

Semesterarbeit “Replicated Mailserver”

Dieses Dokument gibt den Rahmen der Semesterarbeit von Gabor Cselle im WS 2004/2005 vor. Abweichungen oder Änderungen sind in gegenseitiger Absprache möglich. Die vorgesehene Arbeitszeit von 150 Stunden ist für die folgenden Punkte einzuplanen, genauere Angaben sind der „Semester Project Description“ (siehe unten) zu entnehmen:

- 1) Einarbeitung in den Java-Mailserver „James“.
- 2) Entwicklung einer „Access Box“ für den POP3-Zugriff.
- 3) Entwicklung eines einfachen „Storage Systems“.
- 4) Verschlüsselung der Daten im Storage System.
- 5) Replikation der Access-Boxen.
- 6) Replikation/Verteilung des Storage Systems.

Abschliessend ist ein kurzer (10 Seiten) Bericht über die Arbeit und deren Resultate zu erstellen. Der Bericht soll auch eine kritische Beurteilung der eigenen Arbeit enthalten.

Allgemeines:

- Selbstständiges Arbeiten ist Voraussetzung.
- Nach der Arbeit ist eine Abschluss-Präsentation vorgesehen.
- Es werden (regelmässig) Treffen mit Keno abgehalten.

Kontaktpersonen:

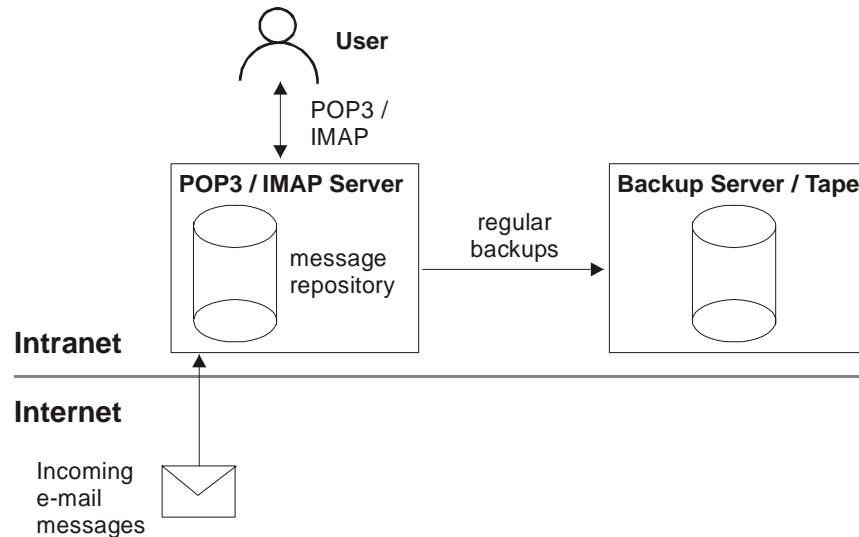
- | | |
|----------------------|----------------------------|
| 1. Keno Albrecht | kenoa@tik.ee.ethz.ch |
| 2. Roger Wattenhofer | wattenhofer@tik.ee.ethz.ch |

Semester Project Description

Gabor Cselle, gabor@student.ethz.ch –November 9, 2004

Motivation

Traditionally, e-mail systems feature a centralized IMAP or POP3 server. Data on this server is backed up regular intervals.



This setup has some drawbacks:

- There is only one mail server, leading to low reliability. Introducing redundancy via replication or fail-over would raise availability, but this is seldom done due to additional work and hardware requirements.
- Backup copies are kept in the same location as the mail server. A catastrophic failure or destruction of the data center would lead to total data loss. Mail servers are seldom duplicated in a different location, because of hardware, connectivity, and setup costs.
- Mail servers typically have significant maintenance costs. Staff needs to be on hand for adding, deleting or modifying accounts, and getting the server back up in case of downtime.

One solution would be outsourcing mail service entirely to a third party. Organizations may shy away from this because of two reasons:

- E-mail is, by nature, confidential. An organization's e-mail repository in the hands of a competitor may be very dangerous.
- When e-mail is stored and served remotely, a loss of external connectivity in the office would keep employees from their e-mail.

