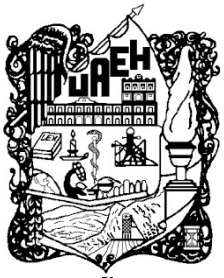


UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

"IMPLEMENTACION DE SISTEMAS DE DETECCION DE INTRUSOS EN AMBIENTES ESCOLARES"

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

PRESENTA:

ADOLFO IÑIGO ROSAS NIETO

ASESOR: M. EN C. GONZALO ALBERTO TORRES SAMPERIO

PACHUCA DE SOTO, HGO **JULIO DE 2014**

Dedicatorias

Cuando empecé este trabajo, existían ya casi diez años de diferencia contra lo que originalmente tenía pensado escribir en estas líneas pero siempre es importante entender el gran compromiso y colaboración que han tenido muchas personas no solo para este trabajo sino para toda mi carrera profesional.

Desde un inicio mi Madre María Corina Nieto Espinosa, con gran constancia y sacrificio, estuvo al tanto de que recibiera esta educación universitaria, pese a que ella solo tuvo la oportunidad de concluir una carrera técnica, en esta nuestra casa de estudio y que me acerca aún más a ella en un vínculo de hermanos de alma mater.

A mi padre Lauro Porfirio Rosas Franco por darme el consejo indicado, cuando así lo necesite, para dar cauce a mi ímpetu juvenil e impulsarme a concluir la carrera, también educarme en las virtudes de la humildad y la tolerancia.

A mis hermanos Aldo Ivan, Aline Ingrid, Arlette y Enrique por darme las dosis perfectas de entendimiento sobre criterios que pueden ser diferentes a los míos y también impulsarme a entender la importancia de que alguien de la familia tomara como profesión las Tecnologías de Información.

A mis tíos Antonieta, Arturo, Juana y Julieta quienes me brindaron su ayuda cuando estaba a punto de claudicar para seguir adelante, con los mejores gestos desinteresados que en ocasiones ni siquiera esperaba de ellos.

A mis hermanos de carrera Marco, Pilar y Franz por enseñarme el trabajo en equipo para vencer las adversidades no siempre las profesionales sino también las personales y darme el valor del estudio.

Y por último y no por eso menospreciar su sacrificio y ayuda, siendo estas las palabras que difieren de lo que pude escribir hace diez años que egrese de mi carrera Mi esposa Olga Neri Aguilar y mis hijas: Evelin Jazmín Rosas Neri y Addien Regina Rosas Neri quienes han sacrificado horas de calidad de familia, desvelos por tener alguien trabajando en el estudio y por los tramites que representa este trabajo.

Todas estas personas me han impulsado a ser mejor cada día y a no claudicar, a llevar la frente en alto al ser un hombre que a pesar de sus defectos puede alcanzar este logro y que puedo alcanzar muchos más.

No tengo palabras para expresar el agradecimiento de encontrar en mí, una persona digna de cambio y de virtud en vez de la piedra en bruto que muchos vieron y no quisieron labrar.

Hoy estas pequeñas líneas van dedicadas a ustedes y espero que puedan entender el sentir que en mi corazón emana por su ayuda y respeto.

GRACIAS

Adolfo Iñigo Rosas Nieto

Índice

Introducción

La seguridad Informática está vinculada a la historia de la computación desde sus inicios. La primera máquina que formalmente tiene la estructura de las computadoras modernas: COLOSUS, fue hecha para descifrar códigos alemanes durante la segunda guerra mundial. La justificación de los equipos de cómputo más modernos tienen fines militares o de inteligencia. Por tal motivo la computación puede deber, en gran parte sus avances a la ciberguerra que se libra desde que la humanidad entendió la importancia de la inteligencia en los conflictos.

Ante las amenazas que existen hoy en día, pequeños grupos pueden usar grandes acumulaciones de equipos de cómputo para crear supercomputadoras virtuales o bootnets que pueden desde hacer cálculos de descifrado de contraseñas y obtención de información de las empresas o de los gobiernos, hasta los famosos ataques de denegación de servicios. Las aplicaciones para estas son tan diversas como el ingenio de la humanidad misma.

Grandes corporaciones pueden utilizar herramientas de alto costo para llevar a cabo la seguridad informática y así librar a sus equipos de ser utilizados para estos fines además de mantener su información a salvo de estos grupos, dichas herramientas cuentan con respaldo de grandes grupos de ingenieros que monitorean estas herramientas y tienen respuesta muy ágil ante incidentes.

Las pequeñas corporaciones no cuentan con estos recursos y en algunos casos ni siquiera tienen un área o personal especializado en sistemas. Esto hace que surjan comunidades en internet que se dediquen a desarrollar herramientas para la seguridad informática y así buscar colaboración de los consumidores de su tecnología para abordar esta gran problemática.

El problema que plantean estas herramientas es que pueden ser muy complicadas para su implementación o incluso para su manejo. Requiriendo de personal especializado en el tema. Un problema más para los pequeños y medianos negocios. Pero que al mismo tiempo representa una gran oportunidad académica para el aprendizaje.

La implementación de tecnologías de seguridad informática basadas en redes en ambientes escolares es el motivo que me ha impulsado a escribir esta tesis y al mismo tiempo obtener el grado.

Espero que juntos podamos llevar a cabo esta aventura y que si eres un estudiante, cuentes con un punto de referencia para llevar tu carrera por el rumbo que tantas alegrías y éxitos me ha dado.

Planteamiento del problema

La seguridad informática plantea grandes retos tecnológicos, debido a que los grupos criminales que se dedican a estas actividades, generalmente tienen gran conocimiento y explotan vulnerabilidades que descubren a lo largo de investigaciones dedicadas.

Esto hace que las herramientas de seguridad informática, frecuentemente sean de altos costos y complejas para implementar, ya que requieren de equipos dedicados y especializados para este fin. Así una solución de seguridad informática requiere también de ingenieros especializados y dedicados a monitorear los incidentes existentes.

La información en la era moderna es uno de los capitales más valiosos y al mismo tiempo uno de los más difíciles de mantener al trasladarla a equipos de cómputo. Al mismo tiempo, los estudiantes representan dos problemáticas, una que es el que no vulneren los sistemas de la escuela o de algún otro lugar desde las instalaciones escolares y la segunda corresponde a protegerlos de las amenazas externas.

Las pequeñas empresas como las escuelas particulares, no cuentan con recursos para implementar herramientas de alta gama, por lo que las herramientas de software libre parecen ser las indicadas para estos temas.

La escuela en la cual se implementará la solución, cuenta solo con un responsable de sistemas el cual se encarga de actividades tan diversas en el la escuela como el hacer invitaciones para eventos especiales como el mantenimiento de equipos. Por lo que sus actividades no cuentan con recursos para poder abordar la problemática de la seguridad informática.

La cual plantea desde la carencia total de un plan de seguridad informática, pasando por redes que no tienen separación lógica o física, navegación insegura de maestros y estudiantes incluso carencia de herramientas de autenticación o diferenciación de usuarios, al mismo tiempo que se requiere de la solución de bajo costo.

Objetivos

El objetivo de este trabajo es dar a conocer un método practico además de replicable para poder implementar Sistemas de Detección de Intrusos en Ambientes Escolares que no cuentan con los recursos para hacer una gran implementación.

A través de los años de consultoría que he hecho a estos ambientes he notado la gran necesidad de establecer metodologías de seguridad. Dado que los alumnos suelen ser curiosos, el acompañarlos en el proceso de descubrimiento de sistemas y redes como Internet suele ser muy complejo, porque además de su curiosidad también dan sus primeros pasos en la intrusión de sistemas.

La detección de intrusos es un trabajo de expertos costosos que no siempre está al alcance de estos ambientes. Además de que muchas veces los estudiantes por desconocimiento o por obviar la seguridad dan pasos arriesgados en la madre de las redes.

Los estudiantes están en edades más tempranas sometidos a internet y pueden encontrar con un simple buscador como el mismo google imágenes o elementos tan fuertes para que pueden generar traumas de por vida o dejarlos al alcance del crimen organizado.

Es por estos y muchos objetivos que se torna importante el implementar un Sistema de Detección de Intrusos.

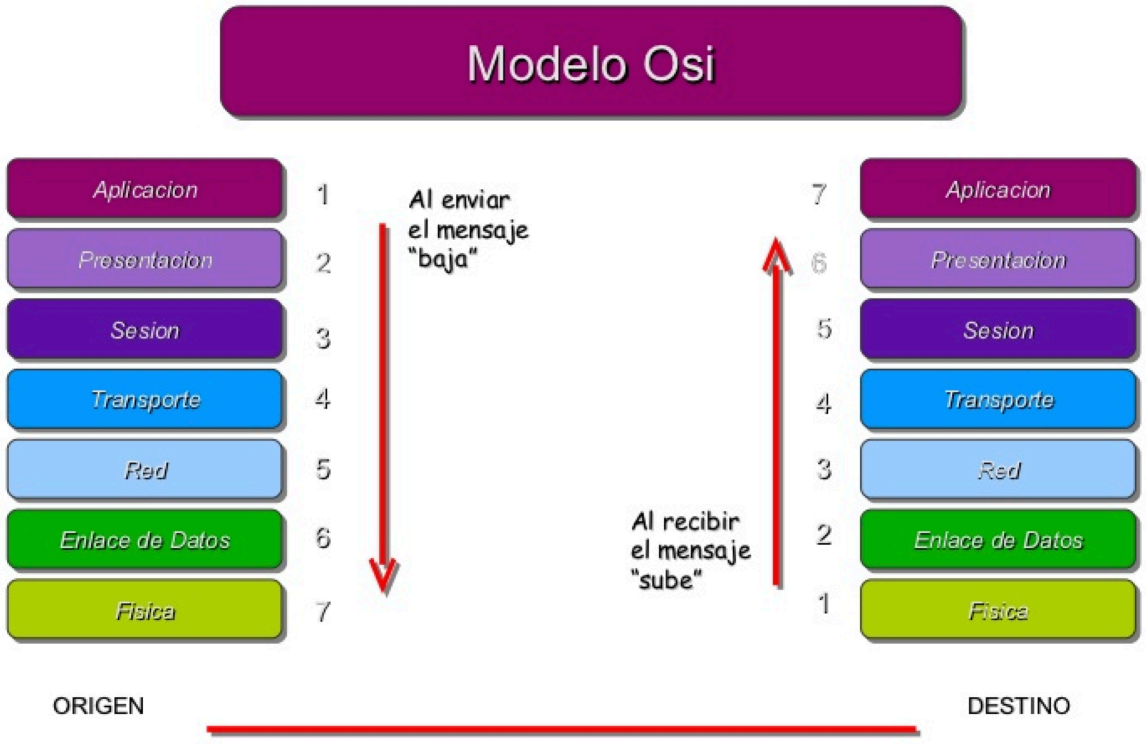
Capítulo 1

1. Fundamentos de Telecomunicaciones con el Modelo OSI.

La base fundamental para las telecomunicaciones modernas se basa en un modelo establecido en la Organización Internacional para la Estandarización. El modelo de referencia para la interconexión de sistemas abiertos (*Open System Interconection*) Tiene su origen en la Asociación de Estándares Internacionales "ISO" obedeciendo a la creciente necesidad de interconexión existente entre las compañías.

Así que en 1984 la Asociación Internacional de estándares Internacionales lanza la propuesta del Modelo OSI estructurándolo en 7 capas que hasta la fecha son utilizadas en la mayoría de los sistemas de telecomunicaciones modernos.

1. Capa Física.
2. Capa de Enlace
3. Capa de Red
4. Capa de Transporte
5. Capa de Servicios
6. Capa de Presentación
7. Capa de Servicios.



El funcionamiento del Modelo OSI es en envío y recepción como se representa en el diagrama. Cuando se envía el mensaje recorre las capas desde la superior hasta la inferior, y cuando se recibe, lo hace en sentido inverso recorriendo las capas desde abajo hacia arriba.

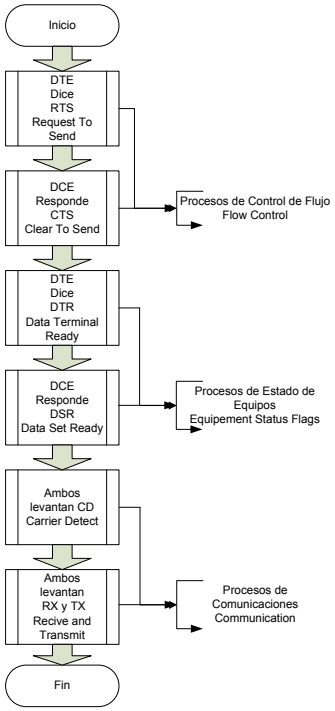
1.1. Capa Física.

La primera capa es la llamada Física (PHYSICAL), e incluye todo lo relacionado a los estándares de cableado como los medios físicos guiados: par trenzado, coaxial, fibra Óptica, etc., y elementos de tipo óptico o radio eléctrico como los medios no guiados: microondas, infrarrojo, láser, etc.

Esta capa se encarga de propiedades que representan la transmisión de los bits y como se establecerá la velocidad de la misma o si será de manera unidireccional o bidireccional (simplex, dúplex o full dúplex).
El análisis de esta resume el tipo de medio que será usado para la transmisión de datos, así también como los beneficios que representan en prestaciones cada medio.

1.1.1. Modelo Host a Host

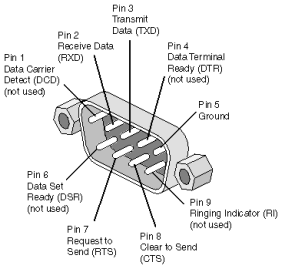
Este establece las conexiones entre dos equipos y es definido por estándares como el RS232 el cual norma que las comunicaciones que se establecen mediante una negociación entre ambas terminales las cuales pueden ser de dos tipos DCE (*Data Communication "Circuit - Terminating" Equipment*) y DTE (*Data Terminal Equipment*) así pues cuando realiza la negociación sigue el diagrama de flujo:



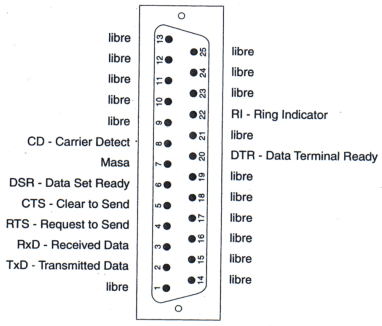
Antiguamente en un modem que es un elemento típico de conexión de esta capa, Una vez establecida la comunicación se podían apreciar la totalidad de las luces encendidas si existiera algún problema en el DTE las señales faltantes serian TXD, DTR y RTS en caso contrario si existiera una desconexión del DCE las señales faltantes serian RXD, DSR, CTS y CD

Así mismo este estándar obedece el realizar diversos conectores uno como son el DB-9 de nueve pines, el DB-25 que tiene 25 pines similar al puerto antes utilizado para las impresoras y finalmente el más utilizado para las telecomunicaciones en terminales de redes WAN el V.35

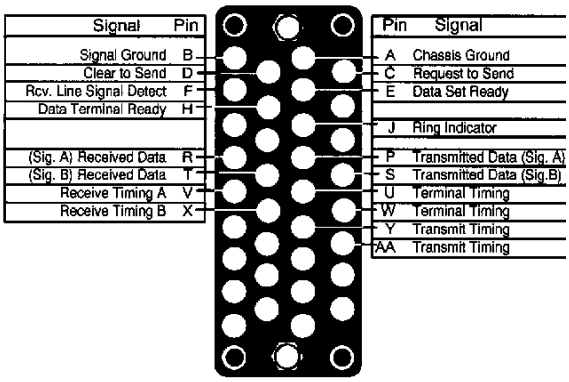
Conector DB-9



Conector DB-25

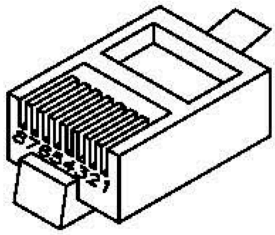


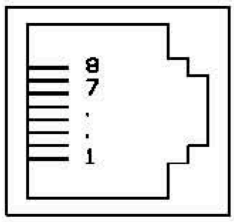
Conector V-35



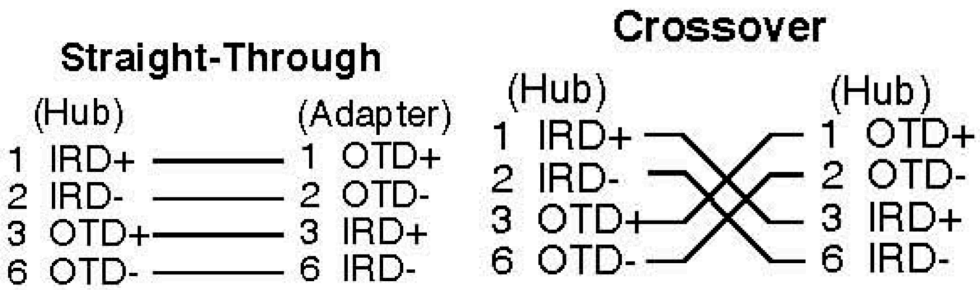
Modelo multi host

Este modelo tiene las conexiones más comunes hoy en las comunicaciones y es el que ocupa el protocolo entre otros Ethernet. Estos elementos son conexiones que se establecen de forma más lógica y el único elemento que examinamos en este será el conector más común el RJ-45 ya que el modelo multi host es administrado por capas superiores como la de red. Además de que este conector tiene la posibilidad de hacer conexiones Host a Host mediante configuración CrossOver.





Pin	MDI-X Assignment (Port 1-8)	MDI Assignment (Cascade Port 8)
1	Receive Data +	Transmit Data +
2	Receive data -	Transmit Data -
3	Transmit Data +	Receive Data +
6	Transmit Data -	Receive data -
4,5,7,8	Not Used	Not Used



1.2. Capa de Enlace

La segunda capa conocida como enlace (DATA LINK) establece los primeros estándares para la comunicación y la corrección de errores y colisiones.

1.2.1. Trama

Se entiende por trama a la estructura básica para el empaquetado de datos, teniendo como elementos básicos a tres: Cabecera, Datos y Control del Error.

Cabecera	Datos	Control de error

Dependiendo del método de transmisión la cabecera y el control de error pueden aumentar.

1.2.2. Detección de errores

La detección de errores es un elemento exclusivo de esta capa por lo que se conocen diferentes metodologías para hacerlo.

Al transmitir datos se tiene que realizar una inserción de elementos en la trama, así que al transmitir elementos, se tiene que la trama P se le suma un elemento Q siendo Q el número de elementos redundantes. A la transformación de esta trama P en Q se le llama detección de errores.

Existen diversos métodos para detectar errores los FEC (*Forward Error Control*) el cual tiene la facilidad de poder detectar errores y su corrección. Y los Feedback Error Control y Backward Error Control, los cuales solo permiten la detección de los errores por códigos de redundancia.

También se entiende por burst error a la cantidad de bits entre dos bits erróneos.

Paridad simple

Consiste en insertar un bit para indicar si la trama es par o impar contando el número de unos que se encuentran implícitos en la trama

Paridad longitudinal

En esta se introduce un carácter de paridad llamado BCC(*Block Check Character*) el cual se ensambla con la cantidad de bits pares longitudinales de cada trama.

CRC (Cyclic Redundancy Check)

Esta metodología permite la detección de más de un bit erróneo y se basa en uso de una división a la trama y la inserción del resultado en la misma, obedeciendo a una función matemática definida.

