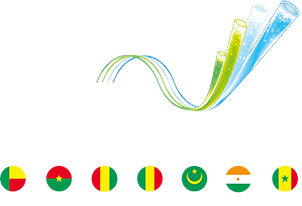
|  |
| --- |
|  |
| Membres du groupe :Fatima Léonie CISSEMohamed Lamarana DIALLO |





***PROJET PYTHON RESEAU***

**Annee:** 2021 – 2022

**Formateur**: Mr. kuassi

# RAPPORT DE PROJET

# Ce rapport a été rédigé dans l’optique de présenter l’ensemble des aspects fonctionnels lors de la conception de l’environnement de travail (Installation-Mise en Réseau-Implémentation Services (DHCP-DNS-IREDMAIL)) et la réalisation des différents scripts demandés (Installation-Dépendances, Communication CLIENT/SERVEUR, Analyse Paquets et Envoie Mail). Un travail réalisé suivant une logique qui sera décrite juste ci-dessous.

* **Présentation du Projet**

Le projet consiste à mettre en place un dispositif permettant d’analyser un packet.

Il était alors question durant la réalisation de ce projet de fonctionner en environnement virtuel en simulant un petit réseau constitué d’un **SERVEUR** avec une adresse fixe ou sera configurées les Services

* DNS
* DHCP
* MAIL
* Une Base de Données

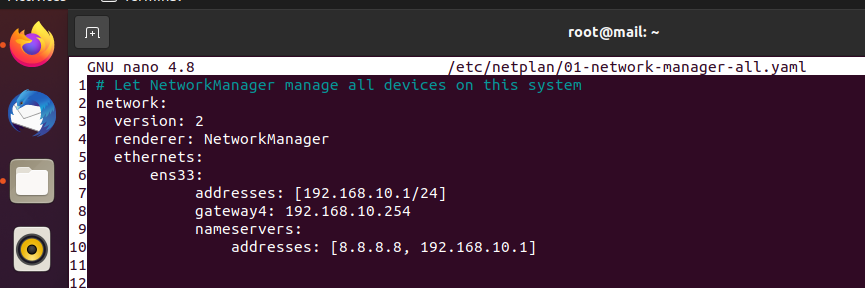
Une machine **cliente** qui bénéficiera de l’ensemble des services énumérées ci-dessous et établiera une communication TCP/IP avec le serveur lors des authentifications ou création de comptes.Une communication qui sera générée par nos scripts écrits en python et sera suivi par une machine kali **Man in the middle** qui grâce à son outil Wire Shark intégré permettra de voir toutes les informations de communication qui seront analysées, exportées puis envoyées par mail à l’Admin grâce à un script.

Sur ce, le travail sera réparti en 5 grandes parties :

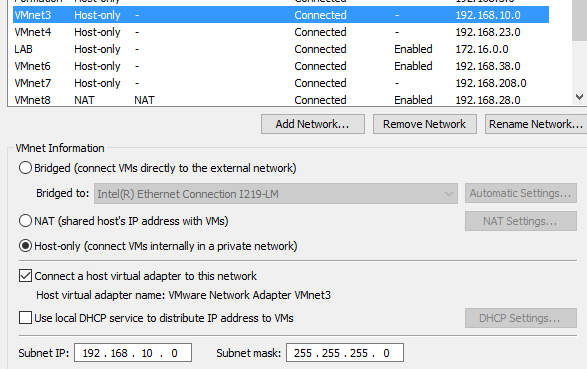
* Implémentation et mise en réseau de l’environnement virtuel
* Implémentation puis tests de nos services DHCP,DNS sur le serveur
* Implémentation et test du script d’installation des dépendances MYSQL/IREDMAIL
* Mise en place et tests des scripts Client/Serveur pour l’authentification
* Simulation Wireshark et mise en place du module PYTHON
* **PHASE PRATIQUE**
* **Implémentation et Mise en réseau de l’environnement virtuel**

Notre environnement est constitué de 3 machines virtuelles installées sur VMWare

* Une machine serveur tournant sur du Ubuntu 20.04 avec une adresse fixée à 192.168.10.1



* Une machine cliente avec comme OS aussi Ubuntu 20.04 car étant la version la plus stable avec une adresse dynamique mise en réseau avec le serveur par un réseau privé HOST-Only (192.168.10.0/24)
* Une machine Man In The Middle tournant sur du kali linux réputé par ses outils de sécurité intégrés et donc des analyseurs de communications préinstallés comme Wireshark.Elle est aussi mise dans le même réseau Client/serveur et est aussi adressée Dynamiquement



* **Implémentation puis tests de nos services DHCP, DNS sur le serveur**

Nos machines cliente et kali étant configurés dynamiquement, elles bénéficieront des adresses distribuées par le service DHCP du serveur et pourront aussi joindre le nom de domaine affecté au serveur.

