INSA de Toulouse
5ème année
Réseaux et Télécommunications

**2011-2012**

# Systèmes de détection d'intrusion

**Rodolphe Ortalo**

Informations document
Titre : Systèmes de détection d'intrusion
Créé le : 2004-05-10 22:34, Rodolphe Ortalo
Modifié le : 2011-12-30 22:18, Rodolphe Ortalo
Dernière impression le : 2006-10-30 09:55, Rodolphe Ortalo
Révision n° : 664
Statistiques :  pages, 72752 caractères, 1 tableau(x), 20 image(s).
Sujet : détection d'intrusion, sécurité informatique
Mots-clefs : informatique, sécurité, IDS

# Table des matières

# Index des tableaux



# Index des illustrations

# 1 La détection d'intrusion

L'objectif de la détection d'intrusion est de repérer les actions d'un attaquant tentant de ou tirant partie des vulnérabilités du système informatique pour nuire aux objectifs de sécurité du système, ou utilisant des techniques recensées comme des techniques d'attaque.

Par rapport à la plupart des autres moyens de la sécurité informatique (au sens de la protection contre les malveillances) qui sont avant tout des moyens de protection ou de prévention des intrusions ou des vulnérabilités, les techniques de détection d'intrusion sont donc orientées vers la surveillance du système informatique et constituent dans une certaine mesure des moyens de tolérance aux intrusions.

La relation entre une intrusion et les objectifs de sécurité du système n'est pas toujours mentionnée, mais nous considérons qu'il est indispensable de se référer à la politique de sécurité du système informatique pour définir précisément ce qui est une intrusion pour un système donné, c'est à dire une défaillance de sécurité.

Il est vrai que, dans l'état de l'art au sens le plus large, il est possible de recenser un certain nombre d'actes ou d'actions techniques qui sont considérées par une large majorité comme des attaques répréhensibles. Dans ce cas, un système détectant l'occurrence de ces attaques peut suffire à identifier des intrusions sans établir de relation directe avec les objectifs de sécurité. Toutefois, dans nombre de situations concrètes, c'est plutôt la situation inverse qui est rencontrée : ce sont des actions suspectes mais dont la malveillance n'est pas avérée qui sont identifiées. C'est alors bien souvent la mise en relation avec les objectifs de sécurité du système qui permet de décider de l'intérêt de suivre ou non ces actions détectées et leurs conséquences. Par ailleurs il nous semble que la politique de sécurité de tout système d'information se doit aussi d'interdire ou tout au moins d'encadrer dans un cadre obligatoire précis l'exécution d'attaques (au sens commun) sur le système informatique.

## *1.1 Terminologie*

La caractérisation des différents systèmes de détection d'intrusion existants explorés dans le domaine de la recherche a conduit à la classification terminologique présentée dans le figure 1. Cette terminologie permet de différencier les systèmes de détection d'intrusion (ou IDS[1]) suivant un certain nombre de caractéristiques.

On peut d'abord faire une distinction assez fondamentale sur la méthode de détection utilisée par l'IDS. Il existe deux grandes catégories de méthodes de détection explorées dans la littérature : celles basées sur une approche comportementale (par exemple l'analyse statistique, l'analyse bayésienne, les réseaux neuronaux) et celles basées sur une approche par scénarios (par exemple la recherche de signatures, le *pattern matching*, ou la simulation de réseaux de Petri par exemple).

1 Pour *Intrusion Detection System*.

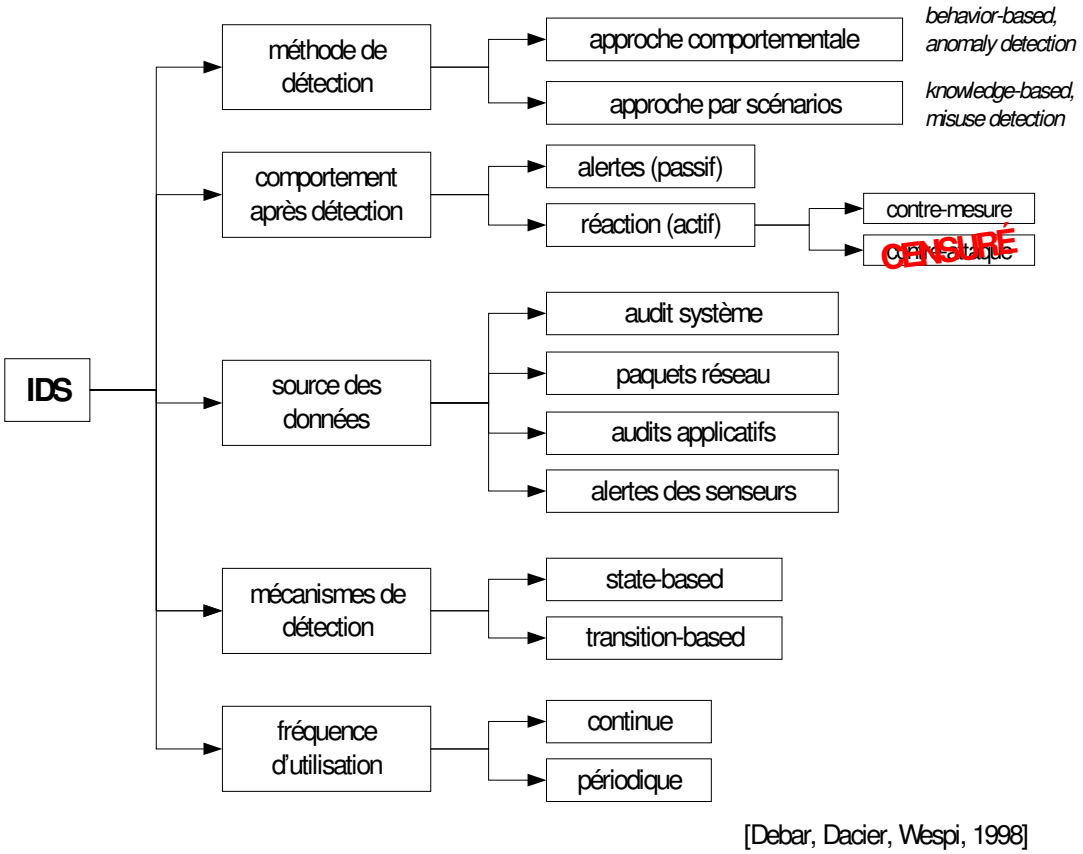



*Illustration 1: Terminologie de la détection d'intrusion*

Globalement, les approches comportementales visent à reconnaître un comportement anormal, que ce soit par rapport à une définition du comportement normal ou anormal fournie au système de détection d'intrusion (par exemple une spécification de protocole de communication) ou par rapport à une modélisation des comportements normaux ou anormaux *apprise* à partir d'une observation préalable du système (en salle blanche, ou tout simplement en réel). Dans le cadre d'une approche comportementale, l'apprentissage semble donc possible, tout comme la possibilité de détecter des attaques inconnues au moment de la conception de l'IDS, à condition qu'elles génèrent des anomalies perceptibles dans le fonctionnement normal.

Par contre, dans une approche par scénarios, l'IDS s'appuie sur une base de connaissance pré-existante décrivant les comportements normaux ou anormaux et utilise cette connaissance pour la reconnaissance des évènements produits par des actions d'intrusions dans le système informatique qu'il observe. Cette méthode implique donc la constitution et la mise à jour régulière d'une base de connaissance référençant les différentes attaques connues susceptibles d'être mises en œuvre dans un système informatique. C'est à partir de ces informations, affinées par l'administrateur en fonction du système surveillé, que l'IDS identifie d'éventuelles attaques ayant lieu dans le système informatique. Dans cette approche, l'IDS se focalise donc sur l'identification des utilisations abusives (*misuse*). Une autre mise en œuvre conforme à la terminologie mais originale dans la pratique de cette approche de détection par scénarios consiste à constituer une base de connaissance des comportements *permis* dans le système (et non des comportements abusifs) pour configurer les actions de détection (des utilisations normales en quelque sorte).

On peut ensuite comparer les systèmes de détection d'intrusion en fonction du mode de fonctionnement des mécanismes de détection qu'ils mettent en œuvre. De manière générale, un IDS peut tenter d'identifier des attaques en s'appuyant sur des informations relatives aux transitions ayant lieu dans le système (l'exécution de certains programmes, de certaines séquences d'instructions, l'arrivée de certains paquets réseau, etc.) ou bien en étudiant l'état de certaines parties du système (par exemple, l'intégrité des programmes stockés, les privilèges des utilisateurs, les transferts de droits, etc.).

Ensuite, les IDS s'appuient généralement sur des sources de données différentes. Certains IDS, dits « réseau » ou NIDS (pour *Network IDS*) examinent les paquets transportés par le réseau. D'autres IDS, dits « système » ou HIDS (pour *Host IDS*) s'appuient sur les traces d'exécution fournies en général par le système d'exploitation mais parfois aussi par

les applications[2]. D'autres IDS sont également en train d'apparaître appuyés sur des fonctions de corrélation et, dans ce cas, on peut considérer qu'il sont eux-mêmes des systèmes utilisant les alertes d'autres senseurs comme source de données.

Lors de la détection d'une attaque, un système de détection d'intrusion peut adopter plusieurs comportements. En règle générale, une réponse passive est adoptée : l'IDS diffuse une alerte identifiant l'attaque détectée vers un système d'analyse ou de diffusion. Toutefois, on peut envisager des réponses plus actives, parmi lesquelles nous distinguons d'abord la mise en place (automatique) de contre-mesures destinées à limiter la portée d'une intrusion éventuelle. Par exemple, l'IDS peut réagir en reparamétrant un *firewall* pour mettre en place des règles de blocage temporaires de certains flux réseaux anormaux. Bien entendu, il importe de bien valider la fiabilité de la détection d'intrusion avant d'activer de telles contre-mesures automatiques. Dans la pratique, peu d'administrateurs sécurité envisagent concrètement de mettre en place ce type de réaction. Nous laissons au lecteur le soin de deviner la dénomination d'une réponse active après détection d'une attaque duale d'une contre-mesure dans le vocabulaire « tactique ». Dans la pratique, nous espérons que peu d'administrateurs chargés de la sécurité envisagent cette dernière catégorie de comportement : compte tenu de la législation française, ce type de réaction ne nous semble pas permis dans le domaine civil.

Enfin, on peut également intégrer la fréquence d'utilisation de l'IDS dans les caractéristiques identifiables. Dans la perception courante d'un IDS, celui-ci est toujours utilisé de manière continue. Toutefois, dans certains cas, il peut également être important de bien identifier une détection effectuée en temps différé[3]. On peut aussi envisager que certains outils dont l'utilité est avérée dans la pratique y compris pour révéler les traces d'une intrusion passée trouvent leur place dans cette classification au niveau d'une fréquence d'utilisation périodique, comme les outils de vérification d'intégrité (type *Tripwire*) ou les outils de recherche de vulnérabilité (type COPS, SATAN, ou Nessus).

### 1.1.1 Alertes et fausses alertes

Parmi les comportement possibles pour un IDS, on peut envisager les quatre possibilités recensées dans le tableau 1 suivant qu'une intrusion est ou non en cours dans le système informatique et que le système de détection d'intrusion a émis ou non une alerte.

| | **Pas d'alerte** | **Alerte** |
|---|---|---|
| **Pas d'attaque** | Vrai négatif | Faux positif |
| **Attaque en cours** | Faux négatif | Vrai positif |

*Tableau 1: Comportements envisageables pour un IDS*

Parmi ces quatre comportements, les vrai négatif et vrai positif correspondent aux comportements souhaités. Toutefois un IDS est généralement imparfait et conduit à l'apparition des deux autres comportements non désirés. Parmi eux, un faux négatif correspond à une attaque non détectée, et un faux positif à l'émission d'une fausse alerte. Les différents IDS souffrent généralement d'imperfections donnant lieu à l'apparition de ces comportements non désirés, mais selon des axes différents suivant les méthodes de détection qu'ils utilisent.

Un reproche fréquemment fait en direction des IDS utilisant une méthode de détection comportementale est de contenir dans leur principe même de fonctionnement la possibilité de fausses alertes (un changement de comportement légitime détecté comme anormal) ou de faux négatifs (par exemple pour une attaque très lente) ; tandis que les approches par scénarios semblent théoriquement être plus exactes. Toutefois, la base de connaissance utilisée dans les IDS par scénarios exige une maintenance constante et, dans la pratique, souffre également nécessairement d'imperfections. Malgré tout, la réputation des IDS comportementaux semble avoir souffert durablement de leur imperfection de principe (à notre sens de manière assez injustifiée), notamment du fait de la possibilité de faux négatifs.

Bien que les faux négatifs soient effectivement le premier des comportements indésirables pour un IDS, les faux positifs sont importants aussi : ils peuvent conduire à une réelle perte de confiance dans les capacités de détection de l'IDS de la part des administrateurs qui peut finir par remettre en cause la finalité de l'IDS. C'est même une des voies d'attaque envisageables contre un système équipé d'un IDS : générer un nombre suffisamment important de fausses alertes pour réduire l'attention des administrateurs et dissimuler une attaque réelle. De plus, dans la pratique, les faux

2 Bien que la distinction ne soit pas très marquée dans la pratique, il est utile de distinguer l'audit de bas niveau disponible au niveau des systèmes d'exploitation (qui peut aller jusqu'à des traces listant les appels systèmes exécutés) des informations d'audit de plus haut niveau, généralement plus riches de sens mais moins exhaustives, fournies par certaines applications.

3 Par exemple du fait d'un HIDS analysant des fichiers traces transmis seulement toutes les heures.

positifs dus à l'environnement de l'IDS ou à des signatures d'attaque un peu trop affirmatives sont souvent nombreux ; et ceci nécessite généralement un reparamétrage de l'IDS pour faciliter son exploitation, au prix de l'introduction de possibilités de faux négatifs. La gestion des faux positifs est le premier problème auxquels sont confrontés les administrateurs d'un IDS, et il est généralement de taille. A notre sens, les IDS basés sur une approche par scénarios, c'est à dire la plupart des IDS courants, souffrent sur ce point d'un réel problème qui demanderait certainement de développer à la fois les possibilités d'adaptation de l'IDS à son environnement (peut-être par des moyens de corrélation) et une meilleure validation des signatures d'attaque disponibles.

L'utilisation de techniques de corrélation d'alertes provenant de plusieurs IDS semble être une des voies envisageables pour traiter ces problèmes d'analyse des alertes et notamment des fausses alertes. Dans ce cadre, la diversification des méthodes de détection utilisées par les différents IDS, ainsi que de leurs sources de données est aussi à nouveau[4] envisageable.

## *1.2 Approches étudiées et tendances*

Les différentes approches de la détection d'intrusion présentées dans la terminologie de la figure 1 ont pour la plupart toutes été étudiées au niveau de système expérimentaux, notamment dans les laboratoires de recherche ou les universités.

