
# Zaštita i sigurnost informacijskih sustava

## Provjera sigurnosti

**prof. dr. sc. Krešimir Fertalj**

**Sveučilište u Zagrebu**
**Fakultet elektrotehnike i računarstva**



# Creative Commons

- **slobodno smijete:**
  - **dijeliti** — umnožavati, distribuirati i javnosti priopćavati djelo
  - **remiksirati** — prerađivati djelo
- **pod sljedećim uvjetima:**
  - **imenovanje**. Morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
  - **nekomercijalno**. Ovo djelo ne smijete koristiti u komercijalne svrhe.
  - **dijeli pod istim uvjetima**. Ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.

U slučaju daljnjeg korištenja ili distribuiranja morate drugima jasno dati do znanja licencne uvjete ovog djela. Najbolji način da to učinite je linkom na ovu internetsku stranicu.

Od svakog od gornjih uvjeta moguće je odstupiti, ako dobijete dopuštenje nositelja autorskog prava.

Ništa u ovoj licenci ne narušava ili ograničava autorova moralna prava.

Tekst licencije preuzet je s http://creativecommons.org/.

# Provjera sigurnosti

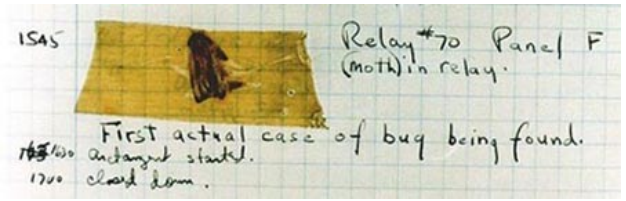
Security Testing

# Provjera ispravnosti softvera (općenito)

- Testiranje programa, provjeravanje programa, ispitivanje programa
  - otkrivanje pogrešaka odnosno nedostataka unutar programa
  - uspješnost testa razmjerna je broju pronađenih pogrešaka
- Prema svrsi testiranja
  - Verifikacija - ovjera ispravnosti (dobra provedba)
  - Validacija - potvrda valjanosti (pravi, prihvatljiv proizvod)
- Prema objektu provjere
  - Strukturalno (white-box testing) - izvorni kod
  - Funkcionalno (black-box) - kompilat
- Prema načinu provjere
  - statička analiza - bez pokretanja, izvorni kod ili .NET MSIL, Java Bytecode
  - dinamička analiza - pokretanjem, što ne isključuje izvorni kod

# Ključni pojmovi

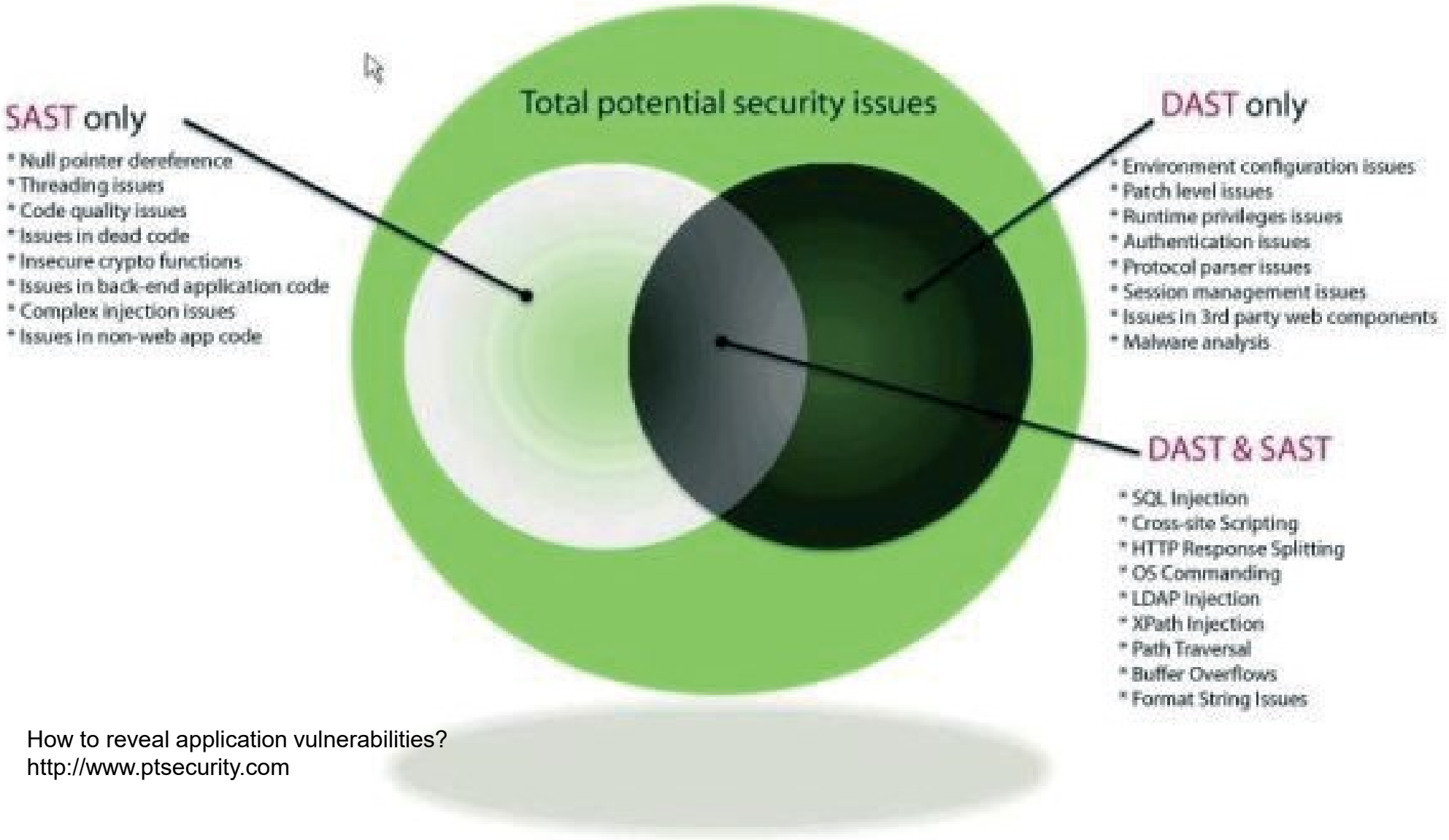
- **[„normalan” ] Test** - provjerava je li neki aspekt softvera ispravan
- **Test sigurnosti** - nastoji dokazati da neki dio ne radi kako treba
- **Pogreška** (error) - propust programera, npr. radi nerazumijevanja
  - dovodi do jednog ili više kvarova
  - razlikujemo u odnosu na "pogrešku" koja znači neželjeno stanje, tj. kvar
- **Kvar** (fault), defekt (defect), neformalno bug - neispravan dio koda
  - npr. pogrešna pretpostavka da se polje indeksira od 1 umjesto 0 izaziva kvar pristupa elementu polja
- **Zastoj** u radu (failure) - stanje izazvano jednim ili više kvarova
  - npr. prestanak rada sustava zbog kvara "buffer overrun"
- **Ispravak** (Fix) - stanje popravka



# Postupci provjere sigurnosti aplikacija

- Nadzor (“Technical Reviews”, “Code Reviews”, "Inspections", ...)
  - testiranje nastoji izazvati zastoj, nadzor traži neispravnost
- Static Application Security Testing (**SAST**) # white-box statička
  - koja ne zahtijeva izvršenje
  - pristup izvornom kodu na klijentu i na serveru
- Dynamic Application Security Testing (**DAST**) # black-box dinamička
  - zahtijeva izvršenje
  - nema pristup izvornom kodu i pogonskoj okolini na serveru
- Interactive Application Security Testing (**IAST**) # white-box dinamička
  - dinamička s pristupom izvornom kodu i pogonskoj okolini na serveru
- Analiza izvornog koda - statička ili dinamička s pristupom čitavom kodu

