For extraction of firmware:

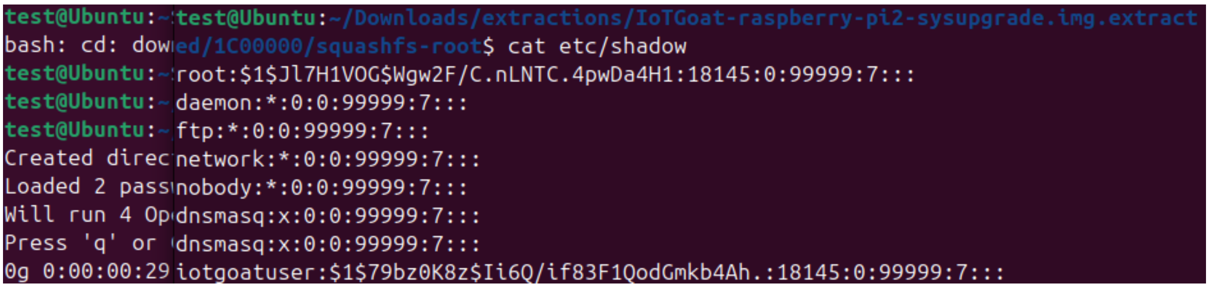
sudo docker run -t -v "$PWD":/analysis binwalkv3 -Me IoTGoat-raspberry-pi2-sysupgrade.img

Credentials

Root credentials:

The command “cat etc/shadow” gave us:

- root:$1$Jl7H1VOG$Wgw2F/C.nLNTC.4pwDa4H1:18145:0:99999:7:::
- iotgoatuser:$1$79bz0K8z$Ii6Q/if83F1QodGmkb4Ah.:18145:0:99999:7:::



```
test@Ubuntu:~test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extract
bash: cd: dowed/1C00000/squashfs-root$ cat etc/shadow
test@Ubuntu:~root:$1$Jl7H1VOG$Wgw2F/C.nLNTC.4pwDa4H1:18145:0:99999:7:::
test@Ubuntu:~ daemon:*:0:0:99999:7:::
test@Ubuntu:~ ftp:*:0:0:99999:7:::
Created direcnetwork:*:0:0:99999:7:::
Loaded 2 passnobody:*:0:0:99999:7:::
Will run 4 Opdnsmasq:x:0:0:99999:7:::
Press 'q' or dnsmasq:x:0:0:99999:7:::
0g 0:00:00:29 iotgoatuser:$1$79bz0K8z$Ii6Q/if83F1QodGmkb4Ah.:18145:0:99999:7:::
```

Decryption:

We used John The Ripper to decrypt the password for “root” and “iotgoatuser” with the following command: “john hashes.txt”

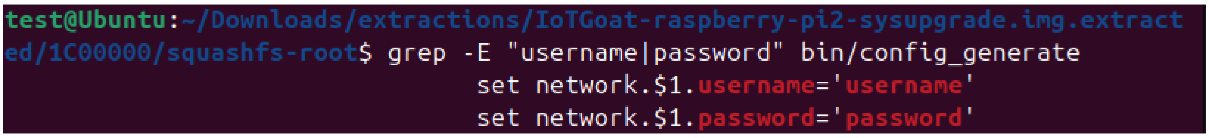
We also downloaded the well-known wordlist “rockyou.txt” to faster decrypt the passwords.

Command we used: “john --wordlist=rockyou.txt hashes.txt”

Network Credentials:

Standard credentials for network, with the command “grep -E "username|password" bin/config_generate” gave us the following:

- set network.$1.username='username'
- set network.$1.password='password'

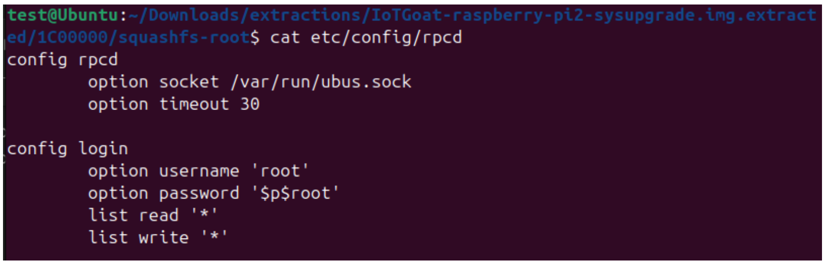


```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extract
ed/1C00000/squashfs-root$ grep -E "username|password" bin/config_generate
                                set network.$1.username='username'
                                set network.$1.password='password'
```

