



**UE 28 / UE 31**


# **Détection d'Intrusion Réseau et Système**

---

**Vincent GLAUME**



➤ **Détection d'Intrusion**

- 
- Les grands principes
  - La détection d'intrusion réseau et système
  - Quelques dérivés : IPS, honeypots

# Systemes de Détection d'Intrusion

---

**NIDS, Honeypots, IPS**

## ➤ **Les principes généraux : introduction aux IDS**

- qu'est-ce qu'un IDS ?
- les grandes familles d'IDS

## ➤ **Détection d'intrusion**

- méthodes de détection
- Systèmes de Détection d'Intrusion Réseau
- Systèmes de Détection d'Intrusion Système
- Comparatif NIDS/HIDS

## ➤ **Exploitation des IDS**

## ➤ **Des IDS particuliers : Honeypots et IPS**

- Honeypots, ou pots de miel
- IPS : Intrusion Prevention Systems

## ➤ **Conclusion**



# Systemes de Détection d'Intrusion

---

## Principes généraux : introduction aux IDS



- **Qu'est-ce qu'une intrusion ?**
- **Les IDS, pourquoi ?**
- **Utilisation des IDS : objectifs divers**
- **Comment détecter une intrusion ?**
- **Et après la détection ?**
- **Normalisation : CIDF**
- **Un peu de vocabulaire**

# Qu'est-ce qu'une intrusion ?

---



- **Un IDS est un système de détection d'intrusion : que considère-t-on comme une intrusion ?**
  - Il s'agit d'une action ayant pour but de compromettre, sur un système d'information :
    - la confidentialité
    - l'intégrité
    - la disponibilité
    - l'imputabilité
  - Cette action peut :
    - avoir réussi
    - avoir échoué
    - être en cours
    - être une simple phase préparatoire
  - Le système d'information visé peut être :
    - un réseau
    - une machine, un système d'exploitation
    - une application

# Les IDS, pourquoi ?

---

## ➤ Pourquoi chercher des signes d'intrusion si nous sommes protégés ?



- principe du « ceinture et bretelles » : protection **ET** détection
  
- quelle confiance peut-on avoir en un système de protection ?
  - problèmes de configuration
  - matériel inadapté
  - erreur humaine
  - backdoor
  
- périmètre : on opte souvent pour une protection « en bordure » du système, mais la détection peut se faire en son cœur
  - ex : pare-feu protégeant le LAN d'Internet, et NIDS sur le LAN contrôlant l'activité des utilisateurs locaux.





- **Rôle global dans une architecture de sécurité :**
  - support pour les défenses
  - contrôle des politiques de sécurité
  - prévention quand son déploiement est connu (intimidation ?)
  
- **Favoriser la mise en place de défenses adaptées**
  - détection d'attaques inconnues
  - détection de phases préliminaires d'attaques
  
- **Assainir un système**
  - l'IDS peut mettre en avant certains comportements « polluants » du système : aide à l'administration classique
  
- **Rôle statistique**
  - source d'informations pour la hiérarchie
  - vision des menaces courantes

# Comment détecter une intrusion ?

---



- **La détection d'intrusion consiste à mettre en oeuvre les moyens permettant de mettre en évidence d'éventuelles intrusions :**
  - en surveillant l'activité du système (réseau, machine ou application) afin d'en recueillir les différents événements
  - en analysant ces événements à la recherche de signes suspects
- **Les IDS ont pour but d'automatiser ces phases de collecte d'information et d'analyse.**



- **Comprendre l'attaque en cours**

- recueillir le maximum d'information :

- cible(s)

- source(s)

- procédé

- **Archiver, tracer : corrélation avec d'autres événements**

- **Préparer une réponse :**

- sur le court terme : black-listage, starrage...

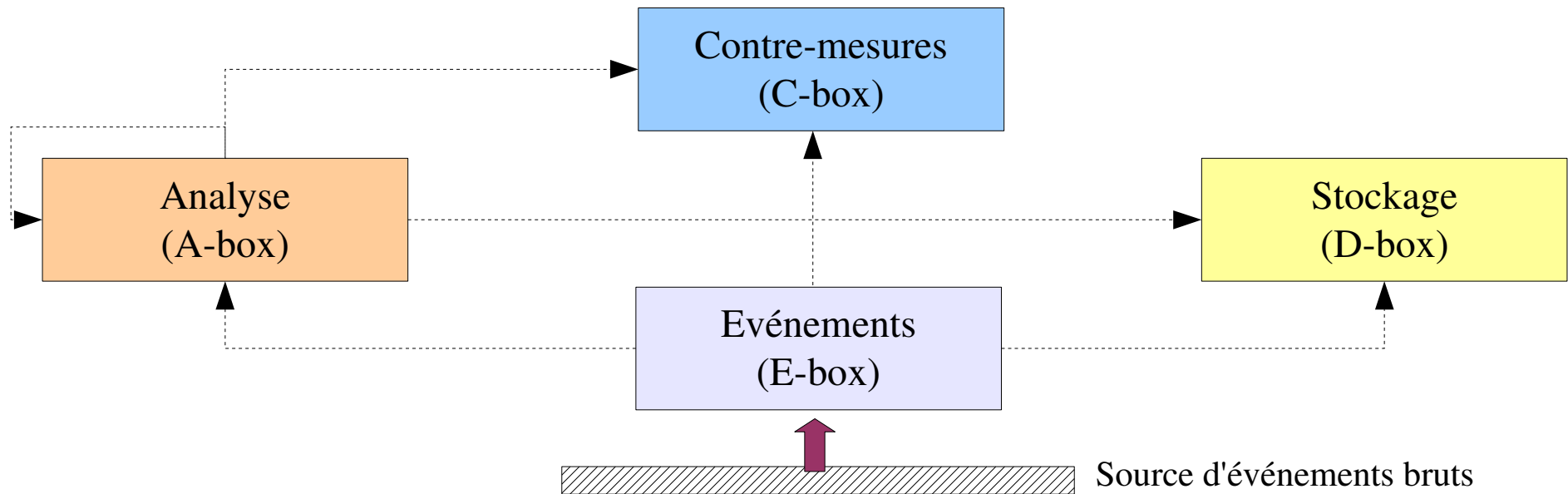
- sur le long terme : application de patches, actions en justice...

## ➤ CIDF : Common Intrusion Detection Framework



## ➤ Ce modèle définit les composants d'un IDS :

- générateur d'événements : E-box
- analyseur d'événements : A-box
- stockage d'information : D-box
- contre-mesure : C-box





➤ **Faux-positif :**

- fausse alerte levée par l'IDS

➤ **Faux-négatif :**

- attaque (au sens large) qui n'a pas été repérée par l'IDS

➤ **Evasion :**

- technique utilisée pour dissimuler une attaque et faire en sorte qu'elle ne soit pas décelée par l'IDS

➤ **Sonde :**

- composant de l'architecture IDS qui collecte les informations brutes



- **Discrimination sur la méthode de détection**
- **Discrimination sur le système d'information surveillé**
- **Discrimination sur le traitement**

## ➤ Détection par signature

- basée sur la reconnaissance de schémas connus (*pattern matching*)
- utilisation d'expressions régulières
- les signatures d'attaques connues sont stockées dans une base, et chaque événement est comparé au contenu de cette base
  - si un événement (paquet réseau, log système) correspond à une signature de la base, l'alerte correspondante est levée par l'IDS
- l'attaque doit être connue pour être détectée
- il y a peu de faux-positifs pour une signature pertinente

## ➤ Détection par anomalie

- basée sur un comportement « normal » du système
- une déviation par rapport à ce comportement est considérée comme suspecte
- le comportement doit être modélisé : on définit alors un profil
- une attaque peut être détectée sans être préalablement connue

## ➤ Réseau

- NIDS : Network Intrusion Detection System
- principe : contrôler le trafic réseau
  - contenu
  - volume
- une sonde permet de surveiller plusieurs machines
- furtivité

## ➤ Système

- HIDS : Host-based Intrusion Detection System
- principe : contrôler l'activité système
  - logs
  - fichiers
  - processus
- une machine nécessite une sonde propre

## ➤ Application

- ~~sous-genre du HIDS~~





## ➤ **Traitement en temps réel**

- la sonde (E-box) envoie instantanément les événements perçus au moteur d'analyse (A-box)
- typique des NIDS
- facilite une réponse à incident rapide

## ➤ **Traitement en différé**

- la sonde (E-box) accumule les événements et les envoie au moteur d'analyse (A-box) selon des critères prédéfinis (de durée, de quantité de données...)
- plus commun sur les HIDS
- peut permettre de gagner en pertinence
- réponse à incident retardée

## ➤ **Le choix de la méthode va dépendre :**

- de la politique en place : peut-on se permettre un délai dans l'analyse
- des performances : que gagne-t-on à différer le traitement
- des possibilités techniques offertes par les outils faisant office de



# Systemes de Détection d'Intrusion

---

## Détection d'intrusion : méthodes



- **Détection d'intrusion par signature**
- **Détection d'intrusion par anomalie**
- **Que choisir ?**

# Détection d'intrusion par signature 1/2

---



- **Méthode la plus simple, basée sur une reconnaissance de caractères**
  - **si** Evenement **matche** Signature **alors** Alerte
- **Facile à implémenter, pour tout type d'IDS**
- **L'efficacité de ces IDS est liée à la gestion de la base de signatures**
  - mise à jour
  - nombre de règles
  - signatures suffisamment précises
- **Exemples :**
  - trouver le motif `/winnt/system32/cmd.exe` dans une requête HTTP
  - trouver le motif `FAILED su for root` dans un log système

## ➤ **Avantages**

- simplicité de mise en oeuvre
- rapidité du diagnostic
- précision (en fonction des règles)
- identification du procédé d'attaque
  - procédé
  - cible(s)
  - source(s)
  - outil ?
- facilite donc la réponse à incident

## ➤ **Inconvénients**

- ne détecte que les attaques connues de la base de signatures
- maintenance de la base
- techniques d'évasion possibles dès lors que les signatures sont connues

- **On parle aussi de détection d'intrusion comportementale**



- **Modélisation du système : création d'un profil *normal***

- phase d'apprentissage pour définir ce profil
- la détection d'intrusion consistera à déceler un écart suspect entre le comportement du réseau et son profil
- Exemples de profil : volume des échanges réseau, appels systèmes d'une application, commandes usuelles d'un utilisateur...

- **Repose sur des outils de complexités diverses**

- seuils
- statistiques
- méthodes probabilistes
- réseaux de neurones
- machines à support de vecteurs

## ➤ **Avantages**


- permet la détection d'attaques non connues a priori
- facilite la création de règles adaptées à ces attaques
- difficile à tromper

## ➤ **Inconvénients**

- les faux-positifs sont relativement nombreux
- générer un profil est complexe :
  - durée de la phase d'apprentissage
  - activité saine du système durant cette phase ?
- en cas d'alerte, un diagnostic précis est souvent nécessaire pour identifier clairement l'attaque

# Que choisir ?

---

- 
- **L'immense majorité des IDS disponibles, commerciaux ou libres, sont basés sur de la détection par signature.**
  - **La détection par anomalie relève à l'heure actuelle davantage du domaine de la recherche.**
  - **Les bases de signatures sont souvent assez proches, ie visent à détecter les mêmes types d'attaques ; en revanche les méthodes de détection par anomalies peuvent toucher à des aspects très variés.**
  - **Une approche comportementale bien adaptée à son environnement peut s'avérer très efficace mais est souvent complexe à mettre en place.**
  - **Utiliser des méthodes de détection comportementale comme support à de la détection d'intrusion par signature permet d'améliorer les 2 méthodes.**



# Systemes de Détection d'Intrusion

---

**NIDS : Network Intrusion Detection Systems**



- **Objectifs des NIDS**
- **Particularités des NIDS**
- **Capture réseau**
- **Analyse : les grandes lignes**
- **Analyse protocolaire**
- **Analyse des données**
- **Premiers obstacles (couches 3 et 4)**
- **Techniques d'évasion (couches 3 et 4)**
- **Gérer les mécanismes réseau**
- **Techniques d'évasion (couches applicatives)**
- **Limitations des NIDS**
- **Quelques outils**
- **Analyse de flux**
- **Apports de l'analyse de flux**
- **Outils d'analyse de flux**

