Konzept neue Plattform v2

Dokumentenname	PFM_Konzept_Basics_v2
Autor	Thomas Bader
Status	Erste Überarbeitung, bereit zur Diskussion
Letzte Änderung	05.11.2015

Übersicht

Die Dienste werden mit virtualisierten Maschinen bereitgestellt. Dazu werden zwei physische Server als Cluster konfiguriert.

Die Konfiguration der einzelnen Services bzw. virtuellen Maschinen wird gesondert dokumentiert. Das vorliegende Dokument dient der Konzeptionierung und einer Übersicht über die Infrastruktur der Virtualisierung.

Basics

Das vorliegende Konzept zielt darauf ab, möglichst wenig Komplexität aufzuweisen, aber dennoch einen hohen Standard an Sicherheit zu ermöglichen.

Es verfolgt den traditionellen Ansatz, bei dem:

- Redundanz über round-robin im DNS erreicht wird
- Daten via NFS verteilt werden
- Replikation innerhalb von Applikationen stattfindet

Generelles Hardening

Services werden nach der Empfehlung von „Applied Crypto Hardening“ (https://bettercrypto.org/static/applied-crypto-hardening.pdf) konfiguriert.

Für kritische Webservices (z.B. der LDAP Account Manager) wird in Apache per mod_geoip der Zugriff zusätzlich zu der Authentifizierung auch gemäss der GeoIP-Datenbank verifiziert.

Systemadministration

Generelles

Auf den Servern wird /etc als git-Repository (Sourcecode-Verwaltung) eingerichtet. Das Tool etckeeper (http://etckeeper.branchable.com/) wird zur Verwaltung von diesem genutzt. Die Administratoren müssen bei Änderungen einen Commit mit einer Anmerkung ausführen. Es wird durch git möglich einzelne Revisionen von Konfigurationsdateien zu vergleichen sowie Änderungen

nachzuvollziehen.

Statistiken

Typ der Auswertung	Datenquelle	Tool
 Bitte hier eine Liste der benötigten Statistiken bzw. Auswertungen einfügen.	-	-

Klassifizierung von virtuellen Maschinen, Daten und Zonen

Die virtuellen Maschinen werden in drei Klassen eingeteilt, wobei sich diese folgendermassen unterscheiden:

Klasse	Eigenschaften	Ausfallsicherheit	Administration
Core	Grundlegende Infrastruktur. Ein System fällt in diese Klasse, sobald andere Systeme eine Abhängigkeit zu diesem aufweisen (Dienste von trash.net können ohne Core-Services nicht funktionieren)	zwingend - entweder per cold-standby der VM oder durch hot-standby mit Replikation auf Anwendungsebene (Synchronisierung von Daten und Konfiguration zwingend)	zwingend durch techstaff von trash.net
Base	Sämtliche virtuellen Maschinen, welche einen Service von trash.net bereitstellen, auf die andere Systeme von trash.net keine Abhängigkeit haben.	optional - eine Ausfallsicherheit kann bereitgestellt werden, es erfolgt jedoch keine Synchronisierung von Daten und Konfiguration	zwingend durch techstaff von trash.net
Extended	sämtliche virtuellen Maschinen, welche einen Service von trash.net bereitstellen, auf die andere Systeme von trash.net keine Abhängigkeit haben.	optional - eine Ausfallsicherheit kann bereitgestellt werden, es erfolgt jedoch keine Synchronisierung von Daten und Konfiguration	root-Rechte nur für techstaff, Administration von einzelnen Diensten durch Mitglieder möglich
Other	Eine virtuelle Maschine, welche einem Projekt oder einem Mitglied zugeteilt wird	optional - ein Failover muss durch den Betreiber der VM implementiert werden und würde bedingen, dass zwei VM an das Projekt/Mitglied vergeben werden.	Mitglied oder beliebige Drittperson; darf nicht in die grüne Zone verbunden sein

Bei dem Design ist es das Ziel, möglichst wenige VM der Klasse Core zu haben. Diese werden als wirklich kritisch betrachtet, da bei einem Ausfall von ihnen ein sofortiger Failover erfolgen muss. Die VM der Klasse Base sind weniger kritisch. Da andere Services keine Abhängigkeit auf sie haben, muss hier bei einem Ausfall nicht sofort reagiert werden.

Folgende Zonen werden implementiert:

Zone	Eigenschaften	Erlaubte Server-Klassen	Klassifizierung der gespeicherten Daten
Internet (rot)	Systeme mit einer öffentlichen IP-Adresse, auf die direkter Zugriff vom Internet möglich ist.	Core + Base	keine Speicherung von Benutzer/Authentifizierungsinformationen erlaubt
Internet (blau)	Systeme mit einer öffentlichen IP-Adresse, auf die direkter Zugriff vom Internet möglich ist.	Other	n/a
Management (grün)	Systeme, auf die kein direkter Zugriff vom Internet möglich ist.	Core + Base	alle Klassen erlaubt

Virtualisierung

Die Virtualisierung wird mit KVM realisiert.