1.2.3. Dirección MAC

Además de las detecciones de errores en esta capa se definen elementos de señalización básicos, tal es el caso de la dirección de control de acceso medio: *"Media Access Control Address"* o comúnmente conocida como dirección MAC, la cual se compone por 48 bits, de la cual, los primeros 24 son definidos para el fabricante y el resto son para la identificación del dispositivo que se denomina NIC (*Network Inreface Card* o tarjeta de interfaz de red)

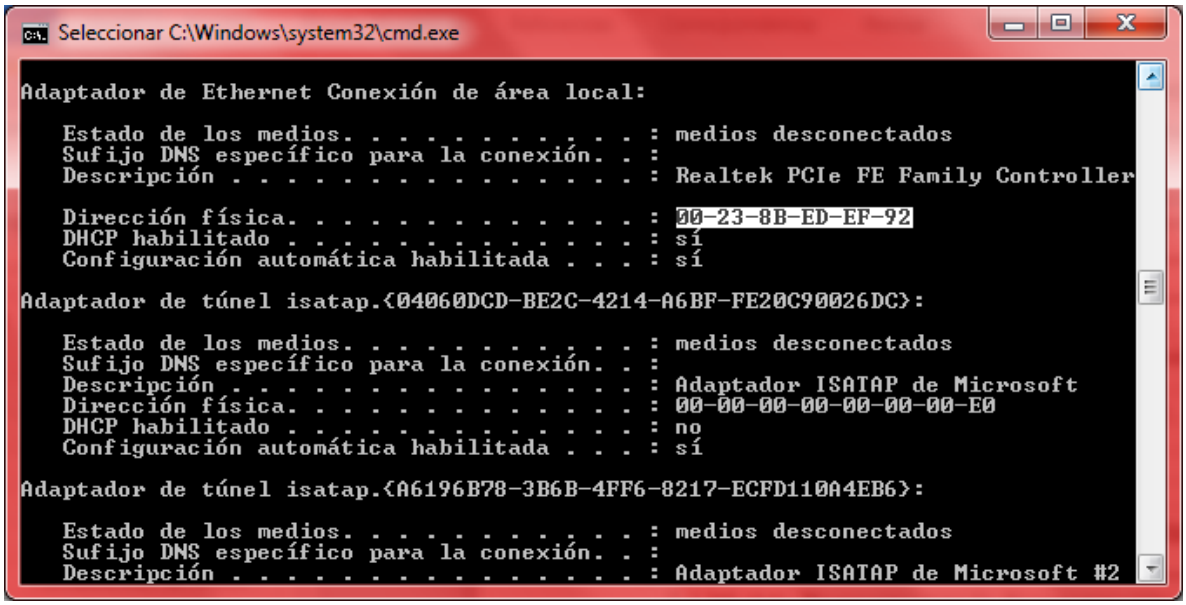
La lista de fabricantes y la relación de los números usados es regulada por el IEEE y la podemos encontrar en su página Web http://standars.ieee.org/regauth/oui/oui.txt

Un ejemplo clásico de una dirección MAC lo pueden encontrar en su computadora. Al teclear el comando "ipconfig" desde una ventana de dos en Windows o bien en sistemas basados Unix desde el Shell con el comando "ifconfig". Si tienen alguna interfaz de red configurada esta presentara su Dirección MAC.

La computador de Windows que utilice para este ejemplo tiene la dirección MAC:

00:23:8B:ED:EF:92

Esta la presenta en un juego de números hexadecimales tras teclear el comando ipconfig el resultado fue algo como lo siguiente:



Así en base a la lista que tiene publicada el IEEE los datos del fabricante corresponden a:

```
00-23-8B    (hex)              Quanta Computer Inc.
00238B      (base 16)          Quanta Computer Inc.
                               NO. 211, WEN HWA 2RD.,KUEI SHAN HSIANG,
TAO YUAN SHIEN,
                               TAIPEI TAIWAN 333
                               TAIWAN, REPUBLIC OF CHINA
```

Elementos de conexión pueden almacenar las direcciones en una tabla para la transferencia de datos entre equipos y a su vez la comunicación de esta hacia capas superiores del modelo OSI

1.2.4. Otros protocolos de la capa 2

Además de la detección de errores esta capa se caracteriza por definir la conexión entre dos o más elementos de la red WAN y esto se logra a través de protocolos como *Frame-Relay*, HDLC y ATM entre otros.

También protocolos de conexión de las VLAN (*Virtual Local Access Network*), trabajan sobre esta capa, tales como el 802.1Q de la IEEE o el ISL (*Inter Switch Link*) propietario de Cisco

1.3. Capa de Red

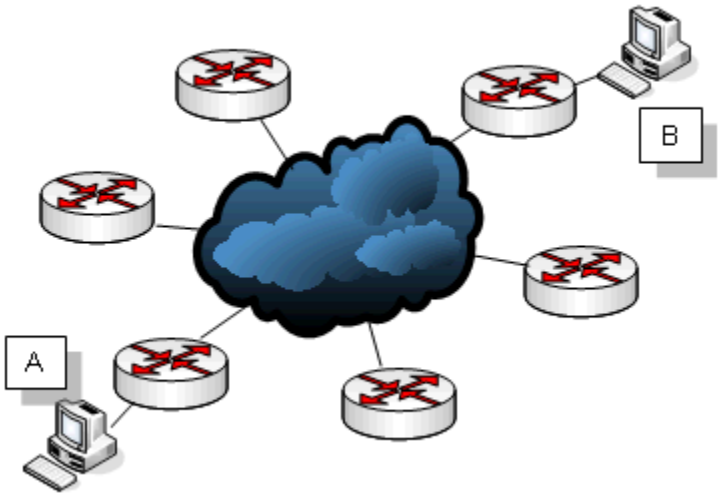
En esta capa se definen los elementos que hacen que los datos lleguen de un punto a otro. En esta capa se encuentran elementos para la interconexión de varios dispositivos. Aquí aparecen controles de congestión de la red.

En esta se definen conceptos tales como su unidad de gestión, el paquete. También esta capa administra el sistema de direccionamiento, destacando el popular IP (*Internet Protocol* o protocolo de internet)

1.3.1. Importancia de la Capa de Red

Como ya lo mencionamos en esta capa se gestionan las conexiones. A diferencia de la capa 2 en la que puede haber una comunicación entre únicamente dos dispositivos, aquí podemos solucionar problemas que obedecen a el crecimiento en las redes, que requieren que los elementos de la red se conectaran con más dispositivos.

Supongamos que tenemos una rede en la que se tienen los dispositivos dispuestos como en la ilustración siguiente



Para que el dispositivo A se comunique con el dispositivo B es necesario establecer una ruta entre ambos elementos a través de los dispositivos que los separan, mediante un esquema definido.

1.3.2. Elementos de comunicación de la capa de Red.

Los elementos que trabajan en esta capa tienen la capacidad de encontrar rutas. En esta capa se toman unidades de direccionamiento de las direcciones MAC, que vimos hace unos momentos y se traduce a protocoles superiores como el popular sistema de direccionamiento IP o Internet Protocol.

En tanto la dirección MAC es asignada por el fabricante de la tarjeta o dispositivo de red que estamos usando la dirección IP es asignada por el administrador de red.

Un equipo que trabaja en esta capa es el Router. Este se ocupa de llevar tráfico entre redes basándose en los protocolos de esta capa. Y es también capaz de encontrar una ruta en la red de manera "inteligente" usando protocolos que evalúan el costo o tiempo de respuesta entre enlaces.

Actualmente también algunos Switches, para no decir que la gran mayoría de las marcas comerciales, actúan sobre esta capa administrando conexiones entre redes virtuales VLAN que se usan para administrar el tráfico de redes corporativas.

Algunas de las marcas más reconocidas de estos elementos son: El líder de mercado mundial CISCO, HP, Extreme Networks, 3Com, Verso Technogies Inc. Entre otros.

Línea Cisco

1.3.3. Segmentación de la capa de red.

Al hablar de redes se torna importante que establecer secciones definidas para un control de las mismas. Es por esto que nos auxiliamos de las segmentaciones, que pueden ayudarnos a separar redes a medida que pueden ayudarnos a una mejor administración.

Las segmentaciones que se utilizan, comúnmente establecen las redes LAN (red de acceso local) MAN (rede de acceso medio) WAN (rede de acceso amplio) y reciente mente algunos autores y fabricantes han agregado el concepto de red de acceso inalámbrico mejor conocida como WLAN que es una variante de la red LAN.

Internet se puede interpretar como un conjunto de segmentos de red unidos entre si. Que es administrado por proveedores. A estos se les puede denominar proveedores de servicios de internet o ISP(*Internet Service Provider*).

1.3.4. Protocolo de Internet TCP/IP

Las siglas TCP quieren decir *Transmision Control Protocol* y las siglas IP significan *Internet Protocol* y esta específicamente limitado a proporcionar las funciones necesarias para enviar un paquete desde un origen a un destino. En un sistema de redes bajo el esquema de mejor esfuerzo, este solo entrega los paquetes de la mejor manera posible y en un esquema completo lo hace con reenvío de paquetes.

El protocolo brinda dos funciones básicas direccionamiento y fragmentación y fue desarrollado por el DARPA *Defence Advanced Research Projects* Agency o en otras palabras el departamento de investigación avanzada de defensa norteamericano

En el contexto de redes el protocolo tiene aplicaciones tan variadas como para poder establecer diversos segmentos en una red LAN y convivir con un protocolo host a host y hacerlo tan extenso como para que haga interactuar las unidades completas de negocios de un corporativo mundial. Mediante el uso de puertas de enlace o Gateway.

El protocolo se basa en el uso de diversos elementos que brindan identificación a cada paquete. Etiquetado por él, para que no se dupliquen o se pierdan en la red.

Está conformado por 4 bytes o 32 bits, y muy comúnmente se representa utilizando su estructura en una división de 4 segmentos: A.B.C.D donde cada segmento puede tomar valores de 0 a 255. Un ejemplo práctico puede ser la dirección IP que soporta la página de la universidad 200.34.44.229.

Para entenderlo mejor podemos decir que puede representarse en valores de números hexadecimales desde 00.00.00.00 hasta FF.FF.FF.FF o bien en binarios desde 0000000.0000000.0000000.0000000 hasta 11111111.11111111.11111111.11111111. Así pues la dirección IP de la página de la universidad seria: en hexadecimal C8.22.2C.E5 o en binario 11001000.00100010.00101100.11100101.

Tomando en cuenta esta representación el protocolo puede tomar valores de 2 elevado a la 32 es decir alrededor de 400 millones de valores únicos.

Hay que tomar en cuenta que no todas las direcciones del rango pueden ser asignadas y generalmente cada dirección está conectada a una red. Así mismo, cada dirección dentro de este segmento empieza con el mismo número. Es por esto que el protocolo conceptualmente tiene un segmento dedicado a identificar la red y otro para identificar al host.

En función del número de host que se ocupen en cada red, se dividen en clases primarias:

- Red de Clase A
- Red de Clase B
- Red de Clase C
- Red de Clase D
- Red de Clase E

Red clase A

Soporta la mayor cantidad de host. Es un identificador de red de 7 bits y el resto son direcciones de host esto hace que soporte hasta 126 redes. En esta se puede clasificar una red corporativa mundial o bien los segmentos públicos de internet. Su trama se divide del siguiente modo.

Primer octeto	Segundo Octeto	Tercer Octeto	Cuarto Octeto
0 + Red	Host	Host	Host

Red clase B

Conformado un segmento de red de 14 bits es un rango medio soporta hasta 16,384 redes, para ser aplicadas en una red que puede abarcar un país o bien un campus universitario.

Primer octeto	Segundo Octeto	Tercer Octeto	Cuarto Octeto
10 + Red	Red	Host	Host

Red clase C

Tienen un identificador de red de 21 bits y soporta hasta 2,097,152 redes está clasificado para pequeños segmentos laboratorios de computo pequeños o sucursales empresariales.

Primer octeto	Segundo Octeto	Tercer Octeto	Cuarto Octeto
110 + Red	Red	Red	Host

Red clase D

Son direcciones reservadas para la Multidifusión (multicast)

Primer octeto	Segundo Octeto	Tercer Octeto	Cuarto Octeto
1110			

Red Clase E

Estas son direcciones reservadas no pueden ser utilizadas.

Primer octeto	Segundo Octeto	Tercer Octeto	Cuarto Octeto
1111			

Existe un registro de ip que es la RFC-1166 en cual se establece a quien pertenece el rango de IP.

Segmentación

De este modo este protocolo puede establecerse en todos los elementos de una red activa. Para esto necesitara segmentaciones de la red. Esto se hace estableciendo una mascar de subred.

La máscara de subred es una es una dirección que establece la delimitación de la red. Por ejemplo las máscaras de subred para clases de redes pueden establecerse con la siguiente tabla.

Clase de red	Máscara
A	255.0.0.0
B	255.255.0.0

C	255.255.255.0

Como hemos estudiado el protocolo hasta el momento tenemos que la máscara de red para la Clase A se expresaría del siguiente modo:

Primer octeto	Segundo Octeto	Tercer Octeto	Cuarto Octeto
11111111	00000000	00000000	00000000
Red	Host	Host	Host

De este modo, si tenemos una dirección establecida 127.30.50.60 y está acompañada por la máscara 255.0.0.0 podemos decir que esta dirección pertenece a la red 127.0.0.0, o bien si esta tuviera una máscara diferente nos podría denotar que pertenece a otra clase de red.

Por ejemplo si esta misma dirección estuviera acompañada por una máscara 255.255.0.0 nos indicaría que pertenece a la red 127.30.0.0 y que sería una red de clase B.

El mismo ejemplo para una red clase C requeriría de una máscara de subred 255.255.255.0 lo cual denotaría que la red es 127.30.50.0.

Tomando esta notación podemos incluso segmentar más la asignación de redes. Llegando a dividirla en segmentos más pequeños al jugar con los valores del último octeto. Esto nos puede ser útil para el manejo de pequeñas redes y ruteo de enlaces.

Podemos ilustrar la segmentación de redes con la siguiente tabla.

Macara de subred	Binario	Número subredes	Host por subred	Denotación de red
255.255.255.0	00000000	1	254	x.x.x.0
255.255.255.128	10000000	2	126	x.x.x.0, x.x.x.128
255.255.255.192	11000000	4	62	x.x.x.0, x.x.x.64, x.x.x.128, x.x.x.192
255.255.255.224	11100000	8	30	x.x.x.0, x.x.x.32, x.x.x.64, x.x.x.96, x.x.x.128, x.x.x.160, x.x.x.192, etc.
255.255.255.240	11110000	16	14	x.x.x.0, x.x.x.16, x.x.x.32, etc.
255.255.255.248	11111000	32	6	x.x.x.0, x.x.x.8, x.x.x.16,etc.
255.255.255.252	11111100	64	2	x.x.x.0, x.x.x.4, x.x.x.8, x.x.x.12, etc.
255.255.255.254	11111110	-	-	No es posible
255.255.255.255	11111111	-	-	No es posible

Aun que las 2 últimas no ilustran una segmentación posible son utilizadas en tablas de ruteo para la unión de redes.

La unión de redes también necesita puertas de enlace (Gateway). Esto puede ser hecho por un router o un servidor dedicado a ello.

De tal modo que un host tiene tres valores básicos indicados para poder establecer una comunicación en redes mediante el protocolo IP.

IP Host
Mascara de subred
Gateway

1.4. Capa de Transporte

Esta capa supone la definición lógica de la información que viaja a través de la red para aplicaciones punto a punto y punto multipunto.

Establece y mantiene circuitos virtuales entre los equipos. Proporciona confiabilidad en la entrega de paquetes o su reenvío en caso de errores de tráfico.

Para estas operaciones la capa de transporte divide la información en elementos llamados paquetes. Estos pueden ser de tipo UDP (*User Datagram Protocol*) Protocolo de datagrama usuario y TCP (*Transmision Control Protoco*l) Protocolo de control de transmisión.

La principal diferencia entre ambos protocolos es la entrega de los paquetes. Mientras que TCP se encarga de comprobar la información recibida y solicita retransmisiones en caso de errores. UDP es un protocolo de mejor esfuerzo. Esto es que solo entrega los paquetes a destino sin pedir retransmisiones.

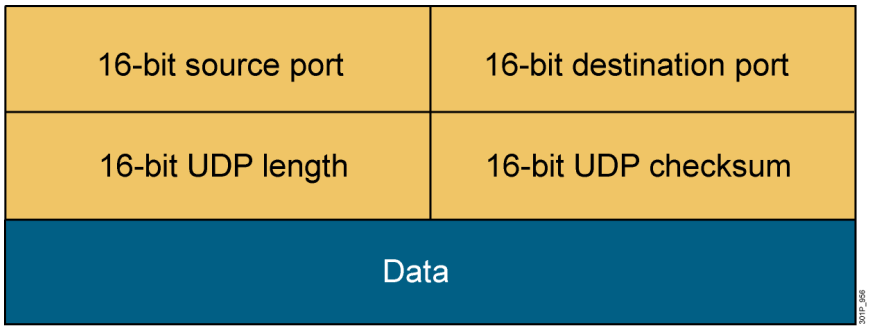
1.4.1. Características de UDP

Opera en la capa 4 del modelo OSI. Proporcionando la conectividad a las aplicaciones sin los retrasos de los protocolos basados en corrección de errores, no establece circuitos virtuales entre equipos, posee un mecanismo simple de detección de errores, proporciona una conectividad de mejor esfuerzo,

Este tipo de paquete proporciona la conexión para aplicaciones que por su naturaleza no requieren de comprobar la llegad de paquetes. El caso más típico para este son las aplicaciones de VoIP (*Voice Over IP*), Voz Sobre IP las cuales por ser aplicaciones en tiempo real no toleran un retraso en la comprobación de la llegada del paquete. Ya que este proceso puede causar que la voz al pasar por el medio de comunicación tenga retrasos o distorsiones.

Otro caso puede referir a los protocolos dedicados a la transmisión de video. Como el paquete no establece ningún estado de conexión, un servidor puede soportar más conexiones y así evitar retardos en los procesos de propios de la compresión y descompresión del video. Además de aplicaciones de multidifusión de paquetes como los servicios de nombres de dominio y los servicios de tiempo en la red o NTP

El paquete UDP maneja un cabecero muy sencillo que comprende 16 bits para describir el puerto origen, 16 bits para el puerto destino, 16 bits para describir la longitud del paquete y 16 bits para el checksum o comprobador de error.

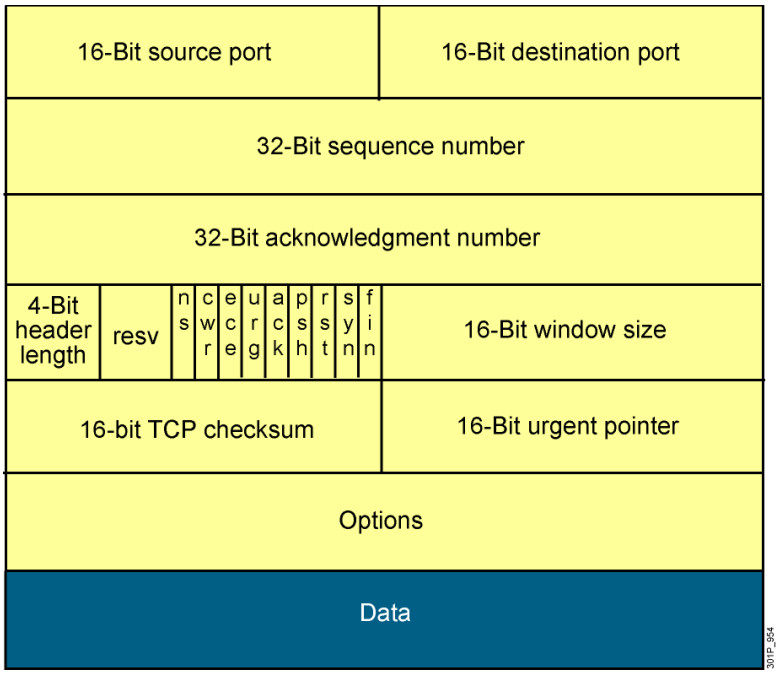


1.4.2. Características TCP

Este provee la capa de transporte para las aplicaciones basadas en el mecanismo TCP/IP. Proporciona acceso a la red para capas superiores. Protocolo orientado conexión. Es de doble vía o "*Full Duplex*". Comprueba errores y solicita retransmisiones en caso de error. Proporciona secuencia a los paquetes.

Este es el paquete más comúnmente usado para los modelos de comunicación actual. Entre los más comunes aparecen el e-mail, la compartición y descarga de archivos, entre otros. Este proporciona características importantes de conexión para protocolos basados en fiabilidad en las conexiones. Tal es el caso de HTTP que viaja a través del puerto 80.

La composición de este paquete es más compleja que la del UDP que solo contiene origen destino, longitud y Checksum. Este tiene los mismos 16 bits de origen, 16 bits de destino, 32 bits de número de secuencia que asegura la correcta secuenciación de la información, 32 bits de Acknowlegment, 4 bits de cabecero, 9 bits de banderas de control, 16 bits de ventana o de octetos, 16 bits para marcar el paquete como urgente y por su puesto la información que será en este caso de longitud variable.



