# Search-based Local Blackbox Deobfuscation: Understand, Improve and Mitigate


Grégoire Menguy – CEA LIST

Sébastien Bardin – CEA LIST

Richard Bonichon – TWEAG I/O

Cauim de Souza Lima – CEA LIST

# Speaker

**Grégoire MENGUY**

*PhD Student at CEA LIST*

*BINSEC Team* (https://binsec.github.io/)

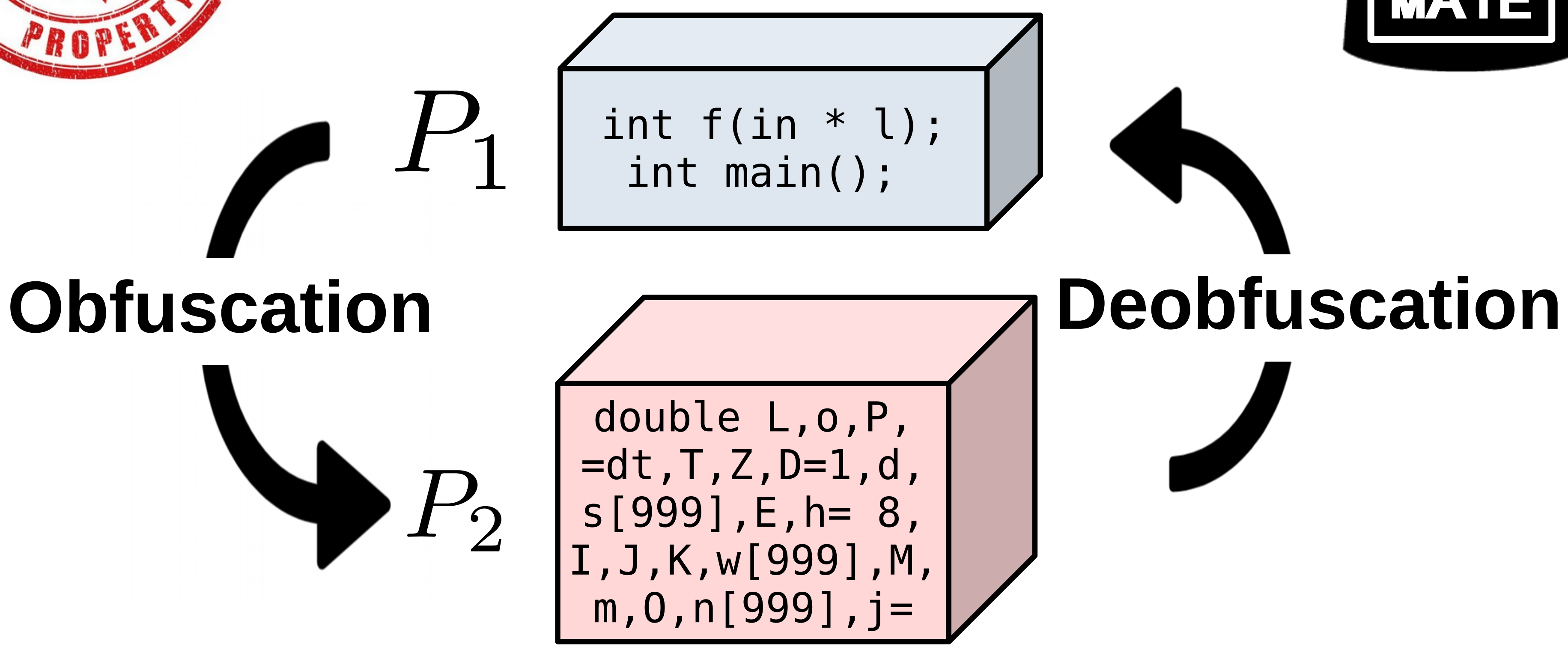
https://www.linkedin.com/in/gregoire-menguy/

@grmenguy

# Context


INTELLECTUAL
PATENTED
PROPERTY


MATE


$P_1$
int f(in * l);
int main();
Obfuscation
Deobfuscation
$P_2$
double L,o,P,
=dt,T,Z,D=1,d,
s[999],E,h= 8,
I,J,K,w[999],M,
m,O,n[999],j=

# Deobfuscation

**Protecting Software through Obfuscation: Can It Keep Pace with Progress in Code Analysis?**

SEBASTIAN SCHRITTWIESER, St. Pölten University of Applied Sciences, Austria
STEFAN KATZENBEISSER, Technische Universität Darmstadt, Germany
JOHANNES KINDER, Royal Holloway, University of London, United Kingdom
GEORG MERZDOVNIK and EDGAR WEIPPL, SBA Research, Vienna, Austria

**A Generic Approach to Automatic Deobfuscation of Executable Code**

Babak Yadegari Brian Johannesmeyer Benjamin Whitely Saumya Debray
Department of Computer Science
The University of Arizona
Tucson, AZ 85721
{babaky, bjohannesmeyer, whitely, debray}@cs.arizona.edu

**Symbolic deobfuscation: from virtualized code back to the original***

Jonathan Salwan[1], Sébastien Bardin[2], and Marie-Laure Potet[3]

**Backward-Bounded DSE: Targeting Infeasibility Questions on Obfuscated Codes***

Sébastien Bardin
CEA, LIST,
91191 Gif-Sur-Yvette, France
sebastien.bardint@cea.fr

Robin David
CEA, LIST,
91191 Gif-Sur-Yvette, France
robin.david@cea.fr

Jean-Yves Marion
Université de Lorraine,
CNRS and Inria, LORIA, France
jean-yves.marion@loria.fr

BINSEC

TRITON
Dynamic Binary Analysis

MIASM

# Deobfuscation

Protecting Software through Obfuscation: Can It Keep Pace with Progress in Code Analysis?

