An Exploratory Endeavor in the Reverse Engineering of a Multi-platform Compiler

Mohamad Mokbel

mfmokbel@live.com

May 06, 2020

ABSRTACT

Reverse engineering software written in a native programming language requires the understanding of different phases of the compilation process in addition to the libraries involved, code optimizations, language standards, file format, generated code, and other intricacies. For malware, they are mostly written in a native programming language such as x86 Assembly, C, C++, Objective-C, Delphi and other similar languages. For those languages we already have good tooling, knowledge base and understanding of how to reverse engineer them and in particular when it comes to the recognition of compiler and OS standard or, specific libraries.

With the advent of new native programming languages such as the Go language by Google, new research had to be carried out so that RE'ers can have a better understanding and easier time dealing with these language binaries.

In the last couple of years, malware authors started using another native programming language called PureBasic. This language is very powerful and produces native code with extensive library support. Moreover, the compiler generates code for all major platforms including Windows, Linux and MacOS. In this talk, we'll delve into the inner workings of how the language works, the compiler architecture, reverse engineering of the language libraries, and more importantly, the release of a complete parser for the libraries, IDA FLIRT signatures, and an IDA plugin. This all for the purpose of making reverse engineering of this language easier. To our knowledge, this is the first research that tackles this language.

DESCRIPTION

PureBasic is a sophisticated programming language with an easy syntax. It provides the perfect machinery for malware authors and tool developers to write cross-platform malicious implants and utilities. Moreover, it provides full access to native Windows APIs. Furthermore, it comes with an extensive set of optimized and specialized libraries that makes it the ideal language for prototyping and writing GUI applications, among others.

Those libraries are stored in proprietary file formats that get extracted upon compilation for statically linking relevant libraries with the program's code. In this talk we will unravel the complete data structure of those different file formats, with a tool that would extract all the stored information, in a contextual manner, targeting Windows, Linux and MacOS versions of the compiler.

Moreover, we will release a full set of IDA FLIRT signatures for all the compilers libraries to make it easier for RE'ers to weed through large binaries compiled with PureBasic. With the release of the parser tool, it should be quite easy to generate the same signatures for other versions of the libraries.

Furthermore, we will release an IDA Pro plugin that would help in annotating all identified PureBasic library functions with a descriptive comment, and attempt to identify binaries compiled with PureBasic compiler.

As a case study, we will demonstrate two malware that were written in the PureBasic language in an attempt to show the difference between functions identified as part of PureBasic libraries and without.