
RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

**UNIVERSITÉ D'ABOMEY-CALAVI**

**INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE**

BP 526 Cotonou Tel : +229 21 14 19 88
http://www.ifri-uac.net Courriel : contact@ifri.uac.bj


# MÉMOIRE

pour l'obtention du

**Diplôme de Licence en Informatique**

**Option :** Sécurité Informatique

**Présenté par** :
Jospy **GOUDALO**

# Proposition d'un système de paiement électronique des factures d'électricité de la SBEE.

**Sous la supervision** :
**Professeur Eugène C. EZIN**
Maître de Conférences en Informatique
Université d'Abomey-Calavi

Année Académique : **2017-2018**

# Table des matières

# Table des figures



# Liste des tableaux

# Dédicace

A

mon Père **Noll GOUDALO**

ma Mère **Nadine SODEDJI**

et mon Frère **Brice GOUDALO**

# Remerciements

Nous exprimons notre vive gratitude aux personnes physiques et institutions qui ont contribué à rendre meilleur notre travail, notamment :

- M. Eugène C. EZIN, notre Maître de mémoire et Directeur de l'Institut de Formation et de Recherche en Informatique (IFRI) ;
- M. Lionel METONGNON, doctorant en sécurité informatique et enseignant à l'IFRI ;
- M. Armand, ... et enseignant à l'IFRI ;
- Tous les enseignants de l'IFRI.

Que tous ceux qui nous ont aidé, de près ou de loin, trouvent ici l'expression de nos sentiments les meilleurs.

# Résumé

Resume en francais

**Mots clés** : sécurité ,paiement, factures, sbee,

# Abstract

Resume en anglais

**Key words**: ....,....,....

# Introduction

L'informatique, science du traitement rationnel et automatisé de l'information, est devenue aujourd'hui un outil indispensable pour l'évolution de toute nation, de toute société. Elle est passée d'un statut luxueux à un statut nécessaire. La preuve: toute structure de renom investit un budget non négligeable pour l'informatisation de ses opértions. Cependant, certains ont très vite profité de sa puissance pour piéger les autres, ignorant le fonctionnement de l'ordinateur dans le but de voler ou altérer des informations précieuses et confidentielles pour des fins diverses: c'est la cybercriminalité. Vu l'ampleur que prend ces differents crimes, il devient necessaire de voir autrement l'importance de la sécurité dans tous les systemes d'informations.[[ok]]

## Contexte et justification

---

La sécurité de l'information revêtant une importance capitale pour la survie d'un peuple, il a fallu mettre en oeuvre un système de défense aux menaces qui surgissent pour réduire l'impact de ces attaques et essayer, au maximum, d'éliminer les risques. Parmi ces différentes attaques, celle opérée par webcam est d'une sévérité élevée, car elle permet de voir en direct l'utilisateur sans qu'il le sache. Le caractère critique de cette dernière devient plus grave quand l'utilisateur se retrouve nu devant son ordinateur. Alors qu'il se croit être seul, il est exposé à d'autres regards. De cette attaque, plusieurs autres telles que l'intrusion dans les systèmes sont opérées. Cette intrusion permet de se comporter comme propriétaire du système victime. Il faut donc répondre de manière éfficace afin de freiner l'ampleur de ces menaces. Notre thème de mémoire "Système de détection des attaques par webcam et de prévention d'intrusion sous Linux" s'inscrit donc dans cette dynamique. Il s'agit de réaliser un système de prévention et de détection de certaines de ces attaques.

## Problématique

---

Les réseaux informatiques constituent des ressources indispensables pour les utilisateurs et un vaste champ de cibles potentielles pour les pirates informatiques. Cependant, nous sommes dans un monde où beaucoup de gens utilisent l'informatique sans se soucier des risques auxquels ils sont confrontés, surtout lorsque leurs ordinateurs ne sont soumis à aucune sécurité. Il est facile aujourd'hui de prendre possession d'un ordinateur non protégé dans un réseau informatique, de prendre possession de ses ressources telles que la webcam et l'audio pour ne citer que ces éléments-là.
Une des croyances les plus dangereuses est celle qui fait penser à une sécurité totale sur les systèmes Linux. De ces constats, nous pouvons dire que la sécurité des systèmes informatiques n'est pas maîtrisée pendant que les attaques sur les réseaux prennent de l'ampleur chaque jour. Les préoccupations abordées dans ce travail sont de deux ordres: la première est relative à la détection de l'espionnage par la webcam et, la deuxième à la prévention des intrusions dans les systèmes. Les deux sont dédiées aux plate-formes Linux.

Il n'est pas rare de constater qu'après de longues heures d'attente au guichet de la SBEE, l'on vous dise: "Désolé monsieur nous avons un problème de connexion. Veuillez repasser demain.". Il faudra donc repasser plus tard pour solder la facture alors que nous n'avons pas forcément assez de temps pour cela.
Nous sommes aussi parfois confrontrés à l'oubli des factures non payées. Cependant quand vous n'etes pas à jour après un délai d'environ un mois, un agent peut passer à tout moment couper le courant et vous devez payer des pénalités.
Notre travail consistera donc à mettre en place une application web sécurisée permettant le paiement des factures depuis un telephone portable ou un ordinateur connecté à Internet.

## Objectifs

Notre projet à pour objectif principal de détecter les tentatives d'attaques par webcam sur les systèmes Linux et de mettre en place un mécanisme pour réduire les risques d'intrusion illicite dans les réseaux.

Plus précisément, il s'agira :

- d'alerter l'utilisateur de l'usage de sa webcam;
- de lui proposer de couper le fonctionnement de la webcam ou de la laisser ouverte quand il est l'utilisateur légal;
- de configurer le pare-feu par défaut de Linux afin de prévenir toute connexion indésirée en local et sur Internet;
- de protéger l'ordinateur des dénis de service;
- de limiter le nombre de connexions maximal par adresse IP.

## Environnement de stage

Ce travail a été réalisé dans les locaux de la Direction des Systemes Informatiques de la Société Béninoise d'Energie Electrique (SBEE) - Direction Générale.

## Organisation du mémoire

Le présent travail se présente en trois (03) chapitres. Le premier concerne la revue de littérature sur le systeme existant (actuel) permettant le paiement des factures et .... . Le deuxième chapitre aborde les choix techniques opérés en vue de la conception et de la réalisation des solutions proposées. Le troisième chapitre, quant à lui, fait une analyse critique des résultats issus de nos tests après les avoir exposés.

Chapitre 1

# Revue de littérature

## Introduction

Pour bien réaliser un projet, un état des lieux permettant de faire un point sur le sujet du projet est requis. Ainsi, ce chapitre présente un état des lieux concernant les intrusions informatiques. Dans un premier temps, nous aborderons les généralités sur les intrusions informatiques, puis nous présenterons quelques solutions existantes.

## Généralités sur les intrusions informatiques

D'après LE PETIT LAROUSSE ILLUSTRE 2010, l'intrusion est définie comme «l'action de s'introduire sans y être invité dans un lieu, une société, un groupe, un système informatique. C'est aussi l'action d'intervenir dans un domaine où l'on n'a aucun titre à le faire». Cette définition de l'intrusion s'applique aussi en informatique [1] [**?**]. En effet, l'intrusion en informatique est définie comme étant toute utilisation d'un système informatique à des fins autres que celles prévues, généralement après acquisition de privilèges de façon illégitime.

