B-2-B E-COMMERCE SOLUTION

AND

RIJNDAEL INTERFACE

PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING – INFORMATION TECHNOLOGY

BHARATHIAR UNIIVERSITY, COIMBATORE.

SUBMITTED BY

Ms. P.SARITHA

Ms. V.LAKSHMI PRIYA

Ms.R.RAMYA

P. 681

External Guidance
Mr. Mahesh Kumar, B.E,
Websea e-guide tech,
Chennai.

Internal Guidance
Ms. S.Rajini B.E,
Senior lecturer,
Dept of Computer Science,
Kumaraguru College of Technology,
Coimbatore.

Department of Computer Science and Engineering
Kumaraguru College of Technology
Coimbatore
MARCH 2002

INTRODUCTION

CHAPTER

>>> INTRODUCTION <<<

1.1 PROJECT SCOPE

The project entitled as " Net Diagnostics " , a dedicated real time web project using java server pages automises the managerial activities of Apollo diagnostic centre. The Patient's ,Doctor's & others get support with the information provided & carry out other activities in a comfortable manner.

1.2 NECESSITY OF THE PROJECT

- To automise the various activities enclosed by the organization.
- To expose the information regarding organization and the activities carried by it.
- To accustom with the existing network technology ie.Internet.
- To make the members to fix the appointment online and carry out other activities through web.

- To provide mailing facility for effective communication between the members of the organization.
- To validate the users while transferring the messages.
- To allow only valid users to communicate with each other.
- To provide very high security and high speed.
- To handle multiple user requests simultaneously.
- To present a user-friendly environment.
- To avoid disturbances during the workflow.
- To ensure efficient management of the resources.
- To receive the reports and to know about things at one place.

1.3 EXISTING SYSTEM

The existing system does the activities manually.The information of various patient's are maintained in records.Normally patient's approach the doctor's for some reasons and the doctor performs a checkup and suggests some tests by referring some labs where they can carry out the test.After testing ,reports are issued to the person for whom the test was carried out.This report will be handed over to the doctor in person and he/she analyse the report and explains suggest the patient's regarding the situation and provide guide for the betterment .

1.4 PROPOSED SYSTEM

The intention of proposed system is to automise the existing system .It enable the diagnostic centre to manage the entire activity through computer systems with the use of web technology (ie., Internet). It provides effective communication between the members.

The Main Modules of Net Diagnostics are;-

(1) Doctor :

In this module,doctor's are allowed to register themselves to become a member of the diagnostics centre.They analysis the out patients and through them to carry out certain tests if necessary,to the diagnostics centre by registering the patient so that fixing an appointment online for carrying out the tests.After completion of test the reports will be published in the web .This reports can be viewed only by the concerned doctor's.They are also allowed to view various information like the test carried out in the lab,various free camps organized by the organization. Mailing facility has been provided for the doctor's with which they can send mails either to the diagnostic centre or the patients.

(2) Patient :

In this module,patient's can themselves register to become a member of the diagnostic centre. They can fix appointments through online. They can send and receive messages from diagnostic center and doctor's.They have access to view various information regarding diagnostic centre and other activities carried out by them .The reports published after carrying out the test can either be viewed or downloaded.

(3) Administrator :

In this module,all the in and around activities are monitored and are under the control of the administrator. Appointments and mails are handled efficiently by the administrator. Various updated are instantly brought to focus by updating the information online.

SYSTEM ANALYSIS

CHAPTER 2

>>> SYSTEM ANALYSIS <<<

2.1 PROBLEM STATEMENT

- **The doctor's and patient's should register to become the member of the organization.**
- **The communication between the members should be carried out through mailing system.**
- **The appointments should be handled via.Online.**
- **The reports should reach the members via.Online.**
- **The information regarding various diagnostics carried out and other activities should be published in the web.**
- **The administrator should be made to handle all the above activities in an efficient manner.**

2.2 System Requirements

2.2.1 Hardware Requirements

PROCESSOR	**Intel Pentium II ,350 Mhz & more**
HARD DISK	**20 GB**
RAM	**128 MB**
KEYBOARD	**Enhanced 101 / 102 Keys**
MOUSE	**Logitech (3 Buttons)**
MONITOR	**COLOR**
MODEM	**Any modem with minimum 56 kbps**
TELEPHONE LINE	**One Line**

2.2.2 SOFTWARE REQUIREMENTS

Operating System	**Windows 2000/NT/XP**
Front end Tools	**HTML 4.0,Frontpage 2000, Dreamweaver Ultradev 4**
Back End	**Microsoft Access 2000**
Network Connectivity	**Java Server Pages 1.1**
Scripting Language	**JavaScript**
Design Tools	**Adobe Photoshop 6.0**
Web server	**Tomcat V.3.2**

2.3 SOFTWARE SPECIFICATION

2.3.1 HTML (Hyper Text Markup Language)

The World Wide Web is the set of all web sites and the documents they can provide to clients(users).Html lies a foundation and builds the WWW. Html is the language that puts the face on the Webby helping to prepare documents for online publications.Html documents are also called Web documents, and each Html document is known as Web Page. A page is what is seen in the browser at any time.Each Web Site, whether on the internet or intranet, is composed of multiple pages and it is possible to switch among them by following hyper links.

A Web Page is basically a text page that contains the text to be displayed and references to elements such as images, sounds and of course hyperlinks to other documents.Html page can be created using simple text editor such as Notepad or WYSIWAG .Web page editors such as Microsoft FrontPage. In either case the result is a plain text file that computers can easily exchange. The browser interprets this text file and renders on the client side.

Html is really a set of codes called "tags " – for creating an entire internet browser presentation.Html is not a programming language, and a Html document is not a program. It's much simpler than all of that, it is a well-devised collection of tags and markers which allow you to turn ordinary text into instructions that a browser can interpret.

Html consists special tags which allow programs to execute external programs.The applet tag enables java other scripting languages like Vb Script and java script..Html supports Hyper media thus making the internet a more interesting area. It is sufficiently general to allow it to be used with a variety of browsers and computers. The language does not specify all the display details, but gives the browser the freedom to choose the configuration. Much of the popularity of the internet can be credited to the Html which has made it easy to create the web pages Html may seem to require advanced technical skills but it does not. Moreover anyone can master Html in a small amount of time. That is the reason the Html is so well known in the world of internet.

Finally Html can be downright fun. It gives certain satisfaction from building a Web page from the ground up.

It's like building our own house, in which every brick, every nail etc are known. This facilitates the easy modification to acquire the desired result. It also makes it much easier to take a look at someone else's page and know how they achieve their effect.

2.3.2 JAVA 2

The java programming language is uniquely suited for distributing executable content over Networks. Java also offers a set of functions similar to many other programming languages. As a language for delivering information on the web, java connects to the web's hyper text markup language using a special tag called applet.

2.3.2.1 ADVANTAGES OF JAVA

2.3.2.1.1 SIMPLE :

One of the design goals behind java was to make it familiar to a large number of users. They have simplifies matters by removing concepts like pointers.

2.3.2.1.2 OBJECT ORIENTED :

Java is Object Oriented. It concentrates on the data rather than procedures.

2.3.2.1.3 DISTRIBUTED :

Java is designed to support networking and network operations right from the start.

2.3.2.1.4 INTERPRETED :

Java is interpreted language rather than a language that is compiled and run. Java can run on any system.

2.3.2.1.5 ROBUST :

Java is strongly typed language that allows run time checking and no pointers to worry about there are no overwriting of distant memory areas and corrupting data.

2.3.2.1.6 SECURE :

Java implements several security mechanisms to protect one from attempts to create viruses or invade one's file system.

2.3.2.1.7 NEUTRAL :

An application written in java will run on all systems to provided that java interpreter is available. it runs on any machine such as Windows or Unix or Even Unix or Even Linux.

2.3.2.1.8 PORTABLE :

In addition to the java interpreter several other aspects contribute to its portability. No assumptions are made about the size of the data types.

2.3.2.1.9 HIGH PERFORMANCE :

Just in time compilers help Java achieve higher performance that java interpreter alone can reach.

2.3.2.1.10 MULTITHREADED :

Java supports multiple threads of execution to handle different tasks.

2.3.2.1.11 DYNAMIC :

Java was designed to adapt to changing environment and can load classes as they needed even across the network.

2.3.3 JSP - Java Server Pages

Java Server Pages (JSP) is a technology for controlling the content or appearance of Web pages through the use of servlets. JSP is comparable to Microsoft's Active Server Page (ASP) technology. JSP technology allows Web developers and designers to

rapidly develop and easily maintain, information – rich, dynamic Web pages that leverage existing business systems. The JSP technology enables rapid development of Web-based applications that are platform independent.

Java Server Pages is presentation layer technology that sits on top of java servlets model and makes working with Html easier. It allow you to mix static Html content with server side scripting to produce dynamic output. By default, Jsp uses Javascript as its scripting language just as Asp can use other languages. so with java will be more flexible and robust than scripting platforms based on simple languages.Jsp provides a robust web application platform and a number of server-side tags that allow developers to perform most dynamic content operations.

Java Server Pages technology is an extension of the Java Servlet API. Servlets are platform-independent, 100% pure java server-side modules that extend the capabilities of a Web server with minimal overhead, maintenance, and support. Together JSP technology and servlets provide an attractive alternative to other types of dynamic Web scripting/ programming that offers platform independence, enhanced performance, separation of logic from display, ease of administration,

extensibility into the enterprise and most importantly, ease of use.

JSP pages share the "Write Once, Run Anywhere" characteristics of Java technology. There are many technologies available for the dynamic content generation on the server. They are CGI, Servlets, ASP and JSP. But JSP is advantageous over other technologies.

2.3.3.1 ADVANTAGES OF JSP

- JSP maintains state between session.
- A new thread is spawned for each request.
- JSP is loaded only once at the time of initiation.
- JSP runs in a JVM as an extension to the Web server.
- JSP provides better separation of page code and template data.
- JSP can be run on all major Web servers.
- JSP is simpler to write and provide a separation of presentation from logic.

2.3.3.2 Java Server Pages (JSP) Vs. Active Server Pages (ASP)

The advantages of JSP are twofold .First ,the dynamic part is in written in Java, not Visual Basic or other MS-specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and Microsoft Web servers.

2.3.4 JDBC

JDBC is a Java API for executing SQL statements.(As a point of interest ,JDBC is a trade marked name and is not an acronym; nevertheless, JDBC is often thought of as standing for "Java Database Connectivity").It consists of a set of classes and interfaces written in the java programming language.JDBC provides a standard API for tool/ database developers and makes it possible to write database applications using a pure Java API.

Using JDBC it is easy to send SQL statements to virtually any relational database. In other words ,with the JDBC API ,it isn't necessary to write one program to access a Sybase database, another program to access an Oracle database and so on. One can write a single program using the JDBC API, and the program will be

able to send SQL statements to the appropriate database. And, with an application written in the Java programming language, one also doesn't have to worry about writing different applications to run on different platforms. The combination of Java and JDBC lets a programmer write it once and run it anywhere.

Java being robust ,secure ,easy to use, easy to understand ,and automatically downloadable on a network, is an excellent language basis for database applications. What is needed is away for Java applications to talk to a variety of different databases.JDBC is the mechanism for doing this.

JDBC extends what can be done in java ,For example, with Java and the JDBC API it is possible to publish a web page containing an applet that uses information obtained from a remote database. Or an enterprise can use JDBC to connect all its employees (even if they are using a conglomeration 0f Windows, Macintosh and Unix machines) to one or more internal databases via intranet. With more and more programmers using Java ,the need for easy database access from Java is continuing to grow.

MIS managers like the combination for Java and JDBC because it makes disseminating information easy and economical. Businesses can continue to use their

installed databases and access information easily even if it is stored on different database management systems. Development time for new applications is short. Installation and version control are greatly simplified. A programmer can write an application or an update once, put it on the server, and everybody has access to the latest version .And for businesses selling information services ,Java and JDBC offer a better way of getting out information updates to external customers.

JDBC is a "low level" interface, which means that it is used to invoke SQL commands directly .It works very well in this capacity and is easier to use than other database connectivity API's, but it was designed also to be a base upon which to build higher level interfaces and tools. A higher-level interface is "user friendly" , using a more understandable or more convenient API that is translated behind the scenes into a low -level interface such as JDBC.

2.3.5 MICROSOFT ACCESS

Microsoft Access is a user- friendly Database system. The reasons for the success of Access are its easiness to use I with the Gm based design and its wide availability without any compromise on the core objects. Microsoft Access offers some significant features.

2.3.5.1 DELAY LOADING

Microsoft Access doesn't load software components that aren't required for all databases ,such as Visual Basic for Applications and Data Access Objects, until they are needed. This shortens the time it takes a database to load and improves overall performance.

2.3.5.2 FAIL ON ERROR PROPERTY

We can optimize bulk update queries for ODBC data sources by sending the query to the server, where all appropriate records are processed instead of one record at a time.

2.3.5.3 IMPROVED CALL TREE LOADING

Microsoft Access does not load modules, including form modules, until the JSP code in the module is executed. This improves overall performance.

2.3.5.4 IMPROVED COMPLIED STATE MGMT

We can maintain the complied state of your database even if you modify it. Only the modified code and any code that depends on the modified code will decompile.

2.3.5.5 PERFORMANCE ANALYSER

This wizard analyses uses database objects and suggests ways to make them as fast as they can be.

2.3.5.6 FEATURES ON THE INTERNET

Microsoft Access provides extensive new features designed to help you easily use the internet and develop a World Wide Web application. Import, Export, Link features use to import or link HTML files. Import or links (read-only) tables or lists from and HTML file using the Get External data command on the file menu.Export objects to HTML format. Export reports to static HTML format and data sheets and forms to static or dynamic HTML format by using the save as/Export command on the File menu. Enhance the appearance, consistency ,and navigation of your web pages by using an HTML template file.

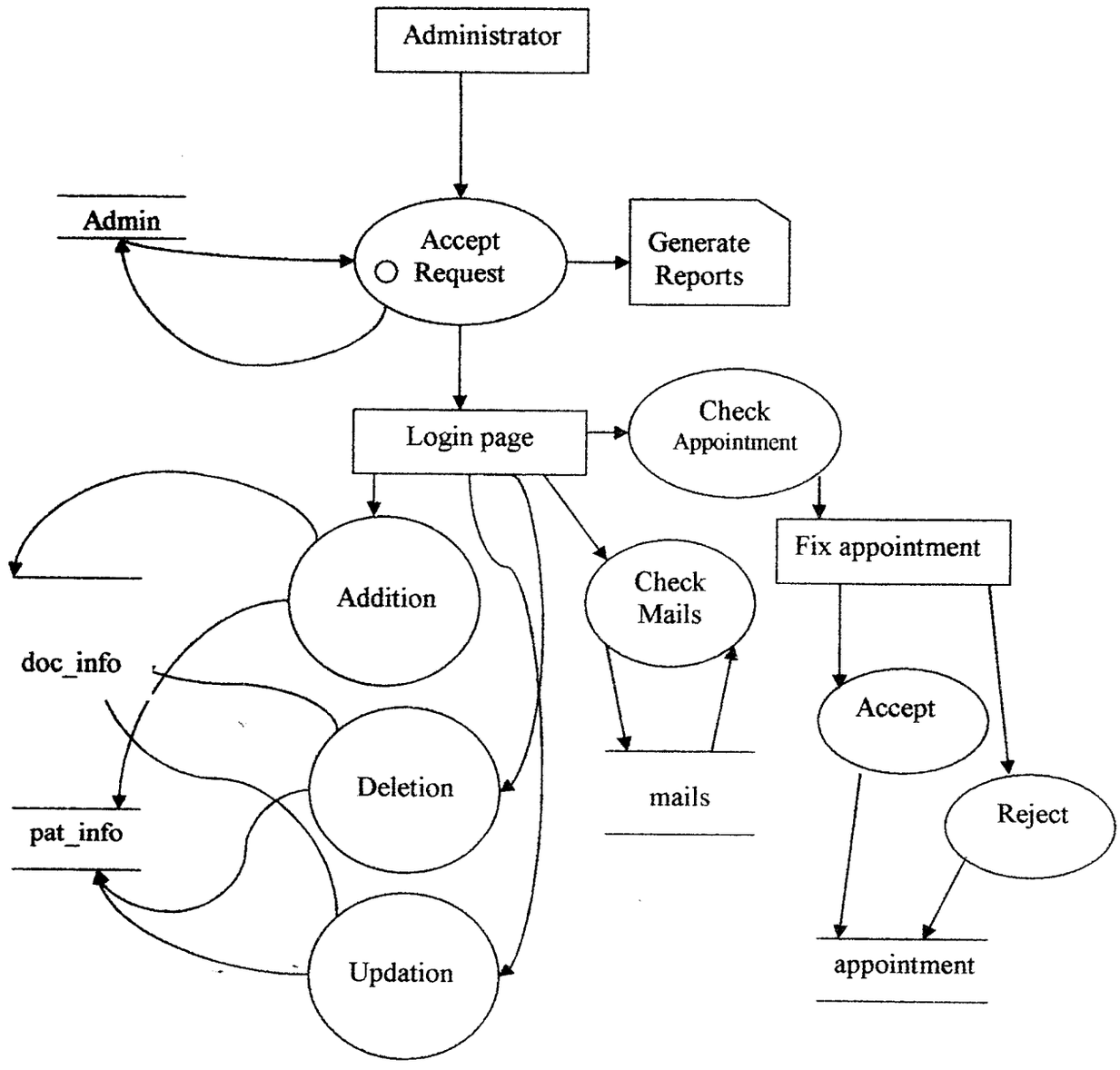
SYSTEM DESIGN

CHAPTER - 3

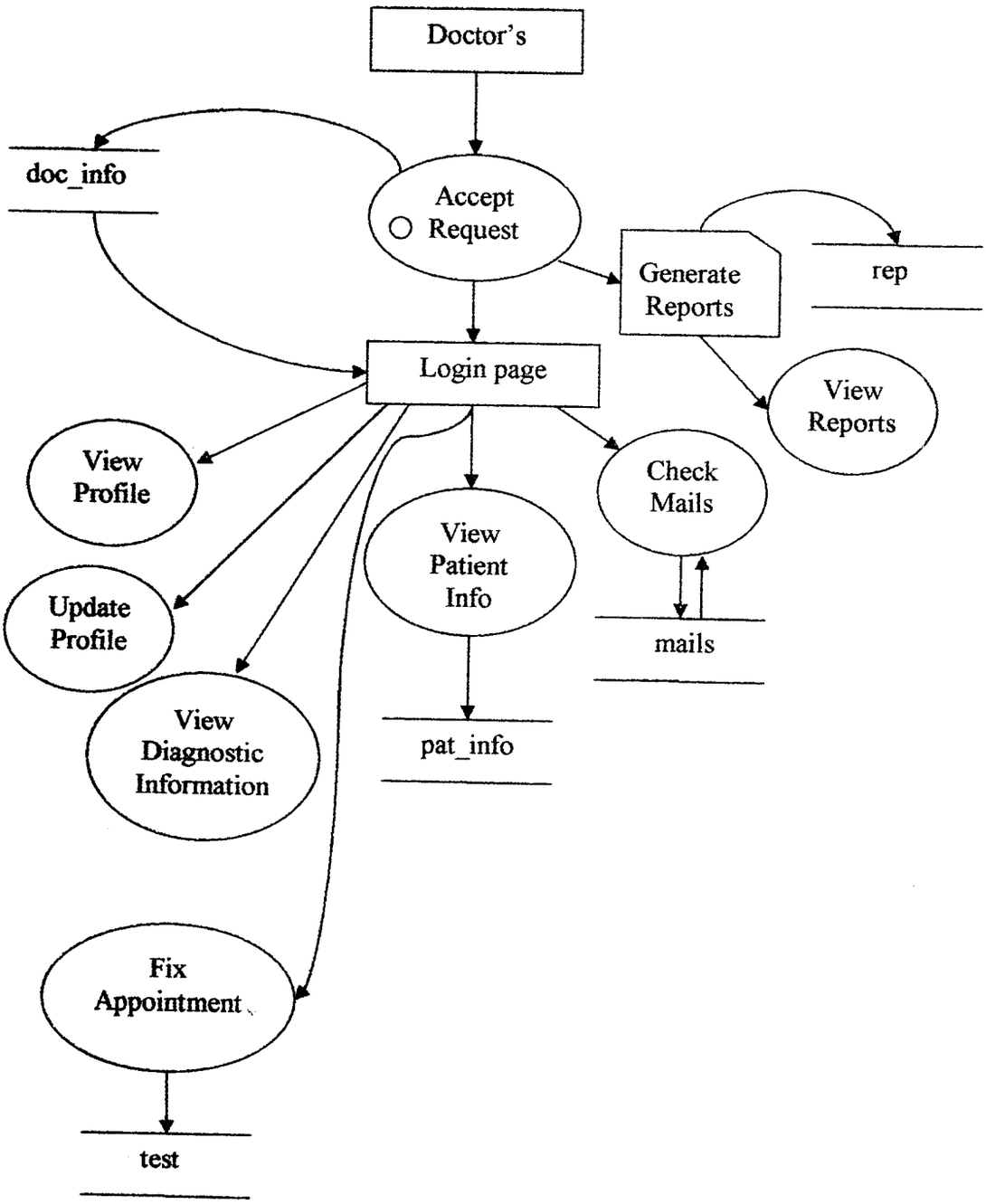
>>> SYSTEM DESIGN <<<

3.1 DATA FLOW DIAGRAM

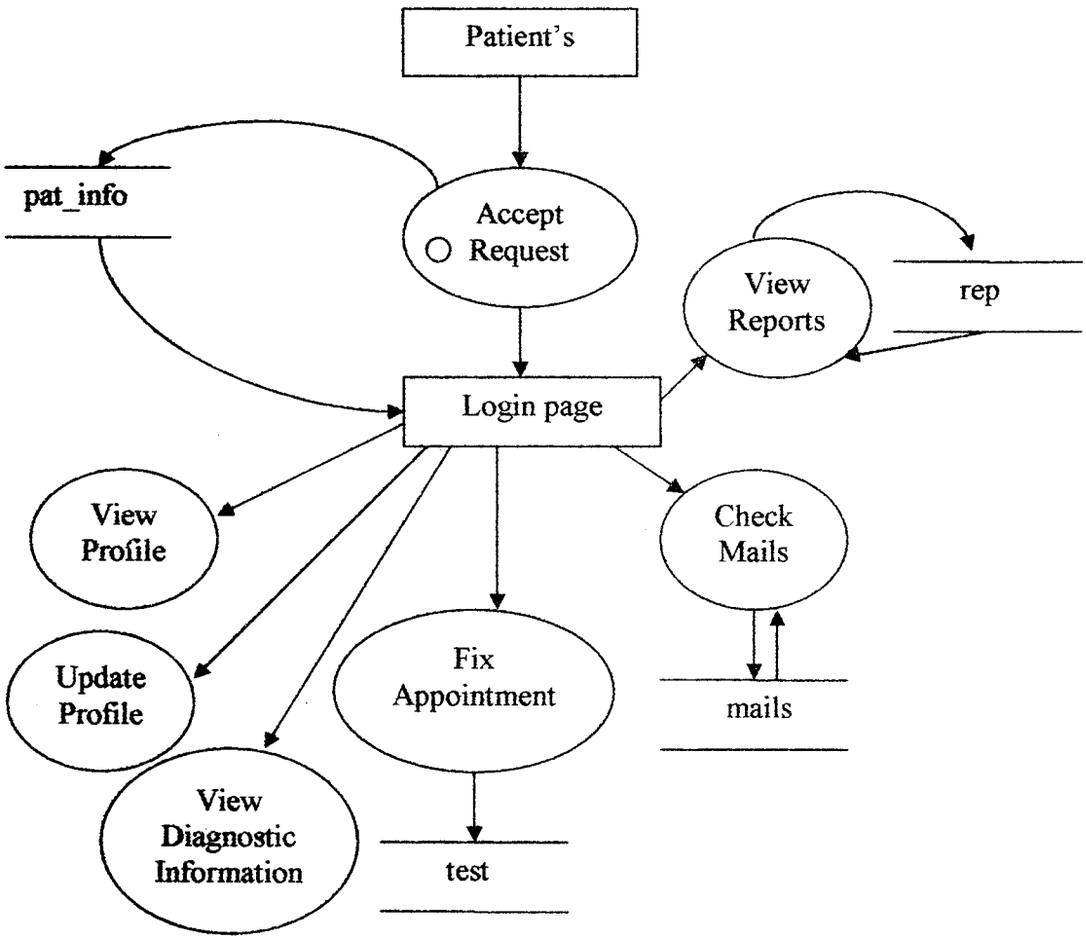
3.1.1 ADMINISTRATOR – DFD



3.1.2 DOCTOR'S - DFD



3.1.3 PATIENT'S - DFD



3.2 TABLE DESIGN

3.2.1 diag_infomas (Diagnostic Information)

