Sympa mit Postfix einrichten

Diese Seite beschreibt die Installation und Einrichtung von Sympa, Postfix und Nginx oder Apache unter Debian Bullseye.

Installation von Sympa

- Sympa installieren:

```
apt install sympa
```

- Bei der Installation wird gleich die Datenbank mit eingerichtet. Dies kann auch manuell durchgeführt werden.

Postfix einrichten

Prinzipiell gibt es zwei Möglichkeiten, wie Sympa mit Postfix zusammen arbeiten kann:

- Alias-Maps: für jede Liste werden eigene Alias-Einträge konfiguriert
- Transport-Maps: allgemein gültige Konfiguration der Listendomain

Die Alias-Maps-Variante ist in der offiziellen Sympa-Dokumentation beschrieben. Die Nutzung der Transport-Map vereinfacht die Konfiguration jedoch erheblich. Diese wird im Folgenden auch beschrieben.

- Anpassen der `/etc/postfix/master.cf`:

```
sympa      unix  -       n       n       -       -       pipe
  flags=hqRu null_sender= user=sympa
argv=/usr/lib/sympa/lib/sympa/queue ${recipient}
sympabounce unix -       n       n       -       -       pipe
  flags=hqRu null_sender= user=sympa
argv=/usr/lib/sympa/lib/sympa/bouncequeue ${user}@${domain}
sympabounce2 unix -       n       n       -       -       pipe
  flags=hqRu null_sender= user=sympa
argv=/usr/lib/sympa/lib/sympa/bouncequeue sympa@${domain}
```

- In `/etc/sympa/sympa/sympa.conf` wird der Pfad zum MTA festgelegt - wichtig ist hier, das „Postfix to Sendmail compatibility interface“ zu nutzen und nicht `/usr/sbin/postfix`:

```
sendmail        /usr/sbin/sendmail
```

- Zur Nutzung der Transport-Maps wird die Alias-Funktionalität von Sympa unter `/etc/sympa/sympa/sympa.conf` deaktiviert:

```
sendmail_aliases  none
```

- Folgende Einstellungen gehören nun in die `/etc/postfix/main.cf`:

```
sympa_destination_recipient_limit = 1
sympabounce_destination_recipient_limit = 1
recipient_delimiter = +
virtual_mailbox_domains = listen.example.org
virtual_alias_maps = regexp:/etc/postfix/sympa_virtual_alias
transport_maps = hash:/etc/postfix/sympa_transport_misc,
pcre:/etc/postfix/sympa_transport
```

- Die Map für Aliasnamen wird festgelegt. Die Umschreibung von `-owner` ist notwendig, da ansonsten die Mail nicht korrekt verarbeitet wird. - `/etc/postfix/sympa_virtual_alias`:

```
/^sympa-(owner|request)@listen\.example\.org$/  postmaster@example.org
/^(.*)-owner@listen\.example\.org$/
$1+owner@listen.example.org
```

- Damit die Mailverarbeitung von Sympa mit dem beschriebenen `-owner`-Mapping korrekt funktioniert, muss der `return_path_suffix` in der Standardeinstellung (`-owner`) belassen werden.
- Die definierte Transport-Map-Tabelle wird nun gefüllt - `/etc/postfix/sympa_transport`:

```
/^.*\+owner@listen\.example\.org$/        sympabounce:
/^(?!bounce\+)(.*)@listen\.example\.org$/ sympa:
```

- Damit die Abweisung von ungültigen Empfängeradressen korrekt funktioniert und wichtige Sympa-Adressen nicht abgelehnt werden, wird nun eine zweite Transport-Map-Tabelle angelegt - `/etc/postfix/sympa_transport_misc`:

```
abuse-feedback-report@listen.example.org sympabounce2:
bounce@listen.example.org                sympabounce2:
listmaster@listen.example.org            sympa:
sympa@listen.example.org                 sympa:
```

- Anschließend:

```
postfix reload
```

- Für jede neue Maildomain werden dann nur noch die entsprechenden Eintrage in den Transport- und Map-Dateien ergänzt.

Reject von ungültigen Empfängern

Mit der aufgeführten Postfix-Konfiguration werden alle Anfragen an die definierte `virtual_mailbox_domains` angenommen und an Sympa weiter geleitet, auch solche, die einen ungültigen, nicht existierenden Listennamen enthalten. Dies kann insbesondere zu Backscatter-Problemen führen. Wünschenswert wäre, dass Postfix prüft, ob eine Liste existiert und bei einem ungültigen Listennamen die Annahme der Mail verweigert. Dies lässt sich mit einer einfachen sql-Abfrage machen. In diesem Beispiel wird nur der user-Teil der Mailadresse überprüft.