Toutefois, à cette diversité des expérimentations s'oppose une focalisation quasi exclusive des solutions techniques disponibles dans le domaine industriel sur les systèmes de détection d'intrusion réseau utilisant une approche par scénarios, et plus précisément de signatures d'attaques. La quasi-totalité des IDS commerciaux utilisent cette approche, peut-être parce que c'était l'une de celles qui étaient les plus directes à mettre en œuvre et à déployer dans les systèmes informatiques courants. Des solutions existent également pour la détection d'intrusion à partir des traces systèmes, toujours en utilisant généralement une approche par scénario (recherche d'évènements dans les traces). Toutefois, la plupart de ces HIDS utilisent les traces standards des systèmes d'exploitation, généralement assez peu exhaustives (en tout cas en comparaison des fonctions d'audit système disponibles sur les systèmes C2 et supérieurs dans la classification du livre orange - ceci dit celles-ci absorbent une part significative des ressources d'une machine seulement dans l'objectif de surveiller l'autre part et sont rarement déployées).

Concrètement, ce sont donc les systèmes de détection d'intrusion réseau utilisant des bases de signatures qui dominent les mises en œuvre opérationnelles.

## *1.3 Schéma d'architecture IDWG*

Plusieurs schémas ont été proposés pour décrire les composants d'un système de détection d'intrusion. Parmi eux, nous avons retenu celui issu des travaux de l'*Intrusion Detection exchange format Working Group* (IDWG)[5] de l'*Internet Engineering Task Force* (IETF)[6] comme base de départ, car il résulte d'un large consensus parmi les intervenants du domaine. L'objectif des travaux du groupe IDWG est la définition d'un standard de communication entre certains composants d'un système de détection d'intrusion. Ce standard de communication s'appuie sur le modèle d'architecture présenté dans la figure 2.

L'architecture IDWG contient des capteurs qui envoient des événements à un analyseur. Un ou des capteurs couplés avec un analyseur forment une sonde. Une sonde envoie des alertes vers un logiciel de gestion (*manager*) qui la notifie à un opérateur humain.

4 Dans un certain sens, il s'agit d'ailleurs de ré-inventer la roue une fois de plus puisque le précurseur des systèmes de détection d'intrusion, nommé IDES, combinait déjà l'utilisation d'une approche comportementale (statistique) et d'une approche à base de règles (système expert), dans les années 1980 du côté de Stanford.

5 http://www.ietf.org/html.charters/idwg-charter.html

6 http://www.ietf.org/

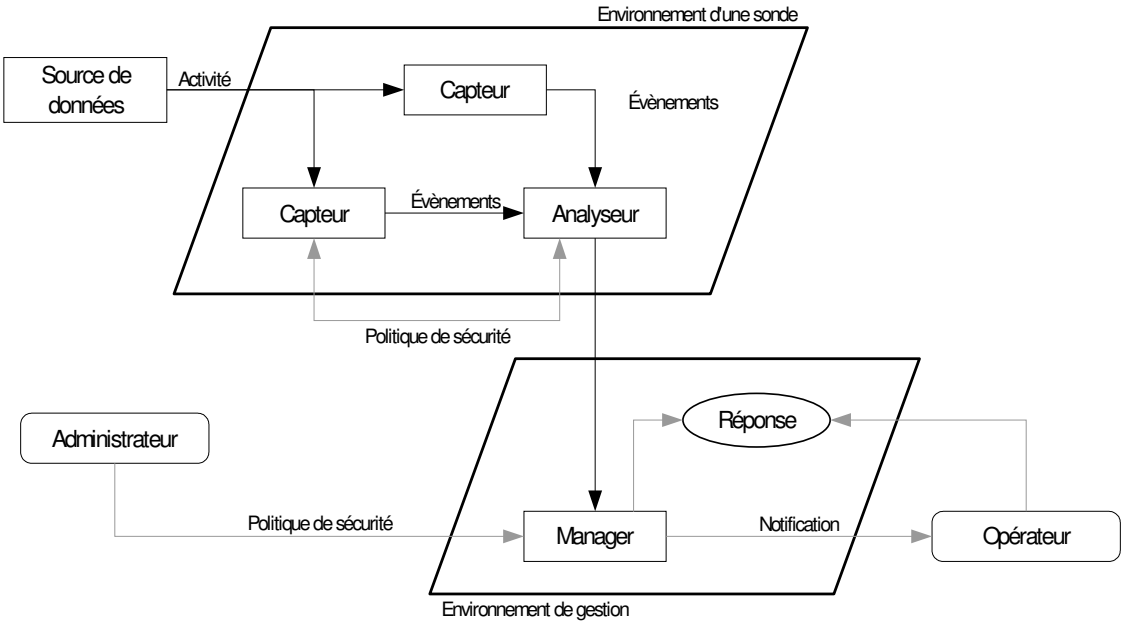



*Illustration 2: Schéma d'architecture IDWG d'un IDS*

## *1.4 Exemple d'architecture*

Dans la pratique, le modèle fonctionnel d'architecture présenté précédemment doit être adapté pour une mise en place concrète des différents éléments techniques du système de détection d'intrusion. Nous présentons dans la figure 3 un exemple de déploiement de système de détection d'intrusion qui nous semble un peu plus représentatif des contraintes réelles de mise en place.

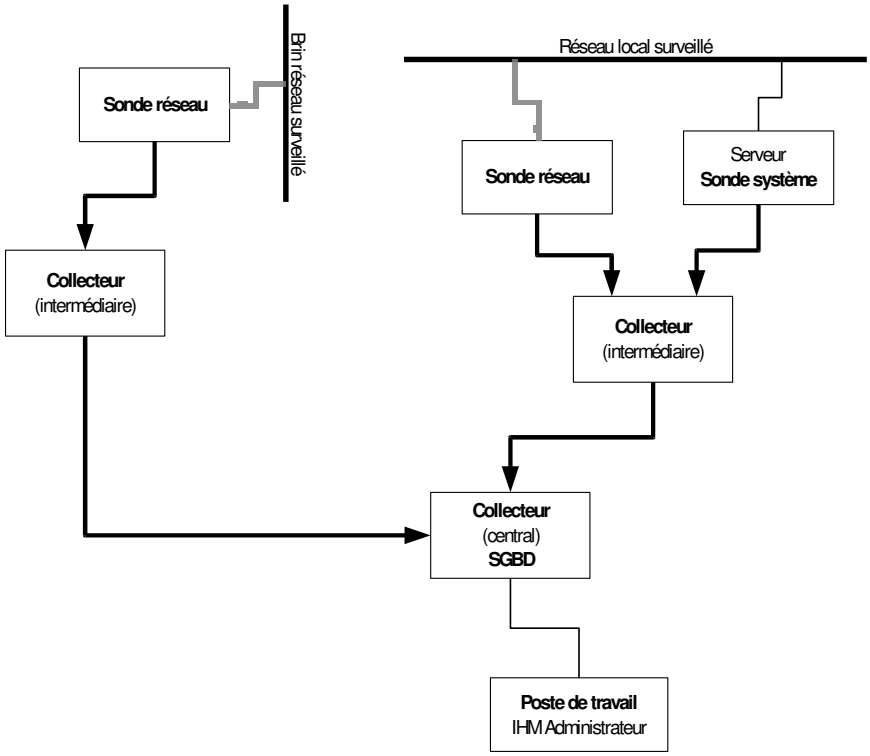



*Illustration 3: Architecture envisageable pour un IDS*

Dans ce schéma, nous faisons apparaître trois sondes de détection d'intrusion : deux sondes réseau et une sonde système. On peut par exemple envisager que les deux sondes réseaux sont déployées à des endroits différents du système informatique l'une au niveau du réseau interne, l'autre par rapport à un point réseau intéressant (on peut aussi

les envisager séparées sur des sites distants les uns des autres ou par des *firewall*). La sonde système peut par exemple être associée à un serveur de centralisation de traces mis en place sur le réseau local. Dans la mise en place présentée sur la figure 3, des collecteurs intermédiaires ont été ajoutés à proximité de deux groupes de sondes : la sonde réseau surveillant le brin réseau distant, et les deux sondes réseau et système s'intéressant au réseau interne. Ces deux collecteurs intermédiaires propagent ensuite les traces vers un collecteur central associé à un SGBD sur lequel les administrateurs de sécurité peuvent consulter et traiter les alertes.

Dans la pratique, ces collecteurs intermédiaires n'apparaissent généralement pas explicitement. Ils correspondent plus ou moins aux capacités de stockage temporaire et de transmission en différée des sondes vers leur *manager*. Le collecteur central correspond lui au *manager* des différentes sondes et offre à la fois des fonctions de gestion des sondes et de consultation des alertes. Toutefois, à notre sens il est intéressant de les faire apparaître explicitement pour mettre en évidence certaines des fonctions qui peuvent être utiles au niveau de la collecte et du transport des alertes.

Les collecteurs intermédiaires peuvent gérer les flux réseaux et optimiser le transport des alertes en tenant compte, par exemple, des problèmes de disponibilités du collecteur central.

Si des fonctions de corrélation ou de groupement d'alertes sont ajoutées dans l'architecture de sécurité, celles-ci peuvent éventuellement être mises en œuvre de manière distribuée au niveau des collecteurs intermédiaires. Dans une architecture comme celle définie par l'IDWG (voir figure 2) ou dans la plupart des architectures disponibles à l'heure actuelle, l'ensemble des traitements de corrélation doivent être mis en œuvre au niveau du collecteur central (*manager* associé éventuellement à un SGBD). Ce point central peut alors devenir un point critique du point de vue des performances. Par ailleurs, certains des traitements de corrélation simples (comme le groupement d'alertes) peuvent avantageusement être réalisées au plus près des sondes, en tirant ainsi parti d'une distribution des calcul. L'alternative à l'introduction (au moins au niveau logique) des collecteurs intermédiaires est alors de modifier le fonctionnement des sondes, ce qui pose aussi des problèmes de performance.

Les collecteurs intermédiaires peuvent aussi être associés plus étroitement à la problématique de gestion opérationnelle des sondes (lesquelles peuvent provenir de constructeurs différents) voire à la mise en œuvre de la réaction.

D'un point de vue décisionnel, il nous semble que ces réflexions ne se limitent pas à l'aspect de la collecte des alertes (ou même de la corrélation), l'introduction de ce niveau intermédiaire entre les sondes d'analyse et le *manager* correspond en quelque sorte à l'introduction d'un niveau décisionnel tactique entre les agents opérationnels (en contact direct avec le terrain) et un niveau d'analyse stratégique auquel les décisions générales d'action peuvent être prises avec plus de recul.

Par contre, dans la pratique, les architectures mise en oeuvre correspondent classiquement à la configuration présentée dans la figure 4 (très proche de celle identifiée par l'IDWG).

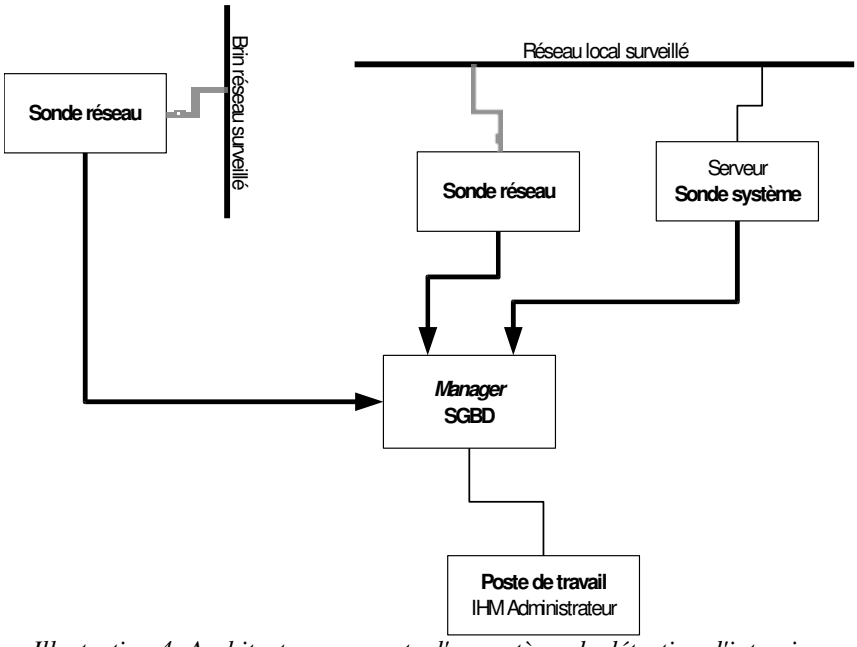



*Illustration 4: Architecture courante d'un système de détection d'intrusion*

## *1.5 Traitement des alertes*

### 1.5.1 Limites

Notamment dans le domaine des sondes de détection d'intrusion réseau, les techniques de détection utilisant une application de signatures sur les évènements observés vont avoir pour effet de sélectionner certains des évènements, contenant un marqueur spécifique à l'origine de l'activation d'une signature. La plupart du temps, chacun des évènements contenant ces marqueurs va être à l'origine de la génération d'une alerte. Pourtant, en général, il serait souhaitable que les alertes soient d'un niveau d'abstraction plus élevé, et qu'elles rassemblent différents marqueurs correspondant à des évènements différents mais associés à une même action.

Dans cette vision, il ne s'agirait pas véritablement de demander à l'IDS d'établir des liaisons de corrélations (cause à effet, topologie, etc.) mais plutôt d'agréger correctement les alertes générées de manière à garantir la correspondance entre chaque alerte et au plus une action. Toutefois, ce besoin correspond à la mise en oeuvre d'une analyse multi-évènements : une même action pouvant être à l'origine de plusieurs évènements (par exemple, le déclenchement d'une attaque réseau peut conduire à la génération de plusieurs paquets IP[7]), la sonde devrait être capable de rechercher des similarités entre marqueurs appartenant à plusieurs d'entre eux. Elle devrait donc pouvoir réaliser une analyse multi-évènements.

Dans la représentation de ce problème que nous donnons dans la figure 5, la forme des marqueurs est ce qui permet à la sonde de les repérer mais, suivant que la couleur des évènements auxquels ils appartiennent est prise également en compte ou non, le nombre d'alertes générées est différent. Ainsi, la génération de l'alerte $A_4$ pourrait être évitée.

