

Supervision de sécurité et détection d'intrusion

Introduction

Guillaume HIET

guillaume.hiet@centralesupelec.fr

Equipe CIDRe, CentraleSupélec-Inria-IRISA(CNRS)

Septembre 2017

Notes

- 1 Présentation du module supervision de sécurité
 - 2 Principes généraux de la détection d'intrusions
 - 3 Approche classique : misuse-based NIDS
 - 4 Architecture de supervision
 - 5 Outils de détection d'intrusions
 - Sondes NIDS
 - Sondes HIDS

Notes

- 1 Introduction à la détection d'intrusions
 - 2 Présentation de quelques approches classiques (aspects opérationnels + recherche)
 - 3 Présentation de plusieurs approches originales (recherche)
 - 4 Introduction à la corrélation d'alertes, présentation de quelques approches (aspects opérationnels + recherche)
 - 5 Découvrir par la pratique : configuration d'une sonde IDS et d'un manager

Notes

Etat art technique

Guide to Intrusion Detection and Prevention Systems (IDPS). Karen A. Scarfone, Peter M. Mell. NIST SP - 800-94.

Corrélation d'alertes

Christopher Kruegel, Fredrik Valeur and Giovanni Vigna. Intrusion detection and correlation. Springer. ISBN 0-387-23398-9. 2005.

Article général sur les IDS

H.Debar, M.Dacier and A.Wespi. A Revised Taxonomy for Intrusion-Detection Systems. Annales des Télécommunications, Vol.55, No7-8, 2000.

En français ...

Marc Dacier. Chapitre « La détection d'intrusions » du traité IC2 sur la sécurité des systèmes d'informations. Hermes. ISBN 2-7462-1259-5. 2006.

Notes

1 Phase 1 : analyse de risques

- Identifier les ressources à protéger
 - Identifier les menaces qui pèsent sur ces ressources
 - Evaluer la vraisemblance de ces menaces

② Phase 2 : définition d'une politique de sécurité

- Qui est autorisé à utiliser la ressource et comment ?
 - Qui est autorisé à accorder des droits sur la ressource ?
 - Qui possède les priviléges de l'administrateur ?

③ Phase 3 : mise en œuvre de la politique de sécurité

- Choix de l'ensemble des mécanismes les plus simples possibles permettant de protéger les ressources de la manière la plus efficace et pour un coût acceptable
 - Approches préventives
 - Surveillance *a posteriori* : **détection d'intrusions**
 - « Nihil tam munitum quod non expugnari pecuna possit », Ciceron (106-43 Avant JC)¹

1. « Aucune place n'est assez fortifiée que l'argent n'en vienne à bout »

Notes

Intrusion

Violation de la politique de sécurité

Attaque

Tentative d'intrusion

Scénario

Suite des étapes élémentaires d'une attaque ou d'une intrusion

Signature (règle)

Selon le contexte :

- symptôme(s) révélateur(s) d'une attaque ou d'une intrusion
 - motif caractéristique d'un comportement estimé « normal »

Notes

Vulnérabilité (faille)

Défaut (bug) exploité par l'attaquant pour mettre en œuvre son attaque

- conception
 - réalisation (codage)
 - administration
 - utilisation
 - etc.

Détection d'intrusions

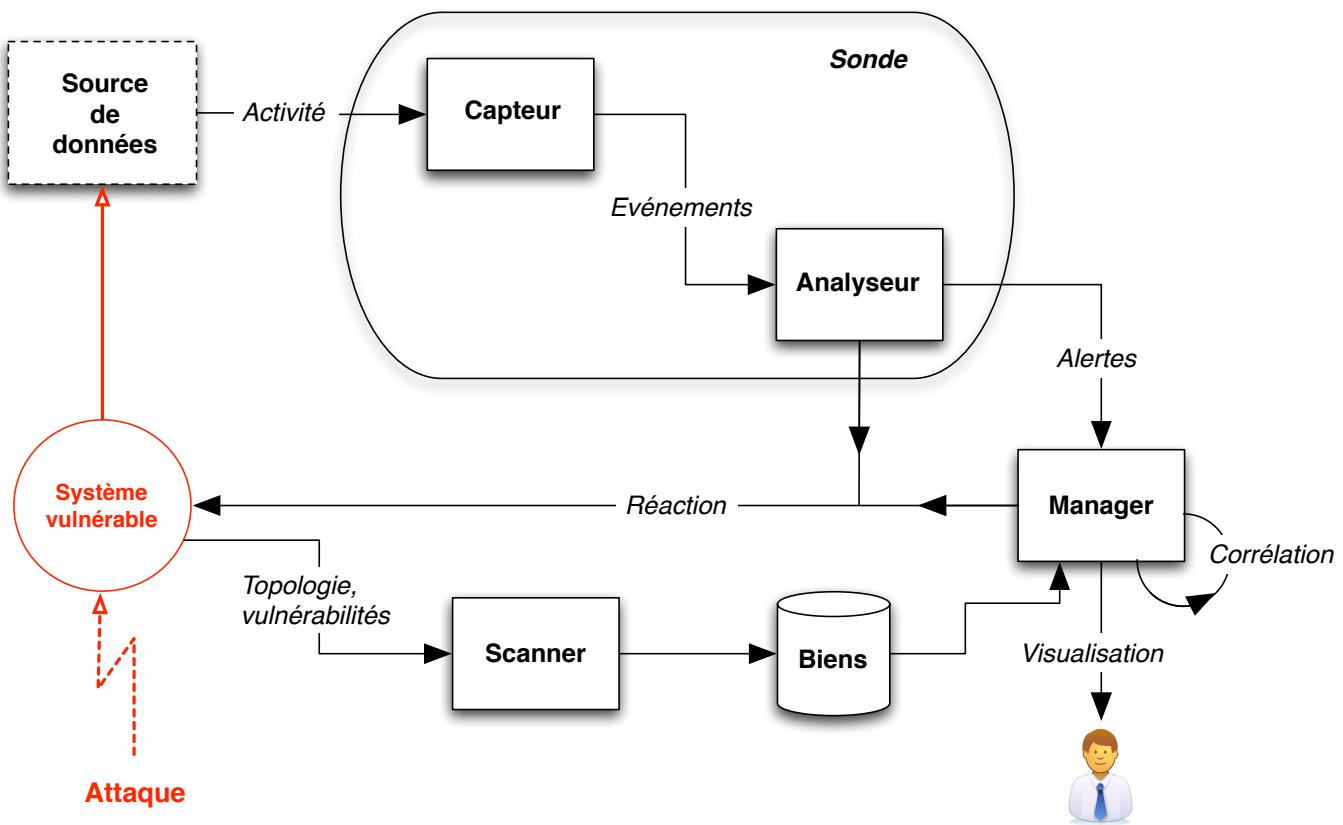
Ensemble de techniques permettant de surveiller un système informatique et d'identifier **automatiquement** les intrusions contre ce système en vue de prendre les contre-mesures adéquates pour ramener le système dans un état sain.

- Détection d'intrusions vs. détection d'attaques
 - Importance de la qualité du diagnostic

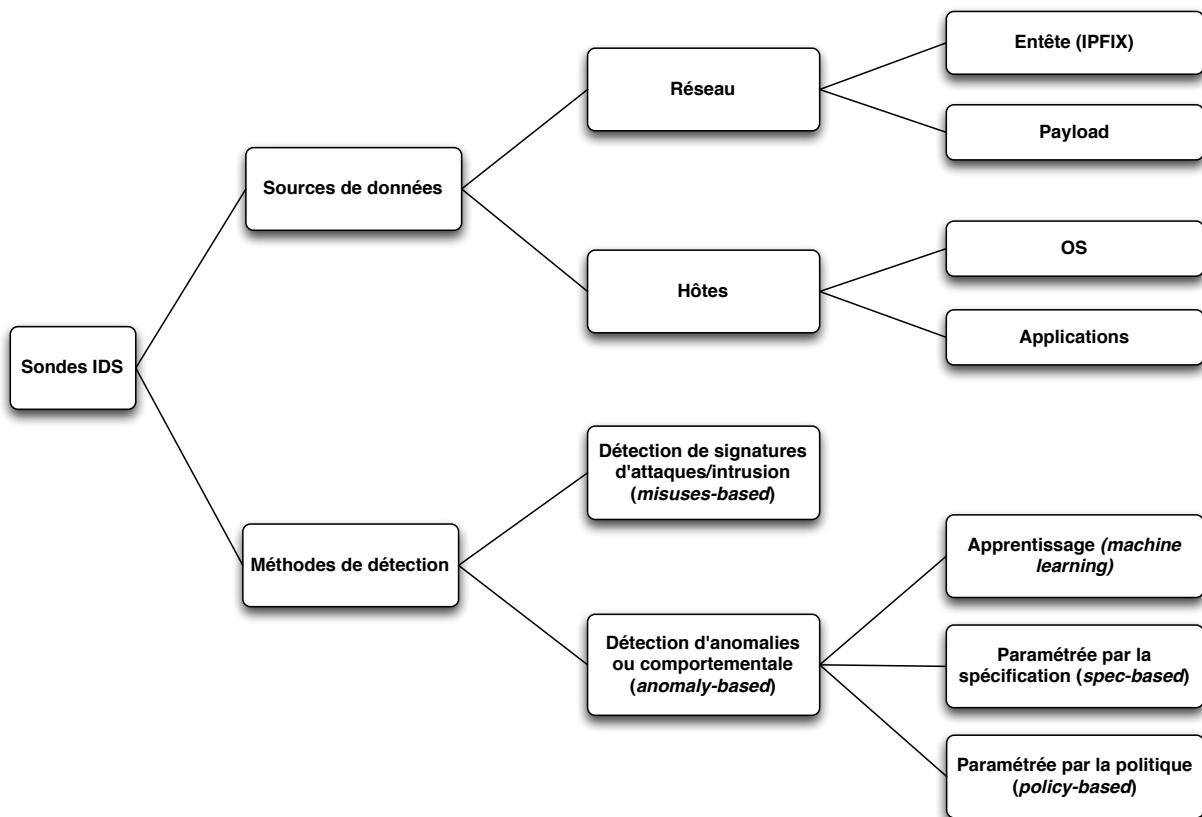
Notes

- Premiers travaux par Anderson [And80] et Denning [Den87]
 - Travaux initiés par le DoD
 - Milieu 80 à milieu 90, début 2000 : purement du domaine de la recherche
 - Depuis : commercialisation et toujours (beaucoup de) travaux de recherche
 - Fin 2000 : une certaine convergence avec les domaines de la lutte anti-virus et de l'analyse de *malware*
 - Récemment : intégration dans la supervision de sécurité

