Journal of Computer Sciences

Review

Issues and Challenges in Anomaly Intrusion Detection for HTTP Web Services

[1]Mohsen Kakavand, [1]Norwati Mustapha,
[1]Aida Mustapha, [1]Mohd Taufik Abdullah and [2]Hamed Riahi

[1]*Faculty of Computer Science and Information Technology,*
University Putra Malaysia, 43400 UPM Serdang, Selangor Darul Ehsan, Malaysia
[2]*Exact Asia Development Center*

Article history
Received: 22-09-2015
Revised: 13-12-2015
Accepted: 28-12-2015

Corresponding Author:
Mohsen Kakavand
Faculty of Computer Science and Information Technology,
University Putra Malaysia,
43400 UPM Serdang, Selangor
Darul Ehsan, Malaysia
Email: kakavandirit@gmail.com

Abstract: In recent years, the development of Web-based applications has made possible novel online activities, such as banking and electronic shopping. This implies significant use of the Hypertext Transfer Protocol (HTTP) as the standard communication protocol enabler for Web services. Due to this role, HTTP has become an essential middle target of bound attacks for intruders. This paper is set to address various problems in anomaly-based intrusion detection for HTTP Web services. We seek to identify common essential methods and solutions, as well as the gaps, limitations and challenges in anomaly intrusion detection in terms of used experimental datasets, features and techniques.

Keywords: Hypertext Transfer Protocol (HTTP), Intrusion Detection, Web Services, Anomaly Detection, SOAP/XML Security

1. Introduction

A software application residing in a server generates Web content in real time and hence is known as a Web application (Corona and Giacinto, 2010). A number of online services developed in the last few years, such as search engines, online banking, email applications and social networks, are implemented as Web applications. Web applications/browsers and Web services communicate through the Hypertext Transfer Protocol (HTTP). The idea is to allow the HTTP client (Web browser/App) to request and obtain a particular response from an HTTP server (Web service) (Kapodistria *et al*., 2011). Due to its frequent use, HTTP has been researched in considerable detail and employed as the standard communication protocol for Web services. As HTTP-based Web services provide numerous kinds of online applications involving large amounts of data, the HTTP protocol has become a middle target for intruders to bound attacks for Web services (Jensen *et al*., 2009). In many cases, hackers have successfully damaged high-profile company networks and Web services (Barot and Toshniwal, 2012).

Robertson *et al*. (2006) have pointed out that many Web applications are written by people with little knowledge of security. Based on data provided by the computer emergency response teams' Coordination Center (CERT/CC), the number of relevant incidents and vulnerabilities to cyber attacks increased exponentially from 1998 to 2002. Although it has been claimed that intrusions were relatively scarce in the early 1990s, a major increase has been witnessed since 2000, with about 25,000 intrusions reported in that year alone (Malek and Harmantzis, 2004). Figure 1 shows the range of attack vectors between 1990 and 2010. Web application vulnerabilities contributed to 25% of total security issues reported in the Common Vulnerabilities and Exposures list (CVE) (Christey and Martin, 2007) and was reported to constitute a similar share according to a recent survey on dealing with security risks in digital networks (Harshini *et al*., 2011).

As shown in Fig. 1, HTTP Web services are vulnerable because they involve the execution of code on a large number of inputs and convert Universal Resource Identifier (URI) parameters to Hypertext Transfer Protocol (HTTP) POST/GET content. Moreover, the new Extensible Markup Language (XML) format allows the content of Web services to dynamically reside on the server side or the client side in the form of Web service messages. Such vulnerabilities vary with the active content technologies used.

This article provides a survey of vulnerabilities in the context of HTTP Web service intrusion detection systems of the recent past and present. We will discuss in detail issues pertaining to three different problems and challenges, namely, datasets for HTTP Web service, features set comparison and intrusion detection algorithms.

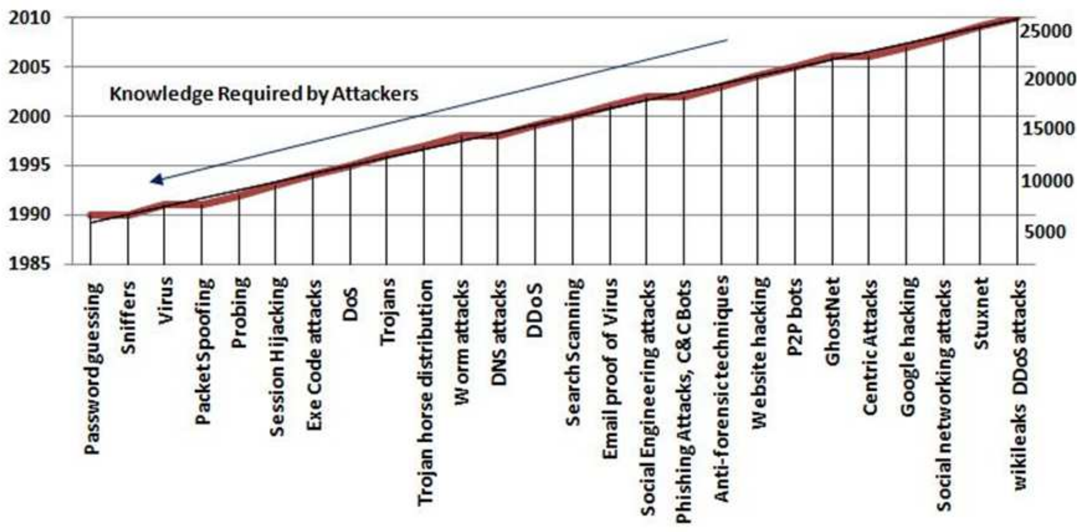

Fig. 1. Security vulnerabilities and threats

This paper is organized as follows: In section 2, we define types of intrusion detection systems and show some advantages and disadvantages of IDS approaches. In Section 3, we describe the premise of HTTP Web services' IDS and provide an attack classification scheme. In Section 4, we discuss the problems and challenges in anomaly-based intrusion detection. We offer our conclusions in Section 5.

2. Intrusion Detection System (IDS)

Intrusion Detection Systems (IDS) were introduced by (Anderson, 1980) as a response to the dramatic rise in attacks by hackers. Since then, intrusion detection has played the most important role in security after firewalls. An IDS monitors and analyzes user and network traffic, verifies system configurations and vulnerabilities and alerts the administrator through alarms. There are two main types of intrusion detection systems. The first type is based on the deployment point (network-based vs. host-based detection systems), whereas the second type is based on the detection method (misuse detection vs. anomaly detection).

