Kerberos – opis systemu i instalacja w OS Linux

Autorzy: Piotr Lasota, Joanna Pliś IVFDS

STRESZCZENIE

W niniejszej pracy został opisany system Kerberos, w szczególności jego zasada działania, sposób uwierzytelniania użytkowników i uzyskiwania dostępu do zdalnych usług.

Opisano także wady i zalety systemu, oraz działanie domen Kerberosa.

Zawarto również szczegółowy opis instalacji i konfiguracji systemu pod OS Linux.

SPIS TREŚCI

1. DZIAŁANIE SYSTEMU KERBEROS

1.1. Założenia i podstawy

Kerberos to system potwierdzania tożsamości użytkowników sieci komputerowej i udostępniania im bezpiecznego dostępu do zdalnych usług.

System pośredniczy w procesie autoryzacji klienta do zdalnego serwera.

Opiera się on na następujących zasadach bezpieczeństwa:

- Nie ma zaufania do komputerów – ufa się natomiast użytkownikom, po potwierdzeniu ich tożsamości i w miarę przysługujących im uprawnień
- Nie ma zaufania do adresów sieciowych użytkowników, bezpieczeństwa transmisji danych, jak również. do bezpieczeństwa systemu operacyjnego użytkowników
- Ufa się bezpieczeństwu serwera Kerberos – w tym znaczeniu, że zakłada się że serwer jest zawsze tym, za kogo się podaje. Dzięki właściwościom protokołu podszycie się pod serwer jest bardzo trudne. Ufa się również bazom danych i bezpieczeństwu fizycznego składowania danych.
- System nie ma możliwości zabezpieczenia przed odkryciem słabego hasła użytkownika. [1]

W serwerze KRB przechowywana jest baza danych kluczy prywatnych, czyli najczęściej haseł zakodowanych w odpowiedni sposób. Bezpieczeństwo klucza jest podstawą działania systemu, musi on zostać przekazany serwerowi w bezpieczny sposób podczas rejestracji użytkownika.

Wstępna wymiana informacji pomiędzy użytkownikiem a Kerberosem opiera się o ten właśnie klucz – w momencie autoryzacji serwer odpowiada klientowi danymi zakodowanymi przy użyciu właśnie tego klucza. Jest to kodowanie symetryczne, więc tylko klient posiadający klucz którym zakodowano informację, może ją rozkodować.

Kerberos zapewnia trzy poziomy bezpieczeństwa. Wykorzystanie konkretnego poziomu jest zakładane przez programistę aplikacji sieciowej, w zależności od potrzeb.

Te poziomy to:

- użytkownik autoryzowany jest jednokrotnie, tylko podczas nawiązywania połączenia ze zdalnym serwerem. Później zakłada się, że wszystkie informacje przychodzące od hosta o podanym adresie sieciowym są autentyczne. Wykorzystuje się ten poziom np. w NFS
- wymagana jest autoryzacja przy przekazywaniu każdego komunikatu, jednak sam komunikat nie jest zakodowany

- najwyższy poziom bezpieczeństwa – wymagana jest autoryzacja przy przekazywaniu każdego komunikatu, komunikat jest zakodowany.

1.2. Części składowe Kerberosa

System składa się z kilku części składowych tworzących moduły. Budowa Kerberosa zmieniała się wraz z jego rozwojem.

a) Kerberos aplication library – biblioteka aplikacji – zawiera funkcje i procedury do interakcji z użytkownikiem, zarówno do programu klienckiego jak i serwera. Są to np. funkcje logowania do systemu, pobierania od użytkownika loginu i hasła.

b) Encryption library – biblioteka kodowania – zawiera funkcje do kodowania i rozkodowywania wiadomości. Kodowanie w Kerberosie opiera się na DES – Data Encryption Standard. Jest to jednakże moduł wymienny, może zostać wymieniony na obsługujący inne rodzaje kodowania.

c) biblioteka bazy danych i programy obsługujące tą bazę - pierwsze wersje Kerberosa – projekt Athena – opierały się na bazie INGRES. Obecnie można wybrać jaką bazą będzie posługiwał się system.

   W bazie danych przechowywane są informacje takie jak nazwa użytkownika, hasło i data jego ważności, oraz szczegółowe dane o tej osobie do celów administracyjnych, w zależności od potrzeb.

d) Administration server – zarządza odczytem/zapisem danych z sieci, oraz dostarcza interfejs sieciowy dla bazy danych. Steruje pracą innych modułów. Odczytuje i zapisuje dane w bazie, dlatego też musi być zainstalowany na komputerze który ma bezpośredni dostęp do niej.

e) Authentication server – serwer autoryzacji – odczytuje i dekoduje informacje wymieniane z klientem. Tworzy klucze sesji na potrzeby połączenia. Dane tylko odczytuje z bazy, nie zapisuje ich, może być więc umieszczony na komputerze posiadającym tylko kopię bazy danych, niekoniecznie na podstawowym serwerze. [2]

1.3. Jak to działa?

W dalszej części pracy zostaną przyjęte następujące oznaczenia:

c -> klient

s -> serwer

addr -> adres sieciowy klienta

life -> czas życia biletu

tgs, **TGS** -> ticket-granting server – serwer przydzielający bilety

Kerberos -> authentication server – serwer autoryzacji

KDBM -> administration server – serwer administracyjny

$\mathbf{K}_{x}$ -> prywatny klucz użytkownika x

$\mathbf{K}_{x,y}$ -> klucz sesji pomiędzy x i y

$\{\mathbf{abc}\}\mathbf{K}_{x}$ -> słowo abc zakodowane przy użyciu klucza K użytkownika x

$\mathbf{T}_{x,y}$ -> bilet użytkownika x do komunikacji z y

WS -> workstation – stacja robocza

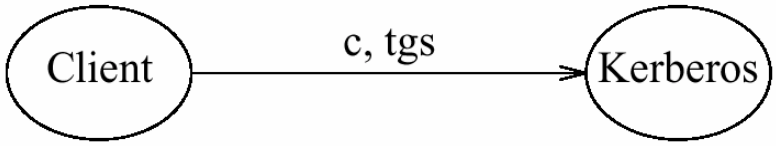
$\mathbf{A}_{x}$ -> authenticator (autoryzator) dla x

Autoryzacja w Kerberosie opiera się na modelu dystrybucji kluczy Needhama i Schroedera. Składa się ona z kilku faz: do serwera Kerberosa wysyłane jest żądanie o autoryzację. Serwer odpowiada na to żądanie udzielając klientowi „biletu na przyznanie biletu". Następnie klient ponownie komunikuje się z Kerberosem, informując go jaką usługę chce uzyskać, i na jakiej zdalnej maszynie. Kerberos odpowiada udzielając mu „listu polecającego" do tej zdalnej maszyny.