## ➤ Les NIDS sont utilisés pour contrôler l'activité d'un réseau

- de taille variable
  - du simple LAN sans routage
  - au réseau d'entreprise comptant plusieurs milliers de machines et plusieurs centaines de sous-réseaux
- en des points stratégiques
  - en périphérie du réseau
    - contrôle des flux échangés avec l'extérieur
  - au coeur du réseau
    - contrôle du comportement des utilisateurs internes
    - recherche d'éventuelles intrusions
- par des techniques variées
  - étude par paquets
    - sans états
    - avec états (reconstruction des flux)
  - analyse de flux



## ➤ **Furtivité**

- différents moyens techniques permettent un accès complètement passif aux données réseau, soit une capture difficilement repérable

## ➤ **Globalisation de la surveillance**

- une sonde unique permet la surveillance d'un réseau entier, donc d'un ensemble de machines
- se placer stratégiquement permet d'optimiser la surveillance
  - au niveau d'une passerelle pour les contrôles périphériques
  - au niveau d'un équipement type répéteur (*switch*) pour un contrôle sur un réseau local
- il est ensuite possible de centraliser les alertes pour corrélérer les informations recueillies sur les différentes zones

## ➤ **Performances**

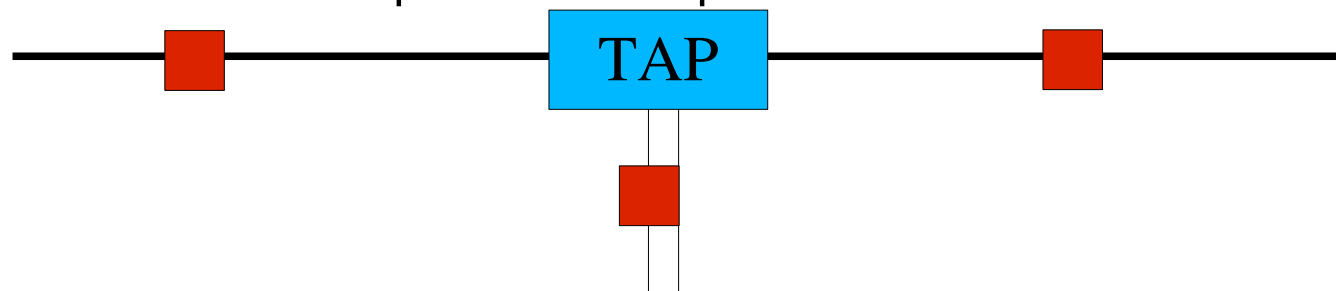
- la sonde doit être adaptée à la vitesse du réseau surveillé

## ➤ Le but d'une sonde NIDS est de capturer le trafic réseau

- de la façon la plus furtive possible
- avec le moins de pertes possibles

## ➤ Moyens techniques

- *hub* : les paquets arrivant sur un port sont recopiés sur tous les autres
  - de moins en moins utilisé (coût, sécurité, performances)
- répéteurs (*switches*) permettant la copie de ports
  - configurables pour recopier tout ou partie du trafic sur un port donné
- boîtiers TAP (*Test Access Port*)
  - le boîtier se place en coupure



## ➤ On distingue 2 types d'analyse réseau

- par paquet : vision macroscopique, nombreuses possibilités
  - la plus courante, et celle que nous développerons le plus
- de flux : vision globale de l'activité du réseau

## ➤ Pour l'analyse de paquets on peut s'intéresser aux différentes couches, chacune étant en mesure de fournir des informations pertinentes

- anomalies au niveau de l'en-tête d'un paquet
  - combinaisons exotiques des flags TCP
  - mauvaise somme de contrôle
- anomalies au niveau du comportement du protocole
  - incohérences des numéros de séquence et d'acquittement TCP
- couches applications malformées
  - négociation SMTP suspecte
- données suspectes



## ➤ ARP

- adresse nulle
- adresse source en **ff:ff:ff:ff:ff:ff**
- gratuitous ARP

## ➤ IP

- mauvais checksum
- mauvaise taille de paquet
- adresse incohérente

## ➤ UDP

- mauvais checksum

## ➤ TCP

- mauvais checksum

---

UE 07 ➤ taille de l'en-tête incorrecte



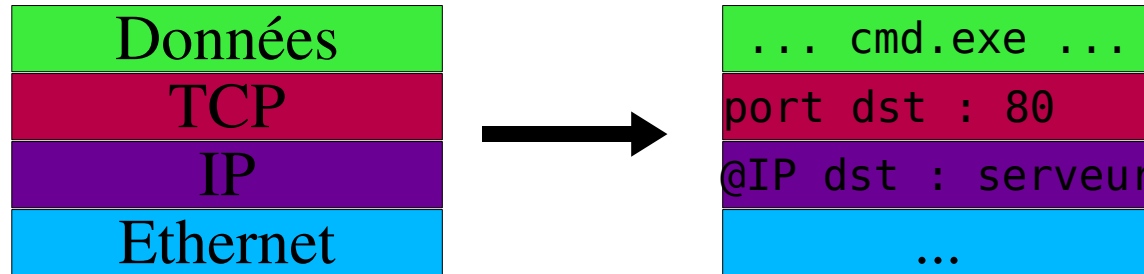
- **Il s'agit ici de rechercher dans les données (*payload*) des chaînes de caractères suspectes, correspondant à des attaques connues.**
  - premier tri selon le protocole et les ports impliqués : quel protocole applicatif a-t-on ?
  - ensuite, on s'intéresse aux attaques connues propres au protocole applicatif en question
  
- **Exemple :**
  - si le protocole de couche 4 est **TCP**
  - si le port sur lequel se fait la connexion est le **port 80**
    - alors le protocole applicatif est **HTTP**
      - si de plus la requête, dans l'url, contient la chaîne **cmd.exe**
        - alors il s'agit vraisemblablement d'une attaque
  - une alerte sera alors levée au vu d'une requête type :  
`http://www.serveur.net/chemin/vers/cmd.exe`



# Premiers obstacles (couches 3 et 4)



- Lors d'une analyse par paquet basique, on considère que la chaîne malicieuse sera contenue entièrement dans un seul paquet :