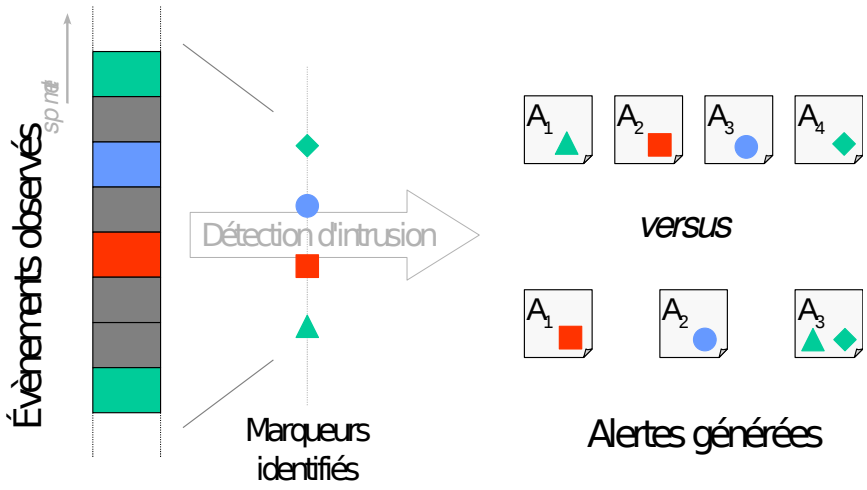



*Illustration 5: Analyse multi-évènements*

Toutefois, on voit bien sur cette illustration que l'analyse multi-évènements exige de prendre en compte les relations existant entre deux évènements successifs, parfois séparés dans le temps par un grand nombre d'autres évènements, et même d'autres alertes. Le travail d'analyse de la sonde est donc largement plus difficile. Par ailleurs, l'alerte $A_3$ dans le cas multi-évènements est générée tardivement par rapport à l'alerte $A_1$ dans le cas mono-évènement. Si l'agrégation de plusieurs marqueurs facilite le diagnostic sécurité, d'un autre côté, ce retard peut aussi être préjudiciable à la sécurité. Ce compromis de mise en oeuvre complique également la conception de la sonde.

Pour aborder ce type de traitement, que ce soit au niveau des sondes elles-mêmes ou au niveau des systèmes de collecte des alertes, il nous semble que les IDS actuels devraient intégrer des fonctions nouvelles, généralement regroupées sous la désignation de corrélation d'alertes.

### 1.5.2 Architecture de corrélation d'alertes

L'architecture IDWG n'est pas totalement adaptée pour couvrir complètement les besoins de la corrélation d'alertes. Une architecture mieux adaptée est présentée dans la figure 6, qui dissocie les consoles de gestion des différentes sondes et la console des alertes qui peut bénéficier des traitement d'autres outils de concentration d'alertes.

7 Et même beaucoup dans les cas où des techniques d'évitement de la détection sont utilisées.

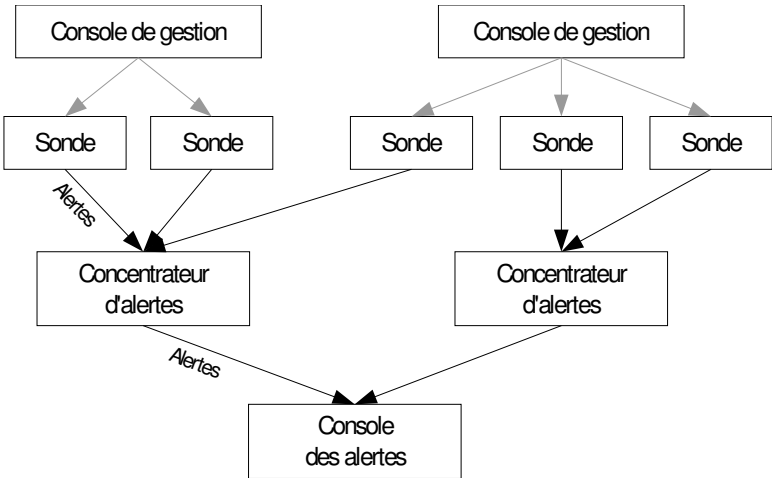



*Illustration 6: Architecture pour la corrélation d'alertes*

Dans cette architecture, les concentrateurs d'alertes peuvent jouer un rôle vis à vis de la collecte et du transport, mais ils peuvent également traiter les alertes pour réaliser des groupements ou certaines corrélations et fournir au niveau de la console des alertes une information agrégée (à la place ou en complément des alertes originelles). Les fonctions de corrélation les plus avancées ou nécessitant la connaissance de l'ensemble des alertes générées dans le système, doivent elles être réalisées au niveau de cette console. (La console n'est pas nécessairement associée directement à l'interface homme-machine de l'IDS.) Les associations entre les différentes sondes et les concentrateurs d'alertes sont établies en fonction de ces possibilités de corrélation, et elles peuvent être différentes des associations nécessaires pour la gestion technique des sondes réalisée par les consoles de gestion. Les secondes peuvent être imposées par les constructeurs par exemple, tandis que les premières peuvent s'appuyer sur la nature des sondes (par exemple, NIDS d'une part, HIDS d'autre part).

### 1.5.3 Groupement

Parmi les fonctions de corrélation envisageables, la première que nous appellerons plutôt une fonction de groupement consiste à rassembler dans une même alerte (virtuelle) des alertes similaires provenant d'un même événement.

Ces groupements peuvent permettre, par exemple, de regrouper des alertes émises par des sondes différentes observant plusieurs fois le même événement (par exemple le long de son trajet pour des sondes réseau). Ce type de regroupement n'est pas si facile à réaliser : dans le cas de sondes utilisant des bases de connaissance différentes, les nomenclatures des attaques peuvent être très différentes et nécessiter la mise au point de tables de correspondance importantes. De tels regroupements sont également plus intéressants qu'il n'y paraît : outre qu'ils diminuent le nombre d'alertes à traiter pour l'opérateur, ils peuvent permettre d'enrichir la description de l'alerte virtuelle globale. En effet, des sondes différentes ont pu renseigner des informations différentes dans les indications de leurs alertes (adresse IP pour l'une, nom de compte pour l'autre par exemple).

Les autres types de regroupement les plus immédiats consistent à rassembler certaines attaques concernant une même source ou une même destination. Ainsi, un *scan* (une recherche active de vulnérabilités) réalisé par un outil donné (ou certains virus) donnera probablement lieu à de nombreuses alertes qui peuvent être regroupées avec une bonne fiabilité si elles suivent une séquence spécifique et ciblent la même destination. C'est également le cas si des alertes similaires provenant de la même source visent des destinations successives. Dans un cas comme dans l'autre, le regroupement est certainement à faire avec une certaine prudence, mais le gain en terme de lisibilité pour l'alerte virtuelle globale par rapport à l'ensemble des alertes élémentaires est très important, tout comme le gain de temps pour les opérateurs.

Enfin, une autre catégorie de regroupement qui semble souhaitable, et qui est liée à la précédente, c'est bien entendu le regroupement temporel. Des alertes périodiques, ou des alertes ayant lieu dans un intervalle de temps réduit peuvent donner lieu à un premier regroupement, ce qui est d'autant plus intéressant si elles présentent des similarités du type évoqué précédemment. Toutefois, les intervalles de temps à utiliser sont difficiles à évaluer ; surtout si on prend en compte le fait qu'un attaquant intelligent puisse prévoir ce type de regroupement et tenter de l'éviter ou de le détourner. La notion de proximité temporelle, pour des alertes de sécurité, reste à manier avec précaution. Mais c'est une information très importante qu'il est impossible d'ignorer quand on réalise un traitement des alertes générées par les IDS. Un effort d'automatisation sur ce point nous semble vraiment souhaitable.

### 1.5.4 Corrélation

Outre le regroupement des alertes en alertes virtuelles plus faciles à gérer pour les administrateurs, on pourrait également attendre d'un IDS qu'il soit en mesure de réaliser un suivi de séquences d'alertes afin d'essayer de fournir un diagnostic plus complet d'une intrusion éventuelle ou d'éliminer des fausses alarmes.

Ce raisonnement part de l'hypothèse qu'en cas d'intrusion, les actions d'attaque effectuées par un intrus suivront une certaine logique d'enchaînement et que des attaques successives (et donc des alertes successives) présenteront des liens entre elles permettant à la fois de confirmer l'alarme suggérée par chaque alerte prise isolément, mais également d'établir un diagnostic plus complet de l'intrusion, de ses objectifs, de son niveau d'avancement, etc. ; et donc du danger associé.

Pour réaliser de telles corrélations, un IDS devrait disposer d'une base de connaissance décrivant non seulement les attaques élémentaires, mais également les enchaînements possibles entre différentes attaques élémentaires pouvant constituer un scénario d'intrusion plus élaboré. Il s'agit schématiquement de décrire les liens de cause à effet entre attaques afin de permettre à l'IDS de raisonner sur les liens existant entre les alertes qu'il connaît (ou plus précisément les attaques auxquelles elles correspondent). Dans certains cas, on peut même envisager que l'IDS soit en mesure d'effectuer des hypothèses sur l'existence de certaines actions non-observées (ou inobservables) afin de compléter des scénarios d'intrusion. Enfin, en établissant le lien entre les préconditions des attaques et les vulnérabilités connues recensées dans le système, ou entre les effets de ces attaques et les objectifs de sécurité du système, l'identification des possibilités de succès de l'intrusion et de son niveau de danger dans le système visé pourrait également être envisagée. De telles fonctions permettraient bien évidemment de faciliter énormément le travail de l'administrateur de sécurité chargé de superviser l'IDS auquel serait fourni une proposition de diagnostic et un niveau de risque plus réfléchis, basés sur les évènements observés par les différentes sondes du système.

L'approche de la corrélation que nous présentons ici est avant tout basée sur une vision logique. Le fonctionnement du moteur de corrélation et la structure du diagnostic sont basés sur des raisonnements logiques concernant les attaques, leur faisabilité, les possibilités d'enchaînement, etc. D'autres approches ont été proposées, appuyées sur des techniques statistiques de caractérisation ou de classification des alertes. Dans tous les cas, ces travaux n'ont pour l'instant pas réellement vus de mise en œuvre effective dans des implémentations opérationnelles de système de détection d'intrusion. Par contre, un certain nombre d'efforts ont déjà était faits dans certaines implémentations pour faciliter le rapprochement entre diverses sources d'information, notamment en vue de limiter le nombre de fausses alertes.

# 2 Approche réseau

Comme nous l'avons déjà mentionné, la plupart des solutions existantes en matière de détection d'intrusion concernent la mise en place de NIDS (IDS réseau) complétés éventuellement par certains HIDS (IDS système) et du logiciel de gestion associé (*manager*). Dans cette section, nous étudierons une des solutions les plus populaires : le NIDS *Snort*. Nous nous intéresserons également à une solution complète d'abord *open-source* et désormais associé à une offre commerciale et qui capitalise d'ailleurs sur certains des succés de *Snort*, le système *Prelude-IDS*.

## *2.1 Snort*

Face aux systèmes de détection d'intrusion complets proposés dans les offres commerciales intégrant sondes, signatures, moyens de distribution, interface graphique, etc. ; les solutions *open-source* ont progressivement trouvé leur place en tirant partie des avantages d'un développement dans le cadre du logiciel libre. Ainsi, l'IDS le plus répandu à l'heure actuelle est certainement le logiciel *open-source* Snort qui est une sonde de détection d'intrusion réseau (NIDS). Le modèle *open-source* appliqué à Snort a notamment permis un développement rapide d'une large base de signatures (plus de 12000 à ce jour) appuyée essentiellement sur le volontariat et un langage de définition totalement public. Cette base est désormais mise à jour très rapidement et diffusée par Internet (sur le Web par exemple). Par ailleurs, la limitation du projet à la réalisation d'une sonde de détection d'intrusion réseau a permis des améliorations successives très importantes en terme de performance et de capacités d'analyse et a également débouché sur la mise à disposition ultérieure d'extensions orientées vers la diffusion des alertes, le stockage dans les bases de données, etc.

Les signatures Snort font désormais partie du bagage que les administrateurs de sécurité doivent pouvoir utiliser pour envisager la détection d'intrusion et la réaction rapide à de nouvelles attaques (ne serait-ce que pour disposer des détails précis de caractérisation d'un flux d'attaque). A titre d'exemple, nous présentons dans les figures 7 et 8 deux signatures d'attaque (et leurs commentaires) utilisables par Snort respectivement pour détecter une tentative d'exploitation d'une

faille grave du service RPC de plusieurs versions de *Microsoft Windows* et pour repérer le flux généré par un vers baptisé « Klez ».

| | |
|---|---|
| **SID** | 2251 |
| **Message** | NETBIOS DCERPC Remote Activation bind attempt |
| **Signature** | alert tcp $EXTERNAL_NET any -> $HOME_NET 135 (msg:"NETBIOS DCERPC Remote Activation bind attempt"; content:"\|05\|"; distance:0; within:1; content:"\|0b\|"; distance:1; within:1; byte_test:1,&,1,0,relative; content:"\|B8 4A 9F 4D 1C 7D CF 11 86 1E 00 20 AF 6E 7C 57\|"; distance:29; within:16; reference:cve,CAN-2003-0352; classtype:attempted-admin; reference:url,www.microsoft.com/technet/security/bulletin/MS03-026.asp; reference:cve,CAN-2003-0715; sid:2251; rev:1;) |
| **Summary** | This event is generated when an attempt is made to exploit a known vulnerablity in Microsoft RPCSS service for RPC. |
| **Impact** | Denial of Service. Possible execution of arbitrary code leading to unauthorized remote administrative access. |
| **Detailed Information** | A vulnerability exists in Microsoft RPCSS Service that handles RPC DCOM requests such that execution of arbitrary code or a Denial of Service condition can be issued against a host by sending malformed data via RPC.<br><br>The Distributed Component Object Model (DCOM) handles DCOM requests sent by clients to a server using RPC. A malformed request to the host running the RPCSS service may result in a buffer overflow condition that will present the attacker with the opportunity to execute arbitrary code with the privileges of the local system account. Alternatively the attacker could also cause the RPC service to stop answering RPC requests and thus cause a Denial of Service condition to occur. |
| **Affected Systems** | Windows NT 4.0 Workstation and Server<br>Windows NT 4.0 Terminal Server Edition<br>Windows 2000<br>Windows XP |

*Illustration 7: Signature Snort pour une attaque vers MS/RPC*

On voit que ces signatures correspondent à la recherche de séquences particulières dans un flux réseau TCP (sur des ports ou pour des services particuliers). La sonde Snort se charge d'effectuer le ré-assemblage de la séquence TCP parmi tous les paquets IP se présentant devant son interface de capture malgré toutes les techniques de dissimulation éventuellement utilisées (fragmentation, déséquencement, etc.) et de réaliser la recherche des différentes signatures efficacement.

La qualité des signatures disponibles autour de Snort (que ce soit dans la distribution principale du logiciel, ou sur le site Web associé[8]) semble globalement bonne[9] ; même si elles sont fournies sur une base de volontariat assisté par l'implication du principal distributeur (Sourcefire).

8 http://www.snort.org/dl/rules/ et http://www.snort.org/snort-db/.

