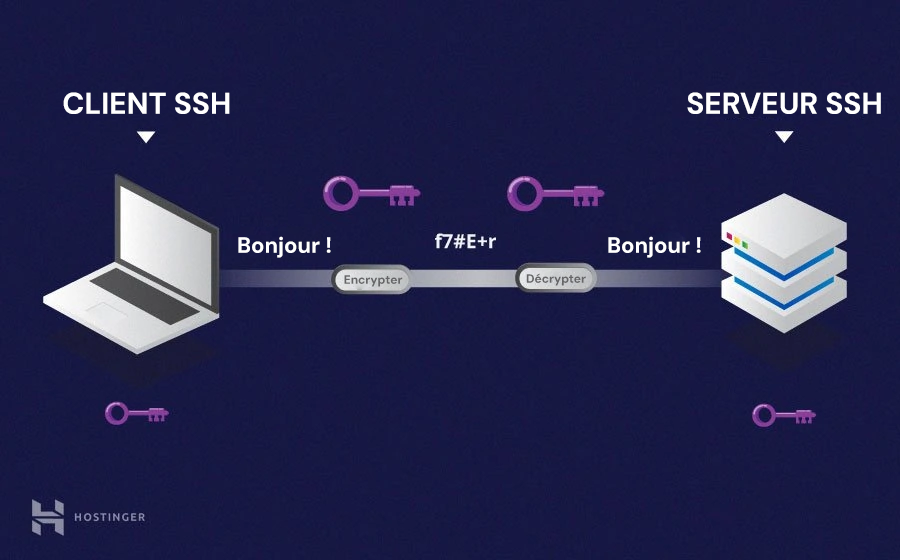
UNIVERSITE ABDELMALEK ESSAADI ECOLE NATIONALE DES SCIENCES

APPLIQUEES AL-HOCEIMA



Rapport : La configuration de SSH & RD



**REALISE PAR :**

Hicham Azeroual **ENCADRE PAR :**

Mohamed l’aamarti PR. MOHAMED CHERRADI

Deuxième année en Génie informatique

**Année Universitaire : 2024/2025**

# TABLE DES MATIERES

## SSH

* 1. **Introduction**
  2. **Histoire du protocole SSH**
  3. **Outils d’OpenSSH**
  4. [Mécanismes du protocole SSH 6](#_bookmark0)
  5. [Implémentation de SSH 6](#_bookmark1)
     1. Mise à jour des paquets 6
     2. Installation de SSH 6
     3. Vérification de l’installation de SSH 7
     4. Connexion des machines avec SSH 7
     5. Connexion d’un client Windows avec le serveur UbuntoSSH …9
  6. [Conclusion 11](#_bookmark2)

1. [RDP](#_bookmark3)
   1. [Introduction 12](#_bookmark4)
   2. [Étapes détaillées pour configurer RDP 12](#_bookmark5)
      1. Installation de xrdp et démarrage du service 12
      2. Configuration du pare-feu 13
      3. Résolution des problèmes liés à SELinux 13
   3. [Connexion depuis un client RDP sous Windows 14](#_bookmark6)
   4. [Problèmes rencontrés 15](#_bookmark7)
   5. [Conclusion 16](#_bookmark8)

# Introduction

Dans un environnement informatique moderne, où coexistent plusieurs systèmes d'exploitation comme **Linux**, **Windows** et **MacOS**, la connectivité distante devient un facteur clé pour assurer une gestion optimale des ressources et des serveurs. Cette connectivité est cruciale non seulement pour l’administration des systèmes, mais aussi pour les interactions quotidiennes entre machines, permettant ainsi de surmonter les barrières liées aux différentes plateformes.

#### 1.1Protocoles Utilisés pour la Connexion Distante :

* **SSH (Secure Shell)** est un protocole de communication sécurisé qui permet d’établir une connexion à distance via la ligne de commande. Il est principalement utilisé pour l’administration des serveurs, notamment sur les systèmes Unix-like (Linux, MacOS), offrant une solution fiable et sécurisée pour exécuter des commandes à distance et transférer des fichiers.
* **RDP (Remote Desktop Protocol)**, quant à lui, est un protocole graphique qui permet d’accéder à une interface utilisateur distante. Il est couramment utilisé dans les environnements Windows, permettant aux utilisateurs de voir et interagir avec le bureau d’une machine distante comme s’ils y étaient physiquement.

Ces deux protocoles, bien qu’ayant des fonctions différentes, sont souvent déployés ensemble dans des environnements hybrides où les utilisateurs ont besoin à la fois de connexions sécurisées en ligne de commande et d'accès graphique.

* 1. **Objectifs**

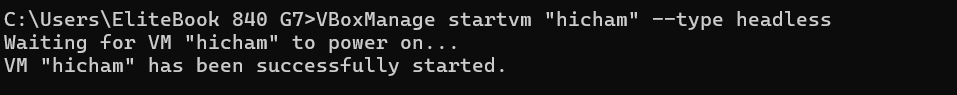
Ce rapport se concentre sur les configurations et l’utilisation des protocoles **SSH** et **RDP** dans un environnement hybride. Plus précisément, il a pour objectifs de :

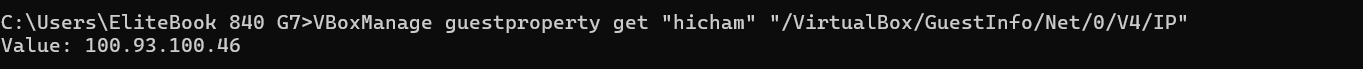
1. **Configurer une connexion sécurisée via SSH**, une étape essentielle pour les administrateurs système et les développeurs, afin de garantir un accès sécurisé à distance aux serveurs et à leurs ressources.
2. **Mettre en place l’accès graphique avec RDP**, qui permet d’interagir avec des machines distantes en utilisant une interface graphique, ce qui est particulièrement utile pour les utilisateurs ayant besoin d’une interaction plus visuelle avec leurs systèmes.
3. **Faciliter le transfert de données entre différentes machines**, en assurant que les protocoles et outils de transfert soient correctement configurés, ce qui simplifie les échanges entre systèmes Linux, Windows et MacOS.
   1. **Importance**

