Using check_* in sendmail 8.8

Last Update 1997-11-02

Content: check_mail | check_rcpt | check_relay | check_compat | Some more hints

Introduction

sendmail 8.8 introduces several new rulesets to check who can use your machine to send e-mail and to avoid UCE from well-known sites. These are:

`check_mail`
: for the `MAIL` command,

`check_rcpt`
: for the `RCPT` command,

`check_relay`
: checking the host name and host address separated by `$|` ,

check_compat
: checking both `MAIL` and `RCPT` also separated by `$|` before delivery.

Moreover, it also defines new macros: `${client_name}, ${client_addr},` and `${client_port}` that have the name, IP address, and port number of the SMTP client, respectively.

After you have read this text carefully, you can download (last update: 1997-11-02) the HACKs for use in your `.mc` file. Be sure you understand what they are trying to accomplish and check them yourself before you use them on a production system!

General Information about the new Rules

These new rulesets can do whatever you want, the only interesting case is when the resolve to the the `$#error` mailer, i.e., `$#error $@ error-code $: error-text` . In this case the indicated message is printed and the command is rejected with an error code. sendmail can return different reply codes, which are defined in RFC 1893 . However, if you don't repeat the error number in the error text, sendmail uses the default value 553.

Just as a reminder, the parts of an (E)SMTP dialogue which are important here are:
`MAIL From:<sender@address>`
to specify the sender's address, and
`RCPT To:<rcpt@address>`
to specify the recipient's address, of which multiple can be given.

The `check_compat` ruleset is called before delivery with `from_addr $| to_addr` as argument. The $| is a new meta-symbol used to separate the two addresses. You can use this ruleset to implement mailrouting policies, e.g., you can prevent the use of your machine as a mailgateway.

`check_compat` is called for all deliveries, while `check_mail` and `check_rcpt` are only called for SMTP connections. So the latter won't work when you run sendmail in queue delivery mode behind smap or something similar. In this case, you should try the patch for `checkcompat()`

written by Kyle Jones.

check_mail

The address supplied through the `MAIL` command can be checked against the `check_mail` ruleset.

A new version uses a map with error messages as right side. It has several advantages over the version presented here.

You can use this to prevent known spammers from sending you e-mail. First, you may have a list of domains in an external file which you want to ban completely:
```
F{SpamDomains} /etc/mail/SpamDomains
```
e.g.,

```
cyberpromo.com
quantcom.com
```

Next, you may have a list of users which you want to ban too:
```
F{Spammer} /etc/mail/Spammer
```
e.g.,

```
lamer@aol.com
```

Now you can use these as follows:

```
Scheck_mail
R<$={Spammer}>          $#error $@ 5.7.1 $: "551 We don't accept junk mail"
R<$={Spammer}.>         $#error $@ 5.7.1 $: "551 We don't accept junk mail"
R$*                     $: $>3 $1
R$*<@$={SpamDomains}.>$*        $#error $@ 5.7.1 $: "551 We don't accept junk ma
R$*<@$={SpamDomains}>$*         $#error $@ 5.7.1 $: "551 We don't accept junk ma
```

In addition, you may want to act on broken mailers which don't use <> around addresses:

```
R$={Spammer}            $#error $@ 5.7.1 $: "551 We don't accept junk mail"
R$={Spammer}.           $#error $@ 5.7.1 $: "551 We don't accept junk mail"
```

If you want to stop receiving mails from subdomains of well known spammers, you can modify the last two rules a bit:

```
R$*<@$*$={SpamDomains}.>$*      $#error $@ 5.7.1 $: "551 We don't accept junk ma
R$*<@$*$={SpamDomains}>$*               $#error $@ 5.7.1 $: "551 We don't accept
```

Next step could be the following: you want also to reject mail from those domains, which are not registered in the DNS. However, this may also be a temporary failure, so you should give back a temporary failure.

```
# if you enable the last rule, you can disable this one.
# host without a . in the FQHN ?
R$*<@$->$*      $#error $@ 4.1.8 $: "451 invalid host name"             no real
# lookup IP address (reverse mapping available?)
# R$*<@[$-.$-.$-.$-]>$* $: $1 < @ $[ [ $2.$3.$4.$5 ] $] > $6
# no DNS entry? this is dangerous!
# R$*<@$*$~P>$* $#error $@ 4.1.8 $: "451 unresolvable host name, check your conf
```

The hint to perform a reverse-mapping of the IP address comes from Jan Kr ger.

Paul Vixie maintains a MAPS Realtime Blackhole List which can be used as follows:

```
R$*                        $: $(dequote "" $&{client_addr} $)
R$-.$-.$-.$-               $: $[ $4.$3.$2.$1.rbl.maps.vix.com $]
R$-.$-.$-.$-.rbl.maps.vix.com.  $#error $@ 5.7.1 $: 551 no access from [$4.$3.$2
```

This code can be appended to `check_mail` or some other rule. Thanks to Thomas 'Mike' Michlmayr for this suggestion.

check_rcpt

The address supplied through the `RCPT` command can be checked against the `check_rcpt` ruleset. On first look, this ruleset doesn't make much sense. Why check the recipient? sendmail does this anyway when trying to deliver, esp. for local recipients. However, this ruleset can be used to check whether your system is (mis)used as a gateway. The `check_compat` ruleset, which seems to be better suited for this purpose, since it gets both addresses (sender and recipient) as parameters, is called too late. To reject a misuse at the earliest moment (and save your bandwith etc), you can refer to the address of the sending system, which is available in the macro `${client_addr}` . However, to use it in a rule, you have to refer to it as: `$(dequote "" $&{client_addr} $)` so sendmail defers evaluation and tokenizes it. The old solution is based on a proposal from Chin Huang: But since there is a problem with these rules, here is a new solution. First, we check whether it is a local client: it can do whatever it want. Next, we remove the local part, maybe repeatedly. If it still has routing information in it, it seems to be a relay attempt. So list in the class

```
F{LocalIP} /etc/mail/LocalIP
```

the IP addresses of the local clients you will allow to relay through your mail server, for example

```
134.245
127.0.0.1
```