9 Ce qui ne veut pas dire qu'elles ne génèrent pas quand même un nombre important de fausses alarmes dans un réseau actif...

| | |
|---|---|
| **SID** | 1800 |
| **Message** | VIRUS Klez Incoming |
| **Signature** | alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"VIRUS Klez Incoming"; flow:to_server,established; dsize:>120; content:"MIME"; content:"VGhpcyBwcm9"; classtype:misc-activity; sid:1800; rev:3;) |
| **Summary** | This event is generated when an incoming email containing the Klez worm is detected. |
| **Impact** | System compromise and further infection of target hosts. |
| **Detailed Information** | W32/Klez.h@MM exploits the vulnerability in Microsoft Internet Explorer (ver 5.01 or 5.5 without SP2), enabling it to execute email attachments.<br><br>Once executed, it can unload several processes including Anti-virus programs.<br><br>The worm is able to propagate over the network by copying itself to network shares (assuming sufficient permissions exist). Target filenames are chosen randomly, and can have single or double file extensions. |
| **Affected Systems** | Microsoft Internet Explorer (ver 5.01 or 5.5 without SP2) |
| **Attack Scenarios** | This virus can be considered a blended threat. It mass-mails itself to email addresses found on the local system, then exploits a known vulnerability, spreads via network shares, infects executables on the local system. |
| **Ease of Attack** | Simple. This is worm activity. |
| **False Positives** | Certain binary file email attachments can trigger this alert. |
| **False Negatives** | None known. |
| **Corrective Action** | Apply the appropriate vendor suppled patches.<br><br>Block incoming attachments with .bat, .exe, .pif, and .scr extensions |
| **Contributors** | Sourcefire Research Team<br>Brian Caswell <bmc@sourcefire.com><br>Nigel Houghton <nigel.houghton@sourcefire.com><br>Snort documentation contributed by Nawapong Nakjang (tony@ksc.net, tonie@thai.com) |
| **References** | |

*Illustration 8: Signature Snort pour le virus Klez*

### 2.1.1 Outils annexes

Snort étant un logiciel focalisé sur la fonction de détection d'attaque réseau proprement dite, avec le temps il a conduit à l'apparition d'un certain nombre d'outil complémentaires assistant à son fonctionnement. Ces outils sont des éléments importants pour le bon fonctionnement du NIDS et nous détaillons les principaux dans cette section.

#### *2.1.1.1 Oikmaster*

Comme indiqué précédemment, la mise à jour des signatures de détection de Snort est un facteur important de la qualité de détection offerte par la sonde de détection d'intrusion.

Le développement de Snort est soutenu par une société, Sourcefire, qui met également à disposition des utilisateurs des mises à jour de la base de détection. Toutefois, hors la version de développement, la version la plus à jour est offerte selon des modalités commerciales. Une version retardée de 30 jours est toutefois disponible modulo un simple enregistrement (d'utilisateur physique) auprès de Sourcefire.

Un outil largement répandu pour automatiser le téléchargement et l'intégration de nouvelles signatures dans une configuration opérationnelle de sonde Snort est l'utilitaire Oinkmaster (http://oinkmaster.sourceforge.net/). Désormais stabilisé depuis 2008 environ, ce script permet de télécharger automatiquement les derniers fichiers de signatures et de les joindre à une base de règles existantes (en tenant éventuellement compte de modifications locales et des codes d'enregistrement nécessaires auprès de Sourcefire).

Son utilisation peut être aussi simple que : `oinkmaster -c -o /etc/snort/rules/` ; sachant que l'option `-c` (*check*) doit être retirée pour modifier effectivement la configuration et inscrire, par exemple, la commande en exécution régulière via `cron`.

#### *2.1.1.2 Barnyard*

Nativement, Snort enregistre les alertes qu'il émet dans un fichier de traces en format texte ou dans un format binaire spécifique plus compact. L'exploitation des alertes ainsi enregistrées (qui peuvent parfois être très nombreuses) est alors relativement malcommode et nécessite de bien maîtriser le fonctionnement de la sonde elle-même. Une attente fréquente est d'enregistrer les alertes ainsi émises dans un système de gestion de base de données (SGBD) afin de pouvoir les analyser plus commodément.

Même s'il existe des composants d'émission en direction de SGBD dans Snort, l'association de Snort à un SGBD pose des problèmes techniques parfois assez complexes : la correspondance entre le modèle de données de stockage et les structures de données internes de Snort (très optimisées pour la détection des signatures), la gestion des performances notamment afin que les performances de l'enregistrement dans le SGBD ne viennent pas avoir un impact sur le moteur d'analyse, etc.

Les développeurs de Snort ont alors volontairement préféré laisser la gestion de ce type de fonctionnalités (au sens large) en dehors du développement de la sonde elle-même en les laissant à la charge d'un programme externe.

Le programme additionnel le plus courant visant à offrir ce type de fonction est désigné sous le nom de Barnyard. C'est une couche applicative qui exploite les événements générés par Snort pour les inscrire dans une base de données en prenant éventuellement en charge les interruptions temporaires de transmission, etc.

Un successeur de Barnyard est désormais également disponible sous le nom de Barnyard2 (http://www.securixlive.com/barnyard2/index.php).

#### *2.1.1.3 Sguil+Squert*

Notamment à partir des alertes stockées dans un SGBD, la problématique de traitement des alertes par l'administrateur système ou réseau, c'est à dire l'utilisateur de l'IDS reste à compléter avec Snort. Cette sonde a donc donné lieu à l'apparition de plusieurs variantes de console visant à offrir des fonctions de visualisation des alertes produites. Nous présentons ici 2 exemples qui nous ont semblé intéressants, mais d'autres sont certainement disponibles.

La première variante est un couple de deux applications nommées respectivement Sguil et Squert[10]. Sguil[11] est une console de visualisation en temps réel des alertes générées, permettant de consulter dans tous ses détails une alerte émise par Snort (y compris éventuellement le contenu des trames réseau capturées). Squert[12] est plutôt orienté vers la consolidation statistique du flux et la compréhension des alertes dans leur contexte. Les 2 outils offrent une interface graphique (l'un via Tcl/Tk, l'autre via un navigateur Web) et correspondent à des implémentations disponibles depuis 2007 environ.

#### *2.1.1.4 Snorby*

Une initiative plus récente visant à offrir une console de gestion des alertes Snort qui nous semble digne d'intérêt est le projet offrant une console complète dénommé Snorby[13]. Partant sur une base technologique assez différente, c'est une application Ruby on Rails pour la gestion de ces alertes qui combine les fonctions précédentes et ajoute l'intégration d'un processus de traitement des alertes par l'utilisateur de la console qui lui permet de les acquitter.

La console Snorby vise aussi à prendre en compte de nouvelles sondes de détection d'intrusion apparues récemment, comme Suricata[14] ou Sagan[15].

10 On notera à cette occasion que la communauté entourant Snort semble posséder une caractéristique tout à fait spécifique - quoique totalement indéfinissable - en matière de dénomination des logiciels.

11 http://sguil.sourceforge.net/index.html

12 http://www.squertproject.org/

13 http://www.snorby.org/

14 http://freecode.com/projects/Suricata

15 http://sagan.softwink.com/

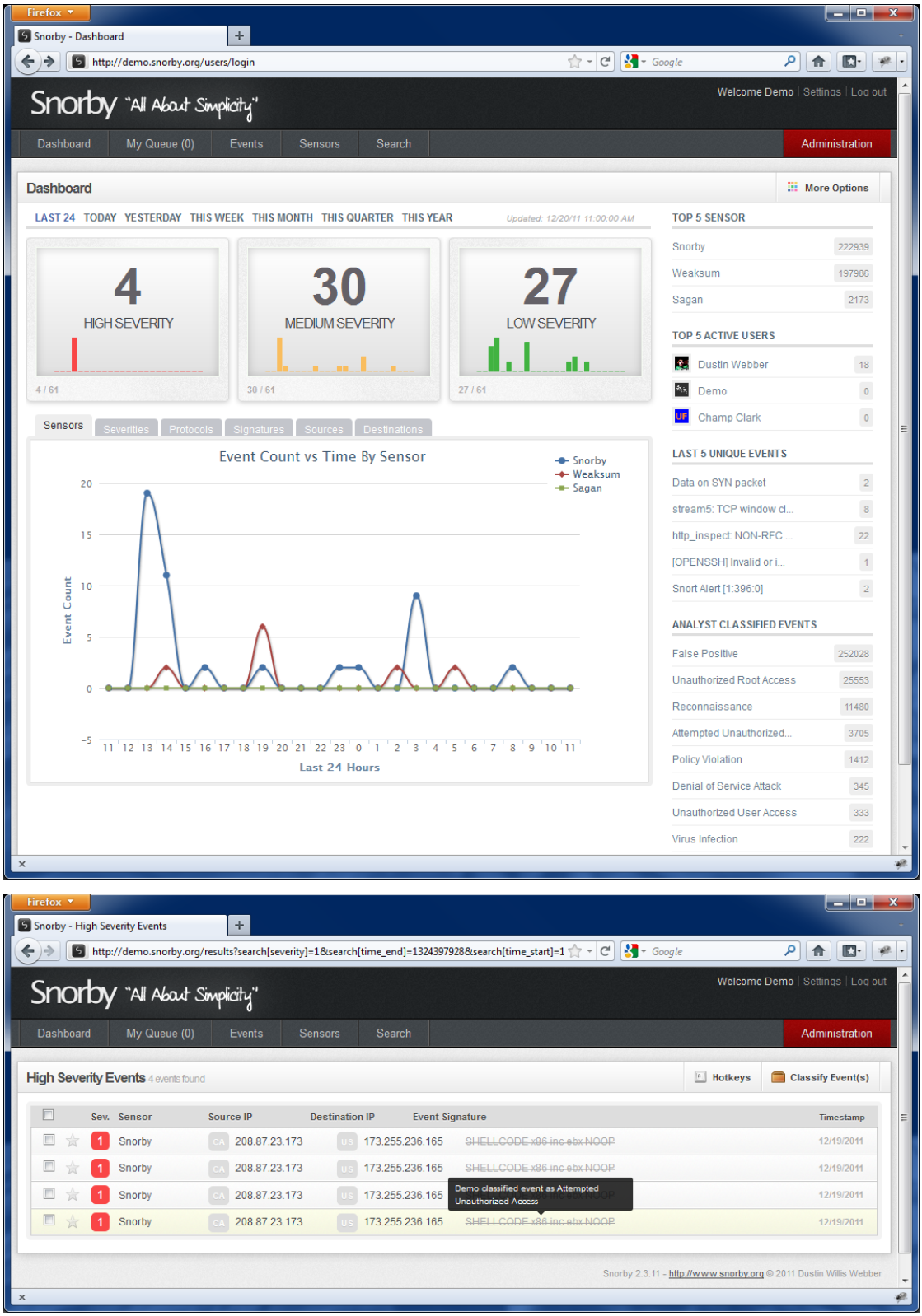



*Illustration 9: Exemples d'interface générale et de classification d'alerte avec Snorby*

## 2.2 Prelude-IDS

Contrairement au projet Snort, focalisé sur la réalisation d'une sonde spécifique, le projet Prelude-IDS propose un système plus complet comprenant :

- une infrastructure logicielle (librairie, format de communication) permettant de développer différentes sondes intégrées dans l'infrastructure Prelude ;
- un *manager* capable de collecter les alertes issues de ces différentes sondes et de les stocker dans une base de données (MySQL en général) ;
- un certain nombre de sondes, avec notamment :
  - un NIDS capable de ré-utiliser les signatures de Snort, et d'effectuer d'autres analyses réseau ;
  - un HIDS permettant d'analyser différents types de traces (en général au format syslog) ;
  - un certain nombre de *patch* permettant d'adapter d'autres logiciels de sécurité pour qu'ils émettent des alertes en direction de Prelude, comptant notamment : Snort lui-même nativement, pflog (traces du *firewall* pf d'OpenBSD), systrace (contrôle des appels système), honeyd (pot de miel) et Nessus (outil de recherche de vulnérabilités) ;
- une interface de visualisation des différentes alertes générées nommée Piwi (en Perl), interface qui est en train d'être remplacée par une autre, nommée Prewikka, en cours de développement pour permettre un contrôle plus complet des autres composants.

D'après notre expérience, l'ensemble logiciel Prelude-IDS est capable de gérer correctement un système de détection d'intrusion composé de 2 ou 3 sondes déployés sur un système informatique réel de bonne taille en production depuis plusieurs années. Les mécanismes de collecte d'alertes et de communication nous semblent donc stables, tout comme le stockage des alertes dans une base de données externe ; même si le schéma de base de données sous-jacent n'est peut-être pas totalement optimal.

Pour l'instant, les possibilités de contrôle du système de manière centralisée depuis l'un des *manager* sont limitées, ces aspects doivent être pris en compte par des moyens d'administration conventionnels, mais comme les différents composants (notamment les sondes) gèrent correctement les arrêt/relance du point de vue de la communication avec le *manager* ceci n'est pas bloquant (avec un nombre limité de sondes).

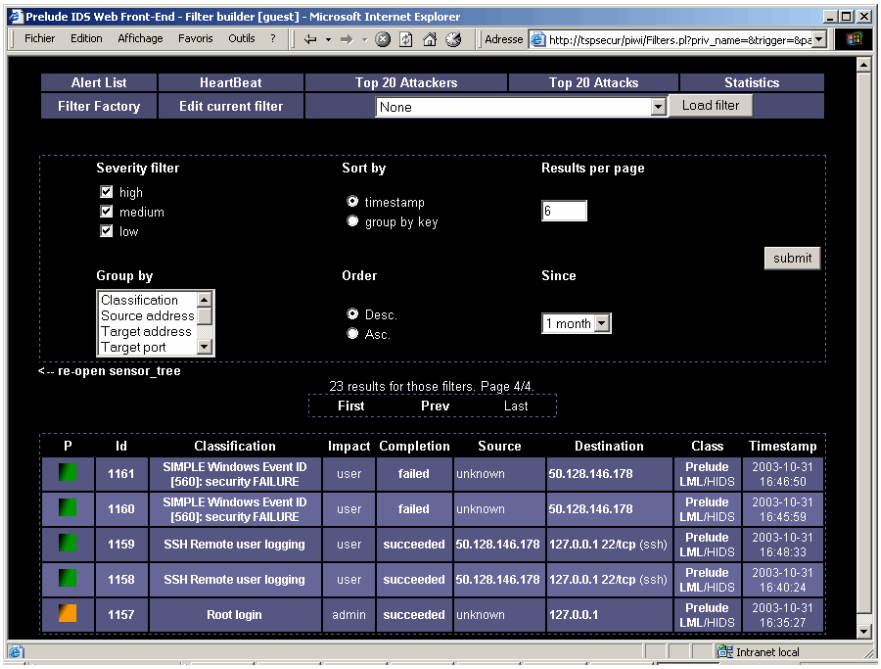



*Illustration 10: Piwi: console historique de visualisation de Prelude-IDS*

L'interface initiale d'accès aux alertes de Prelude était assez limitée. Cette version, Piwi, était limitée à des possibilités de visualisation simples via un navigateur Web et un serveur HTTP (via des cgi Perl). La figure 10 présente un exemple de cette interface.. Il n'est pour l'instant pas possible d'agir sur le contenu de la base de données (ce qui serait, à notre sens, le premier pas vers des fonctions de corrélation et d'agrégation d'alertes).

