Single Sign On, Kerberos

Felhasználó hitelesítés

Single Sign On környezet

- Alap helyzet:
  - Sok gép, sok szolgáltatás, emberek jönnek - mennek
  - Sok jelszó, amit fejben kell tartani, gyakran kéne cserélni. Általában leírják őket
  - Gyorsan új accountok létrehozása, régi jogok törlése

SSO

- Megoldás
  - Single Sign On, avagy Egyszeri belépés
  - csak 1 bejelentkezés, csak 1 jelszó!
- SSO Előnyök
  - Egyetlen jelszó mindenhez: Nem kell fejben tartani, nem kell leírni, nem okoz gondot a sűrű csere
  - Egyetlen adatbázis, könnyebb és gyorsabb adminisztráció. Pl. mi van ha valaki elhagyja a céget...

SSO megközelítések

- Bróker alapú
  - A bróker azonosít és kioszt egy tanúsítványt, amit a szolgáltatásnak be lehet mutatni.
  - Pl.: KERBEROS, SESAME (Secure European System for Applications in a Multi-vendor Environment)
    - Az alkalmazásokat a brókerhez kell igazítani. (Kerberized alkalmazások)
    - Központosított adminisztráció, de mi van, ha megsérül
    - Nagyon sok múlik az azonosításon

SSO megközelítések 2.

- Ügynök alapú
  - Egy ügynök automatikusan azonosít minden alkalmazásnak. (Jelszó listák vagy kulcsok)
    - Pl.: SSH ügynök
  - Előnyök és hátrányok
    - Az alkalmazásokat nem kell módosítani, működik a régiekkel is
    - Nem támogatja az adminisztrációt, sőt még az ügynökök miatt is aggódni kell
    - Az ügynököket biztonságosan kell tárolni

SSO megközelítések 3.

- Token alapú
  - Egy fizikai eszköz egy egyszer használatos kódot szerez, és ezt használja fel.
    - A fizikai eszköz csak plusz komplikáció
    - De biztonságosabb azonosítás
- Átjáró alapú
  - A szolgáltatások egy biztonságos terülten vannak elhelyezve. A hozzáféréshez egy „tűzfalon” kell átjutni.
    - Hasonlít a bróker alapú megoldásra
    - Sok átjáró esetén szinkronizáció kell
- Ügynök és Bróker alapú
  - Egyesíti a két megoldás előnyeit. A felhasználó felé broker alapú, az alkalmazások felé ügynök alapú

Kerberos 5

Kerberos 5

- Kerberos és a jogosítványok
- A felhasználó egy szolgáltatást szeretne igénybe venni egy meghatározott szerveren. Létezik egy azonosítást végző központ, aki azonosítani tudja a szolgáltatást és a felhasználót is. Feladat, hogy a felhasználó azonosítsa magát a szolgáltatásnál.
- De CSAK „Kerberized” alkalmazások!
- MIT fejlesztés
  - 1988 október - Kerberos 4
  - 1993 szeptember - RFC 1510: Kerberos 5

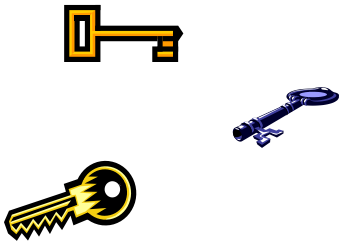
Kerberos azonosítás

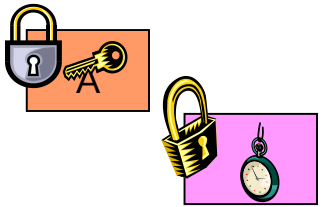
- Biztonság
  - Nincsenek nyílt jelszavak (DES titkosítás használata, a jelszó mint kulcs)
  - A szolgáltatás nem tartja nyilván a felhasználókat
  - A jelszó megszerzése csak 1 felhasználót vagy szolgáltatást tesz kiszolgáltatottá
  - Egyetlen jelszó az összes szolgáltatáshoz (regisztrációnál egyeztetve)

