SAND2013-3273C

Dynamic Information-Theoretic Measures for Security Informatics

Richard Colbaugh
Sandia National Laboratories
Albuquerque, NM USA
colbaugh@comcast.net

Kristin Glass
Sandia National Laboratories
Albuquerque, NM USA
klglass@sandia.gov

Travis Bauer
Sandia National Laboratories
Albuquerque, NM USA
tlbauer@sandia.gov

***Abstract*—Many important security informatics problems require consideration of *dynamical* phenomena for their solution; examples include predicting the behavior of individuals in social networks and distinguishing malicious and innocent computer network activities based on activity traces. While information theory offers powerful tools for analyzing dynamical processes, to date the application of information-theoretic methods in security domains has focused on static analyses (e.g., natural language processing). This paper leverages information-theoretic concepts and measures to quantify the similarity of pairs of stochastic dynamical systems and shows how this capability can be used to solve important problems which arise in security applications. We begin by giving a concise review of the information theory required for our development, and then address two challenging tasks: 1.) characterizing the way influence propagates through social networks, and 2.) distinguishing malware from legitimate software based upon instruction traces of the executables. In each application, case studies involving "real-world" datasets demonstrates the proposed analytic techniques outperform existing methods.**

***Keywords*—**information theory, security informatics, social network dynamics, cyber security, predictive analytics.

I. Introduction

Many important security informatics problems require consideration of *dynamical* phenomena for their solution. For instance, the generation and propagation of influence in social networks is an inherently dynamical process [1], as is the emergence and growth of social and political movements [2], so that understanding and predicting the outcomes of these processes calls for an analysis of the underlying dynamics. Recent work has suggested that the performance of a variety of cyber defense tasks, such as malware and network intrusion detection, can be improved by incorporating aspects of the relevant process dynamics into the analysis [3]. More broadly, it has become apparent that discovering and countering vulnerabilities present in complex infrastructure systems demands analysis of the dynamics of these large-scale interdependent networks [4].

Information theory offers powerful tools for modeling and analyzing stochastic dynamics [5]. However, most applications of information theory in the security informatics domain have focused on *static* problems and data [e.g., 6-8]. For example, a great deal of work has been done to apply analytics from information theory to statistical natural language processing [9]. Other noteworthy research includes [10], which proposes the use of mutual information, an information-theoretic measure, to identify suspect vehicles at border crossing sites. A common theme in much of this work is the use of information-theoretic measures to characterize the similarity of two (static) objects, as mapping the "distances" between pairs of objects in a set is fundamental to a number of analytic tasks (e.g., clustering).

This paper leverages concepts and measures from information theory to quantify the similarity of pairs of stochastic dynamical systems, and shows how this capability can be employed to address important problems in the security informatics domain. We begin by presenting a concise review of the use of information theory to compare statistical objects of various types, culminating with a description of the way the Kullback-Leibler divergence rate can be used to measure the difference between stochastic dynamical systems such as Markov chains. We then illustrate how this simple measure can be used, applying it two important and disparate tasks: 1.) characterizing the propagation of interpersonal influence through social networks, and 2.) distinguishing malware from legitimate software via analysis of the instruction traces of program executables. In each application, empirical analysis of real-world datasets demonstrates that the proposed analytic techniques outperform existing methods. Additionally, the case studies suggest the broad applicability of the proposed approach.

II. Review of Information Theory

This section offers a concise review of the use of information theory to compare statistical objects of various kinds. Readers interested in a more comprehensive discussion of information theory are referred to [5].

The (Shannon) *entropy*

$$H_I = -\Sigma_i\, p(i) \log p(i) \qquad (1)$$

measures the average number of bits needed to optimally encode independent draws of a discrete random variable I which has probability density function p(i). This can be interpreted as a measure of the uncertainty in distribution p(i).

The *Kullback-Leibler divergence*

$$KL_I = \Sigma_i\, p(i) \log (p(i)/q(i)) \qquad (2)$$

measures the excess number of bits, over optimal, needed to encode I if distribution q(i) is used in place of p(i). KL_I can be regarded as a measure of the difference between distribu-

tions p(i) and q(i) (or an inverse measure of the similarity between the two distributions).