# How to reveal application vulnerabilities?

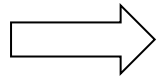



How to reveal application vulnerabilities?
http://www.ptsecurity.com

## Nadzor

Security Review, Code Reviews, Inspection

# Revizija, recenzija (peer review)



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

- Koristi
  - nalaženje defekata ranije u životnom ciklusu - do 80% prije testiranja
  - nalaženje defekata s manje napora nego testiranjem
    - IBM - pregled 3.5 h/defekt, test 15-25 h/defekt
  - nalaženje drugačijih defekata nego testiranjem - problemi dizajna i zahtjeva
  - poduka razvojnika - da ne ponavljaju iste pogreške

# 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
  - Kontrolne liste za specifične ciljeve
  - Prikupljanje podataka za praćenje pogrešaka
  - Određivanje potrebe za narednim inspekcijama

- Opsežna dokumentacija učinkovitosti
  - 16-20 defekt/kLOC inspekcije naspram 3 defekt/kLOC prohoda

# 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 (chek-in)

# Varijante

- Timski pregled
  - Timski pregled ("lagana" inspekcija)
  - Osobe: moderator, recenzenti (koji nisu autori koda)
  - Moduli ili manji skupovi klasa
  - 1-2 sata, < 1 kLOC
- Prohod (walkthrough)
  - Autor vodi sastanak i objašnjava kod
  - Manje formalan proces
  - Nedefiniran proces
  - Nema kontrolnih lista ili metrike

# Drugi postupci

- Programiranje u paru
  - Vodič (driver) i promatrač (observer, navigator)
  - Zamjena uloga ali i partnera
- *Peer deskcheck*
  - Samo jedan recenzent uz autora
  - neformalna recenzija
  - može uključiti kontrolne liste i druge postupke
- *Pass around* (kružno dodavanje?)
  - višestruki, istodobni *Peer deskcheck*
  - više recenzenata (istog koda)

# Statička provjera

Static Analysis

# Statička analiza

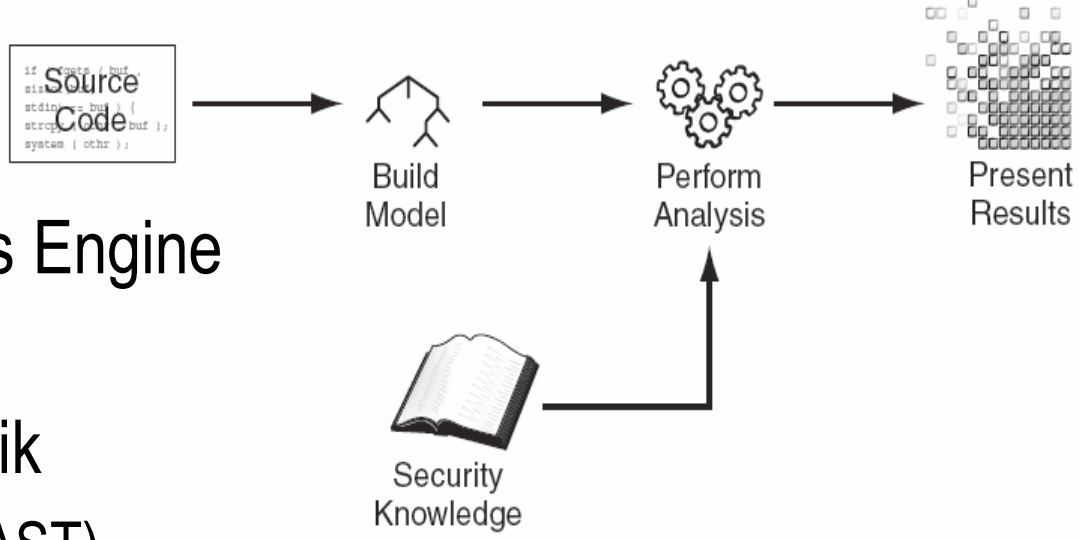
- SAST (Quick and Dirty)
  - Analiza koda bez izvršavanja
  - Obuhvaća sve osim testiranja
  - Korištenje analizatora koda
  - Može biti dio revizije koda

- Ograničenja: pogrešno otkrivanje i pogrešno neprepoznavanje
  - **False Positives** - nepostojeći bugovi, nemoć pri složenom kodu ili vanjskom
  - **False Negatives** - neprepoznavanje bugova, složenost koda, slabost pravila

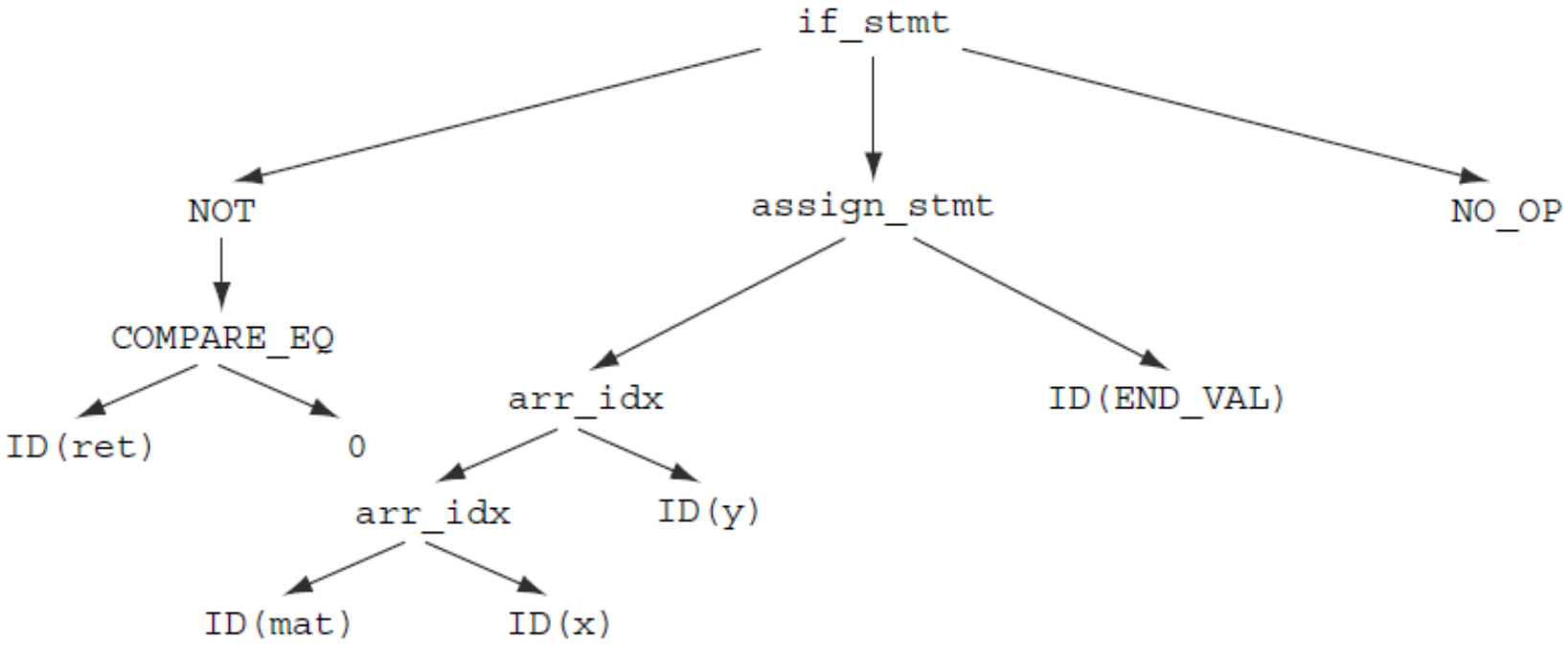
# Vrste statičke analize

- Provjera tipova (Type checking) - dio programskog jezika
  - pr. int i = ”abc”;
- Provjera stila (Style checking) - dobre prakse
  - Pravila - praznine/proredi, nazivlje, komentari, …
- Razumijevanje programa (Program understanding) - zaključivanje značenja
  - Sva korištenja metode, nalaz deklaracije globalnih varijabli, …
- Provjera svojstava (Property checking) - osiguranje da nema lošeg ponašanja
  - Npr. curenje memorije : if (malloc() == NULL)
- Verifikacija programa (Program verification) - osiguranje ispravnog ponašanja
  - Npr. free(mem);
- Traženje pogrešaka (Bug finding) - otkrivanje mogućih pogrešaka
  - Obrasci bugova

