



Etude des vulnérabilités d'un équipement IoT

IOT

Auteurs :

Thinhinane ZEDDAM
Augustin LAOUAR



Avant-propos

Ce rapport a pour but de présenter de manière détaillée le travail de recherche réalisé pour notre projet d'ouverture à la recherche intitulé « Étude des vulnérabilités d'un équipement IoT ». Ce rapport a été rédigé de manière à ce qu'une personne ayant des connaissances de base en informatique puisse le comprendre. Pour faciliter cela, les termes techniques qui ne sont pas définis directement dans le rapport ont été numérotés en exposant et leurs définitions ont été fournies en annexe. Nous souhaitons exprimer nos remerciements à nos encadrants, Thomas Begin et Jean-Patrick Gelas, pour leur accompagnement et leur aide tout au long de ce projet, sans laquelle ce travail n'aurait pas été possible.

Table des matières

Avant propos.....	1
Introduction	5
Présentation de la Caméra.....	5
Analyse réseau.....	6
Installation.....	6
Scénarios	6
Volume de trafic.....	7
Flux sortant de la caméra.....	9
Débit et taille des paquets envoyés.....	9
Destinataires.....	10
Protocoles utilisés.....	11
Flux entrant de la caméra	12
Débit et taille des paquets reçus	12
Destinataires.....	13
Protocoles utilisés.....	14
Conversations.....	14
Paquets NTP	14
Conversations du scénario no-app-no-move-4G.....	15
Conversations du scénario no-app-no-move-wifi	15
Conversations du scénario no-app-move-4G.....	15
Conversations du scénario no-app-move-wifi	16
Conversations du scénario app-no-move-4G.....	17
Conversations du scénario app-no-move-wifi	17
Conversations du scénario app-move-4G.....	18
Conversations du scénario app- move-wifi.....	18
Conclusion	19
Analyse du firmware	20
Méthodologie	20
Outils utilisés	20
Accès au terminal de la caméra	21
Récupération de la mémoire.....	21
U-Boot.....	22
Accéder à U-Boot.....	22

Blocage dans U-Boot.....	22
Memory dump.....	22
Récupération du firmware	23
Obtenir une invite de commande en mode root	23
Copier la mémoire	23
Analyse du firmware.....	23
Processus.....	23
/bin/cet.....	24
usr/bin/relayd.....	25
usr/bin/rtspd.....	25
/bin/cloud-client et /bin/cloud-service	25
/usr/sbin/uhttpd.....	26
Structure du système de fichiers.....	26
Dossiers bin et sbin	26
Dossier etc.....	26
Dossier rom.....	26
Etude des vulnérabilités	27
Injection de commande	27
Injection de script.....	28
Méthode 1 : le dossier /etc/uci-defaults/	28
Méthode 2 : le dossier /etc/init.d et /etc/rc.d.....	28
Problèmes rencontrés	29
Non persistance des modifications apportés au firmware.....	29
Confiance	30
Sécurité.....	30
Transparence du fabricant.....	31
Conclusion.....	32
Annexes	33
Adresses qui reçoivent des paquets de la caméra.....	33
Adresses qui envoient des paquets à la caméra.....	36
Définitions.....	38

Introduction

L'Internet des objets (IoT) révolutionne notre monde en connectant sans cesse de nouveaux appareils à Internet. Les objets connectés sont de plus en plus intégrés dans notre vie quotidienne, que ce soit à travers des caméras de surveillance, des montres connectées, des thermostats intelligents et bien d'autres encore. Ils sont présents dans nos foyers, nos véhicules, et même sur nous-mêmes.

Cependant, cette évolution rapide pose également des problèmes de sécurité, car les appareils IoT peuvent présenter des vulnérabilités qui peuvent être exploitées par des cybercriminels pour accéder à des informations confidentielles ou perturber le fonctionnement normal des appareils. Dans ce contexte, il est crucial d'évaluer le niveau de vulnérabilité d'un équipement IoT donné afin de pouvoir le protéger efficacement contre les cyberattaques. Dans ce projet de recherche, l'objectif est d'étudier une caméra de surveillance, un objet IoT très utilisé par le grand public, pour dans un premier temps comprendre son comportement réseau, puis d'évaluer expérimentalement le niveau de vulnérabilité, à travers différentes analyses de son trafic réseau puis à travers l'extraction de son **firmware**¹ et son analyse.

Présentation de la Caméra



La caméra Tapo C200 est une caméra d'entrée de gamme à destination du grand public fabriquée par TP-Link. Elle est conçue pour permettre la surveillance des activités intérieures au sein de son domicile ou bureau. Elle possède plusieurs fonctionnalités telles que la détection de mouvement, la vision nocturne et d'une fonction haut-parleur. Elle est livrée avec une application qui permet sa configuration puis la prise de contrôle de la caméra à distance, le visionnage en temps réel des images etc.

De plus, cette dernière permet la sauvegarde locale des images à travers l'application sur son téléphone, dans une carte SD de taille maximale de 128 Go ou encore dans le Cloud (fonctionnalité payante).

La caméra Tapo c200 utilise une connexion Wi-Fi afin de communiquer avec notre réseau local et permettre l'accès à distance via internet. Cette dernière est compatible avec les protocoles de sécurité Wi-Fi WPA/WPA2-PSK² et elle prend en charge la norme de chiffrement de données WEP/WPA/WPA2³.



Enfin, TP-Link propose régulièrement des mises à jour du firmware avec des correctifs de sécurité.

Analyse réseau

I. Installation

La caméra Tapo C200 fonctionne uniquement en Wi-Fi. Il faut donc capturer son trafic Wi-Fi. Afin de capturer le trafic réseau de la caméra, une possibilité est d'écouter tout le trafic Wi-Fi (sniffer) et ne garder que les paquets désirés, dans ce cas, ceux émis ou reçus par la caméra.

Pour ce faire, il est nécessaire d'activer le mode **monitoring**⁴ sur la carte réseau d'un ordinateur situé à proximité de la caméra. Toutefois, la majorité des cartes réseaux classiques ne supportent pas ce mode, ce qui est le cas pour les cartes réseaux de nos deux ordinateurs.

Afin de contourner ce problème, nous avons utilisé une carte réseau sans fil USB supportant le mode **monitoring** (Alpha Network AWUS036NHA⁵ mise à disposition par nos encadrants). Une fois cette dernière configurée, il est possible d'écouter tout le trafic Wi-Fi pour un canal donné. Dans notre cas, la caméra communiquait sur le canal 1 (2.412 MHz).

Le logiciel **Wireshark**⁶ a été utilisée pour réaliser la capture des paquets ainsi que pour réaliser une partie des statistiques, ce dernier possédant de nombreux outils de statistiques.

II. Scénarios

Il est nécessaire d'étudier la caméra lors de différents cadres d'utilisations. Huit différents scénarios d'utilisations ont été réalisés :

- Scénario 1 (**no-app-no-move-4G**) : La caméra filme une pièce où il n'y a aucun mouvement. Pas de visionnage du flux vidéo via l'application mobile. Le téléphone est connecté en 4G.
- Scénario 2 (**no-app-no-move-wifi**) : La caméra filme une pièce où il n'y a aucun mouvement. Pas de visionnage du flux vidéo via l'application mobile. Le téléphone est connecté en Wi-Fi sur le même réseau que la caméra.
- Scénario 3 (**no-app-move-4G**) : La caméra filme une pièce où il y a du mouvement. Pas de visionnage du flux vidéo via l'application mobile. Le téléphone est connecté en 4G.
- Scénario 4 (**no-app-move-wifi**) : La caméra filme une pièce où il y a du mouvement. Pas de visionnage du flux vidéo via l'application mobile. Le téléphone est connecté en Wi-Fi sur le même réseau que la caméra.
- Scénario 5 (**app-no-move-4G**) : La caméra filme une pièce où il n'y a pas de mouvement. Visionnage du flux vidéo via l'application mobile. Le téléphone visionnant le flux est n'est pas connecté sur le même réseau que la caméra (utilisation du réseau cellulaire).
- Scénario 6 (**app-no-move-wifi**) : La caméra filme une pièce où il n'y a pas de mouvement. Visionnage du flux vidéo via l'application mobile. Le téléphone visionnant le flux est connecté sur le même réseau Wi-Fi que la caméra.
- Scénario 7 (**app-move-4G**) : La caméra filme une pièce où il y a du mouvement. Visionnage du flux vidéo via l'application mobile. Le téléphone visionnant le flux est n'est pas connecté sur le même réseau que la caméra (utilisation du réseau cellulaire).

- Scénario 8 (**app-move-wifi**) : La caméra filme une pièce où il y a du mouvement. Visionnage du flux vidéo via l'application mobile. Le téléphone visionnant le flux est connecté sur le même réseau Wi-Fi que la caméra.

Chacun de ces scénarios a été réalisé et capturé pendant 3 minutes environ.

En **rouge** : le nom donné au scénario pour une lecture plus simple des résultats.

Le tableau ci-dessous résume ces informations.

Tableau 1 Informations des scénarios

Scénario	Mouvement	Visionnage du flux	Réseau du téléphone visionnant le flux	Durée (secondes)
no-app-no-move-4G	Non	Non	Réseau cellulaire (différent de celui de la caméra).	173.059
no-app-no-move-wifi	Non	Non	Même réseau Wi-Fi que la caméra.	177.308
no-app-move-4G	Oui	Non	Réseau cellulaire (différent de celui de la caméra).	180.518
no-app-move-wifi	Oui	Non	Même réseau Wi-Fi que la caméra.	167.681
app-no-move-4G	Non	Oui	Réseau cellulaire (différent de celui de la caméra).	172.257
app-no-move-wifi	Non	Oui	Même réseau Wi-Fi que la caméra.	181.357
app-move-4G	Oui	Oui	Réseau cellulaire (différent de celui de la caméra).	178.488
app-move-wifi	Oui	Oui	Même réseau Wi-Fi que la caméra	176.853

III. Volume de trafic

Tableau 2 statistiques générales des captures

Scénario	Nombre total de paquets échangés	Durée en secondes	Taux de paquets échangés par seconde	Taille moyenne des paquets Octet	Nombre total d'octet échangés	Débit moyen (octet/s)
no-app-no-move-4G	108	173.059	0.6	56	9 748	90
no-app-no-move-wifi	70	177.308	0.4	90	6 315	35
no-app-move-4G	133	180.518	0.7	189	25 084	139
no-app-move-wifi	178	167.681	1.1	183	32 620	194
app-no-move-4G	8383	172.257	48.7	908	7 614 810	44 k
app-no-move-wifi	10983	181.357	60.6	702	7 708 220	42 k
app-move-4G	26620	178.488	149.1	1039	27 662 653	154 k
app-move-wifi	30386	176.853	171.8	1021	31 035 554	175 k

Ci-dessus, quelques statistiques que nous avons pu recueillir sur le trafic de la caméra (paquets émis ou reçus par la caméra) qui donnent une vision globale sur ce dernier.

Ci-dessous, une figure représentative du volume de trafic moyen en fonction du scénario qui permet d'avoir une vision plus claire.

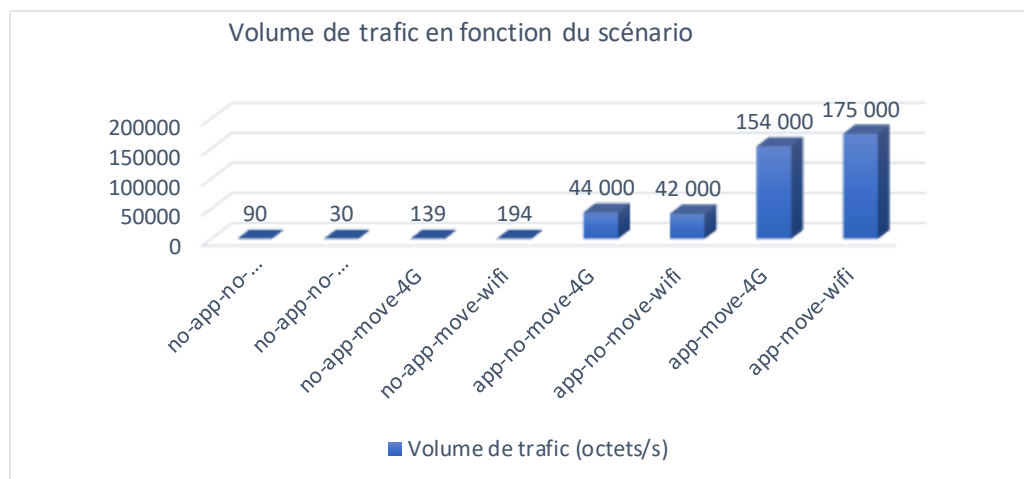


Figure 1 volume du trafic

Sur cette figure, nous pouvons constater que l'utilisation de l'application de visionnage sur le téléphone implique un trafic plus important, ce qui est justifié.

