Osnovni Principi i Šabloni Bezbednosti

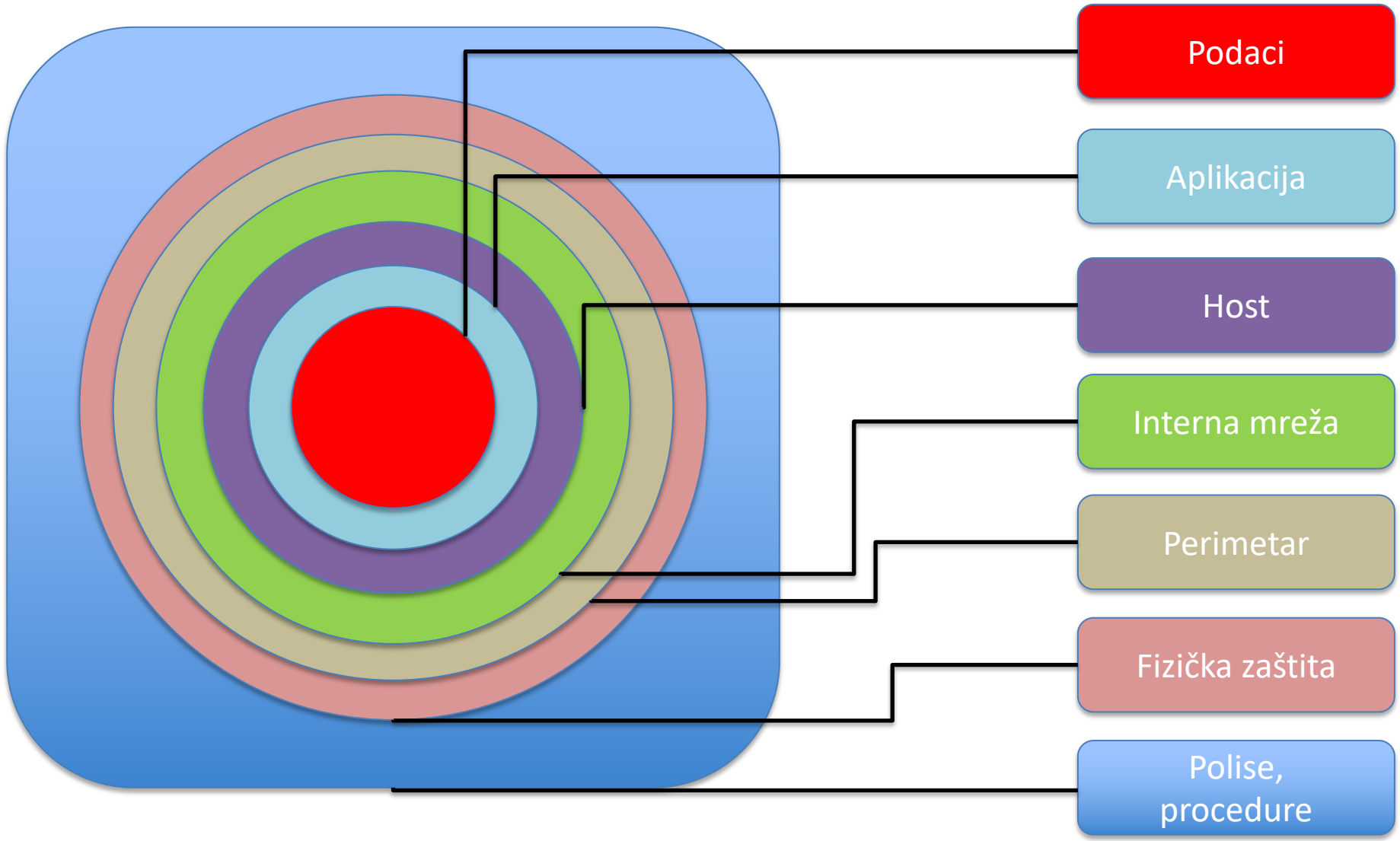
dr Goran Sladić

Sadržaj predavanja

- Osnovni principi bezbednog dizajna
- Bezbedni softverski obrasci
- Privatnost po dizajnu (engl. *privacy by design, PbD*)

Osnovni principi bezbednog dizajna

Odbrana po dubini (engl. *defense in depth*)



Odbrana po dubini (engl. *defense in depth*)

- Ideja je da se koristi slojevit pristup za odbranu sistema i njegove imovine
- Primeri odbrane po dubini uključuju sledeće:
  - Odbrana određene radne stanice bravama, kamerama, *air-gapping*
  - Uvođenje bastion hosta (ili vatrenog zida ☺) između sistema i javnog interneta, zatim *endpoint* agenta u samom sistemu
  - Korišćenje višefaktorske autentifikacije za autentifikaciju, sa vremenskim kašnjenjem koje se eksponencijalno povećava između neuspešnih pokušaja
  - Postavka *honeypot-a npr.* lažne baze podataka sa funkcijama autentifikacije
- Svaki dodatni faktor koji čini napad skupljim u smislu složenosti, novca i/ili vremena je uspešan sloj u odbrani u dubini
- Treba balansirati između toga koliko potrošiti da bi se obezbedio sistem u odnosu na procenjenu vrednost tog sistema

Razdvajanje privilegija (engl. *separation of privilege*)

- *Sepraration of duties*
- Predstavlja razdvajanje pristupa funkcionalnostima ili podacima unutar sistema tako da jedan akter ne drži sva prava
- Povezani koncept je *dual approval* gde jedan korisnik (ili proces) može da zahteva da se izvrši operacija i podesiti neke ili sve parametre, a drugi korisnik ili process je potreban da autorizuje da se transakcija nastavi
- To znači da jedan entitet ne može da obavlja zlonamerne aktivnosti nesmetano ili bez mogućnosti nadzora

Jednostavan dizajn – ekonomija mehanizma

- Održavanje stvari jednostavnim znači izbegavanje prekomernog inženjeringa sistema
- Sa složenošću dolazi i povećan potencijal za nestabilnost, izazove u održavanju i druge aspekte rada sistema, kao i potencijal za neefikasne bezbednosne kontrole
- Mora se voditi računa da se izbegne i preterano pojednostavljivanje (kao što je ispuštanje ili previđanje važnih detalja)
  - Često primer je validacija ulaza gde se pretpostavlja da će korisnik uvek uneti validne i bezbedne podatke
- Umereno jednostavan dizajn u odnosu na komplikovani dizajn će često obezbediti bezbednosne prednosti tokom vremena

Otvoreni dizajn

- Neoslanjati se na tajni dizajn, neznanje napadača ili *security by obscurity* kao na sredstva bezbednosti
- Dizajn sistema treba da bude otporan na napade čak i ako je svaki detalj njegove implementacije poznat i javan
- Trebalo bi da postoji dovoljno bezbednosnih kontrola kako bi aplikacija bila bezbedna bez skrivanja osnovne funkcionalnosti ili izvornog koda
- Kerckhoff-ov princip[1] – kriptografski sistem treba da bude bezbedan čak i ako je sve u vezi sa sistemom, osim ključa, javno poznato
  - Claude Shannon – neprijatelj zna sistem (Shannon's maxim)

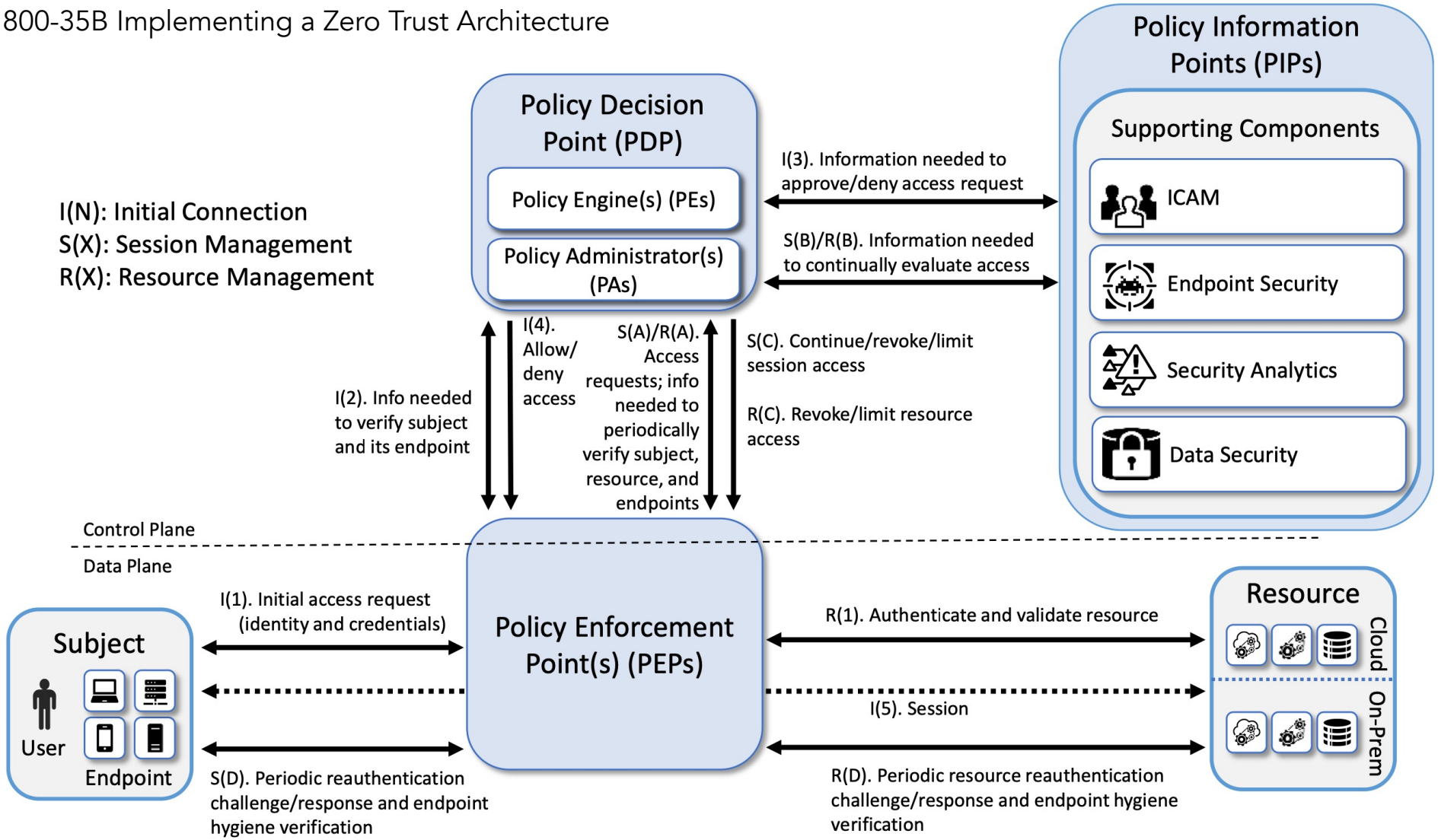
[1] Kerckhoffs's principle - https://en.wikipedia.org/wiki/Kerckhoffs%27s_principle

