		AS_REP TGT-Teil	TGT= TGS_$S_{A,KDC}$ + expTime + Prinzipal-Name des Clients	K_{KDC} =Langzeitschlüssel des		X			Der Client kann das TGT nicht entschlüsseln, weil er K_{KDC} nicht kennt. Somit kann der Client das TGT nicht manipulieren und vorallem nicht den TGS-Session Key verändern. Was würde passieren, wenn er das könnte? {TGS_$S_{A,KDC}$, expiration time, Client Name,....}. K_{KDC}.
									Den TGS-Session Key (TGS_$S_{A,KDC}$) und das TGT in seiner verschlüsselten Form speichert der Client in seinem Credential-Cache (klist) ab !
3	**TGS_REQ (3)** **Ticket Granting Server Request** Client → KDC	**besteht aus 4 Elementen**	**AUTHENTICATOR** enthält: **Client-Prinzipal-Name** **Timestamp** **Checksumme**	TGS_$S_{A,KDC}$ (SessionKey) Diesen Key kennt der Client aus Schritt 2 (AS_REP)	X				Da es sich beim TGS_REQ um einen kerberisierten Zugriff handelt, wird das TGT und der Authentikator gesendet. Der TGS prüft diese Angaben. Wenn ok, dann ist der Client authentifiziert und ....
	Dieser REQ kommt zustande, wenn der Client auf einen kerberisierten Dienst zugreifen will. Dafür benötigt er ein Ticket vom TGS!		TICKET GRANTING TICKET **SERVICE NAME (Dienst im Net** **EXPIRATION TIME des TGT**	K_{KDC} =Langzeitschlüssel des			X		
4	**TGS_REP (4)** **Ticket Granting Server Reply**	**Client-Teil**	**CLIENT TICKET** enthält: **Principal-Name des Service** **Service-Session-Key (Service_K_{AB})** **Expiration Time**	TGS_$S_{A,KDC}$ (-SessionKey)	X		X		... er stellt dann einen neuen Session Key (=Service-Session-Key) für Client und Service. Der TGS entnimmt der KDC-Datenbank den Langzeitschlüssel des Services (Service Key K_B). Der Client-Teil wird hier nicht mit einem Langzeitschlüssel verschlüsselt, sondern mit einem Kurzzeitschlüssel, dem TGS-Session-Key, welcher bei jeder Neuanmeldung immer wieder generiert wird. Der Client kann somit ohne weitere Interaktionen mit diesem Key, welcher in seinem Cache ist, Entschlüsselungen von weiteren Service-Session-Keys vornehmen.
	Das TGS-REP hat 2 Anteile								Er muss also (1) und (2) nicht mehr durchführen.
	Client ← KDC	**Service-Tic**	**SERVICE TICKET** Service-Session-Key (Service_K_{AB}) Client Name Expiration Time	K_B			X	X	Das in Schritt (2) eingegebene Passwort genügt. **SSO wird mit diesem Trick möglich !**
	AP_REQ (5) **Application Server** Client → Server/Service		**AUTHENTICATOR** Timestamp Checksumme	Service_$K_{A,B}$	X		X	(X)	Aus dem Service Ticket kann der Server den Service-Session-Key entnehmen. Mit diesem Key ist er in der Lage den Authenticator zu entschlüsseln, deshalb (X). Den Service-Session-Key kennen nur Dienst, Client und KDC. Dieser Key ist also ein gemeinsames Geheimnis zwischen Dienst und Client. Wenn der Service nun mit diesem Key den Authenticator entschlüsseln kann, hat der Client seine Authentizität bewiesen. Kein anderer als der Client, hätte diesen Authenticator generieren können !
			SERVICE TICKET Service Session Key Service_$K_{A,B}$ Client Name Expiration Time	K_B			X	X	
6	**AP_REP (6)** **Application Serve** Client ← Server/Service		TIMESTAMP	Service_$K_{A,B}$	X		X	X	Optional: Falls der Client sicher sein will, dass es der richtige Server ist. Z.B. Telebanking-Server --> wenn der Client den Zeit-stempel mit Service_$K_{A,B}$ entschlüsseln kann, weiss er, dass dieser vom "richtigen" Server gesendet wurde, weil nur dieser den Schlüssel Service_$K_{A,B}$ mittels K_B kennt.
	OPTIONAL !								

