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
Grégoire MENGUY

PhD Student at CEA LIST

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

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

@grmenguy
Obfuscation

\[
\begin{align*}
\text{int } f(\text{in }* \text{l}); \\
\text{int } \text{main}();
\end{align*}
\]

Deobfuscation

\[
\begin{align*}
\text{double } L, o, P, \\
=dt, T, Z, D=1, d, \\
s[999], E, h= 8, \\
I, J, K, w[999], M, \\
m, 0, n[999], j=
\end{align*}
\]
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 Salwan¹, Sébastien Bardin², and Marie-Laure Potet³

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

Sébastien Bardin
CEA, LIST, 91191 Gif-Sur-Yvette, France
sebastien.bardin@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
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

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

Sébastien Bardin
CEA, LIST,
Paris-Saclay, France
sebastien.bardin@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: {stephens|babak|collberg|debray|cscheid}@cs.arizona.edu
New threat: Blackbox Deobfuscation

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**

<table>
<thead>
<tr>
<th>Understand</th>
<th>Improve</th>
<th>Mitigate</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Propose missing formalization</td>
<td>● S-metaheuristics &gt; MCTS</td>
<td>● Propose 2 protections</td>
</tr>
<tr>
<td>● Refine Syntia evaluation: new strengths and weaknesses</td>
<td>● Implement our approach: Xyntia</td>
<td>● Evaluate them against Xyntia and Syntia</td>
</tr>
<tr>
<td>● Show and explain why MCTS is not appropriate</td>
<td>● Evaluation of Xyntia</td>
<td></td>
</tr>
</tbody>
</table>

Partial evaluation based search is not appropriate

Relies on S-metaheuristics

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

\[(x = 1, y = 2) \rightarrow -1\]
\[(x = 2, y = 5) \rightarrow -3\]
\[(x = 0, y = 6) \rightarrow -6\]
\[\ldots\]

2) Learn

\[(x = 1, y = 2) \rightarrow -1\]
\[(x = 2, y = 5) \rightarrow -3\]
\[(x = 0, y = 6) \rightarrow -6\]
\[\ldots\]

\[x - y\]
Learning engine

Expression Grammar

\[ U := U + U | U - U | U * U \ldots | x | y | 1 \]

\[ U + (x - 1) \quad x + y \quad x - U \]

\[ U \times U \quad (x - y) \times (y - 1) \]
Why blackbox?

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

<table>
<thead>
<tr>
<th>Syntactic complexity</th>
<th>Semantic complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of the expression “$e$”</td>
<td>Size of the smallest expression in $L$ equivalent to “$e$”</td>
</tr>
</tbody>
</table>

**Example**

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

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

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

<table>
<thead>
<tr>
<th>Syntactic complexity</th>
<th>Semantic complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of the expression “$e$”</td>
<td>Size of the smallest expression in $L$ equivalent to “$e$”</td>
</tr>
</tbody>
</table>

**Example**

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

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

Obfuscation increases 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: New results are stable over time.
- Quality: The quality of the results is stable.
- Correctness: The correctness of the results is stable.
- Speed: Speed is not stable.
- Robustness: Robustness is not stable.

Grégoire Menguy | ACM CCS 2021
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

<table>
<thead>
<tr>
<th>Type</th>
<th># Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bool.</td>
<td>2</td>
</tr>
<tr>
<td>Arith.</td>
<td>3</td>
</tr>
<tr>
<td>MBA</td>
<td>4, 5, 6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#Expr.</th>
<th>370</th>
<th>370</th>
<th>370</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>150</td>
<td>600</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>90</td>
<td></td>
</tr>
</tbody>
</table>

**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)

<table>
<thead>
<tr>
<th></th>
<th>1s</th>
<th>60s</th>
<th>600s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Succ. Rate</td>
<td>16.5%</td>
<td>34.5%</td>
<td>42.3%</td>
</tr>
</tbody>
</table>
Why? A Summary

- Syntia manipulates non terminal expressions $U - V$

- Scoring of non terminal expressions can be misleading

- 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 $s^*$ s.t. $\triangleleft f(s^*) \leq f(s), \forall s \in S$

Grégoire Menguy | ACM CCS 2021
New prototype: Xyntia

S-metaheuristics

Can choose between:
 → Hill Climbing
 → Simulated annealing
 → Metropolis Hasting
 → Iterated Local Search

Terminal expressions only

MCTS
Xyntia vs Syntia

B1 (Syntia)

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

B2 (Ours)

Syntia: 41% in 1 s/expr.
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

Use-cases:
- State-of-the-art protections
- VM-based obfuscation
What’s next?

PROTECTIONS ARE BROKEN!

DON'T WORRY...

I HAVE A PLAN

Grégoire Menguy | ACM CCS 2021
Mitigate
Context: Virtualization

Proved to be sensitive to blackbox deobfuscation

![Diagram with flowchart: Fetch → Decode → Execute → Bytecodes → Handlers](image-url)
Why VM-based obf. is vulnerable?

- Handlers are too semantically simple:
  → e.g.  +, −, ×, ∧, ∨
- 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:**

\[
\begin{align*}
  h_0 &= (x + y) + -((a - x^2) - (xy)) \\
  h_1 &= (a - x^2) - xy + (-y - (a \land x)) \times (y \otimes x)) \\
  h_2 &= (y - (a \land x)) \times (y \otimes x) \\
  h &= x + y
\end{align*}
\]
Merged handlers

• Goal:
  - Increase semantic + sampling complexity

• Example:
  \[ h_1(x, y) = x + y \quad \text{and} \quad h_2(x, y) = x \land y \]
  \[ \Rightarrow h(x, y, c) = \text{if } (c = \text{cst}) \text{ then } h_1(x, y) \text{ else } h_2(x, y) \]

• Need to hide conditionals:

```c
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

More results:

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

Figure 8: Xyntia on BP1, 2, 3
Merged handlers: results

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

More results:
- Syntia finds nothing for $\geq 2$ 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