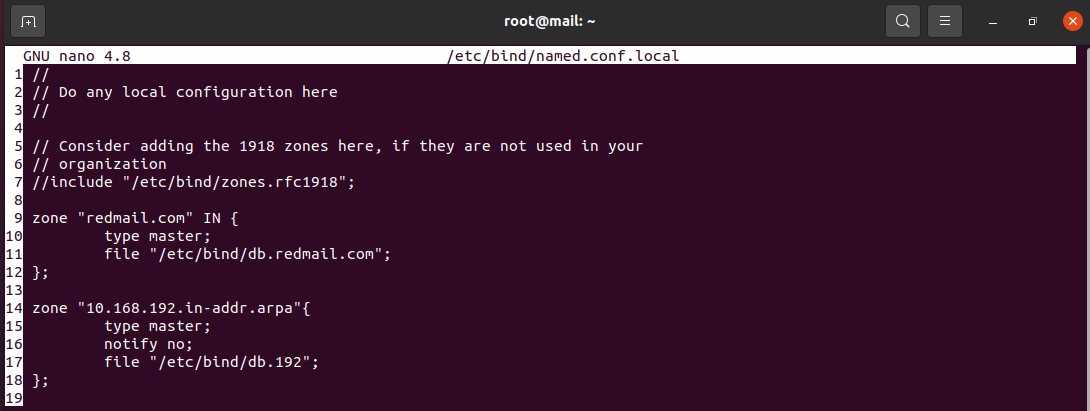
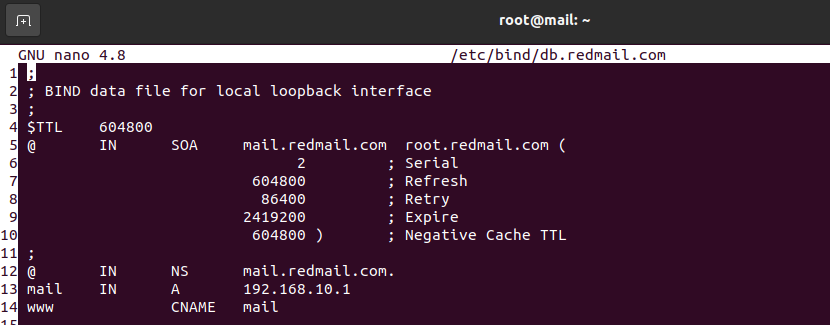
* **SERVICE DNS**

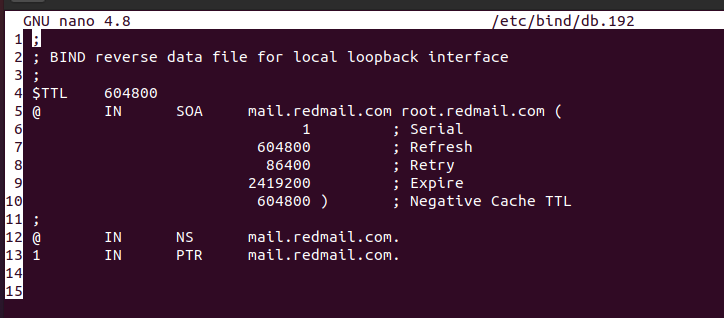
Notre configuration DNS prendra en compte le service principal de Mailing qui sera implémenté par la suite, d’où le choix du nom de domaine **mail.redmail.com** pour notre serveur

