DS-Bench Toolset:
Tools for Dependability Benchmarking with Simulation and Assurance

Hajime Fujita (U. of Tokyo, Japan)[*1]
Yutaka Matsuno (U. of Tokyo, Japan) [*2]
Toshihiro Hanawa (U. of Tsukuba, Japan)
Mitsuhisa Sato (U. of Tsukuba, Japan)
Shinpei Kato (UC Santa Cruz, USA) [*2]
Yutaka Ishikawa (U. of Tokyo, Japan)

[*1] Currently in the University of Chicago, USA
[*2] Currently in Nagoya University, Japan

This work has been conducted as a part of the “Dependable Embedded Operating System for Practical Use (DEOS)” project, funded by JST (Japan Science and Technology Agency).
Introduction (1)

- Increasing demands for highly dependable system
  - Today’s society totally depends on information systems, and suspension of services cost a lot
- A complex system involves a lot of stakeholders (e.g. developer, supplier, user, etc…)
- All of them must agree on what is the “dependability” for their system, e.g. minimum throughput or maximum latency
  - The dependability of the system should be expressed clearly and supported by clear evidences so that every stakeholder agrees that the system is in fact dependable
Introduction (2)

- Some argument on system’s dependability may require quantitative evaluation of the system
- An automated benchmark testing tool is needed
  - Systems are getting more parallel and distributed
  - Testing takes much time and cost
- Results of the tests should be automatically collected as evidences of dependability
DS-Bench Toolset: Overview
D-Case Editor

DSN 2012, Boston, MA, USA
D-Case [Matsuno PRDC2010]

- A kind of Assurance Cases with development tools and runtime monitoring systems
- Assurance Cases
  - “A documented body of evidence that provides a convincing and valid argument that a system is adequately dependable for a given application in a given environment” [Adelard]
  - Becoming a standard for safety-critical systems
  - A Graphical notations GSN (Goal Structuring Notation)
D-Case Editor

- A free assurance case editor
  - An Eclipse plugin using Eclipse GMF
  - Supports GSN
- Key Features
  - Variable type checking and pattern library [Matsuno QSIC2011]
  - Conducting benchmark tests using DS-Bench
DS-Bench

DS-Bench Controller

Benchmark Databases
- Benchmark Scenario
- Anomaly Generator
- Benchmark Programs
- Benchmark Results

Target Machines (Virtual, Cloud)

Target Machines (Physical)

D-Case Editor

D-Cloud Controller

Tester
DS-Bench: Overview

- A framework for automatic benchmark tests
  - Dependability metrics are defined and measured by each program. DS-Bench itself does not define them.
- Supports multiple benchmark programs
  - Existing benchmark programs, as well as user-developed ones, can be executed on DS-Bench

Diagram:
- DS-Bench Controller
- Benchmark Database
- Target Machines
- Benchmark A
- Benchmark B
- Benchmark C
- Deploying & executing benchmark programs
- Returning results
DS-Bench: Benchmark Results

- Handling outputs from various benchmark programs
  - Output style of benchmark programs may vary
  - Usually the output is pre-formatted for human readability
- Benchmark description is prepared for each benchmark program
  - Describes cutting rules for interpreting the raw result so that a text table can be converted to a list of machine readable values
- Converted results are stored in an XML database

```xml
<dsbxml>

  Client connecting to tammie07, TCP port 5001
  TCP window size: 0.02 MByte (default)

  [ 4] local 172.16.128.134 port 51562 connected with
  172.16.128.135 port 5001

  [ 4] 0.0- 1.0 sec  113 MBytes  940 Mbits/sec
  [ 4] 1.0- 2.0 sec  113 MBytes  944 Mbits/sec

  ... 
</dsbxml>
```
DS-Bench: Anomaly Loads

- Anomaly loads simulate several irregular situations that may occur to the system
  - E.g. Whole machine failure, device failure, performance degradation, …

- Essential for dependability benchmark testing
  - We want to know if the system is still dependable under such conditions
### DS-Bench: Pre-installed Programs

