SRS Međuispit

Osnove sigurnosti

Definicija sigurnosti

Sigurnost je kontinuirani proces čijim provođenjem se osigurava određeno stanje (sustava, podataka/informacija). Željeno stanje je definirano zahtjevima.

Ako neki od zahtjeva nije ispunjen, kažemo da se desio **incident**, odnosno, da je **narušena sigurnost**.

Temeljni sigurnosni zahtjevi

1. **Povjerljivost (engl. confidentiality), tajnost (engl.secrecy)**
    - Podaci/informacije moraju biti dostupne samo ovlaštenim entitetima
2. **Cjelovitost, Integritet (engl. integrity)**
    - Jamstvo da su podaci/informacije poslane, primljene ili pohranjene u izvornom i nepromijenjenom obliku
3. **Raspoloživost (engl. availability)**
    - Informacije moraju biti raspoložive, a sustavi i usluge u operativnom stanju, usprkos mogućim neočekivanim i nepredvidljivim događajima

Dodatni sigurnosni zahtjevi

1. **Autentičnost (engl. authenticity)**
    - Potvrda identiteta korisnika; ovjera vjerodostojnosti (autentifikacija) sudionika komunikacije; ovjera izvora podataka
2. **Neporecivost (engl. non-repudiation)**
    - Sudionici ne mogu poreći akciju u kojoj su sudjelovali, npr. nemogućnost naknadnog odricanja slanja, odnosno primanja, poruke

Kako bi se desio **incident** (narušila sigurnost) moraju postojati dva preduvjeta: ranjivost i prijetnja

- **Ranjivost (engl. vulnerability)** je pogreška ili slabost u dizajnu sustava, implementaciji, upotrebi ili upravljanju koja se može iskoristiti za narušavanje sigurnosti sustava ili informacije.
- **Prijetnja (engl. threat)** je bilo koja okolnost ili događaj koji ima potencijal narušiti sigurnost sustava ili informacije
- **Napad** je realizacija namjerne prijetnje

Podjela sigurnosti

1. **Podjela na ofenzivnu i defenzivnu**
    - Znaju dijeliti iste alate, ali naravno za različite svrhe

2. **Podjela na tehničku, taktičku, operativnu i stratešku**
    - Tehnička – direktno vezano uz tehničke aspekte
    - Taktička – povezivanje više tehničkih aspekata u jednu cjelinu
    - Operativna – povezivanje više taktičkih aspekata u jednu cjelinu
    - Strateška – definira smisao i svrhu, iz nje proizlazi operativna razina

Stražnja vrata u sustave (engl. backdoor)

Stražnja vrata su svojstvo sustava da omogućava pristup nekome tko ne bi smio imati pristup.

Prikriveni kanali (engl. covert channel)

Komunikacijski kanal kojim se prenose podaci, a da toga nisu svjesni vlasnici ili legitimni korisnici sustava u kojemu se kanal javlja.

Sporedni kanal (engl. side channel)

Komunikacijski kanal kroz koji prolaze dodatne informacije koje ne želimo otkriti napadaču.

Osnove kriptografije i kriptoanalize

Kerckhoffov princip

Kriptosustav mora biti siguran čak i kada su javno poznati svi detalji rada sustava osim samih ključeva.

Savršena povjerljivost

Šifra pruža savršenu povjerljivost ako je za svakog napadača šansa da pogodi poruku jednaka $1/|M|$.

Simetrične šifre

Simetrična šifra je par algoritama E i D (E: $M \times K \rightarrow C$, D: $C \times K \rightarrow M$) gdje za svaki $k \in K$ i $m \in M$ vrijedi $D(E(m, k), k) = m$.

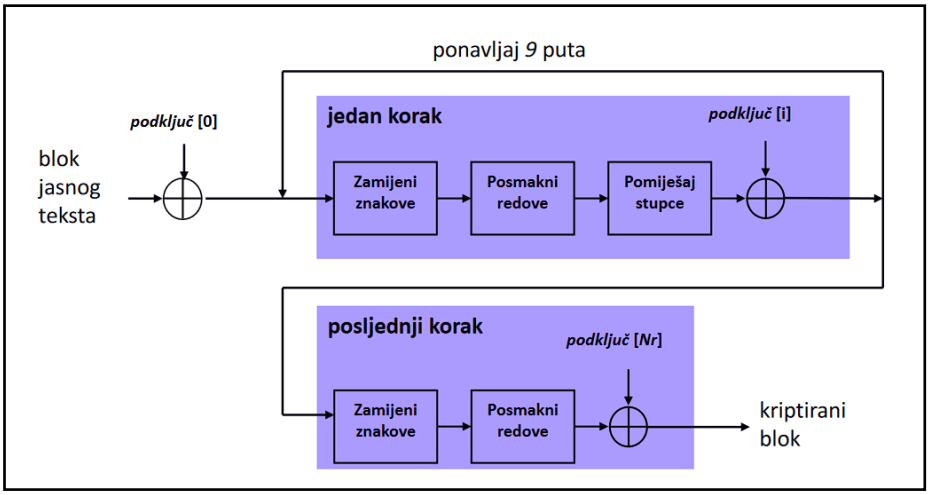
Jednokratna bilježnica

- Enkripcija i dekripcija se vrše tako da se napravi XOR između poruke/šifrata i ključa
- $M = K = C = \{0, 1\}^n$
- Osigurava savršenu povjerljivost

- Nedostatci: ključ mora biti iste duljine kao i poruka i smije koristiti najviše jednom, ne štiti integritet poruke