* **Redéfinition du nom de la machine**

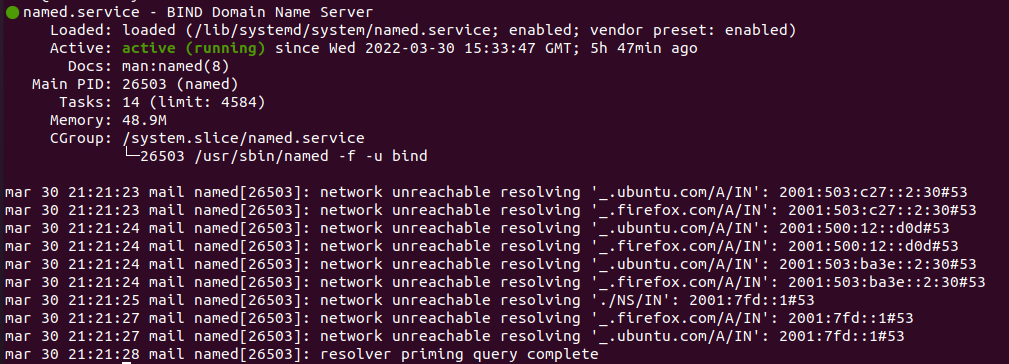


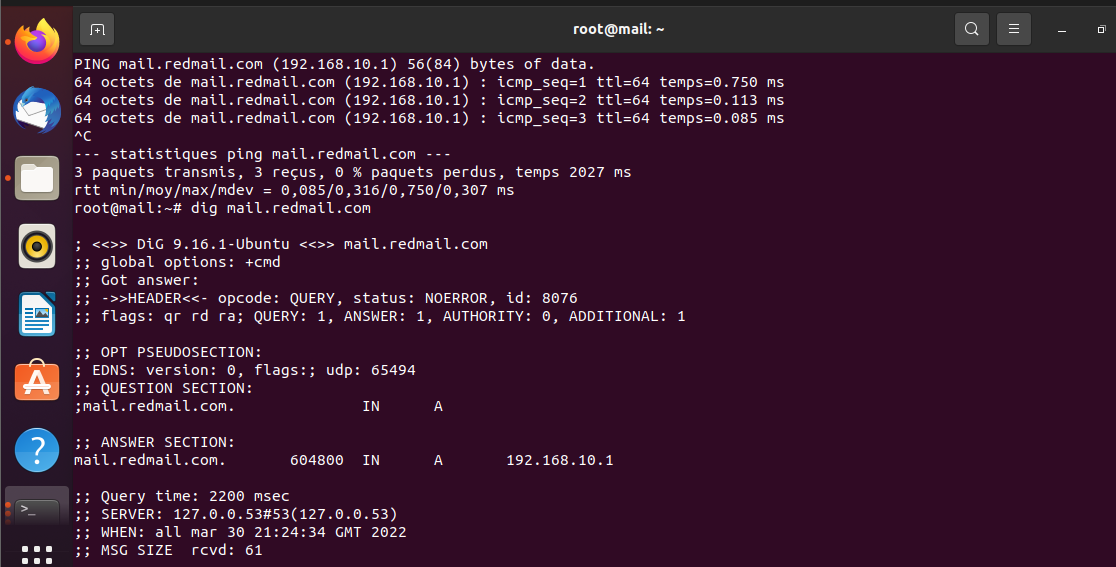
* **Nos fichiers de configurations DNS**