1.3.1. Uzyskiwanie początkowego biletu.

W protokole systemu Kerberos nie ma przesyłu haseł – zakłada się że podczas rejestracji użytkownika hasło zostało dostarczone do serwera w sposób bezpieczny, i więcej już nie jest przesyłane.

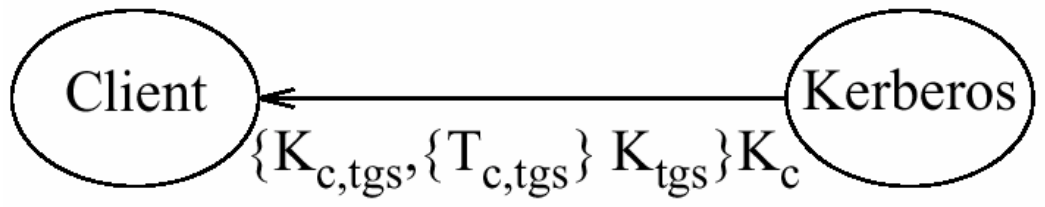
Gdy użytkownik na zdalnym komputerze zechce połączyć się z innym komputerem używając Kerberosa, włącza program kliencki systemu. Program ten pyta go o zarejestrowaną nazwę w systemie. Na tej podstawie tworzona jest wiadomość zawierająca podaną nazwę użytkownika, oraz nazwę serwera Kerberos. Jest ona wysyłana do serwera autoryzacji.



Rys. Żądanie autoryzacji

Serwer autoryzacji sprawdza, czy informacje które otrzymał są poprawne – czy zgadza się nazwa serwera, przeszukuje bazę danych w celu znalezienia informacji o użytkowniku posługującym się podanym loginem. Jeśli wpis został znaleziony – generowany jest losowy klucz sesji, który później posłuży do komunikacji klienta z usługą TGS (Ticket Granting Serwer) Kerberosa.

Następnie jest tworzony bilet zawierający wygenerowany klucz sesji, nazwę użytkownika, nazwę serwera, obecny czas oraz czas życia biletu, oraz adres IP z którego przyszło żądanie o autoryzację. Bilet ten jest kodowany przy użyciu klucza znanego tylko jemu i TGS, i jest przesyłany do usługi TGS w celu poinformowania jej co ma zrobić jeśli otrzyma informacje stworzone przy użyciu tego biletu. Następnie kodowany przy użyciu pobranego z bazy danych hasła klienta i odsyłany do niego.



Rys. Odsyłanie wygenerowanego biletu do klienta

W momencie otrzymania biletu program protokołu Kerberos na komputerze klienckim monituje go o podanie hasła. Zostaje podjęta próba zdekodowania otrzymanej informacji przy użyciu tego hasła. Jeśli powiedzie się ona – zdekodowany klucz sesji przechowywany jest w pamięci, natomiast w celach bezpieczeństwa zamazywane jest podane hasło użytkownika.

Stosowane jest tutaj kodowanie synchroniczne algorytmem DES, co zapewnia, że jeśli informacja została zakodowana jakimś kluczem, można ją odszyfrować tylko przy użyciu tego samego klucza. Dzięki temu, że nawet jakby przesył danych między serwerem Kerberos a klientem był podglądany, haker nie będzie mógł odczytać komunikatów nie znając hasła klienta.

W tym momencie klient dysponuje kluczem sesji potwierdzającym jego tożsamość. [2]

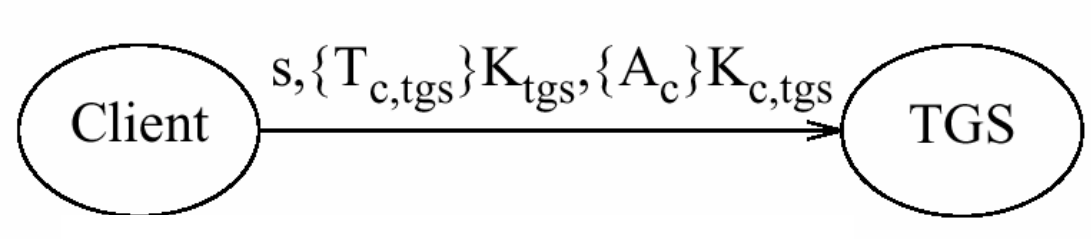
1.3.2. Uzyskiwanie biletu na dostęp do zdalnej usługi.

Aby uzyskać dostęp do usługi na zdalnej maszynie, klient musi najpierw otrzymać od Kerberosa bilet, przy pomocy którego ma się komunikować z tym serwerem.

W tym celu tworzona jest wiadomość, zawierająca:

- identyfikator żądanej usługi i nazwę serwera na którym ta usługa jest aktywna
- otrzymany wcześniej klucz sesji, zakodowany kluczem znanym tylko serwerowi autoryzacji Kerberos i TGS

- autoryzator - jednorazowy bilet zawierający nazwę i adres klienta, oraz timestamp - aktualny czas systemowy klienta. autoryzator jest generowany na komputerze klienta, a następnie szyfrowany przy użyciu otrzymanego wcześniej klucza sesji.



Rys. Żądanie biletu na dostęp do serwera

Po otrzymaniu takiej informacji, TGC (Ticket Granting Server) sprawdza jej poprawność:

- rozkodowuje autoryzator przy użyciu klucza sesji wspólnego dla klienta i serwera Kerberos
- sprawdza poprawność autoryzatora – czy nie został już użyty (jest jednokrotnego użytku) i czy nie upłynął czas jego ważności
- sprawdza ważność klucza sesji

Jeśli wszystkie informacje są poprawne, generowany jest kolejny klucz losowy – posłuży on do komunikacji klienta z serwerem do którego żądał dostępu.

