A Database-based Email System

Rukma Prabhu Verlekar
Neela Sawant
Meghana Kshirsagar

Computer Science and Engineering
IIT Bombay

Oct 31, 2004

Introduction

Goal

To build a database-based email-system.

Motivation: Short-comings in existing email-systems

- Most email-systems have a File-system at the back-end.

Our system aims at improving on these.

Introduction

Goal

To build a database-based email-system.

Motivation: Short-comings in existing email-systems

- Most email-systems have a File-system at the back-end.
- No authentication system while sending mail.
- No support for searching attachments.

Our system aims at improving on these.

Introduction

Goal

To build a database-based email-system.

Motivation: Short-comings in existing email-systems

- Most email-systems have a File-system at the back-end.
- No authentication system while sending mail.
- No support for searching attachments.

Our system aims at improving on these.

- Database system at back-end.
- Authentication of users even when accepting send requests.
- Support for user-defined labels,searching attachments,search on labels.

Introduction

Goal

To build a database-based email-system.

Motivation: Short-comings in existing email-systems

- Most email-systems have a File-system at the back-end.
- No authentication system while sending mail.
- No support for searching attachments.

Our system aims at improving on these.

- Database system at back-end.
- Authentication of users even when accepting send requests.
- Support for user-defined labels,searching attachments,search on labels.

Introduction

Goal

To build a database-based email-system.

Motivation: Short-comings in existing email-systems

- Most email-systems have a File-system at the back-end.
- No authentication system while sending mail.
- No support for searching attachments.

Our system aims at improving on these.

- Database system at back-end.
- Authentication of users even when accepting send requests.
- Support for user-defined labels,searching attachments,search on labels.

Introduction

Goal

To build a database-based email-system.

Motivation: Short-comings in existing email-systems

- Most email-systems have a File-system at the back-end.
- No authentication system while sending mail.
- No support for searching attachments.

Our system aims at improving on these.

- Database system at back-end.
- Authentication of users even when accepting send requests.
- Support for user-defined labels,searching attachments,search on labels.

Our System

Components of our System:

- An SMTP server.
- An LDAP directory server for authentication of users.
- A Database at the back end.

Our System

Components of our System:

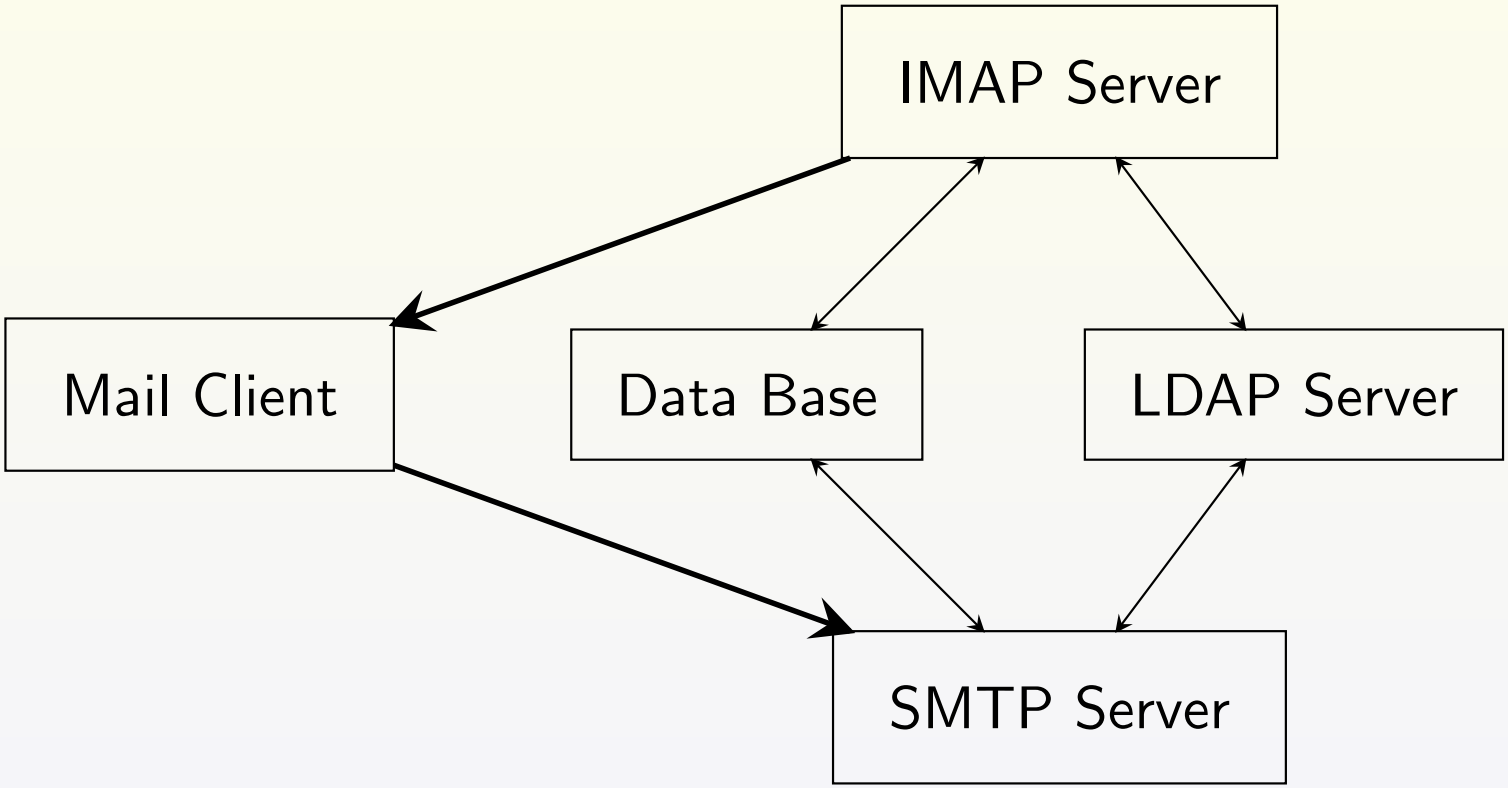
- An SMTP server.
- An LDAP directory server for authentication of users.
- A Database at the back end.