* **Statut et Test de la configuration DNS**

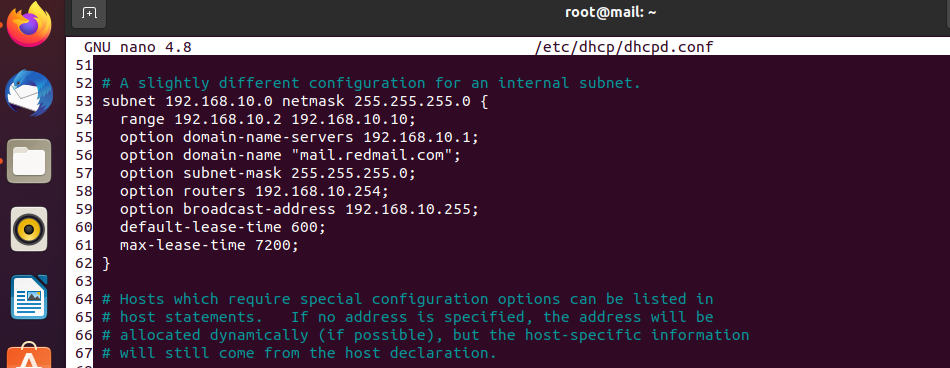




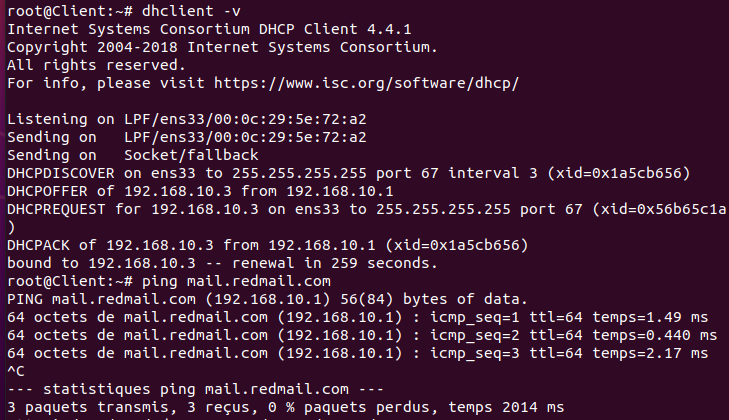
* **SERVICE DHCP**

Après implémentation et test fonctionnel de notre service DNS nous avons implémenté notre DHCP. La plage réseau étant définie de 192.168.10.2 à 192.168.10.10, il nous fallait juste installer le service isc-dhcp-service existant sur les dépôts Ubuntu puis éditer le fichier de configuration avant d’effectuer les tests au niveau du client.

* **Notre fichier de configuration DHCP**

n

* **Test de la configuration DHCP**



Donc cela montre que notre machine client a pu recevoir l’adresse 192.168.10.3 de notre serveur DHCP et arrive à pinger notre nom de Domaine d’où une vérification réussie de la fonctionnalité de nos 2 services.

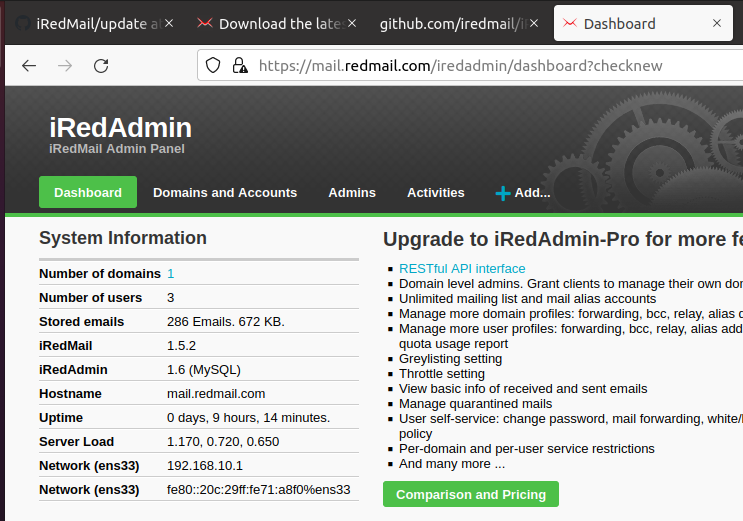
* **Implémentation et test du script d’installation des dépendances MYSQL/IREDMAIL**



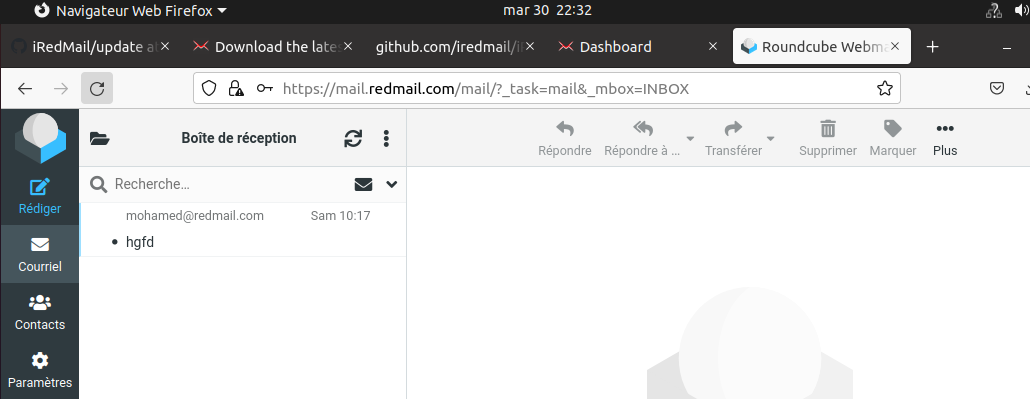
* Ce script d’installations des dépendances stocke dans une variable l’ensemble des packages dont dépendra la base de données MySQL ainsi nous avons les lib dont dépend la plupart des package sous Ubuntu, suivi du package MySQL-client qui indispensable au lancement de Mysql-Server-8.0 qui est la dernière version stable de MySQL sur les dépôts Ubuntu.
* Le même processus a été adopté pour le service mail IREDMAIL qui est un serveur de messagerie complet avec tous ses composants :
* Serveur SMTP Postfix
* Serveur IMAP DOVECOT
* **Serveur Web Nginx** pour servir le panneau d'administration et le Webmail
* OpenLDAP, MySQL/MariaDB ou PostgreSQL pour le stockage des informations utilisateur
* Amavised-nouveau pour la signature et la vérification DKIM
* SpamAssassin pour l'anti-spam
* ClamAV pour l'antivirus
* RoundCube de messagerie Web
* Groupware SOGo, fournissant le Webmail,
* Fail2ban pour protéger SSH
* Gestionnaire de liste de diffusion mlmmj
* Serveur de politique iRedAPD Postfix pour la liste grise

