Escola Universitària
Politècnica de Mataró

Enginyeria Tècnica de Telecomunicació:
Especialitat Telemàtica

IDS
(Intrussion Detection Systems)

Oriol Rius Canals

TARDOR 2001/02

ÍNDEX

ÍNDEX DE FIGURES

1. Introducció

IDS (Intrusion Detection Systems) són components de hardware i/o software que automatitzen el procés de monitoritzar els events que es donen en un sistema o una xarxa, analitzant-los en busca de problemes de seguretat. Degut a l'augment espectacular dels atacs a xarxes i sistemes els últims anys els IDS han començat a ser un element de seguretat necessari a les organitzacions. Aquest projecte preten dónar una visió global sobre aquests elements de seguretat, veient quines són les seves avantatges, desavantatges i fins a quin punt poden ser útils. A més, també es podrà observar com es pot integrar un IDS dins l'entorn de seguretat de l'organització.

2. Objectius

Aquest projecte no preten ser una guia exahustiva en la implementació i coneixement dels IDS, ja que l'espai del que es disposa no és suficient. No obstant això s'intenta dónar una visió el més concreta possible, per tal de poder coneixer amb precisió quin és el funcionament, avantatges i inconvenients dels IDS.

També es preten seguir la implementació d'un IDS. Intentant acostar des d'un cas pràctic tot el que s'explica en la restà del projecte de forma teòrica. Malgrat l'explicació s'intenta fer el més detalladament possible. S'obvien tots els temes referents a configuració, compilació, instal·lació de sistemes operatius i eines complementaries, ja que això dónaria lloc a un altre projecte. Per tant, aquest segon objectiu seria la part pràctica del projecte.

Així doncs, el gran objectiu que persegueix aquest projecte és entrar una mica més en aquestà caixa negre en la que s'acaben convertint els elements de seguretat, concretament en els IDS. No es pot oblidar però, que com tots els elements de seguretat, no és gens trivial ni inmediat entendre el perquè de certes decisions i només la constant formació, l'experiènca i sovint també la imaginació seran les armes que ajudaran a entendre aquests foscos processos.

3. Visió general dels IDS

3.1 Què és un IDS?

La detecció d'intrusos és el procés de monitoritzar events en un ordinador o en una xarxa i analitzar-los buscant fets maliciosos, entenent aquests fets com el compromís de la confidencialitat, integritat, disponibilitat o saltar-se els mecanismes de seguretat d'un ordinador o una xarxa. Les intrusions provenen d'atacants d'internet i usuaris autoritzats que pretenen escalar privilegis de forma no autoritzada. També poden ser fruit del mal ús dels serveis que fan els usuaris autoritzats. Un IDS és un conjunt de software i/o hardware encarregat d'automatizar aquest procés de monitorització i anàlisis.

3.2 Entorn dels IDS?

Aquests sistemes no tenen massa sentit en xarxes molt petites (grups de treball), per tant, el seu entorn natural són les xarxes on almenys hi hagi una *DMZ* (zona desmilitaritzada). Per si mateixos no són gaire útils i actuen com a complement a altres elements de seguretat, com poden ser els firewalls. Els podem trobar doncs, dins les subxarxes on hi ha accessos incontrolats o difícils de controlar. Malgrat puguin actuar com un element preventiu davant de les intrusions el seu gran avantatge el notem sovint després d'un atac, perquè obtenim un complement molt valuós en la informació que ens proporcionen.

Pel que fa als recursos personals que requereix un IDS aquest també marca les pautes d'un entorn molt selectiu. O sigui, que la persona/es encarregada/es de mantenir i fer el seguiment de les sortides del IDS ha de tenir una formació i una experiència gens trivial en conceptes de seguretat. A més, sovint, a aquestà experiència se li ha de sumar una

gran dosi d'imaginació per tal de poder respondre, o fins hi tot avançar-se, a les accions dels intrusos.

3.3 Per què són necessaris els IDS?

Els IDS permeten a les organitzacions protegir els seus sistemes dels nous perills als que s'exposen. Degut a l'increment de la connectivitat dels seus sistemes i de la depéndència d'aquestà connectivitat que tenen pel seu funcionament. Dónada la naturalesa de la seguretat de les xarxes modernes la pregunta no està en decidir si usar un IDS sinó on col·locar i quines propietats cal habilitar en cada IDS.

Algunes raons per les que s'han d'usar IDS són:

· Prevenir atacs i altres intrusions que no poden prevenir les altres mesures de seguretat.

· Detectar i tractar les accions que acostumen a precedir un atac.

· Documentar l'evolució de les intrusions que hi ha dins l'organització.

· En empreses grans i complexes, pot actuar com a control de la qualitat de la seguretat de la mateixa en el disseny de la xarxa.

· Recol·lectar informació sobre les intrusions que tenen lloc a l'organització, permentent diagnosticar, recuperar i corregir les causes de l'error.

3.4 IDS vs Firewalls

Per què necessitem un IDS a la nostra xarxa si ja hi ha un Firewall que ens protegeix? Doncs, al igual que els firewalls els IDS són un element de seguretat de la xarxa, però les seves funcions són complementàries i no pas excloents. Per la seva naturalesa en el disseny, els firewalls poden fer coses que els IDS no poden i al revés.

En principi un firewall és un element intrusiu pel tràfic, això vol dir que els paquets han de travessar-lo per arribar al seu destí. En canvi els IDS són un element passiu en aquest aspecte, perquè que malgrat poden veure aquests mateixos paquets només els analitzen sense intervenir-los. Ràpidament podem adónar-nos que un firewall pot arribar a ser un problema per la nostra xarxa si no és capaç d'absorbir tots els paquets que han de travessar-lo. En canvi un IDS només deixarà d'inspeccionar els paquets que no pot analitzar però no repercutirà en el tràfic de la nostre xarxa.

També pel seu disseny els firewalls són un element de seguretat perimetral, per tant, es dediquen a protegir la xarxa de l'exterior. En canvi els IDS són elements de seguretat tan perimetral com interna, així doncs, també ens permenten la defensa dels usuaris i no només dels accessos externs. No s'ha d'oblidar que s'està fent referència a problemes de disseny; aquests problemes amb una bona implementació es poden usar com una qualitat o una propietat positiva.

Tal com s'explicava anteiorment els firewalls tenen un comportament intrusiu en el tràfic, això els permet dónar funcions de NAT (Network Address Translation), per tant, poden alterar el tràfic que els entra o surt. Això mai ho podrà fer un IDS ja que com a molt amb el seu anàlisi podrà alterar la configuració del firewall perquè aquest alteri el tràfic, però ell mai ho farà de forma directa.

Per definició el que s'expressa en el punt anterior és cert. Però les noves versions de l'Snort (IDS amb llicència lliure) permenten forçar el tancament de connexions TCP i fins hi tot informar a la font del paquet de la il·legitimitat del seu paquet. Per exemple, si un client de la xarxa decideix entrar a pàgines amb contigut eròtic, aquest IDS permet tallar la connexió del client amb el servidor i enviar-li al client una pàgina on s'informa que el seu intent d'accés ha estat registrat pels sistemes de seguretat de la xarxa.

3.5 Pros i contres dels IDS

Pros:

· Ajuden a automatitzar tasques de manteniment a nivell de seguretat als administradors, per tant, simplifiquen molt la seva feina i la fan més eficient.

· Fan possibles tasques que de forma manual seria quasi impossible portar a terme.

· La disponibilitat d'aquests sistemes és molt gran.

· La caiguda del sistema no suposa un problema pel rendiment dels serveis dels sistemes i/o la xarxa.

· La informació que proporcionen els IDS és molt útil. Tan per mantenir i millorar la seguretat de la xarxa i els sistemes, com per coneixer a fons el funcionament dels serveis.

· Ajuden a preveure atacs o problemes de seguretat abans que aquests es donin.

Contres:

· L'automatització excessiva de certs processos pot fer obviar informació necessària.

· El seu comportament sovint converteix als administradors en simples elements passius a l'hora d'analitzar la informació que generen els IDS.

· S'ha d'estàr atent de forma periódica en el manteniment dels sensors dels IDS, ja que aquests tenen tendència a ser inestàbles en xarxes amb molt de tràfic.

· Una caiguda no controlada pot fer inestàble a un sistema (*HIDS: Host IDS*).

· Cal anar en compte a l'hora d'activar les propietats dels sensors sinó la quantiat d'informació que generin pot desbordar.

· Necessiten un manteniment continuat i periòdic, tan a nivell del IDS com de la restà d'elements de seguretat de la xarxa.

· Fa falta personal molt qualificat per poder interpretar de forma correcte les seves sortides.

· No tota la informació que proporcionen és certa, falços positius.

4. Tipus d'IDS

Pel seu comportament, col·locació, forma de funcionament, etc, podem fer diferents classificacions dels IDS.

4.1 NIDS (Network IDS)

4.1.1 Descripció

Sistema format per un hardware i un software que es col·loca en una xarxa sent transparent per aquestà, no tenen ni IP. Mira el tràfic que circula per la xarxa i va comparant els paquets que circulen per la xarxa amb una base de dades de patrons d'intrusions.

Tot el tràfic sospitós el va guardant en un registre (log), aquest registre poden ser des de fitxers de text plans (syslog), fins a sistemes gestors de bases de dades (xSQL). Últimament hi ha molta tendència a guardar-ho en fitxers XML, per tal de compartir aquestà informació de forma estàndard entre IDS de diferents fabricants. Gràcies a això, els CMS (Console Management System) poden concentrar més informació de diferents productes.

La base de dades d'intrusions s'ha d'actualitzar de forma constant, perquè el terreny de la seguretat és molt dinàmic i està en constant evolució. A més la informació que ens proporcionen els registres de l'IDS permeten ajustar de forma més eficaç els paràmetres dels sensors que hi ha activats. Per exemple, el codi d'un enllaç d'una pàgina Web de la nostre empresa pot fer saltar una alarma d'atac d'*Unicode* (estàndard de format de cadenes) en cas de que aquest contingui excessius punts i barres. Aquest atac potser no

té cap sentit detectar-lo perquè el servidor en qüestió és un Unix que no és sensible als atacs *Unicode*.

4.1.2 Esquema d'un NIDS

Degut al seu disseny els NIDS es poden col·locar a diferents llocs d'una xarxa. En un capítol posterior es raonaran diferents possibilitats. De moment, en aquest punt només es descriu el seu funcionament intern:

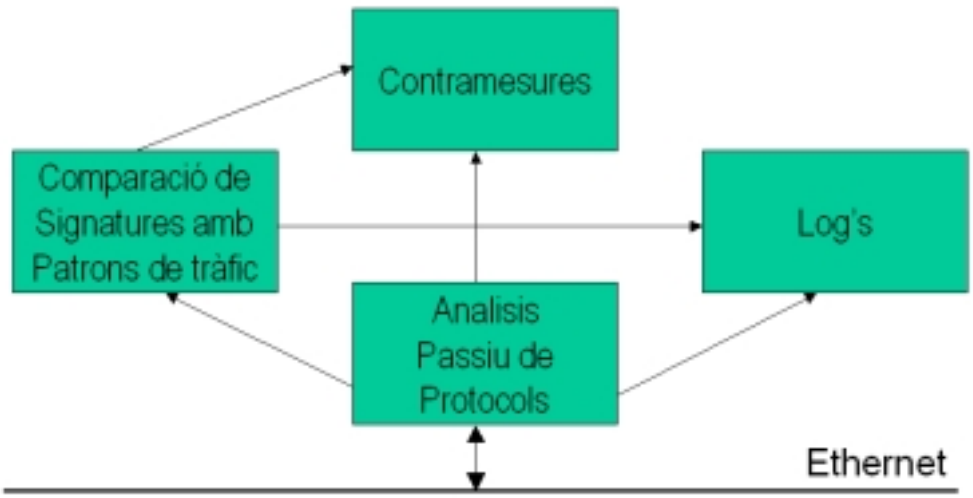

Ilustración 1 - Esquema d'un NIDS

El NIDS posa la targeta de xarxa en mode promiscu per poder veure tot el tràfic que passa per aquell punt. De forma passiva va analitzant tot el tràfic que passa i va comparant-lo amb la base de dades de tràfic maliciós que té. En cas de que en algun moment consideri

que cert tràfic és maliciós el guardarà al registre i en cas de que s'hagi programat alguna contramesura per aquell esdeveniment aquestà es llençarà.

4.1.3 Pros i contres dels NIDS

Pros

· No baixen el rendiment dels sistemes.
· Resistent a les manipulacions.
· Indepéndent de les plataformes i Sistemes Operatius de la xarxa.
· Envia la informació que registre a un sistema de recol·lecció de logs (Bases de dades, arxius plans, XML, etc).
· És capaç de predir un atac abans de que aquest es doni, ja que intervé els paquets abans de que arribin al seu destí.

Contres

· Pot perdre paquets al reensamblar-los.
· No entenen certs protocols obsolets i/o propietaris.
· Pot perdre paquets en xarxes molt carregades.
· No pot capturar dades encriptades (ex.ESP)
· No pot controlar certs comportaments dels usuaris dins dels hosts.
· Requereix hardware addicional. (ex.un host que faci d'IDS).

4.2 HIDS (Host IDS)

4.2.1 Descripció

Monitoritza les activitats d'un sistema (comandes d'un usuari, procés de logon/logout, ús de les dades, etc.) per determinar els possibles intents d'intrusions. Els HIDS acostumen a ser una aplicació més del sistema, normalment treballant com a dimoni o com a servei del sistema. Tot i que també és usual que s'integri fins al kernel del sistema

operatiu per tal de garantir la seguretat en certes crides a sistema crítiques de per si. Per exemple, les crides a sistema que impliquen una escalada de permisos. O bé, en el control de permisos, accessos o d'altres activitats crítiques.

Els HIDS afegeixen funcions que auditen el sistema de forma constant, a canvi de perdre rendiment en el sistema. Tot i que l'alta qualitat de la informació que proporcionen, els HIDS fa que valgui la pena perdre aquest rendiment en els sistemes. La informació que contenen els registres generats pels HIDS és molt densa i acostumen ha haver-hi molts pocs falços positius.

És indepéndent dels protocols de xarxa usats i de si la informació per la xarxa va encriptada, ja que el control es fa a nivell de sistema operatiu o d'aplicació. Per tant, la informació sempre està en text pla. Gràcies a això, també, podem fer seguiments a certs usuaris (UID).

4.2.2 Esquema d'un HIDS

En el següent gràfic podem apreciar fins a quin punt s'arriba a integrar amb el sistema un HIDS, per tal de poder controlar les accions que es realitzen sobre seu:

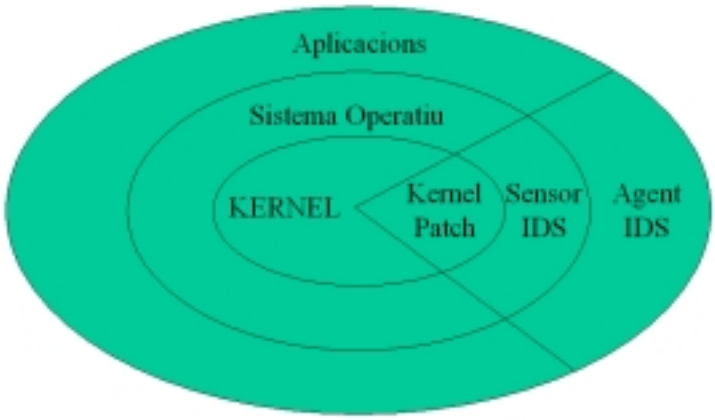

Ilustración 2 - Esquema d'un HIDS

En el gràfic es pot apreciar com des del kernel fins al nivell d'aplicació el HIDS està present per controlar les intereaccions dels usuaris amb el sistema. La part més interna passa a formar part del kernel, la part que intereacciona amb el hardware i el software instal·lat al sistema. En aquestà part bàsicament el que es controlen són les crides al sistema (syscalls) per evitar operacions il·legítimes.

Després a nivell de sistema operatiu es troba en forma de dimoni o de servei. A aquest nivell treballa com un sensor que controla el comportament del sistema a nivell d'aplicació. Sovint aquest dimoni també s'encarrega de vigilar la disponibilitat i la integritat d'altres dimonis del sistema. O fins hi tot, de controlar la integritat de certs fitxers vitals pel funcionament del sistema.

Finalment a nivell d'aplicació hi ha l'agent del HIDS que és amb el que interactua l'administrador del sistema per configurar els paràmetres del HIDS i per comprobar l'estat del mateix. Sovint aquestà part s'executa poc sovint, ja que els resultats del HIDS es consulten en el CMS o algún altre dispositiu que centralitzi els registres.

4.2.3 Pros i Contros dels HIDS

Pros

· La qualitat de la informació que proporcionen és molt alta.
· Permet veure qui accedeix a què.
· La densitat de la informació és molt alta.
· Pot treballar en xarxes encriptades.
· Pot treballar en xarxes montades sobre switch.
· No té depéndència de protocols.
· Permet fer un seguiment d'un UID concret.
· La seva eficàcia és molt gran.

Contres

· Comparteix els recursos del sistema.

· Les captures d'informació estàn molt lligades al tipus de sistema. Això les fa molt difícilment centralitzables.

· Sovint els hosts són els destinataris dels atacs, per tant, si un host cau en mans d'un intrús aquest podrà desactivar l'HIDS.

· Cada sistema té els seus propis HIDS ja que estàn lligats a l'arquitectura de les màquines.

· Hi ha una gran depéndència amb el fabricant del producte a l'hora de comprar un CMS si es preten integrar el sistema de seguretat.

4.3 Híbrids

4.3.1 Descripció

Són sistemes que com el seu nom indica actuen simultàniament de NIDS i de HIDS. Obviament les seves característiques són inferiors a la suma de cada dispositiu per separat. Però hi ha alguns casos, sobretot en xarxes petites en els que interessen sistemes més econòmics i menys distribuits.

Sovint, el que passa també, és que en un servidor es monta un NIDS, perdent algunes de les característiques inherents al seu disseny, per exemple, deixa de ser transparent a la xarxa. Però, en xarxes petites això sovint no és un problema molt gran i es valora més la informació que el NIDS ens pot aportar.

A més és molt fàcil fer que els resultats del NIDS interactuin ràpidament amb els serveis de la màquina evitant i recollint informació molt valuosa sobre els intrusos que

ataquen la nostre màquina. Si a més, en el mateix servidor concentrem el CMS i aquest el fem consultable via Web podem tenir montat de forma ràpida un IDS per una petita xarxa.

Un exemple d'això seria montar un sistema Linux amb un servidor Web Apache+PHP+MySQL i un Snort (NIDS) auditant les connexions que arriben a la màquina. Via web podriem montar un ACID (Analysis Console for Intrusion Database) que ens faria de CMS. Aquest IDS podria ser perfectament vàlid i útil en una xarxa de petites dimensions connectada a Internet.

4.3.2 Pros i Contres dels sistemes Híbrids

Pros

· Economia, ens estàlviem sistemes complementaris.
· Aumenta el nivell de seguretat del Host.
· Centralització del sistema de seguretat.
· Pocs problemes de compatibilitat entre els diferents registres del sistema.

Contres

· Compartim la potència del sistema entre els serveis del sistema i el IDS.
· Si la seguretat del host és compromesa els registres també són compromesos.

4.4 Treball manual

4.4.1 Descripció

Consisteix en aplicar la idea dels cyberpunks, o sigui: 'do yourself' (fes-t'ho tu mateix). Mitjançant la combinació d'eines com el tcpdump (sniffer), filtres BSD, els registres del sistema (syslog) i d'altres eines bàsiques. Amb això i una gran dosi de

paciència i molta dedicació es pot dur a terme totes les funcions que el IDS preten automatitzar. A més, també, s'ha d'estàr molt al dia dels tipus d'atac per no dónar com a legítim certs comportaments qua no ho són.

Bàsicament aquestà solució només és factible en xarxes amb molt pocs servidors, o per organitzacions sense necessitats de seguretat no massa elevades. A més aquest sistema no està lligat a cap arquitectura i és aplicable a qualsevol situació.

4.4.2 Pros i contres

Pros

· Economia

· Pocs recursos

· El nivell de complexitat el decidim nosaltres. Segons fins a quina profunditat volem auditar el sistema.

· No està lligat a cap arquitectura ni tipus de sistema.

