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

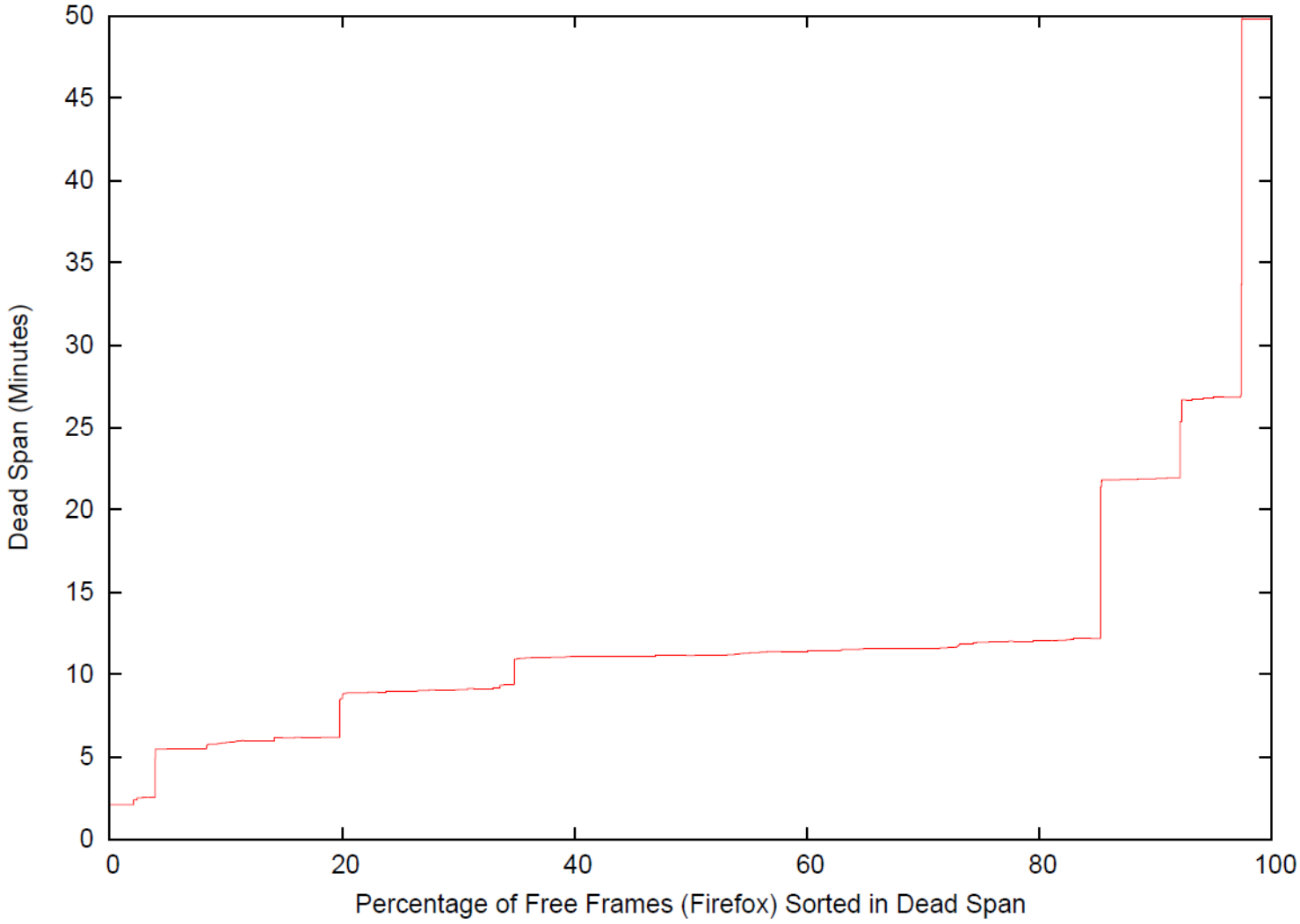
Zhiqiang Lin[1] , Junghwan Rhee[2], Chao Wu[3], Xiangyu Zhang[3]
Dongyan Xu[3]