Blok šifra

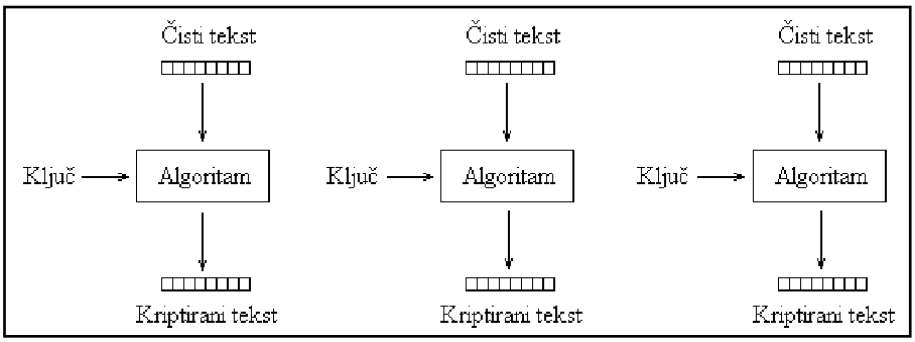
- $M = C = \{0, 1\}^n$
- $K = \{0, 1\}^k$
- E i D deterministički algoritmi
- Primjeri blok šifri: DES, 3DES, IDEA, Blowfish, AES
- **AES**
    - Veličina bloka: 128 bitova
    - Veličine ključa: 128, 192 ili 256 bitova
    - Postoji sklopovska potpora algoritmu (naredbe aesenc, aesenclast)



Slika 1. AES postupak

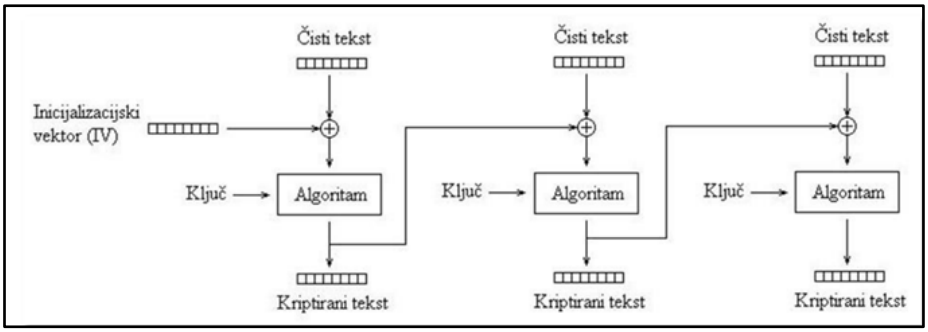
Ostali načini šifriranja:

ECB – Electronic Codebook

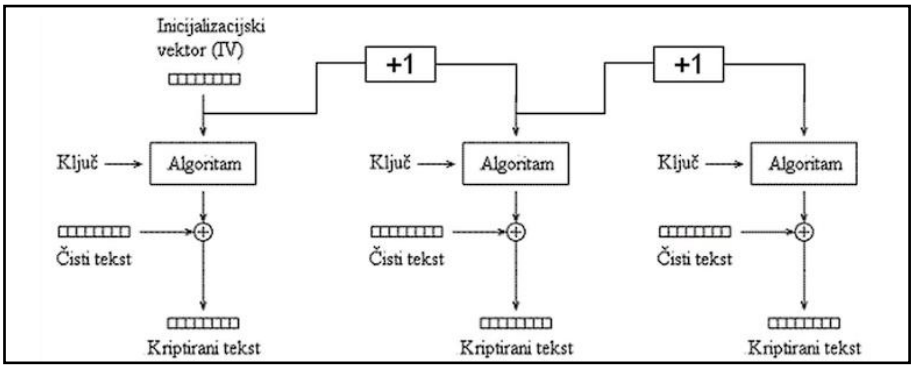


Slika 2. ECB postupak

CBC – Cipher Block Chaining



CTR – Counter Mode



Slika 3. CBC postupak

Protočna šifra

- Generator pseudoslučajnih brojeva na temelju ključa generira niz bitova koji se XOR-a s izvornim tekstom
- Primjeri: RC4, CSS, Salsa20/ChaCha (ključ 128 ili 256 bitova, alternativa AES-u zbog boljih performansi)

Kriptoanaliza blok šifri

- Apsolutne dokaze sigurnosti simetričnih šifri nemamo
- Efektivna veličina ključa je b ako najbolji poznati napad radi red veličine 2^b koraka
- Vrste kriptoanalize:
    - Gruba sila
    - Linearna kriptoanaliza - iskorištava linearne zavisnosti pojedinih bitova poruke, ključa i šifrata
    - Diferencijalna kriptoanaliza - analiza kako promjene poruke utječu na promjene šifrata

Kriptografske funkcije sažetka

H je deterministički algoritam H: {0,1}* → {0,1}^{n} koji proizvoljnoj poruci pridružuje sažetak fiksne duljine.

Kriptografska funkcija sažetka H je otporna na kolizije ako je praktički nemoguće pronaći dvije različite poruke x i y takve da vrijedi $H(x) = H(y)$.

Koriste se za: integritet poruka, zaštitu zaporki, deriviranje ključeva iz zaporki, generiranje pseudoslučajnih brojeva, digitalne potpise.

Kod za integritet poruke - MAC

M je deterministički algoritam M: {0, 1}* × {0, 1}^{k} → {0, 1}^{n} koji proizvoljnoj poruci i ključu pridružuje oznaku fiksne duljine.

Vrste:

1. **HMAC** – MAC pomoću kriptografskih hash funkcija
2. **CBC MAC** – MAC pomoću blok šifre

Osiguravanje povjerljivosti i integriteta

- Encrypt-and-MAC: $E(m, k1)$, $M(m, k2)$ nesiguran
- MAC-then-Encrypt: $E(m \,||\, M(m, k2)$, $k1)$ nesiguran
- **Encrypt-then-MAC: c = $E(m, k1)$, $M(c, k2)$ siguran!**

Autentificirana šifra

Pruža svojstva povjerljivosti i integriteta u jednom paketu (AES-GCM).