Budowany jest nowy bilet zawierający:

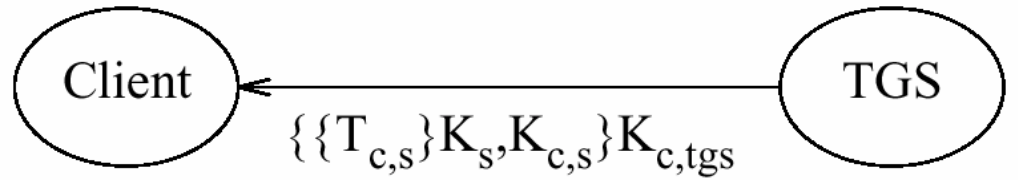
- Bilet do serwera składający się z:

  - nazwy klienta
  - nazwę serwera
  - obecny czas
  - czas wygaśnięcia ważności biletu
  - adres IP klienta
  - klucz sesji właśnie wygenerowany

  Bilet ten jest kodowany przy użyciu klucza znanego tylko serwerowi Kerberos i serwerowi do którego usługa jest żądana.

- Wygenerowany klucz sesji.

Wszystkie te informacje są kodowane przy użyciu klucza sesji pomiędzy Kerberosem i klientem, i odsyłane do klienta.



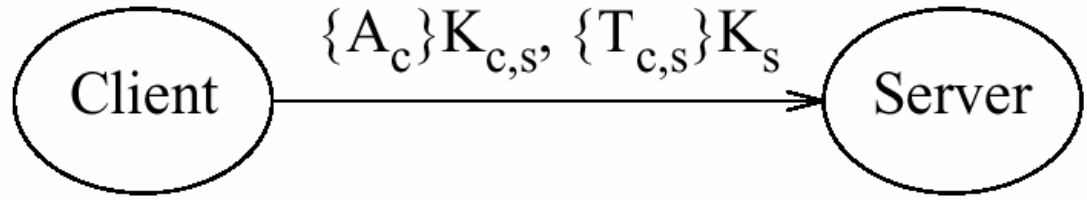
Rys. Informacja zawierająca bilet na dostęp do zdalnego serwera

Po otrzymaniu informacji aplikacja Kerberosa uruchomiona na komputerze klienta dekoduje ją, i dysponuje w tym momencie biletem na żądanie dostępu do usługi do zdalnego serwera, oraz kluczem sesji z tym serwerem, może więc rozpocząć komunikację z nim. [2],[3]

1.3.3. Autoryzacja i uzyskanie usługi na zdalnym serwerze.

W celu uzyskania usługi na zdalnym komputerze, aplikacja Kerberosa w komputerze klienta generuje informację zawierającą:

- Autoryzator (nazwa klienta, jego adres IP oraz aktualny czas systemowy), zakodowany przy użyciu klucza sesji.
- Otrzymany od TGS bilet na dostęp do serwera



Rys. Żądanie dostępu do usługi na zdalnym serwerze

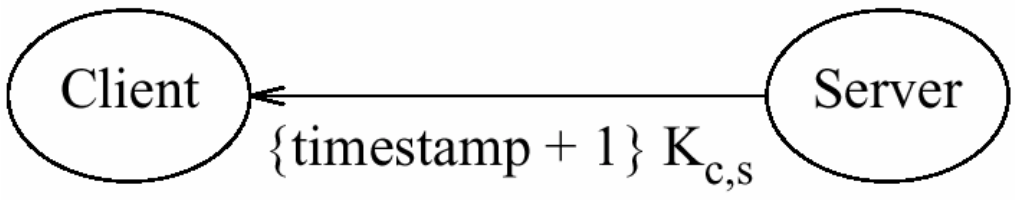
Zdalny serwer po otrzymaniu takiej informacji:

- dekoduje bilet przy użyciu klucza K_S, znanego tylko jemu serwerowi Kerberos
- sprawdza czy informacja zawarta w tym bilecie jest poprawna, czyli czy nie wygasł czas jego ważności, zgadzają się nazwy klienta i serwera oraz adres IP klienta
- przy użyciu klucza sesji zawartego w zdekodowanym bilecie, deszyfruje autoryzator klienta, oraz sprawdza jego poprawność.

Klient może zażądać potwierdzenia autentyczności serwera, czyli dowodu, że komputer z którym nawiązał połączenie jest na pewno tym z którym chciał to połączenie nawiązać.

Nosi to nazwę „mutual authentication” – identyfikacji wzajemnej.

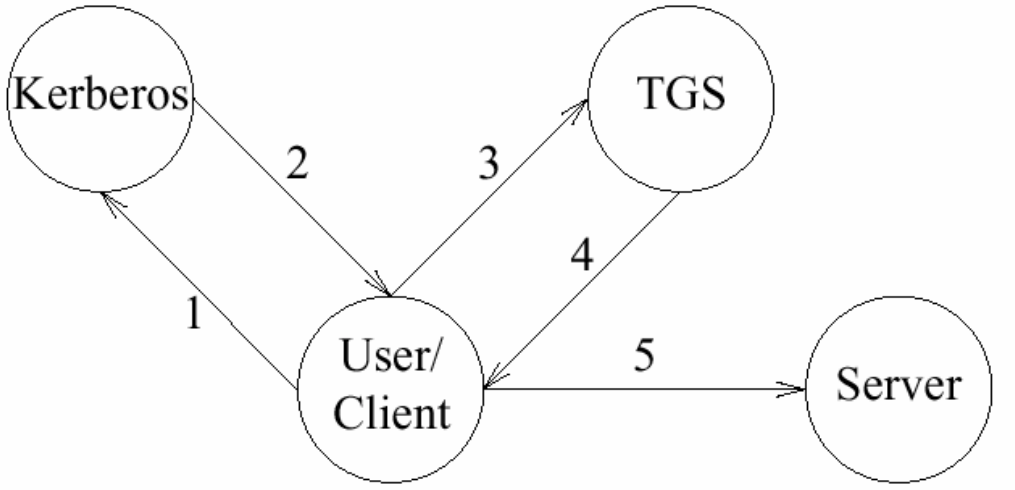
W takim przypadku serwer generuje informację zawierającą odcisk czasu zawarty w autoryzatorze klienta i zwiększony o 1, szyfruje tą informację przy użyciu klucza sesji, i odsyła do klienta.



Rys. Identyfikacja wzajemna – potwierdzenie tożsamości serwera

Po zakończeniu tej wymiany, opierając się na zasadzie zaufania systemowi Kerberos, zarówno serwer udzielający usługi, jak i klient żądający jej mają pewność, że druga strona jest tym, za kogo się podaje.