· Només lligat a errors humans en la interpretació de la informació recollida.

Contres

· Feina tediosa i sovint impossible de posar al dia.

· Difícil de portar a terme amb eficàcia i eficiència.

· Requereix una formació molt alta i un reciclatge constant, per estàr al dia en les vulnerbilitats que es van troben en els nostres sistemes.

· Difícil d'integrar, ja que la manipulació de les dades és totalment manual.

· Poc recomenable si no parlem de xarxes molt petites i amb dades poc susceptibles.

4.5 Honey Pots

4.5.1 Descripció

Són sistemes o xarxes que pretenen simular els servidors i xarxes més importants de l'organització, per tal de fer creure als intrusos que és on han d'atacar. Si aquest cau en la trampa, es captura tot el rastre que va deixant. D'aquestà forma és més fàcil recol·lectar informació de la forma d'atacar dels intrusos.

Aquests sistemes s'intenten dissenyar de forma que tinguin forats de seguretat per tal de que l'intrús aconsegueixi penetrar i així estigui entretingut. Això ens dóna temps a reaccionar ja que els nostres sistemes de seguretat ens advertiran de la presència de l'intrus. En aquest tipus de sistemes la presència de falços positius és pràcticament nul·la, ja que totes les alarmes que es disparin seran fruit d'un accés no autoritzat.

4.5.1.1 Honey Systems

En un sol ordinador es preten simular tota una xarxa de servidors, normalment es simula una *DMZ*, per fer pensar a l'intrús que està entrant en una zona important de l'empresa. Això és possible gràcies als NOS (Network Operative Systems) que permeten instal·lar més d'un sistema operatiu dins d'una màquina, fent-los funcionar a tots de forma simultania. Fins hi tot ens permeten assignar recursos d'aquestà màquina de forma dedicada a certs sistemes que tinguem corrent dins la mateixa.

Gràcies a aquests sistemes sembla que hi hagi tota una DMZ funcionant, quan en realitat tot està corrent sobre una mateixa màquina, dividida internament de forma virtual. Aquestà idea prové dels grans sistemes d'IBM com les series S390, tot i que sistemes com el Linux també permeten implementar infraestructures semblants.

4.5.1.2 Honey Nets

La idea és la mateixa que la del punt anterior, però en aquest cas es monta tota una xarxa que simula una xarxa real de l'organització. En organitzacions de tipus financer o xarxes molt grans, s'arriben a montar Honey Nets que simulen tota l'empresa, a fi i efecte de fer perdre a l'intrús per tot un seguit de sistemes i xarxes completament intrascendents.

En les Honey Nets fins hi tot s'arriben a col·locar generadors de tràfic, de forma que sembli que la xarxa que es simula està plenament operativa. A més aquest tràfic crea confusió a l'intrus, que perdrà molt temps analitzant i estudiant informació completament inútil. Temps que de ben segur li farà cometre algun error que ens permetrà trobar la informació suficient fins a localitzar-lo.

4.5.2 Quan són necessaris?

Els Honey Pots sovint representen un cost addicional innecessari. Per això, no acostumen a ser un element comú en les xarxes convencionals. No obstant, per certs tipus d'organitzacions aquests elements de seguretat són indispensables. Ja que el gran nombre d'atacs que reben diàriament les fan molt vulnerables. Les organitzacions, hi troben en aquests sistemes un complement de seguretat molt útil.

És més que evident que els Honey Pot només tenen sentit en xarxes on la seguretat és un element molt important, fins hi tot vital. Xarxes que pertanyen a organitzacions financeres, empreses de seguretat, multinacionals importants, asseguradores, etc.

4.5.3 Pros i contres

Pros

· Són fàcils d'implementar, ja que no calen grans coneixements de seguretat. Tot i que mai estàn de més, els coneixements.

· La informació que ens proporcionen és molt fiable.

· No tenen cost d'execusió.

· Permeten recollir informació de l'intrús per poder-lo perseguir després. Podem observar els seus passos i veure d'on provenen les seves conexions o la informació que usa sense pressa.

· Permet conèixer el nivell de coneixements que té l'intrús. Segons el seu comportament a l'hora d'entrar en els sistemes del Honey Pot.

· Dóna temps als administradors a respondre davant la intrusió.

· Permet conèixer millor les diferents psicologies dels intrusos a l'hora d'atacar.

Contres

· Assumeix que l'atacant és una mica ignorant o igenu, ja que sovint els Honey Pot són fàcilment detectables.

· Ràrament un intrús experimentat s'empassa l'asqué.

· En moltes xarxes és un cost innecessari.

· Només és útil per xarxes grans, amb més d'una DMZ per protegir.

5. On s'han de col·locar els IDS?

Una de les coses més importants en els IDS és a quin lloc de la xarxa el col·loquem, perque segons on estigui veurà un tipus de tràfic o un altre. Sobre tot, en les xarxes actuals que totes es basen en switch la importància de la situació del IDS és pràcticament vital. Cal anar molt en compte doncs en les decisions que prenem a l'hora de col·locar els IDS.

A vegades cal posar una mica d'imaginació al tema, ja que no sempre són necessaris tants IDS com pensem, doncs simplement col·locant més sensors i una mica d'imaginació es poden estàlviar molts diners i aconseguir un nivell de seguretat més gran.

5.1 DMZ

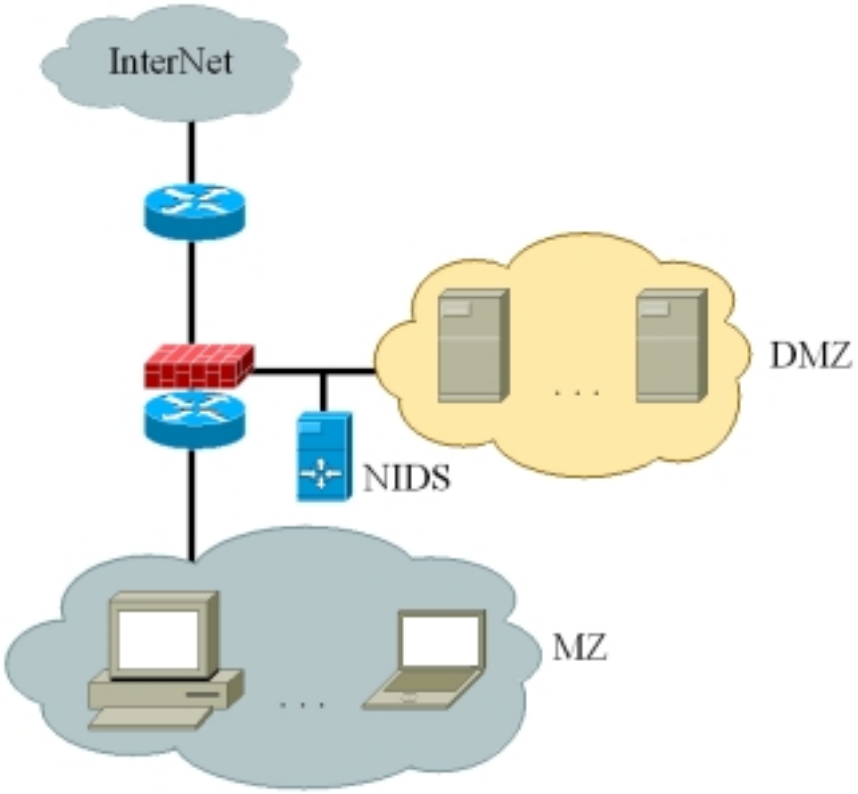

Ilustración 3 - NIDS col·locat a la DMZ

Quant l'única part que es vol auditar són els servidors externs de l'organització és una bona idea col·locar el IDS entre el firewall i la DMZ. Sovint aquestà situació es dóna en Intranets petites on només hi ha una DMZ formada per un sól switch. Una bona idea llavors és connectar el cable que surt del firewall cap al switch a un hub, on hi connectarem el firewall, IDS i el switch de la DMZ. D'aquestà forma el IDS veurà tot el tràfic que passa per aquell punt. Si aquest treballa sense IP aquest serà totalment transparent.

Si el switch usat a la DMZ és programable, llavors no es necessita per res el HUB ja que es pot programar el switch perquè repeteixi les sortides de tots els ports per un sol port. En aquest port llavors s'hi pot penjar l'IDS. En els switch de Cisco això sempre és possible fer-ho. Una bona solució pel cable que col·loquem entre el switch/hub i el IDS és posar-hi un cable sense la pota de TX grimpada. Així el IDS mai podrà enviar informació per aquella interficie de xarxa. Si el canvi no el volem fer a nivell físic, que sempre són els més segurs, també podem prohibir que aquella interficie envii paquets.

Aquest últim punt en un sistema Linux és tan senzill com per mitjà del Netfilter dengar tots els accessos a la interifice de sortida:

```
# iptables –P output –j DROP
```

Simplement amb aquestà ordre és impossible que cap interficie de xarxa envii paquets. Per un sistema xBSD la idea és la mateixa pero l'ordre és *ipfw* en comptes de *iptables*.

5.2 Fora del firewall

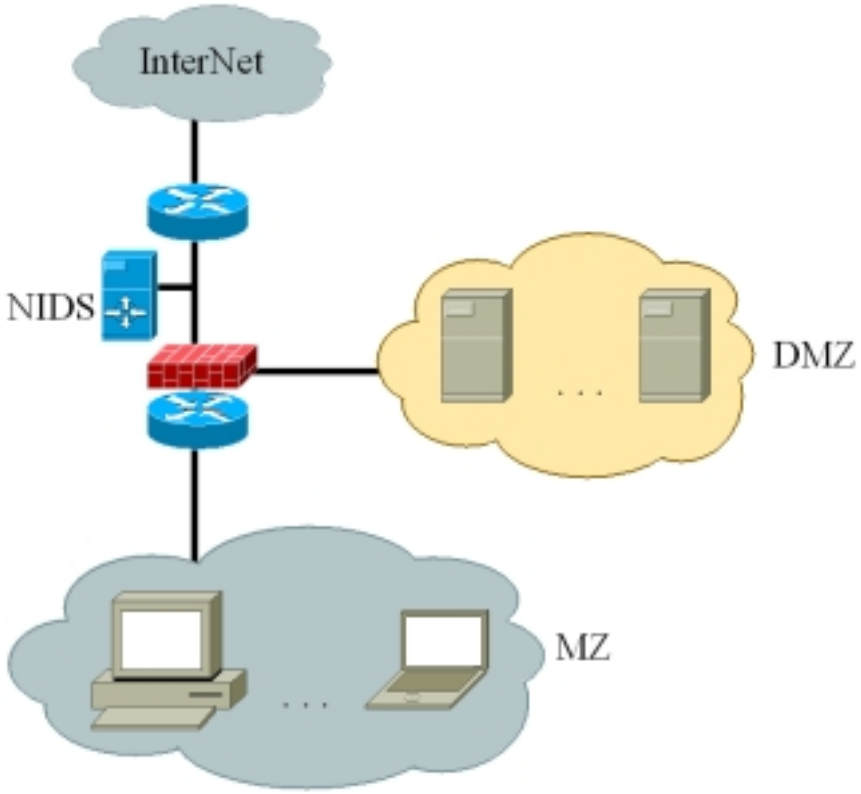

Ilustración 4 - NIDS col·locat fora del Firewall

L'IDS queda entra el router extern i el primer firewall de l'organització. En aquest punt es pot veure tot el tràfic que entre i surt de la nostra Intranet. Les informacions que es dónaran en aquest punt seràn moltíssimes, ja que pel IDS passaran tots els atacs que normalment arriben al firewall. Si decidim col·locar un IDS en aquest punt òbviament haurem d'amagar-lo molt, ja que si es pogués comprometre la seva seguretat seria una *backdoor* d'accés als nostres sistemes.

Amagar-lo pot ser tan senzill com aplicar l'explicat en el punt anterior tan a nivell físic (cable sense TX) com a nivell lògic (denegar la sortida de paquets a la interficie). La interconnexió entre el router extern i el firewall, es pot montar igual que he explicat en el punt anterior amb el hub.

Col·locar un IDS només en aquest punt no té massa sentit si no és per comparar-lo amb la informació d'un altre IDS després del firewall. D'aquestà forma, es pot comprobar que el firewall està ben configurat i que filtra tot el tràfic que ens interessa.

5.3 Distribució

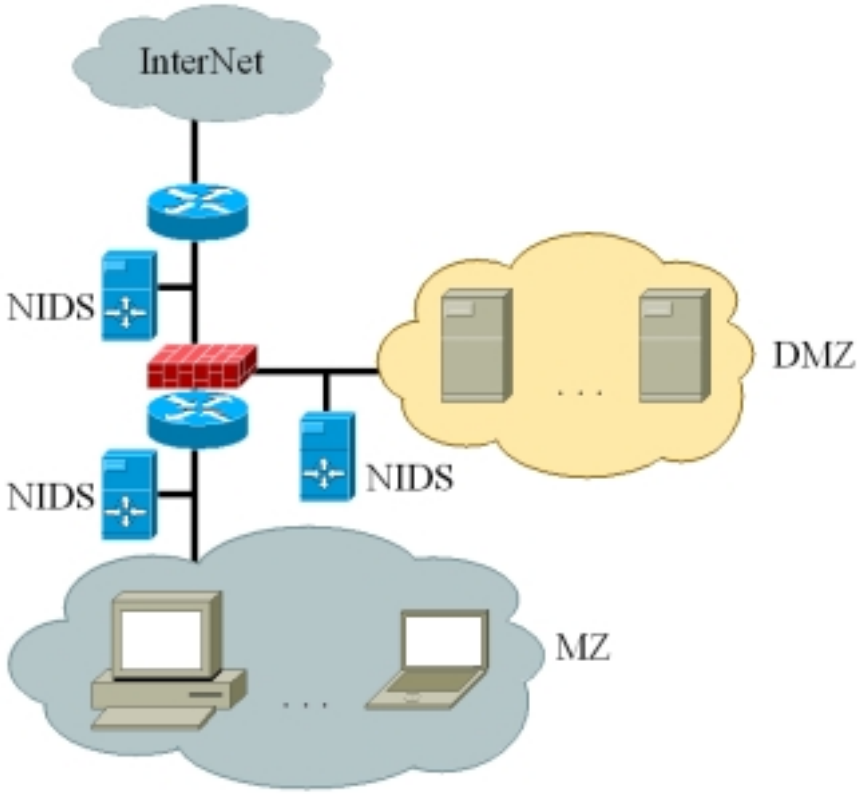

Ilustración 5 - NIDS distribuït per la xarxa

Quan les xarxes comencen a fer-se grans i volem controlar diversos punts de la mateixa, ja no hi ha cap configuració estàndard que s'ajusti a les nostres necessitats; llavors hem de començar a distribuir els sensors per diversos punts de la xarxa. Aquestà sempre acaba sent la solució aplicada, ja que segons el disseny de les xarxes i les necessitats de seguretat de les administracions s'han de prendre un tipus o un altre de decisions.

Seguint amb el model de xarxa que hem estat estudiant fins ara, podem observar en la figura adjunta com quedarien distribuïts els IDS. Amb aquestà configuració es té una visió molt profunda del tràfic sospitós que circula per la nostra xarxa. Quan disposem de configuracions d'aquest tipus és interessant perdre molt de temps configurant la importància de les alarmes i la gravetat dels events que vagin capturant els IDS, ja que gràcies a la seva posició estratègica si es comparen els registres que es generen en els diferents IDS es poden treure conclusions molt interessants i estàlviar un gran nombre de falços positius.

5.4 Què diu l'experiència?

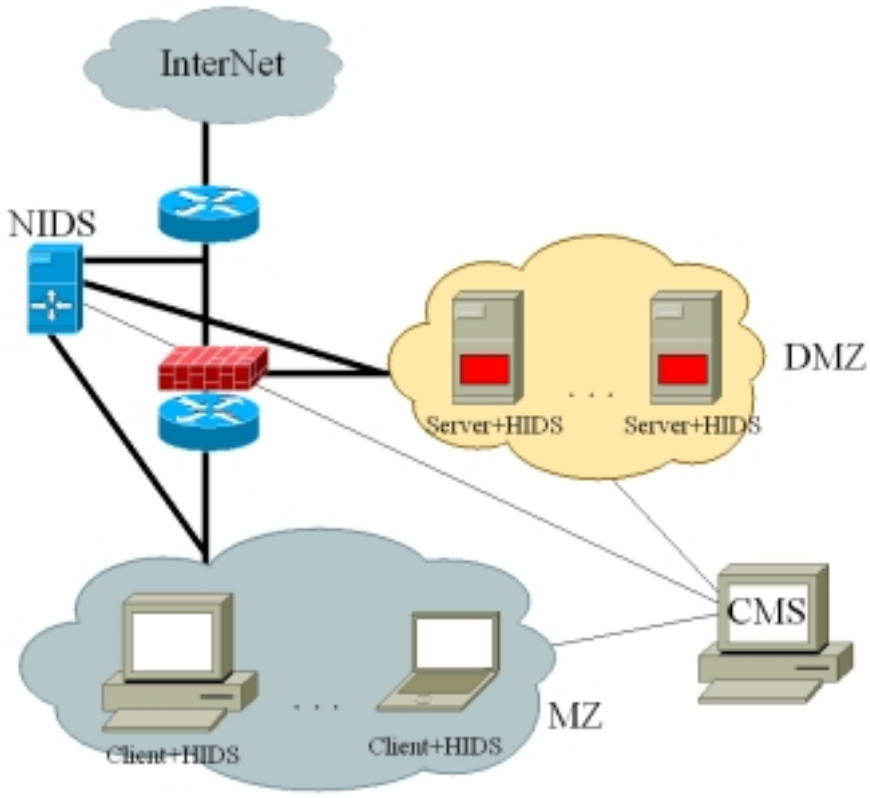

Ilustración 6 - Esquema d'on col·locar un IDS en un cas real

Com sempre en seguretat la millor arma és l'experiència i la imaginació. Així doncs s'aprofitarà aquestà secció per explicar com es configuraria la xarxa que venim estudiant fins ara. Com es reduirien costos i es màximitzaria la optimització del rendiment dels IDS.

En comptes de seguir l'exemple del model de distribució que col·locava tres IDS un a cada branca del firewall, l'experiència diu que si col·loquem un ordinador prou potent amb tres interficies de xarxa connectades una a cada branca del firewall és més fàcil contrastar la informació que s'ha d'analitzar. A més es redueixen costos de forma dràstica perquè ara ja no es necessiten tres IDS un per cada branca. Sinó que amb un IDS i tres interfícies de xarxa es fa tot.

L'IDS per una quarta interficie de xarxa s'ha de connectar al CMS (Console Management System) que és des d'on s'analitzaràn les dades amb més profunditat. A més els servidors de la DMZ és interessant proveïr-los també d'un HIDS que s'ajusti a les necessitats de cada tipus de servidor, ja per últim als clients amb informació més sensible també se'ls pot col·locar un HIDS. Tot això ha d'anar interconnectat amb el CMS per una xarxa parlal·lela a la xarxa de l'organització, per evitar que aquestes dades siguin susceptibles de ser intervingudes.

Malgrat no seria una mala idea crear túnels xifrats dins de la xarxa de la organització per portar aquestà informació. No seria aconsellable seguir aquest camí, ja que malgrat no sigui visible la informació que transporten. Sempre són més fàcilment detectables aquests túnels que no una xarxa paral·lela per on circular la informació referent a la seguretat de l'organització.

5.5 IDS en entorns commutats (switched)

La dificultat d'implementar un IDS en un entorn on tot va connectat a switch, és que el IDS no pot veure el tràfic que passa per la xarxa i que no va dirigit a ell. Per tant, s'ha d'idear alguna forma perquè aquest tràfic a més d'anar cap al seu destí també passi per

l'IDS de forma que aquest malgrat no intervenir en el tràfic, pugui observar-lo en busca de possibles paquets nossius.

Bàsicament s'usen tres idees diferents per solucionar aquest problema: sapnning ports, HUBS i TAPS. Els spanning port són configuracions que es fan als switch perquè facin comportar a un dels seus ports com un HUB. Però no tots els switch suporten aquest tipus de configuracions i si ho soporten i el tràfic és molt elevat es poden crear colls d'ampolla.

