A Project report on

Application of Robust Software Modelling Tool for Web Attacks Detection

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

KALLURI RISHITA (20H51A0535)
LANKA SHRIYA (20H51A05E5)
BALLA GANESH (20H51A05B1)

Under the esteemed guidance of

Mr. A. Vivekanand
(Assistant Professor)

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING& TECHNOLOGY
(UGC Autonomous)
*Approved by AICTE *Affiliated to JNTUH *NAAC Accredited with A+ Grade
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2020- 2024

CMR COLLEGE OF ENGINEERING & TECHNOLOGY
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Major Project Phase I report entitled **"Application of Robust Software Modelling Tool for Web Attacks Detection"** being submitted by Kalluri Rishita (20H51A0535), Lanka Shriya (20H51A05E5), Balla Ganesh (20H51A05B1) in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

Mr. A. Vivekanand
Assistant Professor
Dept. of CSE

Dr. Siva Skandha Sanagala
Associate Professor and HOD
Dept. of CSE

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Mr. A. Vivekanand, Assistant Professor** , Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala,** Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, We extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

Kalluri Rishita 20H51A0535
Lanka Shriya 20H51A05E5
Balla Ganesh 20H51A05B1

TABLE OF CONTENTS

List of Tables

ABSTRACT

Web applications are popular targets for cyber-attacks because they are network-accessible and often contain vulnerabilities. An intrusion detection system monitors web applications and issues alerts when an attack attempt is detected. Existing implementations of intrusion detection systems usually extract features from network packets or string characteristics of input that are manually selected as relevant to attack analysis. Manually selecting features, however, is time-consuming and requires in-depth security domain knowledge. Moreover, large amounts of labeled legitimate and attack request data are needed by supervised learning algorithms to classify normal and abnormal behaviors, which is often expensive and impractical to obtain for production web applications. This paper provides three contributions to the study of autonomic intrusion detection systems. First, we evaluate the feasibility of an unsupervised/semi-supervised approach for web attack detection based on the Robust Software Modeling Tool (RSMT), which autonomically monitors and characterizes the runtime behavior of web applications. Second, we describe how RSMT trains a stacked denoising autoencoder to encode and reconstruct the call graph for end-to-end deep learning, where a low-dimensional representation of the raw features with unlabeled request data is used to recognize anomalies by computing the reconstruction error of the request data. Third, we analyze the results of empirically testing RSMT on both synthetic datasets and production applications with intentional vulnerabilities. Our results show that the proposed approach can efficiently and accurately detect attacks, including SQL injection, cross-site scripting, and deserialization, with minimal domain knowledge and little labeled training data.

CHAPTER 1
INTRODUCTION

1. INTRODUCTION

1.1. Introduction

Web attack detection refers to the process of identifying and preventing unauthorized and malicious activities aimed at web applications and their users. It involves deploying various security mechanisms, tools, and techniques to recognize patterns or behaviors indicative of an attack. These attacks can be broadly categorized into server-side attacks, where the attacker exploits vulnerabilities in the web server or application, and client-side attacks, where the attacker targets the end-user's browser or device.

Halfond, W. G., Viegas, J., & Orso, A [1], Web applications are vulnerable to cyber attacks, with common attacks including SQL injection cross-site scripting Wassermann, G., & Su, Z. [2], and remote code execution. Despite the development of counter-measures like firewalls and intrusion detection systems Di Pietro, R., & Mancini, L. V [3], web attacks remain a significant threat. Research shows that over half of web applications during a 2015-2016 scan contained significant security vulnerabilities. False positive limitations Pietraszek, T. [9] require manual selection of attack-specific features and high false positive rates, making it essential to reduce these systems. An infrastructure that requires less expertise and labeled training data is needed to address these challenges. Fu, X., Lu, X., Peltsverger, B., Chen, S., Qian, K., & Tao, L [12] struggle due to workforce limitations, classification limitations, and false positive limitations. Workforce limitations involve in-depth domain knowledge of web security, while classification limitations involve large amounts of labeled training data and the difficulty of obtaining it for arbitrary custom applications.

