Ranked Leakage-Sensitive Hashing

Rohith Prakash

Abstract

In this paper, we study the problem of real-time anomaly detection of intelligent malware samples disguised within benign applications. We demonstrate that commonly used Dynamic Time Warping (DTW) distance is not suitable on time series of system resource traces when malware samples dynamically adapt their behavior to evade detection. To deal with malware samples in real-time which attempt to hide within benign behavior, we propose a new LSH-based scheme that has low hardware and complexity overhead, iteratively hashes time series based on *behavioral patterns*, triangulates hashing buckets to better categorize behavior, and is able to categorize new, previously unobserved behavior.

1 Introduction

Time series are used almost ubiquitously to represent time-based measurements in various fields. However, it is a well known problem that when these times series represent behavior-related observations of complex systems, unintended information about the system may be leaked through this channel [1, 2]. This problem of side channel leakage has been extensively studied in the past by, and others have proposed correlation-based measures to quantify the amount of leakage [3, 4].

In this paper, we consider the problem of efficiently learning and classifying the behavior of side channels through observed time series using probabilistic hashing techniques. We formally define the probability of information leakage on a set of such time series traces observed at a fixed granularity with respect to a distance measure. By expanding on that intuition, we propose a *ranked leakage-sensitive hashing* scheme based on previous locality-sensitive hashing schemes [5, 6, 7] that exploits the probability of information leakage for nearest neighbor computations in anomaly detection and behavior classification. We believe that these contributions lead to a novel, strong characterization of leakage over side channels, giving rise to a new notion of *dimensionality* of a channel. We show that this can determine the effectiveness of leakage prevention schemes as well as discuss possibilities of projecting onto higher and lower dimensional spaces to improve anomaly detection or improve the effectiveness of leakage prevention schemes.

2 Background and Definitions

2.1 Information Leakage

Side channel exploitation, anomaly detection, and covert channel communication are problems of detecting or exploiting leakage over information channels. Side and covert channels exists when observable differences in system behavior occur as the result of actions performed by a *victim* or sending process. These attacks typically involve an adversary learning secret information over the channel, based on the behavior of a victim process. Anomaly detection, on the other hand, involves detectors running on a system analyzing and categorizing observed behavior in real time. In this setting, a malicious program leaks information about its behavior through an observed channel.

We consider a single information channel as a sequence of observations of a system resource — a *time series* of resource observations. For example, the *trace* of system calls on a system over time is an n-dimensional time series, where each observation determines the number of times each of the n system calls was invoked.

Our primary observation with regards to time series leakage is that information may only be learned from time series observations if the underlying distributions are distinguishable.

Definition 1 ***Distribution-based leakage:*** *Consider two distinct program behaviors x and x' and resulting time series for each behavior drawn from D_x and $D_{x'}$ respectively. Let $t(x)$ and $t(x')$ be two time series resulting from behaviors x, x', drawn from $D_x, D_{x'}$ respectively. Observing $t(x)$ and $t(x')$ can only leak information about x and x' if D_x, $D_{x'}$ are statistically distinguishable.*

While Definition 1 is useful when one can carefully observe many samples from different distributions to assess the distinguishability of the underlying distributions, it is difficult to use in practice. Instead, we propose a slightly different definition of leakage:

Definition 2 ***Time series leakage:*** *Consider two distinct program behaviors x and x' with output distributions D_x and $D_{x'}$, and let $d(\cdot,\cdot)$ be a distance function on space of output time series. Let $T_x = \{t_k\}$ be a sequence of time series drawn from D_x. Define r as the minimum distance such that $d(t_i, t_j) \leq r \ \forall \ t_i, t_j \in T_x$*

We say that observing time series $t(x)$ and $t(x')$ can ***leak information with respect to*** *$d(\cdot,\cdot)$ about x and x' if $d(t(x), t(x')) > r$.*

Definition 2 describes time series leakage with respect to a specific distance function applied on observation points. If time series resulting from behavior x can be separated from time series from x' by a distance of more than r, there is potential information leakage through this channel.

