TINTORERA

Attack Surface Intelligence of Source Code
ME & VULNEX

Simon Roses Fenerling

- Founder & CEO, VULNEX  www.vulnex.com
- @simonroses
- Former Microsoft, PwC, @Stake
- Black Hat, RSA, OWASP, SOURCE, AppSec, DeepSec, TECHNET

VULNEX

- CyberSecurity Startup
- @vulnexsl
- Services & Training
- Products: BinSecSweeper (Binary Analysis)
TALK OBJECTIVES

• GCC & Python, hand to hand

• Transformations: source code to useful data

• Practical code understanding
AGENDA

1. The need of Attack Surface Intelligence of Source Code
2. GCC Overview
3. GCC-Python-Plugin
4. Source Code Intelligence
5. Tintorera Overview
6. Tintorera Analysis Demos
7. Conclusions
8. Q&A
1. The need of Attack Surface Intelligence of Source Code
1. CODE IS GETTING COMPLEX!

<table>
<thead>
<tr>
<th>Software</th>
<th>SLOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firefox</td>
<td>14 Million</td>
</tr>
<tr>
<td>Windows Server 2003</td>
<td>50 Million</td>
</tr>
<tr>
<td>Debian 7.0</td>
<td>419 Million</td>
</tr>
<tr>
<td>Mac OS X 10.4</td>
<td>86 Million</td>
</tr>
<tr>
<td>Linux Kernel 2.6.25</td>
<td>13.5 Million</td>
</tr>
<tr>
<td>Linux Kernel 3.6</td>
<td>15.9 Million</td>
</tr>
</tbody>
</table>
1. DOCUMENTATION

MISSING
IT Documentation

Our server crashed and we’re not super jazzed about it. We don’t know where our backups are, but are pretty sure they exist. Could really use some IT documentation right about now.

CAN YOU HELP?
Please call 778-555-6666
1. TYPICAL CODE REVIEW
1. WHERE TO START?

- File operations
- Networking
- Processes
- Crypto
- Authentication
- ??
1. TOOLS?

grep

rough-auditing-tool-for-security
Rough Auditing Tool for Security (RATS)

eclipse
2. GCC Overview
2. GCC

- Compiler system that supports various programming languages
- Popular UNIX variants
- Supports all major languages: C, C++, Java, Objective-C, etc.
- PLUGINS!!
- FREE
2. GCC INTERNALS

2. GCC TERMINOLOGY

• GENERIC is common representation shared by all front ends
  – Each parser must emit GENERIC

• GIMPLE is a simplified version of GENERIC
  – 3 address representation
  – Simplified control flow

• RTL (Register Transfer Language), assembler for an abstract machine
2. GCC PASSES

The lowering passes

*warm_unused_result
*diagnose_omp_blocks
mudflap1
omp_lower
lower
ehopt
eh
cfg
*warm_function_return
*build_cgraph_edges

The "small IPA" passes

*free_lang_data
visibility
early_local_cleanups
*free_cfg_annotations
*init_datastructures
ompexpp
*referenced_vars
ssa
veclower
*early_warn_uninitialized
*rebuild_cgraph_edges
inline_param
einline
early_optimizations
*remove_cgraph_callee_edges
copyrename
cpp
forwprop
ealias
esra
copyprop
mergephi
cdce
eiga_sra
tailr
switchconv
ehcleanups
profile
local-pure-const
fnsplit
release_ssa
*rebuild_cgraph_edges
inline_param
tree_profile_ipa
feedback_fnsplit

3. GCC-Python-Plugin
3. GCC-PYTHON-PLUGIN

- GCC plugin that embeds Python in GCC 😊

- Now your Python script can access GCC passes and perform analysis

- Developed by David Malcolm (Fedora)

3. GCC-PYTHON-PLUGIN EXAMPLE

# Copyright 2011 David Malcolm <dmalcolm@redhat.com>
# Copyright 2011 Red Hat, Inc.
#
# This is free software: you can redistribute it and/or modify it
# under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful, but
# WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
# General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see
# <http://www.gnu.org/licenses/>.