Azonosítás kellékei

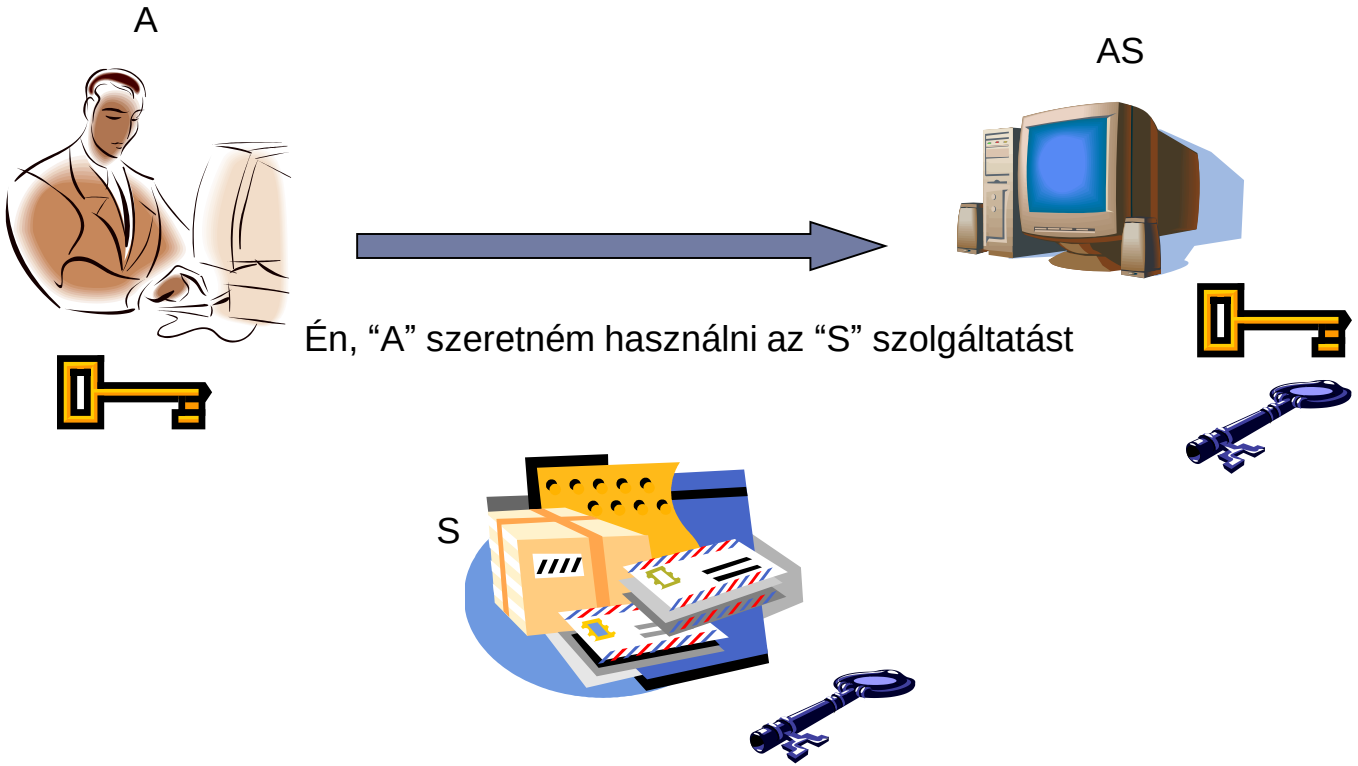
- AS - Authentication Server
- TGS - Ticket Granting Server
- Kulcsok
  - A felhasználó kulcsa (jelszó)
  - A szolgáltatás kulcsa (biztonságos)
  - Az ideiglenes viszonykulcs
- Üzenetek
  - “Ticket”
  - “Authenticator”