Notes



Notes



Notes

Capture

- Généralement, utilisation de sondes dédiées
 - Capture passive : TAP ou hub ou port miroir + carte capture
 - Fonctionnement en ligne (en coupure) possible

Avantages

- Couverture large (dépend du mode de capture)
 - Peu d'impact sur le SI car sondes dédiées (dépend du mode de capture)
 - Format standard des données (sauf pour les protocoles propriétaires)
 - Réaction possible (*Intrusion Prevention Systems*)

Inconvénients

- Réseaux commuté → multiplication des sondes
 - Montée en débit des réseaux
 - Chiffrement des flux

Notes

- Capture passive
 - + furtivité, surveillance au sein d'une zone (via *hub* ou *switch*), non-intrusive (perturbation limitée du réseau surveillé)
 - évasion, réaction moins facile
 - Capture *inline*
 - + facilite la réaction (IPS), normalisation
 - non-furtif, impact les performances, trafic entre zones
 - Dispositif externe de capture :
 - *hub* : uniquement pour petit réseau/faible débit
 - *switch* : miroir de port (*spanning / replication port*)
 - + pas de matériel supplémentaire, surveillance de zone
 - limité en débit, risque de perte, risque de mauvaise configuration
 - TAP
 - + pas de problème de débit, pas de perte sur la transmission
 - coût, utilisation sur un brin (surveillance inter-zone, cf mode *inline*)
 - Remarque : *inline* ≠ blocage, possibilité de ne bloquer que certaines alertes (impact de faux +)

Notes

Capture

- Sondes co-localisées avec les systèmes surveillés
 - Analyse des journaux de l'OS
 - Eventuellement, installation d'un module (noyau) spécifique

Avantages

- Informations précises (utilisateurs, processus, fichiers, etc.)
 - Mécanisme de journalisation ou d'audit fourni par la plupart des OS (syslog, WMI, etc.)

Inconvénients

- Impact sur les performances de l'hôte (CPU, mémoire)
 - Vulnérabilité de la sonde
 - Hétérogénéité des formats de données à analyser
 - Informations très bas niveau + vision locale

Notes

Capture

- Sondes spécialisées et co-localisées
 - Analyse des journaux des applications ou installation d'un module (plugin) spécifique
 - Exemples :
 - Serveurs web (très utilisé en D.I., cf. WAF)
 - SGBD (plus rare)

Avantage

Moins de données à capturer, moins de formats à analyser, classe d'attaques à détecter plus restreinte ⇒ meilleures performances de détection

Inconvénient

- Idem Source 2 (impact sur le système surveillé, vulnérabilité de la sonde)
 - Multiplication des sondes

Notes

Approches par scénarios ou signatures d'attaque (*misuse detection*)

- Modèle d'intrusions (base de signatures d'attaques)
 - Signature = symptômes d'attaque dans activités observées
 - Alerte si présence de symptôme(s)
 - Dans la pratique : analyseur = (multiple-)pattern matching

Approches comportementales (*anomaly detection*)

- Modèle des comportements légaux
 - Alerte si activité observée \neq des comportements normaux
 - Dans la pratique : analyseur = apprentissage et modèle statistique
 - Légal \rightarrow usuel

Notes

Fiabilité (sensibilité)

- Intrusion (attaque) \Rightarrow alerte
 - Pas de faux négatif i.e. intrusion (attaque) non détectée

Pertinence (spécificité)

- Alerte \Rightarrow intrusion (attaque)
 - Pas de faux positif i.e. fausse alerte

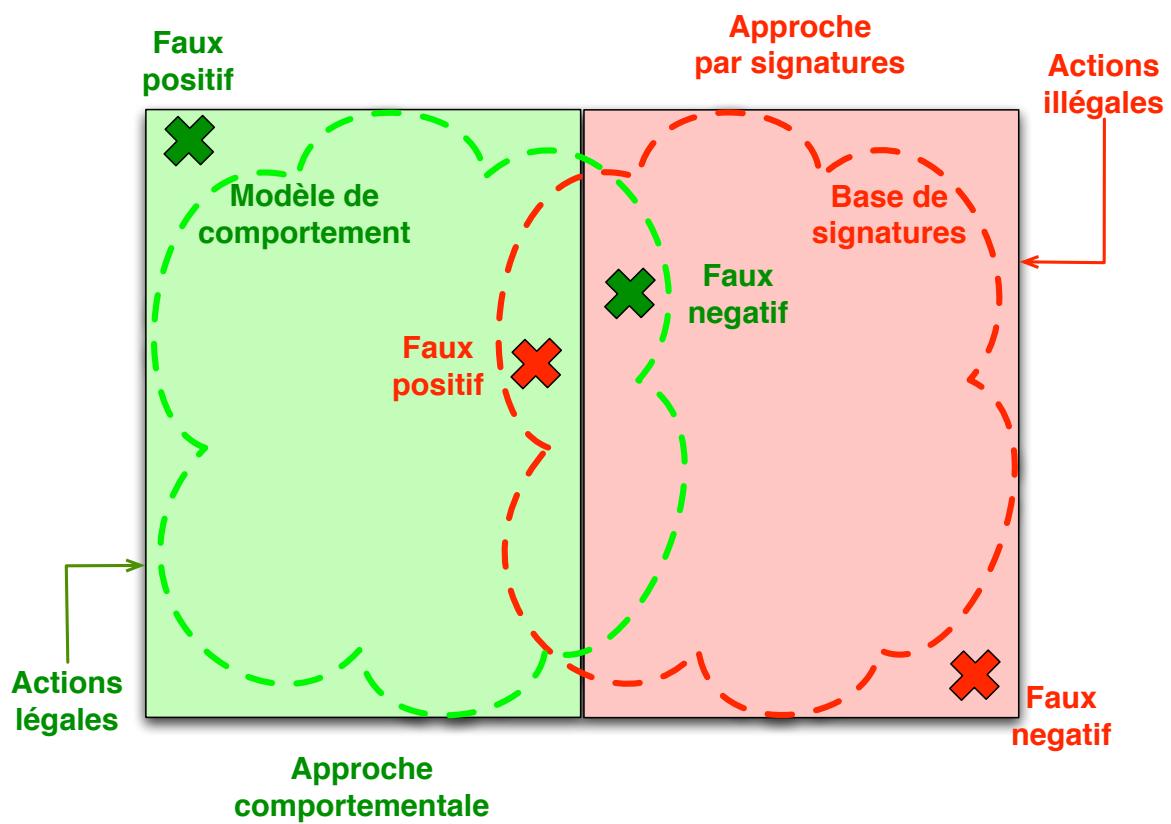
Objectifs

- Une détection fiable et pertinente ...
 - ... soit un détecteur sensible et spécifique

Contraintes

- (Quasi-)temps réel
 - Ressources limitées

Notes



Notes

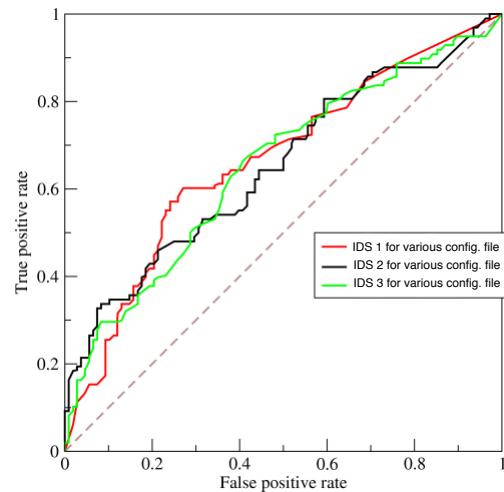
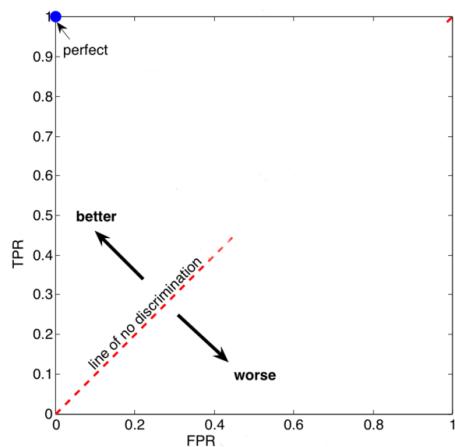
Evénements et alertes

		actual value	
		<i>p</i>	<i>n</i>
prediction outcome	<i>p'</i>	True Positive	False Positive
	<i>n'</i>	False Negative	True Negative

Fiabilité/sensibilité : $\text{TPR} = \text{TP} / (\text{TP} + \text{FN})$
(=1 si pas de faux négatif)

Pertinence/spécificité : $FPR = FP / (FP + TN)$
 $(=0 \text{ si pas de faux positif})$

Courbe ROC (Receiver Operating Characteristic)



Notes

Ensemble des événements analysés : $\Omega = B$ (bénin) + M (malicieux)
 On a aussi : $\Omega = P$ (positif) + N (négatif) = $TP + TN + FN + FP$

Taux d'événement malicieux² (*base rate*) :

$$r = \frac{M}{M+B}$$

Probabilité qu'une alerte soit vraie :

