The Big Picture:

Der OSS-Mailcluster

von Raiffeisen OnLine

Wer oder was ist Raiffeisen OnLine?!

- ISP in Südtirol
- Eigenständige Genossenschaft
- Klassischer Provider: Internet Access, Mail, VoIP Hosting, Web-Projekte
- Früher war die Technik größtenteils ausgelagert
- Mittlerweile verfügen wir über eine eigene kleine aber schlagkräftige System-Abteilung

Worum geht es in diesem Vortrag?

- Eine erfolgreiche Migration
- Interessante Postfix/Amavis-Installation
- Viele kleine Feinheiten rund um selbige
- Kurze Konfigurationsausschnitte
- Strategische Überlegungen
- Eigene Erweiterungen und Entwicklungen
- Spannendes Log-System

Technische Highlights

- Durchgehend redundante Architektur
- Virtualisierung "extreme"
- Überlegungen zum Thema Storage
- Datenbank-Partitionierung
- Loadsharing, ClusterIP, Multicast (Layer 2)
- Gefilterter Log-Zugriff in Echtzeit für Kunden

Vorgeschichte

- Mailfilter im Outsourcing
- Ein eigener kleiner Cluster sollte her
- Budget und Ressourcen (Tests) wurden reserviert
- Eine Eigenlösung (auf OSS-Komponenten basierend) stand **nicht** zur Debatte
- Gründe dagegen: Entwicklungskosten, Wartungsaufwand, Support-Fragen usw.

Erfahrung mit kommerziellen Produkten

- Erste Angebote wurden eingeholt
- Preise teilweise erschreckend hoch, zudem oftmals ohne echte Redundanz (nur Standby)
- Ein kleiner Cluster den wir zu Testzwecken erhielten lief erst mal richtig gut
- Wir haben die Hälfte unseres Mailverkehrs dorthin umgeroutet...
- ...und am Freitag Nachmittag stand dann alles still

Wenn der Gegner einem zuspielt

- Die Tatsache, dass besagter Hersteller (während der Verkaufsphase!) ganze drei Tage benötigte, um...
  - Den Fehler zu untersuchen
  - Das System wieder freizugeben
  - Und uns mitzuteilen, dass die Ursache vermutlich ein Stromausfall in unserem Datacenter war...
- ...wussten wir natürlich entsprechend auszuschlachten
- Denn: wie sieht wohl der versprochene Support aus, wenn das Ding erst mal bezahlt ist?

Die Gunst der Stunde...

- ...haben wir natürlich genutzt!
- Ärmel hochgekrempelt
- Zwei freie Server aufgetrieben
- System mit Postfix, Amavis und den üblichen Verdächtigen vorbereitet
- Ziel: kurzfristig ein weiteres (OSS-basiertes) System mit ins Rennen zu schicken

Auch OSS braucht gute Verkäufer

- Wir waren überzeugt von der Überlegenheit einer eigenen OSS-basierten Plattform – doch diese Überzeugung alleine reichte damals nicht aus
- Erfahrung mit soviel Mail-Traffic hatten wir keine
- Wir konnten mit unseren Argumenten jedoch erwirken, dass unser "eigenes" Produkt mit ins Rennen aufgenommen wurde - mit strengen Auflagen bei der Kostenrechnung

Vorbereitung auf eine harte Prüfung

- Test-Systeme waren schnell aufgesetzt
- Es fehlte ein Zugriff auf die Quarantäne
- Ein Pre-Queue Filter war uns nicht geheuer
- Erweiterungen für's Webmail waren nicht oder nur mit hohem Aufwand möglich (Closed Source)
- Ein zusätzliches Portal war keine schöne Option
- Maia und Co schieden damit aus, eine Eigenentwicklung musste her

Schrittweise Migration

- Das Projekt wurde aufgeteilt in mehrere Schritte:
  - Grund-System mit ausreichend Features, um unseren internen Test zu bestehen
  - Ausarbeitung einer robusten und zukunftssicheren Architektur, schließlich sollten idealerweise weitere eigene Komponenten folgen
  - Tiefe Integration dieser Plattform in unsere internen (Support-Mitarbeiter) und externen (Kunden) Web-Applikationen

Um jetzt nicht zu sehr ins Detail zu gehen:

- Wir konnten mit Zahlen belegen, dass auf drei Jahre gerechnet auch unter sehr strengen Kriterien und pessimistischer Erwartung von hohen Eigenaufwänden unser eigenes Projekt immer noch kostengünstiger war
- Der kleine Cluster (zwei Server, mehrere vServer) lief in der Testphase so gut, dass wir unseren Mailverkehr nie wieder zurückgeswitcht haben

Jetzt aber zum Eingemachten!

- Erster Abschnitt:
  - Aufsplittung in Einzelkomponenten
  - Jede Komponente wird doppelt ausgelegt...
  - ...und erhält einen dedizierten vServer
  - Zentrale Datenbank in mehrere solche aufgeteilt
  - Strategie für SMTP-Pfade

Jeder Dienst / jede Komponente doppelt

- Auch wenn ein Rechner locker die Last aushält
- Cold-Standby ist nett, wir wollen aber möglichst keine Unterbrechung wenn ein Dienst wegfällt
- Teilweise ist in den jeweiligen Protokollen Redundanz bereits vorgesehen...
  - Mehr als einen DNS-Server nutzen
  - Gleichwertige MX-Einträge für eingehendes SMTP
- ...teilweise aber auch nicht

Einfachstes Beispiel: DNS-Resolver

- Per Design redundant, sofern doppelt vorhanden
- Dedizierte Caching-Only DNS-Resolver
- DNS ist immer ein potentieller Angriffspunkt
- Nicht von außen erreichbar (Firewall), was direkte Angriffe erschwert
- Cache locker ein paar hundert MB groß
- Wir nutzen hierfür BIND

Ein dedizierter virtueller Server pro Dienst

- Erhöht auf den ersten Blick die Komplexität
- Hat aber enorme Vorteile:
  - Jeder Dienst bleibt für sich skalierbar
  - Zusätzliche Rechner bei Last-Spitzen
  - Schnellere Reaktionszeiten bei Sicherheitsproblemen
  - Sensible Teile: konservativ mit stabilen Distibutionen
  - Andere hingegen "on the bleeding edge"
- Kurz: man ist flexibler und kann schneller agieren

Überlegungen zur Virtualisierung

- Vollständige oder Paravirtualisierung ist nett, aber letzlich ein ziemlicher Overhead, wenn man lediglich Linux-Instanzen / -Partitionen möchte
- Live-Migration hat ihren Reiz – ist aber häufig nicht das, was man wirklich braucht
- Idealerweise skalieren Applikationen als solche
- Unsere Wahl fiel auf Linux-vServer

Was ist Linux-vServer?

- Ist im Grunde eine Partitionierung der Prozesse eines Systems mit Bordmitteln, welche teilweise erweitert worden sind
- Jeder virtuelle Server läuft in seinem eigenen Namespace, sieht nur die zugehörigen Prozesse
- Capabilities werden abgegeben
- Netzwerk-Administration (und damit auch FW-Regeln usw) ist für Guests nicht möglich

Vorteile von Linux-vServer

- Sehr schlanke Patches
- Wenig bis gar kein Overhead
- Da nur ein Kernel läuft: kein RAM-Verschleiß
- Mehr: http://www.linux-vserver.org

Datenbank-System: MySQL

- Jaaaaa, ich weiß ;-)
- MySQL hat so seine Tücken – und man erlebt Dinge, die man sich von einem RDBMS nie erwarten würde
- Dennoch: Replikationen aufsetzen ist ein Kinderspiel und halbwegs zuverlässig
- Entsprechendes KnowHow ist weit verbreitet

Warum wir gleich drei MySQL-Cluster nutzen

- Unterschiedlicher Workload
- Verschiedene Backup-Strategien
- Bei Bedarf schneller skalierbar
- Unterschiedliche Anforderungen an Verfügbarkeit

