# Bezbednost Aplikacija
# Provera identiteta

Predmet: Bezbednost Aplikacija

Predavač: dr Dušan Stefanović

# VERIFIKACIJA IDENTITETA

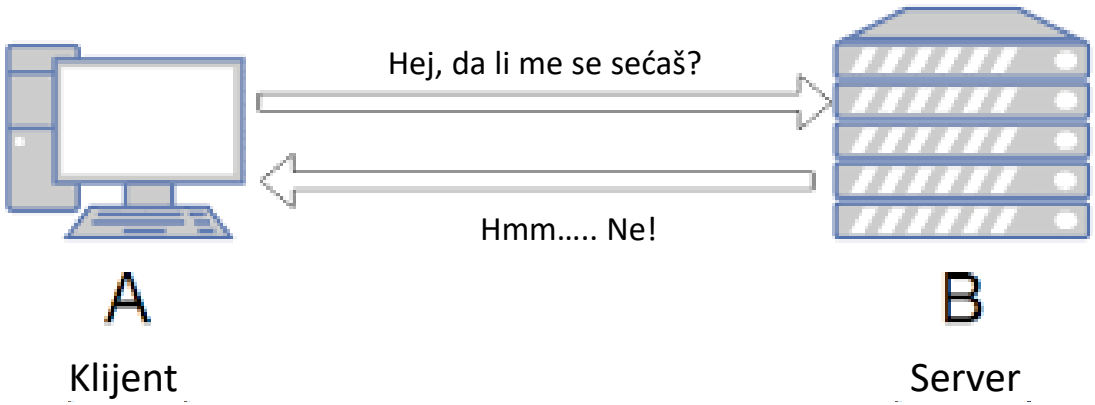
- Informacija na sajtu može da bude javna ili poverljiva tj. dostupna samo određenoj grupi
- Aplikacije zahtevaju od korisnika da dokaže identitet pre nego što se dozvoli pristup podacima
- Verifikacija identiteta zahteva od korisnika da obezbedi neki od sledećih dokaza identiteta
  - Nešto što korisnik zna (korisničko me i lozinka) – najčešći vid autentifikacije
  - Nešto što korisnik ima (specijalni kod poslat na telefon ili email korisnika) – bankarske aplikacije
  - Nešto što korisnik jeste (otisak prsta, prepoznavanje glasa – biometrijski dokaz)



# HTTP – STATELESS PROTOKOL

HTTP protokol je bez stanja i konekcije (stateless)

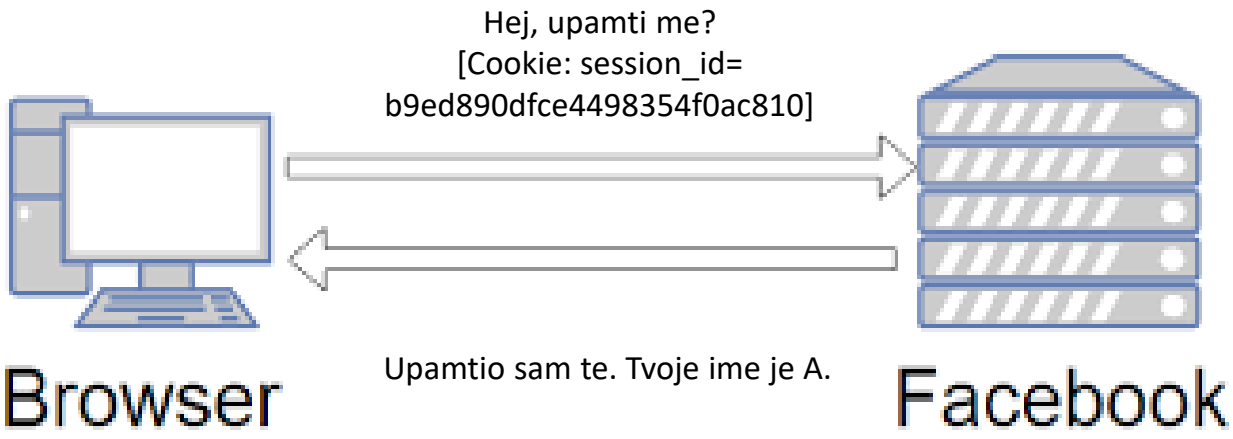
- Svaki zahtev koji server primi od istog klijenta, server tretira kao nepovezan sa prethodnim zahtevom
- Nakon prijave korisnika u aplikaciju, sledeći zahtev od istog korisnika server će tretirati kao prvi.
- Da li klijent treba da pošalje svoja ovlašćenja sa svakim zahtevom?



# HTTP – STATELESS PROTOKOL

Upravljanje sesijom

- Razvijen je veliki broj tehnika koje omogućavaju web aplikacijama da prate aktivnosti korisnika i za razdvajanje korisnika od drugih korisnika bez zahteva za prijavu za svaku akciju koju izvrše.