Die meisten VMs müssen nicht umgezogen werden, da sie paarweise existieren. Einzeln existierende VMs (Messaging, Proxy sowie die Klasse Other) können auf dem Hypervisor in einem via DRBD replizierten Volume abgelegt werden. Im Falle eines Failovers wird, analog zum später besprochenen NFS, auf dem verbleibenden Hypervisor der DRBD-Modus gewechselt sowie die VMs manuell gestartet.

☝ Änderung gegenüber bestehender Plattform: Bisher ist keine Virtualisierung im Einsatz.

Benutzerverwaltung

Die Benutzerverwaltung erfolgt per LDAP. Als Server kommt OpenLDAP zum Einsatz, während die Pflege der LDAP-Objekte mit LAM (https://www.ldap-account-manager.org/) sowie your.trash.net erfolgt. Folgende Funktionen sollen implementiert werden:

- your.trash.net dient in der ersten Phase dazu, die Metainformationen des Benutzers (Adresse, etc.) zu pflegen (bedingt Anpassung des bestehenden PHP-Codes)
- Benutzeraccounts und Gruppen werden in LAM manuell eingerichtet
- Mail-Aliases und virtuelle Mail-Domains werden in LAM manuell gepflegt

Anmerkung: Die Verwaltung von Accounts und Mail-Aliases ist auch per Skript möglich, eine Modifikation der LDAP-Einträge ist mit Perl/Python relativ einfach möglich. Da dies bisher manuell erfolgt, wird eine Automatisierung erst später angestrebt. Es ist möglich, auch von your.trash.net aus die Mail-Aliases und -Domains zu verwalten, sofern die Funktionalität dazu implementiert wird.

Die Administratoren werden neu zwei persönlich Benutzeraccounts erhalten, einen normalen und einen Admin-Account. Mit dem persönlichen, normalen Account können sie wie jedes Mitglied arbeiten. Der SSH-Login auf die Maschinen, welche per Shell nicht für Mitglieder zugänglich sind, muss mit dem administrativen Account erfolgen. Auch die Rechte für sudo hängen an dem persönlichen Admin-Account. Dasselbe gilt für Webinterfaces: Ein Login auf phpMyAdmin oder

Owncloud mit dem normalen Account stellt keine speziellen Privilegien bereit. Erst wenn der persönliche Admin-Account genutzt wird, sind auch die erweiterten Privilegien nutzbar. Auch ein LDAP Account Manager soll nur über den persönlichen Admin-Account zugänglich sein. Durch die Trennung sollen die Administratoren mit ihrem normalen Account exakt nachvollziehen können, welche Usability ein Mitglied hat. Zusätzlich bringt die Trennung einen Schutz bei Kompromittierung eines Accounts (z.B. wenn ein Public-Hotspot durch SSL-Interception eine unprivilegierte Sitzung eines Admins mitschneiden kann).

Ein direkter root-Login auf die Systeme oder *su* mit Passwort ist nicht mehr möglich. Administratoren werden gezwungen, per *sudo* eine privilegierte Shell aufzurufen.

Auf den Systemen ist es zwingend, dass PAM so konfiguriert wird, dass es beide LDAP-Server kennt und von sich aus zwischen beiden wechselt.

Im LDAP Account manager werden folgende Module verwendet:

- Personal (https://www.ldap-account-manager.org/static/doc/manual/ch04s02.html#idp71044480) für Pflege von Adressen und sonstigen Metainformationen
- Unix (https://www.ldap-account-manager.org/static/doc/manual/ch04s02.html#idp71099008) für die Konfiguration der Unix-Parameter auf einem Account (analog /etc/passwd)
- Shadow (https://www.ldap-account-manager.org/static/doc/manual/ch04s02.html#idp71132624) für die Konfiguration der Shadow-Settings (analog /etc/shadow)
- Filesystem quota (https://www.ldap-account-manager.org/static/doc/manual/ch04s02.html#idp71189104)
- Mail aliases (https://www.ldap-account-manager.org/static/doc/manual/ch04s02.html#mailAliasesUser)
- Authorized services (https://www.ldap-account-manager.org/static/doc/manual/ch04s02.html#idp71280176) um selektiv den Zugriff auf IMAP/POP/SMTP/SSH freizuschalten

Für amavisd-new sowie für die Aktivierung von Postgrey hat LAM kein fertiges Modul. Dies muss über den LDAP-Browser (ebenfalls ein Bestandteil von LAM) manuell gemacht werden.

In den ACL von OpenLDAP wird es den Benutzern erlaubt, bestimmte Attribute ihres eigenen Accounts selber zu editieren (z.B. Mailfilterung).

Soll LDAP via TLS verschlüsselt werden?

