Akademija tehničko-vaspitačkih strukovnih studija

INTERNET TEHNOLOGIJE

Prof. dr Zoran Veličković, dipl. inž. el.

2022/23.

Prof. dr Zoran Veličković, dipl. inž. el.

Internet tehnologije

Bezbednost na Web-u

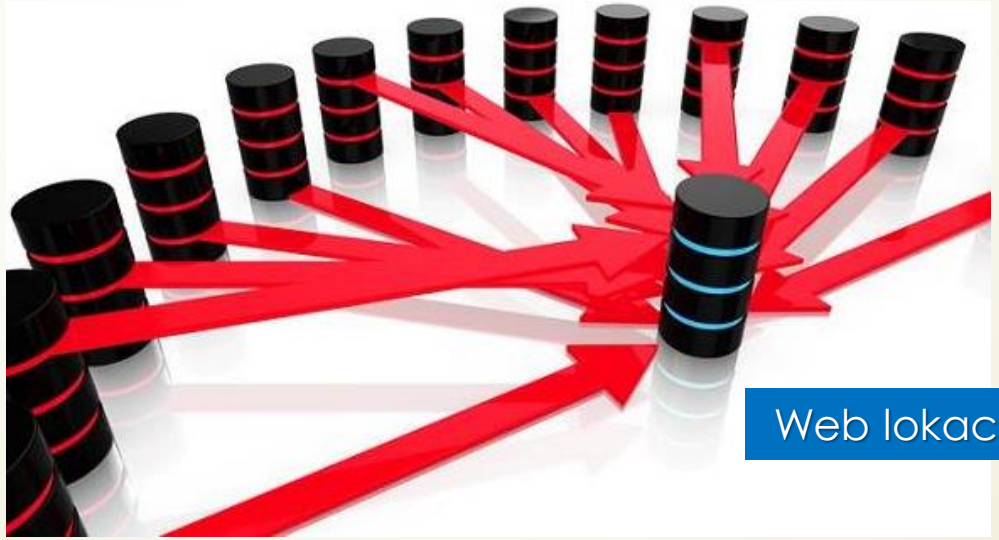
(13)

Sadržaj

Napadi na Web lokacije

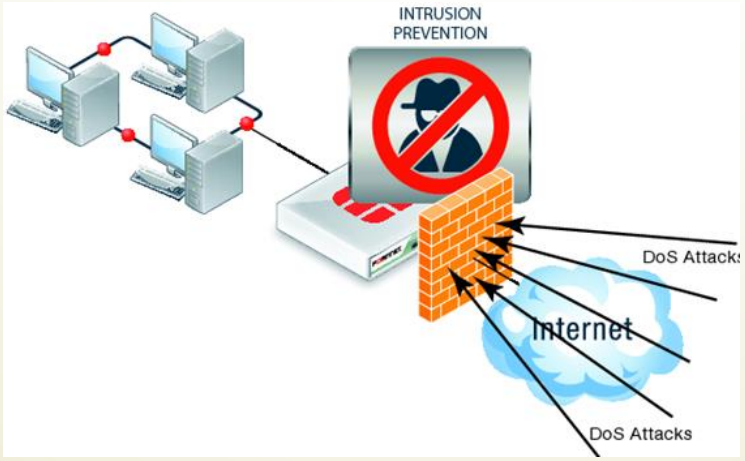
- **WORLD WIDE WEB** je postao **NAJMOĆNIJA PLATFORMA** za ISPORUKU APLIKACIJA.
- Ovde se i krije velika mogućnost za **ZLONAMERNE NAPADE** na Web aplikacije.
- Zaštita samih **WEB LOKACIJA** kao i zaštita **KORISNIČKIH TRANSAKCIJA** na bezbednim stranama je postala **NEOPHODNA** na današnjem Internetu.

Mnogobrojni izvori napada na Web lokaciju



Tipovi napada na Web lokacije

- **NAPADI** na Web lokacije se mogu izvršiti na sledeće načine:
  - **PRESRETANJEM** i **MENJANJEM** HTTP poruka, koje server razmenjuje sa čitačima korisnika.
  - **PRISTUPANJEM** datotekama koje sadrže važne informacije o **PRAVIMA PRISTUPA** sistemu ili o bankovnim kreditnim karticama korisnika.
  - **POKRETANJEM NA HILJADE ZAHTEVA** serveru radi trošenja njegovih resursa čime se sprečava pristup korisnika sistemu.
  - **INFICIRANJE** računara **VIRUSOM**.
  - **ZAUSTAVLJANJE SERVERSKIH SKRIPTOVA** radi **ONEMOGUĆAVANJA** korisničkog pristupa serveru.