Our System

Components of our System:

- An SMTP server.
- An LDAP directory server for authentication of users.
- A Database at the back end.

Bird's Eye View of our System



SMTP

What is SMTP?

The Simple Mail Transfer Protocol (SMTP) is Internet's standard host-to-host mail transport protocol.

- It is a server-server protocol.
- SMTP commands:
  - EHLO
  - MAIL FRM:
  - RCPT TO:
  - DATA
  - VRFY
  - EXPN

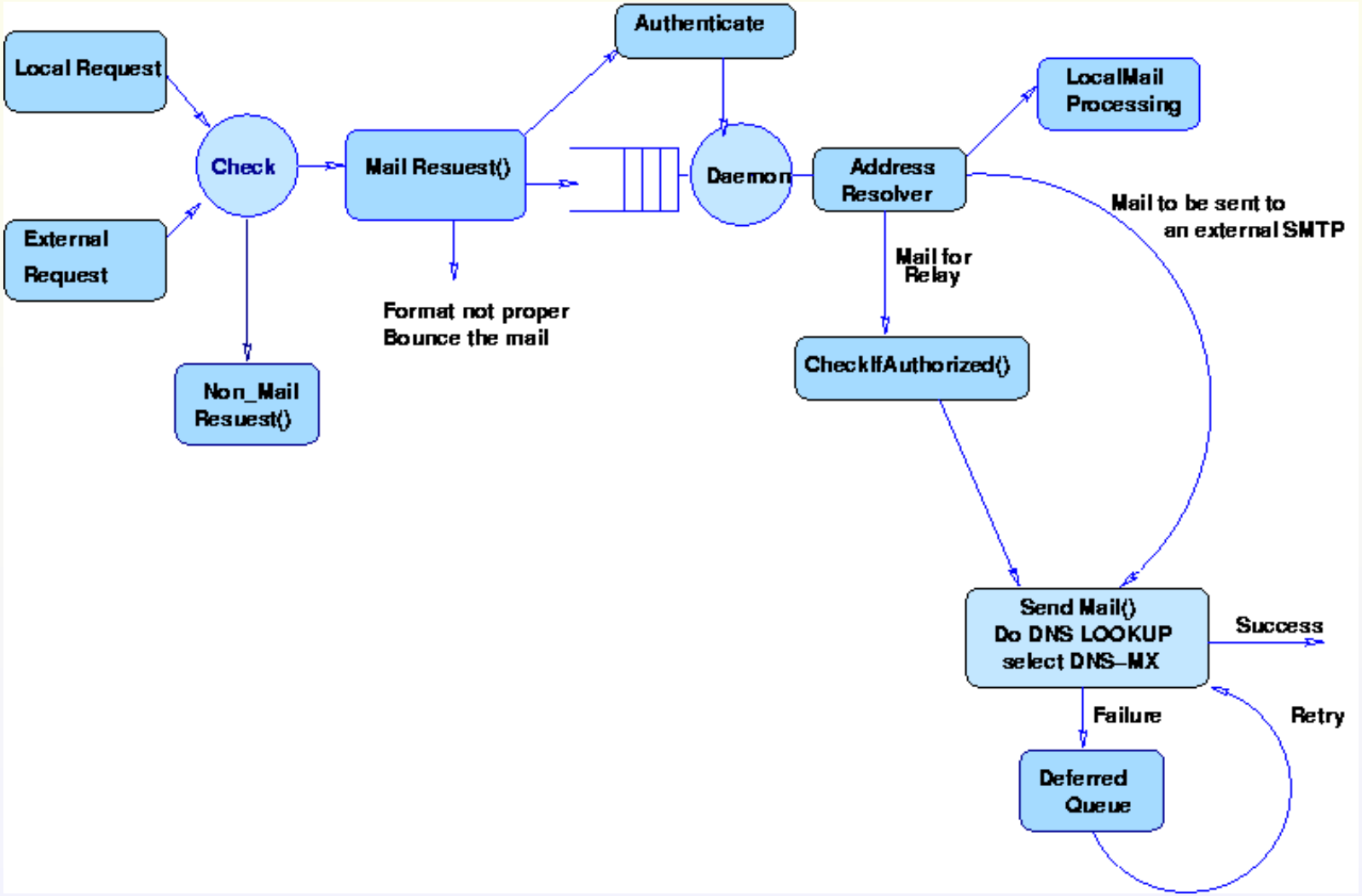
SMTP

What is SMTP?

