CHAPTER I

INTRODUCTION

Today more companies make use of system administrators to maintain their network, regulate user privileges and keep everything up and running. With the growing size of these companies, the amount of work put into these activities also grows enormously. Hence, a single administrator can no longer overlook this meshwork of permissions, exceptions and policies. Apart from the resulting overhead to add, delete or adjust permissions, this may also lead to some severe security issues like malicious user’s access or attack etc. This may also result unauthorized utilization of network resources. Hence, it could be better if a computer system decides by itself to grant access permissions to a user.

Most of the web applications nowadays use web services in their process. To reduce the overhead of administrators who maintain this web application and give privileges to the web users for accessing web services, an access control model can be developed. Since, web services are platform independent, loosely coupled and reusable, an access control model with access control policies should be developed effectively using web services technologies.

This research work develops a trust based web services access control model which allows web service requesters to access the resource based on their trust value. This work considers different factors to evaluate the trust value of web service requester and decisions to allow accessing resources can be made according to trust value. Before evaluating trust, this access control model checks SQL Injection attack and IP spoofing. This work motivates the need of trust in web services access model.
1.1 MOTIVATION

Since majority of business organizations move around web services, a lot of security issues arise in web services environment. Hence, the following security related issues needed to be sorted out to secure the web services.

**Open in Nature:** Web applications that are created with web services are sometimes open in nature. For example, a web-based online store is open to anyone with an internet connection. This kind of applications may deal with unidentified or unfamiliar users. Hence, preventing unauthorized users from accessing resources would be important.

**Malicious Attackers:** Malicious attackers may attack the web service by sending SQL Injection and IP address spoofing. Sometimes, it may lead to unauthorized access of web services. Therefore, developing an efficient method to detect and prevent these attacks is required.

**Expose to Number of users:** Web applications created with web services technology are potentially exposed to large number of users across the globe through web. Web is the medium where the number of users increases enormously. Hence the access control model of service providers must be as scalable as the applications themselves.

**Time Delay:** The access control of web services may take more time to authorize the web service requester. Sometimes, this time delay may allow inefficient accesses to the web services, so, adding any new mechanism may reduce time delay of model.

**Machines as Requesters:** When virtual environments are created, there is a growing tendency in organization to give machines the responsibility to act on behalf of humans. So, there is a need for access control to determine the trustworthiness of requesting machines before access is granted to sensitive operations or resources. To determine the trust of web users, a valuable access control policies may be derived.
1.2 PROBLEM STATEMENT

This research work identifies that access control model based on trust is a fundamental requirement for web applications created for web services. This access control model uniquely addresses specific requirements of web services environments. This research work addresses the following issues.

- **Strength and weakness of existing access control models**

  Various existing access control models have their own strengths and weaknesses. Balancing the competing goals of collaboration between web services entities and security is the difficult task. An investigation thus needs to be conducted into the variety of current access models, to determine the extent to which their mechanism could meet the identified access control requirements of web services.

- **Policies to enforce access control decisions**

  Web services interoperation requires the specification of various types of policies. Quality-of-service needs such as security, privacy, performance and availability that are either required or supported by web services entities can be specified. Policies need to be supported by web services access control, so that web services entities can be given the ability to interoperate in a platform-independent manner.

- **Access Control Requirements**

  A unique characteristic of web services is that they allow business partners to communicate through human legible Simple Object Access Protocol (SOAP) messages. A partner’s secrets may be exposed in a SOAP response and must be protected from threats such as disclosure to malicious parties. Due to the fact that XML is text-based, web services invocations can pass over HTTP. The applications that are used to process the request contained by these XML messages may then be
endangered by false claims or malicious information. Access control is thus a critical requirement for web services to be reliable in operation. As currently there is no standard, agreed-upon method for exposing web services operations over the internet in such a way that only authorized users can call them, access control requirements for web services need to be identified.

- **Deployment of model in web services architecture**

  It is important to consider the architecture of enforcement for web services access control. The access control model to be defined by this research needs to be incorporated into the web services architecture. It should support the different types of policies that are identified, so that access control is uniformly addressed by multiple enforcement points.