RPC Password:

Standard location for RPC passwords are etc/config/rpcd, therefore we ran the command “cat etc/config/rpcd” and that gave us this:

- option username 'root'
- option password '$p$root'



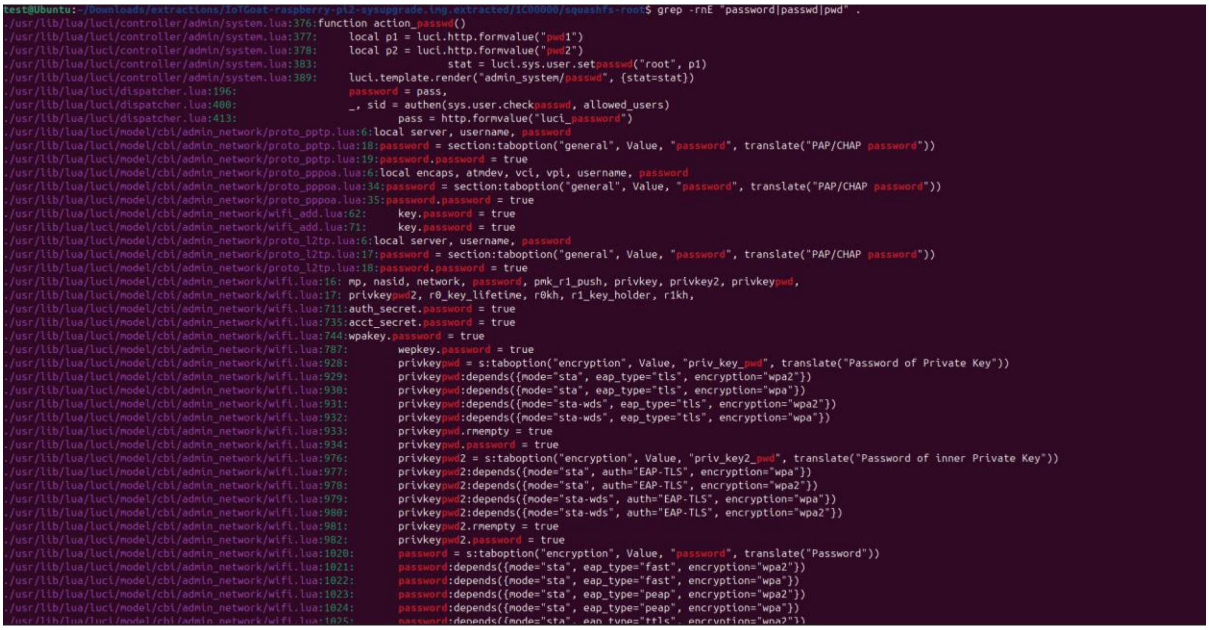
```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extract
ed/1C00000/squashfs-root$ cat etc/config/rpcd
config rpcd
        option socket /var/run/ubus.sock
        option timeout 30

config login
        option username 'root'
        option password '$p$root'
        list read '*'
        list write '*'
```

Search for hidden passwords:

We also performed a grep search for common passwords using the command “grep -rnE "password|passwd|pwd" .”, which gave us some important information:

- Hardcoded network password as mentioned before in bin/config_generate.
- Hardcoded RPC password as mentioned before in /etc/config/rpcd

