Program Obfuscation and Related Topics

Applications and Perspectives

Yury Lifshits

yura@logic.pdmi.ras.ru

Faculty of Mathematics and Mechanics
Saint-Petersburg State University

Spring 2005 – Intel

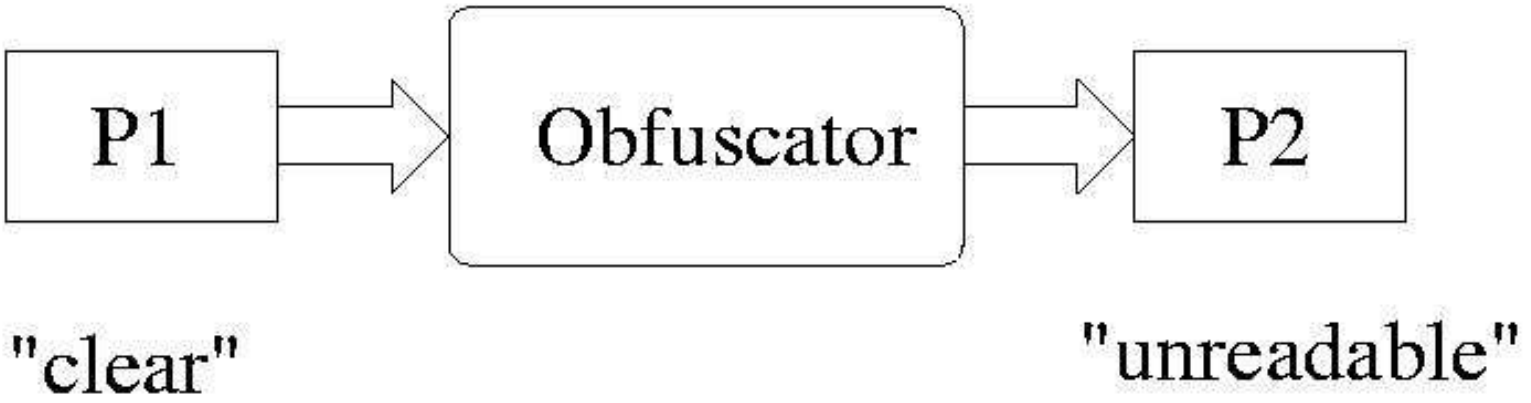
Outline

Talk Objectives

- ⇨ Short and general overview of applications
- ⇨ Short and general overview of results
- ⇨ Search for topic with common interest

Main Concept

So, what is Obfuscator?



- ⇨ Functionality preserving
- ⇨ Increase of code size, time & space requirements are restricted (usually by constant factor)
- ⇨ Obfuscated program is not readable (not understandable)

Topic Info [Propaganda]

Some facts:

- First mention — famous Diffie-Hellman paper (1976)
- More than 30 publications, several Ph.D. thesises
- More than 25 Java obfuscators
- International Contests (C, Perl, PostScript, Ruby)
- Famous Universities involved (Weizmann, Stanford, Princeton, MSU)
- Famous companies involved (Sun, Microsoft)

General Source-to-Source Obfuscators

Observations:

- ⇨ Long list of tricks (layout, data, control flow)
- ⇨ Commercial potential
- ⇨ No guaranteed security
- ⇨ Static analysis of obfuscated program is computationally hard
- ⇨ Arms race against hackers

Low-level Obfuscators

- ⇨ Making exact disassembling hard
- ⇨ Making exact decompilation hard

Same story — arms race with adversary:

New protection ⇨ new analysis ⇨ new protection . . .

Hardware-based program protection

Good recent news:

- ⇨ Some promising solutions are already presented (XOM, 2004)
- ⇨ Model: memory is accessible to adversary, processor is not
- ⇨ To achieve the best level of security program should be obfuscated in special way
- ⇨ Security analysis is not ready yet

RTL-model Obfuscation

New threat: bookmark insertion during chip manufacturing

Solution: chip obfuscation

Most appropriate level for obfuscation usage
[Zakharov, 2005] — RTL model of chip

Specific Protection

What type of attacks are we going to resist?

- Key's extraction
- Modification:
  - Add
  - Delete
  - Edit
  - Reuse
- Vulnerability search
- Bookmarks insertion
- Program state attack

More Applications

Other applications?

- ⇨ Mobile agents protection
- ⇨ White Box Encoding and DRM applications
- ⇨ Digital watermarks
- ⇨ Quality and protection analysis

Current Achievements

Most significant results to the moment:

- ⇨ A lot of obfuscators. Static analysis is now really hard
- ⇨ Definition of “ideal” security
- ⇨ Parameter hiding based on classical cryptography
- ⇨ Hardware solutions (in theory?)
- ⇨ Huge list of tricks

Our Contribution

What have our SPRINT Lab group already done?

- ⇨ Theoretical models for:
  - Program Slowdown
  - Function Sharing
  - Fully Encrypted Computation
  - Condition-protection
- ⇨ Hardware methods survey
- ⇨ Low-level obfuscation survey (+ some original tricks)

Theoretic View

Main questions for obfuscation theory:

- ⇨ Find all obfuscatable programs?
- ⇨ List of modelling examples which require obfuscation (benchmarks)?
- ⇨ Models for specific attacks?
- ⇨ Hardware models?
- ⇨ Quality of obfuscation?
- ⇨ Power of deobfuscation (program understanding)?

What Do We Learn Today?

- ⇨ Obfuscating transformations should make programs harder to understand, analyse and modify
- ⇨ There is a long list of threats based on program understanding
- ⇨ There is no universal protection
- ⇨ Hope for new protection methods

Thanks for your attention! Questions?

Not Covered by the Talk

Viruses
Obfuscation on interpretation level

For Further Reading

Yury Lifshits.

Lecture Notes on Program Obfuscation

`http://cs-seminar.spb.ru/`, “Reports” section

Yury Lifshits

Program Obfuscation. A survey

`http://logic.pdmi.ras.ru/~yura/of/survey1.pdf`