✘ Die aktuellen Skripte zur Benutzerverwaltung und für den Kassier müssen angepasst bzw. neu geschrieben werden. Die Komplexität nimmt dabei ab, da LDAP in der Handhabung einfacher ist als MySQL. Ein Teil der Skripte kann abgelöst werden, da bspw. Mail-Aliase im LDAP Account Manager hinzugefügt werden können. Es soll angestrebt werden, dass die für trash.net relevanten Attribute für die Benutzerverwaltung ggf. im LAM durch eigene Module implementiert werden (https://www.ldap-account-manager.org/static/doc/devel/mod_index.htm)

☝ Änderung gegenüber bestehender Plattform: Bisher wird keine zentrale Benutzerverwaltung verwendet. LDAP ist nicht nur für den SSH-Zugriff interessant. Es können auch andere Dienstleistungen angedockt werden (z.B. Antispam).

Skripte zur Benutzerverwaltung

Skript	Ersatz
add_alias.pl	
add_mysqldb.pl	
add_user.sh	
add_virtserv.pl	
change_passwd.pl	
change_passwd_nomail.pl	
check_entry.pl	
check_payments.pl	
clean-db.pl	
copy_mailaddr.sh	
correct.pl	
delete_idle_nonmemb.pl	
delete_idle_nonmemb.txt	
delete_virtserv.pl	
importdb.cgi	
mgtest.log	
mod_account.sh	
mv2member.pl	
mysql-create-root	
mysql-reset-root	
nonmemberwerbung.pl	
priate_email.pl	
show_address.pl	

Skript	Ersatz
show_privmail.pl	
truniger.pl	
2delete	
autopatch.pl	
autopatch.sh	
backup-config.sh	
backup2local.sh	
change-perm2trash.sh	
check_cluster	
check_expire	
check_failed_sshd.pl	
check_failed_sshd.pl.bu	
check_failed_sshd.pl.new	
check_failed_sshd.pl.orig	
check_homedir.pl	
check_mailsize.pl	
check_quota.sh	
check_sqlgrey	
check_syslogd	
checkpw	
checkquota.sh	
chng_pws.pl	
clean_tmp.sh	
couriergraph.pl	

Skript	Ersatz
cpu-load.sh	
date.sh	
delete_old_beers.pl	
deletemail.sh	
diskspace.mail	
dist2allusers.sh	
dot	
dspam_cssclean.sh	
dump_mysql	
filecomp.sh	
find-users2delete.pl	
find_obsolete_home	
find_setuid.sh	
fix-modes	
grepmail.sh	
kill_death_procs.sh	
lastlogin	
log_prstat	
mail-mgmt	
mailaccess.pl	
mailaccess.pl.20100101	
mailaccess.pl.20110725	
mailaccess.pl.bu	
mailgraph.pl	

Skript	Ersatz
mailgraph.pl-old	
mailstats.orig	
mailstats.sh	
make_anonftp.sh	
make_mailgraph.sh	
make_mailgraph.sh.20100101	
make_snort.sh	
make_sqlgraph.sh	
makesymlinks.claudio	
makevirtusers.sh	
mbox_warn.ksh	
md5mon.sh	
mmfold.py	
monitor-server.pl	
monitor-traffic.sh	
newdovecotlog	
newmaillog	
newspamdlog	
newsqlgreylog	
ntop	
oldstuff	
reinject	
renice_irc.sh	
restart_tomcat.sh	

Skript	Ersatz
rotatelogs.sh	
rrd-create-base	
rrd-create-base.truniger	
rrd-httpdlog.pl	
rrd-update.pl	
satan-1.1.4exp	
smtpstats	
sqlgreygraph.pl	
sqlgreyparse.pl	
start_ipv6.sh	
statgrab-5.5.php	
statgrab.php	
statgrab.php.orig	
statgrab.php.truniger	
status_check.sh	
stinky-sync.tar	
sync_passwd.sh	
sysstats.sh	
system-integrity.sh	
tailwtmp.pl	
tcpdstats	
totallog.sh	
traffic_archive.pl	
traffic_stats.pl	

Skript	Ersatz
tuning-primer.sh	
update_dns.sh	
update_hosts.allow.sh	
update_tinyroot.sh	
updatedb_wrapper.sh	
user-mgmt	
user-stats.sh	
userwarn.pl	

LDAP-Schema

Es wird eine Private Enterprise Number (PEN, https://en.wikipedia.org/wiki/Private_Enterprise_Number) beantragt um eigene Erweiterungen am LDAP-Schema vorzunehmen. Dazu gehören Attribute um bspw. Angaben für den Kassier zu hinterlegen.