2.1. IDS Placement

In general, there are two ways of deployment configurations for IDS: network-based and host-based. A network-based IDS (NIDS) is responsible for detecting intrusions in networks connected to other networks as well as to the Internet. Data appearing over the network are sequential and intrusions occur in the form of anomalous patterns. The patterns represent external hackers trying to gain unauthorized access to the network in order to steal sensitive information or interrupt the network. As researchers have pointed out, the use of NIDS has the following issues (Nadiammai and Hemalatha, 2012; Khalilian *et al*., 2011):

- A NIDS cannot determine whether an attack has been successful
- Accuracy is a challenging problem because the system can lose data during detection
- Encrypted data are problematic in NIDS because it is not possible to decrypt data at the network level
- In large-scale networks, more facilities are required to monitor the network and thus the problem of scalability occurs

In host-based IDS (HIDS), intrusion detection is performed at each host. Although the IDS must be connected to each host, the issues in HIDS are less challenging than those in NIDS. Some advantages of HIDS are as follows (Nadiammai and Hemalatha, 2012; Khalilian *et al*., 2011):

- Because detection is carried out at the host level, HIDS can detect some attacks undetected by NIDS
- HIDS are unaffected by the switching of networks
- HIDS can detect software integrity violations, such as a Trojan horse
- Because HIDS monitors only the host, it can determine intrusions more accurately
- Encrypted messages are not a serious problem for HIDS because they are received in the host and can be decrypted easily

2.2. IDS Detection Approaches

Two broad generalizations of the detection approaches among IDS are misuse detection and anomaly detection.

Table 1. Misuse detection Vs. anomaly detection

Approaches	Advantages	Disadvantages
Misuse Detection	Accurate and generates fewer false alarms	Cannot detect novel attacks and threats
Anomaly Detection	Can detect unknown attacks based on audit	limited by training data Many false alarms and

Misuse detection techniques are responsible for analyzing recorded information and comparing this with patterns inside a large attack signatures database. Patterns for specific intrusion are recorded in a large rule database. Following this, the IDS monitors all activities to find patterns matching the stored patterns. If any event matches the patterns stored in the database, the IDS warns the system of a potentially intrusive activity. The logic of such a mechanism is similar to that of virus detection techniques, but one of the main weaknesses of misuse detection techniques is its inability to detect novel intrusions. This implies that all attack patterns have to be hand coded and stored in the attack pattern database.

On the contrary, anomaly detection techniques can establish normal behavioral profiles and compare all activities with these predefined normal profiles. If any anomaly in activities occurs, the IDS warns the system of potentially intrusive action. Nonetheless, one of the main problems of this technique is the selection of the appropriate set of system features because the activities are mostly ad hoc and experience based. Hence, this technique cannot capture sequential interrelations among activities. Another problem is a high false-alarm rate because its performance is limited by the training data. Table 1 shows some advantages and disadvantages of misuse and anomaly detection (Saboori *et al*., 2010).

3. HTTP Web Services IDS

Web Services (WSs) are independent platforms that are easy to use and have powerful functionalities. They support XML/Simple Object Access Protocol (SOAP) technologies and thus are used by a variety of institutions including banks, government agencies, universities and large corporations. While Web services inherited potential security cyber threats, they also face an increasing number of new, unknown security issues. Web services mostly use application-level protocols, such as HTTP, for data exchange between service providers and requesters. Applications can communicate with each other by sending XML messages via HTTP (Vorobiev, 2006). Hence, the associated port for these protocols (i.e., port 80) needs to be accessible to provide functionality and service. This port is an easy way for attackers to infiltrate organizational networks. Typical security solutions like network firewalls, content filters and ordinary network intrusion prevention or detection systems cannot always effectively detect attackers to block their access to Web services.

A HTTP Web service intrusion detection system is designed as an in-depth defense mechanism and acts behind firewalls within the security structure of an enterprise. According to its role and location in a network, an IDS must be able to monitor packets as well as the behavior of the network elements. It observes a system or user behavior for anomalies. If it finds a suspicious situation, it raises an alarm with the administrator or response systems to prevent malicious activity. Therefore, a Web service IDS is necessary to protect against intruders. Figure 2 shows the network and service architecture of a Web Service IDS.

There are numerous possible types of attacks aimed at HTTP and SOAP/XML Web Services. In the literature, a large number of Web vulnerabilities have been recognized and identified. There are five basic classes of traditional network intrusion methods that may also affect Web application/services: Denial-of-service (DoS) attacks, user-to-root (U2R) attacks, remote-to-(Local)-user (R2U) attacks, probe attacks (probes) and data. Moreover, the Open Web Application Security Project (OWASP) has provided a list of common Web vulnerabilities, listed in Table 2 (The Open Web Application Security Project (OWASP) 2013). For the definitions, see Appendix A.

In order to be able to detect Web service attacks, it is necessary to understand the targets and features characteristic of such attacks. Web service attacks are a mixture of HTTP and its content (XML messages) sent to purposely flood and ruin the communication channel of the Web service providers (Wolter, 2013; Chonka and Abawajy, 2012). Possible Web service attacks are listed in Table 3. For more information, please refer to Appendix. B.

HTTP Web service IDS is inherently a Web Application Firewall (WAF), with intrusion detection as an additional function. They are also less complex to Web applications. Web service IDS understands XML as the basic technology of Web services. In other words, modern Web applications tend to use Web services as their building blocks and hence Web service IDS are useful to protect them (Najjar and Abdollahi Azgomi, 2010). Nonetheless, common security mechanisms, such as content filters and network firewalls, cannot properly detect intrusions and block them to prevent issues in Web services, especially when users send their requests in XML-over-HTTP format. Although firewalls protect organizational networks, they allocate two open Transmission Control Protocol (TCP) ports, for HTTP (port 80) and HTTP Secure (HTTPS) (port 447), which are used to send and receive Web services requests (Karnwal *et al*., 2012).

Because accessing HTTP is not very difficult and the data obtained through Web service requests are usually in human-readable data formats, there is a need for an intrusion detection system to provide extra security features for Web services that enable the analysis of HTTP headers and payloads (Fielding *et al*., 1999).