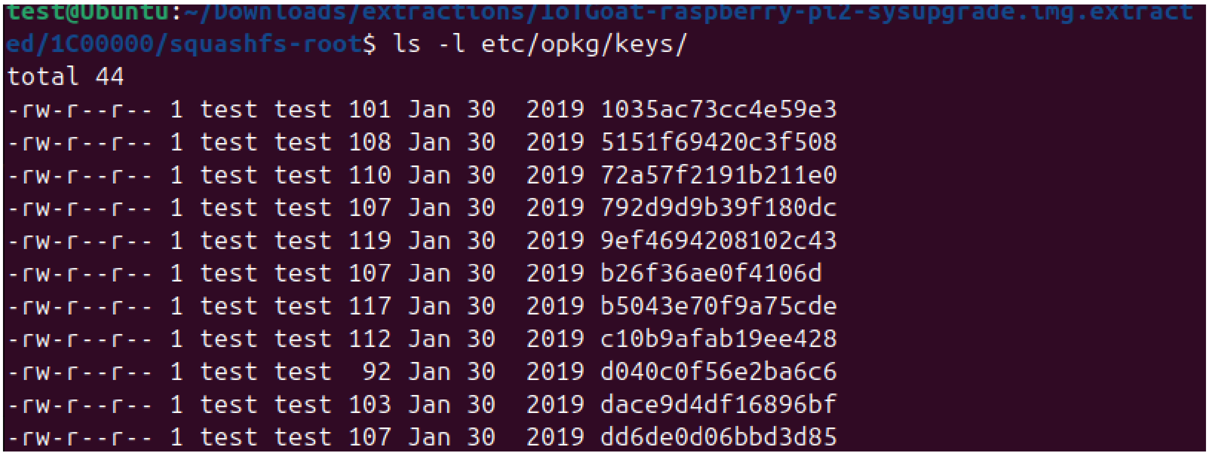


```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extracted/1C00000/squashfs-root$ grep -rnE "password|passwd|pwd" .
./usr/lib/lua/luci/controller/admin/system.lua:376:function action_passwd()
./usr/lib/lua/luci/controller/admin/system.lua:377:     local p1 = luci.http.formvalue("pwd1")
./usr/lib/lua/luci/controller/admin/system.lua:378:     local p2 = luci.http.formvalue("pwd2")
./usr/lib/lua/luci/controller/admin/system.lua:383:                     stat = luci.sys.user.setpasswd("root", p1)
./usr/lib/lua/luci/controller/admin/system.lua:389:     luci.template.render("admin_system/passwd", {stat=stat})
./usr/lib/lua/luci/dispatcher.lua:196:                  password = pass,
./usr/lib/lua/luci/dispatcher.lua:400:                  _, sid = authen(sys.user.checkpasswd, allowed_users)
./usr/lib/lua/luci/dispatcher.lua:413:                          pass = http.formvalue("luci_password")
./usr/lib/lua/luci/model/cbi/admin_network/proto_pptp.lua:6:local server, username, password
./usr/lib/lua/luci/model/cbi/admin_network/proto_pptp.lua:18:password = section:taboption("general", Value, "password", translate("PAP/CHAP password"))
./usr/lib/lua/luci/model/cbi/admin_network/proto_pptp.lua:19:password.password = true
./usr/lib/lua/luci/model/cbi/admin_network/proto_pppoa.lua:6:local encaps, atmdev, vci, vpi, username, password
./usr/lib/lua/luci/model/cbi/admin_network/proto_pppoa.lua:34:password = section:taboption("general", Value, "password", translate("PAP/CHAP password"))
./usr/lib/lua/luci/model/cbi/admin_network/proto_pppoa.lua:35:password.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi_add.lua:62:     key.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi_add.lua:71:     key.password = true
./usr/lib/lua/luci/model/cbi/admin_network/proto_l2tp.lua:6:local server, username, password
./usr/lib/lua/luci/model/cbi/admin_network/proto_l2tp.lua:17:password = section:taboption("general", Value, "password", translate("PAP/CHAP password"))
./usr/lib/lua/luci/model/cbi/admin_network/proto_l2tp.lua:18:password.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:16: mp, nasid, network, password, pmk_r1_push, privkey, privkey2, privkeypwd,
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:17: privkeypwd2, r0_key_lifetime, r0kh, r1_key_holder, r1kh,
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:711:auth_secret.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:735:acct_secret.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:744:wpakey.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:787:        wepkey.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:928:        privkeypwd = s:taboption("encryption", Value, "priv_key_pwd", translate("Password of Private Key"))
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:929:        privkeypwd:depends({mode="sta", eap_type="tls", encryption="wpa2"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:930:        privkeypwd:depends({mode="sta", eap_type="tls", encryption="wpa"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:931:        privkeypwd:depends({mode="sta-wds", eap_type="tls", encryption="wpa2"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:932:        privkeypwd:depends({mode="sta-wds", eap_type="tls", encryption="wpa"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:933:        privkeypwd.rmempty = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:934:        privkeypwd.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:976:        privkeypwd2 = s:taboption("encryption", Value, "priv_key2_pwd", translate("Password of inner Private Key"))
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:977:        privkeypwd2:depends({mode="sta", auth="EAP-TLS", encryption="wpa"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:978:        privkeypwd2:depends({mode="sta", auth="EAP-TLS", encryption="wpa2"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:979:        privkeypwd2:depends({mode="sta-wds", auth="EAP-TLS", encryption="wpa"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:980:        privkeypwd2:depends({mode="sta-wds", auth="EAP-TLS", encryption="wpa2"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:981:        privkeypwd2.rmempty = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:982:        privkeypwd2.password = true
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:1020:       password = s:taboption("encryption", Value, "password", translate("Password"))
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:1021:       password:depends({mode="sta", eap_type="fast", encryption="wpa2"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:1022:       password:depends({mode="sta", eap_type="fast", encryption="wpa"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:1023:       password:depends({mode="sta", eap_type="peap", encryption="wpa2"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:1024:       password:depends({mode="sta", eap_type="peap", encryption="wpa"})
./usr/lib/lua/luci/model/cbi/admin_network/wifi.lua:1025:       password:depends({mode="sta", eap_type="ttls", encryption="wpa2"})
```