1.2. Problem Statement

Our proposed system deals with different types of web attack such as cross-site scripting, sql injection, database attacks etc., algorithms like RSMT, SVM, Naive Bayes, LSTM are used where SVM and naive bayes comes under machine learning algorithms and LSTM comes under deep learning algorithm which require large, labeled datasets for supervised learning.

1.3. Project Objective

In this project we are mainly addressing 3 objectives that are :

- Cross-Site Scripting (XSS) is a common web vulnerability where malicious scripts are injected into web pages, stealing data, manipulating sessions, or redirecting users to harmful sites. Prevention involves input validation, output encoding, secure cookie handling, Content Security Policy implementation, and developer education.

- SQL Injection is a prevalent cybersecurity threat involving the manipulation of input fields to insert malicious SQL code, granting unauthorized access to sensitive data or enabling record modifications. Prevention methods include parameterized statements, input validation, least privilege principles, and routine security audits.

- Database attacks compromise security, exploiting software vulnerabilities or weak authentication to steal information, manipulate data, and disrupt services. Prevention involves strong authentication, regular software updates, data encryption, intrusion detection systems, and frequent security audits.

In our proposed solution we are using RSMT tool which is a web monitoring tool which monitors execution behavior of web application and record in a trace file. Trace file contains low dimensional raw data and it cannot be used for Deep Learning Network. To convert this raw data to deep learning features we are using auto encoder technique. Auto encoder will convert raw data into deep learning features. This features will be passes to propose Auto Encoder algorithm which will preprocess the data and generate train and test data from features where training data is 80% and testing data is 20%. AutoEncoder algorithm require un-label train data to generate model and new test data will be applied on AutoEncoder train model to identify new test data is a normal request or contains attack. If new test data not available in AutoEncoder train model then it will be consider as attack.

1.4 Scope and Limitations of the Project

1.4.1 Scope of the Project

This project explores an unsupervised/semi-supervised intrusion detection approach using RSMT, emphasizing efficient detection of web application attacks, including SQL injection, cross-site scripting, and deserialization. There are several other attacks like Penetration attack, Phishing attack etc. but our project cannot predict such attacks.

1.4.2 Limitations of the Project

Address limitations regarding the availability, diversity, and quality of the sequential web traffic data. Insufficient or biased data might impact the RSMT model's ability to generalize well to real-world scenarios. Acknowledge the complexity of the RSMT architecture. Complex models often require significant computational resources and longer training times. Discuss potential challenges related to computational limitations.

CHAPTER 2
BACKGROUND WORK

2. BACKGROUND WORK

In this section we have studied various implementations of web attack detections & we summarized our findings that we concluded by researching & referencing various papers.They are as below:

Halfond, W. G., Viegas, J., & Orso, A. (2006, March) [1] SQL injection attacks pose a significant security threat to web applications, allowing attackers to access sensitive information. Current methods either fail to address the full scope of the problem or have limitations. This paper reviews different types of SQL injection attacks, discusses detection and prevention techniques, and discusses their strengths and weaknesses in addressing the entire range of attacks. Future evaluations should focus on assessing techniques' precision and effectiveness in practice, using empirical evaluations to compare their performance against real-world attacks and legitimate inputs. Wassermann, G., & Su, Z. (2008, May) [2] presents a static analysis for identifying cross-site scripting (XSS) vulnerabilities in web applications. It addresses weak or absent input validation, combining work on tainted information flow with string analysis. The approach addresses the difficulty of checking for vulnerabilities statically by formalizing a policy based on the W3C recommendation, Firefox source code, and online tutorials about closed-source browsers. The paper provides effective checking algorithms and an extensive evaluation of known and unknown vulnerabilities in real-world web applications. Di Pietro, R., & Mancini, L. V. (Eds.) [3] Anomaly-based network intrusion detection systems (NIDSs) are increasingly effective in detecting attacks, as they focus on packet headers and payloads. A comparison between PAYL and POSEI-DON, two payload-based NIDSS, is presented to support this thesis.

