Neural Reverse Engineering of Stripped Binaries

Yaniv David, Uri Alon, Eran Yahav
Technion, Israel
Reverse Engineering (RE) Binaries: What, Why & How?
RE – What & Why?

Malware?

Bug? find & fix it
RE – How? Disassemblers

call getaddrinfo
mov rax, [rbp-30h]
mov rdx, [rbp-50h]
mov rdx, cs:688588d
mov [rax], rdx
mov rax, [rbp-30h]
mov rdx, [rbp-580h]
mov [rax+8], rdx
mov rax, [rbp-30h]
call strerror
sub rdx, rax
idiv [rbp-28h]
call setsockopt
mov rdx, [rax]
mov eax, [rbx+40h]
cdqe
RE – How? Disassemblers

No Names

No Types

call getaddrinfo
mov rax, [rbp-30h]
mov rdx, [rbp-50h]
mov rdx, cs:688588d
mov [rax], rdx
mov rax, [rbp-30h]
mov rdx, [rbp-580h]
mov [rax+8], rdx
mov rax, [rbp-30h]
call strerror
sub rdx, rax
idiv [rbp-28h]
call setsockopt
mov rdx, [rbp-40h]
mov
### Function name

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>sub_42EFA7</td>
</tr>
<tr>
<td>sub_42F08D</td>
</tr>
<tr>
<td>sub_42F0C0</td>
</tr>
<tr>
<td>sub_42F5EA</td>
</tr>
<tr>
<td>sub_42F680</td>
</tr>
<tr>
<td>sub_42F717</td>
</tr>
<tr>
<td>sub_42F7F8</td>
</tr>
<tr>
<td>sub_42F809</td>
</tr>
<tr>
<td>sub_42F863</td>
</tr>
<tr>
<td>sub_42F878</td>
</tr>
<tr>
<td>sub_42F8CC</td>
</tr>
<tr>
<td>sub_42F901</td>
</tr>
<tr>
<td>sub_42FAC8</td>
</tr>
<tr>
<td>sub_42FDE3</td>
</tr>
<tr>
<td>sub_42FE25</td>
</tr>
<tr>
<td>sub_42FF77</td>
</tr>
<tr>
<td>sub_430094</td>
</tr>
<tr>
<td>sub_4305BE</td>
</tr>
<tr>
<td>sub_4307C1</td>
</tr>
<tr>
<td>sub_430873</td>
</tr>
<tr>
<td>sub_430A13</td>
</tr>
<tr>
<td>sub_430A88</td>
</tr>
</tbody>
</table>

---

```asm
; Attributes: bp-based frame

sub_436B9D proc near
ptr= qword ptr -18h
var_8= qword ptr -8

push  rbp
mov   rbp, rsp
sub   rsp, 20h
mov   [rbp+ptr], rdi
jmp   short loc_436C0A

loc_436C0A:
cmp   [rbp+ptr], 0
jnz   short loc_436BAB

loc_436BAB:
mov   rax, [rbp+ptr]
mov   rax, [rax+28h]
mov   [rbp+var_8], rax
mov   rax, [rbp+ptr]
mov   rax, [rax]
test  rax, rax
jz    short loc_436BD2

nop
leave
retn
sub_436B9D endp
```

---

```asm
mov   rax, [rbp+ptr]
mov   rax, [rax]
mov   rdi, rax
call  sub_438DA9
```
RE – How? Modern Disassemblers

Where to start?
Progress in Other Domains
Progress in the Source Code Domain

```javascript
// Put your JavaScript here that you want to rename, deobfuscate,
// or infer types for:
function chunkData(e, t) {
    var r = e.length;
    var i = 0;
    for (; i < r; i += t) {
        n.push(e.substring(i, i + t));
    }
    return n;
}

// You can also use some ES6 features.
const get = (a, b) => a.getElementById(b);
```

