REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Université SAAD DAHLEB BLIDA
Faculté des sciences
Département d'informatique

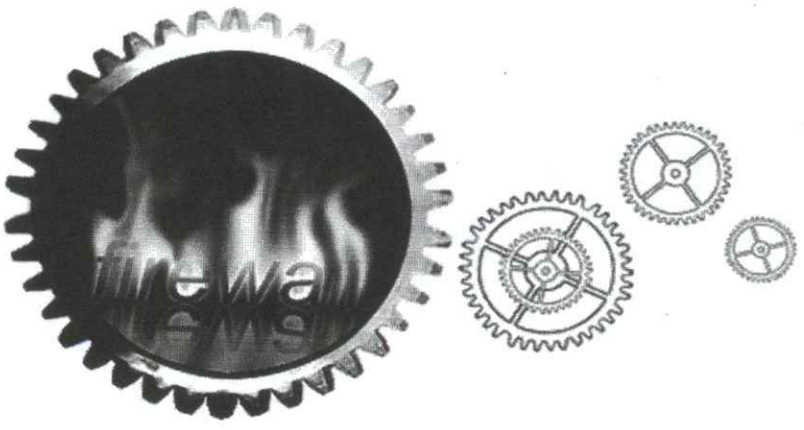
MEMOIRE

POUR L'OBTENTION DU DIPLOME
D'INGENIEUR D'ETAT EN INFORMATIQUE

OPTION INTELLIGENCE ARTIFICIELLE

THEME

Conception d'un Firewall Statique sous Windows



Réalisé par :
M^r HAMADI ABDELKRIM. Eddine.
M^r TALEB MOHAMED.

Promoteur :
M^r Menacer Djamel

Organisme d'accueil :
CNAS D'ALGER (Direction générale).
Département d'informatique.

Encadreur :
M^r AKKA ABDELHAKIM.

Promotion 2005 – 2006

DEDICACES

Je dédie ce modeste travail de fin d'études :

- A mon père qui été d'un support sans égal et sans lui je n'aurai jamais été ce que je suis.
- A ma mère pour son amoure,sa tendresse et son affectation sans limite et que j'adore par-dessus tout.
- A mes frères et sœur (Rafik, Noureddine, Assia, Hichem, Souad).
- A ma grand-mère (Mama Alzhore)qui je l'aime beaucoup.
- A tout ma famille A mes tantes a mes cousins et mes cousines.
- A tout mes amis surtout « Redha,Farouk,Ishak,Krimo… »
- A tout la promotion informatique de BLIDA 2006.

TALEB Mohamed

DEDICACES

Je dédie ce modeste travail
A mon cher père qui a toujours
été a coté de moi et qui m'a
toujours soutenu
A ma chère mère
A mes frères
Mehdi et Khalil Riad
A mon cher grand père Arab
Que dieu l'accueil dans son vaste paradis
A ma grande mère Zina
A mon cher grand père Said
Que dieu l'accueil dans son vaste paradis
A ma grande mère Mebarka
Que dieu l'accueil dans son vaste paradis
A mes oncles
A mes tantes
A mes cousins et mes cousines
A tout mes amis

Remerciements

Ils sont plusieurs à avoir le mérite dans l'accomplissement de ce modeste travail .Nous remercions, particulièrement :

- **Notre promoteur : Mr Menacer Djamel Eddine pour sa disponibilité, son aide et son orientation durant toute l'année.**

- **Notre encadreur : Mr AKKA AbdelHakim pour sa collaboration et ses encouragements.**

- **Tous les enseignants du département d'informatique.**

- **Tout le personnel du département d'informatique**

Sommaire

INTRODUCTION GENERALE

CHAPITRE 1 : La sécurité informatique

CHAPITRE 2 : Technologie de pare-feu

CHAPITRE 4 : Implémentation de la solution

CHAPITRE 5 : Environnement de test

INTRODUCTION GENERALE

INTRODUCTION GENERALE

- Introduction :

Tout système d'information (ou presque) est aujourd'hui, ne serait-ce qu'indirectement connecté a l'Internet, et ce de plus en plus souvent via un accès haut débit.

Une entreprise utilise généralement leur réseau informatique pour accéder aux données, aux applications et à la messagerie électronique. La sécurisation du réseau permet de résister aux attaques système, de protéger les données de la société et de maintenir un accès continu au réseau, 24 heures sur 24, 7 jours sur 7. Une infrastructure de réseau sécurisée peut aider votre société à :

- Empêcher les pannes système.
- Protéger le système d'information des virus et programmes malveillants.
- Gérer facilement les accès aux informations pour les utilisateurs en fonction de leur profil.

Notre objectif est la réalisation d'un système de sécurité réseau et Internet (firewall) qui est totalement personnalisable et il s'adaptera a l'infrastructure de l'entreprise ou il est déployé.

- Pourquoi développé un pare-feu?

- Nous voulons montrer aux utilisateurs qu'il est possible de développer une solution pare-feu lorsque les conditions d'acquisition des produits commerciaux ne sont pas réunies (coût, licence ...).
-Notre solution pourra constituer une solution commerciale à l'avenir.

-Comme la majorité des firewall ne peuvent pas être personnalisé selon un type de réseau spécifique, notre pare-feu sera totalement personnalisable et il s'adaptera à l'infrastructure de l'entreprise ou il est déployé.

-Notre pare-feu ne sera pas gourmand en ressources.

-Notre pare-feu ne demande pas de formation car il sera facile a utilisé.

CHAPITRE 1

LA SECURITE INFORMATIQUE

1 - LA SECURITE INFORMATIQUE

1-1/ Qu'est-ce que la sécurité informatique ?

La sécurité informatique, d'une manière générale, consiste à assurer que les ressources matérielles ou logicielles d'une organisation sont uniquement utilisées dans le cadre prévu ***[14]***.

La sécurité informatique consiste généralement en quatre principaux objectifs ***[14]***:

- L'**intégrité**, c'est-à-dire garantir que les données sont bien celles qu'on croit être.
- La **confidentialité**, consistant à assurer que seules les personnes autorisées aient accès aux ressources
- La **disponibilité**, permettant de maintenir le bon fonctionnement du système informatique.
- La **non répudiation**, permettant de garantir qu'une transaction ne peut être niée.

1-2/Domaines de la sécurité informatique

Parmi les domaines de la sécurité informatique on a :

- **L'authentification :**

Procédure dont le but est de s'assurer de l'identité d'une personne pour contrôler l'accès à un système d'informations où a un logiciel ***[14]***.

- **Le chiffrement :**

En cryptographie, le **chiffrement** (parfois appelé *cryptage*) est le procédé grâce auquel on peut rendre la compréhension d'un document impossible à toute personne qui n'a pas la clé de chiffrement ***[14]***.

- **Le stockage sécurisé :**

Consiste a mettre a l'abri l'ensemble des données.

- **Protection du réseau de l'entreprise /5/ :**

La sécurité du réseau nécessite bien plus que la seule capacité à réagir aux attaques et autres menaces. Pour une sécurité optimale, il est nécessaire d'être proactif. La meilleure protection pour votre réseau et pour vos données consiste en une solution de sécurité capable d'arrêter les attaques système inattendues, anticipées et réelles, de manière efficace. La mise en place d'une solution de sécurité complète peut faire gagner à votre entreprise du temps et de l'argent.

1-3 Pourquoi la sécurité informatique

Le déploiement fulgurant de l'Internet et son omniprésence en tant que moyen de communication auraient du entraîner la prise en compte des risques associés a la visibilité des machines sur ce réseau de réseaux. Il n'en a pas été ainsi : de plus en plus de moyens informatiques se trouvent exposés a la malveillance des pirates.

La démocratisation du haut débit, aussi bien dans les écoles et les universités que dans les entreprises et chez les particuliers, doit s'accompagner d'une prise de conscience des risques liés à la visibilité de machines sur Internet et a la possible malveillance des pirates.
Trois facteurs rendent indispensable le déploiement de la sécurité informatique :

- La préservation du patrimoine de l'entreprise.
- L'existence d'une menace extérieure, même potentielle.
- Les failles des systèmes.

1-4 Enjeux et objectifs de sécurité

Les responsables de certains sites croient parfois a tort que, les données qu'ils abritent n'étant pas confidentielles, l'enjeu de la sécurité est nul pour leur entreprise. Pour autant, accepteraient-ils une indisponibilité de leurs ressources 80% du temps pour cause de réinstallation suite a une
compromission ? Supporteraient –ils que l'accès réseau , qu'ils payent fort cher chaque mois, soit utilisé a 99% pour un site warez et se trouve indisponible pour leurs besoins ? Se satisferaient-ils d'être mis en liste noire par leurs correspondants pour avoir négligé un serveur de messagerie qui autorise le relais ? Accepteraient-ils Que leurs machines soient mises en cause dans la compromission de tel ou tel site renommé ?

Ainsi, quel que soit le site considéré, il existe toujours une exigence minimale de fonctionnement qui justifie la mise en place de mesures de sécurité adaptées.

Il est important que les responsables de l'entreprise soient directement impliqués dans le définition des enjeux de la sécurité informatique pour deux raisons :
- La direction du site est capable mieux que quiconque de définir le type d'incidents que les mesures mises en place doivent permettre d'éviter et à quel prix.
- En outre, si cela s'avère nécessaire, c'est aussi elle qui est le mieux placée pour arbitrer, par exemple, entre le besoin en fonctionnalités et la mise en place d'une mesure contraignante.

D'autre part, il est indispensable de rappeler clairement aux utilisateurs quels sont les objectifs de l'entreprise, pour aboutir à un consensus sur l'arbitrage nécessaire entre convivialité et sécurité.

1-5 La menace

Quelque 250 millions de machines sont aujourd'hui connectées sur l'Internet**[1].** Il est facile d'imaginer que même si la plupart des internautes sont inoffensifs, il en existe que l'envie de nuire ou de jouer amènera à s'attaquer à des machines, même assez bien protégées **[1].** A cette fatalité statistique s'ajoute le sentiment dont jouira un pirate qui s'attaque a votre machine,

connecté depuis une chambre d'hôtel a 12000 km de chez vous[1]. Les pirates l'ont bien compris, ils utilisent de nombreuses astuces pour se protéger [1].

❖ Principaux facteurs de motivation des pirates

Les principaux facteurs de motivation des pirates sont les suivantes :

- Le goût du défi : certains pirates aiment prouver leur habileté et l'étendue de leurs connaissances.
- L'appât du gain : certains sont appâtés par les rémunérations qu'offrent des entreprises peu scrupuleuses qui souhaitent saboter l'outil de travail informatique de leur concurrent et/ou lui dérober des informations confidentielles (devis, plans secrets industriels…).
- La volonté de détourner à son profit des ressources informatiques dont on ne dispose pas(puissance de calcul, espace disque, connexion rapide au réseau…).
- La méconnaissance des conséquences et des risques encourus par des pirates aveuglément hostiles.

❖ Risques liés au type de connexion

Les connexions permanentes à haut débit sont très recherchées par certaines catégories de pirates, dont l'objectif est d'utiliser cette ressource pour distribuer efficacement films et logiciels piratés.

Face à ces pirates qui cherchent des ressources afin d'abriter leurs sites, tant que vous êtes connectés à votre fournisseur d'accès Internet via un bon vieux modem(RTC), le danger reste limité. En effet, la faible probabilité que le pirate vous trouve connecté, ajoutée au manque d'intérêt qu'il aurait à prendre le contrôle d'une ressource connectée par intermittences à 33Kbits/s rend la compromission improbable. Dans ce cas, la sécurité concernera plutôt les problèmes de propagation de virus via la messagerie électronique.

Le fait nouveau aujourd'hui est l'arrivée ou plutôt la démocratisation d'Internet chez les particuliers et dans les petites entreprises via des connexions permanentes à « haut » débit(câble, ADSL).Cette démocratisation se fait pas sans heurts, si la composante sécurité n'est pas correctement prise en compte.

❖ Risques liés aux failles des systèmes

Si les systèmes informatiques ne présentaient aucune faille, ni dans leur conception, ni dans leur configuration, il ne serait pas nécessaire de s'inquiéter de sécurité informatique.
On pourrait alors considérer que la menace décrite ci-dessus ne met pas en péril les enjeux importants pour l'entreprise.
Mais c'est loin d'être le cas et il n'existe hélas pas de système d'exploitation qui ne présente son lot de vulnérabilités.

1-6/Conclusion

L'Internet fut conçu a une époque ou l'on était « entre gens de bonne compagnie ».Mais du fait de la croissance rapide du nombre de machines connectées en permanence, la sécurité informatique est devenue un enjeu.

Toute entité visible sur l'Internet doit définir des objectifs de sécurité, face à une menace devenue importante et face aux failles bien réelles des systèmes d'exploitation.

CHAPITRE 2
TECHNOLOGIE DE PARE-FEU

2 - Technologie de pare-feu

2-1/ Modèle OSI :

Le **modèle OSI** (*Open Systems Interconnection)* modèle de référence d'interconnexion de systèmes ouverts a été créé à la fin des années 1970 par l'ISO (Organisation internationale de normalisation) norme ISO 7498 dans le but d'offrir une base commune à la description du processus de fonctionnement de tout réseau informatique et pour que les matériels et applications provenant de différents constructeurs et éditeurs puissent être compatibles entre eux **[4]**.

Dans ce modèle, l'ensemble des protocoles d'un réseau est décomposé en 7 parties appelées *couches OSI*, numérotées de *1* à *7*. Les couches OSI respectent les principes suivants :

- Chaque couche décrit un protocole indépendamment des autres couches.
- Chaque couche procure des services à la couche immédiatement supérieure .
- Chaque couche requiert les services de la couche immédiatement inférieure .
- La couche 1 utilise le médium (le support de communication) .
- La couche 7 procure des services à l'utilisateur ou à un programme informatique .

Lors d'une communication, l'utilisateur d'un réseau utilise les services de la couche5 (IRC par exemple) via un programme**[4]**.

Cette couche met en forme et enrichit l'information qu'elle reçoit du programme en respectant son protocole, puis elle l'envoie à la couche inférieure lors d'une demande de service. À chaque couche, l'information subit des mises en formes et des ajouts en fonction des protocoles utilisés. Enfin, elle est envoyée sur le médium et reçue par un autre nœud du réseau. Elle parcourt toutes les couches de ce nœud dans l'autre sens pour finir dans le programme IRC du correspondant, dépouillée des différents ajouts liés aux protocoles.

Les 7 couches du modèle OSI se décomposent en deux groupes :

- **applicatif** : les 3 couches supérieures définissent la façon dont les applications vont communiquer entre elles et avec les utilisateurs finaux.

- **transport** : les 4 couches inférieures définissent la façon dont les données sont transmises d'un point à un autre.

Le modèle à 7 couches proposé par l'OSI a fait l'objet d'implémentations chez divers constructeurs, mais sans succès commercial, le marché s'étant largement orienté vers le modèle à 4 couches de TCP/IP, plus facile à comprendre et pour lequel existaient déjà des implémentations portables. Le modèle garde toutefois un intérêt *théorique*, bien que les frontières des 4 couches TCP/IP ne correspondent pas à d'exacts équivalents en OSI**[4]**.

- **Analyse d'un échec**

On peut parler dans le cas de l'OSI d'un échec historique, le modèle n'étant aujourd'hui que partiellement utilisé, en comparaison du modèle TCP/IP **[4]**.

Des éléments d'analyse ont été proposés. Le modèle s'est développé à partir des couches basses, vers le haut. Il a été complété de façon apparemment satisfaisante jusqu'à la couche 4 comprise, dont la complexité était déjà importante. Les couches supérieures, session, présentation et surtout application, n'ont en fait jamais été complétées **[4]**.

La couche application n'était en fait pas une vraie couche, composée d'éléments de service s'utilisant les uns les autres suivant les besoins des configurations **[4]**.

Deux domaines d'applications ont révélé certaines des inadaptations du modèle : la sécurité et la gestion. Ni l'une ni l'autre ne pouvaient se limiter aux couches existantes. De plus, le modèle rendait mal compte de pratiques telles que le *tunneling* dans laquelle l'ordre des couches était remis en question **[4]**.

Cet échec rejaillit sur le modèle ancestral de la décomposition ordonnée (par couches), qui touche là à ses limites en termes de gestion de la complexité **[4]**.

Pile de protocoles	Modèle OSI
Couche	Protocoles
Application	Gopher, Telnet, SSH, FTP, HTTP, HTTPS, NNTP, DNS, SNMP, SMTP, POP3, IMAP, IRC, VoIP, WebDAV, SIMPLE, …
Présentation	Videotex, Unicode, MIME, HTTP/HTML, XML, TDI, ASN.1, XDR, UUCP, NCP, AFP, SSP, SSL, TLS, …
Session	RTSP, H.323, SIP, NFS, Netbios, CIFS, AppleTalk, …
Transport	TCP, UDP, SCTP, RTP, SPX, TCAP, DCCP , …
Réseau	NetBEUI, IPv4, IPv6, ARP, IPX, BGP, ICMP, OSPF, RIP, IGMP, IS-IS, CLNP, WDS, …
Liaison	Ethernet, Anneau à jeton, LocalTalk, FDDI, X.21, X.25, Frame Relay, BitNet, CAN, ATM, Wi-Fi, …
Physique	CSMA/CD, CSMA/CA, Codage NRZ, Codage Manchester, Codage Miller, RS-232, RS-449, V.21-V.23, V.42-V.90, Câble coaxial, 10Base2, 10BASE5, Paire torsadée, 10BASE-T, 100BASE-TX, ISDN, PDH, SDH, T-carrier, EIA-422, EIA-485, SONET, ADSL, SDSL, VDSL, DSSS, FHSS, IrDA, USB, IEEE 1394, Wireless USB, …

Fig1 : Le modèle OSI **[4].**

La **suite des protocoles Internet** est l'ensemble des protocoles qui constituent la pile de protocoles utilisée par Internet. Elle est souvent appelée **TCP/IP**, d'après le nom de deux de ses protocoles : TCP (Transmission Control Protocol) et IP (Internet Protocol), qui ont été les premiers à être définis. Le document de référence sur ce sujet est le RFC 1122**[4]**.

Le modèle OSI , qui décompose les différents protocoles d'une pile en 7 couches, peut être correspondent pas toujours avec les habitudes d'Internet (Internet étant basé sur TCP/IP qui ne comporte que 4 couches). Dans une pile de protocoles, chaque couche résout un certain nombre de problèmes relatifs à la transmission de données, et fournit des services bien définis aux couches supérieures. Les couches hautes sont plus proches de l'utilisateur et gèrent des données plus abstraites, en utilisant les services des couches basses qui mettent en forme ces données afin qu'elles puissent être émises sur un médium physique**[4]**.

Le modèle Internet a été créé afin de répondre à un problème pratique, alors que le modèle OSI correspond à une approche plus théorique, et a été développé plus tôt dans l'histoire des réseaux. Le modèle OSI est donc plus facile à comprendre, mais le modèle TCP/IP est le plus utilisé en pratique **[4]**.

2-2/ Couches dans la pile TCP/IP

Comme les suites de protocoles TCP/IP et OSI ne correspondent pas exactement, toute définition des couches de la pile TCP/IP peut être sujette à discussion.

En outre, le modèle OSI n'offre pas une richesse suffisante au niveau des couches basses pour représenter la réalité ; il est nécessaire d'ajouter une couche supplémentaire d'interconnexion de réseaux (Internetworking) entre les couches Transport et Réseau. Les protocoles spécifiques à un type de réseau particulier, mais qui fonctionnent au-dessus de la couche de liaison de données, devraient appartenir à la couche réseau. ARP, et STP (qui fournit des chemins redondants dans un réseau tout en évitant les boucles) sont des exemples de tels protocoles. Toutefois, ce sont des protocoles locaux, qui opèrent au-dessous de la fonction d'interconnexion de réseaux ,placer ces deux groupes de protocoles (sans parler de ceux qui fonctionnent au-dessus du protocole d'interconnexion de réseaux, comme ICMP) dans la même couche peut prêter à confusion**[4]**.

Le schéma qui suit essaie de montrer où se situent divers protocoles de la pile TCP/IP dans le modèle OSI **[4]**:

7	**Application**	ex. HTTP, SMTP, SNMP, FTP, Telnet, NFS
6	**Présentation**	ex. XDR, ASN.1, SMB, AFP
5	**Session**	ex. ISO 8327 / CCITT X.225, RPC, Netbios, ASP
4	**Transport**	ex. TCP, UDP, RTP, SPX, ATP
3	**Réseau**	ex. IP, ICMP, IGMP, X.25, CLNP, ARP, OSPF, RIP, IPX, DDP
2	**Liaison**	ex. Ethernet, Token Ring, PPP, HDLC, Frame relay, RNIS, ATM
1	**Physique**	ex. électronique, radio, laser

Habituellement, les trois couches supérieures du modèle OSI (Application, Présentation et Session) sont considérées comme une seule couche Application dans TCP/IP. Comme TCP/IP n'a pas de couche session unifiée sur laquelle les couches plus élevées peuvent s'appuyer, ces fonctions sont généralement remplies par chaque application (ou ignorées). Une version simplifiée de la pile selon le modèle TCP/IP est présentée ci-après**[4]** :

5	**Application** *« couche 7 »*	ex. HTTP, FTP, DNS *(les protocoles de routage comme RIP, qui pour des raisons obscures fonctionnent au-dessus d'UDP, peuvent aussi être considérés comme faisant partie de la couche réseau)*
4	**Transport**	ex. TCP, UDP, RTP *(les protocoles de routage comme OSPF, qui fonctionnent au-dessus d'IP, peuvent aussi être considérés comme faisant partie de la couche réseau)*
3	**Réseau**	Pour TCP/IP il s'agit de IP *(les protocoles requis comme ICMP et IGMP fonctionnent au-dessus d'IP, mais peuvent quand même être considérés comme faisant partie de la couche réseau ; ARP ne fonctionne pas au-dessus d'IP)*
2	**Liaison**	ex. Ethernet, Token Ring, etc.
1	**Physique**	ex. réseau physique, et techniques de codage

- **La couche physique**

La couche physique décrit les caractéristiques physiques de la communication, comme les conventions à propos de la nature du médium utilisé pour les communications (les câbles, les liens par fibre optique ou par radio), et tous les détails associés comme les connecteurs, les types de codage ou de modulation, le niveau des signaux, les longueurs d'ondes, la synchronisation et les distances maximales.

- **La couche de liaison de données**

La couche de liaison de données spécifie comment les paquets sont transportés sur la couche physique, et en particulier le *tramage* (i.e. les séquences de bits particulières qui marquent le début et la fin des paquets). Les en-têtes des trames Ethernet, par exemple, contiennent des champs qui indiquent à quelle(s) machine(s) du réseau un paquet est destiné. Exemples de protocoles de la couche de liaison de données : Ethernet , Wireless Ethernet , SLIP, Token Ring et ATM.

PPP est un peu plus complexe, car il a été initialement spécifié pour fonctionner au-dessus d'un autre protocole de liaison de données

Cette couche est parfois subdivisée en LLC et MAC.

- **La couche réseau**

Dans sa définition d'origine, la couche de réseau résout le problème de l'acheminement de paquets à travers un seul réseau. Exemples de protocoles de ce type :X.25 et le Initial Connection Protocol d'ARPANET.

Avec l'avènement de la notion d'interconnexion de réseaux, des fonctions additionnelles ont été ajoutées à cette couche, et plus spécialement l'acheminement de données depuis un réseau source vers un réseau destinataire. Ceci implique généralement le routage des paquets à travers un réseau de réseaux, connu sous le nom d'Internet. Dans la suite de protocoles Internet, IP assure l'acheminement des paquets depuis une source vers une destination, et supporte aussi d'autres protocoles, comme ICMP (utilisé pour transférer des messages de diagnostic liés aux transmissions IP) et IGMP (utilisé pour gérer les données multicast). ICMP et IGMP sont situés au-dessus d'IP, mais assurent des fonctions de la couche réseau, ce qui illustre l'incompatibilité entre les modèles Internet et OSI.

La couche réseau IP peut transférer des données pour de nombreux protocoles de plus haut niveau. Ces protocoles sont identifiés par un *numéro de protocole IP (IP Protocol Number)* unique. ICMP et IGMP sont respectivement les protocoles 1 et 2.

- **La couche transport**

Les protocoles de la couche de transport peuvent résoudre des problèmes comme la fiabilité des échanges (« est-ce que les données sont arrivées à destination ? ») et assurer que les données arrivent dans l'ordre correct. Dans la suite de protocoles TCP/IP, les protocoles de transport déterminent aussi à quelle application chaque paquet de données doit être délivré.