1.4.3. El modelo de negociación en tres pasos

El modelo de negociación de tres pasoso o *Three Way Hanshake* es el modelo en el cual las comunicaciones entre dos dispositivos se llevan a cabo a través de una red. Generalmente un equipo inicia la comunicación mientras que el otro solo escucha comunicaciones. El modelo se basa en el maneo de las banderas.

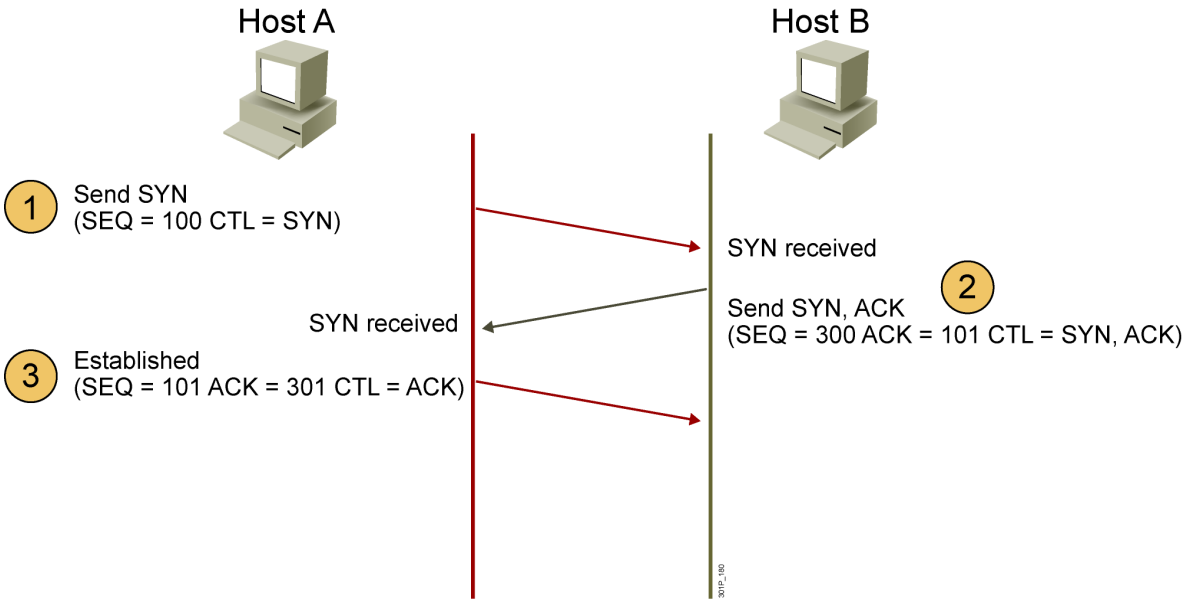
El modelo está implícito en las comunicaciones por TCP y conforma una serie de pasos para poder llevar a cabo la conexión.

Funciona del siguiente modo:

1. El equipo que busca la conexión envía un paquete de sincronización al equipo receptor con lo que inician las comunicaciones. Este utiliza la bandera SYN habilitada y contiene el segmento de sincronización y un valor para el uso del reconocimiento (*acknowledgment*) que se llama ISN

2. El equipo receptor recibe el paquete SYN y en caso de que este recibiendo información por ese puerto fija los parámetros del ACK para que negocie la conexión y reconozca el recipiente de sincronización del emisor. Esto lo hace indicando el siguiente número de secuencia del emisor que envió el paquete ISN y el receptor lo incrementa en uno. En caso que el equipo no esté recibiendo conexiones por este puerto contestara con un segmento que se llama RST con esto indica que ha rechazado la conexión.

3. El equipo que busca la conexión reconoce el segmento de sincronización del equipo receptor y la comienza el envío de información. Se realiza cuando el bit que corresponde a SYN está sin fijar en el cabecero TCP y confirma que el proceso de tres pasos ha terminado.



1.5. Capa de transporte

Esta capa interpreta las integraciones de la capa de red, su unidad se llama datagrama. Dependiendo del tipo de tráfico generado por la capa de red se designan los Sockets en él.

La construcción de los datagramas es utilizando los cabeceros de las tramas de la capa de red si nos damos cuenta es tenemos un margen de 16 bits para ambos casos como puerto origen y puerto destino.

Si hacemos la operación matemática obtenemos que solo pueden existir una cantidad limitada de puertos

$2^{16} = 65536$

La designación de los puertos dependerá de la aplicación que está comunicando el datagrama, para normalizar se hicieron intentos por tomar estos y se generó una lista que se conoce como "*Well Known Ports*" reservando los puertos del 1 al 1024. A continuación se muestra una tabla con una muestra de los primeros 100 puertos comúnmente usados:

Port	Transport	Protocol
0	TCP	Shirt Pocket netTunes. Shirt Pocket launchTunes.
1	TCP	TCPMUX, TCP Port Service Multiplexer.
2	TCP, UDP	Management Utility.
3	TCP, UDP	Compression Process.
4		
5	TCP, UDP	Remote Job Entry.
6		
7	TCP, UDP	Echo.
8		
9	SCTP, TCP, UDP	Discard.
10		
11	TCP, UDP	SYSTAT.
12		
13	TCP, UDP	Daytime.
14		
15		
16		
17	TCP, UDP	Quote, Quote of the Day.
18	TCP, UDP	RWP, Remote Write Protocol. Send, Message Send Protocol.
19	TCP, UDP	Chargen, Character Generator Protocol.
20	TCP	FTP, File Transfer Protocol, data.
21	TCP	FTP, File Transfer Protocol, control.
22	SCTP, TCP	SSH.
23	TCP	Telnet.
24		Any private mail system.
25	TCP	SMTP, Simple Mail Transfer Protocol.

Port	Transport	Protocol
26		
27	TCP, UDP	NSW User System FE.
28		
29	TCP, UDP	MSG ICP.
30		
31	TCP, UDP	MSG Authentication.
32		
33	TCP, UDP	Display Support Protocol.
34		
35		Any private printer server.
36		
37	TCP, UDP	Time, Time Protocol.
38	TCP, UDP	RAP, Internet Route Access Protocol.
39	UDP	RLP, Resource Location Protocol.
40		
41	TCP, UDP	Graphics.
42	UDP	Internet Name Server.
43	TCP	Whois.
44	TCP, UDP	MPM FLAGS Protocol.
45	TCP	Internet Message Protocol.
46	TCP, UDP	MPM [default send].
47	TCP, UDP	NI FTP.
48	TCP, UDP	Digital Audit Daemon.
49	TCP	TACACS+.
	UDP	TACACS.
50	UDP	RMCP, Remote Mail Checking Protocol.
51	TCP, UDP	IMP Logical Address Maintenance.
52	TCP, UDP	XNS Time Protocol.
53	TCP, UDP	DNS, Domain Name System.
54	TCP, UDP	XNS Clearinghouse.
55	TCP, UDP	ISI Graphics Language.
56	TCP, UDP	XNS Authentication.
57	TCP	MTP, Mail Transfer Protocol.
58	TCP, UDP	XNS Mail.
59	TCP	NFILE.
60		
61	TCP, UDP	NI MAIL.
62	TCP, UDP	ACA Services.
63	TCP, UDP	Whois++.
64	TCP, UDP	CI, Communications Integrator.

Port	Transport	Protocol
65	TCP, UDP	TACACS-Database Service.
66	TCP, UDP	Oracle SQL*NET.
67	UDP	BOOTP, Bootstrap Protocol, server.
68	UDP	BOOTP, Bootstrap Protocol, client.
69	UDP	TFTP, Trivial File Transfer Protocol.
70	TCP	Gopher.
71	TCP, UDP	Remote Job Service.
-		
74		
75		Any private dial out service.
76	TCP, UDP	Distributed External Object Store.
77		Any private RJE service.
78	TCP, UDP	vettcp.
79	TCP	Finger.
80	TCP	HTTP, HyperText Transfer Protocol.
81		
82	TCP, UDP	XFER Utility.
83	TCP, UDP	MIT ML Device.
84	TCP, UDP	Common Trace Facility.
85	TCP, UDP	MIT ML Device.
86	TCP, UDP	Micro Focus Cobol.
87		Any private terminal link.
88	UDP	Kerberos.
89	TCP, UDP	SU/MIT Telnet Gateway.
90	TCP, UDP	Used unofficially by Pointcast. DNSIX Securit Attribute Token Map.
91	TCP, UDP	MIT Dover Spooler.
92	TCP, UDP	Network Printing Protocol.
93	TCP, UDP	Device Control Protocol.
94	TCP, UDP	Tivoli Object Dispatcher.
95		SUPDUP.
96	TCP, UDP	DIXIE.
97	TCP, UDP	Swift Remote Virtual File Protocol.
98	TCP, UDP	TAC News.
99	TCP, UDP	Metagram Relay.

Aunque formalmente estos puertos no son reservados se tiene por convención el no usarlos más que para las aplicaciones necesarias.

El resto de los puertos se conocen como puertos altos los que se ocupan formalmente para la comunicación.

La asignación de estos puertos dependerá del controlador de la tarjeta de red o del sistema operativo del host y de esto se encargará la capa de sesión.

1.6. Capa Sesión.

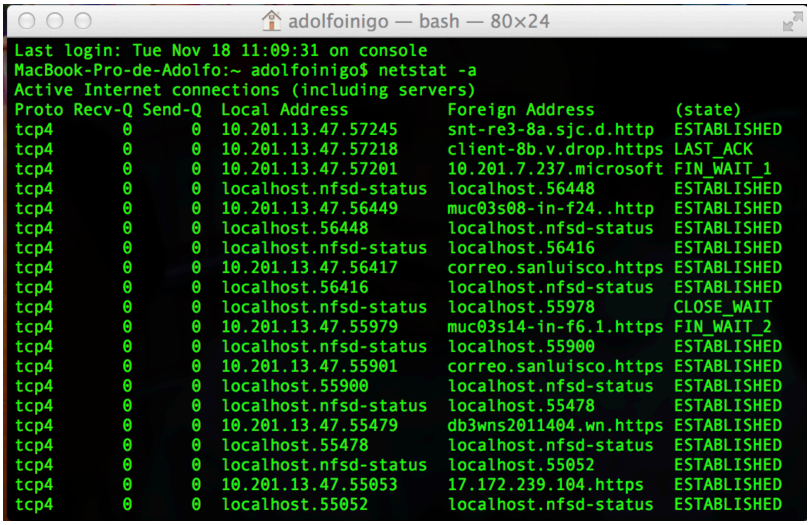
Esta capa es la que mantiene la comunicación entre ambos host y servidor y es dedicada a la programación. Esta es la que administran los firewalls y en la que se designan los puertos de comunicación que vimos en la capa anterior.

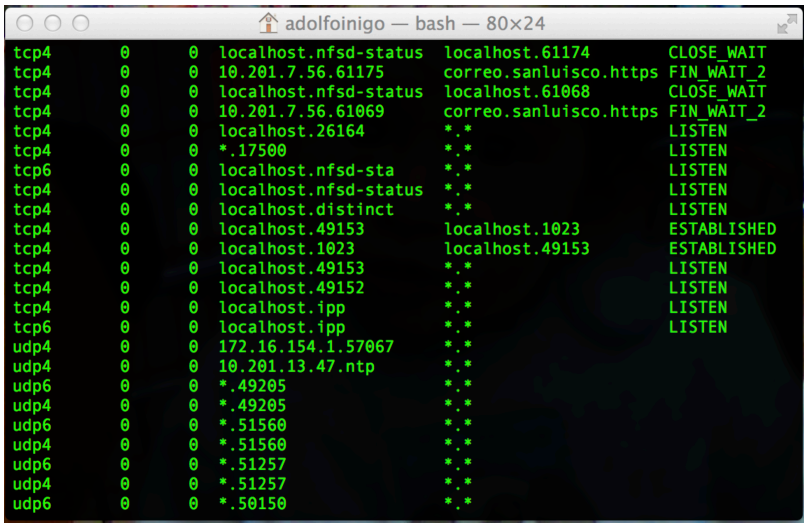
Aunque la designación de puertos implica algoritmos complejos. Puede ser descrita por medio de una analogía.

Imaginemos que tenemos un hotel en el que tenemos 65536 habitaciones pero muchos host buscan la misma habitación. En este ejemplo la habitación 21.

El recepcionista sabe que hay muchas personas buscando la misma habitación. Por lo que no puede ocupar la habitación para no cerrar a todos los que buscan esta habitación. En vez de esto ocupa alguna de las habitaciones disponibles en pisos más altos. Y ahí es donde alojará al visitante.

Al revisar esto en nuestro equipo, podemos observar que se tienen tanto puertos conocidos como puertos altos.





Como se observa en la tabla presentada se tienen los tipos de paquetes TCP o UDP también se tiene la dirección Local y la Dirección lejana. El sistema operativo lo presenta en una notación en la que se puede observar la dirección IP y posteriormente un número.

Si revisamos el numero ninguno de los presentados excede los 65536 par a las conexiones que se realizaron al host se tienen las direcciones conocidas.

Por ejemplo podemos apreciar el puerto UDP:

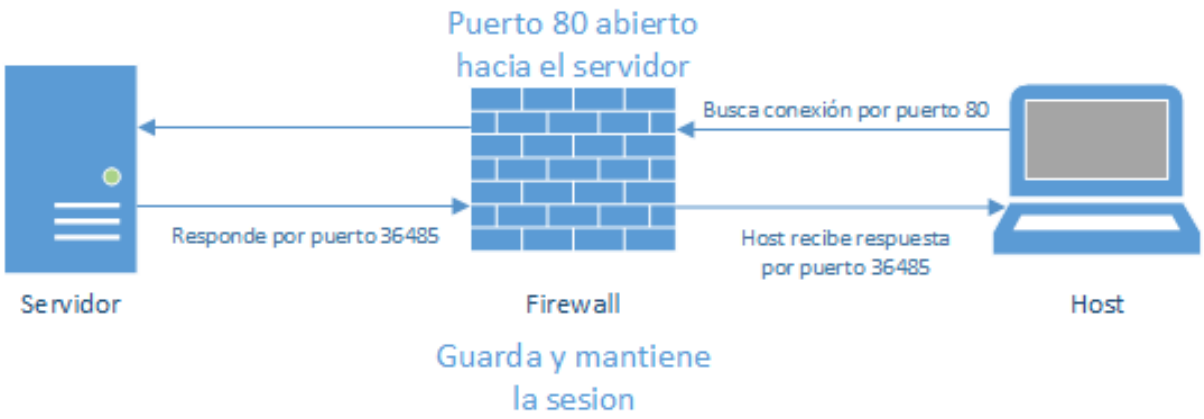
10.201.13.47.ntp esto se traduciría como 10.201.13.47:123

Como podemos también observar el equipo está conectado a través de una red interna y un equipo que da servicios como el de NTP y examina el tráfico. Este tipo de equipos son de los que hablaremos más adelante.

1.6.1. Firewalls

Para los firewalls no solo se debe de tener conexión en un solo sentido para que se mantenga la integridad de la comunicación. Lo que representa un reto para estos y que deben de guardar una memoria de sesión de los equipos que pasan a través de ellos.

Supongamos que un firewall permite o no conexión entre dos elementos un Host y un servidor web. Podemos ver en la tabla anterior de puertos conocidos el servidor web debe tener abierto el puerto 80 el puerto no debe de ser abierto desde el servidor hacia el mundo ya que esto le permitirá conexión y una amenaza en caso de que lo comprometan. El puerto únicamente se debe de abrir en un sentido. El sentido se mantiene en la memoria del firewall para permitir la conexión en ambas direcciones. Tal como se observa a continuación.



Una programación inadecuada del firewall puede generar vulnerabilidades, por lo que estos deben de tener control sobre las sesiones en una forma muy efectiva.

1.7. Capa Aplicación.

Esta es la última capa y es la que se aloja a las aplicaciones que están abriendo las comunicaciones. Es común que estas sean confundidas con los protocolos.

Tal es el caso de aplicaciones orientadas a comunicaciones de red como:

- FTP
- http
- Ping
- telnet
- ssh
- SFTP

Pero en realidad estas aplicaciones ocupan el protocolo en capas inferiores y entregan la información al usuario o a otras aplicaciones.

Capítulo 2

2. Preparación del IDS

2.1. Selección del IDS

Como a todos los proyectos, el proceso de selección suele ser un dilema, dadas las soluciones que se ofrecen en el mercado contra lo que se puede requerir y por supuesto el presupuesto.

Existen modelos establecidos para la toma de estas decisiones basadas en gobierno de TI y el destino de la corporación. Empresas especializadas en tendencia como Gartner ven en esto un nicho de mercado y una herramienta que puede ayudar a dichas decisiones. A continuación presento el cuadrante mágico de Gartner para sistemas de prevención de intrusos.

Para este caso, el hecho de tener una escuela con muy bajo presupuesto. La sola toma de decisión requiere dinero que realmente no tienen.

Además de que un Sistema de Prevención de Intrusos puede generar algunos problemas al momento de ser implementado que tampoco podrían tolerar, ya que este hace una desconexión

de los dispositivos a través de envío de paquetes RST aun cuando estos solo tengan tráfico que solo parezca sospechoso.

Es por este motivo que se toma como solución al Sistema de Detección de Intrusos SNORT que es un software libre bajo licencia GNU, que se basa en el líder del cuadrante Sourcefire y que puede ser instalado en sistemas operativos con la misma licencia. Para este caso también seleccionamos un sistema sencillo llamado CentOS que es un proyecto también libre bajo licencia GNU que está basado en un sistema muy popular llamado Red Hat el cual si posee soporte y se utiliza en ambientes corporativos y que ha madurado el proyecto a una gran estabilidad y que hereda a CentOS

2.2. Instalación del IDS

El IDS que se eligió por sus capacidades y flexibilidad es SNORT en un sistema operativo virtualizado CentOS, la combinación de ambos establece las mejores condiciones para poder implementarse desde una máquina virtual, que bien puede ser implementada en una computadora personal o en un servidor dedicado.

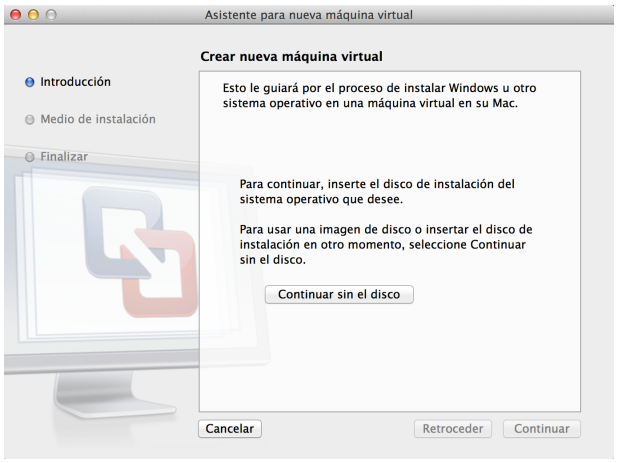
Por lo que empezaremos por establecer la máquina virtual en un equipo host con sistema operativo OSX 10.9, este utiliza versión de VMWare llamada Fusion ya que las máquinas virtuales en este sistema son ya un estándar en la industria a pesar de que existen diversas plataformas para virtualización, además de que las máquinas virtuales presentan mucha flexibilidad para poder incrementar o disminuir recursos bajo demanda de los servicios.

Empecemos por descargar las versiones desde las páginas de los proyectos; para SNORT se descarga desde la página: www.snort.org ; y para CentOS desde www.centos.org

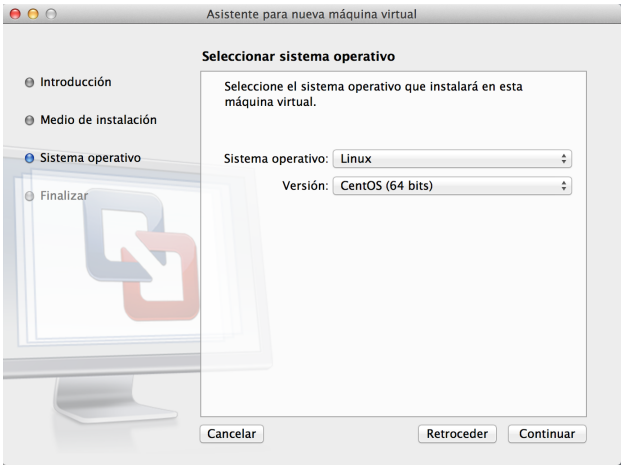
Una vez descargados procederemos a crear la maquina virtual, para esto se comienza por crear un disco duro virtual el cual albergará el sistema operativo y la informacion del maquina virtual.