- Map definieren - `/etc/postfix/main.cf`:

```
virtual_mailbox_maps = hash:/etc/postfix/sympa_transport_misc,
proxy:mysql:/etc/postfix/sympa_virtual_mailbox_maps.cf
```

- SQL-Abfrage definieren - `/etc/postfix/sympa_virtual_mailbox_maps.cf`:

```
user = sympa
password = strenggeheim
hosts = localhost
dbname = sympa
query = SELECT 'present' FROM list_table WHERE name_list='%u'
        or    name_list = replace('%u', '-request', '')
        or    name_list = replace('%u', '-editor', '')
        or    name_list = replace('%u', '-subscribe', '')
        or    name_list = replace('%u', '-unsubscribe', '')
```

Webserver einrichten

- Apache als Webserver für Sympa scheint um einiges performanter als ein Setup mit Nginx zu sein. Zudem gibt es einen Bug, der in der folgenden Nginx-Konfiguration aufgritt, bei Apache hingegen nicht.
- Die nachfolgende Konfiguration von Nginx und Apache geht davon aus, dass Sympa unter der Root-Domain ohne Pfad erreichbar ist (also `https://listen.example.org` statt `https://listen.example.org/sympa`).

Systemd-Unis anlegen

- Für die Weboberfläche müssen zwei Systemd-Units angelegt werden, die aktuell noch nicht im Debian-Paket enthalten sind.
- Die Datei `/etc/systemd/system/sympa.wwsympa.service` anlegen:

```
[Unit]
Description = Sympa Web Interface FastCGI backend

[Service]
User = sympa
Group = sympa
ExecStart = /usr/lib/cgi-bin/sympa/wwsympa.fcgi
StandardOutput = null
StandardInput = socket
StandardError = null

Restart = always
RestartSec = 5

[Install]
WantedBy = multi-user.target
```

- Die Datei `/etc/systemd/system/sympa.wwsympa.socket` anlegen:

```
# /etc/systemd/system/wwsympa.socket
[Unit]
Description = Sympa Web Interface Socket

[Socket]
SocketUser = www-data
SocketGroup = www-data
SocketMode = 0660

RuntimeDirectory = sympa
ListenStream = /run/wwsympa/wwsympa.socket

[Install]
WantedBy = sockets.target
```

- Die neuen Services aktivieren:

```
systemctl daemon-reload
systemctl enable wwsympa.service
systemctl enable wwsympa.socket
```

- Und starten:

```
systemctl start wwsympa.service
systemctl start wwsympa.socket
```

Nginx

WWSympa, die Weboberfläche für Sympa, ist ein Perl-CGI-Skript. Zur Beschleunigung kann Fastcgi genutzt werden. Da Nginx lediglich Fastcgi unterstützt, wird zusätzlich das Debian-Paket Fcgiwrap benötigt.

- Installation der Pakete:

```
apt install nginx-light fcgiwrap
```

- Fastcgi in `/etc/sympa/sympa/sympa.conf` aktivieren:

```
use_fast_cgi 1
```

- Seiten-Konfiguration unter `/etc/nginx/sites-available/sympa` anlegen:

```
server {
    server_name  listen.example.org;
    root         /usr/share/sympa;
    access_log   /var/log/nginx/sympa.access.log;
    error_log    /var/log/nginx/sympa.error.log;
    error_page   403 500 502 503 504 /50x.html;

    # While configuring sympa, you should specify wwsympa_url for each
```

```
robot.
    # if you do not do so, sympa will generate wwsympa_url as
${robot_name}/sympa.
    # So to prevent non-active urls for robots without wwsympa_url, we
do this redirect:

    rewrite ^/sympa/(.*)$ /$1 permanent;
    location ^~ /static-sympa/ {
        alias /var/lib/sympa/static_content/;
        access_log off;
    }
    location ^~ /css-sympa {
        alias /var/lib/sympa/css;
        access_log off;
    }
    location ^~ /pictures-sympa {
        alias /var/lib/sympa/pictures;
        access_log off;
    }
    location /50x.html {
        root /usr/share/nginx/html;
    }
    location ~* \.(php|pl|py|jsp|asp|sh|cgi|bin|csh|ksh|out|run|o)$ {
        deny all;
    }
    location ~ /\.ht {
        deny all;
    }
    location / {
        include /etc/nginx/fastcgi_params;
        fastcgi_pass  unix:/run/wwsympa.socket;
    }
}
```

- Anschließend Seitenkonfiguration aktivieren und Nginx neu laden.