[1]University of Texas at Dallas
[2]NEC Laboratories America
[3]Purdue 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

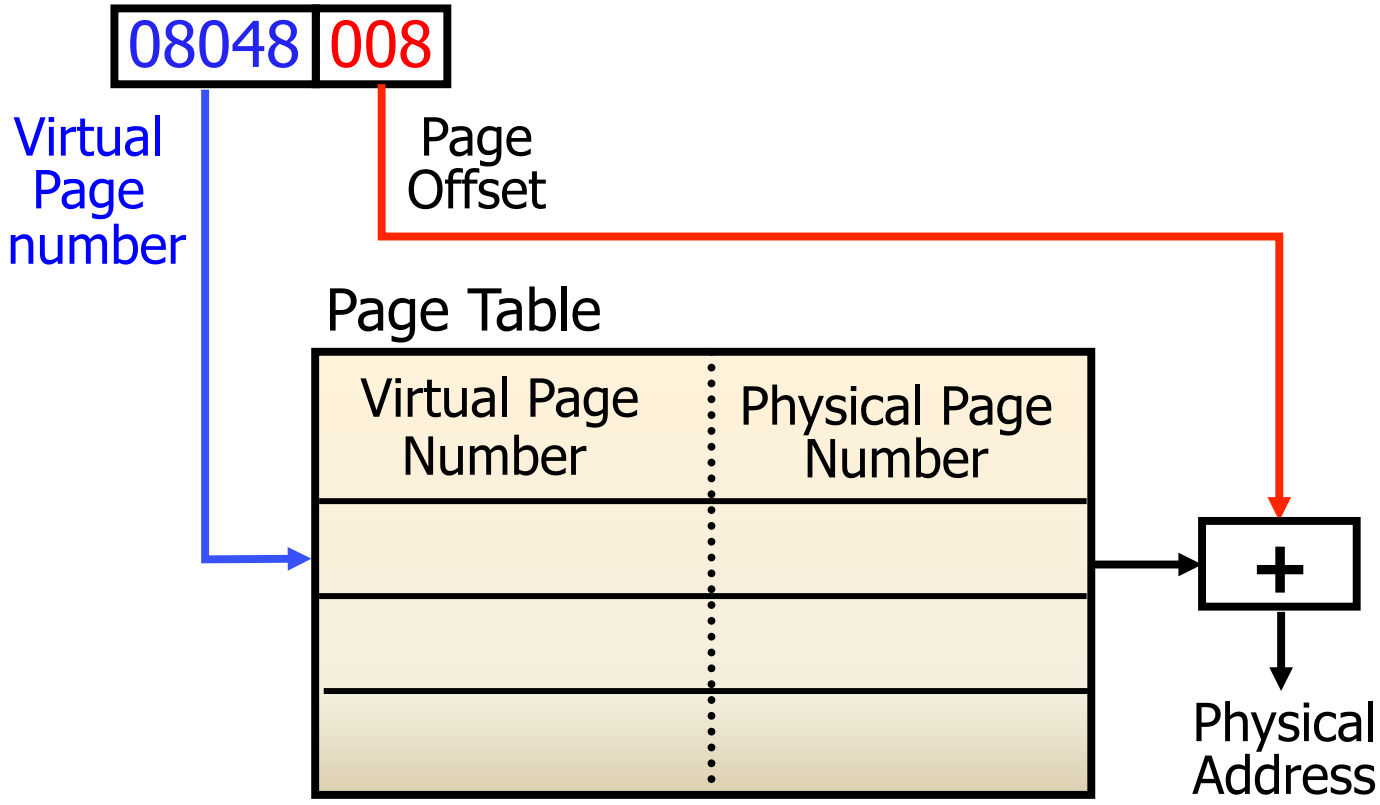
- Value invariant-based approaches
  - Klist [Rutkowska,2003]
  - GREPEXEC [bugcheck, 2006]
  - Volatility [Walters, 2006] [Schuster, 2006]
  - Robust signatures [Dolan-Gavitt et al., CCS'09]