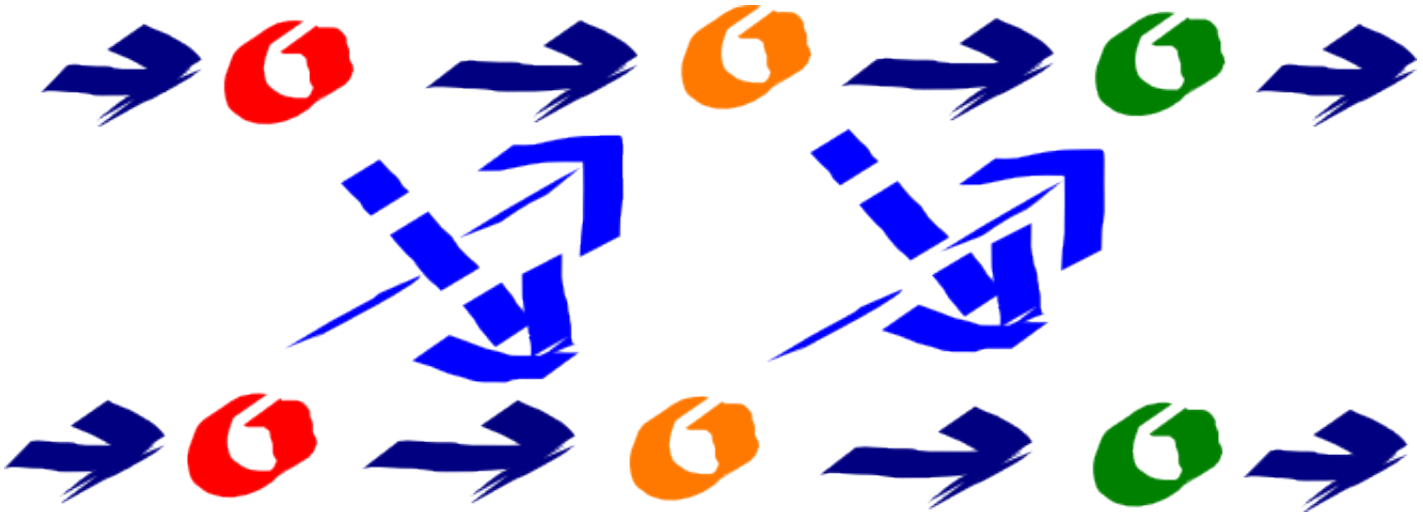
Strategien für SMTP-Pfade

- Ein- und Ausgang trennen schafft Ordnung
- Wer für eine Domain autoritativ ist steht im DNS
- Klassisches Problem: Domain-Übernahme, Mails von lokalen Absendern werden noch ins alte Postfach zugestellt
- SMTP-Server für eigene Kunden sollte NICHT wissen, welche Domains lokal sind – sondern wie jeder andere fremde MTA agieren müssen

Redundante SMTP-Pfade

- Jeder Knoten kann jederzeit ausfallen
- Failover ist wichtig, Failback aber auch!

IN -> Postfix -> Amavis -> Postfix -> OUT



Weitere Vorteile / Überlegungen

- Durch das mehrstufige Konzept kann man schnell die nach außen relayenden Ips auswechseln (BLs)
- Klassifizieren des eigenen Mailverkehrs, unterschiedliche Ausgänge für
  - Authentifizierte Benutzer
  - Relayende VIP-Kunden
  - Hosting-Server
  - Versender von (legalen) Bulk-Mails

Failover-Konfiguration für Postfix

- In der main.cf setzt man einen Filter:

```
content_filter = mf:[1.2.3.4]:10024
```

- ...in der master.cf konfiguriert man das Failover:

```
mf    unix  -  n  -  -  40  smtp
      -o ...
      -o smtp_fallback_relay=[1.2.3.5]:10024
```

Verschiedene Pfade mit Amavis

- Standard-Route:

```
$notify_method  = 'smtp:[1.2.3.5]:25';
$forward_method = 'smtp:[1.2.3.5]:25';
```

- Route für ausgehende Mails:

```
$policy_bank{'relayout'} = {
    originating => 1,
    notify_method  => 'smtp:[1.2.3.6]:25',
    forward_method => 'smtp:[1.2.3.6]:25'
```

Durch die Trennung der SMTP-Pfade...

- ...konnten wir unseren Kunden auch erlauben, autonom Domains auf dem Mailcluster anzulegen
- Dennoch sollte man strenge Checks durchführen
- Bei uns sind daraus zwei neue Produkte entstanden:
  - Das ROL MailCenter
  - Und das ROL ResellerCenter (für Mail-Produkte)

Screenshots ROL Mail Center: Logsuche

RaiffeisenOnline

ROL ResellerCenter

Benutzer: [redacted] abmelden

- ROL MailCenter
  - Domains
  - Quarantäne
  - **Maillog eingehend**
  - Reports
- Verwaltung

Log Suche

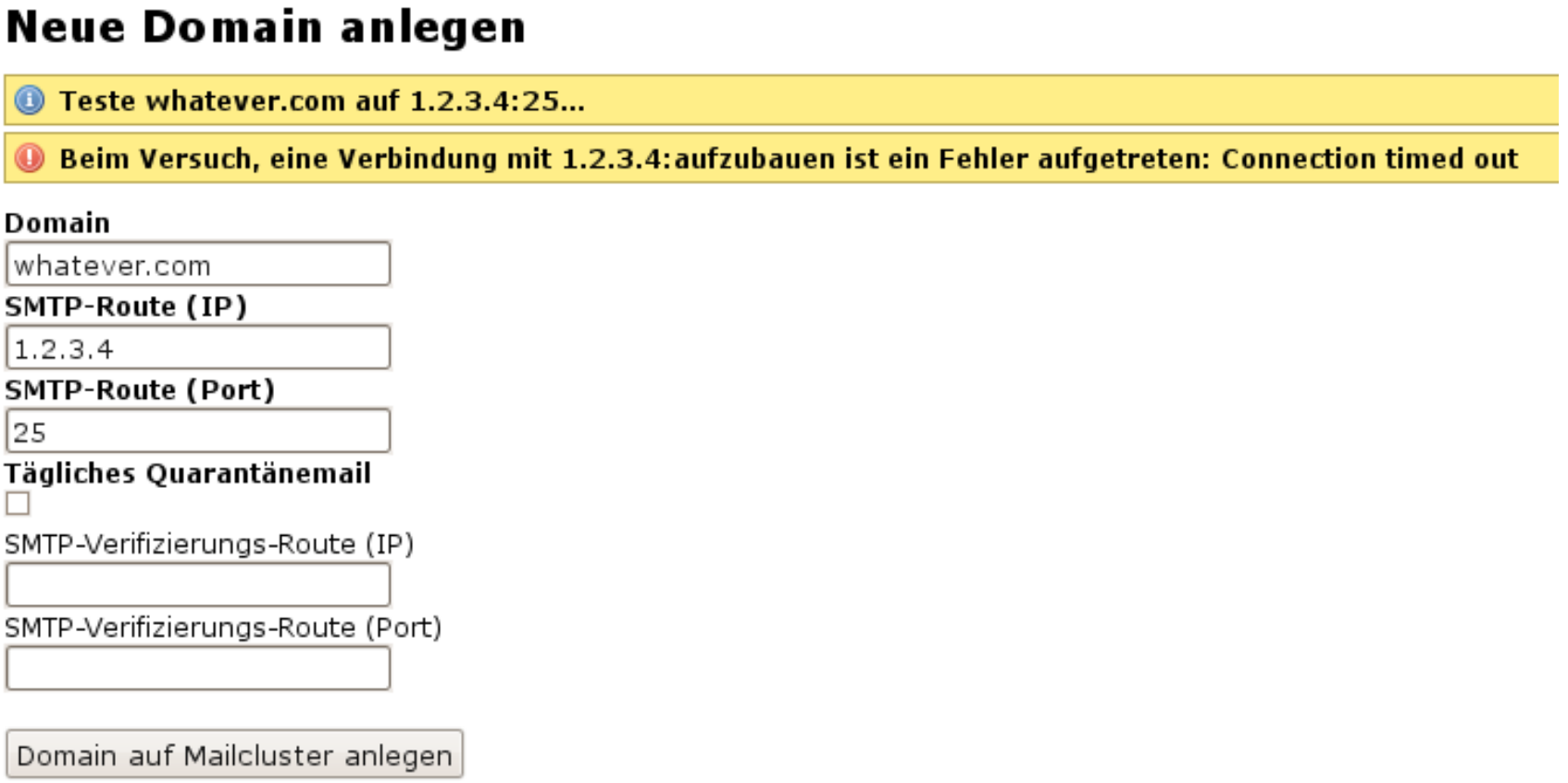
Zeige 15 Mails vom 26.03.2009 im Zeitraum ganzer Tag ,
dabei aber nur Mails von [redacted] an [redacted] @ [redacted]
Zeige zudem nur jene Mails, welche - bitte wählen -

« Zurück 1 2 ... 8 9 10 11 12 **13** 14 15 16 17 ... 81 82 Vorwärts »