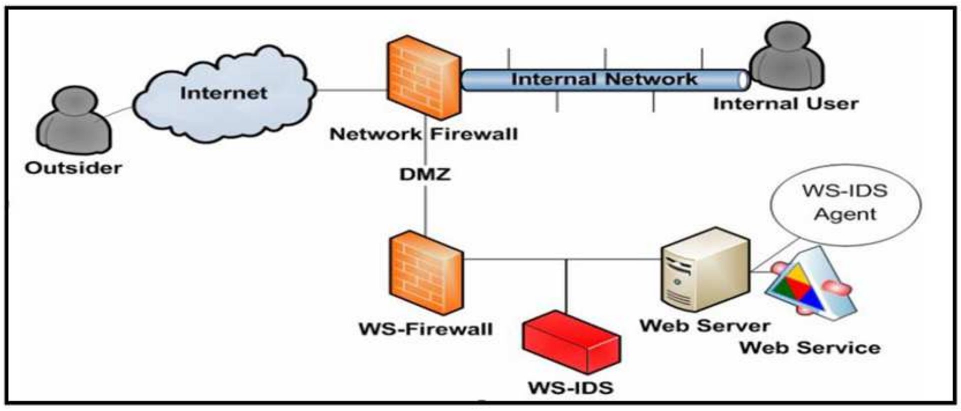

Fig. 2. Web services IDS in a typical network (Jensen *et al*., 2009)

Table 2. Web vulnerabilities (The Open Web Application Security Project (OWASP) 2013)

Injection	Sensitive data exposure
Broken authentication and session management	Missing Function-level Access Control
Cross-site Scripting (XSS)	Cross-site Request Forgery (CSRF)
Insecure Direct Object References	Using Components with known vulnerabilities
Security Misconfiguration	Unvalidated redirects and forwards

Table 3. Web applications attacks

Identity attacks (Moradian and Science, 2006)	Oversized Payloads (Yee *et al*., 2007)	XDoS Attack (Moradian and Science, 2006)
Session attacks (Moradian and Science, 2006)	Schema poisoning (Moradian and Science, 2006)	WSDL Attacks (Moradian and Science, 2006)
Replay attacks (Yee *et al*., 2007)	Overflow attacks (Seo *et al*., 2004)	WSDL Scanning (Park and Park, 2008)
Man-in-the-middle attack (Moradian and Science, 2006)	Code attacks (Park and Park, 2008)	Parameter Tampering (Seo *et al*., 2004)
Parsing attacks (Moradian and Science, 2006)	SQL Injection (Seo *et al*., 2004)	XML Attacks (Vorobiev, 2006)
Recursive payloads (Yee *et al*., 2007)	XPath Injection (Moradian and Science, 2006)	SOAP Attacks (Yee *et al*., 2007)

4. Problems and Challenges in Anomaly-based IDS

As mentioned, there are two detection approaches for IDS; signature- or misuse-based detectors and anomaly-based detectors.

A new generation of systems is now emerging based on anomaly detection. The basic principle underlying anomaly detection systems is that every anomalous event is suspicious. Anomaly-based detection systems model normal or expected behavior in a system and detect deviations of interest that may indicate a security breach or an attempted attack. Anomaly-based detection is often said to be a more powerful mechanism, due to its theoretical potential for addressing novel or unforeseen attacks (Estévez-Tapiador *et al*., 2004). In this section, we discuss the problems and challenges in anomaly-based intrusion detection based on three different properties: dataset, features set and detection algorithm.

4.1. Dataset for HTTP Web Service

Preparing a suitable dataset is critical to any experiment. For research in intrusion detection, the ideal experimental setup employs data collected from the server that is going to be protected, so that the training and testing data represent the Web server as well as attacks experienced by the server. Nonetheless, data from servers are often problematic due to privacy issues, which prevents researchers from testing them, hence limiting comparability. This limitation has forced researchers to use openly available and less representative data, while others have resorted to using closed and private, but more accurate and reliable, datasets. Open and public datasets are beneficial when comparing intrusion detection algorithms, but in-depth analysis is often required to combine, examine and compare the results.

One example of a public dataset is the Defense Advanced Research Projects Agency (DARPA)/Massachusetts Institute of Technology (MIT) Lincoln Laboratories, which has generated and published the most prominent dataset for IDS testing (1998-1999) (Lippmann *et al*., 2000). Many experiments in IDS research have used this dataset because of its large volume. Further, this dataset enables direct comparison with original lab tests. However, it also has its critics, for it is quite dated and Web behavior has evolved significantly since its publication. Despite criticism (Wang and Stolfo, 2004; Mahoney, 2003; Mahoney and Chan, 2002; Caulkins *et al*., 2005; Dokas *et al*., 2005; Perdisci *et al*., 2009), as well as (Jamdagni *et al*., 2009;

2010), have all used this dataset to testing their IDS. Meanwhile, Estévez-Tapiador *et al.* (2004) used this dataset to represent a network's normal behavior, but constructed their own attack database to supplement attacks provided by the Lincoln Labs data.

Considering the limitations and shortcomings of the Lincoln Labs data, other researchers have used private data sourced from real servers protected by IDS. However, these data are not available for other scholars to use and hence renders direct comparison impossible. For instance, Kruegel *et al.* (2005) evaluated their IDS using comprehensive normal datasets from different sources (including Google, the University of California–Santa Barbara (UCSB), the Vienna University of Technology (TU Wien)) along with attacks against software executed on one of their data sources (Web servers). Wang and Stolfo (2004) used data obtained from their an Web server (CUCS) as an additional source of data. However, they did not filter attacks from the dataset and only used it for testing. Tombini and Ducass (2004) gathered data from two production sources, academic and industrial Web servers, with a total of more than five million HTTP requests from Web server log files. Estévez-Tapiador *et al.* (2004) made use of 1,500 attack requests representing variants of 85 attacks (the largest attack database reported to date). We therefore recognize and identify the following essential gaps and challenges regarding these datasets:

- Commonly used public datasets generated data, but there was no test to prove that this generated data accurately represented real data and no official test to verify that the attacks were a reflection of real attacks
- The Lincoln Labs datasets have only four Web attacks suitable for studies in HTTP Web service intrusion detection
- The Lincoln Labs datasets are obsolete and, when tested on a wide scope, their performance suffers because Web behavior has changed significantly over the years
- Web server log files are a famous data source, but they contain only a small segment of most HTTP requests and attacks that do not exist in the resource path are unlikely to appear in the server log files
- Data obtained from private datasets are usually unavailable for others to use, especially in terms of the payload information, which causes the elimination of direct comparisons
- Researchers will choose either to use popular (outdated) data sources or to create their own data instead of using newly available created data sources. This causes a lack of comprehensive evaluation and comparison of the detection models/systems in question

In essence, such limitations often render many used and available datasets as ineffective and unfit to evaluate research results.

4.2. Feature Set Comparison and Recommendation

Data preprocessing and candidate feature extraction from the dataset are critical steps in intrusion detection as they can effectively increase the performance and accuracy of the detection model. Data preprocessing relies on expert knowledge to recognize the most relevant sections of network traffic and build the basic candidate set of data features. Thus, the selection of feature types is essential to determine the coverage or capabilities of the particular intrusion detector. For instance, the same feature types may enable detection of different attack classes, such as probe, DoS, R2Lm and U2R.

In intrusion detection research, the initial source of selecting features will normally be the network packets. From the point of view of feature selection, packet information can be divided into two parts: packet header and payload. The analysis of packet header information usually minimizes data preprocessing necessities. Headers form only a small segment of the entire network data, because of which processing requires fewer resources, such as CPU, memory and storage. Moreover, features from the packet header have advantages of being fast, with fairly low memory overhead and relatively simple computation and assuage legal and privacy-related concerns regarding network packet analysis. Because of these advantages, many researchers have used packet headers as the main feature in intrusion detection systems (Caulkins *et al.*, 2005; Dokas *et al.*, 2005; Lee and Stolfo, 2001; Lee *et al.*, 2002; Chan *et al.*, 2013). However, although request header feature sets have their own capabilities, they cannot be applied directly to detect attacks bound for the application layer because the attack bytes are typically embedded in the body of the request.

Payload data analysis is comparatively more expensive than packet header data analysis in the sense that it requires deeper packet inspection and a greater number of computations because it deals with a variety of content types (e.g., HTML, XML, etc.) and obfuscation analysis methods. Due to the complexity of payload data analysis, most relevant research focuses on small subsets of the payload data or only the client-side sections of Web content (Estévez-Tapiador *et al.*, 2004; Wang and Stolfo, 2004; Perdisci *et al.*, 2009; Jamdagni *et al.*, 2010; Kruegel and Vigna, 2003; Kruegel *et al.*, 2005; Ezeife *et al.*, 2008).

However, a packet by itself cannot always be used to identify unusual patterns over time. In especially noisy attacks, packet headers or the payload might be normal, but their trends or repetition over time may be abnormal, such as worm propagation, DoS attacks and tunneling behavior. To detect these attack trends, constructing features through multiple application sessions (connection) or a single application session must be taken into account.

Features from multiple connections are generally extracted over a time frame of connections. Most of these attributes are volume based, such as the number of connections to a specific destination address (Internet Protocol (IP)) and port in a given time frame. They help detect anomalous traffic volumes belonging to network attacks, such as scanning behavior or DoS. Features derived from single connections help identify irregular trends across multiple packets, but during a single connection, they provide context whereby contextual anomalies can be found (e.g., indications of a protocol tunneling attack).

Based on the data type and the required analysis, intrusion detection based on packet headers arguably requires less domain knowledge, whereas building content or payload-based anomaly detectors requires greater domain knowledge (to apply over relevant parts of the request content). Table 4 lists the features of anomaly-based intrusion detection based on both header and payload-based attacks following single or multiple session (connection) approaches. The list, however, is not in any order or feature grouping and only aims to display some of the possible features from HTTP packets.

Table 4. Basic features of network traffic

Time	Flag	Character distribution	Source port
Duration	Payload length	Attribute presence or absence	Destination IP address
Source hosts	Payload histograms	Payloads byte frequency distribution	Destination port
Destination hosts	Attribute order	Frequent parameter	Parameter value length
Bytes	Source IP address	Inter-request time delay	Access frequency
Service	Attribute length	Header information	Frequent order

Table 5. Packet header and payload data preprocessing

Model/Ref.	Data input	Feature	Data preprocessing	Detection
MADAM ID (Lee *et al*., 2002)	Packet header	Time, duration, src, dst, bytes, service, flag	Association rules and frequent episodes are applied to network connection records	PROBING, U2R, DoS, R2L
(Tapiador *et al*., 2004)	Packet payload	Payload length, payload histograms	Statistical analysis of payload length, mean probability density and standard deviation	HTTP attacks
PAYL (Wang *et al*., 2004)	Packet payload	Payload byte frequency distribution	1 g used to compute byte-frequency distribution models for each network destination	Worms, Probe, DoS, R2L, U2R
(Lee *et al*., 2002)	Packet header	Harder information bytes	Use of supervised learning techniques on a binary target variable and the use of oversampling.	PROBING, U2R, DoS, R2L
Kruegel *et al*. (2005; 2003)	HTTP Web requests	Length, character distribution, presence or absence, order, access frequency, inter-request time delay	Construct content-based features from user-supplied parameters in the URL	Buffer overflow, Directory traversal, XSS, input validation, code red
DMNID (Dokas *et al*., 2005)	Packet header	Start time, duration, service type, source IP address and port, destination IP address and port	Tcptrace utility software as the packet filtering tool in order to extract information regarding packets from TCP connections.	PROBING, U2R, DoS, R2L
McPAD (Perdisci *et al*., 2009)	Packet payload	Payload byte frequency distribution	2v grams extracted from payload. Feature clustering used to reduce dimensionality	Shellcode attacks on Web servers
SensorWebIDS (Ezeife *et al*., 2008)	HTTP Web requests	Parameter value length, frequent parameter presence, frequent order, frequent value length	Network sensor for extracting parameters and a log digger for extracting parameters from Web log files	XSS, SQL injection, DoS, buffer overflow, cookie poison,
Jamdagni *et al*. (2009; 2010)	Packet payload	Payload byte frequency distribution	Using Wireshark based on four conditions (size of payload, destination address, services and direction of traffic flow)	PROBING, U2R, DoS, R2L
FARM (Chan *et al*., 2013)	Packet header	Input value, Input size, SOAP size, XML content	Validating User ID, password, service request input values, input size and SOAP size to from associative patterns and then matching these patterns with interesting rules obtained from fuzzy association rule mining.	

Table 5 summarizes the preprocessing required for each feature set depending on whether the relevant dataset is sourced from the packet header or the payload content. Based on Table 5, we recognize the following challenges in building an appropriate feature set in IDS research:

- Although methods for extracting distinctive features of packet headers are well acknowledged, approaches for packet contents (payloads) are less strongly defined
- Header features may still be applicable to monitor internal networks, network management and data behavioral analysis, but they are insufficient for Web service anomaly-based IDS
- Building packet header features requires less domain knowledge than extracting payload anomaly detectors to employ over relative parts of the request content
- Features from packets (header and payload) may not be able to identify abnormal trends and patterns over time. In particularly noisy attacks, individual requests may be normal, but their trends or repetition over time may be anomalous
- Most relevant research focuses on attacks occurring in requested resource paths while other attacks target other regions of the request. In this case, how much of the HTTP request the detection model is used to extract the feature is important
- There is a lack of the use of data-mining and preprocessing techniques, such as data transformation, discretization, data cleaning and reduction. They are helpful in boosting the power, efficiency and accuracy of detection models

4.3. Intrusion Detection Algorithms

Several techniques/methods have been proposed to solve the general intrusion detection problem. The approaches can be broadly categorized into machine learning/data mining and statistical. The following works have proposed detection algorithms that use statistical techniques to detect anomalous attacks.

Estévez-Tapiador *et al.* (2004) used Markov chains to model the structure of HTTP traffic. The packet payload was parameterized for the evaluation of requests from incoming traffic: It was partitioned into a specific number of contiguous blocks, which were subsequently quantized based on a trained scalar codebook. The temporal sequence attained for the symbols was then evaluated by means of a model (Markov) obtained from the training phase.