- Pointer navigation-based approaches
  - KOP [Carbone et al.,CCS'09], CRASH [USENIX'05]
  - SigGraph [Lin et al., NDSS'11]

Use of Memory Mapping Information

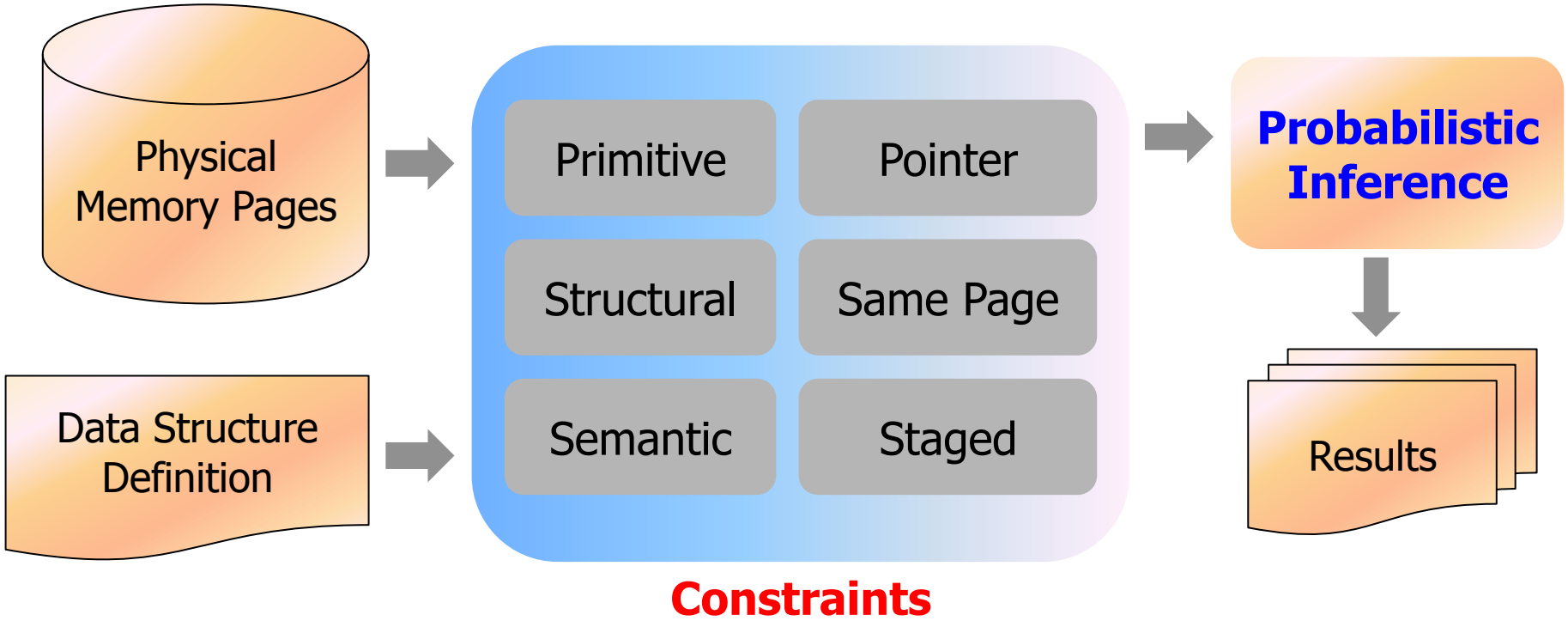
```
000001f0: 08 80 04 08
```

Where is (0x08048008) pointing to?