S.no	Field name	Data type	Field size	Description
1	diag_id	Text	6	Diagnostic id
2	diag_name	Text	25	Diagnostic name
3	diag_gpid	Text	6	Diagnostic group id
4	diag_gpname	Text	25	Diagnostic group name
5	diag_info	Text	255	Diagnostic information
6	diag_place	Text	20	Diagnostic place
7	diag_time	Date/time		Diagnostic duration
8	diag_cost	Number		Diagnostic cost

3.2.2 doc_info (Doctor's Information)

S.no	Field name	Data type	Field size	Description
1	dr_id	Text	6	Doctor id
2	dr_pwd	Text	7	Doctor password
3	dr_name	Text	50	Doctor name
4	dr_sex	Text	6	Doctor sex
5	dr_dob_date	Number	3	Doctor birth-date
6	dr_dob_mon	Text	10	Doctor birth –month
7	dr_dob_year	Number	4	Doctor birth-year
8	dr_area_sp	Text	25	Area of Specialization
9	dr_hadd	Text	50	Doctor House Address
10	dr_hcity	Text	20	City
11	dr_hpin	Number	6	Pincode
12	dr_hph	Number	7	Phone number
13	dr_hcell	Text	11	Cell number

S.no	Field name	Data type	Field size	Description
14	dr_cadd	Text	50	Clinic address
15	dr_ccity	Text	20	City
16	dr_cpin	Number	6	Pincode
17	dr_cphone	Number	7	Clinic phone number
18	dr_cfax	Text	20	Fax number
19	dr_email	Text	50	Email id

3.2.3 pat_info (Patient's Information)

S.no	Field name	Data type	Field size	Description
1	pat_id	Text	6	Patient id
2	pat_pwd	Text	7	Patient password
3	pat_name	Text	50	Patient name
4	pat_add_res	Text	50	Patient address-residence

S.no	Field name	Data type	Field size	Description
5	pat_add_off	Text	50	Patient address-office
6	pat_pno_off	Number	7	Patient phone-office
7	pat_pno_res	Number	7	Patient phone – residence
8	pat_cell	Number	10	Patient cell number
9	pat_dob_day	Number	3	Patient birth-day
10	pat_dob_mon	Text	10	Patient birth-month
11	pat_dob_year	Number	4	Patient birth-year
12	pat_sex	Text	6	Patient gender
13	pat_email	Text	50	Patient email id

3.2.4 admin (Administrator)

S.no	Field name	Data type	Field size	Description
1	Adid	Text	6	Admin id
2	Pass	Text	6	Admin Password

3.2.5 bd_info (Blood Donor's Information)

S.no	Field name	Data type	Field size	Description
1	dor_id	Text	6	Donor id
2	dor_name	Text	40	Donor name
3	dor_add	Text	50	Donor address
4	dor_ph_res	Number	7	Donor phone residence
5	dor_ph_off	Number	7	Donor phone office
6	dor_cell	Number	10	Donor cell number
7	dor_bd_gp	Text	2	Donor blood group
8	dor_bd_rhfac	Text	9	Donor blood Rh factor
9	dor_email	Text	50	Donor email id

3.2.6 app (Appointments)

S.no	Field name	Data type	Field size	Description
1	pat_id	Text	6	Patient id
2	Date	Text	10	Date of visit
3	Time	Text	10	Time of visit
4	Message	Text	255	Messages

3.2.7 bd_vol (Blood availability)

S.no	Field name	Data type	Field size	Description
1	Bgid	Text	6	Patient id
2	Units_ava	Number	4	Units available

3.2.8 mem_mail (Mails)

S.no	Field name	Data type	Field size	Description
1	Fromid	Text	6	Sender id
2	Toid	Text	6	Receiver id
3	mess	Text	255	Message

3.2.9 rep(reports)

S.no	Field name	Data type	Field size	Description
1	pat_id	Text	6	Patient id
2	Rep	Text	50	Report Link

3.2.10 test (Appointments)

S.no	Field name	Data type	Field size	Description
1	pat_id	Text	6	Patient id
2	diag_name	Text	50	Report Link
3	date_submitted	Text	10	Date of Submission
4	Time_submitted	Text	10	Time of Submission
5	doc_ref	Text	30	Doctor referred
6	sug_date	Text	10	Suggested date
7	sug_time	Text	10	Suggested time
8	Status	Text	10	Status

IMPLEMENTATION

CHAPTER - 4

>>> IMPLEMENTATION <<<

4.1 DOCTOR'S MODULE :

4.1.1 New Member Sign up :

The user's who are new, must register there information by filling up the registration form.

4.1.2 Doctor's Check In :

The doctor's those who already got registered , provide the user id and password to enter the authentication process for entering in to the login page.

4.1.3 Doctor's Profile :

The doctor's can view , edit or update there profiles whenever necessity plays a role.

4.1.4 Diagnostic Information :

The doctor's can view the various diagnostic details that are handled by the diagnostic centre.

4.1.5 Fix Appointments :

The doctor's can make up the appointments with the diagnostic centre for the patient's .

4.1.6 Check Mails :

The doctor's can use the mailing system to check, send and receive mails from the member's of the organization (ie., Administrator ,patient's)

4.1.7 Check patient's information :

The doctor's can view the details of the patient for whom the reference was made.

4.1.8 Check reports :

The doctor's can view the test results of the patient's i.e., in the form of reports.

4.2 PATIENT'S MODULE :

4.2.1 New user's – register :

The user's who are new to the organization, should complete the registration process to become the member of the organization.

4.2.2 Patient's Login In :

The patient's those who already got registered , provide the user id and password to enter the authentication process for entering in to the login page.

4.2.3 Patient's Profile :

The patient's can view , edit or update there profiles whenever necessity plays a role.

4.2.4 Diagnostic Information :

The patient's can view the various diagnostic details that are handled by the diagnostic centre.

4.2.5 Fix Appointments :

The patient's fill up the request form to place the appointment with the diagnostic centre .

4.2.6 Check Messages :

The status regarding the request made by the patient towards the appointment put through to the diagnostic centre will be reported by the administrator.

4.2.7 Check Mails :

The patient's can use the mailing system to check, send and receive mails from the member's of the organization (ie., Administrator ,Doctor's).

4.2.8 Check Reports :

The patient's can view the test results i.e.,Reports.They can either just open the report for viewing or otherwise they can download the report.

4.3 ADMINISTRATOR MODULE :

4.3.1 Add Member's :

The administrator can add the information ie., register the details of the person's so that they become the member of the organization.

4.3.2 Delete Member's ;

The administrator can remove the member's from the company profiles.

4.3.3 Modify Member Information :

The administrator can edit or modify the information regarding any member's at any time without any restriction.

4.3.4 Add Diagnostic Information :

The administrator can add any new diagnostic that is to be handled in the lab.

4.3.5 Delete Diagnostic Information :

The administrator can remove any diagnostic details from the database.

4.3.6 Modify Diagnostic Information :

The administrator can edit or modify any diagnostic information.

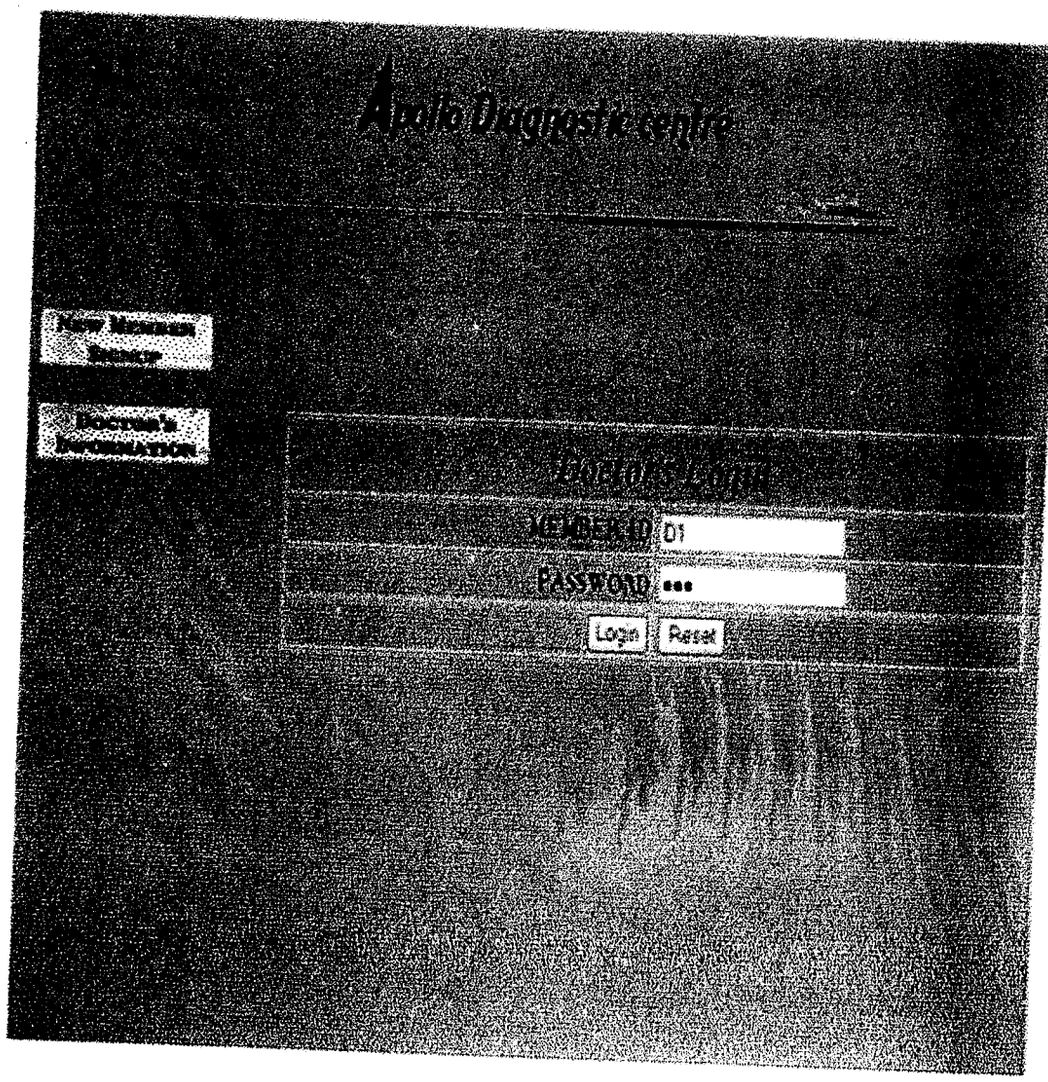
4.3.7 Mails :

The administrator can check up the mails arrived,and also send mails to any or all the member's of the organization.

4.3.8 Appointments :

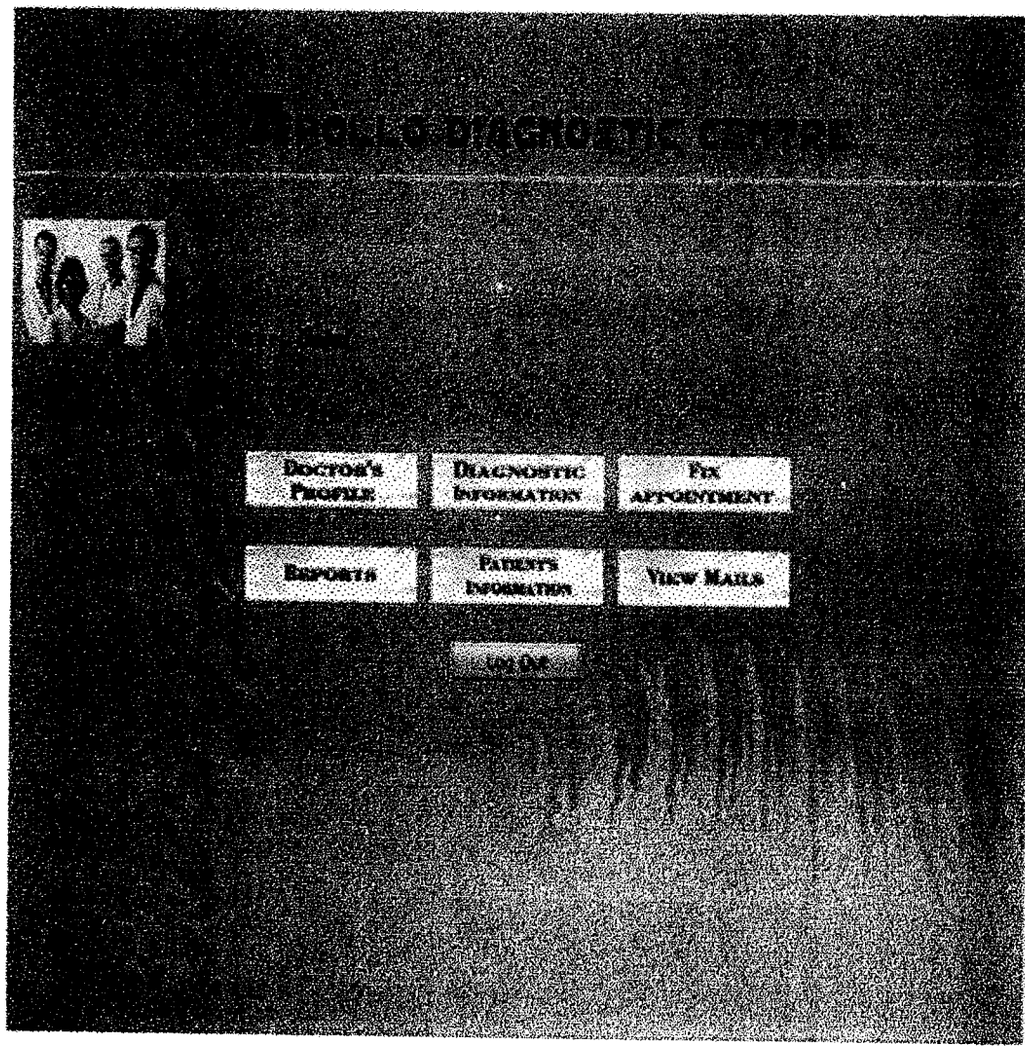
The administrator can view the appointments and make a response by either accepting the request or rejecting it.The administrator forwards a reply message to the member who made the request.

(FIG. 4A)
(DOCTOR'S AUTHENTICATION PAGE)



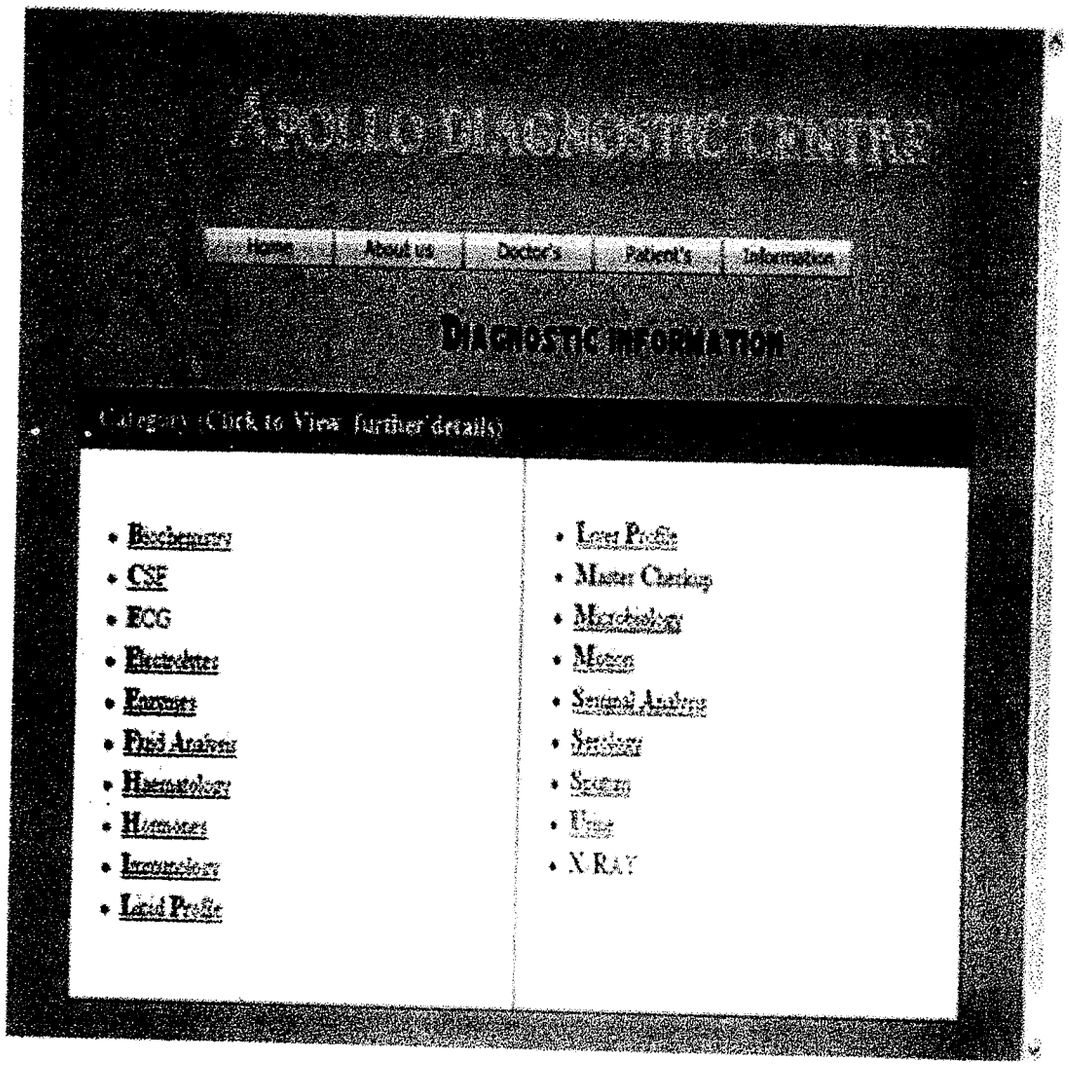
The user should provide the user id and password for authentication process . New user's should register by clicking the button - new member sign up. To know the information regarding doctor's , user's should click doctor's information button.

(FIG : 4B)
(DOCTOR'S LOGIN PAGE)



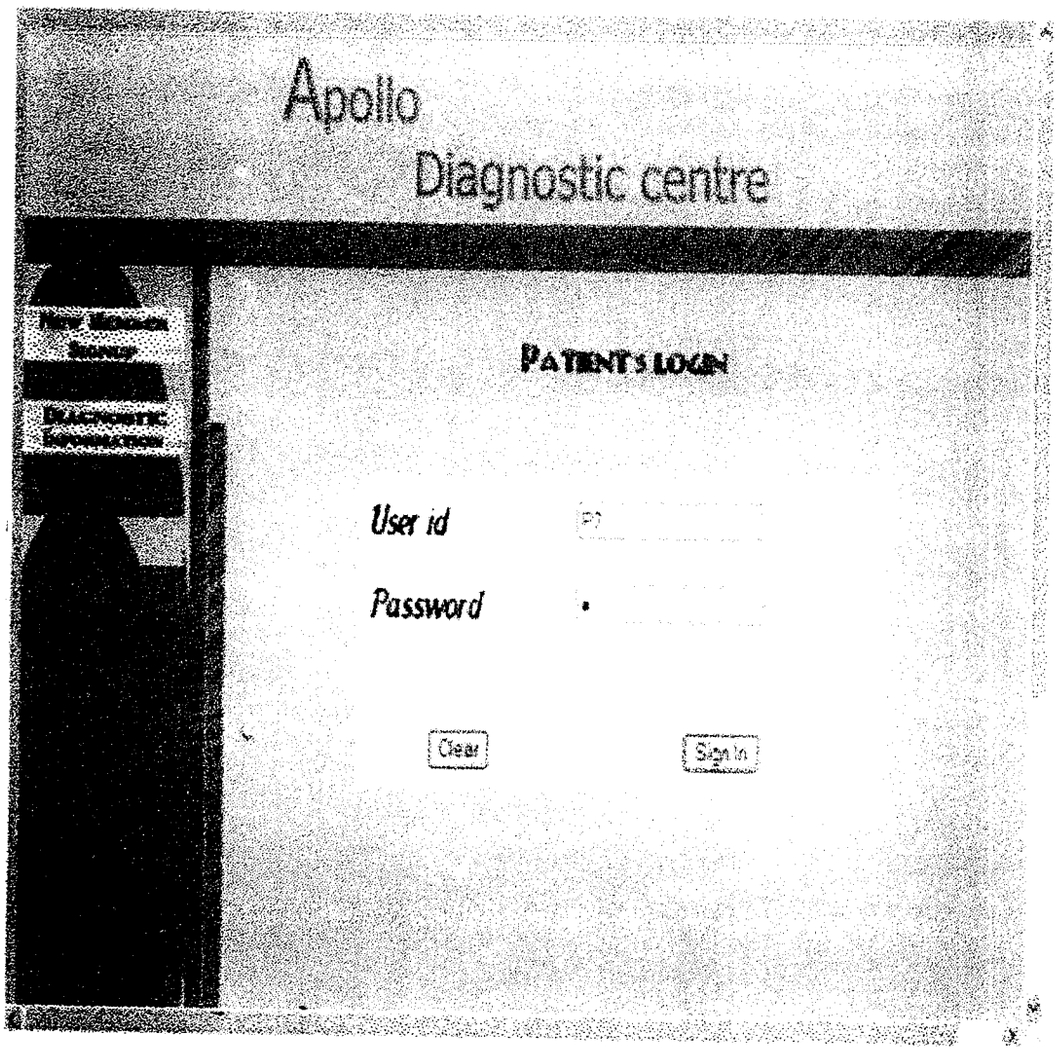
LOGIN PAGE PROVIDES VARIOUS LINKS AS SHOWN FROM THE ABOVE FIGURE TO SURF THROUGH THE WEB.FROM THIS PAGE DOCTOR HANDLES VARIUOS TASKS LIKE MAILING,FIXING APPOINTMENTS,VIEWING REPORTS etc.,

(FIG : 4C)
(DIAGNOSTIC INFORMATION)



VARIOUS TESTS CARRIED OUT ARE LISTED IN THE ABOVE FIGURE.

(FIG : 4D)
(PATIENT'S AUTHENTICATION PAGE)



PATIENT'S WHO HAVE ALREADY REGISTERED PROVIDE THE USER ID AND PASSWORD TO ENTER THE AUTHENTICATION PROCESS.OTHER'S THEY HAVE TO REGISTER TO ENTER AS A AUTHORISED USER .

41 (a)

(FIG : 4D)
(PATIENT LOGIN PAGE)

PATIENT'S ARE PROVIDED WITH LINKS WHERE THEY CAN PERFORM THE ACTIVITY WHICHEVER THEY DESIRE.

APOLLO DIAGNOSTIC CENTER

[Computerised Lab]

30,D.B.Road,R.S.Puram,Coimbatore -641002 Phone : 2454373

Name : Santhosh Kumar Age : 15 yrs Sex : Male

Ref. by Dr. K. MANONMANI MBBS AB (PED) USA

Nature of Specimen : Blood Date : 11/03/2003

Diagnostic Name	*Result*
Haemogoblin	11.4 gms % (75 %)
Total WBC Count	10,750 Cells / cu mm
Differential Count	Poly : 56 % Lymph : 40 % Eosin : 04 %
Blood picture	Erythrocyte : Normcytic mild hypochromic Leucocyte : Appear Normal No toxic granulation and abnormal cells seen
Plaletes Count	2,86,000 / cu mm : (Normal 1 to 3 Lakhs / cu mm)

LAB TECHNOLOGIST

COMPUTERISED ANALYSER FROM **BOEHRINGER MANNHEIM** *, GERMANY*

Branches :

No: 58, Cowley Brown Road ,R.S.Puram,Coimbatore -641 002.Phone : 551128

No. 293,N.H.Road,Next to Naaz Theatre ,Town Hall ,Coimbatore – 641 001. Mobile : 98430 - 45437

4-2 (6)

SOFTWARE TESTING

CHAPTER - 5

>>> SOFTWARE TESTING <<<

5.1 DOCUMENT OF TESTING

Software testing can be looked upon as one of the many processes. This is the last opportunity to correct any possible flaws in the developed system.

Software testing includes selection tests and test data that have more probability of finding errors.

System is the stage of implementation that is aimed at ensuring that the system works accurately and efficiently before live operation commences. In principle, system proving is an ongoing activity throughout the project.