Typ	Uhrzeit	IP Adresse	Absender (Envelope)	Ziel
	03:08:57	74.78.251.129	sladimirrodionov@simplexgroup.ru	[redacted]
	03:09:08	189.105.214.213	reparationsr9@newyorkbase.com	[redacted]
	03:09:08	189.105.214.213	reparationsr9@newyorkbase.com	[redacted]
	03:09:08	189.105.214.213	reparationsr9@newyorkbase.com	[redacted]
	03:09:27	211.136.179.66	valdemar@north.ru	[redacted]
	03:10:03	190.71.91.198	hungereda9@bcsh[redacted]yardworkers.com	[redacted]

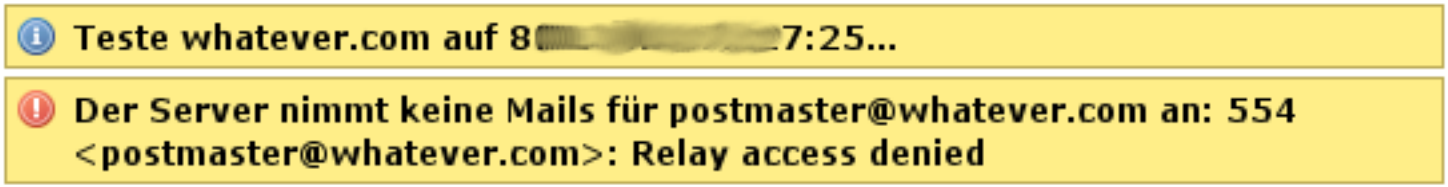
Inhalt: Nicht untersucht
Aktion: Abgewiesen
Details: 504 5.5.2 <MZYCXZBZNZ>: Helo command rejected: need fully-qualified hostname
IP-Adresse: 211.136.179.66
Land: China (CN)
Eingangs-Server: mx2.rolmail.net
HELO: MZYCXZBZNZ

Screenshots ROL Mail Center: Domain anlegen

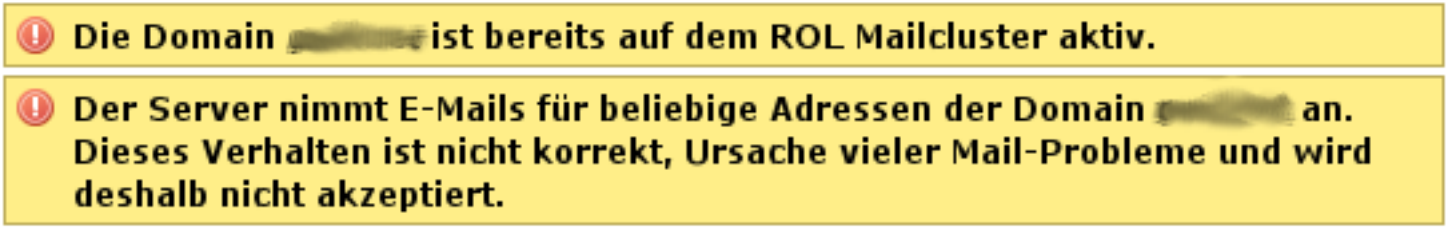


Screenshots ROL Mail Center: weitere Tests

- Postmaster muss existieren:



- Empfänger-Verifizierung muss aktiviert sein:



DB-Replikation 1: Konfigurations-Datenbank

- Master-Master, hochverfügbar
- Zusätzlich alle paar Stunden: SQL-Dump.
- Das sind wenige, aber lebenswichtige Daten!
- Wenn mindestens eine Instanz läuft, kann Postfix Mails annehmen

DB-Replikation 2: Amavis-Datenbank

- Transaktionssichere normalisierte Datenbank
- Genutzt für: Penpals, Betreff/Subject, Quarantäne
- 70-100GB und mehr
- Sehr Ressourcen-intensiv: Garbage Collection
- Deshalb: Partitionierung

DB-Partitionen mit Amavis - Konfiguration

- Kern-Element: Partition-Tag
- Beispiel mit wöchentlich neuer Partition:

```
$sql_partition_tag = sub {
    my($msginfo) = @_;
    iso8601_week($msginfo->rx_time);
};
```

- Wichtig auch für amavisd-release und Log-Parser!

Partitionierte Tabellen mit MySQL 5.1

```
CREATE TABLE maddr (
  id int(10) unsigned NOT NULL AUTO_INCREMENT,
  partition_tag int(11) NOT NULL DEFAULT '0',
  email varbinary(255) NOT NULL,
  domain varchar(255) NOT NULL,
  PRIMARY KEY (id, partition_tag),
  UNIQUE KEY email_key (partition_tag, email)
) ENGINE=InnoDB PARTITION BY LIST (partition_tag) (
  PARTITION p0 VALUES IN (0,1,2,3,4,5),
 …
  PARTITION p8 VALUES IN (48,49,50,51,52,53));
```

- Garbage Collection? Partition DROPpen!

DB-Replikation 3: Maillog

- "Normalization is for Sissies"
- Flache Tabellen, manuell partitioniert
- Daten teilweise redundant
- Keine Funktionen in Suchausdrücken verwenden
- Nicht auf Query-Parser /-Optimizer verlassen
- Applikation formuliert Abfrage je nach Typ anders

Postfix & DB: proxy_read_maps nutzen

- Zwar werden Abfragen dadurch leicht gebremst
- Dafür muss seltener die Verbindung neu aufgebaut werden
- Und, viel wichtiger: es werden weniger parallele DB-Verbindungen benötigt!
- Konfiguration:

```
transport_maps = proxy:mysql:/etc/postfix/transport.cf
address_verify_transport_maps = proxy:mysql:v....port.cf
...
```

Postfix & DB: Datenbank-Failover und -Failback

- Um sicherzustellen, dass Postfix im Ausfall einer DB klaglos weiterläuft, konfiguriert man zwei oder mehr:

```
hosts = db1.mydomain.tld db2.mydomain.tld
user = mx1
password = pass
dbname = mail
...
```

Amavis-Konfiguration für MySQL

- Automatisches Failover (und Failback):

```
@lookup_sql_dsn = (
['DBI:mysql:database=mail;host=db1;port=3306', 'user', 'pass'],
['DBI:mysql:database=mail;host=db2;port=3306', 'user', 'pass']);
```

- Lookup- und Storage-DB getrennt:

```
@lookup_sql_dsn = ( ... );
@storage_sql_dsn = ( ... );
```

Noch ein paar Tipps zu MySQL (I)

- Die MySQL-Konfiguration der Standard-Distros ist bestenfalls für eigene CD-Sammlung brauchbar
- Mit wenigen Schaltern erreicht man sehr viel:
- Für MyISAM:

```
key_buffer = 256M
```

- Für InnoDB:

```
innodb_file_per_table
innodb_buffer_pool_size = 1G
```

Noch ein paar Tipps zu MySQL (II)

- Auch wenn nur ein einziger Server aktiv ist, macht ein MySQL-Slave absolut Sinn (zwei MySQL-Instanzen)
- Grund: Lockings!
- Ein Benutzer, der nervös in seiner Oberfläche rumklickt, könnte sonst z.B. das System böse ausbremsen
- Ein MySQL-User pro Instanz eines Dienstes

Der erste Abschnitt wäre geschafft...

- ...und es geht sofort weiter mit:
  - Redundanz für wirklich alle Dienste
  - Load-Sharing
  - Logging-Strategien und...
  - ...zentrale Aggregation des Logs aller Komponenten

Lastverteilung SMTP-Eingang

- Nichts einfacher als das:
  - Mindestens zwei gleichwertige MX-Einträge
- Wichtig, vor allem am Anfang:
  - Last und Anzahl Prozesse überwachen
  - Log-Einträge überwachen
  - Anzahl Postfix-Prozessen eventuell anpassen:

```
smtp  inet  n  -  -  -  500  smtpd
```