2.2.1. Preparación de la máquina virtual

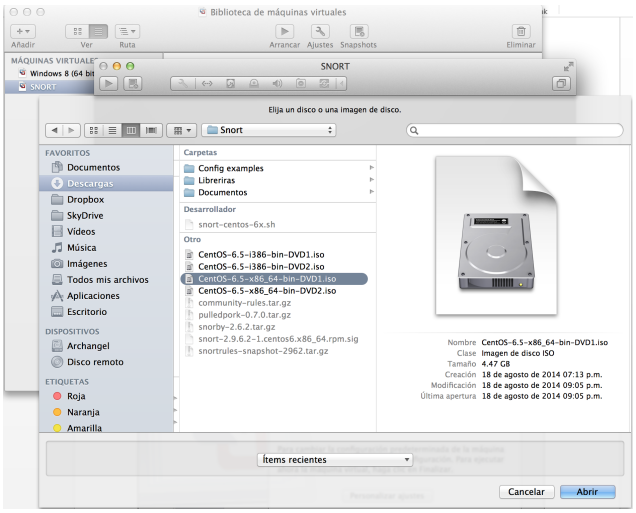
Desde un principio, se debe de espesificar el sistema operativo que utilizaremos en la maquina virtual.



La maquina virtual de acuerdo con las necesidades del sistema seleccionado, debe cumplir con las siguientes caracteristicas.

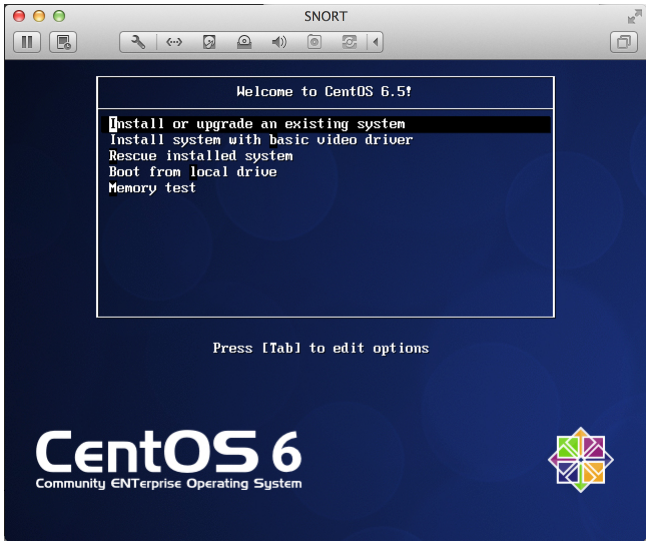


Una vez hecho esto lanzaremos la maquina virtual seleccionando para su unidad de DVD el archivo ISO que descargamos del sistema oeperativo.

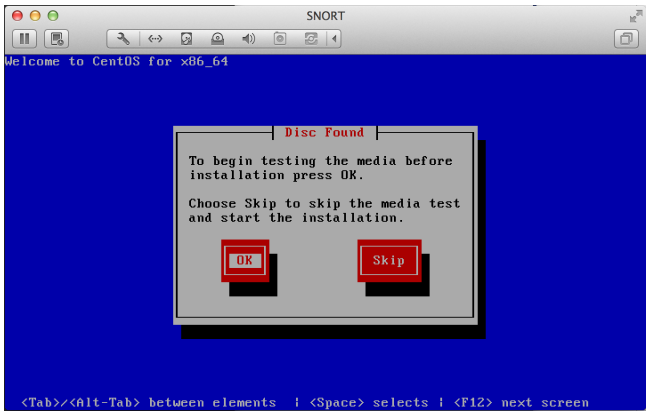


2.2.2 Instalación del sistema operativo

Al arrancar la maquina virtual nos dará la posibilidad de seleccionar las opciones que tiene el disco que hemos utilizado. Para este caso seleccionaremos Install or upgrade an exsisting system

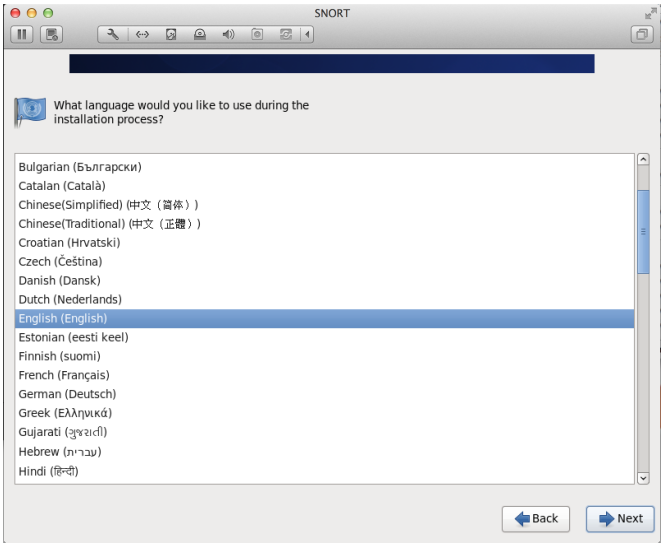


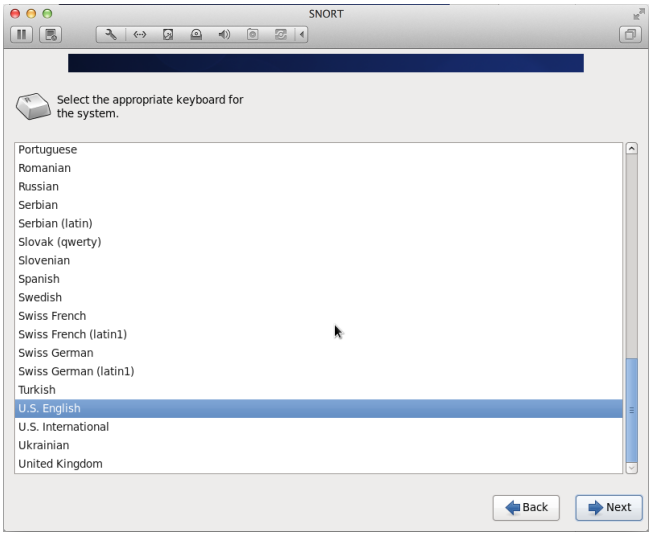
Una ves hecho esto, todas las distribuciones de Linux dan la opcion de hacer una comprovacion del disco que hemos descargado, dado que ha sido descargado desde sitios que pudieran o no tener una version integra.

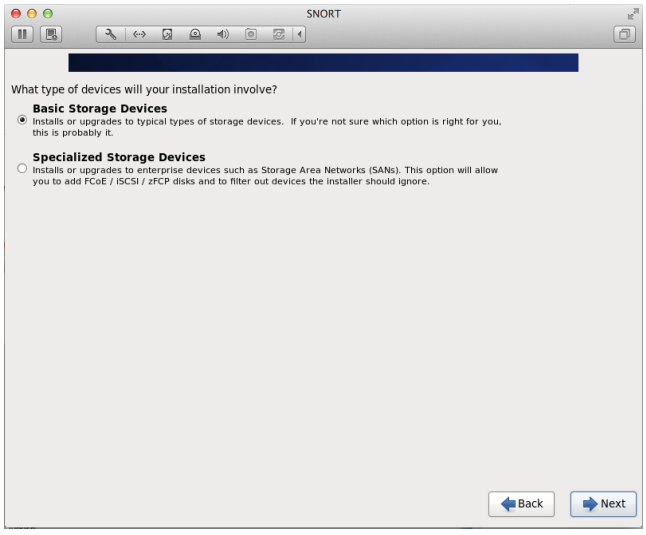


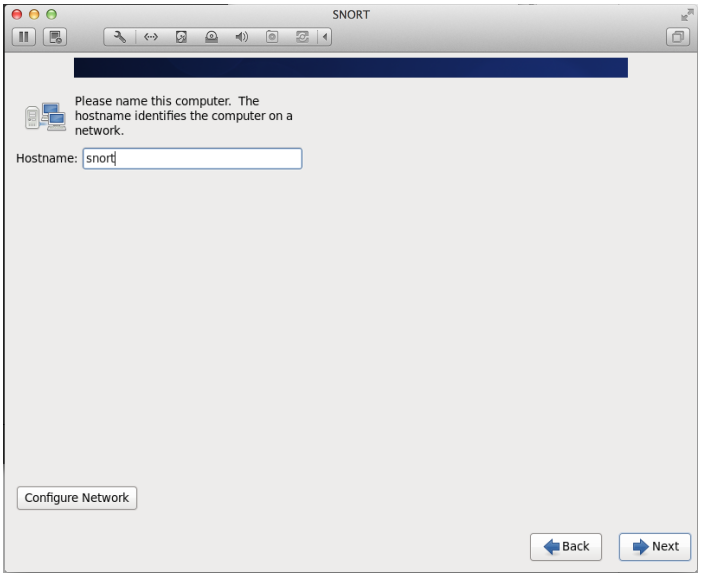
Se puede o no correr la comprovacion del disco que se descargo, esto puede evitar que la instalacion tenga problemas derivados de errores que se puedieran haber presentado durante la descarga para este caso la ignorarémos dado que fue conprovado previamente y entrarémos en la pantalla de instalacion del sistema operativo.

Pasamos a la siguiente pantalla donde podremos seleccionar el lenguaje. Para este caso seleccionaremos como leguaje de instalacion ingles y el lenguaje del teclado sera tambien en ingles, los controladores de almacenamiento serán los básicos, el nombre será SNORT

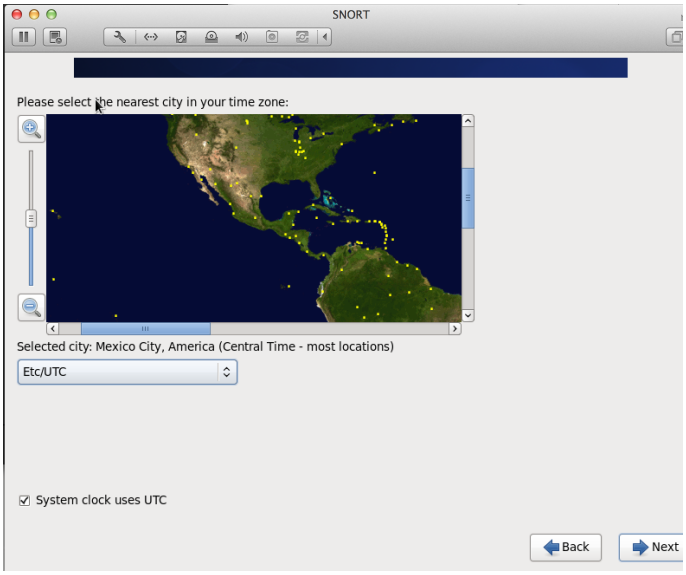

SNORT
What language would you like to use during the installation process?
Bulgarian (Български)
Catalan (Català)
Chinese(Simplified) (中文（简体）)
Chinese(Traditional) (中文（正體）)
Croatian (Hrvatski)
Czech (Čeština)
Danish (Dansk)
Dutch (Nederlands)
English (English)
Estonian (eesti keel)
Finnish (suomi)
French (Français)
German (Deutsch)
Greek (Ελληνικά)
Gujarati (ગુજરાતી)
Hebrew (עברית)
Hindi (हिन्दी)
Back
Next


SNORT
Select the appropriate keyboard for the system.
Portuguese
Romanian
Russian
Serbian
Serbian (latin)
Slovak (qwerty)
Slovenian
Spanish
Swedish
Swiss French
Swiss French (latin1)
Swiss German
Swiss German (latin1)
Turkish
U.S. English
U.S. International
Ukrainian
United Kingdom
Back
Next


SNORT
What type of devices will your installation involve?
Basic Storage Devices
Installs or upgrades to typical types of storage devices. If you're not sure which option is right for you, this is probably it.
Specialized Storage Devices
Installs or upgrades to enterprise devices such as Storage Area Networks (SANs). This option will allow you to add FCoE / iSCSI / zFCP disks and to filter out devices the installer should ignore.
Back
Next

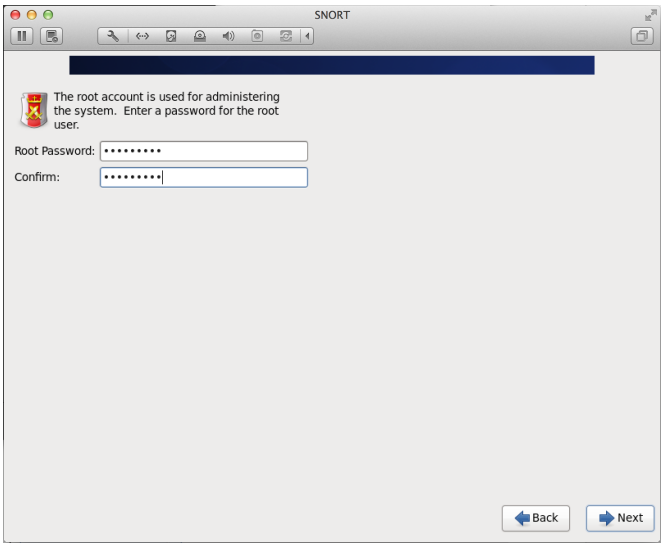


La zona horaria es un elemento importante ya que con este posteriormente se tendran que analizar eventos. Para nuestro caso se utiliza el estandar UTC para el sistema.

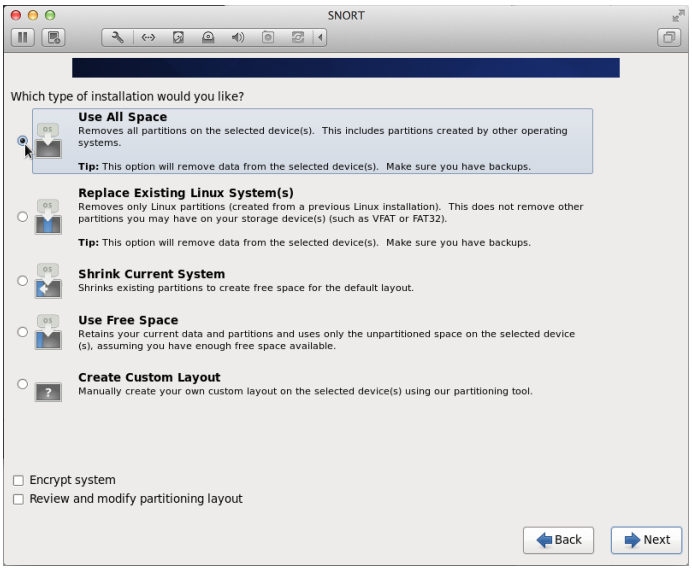


El password tambien es importante ya que este puede hacer que el sistema tenga cierta resistencia a los ataques basados en diccionario por lo que se debe de elegir una complejidad que sea dificil de adivinar y que no tenga elementos continuos como numeraciones continuas.

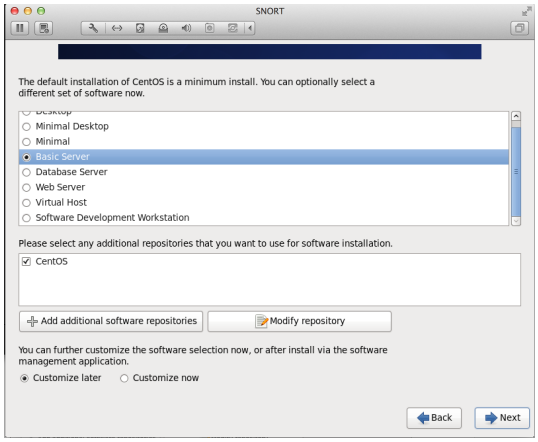
Para este caso se utiliza un metodo llamado “Mary has a little lamb” el cual usa una frase larga, se utilizan los primeras letras de la misma y se combierten a numeros y caracteres. Por obvias razones no se mencionará cual es.

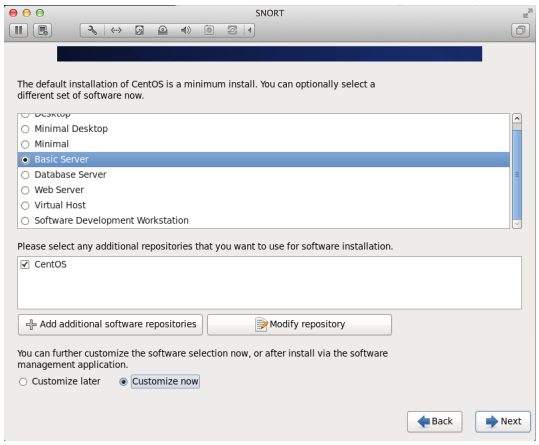


Todas las distribuciones de linux dan la posibilidad de coexistir en el mismo equipo con otro sistema operativo. Como es una maquina virtual dedicada al sistema, se usará todo el espacio al disco



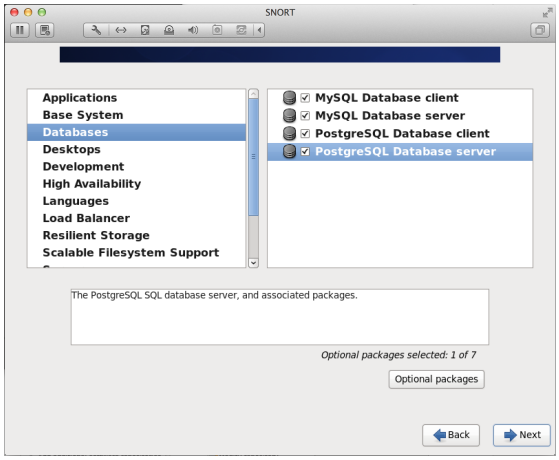
CentOS ofrece diferentes configuraciones previamente hechas para poder ahorrar pasos en el proceso de instalación. En el caso del IDS no podremos ocupar este metodo y darémos como paso inicial un servidor básico y posteriormente modificaremos los paquetes que se instalarán





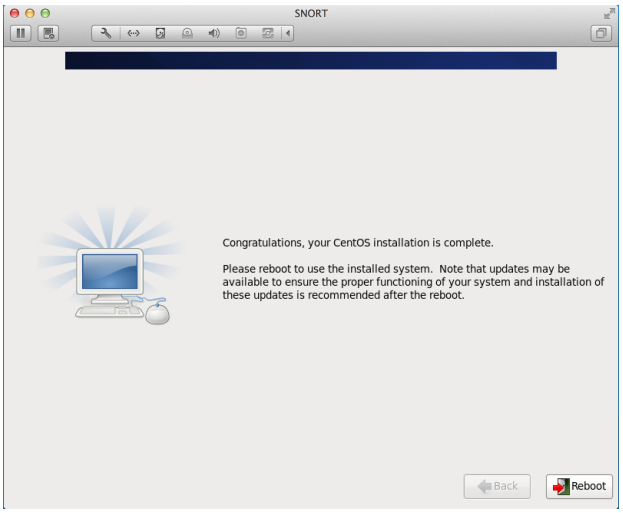
Los paquetes que se seleccionaron se veran mas adelante, entre estos estan librerías y compiladores necesarios para el funcionamiento de SNORT y ademas de esto se instalan los motores de bases de datos.

Para dar compatibilidad con corelacionadores de eventos se instala PostgreSQL pero SNORT ocupa MySQL como motor de bases de datos.

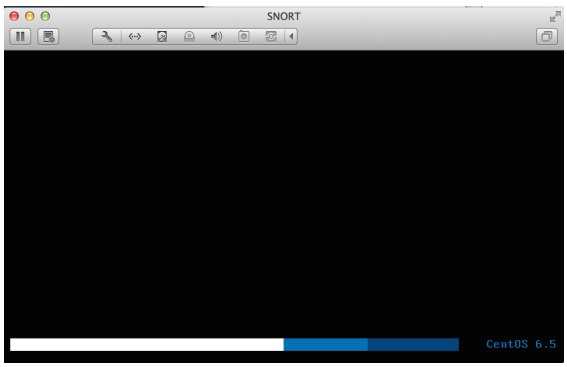


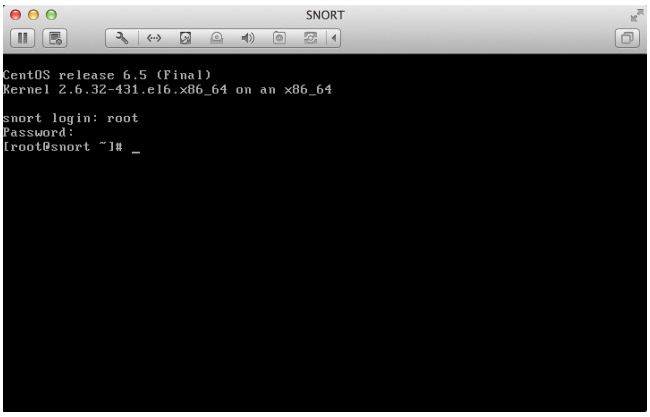
Una vez termnada la selección de los paquetes a instalar se procede con la instalación del sistema operativo.