The logical design and the physical design should be thoroughly and continually examined on paper to ensure that they will work when implemented. Thus the system test in implementation should be a confirmation that all is correct and an opportunity to show the users that the system works. The first step in system testing is to develop a plan that tests all the aspects of the system. Completeness, Correctness, reliability and maintainability

of the software are to be tested for the best quality assurance -an assurance that the system meets the specification and requirements for its intended user and performance. System testing is the most useful practical process of executing a program with explicit intention of finding errors that make the program fail.

5.2 TESTING PROCESS

5.2.1 LOGIN TESTING :

The login process was tested with both authorized and unauthorized login. Access was denied for invalid login ids and correct passwords. Password changing facility was also provided. The results were as expected.

5.2.2 MODULE TESTING :

Each individual program module is tested for any possible errors. They were also tested for specifications, i.e., to see whether they are working as per what the program should do and how it should perform under the various conditions .

5.2.3 CONCURRENCE TESTING :

Since the system is a multi-user it was tested for concurrence problems. The system worked perfectly since the

table locking and other security measures were taken care by the database itself.

5.2.3 INSERT UPDATE TESTING :

The insertion and updating were tried on all required tables in the Database. Checking was done to see whether the corresponding entries were made in the different tables when a new record was created. Updating of non-existent records and duplicate values were tried. The system was found to check and disallow invalid attempts.

5.2.4 DISPLAY TESTING :

The display procedure was tested since the data displayed is of much importance. The data was input in the different modules and it was checked whether the information is properly displayed in the other dependent modules. The consistency of the display and the attractiveness of the display were also tested. A testing is an examination with the intent of finding errors. Concentration was more on errors rather than formatted and perfect outputs. Comments and suggestions from the observers during the test run were later considered. Special care was given to user interface comments.

5.2.4 DATABASE TESTING

The overall objective in the development of database has been to treat the whole database as one. Database management allows the data to be protected and organized separately from other resources. Defining the term database is difficult, it can be defined as an integrated collection of data.

- **Determine the purpose of the database .**
- **Determine the tables needed in the database .**
- **Determine the fields needed in the tables .**
- **Identify the fields with unique values.**
- **Determine the relationship with tables .**
- **Refine design .**

5.2.5 DATABASE OBJECTIVE :

The organization of the data in a database aims to achieve two major objectives: Data Integrity and Data Integration.

5.2.6 DATA INTEGRATION :

Within same computer system, reports or analysis referencing the same logical information are inconsistent owing to the difference in the duplicated physical data. This could for example occur when the changes are made to one file but not to

a copy of the same data in another file or table. One way to solve the problem is to ensure that when the field is updated, all the other copies of that field are updated at the same time. This becomes difficult when the copies of the field are held in separate files, which are used by separate programs. Another way to solve this problem is to store all data in one place and allow each application to access it. This leads to more consistent information. This also leads to less data redundancy.

5.3 EXCEPTION HANDLING :

A java exception is an object that describes an exceptional condition that has occurred in a piece of code. When exceptional condition arises, an object representing that exception is created and thrown in the method that caused the error. If exceptions are not handled & caught properly the java interpreter displays a message & terminates the program. If we want the remaining code to be executed then we have to catch the error condition thrown by the exception object .This is called " *EXCEPTION HANDLING* ".The purpose of this is to detect & report events. There are 5 keywords available in exception handling.

1. try
2. catch
3. throw

4. throws

5. finally

In this project only the first 2 keywords are used often.

The usage of try block is to simply enclose the code that we want to monitor. Immediately following the try block is the catch block.

The usage of catch block is that it specifies the exception type that we wish to catch.

SCOPE FOR FUTURE DEVELOPMENT

CHAPTER - 6

>>> <<<

SCOPE FOR FUTURE DEVELOPMENT

The development phase can be accomplished since this a website construction project .we can incorporate the mailing system using SMTP server which right now is handled via.Database.

Some of famous features like online chats can be included which helps to have an direct communication with the concerned persons for clarifying doubts regarding various aspects in the medical field .

The information regarding various medical news can be published via. Internet.The mails handled can be made to interact even with the mobile technology.

Advancements and improvements does not have an end when working with the network technology with which very high security can be established.

CONCLUSION

CHAPTER

>>> CONCLUSION <<<

This is going to be of great help for the public and also one of the most welcoming website project.. This study is apt for the present era of advanced technology of competing world .

The outcome of the valuable effort put forward for the public made it to accustom with the existing networking environment.

The purpose for which the project was carried out successfully full fills the needs put forward.

The interrupts faced while working out the project helped out to gain extra knowledge about the networking field and brought out to face the hurdles enthusiastically which gave interest to know and view the things in a different manner.

APPENDIXES

>>> <<<

APPENDIX - A

LIST OF TABLES

9	***Rep(Reports)***	*31*
10	***Test (Request Appointment)***	*32*

>>> # APPENDIX - B <<<

LIST OF SCREENS

SYNOPSIS

SYNOPSIS

The medical field has achieved a tremendous advancement in the field of diagnostics especially in the Diagnostic centre. The managerial activities were efficiently carried out by manpower with stress and strain, until last decade.

The project took up mainly focuses on automatizing various activities handled in the diagnostics centre simplifies to the core. The main intention behind is to accustom with the existing technology ie., Networking through web.

The existing system in a diagnostic centre is that, it records various details of the client, tests to be carried out and reports in the written format by in person only possible. All such activities does not compensate with the existing computerized evolution.

The project proposed eliminates various hurdles undergone by the client. The client can approach the diagnostic centre through internet, makes a registration and views the report online within no time. The project also provides the user with the various diagnostics details accomplished in the diagnostic centre .The Doctors also sign up and view the reports of the client referred by them.

CONTENTS

TABLE OF CONTENTS

BIBLIOGRAPHY

>>> BIBLIOGRAPHY <<<

BOOKS REFERRED :

- Java 2 :A Complete Reference

  Patric Naughton & Herbert schildt TMH publications.

- Database Programming with JDBC

  O'Reilly Publications.

- HTML Complete

  Bpb Publications.

- Professional JSP

  Worx press Ltd.

- Web Design : A Complete Reference

  Powell Thomas A

WEBSITES

- www.jspin.com
- www.java.sun.com
- www.jspinsider.com
- www.servlets.com
- www.jguru.com

CERTIFICATE

Department of Computer Science and Engineering
Kumaraguru College of Technology
Coimbatore - 641006.

This is to certify that the project work entitled
"**B2B E-COMMERCE SOLUTION**

AND

RIJNDAEL INTERFACE"
has been submitted by

Ms.Lakshmi Priya.V	9827S0011
Ms.Ramya.R	9827S0023
Ms.Saritha.P	9827S0025

In partial fulfillment of the award of the degree of
Bachelor of Engineering
Information Technology
of Bharathiar University , Coimbatore
during the academic year 2001 - 2002

Rajini K
Guide

S. Thangasamy 15/3/02
Head of the Department

Certified that the candidate was examined by us in the Project Work Viva Voce Examination held on 18-03-2002 and the University Register Number was 9827S0011, 9827S0023, 9827S0025

Rajini K 18-3-02
Internal Examiner

18/3/2002
External Examiner

Dedicated to

our beloved parents

ACKNOWLEDGEMENT

Any endeavor over a long period can be successful only by the advice and support of many well-wishers. We avail this opportunity to express our gratitude and appreciation of all of them.

We express our profound respect and sincere gratitude to our Principal **Dr. K.K.Padmanaban, B.Sc. (Engg), M.Tech, Ph.D.,** for having provided the necessary facilities to complete this project.

We are greatly indebted to our Head of the Department **Dr.S.Thangasamy, B.E (Hons), Ph.D.,** Computer Science and Engineering, Kumaraguru College of Technology, for our source of inspiration and encouragement rendered by him.

We are greatly privileged to express our deep sense of gratitude to our guide **Ms.Rajini B.E.,** Senior lecturer, Department of Computer Science and Engineering, who has been a motivating force behind all our deeds.

We wish to extend our gratitude to **Mr.Mahesh Kumar,** Member – Technical Staff, Websea e-Guide, Chennai for allowing us to carry out this project work at his concern and guiding us to complete this project successfully.

Last but not the least, we wish to thank all our friends for their continuous support and encouragement during the course of this project.

Declaration

We the students involved in this project specified, declare that we have done the project to be submitted to Kumaraguru college of technology for the partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Information Technology under the guidance of Ms.Rajini.S B.E.

Ms.Lakshmi Priya.V

Ms.Ramya.R

Ms.Saritha.P

Place: Coimbatore
Date: 15-3-2002

Countersigned

Project 1

B2B E-Commerce Solution

CONTENTS

Synopsis

1.Synopsis

Active Server Pages (ASP) technology is a very simple idea of mixing HTML and code to make Web sites a powerful tool.ASP lets the user use the power of a Web server to process user requests and provide dynamic, individualized, content based on logic, file and database data and also process the user's individualized data.

Web application consists of series of short conversation between a Web server and a browser. Each browser initiates requests and the server responds immediately. ASP lets the server to treat all the users as a unique entity even though they run on the same machine.

ASP provides:

- A way to save individualized data for every user
- Access to the file system
- Access to the databases
- A means to launch and control any Component Object Model(COM) component

The project deals with the BUY AND SELL option of a dot com product dealing with the air-condition and refrigeration. The project allows the merchants all around the world to register themselves within the site if they have any product to be sold or purchased. Thus after registering they can get into the transaction with the other merchants.

The inputs from the other merchants are got and published in the site and hence those who are in search of a product can identify the product easily. Then the whole transaction of buying and selling takes place without revealing any identity of the buyer or the seller. After the transactions are over and after the site author getting the commission, the identities are revealed.

Introduction

2.Introduction

2.1 Knowing about ASP

ASP is not a single language all by itself. It depends on other collection of tools, languages, techniques and technologies. These didn't spring all of a sudden and they have their own history. Hypertext Markup Language is one tool on which the ASP basis is made of. HTML first took over the whole technology but with the want for personalized pages and applications more than information, ASP technology took over than the HTML.

BENEFITS OF ASP

ASP is language independent:

ASP technology does not depend on a single scripting language. It works with any scripting language, which is compatible with Microsoft Scripting Host Requirements. It can even work with multiple scripting languages on the same page.

ASP is for Non-programmers:

ASP technology need not have any knowledge for creating ASP pages but it does rally need the user to have some knowledge if the user is into any application. This knowledge does not come by just learning or knowledge but only through experience with the ASP technology.

MAJOR ADVANTAGES OF ASP

- ASP code rsides in text files
- ASP code times out after 90 seconds of the start of application
- ASP code is server-safe
- ASP code doesn't require registration
- The applications are usually small
- The applications can be upgraded without stopping IIS

2.2 About IIS

Internet Information Server takes over the whole Web server market invincibly.IIS on NT is much faster than any other server on any other platform.Also there any three more reasons to justify IIS than any other server. First it shares the database connections and launch, control, and participate in transactions. Second,IIS is programmable. Last but not the least, it can be extended and customized over the internet.

FEATURES OF IIS

IIS provides integrated security:

IIS lets the user setup security restrictions on a site-by-site basis.

IIS provides access to content:

IIS natively understands how to treat most common window file formats.

IIS provides an interface for COM:

The user can control many parts of IIS using COM.

Buy and Sell Option

3.Buy and Sell Option

3.1 acrefonline.com

acrefonline.com provides the best way for the merchants to market their products. It provides an opportunity for the air-conditioning and refrigeration mass to sell or buy their products staying at the desktop. The main feature of this project is its reliability where the merchants can rely on the information provided. Also, they can trust on the security of their identity so that there are no partial fulfillment in the site.

There are a few steps to be remembered before we can go into the implementation of the project:

1. Ask the details from the merchants who enter this module and also ask them about their product description for selling or buying.
2. Write the details into the database and keep refreshing the database as and when there are any inclusions in the database.
3. Send a mail to the merchant who is interested in buying or selling of any product. Register the person if there is a reply to the mail.
4. After registering the person, the merchant is to be given a username and a password for his entry into the module.
5. He is allowed to search for his product description or any other detail regarding the product. An important point to be noted is that both the buyer and the seller will not know each other's identity until the transaction is completed.
6. If there is a match in his product description and if the merchant is interested in buying or selling, the transaction takes place and there is a commission for the site manager also.
7. Thus after the whole transaction is over the buyer and the seller are notified about their identities.

3.2 Implementation

The implementation of this project is done in ASP running on Windows 2000 server. An important part is the presence of the IIS that helps the applications to act as a server. The project is implemented in three modules as described in the forth-coming sections.

Registration and Login

In this module which is the first module of the project, the registration and the login takes place. The merchant first views the web site and when he enters the Buy and Sell module, he is asked to register his name and his mail-id along with the product description and the expected price for the product.
The user is then asked if the product is for selling or buying.

Then he is intimated that he has been registered to the site and is asked to mail back for the confirmation of his registration. After receiving the mail from the user, a username and a password is given to the user, which is unique for each user.

Insertion and Updating

As and when the process of registering and login is proceeding there is a parallel process of inserting and updating the databases at the backend. The table at the back end should contain
Name
Address
Tel No
Fax No
Email Id
Brief Product Description (Text 250)
Desired Price
Validity Period (i.e. 7 days, 14 days, 21 days, 1 month)
This database is common to both the buyer and the seller. Also there should be another database in which all the transactions should be noted. The second database will contain about all the products that are sold or bought and if there is a shortage in the product then merchant who has got the availability is notified and thus the product is obtained.

Successful login

When the user logs in the module there will be a report of all the merchants available in the field. He can also find a search engine where he can give the product description and the quoted price and thus search for the best-suited product satisfying his specifications. There is flag that will help the site manager to know if the login has been a successful one or not. If the flag goes to successful position then it means that there has been a successful login else there is a mistake in the login session of the user. Thus the project is best explained in these three modules.

Coding

4. Coding

Page 1:

```
<%@ Language=VBScript %>
<html>

<head>
<meta NAME="GENERATOR" Content="Microsoft FrontPage 3.0">
<title></title>
</head>

<body>

<p> </p>

<p align="center"><strong><em>Select Page for the buyer or
sellers</em></strong></p>

<p> </p>

<form method="get" action="buy.asp">
  <div align="center"><center><p><input id="submit" name="submit1"
  style="HEIGHT: 24px; LEFT: 212px; TOP: 129px; WIDTH: 57px" type="submit"
value="Buy"></p>
  </center></div>
</form>

<form id="form1" name="form1" action="sell.asp">
  <div align="center"><center><p><input id="submit1" name="submit1"
  style="HEIGHT: 24px; LEFT: 216px; TOP: 172px; WIDTH: 54px" type="submit"
value="sell"></p>
  </center></div>
</form>

<p align="center"> </p>
</body>
</html>
```

Page 2:

```
<%@ Language=VBScript %>
<html>

<head>
<meta NAME="GENERATOR" Content="Microsoft FrontPage 3.0">
<title></title>
</head>

<body bgcolor="#5978af">

<p align="left"><img alt src="file:///C:/Inetpub/wwwroot/preethy/logonew"
style="WIDTH: 210px; HEIGHT: 56px" height="56" width="516"> </p>

<p align="center"><b><i><font size="6">Registration Page</font></i></b></p>

<form action="register_work.asp" method="post">
  <div align="center"><center><p> </p>
  </center></div><div
align="center"><center><p><strong><em>      &n
bsp;  
  </p>
  </center></div><div align="center"><center><table border="1" width="572">
    <tr>
      <td width="113"><div align="center"><center><p>Name</td>
      <td width="443"><strong><em><input id="text2" style="WIDTH: 441px; HEIGHT:
22px" size="24"
      name="name"></em></strong> </td>
    </tr>
  </table>
  </center></div><div align="center"><center><table cellSpacing="1" cellPadding="1"
  width="578" border="1">
    <tr>
      <td width="632"><div
align="center"><center><p><em><strong>Address:</strong></em></p>
      </center></div><p><em><strong><textarea
      style="LEFT: 57px; WIDTH: 574px; TOP: 2px; HEIGHT: 76px" name="address"
rows="3" cols="75"></textarea></strong></em></td>
    </tr>
  </table>
  </center></div></em></strong><div align="center"><center><p> </p>
  </center></div><div align="center"><center><table border="1">
    <tr>
      <strong><em><td><strong><em>Telephone Number</em></strong>
</em></strong></td>
      <td><strong><em><input id="text2" style="WIDTH: 441px; HEIGHT: 22px"
size="24"
      name="teleno"></em></strong> </td>
    </tr>
    <tr>
```

Bibliography

6.Bibliography

1. Mastering Active Server Pages by Russell Jones
2. Understanding and using ASP by A.Russell Jones

Conclusion

5.Conclusion

The project has been executed successfully and is ready to be launched. The project has been running effectively and efficiently as expected by the company. Thus the project helped us to know about the working of the ASP applications and the running of the IIS. The project also helped us to learn how to be in a working environment. This is a stepping stone in the voracious journey of the students in search for knowledge. This will help us to build a more complex quest for the knowledge in internet.

```
    <td width="777"><div align="center"><center><p><strong><em><input type="radio"
value="Buy"
    name="buysell" checked>

Buy             
             
    <input type="radio" value="Sell" name="buysell">Sell</em></strong></td>
   </tr>
  </table>
  </center></div><div align="center"><center><p> </p>
  </center></div><div align="center"><center><p><input type="submit" value="Register"
  name="sub" style="WIDTH: 131px; COLOR: #5978af; HEIGHT: 38px;
BACKGROUND-COLOR: black"
  size="35"></p>
  </center></div><div align="center"><center><p> </p>
  </center></div>
</form>
</body>
</html>
```

```
<strong><em><td><strong><em>Fax Number</em></strong> </em></strong></td>
 <td><strong><em><input id="text3" style="WIDTH: 441px; HEIGHT: 22px" size="24"
 name="faxno"></em></strong> </td>
  </tr>
  <tr>
   <strong><em><td><strong><em>E-Mail Address</em></strong>
</em></strong></td>
   <td><strong><em><input id="text4" style="WIDTH: 441px; HEIGHT: 22px" size="24"
    name="email"></em></strong> </td>
   </tr>
   </table>
  </center></div><div align="center"><center><p> </p>
   </center></div><div align="center"><center><table cellSpacing="1" cellPadding="1"
  width="585" border="1">
   <tr>
    <strong><em><td width="588"><div align="center"><center><p>Product
Description:       <strong><em>  </em></strong></p>
     </center></div><div align="left"><p><strong><em><textarea
     style="LEFT: 57px; WIDTH: 579px; TOP: 2px; HEIGHT: 76px" name="proddesc"
rows="3"
     cols="75"></textarea></em></strong> </em></strong></td>
   </tr>
  </table>
   </center></div><div align="center"><center><p> </p>
  </center></div><div align="center"><center><table cellSpacing="1" cellPadding="1"
  width="593" border="1">
   <tr>
    <strong><em><td width="395"><div
align="center"><center><p><strong><em>Validity
     Period:   </em></strong></em></strong></td>
     <td width="771"><div align="center"><center><p>  <strong><em><input
type="radio"
     value="1 Week" name="valperiod" checked>1

Week            &nbs
p;   <input
     type="radio" value="2 Weeks" name="valperiod"> 2

Weeks            &nb
sp;      
     <input type="radio" value="1 Month" name="valperiod">1
Month    </em></strong></td>
   </tr>
  </table>
  </center></div><div align="center"><center><p> </p>
  </center></div><div align="center"><center><table cellSpacing="1" cellPadding="1"
  width="594" border="1">
   <tr>
    <strong><em><td width="391"><div
align="center"><center><p><strong><em>Select Your
     Criteria      </em></strong></em></strong></td>
```

Project 2

Rijndael Interface

CONTENTS

7. Implementation Issues

7.1 Key length requirements

7.2 Key restrictions

7.3 Parameterization of key length ,block size and round number

8. Conclusion

9. Bibliography

10. Appendix

Synopsis

1.SYNOPSIS

Java is a blend of the best elements of its rich heritage combined with the innovative concepts required by its unique environment. In short Java is a small, simple, safe, object-oriented, interpreted or dynamically optimized, byte-coded, architecture-neutral, garbage-collected, multithreaded programming language with a strongly typed exception-handling mechanism for writing distributed, dynamically extensible programs.

Innumerous Java applications are being developed in recent times and the community of Java programmers is constantly on the rise. Though the major development packages for Java were created with fundamental console behavior, recently a lot of building tools have propped up.

The major challenge to implement the general architecture is how to trade off among usability, flexibility, security and performance concerns. But the present day networks provides the minimal amount of the above features. This project tries to provide security for the data to be sent through the insecure medium of net. This project deals with the Encryption and Decryption of the data passed through the net.

The project uses the Rijndael algorithm which inturn uses the concept of Triple DES . The project uses simple mathematics and encrypts the given file or directory into an unreadable form which can be sent. At the receiver end the same project is used to decrypt the data using the same algorithm. Thus the project works with the help of Java software.

Introduction

2. Introduction

This standard specifies the **Rijndael** algorithm [3], a symmetric block cipher that can process **data blocks** of **128 bits**, using cipher **keys** with lengths of **128**, **192**, and **256 bits**. Rijndael was designed to handle additional block sizes and key lengths, however they are not adopted in this standard.

Throughout the remainder of this standard, the algorithm specified herein will be referred to as "the AES algorithm." The algorithm may be used with the three different key lengths indicated above, and therefore these different "flavors" may be referred to as "AES-128", "AES-192", and "AES-256".

This specification includes the following sections:

1. Definitions of terms, acronyms, and algorithm parameters, symbols, and functions;
2. Notation and conventions used in the algorithm specification, including the ordering and numbering of bits, bytes, and words;
3. Mathematical properties that are useful in understanding the algorithm;
4. Algorithm specification, covering the key schedule, encryption, and decryption routines;
5. Implementation issues, such as key length support, keying restrictions, and additional block/key/round sizes.

The standard concludes with several appendices that include Object Identifiers (OIDs) for use with implementations of the AES algorithm, step-by-step examples for Key Expansion and the Cipher, example vectors for the Cipher and Inverse Cipher, and a list of references.

P. 681

Definitions

3. Definitions

3.1 Glossary of Terms and Acronyms

The following definitions are used throughout this standard:

AES	Advanced Encryption Standard
Affine Transformation	A transformation consisting of multiplication by a matrix followed by the addition of a vector.
Array	An enumerated collection of identical entities (e.g., an array of bytes).
Bit	A binary digit having a value of 0 or 1.
Block	Sequence of binary bits that comprise the input, output, State, and Round Key. The length of a sequence is the number of bits it contains. Blocks are also interpreted as arrays of bytes.
Byte	A group of eight bits that is treated either as a single entity or as an array of 8 individual bits.
Cipher	Series of transformations that converts plaintext to ciphertext using the Cipher Key.
Cipher Key	Secret, cryptographic key that is used by the Key Expansion routine to generate a set of Round Keys; can be pictured as a rectangular array of bytes, having four rows and ***Nk*** columns.
Ciphertext	Data output from the Cipher or input to the Inverse Cipher.
Inverse Cipher	Series of transformations that converts ciphertext to plaintext using the Cipher Key.
Key Expansion	Routine used to generate a series of Round Keys from the Cipher Key.
Plaintext	Data input to the Cipher or output from the Inverse Cipher.
Rijndael	Cryptographic algorithm specified in this Advanced Encryption Standard (AES). Suggested alternatives for pronunciation include "**Reign Dahl**", "**Rain Doll**", and "**Rhine Dahl**."
Round Key	Round keys are values derived from the Cipher Key using the Key Expansion routine; they are applied to the State in the Cipher and Inverse Cipher.
State	Intermediate Cipher result that can be pictured as a rectangular array of bytes, having four rows and ***Nb*** columns.
S-box	Non-linear substitution table used in several byte substitution transformations and in the Key Expansion routine to perform a one-for-one substitution of a byte value.

Notations and Conventions

4. Notation and Conventions

4.1 Inputs and Outputs

The **input** and **output** for the AES algorithm each consist of **sequences of 128 bits** (digits with values of 0 or 1). These sequences will sometimes be referred to as **blocks** and the number of bits they contain will be referred to as their length. The **Cipher Key** for the AES algorithm is a **sequence of 128, 192 or 256 bits**. Other input, output and Cipher Key lengths are not permitted by this standard.

The bits within such sequences will be numbered starting at zero and ending at one less than the sequence length (block length or key length). The number i attached to a bit is known as its index and will be in one of the ranges $0 \le i < 128$, $0 \le i < 192$ or $0 \le i < 256$ depending on the block length and key length (specified above).