- **Trust incorporation into an access control model**

  In traditional access control model, access control decisions are determined according to the identities of subjects. In open environments such as web services, web services requestors cannot be identified only by identities in order to access the resources. Access control model based on attributes of the web services requester’s flexibility and scalability are essential to the success of web services environments. But the notion of attribute based access control is not sufficient, as it does not make it possible to distinguish between web services requesters who are trustworthy and those who are not. Thus, access control model based on the trustworthiness of web service requester is needed to grant more and advanced access to the subjects of web services requesters who are more trusted. For every decision, the web services access control model needs to address the trust of services requestor. To evaluate the trust of a user, it should consider different factors, so that better access model can be developed for controlling the access of web services.
• **Increase of Web Services Requesters**

As web service requester population is increasing in the web services environments, it needs better manual centralized control authority. But lack of efficient manual centralized control authority shifts the responsibility for access control and other decisions to participating machines that support web services.

1.3 **RESEARCH OBJECTIVES**

• The ultimate objective is to develop an efficient web services access control model which can detect and prevent the malicious requesters or behavior.

• Developing an effective dynamic access control model to adjust access levels automatically and efficiently for the web services. Hence, effective access controls policy to be developed and implemented.

• To meet the requirements, this access control system is utilized the term called as Trust. This will delight the problem of establishing trust relationships between requester and provider and evaluate the capabilities of trust-based access control.

• To maintain effective trust level of requester, different factors such as server error, success rate, failure rate, time out etc., are considered and trust value was evaluated. Depending upon the trust value, requester will be allowed to access the web service.

• To decrease the work load and time to authorization processes, the system initially authenticates the user by checking SQL injection and IP address spoofing. Once there is no SQL injection and IP address spoofing, the system allows requester to access the web services placed at services provider location.

• The access control system needs to be managing the time factor of the web service requesters so that it can serve more users. To reduce the response times of the model some additional features to be added in the access control model.
1.4 WEB SERVICES ARCHITECTURE

The Web Services is a method which is used to communicate between two electronic devices over the web [1]. Web Services are a set of methods and functions that are described by a Web Services Description Language (WSDL) and published using Universal Description Discovery and Integration (UDDI). Web Services are becoming a popular technology which brings great economic benefits to people in the development of complex web applications. Web services architecture is depicted in Figure 1.1.

![Web Services Architecture Diagram](image)

**Figure 1.1: Web Services architecture**

Web services describes a standardized way of integrating Web-based applications using the XML, Simple Object Access Protocol (SOAP), WSDL and UDDI open standards over an Internet protocol backbone. XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing the services that are available. Web service requester and web service provider play an important role in the architecture. Service provider is the person who provides services to different users through web. Service requester is the person who accesses service from the provider through web. UDDI registry facilitates to discover the service in the directory. It has components which effectively describes services of different providers.
Unlike traditional client/server models, such as a web server/web page system, web services do not provide the user with a GUI. Web services instead share business logic, data and processes through a programmatic interface across a network. Web services allow different applications from different sources to communicate with each other without time-consuming custom coding, and because all communication is in XML, web services are not tied to any one operating system or programming language. Web application uses web services as a service, so that they are sometimes called application services. There are many tools which are used to test the web services without using web applications.

HTTP protocol is the current standard for transferring HTML documents although it is designed to be extensible to almost any document format like XML for example. The simplest HTTP message is "GET url", to which the server replies by sending the named document. If the document does not exist, the server may send an HTML-encoded message stating this. This form of communication represents a typical request/response mechanism.

A client sends a request for a specific document to the server and waits for a response. If the server does not respond with the requested document, it is up to the client to wait for the timeout and request the same document again. This loosely coupled type of communication is very common in client-server architectures. Loosely coupled system provides the feature of platform independence.