De même, ce dernier est plus important lorsqu'il y'a du mouvement que lorsqu'il y'en a pas. Enfin, la différence de trafic n'est pas significative si le téléphone est connecté sur le même réseau que la caméra (Wi-Fi) ou non (cellulaire).

Ainsi le mode de connexion n'a pas d'impact sur le trafic contrairement aux mouvements et l'utilisation de l'application qui engendrent plus de trafic.

IV. Recherche DNS

La caméra utilise le serveur DNS⁷ (Domain Name Service) de son routeur. Cela est courant dans les réseaux locaux où le routeur est configuré pour être le serveur DNS par défaut pour les appareils connectés. Cela permet d'économiser des ressources réseau en évitant des requêtes DNS redondantes vers des serveurs DNS externes.


Your DNS Servers	IP Address :	ISP :	Location :
	 89.2.3.78	NUMERICABLE	France, Paris

Figure 2 informations DNS

Cela implique que la caméra reçoit l'adresse du DNS dans la réponse DHCP⁸ de son routeur. La présence d'un serveur DHCP ainsi que d'un serveur DNS est donc nécessaire pour le bon fonctionnement de l'objet.

V. Flux sortant de la caméra

Dans cette section sera présenté l'analyse réseau concernant uniquement les paquets envoyés par la caméra à destination de son point d'accès wifi.

1. Débit et taille des paquets envoyés

Tableau 3: taille des paquets envoyés par la caméra

Scénario	Taille minimale d'un paquet (Octet)	Taille maximale d'un paquet (octet)	Taille moyenne d'un paquet (Octet)	Nombre total de paquets envoyés	Nombre total d'octets envoyés	Débit moyen (octet/s)
no-app-no-move-4G	66	126	91.29	65	5934	34.29
no-app-no-move-wifi	66	125	91,43	42	3 840	21.66
no-app-move-4G	54	533	127.72	74	9451.28	52.36
no-app-move-wifi	54	533	112,89	114	12 869	76.74
app-no-move-4G	54	1514	1384,50	5346	7 401 537	42 967.99
app-no-move-wifi	62	1514	939,61	7896	7 419 161	40 909.15
app-move-4G	54	1514	1471,37	18414	27 093 807	151 796.24
app-move-wifi	54	1514	1282,07	23790	30 500 445	172 462.13

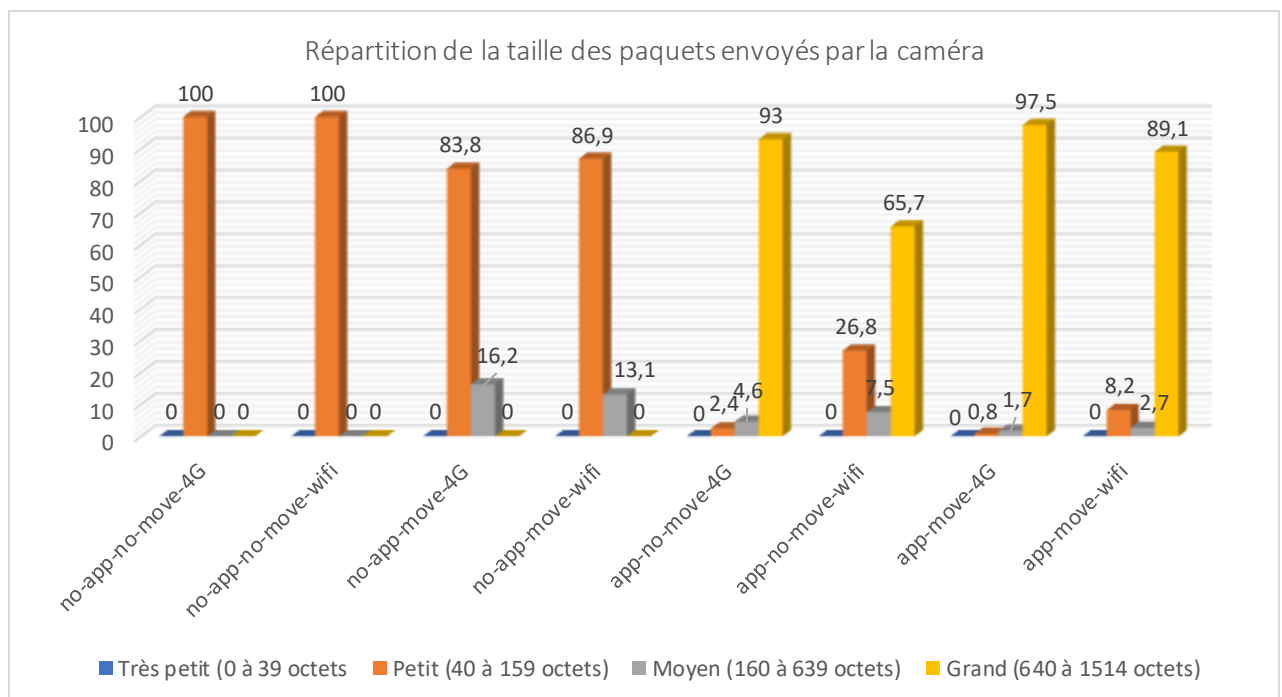


Figure 3 Répartition de la taille des paquets envoyés par la caméra

Comme remarqué précédemment, il existe **une différence de débit importante entre les scénarios utilisant l'application ou pas**. La taille moyenne des paquets pour les scénarios utilisant l'application laisse supposer que les paquets sont majoritairement des paquets vidéo. En parallèle, Les scénarios n'utilisant pas l'application n'envoient que des petits paquets s'il n'y a pas de mouvement et des petits ou moyens paquets s'il y a du mouvement.

Dans les scénarios avec du mouvement où l'application n'est pas utilisée, l'envoi de paquets de tailles moyennes s'explique par le fait que la caméra envoie une notification à l'application mobile à laquelle elle est associée lorsqu'elle détecte du mouvement.

En considérant les scénarios avec mouvement et avec application, lorsque le téléphone est sur le même réseau que la caméra nous relevons que cette dernière envoie plus de paquet que dans les scénarios où le téléphone est connecté au réseau cellulaire.

Cependant, les paquets sont en moyenne plus petit lorsque le téléphone est sur le même réseau. D'après [le tableau 3](#), cette différence provient du nombre important de petits paquets envoyés dans les scénarios Wi-Fi contrairement à ceux avec réseau cellulaire ou leur nombre est très faible voir nul.

2. Destinataires

Durant son utilisation, la caméra envoie des données à un grand nombre d'adresses différentes. Cependant, la grande majorité des données sont envoyées à, une ou deux adresses. Afin de ne pas surcharger cette partie, nous présenterons uniquement l'adresse à laquelle la caméra a envoyé le plus de données. L'intégralité des adresses de destinations est disponible dans la rubrique [annexes](#).

Nous soulignons que les échanges se font en IPv4⁹.

Tableau 4 Destinataires des paquets envoyés par la caméra

Scénario	Destinataire principale	Pourcentage	Propriétaire de l'adresse de destination
no-app-no-move-4G	34.242.49.166	18.46 %	Serveur Amazon Data Service (Dublin, Ireland)
no-app-no-move-wifi	46.51.165.123	21.43 %	Serveur Amazon Web Service (Dublin, Ireland)
no-app-move-4G	52.213.90.205	55.41%	Serveur Amazon Data Service (Dublin, Ireland)
no-app-move-wifi	52.213.90.205	38.35 %	Serveur Amazon Data Service (Dublin, Ireland)
app-no-move-4G	52.210.196.93	98.75 %	Serveur Amazon Data Service (Dublin, Ireland)
app-no-move-wifi	192.168.1.185	92.98 %	Téléphone connecté à l'application
app-move-4G	52.210.196.93	99.44 %	Serveur Amazon Data Service (Dublin, Ireland)
app-move-wifi	192.168.1.185	82.99 %	Téléphone connecté à l'application

Lorsque l'application n'est pas utilisée la majorité des paquets vont sur les serveurs Amazon, certainement loué par TP-Link.

Lorsque l'application est utilisée, cela dépend du réseau sur lequel est connecté le téléphone l'utilisateur. S'il est sur le même réseau que la caméra, **la caméra lui envoie directement les paquets de flux vidéo**. Sinon, **le flux vidéo passent d'abord par les serveurs Amazon avant d'être envoyé au téléphone**.

3. Protocoles utilisés

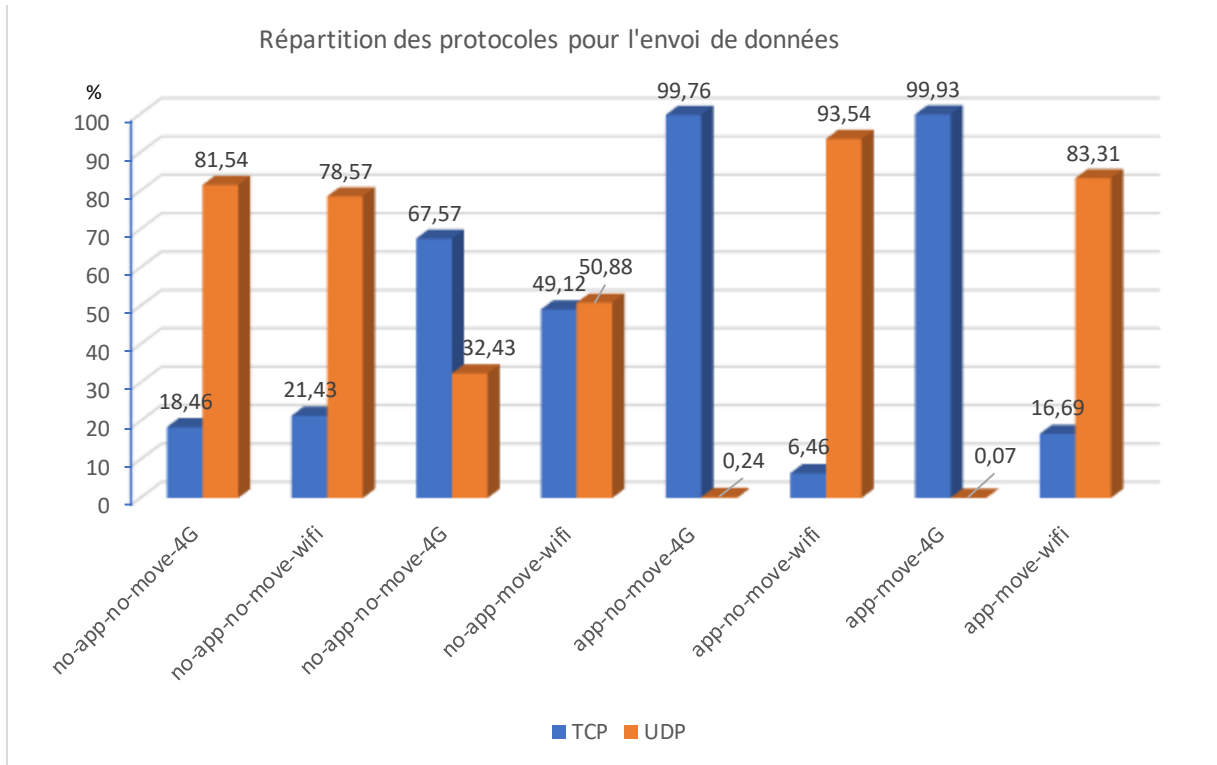


Figure 4 Répartition des protocoles pour l'envoi de données

Lorsqu'il n'y a pas de mouvements et que l'application n'est pas utilisée, la majorité des paquets utilisent le protocole **UDP**¹⁰. Cela peut s'expliquer par le grand nombre de requêtes **NTP**¹¹ (Network Time Protocol, un protocole de synchronisation d'horloge) que fait la caméra, que l'on abordera plus tard dans la section conversation.

Il a été observé que lorsque le téléphone est connecté au même réseau Wi-Fi que la caméra, **la majorité des paquets sont envoyés avec le protocole UDP**. Dans le cas contraire, ils sont **majoritairement envoyés avec TCP**¹². Cette différence est d'autant plus marquée lorsque le téléphone utilise l'application. En conclusion, lorsque la caméra envoie le flux vidéo à l'application mobile et que le téléphone est sur le même réseau que la caméra, elle utilise majoritairement **UDP**. Dans le cas où le téléphone n'est pas sur le même réseau Wi-Fi, elle envoie le flux vidéo au serveur Amazon et utilise majoritairement **TCP**.