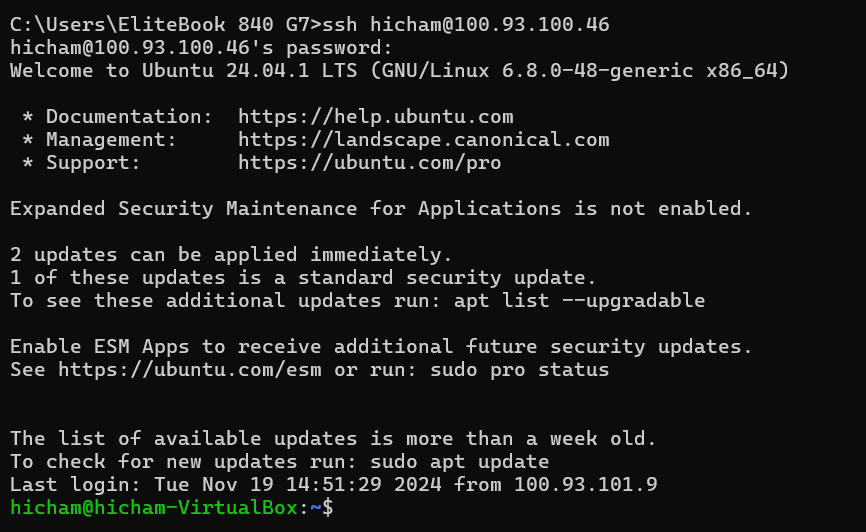
La mise en place correcte de **SSH** et **RDP** dans un environnement professionnel ou personnel est essentielle pour plusieurs raisons clés :

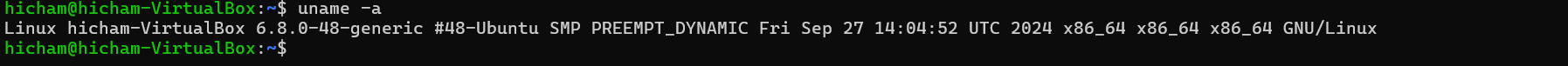
* **Optimisation de la productivité** : Ces protocoles offrent un moyen efficace d’accéder à des machines distantes. SSH permet une gestion à distance rapide via des commandes en ligne, tandis que RDP offre une expérience visuelle intuitive pour interagir avec des machines distantes. Cette combinaison garantit une flexibilité maximale pour les utilisateurs ayant différents besoins en matière de gestion des systèmes.
* **Renforcement de la sécurité** : L'utilisation de SSH pour les connexions garantit que les données échangées sont cryptées, protégeant ainsi les informations sensibles contre toute interception. D’autre part, RDP peut être sécurisé en utilisant des stratégies de réseau telles que les VPN et des pare-feu afin de restreindre l'accès aux utilisateurs autorisés uniquement.
* **Facilitation des interactions dans des environnements multi-OS** : Dans des configurations où plusieurs systèmes d'exploitation coexistent (par exemple, Linux pour les serveurs et Windows pour les stations de travail), l’intégration de SSH et RDP permet de gérer facilement les interactions entre ces différents systèmes, en assurant compatibilité et fluidité des opérations.

# Système d'exploitation utilisé

****

****

****

****

Dans le cadre de ce rapport, le système d'exploitation **Ubuntu** a été sélectionné pour illustrer la configuration des protocoles **SSH** et **RDP**. Voici les caractéristiques techniques de la machine utilisée :

* **Nom du système** : Ubuntu
* **Version du noyau Linux** : 6.8 (GA)
* **Architecture** : x86\_64
* **Version de GNU/Linux** : Ubuntu 24.04 LTS

Ce choix s'explique par la stabilité reconnue d'Ubuntu, sa grande flexibilité, ainsi que son excellent support des outils nécessaires à la mise en place des connexions distantes.



## SSH :

### Introduction :

Le protocole **SSH (Secure Shell)** est une norme de sécurité incontournable pour établir des connexions à distance sécurisées entre un client et un serveur sur un réseau potentiellement non sécurisé. SSH permet non seulement d’exécuter des commandes à distance, mais aussi de transférer des fichiers et d’administrer des machines via un canal entièrement chiffré. Créé pour remplacer des protocoles moins sécurisés comme Telnet et FTP, SSH est aujourd’hui un pilier pour les administrateurs système, les développeurs, et toute personne nécessitant un accès distant fiable et protégé.

### Histoire de protocole SSH :

Le protocole SSH a vu le jour en **1995**, développé par le chercheur finlandais **Tatu Ylönen**. Son objectif était de répondre aux limitations des protocoles existants, tels que Telnet et FTP, qui transmettaient des informations sensibles, comme des mots de passe, en clair sur les réseaux.

* **SSH-1** : La première version du protocole introduisait un mécanisme de chiffrement robuste et une authentification sécurisée, mais présentait encore des failles.
* **SSH-2** : Lancée en **2006**, cette version corrigeait les lacunes de SSH-1 et offrait des améliorations significatives, notamment une sécurité renforcée et de nouvelles fonctionnalités.

Depuis, SSH est devenu un **standard mondial** pour garantir la sécurité des réseaux modernes et l’administration à distance des systèmes.

### Outils de OpenSSH :

**OpenSSH** est une implémentation open-source du protocole SSH, largement utilisée dans les environnements Linux/Unix et compatible avec d’autres systèmes comme Windows et MacOS. OpenSSH offre une suite complète d’outils qui simplifient la gestion des connexions distantes sécurisées :

* **ssh** : Client principal pour établir des connexions à distance sécurisées.
* **sshd** : Serveur SSH qui gère et écoute les connexions entrantes.
* **scp** : Outil de transfert de fichiers sécurisé entre deux machines, basé sur le protocole SSH.
* **sftp** : Alternative moderne à SCP, avec une interface interactive pour gérer les fichiers à distance.
* **ssh-keygen** : Générateur de clés SSH pour une authentification sécurisée sans mot de passe, souvent privilégié pour sa simplicité et sa robustesse.

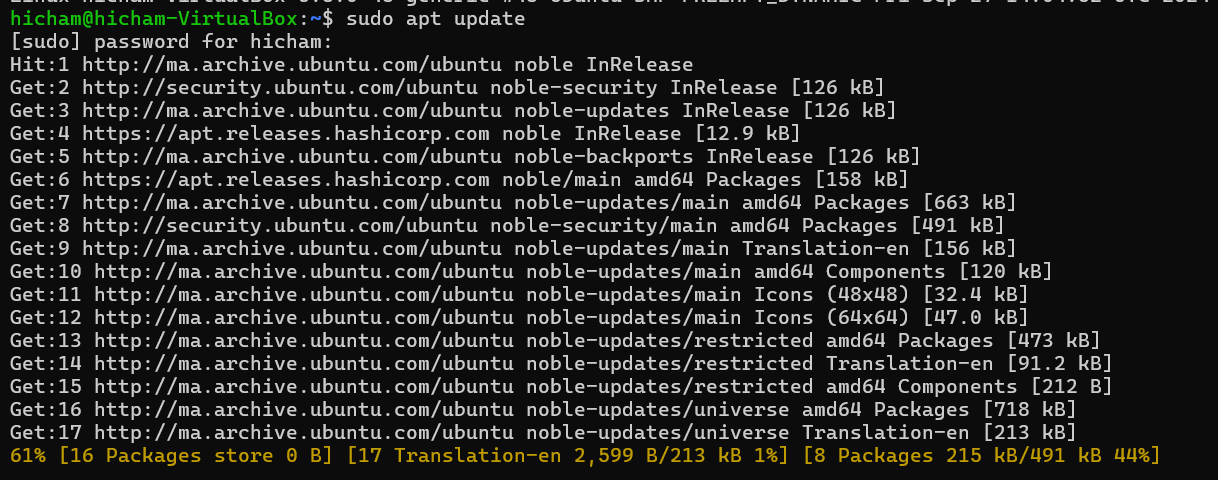
#### Mécanismes du protocole SSH :

