Listening to Programmers

/* -- Taxonomies and Characteristics of Comments in Operating System Code */

Lin Tan

Yoann Padioleau, Lin Tan, Yuanyuan Zhou
University of Illinois, Urbana-Champaign

Motivation

- Many innovations to improve software quality & productivity:
  - PL, IDE, bug detection tools, annotation languages, ...
  - Valgrind, Splint, Linux's Sparse, Microsoft SAL, TagSEA, Mylyn, ...

Motivation

- Many innovations to improve software quality & productivity:
  - PL, IDE, bug detection tools, annotation languages, ...
  - Valgrind, Splint, Linux's Sparse, Microsoft SAL, TagSEA, Mylyn, ...

- Helpful to know what developers want

Comments Reveal Needs

- User studies:
  - Examples: CMU, Microsoft Research HIP
  - Challenges: Hard to collect representative data

Comments Reveal Needs

- User studies:
  - Examples: CMU, Microsoft Research HIP
  - Challenges: Hard to collect representative data

- Our novel observation:
  - **Comments reveal what developers want.**

Potential of Comments (I)

```
opensolaris/sun/io/ms.c:
timeout_id_t  msd_timeout_id; /* id returned
by timeout() */
```

- Developers want to express code relationships

Potential of Comments (I)

```
opensolaris/sun/io/ms.c:
timeout_id_t  msd_timeout_id; /* id returned
by timeout() */
```

- Developers want to express code relationships
- Motivate language support & bug detection tools

Potential of Comments (I)

```
opensolaris/sun/io/ms.c:
timeout_id_t  msd_timeout_id; /* id returned
by timeout() */
```

- Developers want to express code relationships
- Motivate language support & bug detection tools
- Motivate IDE features for better navigation capability

Potential of Comments (II)

```
linux/drivers/scsi/in2000.c:
/* Caller must hold instance lock!*/
static int reset_hardware(…){…}
```

Potential of Comments (II)

```
linux/drivers/scsi/in2000.c:
/* Caller must hold instance lock!*/
static int reset_hardware(…){…}
```

```
static int in2000_bus_reset(…){ …

    reset_hardware(…); …
}
```

Potential of Comments (II)

```
linux/drivers/scsi/in2000.c:
/* Caller must hold instance lock!*/
static int reset_hardware(…){…}
```

```
static int in2000_bus_reset(…){ …
    reset_hardware(…); …
}
```

No lock acquisition ⇒ A bug!

Potential of Comments (II)

```
linux/drivers/scsi/in2000.c:
/* Caller must hold instance lock!*/
static int reset_hardware(…){…}
```

```
static int in2000_bus_reset(…){ …
    No lock acquisition ⇒ A bug!
    reset_hardware(…); …
}
```

- Developers want to express assumptions/intentions.
- Motivate bug detection tools [TanSOSP'07]

Potential of Comments (II)

```
linux/drivers/scsi/in2000.c:
/* Caller must hold instance lock!*/
static int reset_hardware(…){…}
```

```
static int in2000_bus_reset(…){ …
    No lock acquisition ⇒ A bug!
    reset_hardware(…); …
}
```

- Developers want to express assumptions/intentions.
- Motivate bug detection tools [TanSOSP'07]

See our paper for more examples.

Prevalence of Comments

Software	Linux	FreeBSD	OpenSolaris	Mozilla	MySQL	Eclipse
Lines of Comments	1.2M	0.6M	1.1M	1.2M	0.3M	1.7M

(kernel only for OSs, excluding blank lines, including copyright notices)

- Millions lines of comments (23-30%) exist.
  - Various languages: C, C++, Java
  - Written by thousands of developers or more

Our Contributions

- First comprehensive comment study on semantics:
  - Manually examine 2100 randomly sampled comments from 6 large popular software projects (in C, C++, Java)
  - Many findings and implications:
    - Provide guidance to the design of tools/languages
  - New comment taxonomies & analysis tools
    - Available at http://ece.uwaterloo.ca/~lintan/CComment