Dokas *et al.* (2005) proposed distance-and density-based outlier detection schemes, such as Nearest Neighbor (NN), the Mahalanobis Distance Map (MDM) and a Local Outlier Factor (LOF)-based approach. Experiments using real network data showed that the LOF approach was the most successful technique among these for detecting novel intrusions.

PAYL (Wang and Stolfo, 2004) is a payload-based anomaly detector that is fully automatic and unsupervised. It first computes during a training phase a profile byte frequency distribution and their standard deviation of the application payload flowing to a single host and port and then uses the Mahalanobis distance map during the detection phase to calculate the similarity between new data and the pre-computed profile. Buffer overflow attacks often have a distinctive character distribution. Wang and Stolfo (2004) used a character distribution metric on similarly sized packets.

The Geometrical Structure Anomaly Detection (GSAD) model, (Jamdagni *et al.*, 2010) use distance-based outlier detection methods, such as Mahalanobis distance map to discover hidden correlations between features and packets.

Deterministic Finite Automaton (DFA) induction (Ingham *et al.*, 2000) can detect malicious Web requests. This is applied in combination with rules to reduce the variation of requests and heuristics for filtering and categorizing anomalies. Using this setup, a wide range of attacks is detectable with few false positives, even when the system is trained on data containing benign attacks.

Kruegel and Vigna (2003; Kruegel *et al.*, 2005) analyzed client queries that reference server-side programs. The analysis techniques used by their proposed tool took advantage of the particular structure of HTTP queries containing parameters. They developed a linear combination of nine measures (attribute length, attribute character distribution, structural inference, token finder, attribute presence, attribute order, access frequency, inter-request time delay and invocation order) and applied them to Common Gateway Interface (CGI) parameters. The access patterns of such queries and their parameters were then compared with established profiles specific to the program or active documents being referenced. A combination of intrusion detection systems is a rational step if more than one IDS is available.

The model and system created by Kruegel and Vigna (2003; Kruegel *et al.*, 2005) was limited to HTTP CGI requests and included a linear combination of the length, order, character distribution and existence of CGI parameter values. Moreover, it consisted of a test for which CGI values were enumerated (or randomized) and a Markov model to learn the pattern of these values.

Data mining or machine-learning algorithms have also been successfully applied to anomaly-based intrusion detection. For example, the association rules method has been deployed to determine correlations on features obtained following pre-processing.

Table 6. Performance of anomaly detection algorithms

Main algorithm	Dataset	Detection Rate (%)	False positive (%)	Ref.
RIPPER classifier (JRIP)	DARPA1998	80.2	-	Lee *et al.* (2002)
Markov chains	DARPA1999	95.0	40.000	Estévez-Tapiador *et al.* (2004)
Mahalanobis Distance Map (MDM)	DARPA1999	98.0	0.100	Wang *et al.* (2004)
Models of normal usage created for each Web app. Compare requests to models.	Google UCSB TU Vienna	99.0	0.060	Kruegel *et al.* (2005)
Nearest Neighbor (NN)	DARPA1998	78.9	2.000	Dokas *et al.* (2005)
Mahalanobis Distance Map (MDM)		52.6	2.000	
Density-based Local (LOF)		73.7	1.000	
Unsupervised SVM		84.2	4.000	
Association Rule Mining (ARM)	Private	98.3	-	Ezeife *et al.* (2008)
Deterministic Finite Automata (DFA)	Private	-	0.100	Ingham *et al.* (2000)
One-class SVM	DARPA 1999	95.0	0.010	Perdisci *et al.* (2009)
Mahalanobis Distance Map (MDM)	DARPA 1999	100.0	0.087	Jamdagni *et al.* (2010)
Fuzzy Association Rule Mining (FARM)	Private	99.0	0.100	Chan *et al.* (2013)

Some researches (Lee and Stolfo, 2001; Lee *et al.*, 2002) proposed a method to determine when the training audit data are sufficient. The idea is to apply a number of frequent association rules as an indicator to determine whether the audit data are sufficient, apply the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) classifier to labeled datasets and learn about intrusions.

Dokas *et al.* (2005) proposed unsupervised outlier detection schemes, such as an unsupervised Support Vector Machine (SVM)-based approach. Experiments using real network data showed that the unsupervised SVMs was promising in the detection of new intrusions but had a high false-alarm rate. To combat Web intrusions (Ezeife *et al.*, 2008), Sensor Web IDS utilized novel Web IDS, an algorithm based on the empirical rule of the theory of standard deviation (δ), of 99.7% of data lying within 3δ of the mean, to calculate the possible maximum length of input parameters. The association rule mining technique has been employed to mine frequent parameter lists and their sequential orders to identify anomaly and misuse intrusion.

Unsupervised or unlabeled learning approaches for anomaly detection have been recently proposed. Such anomaly-based network IDS can detect (unknown) zero-day attacks, although considerable care has to be dedicated to controlling the amount of false positives generated by the detection system (Perdisci *et al.*, 2009). The current version of McPAD, a new accurate payload-based anomaly detection system, consists of an ensemble of one-class SVMs.

Chan *et al.* (2013) introduced a Fuzzy Association Rule Model (FARM) for SOAP-or XML-based attacks by validating inputs and feed them into FARM. Fuzzy association rule model is a new data mining anomaly detection system proposed to solve network security problems, especially for Web service-based e-commerce applications. Even though there are still some open challenges in intrusion detection. Table 6 summarizes available intrusion detection algorithms in both of the above categories.

The detection algorithms listed in Table 6 include one-class SVM (Perdisci *et al.*, 2009), MDM (Jamdagni *et al.*, 2009; 2010), RIPPER (Lee and Stolfo, 2001; Lee *et al.*, 2002), Markov chains (Estévez-Tapiador *et al.*, 2004), Nearest Neighbor (NN), unsupervised SVM (Dokas *et al.*, 2005) and Association Rule Mining (ARM) (Ezeife *et al.*, 2008). According to the reviewed results, one-class SVM (Perdisci *et al.*, 2009) and MDM (Jamdagni *et al.*, 2009; 2010) recorded notably high accuracy compared to the other detection techniques. Nonetheless, although the accuracies for these techniques might be acceptably high, they suffer from high false-positive rates, which is an arguably dependent metric for accuracy. Only two detection algorithms, DFA (Ingham *et al.*, 2000) (token-based algorithm) and FARM (Chan *et al.*, 2013), reported high accuracies as well as low false-positive rates.