Al finalizar la instalacion debemos de tener una pantalla similar a la que se observa a continucion. Si existiese algun problema con la instalacion es importante hacer referencia CentOS y sus foros de consulta. En este caso no fue necesario.



Los repositorios de CentOS tiene ya precompilada una version de SNORT pero para fines academicos, haremos una compilacion, ademas de que esta operación permite hacer modificaciones que pueden dar mejoras en el rendimiento al moemento de utilizarlo.

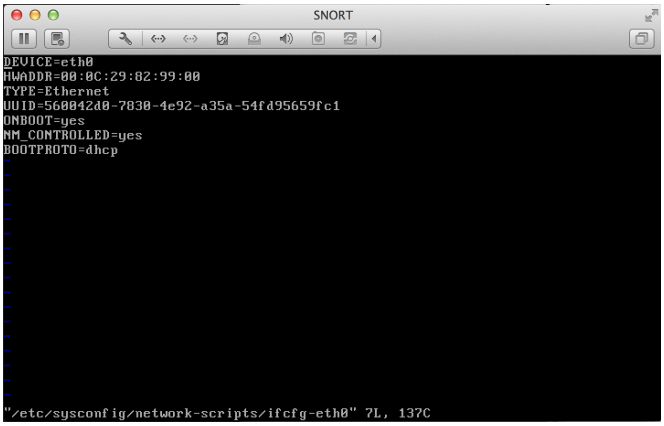




La instalacion default de CentOS tiene desactivadas las interfaces de red por lo que es necesario hacer la activacion de la interface que esta en modo NAT desde el Host que debe de tener acceso a internet.

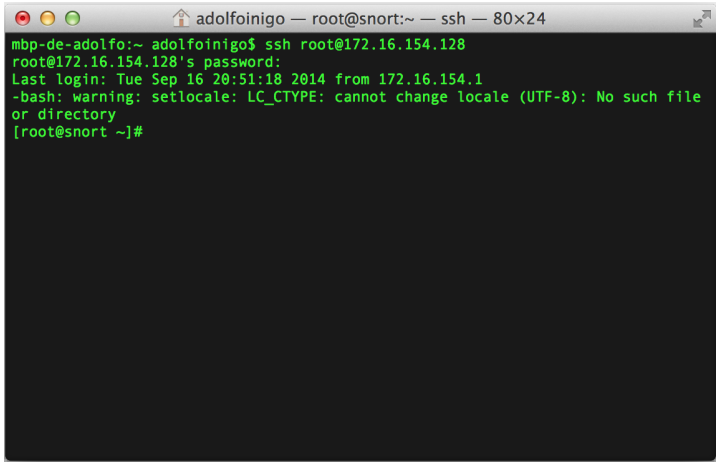
```
[root@snort ~]#vi /etc/sysconfig/network-scripts/ifcfg-eth0
```

Y solo tendremos que cambiar el parametro onboot a yes



Cuando se haya configurado la interface se podra acceder desde el sistema operativo host por medio del protocolo SSH o bien poder seguir trabajando en la maquina virtual.

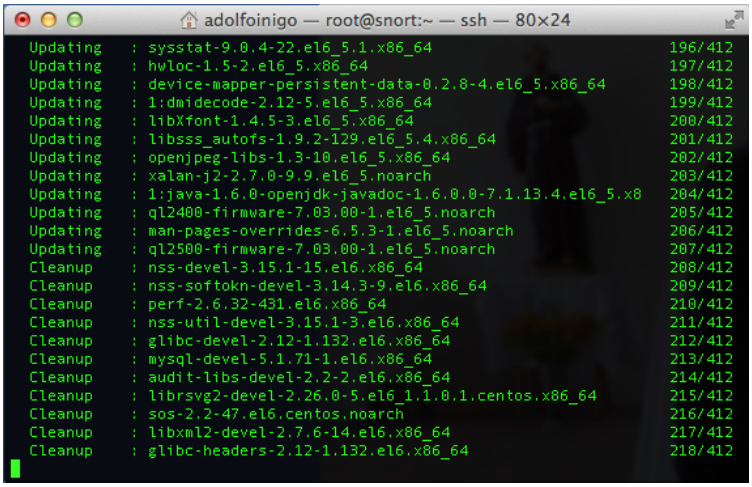
Tambien se tiene la facilidad de que el sistema operativo host brinda internet al sistema operativo CentOS.



Una vez dentro de internet prcederemos actualizar las librerías que tiene el sistema corriendo los siguientes comandos.

```
[root@snort ~]# yum update
```

Que actualizara las aplicaciones instaladas.



Una vez hecho esto nos dejara la versión de kernel mas reciente.

```
[root@snort ~]# uname -a
Linux snort 2.6.32-431.29.2.el6.x86_64 #1 SMP Tue Sep 9 21:36:05 UTC 2014 x86_64 x86_64
x86_64 GNU/Linux
[root@snort ~]#
```

2.2.3 Instalación del IDS SNORT

Teniendo la versión mas recinete del kernel para la versión comezaremos a trabajar para instalar las librerías necesarias para la compilación.

Librerías para compilar

Las librerías necesarias son:

- gcc ver 4.4.6
- flex 2.5.35
- bison 2.4.1
- zlib 1.2.3
- zlib-devel
- libpcap 1.0.0
- libpcap-devel
- pcre 7.84
- pcre-devel
- libdnet 1.11 o 1.12
- libdnet-devel
- tcpdump

Podemos confirmar la existencia de las librerías que estan en la distribucion e instalar las faltantes mediante el comando yum

A continuacion un ejemplo con gcc

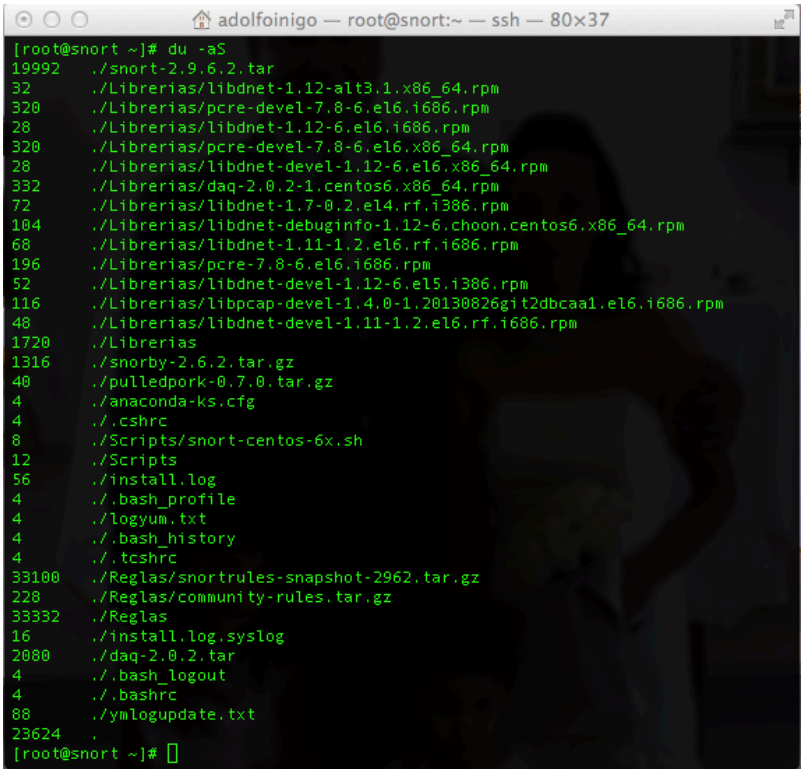
```
[root@snort ~]# yum list | grep gcc
Failed to set locale, defaulting to C
gcc.x86_64                              4.4.7-4.el6                 @anaconda-
CentOS-201311272149.x86_64/6.5
gcc-c++.x86_64                          4.4.7-4.el6                 @anaconda-
CentOS-201311272149.x86_64/6.5
gcc-gfortran.x86_64                     4.4.7-4.el6                 @anaconda-
CentOS-201311272149.x86_64/6.5
libgcc.x86_64                           4.4.7-4.el6                 @anaconda-
CentOS-201311272149.x86_64/6.5
compat-gcc-34.x86_64                    3.4.6-19.el6                base
compat-gcc-34-c++.x86_64                3.4.6-19.el6                base
compat-gcc-34-g77.x86_64                3.4.6-19.el6                base
gcc-gnat.x86_64                         4.4.7-4.el6                 base
gcc-java.x86_64                         4.4.7-4.el6                 base
gcc-objc.x86_64                         4.4.7-4.el6                 base
gcc-objc++.x86_64                       4.4.7-4.el6                 base
libgcc.i686                             4.4.7-4.el6                 base
mingw32-gcc.x86_64                      4.4.6-4.el6                 base
mingw32-gcc-c++.x86_64                  4.4.6-4.el6                 base
mingw32-gcc-gfortran.x86_64             4.4.6-4.el6                 base
mingw32-gcc-objc.x86_64                 4.4.6-4.el6                 base
mingw32-gcc-objc++.x86_64               4.4.6-4.el6                 base
[root@snort ~]#
```

Como observamos en el ejemplo ya esta instado en el sistema. Por lo que no es necesario hacer nada. Pero en caso de necesitarlo se puede convocar a que instale el paquete faltante mediante el agregado install seguido del nombre del paquete a instalar.

Despues de la revision de los paquetes habia librerías faltantes. Por lo que se procedió a su instalacion pero en el caso particular de la librería

- libdnet 1.11 o 1.12
- libdnet-devel

Para fines practicos obiaremos el como poner los archivos en linux. Y solo asumiremos que ya estan en una subcarpeta dentro de root llamda librerías.



Intentando instalar las librerías mediante el comando RPM pero me tope con el siguiente error.

```
[root@snort Librerias]# rpm -ivh libdnet-1.12-alt3.1.x86_64.rpm
error: Failed dependencies:
       rpmlib(SetVersions) is needed by libdnet-1.12-alt3.1.x86_64
```

La alerta corresponde a que hay dependencias presentes, al ver esto intente hacer la instalación mediante el comando yum que resuelve automáticamente las dependencias.

```
[root@snort Librerias]# yum localinstall /root/Librerias/libdnet-1.12-alt3.1.x86_64.rpm
Failed to set locale, defaulting to C
Loaded plugins: fastestmirror, security
Setting up Local Package Process
Examining /root/Librerias/libdnet-1.12-alt3.1.x86_64.rpm: libdnet-1.12-alt3.1.x86_64
Marking /root/Librerias/libdnet-1.12-alt3.1.x86_64.rpm to be installed
Loading mirror speeds from cached hostfile
 * base: centos.blazar.mx
 * extras: centos.blazar.mx
```

```
 * updates: centos.blazar.mx
Resolving Dependencies
--> Running transaction check
---> Package libdnet.x86_64 0:1.12-alt3.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package     Arch       Version           Repository                       Size
================================================================================
Installing:
 libdnet     x86_64     1.12-alt3.1       /libdnet-1.12-alt3.1.x86_64      71 k

Transaction Summary
================================================================================
Install       1 Package(s)

Total size: 71 k
Installed size: 71 k
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
ERROR You need to update rpm to handle:
rpmlib(SetVersions) is needed by libdnet-1.12-alt3.1.x86_64
RPM needs to be updated
 You could try running: rpm -Va --nofiles --nodigest
Your transaction was saved, rerun it with: yum load-transaction /tmp/yum_save_tx-2014-
09-27-19-20XP93Aj.yumtx
[root@snort Librerias]#
```

Como podemos observar el paquete parece estar expirado por lo que se torna necesario buscar el código y compilar la librería.

El código lo pude encontrar en la dirección:

https://libdnet.googlecode.com/files/libdnet-1.12.tgz

Por lo que lo descargué mediante siguiente comando:

```
[root@snort Librerias]# wget https://libdnet.googlecode.com/files/libdnet-1.12.tgz
--2014-09-27 19:27:51--  https://libdnet.googlecode.com/files/libdnet-1.12.tgz
Resolving libdnet.googlecode.com... 64.233.160.82, 2607:f8b0:4003:c06::52
Connecting to libdnet.googlecode.com|64.233.160.82|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 970125 (947K) [application/x-gzip]
Saving to: `libdnet-1.12.tgz'

100%[======================================>] 970,125     20.6K/s   in 27s

2014-09-27 19:28:18 (34.7 KB/s) - `libdnet-1.12.tgz' saved [970125/970125]

[root@snort Librerias]#
```

Una vez que tenemos el archivo en el servidor, procederemos a descomprimirlo a la compilación, esto lo haremos con él con las siguientes instrucciones.

Descomprimir el archivo con tar, esto creara una subcarpeta con el nombre del archivo.

```
[root@snort Librerias]# tar -vzxf libdnet-1.12.tgz
[root@snort Librerias]# cd libdnet-1.12
[root@snort libdnet-1.12]# ls
INSTALL             Makefile.in  acconfig.h  configure.in    man     trunk
LICENSE             README       aclocal.m4  dnet-config.in  python
Makefile.am         THANKS       config      include         src
Makefile.am.common  TODO         configure   libdnet.spec    test
[root@snort libdnet-1.12]#
```

Descomprimido el archivo, hay que correr un script que viene en casi todos los desarrollos de Linux que es configure, el cual revisará las los requisitos que tienen la aplicación y si es instalable o no.

```
[root@snort libdnet-1.12]# ./configure
```

La salida generalmente es algo como esto.

```
…
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
…
...
configure: creating ./config.status
config.status: creating Makefile
config.status: creating dnet-config
config.status: creating include/Makefile
config.status: creating include/dnet/Makefile
config.status: creating man/Makefile
config.status: creating src/Makefile
config.status: creating python/Makefile
config.status: creating python/setup.py
config.status: creating test/Makefile
config.status: creating test/check/Makefile
config.status: creating test/dnet/Makefile
config.status: creating include/config.h
config.status: executing depfiles commands
config.status: executing default commands
[root@snort libdnet-1.12]#
```

Si no tenemos errores noscreará archivos make, que son los archivos formales de compilación.

```
[root@snort libdnet-1.12]# make
```

Generalmente no tenemos salidas más que esperar que no existan problemas pero en este caso la salida debe de ser algo como lo siguiente:

```
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
```

```
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,--rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
```

Ahora solo resta confirmar que no existan más faltantes o que las versiones sean las correctas. Por lo que procederemos con hacer el proceso de instalación mediante compilación de SNORT, que se hará en dos partes primero se instalará la librería de captura DAQ y después SNORT

Instalación de DAQ

El proceso es muy similar al de la instalación de las librerías.

Recordemos que ya descargamos los archivos del sitio de SNORT por lo que únicamente necesitamos realizar la instalación de los mismos.

Disponibles en el sitio de SNORT están dos descargables, uno es el programa en si llamado SNORT y el otro es una librería que actualmente ha tomado el proyecto ya que no tenía quien la siguiera desarrollando que es daq. Para nuestro caso los archivos descargados son:

1 snort-2.9.6.2.tar.gz
2 daq-2.0.2.tar

Como daq es uno de los requisitos comenzaremos por instalarlo. Para esto existe un RPM por lo que únicamente correremos el comando de instalación de la librería.

```
[root@snort ~]# rpm -q daq*
package daq-2.0.2.tar is not installed
[root@snort ~]# cd Librerias/
[root@snort Librerias]# ls
daq-2.0.2-1.centos6.x86_64.rpm
index.html
index.html.1
libdnet-1.11-1.2.el6.rf.i686.rpm
libdnet-1.12
libdnet-1.12-6.el6.i686.rpm
libdnet-1.12-6.el6.x86_64.rpm
libdnet-1.12-alt3.1.x86_64.rpm
libdnet-1.12.tgz
libdnet-1.7-0.2.el4.rf.i386.rpm
libdnet-debuginfo-1.12-6.choon.centos6.x86_64.rpm
libdnet-devel-1.11-1.2.el6.rf.i686.rpm
libdnet-devel-1.12-6.el5.i386.rpm
libdnet-devel-1.12-6.el6.x86_64.rpm
libpcap-devel-1.4.0-1.20130826git2dbcaa1.el6.i686.rpm
pcre-7.8-6.el6.i686.rpm
pcre-devel-7.8-6.el6.i686.rpm
pcre-devel-7.8-6.el6.x86_64.rpm
[root@snort Librerias]# rpm -ivh daq-2.0.2-1.centos6.x86_64.rpm
Preparing...                ########################################### [100%]
   1:daq                    ########################################### [100%]
[root@snort Librerias]#
```

¿Qué es DAQ?

Bueno DAQ es el acrónimo para: Data Acquisition library, for packet I/O (Librería de adquisición de paquetes de entrada y salida) y esta es una capa media entre la aplicación de SNORT y las librerías de capturas de paquetes que de hecho ya descargamos e instalamos que son las libpcap y pcap.

Esta actúa como un lenguaje intermedio para dar más visibilidad a las capturas que requiere realizar SNORT

Preparando la Compilación de SNORT

Bueno ya tenemos todo el camino hecho para instalar SNORT así que empecemos.

Ya tenemos el archivo pero haremos un wget para obtener el más actual y únicamente lo descomprimimos.

```
[root@snort ~]# wget https://www.snort.org/downloads/snort/snort-2.9.6.2.tar.gz
--2014-09-27 20:31:23--  https://www.snort.org/downloads/snort/snort-2.9.6.2.tar.gz
Resolving www.snort.org... 54.243.242.66, 50.19.124.119, 54.225.152.149
Connecting to www.snort.org|54.243.242.66|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location:                                          https://s3.amazonaws.com/snort-org-
site/production/release_files/files/000/000/218/original/snort-
2.9.6.2.tar.gz?AWSAccessKeyId=AKIAIXACIED2SPMSC7GA&Expires=1411871687&Signature=28EmF%2
FhvhiTmnazkKTNA0VLosNI%3D [following]
--2014-09-27          20:31:24--                   https://s3.amazonaws.com/snort-org-
site/production/release_files/files/000/000/218/original/snort-
2.9.6.2.tar.gz?AWSAccessKeyId=AKIAIXACIED2SPMSC7GA&Expires=1411871687&Signature=28EmF%2
FhvhiTmnazkKTNA0VLosNI%3D
Resolving s3.amazonaws.com... 205.251.243.9
Connecting to s3.amazonaws.com|205.251.243.9|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5229232 (5.0M) [,binary/octet-stream]
Saving to: `snort-2.9.6.2.tar.gz'

100%[=============================================================================
=============================================================================
================================>] 5,229,232   14.0K/s   in 2m 54s

2014-09-27 20:34:18 (29.3 KB/s) - `snort-2.9.6.2.tar.gz' saved [5229232/5229232]

[root@snort ~]# tar -vzxf snort-2.9.6.2.tar.gz
```

Ya teniendo la carpeta entramos en ella y corremos el archivo configure.

```
[root@snort ~]# cd snort-2.9.6.2
[root@snort snort-2.9.6.2]# ./configure
```

Una vez realizado esto me arrojo el siguiente resultado:

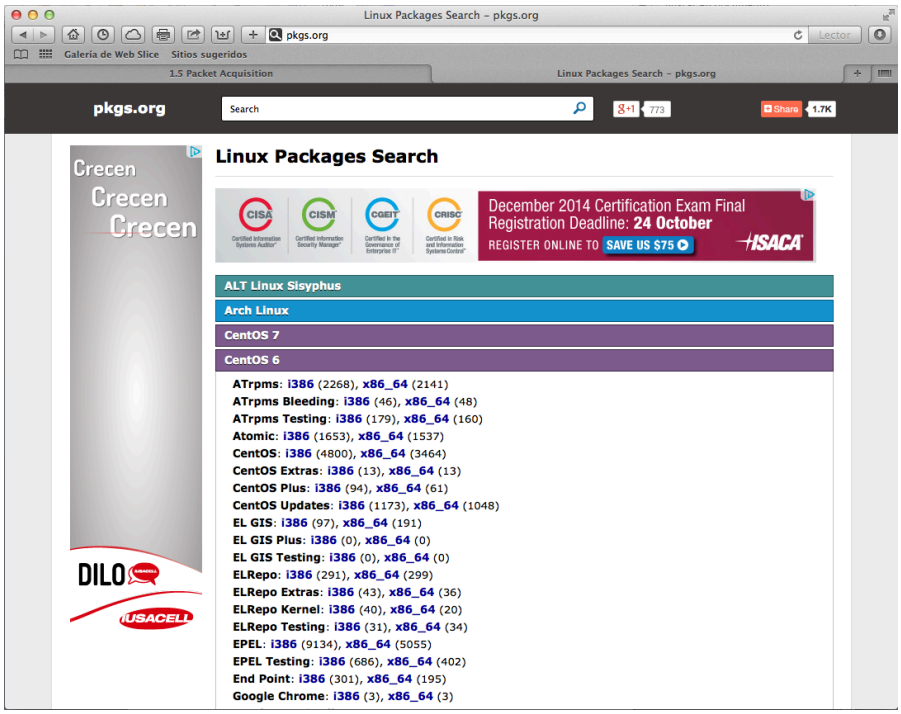
```
…
checking for pfring_open in -lpcap... no

   ERROR!  Libpcap library/headers (libpcap.a (or .so)/pcap.h)
   not found, go get it from http://www.tcpdump.org
```

```
   or use the --with-libpcap-* options, if you have it installed
   in unusual place.  Also check if your libpcap depends on another
   shared library that may be installed in an unusual place
[root@snort snort-2.9.6.2]#
```

Lo que se puede confirmar es que la versión que tenemos instalada no es compatible con la versión que SNORT requiere. Por lo que procedemos a instalar la versión necesaria.