# Mehanizmi statičke analize




- Parser, Model Builder, Analysis Engine
- Parser za svaki programski jezik
  - generira Abstract Syntax Tree (AST)



# Tehnike analize

- Leksička analiza (Lexical Analysis) i parsiranje
  - Primjer, programski odsječak
  - Pripadni slijed simbola (tokena)

```
if (ret) // probably true
  mat[x][y] = END_VAL;
```

```
IF LPAREN ID(ret) RPAREN ID(mat) LBRACKET ID(x) RBRACKET
LBRACKET ID(y) RBRACKET EQUAL ID(END_VAL) SEMI
```

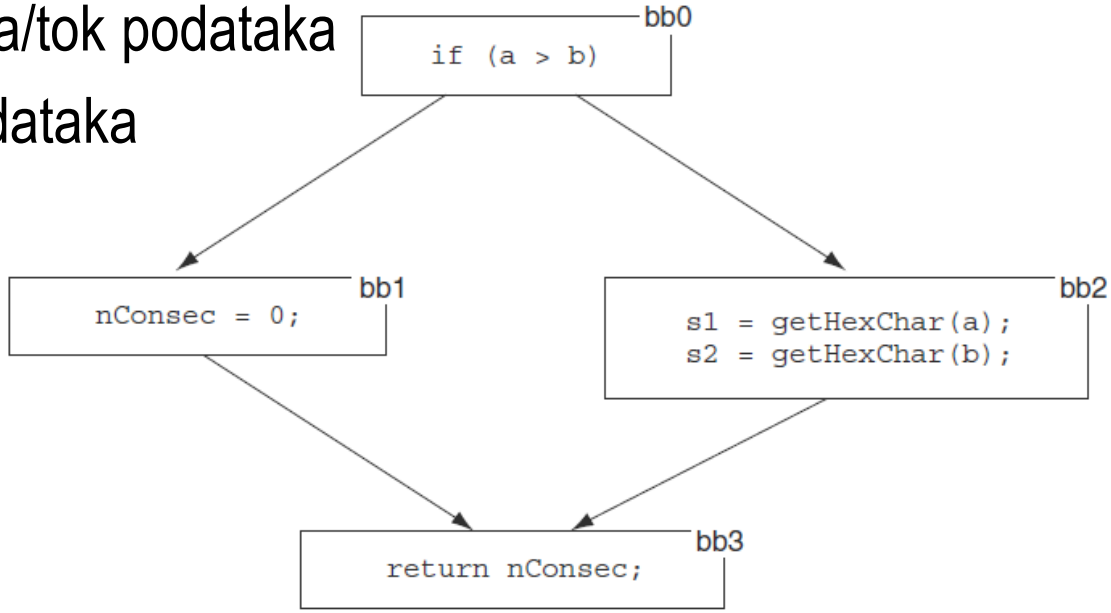
  - Izdvojen sintaksnim pravilima, parsiran prema produkcijama gramatike

```
if                          { return IF; }
(                           { return LPAREN; }
)                           { return RPAREN; }
[                           { return LBRACKET; }
]                           { return LBRACKET; }
=                           { return EQUAL; }
;                           { return SEMI; }
/[ \t\n]+/                  { /* ignore whitespace */ }
/\/\/.*/                    { /* ignore comments */ }
/[a-zA-Z][a-zA-Z0-9]*"/     { return ID; }
```

```
stmt := if_stmt | assign_stmt
if_stmt := IF LPAREN expr RPAREN stmt
expr := lval
assign_stmt := lval EQUAL expr SEMI
lval = ID | arr_access
arr_access := ID arr_index+
arr_idx := LBRACKET expr RBRACKET
```

# Tehnike analize (2)

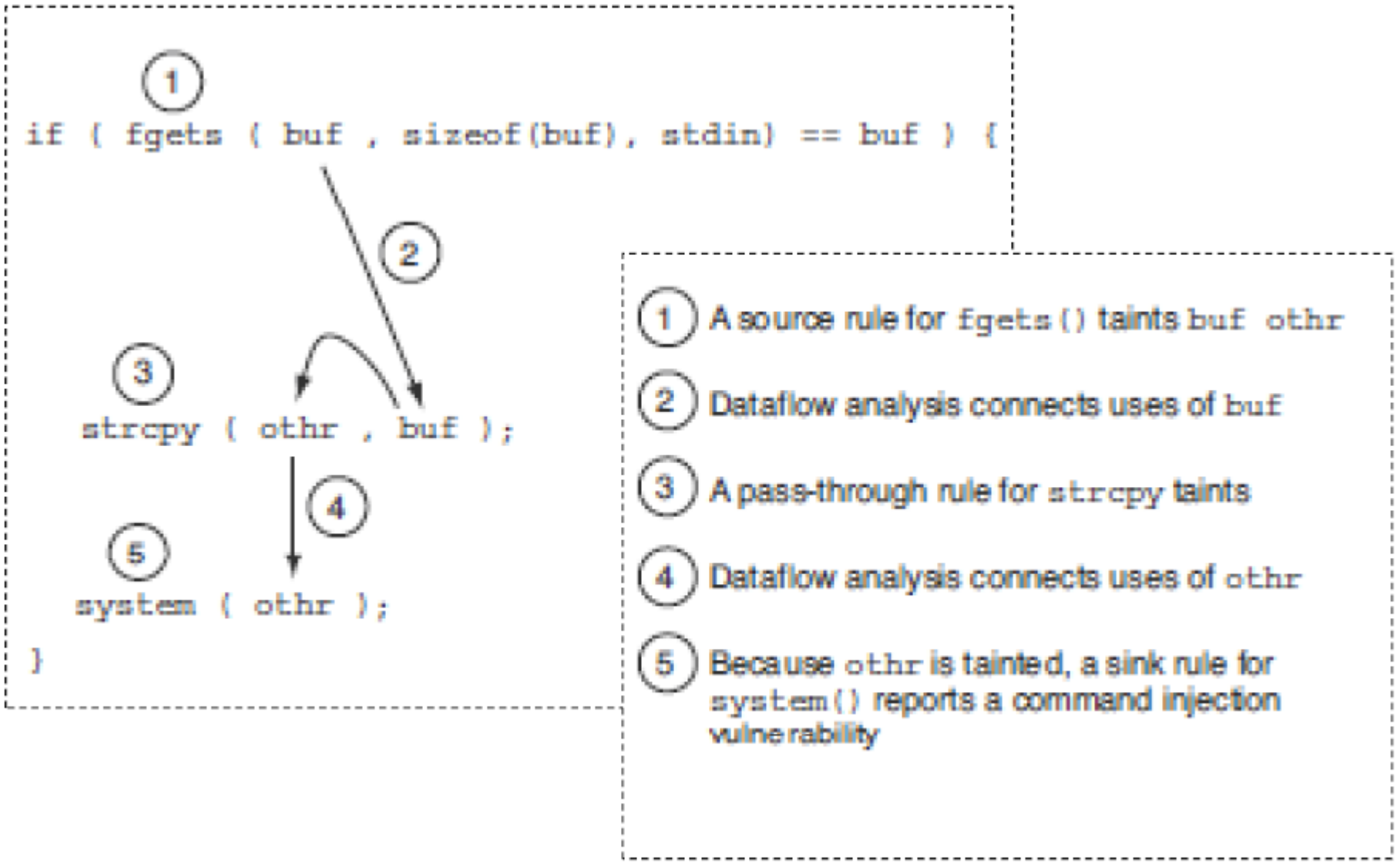
- Analiza toka podataka (Data Flow Analysis)
  - Prikupljanje informacija o kolanju podataka pri izvršenju programa dok je zaustavljen
  - Semantička analiza - na temelju AST i tablice simbola
  - Osnovni blok - slijed naredbi koji se ne zaustavlja ili grana osim na kraju
  - Control Flow Analysis – kontrola/tok podataka
  - Control Flow Path - putanja podataka
- Graf kontrole toka
  - Control Flow Graph (CFG)
  - Primjer: graf s 4 osnovna bloka