Les protocoles de routage dynamique qui se situent réellement dans cette couche de la pile TCP/IP (puisqu'ils fonctionnent au-dessus d'IP) sont généralement considérés comme faisant partie de la couche réseau. Exemple : OSPF (protocole IP numéro 89).

TCP (protocole IP numéro 6) est un protocole de transport « fiable », orienté connexion, qui fournit un flux d'octets fiable assurant l'arrivée des données sans altérations et dans l'ordre,

avec retransmission en cas de perte, et élimination des données dupliquées. Il gère aussi les données « urgentes » qui doivent être traitées dans le désordre (même si techniquement, elles ne sont pas émises hors bande). TCP essaie de délivrer toutes les données correctement et en séquence - c'est son but et son principal avantage sur UDP, même si ça peut être un désavantage pour des applications de transfert ou de routage de flux en temps-réel, avec des taux de perte élevées au niveau de la couche réseau.

UDP (protocole IP numéro 17) est un protocole simple, sans connexion, « non fiable » - ce qui ne signifie pas qu'il est particulièrement peu fiable, mais qu'il ne vérifie pas que les paquets sont arrivés à destination, et ne garantit pas leur

arrivée dans l'ordre. Si une application a besoin de ces garanties, elle doit les assurer elle-même, ou bien utiliser TCP. UDP est généralement utilisé par des applications de diffusion multimédia (audio et vidéo, etc.) pour lesquelles le temps requis par TCP pour gérer les retransmissions et l'ordonnancement des paquets n'est pas disponible, ou pour des applications basées sur des mécanismes simples de question/réponse comme les requêtes DNS, pour lesquelles le surcoût lié à l'établissement d'une connexion fiable serait disproportionné au besoin.

Aussi bien TCP qu'UDP sont utilisés par de nombreuses applications. Les applications situées à une quelconque adresse réseau se distinguent par leur numéro de port TCP ou UDP. Par convention, des *ports bien connus* sont associés avec certaines applications spécifiques.

RTP (Real Time Protocol) est un protocole fonctionnant avec UDP ou TCP, spécialisé dans le transport de données possédant des contraintes temps réel. Typiquement, il sert à transporter des vidéo pour que l'on puisse synchroniser la lecture des images et du son directement, sans les stocker préalablement.

- **La couche application**

C'est dans la couche application que se situent la plupart des programmes réseau.

Ces programmes et les protocoles qu'ils utilisent incluent HTTP (World Wide Web), FTP transfert de fichiers), SMTP

(messagerie), SSH (connexion à distance sécurisée), DNS (recherche de correspondance entre noms et adresses IP) et beaucoup d'autres.

Les applications fonctionnent généralement au-dessus de TCP ou d'UDP, et sont souvent associées à un *port bien connu*. Exemples :

- HTTP port TCP 80 ou 8080 .
- SSH port TCP 22 .
- DNS port UDP (et parfois TCP) 53 .
- RIP port UDP 520.

Ces ports ont été assignés par l'Internet Assigned Numbers Autority
Sous UNIX, on trouve un fichier texte servant à faire les correspondances port↔protocol: /etc/services.

2-3/ Port (logiciel)

Correspondant à la couche session du modèle OSI, la notion de **port** logiciel permet, sur un ordinateur donné, de distinguer différents interlocuteurs. Ces interlocuteurs sont des programmes informatiques qui, selon les cas, écoutent ou émettent des informations sur ces ports **[4]**.

- **Utilité**

Grâce à cette abstraction, on peut exécuter plusieurs logiciels serveurs sur une même machine, et même simultanément des logiciels clients et des serveurs, ce qui est fréquent sur les systèmes d'exploitation multitâches et multiutilisateurs**[4]**.

- **Attribution des ports**

Pour chaque port, un numéro lui est attribué, qui est codé sur 16 bits, ce qui implique qu'il existe un maximum de 65 536 ports (2^{16}) par ordinateur.

L'attribution des ports est faite par le système d'exploitation, sur demande d'une application. Cette dernière peut demander à ce que le système d'exploitation lui attribue n'importe quel port, à condition qu'il ne soit pas déjà attribué. L'application peut ensuite l'utiliser comme bon lui semble **[4]**.

Lorsqu'un logiciel client veut dialoguer avec un logiciel serveur, aussi appelé service, il a besoin de connaître le port écouté par ce dernier. Les ports utilisés par les services devant être connus par les clients, les principaux types de services utilisent des ports qui sont dits réservés. Par convention, ce sont tous ceux compris entre 0 et 1 023 inclus. Les services utilisant ces ports sont appelés les Well-Known Services (les services célèbres) **[4]**.

Sur une machine de type UNIX, le fichier */etc/services* liste ces services célèbres, dont les plus connus sont notamment le port :

- 21, pour l'échange de fichiers via FTP.
- 25, pour l'envoi d'un courrier électronique en direction d'un serveur via SMTP.
- 80, pour la consultation d'un serveur HTTP par le biais d'un navigateur web.
- 110, pour la récupération de son courrier électronique via POP.
- 443, pour les serveurs Web sécurisés HTTPS.

Il est important de noter que rien n'empêche d'exécuter un serveur Web sur un port différent du 80 (il arrive couramment qu'un serveur Web tourne sur le port 8080). Pour que cela fonctionne, il faut juste spécifier au navigateur le port utilisé**[4]**.

2-4/ Les protocoles :

2-4-1/ Transmission Control Protocol

Le ***Transmission Control Protocol*** (**TCP**, « protocole de contrôle de transmissions »), est un protocole de transport fiable, en mode connecté, documenté dans la RFC 793 de l'IETF**[2].**

Dans le modèle TCP/IP, TCP est situé entre la couche de réseau (généralement le protocole IP), et la couche application. Les applications transmettent des flux d'octets sur le réseau. TCP découpe le flux d'octets en *segments*, dont la taille dépend de la MTU du réseau sous-jacent (couche liaison de données).

- **Fonctionnement**

Une session TCP fonctionne en trois phases :

- l'établissement de la connexion.
- les transferts de données.
- la fin de la connexion.

L'établissement de la connexion se fait par une poignée de main en trois temps. La rupture de connexion, elle, utilise une poignée de main en quatre temps. Pendant la phase d'établissement de la connexion, des paramètres comme le numéro de séquence sont initialisés afin d'assurer la transmission fiable (sans perte et dans l'ordre) des données.

- **Structure d'un segment TCP**

<table>
<tr><td colspan="8">Port Source</td><td colspan="8">Port destination</td></tr>
<tr><td colspan="16">Numéro de séquence</td></tr>
<tr><td colspan="16">Numéro d'acquittement</td></tr>
<tr><td>Taille de l'en-tête</td><td colspan="3">réservé</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="8">Fenêtre</td></tr>
<tr><td colspan="8">Checksum</td><td colspan="8">Pointeur de données urgentes</td></tr>
<tr><td colspan="12">Options</td><td colspan="4">Remplissage</td></tr>
<tr><td colspan="16">Données</td></tr>
</table>

Fig2 : Structure d'un segment TCP[2].

Signification des champs :

- Port source : Numéro du port source.
- Port destination : Numéro du port destination.
- Numéro de séquence : Numéro de séquence du premier octet de ce segment.
- Numéro d'acquittement : Numéro de séquence du prochain octet attendu .
- Taille de l'en-tête : Longueur de l'en-tête en mots de 32 bits (les options font partie de l'en-tête) .
- Réservé : Réservé pour un usage futur .
- Drapeaux
  - URG : Signale la présence de données URGentes
  - ACK : Signale que le paquet est un accusé de réception (ACKnowledgement)
  - PSH : Données à envoyer tout de suite (PuSH)
  - RST : Rupture anormale de la connexion (ReSeT)
  - SYN : Demande de SYNchronisation ou établissement de connexion
  - FIN : Demande la fin de la connexion
- Fenêtre : Taille de fenêtre demandée, c'est-à-dire le nombre d'octets que le récepteur souhaite recevoir sans accusé de réception
- Checksum : Somme de contrôle (CRC, Cyclic Redundancy Check) calculé sur l'ensemble de l'en-tête TCP et des données, mais aussi sur un pseudo en-tête (extrait de l'en-tête IP)
- Pointeur de données urgentes : Position relative des dernières données urgentes
- Options : Facultatives
- Remplissage : Zéros ajoutés pour aligner les champs suivants du paquet sur 32 bits, si nécessaire
- Données : Séquences d'octets transmis par l'application (par exemple: +OK POP3 server ready, ...)

- **Établissement d'une connexion**

Même s'il est possible, pour deux systèmes, d'établir une connexion entre eux simultanément, dans le cas général, un système ouvre un 'socket' (point d'accès à une connexion TCP) et se met en attente passive de demandes de connexion d'un autre système. Ce fonctionnement est communément appelé *ouverture passive*, et est utilisé par le côté *serveur* de la connexion. Le côté *client* de la connexion effectue une *ouverture active* en envoyant un segment SYN au serveur, ce qui constitue la première étape de la poignée de mains en trois temps. Le serveur doit répondre à un segment SYN valide par un segment SYN/ACK. Enfin, le client répond au serveur avec un segment ACK, complétant la poignée de main en trois temps, et donc la phase d'établissement de la connexion.

- **Transferts de données**

Pendant la phase de transferts de données, certains mécanismes clefs permettent d'assurer la robustesse et la fiabilité de TCP. En particulier, les numéros de séquence sont utilisés afin d'ordonner les segments TCP reçus et de détecter les données perdues, les checksums permettent la détection d'erreurs, et les acquittements ainsi que les temporisations permettent la détection des segments perdus ou retardés.

Pendant la phase d'établissement de la connexion, les numéros de séquence initiaux sont échangés par les deux interlocuteurs. Ces numéros de séquence sont utilisés pour décompter les données dans le flux d'octets. On trouve toujours deux de ces nombres dans chaque segment TCP, qui sont le *numéro de séquence* et le *numéro d'acquittement*. Le *numéro de séquence* représente le propre numéro de séquence de l'émetteur TCP, tandis que le *numéro d'acquittement* représente le numéro de séquence du destinataire. Afin d'assurer la fiabilité de TCP, le destinataire doit acquitter les segments reçus en indiquant qu'il a reçu toutes les données du flux d'octets jusqu'à un certain numéro de séquence. Une amélioration de TCP, nommée acquittement sélectif (sélective acknowlegement ou SACK), autorise le destinataire TCP à acquitter des blocs de données reçus dans le désordre.

Grâce aux numéros de séquence et d'acquittement, les systèmes terminaux peuvent remettre les données reçues dans l'ordre à l'application destinataire. Les numéros de séquence sont des nombres entiers non signés sur 32 bits, qui reviennent à zéro après avoir atteint 2^32-1. Le choix du numéro de séquence initial est une des clefs de la robustesse et de la sécurité des connexions TCP.

Une somme de contrôle sur 16 bits, constituée par le complément à un de la somme des compléments à un de tous les éléments d'un segment TCP (en-tête et données), est calculée par l'émetteur, et incluse dans le segment émis. Le destinataire recalcule la somme de contrôle du segment reçu, et si elle correspond à la somme de contrôle reçue, on considère que le segment a été reçu intact et sans erreur.

La somme de contrôle en complément à un utilisée par TCP est relativement peu fiable selon les standards modernes. Ceci restreint l'utilisation de TCP à des réseaux offrant des taux d'erreurs faibles. Si TCP était redéfini aujourd'hui, on utiliserait probablement un CRC sur 32 bits au lieu du mécanisme actuel. Ce manque de fiabilité de la somme de contrôle est partiellement compensé par l'utilisation fréquente d'un CRC ou d'un meilleur contrôle d'intégrité au niveau 2 (couche liaison de données), au-dessous de TCP et IP, comme par exemple dans les trames PPP ou Ethernet. Toutefois, cela ne signifie pas que la somme de contrôle TCP est redondante: des études sur le trafic Internet ont montré qu'on rencontre couramment des erreurs matérielles et logicielles qui introduisent des erreurs dans les paquets entre les nœuds protégés par des CRC, et que le principe de somme de contrôle de bout en bout de TCP détecte la plupart de ces erreurs.

Les acquittements des données émises, ou l'absence d'acquittements, sont utilisés par les émetteurs pour interpréter de façon implicite l'état du réseau entre les systèmes finaux. À l'aide de temporisations, les émetteurs et destinataires TCP peuvent modifier le comportement du flux de données. C'est ce qu'on appelle généralement le contrôle de flux.

TCP utilise un certain nombre de mécanismes afin d'obtenir une bonne robustesse et des performances élevées. Ces mécanismes comprennent l'utilisation d'une fenêtre glissante, l'algorithme de démarrage lent (*slow start*), l'algorithme d'évitement de congestion (*congestion avoidance*), les algorithmes de retransmission rapide (*fast retransmit*) et de récupération rapide (*fast recovery*), etc. Des recherches sont menées actuellement afin d'améliorer TCP pour traiter efficacement les pertes, minimiser les erreurs, gérer la congestion et être rapide dans des environnements très haut débit.

- **Terminaison d'une connexion**

La phase de terminaison d'une connexion utilise une poignée de main en quatre temps, chaque extrémité de la connexion effectuant sa terminaison de manière indépendante. Ainsi, la fin d'une connexion nécessite une paire de segments FIN et ACK pour chaque extrémité.

- **Ports TCP**

TCP utilise la notion de numéro de port pour identifier les applications. À chaque extrémité de la connexion TCP est associé un numéro de port sur 16 bits assigné à l'application émettrice ou réceptrice. Les ports peuvent faire partie de trois catégories de base: les ports bien connus, les ports enregistrés et les ports dynamiques/privés. Les ports bien connus sont assignés par l'IANA (Internet Assigned Numbers Authority) et sont souvent utilisés par des processus système ou ayant des droits privilégiés. Les applications bien connues qui fonctionnent en tant que serveur et sont en attente de connexions utilisent généralement ces types de ports. Exemples: FTP (21), Telnet (23), SMTP (25) et HTTP (80). Les ports enregistrés sont généralement utilisés par des applications utilisateur comme ports sources éphémères pour se connecter à un serveur, mais ils peuvent aussi identifier des services non enregistrés par l'IANA. Les ports dynamiques/privés peuvent aussi être utilisés par des applications utilisateur, mais plus rarement. Ils n'ont pas de sens en dehors d'une connexion TCP particulière.

- **Développement de TCP**

TCP est un protocole assez complexe, et en évolution. Même si des améliorations significatives ont été apportées au cours des années, son fonctionnement de base a peu changé depuis le RFC 793, publié en 1981. Le RFC 1122 (Host Requirements for Internet Hosts), a clarifié un certain nombre de pré-requis pour l'implémentation du protocole TCP. Le RFC 2581 (TCP Congestion Control), l'un des plus importants de ces dernières années, décrit de nouveaux algorithmes utilisés par TCP pour éviter les congestions. En 2

001, le RFC 3168 a été écrit afin de présenter un mécanisme de signalisation des congestions (explicit congestion notification ou ECN), et s'ajoute à la liste des RFCs importants qui complètent la spécification originale. Au début du XXI^{e} siècle, TCP est utilisé approximativement pour 95% de tout le trafic Internet. Les applications les plus courantes qui utilisent TCP sont HTTP/HTTPS (world wide web), SMTP/POP3/IMAP (messagerie) et FTP (transfert de fichiers). Son utilisation très répandue est la preuve de la qualité de la conception réalisée par ses créateurs originaux **[2].**

- **Alternatives à TCP**

Toutefois, TCP n'est pas approprié pour de nombreuses applications, et de nouveaux protocoles de transport sont créés et déployés afin de combler certaines de ses lacunes. Par

exemple, de nombreuses applications temps-réel n'ont pas besoin, et peuvent même souffrir, des mécanismes de transport fiable de TCP. Dans ce type d'applications, il est souvent préférable de gérer les pertes, erreurs ou congestions, plutôt que d'essayer de les éviter. Les

applications de diffusion multimédia (audio et vidéo, etc), ou certains jeux multi-joueurs en temps-réel, par exemple, n'utilisent pas TCP. Toute application qui ne nécessite pas la fiabilité de TCP, ou a un besoin limité en fonctionnalité, peut choisir de ne pas l'utiliser. Dans de nombreux cas, UDP (User datagram protocol) peut être utilisé à la place de TCP lorsque seuls les services de multiplexage applicatifs sont requis.

2-4-2/ User Datagram Protocol

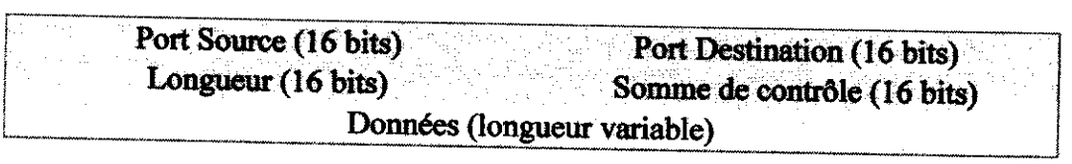
User Datagram Protocol (ou **UDP**, protocole de datagramme utilisateur) est un des principaux protocoles de télécommunication utilisé par Internet. Il fait partie de la couche transport de la pile de protocole TCP/IP : dans l'adaptation approximative de cette dernière au modèle OSI, il appartiendrait la couche 4, comme TCP. Il est détaillé dans la RFC 1662.

Le rôle de ce protocole est de permettre la transmission de paquets (aussi appelés *datagrammes*) de manière très simple entre deux entités, chacune étant définie par une adresse IP et un numéro de port (pour différentier différents utilisateurs sur la même machine). Contrairement au protocole TCP, il travaille en mode non-connecté : il n'y a pas de moyen de vérifier si tous les paquets envoyés sont bien arrivés à destination et ni dans quel ordre. C'est pour cela qu'il est souvent décrit comme étant un protocole non-fiable. Par contre, pour un datagramme UDP donné, l'exactitude du contenu des données est assuré grâce à une somme de contrôle (*checksum*).

- **Structure d'un datagramme UDP**

Le paquet UDP est encapsulé dans un paquet IP. Il comporte un en-tête suivi des données proprement dites à transporter.

L'en-tête (*header* en anglais) d'un datagramme UDP est bien plus simple que celui de TCP**[2]** :



Il contient les 4 champs suivants:

Port Source
: il indique depuis quel port le paquet a été envoyé.

Port de Destination
: il indique à quel port le paquet doit être envoyé

Longueur

il indique la longueur totale du datagramme UDP (en-tête et données). La longueur minimal est donc de 8 octets (taille de l'en-tête)

Somme de contrôle

celle-ci (CRC, Cyclic Redundancy Check) permet de s'assurer de l'intégrité du paquet reçu. Elle est calculée sur l'ensemble de l'en-tête UDP et des données, mais aussi sur un pseudo en-tête (extrait de l'en-tête IP)

Note: la présence de ce pseudo en-tête, interaction entre les deux couches IP et UDP, est une des raisons qui font
que le modèle TCP/IP ne s'applique pas parfaitement au modèle OSI.

- **Utilisation**

Il est utilisé quand il est nécessaire soit de transmettre des données très rapidement, et où la perte d'une partie de ces données n'a pas grande importance, soit de transmettre des petites quantités de données, là où la connexion « 3-WAY » TCP serait trop lourde. Par exemple, dans le cas de la transmission de la voix sur IP, ce n'est pas grave si l'un ou l'autre paquet se perd (il existe des mécanismes de substitution des données manquante), par contre la rapidité de transmission est un critère primordial pour la qualité d'écoute.

Exemples d'utilisation :

- le programme traceroute,
- les protocoles DNS, TFTP,
- les jeux en réseau (exemple : jeux de tir subjectifs)
- le streaming: il est indispensable pour des applications multimédias de par sa faible latence.

2-4-3/ Le protocole ICMP :

- **La gestion des erreurs :**

Le protocole **ICMP** (*Internet Control Message Protocol*) est un protocole qui permet de gérer les informations relatives aux erreurs aux machines connectées. Etant donné le peu de contrôles que le protocole IP réalise, il permet non pas de corriger ces erreurs mais de faire part de ces erreurs aux protocoles des couches voisines. Ainsi, le protocole ICMP est utilisé par tous les routeurs , qui l'utilisent pour signaler une erreur (appelé *Delivery Problem*).

- **Les messages ICMP sont encapsulés :**

Les messages d'erreur ICMP sont transportés sur le réseau sous forme de datagramme, comme n'importe quelle donnée. Ainsi, les messages d'erreur peuvent eux-mêmes être sujet d'erreurs.

Toutefois en cas d'erreur sur un datagramme transportant un message ICMP, aucun message d'erreur n'est délivré pour éviter un effet "boule de neige" en cas d'incident sur le réseau.

Voici à quoi ressemble un message ICMP encapsulé dans un datagramme IP**[2]**:

Message ICMP

En-tête	Type	Code	Checksum	Message
	(8 bits)	(8 bits)	(16 bits)	(taille variable)

Les messages ICMP sont émis en utilisant l'en-tête IP de base. Le premier octet de la section de données du datagramme est le champ de type ICMP. Sa valeur détermine le format du reste des données dans le datagramme ICMP. Tout champ qualifié de "non utilisé" est réservé pour application future et doit être laissé à zéro lors de l'émission, les récepteurs ne devant pas utiliser ces champs (sauf lorsqu'il s'agit de calculer le Checksum). Sauf mention particulière contraire signalée dans chaque description de message spécifique, les valeurs des champs d'en-tête Internet auront la signification suivante**[2]**:

Version :	4
IHL :	Longueur d'en-tête Internet en mots de 32-bits.
Type de Service :	0
Longueur Totale :	Longueur totale du datagramme en octets.
Identification : Bits Contrôles : Fragment Offset :	Utilisés par le mécanisme de fragmentation.
Durée de vie :	Durée de vie du datagramme en secondes; ce champ est diminué d'une unité par chaque module IP traversé dans lesquels le datagramme est traité, la valeur dans ce champ doit être au moins égale au nombre maximum de routeurs que ce datagramme est sensé traverser jusqu'à sa destination finale.
Protocole :	ICMP = 1
Checksum :	Le complément à un sur 16 bits de la somme des compléments à un de l'en-tête Internet pris par mots de 16 bits. Lors du calcul du Checksum, le champ destiné à recevoir ce Checksum sera laissé à zéro. Ce mécanisme de Checksum sera changé dans le futur.
Adresse source :	L'adresse du routeur ou hôte qui compose le message ICMP. Sauf mention contraire, celle-ci peut être toute adresse de routeur.
Adresse destinataire :	L'adresse du routeur ou hôte à qui le message doit être envoyé.

Signification des messages ICMP

Type	Code	Message	Signification du message
8	0	Demande d'ECHO	Ce message est utilisé lorsqu'on utilise la commande PING. Cette commande, permettant de tester le réseau, envoie un datagramme à un destinataire et lui demande de le restituer
3	0	Destinataire inaccessible	Le réseau n'est pas accessible
3	1	Destinataire inaccessible	La machine n'est pas accessible
3	2	Destinataire inaccessible	Le protocole n'est pas accessible
3	3	Destinataire inaccessible	Le port n'est pas accessible
3	4	Destinataire inaccessible	Fragmentation nécessaire mais impossible à cause du drapeau (flag) DF
3	5	Destinataire inaccessible	Le routage a échoué
3	6	Destinataire inaccessible	Réseau inconnu
3	7	Destinataire inaccessible	Machine inconnue
3	8	Destinataire inaccessible	Machine non connectée au réseau (inutilisé)
3	9	Destinataire inaccessible	Communication avec le réseau interdite
3	10	Destinataire inaccessible	Communication avec la machine interdite
3	11	Destinataire inaccessible	Réseau inaccessible pour ce service
3	12	Destinataire inaccessible	Machine inaccessible pour ce service
3	11	Destinataire inaccessible	Communication interdite (filtrage)
4	0	Source Quench	Le volume de données envoyé est trop important, le routeur envoie ce message pour prévenir qu'il sature afin de demander de réduire la vitesse de transmission
5	0	Redirection pour un hôte	Le routeur remarque que la route d'un ordinateur n'est pas optimale et envoie l'adresse du routeur à rajouter dans la table de routage de l'ordinateur
5	1	Redirection pour un hôte et un service donné	Le routeur remarque que la route d'un ordinateur n'est pas optimale pour un service donné et envoie l'adresse du routeur à rajouter dans la table de routage de l'ordinateur

5	2	Redirection pour un réseau	Le routeur remarque que la route d'un réseau entier n'est pas optimale et envoie l'adresse du routeur à rajouter dans la table de routage des ordinateurs du réseau
5	3	Redirection pour un réseau et un service donné	Le routeur remarque que la route d'un réseau entier n'est pas optimale pour un service donné et envoie l'adresse du routeur à rajouter dans la table de routage des ordinateurs du réseau
11	0	Temps dépassé	Ce message est envoyé lorsque le temps de vie d'un datagramme est dépassé. L'en-tête du datagramme est renvoyé pour que l'utilisateur sache quel datagramme a été détruit
11	1	Temps de ré-assemblage de fragment dépassé	Ce message est envoyé lorsque le temps de ré-assemblage des fragments d'un datagramme est dépassé.
12	0	En-tête erroné	Ce message est envoyé lorsqu'un champ d'un en-tête est erroné. La position de l'erreur est retournée
13	0	Timestamp request	Une machine demande à une autre son heure et sa date système (universelle)
14	0	Timestamp reply	La machine réceptrice donne son heure et sa date système afin que la machine émettrice puisse déterminer le temps de transfert des données
15	0	Demande d'adresse réseau	Ce message permet de demander au réseau une adresse IP
16	0	Réponse d'adresse réseau	Ce message répond au message précédent
17	0	Demande de masque de sous-réseau	Ce message permet de demander au réseau un masque de sous-réseau
18	0	Réponse de masque de sous-réseau	Ce message répond au message précédent
17	0	Timestamp reply	La machine réceptrice donne son heure et sa date système afin que la machine émettrice puisse déterminer le temps de transfert des données

- **Pourquoi s'intéresse au protocole ICMP?**

-Le protocole ICMP peut facilement être exploité à des fins malveillants par un pirate , on a recensé les cas suivants :

❖ **Ignorer certains messages ICMP :**

- ICMP Redirect :

Le but des messages ICMP Redirect est d'indiquer à une machine ou à un routeur qu'il y a un chemin plus court pour joindre une destination donnée.

