CS259D HW3 Theoretical Topic: Web Security

Since the creation of the first web server in 1990, the World Wide Web has erupted as a medium of information distribution and communication. Today, companies like Google, Facebook, and Amazon employ gigantic web server systems to handle millions to billions of requests per day. However, these web servers have become targets to an exponentially increasing number of cybersecurity attacks. Popular attacks include distributed denial-of-service (DDoS), buffer overflow, and cross site scripting (XSS).

Of course, much technical research has been conducted on how to address these issues and to create prevention and detection mechanisms. As we saw in class, various models can be built around the contents of HTTP requests and web logs. Features such as attribute length, character distribution, access frequency, and program invocation order are analyzed using machine learning (ML) techniques to detect anomalous behavior. In this report, we will attempt to append to this trajectory by surveying research from three more recent papers. The first paper "Server Side Detection of Content Sniffing Attacks" by Barua et al presents an improved algorithm used for detecting malicious uploads to web servers. The second paper "Detection of Insider Attacks to the Web Server" by Choi et al utilizes an algorithm that detects anomalous outsider behavior to instead detect anomalous insider behavior. The third paper "Discriminating DDoS Attacks from Flash Crowds" by Yu et al presents a way of differentiating between DDoS attacks and flash crowds in existing DDoS detection systems.

The first paper, by Barua et al, discusses content sniffing attacks, a type of XSS attack, in which the browser tries to render non-HTML files that contain malicious HTML or JavaScript, which are generally uploaded by a user to the server. Currently, most browsers employ content sniffing detection algorithms. However, these algorithms are insufficient because they only verify a fixed number of bytes at the beginning of the payload, and they have no way of detecting the malicious impact of the content on browsers. The paper proposes a new detection pipeline with a series of security checks. First, basic checks are done on the MIME type, ensuring the file extension matches the magic header in the payload and ensuring the type conforms to a list of whitelisted MIME types. If this fails, then the file is immediately rejected. Next, the file format is normalized to UTF-8 encoding and is run through an HTML and JavaScript parser. These parsers look for tags in the file signifying HTML or JavaScript code. If such tags exist, the file is flagged as potentially malicious and enters a final check through the mock download tester. The mock download tester emulates a browser and forces the browser to render the file as HTML. If the emulator exhibits any unusual symptoms (i.e. executing JavaScript) while downloading the file, then the file is regarded as malicious and rejected. For its benchmark suite, the paper achieved a false positive rate of 0%, which is the expected rate for current algorithms, and a false negative rate of 0%, which is a marked improvement on current techniques. Overall, the detection system proposed in the paper seems like a clear improvement over current techniques. However, the drawback is that it requires more computation time, which could be prohibitive for larger files or certain applications.

The second paper, by Choi et al, explores techniques for detecting insider attacks, defined as attacks made by those with privileges on the server side of a system. The paper notes that insider-attack-detection and intruder-attack-detection commonly use signature-based or anomaly-based systems. Interestingly, even though intruder detection systems have already evolved to use a hybrid of signature-based and anomaly-based, the literature on insider detection has not explored the use of hybrid detection. This paper fills that gap in the literature, and proposes a hybrid detection system using an open source system called Snort for their signature-based detection and hidden Markov models (HMM) for anomaly-based detection. The system essentially checks all packets with the signature-based detection as the first line of defense and anomaly-based detection as the second line. The signature-based detection is comprised of 107 manually created rules to detect common malicious activities such as XSS, URL spoofing, information leakage, etc. The anomaly-based detection uses a pre-generated hidden Markov model to calculate the probability of an observed sequence of JavaScript/HTML tags being "normal" behavior. The authors tested on two different data sets and found that anomaly detection was able to catch certain malicious activities whereas signature detection was not, and vice-versa. Furthermore, they found that when the two detection schemes were combined, the system was able to detect all malicious documents outbound from the server.

The third paper, by Yu et al, investigates methods for distinguishing DDoS attacks from real flash crowd events. For fear of false positives, many DDoS detection systems act conservatively when an event has characteristics similar to a flash crowd. Thus, attackers have been able to bypass these systems by having their DDoS attack mimic characteristics of flash crowds. This paper makes the interesting observation that the number of botnets (as of writing: 2012) in a typical DDoS attack is in the order of thousands, maybe tens of thousands. However, the number of distinct clients in a flash crowd can be hundreds of thousands or more. This means that the bots have to issue multiple requests each in quick succession in order to cause the same effect. As a result, the variance in a DDoS attack flow is lower than that of a regular flash crowd. In other words, DDoS attack flows are much more similar to each other than network flows from a regular flash crowd. To measure this, the authors first calculate network flows through all edge routers of a server, discretizing the average packet rate over chunks of time. Then the correlations between the flows are calculated, and, without getting too much into the specifics of the hyperparameters involved, thresholds are used to determine whether the correlations between the flows is high enough to be classified as a DDoS attack. After implementing this system, the authors found that it was indeed the case that DDoS attacks had high network flow correlations and for their test case, they were able to see a difference between correlations between DDoS attack flows and regular flash crowd flows. Even though more investigation definitely needs to be done in this area, this paper represents a proof of concept of an effective technique for discriminating flash crowds from DDoS attacks.

Overall, the three papers show progress being made in web security. However, they also illustrate the challenges and open problems web security researchers have to face. Many new techniques, such as that proposed in the first paper to detect content sniffing, are being developed, but because of the real-time component of web security, the computational time cost could be prohibitive. Thus, researchers need to not only find a way to extract meaning from the data, but also to extract it quickly. Also, as illustrated in the second paper, web attacks can come from almost anywhere. How do we address this and preemptively stop attacks before they happen? And finally, as we can see from the third paper, false positives can inhibit the functionality of a web application, but in reducing false positives, we allow attackers to disguise their attacks as anomalous but innocuous events (i.e. flash crowds). The authors of the third paper make progress in addressing this in the case of DDoS attacks, but similar challenges exist in all aspects of web security. How do we separate the attacks from the noise? And how do we do so with high confidence?

In conclusion, the overall impression we get from our brief survey of web security research is a promising one. New algorithms and methodologies are always being presented to address the newest problems. However, there is still a lot of work to do. Thus far, it seems that the field is reacting to the problems of the day instead of proactively preventing them before they occur. Undoubtedly, we will need new breakthroughs and innovations to break this next barrier. It will be interesting to see what will happen in the future!

References.

1. "Server Side Detection of Content Sniffing Attacks" by Barua et al, 2011 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6132950
2. "Detection of Insider Attacks to the Web Server" by Choi et al, 2012 http://isyou.info/jowua/papers/jowua-v3n4-3.pdf
3. "Discriminating DDoS Attacks from Flash Crowds" by Yu et al, 2012 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6060809