Finding Bugs in Open Source Software using Coccinelle

Sune Rievers - sunerievers@stud.ku.dk

April 21, 2009
Presentation of OpenSSL
  Usage of OpenSSL
  Facts about OpenSSL
  Organization
  Third party software

Bugs found
  Using pre-made scripts
  Using custom scripts

Reception from the community

Status
OpenSSL - Library/Toolkit
Used as a cryptography layer in lots of software, including: \(^1\)

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

OpenSSL is cross-platform.

\(^1\)Source: http://openssl.org/related/apps.html
OSS

- Project started in 1999
- Managed by a worldwide community of volunteers
- Apache-style licence - free use for commercial and non-commercial purposes
- Not compatible with GPL due to dual-license
- Currently at version 0.9.8k (stable) and 1.0.0 beta1 (unstable)
- Some documentation available, but sparse and technical
- Codebase contains almost 5,000 files, 550,000 LOC
Team consists of 11 developers
4 developers on the core team and manages the rest
Active dev mailing list
  Big focus on security and critical issues
User support through mailinglist
Users are mostly implementors of OpenSSL
OpenSSL is based on SSLeay library, still uses Libdes for DES etc.

Patent issues due to misc. cryptography algorithms used

- RC4 is a trademark of RSA Security
- IDEA algorithm is patented by Ascom
- RC5 algorithm by RSA Security
- MDC2 algorithm is patented by IBM
- Camellia algorithm by NTT and Mitsubishi
<table>
<thead>
<tr>
<th>Cocci script</th>
<th>Bugs found</th>
<th>False positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>andand</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>badzero</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>find_unsigned</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>isnull</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mini_null_ref</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mini_null2_ref</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>notand</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>notnull</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>null_ref</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Figure:** Results from running pre-made scripts on the 0.9.8j version of OpenSSL
<table>
<thead>
<tr>
<th>Cocci script</th>
<th>Bugs found</th>
<th>False positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>andand</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>badzero</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>continue</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>find_unsigned</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>isnull</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>mini_null_ref</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mini_null2_ref</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>notand</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>notnull</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>null_ref</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

**Figure:** Results from running pre-made scripts on the cvs (rev. 17904) version of OpenSSL
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;
}
Example of notnull bug

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

...

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

- if (!tree)
- {
-   OPENSSL_free(tree);
-   return 0;
- }
```
Example of null_ref false positive

```c
#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  ||  ...
<table>
<thead>
<tr>
<th>Cocci script</th>
<th>Bugs found</th>
<th>False positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malloc_style</td>
<td>26</td>
<td>?</td>
</tr>
<tr>
<td>OpenSSL_malloc</td>
<td>7</td>
<td>?</td>
</tr>
<tr>
<td>Use_after_free</td>
<td>1</td>
<td>?</td>
</tr>
<tr>
<td>OpenSSL_malloc_free</td>
<td>27</td>
<td>?</td>
</tr>
<tr>
<td>Return_codes</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

**Figure**: Results from running custom scripts on the beta1 version of OpenSSL
Malloc_style script

@@
expression E;
@@
- free(E);
+ OPENSSL_free(E);

@@
expression E;
@@
- malloc(E);
+ OPENSSL_malloc(E);
Use_after_free script

@@
expression E;
expression E2 != NULL;
expression f;
@@
Use_after_free script (continued)

- free(E);
...
(E2 = E;
+ free(E);
| f(<+...E...+>);
+ free(E);
)
+ E = NULL;
Use_after_free script (continued)

- OPENSSL_free(E);

...  
(  
E2 = E;
+ OPENSSL_free(E);
|  
f(<+...E...+>);
+ OPENSSL_free(E);
)  
+ E = NULL;
Example of openssl_malloc bug

```c
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;
+  }
```
Not overly positive

Many patches deemed redundant, and as unnecessary optimizations

Issue tracker has a lot of stale reports (some 5+ years old)
Valgrind, Purify and Coverity already applied
Ongoing effort to stabilize OSS in general
None of my patches have yet been applied to OpenSSL
Questions?