Learning Sequential Models for Detecting Anomalous Protocol Usage

Lloyd Greenwald LGREENWALD@LUCENT.COM
Lucent Bell Labs, 67 Whippany Road, Whippany, NJ 07981 USA

Abstract

Ambiguities in complex protocol specifications, such as the TCP/IP protocol stack, lead to implementation variations. Adversaries may take advantage of these ambiguities to exercise potentially untested software inputs and sequences of inputs in an effort to find exploitable vulnerabilities. These intrusion attempts take place over sequences of data items, rather than as singular events. In order to detect these attempts we must understand common protocol usage and isolate sequences that represent uncommon usage. In this work-in-progress we look at the problem of learning sequential models from data in order to detect anomalous use of protocols. Our approach is to make use of protocol traffic data rather than source code or detailed knowledge of protocol specifications.

1. Problem Overview

Protocols are designed to permit systems to interoperate across diverse platforms. For example, every machine that implements the TCP/IP protocol stack should be able to exchange packets regardless of hardware or operating system. Ambiguous protocol specifications permit protocol implementers flexibility to optimize their implementations, set initial parameters, or to include or exclude options. In addition to subtle implementation differences, implementations may have bugs or fail to correctly capture the specification.

Testing complex systems for potential security compromises is a difficult problem. This problem becomes increasingly difficult as these systems must interact with other systems executing differing implementations. A determined adversary can take advantage of this situation to compromise systems by exercising potentially untested protocol features and sequences of features. While a brute-force attack would be as difficult as the testing problem itself, a more directed but unsuspected approach to finding exploitable vulnerabilities might prove successful.

In this work-in-progress we discuss methods to discover untested protocol behavior before an attacker is able to exploit it. Our approach is to learn sequential models representing recent protocol usage patterns. We can then analyze these models to isolate common and uncommon usage and use the models to detect future anomalous protocol usage. This technique provides an alternative to anomaly detection using packet-based signatures or aggregate statistics. Our methods are designed to detect adversary abuses that are designed to look like non-malicious new implementations of ambiguous protocols.

2. Related Work

There has been considerable progress in developing techniques that permit the formal verification of networking protocols (Bishop *et. al.* 2005; Paxson 1997). These techniques require extensive knowledge of specifications and provide understanding of how well implementations follow specifications and what properties can be proved about the protocols based on the specifications. These techniques do not, however, account for how well differing implementations of the same protocol will interact and, thus, cannot ensure node security as networks change over time. They also do not take into account attempts by malicious attackers to abuse protocol specifications. Additionally, formal verification techniques do not account for environmental differences (e.g. link quality changes) that are easily incorporated into learned sequential models.

Recent work on traffic classification (Karagiannis *et. al.* 2005; Xu *et. al.* 2005) provides techniques to understand and characterize communication patterns within a data stream and classify the applications that lead to those patterns. While that work focuses on understanding the patterns in the current stream of traffic, our approach is to use the current stream of traffic to build models to help understand potential future traffic patterns. Our goal is then to analyze these potential future traffic patterns for security compromises.

Our approach builds on our earlier work (Sant *et. al.* 2005) developing stochastic sequential models of web

Appearing in *Proceedings of the Workshop on Machine Learning Algorithms for Surveillance and Event Detection* at the *23rd International Conference on Machine Learning,* Pittsburgh, PA, 2006. Copyright 2006 by the author(s)/owner(s).

application behavior based solely on web log data. The models we built in that work were used to proactively test web applications for previously unobserved errors.

3. Learning Sequential Models

Two primary challenges in learning sequential models to represent protocol usage are:

1. Identifying the underlying stochastic processes in the data, and
2. Trading off accuracy (history) for reliability (more training data)

Once the underlying processes are identified and history-dependence tradeoffs are determined, machine learning methods can be used to build these models automatically from sample data. The subsequent models capture actual usage patterns both from the set of all possible usages specified in the protocol as well as usages not specified in the protocol.

3.1 Identifying Underlying Processes

Typical protocols are multi-threaded, interacting with multiple systems or users simultaneously. A dataset of protocol traffic intersperses these interactions. While we would like to aggregate these multiple interactions into a single sequential model that represents recent protocol usage, to do so we must be able to identify how messages within the data stream relate to each other.

Consider the example of learning a sequential model of web server usage from a web log. The messages corresponding to a finite interaction of a user with the web server is called a *user session* (Elbaum *et. al.* 2003). Individual user sessions can be identified in a web log using session IDs or a combination of client IP address and time range (Sampath *et al.* 2004a; 2004b). In order to build a sequential model of recent user sessions we must first identify the sequence of messages belonging to each session. While web server user sessions are considered to be independent, this is not necessarily true of interspersed traffic in general. These dependencies must be considered in learning sequential models.

In addition to identifying distinct sequences of messages within a dataset, a sequential model of a protocol might best be represented by multiple interacting stochastic processes. For example, in our work on building sequential models from web logs we identified two interacting processes interspersed in the data stream. The first process is the possible sequences of URLs that are visited as a user navigates a web application. We call this the *control* model. The second process is the possible sets of parameter values that are sent as name-value pairs in a request for any specific URL, such as when a user enters data in a form. We call this the *data* model. Both processes are history dependent to different degrees. Furthermore, the processes are interdependent as the sequence of requested URLs determines the required name-value pairs and the values affect the sequence of URLs.