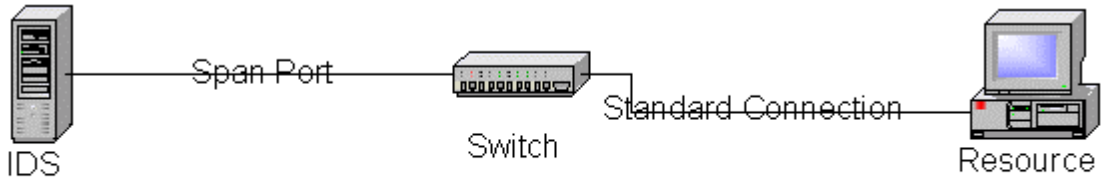

Ilustración 7 - Esquema d'un IDS i un recurs

Les opcions d'usar HUBS o TAPS són similars. Ambdues solucions consisteixen en col·locar un d'aquests dos dispositius entre la màquina a monitoritzar i el switch al que va connectat. La única diferència entre un HUB i un TAP és que els HUBs són bidireccions, permeten rebre i enviar tràfic. En canvi els TAP només permeten rebre tràfic. Per tant, un IDS connectat a un TAP és físicament impossible que pugui enviar paquets a la xarxa.

Exemple de HUB entre la màquina a monitoritzar i el switch:

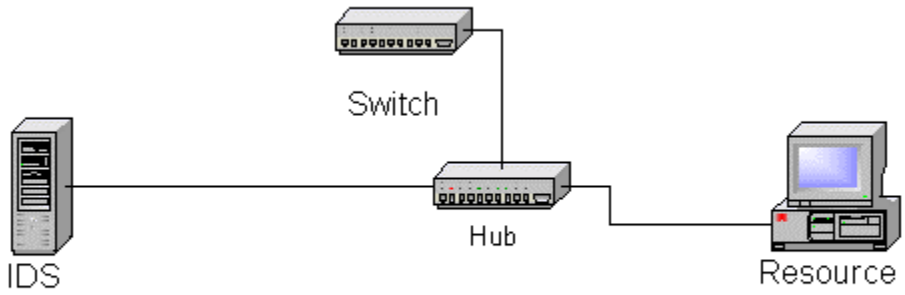

Ilustración 8 - Esquema IDS - HUB - switch

Un parell d'exemples de configuracions amb TAP:

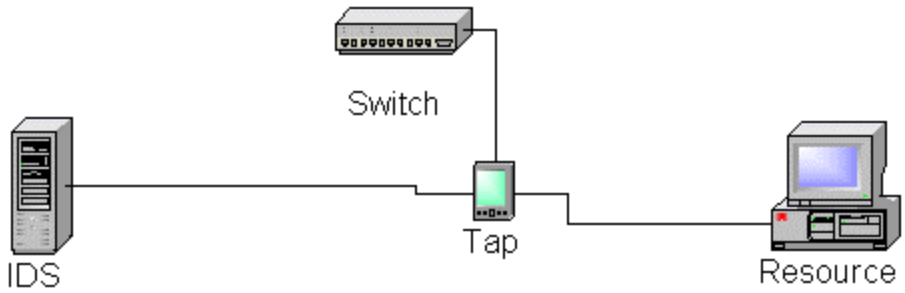

Ilustración 9 - Esquema IDS - TAP –switch

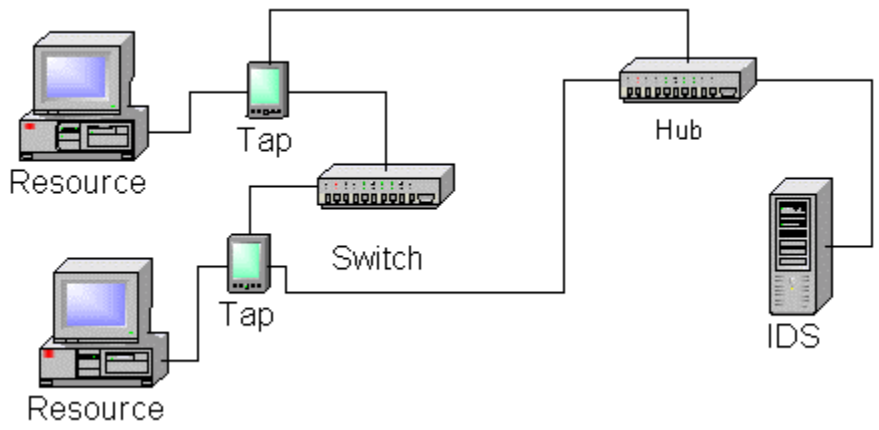

Ilustración 10 - Esquema múltiples TAP - HUB - switch i IDS

6. Eines que complementen els IDS

En aquest apartat es preten dónar consciència de quin és el principal problema dels IDS, com s'intenta solucionar el problema i quines són les eines que ens poden ajudar a complementar la tasca que realitza un IDS.

6.1 Falços positius

6.1.1 Què són els falços positius?

Es donen quan tràfic legítim d'una xarxa és comparat amb un patró de tràfic nociu, aquest per error és considerat com il·legítim. Aquest fet es dóna milers o milions de vegades al dia en qualsevol sistema IDS. Això obviament satura d'events a la sortida del IDS i fa disparar alarmes de forma innecessaria.

Malgrat sembla un fet poc usual després d'explicar la seva descripció, a continuació es posarà un exemple aplicat de com un IDS pot generar un o fins hi tot milers, de falços positius al confondre tràfic legítim per tràfic maliciós.

Per exemple, un usuari que intenta connectar a una unitat de xarxa i que no recorda la seva paraula de pas després de provar amb diferents claus de pas durant una bona estona aquest usuari es desespera i comença a intentar connectar de forma repetida i reiterada la seva unitat de xarxa. Entremig d'aquests intents de connexió hi ha un NIDS que està monitoritzant el comportament de l'usuari. Què veurà aquest NIDS?

· El primer que veurà és un seguit d'intents de connexió a la unitat de xarxa amb diferents paraules d'accés. Això si es fa suficients vegades el pot fer pensar que es tracta d'un intent d'accés per força bruta i començarà a generar events informant d'aquest fet.

· En segon lloc veurà que el client no para de fer intents de connexió a una unitat de xarxa sense parar i amb una freqüència molt elevada. Aquests intents si són prou ràpids i repetitius poden fer pensar al IDS que es tracta d'un intent d'atac de denegació de servei (DOS), generant altre cop una serie d'events informant d'aquest fet.

S'acaba de comprobar com el comportament legítim d'un usuari autoritzat de la nostra pròpia xarxa, ens ha fet generar una sèrie d'events que podem considerar falços positius.

6.1.2 Com es poden evitar?

Elminar aquests events no és gens fàcil, ja que en altres circumstàncies aquest mateix tràfic que ara sembla un falç positiu seria una alerta real. Per tant, s'ha d'estàr molt alerta a l'hora d'activar o desactivar certs tipus de fonts típiques de falços positius. En aspectes com el que ara es tracten és on es fa més important l'experiència i el coneixement profund de l'administrador sobre la seva xarxa: tipus de tràfic, horaris de màxima càrrega, comportament dels usuaris, etc.

De totes formes hi ha una sèrie de mètodes que es poden intentar seguir per evitar aquests falços positius:

· Prova i Error: simplement anar provant les idees que surtin de la imaginació i l'experiència, només els errors poden ajudar a fer-ho bé.

· Analitzar els històrics de tràfic: és una bona font d'on treure l'experiència i el coneixement sobre la nostra xarxa.

· Usar patrons de tràfic estàdístics: montant algún sistema de mesura de càrrega i estudi del tràfic de la xarxa podem aprendre molt sobre el comportament dels usuaris dins de la xarxa. Hi ha una màxima que sempre es compleix: un 10% dels usuaris generen el 80% del tràfic.

· Múltiples polítiques basades en la funcionalitat i la criticitat de la informació: si la informació que es vigila té una criticitat molt alta potser ens interessa no deixar escapar cap falç positiu ja que aquest podria reprensentar una font d'informació molt valuosa.

· Contrastar els events generats per aquest sensor en la informació d'altres registres dels sistems, altres sensors, registres de les aplicacions, etc.

6.2 Analitzadors de logs

6.2.1 Què són els logs?

Els logs són uns registres que generen les aplicacions, sistemes operatius, IDS, etc. És un històric dels events rellevants que es van dónant en aquell servei. Obviament aquests registres tendeixen a crèixer de forma descontrolada i de forma molt ràpida. Això fa que sigui impossible visualitzar-los tots amb el detall que es mereixen.

Per evitar que la quantitat d'informació que es genera no acabi desbordant s'ha d'anar molt en compte amb la sensibilitat que es configuren les aplicacions a l'hora de generar events, ja que si es decideix guardar qualsevol moviment després no es podrà absorvir tota la informació que es genera. Per altre banda, si el que es fa és dónar poca sensibilitat ens pot passar per alt informació necessària.

6.2.2 On es guarden?

Degut a la gran varietat de registres que contenen els sistems i més els IDS cal tenir alguna política d'enmagatzematge de logs, sinó aviat la capacitat dels sistemes es pot veure desbordada. Així doncs, és interessant perdre el temps necessari en dissenyar o en adoptar una política centralitzada d'enmagatzematge d'aquests registres.

Moltes aplicacions guarden aquestes dades en simples fitxers plans, en els directoris creats a cada efecte dins de cada sistema. Però si la política acaba sent deixar-los en aquests fitxers caldrà pensar que aquests fitxers poden crèixer fins a tornar-se intractables i potser el que s'ha de montar llavors és algun sistema d'enmagatzematge circular o alguna altra alternativa que impedeixi la tendència al creixement infinit que tenen aquests fitxers.

Actualment la política que s'acostuma a seguir és tenir un SGBD (Sistema Gestor de Bases de Dades) que enmagatzemi tota la informació, per tenir-la centralitzada i fer-la més tractable. Tot i que aquestes tecnologies ens les infraestructures orientades al PC estàn poc desenvolupades la tendència apunta clarament cap aquest costat.

Al tenir tota la informació centralitzada en un lloc lògic (no té perquè ser un sol lloc físic) és més fàcil desenvolupar o ajustar les nostres eines d'analisis de logs per tal de millorar i optimitzar el seu funcionament. En aquest punt ja es veu doncs, de forma evident quina és la tasca dels analitzadors de logs.

6.2.3 Com es revisen?

Obviament tota aquestà informació no serveix de res si no és filtrada per algun element que en tregui les conclusions necessàries i que sigui capaç de prendre decisions després de l'observació i l'estudi de la informació dels logs. Aquestà tasca tal com es comenta anteriorment es fa pràcticament inviable degut al desmesurat volum d'informació a tractar.

Per tant, ens hem de dotar d'eines com els analitzadors de logs que després de filtrar tots aquests registres seguint unes pautes previament programades ens deixaràn a la vista només la informació que realment ens interessa. Aquest procés d'automatització no és gens trivial i només després de revisar i reconfigurar repetides vegades les pautes de treball de l'analitzador de logs es podrà obtenir un resultat realment bo.

Tota la informació que ofereixen aquests logs en principi no interessa fins que dónada una incidència s'hagi de repassar de forma exahustiva la informació que contenen. Fins que no arribi aquest moment, només repassant la sortida de l'analitzador de logs n'hi ha més que suficient.

Com és evident el principal enemic a l'hora de realitzar aquestà tasca és la gran quantitat d'informació a revisar i la monotonia que això suposa es fa intractable. Per això

cal esmerar-se en ajustar i aconseguir un analitzador de logs prou potent i ben ajustat com per fer l'anterior punt el menys crític possible.

6.3 Comprobadors d'integritat de fitxers

Hi ha alguns HIDS que ja incorporen aquestà funció, perquè és una eina molt útil. Tots els sistemes tenen fitxers de configuració o comandes del sistema operatiu que és interessant saber que no han estat alterades per un intrús, ja que de la resposta d'aquestes ordres o de la bona configuració d'aquests fitxers en depénen les decisions a l'hora de considerar que un sistema està funcionant de forma correcta i que no ha patit cap intrusió.

Per exemple, en els sistemes Unix hi ha unes eines molt conegudes usades pels intrusos anomenades: *rootkits*. Aquestes eines s'encarreguen de substituir algunes de les comanades vitals del sistema per comandes modificades que no deixin cap mena de rastre a les accions que realitza l'intrús que s'ha fet amb el poder del sistema. Si l'administrador no s'adóna d'aquest fet no podrà detectar mai a l'intrús dins la màquina ja que les comenades que li permetrien fer-ho estàn modificades.

Si en canvi hi ha algún procés que s'encarrega de forma periòdica de comprobar que la integritat d'aquest tipus de fitxers no ha canviat, llavors es podrà confiar plenament en la informació que donen aquestes comandes.

6.4 Analitzadors de vulnerabilitats

6.4.1 Actius

Els analitzadors de vulnerabilitat actius a partir d'una base de dades de vulnerabilitats de diferents sistemes es dediquen a llençar una sèrie d'atacs contra les màquines escollides, retornant al final un informe amb totes les vulnerabilitats que s'han trobat en els sistemes analitzats.

Quan un analitzador de vulnerabilitats actiu es posa en funcionament s'ha d'anar molt en compte amb els tipus d'atacs que s'activen, ja que hi ha certs atacs que poden comprometre la continuitat de certs serveis o sistemes.

També cal saber que sovint aquests analitzadors de vulnerabilitats ens retornen respostes poc encertades: falços positius, perquè no sempre s'ataca al sistema per comprobar que és vulnerable a un *bug*. Sovint l'únic que miren és el rastre que deixa un servei: la versió, alguna propietat dels paquets que retorna, informacions de benviguda al servei, etc. Això a vegades els fa treure conclusions equivocades.

Probablement el millor analitzador de vulnerabilitats actualment és el Nessus (http://www.nessus.org) que per sobre de moltíssimes eines comercials, té una de les bases de dades de vulnerabilitats més grans i més actualitzada. A més està dotat d'un llenguatge de programació d'atacs (NASL) molt senzill d'usar i potent.

6.4.2 Passius

Un altre tipus són els que a partir de la informació recollida en els paquets que passen per la xarxa es determina que un sistema té algún tipus de vulnerabilitat. Aquests analitzadors són útils sobretot per identificar quina és la perillositat d'un paquet dins l'entorn d'un atac. Aquests tipus d'analitzadors els acostumen a portar els NIDS.

Per exemple, l'analitzador passiu més conegut i potent que es coneix actualment és l'Snort (http://www.snort.org) també sota llicència GPL igual que el Nessus tenen una base de dades molt actualitzada i potent. A més, també està dotat d'un llenguatge de programació que permet personalitzar els patrons de tràfic i que generin els nostres propis events.

7. Què és un CMS?

7.1 Definició

Quan hi ha més d'un IDS o sensor en una xarxa enviant alertes es fa necessari instal·lar no només un analitzador de logs sinó també un CMS, que mostri la informació més important que hem de mirar. Gràcies a aquests sistemes que actuen com un centre de control d'alertes i permenten profunditzar en els events que han disparat les alarmes, es poden absorvir grans volums d'informació de forma ordenada i sense desbordar l'administrador.

Així doncs, el CMS és el punt des d'on l'administrador ha de recollir les dades enviades pels sensors dels IDS. També pot ser el punt on hi ha el registre on es guarden els events que envien els diferents sensors que hi ha repartits per la xarxa. De forma que l'analitzador de logs el podem fer correr en aquest sistema. Fins hi tot si la xarxa no és massa gran podem fer correr un SGBD per guardar els registres dels events.

Gràcies als CMS també és comú realitzar informes periòdics que ajuden a justificar la despesa de les organitzacions en seguretat. A més també és útil per saber fins a quin punt l'organització està en el punt de mira dels intrusos, i fins a quin punt els sistemes són vulnerables.

7.2 Pros i contres

Pros

· Permet centralitzar el control de la seguretat de la xarxa.

· Dóna un accés ràpid a la informació important.

· Ajuda a identificar les vulnerabilitats que s'intenten explotar en els sistemes.

· Acostuma a proporcionar un entorn de treball agradable i ordenat.

· És fàcil analitzar, configurar i mantenir diferents sistemes des d'un sol lloc.

Contres

· Si s'arriba a comprometre la seguretat d'aquest punt es pot fer vulnerable tota la seguretat de la xarxa.

· En cas d'un atac massiu contra tota la xarxa es podria saturar el CMS per sobrecàrrega (atac DOS indirecte).

· Cal estàr familiaritzat amb el CMS que s'usi sinó ens pot desbordar.

· Cal ser cuidadós en les conclusions que es treuen de la informació que dóna el CMS. No s'ha d'oblidar que la informació que dóna pot estàr influènciada per falços positius mal filtrats.

· El seu cost acostuma a condicionar la seva compra.

· Són difícils d'integrar. No tots els IDS funcionen amb qualsevol CMS. El fabricant del IDS acostuma a condicionar el CMS a usar. D'aquí la necessitat d'usar registres en format XML.

7.3 Fins a quin punt són necessaris els CMS?

Com tots els elements de seguretat l'ús o no d'aquests dispositius depén dels gustos de l'administrador. El que si és evident és que quan més gran és la xarxa a gestionar més palesa es fa la necessitat d'algún element centralitzador de tota la informació que ens arriba.

El CMS només és un entorn que preten integrar el màxim d'informació referent als IDS de la xarxa i mostrar aquestà informació de la forma més resumida, ordenada i vistosa possible, amb la finalitat de simplificar la tediosa i sovint monótona feina del manteniment d'aquests sistemes.

Alguns avantatges com la connexió dels events generats amb bases de dades de patrons d'intrusions, proporcionada per tots els CMS, ajuden a conèixer i a estàr al dia de certes intrusions que podrien ser desconegudes fins aquell moment. D'aquestà forma

redueixen el temps necessari per identificar l'atac i optimitzar la resposta enfront a aquests events (analitzador de vulnerabilitats passiu).

7.4 Alguns CMS

Malgrat totes les firmes comercials relacionades amb el món de la seguretat tenen els seus IDS amb els seus CMS associats, altre cop el món *OpenSource* ens dóna solucions que superen les eines comercials. A continuació es descriuen dos CMS que treballen amb les sortides de l'Snort (IDS).

7.4.1 ACID

ACID (Analysis Console for Intrusion Database) és una part del projecte AirCERT del CERT/CC (http://www.cert.org/kb/acid/). Aquest projecte té moltes més pretencions que la de desenvolupar un CMS per l'Snort. Però, dins del seu projecte ha desenvolupat aquest producte que per si sol ja té un valor molt destàcable, per la seva simplicitat d'ús, potència i claredat a l'hora de mostrar-nos la informació que recull l'Snort.

Programat en PHP, analitza les sortides que l'Snort ha anat posant en un SGBD (MySQL o PostgreSQL). A més de poder analitzar dades que provenen d'un IDS, aquestà eina també és capaç d'analitzar events de certs firewalls i fins hi tot alguns dispositius Cisco. És una d'aquelles eines que la seva potència recau en la seva simplicitat d'ús. No obstant això es troben a faltar certs complements com una pantalla principal més configurable o que doni més informació al iniciar una sessió.

Funciona en qualsevol servidor Web que suporti un pre-processador PHP amb suport de connexió a MySQL o PostreSQL. Per exemple, Linux, FreeBSD, OpenBSD, Win32, etc.

7.4.2 DEMARC

Malgrat la seva llicència no és *GPL*, si que és *OpenSource*. Sobre tot la gran vistosistat de les pantalles i les moltes eines que complementen l'analisi d'events, el fan un CMS molt complert. Algunes de les funcions més destàcables que complementen la seva funció d'analitzar i resumir els events rebuts dels IDS són: control de la integritat de fitxers, control del funcionament de dimonis locals i remots, etc.

Funciona amb una estructura Client-Servidor, d'aquestà forma podem instal·lar clients en certes màquines i usar aquests clients com a HIDS que serveixen informació al servidor (CMS). Potser el problema més gran que té aquestà estructura són les grans depéndències de software que tenen els clients i el servidor per funcionar, la qual cosa, fa que el programa no sigui gens lleuger pel sistema.

Així doncs, el seu principal inconvenient és la velocitat a l'hora de tractar la informació enmagatzemada a la base de dades. Només suporta MySQL i està programat en *Perl*, la qual cosa obliga a tenir un Perl amb molts mòduls que serà la principal dificultat en la seva instal·lació aquest software. La documentació que adjunta és poc complerta i requereix un administrador experimentat o un sistema molt automatitzat per saber-lo configurar.