Sajber pretnje uživo

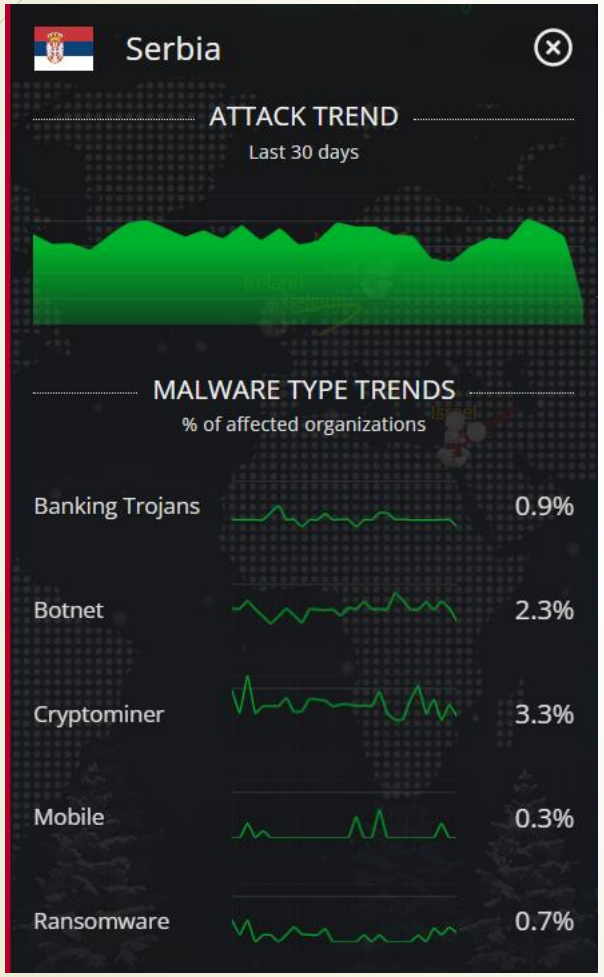
Statistika napada po državama

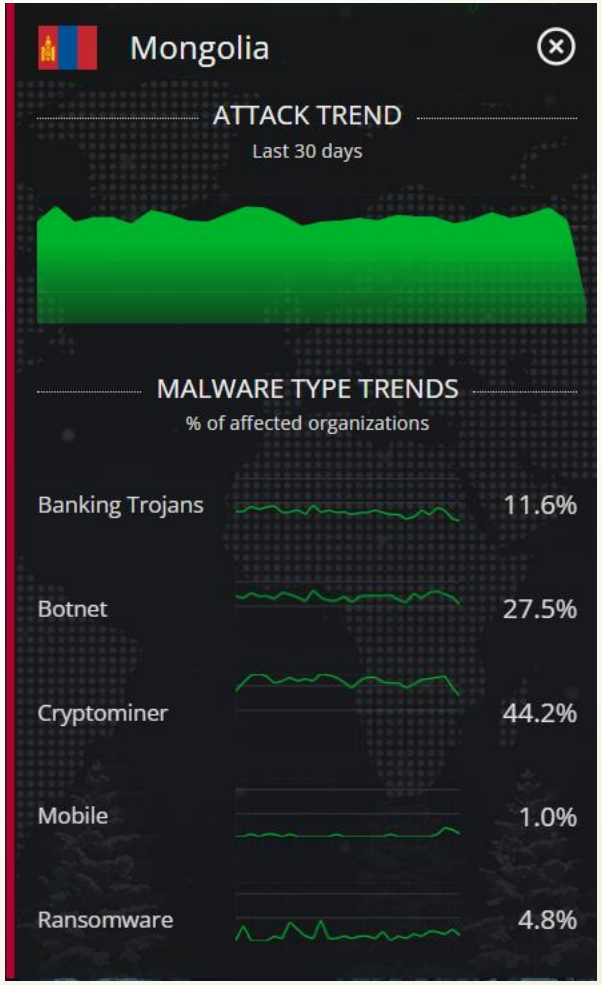
Države

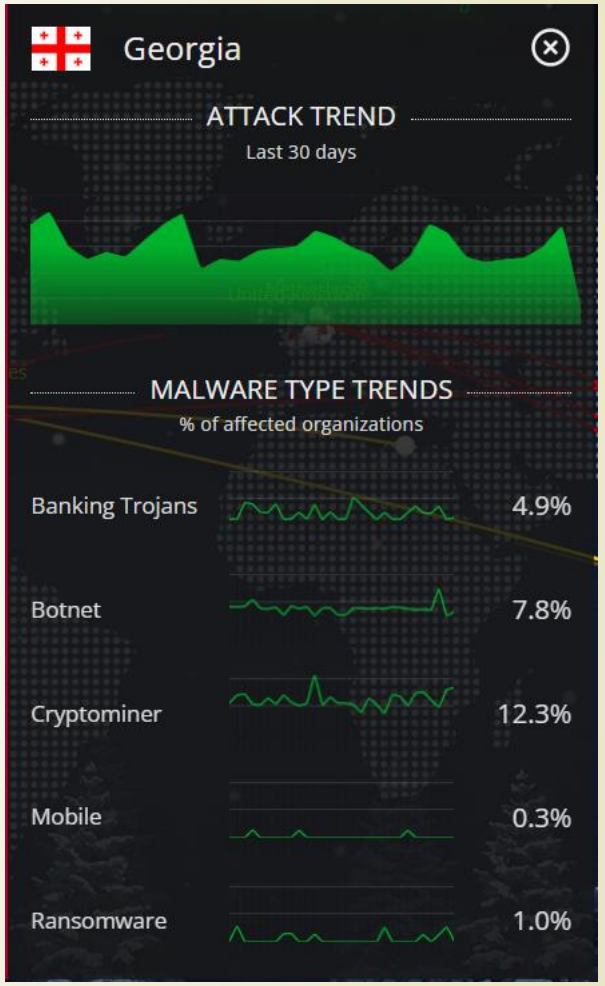
Poslednjih 30 dana

% napadnutih organizacija

Tipovi napada







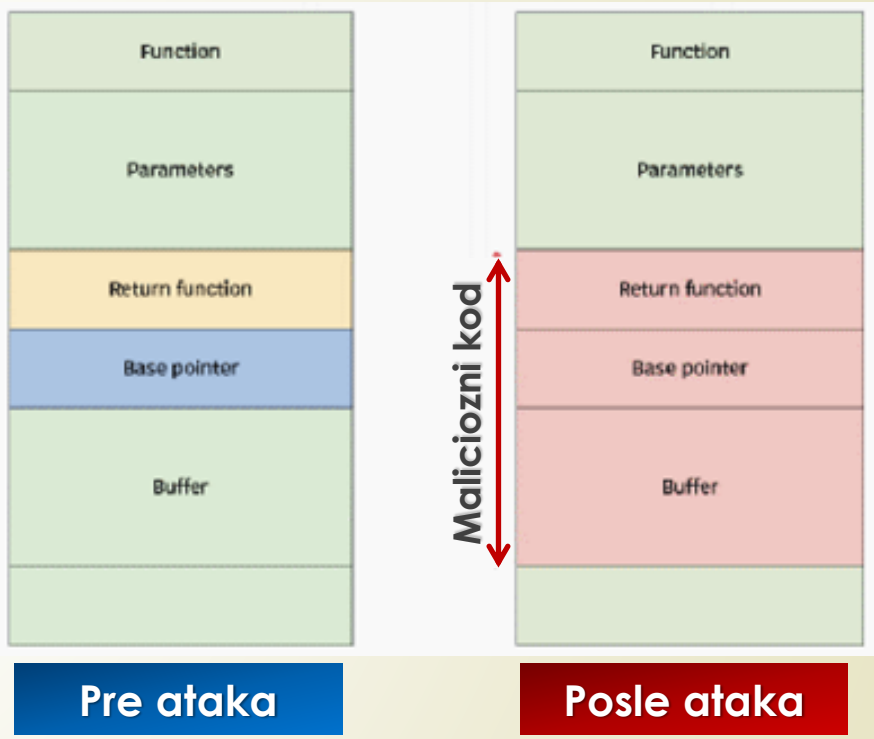
https://threatmap.checkpoint.com/

Opšte bezbednosne pretnje na Web-u

- Karakteristične **BEZBEDNOSTNE PRETNJE** koje su vezane za Web (u originalnim engleskim nazivima) su:
  - *Buffer overflow*
  - *Code injection*
  - *Cross-site scripting (XSS)*
  - *Missing or incorrect encryption*
  - *Operating system command injection*
  - *Parameter manipulation*
  - *Remote code inclusion*
  - *Session hijacking*
  - *SQL injection*
  - *File uploads*
  - *Hardcoded authentication*
  - *Hidden or restricted file/directory discovery*
  - *Missing or incorrect authentication*

Prepunjavanje bafera i ubacivanje koda

- Napadač šalje **VELIKU KOLIČINU PODATAKA** u **ULAZNI BAFER** da bi **PREPUNIO BAFER APLIKACIJE** ili **izlazni bafer** (engl. *Buffer overflow*).
- Kao posledica ovog napada - prepunjavanja bafera, **MEMORIJA IZVAN BAFERA POSTAJE KORUMPIRANA** i izvršni kod koji se nalazi u njoj više nije upotrebljiv.
- Najbolji način za prevazilaženje ovog problema je, **PROVERA OPSEGA** I **VELIČINE PODATAKA NA ULAZU** odnosno, izlazu iz aplikacije.
- Cil ovog napada je da se **DODATI KOD TRETIRA** kao deo **ORIGINALNE STRANICE** iako može da sadrži **ZLONAMERNI KOD** koji će izazvati probleme u radu aplikacije.
- **VALIDACIJA UNETIH PODATAKA** je najznačajniji aspekt bezbednosti Web aplikacije.



Pogrešni tip podataka

- Najčešća vrsta napada uključuje **POGREŠNE TIPOVE PODATAKA** ili **POGREŠNU VELIČINU PODATAKA** koji mogu sadržavati **SPECIJALNE KARAKTERE** kao što su „**escape**" sekvence ili **binarni** kod.
- Pored toga što se nekorektni podaci UNOSE u baze podataka, **NEISPRAVAN FORMAT** može prouzrokovati **BRISANJE PODATAKA** iz baze.
- Korišćenje **NEKOREKTNIH PODATAKA** u drugim skriptovima može izazvati **NEOČEKIVANO PONAŠANJE** Web aplikacije.
- Ovakvo ponašanje Web aplikacije mogu **ZLOUPOTREBITI NAPADAČI** i onesposobiti sistem.

