Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 5, May 2013, pg.219 – 225

RESEARCH ARTICLE

Empowering Intrusion Detection Systems in Multitier Web Applications Using Clustering Algorithms

RANJITH SIVADASAN[1], **P. MADHAVAN**[2]
[1]PG Scholar, Sri Krishna College of Technology, Coimbatore, India
[2]Assistant Professor, Sri Krishna College of Technology, Coimbatore, India

[1] *Ranjithsivadasan@gmail.com;* [2] *madhrace@gmail.com*

Abstract— The Internet, which can be defined as a huge network of networks - both wired and wireless, uses the Internet Protocol Suite (TCP/IP) to make information available beyond geographical boundaries. Computing devices all through the world connect to the World Wide Web via the Client Server architecture. In this architecture, the client requests some information from a web server through a web browser. The web server connects to a database server in turn to fetch data. The connection between the web server and the database is the one that needs to be well secured. This project presents a system used to detect attacks in multi-tiered web services and classify through clustering Algorithm. This approach can create normality models of isolated user sessions that include both the web front-end (HTTP) and back-end (File or SQL) network transactions with respect to Data volumes and Classify them. The project implements a lightweight virtualization technique to assign each user's web session to a dedicated container, an isolated virtual computing environment. The system then use clustering algorithm to accurately associate each web request with the subsequent DB queries and build a causal mapping profile by taking both the web server and DB traffic into account. Finally the system is able to detect attacks both in static and dynamic websites with 100 percent accuracy by reducing the processing time.

Key Terms: - Multitier Web applications; Virtualization; Clustering algorithm; Intrusion Detection Systems; Web application security

I. INTRODUCTION

Over the past few years Web services and applications have increased both popularity and complexity. Due to the large amount of data, web services have moved to multitier design where files are uploaded to the web server run as front end and the data is outsourced to the database server run as backend. Due to their excess of presence, web services have always been the target of attacks.

In Multitier architecture, the backend database is always protected behind a firewall, so they are protected from direct attacks; however the backend systems are susceptible to attacks that use web requests as a means to exploit the backend.

In this paper, a method is used to empower the existing Intrusion detection system by using clustering algorithm to detect attacks more easily. This approach can create normality models of individual user sessions that include both Web (HTTP) requests and SQL queries, and classify them to the corresponding mapping model. By mapping each user behavior (logged queries) with the training dataset, this system can easily detect the abnormal behavior of each isolated user sessions.

II. RELATED WORK

Anomaly-Based Intrusion Detection System [6], is a system for detecting computer intrusions and misuse by monitoring system activity and classifying it as either *normal* or *anomalous*. The classification is based on heuristics or rules, rather than patterns or signatures, and will detect any type of misuse that falls out of normal system operation. This is as opposed to signature based systems which can only detect attacks for which a signature has previously been created.

Intrusion Recovery for Database-backed Web Applications [7] In this paper Users or administrators must manually inspect the application for signs of an attack that exploited the vulnerability, and if an attack is found, they must track down the attacker's actions and repair the damage by hand. When an administrator learns of security vulnerability in a web application, he or she can use WARP to check whether that vulnerability was recently exploited, and to recover from any resulting intrusions.

III. INTRUSION DETECTION SYSTEM

An intrusion detection system (IDS) is a device or software application that monitors network or system activities for malicious activities or policy violations and produces reports to a Management Station. Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system. Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, logging information about them, and reporting attempts. In addition, organizations use IDPSes for other purposes, such as identifying problems with security policies, documenting existing threats and deterring individuals from violating security policies. IDPSes have become a necessary addition to the security infrastructure of nearly every organization.

IDPSes typically record information related to observed events, notify security administrators of important observed events, and produce reports. Many IDPSes can also respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which involve the IDPS stopping the attack itself, changing the security environment (e.g. reconfiguring a firewall), or changing the attack's content

Types

For the purpose of dealing with IT, there are three main types of IDS:

A. Network intrusion detection system (NIDS)

NIDS is an independent platform that identifies intrusions by examining network traffic and monitors multiple hosts developed Network intrusion detection systems gain access to network traffic by connecting to a network hub, network switch configured for port mirroring, or network tap. In a NIDS, sensors are located at choke points in the network to be monitored, often in the demilitarized zone (DMZ) or at network borders. Sensors capture all network traffic and analyses the content of individual packets for malicious traffic. An example of a NIDS is Snort.