KDC: Key Distribution Center

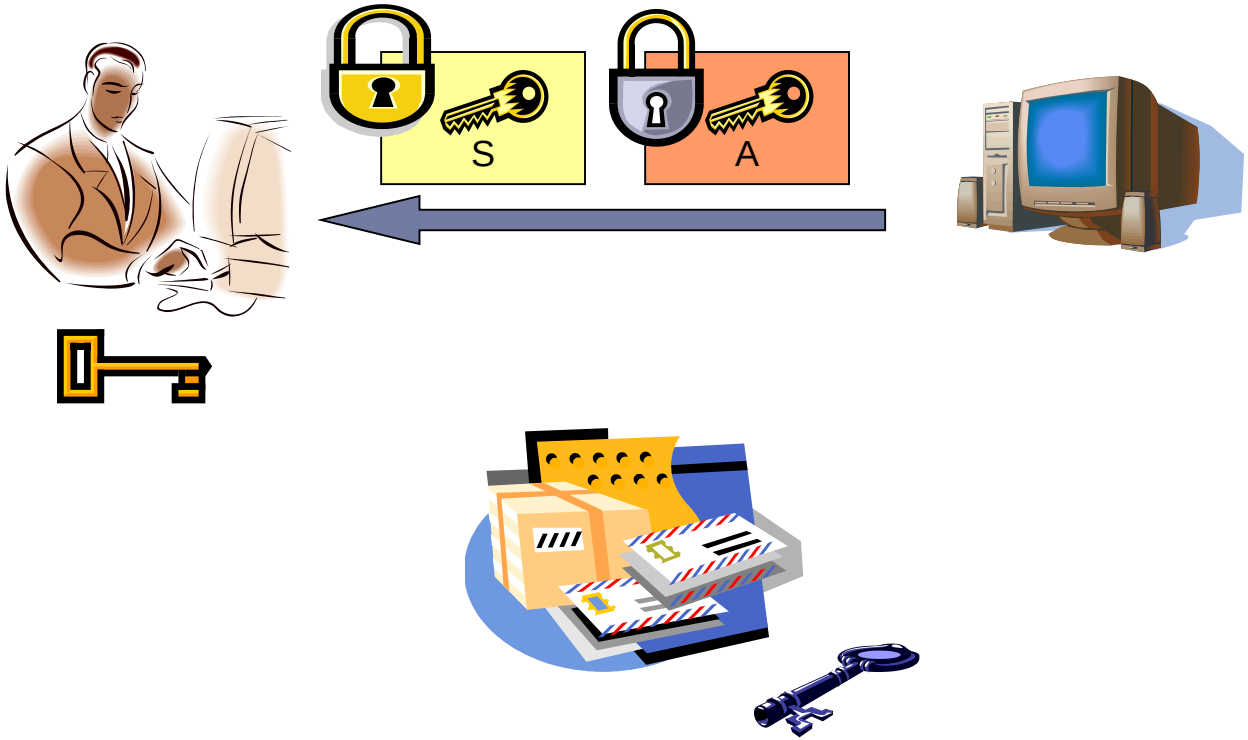




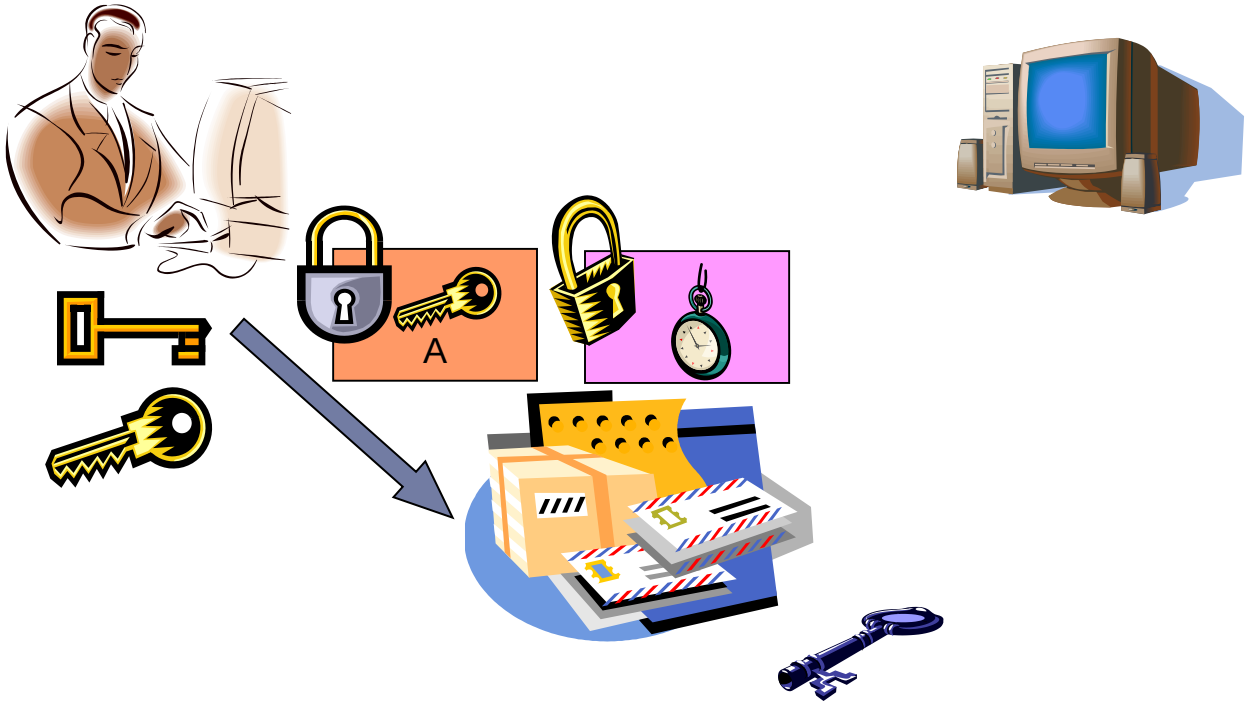
Kerberos AS 1. lépés



Kerberos AS 2. lépés



Kerberos AS 3. lépés

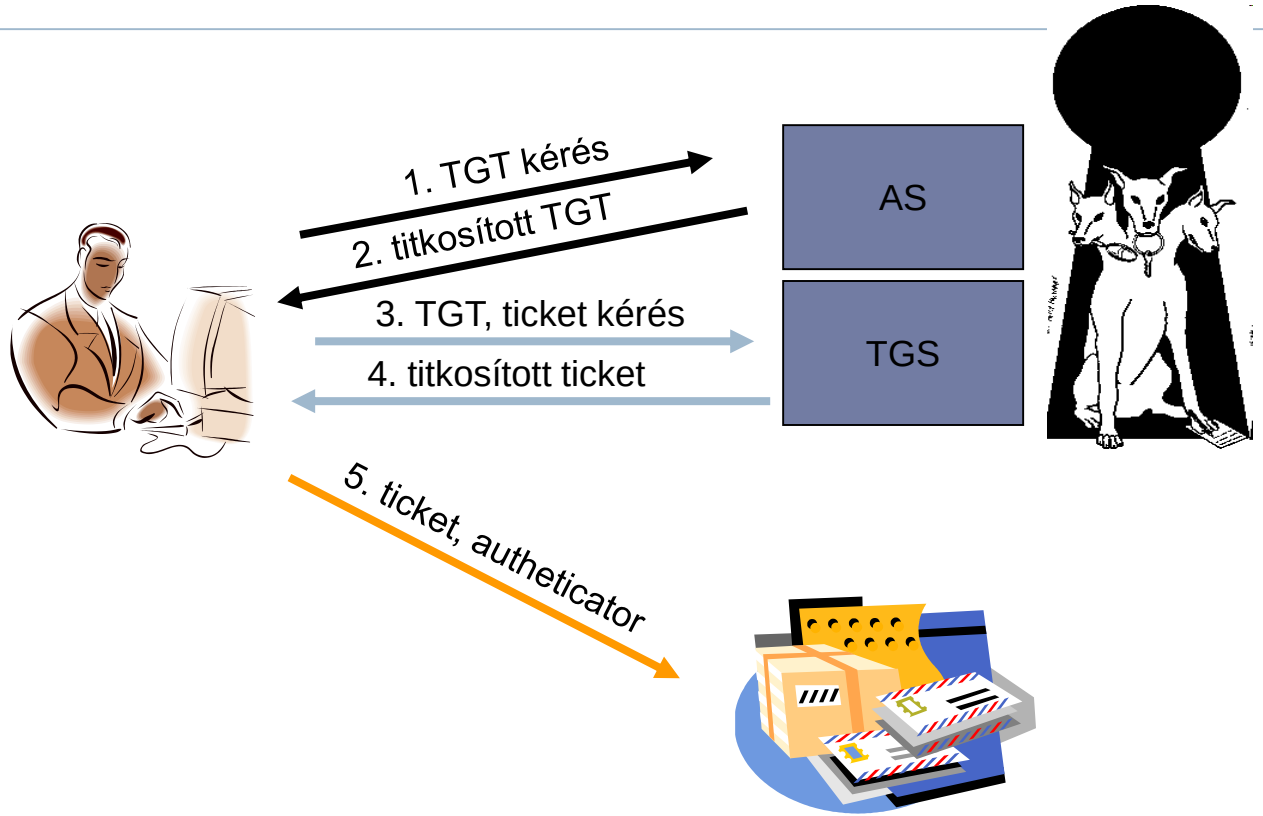


Kerberos AS 4. lépés

Ticket Granting Server (TGS)

- Cél, hogy a felhasználónak ne kelljen minden szolgáltatáshoz jelszavát ismételten megadnia: SSO
- A TGS egy olyan szolgáltatás, amely megszerzi helyette a kívánt viszonykulcsot. Kvázi egy „guest ID” amivel egy vendég bemegy egy épületbe
- TGT: Ticket Granting Ticket, az amit a TGS ad. Ez csak limitált ideig érvényes azonban!
- A TGS külön egység, de sokszor az AS melltett található (együtt: KDC - Key Distribution Center)

Kerberos KDC



Kerberos KDC lépések

1. A -> AS: a
2. AS -> A: $\{K_{a,tgs}\}K_a$, $\{T_{a,tgs}\}K_{tgs}$
3. A -> TGS: $\{A_{a,tgs}\}K_{a,tgs}$, $\{T_{a,tgs}\}K_{tgs}$, s
4. TGS -> A: $\{K_{a,s}\}K_{a,tgs}$, $\{T_{a,s}\}K_s$
5. A -> S: $\{A_{a,s}\}K_{a,s}$, $\{T_{a,s}\}K_s$

