CERT manifest files

The CERT manifest files (“cert_manifest.xml” and “cert_juliet.csv”) are authored by Lori Flynn, David Svoboda, and Andrew Kotov.

Download them (version 1) here.

These files can be used by static analysis tool developers to test their coverage of (some of the) CERT Secure Coding Rules for C, using many of 61,387 test cases in the Juliet test suite v1.2. The format of “cert_manifest.xml” is a slightly-modified version of the SARD manifest format (https://samate.nist.gov/SRD/resources/sard_schema.xsd), designed to enable users of the SARD manifest to easily also use this new CERT manifest.

To determine which test suite data could be used for CERT Secure Coding Rules, we used precise mappings (see here and here for precise mapping information) between CERT Secure Coding Rules and CWEs, combined with analysis of the Juliet Test Suite metadata, and occasionally examining the test suite files (e.g., to check if the type of a variable was a short or a long).

The file “cert_manifest.xml” is modeled after Juliet entries in the SARD manifest. It differs in two major ways:

- Our manifest is for CERT Secure Coding Rules (not CWEs)
- Our manifest includes “fixed” entries that indicate line ranges that do *not* have a violation of the CERT secure coding rule identified (i.e., these identify line ranges for which a static analysis alert for that CERT secure coding rule would be a false positive)

Generally, we tried not to modify the order of existing attribute fields, so users of the SARD manifest could most easily also use our manifest (in case their parsers rely on attribute order). Also, we used test suite attributes with values as in the original Juliet SARD manifest, so the particular version of Juliet files can be identified.

Additional details about how our manifest differs from the SARD manifest, with reasons:

1. We added the following new attributes for the testcase field (attribute and entry values provided in parentheses):
    a. **`alternate-taxonomy.`** (`alternate-taxonomy=”CERT-C-Standard"`) Purpose is to indicate which alternate code flaw taxonomy (eg. CERT rules, CWEs, MISRA rules, etc.) that information will be provided for, as opposed to the code flaw taxonomy that the test suite was originally designed to test.
    b. **`SubmissionDate-alternate-taxonomy`**. (`SubmissionDate-alternate-taxonomy=2018-09-28`) Purpose is to indicate the date of submission of this manifest to SARD, for potential publication on the NIST SARD test suite website. The similarly-named attribute `SubmissionDate` is specific to the testcase itself, and that was used for all manifest entries.
    c. **`alternate-taxonomy-author`**. (`alternate-taxonomy-author="Lori Flynn and David Svoboda and Andrew Kotov"`) Purpose is to identify authors of the new manifest entries. The similarly-named `author` attribute is specific to the testcase itself, and that was used for all manifest entries.

2. For the `False` verdicts, we did particular things for the following fields and attributes (in bold):
    a. We added a **`fixed`** field (same as in the original SARD manifest) that identifies where the identified CERT secure coding rule is *not* violated
        i. For the **`verdict`** attribute, we use the value False (`verdict=”False”`).
    b. For the file field, we added fields and values similar to those for the “mixed” tag (i.e., True verdict entries for Juliet test cases, in the original SARD manifest Juliet entries). Many of the files did not have entries in the original SARD Juliet manifest entries.
        i. **`numberOfFiles.`** (`numberOfFiles="1"`) The purpose of this field for file entries with `True` verdicts is to indicate how many files are in a testcase. As an initial estimate, in `False` verdicts, we assume this count is only the file identified, in each case a single file.
        ii. **`checksum.`** (`checksum =”<SHA1_HASH>”`) The purpose of this attribute is to uniquely identify the file. The other SARD file entries for checksum were derived using SHA1, so we derived a checksum value by running sha1sum.
        iii. **`size.`** (`size =”<SIZE>”`) The purpose of this attribute is to identify the number of bytes in the file. To get this number, we ran the following command in a bash shell: wc -c
    c. **`id`**, (`id="10000000"`) The purpose of this field is to uniquely identify the testcase ID. Initially, we start with the first ID at 10000000 (a number larger than any id in the current SARD manifest), then increase each by 1. These are placeholders, as SARD assigns their own testcase ids.
    d. We simply copied these attributes and values describing the test suite, for the **`testcase`** field: **`id`**`=”86”,` **`submissionDate`**`="2013-05-20",` **`status`**`="Candidate"`
    e. We added the following new attributes for the testcase field, the same as described above for the `True` (“`mixed`”) verdicts: **alternate-taxonomy**, **`SubmissionDate-alternate-taxonomy`**, and **`alternate-taxonomy-author`**.

The file “cert_juliet.csv” contains True and False information for Juliet test suite and CERT Secure Coding Rules, but in a sparser format than the .xml file.

It has entries of 2 types:

1. `<CERT_RULE>, True, <JULIET_FILEPATH>, <SINGLE_LINE>, <CWE>`
2. `<CERT_RULE>, False, <JULIET_FILEPATH>, <LINE_RANGE>, <CWE>`

The filepaths for type-2 (`False`) are different from the filepaths for type-1 (`True`), for the same files. The type-1 filepath is taken from the Juliet test suite's manifest XML file obtained from the SARD site (https://samate.nist.gov/SRD/testsuite.php) by downloading the SARD-type manifest, The type-2 filepath is taken from the filepath starting at the "`testcases`" directory from the source code in the Juliet test suite obtained the Juliet-standalone-test-suite way. To explain: the key to getting the different versions (Juliet-standalone-test-suite or SARD-type) of sourcecode and manifest is how you download (from the same webpage!) from here: https://samate.nist.gov/SRD/testsuite.php

- to download the SARD-type manifest and code, click on the icon for "manifest" in the "SARD Suites" section of the webpage, which is *below* the "Standalone Suites" section.
- to download the Juliet-standalone-test-suite type of manifest and code, get it in the "Standalone Suites" section at the top of the page