Nulto poverenje (engl. *zero trust*)

- Uobičajeni pristup dizajnu sistema i usklađenosti sa bezbednošću je „verujte, ali proverite“ ili nulto poverenje (engl. *trust, but verify*; *zero trust*)
  - Prvo - podrazumeva pretpostavku najboljeg ishoda za operaciju (kao što je uređaj koji se pridružuje mreži ili klijent poziva API)
  - Drugo - izvršiti verifikaciju poverenja u dizajn ili implementaciju
- U okruženju nultog poverenja sistem ignoriše (ili nikada ne uspostavlja) bilo koji prethodni odnos poverenja i umesto toga sve verifikuje pre nego što odluči da uspostavi odnos poverenja (koji tada može biti privremen)
- Poznat i pod terminom *potpuna medijacija* (engl. *complete mediation*)
- Koncept samo zvuči jednostavno - obezbediti da se prava pristupa operaciji proveravaju svaki put kada se pristupi objektu i da se unapred provere prava za tu operaciju pristupa
  - Tj. mora se proveriti da li aktor ima odgovarajuća prava da pristupi objektu svaki put kada se traži pristup

Nulto poverenje (engl. *zero trust*)

Slika iz: NIST SP 1800-35B Implementing a Zero Trust Architecture



Least common mechanism

- Mehanizmi koji se koriste za pristup resursima ne bi trebalo da se dele
- Deljenje resursa obezbeđuje kanal duž kojeg se informacije mogu preneti, pa takvo deljenje treba svesti na minimum
- U praksi, ako operativni sistem pruža podršku za virtuelne mašine, operativni sistem će do određenog stepena automatski primeniti ovu privilegiju
- U suprotnom, pružiće određenu podršku (kao što je virtuelni memorijski prostor), ali neće biti potpuna podršku (jer će se sistem datoteka pojaviti kao deljeni između nekoliko procesa)

Najmanje privilegije (engl. *least privilege*)

- Operacija treba da se izvodi koristeći samo najrestriktivniji nivo privilegija koji još uvek omogućava da operacija uspe
- U svim slojevima i u svim mehanizmima, treba obezbediti da dizajn ograničava operatera na minimalni nivo pristupa koji je potreban da bi se izvršila pojedinačna operacija
- Ako se ne poštuju najmanje privilegije, ranjivost u aplikaciji može dati potpuni pristup operativnom sistemu, a sa njim i sve posledice privilegovanog korisnika koji ima neometan pristup sistemu
- Ovaj princip se primenjuje za svaki sistem koji održava autorizacioni kontekst (npr. operativni sistem, aplikacija, baze podataka, itd.)

Fail secure

- Kada sistem naiđe na stanje greške, potencijalnom protivniku se ne otkriva previše informacija (npr. logovi ili poruke o grešci korisnika) i ne dodeljuje se neadekvatan pristup, (npr. ako postoji greška u podsistemu za autentifikaciju)
- Postoji značajna razlika između *fail safe* i *fail secure*
  - *Fail safe* – kada nestane struje vrata se otključavaju, sigurno za ljude, ne za proctor (struja se primenjuje da se vrata zaključaju po dizajnu)
  - *Fail secure* - kada nestane struje, vrata ostaju zaključana, što se smatra bezbednim stanjem (odbijanje daljih zahteva; struja se primenjuje da se vrata otključaju po dizajnu)
- Koji pad je odgovarajući u datoj situaciji zavisi od konteksta situacije
- Na kraju, *fail secure* znači da ako se komponenta ili logika u sistemu pokvari, rezultat će se i dalje smatrati sigurnim i stanje sistema stabilno

Efikasno logovanje

- Bezbednost nije samo sprečavanje loših stvari da se dese, već i svest da se nešto dogodilo
- Efikasni logovi treba da odgovore na sledeća pitanja:
  - Ko je izvršio određenu radnju da izazove radnju?
  - Kada je izvršena radnja?
  - Kojoj funkciji ili podacima je pristupio proces ili korisnik?
- Neporecivost, koje je usko povezano sa integritetom, znači posedovanje skupa transakcija koje pokazuju ko je šta uradio, pri čemu log o svakoj transakciji održava integritet
- Sa ovim konceptom, nemoguće je da aktor tvrdi da nije izveo određenu radnju

Ljudski factor (engl. *psychological acceptability*)

- Ljudski korisnici se pominju kao najslabija karika u bilo kom sistemu
- tako da koncept psihološke prihvatljivosti mora biti osnovno ograničenje dizajna
- Korisnici koji su frustrirani jakim merama bezbednosti će pokušati da pronađu načine da ih zaobiđu
- Prilikom razvoja bezbednog sistema, ključno je odlučiti koliko će bezbednost biti prihvatljiva za korisnika
  - Loša ideja – MFA sa 10 faktora ☺
- Ako je implementirano previše bezbednosnih kontrola:
  - Korisnik će prestati da koristi sistem
  - Korisnik će pokušati da pronađe rešenja da zaobiđe bezbednosne mere
  - Biznisu je uvek prioritet da korisnici koriste rešenje - *usability over security*

Allowlists over Blocklists

- Dati prednost listama dozvoljenih vrednosti umesto listama blokiranih vrednosti prilikom dizajniranja bezbednosnog mehanizma
- Liste dozvoljenih vrednosti su enumeracije onoga što je bezbedno, tako da su one same po sebi konačne
- Liste blokiranih vrednosti pokušavaju da nabroje sve što nije bezbedno, i čineći to implicitno dozvoljavaju beskonačan skup stvari za koje se misli da su bezbedne

Dizajn po ugovoru (engl. *design by contract*)

- Dizajn po ugovoru je povezan sa principom nultog poverenja i pretpostavlja da će svaki put kada klijent pozove server, ulaz koji dolazi od tog klijenta biti određenog fiksnog formata i neće odstupati od tog ugovora
  - Primer brave i ključa - brava prihvata samo ispravan ključ i ne veruje ničemu drugom
  - U „Securing th Tangled Web“[1] dat je primer kako je Google značajno smanjio količinu XSS grešaka koristeći biblioteku po dizajnu inherentno bezbednih API poziva
- Dizajn po ugovoru rešava nulto poverenje tako što obezbeđuje da svaka interakcija prati fiksni protokol

[1] Kern C. Securing the Tangled Web - https://queue.acm.org/detail.cfm?id=2663760

Built in, not bolt on

- Bezbednost, privatnost i sigurnost treba da budu osnovna svojstva sistema, a sve bezbednosne karakteristike treba da budu uključene u sistem od početka
- O bezbednosti ne bi trebalo da se razmišlja naknadno ili da se oslanja isključivo ili primarno na prisustvo spoljnih komponenti sistema
- Dobar primer ovog obrasca je implementacija bezbedne komunikacije
  - Sistem treba da bude dizajniran da podrži TLS ili sličan metod za očuvanje poverljivosti podataka u tranzitu
  - Oslanjanje na korisnika da instalira specijalizovane hardverske sisteme kako bi se omogućila bezbednost komunikacije od kraja do kraja znači da će, ako korisnik to ne učini, komunikacija biti nezaštićena i potencijalno dostupna zlonamernim akterima
- **Nikad ne pretpostavljati da će korisnici preduzeti akciju umesto Vas kada je u pitanju bezbednost vašeg sistema!**

Nesklonost poverenju (engl. *reluctance to trust*)

- Poverenje bi uvek trebalo da bude eksplicitan izbor, zasnovan na čvrstim dokazima
  - Zahteva se provera autentičnosti koda pre instalacije i korišćenja
  - Zahteva se snažna autentifikacija pre autorizacije

Bezbedni softverski obrasci

Koncept 1/2

- **Softverski obrazac** je opšte rešenje za višestruku upotrebu za uobičajeni problem u datom kontekstu u dizajnu softvera
- **Bezbednosni obrazac** opisuje rešenje problema kontrole (sprečavanja ili mitigacije) specifičnih pretnji koristeći određeni bezbednosni mehanizam, definisan u datom kontekstu
- Bezbednosni obrasci pomažu programerima koji nisu stručni za bezbednost da inkorporiraju bezbednost u svoje dizajne

Koncept 2/2

- Bezbednosni obrasci mogu biti arhitektuuralni obrasci kada opisuju koncepte globalne softverske arhitekture (npr. AD, PKI, mrežna segregacija)
- Bezbednosni obrasci mogu biti obrasci dizajna kada opisuju strukture na nivou koda aplikacije (npr. validacija unosa, logovanje, upravljanje izuzecima)
- Bezbednosni obrasci mogu da obuhvataju i arhitekturu sistema i dizajn aplikacije (npr. kontroler domena i programiranje usmereno na aspekte)
- Naš fokus će biti na obrascima bezbednosnog dizajna koji se odnose na aplikativni softver i obično podrazumevaju neku integraciju na nivou koda

Struktura bezbednosnih obrazaca

- Problem
  - Kontekst u kome problem postoji i rešenje šablona je primenljivo
  - Opis bezbednosnog problema u identifikovanom kontekstu
  - Primer
- Rešenje
  - Koncept obrasca
  - Struktura (statički prikaz) rešenja šablona
  - Dinamički aspekti rešenja
  - Smernice za implementaciju
  - Primer scenarija rezolucija
  - Posledice
  - Poznate slučajevi korišćenja