Problem sa metakarakterima

- Ako se metakarkteri:

  ! $ ^ & * ( ) ~ [ ] \ | { } ' " ; < > ? - `

  pojave u **INPUT POLJIMA**, a koriste se nekorektno, mogu izazvati **RAZLIČITE PROBLEME**.

- Kod formiranju upita **DBMS**-u karakteri „**' " ; **" imaju **SPECIFIČNO ZNAČENJE**, tako da njihovo nepravilno korišćenje može izazvati **BEZBEDNOSNE PROBLEME**.

- U zavisnosti od toga KAKO je upit strukturiran, ovi karakteri se mogu koristiti za **INSERTOVANJE DODATNIH SQL UPITA** i eventualno izvršiti dodatne - bezbednosno kritične upite.

- Karakteri **UNICOD**-a, kao i **neprintajući ASCII** karakteri, takođe mogu izazvati bezbedonosne probleme ako se nađu u **upitima DBMS-u**.

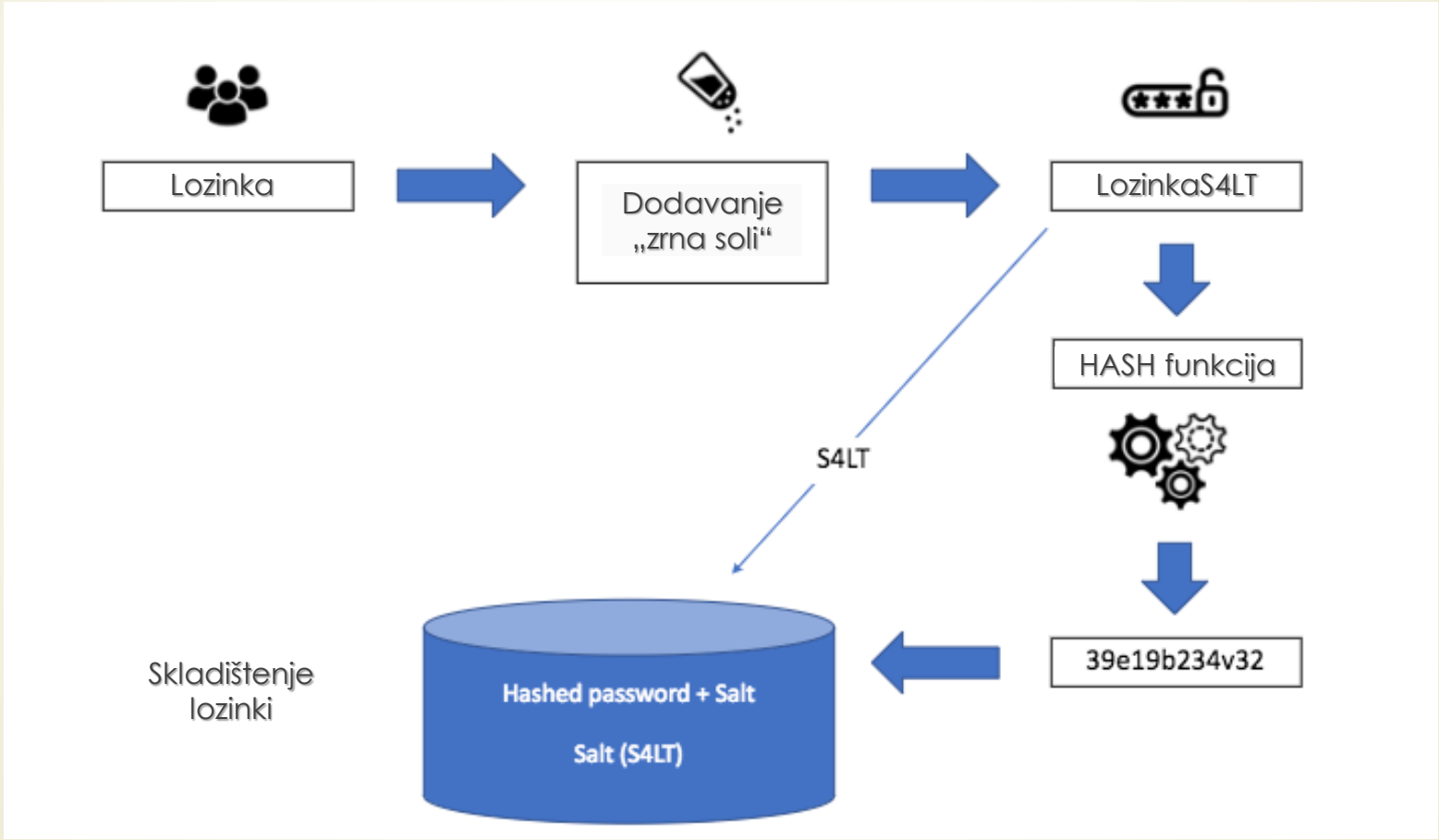
Pogrešan ulazni tip podataka

- **POGREŠAN TIP** i/ili **FORMAT** podataka u **ULAZNIM POLJIMA** rezultira neadekvatnom vrednošću, pa kao takav, može izazvati nepoželjne efekte u Web aplikacijama.
- Već smo pokazali da je to jedan od razloka zbog koga je HTML 5 uveo **NOVE TIPOVE** ulaznih polja.
- **ULAZNE VREDNOSTI** koje su **PREVELIKE** mogu izazvati **PREKORAČENJE ULAZNOG BAFERA** i ponovo izazvati nepoželjno ponašanje Web aplikacije:
  - Ako nije ograničea **VELIČINA TEKSTA/FAJLA** koji se šalje;
  - Ako nisu ograničene **DUŽINE POLJA** za rad sa bazom podataka.
- **SKRIVENI INTERFEJSI**, kao što je recimo **ADMINISTRATIVNI INTERFEJS**, napadač može iskoristiti za maliciozni napad.
- Napadači mogu ubacivati **NEŽELJENE KOMANDE** u **SQL UPITE**.
- Najčešće se odnose na **POLJA ZA PRETRAŽIVAJE** kada se uneti string **KOPIRA** u SQL upit.

Čuvanje lozinki

- **KORISNIČKA IMENA** i **LOZINKE** zapamćene su u **IZVORNOM OBLIKU** – običan ASCII tekst, što može biti **PREDMET NAPADA** zlonamernih korisnika.
- Zbog bezbednosti, dobro je umesto ASCII teksta lozinki čuvati samo njihove **HEŠ VREDNOSTI**.
- Kasnije se u verifikaciji korisnika porede samo **HEŠ VREDNOSTI** lozinki bez poređenja njihovih originalnih vrednosti.
- Dobra osobina ovih algoritama je što na unetu lozinku dodaju tzv. "**ZRNO SOLI**" – nasumično generisani podatak koji se **DODAJE LOZINKI** pre poziva metode za hešovanje lozinki.
- Korišćenjem tehnike „zrna soli" **IZVORNI OBLIK LOZINKE** više **NIJE POZNAT**, za pristup podacima treba znati **SAMO HEŠ VREDNOST LOZINKE**.

