

Strategien im Wachstum

von Jochen Topf

2. Mailserver-Konferenz
Magdeburg, 19. und 20. Mai 2005

Inhalt

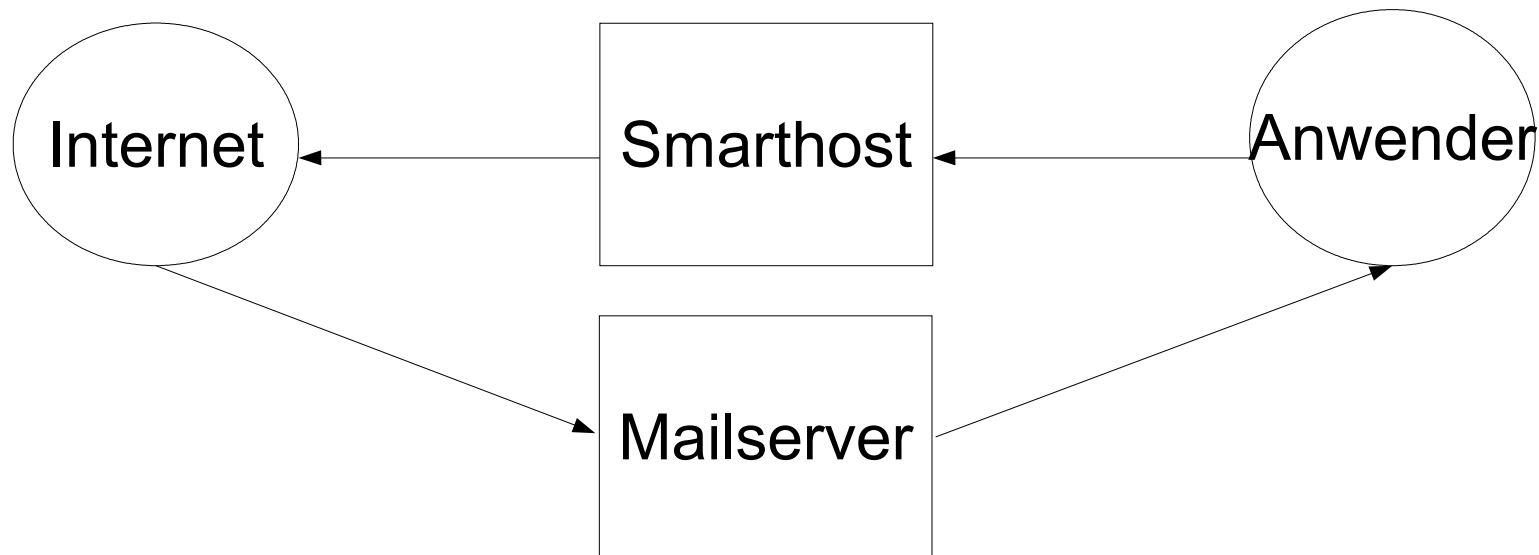
- Architektur eines Mailsystems
- Storage
- Konfiguration
- Tuning
- Zu erwartende Probleme
- Sonstiges

Architektur eines Mailsystems

Einfacher Anfang



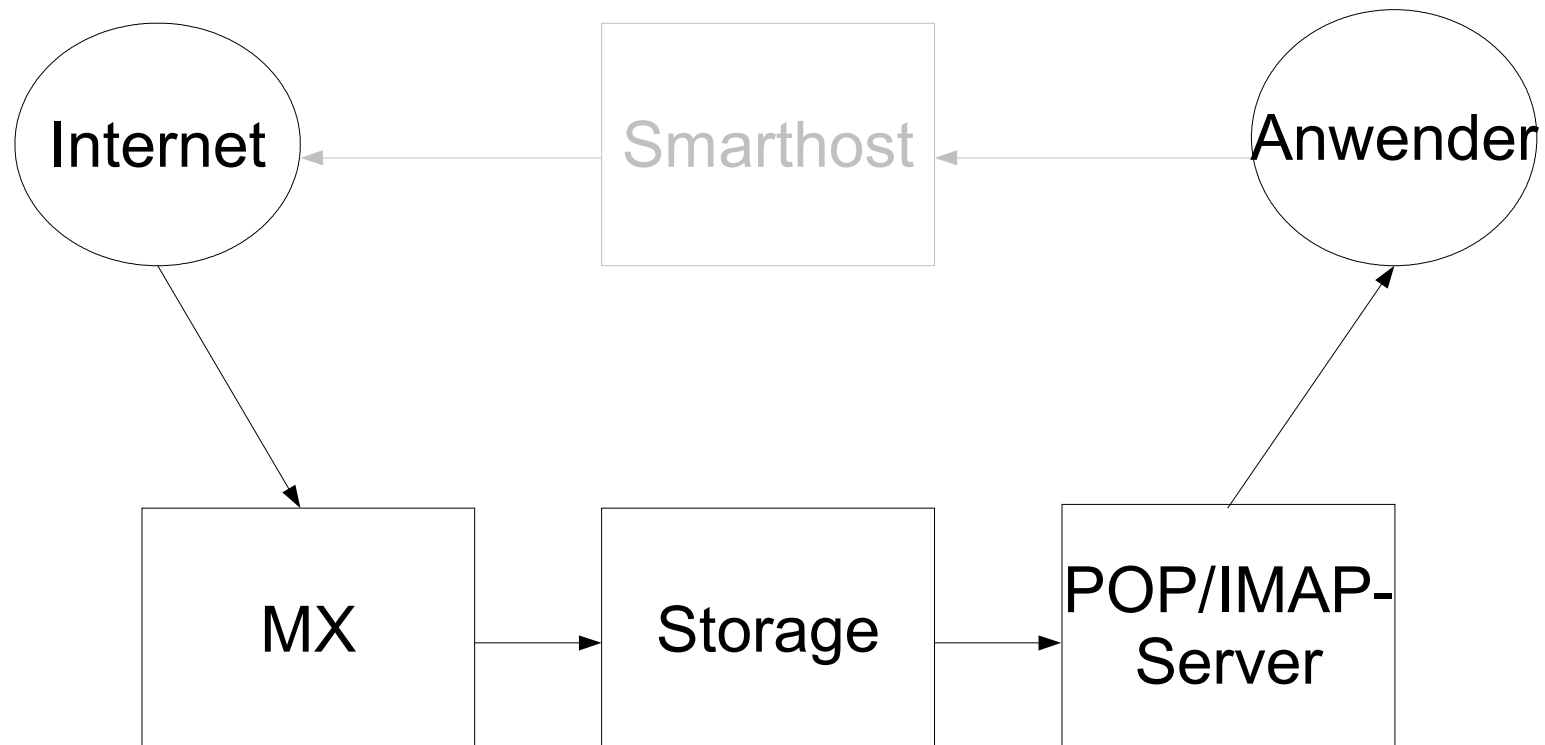
Ein-/Ausgang trennen



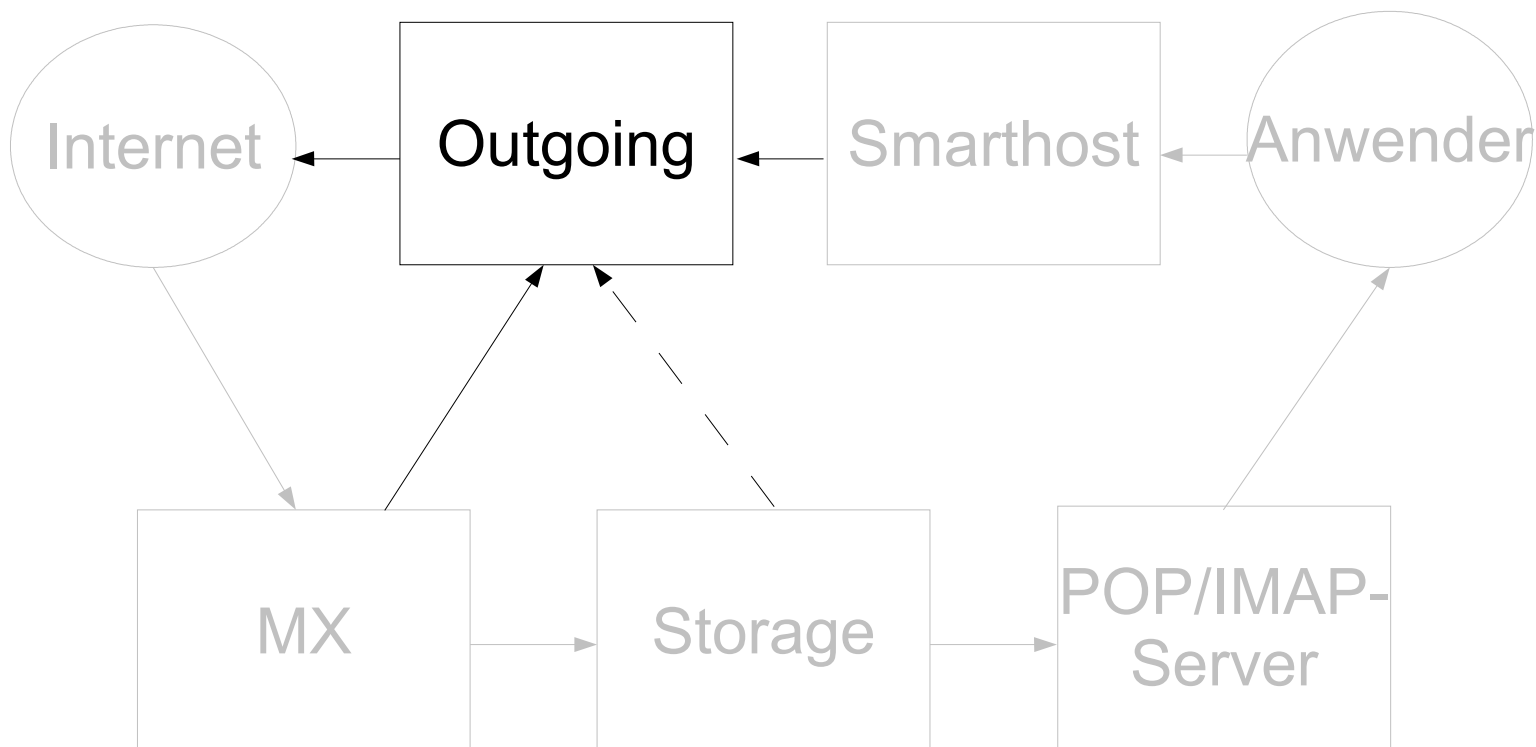
Vorteile der Trennung In/Out

- Reduktion der Last
- Probleme auf einer Seite schlagen nicht durch (z.B. DoS-Attacken)
- Einfachere Konfiguration

