Projektiranje sigurnosti

Modeliranje prijetnji
- sigurnosna analiza koja pomaže u otkrivanju najvećih sigurnosnih opasnosti

- cilj je odrediti koje prijetnje i na koji način treba ukloniti
- pretpostavka - proizvod nije siguran ako se ne procijene prijetnje i smanji rizik

Što omogućuje modeliranje prijetnji?

- bolje shvaćanje aplikacije
- pronalaženje pogrešaka
    - procjena da MP pronađe 50% pogrešaka, a ostatak testiranjem i analizom koda
    - pogreške složenih aplikacija, koje se rijetko pronađu drukčije

Načela i proces modeliranja prijetnji

- dugotrajan posao
- bitno je da se obavi kvalitetno
- najbolje iterativno

Proces modeliranja prijetnji

1. Određivanje ciljeva zaštite
2. Arhitektura aplikacije
3. Dekompozicija aplikacije
4. Određivanje prijetnji
5. Dokumentiranje prijetnji
6. Rangiranje prijetnji

Rezultat modeliranja prijetnji =>

- **Dokument s modelima,** definicijom arhitekture i popisom prijetnji

Korak 1 – identifikacija resursa koje treba zaštititi

Korak 2 – Pregled arhitekture

- Dokumentiranje funkcija aplikacije, arhitekture i tehnologija implementacije
- Modeliranje funkcionalnosti (use cases)
- Provjera (kršenja) poslovnih pravila
- Izrađuje se dijagram visoke razine - opisuje strukturu (komponente) sustava

Korak 3 – Dekompozicija aplikacije

- izrada sigurnosnog profila
- Određivanje:
    - granica povjerenja (trust boundaries)
    - toka podataka
    - mjesta unosa
    - privilegiranog koda

Određivanje granica povjerenja

- analiza okruženja resursa određenog dizajnom aplikacije
- za svaki podsustav, procjena je li ulazni tok ili korisnički unos povjerljiv
    - ako nije – razmotriti kako ih autentificirati i autorizirati
- procjena je li pozivajući programski kod povjerljiv
- provjera povjerenja poslužitelja

Određivanje toka podataka

- iterativna dekompozicija
- analizom tokova između podsustava, pa u dubinu

Dijagram toka podatak – notacija

Proces, višestruki proces

- obrada podataka, ili akcija temeljem podataka
- kolekcija potprocesa, može se dekomponirati

Spremište podataka

- Bilo koji oblik pohrane (datoteka, BP, …)

Granica povjerenja

- oznaka promjene privilegije (razine prava nad podacima)

Vanjski entitet, sudionik

- sve što je izvan aplikacije, a u interakciji putem točke unosa

Tok podataka

- usmjereno kretanje podatka unutar aplikacije

Ostale aktivnosti dekompozicije

- Određivanje točki unosa
- Određivanje privilegiranog koda
- Dokumentiranje profila sigurnosti

Korak 4 - Određivanje prijetnji

- Odrađuju razvojni tim i tim za testiranje

Osnovni pristupi:

1. **STRIDE****

- **S**poofing – zavaravanje, lažiranje
- **T**ampering [with Data] – zlonamjerna izmjena podataka
- **R**epudiation – nepriznavanje, poricanje
- **I**nformation disclosure - otkrivanje informacija
- **D**enial of service - uskraćivanje usluge
- **E**levation of privilege - povišenje ovlasti

postupak:

Provodi se tako da se sustav raščlanjuje u relevantne komponente pa se onda:

- procjenjuje osjetljivost na prijetnje svake komponente (analiza dijagramom toka podataka)
- prijetnje se smanjuju (mitigation) prikladnim svojstvima sigurnosti
- ponavlja se (rekurzivno) do zadovoljavajućeg rezultata

2. **Kategorizirane liste prijetnji** - popis uobičajeno "sumnjivih" prijetnji
3. **Stabla prijetnji**
   Za svaku komponentu dobivenu dekompozicijom
   - određuju se moguće prijetnje
   - utvrđuje se način na koji se prijetnje odražavaju na sustav
   - Primjer
     - korijen predstavlja prijetnju
     - djeca predstavljaju korake koje napadač mora poduzeti da bi ostvario prijetnju
4. **Obrasci napada**
   Općenita reprezentacija uobičajenih napada
   - definira cilj, uvjete, tehniku i rezultat napada
   - Naglasak je na **tehnici napada** (kod **STRIDE** na **ciljevima napadača**)