The *mutual information* of two random variables I, J with joint probability density function p(i,j),

$$MI_{IJ} = \Sigma_i\, p(i,j) \log (p(i,j)/p(i)q(i)), \qquad (3)$$

measures the excess code needed to represent I, J if it is erroneously assumed that I, J are independent; this follows directly from the definition of KL_I (2). The symmetric measure MI_{IJ} can be regarded as the extent to which knowing I (J) reduces uncertainty concerning J (I).

The measures KL_I and MI_{IJ} are naturally viewed as capturing different aspects of the relationship (e.g., distance) between two *static* distributions. To extend this analysis to stochastic *dynamics,* one can introduce dynamical structure by examining transition probabilities rather than static probabilities. Thus we are led to consider a simple but very useful class of dynamical systems: Markov chains [11].

A *Markov chain* (MC) {Q,P} consists of a state set $Q = \{q_1, \ldots, q_n\}$ and a matrix of transition probabilities $P = [P_{ij}] \in \Re^{n\times n}$ which characterize the chain's dynamics:

$$P_{ij} = Pr(q_{t+1} = j \mid q_t = i, q_{t-1} = k, \ldots) = Pr(q_{t+1} = j \mid q_t = i).$$

A *Markov chain of order k* depends upon additional history:

$$Pr(q_{t+1} \mid q_t^{(k)}) = Pr(q_{t+1} \mid q_t^{(k+1)}) = Pr(q_{t+1} \mid q_t, q_{t-1}, \ldots, q_{t-k+1}).$$

where $q_t^{(k)} = (q_t, q_{t-1}, \ldots, q_{t-k+1})$ denotes the k-dimensional delay vector.

We now recall the notion of Kullback-Leibler divergence rate [5], which can be interpreted as a measure of the difference between two (stationary) MC. The *entropy rate*

$$h_I = -\Sigma_i\, p(q_{t+1}, q_t^{(k)}) \log p(q_{t+1} \mid q_t^{(k)}) \qquad (4)$$

for a stationary MC {Q,P} measures the average number of bits needed to encode one additional state in the MC trajectory. This can be regarded as the growth rate of the dynamics' uncertainty.

The *Kullback-Leibler(KL) divergence rate*

$$kl_I = \Sigma_i\, p(q_{t+1}, q_t^{(k)}) \log [p(q_{t+1} \mid q_t^{(k)}) / p(s_{t+1} \mid s_t^{(k)})] \qquad (5)$$

measures the excess number of bits needed for coding if MC {S,R} is used in place of MC {Q,P}. This can be interpreted as a measure of the difference between {Q,P} and {S,R}. If k = 1 and {Q,P} has stationary distribution π then

$$kl_I = \Sigma_i\, \pi_i\, P_{ij} \log (P_{ij} / R_{ij})$$

where it is assumed that $R_{ij} \neq 0$ if $P_{ij} \neq 0$. Notice that kl_I is a nonparametric measure in that it involves no assumptions about distributions or model structure beyond the basic Markov property.

In what follows we apply the KL divergence rate kl_I to two important and quite different domains: social network analysis and cyber security. In each case the problem is formulated in terms of comparing appropriate MC models and the KL divergence rate kl_I is employed to perform the comparison.

III. Influence in Social Networks

A. Approach

In many security informatics applications, the analyst can observe some subset of the activities of a population of interest (e.g., communications, travel) and wishes to use these observations to identify individuals who are "influential" in some way. For instance, such situations arise in counterterrorism, counterproliferation, transportation security, criminology, and several other domains. A high-level approach to this problem consists of two steps:

1. Use observations of activity to infer a "meaningful" social network $G = \{V,E\}$, in which existence of an edge $(v_1,v_2) = e \in E$ reflects the fact that agent $v_1 \in V$ influences the activity of agent $v_2 \in V$ in some relevant way.
2. Leverage the network G to identify influential individuals, that is, individuals whose behavior affects the activities of many others.

