Informaciona bezbednost

Standardi za autentifikaciju

dr Milan Stojkov

Katedra za informatiku

2022.

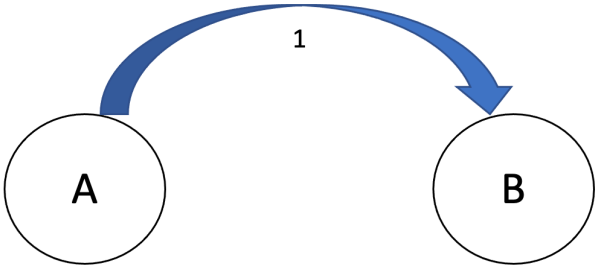
Fakultet tehničkih nauka
Univerzitet u Novom Sadu

X.509 Authentication Service

- Hijerarhijski direktorijumski servis X.500 čuva informacije o korisnicima za potrebe autentifikacije
- Informacije su smeštene u sertifikate potpisane tajnim ključem treće strane kojoj svi veruju
- Definiše tri tipa autentifikacione procedure:
  - *One-way authentication*
  - *Two-way authentication*
  - *Three-way authentication*
- *One-way* i *two-way* autentifikacija se koriste na webu u okviru SSL protokola

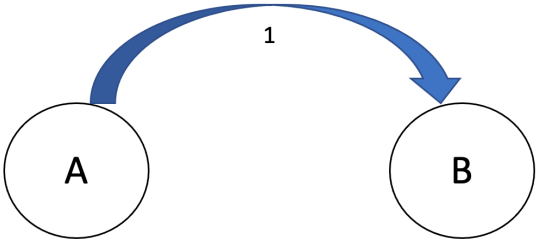
One-way authentication

- *Prover* šalje *verifieru* informacije kojima se potvrđuje:
  - Da je *prover* formirao poruku
  - Da je poruka upućena *verifieru*
  - Da poruka nije bila modifikovana ili ponovo poslata (*replay*) u prenosu
- Utvrđuje se samo identitet *provera*



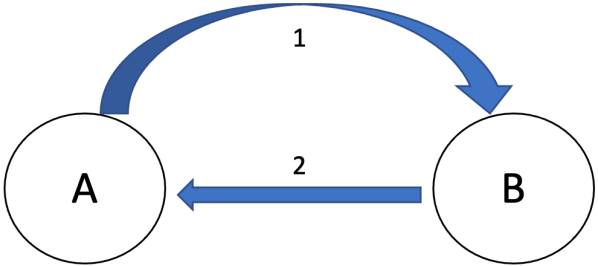
One-way authentication

- Poruka mora da sadrži minimalno:
  - Slučajan broj *rand*, kojim se sprečava *replay* napad
  - Timestamp *tmp*, koji uključuje vreme formiranja poruke
  - Identitet *verifiera*
  - Potpisani podaci koji uključuju *rand* i *tmp*, praćeni sertifikatom *provera*
- A opciono i:
  - Podaci šifrovani javnim ključem *verifiera* koji mogu poslužiti za uspostavljanje *session* ključa



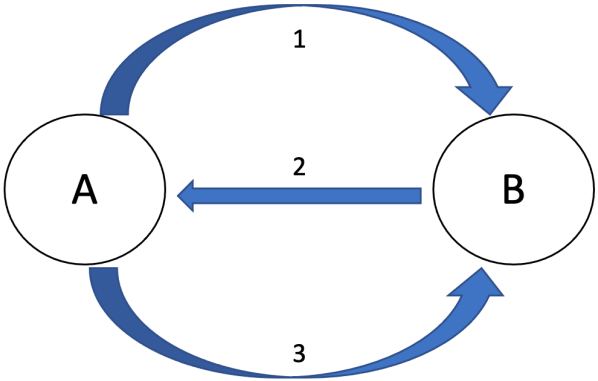
Two-way authentication

- Omogućava obostranu autentifikaciju
- Dodatna poruka kojom *verifier* potvrđuje svoj identitet:
  - Primljeni slučajni broj *rand*
  - Novi slučajni broj *rand'*
  - Potpisani podaci koji uključuju *rand*, *rand'* i identitet *provera* kojima se potvrđuje identitet *verifiera*



Three-way authentication

- Uključuje i treću poruku koju *prover* šalje *verifieru*, koja sadrži:
  - Slučajan broj *rand'*
  - Potpisane podatke koji služe za sinhronizaciju časovnika



HTTP autentifikacija

- Postoje dva tipa autentifikacije po HTTP standardu
  - *Basic Authentication*
    - Relativno često korišćena
  - *Digest Access Authentication*
    - Vrlo retko korišćena
    - Često nepravilno implementirana

HTTP Basic Authentication

- Klijenti se identifikuju na osnovu korisničkog imena i lozinke
- Izvodi se na sledeći način:
  - Klijent traži željeni resurs
  - Server proveri da li je pristup resursu ograničen
  - Ako je ograničen, proveri da li je klijent već ranije poslao `username:password`
  - Ako nije, kao odgovor mu se šalje:

    ```
    HTTP/1.1 401 Unauthorized
    WWW-Authenticate: Basic realm=OBLAST
    ```
- Klijentu se prikaže prozor za unos `username:password` i ponovo traži željeni resurs
- Ako `username:password` nisu ispravni, dobija se isti odgovor (401)

HTTP Basic Authentication

- `username:password` se šalju u okviru zaglavlja HTTP zahteva

```
GET ... HTTP/1.1 ...
Authorization: Basic cHJvYmE6cHJvYmE=
```

- Ime i lozinka se pakuju kao `ime:lozinka` (razdvojeni dvotačkom) i kodiraju **Base64** algoritmom
- **Base64** algoritam nije kriptografski algoritam:
  - Predstavlja brojeve u brojnom sistemu sa osnovom 64
  - Server proveri da li je pristup resursu ograničen
  - Najveća osnova brojnog sistema takva da se brojevi mogu predstaviti ASCII karakterima
  - Služi za pakovanje binarnih sadržaja u tekstualni format
- Primer: email poruke sa zakačenim binarnim fajlovima

