Is code in your project sane enough?

Red Hat
Kamil Dudka
February 6th, 2015

Abstract

This demo session will show how we can easily check the sanity of code in our project. There is a tool named csmock, which takes a source RPM (or upstream tarball) and produces a list of possible defects in its code. Besides plug-ins for C/C++ analyzers (Clang, Cppcheck, and GCC), csmock now comes with experimental plug-ins for static analysis of python and shell scripts. We will also discuss how we can efficiently process the results of these tools and how to integrate them into our workflow.

fedora

Static Analysis

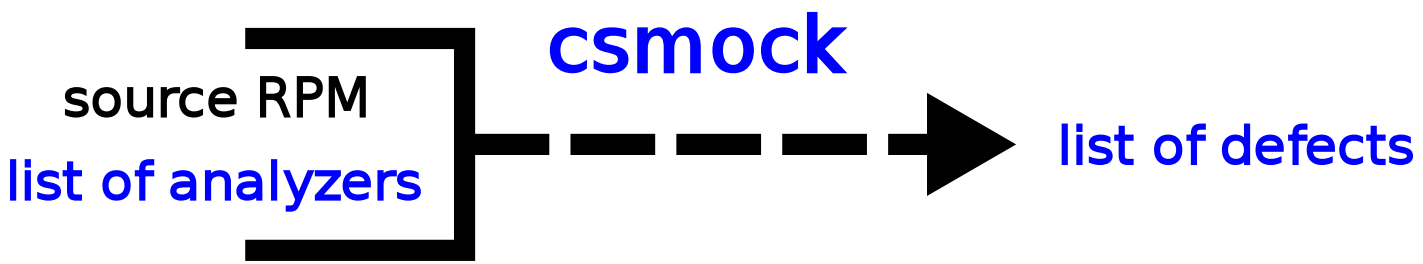
- Can automatically detect SW bugs early in the release cycle.
- Open source static analysis tools: Clang, Cppcheck, ...
- How to run these tools automatically?
- How to process their results?

fedora

Agenda

fedora

Static Analysis of RPM Packages (1/3)



fedora

Static Analysis of RPM Packages (2/3)

- Install csmock:

```
sudo yum install csmock
sudo gpasswd -a $USER mock
```

- Get a source RPM:

```
koji download-build --arch=src curl-7.40.0-1.fc22
```

- Analyze the package by Clang, Cppcheck, and GCC:

```
csmock -t clang,cppcheck,gcc ./curl-7.40.0-1.fc22.src.rpm
```

fedora

Static Analysis of RPM Packages (3/3)

- Additional info about csmock in the Flock 2014 presentation:
  https://kdudka.fedorapeople.org/static-analysis-flock2014.pdf

- Experimental support for Pylint and Shellcheck:

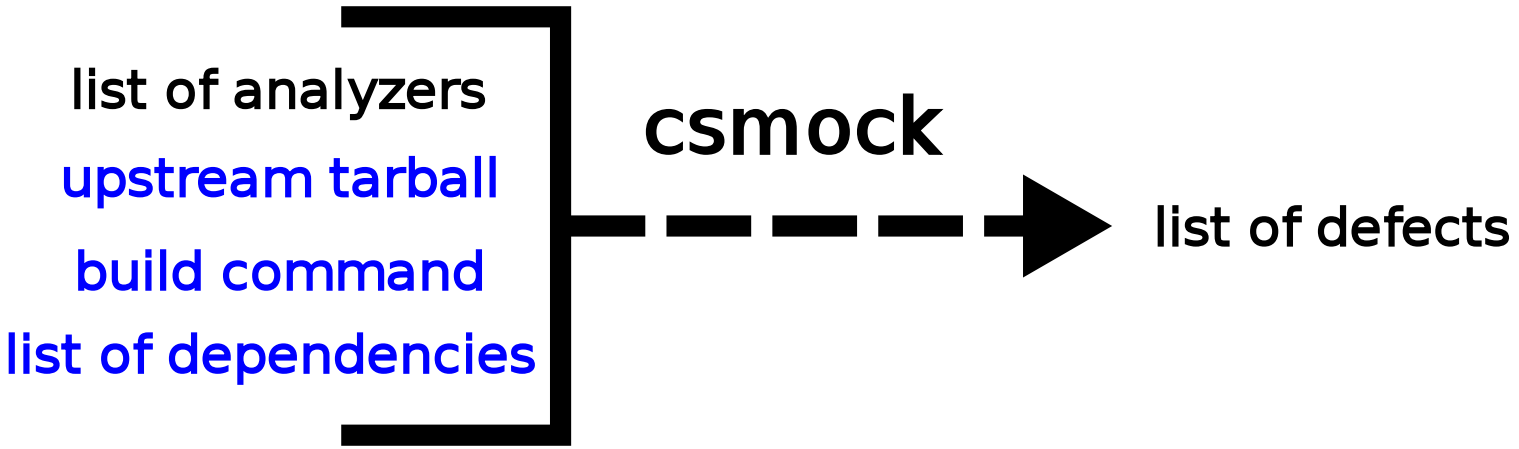
```
sudo yum install csmock-plugin-{pylint,shellcheck}
csmock -t pylint,shellcheck ...
```