8. Vulnerabilitats i tipus d’atacs

8.1 Tipus d’atacs

La majoria dels atacs només arriben a penetrar els sistemes per camins molt concrets. Per exemple, certs atacs poden permetre a l’intrús llegir certs fitxers però no li permeten alterar cap part del sistema. Altres atacs poden permetre parar certs serveis o components del sistema però no permeten accedir a fitxers del sistema. Malgrat la varietat de capacitats que tenen els atacs només hi ha quatre formes diferents de violar aquestà seguretat: disponibilitat, confidencialitat, integritat i control.

· Confidencialitat: quan un atac permet a l’atacant accedir a informació per la qual no està autoritzat pel propietari de la informació.

· Integritat: un atac pot causar una violació d’integritat quan l’atacant, que no està autoritzat, canvia l’estat del sistema o altera alguna dada resident en el sistema.

· Disponibilitat: quan un atac compromet la continuïtat d’un servei o sistema.

· Control: si un atacant aconsegueix violar els controls d’accés a un sistema, de forma que aquest guanyi privilegis dins el mateix. Podem dir no només que s’ha violat el sistema de control del sistema, sinó també la confidencialitat, integritat i la disponibilitat.

8.2 Tipus d’atacs sovint detectats pels IDS

8.2.1 Escàners

Podem dir que un escàner ha tingut lloc quan un atacant envia a una xarxa o a un sistema diferents tipus de paquets. Usant la resposta a aquests paquets l’atacant pot conèixer el sistema que vol atacar. A partir d’aquí l’atacant pot conèixer les característiques i vulnerabilitats de la màquina que vol atacar. Els escàners no

comprometen la seguretat dels sistemes. De fet, un escàner només és una forma de mirar quins recursos està oferint a la xarxa aquest sistema, quina topologia té una xarxa, etc.

Les eines que es dediquen a fer escàners, se'ls anomena: *network mappers, port mappers, network scanners, port scanners, vulnerability scanners*... Els escàners ens poden oferir aquestà informació d'un sistema o xarxa:

· Topologia de la xarxa.
· Tipus de tràfic permés a través del firewall.
· Hosts actius a la xarxa.
· Sistema Operatiu dels sistemes que estàn online.
· Software que estàn corrent els servidors.
· Versions del software que estàn corrent els servidors.

Els escanejadors de vulnerabilitats són un tipus d'escàners que miren les vulnerabilitats que té un host. A més un atacant també pot llençar un escàner de vulnerabilitats contrà una col·lecció de hosts per saber si algún d'aquests host és susceptible a aquella vulnerabilitat.

Amb aquestà informació, un atacant pot identificar de forma precisa els sistemes de la xarxa destí, cosa que li permetrà saber quin tipus d'atac farà vulnerable un host o una xarxa destí. Aquests atacants sempre usen software que prova si un sistema és vulnerable a un atac abans de llençar l'atac real. Arriben a coneixer tant bé les xarxes que volen atacar que ni l'administrador a vegades sabia que tenia certs serveis dins la seva xarxa, certs programes o fins hi tot certs hosts.

Tot i que aquestes eines són usades per atacar sistemes, si un administrador es preocupa per la seguretat dels seus sistemes les usarà contra la seva pròpia xarxa per conèixer millor quines són les seves vulnerabilitats i poder-les corregir. Així doncs, tan per defensar com per atacar un sistema, sovint es fan servir les mateixes eines. Obviament només canvia la finalitat.

Desafortunadament per les víctimes aquestes accions no són il·legals, ja que aquestes eines no permeten veure res que l'administrador prèviament no hagi deixat visible des la xarxa. Així doncs, és tan legal com entrar a un banc i fixar-se en els dispositius de seguretat que té instal·lats: càmeres de seguretat, sistemes d'alarma, guardies de seguretat, etc. L'únic que s'està fent és mirar informació pública. En cap moment hi ha un intent de penetració al sistema.

Hi ha causes que justifiquen l'activitat dels escàners. Un exemple són els buscadors de pàgines web: google, yahoo, altavista, etc. Aquests sistemes d'indexació de pàgines web, tenen els tan coneguts escàners que es dediquen a recollir informació pública dels sistemes en busca de noves pàgines web per tal de poder-les indexar en els seus motors de bases de dades.

Si el nostre IDS és prou potent i està prou ben configurat serà capaç de distingir entre un escàner legítim o un escàner mal·liciós. Perquè no cal oblidar que abans d'un atac sempre i hi ha un escàner que el precedeix. A més l'experiència ens diu que si una màquina està connectada a internet és molt extrany que no rebi almenys un escàner diari.

8.2.2 DOS (Denegació de Servei)

Els atacs DOS (Denial Of Service) pretenen alentir o parar sistemes o serveis. En certes comunitats d'Internet els atacs de tipus DOS són molt comuns. Per exemple, a l'IRC (Internet Relay Chat) les disputes verbals són comuns i una forma d'aconseguir denegar l'accés a algú a aquell servidor d'IRC és per mitjà d'un atac DOS. Una altra pràctica per la que s'ha usat els atacs DOS últimament és per atacar grans organitzacions. Davant d'aquests atacs no hi ha cap organització per gran que sigui que es pugui defensar si l'atac és a prou escala, provocant a l'organització atacada grans pèrdues.

A grans trets hi ha dos tipus d'atacs DOS: l'explotació de defectes (flaw explotation) i els atacs per inundació (flooding). Per un administrador d'IDS és molt important conèixer la diferència entre ells.

8.2.2.1 Flaw exploitation DOS attacks

S'aprofiten dels defectes del software que té la màquina a atacar provocant que aquest software falli i que caigui el servei, provocant una violació de disponibilitat. També es pot aconseguir que aquest servei acabi esgotant els recursos de la màquina fins que aquestà caigui o bé el servei es quedi bloquejat.

Un dels atacs d'aquest tipus més conegut és: el 'Ping of Death'. Aquest atac bàsicament consisteix en enviar paquets de ping molt i molt llargs contra sistemes Windows. El sistema destí no els pot gestionar i el sistema acaba caient.

Quan un atac d'aquest tipus es diu que acaba els recursos del sistema, es vol dir que acaba: el temps de CPU lliure, la memòria, l'espai en disc, l'espai en algún buffer, o fins hi tot l'amplada de banda. En molts casos, només corregint els errors de programació d'aquest software es soluciona el problema, aplicant un 'patch'.

8.2.2.2 Flooding DOS attack

Aquests tipus d'atac l'únic que fan és enviar més informació al sistema de la que aquest pot gestionar. Quan un atacant no té cap forma d'enviar més informació que la que un sistema pot gestionar el que fa és monopolitzar altres sistemes de forma que tots ells enviïn informació contra el sistema que es vol atacar. Usant tota la potència i l'ample de banda de tots aquests sistemes llavors s'intenta acabar amb els recursos del sistema destinatari de l'atac. La xarxa o sistema atacat no té cap forma de defensar-se sobre aquest tipus d'atacs. Només parar els sistemes fins que l'atac hagi acabat i l'organització pugui tornar a oferir els seus serveis a la xarxa.

Quan ens referim al terme DDOS (Distributed DOS) aquest tipus d'atacs són un subconjunt dels atacs DOS. Es diu que un atac és un DDOS quan l'atacant usa múltiples sistemes per llançar l'atac. Tots aquests ordinadors són controlats per un atacant central. És

obvi que un sol atacant no pot saturar la xarxa d'una important organització de comerç electrònic. Però si l'atacant és capaç d'aconseguir el control de 20.000 ordinadors i fer que aquests enviin un atac de forma simultànea contra un sol destí és obvi que el destinatari de l'atac caurà davant aquest atac DDOS.

8.2.3 Atacs de penetració

Els atacs de penetració impliquen aconseguir o alterar uns privilegis no autoritzats dins d'un sistema; també es poden referir a aconseguir recursos o dades d'un sistema als que no s'està autoritzat. Es pot considerar doncs arribats en aquest punt que estem violant la integritat i el control . En contraposició als atacs DOS que només vulneraben la disponibilitat o als escàners que ni tan sols feien res il·legal.

Per tal d'aconseguir guanyar aquest accés als sistemes els atacants s'aprofiten d'errors en el disseny o la programació del software d'un host. Tot i que també és molt comú aprofitar-se de descuits o del desconeixement de l'administrador sobre el seu propi sistema. Malgrat els errors en el software (bugs) varien molt cada dia, els més comuns son:

Usuari a Root: un usuari del sistema aconsegueix el control absolut sobre la màquina on es troba.

Accés remot a Usuari: un atacant des de la xarxa aconsegueix un compte d'usuari a la màquina que està atacant.

Accés remot a Root: un atacant des de la xarxa aconsegueix control total sobre la màquina que està atacant.

Accés remot de lectura a disc: un atacant des de la xarxa aconsegueix accés de lectura a disc en zones privades.

Accés remot d'escriptura a disc: un atacant des de la xarxa aconsegueix accés d'escriptura a zones privades del disc.

8.2.4 Atacs remots vs Atacs locals

Tan els atacs DOS com els de penetració es poden dónar en dues formes: local i remota.

8.2.4.1 Usuari autoritzat

Des d'una compte d'un usuari autoritzat del sistema és llença un atac contra el mateix sistema. Per tant, el que busca aquest atac normalment és escalar privilegis, per tal d'aconseguir més control sobre la màquina.

8.2.4.2 Usuari públic

Aquests atacs es llencen sense comptes del sistema. Normalment s'aprofiten d'algún servei que està dónant el sistema i per mitjà d'aquest punt d'interacció per un accés públic es llença un atac normalment aprofitant una vulnerabilitat del software, un error de programació de la interficie d'intereacció, etc. Per mitjà d'accés públic és busca aconseguir normalment un accés com a usuari del sistema. Si s'aconsegueix això ja s'haurà aconseguit escalar privilegis dins el sistema i llavors només s'haurà de seguir escalant privilegis per fer-se amb el control total.

8.2.5 Com determinar des d'on es produeix un atac

Quan un IDS detecta un atac normalment informa del lloc des d'on s'ha llençat l'atac, sovint s'expressa en forma d'IP origen (source IP). Aquestà IP és només el lloc des d'on aparentment s'ha llençat l'atac, o sigui, la IP des d'on venien els paquets que estàven atacant el sistema.

Sovint quan un intrús vol fer un atac el que fa és canviar la seva IP per fer els atacs, per tal de que la font d'aquests paquets no l'apunti cap a ell. Aquestà tècnica de canviar la IP que ens pertoca s'anomena IP Spoofing.

La clau per saber on és l'atacant està en mirar el tipus d'atac. Segons l'atac que hagi llençat canviarà on hagi d'estàr l'atacant per veure els paquets que retorna el sistema atacat.

Si l'atacant llença un atac d'una sola direcció, com alguns atacs de flood (DOS) aquest no haurà d'esperar cap resposta del sistema i per tant podrà estàr en qualsevol lloc. Fins hi tot és usual en aquests casos que l'atacant usi IP origen aleatories. Quan l'atac que es llença sí que necessita que l'atacant vegi els paquets que retorna l'atacat, per exemple, en un atac de penetració, el més usual és que l'atacant usi algún sistema intermig on dirigir els paquets, o sigui, tornem a la idea de IP spoof.

Exemple de com fer un IP spoof, es localitza un sistema intermig (algun ordinador en el que tinguem accés i que estigui connectat a la xarxa) s'instal·la algún soft per poder-lo usar de porta d'enllaç. Per exemple, es pot instal·lar un simple *port-bouncer*. O també es pot instal·lar un sniffer modificat si aquest soft està en un lloc estratègic, de forma que capturi la informació que va cap a una IP que no existeix i que aquestà informació capturada s'envii encapsulada en paquets des d'aquest host cap al host de d'on es vol fer l'IP Spoof.

Amb aquestes idees podem estàblir connexions TCP completes, connexions que envien els paquets cap a un destí amb l'origen alterat i revent per un camí poc convencional les respostes. Per tal de dur a terme accions com aquestes sense haver de passar hores programant, hi ha eines com el *netcat* que permeten fer coses realment molt poc usuals amb els sockets. Convinant aquestes eines amb d'altres comandes del sistema o clientes de serveis completament convencionals és molt fàcil montar un IP spoof.

En situacions com les descrites anteriorment no es pot determinar sense l'ajuda de l'administrador de la xarxa atacant qui ha fet l'atac, ja que s'escapa del control de la víctima seguir fins aquests extrems els paquets que s'envien. També cal pensar que si es

rep un atac d'aquest nivell de sofisticació els atacants tenen cert nivell i que no serà fàcil seguir-los la pista.

8.2.6 Excessiva informació generada per l'IDS

Quan els IDS generen centenars o milers d'events diaris, es fa completament impossible estudiar-los per separat. El problema no està només en el número d'atacs sinó en com els IDS mostren aquests atacs. Alguns IDS mostren per separat un atac cada cop que es dóna contra un host. Un atacant que escanegi una subxarxa pot llençar centenars de vegades un atac, una per cada host i això generarà centenars d'events. Un per cada atac. Actualment sembla que la tendència ens porta a mostrar tots aquests atacs agrupats i a mostrar en primer llocs els de més importància.

8.2.6.1 Conveni de noms pels atacs

Gràcies a aquest conveni es fa més fàcil comparar l'eficàcia dels diferents IDS davant d'un atac, ja que abans cada fabricant els anomenava d'una forma diferent i era difícil saber quin IDS era el més complet. A més, també es feia difícil coordinar i organitzar la informació que apareixia als CMS provinent dels IDS de diferents fabricants.

Els conveni més extés és: Common Vulnerabilities and Exposures List (CVE): mantingut per MITRE amb l'ajuda de diferents professionals al voltant del món. Des del NIST's ICAT-vulnerability index (http://icat.nist.gov) es pot accedir a la base de dades del CVE , tot i que el seu web site és: http://cve.mitre.org.

8.2.6.2 Intensitats dels atacs

Normalment quan un IDS informa d'un event s'associa el nivell de perillositat de l'event associat. Així els administradors tenen una orientació de la importància de l'event. Tot i que l'impacte d'un atac i el nivell de perillositat són molt subjectius, a més no tenen

perquè ser iguals en diferents sistemes o en diferents organitzacions. Per exemple, si un atacant llença un grup d'atacs per unix i la xarxa atacada no conté cap sistema d'aquest tipus l'impacte serà nul. Però en canvi aquest atac pot ser considerat de perillositat màxima per l'IDS. Per això, com sempre, en els IDS el millor és aprendre del que va dient i anar-lo ajustant al cas concret de l'organització en concret.

8.3 Tipus de vulnerabilitats dels sistemes

Alguns IDS mostren una descripció de l'atac que han detectat; sovint també inclouen el tipus de vulnerabilitat que l'atac està intentant explotar. Aquestà informació és molt útil quan l'administrador vol corregir la vulnerabilitat del sistema. A continuació es pretenen discutir els tipus de vulnerabilitats més comuns i no preten pas ser una llista exahustiva de tots els tipus de vulnerabilitats existents, ja que la llista seria intractable. Les vulnerabilitats més comuns són:

8.3.1 Error en la validació d'entrades

En un error en la validació de les entrades, la entrada rebuda pel sistema no es comproba correctament. Aquestes vulnerabilitats a vegades es poden explotar enviant certes seqüències d'entrada. Hi ha dos tipus d'errors de validació:

8.3.1.1 Buffer Overflow

En aquest cas l'entrada rebuda pel sistema és més llarga que la llargada esperada per aquell tipus d'entrada però el sistema no sap tractar aquestà condició excepcional. Llavors el buffer d'entrada s'omple i sobrepassa la memòria que tenia assignada. Si es construeix una entrada de forma intel·ligent un atacant pot fer que el sistema executi instruccions capturant la seqüència del contador de programa (PC: Program Counter). Un exemple de buffer overflow és explotar la vulnerabilitat del dimoni fingerd en un sisgtema Unix, en el qual, un atacant pot enviar una comanda finger amb una sèrie d'arguments

massa llargs, seguit per una cadena d'ordres col·locades en un punt d'aquestà serie d'arguments que s'executaran quan el sistema no sapiga tractar els arguments enviats.

8.3.1.2 Límit de les condicions d'error

Les entrades rebudes pel sistema poden ser generades per una persona o per una màquina, causant que el sistema excedeixi un límit assumit. Aquestà sobrecàrrega representa una vulnerabilitat. Per exemple, el sistema podria quedar-se sense memòria, sense capacitat al disc o sense ample de banda. Un altre exemple, és que una variable podria arribar al seu màxim valor i superar-lo o al seu mínim valor. Encara un altre exemple podria ser que una equació intentés fer una divisió per zero. Tots aquests són casos que s'haurien de mirar de tractar per tal de que els sistemes no es quedessin bloquejats, provocant problemes de disponibilitat.

8.3.2 Error en la validació d'accés

Els errors d'aquest tipus són molt senzills d'explicar, ja que simplement es pot dir que es donen quan el mecanisme d'autenticació, o de control d'accés fallen. El problema no acostuma a estàr en la configuració de l'usuari sinó el propi mecanisme del control d'accés. O sigui, que serien problemes de disseny.

8.3.3 Error en la captura de condicions excepcionals

En certs programes es donen certes condicions excepcionals que porten a un error que en el moment del disseny no es van pensar. Aquests errors s'haurien de capturar pel software. D'aquestà forma s'evita que l'usuari o atacant arribi en punts del programa que li poden proporcionar informació que no estàn autoritzats a veure. Un error que es dóna sovint en els sistemes Unix, són els *coredump* que és un volcat de la memòria que estàva usant un programa abans de dónar-se una situació d'error no controlada pel programa. Aquest fitxer coredump porta informació a vegades sensible sobre el sistema.

8.3.4 Errors d’entorn

A vegades els sistemes no són insegurs de per si, sinó, que certs programes que s’instal·len els fan vulnerables. Aquests errors indirectes a causa de software sovint innecessari que s’instal·len als sistemes, es consideren errors d’entorn. Normalment són errors que fan el sistema inestàble ja sigui per incompatibilitat entre aplicacions o entre l’aplicació i el sistema operatiu.

8.3.5 Errors de configuració

Aquests errors són molt comuns, sovint perquè els programes s’acostumen a instal·lar ràpid i a configurar el mínim per dónar el servei que interessa aprofundint el mínim necessari en el coneixement del que s’ha instal·lat. Això fa que sovint per la ignorància sobre el software instal·lat apareguin problemes de seguretat fruit de la mala o fins hi tot la no configuració de certes parts del software. Per exemple, si un servei d’FTP porta usuaris per defecte i aquests no s’eliminen es podrà entrar al sistema usant-los.

8.3.6 Errors de disseny

Són errors que es donen per un mal plantejament a l’hora de planificar les configuracions. Això pot provocar que es plantegin incongruències entre diferents configuracions, provocant buits o zones que l’administrador no havia previst. Per exemple, en una xarxa funcionant sota un domini NT si s’instal·la un nou ordinador i aquest no es configura perquè entri dins del domini NT de l’empresa, aquest no usarà el sistema d’autenticació d’usuaris del domini NT.

8.4 Tècniques anti-IDS

D'aquestes tècniques n'hi ha moltíssimes, però en aquest punt es pretén centrar el tema en un cas concret perquè quedi clara la vulnerabilitat dels sistemes IDS intentant detectar atacs. Malgrat pugui semblar que els IDS són realment una gran solució contra els atacs, després d'aquest apartat es podrà observar que això no és tan cert. En concret s'observarà com evitar els patrons de tràfic de que disposen els IDS per detectar els atacs als servidors Web. Malgrat el descrit en aquest punt només és un exemple aplicat sobre HTTP la idea es podria portar a terme en altres serveis.

8.4.1 La sintaxis

És important entendre els components que té una petició HTTP. Tal com es defineix al RFC 1945:

[GET[1] /cgi-bin/foo.cgi[2] HTTP/1.0[3]][4]

1. Per fer una petició HTTP les comandes són: GET, HEAD, POST, etc.
2. URI: Uniform Resource Identifiers, o dit d'altre forma la pàgina que nosaltres estem demanant. Pot ser relativa ('/algun/fitxer') o absoluta ('http://servidor/algun/fitxer').A vegades també s'anomena URL (Uniform Resource Locator) a una URI absoluta.
3. Versió en la que es fa la petició i es vol la resposta d'HTTP.
4. Les 3 components anteriors fa una petició. Cada component es separa per un espai.