SEBASTIAN SCHRITTWIESER, St. Pölten University of Applied Sciences, Austria
STEFAN KATZENBEISSER, Technische Universität Darmstadt, Germany
JOHANNES KINDER, Royal Holloway, University of London, United Kingdom
GEORG MERZDOVNIK and EDGA

Backward-Bounded DSE: Targeting Infeasibility Questions on Obfuscated Codes*

Sébastien Bardin
CEA, LIST,

Robin David
CEA, LIST,
France

Jean-Yves Marion
Université de Lorraine,
CNRS and Inria, LORIA, France
jean-yves.marion@loria.fr

A Generic Approach to Autom

Babak Yadegari Brian Johannesn
Department
The Uni
{babaky, bjohan

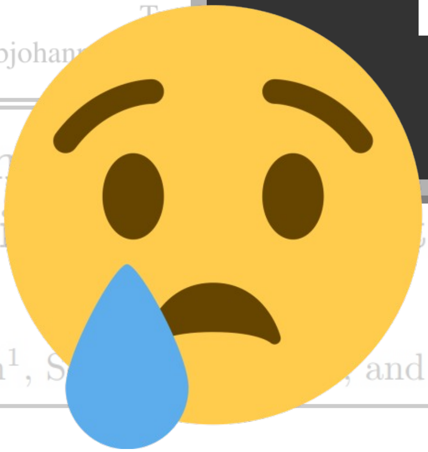
Sy
from virtual o the original*

Jonathan Salwan[1], S , and Marie-Laure Potet[3]

SEC
O N
Dynamic Binary Analysis

MIASM

**Whitebox deobfuscation is highly efficient**



# Whitebox Deobfuscation

## But efficient countermeasures emerge

**Information Hiding in Software with Mixed Boolean-Arithmetic Transforms**

Yongxin Zhou, Alec Main, Yuan X. Gu, and Harold Johnson

Cloakware Inc., USA
{yongxin.zhou,alec.main,yuan.gu,harold.johnson}@cloakware.com

**How to Kill Symbolic Deobfuscation for Free**
**(or: Unleashing the Potential of Path-Oriented Protections)**

Mathilde Ollivier
CEA, LIST,
Paris-Saclay, France
mathilde.ollivier2@cea.fr

Sébastien Bardin
CEA, LIST,
Paris-Saclay, France
sebastien.bardin@cea.fr

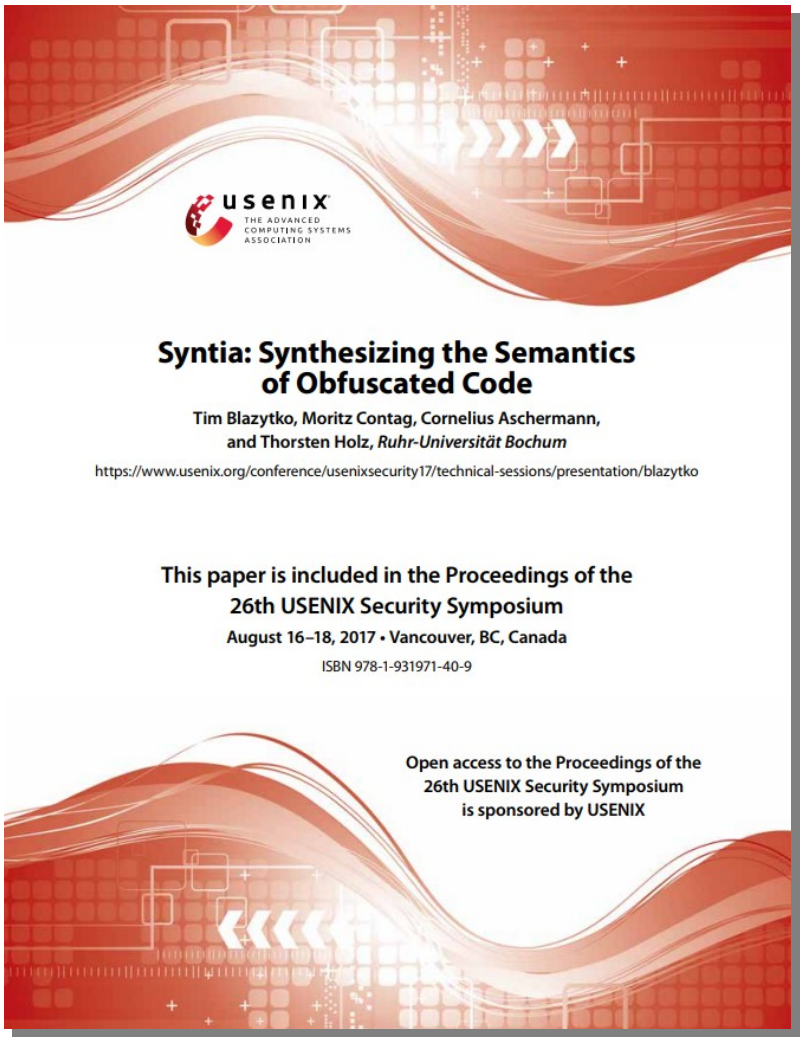
Richard Bonichon
CEA, LIST,
Paris-Saclay, France
richard.bonichon@cea.fr