- Mais la fragmentation IP ou les connexions TCP complexifient l'affaire

- fragmentation IP



- Connexion TCP



# Techniques d'évasion (couches 3 et 4)

## ➤ Ici, on cherche

- soit à échapper à la vigilance du NIDS
- soit à le noyer sous une nuée de faux-positifs

## ➤ **La machine destinatrice du paquet ne verra pas systématiquement la même chose que le NIDS, surtout si celui ne reconstitue aucun flux !**

- découpage des données, qui seront reconstruites par le destinataire
  - connexion TCP
  - fragmentation IP
- superposition de données
  - principe identique au précédent avec chevauchement d'octets : qui prévaut alors ? Dépendant du système d'exploitation cible !
- paquets malformés (mauvais checksum)
  - la machine rejettera le paquet ; quid du NIDS ?
- numéros de séquences erronés
- TTL faible
  - un faible TTL peut faire en sorte que le paquet soit vu par le NIDS



- **Descendre dans les mécanismes des couches 3 et 4 semble nécessaire, mais pas forcément suffisant**
  - implémenter les grands principes de TCP
  - reconstituer les flux
  - déterminer au mieux le contenu de la pile TCP/IP des machines surveillées
  
- **Problèmes**
  - opérations très coûteuses
  - cas difficiles à trancher (chevauchements d'octets)
  - difficultés à réaliser un bon paramétrage
  
- **Solutions possibles**
  - normalisation de trafic en entrée du réseau
  - prise de parti sur certains critères
    - exemple : si  $TTL < n$ , on ignore le paquet

# Techniques d'évasion (couches applicatives)

---



- **Le but est encore de leurrer le NIDS en se basant sur son interprétation des données**
  - jouer sur l'encodage des données (unicode, UTF-8...)
  - jouer sur des spécificités d'implémentation d'un protocole applicatif
    - deux serveurs SMTP différents n'accepteront pas nécessairement les mêmes commandes
  
- **Mais il est également possible d'adapter son attaque pour éviter la signature**
  - shellcodes polymorphiques
  - instructions *inutiles* pour remplacer les *NOP*
  
- **Là encore, il faut adapter ses outils d'analyse**
  - complexité supplémentaire
  - charge supplémentaire pour la machine



- **Outre les problèmes découlant d'attaques spécialement adaptées pour échapper ou nuire à l'IDS, la détection d'intrusion réseau est soumise à certaines contraintes.**
  - flux chiffrés : analyse impossible
    - IPsec
    - ssh
    - https
  - performances
    - équipements adaptés au niveau du réseau
    - perte de paquets
    - machine de traitement suffisamment puissante
  - la cible est-elle vulnérable à l'attaque ?
    - plusieurs politiques possibles
      - on cherche à détecter toute attaque
      - on cherche à détecter les attaques potentiellement dangereuses, ie qui risquent d'aboutir



## ➤ Principaux NIDS libres

- snort
- bro
- prelude IDS

## ➤ Certains constructeurs intègrent des modules IDS à leurs équipements (routeurs, firewalls)

## ➤ Outils de test et de contournement

- stick et snot
- whisker
- fragroute

## ➤ A noter que nombre d'outils d'administration réseau peuvent être utilisés comme briques d'une architecture NIDS

- arpswatch
- p0f
- ntop



- **La vision globale implique qu'on n'est plus sujet aux évasions citées précédemment.**
- **L'objet de l'analyse devient le flux**
  - connexion TCP
  - échange UDP
  - flux ICMP
- **Les paramètres à surveiller**
  - qui communique avec qui ?
  - avec quels protocoles ?
  - quelles quantités de données sont échangées ?
  - y a-t-il des spécificités temporelles ?
- **Il s'agit d'une approche comportementale**
  - ~~quelle activité est normale sur le réseau, quelle activité est suspecte ?~~



- **L'analyse de flux peut être menée parallèlement à de la détection d'intrusion réseau classique.**
  
- **Les bénéfices que l'on peut en tirer**
  - détection de canaux cachés
  - utilisation de services sur des ports autres que le port standard
  - activité réseau conséquente en dehors des heures ouvrables
  - existence de flux supposés être interdits par la politique de sécurité



➤ **On en distingue principalement deux :**

➤ Argus

➤ Netflow (Cisco)

➤ **Il existe des outils de conversion plus ou moins aboutis pour jongler entre ces deux formats**

# Systemes de Détection d'Intrusion

---

## **HIDS : Host-based Intrusion Detection Systems**



- **Contrôle d'intégrité**
  - **Objectifs**
  - **Méthodes**
- **Syslog**
  - **Généralités**
  - **Exploitation**
- **Accounting**
- **Audit**

# HIDS : Host-based Intrusion Detection Systems

---



- **Un contrôle d'intégrité va consister à vérifier, par une méthode donnée, que des fichiers déterminés du système n'ont pas été altérés.**
- **Avant de procéder à ce genre de contrôle, il faut déjà sécuriser au maximum les accès à ces fichiers :**
  - droits Unix sur les fichiers, les répertoires parents
  - stockage de données sur des périphériques accessibles physiquement en lecture seule (CD, disquette...)
  - montage de partitions spécifiques en lecture seule
    - peut être contourné en remontant la partition en lecture/écriture avec les droits root...
  - certains systèmes Unix implémentent des types de fichiers sur lesquels on ne peut modifier les données, simplement en ajouter (FreeBSD, NetBSD)
- **On peut alors s'attaquer au contrôle d'intégrité !**



- **Pour détecter des changements sur un fichier, on dispose des méthodes suivantes :**
  - comparaison avec des copies
    - principe :
      - conserver des copies des fichiers critiques
      - périodiquement comparer le fichier sur le système et la copie
    - c'est la méthode la plus apte à détecter s'il y a eu modification, et quelle modification !
    - mais c'est extrêmement lourd à mettre en oeuvre
      - capacités de stockage pour les copies
      - ressources système pour effectuer les comparaisons
    - où stocker la copie ?
      - sur le système lui-même, dans un répertoire dédié
      - sur un périphérique amovible (lecture seule)
      - sur un disque dur pouvant être monté par le système
      - sur le réseau (comparaison avec un export NFS par exemple)
  - il est primordial que la copie ne puisse être altérée !

