Information Security Project Proposal

Securing Analysis Software from Unauthorized Usage

Brian Malley

CSULB

CECS 579 Information Security

September 11th 2015

Prof. Mehrdad Aliasgari

Motivation: Software companies routinely protect their software from piracy through Digital Rights Management, or DRM. DRM is designed to prevent users from easily using the software without paying. Normally, the impact of a failure of DRM is simply a lost sale, and when sales range in the thousands to million some degree of leeway in terms of security is acceptable. When the software is high value like that used by the government or analysis firms, with license prices reaching into the thousands to millions of dollars, the chance of it falling into the wrong hands is unacceptable.

Problem Statement: Design the security for a package of high value software, with only a few intended users. Since the software is very expensive, it can reasonably be assumed that the adversary trying to crack it will be willing to spend more time and money to crack it. Indeed, hostile governments may attempt to use supercomputers to bypass the security. Here are my assumptions for this project:

1) The adversary has a copy of the software executable

2) The adversary has a large amount of money and computing power available

3) An online server will be used to authenticate usage of the software

4) Messages must be sent between the program and the server to allow usage

5) Since the server must be open to messages, the adversary can prompt it

If a server authentication model is not secure for these assumptions something else will be used.

Proposed Solution: As hinted to earlier, the proposed solution of mine is to have two components. The first is the executable. This software will be intended to be difficult to use without proper authentication. Proper authentication in this case will be determined through a connection to a online server which will verify the software's installation. The software will request authentication over the internet to a server, which will return a message enabling the software if it is determined to be authentic.

Implementation Details: To achieve this I will use C++ since I am most comfortable with that language. Another nonfunctional constraint of my project is my intention to use an open source C++ security library, most likely Crypto++. There will be two code portions, the Server source code and the software executable source code. The high value software package will be a very simple dummy program. MAC will be a very important aspect, as well as the problem of synchronizing timestamps.

Timeline of Project Completion:

First Iteration Security Scheme (block diagram, mathematic proofs) - 10/17

First Iteration Implementation (executable, server code) - 10/24

Second Iteration Security Scheme (improved block diagram, math) - 10/31

Second Iteration Implementation (improved executable, server code) - 11/6

Final Iteration Security Scheme (final block diagram, math) - 11/20

Final Iteration Implementation (final executable, server code) - 12/4

This timeline allows a week for each stage. The first stage is anticipated to have the most work, as everything must be from scratch. The second stage and final stage are anticipated to be more difficult as I attempt to protect the implementation from any attacks.

Final Comments: I don't have a partner, so only my name is on this proposal. Also, a team name and/or logo will be coming forthwith. I work at a small software company which creates probabilistic risk analysis software so I have some personal interest in this topic.