Korak 5 - Dokumentiranje prijetnji
Predložak za evidenciju prijetnji

- svakako se popunjavaju opis i cilj
- rizik se ostavlja za naredni korak
- ostali atributi mogu biti opcionalni (tehnike napada, protumjere)

Korak 6 - Rangiranje prijetnji
često se koriste tehnike za određivanje rizika

- **rizik** = **vjerojatnost događaja * potencijalna šteta**
- **vjerojatnost** npr. u rasponu 1-10
- **šteta** npr. u rasponu 1-10
- **rizik** u rasponu 1-100
- raspodjela u tri grupe (**visok**, **srednji**, **nizak**) koje predstavljaju prioritete

DREAD (model procjene rizika)**

DREAD – klasifikacija računalnih prijetnji

- **Damage potential** – moguća šteta, veličina štete bude li napad uspješan
- **Reproducibility** – reproduktivnost, koliko je jednostavno ponoviti napad
- **Exploitability** – iskoristivost, trud i znanje potrebnih za uspješan napad
- **Affected users** – zahvaćeni korisnici, moguće uspjelim napadom, postotno
- **Discoverability** – mogućnost otkrivanja, teško mjerljivo

Procjena svake prijetnje po navedenim parametrima

- pojedinačno vrijednost od 1 do 10 (najmanje loše – najgore)
- **ukupan rizik** - prosjek 5 pojedinačnih **DREAD** vrijednosti

Bolje – (jednostavna) shema ocjenjivanja

- Nisko, srednje, visoko – preslikano u interval 1 do 3
- Zbrajaju se vrijednosti (1-3) za zadanu prijetnju
    - rezultat je u rasponu 5-15
- pridjeljuje se rizik, npr. 5-7 nizak, 8-11 srednji, 12-15 visok

Razrješenje prijetnji (nakon modeliranja)

- Popraviti (smanjenje, redukcija rizika)
- Ne učiniti ništa (prihvatiti rizik)
- Obavijestiti korisnika te mu prepustiti odluku o korištenju (prijenos)
- Uklanjanje rizičnog svojstva (izbjegavanje)

Smanjenje površine napada

Površina napada - kolekcija ulaznih točaka programskog proizvoda

- mjera "napadljivosti"
- Veća površina napada = više posla zaštite = veća potencijalna šteta
- Površina određuje rizik napada – mjera potencijalnog pristupa i udara

Združeni model površine napada

- kontrole pristupa smanjuju
    - mogućnost da se dosegne sustav
    - broj elemenata koji su vidljivi ili se mogu koristiti

Smanjenje površine napada

Glavni ciljevi

- Smanjenje količine koda koji se izvodi „po viđenju” (by default)
- Smanjenje količine koda kojem mogu pristupiti nepouzdani (untrusted) korisnici, „po viđenju”
- Zatvaranje pristupnih točaka (access points, entry points) – vrata koja se lako otvaraju/iskorištavaju
- Ograničavanje štete u slučaju da pristupna točka bude iskorištena

Krajnji cilj – odbijanje budućih napada

Uobičajena metrika softverske sigurnosti

- Razina programskog koda - brojanje bugova
- Razina proizvoda/sustava
    - Brojanje koliko puta je verzija sustava spomenuta u CERT, MITRE CVE, … biltenima

Mjerenje površine napada

- Mjerenje „avenija” napada
    - Možebitno napadane mogućnosti
- Mjerenje relativne sigurnosti
    - Delta mjerenje – razlike između verzija istog proizvoda
- Postupak
    - Osnovica (baseline) + tjedna mjerenja
    - Određivanje minimalne površine na početku
    - Ako se površina povećava – odrediti kako ju smanjiti

Proces ASR (attack surface reduction)

- Ustanovljavanje pristupnih točki
- Rangiranje točaka - prema korisniku
- Podešavanje