Katalog bezbednosnih obrazaca

Poverljivost

- Simetrična enkripcija
- Asimetrična enkrpicija

Integritet

- Heš funkcije
- Digitalni potpisi
- Validacija podataka

Dostupnost

- Dizajn visoke dostupnosti
- *Performance counters*
- Filtering

Autentifikacija

- Mehanizmi za autentifikaciju

Autorizacija

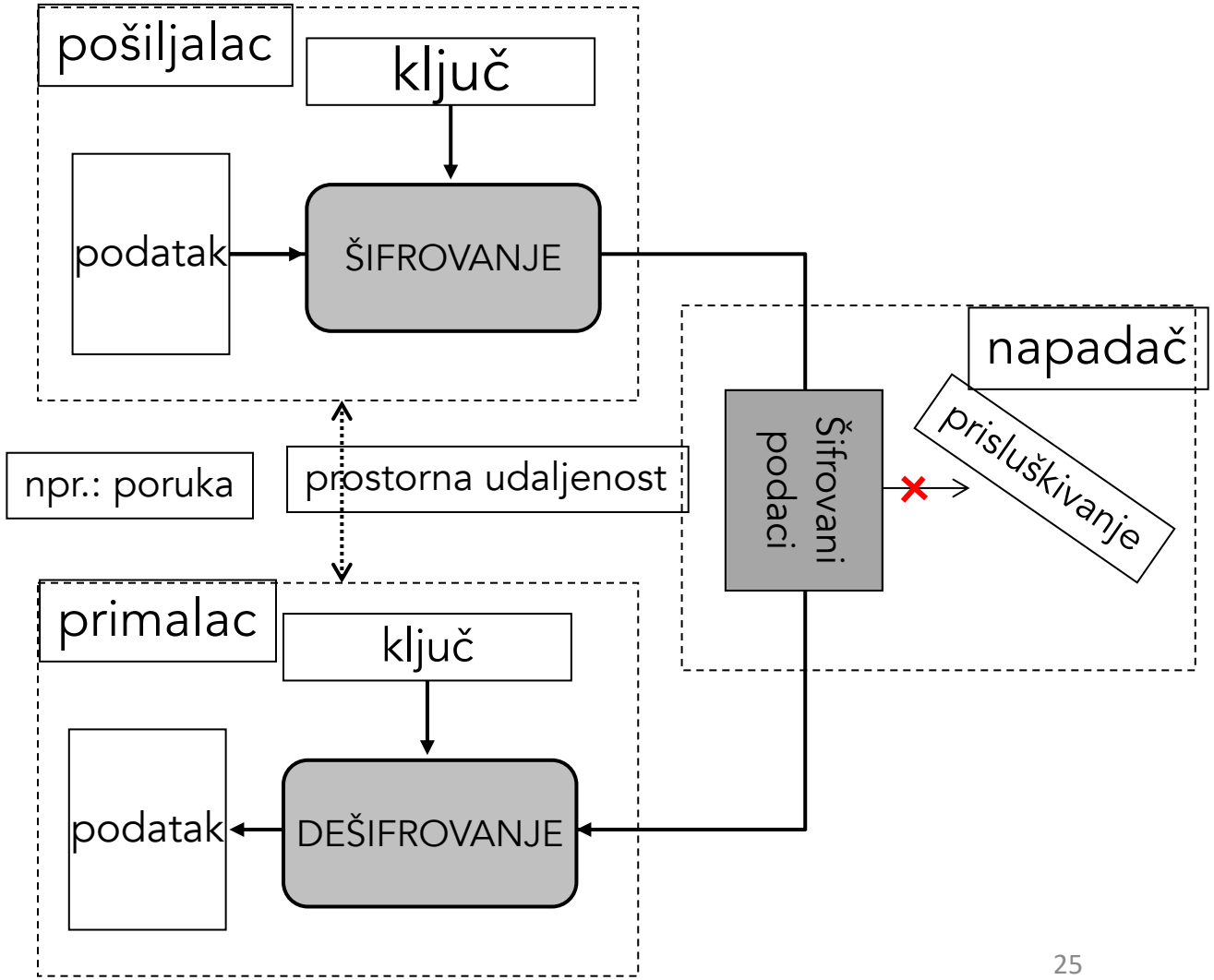
- Role-Based Access Control
- Access Control Lists

Accountability

- Logger

Poveljivost podataka 1/3

- Problem
  - Dva subjekta (e.g., pretraživač i server) žele da razmene tajnu
  - Bez potpune kontrole komunikacione mreže, tajnu bi mogla pročitati treća strana
  - Provajder WiFi pristupne tačke u kafiću evidentira sav saobraćaj i prikuplja korisničke akreditive za elektronsko bankarstvo

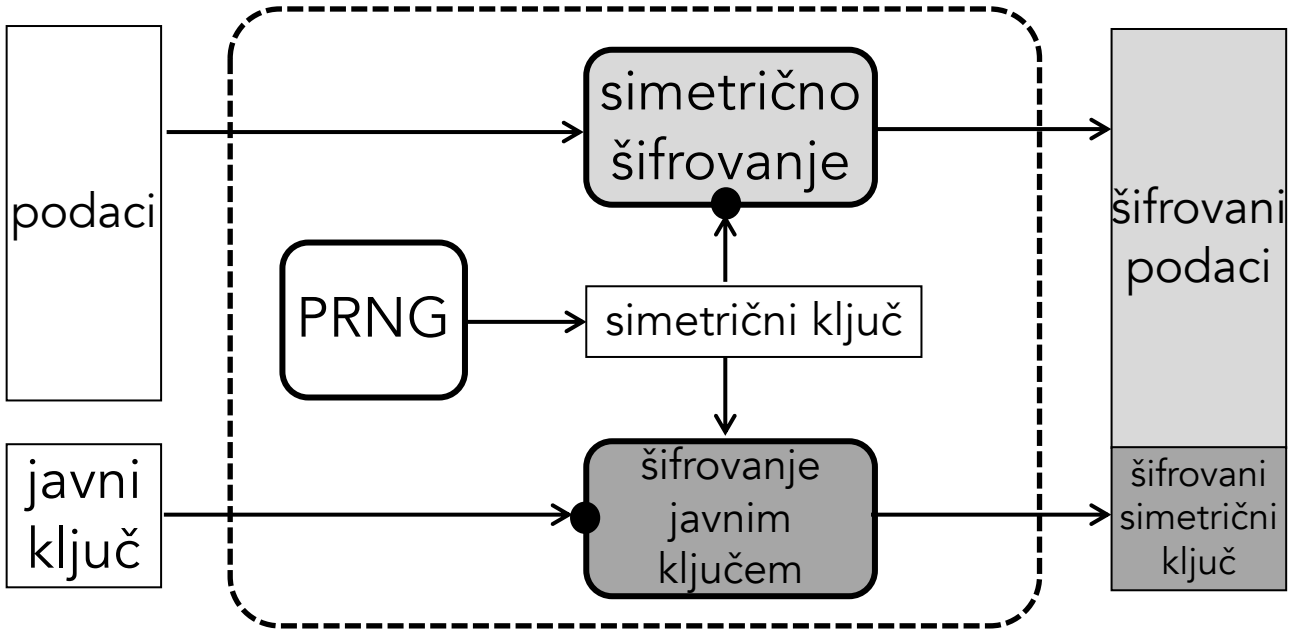


Poveljivost podataka 2/3

- Kriptografija pretvara otvoreni tekst u kriptografski tekst uz pomoć kriptografskog ključa
- Sve dok se ključ čuva u tajnosti (i bezbedni algoritmi se koriste), napadač ne može da otkrije originalni otvoreni tekst
- **Simetrične šifre** – ključ za šifrovanje i dešifrovanje je isti
  - Koje su karakteristike ovog modela?
- **Asimetrične šifre** – jedan ključ za šifrovanje, drugi za dešifrovanje
  - Koje su karakteristike ovog modela?

Poveljivost podataka 3/3

- Hibridna kriptografija
  - Asimetrična kriptografija je sporija od simetrične kriptografije i ključevi su obično veći
  - Hibridna kriptografija rešava problem brzine



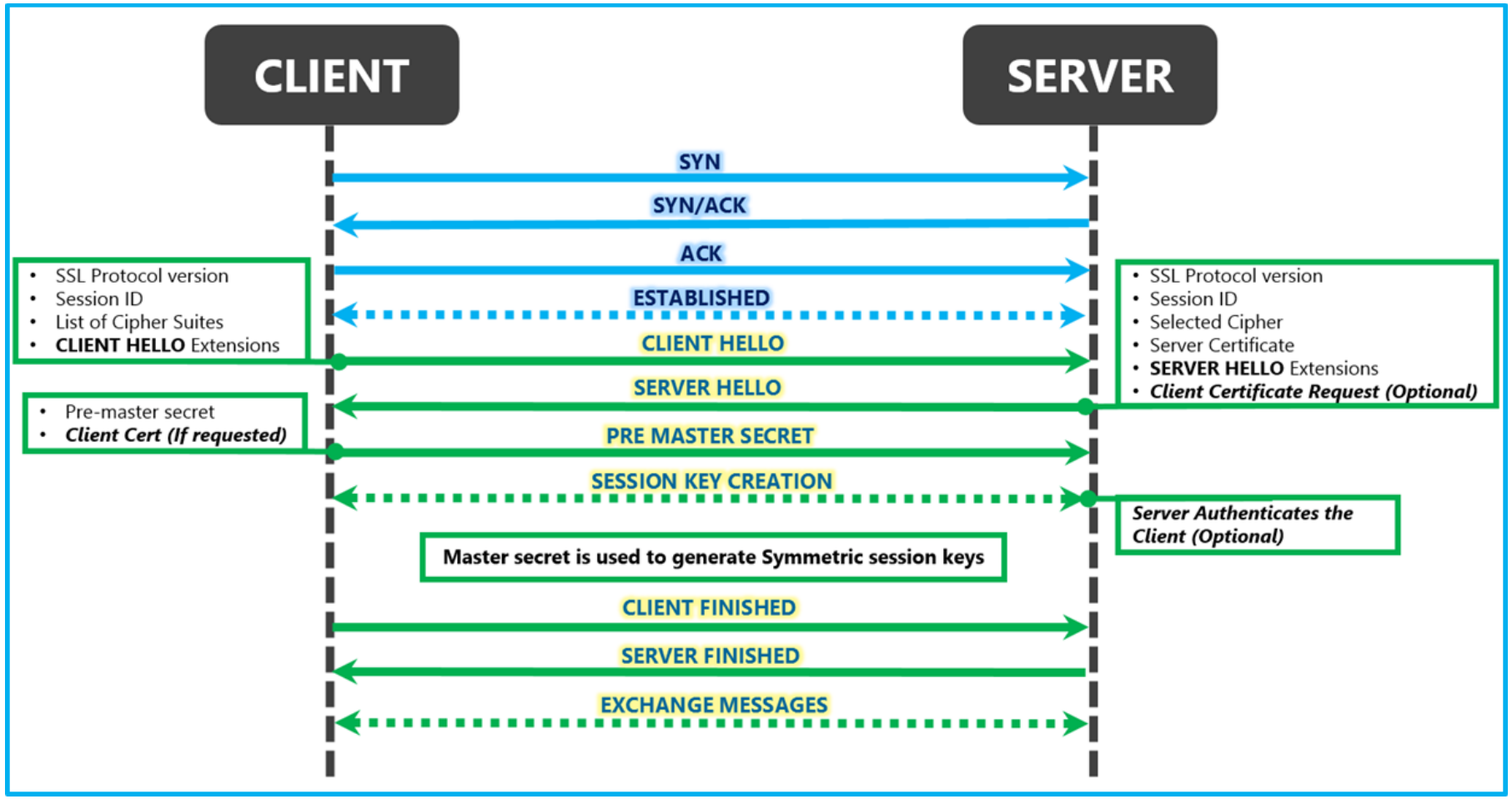
Šifrovanje – greške u praksi?