The Simple Mail Transfer Protocol (SMTP) is Internet's standard host-to-host mail transport protocol.

- It is a server-server protocol.
- SMTP commands:
  - EHLO
  - MAIL FRM:
  - RCPT TO:
  - DATA
  - VRFY
  - EXPN

Control Flow Diagram of the Mailing process



Processes

Check

- If non-mail request.(i.e if VRFY or EXPN)
  Call Non-mail-Request().
- Else If Mail-request Call Mail-Request().
- Else Error.

Non-mail-Request()

- If (command = VRFY or command = EXPN)
  Contact LDAP server to retrieve the requested information.
  (i.e information about user or mailing-list)

Processes Contd...

Mail-Request()

- Check the mail-format. If not proper bounce it.
- If ('From' belongs to my domain)
  it should be a 'send' request from a local client.
  Call Authenticate().
- Else Queue it.

Authenticate()

- If valid user, prompt for password.
  Contact LDAP server for verifying password.
- If authentication succeeds, Queue the request.
- Else Error.

Processes...contd

Address Resolver

- If ('To' belongs to my domain)
  it is an 'incoming' mail for my client.
  Call LocalMailProcessing().
- Else If ('From' my domain)
  it is a 'send' request to external destination.
  Call SendMail().
- Else it is a 'relay' request.
  Call CheckIfAuthorized().

CheckIfAuthorized()

- Check if the requested relay-destination is in the list of authorized-relay-destinations.
  If Yes
  Call SendMail().
  Else Reject it.

Processes Contd...

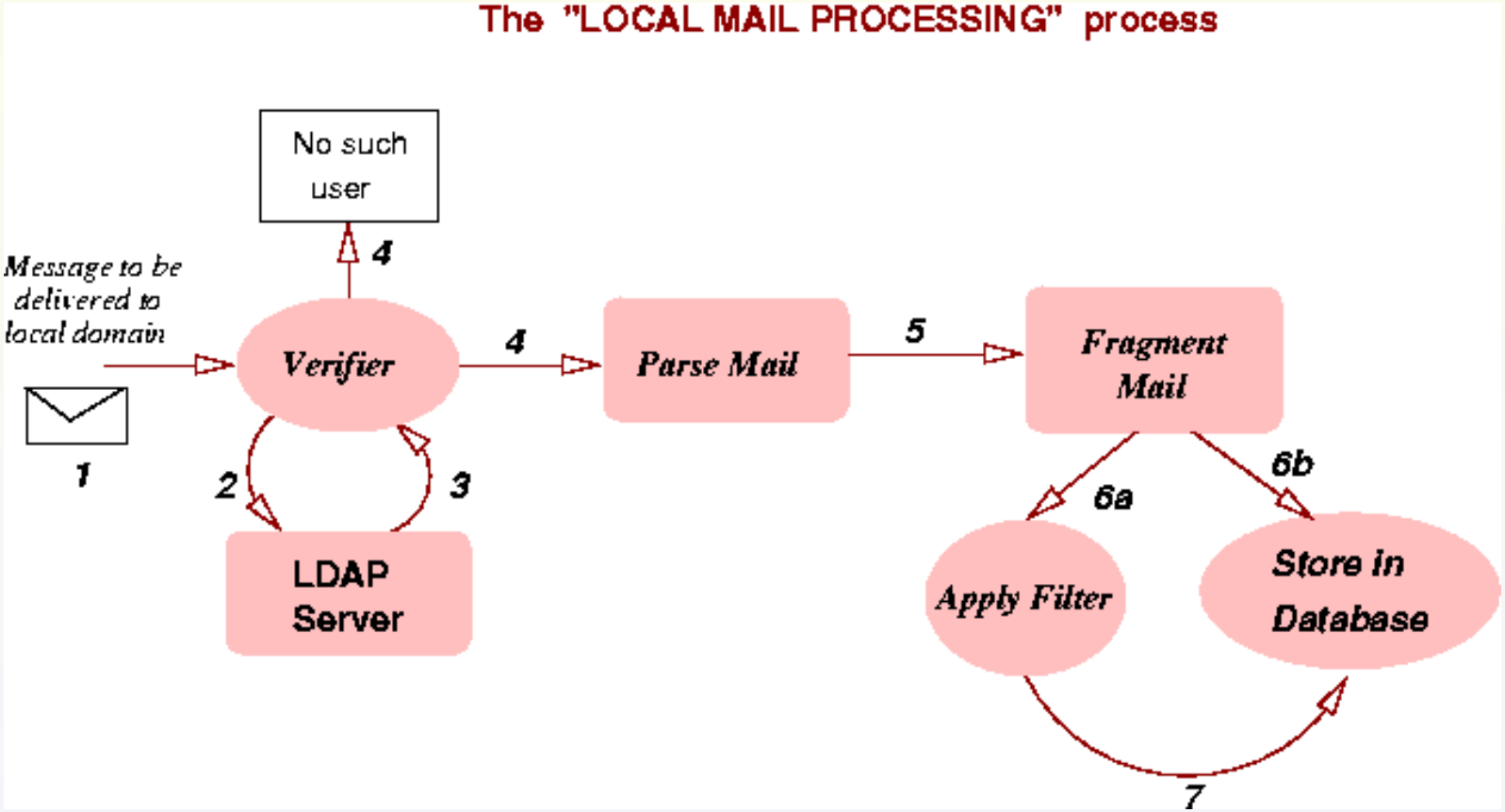
SendMail()

- Do DNSLookup
  Select the DNS-MX record with highest priority for the 'To' domain.
  Send the mail.