In addition to GET requests, clients can also send HEAD and POST requests, of which POSTs are the most important. POSTs are used for HTML forms and other operations that require the client to transmit a block of data to the server. After sending the header and the blank line, the client transmits the data. In this way, it utilizes the HTTP protocol to transmit both data payload and request to a web service.
1.5 WEB SERVICES TECHNOLOGY

- **XML (Extensible Markup Language)**

XML is an abbreviation for eXtensible Markup Language which is designed to describe data and improve the functionality of the web by providing more flexible and adaptable ways of information representation [1]. XML is a Meta language which lets us design our own customized markup languages but unlike HTML, XML does not specify semantics or a set of tags. Since there is no prescribed method for rendering XML documents, semantics will be defined by the application using it or by style sheets. The following example will show the structure of an XML document and how data is represented:

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<note noteID="1">
  <to>Bob</to>
  <from>Al</from>
  <heading>Our dilemma</heading>
  <body>Don't dare to squeal on me man!</body>
</note>
```

This basic XML document starts with the XML declaration in the first line. It defines the XML version and the character encoding used. It is important to specify the character set to avoid misinterpretation of the provided data. The next line describes the root element of the document. Elements are one way to store data in an XML document. The above lines in XML describe the child elements of root (to, from, heading and body). By looking at the elements, it is easy to see that the XML document represents a message. The last line describes the end of the root element,
completing the note from Al to Bob. Along with the root element in the second line comes an attribute called noteID. Attributes are another way to store data and are used to provide additional information about elements, also called meta-data. In this case, it may be used to count the messages sent from Al to Bob.

A list of legal elements that defines the document structure is the Document Type Definition (DTD). A document with correct XML syntax is called "Well Formed" while a "Valid" XML document also conforms to a DTD. More and more applications make use of XML to store information because of its benefits. Some of them are:

- The structure is well-defined and can be passed between different computer systems which would otherwise be unable to communicate.
- Data payload is encapsulated in tags and is readable by human viewers.
- Due to their textual nature, XML-Files are platform-independent.

These advantages made XML the perfect format to communicate between web services. To ensure a platform and language independent use for every web service, SOAP was developed. It is an XML application with defined elements and a predefined structure.

- **SOAP (Simple Object Access Protocol)**

  SOAP is an XML-based protocol [2] for sending messages and making remote procedure calls in a distributed environment. Using SOAP, data can be serialized without regard to any transport protocol although HTTP is typically the protocol of choice. SOAP is an extensible, text-based framework for enabling communication between diverse parties that have no prior knowledge of each other. SOAP specifies a mechanism to perform remote procedure calls therefore it removes the requirement that two systems must run on the same platform or be written in the same language.
The envelope consists of an optional header which may target the nodes that perform intermediate processing, and a mandatory body which is intended for the final recipient of the message. The Header may also hold digital signatures for a request contained in the body.

The body contains the serialized payload. For a request this is the method argument where the surrounding XML tag must have the same name as the called method. The response body contains the return value if it exists. Data types are not delineated in the SOAP envelope explicitly so the type of a result parameter cannot be discovered just by looking at the SOAP message. Client applications define data types either generically through Section 5 encodings, or privately via agreed-upon server contracts.

The anatomy of a SOAP Envelope looks like this:

