Fault Injection Seminar

Summer Semester 2015

Daniel Richter, Lena Herscheid, Prof. Andreas Polze
Operating Systems and Middleware Group
Hasso Plattner Institute

Dependability Threats

- Fault (Fehlerursache)
  - Adjudged or hypothesized cause for an error
  - In software: bugs / defects
  - Wrong source code or static artefact
  - Software faults are permanent design faults
- Error (Fehlerzustand)
  - System state leading to subsequent failure
- Failure (Ausfall)
  - Deviation from specification or intended service

```
void count(void)
{
    int k;
    for (int i = 0; i < 10; i++) k++;

    printf("%d", k);
}
```

Locals

Name	Value	Type
i	0	int
k	-858993460	int

```
Result: -858993450
```

Dependability Evaluation

- How reliable is a piece of code?
  - Fault activation is environment-dependent
- How reliable is a complex software system?
  - Feature interaction complexity, third party components,…

Two classes of approaches:

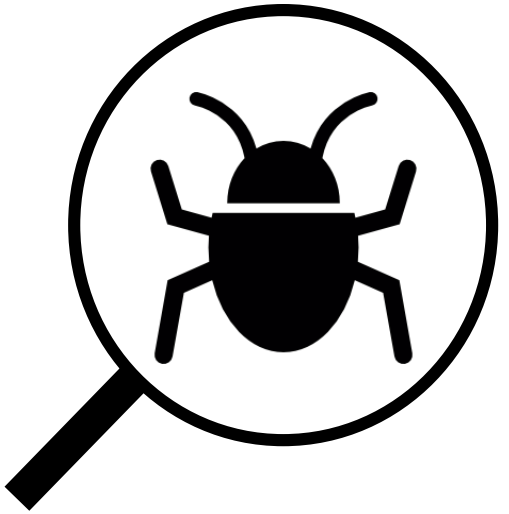
1. Prove the program correct (formal verification)
   - Requires formal specification
   - For all inputs and environments → state space explosion

2. Prove the program wrong (testing)
   - Requires a fault model
   - Discover bugs during runtime → **fault injection**

Fault Injection Testing

- *"…testing […] necessitates **faults** or **errors** to be part of the test patterns…"* – Avizienis et al.
- For fault removal: finding bugs, uncovering lack of fault tolerance
- Fault injection is a software testing method
  - Requiring a workload and a faultload

- In this seminar, we focus on software
  - Software Implemented Fault Injection (SWIFI)
  - Software Fault Injection



Fault Injection – Research Question

- What are the “dependability bottlenecks” / single points of failure?
- How does performance degrade in the presence of faults?
- What is the coverage of error detection and recovery mechanisms?
- How fault tolerant is the system?

- What to inject? → Fault model
- Where and when to inject? → Operational profile

Fault Injection – Targets

- Fault model
  - Set of faults which can be assumed to occur
  - Hardware faults: transient vs permanent, single vs multi bit flip, …
  - Software faults: computation, timing, omission, crash, …
- Purpose
  - Test specific fault tolerance mechanisms
  - Compare different systems' dependability

Fault Injection – Implementation

- Trigger mechanism
  - Time-based: inject faults at periodic time intervals
  - Location-based: inject faults into predetermined memory locations
  - Execution-driven: inject faults based on the control flow during runtime

- Injection time
  - Pre-runtime: e.g. code mutations
  - During runtime: e.g. hardware or software traps
  - At library load time

- Injection level
  - Binary
  - Intermediate code representation
  - Source code

Seminar Topics

Fault Injection testing of Operating Systems

- Operating system robustness is essential to many applications, but not guaranteed by general purpose OSes
- Ballista: Testing at function call level, successfully used in many cases
- CrashMe: Simple tool attempting to execute random byte sequences

Koopman, Philip, and John DeVale. "Comparing the robustness of POSIX Operating Systems." *Fault-Tolerant Computing, 1999. Digest of Papers. Twenty-Ninth Annual International Symposium on*. IEEE, 1999.

http://users.ece.cmu.edu/~koopman/ballista/

Forrester, Justin E., and Barton P. Miller. "An empirical study of the robustness of Windows NT applications using random testing." *Proceedings of the 4th USENIX Windows System Symposium*. 2000.

