

10.000 Clients parallel? Strategien im Wachstum

von Jochen Topf

jochen.topf@jtlic.de

1 Einleitung

Mit dem Wachstum eines Unternehmens oder Internet-Providers und der zunehmenden Nutzung des Mediums E-Mail wachsen auch die Anforderungen an die E-Mail-Infrastruktur. Dazu kommen die Notwendigkeit der Viren- und Spamfilterung, der Aufwand durch die kryptographische Absicherung der E-Mail-Kommunikation und die gestiegenen Anforderungen an die Archivierung der E-Mails als Grund für einen immer weitergehenden Ausbau der E-Mail-Systeme.

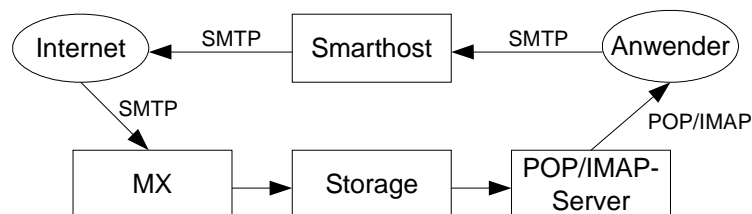
2 Aufgabenverteilung

Wenn der einzelne Rechner, der für die Bearbeitung der gesamten Unternehmens-Mail zuständig war, nicht mehr ausreicht, und auch durch den Hardware-Ausbau dieses Rechners kein Fortschritt mehr zu erzielen ist, muss die Arbeit aufgeteilt werden. Glücklicherweise eignen sich Mailsysteme sehr gut für die Verteilung auf mehrere Rechner.

In einem ersten Schritt können die Mailserver für eingehende E-Mail (MX) und ausgehende E-Mail (Smarthost) getrennt werden. E-Mail, die aus dem Internet eintrifft, wird über einen MX-Record im DNS auf den einen Server geleitet, E-Mail aus dem Unternehmen oder von den Kunden eines Providers per Konfiguration im Client auf den anderen. Mit sehr wenig Aufwand ergibt sich dadurch schon eine Lastverteilung. Zusätzlich ist der ausgehende Mailverkehr vom eingehenden etwas abgeschirmt: Bei massiven Wurm- oder Spamangriffen, die den Server für eingehende E-Mail lahmlegen, ist wenigstens noch ausgehende E-Mail möglich.

Ein weiterer Kandidat für die Trennung ist das Storage-System, das die eigentlichen Mailboxen beherbergt. Es kann ebenfalls auf einen weiteren Rechner ausgelagert werden, wie auch der POP- bzw. IMAP-Server für den Zugriff auf die Mailboxen.

Insgesamt ergibt sich also die folgende Architektur:¹

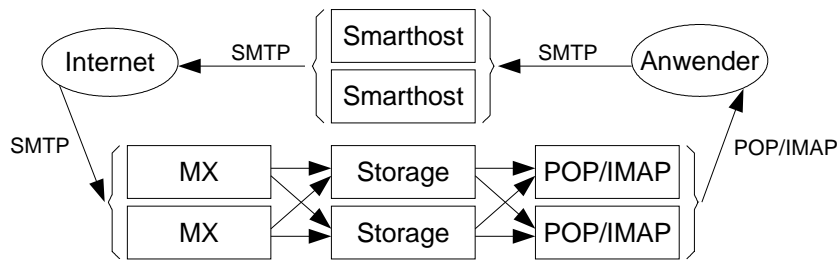


Die Grafik stellt den Fall dar, wie er in einem Unternehmensnetz vorkommen könnte. Beim Mailsystem eines Internet-Providers gibt es meist keine Unterscheidung zwischen dem Internet und dem Netz mit den Anwendern, sie fallen dann also zusammen.