Apache

- Eine Konfigurationsdatei für Sympa ist im Debian-Paket bereits enthalten. Allerdings wurde diese mit der Bullseye-Aktualisierung noch nicht angepasst.
- Eine Beispielkonfiguration könnte so aussehen `/etc/apache2/sites-enabled/listen.example.org.conf` so aussehen (Die SSL-Konfiguration ist in diesem Beispiel nicht aufgeführt):

```
<VirtualHost *:80>
        ServerName      listen.example.org
        DocumentRoot    /usr/share/sympa
        ErrorLog        /var/log/apache2/listen.example.org.log
        include         /etc/apache2/conf-available/sympa.conf
```

```
</VirtualHost>
```

- Noch `/etc/apache2/conf-available/sympa.conf` anpassen:

```
<IfModule mod_proxy_fcgi.c>
    Alias /css-sympa /var/lib/sympa/css
    Alias /pictures-sympa /var/lib/sympa/pictures
    Alias /static-sympa /usr/share/sympa/static_content

    <Directory /usr/share/sympa/static_content>
        Require all granted
    </Directory>

    <Directory /var/lib/sympa/css>
        Require all granted
    </Directory>

    <Directory /var/lib/sympa/pictures>
        Require all granted
    </Directory>

    <LocationMatch "^/(?!.*-sympa)">
       SetHandler "proxy:unix:/run/wwsympa/wwsympa.socket|fcgi://"
       Require all granted
    </LocationMatch>
</IfModule>
```

- Der dargestellte `LocationMatch` leitet alle Sympa-Anfrage über den Proxy weiter. Ausnahme werden für alle Anfragen gemacht, die auf `-sympa` enden. Das umfasst die oben dargestellten Aliase. Diese Anfragen werden ohne Proxy direkt vom Webserver beantwortet.

Authentifizierung in der Weboberfläche

Sympa kann mehrere Quellen zur Authentifizierung nutzen. Diese werden in `/etc/sympa/auth.conf` definiert.

- Im folgenden Beispiel erfolgt die Authentifizierung bei Nutzer mit einer example.org-Mailadresse per LDAP, alle anderen Nutzer werden über die interne Datenbank authentifiziert:

```
# LDAP authentication
ldap
        regexp  example\.org
        host    ldap.example.org
        timeout 20
        suffix  sc=mailAccount,ou=People,o=ldap,dc=example,dc=org
        get_dn_by_uid_filter    (cn=[sender])
        get_dn_by_email_filter  (mail=[sender])
        email_attribute         mail

# Internal authentication by email and password
```

```
user_table
        negative_regexp example\.org
```

- Anmeldungen mit einem LDAP-Account sind sowohl nur mit Nutzername (`get_dn_by_uid_filter`), als auch mit der vollständigen Mailadresse (`get_dn_by_email_filter`) möglich.
- Volle Administrationsrechte für die Weboberfläche erhalten alle Accounts, die unter `listmasters` in `/etc/sympa/sympa/sympa.conf` geführt werden. Listmaster haben auf alle Robots volle Rechte.

Multidomain-Unterstützung

Eine Sympa-Instanz kann beliebig viele Listen-Domains bedienen. Für die Weboberfläche werden dann, ähnlich zu Apache, virtuelle Host (Robots genannt) eingerichtet.

- MX-Einträge und CNAME einrichten
- Listenverzeichnis erstellen und Besitzer (`sympa:sympa`) ändern:

```
mkdir /var/lib/sympa/list_data/ml.example.org
```

- Konfigurationsverzeichnis erstellen und Besitzer ändern:

```
mkdir /etc/sympa/ml.example.org
```

- Konfigurationsdatei für den neuen Robot anlegen - `/ect/sympa/ml.example.org/robot.conf` und Besitzer ändern:

```
http_host  ml.example.org
wwsympa_url  ml.example.org
email  ml.example.org
title  Listenverwaltung 2
```

- Damit die Nutzeranmeldung funktioniert, muss die `auth.conf` nach `/ect/sympa/ml.example.org` kopiert werden
- Eigene Webserver-Seitenkonfiguration anlegen

Nutzern die Listenerstellung erlauben

Ziel ist es, dass nicht nur Admins Listen erstellen können, sondern auch ausgewählte reguläre Nutzer:innen. Dazu werden alle berechtigten Nutzer:innen in einem Scenario aufgelistet.