Per tal d'entendre una mica més com es fa per evitar un IDS, es farà referència a dos tipus d'IDS:

Smart: Implementació lògica que entén el protocol que estem enviant, en aquest cas el HTTP. Després d'optimitzar la petició que es fa, intenta aplicar-li algún patró de tràfic

maliciós. S'intenta comportar com un servidor Web. Un exemple de producte que segueix aquest model de funcionament és el RealSecure de la ISS (http://www.iss.net).

Raw: També coneguts com a 'packet grep IDS', escanegen el payload del paquet sense processar la informació que conté. La millora d'aquest sistema és purament de velocitat. Per exemple, l'Snort usa aquest sistema d'identicació de tràfic maliciós.

Ambdós tipus d'IDS tenen els seus pros i contres, i ambdós tipus poden ser evitats de diferents formes. S'ha trobat interessant referir-se als IDS segons la forma d'identificar el patró de tràfic maliciós per entendre d'una forma més tècnica com evitem el seu sistema d'indentificar tràfic maliciós.

En pro de la velocitat hi ha molts IDS que intenten reduir al màxim les cadenes a identificar dins dels seus patrons de tràfic sospitós. Per exemple, un IDS intentant detectar el típic bug del /cgi-bin/phf, podria ser que només busqués la cadena /phf dins la URI de la petició HTTP. Això obviament provocaria un gran número de falços positius. Però al mateix temps milloraria substancialment la velocitat d'aplicació d'aquell patró. Un exemple en el que fallaria aquest patró de tràfic seria:

GET /imatges/phfax.jpg HTTP/1.0

Si només es busca que la URI contingui /phf, la petició anterior faria saltar un falç positiu. Però també cal anar en compte perquè si a la signatura posessim que s'ha de detectar la cadena /cgi-bin/phf i ens arribes una petició d'aquest tipus:

GET /cgi/phf HTTP/1.0

Això no seria considerat un atac pel nostre IDS i en canvi podria ser-ho. Per tan, el fet d'obviar /cgi-bin/ o /cgi/ ja que és un directori arbitrari que podria variar ens causa un compromís, ja que si fem la cadena massa curta (/phf) podem fer saltar falços positius i si fem la cadena massa llarga (/cgi-bin/phf) podem obviar atacs (/cgi/phf).

Fets com aquest són els que fan inferiors els IDS comercials als IDS de codi lliure, ja que no es pot comprobar quines polítiques estàn usant per detectar els atacs i per tant, no podem saber fins a quin punt són bons els seus criteris per detectar atacs.

8.4.2 Forma de coincidir

Alguns IDS no tenen en compte que no totes les peticions HTTP són a través de la sentència GET, que també poden ser altres comandes com: HEAD o POST. Per exemple, un CGI podria estàr programat per rebre una petició via HEAD i llavors les sentències:

GET /directori/fitxer.cgi
HEAD /directori/fitxer.cgi

serien equivalents i alguns IDS serien incapaços de veure que la segona sentència també és una sentència correcta i que pot representar un atac maliciós. Sovint el mètode de fer la petició no s'inclou.

8.4.3 Codificació de la URL

El sistema clàssic de saltar-se els patrons dels IDS és codificar la URI de la URL amb el seu equivalent en caràcters d'escapament. El protocol HTTP especifica que es poden passar caracters binaris dins la URI usant la notació %xx, on 'xx' és un número en hexadecimal que representa un caràcter. En teoria els raw IDS haurien de fallar, perquè la signatura "cgi-bin" no consisteix amb la cadena "%63%67%69%2d%62%69%6e". També, en teoria, els smart IDS no haurien de permetre passar a un paquet com aquest perquè primer decodificarien la cadena, com un servidor web, i després li aplicarien la comparació.

Actualment això no funciona en cap tipus d'IDS ja que tots d'una forma o altre decodifiquen la cadena abans de comparar-la amb les signatures. Concretament l'Snort, per exemple, usa pre-processadors de paquets per solucionar aquest obsolet problema.

8.4.4 Doble barra

En un esforç per trencar una cadena, el clàssic mètode de la doble barra que substitueix la barra simple. La idea seria canviar cadenes del tipus: /cgi-bin/fitxer.cgi per //cgi-bin//fitxer.cgi. Malgrat, la majoria d'IDS (tan smart com raw) ja solucionen aquest problema i els seus derivats, o sigui, usar múltiples barres. Els smart IDS tendeixen a combinar totes les barres en una, en canvi els raw IDS tenen dos comportaments: igual que el smart o bé, generar un event que informi del fet. Aquest mètode està obviament obsolet i actualment s'acostuma a usar el mètode de auto-referenciar un directori.

8.4.5 Salts inversos de directoris

Un altre clàssic per trencar el mètode de comparació de signatures dels IDS, és saltar cap a un directori i després fer el salt invers, per tornar a quedar al mateix lloc i fer la petició del fitxer que està al directori actual sense que ho sembli. La idea és que volem fer aquestà petició:

GET /cgi-bin/fitxer.cgi

però volem ocultar una mica el codi de la petició:

GET /cgi-bin/foodir/../fitxer.cgi

si ens fixem en la URI veiem que l'únic que estem fent és entrar en un directori per després tornar a sortir i fer la petició d'un fitxer que està al directori inicial.

En aquest tipus de joc els IDS ja no saben detectar que s'està intentant atacar al fitxer.cgi sinó que s'han de limitar a informar per mitjà d'un event de la presència de la cadena '/../' dins la URI de la petició. Aquest tipus d'events acostumen a quedar a la pila

d’events considerats falços positius, ja que una URI del codi HTML d’una pàgina web podria contenir aquestà referència i no seria en cap cas codi maliciós.

Per tant, acabem de trobar un mètode força eficaç per confondre tan els smart com els raw IDS. Tot i amb això, la tàctica dels directoris auto-referenciats és molt millor.

8.4.6 Directoris auto-referenciats

La tècnica més nova i eficaç en aquest “joc de directoris” són els directoris que s’auto-referèncien. Com és sabut per tothom en DOS i en Unix el ‘.’ és una referència cap al propi directori, per exemple: ‘c:\tmp’ és el mateix que ‘c:\tmp\.\.\.\.\.’. L’equivalent en Unix seria: ‘/tmp/’ és igual a ‘/tmp/./././’.

Els smart IDS en un intent d’evitar aquest tipus d’atac se’ls presenten tres opcions:

· Alertar de la presència de la cadena ‘/./’.

· Obviar aquestà cadena, perdent així molts atacs. (mala solució)

· Fer una substitució lògica de la cade ‘/./’ per ‘/’, que és la millor solució, però és massa lenta.

La realitat és que si convinem la tècnica de codificar la URI i la dels directoris auto-referenciats la majoria dels IDS ja siguin del tipus smart o raw no són capaços de fer la comparació de la signatura de forma encertada.

8.4.7 Finalitzar la petició abans de l'esperat

La idea consisteix en finalitzar la cadena de la petició de forma prematura, llavors l'IDS en un intent per optimitzar el seu temps de procés, obviarà la informació extra que envia el client. Per més informació sobre el que envia un navegador web en cada petició al servidor es pot consultar http://oriol.homeip.net/?id=179 (browser fingerprinting) on hi ha un article sobre aquest tema. Una petició web típicament és de la forma:

GET /some.file HTTP/1.0\r\n
Header: blah \r\n
Header: blah \r\n
Header: blah \r\n
Header: blah \r\n
\r\n

un exemple real de petició feta per Mozilla seria:

GET / HTTP/1.1
Host: 192.168.1.1:8000
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:0.9.3) Gecko/20010801
Accept: text/xml, application/xml, application/xhtml+xml, text/html;q=0.9, image/png, image/jpeg, image/gif;q=0.2, text/plain;q=0.8, text/css, */*;q=0.1
Accept-Language: en-us
Accept-Encoding: gzip, deflate, compress;q=0.9
Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66
Keep-Alive: 300
Connection: keep-alive

Obviament un IDS no es pot posar a examinar totes les capçaleres perquè el seu rendiment disminuiria moltíssim. Per tant, el IDS s'ha de limitar a mirar estrictament el

que és la petició. Però s'ha d'anar molt en compte perquè es podria rebre una petició del tipus:

GET /%20HTTP/1.0%0d%0aHeader:%20/../../cgi-bin/some.cgi HTTP/1.0\r\n\r\n

això es traduiria a:

GET / HTTP/1.0\r\nHeader: /../../cgi-bin/some.cgi HTTP/1.0\r\n\r\n

O si es vol, a:

GET / HTTP/1.0\r\n
Header: /../../cgi-bin/some.cgi HTTP/1.0\r\n
\r\n

Això seria una petició vàlida. Assumint que l'IDS ho decodifica, s'aturaria a la primera part, fins la nostra finalització prematura de la petició, aconseguint que no s'examini la segona part i saltant-nos les comparacions de signatures de l'IDS.

8.4.8 Ocultació de paràmetres

Els paràmetres d'una petició típicament es passen, per exemple, així:

fitxer.php?nom=rfp&prog=exemple&param=...

Obviament les dades dels paràmetres no s'han de tractar, si només estem buscant peticions de certs fitxers. Altre cop en un intent d'aconseguir estàlviar temps en el processat dels paquets que ens arriben, un smart IDS pararia d'analitzar la cadena al arribar al '?', ja que obviarà que tot el que segueix són paràmetres. Usant la tàctica del punt 8.4.3 podem fer el següent:

GET /index.htm%3fparam=/../cgi-bin/fitxer.cgi HTTP/1.0

que es traduirà com:

GET /index.htm?param=/../cgi-bin/some.cgi HTTP/1.0

Altre cop, això és una petició vàlida. La solució correcta seria extreure la informació decodificada abans de codificar els caràcters. Això funcionaria obviament amb els smart IDS.

8.4.9 Peticions HTTP mal formades

Com s'ha explicat ja, els smart IDS extreuen la URI d'una petició HTTP, ignorant els paràmetres quan s'està buscant un nom de fitxer. Tal com s'explica al HTTP RFC les peticions en la versió 1.0 són del tipus:

Metode<espai>URI<espai>HTTP/Versió CRLF CRLF

La clau està en que les peticions HTTP per separar les tres components de les peticons usen espais. A més les coponents sempre han d'apareixer en el mateix ordre. Això vol dir que és fàcil d'extreure un component de la petició, perquè només ens hem de fixar on hi ha un espai per extreure el component.

Però això és l'estàndard, si mirem com s'han fet les implementacions de l'estàndard podem veure que l'Apache, per exemple, també accepta aquestà sintaxi:

Metode<tab>URI<tab>HTTP/Versió CRLF CRLF

Això acaba de tirar per terra totes les pre-suposicions que assumeix l'RFC sobre com s'han de formar les peticions HTTP. Aquestes incongruències entre els estàndards (RFC) i les implementacions del mateix (p.ex. Apache) facin passar per alt certs paquets als IDS que només s'han guiat pel que deia l'estàndard o per les implementacions més conegudes d'un estàndard. Obviament que la restà del software funciona igual.

8.4.10 URL llargues

Certs raw IDS per tal d'optimitzar el seu funcionament només es fixen en els primers xx bytes de la petició. En general això funciona bé. Però per molts caràcters que posem en la petició la URI ha d'estàr a la primera línia d'aquestà petició. O sigui, la petició podria ser algo així:

GET /rfprfp<molts caràcters>rfprfp/../cgi-bin/fitxer.cgi HTTP/1.0

La clau està en introduir suficients caràcters com perquè l'IDS no arribi a examinar la part de la URI que podria identificar com un atac. Per saber quants caràcters s'han d'introudir s'ha de mirar el codi font de com està implementat l'IDS. O bé, anar provant fins a comprobar que aquest ja no registra cap event al fer la nostra petició.

8.4.11 Sintaxi de directoris DOS/WIN

Tothom coneix la història de que Microsoft usa '\' per separar els directoris perquè Unix usa '/'. Malgrat que l'estàndard de HTTP, diu que s'ha de posar '/'. Per tant, els IIS (Internet Information Service) de Microsoft internament han de fer el canvi de '/' a '\', al igual que la restà de servidors web basats en DOS/Windows. En canvi si s'usa '\' en les peticions contra un servidor amb plataforma DOS/Windows aquestes peticions són acceptades. Per tant, la URI '/cgi-bin/fitxer.cgi' contra un servidor DOS/Windows podria ser enviada com: '/cgi-bin\fitxer.cgi'. Aquest fet no està contemplat en gairebé cap IDS. Per cert, cal fixar-se en que la URI comença amb '/' i no amb '\'.

8.4.12 Procés del mètode NULL

Moltes llibreries de C usen el caràcter NULL per marcar el final d'una cadena de caràcters. Malgrat es pot dubtar si els IDS s'han programat amb llibreries d'aquest tipus, si

que ens podem adónar que és comú usar el caràcter NULL per denotar el final d'una cadena de caràcters. Per tant, es pot usar aquest fet per de la següent forma:

GET%00 /cgi-bin/fitxer.cgi HTTP/1.0

La teoria de la tàctica seria la següent:

El servidor web rep la petició, separant el mètode de la URI.

El servidor web decodifica el mètode i la URI.

El mètode encara és vàlid, per tant, el servidor web podrà processar la petició malgrat el caràcter NULL. Per altre banda, el IDS al decodificarà la petició sencera i pararà al trobar-se el caràcter NULL. Per tant, no s'adónarà de la URI que segueix a la petició.

Tot això suposant és clar, que l'IDS faci el tractament de la petició tot de cop, la qual cosa és una suposició molt raonable. Per tal d'estàlviar temps. Cal dir però, que el servidor Apache no és capaç de tractar una petició que contingui els caràcters d'escapament '%00' o '%2f'. Malgrat que aquests si que funcionen en els servidors IIS de Microsoft.

8.4.13 Sensibilitat a majúscules i minúscules

Els sistemes DOS/Windows no fan distinció en el seu sistema de fitxers entre majúscules i minúscules mentre que els sistemes Unix si que la fan. Això vol dir que depénen de la plataforma del servidor les peticions al fitxer: 'index.htm', 'INDEX.HTM' i 'Index.Htm' poden ser el mateix. No saber fer les comparacions de forma correcta per aquest motiu es dóna en molt pocs IDS. Però és un fet que s'ha de tenir en compte.

8.4.14 Separant els paquets de la sessió

Molts IDS tan els smart com els raw només examinen el paquet actual. Es tracta d'explotar aquest fet enviant la petició HTTP repartida en múltiples paquets. No es tracta d'un problema de fragmentació. Simplement de separació de les dades en diferents paquets sense que això sigui estrictament necessari. La única motivació d'aquestà acció és que el IDS no detecti quina és la nostre petició. Per tant, l'única forma que té l'IDS és re-ensemblar tota la sessió per saber que deia la nostra petició, la qual cosa li faria perdre molt de temps i no sempre es fa. Un exemple d'aquestà idea pot ser, que en comptes d'enviar "GET / HTTP/1.0" en un sol paquet, ho enviem en múltipes paquets: "GE", "T ", "/", " H", "T", "TP", "/1", ".0".

9. El mercat

L'oferta del mercat en IDS és molt amplia, tan que es fa difícil conèixer tots els IDS que hi ha. Tot i que molts d'aquests productes tenen una cosa en comú i és un preu prohibitiu o molt elevat per les poques, nules o fins hi tot inferiors característiques als productes GPL.

9.1 IDS Comercials

A continuació es pot veure una llista amb els IDS comercials més importants del mercat:

Tabla 1- IDS comercials

Producte	Companyia	Directory service supported	Característiques de seguretat	Preu
DirectoryAlert	NetVision Inc.	Novell NDS Edirectory	Automated enforcement of security policies. Self healing reversal of actions which violate security policies. Secures access to directory, file system and OS.	$19.00 (US) per user less volume discounts.
ServerAlert	NetVision Inc.	Novell NDS EDirectory, Novell Bindery	Automated enforcement of security policies. Self healing reversal of actions which violate security policies. Secures access to directory, file system and OS.	$495.00/per server (US) per user less volume discounts.
Synchronicity	NetVision Inc.	Lotus Notes,Microsoft Active Directory,Novell NDS EDirectory,Exchange Directory,Novell Bindery,NT4 Domain,Unix,Linux	Automated enforcement of security policies. Self healing reversal of actions which violate security policies. Secures access to directory, file system and OS.	Free product requires support agreement at $5.00 (US) per user less volume discounts.
NETDEPLOY GLOBAL	Open Software Associates	Microsoft Active Directory	Digital signatures for file authentication. MD5 cryptography file digests	$100 per seat; volume discounts available

			ensure files are not corrupted during distribution and installation.	
bv-Control	BindView Corp.	Microsoft Active Directory,Novell NDS EDirectory,Exchange Directory,Novell Bindery,NT4 Domain,Unix,Linux	Comprehensive security assessment - out of the box, bv-Control checks for more than 600 areas of risk and can generate automatically more than 600 reports Baselining - BindView's exclusive technology that allows IT administrators to quickly document the content of their directory or the configuration of their file servers to uncover changes within their heterogeneous environments. Documentation against prior data can be performed daily, weekly or monthly to uncover discrepancies Find and Fix - BindView's exclusive technology that allows IT administrators to quickly identify and close security holes, enforce security standards and implement security policies across the enterprise Real time performance monitoring and RapidFire security updates Documents all users with excessive rights to key areas Locates stale or duplicate user accounts Finds user accounts with weak, expired or no passwords Documents effective rights on any file or directory Event log analysis RAZOR Team (security team comprised of "White Hat" hackers) support	For pricing information, please contact a BindView representative at 713-561-4000.
bv-Admin	BindView Corp.	Microsoft Active Directory,Novell NDS EDirectory,Exchange Directory,Novell Bindery,NT4 Domain	Universal access - secure, Web-based administration capabilities Secure Distributed System Administration - unique role-base delegation capabilities enable	Please contact a BindView representative for pricing at 713-561-4000.

			administrators to group discrete management tasks into roles and assign these roles to trustees Administrators can also estàblish enterprise-wide policies for account access controls and password properties.	
WebConsole Universal IT Management Suite	PatchLink Corporation	Novell NDS EDirectory,Novell Bindery,NT4 Domain,Unix,Linux	WebConsole leverages network authentication, only those people with appropriate access rights are authenticated to the network through WebConsole. Additionally, WebConsole obtains Secure Sockets Layer (SSL) and 256-bit RSA encryption from any standard Web browser.	$ 1,995 per license
DS Expert	NetPro Computing, Inc	Microsoft Active Directory,Novell NDS EDirectory	Two levels. i.) Client connectivity requires the appropriate credentials to the directory to access monitor component. ii.) Monitor NLM requires read, compare, and browse access to portion of eDirectory that it will be monitoring (could include entire tree).	$12/user with volume pricing available. Also can be bundled with DS Analyzer, for $18/user.
DS Analyzer	NetPro Computing, Inc	Novell NDS EDirectory	N/A	$12/User object defined
DirectoryAnalyzer	NetPro Computing, Inc	Microsoft Active Directory	Tighter security measures being introduced in the upcoming product offering	$12/User object defined. Volume pricing available
DirectoryInsight	NetPro Computing, Inc	Microsoft Active Directory	Three levels. i.) Client connection (via browser) supports standard and SSL connectivity/authentication. ii.) Access to client component is controlled through Active Directory groups. iii.) Windows 2000 service, during installation you can configure which credentials the service uses in order to read the directory.	$5/user with volume pricing available. Also can be purchased with DirectoryAnalyzer, for $3/user.