HTTP Basic Authentication

- Jednostavan mehanizam za autentifikaciju
- `username:password` putuju od klijenta do servera kao otvoreni tekst
- Nema zaštite od prisluškivanja
- Nizak nivo zaštite

HTTP Digest Access Authentication

- Ispravlja osnovnu manu Basic metode
  - Korisničko ime i lozinka se ne šalju preko mreže, već samo njihov hash kod
  - Na serverskoj strani se ne moraju čuvati ime i lozinka, već njihov hash kod
- Radi po *challenge-response* principu:
  - Server pošalje klijentu kodiranu informaciju
  - Klijent vraća korisničko ime, lozinku i kodiranu informaciju

HTTP Digest Access Authentication

- Ispravlja osnovnu manu Basic metode
  - Korisničko ime i lozinka se ne šalju preko mreže, već samo njihov hash kod
  - Na serverskoj strani se ne mora čuvati ime i lozinka, već njihov hash kod
- Radi po *challenge-response* principu:
  - Server pošalje klijentu kodiranu informaciju
  - Klijent vraća korisničko ime, lozinku i kodiranu informaciju

HTTP Digest Access Authentication

- Tok komunikacije
  - Klijent traži željeni resurs
  - Server proveri da li je pristup resursu ograničen
  - Ako je ograničen, proveri da li je klijent već ranije poslao `username:password`
  - Ako nije, kao odgovor mu se šalje:

    ```
    HTTP/1.1 401 Unauthorized
    WWW-Authenticate: Digest realm=OBLAST, nonce="...", ...
    ```
- Klijentu se prikaže prozor za unos `username:password` i ponovo traži željeni resurs
- Ako `username:password` nisu ispravni, dobija se isti odgovor (401)

HTTP Digest Access Authentication

- Struktura **WWW-Authenticate** zaglavlja:

```
WWW-Authenticate: Digest
realm="IME",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
qop="auth,auth-int",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

  - *realm* – ime zaštićene oblasti
  - *nonce* – jednokratna slučajna vrednost
  - *qop* (*quality of protection*) – *auth* (*authentication*) i/ili *auth-int* (*authentication with integrity protection*)
  - *opaque* – string koji bi klijent trebalo da pošalje za svaki naredni zahtev unutar iste oblasti

HTTP Digest Access Authentication

- Odgovor klijenta stiže u sledećem zahtevu, u zaglavlju Authorization:

```
Authorization: Digest
username="proba",
realm="IME",
qop="auth",
algorithm="MD5",
uri="/protect.html",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
nc=00000001
cnonce="aa671f12a8587e2cffe73519863f9dbb",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
response="f90a24d42f7d59fa0496cbbce6aacfe6"
```

  - *uri* – zahtevani resurs
  - *nc* – brojač zahteva za istim nonce poljem
  - *cnonce* – koristi se za izračunavanje response
  - *response* – sadrži hash kod za korisničko ime i lozinku

HTTP Digest Access Authentication

- Izračunavanje response polja
  - HA1 = MD5(username + ":" + realm + ":" + password)
  - HA2 = MD5(http_method + ":" + uri)
  - response=MD5(HA1 + ":" + nonce + ":" + nc + ":" + cnonce + ":" + qop + ":" + HA2)

HTTP Digest Access Authentication

- Lozinka ne putuje preko mreže, već samo njen hash
- Otkrivanje lozinke prisluškivanjem nije moguće
- Preneti sadržaji i dalje nisu zaštićeni od prisluškivanja (samo lozinka jeste)
- Nema zaštite od man-in-the-middle napada

NTLM - verzije

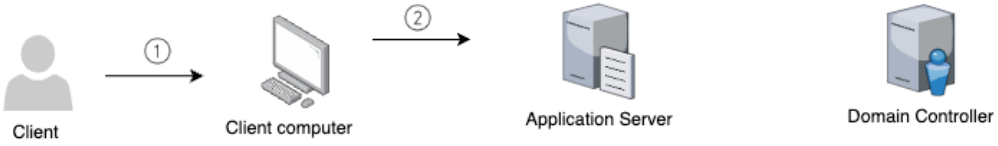
- Lan Manager (LM)
  - Slab metod za heširanje lozinki
- New Technology Lan Manager (NTLMv1)
  - Slab challenge (koristi DES-ECB)
- NTLMv2
  - Podržava bolje kriptografske algoritme (HMAC-MD5)

NTLM - tok komunikacije

1. Korisnik unosi naziv domena, korisničko ime i lozinku
   - Računar izračunava hash lozinke i odbacuje originalnu lozinku

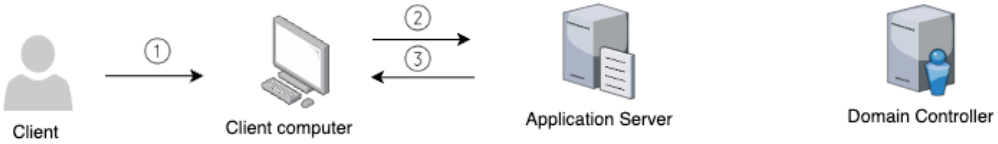
NTLM - tok komunikacije

2. Korisnik šalje korisničko ime serveru (u *plaintextu*)



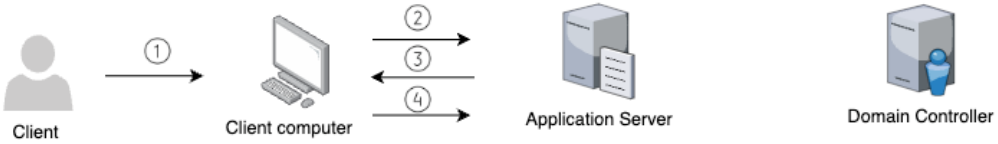
NTLM - tok komunikacije

3. Server generiše nasumučni broj dužine 16 bajta (*challenge* ili *no*nce) i šalje klijentu



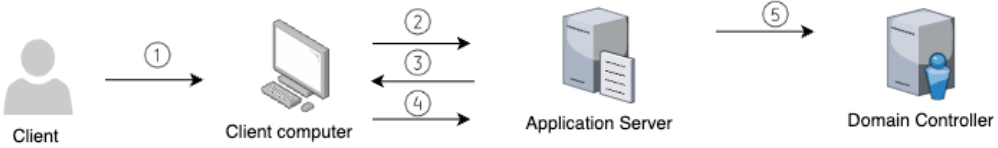
NTLM - tok komunikacije

4. Klijent šifruje *challenge hash*-om korisničke lozinke i vraća rezultat serveru (*response*)



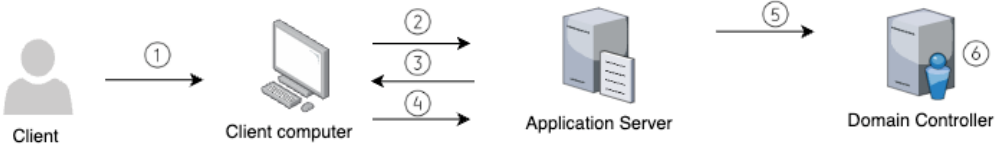
NTLM - tok komunikacije

5. Server šalje domen kontroleru 3 informacije: korisničko ime, *challenge* poslat klijentu i *response* dobijen od klijenta



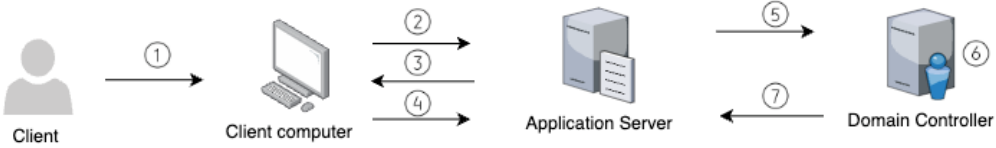
NTLM - tok komunikacije

6. Domen kontroler koristi korisničko ime da dobije *hash* korisnikove lozinke iz *Security Account Manager* (SAM) baze
   - Koristi *hash* te lozinke da šifruje *challenge*



NTLM - tok komunikacije

7. Domen kontroler poredi šifrovani *challenge* koji je izračunao u koraku 6 sa odgovorom koji je izračunao klijent u koraku 4 (*response*)
   - Ako su isti, korisnik je autentifikovan i domen kontroler obaveštava server o tome



LM

- Karakteri koji se mogu koristiti za formiranje lozinke su ANSI printabilni karakteri (95 karaktera)
- Lozinke duže od 7 karaktera se dele na dva dela i svaki deo se hešuje odvojeno
- Napadi grubom silom (*brute-force*)
  - $95^7 \sim 2^{46}$ lozinki sa 7 printabilnih karaktera
  - Ne razlikuje velika i mala slova te lozika koja sadrži samo slova ima $69^7 \sim 2^{43}$ kombinacija
- LM hash ne sadrži *salt*

NTLMv1

- Razvijen da zameni LAN Managera
- Uzima lozinku i računa MD4 heš (128 nasumičnih bita - 16 bajta)
- Čuva tu vrednost, tzv. NTLM hash
  - 2^{92} lozinki sa 14 printabilnih karaktera

NTLMv1

- MD4 algoritam se primenjuje na lozinku i nastaje *NTLM hash*