```xml
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <t:Transaction-ID xmlns:t="some-URI">552511951722</t:Transaction-ID>
  </soap:Header>
  <soap:Body>
    <m:RemoteFunction xmlns:m="some-URI">
      <Parameter1>123</Parameter1>
    </m:RemoteFunction>
  </soap:Body>
</soap:Envelope>
```
In this example, the header contains some additional information enclosed by the Transaction-ID tag. This ID can be processed by any node before the final service node to ensure the request's correctness. The body contains one method which is termed as \textit{RemoteFunction} whereas the methods parameter \textit{Parameter1} is 123. The parameters type may be of integer type but could be a String as well. The client application must decide how to handle it. SOAP messages are fundamentally one-way transmissions from a sender to a receiver, but they are often combined to implement a request/response mechanism.

- **Publishing and Finding Web Services – WSDL**

  With SOAP, a communication between Web services is possible, structured and each participant knows how to send or receive the corresponding SOAP Message. The final step to complete the communication architecture of Web services is to define how to access a service once it is implemented. This is where the Web Service Description Language (WSDL) steps in [3]. WSDL describes services as collections of network endpoints, or ports. Again, it is an XML document with a defined grammar where the abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings. WSDL documents use the following elements to describe a Web service:

  - Types: A container for data type definitions.
  - Message: A definition of the data being passed in a single RPC.
  - Operation: A description of an action (method) supported by the service.
  - Port Type: A set of operations supported by one or more endpoints.
  - Binding: A concrete data format specification for a particular port type.
  - Port: A single endpoint defined as a combination of a binding and the network address. Port plays vital role in network programming.
• Service: A collection of related endpoints.

Now that a Web service can be described completely, the last problem is how a potential user can find the corresponding description (WSDL document).

• **UDDI (Universal Description Discovery and Integration)**

  UDDI provides a method for publishing and finding service descriptions [3]. It is a directory where web services can be registered and assigned to a service provider, therefore forming a structure that resembles a yellow pages directory. Any user can browse the directory to search for a desired service and download the description in case of a match. UDDI is an industry initiative (ARIBA, IBM and Microsoft) to enable businesses to quickly and dynamically find and transact with each other. Searching a UDDI registry can be done by any UDDI browser or by the registry's Web interface if present. Microsoft for example, provides both a Web interface and the original UDDI registry to search for an entry. A query with 'weather' as the search string returns about 10 results at the Microsoft UDDI registry. The corresponding WSDL File can be found in the overview document if it was specified.

  The UDDI registry is implemented as multiple peer nodes where the registration across all nodes is replicated by node operators. Again, the data structure of the registry is XML yet a WSDL document cannot be published without additional precautions. The structure of a UDDI registry differs from a WSDL file because UDDI also supports business and service information. As a result, it has no direct support for WSDL or any other service description mechanism. UDDI registry contains the services general description, port bindings and a reference to the original WSDL File as tModels.

  Every service interface of a WSDL file is published as a tModel in the UDDI registry. A service interface includes types, message, portType and binding. Some
elements of the business service are constructed using information from the WSDL service description. The service name for example, is used as the name of the UDDI's business service entry. Publishing the service description to a UDDI directory was the last step in the process to create a Web service.

1.6 TERMINOLOGY USED

In order to avoid any misclassification, it is important to interpret the terminology used in the thesis correctly. Hence, this section provides brief introduction about the terms such as web services access control model, trust based access control, web server log, SQL Injection Attacks (SQLIAs), XML validation and IP address spoofing used in this research work.

1.6.1 WEB SERVICES ACCESS CONTROL MODEL

Access control model for a web service is to restrict the set of clients or subjects that can invoke the operations offered by the service. Because the clients are usually not known a priori, credentials are adopted to enforce access control. Credentials are assertions describing the properties that are used to implement trust between the service provider and service requester. Access control policies define rules stating that only subjects with certain credentials satisfying specific conditions can interact with a web service.

Access control is not a new paradigm, which has been widely studied in the literature and especially in database systems; only recently work on security for web service has emerged as an important part of the Web service. Most access control approaches assume a single operation model where operations are independent from