4.2 Bytes

The basic unit for processing in the AES algorithm is a **byte,** a sequence of eight bits treated as a single entity. The input, output and Cipher Key bit sequences described in Sec. 4.1 are processed as arrays of bytes that are formed by dividing these sequences into groups of eight contiguous bits to form arrays of bytes (see Sec. 4.3). For an input, output or Cipher Key denoted by a, the bytes in the resulting array will be referenced using one of the two forms, a_n or $a[n]$, where n will be in one of the following ranges:

Key length = 128 bits, $0 \le n < 16$; Block length = 128 bits, $0 \le n < 16$;

Key length = 192 bits, $0 \le n < 24$;

Key length = 256 bits, $0 \le n < 32$.

All byte values in the AES algorithm will be presented as the concatenation of its individual bit values (0 or 1) between braces in the order $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$. These bytes are interpreted as finite field elements using a polynomial representation:

$$b_7 x^7 + b_6 x^6 + b_5 x^5 + b_4 x^4 + b_3 x^3 + b_2 x^2 + b_1 x + b_0 = \sum_{i=0}^{7} b_i x^i \,. \qquad (3.1)$$

For example, {01100011} identifies the specific finite field element $x^6 + x^5 + x + 1$.

It is also convenient to denote byte values using hexadecimal notation with each of two groups of four bits being denoted by a single character as in Fig. 1.

Bit Pattern	Character
0000	0
0001	1
0010	2
0011	3

Bit Pattern	Character
0100	4
0101	5
0110	6
0111	7

Bit Pattern	Character
1000	8
1001	9
1010	a
1011	b

Bit Pattern	Character
1100	c
1101	d
1110	e
1111	f

Figure 1. Hexadecimal representation of bit patterns.

Hence the element {01100011} can be represented as {63}, where the character denoting the four-bit group containing the higher numbered bits is again to the left.

Some finite field operations involve one additional bit (b_8) to the left of an 8-bit byte. Where this extra bit is present, it will appear immediately to the left of the left brace; for example, a 9-bit sequence will be presented as either 1{00011011} or 1{1b}.

4.3 Arrays of Bytes

Arrays of bytes will be represented in the following form:

$$a_0 a_1 a_2 \ldots a_{15}$$

The bytes and the bit ordering within bytes are defined from the 128-bit input sequence

$$input_0\ input_1\ input_2\ \ldots\ input_{126}\ input_{127}$$

as follows:

$$a_0 = \{input_0, input_1, \ldots, input_7\};$$

$$a_1 = \{input_8, input_9, \ldots, input_{15}\};$$

$$\vdots$$

$$a_{15} = \{input_{120}, input_{121}, \ldots, input_{127}\}.$$

The pattern can be extended to longer sequences (i.e., for 192- and 256-bit keys), so that, in general,

$$a_n = \{input_{8n}, input_{8n+1}, \ldots, input_{8n+7}\}. \qquad (3.2)$$

Taking Sections 4.2 and 4.3 together, Fig. 2 shows how bits within each byte are numbered.

Input bit sequence	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	...
Byte number	0								1								2								...
Bit numbers in byte	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	...

Figure 2. Indices for Bytes and Bits.

4.4 The State

Internally, the AES algorithm's operations are performed on a two-dimensional array of bytes called the **State**. The State consists of four rows of bytes, each containing ***Nb*** bytes, where ***Nb*** is the block length divided by 32. In the State array denoted by the symbol s, each individual byte has two indices, with its row number r in the range $0 \le r < 4$ and its column number c in the range $0 \le c < \boldsymbol{Nb}$. This allows an individual byte of the State to be referred to as either $s_{r,c}$ or $s[r,c]$. For this standard, ***Nb***=4, i.e., $0 \le c < 4$ (also see Sec. 7.3).

At the start of the Cipher and Inverse Cipher described in Sec. 6, the input – the array of bytes $in_0, in_1, \ldots in_{15}$ – is copied into the State array as illustrated in Fig. 3. The Cipher or Inverse Cipher operations are then conducted on this State array, after which its final value is copied to the output – the array of bytes $out_0, out_1, \ldots out_{15}$.

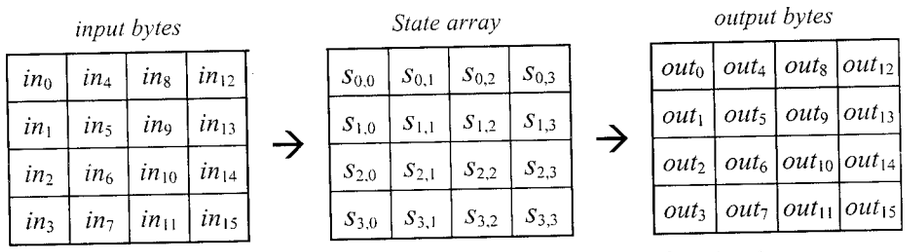


Figure 3. State array input and output.

Hence, at the beginning of the Cipher or Inverse Cipher, the input array, *in*, is copied to the State array according to the scheme:

$$s[r, c] = in[r + 4c] \qquad \text{for } 0 \le r < 4 \text{ and } 0 \le c < Nb, \tag{3.3}$$

and at the end of the Cipher and Inverse Cipher, the State is copied to the output array *out* as follows:

$$out[r + 4c] = s[r, c] \qquad \text{for } 0 \le r < 4 \text{ and } 0 \le c < Nb. \tag{3.4}$$

Note that the four bytes in the columns of the State array form 32-bit **words**, where the row number r provides an index for the four bytes within each word.

For example, if the input block consists of the following bytes:

$$a_0 a_1 a_2 \ldots a_{15},$$

then the block is mapped into the State as in Fig. 4 below.

a_0	a_4	a_8	a_{12}
a_1	a_5	a_9	a_{13}
a_2	a_6	a_{10}	a_{14}
a_3	a_7	a_{11}	a_{15}

Figure 4. Example mapping of an input block into the State.

Mathematical Preliminaries

5. Mathematical Preliminaries

All bytes in the AES algorithm are interpreted as finite field elements using the notation introduced in Sec. 5.2. Finite field elements can be added and multiplied, but these operations are different from those used for numbers. The following subsections introduce the basic mathematical concepts needed for Sec. 7.

5.1 Addition

The addition of two elements in a finite field is achieved by "adding" the coefficients for the corresponding powers in the polynomials for the two elements. The addition is performed with the XOR operation (denoted by $\oplus$) - i.e., modulo 2 - so that $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, and $0 \oplus 0 = 0$. Consequently, subtraction of polynomials is identical to addition of polynomials.

Alternatively, addition of finite field elements can be described as the modulo 2 addition of corresponding bits in the byte. For two bytes $\{a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0\}$ and $\{b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0\}$, the sum is $\{c_7 c_6 c_5 c_4 c_3 c_2 c_1 c_0\}$, where each $c_i = a_i \oplus b_i$ (i.e., $c_7 = a_7 \oplus b_7$, $c_6 = a_6 \oplus b_6$, ... $c_0 = a_0 \oplus b_0$).

For example, the following expressions are equivalent to one another:

$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$ (polynomial notation);

$\{01010111\} \oplus \{10000011\} = \{11010100\}$ (binary notation);

$\{57\} \oplus \{83\} = \{d4\}$ (hexadecimal notation).

5.2 Multiplication

In the polynomial representation, multiplication in GF(2^8) (denoted by $\bullet$) corresponds with the multiplication of polynomials modulo an **irreducible polynomial** of degree 8. A polynomial is irreducible if its only divisors are one and itself. **For the AES algorithm, this irreducible polynomial is**

$$m(x) = x^8 + x^4 + x^3 + x + 1, \tag{4.1}$$

or $1\{1b\}$ in hexadecimal notation.

For example, $\{57\} \bullet \{83\} = \{c1\}$, because

$$\begin{aligned}(x^6 + x^4 + x^2 + x + 1)\,(x^7 + x + 1) &= x^{13} + x^{11} + x^9 + x^8 + x^7 + \\ &\quad x^7 + x^5 + x^3 + x^2 + x + \\ &\quad x^6 + x^4 + x^2 + x + 1 \\ &= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1\end{aligned}$$

and

$$\begin{aligned}x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \ \text{modulo}\ (x^8 + x^4 + x^3 + x + 1) \\ = \quad x^7 + x^6 + 1.\end{aligned}$$

The modular reduction by $m(x)$ ensures that the result will be a binary polynomial of degree less than 8, and thus can be represented by a byte. Unlike addition, there is no simple operation at the byte level that corresponds to this multiplication.

The multiplication defined above is associative, and the element {01} is the multiplicative identity. For any non-zero binary polynomial $b(x)$ of degree less than 8, the multiplicative inverse of $b(x)$, denoted $b^{-1}(x)$, can be found as follows: the extended Euclidean algorithm [8] is used to compute polynomials $a(x)$ and $c(x)$ such that

$$b(x)a(x) + m(x)c(x) = 1. \tag{4.2}$$

Hence, $a(x) \bullet b(x) \bmod m(x) = 1$, which means

$$b^{-1}(x) = a(x) \bmod m(x). \tag{4.3}$$

Moreover, it holds that $a(x) \bullet (b(x) + c(x)) = a(x) \bullet b(x) + a(x) \bullet c(x)$.

It follows that the set of 256 possible byte values, with XOR used as addition and the multiplication defined as above has the structure of the finite field GF(2^8).

5.2.1 Multiplication by *y*

Multiplying the binary polynomial defined in equation (3.1) with the polynomial x results in

$$b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x. \tag{4.4}$$

The result $x \bullet b(x)$ is obtained by reducing the above result modulo $m(x)$, as defined in equation (4.1). If $b_7 = 0$, the result is already in reduced form. If $b_7 = 1$, the reduction is accomplished by subtracting (i.e., XORing) the polynomial $m(x)$. It follows that multiplication by x (i.e., {00000010} or {02}) can be implemented at the byte level as a left shift and a subsequent conditional bitwise XOR with {1b}. This operation on bytes is denoted by `xtime()`. Multiplication by higher powers of x can be implemented by repeated application of `xtime()`. By adding intermediate results, multiplication by any constant can be implemented.

For example, {57} • {13} = {fe} because

$$\begin{aligned}
\{57\} \bullet \{02\} &= \texttt{xtime}(\{57\}) = \{ae\} \\
\{57\} \bullet \{04\} &= \texttt{xtime}(\{ae\}) = \{47\} \\
\{57\} \bullet \{08\} &= \texttt{xtime}(\{47\}) = \{8e\} \\
\{57\} \bullet \{10\} &= \texttt{xtime}(\{8e\}) = \{07\},
\end{aligned}$$

thus,

$$\begin{aligned}
\{57\} \bullet \{13\} &= \{57\} \bullet (\{01\} \oplus \{02\} \oplus \{10\}) \\
&= \{57\} \oplus \{ae\} \oplus \{07\} \\
&= \{fe\}.
\end{aligned}$$

5.3 Polynomials with Coefficients in GF(2^8)

Four-term polynomials can be defined - with coefficients that are finite field elements - as:

$$a(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0 \tag{4.5}$$

which will be denoted as a word in the form $[a_0, a_1, a_2, a_3]$. Note that the polynomials in this section behave somewhat differently than the polynomials used in the definition of finite field elements, even though both types of polynomials use the same indeterminate, x. The coefficients in this section are themselves finite field elements, i.e., bytes, instead of bits; also, the multiplication of four-term polynomials uses a different reduction polynomial, defined below. The distinction should always be clear from the context.

To illustrate the addition and multiplication operations, let

$$b(x) = b_3 x^3 + b_2 x^2 + b_1 x + b_0 \tag{4.6}$$

define a second four-term polynomial. Addition is performed by adding the finite field coefficients of like powers of x. This addition corresponds to an XOR operation between the corresponding bytes in each of the words – in other words, the XOR of the complete word values.

Thus, using the equations of (4.5) and (4.6),

$$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x + (a_0 \oplus b_0) \tag{4.7}$$

Multiplication is achieved in two steps. In the first step, the polynomial product $c(x) = a(x) \bullet b(x)$ is algebraically expanded, and like powers are collected to give

$$c(x) = c_6 x^6 + c_5 x^5 + c_4 x^4 + c_3 x^3 + c_2 x^2 + c_1 x + c_0 \tag{4.8}$$

where

$$\begin{aligned}
c_0 &= a_0 \bullet b_0 & c_4 &= a_3 \bullet b_1 \oplus a_2 \bullet b_2 \oplus a_1 \bullet b_3 \\
c_1 &= a_1 \bullet b_0 \oplus a_0 \bullet b_1 & c_5 &= a_3 \bullet b_2 \oplus a_2 \bullet b_3 \\
c_2 &= a_2 \bullet b_0 \oplus a_1 \bullet b_1 \oplus a_0 \bullet b_2 & c_6 &= a_3 \bullet b_3 \\
c_3 &= a_3 \bullet b_0 \oplus a_2 \bullet b_1 \oplus a_1 \bullet b_2 \oplus a_0 \bullet b_3 .
\end{aligned} \tag{4.9}$$

The result, $c(x)$, does not represent a four-byte word. Therefore, the second step of the multiplication is to reduce reduce $c(x)$ modulo a polynomial of degree 4; the result can be reduced to a polynomial of degree less than 4. **For the AES algorithm, this is accomplished with the polynomial $x^4 + 1$**, so that

$$x^i \bmod (x^4 + 1) = x^{i \bmod 4}. \tag{4.10}$$

The modular product of $a(x)$ and $b(x)$, denoted by $a(x) \otimes b(x)$, is given by the four-term polynomial $d(x)$, defined as follows:

$$d(x) = d_3 x^3 + d_2 x^2 + d_1 x + d_0 \tag{4.11}$$

with

$$\begin{aligned}
d_0 &= (a_0 \bullet b_0) \oplus (a_3 \bullet b_1) \oplus (a_2 \bullet b_2) \oplus (a_1 \bullet b_3) \\
d_1 &= (a_1 \bullet b_0) \oplus (a_0 \bullet b_1) \oplus (a_3 \bullet b_2) \oplus (a_2 \bullet b_3) \\
d_2 &= (a_2 \bullet b_0) \oplus (a_1 \bullet b_1) \oplus (a_0 \bullet b_2) \oplus (a_3 \bullet b_3) \\
d_3 &= (a_3 \bullet b_0) \oplus (a_2 \bullet b_1) \oplus (a_1 \bullet b_2) \oplus (a_0 \bullet b_3)
\end{aligned} \tag{4.12}$$

When $a(x)$ is a fixed polynomial, the operation defined in equation (4.11) can be written in matrix form as:

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{4.13}$$

Because $x^4 + 1$ is not an irreducible polynomial over GF(2^8), multiplication by a fixed four-term polynomial is not necessarily invertible. However, the AES algorithm specifies a fixed four-term polynomial that *does* have an inverse (see Sec. 7.1.3 and Sec. 7.3.3):

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \tag{4.14}$$

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}. \tag{4.15}$$

Another polynomial used in the AES algorithm (see the **`RotWord()`** function in Sec. 7.2) has $a_0 = a_1 = a_2 = \{00\}$ and $a_3 = \{01\}$, which is the polynomial x^3. Inspection of equation (4.13) above will show that its effect is to form the output word by rotating bytes in the input word. This means that $[b_0, b_1, b_2, b_3]$ is transformed into $[b_1, b_2, b_3, b_0]$.

Algorithm Specifications

6. Algorithm Specification

For the AES algorithm, **the length of the input block, the output block and the State is 128 bits.** This is represented by ***Nb*** = 4, which reflects the number of 32-bit words (number of columns) in the State (also see Sec. 7.3).

For the AES algorithm, **the length of the Cipher Key, *K*, is 128, 192, or 256 bits.** The key length is represented by ***Nk*** = 4, 6, or 8, which reflects the number of 32-bit words (number of columns) in the Cipher Key.

The only Key-Block-Round combinations that conform to this standard are given in Fig. 5:

	Key Length *(Nk words)*	**Block Size** *(Nb words)*	**Number of Rounds** *(Nr)*
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Figure 5. Key-Block-Round Combinations.

For both its Cipher and Inverse Cipher, the AES algorithm uses a round function that is composed of four different byte-oriented transformations: 1) byte substitution using a substitution table (S-box), 2) shifting rows of the State array by different offsets, 3) mixing the data within each column of the State array, and 4) adding a Round Key to the State. These transformations (and their inverses) are described in Sec. 6.1.1-6.1.4 and 6.3.1-6.3.4.

The Cipher and Inverse Cipher are described in Sec.6.1 and Sec. 6.3, respectively, while the Key Schedule is described in Sec. 6.2.

6.1 Cipher

At the start of the Cipher, the input is copied to the State array using the conventions described in Sec. 4.4. After an initial Round Key addition, the State array is transformed by implementing a round function 10, 12, or 14 times (depending on the key length), with the final round differing slightly from the first $Nr - 1$ rounds. The final State is then copied to the output as described in Sec. 4.4.

The round function is parameterized using a key schedule that consists of a one-dimensional array of four-byte words derived using the Key Expansion routine described in Sec. 6.2.

The Cipher is described in the pseudo code in Fig. 6. The individual transformations - **SubBytes()**, **ShiftRows()**, **MixColumns()**, and **AddRoundKey()** – process the State and are described in the following subsections. In Fig. 6, the array **w[]** contains the key schedule, which is described in Sec.6.2.

```
Cipher(byte in[4 * Nb], byte out[4 * Nb], word w[Nb * (Nr + 1)])
begin
   byte  state[4,Nb]

   state =  in

   AddRoundKey(state, w)                          // See Sec. 6.1.4

   for round = 1 step 1 to Nr - 1
      SubBytes(state)                             // See Sec. 6.1.1
      ShiftRows(state)                            // See Sec. 6.1.2
      MixColumns(state)                           // See Sec. 6.1.3
      AddRoundKey(state, w + round * Nb)
   end for

   SubBytes(state)
   ShiftRows(state)
   AddRoundKey(state, w + Nr * Nb)

   out = state
end
```

Figure 6. Pseudo Code for the Cipher.[1]

As shown in Fig. 6, all ***Nr*** rounds are identical with the exception of the final round, which does not include the **MixColumns()** transformation.

Appendix C presents an example of the Cipher, showing values for the State array (for ***Nk*** = 4) at the beginning of each round and after the application of each of the four transformations described in the following sections.

6.1.1 SubBytes() Transformation

The **SubBytes()** transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). This S-box (Fig. 8), which is invertible, is constructed by composing two transformations:

1. Take the multiplicative inverse in the finite field GF(2^8), described in Sec. 5; the element {00} is mapped to itself.
2. Apply an affine (over GF(2)) transformation defined by:

$$b_i' = b_i \oplus b_{(i+4) \bmod 8} \oplus b_{(i+5) \bmod 8} \oplus b_{(i+6) \bmod 8} \oplus b_{(i+7) \bmod 8} \oplus c_i \tag{5.1}$$

for $0 \le i < 8$, where b_i is the i^{th} bit of the byte, and c_i is the i^{th} bit of a byte c with the value {63} or {01100011}. Here and elsewhere, a prime on a variable (e.g., b') indicates that the variable is to be updated with the value on the right.

In matrix form, the affine transformation element of the S-box can be expressed as:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}. \tag{5.2}$$

Figure 7 illustrates the effect of the **SubBytes()** transformation on the State.

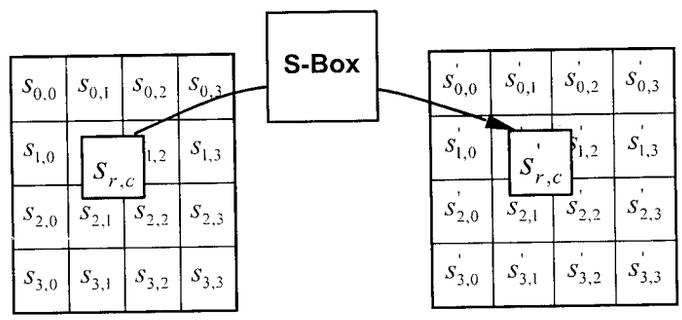


Figure 7. SubBytes() applies the S-box to each byte of the State.

The S-box used in the `SubBytes()` transformation is presented in hexadecimal form in Fig. 8.

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure 8. AES algorithm S-box, showing substitution values for the byte `xy` (in hexadecimal format).

For example, if $s_{1,1} = \{53\}$, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3' in Fig. 8. This would result in $s'_{1,1}$ having a value of $\{ed\}$.

6.1.2 `ShiftRows()` Transformation

In the `ShiftRows()` transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row, Row 0, is not shifted.

Specifically, the `ShiftRows()` transformation proceeds as follows:

$$s'_{r,c} = s_{r,(c+shift(r,Nb)) \bmod Nb} \quad \text{for } 0 < r < 4 \quad \text{and} \quad 0 \le c < \boldsymbol{Nb}, \tag{5.3}$$

where the shift value $shift(r,Nb)$ depends on the row number, r, as follows (recall that $\boldsymbol{Nb} = 4$):

$$shift(1,4) = 1; \quad shift(2,4) = 2; \quad shift(3,4) = 3. \tag{5.4}$$

This has the effect of moving bytes to "lower" positions in the row (i.e., lower values of c in a given row), while the "lowest" bytes wrap around into the "top" of the row (i.e., higher values of c in a given row).

Figure 9 illustrates the `ShiftRows()` transformation.

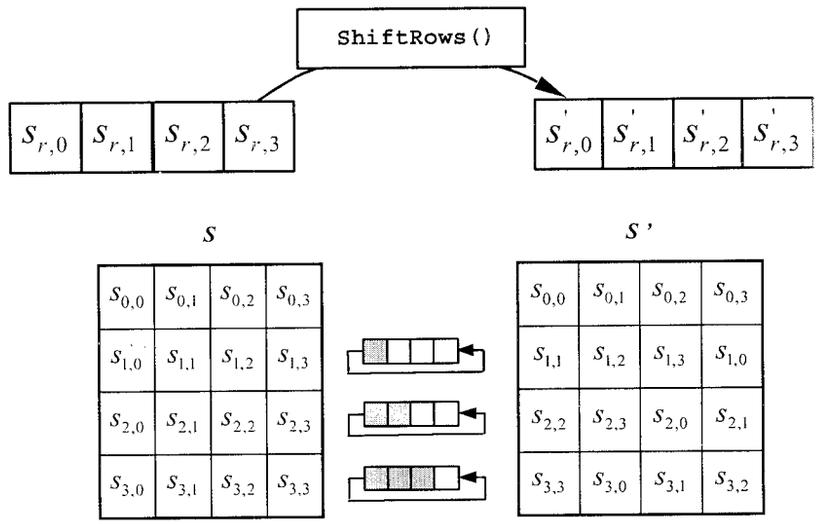


Figure 9. `ShiftRows()` cyclically shifts the last three rows in the State

6.1.3 `MixColumns()` Transformation

The **`MixColumns()`** transformation operates on the State column-by-column, treating each column as a four-term polynomial as described in Sec. 5.2. The columns are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial $a(x)$, given by

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} . \tag{5.5}$$

As described in Sec.5.2, this can be written as a matrix multiplication. Let

$s'(x) = a(x) \otimes s(x)$:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < \boldsymbol{Nb}. \tag{5.6}$$

As a result of this multiplication, the four bytes in a column are replaced by the following:

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}).$$

Figure 10 illustrates the **`MixColumns()`** transformation.

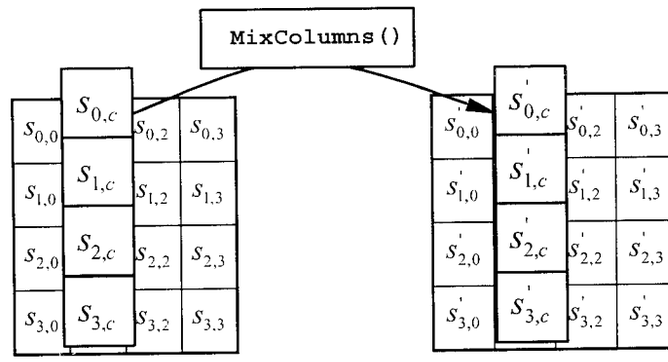


Figure 10. `MixColumns()` operates on the State column-by-column.

6.1.4 `AddRoundKey()` Transformation

In the **`AddRoundKey()`** transformation, a Round Key is added to the State by a simple bitwise XOR operation. Each Round Key consists of ***Nb*** words from the key schedule (described in Sec. 6.2). Those ***Nb*** words are each added into the columns of the State, such that

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus [w_{round*Nb+c}] \quad \text{for } 0 \le c < Nb, \qquad (5.7)$$

where $[w_i]$ are the key schedule words described in Sec. 6.2, and *round* is a value in the range $0 \le round \le Nr$. In the Cipher, the initial Round Key addition occurs when $round = 0$, prior to the first application of the round function (see Fig. 6). The application of the **`AddRoundKey()`** transformation to the ***Nr*** rounds of the Cipher occurs when $1 \le round \le Nr$.