B. Host-based intrusion detection system (HIDS)

It consists of an agent on a host that identifies intrusions by analyzing system calls, application logs, file-system modifications (binaries, password files, capability databases, Access control lists, etc.) and other host activities and state. In a HIDS, sensors usually consist of a software agent. Some application-based IDS are also part of this category. Examples of HIDS are Tripwire and OSSEC.

C. Stack-based intrusion detection system (SIDS)

This type of system consists of an evolution to the HIDS systems. The packets are examined as they go through the TCP/IP stack and, therefore, it is not necessary for them to work with the network interface in promiscuous mode. This fact makes its implementation to be dependent on the Operating System that is being used.

IV. Methodology

Architecture Diagram

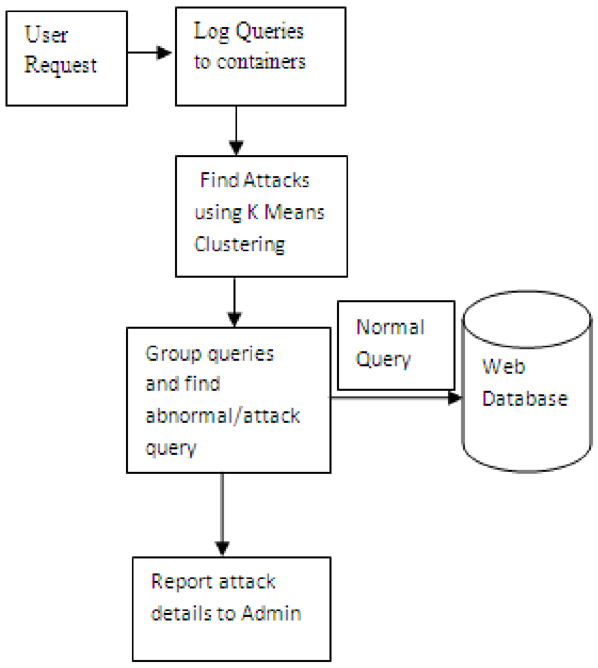
User Request
Log Queries to containers
Find Attacks using K Means Clustering
Group queries and find abnormal/attack query
Normal Query
Web Database
Report attack details to Admin

Fig 1 Architecture Diagram

A. Building normality model

A static testing website is deployed using the Content Management System assign each user session into a container, each container is assigned per each new IP address of the client. The container will log all the Web requests and SQL Queries executed by client. Deterministic Mapping and the Empty Query Set Mapping patterns are discovered from training sessions.

B. Attack scenarios

This system is effective and can reduce the computation time at detecting the following types of attacks:

1) Injection Attack

This SQL injection attack changes the structure of the SQL queries. It would generate SQL queries in a different structure that could be detected as a deviation from the SQL query structure that would normally follow such a web request.

Ex-

Original query=select * from admin where uid="1";

Suspicious query=select * from admin where uid="„ OR 1=1;--„

Here, original query is passed and suspicious query is blocked. Word-list contains the tokens of sql-query strings.

„O"-Original query

„S"-Suspecious query

Ex- („O") select * from admin where uid = „1„;

("S") select * from admin where uid = „ „ OR 1=1;--„

("O") select * from admin where uid ="1" && pwd ="abc";

(„S") select * from admin where uid = „ „ OR 1=1;--„

2) URL Manipulation

A Client manually adjusts the parameters of its request by maintaining the URL's syntax but altering its semantic meaning. This system can easily capture the attack because it would generate SQL queries that actually not match with the training sessions.

3) Privilege Escalation Attack

A privilege escalation attack is a type of network intrusion that takes advantage of programming errors or design flaws to grant the attacker elevated access to the network and its associated data and applications.

Let's consider that the dynamic website serves both regular users and administrators. For a regular user, the web request Ru will trigger the set of SQL queries Qu, and for an administrator, the request Ra will trigger the set of admin level queries Qa. Now an attacker logs into the webserver as a normal user, upgrades his privileges and act as administrator to trigger admin queries to obtain administrators data.

This approach can detect this type of attack since the DB query Qa does not match with the request Ru according to the proposed mapping model.

4) Session Hijacking

Session Hijacking is an attack by which the hacker steals this user's session identifier and then sends this session identifier as their own to the server and tricks the server into thinking they are that user. By hijacking other user sessions, the attacker can eavesdrop, send spoofed replies, and/or drop user requests. A session-hijacking attack can be further categorized as a Spoofing/Man-in-the-Middle attack, an Exfiltration Attack, a Denial-of-Service/Packet Drop attack, or a Replay attack.

According to the mapping model, if found the same session identifier from two different IP would be treated as hijacked session.