Outline

- Motivation
- Methodology & Taxonomies
- OS Comments: Findings and Implications
- Non-OS Comments: Similarities & Differences
- Related Work & Conclusions

Our OS Comment Source

Software	Linux	FreeBSD	OpenSolaris
Lines of Code	5.2M	2.4M	3.7M
Lines of Comments	1.2M	0.6M	1.1M
% of Comments	23.1%	25.0%	29.7%

(kernel only, excluding blank lines, including copyright notices)

Our OS Comment Source

Software	Linux	FreeBSD	OpenSolaris
Lines of Code	5.2M	2.4M	3.7M
Lines of Comments	1.2M	0.6M	1.1M
% of Comments	23.1%	25.0%	29.7%
# of Comments (delimited by /* */ or //)	729,923	289,413	380,111

(kernel only, excluding blank lines, including copyright notices)

Our OS Comment Source

Software	Linux	FreeBSD	OpenSolaris
Lines of Code	5.2M	2.4M	3.7M
Lines of Comments	1.2M	0.6M	1.1M
% of Comments	23.1%	25.0%	29.7%
# of Comments (delimited by /* */ or //)	729,923	289,413	380,111
Sample Size	350	350	350

(kernel only, excluding blank lines, including copyright notices)

- Randomly sampled 1050 comments
  - Causing reasonably small margin of error

Our OS Comment Source

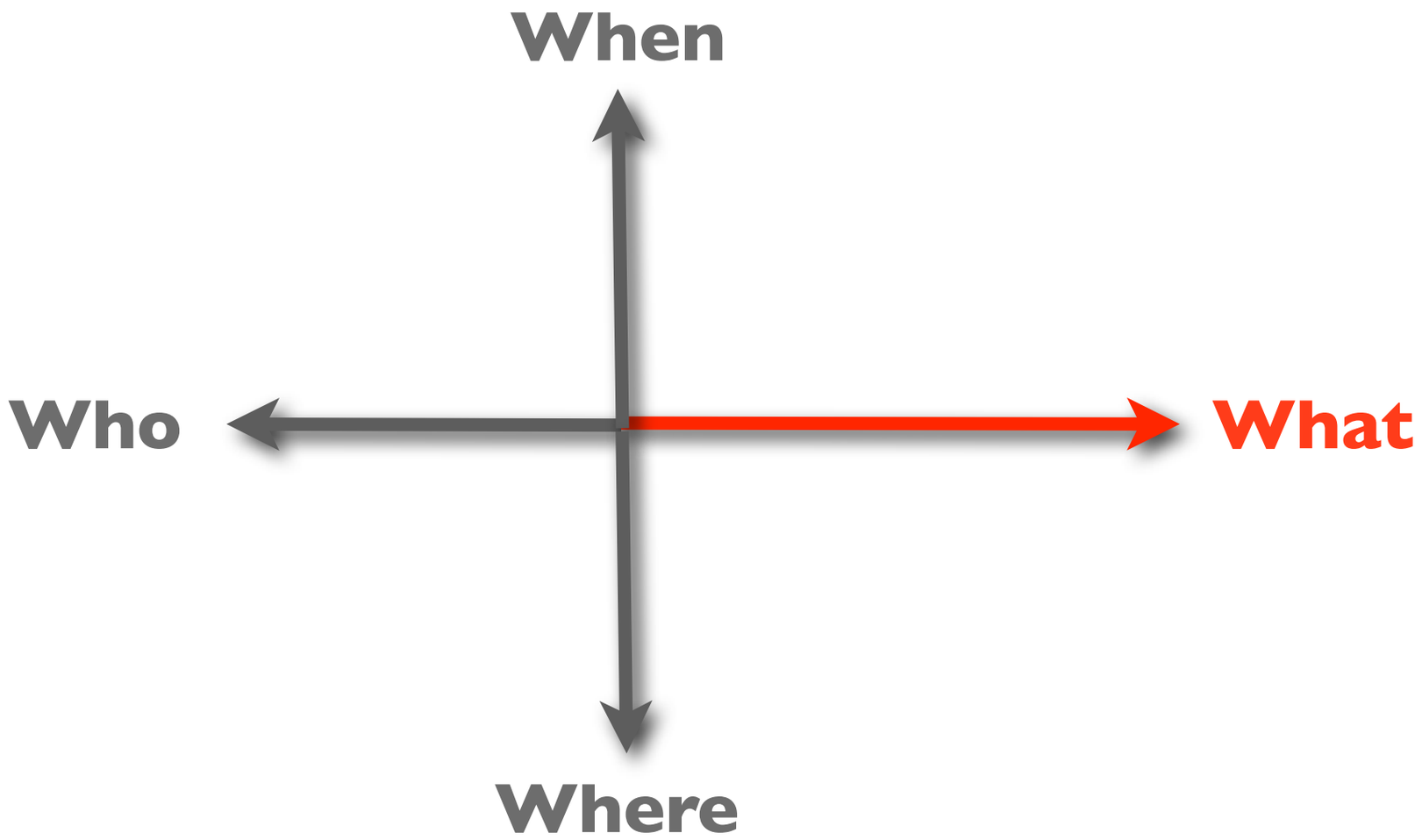
Software	Linux	FreeBSD	OpenSolaris
Lines of Code	5.2M	2.4M	3.7M
Lines of Comments	1.2M	0.6M	1.1M
% of Comments	23.1%	25.0%	29.7%
# of Comments (delimited by /* */ or //)	729,923	289,413	380,111
Sample Size	350	350	350