# Tehnike analize (3)

- Analiza „mrlja” (Taint Analysis)
  - Identifikacija varijabli *uprljanih* korisničkim unosom
  - Praćenje njihove propagacije prema moguće ranjivim funkcijama (*sink*)
  - Ako nisu dezinficirane prije odvoda - ranjivost
- Pravila propagacije mrlja
  - Pravila izvora (source rules) - unos podataka
    - Naredbe *read(), getenv(), getpass(), gets().*
  - Pravila slivnika (sink rules) - lokacije koje ne bi smjele primiti prljave podatke
    - Primjer, Java *Statement.executeQuery()*, C *strcpy()*
  - Pravila propuštanja (pass-through)
    - Ako je *string* zaprljan i *trim(string)* će biti zaprljan
  - Pravila čišćenja (cleanse rules) - validacija unosa
  - Pravila početka (entry-point rule) - slično izvoru, npr. *main(…)*

# Primjer statičke analize



# Prednosti i nedostaci statičke analize

- Prednosti
  - Potpuna pokrivenost koda (code coverage) - u teoriji
  - Potencijal potvrde izostanka čitavih klasa bugova
  - Hvata *bugove* različite u odnosu na dinamičku analizu

- Slabosti
  - Visok postotak pogrešnog otkrivanja
  - Teško oblikovanje testa
  - Složenost izgradnje (alata) - „parser za svaki jezik"
    - nedovoljno kada se koriste dodatni okviri ili biblioteke
  - Neimanje cjelokupnog izvornog koda u praksi (OS, shared libraries, DLLs, …)