# Sample python script, to be run by our gcc plugin
# Show all the passes that get executed
import gcc

def my_pass_execution_callback(*args, **kwargs):
    (optpass, fun) = args
    print(args)

gcc.register_callback(gcc.PLUGIN_PASS_EXECUTION,
                      my_pass_execution_callback)
3. GCC-PYTHON-PLUGIN DEMO
3. GCC-PYTHON-PLUGIN IDEAS

• Write scripts for:
  – malloc/free usage
  – Array boundary checks
  – Code visualizations
  – You name it!
4. Source Code Intelligence
4. CODE UNDERSTATING

- What API are being used?
- Number of functions?
- Inputs / Outputs of functions?
- Function relationship
- What comments said?
- Code complexity
4. CODE METRICS

• Controversial topic but needed

• Metrics:
  – Function complexity (Cyclomatic)
  – Number of:
    • Lines
    • Code
    • Blanks
    • Comments
  – Line Length
  – Number: Bugs per Line
  – You name it....
4. CODE COMPLEXITY

• Counts the number of linearly independent paths through the source code

• Basically we can have an idea of the complexity of functions

• Complexity is security enemy!

• Created by Thomas McCabe
## 4. CODE COMPLEXITY THRESHOLD

<table>
<thead>
<tr>
<th>Cyclomatic Complexity</th>
<th>Risk Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-10</td>
<td>a simple program, without much risk</td>
</tr>
<tr>
<td>11-20</td>
<td>more complex, moderate risk</td>
</tr>
<tr>
<td>21-50</td>
<td>complex, high risk program</td>
</tr>
<tr>
<td>greater than 50</td>
<td>untestable program (very high risk)</td>
</tr>
</tbody>
</table>

http://www.sei.cmu.edu/reports/97hb001.pdf
4. SOURCE CODE ANALYSIS FLOWGRAPH NOTATION

If .. then

If .. then .. else

If .. and .. then

If .. or .. then

Do .. While

While .. Do

Switch

www.mccabe.com/ppt/SoftwareQualityMetricsToIdentifyRisk.ppt
4. SOURCE CODE VISUALS TOO

BINARIES

SOURCE CODE
5. Tintorera Overview
5. TINTORERA – BLUE SHARK

• “Put source code into context”

• Objective: Get a feeling of the code while compiling!!

• Intelligence of source code:
  – Code visualizations
  – Comments analysis
  – API identification
  – Metrics
  – HTML Reports

• C code transformed to JSON files, now you can query and perform analysis on data
5. TINTORERA INTERNALS

• Two files:
  – analyzer.py: To be used while compiling a project
  – do_report_tintorera.py: Use after project has been compiled to generate report

• Composed of:
  – Python code
  – JSON data files
  – HTML / CSS / Javascript
5. TINTORERA STRUCTURE

- Python files
- Folders:
  - data/ : API JSON file
  - templates/ : HTML templates
  - js/ : Javascript code
  - images/
  - Tintorera_lib/ : python code
5. TINTORERA INSTALL & USAGE

1. GCC version 4.7 or later

2. Install gcc-python-plugin (See web doc)

3. Set path:
   1. Export LD_LIBRARY_PATH=/gcc-python-plugin/gcc-c-api

4. Add line to Makefile (CC= tag)
   1. gcc -fplugin=/gcc-python-plugin/python.so -fplugin-arg-python-script=/tintorera/analyzer.py

5. Run make

6. After compile use:
   1. Python do_report_tintorera.py -c tinan.cfg
5. TINTORERA CONFIG FILE

• Edit tinan.cfg to suit your needs

• Set parameters such as:
  – Folder to save analysis report
  – Enable / disable analysis
    • Basic blocks
    • Callgraphs
    • Comments
    • Gimples
    • Etc.
  – Cyclomatic Thresholds
5. TINTORERA DATA FILES

- Folder: /data
- File: tinto_api.json
- JSON file to define APIs
5. CODE TRANSFORMATION

SOURCE CODE → JSON FILES

SOURCE CODE → HTML REPORT
5. TRANSFORMED JSON FILES

• 3 files:

1. tintorera_bb_file.json: code basic blocks
2. tintorera_meta_info.json: general information, file size and code & comments not inside functions
3. tintorera_temp_file.json: functions information
5. TINTORERA_BB_FILE.JSON

```json
{
    "file_name": "loop_tester.c",
    "func_gimples": [
        {
            "bb_data": "gcc.BasicBlock(index=0)",
            "bb_0": 0,
            "desc": "entry"
        },
        {
            "bb_1": 1,
            "bb_data": "gcc.BasicBlock(index=1)",
            "desc": "exit"
        },
        {
            "bb_2": 2,
            "bb_data": [
                {
                    "loc": "loop_tester.c:132",
                    "code": " no_loop();",
                    "repr(stmt)": "gcc.GimpleCall()",
                    "exprcode": "<type 'gcc.CallExpr'>",
                    "str(stmt)": "no_loop ()",
                    "rhs": "[<gcc.AddrExpr object at 0xaf024e8>, None]",
                    "lhs": "None",
                    "exprtype": "<gcc.VoidType object at 0xaf02c98>"
                },
                {
                    "loc": "loop_tester.c:133",
                    "code": " if_stmt();",
                    "repr(stmt)": "gcc.GimpleCall()",
                    "exprcode": "<type 'gcc.CallExpr'>",
                    "str(stmt)": "if_stmt ()",
                    "rhs": "[<gcc.AddrExpr object at 0xaf02518>, None]",
                    "lhs": "None",
                    "exprtype": "<gcc.VoidType object at 0xaf02c98>"
                }
            ]
        }
    ]
}
```
```json
{
    "file_name": "loop_tester.c",
    "file_size": 1541,
    "file_comments": [],
    "file_comments_len": 0,
    "file_lines_len": [
        {
            "line": 0,
            "len": 19
        },
        {
            "line": 1,
            "len": 1
        },
        {
            "line": 2,
            "len": 19
        }
    ],
    "file_ploc": 3,
    "file_loc": 2,
    "file_blank": 1,
    "file_metrics": {
        "total_code": 127,
        "line_max": "38",
        "line_min": "1",
        "total_bb": 74,
        "total_cc": 27,
        "cc_avg": 2,
        "total_funcs": 13,
        "total_lines": 136,
        "line_avg": "11.46666666667",
        "total_blanks": 8,
        "total_comments": 0
    },
    "file_sha256": "",
    "file_code": [
        "#include <stdio.h>\n",
        "\n",
        "void no_loop(void)\n"
    ]
}
```
5. TINTORERA_TEMP_FILE.JSON

```json
{
  "func_inline_asm": "",
  "file_name": "loop_tester.c",
  "func_api_res": [],
  "func_loc": 16,
  "func_code": [
    "int main(int argc, char argv[]) {",
    "  no_loop();",
    "  if_stmt();",
    "  if_else_stmt();",
    "  if_and_stmt();",
    "  if_or_stmt();",
    "  if_and_else_stmt();",
    "  if_or_else_stmt();",
    "  while_stmt();",
    "  for_stmt();",
    "  do_while_stmt();",
    "  switch_stmt();
    "  goto_stmt();
    ",
    "  return 0;
  }
],
  "func_comments_len": 0,
  "func_ploc": 18,
  "func_count_gimples": {
    "gimplecall": 12,
    "gimpleasm": 0,
    "gimplereturn": 1,
    "gimplelenop": 0,
    "gimplephi": 0,
    "gimplecond": 0,
    "gimpleswitch": 0,
    "gimpleassign": 1,
    "gimplelabel": 1
  },
  "func_end_line": 147,
  "func_decl": {
```
5. TINTORERA SOURCE CODE METRICS

- Current metrics:

1. Number of:
   1. Lines
   2. Code
   3. Blanks
   4. Comments
   5. Colons

2. Average line length
3. Minimum line
4. Maximum line
5. Total Basic Blocks
6. Total Cyclomatic Complexity
7. Average Cyclomatic Complexity
5. SOURCE CODE COMMENT ANALYSIS
6. Tintorera Analysis Demos
6. DEMO I: LOOP TESTER
6. DEMO I: LOOP TESTER
6. DEMO I: LOOP TESTER

IF ELSE

WHILE

SWITCH
6. DEMO II: SENDMAIL CRACKADDR (CVE2002-1337)

Pure Complexity....