3.2 Trading Off Accuracy for Reliability

Sequential models can be used to compactly represent a probability distribution over all possible sequences of messages in the recent usage of a protocol. In general, the conditional probability of the next message may depend on the history of all previous messages. A common statistical learning technique is to build compact models by trading off accuracy for reliability. We do so by making use of the chain rule and the conditional independence (Markov) assumption that messages far enough in the past do not affect the probability of the next message, if we know a subset of recent messages. The accuracy of the approximation depends on how much information is lost by ignoring some history. Variants include the 1-Markov (bigram) assumption in which only the previous message is needed to estimate the probability of the next message and the 2-Markov (trigram) assumption in which the previous two messages help estimate the probability of the next message. A 2-Markov model is depicted in Figure 1. In (Sant *et. al.*) we report on experiments with unigram, bigram, and trigram models in learning sequential models.

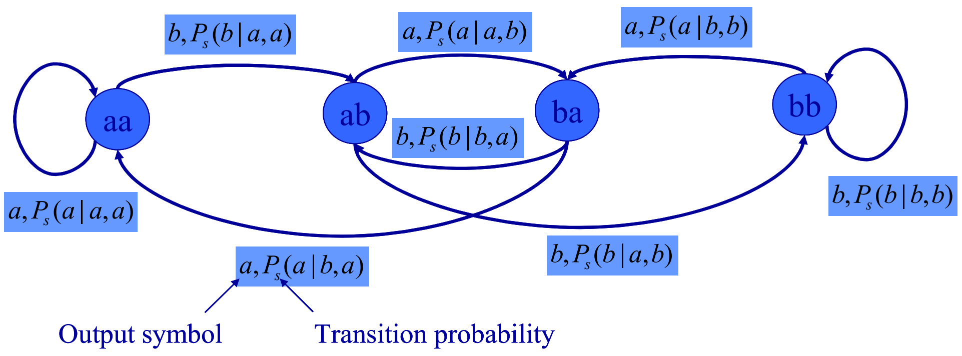

Figure 1. A 2-Markov model capturing the probability of the next message conditioned on the previous two messages, using a two message alphabet.

4. Detecting Anomalous Protocol Usage

A learned sequential model captures a probability distribution over recent protocol usage patterns. The model captures the common and uncommon subsequences within these message sequences, and their relative frequencies. We are investigating analytical and simulation uses of these models to build anomaly detection solutions. For example, these models can be used as sources of test cases to proactively probe instrumented systems for vulnerabilities by constructing plausible but untested sequences of messages. A directed approach to investigating potential security compromises could investigate questions such as: what would happen if a system constructed a message sequence by concatenating two highly likely message sequences that can possibly follow each other in the protocol specification but have not yet been observed or tested? Such a directed approach could find sequences of messages that are not processed correctly by a specific protocol implementation. Our proposed detection methods find these sequences before an attacker can abuse them. Additionally, we can use these models to reactively detect new and unlikely protocol usage patterns before an adversary can complete an attack.

References

Bishop, S., Fairbairn, M., Norrish, M., Sewell, P., Smith, M., and Wansbrough, K. (2005). Rigorous specification and conformance testing techniques for network protocols, as applied to TCP, UDP, and sockets. In *SIGCOMM Comput. Commun. Rev.* 35, 4, 265-276.

Elbaum, S.; Karre, S.; and Rothermel, G. (2003). Improving web application testing with user session data. In *Proceedings of the 25th International Conference on Software Engineering*, 49–59.

Karagiannis, T., Papagiannaki, K., and Faloutsos, M. (2005). BLINC: multilevel traffic classification in the dark. In *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols For Computer Communications* (Philadelphia, Pennsylvania, USA, August 22 - 26). SIGCOMM '05. ACM Press, New York, NY, 229-240.

Paxson, V. (1997). Automated Packet Trace Analysis of TCP Implementations. In *Proc. ACM SIGCOMM '97*. Cannes, France.

Sampath, S.; Mihaylov, V.; Souter, A. L.; and Pollock, L. L. (2004a). Scalable approach to user-session based testing of web applications through concept analysis. In *ASE*, 132– 141.

Sampath, S.; Mihaylov, V.; Souter, A. L.; and Pollock, L. L. (2004b). Composing a framework to automate testing of operational web-based software. In *ICSM*, 104–113.

Sant, J., Souter, A., and Greenwald, L. (2005). An exploration of statistical models for automated test case generation. In *Proceedings of the Third international Workshop on Dynamic Analysis (*St. Louis, Missouri, May 17 - 17). WODA '05. ACM Press, New York, NY, 1-7.

Xu, K., Zhang, Z., and Bhattacharyya, S. (2005). Profiling internet backbone traffic: behavior models and applications. In *SIGCOMM Comput. Commun. Rev.* 35, 4, 169-180.