DIMSUM: Discovering of Semantic Data of Interest from Un-mappable Memory with Confidence

Zhiqiang Lin¹, Junghwan Rhee², Chao Wu³, Xiangyu Zhang³
Dongyan Xu³

¹University of Texas at Dallas
²NEC Laboratories America
³Purdue University
The Problem: Memory Forensics

- **Given:**
  - A set of memory pages
  - A data structure of interest (e.g., contact, cookie, chat history)

- **Identifying:**
  - Instances of that data structure in the memory pages

- **Assuming:**
  - No memory mapping information (e.g., page table)
Observation:
*Dead Pages Left by Terminated Processes*
State of the Art

- **Value invariant-based approaches**
  - Klist [Rutkowski, 2003]
  - GREPEXEC [bugcheck, 2006]
  - Volatility [Walters, 2006] [Schuster, 2006]
  - Robust signatures [Dolan-Gavitt et al., CCS’09]

- **Pointer navigation-based approaches**
  - KOP [Carbone et al., CCS’09], CRASH [USENIX’05]
  - SigGraph [Lin et al., NDSS’11]
Use of Memory Mapping Information

Where is (0x08048008) pointing to?

![Diagram showing memory mapping information with virtual page number 08048 and page offset 008 pointing to a page table with virtual page number and physical page number entries, leading to a physical address.]
DIMSUM Overview

Discovering Information with Semantics from Un-mappable Memory

Constraints

Data Structure Definition

Physical Memory Pages

Probabilistic Inference

Results

Constraints

Primitive

Pointer

Structural

Same Page

Semantic

Staged
type = struct passwd {
    char *pw_name;       //00
    char *pw_passwd;     //04
    __uid_t pw_uid;      //08
    __gid_t pw_gid;      //12
    char *pw_gecos;      //16
    char *pw_dir;        //20
    char *pw_shell;      //24
}

p(pw_name) ∧ p(pw_passwd) ∧ I(pw_uid) ∧ I(pw_gid) ∧ p(pw_gecos) ∧ p(pw_dir) ∧ p(pw_shell)
Semantic Constraint

```c
struct passwd {
    char *pw_name; //00
    char *pw_passwd; //04
    __uid_t pw_uid; //08
    __gid_t pw_gid; //12
    char *pw_gecos; //16
    char *pw_dir; //20
    char *pw_shell; //24
}

p(pw_name) \land p(pw_passwd) \land I(pw_uid) \land I(pw_gid) \land p(pw_gecos) \land p(pw_dir) \land p(pw_shell) \land (pw_uid \geq 0) \land (pw_gid \geq 0)
```
Same-Page (SP) Constraint

type = struct passwd {
    char *pw_name;     //00
    char *pw_passwd;   //04
    __uid_t pw_uid;    //08
    __gid_t pw_gid;    //12
    char *pw_gecos;    //16
    char *pw_dir;      //20
    char *pw_shell;    //24
}


\begin{verbatim}
00000970 00 00 00 00 70 ac e6 08 75 ac e6 08 00 00 00 00
00000980 00 00 00 00 7b ac e6 08 80 ac e6 08 86 ac e6 08
00000990 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000c70 72 6f 6f 74 00 78 00 30 3a 30 3a 72 6f 6f 74 00
00000c80 2f 72 6f 6f 74 00 2f 62 69 6e 2f 62 61 73 68 00
00000c90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
\end{verbatim}
Probabilistic Inference Model

- **Boolean variable** $x$: type property of a memory location
  - $x_1$: There is an instance of passwd at offset 0 of page 100
  - $x_2$: There is a *char pointer at offset 0 of page 100
  - $x_3$: There is a *char pointer at offset 4 of page 100

- **Constraint** $C_j$: type/structural/semantic property for one or more memory locations
  - $C_1$: $x_1 \rightarrow x_2 \land x_3 \land ...$
  - $C_2$: $x_1 \rightarrow same\_page(*0^p \land *4^p \land ...)$
Valuation function $f_{Cj}$: constraint $Cj$’s evaluation with confidence

$\forall \quad f_{c1} = \begin{cases} 1; & \text{if } C1 = 1 \\ 0; & \text{otherwise} \end{cases}$

$\forall \quad f_{c2} = \begin{cases} \delta; & \text{if } \exists p \text{ such that } C2 = 1 \\ 1 - \delta; & \text{otherwise} \end{cases}$
Probabilistic Inference Model (Cont.)