L'arrivée d'Internet a apporté une grande révolution dans le monde mais aussi a entraîné une kyrielle de problèmes dans le domaine de la protection de la vie privée. Ainsi des personnes appelées hackers arrivent à prendre possesion de tout un système d'information et à le paralyser.

### Notion de hacker et de cracker

Un hacker est une personne qui, par jeu, goût du défi ou souci de notoriété, cherche à contourner les protections d'un logiciel, à s'introduire frauduleusement dans un système ou un réseau informatique. [**?**] [2].
Un cracker, quant à lui, s'introduit tout aussi frauduleusement dans un système informatique pour en entraver ou en fausser le fonctionnement. Son action est souvent plus dévastatrice.

### Mode opératoire d'une intrusion informatique

Un test d'intrusion peut être décomposé en une suite d'étapes ou phases. Lorsqu'elles sont réunies, ces étapes forment une méthodologie complète pour mener à bien un test d'intrusion. L'établissement d'une méthodologie permet de décomposer une procédure complexe en une suite de tâches gérables de taille plus

[1] Le petit Larousse Illustré 2010.
[2] Recommandation officielle : fouineur.

réduite. Ainsi nous regroupons cette méthodologie en quatre étapes qui sont **la reconnaissance**, **les scans**, **l'exploitation**, **la postexploitation et le maintien d'accès** [3][**?**].

### La reconnaissance

La reconnaissance, ou recueil d'informations, est probablement la plus importante des quatre phases. Plus le hacker passe du temps à collecter des informations sur sa cible, plus les phases suivantes auront une chance de réussir[**?**]. En effet la reconnaissance permet de connaitre la cible dans les détails, de connaître les points forts et surtout les points faibles afin de notifier les prochaines possibilités d'attaque.

### Les scans

Les scans sont des procédés ayant pour objectif d'identifier les systèmes actifs et les services qui existent sur les systèmes scannés. Dans ce cadre, le hacker prend le soin de vérifier l'activité d'un système, de trouver les portes ouvertes (les ports), de vérifier les processus tournant sur le système et d'aller à la recherche des vulnérabilités. Ce stade requiert une compréhension plus avancée des systèmes informatiques pour mieux comprendre les résultats recueillis [4][**?**].

### L'exploitation

En termes simples, l'exploitation consiste à obtenir un contrôle sur un système. Toutefois, il est à notifier que tout exploit ne conduit pas à la compromission intégrale d'un système. Un hacker peut se servir donc d'un exploit pour télécharger des contenus dont il ne détient pas la propriété pendant qu'un autre utilise un exploit pour crypter les fichiers du système. L'utilisation de l'un des exploits [5] dépend donc de l'objectif visé par le hacker [**?**].

### Post exploitation et maintien d'accès

Cette étape consiste à couvrir les traces de l'intrus agissant afin de ne pas se faire repérer [**?**]. Il permet aussi à ce dernier de facilliter ses prochains accès à la machine victime de ses attaques par l'installation de portes dérobées communément appelées "Backdoor". Ainsi, il n'aura plus besoin de reprendre toutes les étapes de son processus pour accéder à la machine dont il prend le contrôle.

FIGURE 1.1 – Méthodlogie ZEH, Patrcick Engebretson, "Les bases du hacking", PEARSON 2013

La méthodologie d'attaque étant cernée, nous allons maintenant présenter les différentes sortes d'attaque auxquelles sont confontrés les systèmes informatiques.

## Quelques exemples d'attaques informatiques

Parmi les attaques les plus connues, on peut citer les techniques de déni de service (DoS), l'usurpation d'adresse IP, l'usurpation du DNS *Domain Name Server*, l'ingénierie sociale.

[3] La post exploitation et le maintien d'accès forment une étape.
[4] Informations recueillis au cours du scan
[5] Un exploit est le moyen par lequel un attaquant, ou un pentester en l'occurrence, profite d'un défaut dans un système, une application ou un service.

### Le deni de service

D'une manière générale, on parle de déni de service quand une personne ou une organisation est privée d'un service utilisant des ressources qu'elle est en droit d'avoir en temps normal [**?**]. On trouvera par exemple des dénis de service touchant le service de courrier électronique, d'accès à Internet, de ressources partagées (pages Web), ou tout autre service à caractère commercial comme Yahoo ou EBay.

Quoiqu'il en soit, le déni de service est un type d'attaque qui coûte généralement très cher puisqu'il interrompt le cours normal des transactions pour une entreprise ; les sommes et les enjeux sont énormes et cela ne peut aller qu'en s'aggravant tant que des parades réellement efficaces n'auront pas été trouvées.

Il existe plusieurs moyens pour parvenir à un déni de service. Nous ne parlerons que de trois types dont l'attaque avec les paquets **XMAS**, l'attaque **Smurf**, et l'attaque par **Syn flooding**.

**L'attaque avec paquets XMAS.** Un paquet XMAS ou Christmas Tree est un paquet dans lequel les drapeaux *(flag)* de tout protocole sont définis. Les bits FIN, URG et PSH dans l'en-tête TCP de ce type de paquet sont définis. Ce paquet s'appelle paquet Christmas Tree, car tous les champs d'en-tête sont éclairés c'est à dire activés comme un arbre de noël. Ce type de paquet nécessite beaucoup plus de traitements que les paquets habituels, de sorte que le serveur alloue un grand nombre de ressources pour ce paquet. Par conséquent, cela peut être utilisé pour effectuer une attaque DoS sur le serveur.

**L'attaque Smurf.** L'attaquant ici envoie un grand nombre de paquets de diffusion d'écho ICMP, avec des adresses IP source falsifiées par rapport à l'adresse IP de la cible. Toutes les machines du réseau reçoivent ce message de diffusion et répondent à la cible avec un paquet de réponse d'écho.

**Le syn flooding.** Lors de l'initialisation d'une connexion TCP entre un client et un serveur, un échange de messages a lieu. Le principe est celui du three-way handshake, qui dans le cas d'une connexion normale sans volonté de nuire, se déroule en trois étapes :

1. le client demande une connexion en envoyant un message SYN (pour synchronize) au serveur ;
2. le serveur accepte en envoyant un message SYN-ACK (synchronize-acknowledgment) vers le client ;
3. le client répond à son tour avec un message ACK (acknowledgment) ; la connexion est alors établie.

FIGURE 1.2 – 3 ways Handshake

L'attaquant peut créer un grand nombre de requêtes SYN forgées qui ont leurs adresses IP source falsifiées et l'envoyer à la cible. La cible répond avec SYN-ACK et alloue ses ressources pour la connexion, mais ne reçoit jamais la réponse ACK. Les ressources de la machine cible sont épuisées et elles ne permettent pas de répondre à d'autres demandes de machines légitimes.

### L'usurpation d'adresse IP

L'« usurpation d'adresse IP » (également appelée mystification ou en anglais *IP Spoofing*) est une technique consistant à remplacer l'adresse IP de l'expéditeur d'un paquet IP par l'adresse IP d'une autre machine. Cette technique permet ainsi à un pirate d'envoyer des paquets de façon anonyme. Il ne s'agit pas pour autant d'un changement d'adresse IP, mais d'une mascarade de l'adresse IP au niveau des paquets émis [**?**].