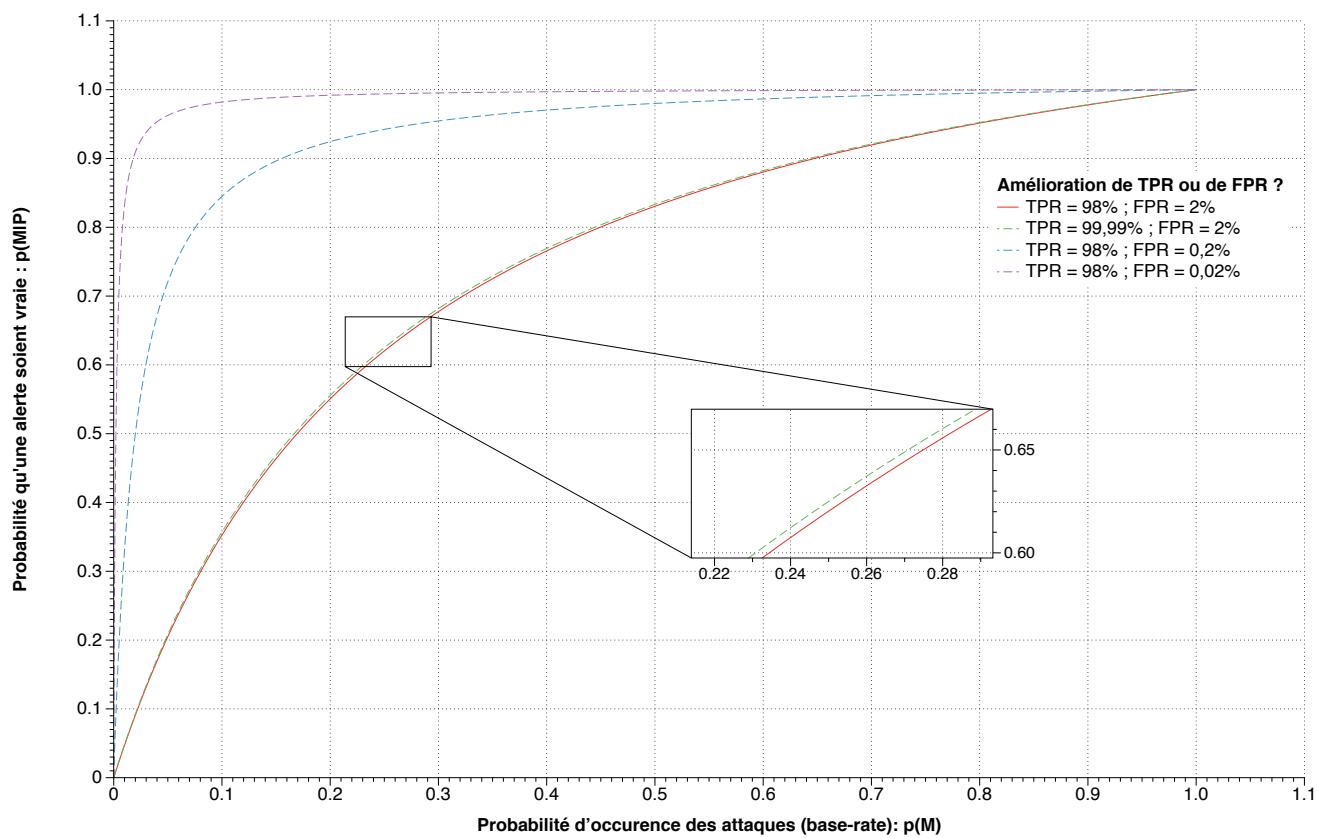
$$p(TP) = \frac{TP}{TP + FP}$$

On a : $TP = TPR.M = TPR.r.\Omega$ et $FP = FPR.B = FPR.(1 - r).\Omega$

$$\Rightarrow p(TP) = \frac{TPR.r.\Omega}{TPR.r.\Omega + FPR.(1 - r).\Omega} = \frac{TPR}{TPR + (1/r - 1)FPR}$$

2. Abus de langage : M désigne ici $\text{Card}(M)$

Notes



Notes

Approche par signatures ou scénarios (*Misuse Detection*)

- + Pas parfaitement fiable, mais peu de faux positifs
 - + Qualification des alertes (diagnostic)
 - Fiabilité décroît si base de signatures mal maintenue
 - En général, pas de détection des nouvelles attaques (« zero day »)
 - Difficultés pour la spécification des signatures :
 - Trop précises → facilement contournables (polymorphisme) + ↗ nombre signatures (↗ consommations des ressources)
 - Trop génériques → ↘ pertinence

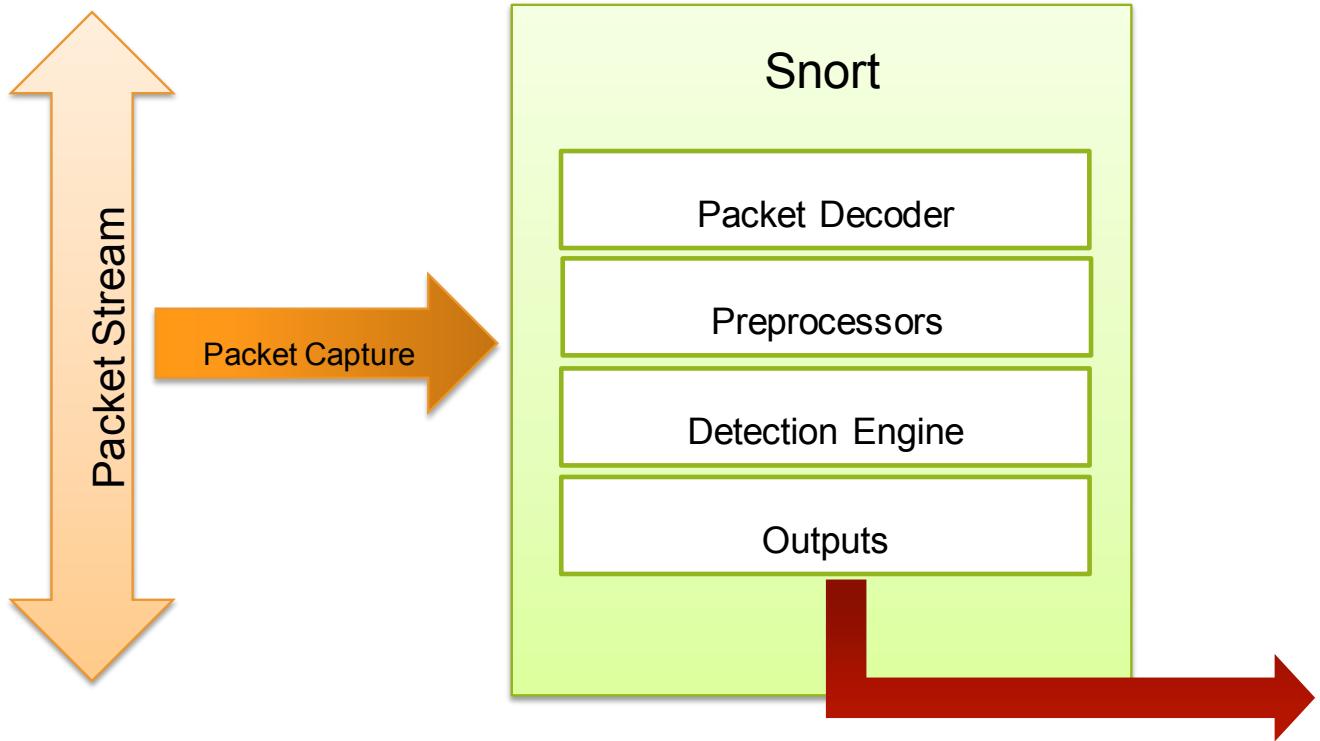
Approche comportementale par apprentissage (*Anomaly Detection*)

- + Pas toujours pertinent, mais peu de faux négatifs
 - + Capacité (théorique) de détection des *zero day*
 - Difficile prise en compte de l'évolution du comportement (ré-apprentissage)
 - Pas de diagnostic associé aux alertes

Notes



- Sonde NIDS/NIPS libre créée par Marty Roesch
- Développée par Sourcefire (rachetée par Cisco en 2013)
- Première version en 98 (pour UNIX)
 - *"Lightweight" intrusion detection*
 - Simple *pattern matching* sur les paquets
 - Pas de défragmentation, pas de reconstruction de flux
- Aujourd'hui, logiciel complet utilisé dans des *appliances* (dont celles de Sourcefire)
 - Frag3, Stream5 → défragmentation, suivi flux TCP/UDP
 - Décodage applicatif (HTTP, FTP, etc.)
 - Différents modules de capture (inline, passif) et de sortie (fichier, BDD, etc.)
- URL : <http://www.snort.org/>



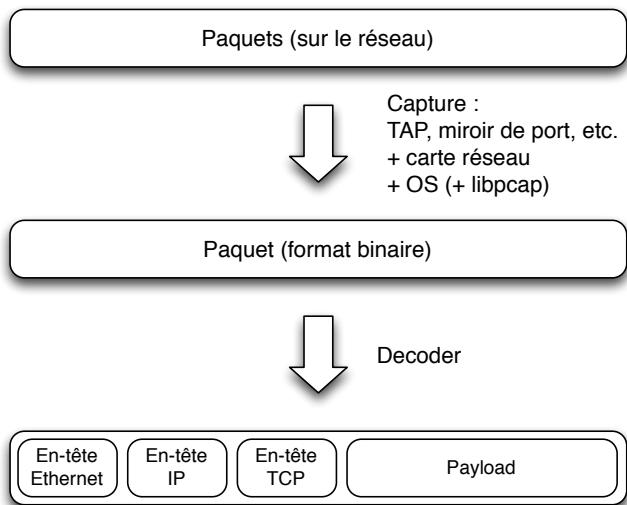
Notes

Modules de capture

- Gérés par LibDAQ (V2.9)

Decoder

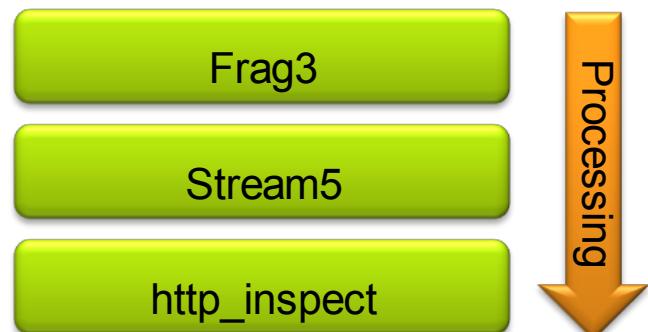
- Analyse protocolaire sommaire
 - Décodage entêtes Ethernet, IP et TCP
 - Quelques vérification simples (taille champs en-têtes, etc.)



Notes

Préprocesseurs

- Défragmentation,
reconstruction de flux
 - Analyse des protocoles
applicatifs
 - Détection d'anomalies,
normalisation, etc.
 - Attention, ils sont exécutés
dans l'ordre ou il sont
déclarés dans le fichier de
configuration !