**RESULT.**

```javascript
'use strict';
/**
 * @param {string} bin
 * @param {number} size
 * @return {?}
*/
function chunkData(bin, size) {
    /** @type {Array} */
    var results = [];
    var length = bin.length;
    /** @type {number} */
    for (; i < length; i += size) {
        if (i + size < length) {
            results.push(bin.substring(i, i + size));
        } else {
            results.push(bin.substring(i, length));
        }
    }
    return results;
}

const get = (doc, key) => {
    return doc.getElementById(key);
};
```
Progress in the Source Code Domain

```java
void f(int[] array) {
    boolean swapped = true;
    for (int i = 0; i < array.length && swapped; i++) {
        swapped = false;
        for (int j = 0; j < array.length - 1 - i; j++) {
            if (array[j] > array[j+1]) {
                int temp = array[j];
                array[j] = array[j+1];
                array[j+1] = temp;
                swapped = true;
            }
        }
    }
}
```

```
sort | 98.54%
bubbleSort | 0.35%
reverse | 0.25%
reverseArray | 0.23%
heapify | 0.15%
```

Un-Stripping Procedure Names
Un-Stripping Procedure Names

Start at the right place
Translate: Assembly Procedure → English
Sequence-To-Sequence (seq2seq) Models

• A basic approach:
  • LSTM encoder
  • LSTM decoder

• LSTM with attention & Transformers are state of the art for seq2seq tasks (machine translation, speech recognition, etc.)
Binary Syntax Is Very Local

call getaddrinfo
mov rax, [rbp-30h]
mov rdx, [rbp-50h]
mov rdx, cs:688588d
mov [rax], rdx
mov rax, [rbp-30h]
mov rdx, [rbp-580h]
mov [rax+8], rdx
mov rax, [rbp-30h]
call strerror
sub rdx, rax
idiv [rbp-28h]
call setsockopt
mov rdx, [rax]
mov eax, [rbx+40h]
cdqe
Binary Syntax Is Very Local

Register allocation is local to instruction/BB

Stack offsets local to procedure

Global offsets local to executable
Finding Prediction Anchors

```
call getaddrinfo
mov rdx, cs:qword_68858
mov rax, [rbp-30h]
mov rdx, [rbp-50h]
mov [rax], rdx
mov rax, [rbp-30h]
mov rdx, [rbp-580h]
mov [rax+8], rdx
mov rax, [rbp-30h]
call strerror
sub rdx, rax
idiv [rbp-28h]
call setsockopt
mov rdx, [rax]
Mov eax, [rbp+var3C]
cdqe
```

Focus On Calls

```
call getaddrinfo
...  
call strerror
...  
call setsockopt
...  
```

Not enough data and context
Finding Prediction Anchors

Combine binary program analysis with machine learning to find a sweet-spot

Not enough data and context
Augmented Call Sites as Learning Features
Using API Calls

API calls

... call getaddrinfo ...
... call strerror ...
... call setsockopt ...

Reconstructed API Call Sites

Calling Conventions + Library information

setsockopt( rdi, rsi, rdx, rcx, r8 )
Augmenting Call Sites

call socket(...)
mov [rbp-58h], rax
mov rax, [rbp-58h]
mov rdi, rax

mov rsi, 1
mov r8, 4

setsockopt(rdi, rsi, rdx, rcx, r8)

In C: setsockopt(sock_var, ..., 1, 4)
Augmenting Call Sites

Using concrete or abstracted values:

1. Concrete value (Integer, Enum, String)
2. ARG – procedure argument
3. GLOBAL - pointer to a global variable
4. RET – a return value from a call
5. STACK – pointer to stack memory
Pointer-Aware Slicing of Call Site Args

```
getaddrinfo(rdi, rsi, rdx, rcx)
```

```
mov rdi, rax
mov rax, [rbp-68h]
```

```
mov [rbp-68h], rdi
```

```
V(rbp), P([rbp-68h])
```

```
V(rax), P([rax])
```

```
∅
```

```
V(rdi)
```

```
P([rdi])
```

```
∅
```

```
∅
```

```
∅
```

```
V
```
Augmenting Call Site Arguments

getaddrinfo(\texttt{rdi}, \texttt{rsi}, \texttt{rdx}, \texttt{rcx})
Augmenting Call Site Arguments

Using concrete or abstracted values:
1. Concrete value (integer, float, string)
2. ARG – procedure argument
3. GLOBAL – pointer to a global variable
4. RET – a return value from a call
5. STACK – pointer to stack memory

getaddrinfo(\texttt{rdi}, \texttt{rsi}, \texttt{rdx}, \texttt{rcx})
Augmenting Call Site Arguments

```
getaddrinfo(ARG, rsi, rdx, rcx)
```
Augmented Control Flow Graph
Augmented Control Flow Graph

