PA193 - Secure coding principles and practices

LABS: Static analysis of source code

Petr Švenda svenda@fi.muni.cz

CRoCS

Centre for Research on
Cryptography and Security

Overview - lab

- Cppcheck/PREfast
  - run tool against vulnerable code (test suite)
  - http://mathind.csd.auth.gr/static_analysis_test_suite/
  - fix problems found
  - detect false positives / negatives
- Write your own simple detection rule
  - Cppcheck

MS Visual Studio and PREfast

1. Set project warning level to /W4
   - Run and compile zuno.cpp and bufferOverflowDemo.cpp
   - New project must be created
   - Fix all warnings for clean compilation in VS /W4
2. Run Code analysis on bufferOverflowDemo.cpp
   - minimum or all rules
   - Analyze→Run code analysis on …
   - You need have Project selected inside Project explorer (otherwise Run code analysis… option will not appear)

Cppcheck

- Download Cppcheck and unpack (or install)
  - run command line, cppcheck bufferOverflow.cpp
  - cppcheck --enable=all bufferOverflow.cpp
- Setup viewer for cppcheck (e.g., Notepad++ zip package)
  - http://sourceforge.net/projects/notepadpp-usb/
- Use regular expression .+, --debug, “pass[word]*=”
  - create XML file with file for automated analysis
- Run against qualitative_analysis.zip
  - inspect problems found

CPPCheck + OpenSSL

- Run against openssl-1.0.1j.tar.gz (Yesterday ☺)
  - https://www.openssl.org/source/openssl-1.0.1j.tar.gz
  - Not clean yet – developers are struggling a bit
- Run against OpenSSL0.9.1c (1998)
  - https://www.openssl.org/source/openssl-0.9.1c.tar.gz
  - what are the bugs?

Hearthbleed bug

- OpenSSL 1.0.1 through 1.0.1f
- Download https://www.openssl.org/source/openssl-1.0.1e.tar.gz
- Locate function dtls1_process_heartbeat(SSL *s)
  - Ssl\t1_lib.c
- Will your static analyzers find anything?
  - Don’t be sad, even Coverity didn’t before bug was exposed
  - http://security.coverity.com/blog/2014/Apr/on-detecting-heartbleed-with-static-analysis.html

Homework

- Nothing this week ☺