2.2 Hashing

Hashing has long been used as a method of easing the curse of dimensionality for tasks such as clustering on a large set of high-dimensional data [8, 9, 10]. Exploiting the probabilistic nature and the computational efficiency of hashing enables approximations to difficult high-dimensional problems quickly and in real-time.

Consider the space of time series S and a distance function d on S. A LSH family is defined as such:

Definition 3 ***Hash family:*** *A hash family $\mathcal{H} = \{h : S \to U\}$ is called (r_1, r_2, p_1, p_2)-sensitive w.r.t. $d(\cdot,\cdot)$ if for any $x, y \in S, h \in \mathcal{H}$*

(i) If $d(x,y) \leq r_1$, then $\Pr[h(x) = h(y)] \geq p_1$

(ii) If $d(x,y) \geq r_2$, then $\Pr[h(x) = h(y)] \leq p_2$

Such a family is only interesting if $p_1 > p_2$. To increase the effectiveness of an LSH technique, the gap between p_1 and p_2 may be *amplified*:

Definition 4 ***LSH Amplification:*** *Consider a (r_1, r_2, p_1, p_2)-sensitive hash family $\mathcal{H}$ w.r.t $d(\cdot,\cdot)$. The LSH hash family can be amplified in the following ways:*

(i) ***AND construction:*** *Define $\mathcal{H}' = \{h' : S \to U^r\}$ such that $h' = [h_1, \ldots, h_r] \subset \mathcal{H}$. $h'(x) = h'(y)$ iff $h_i(x) = h_i(y) \ \forall \ h_i \in h' \ \mathcal{H}'$ is a (r_1, r_2, p_1^r, p_2^r)-sensitive LSH family*

(ii) ***OR construction:*** *Define* $\mathcal{H}' = \{h' : S \to U^b\}$ *such that* $h' = [h_1, \ldots, h_b] \subset \mathcal{H}$. $h'(x) = h'(y)$ *iff* $h_i(x) = h_i(y)$ *for any* $h_i \in h'$ $\mathcal{H}'$ *is a* $(r_1, r_2, 1-(1-p_1)^b, 1-(1-p_2)^b)$*-sensitive LSH family*

(iii) ***AND-OR composition:*** *The composition of* and *with* or *constructions defines a* $(r_1, r_2, 1-(1-p_1^r)^b, 1-(1-p_2^r)^b)$*-sensitive LSH family*

We consider hash families $\mathcal{H} = \{h : S \to \mathbb{R}_+^n\}$ such that $h \in \mathcal{H}$ approximates $d(\cdot,\cdot)$ on time series in S. Intuitively, time series which are "closer" to each other (as defined by d) will be harder to distinguish by any arbitrary classifier and thus present fewer possibilities for information leakage. We expand on this intuition by considering a hashing family $\mathcal{H}$ and hashing function $h \in \mathcal{H}$ which has the following property $\mathscr{A}$:

Definition 5 ***Property*** $\mathscr{A}$*: Two hash families* $\mathcal{H}_{r_1}, \mathcal{H}_{r_2}$ *(*$r_1 \leq r_2$ WLOG*) have property* $\mathscr{A}$ *if they are* (r_1, r_1, p_1, q_1) *and* (r_2, r_2, p_2, q_2)*-sensitive respectively and the following holds: If* $h_{r_1}(x) = h_{r_1}(y)$*, then* $\Pr[h_{r_2}(x) = h_{r_2}(y)] \leq \max(1-q_1, 1-q_2)$

Note that the function $d(\cdot,\cdot)$ used in constructing an LSH family need not be a proper metric, but property $\mathscr{A}$ requires that $d(\cdot,\cdot)$ be sub-additive.

Definition 5 allows for a stratified LSH scheme using a set of LSH families index by a distance r which allows us to confer a notion of closeness between buckets of a lower LSH strata.