- Problem upravljanja ključevima
  - Ključevi se generišu sa slabim PRNG
  - Ključevi se ne prenose i čuvaju na siguran način
- Slabosti protokola
  - Koriste se slabi algoritmi
  - Nebezbedna konfiguracija (režim rada, dužina ključa)
- Implementacioni problem
  - Bagovi
  - *Side channels* (npr. *timing attacks*, *differential power analysis*)
- Čovek

Studija slučaja: HTTPS

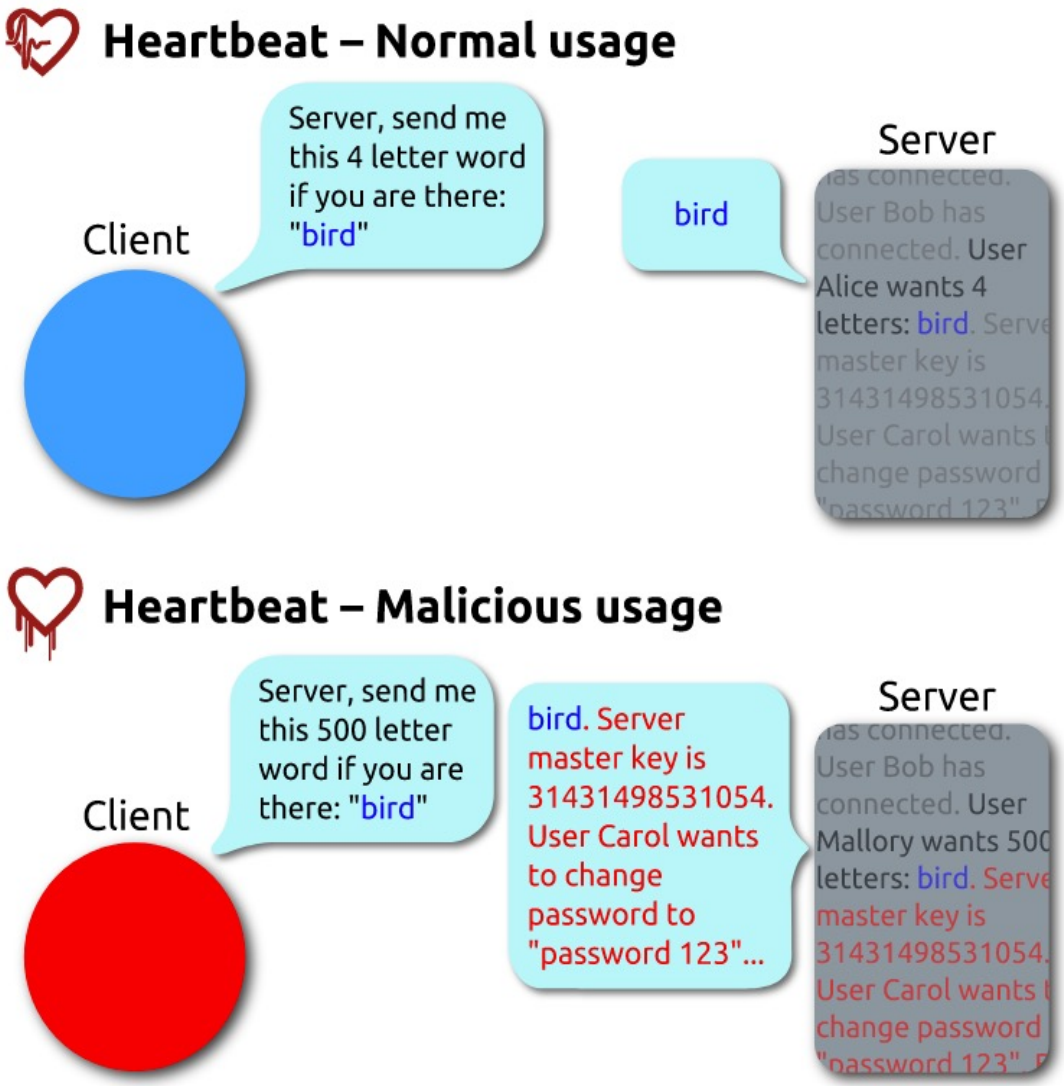
HTTP over TLS

TLS handshake



Heartbleed ranjivost

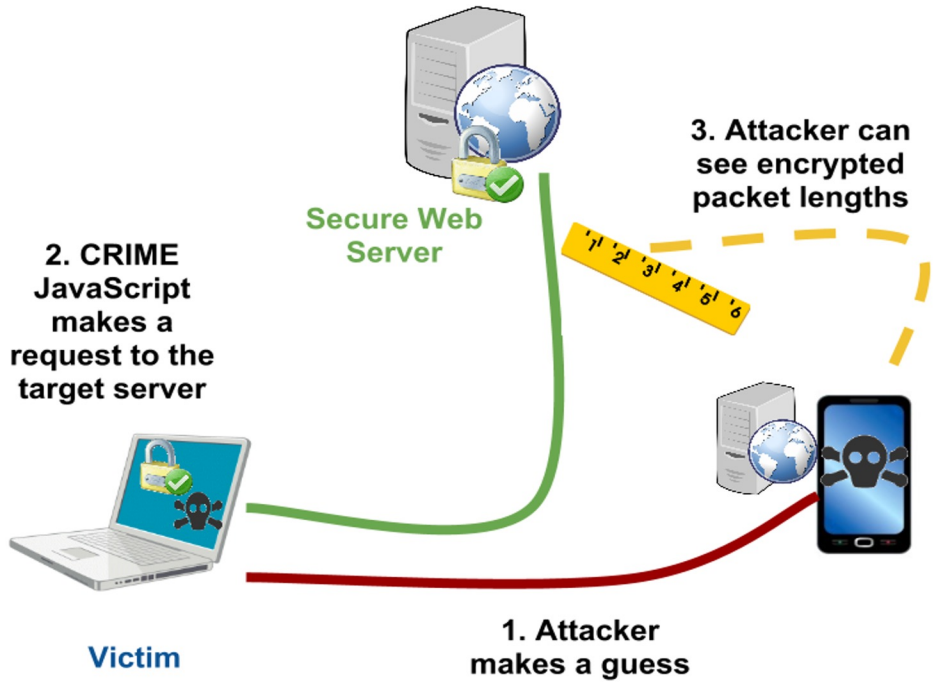
- OpenSSL biblioteka uključuje implementaciju TLS-a u kojoj je postojala *buffer underflow* ranjivost skoro dve godine
- Ovo se može posmatrati kao problem u implementaciji



CRIME napad

- Bazira se na TLS kompresiji i činjenici da je dužina zahteva dostupna napadaču
- len(encrypt(compress(input + public + secret)
  - input – URL
  - public – Known headers
  - secret – Cookie
- Napadač mora da bude u stanju da nadgleda (*sniffing*) saobraćaj i uputi mnogo zahteva u ime žrtve
- Primer slabosti protokola

```
GET /twid=0
Host: twitter.com
User-Agent: Chrome
Cookie:
twid=71bc3e...
```

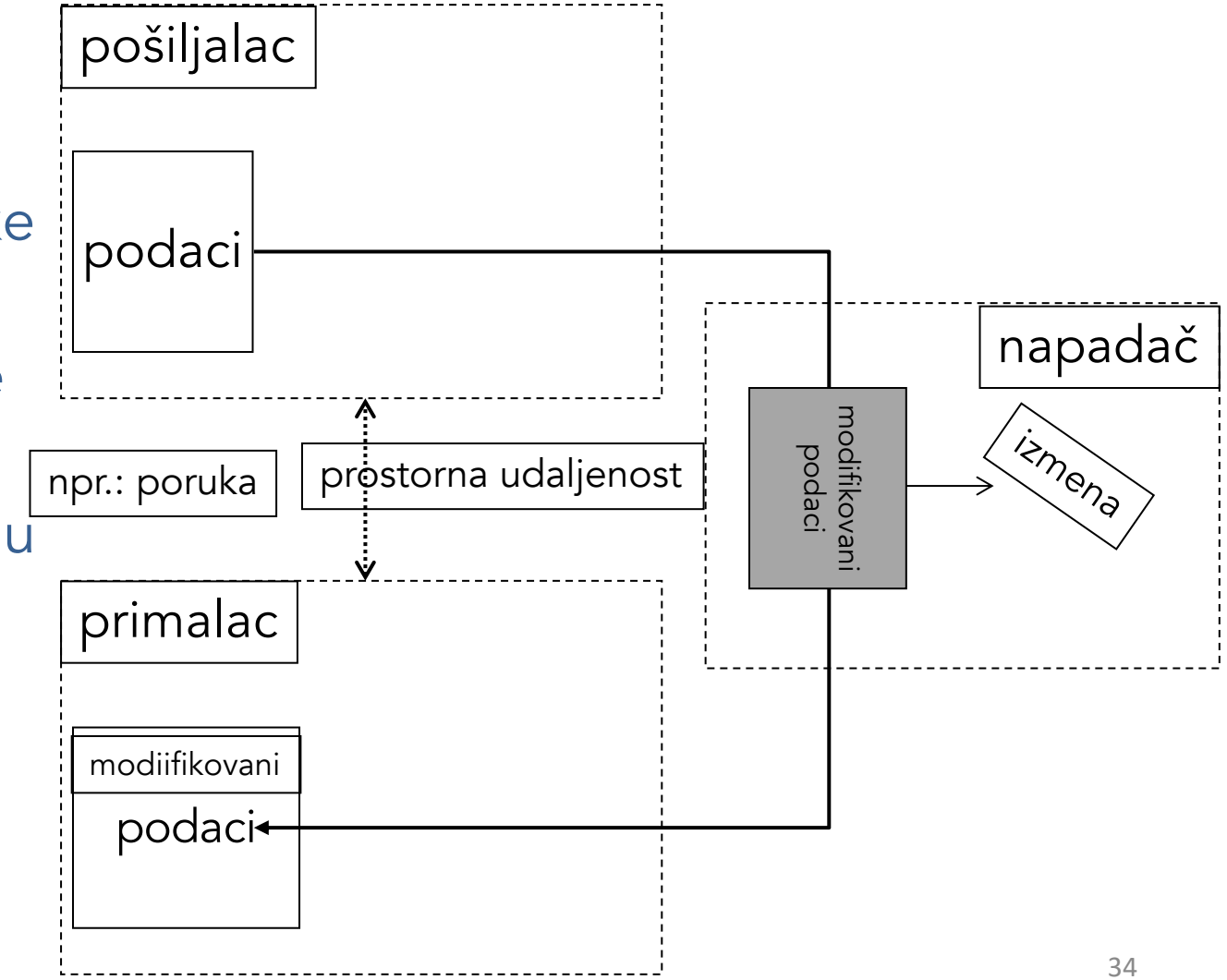


Zaključak

- Čak i najpoznatije bezbednosne kontrole ima ranjivosti
- Samo primena bezbednosne kontrole nije dovljna
- Za sve bezbednosne kontrole potrebno je i:
  - Razumeti njihovu namenu - izbegavati nedostatke dizajna
  - Primeniti odgovarajuću konfiguraciju u skladu sa najboljim praksama – izbegavati pogrešnu bezbednosnu konfiguraciju (*security misconfiguration*)
  - Koristiti rešenja od poverenja – izbegavanje grešaka u implementaciji
  - Biti ažuran (engl. *up-to-date*)