D. Mapping models

Finding deterministic mapping queries

Deterministic Mapping is the most common and perfectly matched pattern. Web request rm appears in all traffic with the SQL queries set Qn. If Qn is present in the session traffic without the corresponding rm is classified as intrusion [1].

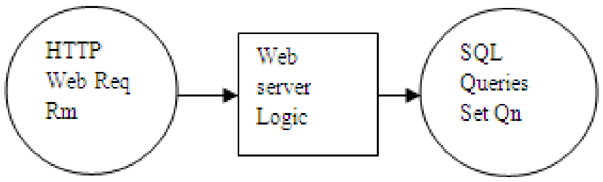

Fig 2 Deterministic Mapping

Finding Empty Query Set

In special cases, the SQL query set may be the empty set. This implies that the web request neither causes nor generates any database queries. For Example, when a web request for retrieving an image GIF file from the same webserver is made, a mapping relationship does not exist because only the web requests are observed. This type of mapping is called rm->O;. During the testing phase, keep these web requests together in the set EQS [1].

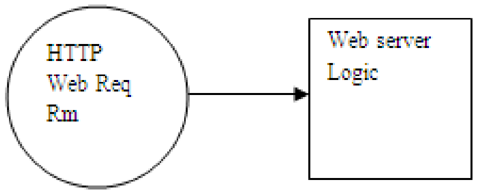

Fig 4 Empty Query Set

No Matched Request

In Some cases, DB queries cannot match up with any web requests, therefore these queries are kept in No Match Request. During testing phase any query within the set No Match Request is considered as legitimate. If found anything abnormal then it is treated as attack.

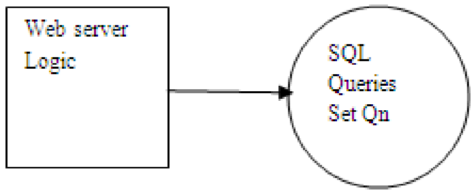

Fig 5 No Matched Request

Non Deterministic Mapping

This happens only within dynamic websites, such as blogs or forum sites. Because the same web request may result in different SQL query sets based on input parameters Each time that the same type of web request arrives, it always matches up with one (and only one) of the query sets The mapping model is Rm → Qi (Qi € {Qp, Qr, Qq}).

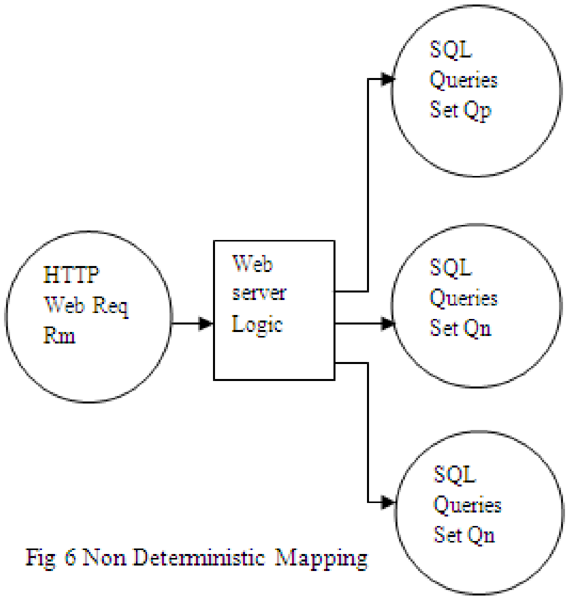

Fig 6 Non Deterministic Mapping

E. Clustering Algorithm for Attack Classification

The algorithm accepts two inputs. The data(container) itself, and "k", the number of clusters. The output is k clusters with input data partitioned among them.

The aim of K-means (or clustering) is this: To group the items into k clusters such that all items in same cluster are as similar to each other as possible. And items not in same cluster are as different as possible. The distance measures are used to calculate similarity and dissimilarity. One of the important concept in K-means is that of centroid. Each cluster has a centroid. Consider it as the point that is most representative of the cluster. Equivalently, centroid is point that is the "center" of a cluster.

The output of the clustering is given as input to above modules to get optimized result for detecting intrusion in website.

K means algorithm for attack classification

1. Initialize logged queries for container
2. Randomly choose k queries and make them as initial centroids.
3. For each point, find the nearest centroid and assign the point to the cluster associated with the nearest centroid.
4. Update the centroid of each cluster based on the items in that cluster. Typically, the new centroid will be the average of all points in the cluster.
5. Repeats steps 2 and 3, till no point queries are clustered.