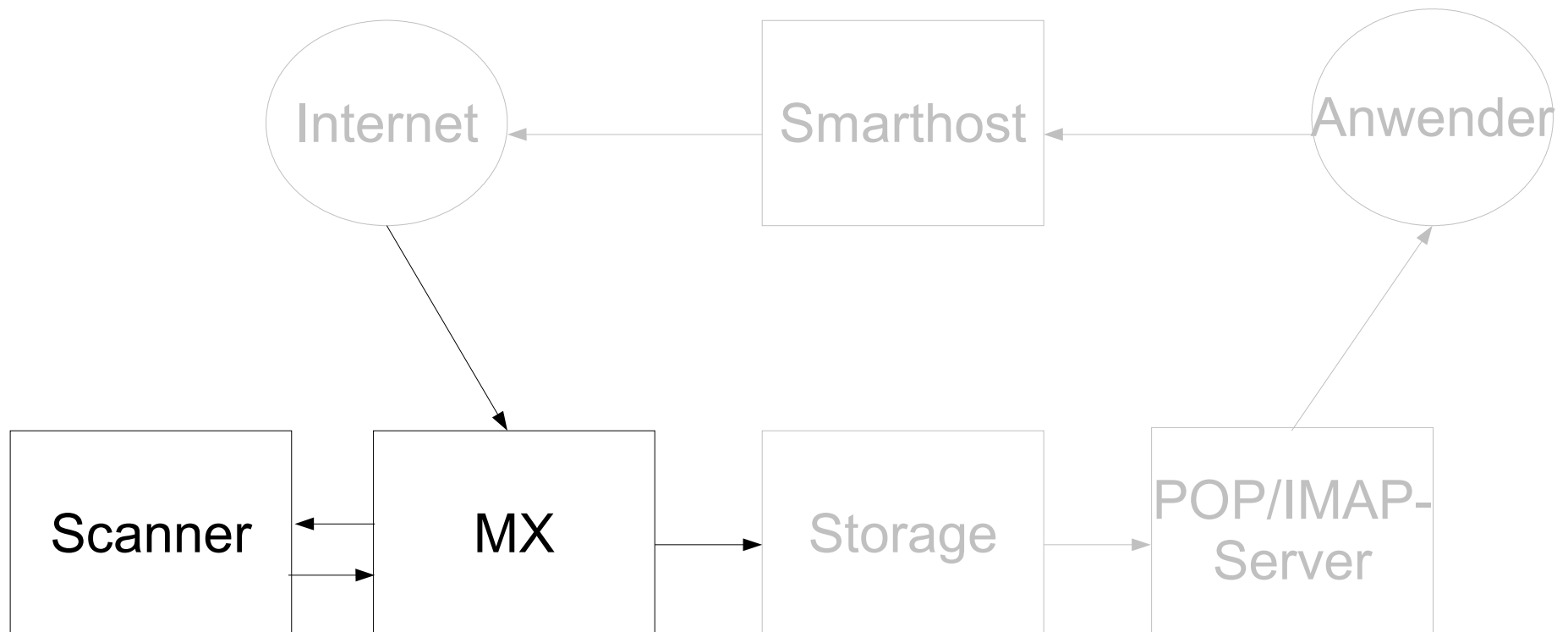
Storage herauslösen



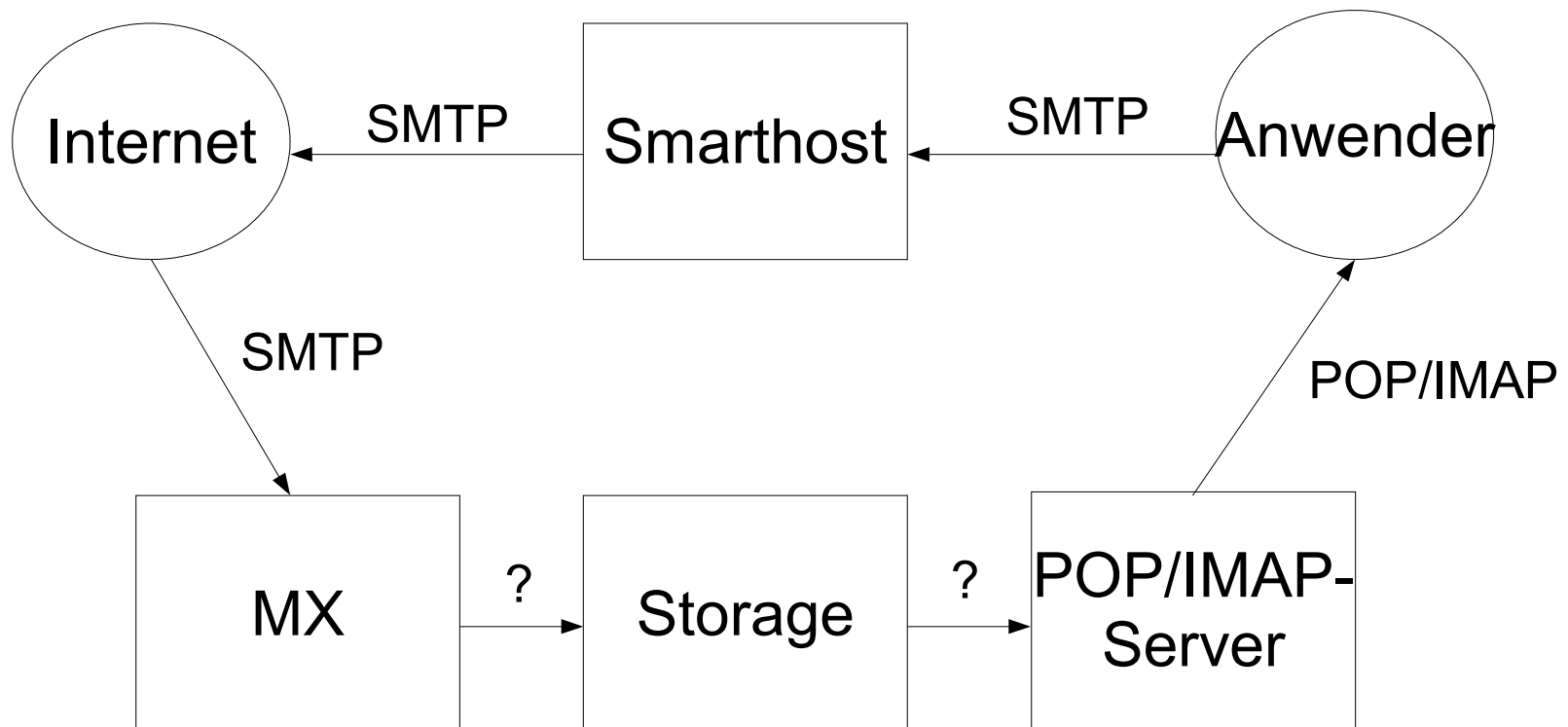
Noch mehr Details



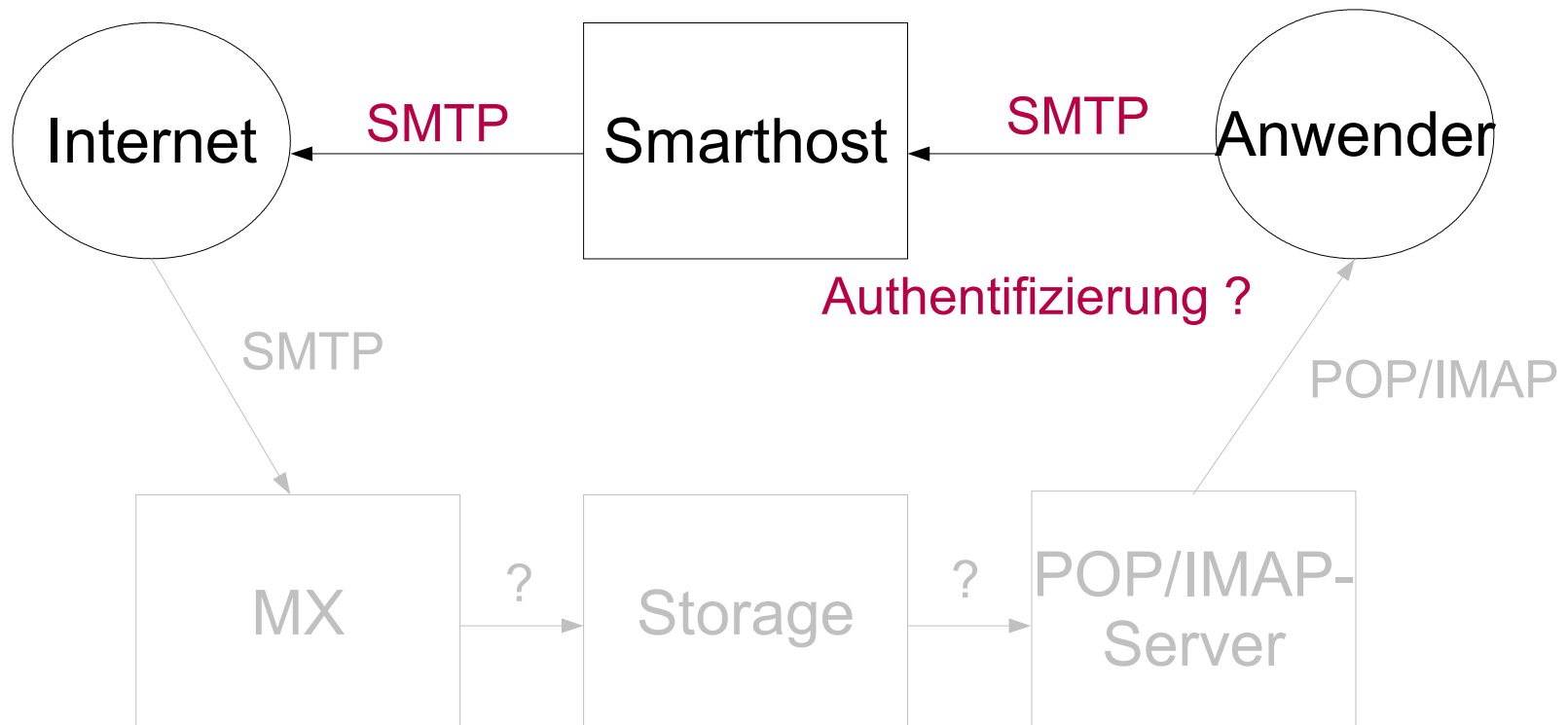
Spam- und Virens Scanner



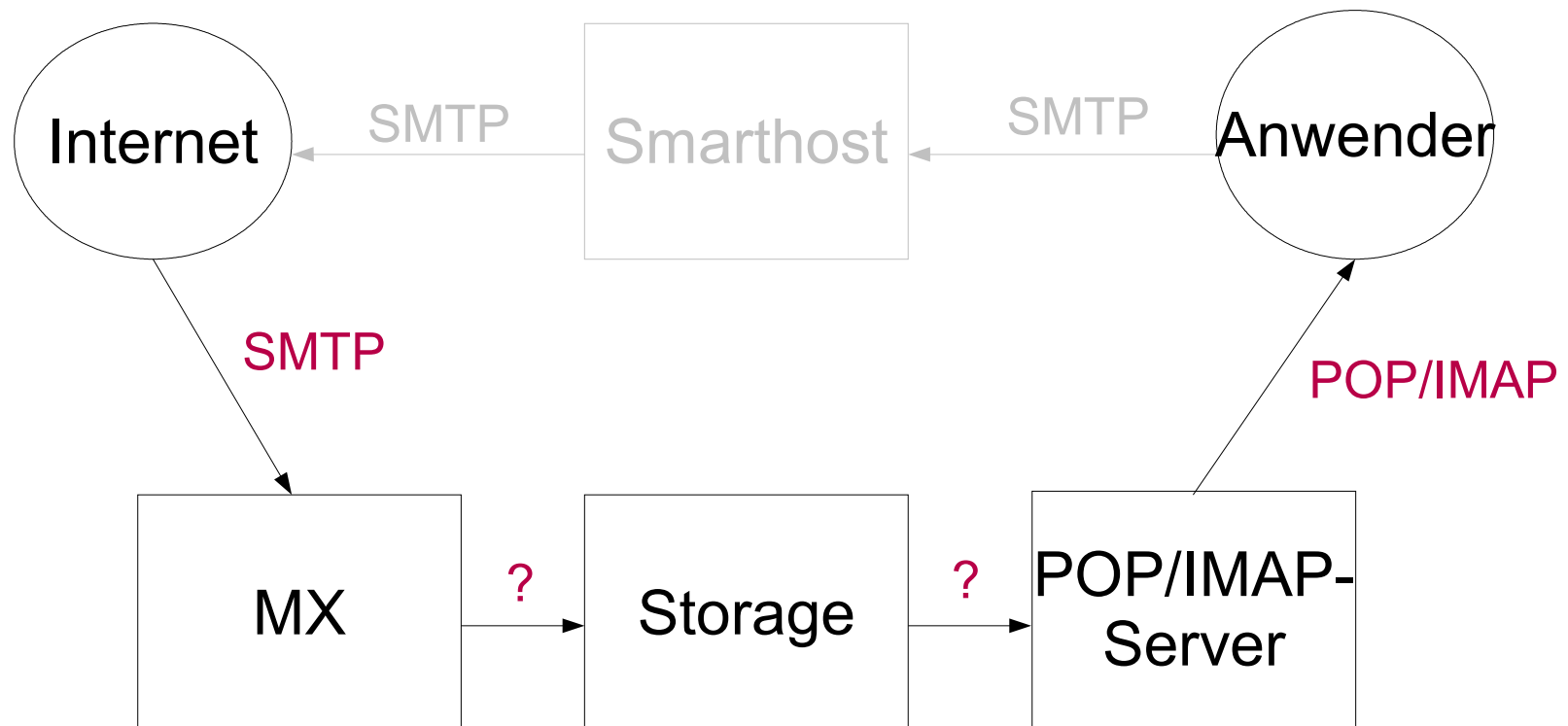
Protokolle



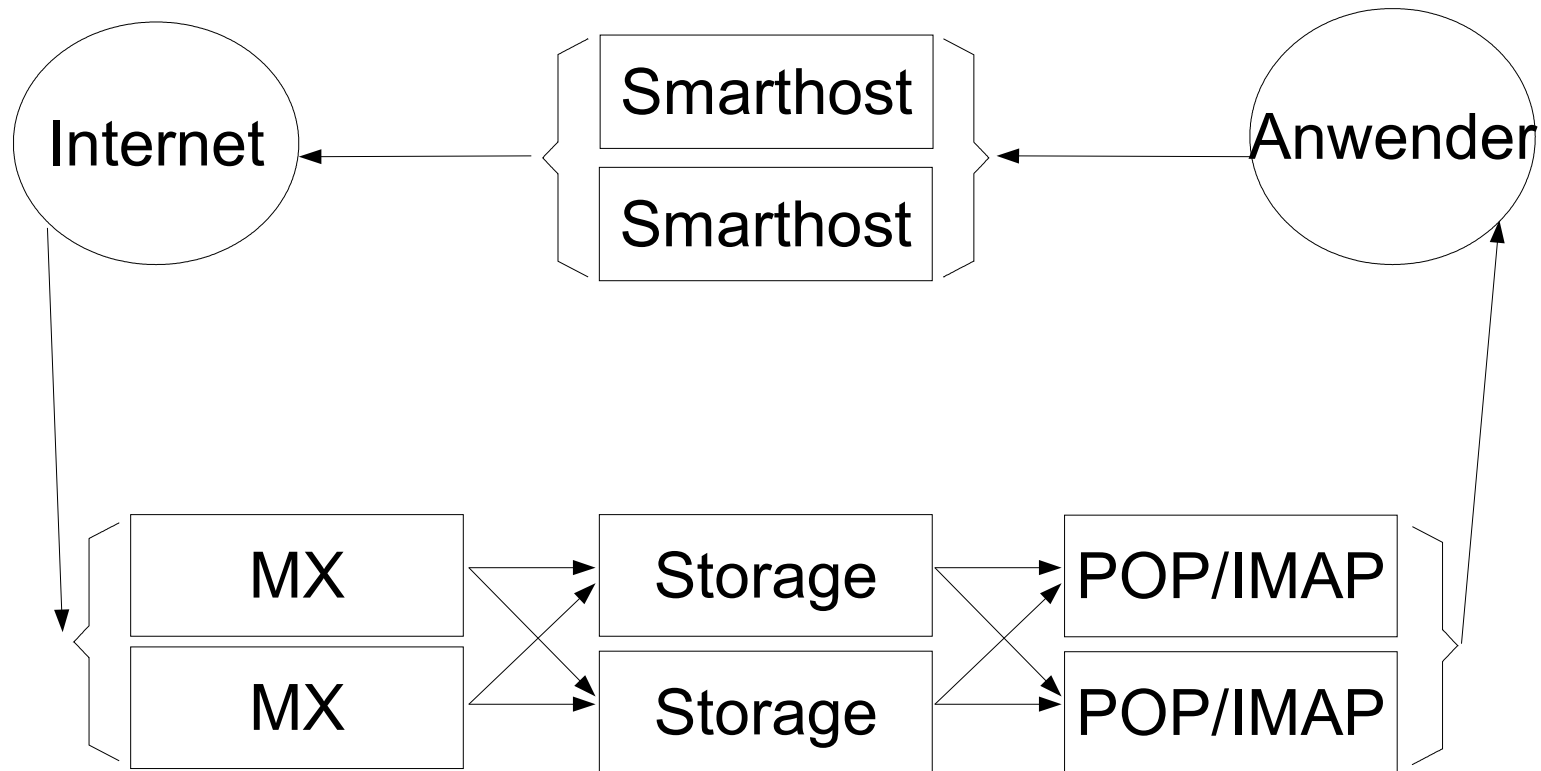
Protokolle



Protokolle



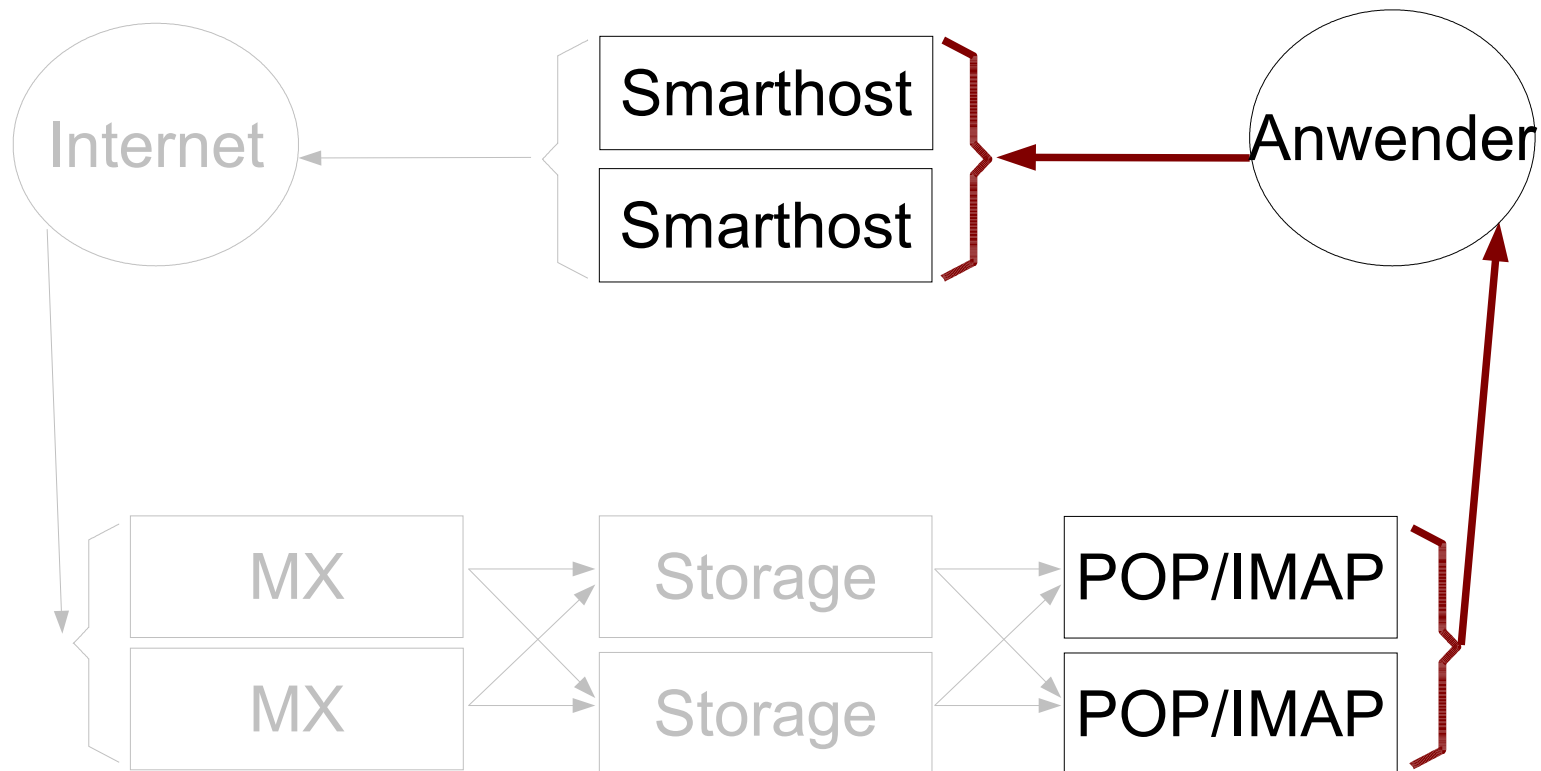
