www.inl.gov

Idaho National Laboratory

Clustering Analysis of Binaries Across Compiler Optimizations

Micah Flack, M.S. Computer Science – Dakota State University

Mentor: Rita Foster (D520), Jed Haile (C321), Zachary Priest (D520), Michael Custshaw (D520)

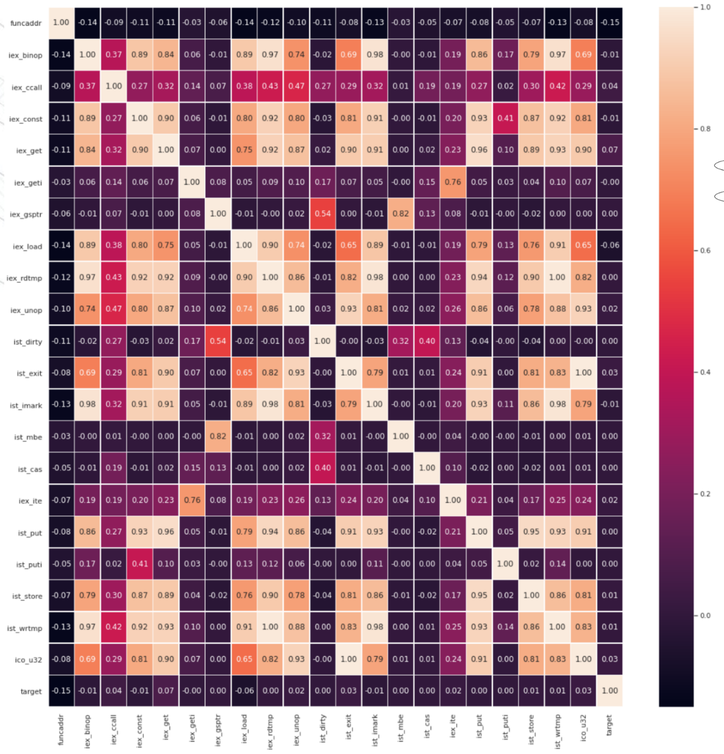
Abstract

One of the biggest issues with intrusion detection/prevention systems (IDS/IPS) today is accurately detecting whether a binary is malicious through means other than hash lists. This poster details some of the clustering analysis methods that can be used to represent the similarity of different binaries using extracted control-flow graph data. While these methods are not typically employed for strict decision-based identification, they aim to assist analysts with graphical representations showing the similarity distance between the categorical groups (e.g. binaries).

Methodology

The features used are VEX commands gathered from control-flow data.

Before knowing which extracted features should be used for modeling, a correlation matrix is created from 22 of the 38 features. This indicates, on a scale from 0.00 to 1.00, the strength of the relationship between any two features. For example, the most interesting feature to predict would be the 'target' – the only other features exhibiting a strong relationship are 'iex_get' or 'iex_ccall'; granted, these are on the lower end of the relationship graph and not as impactful.



With those features chosen the next step was to choose a suitable model for the dataset. The bottom right figure demonstrates using a K-Means clustering algorithm, a method of vector quantization, originally from signal processing to group the two features against each other.

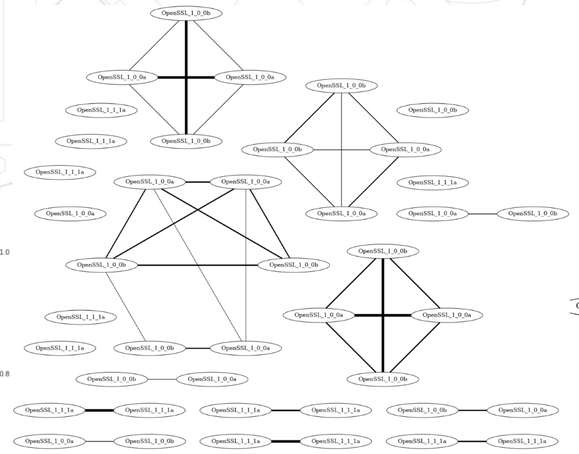
K-Means is then represented as: $$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \arg\min_{\mathbf{S}} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i \ ,$$

a minimization of the cluster sum of squares.

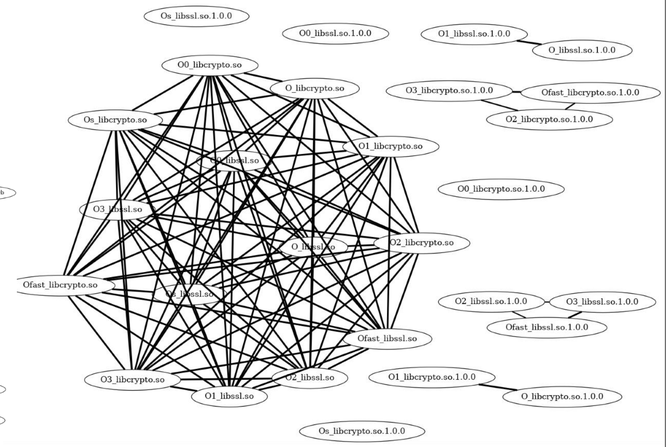
Another method used here was Jaccard-Indexes, also known as Intersection over Union and the Jaccard similarity coefficient, as a statistic used for gauging the similarity and diversity of sample sets. Here it was used to show the similarity between extracted opcodes using Python3 and Objdump.

Created using DOE - Cybersecurity Energy Systems and Emergency Response (CESER) funded Firmware Indicator Translation INL/EXP-20-59338

Results & Conclusion



Jaccard-index similarities w/ opcodes sentences



Jaccard-index similarities w/ string sentences

Immediately the purpose of these graphs may not be clear, however, they can be incredibly useful to researchers and analysts when initially disseminating the data gathered from many samples. Given the similarity in the frequency of opcodes or the pattern of strings, we can represent the shared relationships between identical programs as visual graphs. This information could be easily added to graph databases such as OrientDB or Neo4j for further exporting or use with other models.

When used for strict decision-based identification with a 'Histogram Boosting Classifier', the model achieved an accuracy of ~55% with a training split of 80:20 (training/testing). A higher accuracy could theoretically be achieved with Locality Sensitive Hashing (LSH) that better preserves the semantic structure of the opcodes for similarity-distance comparison.

Binaries modeled:

- OpenSSL
  - libssl.so
  - libcrypto.so
- Ping
- zLib

Compilers used:

- GCC
- Clang

Optimizations

- O0, O1, O2, O3, O, Ofast, Os