Java & C++ code later

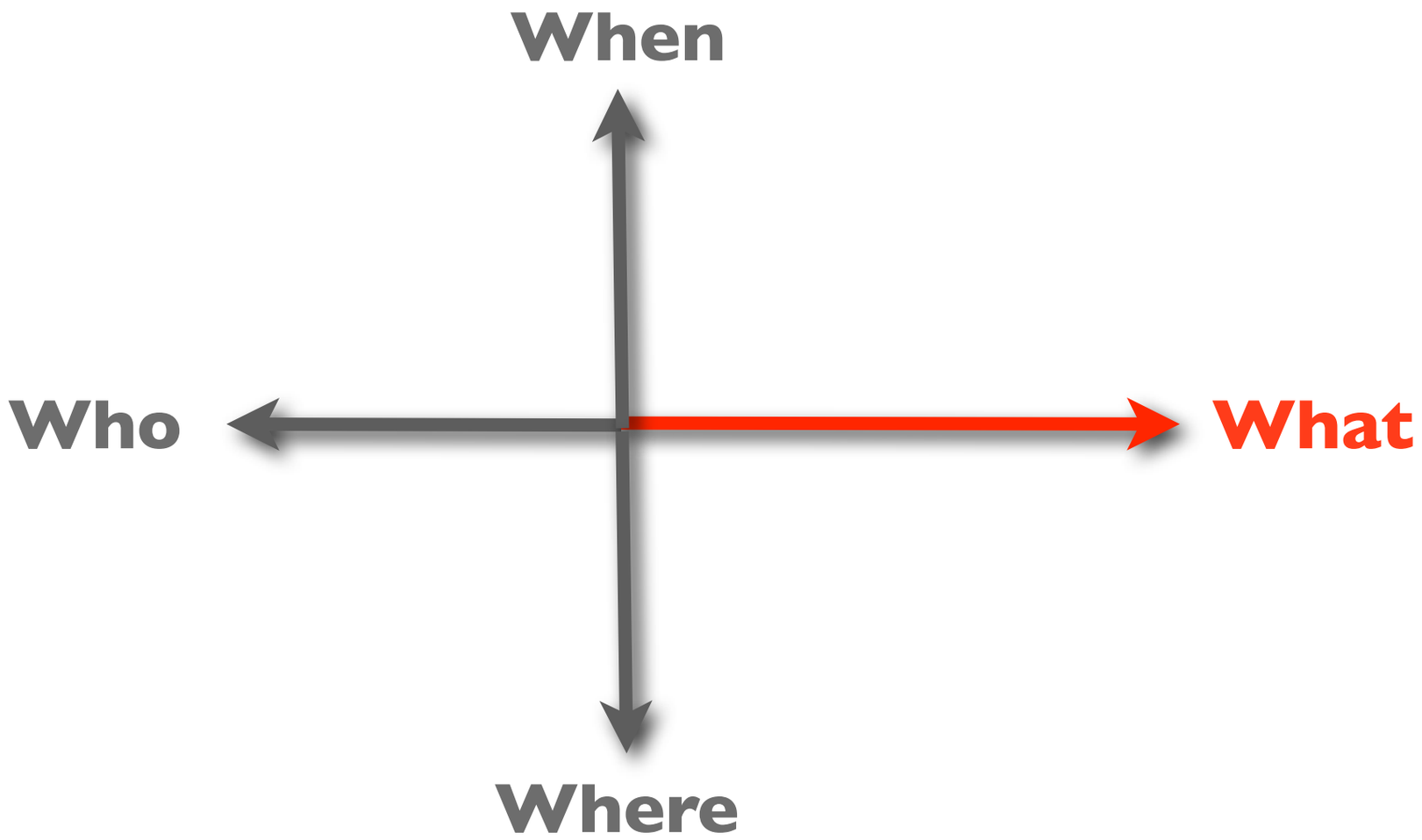
(kernel only, excluding blank lines, including copyright notices)

