Deep Models and Artificial Intelligence for Military Applications:
AAAI Technical Report FS-17-03

Phylogenetic-Inspired Probabilistic Model Abstraction in Detection of Malware Families

Krishnendu Ghosh
Miami University
Hamilton, OH 45011
Email:ghoshk@miamioh.edu

Jeffery Mills
Miami University
Hamilton, OH 45011
Email: millsjw2@miamioh.edu

Joseph Dorr
Miami University
Hamilton, OH 45011
Email:dorrjp@miamioh.edu

Abstract

Lineage of malware has been studied using phylogenetic based methods. Multiple sequence alignment techniques in biology form the foundations of phylogenetic analysis. The analysis of malware trace data using sequence alignment techniques is a drastic simplification from reality. In this work, we describe a framework that incorporates uncertainty in discovering the relationship between malware traces. The framework leverages on probabilistic measures of similarity between stochastic models to compare two malware families. A proof-of-concept of our formalism is demonstrated with the construction of a network of malware relationships.

Introduction

The deluge of malware attacks in recent times requires sophisticated tools for detection. The goal of malware writers is to infect IT infrastructure and create an environment that facilitates malware actions in the systems. The consequences of persistent presence of malware in intrusion of systems, overcoming the antivirus software requires the malware to exist in families. The variants of malware have similar code and often, have similar behavioral characteristics. Malware variants, exhibiting similar behavior are grouped in families. Detection of families of malware is computationally intensive and hence, efficient algorithms are required for analysis. The analogy of mutation of human viruses and morphing of malware such as polymorphic malware (Jang, Brumley, and Venkataraman 2010; Karim et al. 2005) with the operating system is striking. Malware execution traces capture intrinsic behavioral characteristics of the malicious code. Malware detectors based on byte signatures are not effective given the code changes can make the malware undetectable (Canali et al. 2012). Typically, malware writers change the code of a malware and hence, the modified code do have behavioral relationship with the original code. The relationships between the code bases that capture behavioral characteristics encoded in traces are analogous to relationship encoded in biological sequences such as genomic sequences. Therefore, biological sequence analysis methods are promising tools in the analysis of malware traces given the notion of evolution amongst code bases of a malware. A critical step in understanding the evolving threats is to gain knowledge of the processes that are responsible for attacks. Given the behavior of the malicious code is deceptive, reverse engineering on traces is performed to understand the intrinsic processes of attacks. There is a large body of published literature malware detection methods (Idika and Mathur 2007; Christodorescu et al. 2005; Singh and Nene 2013). In this work, we will focus on labeling malware execution traces to predefined groups, malware families based on behavioral characteristics.

Understanding the timeline of malware attacks and identification of latest variant of prior cyber attacks is the key motivation to study lineage of malware. A study representing generic software lineage with details of research problems in software lineage (Jang, Woo, and Brumley 2013). Malware lineage construction is a current research area. A malware family is defined (Anderson, Lane, and Hash 2014) consists of malware instances, have common codebase and have similar behavior during execution. Therefore, the notion of phylogenetic relationships has been explored. An automated construction of malware families and variations (Hayes, Walenstein, and Lakhotia 2009) on phylogenetic models were reported. Recent work addressing malware families have been reported using phylogenetic-insprired methods(Kim, Khoo, and Liò 2012; Carrera and Erdélyi 2004; Khoo and Lió 2011). The construction of phylogenetic relationships in biology is from alignment of gene (biological) sequences (Khoo and Lió 2011). There is no direct mapping with biological sequences with that of sequences of system calls as in malware trace. Biological sequence alignment is scored on the basis of point accepted mutation matrix which is based on protein mutations. The alignment methods used from biology in trace alignment often suffers from lack of additional information.

The phylogenetic analysis from biology is limiting in the study of malware phylogeny, due to code fragment inheritance from different malware instances (Liu, Wang, and Wang 2016). The notion of a known root for phylogeny tree construction of malware is missing. In this work, we focus in evaluation of sibling relationships between the malware families. To the best of our knowledge there has no work that has addressed sibling relationships using malware (dis)-similarity directly from traces without making minimal assumptions.