Qie, X., Pang, R., & Peterson, L. (2002) [4] presents a toolkit to enhance code robustness against DoS attacks. It suggests that software development should focus on implementing protection mechanisms into the code itself, rather than reacting to attacks. The toolkit provides an API for programmers to annotate their code, acting as sensors and actuators for resource abuse detection. Ben-Asher, N., & Gonzalez, C. (2015) [5] explores the impact of knowledge in network operations and information security on detecting intrusions in a simple network. A simplified Intrusion Detection System (IDS) was developed to examine how individuals with or without knowledge detect malicious events. Results showed that more knowledge in cyber security improved the detection of malicious events and reduced false classifications. However, knowledge about a specific network was needed for accurate detection decisions. Expertise and practical knowledge are crucial in triage analysis, which classifies network events as threats and their connections to overall attack decisions, likely driven by the accumulation of information in cyber security. Japkowicz, N., & Stephen, S. (2002) [6] explores the class imbalance problem in machine learning, focusing on understanding its nature, comparing various re-sampling methods, and examining its impact on other classification systems like Neural Networks and Support Vector Machines. It also explores the relationship between concept complexity, training set size, and class imbalance level. Liu, G., Yi, Z., & Yang, S. (2007) [7] Existing intrusion detection models mainly detect misuse or anomaly attacks. A hierarchical model using principal component analysis (PCA) neural networks is proposed, achieving satisfactory performance in classifying network connections based on 1998 DARPA evaluation data sets.

Xu, X., & Wang, X. (2005, July) [8] proposes a novel adaptive intrusion detection method using principal component analysis (PCA) and support vector machines (SVMs). PCA reduces network data patterns dimension, while SVMs construct classification models. The method has good classification performance without parameter tuning, and is superior to SVMs without PCA in training and detection speed. Experimental results show its effectiveness. Pietraszek, T. (2004) [9] Intrusion Detection Systems (IDSs) are used to detect security violations, but they often trigger false positives, making it difficult for analysts to identify true positives. This paper introduces ALAC, an Adaptive Learner for Alert Classification, which helps reduce false positives in intrusion detection by classifying alerts into true positives and false positives. ALAC can also process autonomously high-confidence alerts, reducing the analyst's workload. The prototype implementation and machine learning technique are described. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017) [10] A deep convolutional neural network was trained to classify 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest, achieving top-1 and top-5 error rates of 37.5% and 17.0%, respectively. The network, with 60 million parameters and 650,000 neurons, used non-saturating neurons and efficient GPU implementation. The model also won the ILSVRC-2012 competition with a top-5 error rate of 15.3%.

Below table represents few other references regarding our study on end-end deep learning on web attacks, where we have studied different domines related to edge devices, cyber attack detection techniques, tools, techniques, and methodology of developing robust software, Long short-term memory, End-to-end deep learning of optimization heuristics and many more.

Table 2.1: Comparison matrix table for various research papers studied

Reference	**Author**	**Title**	**Year of Publishing**	**Results**
[14]	Tian, Z., Luo, C., Qiu, J., Du, X., & Guizani, M	A distributed deep learning system for web attack detection on edge devices	2019	Accuracy- 99.410% TPR - 98.91% Detection rate - 99.55%
[15]	Dutta, V., Choraś, M., Pawlicki, M., & Kozik, R	A deep learning ensemble for network anomaly and cyber-attack detection	2020	The proposed framework, combining DNN, LSTM, and a meta-classifier, outperformed existing methods in detecting anomalies on three diverse datasets, eliminating the need for recent network traffic datasets.
[16]	Jayaswal, B. K., & Patton, P. C.	Design for trustworthy software: Tools, techniques, and methodology of developing robust software.	2006	
[17]	Hochreiter, S., & Schmidhuber, J.	Long short-term memory	1997	LSTM outperforms real-time recurrent learning, back propagation, and Elman nets in experiments with artificial data, solving complex, long-time-lag tasks that previous algorithms have struggled with.
[18]	Cummins, C., Petoumenos, P., Wang, Z., & Leather, H.	End-to-end deep learning of optimization heuristics	2017	The network improves model accuracy by 14% and 12% without human intervention, compared to hand-picked features.
[19]	Kendall, K. K. R.	A database of computer attacks for the evaluation of intrusion detection systems	1999	The 1998 DARPA intrusion detection evaluation established the first standard corpus for evaluating computer

				intrusion detection systems, analyzing over 300 attacks from 32 types and 7 scenarios.
[20]	Ng, A.	Sparse autoencoder	2011	The notes discuss feedforward neural networks, backpropagation algorithm for supervised learning, autoencoder construction, and sparse autoencoder development, utilizing notation and symbols for clarity.