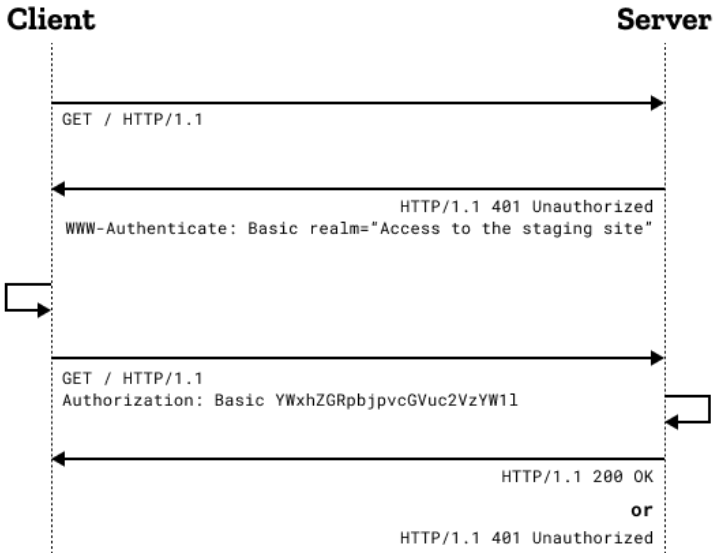
# PROVERA IDENTITETA PLATFORME

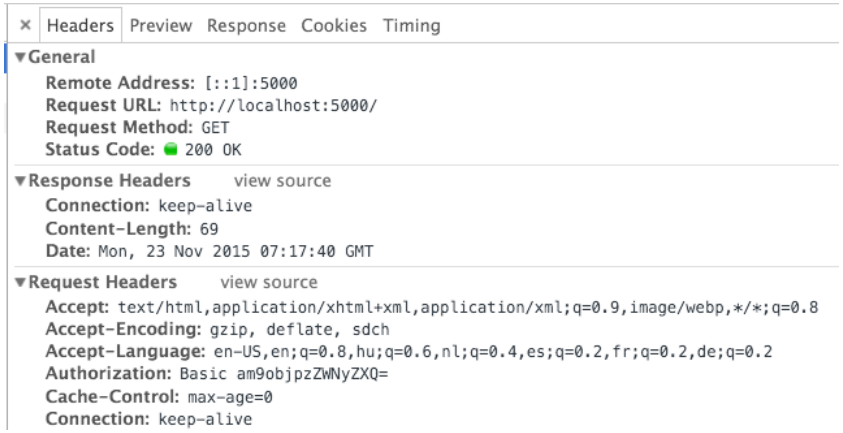
Kod provere identiteta platforme korisnici šalju svoja ovlašćenja u zaglavlje svakog zahteva koristeći promenjivu *Authorization*

Postoji nekoliko načina provere identiteta platforme:

## BASIC

- Korisničko ime i lozinka su poslati u zaglavlju AUTHORIZATION i šifrovani pomoću algoritma base64.
- Base64 nije kriptografski format tj. Lako može da bude dekodirana a samim tim da se pročitaju korisničko ime i lozinka



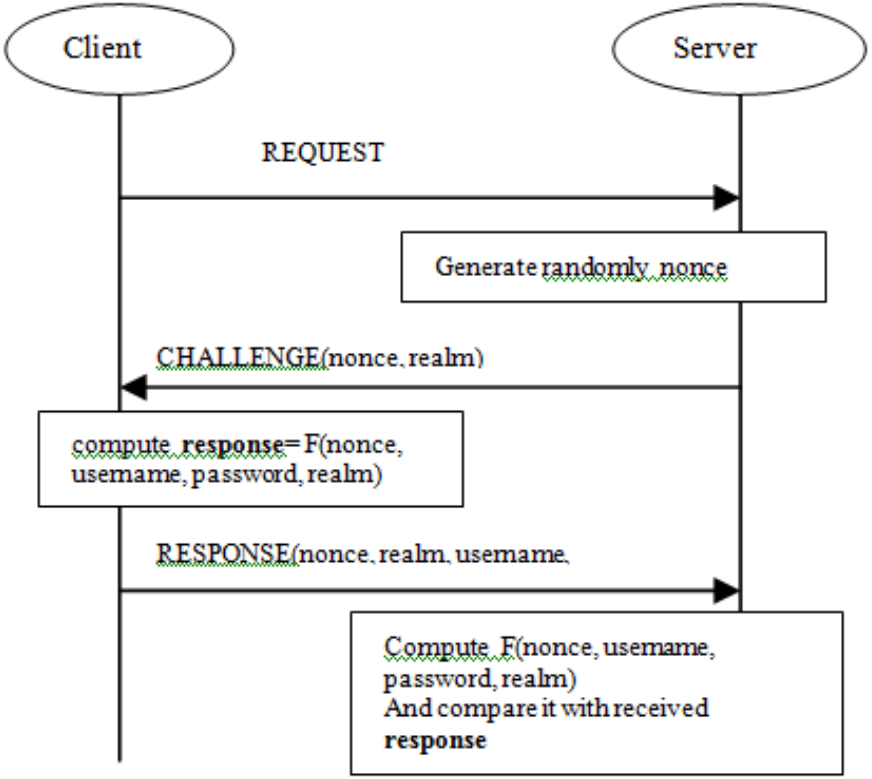



# PROVERA IDENTITETA PLATFORME

## DIGEST

Bezbednija od basic provere identiteta.

- Server klijentu šalje izazov (challange) slučajno izabran niz karaktera
- Klijent koristi dobijeni niz karaktera zajedno sa korisničkim imenom i lozinkom za izračunavanje md5 heša i šalje ga serveru na verifikaciju.



