

P-3050



**MODELING AND ANALYSIS OF A WEB SERVICE FIREWALL
USING COLOURED PETRI NETS**

A PROJECT REPORT

Submitted by



ARAVIND KUMAR.S	71206106005
ARUL JOSEPH.K.P	71206106007
KARTHICK.S	71206106022

In partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

Electronics and Communication Engineering

KUMARAGURU COLLEGE OF TECHNOLOGY

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “MODELING AND ANALYSIS OF A WEB SERVICE FIREWALL USING COLOURED PETRI NETS” is the bonafide work of “ARAVIND KUMAR.S, ARUL JOSEPH.K.P, KARTHICK.S” who carried out the project work under my supervision.


SIGNATURE

Dr. Rajeswari Mariappan Ph.D.,

HEAD OF THE DEPARTMENT,

Department of Electronics and
Communication Engineering
Kumaraguru College of Technology
Coimbatore-641006

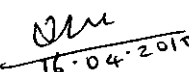

SIGNATURE

Mrs. D. Mohanageetha M.E,

SUPERVISOR,

Assistant Professor,
Department of Electronics and
Communication Engineering
Kumaraguru College of Technology
Coimbatore- 641006

The candidates with university register numbers 71206106005, 71206106007, 71206106022 were examined by us in the project viva voce examination held on 16/04/2010


16.04.2010
INTERNAL EXAMINER


EXTERNAL EXAMINER

April 6, 2010

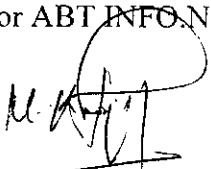
CERTIFICATE

This is to certify that the following, **B.E-ECE Final Year students of KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE**, have successfully completed their Project work entitled "**MODELLING AND ANALYSIS OF WEB SERVICE FIREWALL USING COLOURED PETRINET**", as a part of their course, in our company from **JANUARY 2010 to APRIL 2010**.

1. S.ARAVIND KUMAR - 71206106005
2. K.P.ARUL JOSEPH - 71206106007
3. S.KARTHICK - 71206106022

They have evinced keen interest in absorbing the nature, concept and functions of our organisation and their conduct and character were good during the period.

For ABT INFO.NET



M.K. RANJITH KUMAR
(Head - Technical Operations)
(A division of ABT, Ltd.)
NPT, Mekkinampatti Post
Mekkinampatti Post

ACKNOWLEDGEMENT

All that have started well in well and at this instant of time we would like to thank *Director Dr.J.Shanmugam Ph.D.*, who is instrumental in initiating us to do this work.

We are greatly indebted to our beloved *Principal Dr. S.Ramachandran, Ph.D.*, who has been the backbone of all our deeds.

We take this opportunity to thank **Dr.Rajeswari Mariappan, Ph.D.**, *Head of Department*, Department of Electronics and Communication Engineering, for her continuous motivation towards completion of our project work.

We are highly grateful to our beloved *Project Coordinator Prof.A.Vasuki, M.E.*, and *Project Guide Mrs.D.Mohanageeha, M.E.*, Assistant Professor, for their valuable guidance, throughout the project work.

We would like to express my sincere thanks to **Mr.M.K.Ranjith Kumar, Technical Head, ABT info.net, Pollachi** for his continuous support, valuable suggestion and advice throughout this project.

Last but not least, we express our heartfelt thanks to the Almighty for the blessings. Without His permission and blessings it would not have been possible to complete this project work.

***“WE DEDICATE THIS PROJECT TO OUR PARENTS,
FRIENDS AND OUR PROJECT GUIDE
WHO HAVE INSPIRED US...”***

Abstract

Web services are components defined by WSDL, registered by UDDI and invoked by SOAP protocols. The port used by web services and SOAP is not typically blocked by conventional firewalls. Therefore, a new type of firewall named web service firewall or XML firewall is required. In this project, a novel architectural design for a web service firewall is presented which supports authentication and authorization mechanisms. Instead of filtering packets, WS firewall filters SOAP messages. It also provides prevention of SOAP-based attacks. A formal model for the access control of the proposed architecture using coloured Petri nets (CPNs) is also presented. The CPN model is used for the analysis of the proposed architectural design. The model can also be served as a high-level design for implementation of the web service firewall.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	ABSTRACT	i
	LIST OF TABLES	vi
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	viii
1.	INTRODUCTION	1
2.	LITERATURE REVIEW	
	2.1 Existing system	2
	2.2 Proposed system	3
	2.2.1 The Proposed Architecture of a Web Service Firewall	4
	2.2.1.1 Validation Unit	4
	2.2.1.2 Administrative Interface	6
	2.2.1.3 Database	7
	2.3 A CPN Model for Access Control in Web Service Firewall	7
	2.4 Analysis of the CPN Model	11
3.	SYSTEM SPECIFICATIONS	
	3.1 Hardware Requirements	14
	3.2 Software Requirements	14

4.	WEB DEVELOPMENT ENVIRONMENT	
	4.1 Web Development	15
	4.2 Information Processing Systems	15
	4.2.1 Intranets	16
	4.2.2 Internets	16
	4.2.3 Extranets	16
	4.3 Web-Based	17
5.	THREE-TIER, CLIENT/SERVER ARCHITECTURE	
	5.1 Web Development Skills	21
6.	ABOUT PHP	
	6.1 Configuring PHP Securely	25
	6.1.1 Safe Mode	25
7.	MySQL	
	7.1 MySQL Values	27
	7.2 High Performance only a Main Memory Database can deliver	27
	7.3 Extremely Fast Automatic Failover	28
	7.4 Flexible Distributed Architecture with no Single Point of Failure	28
	7.5 Significantly Reduces Costly Downtime	28
	7.6 Lower Maintenance Costs	29
	7.7 Easy to use Administration	29

8	FEASIBILITY REQUIREMENTS	
	8.1 Classification	30
	8.1.1 Technical Feasibility study	30
	8.1.2 Social Feasibility study	31
	8.1.3 Economic Feasibility study	31
9.	SYSTEM TESTING	
	9.1 Testing	32
	9.1.1 Unit testing	32
	9.1.2 Design the set of tests	33
	9.1.3 Functional testing	34
	9.1.4 Non-functional testing	34
	9.1.4.1 Performance testing	35
	9.1.4.2 Reliability testing	35
	9.1.4.3 Security testing	35
	9.1.5 White Box testing	35
	9.1.6 Black Box testing	36
10.	SYSTEM IMPLEMENTATION	
	10.1 Implementation	37
	10.2 Maintenance	37
11.	CONCLUSION AND FUTURE ENHANCEMENTS	
	11.1 Conclusion	39
	11.2 Future Enhancements	39

APPENDIX

CPN model of access control of WS firewall	40
Sample codes	41
Results	65

REFERENCES	75
-------------------	----

LIST OF TABLES

S.NO	NAME	PAGE NO
1.	Analysis results of the CPN model	13

LIST OF FIGURES

S.NO	NAME	PAGE NO.
1.	The modeling scenario of a web service firewall	3
2.	Architecture of the web service firewall	5
3.	Checking user existence	8
4.	Getting Role and Policy requirements	10
5.	Access Permission	11
6.	Hardware and software layers of a three-tier Information processing system	19
7.	CPN model of access control of Web Service Firewall	40
8.	Screen shot of Home Page	65
9.	Screen shot of Admin access	66
10.	Screen shot of Add user	67
11.	Screen shot of User entrance	68
12.	Screen shot of User policy	69
13.	Screen shot of Session out	70
14.	Screen shot of Database	71
15.	Screen shot of Query statistics	72
16.	Screen shot of Server traffic	74

LIST OF ABBREVIATIONS

SYMBOLS	NAME
XML	eXtensible Markup language
SOAP	Simple Object Access Protocol
DoS	Denial of Service
SQL	Structured Query Language
WSA	Web Service Architecture
SOA	Service-Oriented Architecture
RBAC	Role Based Access Control
CPN	Colored Petri Nets
SCC	Strongly Connected Component
PHP	Hypertext PreProcessor
IE	Internet Explorer
HTML	HyperText Markup Language
EDI	Electronic Data Interchange
WWW	World Wide Web
FTP	File Transfer Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
NOS	Network Operating System
DBMS	Database Management System
SMTP	Simple Mail Transfer Protocol
IIS	Internet Information Server
WSDL	Web Service Description Language
UDDI	Universal Description Discovery and Integration
XHTML	eXtensible HyperText Markup Language
B2B	Business 2 Business

ASP	Active Server Pages
JSP	Java Server Pages
GUI	Graphical User Interface
SAX	Simple API for XML
API	Application Programming Interface

1. Introduction

Web services provide a standard method that supports interoperable machine to machine interaction over the Internet. Web services use the *extensible markup language* (XML) standard for exchanging information. The *simple object access protocol* (SOAP) is a protocol for exchanging XML-based message. As more businesses deploy web services over the Internet, security consideration becomes very critical for the successful deployment of web services application. New attacks are targeting web services that are not protected. A known weakness of web services is their vulnerability to *denial of service* (DoS) attacks exploiting XML processing characteristics. Furthermore, a potential intruder can send malicious SOAP attachments, insert harmful SQL code or executable commands into an XML packet, or send an extremely large XML packet to overload the XML parser on the service provider side. A conventional firewall usually does not examine the content of a packet; thus, it is not able to identify threats such as SQL injection, denial of service, schema poisoning, and XML parameter poisoning. They do not verify user permissions and examining packet contents at the application layer, so they are not suitable for protecting service providers from unauthorized web service invocations.

2. Literature review

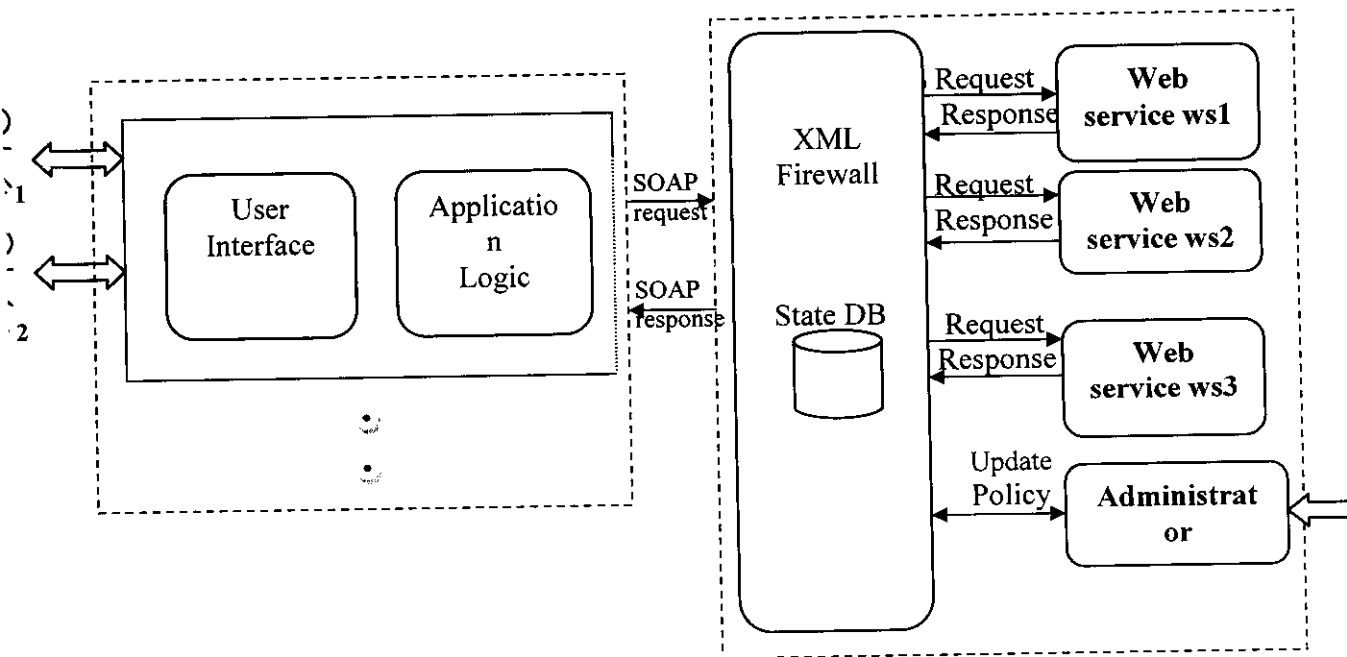
2.1 Existing System

As the best of our knowledge, there are little works on how to protect web service providers from being attacked. Fernandez et al have proposed a pattern-based language for XML firewall description. Yin et al have presented the design of the firewall architecture and filtering policies for an XML firewall. The firewall is implemented using Java language. Bebawy et al. develop *Nedgty*, which is an open source web services firewall. It has the ability to secure web services against denial of service (DoS), buffer overflow, and XML DoS attacks. Cremonini et al have proposed an XML-based approach for combining firewalls and web services security specification. They have discussed about the security requirements of web service architecture (WSA). However, the technical details about implementation of their approach are not accessible. NIST has described the authorization models most relevant to access management in a service-oriented architecture (SOA) – namely rolebased, attribute-based, policy-based, and risk-adaptive access control. Ayachit and Xu have proposed a formal model for XML firewall security using rolebased access control (RBAC). The proposed model supports user authentication and user authorization according to the information stored in a user database and a policy database associated with an XML firewall. The model is formally defined using the ordinary Petri nets. The analysis results show that the Petri net model is live.