Ils sont en général utilisés dans les réseaux d'interconnexion de routeurs et non au sein de réseaux locaux.

Comme on le voit sur la figure1, la passerelle par défaut de la machine Poste1 pointe sur le routeur A, qui fait suivre tous les paquets au routeur B afin de sortir vers Internet.
Quand le routeur B reçoit les paquets , il constate qu'il est sur le même réseau que la machine Poste1 .
Il lui envoie donc un paquet ICMP Redirect pour lui indiquer d'envoyer ses paquets directement à routeur B, ce qui est un chemin plus court que de passer par routeurA.

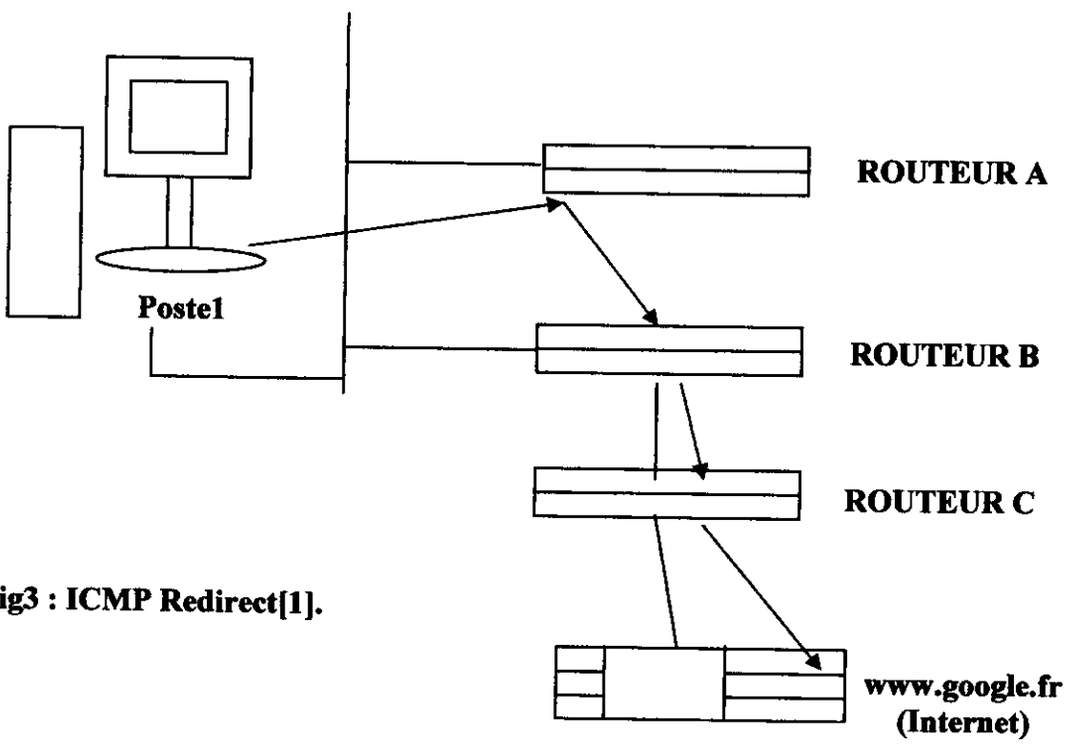


Fig3 : ICMP Redirect[1].

DANGER : ICMP Redirect et Man in Middle attack

Supposons qu'un pirate veux sniffer les paquets échanges pendant une connexion entre un client et sa banque (www.maBanque.com) . Il lui suffit d'envoyer un message ICMP Redirect indiquant au client ou a son routeur d'entrée que le plus court chemin pour joindre www.maBanque.com consiste a passer par sa machine .Si les machines du réseau acceptent les ICMP Redirect , tous les paquets destines a www.maBanque.com vont donc être détournes par la machine pirate. Ce dernier va ensuite relayer ces paquets vers leur
destination , en ayant au passage la possibilité d'en visualiser le contenu ou même de les modifier ! Ce type d'attaque assez générique est qualifié de MiM, de l'anglais Man(Monkey ?) in the Middle Attack.

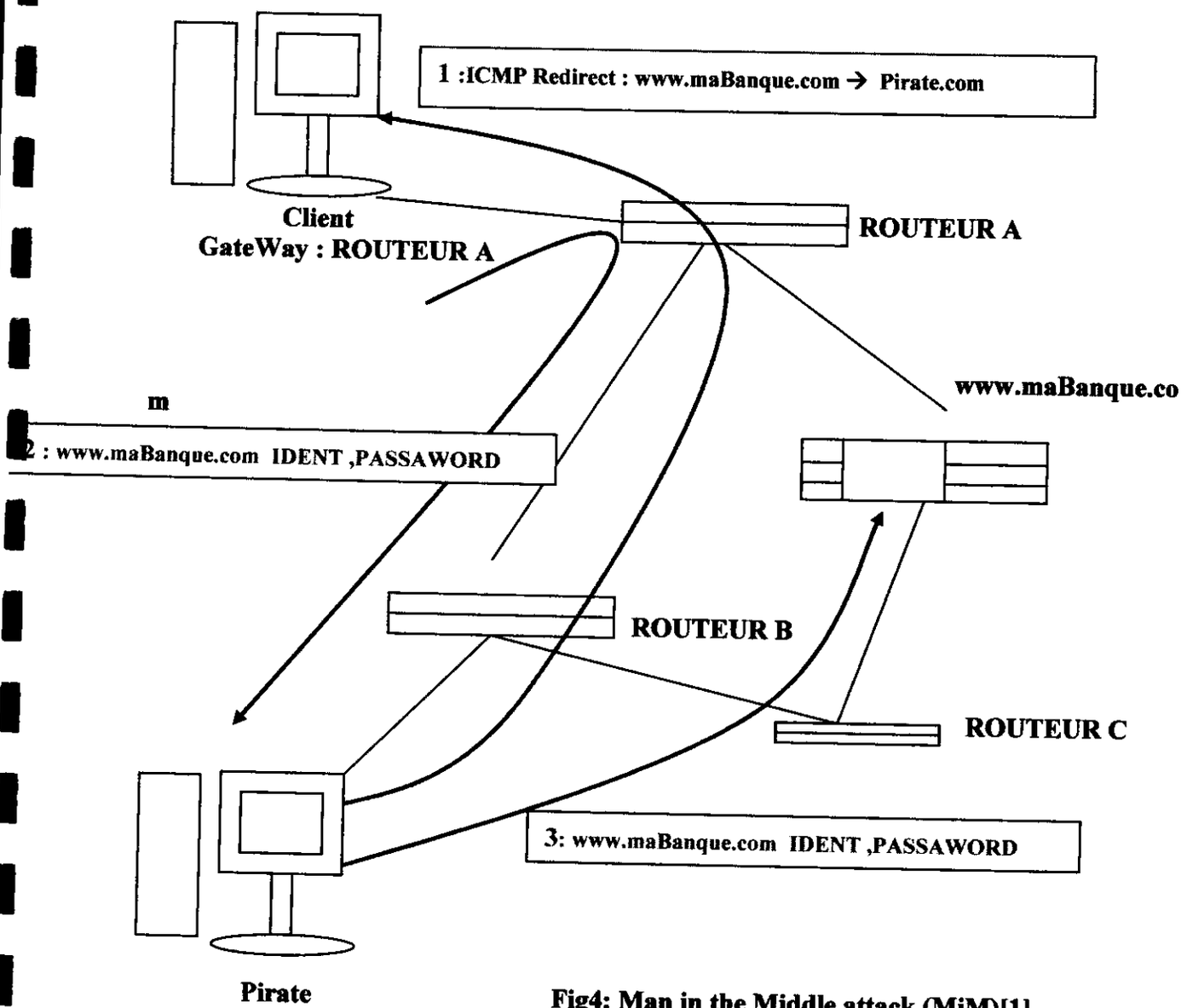


Fig4: Man in the Middle attack (MiM)[1]

- ICMP Echo Request :

Les paquets **ICMP request** envoyé par la commande **ping** peuvent dans certains cas être utilise pour scanner un réseau ou pour provoquer un déni de service , Il est possible de désactiver la prise en compte de ces requêtes .

2-4-4/ Domain Name System

Le **Domain Name System** (ou **DNS**, système de noms de domaine) est un système permettant d'établir une correspondance entre une adresse IP et un nom de domaine et, plus généralement, de trouver une information à partir d'un nom de domaine**[2]**.

- **Associer une adresse IP et un nom de domaine**

Les ordinateurs connectés à un réseau IP, par exemple Internet, possèdent tous une adresse IP. Ces adresses sont numériques afin d'être plus facilement traitées par une machine . Selon IPv4, elles prennent la forme *xxx.yyy.zzz.aaa*, où xxx, yyy, zzz et aaa sont quatre nombres variant entre 0 et 255 (en système décimal) **[2]**.

Il n'est évidemment pas simple pour un humain de retenir ce numéro lorsque l'on désire accéder à un ordinateur d'Internet. C'est pourquoi le DNS fut inventé en 1983 par Paul Mockapetris. Il permet d'associer à une adresse IP, un nom intelligible, humainement plus simple à retenir, appelé nom de domaine. *fr.wikipedia.org*, par exemple, est composé du domaine générique *org*, du domaine déposé *wikipedia* et du nom d'hôte *fr* **[2]**.

Quand un utilisateur souhaite accéder à un site, comme par exemple *www.free.fr*, son ordinateur émet une requête spéciale à un serveur DNS, demandant 'Quelle est l'adresse de www.free.fr ?'.Le serveur répond en retournant l'adresse IP du serveur, qui est dans ce cas-ci, 213.228.0.42.

Il est également possible de poser la question inverse, à savoir 'Quel est le nom de domaine de telle adresse IP ?'. On parle alors de **résolution inverse.**

Plusieurs noms de domaine peuvent pointer vers une même adresse IP. Mais une adresse IP ne peut pointer que sur un unique nom de domaine.

Lorsqu'un service génère un trafic important, celui-ci peut faire appel à la technique du *DNS Round-Robin*, qui consiste à associer plusieurs adresses IP à un nom de domaine. Les différentes versions de Wikipedia, comme *fr.wikipedia.org* par exemple, sont associées à plusieurs adresses IP : 207.142.131.247, 207.142.131.248, 207.142.131.235, 207.142.131.236, 207.142.131.245 et 207.142.131.246. Une rotation circulaire entre ces différentes adresses permet ainsi de répartir la charge générée par ce trafic important, entre les différentes machines, ayant ces adresses IP.

- **Un système réparti**

Il existe en fait des centaines de milliers de serveurs DNS dans le monde entier. Chaque serveur DNS n'a en réalité à sa disposition qu'un ensemble d'informations restreint. Quand notre serveur DNS (par exemple, celui de notre fournisseur d'accès à Internet) doit trouver l'adresse IP de *fr.wikipedia.org*, une certaine communication s'instaure alors avec d'autres serveurs DNS. Tout d'abord, notre serveur demande à des serveurs DNS peu nombreux appelés *root-servers* quels serveurs peuvent lui répondre pour la zone *org*. Parmi ceux-ci, notre serveur va en choisir un pour savoir quel serveur est capable de lui répondre pour la zone *wikipedia.org*. C'est ce dernier qui pourra lui donner l'adresse IP de *fr.wikipedia.org* **[2]**.

Cependant, ce processus de résolution de nom est long. C'est pourquoi, la plupart des serveurs DNS (et notamment ceux des Fournisseurs d'accès à Internet) font aussi office de *DNS cache* : ils gardent en mémoire la réponse d'une résolution de nom afin de ne pas effectuer ce long processus à nouveau ultérieurement**[2]**.

Un nom de domaine peut utiliser plusieurs serveurs DNS. Généralement, les noms de domaines en utilisent deux : un primaire et un secondaire. Seuls ces derniers peuvent fournir une réponse valable à tout instant. On parle de réponse faisant autorité (authoritative answer en anglais). Les serveurs des fournisseurs d'accès à Internet quant à eux fournissent des réponses qui ne sont pas nécessairement à jour, à cause du cache mis en place. On parle alors de réponse ne faisant pas autorité (non authoritative answer en anglais) **[2]**.

Pour trouver le nom de domaine d'une IP, on utilise le même principe. Dans un nom de domaine, la partie la plus générale est à droite: org dans fr.wikipedia.org. Dans une adresse ip, c'est le contraire: 213 est la partie la plus générale de 213.228.0.42. Pour conserver une logique cohérente, on inverse l'ordre des quatre termes de l'adresse et on la concatène au pseudo domaine *in-addr.arpa.*. Ainsi, par exemple, pour trouver le nom de domaine de l'adresse IP 213.228.0.42, on résout 42.0.228.213.in-addr.arpa, qui est un pointeur vers www1.free.fr**[2]**.

Cette architecture garantit au réseau Internet une certaine sécurité. Quand un serveur DNS tombe, le bon fonctionnement de la résolution de nom n'est pas remise en cause. De plus, elle permet de mettre à jour l'adresse IP associée à un nom de domaine dans le monde entier facilement et assez rapidement (un délai de 48 heures est généralement suffisant).

- **Principaux enregistrements DNS**

Les principaux enregistrements définis par un DNS sont**[2]** :

- **A record** ou **address record** qui fait correspondre un nom d'hôte à une adresse IPv4 de 32 bits.
- **AAAA record** ou **IPv6 address record** qui fait correspondre un nom d'hôte à une adresse IPv6 de 128 bits.
- **CNAME record** ou **canonical name record** qui permet de faire d'un domaine un alias vers un autre. Cet alias hérite de tous les sous-domaines de l'original.
- **MX record** ou **mail exchange record** qui définit les serveurs de mail pour ce domaine.
- **PTR record** ou **pointer record** qui définit un nom domaine pour une adresse IP.
- **NS record** ou **name server record** qui définit les serveurs DNS de ce domaine.
- **SOA record** ou **start of authority record** qui définit le serveur DNS fournissant l'information faisant autorité sur un domaine Internet.
- **SRV record** qui généralise la notion de **MX record**, standardisé dans la RFC 2782.
- **NAPTR record** qui donne accès à des règles de réécriture de l'information, permettant des correspondances assez lâches entre un nom de domaine et une ressource. Il est spécifié dans la RFC 3403.
- **TXT record** permet à un administrateur d'insérer un texte quelconque dans un enregistrement DNS. Par exemple, cet enregistrement était utilisé pour implémenter la spécification Sender Policy Framework.

D'autres types d'enregistrements servent simplement à donner des informations (par exemple, un **LOC record** - rarissime - indique l'emplacement physique d'un hôte).

- **Sécurité du DNS**

Comme beaucoup de protocoles Internet, le DNS a été conçu sans se préoccuper de la sécurité. Il ne faut donc pas se fier au DNS pour arriver sur le bon serveur et c'est pour cela que des protocoles comme SSH font leur propre vérification (via la cryptographie). Les principales failles du DNS (décrites dans le RFC 3833) sont :

- Interception du paquet (requête ou réponse) et émission d'un autre paquet à sa place,
- Fabrication d'une réponse (les serveurs DNS acceptent trop facilement des réponses puisque seul un numéro de requête, très petit, sert d'authentification),
- Trahison par un serveur (le secondaire hors-site d'un domaine peut par exemple passer sous le contrôle de personnes malintentionnées), et le traditionnel déni de service.
    - **DNSSEC**

Pour contrer ces vulnérabilités, le protocole DNSSEC a été développé.

2-5/ Introduction à la notion de firewall :

Chaque ordinateur connecté à Internet (et d'une manière plus générale à n'importe quel réseau informatique) est susceptible d'être victime d'une attaque d'un pirate informatique. La méthodologie généralement employée par les pirates informatiques consiste à scruter le réseau (en envoyant des paquets de données de manière aléatoire) à la recherche d'une machine connectée, puis à chercher une faille de sécurité afin de l'exploiter et d'accéder aux données s'y trouvant.

Cette menace est d'autant plus grande que la machine est connectée en permanence à Internet pour plusieurs raisons :

- La machine cible est susceptible d'être connectée sans pour autant être surveillée ;
- La machine cible est généralement connectée avec une plus large bande passante ;
- La machine cible ne change pas (ou peu) d'adresse IP.

Ainsi, il est nécessaire, autant pour les réseaux d'entreprises que pour les internautes possédant une connexion de type câble ou ADSL, de se protéger des intrusions réseaux en installant un dispositif de protection.

2-6/ Qu'est-ce qu'un pare-feu ?

Un **pare-feu** (appelé aussi *coupe-feu, garde-barrière* ou **firewall** en anglais), est un système permettant de protéger un ordinateur ou un réseau d'ordinateurs des intrusions provenant d'un réseau tiers (notamment Internet) **[3]**. Le pare-feu est un système permettant de filtrer les paquets de données échangés avec le réseau, il s'agit ainsi d'une passerelle filtrante comportant au minimum les interfaces réseau suivante :

- une interface pour le réseau à protéger (réseau interne)
- une interface pour le réseau externe

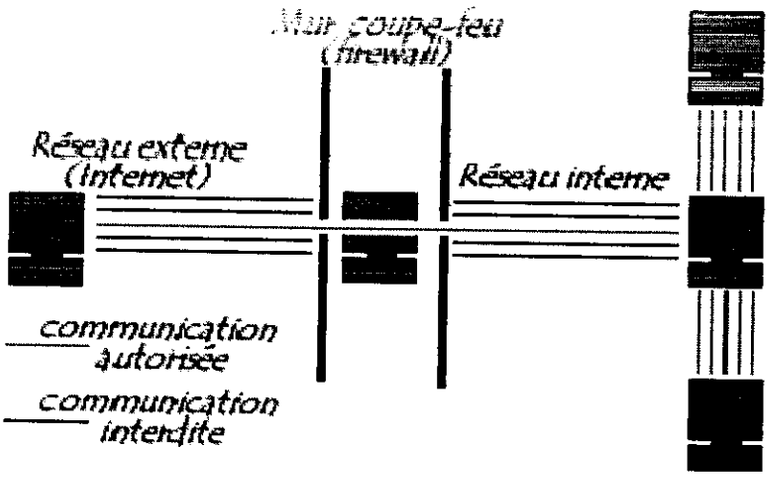


***Fig5 : Le Firewall dans un réseau*[3]**

Le système firewall est un système logiciel, reposant parfois sur un matériel réseau dédié, constituant un intermédiaire entre le réseau local (ou la machine locale) et un ou plusieurs réseaux externes**[3]**. Il est possible de mettre un système pare-feu sur n'importe quelle machine et avec n'importe quel système pourvu que :

- La machine soit suffisamment puissante pour traiter le traffic .
- Le système soit sécurisé .
- Aucun autre service que le service de filtrage de paquets ne fonctionne sur le serveur.

Dans le cas où le système pare-feu est fourni dans une boîte noire « clé en main », on utilise le terme d'« appliance ».

2-7/ Fonctionnement d'un système pare-feu

Un système pare-feu contient un ensemble de règles prédéfinies permettant :

- D'autoriser la connexion (*allow*)
- De bloquer la connexion (*deny*)
- De rejeter la demande de connexion sans avertir l'émetteur (*drop*).

L'ensemble de ces règles permet de mettre en oeuvre une méthode de filtrage dépendant de la **politique de sécurité** adoptée par l'entité**[3]**. On distingue habituellement deux types de politiques de sécurité permettant :

- soit d'autoriser uniquement les communications ayant été explicitement autorisées : "Tout ce qui n'est pas explicitement autorisé est interdit".
- soit d'empêcher les échanges qui ont été explicitement interdits.

La première méthode est sans nul doute la plus sûre, mais elle impose toutefois une définition précise et contraignante des besoins en communication**[3]**.

- **Le filtrage simple de paquets :**

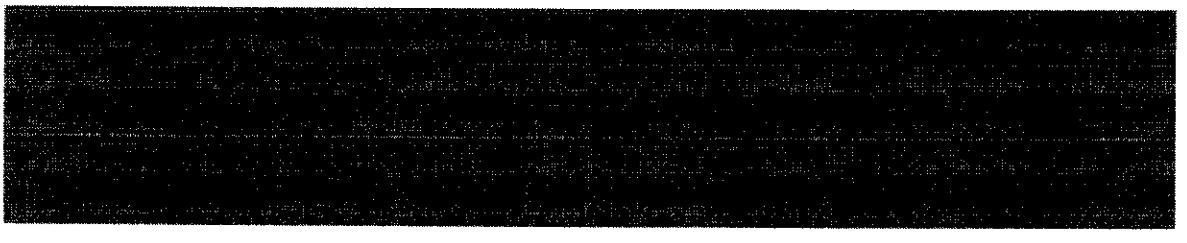
Un système pare-feu fonctionne sur le principe du filtrage simple de paquets (en anglais « *stateless packet filtering* »). Il analyse les en-têtes de chaque paquet de données (*datagramme*) échangé entre une machine du réseau interne et une machine extérieure**[3]**.

Ainsi, les paquets de données échangée entre une machine du réseau extérieur et une machine du réseau interne transitent par le pare-feu et possèdent les en-têtes suivants, systématiquement analysés par le firewall :

- adresse IP de la machine émettrice .
- adresse IP de la machine réceptrice .
- type de paquet (TCP, UDP, etc.).
- numéro de port (rappel: un port est un numéro associé à un service ou une application réseau).

Les adresses IP contenues dans les paquets permettent d'identifier la machine émettrice et la machine cible, tandis que le type de paquet et le numéro de port donnent une indication sur le type de service utilisé**[3]**.

Le tableau ci-dessous donne des exemples de règles de pare-feu**[3]** :



Les ports reconnus (dont le numéro est compris entre 0 et 1023) sont associés à des services courants (les ports 25 et 110 sont par exemple associés au courrier électronique, et le port 80 au Web). La plupart des dispositifs pare-feu sont au minimum configurés de manière à filtrer les communications selon le port utilisé. Il est généralement conseillé de bloquer tous les ports qui ne sont pas indispensables (selon la politique de sécurité retenue) **[3]**.

Le port 23 est par exemple souvent bloqué par défaut par les dispositifs pare-feu car il correspond au protocole Telnet, permettant d'émuler un accès par terminal à une machine distante de manière à pouvoir exécuter des commandes à distance. Les données échangées par Telnet ne sont pas chiffrées, ce qui signifie qu'un individu est susceptible d'écouter le réseau et de voler les éventuels mots de passe circulant en clair. Les administrateurs lui préfèrent généralement le protocole SSH, réputé sûr et fournissant les mêmes fonctionnalités que Telnet**[3]**.

- **Le filtrage dynamique :**

Le filtrage simple de paquets ne s'attache qu'à examiner les paquets IP indépendamment les uns des autres, ce qui correspond au niveau 3 du modèle OSI. Or, la plupart des connexions reposent sur le protocole TCP, qui gère la notion de session, afin d'assurer le bon déroulement des échanges. D'autre part, de nombreux services (le FTP par exemple) initient une connexion sur un port statique, mais ouvrent dynamiquement (c'est-à-dire de manière aléatoire) un port afin d'établir une session entre la machine faisant office de serveur et la machine cliente[3].

