Abstract:
Software obfuscation transforms code such that it is more difficult to reverse engineer. However, it is known that given enough resources, an attacker will successfully reverse engineer an obfuscated program. Therefore, an open challenge for software obfuscation is estimating the time an obfuscated program is able to withstand a given reverse engineering attack. This paper proposes a general framework for choosing the most relevant software features to estimate the effort of automated attacks. Our framework uses these software features to build regression models that can predict the resilience of different software protection transformations against automated attacks. To evaluate the effectiveness of our approach, we instantiate it in a case-study about predicting the time needed to deobfuscate a set of C programs, using an attack based on symbolic execution. To train regression models our system requires a large set of programs as input. We have therefore implemented a code generator that can generate large numbers of arbitrarily complex random C functions. Our results show that features such as the number of
community structures in the graph representation of symbolic path-constraints, are far more relevant for predicting deobfuscation time than other features generally used to measure the potency of controlflow obfuscation (e.g. cyclomatic complexity). Our best model is able to predict the number of seconds of symbolic execution-based deobfuscation attacks with over 90% accuracy for 80% of the programs in our dataset, which also includes several realistic hash functions.

Dewey Dezimalklassifikation (Liste):
000 Informatik, Wissen, Systeme

Kongress- / Buchtitel:
To appear in Usenix Security, 2017

Jahr:
2017

Revied:
ja

Occurences:
Einrichtungen > Fakultäten > Fakultät für Informatik > Lehrstühle der Informatik > Informatik 4 - Lehrstuhl für Software & Systems Engineering (Prof. Pretschner)

· Hochschulbibliographie > 2017 > Fakultäten > Informatik > Informatik 22 - Lehrstuhl für Software Engineering (Prof. Pretschner)

entries: