

# **datatree -- PigeonDeliver Database Structure, concepts and delivery process**

**Carlo Contavalli**

**ccontavalli at masobit.net**

## **Revision History**

Revision 1.0.0 2004/07/03  
First Document Revision

This document describes the structure of the PigeonDeliver (<http://deliver.pigeonair.net/>) Database, how users data is stored into the database, how roles and roles privileges are enforced and how a mail is finally delivered by the PigeonDeliver (<http://deliver.pigeonair.net/>) mailing system.

## **1. Before starting**

This document was written as part of the documentation of the PigeonAir Project to provide help and support to users, system administrators or developers.

While every effort has been made to ensure that the information is accurate at the time of publication, this document may contain errors, omissions, incongruences or wrong technical details. No liability for damages is accepted by the Author/Authors, the publishers or any other organization or person providing the information, arising from any errors or omissions that may appear, however caused.

In case you find an error, you would like to propose better solutions than those discussed in this document or you would like to discuss an idea regarding this document or its content, we would be glad to hear from you and please feel free to contact us by writing to the <[pigeon-dev at ml.pigeonair.net](mailto:pigeon-dev@ml.pigeonair.net)> mailing list or by directly contacting one of the authors.

## **1.1. Intended Audience**

This document is meant to introduce users and administrators to the PigeonDeliver (<http://deliver.pigeonair.net/>) mail delivery process and to provide all the information needed to understand users and administrators privileges, privileges enforcing mechanisms and configuration management.

However, this document does not provide any detail about the PigeonDeliver (<http://deliver.pigeonair.net/>) System Architecture, how services can be distributed among machines or how the system deals with privileges and services in a distributed environment.

## **1.2. Copyright Notice**

This document was written by Carlo Contavalli <[ccontavalli@masobit.net](mailto:ccontavalli@masobit.net)> and is thus Copyright (C) Carlo Contavalli 2003, 2004 and the PigeonAir Project.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts and no Back-Cover Texts.

Any example of program code available in this document should be considered protected by the terms of the GNU General Public License.

You should have received a copy of the GNU General Public License along with this document; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Trademarks are owned by their respective owners.

## **2. Mail delivery process**

Upon every smtpd received email, the PigeonDeliver (<http://deliver.pigeonair.net/>) database is looked up to verify:

- if the recipient exists in the local users database
- if it does not, the domain is looked up to verify if it does accept mail for unknown users
- if none of the conditions above apply, the mail is either considered for relay or rejected with the corresponding smtp error

In case the recipient is known in the local users database or in case the domain is known and accepts emails even for unknown users, the mail is accepted for local delivery and inserted into the “incoming” queue of the mailing system.

Once in the incoming queue, it is passed over to the PigeonDeliver (<http://deliver.pigeonair.net/>) delivery agent that attempts delivery from the “active” queue.

The PigeonDeliver (<http://deliver.pigeonair.net/>) delivery agent (mailMaster) takes the name of each recipient in turn, and splits it up into “username” and “domain”.

It then looks up the PigeonDeliver (<http://deliver.pigeonair.net/>) database looking for the “domain” configuration. Both the domain configuration and the user configuration is made up by a list of “modules” to be run to deliver the email.

Each module performs a specific task of the delivery process and has its own configuration parameters into the database.

Once the domain configuration is fetched, however, the delivery itself is started by calling each of the specified modules in turn. The order in which the modules are run, at time of writing, is determined by the Installer or System Administrator, who decides which modules are allowed and how they are to be called.

The first module that is usually called, if it is enabled in the domain configuration, is called “mailUsers”.

The mailUsers module contains a list of user specific configurations and takes care to fetch those configurations from the PigeonDeliver (<http://deliver.pigeonair.net/>) database.

Once user configurations are fetched, they are then merged with domain configurations by using a mailUsers specific scheme that will be documented later on in this document, and the delivery continues by calling the remaining PigeonDeliver (<http://deliver.pigeonair.net/>) Modules.

The remaining PigeonDeliver (<http://deliver.pigeonair.net/>) Modules are thus run using the mailUsers merged configurations as fetched from the database.

### **3. Configuration structure**

Before talking about how configurations are merged by the mailUsers module, it is necessary to briefly talk about how configurations are stored and how they are structured.

It is however necessary to make it clear from the beginning that the PigeonDeliver (<http://deliver.pigeonair.net/>) system is not bound to any particular database or database engine: the terms used below are those known by LDAP administrators since LDAP was the first database to be supported by PigeonDeliver (<http://deliver.pigeonair.net/>), but datatree handlers can be used to make PigeonDeliver (<http://deliver.pigeonair.net/>) work with other kinds of databases as well without any problem.

The database is considered by PigeonDeliver (<http://deliver.pigeonair.net/>) as a “tree” of structured data, or objects. Each object contains a list of properties.

Properties look like “variable=value” and can be multivalued, which means a variable may have multiple values. Properties are always string based, which means they will be converted to the proper type by the PigeonDeliver (<http://deliver.pigeonair.net/>) System.

Objects are hierarchically organized, much like a directory on the file system, where each object contains a list of properties and can have any number of children objects under it.

## 4. mailUsers Configuration merging

The mailUsers module expects at least two properties to be made available with each and every object in the PigeonDeliver (<http://deliver.pigeonair.net/>) tree:

- the “enable” property, which can be “TRUE” or “FALSE” valued
- the “force” property, which can be “TRUE” or “FALSE” valued

When merging user configurations with domain configurations, each module is considered in turn.

For each module, the user configurations take precedence, unless the “force” flag has value “TRUE” into the domain configuration.