Deferred Queue

- If the Mail couldnt be sent successfully it is put into the deferred queue and the daemon tries to send it later.

Control Flow Diagram for Local Mail processing



Local Processing

Verifier

- Contacts the LDAP server,verifies if recipient exists on this domain. The recipient could either be one/more users (or) a mailing-list.

Parse and Fragment Mail

- Parse and fragment the mail to extract the header fields, body, attachments.
- All attachments are also seperated.

Apply Filters

- Contact the 'FILTERS' table and apply the filters defined by each recipient.
- A filter is applied by executing the SIEVE code corresponding to it.
- The filter determines which of the labels defined by the user to apply to the mail.

Why a Database?

Advantages

- A single copy of mails received for mailing-lists is stored.
- An optimized design of the database makes searching,faster retrieval of mails on different search keys such as subject, sender, header, body and so on.
- Efficient storage by stripping each mail into its components and storing them in a backend database.
- Easy maintenance of information and faster search through standard database techniques.

Why a Database?

Advantages

- A single copy of mails received for mailing-lists is stored.
- An optimized design of the database makes searching,faster retrieval of mails on different search keys such as subject, sender, header, body and so on.
- Efficient storage by stripping each mail into its components and storing them in a backend database.
- Easy maintenance of information and faster search through standard database techniques.

Why a Database?

Advantages

- A single copy of mails received for mailing-lists is stored.
- An optimized design of the database makes searching,faster retrieval of mails on different search keys such as subject, sender, header, body and so on.
- Efficient storage by stripping each mail into its components and storing them in a backend database.
- Easy maintenance of information and faster search through standard database techniques.

Why a Database?

Advantages

- A single copy of mails received for mailing-lists is stored.
- An optimized design of the database makes searching,faster retrieval of mails on different search keys such as subject, sender, header, body and so on.
- Efficient storage by stripping each mail into its components and storing them in a backend database.
- Easy maintenance of information and faster search through standard database techniques.

Database Schema - Tables

MAIL_MESSAGE Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MESSAGE_ID	NUMBER	NOT NULL	Primary key. Message-id associated with a mail.
POINTER_TO_BODY	INTEGER	NOT NULL	Points to location where body is stored in file-system.
SUBJECT	VARCHAR(15)	NULL	Subject field of mail.
DATE	DATE	NOT NULL	Date when mail received.
COUNT	INTEGER	NOT NULL	Number of mailboxes sharing the mail.(should be atleast one)
ATTACHMENT_FLAG	BOOLEAN	NULL	Flag to indicate if this mail has an attachment too.
SIZE	INTEGER	NOT NULL	Total size of mail.

More Tables

FROM Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MESSAGE_ID	NUMBER	NOT NULL	Foreign key of table MAIL_MESSAGE.
FROM	VARCHAR(15)	NOT NULL	Value associated with above field.

TO Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MESSAGE_ID	NUMBER	NOT NULL	Foreign key of table MAIL_MESSAGE.
TO	VARCHAR(15)	NOT NULL	Value associated with above field.

CC Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MESSAGE_ID	NUMBER	NOT NULL	Foreign key of table MAIL_MESSAGE.
CC	VARCHAR(15)	NOT NULL	Value associated with above field.

Tables Contd...

USER Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MAIL_BOX_ID	INTEGER	NOT NULL	Unique id of each user mailbox.
USER_ID	INTEGER	NOT NULL	Userid of user.

MESSAGE_LABEL Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MAIL_BOX_ID	INTEGER	NOT NULL	Mail-box-id of user. Foreign key to user table.
MESSAGE_ID	INTEGER	NOT NULL	Message-id of mail. FK to MAIL_MESSAGE TABLE.
LABEL_ID	INTEGER	NOT NULL	FK to LABEL TABLE. Will be different for each user and each mail.
FLAG	BOOLEAN	NULL	Deleted, Draft, Seen/unseen,Flagged, Recent, Unsent, Labelled, Answered.

Tables Contd...

SEQUENCES Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
SEQUENCE_NAME	VARCHAR(5)	(NOT NULL)	Name of the Sequence. Primary key.
CURRENT_VALUE	INTEGER	NOT NULL	Current value of the sequence.

ATTACHMENTS Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
ATTACHMENT_ID	VARCHAR(15)	NOT NULL	Identifier associated with attachment. (attribute#1 of composite key)
MESSAGE_ID	NUMBER	NOT NULL	Message-id of the mail. (attribute#2 of composite key)
POINTER	INTEGER	NULL	Points to Attachment in the file system.
SIZE	INTEGER	NULL	Size of attachment.
NAME	VARCHAR(15)	NULL	Name of attachment.

Tables Contd...

USER_LABEL Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MAIL_BOX_ID	INTEGER	NOT NULL	Mail-box-id of user. Foreign Key to USER TABLE
LABEL_ID	INTEGER	NOT NULL	Identifier associated with this label.
LABEL	VARCHAR(15)	NOT NULL	Label name.