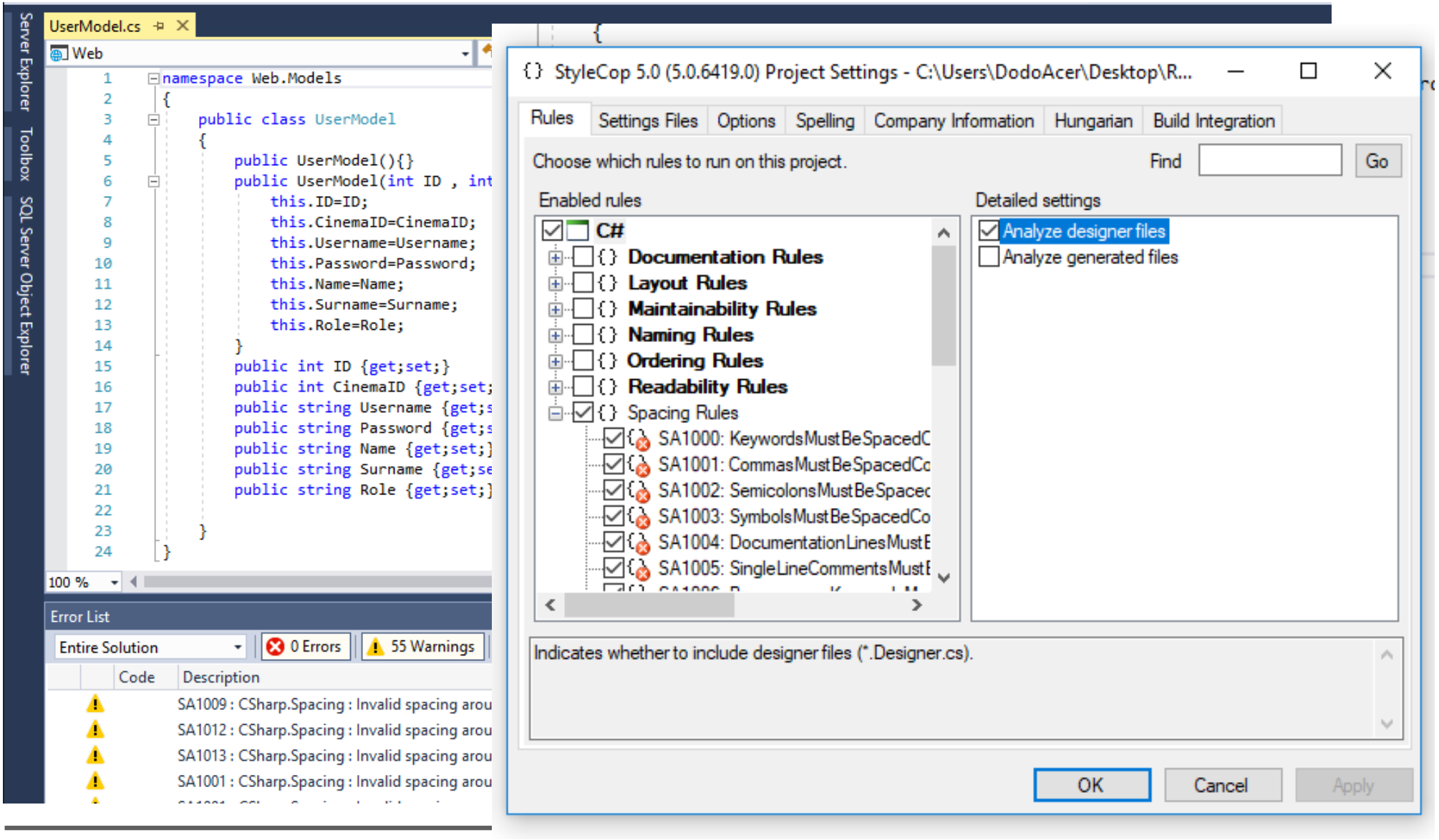
# Alati za statičku analizu

- https://www.owasp.org/index.php/Source_Code_Analysis_Tools

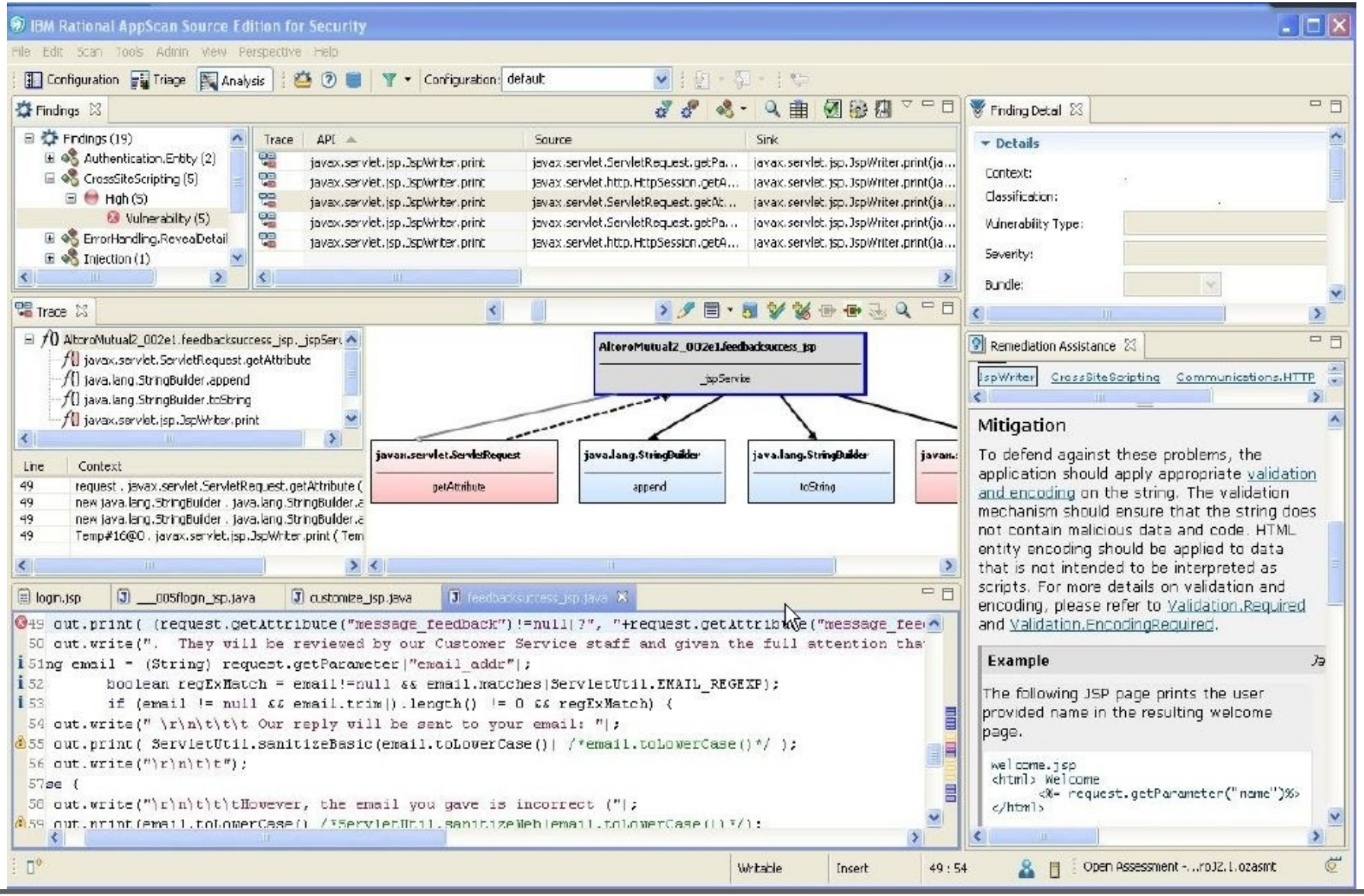
- StyleCop https://github.com/StyleCop/StyleCop - C#
- CodeSmart http://www.axtools.com/ - C#, C++, VB.NET
- NDepend http://www.ndepend.com/ - C#, jDepend za Javu

- VS Code Analysis (FxCop, Roslyn analyzers) - C#, C/C++, …
- PMD - Java, C, C++, C#, Groovy, PHP, Ruby, Fortran, JavaScript, PLSQL, ...
- Fortify Source Code Analyzer - 25 jezika
- Checkstyle - Java
- Klocwork K7 Suite - Java
- FindBugs, Find Security Bugs - Java
- Coverity Prevent - C#, clang, gcc