V. PERFORMANCE EVALUATION

To evaluate the output for this system, different attacks have been analyzed, as discussed in Section 4.2, by implementing real spatial datasets by developing online portal website for real-estate agencies. The datasets given are 50, 150, 250 and the time required for existing and proposed method is also shown below in the table as such that the performance of the proposed system shows the low value of time taken. This system then compare the cost of the algorithms with respect for queries with influence scores. The below table shows the cost of the algorithms by varying the number k of requested results by retrieving time

TABLE I

No of Data's	**Normal Time**	**Clustered time**
50	.97 sec	.85 sec
150	1.25 sec	1.12 sec
250	1.45	1.23 sec

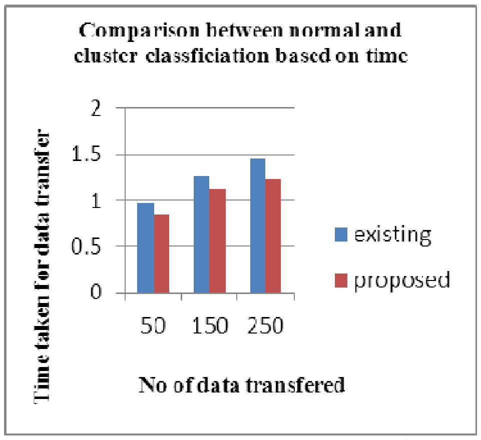

Fig. 7 Comparison between Normal and Cluster classification based on time

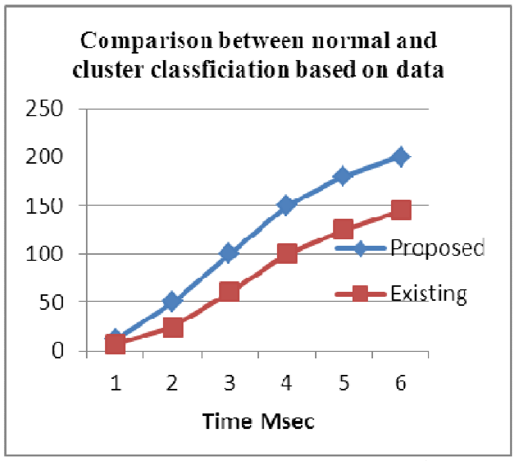

Fig. 8 Comparison between Normal and Cluster classification based on data

VI. Conclusion and Future Enhancement

We presented an intrusion detection system that builds models of normal behavior for multitier web applications from both front-end web (HTTP) requests and back-end database (SQL) queries. Unlike previous approaches that correlated or summarized alerts generated by independent IDSs, This approach forms container-based IDS with multiple input streams to produce alerts. K Means Clustering Algorithm is used efficiently to classify attacks from logged Queries. We have shown that such correlation of input streams provides a better characterization of the system for anomaly detection because the intrusion sensor has a more precise normality model that detects a wider range of threats and reduce classification time largely.

The future work of the detection is to improve accuracy of this approach when we attempted to model static and dynamic web requests with the back-end file system and database queries.

VII. Future Enhancement

For static websites, we built a well-correlated model, which our experiments proved to be effective at detecting different types of attacks. Moreover, we showed that this held true for dynamic requests where both retrieval of information and updates to the back-end database occur using the webserver front end. When we deployed our prototype on a system that employed Apache web server, a blog application, and a MYSQL back end, this approach was able to identify a wide range of attacks with minimal false positives. As expected, the number of false positives depended on the size and coverage of the training sessions we used.

REFERENCES

[1] Meixing Le, Angelos Stavrou, Brent ByungHoon Kang," DoubleGuard: Detecting Intrusions in Multitier Web Applications", IEEE Transactions On Dependable and Secure Computing , VOL. 9, NO. 4, JULY/AUGUST 2012

[2] Y. Shin, L. Williams, and T. Xie, "SQLUnitgen: Test Case Generation for SQL Injection Detection," technical report, Dept. of Computer Science, North Carolina State Univ., 2006.

[3] C. Anley, "Advanced Sql Injection in Sql Server Applications," technical report, Next Generation Security Software, Ltd., 2002.

[4] "Common Vulnerabilities and Exposures," http://www.cve.mitre. org/, 2011.

[5] "Five Common Web Application Vulnerabilities,"http://www.symantec.com/connect/articles/five-common-web-applicationvulnerabilities, 2011.

[6] "A Review of Anomaly based intrusion detection system" International Journal of Computer Applications(0975-8887) volume 28-No.7,August 2011

[7] Intrusion Recovery for Database-backed Web Applications, Ramesh Chandra, Taesoo Kim. ACM 2011