

Juliet Test Suite Overview

Tim Boland NIST

March 29, 2012

Basic Facts

- Juliet Test Suite contains 59493 test cases in the C/C++ and Java programming languages
- Each test case is small
- Most test cases in the test suite are synthetic (created for use)
- Each test case focuses on one type of flaw, but other incidental flaws may be present

Basic Facts (continued)

- In addition to the method(s)/function(s) containing the flaw of focus, each test case provides one or more non-flawed method(s)/function(s) that perform a similar task
- Test case is composed of one or more source code files, as well as auxiliary files providing test case support
- Test cases were developed/tested on the Windows platform but most work on other platforms

Other Aspects

- The test cases are based on the Common Weakness Enumeration (CWE)
- The C/C++ test cases contain examples for 116 different CWEs
- The Java test cases contain examples for 106 different CWEs
- Test cases are compilable/analyzable as a whole, by individual CWEs, or by individual test case

Other Aspects (continued)

- Test case naming format (from left to right): CWE ID, functional variant name, data/control flow indicator, programming language indicator
- If the test case is a Java servlet, the string “servlet” appears in the functional variant name
- Weakness classes (13 total) covered include: buffer handling, code quality, injection, information leaks, and pointer/reference handling
- “Complexities” covered include: loop structure, local control flow, data passing involving functions/methods/classes, data type, container

Test Case Example 1 (summarized)

File name:

```
CWE121_Stack_Based_Buffer_Overflow__char_t  
ype_overrun_memcpy_01.c
```

Initial stuff..

```
bad()
```

```
// bad code – memory overwrite of pointer
```

```
good()
```

```
//good code – no memory overwrite of pointer
```

```
main()
```

```
//used for building testcase on its own or for building binary
```

Test Case Example 2 (summarized)

File name:

CWE78_OS_Command_Injection__char_Environment_execl_01.c

initial stuff..

bad()

//bad code – non-empty environment variable used as input without validation

good()

//good code – benign input used

main()

//used for building testcase on its own or for building binary

Test Case Example 3 (summarized)

File name:

CWE114_Process_Control__basic_01.java

initial stuff..

public class..

bad()

//bad code – attempt to load library without full path

good()

//good code – specify a full path when loading library

main()

//used for building testcase on its own or for building binary

Final Items

- Possible use case: identify true positive, false positive, false negative
- Origin for Name “Juliet” – since these test suites are the tenth major group of tests added to the NIST SAMATE Reference Dataset (SRD), and “J” is the tenth letter of the alphabet, “Juliet” is the International Phonetic Alphabet word for “J”
- Complete downloads of the full Juliet Test Suites are available from the top of page:
<http://samate.nist.gov/SRD/testsuite.php>