International Journal of Computer Theory and Engineering, Vol.3, No.1, February, 2011
1793-8201

Role Based Authentication Schemes for Security Automation

Dharmendra Choukse[1] & Umesh Kumar Singh[2]

Abstract—**Academy Automation implies to the various different computing hardware and software that can be used to digitally create, manipulate, collect, store, and relay Academy information needed for accomplishing basic Operation like admissions and registration to finance, student and faculty interaction, online library, medical and business development. Raw data storage, electronic transfer, and the management of electronic business information comprise the basic activities of an Academy automation system. The main aim of this work was to design and implement a Role Based Authentication (RBA) System wherein each user has certain roles allotted to him/her which defines the user's limits and capabilities of making changes, accessing various areas of the software and transferring/allotting these roles recursively. Strict security measures had kept in mind while designing such a system and proper encryption and decryption techniques are used at both ends to prevent any possibility of any third party attacks. Further, various new age authentication techniques like OpenID and WindowsCardSpace are surveyed and discussed to serve as a foundation for future work in this area.**

IndexTerms—**RBA,Encryption/Decryption,OpenID,WindowsCard -Space**

I. Introduction

Starting in the 1970s, computer systems featured multiple applications and served multiple users, leading to heightened awareness of data security issues. System administrators and software developers alike focused on different kinds of access control to ensure that only authorized users were given access to certain data or resources. One kind of access control that emerged is role-based access control (RBAC). A role is chiefly a semantic construct forming the basis of access control policy. With RBAC, system administrators create roles according to the job functions performed in a company or organization, grant permissions (access authorization) to those roles, and then assign users to the roles on the basis of their specific job responsibilities and qualifications "Role-based access control terms and concepts ". A role can represent specific task competency, such as that of a physician or a pharmacist. A role can embody the authority and responsibility of, say, a project supervisor. Authority and responsibility are distinct from competency. A person may be competent to manage several departments but have the responsibility for only the department actually managed. Roles can also reflect specific duty assignments rotated through multiple users for example, a duty physician or a shift manager. RBAC models and implementations should conveniently accommodate all these manifestations of the role concept. Roles define both the specific individuals allowed to access resources and the extent to which resources are accessed. For example, an operator role might access all computer resources but not change access permissions; a security-officer role might change permissions but have no access to resources; and an auditor role might access only audit trails. Roles are used for system administration in such network operating systems as Novell's NetWare and Microsoft's Windows NT.

In this article present a comprehensive approach to RBAC on the Web. We identify the user-pull and server-pull architectures and analyze their utility. To support these architectures on the Web, for relatively mature technologies and extend them for secure RBAC on the Web. In order to do so, to make use of standard technologies in use on the Web: cookies [*Kristol and Montulli 1999; Moore and Freed* 1999], X.509 [*ITU-T Recommendation X.509* 1993*;* 1997*; Housley et al.* 1998],SSL (Secure Socket Layer [*Wagner and Schneier* 1996*; Dierks and Allen* 1999]), and LDAP (Lightweight Directory Access Protocol [*Howes et al.* 1999]). The Lightweight Directory Access Protocol (LDAP) directory service already available for the purpose of webmail authentication of IPS Academy, Indore users has been used to do the basic Authentication. The client can request the application server for any web application which will ask for the user credentials which will be verified in the LDAP server through an ASP.Net [17] Module. On successful verification, the authorization module will contact the user role database and fetch the roles for that user. In case of return of multiple roles, user will be given the authorization of all the roles. The access to the application will be on the basis of privilege of the role of that particular user. The role database is implementing in Microsoft SQL server [24] database. On successful authentication, the Authentication and authorization module which has been developed for this purpose is called and the role for the user is retrieved. Privileges are granted to roles which in turn roles are granted to users.

The overall database server and application server is considered for possible attacks. The proposed scheme is given in figure 2. The database server and the authentication server are in a private network and separated from the user network by a firewall. These servers can be accessed only through application server, i.e. through the authentication and authorization module. Application server has an interface in the private network but can avail only the specific service

1. Institute of Engineering & Sciences, IPS Academy, Indore
2. Institute of Computer Science, Vikram University, Ujjain
dharmendrachoukse@gmail.com, Umeshsingh@rediffmail.com

which has been explicitly allowed in the firewall. Application server has another interface which is part of user network with a firewall to restrict the clients only to the desired service.