Najbolje prakse

Redukcija koda koji se izvodi *by default*

- Isključiti mogućnost koju ne koristi barem 80% korisnika
- Zaustavljen servis ne može biti napadnut
- Smanjenje pristupa od strane nepouzdanih (untrusted) korisnika
    - Ograničenje pristupa na lokalnu mrežu ili raspon IP adresa
    - Autentifikacija
- Redukcija privilegija radi ograničavanja potencijalne štete
- Definiranje površine napada tijekom dizajna/projektiranja

Provjera sigurnosti

Provjera ispravnosti softvera (općenito)

- Testiranje programa, provjeravanje programa, ispitivanje programa
- Prema svrsi testiranja – verifikacija i validacija
- Prema objektu provjere – strukturalno i funkcionalno
- Prema načinu provjere – statička analiza i dinamička analiza

Ključni pojmovi

- **[„normalan”] Test** - provjerava je li neki aspekt softvera ispravan
- **Test sigurnosti** - nastoji dokazati da neki dio ne radi kako treba
- **Pogreška (error)** - propust programera, npr. radi nerazumijevanja
- **Kvar (fault), defekt (defect), neformalno bug** - neispravan dio koda
- **Zastoj u radu (failure)** - stanje izazvano jednim ili više kvarova
- **Ispravak (Fix)** - stanje popravka

Postupci provjere sigurnosti aplikacija

- Nadzor
- Static Application Security Testing (**SAST**)
- Dynamic Application Security Testing (**DAST**)
- Interactive Application Security Testing (**IAST**)
- Analiza izvornog koda - statička ili dinamička s pristupom čitavom kodu

1. Nadzor

Varijante

- Inspekcija (inspection)
- Timski pregled (team review)
- Prohod (walkthrough)

Nadzor omogućuje:

- nalaženje defekata ranije u životnom ciklusu - do 80% prije testiranja
- nalaženje defekata s manje napora nego testiranjem
- nalaženje drugačijih defekata nego testiranjem - problemi dizajna i zahtjeva

A. Inspekcija**

- Formalni proces
- Temeljita pokrivenost odvojenim ulogama
  - **Moderator** - vodi sastanak, prati probleme
  - **Čitalac** - parafrazira (prepričava) kod, nije autor
  - **Zapisničar** - evidentira defekte
  - **Autor** - osigurava kontekst koda, objašnjava, popravlja nakon pregleda
- **Aktivnosti**:
  - Izrada kontrolnih listi za specifične ciljeve
  - Prikupljanje podataka za praćenje pogrešaka
  - Određivanje potrebe za narednim inspekcijama
- Opsežna dokumentacija učinkovitosti
- ***Proces inspekcije***
- **Planiranje**
  - autor inicira, moderator ekipira, skupa pripreme inspekcijski paket
- **Priprema**
  - recenzenti pregledavaju, koriste kontrolne liste i analitičke alate, označavaju defekte
- **Sastanak**
  - čitalac prepričava, recenzenti komentiraju i zapitkuju, zapisničar evidentira
  - tim zaključuje procjenu koda
- **Prerada** - autor popravlja
- **Kontrola** (follow-up)
  - moderator verificira korektnost promjena, autor prijavljuje kod (check-in)

B. Timski pregled

- Timski pregled ("lagana" inspekcija)
- Osobe: moderator, recenzenti (koji nisu autori koda)
- Moduli ili manji skupovi klasa
- 1-2 sata, < 1 kLOC

C. Prohod (walkthrough)

- Autor vodi sastanak i objašnjava kod
- Manje formalan proces
- Nedefiniran proces
- Nema kontrolnih lista ili metrike

Ostali postupci: *programiranje u paru, peer deskcheck, pass around*

Statička provjera**

- **SAST (Quick and Dirty)**
  - Analiza koda bez izvršavanja
  - Obuhvaća sve osim testiranja
  - Korištenje analizatora koda
  - Može biti dio revizije koda
- Ograničenja: pogrešno otkrivanje (**false postive**) i pogrešno neprepoznavanje (**false negative**)
- **Vrste statičke analize**
  - Provjera tipova
  - Provjera stila
  - Razumijevanje programa – zaključivanje značenja
  - Provjera svojstava - osiguranje da nema lošeg ponašanja
  - Verifikacija programa - osiguranje ispravnog ponašanja
  - Traženje pogrešaka
- **Mehanizmi statičke analize**
  - Parser
  - Model Builder
  - Analysis Engine
- **Tehnike analize**
  - Leksička analiza i parsiranje
  - Analiza toka podataka
  - Analiza „mrlja” - identifikacija varijabli uprljanih korisničkim unosom
  - Pravila propagacije mrlja : pravila izvora, pravila slivnika, pravila propuštanja, pravila čišćenja, pravila početka
- **Prednosti statičke analize**
  - Potpuna pokrivenost koda (code coverage) - u teoriji
  - Potencijal potvrde izostanka čitavih klasa bugova
  - Hvata bugove različite u odnosu na dinamičku analizu