In this work, we propose an alignment free methodology that compares malware traces and evaluate, if the traces are similar in behavior. The goal of this work to group malware traces in families of malware based on (dis)-similarity values based on behavioral features of malware. The membership in a family of malware for gieasven traces is based on (dis)-similarity between the traces. Formally, we are seeking to answer the following query: *Given two malware traces, T_1 and T_2, are the two traces (dis)-similar by a numeric value of x?* We describe a phylogenetic inspired framework that incorporates incomplete knowledge and measures similarity among malware represented by execution traces.

Background

In this section, we review the literature in sequence alignment in biology, stochastic models and theories of phylogeny in malware analysis. These are different theories that form the foundations of our work.

Sequence Alignment methods in Phylogenetics

Sequence alignment algorithms (Durbin et al. 1998) have been used heavily in bioinformatics to study sequences of DNA, RNA and proteins. Multiple sequences are arranged one below another and similarity between the subsequences is recorded. The evaluation of similarity for each position in the sequences are scored. The similar subsequences are *homologous* and critical in understanding phylogenetic relationships. Mutations rate of an element in a specific position of a sequence when compared with other elements over time are quantified by a *substition matrix*. The point accepted mutations (PAM) (Dayhoff, Schwartz, and Orcutt 1978) and BLOSUM(Henikoff and Henikoff 1992) are substitution matrices based on mutation of proteins. Modeling set of execution traces of malware analogous to biological sequences is challenging because there is no notion of proteins in set of instructions. The drawback of sequence alignment methods is that there can be many sequences and call-ret call standards maybe different for malware writers (Khoo and Lió 2011). Khoo et al describes a method aligning sequences based on functional similarities such as kernel space API calls would be similar across different malware.

Alignment free methods in Phylogenetics

Alignment free methods are important in the construction of phylogenetic trees from sequences (Vinga and Almeida 2003) because they make fewer assumptions to address biological problems such as genome rearrangements, modeling of DNA sequences that have undergone recombination (Höhl, Rigoutsos, and Ragan 2006). There are two categories of alignment free methods that have been proposed (Vinga and Almeida 2003): (i) Statistics on words and information theory concepts and (ii) Kolmogorov complexity and Chaos theory. The effectiveness of alignment free methods in phylogeny tree construction was evaluated (Höhl, Rigoutsos, and Ragan 2006).

Stochastic Model and Similarity Measures

Stochastic models namely, markov chains and markov decisions processes have been used to model biological sequences(Durbin et al. 1998). We define markov chain as a state based system (Baier, Katoen, and others 2008).

Definition 1. *Discrete-Time Markov Chains (DTMC): a discrete-time Markov chains is a tuple: $\mathcal{M}\langle S, S_0, \iota_{init}, \mathcal{P}, L\rangle$ where:*

- *S is a finite set of states.*
- *S_0 is the set of initial states.*
- *$\mathcal{P} : S \times S \rightarrow [0,1]$, where $\mathcal{P}$ represents the probability matrix and $\sum_{s,s' \in S} \mathcal{P}(s,s') = 1$.*
- *$\iota_{init} : S \rightarrow [0,1]$ where $\sum_{s \in S} \iota_{init}(s) = 1$ is the initial distribution.*
- *$L : S \rightarrow 2^{AP}$. where AP the set of atomic propositions.*

