Patching a Compiled Binary

Dr. Jared DeMott
CTO AND FOUNDER

@jareddemott www.vdalabs.com

Overview

- Review Homework
- Patching
- Flirt/Flare
- Malware

lab2.c

```c
#include <stdio.h>

int main()
{
    char password[7];
    int i = 1;
    while (i == 1)
    {
    printf("Please Enter a password ");
    fgets(password, 7, stdin);

    password[0] = (password[0] + 121) / 2;              // ends up = 'v' 118
    password[1] = (((password[1] + password[0]) * 2) / 3) - 30;  //  //ends up = 't' 116
    password[2] = 101;      //ends up = 'e' 101
    password[3] = (((password[3] + password[1]) + 1) / 2);    // ends up = 's' 115
    password[4] = (((password[4] + password[3] + password[0]) + 2) / 3);  //ends up = 'p' 112
    password[5] = (((password[5] + password[4]) / 2));      // ends up = 'r' 114

    if (password[0] == 118 && password[1] == 116 && password[2] == 101 && password[3] == 115 && password[4] == 112 && password[5] == 114)
    {
        printf("good job");
        i = 0;
    }
    else
    {
        printf("Not quite\n");
        printf("After conversion your input looks like ");
        printf("%c%c%c%c%c%c", password[0],password[1],password[2],password[3],password[4],password[5]);
        printf("\nTry again\n");
    }
    }
    return 0;
}
```

Binary Patching

Why?

- Fix bugs
- Add bugs
- Crack licenses
- More

Binary Patching

How?

- Hex editor will do, but IDA is more useful
- Patch submenu
  - Enabled by editing cfg/idagui.cfg
- Produce file options
  - Create EXE file
  - Create DIF file

Steps

From Hex View

- In HEX VIEW, R-click and select ‘edit’
  - Change
  - R-click and ‘commit changes’
- Edit → Patch Program
  - Patch a byte, word, or assemble
- IDC can be used as well

File → Produce File → IDA DIF file

- Use Eagle’s ida_patcher program to patch binary
  - ida_patcher.exe –p prog.dif
  - prog.exe will now be patched

More Patching Thoughts

'In function' patching works for small changes

- If significant "upgrades" are required, it's best to patch in "holes"
  - Each section must begin with specific alignment
  - May offer "slack space" opportunities at end of each section
    - Size on disk vs. size in memory
    - Keep in mind that .rdata is initialized and .bss is uninitialized
  - Could put a backdoor in one of these

Demo

Lab 3

- Play game.exe
  - Without a valid license key

Flirting and Flaring

Stripped Binaries

Are a pain, since symbol information has been removed

- Hopefully import info still present

Statically compiled AND stripped are the worst…

- Large amount of time reversing functions like *printf()*
- CTF game organizers love to include these types of binaries to make life hard for the players
- Fortunately there's help

FLAIR

Fast Library Acquisition for Identification and Recognition

Examines a library file and creates signatures for each exported function

You can feed IDA those signatures to match functions in the current database

- Easily could save >60% effort

Install

Extract the flairxxx.zip

- Match IDA version
  - Need paid version to get SDK
- Put in the main IDA directory
- Pat.txt
  - Describes the structure of “pattern” files
- Plb.txt, pcf.txt
  - Describe the use of the pcf and plb library parsers
- Sigmake.txt
  - Details the collision resolution process

Creating a FLIRT Signature

Fast Library Identification and Recognition Technology

- Step 1: Identify the static library used
  - Try the *file* utility to identify the OS
  - Use *strings* to identify library
  - Download identified library
- Step 2: Use the appropriate library parser to create a “pattern” file <parser> <lib> <pat file>
  - example: *pelf.exe libssl.a libssl.pat*

Creating a FLIRT Signature

Step 3/4: Create signatures from patterns

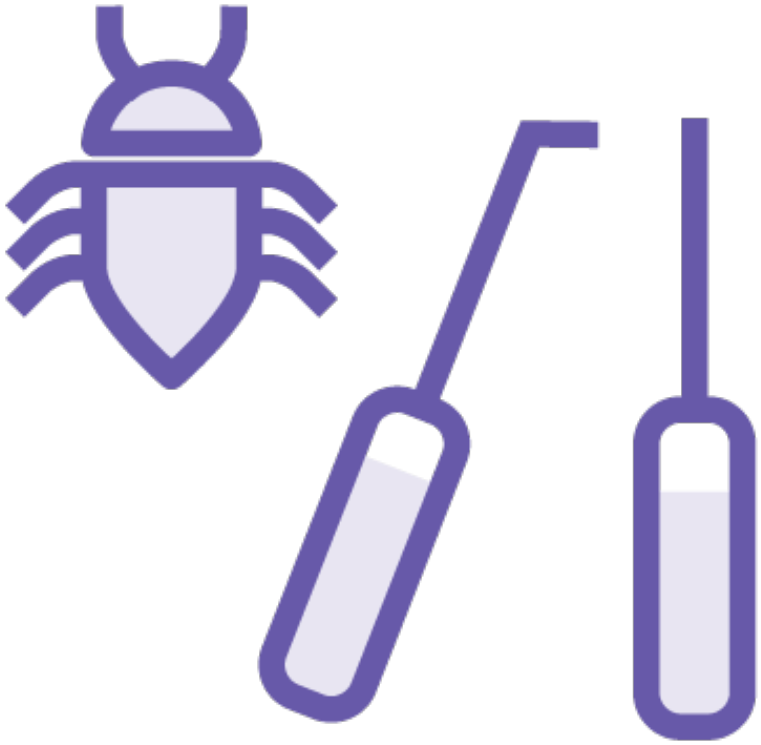
- The *sigmake* utility reads pattern files and generates .sig files
  - Sometimes patterns are ambiguous
    - If so, sigmake generates an *exclusions* (.exc) file
  - All exclusions must be manually resolved, then rerun until no more issues are left
- Example:
  - *sigmake.exe libssl.pat libssl.sig –n "libssl library"*

Add the new FLAIR Signature to IDA

Step 5:

- Close IDA pro
  - This has to be done for plug-ins as well
- Manually copy new .sig file in the \sig directory
- Reopen the IDB database
- File → Load File → Flirt signature file
  - Be sure to choose the file you just created from list

Thoughts on Malware Analysis



Basic malware classes cover RE

- Check

Next

- Tips and tricks
  - Quick tools
    - VirusTotal.com or malwr.com
  - System APIs used
  - Shellcode
  - Analysis tools
  - Static and dynamic analysis

Summary

Patching

SDK

- FLIRT/FLAIR

Malware Analysis Suggestion

Next:

- Reversing C++