# A new definition of obfuscation

Philippe Beaucamps

ESAT - Rennes

Workshop on the Theory of Computer Viruses
May 04, 2006

# Outline

1. Introduction
2. Barak's formalization
   - Definition
   - Impossibility results
3. New formalization
   - Redefining obfuscators
   - $\tau$-obfuscators
4. Impossibility results
5. Deobfuscation and future work
   - Formalizing deobfuscation
   - Future work

# Outline

1. Introduction
2. Barak's formalization
   - Definition
   - Impossibility results
3. New formalization
   - Redefining obfuscators
   - $\tau$-obfuscators
4. Impossibility results
5. Deobfuscation and future work
   - Formalizing deobfuscation
   - Future work

# Introduction

- Obfuscation is the process of making programs unintelligible.
- Applications:
  - Protection of software;
  - Virology: obfuscation implies:
    - Polymorphism;
    - Delaying the analysis.
- Barak et al. formalized obfuscation in 2001.

# Outline

1. Introduction
2. Barak's formalization
   - Definition
   - Impossibility results
3. New formalization
   - Redefining obfuscators
   - $\tau$-obfuscators
4. Impossibility results
5. Deobfuscation and future work
   - Formalizing deobfuscation
   - Future work

# Barak's formalization

## Definition

*A probabilistic algorithm $O$ is an obfuscator if it satisfies the following properties:*

- *Functionality: $P$ and $O(P)$ compute the same function.*
- *Polynomial slowdown: the running time and the size of $O(P)$ are at most polynomially increased.*
- *"Virtual black box": for any adversary $A$, there exists a simulator $S$ such that:*

$$\forall P \in \Pi,\ \Pr\left[A\left(O(P)\right)\right] \approx \Pr\left[S^P\left(1^{|P|}\right)\right]$$

# Barak's impossibility result

Impossibility result:

**Theorem**

*Program obfuscators, as defined by Barak et al., do not exist.*

The impossibility result is kept for approximate obfuscators.

*Proof sketch*:
Decompose the input program $P$ in $P_1$ and $P_2$ and compute $P_2(P_1)$, where $P_2$ tests the functionality of $P_1$. Since the functionality is preserved, the result is preserved too. Yet any simulator with oracle access to $P$ is unable to compute this result.

# More impossibility results

Using Barak's results, we show that:

**Proposition**

*For any non trivial result $R$, there exists a program $P$ such that no obfuscator can hide $R(P)$.*

# Outline

1. Introduction
2. Barak's formalization
   - Definition
   - Impossibility results
3. New formalization
   - Redefining obfuscators
   - $\tau$-obfuscators
4. Impossibility results
5. Deobfuscation and future work
   - Formalizing deobfuscation
   - Future work

# Obfuscators

Any adversary analyzing an obfuscated program $O(P)$ necessarily has access to a program functionally equivalent to $P$.

We define an *oracle program* as a "perfectly obfuscated" program:

## Definition

*For any program $P \in \Pi$, an oracle program $\widetilde{P}$ is a program that is functionally equivalent to $P$ but can only be accessed in an oracle way.*

# Obfuscators

## Definition

*A probabilistic algorithm $O$ is an obfuscator if it satisfies the following properties:*

- *Functionality: $P$ and $O(P)$ compute the same function.*
- *Polynomial slowdown: the running time and the size of $O(P)$ are at most polynomially increased.*
- *"Virtual black box": for any adversary $A$, there exists a simulator $S$ such that:*

$$\forall P \in \Pi,\ \Pr\left[A\left(O(P)\right)\right] \approx \Pr\left[S\left(\widetilde{P}, 1^{|P|}\right)\right]$$

# $\tau$-obfuscators

*Idea*: define an obfuscation that remains effective at least on time $\tau$.
This kind of obfuscation would still be suitable for viruses.

# $\tau$-obfuscators

## Definition

*A probabilistic program $O$ is a $\tau$-obfuscator if it satisfies the following properties:*

- *Functionality: $P$ and $O(P)$ compute the same function.*
- *Polynomial slowdown: the running time and the size of $O(P)$ are at most polynomially increased.*
- *"Virtual black box": for any adversary $A$, there exists a simulator $S$ such that:*

$$\forall P \in \Pi,\ \Pr\left[A\left(O(P), 1^{\tau \times t(O(P))}\right)\right] \approx \Pr\left[S\left(\widetilde{P}, 1^{|P|}\right)\right]$$

# Outline

1. Introduction
2. Barak's formalization
   - Definition
   - Impossibility results
3. New formalization
   - Redefining obfuscators
   - $\tau$-obfuscators
4. Impossibility results
5. Deobfuscation and future work
   - Formalizing deobfuscation
   - Future work

# Impossibility results for obfuscators

**Theorem**

*There exists no program obfuscators, as they were previously redefined.*

The proof relies on the undecidability of the halting problem. It shows more specifically that an obfuscator has to keep the global algorithmic structure of a program.

The impossibility result still holds when restricting to programs that always halt.

# Extending the impossibility results to $\tau$-obfuscators

**Theorem**

*There exists no $\tau$-obfuscators.*

The proof is based on the same result, whose complexity is in $O(|O(P)|)$.

# Outline

1. Introduction
2. Barak's formalization
   - Definition
   - Impossibility results
3. New formalization
   - Redefining obfuscators
   - $\tau$-obfuscators
4. Impossibility results
5. Deobfuscation and future work
   - Formalizing deobfuscation
   - Future work

# Deobfuscation

## Definition

*A probabilistic program $D$ is a deobfuscator if it satisfies the following properties:*

- *Functionality: $P$ and $D(O(P))$ compute the same function.*
- *Intelligibility: for any obfuscator $O$ and result $R$ on programs of $\Pi$, there exists a result $T$ such that:*

$$t(T) \approx t(R) \text{ and } (\forall P \in \Pi,\ \Pr[T(D(O(P)))] \approx \Pr[R(P)])$$

A deobfuscator can be defined relatively to a given obfuscator.

# Deobfuscation

We show that:

**Proposition**

*There exists no deobfuscators, relatively to Barak's definition as to our definition of obfuscators.*

# Future work

- Defining obfuscation methods complying with the $\tau$-obfuscation requirements.
- Polymorphism of viruses : viruses deobfuscating and reobfuscating themselves when replicating.
- Existence of obfuscators requiring a deobfuscation in time at least $\tau$.

Thank your for your attention.