The definition of DTMC is stated in terms of state-based. The comparison of two discrete time markov chains (dtmc) have been studied (Rached, Alajaji, and Campbell 2004; Deng et al. 2009). One of the measures that has been used in the analysis of (dis)-similarity of stochastic models is Kullback-Leibler Divergence. Kullback-Leibler divergence (Kullback and Leibler 1951) or relative entropy is a non-symmetric measure between two probability distributions. A numerical expression for computation of KLD was derived for time invariant finite alphabet markov sources of arbitrary initial distributions (Rached, Alajaji, and Campbell 2004). An information theoretic treatment in the construction of a reduced order markov model from a large scale markov model has been reported (Deng et al. 2009). Formally, Kullback-Leibler Divergence (KLD) is defined (Pham and Zuegg 2004): Given P and Q be two probability distributions over the random variable X, the KLD is denoted by $H(P,Q)$ of P with respect to Q is, $H(P,Q)$ =

$$\sum_{x \in X} P(x) log \frac{P(x)}{Q(x)}$$

Clearly, $H(P,Q)$ is not a metric because $H(P,Q) \neq H(Q,P)$. Jensen-Shannon divergence (JSD) is a distance metric (Lin 1991) that is constructed from KLD.

Malware Trace

We define malware execution trace on a set of finite alphabets, Σ.

Definition 2. *A trace, $T = \alpha_0, \alpha_1, \alpha_2, \ldots \alpha_n$ where $n \in \mathbb{N}$ and $\alpha_n \in \Sigma$.*

A *trace* is a program execution sequence and is represented by set of states. The states often represent system calls and traces provide a snapshot of the behavior of the program with respect to the request to the operating system services. A DTMC (Anderson et al. 2011) was generated to model the transitional relationships between call traces. A probability matrix, P, from trace execution sequences for each malware is constructed. The transition matrix of the markov chain $\mathcal{M}_1$,$\mathcal{P}_1$ is compared with transition matrix of another markov chain $\mathcal{M}_2$, $\mathcal{P}_2$, using KLD. A lower value of $H(\mathcal{P}_2, \mathcal{P}_1)$ implies the malware trace represented by $\mathcal{M}_2$ is proximal $\mathcal{M}_1$.

Model

In our formalism, we construct markov chains from trace data. Let markov chains constructed from traces, T_1 and T_2 be given by $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. The set of unique system calls in T_1 and T_2 is represented by γ_1 and γ_2, respectively. The set of unique system calls in T_1 and T_2 is $\Gamma = \gamma_1 \cup \gamma_2$. Given a set of unique system calls, Γ and a set of states, S , a one-one labeling function is denoted by, $L : S \rightarrow \Gamma$. The set of labeled states is denoted by $\mathcal{S}$. The algorithm, $ConstructMarkov$ generates a dtmc, $\mathcal{M}_1$ based on trace data, T_1 on a set of labeled states, $\mathcal{S}$. The algorithm can be executed to construct dtmc, $\mathcal{M}_2$ from the unknown trace, T_2 constructs a dtmc, $\mathcal{M}_2$ on the identical set of labeled states, $\mathcal{S}$. It is critical for the two dtmc to have the same state space–a requirement for computing KLD.

The steps of the algorithm, $ConstructMarkov$ are the following: From the given set of labeled states, $\mathcal{S}$, an edge labeled graph is constructed. The states have outgoing and incoming edges from every other state. There are no self edges. Each transition in the labeled graph constructed from the states from $\mathcal{S}$ is matched with the trace data, T_1. The number of matches found is added to the edge labels which were initialized to 1. The probability on an edge was the ratio of the sum of number of matches found and one to the sum of the total matches of the transitions with the trace from a state,s.

Algorithm 1 $ConstructMarkov$

Input: Set of labeled states, $\mathcal{S}$ and trace, T_1
Output: A Discrete Markov Chain, $\mathcal{M}_1$