  - Npr, 0x0123456789ABCDEFFEDCBA9876543210
- *NTLM hash* od 16 bajta se dopunjuje nulama do dužine od 21 bajta
- Deli se na tri dela od 7 bajta
  - Key 1: 0123456789ABCD
  - Key 2: EFFEDCBA987654
  - Key 3: 32100000000000
- Svaki od ključeva se DES šifruje zajedno sa challenge porukom (dobijaju se 3 8-bajtne vrednosti)
  - NTLMv1 Response = DES(Key1, Challenge) + DES(Key2, Challenge) + DES(Key3, Challenge)
- Te tri vrednosti se konkateniraju da oforme 24-bajtnu vrednost (*response*)

NTLMv1

- Kreiranje *response* vrednosti zahteva samo *NTLM hash*
- *NTLM hash* postaje isto što i lozinka i ko ima *NTLM hash* ima i pristup bez lozinke (**pass-the-hash**)
- Podložan *dictionary* napadima
- Koristi se DES algoritam koji je nesiguran

NTLMv2

- Dizajniran da eliminiše mogućnost *dictionary* napada
- Koristi klijentski *no*nce uz serverski *challenge*/*nonce* tokom generisanja odgovora (*response*)
- Dodatni *nonce* menja veličinu odgovora

NTLMv2

- Klijent i server generišu nasumičnu *challenge* vrednost:
  - CS = nasumični 8-bajtni server *challenge*
  - CC = nasumični 8-bajtni klijentski *challenge*
  - blob = (time, CC, domain_name)
- Računanje NTLMv2 odgovora
  - v2-Hash = HMAC-MD5(NTLM hash, username, domain_name)
  - NTLM hash je MD4 hash lozinke
  - NTv2 = HMAC-MD5(v2-Hash, CS, blob)
  - NTLMv2 response = CC | NTv2 | blob

NTLMv2

Osobine	LM	NTLMv1	NTLMv2
Lozinka prepoznaje velika/mala slova	Ne	Da	Da
Dužina hash ključa	56 + 56 bit	-	-
Hash algoritam za lozinku	DES (ECB)	MD4	MD4
Dužina hash vrednosti	64 + 64 bit	128 bit	128 bit
Challenge-Response dužina ključa	56 + 56 + 16 bit	56 + 56 + 16 bit	128 bit
Challenge-Response algoritam	DES (ECB)	DES (ECB)	HMAC_MD5
Challenge-Response dužina vrednosti	64 + 64 + 64 bit	64 + 64 + 64 bit	128 bit

Kerberos

- Nastao na MIT-u 1980-tih
- Verzije 1-3 su se koristile samo interno na MIT-u
- Verzija 4 u javnoj upotrebi
- Verzija 5 ispravlja neke nedostatke i postaje široko rasprostranjena (Windows)
- Kerberos = centar za distribuciju ključeva ima tri „glave“
  - Bazu podataka
  - Server za proveru identiteta
  - Server za izdavanje karata
- Omogućava **single sign-on** (SSO): korisnik se prijavi samo jednom a potom (u skladu sa svojim pravima) ima pristup svim resursima na mreži
  - Upravljanje velikim brojem korisničkih naloga
  - Efikasan pristup pojedinačnim resursima
  - Lozinke se nikad ne šalju kao otvoreni tekst

Kerberos

- Centar za proveru identiteta → *Key Distribution Center* (KDC)
  - Baza u kojoj se čuvaju provereni parametri svih učesnika Kerberos sistema
  - Svi učesnici mu bezuslovno veruju
  - Svaki učesnik (korisnik, računar, mrežni servis) predstavljen je imenom u KDC bazi

Kerberos Principal

- Koncept principala: jednoznačno identifikuje učesnika
  - Za principal je vezan tajni ključ za autentifikaciju učesnika kod KDC-a
- Struktura principala: identity/instance@realm
  - *identity*: ime Kerberos učesnika; obavezno polje
  - *instance*: ime računara na kome se nalazi resurs, ili ime korisničke grupe; nije obavezno
  - *realm*: identifikator pojedinačnih Kerberos okruženja; po konvenciji formira se kao uppercase DNS ime domena organizacije, npr. FTN.UNS.AC.RS
- Primeri principala:
  - `goran/nastavnici@FTN.UNS.AC.RS` → korisnik goran član grupe nastavnici
  - `ssh/informatika.ftn.uns.ac.rs@FTN.UNS.AC.RS` → ssh servis na računaru informatika.ftn.uns.ac.rs
  - `zozon/zozon.ftn.uns.ac.rs@FTN.UNS.AC.RS` → računar zozon.ftn.uns.ac.rs

Kerberos KDC

- Sastoji se iz tri komponente:
- Baza principala: evidencija svih principala i njihovih tajnih ključeva
  - Implementacija baze: na Linuxu LDAP, na Windowsu Active Directory
- Server za autentifikaciju (*authentication server*, AS)
  - Izdaje TGT kartu svim klijentima prilikom prijave na sistem
  - Pomoću nje klijenti kasnije traže pristup resursima
  - TGT = *ticket-granting ticket*
- Server za dodelu karata (*ticket granting server*, TGS)
  - Zadužen za izdavanje ST karata namenjenih pojedinim resursima
  - Pomoću ST karata klijenti pristupaju resursima
  - ST = *service ticket*

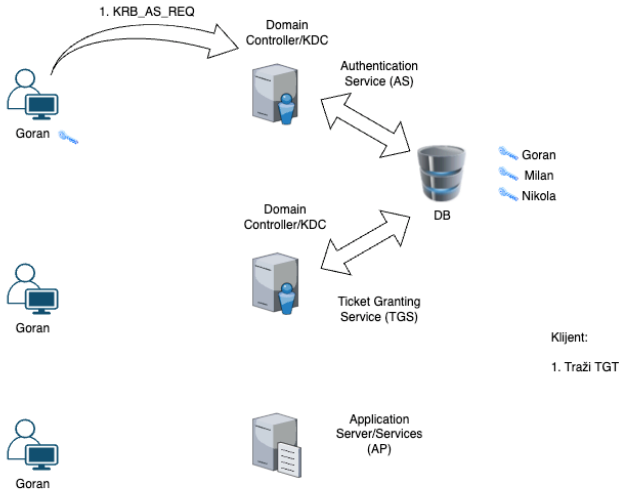
Kerberos KDC

- KDC mora da funkcioniše da bi klijenti mogli pristupati resursima
- Potencijalno usko grlo i SPoF (*single point of failure*)
- KDC na više servera
  - 1 *master*/primarni KDC
  - Više *slave*/sekundarnih KDC - preuzimaju ulogu ukoliko je primarni nedostupan
  - Izmene baze samo na primarnom serveru!

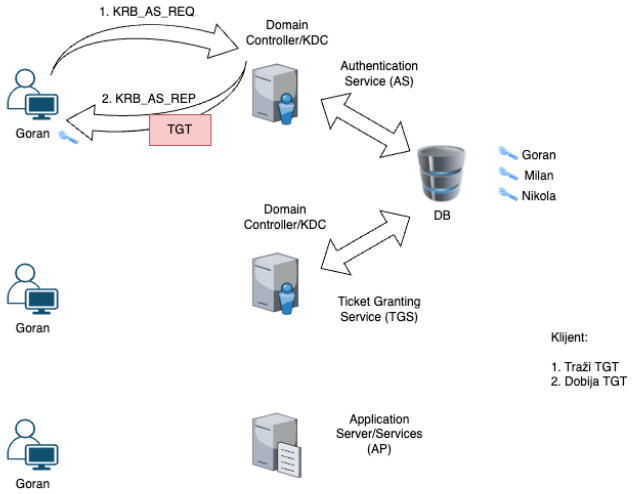
Ticket-granting ticket

- TGT karta sadrži
  - Ime principala koji traži pristup
  - Ime principala kome se želi pristupiti
  - Timestamp (trenutak formiranja zahteva)
  - Rok važenja karte
  - Listu IP adresa sa kojih se može koristiti karta
  - Tajni ključ za komunikaciju sa resursom

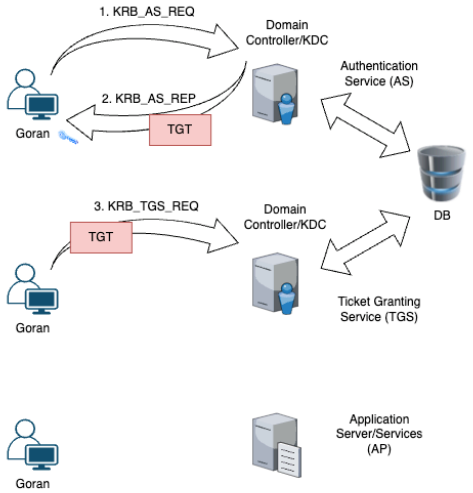
Kerberos - tok komunikacije



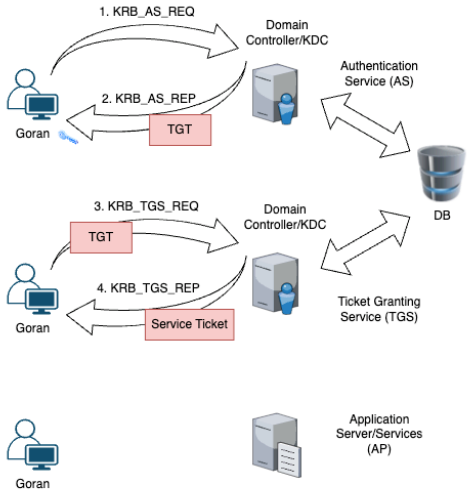
Kerberos - tok komunikacije



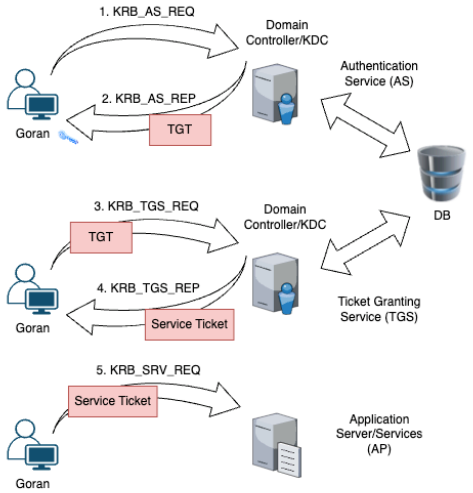
Kerberos - tok komunikacije



Kerberos - tok komunikacije



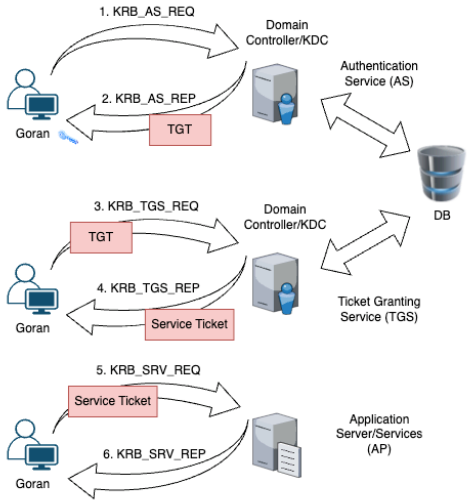
Kerberos - tok komunikacije



Klijent:

1. Traži TGT
2. Dobija TGT
3. Šalje TGT, traži ST
4. Dobija ST
5. Pristupa uz ST

Kerberos - tok komunikacije



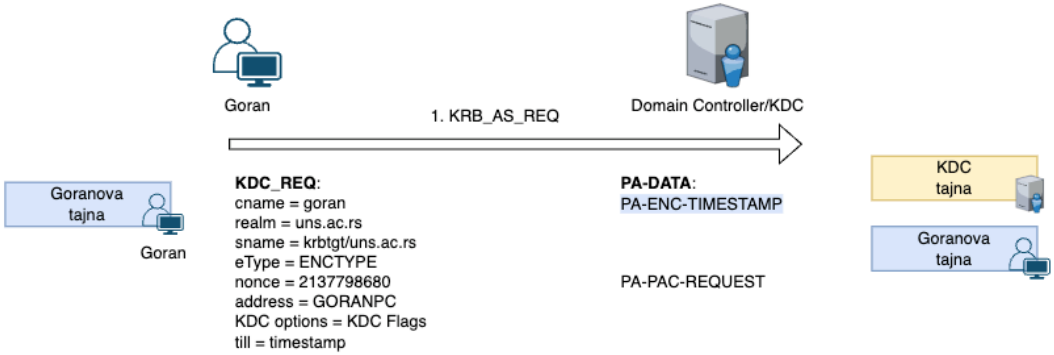
Klijent:

1. Traži TGT
2. Dobija TGT
3. Šalje TGT, traži ST
4. Dobija ST
5. Pristupa uz ST
6. Biva uslužen

Kerberos - tok komunikacije

1. KRB_AS_REQ zahtev - počinje autentifikaciju na KDC AS serveru
   - Šalje se kao otvoreni tekst
   - Sadrži:
     - Ime principala koji šalje zahtev
     - Timestamp - vreme generisanja zahteva
     - Ime principala TGS servera
     - Traženi rok važenja karte

KRB_AS_REQ


Goran
1. KRB_AS_REQ
Domain Controller/KDC
Goranova tajna
Goran
KDC_REQ:
cname = goran
realm = uns.ac.rs
sname = krbtgt/uns.ac.rs
eType = ENCTYPE
nonce = 2137798680
address = GORANPC
KDC options = KDC Flags
till = timestamp
PA-DATA:
PA-ENC-TIMESTAMP
PA-PAC-REQUEST
KDC tajna
Goranova tajna

Kerberos - tok komunikacije

2. KRB_AS_REP odgovor
   - AS proverava da li principal postoji u bazi
   - Ako postoji, proverava se da li je timestamp u dozvoljenom vremenskom okviru
   - Odgovor ima dva dela

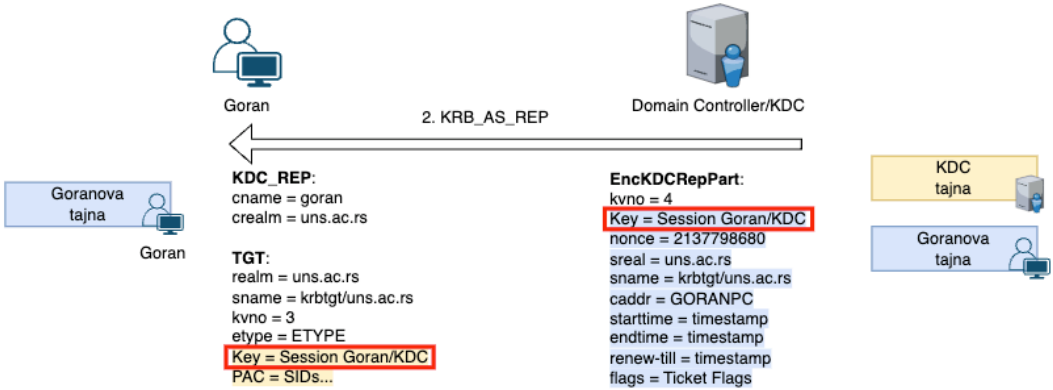
Kerberos - tok komunikacije

2. KRB_AS_REP odgovor
   - Prvi deo se šifruje tajnim ključem poznatom samo KDC-u i učesniku
   - Prvi deo sadrži:
     - Ključ sesije koji će klijent u nastavku koristiti za komunikaciju sa TGS serverom