Schema der aktuellen Benutzerdatenbank

NFS

NFS ist als Komponente generell kritisch, da der Ausfall des NFS-Server alle Prozesse blockiert, welche auf Daten in einem NFS-Mount zugreifen wollen. Deshalb ist die Abhängigkeit soweit möglich zu reduzieren. Auf allen Maschinen werden die Home-Verzeichnisse der Administratoren gemounted, damit diese dort ihr Home verfügbar haben. Wenn es sich umgehen lässt, sollen Daemons aber nicht so konfiguriert werden, dass sie Daten von dort beziehen. Dies bedeutet, dass Daten im lokalen Filesystem der VM abgelegt werden (Daemons nutzen dafür typischerweise /var). Ausnahmen davon sind folgende Dienste, welche Daten ab NFS beziehen:

- Die Mailzustellung ist nicht mehr möglich bei Ausfall des NFS-Server. Die Mailserver werden jedoch so konfiguriert, dass sie eingehende Mails in /var spoolen bis NFS wieder verfügbar ist.
- Sämtliche Webseiten sind nicht mehr zugreifbar, sobald NFS nicht mehr verfügbar ist.
- Ein Login auf den Shell-Server ist möglich, jedoch wenig sinnvoll, da die Benutzer nicht über ihr Homeverzeichnis verfügen.

Zur Vereinfachung des Setup wird auf die Bereitstellung von NFS via virtueller Maschine verzichtet. Ein Hypervisor erhält auf seinem grünen Interface eine Service-IP, die als Endpunkt für die NFS-Mounts und RPC verwendet wird.

Es ist zu berücksichtigen, dass world-readable Daten auf einem NFS-Share immer für alle Prozesse auf

dem System lesbar sind. Im Zusammenhang mit Webapplikationen wird generell mit world-readable Daten gearbeitet, damit Apache den Zugriff darauf hat. Es ist daher ungünstig, alle Typen von Daten auf demselben Share zu haben. Nicht jede Art von Daten muss überall verfügbar sein. Deshalb wird die Storage unterteilt:

- */home* für die Homeverzeichnisse der Benutzer, sowie auch für ihre Webdaten
- */srv/web* für eigene Webapplikationen von trash.net (insbesondere Owncloud)
- */usr/home* für die Homeverzeichnisse von Administratoren

Für die Speicherung der Daten wird per DRBD eine aktiv/passiv Replikation eingerichtet. Sofern der aktiv als NFS-Server agierende Server ausfällt, wird auf DRBD manuell der Modus gewechselt und die Service-IP umgezogen.

☝ Änderung gegenüber bestehender Plattform: Bisher wird keine zentrale Storage via NFS betrieben. Über NFS können Probleme mit dem File-Locking auftreten, wenn das Locking im Code nicht sauber implementiert ist. Es ist ratsam, möglichst auf File-Locking zu verzichten (deshalb der Wechsel von mbox auf Maildir).

DB

Da Datenbanken analog zu LDAP funktionieren (Zugriff via TCP, Nutzung einer Abfragesprache) werden sie auf denselben VM wie LDAP betrieben.

Es ist geplant, zu Beginn lediglich MySQL anzubieten. Einer der beiden VM wird primärer Server und erlaubt den Schreibzugriff. Die Replikationen der Daten erfolgt durch das MySQL Binary-Log.

Für die Datenbank erhält die primäre VM eine virtuelle IP plus einen DNS-Eintrag auf diese IP. Mitglieder nutzen diese Adressierung für den Zugriff auf die Datenbank. Im Falle eines Ausfall wird auf dem sekundären Server der Slave-Modus von MySQL beendet und die virtuelle IP umgezogen.

Schreibzugriff auf die Daten ist nur auf dem primären Server möglich. Der Lesezugriff ist hingegen auf beiden Instanzen möglich. Es ist deshalb möglich, dass rein lesende Zugriffe via DNS round-robin auf beide Instanzen verteilt werden können. Es ist vorerst allerdings nicht vorgesehen, dies anzubieten und die sekundäre Instanz nicht zugänglich zu machen.

✘ Die TCP-Verbindungen zu MySQL werden nicht verschlüsselt sein. Deshalb laufen die Verbindungen zur Datenbank über die grüne Zone. Der root auf einem Webserver kann über tcpdump die Passwörter mitlesen. Dies ist aber nicht kritisch. Die Passwörter stehen typischerweise in der Konfiguration der Webapplikation. Damit Apache diese lesen kann, muss diese word-readable sein, weshalb die Passwörter sowieso einfach zugänglich sind.

☝Änderung gegenüber bestehender Plattform: Zugriff auf MySQL wird nicht mehr über den lokalen Unix-Socket möglich sein. Webapplikationen müssen umkonfiguriert werden, so dass sie via TCP auf die Datenbank verbinden.

OpenSSH

Auf jeder VM ist in der sshd_config eine Liste der Gruppen hinterlegt, deren Mitglieder auf das System

einloggen dürfen. Auf den Shellservern dürfen alle Accounts der normalen Mitgliedergruppe einloggen. Für die restlichen Systeme wird je eine Gruppe pro Service angelegt, in denen die persönlichen Admin-Accounts eingetragen sind, und in der sshd_config eingetragen.