The information flow security has been taken care by secure http. The Asp.Net Application server has the support for HTTPS which was configured to make sure that data passing to and from Application server is encrypted. From the Application Server, a digital certificate in SSL [23] (Secure Socket Layer) has been generated. This needs to be installed on the client machine for server identity verification. Similarly client certificate can also be generated from the Asp.Net which can be used in the client which will update sensitive data. Such operation will be denied without client certificate.

II. Literature Review

A large number of research papers are published in the area of Role Based Authentication. In [5] Raymond emphasized the purpose of Role Based Authentication. Authorization architecture for authorizing access to resource objects in an object-oriented programming environment is discussed in this paper. In one distributed environment, the permission model of JAAS (Java Authentication and Authorization Service) is replaced or enhanced with role-based access control. Thus, users and other subjects (e.g., pieces of code) are assigned membership in one or more roles, and appropriate permissions or privileges to access resource objects are granted to those roles. Permissions may also be granted directly to users. Roles may be designed to group users having similar functions, duties or similar requirements for accessing the resources. Roles may be arranged hierarchically, so that users explicitly assigned to one role may indirectly be assigned to one or more other roles (i.e., descendants of the first role). A realm or domain may be defined as a namespace, in which one or more role hierarchies are established.

Robert et al in [6] discussed about Methods, systems, and computer program products are disclosed for protecting the security of resources in distributed computing environments. The disclosed techniques improve administration and enforcement of security policies. Allowed actions on resources, also called permissions, (such as invocations of particular methods, read or write access of a particular row or perhaps a particular column in a database table, and so forth) are grouped, and each group of permissions is associated with a role name. A particular action on a particular resource may be specified in more than one group, and therefore may be associated with more than one role. Each role is administered as a security object. Users and/or user groups may be associated with one or more roles. At run-time, access to a resource is protected by determining whether the invoking user has been associated with (granted) at least one of the roles required for this type of access on this resource.

In [7] Dixit et al discussed about an actor is associated with a role, a policy type is associated with the role, and a role scope is associated with the role. One or more values are received for one or more corresponding context parameters associated with the actor. A request for access to a resource is received from the actor. A policy instance is determined based on the policy type and the one or more values for the one or more corresponding context parameters associated with the actor. One or more actor-role scope values are determined based on the role scope and the one or more values for the one or more corresponding context parameters associated with the actor. A response to the request is determined based on the policy instance and the actor-role scope values.

Bindiganavale and Ouyang, in [8] presents the most challenging problems in managing large web-applications is the complexity of security administration and user-profile management. Role Based Access Control (RBAC) has become the predominant model for advanced access control because it reduces the complexity and cost of administration. Under RBAC, security administration is greatly simplified by using roles, hierarchies and privileges, and user management is uncomplicated by using LDAP API specification within the J2EE application. System administrators create roles according to the job functions performed in an organization, grant permissions to those roles, and then assign users to the roles on the basis of their specific job Responsibilities and qualifications.

A wireless networks proliferate, web browsers operate in an increasingly hostile network environment. The HTTPS protocol has the potential to protect web users from network attackers, but real-world deployments must cope with misconfigured servers, causing imperfect web sites and users to compromise browsing sessions inadvertently. Force HTTPS is a simple browser security mechanism that web sites or users can use to opt in to stricter error processing, improving the security of HTTPS by preventing network attacks that leverage the browser's lax error processing. By augmenting the browser with a database of custom URL rewrite rules, Force HTTPS allows sophisticated users to transparently retrofit security onto some insecure sites that support HTTPS. We provide a prototype implementation of Force HTTPS as a Firefox browser extension [9].

III. Observations And Problem Description

The whole Collage Academy automation consists of many sections viz. Student Affairs, Academic Section, Research and Development, Training and Placement, Finance and Accounts etc. Different individuals in IPS Academy, Indore should be given access to different aspects of the systems based on their clearance level. For e.g. the Assistant Registrar of Student Affairs should have full access to all the options of Student Affairs database but not that of the Academic Section database. However, provisions have to be made so that he/she is able to perform some student affairs related queries to the student affairs database. Similarly, a student must have read-only access to his/her information in the official records and modifying capabilities some of his/her details in the training and placement section database. This calls for a role-based approach to access the databases. Each person has a certain role attached to it. This role corresponds to the areas of the work his login account can access. If a violation occurs, the user is immediately logged out.