The above table, combining DNN, LSTM, and a meta-classifier, outperformed existing methods in detecting anomalies on three datasets, eliminating the need for recent network traffic datasets. It improved model accuracy by 14% and 12% without human intervention. The 1998 DARPA intrusion detection evaluation established the first standard corpus. Existing implementations of intrusion detection systems usually extract features from network packets or string characteristics of input that are manually selected as relevant to attack analysis. Manually selecting features, however, is time-consuming and requires in-depth security domain knowledge. Moreover, large amounts of labeled legitimate and attack request data are needed by supervised learning algorithms to classify normal and abnormal behaviors, which is often expensive and impractical to obtain for production web applications. While cross validation is widely used in traditional machine learning, it is often not used for evaluating deep learning models because of the great computational cost.

CHAPTER 3
RESULTS AND DISCUSSION

3. RESULTS AND DISCUSSION

In this section we have included the results of two references in which the authors have evaluated the techniques using different criteria. They have evaluated the deployment requirements of each technique.

Halfond, W. G., Viegas, J., & Orso, A. [1] The study evaluated various techniques to assess their ability to address different attack types. Most techniques were evaluated analytically, assuming developers correctly applied defensive coding practices. The techniques were divided into prevention-focused and detection-focused techniques. Prevention-focused techniques identify vulnerabilities in code, propose a different development paradigm, or add checks to enforce defensive coding best practices. Detection-focused techniques detect attacks mostly at runtime. The study used four markings to indicate how a technique performed with respect to a given attack type: "•" to stop all attacks, "×" to stop attacks, and "−" to classify techniques as only partially effective. Half of the prevention-focused techniques effectively handle all attack types considered. Some techniques are only partially effective, such as JDBC-Checker Gould, Security Gateway, SecuriFly, and overall, prevention-focused techniques performed well because they incorporate defensive coding practices in their prevention mechanisms. Most detection-focused techniques performed fairly uniformly against various attack types, except for the IDS-based approach.

Table 3.1: Comparison of detection-focused techniques with respect to attack types

Reference	Taut	Illegal	Piggy-back	Union	Stored Proc	Infer	Alt. Encodings
[26]	●	●	●	●	X	●	●
[33]	●	●	●	●	X	●	X
[36]	○	○	○	○	○	○	○
[25]	-	-	-	-	-	-	-
[35]	●	●	●	●	X	●	●
[22]	●	●	●	●	X	●	●
[21]	●	X	●	●	X	●	X
[37]	●	X	X	X	X	X	X
[32]	●	●	●	●	X	●	X

Table 3.2: Comparison Matrix table for sql Injection Attack

Reference	Taut	Illegal	Piggy-back	Union	Stored Proc	Infer	Alt. Encodings
[24]	-	-	-	-	-	-	-
[29]	●	●	●	●	●	●	●
[23]	●	●	●	●	X	●	●
[34]	-	-	-	-	-	-	-
[30]	-	-	-	-	-	-	-
[31]	●	●	●	●	X	●	●
[27]	○	○	○	○	○	-	○
[28]	●	●	●	●	●	●	●

Source: Halfond, W. G., et all

In this section, authors have evaluated the techniques presented in above table using several different criteria. They first considered which attack types each technique is able to address. For the subset of techniques that are based on code improvement, they look at which defensive coding practices the technique helps enforce. They then identify which injection mechanism each technique is able to handle. Finally, they have evaluated the deployment requirements of each technique.

The study evaluates various attack types and finds that prevention-focused techniques perform well due to their application of defensive coding best practices. Each technique is classified based on the defensive coding practices it enforces, as shown in table.