each other. Access control is either enforced at the level of the entire web services or at the level of single operations.

Initially, the web service may ask the client to provide all the credentials associated with all operations of the Web service in advance. This approach guarantees that a client will always arrive at the end of the conversation. However, this approach has an issue that the client will become aware of all policies based on which access control is enforced.

Another issue is that the client may have to submit more credentials than needed. An alternative strategy is to require only the credentials associated with the next operation that the client wants to perform. This strategy has the advantage of asking the client only for the necessary credentials to access the requested operation. However, the client is continuously solicited to provide credentials for each transition. Access control for Web services is already becoming the hot topic in the field of Web services security. Therefore, an effective access control model is needed to avoid this kind of malicious user behaviors. Some of the traditional access control models are discussed below.

Role-Based Access Control (RBAC) [1] is the one of the most important and widely used Web Service access control scheme. In such access control schemes, clients are assigned roles that contain permissions in order to gain a secure access to specific Web Services.

Attribute-Based Access Control (ABAC) models add more dynamicity to the traditional RBAC systems [2]. These models make use of attributes owned by the clients, the providers, and some other attributes related to the environment. Decisions are made to allow or deny the request based on all these attributes and its values. Example: Geographical Area and Status of customer/provider.
Governance Based Access Control (GBAC) [3], the basic idea of the model is that transactions must be controlled by the relevant legislation to which the organizations sharing the information are accountable. Hence any request for information is verified against the existing laws or regulations before it is granted the permission.

In Session Based Access Control (SBAC) [3], the context of a transaction is limited to a session. Accesses to resources are restricted based on the attributes of the subjects and the properties of the objects but the rights that can be applied at a given time are limited based on the context defined by the access session.

Location-Based Access Control (LBAC) permits the requester to access the resource based on requester's physical location.

Semantic Access Control (SAC) [33] is a new kind of access control model, which uses machine reasoning at a semantic level to determine whether to let the requests pass according to the semantic descriptions of the policies, requests, resources and other entities. This access control model is more scalable, more applicable to dynamic environments with heterogeneous and complex access criteria. Since the foundation of Semantic Access Control is the semantic web technologies, it cannot be applied in all access control fields.

1.6.2 TRUST BASED ACCESS CONTROL

Trust is a complex theory and can be defined in different ways at different levels. Trust is a complicated issue that may be associated to other attributes such as security, honesty, reliability, accuracy, risk, utility, benefit, competence, belief, perception, expertise etc.

Trust is defined in [3] as: “Trust is the dense belief in the ability of an entity to act as expected such that this dense belief is not a fixed value associated with the
entity but rather it is subject to the entity’s behavior and applies only within a specific context at a given time”. From this definition many important properties of trust have been captured. They are (i) Trust is not a static value, it is dynamic, (ii) It is condition and time reliant, and (iii) It depends on entity’s past and present behavior. The trust research can be organized as four major areas [8]. They are as follows

1) **Policy-based trust:** policies are used to establish trust, managing credentials and controls accessing entity. This type of trust generally considers that trust is established by obtaining a sufficient amount of credentials pertaining to a specific party, and use those policies to grant the party certain access rights.

2) **Reputation-based trust:** In this type, reputation plays an important role. Reputation is used to establish trust, where past performances of an entity are combined to determine its future behavior. Research in reputation-based trust uses the history of an entity’s behavior to compute trust, and may use referral based trust in the absence of first-hand knowledge.

3) **General models of trust:** Trust models are useful for analyzing human and agenized trust decisions and to operate computable models of trust. Work in modeling trust describes values that play a role in computing trust, and leans more on work in psychology and sociology for a decomposition of what trust comprises. Modeling research ranges from simple access control polices to analyses of competence, beliefs, risk, importance, utility, etc.

4) **Trust in information resources:** Trust is a common factor in web research because it determines web resources and web sites are trustworthy. Moreover, trust on the Web has different uses and meanings, including getting ratings from users about the quality of information and services they have used, how web site design influences trust on content and content providers, propagating trust over links, etc.