Jedes Element dieses Systems kann zur Lastverteilung und zur Absicherung gegen Ausfälle

¹ Eine ausführlichere Beschreibung dieser Architektur und ihrer Anwendung beim Provider Schlund+Partner findet sich in [Topf]. Ähnliche Ansätze sind in [Horman] und [Christenson] beschrieben.

redundant ausgelegt werden²:



Bei dieser Architektur stellt sich natürlich die Frage, woher die verschiedenen Elemente wissen, mit welchen anderen Elementen sie reden müssen. Einfach ist das für die Elemente, die SMTP sprechen. Benutzer haben einen Smarthost in ihren Clients konfiguriert, der mehrere IP-Adressen hat, die die Verteilung ermöglichen. Ähnlich die MX-Rechner: Entweder gibt es verschiedene MX-Einträge oder einen Eintrag mit mehreren IP-Adressen. Damit läßt sich eine (primitive) Lastverteilung, aber nur eine begrenzte Ausfallsicherheit erreichen. Verbessern läßt sich das noch durch vorgeschaltete Load-Balancer oder einen Failover mit Weitergabe der IP-Adressen zwischen den Rechnern.

Für die Weitergabe der Mails von den MX-Rechnern zu den Storage-Rechnern bzw. die Abfrage der E-Mails von den POP/IMAP-Rechnern von den Storage-Rechnern muss eine zweifache Zuordnung konfiguriert werden. Einerseits muss der MX-Rechner wissen, in welche Mailbox auf welchem Storage-Rechner eine Mail zugestellt werden soll. Andererseits muss der POP/IMAP-Rechner alle Accounts und ihre Paßwörter kennen und wissen, auf welchem Storage-Rechner die E-Mails für diesen Account liegen.

Für den Zugriff von den POP/IMAP-Servern auf die Storage-Rechner gibt es zwei Verfahren. Im einen Fall läuft die POP/IMAP-Server-Software komplett auf den vorgelagerten Servern und der Zugriff auf das Storage-System erfolgt per NFS oder über ein Datenbankprotokoll. Im anderen Fall läuft auf dem POP/IMAP-Server nur ein Proxy, der den Anwender authentifiziert und dann die Verbindung auf den passenden Storage-Server durchschaltet. Dieser übernimmt dann die weitere Interpretation des Protokollablaufes und stellt die Daten zur Verfügung.

3 Spam-undVirenschutz

Kein Mailsystem kann heute ohne Spam- und Virenschutz auskommen. Beides ist sehr rechenintensiv, weil die E-Mails inklusive aller Anhänge ausgepackt und untersucht werden müssen. Viele Hersteller bieten All-In-One-Appliances an, die den MX-Rechnern vor- oder nachgeschaltet werden und diese Filterung übernehmen. Auch hierdurch wird eine weitere Lastverteilung erreicht. Alternativ wird der Spam- und Virenschutz auf den MX-Rechnern integriert. Durch die oben beschriebene Architektur können so viele Rechner parallel betrieben werden, wie es der erhöhte Aufwand für den Spam- und Virenschutz erfordert.

Auch die Filterung ausgehender E-Mails auf den Smarthosts sollte in der heutigen Zeit nicht fehlen. Damit wird verhindert, dass infizierte Mitarbeiter- oder Kundenrechner zu Spamschleudern werden oder Viren und Würmer weiterverteilen.

² In der Grafik sind alle Elemente jeweils zweifach ausgelegt, es spricht aber nichts dagegen, eine weitere – und je nach Notwendigkeit unterschiedliche – Aufteilung vorzunehmen.

4 Storage

Alle Mails müssen gespeichert werden. Eingehende E-Mails müssen auf dem Server mindestens so lange bereitgehalten werden, bis sie gelesen wurde. Heute wird aber immer öfter auch erwartet, dass bereits gelesene E-Mail auch weiter archiviert wird. Und auch ausgehende E-Mail sollte irgendwo gespeichert werden.

Bei der Speicherung ist sowohl der physikalische Ort der Speicherung als auch das logische Format zu berücksichtigen. Beides kann einen erheblichen Einfluß auf die Leistung und Lastverteilung haben.