In this work the design and implementation of the Role

Based Authentication Schemes for Security Automation is described, developed at the IPS Academy, Indore as an ASP.NET [2005] web application in C# server side code, HTML, and JavaScript for use on the Internet. The purpose work to deploy a cost-effective, web-based system that significantly extends the capabilities, flexibility, benefits, and confidentiality of paper-based rating methods while incorporating the ease of use of existing online surveys and polling programs.

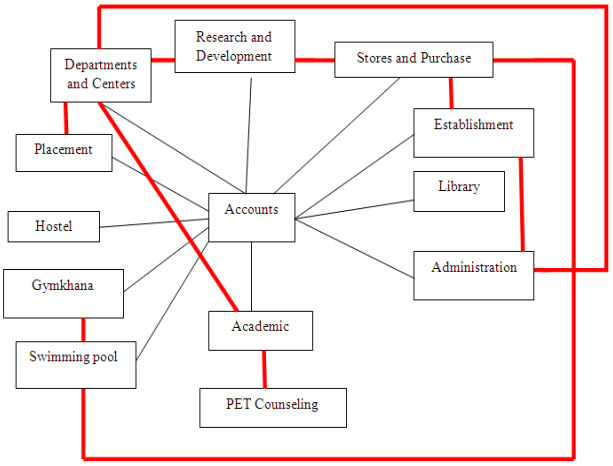

Figure 1: Basic Architecture of Academy

A. *Problem Issues And Challenges*

The Following Problems are as Follows:-

1) The information line must be completely secured.
2) Proper Encryption must be used for storing the Password for the User.
3) The authorization token which is stored on the client side has to be encrypted so that the client cannot modify his authorization clearance level.
4) Each userid-role mapping should have an expiry date beyond which it will be invalid.
5) Role Scoping: Local and Global Roles
6) In each role, we have to have an owner. Normally the role will map to the user id of the owner. The owner can change the mapping and can specify the time period of this change. The newly mapped user is not the owner and so cannot change the ownership, but maybe allowed to map again. For example, HODCSE is the role and the owner's user id is"Ram". Normally, HODCSE maps to Ram. When Prof. Ram goes on leave, he fills up some form electronically and this triggers (among other things) a role change of HODCSE to the user he designates, say Prof.Shayam. Now"Ram" is going on leave till 4/7/2010, so the changed mapping is till 4/7/2010 (to"pshayam"; specified by"Ram" in the form he filled up). Now due to an emergency,

"pshayam" had to leave station on 4/7/2010, making Prof manoj the Head. Since" pshayam" is not the owner, he cannot change the validity date beyond 4/7/2010 and"Ashish" takes over the HODCSE role till 4/7/2010. On 5/7/2010 (or the next query of the role), the role remaps to"Ram". Other cases (like"Ram" having to overstay beyond 4/7) can be handled by the administrator.

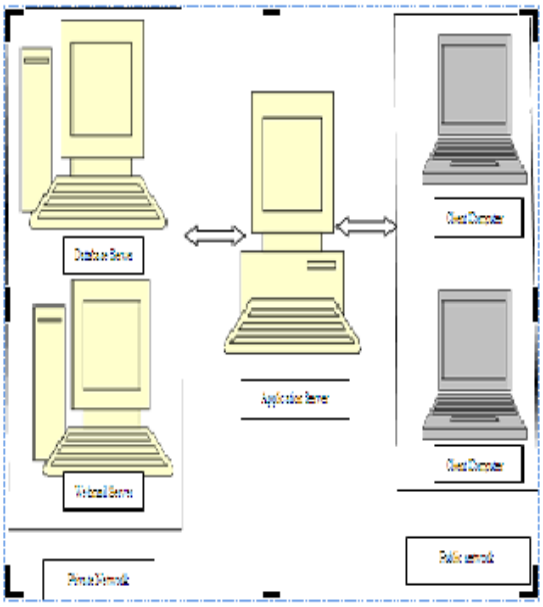

Figure 2: System and Server Security

IV. METHODOLOGIES

1) We have 2 sets of Roles:

Global Roles: These refer to the roles which are common to the entire applications viz. root, Director. Their Role IDs are of single digit: 0, 1, and 2 etc.

Local Roles: These are roles which are specific to a module. For E.g. for Student Affairs, the roles of Assistant Registrar, Academy in charge. Their IDs are of the Form: 10, 11, 12 ... 110 etc. where first digit identifies the application to which all of them are common.