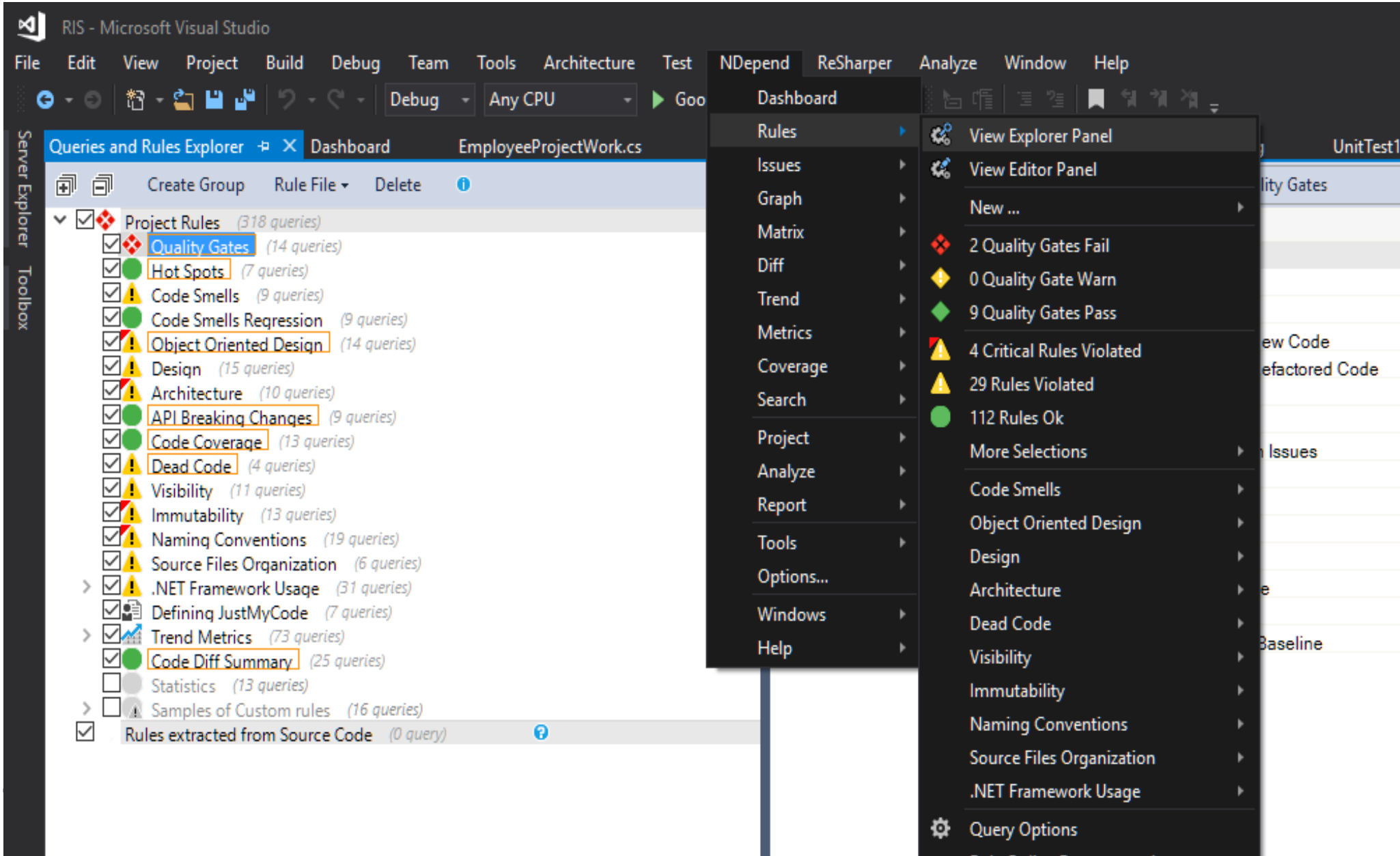
# Primjer: StyleCop



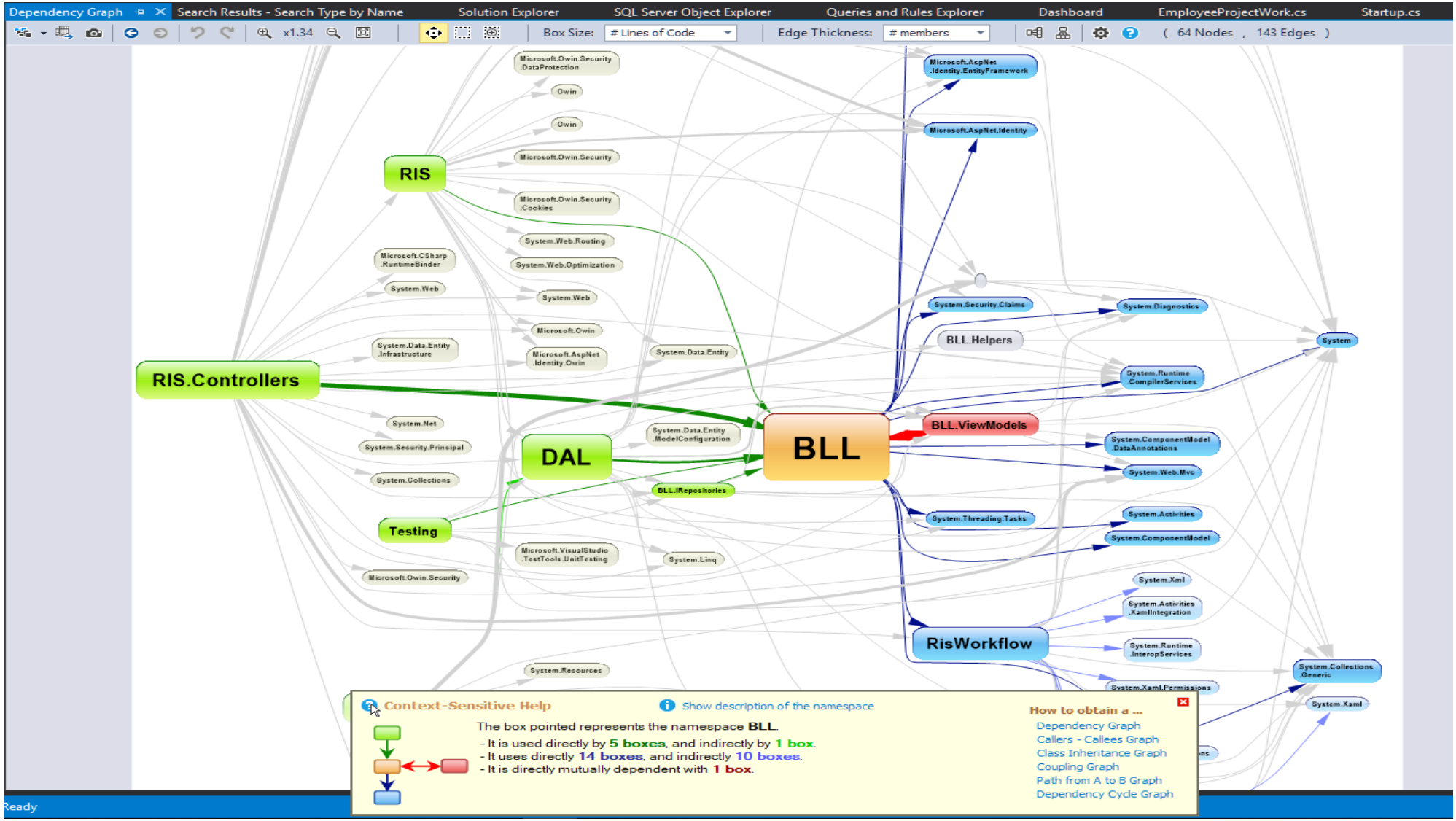
# Primjer: IBM Rational Appscan Source Edition for Security



# Primjer: NDepend



# Primjer: zavisnost komponenti aplikacije



## Dinamička provjera

Dynamic Analysis

Fuzzing

Penetration Testing

# Fuzzing - "pročešljavanje" (eng. fuzz = dlačica)

- DAST (The Good, the Bad and the Ugly)
  - 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, pr. *Buffer overflows, Integer overflows,...*
  - složena primjena na tehnologije (Web 2.0, JSON, Flash, HTML 5.0, Jscript)
  - relativno dugo trajanje (permutiranja uzoraka neispravnih podataka)

# Postupci

- Glupo = Dumb (mutational) fuzzing
  - dovoljno manje znanja o cilju i alatima
  - pseudoslučajne anomalije ispravnih podataka
  - posljedica
    - potrebno više analize
    - redundancija nalaza

```
Standard HTTP GET request
   GET /index.html HTTP/1.1
Anomalous requests
   AAAAAA...AAAA /index.html
        HTTP/1.1
   GET ///////index.html HTTP/1.1
   GET %n%n%n%n%n.html HTTP/1.1
   GET /AAAAAAAAAAAAA.html HTTP/1.1
   GET /index.html
        HTTTTTTTTTTTTP/1.1
   GET /index.html
        HTTP/1.1.1.1.1.1.1.1
```

- Pametno = Smart (generational) fuzzing
  - podaci generirani na temelju modela
  - zahtijeva dubinsko poznavanje cilja i specijaliziranih alata
  - smišljene anomalije poznavanjem formata, standarda, ... (PDF, RFC)
  - posljedica
    - manja potreba za analizom
    - manje dupliciranje nalaza

# Alati za dinamičku analizu

- https://www.owasp.org/index.php/Category:Vulnerability_Scanning_Tools

- CERT Basic Fuzzing Framework (BFF) i Failure Observation Engine (FOE)
  - Otvoreni kod https://github.com/CERTCC/certfuzz

- Peach Fuzzer - *automated security testing platform*
- WebScarab - analiza aplikacija koje koriste HTTP/HTTPS, *intercepting proxy*
- Burp - web aplikacije, buffer overflow, CSS, SQL injection, …
- Fuddly - *fuzzing and data manipulation framework (for GNU/Linux)*
- Hongfuzz - *general fuzzer*

- *Profileri, …*

# 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

- Vrste provjere prema raspoloživosti informacija
  - bez dostupnih informacija (eng. *black-box test*) - kao pravi napad
  - sa svim informacijama (eng. *white-box test*) - najgori slučaj, kad napadač sve zna, ili simulacija napada od strane unutrašnjeg napadača
  - s djelomično dostupnim informacijama (eng. *gray-box test*) - hibrid
- Kriterij početne točke testa
  - Vanjski - s udaljene lokacije (Interneta) prema javno dostupnim sustavima
  - Unutrašnji - s intraneta, simulacija incidenta neovlaštenog pristupa unutrašnjoj mrežnoj infrastrukturi
- 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

| Alat | Osnovna funkcionalnost | Faza |
|---|---|---|
| **Harvester** | Istraživanje javno dostupnih informacija korištenjem tražilica, društvenih mreža itd. | Istraživanje |
| **Nmap** | Istraživanje mreže i skeniranje portova. | Skeniranje |
| **QualysGuard (Network)** | Mrežno skeniranje poznatih ranjivosti. | Skeniranje |
| **QualysGuard (WAS)** | Skeniranje ranjivosti web aplikacija. | Skeniranje |
| **ASPauditor** | Identifikacija ranjivih i loše konfiguriranih ASP.NET poslužitelja. | Skeniranje |
| **Nikto** | Skeniranje web poslužitelja na razne propuste, kao što opasne datoteke itd. | Skeniranje |
| **ZAP** | Multifunkcionalni alat za penetracijsko testiranje web aplikacija. | Skeniranje |
| **Sqlmap** | Otkrivanje i iskorištavanje ranjivosti SQL umetanja. | Provođenje napada |
| **Metasploit** | Multifunkcionalna platforma za iskorištavanje ranjivosti. | Provođenje napada |
| **HTTP DoS** | Uskraćivanje usluge na aplikacijskom sloju. | Provođenje napada |
| **Hydra** | Udaljeno probijanje lozinki. | Provođenje napada |
| **Wireshark** | Snimanje i analiza mrežnog prometa. | Provođenje napada |