This table presents a survey and comparison of current techniques for detecting and preventing SQLIAs. It identifies various types of SQLIAs, evaluates their ability to detect and prevent them, and studies the mechanisms through which SQLIAs can be introduced into applications. The study also identifies a distinction in prevention abilities between prevention-focused and general detection and prevention techniques. Future evaluation should focus on their precision and effectiveness in practice.

Mokarian, A[64] used data mining techniques to identify alarm sequences and create filters for intrusion detection systems (IDS). The effectiveness of intrusion detection systems (IDS) is largely determined by their ability to accurately classify events as normal or attack. The confusion matrix, which shows four possible outcomes, helps evaluate IDS performance. False positive rate (FPR), detection rate (TN), and accuracy (TP) are key parameters. A high FPR can lead to low performance and vulnerability to intrusions. To have an effective IDS, both FP and FN rates should be minimized, along with maximizing accuracy and TP and TN rates. Effective techniques should reduce false positives rates while increasing system accuracy or keeping it constant.

Table 3.3: Comparison Matrix table for False Positive Reduction techniques

References	**FPRT**	**KDD**	**KDD**	**KDD**	**KDD**	**KDD**	**Results**
[43]	SVM	*					1.00%
[43]	C4.5	*					1.44%
[44]	Decision Tree Classification, Rule based classification	*					3.2%
[45]	Decision Tree Classification, Bayesian Clustering	*					NA
[46]	Self-Organizing Map, K-Means Clustering	*				*	0.91 – 2.43%
[47]	Sequential Association Mining						NA
[48], [49]	Clustering (Attribute Oriented Induction)					*	75%, 87%
[41], [50], [51]	Machine Learning, Clustering			*		*	30%, 50%

[52]	Quality Parameters, Normalizing				*		98.03%
[53]	Multi-Level Clustering				*		NA
[54]	Clustering		*				NA
[55], [56]	Classification, clustering			*		*	37%
[57], [58], [59]	Clustering, root cause analysis		*	*		*	82%, 93%, 74%
[38], [60]	Classification, clustering					*	81% - 99%, 43.31%
[40]	Statistical Filtering			*			75%
[61]	Classification			*			36%
[63]	Clustering, GHSOM					*	15% - 4.7%
[39]	Self-Organizing Map, K-Means Clustering			*		*	90%, 87%, 50%
[42]	Rule based classification	*					NA
[62]	Fuzzy Alert Aggregation			*			NA

Source: Mokarian, A., et all

This table reviews research on reducing false positives and alert load in intrusion detection systems over the past decade. It categorizes these studies into detection techniques and alert processing techniques. Data mining techniques have gained interest as a solution to evaluate alert quality and address false positives and states that some algorithms may cause low accuracy and miss real-attack alerts.

CHAPTER 4
CONCLUSION

4. CONCLUSION

This project describes the architecture and results of applying a unsupervised end-to-end deep learning approach to automatically detect attacks on web applications. We instrumented and analyzed web applications using the Robust Software Modeling Tool (RSMT), which autonomically monitors and characterizes the runtime behavior of web applications. We then applied a denoising autoencoder to learn a low-dimensional representation of the call traces extracted from application runtime. To validate our intrusion detection system, we created several test applications and synthetic trace datasets and then evaluated the performance of unsupervised learning against these datasets. While cross validation is widely used in traditional machine learning, it is often not used for evaluating deep learning models because of the great computational cost.

REFERENCES

REFERENCES

[1] Halfond, W. G., Viegas, J., & Orso, A. (2006, March). A classification of SQL-injection attacks and countermeasures. In Proceedings of the IEEE international symposium on secure software engineering (Vol. 1, pp. 13-15). IEEE.

[2] Wassermann, G., & Su, Z. (2008, May). Static detection of cross-site scripting vulnerabilities. In Proceedings of the 30th international conference on Software engineering (pp. 171-180).

[3] Di Pietro, R., & Mancini, L. V. (Eds.). (2008). Intrusion detection systems (Vol. 38). Springer Science & Business Media.

[4] Qie, X., Pang, R., & Peterson, L. (2002). Defensive programming: Using an annotation toolkit to build DoS-resistant software. ACM SIGOPS Operating Systems Review, 36(SI), 45-60.

