

Finding Bugs in Open Source Software using Coccinelle

Sune Rievers
sunerivers@stud.ku.dk

Datalogisk Institut - Københavns Universitet

August 3rd 2009

Overview

- 1 Presentation of OpenSSL
 - Usage of OpenSSL
 - Facts about OpenSSL
 - Reasons for choosing OpenSSL
- 2 Presentation of Coccinelle
 - The Coccinelle Engine
 - Sample Semantic Patch
- 3 Related work
- 4 Bugs found
 - Using pre-made patches
 - Using custom patches
- 5 Conclusion
 - Status
 - Perspectives

Example of Open Source Software: OpenSSL

Presentation of OpenSSL

OpenSSL - Library/Toolkit

Used as a cryptography layer in lots of software, including:

- Apache Web server (SSL)
- Bind (DNSSEC)
- cURL (SSL)
- stunnel (SSL)
- Samba
- Qmail (TLS)
- Postfix (TLS)

OpenSSL is cross-platform.

Facts about OpenSSL

- Open Source Software
- Project started in 1999
- Some documentation available, but sparse and technical
- Codebase contains almost 5.000 files, 550.000 LOC

Reasons for choosing OpenSSL

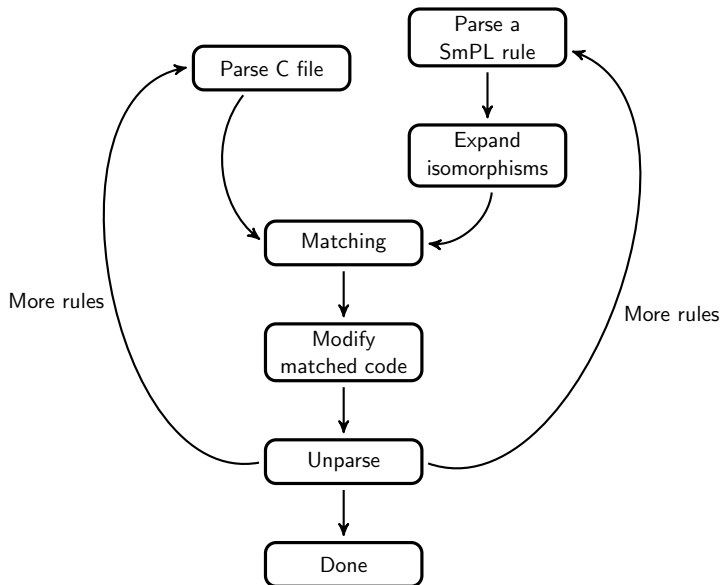
- Open Source
- Public bug tracker
- Active development mailing list
- Widely used in software
- Security critical, i.e. important

Coccinelle: A matching and transformation tool

Presentation of Coccinelle

- Collateral evolutions in source code
- Uses Semantic Patch Language (SmPL)
 - ▶ Declarative language
 - ▶ Standard patch-like syntax
 - ▶ Metavariables
 - ▶ Modifiers (add/remove code)
 - ▶ File and line independent
 - ▶ Isomorphisms
 - ▶ Scriptable with other languages, eg Python
- Transforms C code
- Originally used for Linux Kernel drivers
- Usable for bug finding

The Coccinelle Engine



Sample Semantic Patch - problem

Use after free

Pointer type variable is freed, and then subsequently dereferenced in a possible code path.

Sample Semantic Patch - problem

Use after free

Pointer type variable is freed, and then subsequently dereferenced in a possible code path.

Sample C code

```
int main(int argc, const char *argv[]) {
    char *buff;

    buff = (char *)OPENSSL_malloc(BUFFERSIZE);
    if (!buff) {
        /* Handle error condition */
    }
    /* ... */
    OPENSSL_free(buff);
    /* ... */
    strncpy(buff, argv[1], BUFFERSIZE-1);
}
```

Sample Semantic Patch - solution

Use after free SP

```
@@  
expression E;  
identifier f;  
expression E2 != NULL;  
@@
```

```
- OPENSSL_free(E);
```

```
...
```

```
(  
OPENSSL_free(E);  
|  
E2 = E;  
+ OPENSSL_free(E);  
+ E = NULL;  
|  
f(<+...E...+>);  
+ OPENSSL_free(E);  
+ E = NULL;  
)
```

Product	Languages	Licence
Coverity Prevent	C, C++, Java, C#	Commercial
Polyspace Verifier	C, C++, Ada	Commercial
Klocwork K7	C, C++, Java	Commercial
PR-Miner	C	Unknown
CP-Miner	C, C++	Unknown

Pre-made patches

Pre-made patches

Semantic patch	Description
andand	Use of && instead of or vice versa
badzero	Use of literal zero instead of NULL with pointers
continue	Superfluous continue at bottom of a loop
find_unsigned	Comparison of negative values to unsigned integers
isnull	Dereference of a value known to be NULL
mini_null_ref mini_null_ref2 null_ref	Various cases of pointer dereferencing with a following NULL check
notand	Use of bitwise and (&) on a boolean and a numeric value
notnull	NULL check on already tested values