DIMSUM Overview

Discovering **I**nfor**M**ation with
Semantics from **U**n-mappable **M**emory



Structural 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
}
```

$p(pw\_name) \wedge p(pw\_passwd) \wedge I(pw\_uid) \wedge I(pw\_gid) \wedge p(pw\_gecos) \wedge p(pw\_dir) \wedge p(pw\_shell)$

Semantic 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
}
```

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

(pw_uid >=0) ∧ (pw_gid >=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
}
```

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

(pw_uid >=0) ∧ (pw_gid >=0)

SP(*pw_name, *pw_passwd, *pw_gecos, *pw_dir, *pw_shell)

```
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
```

```
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
```

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 Cj: type/structural/semantic property for one or more memory locations
  - $C1$: $x_1 \rightarrow x_2 \wedge x_3 \wedge \ \ldots$
  - $C2$: $x_1 \rightarrow same\_page(*\,0^p \wedge *\,4^p \wedge \ \ldots)$

Probabilistic Inference Model (Cont.)

- Valuation function f_{Cj}: constraint Cj's evaluation with confidence
  - $f_{C1} = \begin{cases} 1; \ if \ C1 = 1 \\ 0; otherwise \end{cases}$
  - $f_{C2} = \begin{cases} \delta; \ if \ \exists p \ such \ that \ C2 = 1 \\ 1 - \delta; otherwise \end{cases}$

Probabilistic Inference Model (Cont.)

- Joint probability function:
  - $p(x_1, x_2 \ldots x_n) = \frac{f_{C1} \times f_{C2} \times \ldots \times f_{Cm}}{Z}$
  - $Z = \sum_{x_1, x_2 \ldots x_n} (f_{C1} \times f_{C2} \times \cdots \times f_{Cm})$
- Marginal probability:
  - $p(x_i = 1) = \sum_{x_1, x_2 \ldots x_{i\_1} x_{i+1} \ldots x_n} p(x_1, x_2 \ldots 1 \ldots x_n)$
  - $p(x_1 = 1)$: probability of having an instance of passwd at offset 0 of page 100
- Implemented using *Infer.NET* engine

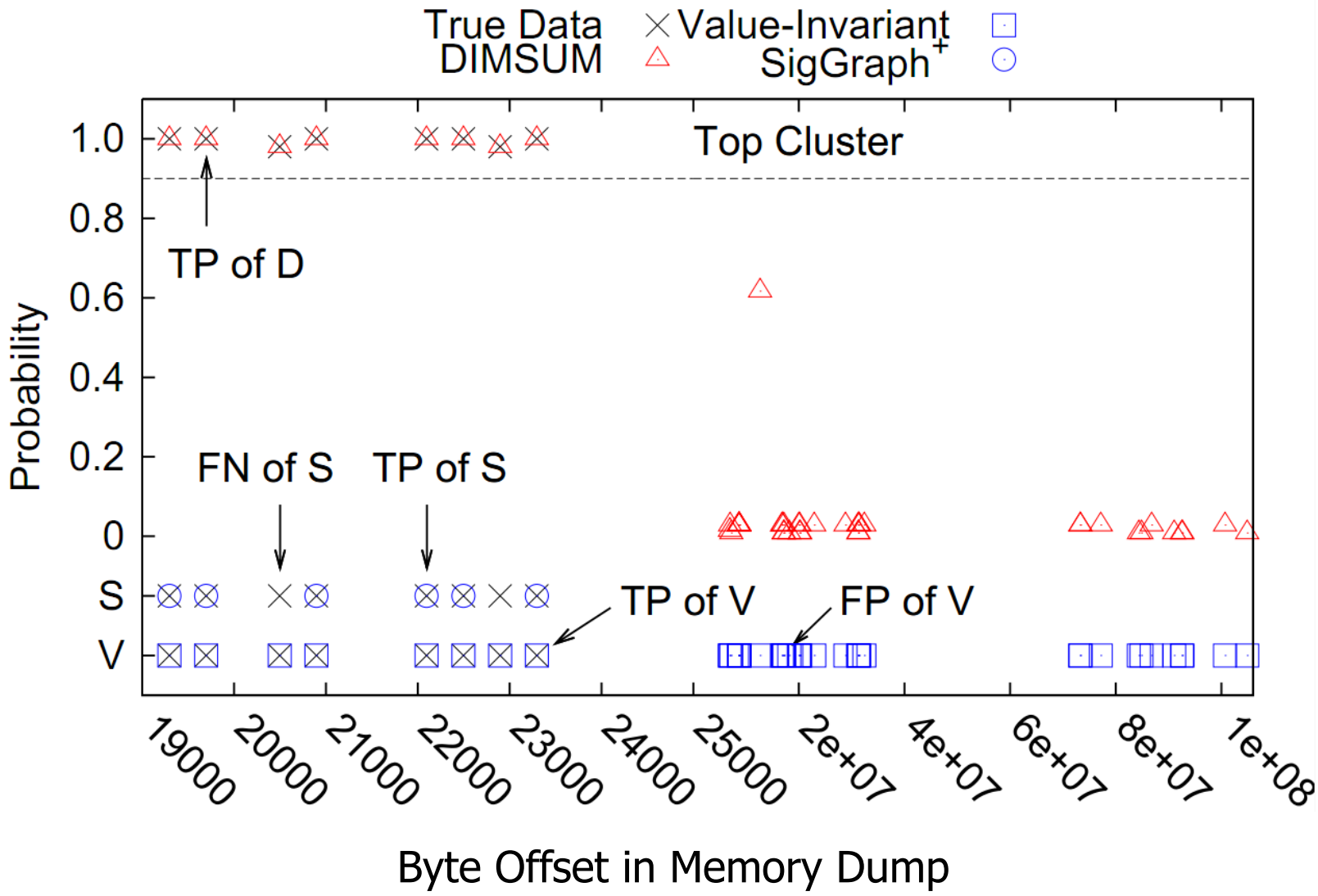
Evaluation Results with Linux-based Applications

Data Structure of Interest	% of Memory Pages	True Instances	SigGraph+		DIMSUM	
			FP%	FN%	FP%	FN%
Login record (`last`)	100.0	8	0.0	25.0	0.0	0.0
	66.7	6	0.0	66.7	0.0	0.0
	33.3	0	0.0	0.0	100.0*	0.0
Browser cookies (`chromium`)	100.0	25	69.5	0.0	44.4	0.0
	66.7	19	66.1	0.0	50.0	0.0
	33.3	9	79.1	0.0	43.8	0.0
Address book (`pine-4.64`)	100.0	124	48.5	4.8	0.0	18.5
	66.7	96	50.1	10.4	0.0	17.7
	33.3	63	56.8	39.7	0.0	33.3
Contact list (`pidgin`)	100.0	300	38.8	0.0	0.0	1.0
	66.7	198	22.8	0.0	0.0	1.0
	33.3	98	23.0	0.0	0.0	1.0

Case Study: Login Record `utmp` in `last`