2.2 Proposed System

Clearly, the existing architectures introduced in the previous section and the existing products and implementations can help to protect web services, but their functionalities are very limited. For example, they do not provide any access control mechanisms for users, and thus, unauthorized user may access web services with insufficient permissions. Furthermore, Ayachit design does not support prevention of some attacks on web services such as buffer overflow, parameter tampering, recursive payload and SQL injection. Different from the existing works, this project designed a WS firewall that supports authentication and authorization mechanisms. It also provides prevention of SOAP-based attacks.

Web service firewall (WS firewall) or XML firewall is a software package or appliance that filters the XML traffic upstream of a web service and blocks unauthorized traffic before it can reach a protected web service.



WS firewall performs security services such as authentication, authorization, auditing (AAA) and XML validation at a message level. A basic functionality within any WS firewall is authentication and access control. The access control of a web service is the part responsible for granting access to the content and/or functions for justified users. Currently, formal modeling is widely used for specification, verification, evaluation and validation purposes. A useful formal method is Petri nets. Colored Petri nets (CPNs) are an extension of the ordinary Petri nets. CPN is a discrete event modeling language combining Petri nets with the functional programming language standard ML. In this system CPNs are used to model the access control of a web service firewall.

2.2.1 The Proposed Architecture of a Web Service Firewall

The architecture of the WS firewall is divided into three modules, namely **validation unit**, **administrative interface** and **database** (as shown in Figure 2.1). These modules are illustrated in the following paragraphs

2.2.1.1 Validation unit

This is the main component that processes and handles SOAP messages. The validation unit is further divided into the following three sub-modules:

SAX parser: Messages that are sent to the WS firewall are intercepted and parsed with SAX parser to check the validity and the authenticity of the contents.

Filtering policy: If the contents of the parsed messages do not conform to the policies that have been set in the *Policy_DB* database, the messages will be dropped by the WS firewall. Messages that pass the validity check will then be forwarded to the access control sub-module.

Access control: In the access control sub-module, the user is authenticated by checking against the *User_DB* database. If the user's identification is valid, a role from the *Role* database is assigned to the user; otherwise, an *access denied* message is sent to the application. The role assignment is based on the current state of the user as well as the state of the system, which is determined by the status of incoming message and the information stored in the *State_DB* database. After the role assignment is done, a user space is created by using policies from the *Policy_DB* database, which contains the access permissions of the user. The user space is then compared with the service request to determine whether the incoming request from the user has permissions to invoke the web service

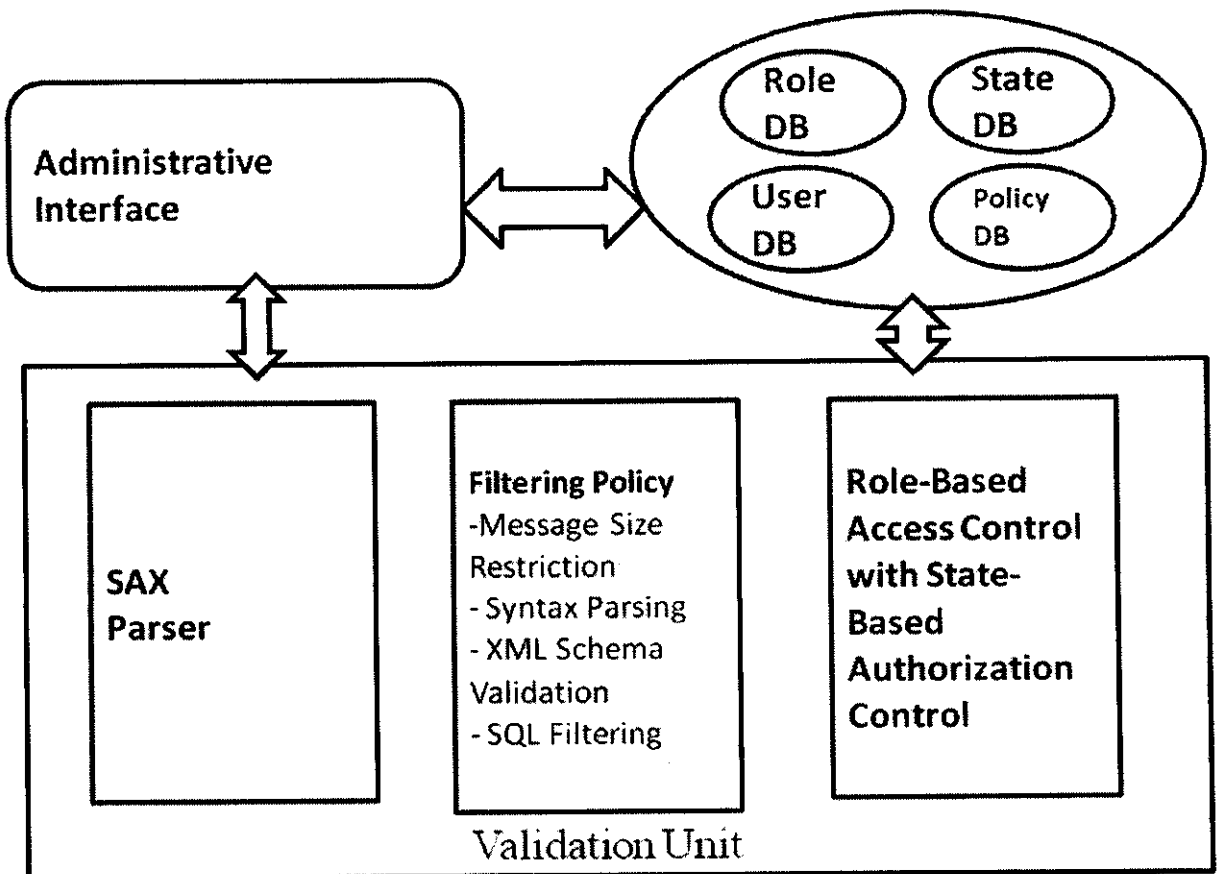


Fig 2.2 Architecture of the web service firewall

If the user has the required permissions, the web service is invoked; otherwise, an

based access control is suggested. However, this strategy is inadequate for web services firewall as it does not consider the context information. The context of a user (i.e. location, time, system resources, network state, network security configuration) is highly dynamic, and granting a user access without taking the user's current context into account can compromise the security. Although a lot of works have been done in the area of access control, most of these works are user-centric, where only credentials of the user are considered when granting access permission. In WS firewall design, the current state of the user is very trivial. The policy assignment considers not only "who the user is?" but also "what is the user's state and the user's environment state?" Hence the user access is granted by user's current status. This allows having more control over the authorized users who can damage the system by using some kinds of attacks such as DoS. In the state-based authorization control system, factors such as time, physical location, previous history, number of requests per minute are taken into account for making a decision. State information can be the number of invocations of a WS per minute associated with a user. If such invocations are too frequent, they shall be denied. This strategy may effectively protect web services from DoS attack.

2.2.1.2 Administrative interface

By using administrative interface, the administrator can add, edit and delete web service profiles. In each profile, the administrator specifies the WS operations, where the WS is located, and the accepted SOAP message formats.

2.2.1.3 Database

The database is used for storing policies, access permissions of the users, valid users' information and their role, and the state information of the users.

2.3 A CPN Model for Access Control in Web Service Firewall

One approach for the presentation of processes and systems is based on a formal model called Petri nets, which was developed in the early 1960s by C. A. Petri. Many extensions to the simple Petri nets have been developed for various modeling and simulation purposes. Coloured Petri nets (CPNs or CP-nets) are an extension of the ordinary Petri nets, which allow different values for a token. The modeling and analysis of CPN models are supported by powerful modeling tools, such as the CPN Tools. The modeling scenario of the WS firewall is illustrated in Figure 2.1. As shown in the figure, the WS firewall protects three web services named *ws1*, *ws2* and *ws3* of a single WS provider. Two roles named *A* and *B* in the system are assumed. Role *A* can invoke *ws1* and *ws2* and role *B* can invoke *ws1* and *ws3*. The WS firewall checks the requests from the application for authenticity and access limitations. If the request is valid, the WS firewall will pass the request to the corresponding web service; otherwise, the request is rejected. Figure 2.3 to 2.5 shows the CPN model of the access control in the WS firewall. For user entrance to the application, *User_Entrance* transition has been considered. Also, an invocation request in *Req_Types* place contains *ws1*, *ws2* and *ws3*. The $1\ ws1$ sign means there is one token with *ws1* value in the place. When the *App_Logic* transition generates a WS invocation request, a token containing user identity and request type is placed into the *WS_Request* place indicating the user

request entrance to WS firewall for access control check. If the user is an existing user, the *Existing_User* transition is enabled. This check is done with guard $[(mem\ j\ id)]$. If the user's identity is not found in the *User_DB* database, the user is recognized as a first time user and the *First_Time_User* transition is enabled. For every first time user, the *BG_Check* transition fires. A first time user becomes a valid user if the user passes the background check and a token is placed into the *Valid_User_Req* place. If the user is a valid user, a token is placed into the *Pass* place and then the *Update_DB* transition will fire to update the *User_DB*, *Role_DB* and *State_DB* database. As a result, a token is placed into the *Valid_User_Req* place. It is assumed that the condition to pass the background check is an even value for the user's identity. In fact this condition is evaluated using a function, which indicates that the user is authenticated or not. Therefore, if the user's identity is odd, the user authentication fails and a token is deposited into the *fail* place.

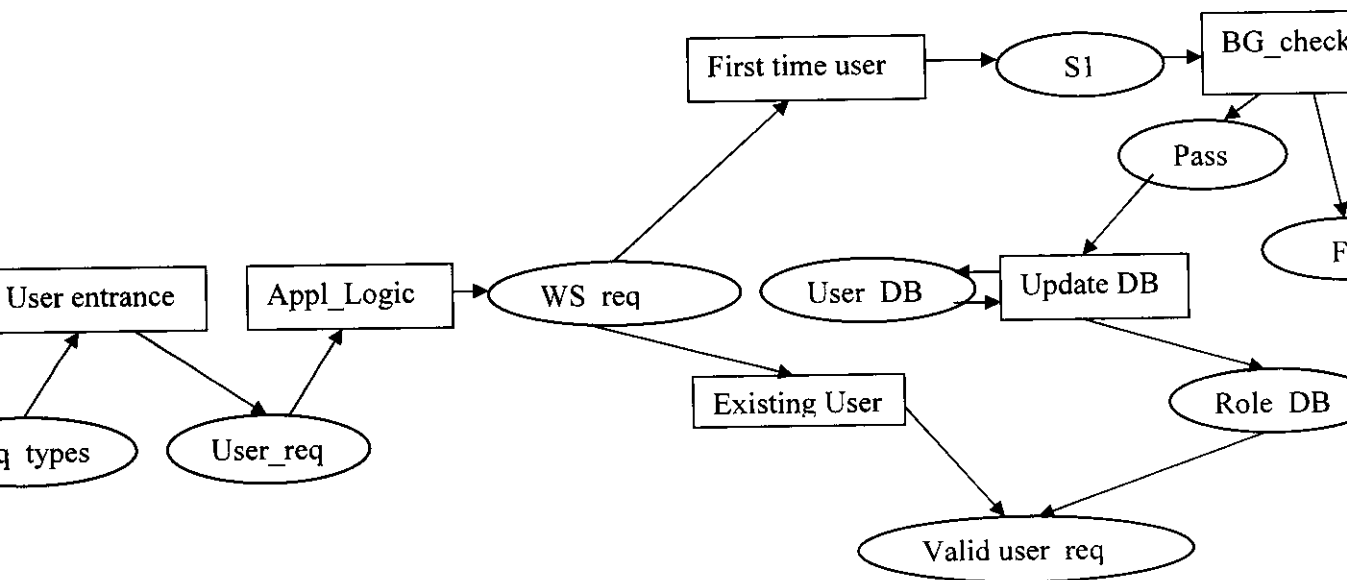


Fig 2.3 Checking user existence

If the user passes the authentication process, a token is deposited into the

Start_Authorization transition. State information for user request is generated by firing *Get_State_Info* transition. The *Get_State_Info* transition can fire when there is a token in the *S3* place. This transition uses state information stored in the *State_DB* database. State information can be configured by the administrator of the firewall.