FILTER Table Metadata			
FIELD	**DATA TYPE**	**NULL**	**DESCRIPTION**
MAIL_BOX_ID	INTEGER	NOT NULL	Mail-box-id of user. Foreign Key to USER TABLE.
FILTER_ID	INTEGER	NOT NULL	Id associated with a particular filter. Primary key.
SIEVE_CODEPTR	VARCHAR(15)	NOT NULL	Pointer to the sieve-code.

Stored Procedures

Input

Headers, Pointer to Body, Pointer(s) to Attachments, Names/Sizes of attachments, Count

- Headers: *From, *To, *CC, Subject, Message-ID, Date
  (* indicates that multiple values are possible)
- Pointer to Body: Points to location of body on the file-system. Body stored on file-system includes headers.
- Pointer(s) to Attachments: Points to location of attachments on the file-system.
- Count: Number of values in (To + CC + From) fields. Indicates how many users have received the same mail.

Tasks Performed

- Generate IDs from sequences for BODY-ID, ATTACHMENT-ID (if there are attachments)
- Insertions:
  - Row: (Message-ID, From) Table: 'FROM'
  - Row: (Message-ID, To) Table: 'TO'
  - Row: (Message-ID, CC) Table: 'CC'
  - Row: (Attachment-ID, Message-ID, Pointer-to-Attachment, Size, Name) Table: 'ATTACHMENTS' table.
  - Row: (Message-ID, POINTER to BODY, Subject, Date, Count, Attachment-flag, Size) Table: 'MAIL MESSAGE'

LDAP - An Introduction

- LDAP or Light-weight Directory Access Protocol is a TCP-IP based protocol for accessing a directory service.
- It has a client-server model.
- Used to store descriptive, attribute-based information.
- The information is arranged in a hierarchical structure.
- The data is optimised for reading.
- The slapd (Stand Alone LDAP daemon) daemon. Has two parts:
  - A Front-end that handles protocol communication with LDAP clients.
  - A Backend that handles database operations.
- Backends supported: LDBM, Shell, PASSWD.

Advantages of LDAP over Databases

Why LDAP for Authentication?

- LDAP directories are limited but specialized databases.
- Compared to databases, LDAP directories are.
  - Optimized for reads
  - Flexible
  - Easier to scale
  - Supported out of the box

Advantages of LDAP over Databases

Why LDAP for Authentication?

- LDAP directories are limited but specialized databases.
- Compared to databases, LDAP directories are.
  - Optimized for queries
  - Flexible
  - Easier to scale
  - Simpler and cheaper to maintain.

Advantages of LDAP over Databases

Why LDAP for Authentication?

- LDAP directories are limited but specialized databases.
- Compared to databases, LDAP directories are.
  - Optimized for queries
  - Flexible
  - Easier to scale
  - Simpler and cheaper to maintain.

Advantages of LDAP over Databases

Why LDAP for Authentication?

- LDAP directories are limited but specialized databases.
- Compared to databases, LDAP directories are.
  - Optimized for queries
  - Flexible
  - Easier to scale
  - Simpler and cheaper to maintain.

Advantages of LDAP over Databases

Why LDAP for Authentication?

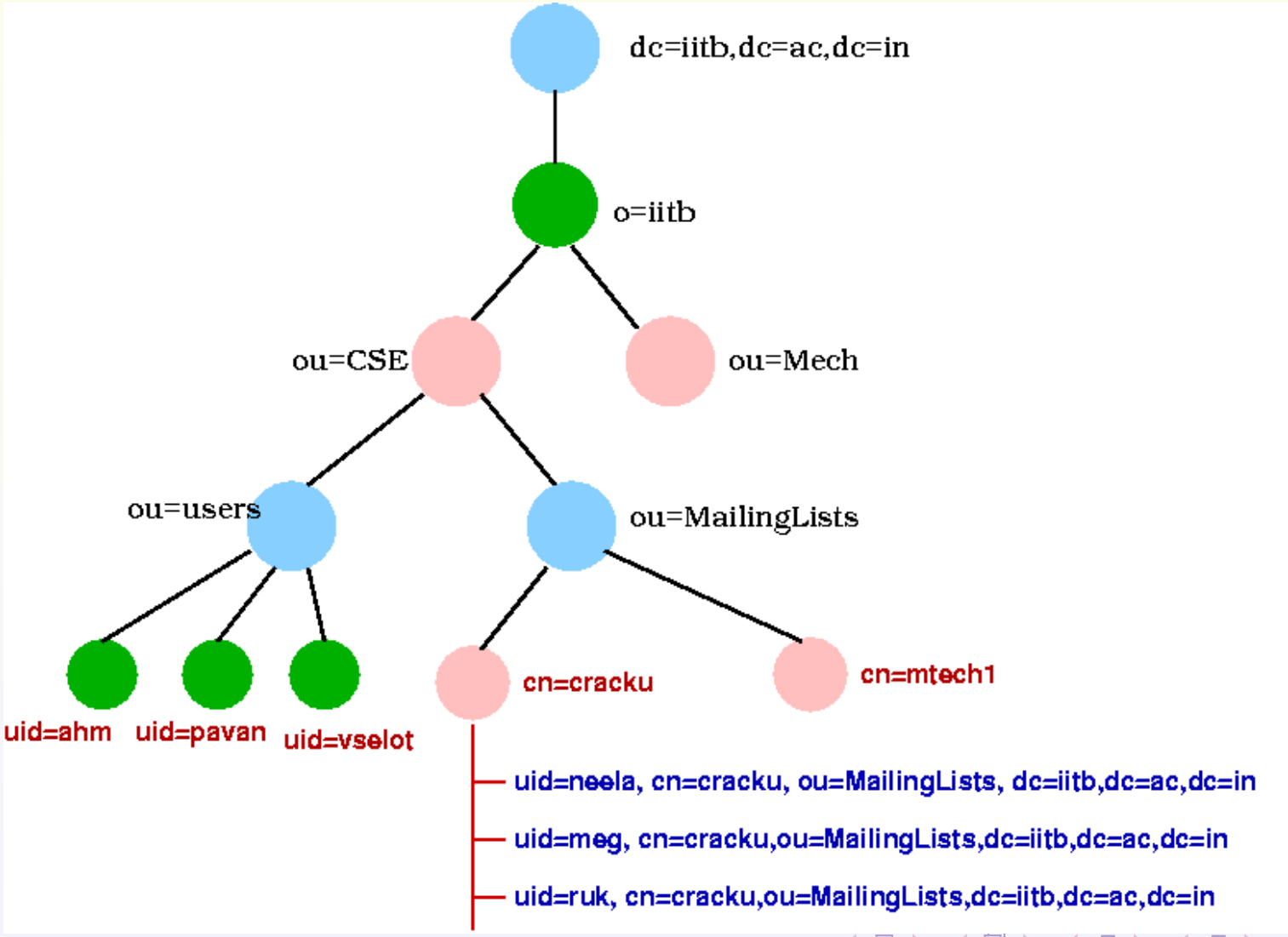
- LDAP directories are limited but specialized databases.
- Compared to databases, LDAP directories are.
  - Optimized for queries
  - Flexible
  - Easier to scale
  - Simpler and cheaper to maintain.