In the web service interactions, the participants can establish trust based on 1) Peer-To-Peer trust model 2) Trust Chain Model (TC) and 3) Trusted Third Party Model (TTP). In P2P Model, participants should share their trust relations on a one-by-one basis which is not scalable. In TC Model, each participant stabilizes their trust relation with its direct ancestor, which is not flexible for dynamic service integrations. In the TTP model, all participants get the credentials from a trusted third party [5].

Trust based access control model is another emerging access control model. In recent years, many researchers are working on the trust based access control model of web services. In many applications such as E-Commerce, Internet Shopping, the interaction between subject and object is established on the basis of trust. Many examples show that the interaction is more reliable based on trust. Hence, it is very essential to adopt the concept of trust in access control.

Trust based access control systems [4] are diverse from the previous access control schemes since clients trust levels are dynamically calculated based on some arithmetical analysis of behaviors, activities and previous access attempts. Thus, service violations and bad client behavior lead to a decrease of the trust level, whereas good behavior leads to an increase in the trust level. Figure 1.2 shows a common outline of the WS-Trust architecture.

![Figure 1.2: General WS-Trust Architecture](image)
Requestor is an entity that attempts to invoke a secure operation above a network connection. In practice, a requester is normally a web service client.

Relying Person refers to an entity that has various services or resources that have to be secured next to unauthorized access. In carry out, a relying Person is classically a web service.

Security Token is a set of security data that a requester sends within a request so as to invoke a secure operation or to get access to a secure resource. In the WS-Trust framework, the idea of a security token is somewhat common and can be used to explain any chunk of security data that might escort a request.

SAML token is a mainly flexible type of security token. The SAML specification describes a common reason XML schema that allows you to enfold approximately any type of security data and enables you to sign and encrypt element or the entire token. SAML is a well suited option of token to employ in the framework of WS-Trust, because SAML has all of the essential features to hold up characteristic WS-Trust authentication scenarios. SAML means that Security Assertion Markup Language.

Claim is a SAML security token is officially defined to consist of a set of claims. Every claim characteristically holds an exacting type of security data.

Policy can symbolize the security configuration of a participant in a secure application. The requester, the relying person, and the security token service are all configured by policies. For instance, a policy can be used to arrange what types of authentication are sustained and necessary.

Security Token Service The security token service (STS) put down at the heart of the WS-Trust security architecture. Service requester and service provider are also other important terms in the trust architecture.
In the WS-Trust typical, the subsequent bindings are definite:

- **Issue binding**—the specification describes this binding as follows: Based on the credential offered/established in the request, a novel token is issued, perhaps with new proof information.

- **Validate binding**—the specification describes this binding as follows: The scope of the specified security token is estimated and an effect is returned. The effect may be a status, a novel token, or both.

- **Renew binding**—the specification describes this binding as follows: formerly issued token with running out is obtainable and a similar token is returned with novel running out semantics.

- **Cancel binding**—the specification describes this binding as follows: When a formerly issued token is no longer wanted, the Cancel binding can be utilized to terminate the token and its use.

In many applications (such as E-Commerce, Internet Shopping), the interaction between subject and object is established on the basis of trust. We usually say we believe someone or someone is credible. That means it is more possible to do good for us about, his/her behavior so as to we can consider to cooperate with him/her. Correspondingly, we say that someone is unreliable, which means the possibility is quite low. Entity makes the decision for above two possibilities according to the subjective initiative. Many examples show that the interaction is more reliable based on trust. Hence, it is very essential to adopt the concept of trust in access control.

### 1.6.3 WEB SERVER LOG

A Web server log, located in web server is the file which stores activity details such as IP address, username, password, date and time of request, URL of web user
when a request is submitted to a web server. These data can be combined into a single file, or separated into distinct logs, such as an access log, error log, or referrer log. However, server logs typically do not collect user-specific information. A statistical analysis of the server log is used to examine traffic patterns by time of day, day of week, referrer, or user agent. The main source of raw data is the web access log which shall be referred as log file. This information is recorded in chronological order.

**Example Fragment of Server Log**

The following is a fragment from the server logs for loganalyzer.net.