Para instalar la versión que requiere SNORT esto hice uso de una página generada para desarrolladores conocida que ya tiene paquetes RPM previamente compilados para las más populares distribuciones de Linux llamada www.pkgs.org .



Ya que el perfil de su servidor no es de desarrollador he optado por descargar esta página las librerías en vez de ver los códigos fuente de todas aquellas faltantes y compilarlas.

Realizando comandos wget en una consola diferente y revisando el configure del SNORT tuve que no eran compatibles las siguientes:

- libpcap
- libpcap-devel
- pcre
- pcre-devel

La razón por las cuales no eran compatibles no me quedan del todo claro, pero tengo entendido que al compilar algunas librerías CentOS puede recortar funciones que son los encabezados que el script de SNORT no encontró ya que algunas versiones como el ejemplo que plasmo de GCC es incluso más actual que la versión requerida por SNORT.

```
[root@snort Librerias]# wget
http://mirror.centos.org/centos/6/os/x86_64/Packages/libpcap-1.4.0-
1.20130826git2dbcaa1.el6.x86_64.rpm
```

```
--2014-09-27 20:36:58--  http://mirror.centos.org/centos/6/os/x86_64/Packages/libpcap-
1.4.0-1.20130826git2dbcaa1.el6.x86_64.rpm
Resolving mirror.centos.org... 69.72.242.210
Connecting to mirror.centos.org|69.72.242.210|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 133568 (130K) [application/x-rpm]
Saving to: `libpcap-1.4.0-1.20130826git2dbcaa1.el6.x86_64.rpm'

100%[======================================>] 133,568     51.1K/s   in 2.6s

2014-09-27 20:37:01 (51.1 KB/s) - `libpcap-1.4.0-1.20130826git2dbcaa1.el6.x86_64.rpm'
saved [133568/133568]

[root@snort Librerias]# rpm -ivh libpcap-1.4.0-1.20130826git2dbcaa1.el6.x86_64.rpm
Preparing...                ########################################### [100%]
       package libpcap-14:1.4.0-1.20130826git2dbcaa1.el6.x86_64 is already installed
[root@snort Librerias]# wget
http://mirror.centos.org/centos/6/os/x86_64/Packages/libpcap-devel-1.4.0-
1.20130826git2dbcaa1.el6.x86_64.rpm
--2014-09-27 20:38:13--  http://mirror.centos.org/centos/6/os/x86_64/Packages/libpcap-
devel-1.4.0-1.20130826git2dbcaa1.el6.x86_64.rpm
Resolving mirror.centos.org... 69.72.242.210
Connecting to mirror.centos.org|69.72.242.210|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 116288 (114K) [application/x-rpm]
Saving to: `libpcap-devel-1.4.0-1.20130826git2dbcaa1.el6.x86_64.rpm'

100%[======================================>] 116,288      195K/s   in 0.6s

2014-09-27 20:38:13 (195 KB/s) - `libpcap-devel-1.4.0-
1.20130826git2dbcaa1.el6.x86_64.rpm' saved [116288/116288]

[root@snort Librerias]# rpm -ivh libpcap-devel-1.4.0-1.20130826git2dbcaa1.el6.
libpcap-devel-1.4.0-1.20130826git2dbcaa1.el6.i686.rpm
libpcap-devel-1.4.0-1.20130826git2dbcaa1.el6.x86_64.rpm
[root@snort Librerias]# rpm -ivh libpcap-devel-1.4.0-
1.20130826git2dbcaa1.el6.x86_64.rpm
Preparing...                ########################################### [100%]
   1:libpcap-devel          ########################################### [100%]
[root@snort Librerias]#
```

Ahora en el configure tuve los siguiente:

```
…
checking for pcre.h... no

   ERROR!  Libpcre header not found.
   Get it from http://www.pcre.org
[root@snort snort-2.9.6.2]#
```

Por lo que también descargue e instalé el rpm de esta librería.

```
[root@snort Librerias]# wget http://mirror.centos.org/centos/6/os/x86_64/Packages/pcre-
7.8-6.el6.x86_64.rpm
--2014-09-27 20:43:40--  http://mirror.centos.org/centos/6/os/x86_64/Packages/pcre-7.8-
6.el6.x86_64.rpm
Resolving mirror.centos.org... 69.20.131.50
Connecting to mirror.centos.org|69.20.131.50|:80... connected.
HTTP request sent, awaiting response... 200 OK
```

```
Length: 199516 (195K) [application/x-rpm]
Saving to: `pcre-7.8-6.el6.x86_64.rpm'

100%[======================================>] 199,516     92.4K/s   in 2.1s

2014-09-27 20:43:43 (92.4 KB/s) - `pcre-7.8-6.el6.x86_64.rpm' saved [199516/199516]

[root@snort Librerias]# rpm -ivh pcre-7.8-6.el6.x86_64.rpm
Preparing...                ########################################### [100%]
       package pcre-7.8-6.el6.x86_64 is already installed
[root@snort Librerias]# wget http://mirror.centos.org/centos/6/os/x86_64/Packages/pcre-
devel-7.8-6.el6.x86_64.rpm
--2014-09-27 20:44:48--  http://mirror.centos.org/centos/6/os/x86_64/Packages/pcre-
devel-7.8-6.el6.x86_64.rpm
Resolving mirror.centos.org... 69.20.131.50
Connecting to mirror.centos.org|69.20.131.50|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 325932 (318K) [application/x-rpm]
Saving to: `pcre-devel-7.8-6.el6.x86_64.rpm.1'

100%[======================================>] 325,932     20.7K/s   in 20s

2014-09-27 20:45:08 (15.7 KB/s) - `pcre-devel-7.8-6.el6.x86_64.rpm.1' saved
[325932/325932]

[root@snort Librerias]# rpm -ivh pcre-devel-7.8-6.el6.x86_64.rpm
Preparing...                ########################################### [100%]
   1:pcre-devel             ########################################### [100%]
[root@snort Librerias]#
```

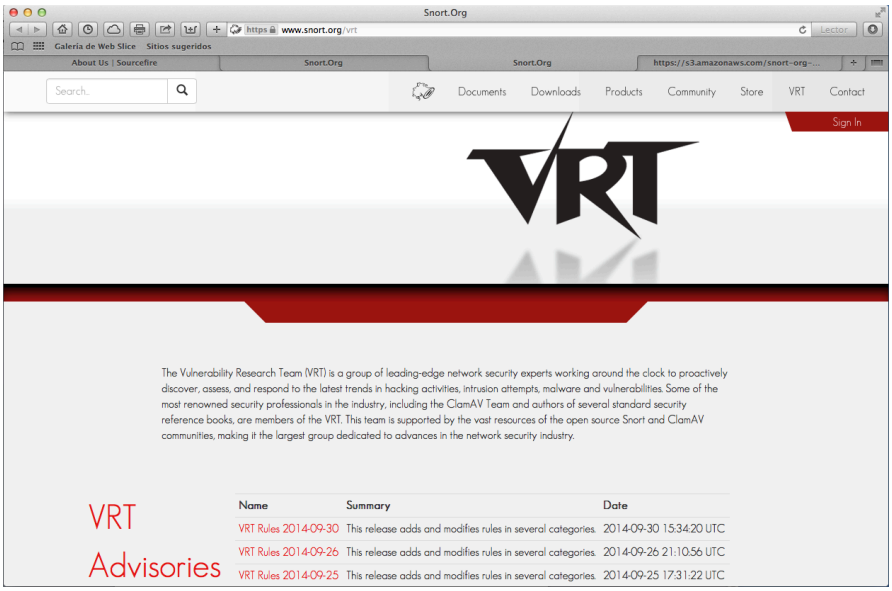
Aquí otro ejemplo, el mismo sistema operativo me mostraba que esta librería ya estaba instalada más sin embargo el script de SNORT no encontraba los encabezados necesarios para instalar hasta que instale la librería desde el RPM. Base a observación únicamente las librerías de devel (que son los cabeceros que ocupa el programa formalmente) son las que parecieron tener este tema.

En el configure de SNORT y no tuve errores por lo que proseguí con la compilación.

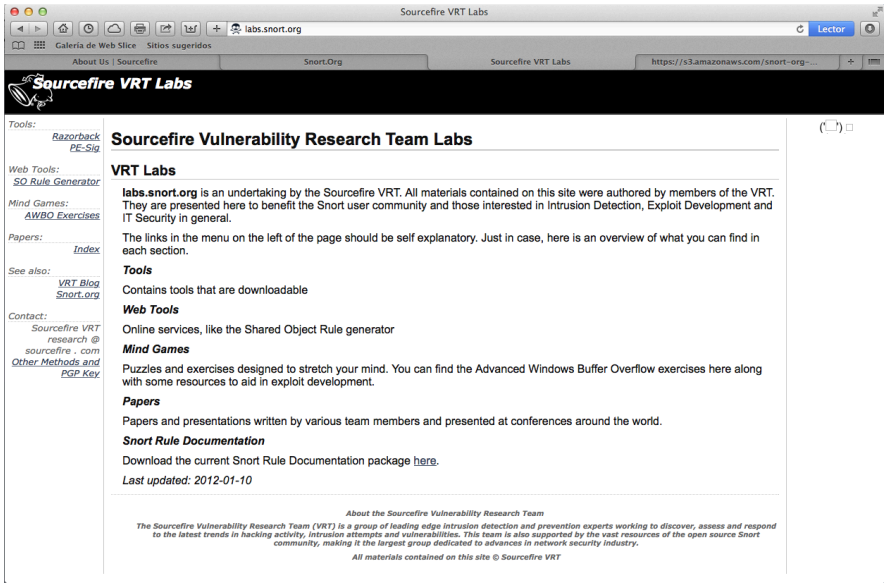
```
[root@snort snort-2.9.6.2]# ./configure –enable-sourcefire
```

El Vulnerability Research Team o VRT

El modificador enable-sourcefire se ocupa para que también habilite los parámetros necesarios para el módulo de sourcefire. Este es un módulo que se encarga de brindar comunicación con el Vulnerability Research Team (equipo de investigación de vulnerabilidades) de SNORT. Una comunidad con expertos y no tan expertos en desarrollo e implementación de reglas.



Es muy común entender que Sourcefire es el nombre comercial de SNORT pero el termino es utilizado por que es estándar de IPS que tiene licenciamiento, pero SNORT al surgir desde un proyecto de software libre mantiene su política abierta que cualquier persona puede formar parte de su comunidad.



Este módulo permite realizar reportes, analizar reglas que están siendo implementaras o compartir las reglas creadas por ustedes.

Es por eso que se toma el modulo y se habilita en el SNORT que estamos instalando.

El final del script generará archivos config.status que formalmente son los resultados, y en los que podremos encontrar información acerca del estado del sistema en compatibilidad con SNORT; pero para términos prácticos nosotros tomaremos el estado de que no nos manda error y nos da promt.

```
config.status: creating templates/Makefile
config.status: creating tools/Makefile
config.status: creating tools/control/Makefile
config.status: creating tools/u2boat/Makefile
config.status: creating tools/u2spewfoo/Makefile
config.status: creating tools/file_server/Makefile
config.status: creating src/win32/Makefile
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
[root@snort snort-2.9.6.2]#
```

La compilación de SNORT

Resuelta esta parte procederemos a la compilación formal. Corriendo el script Make que realizará la compilación de SNORT en nuestro sistema.

```
[root@snort snort-2.9.6.2]# make
```

Esto además de enseñarnos la matrix, también generará las librerías para correr SNORT en el sistema, que como el caso anterior, para términos prácticos estaremos al pendiente de que la compilación no genere errores o exceso de warnings. Y al finalizar que nos mande promt como se muestra a continuación.

```
make[3]: Leaving directory `/root/snort-2.9.6.2/tools/u2spewfoo'
make[3]: Entering directory `/root/snort-2.9.6.2/tools'
make[3]: Nothing to be done for `all-am'.
make[3]: Leaving directory `/root/snort-2.9.6.2/tools'
make[2]: Leaving directory `/root/snort-2.9.6.2/tools'
make[2]: Entering directory `/root/snort-2.9.6.2'
make[2]: Leaving directory `/root/snort-2.9.6.2'
make[1]: Leaving directory `/root/snort-2.9.6.2'
[root@snort snort-2.9.6.2]#
```

Ahora resta solo hacer la instalación de las librerías. Al ser un comando que instalará librerías en sistema es necesario escalar privilegios anteponiendo sudo.

```
 [root@snort snort-2.9.6.2]# sudo make install
```

Los resultados que buscamos es algo como lo que muestro a continuación.

```
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib/snort_dynamicengine

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
```

```
--------------------------------------------------------------------
```

Una comprobación muy sencilla de que se tiene instalado correctamente SNORT es llamarlo desde la terminal y saber que versión se ha instalado.

```
[root@snort snort-2.9.6.2]# snort --version

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.6.2 GRE (Build 77)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.4.0
           Using PCRE version: 7.8 2008-09-05
           Using ZLIB version: 1.2.3

[root@snort snort-2.9.6.2]#
```

Si todo es correcto y no tenemos errores, esto concluirá la instalación del IDS SNORT lo que nos llevara al abstracto mundo la puesta en marcha.

2.3. Puesta en marcha del IDS

Poner en Marcha un IDS no es un proceso que se nos presente de una forma muy sencilla, aunque es relativamente fácil, requiere de un poco de estudio en los archivos de configuración de SNORT y la forma en la que opera.

2.3.1 El archivo snort.conf

El archivo que se debe de ocupar para configurar el IDS es snort.conf que está en la carpeta que descomprimimos en root y que copiaremos en la carpeta /etc/snort

Como la carpeta originalmente no existe tendremos que crearla.

```
[root@snort snort-2.9.6.2]# cd /etc/
[root@snort etc]# mkdir -p snort
[root@snort snort]# cp /root/snort-2.9.6.2/etc/* .
[root@snort snort]# ls
Makefile     attribute_table.dtd    gen-msg.map       threshold.conf
Makefile.am  classification.config  reference.config  unicode.map
Makefile.in  file_magic.conf        snort.conf
[root@snort snort]#
```

Este archivo de configuración tiene internamente las variables y demás ubicaciones que debe de incluir SNORT. El archivo por defecto tiene diferentes comentarios para poder hacer la configuración por ejemplo:

```
#################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom configuration:
#
#  1) Set the network variables.
#  2) Configure the decoder
#  3) Configure the base detection engine
#  4) Configure dynamic loaded libraries
#  5) Configure preprocessors
```

```
#  6) Configure output plugins
#  7) Customize your rule set
#  8) Customize preprocessor and decoder rule set
#  9) Customize shared object rule set
##################################################
```

Aunque no seguiremos las descripciones al pie de la letra si tomaremos el orden para describir las variables del archivo.

Variables de snort.conf

1. Operadores de snort.conf
2. Operadores de decoders
3. Operadores de motor de detección.
4. Operadores de librerías dinámicas GTP
5. Operadores de preprocesadores
6. Operadores de plugins de salida
7. Operadores de reglas

Operadores de snort.conf

Las variables propias de la configuración general de SNORT en la de red, son las que tienen el prefijo ipvar y portvar a continuación se describen:

Las variables ipvar:

- **HOME_NET** .- como su nombre lo indica es la red o redes que son de tu red local
- **EXTERNAL_NET** .- las redes que el IDS tomara como externas
- **DNS_SERVERS** .- los servidores DNS que tiene la red
- **HTTP_SERVERS** .- los servidores web que tiene la red
- **SQL_SERVERS** .- los servidores de SQL que tiene la red
- **TELNET_SERVERS** .- los servidores de telnet que tiene la red
- **SSH_SERVERS** .- los servidores ssh que tiene la red
- **FTP_SERVERS** .- servidores que tienen FTP en la red
- **SIP_SERVERS** .- servidores que tienen SIP en la red.

Las variables portvar:

- **HTTP_PORTS** .- la lista de puertos que tienes en servidores web.
- **SHELLCODE_PORTS** .- lista de puertos que quieres bloquear para que no permitan comandos shell
- **ORACLE_PORTS** .- lista de puertos que se deben de supervisar para ataques de Oracle
- **SSH_PORTS** .- lista de puertos que debes de asegurar para conexiones SSH
- **FTP_PORTS** .- la lista de puertos que corren FTP
- **SIP_PORTS** .- lista de puertos que correrán SIP
- **FILE_DATA_PORTS** .- lista de inspección de archivos
- **GTP_PORTS** .- lista de puertos para procesadores de GTP

Las variables de las rutas de las reglas son las siguientes:

1. **RULE_PATH** ../rules
2. **SO_RULE_PATH** ../so_rules
3. **PREPROC_RULE_PATH** ../preproc_rules

Las variables de los motores de reputación son:

- **WHITE_LIST_PATH** ../rules
- **BLACK_LIST_PATH** ../rules

Operadores de decoders

Sigue la configuración de los decoders. Estos son los motores de análisis de los paquetes y como sabemos los paquetes son diferentes en contenido para cada aplicación es por esto que se requieren convenciones acerca de ellos. Tras la securización de los sistemas operativos, muchos de los ataques se han enfocado en las aplicaciones que suelen tener vulnerabilidades más fuertes. Las variables se caracterizan por tener el prefijo config.

- **disable_decode_alerts** .- Parar de mandar eventos de decodificación.
- **disable_tcpopt_experimental_alerts** .- Parar eventos de TCP experimental
- **disable_tcpopt_obsolete_alerts** .- parar alertas de TCP obsoletas
- **disable_tcpopt_ttcp_alerts** .- Parar alertas de T/TCP
- **disable_tcpopt_alerts** .- Parar alertas de todos las opciones de tipo TCP
- **disable_ipopt_alerts** .- Parar alertas de opciones invalidadas de IP
- **enable_decode_oversized_alerts** .- Alarmar si el valor del tamaño del paquete es mayor que si carga
- **enable_decode_oversized_drops** .- Alarmar si se tira un paquete con tamaño mayor que la carga, solo para modo en línea
- **checksum_mode** .- Configurar revisión de checksum
- **flowbits_size** .-Configuración de numero de bits de tráfico referencia
- **ignore_ports** .- Configuración de puertos a ignorar
- **response: eth0 attempts 2** .- Configuración de interface de respuesta activa si no estamos en modo en línea

Operadores de motor de detección.

A continuación se explican las variables del motor de detección también usan el prefijo config.