Jean-Yves Marion
Université de Lorraine, CNRS, LORIA
Nancy, France
Jean-Yves.Marion@loria.fr

**Probabilistic Obfuscation through Covert Channels**

Jon Stephens Babak Yadegari Christian Collberg Saumya Debray Carlos Scheidegger
*Department of Computer Science*
*The University of Arizona*
*Tucson, AZ 85721, USA*
*Email: {stephensj2, babaky, collberg, debray, cscheid}@cs.arizona.edu*

# New threat: Blackbox Deobfuscation



usenix
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

**Syntia: Synthesizing the Semantics of Obfuscated Code**

Tim Blazytko, Moritz Contag, Cornelius Aschermann, and Thorsten Holz, *Ruhr-Universität Bochum*

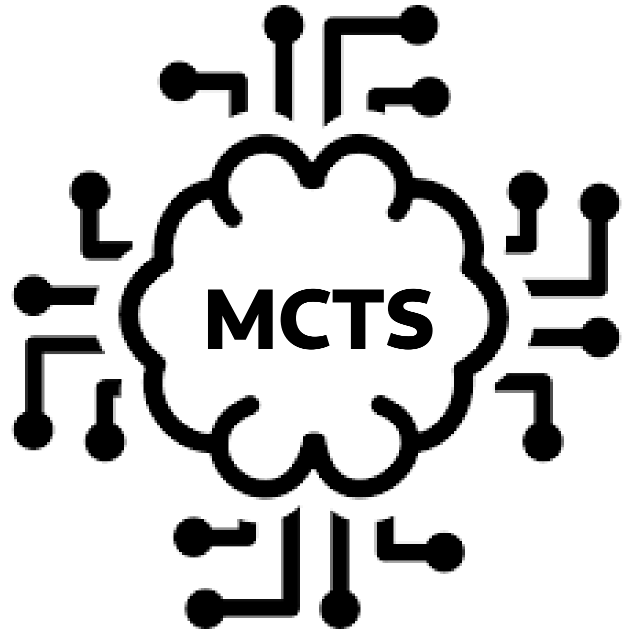
https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/blazytko

This paper is included in the Proceedings of the 26th USENIX Security Symposium

August 16–18, 2017 • Vancouver, BC, Canada

ISBN 978-1-931971-40-9

Open access to the Proceedings of the 26th USENIX Security Symposium is sponsored by USENIX




**Bypasses whitebox methods limitations**

# Open questions

## Understand

- Strengths ?
- Weaknesses ?
- Why ?

## Improve

- Why MCTS ?
- Can be improved?
- Impacted by SoA protections?

## Mitigate

- How to protect ?

# Contributions

## Understand

- Propose missing formalization
- Refine Syntia evaluation: new strengths and weaknesses
- Show and explain why MCTS is not appropriate

Partial evaluation based search is not appropriate

## Improve

- S-metaheuristics > MCTS
- Implement our approach: Xyntia
- Evaluation of Xyntia

Relies on S-metaheuristics

## Mitigate

- Propose 2 protections
- Evaluate them against Xyntia and Syntia

Increase semantic complexity

# The talk in a nutshell

I. Blackbox deobfuscation : what’s that ?

II. Deepen understanding

III. Improve state-of-the art

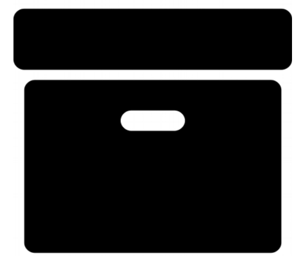
IV. Mitigate

# Blackbox deobfuscation : what’s that ?

# Blackbox deobfuscation

## 1) Sample

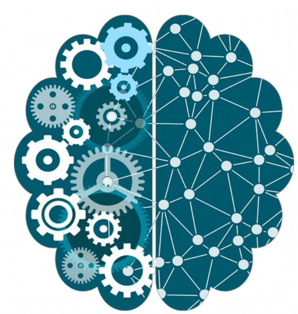
(x = 1, y = 2)
(x = 2, y = 5)
(x = 0, y = 6)
...



-1
-3
-6
...

## 2) Learn

(x = 1, y = 2) → -1
(x = 2, y = 5) → -3
(x = 0, y = 6) → -6
...

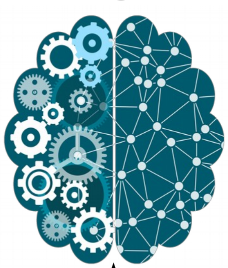


$$x - y$$

# Learning engine

$U + (x-1)$ $\quad x + y$ $\quad x - U$

$U \times U$ $\quad (x-y) \times (y-1)$



1 $\longrightarrow expr$ 2 $\longrightarrow \Delta \left( \begin{array}{l} expr(x=1, y=2), -1 \\ expr(x=2, y=5), -3 \\ expr(x=0, y=6), -6 \\ \quad\quad\quad\quad \cdots \end{array} \right)$

3

**Expression Grammar**

U := U + U |  U – U | U * U …
| x | y | 1

# Why blackbox?

**Given** a language *L* and an expression *"e"* in *L*

| Syntactic complexity | Semantic complexity |
|---|---|
| Size of the the expression *"e"* | Size of the smallest expression in *L* equivalent to *"e"* |