### L'usurpation DNS

Il s'agit de corrompre le cache du serveur DNS *Domain Name Server* afin de faire croire à une machine qu'un nom de machine est relatif à une fausse adresse donnée. Ainsi donc les communications vers la machine ciblée sont redirigées vers celle dont l'adresse est portée en avant.

### Ingénierie sociale

L'ingénierie sociale est une attaque qui s'appuie essentiellement sur les relations humaines pour inciter de façon détournée à enfreindre les procédures de sécurité [**?**]. L'ingénierie sociale fait partie des techniques les plus simples, mais aussi les plus efficaces pour entrer dans un système sécurisé à grande échelle. En effet le système aura beau être sécurisé avec les moyens les plus sophistiqués, une seule information divulguée d'un utilisateur aura suffi pour mettre le système à terre. Il s'agit d'une arme très forte résultant de l'étude de l'être humain, de ses passions, de ses désirs. Cette forme d'attaque est très dévastatrice en fonction de l'importance des informations tirées.
Face à ces différentes menaces, différentes solutions sont nées. Nous allons en présenter quelques-unes.

## Les applications basées sur la détection et la prévention d'intrusion

Plusieurs systèmes ont vu le jour en raison des attaques qui sévissent de jour en jour. Il s'agit des IDS (*Intrusion Detection System*), des IPS (*Intrusion Prevention System*), des firewall (Pare-feu), des antivirus pour ne citer que ceux là.

### Les systèmes de détection d'intrusion

Un système de détection d'intrusion *Intrusion Detection System (IDS) en anglais* est comparable à une alarme domestique contre les voleurs. Si une tentative de pénétration non avenue est découverte, une alerte sera déclenchée. La fiabilité d'un tel système dépend de la puissance des capteurs à détécter les événements pour lequels ils ont été mis en place avec une marge d'erreur minimisée. Certains termes sont souvent employés quand on parle d'IDS et il sied de les comprendre. Il s'agit notamment du faux positif et du faux négatif.
Un faux positif est une alerte provenant d'un IDS mais qui ne correspond pas à une attaque réelle, tandis qu'un faux négatif est une intrusion réelle qui n'a pas été détectée par l'IDS.

Un systéme de détection d'intrusion se base sur trois éléments particuliers :

- la sonde qui collecte les informations sur le système ;
- l'analyseur qui traite ces informations afin de détecter les possibilités d'attaque ;
- la réponse qui est la solution apportée à la menace détectée.

**Quelques exemples d'IDS**

- **Snort**
  Pour effectuer ces analyses, Snort se fonde sur des règles [**?**]. Snort est fourni avec certaines règles de base mais cependant, comme tout logiciel, Snort n'est pas infaillible et demande donc une mise à jour régulière. Snort peut également être utilisé avec d'autres projets open sources tels que SnortSnarf, ACID, sguil et BASE (qui utilise ACID) afin de fournir une représentation visuelle des données concernant les éventuelles intrusions.

- **Bro**
  Bro s'appuie sur les mêmes bases théoriques que Snort (filtrage par motif, formatage aux normes RFC, etc), mais il intègre un atout majeur : l'analyse de flux réseau. Cette analyse permet de concevoir une cartographie du réseau et d'en générer un modèle. Ce modèle est comparé en temps réel au flux de données et toute déviance lève une alerte [**?**].

- **Fail2ban**
  Fail2ban bloque les adresses IP appartenant à des hôtes qui tentent de casser la sécurité du système. Il lit les logs de divers services (SSH, Apache, FTP…) à la recherche d'erreurs d'authentification répétées et ajoute une règle iptables pour bannir l'adresse IP de la source [**?**].

### Les systèmes de prévention d'intrusion

Le besoin de ne pas attendre les attaques avant de réagir s'est ressenti, compte tenu des inconvénients générés par ces attaques. C'est ainsi que sont nés les IPS *Intrusion Prevention System*. Ces derniers utilisent des techniques qui permettent de surveiller les processus d'un système et de tuer ceux dont le comportement paraît douteux tel que les processus qui tentent d'exécuter un dépassement de tampon. Cependant, quelques risques sont liés à leur utilisation. Il s'agit en particulier de l'arrêt des processus mis en activité par l'utilisateur, ce qui devient une restriction anormale.

**Quelques exemples d'IPS [?]**

- **Suricata**
  Suricata est un logiciel open source de prévention d'intrusion et de supervision de sécurité réseau. Il est développé par la fondation OISF *Open Information Security Foundation* [**?**].
- **Snort Inline**
  L'IPS Snort Inline est une version modifiée de l'IDS Snort pour en faire un IPS, une solution capable de bloquer les intrusions ou attaques réseau. Il reçoit les paquets envoyés par le firewall Netfilter avec l'aide de la librairie libipq, les compare avec des règles de signature Snort et les marque en drop s'ils correspondent à une règle, puis finalement les renvoie vers Netfilter où les paquets Snort Inline marqués sont rejetés [**?**].
- **HLBR**
  HLBR est un système de prévention des intrusions. Sa principale caractéristique est qu'il peut s'exécuter directement sur la couche du modèle OSI, ce qui signifie qu'il n'a même pas besoin d'une pile TCP/IP fonctionnant comme un pont [**?**].

### Classification des IDS selon leur emplacement

On distingue deux types d'IDS : les HIDS *Host Intrusion Detection System* et les NIDS *Node Intrusion Detection System*. Les premiers sont remarquables par leur emplacement sur les machines qu'elle protège. Les seconds se ramarquent par leur position en des points stratégiques du réseau permettant ainsi de surveiller les activités du réseau et d'apporter une protection à l'ensemble du réseau.

### Les pare-feux

Les pare-feux sont des systèmes dont le but principal est de limiter le trafic sur un réseau informatique. Ainsi donc, l'administrateur définit les types de trafic voulus et ceux indésirés. A la rencontre d'un trafic non désiré, les paquets relatifs sont rejetés.

### Les antivirus

Les antivirus sont des programmes dont le rôle est de vérifier que l'identité des arrivants ou des programmes s'exécutant n'existent pas dans une liste noire prédéfinie dans une base de données. On parle de signature. Ainsi donc, la puissance d'un antivirus est basée sur la pertinence des informations contenues dans la base de données qu'il faut mettre à jour régulièrement.

## Conclusion

Les attaques informatiques sont nombreuses et de différentes formes. Dans ce chapitre, nous avons présenté le fonctionnement de quelques unes d'entre elles et exposer certaines solutions existantes. Dans le chapitre suivant, nous aborderons notre solution à travers sa conception et les outils utilisés pour sa réalisation.

Chapitre 2

# Conception et Matériels

## Introduction

La programmation d'une application requiert d'abord l'organisation et la documentation de ses idées. La définition des modules induit les différentes étapes de sa réalisation. C'est cette démarche antérieure à l'écriture que l'on appelle modélisation. Ainsi, notre modélisation est axée autour de deux diagrammes : le diagramme des cas d'utilisation recense les différentes fonctionnalités de notre système ; le diagramme de séquence illustre le fonctionnement interne du système dans le temps.