Doppelt hält besser



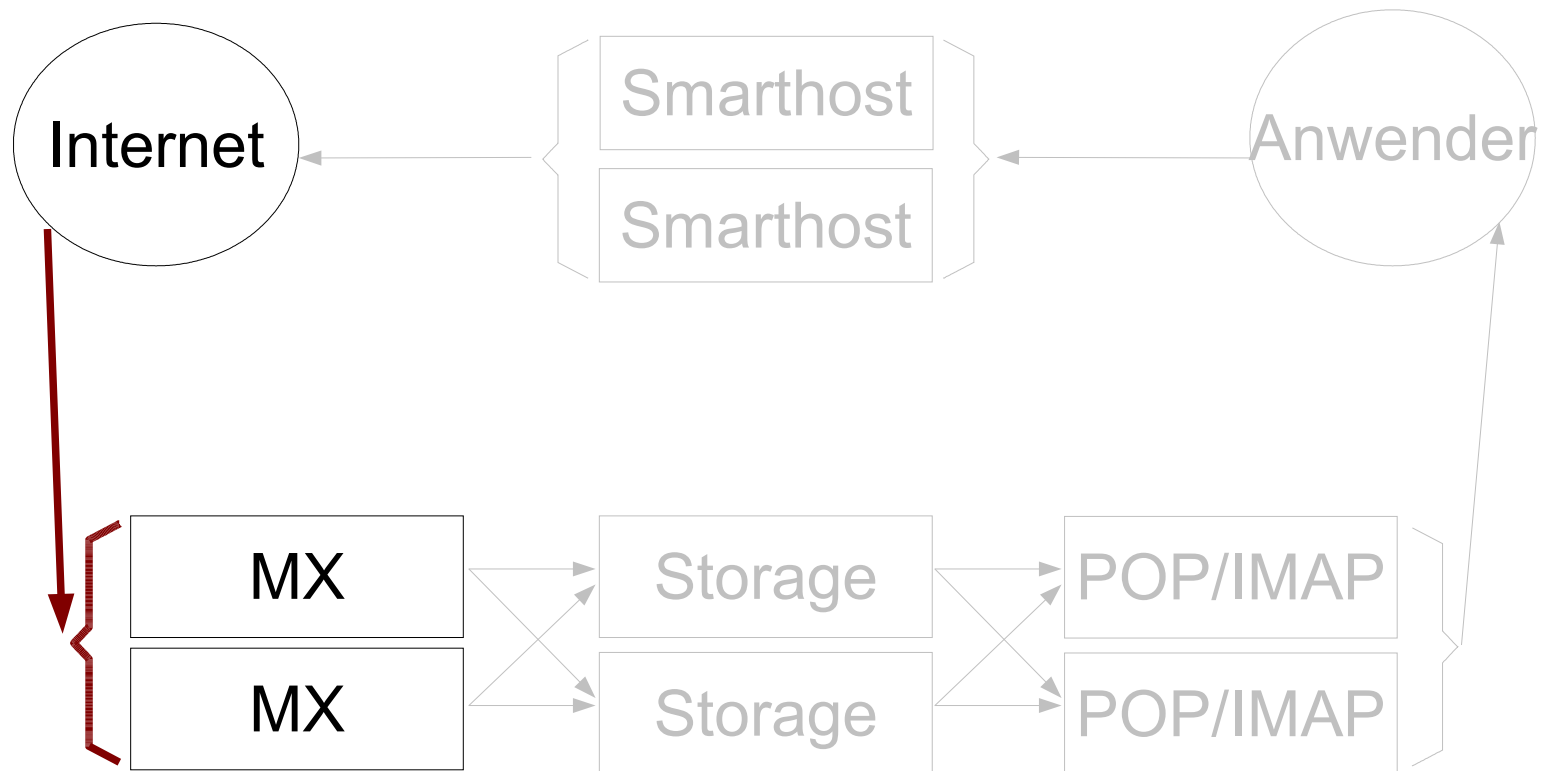
Doppelt hält besser

- Zur Lastverteilung
- Zur Erhöhung der Redundanz gegen Ausfälle
 - Viele kleine statt ein großer Rechner
- Flexibler bei Hardwareauswahl
- Wenn man die einzelnen Komponenten des Mailsystems trennt, lassen sich SPOFs komplett vermeiden