Integritet podataka 1/4

- Problem
  - Dva subjekta (npr. pretraživač i server) žele da razmene podatke
  - Bez potpune kontrole komunikacione mreže, podatke može promeniti treća strana
  - Provajder WiFi pristupne tačke u kafiću menja sve zahteve za plažanje tako da on bude primalac

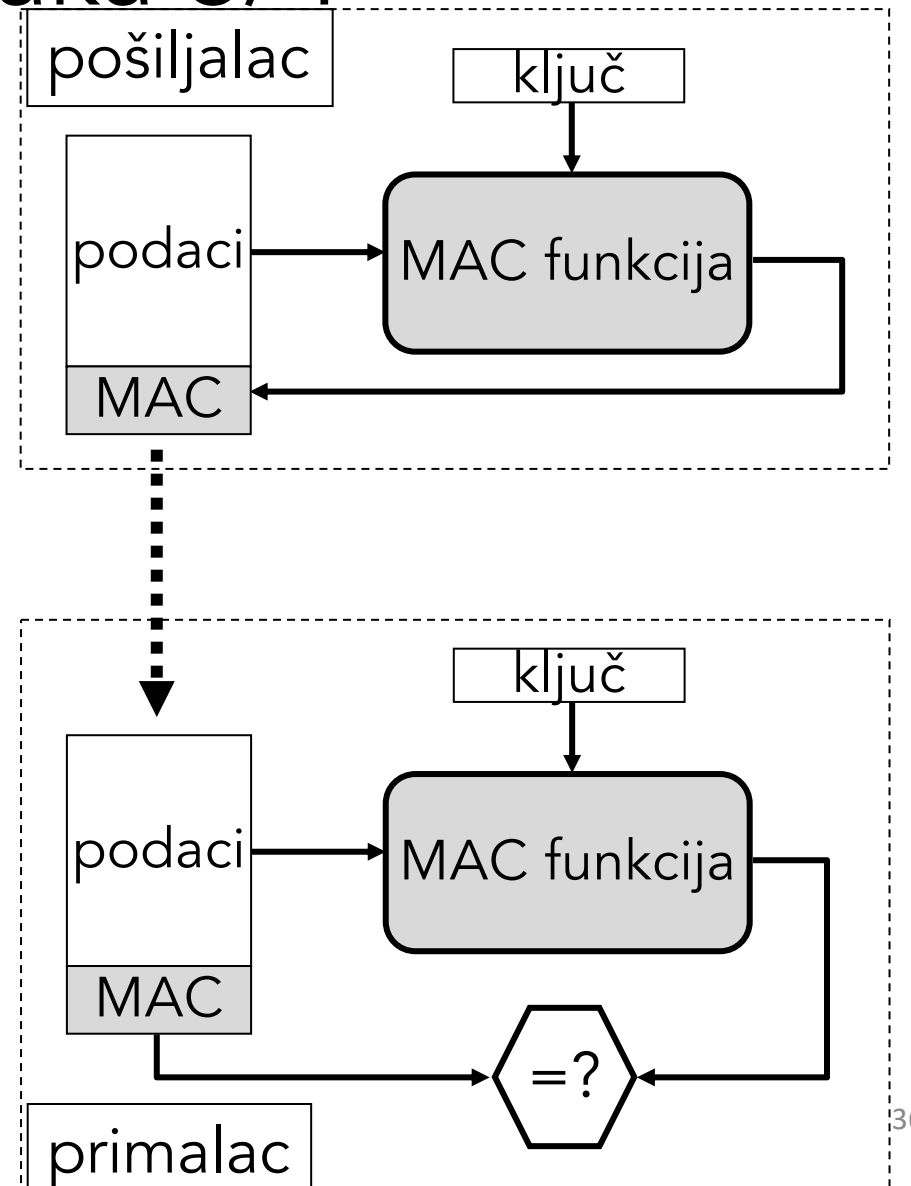


Integritet podataka 2/4

- Kriptografske heš funkcije mapiraju proizvoljno dugačke ulaze u izlaz fiksne dužine (engl. *digest*, heš vrednost)
- Heš funkcija treba da poseduje sledeće osobine:
  - Funkcija je deterministička, tako da ista poruka uvek rezultira istim hešom
  - Relativno brzo se izračuna heš vrednost za bilo koju datu poruku
  - Nemoguće je generisati poruku iz njene heš vrednosti osim isprobavanjem svih mogućih poruka
  - Mala promena u poruci bi može da rezultuje velikim promenama heš vrednosti
  - Ne bi smele da postoje dve različite poruke koje generišu isti heš

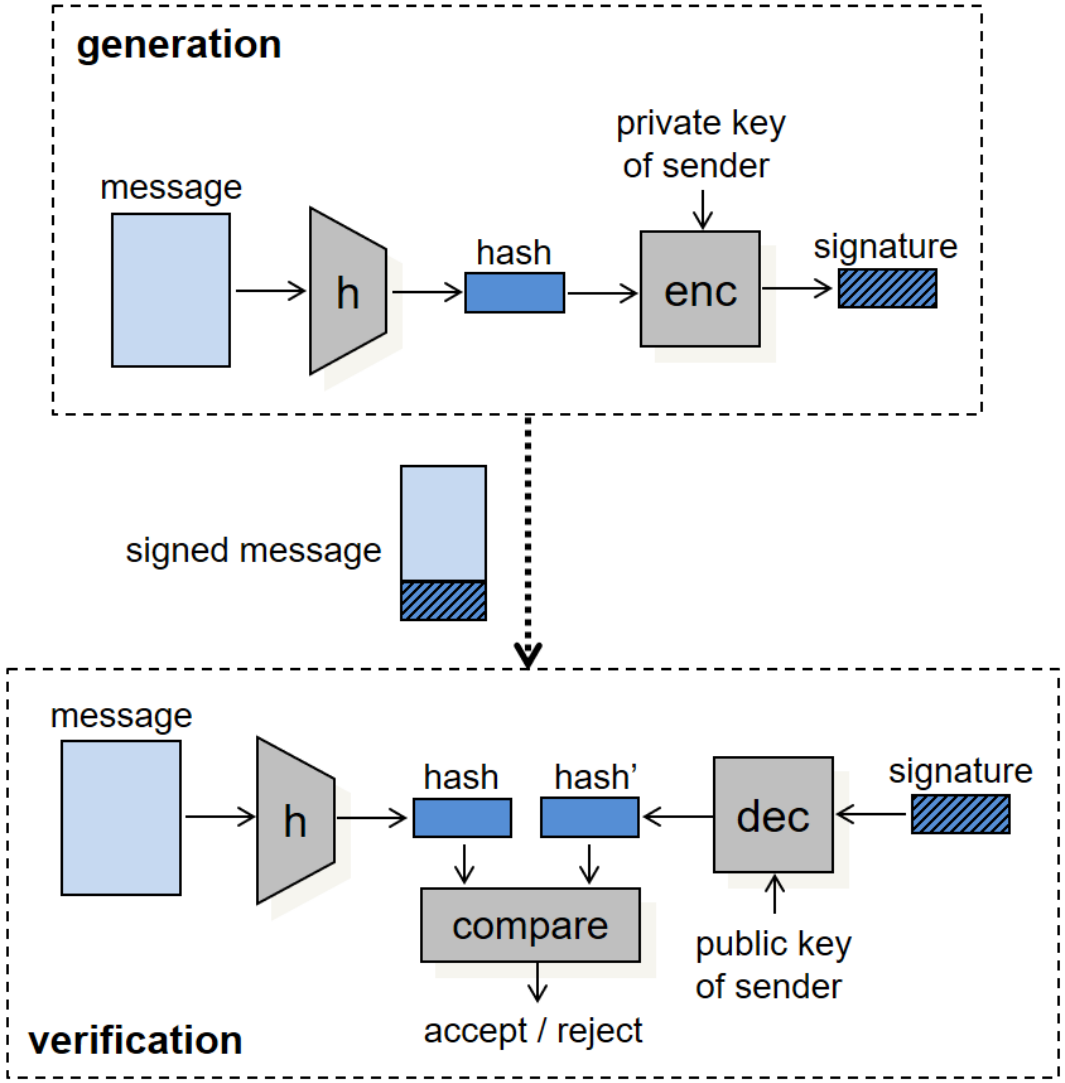
Integritet podataka 3/4

- MAC funkcije
  - Slične heš funkcijama, samo imaju dodatni parametar – simetrični ključ
  - Koriste se za zaštitu integriteta poruke i autentifikaciju porekla poruke



Integritet podataka 4/4

- Digitalni potpisi
  - Slično MAC funkcijama, ali koriste simetrične ključeve
  - Osim zaštite integriteta i autentifikacije porekla, oni takođe obezbeđuju neporecivost



Validacija ulaza 1/3

- Problem
  - Aplikacija prihvata podatke iz izvora izvan aplikacije (npr. veb korisnici, fajl sistem, spoljni servisi u vlasništvu trećih strana)
  - Ulazni podaci mogu uticati na izvršavanje interpretatora komandi (npr. SQL baza, XML parser, OS terminal)
  - Programeri često testiraju pozitivne scenarije, gde se obezbeđuje očekivani ulaz da bi se ustanovilo da li funkcija radi ispravno
  - Šta se dešava kada spoljni subjekt pošalje nasumične, glupe podatke? Šta se dešava kada se pošalje pažljivo osmišljen *command injection* napad?
  - "*Script kiddie*" želi da hakuje sistem brisanjem baze podataka