SSH Keys

Public SSH key:

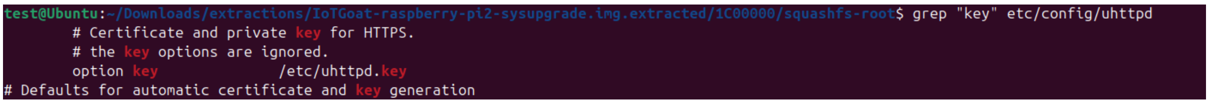
We found some public SSH keys that could be accessed. To find these we used the command “ls -l etc/opkg/keys/”



```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extract
ed/1C00000/squashfs-root$ ls -l etc/opkg/keys/
total 44
-rw-r--r-- 1 test test 101 Jan 30  2019 1035ac73cc4e59e3
-rw-r--r-- 1 test test 108 Jan 30  2019 5151f69420c3f508
-rw-r--r-- 1 test test 110 Jan 30  2019 72a57f2191b211e0
-rw-r--r-- 1 test test 107 Jan 30  2019 792d9d9b39f180dc
-rw-r--r-- 1 test test 119 Jan 30  2019 9ef4694208102c43
-rw-r--r-- 1 test test 107 Jan 30  2019 b26f36ae0f4106d
-rw-r--r-- 1 test test 117 Jan 30  2019 b5043e70f9a75cde
-rw-r--r-- 1 test test 112 Jan 30  2019 c10b9afab19ee428
-rw-r--r-- 1 test test  92 Jan 30  2019 d040c0f56e2ba6c6
-rw-r--r-- 1 test test 103 Jan 30  2019 dace9d4df16896bf
-rw-r--r-- 1 test test 107 Jan 30  2019 dd6de0d06bbd3d85
```

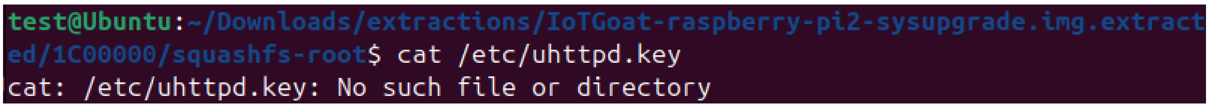
Private SSH key:

We executed the command “grep “key” etc/config/uhttpd” and found that there were private keys in /etc/uhttpd.key.



```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extracted/1C00000/squashfs-root$ grep "key" etc/config/uhttpd
        # Certificate and private key for HTTPS.
        # the key options are ignored.
        option key              /etc/uhttpd.key
# Defaults for automatic certificate and key generation
```

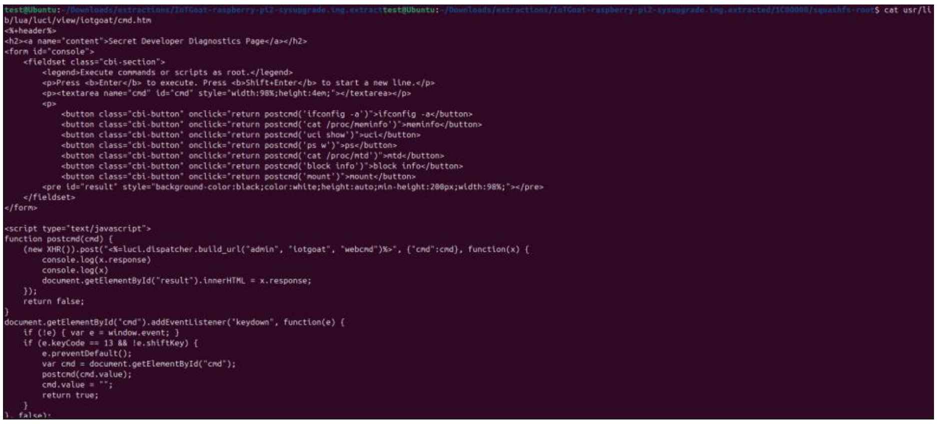
We then ran the command “cat etc/uhttpd.key” but there was no such directory for it. Therefore, we concluded that the directory uhttpd.key is not stored locally in the firmware image. It is most likely gathered from a webserver.



```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extract
ed/1C00000/squashfs-root$ cat /etc/uhttpd.key
cat: /etc/uhttpd.key: No such file or directory
```