# PROVERA IDENTITETA PLATFORME

## KERBEROS

Koristi se Kerberos protokol za proveru identiteta na serveru

Podržan je od strane svih OS-a

Lozinka se ne šalje kroz mrežu

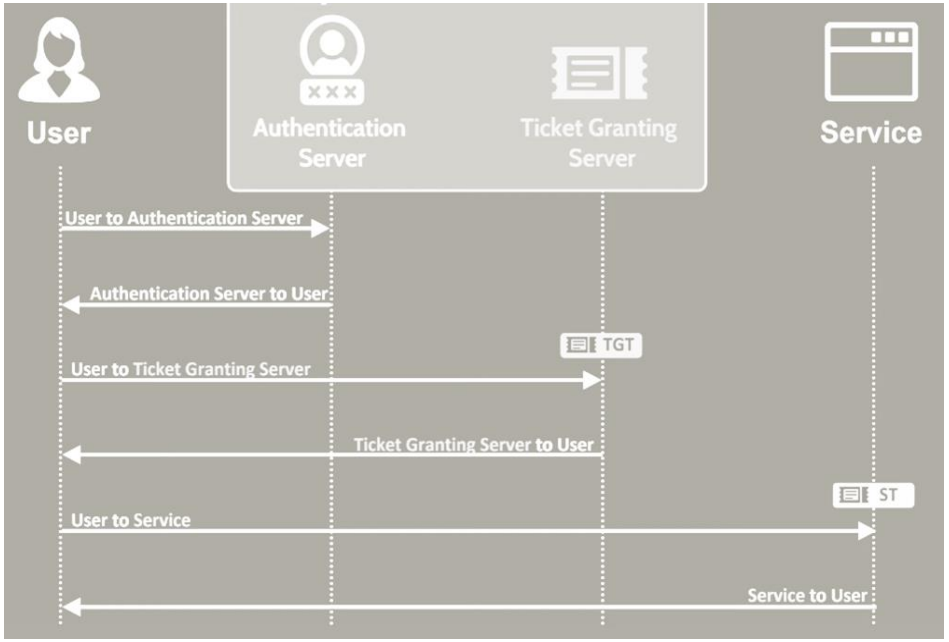
Protokol koristi Autentifikacioni server (AS) odvojeno od web servera tj. Servisa

Podržava samo simetričnu kripciju i iz tog razloga ima problem sa skaliranjem i distribucijom ključeva

Dve Kerberos verzije v4 i v5

Verzija 4 koristi DES algoritam

Verzija 5 podržava više algoritama (AES)




https://www.youtube.com/watch?v=5N242XcKAsM

# PROVERA IDENTITETA PLATFORME

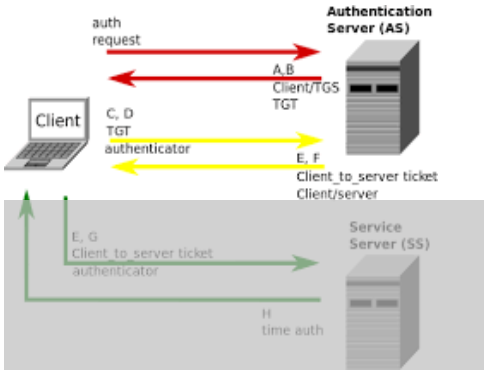
## KERBEROS

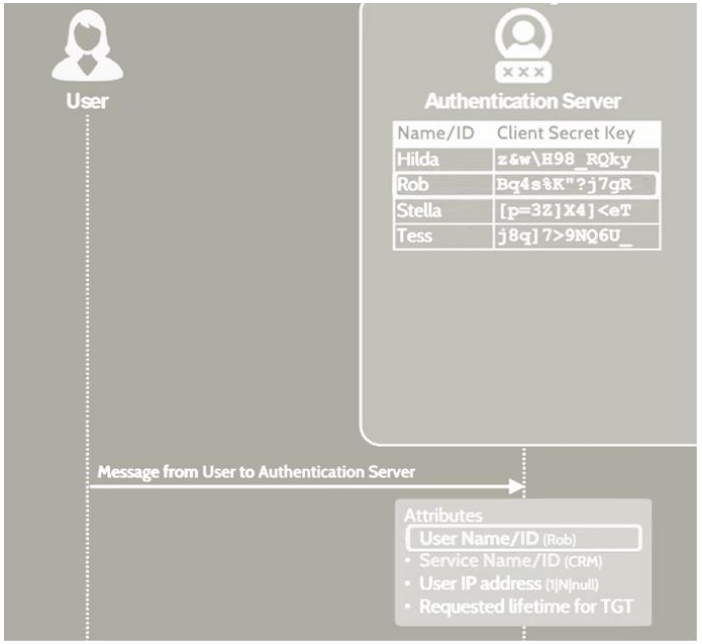
Prva poruka koju korisnik šalje KDC (Kerberos Distribution Centar) je nekriptovana i sadrži

Korisničko ime, ID servisa kome korisnik želi da pristupi, IP adresu korisnika i zahtev za neograničeno trajanje tiketa što Kerberos uglavnom odbija

Ova poruka se prosleđuje servisu za autentifikaciju

Servis za autentifikacuju na osnovu korisničkog imena pronalazi tajni ključ klijenta



# PROVERA IDENTITETA PLATFORME

## KERBEROS

Autentifikacioni server generiše dve poruke

Prva poruka sadrži ID Tiketa

Druga poruka je sam Tiket.