## Conception

### Méthode de modélisation

Afin de modéliser les fonctionnalités de notre solution, nous avons choisi le langage UML *Unified Modeling Language* [**?**]. Issu d'un large consensus, le langage UML garantit la stabilité et la performance d'un projet grâce à son caractère formel et industrialisé. Aussi facilite-t-il la compréhension du système par l'usage de représentations graphiques appelées diagrammes. Ces diagrammes nous ont permis de modéliser notre solution en utilisant les diagrammes de cas d'utilisation et de séquence.

### Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système. C'est le premier diagramme du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre.

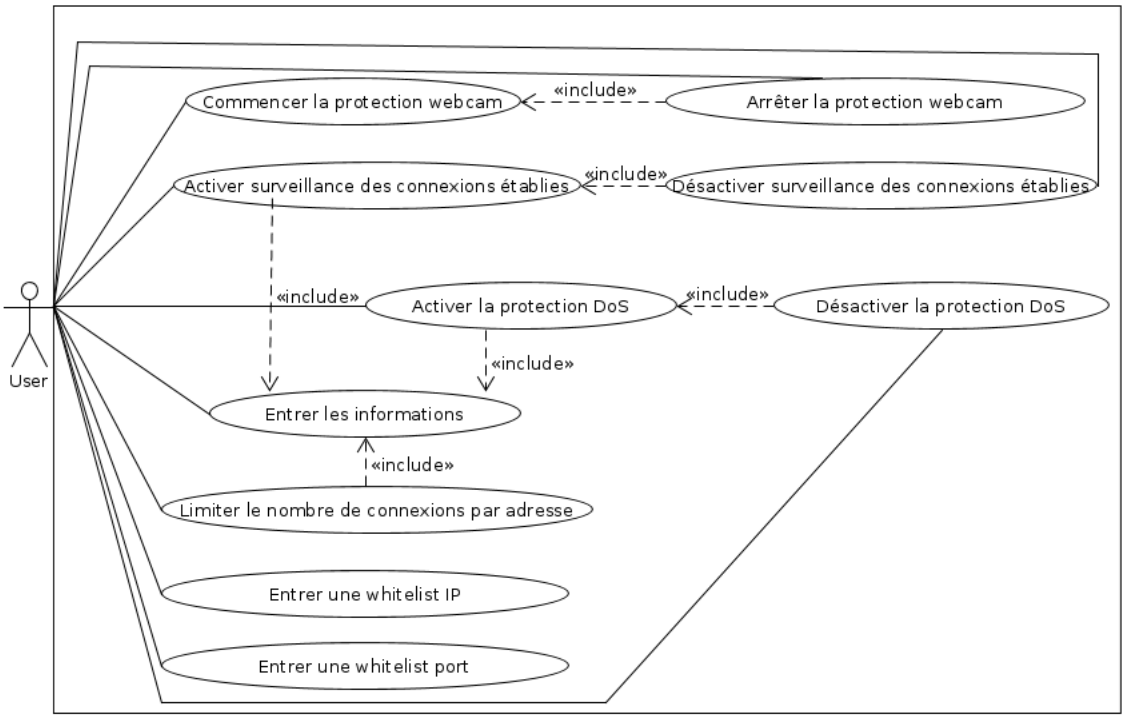


FIGURE 2.1 – Diagramme de cas d'utilisation

Les informations à entrer désignent les identifiants que l'utilisateur doit entrer au début du programme. Ces derniers sont, en particulier, son nom, son email, son numéro de téléphone, son nom d'utilisateur **root** et le mot de passe associé. Le système se servira de cela pour lui envoyer des alertes et effectuer des tâches réservées à un utilisateur root.

### Diagramme de séquence

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Ce mode de représentation effectue la description du fonctionnement dynamique du système. En d'autres termes, il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre.

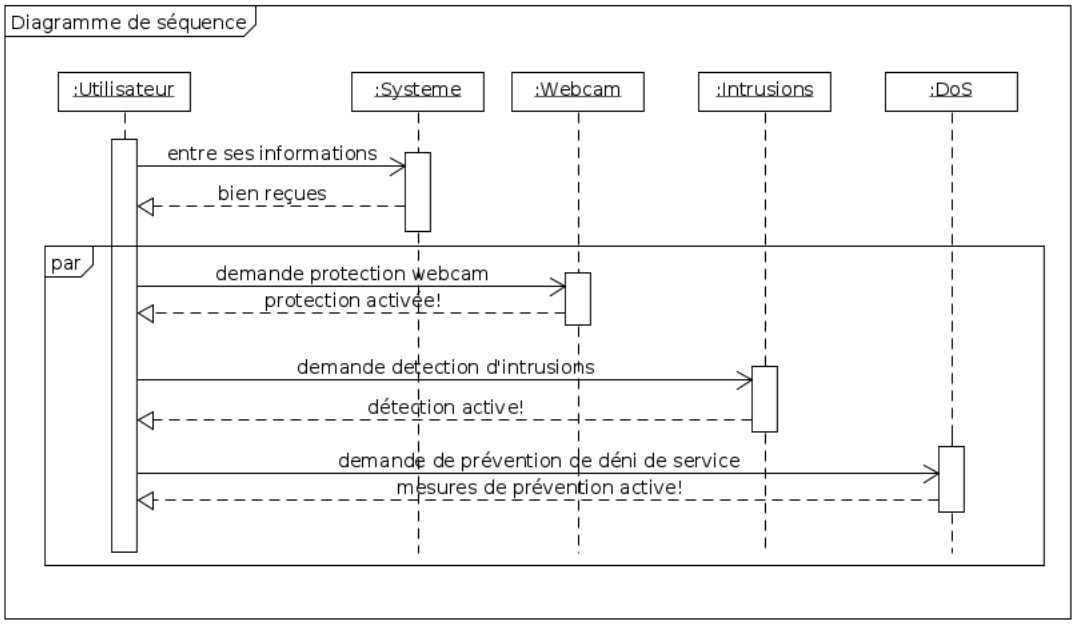


FIGURE 2.2 – Diagramme de séquence

Après que l'utilisateur ait entré ses informations, identifiants pour être administrateur de son système, il peut parallèlement activer la protection webcam, lancer la detection d'intrusion ou prévenir les dénis de service.

# Matériels

---

## Principe de fonctionnement de notre solution

Nos objectifs pour ce travail sont, dans un premier temps, d'arriver à détecter l'espionnage par webcam et dans un second temps de proposer des options à activer pour protéger l'ordinateur hôte de quelques intrusions qui pourraient survenir. La présentation des principes de fonctionnement se fera en deux étapes : le principe de détection d'attaques par webcam et le principe de protection contre les intrusions.

### La webcam

Pour détecter l'utlisation illicite d'une ressource, il revient de monitorer en temps réel les flux provenant de cette ressource et générés par des processus. Ainsi, notre solution permet de surveiller les flux de la webcam à intervalles de temps réguliers de trois secondes pour ne pas utiliser, de façon incontrolée, les ressources de l'ordinateur.
A la découverte d'un processus utilisant la webcam, une alerte est envoyée à l'utilisateur qui choisit si oui ou non il est à la base de l'activité d'un tel processus. Si sa réponse est négative, alors il est victime d'un espionnage et, à ce moment un filtre est fait pour connaitre l'identifiant du processus qui est immédiatement tué. En cas de réponse positive de l'utilisateur (oui c'est bien moi), alors le processus en cours est laissé et la detection de processus utilisant la webcam est désactivée pendant cinq minutes. L'algorithme 1 présente le processus de

détection d'espionnage par webcam.