Shell

Für den Zugriff auf eine Shell stehen zwei VM zur Verfügung. Der Benutzer kann auswählen, auf welchen er verbinden will. Im Falle eines Ausfalls liegt es am Benutzer, mit seinem SSH-Client auf die jeweils andere VM zu verbinden.

Vom NFS-Server wird /home mounted. Die Benutzer können dadurch auf beiden Shell-Servern das identische Home-Verzeichnis nutzen.

Bei der Administration des Shell-Servers wird darauf geachtet, dass beide über dieselben installierten Softwarepakete verfügen. Abgesehen davon ist keine Synchronisierung der Konfiguration notwendig.

☝ Auf Linux werden einige Tools nicht mehr in denselben Pfaden zu finden sein (z.B. kein /sw mehr).

DNS authoritativ

Wie bis anhin werden die DNS-Zonen manuell auf der Shell gepflegt.

Der bisherige DNS-Setup soll vereinfacht werden: Es gibt dazu noch zwei VMs mit einer bind-Instanz auf dem roten Interface. Eine VM ist Master, die andere beschafft sich Kopien der Zonen per AXFR.

Neu wird standardmässig jede Zone per DNSSEC signiert. Mit Inline Signing (http://securityblog.switch.ch/2014/11/13/dnssec-signing-your-domain-with-bind-inline-signing/) sind keine Skripte oder Tools mehr dazu notwendig, bind kann die Erneuerung von Signaturen selber handhaben.

DNS rekursiv

Rekursiver DNS wird für alle VM benötigt. Dazu wird auf den beiden VM für DNS auf dem grünen Interface eine unbound-Instanz betrieben (https://unbound.net/). DNSSEC-Validierung ist auf dieser aktiviert.

VM der Klasse Other nutzen die rekursiven DNS von Init7, oder beliebige andere.

Web

Für Web werden zwei VM betrieben, die jeweils /home per NFS mounten. Die Benutzer können ihre Seiten vom Shell-Server aus so direkt bearbeiten. Falls die Benutzer die Daten nicht via Shell bearbeiten wollen, so müssen sie SCP/SFTP zu einem der Shell-Server nutzen.

Die beiden Server haben dieselbe Konfiguration von Apache.

Es werden im DNS zwei Aliases für die beiden VM eingetragen; web-prod0.trash.net und web-prod1.trash.net (jeweils ein A-Record auf die IP einer VM). Die DNS-Einträge für Webseiten sind ein CNAME auf einen einzelnen der beiden Einträge. Ziel ist es, die Anzahl der Webseiten gleichmässig auf die beiden Aliase zu verteilen, damit die Last auf beide VMs verteilt wird. Im Falle des Ausfalls einer VM wird die IP von web-prod1 oder web-prod2 gewechselt (womit dann -prod1 und -prod2 auf dieselbe VM zeigen). Der Vorteil liegt darin, dass im Falle eines Ausfalls nur die Hälfte der Seiten betroffen ist (da sie über beide VM verteilt sind).

Die Konfiguration von Apache wird vereinfacht:

- Virtuelle Hosts werden standardmässig über symbolische Links konfiguriert. Apache verfügt über einen Default-Virthost. In diesem werden per RewriteRule die eingehenden Requests auf /home/www/<wert-aus-dem-host-header>/docs umgeleitet. Logs werden nach /home/www/<wert-aus-dem-host-header>/logs geschrieben. Bei beiden handelt es sich um einen symbolischen Link in das Homeverzeichnis des Benutzers. Für einen Standard-Vhost müssen somit nur noch Symlinks erstellt werden und keine Modifikationen an der Apache-Konfiguration mehr vorgenommen werden.
- Komplexere Virthosts werden in einem Include-File konfiguriert. Über ein Skript kann dieses Include-File auf die andere VM synchronisiert werden. Es ist gedacht, die Änderung zuerst auf einem Host vorzunehmen, sie dort zu testen, und anschliessend durch manuellen Aufruf des Skript die Synchronisation vorzunehmen.

Die virtuellen Hosts sollen wenn möglich auf derselben IP-Adresse laufen. Für SSL wird SNI eingeführt.

✘ Es wird weiterhin möglich sein, dass der Code von einem Benutzer (z.B. PHP oder CGI) die world-readable Files von anderen Benutzern lesen kann. Dazu gibt es keine Lösung.

Webapplikationen von trash.net

Im Bereich Web ist es zu bevorzugen, ein eigenes Paar von virtuellen Maschinen zu haben um die Services von trash.net zu betreiben (d.H. es würden 2x 2 Webserver betrieben).

Die Services von trash.net sind:

- your.trash.net
- phpMyAdmin
- LDAP Account Manager
- Webmail
- Owncloud
- ???