Preporuke

- NE koristiti način kriptiranja ECB
- IV i salt generirati slučajno
- NE koristiti stalno isti simetrični ključ

Asimetrične šifre

Primatelj ima dva ključa:

- Javni ključ pk: Javno poznat
- Privatni ključ sk: Poznat samo Branku

Jasni tekst se šifrira s javnim ključem pk, skriveni tekst se dešifrira s privatnim ključem sk.

Za svaki par ključeva (pk, sk) generiranih preko G i za svaku poruku p vrijedi $D(E(p, pk), sk) = p$.

- G – algoritam koji generira par ključeva pk, sk
- $E(m, pk)$ – algoritam enkripcije
- $D(c, sk)$ – algoritam dekripcije

Primjeri sustava kriptiranjem javnim kljucem: McEliece, ElGamal, RSA

RSA

Teorija brojeva

Notacija

- N – prirodni broj
- p, q – prosti brojevi
- $\mathbb{Z}_N = \{0, 1, \dots, N-1\}$ – prsten u kojemu se zbraja, oduzima i množi modulo N
- Inverz elementa $x \in \mathbb{Z}_N$ je element $y \in \mathbb{Z}_N$ takav da vrijedi $x \cdot y = 1$ u $\mathbb{Z}_N$
- Broj x ima inverz u $\mathbb{Z}_N$ ako i samo ako je $nzd(x, N) = 1$
- $\mathbb{Z}_N^*$ je skup svih invertibilnih elementa
- Eulerova funkcija $\varphi(N) = |\mathbb{Z}_N^*|$
    - Ako je p prost onda $\varphi(p) = p - 1$
    - Ako su p i q različiti prosti brojevi onda je $\varphi(pq) = (p-1)(q-1)$

RSA - generiranje ključeva !

Algoritam G:

1. Odaberem velike slučajne proste brojeve p i q

2. Izračunam $N = p \cdot q$

3. Izračunam $\varphi(N) = (p-1)(q-1)$

4. Odaberem proizvoljni $e \in \mathbb{Z}_{\varphi(N)}^*$ (u praksi $e = 65537$)

5. Izračunam $d = e^{-1}$ u $\mathbb{Z}_{\varphi(N)}^*$

6. Javni ključ: $pk = (e, N)$

7. Privatni ključ: $sk = (d, N)$

Ako je moguće N efikasno rastaviti na faktore onda je RSA nesiguran.

Ako je moguće efikasno izračunati $\varphi(N)$ onda je RSA nesiguran.

RSA – enkripcija i dekripcija

Algoritam E:

- $E(m, (e, N)) = m^e$ u $\mathbb{Z}_N$
- e je javni eksponent

Algoritam D:

- $D(c, (d, N)) = c^d$ u $\mathbb{Z}_N$
- D je privatni eksponent

Sigurnost

Obični RSA nije siguran sustav kriptiranja javnim ključem.

- Niti od napada poznatim izvornim tekstom.
- Niti od napada odabranim tekstom.

RSA – kombinacija sa simetričnom šifrom

U praksi se RSA gotovo nikada ne koristi za kriptiranje podataka već za kriptiranje ključeva ili materijala za ključeve.

- 1.način: Digitalna omotnica: $E(Pad(k), pk), E_S(m, k)$
- 2.način: Kriptiranje materijala za ključ

H je hash funkcija, E_S simetrična šifra

Algoritam E:

1. Izaberem slučajni $x \in \mathbb{Z}_N$
2. Izračunam $k = H(x)$
3. Izračunam $c1 = E(x, pk)$
4. Izračunam $c2 = E_S(m, k)$
5. Skriveni tekst je $(c1, c2)$

Ako se RSA ispravno koristi smatramo ga sigurnim.

Digitalni potpisi

Pošiljatelj generira potpis svojim privatnim ključem skA.

Primatelj provjerava potpis pošiljatelja javnim ključem pkA.

Autentičnost - Svatko može provjeriti ispravnost digitalnog potpisa ako ima na raspolaganju javni ključ potpisnika

Trojka efikasnih algoritama G, S i V

- G – algoritam koji generira par ključeva pk, sk
- $S(m, sk)$ – algoritam potpisivanja
- $V(m, \sigma, pk)$ – algoritam verifikacije

Primjeri sustava digitalnog potpisa: McEliece, DSA, RSA

Digitalni potpis nije enkripcija sažetka poruke privatnim ključem!

RSA digitalni potpis

H – kriptografska funkcija sažetka

Pad – funkcija nadopunjavanja

Algoritam S:

• $S(m, (d, N)) = Pad(H(m))^d$ u $\mathbb{Z}N$

Algoritam V:

• $V(m, \sigma, (e, N)) = (Unpad(\sigma^e \text{ u } \mathbb{Z}_N) == H(m))$? 1 : 0

Ranjivosti

Ranjivost je moguće uvesti u bilo kojoj fazi životnog ciklusa sustava.

