Intrusion Detection using Parzen-Windows on Provenance Graph Statistics

Kenny Yu
Harvard University
kennyyu@college.harvard.edu

R. J. Aquino
Harvard University
rjaquino@college.harvard.edu

CS261, Fall 2013

Abstract

Provenance is data that describes how a digital artifact came to be in its current state. We hypothesize that intrusions on a system leave behind anomalies in the lineage of digital artifacts. We present an intrusion detection approach to find these anomalies by analyzing centrality metrics on provenance graphs. We use a Parzen-Window approach (TODO CITE) on various provenance graph centrality metrics (TODO CITE) to determine probability density estimates of normal behavior, and we use these density estimates to determine if an intrusion occurred. We used this approach to analyze *user-to-remote* (u2r) intrusions and *remote-to-local* (r2l) intrusions (TODO: include r2l?) from the 1998 DARPA Intrusion Detection data sets (TODO CITE) and achieved up to *TODO true positive rate for intrusions* accuracy in detecting intrusions with only *TODO false positive rate for intrusions* accuracy. We also present future work to extend our intrusion model to an online intrusion detection system.

1 Introduction

For as long as systems have existed, there have been malicious users that attempt to exploit vulnerabilities in systems to gain unintended privileged access. As a result, system designers and administrators place a large effort in securing systems and preventing intrusions. However, intrusions inevitably occur because of bugs or flaws within the system, and as a result, system administrators want to have some automatic way of detecting these intrusions.

Intrusions often leave behind digital artifacts, e.g., unusual output or log files, a process executed with an unusual arguments or environment, unusual system call activity by a process. In addition to unusual digital artifacts, we hypothesize that intrusions also leave behind abnormalities in the *provenance* of these digital artifacts, the lineage of how these digital artifacts were created.

Provenance is data that describes how digital artifacts came to be in their current state. Provenance data is typically structured as a directed acyclic graph with typed nodes and typed edges. Nodes typically include processes and files, and edges are directed from nodes to their dependencies. For example, process nodes have edges directed towards input file nodes and to the parent process's node, and file nodes have edges directed towards previous versions of the file and to the nodes of the processes that modified the file.

Existing intrusion detection systems (IDS) use provenance data only to a limited extent. ??? person analyzes basic statistics on provenance graphs (e.g., number of nodes, number of edges, ... TODO) in order to facilitate easier manual intrusion detection by a human (CITE WORK). Their work, however, does not present a way of using provenance graph to automatically determine intrusions. Other systems use provenance data to determine the scope of an intrusion. For example, Backtracker (CITE WORK) uses provenance graphs to build causality graphs of intrusions: once the system has determined an intrusion has occurred, the system follows the causality graph backwards determine the original source of the attack, and then it follows the causality graph forwards to determine all the objects tainted by the attack. However, the authors do not present a way of using provenance data to automatically detect intrusions.

¿¿¿ INSERT PROVENANCE GRAPH SMALL EXAMPLE HERE SHOWING DATA FLOW

In this paper, we present an approach to use provenance data to automatically detect intrusions. Intrusion detection can be framed as a *novelty detection* problem, in which one attempts to decide whether an unknown test pattern is produced by an underlying distribution corresponding to a training set of normal patterns (CITE WORK). However, ???authors note that in the case of intrusion detection, novel or abnormal patterns are typically difficult to obtain, and as a result, they present a *Parzen-Window* approach for non-parametric density estimation. Using this technique, they obtain a high degree of success in identifying various intrusion types (CITE WORK). Because obtaining abnormal patterns is difficult, we borrow their Parzen-Window approach to build models of normal behavior using various provenance graph statistics and graph centrality metrics, and we evaluate the success of this technique on *user-to-local* (u2l) intrusions (intrusions that provide unauthorized access to local superuser (root) privileges), and *remote-to-local* (r2l) intrusions (intrusions that provide unauthorized access from a remote machine).

The main contributions of this paper are the following:

1. Discuss which provenance graph statistics we chose to use, how we chose them, and why we chose them.

2. Analyze the Parzen-Window technique with various provenance graph statistics on a real intrusion detection data set and evaluate its performance.

3. Discuss the limitations of the approach and present future work to transform the technique into an online intrusion detection system.

POSSIBLY INCLUDE THIS * the papers suck at finding u2r * because they don't look at provenance graph structure * we can do better?

2 Related Work

2.1 Existing IDSs

Many existing intrusion detection systems have made use of the fact that intrusion detection is an example of a *novelty detection* problem where access to novel or abnormal patterns is difficult to obtain. Somayaji and Forrest attempt to solve this problem by analyzing sequences of system calls (CITE WORK). By analyzing the sequences of system calls a process makes over time, their system builds a profile of "normal" process behavior. When the process in the future makes enough patterns of unrecognized sequences of system calls, the system flags this process as behaving abnormally and attempts to stop the process, either through exponentially slowing down system calls or aborting system calls entirely. Notably, their system is trained only by analyzing "normal" data without knowing explicitly what is considered "abnormal."

2.2 Parzen-Windows

??? person attempt to solve the lack of abnormal patterns problem by using Parzen-Windows to build *nonparametric* density estimations of normal behavior (CITE). A density estimation is considered *nonparametric* if the estimation makes no assumptions about the forms of the PDFs, except that PDFs are smooth. By using a nonparametric density estimation, one can build a probability density estimate of normal behavior by only having examples of normal behavior and no examples of abnormal behavior.