Après Avoir lancé le script d’installation des dépendances nous avons intégré dans le script le processus d’installation complet d’IREDMAIL qui est lui-même un script qui installera et configura l’ensemble des composants du serveur de messagerie jusqu’à qu’on ait accès à travers notre domaine :

* Interface admin de gestion des comptes mails

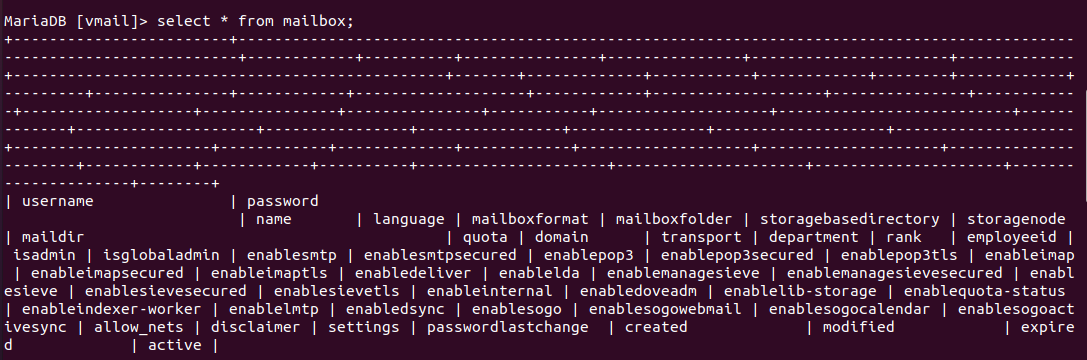


* Interface des comptes clients mail



* **Mise en place et tests des scripts Client/Serveur pour l’authentification**

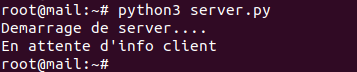
Après avoir géré la mise en place du service mail, une base de données MariaDB a été installée stockant l’ensemble des compte mails crées pour l’ensemble des utilisateurs du service mail avec leur mot de passes et les informations liées à leur compte. Ainsi la logique actuelle est qu’un utilisateur créé tente de se connecter depuis notre machine cliente tout en s’authentifiant au niveau du serveur bien avant de pouvoir accéder au service mail. D’où l’intérêt de nos scripts **server.py** et **client.py**



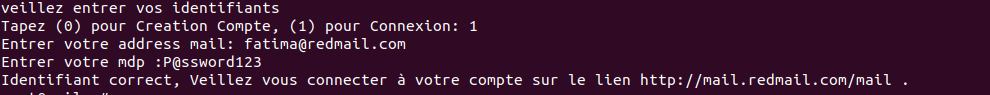
Il s’agit de deux scripts python qui seront lancés successivement depuis le serveur puis le client initiant une connexion socket entre ces 2 puis gérant les fonctions d’authentification ou de création de compte pour les utilisateurs dont leurs identifiants sont stockés dans la base de données vmail de iRedMail à laquelle on fera appel le long du script.

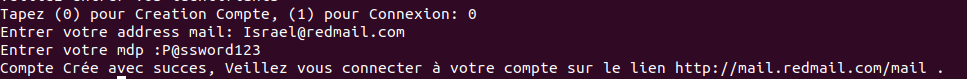
Suite à une connexion réussie l’interaction client-serveur permettra au serveur de demander au client s’il veut se connecter ou créer un compte. Selon l’option choisi par le client, une interface lui sera présentée pour qu’il fournit les identifiants et après envoyer. Par la suite, une insertion dans la base sera faite ou un lien sera proposé au client pour qu’il puisse accéder à son interface de connexion de son compte mail.