Životni ciklus programskih sustava sastoji se od sljedećih faza:

1. Dizajn
2. Implementacija
3. Uvođenje u upotrebu, upravljanje, održavanje
4. Uklanjanje

Ranjivosti u dizajnu sustava

- Tijekom dizajna definiraju se ključne karakteristike sustava
- Najčešće se specificira kroz arhitekturu sustava i načela izgradnje sustava
- Pogreške, odnosno slabosti, nastaju ako se ne predvidi ugrađivanje odgovarajućih sigurnosnih zaštitnih mehanizama
- Za sprečavanje je potrebno od inicijalnog trenutka voditi računa o sigurnosti
- Temeljni mehanizam za ispravno dizajniranje sustava je **modeliranje prijetnji (engl. Threat modeling)**

    - Zadaća modeliranja prijetnji je utvrditi što prijeti sustavu i od čega se štitimo
- **Otkrivanje ranjivosti provođenjem analize sustava**

Ranjivosti u implementaciji

- U ovoj fazi ranjivosti su podskup programskih pogrešaka (bugs).
- Svaki bug nije ranjivost, ali svaka ranjivost jest bug.
- Posebna kategorija su **ranjivosti nultog dana (engl. zero day vulnerability)**
    - Otkrivene ranjivosti za koje ne zna nitko osim onoga tko ih je otkrio
- **Sprečavanje ranjivosti**
    - Edukacija programera
    - Testiranje koda
    - Revizija koda (barem jedan drugi programer provjerava kod)
    - Statička analiza koda
    - Dinamička analiza koda
    - Formalne metode
- Liste najčešćih ranjivosti:
    - OWASP Top 10
    - CWE Top 25

Ranjivosti u upotrebi

- Tijekom uporabe mogu nastati zbog pogrešaka proizvođača ili neispravnog korištenja sustava
- Pogreške proizvođača tek u ovoj fazi postaju vidljive
- **Sprečavanje ranjivosti**
    - Temeljni način otkrivanja pogrešaka je pregledavanje sustava (Ručno pregledavanje i testiranje ili pomoću alata)
    - Sprečavanje pogrešaka i slabosti u upotrebi (korištenje alata koji provjeravaju konfiguracije te upozoravaju na potencijalne manjkavosti)
- **Otkrivanje ranjivosti**
    - Ručno - može otkriti i nove, do sada nepoznate, ranjivosti
    - Automatizirano - potencijalno puno lažno pozitivnih i lažno negativnih, može otkriti samo ono što je algoritamski „pronalazivo” i isprogramirano
- Implementacijske ranjivosti u upotrebi
    - Kada se ranjivost otkrije potencijalno su ranjive sve implementacije u upotrebi
    - Dominantan način ispravljanja je korištenjem **zakrpa (eng. patch)**
    - Proizvođači programske podrške također izdaju upozorenja
    - Ako zakrpa nije odmah dostupna proizvođači daju **privremene mjere zaštite (engl. workaround)**
- Način iskorištavanja ranjivosti
    - Metoda, kod ili nekakav drugi artefakt koji iskorištava ranjivost
    - Shellcod - kratki kod koji iskorištava ranjivost i potom pokreće nešto drugo
- Baza ranjivosti CVE
    - Common Vulnerability Enumeration (CVE) je često korišten način označavanja i katalogiziranja ranjivosti

- Metoda izračuna ozbiljnosti ranjivosti
    - Common Vulnerability Scoring System (CVSS)
    - Raspon mjere je od 0 do 10 u koracima 0.1
    - U izračunu CVSS-a uzimaju se u obzir tri komponente: Bazna komponenta, vremenska komponenta, okruženje

Ranjivosti u upravljanju i uklanjanju

- Ovo su pogreške koje nastaju zbog upravljačkih ili administrativnih propusta
- Ove ranjivosti sprečavaju se definiranjem odgovarajućih politika i procedura
- Kada se sustavi uklanjaju treba paziti na podatke koji se na njima nalaze
- Primjer ranjivosti: skeneri/printeri imaju diskove na kojima se nalaze podaci

Zloćudni kod

Zloćudna funkcionalnost (engl. malicious logic) je sklopovlje, firmware ili programska podrška koja je namjerno uključena ili ubačena u sustav radi štetnih ciljeva.

Zloćudni kod (engl. malware) je značajan mehanizam djelovanja raznih napadača.

Klasifikacija zloćudnog koda

- Način širenja - na koji način dolaze do računala žrtve
- Način pokretanja - na koji način započinje njihovo izvršavanje
- Monolitni ili modularni
- Platforma - kako/gdje se izvršava zloćudni kod
- Perzistencija
- Način prikrivanja od korisnika
- Zloćudna funkcionalnost

Česti spominjan zloćudni kod

- Virusi
    - Temeljna karakteristika je da se šire tako što se ubacuju u izvršne kodove legitimnih programa te se pokreću njihovim pokretanjem
- Crvi
    - Temeljna karakteristika je da se mogu samostalno širiti putem mreže (ranjivi servisi, dijeljeni diskovi)
- Downloader
    - Zloćudni kod koji skida i instalira neki drugi zloćudni kod
- Dropper
    - Zloćudni kod koji sadrži drugi zloćudni kod te ga postavlja na kompromitirano računalo
- Logička bomba (engl. logic bomb)
    - Zloćudni kod koji obavlja nekakvu zloćudnu aktivnost kada se ispune određeni uvijeti

- Špijunski zloćudni kod (engl. spyware)
    - Zloćudni kod koji na neki način izvlači podatke korisnika računala
- Alat za udaljen pristup (engl. remote access tool)
    - Nije nužno zloćudni kod, primjerice downloader može instalirati TeamViewer
- Trojanci (engl. Trojan horse)
    - Temeljna karakteristika je da se pretvaraju kako obavljaju neku korisnu funkciju dok u biti sadrže maliciozni teret
- Ucjenjivački zloćudni kod (engl. Ransomware)
    - Kod koji na nekakav način pokušava ucijeniti vlasnika kompromitiranog računala

Forma zloćudnog koda

- Zloćudni kod može biti gdje god se nalazi programski kod
- Zloćudni kod može biti i u podacima
- Izvršne datoteke operacijskog sustava (npr. EXE)
- Skripte operacijskog sustava (Powershell)
- MS Office dokumenti (Visual Basic kod)
- PDF dokumenti (Javascript kod)
- Mobilne aplikacije
- Web

C&C poslužitelj

- Jako često se zloćudni kod po uspješnoj instalaciji javlja nekom računalu na Internetu (engl. phone home)
- Korištenjem C&C poslužitelja napadač se štiti od otkrivanja

Zaštita od zloćudnog koda

- Antivirus
- Dinamička analiza elektroničke pošte i Web prometa
- Blokiranje C&C poslužitelja

Indikatori kompromitacije (IOC) su podaci koji omogućavaju detekciju zloćudnog koda. Vrlo su efikasan način za utvrditi je li nešto zaraženo.