With regard to the origins of these detection algorithms, SVM is a data mining technique whereas MDM is a statistical technique. Both techniques have reported successful results with the DARPA 1999 dataset and hence are deemed reliable as our benchmark algorithms for HTTP Web service anomaly detection. However, we believe that their capabilities should also be tested and proved against private datasets. We can conclude that unsupervised machine-learning and statistical techniques have yielded successful results (Perdisci *et al.*, 2009; Jamdagni *et al.*, 2009; 2010) and might be reliable for anomaly detection in HTTP Web service requests. However, the nature of anomalous records change constantly, which renders impossible the accurate detection of outliers in data requested of HTTP Web services. At the same time, based on the limitations of quadratic computational complexity, possible inaccuracy occurs when handling high-dimensional data (Amer *et al.*, 2013). In summary, the problems and challenges for intrusion detection algorithms can be formulated as follows:

- Researchers who have used unsupervised methods (one-class SVM and MDM) reported high accuracy values on public datasets, but never tested their detection models on private data sources. Therefore, based on the limitations of public datasets, the accuracy and reliability of the proposed algorithms have not been decidedly proven
- Existing studies have shown many successful unsupervised machine-learning methods in anomaly detection. Nonetheless, most of these works do not consider constant changes in Web services data, hence making it difficult to accurately detect intrusions over HTTP Web service request data. Further, based on the limitations of quadratic computational complexity, there is always possible incorrectness when handling high-dimensional data
- The majority of work reporting notably high detection accuracies suffered from high levels of false-positive rates. Thus, considerable care has to be taken to control the number of false positives generated by the detection system, as it a complementary metric to accuracy
- A few studies have used machine-learning (supervised and unsupervised) techniques in IDS. Trying such methods might help improve the efficiency, performance and accuracy of detection models (based on the success of results in other domains)
- Most active anomaly-based intrusion detection can only detect attacks on network layers and computer systems. The network behavior and patterns among HTTP Web services have also evolved over the years, as have the types of target features (such as frequency, size, variety, etc.) and the type of attacks (SOAP/XML over HTTP intrusions). At present, no sensitivity analyses have been conducted on payload packet features in HTTP Web services, especially SOAP-based XML intrusions

5. Conclusion

We began this paper by introducing HTTP Web services and the ways in which they are at risk from novel security threats. We then reviewed research on intrusion detection systems in general and the issues and challenges in anomaly-based intrusion detection for HTTP Web services in particular. We presented discussions on research gaps, limitations and challenges in this area in terms of datasets, extracted features and the data preprocessing required as well as detection techniques. Our review has shown that in major cases, the datasets used are obsolete because attacks have increased in frequency, size, variety and complexity in recent years. In other cases, the target datasets are not publicly available for validation and comparison. In terms of preprocessing and feature selection, although methods for extracting distinctive features of packet headers are well acknowledged, approaches for packet contents (payloads) are less stringently defined. This may be due to a lack of content knowledge required to build payload-based features. Even though header features are still applicable to different scenarios, they are insufficient for Web services IDS.

Most intrusion detection experiments have reported high accuracies on public data for unsupervised methods (one-class SVM and MDM). However, it was notable that many results suffered from high rates of false positives and they were further undermined by the fact that the experiments were never replicated using different data sources for the sake of comparison. Although research has been conducted on detecting Web application attacks, the relevant studies do not comprehensively address SOAP- or XML-based attacks and a large number of anomaly-based detection models only detect attacks on the network layer and computer systems. Based on the problems and challenges identified with regard to each of the above issues, we hope to propose a new machine-learning technique specifically for anomaly-based Web service intrusion detection over HTTP traffic. The success achieved in other domains indicates that it is worthwhile studying the ways of increasing the efficiency, performance and accuracy of the intrusion detection model when dealing with SOAP/XML-based attacks.

6. Acknowledgment

This research was supported by the Fundamental Research Grant Scheme (FRGS) under the Ministry of Higher Education of Malaysia.

7. Author Contributions

All authors equally contributed to this work.

8. Ethics

This article is original and contains unpublished material. The corresponding author confirms that all the other authors have read and approved the manuscript.

9. References

Amer, M., M. Goldstein and S. Abdennadher, 2013. Enhancing one-class support vector machines for unsupervised anomaly detection. Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description, Aug. 11-14, Chicago, IL, USA, pp: 8-15. DOI: 10.1145/2500853.2500857

Anderson, J.P., 1980. Computer security threat monitoring and surveillance. Fort Washington.

Barot, V. and D. Toshniwal, 2012. A new data mining based hybrid network Intrusion Detection model. Proceedings of the International Conference on Data Science and Engineering, Jul. 18-20, IEEE Xplore Press, Cochin, Kerala, pp: 52-57. DOI: 10.1109/ICDSE.2012.6282310

Caulkins, B.D., J. Lee and M. Wang, 2005. A dynamic data mining technique for intrusion detection systems. Proceedings of the 43rd Annual Southeast Regional Conference, Mar. 18-20, Kennesaw, GA, USA, pp: 148-153. DOI: 10.1145/1167253.1167290

Chan, G.Y., C.S. Lee and S.H. Heng, 2013. Discovering fuzzy association rule patterns and increasing sensitivity analysis of XML-related attacks. J. Netw. Comput. Applic., 36: 829-842. DOI: 10.1016/j.jnca.2012.11.006

Chonka, A. and J. Abawajy, 2012. Detecting and mitigating HX-DoS attacks against cloud web services. Proceedings of the 15th International Conference on Network-Based Information Systems, Sept. 26-28, IEEE Xplore Press, Melbourne, VIC, pp: 429-434. DOI: 10.1109/NBiS.2012.146

Christey, A.S. and R.A. Martin, 2007. Vulnerability type distributions in CVE. U.S. Department of Homeland Security.

Corona, I. and G. Giacinto, 2010. Detection of server-side web attacks. Proceedings of the Workshop on Applications of Pattern Analysis, (APA' 10), pp: 160-166.

Dokas, P., L. Ertoz, V. Kumar, A. Lazarevic and J. Srivastava *et al*., 2005. Data mining for network intrusion detection.

Estévez-Tapiador, J.M., P. Garcı́a-Teodoro and J.E. Dı́az-Verdejo, 2004. Measuring normality in HTTP traffic for anomaly-based intrusion detection. Comput. Netw., 45: 175-193. DOI: 10.1016/j.comnet.2003.12.016

Ezeife, C.I., J. Dong and A.K. Aggarwal, 2008. SensorWebIDS: A web mining intrusion detection system. Int. J. Web Inform. Syst., 4: 97-120. DOI: 10.1108/17440080810865648

Fielding, R., U. Irvine and J. Gettys, 1999. Hypertext Transfer Protocol HTTP/1.0 Specifications: 24-Apr-1998. 1st Edn., toExcel, San Jose, ISBN-10: 1583482709, pp: 96.