A client which connects from one of these IP numbers can send mail through your gateway anywhere.

```
Scheck_rcpt
# first: get client addr
R$+                        $: $(dequote "" $&{client_addr} $) $| $1
R0 $| $*                   $@ ok                  no client addr: directly invoked
R$={LocalIP}$* $| $*       $@ ok                  from here
# not local, check rcpt
R$* $| $*                  $: $>3 $2
# remove local part, maybe repeatedly
R$*<@$=w.>$*               $>3 $1 $3
# still something left?
R$*<@$+>$*                 $#error $@ 5.7.1 $: 551 we do not relay
```

The trailing `$*` after `$={LocalIP}` matches incompletely specified IP addresses on octet boundaries, as can be seen by `134.245` which matches a whole class B subnet.

If you relay mail for other systems (e.g., the secondary MX of a system points to your mailhost or your server is the primary MX, but you forward the mail to another system), use also:

```
F{RelayTo} /etc/mail/RelayTo
```

to list all hosts you relay mail to or accept mail for. For example, we put

```
uni-kiel.de
```

in `RelayTo` . Then change the line

```
R$*<@$=w.>$*             $>3 $1 $3
```

to

```
R$*<@$*$={RelayTo}.>$*          $>3 $1 $4
```

(or just add the latter). The leading `$*` will match with subdomains of those domains in `RelayTo` too. You can also use a map instead of a class, if you slightly change the rules.

Several people asked for a possibility to allow relaying based on the FROM address too. Take a look at another version which should be able to do just this.

Can anyone see a problem with this simpler approach? If somebody has a better solution for this, please let me know.

check_relay

`check_relay` gets the host name and host address of the client separated by `$|` as parameters. This can be used as a substitute for `TCPWRAPPERS` . You can enable the code for `TCPWRAPPERS` by compiling sendmail with `-DTCPWRAPPERS=1` . A small example is:

```
F{DeniedIP} /etc/mail/DeniedIP
F{DeniedNames} /etc/mail/DeniedNames
```

These (plain text) files contain a list of IP addresses and hostnames which are not allowed to access your mailserver.

```
Scheck_relay
R$+ $| $={DeniedIP}$*           $#error $@ 5.7.1 $: "no access from your IP addr
R$*$={DeniedNames} $| $*        $#error $@ 5.7.1 $: "no access from your host"
```

(note the trailing/leading `$*` to match with incompletely specified IP addresses/names).

Access will be refused with the error message:

```
550 Access denied
```

and the error string will be logged.

If you try to use maps, you need a patch for sendmail 8.8.5 (it's fixed in later versions). Have a look at a version which uses the same databases (with some additions) as check_mail . The latest version allows to specify individual error messages.

check_compat

Although `check_compat` gets both addresses (sender and recipient) as parameters to check whether your machine is used as a gateway, it's too late. `check_compat` is called after the whole message has been transmitted. You could do something like this:

```
Scheck_compat
```

```
R$+ $| $+               $: $2 $| $>3 $1 canonicalize sender
R$+ $| $+               $: $2 $| $>3 $1 canonicalize recipient
R$- $| $+               $@ok          from here
R$+ $| $-               $@ok          to here
R$+<@$=w.> $| $+        $@ok          from here
R$+ $| $*<@$=w.>        $@ok          to here
R$*                     $#error $@ 5.7.1 $: "551 we do not support relaying"
```

to prevent (mis)use of your machine as a mail gateway by other people. Maybe you have to use some other class than `w` . If you have a better example for this purpose, please let me know. However, this ruleset has a problem with forwarding. That's one of the reasons why you should use the check_rcpt solution .

Some more hints

Problems with Reply Codes 571, 418

Thanks to Sean Vickery for pointing out a mistake with the reply codes used earlier. The correct SMTP reply codes are `45x` or `55x` according to RFC 821 . The codes `5.7.1, 4.1.8` etc. are *Enhanced Mail System Status Codes* as defined in RFC 1893 .

Nasty?

If you have a good connection to the internet, and you want to be a bit *nasty* , you could change all error codes from permanent (5) to temporary (4), e.g., instead of `"551 ..."` you can use `"451 ..."`. This has the consequence that the spammer will try again later on, so his resources are tied up. The more people implement such a scheme, the longer it will take the spammer to distribute his junk. *But be careful:* this ties up your resources too!

Testing the rules

Before installing these rules on a productions system, you need to test them. Read the guide to debug these new rules. It also contains a link to a script to test mail relaying capabilities .

Other proposals

More examples can be found in Anti-Spam Provisions in Sendmail 8.8 and in this list , which also contains some links to pages with more explanations.

Yet another possibility is to use the `checkcompat()` routine. Kyle Jones proposed the following patch. It is intended to disallow all non-local e-mail traffic through your host.

These rules can act only based on the information given during the (E)SMTP dialogue (the envelope), and on the address of the sender (or recipient). They can't filter based on the header or the content of the e-mail. If you want this, either install procmail as local delivery agent or patch sendmail.

Where to put this in sendmail.cf?

If you don't want to use the supplied HACKs to build a `sendmail.cf` from a `.mc` file then you have to place the new stuff in your `.mc` or `sendmail.cf` by hand. Or read the instructions written by Glenn Fleishman.

[(links)] [Hints] [FAQ] [cf/README] [New]