What is Reverse Engineering?

- Reverse engineering (RE) is the process of extracting the knowledge or design blueprints from anything man made.
- The difference between RE and scientific research is that with RE the artifact being investigated is man made.
- Goal of RE is to obtain missing knowledge, ideas, and design philosophy when such information is unavailable.

Software Reverse Engineering: Reversing

- Reversing is about dissecting a program and examining its internals.
- In most industries RE is used for developing competing products, but this is not the case for the software industry.
  - RE of software is thought to be too complex to make sense financially.
- Common applications of RE in the software industry:
  - Security
  - Software development
Security-related Reversing

- Malicious software
- Reversing cryptographic algorithms
- Digital Right Management (DMR)
- Auditing program binaries

Security-related Reversing: Malicious Software (malware)

- Malicious software (e.g., viruses and worms) spread much faster now that computers are connected to the Internet.
- The infection process used to be slow (e.g., viruses spread by diskette sharing).
- Today’s worms can spread automatically to computers without any human intervention.
- Developers of malware use RE to find vulnerabilities in OS and application software.
- Developers of ant-virus software use RE to develop tools to protect users from malware.

Security-related Reversing: Reversing Crypto Algorithms

- Restricted algorithms, where the encryption is in the algorithm, (e.g., Caesar cipher) can be broken easily using RE.
- Key-based algorithms, where the algorithms are public but the key is secret, are more difficult to break.
  - Vulnerabilities include obtaining the key (need luck)
  - trying all possible combinations until you get the key (need quantum computer)
  - Look for a flaw in the algorithm that exposes the key of the original message (need RE)
Security-related Reversing: Digital Rights Management

- Media content providers have developed or use technologies (DRM) that control the distribution of digital content (e.g., music, movies).
- DMR technologies determine if the content should be made available or not.
- Crackers use RE to attempt to defeat DRM technologies. RE can help to:
  - reveal the secrets of DRM technology;
  - discover simple modifications that can be made to DRM technologies to disable the protection they offer.

Security-related Reversing: Auditing Program Binaries

- Open source software feels safer to run because it has been inspected and approved by thousands of impartial software engineers.
- RE is a viable (albeit limited) alternative for searching for security vulnerabilities when the source code of the program is not available.

Software Development-Reversing

- Achieving Interoperability with proprietary software
- Developing competing software
- Evaluating software quality and robustness
Reverse Engineering (Introduction to Binary Reversing)

Software Development-Reversing:

Achieving Interoperability with proprietary software

- Proprietary software libraries or OS APIs often have insufficient documentation.
- Solutions:
  - Try to get things to work via brute force.
  - Contact vendor for answers.
  - Use RE to figure out how the code works.

Software Development-Reversing:

Developing competing software

- Even if the competitor has unpatented technology it is not cost-effective to RE the entire product.
- The exception may be highly complex or unique designs/algorithms that are very difficult or costly to develop.
- There are legal issues to consider when RE competing software.

Software Development-Reversing:

Evaluating software quality and robustness

- It is possible to audit software to get an estimate of the general quality of the coding practices used in the program.
- Microsoft announced that customers purchasing over 1,000 seats may obtain Windows source code for evaluation purposes.
- The rest have to either trust Microsoft or resort to RE.
- Access to source code is more desirable than binary reversing, however.
System Software

- System software is a generic name for software infrastructure:
  - Compilers
  - Linkers
  - Debuggers
  - Operating systems
  - Assembly languages
- An accomplished RE must develop an understanding of system software.
- RE tools (e.g., disassemblers, decompilers) just provide data, not the answers.

System Software: Assembly Language

- Assembly language is a language that supports a specific computer platform.
- Machine code (a.k.a. binary code, object code) is a different representation of assembly language code.
- Machine code bits map directly to assembly language instructions and vice versa.
  - Assembly programmers use assemblers.
  - Reversers use disassemblers.
- Focus on the Intel IA-32 architecture, on which every 32-bit PC is based.

System Software: Compilers and Interpreters

- Compilers translate high-level programming languages (e.g., C, C++) into machine code that can be executed by a computer’s CPU.
- Some compilers (e.g., for Java and .NET) generate bytecode instead of object code.
  - Bytecode is interpreted by a program not the CPU.
- The greatest obstacle in deciphering compiler-generated code is the optimizations applied by most modern compilers.
  - Optimized code is often unintuitive and difficult to read.
  - E.g., loop unrolling, function in-lining.
System Software: Operating Systems (OS)

- An OS is a program that manages the computer's hardware and software applications.
- A good reverser must understand what the OS does and how it does it.
- We will focus on the Microsoft Windows OS.

The RE Process

- System-level reversing:
  - Techniques that help determine the general structure of the program.
  - Most of the information comes from the OS and, hence, OS monitoring utilities.
- Code-level reversing:
  - Techniques provide detailed information on a selected code segment.
  - Involves extracting design concepts and algorithms from binary code.