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bonnie++</td>
<td>I/O benchmark</td>
</tr>
<tr>
<td>lmbench</td>
<td>Generic performance benchmark</td>
</tr>
<tr>
<td>hackbench</td>
<td>System benchmark; creates a lot of processes</td>
</tr>
<tr>
<td>httperf</td>
<td>Measures performance of HTTP servers</td>
</tr>
<tr>
<td>iperf</td>
<td>Measures network bandwidth</td>
</tr>
<tr>
<td>cpustress</td>
<td>Stresses CPU; just consumes CPU time</td>
</tr>
<tr>
<td>memstress</td>
<td>Stresses memory; just consumes memory</td>
</tr>
<tr>
<td>netcmd</td>
<td>Injecting network anomalies; delay, packet drop, reordering</td>
</tr>
<tr>
<td>terminator</td>
<td>Kills a process</td>
</tr>
</tbody>
</table>

Each program may be used as a benchmark program or anomaly generator. Some of them may be used as both.

e.g. bonnie++ can be used as a benchmark program to measure I/O bandwidth, as well as an anomaly generator that consumes I/O bandwidth.
Benchmark programs and anomaly loads are executed concurrently in a specific timing.
Each scenario may define input parameters and an output result

- Input parameters are adjustable from D-Case Editor
- An output result can be obtained from D-Case Editor

Example:

**Scenario:**
- Client machines access to a web server cluster
- A power failure occurs to one server node
- The server has an automatic failover mechanism

**Input Parameter:**
- Request frequency [reqs/s]

**Output:**
- Maximum observed latency [ms]
D-Cloud: Overview

- D-Cloud manages hardware resources needed for conducting benchmark tests
- Two types of computing resources are provided
  - **Physical machines**
    - For performance-sensitive tests
  - **Virtual machines**
    - Managed by OpenStack, a management software for private clouds
    - An arbitrary number of virtual machines can be created simultaneously
    - For functional, performance-insensitive tests
D-Cloud: Fault Injection

- D-Cloud also performs fault injection
  - Adding anomalies from the outside of target machines
  - Mostly requires special equipment (e.g. intelligent PDUs, IPMI, SNMP-enabled switch), or special software (e.g. VMM)

- FaultVM [Banzai CLOUD2010][Hanawa PRDC2010]
  - A virtual machine monitor that comes with a hardware failure simulation feature
  - Based on QEMU, an open source full system emulator

- Fault injection functions are exported to DS-Bench and can be used as anomaly generators
  - i.e. These fault injectors can be put in a benchmark scenario
## D-Cloud: List of Fault Injection Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LinkRefuse</td>
<td>Physical Machine</td>
<td>Shuts down a network switch port</td>
</tr>
<tr>
<td>SupplyRefuse</td>
<td>Physical Machine</td>
<td>Shuts down a power supply to a target</td>
</tr>
<tr>
<td>IPMICtrl</td>
<td>Physical Machine</td>
<td>Controls a power status of a target via IPMI</td>
</tr>
<tr>
<td>VMMemFlip</td>
<td>Virtual Machine</td>
<td>Injects memory flip into VM</td>
</tr>
<tr>
<td>VMStop</td>
<td>Virtual Machine</td>
<td>Pauses a VM</td>
</tr>
<tr>
<td>VMNicFault</td>
<td>Virtual Machine</td>
<td>Injects a various fault into VM’s virtual NIC device</td>
</tr>
</tbody>
</table>

Fault injection methods for physical machines are implemented as a program, and can be added by users just as benchmark programs.
Typical Workflow with the Toolset

1. Determine the required dependability metrics and describe it in D-Case
2. Find a suitable benchmark scenario in DS-Bench database
   - If not exist, create new one
3. Adjust parameters and run the benchmark test from D-Case Editor
4. Result is returned to D-Case Editor and recorded as an evidence for D-Case
Demonstration

- Evaluating a web server system
- Dependability requirement
  - Keep the access latency lower than 3s even one server node fails
- Environment
  - The server provides a single system view using SSPA, a load balancing/high availability mechanism [Fujita HASE2011]

![Diagram of web server system evaluation](image-url)
Demo: Creating a New Scenario

DS-Bench

- Main
  - Synopsis
    - New Benchmark Scenario
    - Load Benchmark Scenario
  - View Result
  - Configuration