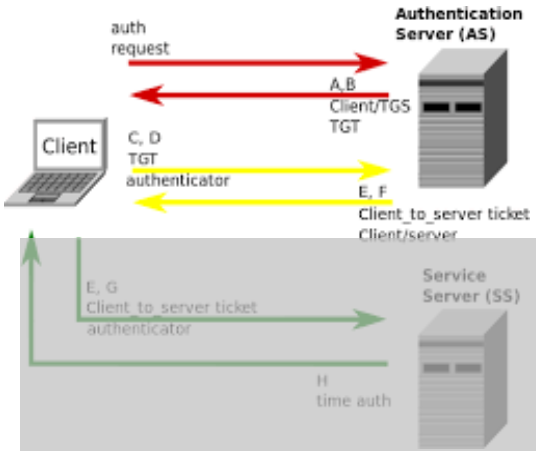
Obe poruke sadrže slučajno generisan simetričan ključ Tiket sesije

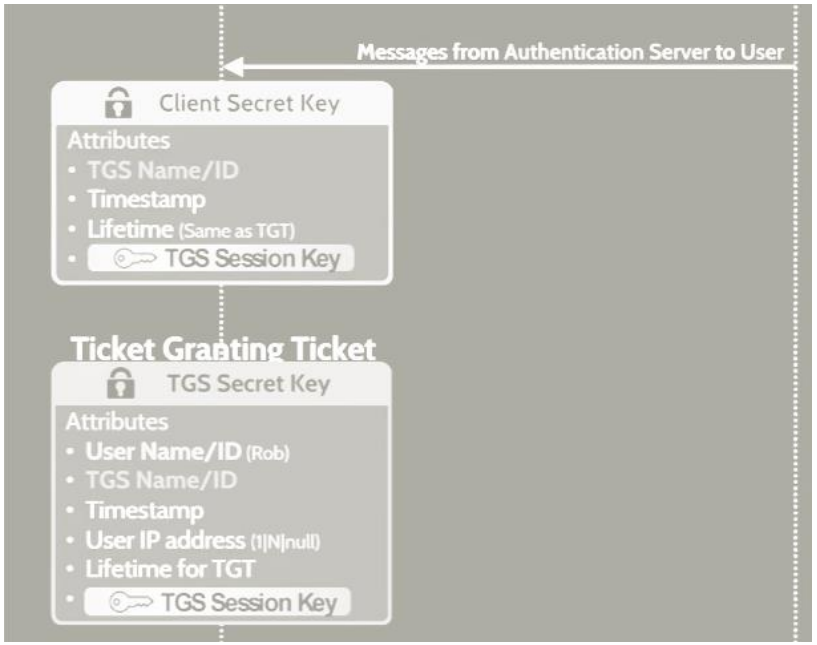
Prva poruka je kriptovana simetričnim ključem klijenta a druga poruka je kriptovana TGS tajnim ključem

Ako korisnik unese ispravnu lozinku moći će da dekriptuje prvu poruku i da pristupi ID Tiketu.

Drugu poruku ne može jer nema TGS tajni ključ

Klijent sada ima TGS ključ sesije



# PROVERA IDENTITETA PLATFORME

## KERBEROS

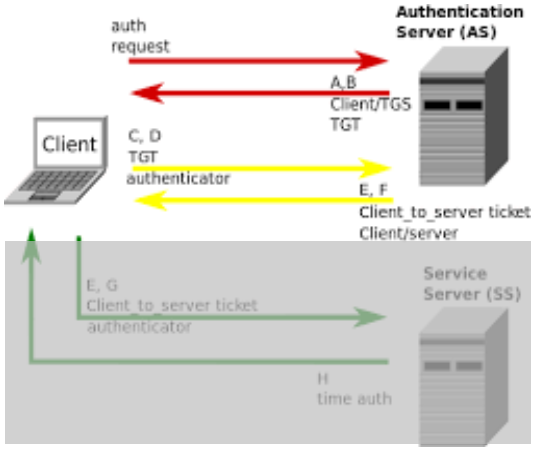
Klijent generiše dve nove poruke

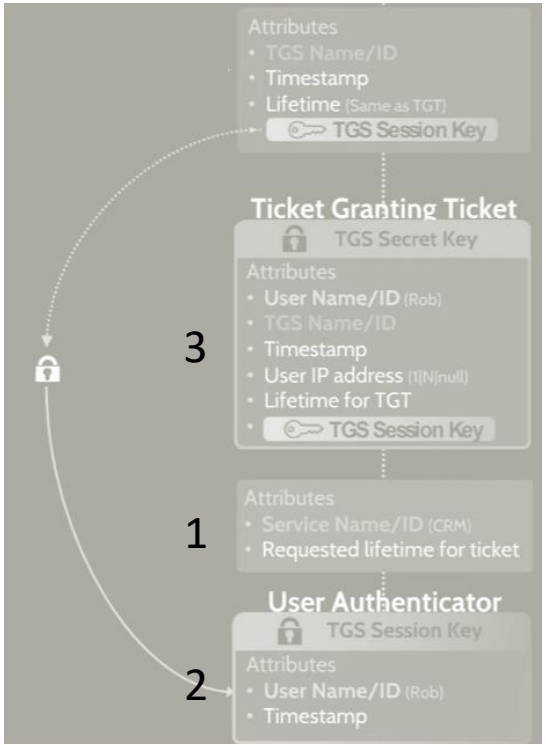
Prva poruka je običan tekst koji ukazuje resurs kome korisnik želi da pristupi

Druga poruka je autentifikator korisnika koja je kriptovana tajnim ključem TGS sesije.