- **pcre_match_limit**: 3500 .- Configuración de límites de coincidencias PCRE
- **pcre_match_limit_recursion**: 1500 .- configuración de límite de coincidencias recursivas de PCRE

El motor de detección es el más importante y por esto es que tomaremos unos minutos más para rescribirlo.

- **config detection** .- Configuración de la máquina de detección.

Los operadores son:

- search-method .- método de búsqueda que puede ser:
  - *ac y ac-q* .- usando el algoritmo Aho-Corasick de búsqueda de cadenas completo.
  - *ac-bnfa y ac-bnfa-q* .- usando el algoritmo Aho-Corasick binario que ocupa poca memoria y alto rendimiento.
  - *lowmem y lowmem-q* .- búsqueda por palabras clave.
  - *ac-split Aho-Corasick* .- completo con cualquier puerto y cualquier grupo es diferente de ac-split-any-any.
  - *Intel-cpm* .- usando librería CPM de Intel se debe de tener configurado SNORT con librerías para hablitar esta.
  - *ac-nq* .- Aho-Corasick completo con método nq de búsqueda.
  - *ac-bnfa-nq* .- Aho-Corasick binario con método nq de búsqueda.
  - *lowmem-nq* .- búsqueda de palabras clave en método de búsqueda nq.

  - *ac-std* .- Aho-Corasick normal.
  - *acs* .- Aho-Corasick distribuido.
  - *ac-banded* .- Aho-Corasick en bandas.
  - *ac sparesebands* .- Aho-Corasick distribuido en bandas.
- search-optimize .- optimización de motores de patrones que se puede poner cuando usas un método de búsqueda como ac o ac-split y ac-bnfa
- max-pattern-len .- número máximo de búsqueda de patrones, es útil para despejar memoria cuando SNORT está funcionando.
- no_stream_inserts .- esto se ocupa cuando quieres que SNORT envíe el paquete corregido. Pero puede costar mucho performance.
- max_que_events .- especifica el número máximo de patrones rápidos por paquete el valor si no se especifica es 5
- enable-single-rule-group .- pone todas las reglas en un solo grupo de puertos. No se recomienda hacerlo.
- bleedover-port-limit .- el número máximo de puertos de origen y destino destinados en una regla antes de la regla sea considerada como any-any en grupo de puertos, el que tiene si no se especifica es 1024

event_queue .- Espesifica SNORT la consulta de eventos sus opciones son:

- max_queue .- máximos eventos soportados
- log .- número de eventos que irán al log
- order_events como ordenar los eventos que irán al log sus opciones son
  - *content_length* .- tamaño del contenido
  - *lpriority* .- prioridad del evento

Operadores de librerías dinámicas GTP

Ahora es tiempo de habilitar el motor GTP. ¿Qué es GTP? es un protocolo para hacer tunes para redes GPRS usualmente viaja arriba de TCP y es comúnmente usado en redes que aplican calidad de servicio la variable solo acepta habilitar o deshabilitar por lo que no ahondaremos en él más de lo que ahora explico. La variable utiliza el prefijo config.

1 enable_gtp

SNORT ofrece posibilidades de configurar temas tan finos como la latencia de la red, perfiles y depuración del flujo por ahora, no ahondaremos en ellos por no ser necesarios en esta instalación pero es un tema que puede bien ofrecer un gran estudio al respecto.

Ahora veremos la parte de las librerías dinámicas. Las librerías dinámicas son introducidas desde la versión de SNORT 2.6 y son librerías independientes dedicadas a detección. Las variables están dedicadas a ubicar dichas librerías en rutas.

2 dynamicpreprocessor directory /usr/local/lib/snort_dynamicpreprocessor/
3 dynamicengine /usr/local/lib/snort_dynamicengine/libsf_engine.so
4 dynamicdetection directory /usr/local/lib/snort_dynamicrules

Las variables de preprocesadores no las ahondaremos por ser implícitas y campo de estudio aparte. Pero podemos decir que son procesadores que pueden ser configurados para cada protocolo en elementos tan complejos como la estructura de sus paquetes y como conforman su comunicación. Solo pondré un ejemplo de ellos que es el preprocesador para SIP

```
reprocessor sip: max_sessions 40000, \
  ports { 5060 5061 5600 }, \
  methods { invite \
        cancel \
        ack \
        bye \
        register \
        options \
        refer \
        subscribe \
        update \
        join \
        info \
        message \
        notify \
        benotify \
        do \
        qauth \
        sprack \
        publish \
        service \
        unsubscribe \
        prack }, \
  max_uri_len 512, \
  max_call_id_len 80, \
  max_requestName_len 20, \
  max_from_len 256, \
  max_to_len 256, \
  max_via_len 1024, \
  max_contact_len 512, \
  max_content_len 2048
```

SNORT es muy flexible y puede entregar sus resultados en diferentes formas. Los logs y alertas están activos todo el tiempo que SNORT funciona y están después de las máquinas de detección. Las salidas como alertas y logs es la parte medular e importante hacer de que queremos obtener del detector de intrusos. Se puede tener la mejor herramienta para detectar intrusiones pero si no entran en un proceso de depuración no sirven al sistema.

Si hay múltiples alertas relacionadas al mismo evento estas se encolan y son presentadas conforme el evento ocurrió. A continuación un listado de las formas en las que puede SNORT entregar los datos:

- **alert_syslog** .- este módulo manda la información al log del sistema
- **alert_fast** .- esto entregará los eventos de SNORT en un formato de una sola línea en un archivo que espesifiquemos.
- **alert_full** .- esto imprimirá el mensaje de alerta de SNORT con todos los cabeceros del paquete. Las alertas serán escritas en el directorio de log de SNORT /var/log/snort
- **alert_unixsock** .- configura un socket de dominio de UNIX y lo envía a él.
- **log_tcpdump** .- Este entrega los paquetes en un formato de tcpdump.
- **csv** .- este plugin entrega los datos de alerta en un archivo separado por comas.
- **unified** .- este es el plugin diseñado para trasferir la información lo más rápido posible a otros programas en formato binario, y es muy recomendable para el uso de correlacionadores de eventos.
- **unified2** .- muy similar al unified solo cambia en el formato de entrega.

- **log_null** .- para cuando necesitan una alerta especifica este módulo nos ayudará deshabilitar la salida en log.
- **<limit>** .- este módulo ayuda a generar logs en base a límites, si un log alcanza el límite establecido se cierra y se genera uno nuevo.

Operadores de reglas

Las reglas es uno de los elementos más importantes de SNORT. Aunque en este momento solo veremos que en el archivo snort.conf solo se configuran las rutas hacia los archivos donde tienes las reglas y solo pondré un ejemplo.

- **include $RULE_PATH/local.rules**

Como podrán notar en este ejemplo, se usa ya la variable que definimos anteriormente para la ruta de las reglas y se declara que se incluya el archivo de reglas local.rules

En esta forma concluiremos la descripción de los elementos que componen el archivo snort.conf y empezaremos a hablar de las reglas.

2.3.2 Las reglas de SNORT

Las reglas son los elementos que nos permitirán realizar diversas acciones usando SNORT; desde solo alertar un evento como una conexión mal hecha, hasta bloquear comunicaciones de aplicaciones maliciosas o incluso bloquear tráfico de forma selectiva.

Las reglas es lo que hace que SNORT sea la plataforma más completa del mercado y líder en el cuadrante mágico de Gartner. Esto es porque lo que los miembros de VRT publican las reglas y estas pueden ser compartidas y descargadas por la comunidad o comprar licencias, para que laboratorios como el mismo VRT o algún otro laboratorio que desarrolle de forma independiente reglas, proporcione estas para detección actualizada a las más recientes amenazas.

Recordemos que en el negocio de la seguridad informática un día de retraso puede significar millones en pérdidas para algunas instituciones como las bancarías. Pero nuevamente al ser una implementación simple. Con el solo hecho de pertenecer a la comunidad VRT que se puede hacer con una simple suscripción, es más que suficiente para obtener las reglas con 30 días de retraso en vez de las más cotizadas de día cero.

Para esto nuevamente iremos al sitio de SNORT. www.snort.org y tras hacer un simple login en sus sitio bajaremos las reglas más recientes.

Para nuestro caso descargamos los siguientes archivos:

- **snortrules-snapshot-2962.tar.gz** .- que es el archivo de las reglas por default de SNORT.
- **community-rules.tar.gz** .- que es el archivo de reglas de comunidad VRT con treinta días de pruebas previas.

Una vez hecho esto los depositaremos en la carpeta conocida y daremos los siguientes comandos:

```
[root@snort Reglas]# ls
community-rules.tar.gz  snortrules-snapshot-2962.tar.gz
[root@snort Reglas]# tar -vzxf snortrules-snapshot-2962.tar.gz
rules/
rules/VRT-License.txt
rules/app-detect.rules
rules/attack-responses.rules
rules/backdoor.rules
rules/bad-traffic.rules
rules/blacklist.rules
rules/botnet-cnc.rules
rules/browser-chrome.rules
.
.
.
so_rules/precompiled/Debian-6-0/x86-64/2.9.6.2/file-pdf.so
so_rules/precompiled/Debian-6-0/x86-64/2.9.6.2/file-flash.so
so_rules/precompiled/Debian-6-0/x86-64/2.9.6.2/indicator-shellcode.so
so_rules/precompiled/Debian-6-0/x86-64/2.9.6.2/file-office.so
so_rules/protocol-voip.rules
so_rules/snmp.rules
so_rules/multimedia.rules
so_rules/chat.rules
so_rules/file-office.rules
so_rules/malware-cnc.rules
so_rules/icmp.rules
so_rules/misc.rules
etc/
etc/classification.config
etc/reference.config
etc/sid-msg.map
etc/snort.conf
etc/threshold.conf
etc/unicode.map
preproc_rules/
preproc_rules/decoder.rules
preproc_rules/preprocessor.rules
preproc_rules/sensitive-data.rules
[root@snort Reglas]# tar -vzxf community-rules.tar.gz
community-rules/
community-rules/community.rules
community-rules/AUTHORS
community-rules/LICENSE
community-rules/sid-msg.map
community-rules/VRT-License.txt
[root@snort Reglas]#
```

Lo que nos debe de dejar una estructura como la que a continuación presento:

```
[root@snort Reglas]# ls
community-rules         preproc_rules                    so_rules
community-rules.tar.gz  rules
etc                     snortrules-snapshot-2962.tar.gz
[root@snort Reglas]#
```

Ahora crearemos los archivos de listas negras y listas blancas y copiaremos las reglas que tiene por default SNORT que son básicamente impresiones o imágenes sencillas de las reglas donde

pueden o no venir y pueden ser solo encabezados que SNORT ocupará para implementar posteriormente motores de reputación. Por lo que haremos lo siguiente:

```
[root@snort Reglas]# mkdir /etc/snort/rules/
[root@snort Reglas]# touch /etc/snort/rules/white_list.rules
[root@snort Reglas]# touch /etc/snort/rules/black_list.rules
[root@snort Reglas]# cp rules/* /etc/snort/rules/
[root@snort Reglas]# ls /etc/snort/rules/
VRT-License.txt              info.rules               protocol-voip.rules
app-detect.rules             local.rules              pua-adware.rules
attack-responses.rules       malware-backdoor.rules   pua-other.rules
backdoor.rules               malware-cnc.rules        pua-p2p.rules
bad-traffic.rules            malware-other.rules      pua-toolbars.rules
black_list.rules             malware-tools.rules      rpc.rules
blacklist.rules              misc.rules               rservices.rules
botnet-cnc.rules             multimedia.rules         scada.rules
browser-chrome.rules         mysql.rules              scan.rules
browser-firefox.rules        netbios.rules            server-apache.rules
browser-ie.rules             nntp.rules               server-iis.rules
browser-other.rules          oracle.rules             server-mail.rules
browser-plugins.rules        os-linux.rules           server-mssql.rules
browser-webkit.rules         os-mobile.rules          server-mysql.rules
chat.rules                   os-other.rules           server-oracle.rules
content-replace.rules        os-solaris.rules         server-other.rules
ddos.rules                   os-windows.rules         server-samba.rules
deleted.rules                other-ids.rules          server-webapp.rules
dns.rules                    p2p.rules                shellcode.rules
dos.rules                    phishing-spam.rules      smtp.rules
experimental.rules           policy-multimedia.rules  snmp.rules
exploit-kit.rules            policy-other.rules       specific-threats.rules
exploit.rules                policy-social.rules      spyware-put.rules
file-executable.rules        policy-spam.rules        sql.rules
file-flash.rules             policy.rules             telnet.rules
file-identify.rules          pop2.rules               tftp.rules
file-image.rules             pop3.rules               virus.rules
file-java.rules              protocol-dns.rules       voip.rules
file-multimedia.rules        protocol-finger.rules    web-activex.rules
file-office.rules            protocol-ftp.rules       web-attacks.rules
file-other.rules             protocol-icmp.rules      web-cgi.rules
file-pdf.rules               protocol-imap.rules      web-client.rules
finger.rules                 protocol-nntp.rules      web-coldfusion.rules
ftp.rules                    protocol-other.rules     web-frontpage.rules
icmp-info.rules              protocol-pop.rules       web-iis.rules
icmp.rules                   protocol-rpc.rules       web-misc.rules
imap.rules                   protocol-scada.rules     web-php.rules
indicator-compromise.rules   protocol-services.rules  white_list.rules
indicator-obfuscation.rules  protocol-snmp.rules      x11.rules
indicator-scan.rules         protocol-telnet.rules
indicator-shellcode.rules    protocol-tftp.rules
[root@snort Reglas]#
```

Ahora copiaremos también las reglas faltantes de sistema de preprocesadores y de comunidad para esto haremos lo siguiente:

```
[root@snort Reglas]# mkdir /etc/snort/so_rules/
[root@snort Reglas]# mkdir /etc/snort/community-rules/
[root@snort Reglas]# mkdir /etc/snort/preproc_rules/
[root@snort Reglas]# cp –rf so_rules/* /etc/snort/so_rules/
[root@snort Reglas]# cp -rf preproc_rules/* /etc/snort/preproc_rules/
[root@snort Reglas]# cp -rf community-rules/* /etc/snort/community-rules/
```

```
[root@snort Reglas]#
```

Estas reglas, se pueden especificar o modificar y se quedan disponibles para la operación de SNORT.

Muy importante es también crear la carpeta para las reglas dinámicas que aun que ahora solo las dejaremos vacías, nos ahorraremos un error en momento de ejecutar SNORT.

```
[root@snort Reglas]# mkdir /usr/local/lib/snort_dynamicrules/
```

Los archivos que acabamos de copiar en su interior si tienen reglas hechas para el sistema y son reglas que pueden ser ocupadas por las más amenazas existentes. Por ejemplo si alguien quiere hacer un ataque de buffer overflow del stack de TCP se puede crear una regla específica para esta situación. O si hay dependencia entre reglas para que se activen entre sí son las reglas dinámicas. Poe ejemplo se detecta una conexión de FTP el preprocesador puede arrancar las reglas para que estas sean activadas y si hay un comportamiento extraño en la conexión, activar una siguiente regla para verificar que sea el comportamiento conocido o invalido.

Formato de reglas SNORT

Las reglas tienen una notación en SNORT y pueden ser configuradas de forma rápida, es un lenguaje simple. Para poder configurar las reglas necesarias, y a medida de las necesidades que tenemos, a continuación pongo un ejemplo de una regla.

alert tcp any any -> 192.168.1.1 111 \ (content:”|00 01 86 a5|”; msg: “mountd access”;)

Esta regla se puede interpretar como: lo que pase por el IDS que provenga de cualquier lado y valla a la IP 192.168.1.1 puerto 111 y tenga el contenido: |00 01 86 a5| hará que SNORT envíe una alerta con el mensaje: mountd Access
Las reglas tienen una estructura muy simple:

1. La acción de la regla
2. El origen
3. El operador de dirección.
4. El destino
5. El contenido a analizar
6. Por ultimo las opciones.

Un ejemplo de sintaxis de una regla puede ser:

activate tcp !$HOME_NET any -> $HOME_NET 143 (flags:PA; \content:"|E8C0FFFFFF|/bin"; activates:1; \msg:"IMAP buffer overflow!";) dynamic tcp !$HOME_NET any -> $HOME_NET 143 (activated_by:1; count:50;)

Las acciones que se pueden tomar son:

- **alert** .- genera una alerta y pasa al log el paquete
- **log** .- pasa a informa al log sobre el paquete
- **pass** .- ignora al paquete
- **activate** .- alerta y después enciende otra regla
- **dynamic** .- permanece en estado pasivo hasta que es activada por una regla y después actúa como una regla de log
- **drop** .- tira el paquete e informa al log
- **reject** .- bloquea al paquete, lo informa al log y manda un TCP RST o un ICMP port **unreachable** dependiendo del caso de protocolo.

- **sdrop** .- bloquea el paquete pero no lo informa al log.

El operador de dirección únicamente acepta dos patrones:

-> .- que indica una sola dirección.
<> .- que indica para ambas direcciones.

El origen y el destino son expresados en direcciones IP y dado que en el operador dirección no existe <- se deben de tomar el orden en la regla para origen y destino y expresar la IP como se muestra a continuación.

Las IPs y puertos pueden seguir la notación conocida IP y puerto:

XXX.XXX.XXX.XXX:XXXX
XXX.XXX.XXX.XXX XXXX

También se pueden especificar rangos de puertos con el espacio

XXX.XXX.XXX.XXX XXXX:XXXX

Para aplicar un filtro inverso también se pueden negar con el signo de admiración.

!XXX.XXX.XXX.XXX

Ejemplos.

192.168.1.1: 111
192.168.1.1 111
192.168.1.1 1:1024
!192.168.1.1 111

Para los mensajes y para las demás opciones se pueden hacer variaciones dependiendo de las necesidades y el sistema con el que convivirá el IDS ya que es común que estos se integren a sistemas más complejos que se llaman SGSI o sistema de gestión de seguridad de la información que correlaciona los eventos de la red con logs de sistemas o alertas de otros sistemas.

Como en la implementación que haremos, es una instalación simple, con configuración de consola y administrada directamente por el responsable de sistemas. No será necesario enviar mensajes más allá del log o de las alertas que es el siguiente formato:

- msg:”<mensaje en texto>”;

Capítulo 3

3. Implementación

2.4. Definición de las necesidades de la escuela.

La implementación en la escuela al igual que su sistema tiene necesidades muy simples y puntuales. Dado que tienen alumnos que son supervisados por los profesores y la red de visitantes está completamente separada de los laboratorios y de los equipos de la escuela y de alumnos en la red.