Ainsi, il est impossible avec un filtrage simple de paquets de prévoir les ports à laisser passer ou à interdire. Pour y remédier, le système de **filtrage dynamique de paquets** est basé sur l'inspection des couches 3 et 4 du modèle OSI, permettant d'effectuer un suivi des transactions entre le client et le serveur. Le terme anglo-saxon est « **stateful inspection** » ou « *stateful packet filtering* », traduisez « *filtrage de paquets avec état* » [3].

Un dispositif pare-feu de type « stateful inspection » est ainsi capable d'assurer un suivi des échanges, c'est-à-dire de tenir compte de l'état des anciens paquets pour appliquer les règles de filtrage. De cette manière, à partir du moment où une machine autorisée initie une connexion à une machine situé de l'autre côté du pare-feu; l'ensemble des paquets transitant dans le cadre de cette connexion seront implicitement acceptés par le pare-feu[3].

Si le filtrage dynamique est plus performant que le filtrage de paquets basique, il ne protège pas pour autant de l'exploitation des failles applicatives, liées aux vulnérabilités des applications. Or ces vulnérabilités représentent la part la plus importante des risques en terme de sécurité[3].

- **Le filtrage applicatif :**

Le filtrage applicatif permet de filtrer les communications application par application. Le filtrage applicatif opère donc au niveau 7 (couche application) du modèle OSI, contrairement au filtrage de paquets simple (niveau 4). Le filtrage applicatif suppose donc une connaissance des protocoles utilisés par chaque application[3].

Le filtrage applicatif permet, comme son nom l'indique, de filtrer les communications application par application. Le filtrage applicatif suppose donc une bonne connaissance des applications présentes sur le réseau, et notamment de la manière dont elle structure les données échangées (ports, etc.) [3].

Un firewall effectuant un filtrage applicatif est appelé généralement « passerelle applicative » (ou « proxy »), car il sert de relais entre deux réseaux en s'interposant et en effectuant une validation fine du contenu des paquets échangés. Le proxy représente donc un intermédiaire entre les machines du réseau interne et le réseau externe, subissant les attaques à leur place. De plus, le filtrage applicatif permet la destruction des en-têtes précédant le message applicatif, ce qui permet de fournir un niveau de sécurité supplémentaire[3].

Il s'agit d'un dispositif performant, assurant une bonne protection du réseau, pour peu qu'il soit correctement administré. En contrepartie, une analyse fine des données applicatives requiert

une grande puissance de calcul et se traduit donc souvent par un ralentissement des communications, chaque paquet devant être finement analysé[3].

Par ailleurs, le proxy doit nécessairement être en mesure d'interpréter une vaste gamme de protocoles et de connaître les failles afférentes pour être efficace[3].

Enfin, un tel système peut potentiellement comporter une vulnérabilité dans la mesure où il interprète les requêtes qui transitent par son biais. Ainsi, il est recommandé de dissocier le pare-feu (dynamique ou non) du proxy, afin de limiter les risques de compromission[3].

2-8/ Notion de pare-feu personnel :

Dans le cas où la zone protégée se limite à l'ordinateur sur lequel le firewall est installé on parle de **firewall personnel** (*pare-feu personnel*) [3].

Ainsi, un firewall personnel permet de contrôler l'accès au réseau des applications installées sur la machine, et notamment empêcher les attaques du type cheval de Troie, c'est-à-dire des programmes nuisibles ouvrant une brèche dans le système afin de permettre une prise en main à distance de la machine par un pirate informatique. Le firewall personnel permet en effet de repérer et d'empêcher l'ouverture non sollicitée de la part d'applications non autorisées à se connecter.

2-9/ Les limites des firewalls :

Un système pare-feu n'offre bien évidemment pas une sécurité absolue, bien au contraire. Les firewalls n'offrent une protection que dans la mesure où l'ensemble des communications vers l'extérieur passe systématiquement par leur intermédiaire et qu'ils sont correctement configurés. Ainsi, les accès au réseau extérieur par contournement du firewall sont autant de failles de sécurité. C'est notamment le cas des connexions effectuées à partir du réseau interne à l'aide d'un modem ou de tout moyen de connexion échappant au contrôle du pare-feu.

De la même manière, l'introduction de supports de stockage provenant de l'extérieur sur des machines internes au réseau ou bien d'ordinateurs portables peut porter fortement préjudice à la politique de sécurité globale.

Enfin, afin de garantir un niveau de protection maximal, il est nécessaire d'administrer le pare-feu et notamment de surveiller son journal d'activité afin d'être en mesure de détecter les tentatives d'intrusion et les anomalies. Par ailleurs, il est recommandé d'effectuer une veille de sécurité (en s'abonnant aux alertes de sécurité des CERT par exemple) afin de modifier le paramétrage de son dispositif en fonction de la publication des alertes.

La mise en place d'un firewall doit donc se faire en accord avec une véritable politique de sécurité.

2-10/ Inspiration :

Avant de commencer le développement de l'application on s'est inspiré de quelques firewall qui existent sur le centre d'accueil dont le plus utilisé était le firewall commercial « OutpostProInstall ».

On a fait des essais sur ce firewall et on a vu l'ensemble des options qu'offre ce dernier au réseau de la CNAS.

2-10-1 / Firewall commerciale (Outpost) :

Ses performances sont tout à fait à la hauteur et il propose un système de "plugins" (modules additionnels) absolument remarquable. En effet, *Outpost* peut, grâce à eux, bloquer les pubs, les popups, les cookies (à la demande), les referers (informations fournies par votre navigateur lorsque vous vous connectez à un site web), etc. Il dispose même d'un "cache DNS" destinée à accélérer la navigation. Certains voient déjà en lui le successeur d'*Atguard* . Sans compter qu'il est moins gourmand en ressources que *ZoneAlarm[13].*

En plus des traditionnels bloqueurs de pièces jointes pour les mails et contrôle des "éléments actifs" des pages web (java, ActiveX, etc.), citons parmi les éléments utiles de cette version : une configuration "automatique" à l'installation des réseaux locaux (plus besoin de taper les adresses IP des autres machines !), un nouveau composant "anti-fuites" permettant un meilleur contrôle des tentatives de connexion des logiciels installés, un dispositif de protection au démarrage de *Windows* qui empêche le lancement d'application suspectes, un meilleur filtrage des paquets (unités de transmission des données sur le réseau), des logs de connexion plus détaillés, un bloqueur de fenêtres popups, en plus d'une meilleure compatibilité avec le SP2 de *WinXP*, *Agnitum* annonce pas mal de nouveautés intéressantes : amélioration du plug-in de détection d'attaque (choix sélectif des ports, liste des ports et protocoles de confiance), assistant avancé de configuration, contrôle de processus ouvert : protection contre l'occupation illégitime de l'espace mémoire par des codes malveillants (tests de vulnérabilité Copycat, Thermite), contrôle des processus cachés, protection par mot de passe pour éviter l'arrêt du service Outpost par des Troyens et autres virus, capacité à télécharger une liste de mots-clés pour le plug-in anti-publicités, blocage sélectif du referrer pour le plug-in de contenu actif, filtrage des paquets localhost, filtrage des paquets de transit pour améliorer la performances des machines routeurs, contrôle des paquets rawsocket, blocage des requêtes DNS malformées, amélioration des réglages de contrôle de composant (fenêtre distincte permettant d'affiner plus facilement ces réglages), accélération du traitement des paquets (IGMP/ICMP), configuration automatique des règles de domaine, support de règles pour le protocole ICMP type 10, etc*[13].*

- **Les inconvénients :**

- Il pose parfois des problèmes sur certaines configurations par exemple il faut attendre d'avoir lancé la connexion câble avant de lancer *Outpost[13].*
- *Outpost* n'aime pas du tout les autres firewalls (ex : *ZoneAlarm*). Il vous faudra généralement les désinstaller pour que le logiciel fonctionne*[13].* De plus, il est pratiquement incompatible avec les réseaux locaux (ex : connexion à Internet partagée par plusieurs machines) *[13].*

❖ **Les avantages :**

Sécurité
- Détection et blocage de toutes intrusions par des pirates*[13]* .
- Blocage de toutes tentatives de vol de données .
- Navigation en mode furtif (PC invisible pour les intrus) *[13]*.
- Analyse de votre courrier électronique *[13]*.
- Mise en Quarantaine de tous fichiers suspects*[13]*.

Contrôle
- Surveillance de l'activité réseau de votre PC .
- Protection des enfants contre des sites illégaux et non appropriés*[13]* .
- Accès complet à l'historique de vos connexions*[13]*.

Vie Privée
- Empêche toutes fuites de données depuis votre PC*[13]*.
- Evite les violations de votre vie privée au travers d'Internet*[13]* .
- Cache vos habitudes Internet lorsque vous naviguez sur le Web *[13]*.

Facilité d'utilisation
- Compatible avec tous les systèmes Windows*[13]* .
- Compatible avec tous les types de connexion Internet*[13]*.
- Auto configuration pour une meilleure protection pendant l'installation *[13]*.
- Mises à jour fréquentes des informations*[13]*.
  (protection contre les nouvelles attaques)

- Nous avons étudié aussi le leader des firewall sous Linux , le firewall NetFilter/IPtables.

2-10-2/ Netfilter/Iptables

IPtables et Netfilter fournissent les fonctionnalités de filtrage disponibles sous Linux du noyau 2.4.

NetFilter est implémenté au niveau des couches réseau du noyau Linux. Il effectue le filtrage proprement dit, tandis que IPtables fournit les commandes nécessaires à la programmation des filtres. IPtables est le successeur d'IPchaines.

- **Fonctionnalités d'IPtables :**

IPtables fournit trois types de fonctionnalités :

❖ Filtrage : statique(stateless) ou dynamique(stateful) et connection tracking.

- NAT : traduction d'adresse IP, Source NAT, Masquerade, Destination NAT, redirection de port(il faut noter que IPtables ne permet pas à ce jour d'effectuer du NAT statique-un pour un- de façon simple).

- Marquage et manipulation de paquets.

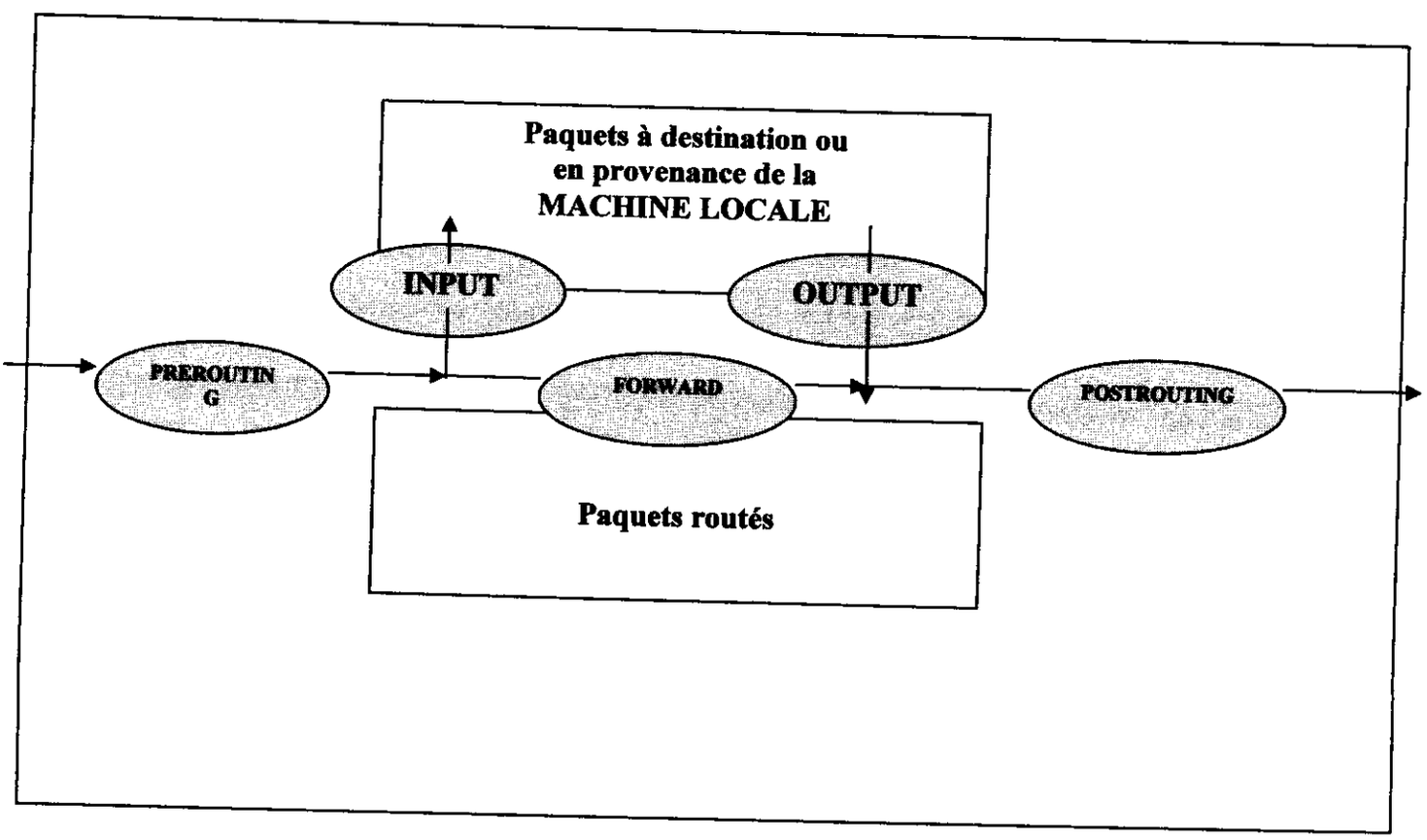
Parmi ces trois fonctionnalités, les deux premières ont été tester.

- **Tables et chaînes**

A chacune des trois fonctions d'IPtables correspond une table qui sert à programmer cette fonction.

- FILTER : pour les règles de filtrage.
- NAT : pour les règles de traduction d'adresses.
- MANGLE : pour la manipulation et le marquage de paquets.

Chaque table contient un certain nombre de chaînes. Ces chaînes contiennent à leur tour une série de règles. La chaîne et les règles qu'elle contient s'appliquent à un moment précis (avant le routage après le routage…) du parcours du paquet(Fig 5). Elle détermine le futur du paquet(transmis, intercepté ….).



- ❖ PREROUTING : le paquet se présente sur une interface de la machine .Les chaînes appliquées sur PREROUTING seront exécutées avant d'analyser l'adresse IP du paquet pour prendre la décision de routage. Le cas d'utilisation typique est le DNAT.

- ❖ POSTROUTING : les chaînes appliquées sur POSTROUTING seront exécutées juste avant d'envoyer le paquet sur l'interface de sortie, alors que le paquet est déjà routé .Le cas d'utilisation typique est le SNAT.

- ❖ FORWARD : le paquet n'est pas destiné à la machine locale , mais il doit être relayé sur une autre interface. Les chaînes appliquées sur FORWARD ne concerneront pas les paquets à destination ou venant de la machine locale.

- ❖ INPUT : les chaînes appliquées sur INPUT concerneront tout paquet à destination de la machine locale.

- ❖ OUTPUT : les chaînes appliquées sur OUTPUT concerneront tout paquet en provenance de la machine locale.

- **Ecriture des règles *[1]***

Iptables	-t filter	-A Input	-p TCP	–s 192.168.153.1	-j drop
	Table	**Chaîne**		**Sélection**	**Action**

Table : NAT | mangle | filter.
Chaîne : prerouting | postrouting | input | output | forward.
Selection : -p protocol, -s source.
Action : -j drop | ACCEPT | REJECT | QUEUE | LOG | MARK | TOS | MIRROR | SNAT | DNAT | MASQUERADE | REDIRECT.

-j Accept	Le paquet est accepté.
-j DROP	Le paquet est rejeté.
-j REJECT	Le paquet est rejeté, l'expéditeur est averti de l'indisponibilité du service.
-j QUEUE	Le paquet est envoyé a une application
-j Log	Le paquet est envoyé au système « syslog ».
-j Mark	Le paquet est marqué
-j TOS	Modifie le « type Of service » du paquet
-j MIRROR	Renvoie le paquet à l'expéditeur.
-j SNAT	L'adresse source du paquet est translatée.
-j DNAT	L'adresse destination du paquet est translatée.
-j MASQUERADE	L'adresse source du paquet est translatée.
-j REDIRECT	R-direction d'un port vers un autre.

- D'autre firewall open source sous Windows tel que : **Net Centurion** ont été l'objet de notre étude.

2-10-3/ Net Centurion

Parmi les services qu'offre ce firewall open source :

- Il utilise le filtrage de paquets TCP/IP.
- Il gère plusieurs interfaces réseaux.
- Améliore la sécurité de votre system.
- Utilise le system de log.
- Les fichiers log sont bien détaillés.

CHAPITRE 3
CONCEPTION D'UNE SOLUTION PARE-FEU PERSONNALISEE

3 – Conception d'une solution pare-feu personnalisée

3-1/ Description de l'application

L'application consiste à réaliser un firewall statique sous la plate forme Windows.

L'application utilise le driver de filtrage IPv4 qui se trouve dans les systèmes d'exploitations Windows XP/Windows Serveur 2000/Serveur 2003 /Serveur Longhorn.

Ce driver est configuré par l'intermédiaire du service RRAS
(Routing and Remote Access).

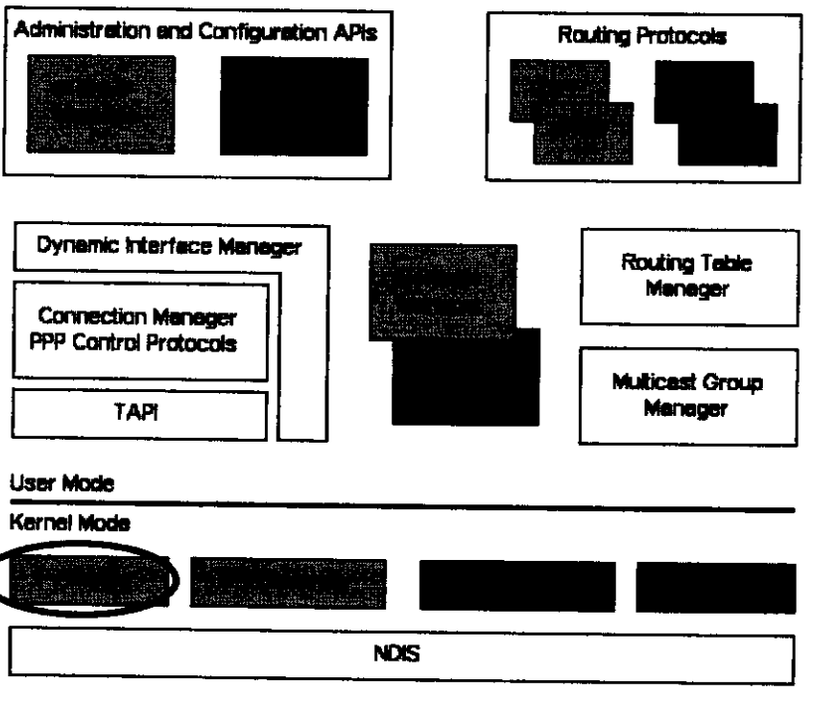


Fig6: Architecture du service RRAS sous Windows 2000[5].

Le driver de filtrage IPv4.

Routing and Remote Access Services. Remplaçant du RAS de Windows, permettant à un tel serveur de faire du routage, donc de devenir un routeur à part entière.
Les API du service RRAS donnent la possibilité de créer des applications pour administrer le service de routage et d'accès a distance.

Le driver de filtrage IPV4 fonctionne au mode noyau (Kernel mode).

NDIS est l'acronyme de Network Driver Interface Spécification, le but de NDIS est de définir des APIS standard pour les cartes réseaux.

3-2/Algorithme de filtrage de paquets :

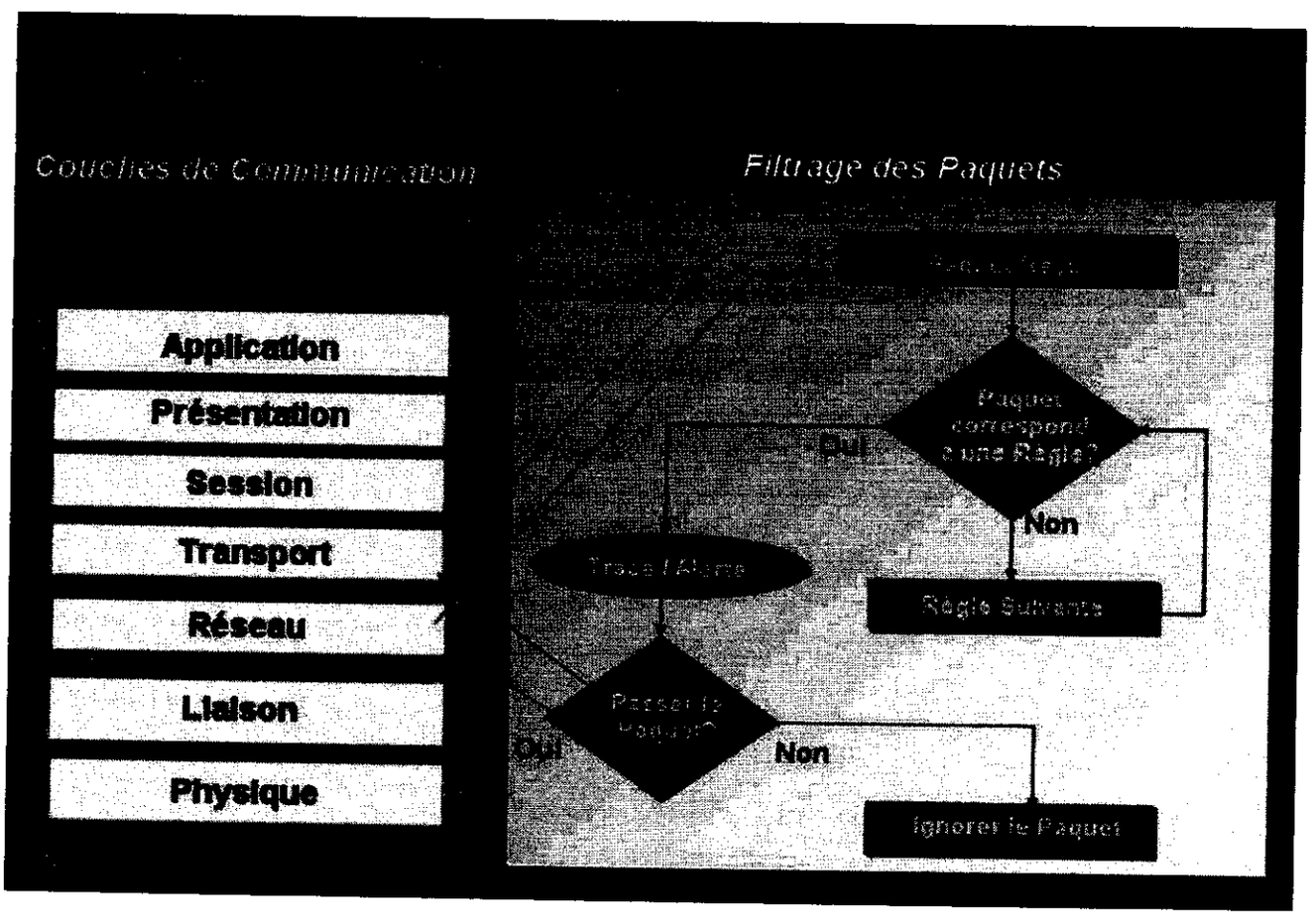


Fig7: Algorithme de filtrage de paquets.

- Une fois qu'on a reçus les paquets de la couche réseau ou bien de la couche transport.

- On teste si le paquet correspond à une règle déjà établie par l'administrateur.

- Si oui, une alerte est écrite dans le fichier journal.

- On distingue deux types de politiques de filtrages :

  - Tout est fermé sauf ce qui est mentionné par l'administrateur sera autorisé.
  - Tout est ouvert sauf ce qui est spécifié sera bloqué (c'est la politique que nous avons adapte).

  La première politique impose toutefois une définition précise et contraignante des besoins en communication.

Selon la politique choisis le paquet peut être ignoré ou bien autoriser a passé.