A: felhasználó

AS: KDC Hitelesítő

TGS: KDC TGS

S: szolgáltatás

K_x: x kulcsa

$\{X\}K_y$: X titkosítva K_y kulccsal

$\{T_{x,y}\}K_y$: ticket =
y, {x, IP cím, idő, lejárat, $K_{x,y}\}K_y$

$\{A_{x,y}\}K_{x,y}$: authenticator =
{ y, idő$\}K_{x,y}$

Kerberos KDC lépések (folyt.)

- Kezdeti ticket megszerzése
  1. A felhasználó elküldi nevét a Kerberos hitelesítőnek.
  2. A hitelesítő visszaküld egy titkosított viszonykulcsot valamint A TGTt (ticket a TGShez), amely tartalmazza a viszonykulcsot, a felhasználó azonosítóját, az időkorlátot. A ticket titkosítva van a TGS kulcsával

- Ticket a szolgáltatáshoz
  3. A felhasználó authenticatort és TGT küld a TGSnek, valamint jelzi melyik szolgáltatást szeretné használni. A titkosított authenticator bizonyítja, hogy ő is ismeri a viszonykulcsot
  4. A TGS egy új viszonykulcsot generál a felhasználó és a szolgáltatás számára és ezt titkosítva megküldi a felhasználó számára. Ezen kívül küld még egy ticketet, amellyel a felhasználó igénybe tudja venni a szolgáltatást

- A szolgáltatás igénybevétele
  5. A felhasználó elküldi a ticketet és az authenticatort a szolgáltatásnak

  - A szolgáltatás használható, a viszonykulcs segítségével a kommunikáció titkosítható