Advantages of LDAP over Databases

Why LDAP for Authentication?

- LDAP directories are limited but specialized databases.
- Compared to databases, LDAP directories are.
  - Optimized for queries
  - Flexible
  - Easier to scale
  - Simpler and cheaper to maintain.

Directory Structure



Definition of entries – Defining the 'user' entry

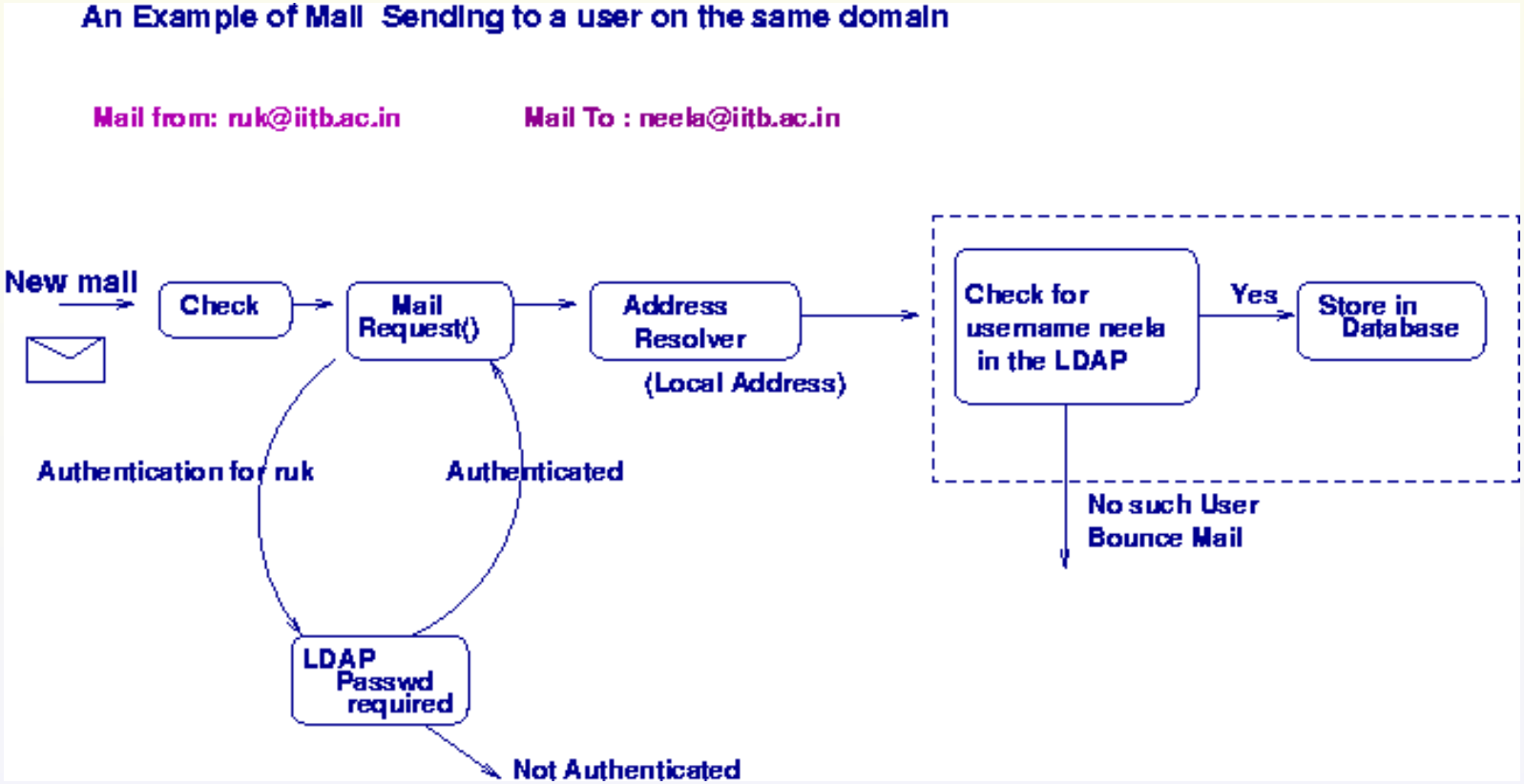
- Distinguished Name: uid="userid",ou=Users,ou=CSE,dc=iitb,dc=ac,dc=in
- objectclass: top account inetOrgPerson organizationalPerson
- attributes:
  - uid
  - mail
  - cn
  - givenname
  - postaladdress
  - phone
  - mailforwardingaddress
  - forwardstatus (TRUE or FALSE)
  - deliverymode
- Example: uid=neela,ou=Users,ou=CSE,dc=iitb,dc=ac,dc=in