RE Tools

- System monitoring tools:
  - Network activity, file access, register access.
- Disassemblers:
  - Translate binary code to assembly language code.
- Debuggers:
  - Reversers use debuggers in disassembly mode to set breakpoints and step through a program's execution (with on-the-fly disassembly).
- Decompilers:
  - Attempt to produce high-level code (e.g., C) from an executable binary file: A reverse compiler!
  - Perfect decompilation impossible for most platforms.
Is RE legal?

- What social and economic impact does RE have on society?
  - Depends on what RE is used for …
- Whether a RE scenario is legal or not depends on many factors.

Legal Issues: Interoperability

- Exposing software interfaces facilitates the development of software that runs on top of the exposed platform.
  - Good for increasing the sales of the platform.
  - Bad if the competition offers a better product on the same platform.
- US court in legal case of Sega versus Accolade ruled in favor of Accolade.
  - Accolade did not violate code copyright of Sega.
  - Competition in the market benefited customers.
  - Court essentially authorized RE for the purpose of interoperability.

Legal Issues: Competition

- RE for interoperability has societal benefits, but when RE is used to develop competing products the situation is more complicated.
- Opponents of RE claim that reversing stifles innovation.
- Illegal (easy to prove): directly stealing code.
- Illegal (hard to prove): Decompiling a program and recompiling it to generate a different binary with the same functionality.
- Legal: RE small parts of a product to gather information, not code. Then develop code independently.
Legal Issues: Copyright Law

- A violation of copyright law is to directly copy protected code sequences from the competitor’s product into your own product.
- Some have claimed that intermediate copies during RE decompilation violates copyright.
  - These claims did not hold in court (e.g., Sega versus Accolade) because intermediate copies are created when a program is installed from a CD onto a hard disk.
- If the final product does not contain anything that was directly copied from the original product, copying is considered fair use.

Legal Issues: Trade Secrets and Patents

- **Patent:**
  - Owner controls the invention for up to 20 years.
  - Details of the patent are publicly disclosed, hence RE of a patented technology does not make sense.
  - Invention becomes public after patent expires.
  - Patent protects owner from a competitor who independently invents the same technology.
- **Trade secret:**
  - Granted automatically if product is kept a secret and significant effort was put into its development.
  - Protects against rogue employee selling secrets to the competition.
  - Does not protect product from RE.

Legal Issues: The Digital Millennium Copyright Act (DMCA)

- DMCA legally protects copyright protection systems from circumvention.
- DMCA is considered anti-RE legislation because such circumvention almost always involves reversing.
- DMCA only applies to copyright protection systems, so most RE applications are not affected by it.
- Two high-profile cases of DMCA violation:
  - Felten versus RIAA (audio recordings protection)
  - US versus Skylarov (Adobe eBook file format)
Legal Issues:
The Digital Millennium Copyright Act (DMCA)

What is prohibited?

1. Circumvention of copyright protection systems.
   - E.g., using a keygen program.
2. The development of circumvention technologies (DRM Digital Rights Management).
   - E.g., developing a keygen program.
   • A keygen program generates a serial number on-the-fly for programs that require a serial number during installation.

Legal Issues:
The Digital Millennium Copyright Act (DMCA)

What is allowed?

• Interoperability
• Encryption research on encryption products
• Security testing
• Educational institutions and public libraries can violate DMCA in order to evaluate a copyrighted work prior to purchasing it.
• Government investigation
• Regulation
• Protection of privacy

Legal Issues:
License Agreement Considerations

• Other than DMCA, there are no laws that directly prohibit or restrict RE.
• DMCA only applies to DRM products.
• Vendors add anti-RE clauses to software license agreements.
• In the US it is not clear that these clauses are enforceable.
• In the EU decompilation for the purposes of interoperability is permissible under the “Directive of the Legal Protection of Computer Programs”.
A Survey of Tools used for the Reverse Engineering of Software Systems

Topics

1. Basic background on assembly language
2. Types of reverse engineering tools and demonstrations of these tools:
   - Hex editors: WinHex, Tsearch
   - Decompilers: REC, DJ
   - Disassemblers/Debuggers: IDAPro, OllyDbg, Win32Dasm, BORG
3. Types of reverse engineering prevention tools
   - Code obfuscators: Y0da’s Cryptor, NFO

Program Abstractions

Computers understand binary code
Binary code can be written in hexadecimal
Hexadecimal code can be encoded in assembly language
Assembly language is human-readable but not as intuitive as source code
Decompilers convert assembly into an easier-to-read source code

11001111 10101 == CD21 == int 21
Assembly language is an abstraction of hexadecimal code.