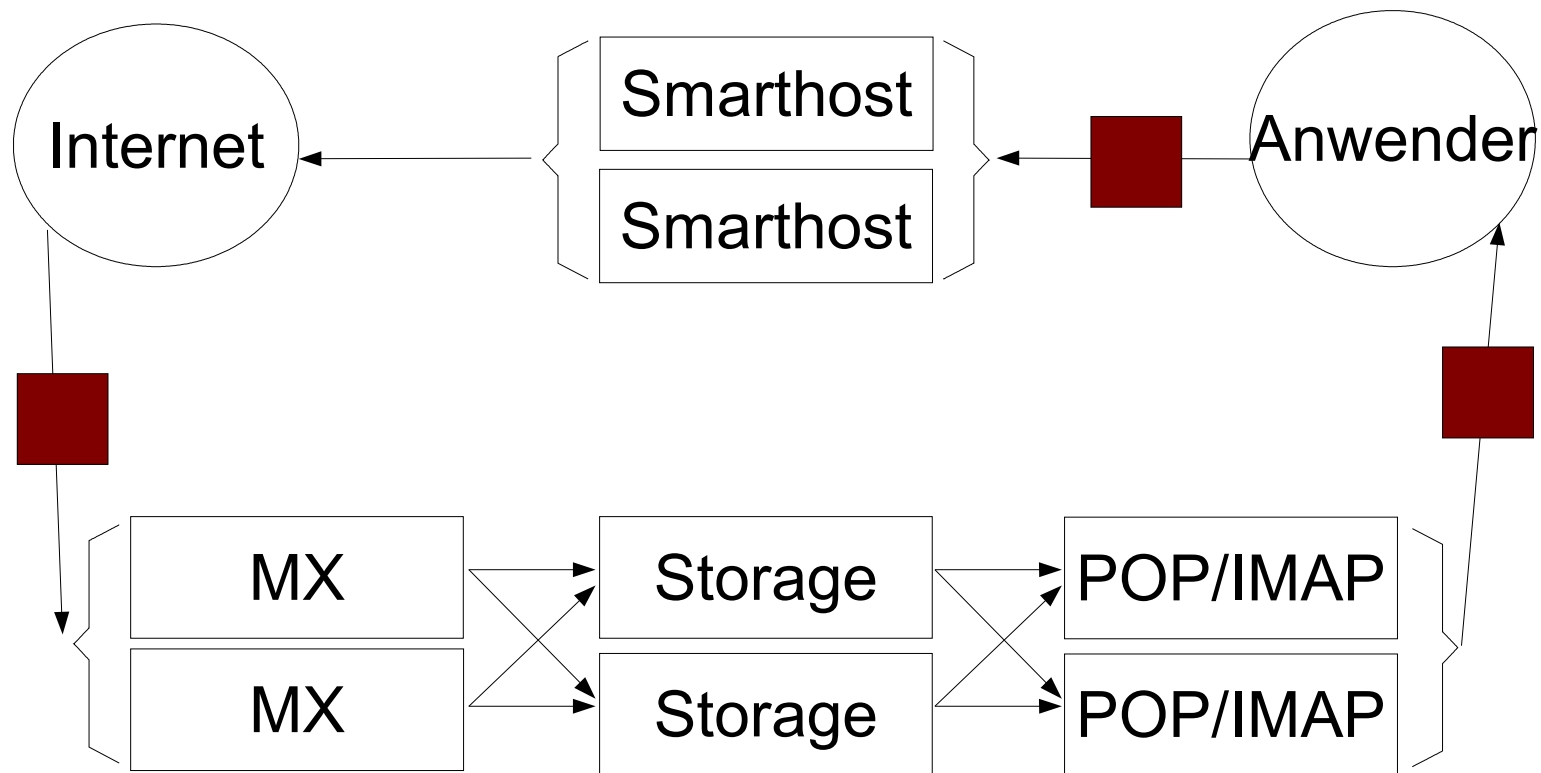
Verteilung per A-Record



Verteilung per MX-Record



Verteilung mit Load-Balancer

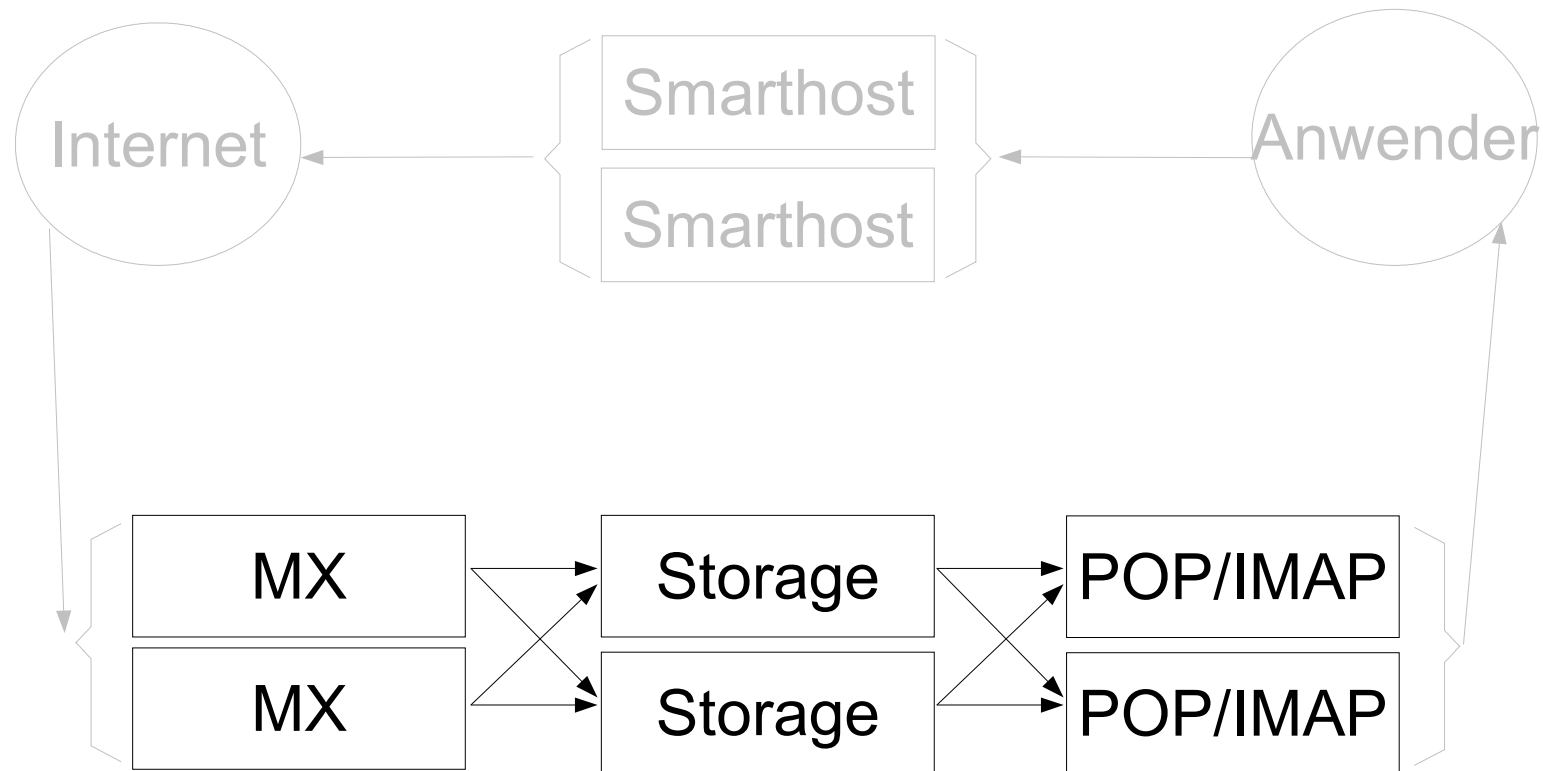


Verteilung mit Load-Balancer

- Arbeitet auf IP-Ebene
- Keine DNS-Umkonfigurierung bei Ausfall oder Erweiterungen
- Flexiblere Lastverteilung (z.B. bei verschiedener Leistungsfähigkeit der Server)
- Muß redundant ausgelegt sein
- Alternative: Clustering der Server mit IP-Failover

Storage

Storage-Architektur



Storage

- Welches Datenformat wird für die Mailboxen verwendet?
- Wie werden die Mailboxen verteilt?
- Welche Protokolle werden zur Ablage bzw. Abfrage benutzt?

Datenformat

- Ein File pro Mailbox
- Ein Directory pro Mailbox, ein File pro E-Mail
- plus Metadaten in Index-Files
- Keine UNIX-User für jeden Mailuser!
- Datenbank

Datenformat: mbox

- Klassisches Unix-Format
- Alle Mails einer Mailbox in einer Datei
- Schneller lesender Zugriff
- Änderungen an einer Mailbox langsam und aufwändig
- Bei mehreren Prozessen Locking notwendig
- Nicht NFS-sicher

Datenformat: Maildir

- Ein File pro E-Mail
- Schneller wahlfreier Zugriff
- Übersicht/Suche langsam, weil viele Dateien geöffnet werden müssen
- Langsamer Zugriff bei sehr vielen E-Mails
- Ändern und Löschen von E-Mails sehr einfach
- Kein Locking notwendig
- NFS-sicher

Datenformat: Indexdateien

- Meta-Informationen in Index-Dateien
- Suchindices in Index-Dateien
- Umgehen der Schwierigkeiten der Mailbox- und Maildir-Formate
- Locking notwendig
- NFS-sicher?

Datenformat: Datenbank

- Relationale Datenbank
- Was wird gespeichert?
 - Komplette E-Mail
 - Nur E-Mail, keine Anhänge
 - Nur Header / Metadaten

Datenformat: Datenbank

- Vorteile:
 - Bekannte Datenbanksoftware mit transaktionssicherem Verhalten kann genutzt werden
 - Verteilung durch Datenbanksoftware möglich
 - Können mit großen Datenmengen umgehen
- Nachteile:
 - Volles ACID-Transaktionsmodell ist aufwändig, eine Datenbank „kann viel mehr als nötig“
 - Dadurch langsamer

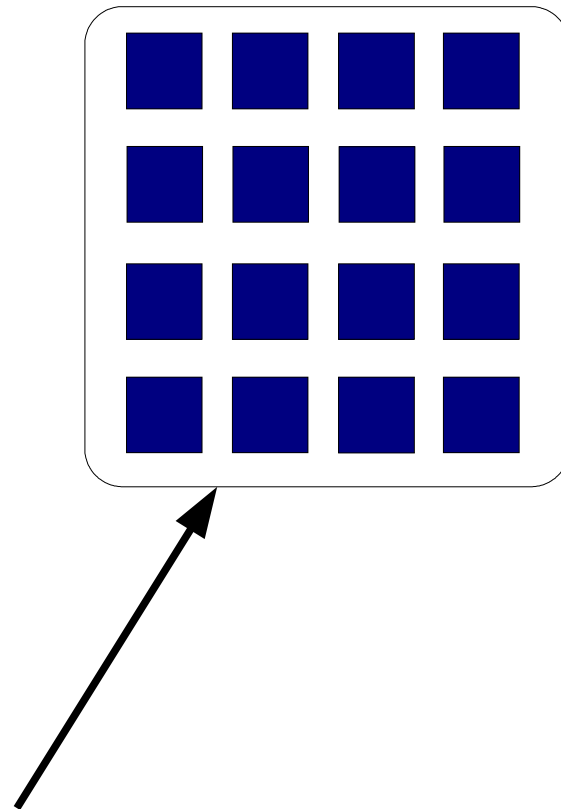
Datenformat: Zeilenumbruch

- Zeilenumbruch unter Unix:
<LF>
- Zeilenumbruch in SMTP und POP/IMAP
<CR> <LF>
- Bei Speicherung mit <LF> zweifache
Konvertierung
- Bei Speicherung mit <CR> <LF> keine
Konvertierung und korrekte Längenberechnung
der E-Mail