Zrno soli – generisanje i čuvanje lozinki



Bezbednost skriptova

- **MREŽNE BARIJERE** omogućavaju da se **KONTROLIŠE DOTOK PORUKA** u lokalnu mrežu čime se **SPREČAVA** da poruke vezane za **ODREĐNE PRIKLJUČKE** ne mogu da uđu u lokalnu mrežu.
- Takođe, treba sprečiti **PRISTUP SKRIPTOVIMA** ili **CGI** programima jer **ONI MOGU PRISTUPITI PODACIMA** smeštenim na **DISKU SERVERA**.
- Haker može da (*zlo*)upotrebi skript tako što ga izvrši koristeći vrednosti koje mu je dodelio **IZVAN OBRASCA** ili ako haker **ZAMENI ORIGINALNI SKRIPT** svojim skriptom.
- **PREOPTEREĆENJE BAFERA** se pojavljuje kada korisnik pošalje **VIŠE PODATAKA** nego što **SKRIPT OČEKUJE** - može da prihvati.
- Ovom prilikom neki skript jezici izazivaju **KVAR SKRIPT PROCESORA** i mogu omogućiti hakeru **PRISTUP SERVERU** i **DATOTEKAMA** koje on poseduje.

Ubacivanje SQL koda (1)

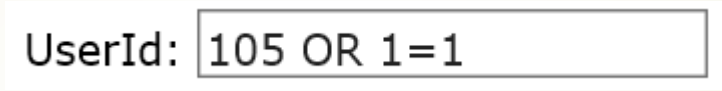
- **SQL INJECTION** je tehnika insertovanja **JS**, **MSQL** (ili nekih drugih) programskih kodova koji mogu uništiti bazu podataka!
- Zlonamerni programski kod se postavlja u **SQL UPITE** posredstvom **KORISNIČKOG UNOSA** sa Web stranice.
- **SQL INJECTION** je jedna od **NAJČEŠĆIH TEHNIKA HAKOVANJA** Weba.
- Najčešća manifestacija ove tehnike je kada se od korisnika traži **UNOS PODATAKA** (`input` polje u obrascima) kao što su `korisničko_ime/lozinka`, a korisnik umesto traženih podataka unosi **SQL IZRAZE** koji će se zlonamerno izvršiti.
- PRIMER pretraživanja:

```
txtUserId = getRequestString("UserId");
txtSQL = "SELECT * FROM Users WHERE UserId = " + txtUserId;
```

Dodavanje promenljive (**txtUserId**) za selekciju stringa

Ubacivanje SQL koda (2)

- Zlonamerni korisnik može da unese "pogrešan - zlonameran" unos, nešto slično kao u primeru:



- Pogledajte kako sada izgleda SQL upit:

Tautologija!

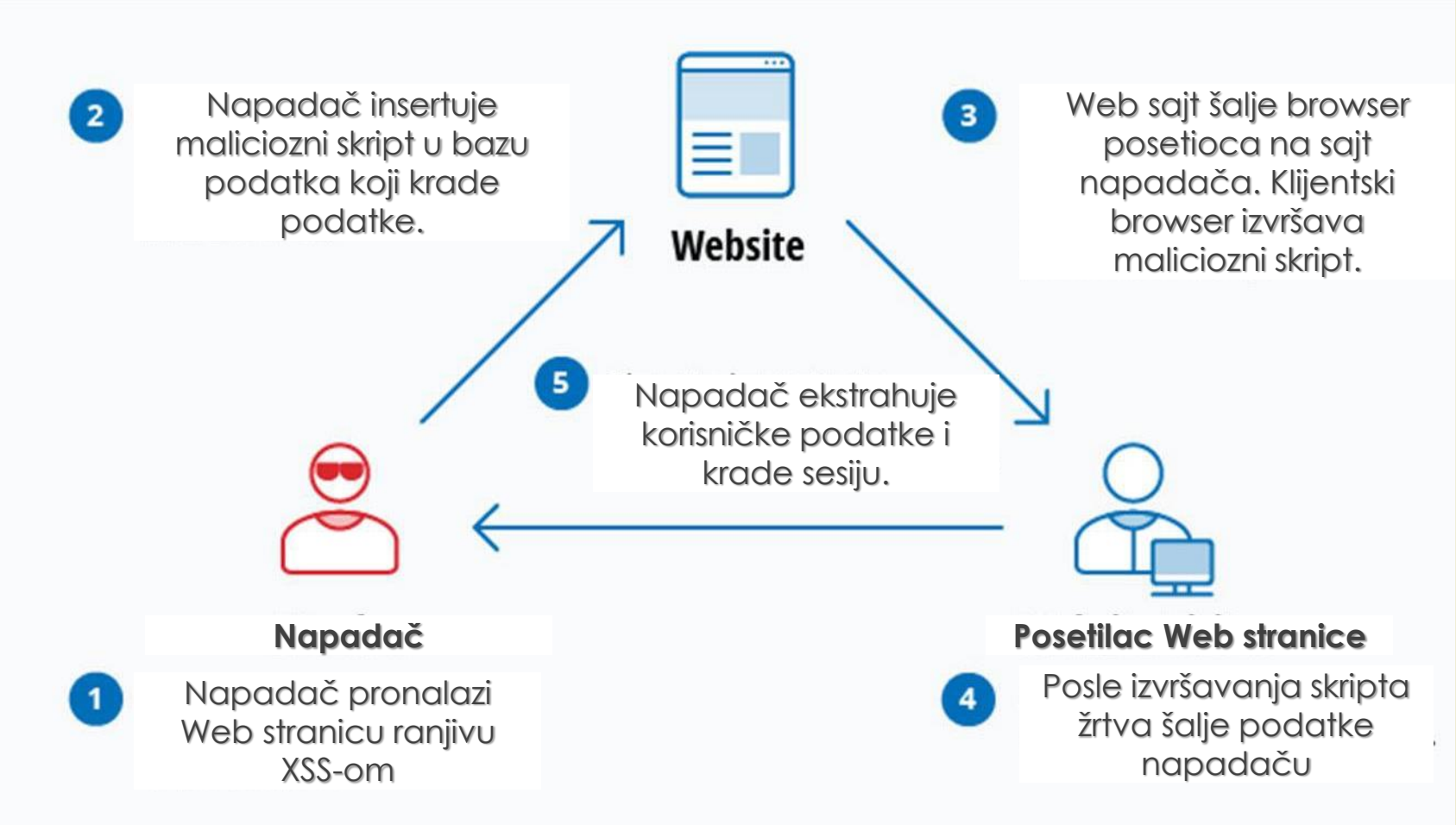
```
SELECT * FROM Users WHERE UserId = 105 OR 1 = 1;
```

- Dobijeni SQL je **VAŽEĆI** i vratiće **SVE REDOVE** iz tabele "`Users`", jer je SQL iskaz uvek **TRUE**!
- Da li gore navedeni primer **IZGLEDA OPASNO**? Šta ako tabela "`Korisnici`" sadrži korisnička imena i lozinke?
- Haker može izlistati **SVA KORISNIČKA IMENA** i **LOZINKE** iz u baze podataka, jednostavnim ubacivanjem stringa **`105 or 1 = 1`** u polje za unos!

Cross-Site Scripting

- Kod „**CROSS-SITE SCRIPTING**“-a se za razliku od insertovanje SQL-a koda, insertuje ZLONAMERNI **HTML** ILI **JAVASCRIPT** kod.
- Ovaj zlonamerni kod pokušava da **ZADOBIJE POVERENJE** korisnika na Web stranici, **PREVAROM** (korisnika ili njegovog pretraživača) tako što šalje podatke **NEKOJ DRUGOJ** – **NEBEZBEDNOJ** Web lokaciji.
- Napadač bi mogao da insertuje HTML koji prikazuje **NEBEZBEDNI LINK**, tako da se **SVE INFORMACIJE** zapravo dostavljaju **NEBEZBEDNOJ LOKACIJI**.
- **ZABRANOM IZVRŠAVANJA** JavaScript-a se mogu sprečiti ovi napadi, samo je pitanje da li su korisnici spremni da se odreknu benefita koje on pruža.