Sendmail

VULNEX
6. DEMO II: SENDMAIL CRACKADDR (CVE2002-1337)
FUNCTION COMPLEXITY

Function = sendmail_crackaddr_cve2002_1337.c -> crackaddr

Function Details

<table>
<thead>
<tr>
<th>Function Name</th>
<th>crackaddr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arguments (1)</td>
<td>[u'char *]</td>
</tr>
<tr>
<td>Return Type</td>
<td>char *</td>
</tr>
<tr>
<td>Function LOC</td>
<td>247</td>
</tr>
<tr>
<td>Function Physical LOC</td>
<td>334</td>
</tr>
<tr>
<td>Function Start Line</td>
<td>56</td>
</tr>
<tr>
<td>Function End Line</td>
<td>390</td>
</tr>
<tr>
<td>Comments</td>
<td>54</td>
</tr>
<tr>
<td>Blank Lines</td>
<td>33</td>
</tr>
</tbody>
</table>

Code Details & Metrics

<table>
<thead>
<tr>
<th>Basic Blocks</th>
<th>155</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Complexity</td>
<td>89</td>
</tr>
<tr>
<td>Metrics-&gt; Count Colons</td>
<td>125</td>
</tr>
</tbody>
</table>
6. DEMO II: SENDMAIL CRACKADDR (CVE2002-1337)
FUNCTION COMPLEXITY
6. DEMO III: MONGOOSE WEB SERVER ANALYSIS

- Mongoose is the most easy to use web server on the planet. A web server of choice for Web developers (PHP, Ruby, Python, etc) and Web designers.
6. DEMO III: MONGOOSE WEB SERVER ANALYSIS

Application Summary

<<

Application Details

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Files</td>
<td>2</td>
</tr>
<tr>
<td>Total Functions</td>
<td>156</td>
</tr>
<tr>
<td>Total Basic Blocks</td>
<td>1930</td>
</tr>
<tr>
<td>Total LOC</td>
<td>3507</td>
</tr>
<tr>
<td>Total Physical LOC</td>
<td>4306</td>
</tr>
<tr>
<td>Total Comments</td>
<td>337</td>
</tr>
<tr>
<td>Total Blanks</td>
<td>460</td>
</tr>
</tbody>
</table>
6. DEMO III: MONGOOSE WEB SERVER ANALYSIS

View of API used in application

- Insecure: 41.7%
- File: 22.6%
- Process: 8.9%
- Stdio.h: 13.1%
- Network: 5.4%
- Inter_com: 1.2%
## 6. DEMO III: MONGOOSE WEB SERVER ANALYSIS

<table>
<thead>
<tr>
<th># All Funcs</th>
<th># File Funcs</th>
<th>File Name</th>
<th>Function Name</th>
<th>Basic Blocks</th>
<th>Cyclomatic Complexity</th>
<th>API Calls</th>
<th>Inline ASM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>main.c</td>
<td>main</td>
<td>7</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>main.c</td>
<td>start_mongoose</td>
<td>17</td>
<td>9</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>main.c</td>
<td>mongoose_callback</td>
<td>6</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>main.c</td>
<td>init_server_name</td>
<td>3</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>main.c</td>
<td>process_command_line_arguments</td>
<td>31</td>
<td>15</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>main.c</td>
<td>set_option</td>
<td>13</td>
<td>6</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>main.c</td>
<td>sdup</td>
<td>6</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>main.c</td>
<td>verify_document_root</td>
<td>9</td>
<td>5</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>main.c</td>
<td>show_usage_and_exit</td>
<td>9</td>
<td>3</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>main.c</td>
<td>die</td>
<td>3</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>main.c</td>
<td>signal_handler</td>
<td>3</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>mongoose.c</td>
<td>mg_start</td>
<td>34</td>
<td>17</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>mongoose.c</td>
<td>mg_stop</td>
<td>6</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>3</td>
<td>mongoose.c</td>
<td>free_context</td>
<td>14</td>
<td>6</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>mongoose.c</td>
<td>master_thread</td>
<td>19</td>
<td>8</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>5</td>
<td>mongoose.c</td>
<td>accept_new_connection</td>
<td>7</td>
<td>3</td>
<td>YES</td>
<td></td>
</tr>
</tbody>
</table>
Function = main.c -> main

Function Details

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arguments (2)</td>
<td>[u'int', u'char **']</td>
</tr>
<tr>
<td>Return Type</td>
<td>int</td>
</tr>
<tr>
<td>Function LOC</td>
<td>16</td>
</tr>
<tr>
<td>Function LOC</td>
<td>17</td>
</tr>
<tr>
<td>Function Start Line</td>
<td>882</td>
</tr>
<tr>
<td>Function End Line</td>
<td>899</td>
</tr>
<tr>
<td>Comments</td>
<td>0</td>
</tr>
<tr>
<td>Blank Lines</td>
<td>1</td>
</tr>
</tbody>
</table>