```
196.109.55.102 xyz abc [08/Oct/2013:04:54:20 -0400] "GET /about.html HTTP/1.1"
20011179"Mozilla/5.0(compatible; Googlebot/2.1; +http://www.google.com)".
```

This log file structure reflects the information of web user as follows:

- Remote IP address or domain name: An IP address is a 32-bit host address defined by the Internet Protocol; a domain name is used to determine a unique Internet address for any host on the internet. One IP address is usually defined for one domain name.
- Username and password if the server requires user authentication.
- Entry and exit date and time.
- Modes of request: GET, POST or HEAD method of Common Gateway Interface.
- Status codes: It is a part of log files which specify error conditions as well as successful communication of data. The HTTP status code returned to the client. Few of common status codes are listed in Table 1.1
- Bytes: The content-length of the document transferred.
- Remote log and agent log, Remote URL and Requested URL
- “request:” The request line exactly as it came from the client.
Log files are plain text (ASCII) files that are independent from the server platform. There are some distinctions between server software, but traditionally there are four types of server logs: Transfer Log, Agent Log, Error Log and Referrer Log. The first two types of log files are standard. The Referrer and Agent Logs may or may not be “turned on” at the server or may be added to the Transfer log file to create an “Extended” Log File format.

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Description</th>
<th>Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>408</td>
<td>Request Time-out</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
<td>409</td>
<td>Conflict</td>
</tr>
<tr>
<td>202</td>
<td>Accepted</td>
<td>413</td>
<td>Request Entity Too Large</td>
</tr>
<tr>
<td>203</td>
<td>Non-Authoritative Information</td>
<td>414</td>
<td>Request-URL Too Large</td>
</tr>
<tr>
<td>305</td>
<td>Use Proxy</td>
<td>500</td>
<td>Server Error</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>502</td>
<td>Bad Gateway</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized</td>
<td>503</td>
<td>Out of Resources</td>
</tr>
<tr>
<td>404</td>
<td>Not found</td>
<td>504</td>
<td>Gateway Time-Out</td>
</tr>
</tbody>
</table>

However, server logs typically do not collect user-specific information. A statistical analysis of the server log may be used to examine traffic patterns by time of day, day of week, referrer, or user agent. The main source of raw data is the web access log which shall be referred as log file. These files are usually not accessible to general Internet users, only to the webmaster or other administrative person. A statistical analysis of the server log may be used to examine traffic patterns by time of day, day of week, referrer, or user agent. Efficient website administration, adequate hosting resources and the fine tuning of sales efforts can be aided by analysis of the
web server logs. Marketing departments of any organization that owns a website should be trained to understand these powerful tools.

1.6.4 SQL INJECTION ATTACKS (SQLIAs)

SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. Any process that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives. It has become a common issue with database driven web sites. This flaw is easily detected, and easily exploited, and as such, any web related application with even a minimal user base is likely to be subject to an attempted attack of this kind. This flaw depends on the fact that SQL makes no real distinction between the control and data planes. The fear of SQL injection attacks has become increasingly frequent and serious.

SQL Injection Example

Consider the following query in the ASP page is of the form;