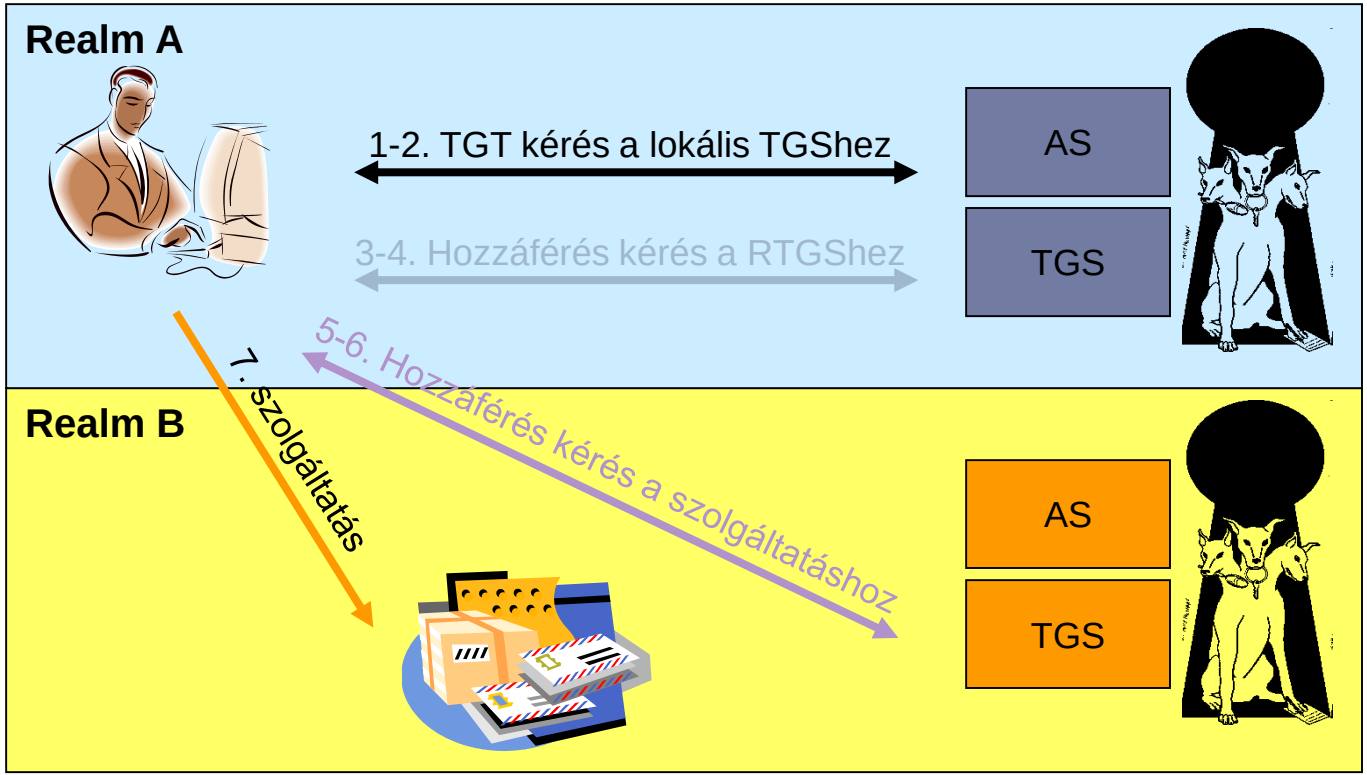
Kerberos előnyök

- Teljes SSO megvalósítás
- A hitelesítés állapotmentes
  - Kiesés esetén a szolgáltatást más Kerberos kiszolgáló tudja folytatni
  - Elosztott Kerberos szolgáltatás
    - Egyetlen „master” adatbázis, a többi kiszolgáló tükrözi és csak olvasható módon használja

(Cross Realm) Tartományok közötti azonosítás

- A hálózat növekedésével az AS és TGS szerverek nem bírják a tempót
- Megoldás a hálózat tartományokra (realm) osztása
- Realm: felhasználók, szolgáltatások, KDC
- Minden tartomány egy saját TGS-t regisztrál a másik tartományokban. (RTGS - Remote TGS) és ezen keresztül éri el a tartományon belüli szolgáltatásokat
  -> Ez viszont nem jó, mert rosszul skálázható
- Az RTGS egységek hiearchikusan is felépíthetőek

Cross Realm hitelesítés



Kerberos 4

- Korábbi változat
  - Az AS csak egy üzenetet küld, amelybe belerejti a “ticket” csomagot is
  - A 4. lépésnél új időbélyeg kerül az üzenetbe
  - Csak peer-to-peer RTGS kapcsolatok, nincs hiearchia

Kerberos alkalmazások

- Bones, E-Bones - A Kerberos Europai változata, az USA törvényei miatt.
- Elsősorban UNIX rendszerek, de:
- Win 2000 már támogatja!
  - SMB fájl-menedzsment
  - Nyomtató sorok
  - QoS kérések, ...
  - Támogatja az eredeti UNIX KDC -t is, de csak amikor nincs megszemélyesítés: pl. adatbázisok.

Kerberos használata

```
% kinit
Password for your_name@your.realm:

% klist
Ticket cache: /var/tmp/krb5cc_1234
Default principal: your_name@your.realm
Valid starting      Expires   Service principal
XXX                           XXX (8 hous)        krbtgt/your.realm@you

% rlogin newhost.domain
…

% klist
Ticket cache: /var/tmp/krb5cc_1234
Default principal: your_name@your.realm
Valid starting      Expires   Service principal
XXX                           XXX (8 hous)        krbtgt/your.realm@you
XXX                           XXX (8 hous)        host/newshost.domain@you
```

Kerberos sebezhetősége

- Üzenetek visszajátszása
  - Az órák szinkronja miatt egy intervallum van megadva. Ezt kivédené egy cache, de gondok vannak az implementációval, főleg UDP esetben. Állapotmentes szolgáltatás!

- Órák szinkronizációjának megtámadása
  - Az idő meghamisítása és így egy kulcs megszerzése. Védekezés egy challange-response rendszerrel lenne lehetséges, de így már állapotok kerülnének a rendszerbe. (+2 üzenet, +állapotok) Lehetne csak opcionális, és használhatnák ekkor a Biztonságos idő szolgáltatásokat

Kerberos sebezhetősége 2.

- Jelszó találgatások
  - A jelszó kódolva utazik a hálózaton, de megfigyelésekkel lehet egy szótáras támadást indítani. Sőt lehet más nevében azonosítást kérni!
  - Megoldás a privát kulcsok alkalmazása (smartcard)

- Hamis AS szerverek
  - Mivel az AS-t nem azonosítják, ezért lehetőség van hamis AS szerverek felállítására

Hitelesítés a jelszavakon túl

Azonosítás módszere

- Egy személyes kulcs
  - Ismerete: jelszó, személyes információ
  - Birtoklása: fizikai kulcs, token, smartcard
  - Viselése: ujjlenyomat, hangminta, aláírás

Többfaktoros azonosítás esetén ezek kombinációja szükséges!

