Reverse Engineering Virtualization Obfuscation

Supervisors: Prof. Dr. Alexander Pretschner, Sebastian Banescu
Email: {alexander.pretschner, sebastian.banescu} @ in.tum.de
Phone: +49 89 289 – 17, 314
Starting date: immediately

Context
Software obfuscation is commonly employed in contexts where the attacker has direct access to the software binary. Obfuscated binaries contain either data or algorithms that their vendors do not wish to disclose. Commercial software vendors employ obfuscation to protect their intellectual property. On the other hand, malware vendors employ obfuscation in order to circumvent signature-based detection techniques used in most anti-virus software nowadays.

Since there are a myriad of obfuscation techniques, this thesis will mainly focus on a particular technique called “virtualization obfuscation” [1]. Virtualization obfuscation transforms a program written for a particular target platform (e.g. x86), into a bytecode program for a random instruction set architecture. This bytecode program is executed on an emulator which interprets the bytecode instructions and executes the corresponding target platform instructions.

Goal
Virtualization obfuscation is deemed difficult to reverse engineer [1], [2], [3], [4]. Some of the challenges of reverse engineering virtualization obfuscation include identification of: the virtual program counter, the bytecode instructions and the emulator code.

The goal of this thesis is to design and implement a technique based on statistical analysis, which given a set of virtualization obfuscated binaries having the same functional behavior, can statically detect the emulator code and the bytecode instructions of these binaries.

Workplan
1. Develop understanding of virtualization obfuscation and known ways of reverse engineering it: [1], [2], [3], [4]
   a. Write state-of-art survey of several reverse engineering techniques.
   b. Write performance analysis of existing techniques.
2. Implement statistical similarity tests for binaries
   a. Choose several statistical similarity measures (e.g. [5], [6]) and apply them on n-grams of bytes of the obfuscated binary files.
   b. The design & implementation decisions must be documented in written form.
3. Evaluate the implemented solution
   a. Generate several virtualization obfuscated binaries using a software tool provided by your advisor.
   b. Apply the implemented statistical similarity tests on the obfuscated binaries.
   c. Analyze and discuss the results in terms of performance of the implementation and strength of virtualization obfuscation against statistical similarity tests.
4. The final thesis document must contain:
   a. Description of the problem and motivation
   b. State of the art survey of reverse engineering virtualization obfuscation
   c. Rationale for choosing certain statistical similarity techniques for implementation
   d. Implementation description
   e. Evaluation of implementation
   f. Conclusions and future work

Deliverables
- Virtual machine able to run a demo of the implementation, including instructions on how to run the demo.
- The VM should also include the source code of the implementation.
- Technical report with comprehensive documentation of the implementation, i.e. design decision, architecture description, API description and usage instructions.
- Final thesis report written in conformance with TUM guidelines.

References