**Example**

$x - y$ is syntactically simpler than $(x \vee -2y) \times 2 - (x \oplus -2y) + y$

**but** they share the same semantic complexity (being equivalent)

# Why blackbox ?

**Given** a language *L* and an expression *“e”* in *L*

| Syntactic complexity | Semantic complexity |
|---|---|
| Size of the the expression *“e”* | Size of the smallest expression in *L* equivalent to *“e”* |

**Example**

$x - y$ is syntactically simpler than $(x \vee -2y) \times 2 - (x \oplus -2y) + y$

**but** they share the same semantic complexity (being equivalent)

Obfuscation increase syntactic complexity

**→ No impact on blackbox methods**

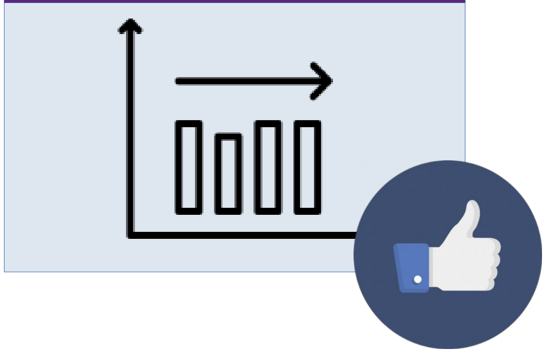
# Understand

# Zoom on SoA: Syntia

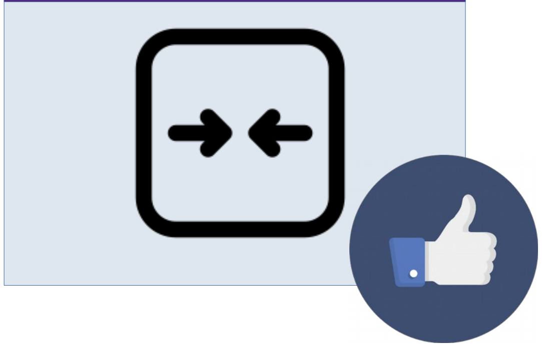
- **Dig into Syntia and deepen its evaluation:**
  - RQ1: stability of Syntia
  - **RQ2: efficiency of Syntia**
  - RQ3: Impact of operators set

# Syntia: new results

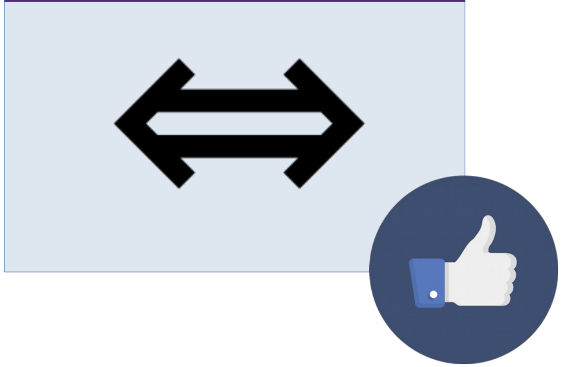
**Stable**



**Quality**



**Correctness**



# Syntia: new results

### Stable

### Quality

### Correctness

### Speed

### Robustness

# Experimental design

**B1 (Syntia)**

- 500 expressions
- Use up to 3 inputs
- **redundancy**
- Unbalanced w.r.t. type

**B2 (ours)**

- 1110 expressions
- Use 2 - 6 inputs
- **No redundancy**
- Balanced w.r.t. type

| | Type | | | # Inputs | | | | |
|---|---|---|---|---|---|---|---|---|
| | Bool. | Arith. | MBA | 2 | 3 | 4 | 5 | 6 |
| #Expr. | 370 | 370 | 370 | 150 | 600 | 180 | 90 | 90 |

**Table 1: Distribution of samples in benchmark B2**

# Evaluation of Syntia

## B1 (Syntia)

- With a 1 s/expr. timeout : 41 % of success rate
- **With a 60 s/expr. timeout : 74 % of success rate**
- With a 600 s/expr. timeout : 88 % of success rate

## B2 (Ours)

Table 2: Syntia depending on the timeout per expression (B2)