Le protocole SSH repose sur plusieurs mécanismes techniques qui assurent la sécurité et la fiabilité des connexions :

* **Chiffrement** : SSH utilise des algorithmes avancés comme **AES**, **RSA**, **ECDSA**, ou **ED25519** pour protéger les données échangées contre toute interception.
* **Authentification** : Plusieurs méthodes d’authentification sont disponibles :
  + Authentification par mot de passe.
  + Authentification par clé publique/privée (comme RSA ou ED25519), garantissant un niveau de sécurité supérieur.
  + Utilisation de certificats pour des besoins spécifiques.
* **Tunneling sécurisé** : Redirection de trafic réseau via le canal SSH, permettant de sécuriser des protocoles vulnérables.
* **Port Forwarding (transfert de port)** : Permet de rediriger des ports spécifiques pour rendre les connexions plus flexibles, notamment pour des applications nécessitant des configurations complexes.
* Grâce à ces mécanismes, SSH assure une communication sécurisée et fiable, essentielle pour protéger les données sensibles et garantir une administration efficace des systèmes.

#### Implémentation de SSH :

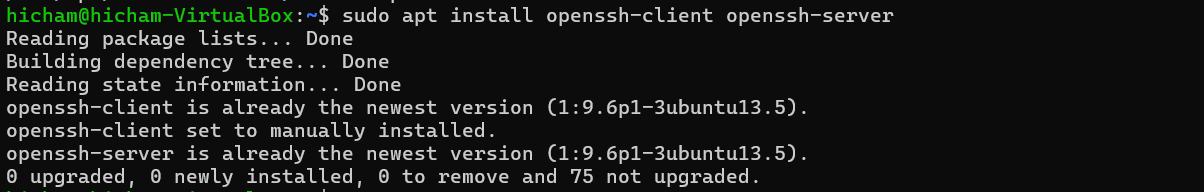
* + 1. Mise à jour de paquet :



En maintenant les paquets à jour, on établit une base solide pour l'installation et la configuration sécurisées de SSH. Une fois cette étape terminée, on peut passer à l'installation de l'outil SSH

* + 1. Installation de SSH :

Pour installer OpenSSH, on utilise la commande suivante :

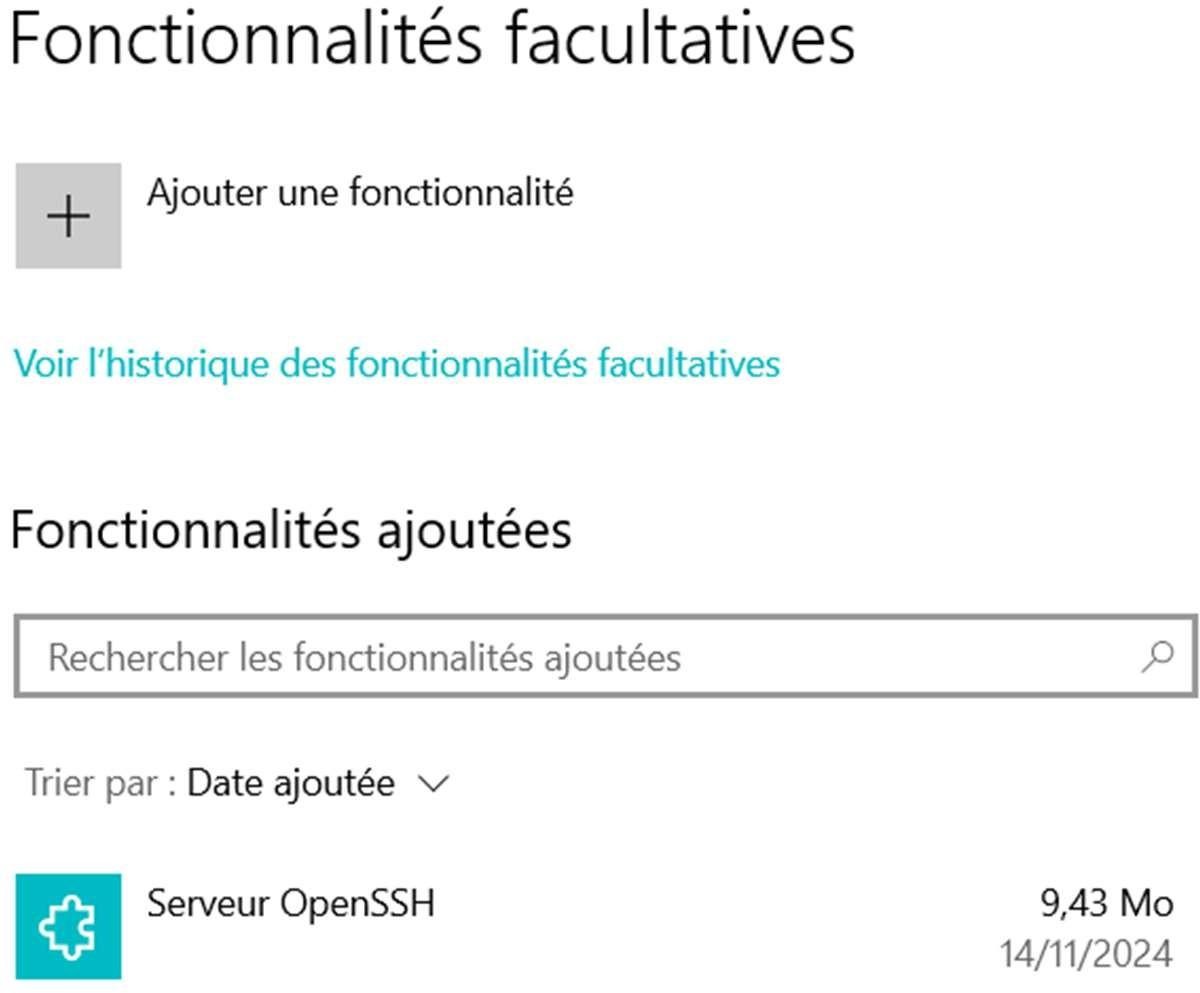


*cole Nationale des Sciences Appliquées d’Al-Hoceima*

* **OpenSSH-Server** : Configure la machine pour qu'elle fonctionne comme un serveur SSH, permettant d'accepter des connexions distantes sécurisées.
* **OpenSSH-Clients** : Fournit les outils nécessaires pour établir des connexions SSH vers d'autres machines de manière sécurisée.

Ensuite, il est nécessaire d'installer à la fois le client et le serveur SSH sur Windows via les **Fonctionnalités Facultatives**, disponibles dans les paramètres système.