- Randomly sampled 1050 comments
  - Causing reasonably small margin of error

Dimensions of Taxonomies

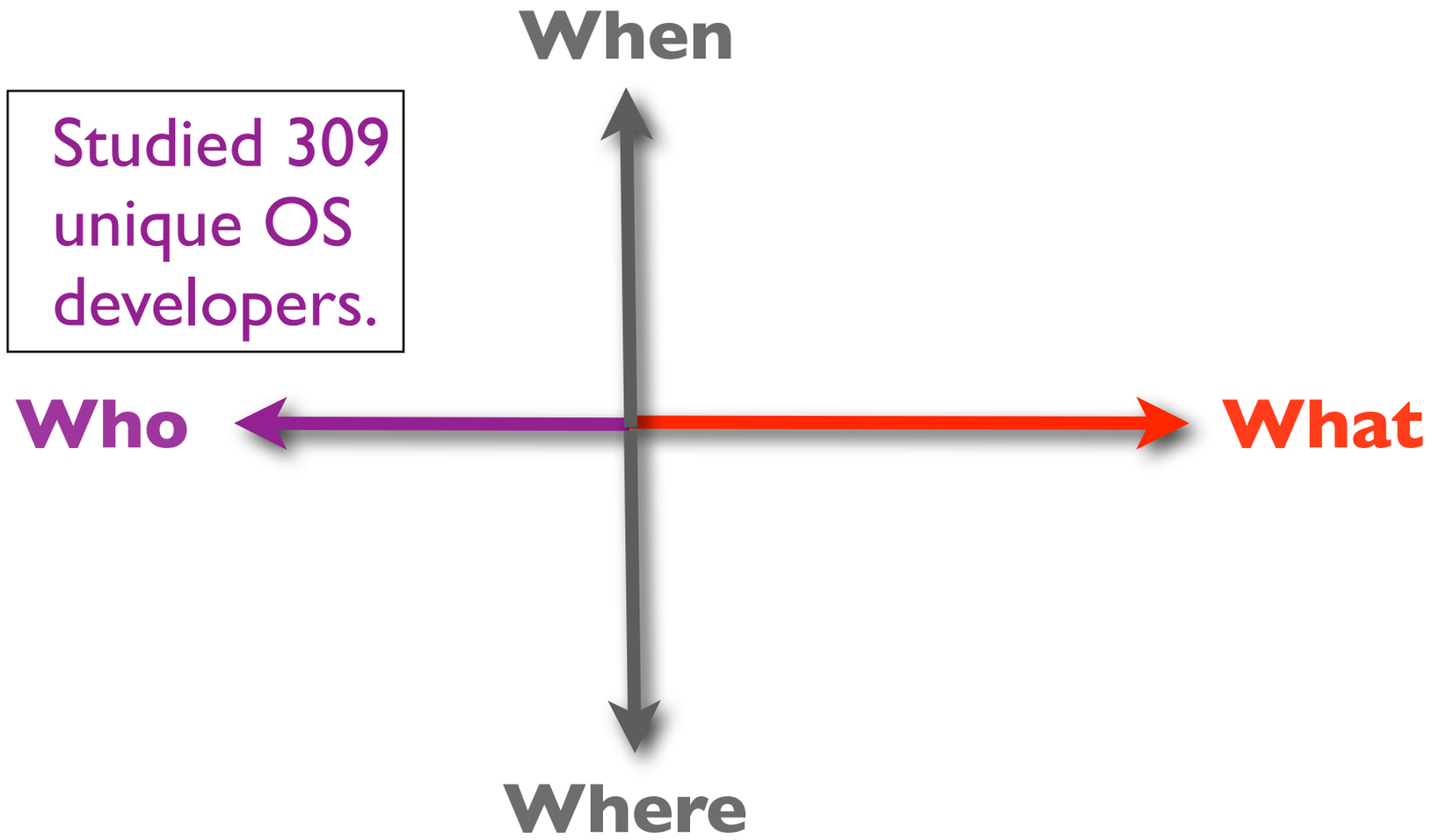


Dimensions of Taxonomies



- What can be utilized & how much?
- How to use these comments?

Dimensions of Taxonomies



- What can be utilized & how much?
- How to use these comments?

Classification Process

- Iterative process
- Double verification
- A tool to help
  - Automatically extract author, time and related entities

Threats to Validity

- Focused on OS/C code (only 3 C++/Java projects)

Threats to Validity

- Focused on OS/C code (only 3 C++/Java projects)
- Sampled 2100 comments

Threats to Validity

- Focused on OS/C code (only 3 C++/Java projects)
- Sampled 2100 comments
- Same amount of comments from each software

Threats to Validity

- Focused on OS/C code (only 3 C++/Java projects)
- Sampled 2100 comments
- Same amount of comments from each software
- Subjectivity

Threats to Validity

- Focused on OS/C code (only 3 C++/Java projects)
- Sampled 2100 comments
- Same amount of comments from each software
- Subjectivity
- Outdated comments

Outline

- Motivation
- Methodology & Taxonomies
- OS Comments: Findings and Implications
- Non-OS Comments: Similarities & Differences
- Related Work & Conclusions

Exploitable Comments

- At least 52.6 ±2.9% or ~736,109 comments in the 3 OSs:
  - Could be leveraged by existing or to-be-proposed techniques
  - Could guide the design of language features, IDE features, annotation languages and bug detection tools

Finding 1: Integers

- 22.1% of the exploitable comments clarify the usage and meaning of integers and integer macros.

```
1. #define E1000_MCC  0x0401C  /* Multiple Collision Count - R/clr */
   #define E1000_RCTL 0x00100  /* Rx Control - RW */
```

Finding 1: Integers

- 22.1% of the exploitable comments clarify the usage and meaning of integers and integer macros.

```
1. #define E1000_MCC  0x0401C  /* Multiple Collision Count - R/clr */
   #define E1000_RCTL 0x00100  /* Rx Control - RW */
```

E1000_**READ**_REG (hw, **E1000_MCC);**

E1000_**WRITE**_REG (hw, **E1000_MCC);**

Finding 1: Integers

- 22.1% of the exploitable comments clarify the usage and meaning of integers and integer macros.