* DEMONSTRATION
* SCRIPT SERVER.PY

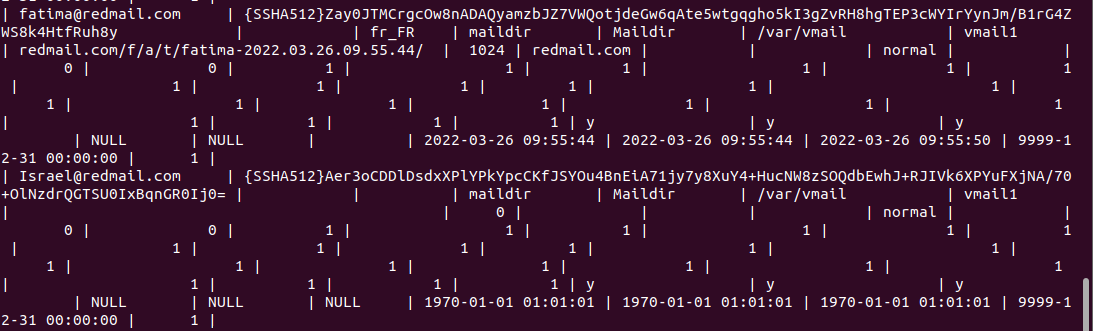


* SCRIPT CLIENT.PY

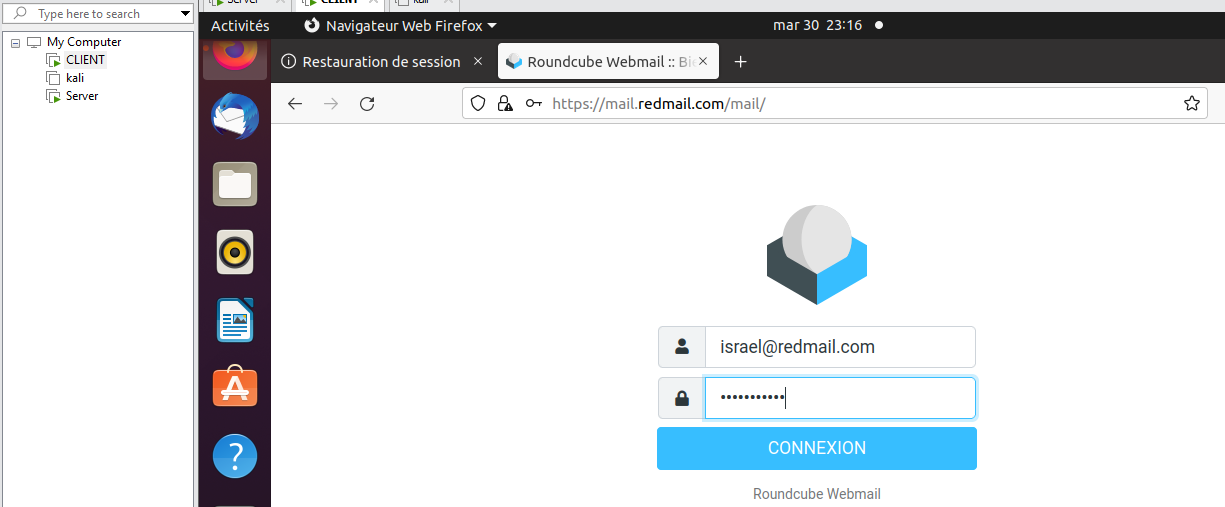




* Compte crée dans la base

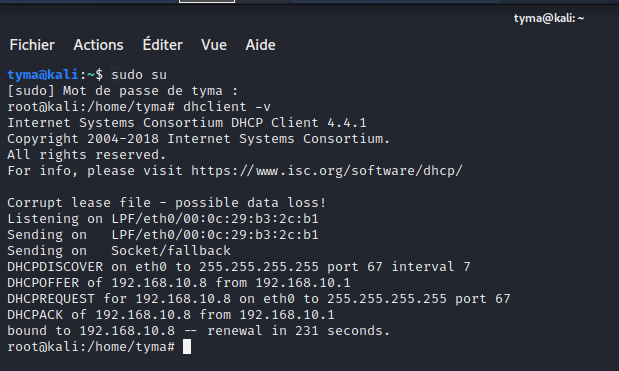


En cliquant sur le lien proposé le client accède directement à sa page de connexion et donc bénéficiera de l’accès à distance du serveur puis l’accès à son service mail suite à l’exécution des scripts