Of course, this procedure must be much more carefully defined to be implementable. To make the approach concrete we use the KL divergence rate kl_I to quantify, for each pair of agents (I,J), the extent to which the activity of I influences or predicts that of J. Given measurements of the activities of I and J over time, denoted $\{i_t, i_{t-1}, i_{t-2}, \ldots\}$ and $\{j_t, j_{t-1}, j_{t-2}, \ldots\}$, it is natural to model I's influence on J via an *influence MC,* defined as $MC_I = \{Q,P,S\}$, where Q and S are the state sets for I and J, respectively, and P is a tensor of transition probabilities:

$$P_{ijk} = Pr(q_{t+1} = j \mid q_t = i, s_t = k).$$

The definition is easily extended to *higher-order influence MC,* for which $P_{ijk} = Pr(q_{t+1} \mid q_t^{(m)}, s_t^{(n)})$.

With these preliminaries in place, we are now in a position to quantify interpersonal influence in a population of interacting agents. We say that agent I *influences* agent J if knowledge of I's activity increases our ability to predict the activity of J. We measure the magnitude of influence from I to J by evaluating the extent to which knowing I's activity increases the predictability of J above the predictability afforded by knowing J's history alone.

This definition is naturally implemented use the KL divergence rate kl_I to characterize the difference between an influence MC model in which I's history affects J and one in which I and J are independent:

$$T_{I\to J} = \sum p(j_{n+1}, j_n^{(k)}, i_n^{(l)}) \log\left(\frac{p(j_{n+1} \mid j_n^{(k)}, i_n^{(l)})}{p(j_{n+1} \mid j_n^{(k)})} \right) \qquad (6)$$

This is the *transfer entropy* (TE) from I to J [12], and measures the excess code needed if it is erroneously assumed that J's behavior is independent of I, conditioned on J's history. Thus $TE_{I\to J}$ quantifies the degree to which I's activity predicts that of J, and gives a measure I's influence on J (indeed, TE is an extension of Granger causality [13]). Notice that TE is not symmetric, so that I may exert more influence over J than J does on I; this matches common experience with social networks.

It is important to test whether a given empirical estimate of TE is significantly different from zero. Here we adopt the simple procedure of comparing the estimated value of TE with the average TE obtained using an ensemble of time series in which the series for I are shuffled [14].

The preceding development can be summarized by stating an algorithm for identifying influential individuals in a population of interacting agents:

Algorithm IA (Influential Agents)

1. Given observations of the sequences of activities of a set of individuals $V = \{v_1, \ldots, v_n\}$ construct the social network $G = \{V,E\}$, where each edge $(v_1,v_2) \in E$ reflects the normalized TE from v_1 to v_2 (specifically, the z-score [15] for the observed TE with respect to the ensemble of time series obtained by shuffling v_1 [14]).
2. For a discrete set of TE z-score values $Z = \{z_1, z_2, \ldots\}$, build a family of social networks $G_z = \{V,E_z\}$ by retaining in G_z only those edges with magnitudes exceeding z.
3. For each social networks $G_z = \{V,E_z\}$, rank the individuals $v \in V$ according to their weighted out-degree. Identify as "influential" those individuals, if any, that achieve a high rank for a significant fraction of the networks G_z.

The performance of Algorithm IA for the task of identifying influential members of a population is illustrated below. It is worth mentioning that this algorithm also can be used as part of a procedure for predicting the behavior of individuals [14].

B. Case Study: Wikipedia Editors

This case study examines the utility of employing Algorithm IA to analyze the social network of editors of Wikipedia (WP), a collectively-authored online encyclopedia with an active user community [16]. In particular, we consider the task of identifying influential editors of WP based upon their past behavior as well as the activity of other editors.

WP provides a rich data source for investigating social influence and its propagation. This study focuses on the collaboration of editors creating Article Pages, which cover one topic per page in the manner of an encyclopedia article and represent the main product of WP. We examined editor collaboration on the Article Pages for Elvis Presley ('Elvis') and 'Anarchism'. These pages were selected for study because each possesses an active user base and they attract different (active) editors. Figure 1 depicts total editing activity for the 'Elvis' Article Page over its lifespan. Observe that the editing activity varies widely over time and often occurs in bursts, which suggests this behavior may be challenging to understand and predict. The editing activity for 'Anarchism' is qualitatively similar.

We collected time series data for all editing activity for the 'Anarchism' and 'Elvis' Article Pages and then restricted attention to the behavior of "active" editors, defined as individuals who made at least three edits to the page under study over the entire lifetime of the page. There were 1218 such editors for the 'Anarchism' page and 1070 for the 'Elvis' page; these individuals make up the vertex sets for the 'Anarchism' and 'Elvis' social networks, respectively (see Algorithm IA).