# HIDS : Host-based Intrusion Detection Systems

---



- comparaison d'attributs du fichier
  - principe :
    - relever des champs caractéristiques des fichiers à contrôler
    - comparer ces champs aux valeurs originales
    - relever les modifications anormales
  - plus léger que la méthode précédente, ce procédé ne permet pas de voir quelles ont été les modifications apportées aux fichiers
  - peut permettre de détecter cependant des modifications autres que celle du contenu du fichier
  - de nombreux paramètres peuvent être pris en compte :
    - propriétaire
    - droits
    - taille
    - dates de création, de dernière modification...
    - numéro d'inode
  - un accès au fichier via le *raw device* ne sera cependant pas décelé
- là encore, les données de référence doivent être fiables !

# HIDS : Host-based Intrusion Detection Systems

---



- comparaison de signatures
  - principe :
    - on calcule une signature de chaque fichier à protéger, avec une fonction cryptographique, type md5
    - on compare périodiquement le checksum courant du fichier avec la valeur originale
  - légère et facile à déployer, cette méthode permet la détection de modifications du contenu d'un fichier, pas de ses attributs
  - nécessite un algorithme suffisamment fort
  - le fichier de référence ne doit pas pouvoir être modifié !

# HIDS : Host-based Intrusion Detection Systems

---



- **Les besoins ne sont pas forcément les mêmes pour tous les fichiers du système que l'on souhaite superviser :**
  - on s'intéressera au contenu des fichiers de configuration
  - mais plutôt aux métadonnées des exécutables
- **Une façon intéressante de procéder consiste à adopter un compromis entre la comparaison des métadonnées et celle des signatures.**
- **Pour cela, il est possible :**
  - d'utiliser des scripts faits à la main, pour des besoins simples
  - d'installer des outils de contrôle d'intégrité implémentant ces fonctionnalités
    - tripwire
      - version Open Source pour Linux
      - version commerciale plus complète
    - aide
      - outil Open Source



# HIDS : Host-based Intrusion Detection Systems

---




- **Syslogd est le démon Unix implémentant la gestion des logs système et noyau.**
- **Il permet la gestion des logs**
  - locaux : utilisation d'un socket Unix (/dev/log)
  - distants : écoute sur le port UDP 514
- **Les messages de logs sont classés**
  - selon une famille (*facility*) : auth, daemon, kern, syslog...
  - puis selon un niveau d'importance (*priority*) : debug, warning, alert, panic...
- **Chaque message envoyé au démon contient la date et le nom de l'hôte l'ayant émis.**
- **Ensuite, le fichier de configuration */etc/syslog.conf* permet de déterminer, selon le couple *facility/priority*, ce qu'il adviendra du message :**
  - envoyer dans un fichier (lequel)
  - afficher en console



- **Pour tester une configuration du démon syslog ou lui envoyer des messages depuis des scripts**
  - logger
    - on spécifie une priorité et éventuellement une famille
    - on envoie alors un message au démon
    - celui-ci le traite alors en fonction de ces paramètres, selon les indications du fichier de configuration
  
- **Pour faciliter l'exploitation simple des logs, on peut mentionner :**
  - logrotate
    - permet l'archivage des logs en renommant et compressant les anciens logs de façon automatique
  - lastlog, wtmp, utmp
    - la commande lastlog et les deux fichiers wtmp et utmp permettent de voir qui est actuellement logué sur le système, et quand un utilisateur s'est connecté pour la dernière fois

# HIDS : Host-based Intrusion Detection Systems

---

- 
- **L'analyse des logs doit idéalement être effectuée régulièrement et avec attention.**
  - **Néanmoins, il s'agit d'un travail fastidieux, d'où l'existence d'outils permettant de faciliter cette exploitation.**
  - **Les fonctionnalités couramment proposées sont :**
    - condenser/résumer les logs
    - trier les logs par catégories
    - mettre en évidence certains logs
      - visuellement, via un système de consultation adapté
      - dans un rapport qui ne présente pas les logs jugés inintéressants
    - effectuer un traitement périodique des logs
      - résumé journalier envoyé par mail à l'administrateur, par exemple
    - effectuer un traitement sur une durée déterminée
    - exécuter une commande donnée lorsqu'un type de log est rencontré
  - **Ceci ne doit pas empêcher une consultation et une conservation traditionnelles des logs.**

# HIDS : Host-based Intrusion Detection Systems

---



- **Les fonctionnalités précédentes sont implémentées de façon plus ou moins exhaustive dans quelques outils.**
- **Bien souvent, ces outils sont codés en Perl ou dans un langage de script adapté au travail optimisé sur des chaînes de caractères avec utilisation d'expressions régulières.**
- **On peut citer :**
  - swatch
    - l'un des plus anciens outils de ce type, assez (trop) simple
  - logsurfer
    - continuité de swatch
    - beaucoup plus de fonctionnalités
      - gestion de contextes
      - règles dynamiques
      - gestion des rotations de fichiers
  - logcheck
    - envoi par mail des lignes de logs jugées intéressantes
    - configuration assez fine possible

# HIDS : Host-based Intrusion Detection Systems

---



- **Il peut être intéressant de ne pas se contenter de loguer les débuts et fins de sessions des utilisateurs, mais aussi les commandes exécutées**
  - dans le cas où l'utilisateur paie pour utiliser le système, en fonction des ressources qu'il consomme
  - dans le cas où l'on veut surveiller l'activité des utilisateurs, les programmes qu'ils utilisent...
  