- Joint probability function:
  \[ p(x_1, x_2 \ldots x_n) = \frac{f_{c_1} \times f_{c_2} \times \ldots \times f_{c_m}}{Z} \]
  \[ Z = \sum_{x_1, x_2 \ldots x_n} (f_{c_1} \times f_{c_2} \times \ldots \times f_{c_m}) \]

- Marginal probability:
  \[ p(x_i = 1) = \sum_{x_1, x_2 \ldots x_{i-1} x_{i+1} \ldots x_n} p(x_1, x_2 \ldots 1 \ldots x_n) \]
  \[ p(x_1 = 1): \text{probability of having an instance of passwd at offset 0 of page 100} \]

- Implemented using *Infer.NET* engine
# Evaluation Results with Linux-based Applications

<table>
<thead>
<tr>
<th>Data Structure of Interest</th>
<th>% of Memory Pages</th>
<th>True Instances</th>
<th>SigGraph+</th>
<th>DIMSUM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>FP%</td>
<td>FN%</td>
</tr>
<tr>
<td>Login record (last)</td>
<td>100.0</td>
<td>8</td>
<td>0.0</td>
<td>25.0</td>
</tr>
<tr>
<td></td>
<td>66.7</td>
<td>6</td>
<td>0.0</td>
<td>66.7</td>
</tr>
<tr>
<td></td>
<td>33.3</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Browser cookies (chromium)</td>
<td>100.0</td>
<td>25</td>
<td>69.5</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>66.7</td>
<td>19</td>
<td>66.1</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>33.3</td>
<td>9</td>
<td>79.1</td>
<td>0.0</td>
</tr>
<tr>
<td>Address book (pine-4.64)</td>
<td>100.0</td>
<td>124</td>
<td>48.5</td>
<td>4.8</td>
</tr>
<tr>
<td></td>
<td>66.7</td>
<td>96</td>
<td>50.1</td>
<td>10.4</td>
</tr>
<tr>
<td></td>
<td>33.3</td>
<td>63</td>
<td>56.8</td>
<td>39.7</td>
</tr>
<tr>
<td>Contact list (pidgin)</td>
<td>100.0</td>
<td>300</td>
<td>38.8</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>66.7</td>
<td>198</td>
<td>22.8</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>33.3</td>
<td>98</td>
<td>23.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>
struct utmplist {
  00: short int ut_type;
  04: pid_t ut_pid;
  08: char ut_line[32];
  40: char ut_id[4];
  44: char ut_user[32];
  76: char ut_host[256];
  332: long int ut_termination;
  336: long int ut_session;
  340: struct timeval ut_tv;
  348: int32_t ut_addr_v6[4];
  364: char __unused[20];
  384: struct utmplist *next;
  388: struct utmplist *prev;
};
All Dead Pages Available

Probability

Binary Value: S, V

Threshold: 0.5

False Negative (FN) of S
False Positive (FP) of V
True Positive (TP) of D
True Positive (TP) of S

Byte Offset in Memory Dump

19000 20000 21000 22000 23000 24000 25000 2e+07 4e+07 6e+07 8e+07 1e+08
## Results with Android 2.1 Applications

<table>
<thead>
<tr>
<th>Data Structure of Interest</th>
<th>% of Mem. Pages</th>
<th>True Instances</th>
<th>SigGraph+</th>
<th>DIMSUM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>FP%</td>
<td>FN%</td>
</tr>
<tr>
<td><strong>Cookie (Browser)</strong></td>
<td>100.0</td>
<td>31</td>
<td>77.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>66.7</td>
<td>25</td>
<td>75.5</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>33.3</td>
<td>6</td>
<td>85.8</td>
<td>16.7</td>
</tr>
<tr>
<td><strong>Phone Contact (Messaging)</strong></td>
<td>100.0</td>
<td>117</td>
<td>0.9</td>
<td>4.3</td>
</tr>
<tr>
<td></td>
<td>66.7</td>
<td>79</td>
<td>0.0</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>33.3</td>
<td>36</td>
<td>2.9</td>
<td>5.6</td>
</tr>
<tr>
<td><strong>Message Conversation (Messaging)</strong></td>
<td>100.0</td>
<td>101</td>
<td>0.0</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>66.7</td>
<td>60</td>
<td>0.0</td>
<td>1.7</td>
</tr>
<tr>
<td></td>
<td>33.3</td>
<td>40</td>
<td>0.0</td>
<td>2.5</td>
</tr>
</tbody>
</table>
Other Related Work

- ColdBoot [Halderman et al, USENIX Security’08]
- Laika [Cozzie et al, OSDI’08]
- DECODE [Walls et al, USENIX Security’11]
Conclusion

- DIMSUM recognizes data structure instances from memory pages
  - Without memory mapping information
  - Based on probabilistic inference
  - Solving constraints about type/structural/semantic properties
  - More accurate than non-probabilistic approaches

Thank you
67% of Dead Pages Available
33% of Dead Pages Available
An Android-Specific Constraint

```java
class test{
    int a;
    string b,
    float c;
}
```