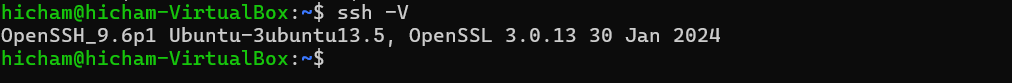
.



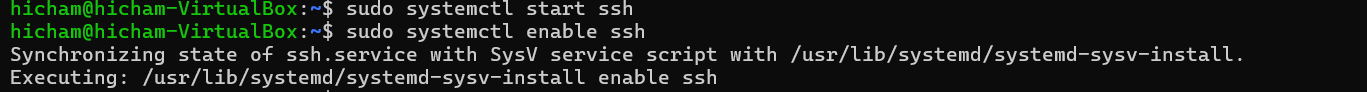
* + 1. Vérification d’installation de SSH :

Il est important de vérifier que les paquets ont été correctement installés. Cela peut être fait en exécutant la commande :

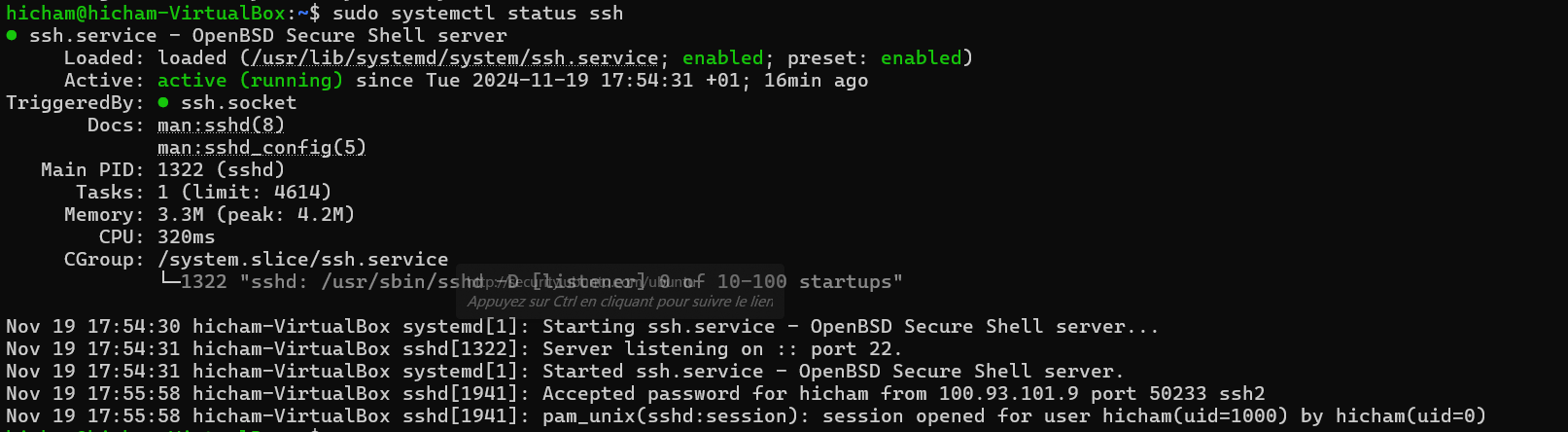
Cette commande affiche la version de SSH installée, confirmant que l'outil est prêt à être configuré et utilisé.



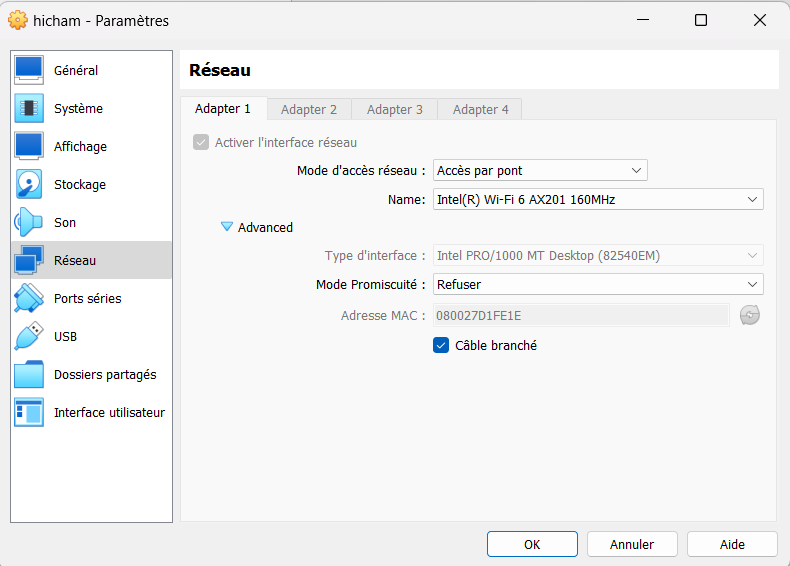
* + 1. Démarrez le service SSH :



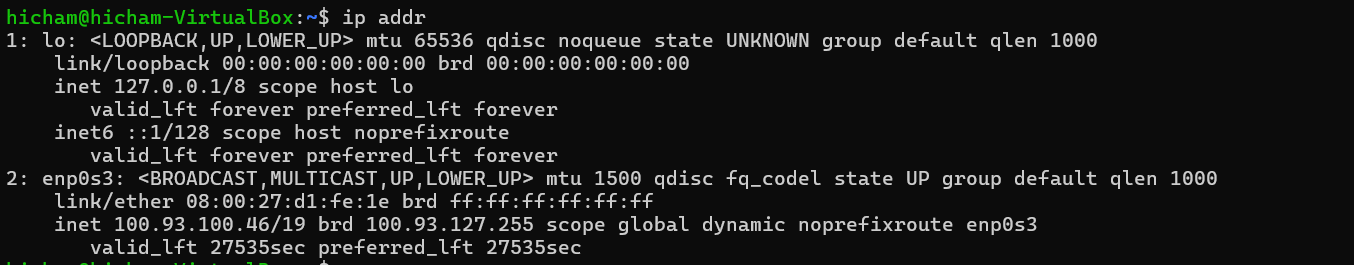
Vérifiez que le service SSH fonctionne correctement

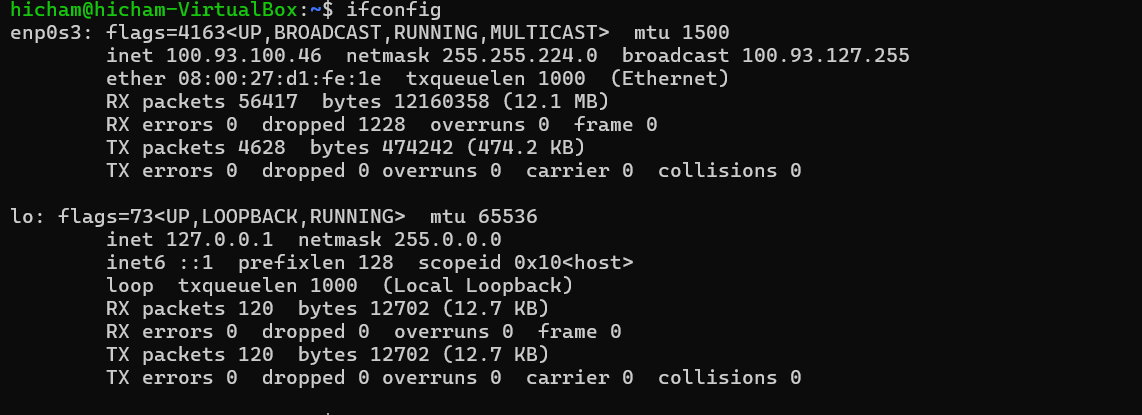


* + 1. Connexion des machines avec SSH :
       1. Remarques primitives :
* On peut initialiser Ubunto comme serveur et Windows comme client, ou inversement, Windows comme serveur et Ubunto comme client.
* Pour la machine virtuelle, on change le mode d'accès au réseau du mode NAT vers le mode Accès par Pont, ceci permet à la machine virtuelle d’avoir sa propre IP adresse et fonctionner comme machine indépendante en termes de réseau.



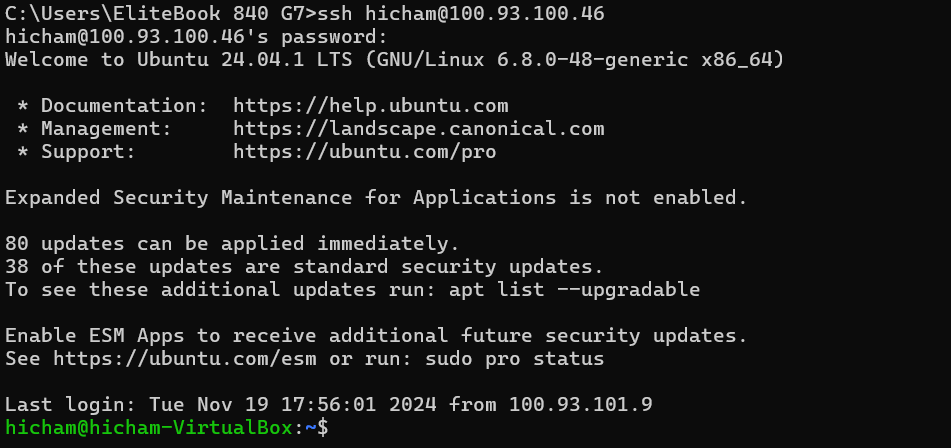
* Il est faut d'abord extraire les adresses IP des différentes machines :



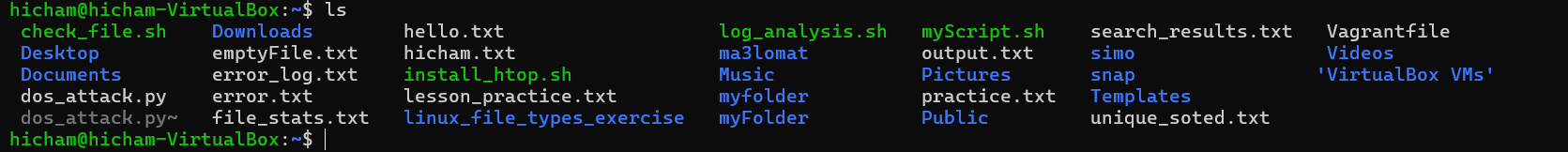


* + - 1. Connexion de client Windows avec le serveur UbuntoSSH : La connexion s’établit par la commande suivante :

#### ssh user@ip\_adress

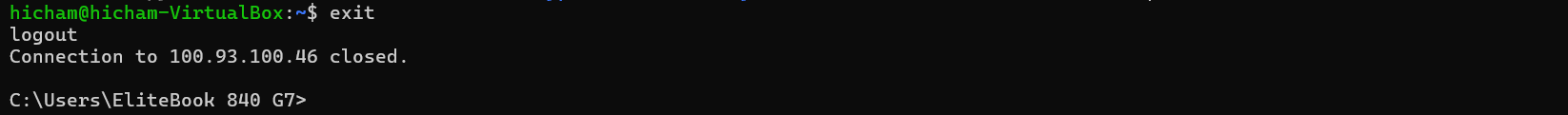
****

Maintenant, notre terminal Windows redirige et traite les commandes saisies dans le terminal de la machine virtuelle tant qu’elle est active, puis il visualise les résultats.

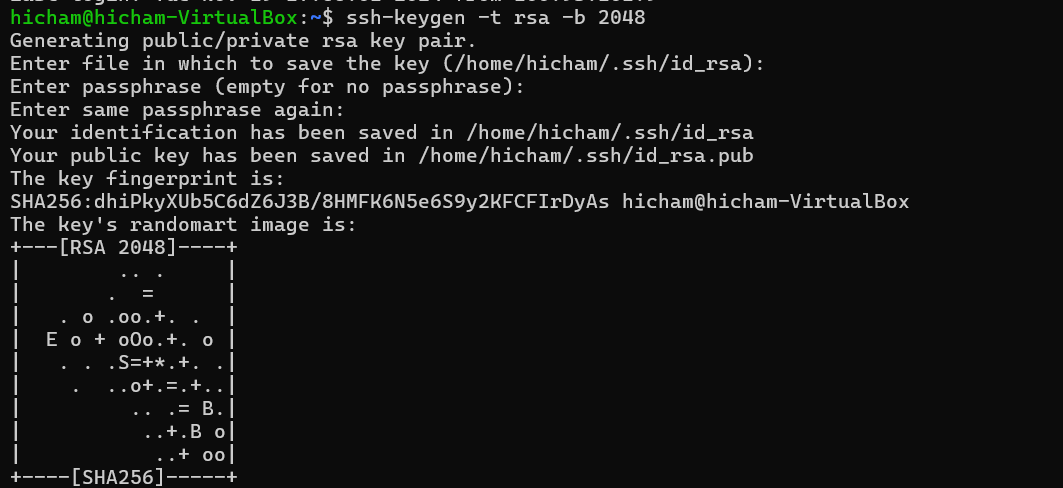


Pour quitter la terminale de la machine virtuelle on est invite à exécuter la commande suivante :

