SATE IV CVE-selected
Procedure and Observations

Vadim Okun, NIST
vadim.okun@nist.gov
March 29, 2012

The SAMATE Project
http://samate.nist.gov/
Analysis procedure for CVE-selected test cases

Selection Methods:

- random subset
- related to CVEs
- to human findings

Tool warnings ~52K

Analyze for correctness and associate

Selected warnings

Analyze the data
Outline

• Procedure for random subset analysis
• Observations from analysis
• Suggestions for tool improvement
Procedure for Subset Analysis

• A selected set of warnings were analyzed by experienced programmers
  – This year it was Aurelien, Vadim, and Paul
Step 1 – select a warning

<table>
<thead>
<tr>
<th>Test case</th>
<th>Unique ID</th>
<th>Tool name</th>
<th>Name</th>
<th>CWE ID</th>
<th>Severity</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>wireshark-vln</td>
<td>9693</td>
<td>cppcheck</td>
<td>nullPointer</td>
<td>476</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>16398</td>
<td>GrammarTech CodeSonar</td>
<td>Buffer Overrun</td>
<td>120</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>237455</td>
<td>Goanna</td>
<td>SPC-uninit-arr-all</td>
<td>457</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>74542</td>
<td>INFER</td>
<td>ARRAY_OUT_OF_BOUNDS_L1</td>
<td>119</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>77377</td>
<td>INFER</td>
<td>NULL_DEREFERENCE</td>
<td>476</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>9656</td>
<td>cppcheck</td>
<td>resource Leak</td>
<td>772</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>235518</td>
<td>Goanna</td>
<td>ARR-inv-index-pos</td>
<td>120</td>
<td>1</td>
<td>0.8</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>77244</td>
<td>INFER</td>
<td>NULL_DEREFERENCE</td>
<td>476</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>75127</td>
<td>INFER</td>
<td>NULL_DEREFERENCE</td>
<td>476</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>78807</td>
<td>INFER</td>
<td>ARRAY_OUT_OF_BOUNDS_L1</td>
<td>119</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>235975</td>
<td>Goanna</td>
<td>PTR-null-assign-fun-pos</td>
<td>476</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>77642</td>
<td>INFER</td>
<td>DIVIDE_BY_ZERO</td>
<td>369</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>235640</td>
<td>Goanna</td>
<td>ATH-div-0-assign</td>
<td>369</td>
<td>1</td>
<td>0.8</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>9669</td>
<td>cppcheck</td>
<td>memleak</td>
<td>401</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>9670</td>
<td>cppcheck</td>
<td>nullPointer</td>
<td>476</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>235604</td>
<td>Goanna</td>
<td>ARR-inv-index-ptr-pos</td>
<td>120</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>235437</td>
<td>Goanna</td>
<td>ARR-Inv-Index-ptr</td>
<td>119</td>
<td>1</td>
<td>0.8</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>235781</td>
<td>Goanna</td>
<td>MEM-stack-global</td>
<td>825</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>235874</td>
<td>Goanna</td>
<td>PTR-null-assign-pos</td>
<td>476</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>237486</td>
<td>Goanna</td>
<td>SPC-uninit-var-some</td>
<td>457</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>77643</td>
<td>INFER</td>
<td>DIVIDE_BY_ZERO</td>
<td>369</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>16763</td>
<td>GrammarTech CodeSonar</td>
<td>File System Race Condition</td>
<td>367</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>71500</td>
<td>INFER</td>
<td>DANGLING_POINTER_DEREFERENCE</td>
<td>465</td>
<td>1</td>
<td>Empty</td>
</tr>
<tr>
<td>wireshark-vln</td>
<td>77226</td>
<td>INFER</td>
<td>DANGLING_POINTER_DEREFERENCE</td>
<td>465</td>
<td>1</td>
<td>Empty</td>
</tr>
</tbody>
</table>
Step 2 – understand the warning

• What does it say about the code?

<table>
<thead>
<tr>
<th>Test case</th>
<th>wireshark-vin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool Information</td>
<td>Goanna. Version: 2.0 (lizard)</td>
</tr>
<tr>
<td>Unique ID</td>
<td>235518</td>
</tr>
<tr>
<td>Tool-specific ID</td>
<td>120</td>
</tr>
<tr>
<td>Weakness name</td>
<td>ARR-Inv-index-pos</td>
</tr>
<tr>
<td>CWE ID</td>
<td>120 (Buffer Copy without Checking Size of Input ('Classic Buffer Overflow'))</td>
</tr>
<tr>
<td>Severity / Probability / Tool Specific Rank</td>
<td>1 / 0.8 / 1</td>
</tr>
</tbody>
</table>

Associated weaknesses

- Current Associations: None
- Suggested Associations: None
- Add an association

Vulnerability paths

- `Browse this path`:

  - `highlight :: doxygen`:
    - `wireshark-1.2.0/epan/dissectors/packet-tnpcp.c (524) doxygen | highlight | explanation`:
      - `function or method: fill Enums Id Vals`:

  - Look for weaknesses in the last line of the path in a range of `Same line`:
  - lines around the given line number.
  - Don't restrict to the same CWE ID

Raw outputs

- Text output: Array `tnpcp Enums Id Vals` 2nd subscript interval [0, 500] may be out of bounds [0, 499]
Step 3 – understand the code

- Does this happen? Could it cause problems?