Useful for training seq2seq or GNN models

socket(2,1,0)

setsockopt(RET,0,10,STK,4)

printf(GLOBAL, ...)

close(...)

printf(GLOBAL, ...)
Extracting Paths From the ACFG

Extract simple paths (no loops)
Our Approach: [Set-Of-Seq]-To-Seq

```
setsockopt(RET, 1, 2, STK, 4)
getaddrinfo(ARG, ARG, STK, STK)
socket(…)
memset(STK, 0, 48)
bind(…)
listen(…)
```
Evaluation

Implementation: Nero
**Evaluation Corpus**

- **GNU software repository**
- **Remove Duplications**
- **67,246 Labeled Procedures**
- **Strip & Obfuscate APIs**
- **8:1:1 Package-Based Split**
Executable Obfuscation Types

• String encoding/encryption

• Code obfuscations (opaque predictions, etc.)

• Commercial (known) / Home-made packers
  • Header manipulation => API calls not visible
Simulating Header Manipulation

- Zeroing `.dynstr` removes imported libraries & procedure names

```plaintext
<table>
<thead>
<tr>
<th>Procedure</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>.free</td>
<td>00000000000400860</td>
</tr>
<tr>
<td>.puts</td>
<td>00000000000400870</td>
</tr>
<tr>
<td>.fread</td>
<td>00000000000400880</td>
</tr>
<tr>
<td>.strlen</td>
<td>00000000000400890</td>
</tr>
<tr>
<td>__stack_chk_fail</td>
<td>000000000004008A0</td>
</tr>
<tr>
<td>__libc_start_main</td>
<td>000000000004008B0</td>
</tr>
<tr>
<td>.strcmp</td>
<td>000000000004008C0</td>
</tr>
<tr>
<td>.fprintf</td>
<td>000000000004008D0</td>
</tr>
<tr>
<td>.memcpy</td>
<td>000000000004008E0</td>
</tr>
<tr>
<td>.malloc</td>
<td>000000000004008F0</td>
</tr>
<tr>
<td>.realloc</td>
<td>00000000000400900</td>
</tr>
<tr>
<td>.atol</td>
<td>00000000000400910</td>
</tr>
<tr>
<td>.fopen</td>
<td>00000000000400920</td>
</tr>
<tr>
<td>.exit</td>
<td>00000000000400930</td>
</tr>
<tr>
<td>.fwrite</td>
<td>00000000000400940</td>
</tr>
<tr>
<td>.strdup</td>
<td>00000000000400950</td>
</tr>
</tbody>
</table>
```
## Evaluation Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Stats</th>
<th>Stripped</th>
<th></th>
<th>Stripped &amp; Obfuscated API Calls</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Prec</td>
<td>Rec</td>
<td>Prec</td>
<td>Rec</td>
</tr>
<tr>
<td>LSTM-text</td>
<td></td>
<td>22.32</td>
<td>21.16</td>
<td>15.46</td>
<td>14.00</td>
</tr>
<tr>
<td>Transformer-text</td>
<td></td>
<td>25.45</td>
<td>15.97</td>
<td>18.41</td>
<td>12.24</td>
</tr>
<tr>
<td>Debin [He et al. 2018]</td>
<td></td>
<td>34.86</td>
<td>32.54</td>
<td>32.10</td>
<td>28.76</td>
</tr>
<tr>
<td>Nero-LSTM</td>
<td></td>
<td>39.94</td>
<td><strong>38.89</strong></td>
<td><strong>39.12</strong></td>
<td>31.40</td>
</tr>
<tr>
<td>Nero-Transformer</td>
<td></td>
<td><strong>41.54</strong></td>
<td>38.64</td>
<td>36.50</td>
<td><strong>32.25</strong></td>
</tr>
</tbody>
</table>

"Debin: Predicting Debug Information in Stripped Binaries", CCS’18
## Ablation Study

<table>
<thead>
<tr>
<th>Components</th>
<th>Prec</th>
<th>Rec</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only Calls → LSTM</td>
<td>23.45</td>
<td>24.56</td>
<td>24.04</td>
</tr>
<tr>
<td>Augmented Call Sites → LSTM</td>
<td>36.05</td>
<td>31.77</td>
<td>33.77</td>
</tr>
<tr>
<td>Paths → Only Calls → LSTM</td>
<td>29.84</td>
<td>24.08</td>
<td>26.65</td>
</tr>
<tr>
<td>Paths → Augmented Call Sites → LSTM</td>
<td>39.94</td>
<td><strong>38.89</strong></td>
<td>39.40</td>
</tr>
</tbody>
</table>
## Prediction Examples