DirectorySim	NetPro Computing, Inc.	Microsoft Active Directory	Client only application with licensing mechanism.	The software is leased to consultants at a list price of $5,000 per month. Enterprise pricing starts at $25,000 and escalates according to a server-based pricing model.
Microsoft Operations Manager 2000	Microsoft Corporation	Microsoft Active Directory	takes advantage of domain security model and only monitors what it has privileges for	- $849 per managed CPU
Directory and Resource Administrator	NetIQ Corp.	Microsoft Active Directory, NT4 Domain, Exchange 2000	Integrates with NetIQ's Security Manager; forwards audit data for alerts, automated risk management and incident response by enabling Security Manager to read event logs that Director and Resource Administrator writes to the Windows NT and Windows 2000 event log; enables policy enforcement, auditing and reporting. Offers a secure 3-tier architecture, creating a protected business layer between the end user and managed data.	Starts at $16 per user
NetIQ Extended Management Pack (XMP) Solutions for Microsoft Operations Manager (MOM) 2000	NetIQ Corp.	Microsoft Active Directory, Novell ADS eDirectory, Exchange Directory, Novell Bindery, NT4 Domain, Unix, Linux	XMP for Anti-Virus extends MOM 2000 to view, monitor and manage antivirus applications from McAfee, Symantec and Trend Micro. The XMP ensures that the protection is running up-to-date and set acccording to your desired configuration.	Contact NetIQ for details on individual XMP pricing

D’aquestà llista en els que cal parar més atenció són:

9.1.1 NFR

NFR Security ha desenvolupat un grup de programes que pretenen ser la millor solució per detectar atacs, mal ús de la xarxa i les anomalies associades amb els recursos dels sistemes. Aquest set de programes preten cubrir les necessitats de seguretat tan a nivell de xarxa, com de servidors i estàcions de treball (workstations). Aquest conjunt de software de NFR està compost per: NFR Network Intrusion Detection (NID), NFR Host Intrusion Detection (HID) i per NFR Secure Log Repository (SLR).

Els productes de NFR pretenen compartir les següents característiques:

· Alertes fàcils d'entendre.
· Una facilitat inherent, a l'hora d'implementar el software i a l'hora de mantenir-lo.

*NID:*orientat a detectar anomalies en els paquets que passen per la xarxa, connectat amb diferents repositoris de vulnerabilitats.

HID: orientat a detectar anomalies a l'interior dels hosts: modificacions, lectures o borrat d'informacions. També controla l'execusió de programes no autoritzats, detallant els logs del sistema i generant informes que ens ajuden a prevenir nous atacs en el futur.

SLR: és un centre de recol·lecció, dispòsit i control de logs que accepta logs de qualsevol font i que afegeix la informació que rep en un sistema de monitorització més visual i comprensible, podent a més generar informes a partir d'aquestà informació.

A més la NFR també ha desenvolupat una eina anomenada BackOfficer Friendly que és un petit sistema d'alarma que ens avisa quan algú intenta penetrar en els nostres sistemes. Obviament la sensibilitat de la mateixa es pot programar. A més està pensada per l'ús personal i detecta de forma automàtica l'intent d'accés a "Trojan Horses" en el nostre sistema. Aquestà aplicació es distribueix de forma lliure, tot i que no és OpenSource.

9.1.2 RealSecure

La proposta de la ISS és RealSecure. Aquest producte és un híbrid entre un NIDS i un HIDS. El que preten fer aquestà eina és comparar la informació que genera el HIDS amb la del NIDS i a partir d'aquí comparar això amb els patrons de tràfic maliciosos per augmentar així la seva eficàcia. A més també integra totes aquestes funcions amb un sistema d'alarmes molt configurable, com és lògic també es poden associar respostes automàtiques a aquestes alarmes.

Aquest producte està dissenyant per simplificar al mínim la tasca de l'administrador de seguretat integrant-se amb les xarxes i els sistemes d'aquestes, controlant les aplicacions, el tràfic, els usuaris, etc. Però sempre intentant dónar el mínim de feina possible a l'administrador. A més els paràmetres de configuració del Realsecure es poden ajustar per situacions de xarxes molt diverses i també es poden integrar amb un sistema de consola centralitzat (CMS).

El cor del sistema Realsecure és el RealSecure Manager, una consola de control que ens permet configurar la restà de components de Realsecure de forma centralitzada: RealSecure OS Sensors i RealSecure Network Sensors. Aquestà consola recopila tota la informació generada pels sensors i informa de forma ordenada i visual del que va revent. A més Realsecure suporta agents tan de Windows NT com de UNIX. El més destàcable d'aquestà aplicació és poder controlar des d'un sol lloc tots els sensors.

Realsecure Network Sensors ha de correr sobre un servidor dedicat per tal de poder treballar de forma òptima. Cada RealSecure Network Sensor es dedica a mirar la xarxa en busca de patrons de tràfic suspitosos; aquest és l'indicador d'intrusions pre-atack del sistema. Aprofitant això es pot programar el RealSecure Network Sensor perquè actui en conseqüència quan detecta tràfic suspitós per la xarxa. Es pot fer que es talli la conexió, s'enviï un email o algun altre tipus d'alerta, que es grabi la sessió, reconfigurar el firewall o prendre alguna altra acció directament contra l'usuari. A més el Real Secure Network Sensor envia una alarma al RealSecure Manager o a la consola d'algun altre CMS perquè l'administrador en tingui constància de forma inmediata.

RealSecure OS Sensor analitza els logs del host on s'instal·la i identifica els atacs, determinant on ha estat originat l'atac, i mostrant informació forànea no disponible a temps real en l'entorn. Cada RealSecure OS Sensor instal·lat en una estàció de treball o en un host. Examinant els logs del sistema per trobar patrons que ens denotin un mal ús de la xarxa o forats de seguretat en el sistema. El RealSecure OS Sensor pot prevenir intrussions futures parant tots els processos del sistema que pertanyen a un usuari suspitós. Dónant més control sobre el sistema l'administrador, a més també es poden ajustar els tipus d'alarma que el RealSecure OS Sensor envia al centre de control, o enviar SNMP Traps, emails, i fins hi tot executar respostes programades.

A més la ISS ha desenvolupat una eina especialment pensada pels servidors: RealSecure Server Sensor aquestà eina és ideal per capturar qualsevol tràfic de la xarxa en organitzacions implementades sobre switch. També ens permet inspeccionar el tràfic de la xarxa a múltiples nivells de les capes de protocols, ja que aquestà inspecció a més de fer-se a la xarxa també es fa a dins el host font o destí de la informació. Això ens permet veure, per exemple, el contingut d'un paquet amb un payload encriptat (ex.ESP).

Per tal d'evitar la inundació del CMS amb falços positius el RealSecure Server Sensor fa un control dels events que s'han dónat. Els compara amb les alarmes que es disparen, per tal de fer arribar als administradors només la informació realment important.

Així doncs, la ISS ha desenvolupat tot un seguit d'eines que intenten cubrir tot els aspectes de la detecció d'intrusos, des de l'interior dels host, passant per la xarxa i sense oblidar les estàcions de treball. Malgrat el gran potencial ofert potser el seu punt feble més clar és la poca quantitat d'arquitectures suportades, la qual cosa, la fa una eina poc integrable en els entorns de treball reals.

9.2 IDS amb llicència lliure

Al món de l'OpenSource podem trobar també software molt interessant. A continuació es donen detalls sobre alguns dels IDS més famosos del món OpenSource.

9.2.1 Snort

NIDS OpenSource per excel·lència amb una càrrega molt petita sobre el sistema instal·lat. El fan una eina molt potent i lleugera. Permet analitzar el tràfic en temps real i capturar paquets en xarxes IP. També permet analitzar protocols buscant i trobant certs continguts i pot ser usat per detectar una gran varietat d'atacs i proves, com buffer overflows, stealth port scans, CGI atacs, proves SMB, OS fingerprinting, i moltes més coses. També té un sistema d'alertes en temps real, amb diferents formes d'avis: syslog, alerta ràpida, alerta completa, alerta SMB (WinPopup), sockets Unix, Tcpdump log, XML, bases de dades, CSV, unificat i SNMP Trap.

A més l'Snort te tres formes de funcionament diferents: pot ser usat coma sniffer, igual que el tcpdump; pot ser usat com a packet logger, per depurar el tràfic de la xarxa, o com un complet NIDS.A més suporta totes les plataformes que podem veure en la taula que hi ha a continuació:

x86	Sparc	M68k/PPC	Alpha	Other	
X	X	X	X	X	Linux
X	X	X			OpenBSD
X			X		FreeBSD
X		X			NetBSD
X	X				Solaris
	X				SunOS 4.1.X
				X	HP-UX
				X	AIX
				X	IRIX
			X		Tru64
		X			MacOS X Server
X					Win32 - (Win9x/NT/2000)

Tabla 2 - Plataformes suportades per l'Snort

L'Snort a diferència del software comercial comentat a l'apartat anterior és un programa molt lleuger, només detecta tràfic maliciós. Però és una eina que ens dóna una potència incomparable si el que es vol és implementar un NIDS. És ideal per ser el cor d'un IDS, ja que té infinitat d'eines que integrades amb la seva potència i en mans d'un bon administrador el fan infinitament millor que qualsevol altre software comercial.

Així doncs, l'Snort és un altre gran exponent de la idea Ciberpunk: 'do yourself'. Amb una llicència OpenSource i uns bons manuals, a més d'un llenguatge propi de programació de patrons de tràfic maliciosos, completament documentat i molt fàcil d'usar, fan de l'Snort una eina tan potent que només la imaginació li pot posar fi.

En contra seva cal ser conscients que no és una eina Plug-n-Play. Calen alguns coneixements per poder treure el rendiment que pot dónar, i depén molt del nivell de l'administrador la qualitat de la informació que aquest ens dónarà. Però potser el més perillós no és això sinó que s'ha d'estàr familiritzat en la constant evolució de l'OpenSource per poder controlar qualsevol possible anomalia del software, ja que a no sé que contractem un servei de mantenimnet d'outsourcing el manteniment de l'aplicació no tindrà altre suport que el que ens vulguin brindar els usuaris del producte.

No obstant al voltant de l'Snort hi ha tot un mercat d'empreses que venen el suport d'aquest producte; potser l'empresa més coneguda és: SiliconDefense (http://www.silicondefense.com/). Empresa que a més ha desenvolupat un seguit d'eines que permenten analitzar les sortides de l'Snort molt interessants:

*Snort Windows Installer:*com diu el seu nom és un instal·lador per la versió de Windows. Aquestà eina facilita molt les coses a les persones que no estàn familiaritzades amb el fet d'haver de compilar un programa abans de poder-lo usar. A més inclou l'eina IDScenter, que ens permet configurar des d'un GUI (Graphics Users Interface) tots els fitxers de configuració de l'Snort.

SnortSnarf: conjunt d'scripts fets en Perl que organitzen totes les alertes que genera l'Snort i les organitza en pàgines web.

Spade (Statistical Packet Anomaly Detection Engine) és una part del projecte SPICE. Aquest projecte desenvolupa un pre-processador que forma part de l'Snort com a plug-in del sistema. Aquest pre-processador envia alertes quan passen a través seu paquets amb anomalies. Aquest projecte encara està en estat experimental.

Com es pot veure en les eines a d'alt descrites SiliconDefense col·labora de forma molt activa amb el projecte Snort i distribueix totes aquestes eines de forma lliure. Tot i que també té tota una sèrie de serveis comercials, com pot ser el suport a l'aplicació Snort, com ja s'ha comentat.

Així doncs, si es sap buscar un entorn d'eines adients l'Snort pot ser la solució a totes les nostres necessitats de NIDS. Algunes bones eines que ajuden a interpretar les sortides de l'Snort han estat comentades anteriorment quan es parlava dels CMS. L'ACID (7.4.1) i el DEMARC (7.4.2) són dos molt bons complements per l'Snort.

9.2.2 LIDS (Linux Intrussion Detection System)

9.2.2.1 Què és LIDS?

En els models de Unix tradicionals, l'usuari root és un usuari que té poder total sobre el sistema. Està fora de totes les regles i regulacions del sistema de fitxers, a més es poden desenvolupar tasques que cap altre usuari té permís de fer. Per exemple, posar interficies en mode promiscu. Aquest accés de poder concentrat en un sol compte del sistema és una cosa molt perillosa, ja que comprometre aquest usuari vol dir comprometre tot el sistema. Per tant, si algún programa tingués una vulnerabilitat i aquest programa estés corrent com a root, s'hauria posat en perill tota la seguretat.

El LIDS (Linux Intrusion Detection System) és un 'patch' del kernel del Linux que permet treure poder a l'usuari root. Per tant, és un HIDS. Gràcies a aquest sistema es pot dónar exactament els permisos que necessiten els programes per funcionar. L'usuari root

pot acabar perdent tots els seus privilegis fins a quedar reduit com un usuari més del sistema. Es pot arribar a tenir un sistema montat sense haver-se de preocupar per si es compromet la seguretat d'un procés o si algún intrús maliciós pot tirar el sistema.

9.2.2.1 Com funciona el LIDS?

Un kernel amb el 'patch' del LIDS té moltes millores. Per començar, incorpora un detector d'escàner de ports, que pot usar-se per advertir a l'administrador d'un possible intrús. A més aquestà opció la podem escollir en el moment de compilar el kernel.

Una altre millora del LIDS és la millora del control d'accessos, o dit d'altre forma l'incorporació del concepte d'ACLs (Access Control Lists). El LIDS suporta dos tipus d'ACLs. Un tipus que controla els accessos a fitxers: read/write/append. I l'altre tipus és la capacitat de controlar els permisos dels processos; com ara canviar l'adreça d'un interficie de xarxa, o canviar l'identificador d'usuari (UID).

Moltes de les capacitats del LIDS s'han de configurar i escollir en el moment de compilar el kernel, ja que el LIDS en esència és un 'patch' del kernel. Les capacitats del LIDS són moltes i per decidir quines capacitats interessen i quines no, cal llegir-se la documentació del programa.

9.2.3 SNARE: Host-Based Linux Intrusion Detection

9.2.3.1 Què és l'SNARE?

SNARE (System iNtrusion Analysis and Reporting Environment) és un HIDS per Linux. Com tots els HIDS es basa en auditar el sistema per tal de detectar els intrusos després de que aquests entrin en acció. A diferència del LIDS el SNARE malgrat treballar en el kernel no s'ha de recompilar tot sencer per treballar amb ell, ja que permet carregar-se com a mòduls dínamics del kernel. A més, després d'instal·lar l'SNARE el nostre

sistema usarà un sistema d'auditoria del tipus C2. Això vol dir que compleix els estàndards de les regulacions del Departament de Defensa dels EU. Per exemple, si s'ha de treballar amb sistemes militars s'exigeix el compliment d'aquests estàndards C2. Obviament es distribueix sota llicència GPL.

SNARE ha estat desenvolupat per InterSect Alliance, una consultoria de seguretat de Camberra, Australia. L'SNARE va sortir al Novembre del 2001 i fins aquell moment no hi havia més HIDSs per Linux.

9.2.3.2 SNARE vs. Network-Based IDSs

L'SNARE (com tots els HIDS) permet detectar quan un intrús entra al nostre host, després de veure el primer signe d'activitat maliciosa. Auditant els events que es van generant es detecta tota l'activitat maliciosa. En canvi els NIDS, per altre banda, en un o més servidors monitoritzant un o més punt de la xarxa buscant patrons de tràfic sospitosos.

Obviament els HIDS són més eficaços que els NIDS, perquè deixen entrar l'intrús a la xarxa i per tant, quan un HIDS genera un event sempre té una raó de ser, en canvi el NIDS és més fàcil que esdevingui un falç positiu.

La principal avantatge de l'SNARE i qualsevol altre HIDS és que l'administrador es pot definir la sensibilitat de les activitats a registrar. Per exemple, en un servidor que conté informació confidencial, es pot especificar que qualsevol intent de pujar directoris sigui regitrat i que s'informi a l'administrador quan hi ha l'intent, el principal avantatge dels NIDS sobre els HIDS és que són més ràpids, fàcils i econòmics (en termes tan d'impentació com de sobrecàrrega).

El principal desavantatge de l'SNARE i els HIDS és que requereixen un administrador que sigui capaç de configurar les polítiques de seguretat que han de detectar els intrusos. A més també s'han de programar els procediments ha seguir quan es detecta un intrús, per tant, s'ha de tenir un coneixement molt profund del sistema que s'està protegint. Els procediments a programar acostumen a ser bloquejar ports a certes IPs, o fins

hi tot arribar a parar algun servei o el servidor en un cas extrem. Alguns exemples d'events crítics poden ser: llegir la base de dades d'autenticacions, escalar privilegis (via su), o provar de borrar algún fitxer dins el directori /etc.

9.2.3.3 Mòduls del kernel

L'SNARE ha estat implementat sobre mòduls del kernel, de forma que no s'incrementa la mida del kernel, evitant fer-lo més gran i lent. Els moduls del kernel del linux són de càrrega dinàmica i per tant, només es carreguen quan fan falta i quan ja no s'usen es descarreguen automàticament.

Gràcies a que treballa amb mòduls es té una gran avantatge que en el LIDS no es té, no cal recompilar el kernel cada cop que decidim incorporar o prescindir d'alguna de les funcionalitats del programa. El problema d'aquest sistema es troba en el risc de que es carregui un modul que pugui ser un *Trojan Horse,* o algun altre software mailiciós que incorpori noves vulnerabilitats, forats o 'back doors' al nostre sistema.

9.2.3.4 Auditant amb nivell C2

L'estàndard C2 normalment requereix un únic ID auditant cada grup de processos. Quan un procés fa una crida al sistema que podria afectar a la seguretat, es guarda un registre. Un exemple, de crides al sistema que han de ser registrades són per exemple els canvis de privilegis d'un usuari i els canvis de permisos de fitxers.

Això fa que els sistemes que segueixen l'estàndard de seguretat C2 generin moltíssima informació que ha de ser monitoritzada i interpretada. Per facilitar aquestà tasca l'SNARE permet aplicar diferents filtres a les crides a sistema. Es pot configurar de diferents formes per reduir o ampliar la informació que ha de generar. A més, la GUI del programa és fàcil d'usar i disposa d'una bona documentació online.

9.2.4 NESSUS

El projecte Nessus preten desenvolupar una eina OpenSource, potent, actualitzada i fàcil d'usar remotament com a escanner de seguretat. Un escanner de seguretat és un software que audita de forma remota una xarxa i determina per on els intrusos podrien entrar, comprometre o fer un ús nossiu dels nostres sistemes.

Nessus mai intentarà comprometre la seguretat dels nostres sistemes, només intentarà aconseguir la informació necessaria per informar d'alguna vulnerabilitat del sistema. A més parteix d'una premisa molt útil, premisa que pocs escàners de seguretat tenen. No associa els serveis amb un port dónat. Això vol dir que perquè hi hagi algún dimoni escoltant el port 22 ell no considera que aquest dimoni és l'SSH fins que no ha vist la seva signatura. A més tampoc considera que un servei té la versió que diu la seva signatura, sinó que intenta explotar la vulnerabilitat per comprobar que realment aquell servei és la versió que diu ser.

L'arquitectura de disseny del Nessus el fan molt ràpid i modular, permetent a l'administrador configurar-lo i ajustar-lo a les seves necessitats. A continuació es llisten algunes de les característiques que el fan tan potent:

· *Arquitectura de Plug-ins*: cada test de seguretat està implementat com un plug-in extern. D'aquestà forma és més fàcil que la comunitat OpenSource afegeixi les seves contribucions al projecte, sense haver de coneixer el codi del mateix.

· *NASL:* (Nessus Attack Scripting Language) és el lleguatge amb el que es programen els escàners que han d'explotar les vulnerabilitats dels sistemes. Aquest llenguatge està dissenyat per ser molt senzill i ràpid d'escriure. A més els escàners de seguretat també es poden implementar en C.

· *Base de dades actualitzada de vulnerabilitats:*la base de dades de tests de vulnerabilitats s'acutalitza diàriament. A més l'equip de desenvolupamet del Nessus està permanenment desenvolupant nous tests i discutin en els forums del programa les formes més óptimes de testejar aquestes vulnerabilitats.

·*Estructura Client-Servidor:* El Nessus està compost per dues parts: un servidor i un client. El servidor és el que fa els atacs i el client és el *frontend* de l'aplicació. Es poden llençar cada una de les aplicacions des d'un sistema diferent. A més els clients suporten diferents arquitectures: X11, Win32 i Java.

·*Pot testejar un número il·limitat de hosts a la vegada:* depénen de la potència de l'estàció on està corrent el servidor de Nessus, es poden testejar a la vegada tots els hosts que es vulguin.

·*Sistema avançat de reconeixament de serveis:* Nessus no considera que els hosts respecten la llista que publica la IANA, on s'associen els ports amb els serveis. Això vol dir, que el Nessus pot reconèixer un servidor d'FTP a un port no estàndard, per exemple.

·*Múltiples serveis:* si un servidor està corrent en més d'un port un mateix servei, el Nessus ho detecta. Per exemple, podem estàr corrent dos servidors Web un al port 80 i un altre al port 8080.