https://crashme.codeplex.com/

Fault Injection in Managed Code

- The CLR provides powerful capabilities for testing software
- Microsoft TestApi
  - Runtime fault injection API for .NET
  - Uses the CLR profiling API under the hood
  - Customizable fault rules: when and what fault to trigger

https://msdn.microsoft.com/en-us/magazine/ff898404.aspx

https://testapi.codeplex.com/

Fault Injection Patterns in Java

- Fault injection code should be separated from the application code
- How to build a extensible, modular fault injection system in OO languages?
- Approach based on Java reflection features

Leme, Nelson GM, Eliane Martins, and Cecília MF Rubira. "A software fault injection pattern system." *Proceedings of the IX Brazilian Symposium on Fault-Tolerant Computing*. 2001.

Martins, Eliane, Cecilia MF Rubira, and Nelson GM Leme. "Jaca: A reflective fault injection tool based on patterns." *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*. IEEE, 2002.

http://www.ic.unicamp.br/~eliane/JACA.html

Fault Injection into Libraries

- Applications often rely on shared libraries without knowing their dependability
  - Incomplete documentation
  - Too many API functions to test manually
  - Frequently closed source
- Inject faults into the interface between application and library

Marinescu, Paul Dan, and George Candea. "LFI: A practical and general library-level fault injector." *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*. IEEE, 2009.

https://github.com/dslab-epfl/lfi

Fault Injection using AOP

- Faults/errors can be injected by modifying the executed source code
  - Manual recompilation of each modified version (mutant) is cumbersome
  - How to automate?
- Apply Aspect Oriented Programming (AOP) is for generating mutants
  - Dynamic code insertion
  - Non-intrusive

Bogacki, Bartosz, and Bartosz Walter. "Aspect-oriented response injection: an alternative to classical mutation testing." *Software Engineering Techniques: Design for Quality*. Springer US, 2007. 273-282.

https://josefbetancourt.wordpress.com/2011/11/13/exception-verification-aop/

Fault Injection in Distributed / Cloud Systems

- Distributed systems need to handle faults in all software stack layers
  - Node crashes
  - Network partitions
  - Software design flaws
- Fault tolerance is an essential characteristic of distributed systems
  - Usually, implemented using redundancy
- ChaosMonkey: randomly kill instances to test fault tolerance

http://techblog.netflix.com/2012/07/chaos-monkey-released-into-wild.html

https://github.com/Netflix/SimianArmy/wiki/Chaos-Monkey

GPU Fault Injection

- GPUs are faulty, as they were designed for applications where small errors don’t matter (image rendering)
- With recent general purpose applications of GPUs, reliability is becoming more important
- GPU-Qin: injects transient hardware faults

Fang, Bo, et al. "GPU-Qin: A methodology for evaluating the error resilience of GPGPU applications." *Performance Analysis of Systems and Software (ISPASS), 2014 IEEE International Symposium on*. IEEE, 2014.

https://github.com/DependableSystemsLab/GPU-Injector

Seminar Organization

- Seminar slots: Thursday, 13:30 – 15:00
  - Up to two presentations per slot
- First student presentation: May 21st

- Email a prioritised list of three topics
  - To: lena.lerscheid@hpi.de, daniel.richter@hpi.de
  - By April 30th
  - Own topic suggestions (with a short description) are also welcome!

Presentation – Requirements

- 30 minutes presentation
- 15 minutes discussion
- Consult your supervisor one week before the presentation
- Grading is based on presentation and discussion

Topics to discuss

- What is the intended system under test?
- What is the fault model?
- Characterize the approach:
  - Trigger mechanism
  - Injection time
  - Injection level
- Discuss:
  - Fault coverage
  - Intended use cases
- Present a small demonstration / practical example / showcase!

Additional Sources

- Avizienis, Algirdas, et al. "Basic concepts and taxonomy of dependable and secure computing." *Dependable and Secure Computing, IEEE Transactions on*1.1 (2004): 11-33.
- Natella, Roberto, et al. "On fault representativeness of software fault injection."*Software Engineering, IEEE Transactions on* 39.1 (2013): 80-96.
- Ziade, Haissam, Rafic A. Ayoubi, and Raoul Velazco. "A survey on fault injection techniques." *Int. Arab J. Inf. Technol.* 1.2 (2004): 171-186.