- Nützliches Tool: pflogsum

Lastverteilung SMTP-Ausgang

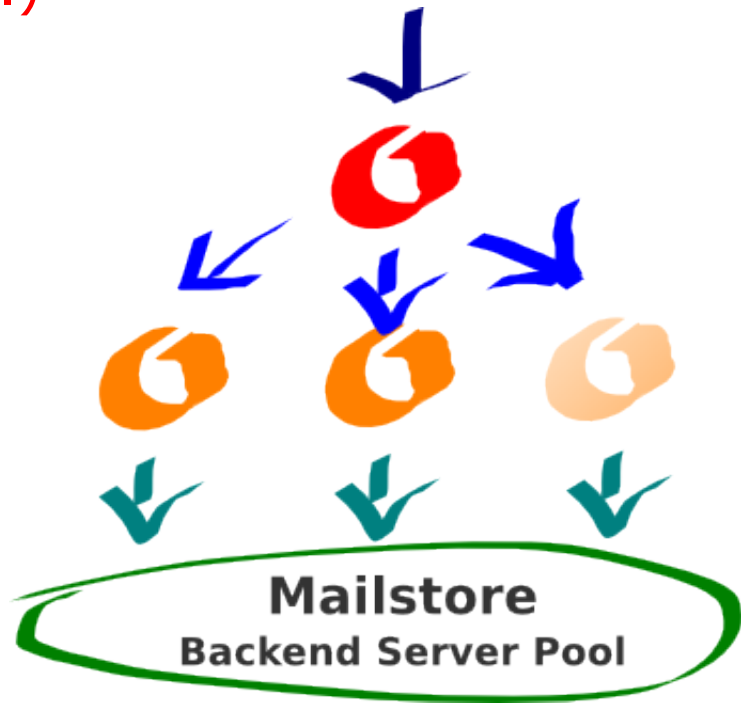
- Schwieriger, da vom Protokoll nicht vorgesehen
- Load-Balancer entweder teuer oder neuer SPoF
- Standby würde ja prinzipiell auch reichen, aber...
- ...mit ClusterIP und Multicast-MAC-Adressen erhält man ein Loadsharing „für arme Leute“
- Keine zusätzliche Hardware, kein SPoF
- Wichtig: Switches und Router müssen mitspielen

Lastverteilung für IMAP und POP3

- Redundanz und Lastverteilung nicht vorgesehen
- Trennung Daten-Backend und IMAP-Frontend:
  - Einsatz von IMAP- und POP3-Proxys
- Wir haben uns für zwei Proxys (samt Load-Sharing mit ClusterIP) entschieden, obwohl:
  - es damals bloß ein Daten-Backend gab
  - der IMAP-Server das alleine geschafft hätte

IMAP und POP3 – Redundanz und Failover

- Nach außen: virtuelle IP und Multicast-MAC
  - imap.mydomain.tld (+ pop3.)
- Mehrere Proxy-Server:
  - imap1.mydomain.tld
  - imap2.mydomain.tld
  - *imapX.mydomain.tld*
- Mailstore: mehrere Backend-Server, Verteilung I/O-Last



Weitere Vorteile von IMAP- und POP3-Proxy

- Verlagerung der Logik für die Lastverteilung auf die Anwendungsebene
- Bedeutet mehr Aufwand, aber auch Flexibilität
- Herstellerunabhängigkeit
- Man wird auch flexibler in Storage-Entscheidungen
- Pay as you grow!

Was sind eigentlich Multicast-MAC-Adressen?

- Ein an eine Multicast-MAC gerichtetes Paket muss von einem Switch an ALLE Ports gesendet werden
- Damit dies nicht zu einem Sicherheitsproblem wird: statische Einträge in die CAM-Tabelle
- Einigen Routern muss man Routing von Paketen von solchen MAC-Adressen explizit erlauben

Konfigurationsbeispiel Cisco-Switch

- Statischer CAM-Table-Eintrag:

```
mac-address-table static 0300.5e11.1111 vlan 200
   interface gi0/11 gi0/12
```

- Pakete für diese MAC-Adresse werden von da an nur noch über die angegebenen Ports rausgesendet

Konfigurationsbeispiel Cisco-Router

- Da der Router keine ARP-Replies von Multicast-MAC-Adressen für Unicast-IPs akzeptiert, ist die einfachste Lösung ein statischer ARP-Eintrag:

  ```
  arp 172.16.1.2 0300.5e11.1111
  ```

- Ohne einen solchen wird die entsprechende IP nur in derselben Broadcast-Domain erreichbar sein

Na schön – aber wie funktioniert das?

- Pakete erreichen jetzt mehrere NICs
- Kernel-Modul ClusterIP muss wissen, welche MAC von wie vielen ClusterIP-Instanzen mit welchem Hash-Algorithmus genutzt wird
- Keine Kommunikation zwischen den Knoten nötig
- Einfaches Beispiel, um sich das vorzustellen:
  - Rechner A beantwortet TCP-Sitzungen für gerade IP-Adressen, Rechner B solche für ungerade IPs

Konfigurationsbeispiel ClusterIP

- Aktivieren von ClusterIP:

```
iptables -I INPUT -d <IP> -i eth0 -p icmp -j CLUSTERIP \
--new --clustermac 01:02:03:04:05:06 --total-nodes 2 \
--local-node 1 --hashmode sourceip
ip address add <IP> dev eth0
echo "+1" > /proc/net/ipt_CLUSTERIP/<IP>
echo "-1" > /proc/net/ipt_CLUSTERIP/<IP>
```

- Auch andere Hash-Werte als 1 sowie auch mehrere Hash-Werte sind möglich

Wie passiert das Failover?

- Linux-HA: heartbeat, heartbeat2, pacemaker
- Failover:

```
ip address delete $IP/24 dev eth0
echo "-1" > /proc/net/ipt_CLUSTERIP/$IP
ssh <othernode> "echo '+1' > /proc/net/ipt_CLUSTERIP/$IP"
```

- Failback:

```
ssh <othernode> "echo '-1' > /proc/net/ipt_CLUSTERIP/$IP"
ip address add $IP/24 dev eth0
echo "+1" > /proc/net/ipt_CLUSTERIP/$IP
```

Logging

- Syslog schont Ressourcen und skaliert
- Wer will kann TCP einsetzen und Logs signieren
- Syslog kann an andere Syslog-Server weitergeleitet werden
- Mit Syslog-Proxies lässt sich so was dann ziemlich gut skalieren
- rsyslog / syslog-ng

Konfiguration des Syslog-Idents

- amavisd.conf:

```
$syslog_ident = 'mf3';
```

- postfix/main.cf:

```
syslog_name = mx2
```

- Ident ist flexibler als der Hostname
- chained hostnames usw. ist dann kein Thema
- Wichtig auch für policy-banks von Amavis oder bei mehreren Postfix-Instanzen pro Host

Wie wird die Log-Datenbank gefüllt?

- Zentraler Syslog-Daemon schreibt in Pipe
- Zentraler Log-Aggregations-Dienst liest aus Pipe
- Sammelt Log-Zeilen, bis alle zu einer Mail gehörigen Informationen beisammen sind
- Schreibt eine Zeile pro Mail und Empfänger in die eingangs erwähnten flachen Tabellen

Klingt jetzt zwar ganz einfach...

- ...man bedenke aber:
  - Dass evtl. nicht alle SMTP-Abbrüche erkannt werden
  - Events in falscher Reihenfolge ankommen können
  - Zeilen verloren gehen können
  - Log-Formate sich zwischen Releases ändern können
  - Sehr lange Zeiten (und viele viele Log-Zeilen) zwischen connect und disconnect liegen können

Log-Zugriff - Sortierung mal andersrum

RaiffeisenOnline

ROL ResellerCenter

Benutzer: [redacted] abmelden

ROL MailCenter

- Domains
- **Quarantäne**
- Maillog eingehend
- Reports

Verwaltung

Quarantäne (sortiert nach Wahrscheinlichkeit)

Zeige 15 Mails vom 26.03.2009 im Zeitraum ganzer Tag ,
dabei aber nur Mails von [redacted] an [redacted] @ [redacted]

« Zurück | 1 | 2 | ... | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | ... | 468 | 469 | Vorwärts »

Typ	Uhrzeit	IP Adresse	Absender (Envelope)	Ziel	Bewertung
	10:03:44	213.252.95.196		vi[redacted]t	96.56 %
	14:17:50	82.106.27.207	gi[redacted]@gmail.com	ai[redacted]t	96.57 %
	11:14:43	82.106.27.207	gi[redacted]@gmail.com	[redacted]t	96.60 %
	03:17:04	66.135.37.149	k[redacted]@pipcom.com	m[redacted]t	96.61 %
	13:05:31	209.85.220.157	mar[redacted]@liveinternet.ru	vi[redacted]t	96.61 %
	12:03:19	65.55.116.24	v[redacted]@msn.com	u[redacted]t	96.61 %
	04:17:16	217.19[redacted]2	payment@hotmail.com	[redacted]t	96.70 %
	07:46:43	145.253.2.46		m[redacted]t	96.70 %
	16:47:17	212.112.231.76	l[redacted]728@163169.net	g[redacted]t	96.70 %

Inhalt: Als SPAM eingestuft
Aktion: In Quarantäne gestellt
IP-Adresse: 217.19[redacted]
Land: Italy (IT)
Größe: 3.122 Bytes
Eingangs-Server: mx2.rolmail.net
WarteschlangenID: 6014210701F5
Gesamtdauer: 5.10s
Einlieferungsdauer: 2.50s
Scandauer: 2.60s
PTR: hp1.alt[redacted]