Une autre version de l'interface a ensuite été proposée, sur des bases totalement différentes. Dénommée Prewikka, cette console est présenté figure 11 en situation réelle. Les travaux autour de cette nouvelle interface de gestion ont ensuite constitué la base de l'offre commerciale entourant Prelude-IDS. Prewikka permet de réaliser un certain nombre de tris et de filtrages sur les requêtes d'interrogation SQL sous-jacentes, mais tout travail d'analyse plus poussé demande d'interroger directement la base de données ou peut-être de disposer des fonctions plus avancées de la version commerciale du système.

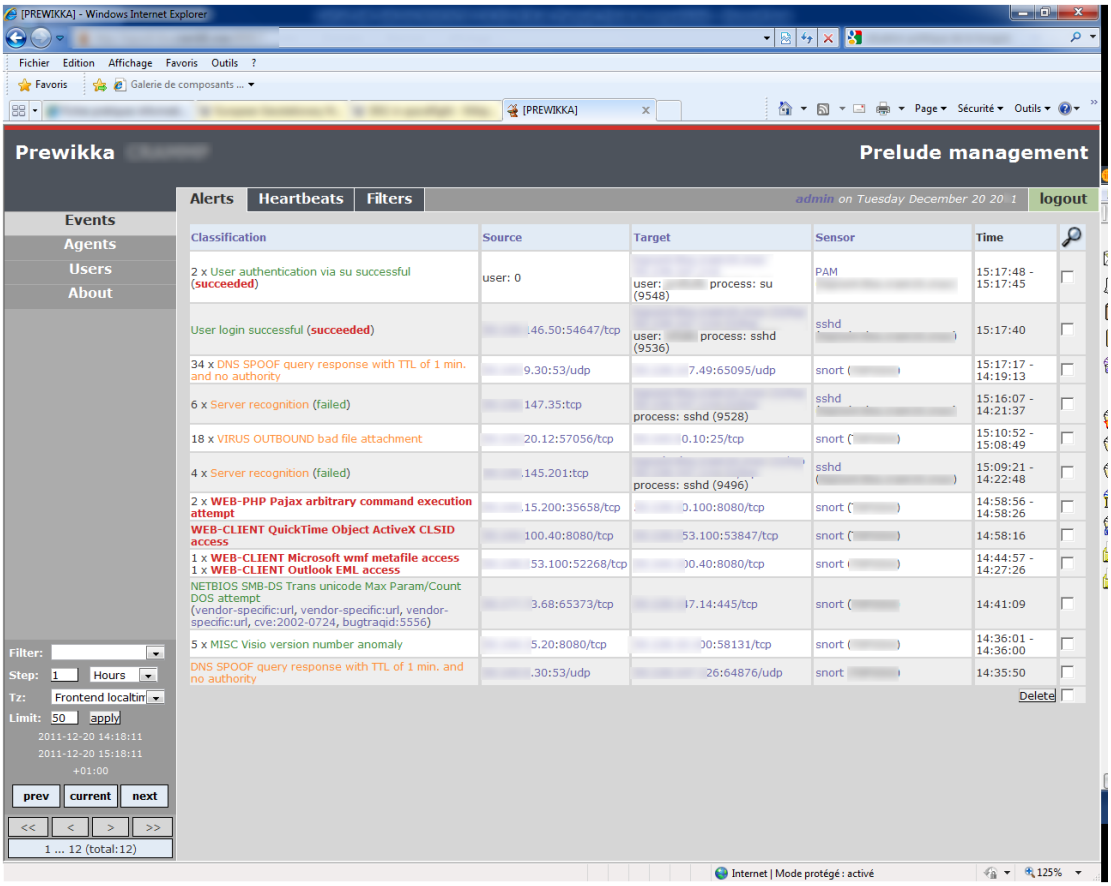



*Illustration 11: Prewikka, console de visuation de Prelude-IDS*

Globalement, Prelude-IDS est donc une solution qui nous semble intéressante, du fait qu'elle a jeté les bases d'un système de détection d'intrusion complet, déjà capable de remplir les fonctions d'une sonde réseau ou système efficace, mais également susceptible de rassembler des informations provenant de plusieurs sondes de différents types dans un même entrepôt de données.

# 3 Centralisation des traces

Au travers de la détection d'intrusion, on voit que la collecte d'information dans le système informatique est une opération importante pour traiter ses problèmes de sécurité. Dans certains cas, la centralisation de cette information est quasiment vue comme une fin en soi, et associée à une propriété de la sécurité qui a été baptisée l'auditabilité. De notre point de vue, cette propriété n'est pas réellement dissociée des autres propriétés de sécurité ; par contre, elle met en relief un certain nombre de besoins de sécurité qui impliquent généralement la collecte et le stockage (souvent de manière centralisée) des sources d'information disponibles dans le système informatique. Nous regroupons ces opérations sous le thème de la centralisation des traces dans le système informatique.

Ce besoin de collecte peut avoir une origine technique, comme dans les cas où ces traces sont nécessaires pour la mise en œuvre d'un système de détection d'intrusion ou la réalisation de contrôles liés à la sécurité. Mais il peut également avoir une origine plus politique, notamment quand la disponibilité de ces données est liée à l'obligation de pouvoir fournir des éléments d'enquête à des organismes habilités, internes (audit interne) ou externes (tutelles, cour des

comptes pour les organismes publics, etc.), voire tout simplement une origine légale pour répondre aux besoins de l'autorité judiciaire. Dans l'ensemble de ces cas, la mise à disposition des traces fournies en standard par les systèmes d'exploitation, les progiciels, les équipements réseau et bien sûr les équipements de sécurité eux-mêmes devrait pouvoir être possible.

Toutefois, pour être réalisable, cette mise à disposition doit réellement avoir été prévue. Par ailleurs, pour être un tant soit peu probantes, les traces collectées doivent présenter un minimum de garanties d'intégrité - la deuxième action d'un attaquant averti étant d'effacer les traces qu'il génère.

Enfin, il ne faut pas oublier que, à partir du moment où ces traces contiennent des informations nominatives - il est quasiment inévitable qu'elles puissent en contenir si elles sont d'un véritable intérêt - elles doivent être gérées en conformité avec les directives de la CNIL[16] et de la loi « Informatique et libertés » (c'est à dire qu'elles ne doivent pas être conservées[17] indéfiniment et qu'elles doivent être protégées pour qu'on ne les détourne pas de leur finalité originelle).

A l'heure actuelle, il nous semble que ces besoins peuvent difficilement être satisfaits autrement que par la mise en place pragmatique d'un système centralisé de consolidation et de stockage des traces, géré directement par les acteurs de la SSI. Il semble même que syslog soit un standard de fait incontournable pour prendre en compte une bonne majorité des types d'équipement existants (même si cela demande des adaptations pour les systèmes basés sur *MS/Windows*).

Dans la pratique, nous recommanderions tout simplement l'utilisation d'un serveur Unix syslog-ng (pour faciliter le tri

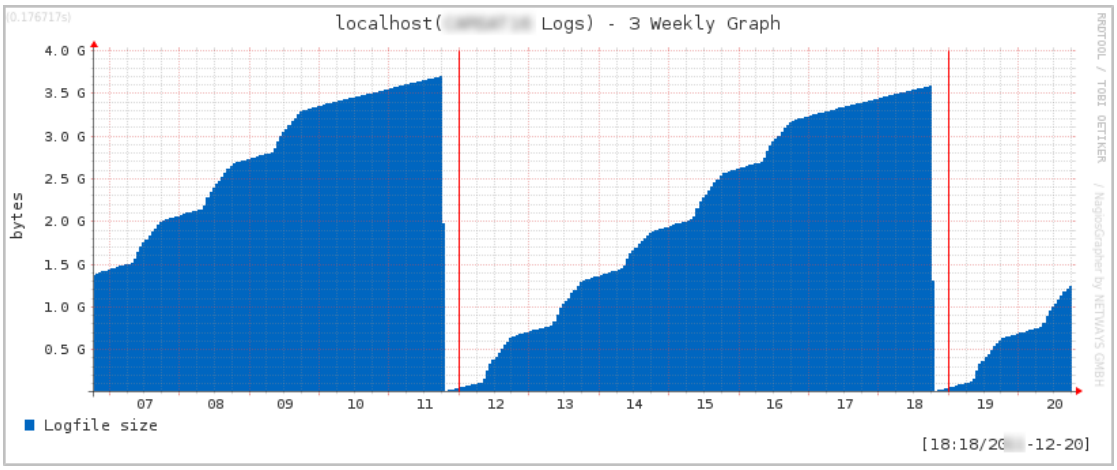



*Illustration 12: Evolution d'un stockage de traces ActiveDirectory*

des traces en fonction de leur origine) détaché du reste du système informatique et décemment configuré du point de vue sécurité, équipé d'une zone de stockage relativement grande[18] et assurant une rotation régulière des traces collectées, par exemple avec l'utilitaire logrotate. (Un système Debian GNU/Linux standard assure ces fonctions sans difficultés dans la configuration par défaut.) Les systèmes incapables d'utiliser Syslog ne sont généralement pas capables de faire plus que de stocker leurs traces dans des fichiers à plat, lesquels dans ce cas doivent être déportés manuellement[19] s'ils sont intéressants, par exemple via SSH.

A moins de pouvoir être effectué dans une situation physique simple, ce type de solution est toutefois peu satisfaisant du point de vue de sa propre sécurité, surtout si on le compare, par exemple, avec les modes de transmission d'alertes de certains IDS (via des connexions SSL authentifiées avec des certificats). Syslog est un protocole fonctionnant sur UDP, notoirement non-sûr. Toutefois, dans la pratique, l'intérêt concret d'un tel système (s'il est utilisé par les autres éléments du système informatique pour stocker leurs traces) est très important, à la fois en terme de données disponibles pour la

16 Voir notamment sur ce point le rapport de la CNIL sur « La cybersurveillance sur les lieux de travail », disponible à l'adresse : http://www.cnil.fr/fileadmin/documents/approfondir/rapports/Rcybersurveillance-2004-VD.pdf.

17 ou sauvegardées...

18 Quoique, en gérant correctement l'espace, un disque dur usuel nous semble déjà constituer un volume de traces suffisamment important à analyser pour occuper un certain nombre d'administrateurs de sécurité pendant un certain temps.

19 Entendre : régulièrement par des scripts, bien entendu.

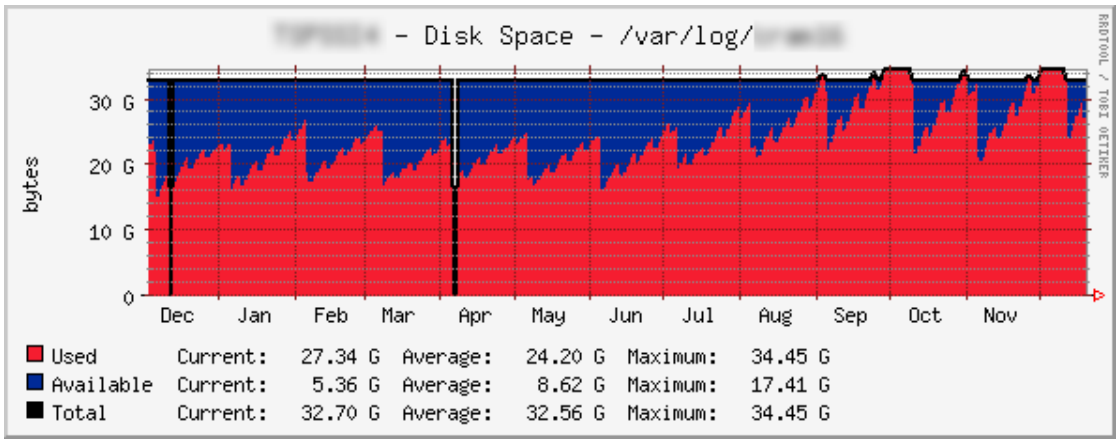



*Illustration 13: Evolution et problématique de long terme*

surveillance de la sécurité et pour pouvoir disposer d'informations en cas d'intrusion grave. Et la protection d'une telle machine est relativement facile à réaliser[20].

## *3.1 Configuration des traces MS/Windows (2000 et ultérieur)*

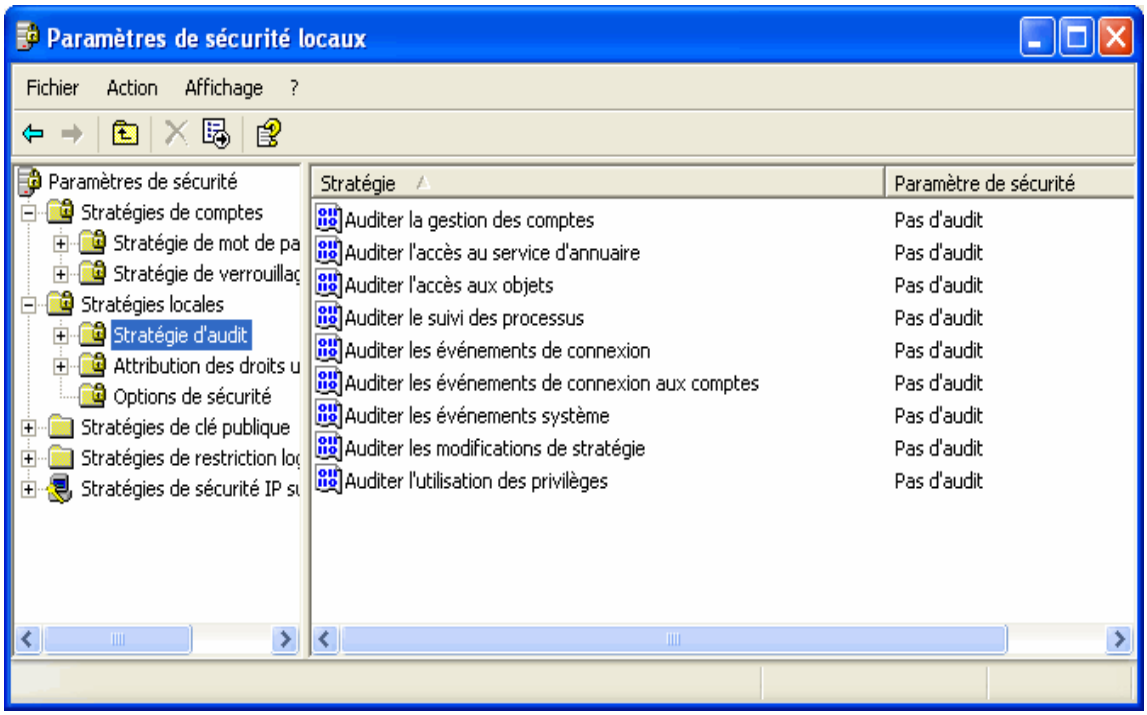



*Illustration 14: Configuration par défaut de l'audit Windows (XP)*

Parmi les différentes traces disponibles dans un système informatique, celles que l'on peut collecter sur les systèmes d'exploitation de la famille de *Microsoft Windows* sont particulièrement communes. On peut en effet généralement disposer de ces traces sur la majorité des postes de travail et une large part des serveurs d'un système informatique d'entreprise usuel. De plus, en y regardant de plus près, ces traces sont assez riches et relativement faciles à activer via les facilités d'administration offertes par l'annuaire centralisé *Active Directory* de *Microsoft*.