Debug information:

For debug information, we ran the command “cat usr/lib/lua/luci/view/iotgoat/cmd.htm” and found a “Secret Developer Diagnostic Page”. This is a security risk because this page allows user to run shell commands that could comprize the whole firmware.

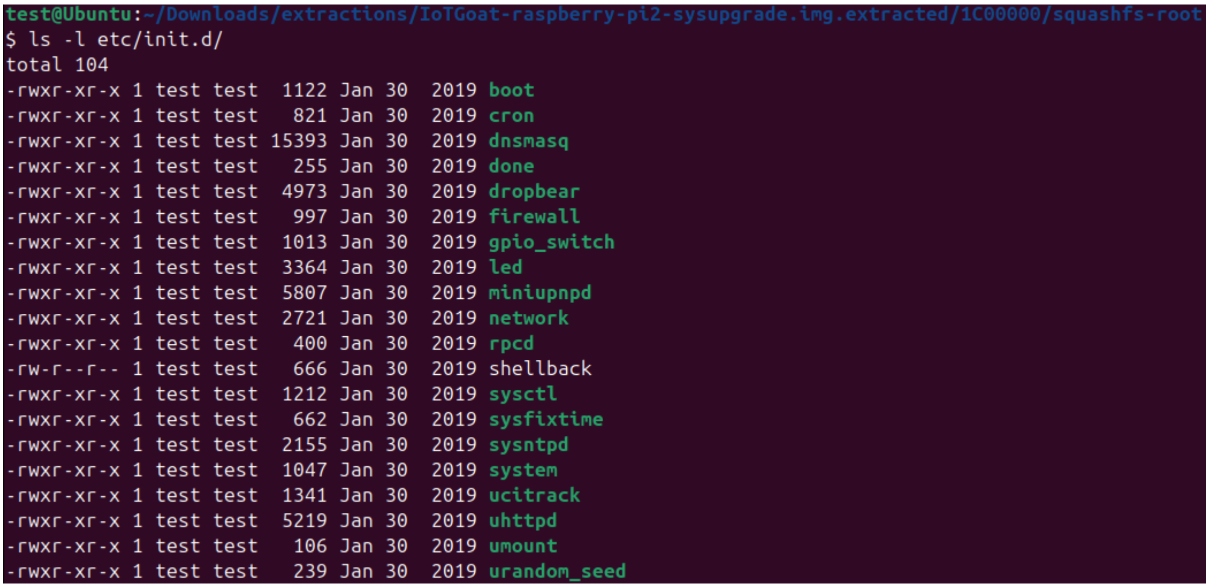


```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extracttest@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extracted/1C00000/squashfs-root$ cat usr/li
b/lua/luci/view/iotgoat/cmd.htm
<%+header%>
<h2><a name="content">Secret Developer Diagnostics Page</a></h2>
<form id="console">
    <fieldset class="cbi-section">
        <legend>Execute commands or scripts as root.</legend>
        <p>Press <b>Enter</b> to execute. Press <b>Shift+Enter</b> to start a new line.</p>
        <p><textarea name="cmd" id="cmd" style="width:98%;height:4em;"></textarea></p>
        <p>
            <button class="cbi-button" onclick="return postcmd('ifconfig -a')">ifconfig -a</button>
            <button class="cbi-button" onclick="return postcmd('cat /proc/meminfo')">meminfo</button>
            <button class="cbi-button" onclick="return postcmd('uci show')">uci</button>
            <button class="cbi-button" onclick="return postcmd('ps w')">ps</button>
            <button class="cbi-button" onclick="return postcmd('cat /proc/mtd')">mtd</button>
            <button class="cbi-button" onclick="return postcmd('block info')">block info</button>
            <button class="cbi-button" onclick="return postcmd('mount')">mount</button>
        <pre id="result" style="background-color:black;color:white;height:auto;min-height:200px;width:98%;"></pre>
    </fieldset>
</form>

<script type="text/javascript">
function postcmd(cmd) {
    (new XHR()).post("<%=luci.dispatcher.build_url("admin", "iotgoat", "webcmd")%>", {"cmd":cmd}, function(x) {
        console.log(x.response)
        console.log(x)
        document.getElementById("result").innerHTML = x.response;
    });
    return false;
}
document.getElementById("cmd").addEventListener("keydown", function(e) {
    if (!e) { var e = window.event; }
    if (e.keyCode == 13 && !e.shiftKey) {
        e.preventDefault();
        var cmd = document.getElementById("cmd");
        postcmd(cmd.value);
        cmd.value = "";
        return true;
    }
}, false);
```