Definition 6 ***Ranked LSH families****: A set of hash families* $\{\mathcal{H}_r\}_{r \in R}$ *is a ranked LSH family if any two* H_{r_1}, H_{r_2} *have property* $\mathscr{A}$ $\forall$ $r_1, r_2 \in R$*. Denote such a set of families as* (R, p_1, p_2)*-sensitive LSH families* $\mathcal{H}_R$*.*

Proposition 7 *There exists* (R, p_1, p_2)*-sensitive ranked LSH families* $\mathcal{H}_R = \{\mathcal{H}_r\}_{r \in R}$ *such that the following hold:*

(i) $h \in \mathcal{H}_r \in \mathcal{H}_R$ *collides with high probability if* $y \in B_r(x)$*: If* $d(x,y) < r, \Pr[h(x) = h(y)] \geq p_1$*.*

(ii) $h \in \mathcal{H}_r$ *has few false collisions: If* $d(x,y) > r$*,* $\Pr[h(x) = h(y)] < p_2$*.*

(iii) Property $\mathscr{A}$ *holds for any two families* $\mathcal{H}_{r_1}, \mathcal{H}_{r_2} \in \mathcal{H}_R$*.*

Proposition 7 corroborates the notion of closeness to collision probabilities, which allows for the grouping of similar time series. Due to Property $\mathscr{A}$, we may apply iterative hashing scheme to *rank* the probabilities of closeness based on the varied parameter r. We discuss this in greater depth in Section 3.1

2.3 Distance Measure

Denote the space of a time series as S, and define a function d:

$$\begin{aligned} d\colon S^2 &\to \mathbb{R}_+ \\ (x, y) &\mapsto r \end{aligned} \tag{1}$$

Where d maps two time series to a non-negative real number that represents some notion of distance between them. Additionally, require d to be sub-additive:

$$d(x,y) \leq d(x,z) + d(z,y) \tag{2}$$

The function $d(\cdot, \cdot)$ defines leakage in our threat model. To give intuition behind this, we consider an arbitrary classification attack on a set of time series. Applying Definitions 1 and 2, there is potential for leakage by observing the resulting time series if, for some function d, there exists separation by $d(\cdot, \cdot)$ between time series of differing classes.

Previous work in time series anomaly detection has largely focused on Euclidean distance and Dynamic Time Warping (DTW) distance. However, we note that there are drawbacks of limiting evaluation to these two measures only. DTW is not a proper metric as it is not sub-additive; this is a result of DTW treating one time series as non-linear time-stretched of the other. In the context of anomaly detection, this limits the ability to mark a small, unexpected change in behavior when compared with other distance measures. Additionally, both of these measures operate under the assumption that observed time series which are similar will be of the same scale. Two time series which exhibit similar "behavior" but take values of a slightly different scale will not be marked as being similar by either measure. Changes in background system activity could therefore affect the ability of these two measure to detect true anomalous activity.

2.4 Kernel Transforms

We have so far defined leakage with respect to an arbitrary, but fixed, distance measure. However, we now consider kernel transforms to define higher dimensional distance measures without explicitly defining the embedding space [11]. This methodology allows us to determine a leakage-sensitive distance measure with computational efficiency.

Kernel transforms have been used extensively in machine learning problems, especially in support vector machine (SVM) classifiers. For example, a kernel transform allows the use of user-specified similarity functions that may be computationally intractable to fully define. However, kernel transforms have also been recently applied to hashing problems in order to tackle even higher dimensional similarity problems [5, 12, 6]. A kernel function $\kappa(\cdot, \cdot)$ thus defines a new similarity measure on a higher dimensional space over which we would not otherwise be able to efficiently hash.