3-3/ Conception de l'application :

3-3-1/ Diagrammes UML :

- **Modéliser les vues statiques de l'application :**
  - ❖ **Diagramme de cas d'utilisation**

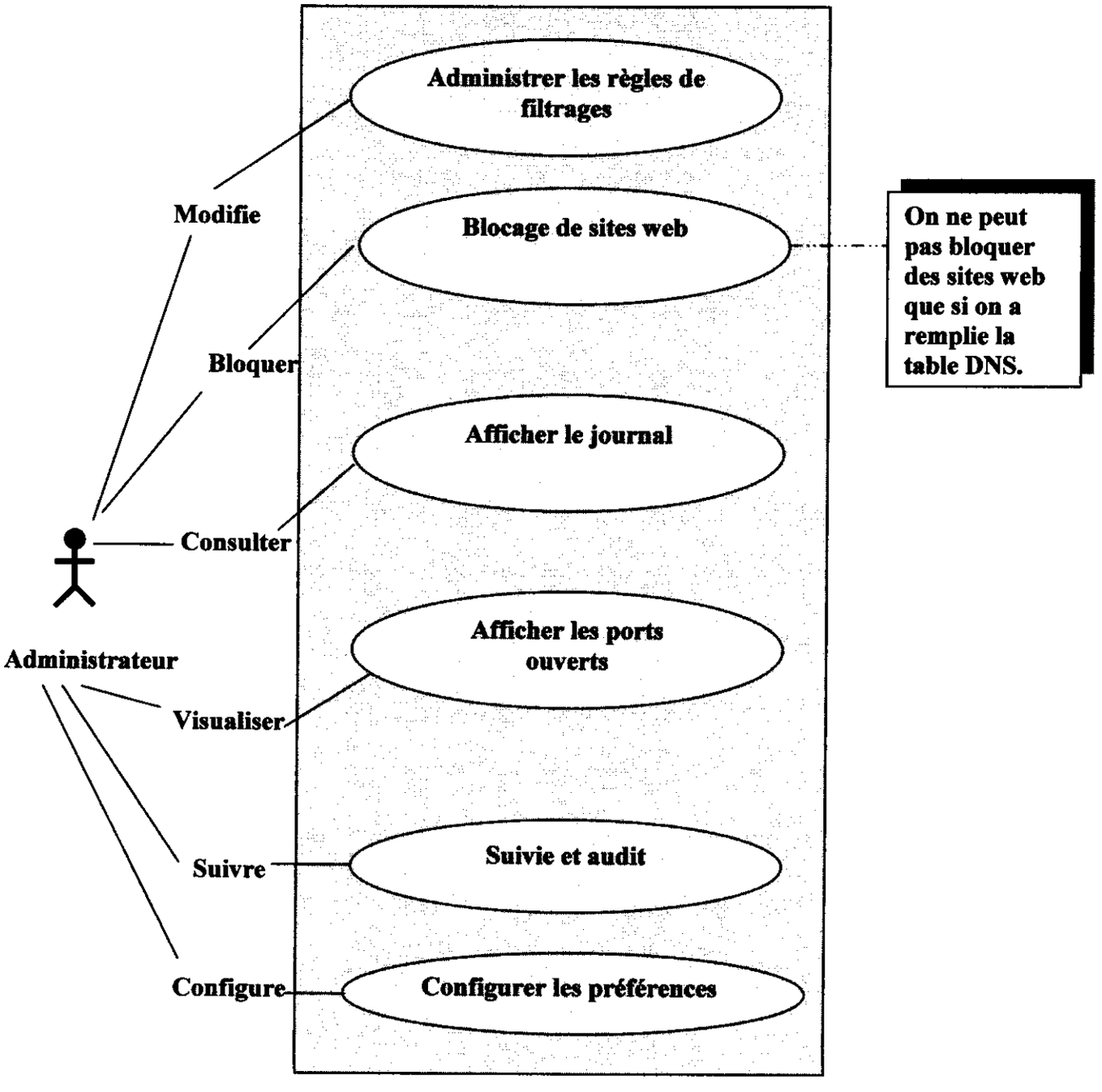
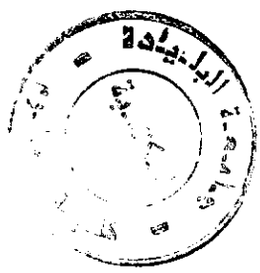


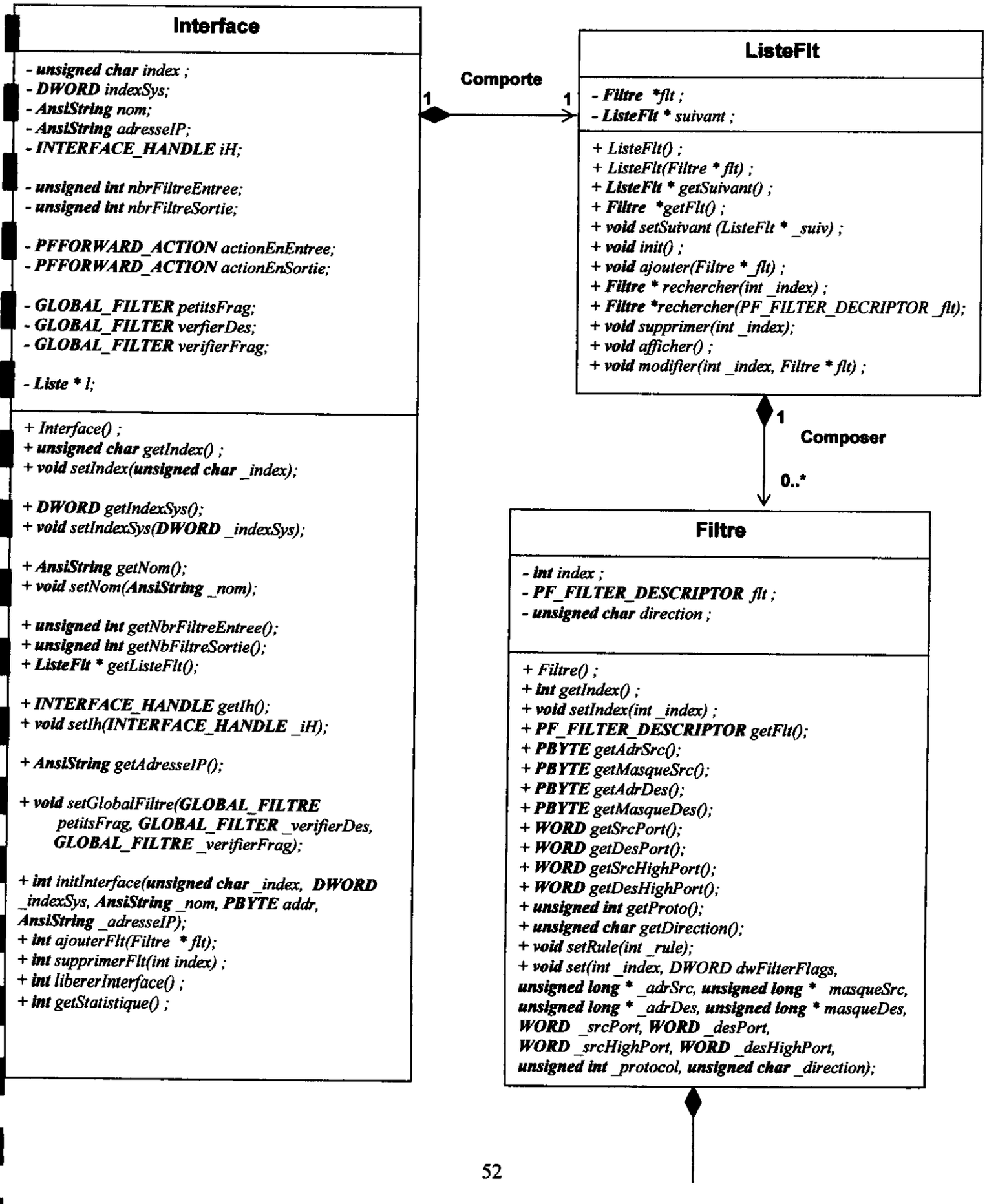
Fig 8 : Diagramme de cas d'utilisation

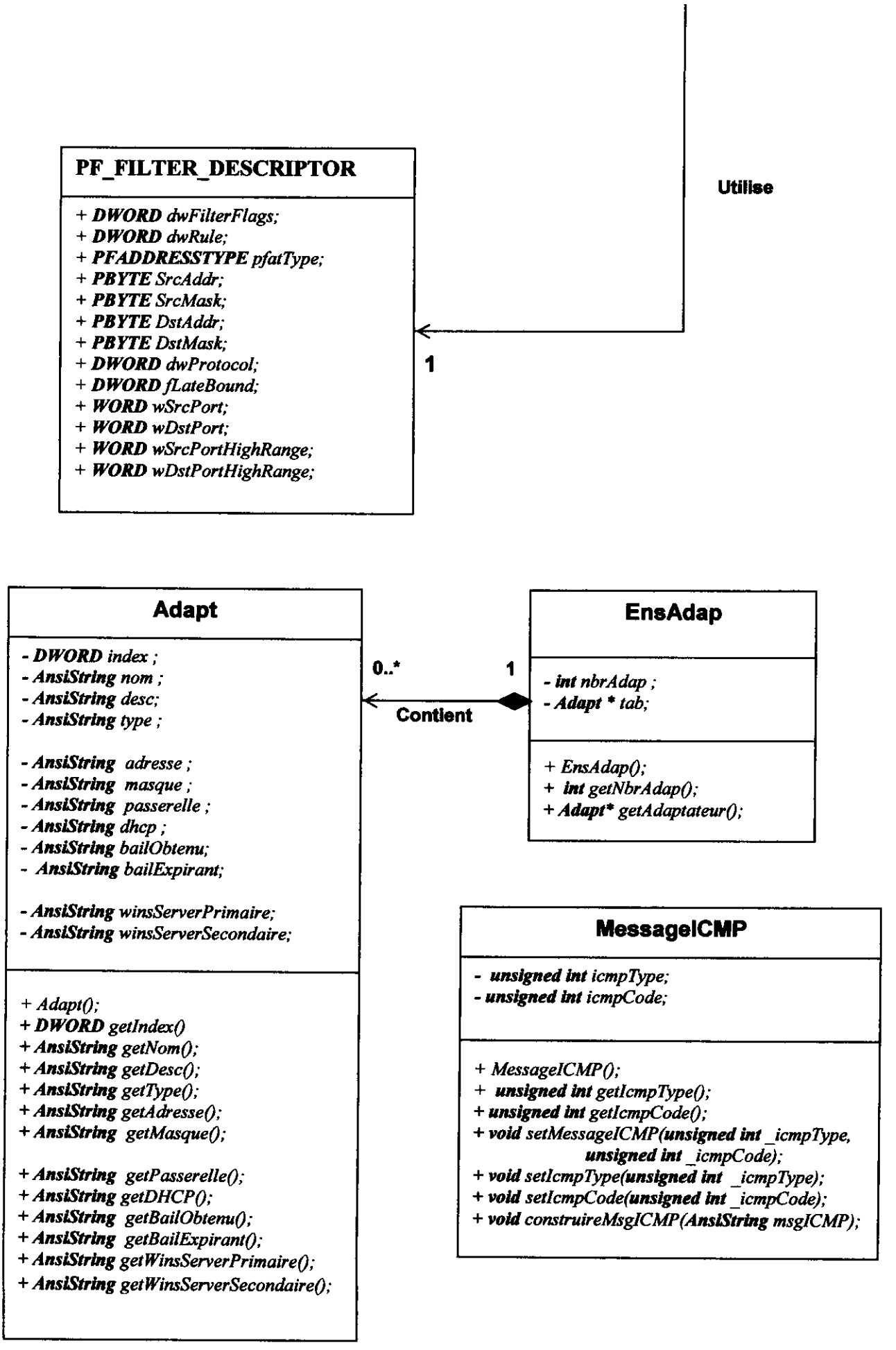
L'administrateur peut administrer les règles de filtrage, bloquer la consultation de sites web et peut a tout moment consulter le fichier journal.

Il peut aussi voir l'ensemble des ports ouvert, suivre le trafic réseau et configurer l'envoie du journal par Email.



3-3-2/Diagramme de classes




PF_FILTER_DESCRIPTOR
+ DWORD dwFilterFlags;
+ DWORD dwRule;
+ PFADDRESSTYPE pfatType;
+ PBYTE SrcAddr;
+ PBYTE SrcMask;
+ PBYTE DstAddr;
+ PBYTE DstMask;
+ DWORD dwProtocol;
+ DWORD fLateBound;
+ WORD wSrcPort;
+ WORD wDstPort;
+ WORD wSrcPortHighRange;
+ WORD wDstPortHighRange;
Utilise
1
Adapt
- DWORD index ;
- AnsiString nom ;
- AnsiString desc;
- AnsiString type ;
- AnsiString adresse ;
- AnsiString masque ;
- AnsiString passerelle ;
- AnsiString dhcp ;
- AnsiString bailObtenu;
- AnsiString bailExpirant;
- AnsiString winsServerPrimaire;
- AnsiString winsServerSecondaire;
+ Adapt();
+ DWORD getIndex()
+ AnsiString getNom();
+ AnsiString getDesc();
+ AnsiString getType();
+ AnsiString getAdresse();
+ AnsiString getMasque();
+ AnsiString getPasserelle();
+ AnsiString getDHCP();
+ AnsiString getBailObtenu();
+ AnsiString getBailExpirant();
+ AnsiString getWinsServerPrimaire();
+ AnsiString getWinsServerSecondaire();
0..*
1
Contient
EnsAdap
- int nbrAdap ;
- Adapt * tab;
+ EnsAdap();
+ int getNbrAdap();
+ Adapt* getAdaptateur();
MessageICMP
- unsigned int icmpType;
- unsigned int icmpCode;
+ MessageICMP();
+ unsigned int getIcmpType();
+ unsigned int getIcmpCode();
+ void setMessageICMP(unsigned int _icmpType, unsigned int _icmpCode);
+ void setIcmpType(unsigned int _icmpType);
+ void setIcmpCode(unsigned int _icmpCode);
+ void construireMsgICMP(AnsiString msgICMP);

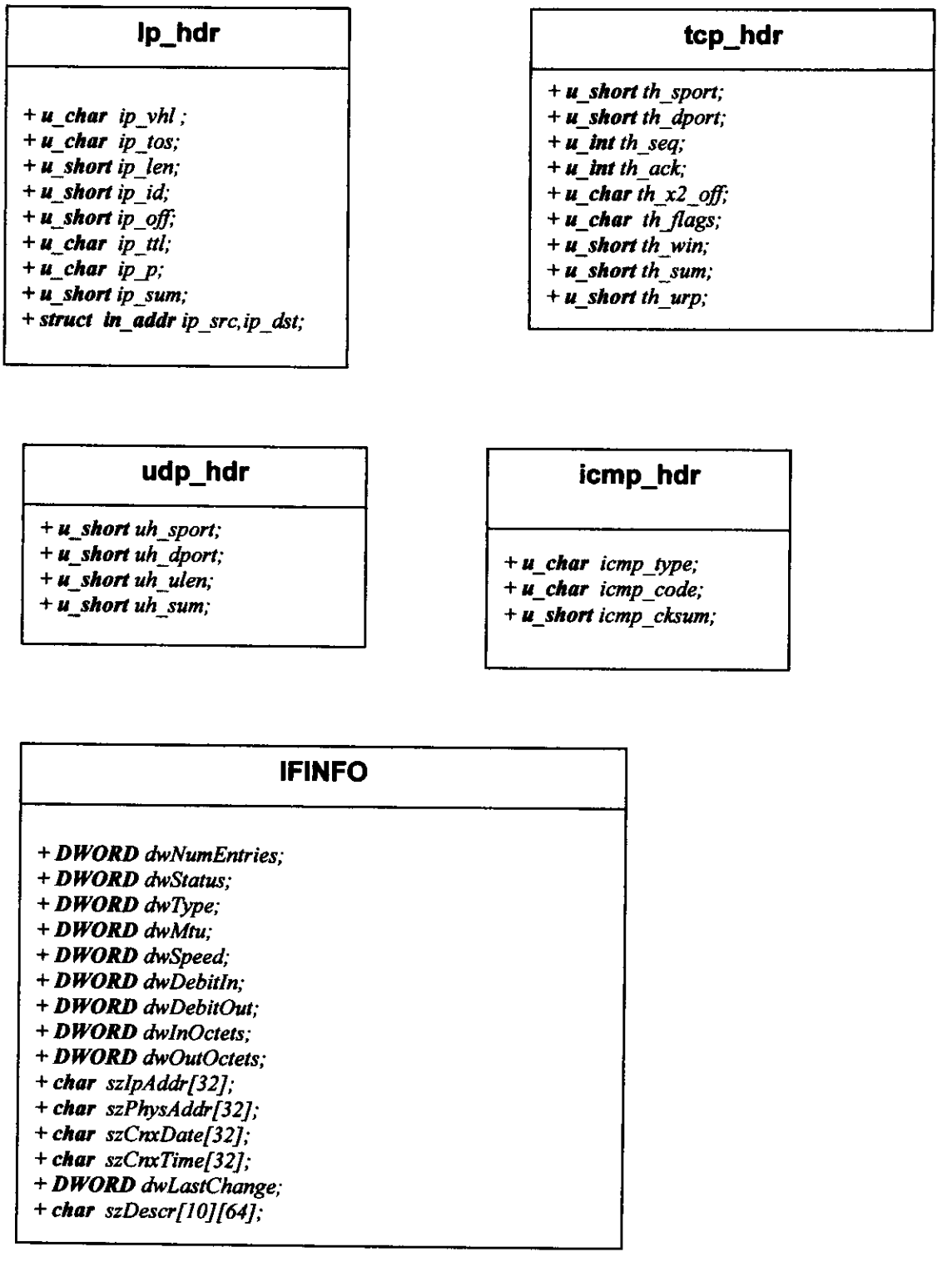


Fig9 : Diagramme de classes

L'application utilise un ensemble de classes qui ne sont pas forcement projeté dans le model conceptuel de données (MCD).

- Chaque interface comporte une seule liste de filtres.

- Une liste de filtre peut contenir plusieurs filtres de paquets.

- Un ensemble d'adaptateur réseaux peut être composé de plusieurs adaptateurs.

-La classe MessageICMP encapsule un message ICMP.

-Les classes ip_hdr , tcp_hdr, udp_hdr, icmp_hdr sont utilisées par le module de journalisation pour établir les alertes.

- La classe IFINFO nous permet de suivre le trafic réseau (débit en entrée, sortie etc....)

3-3-3/*Le modèle conceptuel de données (MCD):*

Ces tables ne sont pas représentées sous forme de classes dans le diagramme de classe mais elles constituent la base de données utilisée par l'application.

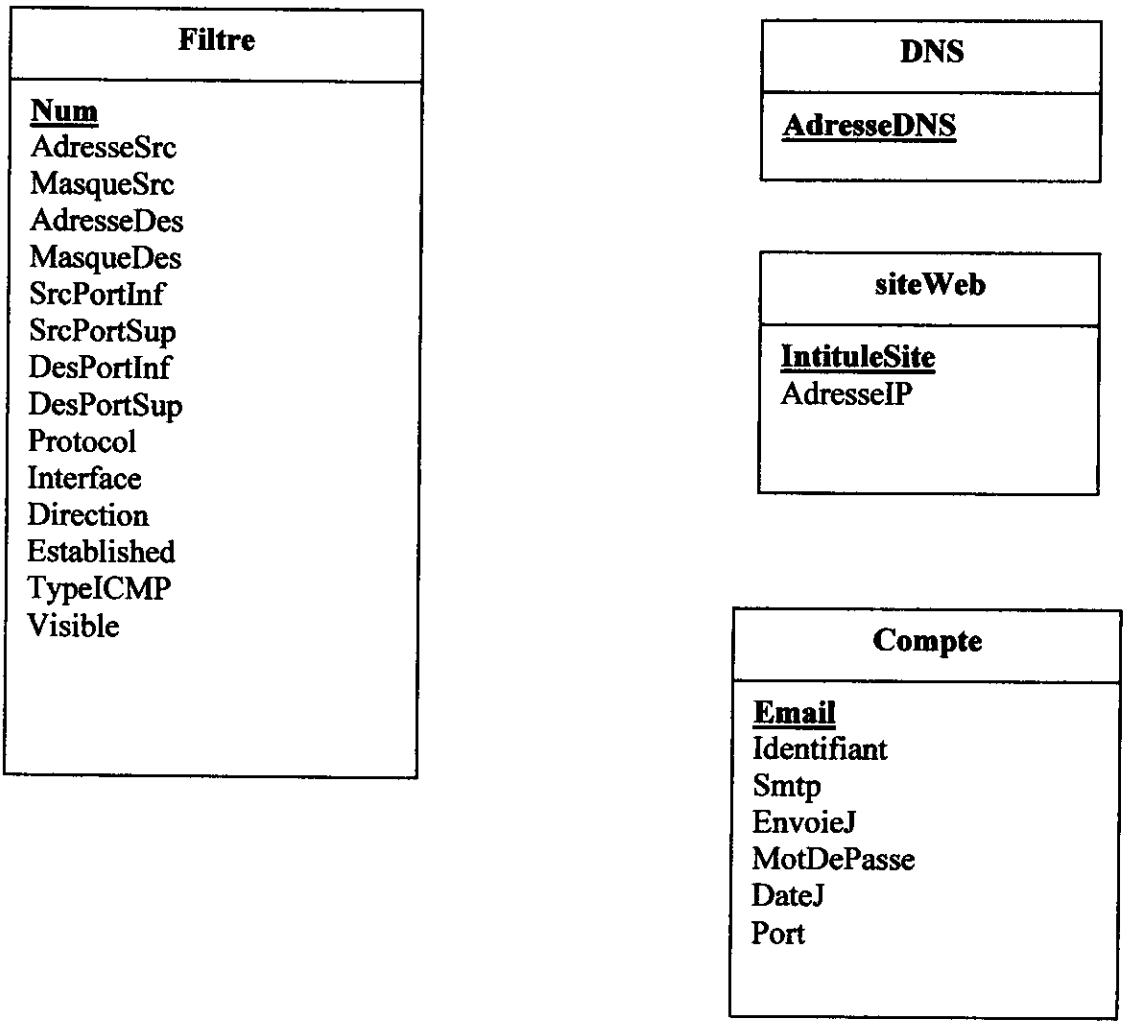


Fig 11 : MCD de l'application

Contrainte : *La table Compte ne contient qu'un seul enregistrement (ou bien un seul compte).*

➢ *Le dictionnaire des données :*

Code	Description	Type	*Mode*
Num	Numéro du filtre.	Numérique.	Mémorisée.
AdresseSrc	Adresse IP source du paquet a bloqué.	Alphanumérique.	Mémorisée.
MasqueSrc	Masque réseau source du paquet a bloqué.	Alphanumérique.	Mémorisée.
AdresseDes	Adresse IP destination du paquet a bloqué.	Alphanumérique.	Mémorisée.
MasqueDes	Masque réseau destination du paquet a bloqué.	Alphanumérique.	Mémorisée.
SrcPortInf	La borne inférieure du port source .	Numérique.	Mémorisée.
SrcPortSup	La borne supérieure du port source.	Numérique.	Mémorisée.
DesPortInf	La borne inférieure du port destination.	Numérique.	Mémorisée.
DesPortSup	La borne supérieure du port destination.	Numérique.	Mémorisée.
Protocol	Le protocole utilisé.	Alphanumérique.	Mémorisée.
Interface	L'interface sur laquelle est établis le filtre.	Alphanumérique.	Mémorisée.
Direction	Direction du filtre en entrée ou en sorite.	Alphanumérique.	Mémorisée.
Established	Accepter ou refuser les ACK.	Booléen.	Mémorisée.
TypeICMP	Message ICMP.	Alphanumérique.	Mémorisée.
Visible	La règle est visible ou non.	Booléen.	Mémorisée.
AdresseDNS	Adresse serveur DNS.	Alphanumérique.	Mémorisée.
IntituleSite	URL du site web a bloqué.	Alphanumérique.	Mémorisée.
AdresseIP	Adresse IP du site web.	Alphanumérique.	Mémorisée.
Email	Email de L'Administrateur.	Alphanumérique.	Mémorisée.
Identifiant	Identifiant de l'émail.	Alphanumérique.	Mémorisée.
Smtp	Serveur SMTP.	Alphanumérique.	Mémorisée.
EnvoieJ	Nombre de jour pour envoyer un émail.	Numérique.	Mémorisée.
MotDePasse	Mot de passe de l'émail.	Alphanumérique.	Mémorisée.
DateJ	Date futur de l'envoie de l'émail.	Date.	Mémorisée.
Port	Port du serveur SMTP.	Numérique.	Mémorisée.

CHAPITRE 4
IMPLEMENTATION D'UNE SOLUTION

4 – Implémentation d'une solution

4-1/ Choix de la plate-forme et du langage de programmation

Le langage de programmation utilisé est le C++ (IDE C++Builder6) et la plate-forme est WindowsXP ou WindowsNT .

4-1-1/Pourquoi choisir la plate-forme Windows et pas Linux ?

- Comme l'ensemble des machines du réseau de la CNAS tournent sous Windows, alors on a décidé que le développement de notre application sera sous le système Windows.