In this model, it is assumed that the state information is the number of invocations of a web service associated with a user. After the state information is generated, a token indicating the current state of the request is deposited into the *State_Info* place. The *Get_State_Info* transition updates the state information in the *State_DB* database. Once a token is deposited into the *State_Info* place, the *Assign_Role* transition will fire using role information stored in the *Role_DB* database. In the modeling scenario each user can invoke the web services only for three times and if the *state_info* is 3, then the major role of the user is not assigned. When there is a token in the *Role_Info* place, the *Get_Policy_Info* transition can fire. The *Policy_DB* database contains users' access permissions. Then, a token containing the user's access permission is deposited into the *Perm_Info* place and user session is created by firing *Create_Session* transition based on the user's identity and their access permissions, and then a token is deposited into the *Session* place. The user session is the period of time a user interfaces with a WS firewall during the web service invocation. After the user session is created, the *Check_Perm* transition is enabled.

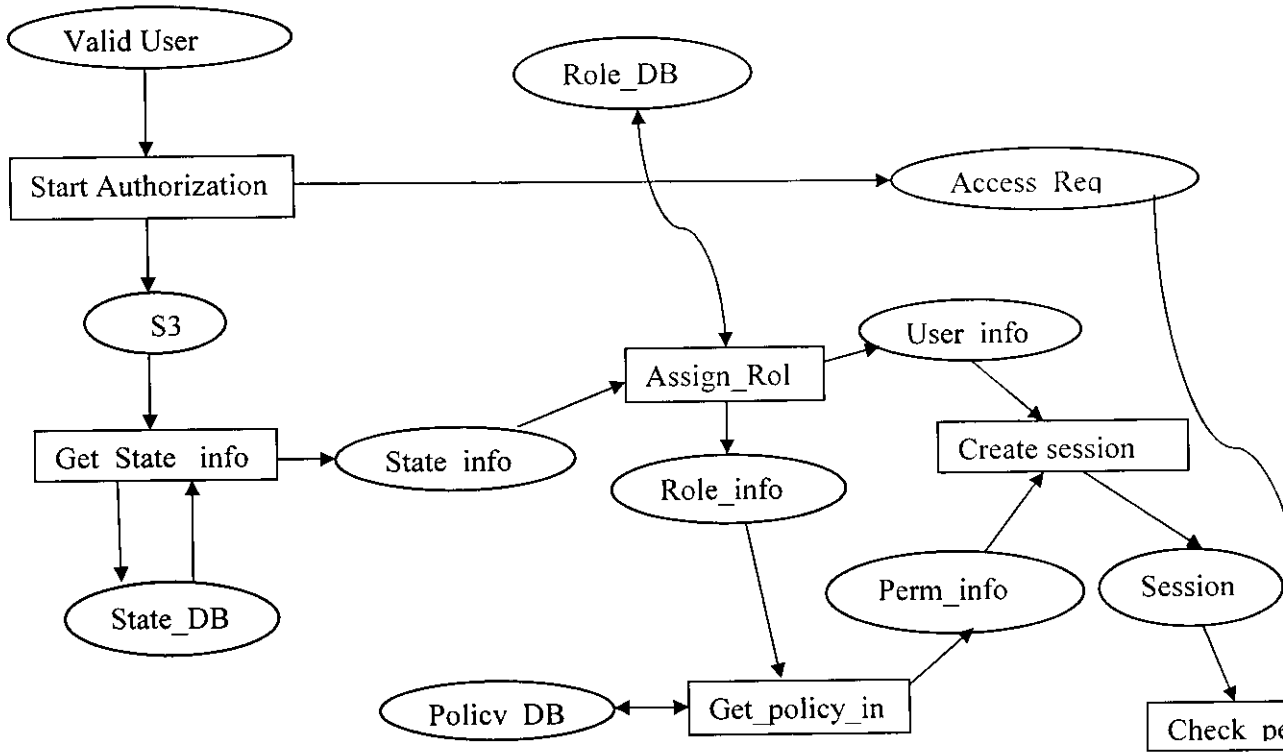
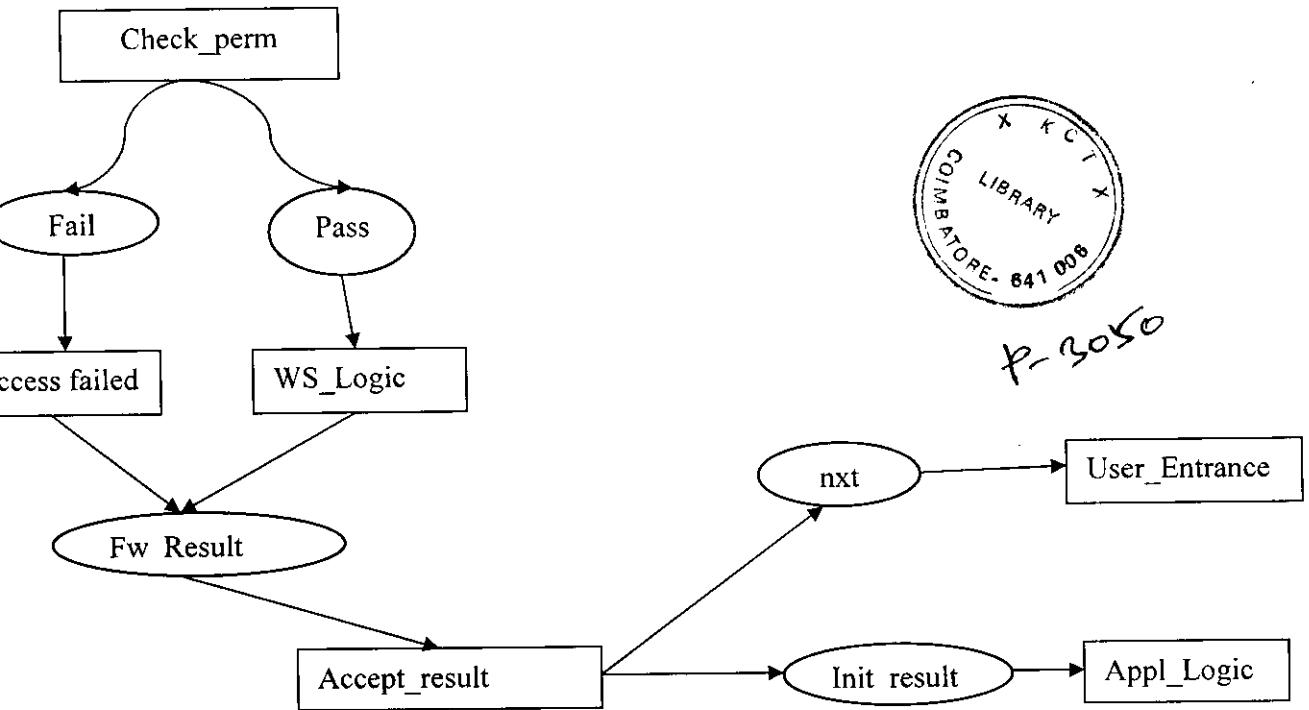


Fig 2.4 Getting Role and Policy requirements

Check_perm transition can fire to check the access request with user access permission. If the user has permissions to invoke a web service, a token that indicates the web service access is passed, will be deposited into the *pass* place, and a web service request will be sent to the corresponding web service. After firing *WS_Logic* transition, a token representing the result of the web service invocation is placed into the *Fw_Result* place. On the other hand, if the user has not enough permission to invoke a web service, a token that indicates the web service access is failed, will be deposited into the *fail* place and then the *Access_failed* transition can fire. When the *Access_failed* transition fires, a token is deposited into the *Fw_Result* place, and then the *Accept_Result* transition will fire.

another token is placed into the *nxt* place. Now the CPN model restarts to its initial state.



P-3050

Fig 2.5 Access Permission

2.4 Analysis of the CPN Model

CPN Tools is a high level modeling tool, which supports the basic Petri nets plus timed Petri nets and CPNs. It has a simulator and a state space analysis tool. Full and partial state spaces can be generated and analyzed. State space report contains information such as *boundedness* and *liveness* properties. The first part of the state space report of the CPN model of the previous section looks as shown in Table 2.1. It contains the statistical information about the size of the state space. The statistical graph of the state space, part also contains information about the strongly connected component (SCC) A SCC is a maximal subgraph in which it is

determine certain kinds of behavioral properties, and they can be calculated by standard algorithms) about the size of the state space. The second part of the state space report contains information about the integer and multi-set bounds. The third part of the state space report provides information about home and liveness properties. A home marking is a marking which is reachable from all reachable markings. A dead marking is a marking with no enabled transitions. The fourth and final part of the state space report provides information about the fairness properties. For model checking, the state space analysis technique of the CPN Tools is used. Analysis results are shown in Table 2.1. The results tell us that there are no dead markings and dead transitions in the net model, and the model is live. The simulation results show that only authenticated and authorized users can access web services. Also it indicates that the access control model is working as expected.

Table 2.1 Analysis results of the CPN model

<i>Statistics for State Space</i>		
Nodes	188	
Arcs	206	
Sees	0	
Status	<i>Full</i>	
<i>Liveness Properties</i>		
Dead Markings	<i>None</i>	
Dead Transition Instances	<i>None</i>	
<i>Live Transition Instances</i>		
Xml FirewallAccept Result	1	
Xml FirewallAccess failed	1	
Xml FirewallApp Logic	1	
Xml FirewallAssign Role	1	
Xml FirewallBFG Check	1	
Xml FirewallCheck Perm	1	
Xml FirewallCreate Session	1	
Xml FirewallExisting User	1	
Xml FirewallFirst Time user	1	
Xml FirewallGet Perm Info	1	
Xml FirewallGet State Info	1	
Xml FirewallStart Authorization	1	
Xml FirewallWS Logic	1	
Xml Firewalluser entrance	1	
<i>Fairness Properties</i>		
Xml FirewallAccept Result	1	<i>Impartial</i>
Xml FirewallAccess failed	1	<i>Fair</i>
Xml FirewallApp Logic	1	<i>Impartial</i>
Xml FirewallAssign Role	1	<i>Fair</i>
Xml FirewallBFG Check	1	<i>Fair</i>
Xml FirewallCheck Perm	1	<i>Fair</i>
Xml FirewallCreate Session	1	<i>Fair</i>
Xml FirewallExisting User	1	<i>Fair</i>
Xml FirewallFirst Time user	1	<i>Fair</i>
Xml FirewallGet Perm Info	1	<i>Fair</i>
Xml FirewallGet State Info	1	<i>Fair</i>
Xml FirewallStart Authorization	1	<i>Fair</i>
Xml FirewallUpdate DB	1	<i>Fair</i>
Xml FirewallWS Logic	1	<i>Fair</i>
Xml Firewalluser entrance	1	<i>Impartial</i>

3. System Specifications

3.1 Hardware Requirements:

1. CISCO 1800 Series
Integrated Service Router : 16 port.10/100 Mbps
2. Power Distribution System
3. Apache HTTP Server
4. Network Racks
5. Processor Type : Pentium III
6. Processor Speed : 1.3GHZ
7. RAM : 512 MB RAM

3.2 Software Requirements

Operating System	:	Windows 2000, XP, 2003
Language	:	PHP, MY SQL, XML
Web Server	:	IIS
Web Browser	:	IE, FireFox
UDDI version	:	UDDI 3.0.2
WSDL version	:	WSDL 2.0

4. Web Development Environment

Authoring Web pages is not a particularly difficult task now-a-days. Many standard desktop software packages come equipped with built-in features to convert word processing documents, spreadsheets, databases, and the like, to coded documents that are ready for access across the Web. Special Web page authoring packages such as Microsoft FrontPage and Macromedia Dreamweaver permit creation of Web pages with drag-and-drop ease. In most of these cases it is not even necessary to know or to be aware of the special **HTML (HyperText Markup Language)** coding that takes place behind the scenes.

4.1 Web Development

Web "development," as contrasted with Web page "authoring," goes well beyond the use of markup codes and a few plug-ins or scripting techniques to make attractive and informative Web pages. The term pertains to the use of special strategies, tools, and methods for producing Web pages and Web sites characterized as **information processing systems**. Consider this term in more detail to understand the broader-ranging purposes for which Web pages and Web sites are developed.

4.2 Information Processing Systems

Web technologies are used to produce not just simple personal or promotional Web sites containing informative, interesting, or entertaining material for public consumption. Rather, they are becoming important means for supporting the foundational "business processes" of modern organizations -- the underlying

supporting these purposes are roughly classified into three types of Web-based systems, termed **intranets**, **internets**, and **extranets**.

4.2.1 Intranets

Intranets are private, internal systems to help carry out the day-to-day information processing, management information, and work-flow activities of organizations. Web-based intranets service the standard internal business functions and in doing so impact basic organizational systems such as accounting and financial reporting systems, marketing and sales systems, purchasing and distribution systems, production systems, and human resource systems. In time, Web-based intranets will become the primary technical means through which organizations function internally to carry out their business processes.