Mail

Vorgesehen sind zwei VMs. Folgende Zugriffe sind möglich:

- MUA mit IMAP/POP verbinden auf mail.trash.net. Im DNS verfügt dieser Eintrag neu über zwei A-Records, Clients werden also per round-robin DNS zufällig auf eine der beiden VMs verbunden. Sofern eine VM ausfällt, wird der round-robin DNS-Eintrag manuell angepasst.
- MUA mit SMTP verbinden auf mail.trash.net, allerdings nicht auf Port 25, sondern auf 587

(Submission). Es wird keine SMTP-Authentifizierung mehr auf Port 25 angeboten. MUA können ihre Mails nur noch per Submission anliefern.
- MTA erhalten bei einer Abfrage des MX für eine Domain zwei gesonderte Einträge für beide VM mit gleicher Priorität zurück. Sofern eine der Mail-VM ausgefallen ist, liegt es am fremden MTA, den jeweils anderen Server zu nutzen.
- Webmail-Clients müssen neu via webmail.trash.net verbinden; der DNS-Eintrag zeigt auf einen der Webserver, nicht auf die Mailserver. Das Webmail wird somit wie jede andere Webseite auf den Webservern implementiert.

Mail-Aliase sowie Adressen in virtuellen Domains werden per LDAP abgerufen. Über den LDAP Account Manager können diese Attribute gepflegt werden. Es erfolgt keine manuelle Pflege von Maps mehr dafür. Eine Mailweiterleitung per .forward soll weiterhin unterstützt werden. *Anmerkung:* Dies würde es ermöglichen, dass Benutzer über your.trash.net Mailaliase in ihrer virtuellen Domain selber konfigurieren können (vorerst nicht geplant).

Als Software wird wie bisher Postfix, Dovecot, Spamassassin und procmail verwendet. Der letzte Release von dspam ist aus 2012. Es ist zu prüfen, ob die weitere Nutzung möglich ist. Zusätzlich wird neu amavisd-new (http://www.amavis.org/) angeboten. Es handelt sich dabei um einen Daemon, welcher Spamassassin und Virenscanner integriert kann. Er verfügt über eine Schnittstelle zu LDAP (http://www.amavis.org/LDAP.schema.txt). In den LDAP-Objekten der Benutzer können Thresholds und andere Details konfiguriert werden.

Für den Scan von Mails können die Benutzer zwei Wege gehen. Es kann wie bisher per procmail und direkten Aufruf von Spamassassin gefiltert werden. Alternativ können die Benutzer in der Benutzerdatenbank (LDAP) für ihren Benutzer den amavisd-new aktivieren und eigene Thresholds für die Filterung setzen. Achtung: Der LDAP Account Manager hat kein eigenes Modul für das Amavis-Schema. Eine Verwaltung der Amavis-Settings müsste daher in your.trash.net implementiert werden (ist über simple LDAP-Queries möglich).

Für Greylisting wird weiterhin postgrey verwendet. Neu soll die Aktivierung per LDAP-Attribut auf dem Konto des Benutzers erfolgen. Es gibt dafür kein LDAP-Schema. Deshalb wird eine eigene Schema-Erweiterung vorgenommen. Eine Pflege der Einstellung soll per your.trash.net möglich sein.

☝Bisher wird Greylisting pro einzelner Mail-Adresse aktiviert. In dem neuen Setup wird Greylisting pro Benutzer (d.H. für sämtliche Adressen und Aliases gemeinsam) aktiviert.

Die Mailserver mounten /home per NFS und die Mailboxen der Benutzer liegen darin. Dies bedeutet, dass die Benutzer vom Shell-Server aus den Zugriff auf die Mailboxen haben. Auch können sie dort ihre .procmailrc editieren oder Spamassassin konfigurieren.

Mailboxen müssen neu zwingend Maildir verwenden. Die Verwendung des alten mbox-Formats wird nicht mehr unterstützt (da via NFS kein zuverlässiges File-Locking möglich ist).

Sofern spezielle Settings vorhanden sind (z.B. 2ndary-MX oder spezifische TLS-Policies o.ä.) werden diese in einem Include-File konfiguriert und per Script auf die jeweils andere VM synchronisiert.

Der Empfang von Mails von fremden MTA ist auch bei Ausfall des NFS-Servers möglich. Sofern /home nicht mehr verfügbar ist, werden eingehende Mails in /var spooled und später in die Mailbox des Benutzers zugestellt.

✘ Sofern beide LDAP-Server nicht mehr verfügbar sind, dann kann Postfix keine Mailaliase, amavisd-new oder Postgrey-Konfiguration mehr abrufen. In einem solchen Fall würden Mails mit temporärem Fehler abgewiesen. Das Risiko dafür ist nicht sehr hoch, da die LDAP-Server redundant ausgelegt sind.

Die beiden VM sind für alle anderen VM auch als Relay nutzbar. Sie akzeptieren über das grüne Interface Mails von den anderen VM. Die VM für Typ Other müssen entweder den Mailversand selber implementieren oder sich via SMTP-Auth auf dem roten Interface der Mailserver anmelden.