Code Details & Metrics

<table>
<thead>
<tr>
<th>Basic Blocks</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Complexity</td>
<td>2</td>
</tr>
<tr>
<td>Metrics -&gt; Count Colons</td>
<td>9</td>
</tr>
</tbody>
</table>
Boa, a high performance web server for Unix-alike computers
Application Summary

<<

Application Details

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Files</td>
<td>24</td>
</tr>
<tr>
<td>Total Functions</td>
<td>182</td>
</tr>
<tr>
<td>Total Basic Blocks</td>
<td>2714</td>
</tr>
<tr>
<td>Total LOC</td>
<td>6237</td>
</tr>
<tr>
<td>Total Physical LOC</td>
<td>8596</td>
</tr>
<tr>
<td>Total Comments</td>
<td>1470</td>
</tr>
<tr>
<td>Total Blanks</td>
<td>865</td>
</tr>
</tbody>
</table>
6. DEMO IV: BOA WEB SERVER

View of API used in application
6. DEMO IV: BOA WEB SERVER

![Total Gimples Graph]

- `gimpleasm`: 4594
- `gimpleassign`: 1626
- `gimplecall`: 1019
- `gimplecond`: 99
- `gimplelabel`: 0
- `gimplenop`: 0
- `gimplephi`: 0
- `gimplereturn`: 175
- `gimpleswitch`: 18
## 6. DEMO IV: BOA WEB SERVER

### Files Results

<table>
<thead>
<tr>
<th># All Funcs</th>
<th># File Funcs</th>
<th>File Name</th>
<th>Function Name</th>
<th>Basic Blocks</th>
<th>Cyclomatic Complexity</th>
<th>API Calls</th>
<th>Inline ASM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>index_dir.c</td>
<td>main</td>
<td>11</td>
<td>4</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>index_dir.c</td>
<td>index_directory</td>
<td>58</td>
<td>26</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>index_dir.c</td>
<td>stat</td>
<td>3</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>index_dir.c</td>
<td>select_files</td>
<td>6</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>index_dir.c</td>
<td>send_error</td>
<td>11</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>index_dir.c</td>
<td>http_escape_string</td>
<td>24</td>
<td>11</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>index_dir.c</td>
<td>html_escape_string</td>
<td>16</td>
<td>5</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>timestamp.c</td>
<td>timestamp</td>
<td>3</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>select.c</td>
<td>loop</td>
<td>42</td>
<td>23</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>select.c</td>
<td>fdset_update</td>
<td>40</td>
<td>21</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>sublog.c</td>
<td>open_gen_fd</td>
<td>8</td>
<td>3</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>sublog.c</td>
<td>open_net_fd</td>
<td>15</td>
<td>6</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>13</td>
<td>3</td>
<td>sublog.c</td>
<td>open_pipe_fd</td>
<td>12</td>
<td>5</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>util.c</td>
<td>parse_debug</td>
<td>19</td>
<td>9</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>util.c</td>
<td>print_debug_usage</td>
<td>6</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>3</td>
<td>util.c</td>
<td>strlower</td>
<td>6</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
</tbody>
</table>
# 6. DEMO IV: BOA WEB SERVER

<table>
<thead>
<tr>
<th>Function Start Line</th>
<th>46</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function End Line</td>
<td>135</td>
</tr>
<tr>
<td>Comments</td>
<td>21</td>
</tr>
<tr>
<td>Blank Lines</td>
<td>10</td>
</tr>
</tbody>
</table>

**Code Details & Metrics**

<table>
<thead>
<tr>
<th>Basic Blocks</th>
<th>42</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Complexity</td>
<td>23</td>
</tr>
<tr>
<td>Metrics-&gt; Count Colons</td>
<td>24</td>
</tr>
</tbody>
</table>

**API Details**

<table>
<thead>
<tr>
<th>API Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>generic</td>
</tr>
</tbody>
</table>

