Reverse Engineering Linux x86 Binaries - NOTES

What is Reverse Engineering?

"Oh, you can't get out backwards. You've got to go forwards to go back, better press on." - Willy Wonka

- The process of examining and probing a program to determining the original design.
- The documentation from this process can be used to
  - Document the characteristics of an unknown program
  - Clarify published interfaces for interacting with the program
  - Implement another program to interact with a proprietry program
  - Modify the features or behaviour of a program

Technical Background

Network Protocols

- Understanding a network protocol or file format can be the fast track to interoperability
  - Sniffers capture and analyse network traffic.
  - Switched environment? Specially configured port, DugSongs ArpSpoof or proxy/gateway.
- The Samba team have done a lot of protocol decoding, if this is your field read Andrew Tridgell's "Network Analysis Techniques" presentation.

Function Calls

- When a program is written, it is logically divided into functions, which call each other to perform tasks.
- Every process has a scratch area referred to as the stack, on which values are stored in a first in last out fashion.
- ESP points to the bottom of the stack, EBP points to the top of the stack for the current function.
- Tracing function calls
  - xtrace

Libraries

- Image: memory layout with program printf-libc write highlighted
- Libraries contain functions that can be used by many programs, eg printf()
- Library usage
  - When a static executable is compiled, library functions are copied into the executable.
  - When a dynamic executable is executed, library functions are loaded from shared library files.
  - An executable can use libdl to load and use library files whilst running.
- Tracing library calls
  - ltrace

Kernel Functions

- Image: memory layout with program printf-libc write-kernel
- Some library calls need the kernels help, for example to perform IO.
- Userland/Kernel seperated, eg by a special instruction or an interrupt. User programs use system calls to interact with the kernel.
- The kernel provides the ptrace call for (amongst other things) tracing system calls.
- Tracing system calls
  - strace

Execution Tracing

- Execute program in a debugger, monitor or alter live data values or program behaviour
- xtrace (functions), ltrace (library functions) and strace (system calls) use the kernels ptrace functionality to intercept calls.
- If a programmer does not want their program traced or reversed, they may:
  - Statically link and strip symbols to hide function names
  - Hide code in library functions
  - Add debugging countermeasures to crash your debugger or wrest control back
  - Use run time code modification, encryption or compression
- Use a virtual machine
  - Easier to monitor
  - Filesystem easy to restore

The ELF File Format

- A format for executable object files (eg. executables, relocatable files, shared object files).
- Divides the file into headers and sections.
- May contain a Program Header Table, describing how to build a process image from the files sections.
- Interesting sections include:

```
.text:                    The programs executable instructions
.symtab:                  The symbol table (imported and exported)
.rel*:                    Relocation entries (where symbols used)
.data and .data1:         Initialised data
.rodata and .rodata1: Initialised read only data
.debug:                   Symbolic debugging information
```

The /proc Filesystem

- The proc filesystem is a representation of kernel structures
- It contains much useful information about running processes, including:
  - cmdline: command line process was invoked with
  - maps: memory map of process and libraries
  - status: process state, privileges, granular memory usage, signal handling and capabilities
  - fd: file descriptors in use by process
  - cwd: link to processes current directory
  - root: link to processes root directory
  - mounts: mount table visible to process

Reverse Engineering Techniques

Reverse Engineering Techniques

- Deadlisting and Disassembly
- Live Debugging and Tracing
- Enumeration
- Library Replacement/Interception

Deadlisting and Disassembly

Examining an ELF binary with binutils

- ldd, nm and objdump can dump and disassemble interesting sections
- dependencies: ldd /usr/bin/yes
- symbols: nm -D -l -S /usr/bin/yes
- sections: objdump -h /usr/bin/who
- data: objdump -s -j .rodata /usr/bin/who
- code: objdump -d -r -j .text /usr/bin/who

Examining an ELF binary with REC

- REC - the Reverse Engineers Compiler
- Compilers translate high level languages into assembly language.
- This process generates "signature" assembly code for program structures such as conditionals (if, else, switch) and loops (for, do, while).
- The original program structure can be guessed by looking for signature code.
- REC automates this process, examining binaries to generate C like code.
- Some structures or optimisations confuse REC, so assembly is intermingled with the output.
- http://www.backerstreet.com/rec/rec.htm

Live Debugging

Live Debugging with GDB

- GNU Debugger, includes the Binary File Descriptor library.
- Supports many CPU architectures, several high level languages, threading.
- A selection of GUIs are available.
- A lot of GDB features are based around having the source code available.

Live Debugging with strace

- strace, ltrace and xtrace can attach to processes
- these tools show calls as they happen

```
# ltrace -e fwrite_unlocked ls
fwrite_unlocked("init.d", 1, 6, 0x401585c0)      = 6
fwrite_unlocked("rc0.d", 1, 5, 0x401585c0)       = 5
fwrite_unlocked("rc2.d", 1, 5, 0x401585c0)       = 5
fwrite_unlocked("rc4.d", 1, 5, 0x401585c0)       = 5
fwrite_unlocked("rc6.d", 1, 5, 0x401585c0)       = 5
fwrite_unlocked("rc.sysinit", 1, 10, 0x401585c0) = 10
init.d  rc0.d  rc2.d  rc4.d  rc6.d     rc.sysinit
fwrite_unlocked("rc", 1, 2, 0x401585c0)          = 2
fwrite_unlocked("rc1.d", 1, 5, 0x401585c0)       = 5
fwrite_unlocked("rc3.d", 1, 5, 0x401585c0)       = 5
fwrite_unlocked("rc5.d", 1, 5, 0x401585c0)       = 5
fwrite_unlocked("rc.local", 1, 8, 0x401585c0)    = 8
rc      rc1.d  rc3.d  rc5.d  rc.local
```