Uzyskanie dostępu do zdalnego serwera za pośrednictwem Kerberosa, można zilustrować następującym schematem:



Rys. Schemat dostępu do usługi przy pośrednictwie systemu Kerberos

1. Żądanie biletu na uzyskanie biletu (ticket granting ticket) do usługi TGS
2. Otrzymanie biletu na uzyskanie biletu
3. Żądanie biletu na usługę
4. Otrzymanie biletu na usługę
5. Uzyskanie dostępu do serwera za pomocą biletu na usługę. [2],[3]

2. UŻYWANIE SYSTEMU KERBEROS

Z punktu widzenia użytkownika, nie ma znacznej różnicy pomiędzy stosowaniem Kerberosa i zwykłych protokołów niezabezpieczonych. Po załogowaniu się do komputera użytkownik jest proszony o podanie loginu na serwer Kerberos. Informacja zawierająca ten login jest przesyłana do serwera, i w momencie otrzymania odpowiedzi użytkownik musi podać hasło, które posłuży do zdekodowania otrzymanej informacji.

Jest to jedyny moment, kiedy użytkownik musi się uwierzytelnić, całą obsługą protokołu Kerberos zajmują się działające w tle programy, czego użytkownik może w ogóle nie zauważać.

Różnicę stanowi tutaj czas ważności uwierzytelnienia – standardowo wynosi on osiem godzin – po upływie tego czasu wszystkie przepustki i bilety które zostały przyznane użytkownikowi przestaję działać, i musi on ponownie autoryzować się w systemie, co spowoduje że serwer Kerberos ponownie wystawi przepustkę ważną na osiem godzin.

W systemach Linux wszystkie przepustki są przechowywane w katalogu /tmp, i są automatycznie niszczone po wylogowaniu się użytkownika.

Pewnym problemem jest to, że w momencie autoryzacji użytkownika system Kerberos ufa wszystkim informacjom pochodzącym z tego hosta. Może więc zajść sytuacja, że gdy na stacji roboczej będzie zalogowanych dwóch użytkowników, i jeden z nich podda się weryfikacji w systemie Kerberos, drugi łamiąc zabezpieczenia systemu operacyjnego wykradnie jego przepustki i będzie mógł się pod niego podszyć.

3. SIEĆ KERBEROS

Obszar działania systemu Kerberos został podzielony na tzw. domeny – realms. Każda dziedzina posiada swój serwer Kerberos, a także swoje, ustalone w zależności od potrzeb, zasady bezpieczeństwa. Są one tym wyższe, im ważniejsze dane są chronione poprzez system. Należy jednak pamiętać, że im bardziej zaawansowana jest ochrona systemu, tym bardziej jest ona też zasobożerna.

Domeny tworzą strukturę hierarchiczną – istnieją w niej domeny potomne, oraz macierzyste, które mogą sobie w określony sposób ufać, i wymieniać dane zarejestrowanych użytkowników. Na przykład – firma tworzy domenę macierzystą, a jej lokalne oddziały – domeny potomne. Zasady zaufania mogą być określone w taki sposób, że użytkownik, który jest zarejestrowany i autoryzowany w domenie macierzystej ma automatycznie dostęp do domen potomnych, ale odwrotnie – niekoniecznie.

Poszczególne domeny Kerberos mogą nawzajem honorować swoje decyzje co do autentyczności użytkowników. Jeśli użytkownik zarejestruje się w swojej domenie Kerberos, i jest ona akceptowana jako wiarygodna w innej domenie, może na podstawie biletu wydanego przez swój TGS uzyskać dostęp do domeny odległej [1], [2].

Dla w pełni poprawnej pracy systemu Kerberos, muszą być spełnione następujące warunki:

- W bazie danych serwera przechowywane są dane o wszystkich użytkownikach, którzy mają prawo korzystać z systemu, razem z informacjami z jakich usług mają oni korzystać, i ich hasłami
- Z każdym serwerem, który korzysta z autoryzacji Kerberosa musi zostać wymieniony i zachowany bezpieczny klucz, który będzie służył do wzajemnej komunikacji
- Z każdą zdalną domeną która jest uwiarygodniona, musi zostać wymieniony i zachowany bezpieczny klucz..

Uzyskanie dostępu do usługi poprzez serwer Kerberosa z odległej domeny jest wielostopniowe. Można to opisać w następujący sposób:

1) Klient komunikuje się ze swoim serwerem Kerberos, żądając „biletu na udostępnienie biletu” – dostępu do TGS (punkt 1.3.1 niniejszej pracy)
2) Serwer wysyła „bilet na uzyskanie biletu” do klienta (punkt 1.3.1)
3) Klient komunikuje się z TGS przy pomocy klucza otrzymanego w powyższym kroku, i podaje do jakiej domeny zdalnej chce mieć dostęp (podobnie jak w sposób opisany punkcie punkt 1.3.2 otrzymywał dostęp do usługi)
4) Serwer TGS odsyła klientowi bilet na dostęp do zdalnej domeny (informacje o kliencie i czasie wygenerowania biletu zakodowane kluczem znanym tylko lo-

kalnemu i zdalnemu serwerowi Kerberos), oraz klucz sesji klienta ze zdalną usługą autoryzacji

5) Klient uwierzytelnia się w zdalnym systemie Kerberos przy użyciu otrzymanych informacji
6) Po poprawnej weryfikacji danych odległy system odpowiada klientowi tak, jakby był on klientem jego własnej domeny, udzielając mu biletu na usługę (punkt 1.3.3)
7) Klient przy użyciu tego biletu uzyskuje dostęp do zdalnej usługi (punkt 1.3.3). [4]

4. WADY SYSTEMU

- Kerberos nie zapewnia w jakikolwiek sposób bezpieczeństwa hosta, a więc z zaufanymi hostami komunikuje się w sposób nie wymagający uwierzytelnień. Jeśli więc bezpieczeństwo hosta zostanie skompromitowane, również Kerberos jest skompromitowany. Zależy to oczywiście jeszcze od tego, do jakiego hosta nastąpi włamanie. Jeśli włamywać ukradnie jedynie bilety, może on podszywać się pod osoby tylko do pewnego czasu, aż ważność biletów wygaśnie. Jeżeli natomiast włamanie będzie do TGS, to skompromitowany zostanie cały realm, gdyż w TGS przechowywane są hasła służące do kodowania.

- W Kerberosie 4, identyfikatory są ważne przez 5 minut. Jeśli więc ktoś przeszukuje sieć w poszukiwaniu identyfikatorów, ma okno 5 minutowe, w którym może powtórnie użyć go i dostać dostęp do tej samej usługi. Kerberos 5 został pod tym względem poprawiony. Została zaimplementowana ‘replay cache’, która zapobiega użyciu po raz kolejny tego samego identyfikatora.

