Este manual indica como configurar la opción de **vacation** en un servidor que utiliza postfix-amavis-mysql-courier-imap y demás, siguiendo estos pasos se consigue de manera sencilla la configuración de este servicio.

Esto trabaja bajo el script vacation.pl, el cual se encarga de enviar la notificaciones y guardar en la base de datos el email a donde fue enviado la notificación, este script viene con el postfixadmin cuando es descargado.

Quizás tengan que cambiar ciertas partes dependiendo si usan FreeBSD o Centos (directorios, usuarios, etc) como sistema operativo, también en la base de datos del postfix tendrán que crear ciertas tablas que no se crean cuando van a usar el postfix.

aquí el manual

```
$ mysql postfix
mysql> insert into transport values('vacation.domain.tld','vacation');
mysql> select * from transport;
+---------------------+----------+
| domain              | transport|
+---------------------+----------+
| domain.tld          | virtual: |
| vacation.domain.tld | vacation |
+---------------------+----------+
```

Donde es domain.tld debes cambiar por el dominio que vas a usar, lo de tester puedes conservarlo.

```
mysql> select * from users;
+--+------+----------------+------------+------+----+---------------+
|id|login |address         |password    |name  |home|maildir        |
+--+------+----------------+------------+------+----+---------------+
|1 |tester|tester@domain.tld|.ieXhigkolJUU|Tester|/   |/postfix/tester/|
+--+------+----------------+------------+------+----+---------------+
mysql> update virtual set goto='tester@domain.tld,tester@vacation.domain.tld'
where address='tester@domain.tld';
mysql> select * from virtual;
+---------------------+-----------------------------------------------------+
| address             | goto                                                |
+---------------------+-----------------------------------------------------+
| tester@domain.tld   | tester@domain.tld,tester@vacation.domain.tld        |
+---------------------+-----------------------------------------------------+
mysql> CREATE TABLE vacation (
  email varchar(255) NOT NULL default '',
  subject varchar(255) NOT NULL default '',
  body text NOT NULL,
  cache text NOT NULL,
  domain varchar(255) NOT NULL default '',
  created datetime NOT NULL default '0000-00-00 00:00:00',
  active tinyint(4) NOT NULL default '1',
  PRIMARY KEY  (email),
  KEY email (email)
) TYPE=MyISAM COMMENT='Postfix Admin - Virtual Vacation';

mysql> INSERT INTO vacation VALUES ('tester@domain.tld','Out Of Office',
'I will be out of the office until January, please do not hesitate to contact me
on my mobile +32 444 55 66', '','domain.tld','0000-00-00 00:00:00',1);

mysql> quit
```

Ok hasta aquí toda la configuración de que usuario tenga el servicio de vacation activado atraves de la consola del MYSQL, tambien puedes darle la potestad al propio usuario de que lo haga por medio del

postfixadmin, mas adelante se indica como se puede usar.

En este caso se esta implementando en un servidor con Centos 4.4, en caso de que se va a usar FreeBSD lo que cambia son los comandos y directorios, ahora se ejecuta lo siguiente:

```
$ useradd vacation
$ groupadd vacation
$ mkdir /var/spool/vacation
```

Con este link se descarga el postfixadmin, puede ser descargado desde cualquier lugar que tengas confianza o desde la misma pagina del postfixadmin.

```
$ wget http://high5.net/postfixadmin/download.php?file=postfixadmin-2.1.0.tgz

$ tar -xzf postfixadmin-2.1.0.tgz
$ cp postfixadmin-2.1.0/VIRTUAL_VACATION/vacation.pl /var/spool/vacation
$ chown -R vacation.vacation /var/spool/vacation
```

Ahora se hace unos cambios en el archivo de master.cf del Postfix.

```
$ vi /etc/postfix/master.cf
[...]
vacation    unix  -       n       n       -       -       pipe
 flags=Rhu user=vacation argv=/var/spool/vacation/vacation.pl

$ ### READ postfixadmin-2.1.0/VIRTUAL_VACATION/INSTALL.TXT !!! ###
```

Con esto se hace que cada correo que reciba mail sea revisado por vacation.pl para ver si tiene activado el servicio de Vacation.

Reiniciamos el servicio.

```
$ /etc/init.d/postfix restart
Stopping mail transport agent: Postfix.
Starting mail transport agent: Postfix.
$
```

De esta manera el usuario que se configure con el servicio vacation tendra un mensaje de respuesta, tambien el mismo usuario puede personalisar el mensaje de ausencia ingresando al administrador web del postfix:

http://host.dominio.com/postfixadmin/users

Aquí pueden ingresar bajo su propio usuario y contraseña, una ves dentro tienen la opción de poner el mensaje de ausente, cambiar la contraseña de correo y si desean poner un alias a su cuenta.