- Doxygen provides call graphs and hyperlinks to functions and definitions.
Step 3 – understand the code

- Original tool output has a lot of information and splices code to show control flow.
Step 4 – write an evaluation

• Include code snippets and reasoning so others can critique it

Evaluation #704 (link) made for the weakness 235518

Correctness: false

Pertinent code is
489 gint i = 0, ....
502 while (fgets(line_in_file, MAX_TPNCP_DB_ENTRY_LEN, file) != NULL) {
    ....
512      enum_val++; i = 0;
524    tpncp Enums IdVals[enum_val][i].value = enum_id;
525    if (i < MAX ENUM ENTRIES) {
526         i++;
527    }
528    else {
529         break;
530     }
531     }
532 }

where MAX ENUM ENTRIES is 500. The warning is
Array 'tpncp Enums IdVals' 2nd subscript interval [0,500] may be out of bounds [0,499]
The 2nd subscript interval is really [0,499].

Evaluation by PAUL :: 2012-03-02

Evaluation #705 (link) made for the weakness 235518

Correctness: security

I erred in the previous evaluation. The subscript interval IS [0,500], so there could be a problem. If i=499 at line 525, the test is true, and i is incremented (to 500)

Evaluation by PAUL :: 2012-03-02
Decision process

- Context
- Path
- Type
- Bug
- ...

- Security
- Quality
- Unknown
- Insignificant
- False
Step 4b – alert developers

- If there is clearly an error
  - and it is easily fixed or high impact
  - and it exists in the current version,

- tell the developers
Step 5 – associate other warnings

```c
protocol_name_len = (unsigned int) strlen(protocol_name);  // 181383

/* Walk protocols list */
for (i = proto_get_first_protocol(&cookie); i != -1; i = proto_get_next_protocol(i)) {
    protocol = find_protocol_by_id(i);

    if (!proto_is_protocol_enabled(protocol))  // 77377 235908 236035
        continue;

    if (protocols_only) {
        const gchar *name = proto_get_protocol_filter_name(i);

        if (!g_ascii_strcasecmp(protocol_name, name, protocol_name_len)) {
            add_to_autocompletion_list(treeview, name);
            if (strlen(name) == protocol_name_len) {  // 181384
                exact_match = TRUE;
            }
            count++;
            if (count == 1)
                first = name;
        }
    } else {
```
### Overlap for true quality/security

<table>
<thead>
<tr>
<th>Software</th>
<th>Count (1 tool)</th>
<th>Count (2 tools)</th>
<th>Count (3 or 4 tools)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dovecot (6 tools)</td>
<td>19</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Wireshark (5 tools)</td>
<td>32</td>
<td>13</td>
<td>3</td>
</tr>
<tr>
<td>Tomcat (2 tools)</td>
<td>14</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Jetty (2 tools)</td>
<td>12</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

More overlap for some weakness categories
CVEs

• Real-life vulnerabilities
• 88 CVEs in the 4 test cases
  – Identify source, sink or path locations
  – Match to tool warnings
Top 5 CWEs for CVEs

- Top CWEs cover 43 of 88 CVEs
- A total of 30 different CWE ids
- Many design flaws
## Related warnings from tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Directly Related</th>
<th>Indirectly Related</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dovecot (8)</td>
<td>1</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Wireshark (43)</td>
<td>5</td>
<td>1</td>
<td>37</td>
</tr>
<tr>
<td>Tomcat (32)</td>
<td>6</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>Jetty (5)</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

- CVEs described better than in SATE 2010
Related Warnings for Top 5 CWEs

<table>
<thead>
<tr>
<th>CWE</th>
<th>Directly related</th>
<th>Indirectly related</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Null deref (5)</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Buffer (12)</td>
<td>2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>XSS (10)</td>
<td>7</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Info leak (8)</td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Path trav (8)</td>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

- Related warnings from tools for 8 CWEs
CVE-2006-7195 Not Found

- JSP Standard Tag Library (JSTL)
  
  ```<td>${header["host"]}</td>`

- Should understand popular libraries and frameworks
On discrimination

- Reporting a weakness when there is one
- Keeping quiet when there is none
- Varies a lot by tool and weakness category
CVE-2009-3550 Found

Vulnerable version:

1314 item = item -> parent;

1318 item = item -> parent;

Tool warning: pointer item last assigned on line 1314 could be NULL and is dereferenced at line 1318
CVE-2009-3550 Found

Fixed version:
```
#define GET_ITEM_PARENT(x) ((x->parent!=NULL)?x->parent:x)

item = GET_ITEM_PARENT(item);

item = GET_ITEM_PARENT(item);
```

No tool warning here. Perfect!
Vulnerable version:

```java
String role = request.getParameter("role");
...
< %= role %>
```

Reported
CVE-2006-7196 / 2009-0781
Not discriminated

Fixed version:
String role = request.getParameter("role");
...
<%={% filter(role) %}>

Reported anyway

• Plenty of much more complex cases
Human Analysis

- Wireshark dissectors are protocol decoders
- Chose Intelligent Platform Management Interface (IPMI) dissector for analysis
  - Fuzzing
  - Manual source code review
Human Analysis Results

- Buffer overrun in vulnerable version
- Corrected in fixed version
- Corresponds to CVE-2009-2559
CVE-2009-2559 Not Found

static const int *tsel[] = { &ett_ipmi_se_XX_b1, 
&ett_ipmi_se_XX_b2, &ett_ipmi_se_XX_b3, &ett_ipmi_se_XX_b4 }; 

for (i = 0; offs < len; i++, offs++) {
    s_tree = proto_item_add_subtree(ti, *tsel[i]);

    \textit{i is not checked and goes out of bounds}

- Tools routinely find such weaknesses. Why not here?
- Did tools find/analyze the code?
Summary

• Find and analyze more code
• Better discrimination
• Better understand libraries and frameworks
• Participate in future SATEs 😊