·*Tests intel·ligents:* els tests no es fan de forma sistemàtica sinó de forma intel·ligent. Per exemple, si es comproba que un FTP no té accés anònim no es passen els tests que pressuposen que tenim un servidor d'FTP anònim.

·*No es creu el comportament dels serveis:* si un servei en la seva versió x no és susceptible a certs tipus d'atac, el Nessus els passarà igualment per comprobar que realment el nostre sistema no és vulnerable a aquell atac. Malgrat que això impliqui fer caure la màquina.

·*Informes complerts:* Nessus no només informe del que està malament en la nostre xarxa, sinó que intenta ajudar a solucionar el problema i ens informa de quin nivell de risc suposa aquest problema pel nostre sistema.

·*Diferents formats d'informes:* Nessus pot generar informes en diferents formats: ASCII, LaTeX, HTML, HTML amb gràfics.

·*Suporta múltiples idiomes:* actualment els informes els pot generar en Anglès o Francés, però es pot dónar suport a més idiomes.

10. Implementar un IDS

10.1 Introducció

En aquestà part del projecte es pretén portar a la pràctica tot el que fins aquest moment s'ha tractat des d'un punt de vista teòric. No es pretén desenvolupar un IDS estrictament formal. Simplement s'intenta veure quina aplicació pràctica té el exposat fins aquest punt.

Degut als recursos econòmics dels que es disposaven per portar a terme aquest cas pràctic, l'entorn ha estat completament desenvolupat amb productes GPL o almenys OpenSource.

10.2 Objectius

L'objectiu d'aquesta part del IDS. Els objectius a intentar observar són:

· Desenvolupar un IDS a partir d'eines OpenSource.
· Comprobar el funcionament d'un producte d'aquestes característiques.
· Entendre en profunditat el funcionament dels IDS.
· Com es creen les signatures dels patrons de tràfic nossiu.
· Observar com treballa un IDS al comparar una signatura de tràfic.
· Discutir quin és el millor Sistema Operatiu pel nostre sistema IDS.
· On col·locar un IDS en un entorn ja operatiu.

10.3 Descripció de l'entorn

Aquest projecte s'ha portat a terme un entorn que ja està operatiu des de fa molts anys. En aquest punt s'intentaràn descriure els components d'aquest entorn. Per tal de poder discutir en posterioritat quins són els millors llocs pels sensors de l'IDS.

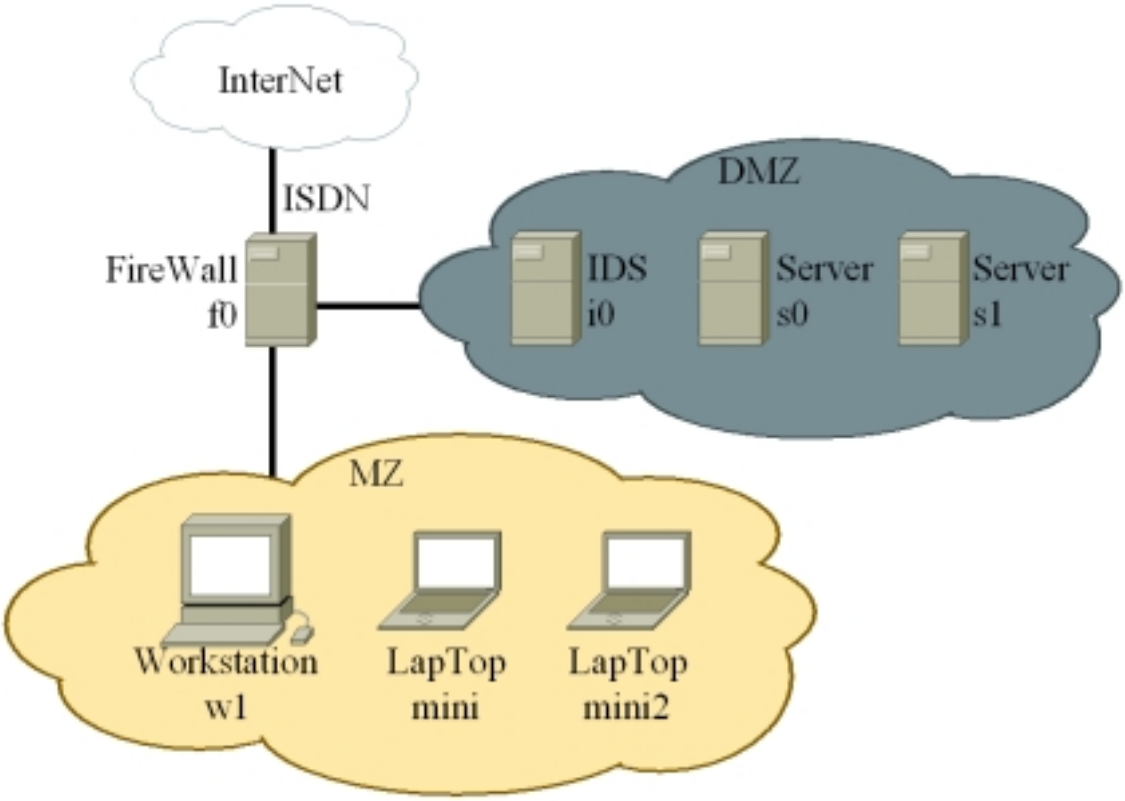

Ilustración 11- Entorn de treball

10.4 Descripció del Hardware

10.4.1 Servidors

· *FireWall – f0.ymbi.net/192.168.1.254 – Linux Slackware*

Pentium 90MHz

64Mb RAM

NE2000 ISA 10Mbps

ISDN Teles 16.3

HD 541Mb

· *Servidor num.0 – s0.ymbi.net/192.168.1.1 – Linux Slackware*

Pentium 120MHz

256Mb RAM

S3Virge 325 2Mb

Intel EtherExpress PRO/100

Audio CMI8330

HD 17Gb

HD 20Gb

· *Servidor num.1 – s1.ymbi.net/192.168.1.10 – Windows 2000 Server*

Pentium III 800MHz Celeron

256+256Mb RAM

AGP Trio3D/2x 8Mb

Intel EtherExpress PRO/100

SB Live

CD-R/RW 8x

CD 40x

FD 3'5"

HD 40Gb

· *IDS – i0.ymbi.net/192.168.1.200 – FreeBSD*

Pentium II 233MHz

128+128Mb RAM

S3 Virge DX/375 2Mb

PCI EtherNet 10/100

HD 8Gb

10.4.2 Workstations

· *Workstation num.1 – w1.ymbi.net/192.168.1.3 – Windows ME*

AMD K6-II 500MHz

256Mb RAM

PCI VGA 1Mb

Intel EtherExpress PRO/100

OnBoard SoundCard

HD 1'7Gb

· *Mini (Laptop) – mini.ymbi.net/192.168.1.100 – Windows 95+Linux Slackware*

Pentium 133MHz

32Mb RAM

HD 3Gb

FD 3’5”

PCMCIA 3C589C 10Mbps

· *Mini2 (Laptop) – mini2.ymbi.net/192.168.1.111 – Windows XP*

Pentium III 1’1GHz

256Mb RAM

HD 20Gb

FD 3’5”

DVD 8X

Ethernet Compaq 100Mpbs

Modem 56k

10.4.3 Connexió a Internet

1 Canal B ISDN a 64kbps connectat amb *Wandoo* 24h/dia. Aquestà connexió l'estàbleix el firewall gràcies a la targeta Teles 16.3. Gràcies a un 'watchdog' aquestà connexió es manté sempre activa. A més, a partir d'un servidor DDNS (Dynamic DNS) el sempre està localitzable des d'internet. Així doncs el firewall fa funcions de router i de firewall, ja que tot està integrat dins el mateix hardware.

10.4.4 Sistema d'alimentació ininterrompuda (SAI)

Tot el hardware està connectat a un sistema d'alimentació ininterrumpuda (SAI) que permet controlar les caigudes de corrent elèctric. Després d'una caiguda prolongada el servidor comunica a la restà d'ordinadors de la xarxa que s'ha interrumput el corrent elèctric i que s'han d'apagar. Això a nivell de servidor es fa de forma automàtica i les workstations només són advertides d'aquest fet.

10.5 Descripció del software

Com diu el seu nom un IDS és un sistema i no pas una única aplicació. Per tant, quan es decideix implementar un IDS no es pot instal·lar un programa i considerar que ja s'ha implementat un IDS.

10.5.1 Elecció del Sistema Operatiu

El sistema operatiu que suporta un IDS, ha de complir diferents requisits però els següents són indispensables i cal escollir el que ens garanteixi:

· Nivell de seguretat adient
· Estàbilitat
· Velocitat de procés, kernel petit i ràpid
· Transparència de funcionament
· Evitar càrregues innecessaries per l'IDS
· Controladors de les interficies de xarxa molt optimitzats
· Nivell de personalització elevat

10.5.2 FreeBSD vs Linux

A l'hora d'escollir el sistema operatiu d'aquest projecte el dubte estàva entre FreeBSD i Linux. Després de tenir en compte els punts de l'apartat anterior es va prendre la decisió d'escollir FreeBSD. Aquestà va ser una decisió molt meditada i estudiada. A continuació s'exposa un raonament entre les diferències i semblances entre aquests dos sistemes operatius *IX.

De fet, comparar Linux i BSD (Berkeley Software Distribution) no té massa sentit, perquè Linux és un SO molt jove en comparació a BSD. BSD fa més de dues dècades que està funcionant en els servidors de la universitat de Berkeley. En canvi Linux és va començar a gestàr a principis de la dècada dels 90. Per tant, BSD porta moltíssima avantatge i experiència acumulada.

Un clar exemple d'aquestà experiència és que fa més de quatre anys que no s'ha descobert un error de seguretat en el control d'accessos en una instal·lació estàndard de BSD. Per tant, moltíssims gurus consideren que BSD és el sistema operatiu més segur mai desenvolupat. BSD té tres distribucions OpenBSD, NetBSD i FreeBSD. Aquest últim bàsicament s'ha desenvolupat per fer la vida més fàcil a la gent que entra al món BSD. NetBSD és clarament un dels sistemes operatius que suporta més plataformes, actualment 70 diferents. Per la seva part, OpenBSD es considera la distribució més segura i seria del món BSD.

Està més que clar que BSD és molt millor com a sistema operatiu corporatiu que Linux. Malgrat la seva popularitat sigui inferior. Però ja se sap, no sempre el millor és el que més s'usa. Això si, només el millor és el que més s'ajusta a la solució d'un problema com el que es planteja.

10.5.3 Perquè NT/2000 són una mala opció

Per tothom és coneguda la infinitat d'errors de programació i de bugs trobats als sistemes operatius de Microsoft. A partir d'una interificie gràfica molt agradable per l'usuari i per una aparent simplicitat a l'hora de resoldre les necessitats de l'usuari s'amaguen infinitat de paràmetres que aquests sistemes no ens deixen control i sovint ni tan sols observar. Aquests motius ja fan suficient l'afirmació del títol de l'apartat.

Només la popularitat dels sistemes Windows i l'afany per vendre obliguen a les companyies a desenvolupar productes de seguretat, llegeixis IDS, per aquestes plataformes. O sigui, que és més una exigència del mercat el haver de programar IDS per aquestes arquitectures que no una opció intel·ligent. Només cal observar quines són les apostes de les grans companyies en materia de seguretat o fiabilitat dels seus sistemes servidors.

Darrera una aparent estreta col·laboració amb Microsoft sempre s'amaguen infinitat de solucions alternatives als sistemes Microsoft. Per exemple, la Nasa, IBM, l'Agència Espacial Europea, AENA, etc. confien per la seva seguretat en sistemes Linux, BSD, Solaris, o sistemes *IX en general. Fins hi tot empreses de la pròpia Microsoft com MSN usen pels seus servidors de DNS sistemes BSD, perquè segons els propis enginyers de l'empresa diuen que és més estàble que Windows 2000.

A més, només cal usar el sentit comú per adónar-se que un sistema que treballa amb una interficie gràfica tan pesada ha de perdre rendiment a l'hora de portar a terme tasques de servidor, enfront a sistemes marcats per l'entorn de caràcter. A més l'administrador només té accés al sistema a través d'una GUI que filtra tota la informació que aquest genera. Per tant, si es donen events no previstos aquestà GUI no és capaç de mostrar tota la

informació que es necessita per tractar aquestà situació excepcional. No oblidem els "pantallassos blaus" on sempre s'acaba dient que hi ha hagut un error del sistema i mai podem saber exactament perquè.

Una altre característica que el fa una mala opció per treballar com a servidor, és la necessitat de reiniciar tot el sistema cada cop que es fa algún canvi en la configuració principal del servidor, o fins hi tot, en el registra del sistema. Sovint els administradors estàn obligats a treballar en xarxes on la disponibilitat dels serveis és crítica i més en matèria de seguretat no es poden permetre el luxe de perdre el temps reiniciant les màquines cada dos per tres, perquè algún component del sistema ha tocat el registra o s'ha canviat la porta d'enllaç (gateway) d'alguna interfície de xarxa.

Es podrien seguir buscant arguments que raonen que mai s'hauria de montar un IDS sobre un Windows 2000 o NT. A menys que aquest sigui un HIDS i s'instal·li per suplir les mancances en seguretat que tenen aquests sistemes, degut a que no hi ha més remei que col·locar un servidor d'aquestà arquitectura dins la xarxa. Arribats en aquest punt, però, es considera suficient el raonament que demostra la inoperativitat d'aquests sistemes en materia de servidors o de seguretat.

10.5.4 Apache

En el IDS s'ha instal·lat també un servidor HTTP, per tal, de poder gestionar de forma remota els CMS que s'hi han instal·lat. A l'Annex també s'hi pot trobar el fitxer de configuració tot i que aquest té poques variacions sobre el fitxer estàndard, només s'ha modificat per treure totes les funcions de l'Apache que no seràn necessaries, augmentant així la seguretat del sistema i reduint el nombre de possibles bugs.

10.5.5 MySQL

Malgrat empreses com Yahoo! Finance, MP3.com, Motorola, NASA, Silicon Graphics, i Texas Instruments usen MySQL en certs projectes de les seves empreses.

MySQL simplement és el SBGD GPL més popular. No implementa moltíssimes funcions de l'estàdard SQL i això el fan un gestor de bases de dades poc potent per certs projectes. Però en aquest projecte ens hem vist obligats a usar-lo per problemes de compatibilitat en el software. Per tant, el fet de que sigui el SBGD més popular ha permés integrar tots els registres que es generen a la xarxa sobre seu.

10.5.6 PHP4

PHP és un pre-processador de pàgines Web en el costat del servidor. Això vol dir que permet programar pàgines web dinàmiques incrustant trossos de codi PHP enmig de les pàgines HTML situades al servidor. Aquest codi serà pre-processat pel servidor cada cop que un client faci una petició a una pàgina HTML del servidor. Retornant al client la sortida d'aquest codi, o sigui, codi HTML. D'aquestà forma el client rep un fitxer HTML el contingut del qual pot variar segons la sortida que hagi tingut la part de codi PHP del servidor.

Aquest paquet l'hem hagut d'instal·lar com a llibreria dinàmica de l'Apache perquè el ACID està programat amb aquest llenguatge d''scripting' i per tant, era necessari tenir-lo instal·lat.

10.5.7 Snort

Obviament el NIDS escollit per treballar en aquest projecte ha estat l'Snort, les propietats i característiques que el fan tan potent ja han estat comentades en l'apartat 9.2.1. Així doncs queda més que patent el perquè s'ha escollit. Aquí només descriurem quina informació rebra aquest Snort i quin tipus de patrons controlarà per cada interfície.

Degut a que com es descriu en l'entorn del projecte la xarxa on s'ha desenvolupat l'experiència està controlada per un switch. Això fa que els NIDS hagin estat instal·lats en cada servidor. S'han fet compilacions estatiques, sense depéndències de llibreries, de l'Snort 1.8.3 amb suport de bases de dades MySQL. Això vol dir que tots els servidors (f0,

s0, s1) tenen un Snort instal·lat connectat amb un sistema gestor de bases de dades, el MySQL. Instal·lat al IDS de la xarxa (i0). Allà es registren tots els events que es van dónant als diferents punts de la xarxa.

En total hi ha 5 sensors de NIDS generant events contra el SGBD. El firewall té dos sensors un per la interfícies connectada a internet (ippp0) i un altre per la interficie connectada a la intranet (eth0). El servidor Linux (s0) i el servidor 2000 (s1) també tenen un sensor cada un d'ells a la interficie de xarxa. Gràcies a tota aquestà informació el CMS instal·lat a i0 pot contrastar els paquets que deixa i que no deixa passar el firewall i quines modificacions pateixen aquests paquets.

Degut a que la intranet només té una IP pública, la de la interficie ISDN (ippp0) del firewall, la restà són privades (192.168.1.0/24), els serveis que donen els servidors: s0 i s1 a internet són a través de les redireccions de ports que fa el firewall (DNAT: Destination Network Address Translation). A causa d'aquest fet no és una tasca trivial analitzar les informacions que generen els sensors i per això és molt útil poder contrastar com es veu un mateix paquets des de diferents punts de la xarxa.

A l'Annex del projecte es poden trobar els fitxers de configuració de cada un dels NIDS instal·lats a la xarxa. Allà es poden observar els canvis de criteri a l'hora de configurar certs paràmetres dels pre-processadors de l'Snort segons on està instal·lat el sensor al que pertany cada fitxer de configuració.

10.5.8 ACID

Aquest CMS descrit a la secció 7.4.1, no necessita cap configuració especial. Gràcies a aquest software podem accedir de forma ràpida i amb filtres molt avançats per localitzar paquets o fins hi tot sessions capturades. El seu gran avantatge sobre el DEMARC i el que fa que s'hagi inclós en aquest projecte és la gran velocitat que té respecte el DEMARC, ja que el codi pre-processat en PHP és molt més ràpid que el pre-processat en Perl.

10.5.9 DEMARC

Aprofitant la seva estructura client servidor, s'ha instal·lat un client al servidor s0, per tal de controlar la integritat dels fitxers vitals pel sistema. A més de controlar els serveis principals que s'ofereixen des d'aquest servidor. Finalment també s'ha instal·lat un client que vigila els fitxers de sistema i el servei d'ssh (Secure Shell) al firewall. Tot això ho podem consultar des del servidor HTTP d'IDS (i0). Ja que com es descriu en la secció 7.4.2 el control, configuració i gestió d'aquestà aplicació es fa via Web.

En la següent captura de pantalla podem veure els serveis que està vigilant el DEMARC tan al servidor com al firewall:

10.5.10 PortSentry

És una aplicació que suporta qualsevol arquitectura *IX. Funciona com un dimoni del sistema i és capaç de dur a terme funcions de HIDS, sobre tot en materies referents als aspectes de la xarxa. És capaç de detectar i actuar enfront a escàners de ports ocults: SYN/half-open, FIN, NULL, X-MAS i qualsevol altre tipus d'escàner de ports. La propietat més important d'aquest software és el dinamisme que permet en la seva configuració de les mesures a prendre quan detecta activitat hostil en el host.

10.5.11 Logcheck

Aquestà aplicació que no s'havia fet referència a ella fins ara és molt senzilla. La seva potència resideix en aquestà simplicitat de funcionament. Bàsicament l'únic que fa és vigilar el fitxer de logs en busca d'events (comportament semblant a un HIDS). Quan aquests events es donen executa l'aplicació que nosaltres li diguem. Aquestes aplicacions les podem usar per generar alarmes, o per protegir-se d'aquestes anomalies parant el servei.

Així doncs convinant la potència del PortSentry i el Logcheck, aconseguim un HIDS força potent, però sobre tot molt configurable i ajustable a les nostres necessitats. A més ambdós suporten moltes plataformes:

- Linux®
- SunOS®
- Solaris®
- HPUX®
- Digital OSF/1®
- FreeBSD®
- BSDI®
- OpenBSD®
- NetBSD®

10.6 Exemples de funcionament

En aquest punt es pretenen mostrar un parell de casos pràctics. El primer cas es programa una signatura que generi un event quan algún usuari de la nostre xarxa pretengui connectar a un servei d'IRC. En el segon exemple es descriurà l'atac de 'CodeRed' que es va fer tan famós fa uns mesos i com l'Snort el detectava.