Cabe señalar que cuando se envía un correo a un usuario que tiene configurado el vacation o auto-response como también se lo conoce, la persona que envía el correo solo va a recibir una sola notificación de que el usuario en cuestión esta de vacaciones, no recibira mas, en caso de querer sacar esta opción se debe tener en cuenta de que se pueden causar bucles o loops en el servidor de correo si alguien que tiene configurado el vacation envía a otro usuario que también lo tiene.

También encontraran el código del script vacation.pl para que vean como funciona cuandoun usuario tiene activado esta opción y que es lo pasa.

vacation.pl

```
#!/usr/bin/perl -w
#
# Virtual Vacation 3.1
# by Mischa Peters <mischa at high5 dot net>
# Copyright (c) 2002 - 2005 High5!
# License Info: http://www.postfixadmin.com/?file=LICENSE.TXT
#
# Additions:
# 2004/07/13  David Osborn <ossdev at daocon.com>
#             strict, processes domain level aliases, more
#             subroutines, send reply from original to address
#
# 2004/11/09  David Osborn <ossdev at daocon.com>
#             Added syslog support
#             Slightly better logging which includes messageid
#             Avoid infinite loops with domain aliases
#
use DBI;
use strict;

my $db_type = 'mysql';
my $db_host = 'localhost';
my $db_user = 'postfixadmin';
my $db_pass = 'postfixadmin';
my $db_name = 'postfix';
my $sendmail = "/usr/sbin/sendmail";
my $logfile = "";    # specify a file name here for example: vacation.log
my $debugfile = "";  # sepcify a file name here for example: vacation.debug
my $syslog = 0;   # 1 if log entries should be sent to syslog

my $dbh = DBI->connect("DBI:$db_type:$db_name:$db_host", "$db_user", "$db_pass",
{ RaiseError => 1 });

# used to detect infinite address lookup loops
my $loopcount=0;

sub do_query {
   my ($query) = @_;
   my $sth = $dbh->prepare($query) or die "Can't prepare $query: $dbh->errstr\n";
   $sth->execute or die "Can't execute the query: $sth->errstr";
   return $sth;
}

sub do_debug {
   my ($in1, $in2, $in3, $in4, $in5, $in6) = @_;
   if ( $debugfile ) {
   my $date;
   open (DEBUG, ">> $debugfile") or die ("Unable to open debug file");
   chop ($date = `date "+%Y/%m/%d %H:%M:%S"`);
   print DEBUG "====== $date ======\n";
      printf DEBUG "%s | %s | %s | %s | %s | %s\n", $in1, $in2, $in3, $in4, $in5,
$in6;
   close (DEBUG);
   }
}

sub do_cache {
   my ($to, $from) = @_;
   my $query = qq{SELECT cache FROM vacation WHERE email='$to' AND
FIND_IN_SET('$from',cache)};
   my $sth = do_query ($query);
   my $rv = $sth->rows;
   if ($rv == 0) {
```

```
      $query = qq{UPDATE vacation SET cache=CONCAT(cache,',','$from') WHERE
email='$to'};
      $sth = do_query ($query);
   }
   return $rv;
}

sub do_log {
   my ($messageid, $to, $from, $subject) = @_;
   my $date;
   if ( $syslog ) {
       open (SYSLOG, "|/usr/bin/logger -p mail.info -t Vacation") or die ("Unable
to open logger");
       printf SYSLOG "Orig-To: %s From: %s MessageID: %s Subject: %s", $to,
$from, $messageid, $subject;
       close (SYSLOG);
   }
   if ( $logfile ) {
   open (LOG, ">> $logfile") or die ("Unable to open log file");
   chop ($date = `date "+%Y/%m/%d %H:%M:%S"`);
       print LOG "$date: To: $to From: $from Subject: $subject MessageID:
$messageid \n";
   close (LOG);
   }
}

sub do_mail {
   my ($from, $to, $subject, $body) = @_;
   open (MAIL, "| $sendmail -t -f $from") or die ("Unable to open sendmail");
   print MAIL "From: $from\n";
   print MAIL "To: $to\n";
   print MAIL "Subject: $subject\n";
   print MAIL "X-Loop: Postfix Admin Virtual Vacation\n\n";
   print MAIL "$body";
   close (MAIL);
}

sub find_real_address {
   my ($email) = @_;
   if (++$loopcount > 20) {
      do_log ("find_real_address loop!", "currently: $email", "ERROR", "ERROR");
      print ("possible infinite loop in find_real_address for <$email>. Check for
alias loop\n");
      exit 1;
   }
   my $realemail;
   my $query = qq{SELECT email FROM vacation WHERE email='$email' and active=1};
   my $sth = do_query ($query);
   my $rv = $sth->rows;

   # Recipient has vacation
   if ($rv == 1) {
      $realemail = $email;

   } else {
      $query = qq{SELECT goto FROM alias WHERE address='$email'};
      $sth = do_query ($query);
      $rv = $sth->rows;

      # Recipient is an alias, check if mailbox has vacation