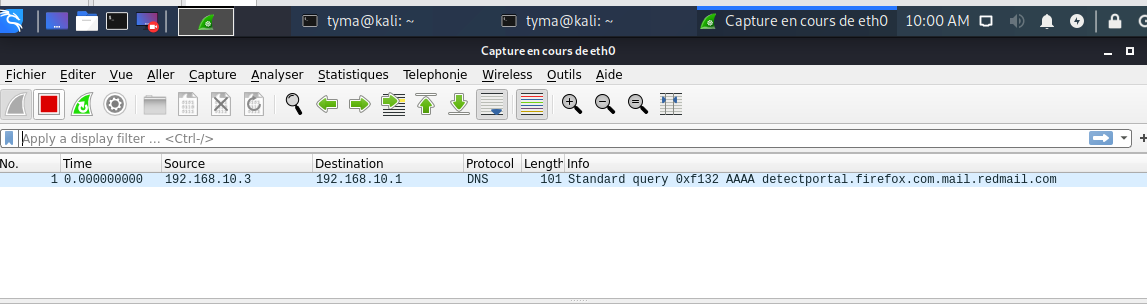
* **Simulation Wireshark et mise en place du module PYTHON**

Dans cette partie, on note l’intervention de notre 3e machine virtuelle kali. Etant configuré en dynamique il cherchera à obtenir une Adresse IP depuis notre serveur.

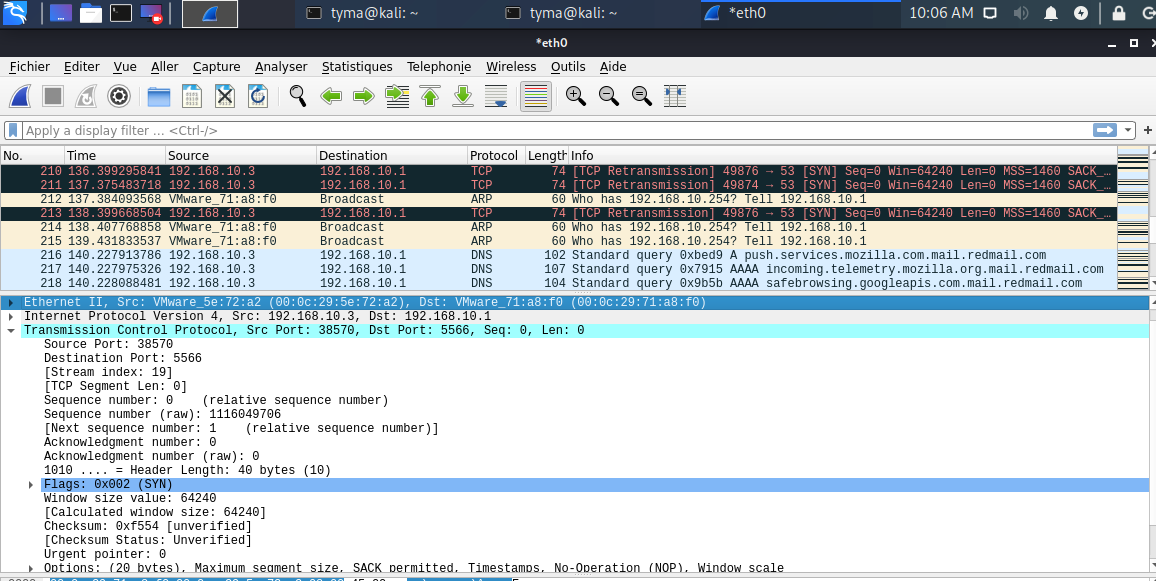


Donc là nous avons notre machine kali qui à pu intégrer le réseau Client/Serveur à travers l’adresse 192.168.10.8 fourni par notre serveur DHCP.

Ainsi étant dans le réseau, on lance notre outil wireshark qui écoutera tout ce qui se passe dans le réseau 192.168.10.0



Pendant ce temp, nous exécuterons nos 2 scripts Client/Serveur et observer les captures de wireshark.