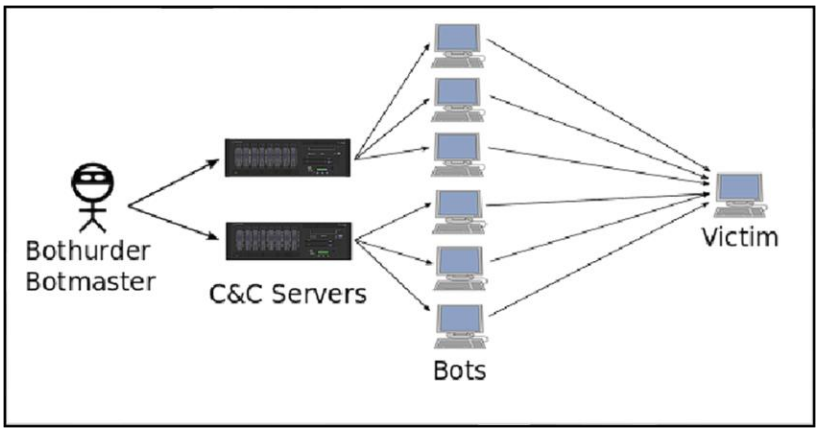
Napadačima je jednostavno promijeniti IOC-e.

Analiza zloćudnog koda

- **Reverzno inženjerstvo** - metode, tehnike i procesi uz pomoć kojih se pokušava saznati što i kako nešto radi
    - **Metode:**
        - Pokretanje koda u zaštićenoj okolini (engl. sandbox) i praćenje rezultata
        - Analiza koda u debugeru
        - Statička analiza
    - Zloćudni kod pisan je u raznim jezicima i alatima i ne postoji univerzalni skup alata za sve potrebe reverzanja

Botnet

- Skup zaraženih računala kojima upravlja botmaster
- Svako računalo izvršava maliciozni kod koji čini računalo dijelom botneta
- Sastoji se od napadača, izvora napada, C&C poslužitelja, botova, žrtve i komunikacijskih puteva



Slika 4. Botnet

Izvori prijetnji i prijetnje

Primjeri prijetnji i pridruže ranjivosti

- Pogađanje kriptografskog ključa
- Preplavljivanje poslužitelja zahtjevima
- Lažiranje poruka elektroničke pošte
- DDoS napada na tvrtku
- Poplava u sistemskoj sali
- Napajanje poslužitelja prestane raditi
- Ransomware

Ne postoji formalni popis prijetnji.

Ne postoji formalni popis ranjivosti koje svaka prijetnja iskorištava.

Prijetnje mogu biti tehničke, taktičke, operativne i strateške.

Dodatni pojmovi vezani uz prijetnje

- **Agent prijetnje** (engl. threat agent) je subjekt koji provodi prijetnju
- **Izvor prijetnje** (engl. threat source) je onaj koji je prijetnju potaknuo
- Ova dva termina se često poklapaju
- **Napad** je kombinacija izvora prijetnje, namjere, prijetnje i posljedice

Podjela izvora prijetnji

- Prirodni izvori (npr. prirodne nepogode)
- Ljudski izvori
    - Postoje dvije podjele:
        - **Namjerni** (napadači) ili slučajni (posljedice grešaka)
        - Vanjski ili unutarnji (u odnosu na sustav koji se štiti)

- Postoji jako puno izvora prijetnji na Internetu i grupiramo ih da se lakše nosimo s njima
- Podjelu je moguće napraviti na temelju sljedećih karakteristika
    - Raspoloživi materijalni resursi
    - Motiv i ciljevi – Što žele postići, odnosno, do čega žele doći te koga napadaju
    - Ustrajnost – Koliko je bitno da dođu do zadanog cilja
    - Količina ljudskih resursa i njihove kompetencije

Izvori prijetni

- Napredne ustrajne prijetnje (APT)
    - Napredne ustrajne prijetnje su dijelovi obavještajnih službi ili dio vojne organizacije
    - Skoro neograničene količine resursa
    - Raspoloživi ljudi imaju vrlo visoke kompetencije
    - Vrlo su ustrajni i ciljani u svom djelovanju
    - Prikrivaju svoju prisutnost
- Kibernetički kriminalci
    - Motiv je zarada, a ustrajnost srednja
    - Dvije vrste kriminalnih aktivnosti:
        - Tradicionalne kriminalne aktivnosti koje se obavljaju putem kibernetičkog prostora
        - Kriminalne aktivnosti koje je omogućio kibernetički prostor
    - Načini zarade:
        - Krađa podataka
        - Prodaja ilegalne robe i usluga
        - Ucjenjivanje i iznuđivanje

        - Prijevare
- Haktivisti
    - Slabo povezana grupa anonimaca
    - Ne pretjerano velikih kompetencija i resursa
    - Promoviraju nekakve političke, svjetonazorske i slične stavove
    - Trude se biti što vidljiviji
    - Uporni su prema grupi ciljeva, a ne specifičnim ciljevima
- Pojedinačni napadači
    - Napadači potencijalno vrlo velikih vještina i kompetencija
    - Razlikuju se po tome djeluju li etično (white hats) ili ne (black hats)
    - Vrsta Gray Hats su na granici
    - Script Kiddies - napadači najmanje razine vještine
- Automatizirane probe
    - Dvije vrste automatiziranih proba:
        - Skeneri koji stalno pretražuju Internet za ranjivim servisima
        - Crvi koji se pokušavaju zaraziti druga računala na mreži
    - Više spadaju pod smetnju/dosadu nego nešto ozbiljno, lako obranjivi

Kako izgleda napad?