- **Un support de cette fonctionnalité est nécessaire au niveau noyau.**
  
- **Les informations obtenues sont du type**
  - utilisateur
  - commande
  - conditions d'exécution, de terminaison
  - date de fin, temps

# HIDS : Host-based Intrusion Detection Systems

---



- **La consultation des logs est une activité nécessaire mais non suffisante pour veiller sur un système.**
- **Certains événements ou certaines modifications de taille peuvent être intervenues sur le système sans pour autant avoir laissé de trace dans les logs.**
- **Le système peut aussi présenter des points jugés peu sûrs, voire dangereux, selon la politique de sécurité en vigueur.**
- **Une démarche salubre consiste à vérifier périodiquement un certain nombre de paramètres afin d'en évaluer la sécurité :**
  - vérification de la robustesse des mots de passe
  - vérification des droits sur divers fichiers sensibles
  - contrôle des services et processus actifs
  - vérification des HOME des utilisateurs :
    - .rhosts, .netrc...
    - quels exécutable y sont stockés
  - analyse des tâches lancées avec cron
- **De telles vérifications peuvent être effectuées périodiquement ou lors**

# HIDS : Host-based Intrusion Detection Systems

---

- Des outils tels que **tiger** et **satan** offrent ce genre de fonctionnalités.
- Ces outils sont assez anciens, mais permettent d'avoir une bonne compréhension de la notion d'audits.
- Les résultats sont disponibles sous formes de logs, de rapports html, et peuvent être envoyés par mail au responsable.
- Sous Debian, tiger présente un intérêt particulier puisqu'il effectue des vérifications propres à la distribution, comme le contrôle des packages, particulièrement la vérification des versions pour signaler les packages à upgrader pour raison de sécurité...
- De tels outils doivent rester une aide à l'administration de sécurité, il est dangereux de se reposer entièrement dessus :
  - faux positifs courants : un événement normal pourra être signalé comme dangereux par le programme d'audit
  - inversement, le programme peut oublier un certain nombres de points représentant une faiblesse du système

# Systemes de Détection d'Intrusion

---

## Comparaison NIDS/HIDS





- **Déploiement**
- **Furtivité**
- **Pertinence**
- **En conclusion...**



- **Le déploiement de NIDS est généralement beaucoup plus simple que celui de HIDS :**
  - NIDS
    - 1 sonde surveille un ensemble de machines
    - Sondes indépendantes du système surveillé
    - Peu ou pas d'impact sur les ressources du système surveillé
  - HIDS
    - 1 sonde pour 1 système à surveiller
    - Sondes très liées au système surveillé
    - Utilisation d'une partie des ressources du système surveillé
  
- **Un NIDS est généralement moins vulnérable qu'un HIDS**
  - Facilité d'isolation de la machine NIDS
    - Communications réseau unidirectionnelles
  - Le HIDS est sur la machine attaquée
    - Une attaque réussie peut aller jusqu'à stopper l'activité IDS locale



➤ **L'activité de détection d'intrusion réseau est généralement transparente**

- Copie de paquets vers une machine d'analyse : pas d'effet de bord notable
- La principale façon de détecter une activité IDS est de mettre en évidence une réponse automatique aux attaques (IPS)

➤ **Il est plus facile de repérer un HIDS**

- Processus particulier sur la machine
- Logs
- Tout simplement, en voyant qu'un outil particulier est installé !



- **Quelle est la pertinence d'un outil de détection d'intrusions, ie comment peut-on au mieux limiter les faux positifs ?**
  - La détection d'intrusion réseau permet rarement de savoir si une attaque a abouti
    - Détection de la signature de l'attaque
    - Mais quid de la suite : obtention d'un shell, élévation de privilèges, déni de service sur la machine visée...???
  - La détection d'intrusion système facilite l'obtention de réponses à ces questions
    - Processus visé toujours actif ?
    - Changement d'uid d'un processus ?

- **Les rôles des NIDS et HIDS sont largement complémentaires**



- **Une architecture idéale allie les 2**

- Selon la nature des systèmes à surveiller et de l'activité qui leur est liée, un choix judicieux sur un site ne le sera pas sur un autre
- Corréler les informations fournies par les 2 types de sondes fournit une valeur ajoutée non négligeable

- **Les contraintes d'exploitation ne permettent cependant pas toujours d'optimiser la surveillance**

- Utilisation des ressources pour les HIDS
- Coûts au niveau de l'infrastructure réseau pour les NIDS

# Systemes de Détection d'Intrusion

---

## Exploitation des IDS



- **Détecter, et ensuite ?**
- **Visualisation des alertes**
- **Centraliser**
- **Corréler**
- **L'IDS en tant qu'architecture**
- **Informations pertinentes**
- **Normalisation**
- **Analyse poussée et outils complémentaires**

## ➤ **La détection est un premier pas**

- événement suspect
  - levée d'alerte
    - l'accumulation d'alertes est sans intérêt
    - appelle une réaction !

## ➤ **Besoins :**

- alerter
  - moyens techniques : logs, pager, astreintes...
- visualiser les alertes
- analyser
  - recueil d'informations annexes sur l'événement suspect
  - connaissances (ou bases de connaissances) disponibles pour comprendre l'attaque
- synthétiser
  - rapports d'incidents précis



## ➤ Une analyse efficace se basera sur une visualisation des alertes

➤ ergonomique

➤ claire

➤ ne pas noyer l'admin sous les informations secondaires

➤ complète

➤ toutes les informations nécessaires doivent cependant être accessibles facilement

## ➤ Les premiers paramètres recherchés

➤ type d'attaque (scan, attaque web, shellcode...)

➤ source(s)

➤ cible(s)

➤ horodatage et éventuellement fréquence

➤ protocoles, ports impliqués

## ➤ Ce qu'on cherchera ensuite

➤ antécédents

➤ origine (pays) de l'attaque

➤ actions potentiellement liées à l'incident



- **Pour gagner en pertinence, la centralisation d'information est un plus**
  - L'architecture CIDF permet de regrouper au sein d'une même A-box les événements/alertes venant de sondes distinctes
  - La visualisation peut donc se faire sur une machine unique, sécurisée
    - le regroupage n'est pas nécessairement global
      - grouper des sondes surveillant un même périmètre
      - grouper des sondes surveillant le même type d'activité sur différents sites
  
- **On peut dès lors mutualiser des ressources**
  - humaines
  - machines
  
- **Et gagner en efficacité dans l'analyse, notamment grâce à la corrélation**



- **Plus que de la centralisation, la corrélation apporte du sens**
  - des alertes mises en commun peuvent prendre une dimension invisible lorsqu'elles sont traitées indépendamment
    - exemple :
      - scan d'un réseau,
      - attaque (shellcode) d'une machine de ce réseau,
      - attaque depuis cette machine d'un serveur sensible,
      - compromission de données sensibles sur le serveur
    - la corrélation va permettre de retracer l'historique, la démarche de l'attaque
  
- **Elle reste difficile à automatiser, mais se basera sur des critères récurrents**
  - notion de temps
  - sources/destinations communes
  - phases caractéristiques d'attaques (repérage/offensive/exploitation)



- **Pour aboutir à des résultats au niveau centralisation et corrélation, il faut pouvoir se dédouaner de problèmes tels que**
  - la séparation de zones
    - physiques
    - politiques
  - la latence et les problèmes d'horodatage
    - les systèmes doivent être synchronisés (ntp)
  - une trop grande hétérogénéité de l'information traitée
    - au niveau de la qualité de l'information
    - mais aussi de son format !
  