Il convient de souligner que, même lors de la transmission directement par le Wi-Fi du flux vidéo de la caméra vers le téléphone, un volume important de paquets **TCP** à destination des serveurs Amazon est toujours impliqué dans l'échange de données. On peut donc supposer que même si la caméra envoie le flux vidéo directement au téléphone, une partie de la vidéo pourrait également être envoyée aux serveurs d'Amazon.

VI. Flux entrant de la caméra

Dans cette section sera présenté l'analyse réseau concernant uniquement les paquets envoyés par la caméra.

1. Débit et taille des paquets reçus

Tableau 5 : Taille des paquets reçus par la caméra

Scénario	Taille minimale d'un paquet (octet)	Taille maximale d'un paquet (octet)	Taille moyenne d'un paquet (octet)	Nombre total de paquets envoyés	Nombre total d'octets envoyés	Débit moyen (octet/s)
no-app-no-move-4G	66	124	88.70	43	3 814	22.04
no-app-no-move-wifi	66	123	88.39	28	2 475	13.95
no-app-move-4G	66	1514	264.97	59	15 633	86.60
no-app-move-wifi	66	1514	287.83	71	20 436	121.87
app-no-move-4G	66	1514	70.22	3037	213 258	1 238.02
app-no-move-wifi	62	1514	93.74	3103	290 875	1 603.88
app-move-4G	66	1514	66.31	8206	544 138	3 048.60
app-move-wifi	62	1514	81.21	6615	537 204	3037.57

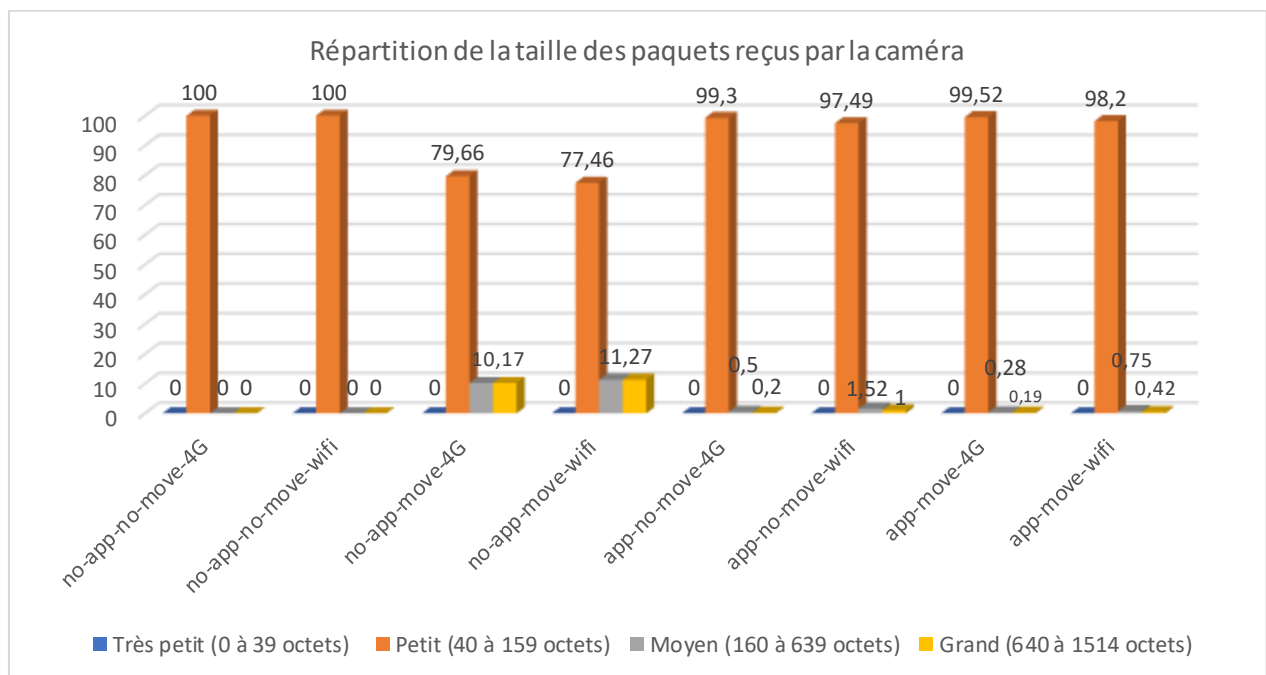


Figure 5: Répartition de la taille des paquets reçus par la caméra

Le flux entrant présente des caractéristiques similaires à celles du flux sortant en termes de débit moyen et de taille de paquets.

Toutefois, dans les scénarios no-app-move-4G et no-app-move-wifi, on observe une taille moyenne de paquet anormalement élevée. Cette anomalie est causée par les paquets "server hello" du protocole TLS (Transport Layer Security) qui ont une taille de 1514 octets. Ces paquets sont utilisés pour établir une connexion sécurisée avec le serveur.

2. Destinataires

De même que dans la section "Flux sortant de la caméra", afin de ne pas alourdir cette partie, seule l'adresse qui a envoyé le plus de données à la caméra sera présentée. Toutes les adresses sont disponibles dans la section [annexes](#).

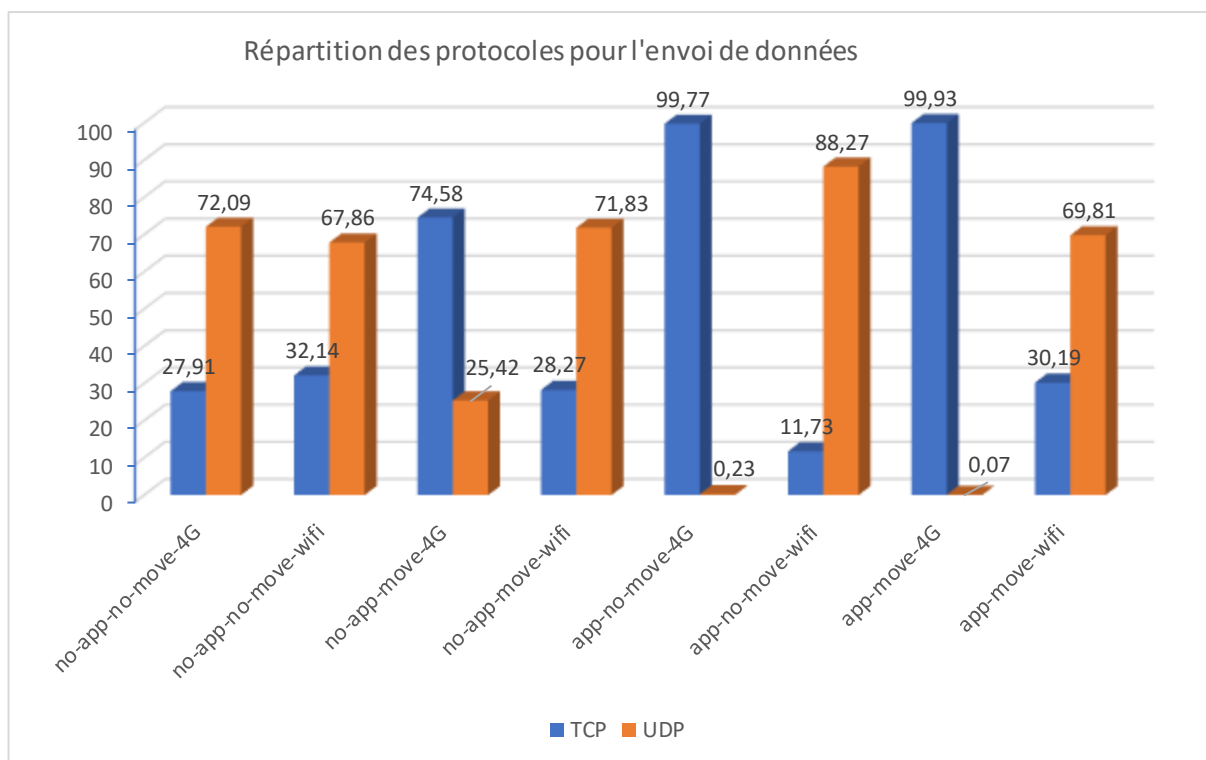
Tableau 6: Sources des paquets reçus par la caméra

Scénario	Source principale	Pourcentage	Propriétaire de l'adresse de destination
no-app-no-move-4G	34.242.49.166	27.91%	Serveur Amazon Data Service (Dublin, Ireland)
no-app-no-move-wifi	46.51.165.123	32.14 %	Serveur Amazon Web Service (Dublin, Ireland)
no-app-move-4G	52.213.90.205	59.32%	Serveur Amazon Data Service (Dublin, Ireland)
no-app-move-wifi	52.213.90.205	53.12 %	Serveur Amazon Data Service (Dublin, Ireland)
app-no-move-4G	52.210.196.93	98.06 %	Serveur Amazon Data Service (Dublin, Ireland)
app-no-move-wifi	192.168.1.185	87.66 %	Téléphone connecté à l'application
app-move-4G	52.210.196.93	98.95 %	Serveur Amazon Data Service (Dublin, Ireland)
app-move-wifi	192.168.1.185	69.39 %	Téléphone connecté à l'application

Nous pouvons souligner qu'il existe une cohérence entre le flux entrant et le flux sortant à travers l'analyse de ce tableau et sa comparaison avec [le tableau 5](#) du flux sortant. En effet, nous retrouvons les mêmes adresses et des pourcentages cohérents.

3. Protocoles utilisés

Figure 3: Répartition des protocoles pour l'envoi de données



Résultat cohérent avec ce qui a été analysé dans le flux sortant de la caméra.

VII. Conversations

Dans cette section seront présentées les principales conversations de la caméra, en fonction du scénario. Pour chacun des scénarios seront présentés uniquement les conversations jugées utiles à l'analyse du réseau. Leur totalité est disponible en annexe.

Paquets NTP

Un grand nombre de conversations concerne uniquement des requêtes NTP que la caméra a envoyé à différents serveurs NTP. Certains scénarios contiennent plus de 50 requêtes NTP à des serveurs différents. Pour un souci de lisibilité, nous n'avons pas mis ces conversations dans cette partie. L'intégralité des conversations de la caméra en fonction des scénarios est disponible en [annexe](#).

Conversations du scénario no-app-no-move-4G

Port caméra	Interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (Octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (Seconde)	Volume de trafic moyen interlocuteur vers caméra (Octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (Seconde)
34014	34.242.49.166 (Amazon Data Services Ireland Limited)	443	TCP	166.404	8.23	13.87	6	8.23

La caméra échange simplement avec un serveur TP-Link (loué chez Amazon) tout au long de la capture

Conversations du scénario no-app-no-move-wifi

Port caméra	Interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (Octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (Seconde)	Volume de trafic moyen interlocuteur vers caméra (Octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (Seconde)
60698	46.51.165.123 (Ireland Dublin Amazon Web Services Elastic Compute Cloud Ec2 Eu)	443	TCP	112.7	7.8	12.5	6.8	12.5

Conversations du scénario no-app-move-4G

Port caméra	Interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (Octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (Seconde)	Volume de trafic moyen interlocuteur vers caméra (Octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (Seconde)
42777	52.19.71.215 (Amazon Data Services Ireland Limited)	443	TCP	110.996	7.865	12.2	6.91	12.2
59168	52.213.90.205 (Amazon Data Services Ireland Limited)	443	TCP	1.326	1588.99	10.55	3334	9.04
59169	52.213.90.205 (Amazon Data Services Ireland Limited)	443	TCP	1.328	1545.93	9.78	3280	8.28
59170	52.213.90.205 (Amazon Data Services Ireland Limited)	443	TCP	1.325	1590.19	10.56	3336	9.05

Conversations du scénario no-app-move-wifi

Port caméra	Adresse IP interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (Octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (Seconde)	Volume de trafic moyen interlocuteur vers caméra (Octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (Seconde)
60698	46.51.165.123 (Ireland Dublin Amazon Web Services Elastic Compute Cloud Ec2 Eu)	443	TCP	111	7.8	12.3	6.8	12.3
56623	52.18.5.103 (Amazon Data Services Ireland Limited)	443	TCP	0.9	1143.3	0.1125	4634.4	0.1125
58759	52.213.90.205 (Amazon Data Services Ireland Limited)	443	TCP	1.33	1536	0.1	3275.2	0.12
58759	52.213.90.205 (Amazon Data Services Ireland Limited)	443	TCP	1.32	1536	0.1	1536	0.12
58759	52.213.90.205 (Amazon Data Services Ireland Limited)	443	TCP	1.27	1708.6	0.08	3481.1	0.1
17271	3.248.99.37 (Amazon Data Services Ireland Limited)	3478	UDP	1.48	831	0.08	0	-
17270	77.196.128.168 (France Lyon Dynamic Pools sfr)	65558	UDP	1.25	392	0.18	0	-