4-1-2/Pourquoi choisir le C++ comme langage de programmation et pas Java ?

- L'ensemble des API de filtrage sont écrits pour être utilisé par le langage C++ et sous la plate forme Windows.

- L'application n'est pas destinée a être utilisée sous plusieurs systèmes d'exploitations(portabilité) d'ou le langage C++ est le mieux adapté pour le développement de ce logiciel , en plus de ça le langage Java (Machine virtuelle) est plus lent que le C++.

4-2/ Architecture de l'application

-L'application repose sur plusieurs modules :

- Module de base de données.
- Module de filtrage de paquets.
- Module de résolution de noms de domaines.
- Module de blocage de sites web.
- Module journal.
- Module ports ouverts.
- Module d'audit.
- Module d'envoie d'emails.
- Module d'interfaces réseaux.

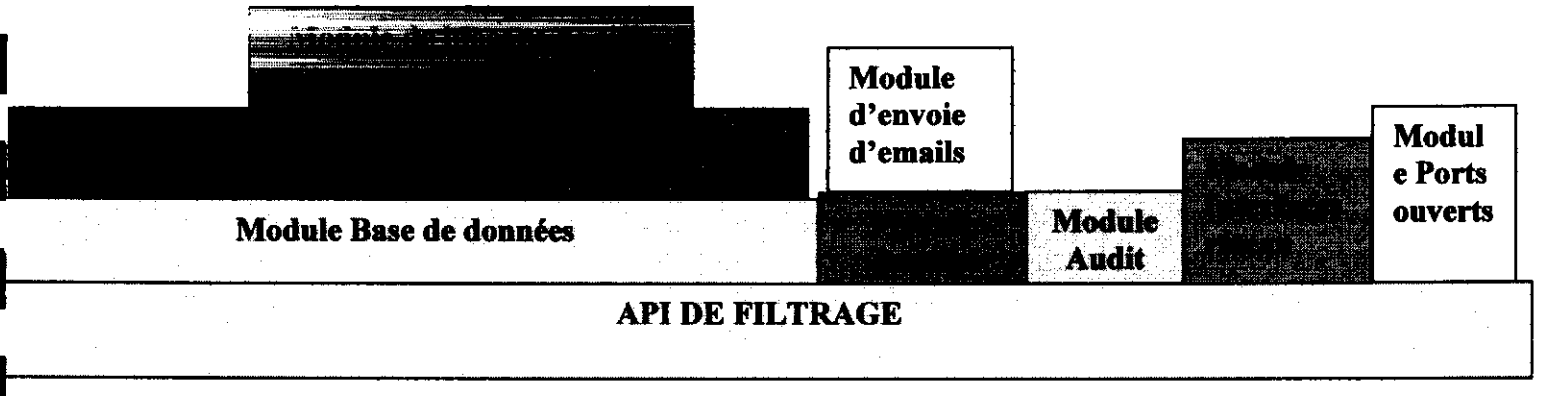


Fig 10 : Architecture de l'application.

4-2-1/ Description des modules de l'application :

- Module de base de données :

- La module de base de données comporte l'ensemble des tables utilisées dans l'application.
L'application utilise quatre tables :

  - La table **filtre** contient l'ensemble des règles de filtrage établi par l'administrateur.

  - La table **DNS** contient les adresses des serveurs DNS utilisés pour faire la résolution de noms
    des sites web à bloquer.

  - La table **siteWeb** comporte l'ensemble des sites web bloqués ainsi que leurs adressesIP correspondantes. (Les adresses IP des sites sont déduits après l'opération de résolution de noms).

  - La table **Compte** contient des informations relatives à l'administrateur.

➢ *Le modèle conceptuel de données (MCD):*

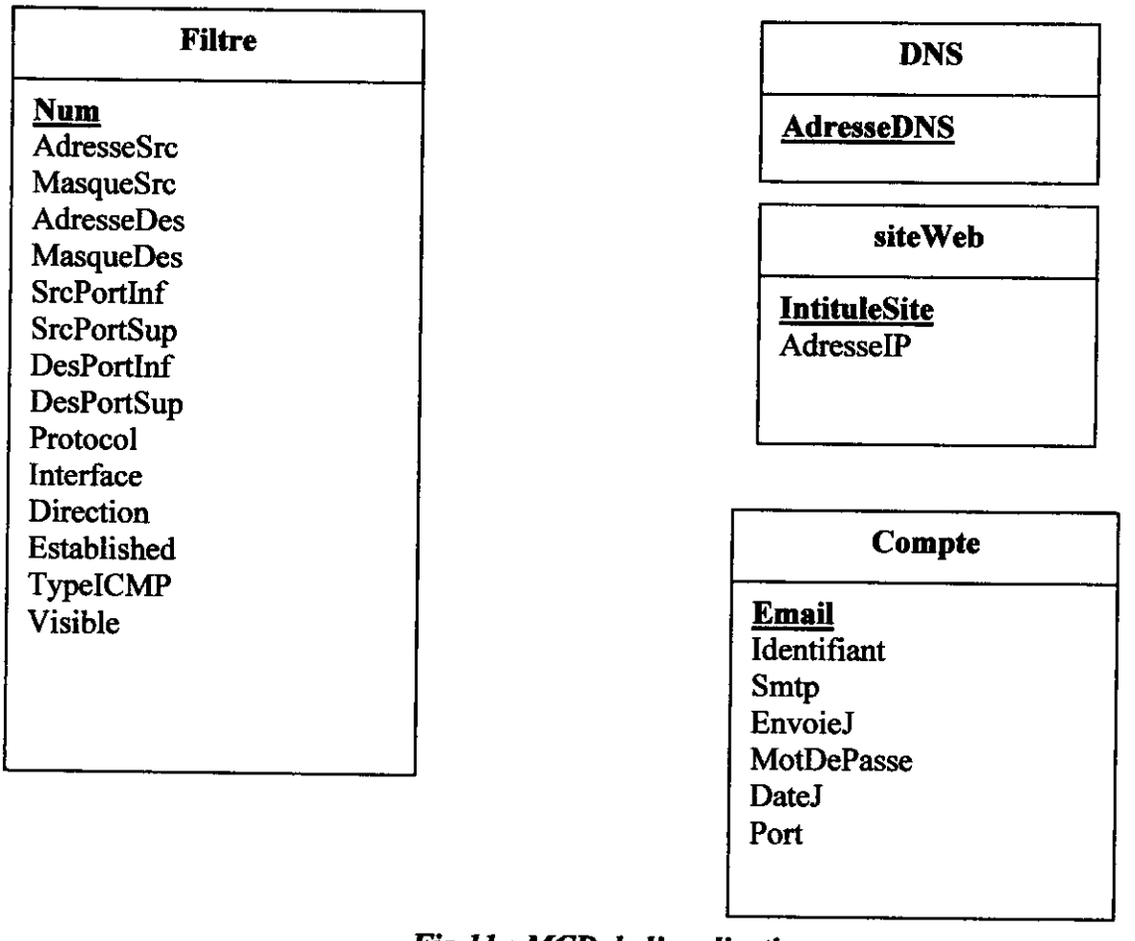


Fig 11 : MCD de l'application

Contrainte : *La table Compte ne contient qu'un seul enregistrement (ou bien un seul compte).*

➢ *Le dictionnaire des données :*

Code	Description	Type	*Mode*
Num	Numéro du filtre.	Numérique.	Mémorisée.
AdresseSrc	Adresse IP source du paquet a bloqué.	Alphanumérique.	Mémorisée.
MasqueSrc	Masque réseau source du paquet a bloqué.	Alphanumérique.	Mémorisée.
AdresseDes	Adresse IP destination du paquet a bloqué.	Alphanumérique.	Mémorisée.
MasqueDes	Masque réseau destination du paquet a bloqué.	Alphanumérique.	Mémorisée.
SrcPortInf	La borne inférieure du port source .	Numérique.	Mémorisée.
SrcPortSup	La borne supérieure du port source.	Numérique.	Mémorisée.
DesPortInf	La borne inférieure du port destination.	Numérique.	Mémorisée.
DesPortSup	La borne supérieure du port destination.	Numérique.	Mémorisée.
Protocol	Le protocole utilisé.	Alphanumérique.	Mémorisée.
Interface	L'interface sur laquelle est établis le filtre.	Alphanumérique.	Mémorisée.
Direction	Direction du filtre en entrée ou en sorite.	Alphanumérique.	Mémorisée.
Established	Accepter ou refuser les ACK.	Booléen.	Mémorisée.
TypeICMP	Message ICMP.	Alphanumérique.	Mémorisée.
Visible	La règle est visible ou non.	Booléen.	Mémorisée.
AdresseDNS	Adresse serveur DNS.	Alphanumérique.	Mémorisée.
IntituleSite	URL du site web a bloqué.	Alphanumérique.	Mémorisée.
AdresseIP	Adresse IP du site web.	Alphanumérique.	Mémorisée.
Email	Email de L'Administrateur.	Alphanumérique.	Mémorisée.
Identifiant	Identifiant de l'émail.	Alphanumérique.	Mémorisée.
Smtp	Serveur SMTP.	Alphanumérique.	Mémorisée.
EnvoieJ	Nombre de jour pour envoyer un émail.	Numérique.	Mémorisée.
MotDePasse	Mot de passe de l'émail.	Alphanumérique.	Mémorisée.
DateJ	Date futur de l'envoie de l'émail.	Date.	Mémorisée.
Port	Port du serveur SMTP.	Numérique.	Mémorisée.

- Module de filtrage de paquets.

-Le module de filtrage de paquets utilise les API de filtrage de paquets(*voir dans l'annexe*) , l'enregistrement des règles de filtrage se fait dans la table **Filtre**.

-L'administrateur du firewall peut ajouter ou supprimer une règle, comme il peut visualiser l'ensemble des règles de filtrages établis.

- Module de résolution de noms de domaines :

- Le module de résolution de noms de domaines se charge de trouver l'adresse IP correspondante au nom de domaine d'un site web , il utilise la table **DNS** (Les adresses des serveurs DNS) pour faire de la résolution de nom.

- Une fois l'opération de résolution est faite , le résultat est communiqué au module de blocage de sites web.

- Module de blocage de sites web :

- Ce module offre a l'administrateur la possibilité de bloquer n'importe quel site web.

- Une fois que l'administrateur a spécifié le site web à bloquer, le module de blocage de site web va solliciter le module de résolution de noms pour lui trouver l'adresse IP correspondante au site web , après la terminaison de l'opération de résolution de noms, le module de blocage de sites web va transmettre l'adresse IP du site web au module de filtrage de paquets afin qu'il établisse une règle de filtrage pour bloquer l'accès vers ce site web.

- L'administrateur peut ajouter ou supprimer un site web de la liste des sites web bloqués.

- Module journal :

- La journalisation est une fonction importante d'un pare-feu. Elle sera très utile pour suivre la trace des paquets bloqués.

- Le module de journalisation permet a l'administrateur de suivre l'ensemble des paquets bloqués par le logiciel firewall.

- L'administrateur peut consulter le fichier journal et l'enregistrer sous forme de page HTML.

- Module ports ouverts:

- Le module ports ouverts affiche l'ensemble des ports ouverts sur la machine contenant le logiciel pare-feu.

- Module d'audit :

- Le module d'audit visualise la volumétrie du réseau, il permet d'avoir une vision sur la consommation du réseau en particulier en entrée .Les graphiques donnent le volume transféré au cours du temps. Les informations sont stockées dans le graphique, leur volume est donc constant au cours du temps.

- La visualisation du trafic réseau est une information importante pour la sécurité. Il arrive fréquemment qu'une consommation réseau anormale soit la conséquence de la compromission d'une machine. C'est le cas quand la machine est utilisée comme serveur warez par les pirates.

- La métrologie est également très utile pour une gestion du réseau, en particulier pour une meilleur utilisation de la bande passante disponible.

- Ce module offre aussi des statistiques (Le nombre de paquets bloqués , le nombre de paquets loguer , Le nombre de paquets perdus etc...).

- L'administrateur peut suivre l'ensemble des adaptateurs réseaux présent sur la machine contenant le pare-feu.

- Module d'envoi d'emails :

- L'administrateur peut configurer les préférences du pare-feu pour recevoir des emails en lui spécifiant la fréquence d'envoi d'émail en nombre de jours.

- L'émail sera accompagné du fichier journal comme pièce jointe.

- Module d'interfaces réseaux :

Ce module permet à l'administrateur de visualiser l'ensemble des adaptateurs réseaux.

4-3 / Interfaces graphique de l'application :

Voici quelques images illustrant l'interface graphique de l'application :

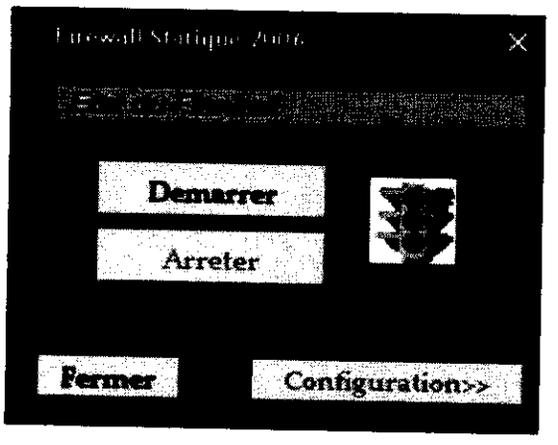


Fig 12 : Etat du firewall (Arrêter)

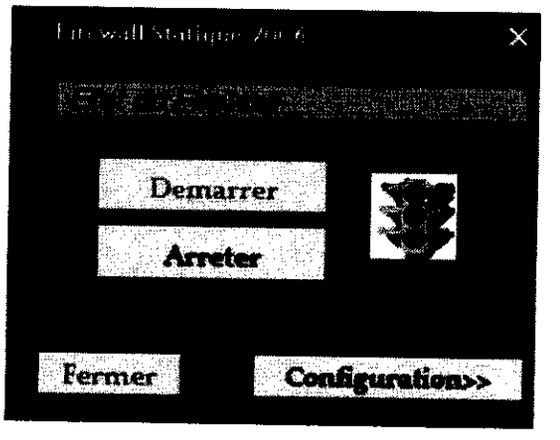


Fig 13 : Etat du firewall (Démarrer)