- **Si le principe reste simple, sa mise en place est donc loin d'être évidente !**



- **Les informations intéressantes au niveau de l'alerte sont assez facilement identifiables, mais qu'en est-il des informations plus globales ?**
  
- **Là encore, la visualisation de l'information aura un rôle prépondérant, et on s'intéressera surtout :**
  - à des notions statistiques
    - qui est le plus visé ?
    - quelles sont les sources les plus virulentes ?
    - quelles sont les types d'attaques les plus menés contre notre réseau ?
  - à des notions périodiques
    - ex : apparition de worm
  - à des analyses spécifiques
    - répétitions de phénomènes précis
      - scans lents ? bots ?



- **Pour permettre la mise en commun des informations disponibles au sein de chaque cellule de l'architecture de détection d'intrusion, il importe de normaliser les moyens d'échange**
  - formats divers
  - contenus divers
  
- **Un objectif clair : l'interopérabilité !**
  
- **L'IETF travaille sur le draft IDMEF**
  - Intrusion Detection Message Exchange Format
  - basé sur XML
  - fournit un descriptif de messages pouvant être échangés par des IDS, basé sur des tags
  
- **Son utilisation est cependant encore très marginale, les solutions propriétaires étant souvent préférées**



- **Une fois que l'IDS a mis en évidence une attaque et permis de recueillir les informations nécessaires à une véritable analyse, le travail devient plus purement humain.**
  - forensics réseau ou système : large panel d'outils d'analyses !
  - grande dépendance vis-à-vis du système attaqué et de ses spécificités
    - la démarche ne sera pas la même sur un LAN de stagiaires et sur la DMZ !
  
- **L'aspect technique n'est pas le seul à considérer :**
  - report d'incident
  - mesures à prendre
    - modification des politiques de sécurité ?
    - évolution des systèmes de protection, des architectures ?
    - campagnes de patches ?
  - coût de l'attaque
  - répercussion sur l'image de l'entreprise ?

# Systemes de Détection d'Intrusion

---

## Des IDS particuliers : Honeypots & IPS





- **Qu'est-ce qu'un pot de miel ?**
- **Objectifs**
- **Les différents types de pots de miel**
- **Difficultés**
- **En pratique : exemples**
- **Challenges**
- **Leurrer**
- **Surveiller**
- **Contrôler**
- **Efficacité ?**
- **Honeynets**

# Qu'est-ce qu'un pot de miel ?

---



- ***A honeypot is a security resource whose value lies in being probed, attacked or compromised.***

*L. Spitzner*

- **La démarche consiste donc à *offrir* un système d'informations en pâture aux attaquants.**

# Objectifs

---



- **Les intérêts en terme de sécurité sont divers, et ne sont pas restreints à la détection d'intrusion !**
- **Un IDS particulier : toute interaction avec le pot de miel est suspecte puisqu'il ne s'agit pas d'un système supposé converser avec le monde.**
  - Réduction des faux positifs
  - Ce n'est pas un système de production mais un appât : ressources dédiées
- **Occuper le pirate**
  - Le pot de miel peut faire perdre du temps à un attaquant !
- **Découvertes de nouvelles vulnérabilités ou d'exploits**
  - Aide à la génération de nouvelles signatures

# Les différents types de pots de miel

---

## ➤ On peut distinguer 2 aspects discriminants pour les pots de miel

- Le degré d'interaction
- La nature-même du pot de miel

## ➤ Interaction

- Faible interaction
  - Simulation de services sans réel système sous-jacent
  - Relative simplicité au niveau de l'émulation
- Forte interaction
  - Le honeypot représente un véritable système d'exploitation

## ➤ Nature

- Physique
  - Le honeypot est une machine réelle, munie d'outils adaptés pour tracer l'activité

## ➤ Virtuelle

- Tout le système est émulé (machine virtuelle par exemple)

## ➤ Organisationnelles

- Légalité du procédé ? (Incitation ?)
- En cas de compromission, risques que le honeypot devienne point de rebond pour de nouvelles attaques
- Un système complet et complexe de plus à gérer

## ➤ Techniques

- Analyse l'activité : ce peut être très complexe
- Spécificité de la maintenance d'un tel système : doit à la fois paraître vulnérable et être assez sécurisé pour éviter une véritable compromission
- Domaine en pleine évolution : aspect veille technologique non négligeable et donc mobilisation de ressources humaines

## ➤ Autres

- Surenchère vis-à-vis de l'attaquant : déployer des honeypots peut

## ➤ Mise en place d'un honeypot à faible interaction

### ➤ Outils

- Le plus basique : netcat sur un port particulier
- Plus complet : émulation de services, de machines, de réseaux avec honeyd

### ➤ Risques

- Ils reposent alors essentiellement sur l'application utilisée pour l'émulation

## ➤ Mise en place d'un honeypot à forte interaction

### ➤ Offrir un véritable OS aux pirates

- Une machine peu sécurisée (vieil OS, services vulnérables...)
- Utiliser une machine virtuelle (VMWare, User Mode Linux)

### ➤ Risques

- Prévenir les rebonds repose principalement sur l'architecture réseau dans laquelle s'inscrit le honeypot


➤ **Attirer le pirate**

- Lui permettre d'accéder à un shell (réel ou non !)

