A Dynamic Graph-based Malware Classifier

Hossein Hadian Jazi, and Ali A. Ghorbani

Faculty of Computer Science, University of new Brunswick

ISCX Information Security Centre of Excellence

UNB

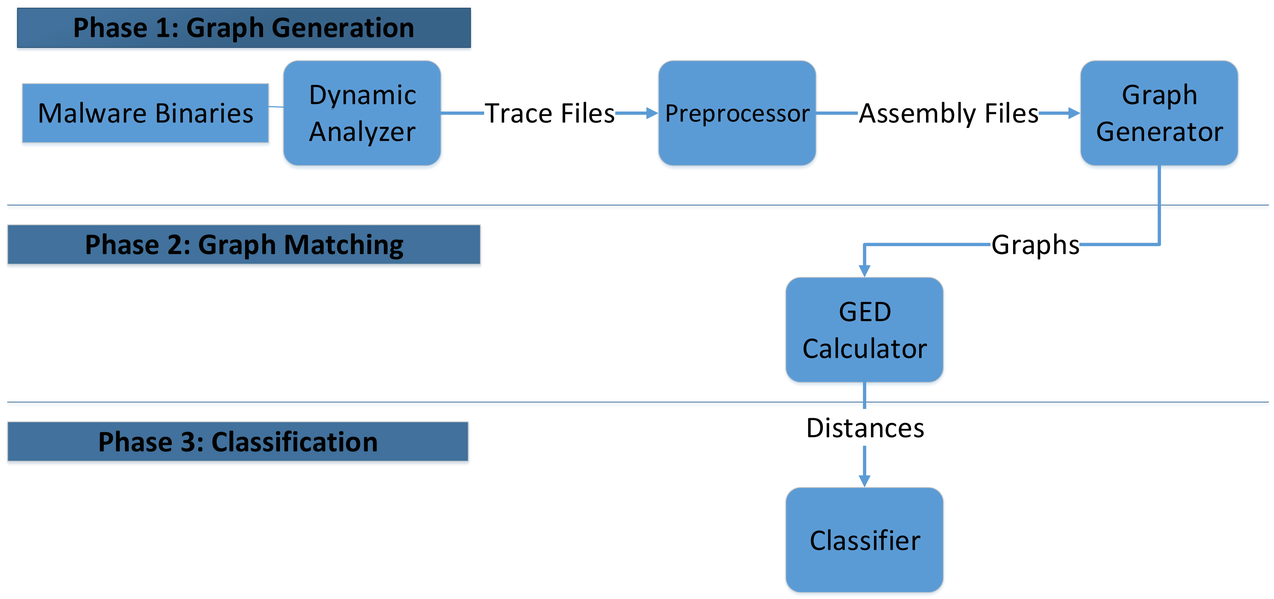
Problem Statement

- The anti-virus industry receives a sheer amount of new malware samples on a daily basis. The prevalence of new sophisticated instances, for most of which no signature is available, coupled with the significant growth of potentially harmful programs have made the adoption of an effective automated classifier almost inevitable.
- Due to the vast majority of obfuscation techniques employed by malware authors, the extraction of a high level representation of malware structure is required. Control flow graphs (CFGs) and function call graphs (FCGs) are the most common abstract representations of an executable. These graphs provide distinctive characteristics of a binary that is identifiable over strains of malware variants.
- Both CFGs and FCGs have been widely used as basic components of most malware detection approaches. However, these methods suffer from the following limitations:
  - Resort to static analysis to generate CFG and FCG graphs mostly by employing PE-Explorer or IDA Pro disassemblers. The static analysis can easily be bypassed in obfuscated cases such as using packers. To alleviate this drawback some of the approaches have used unpacker tools to remove the obfuscated layer of the executable before disassembling it. However, most of them are limited to a fixed set of known packers or are restricted by the fidelity of the emulation environment.
  - Scalability of the approaches is affected by the employed graph comparison algorithm. Unfortunately full-graph comparison and computing the largest common sub-graph, are computationally hard, which makes the scalability of a system questionable.
  - The static approaches cannot handle on-line streams of malicious executables due to their off-line classification or clustering algorithms.