- Napad (ponašanje napadača) pokušavamo opisati modelom
- Najpoznatiji model napada je Cyber Kill Chain
- Sastoji se od sljedećeg niza koraka:
    1. Istraživanje
    2. Naoružavanje
    3. Isporuka
    4. Iskorištavanje
    5. Instalacija
    6. Uspostava upravljačkog kanala
    7. Djelovanje

- Organizacija MITRE slaže bazu taktika, tehnika i procedura uočenih u napadima

Česte tehnike: Društveni inženjering

- Čovjek je najslabija karika
- Napadači često iskorištavaju ljudske slabosti jer im je to najjednostavnije
- Phishing, Spear Phishing

Problem atribucije

- Atribucija: Odgovor na pitanje tko stoji iza napada?
- Zbog načina kako Internet radi teško je dati odgovor na to pitanje
- Sposobniji napadači prikrivaju svoj pravi identitet (npr. pomoću VPN-a)

Kontrola pristupa

Kontrola pristupa sastoji se od dva koraka:

- **Autentifikacija** - proces provjere identiteta SUBJEKTA (korisnika, procesa ili uređaja)
- **Autorizacija** - proces odlučivanja može li SUBJEKT obaviti točno određenu OPERACIJU nad OBJETKOM

Metode autentifikacije

- Temelje se na jednom ili kombinaciji više faktora
- **Jedno-faktorska autentifikacija (1FA)**
  - Najčešće korištena, u dosta slučajeva nedovoljna
- **Dvo-faktorska autentifikacija (2FA)**
  - Sve više korištena i dovoljno visoke razine zaštite
- **Višefaktorska autentifikacija (MFA)**
  - Rijetko korištena

Primjeri metoda autentifikacije

1. Lozinke
2. Dijeljene tajne
3. Fraze
4. Jednokratne lozinke
5. Pametne kartice
6. Biometrijske metode

Lozinke

- Niz znakova, temeljene na onome što znamo
- Jednostavne za implementaciju i korištenje
- **Ranjivosti i prijetnje na sigurnost lozinki**:
  - Niska razina slučajnosti i kompleksnosti (ranjivost)
  - Upotreba iste lozinke na više raznih mjesta (ranjivost)
  - Krađa lozinki (prijetnja)
  - Presretanje/krađa lozinki tijekom prijenosa (prijetnja)
  - Sustav obnavljanja lozinki
- **Smanjenje ranjivosti lozinka**
  - Ispravno ih pohranjivati
  - Trebaju biti odgovarajuće kompleksnosti i često se mijenjati
  - Spriječiti pogađanje
  - Korisnik ih ne smije dijeliti s nikim više te koristiti jedinstvene lozinke za svaku uslugu
  - Tijekom unosa paziti da ih netko ne otkrije

    - Zaštititi tijekom prijenosa
- **Sigurna pohrana lozinki**
    - Svakoj lozinki se dodaje slučajna vrijednost (salt)
    - Slučajna vrijednost i lozinka se propuštaju kroz kriptografsku funkciju sažetka
    - Na disk se pohranjuju slučajna vrijednost i rezultat kriptografske funkcije sažetka
    - Salt onemogućava dvije stvari
        - Ista lozinka ima isti zapis u bazi
        - Napad korištenjem tzv. Rainbow tables
- **Sprečavanje pogađanja lozinki**
    - Pogađanje može biti on-line ili off-line
    - Obje vrste napada otežavamo ako su lozinke minimalne određene kompleksnosti
    - Kako bi spriječili pogađanja lozinke često se uvode dva dodatna ograničenja:
        - Nakon svakog neuspjelog pokušaja upisivanja lozinke povećava se vrijeme čekanja
        - Nakon određenog broja neuspjelih pokušaja korisnik se blokira, korisnički račun se zaključava, generira se upozorenje vlasniku sustava
- Periodički se traži promjena lozinke
- Za očekivati je da će ljudi zaboravljati lozinke i često se u raznim aplikacijama na Internetu ugrađuje mogućnost resetiranja lozinke

Prijenos lozinke preko mreže

- Problem ako mjesto gdje korisnik upisuje lozinku i mjesto gdje se provjerava lozinka nisu na istom računalu
- Rješenje tog problema je izazov-odgovor način provjere
- **Izazov odgovor način provjere (challenge-response)**
    1. Poslužitelj šalje slučajan broj na klijent
    2. Korisnik upisuje korisničko ime i lozinku
    3. Klijent slučajan broj i lozinku propušta kroz funkciju sažetka
    4. Korisničko ime i rezultat funkcije sažetka se šalju na poslužitelj
    5. Poslužitelj na temelju korisničkog imena dohvaća pohranjenu lozinku te izračunava funkciju sažetka na temelju lozinke i slučajnog broja kojeg je poslao klijentu
    6. Ako je izračunata vrijednost ista kao i primljena verzija autentifikacija je uspješna

Dijeljene tajne

- Služe za međusobnu autentifikaciju – i jedna i druga strana dokazuju poznavanje dijeljene tajne

Fraze

- Dvije razlike u odnosu na lozinke:
    - Značajno su dulje (rečenice i slično)
    - Uz njih nije vezano korisničko ime