➤ **Leurrer le pirate**

- Qualité de l'émulation
- Difficulté à fingerprinter le système
  - Application (honeypot...)
  - Machine virtuelle
- Activité de la machine : dissimuler les outils de surveillance

➤ **Surveiller le pirate**

- Traçabilité avec des outils spécifiques
- Gestion des logs

➤ **Contrôler le pirate**

- Maîtriser suffisamment le honeypot et le réseau pour éviter tout débordement
  - Vers l'OS sous-jacent pour une machine virtuelle
  - Vers un réseau ou une ressource sensible de façon générale



- **Il faut à la fois dissimuler les mécanismes de surveillance et émuler un système réel !**
- **Des outils de surveillance trop visibles seront un frein à l'efficacité du honeypot**
  - Limiter la visibilité qu'il peut avoir du système
  - Déporter l'analyse
  - Travailler au niveau noyau plutôt qu'application ?
- **Emulation**
  - Réseau
    - Empreintes : nmap, xprobe, p0f...
    - Topologie et activité : une machine trop isolée peut sembler suspecte
  - Système
    - Empreinte basée sur des informations bas niveau (cpu, bios, devices)
    - Simuler une activité





- **Il faut à la fois dissimuler les mécanismes de surveillance et émuler un système réel !**
- **Des outils de surveillance trop visibles seront un frein à l'efficacité du honeypot**
  - Limiter la visibilité qu'il peut avoir du système
  - Déporter l'analyse
  - Travailler au niveau noyau plutôt qu'application ?
- **Emulation**
  - Réseau
    - Empreintes : nmap, xprobe, p0f...
    - Topologie et activité : une machine trop isolée peut sembler suspecte
  - Système
    - Empreinte basée sur des informations bas niveau (cpu, bios, devices)
    - Simuler une activité : utilisateurs, processus, charge machine...



- **Le mécanisme de surveillance est directement lié aux méthodes de leurre mises en place**
  - Logs honeyd
  - Logs de scripts personnalisés
  - Noyau patché pour tracer les appels systèmes
  - Logs du système viruel ou réel dans le cas de honeypots à forte interaction
  
- **Il est particulièrement intéressant dans ce genre de situations de déporter les logs sur un serveur central !**
  
- **En plus de ces logs système un contrôle permanent de l'activité réseau s'impose**
  - Tentatives de joindre d'autres machines
  - Types d'exploits utilisés
  - Activité échappant au contrôle local ?!



- **Il faut impérativement permettre à l'attaquant de s'introduire sur le honeypot sans toutefois lui permettre de rebondir vers d'autres systèmes**
  - Soit de notre propre réseau
  - Soit de l'extérieur
  
- **Contrôle des flux réseau**
  - Entrée autorisée, vers des adresses bien déterminées
  - Sorties scrupuleusement limitées
    - Utilisation de pare-feu à état
    - Limitation à des protocoles bien identifiés (DNS !)
    - Proxys applicatifs
    - Redirections de trafic
    - QoS restrictive : offrir un débit dérisoire et dissuasif en sortie



- **L'exploitation et l'analyse de tels systèmes sont loin d'être triviales**
  - Nombreuses traces
  - Nombreux formats
  - Interprétation isolée des événements
  - Corrélation difficile
  - Expertise nécessaire
  
- **Il apparaît intéressant d'inscrire les honeypots dans une architecture de surveillance plutôt que de les utiliser isolément**
  - Analyse corrélée
  - Honeynets



- **Il s'agit d'un type particulier de honeypot dont la structure a été travaillée et standardisée par le Honeynet Project.**
  
- **Ils reposent sur des honeypots à forte interaction et visent à capturer des informations exploitables !**
  
- **Il ne s'agit pas d'un produit, mais d'une proposition d'architecture**
  - Architecture réseau fortement contrôlée
    - NIDS classiques
    - Analyse de flux
  - Tout trafic entrant ou sortant du honeynet est suspect !
  - Il existe différentes générations de cette architecture



- **Le concept d'IPS**
- **Contre-mesure**
- **Exemples de contre-mesure réseau**
- **Dangers**

- **Intrusion Prevention System : mieux vaut prévenir que guérir ?**



- **Le constat est le suivant :**

- on suppose pouvoir détecter une intrusion
- pourquoi ne pas en profiter pour y mettre un terme ?

- **Evolution des termes : de l'IDS à l'IPS**

- terme à la base plutôt marketing
- y a-t-il, techniquement
  - un réelle intérêt ?
  - de réelles possibilités ?

# Contre-mesure

---



- **Au niveau réseau la contre-mesure peut s'appliquer de nombreuses façons, sur différentes couches protocolaires.**
- **Mais si l'on parle de contre-mesure, l'outil devient un outil actif !**
- **Conception d'architecture, déploiement différents d'un NIDS standard.**
- **On conserve le potentiel d'un outil pour assurer une fonction de sécurité sur un réseau entier, mais on perd la furtivité.**





- **Les objectifs peuvent être multiples**

- interrompre une connexion
- ralentir une connexion
- black-lister une source d'attaque

- **Les moyens sont également variés**

- émission de reset TCP (par un équipement en coupure)
- émission de messages ICMP (host/network unreachable)
- paramétrage dynamique de règles de filtrage
  - pour une source donnée (ou plusieurs)
  - vers un service donné
- QoS
- intervention au niveau applicatif
  - reverse proxy



- **Les faux-positifs restent un problème majeur des IDS**
  - conséquences en cas d'erreur ?
  - risques de verrouillage ?
  - risques de retourner l'outil contre son propre réseau ?
    - nécessité d'être précautionneux
      - white lists
      - certains services/liens ne doivent pas être interrompus
    - procéder par étapes
      - tester la détection simple, passer à la prévention progressivement
  
- **Un NIDS reste relativement facile à tromper, sa furtivité peut être un atout majeur**
  - mais dans le cas de l'IPS, forger une attaque pour que l'IPS paralyse le réseau surveillé peut être un challenge intéressant pour le pirate !

# Systemes de Détection d'Intrusion

---

## Conclusion

# Conclusion

---

## ➤ **Les IDS ne dispensent pas**

- d'une architecture de sécurité purement défensive
  - filtrage réseau, systèmes d'authentification, blindage des serveurs
- d'une surveillance humaine intelligente du système
- d'une veille sur les attaques possibles

## ➤ **Ils sont pas parfaits quant**

- aux attaques inconnues
- aux problèmes de performance
- aux méthodes d'évasion

## ➤ **Mais malgré tout ils peuvent**

- assurer une solide aide aux administrateurs de sécurité
- donner une bonne vision des attaques subies
- permettre d'évaluer le niveau de défense du réseau
- aider à mieux comprendre son réseau !