

acct_smtp: FreeBSD kernel protection measures against SMTP DDoS and DoS attacks

Martin Blapp

Abstract:

This paper is about a smtp accept filter [1] kernel module to protect mail gateways against overload- and downtime-situations, most notably against DDoS-attacks. The system only affects the smtp protocol and protocols based on top of the SMTP protocol like SMTPS or TLS (STARTTLS). Heavy load situations caused by those DDoS attacks, or many simultaneous connections from different originating IP-addresses accessing software/hardware at the same time can lead to software or hardware deadlocks or to very slow unresponsive reply times of the underlying operation system (OS) and the software running within the operating system. The kernel module solves these problems with a new innovative approach.

1. Introduction

Bots are one of the most sophisticated and popular types of cybercrime today. They allow hackers to take control of many computers at a time, and turn them into "zombie" computers, which operate as part of a powerful "botnet" to spread viruses, generate spam. Recently those Botnets are growing rapidly, these days some botnets contain millions of zombie computers. And of course, botnets are also the origin of DDoS attacks. Because there are that many different IP-Addresses involved in such a DDoS attack, generic DoS prevention systems like IP-address rate limiting or connection limits don't have any protection effect, cause they are just operating on single IPs basis.

2. Description

Traditionally, a smtp daemon accepts all connections until it hits the system limit or the limits set by its own config. The mechanism used here is called FIFO, (First In, First Out), see Fig 1. New arriving connections are rejected in such a situation. The daemon does

reject all connections and can't distinguish between attacker and normal connections, which can lead to prolonged mail delays. Because accepted connections have to be processed in some way, all the attackers have to do is to open connections up to the limit and wait until the server hits the connection timeout, which should be 300 seconds [2] according to RFC2821.

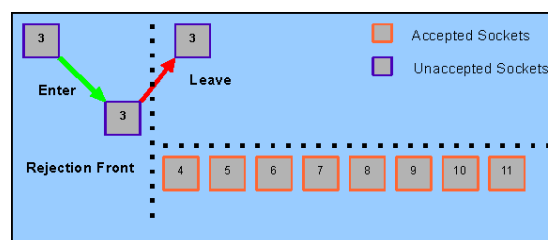


Fig 1, traditional FIFO queueing method

Sendmail for example offers the m4 configuration macro value `conf_MAX_DAEMON_CHILDREN` for exactly this purpose. The problem here is that attackers can block all your connections if the server hits the limit, and thus blocking your smtp server completely for some time period.

The `acct_smtp(9)` kernel module prevents an smtp daemon like sendmail or postfix from receiving the connected descriptor via `accept(2)`, if the data from the client is incomplete or wrong, or if commands are received before the SMTP greeting banner has been sent from the server side. Therefore `acct_smtp` can protect smtp-server daemons against denial-of-service attacks. A further benefit of `acct_smtp` is such that a server will not have to context switch several times before performing the initial parsing of the smtp HELO/EHLO request. This effectively reduces the amount of required CPU utilization to handle incoming requests by keeping active processes in forking servers such as Sendmail low and reducing the size of the file descriptor set that needs to be managed by interfaces such as `select(2)`, `poll(2)` or `kevent(2)` based servers. The

accf_smtp kernel module option is also a module that can be enabled at runtime via `kldload(8)` if the `INET` option has been compiled into the kernel.

Normal connections are directly handed over to the mail transfer agent in the underlying operating system as usual without a filter in place. The concept used here is similar to other existing accept filters like `accf_http(9)` [3] for the `http` protocol. Fig.2 shows the flow of the connection with the `accf_smtp` accept filter in place.

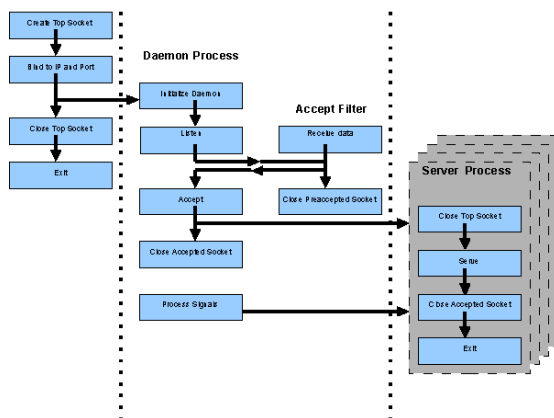


Fig. 2, Concurrent smtp server with `accf_smtp`

The `accf_smtp` accept filter itself uses and needs enlarged listen queues to keep the connections open, but uses a different overflow mechanism to prevent such attacks in contrary to the FIFO method. The method used with an activated accept filter is FILO (First In, Last Out), see Fig. 3. It shows a sample daemon listen queue within an operating system with a `smtp` accept filter in place.