- Każdy może także wysłać prośbę o bilet podszywając się pod inną osobę. Otrzymuje zakodowany bilet hasłem danej osoby i może próbować w ten sposób otrzymać hasło rozkodowując bilet. Kerberos 5 rozwiązał jednak i ten problem stosując preweryfikację.

- Hosty także mogą używać protokołów do synchronizacji czasu, które nie wymagają uwierzytelnienia. Takie protokoły nie są odporne na ataki. Haker bardzo prosto może zmienić czas hostu, przez co może użyć stare identyfikatory.

- Ponadto w stacjach roboczych, intruz łatwo może podmienić program login, na własną wersję, która przed użyciem w Kerberosie będzie zapisywała gdzieś dane hasło. Łamie więc to zasadę, że hasła nie są nigdzie zapisane jako czysty tekst. Rozwiązaniem tego problemu mogłoby być używanie jednorazowych haseł, jednakże Kerberos nie wspiera takiego postępowania.

5. INSTALACJA I KONFIGURACJA KERBEROSA W SYSTEMIE LINUX

Definicje użyte w dalszej części rozdziału:

klient
jednostka, która może otrzymać bilet. Jest to zazwyczaj albo użytkownik albo host.

host
komputer, który może być dostępny przez sieć.

KDC
Key Distribution Center – centrum dystrybucji kluczy.

keytab
a plik, który zawiera klucz, lub klucze. Host albo usługa korzysta z keytab w podobny sposób jak użytkownik korzysta z hasła.

principal
string, nazywający daną jednostkę, która może otrzymać uwierzytelnienie. Składa się z trzech częsci:

primary
pierwsza część, w przypadku użytkownika jest to nazwa danego użytkownika. W przypadku zaś usługi, jest to nazwa usługi.

instance
druga część, daje informacje, które określają primary. Instance może być null. W przypadku hosta, jest to pełna nazwa danego hosta.

realm
sieć logiczna, obsługiwana przez bazę danych Kerberosa i ustalona przez KDC. Przyjęto stosowanie wielkich liter w nazwach realmów, aby odróżnić je od domen internetowych.

Typowy format Kerberos'owego principala to: primary/instance@REALM.

usługa
każdy program czy też komputer, który może być dostępny przez sieć. Przykłady usług zawierają „host" (np. kiedy używasz telnet i rsh), „ftp", „krbtgt" i „pop" (e-mail).

bilet
tymczasowo ustalone elektroniczne uwierzytelnienie, które weryfikuje tożsamość klienta dla danej usługi.

TGT
Ticket-Granting Ticket. Specjalny bilet Kerberosa, który pozwala klientowi na otrzymanie dodatkowych biletów Kerberosa w daneym realmie.

5.1. Instalacja serwera Kerberos

5.1.1. Rozpakowanie i zainstalowanie binariów.

Załóżmy, że rozpakowaliśmy pliki do katalogu: '`/krb/krb5-1.2'`.
Oczywiście wszystkie te operacje wykonujemy jako root.

Wybieramy najprostszą opcję instalacji w jednym katalogu, a więc piszemy:

1. `cd /krb/krb5-1.2/src`
2. `./configure`
3. `make`

Sprawdzamy poprawność instalacji przez:

```
% make check
```

Dodatkowo możesz wybrać różne opcje instalacji. Na przykład:

`--help`
: Wyświetla pomoc i przykłady najczęściej używanych opcji przy instalacji Kerberosa

`--prefix=PREFIX`
: Domyślnie Kerberos instaluje pliki pakietu w '/usr/local'. Użyj tą opcję, jeśli chcesz zmienić tą lokalizację.

`--exec-prefix=EXECPREFIX`
: Ta opcja pozwala na oddzielenie architektury niezależnych programów od plików konfiguracyjnych i manualów.

Plik konfiguracyjny, który można sprawdzić po instalacji, to `` `include/krb5/stock/osconf.h' `` Zawiera on poniższe zmienne:

`DEFAULT_PROFILE_PATH`
: Ścieżka do pliku, który zawiera ustawienia do znanych realmów, ich KDC.

`DEFAULT_KEYTAB_NAME`
: Typ i ścieżka do domyślnego pliku serwera keytab.

`DEFAULT_KDC_ENCTYPE`
: Domyślny typ kodowania dla KDC.

`KDCRCACHE`
: Nazwa cachu używanego do powtórzeń przez KDC.

`RCTMPDIR`
: Katalog, w którym znajduje się cache powtórzeń.

`DEFAULT_KDB_FILE`
: Lokalizacja domyślnej bazy danych.

[5]

5.1.2. Edycja plików konfiguracyjnych

Pierwszym plikiem jest **krb5.conf**. Znajduje się on w katalogu: `/etc/krb5.conf`. Zawiera opcje konfiguracji dla klientów takie jak lokalizacja wszystkich KDC, który KDC jest lokalny, a także miejsce plików loggingu itd.

Przykład skonfigurowanego pliku:

```
[libdefaults]
    ticket_lifetime = 600
    default_realm = JOANKA.PRZ.RZESZOW.PL
    default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
    default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc

[realms]
    JOANKA.PRZ.RZESZOW.PL = {
        kdc = kerberos.prz.rzeszow.pl:88
        kdc = kerberos-1.prz.rzeszow.pl:88
        kdc = kerberos-2.prz.rzeszow.pl:88
        admin_server = kerberos.prz.rzeszow.pl:749
        default_domain = prz.rzeszow.pl
    }

[domain_realm]
    .prz.rzeszow.pl = JOANKA.PRZ.RZESZOW.PL
    prz.rzeszow.pl = JOANKA.PRZ.RZESZOW.PL

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

Sprawdzamy również poprawność pliku **kdc.conf**. Znajduje się on w katalogu `/usr/local/var/krb5kdc/kdc.conf`

Przykład pliku:

```
[kdcdefaults]
    kdc_ports = 88,750

[realms]
    JOANKA.PRZ.RZESZOW.PL = {
        database_name = /usr/local/var/krb5kdc/principal
        admin_keytab = /usr/local/var/krb5kdc/kadm5.keytab
        acl_file = /usr/local/var/krb5kdc/kadm5.acl
        dict_file = /usr/local/var/krb5kdc/kadm5.dict
        key_stash_file = /usr/local/var/krb5kdc/.k5.JOANKA.PRZ.RZESZOW.PL
        kadmind_port = 749
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = des3-hmac-sha1
        supported_enctypes = des3-hmac-sha1:normal des-cbc-crc:normal
    }
```

Edytujemy teraz plik / **.k5login.** Każda linia zawiera nazwę zasady, która wskazuje prawa do ksu (kerberowska wersja su) dla roota. To po prostu prosta lista kontroli dostępu.

Plik **/etc/services** zawiera numery portów, które są skojarzone z odpowiednimi usługami. Domyślnie kerberos zajmuje porty 88 dla KDC, a 749 dla serwera administratora. W tym miejscu konfigurujemy po prostu firewall, aby pracował z Kerberosem. Poniższe linie przedstawiają domyślne ustawienia dla portów w tym pliku:

```
ftp             21/tcp          # Kerberos ftp and telnet use the
telnet          23/tcp          # default ports
```

```
kerberos      88/udp    kdc    # Kerberos V5 KDC
kerberos      88/tcp    kdc    # Kerberos V5 KDC
klogin        543/tcp          # Kerberos authenticated rlogin
kshell        544/tcp   cmd    # and remote shell
kerberos-adm  749/tcp          # Kerberos 5 admin/changepw
kerberos-adm  749/udp          # Kerberos 5 admin/changepw
krb5_prop     754/tcp          # Kerberos slave propagation
eklogin       2105/tcp         # Kerberos auth. & encrypted rlogin
krb524        4444/tcp         # Kerberos 5 to 4 ticket translator
```

Aby upewnić się, że w czasie bootowania daemony krb5kdc i kadmin zostaną uruchmione można dodać linie w pliku **/etc/rc.<hostname>.**
[5],[6]

5.1.3. Stworzenie bazy danych.

Do utworzenia bazy danych i ewentualnego pliku stash użyjemy komendy kdb5_util. Plik stash jest lokalną kopia master key, który umiejscowiony jest w zakodowanej formie na lokalnym dysku KDC. Plik ten używany jest do samoautentyzacji KDC automatycznie przed wystartowaniem daemonów kadmin i krb5kdc. Plik stash, tak samo jak plik keytab jest słabym punktem, który może posłużyć włamaniu. Przejęcie go pozwoli na nieograniczony dostęp do bazy danych Kerberosa. Powinien więc on istnieć wyłącznie na dysku lokalnym KDC i być do odczytu jedynie przez roota.

Polecenie **kdb5_util** poprosi o wpisanie master key dla bazy danych Kerberosa. Może to być dowolny string, lecz należy go dobrać jak każde hasło, które nie jest łatwe w odgadnięciu.

Poniżej przykład tworzenia bazy danych:

```
shell% /usr/local/sbin/kdb5_util create -r JOANKA.PRZ.RZESZOW.PL -s
Initializing database '/usr/local/var/krb5kdc/principal' for realm
JOANKA.PRZ.RZESZOW.PL',
master key name 'K/M@JOANKA.PRZ.RZESZOW.PL'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:  @doubleleftarrow{ Type the master password.}
Re-enter KDC database master key to verify:  @doubleleftarrow{ Type it again.}
shell%
```

Tworzy to pięć plików w katalogu, który został określony w kdc.conf: dwa pliki bazy danych Kerberosa (principal.db, principal.ok.), plik administracyjny bazy (principal.kadm5), plik administracyjny bezpieczeństwa (principal.kadm5.lock) i plik stash (.k5stash). Katalogiem domyślnym jest /usr/local/var/krb5kdc. Gdy nie chcemy pliku stash, uruchamiamy powyższe polecenie bez opcji –s.

5.1.4. Dodanie administratorów do pliku acl i do bazy danych

Następną sprawą jest utworzenie pliku ACL (Access Control List). Nazwa pliku powinna korespondować z podaną w pliku kdc.conf. Domyślna to **kadm5.acl**. Format pliku jest następujący:
principal Kerberosa pozwolenia principal (opcjonalny)

Principal Kerberosa (oraz opcjonalny principal) mogą zawierać ”*”. Na przykład jeśli chcemy dać nieograniczone pozwolenie wszystkim uprawnionym, którzy mają w instance admin, użyjemy składni: „*/admin@REALM” gdzie „REALM” jest nazwą realma Kerberosa.

Możliwe są następujące opcje pozwoleń:

a
pozwala na dodawanie osoby uprawnionej albo zasady polityki w bazie danych.

A
zabrania na dodawanie osoby uprawnionej albo zasady polityki bazy danych.

d
pozwala na wykasowanie osoby uprawnionej albo zasady polityki z bazy danych.

D
zabrania kasowanie osób uprawnionych lub zasad polityki.

m
pozwala na modyfikacje osób lub zasad bazy danych.

M
zabrania modyfikacji osób lub zasad.

c
pozwala na zmianę hasła dla osób w bazie danych.

C
zabrania zmiany hasła osobom w bazie danych.

i
pozwala na zapytania do bazy danych.

I
zabrania zapytań do bazy.

l
pozwala wyświetlić listę osób albo zasad bazy.

L
zabrania wyświetlania listy osób albo zasad.

*****
oznacza wszystkie przywileje (admcil).

x
oznacza wszystkie przywileje (admcil); identyczne jak "*".

Na przykład, żeby dać uprawnionym */admin@JOANKA.PRZ.RZESZOW.PL wszelkie uprawnienia w bazie danych piszemy:

```
*/admin@ATHENA.MIT.EDU  *
```

Natomiast, aby osobie pitadmin@JOANKA.PRZ.RZESZOW.PL dać pozwolenie na dodawanie, wyświetlanie listy i zapytania wszystkich uprawnionych, których instance to root, piszemy:

```
pitadmin@JOANKA.PRZ.RZESZOW.PL  ali  */root@JOANKA.PRZ.RZESZOW.PL
```

Następnym krokiem jest dodanie uprawionych osób do bazy danych Kerberosa. (Przy instalacji wymagana jest przynajmniej jedna osoba.) Używamy tym razem polecenia **kadmin.local** . Administratorzy, których chcemy utworzyć w bazie danych, powinni być dodani do pliku ACL.

W poniższym przykładzie dodawany jest administrator admin/admin:

```
shell% /usr/local/sbin/kadmin.local
kadmin.local: addprinc admin/admin@JOANKA.PRZ.RZESZOW.PL
WARNING: no policy specified for "admin/admin@JOANKA.PRZ.RZESZOW.PL";
defaulting to no policy.
Enter password for principal admin/admin@JOANKA.PRZ.RZESZOW.PL:
@doubleleftarrow{ Enter a password.}
Re-enter password for principal admin/admin@JOANKA.PRZ.RZESZOW.PL:
@doubleleftarrow{ Type it again.}
Principal "admin/admin@JOANKA.PRZ.RZESZOW.PL" created.
kadmin.local:
```

Poźniejsze dodawanie administratorów bazy danych wymaga jedynie polecenia **kadmin**.
[5],[6]

5.1.5. Utworzenie kadmind keytab

Kadmind keytab jest kluczem, który kadmind będzie używał do rozkodowania biletów Kerberosa dla administratorów, żeby określić czy dać czy nie dać im dostęp do bazy danych. Trzeba utworzyć kadmin keytab z wejściami dla kadmin/admin i kadmin/changepw, którzy są tworzeni automatycznie w czasie tworzenia bazy danych. Aby stworzyć kadmin keytab, uruchamiamy kadmin.local i używamy komendy ktadd.

Przykład:

```
shell% /usr/local/sbin/kadmin.local
kadmin.local: ktadd -k /usr/local/var/krb5kdc/kadm5.keytab
=> kadmin/admin kadmin/changepw
Entry for principal kadmin/admin@JOANKA.PRZ.RZESZOW.PL with
     kvno 3, encryption type DES-CBC-CRC added to keytab
     WRFILE:/usr/local/var/krb5kdc/kadm5.keytab.
Entry for principal kadmin/changepw@JOANKA.PRZ.RZESZOW.PL with
     kvno 3, encryption type DES-CBC-CRC added to keytab
     WRFILE:/usr/local/var/krb5kdc/kadm5.keytab.
kadmin.local: quit
shell%
```

Argument ‘-k’ powoduje, że ktadd zachowa wydzieloną keytab jako /usr/local/var/krb5kdc/kadm5.keytab. Nazwa pliku musi się zgadzać z wpisaną do pliku kdc.conf.
[5]

5.1.6. Uruchomienie deamonów kerberosa

W tej chwili można już wystartować deamony Kerberosa.

```
shell% /usr/local/sbin/krb5kdc
shell% /usr/local/sbin/kadmind
```

Każdy daemon będzie biegł w tle.

Można sprawdzić czy deamony wystartowały poprawnie przez sprawdzenie pliku loggującego (zdefiniowanego w krb5.conf). Na przykład:

```
shell% tail /var/log/krb5kdc.log
Dec 02 12:35:47 beeblebrox krb5kdc[3187](info): commencing operation
shell% tail /var/log/kadmin.log
```

```
Dec 02 12:35:52 beeblebrox kadmind[3189](info): starting
```

Wszelkie błędy podczas uruchamiania deamonów byłyby również tutaj wypisane.

Kerberowanie aplikacji.

Najtrudniejszą częścią jest używanie Kerberosa w aplikacjach, które piszemy. Muszą one bowiem zawierać:

1. Identyfikacja użytkownika
2. Zlokalizowanie jego uwierzytelniającego cachu.
3. Sprawdzenie, czy ma on bilet na usługę.
4. Jeśli nie, używając TGT i klucza sesji wysłać prośbę o otrzymanie

5.2. Administrowanie Kerberosem

5.2.1. Baza danych

Używane są głownie dwa programy, znane nam wcześniej z instalacji: **krb5_util** i **kadmin**. Pierwszy z nich służy generalnie do obsługi bazy danych jako całości. Drugi zaś modyfikuje jednostki bazy.

Kadmin zajmuje się głownie prinicipalami, zasadami polityki KADM5 i keytabs. Jest on można by powiedzieć w dwóch wersjach: lokalnej jako kadmin.local i zdalnej jako kadmin/admin. Różnią się jedynie wymogiem autoryzacji w przypadku zdalnej sesji.

Opcje uruchamiania kadmina:

-r *REALM*
: Używa REALM jako domyślnego realm'a dla bazy danych Kerberosa.

-p *principal*
: Używa principala *principal* do identyfikacji w Kerberosie. Bez tej opcji kadmin dołączy „admin" do pierwszej części nazwy principala, środowiskową zmienną USER albo nazwę użytkownika uzyskaną przez getpwuid, w kolejności zależnej od preferencji.

-k *keytab*
: Używa keytab *keytab* do odkodowania odpowiedzi KDC zamiast pytania o hasło. W tym przypadku principal będzie 'host/hostname'.

-c *credentials cache*
: Używa *credentials_cache* jako cache uwierzytelnień. Powinna ona zawierać bilet dla usługi kadmin/admin, który może być uzyskany przez kinit. Bez tej opcji kadmin prosi o nowy bilet z KDC i przechowuje go we własnej tymczasowej cache.

-w *password*
: Używa *password* jako hasło zamiast pytania się o nie później. Nie powinno się umieszczać hasło w skryptach, gdyż może być łatwo przejęte.

-q *query*
: Kieruje zapytanie prosto do kadmin, użyteczne w pisaniu skryptów.

-e *"enctypes ..."*
: **(tylko dla `kadmin.local`)** Ustawia listę kryptosystemu i używa danych typów do tworzenia nowych kluczy. Możliwe typy: `` `des3-cbc-sha1:normal' ``, `` `des-cbc-crc:normal' ``, i `` `des-cbc-crc:v4' ``.

Polecenia kadmina:

get_principal *principal* - (alias: getprinc) otrzymujemy listę atybutów związanych z principalem

list_principals [*expression*] - (alias: listprinc) otrzymujemy listę principalów, w expression możemy używać * ? [], zostaną wyświetlone wszystkie nazwy pasujące do podanego expression

add_principal [*options*] *principal* – (alias: addprinc) dodawanie nowego principala z odpowiednimi opcjami, niektóre z nich:
- expire *date*
- pwexpire *date*
- maxlife *maxlife*
- policy *policy*

-/+ needchange
-/+ allow_postdated
-/+ allow_forwardable
-/+ requires_preauth

modify_principal [*options*] *principal* – modyfikuje przywileje danego principala, opcje takie same jak przy add_principal

delete_principal [**-force**] *principal* – (alias: delprinc) kasuje użytkownika, przy –force nie wymaga potwierdzenia

change_password [*options*] *principal* – (alias: cpw) zmienia hasło principala, niektóre opcje:
- salt *salttype*
- randkey

get_policy *policy* - (alias: getpol) otrzymujemy charakterystykę polityki *policy*

list_policies [*expression*] - (alias: listpols) otrzymujemy listę polityk, analogicznie jak principalów

add_policy [options] policy_name – (alias: addpol) dodaje nową politykę, opcje:
- **maxlife *time***
- **-minlife *time***
- **-minlength *length***
- **-minclasses *number***
- **-history *number***

modify_policy [options] policy_name – (alias: modpol) modyfikuje daną politykę, opcje j.w.

delete_policy *policy_name* - kasuje daną politykę

Aby zrzucić bazę danych do pliku używamy polecenia:

kdb5_util dump [**-old**] [**-b6**] [**-b7**] [**-ov**] [**-verbose**] [*filename* [*principals...*]]

Opcje polecenia:

-old
format zrzutu w wersji starszej np. Kerberos 5 Beta 5 i wcześniejszych.

-b6
format zrzutu Kerberosa 5 Beta 6

-b7
format zrzutu Kerberosa 5 Beta 7

-ov
zrzut w formacie ovsec_adm_export

-verbose

nazwa każdego principala i polityki wypisana jeśli już przerzucona do pliku

Odzyskiwanie bazy danych z pliku za pomocą polecenia:

kdb5_util load [**-old**] [**-b6**] [**-b7**] [**-ov**] [**-verbose**][**-update**] *dumpfilename dbname* [*admin_dbname*]

Opcje polecenia:

-old
plik w formacie Kerberos 5 Beta 5 i starszej

-b6
analogicznie j.w.

-b7
analogicznie j.w.

-ov
analogicznie j.w.

-verbose
analogicznie j.w.

-update
rekordy z pliku uaktualniane lub dodawane do istniejącej bazy

[5], [6]

5.2.2. Keytabs

Dodawanie principala do keytabs odbywa się za pomocą polecenia kadmin:

ktadd [**-k** *keytab*] [**-q**] [*principal* | **-glob** *princ_exp*] [*...*]

Komenda ma następujące opcje:

-k keytab
używa *keytab* jako pliku keytaba. W przeciwnym razie używa domyślnego pliku (`/etc/krb5.keytab`).

-q
uruchamia w cichej wersji, wyświetla mniej informacji.

principal* | -glob *principal expression
dodaje *principal*, albo wszytkich principalów, którzy pasują do pr*incipal expression* do keytab. (*principal expression* patrz kadmin list_principals)

Usuwanie z keytabu:

ktremove [**-k** *keytab*] [**-q**] *principal* [*kvno* | **all** | **old**]

Opcje:

-k keytab
używa *keytab* jako pliku keytab. W przeciwnym razie usunie domyślny plik (`/etc/krb5.keytab`).

-q
j.w.

principal
nazwa principala do usunięcia (wymagane!)

kvno
usuwa wszystkie wejścia dla danego principala, które pasują do *kvno*.

all
usuwa wszystkie wejścia dla danego principala

old
usuwa wszystkie wejścia dla danego principala z najwyższymi kvno.

[5]

5.3. Kerberos dla użytkowników Linuxa

Pierwszą rzeczą, którą trzeba ustalić, aby używać Kerberosa, to principal (nazwa uprawnienia, coś jak stałe konto). Zazwyczaj wygląda to jak: nazwa_własna@JAKIŚ.REALM, np. pit@JOANKA.PRZ.RZESZOW.PL . Tak jak w normalnym koncie, przed małpą jest nazwa, która sami wybraliśmy, zaś po małpie nazwa naszego realma. Tu jednak kończą się podobieństwa z normalnym kontem.

W bazie danych Kerberosa znajdują się informacje o każdym principalu, takie jak: nazwa, hasło itd. Są one zakodowane master key i nie mogą być odczytywane przez wszystkich.

Dla użytkownika Kerberos jest prawie niewidoczny, ale jest parę usług, które wymagają autoryzacji przez Kerberosa. Na przykład `rlogin`. Aby użyć tego polecenia potrzebujemy najpierw otrzymać TGT (Ticket-Granting Ticket). Wpisujemy więc komendę **kinit**.

