Classification features for detecting Server-side and Client-side Web attacks

Benferhat Salem and Tabia Karim

Abstract During last years, the number and cleverness of attacks against Web related applications are steadily growing as Web services become more popular. In this paper, we propose relevant classification features for detecting Web attacks targeting either server-side or client-side applications. Four kinds of features are provided: Request general features, Request content features, Response features and Request history features. Experimental studies carried on real[1] and simulated *http* traffic including normal data and several attacks show the efficiency of our feature set in detecting Web related attacks.

1 Introduction

Web technologies are widely deployed in nowadays information systems. For the attackers, this fact offers two opportunities: Firstly, *http/https* traffic is often the only service allowed through fire-wall and filtering technologies. The second opportunity lies in increasing numbers of Web related application vulnerabilities. In spite of the importance of Web application security, there are few works proposing classification features in order to detect malicious Web activities using machine learning and data mining techniques. Moreover, most proposed feature sets in intrusion detection are network oriented [5][6] while most nowadays attacks are targeting Web related applications [4][7]. In addition, to our knowledge there is currently no preprocessing tool for extracting Web oriented features directly from network traffic.
This paper proposes a relevant feature set suitable for detecting Web related attacks. Our feature set includes basic features of *http* connections as well as derived fea-

Benferhat Salem and Tabia Karim
CRIL - CNRS UMR8188, Université d'Artois,
Rue Jean Souvraz SP 18 62307 Lens, Cedex, France, e-mail: {benferhat, tabia}@cril.univ-artois.fr

[1] Thanks to DADDi project for providing us such real network traffic (URL: http://www.rennes.supelec.fr/DADDi/)

tures summarizing past *http* connections and providing useful information for revealing suspicious behaviors involving several *http* connections. While most works focus on *http* requests, we designed features characterizing both *http* requests and their corresponding responses. Note that our feature set is directly extracted from network packets instead of using Web application logs. Processing whole *http* traffic is the only way for detecting suspicious activities and attacks in both inbound and outbound traffic.

2 Web attacks

Web attacks use Web protocols (namely *http* [1] and *https* [2]) to perform malicious actions exploiting Web application vulnerabilities. They target either Web servers, Web clients or any Web related application using Web connections. Most Web attack taxonomies [11] [4] rely on attack techniques and group them into the following categories:

1. **Input validation attacks:** Input validation attacks refer to those bypassing input validation procedures in order to exploit Web application vulnerabilities. The strategy of such attacks is to send especially crafted requests exploiting vulnerabilities in Web server/client applications. Buffer-overflow, SQL injection, Directory traversal, Cross Site Scripting, etc. are well-known examples of input validation attacks Note that input validation attacks can cause denial of service, unauthorized access or command execution and full control of victim systems.
2. **Web authentication/authorization attacks**: They are Web attacks bypassing authentication/authorization restriction mechanisms. For example, some authentication mechanisms which aim at authenticating users, can be bypassed by brute-force and dictionary attacks.
3. **Web site scan and flooding attacks**: As the number of Web sites and contents grow rapidly, attackers often use scripted and automated scanning tools such as W3af [10] to search for possible vulnerabilities in Web sites. As for flooding attacks, users may abuse in some functionalities in order to prevent other legitimate users for having access to services.

3 Classification features for detecting Web attacks

In order to design effective feature set for detecting Web attacks, we analyzed several Web-based attacks and extracted features according to common attacks techniques. Our classification features are grouped into four categories:

1. **Request general features**: They are features that provide general information on *http* requests. The following table gives detailed examples of request general features:

Table 1 Request general features

Name	Description	Type	Target attacks
Req-length	Request length	Positive Integer	Buffer overflow attacks
URI-length	URI length	Positive integer	Buffer overflow, Value misinterpretation, URI decoding errors
Req-method	Request method (GET, POST HEAD...)	Symbolic	–
Req-resource-type	Type of requested resource (html, asp, cgi, php, exe, ...)	Nominal	–
Num-param	Number of parameters	Positive Integer	Input validation
Num-arg	Number of arguments	Positive Integer	Input validation
Is-req-correct	Does the request comply with *http* protocol (ex. Is there a request method in the request?)	Boolean	URL anomalies, URL decoding errors

2. **Request content features**: These features search for particularly suspicious patterns in *http* requests. The number of meta-characters, number of directory traversal patterns, etc. are examples of features describing request content.