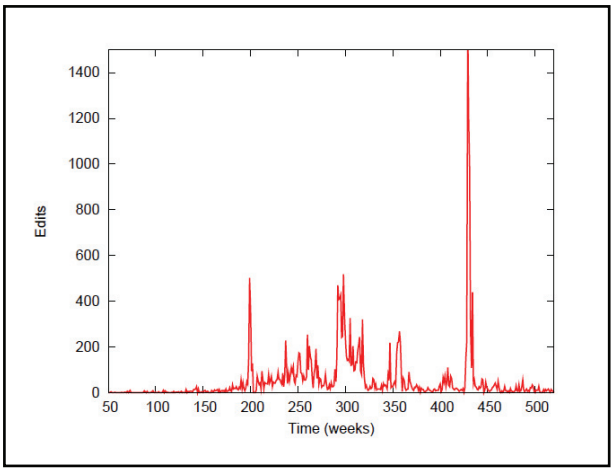

Figure 1. Illustrative example of WP editing behavior. Plot shows total number of edits per week for 'Elvis' Article Page over its lifespan.

We now apply Algorithm IA to the task of identifying influential editors, if any, for the 'Anarchism' and 'Elvis' Article Pages. Figure 2 shows a visualization of the TE graph $G_z = \{V_z,E_z\}$ for the 'Anarchism' page, where the z-score threshold is selected to yield a graph with ~1300 edges. Interestingly, for both the 'Anarchism' and 'Elvis' pages, there exist very small sets of editors who wield substantial influence over other editors, at least as measured by Algorithm IA. Specifically, we find that five editors are the "sources" for a large fraction of the high weight TE edges for the 'Anarchism' page for a range of z-score threshold values. In the case of the 'Elvis' page, there are six such editors. These editors represent a small fraction of the active editors for these pages (~0.5% in each case).

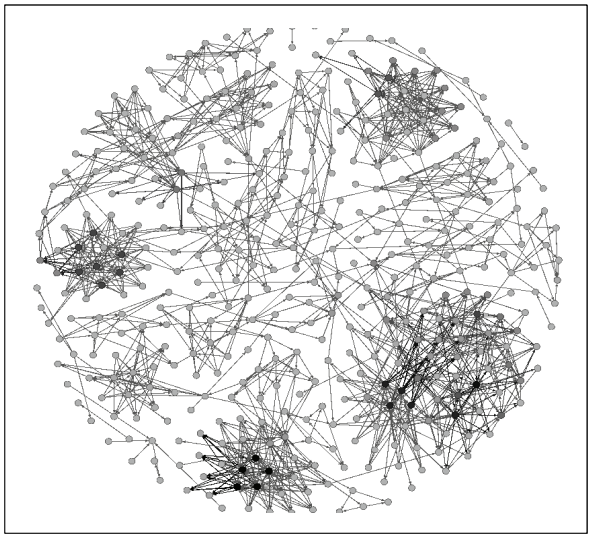

Figure 2. Visualization of TE graph $G_z = \{V_z,E_z\}$ for 'Anarchism' page. Darker vertices have higher out-degree and so are more "influential" according to Algorithm IA.

IV. MALWARE DETECTION

A. Approach

Many security informatics applications involve distinguishing malicious and innocent activities using machine learning (ML) methods [15]. Roughly speaking, ML algorithms automatically learn relationships between observed variables from examples presented in the form of training data; the learned relationships are then used to generate predictions in new situations, i.e., for the test data. ML's capacity to learn from examples, scale to large datasets, and adapt to new conditions make this an attractive approach to predictive analytics in a range of security domains, including cyber defense, transportation security, counterterrorism, and crime prevention.

Consider the problem of using ML to distinguish malware from legitimate executables. Recent work has suggested that including dynamical features of programs, such as program instruction traces, can improve the accuracy of malware detection algorithms [3]. In what follows we derive a simple method for malware classification which uses the KL divergence rate to characterize the dynamics of the executables to be classified.