Conversations du scénario app-no-move-4G

Port caméra	Adresse IP interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (seconde)	Volume de trafic moyen interlocuteur vers caméra (octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (seconde)
60698	46.51.165.123 (Ireland Dublin Amazon Web Services Elastic Compute Cloud Ec2 Eu)	443	TCP	152.47	70.28	4.6	36.77	4.6
43599	52.210.196.93 (Ireland Dublin Amazon Data Services Ireland Limited)	443	TCP	172.25	40 890	0.033	1119.45	0.057

Conversations du scénario app-no-move-wifi

Port caméra	Adresse IP interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (Octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (Seconde)	Volume de trafic moyen interlocuteur vers caméra (Octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (Seconde)
60698	46.51.165.123 (Ireland Dublin Amazon Web Services Elastic Compute Cloud Ec2 Eu)	443	TCP	169.81	127.72	2.35	72.4	2.46
43539	52.210.196.93 (Ireland Dublin Amazon Data Services Ireland Limited)	443	TCP	12.91	36 128	0.036	1412	0.06
39388	192.168.1.185	53603	UDP	64.37	38 418	0.023	1226.3	0.063
45277	192.168.1.185	57359	UDP	67.75	39 808	0.023	1237	0.063
48891	192.168.1.185	55648	UDP	10.76	36 288	0.025	1226	0.067
56265	192.168.1.185	51861	UDP	28.22	36 534	0.024	1267	0.062

Conversations du scénario app-move-4G

Port caméra	Adresse IP interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (Octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (Seconde)	Volume de trafic moyen interlocuteur vers caméra (Octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (Seconde)
60698	46.51.165.123 (Ireland Dublin Amazon Web Services Elastic Compute Cloud Ec2 Eu)	443	TCP	171.42	108.35	3.36	53.76	3.65
43608	52.210.196.93 (Ireland Dublin Amazon Data Services Ireland Limited)	443	TCP	178.24	144 816	0.0097	2962	0.022

Conversations du scénario app- move-wifi

Port caméra	Adresse IP interlocuteur	Port interlocuteur	Protocole	Durée (seconde)	Volume de trafic moyen caméra vers interlocuteur (Octet / s)	Délai moyen entre chaque paquet caméra vers interlocuteur (Seconde)	Volume de trafic moyen interlocuteur vers caméra (Octet / s)	Délai moyen entre chaque paquet interlocuteur vers caméra (Seconde)
43570	52.210.196.93 (Ireland Dublin Amazon Data Services Ireland Limited)	443	TCP	11.26	155 950	0.009	3523	0.018
43580	52.210.196.93 (Ireland Dublin Amazon Data Services Ireland Limited)	443	TCP	12.8	131 953	0.01	3160.5	0.022
43587	52.210.196.93 (Ireland Dublin Amazon Data Services Ireland Limited)	443	TCP	12.6	120 793	0.011	3169.7	0.023
43593	52.210.196.93 (Ireland Dublin Amazon Data Services Ireland Limited)	443	TCP	172.08	2300.3	0.604	82.04	1.14
60698	46.51.165.123 (Ireland Dublin Amazon Web Services Elastic Compute Cloud Ec2 Eu)	443	TCP	174.96	123.7	2.57	70.85	2.69

46515	192.168.1.185	52042	UDP	80.59	143 578	0.008	2089.3	0.03
54832	192.168.1.185	64023	UDP	4.82	97 053	0.012	1673.4	0.05
56939	192.168.1.185	61262	UDP	17.16	136 130	0.009	2029.8	0.038
58454	192.168.1.185	56213	UDP	65.34	143 342	0.008	2076.2	0.036

A travers ces conversations, nous avons pu constater que **les conversations de la caméra sont toujours avec les même adresses IP qui sont des serveurs Amazon situés en Irlande**. Les ports utilisés sont en cohérence avec les protocoles. (443 TCP)

Conclusion

A travers nos analyses, nous avons pu constater que la caméra communique avec des protocoles fiables (TLS utilisé pour tous les paquets TCP sortant de la caméra), elle utilise les protocoles attendus au moment de l'envoi de données (TCP, UDP) notamment TCP lorsque l'envoi des paquets doit passer par Internet, les adresses IP avec lesquelles elle communique correspondent toutes à des serveurs que l'on peut considérer comme fiable, c'est-à-dire des serveurs appartenant à un organisme reconnu comme étant fiable (Amazon Web Services).

Elle utilise également des algorithmes tels qu'AES13 pour chiffrer les données avant de les envoyer, ce qui est important pour garantir des normes de sécurité minimales. Ainsi, le comportement réseau de cette dernière semble cohérent, nous n'avons pas relevé de problème compromettant la sécurité.

Analyse du firmware

I. Méthodologie

Dans cette section sera présentée la méthodologie utilisée pour pouvoir récupérer le **firmware** de la caméra ainsi que les différents outils qui ont été nécessaires.

1. Outils utilisés

Le port série de la caméra se présentait sous la forme de quatre pastilles nécessitant la soudure de fils pour pouvoir être utilisé. Un **convertisseur USB vers TTL**¹⁴ a donc été requis pour cette opération.

Une fois les fils soudés, l'accès à la caméra a été fait à l'aide de **PuTTY**¹⁵, qui permet de configurer facilement certains paramètres nécessaires tels que la vitesse de transmission des données (speed) et de sauvegarder sa configuration. PuTTY permet également de sauvegarder dans un fichier tout ce qui a été envoyé par la caméra à travers le port série.



Figure 7 Convertisseur USB vers TTL

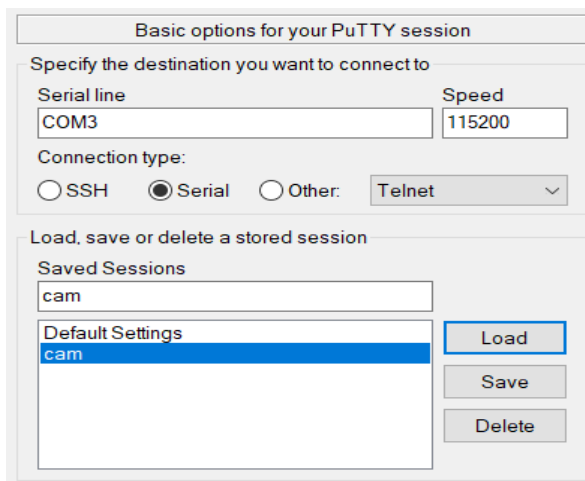


Figure 8 Configuration PuTTY pour se connecter à la caméra

Pour récupérer le contenu de la mémoire de la caméra, une carte micro SD ainsi qu'un adaptateur micro SD – USB ont été utilisés.

Des logiciels d'ingénierie inverse tel que **Ghidra**¹⁶, ou **Binwalk**¹⁷ (outil open-source utilisé pour l'analyse et l'extraction de fichiers binaires) ont été utilisés afin d'analyser le contenu binaire du firmware.

2. Accès au terminal de la caméra

Comme présenté précédemment, un convertisseur USB vers TTL ainsi que PuTTY ont été utilisés pour accéder au terminal de la caméra.

Dans un premier temps, il a été nécessaire de déconnecter les moteurs responsables de la rotation de la caméra afin d'éviter qu'ils n'endommagent les fils soudés lors du démarrage de la caméra puisque cette dernière réalise des rotations pour son auto-calibration.

Ensuite, nous avons procédé à la soudure des fils du convertisseur sur les pastilles **TX**, **RX** et **GND** (correspondant respectivement à la sortie, l'entrée et la masse). Nous n'avons pas jugé nécessaire de souder un fil sur la pastille **VCC** (correspondant à la tension d'alimentation positive) étant donné que l'alimentation est déjà fournie par le câble d'alimentation.

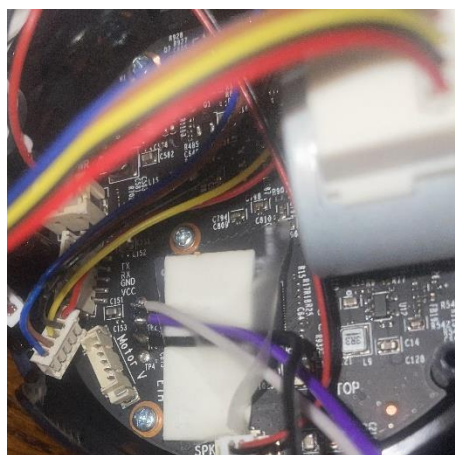


Figure 9 Soudure sur les pastilles de la caméra

Une fois les fils soudés, la dernière étape consistait à dialoguer avec la caméra à l'aide de PuTTY. Avant cela, il était cependant nécessaire de déterminer la vitesse de transmission du port série. Cette étape n'est pas très complexe car les valeurs sont normalisées et peu nombreuses. Dans notre cas, la vitesse de transmission était de **115 200 Bauds**.

3. Récupération de la mémoire

Plusieurs méthodes étaient possibles pour récupérer la mémoire de la caméra :

- Depuis le **U-boot**¹⁸ (Universal Boot Loader), afficher tout le contenu de la mémoire grâce à la commande **md** (memory display) et rediriger la sortie vers un fichier. Cela permet de **récupérer tout le contenu binaire de la mémoire**. Il faudrait ensuite l'analyser avec des outils tels que Binwalk.
- Lancer une mise à jour du firmware à partir de l'application de la caméra et capturer l'adresse à laquelle la caméra télécharge le nouveau firmware. Ensuite, il suffit de télécharger ce firmware depuis un ordinateur. Cependant, cette méthode présente un inconvénient : elle permet de récupérer une **version postérieure** du firmware. Si la dernière version est déjà installée, cette méthode sera donc inutilisable.
- Passer en mode root sur l'invite de commande de la caméra et copier le contenu de la mémoire sur une carte SD insérée dans la caméra, à l'aide des commandes **cp** et **dd**¹⁹.

Nous avons essayé d'utiliser la première méthode, mais l'analyse de la mémoire avec Binwalk s'est avérée très complexe (nous n'avons jamais utilisé le logiciel). Ainsi, nous avons fait le choix d'utiliser la dernière méthode évoquée. Nous avons tout de même expliqué notre démarche pour la première méthode dans la [section 4](#).

Une fois le système de fichier copié sur la carte SD, le contenu a été transféré sur un ordinateur à l'aide d'un adaptateur micro SD – USB.

4. U-Boot

Une fois que nous avons réussi à accéder au terminal de la caméra, notre premier objectif a été d'obtenir un accès à l'invite de commande (en mode root ou non) pour explorer le système de fichiers. Malheureusement, l'accès à l'invite de commande était protégé par un mot de passe, et celui-ci n'ayant pas fuité, il a été impossible pour nous de le trouver. En effet, une attaque par force brute ou par dictionnaire auraient pris trop de temps. Par la suite, nous avons essayé d'accéder au BIOS de la caméra nommé U-Boot. En réalité, U-Boot n'est pas réellement un BIOS mais plutôt un « chargeur d'amorçage » open-source pour les systèmes embarqués. Il remplit un rôle similaire à celui d'un BIOS traditionnel, mais avec des fonctionnalités plus étendues.

A) Accéder à U-Boot

Afin d'accéder à U-Boot, il faut redémarrer la caméra et appuyer rapidement sur les touches clavier « S », « L » et « P » simultanément lorsque le message « Autobooting in 1 seconds » s'affiche. Si la manœuvre a été réalisée avec succès, une invite de commande U-Boot apparaît.

Il peut être nécessaire de refaire plusieurs fois la manipulation avant de parvenir à avoir l'accès à l'invite de commande.

B) Blocage dans U-Boot

Malheureusement, à cause d'une mauvaise commande, nous avons été bloqués dans le U-Boot. En effet, les variables d'environnements de U-Boot ont et l'adresse du point d'amorçage ont été modifiées (mauvaise utilisation de la commande **setenv**²⁰ de U-Boot). Cela a eu pour conséquence l'impossibilité pour U-Boot de démarrer le système de la caméra. La caméra était donc bloquée dans cet état.

Afin de pouvoir continuer à avancer sur notre projet, nous avons acheté une nouvelle caméra identique à la première. Grâce à cette nouvelle caméra et l'aide de nos encadrants, nous avons réussi à retrouver l'adresse du point d'amorçage en consultant les variables d'environnement de cette dernière pour corriger les modifications survenues sur la première et permettre donc à son U-Boot de démarrer le système.