1: **for** each $i \in 1, \ldots, \mid \mathcal{S} \mid$ **do**
2: **for** each $j \in 1, \ldots, \mid \mathcal{S} \mid$ **do**
3: **if** $(i \neq j)$ **then**
4: Initialize: $\alpha_{ij} \leftarrow 1$.
5: Construct transitions from γ_i to every $\gamma_j, \gamma_i \xrightarrow{\alpha_{ij}} \gamma_j$, where $\gamma_i, \gamma_j \in \mathcal{S}$.
6: **end if**
7: **end for**
8: **end for**
9: **for** each successive transition, $(s \rightarrow s')$ in T_1 **do**
10: **for** each transition, $(\gamma \rightarrow \gamma')$ **do**
11: **if** $(s = \gamma \wedge s' = \gamma')$ **then**
12: $\alpha_{ij} \leftarrow \alpha_{ij} + 1$
13: **end if**
14: **end for**
15: **end for**
16: **for** each $s \in \mathcal{S}$ **do**
17: P(s_i, s_j) = $\frac{\alpha_{ij}}{\sum_j \alpha_{ij}}$ where $1 \leq j \leq \mid \mathcal{S} \mid$ and $j \neq i$.
18: **end for**
19: $\mathcal{M}_1\langle S, S_0, R, P, L\rangle$.

The algorithm $ConstructMarkov$ terminates after finite number of steps. The for-loop in lines (1)-(2) executes finite number of steps, $\mid \mathcal{S} \mid^2$ The for-loop in lines (9)-(10) executes finite number of steps, $m_1 \cdot m_2$ where $m_1 = \mid \mathcal{S} \mid, m_2$ = maximum length of T_1 and $m_1, m_2 \in \mathbb{N}$. The for-loop in line (16) terminates after $\mid \mathcal{S} \mid$. The order of time complexity of the algorithm is $O(\mid \mathcal{S} \mid^2 + m_1 \cdot m_2 + \mid \mathcal{S} \mid)$.

Theorem 1. *$\mathcal{M}_1$ is a Discrete Markov Time Chain.*

Proof. The labels,α_{ij} on every transition $s_i \in \mathcal{S}$ to s_j is initialized by 1 in line(4) of $ConstructMarkov$. Here, $i \neq j$ and $1 \leq i, j \leq \mid \mathcal{S} \mid$. In lines(9)-(15), the labels are updated based on the frequencies of (s_i, s_j) in T_1. The frequencies of any transition from s, appearing in the set of traces, $\mathcal{T}$ are recorded. The sum of the labels with nonzero values, α_{ij} after the update is, $\sum_j \alpha_{ij}$. The set of weighted labels, $P(s_i, s_j)$ from s_i form a probability distribution, P is given, $P(s_i, s_j) = \frac{\alpha_{ij}}{\sum_j \alpha_{ij}}$ where $1 \leq j \leq \mid \mathcal{S} \mid$ and $j \neq i$. Therefore, $s_i \in \mathcal{S}, \sum_j P(s_i, s_j) = 1$. □

Theorem 2. *Given a probability distibution, P from a state, s in $\mathcal{M}_1$ with probability p on (s, s'), there exists a transition, s, s' in T_1 occuring with a frequency of m.*

Proof. The probability distribution on k transitions from state, s in $\mathcal{M}_1$ is $p_1, p_2, \ldots, p_k$ where $0 < p_k \leq 1, k \in \mathbb{N}, \sum_k p_k = 1$. The frequencies on the set of k transitions from s after using the trace, T_1 are $q_1, q_2, \ldots, q_k$ where $m, q_1, q_2, \ldots, q_k \in \mathbb{N}$. Hence, the occurence of (s, s') is $m = p_k.\sum(q_1 + q_2 + \ldots + q_k) - 1$. □

Simulation

Trace data was used from the publicly available dataset (dat Online accessed 5 May 2017). Five traces were prepared for experiments, namely, Tr_1, Tr_2, Tr_3, Tr_4 and Tr_2. We show the results using pairwise comparisons. In the first step, the distinct set of systems calls from the two traces that were to be compared was extracted. The dtmc for each of the traces was created. The KLD was computed (Rached, Alajaji, and Campbell 2004) and the Jensen-Shannon divergence from the transition matrices of the markov chains constructed from the traces. In the Table 1, KLD(1) and KLD(2) represent trace A being compared to trace B and trace B being compared to trace A, respectively.