<table>
<thead>
<tr>
<th>Model</th>
<th>Prediction</th>
<th>Ground Truth</th>
<th>Model</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debin [He et al. 2018]</td>
<td>bt open</td>
<td>read index</td>
<td>check new watcher</td>
<td>get user groups</td>
</tr>
<tr>
<td>LSTM-text</td>
<td>&lt;unk&gt;</td>
<td>check opt</td>
<td>display</td>
<td>signal setup</td>
</tr>
<tr>
<td>Transformer-text</td>
<td>ipmi disable</td>
<td>&lt;unk&gt;</td>
<td>config file ipmi</td>
<td>regfree</td>
</tr>
<tr>
<td></td>
<td>coredump</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nero-LSTM</td>
<td>vfs read file</td>
<td>check file</td>
<td>get ip groups</td>
<td>install handlers</td>
</tr>
<tr>
<td></td>
<td>read file</td>
<td>check state</td>
<td>get user groups</td>
<td>install signal</td>
</tr>
<tr>
<td>Nero-Transformer</td>
<td>system list</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


## Qualitative Evaluation

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Package</th>
<th>Ground Truth</th>
<th>Predicted Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programmers Vs English Language</td>
<td>wget</td>
<td>i18n_initialize</td>
<td>i18n_init</td>
</tr>
<tr>
<td></td>
<td>direvent</td>
<td>split_cfg_path</td>
<td>split_config_path</td>
</tr>
<tr>
<td></td>
<td>gzip</td>
<td>add_env_opt</td>
<td>add_option</td>
</tr>
<tr>
<td>Date Structure Name Missing</td>
<td>gtypist</td>
<td>get_best_speed</td>
<td>get_list_item</td>
</tr>
<tr>
<td></td>
<td>wget</td>
<td>ftp_parse_winnt_ls</td>
<td>parse_tree</td>
</tr>
<tr>
<td></td>
<td>gzip</td>
<td>abort_gzip_signal</td>
<td>fatal_signal_handler</td>
</tr>
<tr>
<td>Verb Replaced</td>
<td>units</td>
<td>read_units</td>
<td>parse</td>
</tr>
<tr>
<td></td>
<td>findutils</td>
<td>share_file_fopen</td>
<td>add_file</td>
</tr>
<tr>
<td></td>
<td>mcsim</td>
<td>display_help</td>
<td>show_help</td>
</tr>
</tbody>
</table>
## Qualitative Evaluation

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Package</th>
<th>Ground Truth</th>
<th>Predicted Name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Programmers Vs English Language</strong></td>
<td>wget</td>
<td>i18n_initialize</td>
<td>i18n_init</td>
</tr>
<tr>
<td></td>
<td>direvent</td>
<td>split_cfg_path</td>
<td>split_config_path</td>
</tr>
<tr>
<td></td>
<td>gzip</td>
<td>add_env_opt</td>
<td>add_option</td>
</tr>
<tr>
<td><strong>Date Structure Name Missing</strong></td>
<td>gtypist</td>
<td>get_best_speed</td>
<td>get_list_item</td>
</tr>
<tr>
<td></td>
<td>wget</td>
<td>ftp_parse_winnt_ls</td>
<td>parse_tree</td>
</tr>
<tr>
<td></td>
<td>gzip</td>
<td>abort_gzip_signal</td>
<td>fatal_signal_handler</td>
</tr>
<tr>
<td><strong>Verb Replaced</strong></td>
<td>units</td>
<td>read_units</td>
<td>parse</td>
</tr>
<tr>
<td></td>
<td>findutils</td>
<td>share_file_fopen</td>
<td>add_file</td>
</tr>
<tr>
<td></td>
<td>mcsim</td>
<td>display_help</td>
<td>show_help</td>
</tr>
</tbody>
</table>
Takeaway Messages

Translate: Assembly Procedure
→ English

Use Augmented Call Sites as Learning Features

In C: setsockopt(sock_var, ..., 1, 4)