**Algorithme 1 :** Algorithme de détection d'espionnage par webcam

```
for chaqueTroisSecondes do
    etatWebcam ← enUtilisation()
    if etatWebcam = ouvert then
        action ← etesVousAuteur()
        if action = non then
            chercherIdProcessus()
            arreterProcessus()
        else
            cinqMinutesDePause()
        end
    end
end
return action
```

### Les intrusions

Quand un pirate arrive à prendre possession d'une machine dans un réseau, c'est qu'il est arrivé à établir une connexion entre sa machine et la victime. Ainsi, dans la liste des connexions établies (ESTABLISHED), on peut donc lister la connexion permettant au pirate d'interagir avec sa victime à son insu. Notre solution pour ces attaques est de proposer une option à activer et dont le but est d'alerter l'utilisateur dès qu'une connexion est établie. A cet effet, une vérification des connexions ouvertes se fait toutes les trois secondes. Si l'utilisateur ne reconnait pas cette connexion comme étant légale, alors il le signale par un bouton qu'il appuie et à ce moment, l'adresse source ainsi que le port source de connexion sont mis dans une liste noire. L'algorithme 2 présente les étapes de détection d'intrusions.

**Algorithme 2 :** Algorithme de détection d'intrusions

```
for chaqueTroisSecondes do
    list ← listeConnexionsEtablies()
    for connexion as list do
        alerterConnexionEtablie()
        action ← autoriser()
        if action = non then
            blacklistAdresseEtPort()
        end
    end
end
return action
```

Lorsque l'attaque est faite au cours des trois secondes, l'application ne le détecte pas. C'est une de ses faiblesses.

### Les whitelist

Il y a des connexions importantes que la machine établit en local avec des adresses locales telle que le 127.0.0.1 qui n'est pas une attaque. Cependant, à l'activation de la détection d'intrusion, notre logiciel alerte l'utilisateur d'une connexion établie en local, ce qui pourrait être considéré comme non importante. C'est ainsi que nous avons proposé une whitelist d'adresses IP où l'utilisateur pourra ajouter ou supprimer les adresses dont il ne s'inquiète pas des connexions établies. Il en est de même pour les ports de connexion tels que le 443 et le 80 qui sont des ports de navigation web. L'utilisateur a donc la possiblité d'ajouter ou de

supprimer des ports dans la whitelist, ce qui ne déclenchera plus d'alerte à la découverte d'une connexion avec les paramètres entrés en whitelist. Après avoir modifié la base de données de la whitelist, il faut rafraîchir afin de faire considérer à l'application la nouvelle liste à prendre en compte.

### Les dénis de service

Comme nous l'avons présenté au chapitre précédent, un déni de service peut s'effectuer de plusieurs façons. Notre solution, pour cela, propose une option où lorsqu'elle est activée rejette tous les paquets au statut invalide et les paquets avec tous les flags activés (XMAS). Aussi limite-elle le nombre de paquets ICMP à deux (02) par seconde pour eviter les attaques de type Smurf.

### Limiter le nombre de connexion par adresse

Toujours dans l'optique d'arriver à une fin de déni de service, l'attaquant peut falsifier les adresses IP et envoyer des demandes de connexion, la cible peut se retrouver avec une immensité de connexions ouvertes pour une seule adresse. Pour eviter cela, nous proposons à l'utilisateur d'entrer le nombre de connexions maximales qu'il admet pour une adresse IP vers sa machine en dépendance des services qu'il fournit au réseau auquel il appartient. Une fois cette valeur entrée et validée, une règle est créée pour rejeter toute demande de connexion d'une quelconque adresse après avoir atteint la limite entrée par l'utilisateur. Ce dernier a la possibilité de modifier cette valeur à tout moment.

## Langage de développement et outils

Notre solution fonctionne sous les systèmes Linux. Cette plate-forme a été choisie en raison de son ouverture, de l'accessibilité au code source et de sa flexibilité. Pour atteindre nos objectifs, nous avons utilisé les langages java, awk et bash ; les deux derniers étant des langages de script de Linux. Il est à noter que les versions Debian sont les plus concernées par notre travail. Le tableau suivant fait la synthèse des outils utilisés.

TABLE 2.1 – Synthèse des outils utilisés

| Langages | Systèmes d'exploitation | Autres Outils |
|---|---|---|
| Java | Ubuntu 16.04 LTS | Iptables |
| Bash | Kali Linux 2.0 Rolling | |
| Awk | | |

Les langages et outils présentés dans le tableau sont ceux utilisés pour le développement de notre application pendant que les systèmes d'exploitations présentés sont ceux utilisés pour le cas pratique.

## A propos de Debian

Debian est un système d'exploitation libre. Il est simple et est constitué de plus de 4000 paquets. Les paquets sont des composants logiciels précompilés conçus pour s'installer facilement sur la machine hôte. L'ouverture de Debian permet à plusieurs programmeurs de pouvoir identifier les failles de sécurité ou de créer plusieurs modules pour faire des tâches qui s'avèrent indispensables. C'est ainsi que la communauté de Debian devient de plus en plus robuste et flexible. Plusieurs systèmes d'exploitation sont dérivés de Debian dont Kali Linux spécialisé dans les tests d'intrusion.

## Conclusion

Dans ce chapitre, nous avons présenté les choix techniques opérés ainsi que notre solution à travers sa modélisation, son principe de fonctionnement et les outils utilisés. Le chapitre suivant exposera les différents résultats et quelques critiques.

Chapitre 3

# Résultats et Discussion

## Introduction

Dans ce chapitre, nous présentons les résultats des simulations faites pour tester le fonctionnement de notre solution. Dans un premier temps, nous allons présenter quelques fonctionnalités développées dans l'application et nous aborderons ensuite une discussion.

## Présentation des résultats des tests

Pour réaliser les simulations, nous avons choisi d'utiliser deux machines toutes linux ; l'une sous kali linux 2.0 Rolling, l'autre sous ubuntu 16.04 LTS.

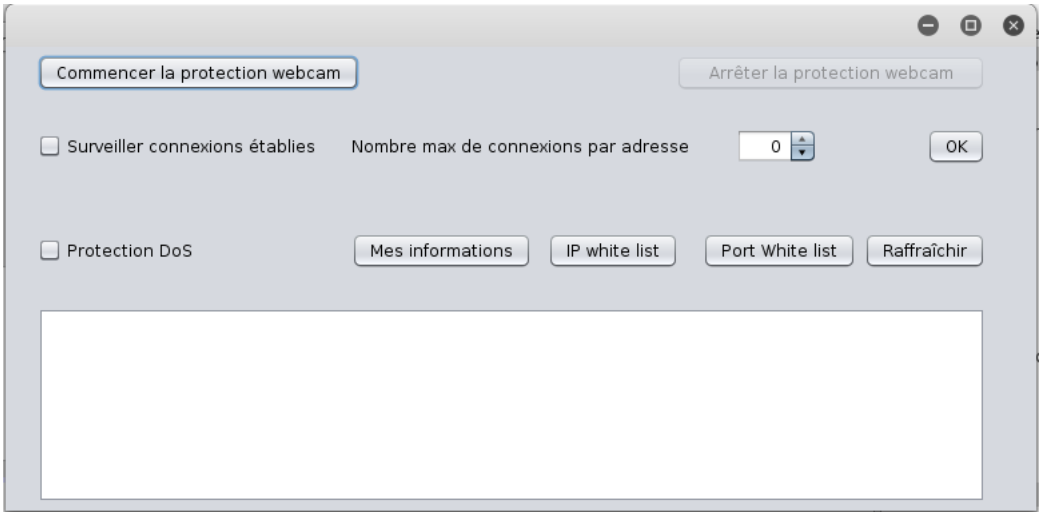


FIGURE 3.1 – Ecran d'accueil de notre solution

Cette figure présente l'écran à l'ouverture de notre logiciel avec les différentes options. Lorsque l'utilisateur clique sur whitelist IP, il est redirigé vers une autre page dont l'image suit.