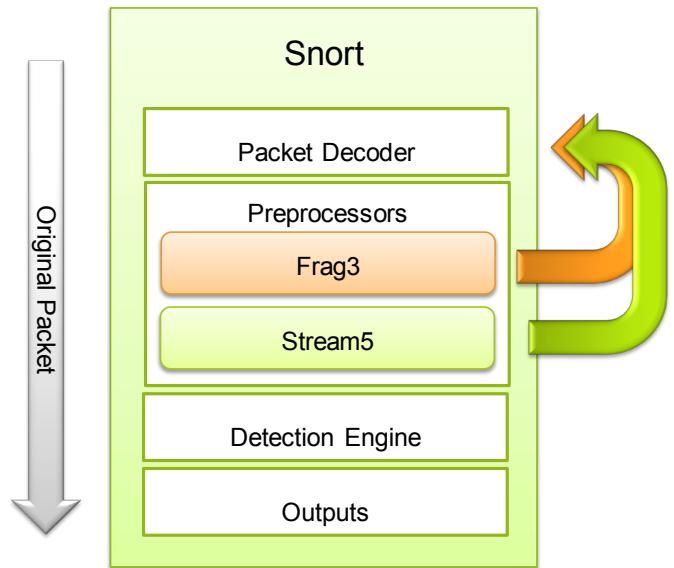


Notes

- Moteur de détection + règles (signatures)
 - Moteur de reconnaissances de signatures
 - Les signatures (dont dépendent beaucoup les résultats)
 - Modules dynamiques
 - Pré-processeurs dynamiques
 - Règles dynamiques (*so-rules*)
 - Modules de sortie :
 - Syslog
 - Ecriture directe en BDD
 - Unified1 et 2 (format binaire, rapide, permet de faire tampon avant traitement en BDD)
 - Export Prelude
 - Réaction (IPS), etc.

Notes

- Frag3 :
 - Défragmentation IP
 - Détection d'attaques liées à la fragmentation
 - Génération d'un pseudo paquet qui est réinjecté (les paquets fragmentés suivent leur chemin)
 - Stream5 :
 - Réassemblage flux TCP, suivi états
 - Détection attaques liées à la « fragmentation » TCP
 - Génération d'un pseudo paquet
 - Configuration par port



Notes

- `Http_inspect` :
 - Normalisation URI
 - Création tampon URI (qui peut être inspecté dans les règles → `uricontent`)
 - Détection des tentatives d'évasion et des anomalies HTTP
 - Attention au paramétrage de `flow_depth` : faux négatifs vs. consommation de ressources importantes

GET /downloads/..../cgi-bin/..../pics\..../downloads/.snort.tar.gz HTTP/1.0

/downloads/snort.tar.gz

Notes

- Signatures « officielles » VRT (Sourcefire)
 - Non-libres
 - Gratuites pour une utilisation non-commerciale (disponibles 1 mois après parution)
 - Abonnement payant : utilisation commerciale et disponibilité immédiate
 - Signatures fournies par des tiers : par exemple Emerging Threats³
 - Règle (signature) = en-tête + options
 - Options :
 - **general** : msg, reference, sid, classtype, priority, etc.
 - **payload** : content, uricontent, isdataat, pcre, byte_x, etc.
 - **non-payload** : flow, ttl, tos, id, fragbites, dsizes, flags, flowbits, etc.
 - **post-detection** : resp, react, tag, activate, activate_by, count, replace, etc.

3. <http://www.emergingthreats.net/>

Notes

```
alert tcp $EXTERNAL_NET any -> $HOME_NET  
4000 (msg:"EXPLOIT Alt-N SecurityGateway  
username buffer overflow attempt"; flow:established,  
to_server; content:"username=".content:nocase;  
isdataat:450,relative; content:"!&"; within:450;  
content:"|0A|"; within:450; metadata:policy balanced-  
ips drop, policy connectivity-ips drop, policy security-  
ips drop; reference:url,secunia.com/advisories/30497/;  
classtype:attempted-admin; sid:13916; rev:2;)
```

Notes

Exemple de vulnérabilité : *SQL injection*

code PHP :

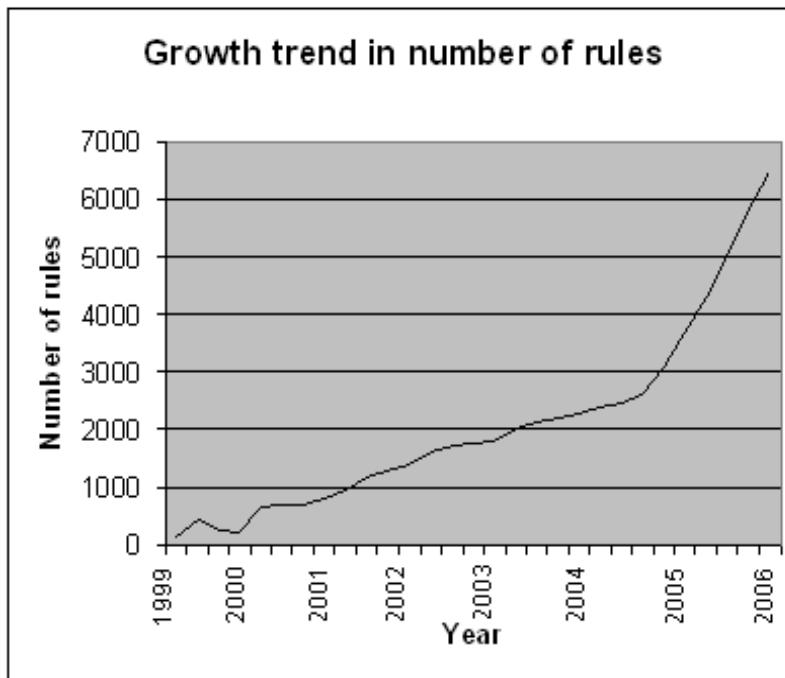
```
req = "SELECT * FROM users WHERE name = '" + user + "';"
```

Une signature très (trop) générique

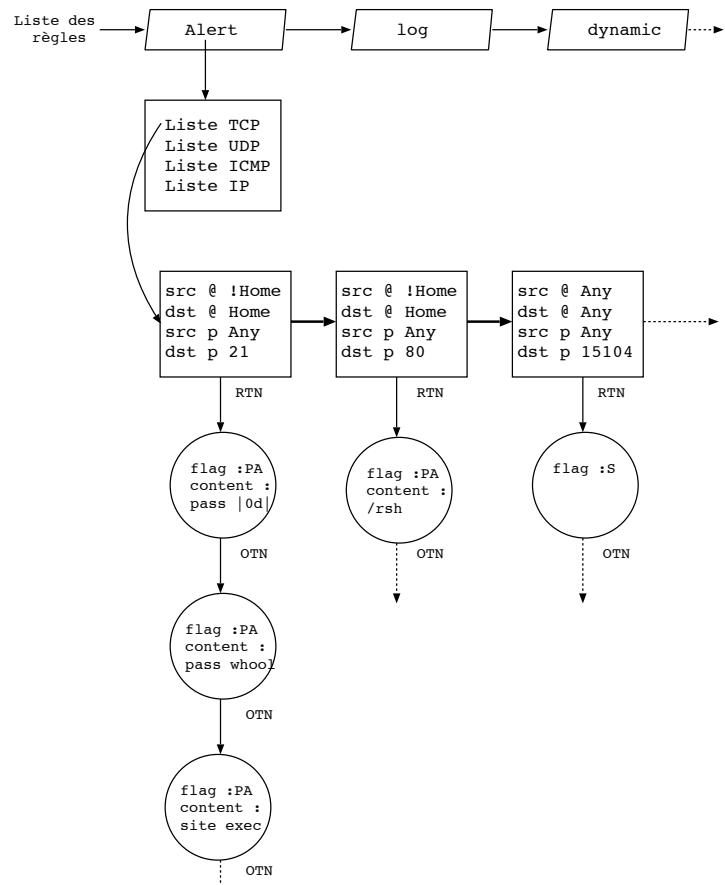
```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS  
(msg:"SQL Injection"; flow:to_server,established;  
uricontent:".php"; pcre:"/(%\27)|(\')|(\-\-)|(%23)|(\#)/i";  
classtype:Web-application-attack; sid:9099; rev:5; )
```

Notes

- Problème : nombre de règles ↗ \Rightarrow nombre motifs ↗
 \Rightarrow Optimiser la gestion des règles pour limiter la vérification des motifs



Notes



Notes

Pour chaque (pseudo) paquet :

- Identifier les RTN suivant en-tête TPC/IP du paquet
 - Vérifier chaque OTN :
 - motifs fixes (ex : content) : Boyer-Moore (recherche séquentielle) ou Aho-Corasick (recherche en parallèle)
 - regexp (pcre) : moteur NFA/DFA
 - algorithmes *ad hoc* pour les autres types de motifs (non-payload, isdataat, byte_x, etc.)
 - Les OTN sont évalués séquentiellement ou en parallèle suivant une stratégie complexe (compromis temps/mémoire)
 - Les algorithmes doivent être robustes pour éviter les attaques en complexité algorithmique
 - Identification des règles vérifiées en fonction des motifs vérifiés et réalisation de l'action correspondante (génération alerte, etc.)

Notes

Améliorer la qualité et la précision du décodage protocolaire

- Fournir des événements de « haut-niveau » (établissement de connexion, téléchargement d'un fichier, etc.)
 - Permettre de restreindre la recherche sur des champs protocolaires précis

Améliorer les performances du moteur de détection

- Analyser les règles pour proposer des stratégies de regroupement (compromis temps/mémoire)
 - Utilisation de matériel dédié (FPGA, ASIC, GPU)
 - Parallélisation du processus de détection (cluster)

Notes

- ① Présentation du module supervision de sécurité
 - ② Principes généraux de la détection d'intrusions
 - ③ Approche classique : misuse-based NIDS
 - ④ Architecture de supervision
 - ⑤ Outils de détection d'intrusions
 - Sondes NIDS
 - Sondes HIDS