- Zuerst wird in der `sympa.conf` festgelegt, welches Scenario die Rechte für die Listenerstellung definiert:

```
create_list     list_creators
```

- Nun wird ein entsprechendes Scenario angelegt. Bei nur einem Robot liegt sie unter `/etc/sympa/scenari`. Bei verschiedenen Robots unter `/ect/sympa/$ROBOT/scenari`. Der Dateiname muss eine Kombination aus value.wert entsprechend der `sympa.conf` sein - in

diesem Fall also create_list.list_creators:

```
title Users that can create lists
equal([sender], 'foo@example.org') md5 -> do_it
```

- Das Scenario definiert also, dass eine bestimmte Nutzerin (`equal([sender], 'foo@example.org'`) nach der Anmeldung im Webinterface (`md5`) entsprechende Rechte (`–> listmaster`) erhält.
- Nach dem selben Schema können nun weitere Nutzer:innen definiert werden
- Mit der Definition von `create_list` verlieren die listmaster die Möglichkeit, Listen anzulegen - deshalb kann das ursprüngliche Scenario am Ende des neuen Scenarios eingebunden werden. Für das Scenario gilt: First match wins:

```
title Users that can create lists
equal([sender], 'foo@example.org') md5 -> do_it
include  create_list.listmaster
```

- Da der erwartete Dateiname nun `include.create_list.listmaster` ist, muss noch ein Symlink erstellt werden:

```
ln -s /usr/share/sympa/default/scenari/create_list.listmaster
/etc/sympa/scenari/include.create_list.listmaster
```

- Abschließend muss Sympa neu gestartet werden

Umgang mit Spam

Findet durch Sympa keine Spambehandlung statt, werden Spam-Nachrichten an alle Abonnent:innen zugestellt. Verweigern dann die finalen Mailserver die Annahme, steigt der Fehlerstatus der Liste. Dies kann dann dazu führen, dass Sympa für die jeweilige Liste überhaupt keine Nachrichten mehr annimmt.

Voraussetzung für die Sympa-interne Erkennung und Behandlung von Spam ist das vorgeschaltete Prüfen und Markieren von eingehenden E-Mails durch einen Spamfilter. Üblicherweise erhalten die Mails dann einen zusätzlichen X-Spam-Header, bspw. bei Rspamd `X-Spam-Status: Yes`. Sympa kann aufgrund dieses Headers die Spam-Nachricht intern auch als solche markieren und speziell behandeln.

Tag-basierte Spamerkennung

Die Tag-basierte Spambehandlung wird über die `sympa.conf` konfiguriert. Folgende Optionen stehen zur Verfügung:

```
antispam_feature on
antispam_tag_header_name X-Spam-Status
antispam_tag_header_spam_regexp ^\s*Yes
antispam_tag_header_ham_regexp ^\s*No
```

Wobei die Option `antispam_feature on` die Spamassassin-Defaults für die anderen antispam-

Optionen aktiviert. Zudem sorgt diese Option dafür, dass Spam-Nachrichten im Webinterface speziell markiert werden.

Scenario-basierte Spamerkennung

Diese ist standardmäßig über das Scenario `spam_status.x-spam-status` aktiviert. Über die Option `spam_status` kann in der `/etc/sympa/sympa.conf` jedoch auch ein anderes Scenario definiert werden. Im Scenario selbst kann dann die Prüfung mehrerer unterschiedlicher Header definiert werden.

Damit Spam-Nachrichten auch als solche im Webinterface angezeigt werden, sollte ebenfalls die Option `antispam_feature on` aktiviert werden.

Spambehandlung

Ziel ist es, dass Sympa eine als Spam markierte Nachricht (`X-Sympa-Spam-Status: Yes`) in die Moderationswarteschlange leitet. Da passiert in zwei Schritte: Über das Scenario `/usr/share/sympa/default/scenari/spam_status.x-spam-status` wird definiert, wann eine Mail als Spam gilt. Für eigene Anpassungen kann die Datei nach `/etc/sympa/scenari` kopiert werden. Wir verwenden Rspamd für die Spammarkierung. Entsprechend wird die Datei angepasst. Anmerkung: Rspamd befüllt den `X-Spam-Header` mit `Yes` statt mit `yes` wie Spamassassin:

```
title.gettext test x-spam-status  header

match([header->X-Spam-Status][-1],/^\s*(?i)yes/)        smtp,dkim,smime,md5
-> spam
true()                                                  smtp,dkim,md5,smime
-> ham
```

Anschießend muss ein `send`-Scenario definiert werden, das die Mail in die Moderationswarteschlage verschiebt. Praktischerweise lässt sich ein gemeinsames Scenario definieren, dass für alle Listentypen genutzt wird. Es wird als `/etc/sympa/scenari/include.send.header` angelegt:

```
match([msg->spam_status], /spam/) smtp,dkim,md5,smime -> editorkey,quiet
```

Links

- https://www.sympa.community/manual/customize/basics-acceptance.html#loop-prevention –> stille Verweigerung von Listenmails (bspw. gleiche Message-ID)
- Nginx-Konfiguration
- Nginx-Konfiguration im Debian-Wiki
- Topics anpassen: /etc/sympa/topics.conf

From:
https://howto.wikis.systemausfall.org/ - **Das HowTo-Wiki**

Permanent link:
https://howto.wikis.systemausfall.org/linux/sympa_mit_postfix

Last update: **2022/11/20 20:17**