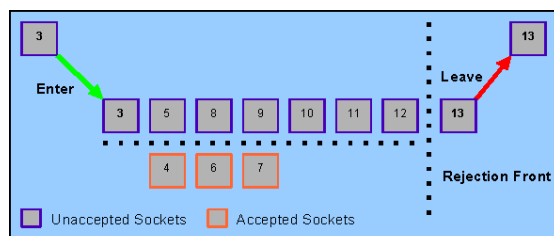


Fig. 3, FILO queuing method

If the greetpause feature is activated, the connections are sleeping for a predefined

interval in the kernel. Of course it is also possible to implement such a greetpause in userland, like `sendmail` currently does, but this has undesired sideeffects. The higher the greetpause timeout is set, the longer the connection queue from `sendmail` gets, and the more probable a DDoS situation is reached. This is because `sendmail` is process based, not thread based. Of course one could solve this with a threaded MTA, which could use internally thousands of threads, but a threaded MTA has other problems. A single failing instance (Sig 11) can crash all other threads. If the greetpause happens in the kernel, the application doesn't have to care about thousands of sleeping processes. A mail gateway with this SMTP-delay concept can handle a lot more SMTP traffic and is immutable against direct attacks originating from big botnets. Because the socket memory footprint in the OS-Kernel is a lot smaller than it is in userland, such a system can handle much more traffic until the mail gateway hits some connection limits for suspicious connections. And the system does save CPU-context switches because it doesn't have to forward all connections directly to the mailserver application in userland. The benefit of this concept is to save CPU resources, ensure system stability and to allow the user to consolidate two or more mail gateways down to only one server.

3. Configuration

Assuming `ACCEPT_FILTER_SMTP` has been included in the kernel config file or the `accf_smtp` module has been loaded, this will enable the `smtp` accept filter on the socket so:

```
struct accept_filter_arg afa;
bzero(&afa, sizeof(afa));
strcpy(afa.af_name, "smtp prefilter");
setsockopt(so, SOL_SOCKET, SO_ACCEPTFILTER, &afa, sizeof(afa));
```

The kernel module also offers number of global parameters which can be set using `sysctl(8)`:

`net.inet.accf.smtp.needhelo`

The SMTP command `HELO` is required to accept the connection. If another command is

sent first, the connection is rejected. Enabled by default.

net.inet.accf.smtp.tempfail

If set, all connections are rejected with a tempfail message. Disabled by default.

net.inet.accf.smtp.allowfrom

If set, the SMTP command FROM is accepted too as first command. Disabled by default.

net.inet.accf.smtp.greetpause

If the greetpause sysctl value is set to a non zero value, the greeting banner is delayed by value milliseconds. All incoming connections are rejected if the client sends something before the kernel sends the greeting banner to the client. The greetpause setting is ignored if tempfail mode is activated. Enabled by default and set to 500 ms.

net.inet.accf.smtp.linemaxhelo

The maximum line length of a HELO command The default is 262 chars.

net.inet.accf.smtp.linemaxfrom

The maximum line length of a FROM command The default is 334 chars.

net.inet.accf.smtp.greetmessage

SMTP-error return code 220: A faked greeting banner the kernel sends to the client side. A useful default is set at kernel module loading time.

net.inet.accf.smtp.closemessage

SMTP-error return code 221: A close banner the kernel sends to the client side. A useful default is set at kernel module load time.

net.inet.accf.smtp.tmpfmessage

SMTP-error return code 421: The SMTP-error the server sends if the tempfail mode is activated. Useful for server maintenance downtime.

net.inet.accf.smtp.syntaxerror

SMTP-error return code 500: The SMTP-error the server sends if the command was not unrecognized.

net.inet.accf.smtp.maxhelomessage

SMTP-error return code 500: The SMTP-error the server sends if the HELO command exceeds its length limits.

net.inet.accf.smtp.maxfrommessage

SMTP-error return code 500: The SMTP-error the server sends if the FROM command exceeds its length limits.

net.inet.accf.smtp.helomessage

SMTP-error return code 503: The SMTP-error the server sends to the client if the FROM command has been called before HELO and the needhelo check is activated. A useful default is set at kernel module load time.

net.inet.accf.smtp.pregreeting

SMTP-error return code 554: The pregreeting error the server sends to the client if pregreeting traffic was received. A useful default is set at kernel module load time.