The following definitions let us formally define kernel transforms on time series:

Definition 8 ***Hilbert space****: A vector space H over a field F with an inner product $\langle \cdot, \cdot \rangle_H : H \times H \to F$ that also defines a complete*[1] *metric space is called a Hilbert space.*

The key property of Hilbert spaces we wish to leverage is the norm induced by the inner product $\langle \cdot, \cdot \rangle_H$. This inner product define the higher order distance measure we wish to use on the raw observation space.

Definition 9 ***Kernel transform****: Let X be an arbitrary space and H be a Hilbert space with inner product $\langle \cdot, \cdot \rangle_H$. $\kappa(\cdot, \cdot) : X \times X \to H$ is a kernel transform if $\kappa(x, y) = \langle \phi(x), \phi(y) \rangle_H$ for some $\phi : X \to H$.*

Note that in Definition 9, the mapping function ϕ need not be explicitly defined. In fact, κ being a positive-semidefinite function (or matrix over discrete spaces) implies the existence of a satisfactory function ϕ. Thus, we can consider arbitrary higher-order distance measures as any positive-semidefinite mapping $\kappa : X \times X \to H$ guarantees a similarity measure in H.

[1] A space X is complete if every Cauchy sequence converges in X. A Cauchy sequence is a sequence $\{x_n\}_{n \in \mathbb{N}}, x_n \in X$ with $\lim_{(m,n) \to \infty} |x_m - x_n| = 0$.

Definition 10 ***Reproducing kernel Hilbert space (RKHS)****: Let H be a Hilbert space of real-valued functions on an arbitrary set X. H is a reproducing kernel Hilbert space if there exists a* ***reproducing kernel****, $\kappa_x \; \forall x \in X$, where $f(x) = \langle f, \kappa_x \rangle_H \; \forall \; f \in H$.*

Note that $\kappa(x, y) = \langle \kappa_x, \kappa_y \rangle_H$, and thus the kernel transform in Definition 9 defines a RKHS. We have thus demonstrated that we can consider arbitrary higher-order similarity measures using kernel functions on the space of observed samples.

Definition 11 ***Similarity measure****: A kernel function $\kappa(\cdot, \cdot)$ is a similarity measure on X if $\kappa(x, x) = 1$ for any $x \in X$ and $\kappa(x, y)$ decreases as $d(x, y)$ increases for some distance measure d.*

We are only interested in kernel functions which are similarity measures, as arbitrary kernel transforms without this structure are not useful for the purposes of constructing higher-order distance measures.

3 Kernelized Hashing Model for Time Series

We now propose a specific hashing model for time series with the goal of anomaly detection in mind.

Consider a set of n samples $D = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ with similarity measure $\kappa \colon D \times D \to \mathbb{R}$ defined by $\kappa(x, y) = \langle \phi(x), \phi(y) \rangle_H$, with H being a RKHS. We now consider the effects of a positive-semidefinite similarity measure $\kappa(\cdot, \cdot)$ as well as its corresponding distance measure $\tilde{\kappa}(x, y) = \|\phi(x) - \phi(y)\|_H$, the norm induced by the RKHS. Note that from Definition 9, we do not need to explicitly define the mapping $\phi(\cdot)$ to an RKHS.

Definition 12 ***Kernelized LSH****: $\mathcal{H}_R = \{\mathcal{H}_r\}_{r \in R}$ is a (R, p, q)-sensitive ranked, kernelized LSH family if, for any $x, y \in D$:*

(i) If $\tilde{\kappa}(x, y) \leq r, \Pr[h_r(x) = h_r(y)] > p$

(ii) If $\tilde{\kappa}(x, y) \geq r, \Pr[h_r(x) = h_r(y)] < q$

(iii) Property $\mathscr{A}$ holds for any two LSH families $\mathcal{H}_{r_1}, \mathcal{H}_{r_2} \in \mathcal{H}_R$

Denote such a family a ***RKLSH*** *family.*

We now consider how to apply a RKLSH family to time series for anomaly detection and construct such a family in Section 4.