```
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_etermination;
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



Results with Android 2.1 Applications

Data Structure of Interest	% of Mem. Pages	True Instances	SigGraph+		DIMSUM	
			FP%	FN%	FP%	FN%
Cookie (Browser)	100.0	31	77.0	0.0	0.0	0.0
	66.7	25	75.5	0.0	0.0	0.0
	33.3	6	85.8	16.7	0.0	0.0
Phone Contact (Messaging)	100.0	117	0.9	4.3	0.0	0.0
	66.7	79	0.0	3.8	0.0	0.0
	33.3	36	2.9	5.6	0.0	0.0
Message Conversation (Messaging)	100.0	101	0.0	2.0	0.0	0.0
	66.7	60	0.0	1.7	0.0	0.0
	33.3	40	0.0	2.5	0.0	0.0

Other Related Work

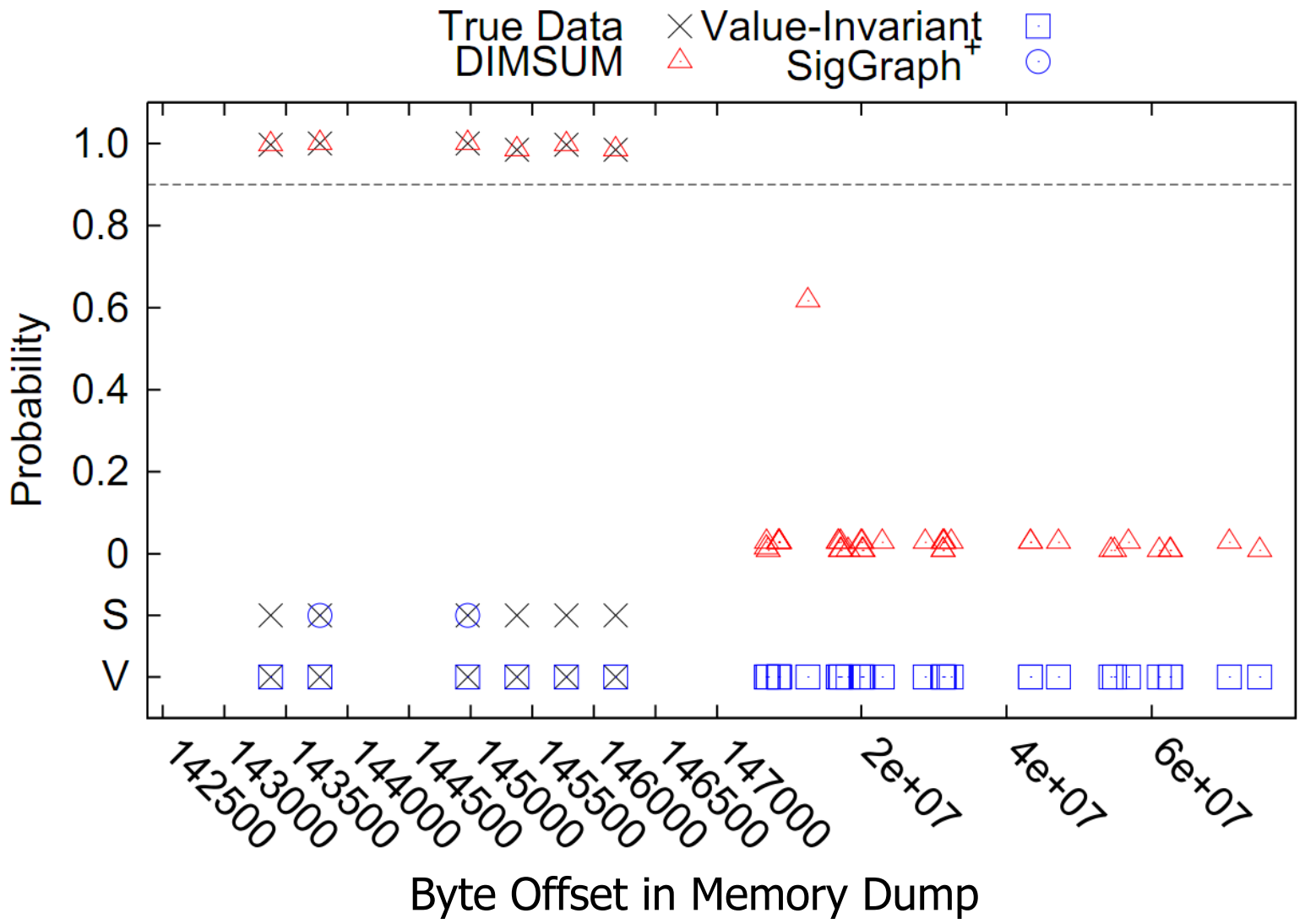
- ColdBoot [Halderman et al, USENIX Security'08]
- Laika [Cozzie et al, OSDI'08]
- DEC0DE [Walls et al, USENIX Security'11]