Benchmark Scenario Name | Description
------------------------|-------------
httpperf               |             
bonnie                 |             
View Result            |             
Configuration          |             
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a New Scenario
Demo: Creating a D-Case Diagram
Demo: Creating a D-Case Diagram
Demo: Creating a D-Case Diagram
Demo: Creating a D-Case Diagram
Demo: Creating a D-Case Diagram
Demo: Selecting a Benchmark Scenario
Demo: Selecting a Benchmark Scenario
Demo: Selecting a Benchmark Scenario
Demo: Selecting a Benchmark Scenario
Demo: Selecting a Benchmark Scenario
Demo: Setting Benchmark Parameters
Demo: Setting Benchmark Parameters
Demo: Setting Benchmark Parameters
Demo: Benchmark Execution
Demo: Benchmark Execution
Demo: Benchmark Execution

DS-Bench

Main
- New Benchmark Scenario
- Load Benchmark Scenario

Synopsis
- Benchmark Scenario Name
  - demo
  - httperf
  - bonnie
- View Result
- Configuration

View Result

Configuration

http://192.168.100.121/cgi-bin/synopsiscgi
## Demo: Benchmark Execution

### DS-Bench

<table>
<thead>
<tr>
<th>ID</th>
<th>Date</th>
<th>Status</th>
<th>Comment</th>
<th>Benchmark Program</th>
<th>Anomaly Load</th>
<th>Starter</th>
<th>View Scenario detail</th>
<th>Stop</th>
</tr>
</thead>
<tbody>
<tr>
<td>353</td>
<td>2012/06/21 18:39:36</td>
<td>run (10%) 1min</td>
<td></td>
<td>Httperf Httperf</td>
<td>NetCMD</td>
<td>external</td>
<td></td>
<td></td>
</tr>
<tr>
<td>321</td>
<td>2012/06/21 14:03:25</td>
<td>stopping</td>
<td></td>
<td>Httperf Httperf</td>
<td>NetCMD</td>
<td>gui</td>
<td></td>
<td></td>
</tr>
<tr>
<td>289</td>
<td>2012/06/21 11:15:43</td>
<td>install</td>
<td></td>
<td>Httperf Httperf</td>
<td>NetCMD</td>
<td>gui</td>
<td></td>
<td></td>
</tr>
<tr>
<td>156</td>
<td>2012/06/14 12:33:21</td>
<td>stopping</td>
<td></td>
<td>Httperf Httperf</td>
<td>NetCMD</td>
<td>gui</td>
<td></td>
<td></td>
</tr>
<tr>
<td>108</td>
<td>2012/06/11 15:01:37</td>
<td>stopping</td>
<td>test</td>
<td>Memstress</td>
<td>gui</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>99</td>
<td></td>
<td>stopping</td>
<td></td>
<td></td>
<td>gui</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>2012/06/08 10:42:56</td>
<td>stopping</td>
<td>test</td>
<td>Memstress</td>
<td>gui</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>84</td>
<td></td>
<td>st_error DS-Bench</td>
<td>Error: failure to release calculation resource.</td>
<td></td>
<td>gui</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td></td>
<td>stopping</td>
<td></td>
<td></td>
<td>gui</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Demo: Benchmark Execution
Demo: Benchmark Execution
Demo: Benchmark Execution
Demo: Benchmark Execution
Demo: Benchmark Result
Demo: Benchmark Result
Demo: Benchmark Result
Related Work

- **ASCE Tool [Adelard]**
  - An well-used assurance case editor. Mainly for generating certification documents

- **D-Bench [Kanoun 2004, Durães 2004]**
  - A project aimed at establishing dependability benchmarking methods for several domains
  - Proposed dependability metrics and measurement tools for several target domains, however did not provide a total software framework to integrate multiple benchmarks

- **Fault injection tools**
  - As far as we know, no effort have been made to combine assurance cases and benchmark results dynamically
Summary

- An automated tool for dependability benchmarking, with dependability assurance, is needed
- DS-Bench Toolset
  - D-Case Editor, an assurance case editor
  - DS-Bench, a framework for benchmark test
  - D-Cloud, a system for managing hardware resources for benchmark test
- Tight collaboration between D-Case Editor and DS-Bench
  - Dependability requirements are described in D-Case
  - DS-Bench conducts benchmark tests to obtain a quantitative evidence for D-Case
Questions?

- D-Case Editor is available from
  - http://www.dependable-os.net/tech/D-CaseEditor/

- DS-Bench/D-Cloud will be available from
  - http://www.dependable-os.net/tech/DSBenchDCloud/

- We thank Hajime Ueno for designing D-Case Editor with us. We also thank Shingo Takeda and Hideaki Koizumi for helping us with preparing the demonstration.