Mail system for distributed network

Andrey Zakharchenko

avz@jscc.ru
Joint SuperComputer Center, Russian Academy of Sciences

ABSTRACT

Sometimes it is necessary to organize a mail domain for large and geographically distributed network, which may consist of independent subnets with their own separate Internet connections (and some of those connections may be not very good, or not very fast, or not very cheap). But users of the network need mail system, and it should be fast, convenient, and reliable.

In this paper some ways of distributed mail domain implementation are discussed, and a new one is introduced. The new method allows us to implement distributed mail system – fast and convenient for end users, convenient for administrators, using network traffic sparingly, and reliable enough (at least without single point of failure).

1. Introduction

E-mail is one of the most important communication ways in the Internet. Sometimes an organization work depends on mail system functionality, and somewhere mail transfers may form a large part of total network traffic. A special case we have when organization use one mail domain in a large and geographically distributed network (e.g. central office and regional departments), because it becomes impossible to keep even internal mail messages inside one local network.

The goal of the work was to design a fast, convenient and reliable distributed mail system. The following criteria were formulated to choose a mail system architecture:

- High speed and low latency for end users. Operations of submitting or reading messages should be as fast as possible, even in subnetwork which is connected to the Internet (or to another subnetwork) with slow and narrow channel.
- No single point of failure. If any mail server or network channel goes down (maybe bringing one of subnetworks offline), the rest of the network should be able to send and receive internal and external messages.
- Single point of administration. Any operations with user database, including user movement from one subnetwork to another, should be performed from one central administrative point.
- Economy of traffic, especially on slow channels. Undeliverable letter, virus, or spam should be refused by the first server seeing it without additional network communications. And any letter should travel any particular network channel at most once.

2. Some classic architectures of mail systems

2.1 Single centralized domain

This is the simplest way of implementing mail system. We can use one domain, one mail server, and all users have to connect to this server to submit or read their messages. This configuration is very convenient for administrator and good for local users. But if we have several subnetworks, the situation is not so brilliant. Any internal letter within subnet B will travel to mailbox on the central server in subnet A, and then from mailbox back to subnet B to user's client machine. If the channel A-B is not very fast, the users can get some delays and timeouts. If the letter is addressed to local mailing list, it'll go back and forth many times flooding the internetwork channels. And, last but not least, this architecture contains an evident single point of failure. Bring the only server (or its network connection) down, and the whole network will have no mail at all.

2.2 Set of parallel sub domains

This is the simplest way of implementing mail system without single point of failure. We can install mail servers in every subnetwork and give them unique domain names. This configuration gives very fast and convenient mail connection for all end users (they never use slow long-distance channels directly), and inter-subnet deliveries from one user to another are optimized. But we get some problems when we use local mailing lists. Every list can be maintained on one server only, and letters sent to the list can travel back and forth several times. If the server maintaining any list is offline, this list doesn't work for online users. If a user moves from one subnet to another, his/her address changes (retaining it requires maintaining aliases on old servers, which lead to non-optimal letters delivery). And administrator should maintain a set of user databases, one for every subnet.

2.3 Tree structure

Some of deficiencies of the previous architecture can be avoided implementing a tree structure of main domain and subdomains. Every user gets local address in his/her subnet and an alias in the main domain (on the central server), which is an official address. But this way leads to some deficiencies of centralized domain – single point of failure, duplicated transfers etc. And administrator's work is nearly doubled, because every user has to be inserted into both local and central database.

3. Existing distributed mail domain solutions

Some interesting solutions were found in the Internet. We can discuss two different approaches.

3.1 Postfix per-user transport tables [1]

We can maintain a list of all users on every mail server and keep an optimal path of delivery for each of them. It gives us a mail system without single point of failure, without duplicated mail transfers, fast and convenient for end users, but its administration can be a nightmare! If we try to simplify the administrator's life and introduce default central server (to receive mail for all unknown users of our domain), the architecture becomes ordinary tree structure with all its pros and contras.

3.2 IceWarp Merak Email Server: Distributed Domain [2]

This commercial product allows us to install several mail servers for one domain and distribute the domain users between the servers. If any server receives letter for its local user, it accepts it; if the letter is addressed to another server's user, it tries to find the destination server using VRFY command of SMTP. This architecture doesn't contain single point of failure, but it doesn't contain single point of administration either. VRFY requests can use a lot of network traffic on busy systems even if the results are cached, and delivery to local mailing lists cannot be optimized because VRFY command doesn't return mailing list expansion.

4. Developed architecture

No one of classic (or found) solutions meet all our criteria, that's why a new architecture for distributed mail domain was developed.

First of all, the only way to optimize internetwork traffic (and maintain low latency for end users) is to install a mail server in every subnetwork. Every local server should know all users of the domain, that's why user information is to be stored in the replicated database. Each user has a mailbox in one of the subnetworks, so the user's record in the database contains "subdomain" property, so any user has two addresses: official <username@domain> and local <username@sub.domain>. Mailing lists and aliases are not wired to any special subdomain and belong to the main domain. They are stored in the replicated database too.