Suppose we wish to learn a classifier which can accurately predict whether a target executable is malware or a legitimate program based upon the sequence of instructions generated by the executable. A natural way to model an executable's instruction trace is to transform the trace into a Markov chain $MC_E = \{Q_E, P_E\}$ [3]. For a given executable, the state set Q_E is the collection of instructions used in the program and matrix P_E encodes the program dynamics in terms of probabilities of transitions between instructions (as estimated from the sequence of instructions observed in the trace).

Assume a set of labeled training data $\{y_i, \mathbf{s}_i\}^n_{i=1}$ is available, where $\mathbf{s}_i = \{s_i^1, s_i^2, \dots\}$ is the sequence of instructions s^j generated by program i and $y_i \in \{-1,+1\}$ is i's label (legitimate program or malware instance). These data can be expressed as $\{y_i, MC_{Ei}\}^n_{i=1}$, where MC_{Ei} is the MC model for sequence $\mathbf{s}_i$. Finally, the KL divergence rate kl_I can be leveraged to represent these data in a form amenable to machine learning. More precisely, define the "symmetrized" KL divergence rate between two (stationary) Markov chains $MC_J=\{Q,P\}$ and $MC_K=\{S,R\}$ as follows:

$$kl_S = kl_I(J,K) + kl_I(K,J) = \Sigma_i \pi_i P_{ij} \log(P_{ij} / R_{ij}) + \Sigma_i \theta_i R_{ij} \log(R_{ij} / P_{ij}) \quad (7)$$

where π and θ are the stationary distributions for the MC (Q,P) and {S,R}, respectively. The quantity kl_S permits the training data to be modeled as $\{y_i, \mathbf{x}_i\}^n_{i=1}$, where $\mathbf{x}_i \in \Re^n$ is the feature vector for program i and consists of the kl_S-derived distances $\{x_i^1, \dots, x_i^n\}$ between i and each of the other executables (i.e., x_i^j is the distance between MC_{Ei} and MC_{Ej}).

Given training data $\{y_i, \mathbf{x}_i\}^n_{i=1}$, there are numerous methods for learning a classifier which can estimate the class of a *new* instance $\mathbf{x}$ drawn from test data, returning +1 or −1 for malware or a legitimate program [15]. Here we adopt the linear classifier class = sign($\mathbf{w}^T\mathbf{x}$) to perform this prediction task, where $\mathbf{w} \in \Re^n$ is learned using the bipartite graph-based semi-supervised algorithm proposed in [17].

The preceding development can be summarized by stating an algorithm for distinguishing malware from legitimate software:

Algorithm MD (Malware Detection)

1. Given a collection of training data $\{y_i, \mathbf{s}_i\}^n_{i=1}$, where $\mathbf{s}_i = \{s_i^1, s_i^2, \dots\}$ is the sequence of instructions generated by executable i and $y_i \in \{-1,+1\}$ is i's label (legitimate program or malware), transform these data to $\{y_i, MC_{Ei}\}^n_{i=1}$, where MC_{Ei} is the MC model for sequence $\mathbf{s}_i$.
2. Use the symmetrized KL divergence rate kl_S to model the training data as $\{y_i, \mathbf{x}_i\}^n_{i=1}$, where $\mathbf{x}_i \in \Re^n$ is the feature vector for program instance i and consists of the kl_S-based distances $\{x_i^1, \dots, x_i^n\}$ between i and the other executables.
3. Use the training data to learn classifier class = sign($\mathbf{w}^T\mathbf{x}$), where $\mathbf{w} \in \Re^n$ is computed using the bipartite graph-based semi-supervised algorithm proposed in [17].
4. Estimate the class of any new executable $\mathbf{x}$ as either malware (+1) or legitimate (−1) via class = sign($\mathbf{w}^T\mathbf{x}$).

B. Case Study: Malware Classification

This case study examines the performance of Algorithm MD for the malware detection problem. We obtained instruction traces for 780 malicious programs and 776 benign programs from [18]. These traces were collected by running each executable for five minutes using the Xen virtual machine and the Intel Pin program [18]. The resulting traces contain 237 unique instructions, so $|Q_E| = 237$ for each Markov model. (Note that most executables do not make use of all instructions, so the transition probability matrices for these chains included some rows of zeroes.) To permit comparison with a standard n-gram malware detection techniques [19], raw byte information contained in the binary executables was also collected and converted into n-grams for n = 2, 3, 4.