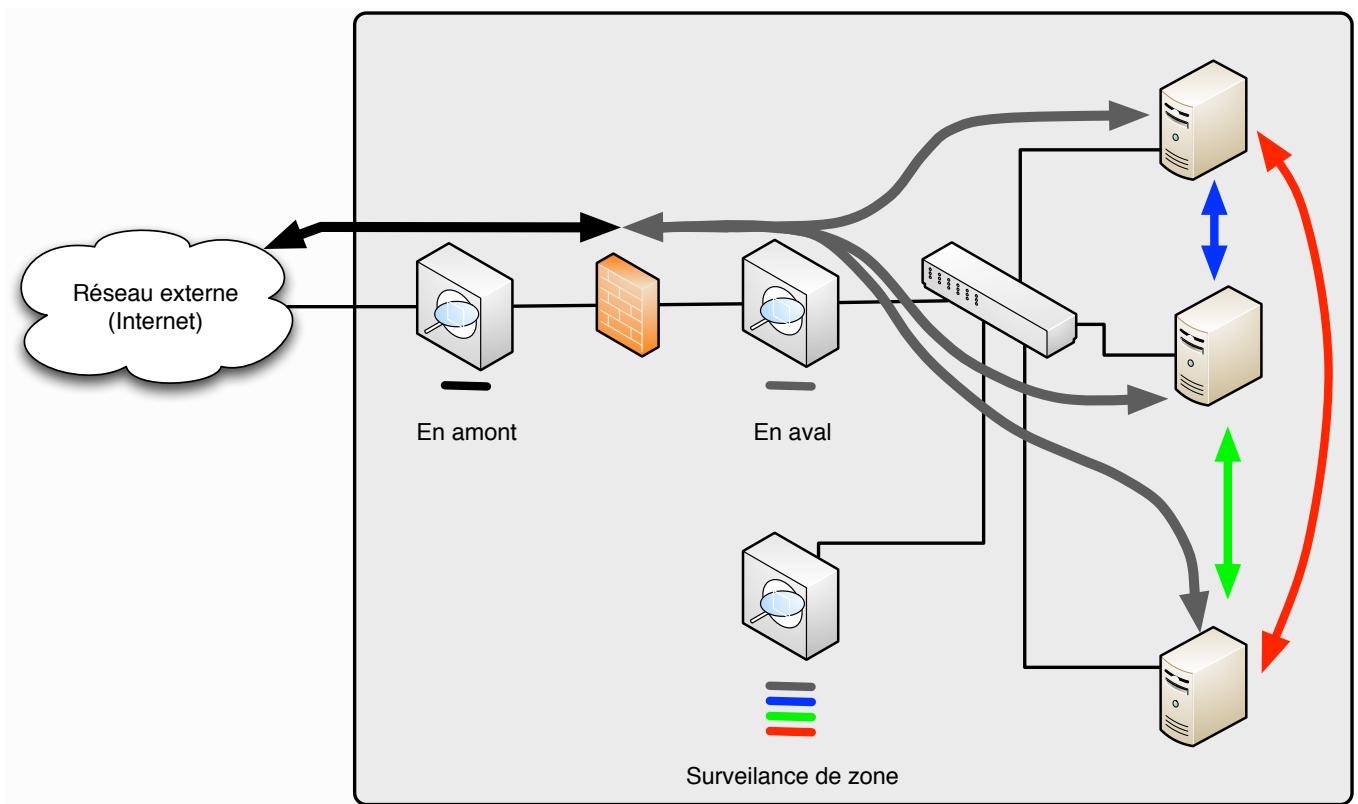
Notes

- D'abord, définir ses besoins (politique de sécurité, analyse de risques / d'impacts)
 - Choix des sondes :
 - Capacité de détection/prévention
 - Robustesse (attaques contre l'IDS, sûreté de fonctionnement)
 - Performances (paquets/s, connexions simultanées, etc.)
 - Facilité de gestion (documentation, support technique, services, mise à jour, interopérabilité)
 - Coût durant tout le cycle de vie (initial + maintenance)
 - Choix de l'emplacement et du type de capture (passive ou en coupure)
 - Interface dédiée pour l'administration
 - Garantir la furtivité (capture passive)
 - Vulnérabilités de l'interface d'administration

Notes

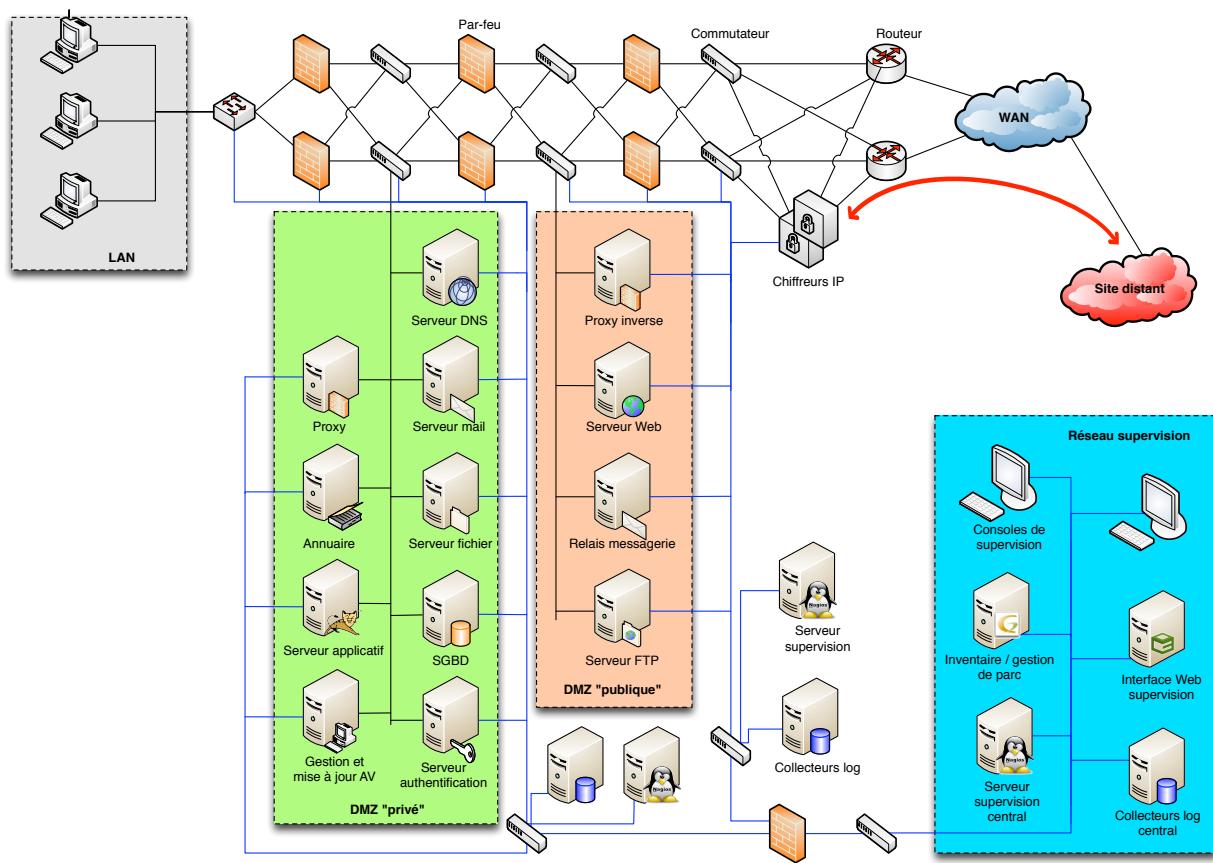
- IPS \simeq pare-feu : cloisonner des zones de confiance \rightarrow positionnement aux interfaces, en coupure
 - IDS = surveiller une zone
 - Définition des zones d'intérêt à surveiller
 - Intérêt : $I = \frac{V}{C}$
 - Valeur : $V = f(\text{impact}_{CID})$
 - Confiance : $C = f(\text{niveau protection, vulnérabilité}) = ?$
 - Autre approche (complémentaire) : cloisonner en fonction des applications (pour spécialiser les sondes)
 - Placement / pare-feu
 - ① En amont :
 - + visibilité maximale, connaissance de la menace
 - beaucoup de trafic à analyser, risque d'alertes peu pertinentes, risque d'évasions
 - ② En aval :
 - + moins de trafic, protection, vérification de la politique du FW
 - certaines attaques sont filtrées
 - ③ A la place (mode *inline*) : revient au cas 1 ou 2 selon configurations