[5] Ben-Asher, N., & Gonzalez, C. (2015). Effects of cyber security knowledge on attack detection. Computers in Human Behavior, 48, 51-61.

[6] Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. Intelligent data analysis, 6(5), 429-449.

[7] Liu, G., Yi, Z., & Yang, S. (2007). A hierarchical intrusion detection model based on the PCA neural networks. Neurocomputing, 70(7-9), 1561-1568.

[8] Xu, X., & Wang, X. (2005, July). An adaptive network intrusion detection method based on PCA and support vector machines. In International conference on advanced data mining and applications (pp. 696-703). Berlin, Heidelberg: Springer Berlin Heidelberg.

[9] Pietraszek, T. (2004). Using adaptive alert classification to reduce false positives in intrusion detection. In Recent Advances in Intrusion Detection: 7th International Symposium, RAID 2004, Sophia Antipolis, France, September 15-17, 2004. Proceedings 7 (pp. 102-124). Springer Berlin Heidelberg.

[10] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

[11] Sun, F., Zhang, P., White, J., Schmidt, D., Staples, J., & Krause, L. (2017, June). A feasibility study of autonomically detecting in-process cyber-attacks. In 2017 3rd IEEE International Conference on Cybernetics (CYBCONF) (pp. 1-8). IEEE.

[12] Fu, X., Lu, X., Peltsverger, B., Chen, S., Qian, K., & Tao, L. (2007, July). A static analysis framework for detecting SQL injection vulnerabilities. In 31st annual international computer software and applications conference (COMPSAC 2007) (Vol. 1, pp. 87-96). IEEE.

[13] Pan, Y., Sun, F., Teng, Z., White, J., Schmidt, D. C., Staples, J., & Krause, L. (2019). Detecting web attacks with end-to-end deep learning. Journal of Internet Services and Applications, 10(1), 1-22.

[14] Tian, Z., Luo, C., Qiu, J., Du, X., & Guizani, M. (2019). A distributed deep learning system for web attack detection on edge devices. IEEE Transactions on Industrial Informatics, 16(3), 1963-1971.

[15] Dutta, V., Choraś, M., Pawlicki, M., & Kozik, R. (2020). A deep learning ensemble for network anomaly and cyber-attack detection. Sensors, 20(16), 4583.

[16] Jayaswal, B. K., & Patton, P. C. (2006). Design for trustworthy software: Tools, techniques, and methodology of developing robust software. Pearson Education.

[17] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[18] Cummins, C., Petoumenos, P., Wang, Z., & Leather, H. (2017, September). End-to-end deep learning of optimization heuristics. In 2017 26th International Conference on Parallel Architectures and Compilation Techniques (PACT) (pp. 219-232). IEEE.

[19] Kendall, K. K. R. (1999). A database of computer attacks for the evaluation of intrusion detection systems (Doctoral dissertation, Massachusetts Institute of Technology).

[20] Ng, A. (2011). Sparse autoencoder. CS294A Lecture notes, 72(2011), 1-19.

[21] Boyd, W. B., & Keromytis, D. A. (2004). SQLrand: Preventing SQL Injection Attacks In Proceedings of the 2nd Applied Cryptography and Network Security (ACNS) Conference.

[22] Buehrer, G., Weide, B. W., & Sivilotti, P. A. (2005, September). Using parse tree validation to prevent SQL injection attacks. In Proceedings of the 5th international workshop on Software engineering and middleware (pp. 106-113).

[23] Cook, W. R., & Rai, S. (2005, May). Safe query objects: statically typed objects as remotely executable queries. In Proceedings of the 27th international conference on Software engineering (pp. 97-106).

[24] Gould, C., Su, Z., & Devanbu, P. (2004, May). JDBC checker: A static analysis tool for SQL/JDBC applications. In Proceedings. 26th International Conference on Software Engineering (pp. 697-698). IEEE.

[25] Haldar, V., Chandra, D., & Franz, M. (2005, December). Dynamic taint propagation for Java. In 21st Annual Computer Security Applications Conference (ACSAC'05) (pp. 9-pp). IEEE.