- Example – analyze an RPM package by Pylint:

```
koji download-build --arch=src python-urlgrabber-3.10.1-6.fc22
csmock -t pylint ./python-urlgrabber-3.10.1-6.fc22.src.rpm
```

fedora

Static Analysis of Upstream Tarballs (1/2)



fedora

Static Analysis of Upstream Tarballs (2/2)

- Get an upstream tarball:

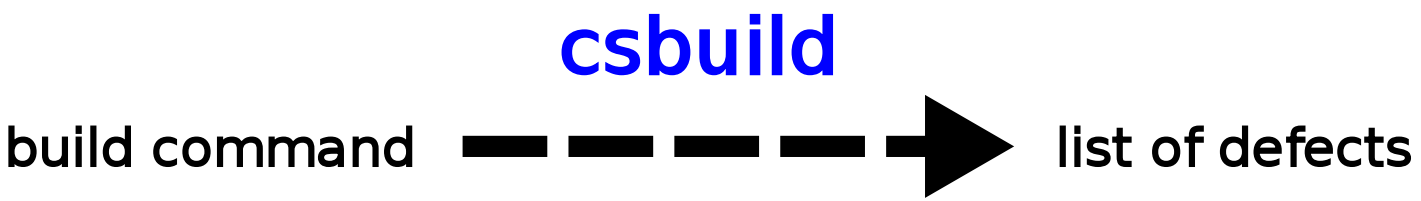
```
curl -O http://curl.haxx.se/download/curl-7.40.0.tar.lzma
```

- Analyze the upstream tarball by Clang, Cppcheck, and GCC:

```
csmock -t clang,cppcheck,gcc ./curl-7.40.0.tar.lzma \
       -c "./configure --with-libssh2 && make -j5"  \
       --install "libssh2-devel openssl-devel"
```

fedora

Static Analysis for Upstream Developers (1/3)



fedora

Static Analysis for Upstream Developers (2/3)

- Install the csbuild tool:

```
sudo yum install csbuild
```

- Get latest upstream source code and prepare it for build:

```
git clone --branch curl-7_40_0 --depth 8 \
    https://github.com/bagder/curl.git
cd curl
git checkout -b master
./buildconf
./configure
```

- Analyze the current source code:

```
csbuild -c "make -j5"
```

fedora

Static Analysis for Upstream Developers (3/3)

- Change something in the code:

```
sed "0,/e == NULL/s/e == NULL/e = NULL/" -i lib/llist.c
```

- Analyze the source files we have changed:

```
csbuild -c "make -j5"
```

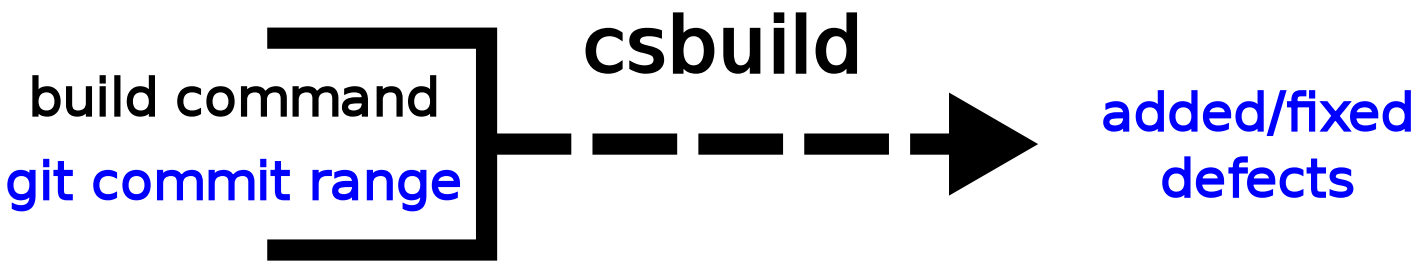
```
Error: CPPCHECK_WARNING:
lib/llist.c:113: error[assignBoolToPointer]: Boolean value assigned to pointer.

Error: COMPILER_WARNING:
lib/llist.c: scope_hint: In function "Curl_llist_remove"
lib/llist.c:113:8: warning: assignment makes pointer from integer without a cast
#   if(e = NULL || list->size == 0)
#        ^

Error: CLANG_WARNING:
lib/llist.c:117:18: warning: Access to field "next" results in a dereference
of a null pointer (loaded from variable "e")
...
```

fedora

Static Analysis and Git (1/3)



fedora

Static Analysis and Git (2/3)

- Commit our changes:

```
git commit -a -m "break something"
```

- Print defects instroduced between a pair of git revisions:

```
csbuild -g curl-7_40_0..master -c "make -j5"
```

- Print also defects that were fixed:

```
csbuild -g curl-7_40_0..master --print-fixed \
        -c "make clean && make -j5"
```

fedora

Static Analysis and Git (3/3)

