



REVERSE ENGINEERING

EVALUATING OPEN
SOURCE TOOLS

InquisIT, LLC.

14900 Bogle Drive, Suite 203
Chantilly, VA 20151

Info@inquisitllc.com

www.inquisitllc.com

All Rights Reserved.

INTRODUCTION

Reverse engineering provides insight into how examined code works from an outside standpoint. Primary reasons for performing reverse engineering on malware are providing insight into the malware's capabilities and behavior. Additionally, the reviewed malware's uniquely identifying characteristics may be included in the form of signatures to an organizations existing Antivirus definition, as well as security and network monitoring tools.

The reverse engineering process serves to provide a quantifiable method by which threats executed against an organization's network may be confirmed, functionally identified, and disseminated throughout the organization's chain of command.

Software and malware reverse engineering have practical application in the Federal space because they offer tangible data showcasing unmediated vulnerabilities, misconfigurations and other gray areas allowing malicious actors' access to Federal assets.

In this whitepaper, you will read about three open source tools that can assist in executing a reliable and repeatable reverse engineering process.



SECURITY MINDSET

Security should be everyone's concern, regardless of location within the hierarchy. Having a security mindset is a skill that must be organizationally reinforced through the implementation of best practices with continual refreshment of knowledge and training. However, engineers take these concepts

to the next level and not only repel the threat(s) or attack(s), but also face the unknown head-on. Often peeking behind the curtain of compiled, packed, frequently obfuscated, and forensically resistant code to understand its true capability and the intention of its authors.

“What we should actually be doing is thinking about what are our key controls that will mitigate the risks. How do we have those funneled and controlled through the team that we have, how do we work through that in a well formatted, formulated process...”

Dr. Chris Pierson, CEO, Binary Sun Cyber Risk Advisors,
at SecureWorld Charlotte



WHY USE OPEN SOURCE TOOLS?

Most growing businesses and their cost-conscious government counterparts continually seek ways to reduce dependency on expensive applications and systems as well as redundant man-hour intensive programming efforts.

Unexpected expenses can quickly stack up for legacy systems that are no longer vendor maintained or supported. Proprietary customizations and third-party add-ons or plugins may then be required to maintain the needed baseline functionality and compatibility of these systems. This scenario often involves minimal support, with upgrades and patches provided, if at all, by development personnel in a process totally invisible to your organization.







Open source tools give engineers and their organizations free access to source code and applications that fit their needs. Additionally, organizations may reduce the amount of redundant coding being performed by collaborating with other agencies and the community during development phases.

By being involved in the development process, engineers and decision makers become more aware of the influencers of key application development, directional changes, and trends in security and innovation, allowing for greater foresight and lifecycle planning.

REVERSE ENGINEERING TOOLS

Similar to pathogen discovery via electron microscopy, reverse engineering allows us to look at and understand source code in a way that would be otherwise inaccessible. From the information gathered we can better visualize and make decisions regarding development, as well as detect and remediate bugs that could cause instability and even be used as attack vectors.

Reverse engineering can also help detect malicious code written into legitimate software, as well as be used to detect vulnerabilities in cryptographic standards implementations.

Tool Features	 x64dbg	 Radare2	 Yara
	X64/x86 Support with Single Interface	Debug Natively or Utilize External Targets	Create Descriptions of Malware Families Based on Textual or Binary Patterns
	Built on Open Source Libraries	Various OS Types, Architectures and File Formats Supported	Runs on Win+MacOS+Linux
	Basic Debug Symbol (pdb) Support	Frequent Build Updates	Extremely Flexible Rule String / Boolean Condition Syntax

REVERSE ENGINEERING TOOLS



X64DBG

AN OPEN-SOURCE X64/X32 DEBUGGER FOR WINDOWS

X64dbg is a binary debugger that boasts not only an open source price tag, but also many professional features that the reverse engineer will find useful. Some of its key features are: Memory Map, Symbol View, Thread View, Source Code View, Content Sensitive Register View, fully customizable color scheme, Integrated Import Reconstructor, Disassembler, JSON user database for mark-ups and documentation, Built-in Assembler, Executable Patching, Pattern Matching, and a Decompiler, to name a few. Existing functionality may be extended by the use of supported plugins. The executable is offered as well as the option to compile from source.



RADARE2 (R2)

AN LGPL PORTABLE REVERSING FRAMEWORK

Radare2 is a powerful portable reversing framework that provides many capabilities to the reverse engineer. These offerings include Disassembly and Assembly for multiple architectures, debugging with local native and remote debuggers, file system data carving and forensic discovery, data structure visualization, program patching to remediate vulnerabilities or discover undocumented features, embedded webserver for collaborative analysis, software exploitation testing. Radare2 may be scripted with javascript, python, GO and many more well-known languages. Additionally, platform support is available for Linux, *BSD, Windows, MacOS, Android, iOS, QNX, Solaris, and Haiku.



YARA

THE PATTERN MATCHING SWISS KNIFE FOR MALWARE RESEARCHERS (AND EVERYONE ELSE)

YARA is a tool capable of assisting reverse engineers and malware researchers with identifying and classifying malware samples as well as creating descriptions of malware families based on unique criteria. Rules, or descriptions in YARA, are comprised of a set of strings and a Boolean expression which determine the logic of the description. YARA descriptions may utilize regular expressions, wild cards, special operators, and case insensitive strings (which must be declared as described in the documentation). Currently, YARA is supported on Windows, Linux, MacOS, and may be invoked and executed via cli or via custom Python scripts using the Yara-python extension.

KEY TAKEAWAYS

fees.

Including reverse engineering processes within an organization's security posture is a reliable way to identify security gaps, bugs, and

As demonstrated above, many robust and modern toolsets exist to aid in your organization's effort to mitigate risk and achieve a more actionable awareness of the threats and vulnerabilities hidden in plain sight.

By using open-source tools, you are assured to have available the best the industry has to offer, as well as in many cases, the opportunity to perform in-house customizations to those same tools without incurring fees from support contracts/consultants and additional licensing.

There is a very robust capability set available using open source tools. The high availability and accessibility of these tools can help any organization who wants to reduce costs for IT