Conclusion

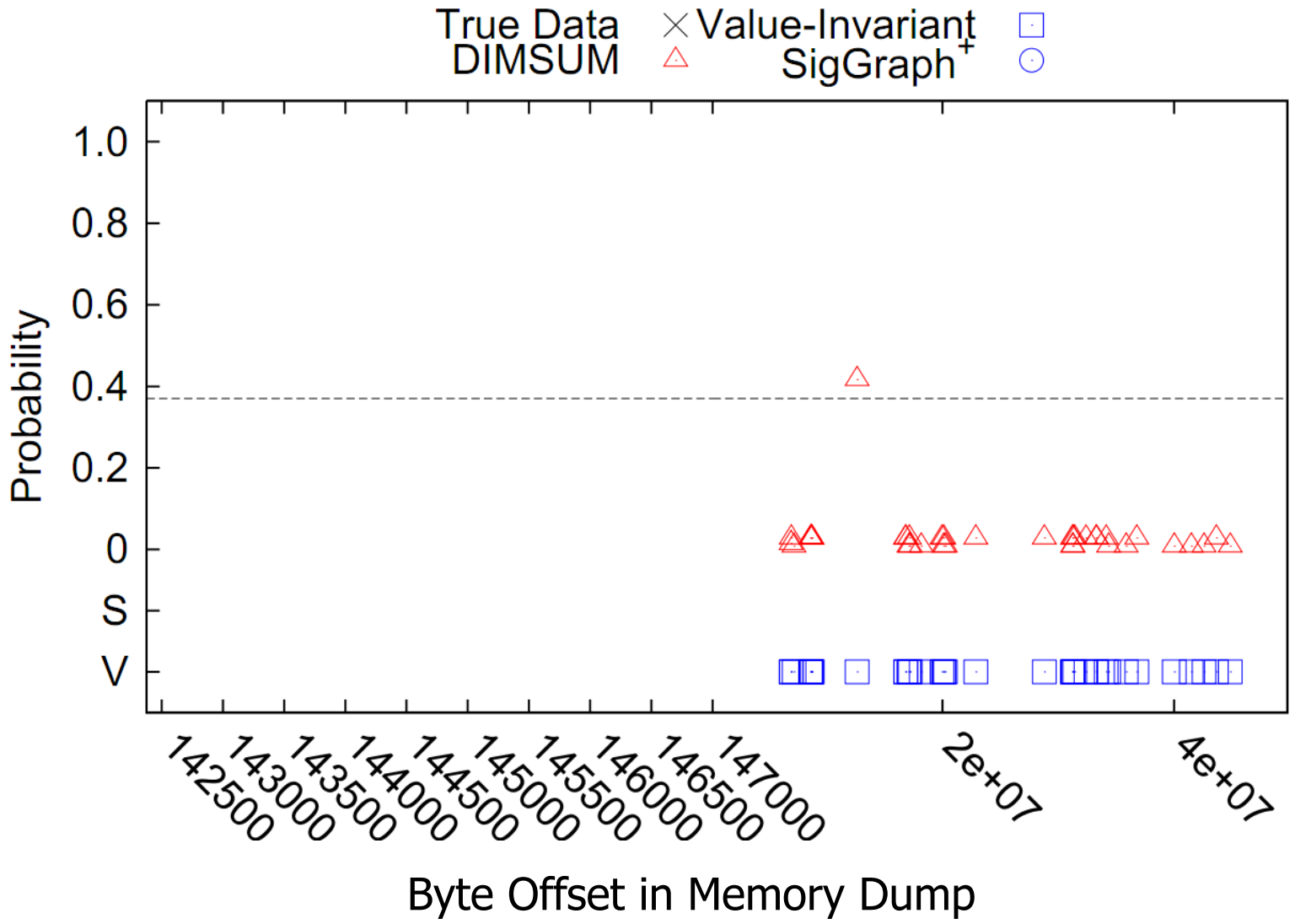
- DIMSUM recognizes data structure instances from memory pages
  - Without memory mapping information
  - Based on probabilistic inference
  - Solving constraints about type/structural/semantic properties
  - More accurate than non-probabilistic approaches

Thank you

67% of Dead Pages Available



33% of Dead Pages Available



An Android-Specific Constraint