- Find a commit introducing new defects:

```
csbuild -g curl-7_40_0..master --git-bisect \
        -c "make clean && make -j5"
```

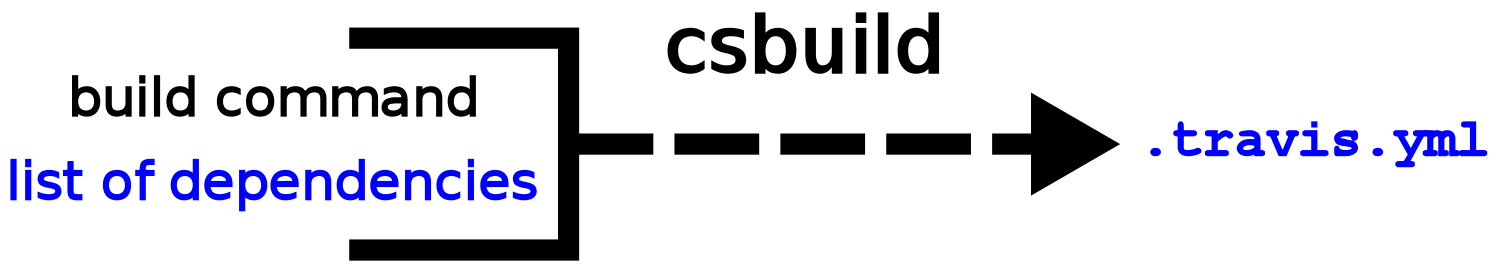
```
5793ba8bb8985eb69e57a7b771953729d3eaa094 is the first bad commit
commit 5793ba8bb8985eb69e57a7b771953729d3eaa094
Author: Kamil Dudka <kdudka@redhat.com>
Date:   Wed Feb 4 14:26:04 2015 +0100

    break something

:040000 040000 74ec8a1206bd4713e5885fa6c038bd0f60d60f35
73adf7a6515dbf1804af1a38ffb48cdf10b55633 M      lib
bisect run success
```

fedora

Static Analysis and Continuous Integration (1/2)



fedora

Static Analysis and Continuous Integration (2/2)

- Generate Travis CI metadata for differential static analysis:

```
csbuild -c "./buildconf && ./configure && make -j5" \
        --install libtool --git-bisect \
        --gen-travis-yml > .travis.yml
```

- Commit and push `.travis.yml`:

```
git add .travis.yml
git commit -m "notify me about newly introduced defects"
git push
```

- Enable Travis CI for my project on Github:

  https://travis-ci.org/profile

fedora

Agenda

1 Running Static Analyzers

2 Processing the Results of Static Analyzers

fedora

Comparing Lists of Defects

- Download a pair of defect lists for our experiments:

```
curl -O https://kdudka.fedorapeople.org/devconf15/warnings-old.txt
curl -O https://kdudka.fedorapeople.org/devconf15/warnings-new.txt
```

- Try to compare the lists of defects line by line:

```
diff warnings-{old,new}.txt
```

- Use csdiff to show defects added/fixed in the new version:

```
csdiff warnings-{old,new}.txt
csdiff warnings-{old,new}.txt --fixed
csdiff warnings-{old,new}.txt --ignore-path
```

fedora

Filtering Lists of Defects

- Try to filter a list of defects line by line:

```
grep ^/builddir/build/BUILD/ warnings-new.txt
grep ^/builddir/build/BUILD/ warnings-new.txt --invert-match
```

- Filter the lists of defects by path using csgrep:

```
csgrep --path ^/builddir/build/BUILD/ warnings-new.txt
csgrep --path ^/builddir/build/BUILD/ warnings-new.txt --invert-match
```

- Other options recognized by csgrep: `--event`, `--error`, `--msg`, `--remove-duplicates`, `--strip-path-prefix`

fedora

Sorting and Publishing Lists of Defects

- Try to sort a list of defects line by line:

```
sort warnings-new.txt
```

- Use cssort to sort the list of defects:

```
cssort warnings-new.txt
cssort warnings-new.txt --key checker
```

- Use cshtml to generate an HTML report:

```
cshtml warnings-new.txt > warnings-new.html
```

fedora

Reviewing Defects in Source Code

- csgrep defaults to an extended GCC format (also produced by Coverity).
- The format is recognized by many editors and IDEs (Integrated Development Environments).
- Example – review a list of defects using vim:

```
:cfile curl-7.40.0-1.fc22/scan-results.err
:copen
:map <F2> :cp^M
:map <F3> :cn^M
```

fedora

Future Plans

- Improve the support of python and shell.
- Support additional CI services – Taskotron?
- Support additional languages – Python, Java, ...

fedora

Conclusion

- There are open source static analyzers that are able to automatically detect bugs early in the release cycle.
- csmock makes it easy to run them on a given source RPM.
- csbuild makes static analysis easier for upstream developers.
- Tools like csdiff and csgrep are now available in Fedora.

- You can provide feedback at: `http://devconf.cz/f/131`