The action of this transformation is illustrated in Fig. 11, where $l = round * Nb$. The byte address within words of the key schedule was described in Sec.4.1.

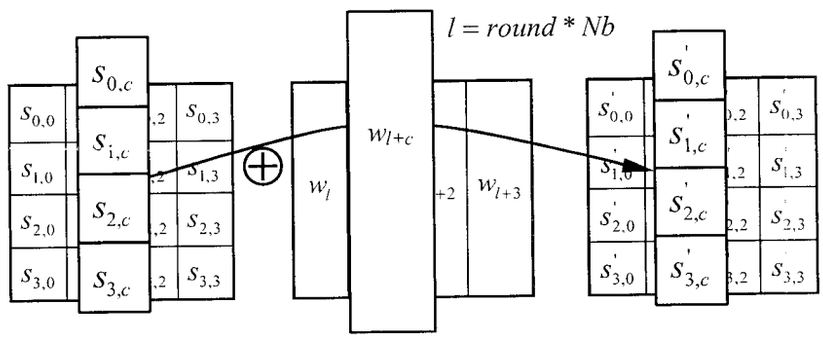


Figure 11. `AddRoundKey()` XORs each column of the State with a word from the key schedule.

6.2 Key Expansion

The AES algorithm takes the Cipher Key, ***K***, and performs a Key Expansion routine to generate a key schedule. The Key Expansion generates a total of ***Nb*** (***Nr*** + 1) words: the algorithm requires an initial set of ***Nb*** words, and each of the ***Nr*** rounds requires ***Nb*** words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted $[w_i]$, with i in the range $0 \le i < Nb(Nr + 1)$.

The expansion of the input key into the key schedule proceeds according to the pseudo code in Fig. 12.

`SubWord()` is a function that takes a four-byte input word and applies the S-box (Sec. 6.1.1, Fig. 8) to each of the four bytes to produce an output word. The function `RotWord()` takes a word $[a_0,a_1,a_2,a_3]$ as input, performs a cyclic permutation, and returns the word $[a_1,a_2,a_3,a_0]$. The round constant word array, `Rcon[i]`, contains the values given by $[x^{i-1},\{00\},\{00\},\{00\}]$, with x^{i-1} being powers of x (x is denoted as $\{02\}$) in the field GF(2^8), as discussed in Sec.5.2 (note that i starts at 1, not 0).

```
KeyExpansion(byte key[4 * Nk], word w[Nb * (Nr + 1)], Nk)
begin
    i=0
    while (i < Nk)
       w[i] = word[key[4*i],key[4*i+1],key[4*i+2],key[4*i+3]]
       i = i + 1
    end while

    i = Nk
    while (i < Nb * (Nr + 1))
       word temp = w[i - 1]
       if (i mod Nk = 0)
          temp = SubWord(RotWord(temp)) xor Rcon[i / Nk]
       else if (Nk = 8 and i mod Nk = 4)
          temp = SubWord(temp)
       end if
    w[i] = w[i - Nk] xor temp
    i = i + 1
    end while
end
```

Note that ***Nk*=4, 6,** and **8** do not all have to be implemented; they are all included in the conditional statement above for conciseness. Specific implementation requirements for the Cipher Key are presented in Sec. 6.1.

Figure 12. Pseudo Code for Key Expansion.[2]

From Fig. 12, it can be seen that the first *Nk* words of the expanded key are filled with the Cipher Key. Every following word, `w[i]`, is equal to the XOR of the previous word, `w[i-1]`, and the word *Nk* positions earlier, `w[i-`*Nk*`]`. For words in positions that are a multiple of *Nk*, a transformation is applied to `w[i-1]` prior to the XOR, followed by an XOR with a round constant, `Rcon[i]`. This transformation consists of a cyclic shift of the bytes in a word (`RotWord()`), followed by the application of a table lookup to all four bytes of the word (`SubWord()`).

It is important to note that the Key Expansion routine for 256-bit Cipher Keys (*Nk* = 8) is slightly different than for 128- and 192-bit Cipher Keys. If *Nk* = 8 and `i-4` is a multiple of *Nk*, then `SubWord()` is applied to `w[i-1]` prior to the XOR.

Appendix B presents an example of the Key Expansion.

6.3 Inverse Cipher

The Cipher transformations in Sec. 5.1 can be inverted and then implemented in reverse order to produce a straightforward Inverse Cipher for the AES algorithm. The individual transformations used in the Inverse Cipher - `InvShiftRows()`, `InvSubBytes()`,`InvMixColumns()`, and `AddRoundKey()` – process the State and are described in the following subsections.

The Inverse Cipher is described in the pseudo code in Fig. 13. In Fig. 13, the array `w[]` contains the key schedule, which was described previously in Sec. 6.2.

```
InvCipher(byte in[4 * Nb], byte out[4 * Nb], word w[Nb * (Nr + 1)])
begin
   byte  state[4,Nb]

   state = in

   AddRoundKey(state, w + Nr * Nb)                  // See Sec. 6.1.4

   for round = Nr - 1 step -1 to 1
      InvShiftRows(state)                           // See Sec. 6.3.1
      InvSubBytes(state)                            // See Sec. 6.3.2
      AddRoundKey(state, w + round * Nb)
      InvMixColumns(state)                          // See Sec. 6.3.3
   end for

   InvShiftRows(state)
   InvSubBytes(state)
   AddRoundKey(state, w)

   out = state
end
```

Figure 13. Pseudo Code for the Inverse Cipher.[3]

6.3.1 `InvShiftRows()` Transformation

`InvShiftRows()` is the inverse of the `ShiftRows()` transformation. The bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row, Row 0, is not shifted. The bottom three rows are cyclically shifted by $\boldsymbol{Nb} - shift(r, Nb)$ bytes, where the shift value *shift(r,Nb)* depends on the row number, and is given in equation (5.4) (see Sec. 6.1.2).

Specifically, the `InvShiftRows()` transformation proceeds as follows:

$$s'_{r,(c+shift(r,Nb)) \bmod Nb} = s_{r,c} \quad \text{for } 0 < r < 4 \quad \text{and} \quad 0 \le c < \boldsymbol{Nb} \tag{5.8}$$

Figure 14 illustrates the `InvShiftRows()` transformation.

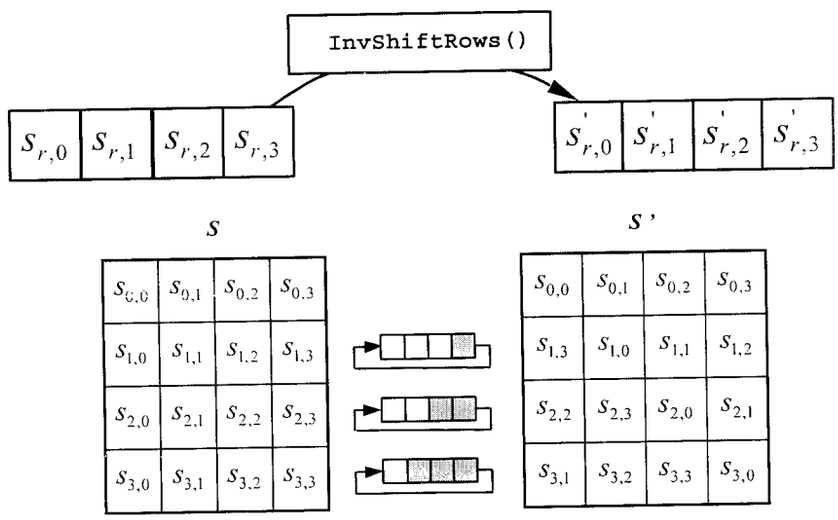


Figure 14. `InvShiftRows()` cyclically shifts the last three rows in the State.

6.3.2 `InvSubBytes()` Transformation

`InvSubBytes()` is the inverse of the byte substitution transformation, in which the inverse S-box is applied to each byte of the State. This is obtained by applying the inverse of the affine transformation (5.1) followed by taking the multiplicative inverse in $GF(2^8)$.

The inverse S-box used in the `InvSubBytes()` transformation is presented in Fig. 15:

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Figure 15. AES algorithm Inverse S-box, showing substitution values for the byte xy (in hexadecimal format).

6.3.3 `InvMixColumns()` Transformation

`InvMixColumns()` is the inverse of the `MixColumns()` transformation. `InvMixColumns()` operates on the State column-by-column, treating each column as a four-term polynomial as described in Sec. 5.2. The columns are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial $a^{-1}(x)$, given by

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}. \tag{5.9}$$

As described in Sec. 5.2, this can be written as a matrix multiplication. Let

$s'(x) = a^{-1}(x) \otimes s(x)$:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < \boldsymbol{Nb}. \tag{5.10}$$

As a result of this multiplication, the four bytes in a column are replaced by the following:

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$

$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$

$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

6.3.4 Inverse of the `AddRoundKey()` Transformation

`AddRoundKey()`, which was described in Sec. 6.1.4, is its own inverse, since it only involves an application of the XOR operation.

6.3.5 Equivalent Inverse Cipher

In the straightforward Inverse Cipher presented in Sec. 6.3 and Fig. 13, the sequence of the transformations differs from that of the Cipher, while the form of the key schedules for encryption and decryption remains the same. However, several properties of the AES algorithm allow for an Equivalent Inverse Cipher that has the same sequence of transformations as the Cipher (with the transformations replaced by their inverses). This is accomplished with a change in the key schedule.

The two properties that allow for this Equivalent Inverse Cipher are as follows:

1. The order of the **`SubBytes()`** and **`ShiftRows()`** transformations does not matter. The same is true for their inverses, **`InvSubBytes()`** and **`InvShiftRows()`**.This is true because **`SubBytes()`** and **`InvSubBytes()`** operate on individual byte values, while **`ShiftRows()`** and **`InvShiftRows()`** move bytes without changing their values.
2. The column mixing operations - **`MixColumns()`** and **`InvMixColumns()`** - are linear with respect to the column input, which means

```
InvMixColumns(state XOR Round Key) =
                    InvMixColumns(state) XOR InvMixColumns(Round Key)
```

These properties allow for the order of **`InvSubBytes()`** and **`InvShiftRows()`** to be reversed. **`AddRoundKey()`** and **`InvMixColumns()`** can also be reversed, provided that the columns (words) of the decryption key schedule are transformed using **`InvMixColumns()`**. This latter operation shall *not* be performed on the first or the last ***Nb*** words in the key schedule, since those do not operate with **`InvMixColumns()`**.

Given these changes, the resulting Equivalent Inverse Cipher offers a more efficient structure than the Inverse Cipher described in Sec. 6.3 and Fig. 13. Pseudo code for the Equivalent Inverse Cipher appears in Fig. 16. (The word array **`dw[]`** contains the modified decryption key schedule. The modification to the Key Expansion routine is also provided in Fig. 16.)

```
EqInvCipher(byte in[4 * Nb], byte out[4 * Nb], word dw[Nb * (Nr + 1)])
begin
   byte  state[4,Nb]

   state = in

   AddRoundKey(state, dw + Nr * Nb)

   for round = Nr - 1 step -1 to 1
      InvSubBytes(state)
      InvShiftRows(state)
      InvMixColumns(state)
      AddRoundKey(state, dw + round * Nb)
   end for

   InvSubBytes(state)
   InvShiftRows(state)
   AddRoundKey(state, dw)

   out = state
end
```

For the Equivalent Inverse Cipher, the following pseudo code is added at the end of the Key Expansion routine (Sec. 6.2):

```
   for i = 0 step 1 to (Nr + 1) * Nb - 1
      dw[i] = w[i]
   end for

   for rnd = 1 step 1 to Nr - 1
      InvMixColumns(dw + rnd * Nb)  // note change of type
   end for
```

Note that, since **InvMixColumns** operates on a two-dimensional array of bytes while the Round Keys are held in an array of words, the call to **InvMixColumns** in this code sequence involves a change of type (i.e. the input to **InvMixColumns()** is normally the State array, which is considered to be a two-dimensional array of bytes, whereas the input here is a Round Key computed as a one-dimensional array of words).

Figure 16. Pseudo Code for the Equivalent Inverse Cipher.

Implementation Issues

7. Implementation Issues

7.1 Key Length Requirements

An implementation of the AES algorithm shall support *at least one* of the three key lengths specified in Sec. 6: 128, 192, or 256 bits (i.e., $Nk = 4$, 6, or 8, respectively). Implementations may optionally support two or three key lengths, which may promote the interoperability of algorithm implementations.

7.2 Keying Restrictions

No weak or semi-weak keys have been identified for the AES algorithm, and there is no restriction on key selection.

7.3 Parameterization of Key Length, Block Size, and Round Number

This standard explicitly defines the allowed values for the key length (Nk), block size (Nb), and number of rounds (Nr) – see Fig. 5. However, future reaffirmations of this standard could include changes or additions to the allowed values for those parameters. Therefore, implementers may choose to design their AES implementations with future flexibility in mind.