     - Ime principala TGS servera
     - Rok važenja karte

Kerberos - tok komunikacije

2. KRB_AS_REP odgovor
   - Drugi deo sadrži:
     - TGT kartu šifrovanu tajnim ključem koji koriste AS i TGS (TGS key)
     - Session ključ za komunikaciju klijent-TGS
     - Ime principala Kerberos klijenta
     - Rok važenja karte
     - Timestamp KDC servera
     - Klijentovu IP adresu
   - Klijent ne može da pročita ovaj deo poruke!
   - Klijent će ga sačuvati u kešu i koristiti prilikom slanja zahteva za pristup resursima

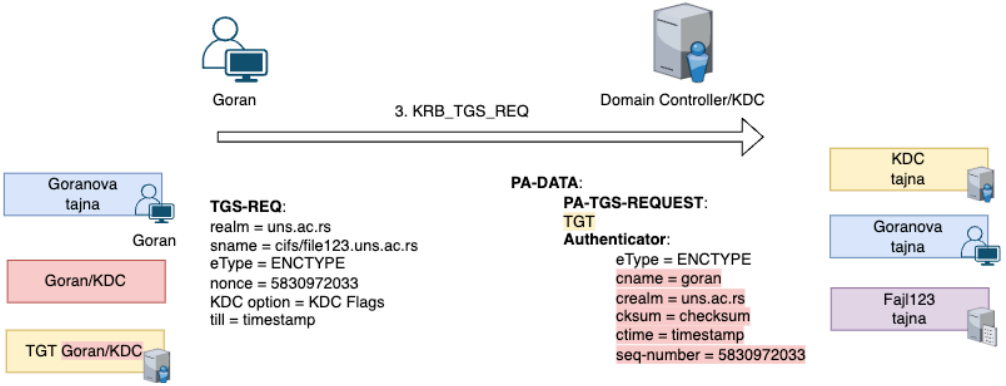
KRB_AS_REP


Goran
Domain Controller/KDC
2. KRB_AS_REP
Goranova tajna
Goran
KDC_REP:
cname = goran
crealm = uns.ac.rs
TGT:
realm = uns.ac.rs
sname = krbtgt/uns.ac.rs
kvno = 3
etype = ETYPE
Key = Session Goran/KDC
PAC = SIDs...
EncKDCRepPart:
kvno = 4
Key = Session Goran/KDC
nonce = 2137798680
sreal = uns.ac.rs
sname = krbtgt/uns.ac.rs
caddr = GORANPC
starttime = timestamp
endtime = timestamp
renew-till = timestamp
flags = Ticket Flags
KDC tajna
Goranova tajna

Kerberos - tok komunikacije

3. KRB_TGS_REQ zahtev
   - Klijent dešifruje prvi deo KRB_AS_REP odgovora
   - Ključ sesije i šifrovanu TGT kartu smešta u keš
   - Zahtev za pristup konkretnom resursu klijent šalje TGS serveru
   - Sadrži:
     - Ime principala resursa kome klijent želi da pristupi
     - Traženi rok važenja karte
     - TGT kartu iz prethodnog koraka
     - Autentifikator
       – Obezbeđuje jedinstvenost svakog zahteva za pristup resursu
       – Potvrđuje da korisnik ima prethodno ugovoren tajni ključ sesije

KRB_TGS_REQ


Goran
Domain Controller/KDC
3. KRB_TGS_REQ
Goranova tajna
Goran
Goran/KDC
TGT Goran/KDC
TGS-REQ:
realm = uns.ac.rs
sname = cifs/file123.uns.ac.rs
eType = ENCTYPE
nonce = 5830972033
KDC option = KDC Flags
till = timestamp
PA-DATA:
PA-TGS-REQUEST:
TGT
Authenticator:
eType = ENCTYPE
cname = goran
crealm = uns.ac.rs
cksum = checksum
ctime = timestamp
seq-number = 5830972033
KDC tajna
Goranova tajna
Fajl123 tajna

Kerberos - tok komunikacije

4. KRB_TGS_REP odgovor
   - TGS server formira novi ključ sesije koji će klijent koristiti za komunikaciju sa resursom
   - Odgovor (ponovo) ima dva dela

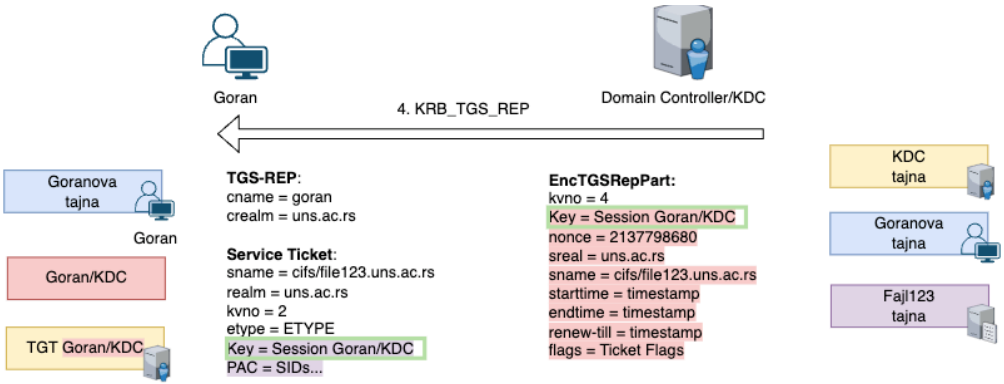
Kerberos - tok komunikacije

4. KRB_TGS_REP odgovor
   - Prvi deo je šifrovan ključem sesije koji je ustanovljen u koracima 1 i 2
     - Ime principala resursa kome klijent želi da pristupi
     - Rok važenja karte
     - Ključ sesije za komunikaciju sa resursom kome klijent želi da pristupi

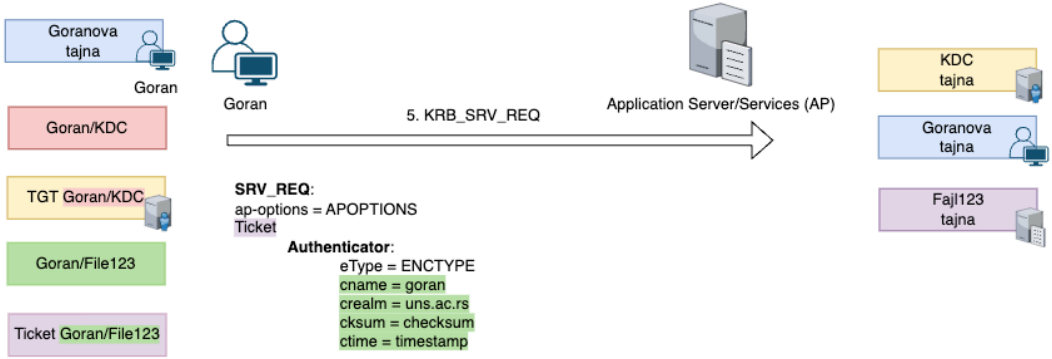
Kerberos - tok komunikacije

4. KRB_TGS_REP odgovor
   - Drugi deo je ST karta za pristup resursu šifrovana tajnim ključem koji dele TGS i resurs
     - Ključ sesije za komunikaciju klijent-resurs
     - Ime principala klijenta
     - Rok važenja karte
     - Timestamp KDC servera
     - Klijentova IP adresa

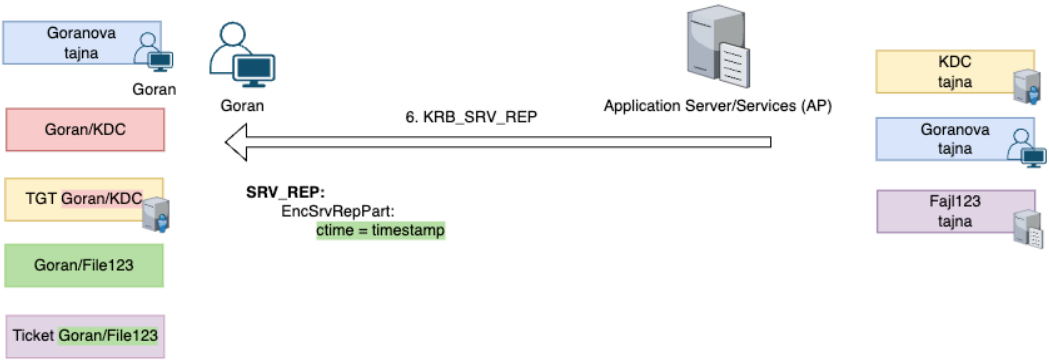
KRB_TGS_REP


Goran
Domain Controller/KDC
4. KRB_TGS_REP
Goranova tajna
Goran
Goran/KDC
TGT Goran/KDC
TGS-REP:
cname = goran
crealm = uns.ac.rs
Service Ticket:
sname = cifs/file123.uns.ac.rs
realm = uns.ac.rs
kvno = 2
etype = ETYPE
Key = Session Goran/KDC
PAC = SIDs...
EncTGSRepPart:
kvno = 4
Key = Session Goran/KDC