# Primjer: Nmap i QualysGuard

```
root@bt:~# nmap -sS -sV 22.22.22.22
Host is up (0.0027s latency).
Not shown: 992 filtered ports
PORT     STATE  SERVICE       VERSION
21/tcp   open   ftp           Microsoft ftpd
25/tcp   open   smtp?
80/tcp   open   http          Microsoft IIS httpd 6.0
110/tcp  open   pop3          Microsoft Windows 2003 POP3 Service1.0
443/tcp  open   ssl/http      Microsoft IIS httpd 6.0
3389/tcp open   microsoft-rdp Microsoft Terminal Service
```

Summary of Vulnerabilities

| Vulnerabilities Total | 24 | Security Risk (Avg) | 5.0 |
|---|---|---|---|

by Severity

| Severity | Confirmed | Potential | Information Gathered | Total |
|---|---|---|---|---|
| 5 | 1 | 0 | | |
| 4 | 0 | 0 | | |
| 3 | 0 | 0 | | |
| 2 | 1 | 0 | | |
| 1 | 0 | 0 | | |
| Total | 2 | 0 | | |

5 Biggest Categories

| Category | Confirmed | Potential |
|---|---|---|
| Information gathering | 0 | 0 |
| TCP/IP | 0 | 0 |
| Windows | 2 | 0 |
| Web server | 0 | 0 |
| CGI | 0 | 0 |
| Total | 2 | 0 |

5 Microsoft Windows Remote Desktop Protocol Remote Code Execution

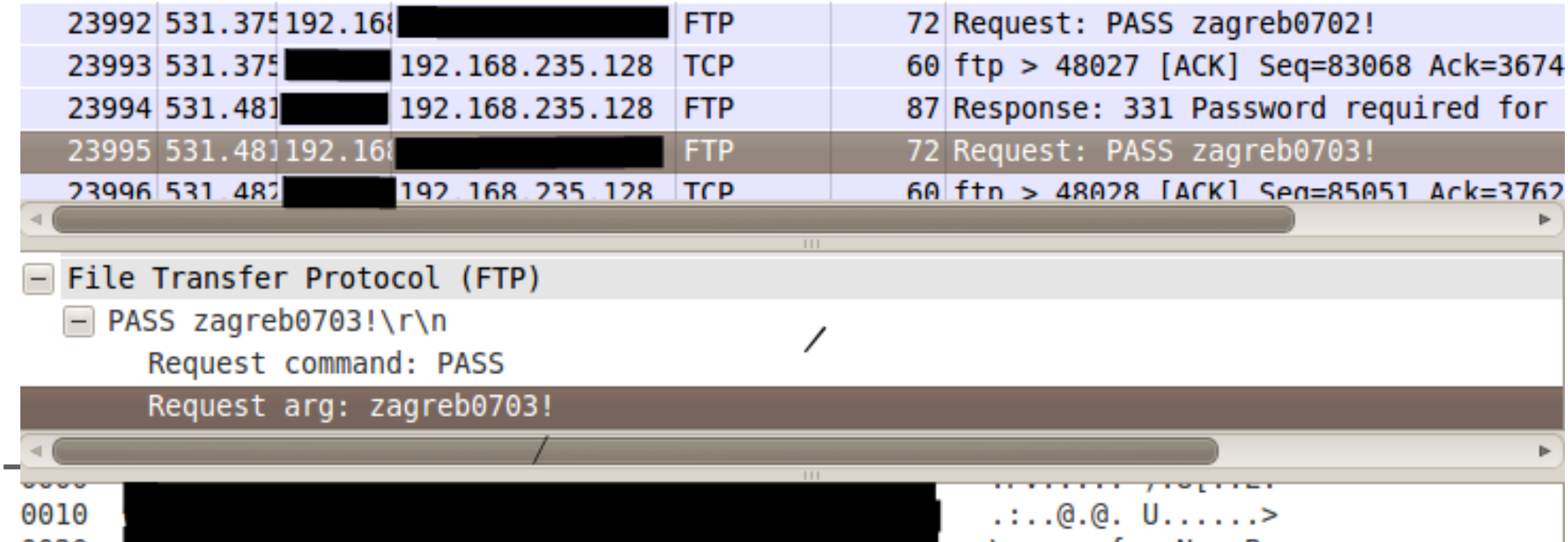
| | |
|---|---|
| QID: | 90783 |
| Category: | Windows |
| CVE ID: | CVE-2012-0002, CVE-2012-0152 |
| Vendor Reference: | MS12-020 |
| Bugtraq ID: | - |
| Service Modified: | 03/29/2012 |
| User Modified: | - |

# Primjer: Hydra, Wireshark

- Napad rječnikom alatom Hydra

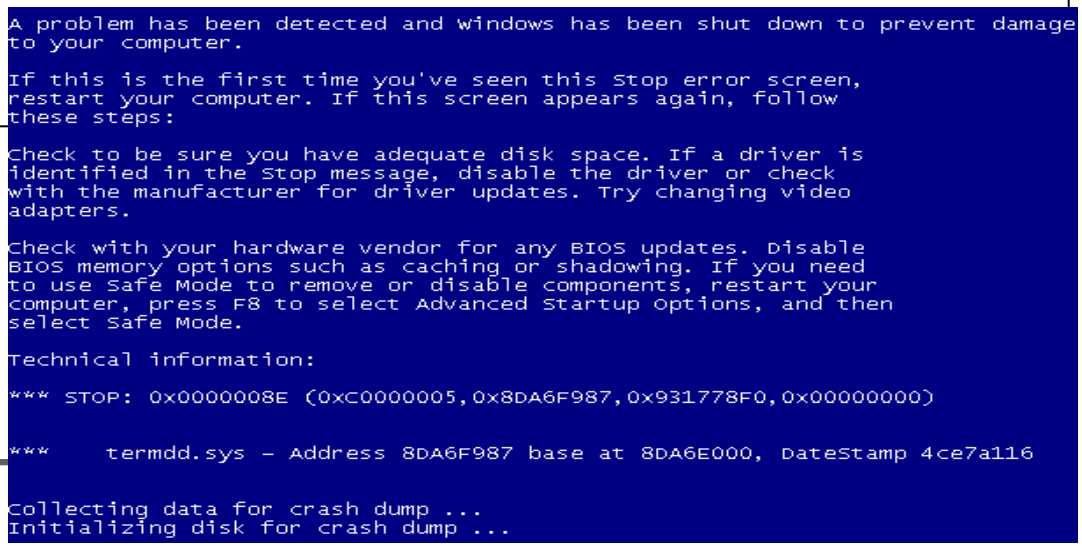
```
[STATUS] 446.66 tries/min, 187152 tries in 06:59h, 0 todo in 00:01h
[STATUS] attack finished for 22.22.22.22 (waiting for children to finish)
1 of 1 target successfuly completed, 0 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2012-06-10 17:20:30
```