2) There is a Global role to role_id mapping table.
3) Also there is a local mapping table for each section. Insertion/modification or deletion of any entry in the local table generates a Microsoft SQL trigger for its 'encoded' entry addition in the global table.

TABLE 1: VARIOUS ROLES AND THEIR IDS

Role	Role ID
Administrator	0
Student	1
Faculty	2
Assistant Registrar (Student Affairs)	10
Assistant Registrar (Academic)	20
Assistant Registrar (RND)	30
Assistant Registrar (TNP)	40
Assistant Registrar (Finance)	50
Registrar	3
Director	4
Head of Departments	5

TABLE 2: USER NAME ID RELATION

User_name	User_id
root	1
dharmendra	2
try	3

TABLE 3: USER ROLE RELATION

s_no	user_id	role_id	valid_from	valid_upto
1	1	12	2009-01-01	2009-01-02
2	1	13	2009-01-01	2009-05-06
3	2	12	2008-01-01	2009-01-01

A web interface which is accessed by any member and is used to assign his role to any other member for a specified period. The role validity period of the other person cannot exceed the validity period of the assigner. So, whenever a role has to be transferred, an entry is made in the user role relation table corresponding to the user ID of the assigned person and it is made sure that the validity period of the assigned is less than the validity period of assigner from the same user role relation table

A. Database Table Structure

We will have a common login page for all the sections of the Academy Automation. The looks up table of the corresponding IDs are shown in table 1, 2 & 3.

B. Asp.Net Authentication

Now, each webpage has a small Asp.Net code which expects to read the system cookie of a specified number of roles before displaying the page. If unsuccessful, this page re-directs the user to the logout page and deletes the session cookies else the corresponding web page is displayed. The authentication module which is imported at the beginning of every Asp.net page looks like:

We must create a Forms Authentication login system the supports roles. The process of creating the authentication ticket and the cookie has to be stored under the right name - the name matching the configured name for Forms Authentication root Web.config file. If these names don't match, ASP.NET won't find the Authentication Ticket for the Web Application and will force a redirect to the login page.

```
<%@ Page Language="C#" %>

<%@ Import Namespace="System.Data" %>

<%@ Import Namespace="System.Data.SqlClient" %>
<html>
 <head>
  <title>Login</title>
 </head>
 <script runat="server">
 // If we using code-behind, make sure that change "private" to
 // "protected" since the .aspx page inherits from the .aspx.cs
 // file's class
 private void btnLogin_Click(Object sender, EventArgs e)
    {
  // Initialize FormsAuthentication, for what it's worth
  FormsAuthentication.Initialize();
      // Create our connection and command objects
  SqlConnection conn =
   new SqlConnection("Data Source=localhost;Initial Catalog=web;");
  SqlCommand cmd = conn.CreateCommand();
  cmd.CommandText = "SELECT roles FROM web WHERE username=@username " +
   "AND password=@password";
     // Fill our parameters
  cmd.Parameters.Add("@username", SqlDbType.NVarChar, 64).Value = Username;
  cmd.Parameters.Add("@password", SqlDbType.NVarChar, 128).Value =
   FormsAuthentication.HashPasswordForStoringInConfigFile(Password);
     // Execute the command
  SqlDataReader reader = cmd.ExecuteReader();
  if (reader.Read())
     {
   // Create a new ticket used for authentication
  FormsAuthenticationTicket ticket = new FormsAuthenticationTicket(
   1, // Ticket version
   Username, // Username associated with ticket
   DateTime.Now, // Date/time issued
   DateTime.Now.AddMinutes(30), // Date/time to expire
   true, // "true" for a persistent user cookie
   reader.GetString(0), // User-data, in this case the roles
   FormsAuthentication.FormsCookiePath); // Path cookie valid for
     // Hash the cookie for transport
  string hash = FormsAuthentication.Encrypt(ticket);
  HttpCookie cookie = new HttpCookie(
   FormsAuthentication.FormsCookieName, // Name of auth cookie
   hash); // Hashed ticket
   // Add the cookie to the list for outgoing response
  Response.Cookies.Add(cookie);
     // Redirect to requested URL, or homepage if no previous page requested
  string returnUrl = Request.QueryString["ReturnUrl"];
  if (returnUrl == null) returnUrl = "/";
     // Don't call FormsAuthentication.RedirectFromLoginPage since it could
   // replace the authentication ticket (cookie) we just added
  Response.Redirect(returnUrl);
     }
  else
     {
   // Never tell the user if just the username is password is incorrect.
   // That just gives them a place to start, once they've found one or
   // the other is correct!
  ErrorLabel = "Username / password incorrect. Please try again.";
  ErrorLabel.Visible = true;
     }
    }
 </script>
 <body>
  <p>Username: <input id="Username" runat="server" type="text"/><br />
```

```
Password: <input id="Password" runat="server"
type="password"/><br />
<asp:Button id="btnLogin" runat="server"
OnClick="btnLogin_Click"
Text="Login"/>
<asp:Label id="ErrorLabel" runat="Server"
ForeColor="Red"
Visible="false"/></p>
</body>
</html>
```