nonce = 2137798680
sreal = uns.ac.rs
sname = cifs/file123.uns.ac.rs
starttime = timestamp
endtime = timestamp
renew-till = timestamp
flags = Ticket Flags
KDC tajna
Goranova tajna
Fajl123 tajna

KRB_SRV_REQ


Goranova tajna
Goran
Goran/KDC
TGT Goran/KDC
Goran/File123
Ticket Goran/File123
Goran
5. KRB_SRV_REQ
Application Server/Services (AP)
KDC tajna
Goranova tajna
Fajl123 tajna
SRV_REQ:
ap-options = APOPTIONS
Ticket
Authenticator:
eType = ENCTYPE
cname = goran
crealm = uns.ac.rs
cksum = checksum
ctime = timestamp

KRB_SRV_REP


Goranova tajna
Goran
Goran/KDC
TGT Goran/KDC
Goran/File123
Ticket Goran/File123
Goran
Application Server/Services (AP)
6. KRB_SRV_REP
SRV_REP:
EncSrvRepPart:
ctime = timestamp
KDC tajna
Goranova tajna
Fajl123 tajna

Kerberos - tipovi ticketa

- *Renewable Ticket*
  - Svaki *ticket* ima ograničeni rok važenja, van toga ne može se izvršiti razmena za autentifikaciju
  - Aplikacije možda žele da zadrže *ticket* koji može da važi duži vremenski period
  - Ovo može da izloži njihov tajni *session* ključ krađi, a ti ukradeni ključevi bi važili do isteka *ticketa*
  - Korišćenje kratkotrajnih *ticketa* i dobijanje novih zahtevalo bi od klijenta dugoročan pristup svom tajnom ključu što je još veći rizik
  - Klijent koji ima *renewable ticket* mora da je pošalje, uz ponovnu autentifikaciju u KDC na obnavljanje pre nego što prođe vreme trajanja *ticketa* (radi se osvežavanje ključa sesije)
- *Post Dated Ticket*
  - Aplikacije mogu ponekad da dobiju *ticket* za kasnije korišćenje
  - Npr. sistem za obradu podataka u *batch*-u zahteva da *ticket* bude validan u trenutku kada se *batch* posao bude obavljao
  - Opasno je držati važeće *tickete* u *batch* nizu, jer će oni duže biti dostupni podložnije krađi
  - *Post Dated ticketi* omogućavaju njihovo dobavljanje od AS-a u vreme podnošenja zahteva za *batch* izvršavanje, ali se oni ostavljaju „u mirovanju“ dok se ne aktiviraju i validiraju daljim zahtevom AS-a

Kerberos - tipovi ticketa

- *Proxiable Ticket*
  - Može biti neophodno da principal dozvoli servisu da izvrši operaciju u njegovo ime
  - Servis mora biti u mogućnosti da preuzme identitet klijenta, ali samo za određenu svrhu
  - Principal može dozvoliti servisu da preuzme identitet za određenu svrhu tako što će mu dodeliti *proxy*
  - Ovo omogućava klijentu da prosledi *proxy* serveru da izvrši zahtev u njegovo ime
  - Npr, klijent može da da serveru za štampanje *proxy* pristup svojim datotekama na fajl serveru kako bi zadovoljio zahtev za štampanje
- *Forwardable Ticket*
  - Prosleđivanje autentifikacije je primer *proxy*-ja u kojem se servisu dodeljuje kompletna upotreba identiteta klijenta
  - Npr, korisnik se prijavi na udaljeni sistem i želi da autentifikacija radi sa tog sistemu kao da korisnik radi u lokalu

Kerberos - Cross Realm Authentication

- Kerberos protokol je dizajniran da radi van granica organizacije
- Klijent u jednoj organizaciji može biti autentifikovan na serveru u drugoj
- Svaka organizacija koja želi da ima Kerberos server uspostavlja svoju sopstvenu oblast (*realm*)
- Naziv oblasti u kojoj je klijent registrovan je deo imena klijenta i tu informaciju može da koristi servis da odluči da li će obraditi zahtev