Notes



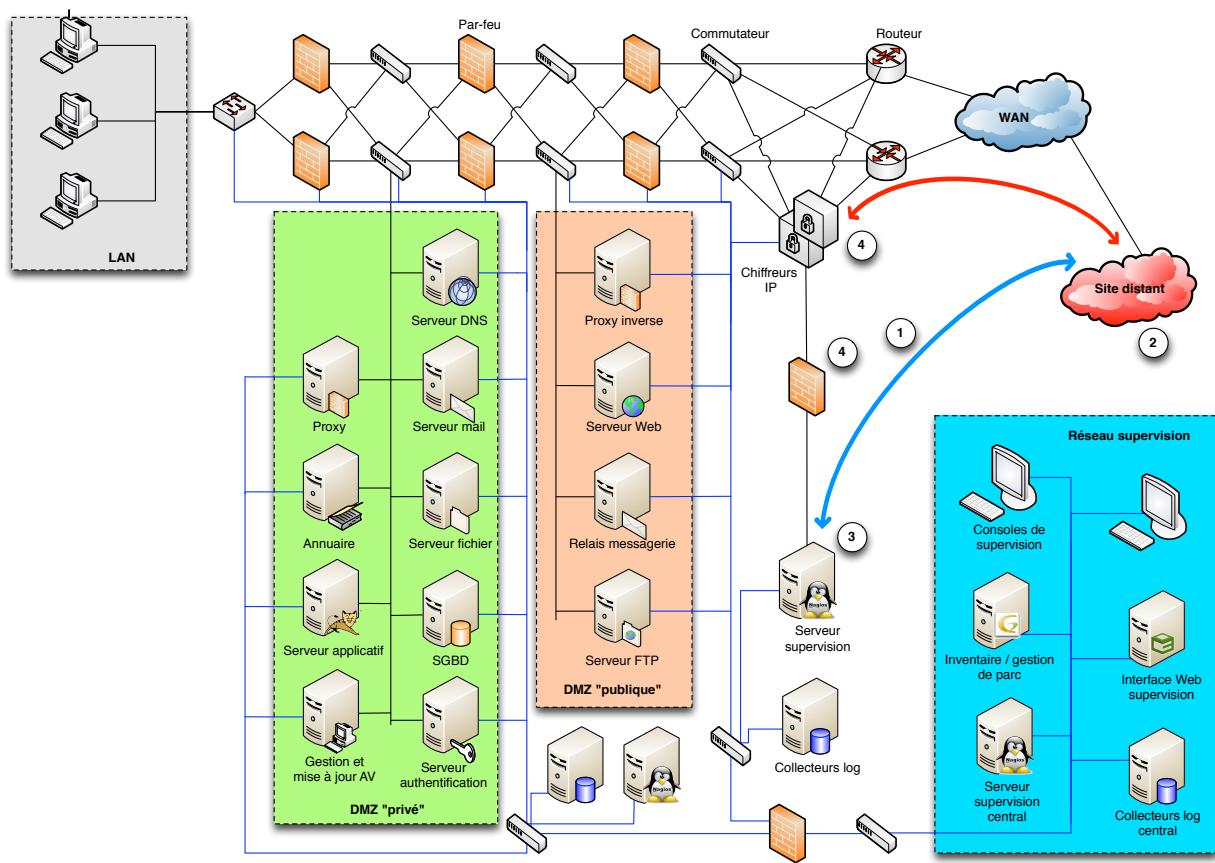
Notes

Exemple : architecture initiale



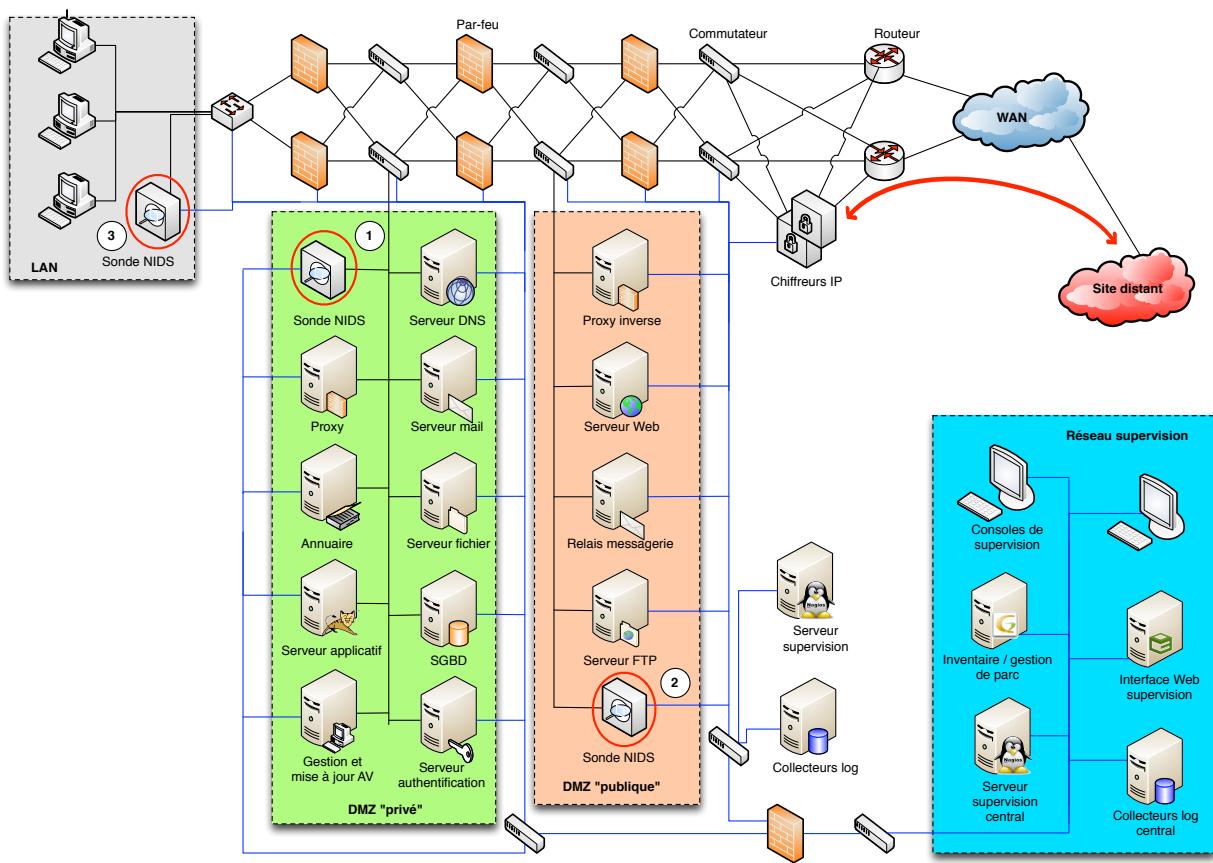
Notes

Exemple : liaison inter-sites



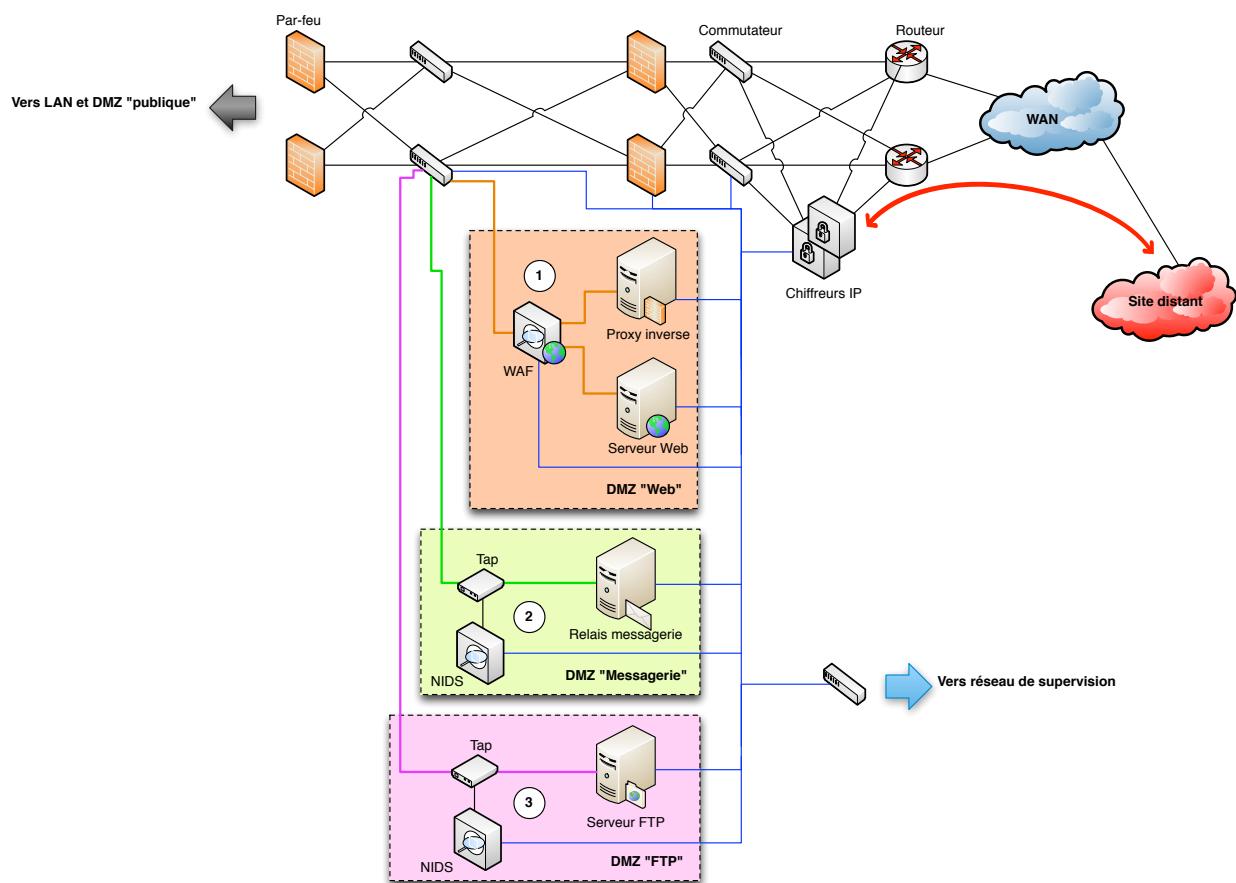
Notes

Exemple : sondes NIDS passives

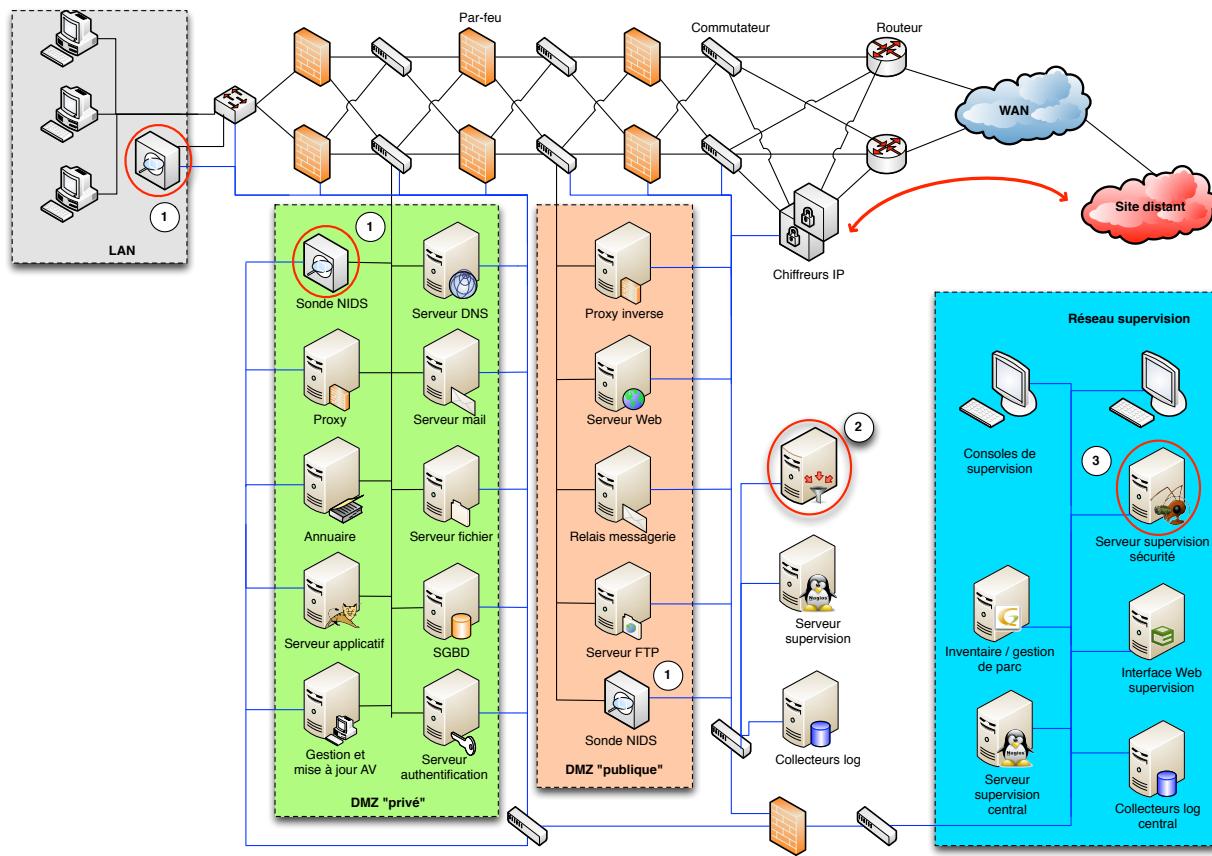


Notes

Exemple : cloisonnement horizontal

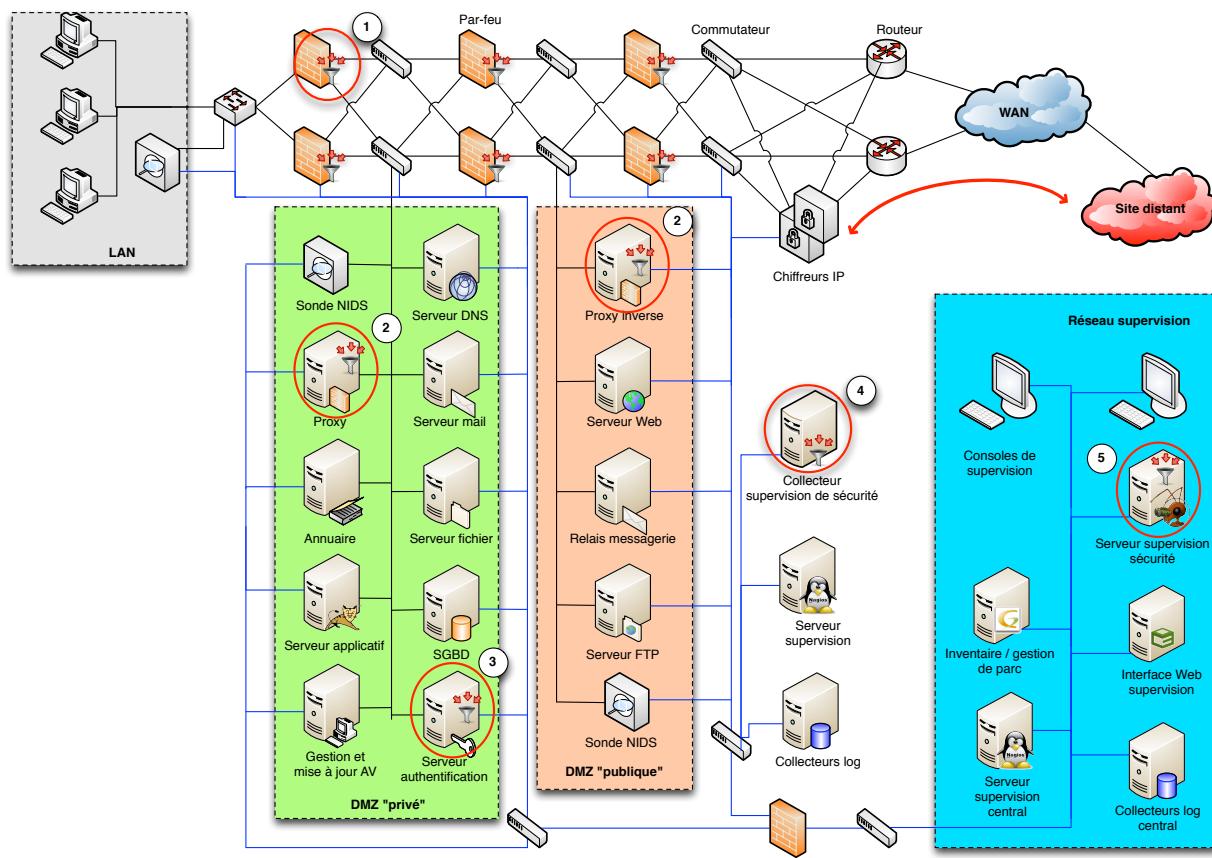


Exemple : collecte des alertes (SIEM)