Special addresses <all@domain> and <all@sub.domain> are expanded into list of all users of the mail domain or all users of particular subnet.

The mail servers in our system can be divided into three groups ("levels") depending on their configuration, DNS records and user database records.

- Level 0. The server is high-priority MX for our domain, and it doesn't have any local users. Its task is to receive incoming external mail for the whole domain and filter spam, malware, undeliverable letters etc. You can install such a server if you have fast unlimited Internet connection. This level is not mandatory.
- Level 1. The server is low-priority MX for our domain, and it has local users. Usually it will receive letters for its local users only, but if all "level 0" servers go offline, it'll perform their work. If you have several "level 1" servers, their MX priority can be ordered by number of active local users. This is the most common type of server, but you can develop working system without this level using levels 0 and 2 only.
- Level 2. The server is not listed in DNS as MX for our domain and receive letters for local user only. This level is for subnets with slow, irreliable, or expensive Internet connection, or for subnets without real IP (behind NAT). In the latter case our "level 0-1" servers should be able to connect to this one some way, e.g. via VPN or port tunnelling. Sometimes "level 2" servers doesn't send their outgoing mail directly to destination and use "level 0-1" servers as mail relays.

5. Some details of implementation and results

In the current implementation mail servers work under FreeBSD 7 and use Exim as MTA[3]. Information about users is stored in PostgreSQL database[4] replicated by Slony 1[5]. A simple web-interface was developed for those mail administrators who don't like PSQL command line interface. The same architecture, of course, can be installed with other OS, MTA, or database.

It may be interesting to implement replicated user database with subdomains as LDAP tree. Author used PSQL because he already had working Exim configuration using greylisting system implemented as PSQL stored procedure (greylisting tables are not replicated). Moreover, old mail system in the experimental network already had an SQL base of users, and it was more easy to move them to another SQL base then to LDAP directory.

Slony 1 DB replication system was designed to work with LAN-connected database clusters. Our inter-server channels may be very slow when compared to LAN, that's why I've patched Slony to allow them to wait 10 minutes between base synchronizations. Such a configuration leads to about 2Mb of inter-subnet traffic per day for database replication. This overhead may be avoided or substantially reduced using manual DB replication or more intelligent user's and administrator's interface, but the former way is inconvenient, and the latter one is not even planned yet.

The mail system was successfully tested in a relatively small network (three servers with three different levels, three geographically separated subdomains, two separated networks, more then a hundred active users, and up to 30 aliases and local mailing lists). It replaced old centralized mail system (nearly collapsed at the time of replacement). The “letter latency” for end users was lowered from 15—20 seconds (subnet 1) and 30—120 seconds (subnet 2) to 1—2 seconds. Intra-subnet mail never leaves its network of origin, and delivery to local mailing lists is optimized to avoid flooding external channels.

Here is a small table showing work of the servers during 8 days of testing:

level	**letters**			
	received	accepted	delivered	sent
0	281904	2267	0	2317
1	97019	2802	2947	546
2	868	838	803	114

"Received" is total number of incoming connections, "accepted" is number of letters accepted for delivery (without spam, viruses, etc), "delivered" is number of letters delivered to local mailboxes, and "sent" is number of letters sent to other servers via SMTP.

"Delivered+sent" is greater then "accepted", because one accepted letter can be delivered to several mailboxes. "Level 2" server accepted nearly all received letters, because it is not listed in DNS as MX for our domain, and the only type of rejected letters are those with mistyped addresses from local users.

6. Conclusions and future work

The new architecture of distributed mail system seems to be fast, economical, convenient for end users and administrators, and reliable enough (turning off "level 0" server for maintenance wasn't even noticed by end users). The plans of future work contain improving web-interface (and its translation into English if it is desired, at the moment of writing it exists in Russian only) and further reliability improvement by trying to replace "level 1" server with two-machine cluster.

Acknowledgments

I thank Kirill Reutov, who stated the problem, and REA "Prostor", Moscow, Russia, for providing experimental network.

Availability

This work is not resulted in any program product, but the following texts are available from the author:

- Schema of PostgreSQL database storing all information for the distributed mail system, except the messages themselves.
- Two-line patch for Slony1 allowing us using the system with slow Internet connections.
- Examples of configuration files for Exim 4 MTA for using with our database.
- Recommended DNS records for domain using the system.
- Simple web interface for mail system administrators. (In Russian. Is anyone interested in English translation?)
- Simple web interface for end users. (In Russian. What about i18n and l10n?)

References

[1] http://www.irbs.net/internet/postfix/0401/1012.html

[2] http://www.icewarp.com/about_us/technology_news/full_texts/distributeddomain_webversion.php

[3] http://www.exim.org/

[4] http://www.postgresql.org/

[5] http://www.slony.info/