Configurations & Network Services

We ran the command “ls -l etc/init.d/” and got a list of all the start scripts:



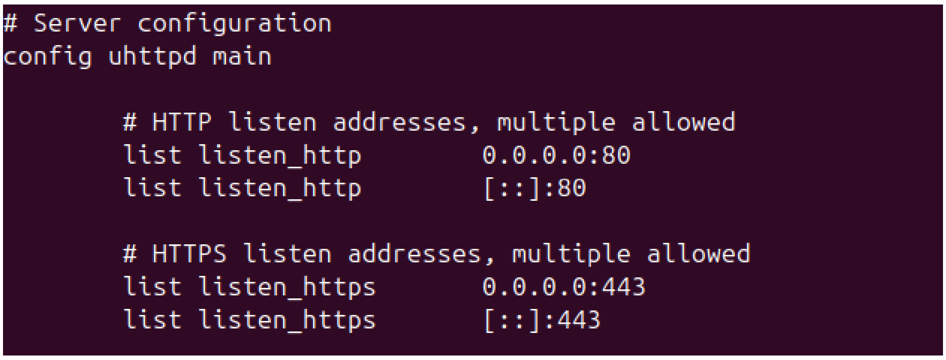
```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extracted/1C00000/squashfs-root
$ ls -l etc/init.d/
total 104
-rwxr-xr-x 1 test test  1122 Jan 30  2019 boot
-rwxr-xr-x 1 test test   821 Jan 30  2019 cron
-rwxr-xr-x 1 test test 15393 Jan 30  2019 dnsmasq
-rwxr-xr-x 1 test test   255 Jan 30  2019 done
-rwxr-xr-x 1 test test  4973 Jan 30  2019 dropbear
-rwxr-xr-x 1 test test   997 Jan 30  2019 firewall
-rwxr-xr-x 1 test test  1013 Jan 30  2019 gpio_switch
-rwxr-xr-x 1 test test  3364 Jan 30  2019 led
-rwxr-xr-x 1 test test  5807 Jan 30  2019 miniupnpd
-rwxr-xr-x 1 test test  2721 Jan 30  2019 network
-rwxr-xr-x 1 test test   400 Jan 30  2019 rpcd
-rw-r--r-- 1 test test   666 Jan 30  2019 shellback
-rwxr-xr-x 1 test test  1212 Jan 30  2019 sysctl
-rwxr-xr-x 1 test test   662 Jan 30  2019 sysfixtime
-rwxr-xr-x 1 test test  2155 Jan 30  2019 sysntpd
-rwxr-xr-x 1 test test  1047 Jan 30  2019 system
-rwxr-xr-x 1 test test  1341 Jan 30  2019 ucitrack
-rwxr-xr-x 1 test test  5219 Jan 30  2019 uhttpd
-rwxr-xr-x 1 test test   106 Jan 30  2019 umount
-rwxr-xr-x 1 test test   239 Jan 30  2019 urandom_seed
```

Some important ones for network sevices were dropbear, uhttpd and firewall.

All these files are in etc/config, therefore we ran the command “cat” on those services we thought were interesting:

- “cat etc/config/uhttpd”

Here we found two exposed ports: 80 (HTTP) and 443 (HTTPS).