4.2.2 Internets

Internets are public information systems. They include public sites that provide news, information, and entertainment; electronic commerce sites to market and sell products and services; governmental sites to inform or service the general public; and educational sites to provide local and remote access to education and training. In all sectors of society, public internets are providing goods, services, and information to the public through the World Wide Web and its associated networks and services.

4.2.3 Extranets

Extranets are business-to-business (B2B) systems that manage **electronic data interchange (EDI)** between business enterprises. These systems facilitate the flow of information between organizations -- between a company and its suppliers

purchasing, production, and distribution. Electronic data interchange helps eliminate the paper flow accompanying business transactions by using Web technologies to transfer electronic documents for processing between computers rather than between people. As Web-based systems, EDI applications eliminate the difficulties of communicating information among different hardware and software platforms with inherently different information formats and with different protocols for exchanging information.

The Web is becoming the primary technological basis, the electronic highway, for information collection, processing, and distribution in all types of organizations -- in commercial and financial enterprises, educational institutions, government agencies, health-care facilities, news and entertainment industries, and in most other formal organizations both large and small. It is the pervasive technology for developing information processing systems in all sectors of society.

4.3 Web-Based

The term "Web-based" refers to the fact that information processing systems rely on the technology of the Internet, particularly that portion known as the **World Wide Web (WWW)**. Thus, Web-based systems operate within a technical framework with the following characteristics:

First, systems operate across public, rather than private, data networks. They communicate over the Internet, the world-wide interconnected networks of computers that are publicly accessible.

Second, the communications networks are based on open and public technical standards such as Ethernet architectures, TCP/IP transmission protocols,

and HTTP and FTP application protocols. These are not private or proprietary standards, but are fundamentally open and free to public use.

Third, Web-based processing systems use common, often-times free, software for development and operation. Processing activities take place through Web browsers rather than specially written software for the user interface and for front-end data collection and processing. Microsoft Internet Explorer and Netscape Navigator, among others, are the means through which individuals interact with information processing systems. Also, common Web server computers perform the back-end business processing functions, and database servers provide information storage, access, and retrieval.

Thus, common, non-specialized, non-proprietary hardware and software systems provide the technical environment for developing information processing systems and for operating and managing information processing activities.

5. Three-tier, Client/Server Architecture

The term "client/server" pertains to the use of server-based networks to manage resource sharing and to distribute processing tasks among hardware and software components. Within Web-based client/server networks the distribution of processing tasks occurs in three tiers that correspond to the three primary hardware/software components of the system.

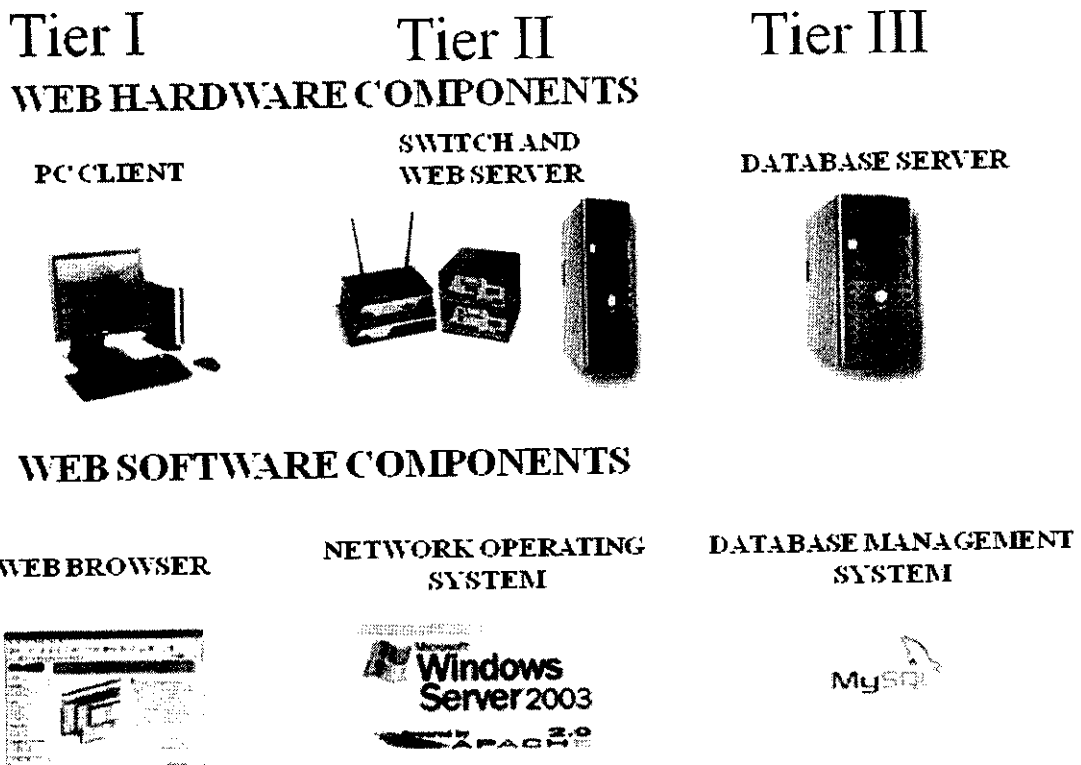


Fig 5.1 Hardware and software layers of a three-tier information processing system

In Tier 1 the desktop **PC client** handles the user interface activities of the system; in Tier 2 the **Web server** handles the primary processing functions of the system; and in Tier 3 the **database server**, and in certain cases the **media server**, handles information storage and retrieval functions required by the system.

In turn, each of the three hardware components host corresponding software. The client software is a **Web browser**. The Web server runs a **network operating system (NOS)** such as Windows Server software, Unix Server software, or Linux Server software and through component software such as Internet Information Server or Apache Web Server hosts World Wide Web, FTP, SMTP mail, and other Internet services. The database server runs a **database management system (DBMS)** such as MySQL, Oracle, Access, and other popular packages. Thus, separate components perform separate processing tasks that are integrated through the Web into a complete information processing system.

Consider, for instance, a visit to an e-commerce Web site such as Amazon.com. The Web browser is the interface with the site. In response to various "input" requests that are submitted as they navigate the items for sale, various "output" pages are produced. The requests are entered into the system through Web links and form submissions; system responses produce HTML pages delivered back to the browser for display on the screen. The browser performs the input and output activities needed to interface with the site.

Behind the scenes special information processing tasks are taking place on the Web server. On requesting a book search, for example, programs are run to search databases to extract matching books and to format the output for delivery to the browser. When viewing the shopping cart, other routines retrieve the choices and calculate the order total. While checking out, special programs are run to link into the credit checking and banking systems so that appropriate accounts are debited and credited. Myriad processing task associated with the browsing and purchasing take place on Web servers, hidden from the view but crucial to shopping experience and to formalizing the business transactions that result.

Most of the information that is captured and generated by the shopping visit is kept in large databases hosted on separate database servers. All of the book information that seen on screen is extracted from database tables. The purchase selections are stored in database tables. Virtually every piece of information regarding the products viewed and the purchased transactions are maintained in massive databases within the e-commerce system itself or in associated databases central to the accounting, purchasing, and distribution systems that surround it.

In even the smallest Web-based commercial systems the same functionality is present. The Web browser provides the user interface to the system, special processing pages handle the business transactions, and one or more databases maintain information flowing through the system. The point is that in Web-based systems of any size the three primary tiers of functionality exist. From the standpoint of the Web developer, then, the task is to build these three separate components -- the user interface, the business processing routines, and the database maintenance components -- and to integrate them into a fully functioning information processing system.

5.1 Web Development Skills

Web development pertains to the use of Web technologies to build client and server processing components, to integrate them as applications within intranet, internet, and extranet processing systems, and to deploy them across the Web to conduct the private and public business affairs of organizations. The skill set to accomplish these tasks range well beyond the ability to save Web pages from a

word processing program, to drag and drop a simple Web site with a desktop package, or even to hard-code pages with XHTML and a scattering of plug-ins.

In the final analysis, the Web developer needs insight into the operational and management processes of organizations, an understanding of how work flows produce and rely upon information flows in the production of goods and services, an ability to abstract and model these business systems around the hardware and software technologies available, and the skills to employ those technologies to build Web-based systems that operationalize those models.

6. About PHP

PHP stands for Hypertext Preprocessor. It is a server-side programming language specifically designed for creating dynamic web pages. The language was originally developed in 1994 by Rasmus Lerdorf and has since been expanded to become one of the WWW's most popular scripting languages. According to 2005 Netcraft statistics, PHP is currently being used in over 23,000,000 domains. Like other types of server-side languages such as ASP, ASP.NET, and JSP, PHP code is processed on the web server and generates the XHTML code or other output that can be viewed in the browser. Unlike other server-side languages, PHP is an open source product, meaning everyone has access to the source code and can use, alter, and redistribute it all without charge.

The current version of PHP and the version covered in this project is 5. PHP 5 can be run on just about any type of operating system and Web server. However, in order for PHP scripts to be processed, the PHP interpreter must be installed. The software is available in two forms - complete source code and executable binaries. These days, most Linux systems come with the PHP source code

Any Web server can be thought of as a castle under constant attack by a sea of barbarians. And, as the history of both conventional and information warfare shows, often the attackers' victory isn't entirely dependent upon their degree of skill or cunning, but rather on an oversight by the defenders. As keepers of the electronic kingdom, the user is faced with no shortage of potential ingresses most notably:

- **User input:** Exploiting disregarded user input is perhaps the easiest way to cause

... application infrastructure an assertion

backed up by the numerous reports of attacks launched on high-profile Web sites in this fashion. Deft manipulation of parameters emanating from Web forms, URL parameters, cookies, and other readily accessible routes enables attackers to exploit a multitude of routes to strike the very heart of the application logic.

- **Software vulnerabilities:** Web applications are often constructed from numerous technologies, typically a database server, a Web server, and one or more programming languages, all of which run on one or more operating systems. Therefore, it's crucial to constantly keep abreast of exposed vulnerabilities and take the steps necessary to patch the problem before someone takes advantage of it.
- **The inside job:** Shared host servers, such as those often found in ISPs and educational hosting environments, are always susceptible to damage, intentional or otherwise, by a fellow user's actions. Because each scenario poses significant risk to the integrity of your application, all must be thoroughly investigated and handled accordingly.
- Securely configuring PHP via its configuration parameters
- The safe mode security option
- The importance of validating user data
- Protecting sensitive data through common sense and proper server configuration
- PHP's encryption capabilities

Perhaps the best place to start is with a review of PHP's configuration parameters, because taking advantage of them right from the very start, prior to doing anything else with the language.

6.1 Configuring PHP Securely

PHP offers a number of configuration parameters that are intended to greatly increase PHP's level of security awareness. This section introduces many of the most relevant options

■**Note:** Disabling the `register_globals` directive aids tremendously in the prevention of user-initiated attempts to trick the application into accepting otherwise dangerous data.

6.1.1 Safe Mode

Safe mode is of particular interest to those running PHP in a shared-server environment. When safe mode is enabled, PHP always verifies that the executing script's owner matches the owner of the file that the script is attempting to open. This prevents the unintended execution, review, and modification of files not owned by the executing user, provided that the file privileges are also properly configured to prevent modification. Enabling safe mode also has other significant effects on PHP's behavior, in addition to diminishing, or even disabling, the capabilities of numerous standard PHP functions.

7. MySQL

MySQL is a relational database system that is used to store information. MySQL can store many types of data from something as tiny as a single character to as large as complete files or graphics. Although it can be accessed by most programming languages, it is often coupled with PHP because they work together with ease.

Information stored in a MySQL database hosted on a web server can be accessed from anywhere in the world with a computer. This makes it a good way to store information that needs the ability to change over time, but also needs to be accessed over the net. Some examples that can utilize MySQL are a web message board or a customer's shipping status.

MySQL is the world's most popular open source database software, with over 100 million copies of its software downloaded or distributed throughout its history. With its superior speed, reliability, and ease of use, MySQL has become the preferred choice for Web, Web 2.0, SaaS, ISV, Telecom companies and forward-thinking corporate IT Managers because it eliminates the major problems associated with downtime, maintenance and administration for modern, online applications.

Many of the world's largest and fastest-growing organizations use MySQL to save time and money powering their high-volume Web sites, critical business systems, and packaged software — including industry leaders such as Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, Wikipedia, and Booking.com.

The flagship MySQL offering is MySQL Enterprise, a comprehensive set of production-tested software, proactive monitoring tools, and premium support services available in an affordable annual subscription.

MySQL is a key part of LAMP (Linux, Apache, MySQL, PHP / Perl / Python), the fast-growing open source enterprise software stack. More and more companies are using LAMP as an alternative to expensive proprietary software stacks because of its lower cost and freedom from platform lock-in.

