Lab: Abstract Interpretation

(Week 10)

Yulei Sui, Mohamad Barbar, Hanyuan Li, Angela He

School of Computer Science and Engineering

University of New South Wales, Australia

Update your Assignment-3 Python Starter Code

- If you are using Python for implementation, **please do the following two**:
  - Update `pysvf` via `pip install pysvf -U` or `pip install -U pysvf --force-reinstall`
  - **`git pull`** or **`docker pull`** to update `AEHelper.py` (We made updates in `AEHelper.py`)!

Lab-3 and Assignment-3 Marking

- Assignment-3 marking based on three levels of test cases
  - Our marking criteria based on the true positive and false positive reported by `AEReporter`.
  - `https://github.com/SVF-tools/Software-Security-Analysis/wiki/Assignment-3#31-marking-and-internal-levels-of-test-cases`.
  - Six basic features to be implemented to pass level 1 tests
  - Possibly more features and algorithms for levels 2 and 3 large tests
  - An open-ended assignment; you will be expected to create your own C tests!
- Marks will be released after teaching week (Week 11 for Lab-3 and Week 12 for Assignment-3)
- If you have questions, feel free to post on Discourse forums or email our course mailing list.

Assignment 3 Mini Examples

- https://github.com/SVF-tools/Software-Security-Analysis/wiki/AE-CPP-APIs
- https://github.com/SVF-tools/Software-Security-Analysis/wiki/AE-Python-APIs

Building Larger Programs into LLVM IR / SVF IR

Want to analyse real, not just toy, programs with multiple files?

Building Larger Programs with Multiple Files: GLLVM (Optional)

Wrapper for `clang`/`clang++` that keeps bitcode around.

Homepage

`https://github.com/SRI-CSL/gllvm`

Usage

- Compile as you would with `clang`, just use `gclang` instead.
- Retrieve the bitcode with `get-bc` on the produced binary.

Installation

```
go install github.com/SRI-CSL/gllvm/cmd/...@latest
```

Tools now available in `$GOPATH/bin`

Building Larger Programs: Example (dvtm)

1. Retrieve source.
   - `wget https://www.brain-dump.org/projects/dvtm/dvtm-0.15.tar.gz`
   - `tar -xvf dvtm-0.15.tar.gz`
   - `cd dvtm-0.15`
2. Optionally, add a null pointer dereference or buffer overflow.
3. Build with `gclang`.
   - `CC=$HOME/go/bin/gclang make`
4. Retrieve bitcode.
   - `get-bc dvtm`

Now you should have a `dvtm.bc` to analyse. If you'd like to read the LLVM, disassemble the bitcode file with `llvm-dis`.

Easy to Build Programs

Here are some suggestions on programs which may be easy to build with GLLVM. These do not necessarily provide comprehensive coverage.

Lua, AWK (various implementations; gawk, one true awk, etc.), mblaze, abduco, bash (various shells generally), qbe (compiler backend), cproc (C compiler), coreutils (various implementations; GNU, busybox, etc.), ninja, sqlite, mutt, dhcpcd, mruby, tmux, janet, nano (text editor), dpkg, vis (text editor).

See also software coming from or used by minimalist communities like Suckless or Oasis Linux.

Using SVFIR

Use `/bin/svfir` to generate ICFGs of your test programs!

- Map the nodes and var IDs with `printAbstractState`