Einzelnes Mail im Detail

- Transparenz gegenüber dem Kunden:

Live-Zugriff VS Archivierung

- Sind voneinander unabhängige Vorgänge
- Log wird dreimal erfasst:
  - Lokal für fünf Tage (Plan B)
  - In der Datenbank für 14-30 Tage
  - Verschlüsselt und archiviert (gesetzliche Vorgaben)
- Datenbank wird damit unkritisch

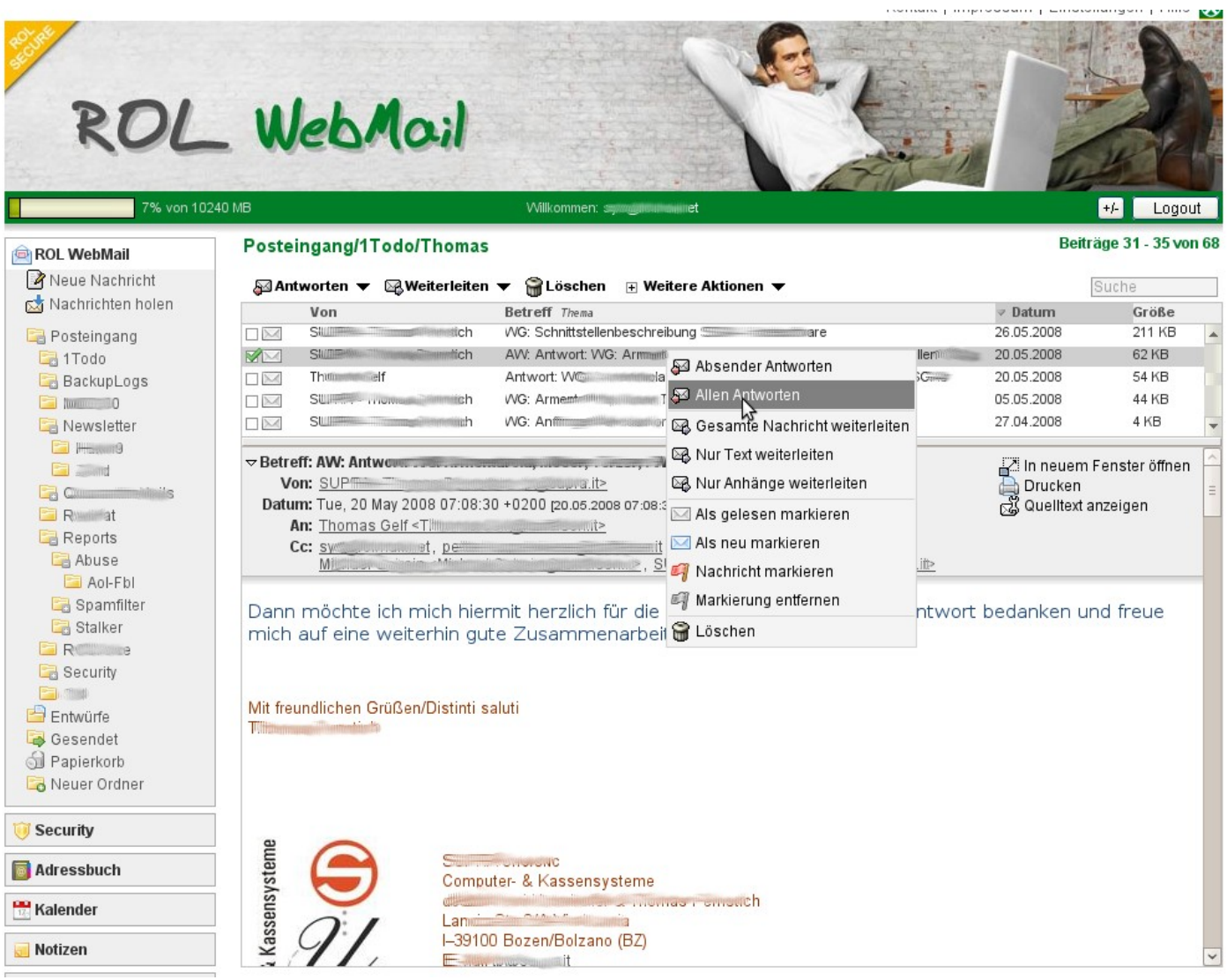
Zum Abschluss...

- ...noch ein paar Screenshots:
  - Webmail
  - Interne Tools
- Als Denkanregung und Inspiration