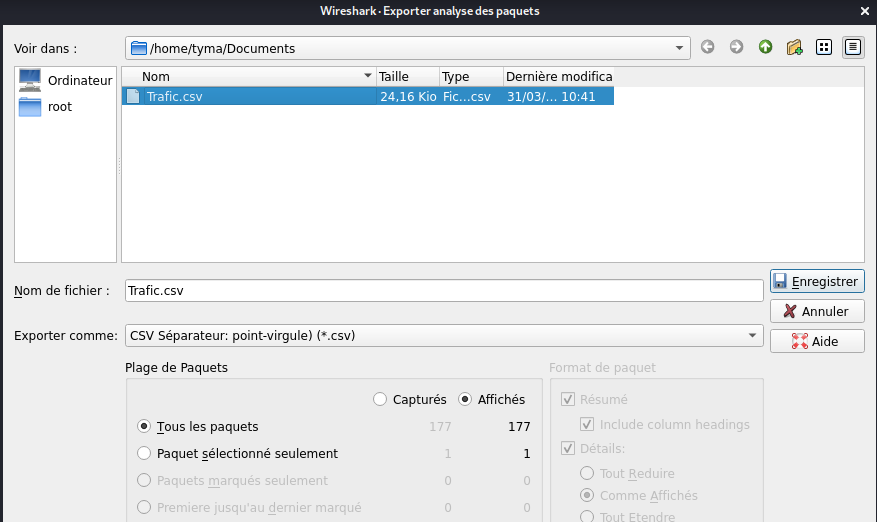


A travers cette capture on peut voir l’outil wireshark qui capture le flux TCP/IP entre le client et le serveur et une vue plus globale de la structure de la trame permet de voir les information depuis

* La couche physique avec les adresse mac de chaque machine source/destination
* La couche réseau avec les adresses IP client(192.168.10.3)/serveur(192.168.10.1)
* Et la couche transport qui nous intéresse avec les ports source et destination(5566 = port d’écoute défini dans nos scripts)

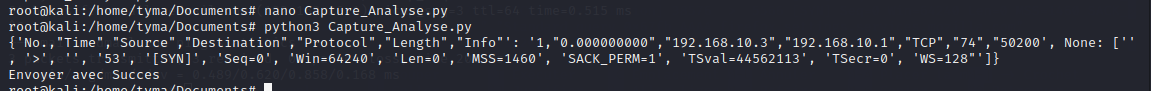
Ainsi wireshark nous permet de voir tout ce qui passe lors de la communication client serveur avec toutes les services requises(DNS,DHCP,WEB)

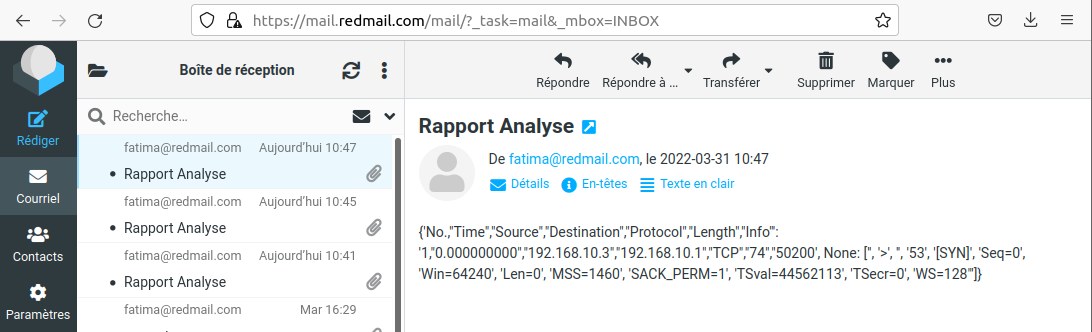
Toutes ces informations seront extraits dans un fichier csv qui sera recupérés par notre dernier script qui se chargera d’analyser puis d’envoyer tout ce qu’il aura comme informations capitales dans cette communication.



Le script **Capture\_Analyse.py** recupere les informations de l’analyse du reseau contenu dans le fichier **Trafic.csv** puis l’envoi en utilisant le service de messagerie mis en place dans la boite de **mohamed@redmail.com**.







On constate ici que le mail a été envoyer avec succes dans la boite de [**mohamed@redmail.com**](mailto:mohamed@redmail.com)avec l’analyse du reseau.

**CONCLUSION**

Ce projet ayant pour objectif la mise en place d’un service de messagerie et l’analyse du reseau.Nous avons mis en place un script shell pour l’installation des dependances et des scripts python pour la simulation du service de messagerie et l’envoi de l’anlyse reseau par mail. Enfin la creation d’un depot github pour une sauvegarde du projet.