- A kulcs azonosítja a felhasználót a hitelesítő felé
  - Csak a felhasználó ismerheti / birtokolhatja / viselheti a kulcsot
  - Ha valaki más felhasználó használja a kulcsot, akkor megszemélyesíti a kulcshoz társított felhasználót
- A kulcsot fel kell ismernie a hitelesítőnek
  - Általában a hitelesítő is ismeri a kulcsot
  - Erős azonosítás, ha a felhasználó a hitelesítéshez kulcsot nem fedi fel még a hitelesítő előtt sem!
    - A hitelesítő ilyenkor nem ismeri a kulcsot

Azonosítás támadása

- A támadó célja a megszemélyesítés
  - Más azonosítójának felvétele. A támadás után a megszemélyesített felhasználó felel a támadó tetteiért.
- Aktív támadás
  - A hitelesítő támadása
  - Programhibák kihasználása
  - On-line próbálkozások
- Passzív támadás
  - A forgalom megfigyelése, hallgatózás
  - A megfigyelt forgalomból a hitelesítő kulcs megfejtése (off-line)

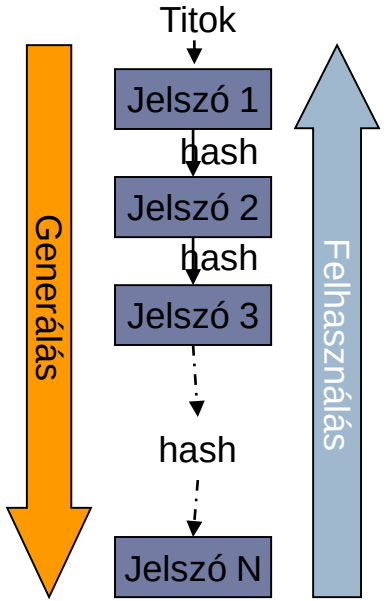
Egyszeri jelszavak

- A visszajátszásos támadások ellen véd, ha minden jelszavat csak egyetlen egyszer használunk
- One Time Passwords (OTP)

- A titokból az azonosításhoz egy egyszer használatos jelszót képzek
  - A titkot nem feltétlenül kell ismernie mindkét félnek!

Egyszeri jelszó: Lamport séma

- Adott a titok több, egymást követő egyirányú transzformáltja
  $h(x), h^2(x), h^3(x), \ldots h^{100}(x)$ jelszavak
- A jelszavak használata a generálással ellentétes irányban történik
- A hitelesítő tárolja a felhasználó nevét és az utolsóként használt jelszavat
- A felhasználó a hitelesítésnél a nála következő (egyel kisebb sorszámú) jelszót használja
  - A hitelesítő nem ismeri a jelszót, azonban, ha az általa tárolt jelszóhoz az adott egyirányú függvénnyel el lehet jutni, akkor a hitelesítés sikeres
  - Sikeres hitelesítésnél a hitelesítő frissíti az utolsó jelszót
- Ha az összes jelszót elhasználták, akkor újakat kell előállítani



S/Key

- Lamport sémája alapján

- Módszer:
  - Adott a felhasználó jelszava (titka) és egy inicializáló mag (seed)
    - A seed segíti, hogy ugyanazt a titkot többször is fel lehessen használni. (Más hitelesítők, vagy új jelszavak generálása)
  - 64 bites (8 bájtos) egyszeri jelszavak
    - Kompromisszum a biztonság és a begépelhetőség között
  - MD4 alapú hash funkció
    - Input: 8 bájt
    - Output: 8 bájt, a 16 bájt MD4 kimenet két 8 bájtos részének XOR kapcsolata
  - A begépelhető jelszavak miatt a 64 bites érték 6 rövid (1-4 betű) szócskára fordítódik egy egységes 2048 szavas nyilvános szótár alapján

S/Key használat

- Cél: hogy lehallgatással ne lehessen támadni
- Nincs titkos algoritmus, A titkot nem tároljuk!

- A hitelesítő az utolsó helyes hitelesítéshez használt jelszót jegyzi, illetve a sikeres hitelesítések számát
  - Hitelesítésnél közli az inicializáló magot (seed) valamint, hogy hányadik jelszót kéri
  - A felhasználó a saját titkából kiindulva meghatározza a kért jelszót

- A jelszavakat előre is lehet kalkulálni, ha a környezet nem biztonságos hozzá -> kódkönyv