Jednokratne lozinke

- Najčešće broj od 4 i više znamenki koji se generira po nekom poznatom algoritmu
- Algoritmi: TOTP – RFC6238, HOTP – RFC4226

Pametne kartice

- Kategorija ono što imamo (kartica) i ono što znamo (PIN)
- Dvofaktorska autentifikacija
- Temelj za autentifikaciju su privatni i javni ključevi
- Privatni ključ se nalazi na kartici i nikada ne izlazi van, javni ključ je svima poznat

Biometrijske metode

- Koristi se jedinstvenim biološkim karakteristikama
- Otisak prsta, slika lica, slika rožnice, stil tipkanja, otisak dlana, ...
- Problematični u slučaju krađe autentifikacijskih podataka

Autorizacija

- Nakon što je korisnik identificiran utvrđuje se pravo korisnika da provede nekakvu operaciju nad nekim resursom
- **Subjekti** su korisnici, odnosno, procesi koji djeluju u ime korisnika
- Koji **objekti** i dozvole postoje ovise o aplikaciji i onima koji su je razvijali
- **Autorizacija bazirana na dozvolama**
    - Objekt za svaki subjekt ima definirano može li obaviti pojedinu operaciju
    - Dodavanje/uklanjanje dozvola svodi se na modifikaciju odgovarajućih struktura podataka
- **Autorizacija bazirana na ulogama**
    - Prava se grupiraju u uloge, a uloge se dodjeljuju subjektima
    - Puno veća fleksibilnost i upravljivost

Sigurnost programske podrške – ranjivosti i napadi

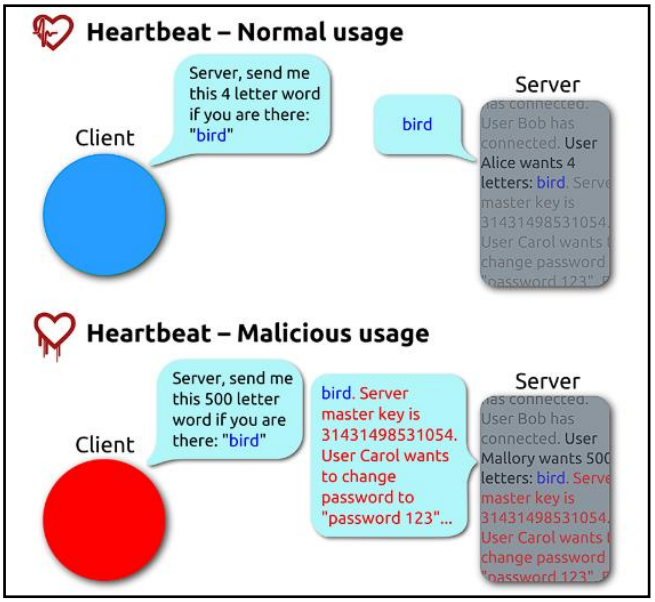
Defenzivno programiranje

- „Obično” programiranje - program u razumnim okolnostima, za razumne ulaze treba davati razumne izlaze
- Defenzivno programiranje - čak i u potpuno nerazumnim okolnostima i za potpuno nerazumne ulaze, program se ne smije ponašati nerazumno
- Potrebno je razumjeti napade kako bi shvatili rizike i oblikovali obrane
- Primjeri čestih i opasnih ranjivosti programske podrške:

- Provjera ispravnosti ulaza (input validation)
- Napadi prelijevanjem međuspremnika (buffer overflow attacks)
- Interakcija s okolinom i operacijskim sustavom

Provjera ispravnosti ulaza

- Provjera sintakse ulaza - je li ulaz ispravne veličine i ispravno formatiran
- Provjera semantike (značenja) ulaza - ima li ulaz smisla
- Primjer: Heartbleed ranjivost
  - Ranjivost u OpenSSL kriptografskoj biblioteci



Slika 5. Heartbleed ranjivost

- Primjer: Command injection (općenita verzija SQL injectiona)

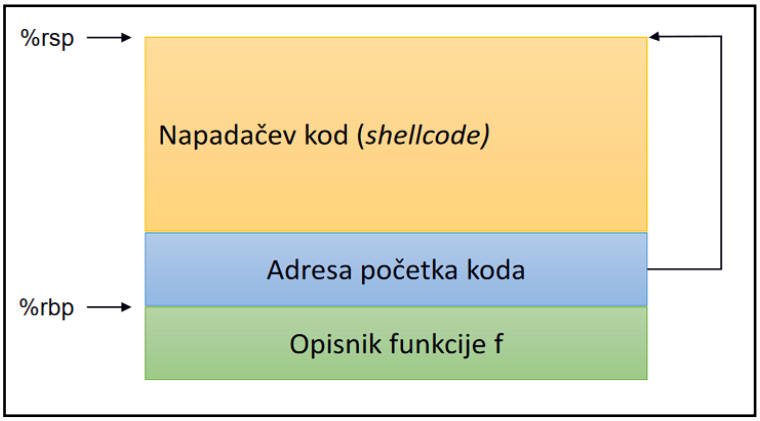
```
$userName = $_POST["user"];
$command = 'ls -l /home/' . $userName;
system($command);
```

Slika 6. Command injection

- Treba voditi računa o različitim jezicima, kodiranjima znakova.
- Blacklist vs whitelist pristup
- Preporuka: sve pretvarati u kanonski oblik pa onda provjeravati ispravnost

Preljev međuspremnika (buffer overflow)

- Primjeri iz povijesti: Morris crv, Code Red crv, Iphone ranjivosti
- Napadač kao ulaz daje strojni kod koji želi da se izvrši te njegovu očekivanu adresu
- Ulaz je pažljivo namješten tako da adresa točno prepiše adresu za povratak na spremljenu na stogu
- Nakon instrukcije ret izvršava se kod napadača
- Shellcode - najčešće kratak strojni kod koji omogućuje ili olakšava napadaču postizanje cilja