C) Memory dump

Lorsque la caméra était bloquée dans le U-Boot, la seule option possible était d'essayer de récupérer le contenu de la mémoire flash. Pour ce faire, nous avons utilisé la commande **md** (memory display) de U-Boot, qui s'utilise de la manière suivante : **MD [adresse] [taille]** avec [adresse] l'adresse de départ à partir de laquelle les octets seront lus, et [taille] le nombre d'octets à lire. Il suffit ensuite de rediriger la sortie avec PuTTY pour qu'elle soit écrite dans un fichier.

La difficulté était donc de trouver la bonne adresse de départ et le bon nombre d'octets à lire. En affichant certains blocs de mémoire (choisi de manière aléatoire), certaines informations ont pu être récupérées car elles étaient écrites en clair dans la mémoire (par exemple l'adresse de points d'entrée). Cependant, il était difficile de savoir si les informations trouvées correspondaient à ce que l'on recherche. Nous avons essayé de passer certains blocs de mémoire récupérer dans Binwalk, sans succès.

Comme dit précédemment, nous avons finalement abandonner cette méthode.

5. Récupération du firmware

Afin de récupérer le firmware, la méthode utilisée a donc été d'essayer d'obtenir une **invite de commande en mode root** sur la caméra pour copier le système de fichier sur la carte SD.

A) Obtenir une invite de commande en mode root

Comme cité précédemment, l'accès à l'invite de commande est protégé par un mot de passe. Ainsi, nous avons cherché une méthode afin de contourner ce problème. De ce fait, nous nous sommes demandé à quel moment ce mot de passe était configuré et à partir de quel moment il devient nécessaire pour accéder à l'invite de commande. Nous avons eu comme idée de laisser la caméra en mode usine, ce qui implique ne pas la configurer.

Ensuite, nous avons essayé d'accéder à l'invite de commande et étant donné que la caméra n'était pas configurée, **nous avons eu accès à cette dernière en tant que root sans qu'aucun mot de passe ne soit requis.**

Une fois en mode root, nous avons choisi un mot de passe pour l'utilisateur « root ». Il était également possible de créer un autre utilisateur avec le droit root. Nous aurions pu utiliser cette méthode pour éviter que le mot de passe du root ne soit changé au moment de la configuration auquel cas ne n'aurions plus accès à l'invite de commande.

Cependant, la caméra utilise **Busybox**²¹, un programme qui fournit des commandes Unix de base. Très peu de commandes étaient donc disponibles. Les commandes pour créer un utilisateur n'étant pas disponibles, il aurait fallu le faire manuellement en modifiant les fichiers `/etc/passwd` et `/etc/shadow`²².

B) Copier la mémoire

La méthode la plus simple et qui est celle choisie pour copier la mémoire est d'introduire une carte micro SD dans le port prévu à cet effet sur la caméra, puis le monter en mémoire avec la commande `mount`²³ (il faut faire attention à trouver le bon emplacement à monter en mémoire). Il suffit ensuite de copier l'ensemble du système de fichier sur la carte SD avec la commande `cp` ou `dd`.

6. Analyse du firmware

Dans cette section seront présentées les analyses du comportement du firmware.

Processus

Afin de lister les différents processus en exécution sur la caméra, plusieurs scénarios ont été définis :

- Un scénario sans mouvements devant la caméra et sans l'utilisation de l'application.
- Un scénario sans mouvements devant la caméra et avec l'utilisation de l'application.
- Un scénario avec mouvements devant la caméra et sans l'utilisation de l'application.
- Un scénario avec mouvements devant la caméra et avec l'utilisation de l'application.

Pour lister les processus, les commandes **top** et **ps** ont été utilisés.

```
Mem: 37148K used, 4080K free, 0K shrd, 756K buff, 9708K cached
CPU: 17% usr 10% sys 0% nic 70% idle 0% io 0% irq 1% sirq
Load average: 2.77 2.74 2.87 3/136 23338
```

PID	PPID	USER	STAT	VSZ	VSZ%	%CPU	COMMAND
1156	1	root	S	443m1098k	22%	22%	/bin/cet
1058	1	root	S	45560	110%	4%	/usr/bin/relayd
1160	2	root	DW	0	0%	2%	[isp_fw_process]
969	1	root	S	46200	112%	0%	/bin/cloud-client
499	1	root	S	37400	90%	0%	/usr/sbin/wlan-manager
1060	1	root	S	10588	26%	0%	/bin/storage_manager
309	1	root	S	1092	3%	0%	/sbin/ubusd
1082	2	root	SW	0	0%	0%	[irq/37-isp-m0]
19	2	root	SW	0	0%	0%	[kworker/0:2]
7	2	root	SW	0	0%	0%	[rcu_preempt]
981	1	root	S	61744	149%	0%	/bin/cloud-service
1069	1	root	S	60792	147%	0%	/usr/bin/p2pd
864	1	root	S	53420	129%	0%	/usr/bin/dsd
1056	1	root	S	44992	109%	0%	/usr/bin/rtspd
1128	1	root	S	29644	72%	0%	/bin/nvid
1130	1	root	S	26536	64%	0%	/bin/wtd
334	1	root	S	20480	50%	0%	tp_manage
452	1	root	S	9760	24%	0%	/usr/bin/ledd
878	1	root	S	4628	11%	0%	/bin/cloud-brd -c /var/etc/cloud_brd_c
1120	1	root	S	4496	11%	0%	/bin/telemetry

Figure 10 top avec mouvement avec application

```
Mem: 37148K used, 4080K free, 0K shrd, 756K buff, 9708K cached
CPU: 17% usr 13% sys 0% nic 68% idle 0% io 0% irq 0% sirq
Load average: 2.67 2.79 2.92 8/136 21863
```

PID	PPID	USER	STAT	VSZ	VSZ%	%CPU	COMMAND
1156	1	root	R	443m1098k	21%	21%	/bin/cet
1160	2	root	RW	0	0%	3%	[isp_fw_process]
1082	2	root	SW	0	0%	1%	[irq/37-isp-m0]
309	1	root	S	1092	3%	0%	/sbin/ubusd
981	1	root	S	61744	149%	0%	/bin/cloud-service
969	1	root	S	46200	112%	0%	/bin/cloud-client
499	1	root	S	37400	90%	0%	/usr/sbin/wlan-manager
1130	1	root	S	26536	64%	0%	/bin/wtd
1060	1	root	S	10588	26%	0%	/bin/storage_manager
19	2	root	RW	0	0%	0%	[kworker/0:2]
687	2	root	SW	0	0%	0%	[RTW_CMD_THREAD]
1069	1	root	S	60792	147%	0%	/usr/bin/p2pd
864	1	root	S	53420	129%	0%	/usr/bin/dsd
1058	1	root	S	45560	110%	0%	/usr/bin/relayd
1056	1	root	S	44992	109%	0%	/usr/bin/rtspd
1128	1	root	S	29644	72%	0%	/bin/nvid
324	1	root	S	20480	50%	0%	tp_manage
452	1	root	S	9760	24%	0%	/usr/bin/ledd
878	1	root	S	4628	11%	0%	/bin/cloud-brd -c /var/etc/cloud_brd_c
1120	1	root	S	4496	11%	0%	/bin/telemetry

Figure 11 top avec mouvement sans application

```
Mem: 37184K used, 4044K free, 0K shrd, 756K buff, 9708K cached
CPU: 20% usr 8% sys 0% nic 69% idle 0% io 0% irq 1% sirq
Load average: 2.60 2.73 2.88 1/136 22653
```

PID	PPID	USER	STAT	VSZ	VSZ%	%CPU	COMMAND
1156	1	root	S	443m1098k	22%	22%	/bin/cet
1058	1	root	S	45560	110%	4%	/usr/bin/relayd
1160	2	root	DW	0	0%	3%	[isp_fw_process]
1082	2	root	SW	0	0%	1%	[irq/37-isp-m0]
981	1	root	S	61744	149%	0%	/bin/cloud-service
969	1	root	S	46200	112%	0%	/bin/cloud-client
499	1	root	S	37400	90%	0%	/usr/sbin/wlan-manager
1130	1	root	S	26536	64%	0%	/bin/wtd
4253	1	root	S	1884	5%	0%	/usr/sbin/ntpd -n -p time.nist.gov -p
22624	240	root	R	1876	5%	0%	top
309	1	root	S	1092	3%	0%	/sbin/ubusd
7	2	root	SW	0	0%	0%	[rcu_preempt]
687	2	root	SW	0	0%	0%	[RTW_CMD_THREAD]
3	2	root	SW	0	0%	0%	[ksftirqd/0]
1069	1	root	S	60792	147%	0%	/usr/bin/p2pd
864	1	root	S	53420	129%	0%	/usr/bin/dsd
1056	1	root	S	44992	109%	0%	/usr/bin/rtspd
1128	1	root	S	29644	72%	0%	/bin/nvid
334	1	root	S	20480	50%	0%	tp_manage
1060	1	root	S	10588	26%	0%	/bin/storage_manager

Figure 6 top sans mouvement sans application

```
Mem: 37148K used, 4080K free, 0K shrd, 756K buff, 9708K cached
CPU: 12% usr 19% sys 0% nic 67% idle 0% io 0% irq 0% sirq
Load average: 2.89 2.85 2.95 2/136 20968
```

PID	PPID	USER	STAT	VSZ	VSZ%	%CPU	COMMAND
1156	1	root	S	443m1098k	20%	20%	/bin/cet
1160	2	root	DW	0	0%	2%	[isp_fw_process]
499	1	root	S	37400	90%	0%	/usr/sbin/wlan-manager
969	1	root	S	46200	112%	0%	/bin/cloud-client
1060	1	root	S	10588	26%	0%	/bin/storage_manager
878	1	root	S	4628	11%	0%	/bin/cloud-brd -c /var/etc/cloud_brd_c
1120	1	root	S	4496	11%	0%	/bin/telemetry
18624	240	root	R	1876	5%	0%	top
19	2	root	SW	0	0%	0%	[kworker/0:2]
7	2	root	SW	0	0%	0%	[rcu_preempt]
981	1	root	S	61744	149%	0%	/bin/cloud-service
1069	1	root	S	60792	147%	0%	/usr/bin/p2pd
864	1	root	S	53420	129%	0%	/usr/bin/dsd
1058	1	root	S	45560	110%	0%	/usr/bin/relayd
1056	1	root	S	44992	109%	0%	/usr/bin/rtspd
1128	1	root	S	29644	72%	0%	/bin/nvid
1130	1	root	S	26536	64%	0%	/bin/wtd
334	1	root	S	20480	50%	0%	tp_manage
452	1	root	S	9760	24%	0%	/usr/bin/ledd
1037	1	root	S	4068	10%	0%	/usr/sbin/uhttpd -f -h /www -T 180 -A

Figure 7 top sans mouvement avec application

On peut constater qu'il n'y a pas de réelle corrélation entre l'utilisation de l'application ou pas ou le fait qu'il y ai du mouvement ou pas et le taux d'utilisation du CPU par certaines applications.

Cependant, on peut constater que la majorité de la puissance de calcul est utilisée par le processus **/bin/cet**, on peut donc en déduire que la majorité du comportement de la caméra est contrôlé par ce processus.

Ci-dessous seront présentées les différentes informations trouvées sur les processus jugés les plus intéressants en fonctionnement sur la caméra.

/bin/cet

Comme dit précédemment, ce fichier binaire gère la majorité des fonctionnalités de la caméra. Un port est ouvert pour cette application, on peut donc en déduire que ce programme s'occupe également de certaines fonctionnalités réseau.

Avec le logiciel Ghidra, nous avons décompilé ce fichier binaire pour essayer de comprendre son fonctionnement. La compréhension du code source est assez difficile étant donné qu'après décompilation, le code source n'est pas la même qu'à l'original (plus les mêmes noms de variables, plus les mêmes noms de fonctions, pas d'accès à la définition des structures...). Cependant, en analysant les librairies importées, on peut avoir une idée de quels services rend cette application.

Voici une liste (non exhaustive) des bibliothèques importées qui peuvent s'avérer intéressantes pour comprendre les services remplis par ce fichier binaire :

- **libaes.so**: cette bibliothèque est souvent utilisée pour fournir des fonctions de cryptage et de décryptage avec l'algorithme AES.
- **liblog.so**: cette bibliothèque fournit des fonctions pour la journalisation et la gestion des journaux de l'application.
- **libaudioProcess.so**: cette bibliothèque peut être utilisée pour fournir des fonctionnalités de traitement audio.
- **libcrypto.so.1.0.0 et libssl.so.1.0.0**: ces bibliothèques sont liées à **OpenSSL**, qui est une bibliothèque de cryptographie largement utilisée.
- **libdecrypter.so**: cette bibliothèque peut être utilisée pour fournir des fonctions de décryptage pour certains types de données.
- **libdsd_client.so**: cette bibliothèque peut être liée à une application de communication.
- **libptz.so**: cette bibliothèque peut être utilisée pour fournir des fonctions de contrôle de la caméra de surveillance pour le mouvement panoramique / inclinaison / zoom.
- **libsmart_detection.so**: cette bibliothèque peut être utilisée pour fournir des fonctions de détection d'objet pour la vidéo-surveillance.
- **libstorage_manager.so**: cette bibliothèque peut être utilisée pour fournir des fonctions de gestion du stockage.
- **libsysutils.so**: cette bibliothèque fournit des fonctions utilitaires pour les applications système Android.
- **libubox.so et libubus.so**: ces bibliothèques sont souvent utilisées avec **OpenWrt**²³ et fournissent des fonctions pour les communications système.
- **libuci.so**: cette bibliothèque fournit des fonctions pour la gestion de la configuration système.

