Anomalous Payload-Based Network Intrusion Detection

Ke Wang and Salvatore J. Stolfo

Computer Science Department, Columbia University
500 West 120th Street, New York, NY, 10027
{kewang,sal}@cs.columbia.edu

Abstract. We present a payload-based anomaly detector, we call PAYL, for intrusion detection. PAYL models the normal application payload of network traffic in a fully automatic, unsupervised and very effecient fashion. We first compute during a training phase a profile byte frequency distribution and their standard deviation of the application payload flowing to a single host and port. We then use Mahalanobis distance during the detection phase to calculate the similarity of new data against the pre-computed profile. The detector compares this measure against a threshold and generates an alert when the distance of the new input exceeds this threshold. We demonstrate the surprising effectiveness of the method on the 1999 DARPA IDS dataset and a live dataset we collected on the Columbia CS department network. In once case nearly 100% accuracy is achieved with 0.1% false positive rate for port 80 traffic.

1 Introduction

There are many IDS systems available that are primarily signature-based detectors. Although these are effective at detecting known intrusion attempts and exploits, they fail to recognize new attacks and carefully crafted variants of old exploits. A new generation of systems is now appearing based upon anomaly detection. Anomaly Detection systems model normal or expected behavior in a system, and detect deviations of interest that may indicate a security breach or an attempted attack.

Some attacks exploit the vulnerabilities of a protocol, other attacks seek to survey a site by scanning and probing. These attacks can often be detected by analyzing the network packet headers, or monitoring the network traffic connection attempts and session behavior. Other attacks, such as worms, involve the delivery of bad payload (in an otherwise normal connection) to a vulnerable service or application. These may be detected by inspecting the packet payload (or the ill-effects of the worm payload execution on the server when it is too late after successful penetration). State of the art systems designed to detect and defend systems from these malicious and intrusive events depend upon “signatures” or “thumbprints” that are developed by human experts or by semi-automated means from known prior bad worms or viruses. They do not solve the “zero-day” worm problem, however; the first occurrence of a new unleashed worm or exploit.

E. Jonsson et al. (Eds.): RAID 2004, LNCS 3224, pp. 203–222, 2004.

Systems are protected after a worm has been detected, and a signature has been developed and distributed to signature-based detectors, such as a virus scanner or a firewall rule. Many well known examples of worms have been described that propagate at very high speeds on the internet. These are easy to notice by analyzing the rate of scanning and probing from external sources which would indicate a worm propagation is underway. Unfortunately, this approach detects the early onset of a propagation, but the worm has already successfully penetrated a number of victims, infected it and started its damage and its propagation. (It should be evident that slow and stealthy worm propagations may go unnoticed if one depends entirely on the detection of rapid or bursty changes in flows or probes.)

Our work aims to detect the first occurrences of a worm either at a network system gateway or within an internal network from a rogue device and to prevent its propagation. Although we cast the payload anomaly detection problem in terms of worms, the method is useful for a wide range of exploit attempts against many if not all services and ports.

In this paper, the method we propose is based upon analyzing and modeling normal payloads that are expected to be delivered to the network service or application. These normal payloads are specific to the site in which the detector is placed. The system first learns a model or profile of the expected payload delivered to a service during normal operation of a system. Each payload is analyzed to produce a *byte frequency distribution* of those payloads, which serves as a model for normal payloads. After this *centroid* model is computed during the learning phase, an anomaly detection phase begins. The anomaly detector captures incoming payloads and tests the payload for its consistency (or distance) from the centroid model. This is accomplished by comparing two statistical distributions. The distance metric used is the Mahalanobis distance metric, here applied to a finite discrete histogram of byte value (or character) frequencies computed in the training phase. Any new test payload found to be too distant from the normal expected payload is deemed anomalous and an alert is generated. The alert may then be *correlated* with other sensor data and a decision process may respond with several possible actions. Depending upon the security policy of the protected site, one may filter, reroute or otherwise trap the network connection from being allowed to send the poison payload to the service/application avoiding a worm infestation.

There are numerous engineering choices possible to implement the technique in a system and to integrate the detector with standard firewall technology to prevent the first occurrence of a worm from entering a secured network system. We do not address the correlation function and the mitigation strategies in this paper; rather we focus on the method of detection for anomalous payload.

This approach can be applied to any network system, service or port for that site to compute its own "site-specific" payload anomaly detector, rather than being dependent upon others deploying a specific signature for a newly detected worm or exploit that has already damaged other sites. As an added benefit of the approach described in this paper, the method may also be used to detect encrypted channels which may indicate an unofficial secure tunnel is operating against policy.