Der OSS-Mailcluster von Raiffeisen Onine

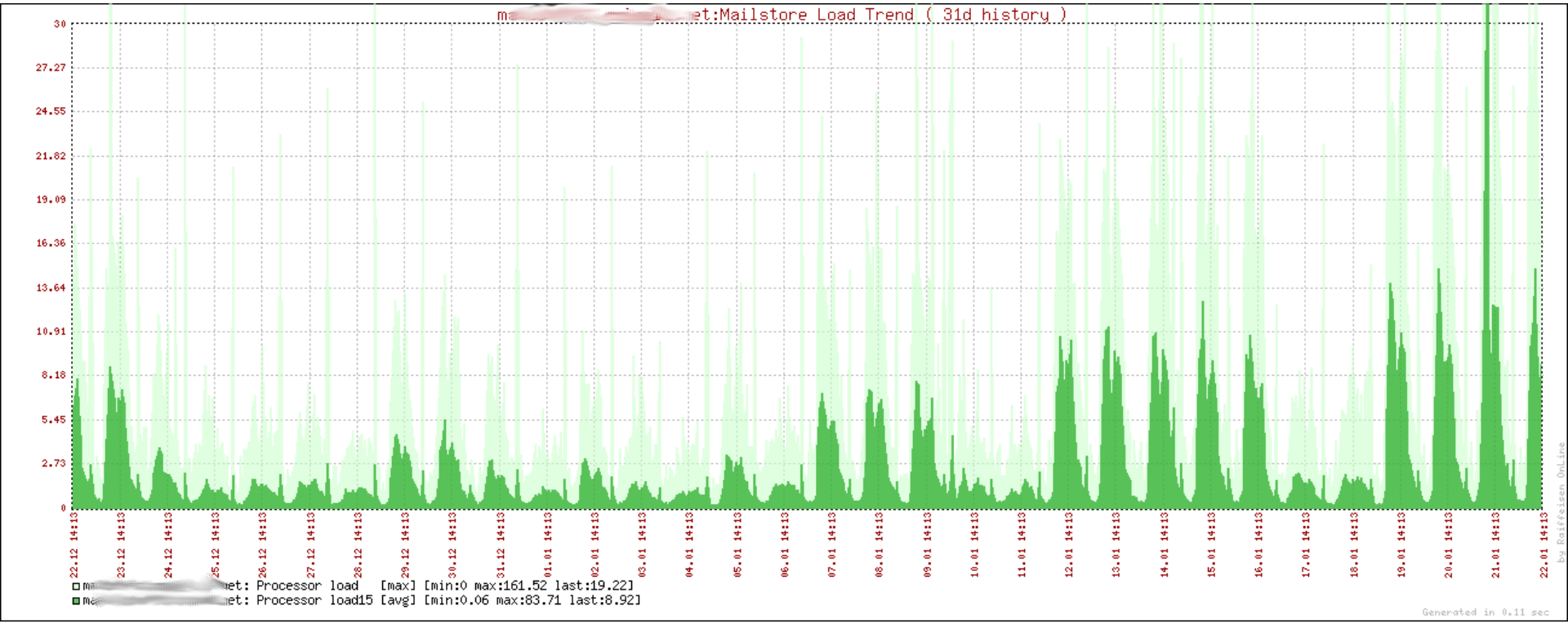
Webmail

- Horde/IMP
- Angepasst
- Erweitert
- Memcache
- Reverse-Proxy



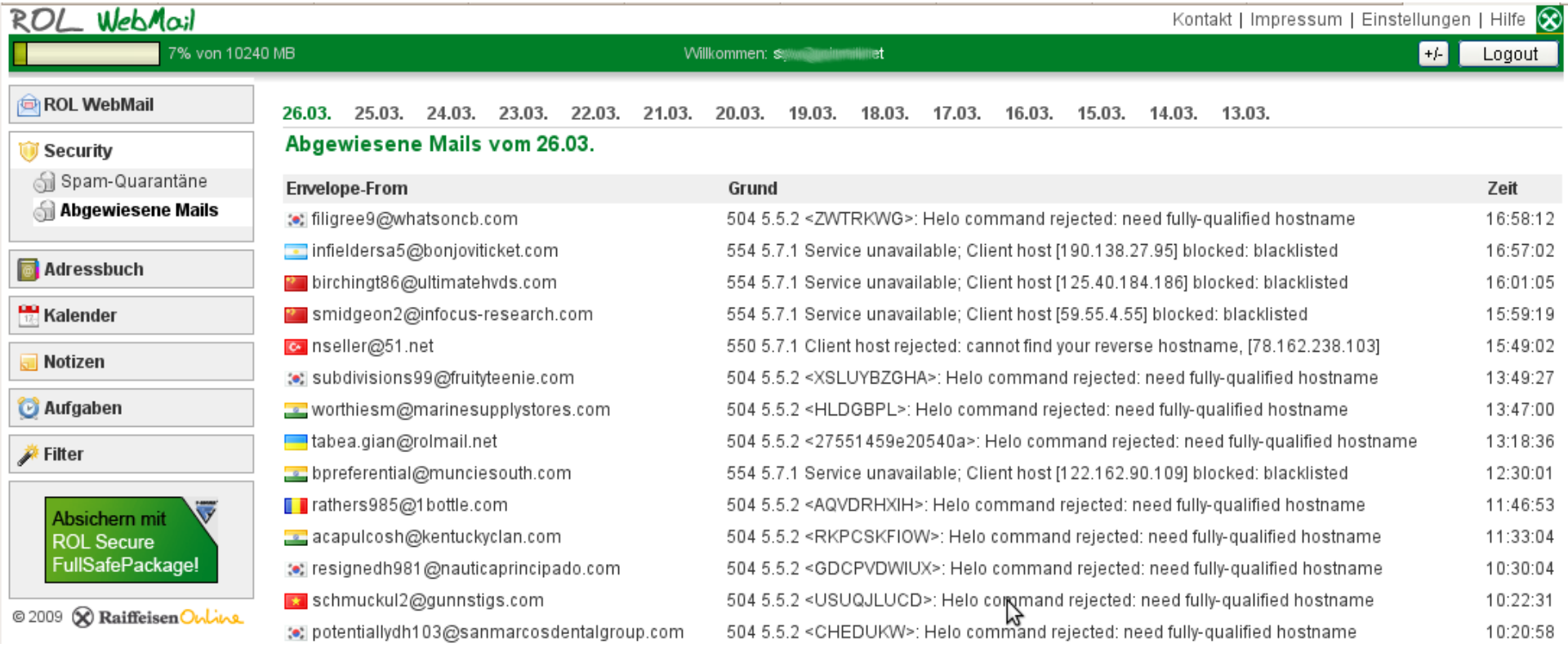
Load-Anstieg durch neues Webmail

- Mehr IMAP-Aktivität, mehr I/O Last:



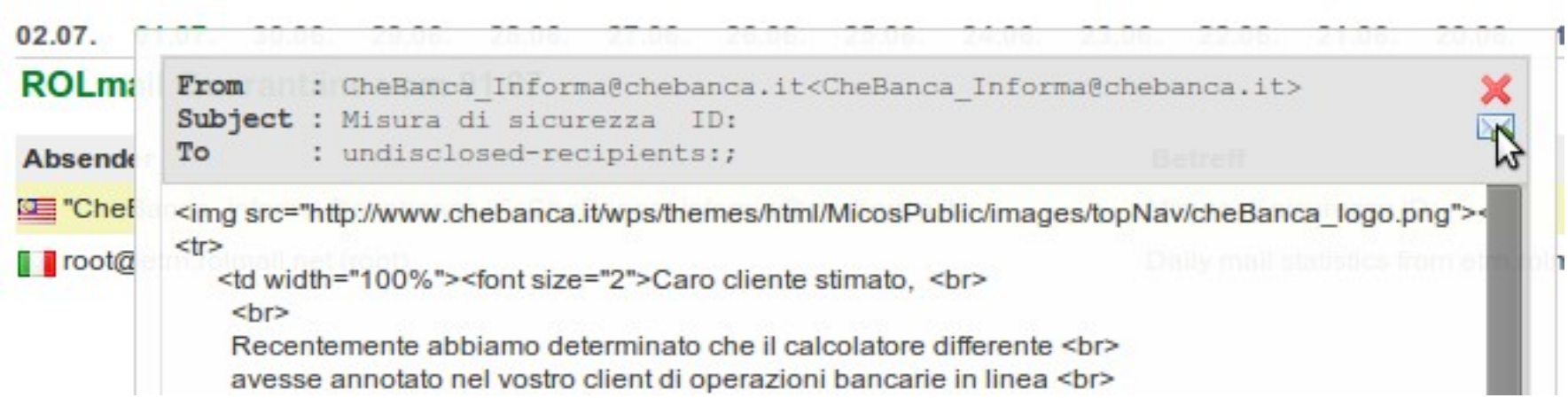
Log-Zugriff im Webmail

- Abgewiesene Mails:

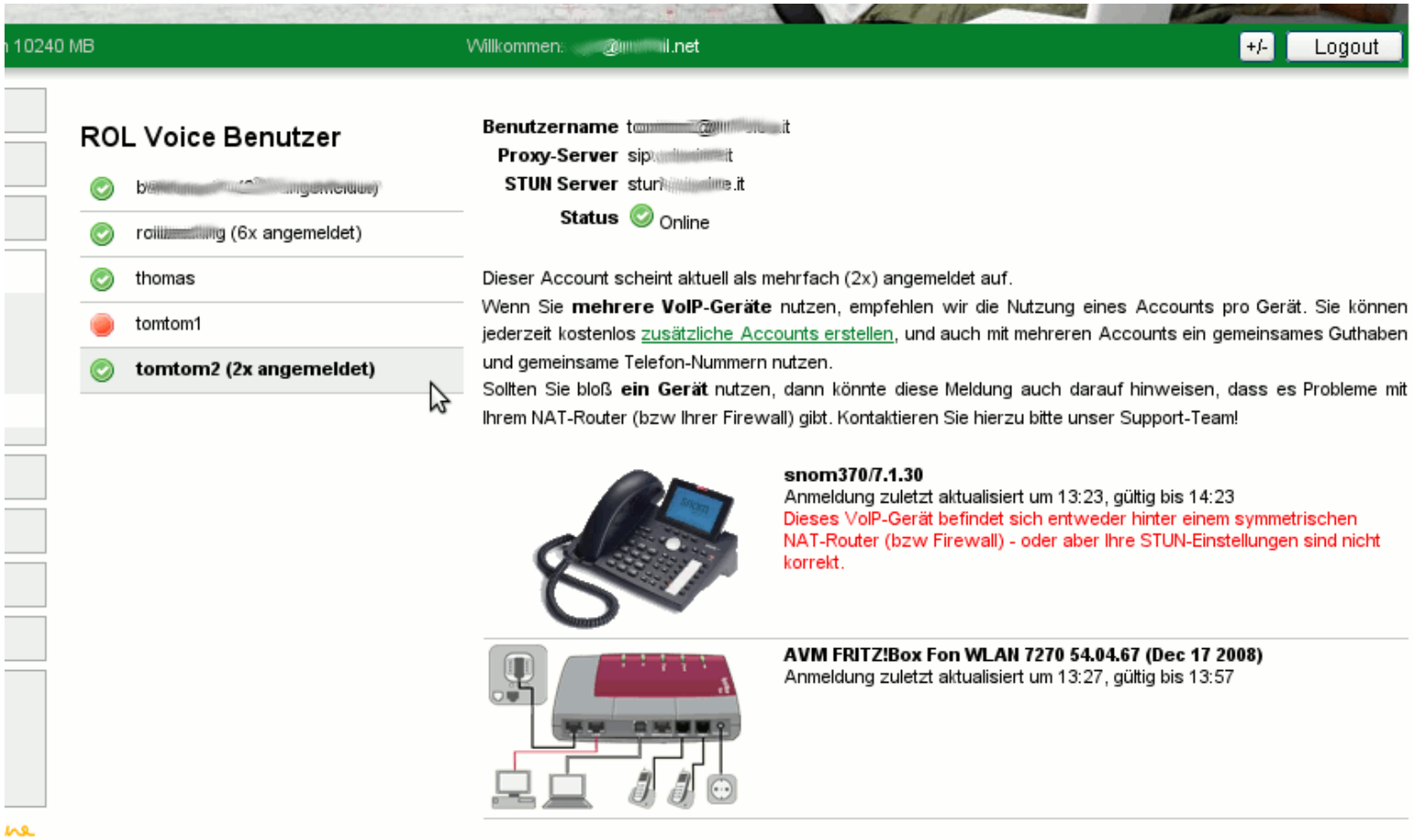


Zugriff auf Quarantäne

- Mails werden im Text-Modus angezeigt
- Kunde kann sich Mails selbst zustellen



Aktuell: Integration mit Telefonie



Die Nanny: Zustand von Amavis

Die Nanny ist ein gutes Werkzeug.