[26] Halfond, W. G., & Orso, A. (2005, November). AMNESIA: analysis and monitoring for neutralizing SQL-injection attacks. In Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering (pp. 174-183).

[27] Huang, Y. W., Huang, S. K., Lin, T. P., & Tsai, C. H. (2003, May). Web application security assessment by fault injection and behavior monitoring. In Proceedings of the 12th international conference on World Wide Web (pp. 148-159).

[28] Huang, Y. W., Yu, F., Hang, C., Tsai, C. H., Lee, D. T., & Kuo, S. Y. (2004, May). Securing web application code by static analysis and runtime protection. In Proceedings of the 13th international conference on World Wide Web (pp. 40-52).

[29] Livshits, V. B. (2005). Finding security errors in Java programs with static analysis. In Proc. 14th USENIX Security Symposium, 2005.

[30] Martin, M., Livshits, B., & Lam, M. S. (2005). Finding application errors and security flaws using PQL: a program query language. Acm Sigplan Notices, 40(10), 365-383.

[31] McClure, R. A., & Krüger, I. H. (2005, May). SQL DOM: compile time checking of dynamic SQL statements. In Proceedings of the 27th international conference on Software engineering (pp. 88-96).

[32] Nguyen-Tuong, A., Guarnieri, S., Greene, D., Shirley, J., & Evans, D. (2005, May). Automatically hardening web applications using precise tainting. In IFIP International Information Security Conference (pp. 295-307). Boston, MA: Springer US.

[33] Pietraszek, T., & Berghe, C. V. (2006). Defending against injection attacks through context-sensitive string evaluation. In Recent Advances in Intrusion Detection: 8th International Symposium, RAID 2005, Seattle, WA, USA, September 7-9, 2005. Revised Papers 8 (pp. 124-145). Springer Berlin Heidelberg.

[34] Scott, D., & Sharp, R. (2002, May). Abstracting application-level web security. In Proceedings of the 11th international conference on World Wide Web (pp. 396-407).

[35] Su, Z., & Wassermann, G. (2006). The essence of command injection attacks in web applications. Acm Sigplan Notices, 41(1), 372-382.

[36] Valeur, F., Mutz, D., & Vigna, G. (2005). A learning-based approach to the detection of SQL attacks. In Detection of Intrusions and Malware, and Vulnerability Assessment: Second International Conference, DIMVA 2005, Vienna, Austria, July 7-8, 2005. Proceedings 2 (pp. 123-140). Springer Berlin Heidelberg.

[37] Wassermann, G., & Su, Z. (2004, October). An analysis framework for security in web applications. In Proceedings of the FSE Workshop on Specification and Verification of component-Based Systems (SAVCBS 2004) (pp. 70-78).

[38] Vaarandi, R. (2009, October). Real-time classification of IDS alerts with data mining techniques. In MILCOM 2009-2009 IEEE Military Communications Conference (pp. 1-7). IEEE.

[39] Tjhai, G. C., Furnell, S. M., Papadaki, M., & Clarke, N. L. (2010). A preliminary two-stage alarm correlation and filtering system using SOM neural network and K-means algorithm. Computers & Security, 29(6), 712-723.

[40] Spathoulas, G. P., & Katsikas, S. K. (2010). Reducing false positives in intrusion detection systems. computers & security, 29(1), 35-44.

[41] Pietraszek, T. (2004). Using adaptive alert classification to reduce false positives in intrusion detection. In Recent Advances in Intrusion Detection: 7th International Symposium, RAID 2004, Sophia Antipolis, France, September 15-17, 2004. Proceedings 7 (pp. 102-124). Springer Berlin Heidelberg.

[42] Sabri, F. N. M., Norwawi, N. M., & Seman, K. (2011). Identifying false alarm rates for intrusion detection system with data mining. IJCSNS: International Journal of Computer Science and Network Security, 11(4), 95-99.

[43] Wu, S. Y., & Yen, E. (2009). Data mining-based intrusion detectors. Expert Systems with Applications, 36(3), 5605-5612.

[44] Anuar, N. B., Sallehudin, H., Gani, A., & Zakaria, O. (2008). Identifying false alarm for network intrusion detection system using hybrid data mining and decision tree. Malaysian journal of computer science, 21(2), 101-115.

