Reverse Engineering Preprocessor

REPr

Jacob Greenberg

Project Information

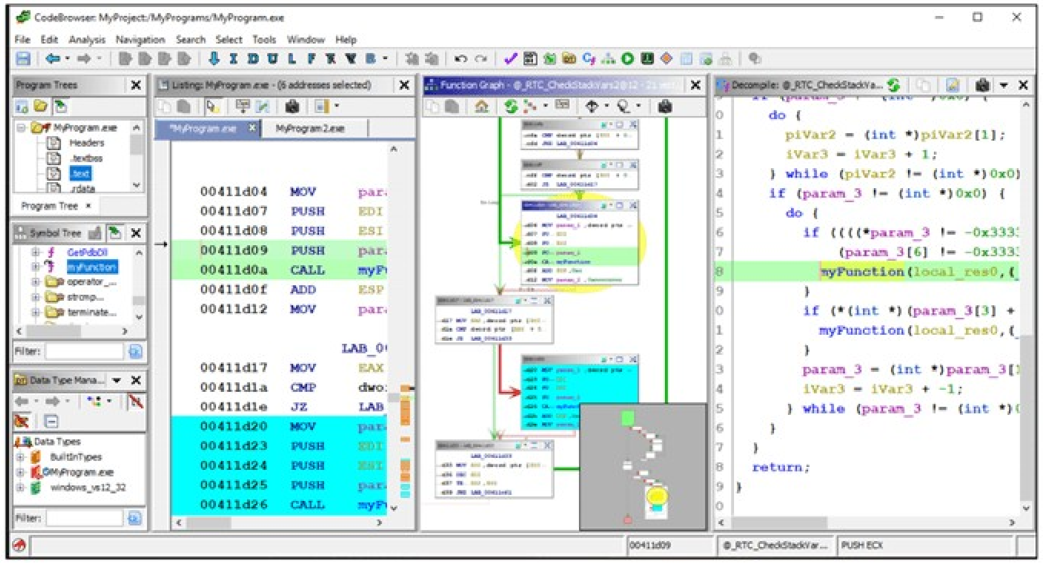
Team

- Jacob Greenberg
- Gary S. (SWE @ Battelle)

Overview

The RE Preprocessor is a tool to make reverse engineering more streamlined. It acts as a modular file identification and extraction tool, reducing the steps between firmware and decompilation.

What is Software Reverse Engineering?


CodeBrowser: MyProject:/MyPrograms/MyProgram.exe
File Edit Analysis Navigation Search Select Tools Window Help
Program Trees
Symbol Tree
Data Type Mana...
Listing: MyProgram.exe - (6 addresses selected)
Function Graph - @_RTC_CheckStackVars2@12
Decompile: @_RTC_CheckStackVa...

```
00411d04    MOV     par
00411d07    PUSH    EDI
00411d08    PUSH    ESI
00411d09    PUSH    par
00411d0a    CALL    myF
00411d0f    ADD     ESP
00411d12    MOV     par

                 LAB_0
00411d17    MOV     EAX
00411d1a    CMP     dwo
00411d1e    JZ      LAB
00411d20    MOV     par
00411d23    PUSH    EDI
00411d24    PUSH    ESI
00411d25    PUSH    par
00411d26    CALL    myF
```

```
    do {
      piVar2 = (int *)piVar2[1];
      iVar3 = iVar3 + 1;
    } while (piVar2 != (int *)0x0)
    if (param_3 != (int *)0x0) {
      do {
        if ((((*param_3 != -0x333
            (param_3[6] != -0x333
          myFunction(local_res0,(
        }
        if (*(int *)(param_3[3] +
          myFunction(local_res0,(
        }
        param_3 = (int *)param_3[
        iVar3 = iVar3 + -1;
      } while (param_3 != (int *)(
    }
  }
  return;
}
```

00411d09 | @_RTC_CheckStackVar... | PUSH ECX

- Reverse engineering is a technique used to identify the purpose of a compiled or obfuscated program
- RE is frequently used in vulnerability research and malware analysis

What is Software Reverse Engineering?

- Manufacturers know about reverse engineering and often take steps to make it more difficult
  - Packing or obfuscating code
  - Using arbitrary file formats
  - Hiding code within other files
- Most RE software is only setup to handle executable code and is easily confused by these techniques

Reverse Engineering Example

A company has hired you to find vulnerabilities in a security camera they manufacture.

You disassemble the camera and are able to extract the embedded firmware.

Reverse Engineering Example

But the firmware isn’t just compiled code, it is often compressed and usually contains an entire filesystem.

How can we get the programs we care about out of this messy binary file?