coding

```
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.filechooser.*;
import javax.swing.filechooser.FileFilter;


class Encryption
{
	int[] keyexp(int[] tempkey)
	{
		int i,o;
		int wkey[] = new int[Rijndaelinterface.nb*(Rijndaelinterface.nr+1)];
		int temp = 0;
		int y;
		int tpkkey[] = new int[4];
		int rcon[] = new int[30];

		y = 1;

		for(i=1;i<30;i++)
		{
			y = y<<24;
			y = y>>>24;
			rcon[i] = y;
			y = Rijndaelinterface.xtime(y);
		}

		for(i=0;i<Rijndaelinterface.nk;i++)
		{
			for(o=0;o<4;o++)
			{
				tpkkey[o] = tempkey[4*i + o];
				tpkkey[o] = (tpkkey[o])<<24;
				tpkkey[o] = (tpkkey[o])>>>24;
			}

			wkey[i] = pack(tpkkey);
		}
```

```
        for(i=Rijndaelinterface.nk;i<Rijndaelinterface.nb*(Rijndaelinterface.nr+1);i++
)
                {
                        temp = wkey[i-1];

                        if(Rijndaelinterface.nk <= 6)
                        {
                                if((i % Rijndaelinterface.nk) == 0)
                                        temp = subbyte(rotr(temp)) ^
rcon[i/Rijndaelinterface.nk];
                        }
                        else if(Rijndaelinterface.nk > 6)
                        {
                                if(i%Rijndaelinterface.nk == 0)
                                        temp = subbyte(rotr(temp)) ^
rcon[i/Rijndaelinterface.nk];
                                else if(i%Rijndaelinterface.nk == 4)
                                        temp = subbyte(temp);
                        }

                        wkey[i] = wkey[i-Rijndaelinterface.nk] ^ temp;
                }

                return wkey;
        }

        int[] encrypt(int[] iblock,int[] iw)
        {
                int i,k;
                int rkey[] = new int[Rijndaelinterface.nb];

                for(i=0;i<Rijndaelinterface.nb;i++)
                        rkey[i] = iw[i];

                iblock = addrkey(iblock,rkey);

                for(i=1;i<Rijndaelinterface.nr;i++)
                {
                        for(k=0;k<Rijndaelinterface.nb;k++)
                                rkey[k] = iw[Rijndaelinterface.nb*i + k];
```

```
            iblock = round(iblock,rkey);
        }

        for(k=0;k<Rijndaelinterface.nb;k++)
            rkey[k] = iw[Rijndaelinterface.nb*Rijndaelinterface.nr + k];

        iblock = fround(iblock,rkey);

        return iblock;
}

int[] addrkey(int[] inb,int[] rk)
{
        int i,j;
        int tmp[] = new int[4];

        for(i=0;i<Rijndaelinterface.nb;i++)
        {
            tmp = unpack(rk[i]);

            for(j=0;j<4;j++)
            {
                inb[4*i + j] = (inb[4*i + j] ^ tmp[j])<<24;
                inb[4*i + j] = (inb[4*i + j])>>>24;
            }
        }

        return inb;
}

int[] round(int[] rblock,int[] rokey)
{
        int i,k;

        for(i=0;i<(Rijndaelinterface.nb*4);i++)
        {
            rblock[i] = (Rijndaelinterface.fbsub[rblock[i]])<<24;
            rblock[i] = (rblock[i])>>>24;
        }
```

```
        rblock = shiftrow(rblock);
        rblock = mixcol(rblock);
        rblock = addrkey(rblock,rokey);

        return rblock;
}

int[] fround(int[] rblock,int[] rokey)
{
        int i,k;

        for(i=0;i<(Rijndaelinterface.nb*4);i++)
        {
                rblock[i] = (Rijndaelinterface.fbsub[rblock[i]])<<24;
                rblock[i] = (rblock[i])>>>24;
        }

        rblock = shiftrow(rblock);
        rblock = addrkey(rblock,rokey);

        return rblock;
}

int[] shiftrow(int[] arr6)
{
        int p,r,s;
        int arr7[] = new int[Rijndaelinterface.nb];

        for(p=1;p<=3;p++)
        {
                for(r=0;r<Rijndaelinterface.nb;r++)
                        arr7[r] = arr6[4*r + p];

                switch(p)
                {
                        case 1: for(s=0;s<Rijndaelinterface.c1;s++)
                                        arr7 = rotrow(arr7);

                                break;

                        case 2: for(s=0;s<Rijndaelinterface.c2;s++)
```

```
                                arr7 = rotrow(arr7);

                            break;

                        case 3: for(s=0;s<Rijndaelinterface.c3;s++)
                                arr7 = rotrow(arr7);

                            break;
                }

                for(r=0;r<Rijndaelinterface.nb;r++)
                    arr6[4*r + p] = arr7[r];
        }

        return arr6;
}

int[] rotrow(int[] arr8)
{
        int intemp;
        int t;

        intemp = arr8[0];

        for(t=0;t<Rijndaelinterface.nb;t++)
        {
                if(t == (Rijndaelinterface.nb) - 1)
                        arr8[t] = intemp;
                else
                        arr8[t] = arr8[t+1];
         }

        return arr8;
 }

int[] mixcol(int[] arr9)
{
        int u,v,w;
        int arr10[] = new int[Rijndaelinterface.nb*4];

        int conval1[][] = {
```

```
                                {2,3,1,1},
                                {1,2,3,1},
                                {1,1,2,3},
                                {3,1,1,2}
                        };

                for(u=0;u<(Rijndaelinterface.nb*4);u++)
                        arr10[u] = 0;

                for(u=0;u<(4*Rijndaelinterface.nb);u=u+4)
                {
                        for(v=0;v<4;v++)
                        {
                                for(w=0;w<4;w++)
                                {
                                        arr10[u+v] = (arr10[u+v] ^
prod(conval1[v][w],arr9[u+w]))<<24;
                                        arr10[u+v] = (arr10[u+v])>>>24;
                                }
                        }
                }

                return arr10;
        }

        int prod(int v1,int v2)
        {
                if((v1 != 0) && (v2 != 0))
                        return Rijndaelinterface.ptab[(Rijndaelinterface.ltab[v1] +
Rijndaelinterface.ltab[v2]) % 255];
                else
                        return 0;
        }

        int rotr(int w)
        {
                int nw;

                nw = ((w >>> 8) | (w << 24));
                return nw;
        }
```

```
int pack(int b[])
{
        int x,y;

        x = b[3];
        y = x << 24;
        x = b[2];
        y = y | (x << 16);
        x = b[1];
        y = y | (x << 8);
        x = b[0];
        y = y | x;

        return y;
 }

int subbyte(int w)
{
        int b[] = new int[4];
        int x,y;

        b = unpack(w);

        b[0] = (Rijndaelinterface.fbsub[b[0]])<<24;
        b[0] = (b[0])>>>24;
        b[1] = (Rijndaelinterface.fbsub[b[1]])<<24;
        b[1] = (b[1])>>>24;
        b[2] = (Rijndaelinterface.fbsub[b[2]])<<24;
        b[2] = (b[2])>>>24;
        b[3] = (Rijndaelinterface.fbsub[b[3]])<<24;
        b[3] = (b[3])>>>24;

        y = pack(b);

        return y;
 }

int[] unpack(int wd)
{
        int a[] = new int[4];
```

```
            a[0] = wd<<24;
            a[0] = a[0]>>>24;
            a[1] = (wd >> 8)<<24;
            a[1] = a[1]>>>24;
            a[2] = (wd >> 16)<<24;
            a[2] = a[2]>>>24;
            a[3] = (wd >> 24)<<24;
            a[3] = a[3]>>>24;

            return a;
        }
}

class Decryption
{
        int[] keyexp(int[] tempkey)
        {
                int i,o;
                int wkey[] = new int[Rijndaelinterface.nb*(Rijndaelinterface.nr+1)];
                int temp = 0;
                int y;
                int tpkkey[] = new int[4];
                int rcon[] = new int[30];

                y = 1;

                for(i=1;i<30;i++)
                {
                        y = y<<24;
                        y = y>>>24;
                        rcon[i] = y;
                        y = Rijndaelinterface.xtime(y);
                 }

                for(i=0;i<Rijndaelinterface.nk;i++)
                {
                        for(o=0;o<4;o++)
                        {
                                tpkkey[o] = tempkey[4*i + o];
                                tpkkey[o] = (tpkkey[o])<<24;
```

```
                    tpkkey[o] = (tpkkey[o])>>>24;
                }

                wkey[i] = pack(tpkkey);
            }


        for(i=Rijndaelinterface.nk;i<Rijndaelinterface.nb*(Rijndaelinterface.nr+1);i++
)
            {
                temp = wkey[i-1];

                if(Rijndaelinterface.nk <= 6)
                {
                    if((i % Rijndaelinterface.nk) == 0)
                        temp = subbyte(rotr(temp)) ^
rcon[i/Rijndaelinterface.nk];
                }
                else if(Rijndaelinterface.nk > 6)
                {
                    if(i%Rijndaelinterface.nk == 0)
                        temp = subbyte(rotr(temp)) ^
rcon[i/Rijndaelinterface.nk];
                    else if(i%Rijndaelinterface.nk == 4)
                        temp = subbyte(temp);
                }

                wkey[i] = wkey[i-Rijndaelinterface.nk] ^ temp;
            }

            return wkey;
        }

        int[] decrypt(int[] iblock,int[] iw)
        {
            int i,k;
            int rkey[] = new int[Rijndaelinterface.nb];

            for(k=0;k<Rijndaelinterface.nb;k++)
                rkey[k] = iw[Rijndaelinterface.nb*Rijndaelinterface.nr + k];
```

```
        iblock = fround(iblock,rkey);

        for(i=(Rijndaelinterface.nr)-1;i>0;i--)
        {
                for(k=0;k<Rijndaelinterface.nb;k++)
                        rkey[k] = iw[Rijndaelinterface.nb*i + k];

                iblock = round(iblock,rkey);
        }

        for(i=0;i<Rijndaelinterface.nb;i++)
                rkey[i] = iw[i];

        iblock = addrkey(iblock,rkey);

        return iblock;
}

int[] round(int[] rblock,int[] rokey)
{
        int i,k;

        rblock = addrkey(rblock,rokey);
        rblock = mixcol(rblock);
        rblock = shiftrow(rblock);

        for(i=0;i<(Rijndaelinterface.nb*4);i++)
        {
                rblock[i] = (Rijndaelinterface.rbsub[rblock[i]])<<24;
                rblock[i] = (rblock[i])>>>24;
        }

        return rblock;
}

int[] fround(int[] rblock,int[] rokey)
{
        int i,k;

        rblock = addrkey(rblock,rokey);
        rblock = shiftrow(rblock);
```

```
        for(i=0;i<(Rijndaelinterface.nb*4);i++)
        {
                rblock[i] = (Rijndaelinterface.rbsub[rblock[i]])<<24;
                rblock[i] = (rblock[i])>>>24;
         }

        return rblock;
 }

int[] addrkey(int[] inb,int[] rk)
{
        int i,j;
        int tmp[] = new int[4];

        for(i=0;i<Rijndaelinterface.nb;i++)
        {
                tmp = unpack(rk[i]);

                for(j=0;j<4;j++)
                {
                        inb[4*i + j] = (inb[4*i + j] ^ tmp[j])<<24;
                        inb[4*i + j] = (inb[4*i + j])>>>24;
                 }
         }

        return inb;
 }

int[] shiftrow(int[] arr6)
{
        int p,r,s;
        int arr7[] = new int[Rijndaelinterface.nb];

        for(p=1;p<=3;p++)
        {
                for(r=0;r<Rijndaelinterface.nb;r++)
                        arr7[r] = arr6[4*r + p];

                switch(p)
                {
```

```
                        case 1: for(s=0;s<(Rijndaelinterface.nb -
Rijndaelinterface.c1);s++)
                                arr7 = rotrow(arr7);

                        break;

                        case 2: for(s=0;s<(Rijndaelinterface.nb -
Rijndaelinterface.c2);s++)
                                arr7 = rotrow(arr7);

                        break;

                        case 3: for(s=0;s<(Rijndaelinterface.nb -
Rijndaelinterface.c3);s++)
                                arr7 = rotrow(arr7);

                        break;
                }

                for(r=0;r<Rijndaelinterface.nb;r++)
                        arr6[4*r + p] = arr7[r];
        }

        return arr6;
    }

    int[] rotrow(int[] arr8)
    {
        int intemp;
        int t;

        intemp = arr8[0];

        for(t=0;t<Rijndaelinterface.nb;t++)
        {
            if(t == (Rijndaelinterface.nb) - 1)
                arr8[t] = intemp;
            else
                arr8[t] = arr8[t+1];
        }
```

```
        return arr8;
    }

    int[] mixcol(int[] arr9)
    {
        int u,v,w;
        int arr10[] = new int[Rijndaelinterface.nb*4];

        int conval1[][] = {
                            {0x0E,0x0B,0x0D,0x09},
                            {0x09,0x0E,0x0B,0x0D},
                            {0x0D,0x09,0x0E,0x0B},
                            {0x0B,0x0D,0x09,0x0E}
                          };

        for(u=0;u<(Rijndaelinterface.nb*4);u++)
            arr10[u] = 0;

        for(u=0;u<(4*Rijndaelinterface.nb);u=u+4)
        {
            for(v=0;v<4;v++)
            {
                for(w=0;w<4;w++)
                {
                    arr10[u+v] = (arr10[u+v] ^
prod(conval1[v][w],arr9[u+w]))<<24;
                    arr10[u+v] = (arr10[u+v])>>>24;
                }
            }
        }

        return arr10;
    }

    int prod(int v1,int v2)
    {
        if((v1 != 0) && (v2 != 0))
            return Rijndaelinterface.ptab[(Rijndaelinterface.ltab[v1] +
Rijndaelinterface.ltab[v2]) % 255];
        else
            return 0;
```

```
}

int rotr(int w)
{
        int nw;

        nw = ((w >>> 8) | (w << 24));
        return nw;
}

int pack(int b[])
{
        int x,y;

        x = b[3];
        y = x << 24;
        x = b[2];
        y = y | (x << 16);
        x = b[1];
        y = y | (x << 8);
        x = b[0];
        y = y | x;

        return y;
}

int subbyte(int w)
{
        int b[] = new int[4];
        int x,y;

        b = unpack(w);

        b[0] = (Rijndaelinterface.fbsub[b[0]])<<24;
        b[0] = (b[0])>>>24;
        b[1] = (Rijndaelinterface.fbsub[b[1]])<<24;
        b[1] = (b[1])>>>24;
        b[2] = (Rijndaelinterface.fbsub[b[2]])<<24;
        b[2] = (b[2])>>>24;
        b[3] = (Rijndaelinterface.fbsub[b[3]])<<24;
        b[3] = (b[3])>>>24;
```

```
            y = pack(b);

            return y;
        }

        int[] unpack(int wd)
        {
            int a[] = new int[4];

            a[0] = wd<<24;
            a[0] = a[0]>>>24;
            a[1] = (wd >> 8)<<24;
            a[1] = a[1]>>>24;
            a[2] = (wd >> 16)<<24;
            a[2] = a[2]>>>24;
            a[3] = (wd >> 24)<<24;
            a[3] = a[3]>>>24;

            return a;
        }
}
class Rijndaelinterface
{
        static int fbsub[] = new int[256];
        static int rbsub[] = new int[256];
        static int ptab[] = new int[256];
        static int ltab[] = new int[256];
        static int c1,c2,c3,nb,nk,nr;
        static int i;
        static int j;

        static
        {
            ltab[0] = 0;
            ptab[0] = 1;
            ltab[1] = 0;
            ptab[1] = 3;
            ltab[3] = 1;

            try
```

```
        {
                for(i=2;i<=255;i++)
                {
                        j = (ptab[i-1])<<24;
                        j = j>>>24;
                        ptab[i-1] = j;
                        j = ptab[i-1] ^ xtime(ptab[i-1]);
                        ptab[i] = j<<24;
                        ptab[i] = (ptab[i])>>>24;
                        ltab[ptab[i]] = i;
                }
        }
        catch(ArrayIndexOutOfBoundsException e)
        {
                System.out.println("I've found the error : " +e);
        }

        fbsub[0] = (int)0x63;
        rbsub[0x63] = 0;

        for(i=1;i<256;i++)
        {
                j = bytesub(i);

                fbsub[i] = j;
                rbsub[j] = i;
        }
}

static int xtime(int a)
{
        int b;

        if((a & (int)0x80) != 0)
                b = (int)0x1b;
        else
                b = 0;

        a <<= 1;
        a ^= b;
```

```
            return a;
        }

        static int bytesub(int c)
        {
            int y;

            y = ptab[255 - ltab[c]];

            c = y;
            c = ((c >> 7) | (c << 1));
            y ^= c;
            c = ((c >> 7) | (c << 1));
            y ^= c;
            c = ((c >> 7) | (c << 1));
            y ^= c;
            c = ((c >> 7) | (c << 1));
            y ^= c;
            y ^= (int)0x63;

            y = y<<24;
            y = y>>>24;
            return y;
        }
public static class ImageFilter extends FileFilter
{
  public boolean accept(File f)
  {
    if (f.isDirectory())
      {
       return true;
    }
    String extension =  getExtension(f);
     if (extension != null)
     {
     if (extension.equals("rock"))
     return true;
    }
     else
     {
     return false;
```

```
    }
    return false;
  }
  public String getDescription()
  {
    return "Just Rock";
  }
  public static String getExtension(File f)
  {
    String ext = null;
    String s = f.getName();
    int i = s.lastIndexOf('.');
  if (i > 0 &&  i < s.length() - 1) {
      ext = s.substring(i+1).toLowerCase();
    }
    return ext;
  }
}
public  static class RijndaelinterfaceDemo extends JFrame {

    private final String newline = "\n";
    String infile;
    String outfile;
    String encdecfilename=null;
    String decencfilename=null;
    boolean check=true;
    FileInputStream fin;
    FileOutputStream fout;
    int enc1=0;
    int dec1=0;
    int rembyte=0;
    JTextArea log;
    int gogo=0;
    int donego=0;
    int gogo1=0;
    JPasswordField p= new JPasswordField(18);
    final JProgressBar pb=new JProgressBar();
    int[] kk=new int[32];
    static int aa=0,c=1,ll=0,dd=0;
    static String directoryname=null;
    static int direcrypt=0;
```

```
static char pass[];
public RijndaelinterfaceDemo()
{
      this.setTitle("CRYPT 'O' CRYPT");
this.setSize(new Dimension(500, 500));
      this.setResizable(false);
      ImageIcon image = new ImageIcon("c:\\crypt\\brick.gif");
      this.setIconImage(image.getImage());
        try
        {
        System.out.println("this is in the rdemo");
              final File f1=new File("C:\\");
              final JButton done = new JButton("Ok");
            ImageIcon openIcon = new ImageIcon("c:\\crypt\\encrypt.gif");
              JButton encButton = new JButton("Encrypt",openIcon);
              ImageIcon openIcon1 = new ImageIcon("c:\\crypt\\decrypt.gif");
            JButton decButton = new JButton("Decrypt",openIcon1);
              encButton.setToolTipText("Select a File to Encrypt ");
              decButton.setToolTipText("Select a File to Decrypt ");
              final JButton cancel = new JButton("Cancel");
              final JPanel buttonPanel = new JPanel();
            buttonPanel.add(encButton);
            buttonPanel.add(decButton);
              Icon ic=new ImageIcon("c:\\crypt\\logo.gif");
              Icon ic1=new ImageIcon("c:\\crypt\\file.gif");
              Icon ic2=new ImageIcon("c:\\crypt\\dir.gif");
              final JButton img= new JButton(ic);
              JLabel l= new JLabel();
              Icon lic=new ImageIcon("c:\\crypt\\pwd1.gif");
              l.setIcon(lic);
              final JPanel passwd = new JPanel();
              passwd.add(l);
              passwd.add(p);
              img.setToolTipText("FINAL IT");
              JLabel jl1=new JLabel("0%");
              JLabel jl2=new JLabel("100%");
              final JPanel jp=new JPanel();
              jp.add(jl1);
              pb.setAlignmentX(Component.CENTER_ALIGNMENT);
              pb.setMaximumSize(new Dimension(200,80));
              jp.add(pb);
```

```
                jp.add(jl2);
                jp.add(cancel);
                final JButton filecr = new JButton("FILE CRYPT",ic1 );
                final JButton direcr = new JButton("DIRE CRYPT",ic2 );
                final JPanel pp = new JPanel();
                final JPanel cry = new JPanel();
                pp.setLayout(new BoxLayout(pp,BoxLayout.Y_AXIS));
                img.setAlignmentX(Component.CENTER_ALIGNMENT);
                pp.add(passwd);
                pp.add(jp);
                pp.add(img);
                cry.add(filecr);
                cry.add(direcr);
encButton.addActionListener(new ActionListener()
  {
   public void actionPerformed(ActionEvent e)
     {
        final JFileChooser fc = new JFileChooser( );
        pb.setValue(0);
        c=1;
        fc.setCurrentDirectory(f1);
        if(direcrypt==1)
        fc.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
        int returnVal = fc.showDialog(RijndaelinterfaceDemo.this,"Encrypt");
       if (returnVal == JFileChooser.APPROVE_OPTION)
        {
                File file = fc.getSelectedFile();
                infile= new String(file.getPath());
                System.out.println(infile);
                encdecfilename= new String(file.getName());
                try
                {
                gogo=1;
                dec1=0;
                enc1=1;
                passwd.setVisible(true);
                buttonPanel.setVisible(false);
                }catch(Exception ee)
                {System.out.println("ERROR"+ee);
                }
        }
```

```
            else
            {
                  p.setVisible(false);
                  done.setVisible(false);
            }
      }
});
decButton.addActionListener(new ActionListener()
  {
      public void actionPerformed(ActionEvent e)
      {
          final JFileChooser fc = new JFileChooser( );
          fc.addChoosableFileFilter(new ImageFilter());
           pb.setValue(0);
          c=1;
//        FileFilter filter;
//        filter.addExtension("jpg");
          fc.setCurrentDirectory(f1);
          if(direcrypt==1)
          fc.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
        int returnVal = fc.showDialog(RijndaelinterfaceDemo.this,"Decrypt");
        if (returnVal == JFileChooser.APPROVE_OPTION)
          {
            File file = fc.getSelectedFile();
            infile= new String(file.getPath());
                System.out.println(infile);
                decencfilename= new String(file.getName());
                try
                {
                gogo=1;
                dec1=1;
                enc1=0;
                passwd.setVisible(true);
                buttonPanel.setVisible(false);
                }catch(Exception eee)
                {System.out.println("ERROR"+eee);
                }
          }
           else
          {
                  passwd.setVisible(false);
```

```
        }

        }
    });
    p.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            pass=p.getPassword();
            System.out.print ("The Password is ");
            for(int i=0;i<pass.length;i++)
            {
                System.out.print(pass[i]);
                kk[i]=pass[i];
                pass[i]=0;
            }
            System.out.println("  ");
            gogo1=1;
            getdir(infile);
            p.setText("");
            passwd.setVisible(false);
            buttonPanel.setVisible(true);

        }

    });
    img.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
                String url = "http://www.ealcatraz.com/";
            String browserName;
                browserName = "C:\\Progra~1\\Intern~1\\Iexplore.exe"; //
file.getPath();
                try
                {
                Runtime.getRuntime().exec(new String[] {browserName.
url});
                }
                catch (IOException exc)
                {
```

```
                        exc.printStackTrace();
                    }
        }
});

filecr.addActionListener(new ActionListener()
{
        public void actionPerformed(ActionEvent e)
        {
                    RijndaelinterfaceDemo.this.setTitle("FILE CRYPT"),
                    direcrypt=0;
                    cry.setVisible(false);
                    buttonPanel.setVisible(true);
                    passwd.setVisible(false);
                    pp.setVisible(true);
                    cancel.setVisible(true);
        }
});
direcr.addActionListener(new ActionListener()
{
        public void actionPerformed(ActionEvent e)
        {
                    RijndaelinterfaceDemo.this.setTitle("DIRE CRYPT");
                    direcrypt=1;
                    cry.setVisible(false);
                    buttonPanel.setVisible(true);
                    passwd.setVisible(false);
                    pp.setVisible(true);
                    cancel.setVisible(true);
        }
});

cancel.addActionListener(new ActionListener()
{
        public void actionPerformed(ActionEvent e)
        {
                    infile=null;
                    pb.setValue(0);
                    p.setText("");
                    cry.setVisible(true);
                    buttonPanel.setVisible(false);
```

```
                        passwd.setVisible(false);
                        pp.setVisible(true);
                        cancel.setVisible(false);
                        RijndaelinterfaceDemo.this.setTitle("CRYPT 'O'
CRYPT");
                }
        });
      encButton.setNextFocusableComponent(decButton);
      decButton.setNextFocusableComponent(encButton);
      Container contentPane = getContentPane();
      contentPane.add(buttonPanel, BorderLayout.NORTH);
        contentPane.add(passwd,BorderLayout.CENTER);
        contentPane.add(pp, BorderLayout.SOUTH);
        cancel.setVisible(false);
        pack();
        contentPane.add(cry, BorderLayout.NORTH);
        buttonPanel.setVisible(false);
        passwd.setVisible(false);
        setVisible(true);
        }catch(Exception e)
        {
                System.out.println("Error "+e);
        }


                public void getdir(String sse)
                {
                        try
                        {
                        String tempname;
                        File f=new File(sse);
                        if(f.isDirectory())
                        {
                        String s[];
                        s=f.list();
                        for(int i=0;i<s.length;i++)
                        {
                                File nf=new File(sse + "\\" +s[i]);
                                if(nf.isDirectory())
                                {
                                        String spsp;
```

```
                    spsp=nf.getAbsolutePath();

                    c++;
                    getdir(spsp);
                }
                else
                {
                    tempname=nf.getAbsolutePath();
                    if(enc1==1)
                    {
                    outfile=tempname+".rock";
                    System.out.println("Encryption file name
is"+tempname);
                    System.out.println("Encrypted file name is"+outfile);
                    dowork(tempname);
                    }
                    if(dec1==1)
                    {
                    int c=tempname.length();
                    int q=c-5;
                    char[] cc=new char[q];
                    char[] cc1=new char[q];
                    tempname.getChars(0,q,cc,0);
                    tempname.getChars(q,c,cc1,0);
                    int opop=0;
                    outfile= new String(cc);
                    System.out.println("Decryption file name
is"+tempname);
                    System.out.println("Decrypted file name is
"+outfile);
                    dowork(tempname);
                    }

                }
            System.out.println("getdir");
            }
            }
            else
            {
                    tempname=f.getAbsolutePath();
                    if(enc1==1)
```

```
                                {
                                outfile=tempname+".rock";
                                System.out.println("Encryption file name
is"+tempname);
                                System.out.println("Encrypted file name is"+outfile);
                                dowork(tempname);
                                }
                                if(dec1==1)
                                {
                                int c=tempname.length();
                                int q=c-5;
                                char[] cc=new char[q];
                                char[] cc1=new char[q];
                                tempname.getChars(0,q,cc,0);
                                tempname.getChars(q,c,cc1,0);
                                int opop=0;
                                outfile= new String(cc);
                                System.out.println("Decryption file name
is"+tempname);
                                System.out.println("Decrypted file name is
"+outfile);
                                dowork(tempname);
                                }

                        }

                }catch(Exception qe)
                {
                        System.out.println(" Exception "+qe);
                }

        }
                public   void dowork(String sinp)
                {

                        try
                                        {
                                                Encryption enc = new Encryption();
                                               Decryption dec = new Decryption();
                                               int keyst=0,blkst=0;
                                               int sl=0;
```

```
int op=0;
int val1=0,val2=0;
int inval;

keyst = 256;
blkst =256;

nb = blkst/32;
nk = keyst/32;

if(nb >= nk)
        nr = nb + 6;
else
        nr = nk + 6;

c1 = 1;

if(nb < 8)
{
        c2 = 2;
        c3 = 3;
 }
else
{
        c2 = 3;
        c3 = 4;
 }

int block[] = new int[nb*4];
int key[] = new int[nk*4];
int w[] = new int[nb*(nr+1)];
```

```
FileInputStream fin = new FileInputStream(sinp);

FileOutputStream fout = new FileOutputStream(outfile);
```

```
int jp=(nk*4)-( kk.length);

for(i=0;i<( kk.length);i++)
{
```

```
                key[i] =  kk[i];
            }
            if(jp!=0)
            for(i=( kk.length+1);i<(nk*4);i++)
            {
                key[i] = kk[i];
            }

            int max=0;
            for(i=0;i<(nk*4);i++)
            {

                System.out.print((char)key[i]);
            }
                System.out.println("");
            if( enc1==1)
            {
                w = enc.keyexp(key);
                File filelen = new File(sinp);
                long flength = filelen.length();
                rembyte = (int)(flength%32);
                max=(int)(flength/32);
                fout.write(rembyte);
            }
            else
            if( dec1==1)
            {
                File filelen = new File(sinp);
                long flength = filelen.length();
                 max=(int)(flength/32);
                w = dec.keyexp(key);
                rembyte = fin.read();
            }
        int min = 0;

pb.setValue(min);
        pb.setMinimum(min);
        pb.setMaximum(max+1);

            int enc12=0;
            int dec12=0;
```

```
                                inval = 0;
inval = fin.read();
do
{
                min++;
                 pb.setValue(min);
                min++;

        sl = 0;

        for(i=0;i<(nb*4);i++)
        {
                if(inval == (-1))
                {
                        block[i] = 0<<32;
                        block[i] = block[i]>>>32;
                 }
                else
                {
                        block[i] = inval;
                        block[i] = (block[i])<<24;
                        block[i] = (block[i])>>>24;
                        sl = sl + 1;
                 }

                inval = fin.read();
         }

        if(enc1==1)
        block = enc.encrypt(block,w);
        else
        if(dec1==1)
        block = dec.decrypt(block,w);
        if(inval == (-1))
        {
                if(enc1 == 1)
                {
                        for(i=0;i<nb*4;i++)
                        {
                                block[i] = (block[i])<<24;
```

```
                        block[i] = (block[i])>>>24;
                        fout.write((char)block[i]);
                    }
                }
                else if(dec1 == 1)
                {
                    for(i=0;i<rembyte;i++)
                    {
                        block[i] = (block[i])<<24;
                        block[i] = (block[i])>>>24;
                        fout.write((char)block[i]);
                    }
                }
            }
            else
            {
                for(i=0;i<nb*4;i++)
                {
                    block[i] = (block[i])<<24;
                    block[i] = (block[i])>>>24;
                    fout.write((char)block[i]);
                }
            }
                pb.setValue(min);
    } while(inval != (-1));
                        System.out.println("DONE.");

                    fin.close();
                        fout.close();
     File delf= new File(sinp);
    delf.delete();
                        for(i=0;i<(nk*4);i++)
                            {

                            key[i]=0;
                             }
                            }catch(Exception eee)
                        {
                            System.out.println("Error"+eee);
                        }
```

```
        }

}

    public static void main(String args[]) throws IOException
    {
                JFrame  frame = new RijndaelinterfaceDemo();

                 frame.addWindowListener(new WindowAdapter()
          {
          public void windowClosing(WindowEvent e)
            {
             System.exit(0);
          }
        });
}
}
```

Conclusion

8.Conclusion

The Rijndael Algorithm has been successfully implemented and tested. The software works effectively and efficient to its requirement. The sender and the receiver alone know what the password is and have sole rights over the system. . Therefore, this is a very advanced encryption standard, which is very effective in this world where the threat to the data is very high. The project has helped us to gain a wide knowledge in Java. A little has been done, a lot more to go as the project has good scope for further development and for maintaining security.

Bibliography

9. Bibliography

Text References

1. Java Complete Reference - Patrick Naughton & Herbert Schildt- Tata McGraw Hill

2. Java 1.1 Unleashed By Michael Morrison

3. Cryptography and Secure Communications by Man Young Rhee

Web References

http://www.fp.gladman.plus.com/cryptography_technology/

http://www.nist.gov/aes/

Appendix

10.Appendix A

The object identifiers (OIDs) listed below have been registered for the AES and are for use with the four basic modes of operation for block ciphers: Electronic Codebook (ECB), Cipher Block Chaining (CBC), Output Feedback (OFB), and Cipher Feedback (CFB). These OIDs may be used to identify the modes and pass parameters (where appropriate) in protocols that use the ASN.1 structured type Algorithm.

The complete and current ASN.1 for these objects and any associated parameters is available at the Computer Security Objects Register (CSOR), located at http://csrc.nist.gov/csor/ [2].

aes OBJECT IDENTIFIER ::= { nistAlgorithms 1 }

-- 128 bit AES information object identifiers --

id-aes128-ECB OBJECT IDENTIFIER ::= { aes 1 }

id-aes128-CBC OBJECT IDENTIFIER ::= { aes 2 }

id-aes128-OFB OBJECT IDENTIFIER ::= { aes 3 }

id-aes128-CFB OBJECT IDENTIFIER ::= { aes 4 }

-- 192 bit AES information object identifiers --

id-aes192-ECB OBJECT IDENTIFIER ::= { aes 21 }

id-aes192-CBC OBJECT IDENTIFIER ::= { aes 22 }

id-aes192-OFB OBJECT IDENTIFIER ::= { aes 23 }

id-aes192-CFB OBJECT IDENTIFIER ::= { aes 24 }

-- 256 bit AES information object identifiers --

id-aes256-ECB OBJECT IDENTIFIER ::= { aes 41 }

id-aes256-CBC OBJECT IDENTIFIER ::= { aes 42 }

id-aes256-OFB OBJECT IDENTIFIER ::= { aes 43 }

id-aes256-CFB OBJECT IDENTIFIER ::= { aes 44 }

Appendix B

This appendix shows the development of the key schedule using

```
Cipher Key = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c
```

for $Nk = 4$, which results in

w_0 = 2b7e1516 w_1 = 28aed2a6 w_2 = abf71588 w_3 = 09cf4f3c

Note that multi-byte values are presented using the notation described in Sec. 4. The intermediate values produced during the development of the key schedule (see Sec. 6.2) are given in the following table (all values are in hexadecimal format, with the exception of the index column (i)).

i (dec)	temp	After RotWord()	After SubByte()	Rcon[i/Nk]	After XOR with Rcon	w[i-Nk]	w[i] = temp XOR w[i-Nk]
4	09cf4f3c	cf4f3c09	8a84eb01	01000000	8b84eb01	2b7e1516	a0fafe17
5	a0fafe17					28aed2a6	88542cb1
6	88542cb1					abf71588	23a33939
7	23a33939					09cf4f3c	2a6c7605
8	2a6c7605	6c76052a	50386be5	02000000	52386be5	a0fafe17	f2c295f2
9	f2c295f2					88542cb1	7a96b943
10	7a96b943					23a33939	5935807a
11	5935807a					2a6c7605	7359f67f
12	7359f67f	59f67f73	cb42d28f	04000000	cf42d28f	f2c295f2	3d80477d
13	3d80477d					7a96b943	4716fe3e
14	4716fe3e					5935807a	1e237e44
15	1e237e44					7359f67f	6d7a883b
16	6d7a883b	7a883b6d	dac4e23c	08000000	d2c4e23c	3d80477d	ef44a541
17	ef44a541					4716fe3e	a8525b7f
18	a8525b7f					1e237e44	b671253b
19	b671253b					6d7a883b	db0bad00
20	db0bad00	0bad00db	2b9563b9	10000000	3b9563b9	ef44a541	d4d1c6f8
21	d4d1c6f8					a8525b7f	7c839d87
22	7c839d87					b671253b	caf2b8bc
23	caf2b8bc					db0bad00	11f915bc
24	11f915bc	f915bc11	99596582	20000000	b9596582	d4d1c6f8	6d88a37a
25	6d88a37a					7c839d87	110b3efd
26	110b3efd					caf2b8bc	dbf98641

27	dbf98641					11f915bc	ca0093fd
28	ca0093fd	0093fdca	63dc5474	40000000	23dc5474	6d88a37a	4e54f70e
29	4e54f70e					110b3efd	5f5fc9f3
30	5f5fc9f3					dbf98641	84a64fb2
31	84a64fb2					ca0093fd	4ea6dc4f
32	4ea6dc4f	a6dc4f4e	2486842f	80000000	a486842f	4e54f70e	ead27321
33	ead27321					5f5fc9f3	b58dbad2
34	b58dbad2					84a64fb2	312bf560
35	312bf560					4ea6dc4f	7f8d292f
36	7f8d292f	8d292f7f	5da515d2	1b000000	46a515d2	ead27321	ac7766f3
37	ac7766f3					b58dbad2	19fadc21
38	19fadc21					312bf560	28d12941
39	28d12941					7f8d292f	575c006e
40	575c006e	5c006e57	4a639f5b	36000000	7c639f5b	ac7766f3	d014f9a8
41	d014f9a8					19fadc21	c9ee2589
42	c9ee2589					28d12941	e13f0cc8
43	e13f0cc8					575c006e	b6630ca6