Reverse Engineering Tool Categories

- **Hex Editors**
- Decompilers
- Disassemblers/Debuggers

Reverse Engineering Prevention Tools

- Code Obfuscators

**Hex Editors**

Hex editors read executing programs from RAM.
- Display their contents in hexadecimal code.
- Enable the editing of the running hexadecimal code.

<table>
<thead>
<tr>
<th>Hex editors</th>
<th>RAM editor</th>
<th>Data recovery tools</th>
<th>RAM Disassembler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tsearch</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>WinHex</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reverse Engineering (Introduction to Binary Reversing) © SERG
Hex Editors: WinHex

Author: sf-soft
Platform: Windows
License: Shareware
Website: http://www.sf-soft.de

Hex Editors: WinHex Features

• Display and edit (in hexadecimal) any program running in memory.
• Recover data that has been deleted (but not overwritten) from the hard-drive.

<table>
<thead>
<tr>
<th>Hex editors</th>
<th>RAM editor</th>
<th>Data recovery tools</th>
<th>RAM Disassembler</th>
</tr>
</thead>
<tbody>
<tr>
<td>WinHex</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DeepHex</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Hex Editors: WinHex
Reverse Engineering Tool Categories

- Hex Editors
- Decompilers
- Disassemblers/Debuggers

Reverse Engineering Prevention Tools

- Code Obfuscators

Decompilers

- Decompile a binary programs into readable source code.
- Replace all binary code that could not be decompiled with assembly code.

Decompilers: REC

**Author**: Backer Street Software
**Platform**: Windows/Linux
**License**: Freeware
**Website**: http://www.backerstreet.com/rec/rec.htm
**Decompilers: REC Features**

- Decompiles a program from binary code to C pseudo-code.
- Translates any binary it cannot decompile into assembly code.
- Typically generates about 60-70% of the program source code.

**Decompilers: REC**

**Reverse Engineering Tool Categories**

- Hex Editors
- Decompilers
- **Disassemblers/Debuggers**

**Reverse Engineering Prevention Tools**

- Code Obfuscators
Disassemblers/Debuggers

- Convert binary code into its assembly equivalent.
- Extract ASCII strings and used libraries.
- View memory, stack and CPU registers.
- Run the program (with breakpoints).
- Edit the assembly code at runtime.

Disassemblers/Debuggers
Programs & Features chart

<table>
<thead>
<tr>
<th>Product</th>
<th>Dis-</th>
<th>Proven-</th>
<th>Debug-</th>
<th>String</th>
<th>Memory</th>
<th>Feature</th>
<th>Feature</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>BORG</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>W32Dasm</td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>OllyDbg</td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>IDAPro</td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
</tbody>
</table>

Disassemblers/Debuggers: OllyDbg

Author: Oleh Yusuchuk
Platform: Windows
License: Freeware
Website: [http://home.t-online.de/home/Ollydbg/](http://home.t-online.de/home/Ollydbg/)
**Disassemblers/Debuggers:**

**OllyDbg Features**

- Executes program in a controlled environment.
- Allows the flow of the program to be controlled.
- Uses a convenient layout showing hexadecimal, assembly, CPU registers and stack.
- Allows the program to be dumped from the memory onto the hard-disk.
- Highlights recently changed values in memory/stack/CPU registers.

**Reverse Engineering Tool Categories**

- Hex Editors
- Decompilers
- Disassemblers/Debuggers

**Reverse Engineering Prevention Tools**

- Code Obfuscators
**Code Obfuscators**

- Encrypts the code of a program so you cannot view it in assembly.
- Disallows the program to be run if it detects a known disassembler or debugger running.

<table>
<thead>
<tr>
<th>Obfuscator</th>
<th>Encryption</th>
<th>Anti-disassembly</th>
<th>Geoblock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y0da's Cryptor</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>XFO</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

**Diagram of Code Obfuscators**

Diagram showing the process of obfuscation with an executable, obfuscator, and encrypted executable.

**Code Obfuscators: Y0da's Cryptor**

- **Author:** Y0da
- **Platform:** Windows
- **License:** Freeware
- **Website:** y0da.cjb.net
**Code Obfuscators:**

**Y0da's Cryptor Features**

- Encrypts the code of a program and inserts a decryption program in the "main" function of the code. (Obfuscation)
- Detects all major debuggers and disassemblers. (Anti-Debugging)

<table>
<thead>
<tr>
<th>Obfuscator</th>
<th>Obfuscation</th>
<th>Anti-Debugging</th>
<th>GUI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y0da's Cryptor</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Code Obfuscators:**

**Y0da’s Cryptor**

**Conclusions**

- We are conducting an extensive survey of RE tools.
- Most of the tools we have surveyed fall into the categories of:
  - Hex Editors
  - Decompilers
  - Code Obfuscators
  - Disassemblers/Debuggers