Angebotene Dienste

Service	Klasse	Zone	Abhängigkeiten	Ausfallsicherheit
LDAP	Core	grün	LDAP, mounted /usr/home per NFS	Es werden zwei VM betrieben (Master und Slave), wobei die Ausfallsicherheit des Lesezugriffs über einen round-robin DNS-Eintrag sowie Replikation innerhalb von OpenLDAP sichergestellt ist. Für den Schreibzugriff wird keine Ausfallsicherheit implementiert. Modifikationen der Benutzerdatenbank sind nur auf dem Master möglich. Die Systeme sind so konfiguriert, dass der Betrieb bei einem Ausfall des NFS-Servers nicht unterbrochen wird (der Mount von /usr/home dient nur dazu, dass Administratoren ihr Homeverzeichnis auf dem Server verfügbar haben).
NFS	Core	grün	LDAP	Wird auf dem Hypervisor betrieben. Replikation der Daten per DRBD.
DB	Core	grün	LDAP, mounted /usr/home per NFS	Wird auf gemeinsam mit LDAP auf denselben VM betrieben. Der Mount von /home dient nur dazu, dass Administratoren ihr Homeverzeichnis auf dem Server verfügbar haben. Die Daten der Datenbanken werden auf Applikationsebene repliziert.
Mail	Base	rot	LDAP, mounted /usr/home und /home per NFS	Es werden zwei VM betrieben, auf die der Load via round-robin DNS verteilt wird. Mailboxen liegen in /home. Sofern ein Mailserver die Verbindung zu NFS verliert, werden die eingehenden Mails in /var spooled. Wenn ein Mailserver komplett ausfällt, wird sein Eintrag aus dem round-robin DNS entfernt
DNS	Base	rot	LDAP,mounted /usr/home	Es werden zwei VM betrieben, die Replikation der DNS-Zonen erfolgt per Zonentransfer. Die Systeme sind so konfiguriert, dass der Betrieb bei einem Ausfall des NFS-Servers nicht unterbrochen wird (der Mount von /usr/home dient nur dazu, dass Administratoren ihr Homeverzeichnis auf dem Server verfügbar haben).
Shell	Base	rot	LDAP, mounted /usr/home, /srv/web und /home per NFS	Es werden zwei VM betrieben mit getrennten DNS-Einträgen. Benutzer können dadurch auswählen, auf welchen Shell-Server sie verbinden wollen. Im Falle eines Ausfalls einer VM erfolgt kein Fail-Over; die Benutzer müssen mit ihrem SSH-Client dann auf die jeweils andere Maschine verbinden

Service	Klasse	Zone	Abhängigkeiten	Ausfallsicherheit
Web	Extended	rot	LDAP, mounted /usr/home und /home per NFS (zusätzlich /srv/web, wenn die eigenen Services von trash.net darauf betrieben werden)	Die Webseiten werden per DNS auf beide VM verteilt. Sofern eine VM ausfällt, kann durch Anpassung im DNS der Fail-Over herbeigeführt werden.
Messaging	Extended	rot	LDAP, mounted /usr/home per NFS	VM ist alleinstehend.
Proxy	Extended	rot	LDAP, mounted /usr/home per NFS	VM ist alleinstehend.

Das gesetzte Ziel, möglichst wenige VM der Klasse Core zu haben, wird somit erreicht. Wirklich kritisch sind nur NFS und LDAP. Hier ist bei einem Ausfall eine schnelle Reaktion notwendig. Bei allen anderen VM ist keine sofortige Intervention notwendig, da diese Services unabhängig voneinander laufen.

Es ist auch zu beachten, dass der Failover nur bei LDAP und NFS komplex btw. automatisiert ist. Bei den anderen Services sind entweder keine Anpassungen oder nur manuelle Änderungen am DNS notwendig. Die Komplexität wird dadurch auf ein Minimum reduziert.

Weitere Dienste

Service	Implementation
Mail-Forwarding	Auf den Mail-VM, basierend auf LDAP-Lookup
Anti-Spam	Auf den Mail-VM, basierend auf Spamassassin, Konfiguration ist userspezifisch in /home
procmail	Auf den Mail-VM, basierend auf Spamassassin, Konfiguration ist userspezifisch in /home
cronjobs	Auf beiden Shell-VM möglich. Aber ohne Fail-Over.
Web mit CGI, Perl, Python, PHP	Auf den Web-VM. Es ist zu entscheiden, ob man ein Paar VM nutzt für alles, oder zwei Paar VM und separiert zwischen Mitgliedern/trash.net Services.
Datenupload mit SFTP	Über die Shell-Server. Daten sind auf den Webservern via /home NFS-Mount verfügbar.
2ndary MX	Auf den Mail-VM.
Proxy	eigene Proxy-VM der Klasse Extended, kein Failover.
Stunnel	eigene Proxy-VM der Klasse Extended, kein Failover.
DNSTunnel	eigene Proxy-VM der Klasse Extended, kein Failover.
Mailinglisten	eigene Messaging-VM der Klasse Extended, kein Failover.
Jabber	eigene Messaging-VM der Klasse Extended, kein Failover.
Owncloud	Auf den Web-VM. Es ist zu entscheiden, ob man ein Paar VM nutzt für alles, oder zwei Paar VM und separiert zwischen Mitgliedern/trash.net Services.
CalDAV/CardDAV	zusammen mit Owncloud
NTP	die beiden DNS-Server können zusätzlich mit NTP ausgestattet werden und im ntp.org Pool publiziert werden