Klijent šalje 3 poruke AS serveru

Tiket koji je dobio u prvom koraku i dve nove poruke koje je kreirao.



# PROVERA IDENTITETA PLATFORME

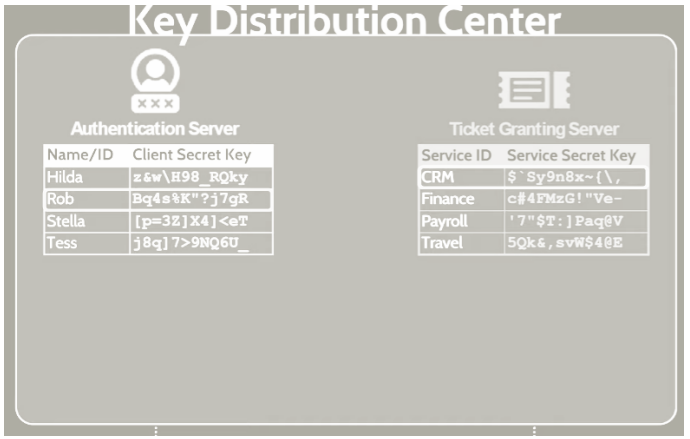
## KERBEROS

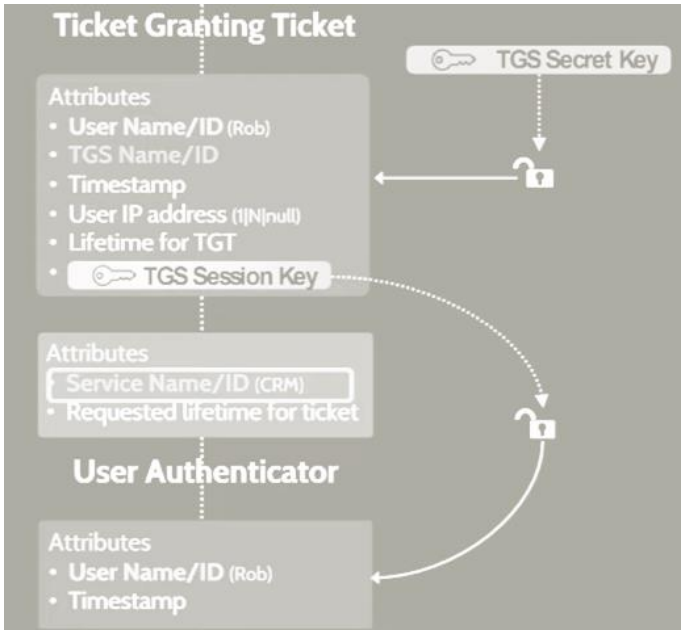
AS Server kada primi poruke kreće od service ID vrednosti.

Na osnovu te vrednosti pronalazi tajni ključ za taj servis

Otključava tiket na osnovu TGS ključa a zatim otključava drugu poruku na osnovu TGS ključa sesije

Pošto su sve poruke otključane, AS radi verifikaciju poruka tj. Da li se podaci podudaraju



# PROVERA IDENTITETA PLATFORME

## KERBEROS

AS server kreira dve poruke

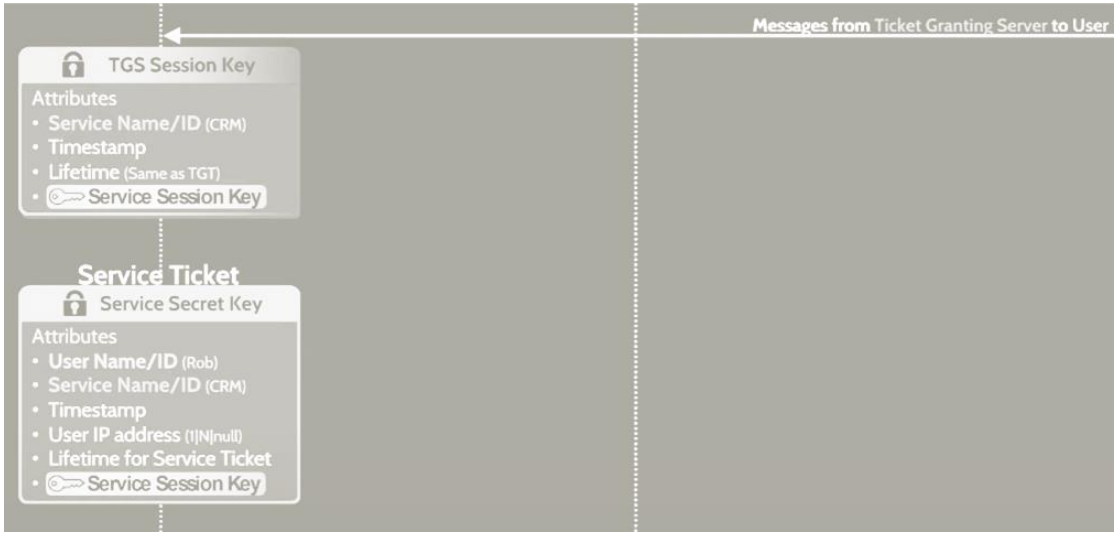
Prva sadrži ID servisa kome korisnik pristupa

