



Information System Security Operation State-of-the-Art in Decompilation and Disassembly

Protection Against Reverse Engineering - Study Of Means Available To Attackers Of Software Protection Schemes

Overview

Reverse engineering is the process of deriving, from a compiled program, a higher-level engineering description of its function, in the form of source code or other specification. Traditionally, the government has sought to protect its software property by limiting access to the powerful computers the software runs on, but cheap computers have made this strategy less effective. If the software is to be protected by other means, it is necessary to have an idea of the means available to reverse engineers and the possible countermeasures to these means.

We performed a study of this critical area and produced a report on the State of the Art in Disassembly and Decompilation. This investigation into the state of the art in decompilation and disassembly was designed to provide comprehensive information relevant to protecting Government software from reverse engineering by a determined, well-equipped adversary equipped with the best tools available.

We developed and reported on a framework for analysis of static and dynamic reverse engineering tools. We then systematically investigated available decompilers and disassemblers for multiple languages and platforms. We evaluated 14 tools in depth, employing a total of over 1000 test cases. We also analyzed six tools that didn't fit into the formal framework in a less structured fashion. These results were described in a second report. Our final report assessed the results of

the investigation and provided an estimate of the tools currently available to adversaries and likely to become available in the 5-7 year time frame.

Research Approach

The main tools for software reverse engineers come from the class of disassemblers and disassembly/decompilation tools. These tools change machine code into assembly language or high-level language, and can be used to provide insight into the execution plan and structure of a program and the algorithms used in the program. As software becomes more sophisticated and improved capabilities for software protection are developed, the importance of disassembly and decompilation tools will increase, since they are the best means the attacker has for locating and assessing software protection techniques as well as analyzing the software being attacked.

We surveyed the field of reverse engineering, including both academic research and practical experience, and created a framework for evaluation of reverse engineering tools. The resulting framework had separate sets of attributes for evaluating static decompilers and disassemblers, dynamic decompilation and program observation tools, and auxiliary tools. We constructed a standard set of test cases, chosen to exercise the features of decompilers and disassemblers, including obfuscated programs. Our framework gave weight to both the theoretical and practical aspects of the field, based on our experience. We published a report describing our evaluation framework and sought comment from tool producers, knowledgeable researchers, and industry colleagues.

This work sponsored by the Air Force Research Laboratory (AFRL), through the Advanced Technology Software Protection Initiative (AT-SPI) program, Contract Number F33615-02-C-1296, with McAfee Research, which is now the Security Research Division of SPARTA.



State-of-the-Art in Decompilation and Disassembly

Protection Against Reverse Engineering - Study Of Means Available To Attackers Of Software Protection Schemes

We evaluated dynamic analysis in addition to decompilation because some of the defenses against static analysis suggested by an evaluation of decompilers alone would be insufficient to defend against dynamic methods. Effective defenses must address both static and dynamic reverse engineering techniques.

Our final report on “State of the Art in Decompilation and Disassembly” drew conclusions about the state of the art from the data observed, including a description of areas where tools are not available, and the extent to which reverse engineering can be hindered by use of countermeasures such as obfuscation. This report includes an evaluation and critique of each approach, a discussion of proven and possible countermeasures for each approach, and an assessment of the computer languages that have the best defensive capabilities against disassembly and decompilation. It also includes an assessment of processor sophistication (i.e., machine instruction set) versus disassembler capability, and a demonstration of pertinent disassembler and decompiler software if it exists. Our reports document our experiments with individual reverse engineering tools and our observations and evaluations, in order to enable other researchers to duplicate demonstrations, experiments, and results and also to verify conclusions. We reviewed and analyzed related

and supporting work, including a comprehensive bibliography and references.

Conclusions

We found that most current decompilation and disassembly tools are designed to deal with well-formed programs, and are intended for use in program maintenance. Elementary countermeasures often cause such tools to fail completely. Existing tools also require highly trained operators to use or understand the output. There are several areas where tools that would assist reverse engineering are not available at all; there is no intrinsic barrier to construction of such tools other than cost.

The tools we examined, however, were built for other purposes than defeating anti-tamper measures. An adversary who wishes to overcome the protection on Government software and either steal the technology embodied in the program, or use the program in an unauthorized way, could use tools that represent the current state of the art for purposes other than those for which they were designed. To achieve greater success, such an adversary would have to build tools specifically designed to counter anti-reverse-engineering measures. In the 5-7 year time frame, we believe that decompilers and disassemblers could be built that are far more capable than those now available, and which are able to overcome most code protection measures.