Definition of entries

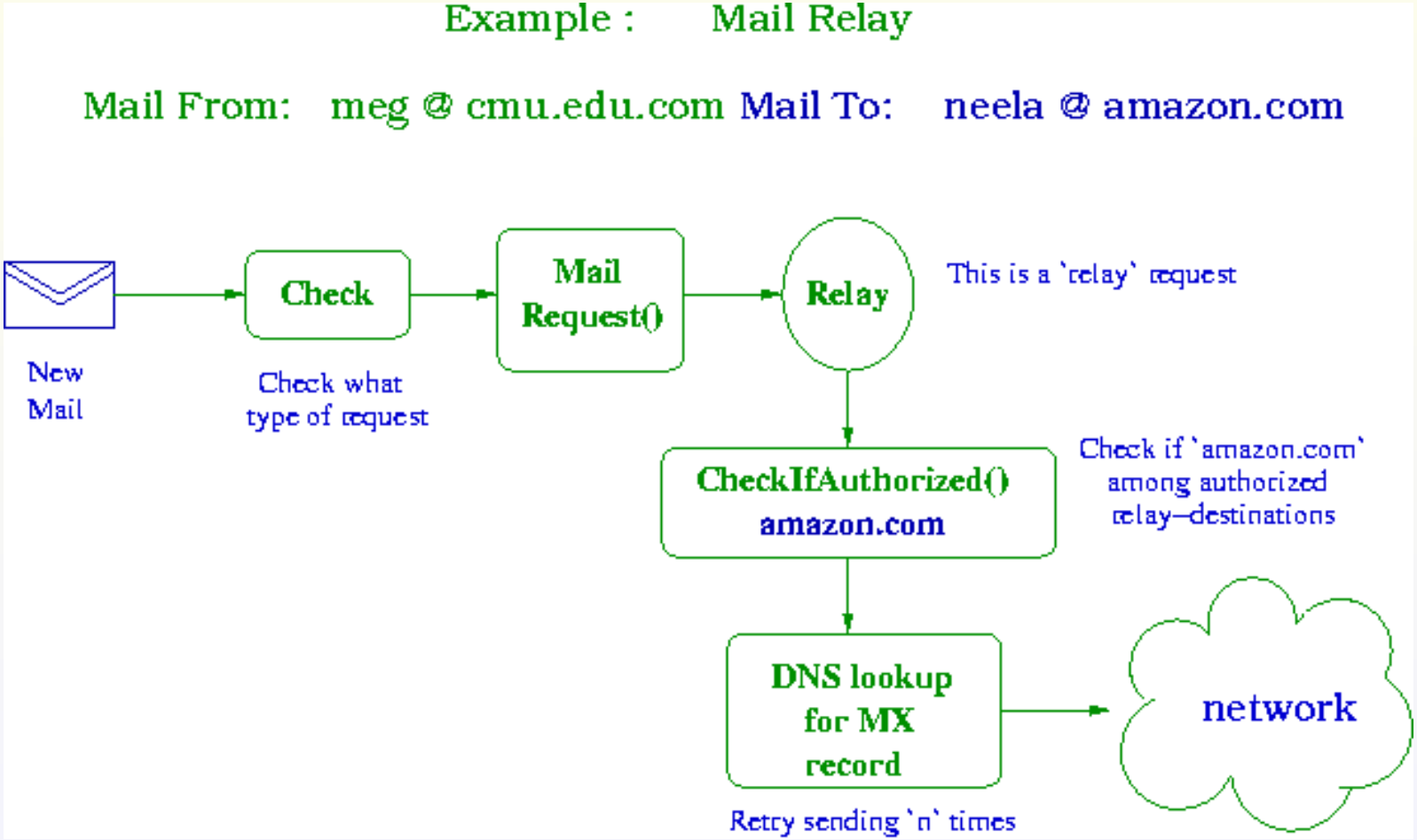
Definition of the 'mailing-list' entry:

- Distinguished Name: cn="ListName",ou=MailingLists,dc=iitb,dc=ac,dc=in (for mailing lists)
- objectclass: top groupOfUniqueNames
- attributes:
  - creationDate
  - modifiedDate
- Distinguished Name: uid=userid,cn="ListName",ou=MailingLists,dc=iitb,dc=ac,dc=in
- objectclass: top person organizationalPerson
- Example: cn=mtech1,ou=MailingLists,dc=iitb,dc=ac,dc=in

Example: Sending Mail



Example: Mail Relay



Advantages of our system

Advantages:

- Faster retrieval of data.
- Savings in storage.
- Faster search.
- Only authorized users allowed to send mail.
- Mailing-list support.