Pourtant, ces traces sont certainement sous-exploitées dans la majeure partie des configurations. En effet, leur disponibilité n'est pas toujours connu, leur analyse n'est pas évidente (elles sont très nombreuses) et leur centralisation n'est pas immédiatement réalisable via les seules fonctionnalités du système d'exploitation *Microsoft*. Pour rester dans

20 Il importe d'abord de vérifier qu'on puisse difficilement saturer ses zones de stockage et que cela ne perturbe pas le reste de la machine.

un univers logiciel homogène, il est nécessaire d'acquérir d'autres solutions de l'éditeur (*Microsoft Operations Manager* ou *MOM*) pour la centralisation des traces. Toutefois, des solutions alternatives existent, en s'appuyant sur des logiciels plus spécifiques (et notamment le protocole et les serveurs syslog et le logiciel libre Ntsyslog[21]).

Avant de réaliser leur centralisation, il faut identifier les principaux paramètres de configuration des traces sur le système d'exploitation. Ceux-ci figurent globalement dans la « stratégie d'audit » du système d'exploitation, parmi les « stratégies locales » de sécurité. La figure 14 présente les différents paramètres disponibles ainsi que leur valeur par défaut (dans le contexte graphique de *Windows XP*). On note que par défaut cet audit est *désactivé*.

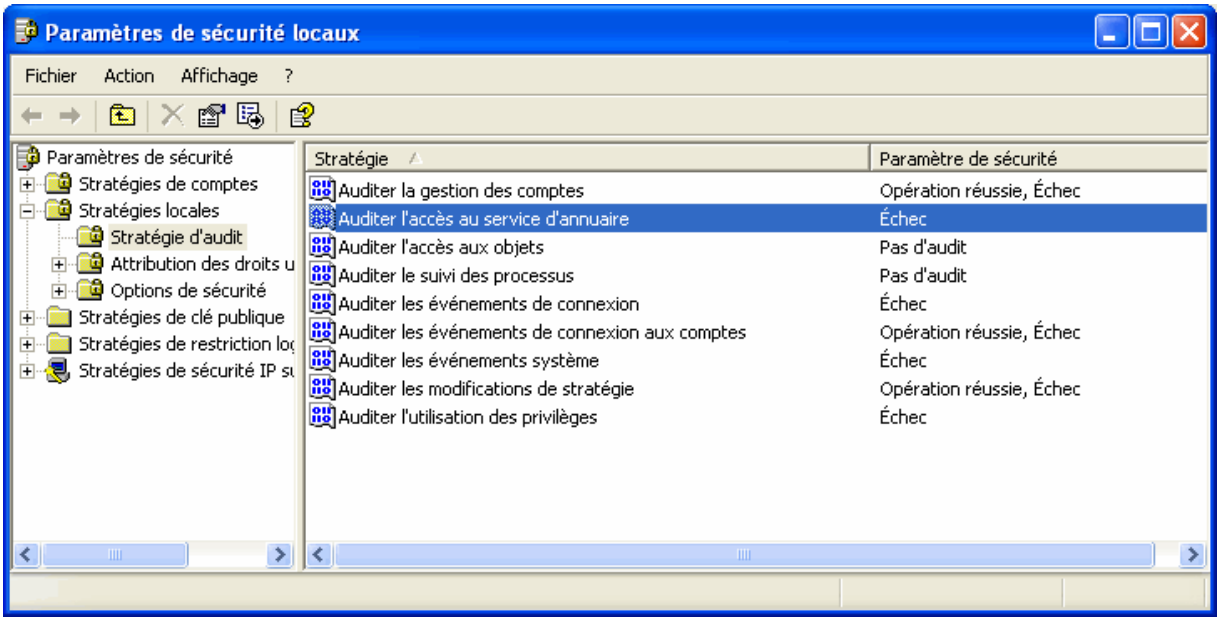



*Illustration 15: Configuration souhaitable de l'audit Windows (XP)*

Dans la figure 15, nous présentons un exemple de configuration recommandée pour l'audit *Windows* de notre point de vue. Il s'agit là d'une configuration assez exhaustive, susceptible d'être restreinte en cas de problèmes de performance, mais qui donne satisfaction dans la plupart des cas. La configuration optimale dépend du système informatique concerné.

D'autres paramètres à prendre en compte pour la configuration de l'audit *Windows* concernent les fichiers journaux utilisés pour le stockage des traces. Ils sont présentés dans la figure 16. Il faut notamment indiquer la taille maximale des fichiers journaux ainsi que la stratégie de remplacement à utiliser quand les journaux sont pleins. Les deux paramètres sont en fait étroitement liés.

Si une taille importante (de l'ordre de la dizaine de MiB) est utilisable ; le fichier journal pourra certainement contenir l'ensemble des traces normalement générées pendant la période de rétention désirée. Dans ce cas, une stratégie de remplacement basée sur l'âge des traces est à notre sens préférable. En effet, elle évite de laisser à un attaquant l'opportunité de masquer ses traces en saturant le fichier journal par un événement anodin répété suffisamment pour entraîner une rotation rapide. Ceci correspond globalement à une politique consistant à conserver les traces sur la machine sur laquelle elles sont collectées.

Si on préfère réserver une taille limitée pour le fichier journal sécurité et si celui-ci n'est utilisé que comme tampon intermédiaire avant déport des traces vers un serveur de centralisation, il est alors préférable d'utiliser une taille très réduite pour le journal (quelques dizaines de KiB) et d'adopter une stratégie de remplacement au besoin pour éviter qu'un pic d'activité (normal) de la machine ne conduise à une perte des traces.

Enfin, il faut penser que, dans la plupart des cas, l'activation des traces ne doit pas se faire à l'aveuglette. La génération, le stockage et la *destruction* de ces journaux doit être maîtrisée. D'abord parce que, par exemple, un serveur syslog mal configuré est un des meilleurs moyens de remplir rapidement un disque dur (intentionnellement ou non). Ensuite parce que, en France, la protection des données personnelles inclut une obligation de protéger ces données et l'utilisation qui en est faite de manière effective ainsi qu'un *droit à l'oubli* dont l'esprit est de prononcer la destruction systématique de

21 http://ntsyslog.sourceforge.net/

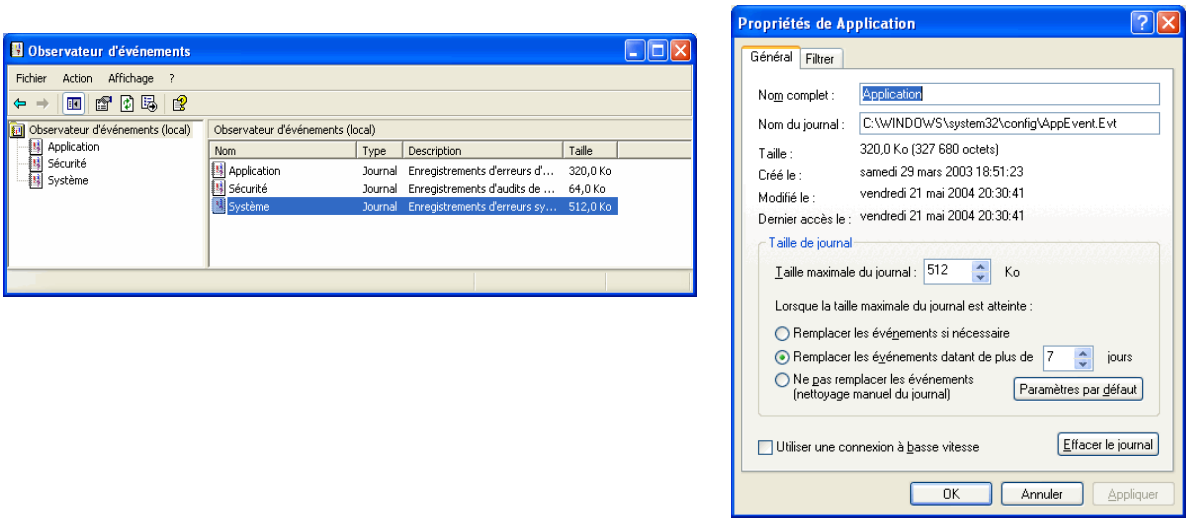



*Illustration 16: Configuration du fichier d'évènements*

données personnelles après leur péremption (surtout si elles ne sont pas utilisées)[22]. Une bonne maîtrise du cycle de vie des journaux est donc nécessaire avant de s'engager dans leur activation.

Par ailleurs, la collecte et même la centralisation des journaux ne sont qu'une première étape, qui fournit avant tout un garde-fou en cas d'enquête approfondie. C'est l'exploitation de ces traces qui permet de progresser dans la gestion de la sécurité.

# 4 Analyse du poste de travail et antivirus

Les systèmes antivirus font partie des systèmes les plus répandus dans le domaine de la sécurité informatique. De manière générale, il s'agit de systèmes de détection de codes malveillants basés sur la détection de signatures. Ces signatures de détection sont identifiées par les éditeurs d'antivirus qui les mettent à disposition de leurs clients, généralement via des systèmes automatiques de téléchargement. On trouve des logiciels antivirus utilisant ces signatures à plusieurs endroits dans le système informatique, notamment sur les postes de travail et sur les principales passerelles applicatives (messagerie, relais HTTP).

## *4.1 Poste de travail*

Initialement, les antivirus ont été déployés sur les postes de travail. Ils permettent ainsi de réaliser une analyse complète des éléments du système de fichiers afin de détecter d'éventuels virus présents dans les fichiers et de les éliminer soit en corrigeant, soit en détruisant le fichier concerné. Associés à des composants du système d'exploitation, ils permettent d'effectuer une analyse plus dynamique, souvent dite « en temps réel », des différents fichiers ouverts par les applications, de manière à détecter les virus au plus tôt, notamment dans le cas de leur diffusion au sein de documents. Les analyses complètes programmées (nocturnes) sont généralement utilisées sur les serveurs (pour lesquels une analyse dynamique poserait des problèmes de performance en raison du grand nombre de fichiers accédés) tandis qu'une analyse temps réel est généralement préférable sur les postes de travail.

22 Toutefois, contrairement à certaines idées reçues, la CNIL est loin de s'opposer systématiquement à la collecte de traces. *Au contraire*, elle nous semble même le recommander dans le cas où cette collecte présente une finalité d'amélioration de la sécurité des systèmes (notamment ceux traitant des données personnelles). Par contre, bien évidemment, une collecte sans objectif précis et un archivage indéfini de traces contenant des données à caractère personnel sont totalement contraires à l'esprit de la loi française. Et sur ce point nous y souscrivons entièrement.


*Illustration 17: Interface principale de l'antivirus*

Sur le poste de travail de l'utilisateur final, l'interface de l'antivirus permet d'afficher un certain nombre d'informations générales sur les détections éventuelles effectuées par l'antivirus (voir figure 18) ainsi que de lancer manuellement des analyses complètes du système de fichiers à la demande. En général, l'utilisateur a peu de contrôle sur les paramètres de fonctionnement de l'antivirus qui restent réservés aux administrateurs et sont diffusés par le serveur de l'entreprise. Notamment, il est important qu'un utilisateur peu averti ne puisse pas désactiver facilement l'antivirus[23]. Dans le logiciel présenté en exemple, on note aussi la présence d'un système de quarantaine permettant d'isoler les fichiers contaminés par des virus pour des analyses ultérieures plus détaillées.

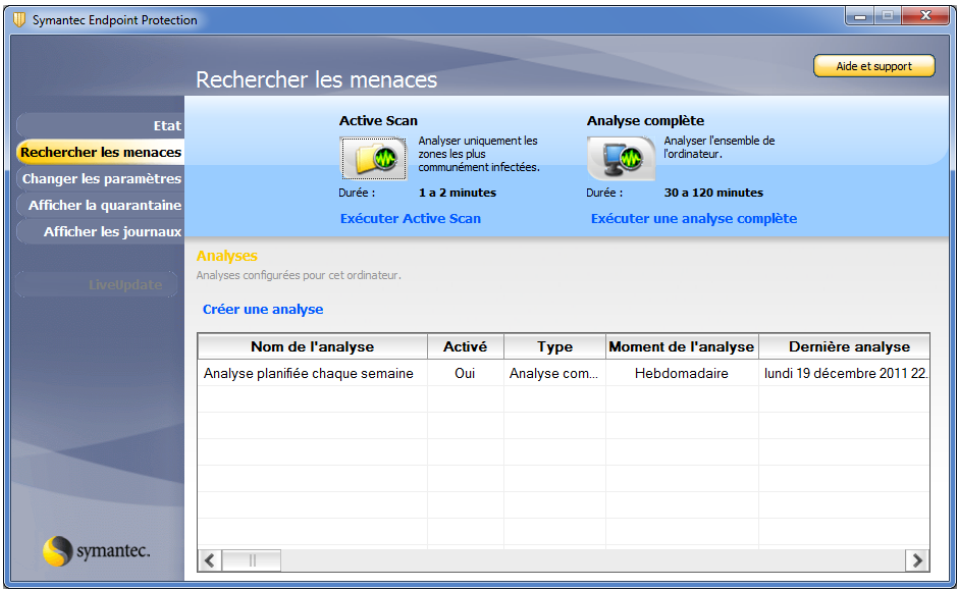



*Illustration 18: Fonctions de l'antivirus accessibles à l'utilisateur final*

23 Et surtout oublier de le réactiver.

La figure 17 présente les principaux paramètres de l'antivirus du poste de travail. On y trouve d'abord identifié le serveur parent du poste de travail du point de vue de l'antivirus : c'est ce serveur qui distribue à ce poste les mises à jour de signatures permettant de détecter de nouveaux virus, ainsi que d'éventuelles évolutions du moteur d'analyse lui-même. Sont ensuite listés les différentes versions du programme, du moteur d'analyse, et surtout du fichier de signatures (nommé fichier de définitions virales dans l'exemple présenté). Afin de pouvoir détecter efficacement tous les virus connus et surtout les plus récents, ce fichier de signatures doit bien évidemment être aussi récent et aussi complet que possible.

*Remarque* : En 2003, le nombre de signatures prises en compte était déjà d'environ 65 500. Au 21 décembre 2011, le nombre de signatures de détection revendiqué par le même éditeur était passé à 15 303 281 avec une croissance exponentielle entre les 2 périodes.

Enfin, la figure 19 présente un exemple d'alerte émis en direction de l'utilisateur en cas de détection de virus. Il s'agit dans ce cas du virus de test présenté au §4.3. Parmi les informations affichées, on note l'action effectuée par l'antivirus : il s'agit en général soit d'une opération de correction du fichier (nettoyage), soit d'une suppression du fichier concerné, soit éventuellement d'une mise en quarantaine.

