# شیوه های تشخیص نقاط ضعف نرم افزار از روی سورس کد

محمد ولی زاده - دکتر محمدحسام تدین
پژوهشگاه ارتباطات و فناوری اطلاعات

www.MohandesPishegan.com
www.itrc.ac.ir

# عناوین بحث

- مفهوم نقطه ضعف نرم افزار
- ضرورت بررسی نقاط ضعف سیستم در لایه برنامه کاربردی
- طبقه بندی انواع خطاهای نرم افزاری
- سطوح بررسی نقاط ضعف نرم افزار
- اهمیت بررسی نقاط ضعف نرم افزار از روی سورس کد
- استانداردهای تشخیص نقاط ضعف برنامه نویسی
- روشهای تشخیص نقاط ضعف برنامه
- ابزارهای تشخیص نقاط ضعف برنامه

# مفهوم نقطه ضعف نرم‌افزار (Software Weakness)

- هر گونه خطا در تحلیل، طراحی یا پیاده سازی نرم افزار که سیستم یا شبکه را در مقابل حملات آسیب پذیر می نماید.
- نمونه هایی از آسیب پذیری‌های نرم افزاری عبارتند از:
  - Validity problems –
  - resource management errors –
  - authentication errors –

# ضرورت بررسی نقاط ضعف سیستم در لایه برنامه کاربردی

- اکثر حملات سایبری جهت بهره برداری (exploit)، نیاز به پیدا کردن نقطه ضعف (weakness) دارند.
- چنین نقاط ضعفی معمولا ناشی از خطاهای نرم افزار (software flaws) در سطح برنامه کاربردی است که می تواند امنیت کل سیستم را به مخاطره اندازد.
- لذا تضمین نرم افزار (software assurance)، اجتناب ناپذیر است.

## ضرورت بررسی نقاط ضعف سیستم در لایه برنامه کاربردی (ادامه)

- ۹۲٪ آسیب پذیری ها در سطح برنامه کاربردی وجود دارد و نه در سطح شبکه (NIST).
- ۷۵٪ از hack ها در سطح برنامه کاربردی اتفاق می افتد (Gartner).
- اکثر اقلام موجود در فهرست آسیب پذیری های کلیدی (Top 10)، ناشی از کدنویسی و تست ضعیف می باشد (SANS).

# طبقه بندی انواع خطاهای نرم‌افزاری

- خطاهای کارکردی
- خطاهای غیر کارکردی
  - خطاهای کارایی
    - نشتی حافظه
    - نشتی منبع
  - خطاهای امنیتی
    - امکان ورود غیرمجاز به سیستم
    - امکان دور زدن کنترل دسترسی

# سطوح بررسی نقاط ضعف نرم‌افزار

- سطح تحلیل نیاز
  - عدم کنترل دسترسی متمرکز
- سطح طراحی
  - طراحی نامناسب پایگاه داده از دید کارایی
- سطح پیاده سازی
  - استفاده از کتابخانه های آسیب پذیر
- سطح استقرار
  - پیکربندی نامناسب زیرساخت نرم افزاری

# اهمیت بررسی نقاط ضعف نرم‌افزار از روی سورس کد

- اطلاعات پیاده سازی سیستم در سطح سورس کد مشهود است.
- اطلاعات تحلیل و طراحی سیستم در سطح سورس کد منعکس شده است.
- بررسی در سطح سورس کد، یک روش white-box محسوب می شود و خیلی دقیق تر از روشهای black-box می باشد.
- از آنجاییکه سورس کد به صورت فرمال بیان می شود، از ابزارهای زیادی برای تشخیص خودکار خطا می توان استفاده نمود.

# استانداردهای تشخیص نقاط ضعف برنامه نویسی

- CERT Secure Coding Standard (C, C++, Java)
- CWE/SANS Top 25 Most Dangerous Programming Errors
- OWASP Secure Coding Practices
- PCI Data Security Standard
- NIST SAMATE (Software Assurance Metrics And Tool Evaluation)

# روشهای تشخیص نقاط ضعف نرم افزار از روی سورس کد

- مرور دستی کد
- تحلیل ایستا
- تحلیل پویا
- آزمون واحد

# مرور دستی کد
# (Manual Code Review)

- در این روش، سورس کد به صورت دستی توسط یک شخص/تیم جهت استخراج نقاط ضعف برنامه مورد ارزیابی قرار می گیرد.
- ملاحظات ارزیابی سورس کد
  - به صورت فرمال/غیرفرمال
  - توسط تولید کننده/ غیر تولیدکننده
  - درونسپاری/برونسپاری
  - کاملا دستی/نیمه دستی

# مرور دستی کد
# (Manual Code Review)

- مزایا
  - توانایی تشخیص اشتباهات منطقی (به دلیل استفاده از هوش و تجربه انسانی)
  - false-positive های کمتر

- معایب
  - زمانبر بودن (فرد آموزش دیده، در یک روز می تواند حداکثر سه هزار خط از برنامه را مرور و بازبینی نماید)
  - عدم مقیاس پذیری برای برنامه های کاربردی بزرگ
  - عدم تشخیص آسیب پذیریهای ناشی از ارتباطات بین کدها و برنامه ها