| | 1s | 60s | 600s |
|---|---|---|---|
| Succ. Rate | 16.5% | 34.5% | 42.3% |

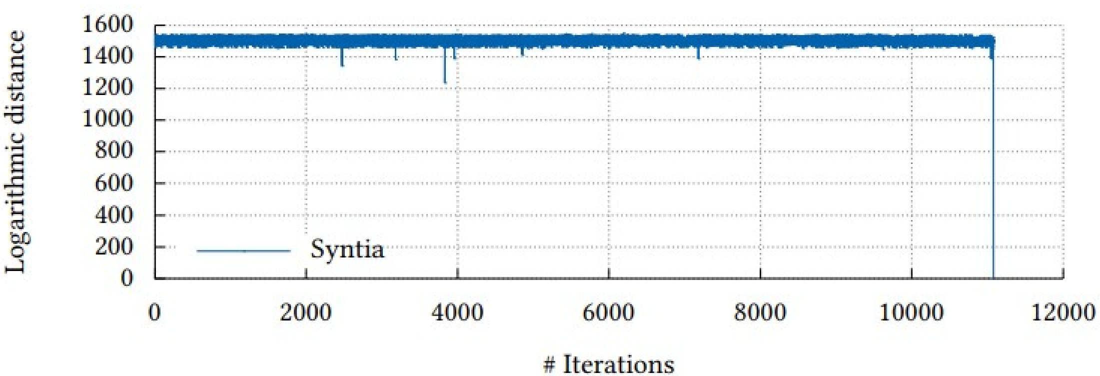
# Why ? A Summary

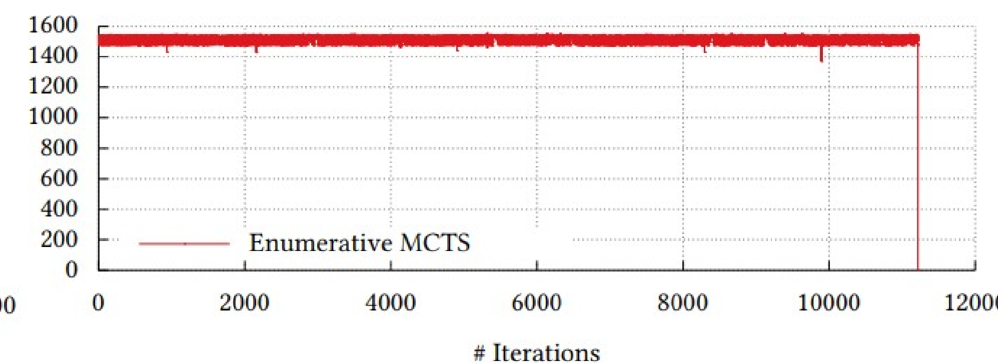
- **Syntia manipulates non terminal expressions** $U - V$

- **Scoring of non terminal expressions can be misleading**

$$\boxed{a - b} \qquad U - V \rightsquigarrow \begin{cases} a - b \\ b - 1 \\ 1 - 1 \end{cases}$$

- **Syntia (i.e. MCTS) = "almost BFS"**



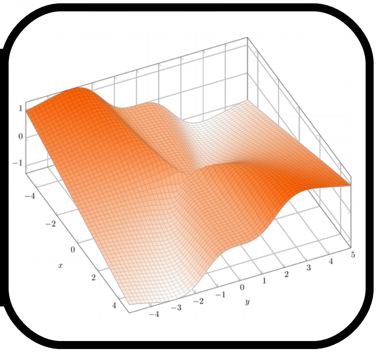



# Improve

# Blackbox deobf., an optimization pb

**Syntia** sees blackbox deobfuscation as a
**single player game**

We propose to see it as an
**optimization problem**



**Goal :** find $\underbrace{s^*}_{\text{an expr.}}$ s.t. $\underbrace{f}_{\Delta}(s^*) \leq f(s), \forall s \in S$

# New prototype: Xyntia

Xyntia

**S-metaheuristics**

**Terminal expressions only**

**Can choose between:**

→ Hill Climbing
→ Simulated annealing
→ Metropolis Hasting
→ **Iterated Local Search**

**MCTS**

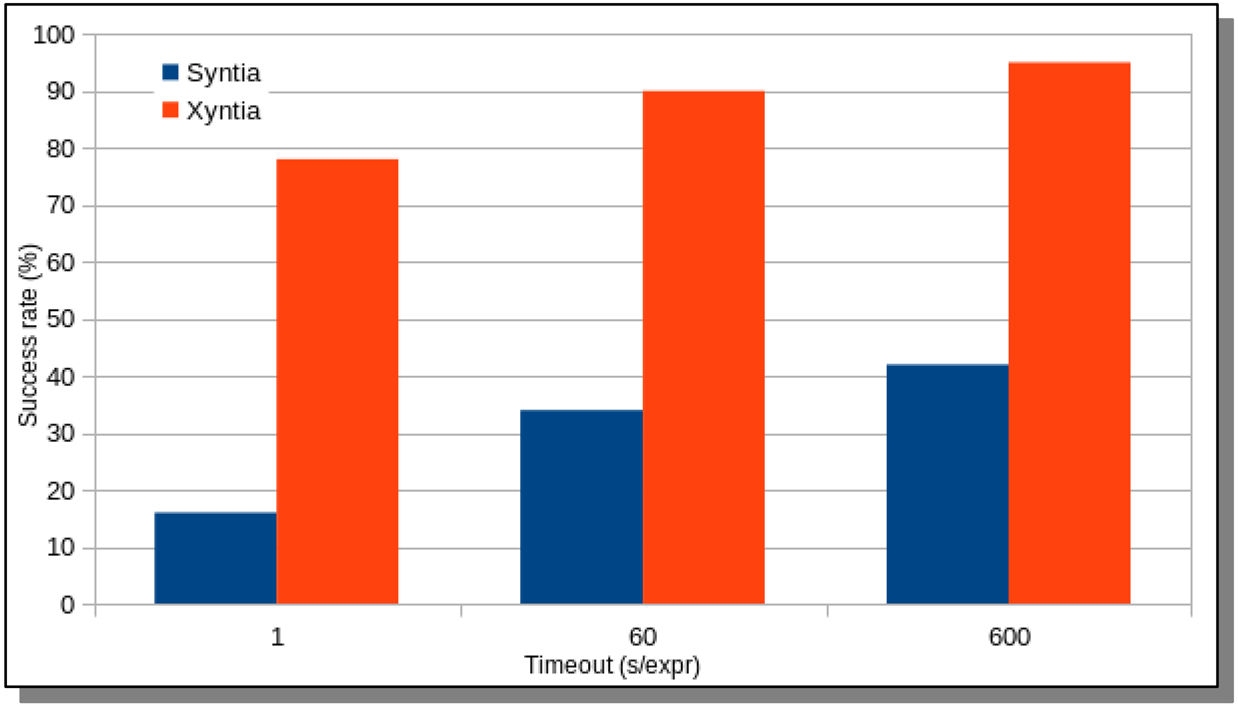
# Xyntia vs Syntia

## B1 (Syntia)

- **100 %** success rate in **1 s/expr.**

Syntia: 41% in 1 s/expr.