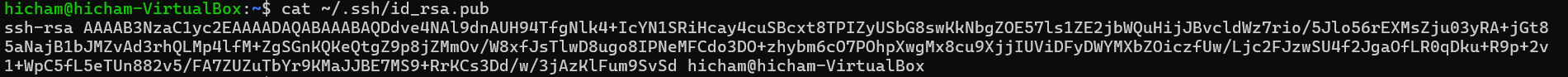
exit



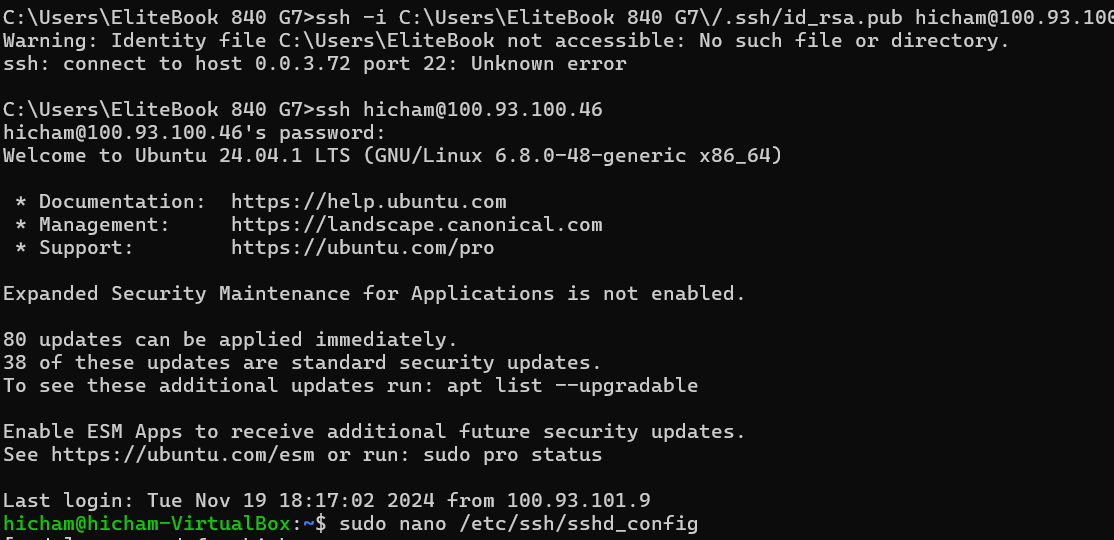
* + - 1. Connexion utilisant private/public key generation des public/private key

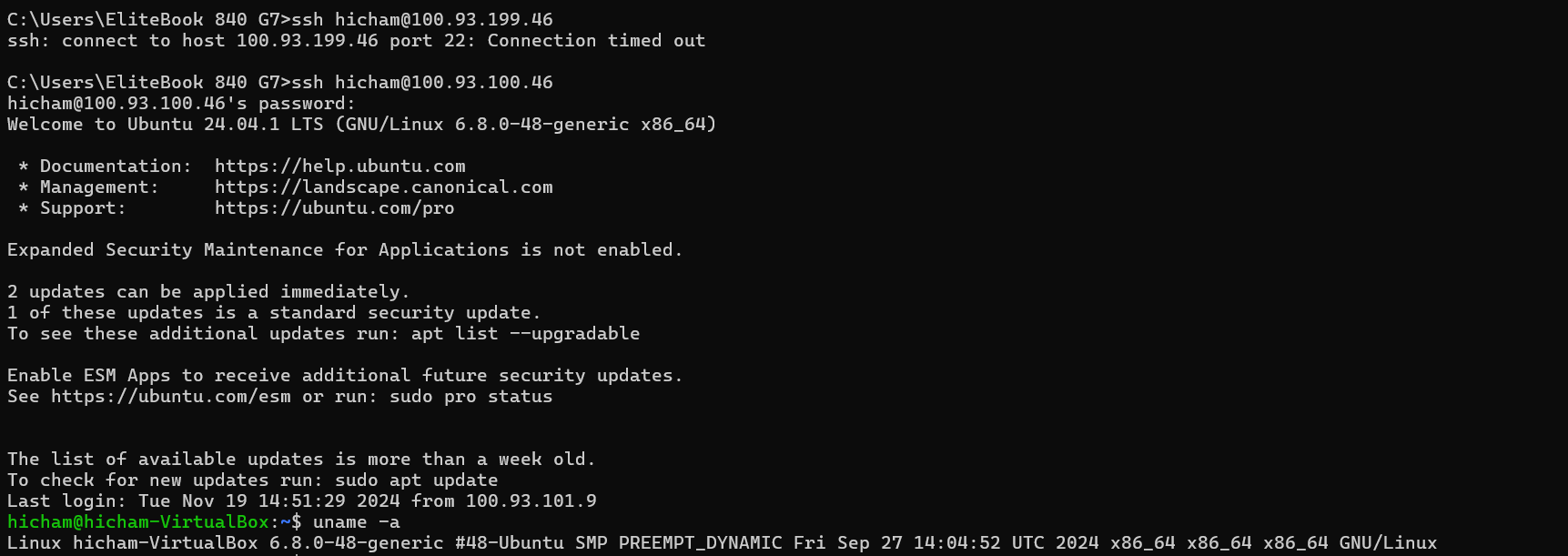


Copy the public key to the .ssh



Connection en utilisant keys :





### Conclusion :

La configuration de SSH, décrite dans ce rapport, illustre les étapes nécessaires pour établir des connexions sécurisées entre des machines distantes. Bien que le fichier sshd\_config n'ait pas été modifié dans ce cas, son importance reste centrale pour personnaliser et renforcer la sécurité du service SSH. Ce fichier permet de configurer des paramètres essentiels tels que :

* Le port d’écoute (par défaut 22), pour limiter les tentatives d’accès non autorisées.
* Les protocoles autorisés, garantissant l’utilisation des versions sécurisées de SSH.
* Les restrictions d’accès par utilisateur ou adresse IP, pour un contrôle précis des connexions.
* L’authentification par clé, offrant une méthode plus sécurisée que les mots de passe.

Adopter une configuration de base robuste et ajuster le fichier sshd\_config selon les besoins spécifiques permet de renforcer la fiabilité, la sécurité, et la performance du service SSH. Cela fait de SSH un outil indispensable pour une gestion à distance sécurisée et personnalisée dans des environnements réseau modernes.

## RDP :

Le Remote Desktop Protocol (RDP) offre une solution performante pour contrôler une machine à distance avec un environnement graphique fluide et sécurisé. Ce rapport explique le processus de configuration de RDP sur une machine Ubuntu afin de permettre un accès à distance depuis d'autres appareils, notamment un ordinateur sous Windows. L'installation repose sur le serveur RDP open-source **xrdp**, qui garantit une compatibilité avec Ubuntu et une interaction fluide avec l'environnement graphique GNOME.