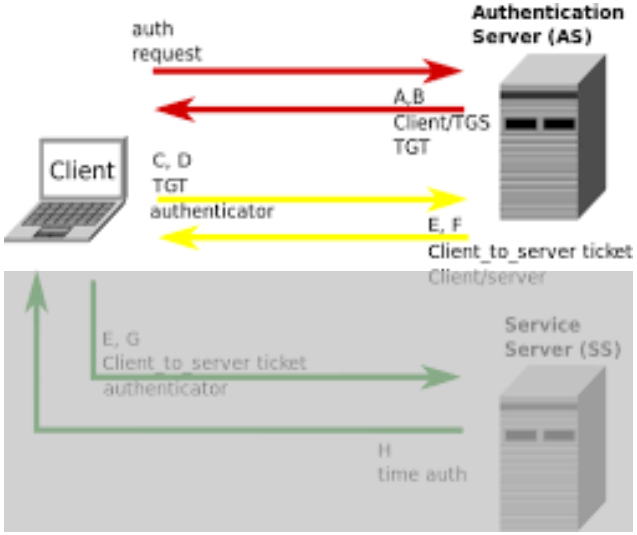
Druga poruka je Tiket servis

U obe poruke se nalazi slučajno generisan tajni ključ servisa sesije

Prva poruka je kriptovana TGS ključem sesije

Druga poruka je kriptovana tajnim ključem servisa



# PROVERA IDENTITETA PLATFORME

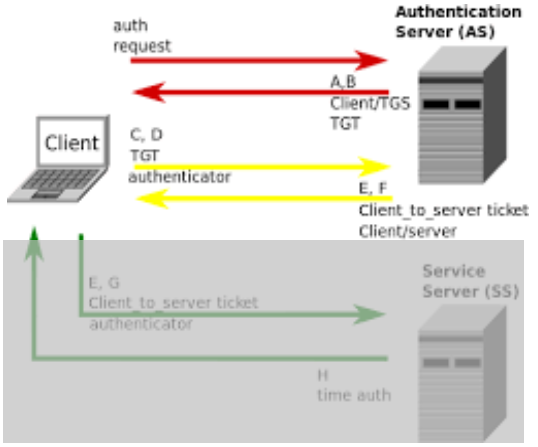
## KERBEROS

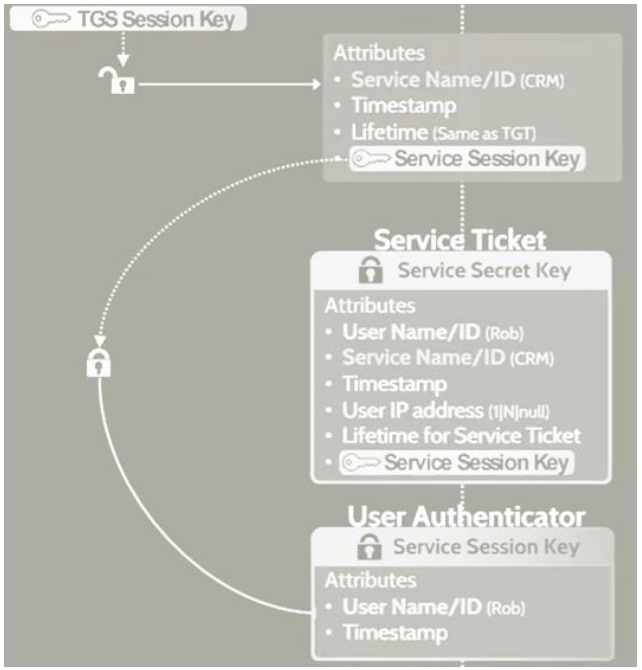
Klijent prima obe poruke

Poruku koja je bila zaključana TGS tajnim ključem dekriptuje jer je od autentifikacionog servisa u prethodnoj iteraciji dobio ključ.

Klijent ima kopiju ključa za servis (Service session key)