On peut donc constater que cette application gère la **majorité des actions de la caméra**, telle que le cryptage des données à transmettre, le décryptage des données reçus, la gestion de l'audio, la détection de forme, le contrôle des mouvements de la caméra ...

usr/bin/relayd

Service généralement associé à la **distribution Unix OpenBSD**²⁴. Ce service permet de gérer le trafic réseau en réalisant diverses tâches telles que de la répartition de charge, le monitoring ou encore faire office de pare-feu.

usr/bin/rtspd

Service généralement utilisé pour un serveur de streaming en temps réel. Il permet de gérer efficacement les demandes de streaming de clients, en utilisant le protocole **RTSP**²⁵. Ce service est sûrement utilisé pour pouvoir avoir les images de la caméra en temps réel depuis l'application.

/bin/cloud-client et /bin/cloud-service

Service probablement utilisé pour interagir avec le service de stockage cloud de la caméra pour stocker des images/vidéos sur les serveurs du fabricant.

/usr/sbin/uhttpd

Il s'agit généralement d'un serveur web léger et rapide utilisé pour héberger des sites web sur des systèmes embarqués. Il peut être configuré pour prendre en charge différents protocoles, comme **HTTP**, **HTTPS**²⁶, et d'autres. Il peut également offrir des fonctionnalités de routage, de gestion de la sécurité, d'authentification, de gestion des certificats **SSL/TLS**²⁷, de gestion des sessions, et d'autres fonctionnalités courantes des serveurs web.

Structure du système de fichiers

Pour comprendre le fonctionnement du firmware, il est important de comprendre comment est organisé le système de fichier. Etant donné que la caméra utilise un système Unix, l'aménagement de son système de fichier est identique à celui de n'importe quel autre appareil utilisant un OS Unix.

Dossiers bin et/sbin

Contiennent la majorité des fichiers exécutables disponibles sur la caméra.

Dossier/etc

Contient les fichiers de configuration classiques tels que **passwd** ou **shadow**.

Il contient également le dossier **rc.d** qui contient des liens symboliques sur des scripts du dossier **init.d** qui permettent d'initialiser certains services systèmes, notamment le service **/bin/cet** présenté plus haut.

Divers dossiers contenant des informations (comme des fichiers de configuration) pour certaines applications sont également présents, par exemple un certificat de sécurité pour le service **cloud-client**.

Dossier/rom

Ce dossier contient un autre système de fichier, en mode lecture seule, identique à celui placé à la racine, sauf pour ses dossiers **mnt** et **rom**. Le dossier **mnt** contient lui aussi un autre système de fichier, le dossier **rom** lui contient uniquement un fichier texte avec écrit :

« **SQUASHFS USERS:**

After firstboot has been run, / will be jffs2 and /rom will be squashfs

(except when in failsafe) »*

En d'autres termes, lors du premier démarrage de la caméra ou après l'exécution de la commande **firstboot**, le système de fichiers utilisé pour **/** (la racine) sera modifié pour **jffs2**, qui est un autre type de système de fichiers utilisé dans les systèmes embarqués utilisant Unix. Le système de fichiers **/rom**, quant à lui, restera en **SQUASHFS**, qui est un système de fichiers en lecture seule généralement utilisé pour stocker des données compressées en lecture seule.

Il est possible que cette configuration soit utilisée pour assurer la sécurité et la stabilité du système, en isolant les parties du système de fichiers qui sont modifiables (**jffs2**) de celles qui sont en lecture seule (**SQUASHFS**). Cela peut également permettre de restaurer les paramètres par défaut du système en exécutant la commande "**firstboot**" en cas de besoin, sans affecter les parties en lecture seule du système.

La totalité du firmware est disponible sur le dépôt github du projet : https://github.com/augustin-laouar/POM/tree/main/firmware/copie_firmware

Etude des vulnérabilités

Dans cette partie seront présentées les principales vulnérabilités qui ont été trouvées sur la caméra Tapo C200, aussi bien sur la caméra elle-même que sur l'application mobile qui lui est associée.

I. Injection de commande

L'une des failles les plus intéressantes n'est plus exploitable dans la version du firmware de la caméra sur laquelle nous travaillons (1.1.22). Cependant, cette faille reste tout de même intéressante à présenter.

Cette faille provient du service **uhttpd**, présenté précédemment. En décompilant son fichier binaire, une faille a été trouvée par un informaticien (du pseudo de « *hacefresko* ») l'ayant détaillé sur son site internet et un script permettant son exploitation a été mis en ligne sur GitHub (<https://github.com/hacefresko/CVE-2021-4045-PoC>). La faille provient d'une fonction nommée « **uh_slp_proto_request** » qui traite des **requêtes POST**²⁸. En envoyant une requête POST à la racine ("/") du système de la caméra avec un contenu spécifique, la fonction "**uh_slp_proto_request**" traitera les données et appellera la méthode "**set_language**". Le contenu injecté (ligne 36 et 42 du script disponible à l'adresse <https://github.com/augustin-laouar/POM/blob/main/injection.py>) dans la méthode "**set_language**" sera ensuite exécutée par la fonction "**exec_and_get_result**". Etant donné que la fonction "**set_language**" ne nécessite pas d'authentification, il est possible de réaliser une élévation de privilèges (situation où un utilisateur obtient un niveau de privilèges supérieur à celui qui lui était initialement attribué) et donc d'obtenir une invite de commande en mode root. Il est important de préciser que pour exploiter cette faille, **il faut être sur le même réseau local que la caméra**, car sinon le pare-feu du routeur risquera de bloquer la requête.

Une autre exploitation de la faille était de configurer à distance les identifiants d'accès au flux RTSP de la caméra afin de pouvoir récupérer le streaming vidéo de la caméra depuis une machine distante. Les identifiants d'accès au flux RTSP sont initialement configurés avec les identifiants d'accès à la caméra.

Nous avons réussi à obtenir le flux RTSP depuis une machine distante en l'initialisant directement sur la caméra et ce car nous y avons accès en mode root via le port série. Pour ce faire, nous avons utilisé ces commandes suivantes :

- `uci set user_management.third_account.username=test`
- `uci set user_management.third_account.passwd=mdp_hache_avec_MD5`
- `uci set user_management.third_account.ciphertext=rtsp_ciphertext_MD5`
- `uci commit user_management`
- `/etc/init.d/cet terminate`
- `/etc/init.d/cet resume`

Avec « test » un nom d'utilisateur, « mdp_hache_avec_MD5 » un mot de passe haché avec la fonction **MD5**²⁹ et « rtsp_ciphertext_MD5 » contient des données hachées avec **MD5** concernant le compte utilisateur. Ces commandes affectent l'utilisateur « **third_account** » qui représente un compte utilisateur supplémentaire ou tiers dans un système basé sur **UCI** (Unified Configuration Interface). UCI est un système de configuration unifié utilisé dans certaines distributions de Linux, telles que OpenWrt, pour gérer les configurations système et les paramètres des applications.

Après avoir réalisé ces commandes, il a suffi d'ouvrir le flux RTSP sur le lien `rtsp://ip_camera/stream2` (avec `ip_camera` l'IP de la caméra). Dans notre cas, nous avons utilisé VLC media player pour ouvrir ce lien, après authentification avec le nom d'utilisateur et le mot de passe que nous avons configuré précédemment, nous obtenons un flux vidéo en direct provenant de la caméra.

Bien qu'il soit nécessaire d'avoir un accès direct à la caméra via le port série en mode root rende cette faille difficilement exploitable, son exploitation reste possible et ce car la configuration réalisée sur la caméra est persistante car même après un redémarrage, la configuration n'est pas écrasée. Il est donc possible, dans un scénario idéal, qu'un attaquant ait la caméra à sa disposition et initialise manuellement ce flux vidéo, puis donne la caméra à une victime tout en aillant toujours accès au flux vidéo. Un exemple d'attaquant peut être un vendeur Amazon et la victime un acheteur qui commande cette caméra.

Nous sommes tout de même conscients que ce type de scénario est très peu probable, et que l'on peut toujours considérer la caméra comme étant fiable.

II. Injection de script

Deux méthodes ont été identifiées pour créer un script qui sera exécuté à chaque démarrage de la caméra, dans lequel il serait possible de mettre du code malveillant par exemple, donner un accès distant à une invite de commande ou au flux vidéo de la caméra.

Méthode 1 : le dossier `/etc/uci-defaults/`

Dans le dossier `/etc/init.d/boot`, un fichier texte contient la note suivante :

```
# Set some defaults at the first time the device boots.
# This is a light version of 'uci-defaults'. by lizheng.
# NOTE:
# All scripts in the folder /etc/uci-defaults/ are automatically executed,
# and if they exited with code 0 deleted afterwards (scripts that did not
# exit with code 0 are not deleted and will be re-executed during the next
# boot until they also successfully exit).
# You can put your own default files in that folder.
```

D'après cette note, tous les scripts dans le dossier `/etc/uci-defaults` seront exécutés automatiquement à chaque démarrage.

En fonction du code de sortie, le script sera supprimé ou non.

On pourrait donc mettre notre propre script dans ce dossier, dont le code de terminaison sera 1, **afin qu'il soit exécuté à chaque démarrage de la caméra.**

Méthode 2 : le dossier `/etc/init.d` et `/etc/rc.d`

D'après la documentation, les scripts exécutés au démarrage sur le système sont dans le dossier `/etc/init.d` et des liens symboliques leurs sont associés dans le dossier `etc/rc.d`. Il serait donc possible d'ajouter un script de démarrage le dossier `/etc/init.d` et créer son lien symbolique dans `etc/rc.d`.

Ces deux méthodes n'ont pas abouti, car toutes les modifications apportées ne sont pas conservées après un redémarrage de la caméra. Ce problème sera plus détaillé dans la section suivante.

III. Problèmes rencontrés

Lors de la recherche de failles, nous avons rencontré un certain nombre de problèmes qui seront énoncés dans cette section.

Non persistance des modifications apportés au firmware

L'ajout, la modification ou la suppression d'un fichier ou d'un dossier n'est pas persistante. En effet, aucune modification n'est prise en compte après redémarrage.

Nous avons deux hypothèses concernant ce problème :

- A chaque démarrage, la caméra charge en RAM son système de fichier depuis sa mémoire flash. Les modifications apportées seront donc perdues dès que la caméra éteinte.
- La caméra utilise un mécanisme de **checksum** pour vérifier l'intégrité de son firmware en le calculant à partir d'une image de référence.

Le checksum est une valeur numérique calculée à partir des octets de l'image du firmware à l'aide d'un algorithme de hachage ou d'un algorithme de somme de contrôle. Cela crée une valeur unique qui représente l'intégrité de l'image du firmware. Le checksum calculé est ensuite stocké dans un emplacement sécurisé de la caméra, généralement dans une zone de mémoire protégée ou dans un registre sécurisé dédié. Cela permet de conserver le checksum de référence pour comparaison ultérieure. A chaque démarrage du système embarqué, le firmware en cours d'exécution est comparé avec le checksum de référence stocké. Cela se fait en recalculant le checksum de l'image du firmware en cours d'exécution et en le comparant avec le checksum de référence stocké. Si les checksums ne correspondent pas, alors une image de référence est rechargée.

La non-persistance des modifications nous a empêchée d'exploiter certaines failles potentielles, ce qui est un atout pour la sécurité de la caméra.

Cependant, il est intéressant de noter que les modifications apportées aux variables d'environnement UCI (Unified Configuration Interface) restent valables même après un redémarrage.

Confiance

Il est important de pouvoir avoir confiance en l'entreprise à qui l'on confie nos données. Afin de savoir si l'entreprise ou le service utilisé est éthiquement fiable, il faut se poser plusieurs questions : où sont stockées les données ? Qui a accès aux données ? Avec qui le service utilisé communique-t-il ? Le canal de communication est-il fiable ?