Arhitektura **Cross-Site Scripting** napada

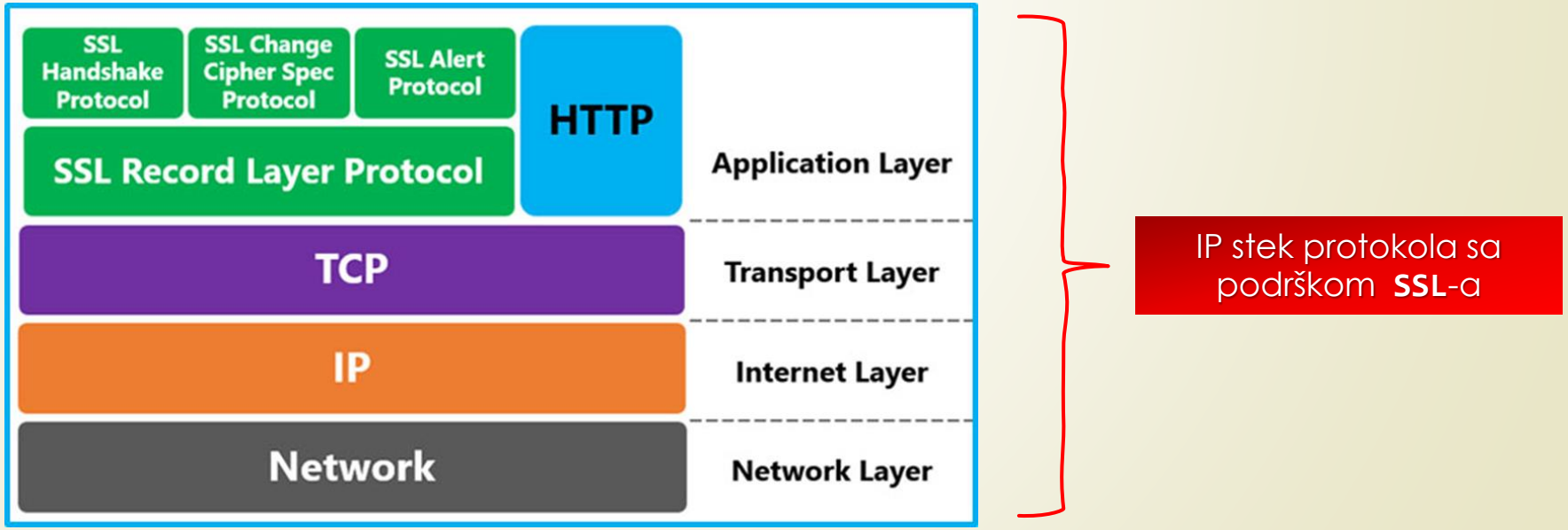


Hakerske tehnike napada

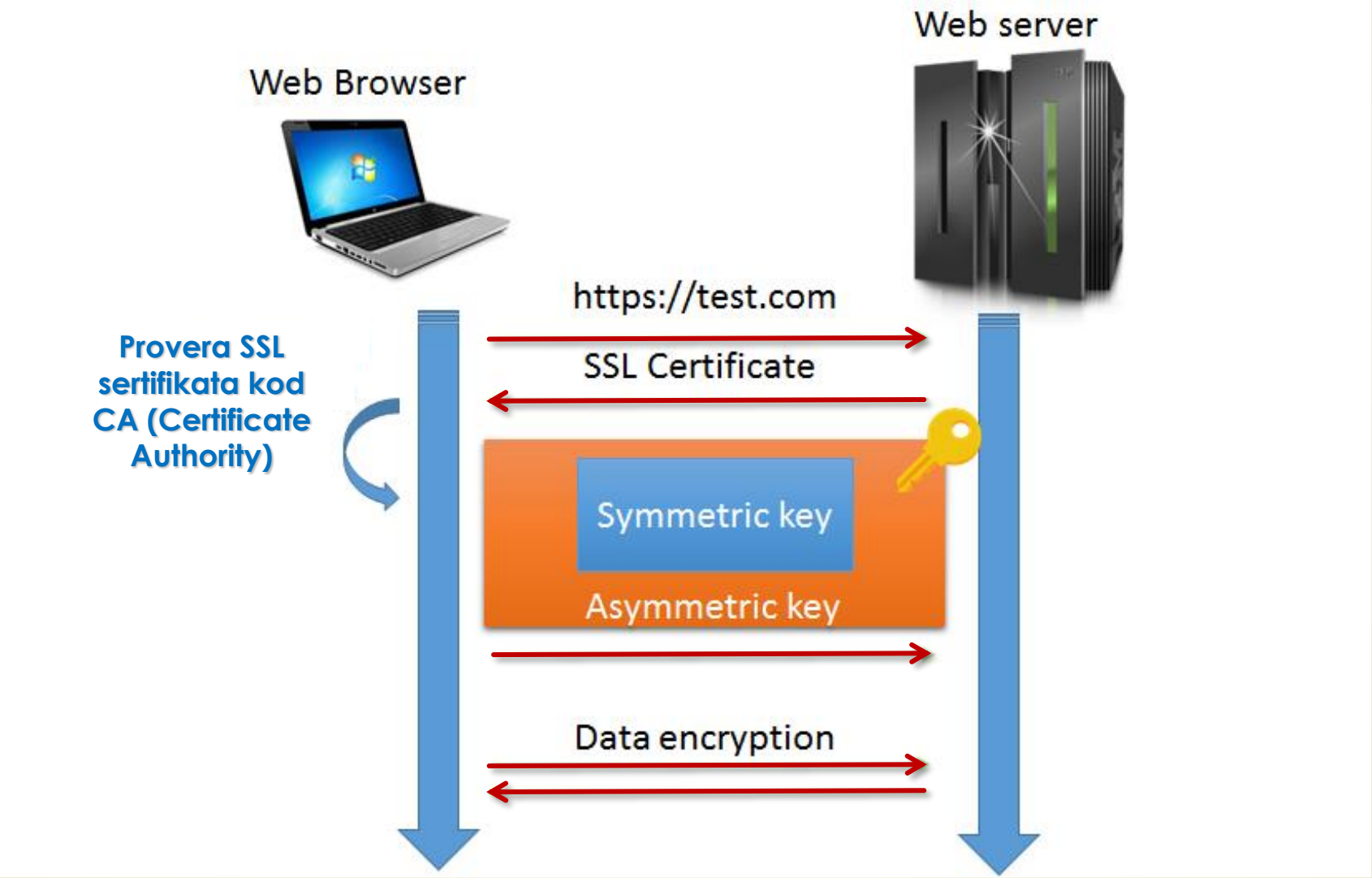
- **HTTP PROTOKOLOM** se prosleđuju poruke kao **OBIČAN TEKST**, što znači da se one veoma lako mogu da **PREGLEDAJU** ili čak **IZMENE**.
- Da bi se **ZAŠTITILE PORUKE** koje šalje/prime bezbedna Web lokacija, treba koristiti **ŠIFROVANJE** i **BEZBEDNE WEB STRANICE**.
- Da bi pristupio mreži koja koristi daljinski pristup, zlonamerni programer (haker) mora da poznaje **VAŽEĆE KORISNIČKO IME** i **LOZINKU**.
- Haker se može poslužiti:
  - Program za **RAZBIJANJE** lozinke,
  - Napasti **DATOTEKU LOZINKI** sistema,
  - Opštim - **PODRAZUMEVANIM NALOZIMA**,
  - **DOBIJA PODATKE** od korisnika s važećim nalogom.