## B2 (Ours)



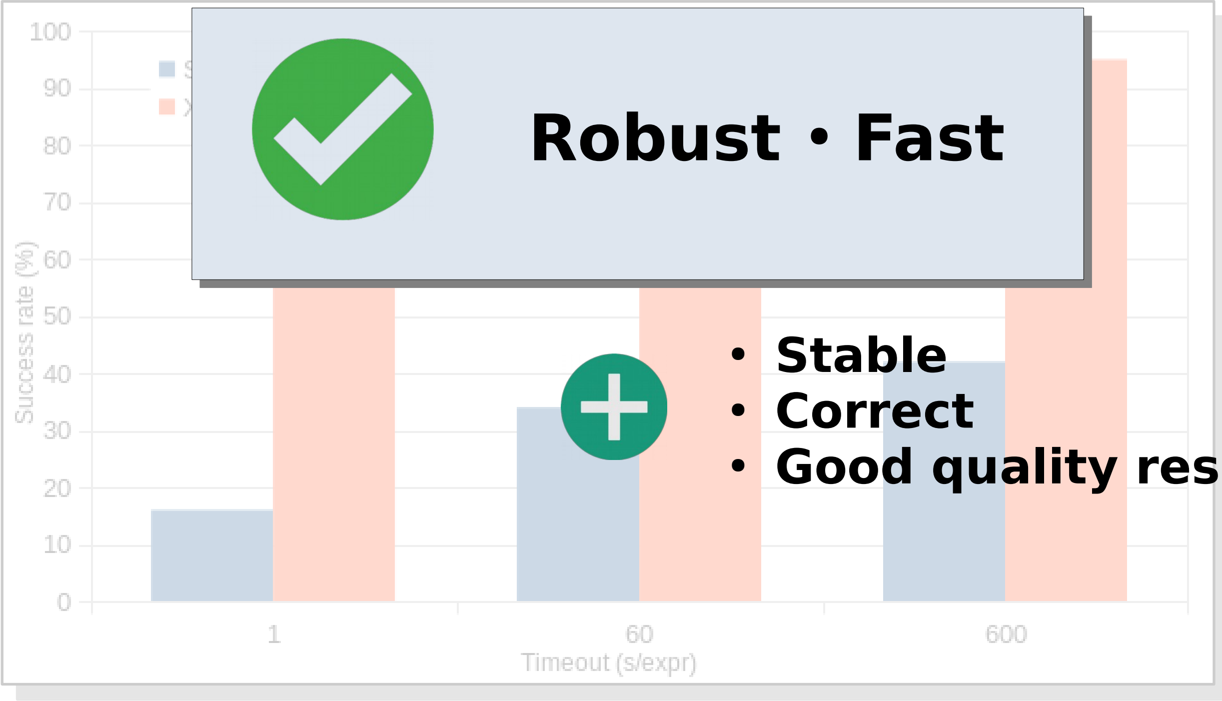
# Xyntia vs Syntia

## B1 (Syntia)

- **100 %** success rate in **1 s/expr.**

Syntia: 41% in 1 s/expr.

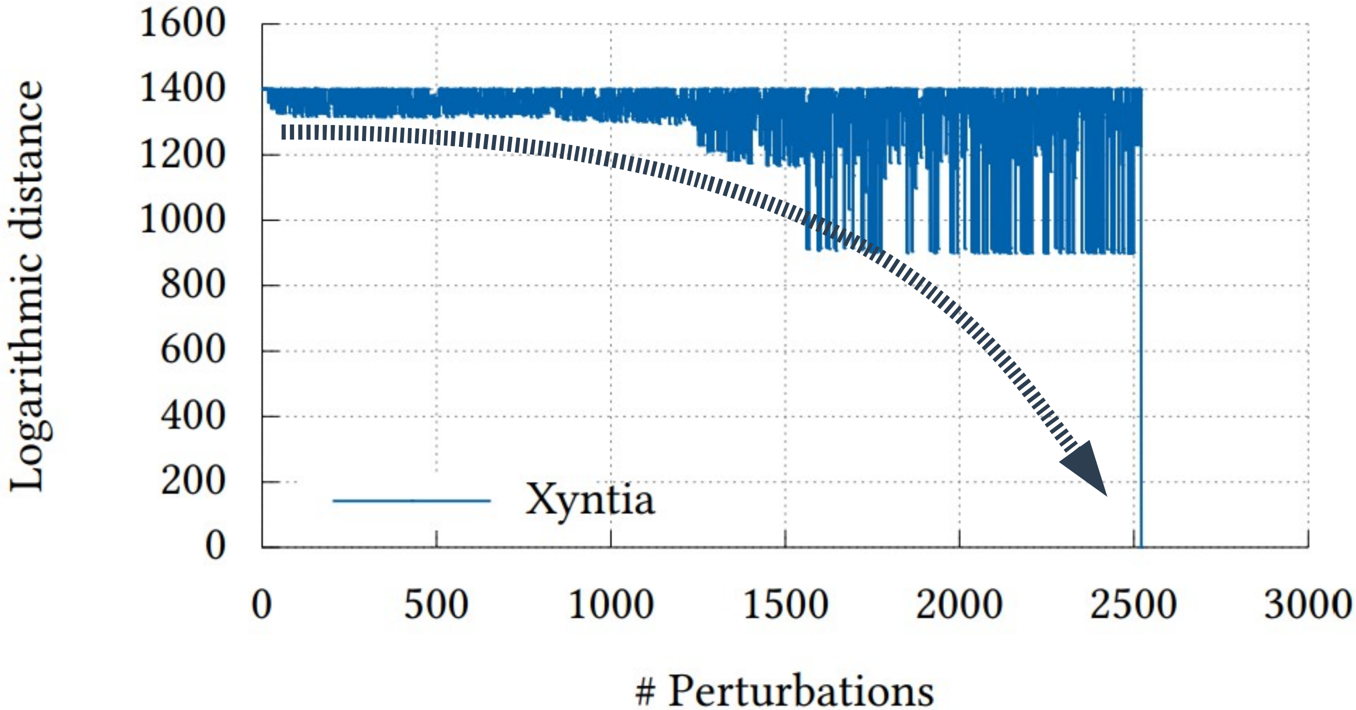
## B2 (Ours)