4.1 Ort der Speicherung

Da eine zentrale Speicherung auf einem Rechner früher oder später nicht mehr ausreicht, muss über einen geeigneten Verteilmechanismus nachgedacht werden. Hier ergeben sich mehrere grundverschiedene Modelle:

Soll die E-Mail klassisch im Filesystem gespeichert werden, muss ein ausreichend großes Filesystem für alle E-Mails zur Verfügung stehen, etwa auf einem NFS-Server oder einem SAN. Dafür gibt es spezielle Hard- und Software, die auch mit sehr großen Datenmengen umgehen kann. Alternativ können die Mailboxen auf mehrere Rechner aufgeteilt werden. Für jede Mailbox wird ein Rechner bestimmt, auf dem sie abgelegt wird. Das kann entweder statisch erfolgen, beispielsweise anhand einer Mailboxnummer (gerade Nummern auf diesem Server, ungerade auf jenem) oder dynamisch mit Hilfe einer zusätzlichen Datenbank, die die Zuordnung verzeichnet. Durch die zusätzliche Datenbank entsteht zwar ein höherer Aufwand, dafür ist man bei der Lastverteilung flexibler.

4.2 Format

Neben dem klassischen Mailbox-Format, bei dem alle E-Mails einer Mailbox in einer Datei gespeichert werden, haben sich in den letzten Jahren viele Alternativen herausgebildet.

Das alte Mailbox-Standardformat³ hat als wesentlichen Nachteil, dass der gleichzeitige Zugriff durch mehrere Programme nicht möglich ist. Stattdessen müssen die Programme sich durch Locking-Mechanismen gegenseitig „absprechen“. Um diesen Problemen zu entgehen, vor allem in Zusammenhang mit NFS, das Locking direkt nicht unterstützt, wurde das Maildir-Format⁴ entwickelt, bei dem jede E-Mail in einer eigenen Datei gespeichert wird.

Für POP-Server reichen solche einfachen Formate noch aus; IMAP-Server, die Unterorder und die Suche auf dem Server unterstützen, brauchen aber in der Regel komplexere und damit aufwändigere Formate. Standards haben sich hier noch keine gefunden, jede Software benutzt ihr eigenes Format.

Manche Software nutzt auch relationale Datenbanken für die Speicherung der E-Mails oder zumindest der Metadaten. Erfolgt der Zugriff auf die Mailbox nur per POP, wenn also nur die Operationen „Neue E-Mail anlegen“, „E-Mails auflisten“, „E-Mail lesen“ und „E-Mail löschen“ zur Verfügung stehen, dann ist eine relationale Datenbank sicherlich Overkill. All die Funktionen zur Transaktionssicherheit bei Änderungen kleiner Datensätze in großen Datenbanken bleiben ungenutzt und kosten nur unnötig Rechenzeit. Für die komplexeren Mailsysteme auf Basis von IMAP kann eine Speicherung in einer relationalen Datenbank aber durchaus sinnvoll sein. Trotzdem wird natürlich

³ http://wiki.allthingsemail.org/w/Mbox_format

⁴ <http://cr.yip.to/proto/maildir.html>

eine relationale Datenbank einer speziell für den Anwendungsfall E-Mail erstellten Datenbank – auf gleicher Hardware – immer unterlegen sein.

Relationale Datenbanken haben gegenüber einfachen filesystembasierten Verfahren allerdings den Vorteil, dass sie vielfach eine Replizierung unterstützen, d. h. die E-Mails liegen nicht nur auf einem Server, sondern auf mehreren. Der Ausfall eines Servers führt also nicht zum Ausfall der Mailboxen. Werden die E-Mails im Filesystem abgelegt, fallen mit einem Server alle Mailboxen aus, die auf diesem Server abgelegt wurden.

Um diese Problematik zu umgehen, gibt es Systeme, die jede E-Mail automatisch n-fach ablegen [Saito]. Eine bessere Lastverteilung und Fehlertoleranz wird hier mit einer höheren Komplexität des Systems erkaufte.

4.3 Backup

Je nach dem genutzten Speicherformat muss auch das Backup der E-Mails anders geregelt werden. Im Idealfall ist ein gesondertes Backup nicht notwendig, nämlich dann, wenn das Mailsystem z. B. durch die Nutzung replizierter Datenbanken schon genug Redundanz bietet. Ist das nicht der Fall, kann natürlich wie üblich ein regelmäßig Backup, z. B. jede Nacht, gefahren werden. Allerdings werden viele E-Mails dann niemals gesichert, weil sie das Mailsystem im Laufe des Tages erreichen und am gleichen Tag noch bearbeitet und gelöscht werden. Will man das vermeiden, müssen die E-Mails länger aufbewahrt werden. Ein häufigeres Backup während des Tages wird in den meisten Fällen nicht möglich sein, da die zusätzliche I/O-Last die meisten Systeme in die Knie zwingen wird.