Wer sich die Mühe machen will, kann den Output parsen und in eine Web-oberfläche einbauen


mf3.rolmail.net mf4.rolmail.net

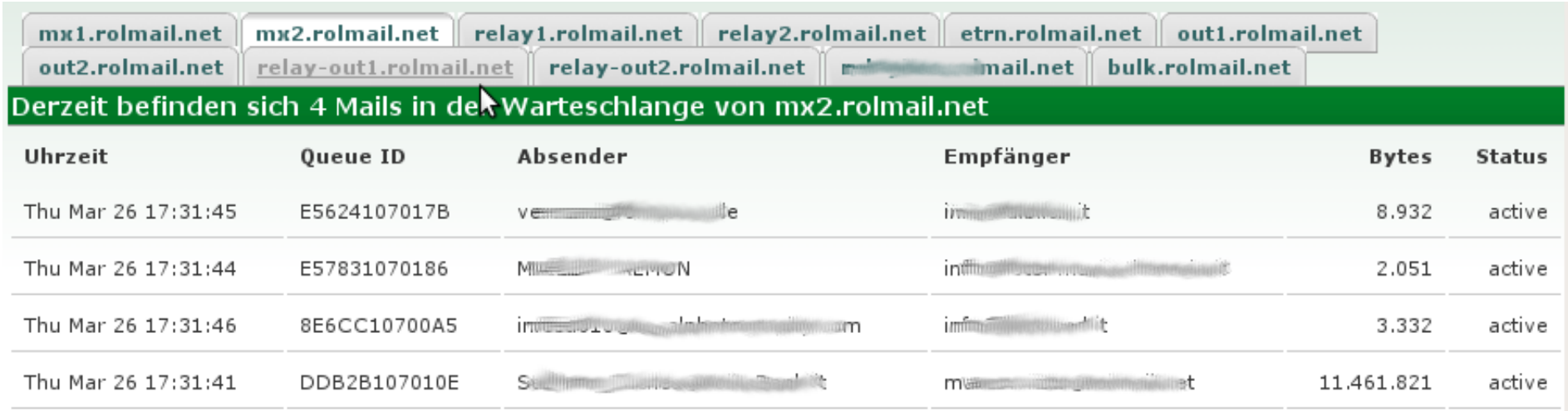
There are 39 filter processes running on mf4.rolmail.net

PID	INFO	Time	Progress
14830		0:00:47	.....:.....:.....:.....
14884		0:04:21	.....:.....:.....:.....
15105		0:01:09	.....:.....:.....:.....
15420		0:00:35	.....:.....:.....:.....
15426		0:00:52	.....:.....:.....:.....
15429	15429-10	0:00:00	s
15430		0:00:13	.....:...
15444		0:00:34	.....:.....:.....:....
15502		0:06:52	.....:.....:.....:.....
15548	15548-09-2	0:00:01	s
15609		0:00:48	.....:.....:.....:.....
15760		0:00:39	.....:.....:.....:.....
15846	15846-09-2	0:00:01	=s

Neu in Amavis: SNMP

- Amavis enthält mittlerweile einen SNMP-Agent
- Letzte Releases (aktuell: 2.6.4) bieten eine umfassende MIB
- Für Überwachung und Monitoring definitiv zu bevorzugen
- Nettes Feature von Nanny: kümmert sich nebenbei um Prozess-Leichen

Immer die Queue im Auge: mailq



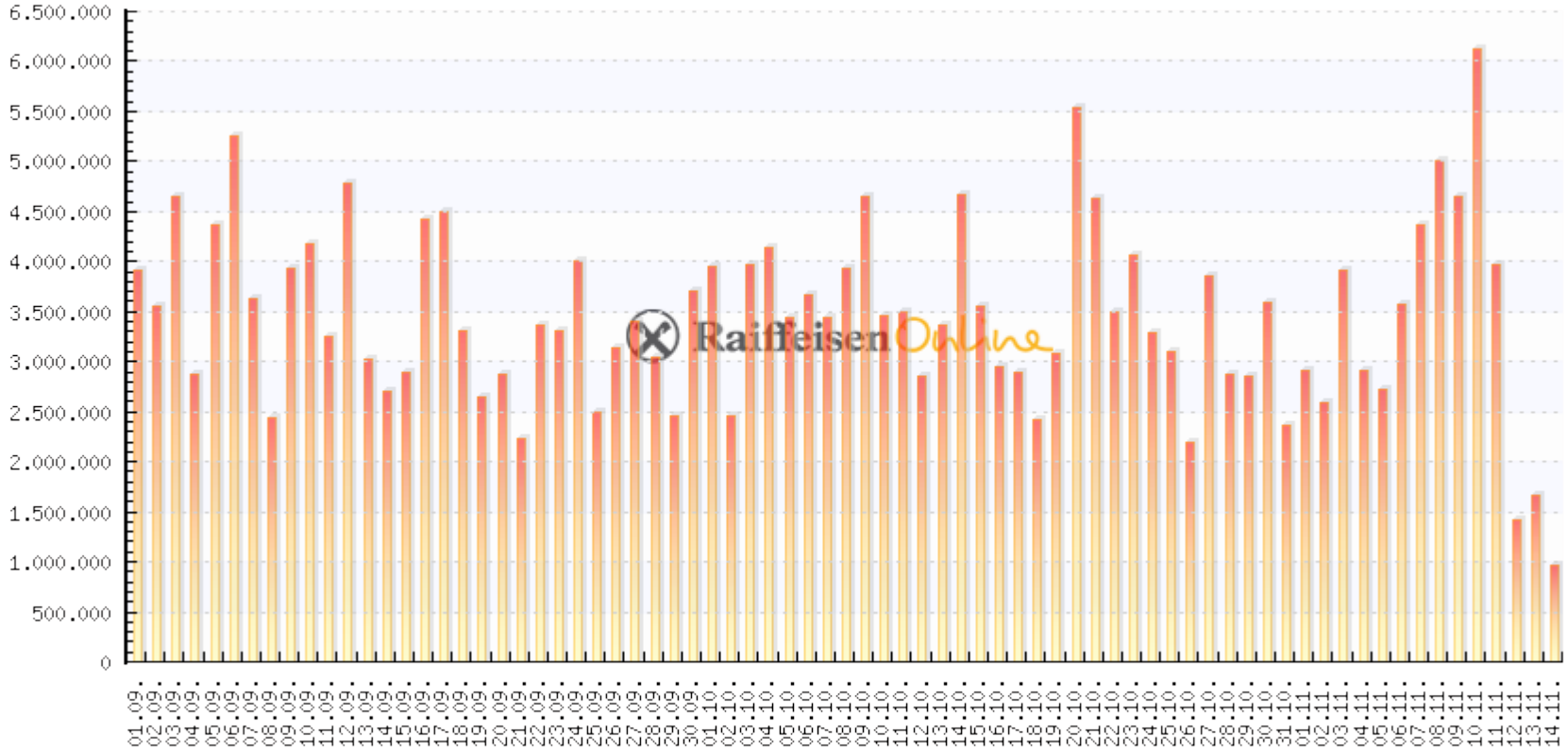
- Kleiner XML-RPC Wrapper
- Vorsicht bei Systemen mit einem "Grundrauschen" von mehreren 1000 Mails!

Nicht nur der Ausgang ist wichtig!

- Wer hat wann von wo seine Mails abgeholt?