7.1 MySQL Values

MySQL database to be:

- The best and the most-used database in the world for online applications
- Available and affordable for all
- Easy to use
- Continuously improved while remaining fast, secure and reliable
- Fun to use and improve
- Free from bugs

7.2 High Performance Only a Main Memory Database Can Deliver

MySQL provides the response time and throughput to meet the most demanding high volume enterprise applications. MySQL achieves its performance advantage by being a main memory clustered database solution, which keeps all data in memory and limits IO bottlenecks by asynchronously writing transaction logs to disk. MySQL also enables servers to share processing within a cluster, taking full advantage of all hardware. Typical response times for MySQL are in the range of a few milliseconds and MySQL has been proven to handle tens of

thousands of distributed transactions per second that are also replicated across database nodes.

7.3 Extremely Fast Automatic Failover

MySQL delivers extremely fast automatic failover time with sub-second response so that applications can recover quickly in the event of application, network or hardware failure. MySQL uses synchronous replication to propagate transaction information to all the appropriate database nodes so applications can automatically fail over to another node extremely quickly. This eliminates the time consuming operation of recreating and replaying log files required by 'Shared-Disk' architectures to fail over successfully. Plus, MySQL database nodes are able to automatically restart, recover, and dynamically reconfigure themselves in case of failures without having to program advanced features into the application.

7.4 Flexible Distributed Architecture with No Single Point of Failure

The parallel server architecture combines database nodes, management server nodes, and application nodes that can be distributed across computers and geographies to ensure there is no single point of failure. Any node can be stopped or started without stopping the applications that use the database. And MySQL is highly configurable so that it can implement the appropriate level of performance, scalability and fault tolerance to match the application requirements.

7.5 Significantly Reduce Costly Downtime

MySQL not only lowers up-front license costs with affordable commercial licensing under a dual licensing mechanism, but it also significantly reduces system downtime - the number one contributor to the Total Cost of Ownership

environment allows the user to cost-effectively distribute their applications using commodity hardware and open source software infrastructure.

7.6 Lower Maintenance Costs

MySQL is designed to be largely self-governing so very few system parameters actually need fine-tuning, further reducing the risk of costly errors. As a result, there are typically fewer conflicts with other software and hardware, and less need for manual intervention. This also means that MySQL will have a much lower maintenance costs, with less fine tuning required by Database Administrators.

7.7 Easy-to-use Administration

MySQL includes easy to use and powerful tools for administering your clustered environment. Command line tools enable you to monitor database nodes, control access to applications, and create and restore backups. \

8. Feasibility Requirements

The development and implementation of a new system is definitely expensive. It requires system resources, manpower, time and money, so it improves the necessity of the feasibility study based on the proposed system requirements. During system analysis, the feasibility study of the proposed system is to be carried out. The main objective of this study is to determine whether the proposed system is feasible or not i.e. to ensure that proposed system is not a burden to the organization.

8.1 Classification

The study can be categorized into three types. They are:

- **Technical Feasibility**
- **Behavioral Feasibility**
- **Economic Feasibility**

8.1.1 Technical Feasibility study

This study is carried out to check the technical facility, i.e. the technical requirements of the system. The assessment of technical feasibility must be based on an outline design of system requirements in terms of input, output, files, programs and procedures. This can be qualified in terms of volumes of data, trends, frequency of updating, cycles of activity etc, in order to give an introduction of technical system.

The system should thus provide technical guarantee of accuracy, reliability, ease of access and data security and through this the hardware and software requirements should be satisfied.

The developed system (Footprint Recognition System) has a modest technical requirement, as only minimal or null changes are required in implementing the system.

8.1.2 Social Feasibility study

This aspect of study is to check the level of acceptance of the system by the user. The levels of the acceptance by the users solely depend on the methods that are employed to educate the user about the system and to make him familiar with it. The level of confidence must be raised so that the user is also able to make some constructive criticism, which is welcomed.

8.1.3 Economic Feasibility study

This study is carried out to check the economic impact that the system will have the organization. The technique of cost benefit analysis is often used a basis for assessing economic feasibility. The system “Graphical Password Authentication” can be developed at a reasonable cost with the available hardware, software and manpower. So its benefits outweigh the cost.

9. System Testing

9.1 Testing

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. Inadequate testing or non-testing leads to errors that may not appear until months later. This creates two problems. The time lag between the cause and appearance of the problem. The effect of system errors on files and records within the system. A small system error can conceivably explode into much larger problem. Effective early in the process translates directly into long term cost savings from a reduced number of errors. Testing is the process of executing a program with the intent of finding any errors. A good test of course has the high probability of finding a yet undiscovered error. A successful testing is the one that uncovers a yet undiscovered error.

9.1.1 Unit testing

A program represents the logical elements of a system. For a program to run satisfactorily, it must compile and test data correctly and tie in properly with other programs. Achieving an error free program is the responsibility of the programmer. Program testing checks for two types of errors: syntax and logical. Syntax error is a program statement that violates one or more rules of the language in which it is written.

An improperly defined field dimension or omitted keywords are common syntax errors. These errors are shown through error message generated by the

When a program is tested, the actual output is compared with the expected output. When there is a discrepancy the sequence of instructions must be traced to determine the problem the process the is facilitated by breaking the program down into self-contained portions, each of which can be checked at certain key points .The idea is to compare program values against desk-calculated values to isolate the Problems.

Unit testing has been performed the module. The syntax and logical error have been corrected than and there. All this syntax have been rectified during compilation. The output has been tested with the manual input. All the data are stored correctly.

9.1.2 Design the set of tests:

The language used for the development of the product is java, so that every action can be tested and hope to get the expected response. A template is created for every test.

So the two test sets are GUI based tests,

Questions:

Test Set 1: For Windows

1. Are all functions that relate to the window operational?
2. Are all relevant, controls, dialog boxes, and buttons available and properly displayed for the window?

Test Set 2: Mouse operations:

2. Does each function perform as advertised?
3. Do multiple or incorrect mouse picks within the window caused unexpected side effects?

Solutions:

Test Set 1:

1. Yes, all functions relate to that window.
2. Yes, all controls & objects are displayed at their appropriate places.

Test Set 2:

1. Yes, all are addressable by the mouse.
2. Yes, function performs as advertised.

9.1.3 Functional testing

Functional testing of an application is used to prove the application delivers correct results, using enough inputs to give an adequate level of confidence that will work correctly for all sets of inputs. The functional testing will need to prove that the application works for each client type and that personalization function work correctly.

9.1.4 Non-functional testing

This testing used to check that an application will work in the operational environment.

Non-functional testing includes:

- **Usability testing**
- **Reliability testing**
- **Security testing**

9.1.4.1 Performance testing

This is required to assure that an application performs adequately, having the capability to handle any workloads, delivering its results in expected time and using an acceptable level of resource and it is an aspect of operational management.

9.1.4.2 Reliability testing

This is to check that the application is rugged and reliable and can handle the failure of any of the components involved in providing the application.

9.1.4.3 Security testing

It is necessary to check that the application's data is secured. It involves checking that the user identification is authenticated. The user is authorized to do what they are doing. All information retains its integrity.

9.1.5 White Box testing

White box testing, sometimes called glass-box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing method, the software engineer can derive test cases. Guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structures to ensure their validity.

9.1.6 Black Box testing

Black box testing, also called behavioral testing, focuses on the functional requirements of the software. That is, black testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing is not alternative to white box techniques. Rather it is a complementary approach that is likely to uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

1. Incorrect or missing functions.
2. Interface errors.
3. Errors in a data structures or external data base access.
4. Behavior or performance errors.
5. Initialization and termination errors.

10 System Implementation

10.1 Implementation

Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and giving confidence on the new system for the users that it will work efficiently. It involves careful planning, investing of the current system, and its constraints on its implementation, design of methods to achieve the change over, and evaluation of the change over methods.

The implementation process begins with preparing a plan for the implementation of the system. According to the plan, the activities has to be carried out, discussion has been made regarding the equipment, resources and how to test the activities.

The coding step translates a detail design representation into a programming language realization. The coding should have some characteristics:

- **Ease of design to code translation**
- **Code efficiency**
- **Memory efficiency**
- **Maintainability**

10.2 Maintenance

Software maintenance is a set of software engineering activities that occur

success of software and the project relies and the maintenance procedure performed. The maintenance is performed at regular intervals to keep the project safe and reliable. Every time changes attempted on the software will cause serious and unexpected side effects. So the maintenance of the software should be considered seriously. Software maintenance is of course far more than fixing mistakes. Maintenance can be described as the activities that are to be undertaken after software is released for the use. The different types of maintenance are:

- **Corrective Maintenance**
- **Adaptive Maintenance**
- **Perfective Maintenance**

The corrective maintenance deals with the problems that may occur to software and what sort of corrective measures can be provided to the user on such situations. The 80% is spent adapting existing systems to change in their external environment since platform independency is used, the software will be adaptive to all the hardware and software environments.

Difficulties

1. Database migration needs to change the provides and the database
 2. plan and sub-plan modification and changes
 3. accounting maintenance and reporting
- risk management issues.

11. CONCLUSION AND FUTURE ENHANCEMENTS

11.1 Conclusion

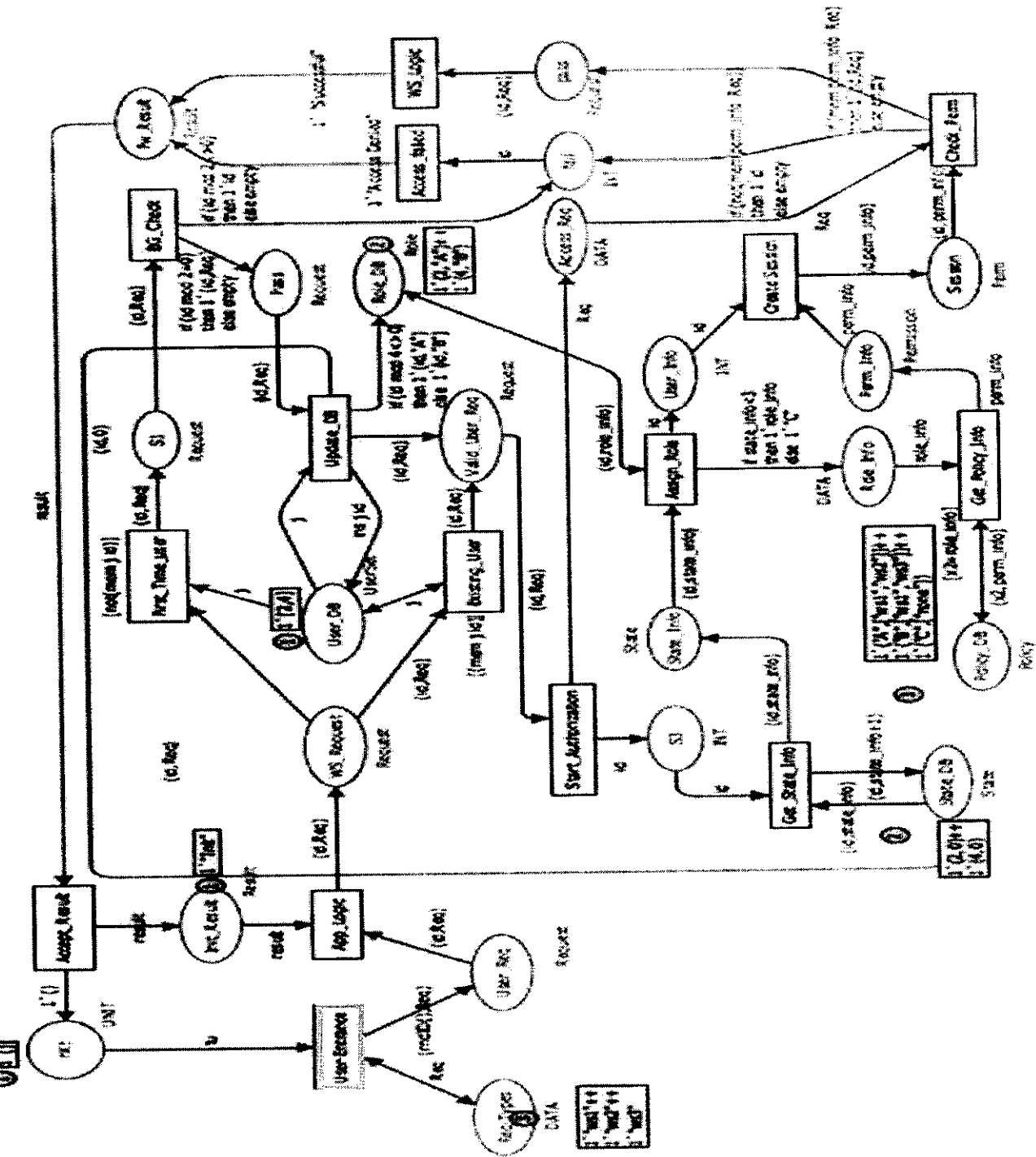
As XML-based services are becoming popular, various security threats emerge targeting XML applications. Due to the fact that network firewalls are unable to overcome these problems, a new type of firewall is required. A WS firewall can be used to solve the shortcomings of network firewall. Since it is an application-based firewall, it can understand application layer protocols. Instead of filtering packets, WS firewall filters SOAP messages. In this project a proposed architecture for a WS firewall for protecting web services is designed. The proposed WS firewall is stateful. Furthermore a formal model for access control of the proposed WS firewall architecture using coloured Petri nets (CPNs) is developed. The analysis results of this model tell us that there are no dead markings in the net model, and the model is live. In addition, the simulation results show that only authenticated and authorized users can access web services.