Begriff	Beschreibung
Directory Service	Verzeichnissdienst für Netzwerkobjekte (User, Drucker, Mail)
Authentisierung	Nachweis der eigenen Identität (Kerberos´s hauptaufgabe
Autorisierung	Berechtigungen des Benutzers auf ressourcen im Netzwerk
SSO	Zentrale Anmeldung Netzwerkweit bis ablauf von Ticket
Daten Speicherung passiert im	Lagnzeitschlüssel (Keytab datei) und TGT + TGS (Credential Cache) bei client

Begriff	Ausgeschrieben
DIT	Directory Information Tree
DN	Distinguished Name
OU	Organizational Unit
CN	Common Name
KDC	Key Distribution Center auch genannt Trusted Third Party
DC	Domain Component
TGS	Ticket Granting Service/Server, im TGS Session Key wird die authentizität des Clients gespeichert
TGT	Ticket Granting Ticket, das TGT besteht aus dem TGS Session Key + expTime + Principal Name des client
RDN	Relative Distinguished Names
Sessionkey	Ist temporär
Servicekey	Bleibt gleich



Die gesamte Kommunikation mit dem KDC erledigt der Client. Eine Kommunikation zwischen dem Dienst und dem KDC findet nicht statt. Das KDC muss für den Dienst also während der Client-Authentifizierung nicht erreichbar sein.

Es gibt zwei Varianten, wie ein Client Tickets beziehen kann. Eine davon ist die Verwendung des Authentication
Service (AS), bei der anderen bezieht der Client die Tickets vom Ticket-Granting Service (TGS). Beide Dienste sind sich relativ ähnlich, der TGS ist aber ein kerberisierter Dienst. Der Client benötigt seinen Langzeitschlüssel (also in der Regel das User-Passwort), um den Inhalt der Antworten des AS zu entschlüsseln.

Command	Description
Kdestroy	Löscht tickets
Kinit	TGT abrufen und im cache speichern bzw cache erzeugen
Klist	Aktuelle tickets anzeigen (Credential cache anzeigen)
Kpasswd	Kerberos password ändern
Ktutil	Schlüsseltabellendateien verwalten
Kadmin	Datenbank verwalten

Man in the Middle Angriff

Kerberos verhindert Man in the Middle Attacken, in dem es die 3 Parteien Authentifizierung verwendet und somit immer weiss, wer den
Service verwendet.

LDIF: Bei einer LDIF-Datei handelt es sich um ein standardmäßiges Klartext-Datenaustauschformat zur Darstellung
von LDAP-Verzeichnisinhalten und Aktualisierungsanforderungen (LDAP = Lightweight Directory Access Protocol).
LDIF übermittelt Verzeichnisinhalte als Datensatzgruppe mit einem Datensatz für jedes Objekt (oder jeden Eintrag).
Auch Aktualisierungsanforderungen wie z. B. Hinzufügen, Ändern, Löschen und Umbenennen werden als
Datensatzgruppe mit einem Datensatz für jede Aktualisierungsanforderung dargestellt.

Principal:

Ein Principal ist eine eindeutige Identität, der ein Kerberos Ticket zugewiesen werden kann. Principals können eine beliebige Anzahl von
Komponenten haben. Jede Komponente wird durch ein Trennzeichen getrennt, meistens «/». Die letzte Komponente ist der Realm, der vom
rest des Principals durch das Realm-Trennzeichen, in der Regel @ getrennt wird.
Bsp: jennifer@ATHENA.MIT.EDU -> Endung in Grossbuchstaben
- Computer und User Principal