Slika 7. Prelijevanje međuspremnika

- Napadač ne može skroz pouzdano predvidjeti točnu adresu vrha stoga
  - Rješenje: prije samog koda staviti dugačak niz nop instrukcija
- Obrana: write xor execute
  - Razlikujemo podatke i programe: procesoru kažemo da neke memorijske stranice sadrže podatke te da se ne smiju izvršavati
  - Većina modernih sustava ima uključenu ovu obranu.
- **Zaobiliženje NX bita – return-to-libc napad**
  - Napadač prepisuje stog tako da se poziva postojeća funkcija
  - Dodatno, napadač na prepisani stog stavlja i argumente za tu funkciju po njegovoj želji
  - Obrana: randomizacija memorijskog prostora
    - Adrese memorijskih stranica su (djelomično) slučajno odabrane
  - Obrana: kanarinci na stogu
    - Prilikom pozivanja funkcije stavlja na stog određenu vrijednost nepoznatu napadaču
    - Prije završetka funkcije provjerava je li točno ta vrijednost još uvijek na stogu

Sigurnost programske podrške 2

Taksonomija pogrešaka

- Namjerne
  - Zlonamjerne
  - Nezlonamjerne
- Nenamjerne
  - Pogreške pri provjeri valjanosti
  - Logičke pogreške
  - Neadekvatna identifikacija ili autentifikacija

Nezlonamjerne pogreške

- Preljevi spremnika
- Nepotpuna provjera valjanosti ulaznih parametara
- Sinkronizacija provjere i pristupa

Statička i dinamička analiza koda

- Statička analiza = analiza izvornog koda
- Dinamička analiza = analiza binarnog (izvršnog) koda

Problemi

- Koliko programski jezik pazi na način pisanja koda u smislu rukovanja tipovima podataka – možemo li napisati kod koji će biti besmislen/pogrešan?
- Možemo li biti sigurni da će se program izvoditi kako smo mi zamislili?

Strogo tipiziran jezik (strongly typed) - rješenje

- Strog u definiranju i rukovanju tipovima podataka – „pazi umjesto nas”
- Provjerava kod u nekom trenutku – najčešće prilikom prevođenja ili tijekom izvođenja (baca grešku)
- Java, C#, Rust

Sigurni razvoj programske podrške

Cilj nam je uključivanje sigurnosti u dizajn sustava od faze skupljanja zahtjeva.

Koraci prije implementacije sustava

- Što moramo osigurati?
- Profili napadača
- Identifikacija i klasifikacija entiteta u sustavu
- Postavljanje arhitekture

Smjernice za siguran dizajn

- **Minimizacija prostora za napad**
  - Princip minimizacije jest da ne dodajemo funkcionalnosti koje nisu doista potrebne čime ćemo efektivno smanjiti sigurnosne rizike
- **Definiranje sigurnih početnih postavki**
  - Iako se korisnicima želi olakšati, početne postavke trebale bi biti maksimalno sigurne
  - Omogućiti korisnicima da na vlastitu odgovornost smanje razinu sigurnosti i olakšaju korištenje
- **Princip najmanjih prava**
  - Korisnici / dijelovi sustava / servisi bi trebali imati samo ona prava na resurse koja im doista trebaju
- **Princip obrane u dubinu**
  - Princip obrane u dubinu zapravo govori da uvijek treba biti više mehanizama obrane
  - Dobra praksa - više neovisnih mehanizama koji štite isti dio sustava
- **Sigurno ispadanje**
  - Svaki sustav će najvjerojatnije nekada ispasti, važno je što će sustav izvesti u tom trenutku
- **Ne vjerujte vanjskim uslugama**
- **Razdvajanje zaduženja**
  - Nema preklapanja između funkcionalnosti
  - Npr. administratori nikada ne bi smjeli imati ulogu običnih korisnika
- **Izbjegavajte sigurnost prikrivanjem**
  - Kod bi trebao biti siguran i onda kada je javno dostupan
- **Jednostavna sigurnost**
  - Sve riješiti na najjednostavniji mogući način
  - Ipak, razmisliti o obrani u dubinu i ne zanemariti više razina obrane
- **Ispravne sigurnosne zakrpe**
  - Ne žuriti s popravkom nego pokušati doći do stvarnog uzroka

Sigurnost operacijskih sustava

Razdvajanje

- Osnova zaštite – objekti jednog korisnika nisu vidljivi drugom korisniku
- Sandboxing
- Osim razdvajanja, želimo i dijeljenje resursa (sharing)

Dijeljenje

- Različite razine
  - Dijeljenje uz ograničenje pristupa
  - Dijeljenje prema mogućnosti
  - Ograničenje korištenja objekta
- Pristup podacima na više razina: bit, byte, word, polje, zapis, datoteka, disk – granularnost

Ovlasti

- Podjela na tri skupine ovlasti
  - Vlasnik (owner) - u
  - Grupa korisnika (group, staff) - g
  - Svi ostali (other, all, everyone) – o, a
- Podjela na tri skupine akcija
  - Read - r
  - Write - w
  - Execute - x
  - Bez ovlasti –
- r=4, w=2, x=1
- rwx = 4+2+1 = 7 -> rwxrwxrwx = 777
- Primjer promjene ovlasti
  - „chmod 644 file.txt“

Sandboxing kao koncept

- Koncept kod kojeg se svaki proces/aplikacija izvodi u svojem „pješčaniku”
- Za „izlazak iz pješčanika” treba posebne dozvole

Šifriranje datoteka/diska

- Još jedna mogućnost zaštite podataka
- Različita rješenja na sustavima