Introduction :

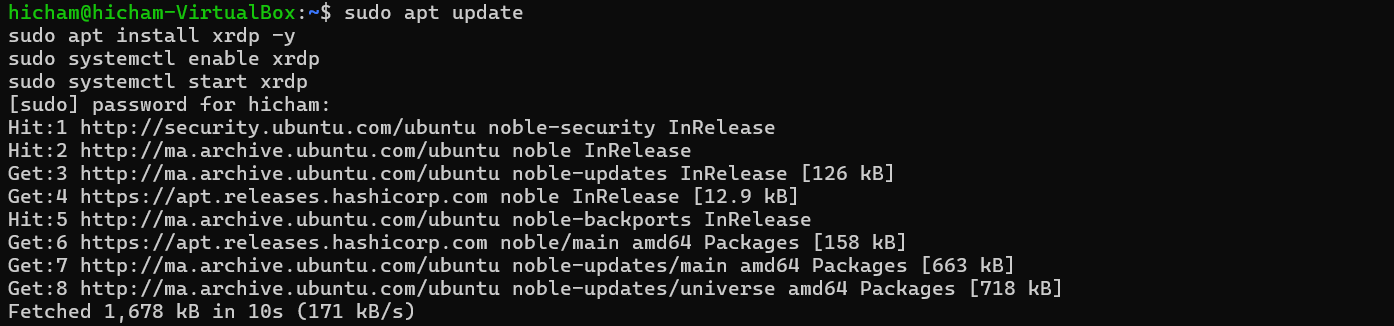
L'accès à distance à une machine Linux est souvent essentiel dans de nombreux contextes, tels que l'administration système, le travail collaboratif ou l'exécution d'applications nécessitant une interface graphique. Contrairement à SSH, limité à une interface en ligne de commande, **RDP** permet de contrôler un environnement graphique complet à distance, offrant une expérience utilisateur intuitive.

Fedora, par exemple, utilise par défaut l'environnement GNOME basé sur Wayland, ce qui peut poser des problèmes de compatibilité avec **xrdp**. Ce document propose des étapes détaillées pour contourner ces limitations et configurer correctement RDP sur Fedora, ou des distributions similaires, comme Ubuntu.

Étapes détaillées pour configurer RDP :

* 1. Installation de xrdp et démarrage du service :

Le serveur **xrdp** est l'élément central pour permettre une connexion RDP. Il écoute les requêtes sur le port par défaut **3389** et offre un accès à l’environnement graphique GNOME. L’installation est réalisée à l’aide des commandes suivantes :

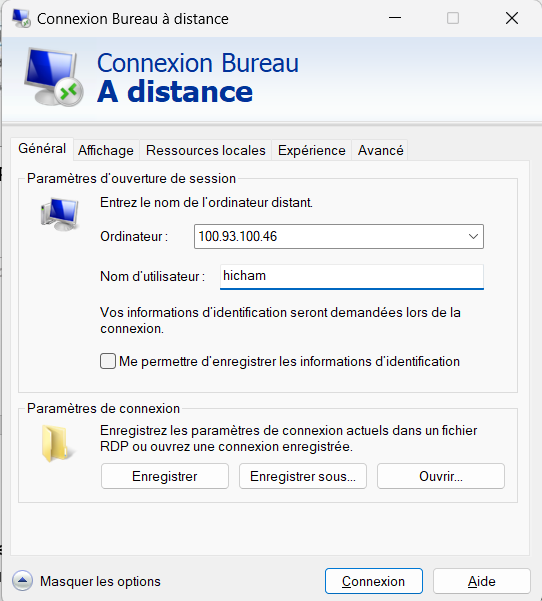


* 1. Configuration du pare-feu :

1. Pour permettre les connexions RDP, il est indispensable d’ouvrir le port **3389** dans le pare-feu de Fedora. Sans cette configuration, les tentatives de connexion depuis des clients distants seront bloquées. Les commandes suivantes permettent d’ajouter une règle permanente pour autoriser le trafic RDP et d’appliquer les modifications immédiatement :
2. Cette configuration garantit que le service **xrdp** est accessible aux autres appareils du réseau, une étape essentielle pour établir des connexions RDP sécurisées et fonctionnelles.
   1. Résolution des problèmes liés à SELinux :
3. Dans un environnement **Ubuntu** où **SELinux** est activé, certaines fonctionnalités de **xrdp** peuvent être bloquées pour des raisons de sécurité. Afin d'éviter ces restrictions, des permissions spécifiques ont été configurées à l'aide des commandes suivantes :
4. Ces ajustements garantissent que **SELinux** autorise les connexions via le port utilisé par **xrdp**, assurant ainsi le bon fonctionnement du service.

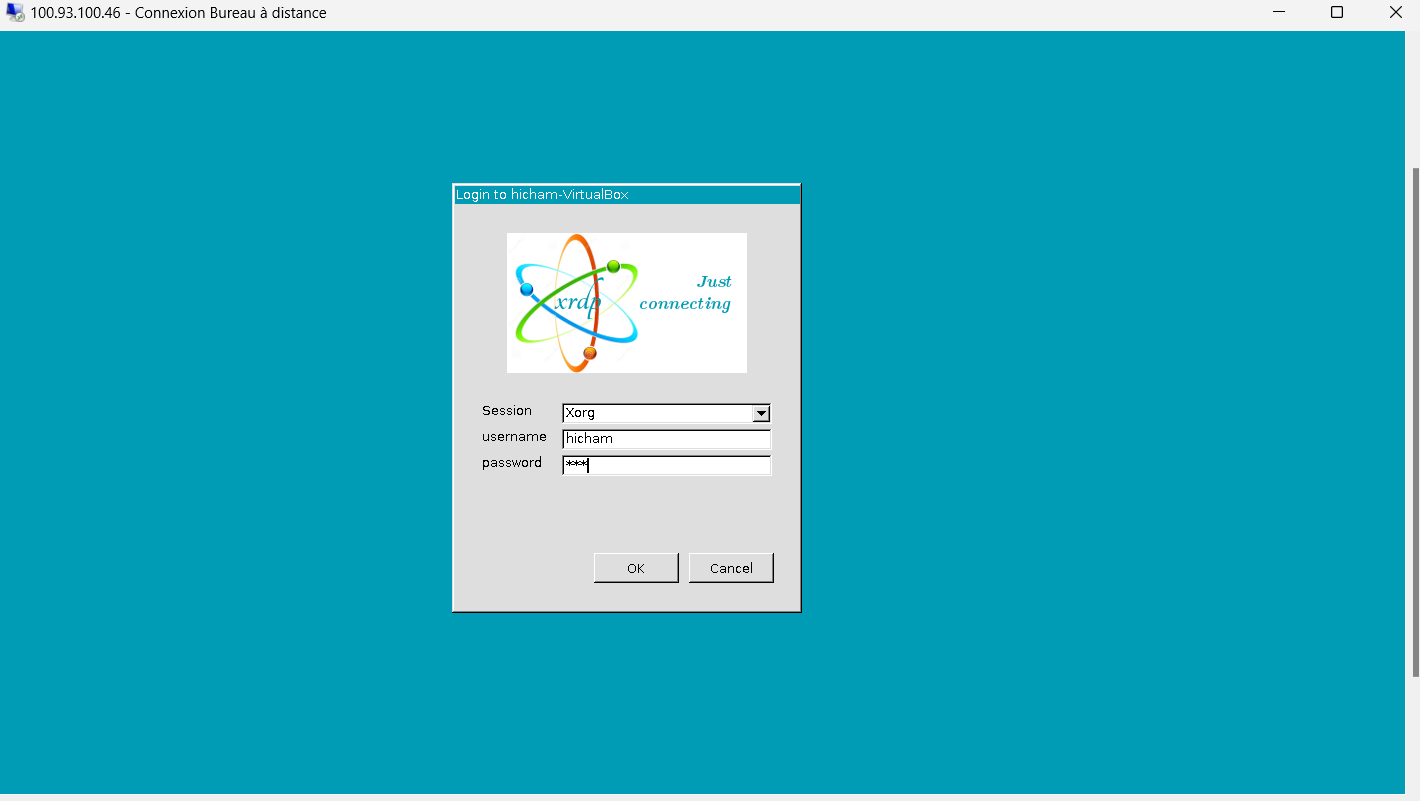
### Connexion depuis un client RDP sous Windows :