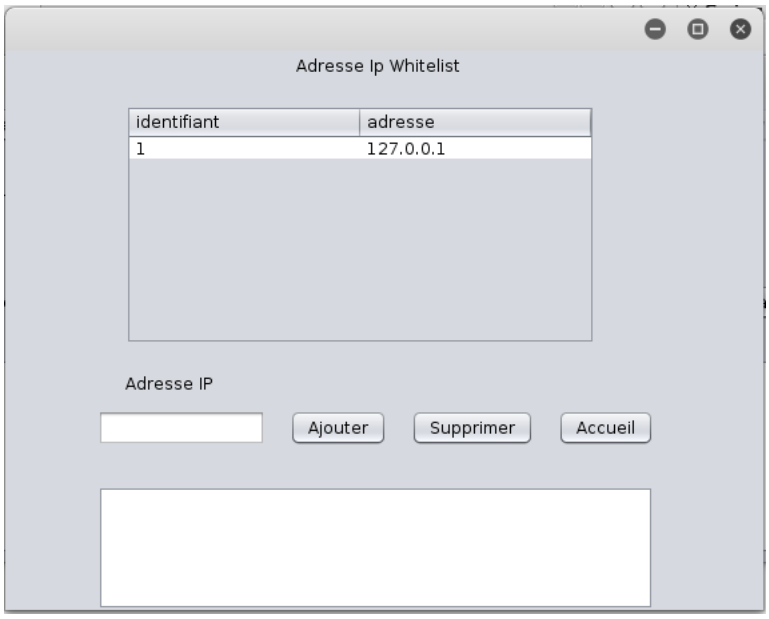


FIGURE 3.2 – Page de la whitelist IP

Lorsqu'il clique sur le bouton réservé à la whitelist des ports, il est renvoyé vers un autre écran dont l'image suit.

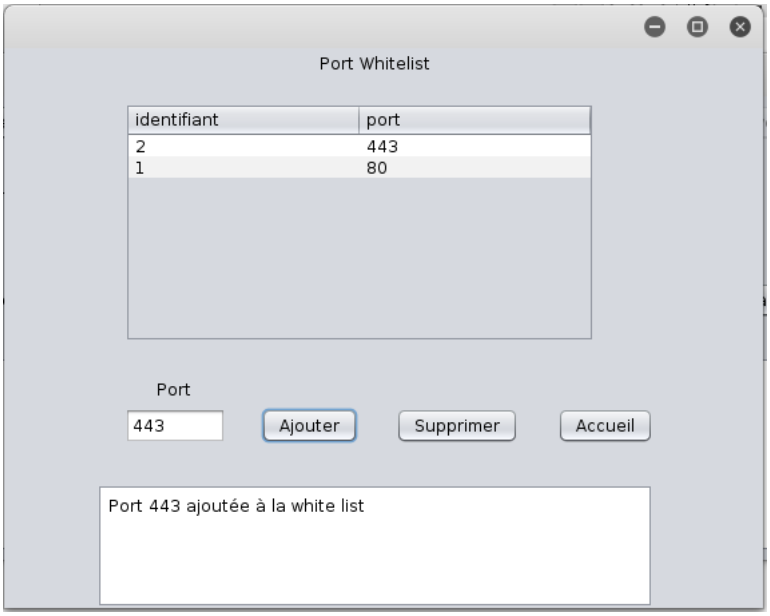


FIGURE 3.3 – Page de la whitelist Port

Les données visualisées sont présentes parce qu'elles ont été entrées préalablement dans la base de données.

## Cas pratique

Notre machine cible reçoit un mail comportant un lien pour voir les nouveautés de la communuauté Ubuntu. Après un clic sur le lien, il est redirigé vers un site ressemblant fortement à `community.ubuntu.com`, celui de la communauté Ubuntu, mais qui n'est rien d'autre qu'un site piégé dont voici l'aperçu.