Il est possible de désactiver ces messages d'alerte (surtout s'ils sont aussi collectés par d'autres moyens). D'après notre expérience, le fait de les désactiver ou non est une question assez difficile à trancher.

En général, en cas d'infection virale, la plupart des utilisateurs même avertis, ont une vision très peu fiable de la réalité de l'infection de tel ou tel poste de travail. Ainsi, la présence d'une alerte de l'antivirus est en fait certainement le signe que le poste de travail n'a pas été infecté ; tandis que son voisin qui, lui, n'a vu aucune alerte, peut très bien avoir été corrompu (notamment s'il disposait d'un fichier de définitions virales un peu plus ancien). Dans un cas comme dans l'autre, caractériser précisément la présence ou l'absence d'un virus particulier demande une connaissance précise de ses caractéristiques techniques (les paramètres systèmes qu'il altère par exemple). L'alternative (notamment pour des informaticiens peu familier des problèmes de sécurité) est d'effectuer une analyse complète du système de fichiers après avoir vérifié la mise à jour des signatures utilisées par l'antivirus ; mais cette alternative n'est pas toujours aussi fiable qu'une étude manuelle, notamment du fait que certains virus récents commencent à essayer d'altérer le fonctionnement des antivirus courants.

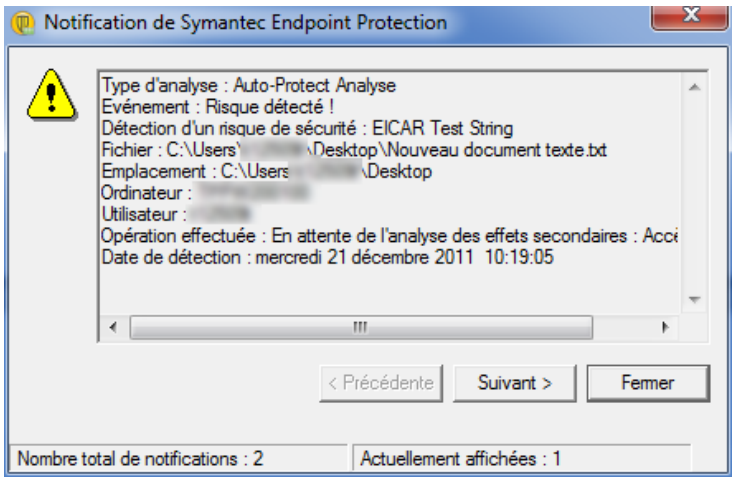



*Illustration 19: Exemple de message d'alerte*

Dans cette situation, le déclenchement des alertes conduit généralement les utilisateurs à compliquer la gestion d'une infection virale en occupant l'attention des administrateurs. Ceci tendrait à favoriser leur désactivation. Toutefois, en cas de détection d'un virus, il peut aussi sembler normal d'avertir l'utilisateur direct du poste de travail, qui est le premier concerné par les données mises en danger et qui peut ainsi comprendre le travail que réalise en permanence l'antivirus sur son poste.

## *4.2 Console de gestion*

Du point de vue de l'administrateur, les différents moyens de gestion de l'antivirus sont généralement centralisés autour d'une console de gestion (souvent associée au serveur de distribution du fichier des signatures). La figure 20 montre un exemple de la console de gestion du produit présenté précédemment (*Symantec Antivirus*). On y voit les différents

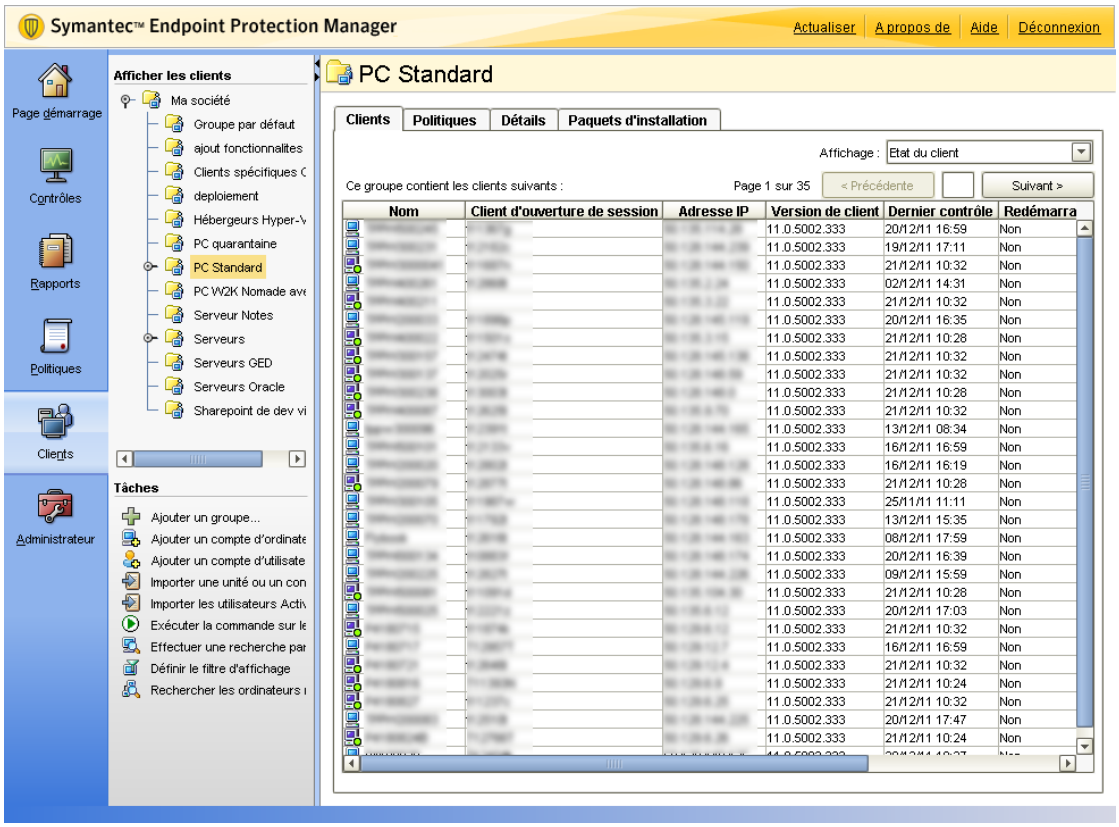



*Illustration 20: Console centralisée de gestion de l'antivirus*

serveurs utilisés pour la distribution du fichier des signatures, ainsi que les événements correspondant aux connexions des postes de travail.

La console permet également de définir différentes politiques de gestion pour les clients antivirus afin de s'adapter à leurs contraintes de fonctionnement. La figure 21 présente un certain nombre de politiques (standards ou spécifiques).

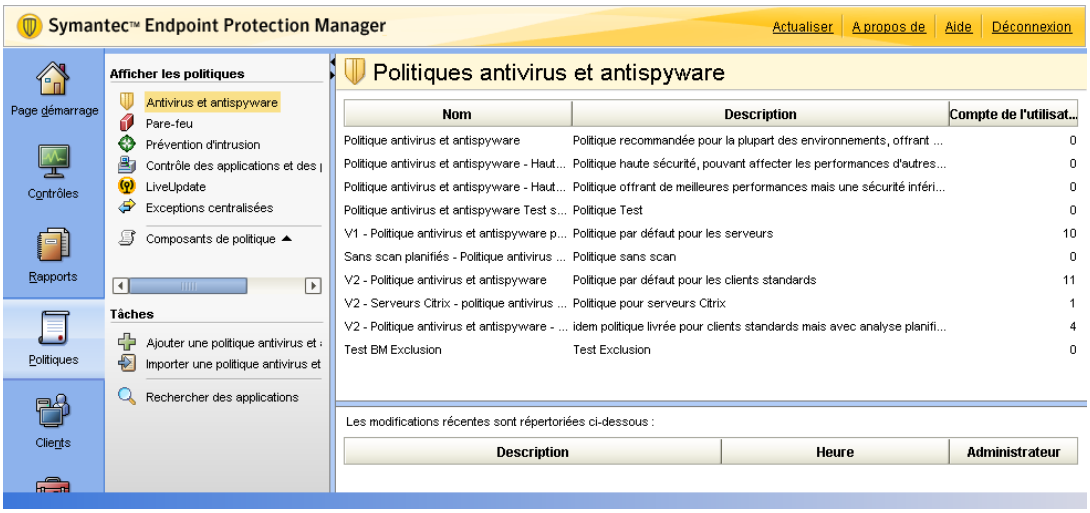



*Illustration 21: Console générale de gestion*

La console de gestion centralise également l'ensemble des alertes de détection de virus déclenchées dans l'entreprise. La figure 22 présente l'affichage synthétique offert par l'interface de gestion à la connexion de l'administrateur pour la supervision du fonctionnement, en mettant en évidence les événements de sécurité (partie gauche).

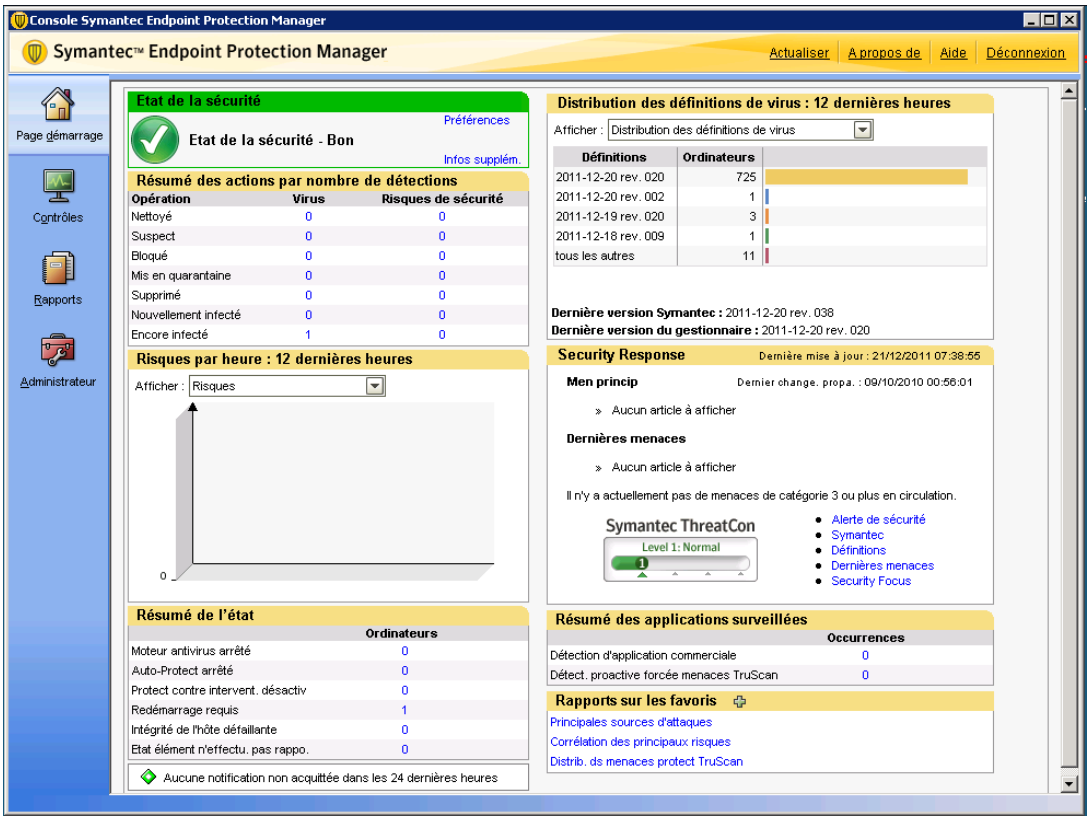
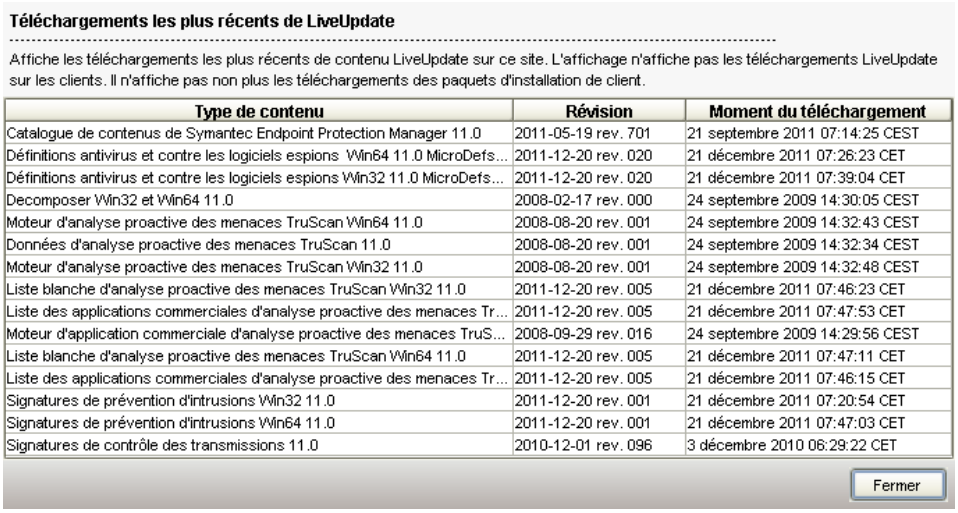



*Illustration 22: Console générale de gestion*

Cet écran initial indique notamment (en haut à droite) le niveau de mise à jour de l'ensemble des antivirus du parc géré. En règle générale, ce niveau de mise à jour doit être le plus large possible pour garantir une protection contre les menaces les plus récentes. Mais il faut aussi garder à l'esprit le fait qu'un poste de travail peut prendre un certain retard par rapport à la dernière version disponible (par exemple si son utilisateur normal est absent pendant un certain temps et que le poste est arrêté). Par contre, il peut être utile d'identifier les postes de travail dont les mises à jour ne se font pas de manière normale, c'est à dire ceux qui sont absents de cet historique ou ceux dont les mises à jour échouent. (Suivant les produits, ces informations-là sont parfois assez difficiles à consolider.)

Actuellement, un antivirus récupère désormais un grand nombre d'informations de mise à jour ayant trait non seulement aux signatures de détection, mais aussi aux mises à jour de son propre moteur logiciel ou à des règles plus sophistiquées de classification de comportement des programmes ou de reconnaissance de leur fonctionnement. La figure 23 présente un historique de ce type de téléchargement via le service de l'éditeur concerné ici (baptisé LiveUpdate).