3.1 Ranked Hashing by Iteration

Let $\mathcal{H}_R$ be a RKLSH family which approximates a measure on time series distance with R being a set of stratifying distance thresholds, and let S be the space of all possible time series.

Definition 13 *Let X be an input space and $\mathcal{H}_R$ an RKLSH, with $R = \{r_1, r_2, \ldots\}$. Two data samples $x, y \in X$ have* ***rank*** *n if $n = \inf_n \{r_n \in R | h_{r_n}(x) = h_{r_n}(y)\}$.*

By applying this tiered nearest neighbor calculation, we are able to *rank* the similarity of two elements by examining the threshold at which their hashes collide.

3.2 Anomaly Detection on time series

What this theory describes is that similar time series may be grouped together and ranked by their closeness via a kernel distance measure $\tilde{\kappa}$ [13]. In the context of anomaly detection, we can apply this scheme to a set of *normal* traces (time series of utilization, label encoded syscalls, etc.) to determine what thresholds and clusters constitute normal execution with a finer granularity. Instead of a binary label of *normal* vs *anomalous*, we can instead stratify applications by types of execution. We speculate that this makes our detection algorithm more robust to adversarial interference.

Given the initial LSH clustering of normal traces, we can continuously apply the hashing scheme to testing traces over a rolling window to categorize their behavior. Traces which do not match enough previously "normal" clusters over the set of thresholds may be considered to represent anomalous activity.

The streams of time series our LSH algorithm considers have indeterminate length, so we consider fixed fixed observation periods and hash time series segments. However, performing this real-time anomaly detection still leaves the issue of a malicious program changing behavior to match different benign programs during different observation periods. Unless such behavioral changes were present in the original training set, this behavior should be considered anomalous.

To combat behavioral changes in malicious programs, we randomize the observation periods for each distance threshold r_i in the LSH algorithm. This guarantees that such behavior changes will be observed during the hashing process for some distance thresholds.

3.3 Unsupervised, Intelligent Adversary

A similar approach to Section 3.2 can be applied without a predetermined training set for time series classification through clustering. Adjusting the set of threshold values confers the notion of closeness, which may be used to determine which sets of output traces resulted from the same input to a program, for example.

4 Proposed Scheme

Consider the Minkowski distance:

$$M(x, y) = \left(\sum_k |x_k - y_k|^p\right)^{1/p}$$

For $p \geq 1$, $M(\cdot, \cdot)$ is a measure due to Minkowski's inequality [14, p. 190]. Additionally, for $p = 2$, this is the standard Euclidean distance. Like the Euclidean distance, this distance metric fails to account for phase and shape changes between time series.

To this end, we consider two different measures based on a p-dimensional Minkowski distance. The first is due to Batista et al. [15] which attempts to account for time series complexity and has been shown to greatly improve the mean accuracy rates of time series comparison [16]. Consider the following *complexity measure*:

$$C(x) = \sqrt{\sum_i (q_i - q_{i+1})^2} \tag{3}$$

Using this complexity measure, we define a *complexity-invariant* distance measure based on the Minkowski distance $M(\cdot, \cdot)$:

$$d(x, y) = M(x, y) * \frac{\max(C(x), C(y))}{\min(C(x), C(y))} \tag{4}$$

The second measure we consider attempts to correlate Minkowski distance across time with an additional penalty term for time series which are out of phase:

$$d(x, y) = \min_{i,j<\frac{n}{2}} \{M(x[i:N], y[j:N]) + c * (i + j)\} \tag{5}$$

where x, y have length n and $N = n - \max(i, j)$. c is a scalar penalty factor for out of phase alignment between x and y.

For both of these measures, we consider p-dimensional Minkowski distance where the original time series have length p.

4.1 Proposed Hash

Let S be the space of n-point time series to be processed. Pick a random lines in $\mathbb{R}^2$ and project each point x_i onto each line h_k. Each line will be partitioned into buckets of size $\frac{r}{n}$ (here, we see the parametrization of the hash). In practice, we fix a number M (say, 2^{32}), and apply $h_r(x) = \frac{\pi(x)*n}{r} \bmod M$.