**Robust • Fast**

- **Stable**
- **Correct**
- **Good quality results**

# Is Xyntia well guided ?




**Xyntia is guided by the objective function**

# Other experiments

- Xyntia against QSynth
- Xyntia against "compiler like simplifications"
- Xyntia against program synthesizer **CVC4**
- Xyntia against superoptimizer **STOKE**

404

Not Found

The resource requested could not be found on this server!

- Use-cases:
  - State-of-the-art protections
  - **VM-based obfuscation**

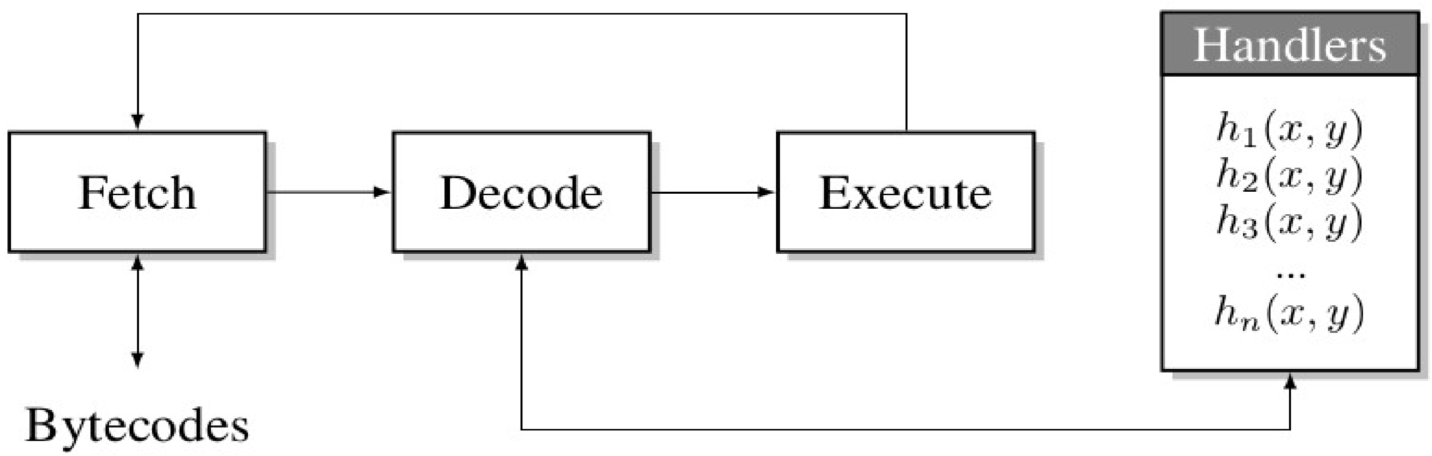
# What's next?


PROTECTIONS ARE BROKEN!


DON'T WORRY...
I HAVE A PLAN
memegenerator.net

# Mitigate

# Context : Virtualization




**Proved to be sensitive to blackbox deobfuscation**

Themida®
ADVANCED WINDOWS SOFTWARE PROTECTION

# Why VM-based obf. is vulnerable ?

- Handlers are too semantically simple:
  → e.g. $+, -, \times, \wedge, \vee$

- Obfuscation increases syntactic complexity
  → **Blackbox deobf. is not impacted**

We need to move ...

