شیوه های تشخیص خطاهای نرم افزار
در سطح برنامه نویسی

محمد ولی زاده
خرداد نودوچهار

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

عناوین بحث

- طبقه بندی انواع خطاهای نرم افزاری
- سطوح بررسی خطاهای نرم افزار
- اهمیت بررسی خطاهای نرم افزار درسطح برنامه نویسی
- استانداردهای تشخیص خطاهای برنامه نویسی
- روشهای تشخیص خطاهای برنامه
- ابزارهای تشخیص خطاهای برنامه
- دموی ابزارهای تضمین کیفیت در سطح برنامه نویسی

واژه های مرسوم

- bug: خطای پیاده سازی
- flaw: خطای طراحی
- defect: خطای طراحی یا پیاده سازی
- error: خطای انسانی که منجر به خطای نرم افزار می گردد.
- failure: خطایی که به مرحله بروز رسیده است.

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 خطاهای نرم‌افزار (با تمرکز برروی امنیت سیستم)

- 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 خطاهای نرم‌افزار در سطح برنامه نویسی

- 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 ابزارهای زیادی برای تشخیص خودکار خطا در سطح سورس کد می توان استفاده کرد.

استانداردهای تشخیص اشکالات برنامه نویسی (با تمرکز برروی امنیت سیستم)

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

روش های تشخیص خطاهای برنامه

- مرور دستی کد (manual inspection)
- تحلیل ایستا (static analysis)
- تحلیل پویا (dynamic analysis)
- آزمون واحد (unit testing)

مرور دستی کد
(Manual Code Review)

- در این روش، سورس کد به صورت دستی توسط یک شخص/تیم جهت شناسایی خطاهای 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 خطاهای ناشی از ارتباطات بین کدها و برنامه ها

تحلیل ایستا
(Static Analysis/Test)

- در این روش، ارزیابی خطاهای نرم افزار از طریق تکنیک های تحلیل سورس کد صورت می گیرد.
- Gartner استفاده از این روش را برای تشخیص مشکلات امنیتی سیستم در سطح سورس کد 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)
  - شیوه پیشگیرانه (در مقابل تشخیص)

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

میزان بلوغ تحلیل ایستا
Static Analysis Maturity Level

1. **Ad-Hoc**
   - Some individuals (1-2 developers)
   - No source control
   - Rely on individual experience
   - No static test

2. **Reactive**
   - A development team
   - A centralized source control tool
   - Software works well under specific conditions, but frequently fails in real-world deployments.
   - Missed deadlines and failing to meet functionality goals is common.
   - Success depends on how well teams work together

میزان بلوغ تحلیل ایستا
(Static Analysis Maturity Level)

3. Proactive

- Quality and reliability become as important as just getting the software to work.
- Work is split into smaller tasks and visibility into progress is established early.
- Testing becomes a critical release gate
- Some preventative measures are introduced. Static analysis moves from a QA/testing practice to the IDE
- Full test environment is available

4. Managed

- Managing the process becomes the means to control the software, in addition to traditional testing/QA.
- Quality meets expectations with few issues found in the field.
- Testing is rarely bypassed to meet a schedule.
- Test and development environments are virtualized and available on demand.

میزان بلوغ تحلیل ایستا
Static Analysis Maturity Level

5. Optimized

- Corporate policies guide the entire SDLC.
- Software is not only on-time with correct features, but is reliable, secure, and scalable.
- Full traceability and visibility is achieved.
- Change-based testing becomes a reality because the cost of change is known in advance.