In this case, the domain configuration is used unless the “force” flag into the user configuration is also set to “TRUE”, leading to four combinations (considering the force flag):

user FALSE, domain FALSE

    user configuration is used

user TRUE, domain FALSE

    user configuration is used - this combination is often used by web interfaces to forbid access to simple users (meaningless to the server).

user TRUE, domain TRUE

user configuration is used

user FALSE, domain TRUE

domain configuration is used

The above considerations were made upon the assumption that a single module is available both in the domain configuration and the user configuration. If this assumption is not true, the only available module is used.

Note, however, that the absence of a module implies that nothing is said about the module, while its availability either implies that the module is enabled or disabled, depending on the enable flags and depending on how the force flags interacts with user/domain modules.

## 5. Datatree Structure

The whole PigeonDeliver (<http://deliver.pigeonair.net/>) system abstract databases (also known as Backend Databases) as systems storing objects, or group of options, in a hierarchically organized manner, where we have a root with children and where children may have their own children but just one parent as in any tree.

Each object has been given a name and is known to contain a given list of attributes, some of which are mandatory while others may be omitted, and where some may have a single value while others may be multiple valued.

This structure fills well with LDAP Databases (LDAP as a PigeonDeliver (<http://deliver.pigeonair.net/>) Backend Database), but may be realized by using any kind of database (backends will be discussed later on in this document).

The PigeonDeliver (<http://deliver.pigeonair.net/>) system thus expects a so called “datatree” module to provide a layer of abstraction over the database, that makes it look like a tree made of:

- a root, which contains the whole PigeonDeliver (<http://deliver.pigeonair.net/>) tree and the list of domains.
- a list of domains, represented by “mailDomain” objects. By domain we mean a domain name (as those in the DNS) the system will be handling mail for.
- for each domain, a list of objects, each one in its own objectclass (depending on the objectclass), representing the list of modules to be called to deliver mails to the users of this domain.

Each module, however, has at least the attributes defined by the “mailModuleBase” object.

- depending on the kind of object, each object may have its own list of objects below it. At time of writing, the only two modules making use of this feature are the “mailNewsLetter” object and the “mailUsers” object.
- in particular, the “mailUsers” object contains a list of “mailUser” (note the “s”) objects, each one representing a single user.
- below each user, you may find the list of objects and the corresponding configurations of the single user. Like under the domain object, each object below the user object represents a module to be called to deliver the mail to the user and the configuration to be used.
- and so on...

You may now wonder which attributes are mandatory for each objectclass and which are optional.

However, we won't discuss them here. Please refer to the documentation specific to the module using the indicated objectclass or read the LDAP schema (dscm.schema, available in the PigeonDeliver sources) describing all the objects in greater detail and in a formal way.

Right now, you just need to know how the tree is structured and that each object has at least three attributes:

- a “force” flag, as described previously
- an “enable” flag, that tells if the object (or module) is enabled or disabled
- the “name” of the object or of the objectclass

Additionally, each object has its own attributes that are useful for the mail server module that uses the given object. For example, the “mailVacation” module, which reads the “mailVacation” object, may use the “message” attribute, containing the whole message to generate.

Also note that there is a one-to-one mapping between mail server modules name (processes able to add features to the mail server) and the name of the objects stored into the database.

In short, the PigeonDeliver (<http://deliver.pigeonair.net/>) mail delivery system reads (part of) the tree in memory. It then runs a module called as each object found in the “domain directory”. The first module is usually the mailUsers module, which takes care of reading the user specific “slice of the tree” in memory, merging the objects as described previously. The delivery is then continued with the remaining objects, which are passed to the corresponding modules of the mail server.

A new privilege enforcing mechanism is currently being studied. Changing it would require us to rewrite just the mailUsers module. No other changes to the system will be necessary.

## 6. Datatree Implementations using various backends

### 6.1. LDAP Implementation

LDAP is the most straightforward implementation of a datatree module for the PigeonDeliver (<http://deliver.pigeonair.net/>) mailing system.

The datatree database as described in the previous sections is just represented as an LDAP tree, where each object is defined as described in the LDAP schema.

The main configuration parameter for the LDAP Datatree backend is the root of the datatree, or the “dn” of the base object of the PigeonDeliver (<http://deliver.pigeonair.net/>) tree, and the authentication parameters to be used.

The current implementation of the LDAP Datatree backend was tested with the openldap library, and supports:

- referrals, even through different servers
- aliases
- fallbacks using the LDAP API

### 6.2. Preliminary SQL Support

SQL support can be achieved by either using an LDAP Server such as slapd and configuring it to use a SQL backend or by using the PigeonDeliver (<http://deliver.pigeonair.net/>) SQL Datatree support.

In the PigeonDeliver (<http://deliver.pigeonair.net/>) SQL Datatree, each objectclass is represented by a table, where each attribute is represented by a table field. If the field is a string, the attribute is contained directly in the table, while if it is an integer, the integer is used as a lookup key in a special table used to store multi valued elements.

If each objectclass is represented by a specific table, the tree structure itself is stored in its own table, where the key is a sort of cursor (much like the LDAP DN) used to lookup all the modules available under a given level. This table has mainly three fields: one indicating the “cursor” described previously, the second one indicating the objectclass of the contained object, and the third one indicating a unique “id” used to identify the attributes of the specific module in the object table.

Many of what you would think of as strings (as described above) are in reality ids used to avoid having to store the whole string for each and every object.

## **7. Final considerations**

The configuration scheme highlighted in the previous sections can be used by web interfaces or applications accessing the datatree storage (LDAP Directory Tree, SQL database, or so...) to enforce simple privilege schemes and to allow to organize mailboxes in a simple hierarchically organized scheme.

The description of how those privileges are used by PigeonDeliver (<http://deliver.pigeonair.net/>) Administration interfaces and the administrative schemes they create is left to manual of the specific interfaces being used.