Verteilung: Einführung

- Mehrere Varianten für eine Verteilung
- Immer aus der Sicht des Mailsystems
- Das darunterliegende Storage-System kann schon eine Verteilung vorsehen, von der das Mailsystem nichts weiß (RAID, Datenbank)
- Verteilung sagt erstmal nichts über die Duplizierung einzelner E-Mails (Redundanz)

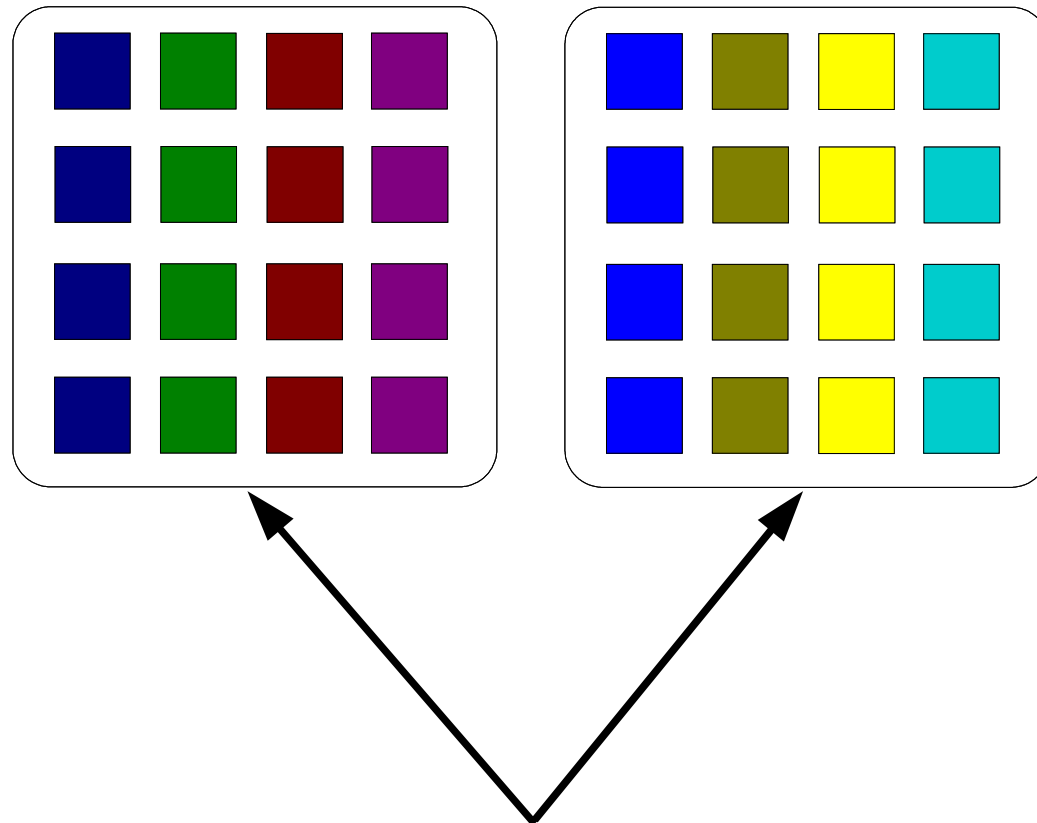
Verteilung: Zentral



Verteilung: Zentral

- Vorteil:
 - Einfach zu konfigurieren
- Nachteil:
 - Single Point of Failure?
 - Effizienz hängt vom zugrundeliegenden Mechanismus ab

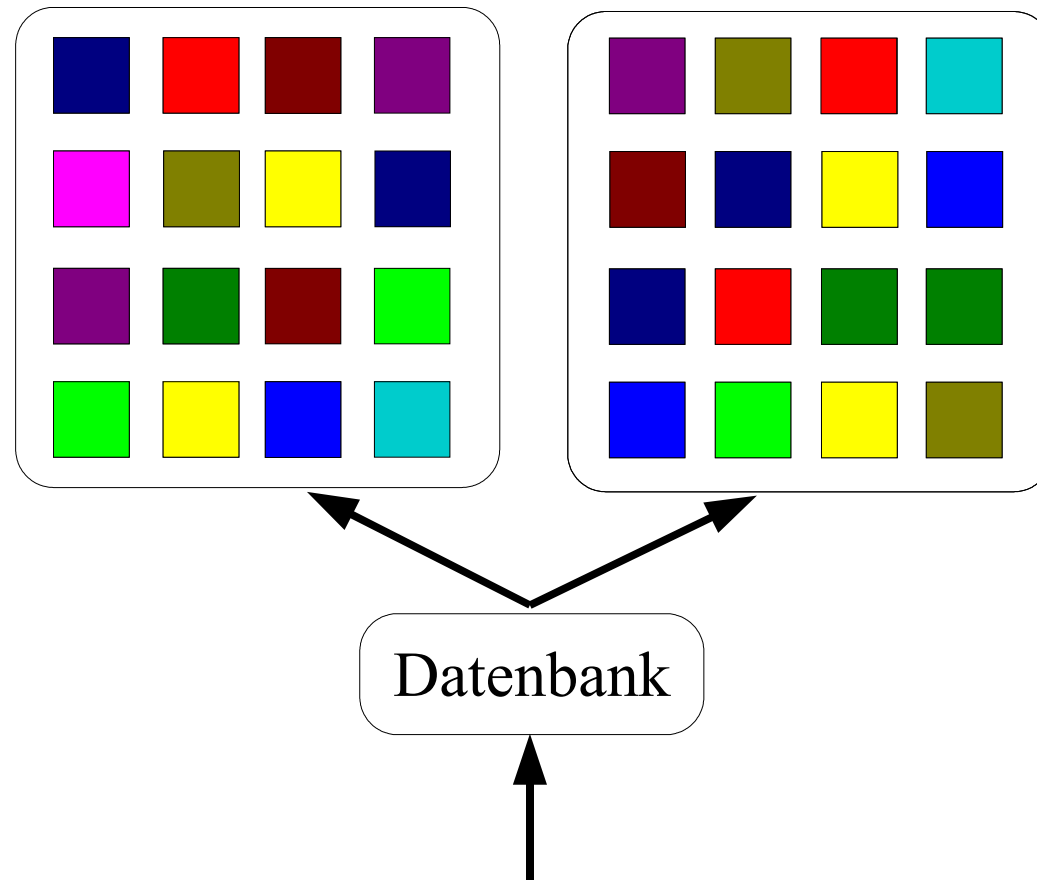
Verteilung: algorithmisch/statisch



Verteilung: algorithmisch/statisch

- Vorteil:
 - Zugriff einfach
- Nachteil:
 - Verteilung muss einmal für immer festgelegt werden, nachträgliche Änderungen kaum möglich
 - Bei ungleichmäßiger Belegung der Mailboxen oder verschiedener Hardware Lastverteilung schwierig

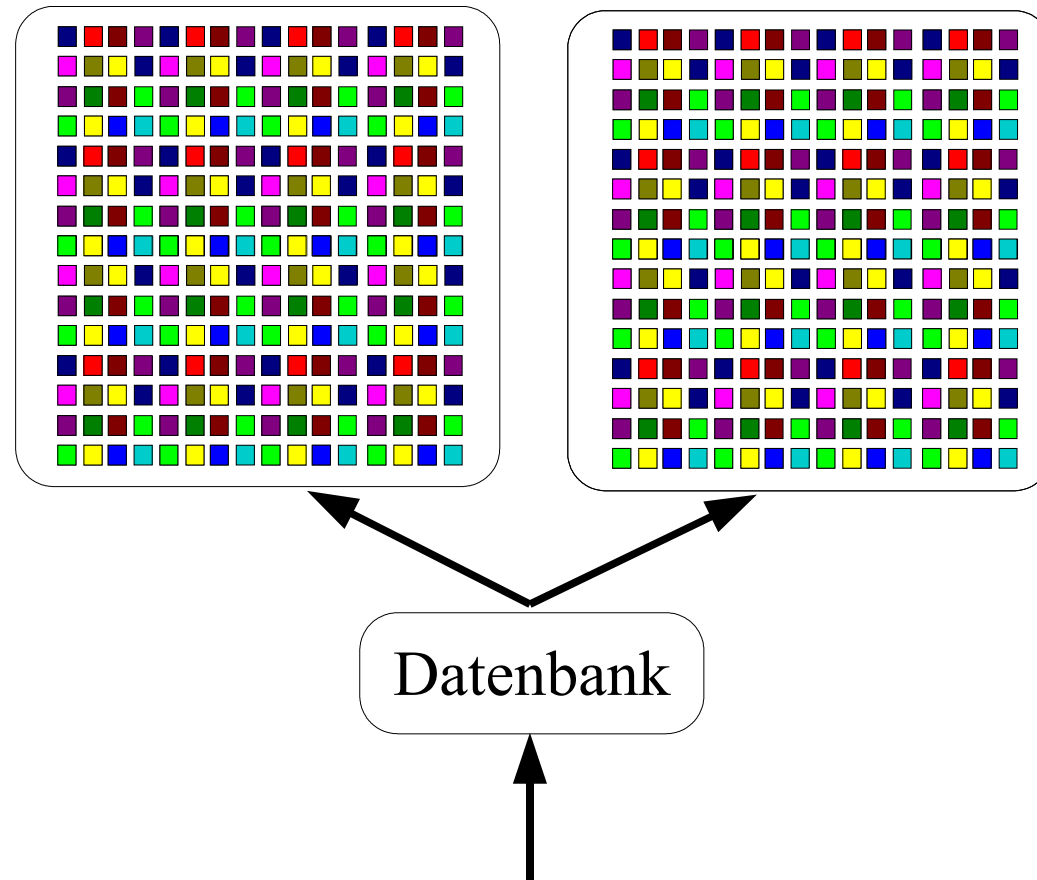
Verteilung: zufällig/dynamisch



Verteilung: zufällig/dynamisch

- Vorteil:
 - Lastausgleich bei unterschiedlicher Nutzung der Mailboxen oder verschiedener Hardware möglich
 - Mailboxen können migriert werden
- Nachteil:
 - Zusätzliche Datenbank für Zuordnung erforderlich (die auch wieder verteilt sein muss)