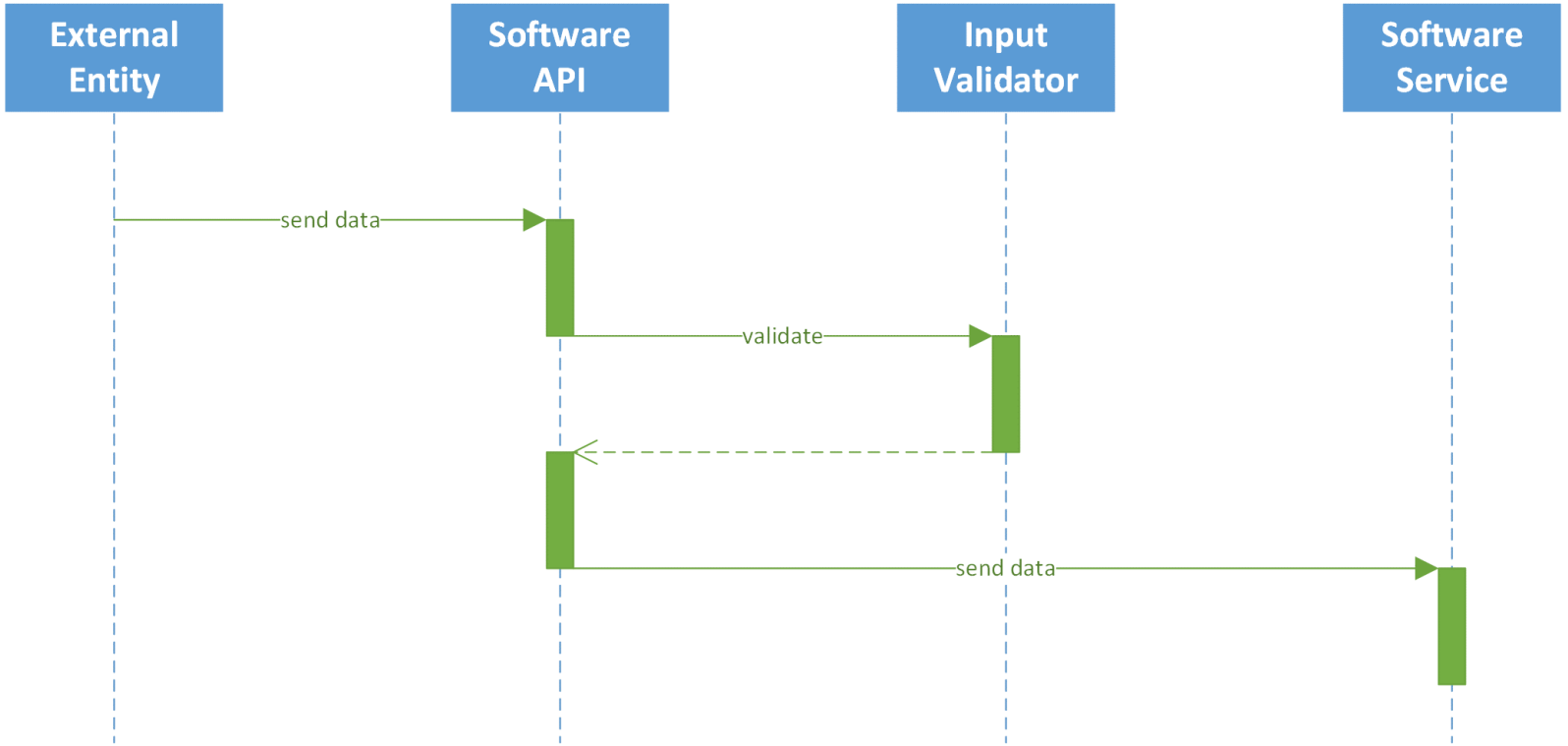
Validacija ulaza 2/3

- Rešenje
  - Testiranje bilo kog ulaza dobijenog od eksternog izvora (npr. korisnik, aplikacija)
  - Sprečavanje unosa nepravilno formiranih podataka u informacioni sistem
  - Ulazne podatke treba ispitati za obrasce koji su uvek istiniti (npr. polje obrasca je niz od najviše 20 znakova ili broj ne veći od 1000)
  - Standardni ulazni tipovi (npr. JSON, XML, SQL upit) imaju isprobane i testirane mehanizme za validaciju unosa (npr. XSD šeme, *prepared statements*)

Validacija ulaza 3/3

- Obrazac

  Kao i kod mnogih bezbednosnih kontrola, može se definisati kao aspekt u OO jezicima



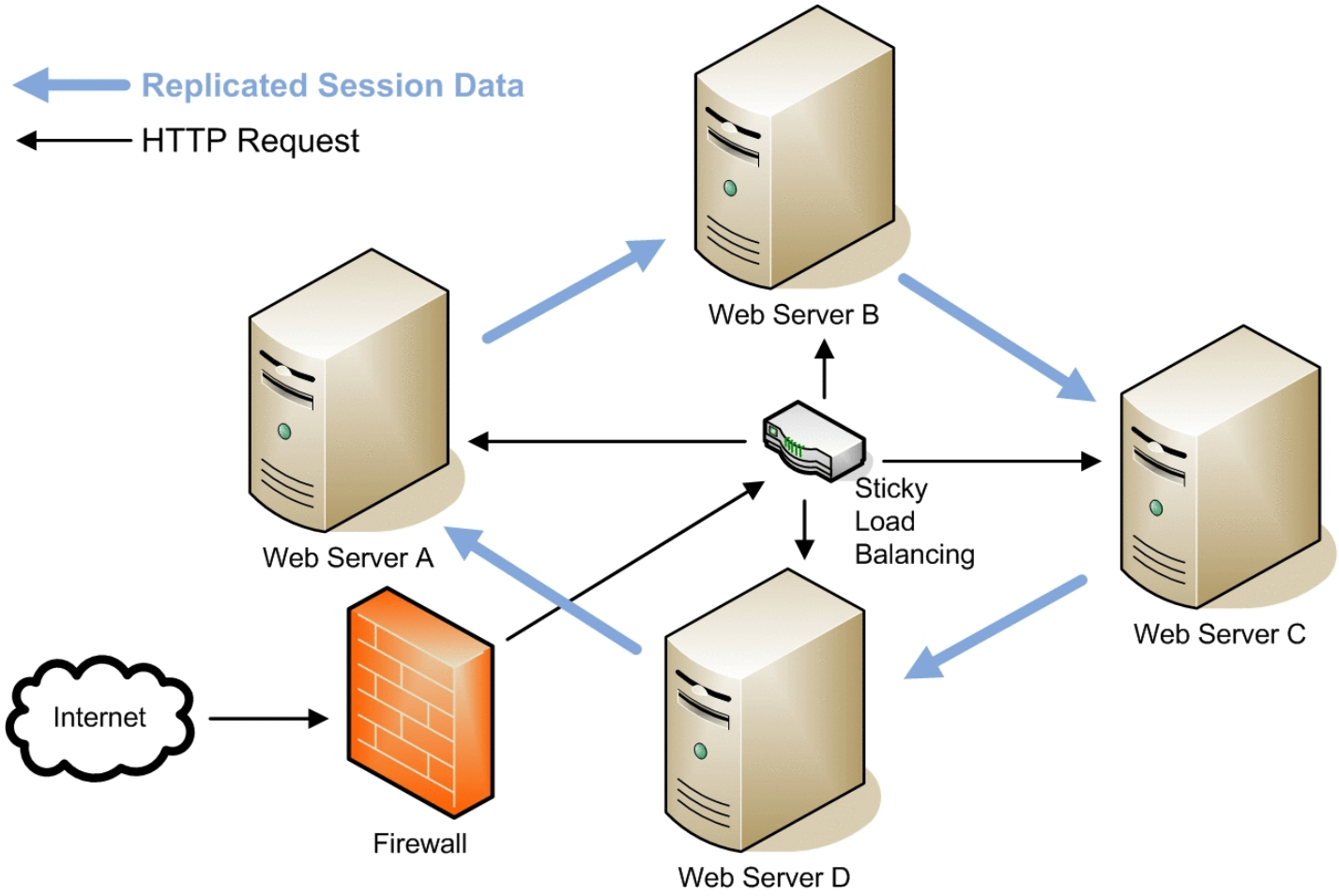
Dostupnost softvera 1/3

- Problem
  - Opterećenje aplikacije u razvoju je obično samo delić produkcionog opterećenja
  - DoS može relativno lako nastati time što se ne obezbedi dovoljno resursa za obradu redovnih zahtevima korisnika – šta bi se dogodilo ako bi Facebook pokrenuo samo jedan Tomcat?
  - Suparnička korporacija može angažovati botnet DDoS da onemogući dostupnosti naših servisa
  - Kvar hardvera ili prirodna katastrofa mogu uništiti server na kome se izvršavaju naši servisi

Dostupnost softvera 2/3

- Rešenje
  - Kako je bezbednost vođena zaštitom najslabije karike, tako je i dostupnost vođena otkrivanjem i rešavanjem uskih grla u performansama
  - Brojači performansi (engl. *performance counters*) pomažu programerima da otkriju delove aplikacije koji se često koriste i koji zahtevaju velike resurse
  - Redundantna postavka aplikacija (npr. klasteri sa balansom opterećenja) značajno povećava dostupnost sistema
  - Redundantna postavka (npr. *primary* i *disaster recovery* geografski odvojene lokacije) pomažu u održavanju dostupnosti sistema kada je čitava lokacija ugrožena (npr. zbog ekološke katastrofe, kao što je požar ili zemljotres)
  - Vatreni zidovi ☺ pomažu u filtriranju saobraćaja i mogu zaštititi od DDoS napada

Dostupnost softvera 3/3



Efikasna bezbedna razmena poruka

primer dizajna

- Alice šalje poruku Bobu, gde:
  - Nijedna treća strana ne može da pročita poruku
  - Bob zna da je poruka stigla od Alice
  - Alice ne može poreći da je poslala poruku
  - Poruka nema zlonamerni sadržaj
  - Koristi se minimalna količina propusnog opsega