Alternativ kann man auch bei der Einlieferung eine Kopie jeder E-Mail „abzweigen“ und gesondert speichern. Die Kopie ist damit sicher vor versehentlichem Löschen oder einem Ausfall im Hauptmailsystem. Allerdings wird die Wiederbeschaffung einer verlorenen E-Mail auch entsprechend aufwändiger sein.

5 POP vs. IMAP

Das Post Office Protocol (POP) ist schon relativ alt und sehr einfach. Der Client authentifiziert sich gegenüber dem Server und kann dann den Inhalt einer Mailbox auflisten, einzelne E-Mails abfragen und löschen. Viel mehr bietet das Protokoll nicht, wird aber auch in vielen Fällen nicht benötigt. Gedacht ist POP für den Fall, dass ein Anwender seine E-Mails regelmäßig überprüft und komplett herunterlädt und dann lokal verwaltet.

IMAP ist eine spätere Entwicklung, die deutlich mehr Funktionen bietet. Der Fokus ist hier ein etwas anderer: Der Anwender lädt seine E-Mail nicht mehr komplett herunter, sondern er belässt sie auf dem Server. Die Verwaltung seiner E-Mails, das Löschen, Verschieben in Unterorder, Suchen in den E-Mails usw. wird direkt auf dem Server erledigt. IMAP ist daher erheblich komplexer als POP.

Für den Betreiber eines großen Mailsystems ergeben sich aus der Wahl zwischen POP und IMAP vor allem drei Konsequenzen: Erstens sind POP-Mailboxen typischerweise klein, weil der Client die E-Mail immer wieder abfragt und löscht, IMAP-Mailboxen dagegen sind typischerweise groß, weil sie die gesamte E-Mail-Kommunikation des Anwenders speichern. Dazu kommt zweitens, dass IMAP-Verbindungen länger – zum Teil über Stunden oder Tage – offen gehalten werden. POP-Verbindungen müssen immer wieder neu aufgebaut werden, weil man aufgrund der Einschränkungen im Protokoll bei bestehender POP-Verbindung – im Gegensatz zu IMAP – neue E-Mails nicht

sehen kann. Und drittens führen die Möglichkeiten der E-Mail-Verwaltung auf dem Server zu höherer Last (z. B. durch die Suchfunktion) und zu komplexeren Datenstrukturen (Verwaltung von E-Mails in mehreren Foldern), die die Implementierung aufwändiger machen. Insgesamt ergibt sich durch die Nutzung von IMAP im Vergleich zu POP serverseitig ein wesentlich höherer Aufwand.

6 Namensvergabe

Als nützliche Vorbereitung auf ein späteres Wachstum sollte man schon von Anfang an für verschiedene Dienste verschiedene Rechnernamen vorsehen. Beispielsweise kann der POP-Server unter `pop.example.com`, der IMAP-Server unter `imap.example.com` und der Mailserver für ausgehende E-Mail (Smarthost) unter `smtp.example.com` erreichbar sein. In der Anfangszeit zeigen alle diese Namen auf den selben Rechner, bei einer späteren Änderung muss nur noch der Eintrag im DNS geändert, die Client-Konfigurationen aber nicht mehr angepaßt werden.

7 Verschlüsselung

Alle Mailprotokolle bieten die Möglichkeit, die Verbindungen mit SSL/TLS zu verschlüsseln. Dies ist besonders wichtig bei POP und IMAP sowie bei der Nutzung von SMTP AUTH, weil sonst Passwörter unverschlüsselt über die Leitung gehen. Die Nutzung aufwändiger Krypto-Algorithmen führt aber auch zu einem erheblichen Rechenaufwand, der die Menge an gleichzeitig möglichen Verbindungen erheblich reduziert. Natürlich kann durch den Einsatz moderner CPUs hier eine Menge getan werden, aber auch dann stößt man unweigerlich an Grenzen. Auf dem Markt werden deshalb besondere Krypto-Chips auf PCI-Karten angeboten, sogenannte Beschleuniger-Karten (acceleration cards).⁵ Alternativ kann auch wieder der oben beschriebene Proxy-Ansatz gewählt werden, um die Last auf viele Rechner zu verteilen.