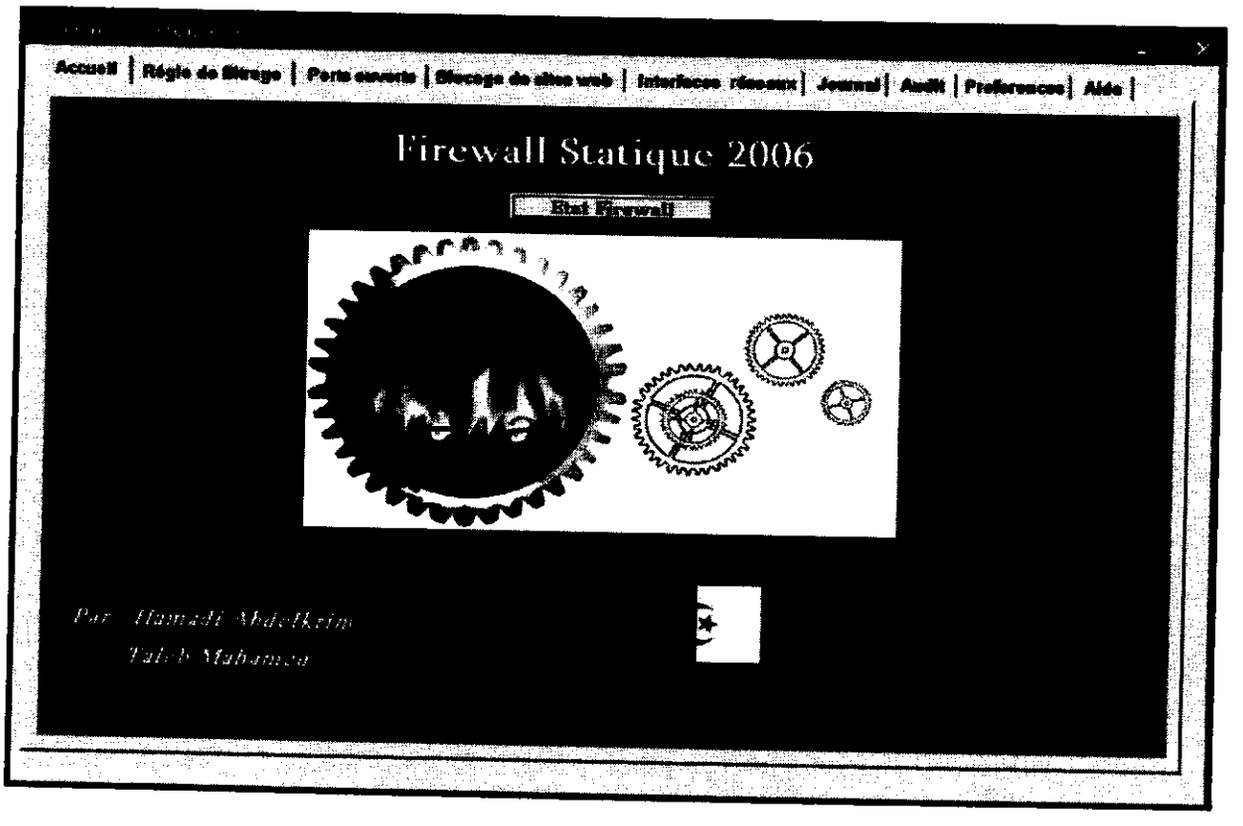


Fig 14 : Centre de configuration

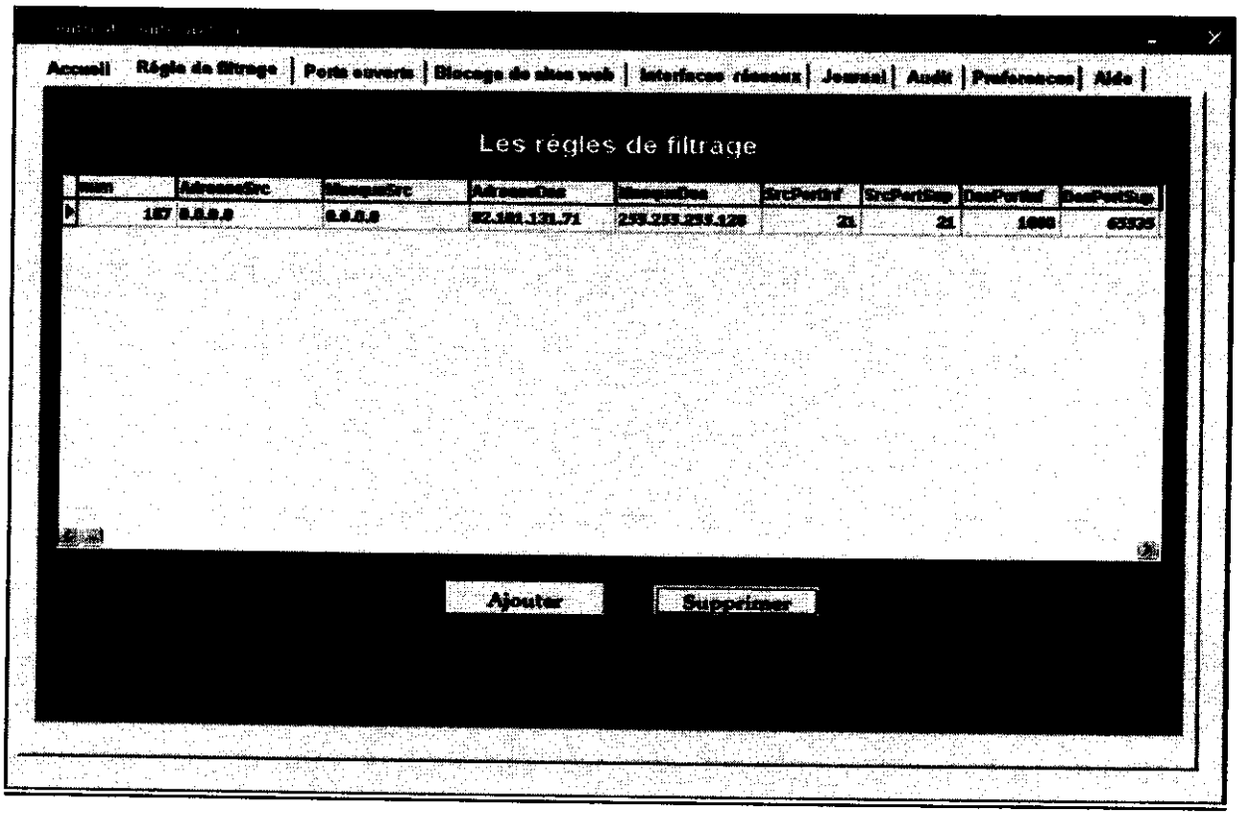


Fig 15 : Onglet Règle de filtrage

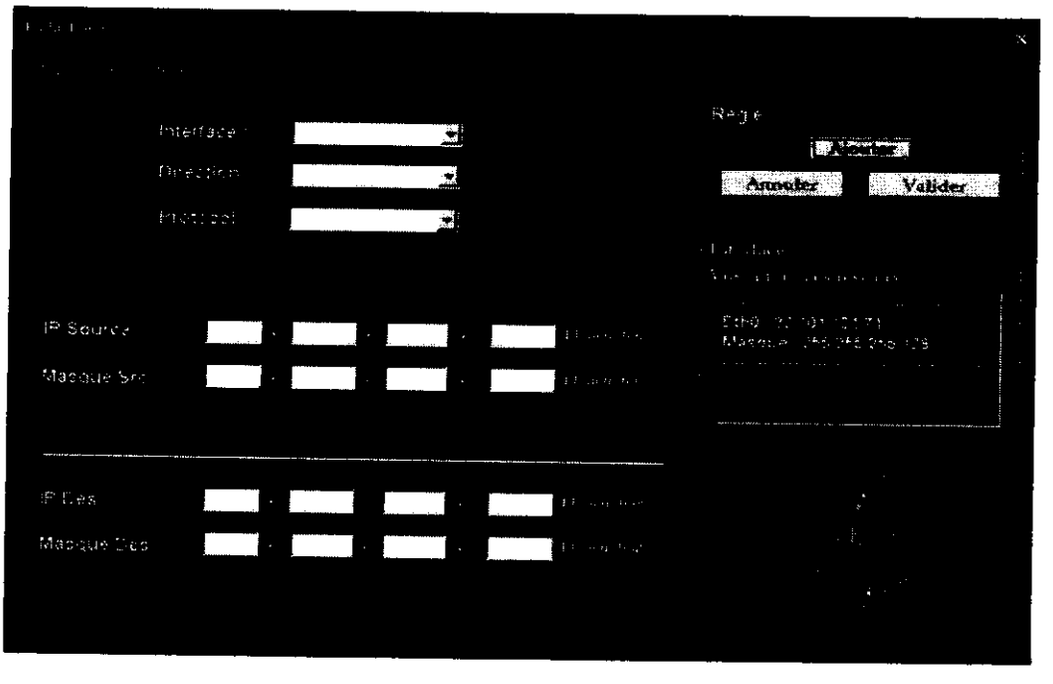


Fig 16 : Fiche règle de filtrage

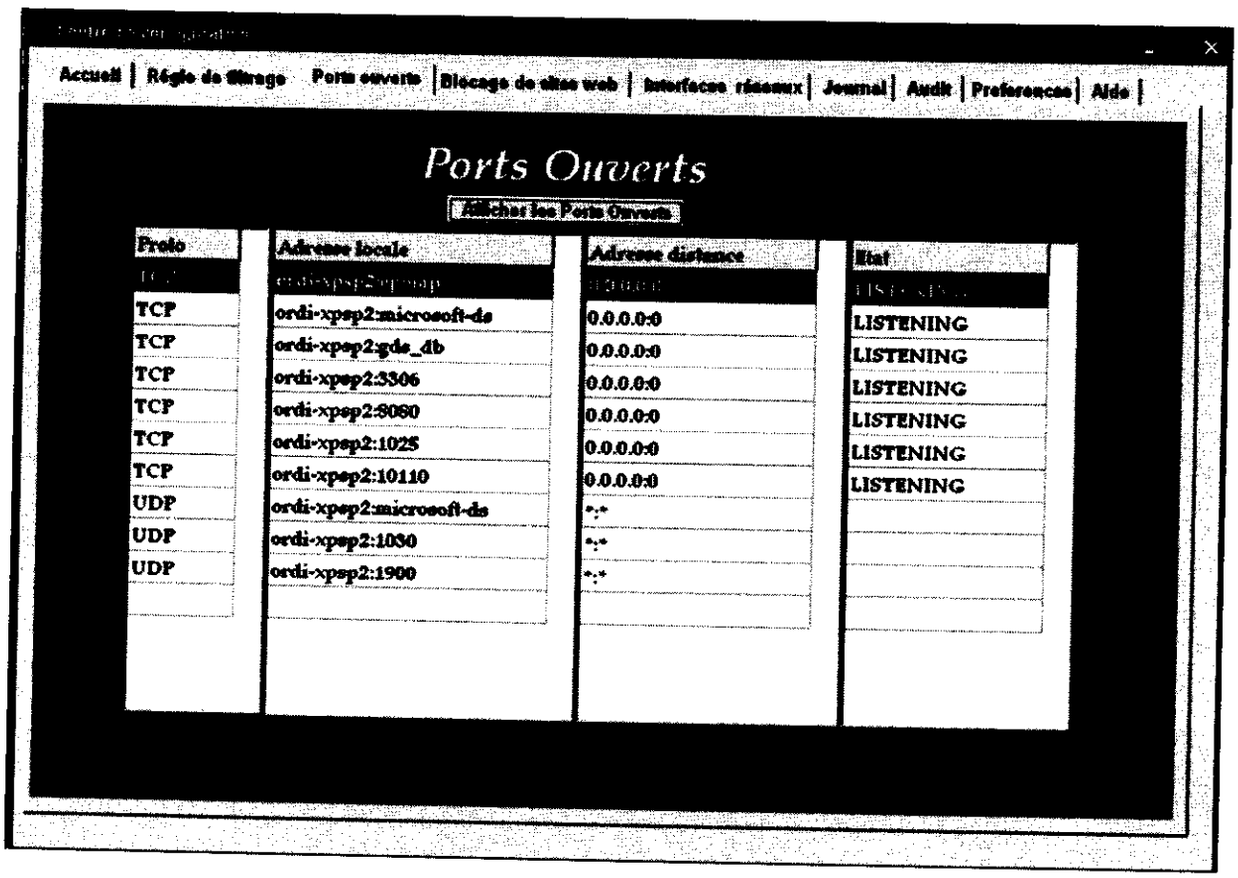


Fig 17 : Onglet ports ouverts

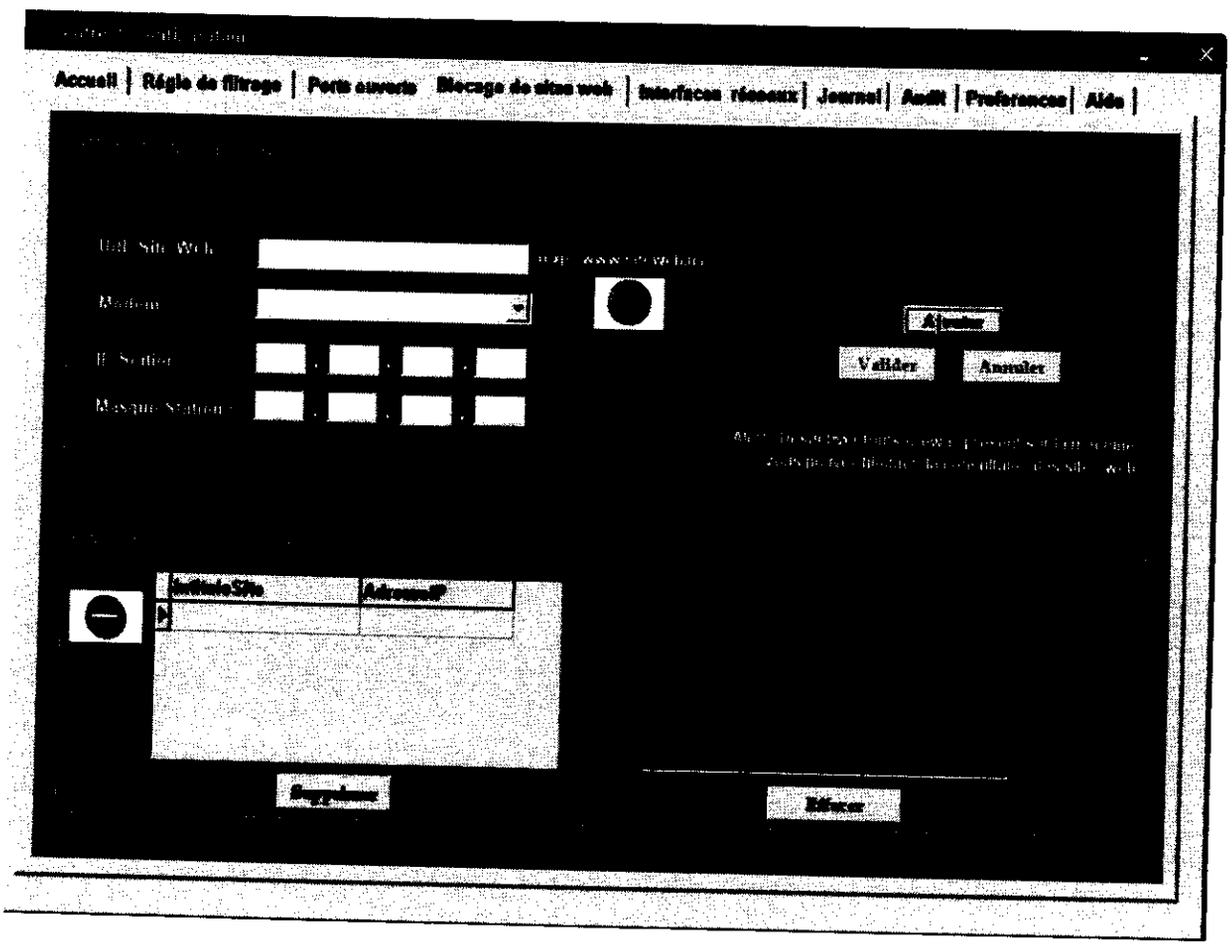


Fig 18 : Onglet blocages de sites web.

Fig 19 : Onglet adaptateurs réseaux

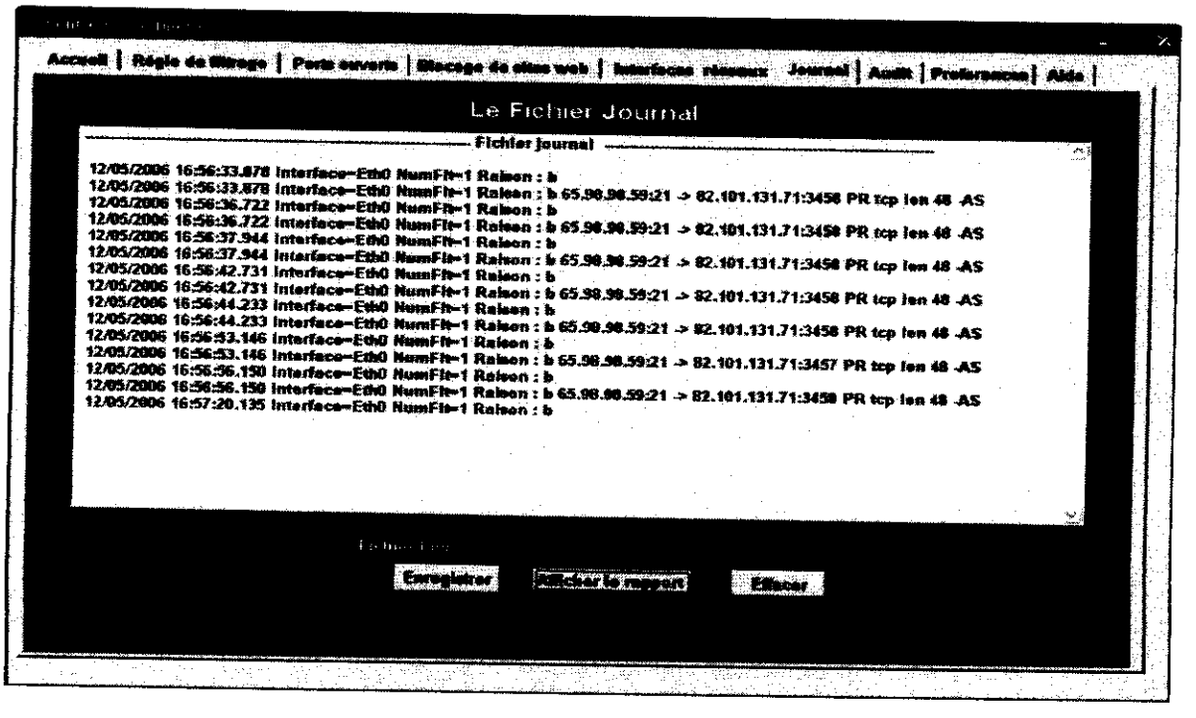


Fig 20 : Onglet journal

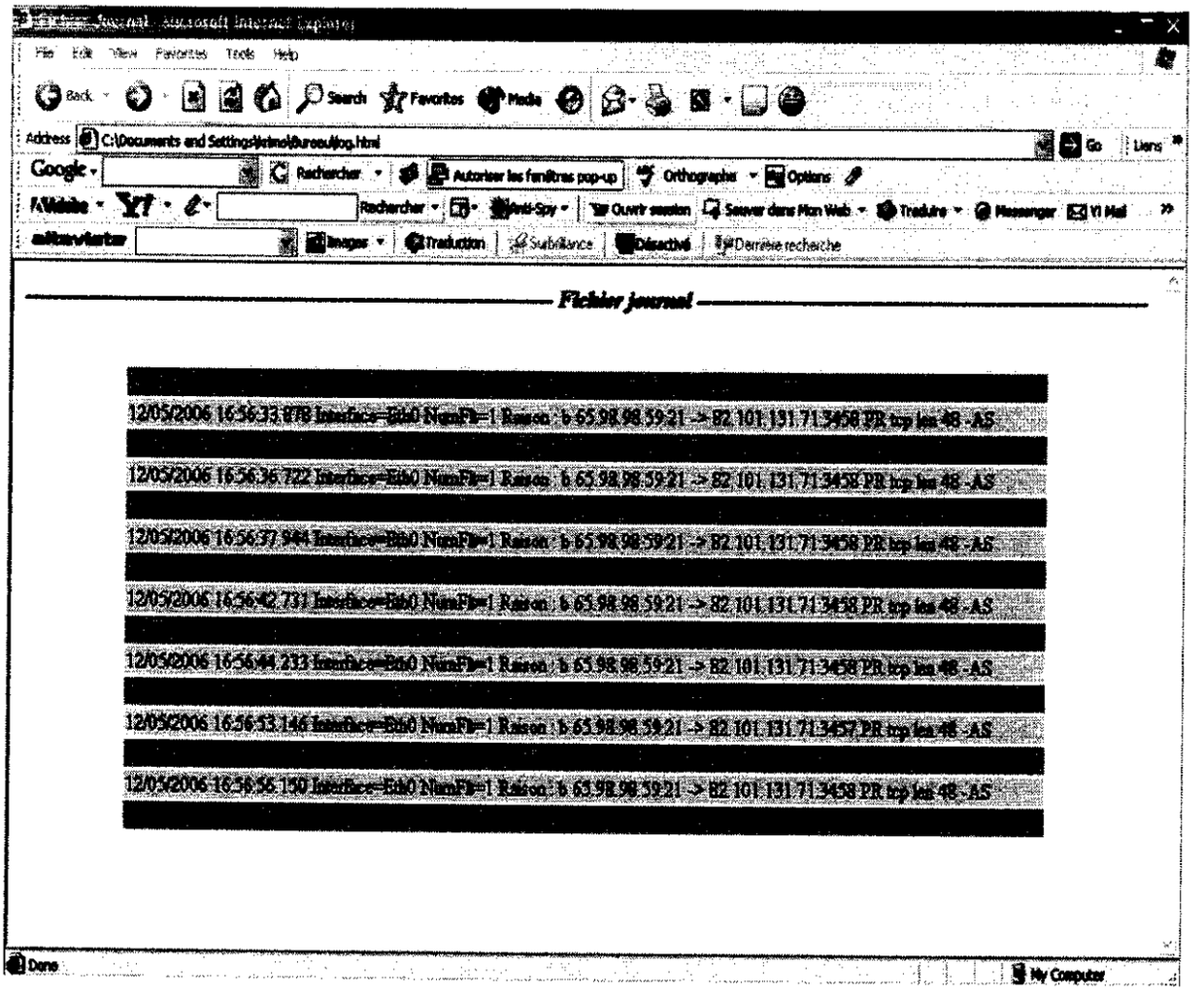


Fig 21 : Aspect du fichier journal enregistré
(Sous forme de page HTML)

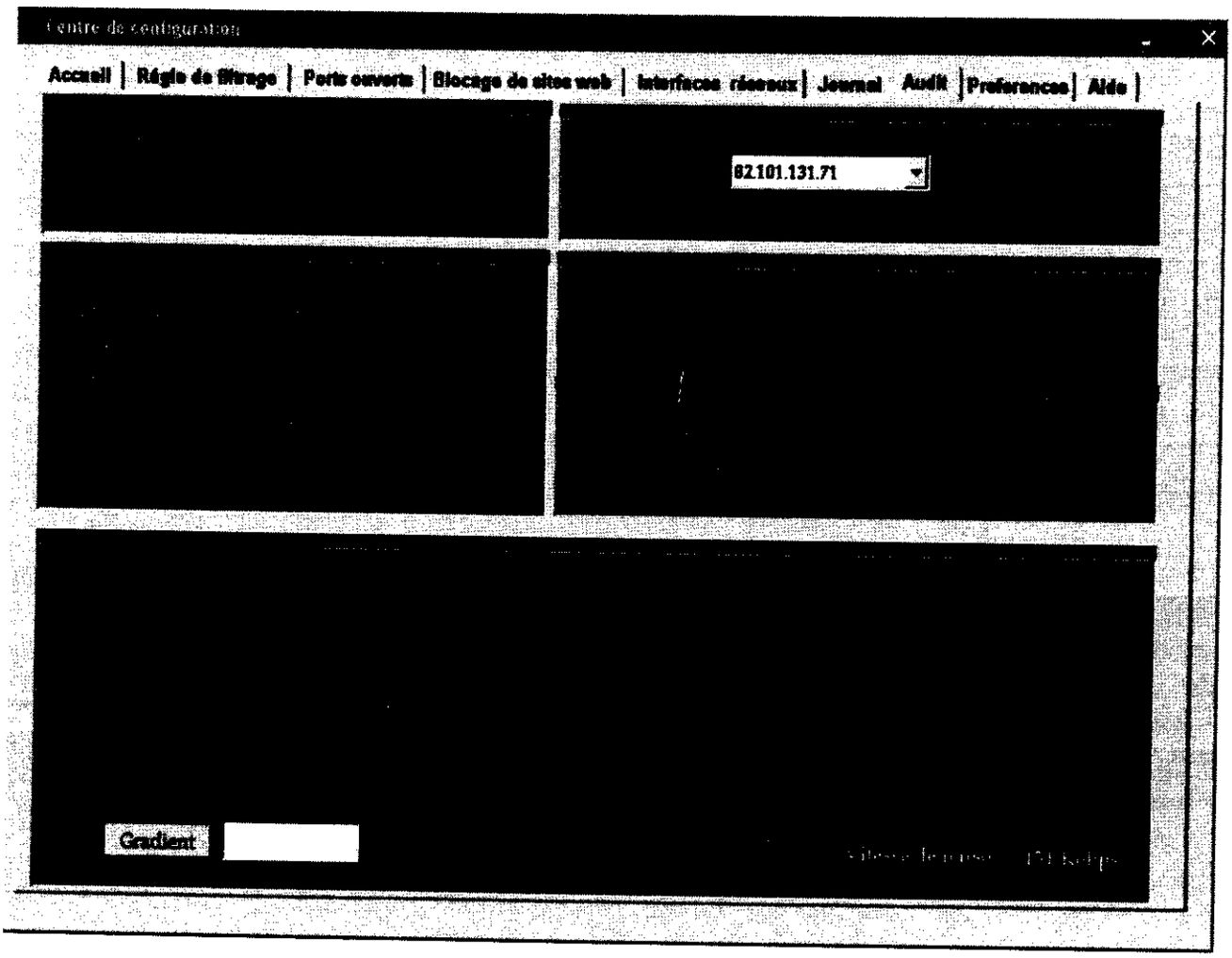


Fig 22 : Onglet audit

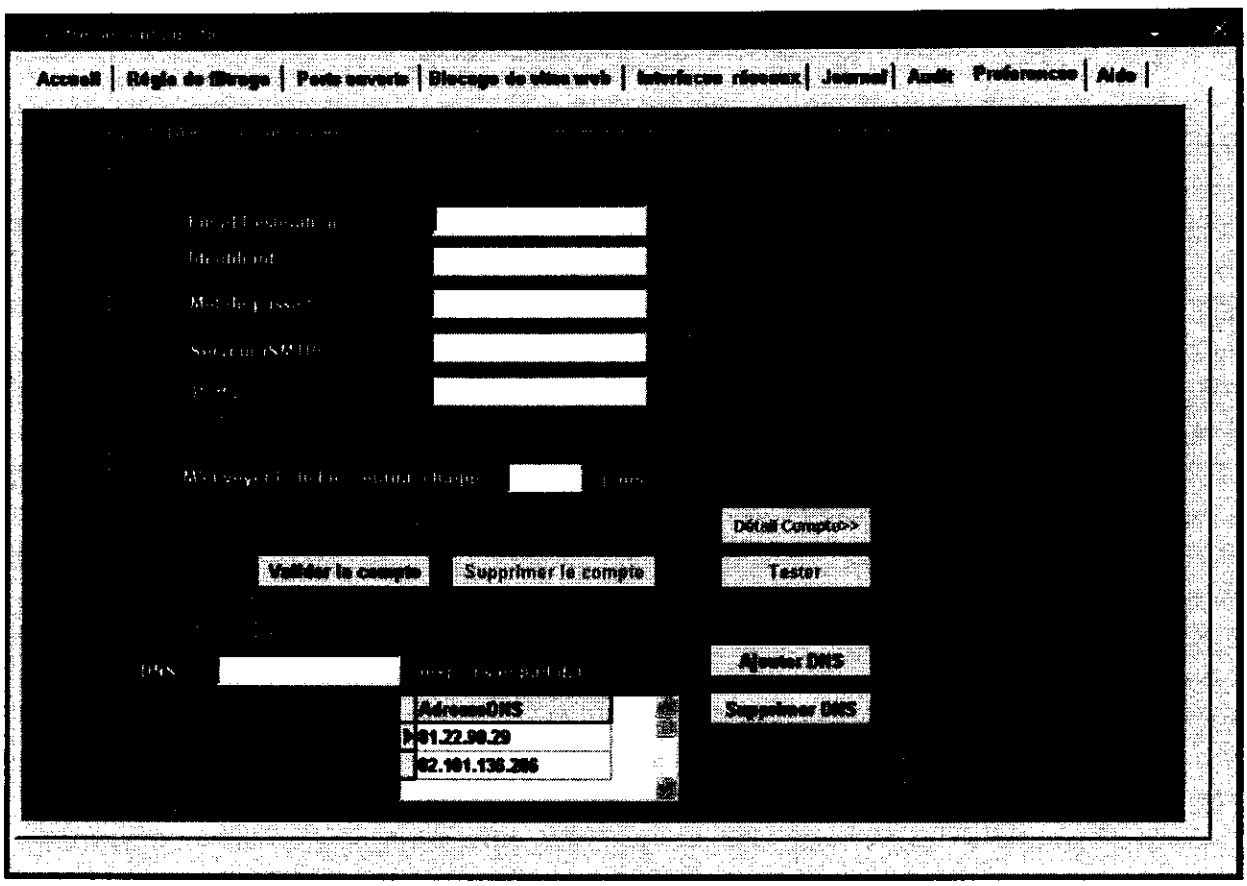


Fig 23 : Onglet préférences

CHAPITRE 5 : ENVIRONNEMENT DE TEST

5 – Environnement de test

5-1/ Présentation de l'organisme d'accueil

La CNAS est un établissement à caractère public , elle prend en charge la gestion de la sécurité du régime des salariés, en mettant en place un système de solidarité nationale**[2].**

la C.N.A.S constitue la pièce maîtresse et la pierre angulaire de tout le système algérien de protection sociale. En effet, elle concerne et protège plus de 80% de la population contre la quasi totalité des risques de la vie quotidienne et verse des revenus substantiels de subsistance sous forme de prestations diverses à près de 6 millions de bénéficiaires directs**[2].**

5-2/La sécurité informatique dans La CNAS

Dans le cadre général d'une politique de promotion en faveur d'une informatique plus sécurisée et efficace dans le secteur public, la direction générale de la Caisse Nationale des Assurances Sociales (CNAS) avait mesuré l'importance d'avoir un réseau performant et sécurisé.

Les précédents schémas de l'informatique avaient omis d'intégrer dans leurs objectifs les politiques de sécurité.

Mais Aujourd'hui la CNAS suit l'évolution de l'informatique et les enjeux de la sécurité informatique dans la préservation de leurs données , c'est pour cela des moyens draconiens sont déployés pour protéger leur réseau informatique des attaques malveillantes .
La CNAS utilise des techniques modernes tel que la segmentation et le DMZ dans leur architecture réseaux**[2].**

5-3/Réseau de la CNAS :

Voici une image réduite du réseau national de la CNAS :

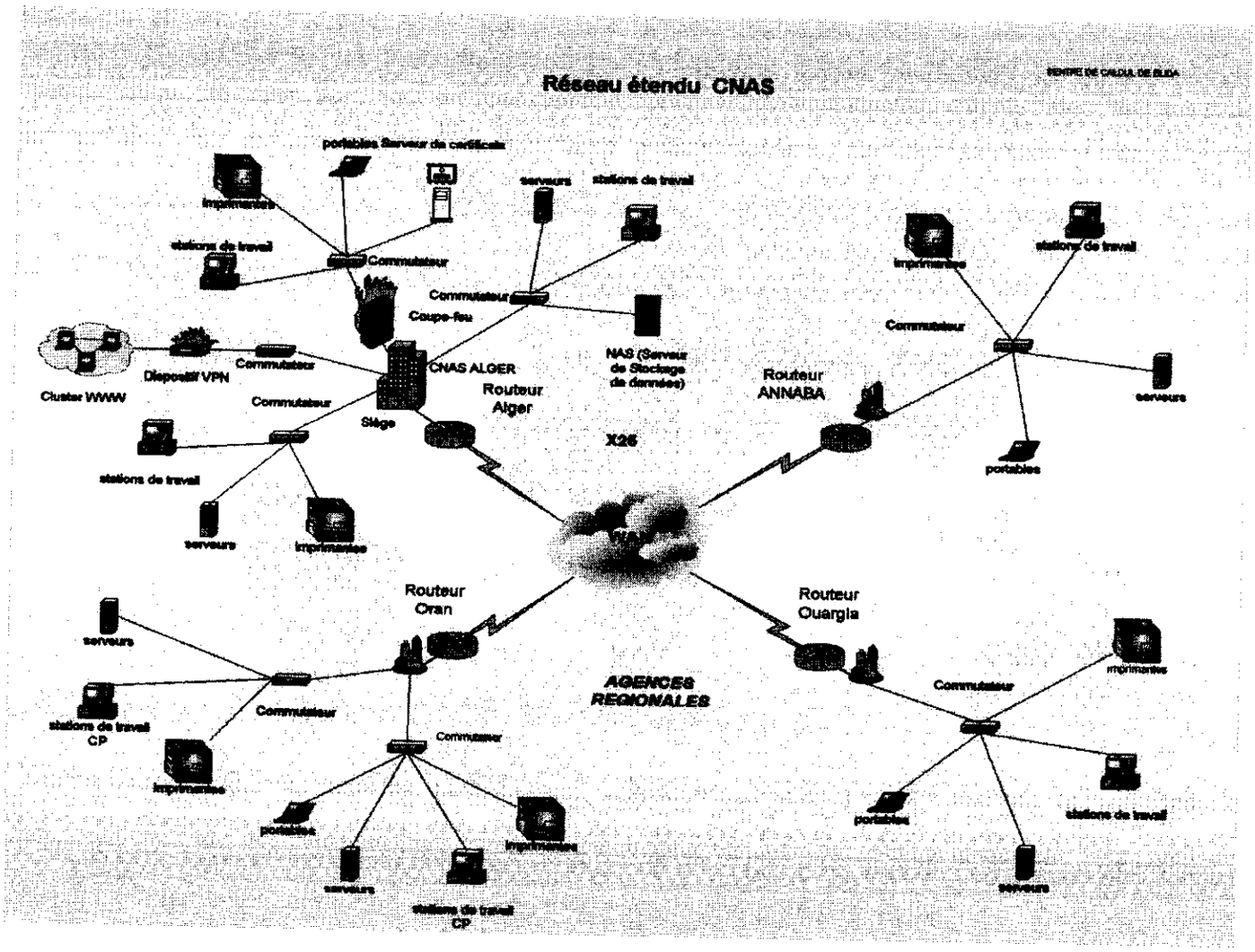


***Fig 24 :Réseau national de la CNAS*[2].**

5-4/ Exemple d'exécution

On a mis notre firewall logiciel sur le point Coupe-feu de l'image réduite du réseau national de la CNAS, d'ou on contrôlait le trafic réseau entrant et sortant du siège CNAS ALGER.