**Téléchargements les plus récents de LiveUpdate**

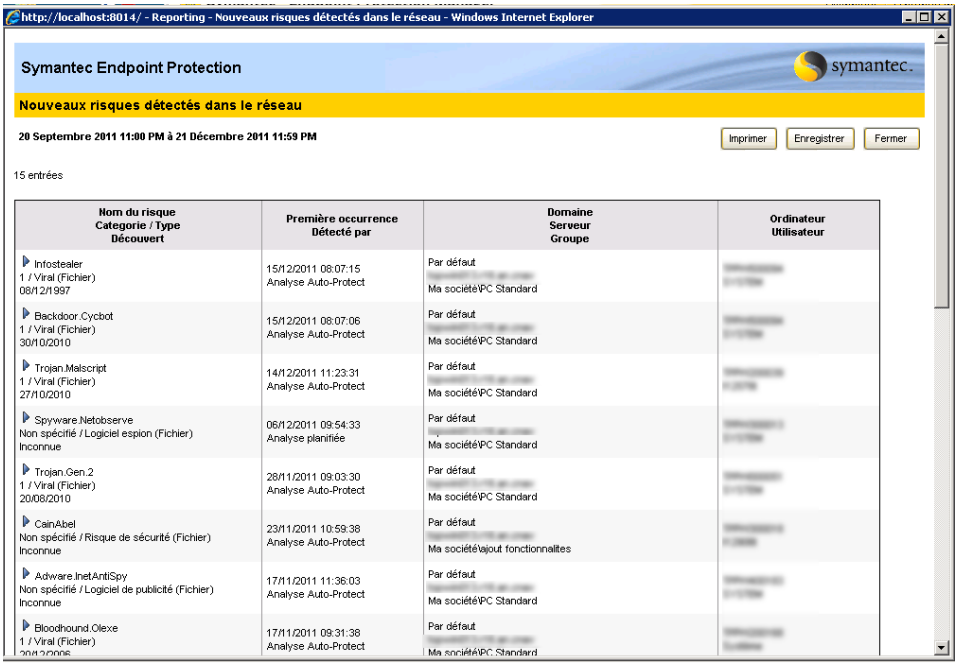
Affiche les téléchargements les plus récents de contenu LiveUpdate sur ce site. L'affichage n'affiche pas les téléchargements LiveUpdate sur les clients. Il n'affiche pas non plus les téléchargements des paquets d'installation de client.

| Type de contenu | Révision | Moment du téléchargement |
|---|---|---|
| Catalogue de contenus de Symantec Endpoint Protection Manager 11.0 | 2011-05-19 rev. 701 | 21 septembre 2011 07:14:25 CEST |
| Définitions antivirus et contre les logiciels espions Win64 11.0 MicroDefs... | 2011-12-20 rev. 020 | 21 décembre 2011 07:26:23 CET |
| Définitions antivirus et contre les logiciels espions Win32 11.0 MicroDefs... | 2011-12-20 rev. 020 | 21 décembre 2011 07:39:04 CET |
| Decomposer Win32 et Win64 11.0 | 2008-02-17 rev. 000 | 24 septembre 2009 14:30:05 CEST |
| Moteur d'analyse proactive des menaces TruScan Win64 11.0 | 2008-08-20 rev. 001 | 24 septembre 2009 14:32:43 CEST |
| Données d'analyse proactive des menaces TruScan 11.0 | 2008-08-20 rev. 001 | 24 septembre 2009 14:32:34 CEST |
| Moteur d'analyse proactive des menaces TruScan Win32 11.0 | 2008-08-20 rev. 001 | 24 septembre 2009 14:32:48 CEST |
| Liste blanche d'analyse proactive des menaces TruScan Win32 11.0 | 2011-12-20 rev. 005 | 21 décembre 2011 07:46:23 CET |
| Liste des applications commerciales d'analyse proactive des menaces Tr... | 2011-12-20 rev. 005 | 21 décembre 2011 07:47:53 CET |
| Moteur d'application commerciale d'analyse proactive des menaces TruS... | 2008-09-29 rev. 016 | 24 septembre 2009 14:29:56 CEST |
| Liste blanche d'analyse proactive des menaces TruScan Win64 11.0 | 2011-12-20 rev. 005 | 21 décembre 2011 07:47:11 CET |
| Liste des applications commerciales d'analyse proactive des menaces Tr... | 2011-12-20 rev. 005 | 21 décembre 2011 07:46:15 CET |
| Signatures de prévention d'intrusions Win32 11.0 | 2011-12-20 rev. 001 | 21 décembre 2011 07:20:54 CET |
| Signatures de prévention d'intrusions Win64 11.0 | 2011-12-20 rev. 001 | 21 décembre 2011 07:47:03 CET |
| Signatures de contrôle des transmissions 11.0 | 2010-12-01 rev. 096 | 3 décembre 2010 06:29:22 CET |

Fermer

*Illustration 23: Téléchargements*

Enfin, la figure 24 présente un des rapports de gestion de l'antivirus, disponible au niveau de la console de gestion. Ce rapport est focalisé sur les événements de détection de risques identifiés (et neutralisés) par l'antivirus.



http://localhost:8014/ - Reporting - Nouveaux risques détectés dans le réseau - Windows Internet Explorer

Symantec Endpoint Protection

symantec.

**Nouveaux risques détectés dans le réseau**

**20 Septembre 2011 11:00 PM à 21 Décembre 2011 11:59 PM**

Imprimer | Enregistrer | Fermer

15 entrées

| Nom du risque<br>Categorie / Type<br>Découvert | Première occurrence<br>Détecté par | Domaine<br>Serveur<br>Groupe | Ordinateur<br>Utilisateur |
|---|---|---|---|
| Infostealer<br>1 / Viral (Fichier)<br>08/12/1997 | 15/12/2011 08:07:15<br>Analyse Auto-Protect | Par défaut<br>Ma société\PC Standard | |
| Backdoor.Cycbot<br>1 / Viral (Fichier)<br>30/10/2010 | 15/12/2011 08:07:06<br>Analyse Auto-Protect | Par défaut<br>Ma société\PC Standard | |
| Trojan.Malscript<br>1 / Viral (Fichier)<br>27/10/2010 | 14/12/2011 11:23:31<br>Analyse Auto-Protect | Par défaut<br>Ma société\PC Standard | |
| Spyware.Netobserve<br>Non spécifié / Logiciel espion (Fichier)<br>Inconnue | 06/12/2011 09:54:33<br>Analyse planifiée | Par défaut<br>Ma société\PC Standard | |
| Trojan.Gen.2<br>1 / Viral (Fichier)<br>20/08/2010 | 28/11/2011 09:03:30<br>Analyse Auto-Protect | Par défaut<br>Ma société\PC Standard | |
| CainAbel<br>Non spécifié / Risque de sécurité (Fichier)<br>Inconnue | 23/11/2011 10:59:38<br>Analyse Auto-Protect | Par défaut<br>Ma société\ajout fonctionnalites | |
| Adware.InetAntiSpy<br>Non spécifié / Logiciel de publicité (Fichier)<br>Inconnue | 17/11/2011 11:36:03<br>Analyse Auto-Protect | Par défaut<br>Ma société\PC Standard | |
| Bloodhound.Olexe<br>1 / Viral (Fichier) | 17/11/2011 09:31:38<br>Analyse Auto-Protect | Par défaut<br>Ma société\PC Standard | |

*Illustration 24: Rapport de suivi des risques*

## *4.3 Tester un antivirus*

Le test du bon fonctionnement d'un logiciel antivirus est un sujet plus délicat qu'il n'y paraît.

D'abord c'est un point important : un logiciel antivirus n'a d'intérêt que s'il est réellement en mesure de stopper des virus. Tout dysfonctionnement devrait être détecté.

Ensuite, il est absolument hors de question de tester le bon fonctionnement d'un antivirus en introduisant des virus réels dans un système informatique ! D'abord, il n'est pas forcément facile d'obtenir un virus réel, la préoccupation courante en sécurité informatique étant plutôt de les éradiquer que de les conserver. Par ailleurs, même si ce type de test était effectué sur un système isolé en salle blanche, totalement déconnecté de tout autre système, le risque resterait important de déclencher une propagation réelle (fut-ce par une erreur de manipulation). Par ailleurs, dans ce cas, l'intérêt du test

lui-même serait assez limité car un système isolé ne permet pas de tester l'environnement d'exploitation réel (et notamment par exemple la remontée des alertes vers une console de gestion).

Pour pallier à ces difficultés de test du déploiement, la plupart des constructeurs d'antivirus ont implémenté une signature de détection d'un virus factice, nommé EICAR. La définition de référence de ce virus est accessible à l'URL suivante : http://www.eicar.org/anti_virus_test_file.htm . Ce virus est sans danger car il correspond tout simplement à un fichier texte contenant les 68 caractères suivants et ayant une longueur d'exactement 68 octets :

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

Ce fichier est par ailleurs un programme DOS exécutable valide dont le seul effet inoffensif est d'imprimer le message suivant : « EICAR-STANDARD-ANTIVIRUS-TEST-FILE! ».

Créer ou copier un tel fichier dans un espace de test sur le disque dur d'un système est donc un excellent moyen de tester le bon fonctionnement du logiciel antivirus. Par contre, il faut noter qu'en règle générale ce fichier sera immédiatement traité par le logiciel antivirus, c'est à dire soit effacé, soit mis en quarantaine.

### *4.4 Antivirus de flux : messagerie, flux HTTP (entrant)*

Les virus utilisant d'autres moyens de propagation, notamment les messages électroniques et les flux HTTP (via les possibilités de téléchargement ou d'exécution de code offertes par les clients de messagerie et les navigateurs) de nouveaux besoins d'analyse antivirale sont apparus. Au niveau des passerelles de messagerie ou des relais HTTP, ces antivirus fonctionnent de manière assez similaire à ceux présents sur les postes de travail en s'appuyant sur des définitions de signatures. Le fonctionnement de l'antivirus est surtout différent en ce sens qu'il traite un flux de communication qu'il doit essayer de ne pas perturber, notamment du point de vue de ses performances.

En ce qui concerne les flux de messagerie, l'antivirus doit analyser les pièces jointes aux messages électroniques[24]. Celles-ci pouvant être incluses dans des fichiers compressés, le travail d'analyse est conséquent et le dimensionnement des plate-formes matérielles réalisant ces analyses est assez délicat.

En ce qui concerne les flux HTTP, la principale difficulté consiste à réaliser une analyse à la volée sans bloquer le flux de communication. Plusieurs stratégies sont envisageables et certaines peuvent modifier assez sensiblement le ressenti des utilisateurs finaux (notamment dans le cas où les fichiers sont stockés transitoirement par l'antivirus).

Enfin, tenter de télécharger depuis l'URL de référence un des fichiers disponibles contenant le virus factice EICAR est un bon moyen de tester la présence et le bon fonctionnement d'un antivirus HTTP s'il existe (voir §4.3). Le test d'un antivirus de messagerie peut être effectué en envoyant le fichier dans un message électronique[25].

## 5 Outils complémentaires

### *5.1 Wireshark*

Wireshark[26] (anciennement Ethereal[27]) est un outil d'analyse réseau qui permet à la fois de réaliser une capture du trafic visible depuis une interface réseau de la machine sur laquelle il s'exécute, et de visualiser plus précisément le contenu des paquets capturés en fonction du protocole réseau auquel ils correspondent. Wireshark est désormais capable d'analyser la plupart des protocoles réseaux existants. Il est de ce fait devenu un outil incontournable dans la boite à outil de l'administrateur sécurité, notamment parce qu'il permet d'étudier en détail le contenu d'une capture réseau (obtenue éventuellement par des moyens différents) et parce qu'il permet d'identifier précisément les flux réseaux associés à une application (en observant le trafic réseau associé à son exécution) et donc de mieux comprendre son fonctionnement. Ce dernier point peut être particulièrement utile pour comprendre un trafic réseau particulier identifié par une sonde NIDS et notamment valider son innocuité.

Wireshark est un outil appuyé sur une interface graphique, mais on dispose aussi d'une version fonctionnant purement dans un terminal texte : tshark. Cette version permet aussi de réaliser des captures réseau directes en ligne de commande et peut se substituer avec bénéfice à l'outil classiquement utilisé sous Unix pour ce type d'opération : tcpdump.

---

24 C'est une vérification parfois redondante avec celle effectuée par l'antivirus du poste de travail au moment de la réception du message, mais qui semble rentable en général.

25 A condition que l'antivirus du poste de travail vous permette de le créer ! Il faut généralement le désactiver pour y arriver.

26 http://www.wireshark.org/

27 L'outil a changé de nom en juin 2006 ; créant probablement une fracture irrémédiable entre les différentes générations d'administrateurs réseau.

## *5.2 Tripwire, Samhain, AIDE : contrôle d'intégrité des fichiers*

A la frontière entre la détection d'intrusion et les mécanismes de protection de la sécurité informatique se trouve une catégorie d'utilitaires tout à fait intéressante qui, avec le temps, est de plus en plus associée à la famille des outils de détection d'intrusion. Ils s'agit des utilitaires qui permettent de construire une base d'empreintes de tout un ensemble de fichiers afin de vérifier ultérieurement (ou périodiquement) que ces fichiers n'ont pas subi d'altérations. Ces utilitaires sont largement associés à l'utilisation assez directe des algorithmes cryptographiques d'empreinte (ou *secure hash*)[28]. Le premier représentant de cette famille et le plus connu est l'utilitaire Tripwire, développé initialement à l'université de Purdue entre 1992 et 1994 par Gene Kim et Gene Spafford.[29] Actuellement, le logiciel Tripwire est le logiciel commercialisé sous cette marque par la société Tripwire Inc., fondée par un des deux auteurs de la version d'origine non-commerciale.

Parmi les solutions les plus accessibles disponibles actuellement pour réaliser ce type de contrôle on trouve notamment les logiciels Samhain[30] et AIDE[31]. Par rapport à une utilisation directe des programmes de calcul de fonctions de hachage (comme `md5sum` ou `sha1sum`), ces outils offrent les fonctions de gestion et d'actualisation de la base de données des empreintes qui sont nécessaires pour étendre les possibilités de contrôle offertes par les fonctions élémentaires à l'ensemble d'un système de fichiers réels avec toutes ces caractéristiques (binaires exécutables, fichiers de configuration, fichiers sensibles, fichiers traces en mode ajout seulement, fichiers spéciaux, zones temporaires, etc.).

Ces outils visant la vérification d'intégrité d'un système de fichiers ne partagent pas le caractère d'utilisation permanent classiquement associé aux NIDS ou HIDS, ils correspondent à un mode d'utilisation périodique. Mais cela n'enlève rien à leur intérêt pratique en matière de protection d'un système informatique. (On notera d'ailleurs qu'ils étaient déjà pris en compte par les auteurs de la classification présentée figure 1.)

28 On peut même considérer que certaines fonctions cryptographiques d'empreintes ont été conçus pour ces outils et pas nécessairement l'inverse.

29 Actuellement, le logiciel Tripwire est le logiciel commercialisé sous cette marque par la société Tripwire Inc., fondée par un des deux auteurs de la version d'origine non-commerciale, qui a acquis la marque auprès de l'université de Purdue en 2000.

30 http://www.la-samhna.de/samhain/index.html

31 http://aide.sourceforge.net/ - d'ailleurs baptisé « *Advanced Intrusion Detection Environment* ».