We do one other thing with our passwords: we hash them. Hashing is a one-way algorithm that makes a unique array of characters. Even changing one letter from upper-case to lower-case in your password would generate a completely different hash. We'll store the passwords in the database as hashes, too, since this is safer. In a production environment, we'd also want to consider having a question and response challenge that a user could use to reset the password. Since a hash is one-way, we won't be able to retrieve the password. If a site is able to give our old password to us, I'd consider steering clear of them unless you were prompted for a client SSL certificate along the way for encrypting your passphrase and decrypting it for later use, though it should still be hashed.

C. Securing Directories With Role-Based Forms Authentication

In order to make the role Based authentication work for Forms Authentication, it is required to have a Web.config file in web application root. In authentication setup, this particular Web.config file must be in Web Application's document root. We can override the <authorization/> in Web.config files for sub-directories.

```
<configuration>
<system.web>
<authentication mode="Forms">
<forms name="MYWEBAPP.ASPXAUTH"
loginUrl="login.aspx"
protection="All"
path="/"/>
</authentication>
<authorization>
<allow users="*"/>
</authorization>
</system.web>
</configuration>
```

To control authorization (access by a particular user or group), we can either

1) add some more elements to the Web.config file from above, or
2) Create a separate Web.config file in the directory to be secure.

```
<configuration>
<system.web>
<authentication mode="Forms">
<forms name="MYWEBAPP.ASPXAUTH"
loginUrl="login.aspx"
protection="All"
path="/"/>
</authentication>
<authorization>
<allow users="*"/>
</authorization>
</system.web>
<location path="administrators">
<system.web>
<authorization>
<!-- Order and case are important below -->
<allow roles="Administrator"/>
<deny users="*"/>
</authorization>
</system.web>
</location>
<location path="users">
<system.web>
<authorization>
<!-- Order and case are important below -->
<allow roles="User"/>
<deny users="*"/>
</authorization>
</system.web>
</location>
</configuration>
```

D. Conditionally Showing Controls With Role-Based Forms Authentication

The IPrincipal interface, which the GenericPrincipal class we used above implements, has a method called "IsInRole()", which takes a string designating the role to check for. So, if we can only want to display content if the currently logged-on user is in the "Administrator" role.

```
<html>
<head>
<title>Welcome</title>
<script runat="server">
protected void Page_Load(Object sender, EventArgs e)
{
if (User.IsInRole("Administrator"))
AdminLink.Visible = true;
}
</script>
</head>
<body>
<h2>Welcome</h2>
<p>Welcome, anonymous user, to our web site.</p>
<asp:HyperLink id="AdminLink" runat="server"
Text="Administrators, click here."
NavigateUrl="administrators/"/>
</body>
</html>
```

V. CONCLUSIONS

The research problem and goal of the Academy Automation is to design a highly secure and efficient framework based on Service Oriented Architecture. Keeping in mind the policies of minimum data redundancy and efficient security, the work revolved around designing a plug in for secure role based authentication. Presently the authentication is based on the traditional userid-password

based approach, but as is suggested in this report, future work can be done to incorporate various new-age technologies such as OpenID, Windows CardSpace etc.