10.6.1 Exemple d'atac i detecció 1

10.6.1.1 Què volem controlar?

Hi ha una intranet (192.168.1.0/24) en la que els usuaris malgrat poden accedir als serveis d'IRC perquè el firewall no té filtrades aquestes connexions. L'administrador vol saber quan hi ha connexions cap aquests ports.

10.6.1.2 Patró per detectar la signatura

Quan un usuari obri una connexió cap a un servidor d'IRC aquest enviarà un paquet TCP com aquest, per exemple:

SYN 192.168.1.100 1025 -> servidor.irc.net 6667

Per tant, amb la informació que ens dóna aquest paquet ja podem escriure un patro de tràfic per l'Snort que ho detecti, per exemple:

alert tcp $HOME_NET any -> $EXTERNAL_NET 6667 (msg "INFO Possible IRC Access"; flags: S ; classtype: not-suspicious: sid: 542; rev:2;)

Amb aquest patró de tràfic, examinem tots els paquets que surten de la intranet cap a fora, el port origen pot ser qualsevol i el desti ha de ser el 6667. A més el paquet també ha de tenir el flag SYN actiu. La restà d'informació de la signatura podria variar, ja que només són codis estàndars per informar de la grabetat del problema trobat, el codi del problema, etc.

Tot i que la signatura anterior és correcta encara seria més correcta la següent:

alert tcp $HOME_NET any -> $EXTERNAL_NET 6666:7000 (msg:"INFO Possible IRC Access"; flags: A+; content: "NICK "; classtype:not-suspicious; sid:542; rev:2;)

Aquestà és una regla més genèrica i eficaç, ja que espera que l'usuari hagi començat a estàblir la connexió amb el servidor d'IRC. Per tal de llençar l'event. A més, no només mira les connexions al port 6667 sinó que comproba des del 6666 fins al 7000 que també són ports on hi pot haver un servidor d'IRC.

10.6.1.3 Acció i detecció

A continuació podem veure com un client de la intranet (192.168.1.10) intenta connectar-se a un servidor d'IRC:

```
01/07-18:55:55.412254 0:2:44:30:64:75 -> 0:40:5:49:12:EB type:0x800 len:0x3E
192.168.1.10:1772 -> 62.81.156.181:6667 TCP TTL:128 TOS:0x0 ID:21004 IpLen:20 DgmLen:48 DF
******S* Seq: 0xD88B7C68  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

01/07-18:55:58.353225 0:2:44:30:64:75 -> 0:40:5:49:12:EB type:0x800 len:0x3E
192.168.1.10:1772 -> 62.81.156.181:6667 TCP TTL:128 TOS:0x0 ID:21029 IpLen:20 DgmLen:48 DF
******S* Seq: 0xD88B7C68  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

01/07-18:55:58.759852 0:40:5:49:12:EB -> 0:2:44:30:64:75 type:0x800 len:0x3E
62.81.156.181:6667 -> 192.168.1.10:1772 TCP TTL:53 TOS:0x0 ID:0 IpLen:20 DgmLen:48 DF
***A**S* Seq: 0xC28B6D5D  Ack: 0xD88B7C69  Win: 0x16D0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

01/07-18:55:58.760352 0:2:44:30:64:75 -> 0:40:5:49:12:EB type:0x800 len:0x3C
192.168.1.10:1772 -> 62.81.156.181:6667 TCP TTL:128 TOS:0x0 ID:21038 IpLen:20 DgmLen:40 DF
***A**** Seq: 0xD88B7C69  Ack: 0xC28B6D5E  Win: 0x4470  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

01/07-18:55:58.761854 0:2:44:30:64:75 -> 0:40:5:49:12:EB type:0x800 len:0x40
192.168.1.10:1772 -> 62.81.156.181:6667 TCP TTL:128 TOS:0x0 ID:21039 IpLen:20 DgmLen:50 DF
***AP*** Seq: 0xD88B7C69  Ack: 0xC28B6D5E  Win: 0x4470  TcpLen: 20
NICK UriX.
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Obviament s'han omés totes les informacions innecessaries, perquè l'únic que es vol dennotar és la seqüència d'estàblient d'una sessió d'IRC, on queda clar que el pas d'assignació del mnemonic que s'usarà per conversar (Nick) es dóna en la forma que la signatura espera i que farà que es generi un event.

A continuació podem veure com el CMS de la xarxa rep l'alerta del sensor de l'Snort col·locat al firewall:

Meta

ID #	Time	Triggered Signature
2 - 204	2002-01-07 18:55:58	INFO Possible IRC Access

Sensor	name	interface	filter
	192.168.1.254	ippp0	*none*

Alert Group	*none*

IP

source addr	dest addr	Ver	Hdr Len	TOS	length	ID	flags	offset	TTL	chksum
192.168.1.10	62.81.156.181	4	5	16	65	61183	0	0	64	44791

FQDN	Source Name	Dest. Name
	s1.ymbi.net	pulsar1.iddeo.es

Options	*none*

TCP

source port	dest port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	seq #	ack	offset	res	window	urp	chksum
39814	6667				X	X				1181837482	1194003362	8	0	5840	0	11827

Options		code	length	data
	#1	NOP	0	
	#2	NOP	0	
	#3	TS	10	040089C8096C3F4C

Payload

```
length = 13

000 : 4E 49 43 4B 20 79 6F 75 6D 69 6E 0D 0A            NICK UriX..
```

Tabla 3 - Captura del ACID

Com que a l'hora de programar el patró de tràfic no s'ha informat de que aquest event pertanyi a cap repositori d'events suspitosos de tràfic, aquestà referència del CMS no té cap enllaç a pàgines amb més informació de l'acció.

10.6.1.4 Resum

S'acaba de demostrar com treballa un NIDS. Des de la localització del que es vol controlar fins a generar un patró de tràfic que detecta quan es dóna certa situació. Primer de tot s'ha d'estudiar la situació a localitzar, tipus de paquets que genera, si aquests es generen sempre i en quin ordre. Després es fa l'abstracció i es programa la regla que ha de coinsidir amb la signatura del paquet que corre per la xarxa. Un cop fet tot això només cal re-iniciar el NIDS i esperar a que es generin events d'aquest tipus perquè podem veure que tot funciona correctament.

Un consell a l'hora de fer aquest tipus de coses és llegir-se sempre el manual de l'aplicació i abans de programar cap regla, mirar bé les regles existens no fos cas que ja estés programat el que es vol fer, cosa que acostuma a passar.

10.6.2 Exemple d'atac i detecció 2

10.6.2.1 Què volem controlar?

Fa uns mesos hi va haver una infinitat de sistemes que varen estàr infectats pel cuc anomenat CodeRed. Això feia que els servidors webs no paressin de rebre peticions una rera l'altre buscant si el sistema era un vulnerable per tal de poder explotar la vulnerabilitat i seguir propagant-se per la xarxa. El que es vol controlar és quan es rep una petició d'aquest tipus, per poder informar l'administrador d'aquest fet.

10.6.2.2 Patró per detectar la signatura

Malgrat la complexitat del motiu pel qual els IIS pateixen l'error de seguretat anomentat 'CodeRed'. Explicació que es pot trobar, per exemple a: http://www.securityfocus.com/archive/1/191873. El que interessa saber és quina marca deixen sempre aquests atacs a nivell de xarxa. Cosa tan senzilla com fixar-se que la URI de la petició HTTP sempre conté la següent cadena: 'scripts/root.exe?'. Per tant, programar el patró serà tan senzill com:

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-IIS CodeRed v2 root.exe access"; flags: A+; uricontent:"scripts/root.exe?"; nocase; classtype:web-application-attack; sid: 1256; rev:2;)

10.6.2.3 Acció i detecció

En aquest exemple l'acció que desencadena l'event del NIDS s'escriurà a mà, ja que les dades registrades al CMS que a continuació s'adjunten són dades d'atacs reals que va detectar el NIDS. Per tant, la petició s'escriurà com si l'hagués fet el cuc que la va generà:

62.227.194.55:3251 -> 192.168.1.1:80 TCP TTL:53 TOS:0x0 ID:0 IpLen:20 DgmLen:48 DF
***A**P* Seq: 0xC28B6D5D Ack: 0xD88B7C69 Win: 0x16D0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
GET /scripts/root.exe?/c+dir HTTP/1.0

A continuació podem veure la resposta del NIDS vista des del CMS, aquest cas l'ACID.

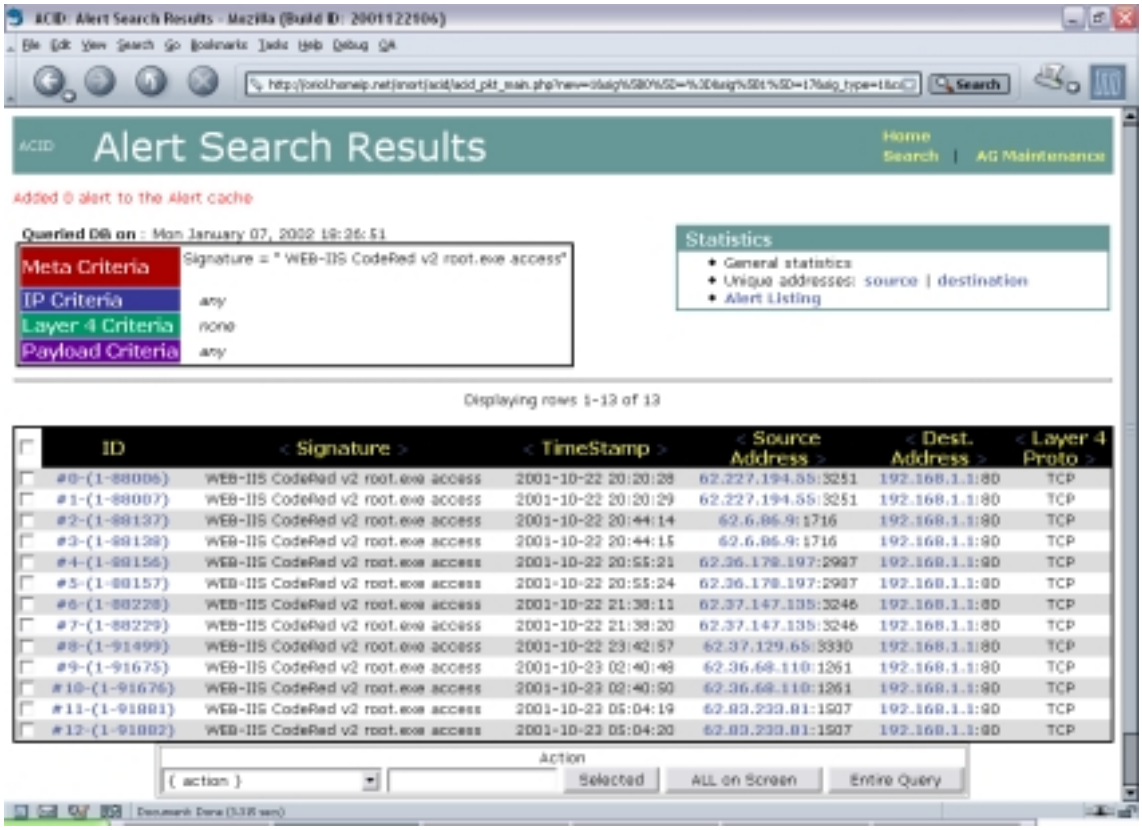

Ilustración 12 - Captura 1 del ACID

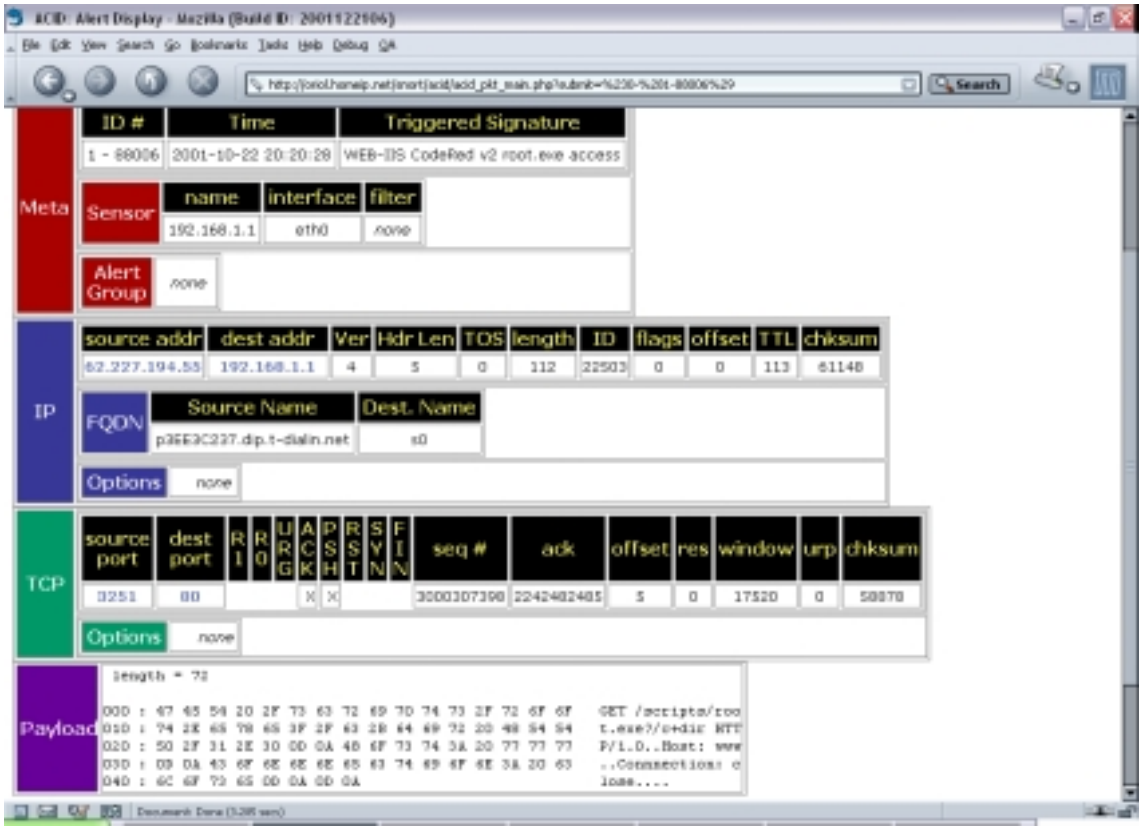

Ilustración 13 - Captura 2 del ACID

10.6.2.4 Resum

S’acaba d’escriure en aquest segon exemple un patró per un atac real que fins fa un parell de mesos portava de cap a tots els administradors, sobre tot els de sistemes IIS, ja que els sistemes Apache, o d’altres no eren vulnerables a aquest atac. Un *trick* que s’usava per tal d’adverti als adminstradors infectats per aquest cuc i que no eren conscients d’aquest fet era reprogramar el servidor HTTP perquè quan li fessin la petició anterior contestés executant un script que informés a l’administrador del sistema remot amb un missatge al seu monitor que informava de la seva vulnerabilitat.

10.7 Conclusions de la part pràctica

En aquest punt es pot observar com s'ha repetit infinitat de cops durant tota la memòria el que realment és important a l'hora de implementar, configurar i mantenir un IDS és l'experiència i la formació de l'administrador. En aquest punt queda palés que el coneixement dels atacs, protocols, tècniques, etc. dels intrusos o dels comportaments nocius pels nostres interessos no són en cap cas obvis i que necessiten una formació prèvia per tal de poder analitzar i treure les conclusions encertades a través de la informació generada.

Per altre banda, partint d'aquestà experiència i formació només cal imaginació i interés perquè els sistemes i xarxes comencin a tenir un nivell de seguretat acceptable. Malgrat la seva complexitat el tema de la seguretat arriba un moment que per pocs llocs ens pot arribar a sorprendre. Tots els atacs i comportaments acaben sent sempre més del mateix i l'únic que cal és estàr una mica al dia per tenir consciència de quins són els comportaments intrusius que més es porten a cada temporada.

11. Conclusions

Aquest projecte ha pretés donar una visió global i el més profunda possible dins el límit de l'espai. Sobre el món dels sistemes detectors d'intrusos (IDS). Per sobre de la qualitat de la informació aquí recopilada s'ha pretés mantenir una formalitat en la mateixa, per tal d'intentar deixar els conceptes el més clars possibles.

Es podria haver aprofundit molt més en els detalls de la implementació de l'IDS, però aquestà implementació només ha estat l'excusa per dónar la nota pràctica dins d'aquest projecte majoritariament teòric. S'ha cregut innecessari aprofundir en informacions que estàn més que documentades, com poden les comandes usades per instal·lar un dimoni, o quina és la millor opció per llençar certs programes.

Per sobre d'aquests aspectes concrets s'ha valorat molt més la comprensió i el coneixement formal dels IDS, vulnerabilitats, atacs, tipus d'intrusions, etc. que no el fet de si ha estat més o menys difícil posar en marxa aquestà o l'altre servei necessaris per implementar l'IDS. A més, s'ha cregut que des del punt de vista d'un administrador mitjanament experimentat no s'ha configurat res que no estigui directament documentat per cada producte usat.

Com a conclusió final només afegir que els IDS estàn en un mercat que cada dia està més en alça: El mercat de la seguretat. Cada cop augmenta més el número d'intrusions en les xarxes de les organitzacions; això fa que el pànic sigui cap cop major i que la necessitat de tenir més i millors elements de seguretat en les xarxes de les organitzacions sigui cada cop una necessitat més gran.

Així doncs, com sempre, el mercat haurà de respondre a aquestà demanda creixent d'elements de seguretat cada cop més sofisticats i efectius. Malgrat l'oferta avui en dia no és pas petita la qualitat dels IDS està encara molt qüestionada sobre tot per la seva joventut a nivell de codi. Però només el temps i l'esforç de tota la comunitat internet aconseguiran convertir aquestà societat de les TI en un lloc més segur.

Glossari

Backbone: cor de la xarxa d'una organització, acostuma a ser la part més ràpida.

Bug: error d'una aplicació.

Exploit: programa que explota un bug.

Frontend: part del software que interactua amb l'usuari.

GPL: GNU Public License, llicència molt popular, sota la qual es va llençar Linux. (http://www.gnu.org).

GUI: Graphics User Interface, interficie gràfica d'un programa amb la que interactua l'usuari.

iptables: comanda amb la que s'executen les ordres referents al filtratge de paquets en linux per NetFilter.

*IP Spoofing:*tècnica dels intrusos, que consisteix en falsejar la IP amb la que s'esta accedint a un servei.

ipfw: ordre de BSD amb la que s'estableixen les ordres de filtratge de paquets.

OpenSource: software que es distribueix amb el seu codi font.

Patch: pedaç al codi d'un programa que millora o corregeix el seu funcionament.

Port-bouncer: servei que es llença en una màquina, quan des d'una altre màquina ens connectem a una tercera a través del port de la primera, accedim amb la IP de la primera màquina. Per tant, aconseguim falcejar la nostre identitat.

Rootkit: conjunt de programes que permeten a un instrús explotar les vulnerabilitats d'un sistema i amagar la seva presència dins d'aquest.

*SGBD:*Sistema Gestor de Bases de Dades, grup de programes que permeten mantenir una Base de Dades tan de forma centralitzada com distribuida.

SQL: és el llenguatge estàndard de les bases de dades relacionals.

TI: Tecnologies de la Informació, sigles molt usades des de l'aparició de la famosa societat de la informació.

Trojan Horse: programa que falseja la seva activitat davant l'usuari, donant privilegis a un segon usuari sense que el primer ho sapigue.

Unicode: proporciona un número únic per cada caràcter, sense importar la plataforma, ni el programa ni l'idioma.

*Wanadoo:*ISP de la companyia Uni2.

Bibliografia

Localització URL:

[1] http://oriol.homeip.net, Oriol News Portal

[2] http://www.securityfocus.com, WhitePapers about IDS

[3] http://www.nsa.gov, WhitePapers

[4] http://www.nist.gov, NIST Publications

[5] http://www,snort.org, Snort Home Page

[6] http://www.demarc.org, DEMARC Home Page

[7] http://www.cert.org, CERT Home Page

[8] http://www.whitehats.com, WhiteHats IDS signatures repository

[9] http://cve.nist.gov, CVE repository

[10] http://www.sans.org, IDS FAQ

[11] http://www.linuxsecurity.com/docs/HOWTO/Snort-Statistics-HOWTO/index.html, Snort-Setup for Statistics HOWTO