[45] Xiang, C., Yong, P. C., & Meng, L. S. (2008). Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees. Pattern Recognition Letters, 29(7), 918-924.

[46] Lee, S., Kim, G., & Kim, S. (2011). Self-adaptive and dynamic clustering for online anomaly detection. Expert Systems with Applications, 38(12), 14891-14898.

[47] Clifton, C., & Gengo, G. (2000, October). Developing custom intrusion detection filters using data mining. In MILCOM 2000 Proceedings. 21st Century Military Communications. Architectures and Technologies for Information Superiority (Cat. No. 00CH37155) (Vol. 1, pp. 440-443). IEEE.

[48] Julisch, K., & Dacier, M. (2002, July). Mining intrusion detection alarms for actionable knowledge. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 366-375).

[49] Julisch, K. (2003). Using root cause analysis to handle intrusion detection alarms.

[50] Pietraszek, T., & Tanner, A. (2005). Data mining and machine learning—towards reducing false positives in intrusion detection. Information security technical report, 10(3), 169-183.

[51] Pietraszek, T. (2006). Alert classification to reduce false positives in intrusion detection (Doctoral dissertation, Verlag nicht ermittelbar).

[52] Bakar, N. A., Belaton, B., & Samsudin, A. (2005, November). False positives reduction via intrusion alert quality framework. In 2005 13th IEEE International Conference on Networks Jointly held with the 2005 IEEE 7th Malaysia International Conf on Communic (Vol. 1, pp. 6-pp). IEEE.

[53] Siraj, A., & Vaughn, R. B. (2005, June). Multi-level alert clustering for intrusion detection sensor data. In NAFIPS 2005-2005 Annual Meeting of the North American Fuzzy Information Processing Society (pp. 748-753). IEEE.

[54] Long, J., Schwartz, D., & Stoecklin, S. (2006, April). Distinguishing false from true alerts in snort by data mining patterns of alerts. In Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2006 (Vol. 6241, pp. 99-108). SPIE.

[55] Perdisci, R., Giacinto, G., & Roli, F. (2006). Alarm clustering for intrusion detection systems in computer networks. Engineering Applications of Artificial Intelligence, 19(4), 429-438.

[56] Giacinto, G., Perdisci, R., & Roli, F. (2005). Alarm clustering for intrusion detection systems in computer networks. In Machine Learning and Data Mining in Pattern Recognition: 4th International Conference, MLDM 2005, Leipzig, Germany, July 9-11, 2005. Proceedings 4 (pp. 184-193). Springer Berlin Heidelberg.

[57] Al-Mamory, S. O., & Zhang, H. (2010). New data mining technique to enhance IDS alarms quality. Journal in computer virology, 6, 43-55.

[58] Al-Mamory, S. O., Zhang, H., & Abbas, A. R. (2008, June). IDS alarms reduction using data mining. In 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence) (pp. 3564-3570). IEEE.

[59] Al-Mamory, S. O., & Zhang, H. (2009). Intrusion detection alarms reduction using root cause analysis and clustering. Computer Communications, 32(2), 419-430.

[60] Vaarandi, R., & Podiņš, K. (2010, October). Network ids alert classification with frequent itemset mining and data clustering. In 2010 International Conference on Network and Service Management (pp. 451-456). IEEE.

[61] Tian, Z., Zhang, W., Ye, J., Yu, X., & Zhang, H. (2008, June). Reduction of false positives in intrusion detection via adaptive alert classifier. In 2008 International Conference on Information and Automation (pp. 1599-1602). IEEE.

[62] Maggi, F., Matteucci, M., & Zanero, S. (2009). Reducing false positives in anomaly detectors through fuzzy alert aggregation. Information Fusion, 10(4), 300-311.

[63] Mansour, N., Chehab, M. I., & Faour, A. (2010). Filtering intrusion detection alarms. Cluster Computing, 13, 19-29.

[64] Mokarian, A., Faraahi, A., & Delavar, A. G. (2013). False positives reduction techniques in intrusion detection systems-a review. International Journal of Computer Science and Network Security (IJCSNS), 13(10), 128.