11.2 Future Enhancements

In this project the malicious attacks only from port 80 is considered, which is assigned uniquely for HTTP. This project can be extended such that the firewall can counter attacks from other ports also.

APPENDIX

CPN model of access control of WS firewall



SAMPLE CODE

Home.php

```
<?
include("../inc.php");
admin_redirect();
?>
<html>
<head>
<title>
<?=$admin_title?>
</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<style type="text/css">
</style>
<link href="./s.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="775" border="0" align="center" cellpadding="0" cellspacing="0">
<tr><td><? include("t.php");?></td></tr>
<tr>
<td><table width="100%" border="0" cellspacing="0" cellpadding="0">
<tr valign="top">
<td width="150" class="rowtitbg"><? include("l.php");?></td><td><br>
<table width="95%" border="0" align="center" cellpadding="0" cellspacing="0"
class="black-text"><tr>
<td valign="top"><table width="50%" border="0" cellpadding="2"
cellspacing="2" class="white-text"><tr>
<td class="rowtitbg">&nbsp;  <strong>Welcome&nbsp;  </strong></td>
</tr></table></td></tr> <tr>
<td valign="top" class="brdr"><table width="100%" border="0" cellpadding="2"
cellspacing="2" class="black-text"><tr>
<td valign="top">Welcome to Modelling web service firewall panel
.Please Select a option form the left side menu.Thanks. <br>
<br>
<? if($_GET[act]=="pp"){?>
<span class="error_text">[ Admin Paynal Address Updated.. ] </span>
```

```

<?}?>
<br></td> </tr> </table></td></tr> </table><br></td></tr></table></td></tr> <tr>
<td></td> </tr> <tr><td><?include("b.php");?></td></tr></table>
</body>
</html>

```

Index.php

```

<? error_reporting(0);
echo $_GET[id]
?>
<?
    $ip=$_SERVER['REMOTE_ADDR'];
    echo $ip;
    $sql1="select * from cmsv2_admin where IPaddress='$ip' ";
    // echo $sql1;

    $res=mysql_query($sql1);

    $i=0;
    while($row=mysql_fetch_object($res))
    {
        $i=1;
    }
    if($i==0)
    {
        //$cdate=date('Y/m/d');
        //$ctime=date('h:m:s');
        //$sql1="insert into adminip(IPADD,cdate)
values('$ip','$cdate <--> $ctime)";
        //mysql_query($sql1);

        //header("location:../forbidden.php");
        //exit;
    }
?>
<html>
<head>
<title>
<?=$admin_title?>
</title>

```



```

        echo "<span class=htext><br>Your User Account is Locked....</span>";
if($_GET[act]=="lout")
    echo "<span class=htext><br>You Have Sucessfully Logged
Out.Thanks.</span>";
    ?>
    </b></td>
</tr>
<tr>
    <td>&nbsp;</td>
</tr>
<tr>
    <td> <table width="300" border="1" cellspacing="0" cellpadding="0"
bordercolor="#345075" align="center" class="font">
    <tr bgcolor="#A85400">
        <td height="28" align="center" class="rowtitbg"><font
color="#FFFFFF"><b class="white-text">Administration Login
Panel</b></font></td>
    </tr>
    <tr>
        <td> <table width="100%" border="0" cellspacing="0" cellpadding="0"
align="center" class="font">
            <tr valign="top">
                <td> <form method="POST" action="soap.php" name="f1"
onSubmit="return checkthis()">
                    <table border="0" cellspacing="0" cellpadding="5" class="black-text"
width="100%" align="center">
                        <tr valign="top">
                            <td width="39%">&nbsp;</td>
                            <td width="61%">&nbsp;</td>
                        </tr>
                        <tr valign="top">
                            <td width="39%" class="menubold" align="center"
valign="middle">User Name</td>
                            <td width="61%"> <input type="text" name="user" size="20"
class="inpborder"> </td>
                        </tr>
                        <tr valign="top">
                            <td width="39%" class="menubold" align="center"
valign="middle">Password</td>

```

```

        <td width="61%"> <input type="password" name="pass" size="20"
class="inpborder"> </td>
    </tr>
    <tr valign="top">
        <td width="39%"> <div align="center">
            <input type="submit" name="Submit" value="Login"
class="inpborder">
        </div></td>
        <td width="61%"> <div align="center">
            <input type="button" name="Cancel" value="Cancel"
onClick="javascript:document.location = '../'" class="inpborder">
        </div></td>
    </tr>
</table>
</form></td>
</tr>
</table></td>
</tr>
</table></td>
</tr>
</table>
</body>
</html>

```

Newu.php

```

<?
include("../inc.php");
admin_redirect();
if($_GET[act]=="a")
{
    $sql="delete from cmsv2_admin where id=$_GET[id]";
    mysql_query($sql) ;
}
?>
<html>
<head>

```



```

{
$cnt++;
$un=1;
}
if($_POST[pass]== "")
{
$cnt++;
$p=1;
}
if($_POST[pass1]== "")
{
$cnt++;
$p1=1;
}
if($_POST[pass] != $_POST[pass1])
{
$cnt++;
$p=2;
}
if($cnt==0)
{
// $sql="insert into qbit_admin (admin_uname,admin_pass,IPAddress) values
($_POST[uname],$_POST[pass],$_POST[IP])";
$sql="insert into cmsv2_admin (uname,pwd,designation,IPAddress) values
($_POST[uname],$_POST[pass],'Administrator',$_POST[IP])";
mysql_query($sql) or die("Error in Query");
echo "The user has been added succesfully";
}
}
?>
</strong></td> </tr>
<tr valign="top" bgcolor="#ebebeb">
<td height="20" align="right" valign="middle"
class="menubold">Username : </td>
<td height="20"><input name="uname" type="text" id="uname"
value="<?=$_POST[uname]?>" size="30">
<?
if($un==1)
{

```

```

        <span class="error-text"> [ REQUIRED ] </span>
        <?
                                }
                                ?>

    </td></tr>
    <tr valign="top" bgcolor="#f4f4f4">
        <td width="40%" height="20" align="right" valign="middle"
class="menubold"> Password :</td>
        <td height="20"><input name="pass" type="password"
class="inpborder" value="<?=$_POST[pass]?>">
        <?
                                if($p==1)
                                {
                                ?>
                                <span class="error-text"> [ REQUIRED ] </span>
                                <?
                                                }
                                                ?></td>

    </tr>
    <tr valign="top" bgcolor="#ebebeb">
        <td align="right" valign="middle" class="menubold">Confirm
Password :</td>
        <td><input name="pass1" type="password" class="inpborder"
value="<?=$_POST[pass1]?>">
        <?
                                if($p1==1)
                                {
                                ?>
                                <span class="error-text"> [ REQUIRED ] </span>
                                <?
                                                }
                                                ?>

        <?
                                if($p==2)
                                {
                                ?>
                                <span class="error-text"> [ Not Matching ] </span>
                                <?
                                                }

```

```

</tr>
<tr valign="top" bgcolor="#f4f4f4">
  <td align="right" valign="middle" class="menubold">IP address :
</td>
  <td><input name="IP" text" class="inpborder"
value="<?=$_POST[IP]?>"></td>
</tr>
<tr valign="top">
  <td align="right" valign="middle" bgcolor="#ebebeb"
class="menubold">&nbsp;</td>
  <td bgcolor="#ebebeb"><input type="submit" name="Submit"
value="Submit" class="inpborder"></td>
</tr></table></form></td></tr></table></td></tr></table>

```

```

<br>
<table width="100%" border="1">
<tr>
  <td width="20%">UUsername</td>
  <td width="15%">Password</td>
  <td width="21%">IP Address </td>
  <td width="21%">Action</td>
  <td width="23%">&nbsp;</td>
</tr>

```

<?

```

$username="admin";
$sql1="select * from cmsv2_admin where
$res=mysql_query($sql1);
$i=0;
while($row=mysql_fetch_object($res))
{
?>

```

```

<tr>
  <td><?=$row->uname?></td>
  <td><?=$row->pwd?></td>
  <td><?=$row->IPaddress?></td>
  <td>[<a href="newu.php?id=<?=$row->id?>&act=a">Delete</a>]</td>
  <td>&nbsp;</td>
</tr>

```

<? } ?>

```
</tr></table></td></tr><tr>
<td height="1" bgcolor="#FFFFFF"></td></tr>
<tr>
<td><? include("b.php");?></td>
</tr></table></body>
</html>
```

Pg.php

```
<?
include("../inc.php");
admin_redirect();
include('lzw.php');

function Read1($file)
{
$kk="";
$fp=fopen($file,"r");
while(!feof($fp))
{
$kk.=fgets($fp,4096);
}
fclose($fp);
$kk=ereg_replace("[\n]","", $kk);
return $kk;
}

//Writing into a file
function Write1($file,$contents)
{
$com = new LZW();
$contents=$com->compress($contents);
@chmod($file,0777);
$fp=fopen($file,"w+");
```



```
fclose($fp);
}

?>
<html>
<head>
<title>
<?=$admin_title?>
</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<style type="text/css">
<!--
body {
    background-color: #DCE1E6;
    margin-left: 0px;
    margin-top: 0px;
    margin-right: 0px;
    margin-bottom: 0px;
}
-->
</style>
<link href="../s.css" rel="stylesheet" type="text/css">
<script>
function del(a)
{
if(a==1)
{
alert("Cannot Delete the Site Home Page Sorry!..");
}
else
{
if(confirm("Site Page Deletion Confirmation!\n\nAre you sure that you want to
delete this page and its contents?"))
{
window.navigate("<?=$PHP_SELF?>?act=d&id="+a);
}
else
{
return;
```

```
}
}
</script>
</head>
<?
$act=$_GET[act];
$id=$_GET[id];

if($act=="d")
{
if($id>1)
{
if($_SESSION[des]=="Administrator") {
mysql_query("delete from cmsv2_pages where id='$id'");
@unlink("pages/$id.txt");
@unlink("backups/$id.txt");
}
}
$msg="Page Deleted";
}

if(isset($_POST[Add]))
{
if($_SESSION[des]=="Administrator") {
$cnt=0;
if($_POST[lname]=="")
{
$cnt++;
$ln=1;
}
if($_POST[info]=="")
{
$cnt++;
$te=1;
}
$nn=mysql_num_rows(mysql_query("select * from cmsv2_pages where
lname='$_POST[lname]'"));
if($nn>0)
{
```

```

$ln=2;
}
$nn=mysql_num_rows(mysql_query("select * from cmsv2_pages where
dispord='$_POST[dispord]' and tb='$_POST[tb]"));
if($nn>0)
{
$cnt++;
$dpr=1;
}
if($cnt==0)
{
mysql_query("insert into cmsv2_pages(lname,tb,dispord)
values('$_POST[lname]','$_POST[tb]','$_POST[dispord]"));
$mn=mysql_insert_id();
Write("pages/$mn.txt",$_POST[info]);
if($dir=@opendir("pages")) {
while(($file=readdir($dir))!= false) {
if($file!="." && $file!="..")
{
Write("backups/$file","Tested");
copy("pages/$file","backups/$file");
$s= Read1("backups/$file");
Write1("backups/$file","$s"." uday");

}
}
closedir($dir);
}
$msg="Page Added";
$act="Added";
}
else
{
$act="a";
}
}
}
}
}

```

```

if(isset($_POST[Update]))

```

```

/* $cnt=0;
if($_POST[lname]=="")
{
$cnt++;
$ln=1;
} */
$cnt=0;
if($_POST[lname]=="")
{
$cnt++;
$ln=1;
}
if($_POST[info]=="")
{
$cnt++;
$te=1;
}
$nn=mysql_num_rows(mysql_query("select * from cmsv2_pages where
id!=$_POST[dd] and lname='$_POST[lname]'"));
if($nn>0)
{
$cnt++;
$ln=2;
}
$nn=mysql_num_rows(mysql_query("select * from cmsv2_pages where
id!=$_POST[dd] and dispord='$_POST[dispord]' and tb='$_POST[tb]'"));
if($nn>0)
{
$cnt++;
$dp=1;
}

if($cnt==0)
{
Write1("backups/$_POST[dd].txt",$_POST[bks]);
}