Trace	Trace	KLD(1)	KLD(2)	JSD
Tr_1	Tr_2	17.764	19.992	3.87
Tr_1	Tr_3	29.141	26.059	5.561
Tr_1	Tr_4	11.772	9.228	2.272
Tr_1	Tr_5	12.409	9.799	2.412
Tr_2	Tr_3	30.964	22.418	5.217
Tr_2	Tr_4	17.863	12.502	3.031
Tr_2	Tr_5	17.647	12.255	2.988
Tr_3	Tr_4	18.304	21.619	3.896
Tr_3	Tr_5	10.168	21.346	4.001
Tr_4	Tr_5	0.649	0.523	0.141

Table 1: Pairwise comparision of Traces using KLD and JSD

In the Table 1, the KLD(1) and KLD(2) values for Tr_4 and Tr_5 are close. Hence, the JSD value is less. The pairwise comparison of traces, Tr_1, Tr_2 and Tr_3 with traces, Tr_4 or Tr_5 show are similar in the KLD(2) values and JSD values. The values provide inside that similar traces have similar divergence or similarity values with unknown traces.

If there are multiple traces for a family, there would be a dtmc constructed based on the traces from the family. The dtmc would then will be compared to another dtmc for an unknown trace or multiple traces. The KLD and JSD values of the pairwise comparisons would provide insights of similarity or dissimilarity between the two malware families by a quantitative value.

Pruning of the Edges

A pruning mechanism based on the values of the divergences on the graph is demonstrated. Figure 1 shows a graph with nodes labeled with the names of traces and the edge labels are the JSD values of the traces. The reading of the edges is: *The JSD value, e is the edge label on the edge between n_1 and n_2*, where n_1 and n_2 are labeled with traces. Similarly, the directed graph in Figure 2 shows the KLD values on the edge labels. Each direction of the edges represent the KLD values of the pair of traces, represented by nodes. The reading of the directed edge is *The KLD, e is the edge label on the edge directed from n_1 and n_2 when compared n_1 with n_2*. Figure 3 shows only the edges that have value less than 15. The number of edges are significantly less when compared with all the edges in Figure 2. The pruning helps to focus on the relationships among the nodes that are closest, that is the malware traces that more similar than others. In other words, KLD values of the traces are closest if they have similar behavior. The numerical value on the edge labels below which the edges are not pruned in the edge labeled graph is the *threshold*.

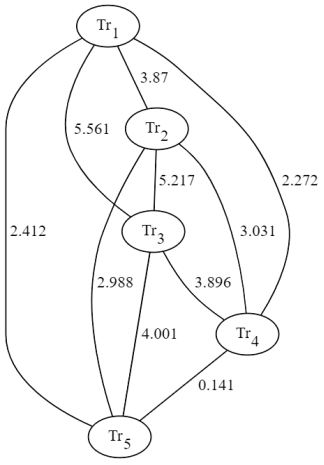


Figure 1: Jenson-Shannon divergence of Five traces

We show the pruning on 150 traces from the dataset. Initially, we construct an edge labeled graph with KLD values on the edges and the nodes representing each traces. The