Dans un premier temps on a bloqué l'accès HTTP via le port 80(en laissant passer que les postes qui ont le droit de surfer sur Internet), puis on a bloqué l'accès FTP via le port 21, et on a désactivé la prise en compte des requêtes « **ICMP Echo Request** ».
On a essayé le module de blocage de site web en bloquant une liste de sites web qui était considéré comme une liste noire de sites web à ne pas consulter par les employés de la CNAS.

Une panoplie d'outils offerts par notre firewall logiciel a été mis a la disposition de l'administrateur réseau de la CNAS pour le suivi des effets de règles mises en œuvre(l'audit). L'administrateur pouvait à tout moment consulter le fichier journal pour voir l'ensemble des paquets bloqués en plus de cela il pouvait visualiser le débit réseau ainsi que les statistiques sur l'ensemble des paquets qui traversent le pare-feu.

Voici l'ensemble des règles utilisées :

	Regle1(Blocage HTTP)	***Regle2(Blocage FTP)***
Num	1	2
AdresseSrc	0.0.0.0	0.0.0.0
MasqueSrc	0.0.0.0	0.0.0.0
AdresseDes	81.101.130.71	81.101.130.71
MasqueDes	255.255.255.128	255.255.255.128
PortSrcInf	80	21
PortSrcSup	80	21
PortDesInf	1000	1000
PortDesSup	65535	65535
Protocol	TCP	TCP
Interface	81.101.130.71	81.101.130.71
Direction	Entree	Entrée
Established	True	True
Type ICMP	Aucun	Aucun

	Regle3(Blocage du ping)
Num	3
AdresseSrc	0.0.0.0
MasqueSrc	0.0.0.0
AdresseDes	81.101.130.71
MasqueDes	255.255.255.128
Protocol	ICMP
Interface	81.101.130.71
Direction	Sortie
Type ICMP	Echo request

Conclusion générale

Notre perspective est de permettre une administration du Firewall à distance par le bais d'un serveur Web, et de développer les deux autres modes de filtrages (Dynamique et applicatif).

FIGURES

REFERENCES

[1] : Cahiers de l'admin :Sécuriser un réseau Linux
par Bernard Boutherin et Benoit Delaunay.

[2] : Sources CNAS direction générale département d'informatique.

[3] : www.commentcamarche.net

[4] : www.wekipidia.org

[5] : www.microsoft.com

[6] : www.hsc.fr

[7] : O'Reilly - Building Internet Firewalls.

[8] : O'reilly - Cisco Cookbook – 2003.

[9]: Hacking Exposed - Network Security Secrets & Solutions, 2Nd Edition.

[10] : UML and Data Modeling

[11] : UML Tutorial Complex Transitions.

[12] : UML - Understanding Use Case Modeling.

[13] : http://www.agnitum.com

[14]: Dictionnaire informatique.

ANNEXE

API de filtrages[5] :

Voici l'ensemble des API de filtrages (Tirés de la MSDN).

PfCreateInterface :

La fonction **PfCreateInterface** crée une nouvelle interface .Utiliser cette interface pour commander l'ajouter et la suppression des filtres.

```
DWORD PfCreateInterface(
  DWORD dwName,
  PFFORWARD_ACTION inAction,
  PFFORWARD_ACTION outAction,
  BOOL bUseLog,
  BOOL bMustBeUnique,
  INTERFACE_HANDLE* ppInterface
);
```

Paramètres :

dwName
Indique le nom de l'interface. Une valeur nulle indique une nouvelle, unique interface. N'importe quelle autre valeur est une interface potentiellement partagée.
Le paramètre bMustBeUnique peut transformer une interface partagée en unique. Cependant, employer le bMustBeUnique de cette façon peut faire échouer la fonction.

inAction
Indique une action par défaut pour un paquet en entrée. Ce champ peut avoir l'une des valeurs suivantes.

Valeur	Signification
PF_ACTION_FORWARD	accepter le paquet.
PF_ACTION_DROP	supprimer le paquet

outAction

Indique une action par défaut pour un paquet en sortie. Ce champ peut avoir l'une des valeurs suivantes.

Valeur	Signification
PF_ACTION_FORWARD	accepter le paquet.
PF_ACTION_DROP	supprimer le paquet.

BUseLog
Spécifie si la journalisation doit être active ou non.

bMustBeUnique
Indique si l'interface est unique ou partagée. Si ce membre est VRAI, cette interface est unique, c.-à-d., elle ne peut pas être partagée.

PpInterface
Pointeur sur pointeur, si l'opération de création de l'interface a réussit alors ppInterface pointera sur le handle de l'interface.

Valeur De retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.
Si la fonction échoue, employer FormatMessage pour obtenir le code du message pour l'erreur retournée.

Remarques :

Une interface peut être unique à un processus ou partagé. Si une interface est partagée, d'autres processus peuvent ajouter ou enlever des filtres.

PfAddGlobalFilterToInterface

La fonction **PfAddGlobalFilterToInterface** ajoute un filtre global à l'interface indiquée.

```
DWORD PfAddGlobalFilterToInterface(
  INTERFACE_HANDLE pInterface,
  GLOBAL_FILTER gfFilter
);
```

Paramètres :

pInterface
Handle de l'interface.

gfFilter
Indique le filtre global a ajouté à l'interface.

Valeurs De retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.
Si la fonction échoue, employer FormatMessage pour obtenir la code de message pour l'erreur retournée.

Remarques
Le filtre global agit à travers tous les filtres sur l'interface.

PfBindInterfaceToIPAddress :

La fonction **PfBindInterfaceToIPAddress** associe une interface à l'index de pile IP ayant l'adresse indiquée.

```
DWORD PfBindInterfaceToIPAddress(
  INTERFACE_HANDLE pInterface,
  PFADDRESSTYPE pfatType,
  PBYTE IPAddress
);
```

Paramètres :

pInterface
[in] Indique le handel de l'interface associer à l'index de la pile IP.
pfatType
[in] Indique le type d'adresse pour l'interface. (IPV4 ou bien IPV6)
IPAddress
[in] L'adresse IP de l'interface.

Valeurs De retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.
Si la fonction échoue, employer FormatMessage pour obtenir le code du message pour l'erreur retournée.

PfAddFiltersToInterface :

La fonction **PfAddFiltersToInterface** ajoute un filtre à l'interface.

```
DWORD PfAddFiltersToInterface(
  INTERFACE_HANDLE ih,
  DWORD cInFilters,
  PPF_FILTER_DESCRIPTOR pfiltIn,
  DWORD cOutFilters,
  PPF_FILTER_DESCRIPTOR pfiltOut,
  PFILTER_HANDLE pfHandle
);
```

Paramètres :
ih
[in] Indique le handle de l'interface.
cInFilters
[in] Indique le nombre de filtre en entrée.
pfiltIn
[in] Pointeur sur un tableau de filtres en entrée.
cOutFilters
[in] indique le nombre de filtre en sortie.
pfiltOut
[in] Pointeur sur un tableau de filtres en sorties.
pfHandle
[out] Pointeur sur un tampon qui contient un tableau de filtres. Le développeur peut mettre ce champ à NULL.

Valeurs De retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.
Si la fonction échoue, la valeur de retour est l'une des codes d'erreurs suivants.

Valeur	Signification
PFERROR_NO_FILTERS_GIVEN	Aucune description de filtre n'a été fournie.
Autre	Utilisation de **FormatMessage** pour obtenir le code du message pour l'erreur retournée.

Remarques :

Un filtre inverse la règle de traitement par défaut de l'interface, c.-à-d., la règle qui a été indique lors de l'appel a **PfCreateInterface**.

PfRemoveFiltersFromInterface :

La fonction **PfRemoveFiltersFromInterface** supprime un filtre de l'interface.

```
DWORD PfRemoveFiltersFromInterface(
  INTERFACE_HANDLE ih,
  DWORD cInFilters,
  PPF_FILTER_DESCRIPTOR pfiltIn,
  DWORD cOutFilters,
  PPF_FILTER_DESCRIPTOR pfiltOut
);
```

Paramètres :

ih
[in] Indique le handle de l'interface.
cInFilters
[in] Indique le nombre de filtre en entrée.
pfiltIn
[in] Pointeur sur un tableau de filtres en entrée.
cOutFilters
[in] indique le nombre de filtre en sortie.
pfiltOut
[in] Pointeur sur un tableau de filtres en sorties.

Valeurs De retour

Si la fonction réussit, la valeur de retour est NO_ERROR.

Valeur	Signification
PFERROR_NO_FILTERS_GIVEN	Aucune description de filtre n'a été fournie.
Autre	Utilisation **FormatMessage** pour obtenir le code du message pour l'erreur retournée.

PfUnBindInterface

La fonction de PfUnBindInterface libérer l'interface de la pile.

DWORD PfUnBindInterface(
INTERFACE_HANDLE *pInterface*
);

Paramètres :

pInterface
[in] Indique l'interface a libéré de la pile.

Valeurs de retours :

Si la fonction réussit, la valeur de retour est NO_ERROR.

Si la fonction échoue, employer **FormatMessage** pour obtenir la code de message pour l'erreur retournée.

Remarques :

Libérer l'interface ne détruit ni l'interface ni les filtres attaché a elle.

PfDeleteInterface

La fonction **PfDeleteInterface** supprime une interface qui a été précédemment crée par la fonction **PfCreateInterface** .

```
DWORD PfDeleteInterface(
  INTERFACE_HANDLE pInterface
);
```

Paramètres :

pInterface

[in] Indique un ensemble d'interfaces obtenue à partir d'un appel a **PfCreateInterface**.

Valeurs de retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.

Si la fonction échoue, employer FormatMessage pour obtenir le code du message de l'erreur retournée.

PfGetInterfaceStatistics

La fonction **PfGetInterfaceStatistics** présente des statistiques sur l'interface indiquée. Cette fonction peut également indiquée des statistiques des filtres liés à cette interface.

```
DWORD PfGetInterfaceStatistics(
  INTERFACE_HANDLE pInterface,
  PPF_INTERFACE_STATS ppfStats,
  PDWORD pdwBufferSize,
  BOOL fResetCounters
);
```

Paramètres :

pInterface [in] Handle de l'interface.

ppfStats Pointeur sur le tampon qui va recevoir les statistiques de l'interface.

Si l'application appelante exige seulement les statistiques pour une seule interface, ce pointeur devrait être de taille égale à la Structure **de PF_INTERFACE_STATS**.

Si l'utilisateur fournit un pointeur qui est plus petit que cette taille, **PfGetInterfaceStatistics** renvoie PFERROR_BUFFER_TOO_SMALL, et le

pointeur *pdwBufferSize* contient une taille égale a la taille de la structure **PF_INTERFACE_STATS.**

Si l'utilisateur a besoin des statistiques de l'interface et de ses filtres associés, le pointeur devrait être d'une taille plus grande que **PF_INTERFACE_STATS.** Si le pointeur n'est pas assez grand, **PfGetInterfaceStatistics** renvoie l'erreur ERROR_INSUFFICIENT_BUFFER, et *pdwBufferSize* pointe sur une variable de type **DWORD** qui contient la taille qu'il faut allouer pour contenir les statistiques de l'interface et de ses filtres.

pdwBufferSize
[in] Pointeur sur une variable de type **DWORD** qui indique la taille du tampon pointé par la structure *ppfStats*.

fResetCounters
[in] Indique si les compteurs statistiques doivent être réinitialise ou non, si ce paramètre est à vrai alors les compteurs seront réinitialise.

Valeurs De retour

Si la fonction a réussit, la valeur de retour est NO_ERROR.

Si la fonction a échoué, la valeur de retour est l'une des codes d'erreur suivants.

Valeur	Signification
ERROR_INSUFFICIENT_BUFFER	Cette erreur est spécifique à **PfGetInterfaceStatistics** et signifie que le pointeur fourni par utilisateur est trop petit pour les filtres.
PFERROR_BUFFER_TOO_SMALL	Cette erreur est spécifique à

	PfGetInterfaceStatistics et signifie que le pointeur d'utilisateur est trop petit même pour les statistiques d'interface. La taille retournée est la taille des statistiques de l'interface, mais n'inclut pas l'espace des filtres.
Autre	Utilisation **FormatMessage** pour obtenir le code du message de l'erreur retournée.

Remarques :

L'utilisateur peut appeler **PfGetInterfaceStatistics** deux fois. Au départ, pour obtenir la taille correct du pointeur, puis une deuxième fois pour rechercher les statistiques. Si le l'utilisateur appelle **PfGetInterfaceStatistics** deux fois pour une interface partagée, le deuxième appel peut échouer.

Cette erreur se produit si d'autres processus partageant l'interface dans l'intervalle entre les deux appels. Ce type d'erreur ne devrait pas se produire pour une interface UNIQUE.

PfMakeLog

La fonction **PfMakeLog** crée un journal pour une ou plusieurs interfaces.

```
WORD PfMakeLog(
  HANDLE hEvent
);
```

Paramètres :

hEvent

[in] Handle sur un objet événement. L'utilisateur peut utiliser cet objet d'événement pour obtenir des notifications quand un nombre spécifique de bytes ont accosté sur le tampon du journal, ou quand un certain nombre

d'entrées ont été créées dans la notification. Pour plus d'information, allez voir **PfSetLogBuffer** .

Valeurs De retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.

Si la fonction échoue, employer **FormatMessage** pour obtenir le code du message de l'erreur retournée.

Remarques :

Une seule notification existe .La notification peut être utiliser par plusieurs interfaces.
La notification d'interface doit être créée avant la création des interfaces avec lesquelles elle est employée. Il n'est pas possible d'associer une notification à une interface déjà existante.

PfSetLogBuffer

La fonction **PfSetLogBuffer** échange le tampon journal courant par un nouveau tampon.

```
DWORD PfSetLogBuffer(
  PBYTE pbBuffer,
  DWORD dwSize,
  DWORD dwThreshold,
  DWORD dwEntries,
  PDWORD pdwLoggedEntries,
  PDWORD pdwLostEntries,
  PDWORD pdwSizeUsed
);
```

Paramètres :

pbBuffer
[in] Pointeur sur un nouveau tampon.
dwSize
[in] Spécifie la taille en bytes du nouveau tampon .

dwThreshold
Spécifie le seuil en bytes pour déclencher un événement journal.
dwEntries
[in] Spécifie le nombre des entrées dans le tampon journal qui causera le déclenchement d'un objet événement qui sera signalé.
pdwLoggedEntries
[out] Pointeur sur une variable de type DWORD qui recevra le nombre d'entrées dans l'ancien tampon.
pdwLostEntries
[out] Pointeur sur une variable de type DWORD qui recevra le nombre d'entrées qui ne sont pas mis dans l'ancien tampon.
pdwSizeUsed
[out] Pointeur sur une variable de type DWORD qui recevra le nombre de bytes utiliser dans l'ancien tampon.

Valeurs De retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.

Si la fonction échoue, employer **FormatMessage** pour obtenir le code du message de l'erreur retournée.

PfDeleteLog

La fonction **PfDeleteLog** désactive la journalisation sur touts les interfaces au quelles elle est rattachée.
Le log est supprimé une fois que toutes les interfaces rattachées à ce dernier sont supprimées.

DWORD PfDeleteLog(void);

Paramètres

Cette fonction n'a pas de paramètres.

Valeurs De retour :

Si la fonction réussit, la valeur de retour est NO_ERROR.

Si la fonction échoue, employer **FormatMessage** pour obtenir le code du message de l'erreur retournée.

PF_FILTER_DESCRIPTOR

La structure **PF_FILTER_DESCRIPTOR** contient des informations qui définie le filtre de paquet.

```
typedef struct _PF_FILTER_DESCRIPTOR {
  DWORD dwFilterFlags;
  DWORD dwRule;
  PFADDRESSTYPE pfatType;
  PBYTE SrcAddr;
  PBYTE SrcMask;
  PBYTE DstAddr;
  PBYTE DstMask;
  DWORD dwProtocol;
  DWORD fLateBound;
  WORD wSrcPort;
  WORD wDstPort;
  WORD wSrcPortHighRange;
  WORD wDstPortHighRange;
} PF_FILTER_DESCRIPTOR,
*PPF_FILTER_DESCRIPTOR;
```

Membres :

dwFilterFlags

En ce moment un seul flag est supporté par ce champ.

FD_FLAGS_NOSYN

dwRule

Spécifie la règle du filtre.

pfatType

Spécifie le type des adresses du filtre (IPV4 ou IPV6).

SrcAddr

Spécifie l'adresse source du filtre de paquets.

SrcMask

Spécifie le masque réseau source du filtre de paquets.

DstAddr

Spécifie l'adresse destination du filtre de paquets.

DstMask

Spécifie le masque réseau destination du filtre de paquets.

dwProtocol

Spécifie le protocole utilisé par le filtre de paquets.
Ce champ peut prendre une des valeurs suivantes :

Valeur	Description
FILTER_PROTO_ANY	Touts les protocoles
FILTER_PROTO_ICMP	Internet Control Message Protocol(ICMP)
FILTER_PROTO_TCP	Transmission Control Protocol(TCP)
FILTER_PROTO_UDP	User Datagram Protocol(UDP)

fLateBound

Spécifie les informations sur les adresses qui sont mis a jour quand le filtre a rebondit. Ce champ peut être une combinaison des valeurs suivantes :

LB_SRC_ADDR_USE_SRCADDR_FLAG
LB_SRC_ADDR_USE_DSTADDR_FLAG
LB_DST_ADDR_USE_SRCADDR_FLAG
LB_DST_ADDR_USE_DSTADDR_FLAG

wSrcPort

Spécifie le port source du filtre de paquet.

wDstPort

Spécifie le port destination du filtre de paquet.

wSrcPortHighRange

Spécifie la borne supérieure de port source du filtre.

DstPortHighRange

Spécifie la borne supérieure de port destination du filtre.

PF_INTERFACE_STATS

La structure **PF_INTERFACE_STATS** contient des statistiques sur l'interface.

```
typedef struct _PF_INTERFACE_STATS {
  PVOID pvDriverContext;
  DWORD dwFlags;
  DWORD dwInDrops;
  DWORD dwOutDrops;
  PFFORWARD_ACTION eaInAction;
```

```
 PFFORWARD_ACTION eaOutAction;
 DWORD dwNumInFilters;
 DWORD dwNumOutFilters;
 DWORD dwFrag;
 DWORD dwSpoof;
 DWORD dwReserved1;
 DWORD dwReserved2;
 LARGE_INTEGER liSYN;
 LARGE_INTEGER liTotalLogged;
 DWORD dwLostLogEntries;
 PF_FILTER_STATS FilterInfo[1];
} PF_INTERFACE_STATS,
*PPF_INTERFACE_STATS;
```

Membres :

pvDriverContext
Ce champ n'est pas actuellement utilisé.

dwFlags
Ce champ n'est pas actuellement utilisé.

dwInDrops
Spécifie le nombre de paquets en entrée qui sont supprimés.

dwOutDrops
Spécifie le nombre de paquets en sortie qui sont supprimés.

eaInAction
Spécifie l'action par défaut en entrée.

eaOutAction
Spécifie l'action par défaut en sortie.

dwNumInFilters
Spécifie le nombre de filtres en entrée.

dwNumOutFilters
Spécifie le nombre de filtres en sortie.

dwFrag
Spécifie l'état du filtre global « vérification des fragments ».

dwSpoof
Spécifie l'état du filtre global « vérification de l'adresses destination ».

dwReserved1
Réserver, Ce champ doit être a zéro.

dwReserved2
Réserver, Ce champ doit être a zéro.

liSYN
Spécifie le nombre de paquets synchrones rejetés.

liTotalLogged

Spécifie le nombre de paquets logués .

dwLostLogEntries

Spécifie le nombre de paquets perdus a cause des problèmes de tampons.

FilterInfo

Spécifie un tableau d'éléments de type **PF_FILTER_STATS.**
Le tableau contient un élément pour chaque filtre associe a l'interface.
Chaque élément contient une description du filtre et le nombre de paquets filtrés par ce filtre.

GetIfTable

La fonction **GetIfTable** fonction retrouve le MIB-II interface table.

DWORD GetIfTable(
PMIB_IFTABLE *pIfTable*,
PULONG *pdwSize*,
BOOL *bOrder*
);

Paramètres :

pIfTable

[out] Pointeur sur le tampon qui recevra la table **MIB_IFTABLE.**

pdwSize

[in, out] En entrée, spécifie la taille du tampon pointe sur le paramètre

pIfTable .

En sortie, si le tampon n'est pas aussi large pour contenir la table d'interface, la fonction met dans ce paramètre la taille qu'il faut.

bOrder

[in] Spécifie l'ordre applique sur la table des interfaces , si ce paramètre est à TRUE c'est-à-dire que la table sera ordonnée.

Return code	Description
ERROR_INSUFFICIENT_BUFFER	Le tampon qui pointe sur le paramètre *pIfTable* est insuffisant.
ERROR_INVALID_PARAMETER	Le paramètre *pdwSize* est NULL, ou bien **GetIfTable** n'est pas accessible en écriture.

ERROR_NOT_SUPPORTED	Cette fonction n'est pas supporte par le système d'exploitation.

Composants INDY :

L'IDE C++BUILDER contient plusieurs onglets contenant des composants INDY.
Notre application utilise trois composants INDY :

- TIdDNSResolver : pour la résolution de noms de domaines.
- TidMessage : encapsule les messages Internet.
- TidSMTP : permet d'envoyer des emails sur un serveur mail en utilisant le protocole SMTP.

- Exemple d'utilisation du composant «TIdDNSResolver » :

```
//Nom du serveur DNS de votre fournisseur Internet
DNSResolver->Host = ns->Text; //example : ns.eepad.dz.
DNSResolver->ClearVars();//Initialisation des variables du composant.
DNSResolver->DNSAnList->Clear();
DNSResolver->Active = true;
DNSResolver->DNSHeader->QDCount = 1;// Représente le nombre de questions
                                    dans une requête DNS.
DNSResolver->DNSQDList->Clear();//Initialise la requête DNS.
TQuestionItem * QT = DNSResolver->DNSQDList->Add();
QT->QName = edDomain->Text;//Nom de domaine pour lequel en recherche son IP.
QT->QType = cA;//paramètre de la requête (récupérer l'adresse IP).
QT->QClass = cIN; //Chercher le domaine sur le réseau Internet.
DNSResolver->ResolveDNS();//Resolution de nom.
```

- Exemple d'envoie d'émail :

```
IdMessage1->Clear();//Efface l'entête et le corps du message.
IdMessage1->From->Text = "firewallStatique@securite.fr";//Adresse émail de l'expéditeur.
IdMessage1->ReplyTo->EMailAddresses =
        "infoabdelkrim@yahoo.fr";//Adresse émail où le destinataire pourra répondre.
IdMessage1->Recipients->EMailAddresses = Email->Text; //Adresse émail du destinataire.
IdMessage1->Date = Date(); //Date de l'envoi du message.
IdMessage1->Subject = "Test envoyé par le Firewall Statique 2006";//Sujet du message.
IdMessage1->Priority = mpNormal; //Priorité du message.

//Attacher le fichier journal au message.
TIdAttachment(IdMessage1->MessageParts, currentDirectoy + "\\Journal.html");
```

```
//Connexion au serveur smtp.
IdSMTP1->AuthenticationType = atLogin; //Type d'authentification.

IdSMTP1->Host = smtp->Text;//serveur SMTP (par exemple pour Free : smtp.free.fr).

IdSMTP1->Port = port->Text ; //Port d'écoute du serveur SMTP (par exemple pour Free : 25).

IdSMTP1->UserId = identifiant->Text; //Nom d'utilisateur (exemple : infoabdelkrim)

IdSMTP1->Password =  motDePasse->Text;//Mot de passe.
try
{
     IdSMTP1->Connect(); //Connexion.
     if(IdSMTP1->Authenticate())
     {
          IdSMTP1->Send(IdMessage1); //Envoi du message.
          IdSMTP1->Disconnect(); //Déconnexion.
     }
}
catch(Exception &ex)
{
    IdSMTP1->Disconnect(); //Déconnexion.
    ShowMessage("Echec de l'envoie de l'email!!!");
     return;
}
```