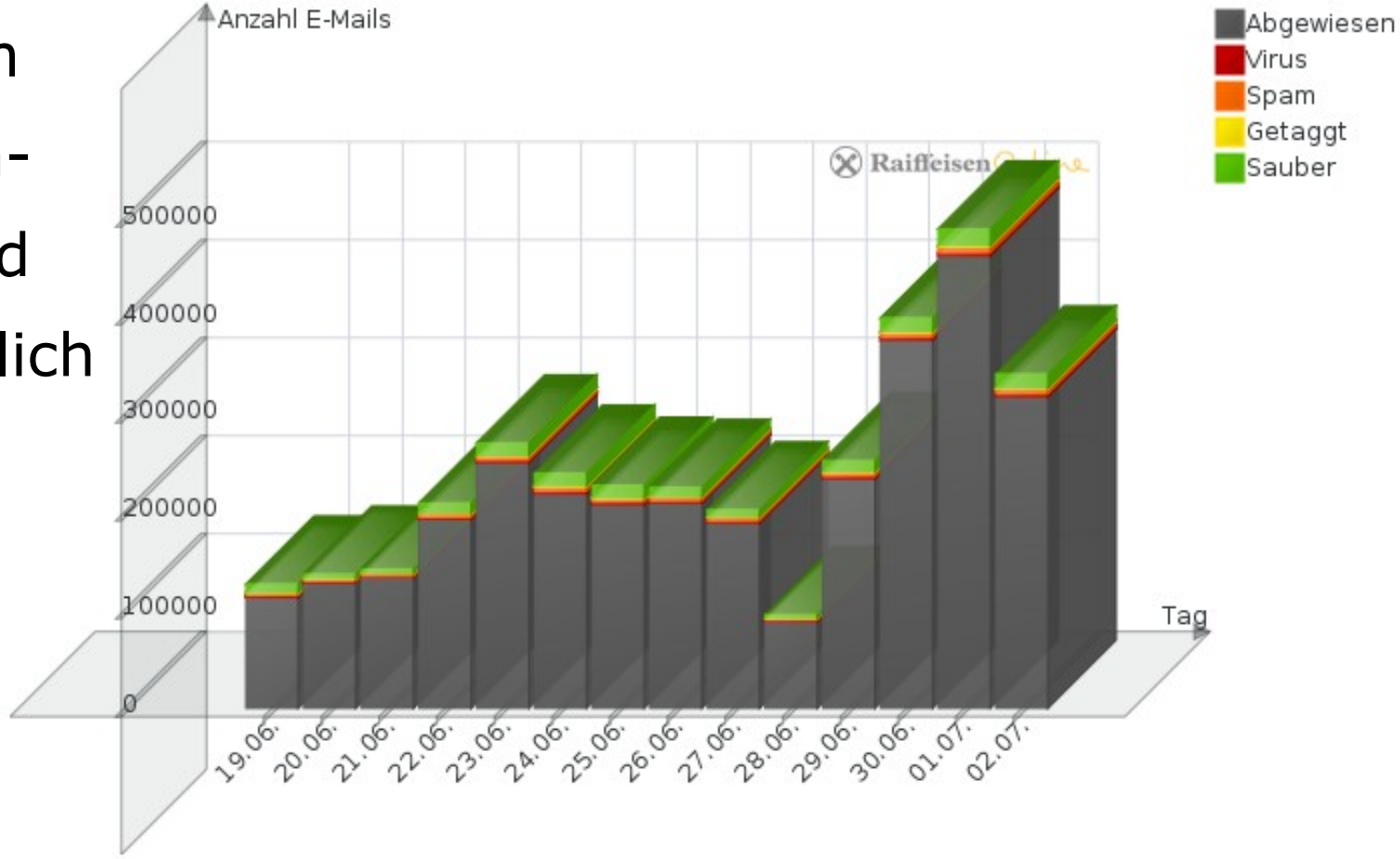
Uhrzeit	Details	Server [PID]
00:00:00	New connection from **195.25**[redacted].	srv1 [16749]
00:00:00	Login for **monitoring@**[redacted] succeeded from **195.254**[redacted], IMAP to rolmail.net	srv1 [16749]
00:00:00	New connection from **195.25**[redacted].	srv2 [18250]
00:00:00	Login for **a**[redacted] succeeded from **195.25**[redacted], POP3 to rolmail.net	srv2 [18250]
00:00:00	**195.25**[redacted] logged out. (User: **a**[redacted]) Bytes up / down: 12 / 60	srv2 [18250]
00:00:00	Login for **f**[redacted] succeeded from **21**[redacted], IMAP to rolmail.net	srv2 [8971]
00:00:03	Login for [redacted] succeeded from [redacted], IMAP to rolmail.net	srv1 [15871]
00:00:03	New connection from [redacted].	srv1 [16763]
00:00:03	Login for [redacted] succeeded from [redacted], IMAP to rolmail.net	srv1 [16763]
00:00:04	Login for **h**[redacted] succeeded from **2**[redacted], IMAP to rolmail.net	srv1 [18192]
00:00:05	**19**[redacted] logged out. (User: **moni**[redacted]) Bytes up / down: 2.178 / 107.848	srv1 [16749]
00:00:06	[redacted] logged out. (User: [redacted]) Bytes up / down: 2.341 / 108.433	srv1 [16763]

Aktuelles "Klima" am Beispiel McColo



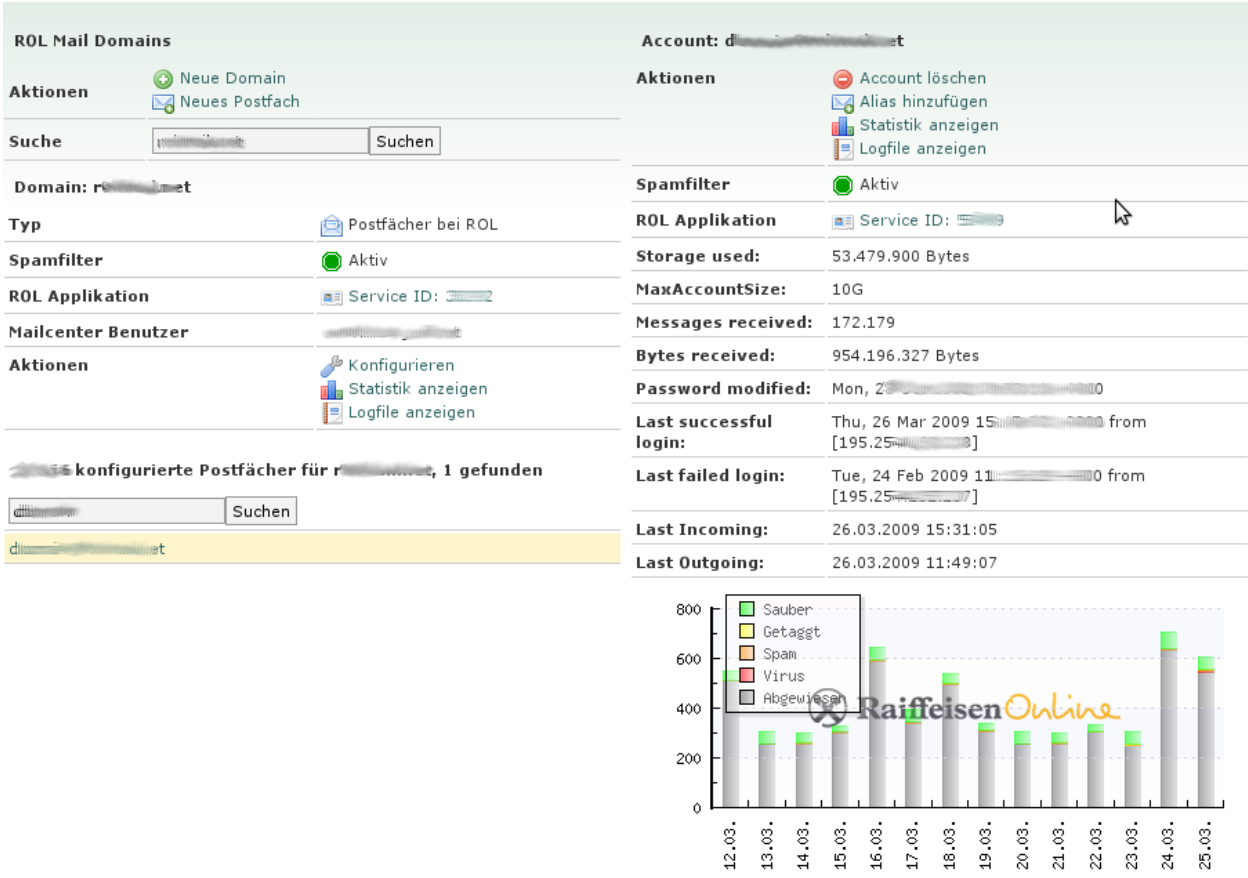
Domainspezifische Reports

- Auch im Kunden-Backend zugänglich



Zentrales Werkzeug für Helpdesk

- Steuert alle Komponenten des Clusters
- Konsolidierung von Web-Applikationen
- Einfachere Migrationen



Noch eine letzte Empfehlung für Entwickler

- Abstraktions-Schicht zwischen Frontend-Applikationen und Mailcluster ist sehr wichtig
- Vereinfacht Evolution und Migrationen
- Einfaches Beispiel:

```
Mailcluster::getInstance()->createDomain('whatever.tld');
```

- Frontend-Programmierer muss wenig über Vorgänge im Hintergrund wissen

Vielen Dank für Ihre Aufmerksamkeit!

Noch Fragen?