**Inline ASM Details**

```asm
volatile ("cld; rep; stosl" : 
"c" _d0, 
"D" _d1 :
"a" 0, 
"0" 32, 
"1"
&block_read_fdset._fds_bits[0] :
"memory");
```
6. DEMO V: OBFUSCATED C CODE ANALYSIS, ENDOH4.C

```c
int **F,**
V,M, N,i;
#define S 70,23
#endif/* 000-2E5*/
#define/* 2E5-2E5,2E5
*/_POSIX_C_SOURCE 199309
#include/*-2E5XXX*/<time.h>
/* 2E5-2E5X*/#include<stdio.h>
#include<stdlib.h>/*-2E5-2E5XX*//
struct timespec R={0,1E6};int j,k,m,
#define U/* -2E5X*//rand()*/2./RAND_MAX-1
#define/* 2E5*//0(p,q,i)(P[p*3+i] -P[q*3+i])
*/ IOCCC2013 IOCCC2013/*/define B(p,q,\n r)(0,q,p,0)*0(r,p,1)-0(q,p,1)*0(r,p,0))
#define A(t,n)( t*)malloc( sizeof ( t)*n)
#define E(p,q,r,s)B(p,q,r)*0(s ,p,2)+B(\n p,r,s)*0(q,p,2)+B(p,s,q)*0( /*XX*//r,p,2)
#define D(e,f)/*(c-a))s=a, a=e,e=s,f,f=\n d,d=s:0;u=a+.5;m=u+1; T[01]=91;T[2]=060;
#define C (Q[u]-X) *a+(Q[u+1]-Y)*b+(Q[u\n +2]-Z)*c,g=e=c- f*b,h=f*a-d*c,f=c,c=d*b\n -e*a,d=a,a=g ,e=b,b=h,P[k]=W/2-q/s/p*3*/
W,P[k+1]=H/2+r/s/p*H/2,T[3]=0x48,*T=033
n,u,v , w,t,WH, double *P, *Q, I, J, K, L, x, y, z ,X, Y,Z,a,b,c,d,e,f,g,h,p,q,r,s ;void o( double x){for(p=q=i=0,s=r=1;i<99; s=(s+x
 /s)/2)i%2?q+=r,r=-r:(p+=r),r*=3.14*x++/i;}
int G(int p,int q,int s,int g,int f){1<
```
## Files Results

<table>
<thead>
<tr>
<th># All Funcs</th>
<th># File Funcs</th>
<th>File Name</th>
<th>Function Name</th>
<th>Basic Blocks</th>
<th>Cyclomatic Complexity</th>
<th>API Calls</th>
<th>Inline ASM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>endoh4.c</td>
<td>main</td>
<td>93</td>
<td>37</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>endoh4.c</td>
<td>G</td>
<td>38</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>endoh4.c</td>
<td>o</td>
<td>9</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6. DEMO V: OBFUSCATED C CODE ANALYSIS, ENDOH4.C

The International Obfuscated C Code Contest - http://www.ioccc.org/
6. DEMO V: OBFUSCATED C CODE ANALYSIS, ENDOH4.C

O function

The International Obfuscated C Code Contest - http://www.ioccc.org/
6. DEMO VI: OBFUSCATED C CODE ANALYSIS, MISAKA
6. DEMO VI: OBFUSCATED C CODE ANALYSIS, MISAKA

Warning: nobold #800040 is not a known color.
Warning: nobold #800040 is not a known color.
c1: gcc-python-tree.c:549: PyObject* PyGccFunction_TypeObj_get_argument_types(PyGccTree*, void*): Assertion `size>0' failed.
*** WARNING *** there are active plugins, do not report this as a bug unless you can reproduce it without enabling any plugins.

Event | Plugins
--- | ---
PLUGIN_FINISH | python python
PLUGIN_PASS_EXECUTION | python

misaka.c: In function 'S':

misaka.c:15:33: internal compiler error: Aborted

Please submit a full bug report, with preprocessed source if appropriate.
See <file:///usr/share/doc/gcc-4.7/README.Bugs> for instructions.

root@kali:~/Desktop/hitb_demos/code/obfuscated_c/2013/misaka#
6. DEMO VI: OBFUSCATED C CODE ANALYSIS, MISAKA

The International Obfuscated C Code Contest - http://www.ioccc.org/
7. Conclusions
7. DRAWBACKS

- gcc-python-plugin needs more work, fails many times

- So do Tintorera...

- Only C / C++ code
7. CONCLUSIONS

• Tintorera helps to analyze C code faster & better

• Practical code understanding for:
  – Saving time
  – Security reviews
  – Fuzzing: what and where to fuzz
7. NEXT STEPS

• Better & focused analysis (security, etc.)

• Vulnerabilities Detection

• More metrics

• Code Diff

• Cooler reports!

• Other languages ¿?
8. Q&A

- Thanks!

- @simonroses / @vulnexsl

- www.vulnex.com