Comme indiqué dans la partie réseau, la caméra stock ses données sur des serveurs AWS (Amazon Web Services), sûrement loués par TP-link. Cependant, il est difficile de savoir qui a accès à ces serveurs et quel est leur niveau de sécurité. Cependant, les serveurs AWS sont généralement considérés comme fiables pour le stockage de données. AWS utilise des infrastructures de datacenters de pointe, des technologies de sécurité avancées et des pratiques de gestion de la sécurité rigoureuses pour protéger les données de ses clients.

En ce qui concerne les communications, tous les paquets de données sont chiffrés avec AES 128, qui est considéré comme étant très fiable. Pour l'envoi de données, la caméra communique uniquement avec les serveurs Amazon dédiés aux stockages de données et le téléphone hébergeant l'application associée à la caméra.

Nous n'avons donc relevé aucun problème d'éthique sur la caméra Tapo C200.

Sécurité

Bien que la caméra soit éthiquement fiable, il reste possible que nos données ne soient pas en sécurité si des failles sont exploitables par de potentiels attaquants.

Une attaque peut aussi bien être physique que distante. Nous avons donc pris en compte ces deux types de scénarios. L'attaque peut être du vol d'informations, une mise hors service du matériel, un troncage des données (par exemple envoyé les mauvaises images) ... Les failles de sécurités peuvent aussi bien concerner la caméra que l'application mobile associée.

Toutes les failles présentées dans la partie « Etude des vulnérabilités » **ne sont plus exploitables actuellement**, car sur la version du firmware présente dans les caméras Tapo C200 vendus elles y sont corrigées. En effectuant des recherches en ligne, on constate rapidement qu'aucune autre faille n'a été trouvée.

Cependant, il faut garder à l'esprit que bien que nous n'aillions pas trouvés de faille exploitable actuellement, la caméra n'est pas sûre pour autant et des failles de sécurité ont pu être trouvés par d'autres utilisateurs qui ne les auraient pas partagés en ligne, ou pourront peut-être être trouvés plus tard. En effet, aucun programme ou équipement n'est fiable à 100 % en matière de sécurité informatique.

Transparence du fabricant

La transparence est un élément très important pour le consommateur qui souhaite avoir la certitude que sa caméra est fiable. Pour cela, plusieurs critères sont à prendre en compte : l'accès au code source du firmware de la caméra, des détails sur les protocoles de communication et de sécurité utilisés ainsi que sur le stockage des données sur le cloud, etc...

Premièrement, le code du firmware n'est pas open source. La caméra utilisant un système d'exploitation OpenBSD qui est open source, mais **dont la licence est peu restrictive et n'oblige pas l'utilisateur à mettre open source le code du projet pour lequel OpenBSD est utilisé**, on peut donc penser que c'est l'une des raisons de l'utilisation de ce système d'exploitation.

Le fait que le code ne soit pas open source pose plusieurs problèmes :

- **Manque de confiance** : Lorsque le code source d'une caméra n'est pas open source, cela signifie que les utilisateurs n'ont pas la possibilité de vérifier le code pour s'assurer qu'il ne contienne pas de fonctionnalités indésirables, telles que des backdoors ou des fonctionnalités de surveillance cachées. Cela peut entraîner un manque de transparence et de confiance envers le fabricant de la caméra, ce qui peut poser des problèmes éthiques.
- **Risques de sécurité** : Un code source fermé peut rendre difficiles la détection et la correction rapide de failles de sécurité. Si le code source d'une caméra n'est pas accessible pour être examiné par la communauté de sécurité, il peut y avoir des risques de failles non détectées qui pourraient être exploitées par des attaquants pour accéder illégalement à la caméra, aux données enregistrées ou à d'autres informations sensibles.
- **Monopole du fabricant** : Un code source fermé peut conférer au fabricant de la caméra un monopole sur les mises à jour, les correctifs de sécurité et les nouvelles fonctionnalités de la caméra, ce qui peut limiter la concurrence et l'innovation dans le marché des caméras. Cela peut également créer une dépendance envers le fabricant pour le support technique et les mises à jour, ce qui peut être problématique si le fabricant cesse de soutenir la caméra ou de fournir des mises à jour de sécurité.
- **Respect de la vie privée** : Les caméras sont souvent utilisées pour la surveillance et la collecte de données, ce qui soulève des préoccupations importantes en matière de respect de la vie privée. Lorsque le code source d'une caméra n'est pas open source, il peut être difficile pour les utilisateurs de savoir exactement ce que la caméra fait avec les données qu'elle collecte et comment ces données sont traitées. Cela peut entraîner des problèmes d'éthique liés à la confidentialité et à la protection des données des utilisateurs.

Le fabricant ne donne que très peu d'informations sur le type de données envoyés sur les serveurs, l'utilisation de ces données et sur la manière dont elles sont envoyées.

Il n'y a également pas d'informations sur l'infrastructure nécessaire au bon fonctionnement de la caméra, tel qu'un serveur DNS et un serveur DHCP, leur absence pourrait nuire au bon fonctionnement de la caméra. Le constructeur estime que tous les consommateurs utiliseront une

infrastructure classique (la caméra sera connectée sur la box de leur fournisseur internet), ce qui n'est pas toujours le cas, une documentation à ce sujet est donc nécessaire.

On peut donc constater qu'il y a un réel problème de transparence de la part du fabricant de l'appareil.

Conclusion

La caméra Tapo C200 peut être considérée comme étant suffisamment sécurisée pour une utilisation domestique. L'analyse du comportement réseau n'a rien révélé de suspect vis-à-vis de l'utilisation des données.

Cependant, le manque total de transparence de la part du fabricant est problématique. Des failles de sécurité qui ne sont pas connues à ce jour pourraient être découvertes, ou sont peut-être même actuellement connues par des attaquants. Avoir le code du firmware open source aurait permis de vérifier la qualité et la sécurité du code par la communauté et d'identifier et de corriger rapidement les éventuelles vulnérabilités de sécurité.

Le manque de transparence sur l'utilisation des données est également un problème, car bien que la caméra n'envoie des données qu'aux serveurs cloud, nous n'avons aucune information sur l'utilisation de ces données par le fabricant (sont-elles vendues à des sociétés tiers par exemple ?). La caméra Tapo C200 peut donc convenir pour une utilisation domestique simple, telle que surveiller le jardin ou bien la chambre d'un nourrisson, car son prix est tout de même très attractif, bien que nous recommandons aux potentiels acheteurs de prendre conscience du problème que peut poser le manque de transparence du fabricant.

Cependant, elle ne pourrait pas convenir pour une utilisation professionnelle ou publique, telle que la surveillance d'un bureau ou bien d'un lieu public, car une telle utilisation est souvent soumise à des lois et réglementations spécifiques en matière de protection de vie privée, de protection des données, de consentement des individus, etc. Si les fabricants ne respectent pas ces lois et réglementations, cela peut entraîner des problèmes de conformité légaux pour l'entreprise utilisatrice des caméras.

Annexes

Adresses qui reçoivent des paquets de la caméra

Scénario no-app-no-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
> Source IPv4 Addresses	42				0,0002	100%	0,0200	33,407
▼ Destination IPv4 Addresses	42				0,0002	100%	0,0200	33,407
51.145.123.29	3				0,0000	7,14%	0,0100	40,684
46.51.165.123	9				0,0001	21,43%	0,0100	32,549
192.36.144.22	2				0,0000	4,76%	0,0100	50,613
132.163.96.6	3				0,0000	7,14%	0,0100	43,305
131.107.13.100	6				0,0000	14,29%	0,0100	0,000
129.6.15.29	4				0,0000	9,52%	0,0100	33,407
129.6.15.28	6				0,0000	14,29%	0,0100	6,389
128.138.140.44	3				0,0000	7,14%	0,0100	33,407
103.242.68.68	3				0,0000	7,14%	0,0100	41,772
103.126.53.123	3				0,0000	7,14%	0,0100	4,004

Scénario no-app-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	107				0,0006	100%	0,0700	43,241
192.168.137.67	107				0,0006	100,00%	0,0700	43,241
▼ Destination IPv4 Addresses	107				0,0006	100%	0,0700	43,241
77.196.128.168	7				0,0000	6,54%	0,0100	43,329
52.213.90.205	41				0,0002	38,32%	0,0300	138,020
52.18.5.103	8				0,0000	7,48%	0,0300	49,237
51.145.123.29	1				0,0000	0,93%	0,0100	115,803
46.51.165.123	9				0,0001	8,41%	0,0200	42,585
3.250.192.158	1				0,0000	0,93%	0,0100	43,299
3.248.99.37	21				0,0001	19,63%	0,0500	43,241
192.36.144.22	1				0,0000	0,93%	0,0100	125,566
192.168.137.1	1				0,0000	0,93%	0,0100	49,655
132.163.96.6	1				0,0000	0,93%	0,0100	122,255
131.107.13.100	4				0,0000	3,74%	0,0100	0,000
129.6.15.29	4				0,0000	3,74%	0,0100	20,061
129.6.15.28	4				0,0000	3,74%	0,0100	11,042
128.138.140.44	1				0,0000	0,93%	0,0100	47,409
103.242.68.68	1				0,0000	0,93%	0,0100	115,803
103.126.53.123	2				0,0000	1,87%	0,0100	21,026

Scénario app-no-move-4G

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	5346				0,0311	100%	0,2500	147,911
192.168.137.67	5346				0,0311	100,00%	0,2500	147,911
▼ Destination IPv4 Addresses	5346				0,0311	100%	0,2500	147,911
52.213.90.205	13				0,0001	0,24%	0,0200	115,449
52.210.196.93	5279				0,0307	98,75%	0,2500	147,911
52.18.5.103	8				0,0000	0,15%	0,0300	35,466
51.145.123.29	1				0,0000	0,02%	0,0100	124,560
46.51.165.123	33				0,0002	0,62%	0,0300	149,339
192.36.144.22	1				0,0000	0,02%	0,0100	136,977
192.168.137.1	1				0,0000	0,02%	0,0100	35,877
132.163.96.6	1				0,0000	0,02%	0,0100	141,156
131.107.13.100	2				0,0000	0,04%	0,0100	53,746
129.6.15.29	2				0,0000	0,04%	0,0100	65,062
129.6.15.28	2				0,0000	0,04%	0,0100	46,746
128.138.140.44	1				0,0000	0,02%	0,0100	59,752
103.242.68.68	1				0,0000	0,02%	0,0100	124,560
103.126.53.123	1				0,0000	0,02%	0,0100	78,068

Scénario app-no-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	7880				0,0435	100%	0,4900	16,081
192.168.137.67	7880				0,0435	100,00%	0,4900	16,081
▼ Destination IPv4 Addresses	7880				0,0435	100%	0,4900	16,081
77.196.128.168	16				0,0001	0,20%	0,0400	4,196
52.210.196.93	438				0,0024	5,56%	0,2400	16,081
51.145.123.29	2				0,0000	0,03%	0,0100	0,000
46.51.165.123	72				0,0004	0,91%	0,0400	139,624
3.248.99.37	8				0,0000	0,10%	0,0100	4,049
192.36.144.22	2				0,0000	0,03%	0,0100	11,570
192.168.1.185	7327				0,0404	92,98%	0,3000	56,337
132.163.96.6	2				0,0000	0,03%	0,0100	5,220
131.107.13.100	3				0,0000	0,04%	0,0100	48,171
129.6.15.29	3				0,0000	0,04%	0,0100	49,172
129.6.15.28	3				0,0000	0,04%	0,0100	43,164
128.138.140.44	1				0,0000	0,01%	0,0100	67,187
103.242.68.68	1				0,0000	0,01%	0,0100	131,590
103.126.53.123	2				0,0000	0,03%	0,0100	40,802

Scénario app-move-4G

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	18414				0,1032	100%	0,5700	124,793
192.168.137.67	18414				0,1032	100,00%	0,5700	124,793
▼ Destination IPv4 Addresses	18414				0,1032	100%	0,5700	124,793
52.213.90.205	40				0,0002	0,22%	0,0300	4,749
52.210.196.93	18311				0,1026	99,44%	0,5700	124,793
51.145.123.29	1				0,0000	0,01%	0,0100	157,984
46.51.165.123	51				0,0003	0,28%	0,0300	0,000
192.36.144.22	1				0,0000	0,01%	0,0100	175,249
132.163.96.6	1				0,0000	0,01%	0,0100	169,053
131.107.13.100	2				0,0000	0,01%	0,0100	61,909
129.6.15.29	2				0,0000	0,01%	0,0100	77,904
129.6.15.28	2				0,0000	0,01%	0,0100	53,904
128.138.140.44	1				0,0000	0,01%	0,0100	94,920
103.242.68.68	1				0,0000	0,01%	0,0100	150,571
103.126.53.123	1				0,0000	0,01%	0,0100	115,187