- Autentifikacijski podaci tijekom napada snimljeni alatom Wireshark



# Primjer: Metasploit

```
msf > use auxiliary/dos/windows/rdp/ms12_020_maxchannelids
msf auxiliary(ms12_020_maxchannelids) > show options
Module options (auxiliary/dos/windows/rdp/ms12_020_maxchannelids):
   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOST                    yes       The target address
   RPORT   3389             yes       The target port
msf auxiliary(ms12_020_maxchannelids) > set RHOST 11.11.11.11
RHOST => 11.11.11.11
msf auxiliary(ms12_020_maxchannelids) > exploit
[*] 11.11.11.11:3389 - Sending MS12-020 Microsoft Remote Desktop Use-After-
Free DoS
[*] 11.11.11.11:3389 - 210 bytes sent
[*] 11.11.11.11:3389 - Checking RDP
[+] 11.11.11.11:3389 seems down
```

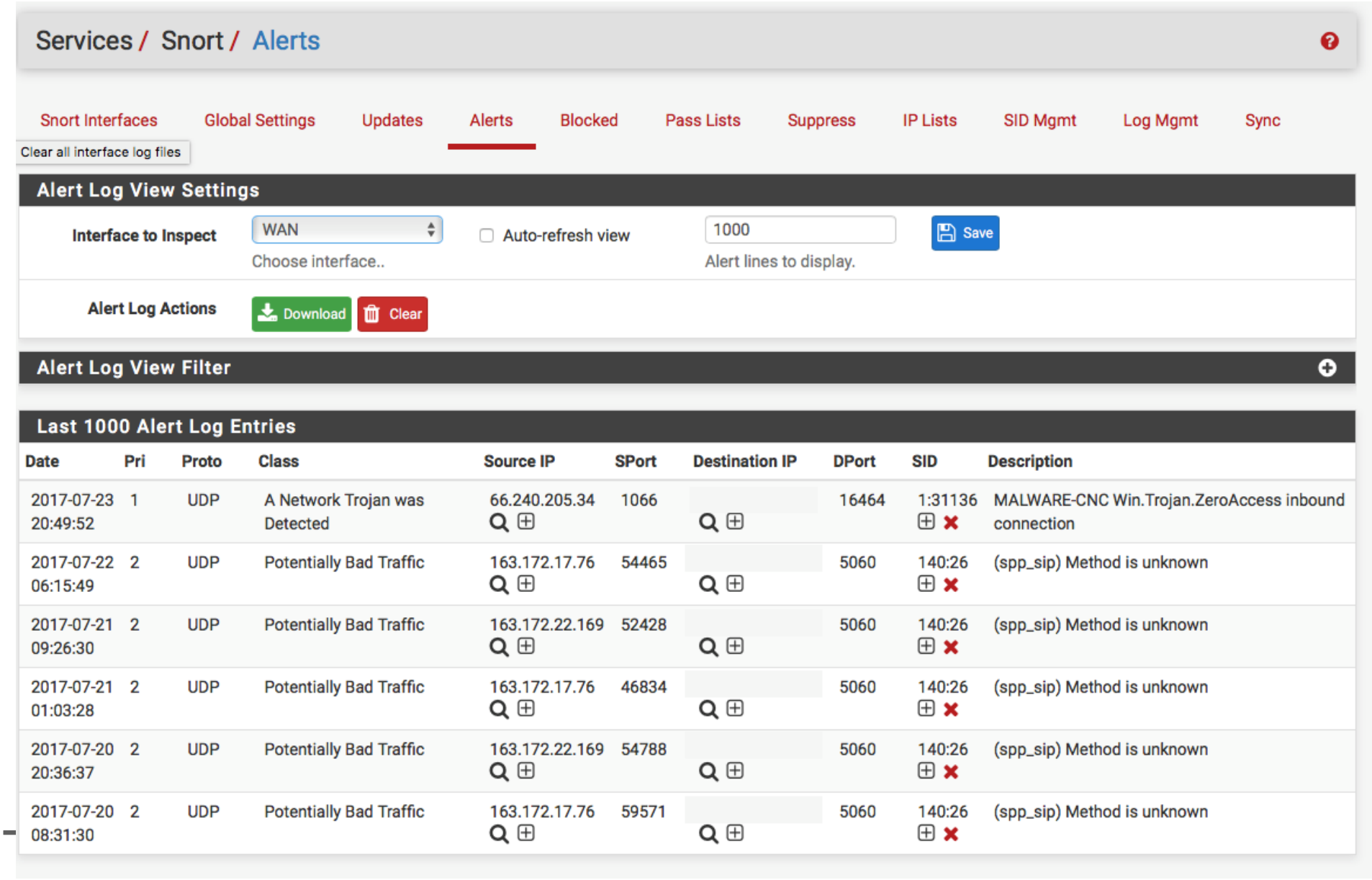


# Alati za penetracijsko testiranje i detekciju upada

- Pentest
  - https://www.owasp.org/index.php/Category:Penetration_Testing_Tools

  - Aircrack-ng - WIFI skaner - otvoreni kod
  - Burp Suite - skaner web ranjivosti
  - Cain & Abel - „password recovery tool" - packet sniffer, password cracker, …
  - Ettercap - suite for man in the middle attacks - otvoreni kod
  - John The Ripper - password cracker
  - Nessus - skener ranjivosti - free trial
  - Kismet - mrežni detektor, packet sniffer, IDS - freeware
  - Zed Attack Proxy (**ZAP**) - web application security scanner - Apache 2 License
- ID/PS
  - https://www.comparitech.com/net-admin/network-intrusion-detection-tools/
  - **Snort**, OSSEC, …, Solarwinds Log and Event Manager, …

# Primjer: Snort



Services / Snort / Alerts

Snort Interfaces | Global Settings | Updates | Alerts | Blocked | Pass Lists | Suppress | IP Lists | SID Mgmt | Log Mgmt | Sync

Clear all interface log files

**Alert Log View Settings**

Interface to Inspect: WAN — Choose interface..

Auto-refresh view

1000 — Alert lines to display.

Save

Alert Log Actions: Download | Clear

**Alert Log View Filter**

**Last 1000 Alert Log Entries**

| Date | Pri | Proto | Class | Source IP | SPort | Destination IP | DPort | SID | Description |
|---|---|---|---|---|---|---|---|---|---|
| 2017-07-23 20:49:52 | 1 | UDP | A Network Trojan was Detected | 66.240.205.34 | 1066 | | 16464 | 1:31136 | MALWARE-CNC Win.Trojan.ZeroAccess inbound connection |
| 2017-07-22 06:15:49 | 2 | UDP | Potentially Bad Traffic | 163.172.17.76 | 54465 | | 5060 | 140:26 | (spp_sip) Method is unknown |
| 2017-07-21 09:26:30 | 2 | UDP | Potentially Bad Traffic | 163.172.22.169 | 52428 | | 5060 | 140:26 | (spp_sip) Method is unknown |
| 2017-07-21 01:03:28 | 2 | UDP | Potentially Bad Traffic | 163.172.17.76 | 46834 | | 5060 | 140:26 | (spp_sip) Method is unknown |
| 2017-07-20 20:36:37 | 2 | UDP | Potentially Bad Traffic | 163.172.22.169 | 54788 | | 5060 | 140:26 | (spp_sip) Method is unknown |
| 2017-07-20 08:31:30 | 2 | UDP | Potentially Bad Traffic | 163.172.17.76 | 59571 | | 5060 | 140:26 | (spp_sip) Method is unknown |

# Reference

- OWASP Category: Vulnerability Scanning Tools
- OWASP Code Review Guide
- OWASP Testing Guide

- Još alata …
  - http://sectools.org/
  - https://www.owasp.org/index.php/Appendix_A:_Testing_Tools
  - Software Security Assessment Tools Review, 2009