Servis tiket poruka ostaje zaključana

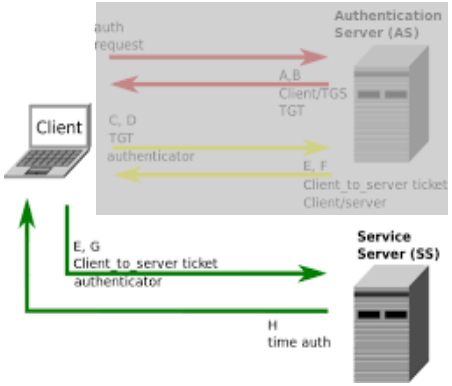


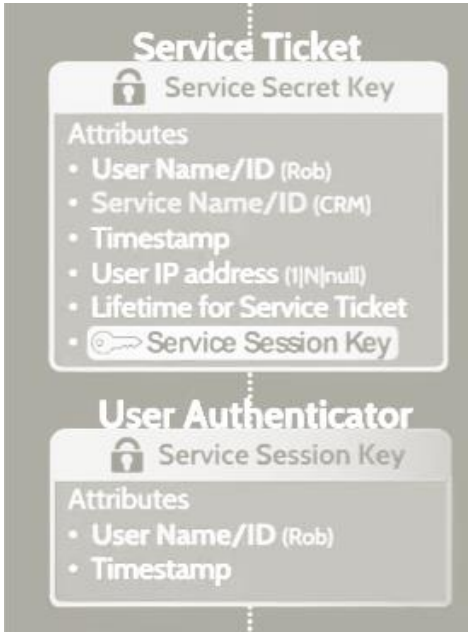



# PROVERA IDENTITETA PLATFORME

Klijent šalje dve poruke

Tiket za servis i korisničk autentifikator koji je zaključan ključem za servis sesije koji je klijent dobio u prethodnom koraku.



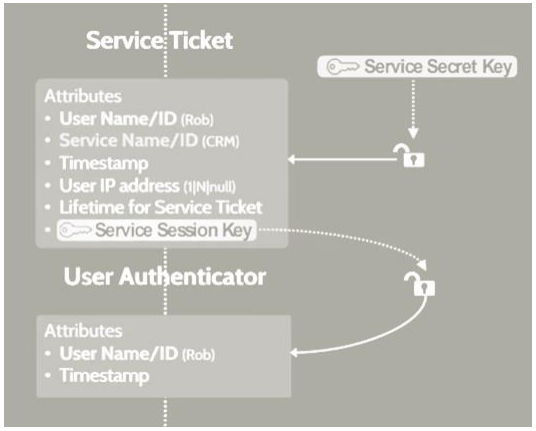




Server dekriptuje obe poruke

Prvo dekriptuje Servis tiket na osnovu svog ključa Service Secret ključa a zatim i drugu poruku na osnovu ključa iz prve poruke.

Nakon toga server će odraditi verifikaciju sadržaja u obe poruke

Ukoliko je sve u redu ubaciće u svoj keš parametre iz poruke user authenticator



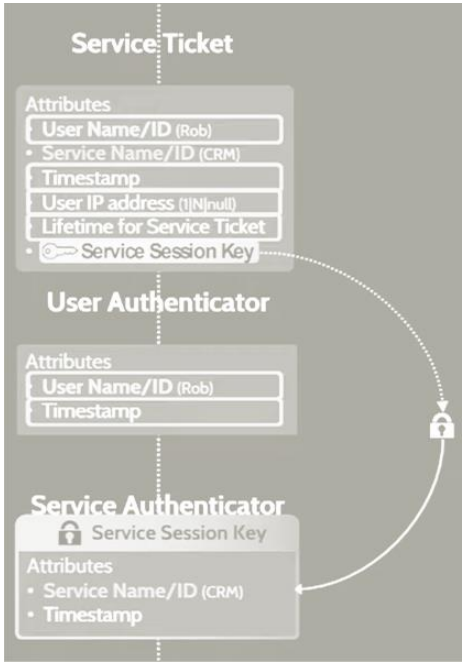
# PROVERA IDENTITETA PLATFORME

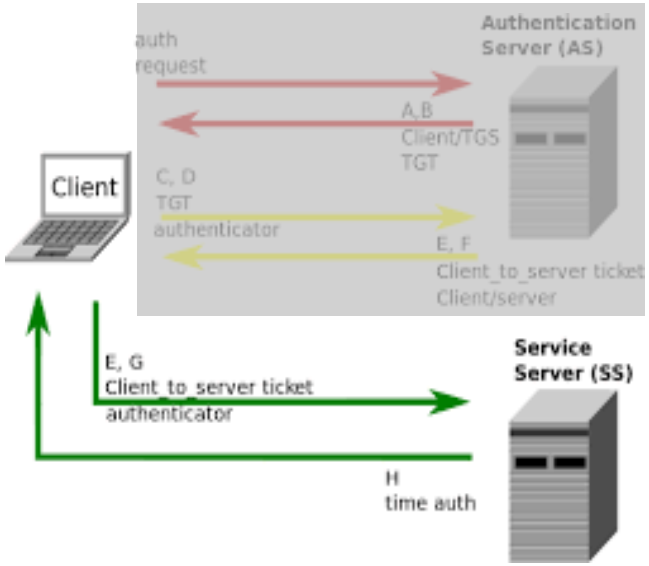
## KERBEROS

Konačno servis šalje korisniku Servis autheticator poruku koja je kriptovana ključem servisa sesije.