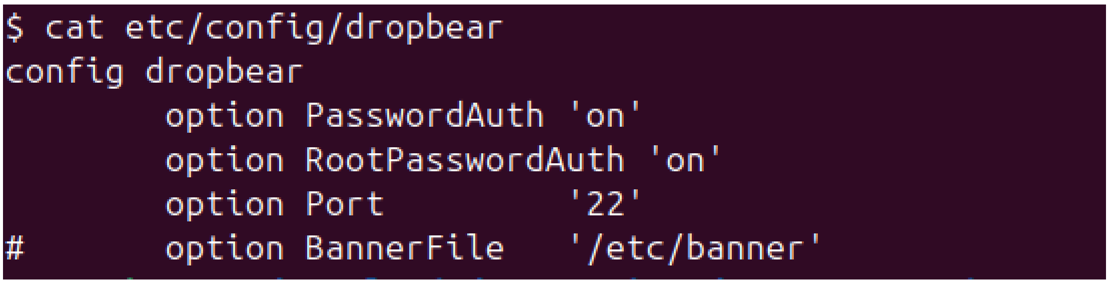
```
# Server configuration
config uhttpd main

        # HTTP listen addresses, multiple allowed
        list listen_http        0.0.0.0:80
        list listen_http        [::]:80

        # HTTPS listen addresses, multiple allowed
        list listen_https       0.0.0.0:443
        list listen_https       [::]:443
```

- "cat etc/config/dropbear"

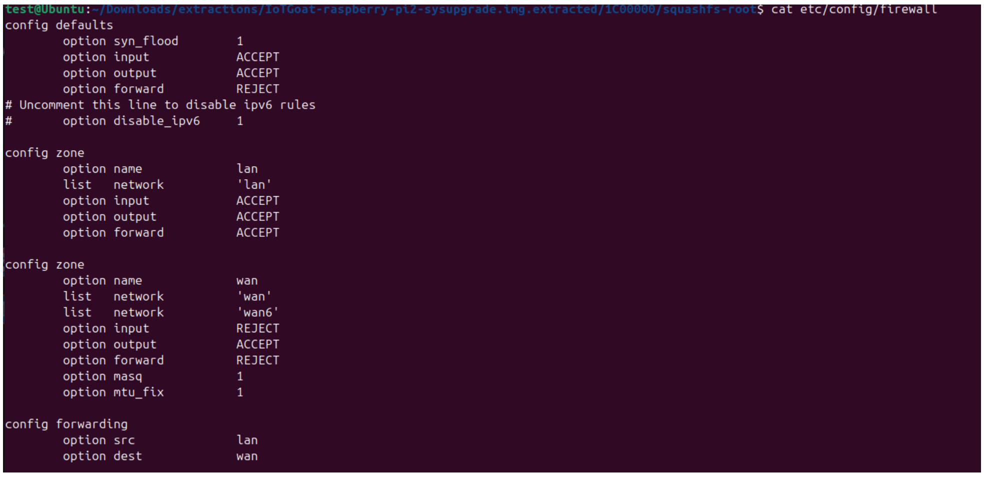
This command gave us the port for dropbear (the SSH server) which was 22, here we also noticed that RootPasswordAuth was on. We could use the root password to access the SSH server.



```
$ cat etc/config/dropbear
config dropbear
        option PasswordAuth 'on'
        option RootPasswordAuth 'on'
        option Port         '22'
#       option BannerFile   '/etc/banner'
```

- "cat etc/config/firewall"

Looking at the firewall gave us a better clue on how the network was set up in the firmware. Lan was set to accept both outputs and inputs. Meaning that all services like SSH and HTTP are unprotected for everyone that is on the local network. It is also worth noting that wan has its input set to "reject", meaning that most attacks on the network would be through the local network.



```
test@Ubuntu:~/Downloads/extractions/IoTGoat-raspberry-pi2-sysupgrade.img.extracted/1C00000/squashfs-root$ cat etc/config/firewall
config defaults
        option syn_flood        1
        option input            ACCEPT
        option output           ACCEPT
        option forward          REJECT
# Uncomment this line to disable ipv6 rules
#       option disable_ipv6     1

config zone
        option name             lan
        list   network          'lan'
        option input            ACCEPT
        option output           ACCEPT
        option forward          ACCEPT

config zone
        option name             wan
        list   network          'wan'
        list   network          'wan6'
        option input            REJECT
        option output           ACCEPT
        option forward          REJECT
        option masq             1
        option mtu_fix          1

config forwarding
        option src              lan
        option dest             wan
```