John Lancaster February 12, 2012

In Manners 2011 article "The User Agent Field: Analyzing and Detecting the Abnormal or Malicious in your Organization," he thoroughly illustrates a vulnerability known as "active user agent injections." This vulnerability can potentially leave organizations open to a range of attacks, from cross-site scripting (XSS), SQL injections, to denial of service. While Manners' article is well thought out and educational on a technical level, I feel as though it's scope is too narrow, the context surrounding the problem space is inadequately described, and a holistic solution is not given. In this day and age, adequate solutions must transcend reactionary measures and static defenses and move towards dynamic, orthogonal layers of security that proactively and automatically mitigate threats. To support my assertion I will provide alternative techniques to bolster Manners recommendations.

Every organization that does business over the Internet is susceptible to user-agent type attacks due to the fact that exploits hide themselves within seemingly normal HTTP traffic. These exploits aren't exactly new. According to the Open Web Application Security Project (OWASP) 2007 and 2010 "Top 10" reports, injections have remained the number one threat in the Internet landscape over the last 5 years. Manners' article focuses on user-agent injections to highlight this class of vulnerabilities.

A user-agent injection is just one attack vector out of many within the grater context of injections, a point that is not thoroughly developed in the article. It is used by hackers to insert malicious code into HTTP headers of a request in the classic client server model. Hackers are able to exploit this vulnerability for one main underlying reason, organizations are not careful about what data they are taking in, or sending out. Manners identifies this underlying vulnerability as an issue of "trust," since traditionally the user-agent field was not used for nefarious activities. To illustrate how a hacker could abuse the "trust" between a client and a server, Manners maps out the anatomy of an XSS attack. A XSS attack is when a hacker injects malicious code to a server, which then gets

<sup>&</sup>lt;sup>1</sup> http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202010.pdf p. 4

relayed to other users accessing resources from the same server.<sup>2</sup> Again, this is possible because someone is not being careful as to what data they are taking in or sending out; a process generally called data sanitization.

To illustrate a user-agent injection attack Manners profiled a web-analytics application. This application gives website owners metrics on their customer's web browsers based on information gleaned from their user-agents. These details can be safely ascertained from the user-agent, unless a hacker has loaded it with a malicious script instead. In this case, instead of displaying a list of valid user-agents to the analyst reviewing metrics, a script is sent from the server and silently executed on the analysts' computer. At this juncture the payload and intent of the attack is unclear, however the hacker has successfully inserted un-trusted code into their environment. Manners offers a twofold approach for defense. Part one relies on identifying malicious user-agents; part two involves validating and sanitizing user-agents. While his general approach is valid, his implementation is rigid, reactionary, non-adaptive, and maintenance intensive.

The first step is identifying malicious user-agents. This can be accomplished by acquiring a list of known malicious user-agents and denying their requests. Blacklists of this nature exist and are openly available online.<sup>3</sup> There are also white-lists that enumerate all known legitimate user-agents<sup>4</sup>. Manners suggests an organization define a sub-set of legitimate user-agents that are allowed access to their servers. For example, only allowing Internet Explorer user agents. However, what if someone wants to access their site from an iPhone? If the user-agent isn't allowed they can't see the site. This practice highlights a tradeoff between accessibility and confidentiality, which is a decision needs to be made on a per organization basis. This approach is inflexible and does not consider future innovations that may don new user-agents.

<sup>&</sup>lt;sup>2</sup> The user agent field: Analyzing and detecting the abnormal or malicious in your organization p. 29

<sup>&</sup>lt;sup>3</sup> http://virtech.org/home/blacklist.php

<sup>4</sup> http://user-agents.org

The second part is validating and sanitizing data. This practice should be standard for every piece of data, not just user-agents. Proper stewardship of all data ensures that the three tenants of information systems security, confidentiality, accessibility, and integrity are maintained. Data validation plays a direct role in supporting those goals. This point is not emphasized enough in Manners article. Further he does not correlate the practice of defending against user-agent attacks to injection exploits in general, which permeate every tier of the technology stack, not just HTTP headers.

A key concept in systems security is defense in depth. Manners' mitigation techniques are one-dimensional because they only consider the application layer, and retroactively at that. To properly secure the system on all fronts, a robust, maintainable, and orthogonal approach must be taken. The following three academic journals outline novel techniques that extend and enhance Manners' methods. Each one seeks to solve problem at hand in different ways, which in turn supports the concept of defense in depth.