Harshini, E., S. Selvakumar and T.N. State, 2011. SSENet-2011: A network intrusion detection system dataset and its comparison with KDD CUP 99 dataset. Proceedings of the 2nd Asian Himalayas International Conference on Internet, Nov. 4-6, IEEE Xplore Press, Kathmandu, pp: 1-5. DOI: 10.1109/AHICI.2011.6113948

Ingham, K.L., A. Somayaji, J. Burge and S. Forrest *et al*., 2007. Learning DFA representations of HTTP for protecting web applications. Comput. Netw., 51: 1239-1255. DOI: 10.1016/j.comnet.2006.09.016

Jamdagni, A., Z. Tan, P. Nanda, X. He and R.P. Liu, 2009. Intrusion detection using geometrical structure. Proceedings of the 4th International Conference on Frontier of Computer Science and Technology, Dec. 17-19, IEEE Xplore Press, Shanghai, pp: 327-333. DOI: 10.1109/FCST.2009.97

Jamdagni, A., Z. Tan, P. Nanda, X. He and R.P. Liu, 2010. Intrusion detection using GSAD model for HTTP traffic on web services. Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, Jun. 28-Jul. 02, Caen, France, pp: 1193-1197. DOI: 10.1145/1815396.1815669

Jensen, M., N. Gruschka and R. Herkenhoner, 2009. A survey of attacks on web services. Comput. Sci. Res. Develop., 24: 185-197. DOI: 10.1007/s00450-009-0092-6

Kapodistria, H., S. Mitropoulos and C. Douligeris, 2011. An advanced web attack detection and prevention tool. Inform. Manage. Comput. Security, 19: 280-299. DOI: 10.1108/09685221111188584

Karnwal, T., T. Sivakumar and G. Aghila, 2012. A comber approach to protect cloud computing against XML DDoS and HTTP DDoS attack. Proceedings of the IEEE Students' Conference on Electrical, Electronics and Computer Science, Mar. 1-2, IEEE Xplore Press, Bhopal, pp: 1-5. DOI: 10.1109/SCEECS.2012.6184829

Khalilian, M., N. Mustapha, M.N. Sulaiman and A. Mamat, 2011. Intrusion detection system with data mining approach: A review. Global J. Comput. Sci. Technol., 11: 29-34.

Kruegel, C. and G. Vigna, 2003. Anomaly detection of web-based attacks. Proceedings of the 10th ACM Conference on Computer and Communication Security, Oct. 27-30, Washington, DC, USA, pp: 251-261. DOI: 10.1145/948109.948144

Kruegel, C., G. Vigna and W. Robertson, 2005. A multi-model approach to the detection of web-based attacks. Comput. Netw., 48: 717-738. DOI: 10.1016/j.comnet.2005.01.009

Lee, W. and S.J. Stolfo, 2001. A framework for constructing features and models for intrusion detection systems. ACM Trans. Inform. Syst. Security, 3: 227-261. DOI: 10.1145/382912.382914

Lee, W., S.J. Stolfo and K.W. Mok, 2002. Algorithms for mining system audit data. Springer, 95: 166-189. DOI: 10.1007/978-3-7908-1791-1_8

Lippmann, R., J.W. Haines, D.J. Fried, J. Korba and K. Das *et al*., 2000. The 1999 DARPA off-line intrusion detection evaluation. Comput. Netw., 34: 579-595. DOI: 10.1016/S1389-1286(00)00139-0

Mahoney, M.V., 2003. Network traffic anomaly detection based on packet bytes. Proceedings of the ACM Symposium on Applied Computing, Mar. 09-12, Melbourne, FL, USA, pp: 346-350. DOI: 10.1145/952532.952601

Mahoney, M.V. and P.K. Chan, 2002. Learning nonstationary models of normal network traffic for detecting novel attacks. Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Jul. 23-25, Edmonton, AB, Canada, pp: 376-385. DOI: 10.1145/775047.775102

Malek, M. and F. Harmantzis, 2004. Data mining techniques for security of web services. Proceedings of the 1st International Conference on E-Business and Telecommunication Networks, Aug. 24-28, Setbal, Portugal, pp: 1-14.

Moradian, E. and C. Science, 2006. Possible attacks on XML web services. Int. J. Comput. Sci. Netw. Security, 6: 154-170.

Nadiammai, G.V. and M. Hemalatha, 2012. An evaluation of clustering technique over intrusion detection system. Proceedings of the International Conference on Advances in Computing, Communications and Informatics, Aug. 03-05, CHENNAI, India, pp: 1054-1060. DOI: 10.1145/2345396.2345565

Najjar, M.S.A. and M. Abdollahi Azgomi, 2010. A distributed multi-approach intrusion detection system for web services. Proceedings of the 3rd International Conference on Security of Information and Networks, Sept. 07-11, Taganrog, Rostov-on-Don, Russian Federation, pp: 238-244. DOI: 10.1145/1854099.1854147

Park, Y. and J. Park, 2008. Web application intrusion detection system for input validation attack. Proceedings of the 3rd International Conference on Convergence and Hybrid Information Technology, Nov. 11-13, IEEE Xplore Press, Busan, pp: 498-504. DOI: 10.1109/ICCIT.2008.338

Perdisci, R., D. Ariu, P. Fogla, G. Giacinto and W. Lee, 2009. McPAD: A multiple classifier system for accurate payload-based anomaly detection. Comput. Netw., 5: 864-881. DOI: 10.1016/j.comnet.2008.11.011

Robertson, W., G. Vigna, C. Kruegel and R.A. Kemmerer, 2006. Using generalization and characterization techniques in the anomaly-based detection of web attacks. Proceedings of the Network and Distributed System Security Symposium, (DSS' 06), pp: 1-15.

Saboori, E., S. Parsazad and Y. Sanatkhani, 2010. Automatic firewall rules generator for anomaly detection systems with Apriori algorithm. Proceedings of the 3rd International Conference on Advanced Computer Theory and Engineering, Aug. 20-22, IEEE Xplore Press, Chengdu, pp: V6-57-V6-60. DOI: 10.1109/ICACTE.2010.5579365

Seo, J., H.S. Kim, S. Cho and S. Cha, 2004. Web server attack categorization based on root causes and their locations. Proceedings of the International Conference on Information Technology: Coding and Computing, Apr. 5-7, IEEE Xplore Press, pp: 90-96. DOI: 10.1109/ITCC.2004.1286431

Tombini, E. and M. Ducass, 2004. A serial combination of anomaly and misuse IDSes applied to HTTP traffic. Proceedings of the 20th Annual Computer Security Applications Conference, Dec. 6-10, IEEE Xplore Press, pp: 428-437. DOI: 10.1109/CSAC.2004.4

Vorobiev, A., 2006. Security attack ontology for web services. Proceedings of the 2nd International Conference on Semantics Knowledge and Grid, Nov. 1-3, IEEE Xplore Press, Guilin, pp: 42-42. DOI: 10.1109/SKG.2006.85

Wang, K. and S.J. Stolfo, 2004. Anomalous payload-based network intrusion detection. Proceedings of the 7th International Symposium RAID, Sept. 15-17, Sophia Antipolis, France, pp: 203-222. DOI: 10.1007/978-3-540-30143-1_11

Wolter, R., 2013. XML Web Services Basics. Microsoft Corporation-Library.