Appendix C

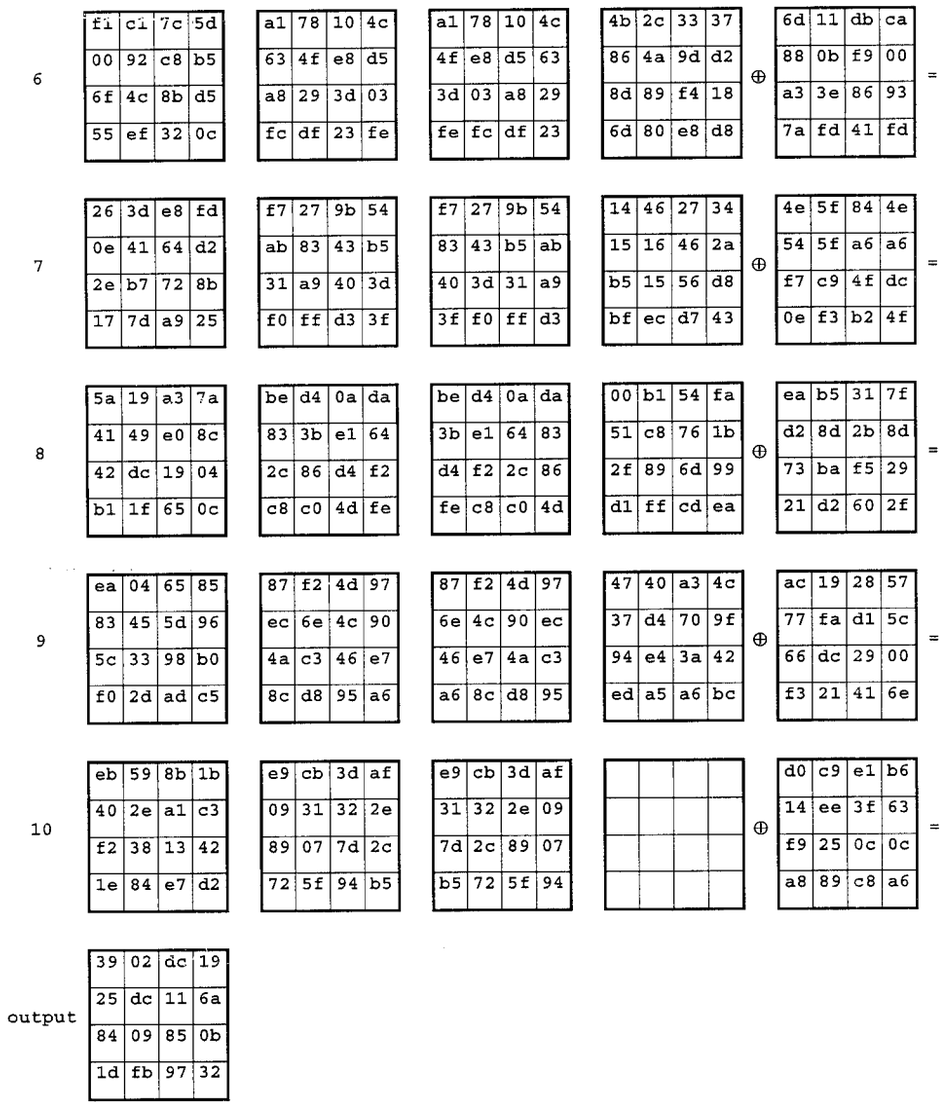
The following diagram shows the values in the State array as the Cipher progresses for a block length and a Cipher Key length of 16 bytes each.

```
Input =        32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34

Cipher Key = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c
```

The Round Key values are taken from the Key Expansion example in Appendix B.

Round Number	Start of Round	After SubBytes	After ShiftRows	After MixColumns	Round Key Value	
input	32 88 31 e0 43 5a 31 37 f6 30 98 07 a8 8d a2 34				2b 28 ab 09 7e ae f7 cf 15 d2 15 4f 16 a6 88 3c	⊕ =
1	19 a0 9a e9 3d f4 c6 f8 e3 e2 8d 48 be 2b 2a 08	d4 e0 b8 1e 27 bf b4 41 11 98 5d 52 ae f1 e5 30	d4 e0 b8 1e bf b4 41 27 5d 52 11 98 30 ae f1 e5	04 e0 48 28 66 cb f8 06 81 19 d3 26 e5 9a 7a 4c	a0 88 23 2a fa 54 a3 6c fe 2c 39 76 17 b1 39 05	⊕ =
2	a4 68 6b 02 9c 9f 5b 6a 7f 35 ea 50 f2 2b 43 49	49 45 7f 77 de db 39 02 d2 96 87 53 89 f1 1a 3b	49 45 7f 77 db 39 02 de 87 53 d2 96 3b 89 f1 1a	58 1b db 1b 4d 4b e7 6b ca 5a ca b0 f1 ac a8 e5	f2 7a 59 73 c2 96 35 59 95 b9 80 f6 f2 43 7a 7f	⊕ =
3	aa 61 82 68 8f dd d2 32 5f e3 4a 46 03 ef d2 9a	ac ef 13 45 73 c1 b5 23 cf 11 d6 5a 7b df b5 b8	ac ef 13 45 c1 b5 23 73 d6 5a cf 11 b8 7b df b5	75 20 53 bb ec 0b c0 25 09 63 cf d0 93 33 7c dc	3d 47 1e 6d 80 16 23 7a 47 fe 7e 88 7d 3e 44 3b	⊕ =
4	48 67 4d d6 6c 1d e3 5f 4e 9d b1 58 ee 0d 38 e7	52 85 e3 f6 50 a4 11 cf 2f 5e c8 6a 28 d7 07 94	52 85 e3 f6 a4 11 cf 50 c8 6a 2f 5e 94 28 d7 07	0f 60 6f 5e d6 31 c0 b3 da 38 10 13 a9 bf 6b 01	ef a8 b6 db 44 52 71 0b a5 5b 25 ad 41 7f 3b 00	⊕ =
5	e0 c8 d9 85 92 63 b1 b8 7f 83 35 be e8 c0 50 01	e1 e8 35 97 4f fb c8 6c d2 fb 96 ae 9b ba 53 7c	e1 e8 35 97 fb c8 6c 4f 96 ae d2 fb 7c 9b ba 53	25 bd b6 4c d1 11 3a 4c a9 d1 33 c0 ad 68 8e b0	d4 7c ca 11 d1 83 f2 f9 c6 9d b8 15 f8 87 bc bc	⊕ =


6
f1 c1 7c 5d
00 92 c8 b5
6f 4c 8b d5
55 ef 32 0c
a1 78 10 4c
63 4f e8 d5
a8 29 3d 03
fc df 23 fe
a1 78 10 4c
4f e8 d5 63
3d 03 a8 29
fe fc df 23
4b 2c 33 37
86 4a 9d d2
8d 89 f4 18
6d 80 e8 d8
⊕
6d 11 db ca
88 0b f9 00
a3 3e 86 93
7a fd 41 fd
=
7
26 3d e8 fd
0e 41 64 d2
2e b7 72 8b
17 7d a9 25
f7 27 9b 54
ab 83 43 b5
31 a9 40 3d
f0 ff d3 3f
f7 27 9b 54
83 43 b5 ab
40 3d 31 a9
3f f0 ff d3
14 46 27 34
15 16 46 2a
b5 15 56 d8
bf ec d7 43
⊕
4e 5f 84 4e
54 5f a6 a6
f7 c9 4f dc
0e f3 b2 4f
=
8
5a 19 a3 7a
41 49 e0 8c
42 dc 19 04
b1 1f 65 0c
be d4 0a da
83 3b e1 64
2c 86 d4 f2
c8 c0 4d fe
be d4 0a da
3b e1 64 83
d4 f2 2c 86
fe c8 c0 4d
00 b1 54 fa
51 c8 76 1b
2f 89 6d 99
d1 ff cd ea
⊕
ea b5 31 7f
d2 8d 2b 8d
73 ba f5 29
21 d2 60 2f
=
9
ea 04 65 85
83 45 5d 96
5c 33 98 b0
f0 2d ad c5
87 f2 4d 97
ec 6e 4c 90
4a c3 46 e7
8c d8 95 a6
87 f2 4d 97
6e 4c 90 ec
46 e7 4a c3
a6 8c d8 95
47 40 a3 4c
37 d4 70 9f
94 e4 3a 42
ed a5 a6 bc
⊕
ac 19 28 57
77 fa d1 5c
66 dc 29 00
f3 21 41 6e
=
10
eb 59 8b 1b
40 2e a1 c3
f2 38 13 42
1e 84 e7 d2
e9 cb 3d af
09 31 32 2e
89 07 7d 2c
72 5f 94 b5
e9 cb 3d af
31 32 2e 09
7d 2c 89 07
b5 72 5f 94
⊕
d0 c9 e1 b6
14 ee 3f 63
f9 25 0c 0c
a8 89 c8 a6
=
output
39 02 dc 19
25 dc 11 6a
84 09 85 0b
1d fb 97 32

Appendix

This appendix contains example vectors [6] - including intermediate values – for all three AES key lengths (Nr = 4, 6, and 8), for the Cipher, Inverse Cipher, and Equivalent Inverse Cipher that are described in Sec. 5.1, 5.3, and 5.3.5, respectively.

All vectors are in hexadecimal notation, with each pair of characters giving a byte value in which the left character of each pair provides the bit pattern for the 4 bit group containing the higher numbered bits using the notation explained in Sec. 4.2, while the right character provides the bit pattern for the lower-numbered bits. The array index for all bytes (groups of two hexadecimal digits) within these test vectors starts at zero and increases from left to right.

```
Legend for CIPHER (ENCRYPT) (round number r = 0 to 10, 12 or 14):

   input:   cipher input
   start:   state at start of round[r]
   s_box:   state after SubBytes()
   s_row:   state after ShiftRows()
   m_col:   state after MixColumns()
   k_sch:   key schedule value for round[r]
   output:  cipher output


Legend for INVERSE CIPHER (DECRYPT) (round number r = 0 to 10, 12 or 14):

   iinput:  inverse cipher input
   istart:  state at start of round[r]
   is_box:  state after InvSubBytes()
   is_row:  state after InvShiftRows()
   ik_sch:  key schedule value for round[r]
   ik_add:  state after AddRoundKey()
   ioutput: inverse cipher output


Legend for EQUIVALENT INVERSE CIPHER (DECRYPT) (round number r = 0 to 10, 12
   or 14):

   iinput:  inverse cipher input
   istart:  state at start of round[r]
   is_box:  state after InvSubBytes()
   is_row:  state after InvShiftRows()
   im_col:  state after InvMixColumns()
   ik_sch:  key schedule value for round[r]
   ioutput: inverse cipher output
```

D.1 AES-128 (Nk=4, Nr=10)

```
PLAINTEXT:          00112233445566778899aabbccddeeff
KEY:                000102030405060708090a0b0c0d0e0f
```

```
CIPHER (ENCRYPT):
round[ 0].input     00112233445566778899aabbccddeeff
round[ 0].k_sch     000102030405060708090a0b0c0d0e0f
round[ 1].start     00102030405060708090a0b0c0d0e0f0
round[ 1].s_box     63cab7040953d051cd60e0e7ba70e18c
round[ 1].s_row     6353e08c0960e104cd70b751bacad0e7
round[ 1].m_col     5f72641557f5bc92f7be3b291db9f91a
round[ 1].k_sch     d6aa74fdd2af72fadaa678f1d6ab76fe
round[ 2].start     89d810e8855ace682d1843d8cb128fe4
round[ 2].s_box     a761ca9b97be8b45d8ad1a611fc97369
round[ 2].s_row     a7be1a6997ad739bd8c9ca451f618b61
round[ 2].m_col     ff87968431d86a51645151fa773ad009
round[ 2].k_sch     b692cf0b643dbdf1be9bc5006830b3fe
round[ 3].start     4915598f55e5d7a0daca94fa1f0a63f7
round[ 3].s_box     3b59cb73fcd90ee05774222dc067fb68
round[ 3].s_row     3bd92268fc74fb735767cbe0c0590e2d
round[ 3].m_col     4c9c1e66f771f0762c3f868e534df256
round[ 3].k_sch     b6ff744ed2c2c9bf6c590cbf0469bf41
round[ 4].start     fa636a2825b339c940668a3157244d17
round[ 4].s_box     2dfb02343f6d12dd09337ec75b36e3f0
round[ 4].s_row     2d6d7ef03f33e334093602dd5bfb12c7
round[ 4].m_col     6385b79ffc538df997be478e7547d691
round[ 4].k_sch     47f7f7bc95353e03f96c32bcfd058dfd
round[ 5].start     247240236966b3fa6ed2753288425b6c
round[ 5].s_box     36400926f9336d2d9fb59d23c42c3950
round[ 5].s_row     36339d50f9b539269f2c092dc4406d23
round[ 5].m_col     f4bcd45432e554d075f1d6c51dd03b3c
round[ 5].k_sch     3caaa3e8a99f9deb50f3af57adf622aa
round[ 6].start     c81677bc9b7ac93b25027992b0261996
round[ 6].s_box     e847f56514dadde23f77b64fe7f7d490
round[ 6].s_row     e8dab6901477d4653ff7f5e2e747dd4f
round[ 6].m_col     9816ee7400f87f556b2c049c8e5ad036
round[ 6].k_sch     5e390f7df7a69296a7553dc10aa31f6b
round[ 7].start     c62fe109f75eedc3cc79395d84f9cf5d
round[ 7].s_box     b415f8016858552e4bb6124c5f998a4c
round[ 7].s_row     b458124c68b68a014b99f82e5f15554c
round[ 7].m_col     c57e1c159a9bd286f05f4be098c63439
round[ 7].k_sch     14f9701ae35fe28c440adf4d4ea9c026
round[ 8].start     d1876c0f79c4300ab45594add66ff41f
round[ 8].s_box     3e175076b61c04678dfc2295f6a8bfc0
round[ 8].s_row     3e1c22c0b6fcbf768da85067f6170495
round[ 8].m_col     baa03de7a1f9b56ed5512cba5f414d23
round[ 8].k_sch     47438735a41c65b9e016baf4aebf7ad2
round[ 9].start     fde3bad205e5d0d73547964ef1fe37f1
round[ 9].s_box     5411f4b56bd9700e96a0902fa1bb9aa1
round[ 9].s_row     54d990a16ba09ab596bbf40ea111702f
round[ 9].m_col     e9f74eec023020f61bf2ccf2353c21c7
round[ 9].k_sch     549932d1f08557681093ed9cbe2c974e
round[10].start     bd6e7c3df2b5779e0b61216e8b10b689
round[10].s_box     7a9f102789d5f50b2beffd9f3dca4ea7
round[10].s_row     7ad5fda789ef4e272bca100b3d9ff59f
round[10].k_sch     13111d7fe3944a17f307a78b4d2b30c5
round[10].output    69c4e0d86a7b0430d8cdb78070b4c55a

INVERSE CIPHER (DECRYPT):
round[ 0].iinput    69c4e0d86a7b0430d8cdb78070b4c55a
round[ 0].ik_sch    13111d7fe3944a17f307a78b4d2b30c5
```

```
round[ 1].istart    7ad5fda789ef4e272bca100b3d9ff59f
round[ 1].is_row    7a9f102789d5f50b2beffd9f3dca4ea7
round[ 1].is_box    bd6e7c3df2b5779e0b61216e8b10b689
round[ 1].ik_sch    549932d1f08557681093ed9cbe2c974e
round[ 1].ik_add    e9f74eec023020f61bf2ccf2353c21c7
round[ 2].istart    54d990a16ba09ab596bbf40ea111702f
round[ 2].is_row    5411f4b56bd9700e96a0902fa1bb9aa1
round[ 2].is_box    fde3bad205e5d0d73547964ef1fe37f1
round[ 2].ik_sch    47438735a41c65b9e016baf4aebf7ad2
round[ 2].ik_add    baa03de7a1f9b56ed5512cba5f414d23
round[ 3].istart    3e1c22c0b6fcbf768da85067f6170495
round[ 3].is_row    3e175076b61c04678dfc2295f6a8bfc0
round[ 3].is_box    d1876c0f79c4300ab45594add66ff41f
round[ 3].ik_sch    14f9701ae35fe28c440adf4d4ea9c026
round[ 3].ik_add    c57e1c159a9bd286f05f4be098c63439
round[ 4].istart    b458124c68b68a014b99f82e5f15554c
round[ 4].is_row    b415f8016858552e4bb6124c5f998a4c
round[ 4].is_box    c62fe109f75eedc3cc79395d84f9cf5d
round[ 4].ik_sch    5e390f7df7a69296a7553dc10aa31f6b
round[ 4].ik_add    9816ee7400f87f556b2c049c8e5ad036
round[ 5].istart    e8dab6901477d4653ff7f5e2e747dd4f
round[ 5].is_row    e847f56514dadde23f77b64fe7f7d490
round[ 5].is_box    c81677bc9b7ac93b25027992b0261996
round[ 5].ik_sch    3caaa3e8a99f9deb50f3af57adf622aa
round[ 5].ik_add    f4bcd45432e554d075f1d6c51dd03b3c
round[ 6].istart    36339d50f9b539269f2c092dc4406d23
round[ 6].is_row    36400926f9336d2d9fb59d23c42c3950
round[ 6].is_box    247240236966b3fa6ed2753288425b6c
round[ 6].ik_sch    47f7f7bc95353e03f96c32bcfd058dfd
round[ 6].ik_add    6385b79ffc538df997be478e7547d691
round[ 7].istart    2d6d7ef03f33e334093602dd5bfb12c7
round[ 7].is_row    2dfb02343f6d12dd09337ec75b36e3f0
round[ 7].is_box    fa636a2825b339c940668a3157244d17
round[ 7].ik_sch    b6ff744ed2c2c9bf6c590cbf0469bf41
round[ 7].ik_add    4c9c1e66f771f0762c3f868e534df256
round[ 8].istart    3bd92268fc74fb735767cbe0c0590e2d
round[ 8].is_row    3b59cb73fcd90ee05774222dc067fb68
round[ 8].is_box    4915598f55e5d7a0daca94fa1f0a63f7
round[ 8].ik_sch    b692cf0b643dbdf1be9bc5006830b3fe
round[ 8].ik_add    ff87968431d86a51645151fa773ad009
round[ 9].istart    a7be1a6997ad739bd8c9ca451f618b61
round[ 9].is_row    a761ca9b97be8b45d8ad1a611fc97369
round[ 9].is_box    89d810e8855ace682d1843d8cb128fe4
round[ 9].ik_sch    d6aa74fdd2af72fadaa678f1d6ab76fe
round[ 9].ik_add    5f72641557f5bc92f7be3b291db9f91a
round[10].istart    6353e08c0960e104cd70b751bacad0e7
round[10].is_row    63cab7040953d051cd60e0e7ba70e18c
round[10].is_box    00102030405060708090a0b0c0d0e0f0
round[10].ik_sch    000102030405060708090a0b0c0d0e0f
round[10].ioutput   00112233445566778899aabbccddeeff

EQUIVALENT INVERSE CIPHER (DECRYPT):
round[ 0].iinput    69c4e0d86a7b0430d8cdb78070b4c55a
round[ 0].ik_sch    13111d7fe3944a17f307a78b4d2b30c5
round[ 1].istart    7ad5fda789ef4e272bca100b3d9ff59f
round[ 1].is_box    bdb52189f261b63d0b107c9e8b6e776e
round[ 1].is_row    bd6e7c3df2b5779e0b61216e8b10b689
```

```
round[ 1].im_col    4773b91ff72f354361cb018ea1e6cf2c
round[ 1].ik_sch    13aa29be9c8faff6f770f58000f7bf03
round[ 2].istart    54d990a16ba09ab596bbf40ea111702f
round[ 2].is_box    fde596f1054737d235febad7f1e3d04e
round[ 2].is_row    fde3bad205e5d0d73547964ef1fe37f1
round[ 2].im_col    2d7e86a339d9393ee6570a1101904e16
round[ 2].ik_sch    1362a4638f2586486bff5a76f7874a83
round[ 3].istart    3e1c22c0b6fcbf768da85067f6170495
round[ 3].is_box    d1c4941f7955f40fb46f6c0ad68730ad
round[ 3].is_row    d1876c0f79c4300ab45594add66ff41f
round[ 3].im_col    39daee38f4f1a82aaf432410c36d45b9
round[ 3].ik_sch    8d82fc749c47222be4dadc3e9c7810f5
round[ 4].istart    b458124c68b68a014b99f82e5f15554c
round[ 4].is_box    c65e395df779cf09ccf9e1c3842fed5d
round[ 4].is_row    c62fe109f75eedc3cc79395d84f9cf5d
round[ 4].im_col    9a39bf1d05b20a3a476a0bf79fe51184
round[ 4].ik_sch    72e3098d11c5de5f789dfe1578a2cccb
round[ 5].istart    e8dab6901477d4653ff7f5e2e747dd4f
round[ 5].is_box    c87a79969b0219bc2526773bb016c992
round[ 5].is_row    c81677bc9b7ac93b25027992b0261996
round[ 5].im_col    18f78d779a93eef4f6742967c47f5ffd
round[ 5].ik_sch    2ec410276326d7d26958204a003f32de
round[ 6].istart    36339d50f9b539269f2c092dc4406d23
round[ 6].is_box    2466756c69d25b236e4240fa8872b332
round[ 6].is_row    247240236966b3fa6ed2753288425b6c
round[ 6].im_col    85cf8bf472d124c10348f545329c0053
round[ 6].ik_sch    a8a2f5044de2c7f50a7ef79869671294
round[ 7].istart    2d6d7ef03f33e334093602dd5bfb12c7
round[ 7].is_box    fab38a1725664d2840246ac957633931
round[ 7].is_row    fa636a2825b339c940668a3157244d17
round[ 7].im_col    fc1fc1f91934c98210fbfb8da340eb21
round[ 7].ik_sch    c7c6e391e54032f1479c306d6319e50c
round[ 8].istart    3bd92268fc74fb735767cbe0c0590e2d
round[ 8].is_box    49e594f755ca638fda0a59a01f15d7fa
round[ 8].is_row    4915598f55e5d7a0daca94fa1f0a63f7
round[ 8].im_col    076518f0b52ba2fb7a15c8d93be45e00
round[ 8].ik_sch    a0db02992286d160a2dc029c2485d561
round[ 9].istart    a7be1a6997ad739bd8c9ca451f618b61
round[ 9].is_box    895a43e485188fe82d121068cbd8ced8
round[ 9].is_row    89d810e8855ace682d1843d8cb128fe4
round[ 9].im_col    ef053f7c8b3d32fd4d2a64ad3c93071a
round[ 9].ik_sch    8c56dff0825dd3f9805ad3fc8659d7fd
round[10].istart    6353e08c0960e104cd70b751bacad0e7
round[10].is_box    0050a0f04090e03080d02070c01060b0
round[10].is_row    00102030405060708090a0b0c0d0e0f0
round[10].ik_sch    000102030405060708090a0b0c0d0e0f
round[10].ioutput   00112233445566778899aabbccddeeff
```