Scénario app-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	23771				0,1344	100%	0,6400	159,785
192.168.137.67	23771				0,1344	100,00%	0,6400	159,785
▼ Destination IPv4 Addresses	23771				0,1344	100%	0,6400	159,785
99.80.203.8	24				0,0001	0,10%	0,0400	67,841
77.196.128.168	23				0,0001	0,10%	0,0500	67,988
54.216.1.231	2				0,0000	0,01%	0,0100	67,988
52.213.90.205	42				0,0002	0,18%	0,0300	65,741
52.210.196.93	3861				0,0218	16,24%	0,3500	159,656
51.145.123.29	2				0,0000	0,01%	0,0100	6,887
46.51.165.123	68				0,0004	0,29%	0,0400	0,405
3.248.99.37	2				0,0000	0,01%	0,0100	0,458
192.36.144.22	2				0,0000	0,01%	0,0100	15,070
192.168.1.185	19727				0,1115	82,99%	0,5400	22,535
132.163.96.6	2				0,0000	0,01%	0,0100	15,070
131.107.13.100	4				0,0000	0,02%	0,0100	31,274
129.6.15.29	4				0,0000	0,02%	0,0100	29,284
129.6.15.28	4				0,0000	0,02%	0,0100	22,258
128.138.140.44	1				0,0000	0,00%	0,0100	71,302
103.242.68.68	2				0,0000	0,01%	0,0100	3,477
103.126.53.123	1				0,0000	0,00%	0,0100	95,596

Adresses qui envoient des paquets à la caméra

Scénario no-app-no-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	28				0,0002	100%	0,0200	32,921
51.145.123.29	3				0,0000	10,71%	0,0100	40,731
46.51.165.123	9				0,0001	32,14%	0,0200	32,921
192.36.144.22	2				0,0000	7,14%	0,0100	50,739
132.163.96.6	3				0,0000	10,71%	0,0100	43,571
129.6.15.29	1				0,0000	3,57%	0,0100	33,535
129.6.15.28	1				0,0000	3,57%	0,0100	166,145
128.138.140.44	3				0,0000	10,71%	0,0100	33,637
103.242.68.68	3				0,0000	10,71%	0,0100	42,240
103.126.53.123	3				0,0000	10,71%	0,0100	4,351

Scénario no-app-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	64				0,0004	100%	0,0400	15,305
52.213.90.205	34				0,0002	53,12%	0,0400	15,305
52.18.5.103	8				0,0001	12,50%	0,0400	49,200
51.145.123.29	1				0,0000	1,56%	0,0100	115,892
46.51.165.123	9				0,0001	14,06%	0,0200	42,679
3.250.192.158	1				0,0000	1,56%	0,0100	43,465
3.248.99.37	2				0,0000	3,12%	0,0100	43,264
192.36.144.22	1				0,0000	1,56%	0,0100	125,694
192.168.137.1	1				0,0000	1,56%	0,0100	49,730
132.163.96.6	1				0,0000	1,56%	0,0100	122,519
129.6.15.29	1				0,0000	1,56%	0,0100	167,681
129.6.15.28	1				0,0000	1,56%	0,0100	156,926
128.138.140.44	1				0,0000	1,56%	0,0100	47,664
103.242.68.68	1				0,0000	1,56%	0,0100	116,174
103.126.53.123	2				0,0000	3,12%	0,0100	21,347

Scénario app-no-move-4G

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	3037				0,0176	100%	0,1200	130,052
52.213.90.205	11				0,0001	0,36%	0,0400	115,611
52.210.196.93	2978				0,0173	98,06%	0,1200	130,052
52.18.5.103	8				0,0000	0,26%	0,0400	35,430
51.145.123.29	1				0,0000	0,03%	0,0100	124,652
46.51.165.123	33				0,0002	1,09%	0,0300	149,505
192.36.144.22	1				0,0000	0,03%	0,0100	137,115
192.168.137.1	1				0,0000	0,03%	0,0100	35,958
132.163.96.6	1				0,0000	0,03%	0,0100	141,342
128.138.140.44	1				0,0000	0,03%	0,0100	60,005
103.242.68.68	1				0,0000	0,03%	0,0100	124,935
103.126.53.123	1				0,0000	0,03%	0,0100	78,438

Scénario app-no-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	3087				0,0170	100%	0,1600	12,248
52.210.196.93	295				0,0016	9,56%	0,1100	14,294
51.145.123.29	2				0,0000	0,06%	0,0100	0,163
46.51.165.123	69				0,0004	2,24%	0,0300	73,179
3.248.99.37	8				0,0000	0,26%	0,0100	4,159
192.36.144.22	2				0,0000	0,06%	0,0100	11,740
192.168.1.185	2706				0,0149	87,66%	0,0800	40,221
132.163.96.6	2				0,0000	0,06%	0,0100	5,489
128.138.140.44	1				0,0000	0,03%	0,0100	67,442
103.242.68.68	1				0,0000	0,03%	0,0100	131,961
103.126.53.123	1				0,0000	0,03%	0,0100	41,125

Scénario app-move-4G

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	8206				0,0460	100%	0,2300	87,660
52.213.90.205	33				0,0002	0,40%	0,0400	3,994
52.210.196.93	8120				0,0455	98,95%	0,2300	87,660
51.145.123.29	1				0,0000	0,01%	0,0100	158,034
46.51.165.123	47				0,0003	0,57%	0,0200	0,205
192.36.144.22	1				0,0000	0,01%	0,0100	175,414
132.163.96.6	1				0,0000	0,01%	0,0100	169,199
128.138.140.44	1				0,0000	0,01%	0,0100	95,132
103.242.68.68	1				0,0000	0,01%	0,0100	150,954
103.126.53.123	1				0,0000	0,01%	0,0100	115,509

Scénario app-move-wifi

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Source IPv4 Addresses	6596				0,0373	100%	0,2400	76,057
99.80.203.8	7				0,0000	0,11%	0,0200	67,949
54.216.1.231	2				0,0000	0,03%	0,0100	68,156
52.213.90.205	35				0,0002	0,53%	0,0400	4,053
52.210.196.93	1897				0,0107	28,76%	0,1800	160,018
51.145.123.29	2				0,0000	0,03%	0,0100	7,023
46.51.165.123	65				0,0004	0,99%	0,0300	0,367
3.248.99.37	2				0,0000	0,03%	0,0100	0,571
192.36.144.22	2				0,0000	0,03%	0,0100	15,212
192.168.1.185	4577				0,0259	69,39%	0,1000	2,528
132.163.96.6	2				0,0000	0,03%	0,0100	15,244
129.6.15.29	1				0,0000	0,02%	0,0100	176,092
128.138.140.44	1				0,0000	0,02%	0,0100	71,536
103.242.68.68	2				0,0000	0,03%	0,0100	3,859
103.126.53.123	1				0,0000	0,02%	0,0100	95,907

Définitions

1. **Firmware** : type de logiciel installé sur un appareil électronique, tel qu'un routeur ou un téléphone portable, qui contrôle son fonctionnement interne et peut être mis à jour pour améliorer ses performances.
2. **WPA/WPA2-PSK** : protocole de sécurité utilisé pour protéger les réseaux Wi-Fi domestiques et professionnels. Il utilise un mot de passe partagé pour chiffrer les communications entre les appareils connectés et le routeur Wi-Fi.
3. **WEP/WPA/WPA2** : WEP, WPA et WPA2 sont des protocoles de sécurité utilisés pour protéger les réseaux Wi-Fi. WEP est le plus ancien et le moins sécurisé, tandis que WPA et WPA2 offrent une meilleure sécurité en utilisant des algorithmes de chiffrement plus forts.
4. **Monitoring** : mode opératoire dans lequel une carte réseau sans fil peut être configurée pour écouter tous les paquets de données transitant sur une fréquence ou un canal radio spécifique, sans être associée à un réseau sans fil.
5. **Alpha Network AWUS036NHA** : carte réseau sans fil USB qui prend en charge les normes Wi-Fi 802.11 b/g/n et offre une portée de signal étendue. Elle supporte également le mode monitoring.
6. **Wireshark** : logiciel open-source de capture et d'analyse de paquets de données réseau. Il permet aux utilisateurs de visualiser le trafic réseau en temps réel et d'analyser les problèmes de réseau.
7. **DNS** : système qui permet de traduire les noms de domaine en adresses IP. Il permet aux utilisateurs d'accéder à des sites web en entrant simplement leur nom de domaine, plutôt que leur adresse IP numérique.
8. **DHCP** : protocole réseau qui permet aux appareils de se connecter automatiquement à un réseau et de recevoir une adresse IP et d'autres paramètres de configuration réseau.
9. **IPv4** : protocole de communication Internet qui utilise des adresses IP à 32 bits. Il est largement utilisé mais limité en termes de nombre d'adresses IP disponibles.
10. **UDP** : protocole de communication réseau qui permet aux applications de transférer des données sans établir de connexion préalable. Il est souvent utilisé pour les applications de streaming et de jeu en ligne.
11. **NTP** : protocole de synchronisation de l'horloge qui permet aux appareils de maintenir l'heure et la date précises en se synchronisant avec des serveurs de temps.
12. **TCP** : protocole de communication réseau qui établit une connexion fiable et assure la livraison des données sans perte ni duplication.
13. **AES** : algorithme de chiffrement utilisé pour protéger les données en transit ou stockées sur des dispositifs électroniques.
14. **Convertisseur USB vers TTL** : qui permet de connecter un appareil électronique avec une interface TTL à un ordinateur via un port USB.
15. **PuTTY** : émulateur de terminal open-source qui permet aux utilisateurs de se connecter à des serveurs distants via SSH, Telnet, Rlogin et d'autres protocoles de communication.
16. **Ghidra** : logiciel d'analyse de code binaire développé par la National Security Agency (NSA). Il permet aux chercheurs en sécurité de décompiler et d'analyser les programmes exécutables pour identifier les vulnérabilités et les failles de sécurité.
17. **Binwalk** : outil d'analyse de firmware qui permet d'extraire des informations à partir de fichiers binaires, tels que des images disque, des firmwares, des fichiers compressés, etc.
18. **U-Boot** : chargeur d'amorçage open-source utilisé pour démarrer des systèmes embarqués tels que des routeurs, des modems, des consoles de jeu, etc.

19. **cp et dd** : cp et dd sont des commandes Unix/Linux utilisées pour copier des fichiers. La commande "cp" copie des fichiers d'un emplacement à un autre, tandis que la commande "dd" peut copier des fichiers ou des partitions d'un disque dur à un autre.
20. **setenv** : commande utilisée pour définir des variables d'environnement dans U-Boot. Les variables d'environnement sont des paramètres qui contrôlent le comportement du système.
21. **Busybox** : ensemble d'outils Unix/Linux utilisé pour créer des systèmes embarqués légers et compacts. Il fournit des versions simplifiées des commandes Unix/Linux telles que ls, cp, rm, etc.
22. **/etc/passwd et /etc/shadow** : fichiers de configuration utilisés pour stocker les informations des utilisateurs et les mots de passe chiffrés dans les systèmes Unix/Linux.
23. **OpenWrt** : système d'exploitation open-source pour les routeurs sans fil et autres appareils embarqués. Il est conçu pour être flexible et personnalisable, permettant aux utilisateurs d'ajouter des fonctionnalités et de personnaliser le système en fonction de leurs besoins.
24. **OpenBSD** : un système d'exploitation open-source dérivé de BSD Unix. Il est connu pour sa sécurité renforcée, sa documentation complète et sa communauté de développement active.
25. **RTSP** : protocole de streaming en temps réel utilisé pour diffuser de l'audio et de la vidéo en direct sur Internet.
26. **HTTP, HTTPS** : HTTP (Hypertext Transfer Protocol) et HTTPS (Hypertext Transfer Protocol Secure) sont des protocoles de communication utilisés pour transférer des données entre les navigateurs web et les serveurs web. HTTPS utilise une couche de chiffrement supplémentaire pour sécuriser les données transférées.
27. **SSL, TLS** : SSL (Secure Sockets Layer) et TLS (Transport Layer Security) sont des protocoles de sécurité utilisés pour sécuriser les communications sur Internet. Ils sont utilisés pour chiffrer les données transférées entre les navigateurs web et les serveurs web.
28. **Requête POST** : méthode utilisée pour envoyer des données à un serveur web. Elle est utilisée pour soumettre des formulaires web et envoyer des données dans le corps de la requête HTTP.
29. **MD5** : MD5 (Message-Digest Algorithm 5) est une fonction de hachage cryptographique largement utilisée pour vérifier l'intégrité des données.