Points are considered to intersect if they hash to the same bucket on a fixed number of hash function. We propose that the number of hash functions on which points must collide to determine intersection be a tunable parameter. Time series which intersect on more than $\frac{n}{2}$ points (in time order) are said to be candidate pairs for r-closeness.

5 Key Takeaways

- Use hash that approximates time series distance.
- Compose multiple kernels or hash families to obtain ranked "normality" measure.
- Apply ranking to classification by an unsupervised classifier to determine amount of leakage possible without prior knowledge.
- Apply ranking to normality (anomaly) detection.
- Perform continuous hashing on rolling windows of execution across channels of multiple resources to classify activity in real-time.
- To be considered: Overlay results with ShapeGD.

References

[1] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9-13, 2009*, pages 199–212. ACM, 2009.

[2] Yinqian Zhang, Ari Juels, Alina Oprea, and Michael K. Reiter. Homealone: Co-residency detection in the cloud via side-channel analysis. In *32nd IEEE Symposium on Security and Privacy, S&P 2011, 22-25 May 2011, Berkeley, California, USA*, pages 313–328. IEEE Computer Society, 2011.

[3] John Demme, Robert Martin, Adam Waksman, and Simha Sethumadhavan. Side-channel vulnerability factor: A metric for measuring information leakage. In *Proceedings of the 39th Annual International Symposium on Computer Architecture*, ISCA '12, pages 106–117, Washington, DC, USA, 2012. IEEE Computer Society.

[4] Tianwei Zhang, Fangfei Liu, Si Chen, and Ruby B. Lee. Side channel vulnerability metrics: The promise and the pitfalls. In *Proceedings of the 2Nd International Workshop on Hardware and Architectural Support for Security and Privacy*, HASP '13, pages 2:1–2:8, New York, NY, USA, 2013. ACM.

[5] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1092–1104, June 2012.

[6] Ke Jiang, Q. Que, and B. Kulis. Revisiting kernelized locality-sensitive hashing for improved large-scale image retrieval. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4933–4941, June 2015.

[7] Y. B. Kim, E. Hemberg, and U. M. O'Reilly. Stratified locality-sensitive hashing for accelerated physiological time series retrieval. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2479–2483, Aug 2016.

[8] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM.

[9] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99, pages 518–529, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[10] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG '04, pages 253–262, New York, NY, USA, 2004. ACM.

[11] Bernhard Schölkopf. The kernel trick for distances. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'00, pages 283–289, Cambridge, MA, USA, 2000. MIT Press.

[12] D. C. Kale, D. Gong, Z. Che, Y. Liu, G. Medioni, R. Wetzel, and P. Ross. An examination of multivariate time series hashing with applications to health care. In *2014 IEEE International Conference on Data Mining*, pages 260–269, Dec 2014.

[13] H. Hachiya and M. Matsugu. Nsh: Normality sensitive hashing for anomaly detection. In *2013 IEEE International Conference on Computer Vision Workshops*, pages 795–802, Dec 2013.

[14] R.L. Wheeden. *Measure and Integral: An Introduction to Real Analysis, Second Edition*. Chapman & Hall/CRC Pure and Applied Mathematics. Taylor & Francis, 2015.

[15] Gustavo E. Batista, Eamonn J. Keogh, Oben Moses Tataw, and Vinícius M. Souza. Cid: An efficient complexity-invariant distance for time series. *Data Min. Knowl. Discov.*, 28(3):634–669, May 2014.

[16] Rafael Giusti and Gustavo E. A. P. A. Batista. An empirical comparison of dissimilarity measures for time series classification. In *Proceedings of the 2013 Brazilian Conference on Intelligent Systems*, BRACIS '13, pages 82–88, Washington, DC, USA, 2013. IEEE Computer Society.