D.2 AES-192 (*Nk*=6, *Nr*=12)

```
PLAINTEXT:  00112233445566778899aabbccddeeff
KEY:        000102030405060708090a0b0c0d0e0f1011121314151617

CIPHER (ENCRYPT):
round[ 0].input     00112233445566778899aabbccddeeff
```

```
round[ 0].k_sch     000102030405060708090a0b0c0d0e0f
round[ 1].start     00102030405060708090a0b0c0d0e0f0
round[ 1].s_box     63cab7040953d051cd60e0e7ba70e18c
round[ 1].s_row     6353e08c0960e104cd70b751bacad0e7
round[ 1].m_col     5f72641557f5bc92f7be3b291db9f91a
round[ 1].k_sch     101112131415161758462f2f95c43f4fe
round[ 2].start     4f63760643e0aa85aff8c9d041fa0de4
round[ 2].s_box     84fb386f1ae1ac977941dd70832dd769
round[ 2].s_row     84e1dd691a41d76f792d389783fbac70
round[ 2].m_col     9f487f794f955f662afc86abd7f1ab29
round[ 2].k_sch     544afef55847f0fa4856e2e95c43f4fe
round[ 3].start     cb02818c17d2af9c62aa64428bb25fd7
round[ 3].s_box     1f770c64f0b579deaaac432c3d37cf0e
round[ 3].s_row     1fb5430ef0accf64aa370cde3d77792c
round[ 3].m_col     b7a53ecbbf9d75a0c40efc79b674cc11
round[ 3].k_sch     40f949b31cbabd4d48f043b810b7b342
round[ 4].start     f75c7778a327c8ed8cfebfc1a6c37f53
round[ 4].s_box     684af5bc0acce85564bb0878242ed2ed
round[ 4].s_row     68cc08ed0abbd2bc642ef555244ae878
round[ 4].m_col     7a1e98bdacb6d1141a6944dd06eb2d3e
round[ 4].k_sch     58e151ab04a2a5557effb5416245080c
round[ 5].start     22ffc916a81474416496f19c64ae2532
round[ 5].s_box     9316dd47c2fa92834390a1de43e43f23
round[ 5].s_row     93faa123c2903f4743e4dd83431692de
round[ 5].m_col     aaa755b34cffe57cef6f98e1f01c13e6
round[ 5].k_sch     2ab54bb43a02f8f662e3a95d66410c08
round[ 6].start     80121e0776fd1d8a8d8c31bc965d1fee
round[ 6].s_box     cdc972c53854a47e5d64c765904cc028
round[ 6].s_row     cd54c7283864c0c55d4c727e90c9a465
round[ 6].m_col     921f748fd96e937d622d7725ba8ba50c
round[ 6].k_sch     f501857297448d7ebdf1c6ca87f33e3c
round[ 7].start     671ef1fd4e2a1e03dfdcb1ef3d789b30
round[ 7].s_box     8572a1542fe5727b9e86c8df27bc1404
round[ 7].s_row     85e5c8042f8614549ebca17b277272df
round[ 7].m_col     e913e7b18f507d4b227ef652758acbcc
round[ 7].k_sch     e510976183519b6934157c9ea351f1e0
round[ 8].start     0c0370d00c01e622166b8accd6db3a2c
round[ 8].s_box     fe7b5170fe7c8e93477f7e4bf6b98071
round[ 8].s_row     fe7c7e71fe7f807047b95193f67b8e4b
round[ 8].m_col     6cf5edf996eb0a069c4ef21cbfc25762
round[ 8].k_sch     1ea0372a995309167c439e77ff12051e
round[ 9].start     7255dad30fb80310e00d6c6b40d0527c
round[ 9].s_box     40fc5766766c7bcae1d7507f09700010
round[ 9].s_row     406c501076d70066e17057ca09fc7b7f
round[ 9].m_col     7478bcdce8a50b81d4327a9009188262
round[ 9].k_sch     dd7e0e887e2fff68608fc842f9dcc154
round[10].start     a906b254968af4e9b4bdb2d2f0c44336
round[10].s_box     d36f3720907ebf1e8d7a37b58c1c1a05
round[10].s_row     d37e3705907a1a208d1c371e8c6fbfb5
round[10].m_col     0d73cc2d8f6abe8b0cf2dd9bb83d422e
round[10].k_sch     859f5f237a8d5a3dc0c02952beefd63a
round[11].start     88ec930ef5e7e4b6cc32f4c906d29414
round[11].s_box     c4cedcabe694694e4b23bfdd6fb522fa
round[11].s_row     c494bffae62322ab4bb5dc4e6fce69dd
round[11].m_col     71d720933b6d677dc00b8f28238e0fb7
round[11].k_sch     de601e7827bcdf2ca223800fd8aeda32
round[12].start     afb73eeb1cd1b85162280f27fb20d585
```

```
round[12].s_box     79a9b2e99c3e6cd1aa3476cc0fb70397
round[12].s_row     793e76979c3403e9aab7b2d10fa96ccc
round[12].k_sch     a4970a331a78dc09c418c271e3a41d5d
round[12].output    dda97ca4864cdfe06eaf70a0ec0d7191

INVERSE CIPHER (DECRYPT):
round[ 0].iinput    dda97ca4864cdfe06eaf70a0ec0d7191
round[ 0].ik_sch    a4970a331a78dc09c418c271e3a41d5d
round[ 1].istart    793e76979c3403e9aab7b2d10fa96ccc
round[ 1].is_row    79a9b2e99c3e6cd1aa3476cc0fb70397
round[ 1].is_box    afb73eeb1cd1b85162280f27fb20d585
round[ 1].ik_sch    de601e7827bcdf2ca223800fd8aeda32
round[ 1].ik_add    71d720933b6d677dc00b8f28238e0fb7
round[ 2].istart    c494bffae62322ab4bb5dc4e6fce69dd
round[ 2].is_row    c4cedcabe694694e4b23bfdd6fb522fa
round[ 2].is_box    88ec930ef5e7e4b6cc32f4c906d29414
round[ 2].ik_sch    859f5f237a8d5a3dc0c02952beefd63a
round[ 2].ik_add    0d73cc2d8f6abe8b0cf2dd9bb83d422e
round[ 3].istart    d37e3705907a1a208d1c371e8c6fbfb5
round[ 3].is_row    d36f3720907ebf1e8d7a37b58c1c1a05
round[ 3].is_box    a906b254968af4e9b4bdb2d2f0c44336
round[ 3].ik_sch    dd7e0e887e2fff68608fc842f9dcc154
round[ 3].ik_add    7478bcdce8a50b81d4327a9009188262
round[ 4].istart    406c501076d70066e17057ca09fc7b7f
round[ 4].is_row    40fc5766766c7bcae1d7507f09700010
round[ 4].is_box    7255dad30fb80310e00d6c6b40d0527c
round[ 4].ik_sch    1ea0372a995309167c439e77ff12051e
round[ 4].ik_add    6cf5edf996eb0a069c4ef21cbfc25762
round[ 5].istart    fe7c7e71fe7f807047b95193f67b8e4b
round[ 5].is_row    fe7b5170fe7c8e93477f7e4bf6b98071
round[ 5].is_box    0c0370d00c01e622166b8accd6db3a2c
round[ 5].ik_sch    e510976183519b6934157c9ea351f1e0
round[ 5].ik_add    e913e7b18f507d4b227ef652758acbcc
round[ 6].istart    85e5c8042f8614549ebca17b277272df
round[ 6].is_row    8572a1542fe5727b9e86c8df27bc1404
round[ 6].is_box    671ef1fd4e2a1e03dfdcb1ef3d789b30
round[ 6].ik_sch    f501857297448d7ebdf1c6ca87f33e3c
round[ 6].ik_add    921f748fd96e937d622d7725ba8ba50c
round[ 7].istart    cd54c7283864c0c55d4c727e90c9a465
round[ 7].is_row    cdc972c53854a47e5d64c765904cc028
round[ 7].is_box    80121e0776fd1d8a8d8c31bc965d1fee
round[ 7].ik_sch    2ab54bb43a02f8f662e3a95d66410c08
round[ 7].ik_add    aaa755b34cffe57cef6f98e1f01c13e6
round[ 8].istart    93faa123c2903f4743e4dd83431692de
round[ 8].is_row    9316dd47c2fa92834390a1de43e43f23
round[ 8].is_box    22ffc916a81474416496f19c64ae2532
round[ 8].ik_sch    58e151ab04a2a5557effb5416245080c
round[ 8].ik_add    7a1e98bdacb6d1141a6944dd06eb2d3e
round[ 9].istart    68cc08ed0abbd2bc642ef555244ae878
round[ 9].is_row    684af5bc0acce85564bb0878242ed2ed
round[ 9].is_box    f75c7778a327c8ed8cfebfc1a6c37f53
round[ 9].ik_sch    40f949b31cbabd4d48f043b810b7b342
round[ 9].ik_add    b7a53ecbbf9d75a0c40efc79b674cc11
round[10].istart    1fb5430ef0accf64aa370cde3d77792c
round[10].is_row    1f770c64f0b579deaaac432c3d37cf0e
round[10].is_box    cb02818c17d2af9c62aa64428bb25fd7
round[10].ik_sch    544afef55847f0fa4856e2e95c43f4fe
```

```
round[10].ik_add    9f487f794f955f662afc86abd7f1ab29
round[11].istart    84e1dd691a41d76f792d389783fbac70
round[11].is_row    84fb386f1ae1ac977941dd70832dd769
round[11].is_box    4f63760643e0aa85aff8c9d041fa0de4
round[11].ik_sch    101112131415161758462f95c43f4fe
round[11].ik_add    5f72641557f5bc92f7be3b291db9f91a
round[12].istart    6353e08c0960e104cd70b751bacad0e7
round[12].is_row    63cab7040953d051cd60e0e7ba70e18c
round[12].is_box    00102030405060708090a0b0c0d0e0f0
round[12].ik_sch    000102030405060708090a0b0c0d0e0f
round[12].ioutput   00112233445566778899aabbccddeeff

EQUIVALENT INVERSE CIPHER (DECRYPT):
round[ 0].iinput    dda97ca4864cdfe06eaf70a0ec0d7191
round[ 0].ik_sch    a4970a331a78dc09c418c271e3a41d5d
round[ 1].istart    793e76979c3403e9aab7b2d10fa96ccc
round[ 1].is_box    afd10f851c28d5eb62203e51fbb7b827
round[ 1].is_row    afb73eeb1cd1b85162280f27fb20d585
round[ 1].im_col    122a02f7242ac8e20605afce51cc7264
round[ 1].ik_sch    d6bebd0dc209ea494db073803e021bb9
round[ 2].istart    c494bffae62322ab4bb5dc4e6fce69dd
round[ 2].is_box    88e7f414f532940eccd293b606ece4c9
round[ 2].is_row    88ec930ef5e7e4b6cc32f4c906d29414
round[ 2].im_col    5cc7aecce3c872194ae5ef8309a933c7
round[ 2].ik_sch    8fb999c973b26839c7f9d89d85c68c72
round[ 3].istart    d37e3705907a1a208d1c371e8c6fbfb5
round[ 3].is_box    a98ab23696bd4354b4c4b2e9f006f4d2
round[ 3].is_row    a906b254968af4e9b4bdb2d2f0c44336
round[ 3].im_col    b7113ed134e85489b20866b51d4b2c3b
round[ 3].ik_sch    f77d6ec1423f54ef5378317f14b75744
round[ 4].istart    406c501076d70066e17057ca09fc7b7f
round[ 4].is_box    72b86c7c0f0d52d3e0d0da104055036b
round[ 4].is_row    7255dad30fb80310e00d6c6b40d0527c
round[ 4].im_col    ef3b1be1b9b0e64bdcb79f1e0a707fbb
round[ 4].ik_sch    1147659047cf663b9b0ece8dfc0bf1f0
round[ 5].istart    fe7c7e71fe7f807047b95193f67b8e4b
round[ 5].is_box    0c018a2c0c6b3ad016db7022d603e6cc
round[ 5].is_row    0c0370d00c01e622166b8accd6db3a2c
round[ 5].im_col    592460b248832b2952e0b831923048f1
round[ 5].ik_sch    dcc1a8b667053f7dcc5c194ab5423a2e
round[ 6].istart    85e5c8042f8614549ebca17b277272df
round[ 6].is_box    672ab1304edc9bfddf78f1033d1e1eef
round[ 6].is_row    671ef1fd4e2a1e03dfdcb1ef3d789b30
round[ 6].im_col    0b8a7783417ae3a1f9492dc0c641a7ce
round[ 6].ik_sch    c6deb0ab791e2364a4055fbe568803ab
round[ 7].istart    cd54c7283864c0c55d4c727e90c9a465
round[ 7].is_box    80fd31ee768c1f078d5d1e8a96121dbc
round[ 7].is_row    80121e0776fd1d8a8d8c31bc965d1fee
round[ 7].im_col    4ee1ddf9301d6352c9ad769ef8d20515
round[ 7].ik_sch    dd1b7cdaf28d5c158a49ab1dbbc497cb
round[ 8].istart    93faa123c2903f4743e4dd83431692de
round[ 8].is_box    2214f132a896251664aec94164ff749c
round[ 8].is_row    22ffc916a81474416496f19c64ae2532
round[ 8].im_col    1008ffe53b36ee6af27b42549b8a7bb7
round[ 8].ik_sch    78c4f708318d3cd69655b701bfc093cf
round[ 9].istart    68cc08ed0abbd2bc642ef555244ae878
round[ 9].is_box    f727bf53a3fe7f788cc377eda65cc8c1
```

```
round[ 9].is_row    f75c7778a327c8ed8cfebfc1a6c37f53
round[ 9].im_col    7f69ac1ed939ebaac8ece3cb12e159e3
round[ 9].ik_sch    60dcef10299524ce62dbef152f9620cf
round[10].istart    1fb5430ef0accf64aa370cde3d77792c
round[10].is_box    cbd264d717aa5f8c62b2819c8b02af42
round[10].is_row    cb02818c17d2af9c62aa64428bb25fd7
round[10].im_col    cfaf16b2570c18b52e7fef50cab267ae
round[10].ik_sch    4b4ecbdb4d4dcfda5752d7c74949cbde
round[11].istart    84e1dd691a41d76f792d389783fbac70
round[11].is_box    4fe0c9e443f80d06affa76854163aad0
round[11].is_row    4f63760643e0aa85aff8c9d041fa0de4
round[11].im_col    794cf891177bfd1d8a327086f3831b39
round[11].ik_sch    1a1f181d1e1b1c194742c7d74949cbde
round[12].istart    6353e08c0960e104cd70b751bacad0e7
round[12].is_box    0050a0f04090e03080d02070c01060b0
round[12].is_row    00102030405060708090a0b0c0d0e0f0
round[12].ik_sch    000102030405060708090a0b0c0d0e0f
round[12].ioutput   00112233445566778899aabbccddeeff
```

D.3 AES-256 (Nk=8, Nr=14)

```
PLAINTEXT:  00112233445566778899aabbccddeeff
KEY:        000102030405060708090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f

CIPHER (ENCRYPT):
round[ 0].input     00112233445566778899aabbccddeeff
round[ 0].k_sch     000102030405060708090a0b0c0d0e0f
round[ 1].start     00102030405060708090a0b0c0d0e0f0
round[ 1].s_box     63cab7040953d051cd60e0e7ba70e18c
round[ 1].s_row     6353e08c0960e104cd70b751bacad0e7
round[ 1].m_col     5f72641557f5bc92f7be3b291db9f91a
round[ 1].k_sch     101112131415161718191a1b1c1d1e1f
round[ 2].start     4f63760643e0aa85efa7213201a4e705
round[ 2].s_box     84fb386f1ae1ac97df5cfd237c49946b
round[ 2].s_row     84e1fd6b1a5c946fdf4938977cfbac23
round[ 2].m_col     bd2a395d2b6ac438d192443e615da195
round[ 2].k_sch     a573c29fa176c498a97fce93a572c09c
round[ 3].start     1859fbc28a1c00a078ed8aadc42f6109
round[ 3].s_box     adcb0f257e9c63e0bc557e951c15ef01
round[ 3].s_row     ad9c7e017e55ef25bc150fe01ccb6395
round[ 3].m_col     810dce0cc9db8172b3678c1e88a1b5bd
round[ 3].k_sch     1651a8cd0244beda1a5da4c10640bade
round[ 4].start     975c66c1cb9f3fa8a93a28df8ee10f63
round[ 4].s_box     884a33781fdb75c2d380349e19f876fb
round[ 4].s_row     88db34fb1f807678d3f833c2194a759e
round[ 4].m_col     b2822d81abe6fb275faf103a078c0033
round[ 4].k_sch     ae87dff00ff11b68a68ed5fb03fc1567
round[ 5].start     1c05f271a417e04ff921c5c104701554
round[ 5].s_box     9c6b89a349f0e18499fda678f2515920
round[ 5].s_row     9cf0a62049fd59a399518984f26be178
round[ 5].m_col     aeb65ba974e0f822d73f567bdb64c877
round[ 5].k_sch     6de1f1486fa54f9275f8eb5373b8518d
round[ 6].start     c357aae11b45b7b0a2c7bd28a8dc99fa
round[ 6].s_box     2e5bacf8af6ea9e73ac67a34c286ee2d
round[ 6].s_row     2e6e7a2dafc6eef83a86ace7c25ba934
```

```
round[ 6].m_col     b951c33c02e9bd29ae25cdb1efa08cc7
round[ 6].k_sch     c656827fc9a799176f294cec6cd5598b
round[ 7].start     7f074143cb4e243ec10c815d8375d54c
round[ 7].s_box     d2c5831a1f2f36b278fe0c4cec9d0329
round[ 7].s_row     d22f0c291ffe031a789d83b2ecc5364c
round[ 7].m_col     ebb19e1c3ee7c9e87d7535e9ed6b9144
round[ 7].k_sch     3de23a75524775e727bf9eb45407cf39
round[ 8].start     d653a4696ca0bc0f5acaab5db96c5e7d
round[ 8].s_box     f6ed49f950e06576be74624c565058ff
round[ 8].s_row     f6e062ff507458f9be50497656ed654c
round[ 8].m_col     5174c8669da98435a8b3e62ca974a5ea
round[ 8].k_sch     0bdc905fc27b0948ad5245a4c1871c2f
round[ 9].start     5aa858395fd28d7d05e1a38868f3b9c5
round[ 9].s_box     bec26a12cfb55dff6bf80ac4450d56a6
round[ 9].s_row     beb50aa6cff856126b0d6aff45c25dc4
round[ 9].m_col     0f77ee31d2ccadc05430a83f4ef96ac3
round[ 9].k_sch     45f5a66017b2d387300d4d33640a820a
round[10].start     4a824851c57e7e47643de50c2af3e8c9
round[10].s_box     d61352d1a6f3f3a04327d9fee50d9bdd
round[10].s_row     d6f3d9dda6279bd1430d52a0e513f3fe
round[10].m_col     bd86f0ea748fc4f4630f11c1e9331233
round[10].k_sch     7ccff71cbeb4fe5413e6bbf0d261a7df
round[11].start     c14907f6ca3b3aa070e9aa313b52b5ec
round[11].s_box     783bc54274e280e0511eacc7e200d5ce
round[11].s_row     78e2acce741ed5425100c5e0e23b80c7
round[11].m_col     af8690415d6e1dd387e5fbedd5c89013
round[11].k_sch     f01afafee7a82979d7a5644ab3afe640
round[12].start     5f9c6abfbac634aa50409fa766677653
round[12].s_box     cfde0208f4b418ac5309db5c338538ed
round[12].s_row     cfb4dbedf4093808538502ac33de185c
round[12].m_col     7427fae4d8a695269ce83d315be0392b
round[12].k_sch     2541fe719bf500258813bbd55a721c0a
round[13].start     516604954353950314fb86e401922521
round[13].s_box     d133f22a1aed2a7bfa0f44697c4f3ffd
round[13].s_row     d1ed44fd1a0f3f2afa4ff27b7c332a69
round[13].m_col     2c21a820306f154ab712c75eee0da04f
round[13].k_sch     4e5a6699a9f24fe07e572baacdf8cdea
round[14].start     627bceb9999d5aaac945ecf423f56da5
round[14].s_box     aa218b56ee5ebeacdd6ecebf26e63c06
round[14].s_row     aa5ece06ee6e3c56dde68bac2621bebf
round[14].k_sch     24fc79ccbf0979e9371ac23c6d68de36
round[14].output    8ea2b7ca516745bfeafc49904b496089

INVERSE CIPHER (DECRYPT):
round[ 0].iinput    8ea2b7ca516745bfeafc49904b496089
round[ 0].ik_sch    24fc79ccbf0979e9371ac23c6d68de36
round[ 1].istart    aa5ece06ee6e3c56dde68bac2621bebf
round[ 1].is_row    aa218b56ee5ebeacdd6ecebf26e63c06
round[ 1].is_box    627bceb9999d5aaac945ecf423f56da5
round[ 1].ik_sch    4e5a6699a9f24fe07e572baacdf8cdea
round[ 1].ik_add    2c21a820306f154ab712c75eee0da04f
round[ 2].istart    d1ed44fd1a0f3f2afa4ff27b7c332a69
round[ 2].is_row    d133f22a1aed2a7bfa0f44697c4f3ffd
round[ 2].is_box    516604954353950314fb86e401922521
round[ 2].ik_sch    2541fe719bf500258813bbd55a721c0a
round[ 2].ik_add    7427fae4d8a695269ce83d315be0392b
round[ 3].istart    cfb4dbedf4093808538502ac33de185c
```

round[3].is_row cfde0208f4b418ac5309db5c338538ed
round[3].is_box 5f9c6abfbac634aa50409fa766677653
round[3].ik_sch f01afafee7a82979d7a5644ab3afe640
round[3].ik_add af8690415d6e1dd387e5fbedd5c89013
round[4].istart 78e2acce741ed5425100c5e0e23b80c7
round[4].is_row 783bc54274e280e0511eacc7e200d5ce
round[4].is_box c14907f6ca3b3aa070e9aa313b52b5ec
round[4].ik_sch 7ccff71cbeb4fe5413e6bbf0d261a7df
round[4].ik_add bd86f0ea748fc4f4630f11c1e9331233
round[5].istart d6f3d9dda6279bd1430d52a0e513f3fe
round[5].is_row d61352d1a6f3f3a04327d9fee50d9bdd
round[5].is_box 4a824851c57e7e47643de50c2af3e8c9
round[5].ik_sch 45f5a66017b2d387300d4d33640a820a
round[5].ik_add 0f77ee31d2ccadc05430a83f4ef96ac3
round[6].istart beb50aa6cff856126b0d6aff45c25dc4
round[6].is_row bec26a12cfb55dff6bf80ac4450d56a6
round[6].is_box 5aa858395fd28d7d05e1a38868f3b9c5
round[6].ik_sch 0bdc905fc27b0948ad5245a4c1871c2f
round[6].ik_add 5174c8669da98435a8b3e62ca974a5ea
round[7].istart f6e062ff507458f9be50497656ed654c
round[7].is_row f6ed49f950e06576be74624c565058ff
round[7].is_box d653a4696ca0bc0f5acaab5db96c5e7d
round[7].ik_sch 3de23a75524775e727bf9eb45407cf39
round[7].ik_add ebb19e1c3ee7c9e87d7535e9ed6b9144
round[8].istart d22f0c291ffe031a789d83b2ecc5364c
round[8] is_row d2c5831a1f2f36b278fe0c4cec9d0329
round[8].is_box 7f074143cb4e243ec10c815d8375d54c
round[8].ik_sch c656827fc9a799176f294cec6cd5598b
round[8].ik_add b951c33c02e9bd29ae25cdb1efa08cc7
round[9].istart 2e6e7a2dafc6eef83a86ace7c25ba934
round[9].is_row 2e5bacf8af6ea9e73ac67a34c286ee2d
round[9].is_box c357aae11b45b7b0a2c7bd28a8dc99fa
round[9].ik_sch 6de1f1486fa54f9275f8eb5373b8518d
round[9].ik_add aeb65ba974e0f822d73f567bdb64c877
round[10].istart 9cf0a62049fd59a399518984f26be178
round[10].is_row 9c6b89a349f0e18499fda678f2515920
round[10].is_box 1c05f271a417e04ff921c5c104701554
round[10].ik_sch ae87dff00ff11b68a68ed5fb03fc1567
round[10].ik_add b2822d81abe6fb275faf103a078c0033
round[11].istart 28db34fb1f807678d3f833c2194a759e
round[11].is_row 884a33781fdb75c2d380349e19f876fb
round[11].is_box 975c66c1cb9f3fa8a93a28df8ee10f63
round[11].ik_sch 1651a8cd0244beda1a5da4c10640bade
round[11].ik_add 810dce0cc9db8172b3678c1e88a1b5bd
round[12].istart ad9c7e017e55ef25bc150fe01ccb6395
round[12].is_row adcb0f257e9c63e0bc557e951c15ef01
round[12].is_box 1859fbc28a1c00a078ed8aadc42f6109
round[12].ik_sch a573c29fa176c498a97fce93a572c09c
round[12].ik_add bd2a395d2b6ac438d192443e615da195
round[13].istart 84e1fd6b1a5c946fdf4938977cfbac23
round[13].is_row 84fb386f1ae1ac97df5cfd237c49946b
round[13].is_box 4f63760643e0aa85efa7213201a4e705
round[13].ik_sch 101112131415161718191a1b1c1d1e1f
round[13].ik_add 5f72641557f5bc92f7be3b291db9f91a
round[14].istart 6353e08c0960e104cd70b751bacad0e7
round[14].is_row 63cab7040953d051cd60e0e7ba70e18c
round[14].is_box 000102030405060708090a0b0c0d0e0f