Weitere Dienste können wie folgt verteilt werden:

- Bei web-basierenden Diensten erfolgt die Platzierung auf den Web-VM. Falls eine Aufteilung in zwei Paare von Web-VM umgesetzt wird, dann sollen offizielle trash.net Services auf dem Paar mit den trash.net-Services gehosted werden.
- Telefonie/Conferencing kann auf der Messaging-VM hosted werden
- Tor als Relay oder Bridge kann in einer eigenen VM platziert werden
- Tor für Hidden Services kann auf einer eigenen VM platziert werden und als Proxy zu Shell/Web/Mail agieren

Ressourcenauslastung

Es werden wie oben beschrieben 5 VM paarweise benötigt (allenfalls 6, wenn Web unterteilt wird), plus eine Messaging-VM sowie eine Proxy-VM.

VM	**Auslastung**
LDAP/DB	mittlere Auslastung aufgrund der DB
Mail	hohe Auslastung, primär durch den Spamfilter
DNS	Load ist vernachlässigbar
Web	hohe Auslastung
Proxy	tiefe Auslastung
Messaging	tiefe Auslastung

Es kann davon ausgegangen werden, dass nur die VM von Mail+Web wirklich Load erzeugen.

Deshalb sollte nochmals über den Ansatz von einer Unterteilung in einen stärkeren und schwächeren Hypervisor unterschieden werden. Es hilft vermutlich nicht viel, eine der beiden Maschinen deutlich stärker auszulegen.

Vorgehen bei Ausfällen und Wartung

Service	**Fail-Over**	**Wartung**
LDAP	Automatisch, da beide VMs ihre Daten replizieren. Ausnahme ist der Schreibzugriff. Modifikationen an der Benutzerdatenbank sind in einem Failover-Szenario nicht möglich.	In den lokalen iptables-Regeln wird Zugriff auf LDAP mit einem Reject abgelehnt. LDAP-Clients verbinden dann auf die andere VM und Wartungsarbeiten können durchgeführt werden.
NFS	Manuell, Umzug der virtuellen Service-IP plus Wechsel des DRBD-Modus.	Da kein Clustering aktiv ist, bedingt die Wartung eine Abschaltung des NFS-Services. Dies ist vernachlässigbar, da NFS auf dem Hypervisor läuft und eine Wartung auf dem Hypervisor sowieso Unterbrüche bei den VM zur Folge hat.
DB	Umzug der virtuellen IP auf die sekundäre VM, Deaktivierung des Slave-Modus in MySQL.	Kompletter Shutdown der DB notwendig.
Mail	Manuell durch Anpassung des mail.trash.net Eintrags.	Der mail.trash.net-Eintrag wird angepasst sowie in den lokalen iptables-Regeln der Zugriff auf SMTP rejected. Alle MUA und MTA verbinden dann auf die andere VM.

Service	Fail-Over	Wartung
Authoritative DNS	Automatisch, da beide VMs ihre Daten replizieren.	Kein spezielles Vorgehen.
Shell	Nicht notwendig, da Benutzer jederzeit auf die übrig bleibende VM wechseln können.	/etc/motd anpassen und Benutzer darauf hinweisen, auf die andere VM zu verbinden.
Web	Manuell durch Anpassung der web-prod0/web-prod1 DNS-Aliases.	Anpassung des web-prod0/web-prod1 Aliases, wodurch alle Clients auf die andere VM verbinden und Wartungsarbeiten möglich sind.
Proxy, Messaging und weitere VM	Manuell durch Start der VM auf dem übrig bleibenden Hypervisor.	Keine speziellen Vorkehrungen für die Wartung erforderlich, die VM sind alleine stehend.

Netzwerk

Auf die Implementation einer Firewall in Form einer virtuellen Maschine oder physischer Appliance wird verzichtet. Jedes Systeme der Klasse Core und Base verfügt über eine host-based Firewall (Shorewall, http://shorewall.net/). Die Umsetzung einer host-based Firewall auf Systemen der Klasse Other liegt im Ermessen des Betreibers der VM.

Für die Anbindung an das Netzwerk von Init7 wird ein Switch benutzt, welcher auf einem Port den Uplink zu Init7 hat, an den beiden anderen Ports je einen Link zu jedem Hypervisor.