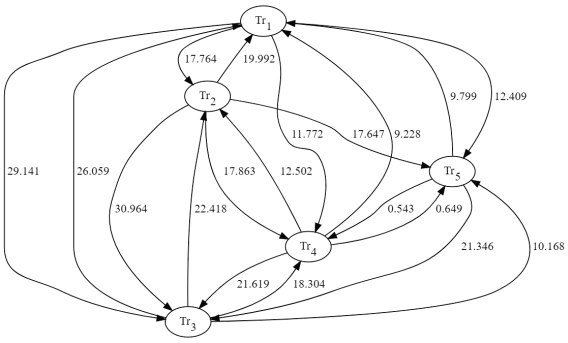


Figure 2: KLD divergence of Five traces

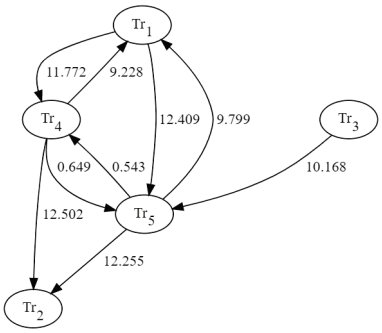
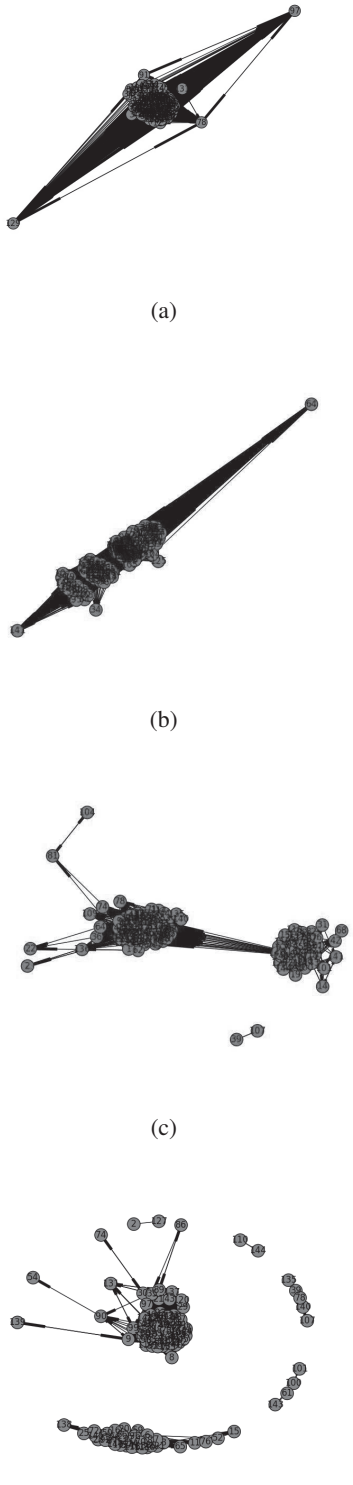


Figure 3: KLD divergence of Five traces with less than 15

KLD is computed by pairwise comparison of the 150 traces. Figure (a)-(d) shows the pruning on the edges with threshold, x where $x = 30, 20, 10, 5$. The smaller the value of the KLD, the shorter are the edges in the Figure . Some of the edges in the Figure (a) are long. As the threshold value decreased, the edges became short denoting that only those edges remained in the graph that were below the threshold. Clearly, the clusters of the traces based on the KLD is prominent with threshold 5.

Conclusion

In this paper, we describe a formalism that is motivated by phylogenetic methods to evaluate the similarity among malware families represented trace data. The KLD rate measures the proximity of an unknown malware trace to a set of given malware families by comparing the dtmc representation from the traces. Currently, the model is implemented and rigorous analysis on trace data is performed to identify the behaviors that are likely to occur in a malware family. The formalism will be modified for efficient analysis for large amount of trace data and comparing the unknown malware trace to a large number of families of malware. We have demonstrated the relationships between the traces in the form of network. The weights on the edges of the network quantify the similarity among the traces represented by nodes. Bayesian approaches on the system calls in the similar traces will be analyzed to understand behavioral characteristics of evolving attacks.



(a)

(b)

(c)

(d)

Figure 4: The pruning of edge labled graph representing pairwise KLD values of 150 traces at different thresholds of KLD. (a) KLD < 30 (b) KLD < 20 (b) KLD < 10 (b) KLD < 5.

References

Anderson, B.; Quist, D.; Neil, J.; Storlie, C. B.; and Lane, T. 2011. Graph-based malware detection using dynamic analysis. *Journal in Computer Virology* 7:247–258.

Anderson, B.; Lane, T.; and Hash, C. 2014. Malware phylogenetics based on the multiview graphical lasso. In *International Symposium on Intelligent Data Analysis*, 1–12. Springer.

Baier, C.; Katoen, J.-P.; et al. 2008. *Principles of model checking*, volume 26202649. MIT press Cambridge.