تحلیل پویا
(Dynamic Analysis)

- در این روش، ارزیابی خطاهای برنامه 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
  - بهنگام سازی
  - ترکیب با روش تحلیل ایستا

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)

مزایا و معایب آزمون واحد

- 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

میزان بلوغ آزمون واحد
(Unit Testing Maturity Level)

	Ad-Hoc	Reactive	Proactive	Managed	Optimized
Focus	Individual: Developers independently choose to create and run unit tests while developing functionality, but tests are not saved or maintained. Instead, they are isolated on independent machines.	Team: There is a stated Management desire to unit test. Development activity is not aligned with business expectations, and implementation is inconsistent across teams. Due to lack of visibility, adherence requires Management vigilance.	Project/Product: A common unit testing policy is clearly documented, and multiple teams are on board. Developers are standardized on a Development Testing Platform to create and extend unit tests in the normal course of work.	Organizational: Due to Management's increased visibility, the unit testing process is managed by a centrally-defined policy that distributes tasks directly to developer desktops based upon coverage requirements, risk of failure, and other management-defined metrics.	Enterprise: Policies are regularly updated as part of a root-cause analysis effort to prevent defects from being discovered in QA.
Characteristics	Test reports are non-existent or unverifiable. Tests are not maintained in an SCM. No scheduled automation. Reintroduction of defects is accepted as "normal" or "unavoidable."	Unit tests are maintained in an SCM with the product code. Unit tests are run manually during a "smoke test" or "pre-release testing" phase. Tests are primarily host- or simulator-based. Developers use unit testing to find defects.	Every code change includes corresponding new/modified unit tests. On-target testing is common. Scheduled automation of test runs and reports provide full traceability. The organization uses unit testing to monitor and reduce that reintroduction of defects.	Developers resolve unit test failures within 24 hours or less in a Continuous Integration workflow. Fewer tests fail with each change due to pre-commit desktop runs. Function and object virtualization are used to extend coverage and test error conditions.	Unit tests are a verification mechanism to verify that policy and process are in synch. Test results are linkable and bi-directionally traceable to all data associated with software and device development. Traceability extends beyond the traditional borders of the SDLC.

چالش های فنی آزمون واحد

- حذف وابستگی به محیط (mock object)
- رسیدن به پوشش بالا
- جدا سازی test-data از test-script
- انتخاب test-items (class/package/layer/component)

چالش‌های فرایندی آزمون واحد

- امکان جایگزینی آزمون واحد با آزمون سیستم؟!
- امکان جایگزینی آزمون واحد با تحلیل ایستا؟!
- مسئولیت آزمون واحد (تیم تولید/تست/برونسپاری)؟
- زمان تولید و به‌روزرسانی آزمون های واحد؟

ابزارهای تشخیص خطاهای برنامه نویسی

- نوع ابزار
  - ابزار تحلیل ایستا/تحلیل پویا/آزمون واحد/مدیریت فرایند بازبینی
- 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 نقاط ضعف برنامه از دید امنیت
(تحلیل ایستا و تحلیل پویا)

ابزارهای HP

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

ابزارهای Microsoft

- Microsoft Visual Studio Test Tools including:
  - Unit Testing
    - Pex & Code digger plug-in
    - Code Digger uses the Pex engine and Microsoft Research's Z3 constraint solver to analyze all branches in the code, trying to generate a test suite that achieves high code coverage.
  - Static Analysis
    - Phoenix engine for data-flow analysis
  - Profiling

ابزارهای Parasoft

- Parasoft Jtest for Java (static analysis, unit testing, run-time memory analysis)
- Parasoft dotTest for .Net
- Parssoft C++Test for C and C++

منابع و مراجع

- Gartner, Static Application Security Testing
- The Importance of Manual Secure Code Review, www.mitre.org
- Static program analysis, Wikipedia, en.wikipedia.org
- Dynamic program analysis, Wikipedia, en.wikipedia.org
- CWE (Common Weakness Enumeration), cwe.mitre.org
- OWASP Secure Coding Practices, www.owasp.org
- CERT Secure Coding Standards, www.securecoding.cert.org
- PCI SSC Data Security Standards, www.pcisecuritystandards.org/security_standards
- Parasoft, Development Testing
- Microsoft Research, Code Digger
- Microsoft MSDN Guide

دموی چند نمونه ابزار تضمین کیفیت برنامه نویسی

- دموی یک نمونه ابزار تحلیل ایستا
- دموی یک نمونه ابزار تحلیل پویا

با سپاس از توجه و حضور شما

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

Persian Software Testing & Quality Assurance Group