4. Side effects

Implementing the greetpause feature hasn't been easy. The IP-Stack or an integrated firewall can shut down or drop existing connections. To prevent any bad side effects like crashes or deadlocks, the accept filter needs to be informed about such changes, most notably if a socket connection is sleeping in the accept filter greetpause syscall callout(9) [5] when the dropping happens.

Also important: the kernel keeps idle connections open until there are a certain number of them and closes then the oldest of them. The number permitted depends on the backlog parameter passed to listen(). This behavior conflicts with the greetpause feature of accf_smtp until you manually adjust the

number to some sensible default in the MTA application. To prevent any abuse of a smtp-server, one should set up a packet filter like pf(4) and set the options max-src-conn and max-src-conn-rate:

```
table <deny_smtp> persist
```

```
pass in on $ext_if proto tcp from !<deny_smtp> to ($ext_if) port \
smtp flags S/SA keep state (source-track rule max-src-nodes 5000 \
max-src-conn 20 max-src-conn-rate 100/10 overload <deny_smtp> \
flush global)
```

It is also a good idea to use lower tcp timeouts for busy smtp servers, to reduce the connections in the *FIN_WAIT_2* . There is no need to keep smtp connections open after the remote smtp servers have closed the connection.

```
set timeout tcp.first 300 # default is 600t
set timeout tcp.established 86400 # set lower, just one day
set timeout tcp.closing 300 # default is 600
set timeout tcp.closed 10 # set very low for smtp servers
set timeout adaptive.start 10000
set timeout adaptive.end 50000
```

Another needed change is the removal of the original greeting message of an smtp daemon. There are patches needed for any smtp server using the accf_smtp kernel module to skip those greeting messages and have them replaced by the kernel greeting message support.

5. Examples

A SMTP-server with an activated accf_smtp kernel module acts similar to a SMTP proxy. To be able to see which parts of the communication belong to the kernel or to the smtp server dialog, I've used colours to mark the log:

Text	Local communication (sended)
Text	Kernel communication (received)
Text	Userland communication (received)

A successful connection attempt with telnet looks like the following example. Only in this case the connection is handed over to the userland daemon:

```
$ telnet server 25
Trying 192.168.1.1
Connected to server.tld.ch.
Escape character is '^]'.
220 ESMTP Server ready
HELO localhost
250 server.tld.ch- Hello localhost [192.168.1.1], pleased to meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-8BITMIME
250-SIZE 20000000
250-DSN
250-ETRN
250-AUTH GSSAPI DIGEST-MD5 CRAM-MD5
250-STARTTLS
250-DELIVERBY
250 HELP
```

Failed connections look like the following examples. There is no userland part at all. All those magenta coloured return messages are from the kernel module accf_smtp.

No HELO command, just a FROM received:

```
$ telnet server 25
Trying 192.168.1.1
Connected to server.tld.ch.
Escape character is '^]'.
220 ESMTP Server ready
MAIL FROM:<spammer@spammer.ch>
503 ESMTP EHLO or HELO command required before MAIL command
221 ESMTP Server closing transmission channel
Connection closed by foreign host.
```

Junk instead of HELO command received:

```
$ telnet server 25
Trying 192.168.1.1
Connected to server.tld.ch.
Escape character is '^]'.
220 ESMTP Server ready
junkjunk
503 ESMTP Syntax error, command not recognized
221 ESMTP Server closing transmission channel
Connection closed by foreign host.
```

No waiting for the greeting message:

```
$ telnet server 25
Trying 192.168.1.1
Connected to server.tld.ch.
Escape character is '^]'.
HELO localhost
554 ESMTP Server rejected connection due to pre-greeting traffic
221 ESMTP Server closing transmission channel
Connection closed by foreign host.
```

6. Availability

The current unreleased implementation of `accf_smtp` is based on FreeBSD 7. The plan is to commit it to FreeBSD 9. Unfortunately the kernel accept filter interface has been changed recently, so this needs to be solved before any commit can happen.

References

[1] The accept filter concept was pioneered by David Filo at Yahoo! and refined to be a loadable module system by Alfred Perlstein. The accept filter concept has been published June 25, 2000 to the FreeBSD UNIX operating system.

[2] RFC2821 - Simple Mail Transfer Protocol, chapter 4.5.3.2, SMTP Timeouts.

[3] `accf_http`, developed by Alfred Perlstein, first published to FreeBSD 4.

[4] D. Hartmeier. Design and performance of the `openbsd` stateful packet filter (`pf`). In Proceedings of the USENIX 2002 Annual Technical Conference, Freenix Track, pages 171–180, 2002.

[5] Adam M. Costello and George Varghese. Redesigning the BSD Callout and Timer Facilities, Washington University Department of Computer Science, 2. November 1995