```
% kinit
Password for your_name@YOUR.REALM:
```

Po wpisaniu hasła, program kinit wysyła prośbę o TGT do AS (Authentication Service). Hasło jest potrzebne do obliczenia klucza, którym zostanie użyty do odkodowania części odpowiedzi z AS. Jeśli hasło jest prawidłowe, otrzymaliśmy TGT. Można to sprawdzić przez komendę **klist**:

```
% klist
Ticket cache: /var/tmp/krb5cc_1234
Default principal: your_name@YOUR.REALM

  Valid starting       Expires          Service principal
24-Dec-02 12:58:02  24-Dec-02 20:58:15  krbtgt/YOUR.REALM@YOUR.REALM
```

„Ticket cache” pokazuje, który plik zawiera nasz uwierzytelniający cache, „default principal” to nazwa osoby dla której jest bilet. W następnych liniach wypisuje wszystkie nasze istniejące bilety. Na razie mamy tylko jeden. Jak widać jest to bilet TGT (w Service principal – krbtgt), który jest ważny jedynie przez 8 godzin.

Teraz używając Kerberowskiej wersji rlogin, będzie on używać TGT aby otrzymać bilet na rlogin deamon, na maszynie do której się logujemy. Wszystko to dzieje się automatycznie:

```
% rlogin nowa.domena
Last login: Fri Jul 21 12:04:40 from etc etc
```

Jedynym sposobem, aby zobaczyć, że naprawdę coś dzieje się z biletami to sprawdzić nasz cache:

```
% klist
Ticket cache: /var/tmp/krb5cc_1234
Default principal: your_name@YOUR.REALM
```

```
  Valid starting       Expires          Service principal
24-Dec-02 12:58:02  24-Dec-02 20:58:15  krbtgt/YOUR.REALM@YOUR.REALM

24-Dec-02 13:03:33  24-Dec-02 20:58:15  host/nowa.domena@YOUR.REALM
```

Jak widać pojawił się nowy bilet, który jest przepustką do nowej.domeny, a którego ważność wygasa o tej samej godzinie co TGT.

Po użyciu biletu, zazwyczaj zostawia się go w cachu, nawet jeśli nie będziemy go już więcej potrzebować. Ale można także nie chcieć, aby nasze uwierzytelnienia zostawały gdzieś w komputerze. Polecenie kdestroy usuwa wszystkie bilety z cachu (łącznie z TGT).

```
% kdestroy
% klist
klist: No credentials cache file found while setting cache flags
(ticket cache /var/tmp/krb5cc_1234)
```

[5], [6]

LITERATURA

[1] http://www.cc.com.pl/security/bezpkerb.html

[2] http://www.ibiblio.org/pub/docs/about-the-net/kerberos-documentation/

[3] http://web.mit.edu/kerberos/www/

[4] William Stallings *Ochrona danych w sieci i intersieci* Warszawa 1997

[5] http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.7/doc/install_toc.html

[6] http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html