Live Debugging with Fenris

- Fenris is a set of tools; Fenris, Aegir, Ragnarok, Dress
- Fenris: backend, tracer
- Aegir: client, interactive debugger
- Ragnarok: trace to HTML
- Dress: rebuilds stripped symbol table

```
[0804b158] 02   local fnct_11 (g/805ace7 "fenris.h")
[0804b158] 02   + 805ace7 = 805ace7:9  (first seen in F fnct_13:fnct_14)
[4009ee3e] 03    L strlen (805ace7 "fenris.h") = 8
[4009ee3e] 03    + 805ace7 = 805ace7:9  (first seen in F fnct_13:fnct_14)
[0804eb74] 03    local fnct_6 (9) (Click here for trace of this function)
[0804eaf3] 03    ...return from function = ""
[4009e8cf] 03    L strcpy (805bc58, 805ace7 "fenris.h") = 805bc58
[4009e8cf] 03    + 805ace7 = 805ace7:9  (first seen in F fnct_13:fnct_14)
[4009e8cf] 03    + 805bc58 = 805bc58:9  (first seen in L fnct_14:malloc)
[4009e8cf] 03    \ buffer 805bc58 modified.
[4009e8cf] 03    \ data migration: 805ace7 to 805bc58
[0804eb84] 02   ...return from function = ""
```

Enumeration

Network Enumeration

- Given reasonable knowledge of the protocol, alter one value over its range and observe results.
  - May trigger informative error messages.
- Libnet/LibPCap can be used to quickly build custom packets.

Enumeration with Plugins

- A variation on "poke and fsck"
- Stick values into the API and see what happens.
- eg. Determining data available to Netscape Directory Server plugins:
  - plugins are shared libraries
  - parameters passed in "parameter block"
  - slapi_pblock_get(pb, SLAPI_*, &value);
  - SLAPI_* is actually a number
    - Calling for SLAPI_* 1 to 300 reveals all parameters
    - Interpreting the return values is another challenge
    - Some values actually crash slapd
    - Some values cause warnings in logs

Library Replacement/Interception

Library Replacement/Interception

- LD_PRELOAD instructs the linker to preload shared libraries.
- This can be used to replace standard library functions.
  - eg. replace time(2) to return a different date to test for Y3k bugs.
  - or replace socket(), connect(), read(), write() for network or file interception
- The linker ignores LD_PRELOAD for set[ug]id executables.

Case Studies

The Via Mini-ITX Motherboard

The Problem

- Via produce a great little (22cm x 19cm) motherboard.
- The only produce binary drivers for the on board MPEG decoder.
- Drivers include a shared library and kernel modules.
- Ivor Hewitt reverse engineered the MPEG library.

Methodology

- By examining a disassembly, write similar C code.
- Examine function calls to determine arguments.
- Analyse function call tree to determine structure.

Tools

- objdump
- header files
- IDA-Pro

Results

- Reproduced source code for library.
- Source code reveals how to talk to kernel module.
- Definately not and example of clean room reverse engineering.

Documentation

Reasons for Documentation

- Make useful insight available
- Provide a history of your discoveries
- Document progress to prevent repetition

The Call Tree

- Shows what functions are called where, as a tree.
- Useful for getting a feel for the program flow.
- Can identify the purpose of a function.
- Fenris' Ragnarok can generate a similar tree from an execution trace.

```
.- main
| .- setlocale
| | brk
| | brk
| | brk
| | open
| | fstat64
| | mmap
| | read
| | brk
| | read
| | close
| | munmap
```

Writing API Documentation

- For each callable function, document the purpose and parameters.
- Almost a prerequisite for interacting with an API.
- See Unix's section 2 and 3 man pages for excellent examples of API documentation.
  - Name
  - Synopsis
  - Description
  - Parameters
  - Return Value
  - See Also

Writing Pseudocode

- For each interesting function, or even the entire program, write a pseudocode representation of the alogrithm.
- Pseudocode seperates the code from the concepts and ideas.
  - eg. bubblesort:

```
Assume the list is not sorted (sorted is set to false)
while(!sorted)
{
   Assume the list is sorted (sorted is set to true)
   Set i to loop through list from first to last - 1
   {
      if list[i] and list[i + 1] are not in the right order
      {
         switch them (called a swap)
         set sorted to false
      }
   }
}
```

example from http://www.rrcc-online.com/~julies/csc160/bubble.htm

Network Protocols

- RFCs contain many examples of how to document network protocols.
- Eg, from RFC 1885: ICMPv6:

```
The ICMPv6 messages have the following general format:

       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |     Type      |     Code      |          Checksum             |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                         Message Body                          +
      |                                                               |

   The type field indicates the type of the message. Its value
   determines the format of the remaining data.

   The code field depends on the message type. It is used to create an
   additional level of message granularity.

   The checksum field is used to detect data corruption in the ICMPv6
   message and parts of the IPv6 header.
```

File Formats

- Hex editors are great if you can spot a pattern.
- If not, you can use strace to examine file access patterns, or Fenris' aegir to analyse memory buffer accesses
- Failing that, you may have to disassemble the file access routines.
- Once you have the file format, documentation should be similar to that for network protocols

Legalities

- If in doubt, seek professional legal advice
  - Copyright Amendment (Digital Agenda) Act 2000
    - "Under s47D of the amended Act, a person may reverse engineer copies of a program owned by someone else, but only if they intend to make a product that interoperates with that program...it would appear that the development of emulators...is not fair play under current laws." [ipcr 2000]
    - Interfering with access controls on copyright material and removal of digital restrictions management is usually a breach.
  - Trade secrets are covered by contract law.
  - In some countries, software patents retard progress.
  - Try not to muddy the water by releasing other peoples intellectual property (eg. reverse engineered source code snippets).