1. Koje operacije mogu da se realizuju?
2. Kojim redosledom?
3. Šta su preduslovi?
4. Šta su pretpostavke o okruženju?

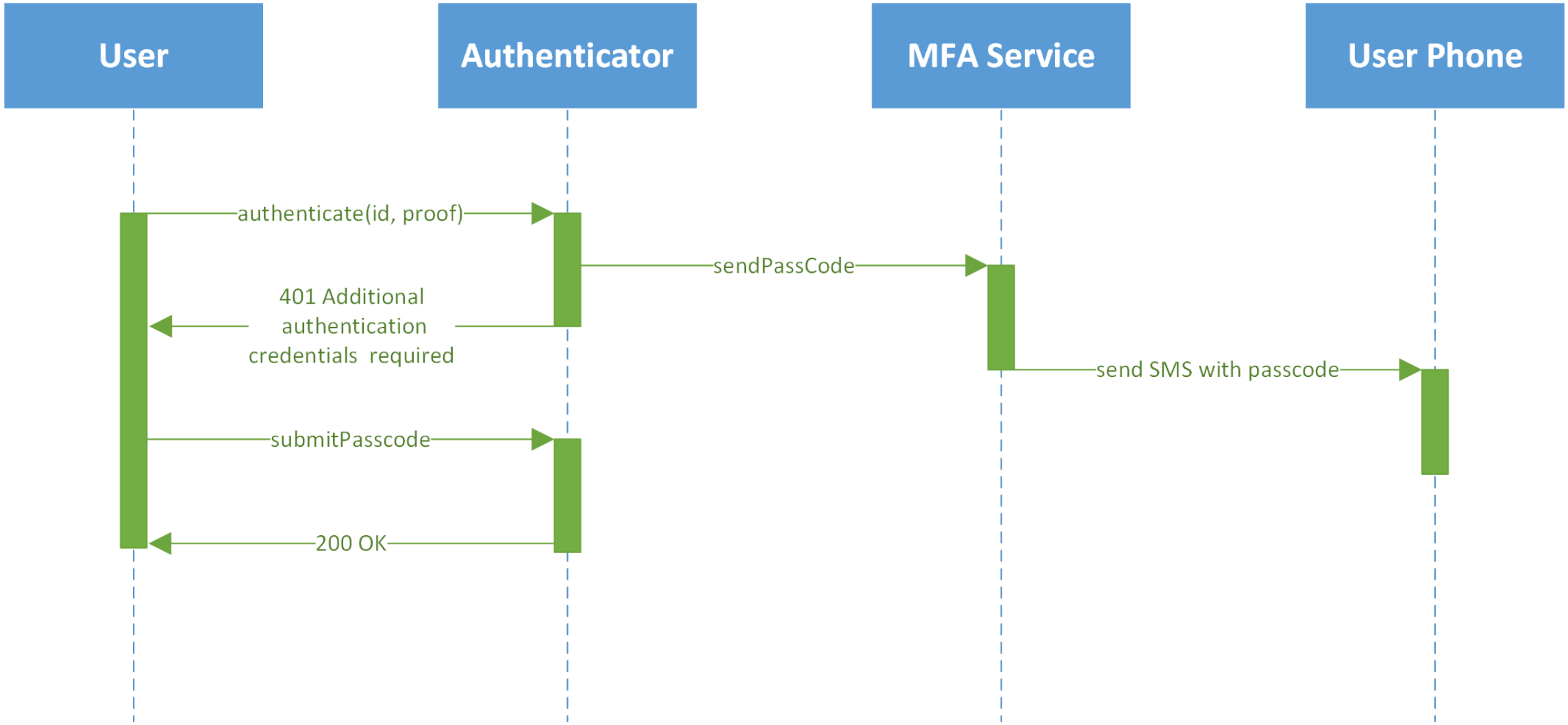
Autentifikacija 1/3

- Problem
  - Eksterni subjekat (npr. korisnik, servis) želi da pristupi podacima ili poziva neku funkciju aplikacije
  - Vlasnik aplikacije želi da kontroliše ko može da pristupi podacima i funkcijama
  - Aplikacija treba da podržava identifikaciju, pri čemu sadrži listu digitalnih identiteta za svakog registrovanog eksternog subjekta
  - Šta se dešava kada jedan eksterni subjekat tvrdi da je drugi, obezbeđivanjem odgovarajućeg ID-a?

Autentifikacija 2/3

- Rešenje
  - Autentifikacija je proces dokazivanja istinitosti tvrdnje – dokazivanje identiteta subjekta
    - "Nešto što korisnik zna" – lozinka, PIN
    - "Nešto što korisnik poseduje" – PKI, hard i soft tokeni
    - "Nešto što korisnik jeste" – biometrija
    - Multifaktorska autentifikacija

Autentifikacija 3/3



Autorizacija

- Problem
  - Autentifikovani subjekat (npr. korisnik, servis) želi da pristupi podacima ili pozove funkciju aplikacije
  - Aplikacija sadrži osetljive podatke i funkcije koje ne bi trebalo da budu dostupne svim korisnicima
  - Kontrolu pristupa treba sprovesti za sve subjekte i objekte na fleksibilan način, uzimajući u obzir da:
    - Subjekti i objekti se mogu dodavati, modifikovati ili ukloniti iz sistema
    - Administracija prava pristupa treba da bude fleksibilna i jednostavna

Jednostavna kontrola pristupa

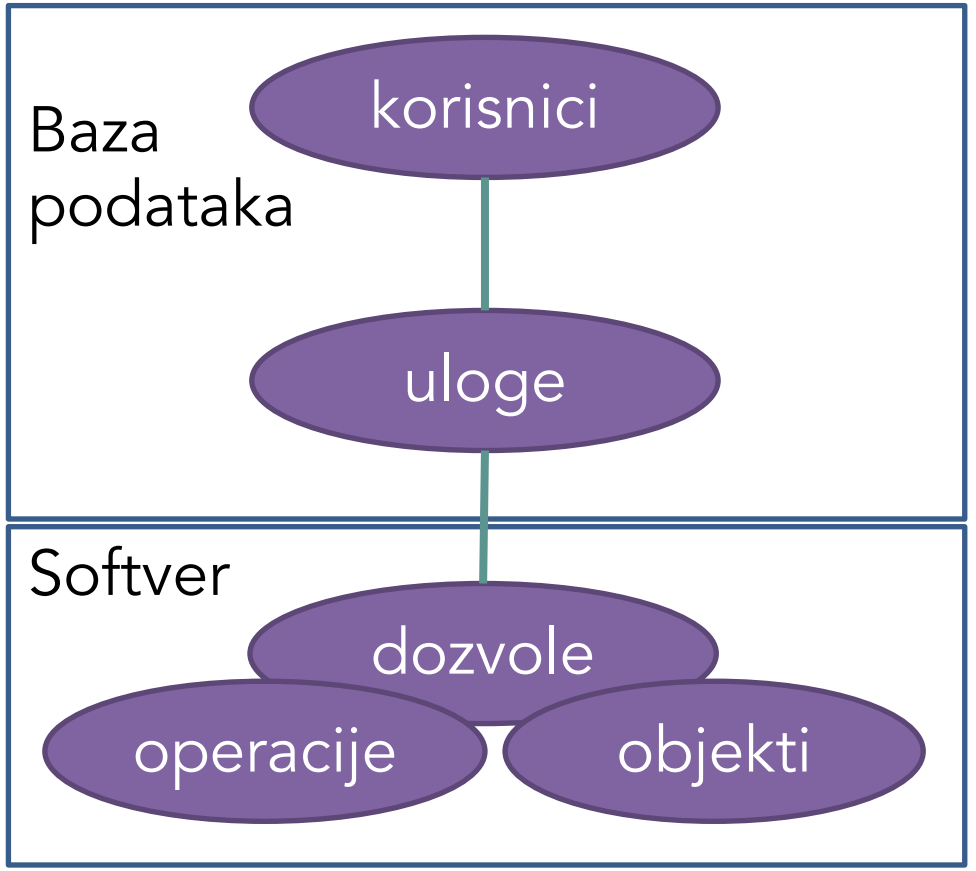
primer rešenja

- Sistem ima 5 objekata i 5 subjekata
- Sistem ima 5000 objekata i 5 subjekata
- Sistem ima 5 objekata i 5000 subjekata
- Sistem ima 5000 objekata i 5000 subjekata

- Koliko je teško promeniti prava pristupa?

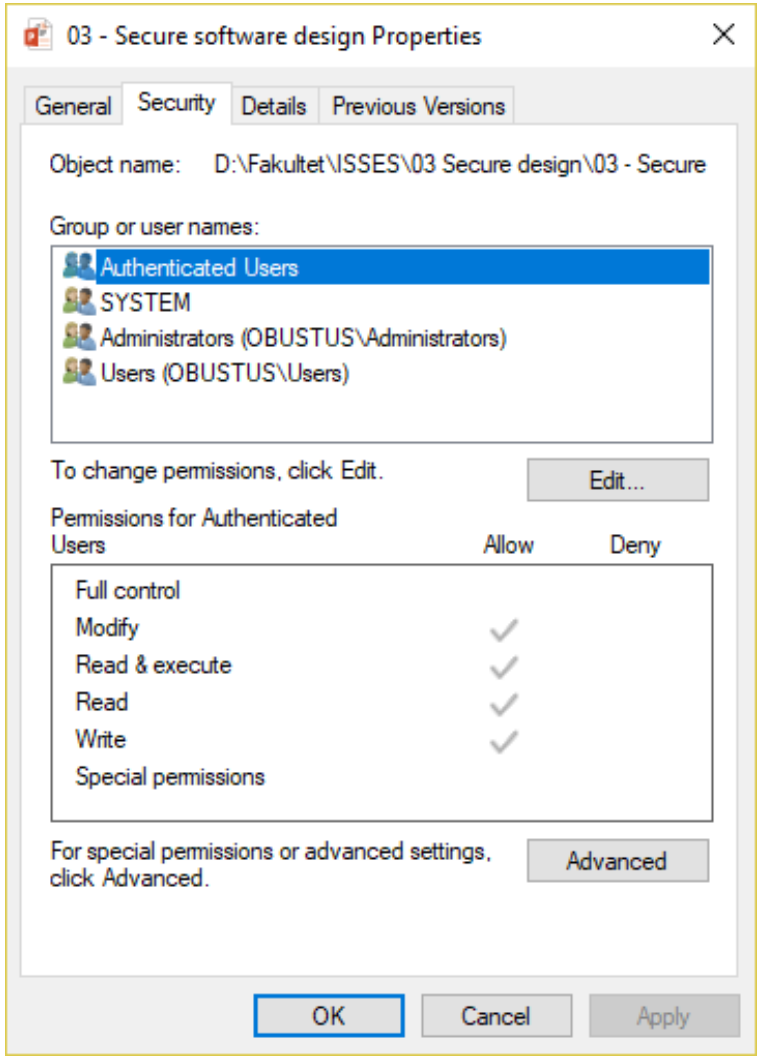
Role-Based Access Control

- Većina organizacija ima različite uloge – programer, QA, menadžer, administrator
- Svaka uloga podrazumeva različite odgovornosti – razvoj koda, izvršavanje testova, upravljanje zaostalim radovima, održavanje infrastrukture
- RBAC model je zasnovan na organizacionoj strukturi i nudi fleksibilnost za centralizovanu administraciju i sistem koji se menja



Access Control Lists

- RBAC model je pogodan kada se mogu definisati klase korisnika (tj. uloge) i objekata (poput API-ja, tabele baze podataka)
- Operativni sistem može imati neograničenbroj korisnika, od kojih svaki ima svoj fajl sistem
- U ovom slučaju RBAC nije baš zgodan
- ACL-ovi rešavaju ovo tako što distribuiraju prava pristupa objektima, komplikuju upravljanje, ali povećavaju efikasnost



Odgovornost (Accountability)

- Problem
  - Subjekti mogu tvrditi da se nešto dogodilo iako nije
  - Subjekti mogu tvrditi da se nešto nije dogodilo iako jeste
  - Poricanje (Repudiation) predstavlja ozbiljnu pretnju za svaki sistem
- Rešenje
  - Digitalni potpisi obezbeđuju neporecivost poruka
  - Šta da radimo kada je PKI preskup ili nepraktičan za implementaciju?
  - Sistem za upravljanje događajima (event logging system), kada je pravilno dizajniran, može ovo da reši

Mehanizmi logovanja

primer rešenja

- Kompletnost – Događaji sadrže dovoljno podataka i svi događaji za koje je potrebno neporicanje se evidentiraju
- Pouzdanost – Obezbediti dostupnost mehanizma i integrtet logova
- Tačnost – Svi događaji imaju precizno vreme nastajanja
- Upotrebljivost – Bezbednosni događaji se mog ulako identifikovati i izdvojiti
- Minimalizam – Realizovati minimalnu količinu logova koji će služiti svojoj svrsi