Verteilung: Auf Mailebene



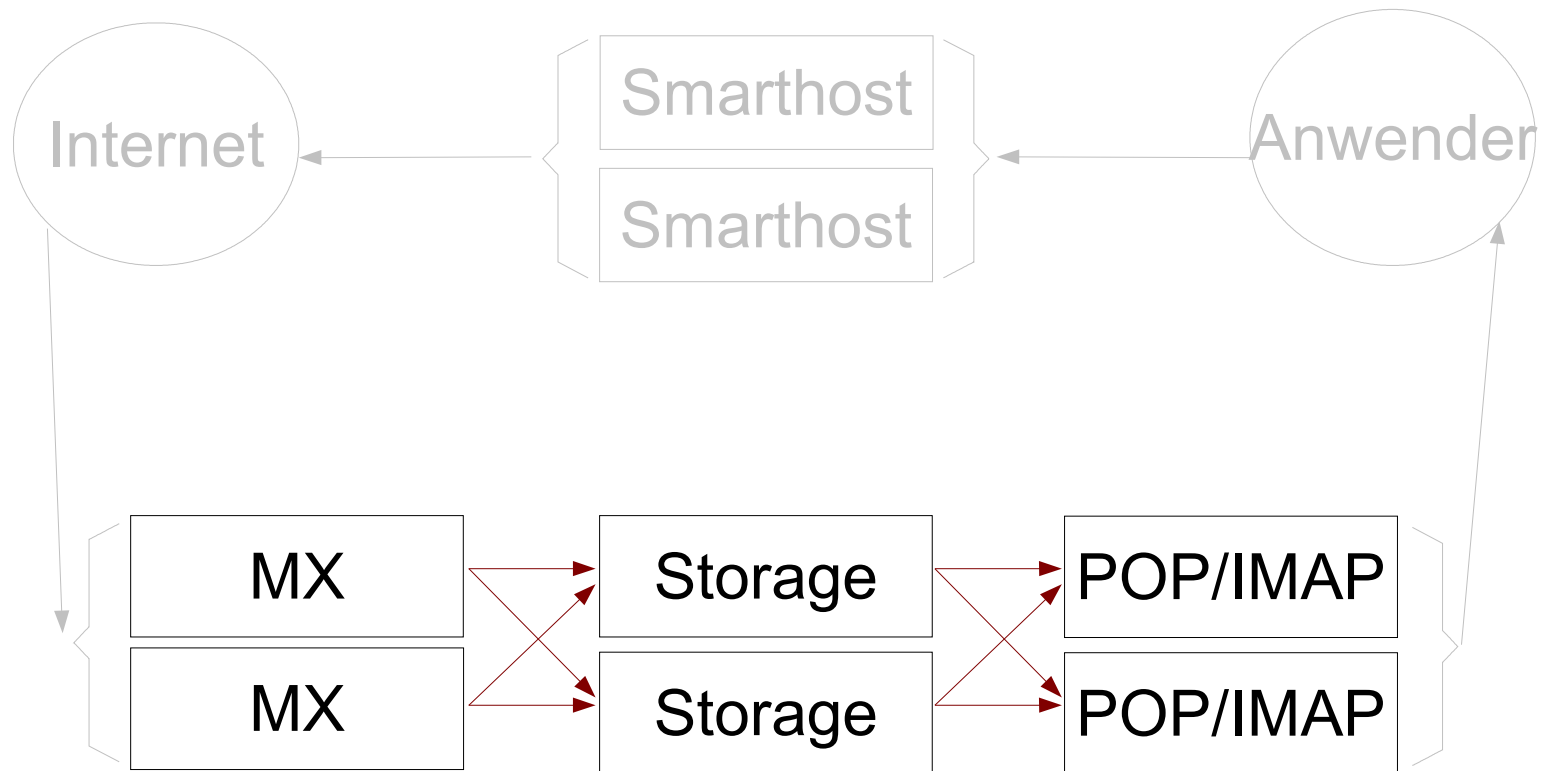
Verteilung: Auf Mailebene

- Vorteil:
 - Beste Lastverteilung
 - E-Mail an mehrere Adressaten braucht nur einmal Speicherplatz
- Nachteil:
 - Zusätzliche Datenbank für Zuordnung erforderlich (die auch wieder verteilt sein muss)

Verteilung: Replizierung

- Replizierung gegen Ausfälle auf drei Ebenen:
 - Hardware/OS: RAID
 - Datenbank (wenn vorhanden)
 - E-Mail-System
- Zwei- oder mehrfach
- Möglichst autokonfigurierend
 - Beispiel: Porcupine [Saito]

Storage: Protokolle



Storage: Protokolle

- Wie erfolgt die Ablage von E-Mails im Storage?
- Wie wird eine E-Mail aus dem Storage abgefragt?

Storage: Protokolle

- NFS-Mounts/SAN
- Datenbank-Protokolle (SQL)
- „Eigenentwicklungen“
- SMTP oder LMTP
- POP3/IMAP

NFS (Network File System)

- Locking-Problematik
- Hängende Clients, wenn Server offline

Datenbank-Protokolle

- Wenn eine relationale Datenbank benutzt wird
- Große Anzahl an kurzen Verbindungen sind problematisch für die meisten Datenbanken
- Daher Connection-Caching nötig

„Eigenentwicklungen“

- Notwendig bei Verteilung auf E-Mail-Ebene
- Anpassung aller Software notwendig

SMTP / LMTP

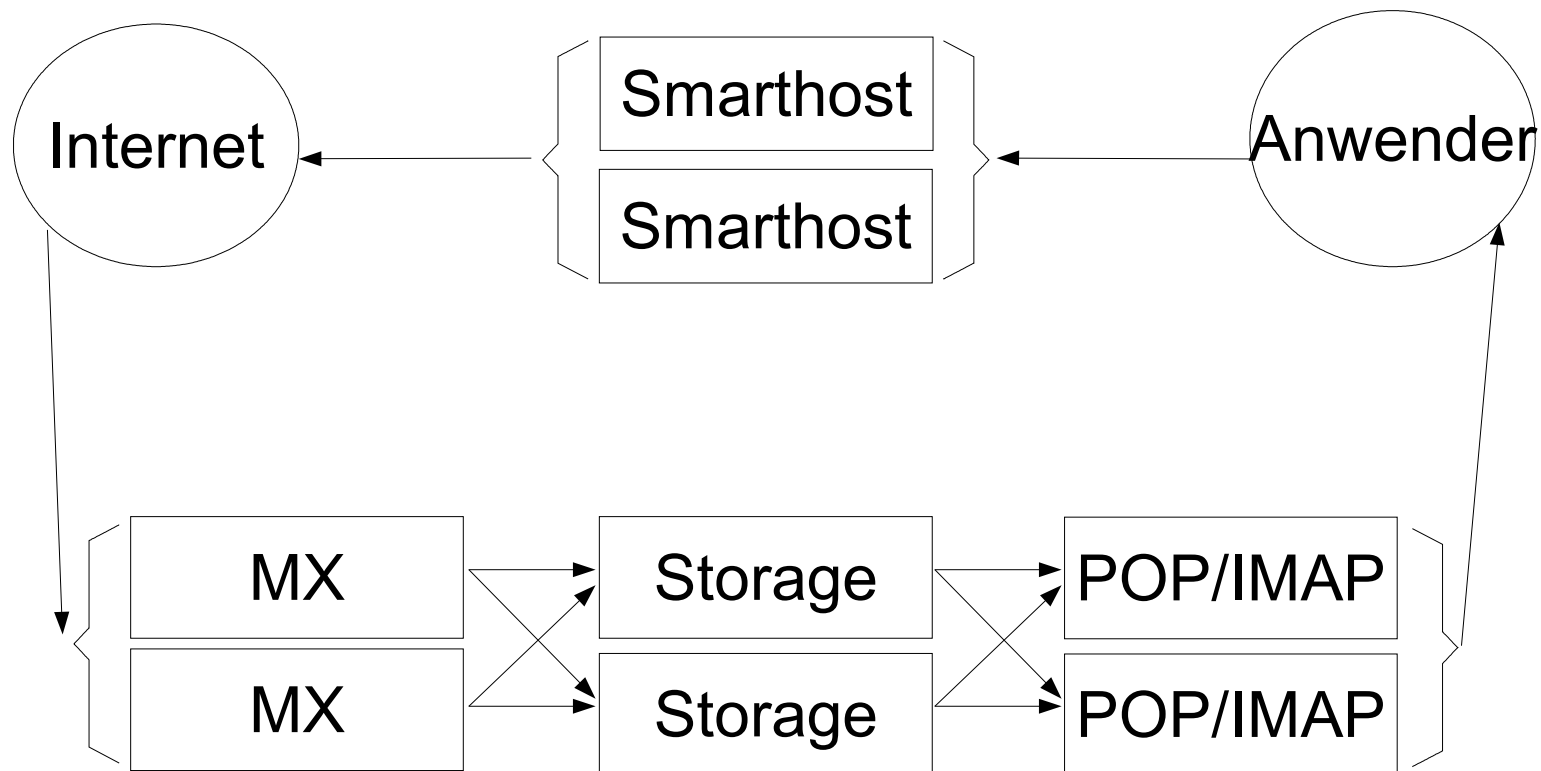
- „Normaler Mailserver“ auf Storage-Rechnern
- Nutzung von Standard-Protokollen
 - SMTP
 - LMTP (Local Mail Transfer Protocol, RFC2033, vereinfachtes SMTP ohne Queueing)

POP / IMAP

- Nutzung eines POP/IMAP-Proxyes
- Authentifizierung auf dem Proxy, danach Weitergabe der Verbindung
- Eventuell Änderung des Protokolls, weil zwischen Proxy und Server keine Authentifizierung nötig ist
- Beispiele: POPular, Perdition [Horman]

Konfiguration

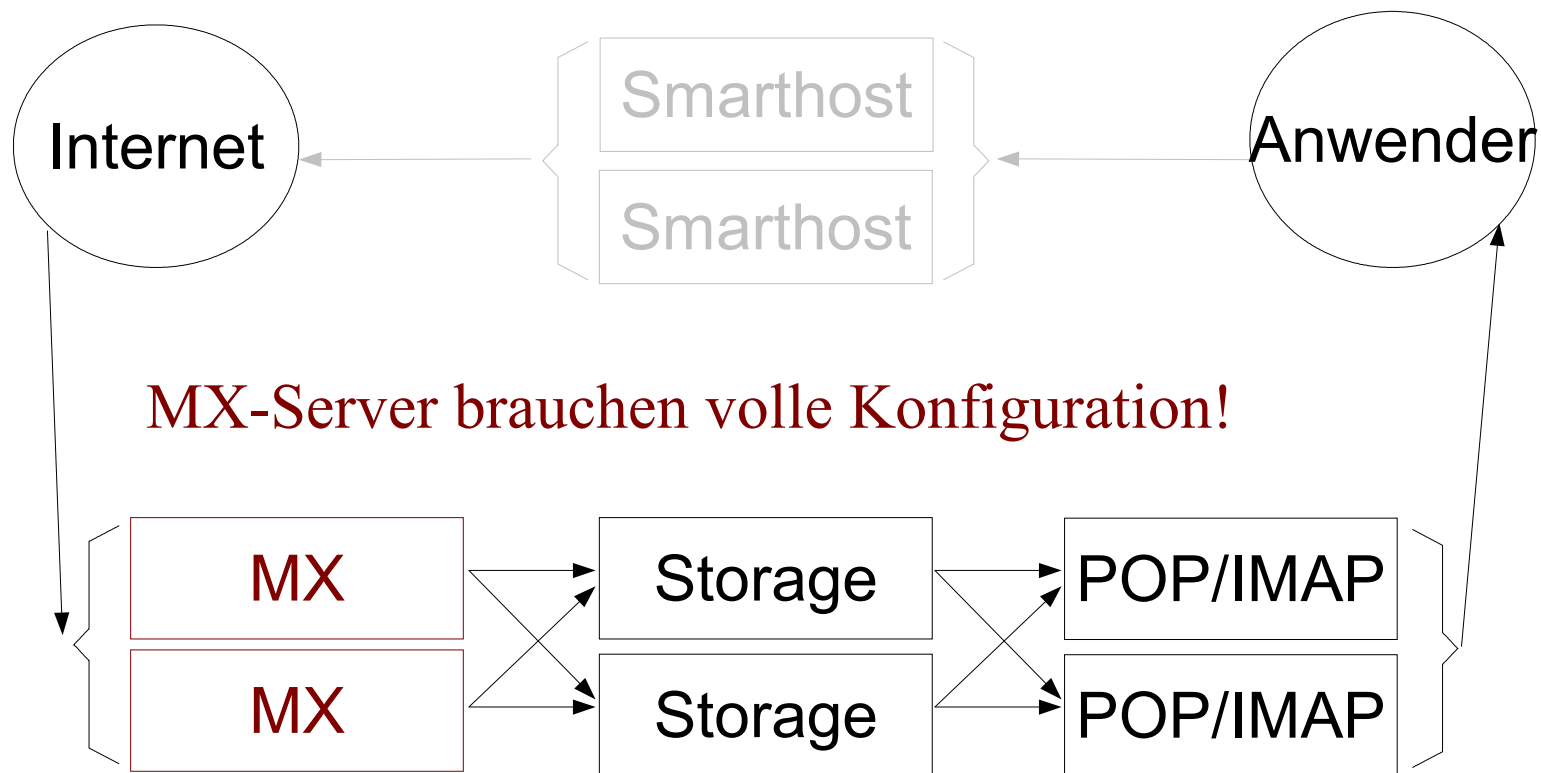
Konfiguration



Was ist zu konfigurieren?