```

```

mysql_query("update cmsv2_pages set
lname='$_POST[lname]',tb='$_POST[tb]',dispord='$_POST[dispord]' where
id='$_POST[dd]");
$mn=mysql_insert_id();
Write("pages/$_POST[dd].txt",$_POST[info]);
$msg="Page Updated";
$act="Updated";
}
else
{
$act="e";
$id=$_POST[dd];
}
}

if($act=="e")
{
$rk=mysql_fetch_object(mysql_query("select * from cmsv2_pages where
id='$id'"));
}
?>
<body>
<table width="775" border="0" align="center" cellpadding="0" cellspacing="0">
<tr>
<td><? include("t.php");?></td>
</tr>
<tr>
<td height="1" bgcolor="#FFFFFF"></td>
</tr>
<tr>
<td bgcolor="#FFFFFF"><table width="100%" border="0" cellspacing="0"
cellpadding="0">
<tr valign="top">
<td width="150" class="rowtitbg"><?include("l.php");?></td>
<td><br>
<table width="95%" border="0" align="center" cellpadding="0"
cellspacing="0" class="black-text">
<tr>
<td valign="top"><table width="70%" border="0" cellpadding="2"

```

```

        <tr>
            <td class="rowtitbg">&nbsp;&nbsp;&nbsp;<strong>Site Pages
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<span class="head-text">
                <?
                    if($msg!="")
                    echo "[ $msg ]";
                    ?>

            </span></strong></td>
        </tr>
    </table></td>
</tr>
<tr>
    <td valign="top" class="brdr"><table width="100%" border="0"
cellpadding="2" cellspacing="2" class="black-text">
        <tr>
            <td valign="top"><table width="100%" border="0" cellpadding="2"
cellspacing="2" class="black-text">
                <tr align="center" bgcolor="#cccccc">
                    <td width="29%" bgcolor="#cccccc"><strong>Page Name
</strong></td>
                    <td width="71%"><strong>Actions</strong></td>
                </tr>
            <?
                $cnt=0;
                $rs=mysql_query("select * from cmsv2_pages order by id");
                while($rw=mysql_fetch_object($rs))
                {
                    $cnt++;
                    if($cnt%2==0)
                    $clr="#ebebeb";
                    else
                    $clr="#f4f4f4";
                    ?>
                    <tr bgcolor="<?=$clr?>">
                        <td><li><strong><?=$rw->lname?></strong></td>
                        <td align="center">[ <a
href="<?=$PHP_SELF?>?act=e&id=<?=$rw->id?>">Edit</a> ] <?
if($_SESSION[des]=="Administrator") {?>[ <a href="Javascript:del(<?=$rw-
->id?>);">Delete</a> ] <? }?> </td>

```

```

<?
}
?>
        <tr align="center" bgcolor="#cccccc">
            <? if($_SESSION[des]=="Administrator")
{?>
                <td colspan="2">[ <a href="<?=$PHP_SELF?>?act=a">Add
New Page</a> ] </td>
                    <? } ?>
            </tr>
        </table></td>
    </tr>
</table></td>
</tr>
</table>
<br>
<?
if($act=="a")
{
if($_SESSION[des]=="Administrator") {
?>
    <br>
    <table width="95%" border="0" align="center" cellpadding="0"
cellspacing="0" class="black-text">
        <tr>
            <td valign="top"><table width="50%" border="0" cellpadding="2"
cellspacing="2" class="white-text">
                <tr>
                    <td class="rowtitbg">&nbsp;<strong>Add New Page
</strong></td>
                </tr>
            </table></td>
        </tr>
    </table>
    <tr>
        <td valign="top" class="brdr"><table width="100%" border="0"
cellpadding="2" cellspacing="2" class="black-text">
            <form name="f2" method="POST" action="<?=$PHP_SELF?>">
                <tr>
                    <td valign="top" class="black-bold"><table width="100%"

```

```

        <tr>
            <td width="35%" align="right">Link Name: </td>
            <td><input name="lname" type="text" id="lname"
value="<?=$_POST[lname]?" size="30"><span class="error-text">
<?
if($ln==1)
echo "[ Blank ]";
if($ln==2)
echo "[ Already Exist ]";
?>
</span></td>
        </tr>
        <tr>
            <td align="right">Display Order : </td>
            <td><select name="dispord" id="dispord">
<?
for($i=1;$i<21;$i++)
{
?>
<option value="<?=$i?" <?if($_POST[dispord]==$i){echo
"SELECTED";}?>><?=$i?"</option>
<?
}
?>
            </select>
            <span class="error-text">
<?
if($dp==1)
echo "[ Already Exist ]";
?>
            </span></td>
        </tr>
    </table></td>
</tr>
<tr>
    <td valign="top" class="black-bold">Page Contents :
    <hr></td>
</tr>
<tr>

```



```

        <td align="center" valign="top"><textarea name="info" cols="70"
rows="20" id="info"><?=$_POST[info]?>
</textarea>
        <span class="error-text">
<?
if($te==1)
echo "[ Blank ]";
?>
</span>          <Script language="javascript1.2" defer>
editor_generate('info');
</Script></td>
        </tr>
        <tr>
        <td align="center" valign="top">Page Alignment :
        <input name="tb" type="radio" value="1" checked>
        Top navigation
        <input name="tb" type="radio" value="2">
        Bottom Navigation
        <input name="tb" type="radio" value="3">
        Both Sides </td>
        </tr>
        <tr>
        <td align="center" valign="top"><input name="Add"
type="submit" id="Add" value="Add it"></td>
        </tr>
        </form>
        </table></td>
        </tr>
        </table>
        <?
}
}
?>
        <br>
        <?
if($act=="e")
{
?>
        <table width="95%" border="0" align="center" cellpadding="0"

```



```

        <?
    }
    ?>
        </select></td>
    </tr>
</table></td>
</tr>
<tr>
    <td valign="top" class="black-bold">Page Contents :
        <hr></td>
</tr>
<tr>
    <td align="center" valign="top">
        <input name="bks" type="hidden" id="bks"
value="<?=Read("pages/$rk->id.txt");?>">
        <textarea name="info" cols="70" rows="20"
id="info"><?=Read("pages/$rk->id.txt");?> </textarea>
        <input name="dd" type="hidden" id="dd" value="<?=$rk->id?>">
        <span class="error-text">
        <?
if($te==1)
echo "[ Blank ]";
?>
        </span>
        <Script language="javascript1.2" defer>
editor_generate('info');
        </Script></td>
    </tr>
<tr>
    <td align="center" valign="top">Page Alignment :
        <input name="tb" type="radio" value="1" checked>
        Top navigation
        <input name="tb" type="radio" value="2" <? if($rk->tb==2){echo
"CHECKED";}?>>
        Bottom Navigation
        <input name="tb" type="radio" value="3" <? if($rk->tb==3){echo
"CHECKED";}?>>
        Both Sides
</td>
</tr>

```

```

        <td align="center" valign="top"><input name="Update"
type="submit" id="Update" value="Update Page"></td>
        </tr>
        </form>
        </table></td>
        </tr>
        </table>
        <?
}
?> </td>
        </tr>
        </table></td>
</tr>
<tr>
        <td height="1" bgcolor="#FFFFFF"></td>
</tr>
<tr>
        <td><?include("b.php");?></td>
</tr>
</table>
</body>
</html>

```

Database

```

# MySQL-Front Dump 2.5
#
# Host: localhost  Database: cms_v2
# -----
# Server version 3.23.33

#
# Table structure for table 'cmsv2_admin'
#
DROP TABLE IF EXISTS cmsv2_admin;
CREATE TABLE cmsv2_admin (
  id int(5) NOT NULL auto_increment,
  uname longtext,

```

```
designation varchar(20) default NULL,  
IPaddress varchar(255) default NULL,  
invalid tinyint(3) unsigned default NULL,  
PRIMARY KEY (id),  
KEY id(id)  
) TYPE=MyISAM;
```

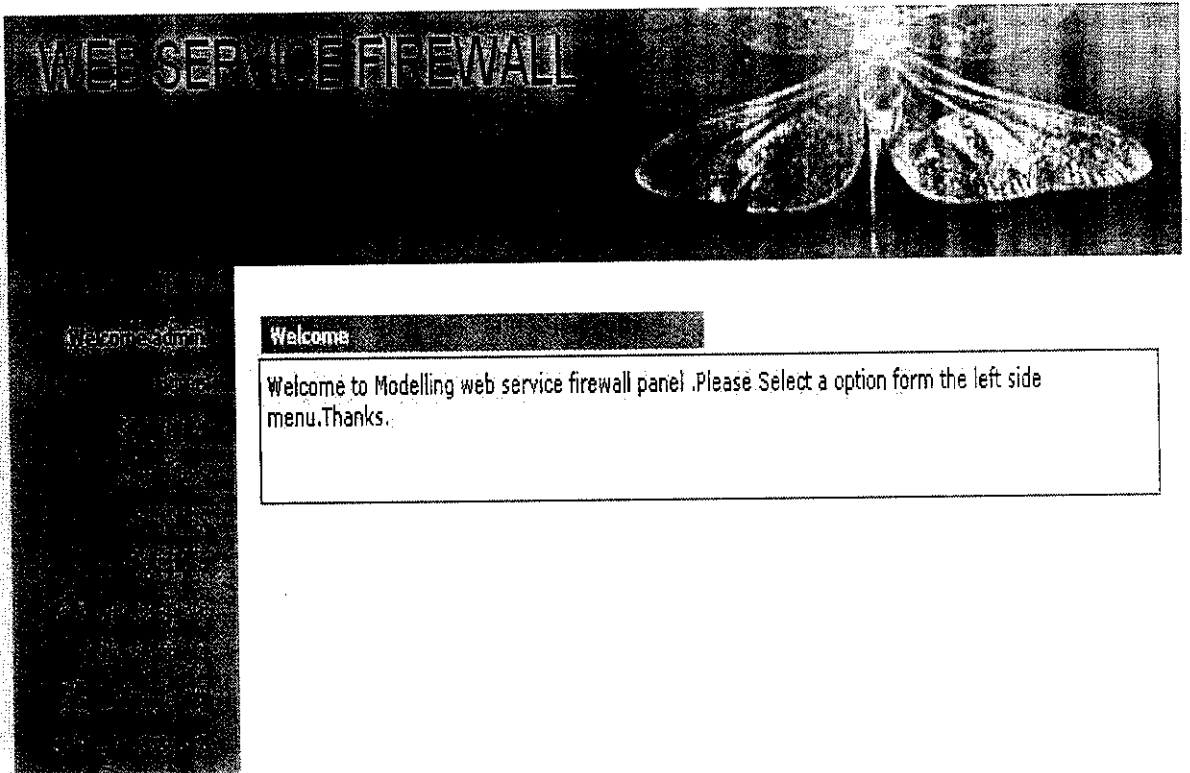
```
#  
# Dumping data for table 'cmsv2_admin'  
#  
INSERT INTO cmsv2_admin (id, uname, pwd, designation, IPaddress, invalid)  
VALUES("1", "admin", "admin", "Administrator", "127.0.0.1", "5");  
INSERT INTO cmsv2_admin (id, uname, pwd, designation, IPaddress, invalid)  
VALUES("2", "aruljoseph00@gmail.com", "Paypal Address", "Employee",  
"127.0.0.1", NULL);  
INSERT INTO cmsv2_admin (id, uname, pwd, designation, IPaddress, invalid)  
VALUES("3", "KUMARAGURU COLLEGE OF TECHNOLOGY", "Site Title",  
"Employee", "127.0.0.1", NULL);  
INSERT INTO cmsv2_admin (id, uname, pwd, designation, IPaddress, invalid)  
VALUES("4", "Web Service CMS Administration Panel", "Admin Title",  
"Employee", "127.0.0.1", NULL);  
INSERT INTO cmsv2_admin (id, uname, pwd, designation, IPaddress, invalid)  
VALUES("5", "ram", "ram", "Employee", "127.0.0.1", NULL);  
INSERT INTO cmsv2_admin (id, uname, pwd, designation, IPaddress, invalid)  
VALUES("6", "anu", "anu", "Employee", "127.0.0.1", NULL);  
INSERT INTO cmsv2_admin (id, uname, pwd, designation, IPaddress, invalid)  
VALUES("7", "ramya", "ramya", "Employee", "127.0.0.1", NULL);
```

```
#  
# Table structure for table 'cmsv2_sinfo'  
#  
DROP TABLE IF EXISTS cmsv2_sinfo;  
CREATE TABLE cmsv2_sinfo (  
  id int(5) NOT NULL auto_increment,  
  aboutus longtext,  
  services longtext,  
  newarrivals longtext,  
  contactus longtext,  
  policies longtext
```

```
corp_overview longtext,  
terms_conditions longtext,  
faq longtext,  
business_needs longtext,  
features longtext,  
PRIMARY KEY (id),  
UNIQUE KEY id(id),  
KEY id_2(id)  
) TYPE=MyISAM;
```