Our contribution

- Employing dynamic analysis to generate dynamic CFG and FCG.
- Devising a new algorithm to generate dynamic FCGs.
- Improving the simulated annealing algorithm to compute similarity between graphs by employing stochastic beam search concept. It improves the accuracy of the classification while maintaining low computational complexity.
- Developing an on-line stream clustering algorithm for clustering streams of FCGs.

Proposed System



Dynamic FCG Generation Algorithm

We devise a new algorithm to generate dynamic function call graphs since static algorithms are not suitable to generate dynamic graphs. The algorithm works as follow:

- For the first instruction, create a function object at the address of that instruction.
- For each call statement or push + ret, create a function object and add an edge from the current function to this new function object.
- For each new call statement, create or reuse a function object and add an edge from the current function to the new or already known function object.
- After a ret instruction, change the current function to the previous one.

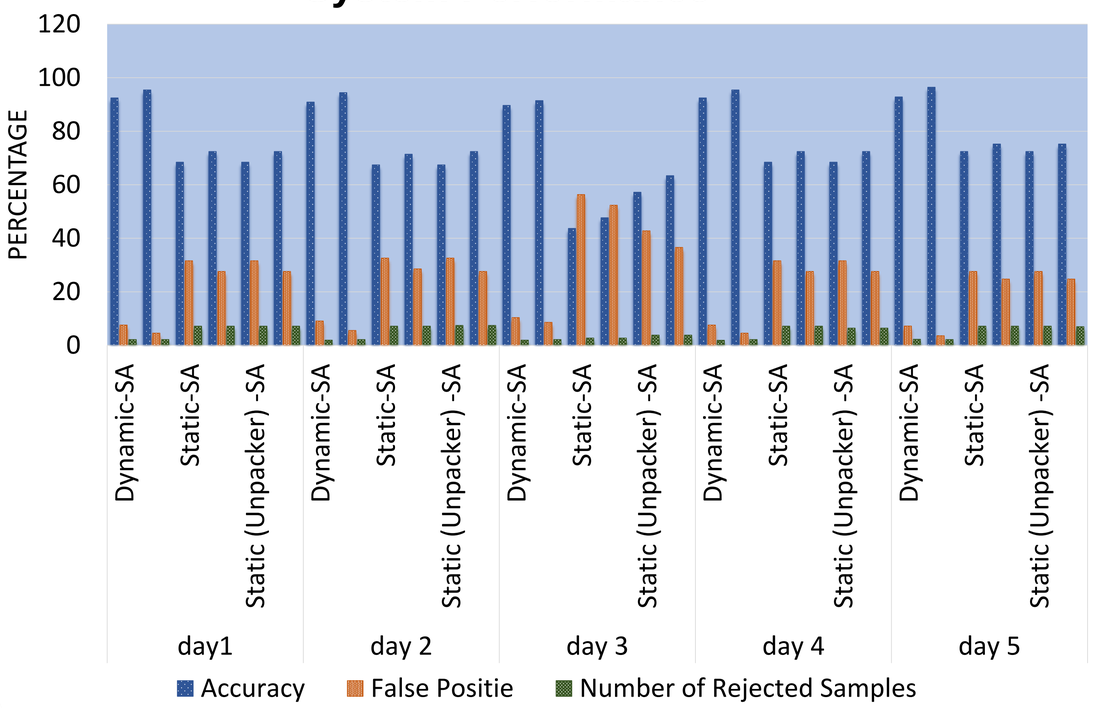
Graph Matching & Classifier

- To make our method scalable, we adapt an graph edit distance approximation algorithm called Simulated Annealing to improve the accuracy of the system while maintaining low computational complexity.
- we follow an approximate classification algorithm called nearest prototype classification which includes three steps: Prototype extraction, clustering and classification. Our classification is applied incrementally and graphs are processed in chunks. This significantly reduces the run-time and memory requirements of our system

Proposed System Evaluation

To evaluate the proposed system, the large number of benign and malicious executable has been collected from five well-known sources: Ether dataset, Malicia dataset, Virus total repository, Virus share and Virus sign. The collected dataset contains 9850 benign executables and 40000 malicious binaries form 346 different families.

System Performance



Comparative Evaluation