1. #define **E1000_MCC** 0x0401C /* Multiple Collision Count - **R**/clr */
   #define **E1000_RCTL** 0x00100 /* Rx Control - **RW** */

E1000_**READ**_REG (hw, **E1000_MCC);**

E1000_**WRITE**_REG (hw, **E1000_MCC);**

Const doesn't solve the problem!

Finding 1: Integers

- 22.1% of the exploitable comments clarify the usage and meaning of integers and integer macros.

1. #define **E1000_MCC** 0x0401C /* Multiple Collision Count - **R**/clr */
   #define **E1000_RCTL** 0x00100 /* Rx Control - **RW** */

E1000_**READ**_REG (hw, **E1000_MCC);**

E1000_**WRITE**_REG (hw, **E1000_MCC);**

Const doesn't solve the problem!

2. #define EMU_DOCK_MINOR_REV 0x26 /* **0000xxx 3 bit** ... Minor rev */

3. int mem; /* memory in 128 **MB** units */

Finding 1: Integers

- 22.1% of the exploitable comments clarify the usage and meaning of integers and integer macros.

1. #define **E1000_MCC** 0x0401C /* Multiple Collision Count - **R**/clr */
   #define **E1000_RCTL** 0x00100 /* Rx Control - **RW** */

E1000_**READ**_REG (hw, **E1000_MCC);**

E1000_**WRITE**_REG (hw, **E1000_MCC);**

Const doesn't solve the problem!

2. #define EMU_DOCK_MINOR_REV 0x26 /* **0000xxx 3 bit** ... Minor rev */

3. int mem; /* memory in 128 **MB** units */

- Should pay more attention to integers and integer macros
  - Domain specific languages, extended types, bug detection tools, ...

Finding 2: Code Relationships

- 16.8% of exploitable comments are about code relationships.

Finding 2: Code Relationships

- 16.8% of exploitable comments are about code relationships.

```
1. /* See comment in struct sock definition to understand ... */
```

Finding 2: Code Relationships

- 16.8% of exploitable comments are about code relationships.

```
1. /* See comment in struct sock definition to understand ... */
2. /* Called from mmioctl_page_retire ... */
```

Finding 2: Code Relationships

- 16.8% of exploitable comments are about code relationships.

```
1. /* See comment in struct sock definition to understand ... */
2. /* Called from mmioctl_page_retire ... */
3. /* Call scsi_free before mem_free since ... */
```

Finding 2: Code Relationships

- 16.8% of exploitable comments are about code relationships.

```
1. /* See comment in struct sock definition to understand ... */
2. /* Called from mmioctl_page_retire ... */
3. /* Call scsi_free before mem_free since ... */
```

```
/* WARNING: If you change any of these defines, make sure
to change ... */
```

Finding 2: Code Relationships

- 16.8% of exploitable comments are about code relationships.

```
1. /* See comment in struct sock definition to understand ... */
2. /* Called from mmioctl_page_retire ... */
3. /* Call scsi_free before mem_free since ... */
```

```
/* WARNING: If you change any of these defines, make sure
to change ... */
```

- Exploit such comments to provide better navigation capabilities
- Inspire techniques to express code relationships and evolution

Finding 3: Locking

- 4.7% of the non-trivial comments are synchronization/lock related.

```
1. /* Caller must hold instance lock! */
   static int reset_hardware(…) {…}
```

Finding 3: Locking

- 4.7% of the non-trivial comments are synchronization/lock related.

1. /* **Caller must hold instance lock!** */
   static int **reset_hardware**(…) {…}

2. /* Locking key to struct socket:
   * (a) constant after allocation, no locking required.
   *
   * ...
   * **(h) locked by global mutex so_global_mtx.** ... */
   struct socket { ...
   short so_type; /* (a) generic type, see socket.h */
   ...
   so_gen_t so_gencnt; /* **(h) generation count** */ ...
   }

Finding 3: Locking

- 4.7% of the non-trivial comments are synchronization/lock related.

1. /* **Caller must hold instance lock!** */
   static int **reset_hardware**(…) {…}

2. /* Locking key to struct socket:
   * **(a) constant after allocation, no locking required.**
   *
   * **(h) locked by global mutex so_global_mtx.** ... */
   struct socket { ...
   short so_type; /* **(a) generic type, see socket.h** */
   ...
   so_gen_t so_gencnt; /* **(h) generation count** */ ...
   }

Finding 3: Locking

- 4.7% of the non-trivial comments are synchronization/lock related.

1. /* **Caller must hold instance lock!** */
   static int **reset_hardware**(…) {…}

2. /* Locking key to struct socket:
   * **(a) constant after allocation, no locking required.**
   *
   * **(h) locked by global mutex so_global_mtx.** ... */
   struct socket { ...
   short so_type; /* **(a) generic type, see socket.h** */
   ...
   so_gen_t so_gencnt; /* **(h) generation count** */ ...
   }

- Design easy-to-use annotations to express lock-related concerns

Finding 4: Annotation Convertible

- At least 10.7% of the exploitable comments can be expressed by existing annotations languages.
  - Linux's Sparse, Microsoft's SAL, Sun's Lock_Lint, Splint, Deputy

opensolaris/intel/io/acpica/resources/rscalc.c:

```
/* … AmlBufferLength - Size of AmlBuffer … */
ACPI_STATUS AcpiRsGetListLength (
    UINT8                               *AmlBuffer,
    UINT32                              AmlBufferLength, ...)
```

Finding 4: Annotation Convertible

- At least 10.7% of the exploitable comments can be expressed by existing annotations languages.
  - Linux's Sparse, Microsoft's SAL, Sun's Lock_Lint, Splint, Deputy

```
opensolaris/intel/io/acpica/resources/rscalc.c:
/* … AmlBufferLength - Size of AmlBuffer … */
ACPI_STATUS AcpiRsGetListLength (
    UINT8                               *AmlBuffer,
    UINT32  __ecount(AmlBuffer)          AmlBufferLength, ...)
```

Finding 4: Annotation Convertible

- At least 10.7% of the exploitable comments can be expressed by existing annotations languages.
  - Linux's Sparse, Microsoft's SAL, Sun's Lock_Lint, Splint, Deputy

```
opensolaris/intel/io/acpica/resources/rscalc.c:
/* … AmlBufferLength - Size of AmlBuffer … */
ACPI_STATUS AcpiRsGetListLength (
    UINT8                               *AmlBuffer,
    UINT32  __ecount(AmlBuffer)          AmlBufferLength, ...)
```

- Automatically convert these comments into annotations

Open vs. Closed Source

Software						
Linux						
FreeBSD						
OpenSolaris						

Open vs. Closed Source

Software	Comments					
Linux	23.1%					
FreeBSD	25%					
OpenSolaris	29.7%					

Open vs. Closed Source

Software	Comments	Exploitable				
Linux	23.1%	55.7%				
FreeBSD	25%	51.7%				
OpenSolaris	29.7%	50.3%				

Open vs. Closed Source

Software	Comments	Exploitable	Integers	Relationship	Locking	Annotation Convertible
Linux	23.1%	55.7%	26.7%	19.5%	5.1%	8.2%
FreeBSD	25%	51.7%	21.5%	14.9%	5.5%	12.7%
OpenSolaris	29.7%	50.3%	17.6%	15.9%	3.4%	11.4%

Open vs. Closed Source

Software	Comments	Exploitable	Integers	Relationship	Locking	Annotation Convertible
Linux	23.1%	55.7%	26.7%	19.5%	5.1%	8.2%
FreeBSD	25%	51.7%	21.5%	14.9%	5.5%	12.7%
OpenSolaris	29.7%	50.3%	17.6%	15.9%	3.4%	11.4%

- OpenSolaris (started as closed software) exhibits similar characteristics from its open source counterparts.
  - Complement the results of previous study [SpinellisICSE'08]
  - Our findings are general across different OSs.