- **Slabosti statičke analize**
    - Visok postotak pogrešnog otkrivanja
    - Teško oblikovanje testa
    - Složenost izgradnje (alata) - „parser za svaki jezik"
    - Neimanje cjelokupnog izvornog koda u praksi
- **Alati za statičku analizu – StyleCop, CodeSmart**

Dinamička provjera**

Fuzzing - "pročešljavanje"

- ubrizgavanje kvara u aplikaciju (fuzzing, fuzz testing)
- slanje neispravnih, neočekivanih ili nasumičnih podataka ulazu programa
- slično regresiji, samo s lošim podacima
- „češljanje" aplikacija, protokola, datoteka
- **PREDNOSTI**: jednostavnost, nezavisnost o platformi, jeziku
- **NEDOSTACI**:
    - primjena na uzak skup povredivosti
    - složena primjena na tehnologije
    - relativno dugo trajanje
- **Postupci:**
    - Glupo = Dumb (mutational) fuzzing
        - dovoljno manje znanja o cilju i alatima
        - pseudoslučajne anomalije ispravnih podataka
    - Pametno = Smart (generational) fuzzing
        - podaci generirani na temelju modela
        - zahtijeva dubinsko poznavanje cilja i specijaliziranih alata
        - Smišljene anomalije poznavanjem formata, standarda
- Alati: **CERT BFF** i **FOE**

Penetracijsko testiranje (Pen Test), etičko hakiranje

- procjena sigurnosti sustava ili mreže simuliranjem zlonamjernog napada
- osoba, ekipa, poželjno vanjski konzultanti (?)
- pismena dozvola vlasnika (provedbe nezakonitih aktivnosti)

Svrha

- Potvrda funkcionalnosti sigurnosnih kontrola
- Pravovremeno uočavanje sigurnosnih propusta
- Prevencija sigurnosnih incidenata
- Opravdavanje investicije
- Ispunjavanje regulatornih zahtjeva

Pristup penetracijskom testiranju

- bez dostupnih informacija
- sa svim informacijama
- s djelomično dostupnim informacijama

Kriterij početne točke testa

- Vanjski - s udaljene lokacije
- Unutrašnji - s intraneta

Ostali kriteriji - opseg, prikrivenost, tehnike, agresivnost
Izvođenje penetracijskog testa

Istraživanje (eng. reconnaissance), izviđanje

- ispitivač pokušava prikupiti što više informacija.
- **pasivno** - javno dostupne informacije (npr. podaci s društvenih mreža, Google)
- **aktivno** - istraživački alati (npr. nslookup), da bi se odredili određeni parametri

Skeniranje (eng. scanning)

- ispitivač skenira otvorene portove (port scanning) korištenjem alata (npr. Nmap)
- cilj - enumeracija servisa, verzije enumeriranih servisa i OS (OS and service fingerprinting).
- skeniranje ranjivosti (vulnerability scanning), automatiziranim alatima (npr. OpenVAS)

Dobivanje pristupa (eng. obtaining access)

- iskorištavanje ranjivosti, ručno ili alatom (npr. Metasploit),
- ovisno o dogovoru s vlasnikom, neke ranjivosti se neće iskorištavati (npr. rušenje poslužitelja)

Zadržavanje pristupa (eng. maintaining access)

- ispitivač instalira zloćudne backdoor i rootkit programe za daljnji pristup sustavu
- ova i naredna faza se u praksi najčešće ne provode ali predstavljaju scenarij realnog napada

Brisanje tragova (eng. erasing evidence)

- ispitivač pokušava izbrisati dnevničke zapise koji bi ukazivali na njihov neovlašteni pristup

Alati za penetracijsko testiranje i detekciju upada

- Brutus (lozinke), Snort