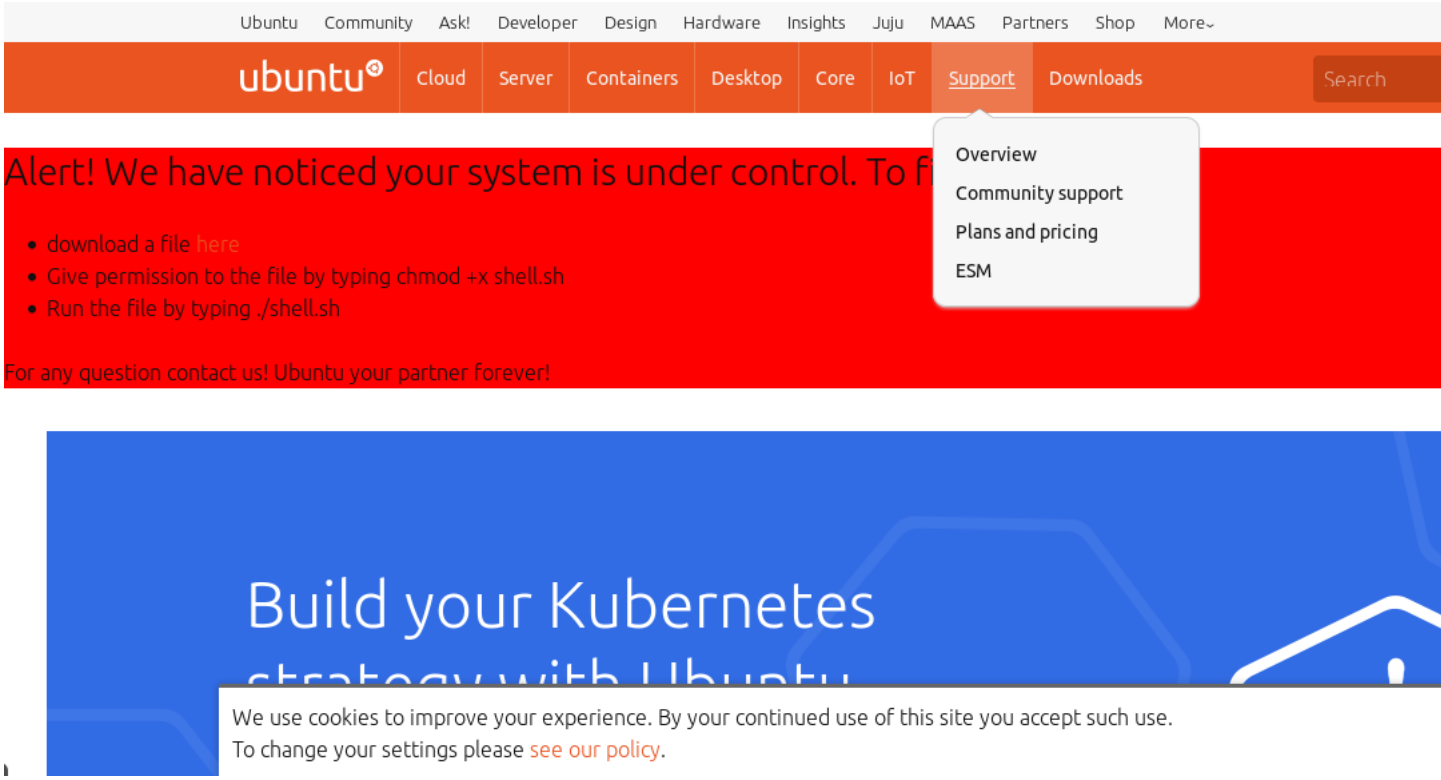


FIGURE 3.4 – Faux site conçu par l'attaquant

L'attaquant ajoute du contenu qui fait croire à la victime que son poste est sous contrôle et qu'il faudrait télécharger et exécuter un fichier contenant un script. Cela met la victime en confiance puisque, selon lui, il est audité depuis la communauté Ubuntu. Cependant, il ne fait que se piéger. L'attaquant utilise le framework Metasploit [**?**] pour lancer une écoute de session (listener[1]) vers toute cible essayant d'exécuter le fichier supposé résoudre le problème notifié sur le faux site. Après exécution du fichier, l'attaquant prend possesion de la machine cible.

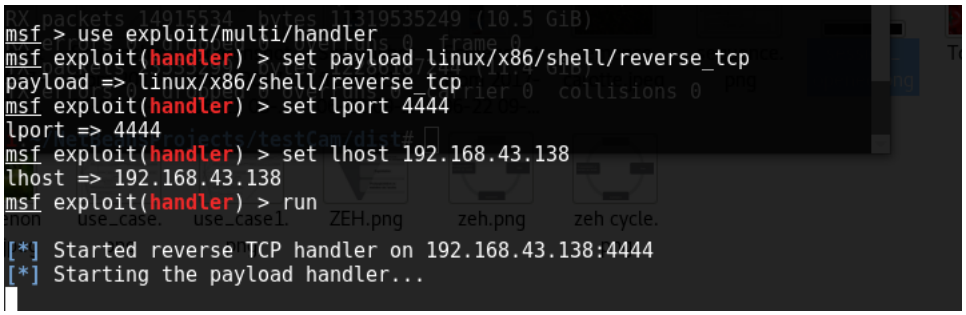


FIGURE 3.5 – Session en attente.

[1]Un listener est un composant de Metasploit qui attend une connexion entrante de tout type.

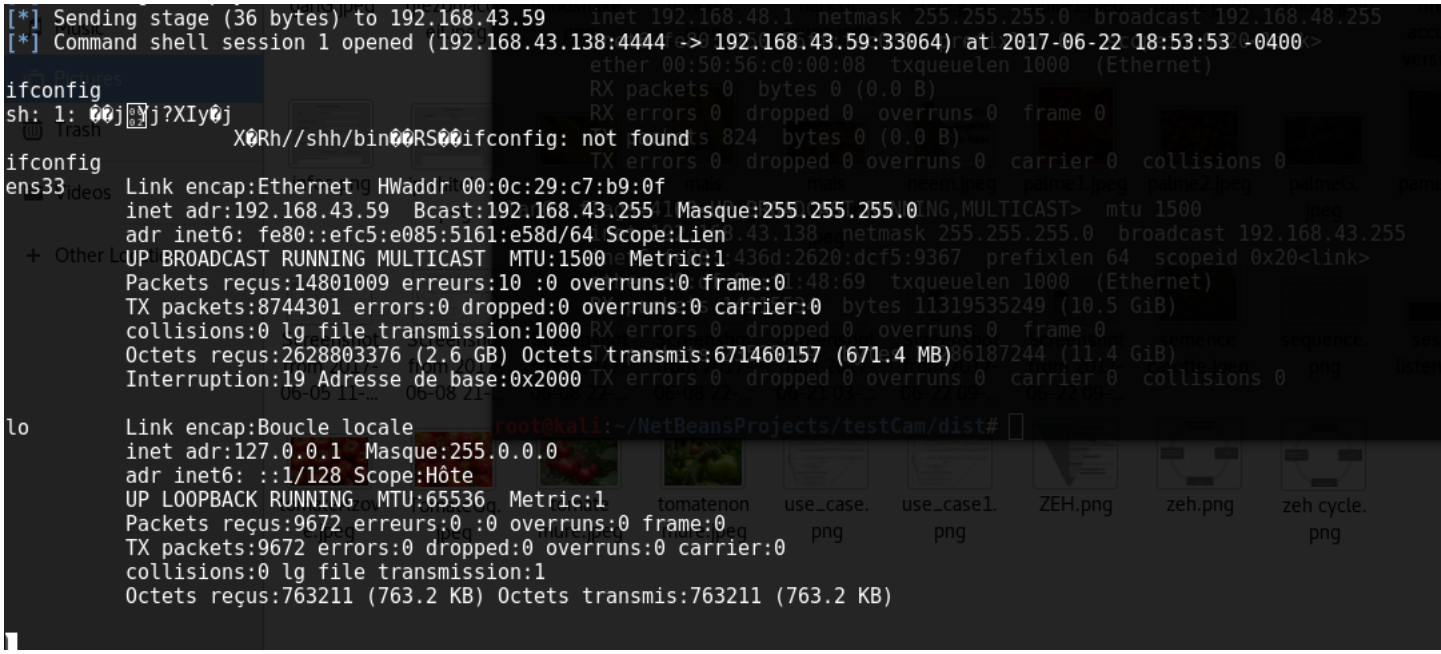


FIGURE 3.6 – Session ouverte vers la machine cible.

Lorsque l'utilisateur de la machine cible active la détection d'intrusions, une alerte lui est envoyée en précisant la connexion établie.

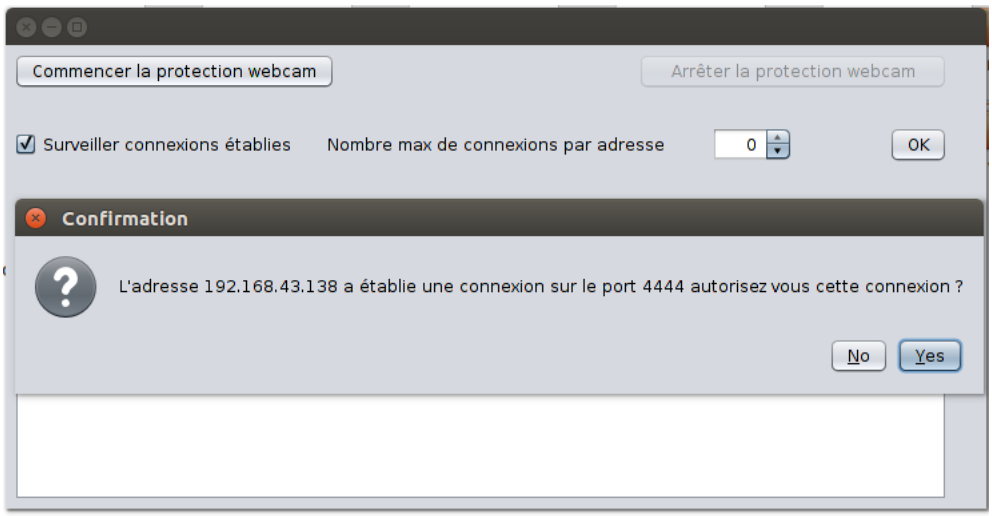


FIGURE 3.7 – Alerte connexion établie

Ne reconnaissant pas cette connexion comme légale, l'utilisateur décide d'arrêter. Cela met fin à la connexion établie et met en liste noire l'adresse source de l'attaquant.