A *Parzen-window* estimate of a probability density function $p((x))$ based on n examples in a dataset D drawn from model $\mathcal{M}$ is given by:

$$p(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\delta_n(\mathbf{x}-\mathbf{x_i})$$

where $\delta_n(\cdot)$ is a kernel function. They chose to use radially-symmetric Gaussian kernel functions because Gaussian functions are smooth and therefore $p(\mathbf{x})$ will be smooth, and because a radially-symmetric Gaussian function can be specified by a single parameter, the variance of the kernel. Using a common variance σ^2 for all the Gaussian kernels, we can rewrite $p(\mathbf{x})$ as:

$$p(\mathbf{x}) = \frac{1}{n(2\pi)^{d/2}\sigma^d}\sum_{i=1}^{n}\exp\left\{-\frac{||\mathbf{x}-\mathbf{x_i}||^2}{2\sigma^2}\right\}$$

where d is the dimensionality of the feature space $\mathbf{x}$.

Let ω_1 denote the state of being normal and ω_0 the state of being abnormal. To test if an example $\mathbf{x} \in \omega_1$, they reframe the problem using hypothesis testing. Let $\mathbf{y}$ be an arbitrary example from D drawn from $\mathcal{M}$, and let

$$L(\mathbf{y}) = \log p(\mathbf{y})$$

be the log-likelihood of $\mathbf{y}$ with respect to $\mathcal{M}$. Then we test the hypothesis that $L(\mathbf{x})$ is drawn from the distribution of the log-likelihood of the random examples in D with:

$$\begin{aligned} P(L(\mathbf{y}) \leq L(\mathbf{x})) &= \frac{\#\mathbf{y} \text{ with } L(\mathbf{y}) \leq L(\mathbf{x})}{n} \\ &> \psi \end{aligned}$$

for some threshold $0 \leq \psi \leq 1$, called the *false detection rate*. Thus, $\mathbf{x} \in \omega_1$ is behaving normally if and only if $P(L(\mathbf{y}) \leq L(\mathbf{x})) > \psi$.

In this paper, we use this same model to build profiles of "normal" behavior for each process. We use one-dimensional feature vectors x, and we use various provenance graph statistics and graph centrality metrics as our features. For the common variance of our Gaussian kernels, we chose the variance to be the variance of the data set D for each unique process name. We vary the features we used and values of ψ and evaluate its performance on an intrusion detection data set.

inspiration - novelty detection problem with intrusion detection - somayaji use sequences of system calls - system calls and BSM data can be used to build provenance graphs; we believe that provenance graphs provide more information than somayaji and can help detect intrusions with better results

use provenance to automatically detect intrusions - existing machine learning techniques on KDD data set: use features of an intrusion (number of data) - but not actually using graph structure - parzen window–allow us to build a profile of normal with lack of abnormal examples explain the math of this

we limit our work to these intrusions - explain various kinds - why just user 2 local

2.3 Provenance Graph Features

how we choose features to look at - basic statistics for manual detection - graph centrality and clustering can help us identify the different kinds of tasks running - use these metrics for kernel estimation and

3 Design and Implementation

3.1 Selecting Metrics

talk about PASS and SPADE here.

put that table here

4 Evaluation

4.1 Experimental Setup

4.2 Results

4.3 Discussion

5 Conclusion

6 Limitations & Future Work

References

[1] Cao, D., Qiu, M., Chen, Z., Hu, F., Zhu, Y., and Wang, B. Intelligent Fuzzy Anomaly Detection of Malicious Software. In *Internal Journal of Advanced Intelligence*, vol. 4, no. 1, pp 69-86 (December 2012).

[2] Inoue, H. and Somayaji, A. Lookahead Pairs and Full Sequences: A Tale of Two Anomaly Detection Methods. In *2nd Annual Symposium on Information Assurance* (June 2007).

[3] King, S. T. and Chen, P. M. Backtracking Intrusions. In *SOSP'03 Proceedings of the nineteenth ACM symposium on Operating systems principles* (December 2003).

[4] King, S. T., Mao Z. M., Lucchetti, D. G., and Chen, P. M. Enriching intrusion alerts through multi-host causality. In *Proceedings of the 2005 Network and Distributed System Security Symposium* (February 2005).

[5] Lei, H. and Duchamp, D. An Analytical Approach to File Prefetching. In *Proceedings of the USENIX 1997 Annual Technical Conference* (January 1997).

[6] Macko, P., Margo, D., Seltzer, M. Local Clustering in Provenance Graphs (Extended Version). In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management* (August 2013).

[7] Macko, P. and Seltzer, M. Provenance Map Orbiter: Interactive Exploration of Large Provenance Graphs. In *TaPP'11 Proceedings of the 2nd conference on Theory and practice of provenance* (June 2011).

[8] Margo, D., and Smogor, R. Using Provenance to Extract Semantic File Attributes. In *TaPP'10 Proceedings of the 2nd conference on*

Theory and practice of provenance (February 2010).

[9] Muniswamy-Reddy, K., Braun, U., Holland, D. A., Macko, P., Maclean, D., Margo, D., Seltzer, M., and Smogor, R. Layering in Provenance Systems. In *Proceedings of the 2009 USENIX Annual Technical Conference* (June 2009).

[10] Muniswamy-Reddy, K., Holland, D. A., Braun, U., and Seltzer, M. Provenance-Aware Storage Systems. In *Proceedings of the 2006 USENIX Annual Technical Conference* (June 2006).

[11] Offensive Security, Inc. The Exploit Database. `http://www.exploit-db.com`.

[12] Rapid 7 Inc. Metasploit Framework. `http://www.metasploit.com`.

[13] Somayaji, A. and Forrest, S. Automated Response Using System-Call Delays. In *Proceedings of the 2000 USENIX Annual Technical Conference* (August 2000).

[14] Tariq, D., Baig, B., Gehani, A., Mahmood, S., Tahir, R., Aqil, A., and Zaffar, F. Identifying the provenance of correlated anomalies. In *SAC'11 Proceedings of the 2011 ACM Symposium on Applied Computing* (March 2011).