**From syntactic to semantic complexity**

# Semantically complex expressions

- **Goal:**
  - Increase the semantic complexity of each handlers
  - Keep a Turing complete set of handlers

- **Example:**

$$
\begin{array}{rrcl}
 & h_0 & = & (x+y) + -((a-x^2) - (xy)) \\
+ & h_1 & = & (a-x^2) - xy + (-(y-(a \wedge x)) \times (y \otimes x)) \\
+ & h_2 & = & (y-(a \wedge x)) \times (y \otimes x) \\
\hline
 & h & = & x+y
\end{array}
$$

# Merged handlers

- **Goal:**
  - Increase semantic + sampling complexity

- **Example:**

$$h_1(x, y) = x + y \quad \text{and} \quad h_2(x, y) = x \wedge y$$

$$\rightarrow \quad h(x, y, c) = \text{if } (c = cst) \text{ then } h_1(x, y) \text{ else } h_2(x, y)$$

- **Need to hide conditionals:**

```
int32_t h(int32_t a, int32_t b, int32_t c) {
    // if (c == cst) then h1(a,b,c) else h2(a,b,c);
    int32_t res = c - cst ;
    int32_t s = res >> 31;
    res = (-((res ^ s) -s) >> 31) & 1;
    return h1(a, b, c)*(1 - res) + res*h2(a, b, c);
}
```

# Semantically complex handlers: results

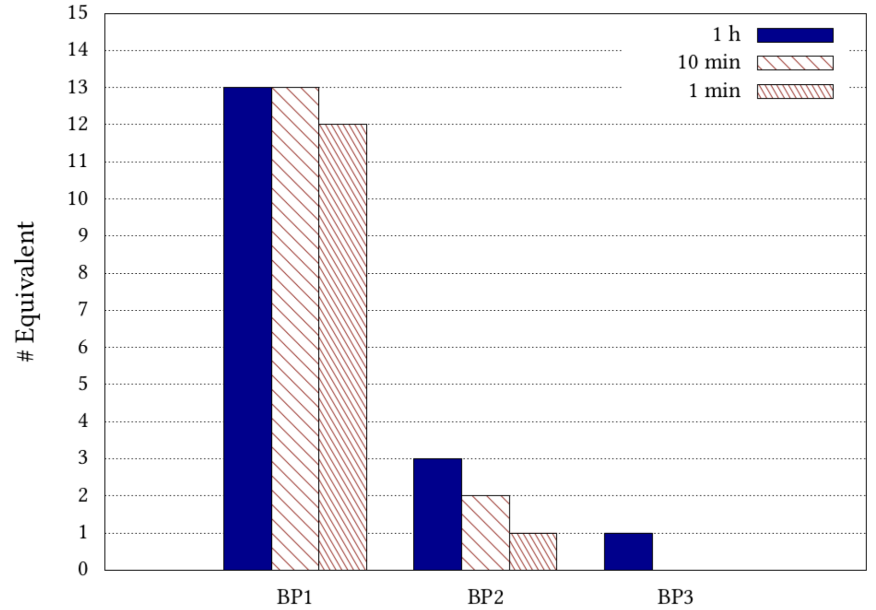



**Figure 8: Xyntia on BP1,2, 3**

**More results:**

- Syntia with 12h/exprs. → 1/15 on BP1

# Merged handlers: results

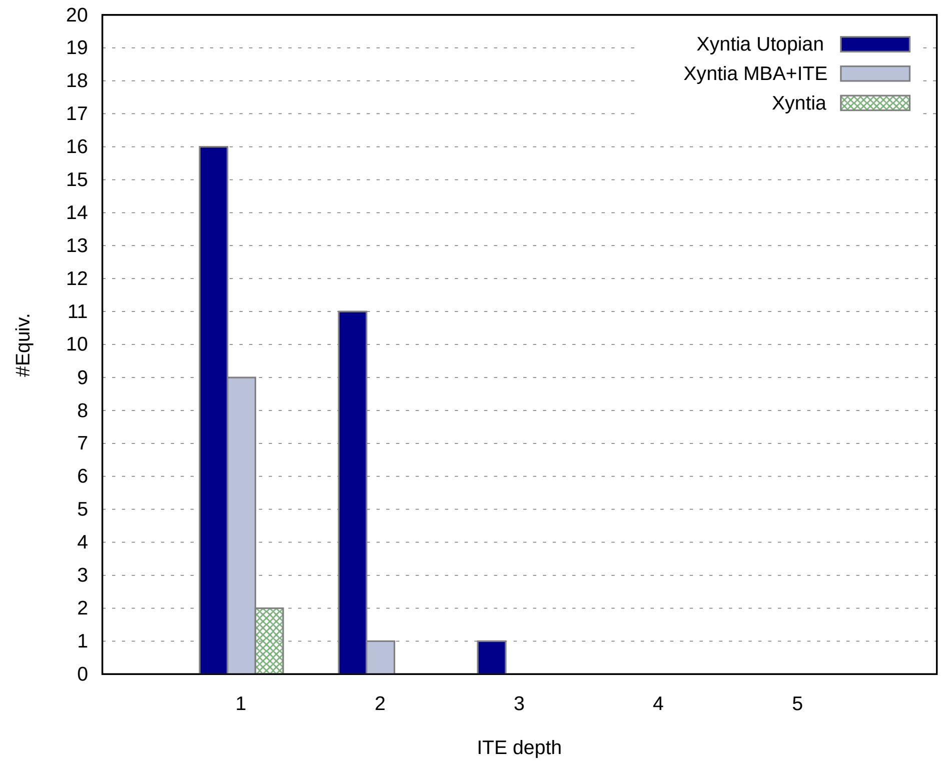



**Figure 10: Merged handlers: Xyntia (timeout=60s)**

## More results:

- Syntia finds nothing for ≥ 2 nested ITE

# Conclusion

**MCTS is not appropriate for blackbox deobfuscation**
→ Search space too unstable
→ Estimation of non terminal expressions pertinence is misleading

**S-metaheuristics yields a significant improvement**
→ More robust
→ Much Faster

**Moving for syntactic to semantic complexity**
→ 2 efficient methods to protect against blackbox deobfuscation

# Thank you for your attention