Outline

- Motivation
- Methodology & Taxonomies
- OS Comments: Findings and Implications
- Non-OS Comments: Similarities & Differences
- Related Work & Conclusions

Non-OS Comment Source

Software	Linux	FreeBSD	OpenSolaris	Mozilla	MySQL	Eclipse
Software Type	OS	OS	OS	Browser	DB Server	IDE
Language	C	C	C	C/C++	C/C++	Java
Lines of Code	5.2M	2.4M	3.7M	4.5M	1.2M	6.1M
Lines of Comments	1.2M	0.6M	1.1M	1.2M	0.3M	1.7M
Sample Size	350	350	350	350	350	350

- Randomly sampled 1050 comments

Non-OS Comment Findings

Software						
OS						
Non-OS						

- Mostly similar, but OS has more locking & integer comments
  - See our paper for other differences

Non-OS Comment Findings

Software	Comments					
OS	23.1-29.7%					
Non-OS	21.8-28.5%					

- Mostly similar, but OS has more locking & integer comments
  - See our paper for other differences

Non-OS Comment Findings

Software	Comments	Exploitable				
OS	23.1-29.7%	52.6±2.9%				
Non-OS	21.8-28.5%	57.5±2.9%				

- Mostly similar, but OS has more locking & integer comments
  - See our paper for other differences

Non-OS Comment Findings

Software	Comments	Exploitable	Relationship	Annotation Convertible	Locking	
OS	23.1-29.7%	52.6±2.9%	16.8%	10.7%	4.1%	
Non-OS	21.8-28.5%	57.5±2.9%	21.2%	13.7%	1.7%	

- Mostly similar, but OS has more locking & integer comments
  - See our paper for other differences

Non-OS Comment Findings

Software	Comments	Exploitable	Relationship	Annotation Convertible	Locking	Integers
OS	23.1-29.7%	52.6±2.9%	16.8%	10.7%	4.1%	22.1%
Non-OS	21.8-28.5%	57.5±2.9%	21.2%	13.7%	1.7%	6.3%

- Mostly similar, but OS has more locking & integer comments
  - See our paper for other differences

Non-OS Comment Findings

Software	Comments	Exploitable	Relationship	Annotation Convertible	Locking	Integers
OS	23.1-29.7%	52.6±2.9%	16.8%	10.7%	4.1%	22.1%
Non-OS	21.8-28.5%	57.5±2.9%	21.2%	13.7%	1.7%	6.3%

- Mostly similar, but OS has more locking & integer comments
  - See our paper for other differences
  - Exceptions are not always used.
  - Still use comments to explain the exception types

```
/* return 1 if ACK, 0 if NAK, -1 if error. */
static int slhci_transaction(...) { ... }
```

Related Work

- Comment studies

  [WoodfieldICSE'81], [Etzkorn'99], [Stamelos'02], [Warren'02], [YingMSR05], [Marin'05], [JiangMSR'06], [Fluri'07], [StoreyICSE'08], ...

  - Usefulness of comments for program understanding [WoodfieldICSE'81]
  - Impact of already commented code [Marin'05]
  - TODO comments [YingMSR05], [StoreyICSE'08]

Conclusions

- **Comments reveal interesting findings**, which guide the design of new tools and languages:
  - *Abusive use of integers*
  - *Lack of expressing power on code relationship and evolution*
  - *Many can be expressed by existing annotation languages*
- New taxonomies, tools & examined comments
  - Available at http://ece.uwaterloo.ca/~lintan/CComment

More findings and
implications in our paper