- Routinginformationen E-Mail-Adressen
 - Zuordnung E-Mail-Adresse zu Mailbox
 - Weiterleitung, Autoresponder („vacation“), SMS, ...
- Zuordnung Mailboxname zu Storage
- Passwörter
 - POP/IMAP-Mailboxen, SMTP AUTH
- User-Preferences
 - Spamfilter, Folder

MX-Server-Konfiguration



Konfigdatenbank

- Files kommen bei dieser Größenordnung nicht mehr in Frage
- Optionen:
 - Relationale Datenbank
 - LDAP

Relationale Datenbank

- Relativ langsamer Zugriff
- Dauerhafte Verbindungen nötig
- Komplexes Datenmodell mit Überprüfung im Server
- Transaktionssicherheit, Replikation eingebaut
- Komplexe Software, die schwer 24/7 online zu halten ist (Backups, Schema-Änderungen, ...)

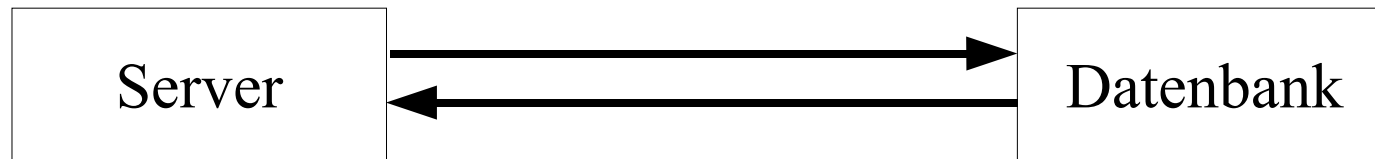
LDAP-Datenbank

- Abfrage (relativ) leichtgewichtig
- Keine komplexen Datenmodelle
- Integration mit anderen Daten im Unternehmen

Direktzugriff vs Dezentral

- Drei Alternativen für den Zugriff:
 - Direktzugriff auf die Datenbank von den Server aus
 - Direktzugriff mit Caching auf einzelnen Servern
 - Verteilung der Daten aus zentraler Datenbank an die einzelnen Rechner
- Wie schnell werden Änderungen aktiv?
- Single Point of Failure?

Direktzugriff



- Änderungen werden sofort wirksam
- Viele konkurrierende Zugriffe
- Bei Ausfall der Datenbank geht nichts mehr

Direktzugriff mit Cache



- Entlastung des Datenbankservers
- Aktualität je nach Caching-Strategie
- Cache-Invalidierung?
- Proxy-Server kann gleichzeitig die Verbindung zur Datenbank halten/multiplexen

Verteilung (Push)



- Regelmäßiger oder kontinuierlicher Datenbank-Export
- Lastverteilung
- Kein SPOF mehr

Verteilung (Push)

- Sehr schneller Zugriff
- Funktioniert deshalb, weil Abfragen viel häufiger sind als Änderungen
- Ab gewisser Größe kein kompletter Export mehr in vernünftiger Zeit!

Konfigurationsänderungen

- Alle Konfigurationsänderungen sollten im Betrieb ohne Downtime möglich sein
- Restartzeiten vermeiden
- Selbst Ausfälle im Sekundenbereich können viele User betreffen

Tuning

Tuning

- Hardware
- Software
- Erst messen, dann optimieren!

Tuning: Hardware

- Ressourcen:
 - CPU
 - RAM
 - Hard Disk
 - Netzwerk

Tuning: Hardware

- Schon mit einem Rechner kann man eine Menge E-Mail verarbeiten
- Mehr RAM ist immer gut
- Schnelle Platte/RAID lohnt sich, ist aber nicht alles

Tuning: Software (I)

- Software „von der Stange“ ist nicht unbedingt performant
- Forks/Execs sparen
- UDP ist schneller als TCP
- Keine Konvertierungen und In-Memory-Copies
- DB-Files statt Flatfiles
- DB-Lookups über Indices

Tuning: Software (II)

- Keine zu großen Verzeichnisse (Queues, Maildir)
- Was passiert, wenn die Queue vollläuft?
- Caching?

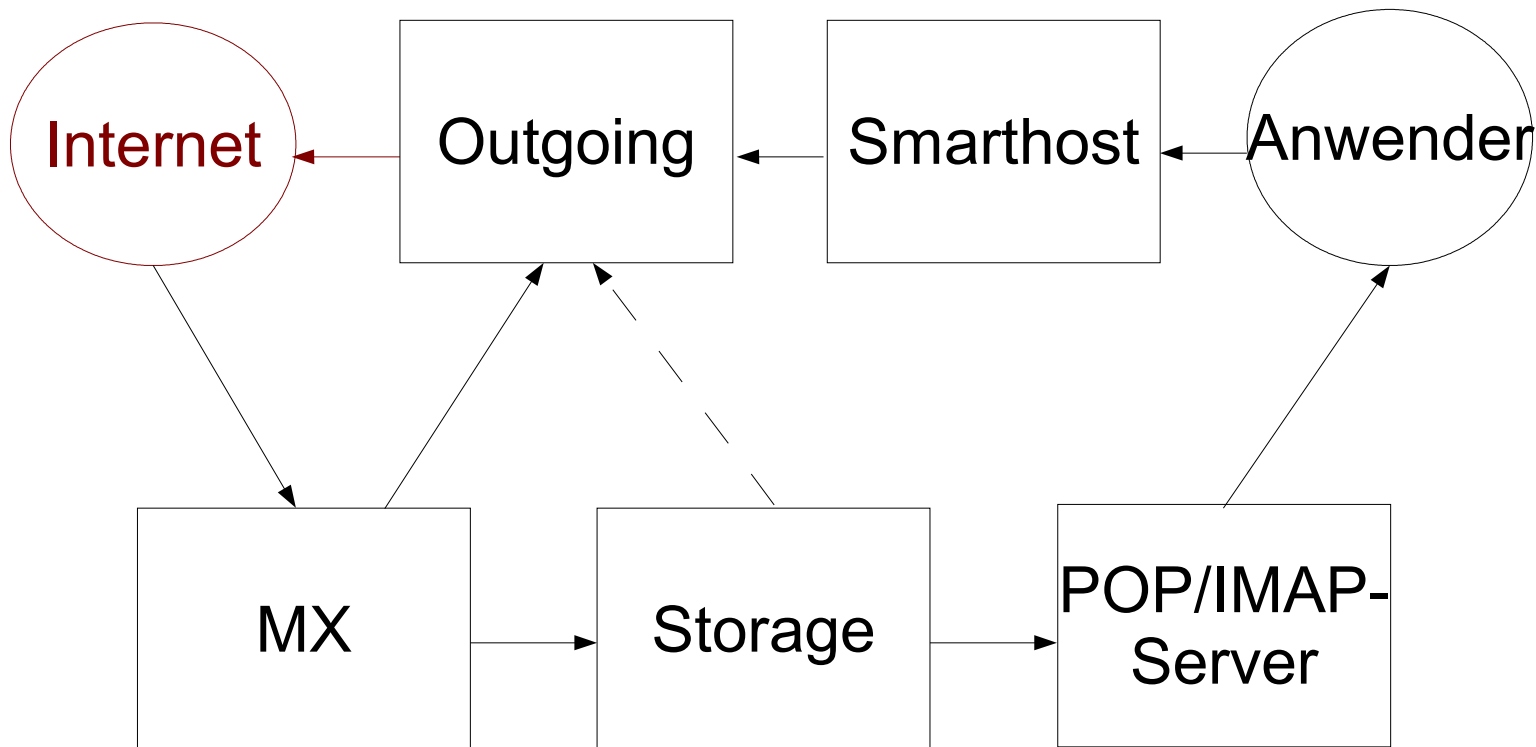
- Weniger ein Problem: Socketanzahl und dergl., weil man vorher gegen andere Limits stößt

Zu erwartende Probleme

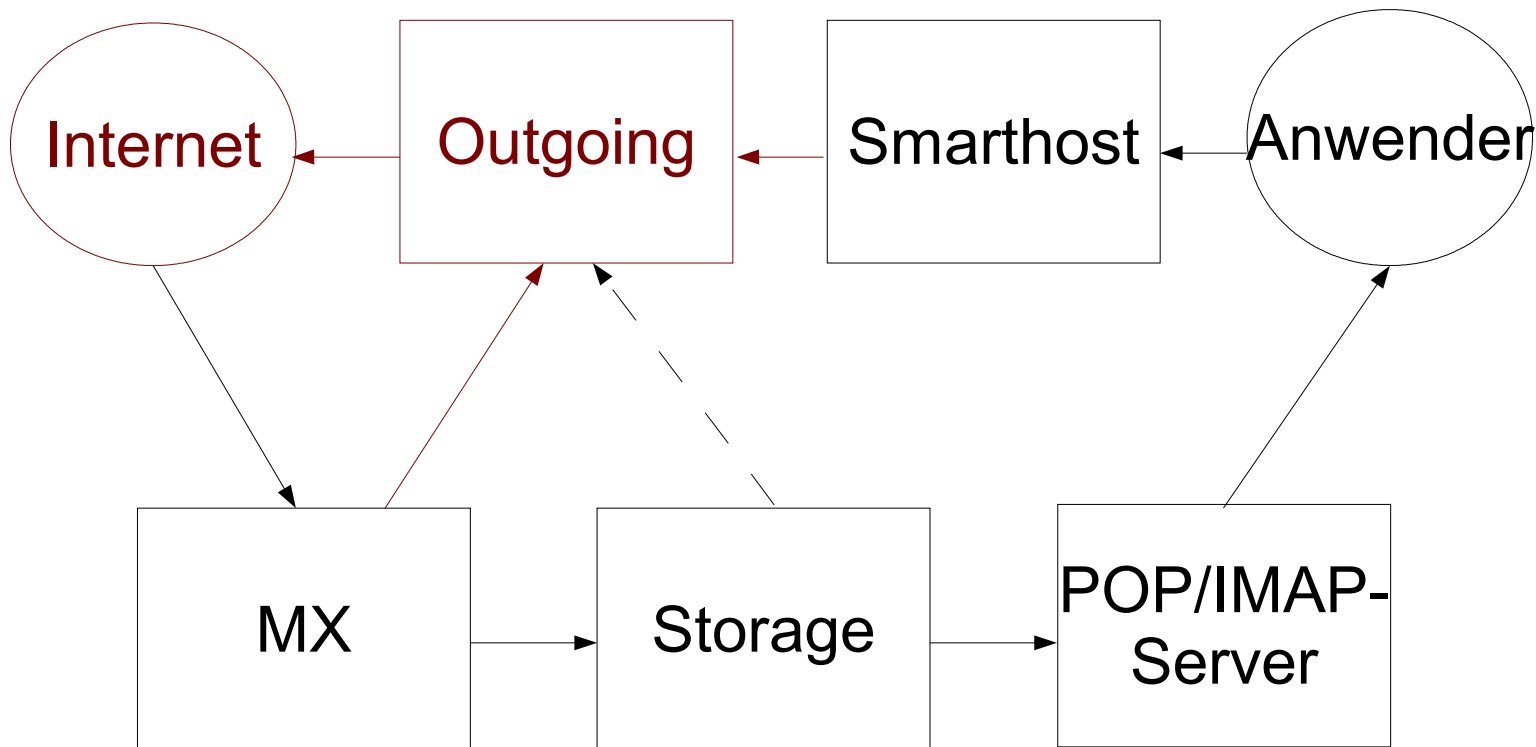
Zu erwartende Probleme

- Es gibt keine „kleinen Probleme“ mehr in einem großen Mailsystem
- Bounces verstopfen die Queue
- E-Mail-Loops
- Rückstaus