S/Key gondok

- Utolsó jelszó tárolása
  - A hitelesítő nem tárolja titkot, de ugyanakkor tárolja az utolsó sikeres jelszót. Ha át tudjuk írni, akkor megszemélyesítettük a felhasználót
- Kommunikáció lehallgatása
  - A kommunikáció lehallgatásával nem tudjuk meg a titkot, de egy elkapott jelszó esetén még off-line találgathatunk
- Több hitelesítő
  - Több hitelesítő esetén a hitelesítőket szinkronizálni kell. Hibás szinkron esetén egy elkapott jelszó visszajátszható
- Kódkönyvek használata
  - A kódkönyvet kilesve megtudjuk a jelszavakat

Secure ID

- A hitelesítő ismeri a titkot, de a felhasználó nem
  - Ellenben van egy eszköze, ami ismeri
  - Két faktoros hitelesítéshez

- Secure ID
  - Eszközök (kb. kulcstartó nagyság) a hitelesítés erősítéséhez
  - Az idő függvényében egyszeri jelszavakat állít elő a hitelesítéshez
    - 64 bites kulcs + 64 bites idő -> 6-8 digites jelszó
    - Az kód egységenként (30 vagy 60 mp) változik
  - PIN kód segítségével + 4-6 digites jelszó

  - Gondot okoz, ha az órák nincsenek szinkronizálva
  - TPM használata ajánlott, a szoftveres megoldás nem az igazi

Secure ID (problémák)

Scientists crack RSA SecurID 800 tokens, steal cryptographic keys

Scientists penetrate hardened security devices in under 15 minutes.

by **Dan Goodin** - Jun 25, 2012 3:20 pm UTC

NATIONAL SECURITY WHITE HAT 67

Padding Oracle támadás

RSA-ba betörés után

Cserélik a SecurID tokeneket

2011.06.07. 14:41 | buherator | 5 komment

Laza három hónappal az **első bejelentés** után az RSA **elismerte**, hogy a tőle megszerzett, SecurID technológiával kapcsolatos adatok veszélyt jelenthetnek az autentikációs megoldást használó rendszerek biztonságára, valamint hogy az incidens kapcsolatban áll a Lockheed Martinnál **történtekkel.**

Smart kártyák

- Nyilvános/privát kulcson alapuló egyeztetés
  - A titkot csak a felhasználó ismeri, a hitelesítőnél a nyilvános része van
  - Csak smart kártyával lehetséges, hiszen a felhasználó képtelen kiszámolni, vagy bevinni az ilyen jelszavakat

- Smart kártya – TPM (Trusted Platform Module)
  - Önálló CPU és memória
  - A privát kulcsot a kártya őrzi és az nem kerül a kártyán kívülre!
  - Minden műveletet a kártya végez
  - A kártyát befogadó gép iránt sincs bizalom
  - A felhasználót ő is azonosítja (PIN kód)

Biometrikus jelszavak

- Mindig „kéznél lévő” nagy bonyolultságú jelszavak
  - Ujjlenyomat
    - Egyre gyakoribb. Egerekbe, laptopokba, telefonba, érintőkijelzőkbe építve.
    - Kis helyigény, gyors használat
    - ! Könnyen ellopható
    - ! Nem megváltoztatható
  - Hangminta
    - Telefonon keresztül is végezhető! (Ügyfélszolgálatok)
    - ! Háttérzajok, betegség, stressz, alkohol befolyásolja
    - ! Visszajátszható

Ujjlenyomat…

Biometrikus jelszavak (folyt.)

- Retina letapogatás
  - Nagyon közelről kell letapogatni (IR lézer) (véredényazonosítás)
  - Nagyon biztos eljárás
- Írisz letapogatás
  - Kb. 30 centi távolságból elegendő
  - Fejlődik (okostelefon?)

- Egyéb, még nem túl elterjedt módszerek
  - Arcfelismerés, kézfelismerés, gépelési sebesség, Aláírás dinamikussága

- Többfaktoros hitelesítés szükséges!

Azonosítás javítása

- Jelszavak bonyolultságának növelése
  - Hosszú jelszavak, gyakori cserék
    - De ezek megjegyezhetetlenek
      - Le kell írni őket vagy túl egyszerűek (összefüggő jelszavak: kukutyin1, kukutyin2, …)
  - A több nem mindig erősebb: UNIX 16 byte, LMHASH

- Megoldás
  - Oktatás a jelszóhasználatról
  - Jelszó helyett más azonosítás
    - Többfaktoros azonosítás
  - SSO megvalósítása

A jelszó továbbítás védelme

- A jelszót továbbító csatorna erős védelme
  - Pl.: SSL, SSH
- Több csatorna egyidejű használata
  - Számítógépes azonosítás + SMS
    - Egy SMS-en kapott kód, mint második jelszó
    - Pl.: Internetes bankoknál