Advantages of our system

Advantages:

- Faster retrieval of data.
- Savings in storage.
- Faster search.
- Only authorized users allowed to send mail.
- Mailing-list support.

Advantages of our system

Advantages:

- Faster retrieval of data.
- Savings in storage.
- Faster search.
- Only authorized users allowed to send mail.
- Mailing-list support.

Advantages of our system

Advantages:

- Faster retrieval of data.
- Savings in storage.
- Faster search.
- Only authorized users allowed to send mail.
- Mailing-list support.

Advantages of our system

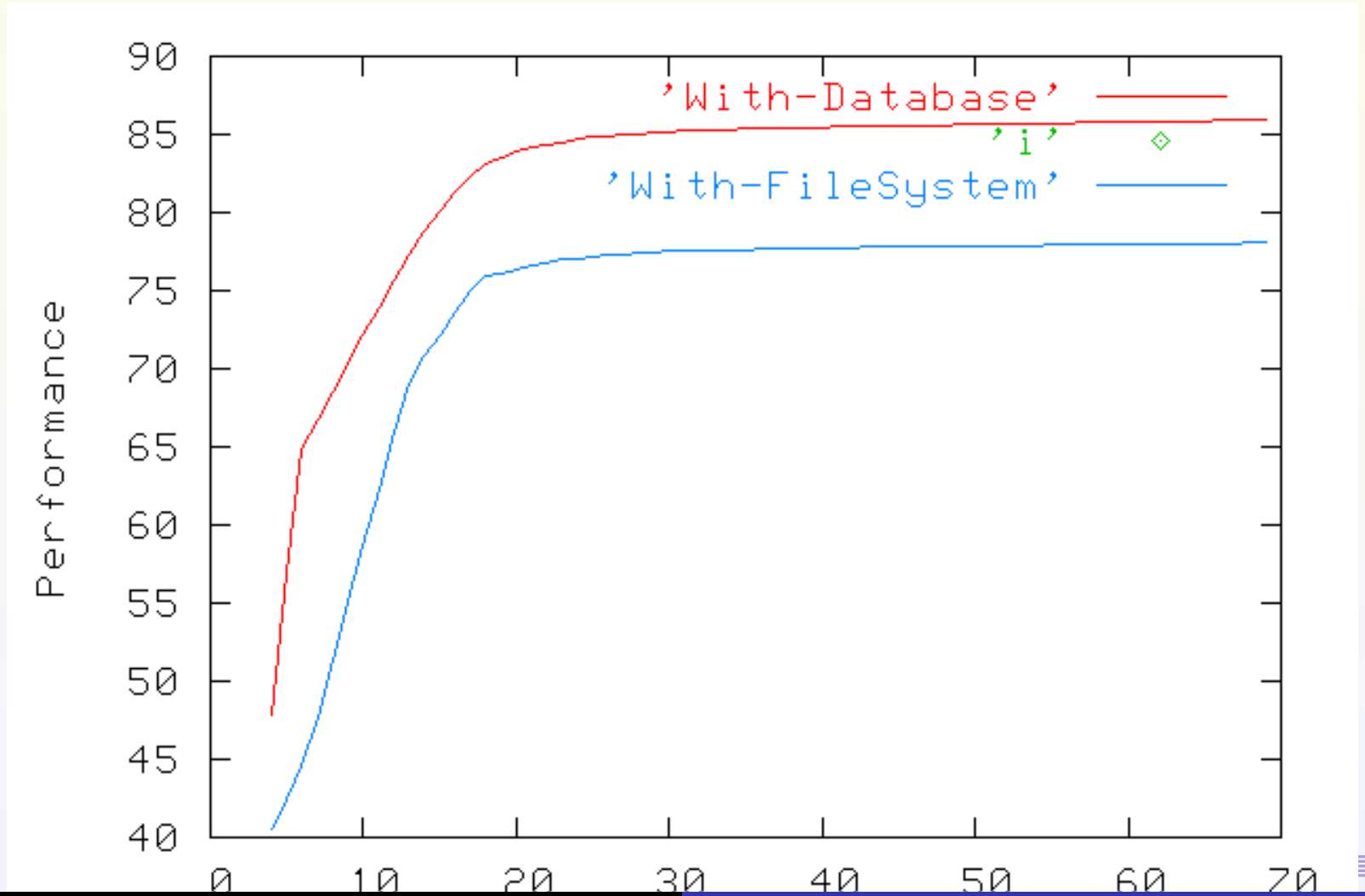
Advantages:

- Faster retrieval of data.
- Savings in storage.
- Faster search.
- Only authorized users allowed to send mail.
- Mailing-list support.

References

- http://www.zvon.org/tmRFC/output
- http://quark.humbug.org.au
- http://www.rscott.org/dns/mx.html
- http://www.umich.edu/ dirsvcs

Graph: Database based Email sytem Vs File system Based Email system



The End

THANK YOU