Project Abstract

REPr is a modular file identification and extraction tool

Users are able to create plugins which can aid in identifying or extracting a specific type of file

Identifiers will return how confident they are about a given file type, leaving room for ML-based file identifiers

Users are able to easily share plugins between themselves to prevent duplicated work

How can a Preprocessor Help?

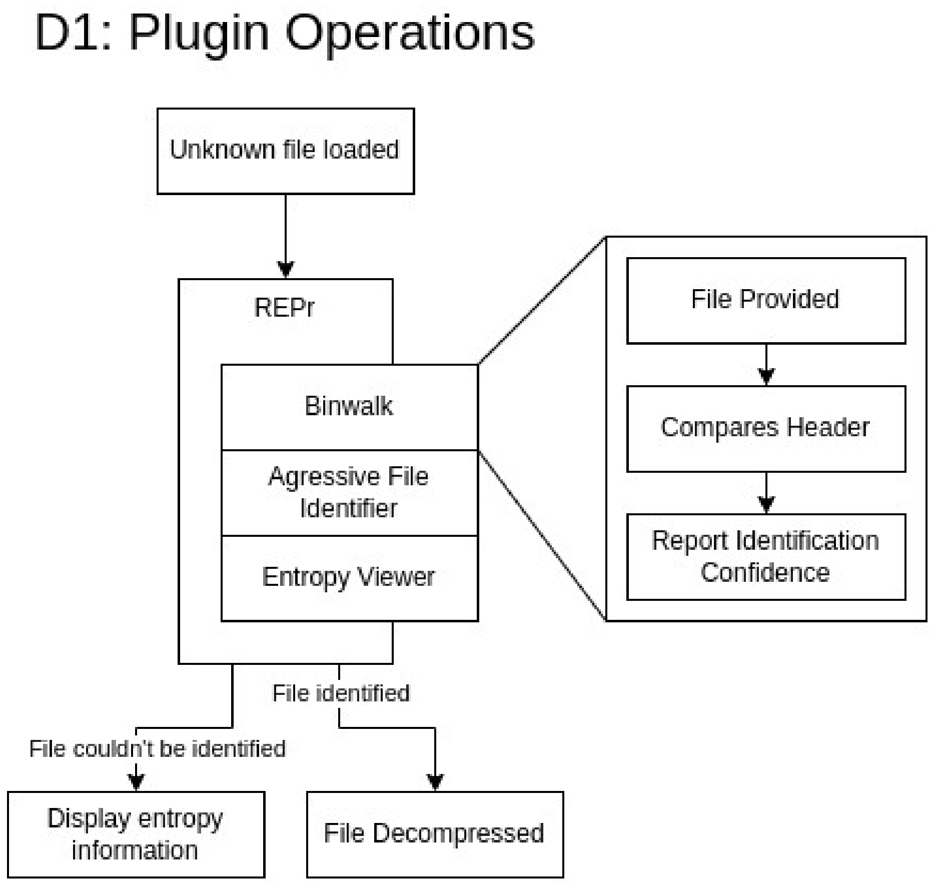
In many cases, reverse engineers face difficulties identifying files they are working with

The author of malware might obfuscate their virus with a special program to make it more difficult to reverse

The preprocessor’s goal is to provide additional information to the reverse engineer so they can make more informed decisions

Modularity makes it easier to share file identification plugins between users

Design Diagram



The diagram above shows the general process of how an unknown file is identified.

User Stories

- A software reverse engineer would like to be able to quickly identify and extract a file so they can begin reverse engineering it. If the file cannot be identified programmatically, they’d like as much information as they can get about it.
- A software reverse engineer doesn’t want to waste time processing a known file type, they’d like a way to load a file with as few clicks as possible.
- A project manager wants to maximize their employee's productivity. They’d like to know about potential delays caused by unknown file formats ahead of time.
- A division head wants to see different projects and teams sharing resources and minimizing duplicated effort.

Constraints

Professionalism
By automating a common sticking point in RE work, reverse engineers could become less aware of methods to manually identify a file

Legality
While the tool itself doesn’t represent a legal issue, plugins generated by users could violate intellectual property agreements. Plugin authors should be aware of where the code used in their plugins comes from

Project Progress

Completed Goals

- Plugin system
- Basic identification and extractor plugins
- Informational plugins

Future Goals

- Integrate with popular disassemblers
- Support for additional file types
- Train an ML model for file classification (stretch)

Expo Demo

While not the flashiest project, the expo demo will show off all the time the tool can save. The expo will consist of demonstrating the speedup a reverse engineer might see when using the tool. Judges would be encouraged to bring their own obscure files to test.

University of CINCINNATI | COLLEGE OF ENGINEERING AND APPLIED SCIENCE