REFERENCES

[1] William Stallings, "Cryptography and Network Security Principles and Practices", 3rdEdition,Prentice Hall, 2003.
[2] Eric Cole, Ronald L. Krutz, James Conley, "Network Security Bible", 2nd Edition, Wiley Publication, 2005.
[3] Yih-Cheng Lee, Chi-Ming Ma and Shih-Chien Chou," A Service-Oriented Architecture for Design and Development of Middleware," Proceedings of the 12th Asia-Pacific Software Engineering Conference (APSEC05) 0-7695- 2465-6/05
[4] Wagner, David; Schneier and Bruce,"Analysis of the SSL 3.0 Protocol," The Second USENIX Workshop on Electronic Commerce Proceedings, USENIX Press. Nov 1996.
[5] Ng, Raymond K."Distributed capability-based authorization architecture using roles"2004.
[6] Robert Jr., Howard High (Round Rock, TX, US), Nadalin, Anthony Joseph (Austin, TX, US), Nagaratnam,Nataraj (Morrisvile, CA, US) ,"Role-permission model for security policy administration and enforcement" 2003.
[7] Dixit, Royyuru (Wilmington, MA, US), Hafeman, Joseph Edward (Holliston, MA, US), Vetrano, Paul Michael(Franklin, MA, US), Spellman, Timothy Prentiss (Framingham, MA, US), "Role-based access in a multi customer computing environment",2006.
[8] [8] Vinith Bindiganavale and Jinsong Ouyang, Member, IEEE [2001]." Role Based Access Control in Enterprise Application – Security Administration and User Management"
[9] Collin Jackson and Adam Barth," Force HTTPS: Protecting High Security Web Sites from Network Attacks"
[10] Sanin, Aleksey (Sunnyvale, CA, US),"Web service security filter"
[11] Chung, Hyen V. (Round Rock, TX, US), Nakamura, Yuhichi (Yokohama-Shi, JP), Satoh, Fumiko (Tokyo, JP), "Security Policy Validation for Web Services".
[12] Kou, Wei Dong (Pokfulam, HK) ,Mirlas, Lev (Thornhill, CA), Zhao and Yan Chun (Toronto, CA)," On Secure session management and authentication for web sites",2005.
[13] Akram Alkouz and Samir A. El-Seoud (PSUT) Jordan," Web Services Based Authentication System For E-Learning", 2005.
[14] Thomas Price, Jeromie Walters and Yingcai Xiao, "Role –Based Online Evaluation System", 2007.[15] Srinath Akula, Veerabhadram Devisetty , St. Cloud, MN 56301." Image Based Registration and Authentication System", 2002.Rivest, Shamir and Adelman,"RSA public-key encryption".
[15] Microsoft,"Asp.net".Website:http://www.asp.net/index.html.
[16] Netfilter Core team, Iptables, an userspace packet filtering program
[17] Website:http://www.netfilter.org/projects/iptables/index.html.
[18] Digital Certificates for Internet Security and Acceleration Server 2004, for Microsoft Forefront Threat Medium Business Edition, or for Windows Essential Business Server 2008. Website: http://support.microsoft.com/kb/888716
[19] OpenID Foundation. Website: openid.net/
[20]] Microsoft,"Visual Studio .Net". Website:http://msdn.microsoft.com/vstudio/
[21] Microsoft Corporation. Website: www.passport.net/
[22] OpenSSL team. Website: http://www.openssl.org/

AUTHOR BIOGRAPHIES

Dharmendra Choukse holds a M.Tech in Information Technology from Devi Ahilya University, Indore-NDIA. He is currently Pursuing Ph.D. in Computer Science From Institute of Computer Science, Vikram University, Ujjain-INDIA.and He is also currently Sr Software Engineer in Institute of Engineering & Sciences,IPS Academy, Indore-INDIA. A He served as Software Engineer in Choksi Laboratories ltd,Indore. His research interest includes network security, secure electronic commerce, client-server computing and IT based education.

Dr. Umesh Kumar Singh obtained his Ph.D. in Computer Science from Devi Ahilya University, Indore-INDIA. He is currently Reader in Institute of Computer Science, Vikram University, Ujjain-INDIA. He served as professor in Computer Science and Principal in Mahakal Institute of Computer Sciences (MICS-MIT), Ujjain. He is formally Director I/c of Institute of Computer Science, Vikram University Ujjain. He has served as Engineer (E&T) in education and training division of CMC Ltd., New Delhi in initial years of his career. He has authored a book on " Internet and Web technology " and his various research papers are published in national and international journals of repute. Dr. Singh is reviewer of International Journal of Network Security (IJNS), ECKM Conferences and various Journals of Computer Science. His research interest includes network security, secure electronic commerce, client-server computing and IT based education.