      if ($rv == 1) {
         my @row = $sth->fetchrow_array;
         my $alias = $row[0];
         $query = qq{SELECT email FROM vacation WHERE email='$alias' and
active=1};
         $sth = do_query ($query);
```

```
         $rv = $sth->rows;

         # Alias has vacation
         if ($rv == 1) {
            $realemail = $alias;
         }

      # We still have to look for domain level aliases...
      } else {
         my ($user, $domain) = split(/@/, $email);
         $query = qq{SELECT goto FROM alias WHERE address='\@$domain'};
         $sth = do_query ($query);
         $rv = $sth->rows;

         # The receipient has a domain level alias
         if ($rv == 1) {
            my @row = $sth->fetchrow_array;
            my $wildcard_dest = $row[0];
            my ($wilduser, $wilddomain) = split(/@/, $wildcard_dest);

            # Check domain alias
            if ($wilduser) {
               ($rv, $realemail) = find_real_address ($wildcard_dest);
            } else {
               my $new_email = $user . '@' . $wilddomain;
               ($rv, $realemail) = find_real_address ($new_email);
            }
         }
      }
   }
   return ($rv, $realemail);
}

sub send_vacation_email {
   my ($email, $orig_subject, $orig_from, $orig_to, $orig_messageid) = @_;
   my $query = qq{SELECT subject,body FROM vacation WHERE email='$email'};
   my $sth = do_query ($query);
   my $rv = $sth->rows;
   if ($rv == 1) {
      my @row = $sth->fetchrow_array;
      #esta opción la que permite enviar una sola ves el mail, si se comenta
      #este if se podrá enviar indefinidamente correos de notificación
      if (do_cache ($email, $orig_from)) { return; }
      do_debug ("[SEND RESPONSE] for $orig_messageid:\n", "FROM: $email (orig_to:
$orig_to)\n", "TO: $orig_from\n", "SUBJECT: $orig_subject\n", "VACATION SUBJECT:
$row[0]\n", "VACATION BODY: $row[1]\n");
      #con esta linea se arregla un problema que se genera cuando se envia
      #correo desde algun cliente que maneja Outlook Express o Microsoft Outlook
      #el cual es que cuando envia mail el correo que tiene el servicio de
      #vacation llegaba con #caracteres extraños (“'?, etc) y eso impedia que el
      #script pueda enviar las notificaciones de manera normal, en cambio cuando
      #se usa como cliente a Evolution  u otro programa, el correo se enviaba de
      #manera normal, lo que hace es usar el correo que esta guardado en la base
      #de datos y no el que llegaba desde el programa de correo y es enviado a
      #la funcion do_mail.
      do_mail ($row[0], $orig_from, $row[1], $row[2]);
      #do_mail ($email, $orig_from, $row[0], $row[1]);
      do_log ($orig_messageid, $orig_to, $orig_from, $orig_subject);
   }

}

############################ main ###################################
```

```
my ($from, $to, $cc, $subject, $messageid);

# Take headers apart
while (<STDIN>) {
   last if (/^$/);
   if (/^from:\s+(.*)\n$/i) { $from = $1; }
   if (/^to:\s+(.*)\n$/i) { $to = $1; }
   if (/^cc:\s+(.*)\n$/i) { $cc = $1; }
   if (/^subject:\s+(.*)\n$/i) { $subject = $1; }
   if (/^message-id:\s+(.*)\n$/i) { $messageid = $1; }
   if (/^precedence:\s+(bulk|list|junk)/i) { exit (0); }
   if (/^x-loop:\s+postfix\ admin\ virtual\ vacation/i) { exit (0); }
}

# If either From: or To: are not set, exit
if (!$from || !$to) { exit (0); }

$from = lc ($from);

# Check if it's an obvious sender, exit
if ($from =~ /([\w\-.%]+\@[\w.-]+)/) { $from = $1; }
if ($from eq "" || $from =~ /^owner-|-(request|owner)\@|^(mailer-daemon|
postmaster)\@/i) { exit (0); }

# Strip To: and Cc: and push them in array
my @strip_cc_array;
my @strip_to_array = split(/, */, lc ($to) );
if (defined $cc) { @strip_cc_array = split(/, */, lc ($cc) ); }
push (@strip_to_array, @strip_cc_array);

my @search_array;

# Strip email address from headers
for (@strip_to_array) {
   if ($_ =~ /([\w\-.%]+\@[\w.-]+)/) {
      push (@search_array, $1);
      do_debug ("[STRIP RECIPIENTS]: ", $messageid, $1, "-", "-", "-");
   }
}

# Search for email address which has vacation
for (@search_array) {
   my ($rv, $email) = find_real_address ($_);
   if ($rv == 1) {
         do_debug ("[FOUND VACATION]: ", $messageid, $from, $to, $email,
$subject);
       send_vacation_email( $email, $subject, $from, $to, $messageid);
   }
}

0;
```