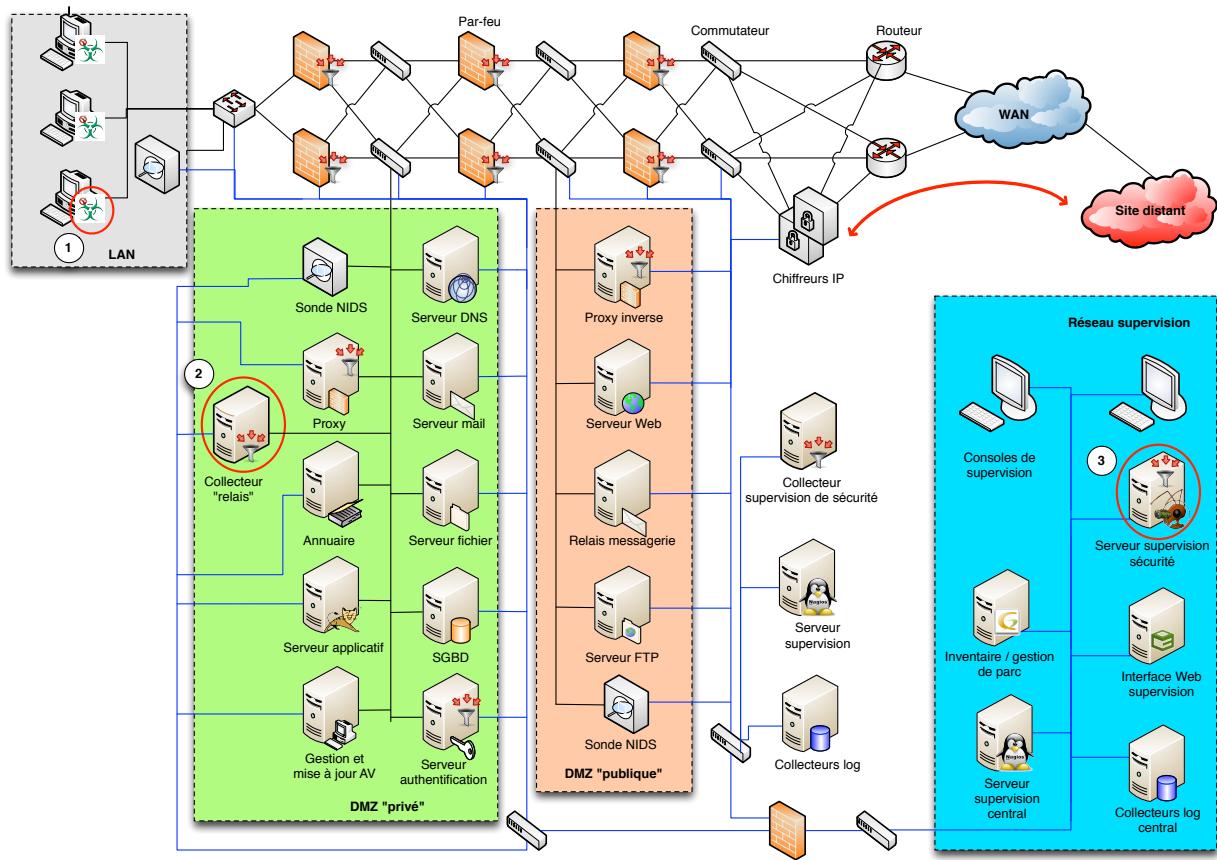
Notes

Exemple : autres événements de sécurité



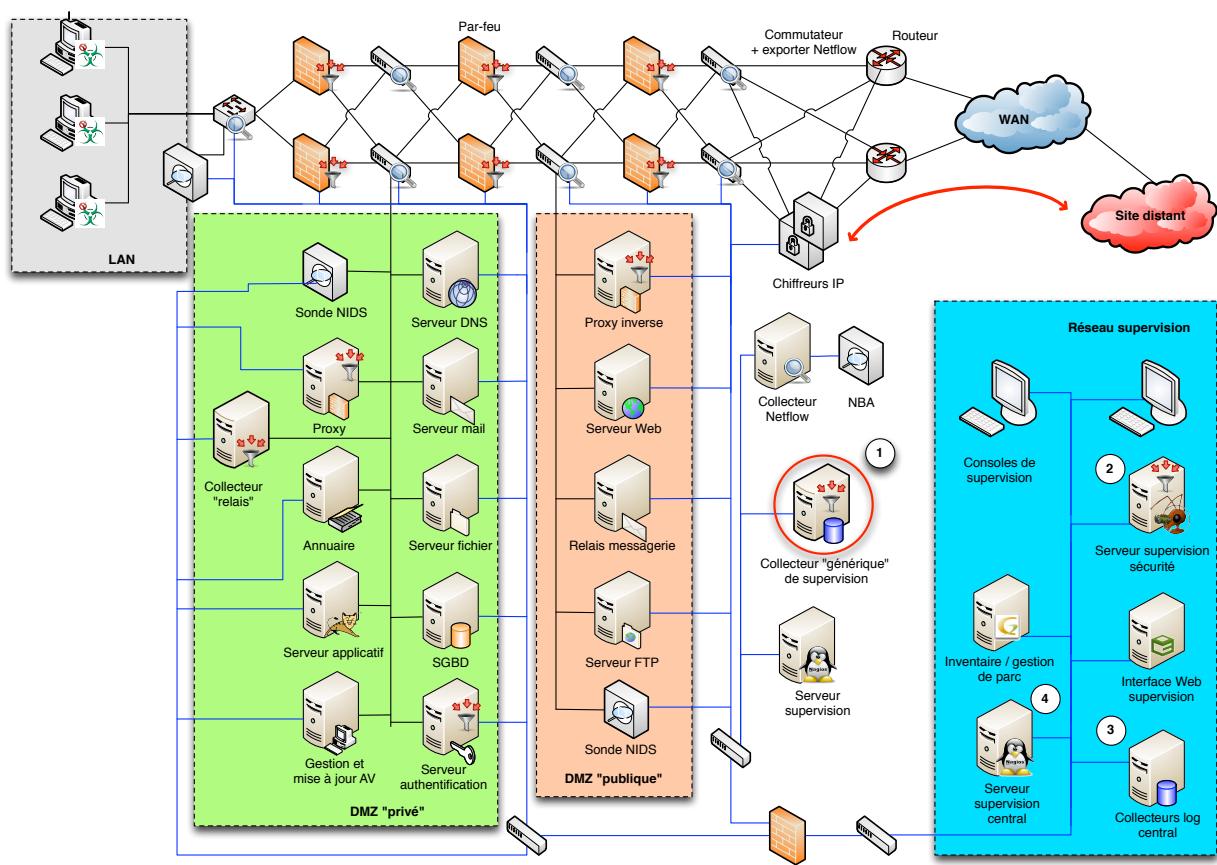
Notes

Exemple : collecte out-of-band



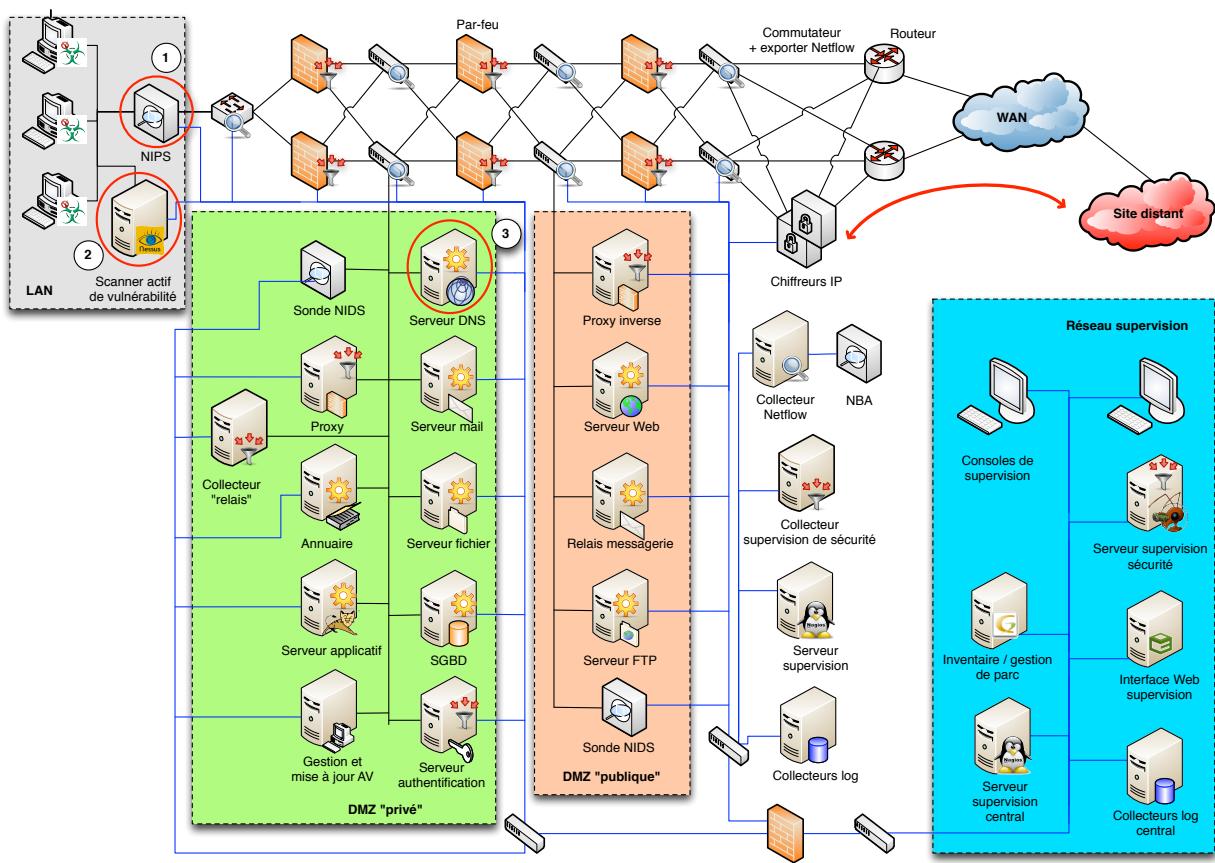
Notes

Exemple : intégration avec la supervision fonctionnelle



Notes

Exemple : HIDS, NIPS et scanner de vulnérabilité



Notes

- ① Présentation du module supervision de sécurité
 - ② Principes généraux de la détection d'intrusions
 - ③ Approche classique : misuse-based NIDS
 - ④ Architecture de supervision
 - ⑤ Outils de détection d'intrusions

Sondes NIDS

Sondes HIDS

Notes

Reconnaissance de motifs

- Technique la plus courante (Snort, etc.) et la plus « mature »
 - Importance des signatures
 - Qui fournit les signatures ? Abonnement ?
 - Les signatures sont-elles modifiables, redistribuables ?
 - Différence d'expressivité entre les signatures propriétaires et celles que l'utilisateur peut définir.
 - Sur quoi porte la recherche :
 - Des paquets ?
 - Des flux TCP ?
 - Des contexts applicatifs (URI, etc.) ?
 - Complexité : reconnaissance mono ou multi-événementielle, prise en compte de la réponse (*cf. corrélation*)
 - Efficace pour détecter un exploit particulier → détecteur d'attaques
 - Peu robuste à l'évasion

Notes

Détection d'anomalies de protocoles

- Détection de non-conformités à la spécification (RFC), valeurs suspectes ou incohérentes de certains champs
 - Nécessite un décodage précis (couteux)
 - Limites : performance, hétérogénéité des implémentations des piles protocolaires, protocoles propriétaires

Détection d'anomalies réseau

- Détection de l'importance anormale de certains flux
 - Simple mais assez efficace pour détecter certaines attaques :
 - scans (*one to many*), DDOS (*many to one*)
 - propagation de vers, *backdoors*, etc.
 - Spectre limité mais ne dépend pas d'un exploit particulier
 - Limites : apprentissage, faux +, difficulté du diagnostic

Blacklistage d'adresses IP

Notes

Solutions dédiées : plateformes matérielles dédiées (*appliance*)

- Souvent PC avec un IDS logiciel (Snort ou autre)
 - Parfois logique câblée (ASIC, FPGA), carte d'acquisition optimisée
 - Plus simple à déployer, composants dédiés et optimisés

Solutions intégrées

- Modules matériels ou logiciels (*blade*) pour switch ou routeur
 - *Universal Threat Management* (UTM) : pare-feu + VPN + IPS + ...
 - Solutions « tout-en-un » mais des ressources à partager
 - Selon NSS Labs, performances proches des *appliances* dédiées

Solutions spécifiques

- *Network Behavior Analyzer* (NBA) : détection d'anomalies réseau
 - *Wireless IPS* (WIPS) : détection d'anomalies protocolaires (WIFI)
 - *Web Application Firewall* (WAF) : HTTP et applications Web

Notes

Les produits commerciaux (*appliance* et UTM)

- Intel (McAfee), Cisco (Sourcefire), IBM (ISS), CheckPoint (NFR), HP (TippingPoint), Palo Alto, Stonesoft, Extreme Networks (Enterasys / Dragon), Radware, Corero (Top Layer), Tech Mahindra (iPolicy networks)
 - Des *appliances* à base de Snort : Fortinet, McAfee Nitro Security, e-Cop Cyclops, autres (pas toujours affiché...) ?
 - Les leaders selon Gartner : Intel (McAfee), Cisco (Sourcefire)
 - Les meilleurs selon NSS Labs : Cisco (Sourcefire), Fortinet, Palo Alto, IBM

Les logiciels open-source

- Snort (Sourcefire)
 - Bro (LBNL)
 - Suricata (OISF, Homeland Security)

Notes

- ① Présentation du module supervision de sécurité