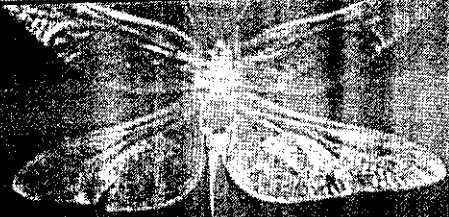
RESULTS

HOME PAGE



ADMIN ACCESS

WEB SERVICE FIREWALL



Welcome Admin

Site Pages

Page Name	Actions
• Home1	[Edit] [Delete]
• Privacy Policy	[Edit] [Delete]
• FAQ	[Edit] [Delete]
• About Us	[Edit] [Delete]
• Contact Us	[Edit] [Delete]
• Services	[Edit] [Delete]
• Products	[Edit] [Delete]
• test	[Edit] [Delete]

ADD USER

Welcome admin

Home

Site Map

Username :

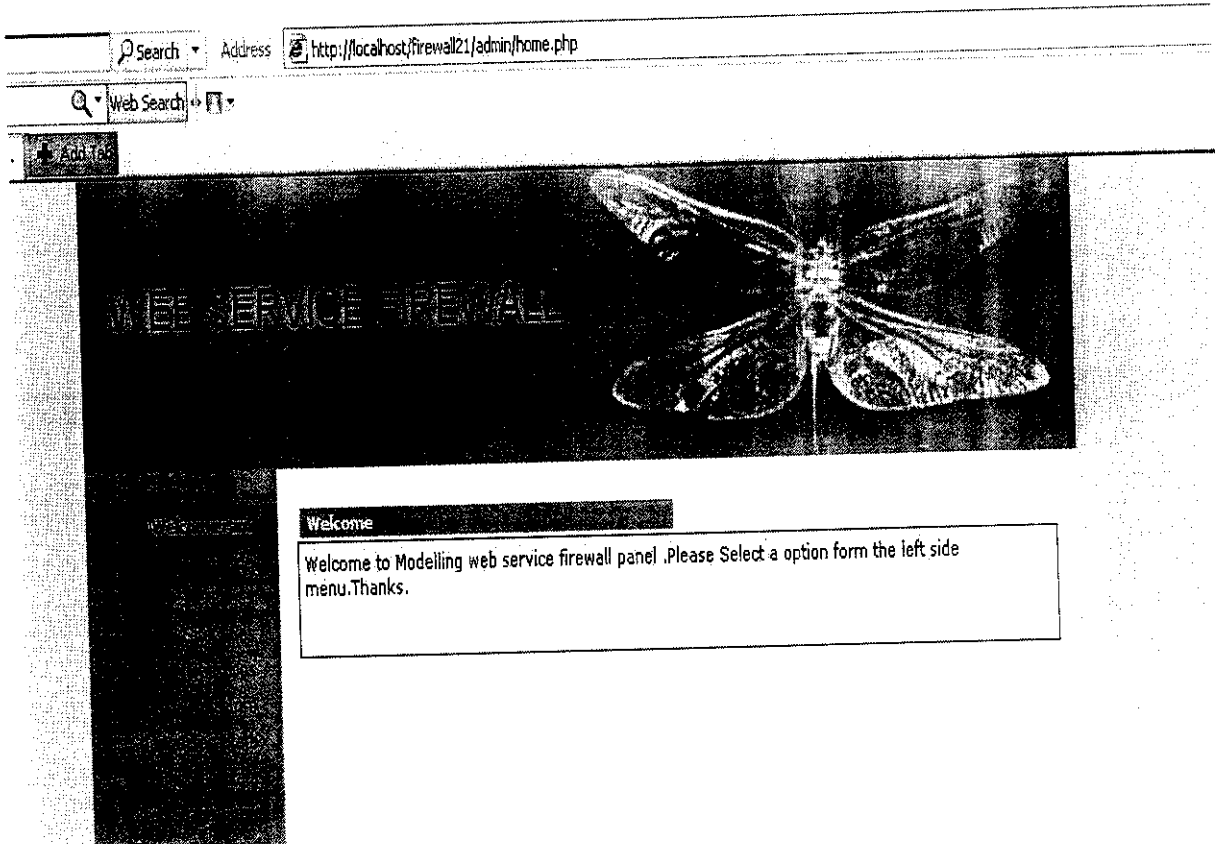
Password :

Confirm Password :

IP address :

USername	Password	IP Address	Action	
a	a	127.0.0.1	[Delete]	
b	ba	127.0.0.1	[Delete]	
dd	12	127.0.0.1	[Delete]	
ram	ram	127.0.0.1	[Delete]	
anu	anu	127.0.0.1	[Delete]	
ramya	ramya	127.0.0.1	[Delete]	

USER ENTRANCE

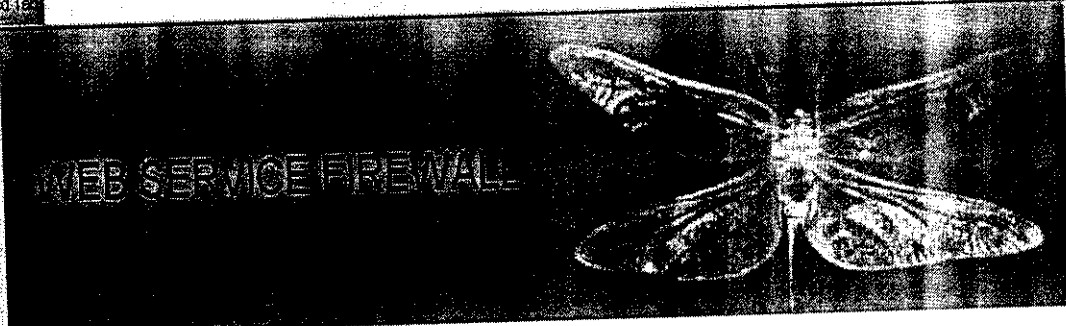


USER POLICY

Search Address <http://localhost/firewall21/admin/pg.php>

Web Search FT

+ Add Tab



WEB SERVICE FIREWALL


Navigation

Site Pages

Page Name	Actions
• Home1	[Edit]
• Privacy Policy	[Edit]
• FAQ	[Edit]
• About Us	[Edit]
• Contact Us	[Edit]
• Services	[Edit]

SESSION OUT



Address:  http://localhost/firewall21/admin/index.php?act=lout

You Have Successfully Logged Out.Thanks.

Administration Login Panel	
User Name	<input type="text"/>
Password	<input type="text"/>
<input type="button" value="Login"/>	<input type="button" value="Cancel"/>

DATA BASE

phpMyAdmin

Database: **cms_v2 (7)**

- cms_v2 (7)
 - cmsv2/rollback
 - cmsv2/sonar
 - cmsv2/images
 - cmsv2/pages
 - cmsv2/server
 - cmsv2/site
 - cmsv2/templates

Show: row(s) starting from record #

in mode and repeat headers after cells

Sort by key:

+ Options

	id	uname	pwd	designation	IPaddress	invalid
<input type="checkbox"/>	1	admin	admin	Administrator	127.0.0.1	0
<input type="checkbox"/>	8	a	a	Administrator	127.0.0.1	NULL
<input type="checkbox"/>	9	b	ba	Administrator	127.0.0.1	NULL
<input type="checkbox"/>	10	ad	12	Administrator	127.0.0.1	NULL
<input type="checkbox"/>	5	ram	ram	Employee	127.0.0.1	0
<input type="checkbox"/>	6	anu	anu	Employee	127.0.0.1	0
<input type="checkbox"/>	7	ramya	ramya	Employee	127.0.0.1	4

↑ Check All / Uncheck All With selected:

Show: row(s) starting from record #

in mode and repeat headers after cells

Query results operations

Print view Print view (with full texts) Export CREATE VIEW

QUERY STATISTICS

Query statistics: Since its startup, 159 queries have been sent to the server.

Total	per hour	per minute	per second
159	824.78	13.75	0.00 x

Query type	per hour	%	Query type	per hour	%
admin commands	0	0.00%	restore table	0	0.00%
assign to keycache	0	0.00%	revoke	0	0.00%
alter db	0	0.00%	revoke all	0	0.00%
alter db upgrade	0	0.00%	rollback	0	0.00%
alter event	0	0.00%	rollback to savepoint	0	0.00%
alter function	0	0.00%	savepoint	0	0.00%
alter procedure	0	0.00%	select	74	383.86%
alter server	0	0.00%	set option	25	129.68%
alter table	0	0.00%	show authors	0	0.00%
alter tablespace	0	0.00%	show binlog events	0	0.00%
analyze	0	0.00%	show binlogs	4	20.75%
backup table	0	0.00%	show charsets	1	5.19%
begin	0	0.00%	show collations	1	5.19%
binlog	0	0.00%	show column types	0	0.00%
call procedure	0	0.00%	show contributors	0	0.00%
change db	17	88.18%	show create db	0	0.00%
change master	0	0.00%	show create event	0	0.00%

phpMyAdmin



- abc (7)
- cms_v2 (7)
- information_schema (20)
- mysql (23)

Please select a database

Command	Count	Min	Max	Command	Count	Min	Max
create user	0	0.00	0.00%	show grants	1	5.19	0.73%
create view	0	0.00	0.00%	show keys	0	0.00	0.00%
delete	0	0.00	0.00%	show master status	1	5.19	0.73%
delete multi	0	0.00	0.00%	show new master	0	0.00	0.00%
do	0	0.00	0.00%	show open tables	0	0.00	0.00%
drop db	0	0.00	0.00%	show plugins	6	31.12	4.38%
drop event	0	0.00	0.00%	show privileges	0	0.00	0.00%
drop function	0	0.00	0.00%	show procedure status	0	0.00	0.00%
drop index	0	0.00	0.00%	show processlist	0	0.00	0.00%
drop procedure	0	0.00	0.00%	show profile	0	0.00	0.00%
drop server	0	0.00	0.00%	show profiles	0	0.00	0.00%
drop table	0	0.00	0.00%	show slave hosts	0	0.00	0.00%
drop trigger	0	0.00	0.00%	show slave status	1	5.19	0.73%
drop user	0	0.00	0.00%	show status	2	10.37	1.46%
drop view	0	0.00	0.00%	show storage engines	0	0.00	0.00%
empty query	0	0.00	0.00%	show table status	0	0.00	0.00%
flush	0	0.00	0.00%	show tables	4	20.75	2.92%
grant	0	0.00	0.00%	show triggers	0	0.00	0.00%
ha close	0	0.00	0.00%	show variables	1	5.19	0.73%
ha open	0	0.00	0.00%	show warnings	0	0.00	0.00%
ha read	0	0.00	0.00%	slave start	0	0.00	0.00%
help	0	0.00	0.00%	slave stop	0	0.00	0.00%
insert	0	0.00	0.00%	stmt close	0	0.00	0.00%

SERVER TRAFFIC

Server: localhost

[Databases](#)
[SQL](#)
[Status](#)
[Variables](#)
[Charsets](#)
[Engines](#)
[Privileges](#)
[Binary log](#)
[Processes](#)
[Export](#)
[Import](#)

Runtime Information

Refresh Reset

This MySQL server has been running for 0 days, 0 hours, 11 minutes and 34 seconds. It started up on Apr 07, 2010 at 04:00 PM.

This MySQL server works as master and in replication process. For further information about replication status on the server, please visit the replication section.

[SQL query](#)
[InnoDB](#)
[SSL Handler](#)
[Query cache](#)
[Threads](#)
[Binary log](#)
[Temporary data](#)
[Delayed inserts](#)
[Key cache](#)
[Joins](#)
[Replication](#)
[Sorting Tables](#)
[Transaction coordinator](#)

Server traffic: These tables show the network traffic statistics of this MySQL server since its startup.

	Traffic	per hour	Connections	per hour	%
Received	5,695 B	30 K1B	max. concurrent connections	3	
Sent	69 K1B	358 K1B	Failed attempts	2	10.37 91.09%
Total	75 K1B	388 K1B	Aborted	0	0.00 0.00%
			Total	22	114.12 100.00%

Query statistics: Since its startup, 159 queries have been sent to the server.

References

- [1] Zobeideh Aliannezhadi and Mohammad Abdollahi Azgomi, "Modeling and Analysis of a Web Service Firewall Using Coloured Petri Nets", 2008 IEEE Asia-Pacific Services Computing Conference
- [2] Web Services and Security measures of Web Services from www.acunetix.com/websitesecurity/web-services-wp.htm
- [3] SQL injection and its prevention from www.acunetix.com/websitesecurity/sql-injection.htm
- [4] "CPN Tools," CPN Group, University of Aarhus, Denmark, URL: <http://wiki.daimi.au.dk/cpntools>.
- [5] Jensen, K, Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use, Vol. 1: Basic Concepts, EATCS Monographs in Theoretical Computer Science, Springer. 1992.
- [6] K. Jensen, "An Introduction to the Practical Use of Coloured Petri Nets", *Lecture Notes in Computer Science*, No. 1492, Springer, 1998, pp. 237-292.