Klijent dekriptuje poruku ključem servisa sesije koju je dobio od AS servera u prethodnom koraku

Nakon verifikacije, korisnik u svom kešu sačuvaće servis tiket za kasnije korišćenje



# PROVERA IDENTITETA PLATFORME

## NEDOSTACI

KERBEROS šema se smatra bezbednom ali i DIGEST i BASIC provere identiteta mogu da se upotrebe pomoću TLS-a sa malom verovatnoćom da se iz presretnute komunikacije ukradu ovlašćenja

Nedostatak je što nema opcije za odjavu ili vreme isteka sesije jer se automatski obnavlja kad sesija istekne, napadač ako dobije pristup mašini korisnika tj. nalogu imaće odmah pristup i aplikaciji

Nije pogodna za javne aplikacije jer zahteva više administrativnog posla od najpopularnije provere zasnovane na proveri identiteta na obrascu

# PROVERA IDENTITETA ZASNOVANA NA FORMAMA

## HTML FORMA

Najčešći vid provere identiteta u aplikacijama

Sadrži polja za korisničko ime i lozinku

Implementacija u potpunosti zavisi od aplikacije


- Korisničko ime i lozinka se šalju na server kao tekst osim ukoliko se ne koristi mehanizam kripcije na strani klijenta
- Server proverava kredencijale korisnika u svojoj bazi
- Ukoliko je verifikacija uspešna, server preusmerava korisnika na njegovu početnu stranu i šalje identifikator sesije (kolačić) da korisnik nebi ponovo slao svoje kredencijale za verifikaciju
- Klijent prima odgovor, skladišti identifikator sesije i preusmerava korisnika na početnu stranu

# PROVERA IDENTITETA ZASNOVANA NA DVA FAKTORA

## 2FA (DVOFAKTORNA AUTENTIFIKACIJA)

Zahteva da se obezbede sledeći identifikatori:

- nešto što znamo (korisničko ime i lozinka)
- nešto što imamo (sms kod ili email kod)
- Nešto što jesmo (biometrijski dokaz)


Svaki od identifikatora se naziva faktor a provera identiteta zasnovana na više faktora je MFA.

2FA u većini aplikacija podrazumeva

Faktor1: korisničko ime i lozinku

Faktor2: jednokratna lozinka (OTP) koju je generisao nasumično server na email adresu ili sms poruku

Većina aplikacija za bankarstvo implementira više faktornu verifikaciju.

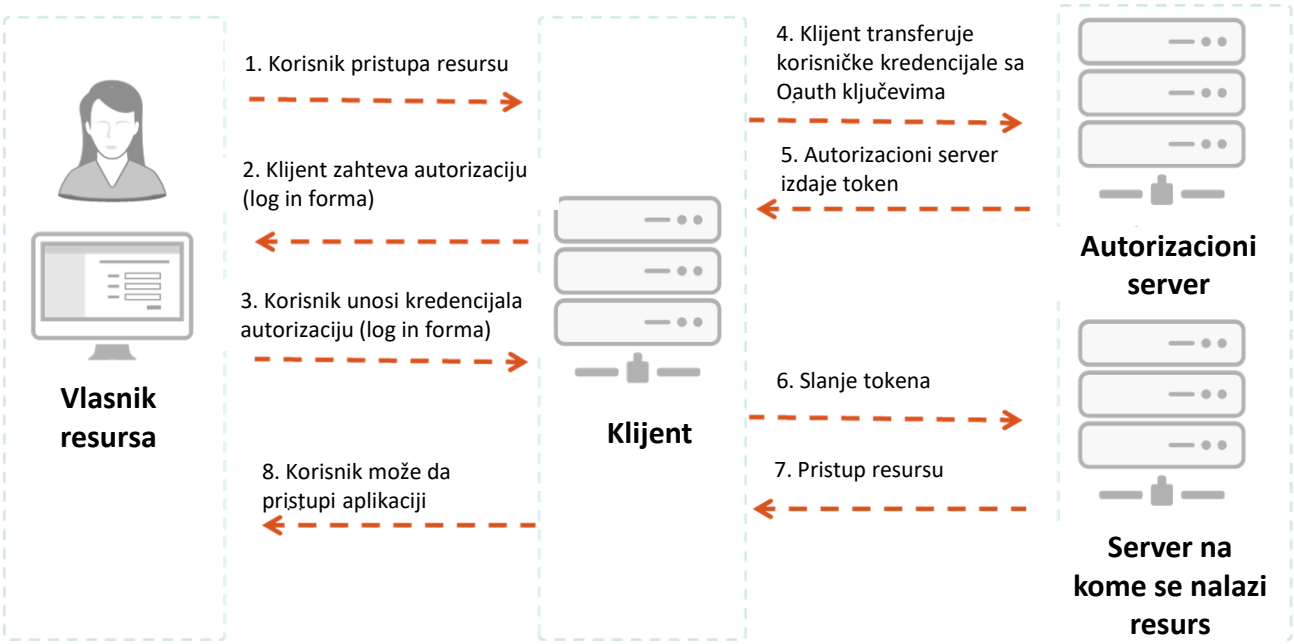
# PROVERA IDENTITETA

## OAuth

Standard za prosleđivanje pristupa (autorizacija između servisa)

Deljenje ovlašćenja nalozima aplikacije od strane drugih aplikacija (npr. Facebook korisnik omogući drugim aplikacijama da pristupe njegovom nalogu)

Provajderi servisa (Facebook) dele specijalne tokene pristupa koji omogućavaju trećim aplikacijama da preuzmu određene informacije sa naloga tog korisnika



# OAuth

Vlasnik resursa je bilo ko ko treba da odobri pristup resursu

Klijent je aplikacija koja u ime i sa odobrenjem vlasnika resursa postavlja zahtev za pristup tom resursu.

Autorizacioni server je server koji nakon autorizacije svih strana u pristupu izdaje pristupni token klijentu

Server na kome se nalazi resurs je server koji prihvata token i da odgovori isporukom resursa