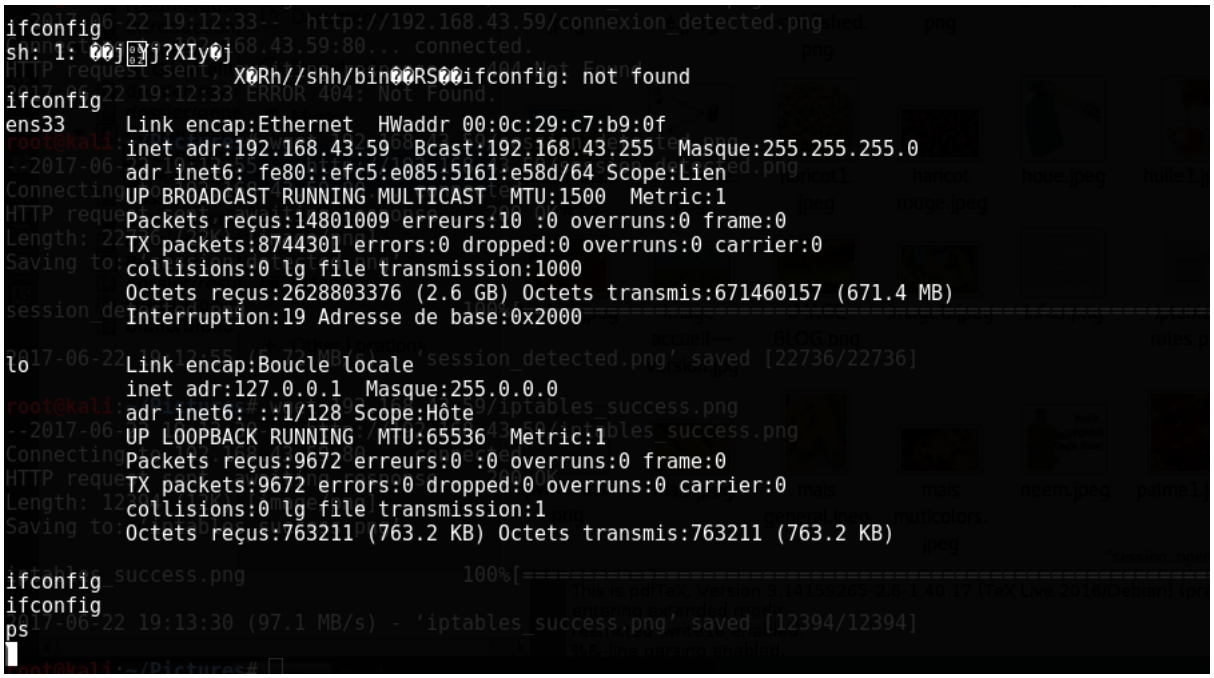


FIGURE 3.8 – Les commandes du shell ne fontionnant plus

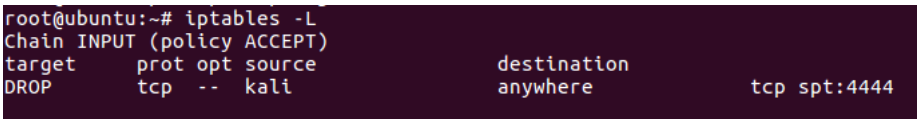


FIGURE 3.9 – Adresse et port en blacklist

## Discussion

Notre projet est né de plusieurs constats généraux de l'ordre de la sécurité informatique pour la protection de l'information et de la vie privée.

Les recherches effectuées nous ont permis de constater l'existence de certaines solutions. Cependant, ces solutions, loin de contrer tout type d'intrusion, ne peuvent pas être prises pour seuls moyens de sécurité.

Notre solution apporte un outil de défense contre certaines attaques en réseau informatique parmi lesquelles l'espionnage par webcam, les intrusions clandestines, les dénis de service. Aussi permet-elle de lever le mythe selon lequel on est en sécurité absolue sur les systèmes Linux (un exemple se présente dans le cas pratique).

Ainsi, d'une part, notre solution est simple à utiliser et est accessible à un grand public, celui regroupant les utilisateurs Linux. Elle permet aux utilisateurs d'être informés de quelques processus tournant à leur insu.

D'autre part, notre solution est exclusivement destinée aux systèmes Linux, ce qui représente une limite car, aujourd'hui beaucoup d'ordinateurs tournent sous les sytèmes Windows et Mac, sans compter la multiplicité des versions Linux qui existent.

## Conclusion

Dans ce chapitre, nous avons exposé les résultats des tests de notre application et réalisé quelques critiques concernant ses performances et ses insuffisances. Ces insuffisances peuvent être perçues comme des perspectives afin d'améliorer le travail fait pour une utilisation plus éfficiente.

# Conclusion

# Conclusion

La sécurité informatique passant par la sécurité de l'information, la sécurité de la vie privée n'est plus aujourd'hui un sujet inconnu. Aussi le hacking constitue-t-il aujourd'hui une arme de guerre assez dévastatrice et silencieuse. Dans cette vision, beaucoup d'outils de sécurité se développent de jour en jour afin de restreindre le champ d'attaque des pirates informatiques qui s'arment davantage.

L'objet de ce mémoire a été de mettre en place un système de détection des attaques par webcam et de prévenir par des moyens simples mais efficaces les tentatives d'intrusion. Les solutions proposées proviennent des analyses éffectuées sur les systèmes piratés volontairement. Ainsi les utilisateurs de notre application peuvent être protégés des différentes menaces que nous couvrons.

Dans ce mémoire, nous avons tout d'abord fait une revue de littérature autour des intrusions informatiques. Nous avons ensuite réalisé la conception de la solution que nous avons proposée avant d'exposer les résultats et critiques de l'application que nous avons implémentée.

Bien que notre solution réponde au besoin énoncé, il faut noter certaines insuffisances telles que l'inactivité de l'application dans les intervalles d'attente. Bien que cela soit fait pour éviter l'utilisation abusive des ressources du système, il serait préférable de mettre le système en mode écoute. D'une part, un autre axe de recherche en ce qui concerne les travaux futurs sera d'explorer la possibilité de rendre l'application accessible sur d'autres plateformes telles que Windows, MAC OS et toutes autres distributions Linux. D'autre part, on pourrait envisager de mettre la recherche de symptômes d'attaques en mode écoute en prenant des mesures autonomes. [**?**]

# Bibliographie