SSL protokol

- Najčešći način za kreiranje **BEZBEDNIH** Web stranice je **RAZMENA ŠIFROVANIH PORUKA** korišćenjem bezbednog protokola **SSL** (engl. *Secure Sockets Layer*).
- Takođe, postojanje mrežne **BARIJERE** koja **FILTRIRA MREŽNE PORUKE** pre ulaska na samu Web lokaciju je dobar način zaštite podataka i transakcija na Web loikaciji.



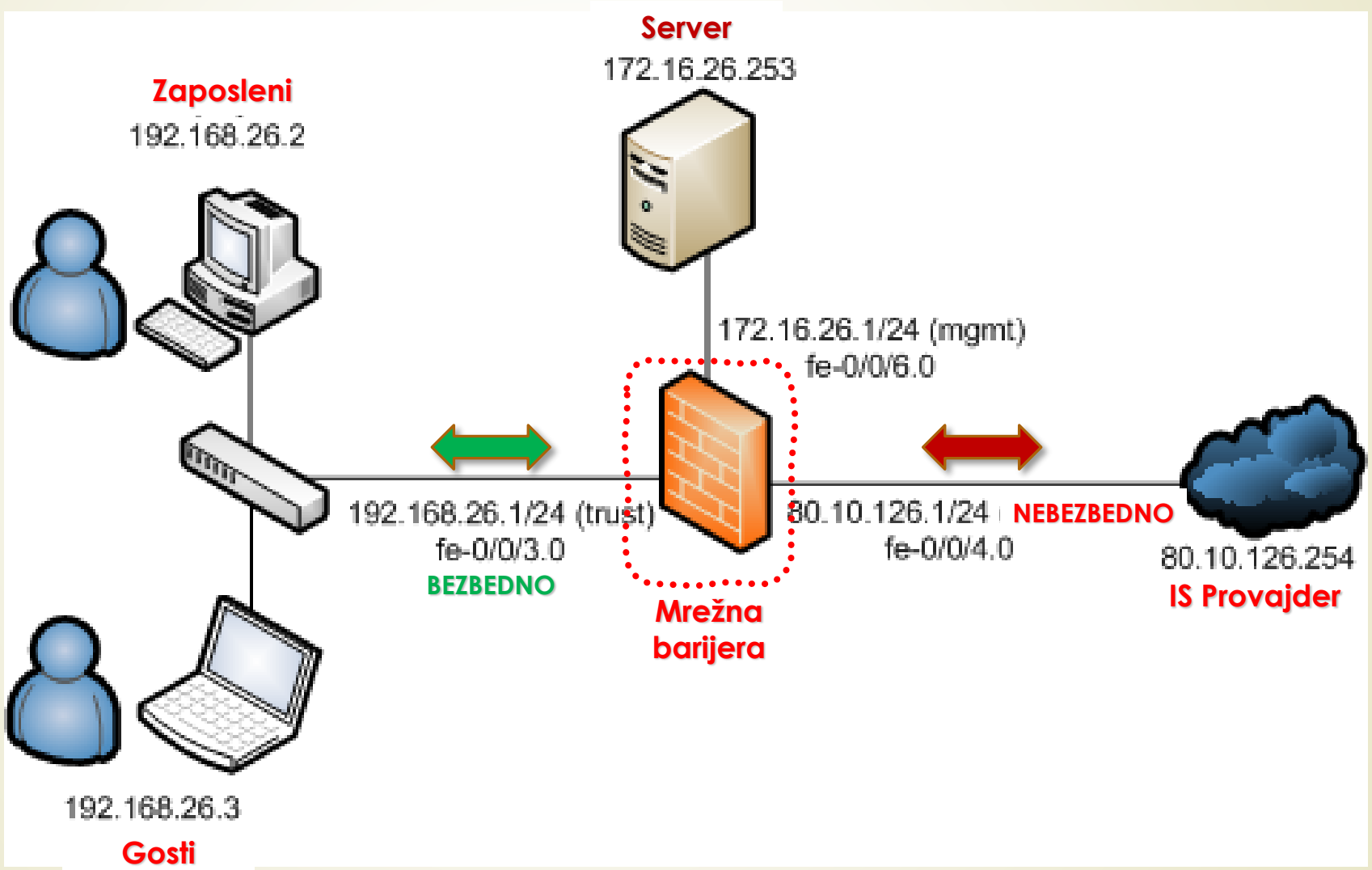
Šifrovanje podataka



Mrežne barijere

- Da bi se sprečilo **ZLONAMERNO** korišćenje baze **LOZINKI** – treba ih **ŠIFROVATI**.
- Vremenski treba **OGRANIČITI TRAJANJE** korisničkih naloga – primer: pristup samo tokom radnih sati.
- Da bi se zaštitili od napada koji **TROŠE RAČUNARSKE RESURSE** – treba koristiti **MREŽNU BARIJERU** (engl. *firewall*) , koja prati **ponavljajuće HTTP** ili slične zahteve.
- **MREŽNU BARIJERA** može da bude realizovana u vidu **POSEBNOG HARDVERSKOG DODATKA** ili na klijentskom **RAČUNARU** na kojem se pokreće odgovarajući softver koji **FILTRIRA PORUKE** koje stiž sa mreže.
- Mrežni programi **IDENTIFIKUJU UDALJENE APLIKACIJE** korišćenjem dobro poznatih brojeva - **BROJEVA PRIKLJUČKA** aplikacije.
- Da bi poslao poruku Web serveru, čitač šalje poruku na **PRIKLJUČAK** sa brojem **80** koji odgovara HTTP protokolu (Lab. Vežba 1).