Canali, D.; Lanzi, A.; Balzarotti, D.; Kruegel, C.; Christodorescu, M.; and Kirda, E. 2012. A quantitative study of accuracy in system call-based malware detection. In *Proceedings of the 2012 International Symposium on Software Testing and Analysis*, 122–132. ACM.

Carrera, E., and Erdélyi, G. 2004. Digital genome mapping–advanced binary malware analysis. In *Virus bulletin conference*, volume 11.

Christodorescu, M.; Jha, S.; Seshia, S. A.; Song, D.; and Bryant, R. E. 2005. Semantics-aware malware detection. In *Security and Privacy, 2005 IEEE Symposium on*, 32–46. IEEE.

Online; accessed 5-May-2017. Csmining group.

Dayhoff, M.; Schwartz, R.; and Orcutt, B. 1978. 22 a model of evolutionary change in proteins. In *Atlas of protein sequence and structure*, volume 5. National Biomedical Research Foundation Silver Spring, MD. 345–352.

Deng, K.; Sun, Y.; Mehta, P. G.; and Meyn, S. P. 2009. An information-theoretic framework to aggregate a markov chain. In *American Control Conference, 2009. ACC'09.*, 731–736. IEEE.

Durbin, R.; Eddy, S. R.; Krogh, A.; and Mitchison, G. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press.

Hayes, M.; Walenstein, A.; and Lakhotia, A. 2009. Evaluation of malware phylogeny modelling systems using automated variant generation. *Journal in Computer Virology* 5(4):335–343.

Henikoff, S., and Henikoff, J. G. 1992. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences* 89(22):10915–10919.

Höhl, M.; Rigoutsos, I.; and Ragan, M. A. 2006. Pattern-based phylogenetic distance estimation and tree reconstruction. *arXiv preprint q-bio/0605002*.

Idika, N., and Mathur, A. P. 2007. A survey of malware detection techniques. *Purdue University* 48.

Jang, J.; Brumley, D.; and Venkataraman, S. 2010. Bitshred: Fast, scalable malware triage. *Cylab, Carnegie Mellon University, Pittsburgh, PA, Technical Report CMU-Cylab-10* 22.

Jang, J.; Woo, M.; and Brumley, D. 2013. Towards automatic software lineage inference. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, 81–96.

Karim, M. E.; Walenstein, A.; Lakhotia, A.; and Parida, L.

2005. Malware phylogeny generation using permutations of code. *Journal in Computer Virology* 1(1-2):13–23.

Khoo, W. M., and Lió, P. 2011. Unity in diversity: Phylogenetic-inspired techniques for reverse engineering and detection of malware families. In *SysSec Workshop (SysSec), 2011 First*, 3–10. IEEE.

Kim, H.; Khoo, W. M.; and Liò, P. 2012. Polymorphic attacks against sequence-based software birthmarks. In *2nd ACM SIGPLAN Workshop on Software Security and Protection*.

Kullback, S., and Leibler, R. A. 1951. On information and sufficiency. *Ann. Math. Statist.* 22(1):79–86.

Lin, J. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory* 37(1):145–151.

Liu, J.; Wang, Y.; and Wang, Y. 2016. Inferring phylogenetic networks of malware families from api sequences. *2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)* 14–17.

Pham, T. D., and Zuegg, J. 2004. A probabilistic measure for alignment-free sequence comparison. *Bioinformatics* 20(18):3455–3461.

Rached, Z.; Alajaji, F.; and Campbell, L. L. 2004. The kullback-leibler divergence rate between markov sources. *IEEE Transactions on Information Theory* 50(5):917–921.

Singh, J., and Nene, M. J. 2013. A survey on machine learning techniques for intrusion detection systems. *International Journal of Advanced Research in Computer and Communication Engineering* 2(11):4349–4355.

Vinga, S., and Almeida, J. 2003. Alignment-free sequence comparisona review. *Bioinformatics* 19(4):513–523.