 - ② Principes généraux de la détection d'intrusions
 - ③ Approche classique : misuse-based NIDS
 - ④ Architecture de supervision
 - ⑤ Outils de détection d'intrusions

Sondes NIDS

Sondes HIDS

Notes

- Contrôle d'intégrité (fichiers, base de registre, noyau, etc.)
 - Vérification (trop) simple
 - Prise en compte des changements légitimes (mise-à-jour, etc.)
 - Recherche de motifs dans les journaux (*cf.* limites des NIPS par signatures)
 - Heuristique : détection de malware, rootkit
 - Lien avec les Anti-Virus
 - Analyse du comportement des applications (*spec-based, policy-based*)
 - Lien avec certains mécanismes de contrôle d'accès mandataire (SE-Linux, AppArmor, Blare, etc.)
 - Nécessite d'avoir des politiques ou des profils corrects et complets
 - Attention aux faux + si profils ou politiques trop restrictifs
 - Analyse du comportement des utilisateurs

Notes

Produits commerciaux

- Contrôle d'intégrité
 - Tripwire, NetIQ Change Guardian, McAfee Integrity Control, NNT Change Tracker, etc.
 - Protection du poste client/serveur
 - « Suites de sécurité » (*Endpoint Protection*)
 - Anti-Virus, Firewall personnel, HIPS, Anti-malware, etc.
 - Les éditeurs « classiques » : AVG, Avira, Kaspersky, Symantec, ESEC, Sophos, McAfee, etc.
 - Autres : Stormshield Endpoint Security, Trend Micro (Third Brigade) Deep Security, PowerBroker Endpoint Protection, etc.
 - « Firewall personnel » :
 - SoftSphere DefenseWall, Online-Armor, Emsi Software Mamutu, etc.

Notes

Logiciels libres

- Détection de motifs simples dans les journaux
 - Prelude LML (CS)
 - Détection motifs + contrôle intégrité + détection rootkit
 - Samhain
 - OSSEC (TrendMicro)
 - OSIRIS
 - Contrôle intégrité simple
 - AIDE

Notes

- [0707] World intrusion detection and prevention systems markets, #N22B-74 - Frost & Sullivan, 2007.
 - [And80] James P. Anderson,
Computer Security Threat Monitoring and Surveillance, Tech. report, James P. Anderson Company, Fort Washington, Pennsylvania, April 1980.
 - [Axe99] Stefan Axelsson, The base-rate fallacy and its implications for the difficulty of intrusion detection, Proceedings of the 6th ACM Conference on Computer and Communications Security, November 1999, pp. 1–7.
 - [BB05] David Bizeul and Olivier Badet, Systèmes de détection d'intrusion réseau, architecture et paramétrage, 2005.
 - [Den87] Dorothy E. Denning, An Intrusion-Detection Model, IEEE transaction on Software Engineering **13** (1987), no. 2, 222–232.

Notes

- [Jac99] Kathleen A. Jackson, Intrusion detection system (ids) product survey, Tech. Report LA-UR-99-3883, Los Alamos National Laboratory, June 1999.
 - [Lab10] NSS Labs, Network ips group test q4 2010, 2010.
 - [SM07] Karen Scarfone and Peter Mell, Guide to intrusion detection and prevention systems (idps), Recommendations of the National Institute of Standards and Technology (NIST Special Publication 800-94), February 2007.
 - [YP10] Greg Young and John Pescatore, Magic quadrant for network intrusion prevention systems, Gartner RAS Core Research Note G00208628, decembre 2010.

Notes

- Ludovic Mé, *détection d'intrusions* (transparents), Supélec, 2011.
 - Steve Sturges, *Snort 2.8.5 Overview*, SourceFire
 - Steve Sturges, *Snort 2.8.4 Overview*, SourceFire
 - Steve Sturges, *Performance Tuning Snort*, SourceFire
 - Soumya Sen, *Performance Characterization and Improvements of SNORT* Lucent Technologies, July 2006

Notes