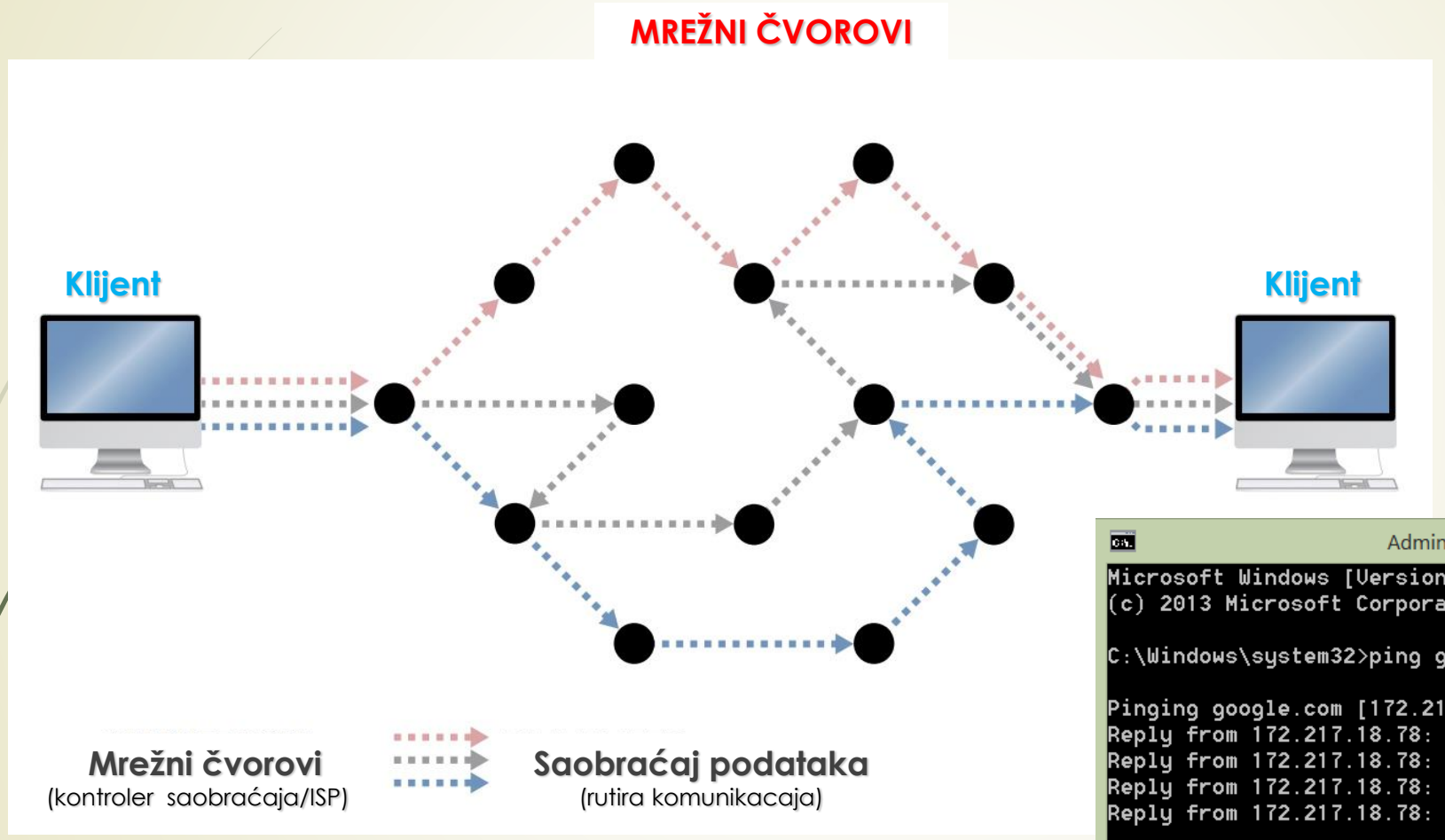
Bezbednosna - mrežna barijera



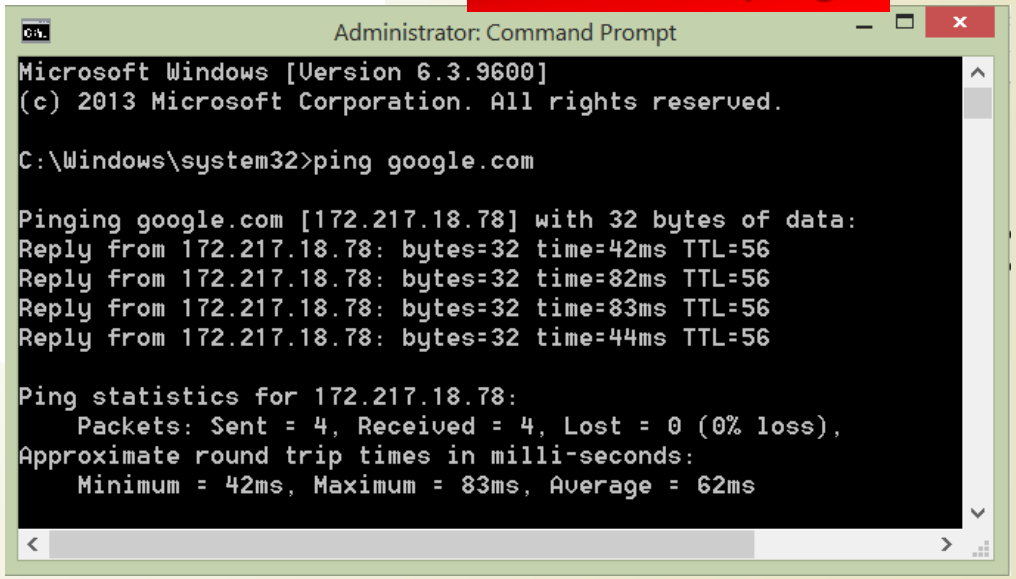
Bezbedni Internet

- **INFORMACIJE** putuju Internetom **POSREDSTVOM PORUKA** koje prolaze kroz **VELIKI BROJ ČVOROVA** na mreži.
- Komanda „**`tracert`**" daje **SPISAK MREŽNIH ČVOROVA** kroz koje poruka putuje do udaljene lokacije, primer: **`c:\>tracert yahoo.com`**
- Po pristizanju poruke na mrežni čvor, **MREŽNI SOFTVER** utvrđuje **DA LI** je poruka namenjena upravo **TOM ČVORU**.
- Ako je odgovor potvrdan, **PORUKA** se prolsleđuje **ODGOVARAJUĆEM PROGRAMU** na računaru (recimo aplikaciji za elektronsku poštu).
- Ako poruka nije namenjena toj lokaciji, **MREŽNI SOFTVER PROSLEĐUJE PORUKU DRUGOM RAČUNARU** i na taj način je **PRIBLIŽAVA** njenom **KONAČNOM ODREDIŠTU**.
- Prilikom prolaska poruke kroz mrežu ona se **MOŽE PROČITATI** i eventualno **IZMENITI** njen sadržaj!

Korišćenje `ping` komande



Komanda **ping**



Šifrovanje poruka

- **ŠIFROVANJE PORUKA**: Da bi izmenjivali šifrovane poruke, dva programa, kao što su programi čitača i servera, moraju prvo da **USVOJE ZAJEDNIČKI ALGORITAM** koji će koristiti da bi **ŠIFROVALI PORUKE**.
- Kada čitač Weba **ZAHTEVA BEZBEDAN PRENOS**, on prvo šalje serveru **SPISAK ALGORITAMA ŠIFROVANJA** koje može podržati.
- **SERVER BIRA ALGORITAM** koji ZAJEDNO PODRŽAVAJU i onda čitaču šalje poruku koja zadaje izabrani algoritam.
- Da bi sprečio hakera da **PRESRETNE KLJUČ ZA ŠIFRU I ALGORITAM**, server koristi posebni ključ za šifrovanje - **JAVNI KLJUČ** - za šifrovanje ključne vrednosti koje će čitač i server kasnije koristiti kada šifruju poruke za vreme **BEZBEDNE SESIJE**.
- Ovo znači da pre nego što klijent i server otpočnu bezbednu sesiju, Web **SERVER MORA IMATI JAVNI KLJUČ**.

Tajni i javni ključ

- **JAVNI KLJUČ**: Da bi slali šifrovane elektronske poruke ili uspostavili bezbedne veze sa Web lokacijom, koristi se **ŠIFROVANJE JAVNIM KLJUČEM**.
- Da bi mogao da prima šifrovane poruke, korisnik treba da DOBIJE **DVA POSEBNA KLJUČA**:
  - **PRIVATNI**, koji čuva, i
  - **JAVNI**, koji može slobodno davati svima.
- Vašim javnim ključem može se **SAMO ŠIFROVATI** poruka koju vi dešifrujete korišćenjem **SVOG PRIVATNOG KLJUČA**.
- Naravno, javni ključ se **RAZLIKUJE** od privatnog ključa, tako da se **NE MOŽE** upotrebiti da bi dešifrovao poruke koje su Vam drugi korisnici poslali.