Privatnost po dizajnu (engl. *privacy by design, PbD*)

Privatnost po dizajnu (engl. *privacy by design*)

- Privatnost po dizajnu znači da je privatnost podrazumevano integrisana u proizvode, servise i dizajn sistema
- Zaštita podataka o klijentima postaje vodeći problem u korisničkom iskustvu, uzimajući isti nivo važnosti kao i funkcionalnost
- Principi privatnosti po dizajnu mogu se primeniti na celokupne procese, uključujući:
  - Dizajn sistema
  - Organizacione prioritete
  - Ciljeve projekta
  - Standarde i protokole
  - Poslovne prakse

Privatnost po dizajnu (engl. *privacy by design*)

- Privatnost podizajnu je holistički pristup privatnosti koji obuhvata 7 osnovnih principa:
  1. Proaktivno ne reaktivno; Preventivna, ne popravka
  2. Privatnost kao podrazumevana postavka
  3. Privatnost ugrađena u dizajn
  4. Puna funkcionalnost — pozitivna suma, ne nulta suma
  5. Bezbednost od kraja do kraja — Zaštita životnog ciklusa
  6. Vidljivost i transparentnost
  7. Poštovanje privatnosti korisnika

Proaktivno ne reaktivno

- PbD počinje sa eksplicitnim priznavanjem vrednosti i prednosti proaktivnog usvajanja jakih praksi privatnosti, rano i dosledno
  - Npr., sprečavanje (internih) *data breach*-ova od dešavanja
- Ovo implicira:
  - Jasnu posvećenost postavljanju i sprovođenju visokih standarda privatnosti – generalno viši nivo od standarda postavljenih globalnim zakonima i propisima
  - Uspostavljene proaktivnih metoda za prepoznavanje lošeg dizajna privatnosti, loše prakse privatnosti, i ispravku svih negativnih uticaja, mnogo pre nego što se oni pojave

Privatnost kao podrazumevana postavka

- Korisnici ne bi trebalo da brinu o svojim podešavanjima privatnosti kada pretražuju veb sajt ili otvaraju aplikaciju
- Ovaj princip automatski postavlja privatnost korisnika na najviši nivo zaštite, bez obzira da li korisnik stupa u interakciju sa tim podešavanjima ili ne

Privatnost kao podrazumevana postavka

- Takve podrazumevane postavke uključuju, između ostalog:
  - Specifikacija svrhe korišćenja podataka
    - Svrhe za koje se lični podaci prikupljaju, koriste, zadržavaju i otkrivaju treba da budu saopšteni korisniku (vlasniku podataka) u vreme ili pre vremena kada informacija se prikuplja
    - Navedene svrhe treba da budu jasne, ograničene i relevantne za okolnosti
  - Ograničenje prikupljanja podataka
    - Prikupljanje ličnih podataka mora biti pošteno, zakonito i ograničeno na samo ono što je neophodno za navedene svrhe

Privatnost kao podrazumevana postavka

- Takve podrazumevane postavke uključuju, između ostalog:
  - Minimizovanje podataka
    - Prikupljanje ličnih podataka treba da bude svedeno strogo na minimum
    - Dizajn aplikacija i sistema trebalo bi da počne, kao porazumevano, sa interakcijama i transakcijama koje ne mogu identifikovati korisnika
    - Gde god je moguće, identifikaciju, vidljivost i povezivost ličnih podataka treba svesti na minimum
  - Ograničenje upotrebe, zadržavanja i otkrivanja
    - Korišćenje, zadržavanje i otkrivanje ličnih informacije treba da budu ograničene na relevantne svrhe objašnjene korisniku, za koje je korisnik dao saglasnost, osim ako je drugačije propisano zakonom
    - Lični podaci se čuvaju samo kao onoliko koliko je potrebno da se ispune navedene svrhe, a zatim bezbedno uništavaju

Privatnost ugrađena u dizajn

- Privatnost mora biti ugrađena u tehnologije, operacije i arhitekture na holistički, uniforman i kreativan način:
  - **Holistički**, jer se uvek moraju uzeti u obzir dodatni, širi konteksti
  - **Uniformno**, jer treba konsultovati sve različite zainteresovane strane i interese
  - **Kreativno**, jer ugrađivanje privatnosti ponekad znači ponovno izmišljanje postojećih načina jer su alternative neprihvatljive

Privatnost ugrađena u dizajn

- Treba usvojiti sistemski pristup ugrađivanju privatnosti – koji se oslanja na prihvaćene standarde i smernice, koji su podložni eksternim revizijama
- Sve informacione prakse treba da se primenjuju sa jednakom rigoroznošću, na svakom koraku u projektovanju i radu
- Kad god je moguće, treba izvršiti i objaviti detaljne procene uticaja na privatnost i rizika, jasno dokumentujući rizike privatnosti i sve mere preduzete za ublažavanje tih rizika, uključujući razmatranje alternativa i izbor metrike
- Uticaji na privatnost rezultirajuće tehnologije, operacije ili arhitekture i njihove upotrebe, treba da bude svedeno na minimum, a ne da se lako degradira upotrebom, pogrešnom konfiguracijom ili greškom

Puna funkcionalnost — pozitivna suma, ne nulta suma

- PbD ne uključuje samo pravljenje obaveza na polju privatnosti već se odnosi na zadovoljavanje svih legitimnih ciljeva
- Kada se privatnost ugrađuje u datu tehnologiju, proces ili sistem, to treba učiniti na takav način da nije narušena puna funkcionalnost i u najvećoj mogućoj meri da su svi zahtevi optimizovani
- Privatnost se često shvata na način "nulte sume", tj. kao da mora da se takmiči sa drugim legitimnim zahtevima, dizajnom i tehničkim mogućnostima

Puna funkcionalnost — pozitivna suma, ne nulta suma

- PbD ne specificira takav pristup već obuhvata legitimne ciljeve koji nisu vezani za privatnost i prilagođava ih, u inovativan način "pozitivne sume"
- Svi interesi i ciljevi moraju biti jasno dokumentovani, željene funkcije artikulisane, metrike dogovorene i primenjene, a kompromisi odbačeni kao često nepotrebni, u korist pronalaženja rešenja što za rezultat omogućava funkcionalnost

Bezbednost od kraja do kraja — Zaštita životnog ciklusa

- Privatnost mora biti kontinuirano zaštićena u celom domenu i tokom celo životnog ciklusa obitavanja podataka u sistemu
- Ne bi trebalo da postoje praznine ni u zaštiti ni u odgovornosti
- Princip „bezbednosti“ je ovde posebno važan jer bez jake bezbednosti ne može biti privatnosti

Bezbednost od kraja do kraja — Zaštita životnog ciklusa

- Subjekti moraju preuzeti odgovornost za bezbednost ličnih podataka (generalno srazmerno stepenu osetljivosti) tokom celog životnog ciklusa, u skladu sa standardima koje su razvila priznata tela za razvoj standarda
- Primenjeni bezbednosni standardi moraju da obezbede poverljivost, integritet i dostupnost ličnih podataka tokom svog životnog ciklusa uključujući i metode bezbednog uništenja, odgovarajuće šifrovanje, i jake metode kontrole pristupa i logovanja

Vidljivost i transparentnost

- Vidljivost i transparentnost su od suštinskog značaja za uspostavljanje odgovornosti i poverenja
- Odgovornost – Prikupljanje ličnih podataka podrazumeva obavezu brige o njihovoj zaštiti
- Odgovornost za sve polise i procedure u vezi sa privatnošću moraju da budu dokumentovana i saopštena prema potrebi i dodeljena određenoj osobi
- Prilikom prenosa ličnih podataka trećeim stranama, mora biti obezbeđena ekvivalentna zaštita privatnosti putem ugovora ili na drugi način

Vidljivost i transparentnost

- Otvorenost – Otvorenost i transparentnost su ključni za odgovornost
- Informacije o polisama i prakse koje se odnose na upravljanje ličnim informacijama moraju da budu lako dostupne korisnicima
- Usklađenost – Trebalo bi uspostaviti mehanizme za žalbe i obeštećenje i te informacije komunicirati korisnicima
- Usklađenosti sa polisama i procedurama privatnosti treba pratiti i revidirati

Poštovanje privatnosti korisnika

- Najbolji rezultati PbD su obično oni koji su svesno osmišljeni oko interesa i potrebe pojedinačnih korisnika, koji imaju najveći interes u upravljanju sopstvenim ličnim podacima
- Ohrabrivanje vlasnika podataka da igraju aktivnu ulogu u upravljanju sopstvenim podacima može biti najefikasnija provera protiv zloupotreba ličnih podataka

Poštovanje privatnosti korisnika

- Saglasnost – potrebna je besplatna i posebna saglasnost korisnika za prikupljanje, korišćenje ili otkrivanje ličnih podataka, osim kada je drugačije dozvoljeno zakonom. Što je veća osetljivost podataka, to je jasniji i konkretniji zahtev za saglasnošću. Saglasnost se može povući kasnije.
- Tačnost – lični podaci trebaju biti onoliko tačni, potpuni i ažurni koliko je potrebno ispuniti navedene svrhe

Poštovanje privatnosti korisnika

- Pristup – korisnicima treba biti omogućen pristup svojim ličnim podacima i oni treba da budu obavešteni ako njihove podatke neko koristi. Korisnici mogu da osporavaju tačnost i potpunost informacija i da ih po potrebi izmene.
- Usklađenost – Organizacije moraju uspostaviti mehanizme za žalbe i obeštećenje, i saopštavaju javnosti informacije o njima, uključujući i način pristupa sledećem nivou žalbe

Zaključak

- Dizjaniranje bezbednosti od početka razvoja softvera je znato jeftinije i efikasnije od "dodavanja" (*bolting on*) bezbednosnih kontrola kada je softver završen
- Rano razmišljanje o bezbednosti je važno za efikasno planiranje projekta
- Razvoj bezbednog dizajna zahteva razumevanje i istraživanje obrazaca bezbednosnog dizajna i primenu principa bezbednosnog dizajna na svakom koraku

Reference

- Fernandez-Buglioni E. Security Patterns in Practice: Designing Secure Architectures Using Software Patterns. Wiley 2013.
- Tarandach I., J. Coles M. J. Threat Modeling – A Practical Guide for Development Teams. O'Reilly 2021.
- Saltzer J. H., Schroeder M. D. The Protection of Information in Computer Systems. https://web.mit.edu/Saltzer/www/publications/protection/
- Black Hat Briefings. Security Design Patterns. https://www.blackhat.com/presentations/bh-federal-03/bh-fed-03-peterson-up.pdf
- Cavoukian A. Privacy by Design – The 7 Foundational Principles. Implementation and Mapping of Fair Information Practices. https://privacy.ucsc.edu/resources/privacy-by-design---foundational-principles.pdf