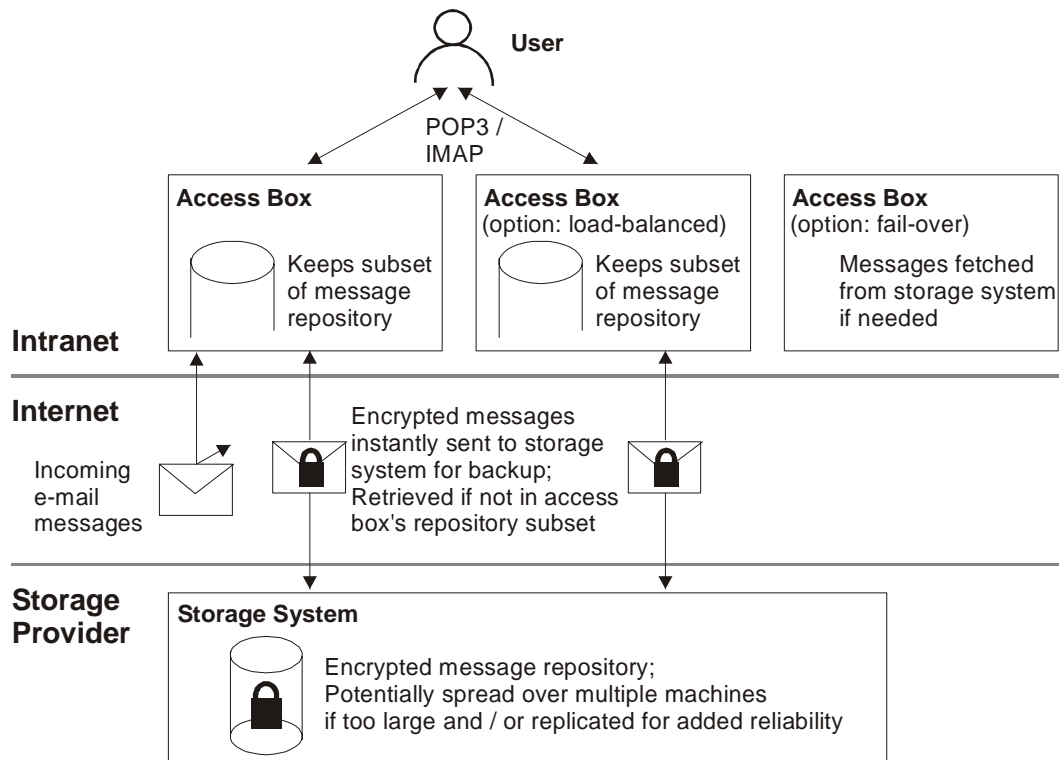
Often, e-mail data is kept on a central server (e.g. with IMAP) and is not permanently downloaded to user's desktops (e.g. with POP3). Proprietary information contained in e-mail stays within the organization – stealing or deleting all e-mail would require a conscious effort by the user. In effect, an e-mail system needs to have a good facility to store large amounts of e-mail data.

Proposed Solution

In my semester thesis, I will implement and analyze an approach to e-mail system architectures that will attempt to solve the problems described above.

This architecture features access boxes that act as IMAP or POP3 servers. They copy incoming messages to a remote storage system in encrypted form.

Additional access boxes can be deployed to improve availability: Since data is backed up independently, all boxes can retrieve stored e-mail. If one fails, users are redirected to a different box.



Detailed Design

In my project, I will implement a simple proof-of-concept for the architecture above.

Implementation Language: Java.

E-Mail Protocol: POP3 only. POP3 is a much simpler protocol than IMAP and allows me to build on top of the existing Apache James [1] server. Drawbacks are the lack of support for folders and the fact that e-mail reader software will need to be set to "Don't delete mail on server."

Data Backup Encryption: Backup data is encrypted via a symmetric algorithm (e.g. AES). All access boxes know a common secret key, which is never known to the storage provider. This leads to a question: how best to guarantee that the secret key never leaves the access box. Will it be enough to limit the lines of code that touch the key to a minimum and make it easily understandable?

Redirection: When access boxes fail, users need to be redirected to those still alive. This could be done through modifying DNS entries in the still-alive servers.

Replication: There may be multiple storage servers that replicate data among each other, for added reliability. Coming up with sophisticated replication strategies is, however, beyond the scope of my project.

Testing: Each component of the system must have a "play dead" mode, so failure scenarios can be tested without actually turning off the hardware or pulling the network cord.

Possible Enhancements

Here are some more ideas that will not be part of my project:

Full Text Search: Gmail has shown that full-text search in e-mail is an extremely useful feature. Can we store a full text index in the storage system without being able to reconstruct e-mails in the encrypted repository?

Synchronization with LDAP User Directory: Organizations typically have an LDAP directory with a list of all users. New users are added and removed there. Synchronizing this directory with our system's user database would ease administration.

More Protocols: Aside from POP3, an obvious extension would be IMAP; however, it seems too hard to implement. Another useful feature would be a web interface for users without access to e-mail reader software.

Storage: Webmail applications are often misused as storage space. What modifications would have to be made to offer file storage through the access boxes?

Other Considerations

Access boxes with remote storage could also be a viable business. Typically, companies offering e-mail such as Google (GMail), Yahoo (Yahoo Mail), or Microsoft (Hotmail) have built up large amounts of storage space for user data, along with distributed storage systems [2, 3]. They could offer e-mail service to companies by selling access boxes with a storage service, and promising to:

- improve availability through replication,
- make total loss events less likely, and
- keeping companies' information private, while
- reducing maintenance costs

Also, such an offering could come with a advanced web interface, such as GMail's, which users may prefer over their traditional e-mail client.

References

- [1] Java Apache Mail Enterprise Server ("Apache James"):
<http://james.apache.org/>
- [2] S. Ghemawat, H. Gobioff, S. Leung: The Google File System
<http://labs.google.com/papers/gfs.html>
- [3] Network Appliance, Inc.: Filer Server Product Documentation
<http://www.netapp.com/products/filer/>