Bugs found - examples

Example of badzero bug

```
- if ((m == 0) || (r == 0) || (f == 0))  
+ if ((m == NULL) || (r == NULL) || (f == NULL))  
  return 0;
```

Bugs found - examples

Example of badzero bug

```
- if ((m == 0) || (r == 0) || (f == 0))  
+ if ((m == NULL) || (r == NULL) || (f == NULL))  
    return 0;
```

Example of find_unsigned bug

```
unsigned int ret;  
if (ret < 0)  
{  
    IBMCAerr(IBMCA_F_IBMCA_RAND_BYTES,  
            IBMCA_R_REQUEST_FAILED);  
    goto err;  
}
```

Bugs found - examples

Example of notnull bug

```
if (!tree)
    return 0;
```

...

```
tree->auth_policies = NULL;
tree->user_policies = NULL;
```

```
- if (!tree)
- {
-     OPENSSL_free(tree);
-     return 0;
- }
```

False positives found - example

Example of null_ref false positive

```
#ifdef KSSL_DEBUG
{
    ...
    krb5ticket->enc_part2->times.starttime,
    krb5ticket->enc_part2->times.authtime,
    krb5ticket->enc_part2->times.endtime);
}
#endif /* KSSL_DEBUG */

krb5rc = KRB5_NO_TKT_SUPPLIED;
if (!krb5ticket || !krb5ticket->enc_part2 || ...
```

Bugs found using pre-made patches

Results from running pre-made patches on the 0.9.8j version of OpenSSL

Semantic patch	Matches	False positives
andand	0	0
badzero	5	0
find_unsigned	0	0
isnull	0	0
mini_null_ref	0	0
mini_null_ref2	0	0
notand	0	0
notnull	11	0
null_ref	0	0

Figure: Total: 16 bugs

Bugs found using pre-made patches

Results from running pre-made patches on the cvs (rev. 17904) version of OpenSSL

Semantic patch	Matches	False positives
andand	0	0
badzero	6	0
continue	2	0
find_unsigned	1	0
isnull	1	0
mini_null_ref	0	0
mini_null_ref2	0	0
notand	0	0
notnull	10	0
null_ref	5	1

Figure: Total: 24 bugs

Custom patches

Custom patches

Semantic patch	Description
malloc_style	Detects incorrect usage of memory allocation functions in OpenSSL
openssl_malloc	An allocated variable is never freed (using OpenSSL functions)
use_after_free	A variable is dereferenced after it has been freed

Malloc_style SP

```
@@
```

```
expression E;
```

```
@@
```

```
- free(E);
```

```
+ OPENSSL_free(E);
```

```
@@
```

```
expression E;
```

```
@@
```

```
- malloc(E);
```

```
+ OPENSSL_malloc(E);
```

Custom patches

Use_after_free SP

```
@@  
expression E;  
expression E2 != NULL;  
identifier f;  
@@
```

```
- free(E);
```

```
...
```

```
(  
free(E);  
|  
E2 = E;  
+ free(E);  
+ E = NULL;  
|  
f(<+...E...+>);  
+ free(E);  
+ E = NULL;  
)
```

Custom patches

Use_after_free SP (continued)

```
@@  
expression E;  
expression E2 != NULL;  
identifier f;  
@@
```

```
- OPENSSL_free(E);
```

```
...
```

```
(  
OPENSSL_free(E);  
|  
E2 = E;  
+ OPENSSL_free(E);  
+ E = NULL;  
|  
f(<+...E...+>);  
+ OPENSSL_free(E);  
+ E = NULL;  
)
```

Bugs found using custom patches

Example of openssl_malloc bug

```
enctmp = OPENSSL_malloc(keylen + 8);
if (!enctmp)
{
    PEMerr(PEM_F_DO_PVK_BODY, ERR_R_MALLOC_FAILURE);
    return NULL;
}
if (!derive_pvk_key(keybuf, p, saltlen,
- (unsigned char *)psbuf, inlen))
-     return NULL;
+ (unsigned char *)psbuf, inlen)) {
+     OPENSSL_free(enctmp);
+     return NULL;
+ }
```

Bugs found using custom patches

Results from running custom patches on the beta2 version of OpenSSL

Semantic patch	Matches	False positives
malloc_style	19	?
openssl_malloc	1	0
use_after_free	2	1

Figure: Total: 2-21 bugs

- Coccinelle is perfectly applicable of finding (and fixing) bugs
- Valgrind, Purify and Coverity already applied to OpenSSL
- Ongoing effort to stabilize OSS in general
- None of my patches have yet been applied to OpenSSL
- Many patches deemed redundant, and as unnecessary optimizations

Conclusion - Perspectives

- Create more Semantic Patches to catch more bugs
- Extend Coccinelle to work on other languages
 - ▶ e.g. C#, C++, Java
- Fix minor usability issues