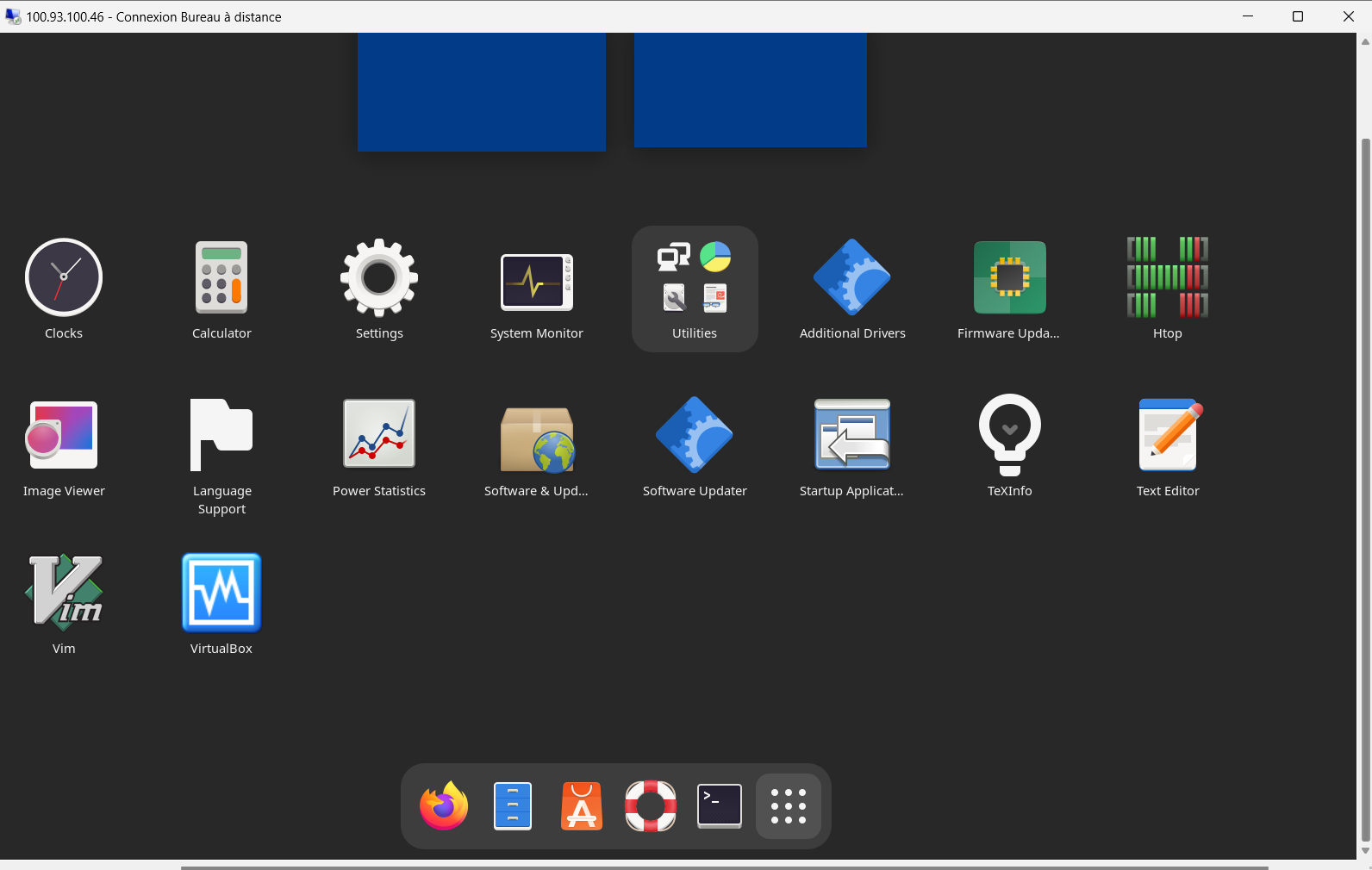
Après avoir finalisé la configuration du serveur RDP sur **Fedora**, la connexion a été testée en utilisant le client RDP intégré à **Windows**. Pour se connecter à un bureau à distance, il suffit d’accéder à la fenêtre **Connexion Bureau à distance** disponible dans le menu **Démarrer** de Windows, puis de saisir les informations nécessaires.



L’adresse IP de mon machine Ubunto et le nom d’utilisateur est fatimaelhaddouchi.

Un avertissement concernant le certificat de sécurité est apparu, car xrdp utilise un certificat auto-signé par défaut. Cet avertissement a été ignoré pour établir la connexion. La session distante a été ouverte avec succès, offrant un accès complet à l’environneme de bureau GNOME.





N.B. : Il est conseillé de se déconnecter du compte dans la machine virtuelle avant de se connecter.

### Problèmes rencontrés :

 **Avertissement lié au certificat auto-signé** : L'utilisation d'un certificat auto-signé avec xrdp entraîne un avertissement lors de la connexion. Bien que ce ne soit pas un problème majeur pour un réseau privé, il est recommandé de remplacer ce certificat par un certificat émis par une autorité de certification pour renforcer la sécurité des connexions.

 **Compatibilité avec Wayland** : GNOME utilise Wayland par défaut, ce qui a nécessité de basculer manuellement vers Xorg pour garantir le bon fonctionnement de xrdp.

 **Édition Familiale de Windows 10** : Bien que l’utilisation du client RDP pour accéder à Fedora ait fonctionné sans problème, l’édition Familiale de Windows 10 ne permet pas de partager la machine Windows elle-même via RDP, limitant ainsi certaines fonctionnalités de bureau à distance.



### Conclusion :

La mise en place de RDP sur Ubuntu via le serveur xrdp a été réussie, offrant ainsi une solution fiable pour l'accès à distance à un environnement graphique Fedora. Bien que certaines étapes aient nécessité des ajustements, tels que la configuration du serveur Xorg et l'acceptation d'un certificat auto-signé, la solution a démontré son efficacité pour un usage à distance. Toutefois, pour un usage professionnel, il est fortement recommandé de renforcer la sécurité des connexions RDP en adoptant un certificat SSL valide et en restreignant l'accès à partir du réseau local. Cette approche contribuera à améliorer la confidentialité et la sécurité des échanges à distance