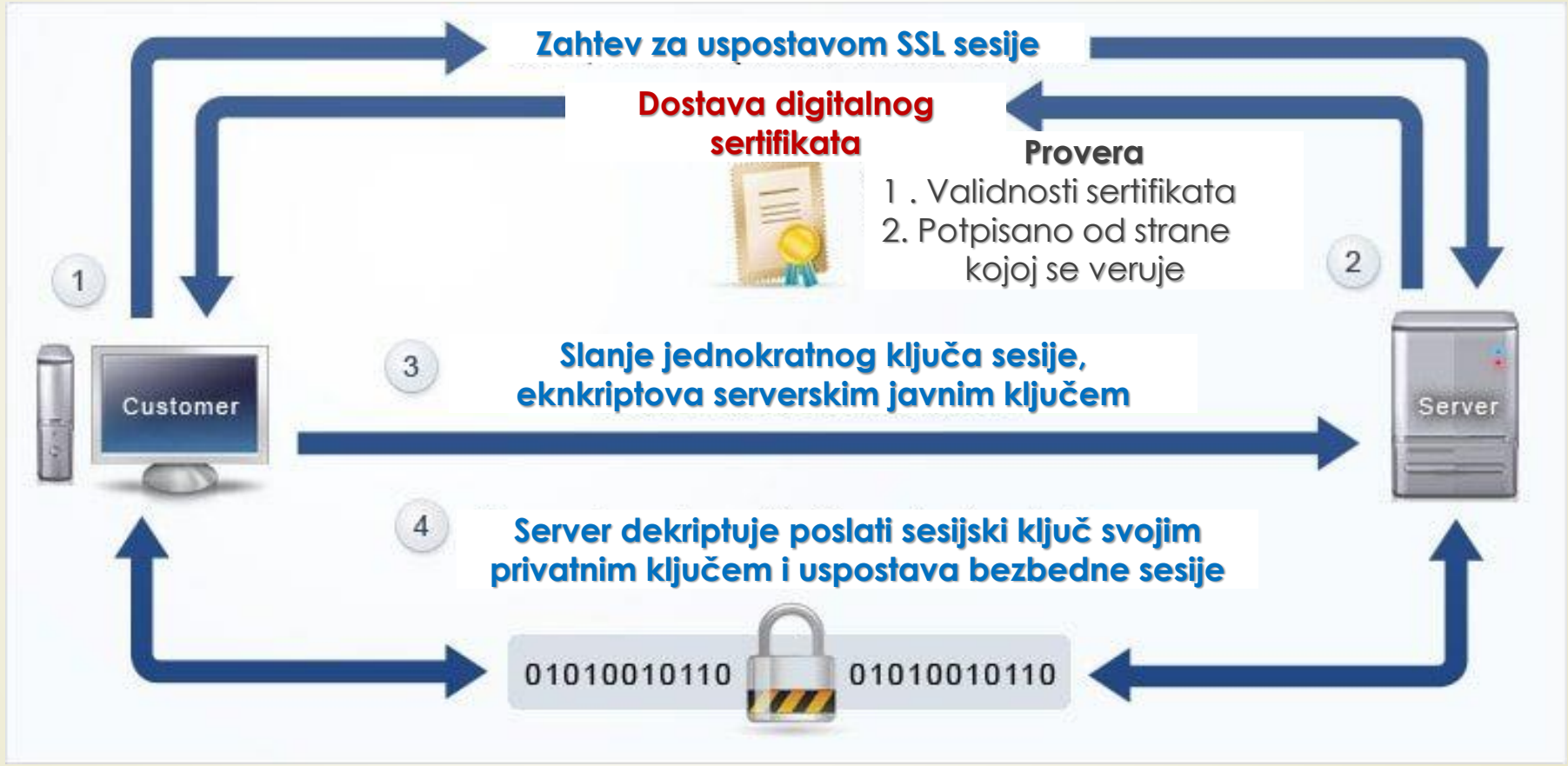
Digitalni potpis, sertifikat, virusi

- **DIGITALNI POTPIS** je **JEDINSTVENA VREDNOST** koja se **DODAJE PORUCI** da bi primalac bio siguran da je niko **NIJE IZMENIO** pri prolasku kroz režu.
- Server mora posedovati **DIGITALNI SERTIFIKAT** (engl. *Digital Certificate*) čiji je glavni deo **JAVNI KLJUČ** koje koristi softver Web servera.
- Kada se **OBEZBEDI SERTIFIKAT** korisnici mogu **ZAHTEVATI BEZBEDNE** stranice sa servera koristeći **`https`** protokol.
- **IZLOŽENOST VIRUSIMA**: Računarski virusi su **PRETNJA SVAKOM RAČUNARU** koji se poveže na mrežu.
- Savremene aplikacije sadrže **MAKROE** koji **AUTOMATIZUJU ODREĐENE POSLOVE** i ozbiljna su **PRETNJA BEZBEDNOSTI**.
- Korisnici mogu da **INFICIRAJU SVOJE SISTEME** virusima čim otvore *Word* ili *Excel* dokument koji sadrži **MAKRO VIRUS**.

Sesijski ključ

- Da bi se **SMANJIO RIZIK** od **INFICIRANJA**, svaki sistem u mreži - i serveri i radne stanice - moraju da pokrenu **SOFTVER ZA OTKRIVANJE VIRUSA**.
- Ako lokaciju obezbeđujete **BARIJEROM**, ona takođe može da podržava **OTKRIVANJE VIRUSA** i to treba da koristite kao **PRVU LINIJU ODBRANE**.
- Softver za otkrivanje virusa se **MORA REDOVNO AŽURIRATI**.
- Treba preuzimati i instalirati samo **POTPISANE `ActiveX` OBJEKTE** koji imaju **DIGITALNI SERTIFIKAT** o autoru.
- Kada čitač pristupi bezbednoj lokaciji, **SERVER ŠALJE ČITAČU NEŠIFROVANU PORUKU** koja sadrži **SERVEROV JAVNI KLJUČ**.
- Čitač koristi **POSLATI KLJUČ** za **DEŠIFROVANJE** poruke sa brojevima koje **OBA** mogu da koriste za generisanje **SESIJSKOG KLJUČA**.
- Kada se **SESIJSKI KLJUČ** ugovori i server i čitač, koriste **SESIJSKI KLJUČ** za **ŠIFROVANJE** I **DEŠIFROVANJE** poruka.

Bezbedna sesija, razmena ključeva



NTFS sistem dozvola za pristup

- **NTFS** (engl. ***N****ew* ***T****echnology* ***F****ile* ***S****ystem*) tehnologija je razvijena za rad sa **DATOTEKAM** i nalazi se u vlasništvu Microsofta.
- Upotreba **NTFS SISTEMA DATOTEKA** se svakoj **DATOTECI** i svakom **DIREKTORIJUMU** mogu dodeliti **POSEBNE DOZVOLE** za pristup.
- Ove dozvole kontrolišu **KOJI KORISNICI** mogu da pristupe resursu i **ŠTA** svaki korisnik može sa njim da radi.
- Liste za upravljanje pristupom **ACL** (engl. ***A****ccess* ***C****ontrol* ***L****ist*) kao i **DACL** (engl. ***D****iscretionary* ***A****ccess* ***C****ontrol* ***L****ist*) liste definišu tačno koji tip interakcije je dozvoljen (ili zabranjen).
- Interakcije kao što su čitanje, upis, izvršavanje ili brisanje se dodeljuju **POJEDINIM** korisnicima ili **GRUPI** korisnika.

Preporuke za bezbednost na Web-u

- Preveniranje *Cross-Site Scripting*-a,
- Preveniranje preuzimanje sesija,
- Obezbeđivanje *RESTfull* servisa,
- Korišćenje *CAPTCHA* identifikacije,
- Authentifikacija, autorizacija i logovanje korisnika,
- Preveniranje gubitaka podataka,
- Obezbeđivanje izvršenja sistemskih i udaljenih metoda,
- Obezbeđivanje baze podataka,
- Korišćenje šifrovanja,
- Obezbeđivanje mrežne konekcije pomoću SSL-a.