We now apply Algorithm MD to the malware dataset summarized above. The classification accuracy for Algorithm MD, the n-gram technique [19], and a commercial anti-virus program are estimated using ten-fold cross-validation [15]. Sample results are displayed in Figure 3. It is seen that Algorithm MD outperforms both the n-gram method and the commercial anti-virus program for the task of distinguishing malware from legitimate programs. These results indicate the value of leveraging dynamical information for malware detection and of using information theory to represent this information in a manner compatible with the process dynamics.

Malware Detection

method	accuracy
Algorithm MD	91.8%
n-gram	85.1%
commercial AV	75.3%

Figure 3. Sample results for malware detection case study.

V. Concluding Remarks

Many important security informatics problems require consideration of dynamical phenomena for their solution. This paper leverages the Kullback-Leibler divergence rate to quantify the difference between pairs of stochastic dynamical systems such as Markov chains, and shows how this distance measure can be employed to address important security problems. The efficacy of the proposed analytic approach is demonstrated through case studies involving characterization of the propagation of interpersonal influence over social networks and detection of malware via analysis of instruction traces of program executables. These case studies indicate that the proposed approach may be applicable to a broad range of domains. Indeed, preliminary investigation indicates that Algorithm IA may be used as part of a procedure for predicting the behavior of individuals in a social network [14]. Future work will explore this possibility more systematically.

Acknowledgements

This work was supported by the Laboratory Directed Research and Development Program at Sandia National Laboratories.

References

[1] Bakshy, E. et al., "Everyone is an influencer: Quantifying influence on Twitter", *Proc. WSDM '11,* Hong Kong, February 2011.

[2] Colbaugh, R. and K. Glass, "Early waring analysis for social diffusion events", *SecurityInformatics,* Vol. 1 (18), 2012.

[3] Anderson, B. et al., "Graph-based malware detection using dynamic analysis", *J. Computer Virology,* Vol. 7, pp. 247-258, 2011.

[4] Lewis, T., *Critical Infrastructure Protection in Homeland Security*, Wiley, Hoboken, 2006.

[5] Cover, T. and J. Thomas, *Elements of Information Theory*, Second Edition, Wiley, Hoboken, 2006.

[6] *Proc. 2010 IEEE International Conference on Intelligence and Security Informatics*, Vancouver, BC Canada, May 2010.

[7] *Proc. 2011 IEEE International Conference on Intelligence and Security Informatics*, Beijing, China, July 2011.

[8] *Proc. 2012 IEEE International Conference on Intelligence and Security Informatics*, Washington, DC USA, June 2012.

[9] Manning, C. and I. Schutze, *Foundations of Statistical Natural Language Processing*, The MIT Press, Cambridge, 1999.

[10] Kaza, S., Y. Wang, and H. Chen, "Enhancing border security: Mutual information analysis to identify suspect vehicles", *Decision Support Systems,* Vol. 43, pp. 199-210, 2007.

[11] Bremaud, P., *Markov Chains*, Springer, New York, 1999.

[12] Schreiber, T., "Measuring information transfer", *Physical Review Letters,* Vol. 85, pp. 461-464, 2000.

[13] Barnett, L., A. Barrett, and A. Seth, "Granger causality and transfer entropy are equivalent for Gaussian variables", *Physical Review Letters,* Vol. 103, 238701, 2009.

[14] Colbaugh, R., K. Glass, and T. Bauer, "Information-theoretic measures for security informatics", *Homo Collectivus* Talk, Sandia National Laboratories, December 2012.

[15] Hastie, T., R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Second Edition, Springer, New York, 2009.

[16] Bauer, T., R. Colbaugh, K. Glass, and D. Schnizlein, "Use of transfer entropy to infer relationships from behavior", *Proc. CSIIRW '12*, Oak Ridge, TN, November 2012.

[17] Glass, K. and R. Colbaugh, "Estimating the sentiment of social media content for security informatics applications", *SecurityInformatics,* Vol. 1 (3), 2012.

[18] Anderson, B., Personal communication, Summer 2012.

[19] Krishna, D. et al., "New malicious code detection using variable length n-grams", in *Information System Security,* LNCS 4332, Springer, Berlin, 2006.