# تحلیل ایستا
# (Static Analysis/Test)

- در این روش، ارزیابی نقاط ضعف نرم افزار از طریق تکنیک های تحلیل سورس کد صورت می گیرد.
- Gartner استفاده از این روش را برای تولیدکنندگان نرم افزار ضروری می داند.
- ملاحظات تحلیل ایستای سورس کد
  - استفاده از تکنیک rule (syntax) checking
  - استفاده از تکنیک data flow analysis
  - تحلیل برروی سورس اصلی/کامپایل شده/باینری
  - ترکیب با روش مرور دستی

# تحلیل ایستا
# (Static Analysis/Test)

- مزایا
  - سرعت بالا
  - قابلیت استفاده در برنامه های کاربردی بزرگ (خصوصا بالای صدهزار خط سورس کد)
  - قابلیت تشخیص بهتر خطاهای اعتبارسنجی همچون SQL Injection، XSS Injection و Buffer Overflow

- معایب
  - false-positive های نسبتا بیشتر
  - نیاز به تهیه و یادگیری ابزار مناسب دارد
  - خطاهای منطقی را معمولا نمی تواند تشخیص دهد

# تحلیل پویا
# (Dynamic Analysis)

- در این روش، ارزیابی خطاها و نقاط ضعف نرم افزار از طریق تحلیل برنامه‌ی در حال اجرا صورت می گیرد
- ملاحظات تحلیل پویای برنامه
  - استفاده از روش sampling
  - استفاده از روش instrumentation
    - سورس اصلی/کامپایل شده/باینری
  - ترکیب با روش تحلیل ایستا

# تحلیل پویا
# (Dynamic Analysis)

- مزایا
  - دقت بالا در تشخیص خطاهای زمان اجرا
  - false-positive پایین

- معایب
  - نیاز به تهیه و یادگیری ابزار مناسب دارد
  - فقط برای تشخیص خطاهای زمان اجرا کاربرد دارد
  - برخی از روشهای استفاده از آن، باعث کندی سیستم می گردد

# آزمون واحد
# (Unit Test)

- در این روش، ارزیابی خطاها و نقاط ضعف نرم افزار از طریق پیاده سازی و اجرای آزمونهای برنامه نویسی صورت می گیرد
- ملاحظات آزمون واحد
  - دستی/ابزاری
  - میزان پوشش
  - تشخیص نوع خطاها
  - ترکیب با روش تحلیل ایستا

# آزمون امنیت در سطح برنامه

- اهداف آزمون واحد
  - پیدا کردن خطاهای برنامه
  - بررسی صحت کارکرد برنامه
  - مقاوم سازی کد در برابر مقادیر مرزی
  - بررسی رفتار برنامه در برابر خطاها (error handling)

- اهداف مذکور تناظر یک به یک با اهداف امنیتی دارند.

# آزمون واحد
# (Unit Test)

- مزایا
  - پوشش بالای خطاهای منطقی سیستم
  - false-positive پایین
  - امکان بررسی (اکثریت) مسیرهای برنامه

- معایب
  - تولید آزمونهای واحد زمانبر است
  - در صورت تغییر سورس کد، نیاز به بهنگام سازی دارد
  - عمدتا برای تشخیص خطاهای کارکردی برنامه مناسب است

# ابزارهای تشخیص نقاط ضعف برنامه نویسی

- نوع ابزار
  - ابزار تحلیل ایستا/تحلیل پویا/آزمون واحد
- نحوه تهیه ابزار
  - تولید ابزار/استفاده از ابزار رایگان/خرید ابزار تجاری/اجاره ابزار
- بستر ابزار
  - Java/dotNet/C++/PHP/Legacy
- نحوه ارزیابی ابزار
  - معیار ارزیابی (عمومی، خاص نیازمندیهای یک پروژه یا سازمان)

# ابزارهای تشخیص آسیب پذیری های برنامه (تحلیل ایستا، تحلیل پویا و آزمون امنیت)

# ابزارهای شرکت HP

- HP Fortify for static analysis (Multilingual)
- HP WebInspect for dynamic analysis and penetration testing
- The HP Application Automation Tools plugin for continuous integration

# ابزارهای شرکت Parasoft

- Parasoft Jtest for java (static analysis, unit testing, run-time memory analysis)
- Parasoft dotTest for .Net
- Parssoft C++Test for C and C++
- Parasoft SOATest for functional, security and load testing

# منابع و مراجع


- CWE (Common Weakness Enumeration), cwe.mitre.org
- OWASP Secure Coding Practices, www.owasp.org
- CERT Secure Coding Standards, www.securecoding.cert.org
- SAMATE (Software Assurance Metrics And Tool Evaluation), samate.nist.gov
- PCI SSC Data Security Standards, www.pcisecuritystandards.org/security_standards
- The Importance of Manual Secure Code Review, www.mitre.org
- Static program analysis, Wikipedia, en.wikipedia.org
- Dynamic program analysis, Wikipedia, en.wikipedia.org
- Leveraging unit testing to prevent security vulnerabilities, www.parasoft.com
- https://www.checkmarx.com/technology/static-code-analysis-sca/#sthash.NQeiqug6.dpuf

# با سپاس از توجه و حضور
# سروران گرامی

اسلایدها در Linked-in در گروه زیر به اشتراک گذاشته خواهد شد.

Persian Software Testing & Quality Assurance Group