Yee, C.G., W.H. Shin and G.S.V.R.K. Rao, 2007. An adaptive Intrusion Detection and Prevention (ID/IP) framework for web services. Proceedings of the International Conference on Convergence Information Technology, Nov. 21-23, IEEE Xplore Press, Gyeongju, pp: 528-534. DOI: 10.1109/ICCIT.2007.422

Appendix A

Injection: Injection flaws, such as SQL, OS and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

Broken Authentication and Session Management: Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities.

Cross-site Scripting (XSS): XSS flaws occur whenever an application sends untrusted data to a Web browser without proper validation or escaping. XSS allows attackers to execute scripts in the victim's browser and these can hijack user sessions, deface websites, or redirect the user to malicious sites.

Insecure Direct Object References: A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.

Security Misconfiguration: Good security requires a secure configuration that is defined and deployed for the application, frameworks, application server, Web server, database server and platform. Secure settings should be defined, implemented and maintained, as defaults are often insecure. Moreover, software should be kept up to date.

Sensitive Data Exposure: Many Web applications do not properly protect sensitive data, such as credit cards, tax IDs and authentication credentials. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserve extra protection, such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.

Missing Function-level Access Control: Most Web applications verify function-level access rights prior to making the relevant functionality visible in the user interface. However, applications need to perform the same access control checks on the server when each function is accessed. If the requests are not verified, attackers can forge requests in order to access the functionality without proper authorization.

Cross-site Request Forgery (CSRF): A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable Web application. This allows the attacker to force the victim's browser to generate requests that the vulnerable application thinks are legitimate requests from the victim.

Using Components with Known Vulnerabilities: Components, such as libraries, frameworks and other software modules, almost always run with full privileges. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications using components with known vulnerabilities may undermine application defenses and enable a range of possible attacks and impacts.

Unvalidated Redirects and Forwards: Web applications frequently redirect and forward users to other pages and websites and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages.

Appendix B

Identity Attacks: Identity attacks consist of authentication and authorization attacks, such as dictionary attacks, Internet Protocol (IP) spoofing, data tampering and message eavesdropping attacks. These attacks are old, i.e., Web application attacks that may change their target on XML Web services.

Session Attacks: Session attacks change the target on the Web services. Attackers can use session attacks to capture messages or insert false instructions.

Replay Attacks: Replay attacks occur when an attacker sends repetitive SOAP messages to overload a Web service.

Man-in-the-middle Attack: This is an intermediate-station open possibility for attackers to carry out man-in-the-middle attacks by inserting fake/bluff routing instructions, so that the message travels to a malicious location from there the attacker can send malicious instructions to the original destination.

Parsing Attacks: Parsing attacks, including recursive payloads, oversized payloads and schema poisoning, are aimed at the XML parser.

Recursive Payloads: Recursive payloads attacks occur when an attacker exploits the capability of XML by creating a document thousands of elements deep to stress and break the parser.

Oversized Payloads (Yee *et al*., 2007): While a genuine XML document can be hundreds of megabytes or gigabytes in size, e.g., with a digital video file attached, it can also have be created by an attacker attempting to exhaust the server's memory and break the XML parser, hence causing an oversized payload attack that can lead to a DoS attack as well.

Schema Poisoning (Moradian and Science, 2006): XML schemas that provide formatting instructions for parsers when interpreting XML documents are used for all major XML standard grammars. Because these schemas describe necessary preprocessing instructions, they are susceptible to poisoning. Attackers compromise an XML schema and replaces it with a similar but modified one. When an XML schema is compromised, attackers can manipulate data processed by the application. Successful schema poisoning may lead to XML Denial-of-Service attacks.

Overflow Attacks (Seo *et al*., 2004): Overflow attacks are aimed at the service endpoint and the SOAP engine through the Web server. Attackers send parameters longer than the program can handle, which can cause the service to crash. One example of a strange request is a username with more characters than expected.

Code Attacks (Park and Park, 2008) Malicious code carried by XML messages passes through port 80. Code attacks are intended to affect applications that run Web services. Successful code attacks provide attackers with opportunities, such as extracting the entire XML database.

SQL Injection (Seo *et al*., 2004) SQL injection attacks occur when malicious SQL statements are inserted into XML in order to disrupt the back-end system, in order to force a SOAP endpoint, i.e., a server, to do something it was not intended to. For example, retrieved data may become inaccessible; data can even be destroyed through SQL injection and content can be manipulated within a SOAP message. This results in receiving end-point and consumes excessive resources, e.g., buffer overflow and the system crashes or becomes unresponsive.

XPath injection (Moradian and Science, 2006) An XML document has no access control or privilege system associated with it. By performing XPath injection, it is possible for attackers to extract the entire XML database.

XDoS Attack (Moradian and Science 2006) Denial-of-Service attacks occur when an attacker attempts to prevent legitimate users from accessing a service. In traditional DoS attacks, the attacker's goal is to disable a user computer or network. XDoS changes the target and tries to disable important services such that legitimate users cannot use them. XDoS attacks consume resources by forcing messages to do useless work.

WSDL Attacks (Moradian and Science, 2006) Web Services Definition Language (WSDL) attacks include WSDL scanning and parameter tampering. Through WSDL attacks, the attackers can analyze and misuse WSDL information and tamper with parameters within WSDL documents.

WSDL Scanning (Park and Park, 2008) WSDL describes the logical and concrete details of a Web service. A WSDL document contains information that describes how to use parameters and information, types of input/output (I/O) and parameters of methods. Lindstrom [30] has pointed out that by scanning a WSDL document, an attacker can obtain sensitive information, such as types, messages, operations, port types, bindings and can guess other methods.

Parameter Tampering (Seo *et al*., 2004), (Yee *et al*., 2007) UDDI is a registry that lists Web services across multiple servers, hence providing a contact point for attackers to find all information required to launch attacks on a WS. A WSDL document consists of parameters defined for service operations and it is easy for an attacker to search through the document to obtain unintentionally published information that is useful for launching attacks.

XML Attacks (Vorobiev, 2006) The class of XML attacks has three subclasses: Parsing attacks, XML injection and XPath injection attacks.

SOAP Attacks (Yee *et al*., 2007) SOAP header attacks can lead to WS-DoS when an attacker creates a complex SOAP header with a source routing message to bypass the security check.