8 DNS

Für jede E-Mail, die versandt oder empfangen wird, werden viele DNS-Abfragen nötig. Neben der Abfrage des MX-Rechners und dessen IP-Adresse können das z. B. die Abfrage von Reverse-Records oder von DNSBLs zur Spamabwehr sein. Unter Umständen lohnt es sich, einen oder mehrere spezielle DNS-Server für das Mailsystem bereitzustellen, die mit einem großen Cache ausgerüstet sind.

9 Verwaltung und Konfiguration

Jedes größere Mailsystem hat neben den eigentlichen E-Mails auch mit einer Unmenge an Verwaltungsdaten umzugehen: Domains, Mailadressen, Passwörter für Mailboxen und SMTP AUTH, Weiterleitungen, Vacation-Meldungen, Fax- und SMS-Nummern für Unified Messaging, Spamfilter-Konfigurationen usw. Auch diese Daten müssen verwaltet werden und schnell zugreifbar sein. Wenn tausende Clients eine POP-Mailbox im Minutentakt abfragen, dann muss auch die Datenbank mit den Passwörtern mit der Anfragemenge zurecht kommen.

Heute gibt es dafür zwei wesentliche Optionen: Relationale Datenbanken, die über SQL abgefragt werden, oder LDAP-Datenbanken. Die Entscheidung für das eine oder andere Modell wird dabei in vielen Fällen von der bestehenden Infrastruktur in einem Unternehmen abhängen. Hat man bereits eine LDAP-Infrastruktur für andere Anwendungen (z. B. für einen unternehmensweiten Login), so bietet sich die Nutzung dieser Infrastruktur an. Es darf aber nicht unterschätzt werden, welche zu-

⁵ Für weitere Details siehe [Rescorla], Chapter 6 „SSL Performance“

sätzliche Last ein großes Mailsystem bringt. Die Auslieferung einer einzigen E-Mail kann ein dutzend oder mehr Anfragen an die Datenbank nach sich ziehen!

Wegen der Bedeutung des Mediums E-Mail ist auch unbedingt auf eine redundante Auslegung der Datenbank zu achten. Ein Ausfall von wenigen Minuten kann schon zu Beschwerden von Benutzern und zu erheblichen Mailstaus führen.

10 Literatur

- [Christenson] Nick Christenson, Tim Bosserman, David Beckemeyer: A Highly Scalable Electronic Mail Service Using Open Systems. USENIX Symposium on Internet Technologies and Systems, 1997.
<http://www.usenix.org/publications/library/proceedings/usits97/christenson.html>
- [Golanski] Yann Golanski: The Exim Mail Transfer Agent in a Large Scale Deployment. 2000
<http://www.nndg.york.ac.uk/staff/yann/lsm.ps>
- [Horman] Simon Horman: High Capacity Email. http://www.vergenet.net/linux/mail_farm/
- [Knowles] Brad Knowles, Nick Christenson: Design and Implementation of Highly Scalable E-mail Systems. LISA Conference, 2000.
<http://www.shub-internet.org/brad/papers/dihses/>
- [Levitt] Lee Levitt, Donald Livengood, Andrew MacFarlane: IMAP Servers. What differentiates standards-based messaging systems? Proceedings JENC8, 1997.
<http://www.terena.nl/conferences/archive/jenc8/papers/722.ps>
- [Rescorla] Eric Rescorla: SSL and TLS. Designing and Building Secure Systems. Addison Wesley, 2001. ISBN 0-201-61598-3.
- [Saito] Yasushi Saito, Brian N. Bershad, Henry M. Levy: Manageability, availability and performance in Porcupine: a highly scalable, cluster-based mail service. 17th ACM Symposium on Operating System Principles (SOSP 99) Published as Operating Systems Review 34(5):1 15, Dec. 1999. <http://citeseer.ist.psu.edu/306583.html>
- [Topf] Jochen Topf: Briefsortierer. Hunderttausende E-Mails pro Tag unter Linux. iX 11/99, Seite 128.