"A Semantic Data Validation Service for Web Applications" (2010) seeks to defend against XSS and SQL injections by putting a novel spin on a traditional method. Leveraging semantic markup with RDFa, they propose an ontology-based architecture called "Semantic Data Validation" (SDV) to secure web applications at the application layer and at the network interface layer. It works by having application developers semantically markup input forms with RDFa. After a request is sent from a client's browser, the SDV architecture intercepts the request, parses the RDFa, and sends it off to the "Data validator mechanism." This mechanism leverages semantic data to conduct integrity checks and detect if user input has been maliciously modified between the client and server via a man in the middle attack. This method is effective because it exists at two levels, the application and network. Secondly, it introduces a standards based approach to validating data. By defining ontology's of acceptable input, attributes can be validated based on "length, data type,

<sup>&</sup>lt;sup>5</sup> Aljawarneh, Alkhateeb, Maghayreh, et al.

<sup>&</sup>lt;sup>6</sup> Semantic Data Validation for Web Applications p. 47

<sup>&</sup>lt;sup>7</sup> Semantic Data Validation for Web Applications p. 51

minimum length, and if the values contain code or special characters." This is a big win because data validation can be notoriously hard to get right in an application. Abstracting this functionality is a more maintainable approach because validation definitions can now be altered independent from the underlying application.

In "An Advanced Web Attack Detection and Prevention Tool" (2010), Kapodistria et al. advocate for WebDefender, a flexible, pattern matching web application firewall to secure traffic. It validates user input at the HTTP level, independent of the application layer. WebDefender acts as a proxy much in the same way as the SDV, by sitting between the client and the server. However, it uses regular expressions to detect and bounce bad traffic based on five attack categories: "XSS, stored XSS, SQL injection, path traversal, and command injection." Approaching security at this level is desirable because the mechanism can sit atop a web application suite and provide protection at scale across many applications, instead of on a per-application basis. Both Manners' and the SDV approach define security parameters at application layer, which is desirable, but at the same time is narrower in scope. Also, WebDefender shares the same signature-based approach that Manners advocates which is only as good as the rule set that supports it.

Shon and Moon's "A Hybrid Machine Learning Approach to Network Anomaly Detection" (2007) is by far the most interesting and novel approach to securing against malicious input. It attacks core issues with using signature-based rules to detect malicious activity. One issue is that signature-based rules rely on individuals to maintain a list of malicious user-agents or input values, which is labor intensive and inherently reactive. Additionally, signatures represent a single point of failure because if a rule is written wrong, a blacklist doesn't get updated, or a zero day attack is developed you are vulnerable. In order to mitigate against these scenarios we need "systems that do not rely on human intervention...[that] are anomaly detection systems based on machine learning, data mining, or statistical algorithms." Using the same techniques as spam filtering — Bayesian

<sup>&</sup>lt;sup>8</sup> Semantic Data Validation for Web Applications p. 48

<sup>&</sup>lt;sup>9</sup> An Advanced Web Attack Detection and Prevention Tool p. 289

<sup>&</sup>lt;sup>10</sup> A Hybrid Machine Learning Approach to Network Anomaly Detection p. 3

## Out of the Weeds: A Holistic Approach to Input Validation

probability— anomaly detection systems can learn the "normal" traffic patterns of a system.

Therefore, when attackers attempt abnormal activity, i.e. injection attacks, these learning systems can identify the activity as malicious outliers and discard the connection.

This approach is clearly the future security specialists should work towards. With the ever-increasing volume of traffic and new attacks coming out, system security administrators need all the automated, adaptive, proactive help they can get. With these enhanced methods to combat injections available, it is hard to accept Manners' recommendation for a simple signature-based defense. That being said there are always tradeoffs and compromises. Implementing these three solutions add a layer of defense, but also a layer of complexity, maintenance, cost, and overhead.

Out of the Weeds: A Holistic Approach to Input Validation

Manners, Darren. "The User Agent Field: Analyzing and Detecting the Abnormal or Malicious in your Organization."

The Sans Institute Reading Room. (2011). Accessed February 12, 2012, <a href="http://www.sans.org/reading\_room/whitepapers/hackers/user-agent-field-analyzing-detecting-abnormal-malicious-organization\_33874">http://www.sans.org/reading\_room/whitepapers/hackers/user-agent-field-analyzing-detecting-abnormal-malicious-organization\_33874</a>.

Aljawarneh, Shadi, Alkhateeb, Faisal, and Al Maghayreh, Eslam. "A Semantic Data Validation Service for Web Applications."

Journal of Theoretical and Applied Electronic Commerce Research (2010). Accessed February 12, 2012.

http://www.jtaer.com/portada.php?agno=2010&numero=1#

Kapodistria, Helen, Mitropoulos, Sarandis, and Douligeris, Christos. "An Advanced Web Attack Detection and Prevention Tool."

Emerald Group Publishing Limited (2011). Accessed February 12, 2012.

http://www.emeraldinsight.com/journals.htm?articleid=17004025

Shon, Taeshik and Moon, Jongsub. "A Hybrid Machine Learning Approach to Network Anomaly Detection."

Elsevier Information Sciences International Journal (2007). Accessed February 12, 2012.

http://www.sciencedirect.com/science/article/pii/S0020025507001648