Table 2 Request content features

Name	Description	Type	Target attacks
Num-NonPrintChars	Number of special and meta-characters and shell codes in the *http* request (x86, carriage return , semicolon...)	Positive Integer	Buffer overflows, shell codes, URL decoding errors and anomalies
SQL-cmds-tricks	Does the request contain SQL commands (”- -, OR 1 == 1, ...)	Boolean	SQL injection
Shell-cmds	Does the request contain shell commands (All operating systems shell commands)	Boolean	Command injection
Sensitive-files	Does the request reference sensitive files? (etc/passwd,...)	Boolean	Information leak, unauthorized access, ...
Directory-traversal	Does the request contain directory traversal tricks (Presence of token like ”../”,...)	Boolean	Directory traversal
Oversized-values	Does the request contain potentially oversized numeric values	Boolean	Value misinterpretations
Default-login-passwd	Does the request include factory default logins and passwords (Guest, anonymous, root, admin,...)	Boolean	Dictionary attacks, brute-force...
Script-injection	Does the URI contain a script tag (”<script”, ”<meta”,...)	Boolean	Cross Site Scripting

3. **Response features**: These features can for instance reveal suspicious *http* content in the response, in which case Web clients are targeted by a possible attack. Table 3 provides detailed response feature examples:

Table 3 Response features

Name	Description	Type	Target attacks
Resp-Code	Response code to *http* request (200, 404, 500...)	Nominal	–
Is-html-Response	Is the response an *html* file?	Boolean	–
Response-time	Time elapsed since the corresponding http request	Real	DoS
Script-type	The type of script included in the response (Java, Visual basic, ...)	Nominal	Cross Site Scripting
Writing-script	Does the response flow include script writing functions (document.write()...)	Boolean	Session ID fixation,...

4. **Request history features**: In section 2, we pointed out that there are Web related attacks that perform through several connections. We accordingly designed derived features summarizing past *http* connections. Note that these features can be computed using a time-window or a connection-window that is fixed accord-

ing to the needed tradeoff between processing overload and detection rate. The following table contains examples of request history features:

Table 4 Request history features

Name	Description	Type	Target attacks
Num-Req-Same-Host	Number of requests issued by same source	Positive integer	Flooding, vulnerability scans
Num-Req-Same-URL	Number of requests with same URL	Positive integer	Flooding from same source/multiple sources
Num-Req-Same-Host-Diff-URI	Number of requests issued by same source and requesting different URLs	Positive integer	Vulnerability scans
Inter-Req-Interval	Inter request time interval	Positive integer	Flooding and vulnerability scans...

4 Experimental studies

In order to evaluate the relevance of our classification feature set, we carried out experimental studies on real and simulated *http* traffic using a C4.5 decision tree [15] which is among the most efficient classifiers. We extracted *http* traffic and preprocessed it into connection records using only packet payloads. Each *http* connection is characterized by the four feature categories presented in Section 3. Note that in order to label the preprocessed *http* traffic, we analyzed this data using Snort IDS[12] as well as manual analysis. As for other attacks, we simulated most of the attacks involved in [13][14] which is to our knowledge the most extensive and upto-date Web-attack data set. In addition to these Web attacks, we played vulnerability scanning sessions using W3af [10]. The following table provides details about our experimentations:

Table 5 Training and testing data set distributions and C4.5 evaluation results

	Training data		**Testing data**		Evaluation on	Evaluation on
	Number	%	Number	%	Training data	Testing data
Normal	55342	55.877%	61378	88.88 %	100%	99.8%
Vulnerability scan	31152	31.453%	4456	6.45 %	99.99%	0.00%
Buffer overflow	9	0.009%	15	0.02%	100%	20%
Input validation	44	0.044%	4	0.01 %	99.99 %	99.99 %
Value misinterpretation	2	0.002%	0	0%	0.00%	–
Poor management	3	0.003%	0	0%	66.76%	–
URL decoding error	3	0.003%	0	0%	0.00%	–
Flooding	12488	12.609%	3159	4.57	99.99%	99.99%
Cross Site Scripting	0	0%	**6**	**0.01 %**	–	0.00%
SQL injection	0	0%	**9**	**0.01 %**	–	0.00%
Command injection	0	0%	**12**	**0.02 %**	–	0.00%
Total	99043	100%	69059	100%	PCC=99.93%	PCC=93.31%

In order to evaluate the ability to detect new attacks, we build a testing data set including normal real *http* connections as well as known attacks and new ones. When trained and tested using the same training data, the decision tree's PCC (Percent of Correct Classification) is too close to 100%. Then this is the indication that training data set is free from incoherences. The results of building the C4.5 decision tree on

training data and evaluating it on testing data set show that in spite of a good PCC, new attacks are not detected since they are completely different from those included in training data. This is a recurring problem affecting most classifiers in intrusion detection [16][17]. Note that most miss-classifications are false negatives as it is the case with most classifiers used in intrusion detection [16][17].

5 Conclusion

In this paper, we proposed a relevant feature set for detecting Web attacks. We proposed four classification feature categories relative to *http* request general features, content features, response features and finally history features. Experimental studies carried out on real and simulated *http* traffic showed that most tested attacks are correctly detected and identified using our feature set. Future work will address extending this feature set in order to take into account most Web attacks as well as building an extensive and open data set of Web related attacks.

References

1. www.ietf.org/rfc/rfc2616.txt
2. www.ietf.org/rfc/rfc2818.txt
3. www.securityfocus.com/
4. http://cwe.mitre.org/documents/vuln-trends.html
5. W. Lee, A Data Mining Framework for Constructing Features and Models for Intrusion Detection Systems, PhD thesis, Columbia University, 1999.
6. I. V. Onut, A. A. Ghorbani, A Feature Classification Scheme for Network Intrusion Detection, In International Journal of Network Security, Vol. 5, N. 1, pp. 1-15, July 2007.
7. Common Vulnerabilities and Exposures, http://cve.mitre.org/
8. C. Krugel, G. Vigna: Anomaly detection of web-based attacks, ACM Conference on Computer and Communications Security, pp:251-261, 2003.
9. C. Kruegel, G. Vigna, W. Robertson, A multi-model approach to the detection of web-based attacks, In Computer Networks 48(5), pp. 717-738, 2005.
10. A. Riancho, w3af - Web Application Attack and Audit Framework, 2007. http://w3af.sf.net/
11. J. Undercoffer and J. Pinkston, Modeling Computer Attacks: A Target-Centric Ontology for Intrusion Detection, The Sixth International Symposium on Recent Advances in Intrusion Detection, Pittsburgh, PA, 2003.
12. www.snort.org/
13. K. L. Ingham, Anomaly Detection for HTTP Intrusion Detection: Algorithm Comparisons and the Effect of Generalization on Accuracy, PhD thesis, B.S. cum laude, University of New Mexico & M.S., University of New Mexico, May 2007.
14. http://www.i-pi.com/HTTP-attacks-JoCN-2006.
15. J. R. Quinlan,C4.5 : Programs for Machine Learning, Morgan Kaufmann Pub., San Mateo, CA, 1993.
16. C. Elkan, Results of the KDD'99 Classifier Learning, In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA 1(2), pp.63-64, 2000.
17. S. Benferhat, K. Tabia, Y. Djouadi, Integrating Anomaly-Based Detection with Bayesian-Decision tree Classifiers, International Conference on Advances in Information and Communication Technologies ICICOT07, Manipal, India, 2007.