Kerberos - Cross Realm Authentication

- Uspostavljanjem ključeva između oblasti (*inter-realm*), administratori dva domena mogu da dozvole klijentu koji je autentifikovan u lokalnom domenu da koristi svoju autentifikaciju na udaljenom domenu
- Razmena ovih ključeva omogućava da se registruje TGS svake oblasti kao principal u drugoj oblasti
- Klijent tada može da dobije TGT za udaljenu oblast iz svog lokalne oblasti
- Kada se koristi TGT, TGS koristi *inter-realm* ključ (koji se obično razlikuje od lokalnog TGS ključa) da dešifruje TGT i tako potvrđuje da ga je izdao klijentov TGS
- *Ticketi* koje izdaje udaljeni TGS daće informaciju krajnjem servisu da je klijent autentifikovan iz druge oblasti
- Poseban *inter-realm* ključ se može koristiti za svaki smer komunikacije

Kerberos - bezbednost

- Korisnikov TGT zahtev je slaba tačka protokola
  - Koristi se korisnikova tajna za šifrovanje podataka za autentifikaciju (podložno *offline dictionary* napadima)
  - Može se zaštititi pomoću Flexible Authentication Secure Tunneling (FAST) ili tzv. Kerberos *armoringa* (dostupno na Windows OS)
    - Štiti Kerberos podatke pre autentifikacije za KRB_AS_REQ koristeći LSK (*randomly generated logon session key*) iz TGT-a kao zajedničku tajnu za potpuno šifrovanje Kerberos poruka i potpisivanje svih mogućih Kerberos poruka o greškama
    - Zajednička tajna daje dodatni *salt* u Kerberos procesu autentifikacije
    - Ovo rezultira produženim vremenom obrade, ali ne menja veličinu tiketa usluge Kerberosa
    - Zajednička tajna pruža DC-ovima mogućnost da vrate greške o Kerberos autentikaciji, što zauzvrat štiti od spoofing-a, man-in-the-middle i drugih napada

Kerberos - bezbednost

- Ako se *ticket* ukrade, može se koristiti za lažno predstavljanje korisnika
  - Pass-the-ticket
- Ako se lozinka servisnog naloga ukrade, može da koristiti za lažno predstavljanje korisnika
  - Silver Ticket
- Ako se *krbtgt hash* ukrade, može se koristiti za lažno predstavljanje bilo koga u komunikaciji
  - Golden Ticket 

Poverenje u sisteme za autentifikaciju

- *Ken Thompson: Reflections on Trusting Trust*
  - Govor povodom dobijanja Tjuringove nagrade
  - Opisuje korišćenje samoizmenjujućih programa za postavljanje zadnjih vrata (*backdoor*)

1. **Korak 1**: izmeniti C kompajler tako da kada prevodi UNIX login komandu za određenu lozinku korisniku daje sva prava
   - Ovakav trojanski konj je lako uočljiv pregledom izvornog koda kompajlera
2. **Korak 2**: izmeniti C kompajler tako da, kada se prevodi C kompajler, ugradi prethodna izmena
   - Ovo se teže otkriva
   - Niko ne proverava prevedeni kod kompajlera!
3. **Korak 3**: ukloniti prethodne izmene u izvornom kodu C kompajlera i prevesti ga ponovo
   - ”Trojanske” izmene će postojati u prevedenom kodu kompajlera iako ih nema u izvornom kodu!

Poverenje u sisteme za autentifikaciju

- Zaključak: ne možemo verovati nijednom programu koji nismo sami napisali
  - Naročito ne programima koje proizvode kompanije koje zapošljavaju ljude kao što je Ken Thompson ☺
  - ”Trojanca” je moguće podmetnuti i u disasembler tako da se izmena ne može videti ni inspekcijom prevedenog koda
  - Jedino da sami pišemo disasembler, ali ko to još danas radi ☺