```
SELECT * FROM EMPLOYEE WHERE NAME='$login' AND PASS='$pass'
```

If the login and password as provided by the user are used, the query to be submitted to the database takes the form;

```
SELECT * FROM EMPLOYEE WHERE NAME='abc' AND PASS='xyz'
```

A web site that uses this asp would be vulnerable to SQLIAs. If the user were to enter [' OR 1=1 --'] and ['] instead of [abc] and [xyz], the query would take the form;

```
SELECT PROFILE FROM EMPLOYEE WHERE NAME=' ' OR 1=1 -- ' AND PASS=''.
```

The characters “--” mark the beginning of a comment in SQL, and everything after that is ignored. The code injected in the conditional (OR 1=1) transforms the entire WHERE clause into a tautology the query evaluates to true for
each row in the table and returns all of them. Thus an attacker can bypass all authentication modules in place and gain unrestricted access.

**Types of SQL Injection**

There are different types of SQL Injection are available. Very few are listed below:

**Tautologies**

SQL injection queries are injected into one or more conditional statements so that they are always evaluated to be true.

**Logically Incorrect Queries**

This type of injection uses error messages to find useful data facilitating injection of the backend database.

**Union Query**

Injected query is joined with a safe query using the keyword UNION in order to get information related to other tables from the application.

**Stored Procedure**

Many databases have built-in stored procedures. The attacker executes these built-in functions using malicious SQL Injection codes.

**Piggy-Backed Queries**

Additional malicious queries are inserted into an original injected query.

**Alternate Encodings**

It aims to avoid being identified by secure defensive coding and automated prevention mechanisms. Hence, it helps the attackers to evade detection. It is usually combined with other attack techniques.

**External Entity Attack**

Another benefit of XML is its ability to build documents dynamically at the time of insertion by pointing to a URI where the actual data exists. These external entities
may not be trustworthy. An attacker can then replace the data being collected with malicious data.

**WSDL Scanning**

It is a mechanism for web services to dynamically describe the parameters used when connecting with specific methods. These files are often built automatically using utilities. These utilities, however, are designed to expose and describe all of the information available in a method. In addition, the information provided in a WSDL file may allow an attacker to guess at other methods. For example, a service that offers stock quoting and trading services may advertise query methods like requestStockQuote, however also includes an unpublished transactional method such as tradeStockQuote. It is simple for a persistent hacker to cycle thru method string combinations in order to discover unintentionally related or unpublished application programming interfaces. The types of Injection attack are given in Table 1.2

<table>
<thead>
<tr>
<th>Vulnerability Types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type I</td>
<td>Lack of clear distinction between data types accepted as input in the programming language used for the Web application development.</td>
</tr>
<tr>
<td>Type II</td>
<td>Delay of operation analysis till the runtime phase where the current variables are considered rather than the source code expressions.</td>
</tr>
<tr>
<td>Type III</td>
<td>Weak concern of type specification in the design: a number can be used as a string or vice-versa.</td>
</tr>
<tr>
<td>Type IV</td>
<td>The validation of the user input is not well defined or sanitized. Inputs are not checked.</td>
</tr>
</tbody>
</table>

**1.6.5 XML VALIDATION**

The XML validation check is performed against any XML Schema (XSchema) or Document Type Definition (DTD) declared inside the XML document. During this process, well-formedness and meaning of XML document will be
checked. XSchema is used to structuring the XML document so that receiver can make sure that XML document content is not tampered.

### 1.6.6 IP ADDRESS SPOOFING

IP Address Spoofing or IP Spoofing is a technique that involves replacing the IP address of an IP Packet’s sender with another machine’s IP address. This technique allows attackers to send the packets anonymously or unauthorized manner. IP address spoofing can be done in different ways to access the particular resource.

### 1.7 ORGANIZATION OF THESIS

The development of access control model was completed effectively and result of the work was analyzed. Also, performance analysis was conducted with the existing models based on some performance criteria. The result of performance analysis shows the system is working properly and performance is also good. The contents of thesis are organized as follows:

**CHAPTER 2** gives the survey of existing models. A number of researchers have done their research in the web services access control. In 1980’s accessing of resources were possible by role based access control model. When the information technology grows, different access control mechanisms are discovered by researchers such as Trust based access model, Fuzzy based access model, Governance based access model, Context based access model etc. Hence, literature survey explains all researchers’ work towards web service access control.

**CHAPTER 3** explains about the overall architecture of multifactor trust based access control model and its components such as SQL Injection Attacks (SQLIAs) Manager, IP address manager and Trust manager briefly. Also it shows the relationship and data flow between the components.
CHAPTER 4 explains about Authentication Manager where actual authentication process takes place by analyzing and retrieving details of requester. SQLIA's manager and IP address manager are the modules that support to detect and prevent SQL injection and IP address spoofing respectively in the authentication manager. In this chapter, the working principles of authentication process are explained with algorithm.

CHAPTER 5 describes about how Authorization manager and Trust manager decide whether to allow or deny the accessing of web service by evaluating the trust value of web service requester based multifactor such as success rate, failure rate, server error, and timeout etc. The components of trust manager such as Trust Decision Point (TDP), Trust Management Point (TDP) and Trust Negotiation Point (TDP) are essential modules to calculate and negotiate the trust level of requesters dynamically. If registered requesters trust level is greater than threshold value then requester will be allowed otherwise service access will be ignored.

CHAPTER 6 explains about prototype implementation details and a number of experiments are carried out while analyzing the result analysis. Those results are analyzed to check the system efficiency.

CHAPTER 7 focuses on performance analysis of the developed access control model which was conducted with existing traditional access control models such as role based access model and attribute access control model.

CHAPTER 8 concludes the features of research work and discusses future research directions in the newly developed model.