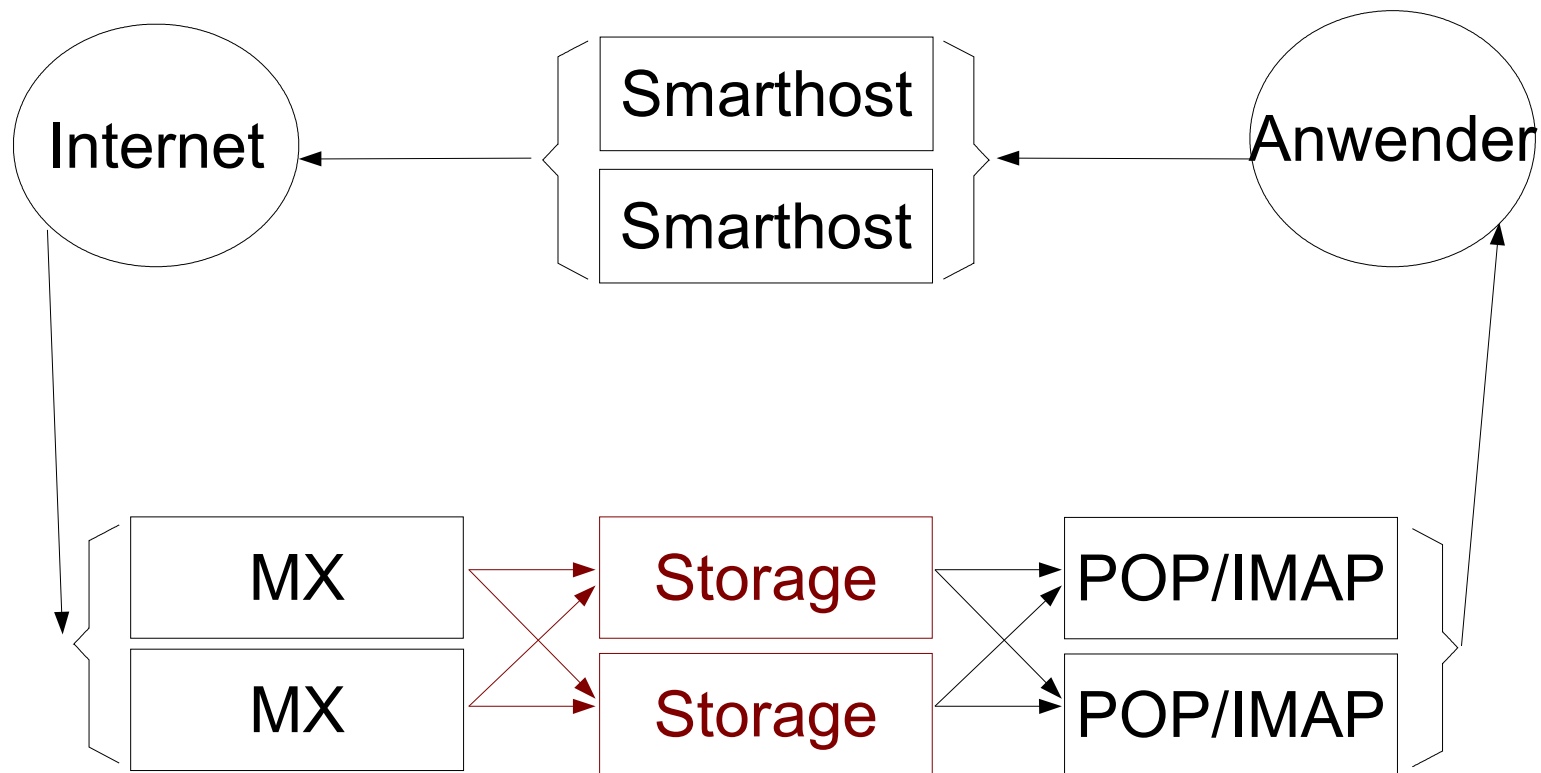
Rückstau auf dem Smarthost



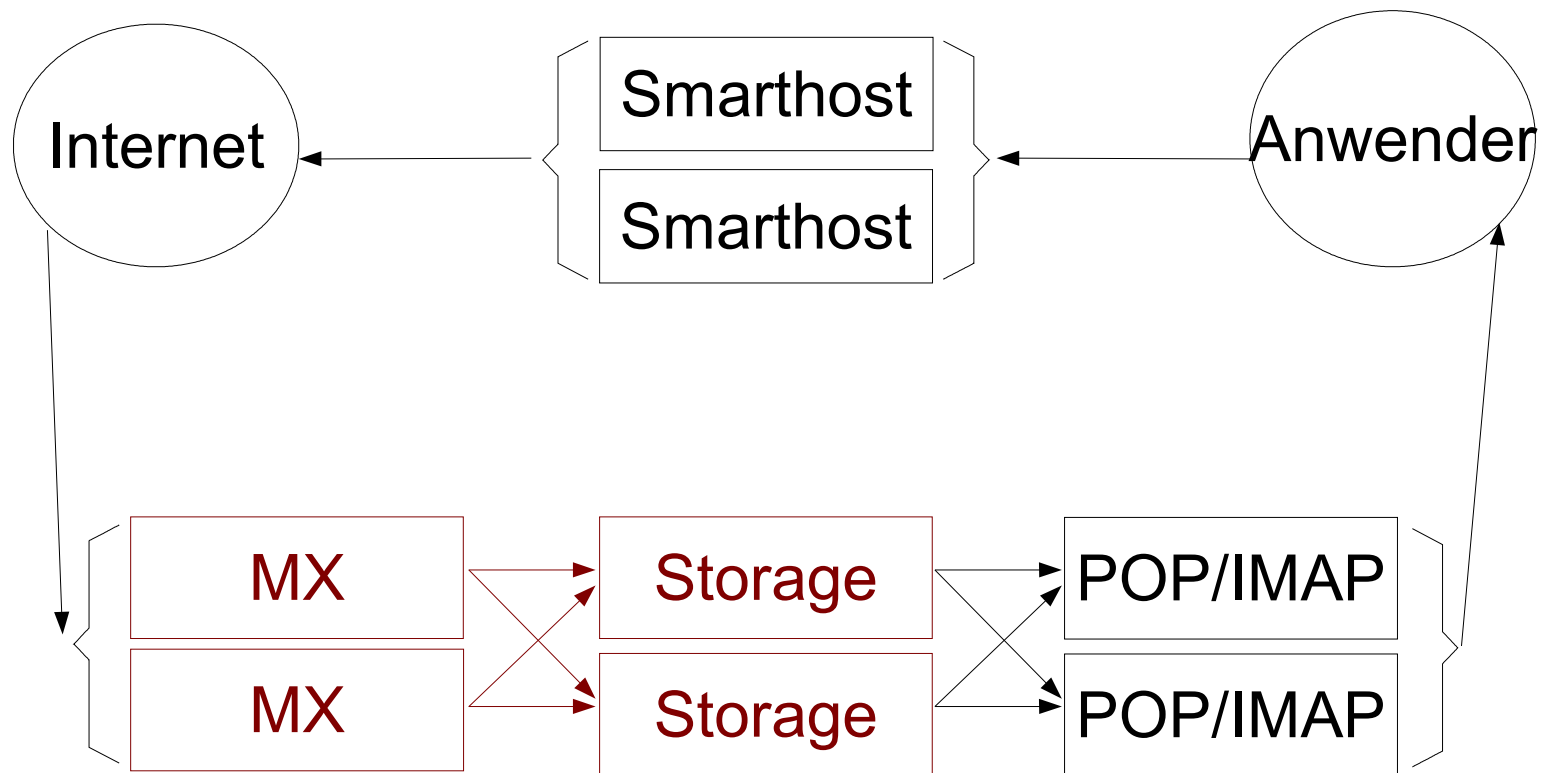
Rückstau auf dem Smarthost



Rückstau im Storage-Server



Rückstau im Storage-Server



Rückstau

- Analoge Probleme, wenn Spam- oder Virenfiler nicht nachkommt
- Begrenzung des Load bzw. der Prozessanzahl
- Queue only
- Monitoring und Umkonfiguration?
- Testen!

Diverses

POP vs IMAP

- POP dient zum **Abholen** von E-Mails aus einer Mailbox
 - Operationen: Liste anzeigen, Lesen, Löschen
- IMAP dient zum **Verwalten** von E-Mails auf dem Server
 - Zusätzliche Funktionen: Folder, Suchen auf dem Server, Flags

POP vs IMAP

- Beides hat seinen Platz
- IMAP erzeugt eine viel höhere Last: Mehr Daten auf dem Server, offene Verbindungen, Suche, ...
- IMAP-Protokoll viel komplexer

Öffentliche Hostnamen

- Möglichst von Anfang an verschiedene Rechnernamen für verschiedene Aufgaben vergeben
- Beispiele:
 - smtp.example.com / mail.example.com
 - pop.example.com
 - imap.example.com

Verschlüsselung

- SSL/TLS möglich bei SMTP, POP, IMAP
- Wichtig vor allem für authentifizierte Verbindungen (auch bei SMTP AUTH)
- Wesentlich mehr Rechenaufwand für beide Seiten, vorallem beim Verbindungsaufbau
- Nutzung eines SSL-Proxies?
- Hardwareunterstützung möglich (acceleration cards)

DNS

- Ohne DNS keine E-Mail
 - Abfrage von MX- und A-Records
 - DNSBLs
 - SPF/SenderID, DomainKeys, MTAMARK
- Ebenfalls Verteilung der DNS-Server zur Lastverteilung und Erhöhung der Zuverlässigkeit
- DNS-Cache-Größe beachten
- DNS-Cache auf/für jeden Outgoing Mailserver?

Spam und Viren

- Gestern im Vortrag von Marc Martinec über amavisd-new:
 - 1/2 throughput durch NULL content filter im postfix
 - 1/2 throughput durch NULL amavisd
 - 1/2 throughput durch SpamAssassin
 - 1/2 throughput wenn alle Tests aktiviert
- Können wir mit 1/16 des throughputs leben? Das heißt 16mal so viel Hardware!

Spam und Viren

- Möglichst früh abfangen
- IP-basierte Filter statt Inhaltsanalyse
- Software-Performance ist wichtig
- Möglichst nicht annehmen, sondern direkt ablehnen
 - Einbau in MX-Server
- Auch Outbound-Filterung

DNSBLs (DNS Blacklists)

- Wichtiges Anti-Spam-Verfahren
- Bei hoher Mail-Last sind die Laufzeiten zu groß (Timeouts!)
- Lokale Kopie verwenden (rsync)
- Integration mehrerer DNSBLs in eine lokale Liste erspart Anfragen
- „latency does matter“ (wenn das System an der Lastgrenze fährt)

Logging

- Ausführliches Logging
- Debuglogging flexibel einstellbar und im Betrieb an- und abstellbar
- Logging wohin?

Monitoring

- Das übliche: Load, HD-IO, Netzwerk-IO, ...
- Queue-Größen
- Maildurchsatz
- Spam- und Viren-Erkennungsraten
- Top-Absender, -Empfänger
- Was machen die Prozesse?

Bounces

- Für jede E-Mail, die angenommen wurde, aber nicht ausgeliefert werden kann, muss ein Bounce erzeugt werden
- Das wo irgend möglich verhindert werden, damit die Bounces nicht die Queue verstopfen
- Komplette Konfiguration im MX
- Check auf volle Mailboxen vom MX aus
- Spamprüfung vor Annahme

Was machen bei Ausfällen? (I)

- Ausgefallene Elemente sollten möglichst automatisch umgangen werden
- Selbst sehr kurze Ausfälle werden von tausenden Anwendern gesehen!
- Ausfälle lokal begrenzen (Rückstaus!)
- Mit Ausfällen bei anderen umgehen

Was machen bei Ausfällen? (II)

- Welche Ausfälle kann der User sehen?
 - POP/IMAP besonders kritisch
- Wie informieren wir den User?
- Welche können wir vor ihm „verheimlichen“?

Routing ausgehender E-Mail

- Die meisten E-Mails werden wir sofort los
- Einige bleiben ewig liegen
- Weiterleitung auf Extra-Server?

Backup

- Während des Tages aus Lastgründen meist nicht möglich
- Sehr kurzlebige Daten: Viele E-Mails werden nie gesichert werden, weil sie gelesen und gelöscht wurden, bevor das Backup läuft
- Backup weniger wichtig, wenn Storage redundant ausgelegt ist
- „Abzweigen“ aller E-Mails am Eingang?

Fazit

- Mailsysteme lassen sich gut verteilt ohne SPOFs betreiben
- Es gibt viele verschiedene Ansätze, die alle ihre Vor- und Nachteile haben
- Bei einem größeren Mailsystem ist nicht einfach nur alles größer, es stellen sich neue Probleme
- Optimieren Sie nicht zu früh!

Noch Fragen?

Jochen Topf – www.jtic.de – jochen.topf@jtic.de