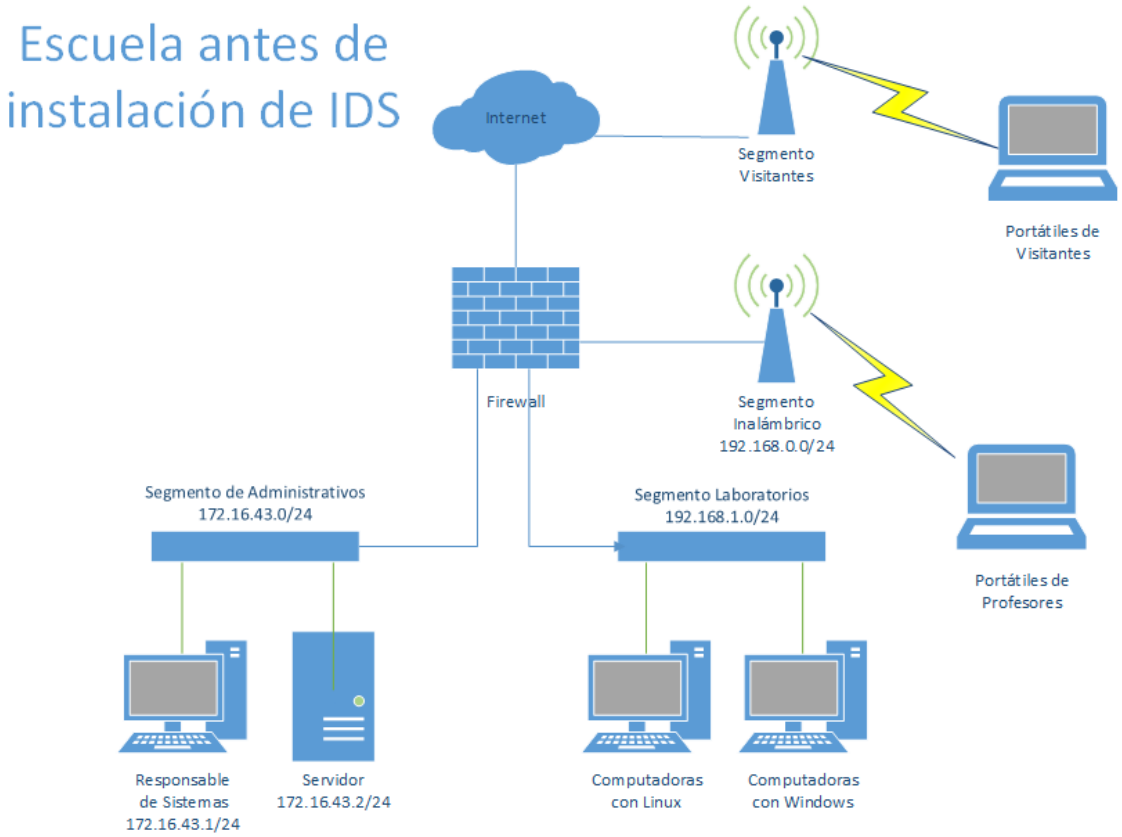
2.4.1. Alcance

Una vez en sitio y determinadas las necesidades básicas de la escuela se tiene el siguiente alcance:

"Detectar los posibles ataques desde la red externa hasta la red local"

2.4.2. Infraestructura de la escuela

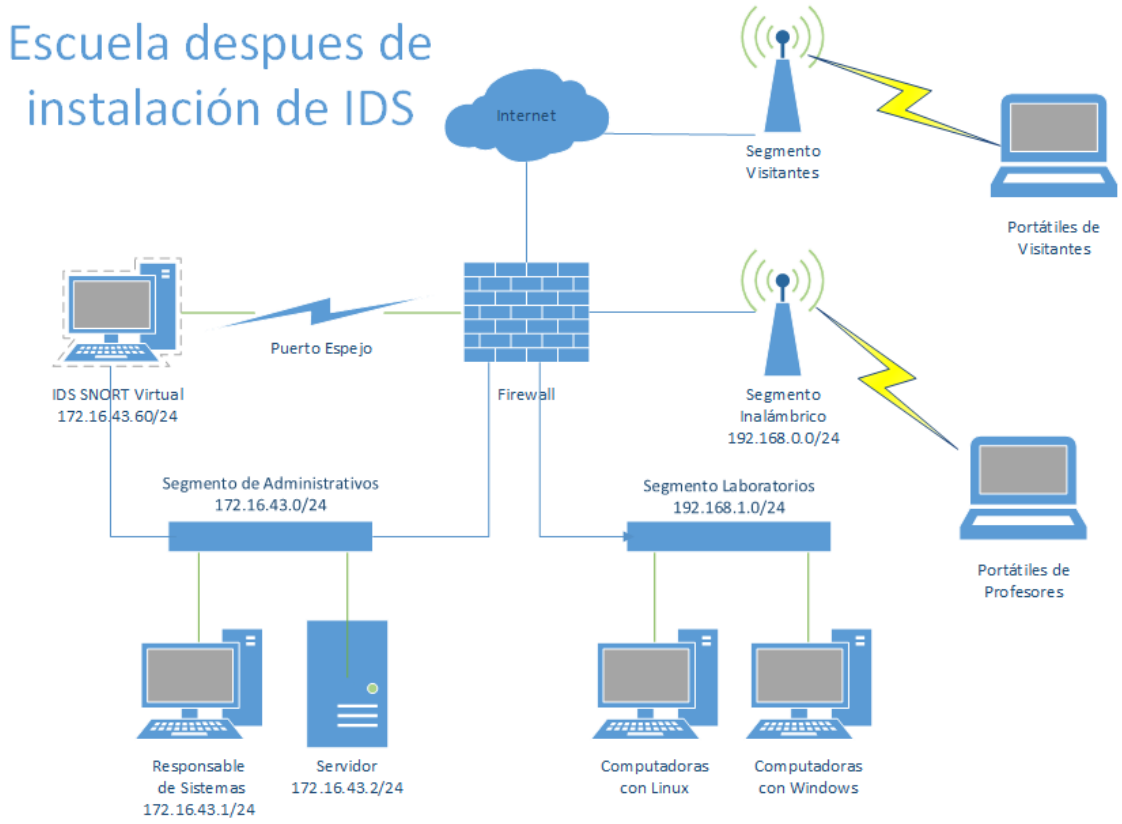
Para esto se ilustra la siguiente red en el sitio que es muy simple. No tiene servicios web publicados al mundo y sus segmentaciones son físicas, es decir no maneja VLANs. Utiliza un firewall que solo hace filtrado de contenido y un servidor muy simple para su sistema de control escolar que vive en el mismo segmento que los administrativos.



2.4.3. Ubicación del IDS

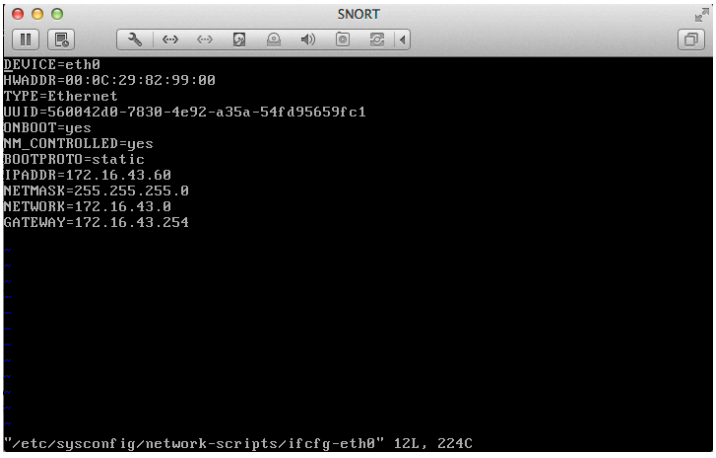
El IDS se instalará en un puerto espejo del firewall que reportará todo el tráfico hacia el. No existirán medios intermedios y se pondrá una IP de administración que ira hacia el segmento de administradores.

Esto se ilustra en el diagrama siguiente:



2.4.4. Configuración del IDS

Ahora que sabemos la IP, debemos de configurar una IP fija para poder tener una consola de administración del servidor con los siguientes datos:



Ahora solo conectamos el adaptador dos que tenemos vinculado físicamente a el puerto espejo y tal como lo hicimos en el caso de la primera interface la habilitaremos en el sistema operativo.

Un detalle importante al momento de generar un dispositivo de red para máquinas virtuales es que se tiene que hacer mientras la máquina virtual está apagada y en unos casos, Linux no

reconoce de forma automática y genera archivos de configuración para la tarjeta de red nueva, por lo que tienen que generar un archivo nuevo.

En este caso el adaptador no fue reconocido ni asigno adaptador para la nueva tarjeta de red. Por lo que tuve que regenerar el archivo 70-persistent-net.rules que está en la ruta /etc/rules.d/ del sistema operativo. Reinicié el equipo y se regenero de forma automática a continuación los pasos:

```
[root@snort ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:82:99:00 brd ff:ff:ff:ff:ff:ff
    inet 172.16.43.60/24 brd 172.16.43.255 scope global eth0
    inet6 fe80::20c:29ff:fe82:9900/64 scope link
       valid_lft forever preferred_lft forever
[root@snort ~]#
[root@snort ~]# rm /etc/udev/rules.d/70-persistent-net.rules
rm: remove regular file `/etc/udev/rules.d/70-persistent-net.rules'? y
[root@snort ~]# shutdown -r now

Broadcast message from root@snort
     (/dev/pts/0) at 18:10 ...

The system is going down for reboot NOW!
[root@snort ~]# Connection to 172.16.43.60 closed by remote host.
Connection to 172.16.43.60 closed.
MacBook-Pro-de-Adolfo:~ adolfoinigo$
MacBook-Pro-de-Adolfo:~ adolfoinigo$ ssh root@172.16.43.60
root@172.16.43.60's password:
Last login: Tue Aug 5 17:59:22 2014 from 172.16.43.1
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or
directory
[root@snort ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:82:99:00 brd ff:ff:ff:ff:ff:ff
    inet 172.16.43.60/24 brd 172.16.43.255 scope global eth0
    inet6 fe80::20c:29ff:fe82:9900/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:82:99:0a brd ff:ff:ff:ff:ff:ff
    inet6 fe80::20c:29ff:fe82:990a/64 scope link
       valid_lft forever preferred_lft forever
[root@snort ~]#
```

Con esto pudo reconocer la tarjeta de red nueva. Tras verificar que la direcciones MAC son las mismas que reporta el anfitrión. Procedí a hacer la configuración de la tarjeta Ethernet.

```
[root@snort ~]# cp /etc/sysconfig/network-scripts/ifcfg-eth0 /etc/sysconfig/network-
scripts/ifcfg-eth1
[root@snort ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth1
```

```
.
.
.
DEVICE=eth1
HWADDR=00:0c:29:82:99:0a
TYPE=Ethernet
#UUID=560042d0-7830-4e92-a35a-54fd95659fc1
ONBOOT=yes
NM_CONTROLLED=yes
BOOTPROTO=yes
#IPADDR=172.16.43.61
#NETMASK=255.255.255.0
#NETWORK=172.16.43.0
#GATEWAY=172.16.43.254

IPV6INIT=no
USERCTL=no
~
~
~
~
~
~
~
~
~
"/etc/sysconfig/network-scripts/ifcfg-eth1" 14L, 249C
.
.
.
[root@snort ~]# service network restart
Shutting down interface eth0:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:  Determining if ip address 172.16.43.60 is already in use
for device eth0...
                                                           [  OK  ]
Bringing up interface eth1:                                [  OK  ]
```

Una vez hecho tenemos todo listo para generar nuestro archivo snort.conf como ya les he explicado los operadores y las variables únicamente plasmaré la configuración.

```
[root@snort ~]# cat /etc/snort/snort.conf
##################################################
#
#Variables generales
#
##################################################

ipvar HOME_NET [172.16.43.0/24,192.168.1.0/24,192.168.0.0/24]

ipvar EXTERNAL_NET !$HOME_NET

ipvar DNS_SERVERS $HOME_NET

ipvar SMTP_SERVERS $HOME_NET

ipvar HTTP_SERVERS $HOME_NET

ipvar SQL_SERVERS $HOME_NET
```

```
ipvar TELNET_SERVERS $HOME_NET

ipvar SSH_SERVERS $HOME_NET

ipvar FTP_SERVERS $HOME_NET

ipvar SIP_SERVERS $HOME_NET

portvar HTTP_PORTS
[36,80,81,82,83,84,85,86,87,88,89,90,311,383,555,591,593,631,801,808,818,901,972,1158,1
220,1414,1533,1741,1830,1942,2231,2301,2381,2578,2809,2980,3029,3037,3057,3128,3443,370
2,4000,4343,4848,5000,5117,5250,5600,6080,6173,6988,7000,7001,7071,7144,7145,7510,7770,
7777,7778,7779,8000,8008,8014,8028,8080,8081,8082,8085,8088,8090,8118,8123,8180,8181,82
22,8243,8280,8300,8333,8344,8500,8509,8800,8888,8899,8983,9000,9060,9080,9090,9091,9111
,9290,9443,9999,10000,11371,12601,13014,15489,29991,33300,34412,34443,34444,41080,44449
,50000,50002,51423,53331,55252,55555,56712]

portvar SHELLCODE_PORTS !80

portvar ORACLE_PORTS 1024:

portvar SSH_PORTS 22

portvar FTP_PORTS [21,2100,3535]

portvar SIP_PORTS [5060,5061,5600]

portvar FILE_DATA_PORTS [$HTTP_PORTS,110,143]

portvar GTP_PORTS [2123,2152,3386]

# Variables propietarias de SNORT no remover o modificar
ipvar AIM_SERVERS
[64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/24,205.188.3.0/2
4,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.153.0/24,205.188.179.0/24,205.18
8.248.0/24]

##################################################
#
# Variables de rutas
#
##################################################
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules
var RULE_COMMUNITY_PATH /etc/snort/community-rules

var WHITE_LIST_PATH /etc/snort/rules
var BLACK_LIST_PATH /etc/snort/rules

##################################################
#
# Variables de decoders
#
##################################################

config disable_decode_alerts
config disable_tcpopt_experimental_alerts
config disable_tcpopt_obsolete_alerts
config disable_tcpopt_ttcp_alerts
```

```
config disable_tcpopt_alerts
config disable_ipopt_alerts
config checksum_mode: all

##################################################
#
#Configuración del motor de detección.
#
##################################################

config pcre_match_limit: 3500
config pcre_match_limit_recursion: 1500
config detection: search-method ac-split search-optimize max-pattern-len 20
config event_queue: max_queue 8 log 5 order_events content_length

##################################################
#
#Configuración de librerías dinámicas.
#
##################################################


#config enable_gtp

config paf_max: 16000


dynamicpreprocessor directory /usr/local/lib/snort_dynamicpreprocessor/
dynamicengine /usr/local/lib/snort_dynamicengine/libsf_engine.so
dynamicdetection directory /usr/local/lib/snort_dynamicrules

##################################################
#
#Configuración de preprocesadores.
#
##################################################

preprocessor normalize_ip4
preprocessor normalize_tcp: ips ecn stream
preprocessor normalize_icmp4
preprocessor normalize_ip6
preprocessor normalize_icmp6
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10
min_fragment_length 100 timeout 180

preprocessor stream5_global: track_tcp yes, \
   track_udp yes, \
   track_icmp no, \
   max_tcp 262144, \
   max_udp 131072, \
   max_active_responses 2, \
   min_response_seconds 5
preprocessor stream5_tcp: policy windows, detect_anomalies, require_3whs 180, \
   overlap_limit 10, small_segments 3 bytes 150, timeout 180, \
    ports client 21 22 23 25 42 53 70 79 109 110 111 113 119 135 136 137 139 143 \
        161 445 513 514 587 593 691 1433 1521 1741 2100 3306 6070 6665 6666 6667 6668
6669 \
        7000 8181 32770 32771 32772 32773 32774 32775 32776 32777 32778 32779, \
    ports both 36 80 81 82 83 84 85 86 87 88 89 90 110 311 383 443 465 563 555 591 593
631 636 801 808 818 901 972 989 992 993 994 995 1158 1220 1414 1533 1741 1830 1942 2231
```

```
2301 2381 2578 2809 2980 3029 3037 3057 3128 3443 3702 4000 4343 4848 5000 5117 5250
5600 6080 6173 6988 7907 7000 7001 7071 7144 7145 7510 7802 7770 7777 7778 7779 \
        7801 7900 7901 7902 7903 7904 7905 7906 7908 7909 7910 7911 7912 7913 7914 7915
7916 \
        7917 7918 7919 7920 8000 8008 8014 8028 8080 8081 8082 8085 8088 8090 8118 8123
8180 8181 8222 8243 8280 8300 8333 8344 8500 8509 8800 8888 8899 8983 9000 9060 9080
9090 9091 9111 9290 9443 9999 10000 11371 12601 13014 15489 29991 33300 34412 34443
34444 41080 44449 50000 50002 51423 53331 55252 55555 56712
preprocessor stream5_udp: timeout 180

preprocessor http_inspect: global iis_unicode_map unicode.map 1252 compress_depth 65535
decompress_depth 65535
preprocessor http_inspect_server: server default \
    http_methods { GET POST PUT SEARCH MKCOL COPY MOVE LOCK UNLOCK NOTIFY POLL BCOPY
BDELETE BMOVE LINK UNLINK OPTIONS HEAD DELETE TRACE TRACK CONNECT SOURCE SUBSCRIBE
UNSUBSCRIBE PROPFIND PROPPATCH BPROPFIND BPROPPATCH RPC_CONNECT PROXY_SUCCESS BITS_POST
CCM_POST SMS_POST RPC_IN_DATA RPC_OUT_DATA RPC_ECHO_DATA } \
    chunk_length 500000 \
    server_flow_depth 0 \
    client_flow_depth 0 \
    post_depth 65495 \
    oversize_dir_length 500 \
    max_header_length 750 \
    max_headers 100 \
    max_spaces 200 \
    small_chunk_length { 10 5 } \
    ports { 36 80 81 82 83 84 85 86 87 88 89 90 311 383 555 591 593 631 801 808 818 901
972 1158 1220 1414 1533 1741 1830 1942 2231 2301 2381 2578 2809 2980 3029 3037 3057
3128 3443 3702 4000 4343 4848 5000 5117 5250 5600 6080 6173 6988 7000 7001 7071 7144
7145 7510 7770 7777 7778 7779 8000 8008 8014 8028 8080 8081 8082 8085 8088 8090 8118
8123 8180 8181 8222 8243 8280 8300 8333 8344 8500 8509 8800 8888 8899 8983 9000 9060
9080 9090 9091 9111 9290 9443 9999 10000 11371 12601 13014 15489 29991 33300 34412
34443 34444 41080 44449 50000 50002 51423 53331 55252 55555 56712 } \
    non_rfc_char { 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 } \
    enable_cookie \
    extended_response_inspection \
    inspect_gzip \
    normalize_utf \
    unlimited_decompress \
    normalize_javascript \
    apache_whitespace no \
    ascii no \
    bare_byte no \
    directory no \
    double_decode no \
    iis_backslash no \
    iis_delimiter no \
    iis_unicode no \
    multi_slash no \
    utf_8 no \
    u_encode yes \
    webroot no

preprocessor rpc_decode: 111 32770 32771 32772 32773 32774 32775 32776 32777 32778
32779 no_alert_multiple_requests no_alert_large_fragments no_alert_incomplete

preprocessor bo

preprocessor ftp_telnet: global inspection_type stateful encrypted_traffic no
check_encrypted
preprocessor ftp_telnet_protocol: telnet \
```

```
    ayt_attack_thresh 20 \
    normalize ports { 23 } \
    detect_anomalies
preprocessor ftp_telnet_protocol: ftp server default \
    def_max_param_len 100 \
    ports { 21 2100 3535 } \
    telnet_cmds yes \
    ignore_telnet_erase_cmds yes \
    ftp_cmds { ABOR ACCT ADAT ALLO APPE AUTH CCC CDUP } \
    ftp_cmds { CEL CLNT CMD CONF CWD DELE ENC EPRT } \
    ftp_cmds { EPSV ESTA ESTP FEAT HELP LANG LIST LPRT } \
    ftp_cmds { LPSV MACB MAIL MDTM MIC MKD MLSD MLST } \
    ftp_cmds { MODE NLST NOOP OPTS PASS PASV PBSZ PORT } \
    ftp_cmds { PROT PWD QUIT REIN REST RETR RMD RNFR } \
    ftp_cmds { RNTO SDUP SITE SIZE SMNT STAT STOR STOU } \
    ftp_cmds { STRU SYST TEST TYPE USER XCUP XCRC XCWD } \
    ftp_cmds { XMAS XMD5 XMKD XPWD XRCP XRMD XRSQ XSEM } \
    ftp_cmds { XSEN XSHA1 XSHA256 } \
    alt_max_param_len 0 { ABOR CCC CDUP ESTA FEAT LPSV NOOP PASV PWD QUIT REIN STOU
SYST XCUP XPWD } \
    alt_max_param_len 200 { ALLO APPE CMD HELP NLST RETR RNFR STOR STOU XMKD } \
    alt_max_param_len 256 { CWD RNTO } \
    alt_max_param_len 400 { PORT } \
    alt_max_param_len 512 { SIZE } \
    chk_str_fmt { ACCT ADAT ALLO APPE AUTH CEL CLNT CMD } \
    chk_str_fmt { CONF CWD DELE ENC EPRT EPSV ESTP HELP } \
    chk_str_fmt { LANG LIST LPRT MACB MAIL MDTM MIC MKD } \
    chk_str_fmt { MLSD MLST MODE NLST OPTS PASS PBSZ PORT } \
    chk_str_fmt { PROT REST RETR RMD RNFR RNTO SDUP SITE } \
    chk_str_fmt { SIZE SMNT STAT STOR STRU TEST TYPE USER } \
    chk_str_fmt { XCRC XCWD XMAS XMD5 XMKD XRCP XRMD XRSQ } \
    chk_str_fmt { XSEM XSEN XSHA1 XSHA256 } \
    cmd_validity ALLO < int [ char R int ] > \
    cmd_validity EPSV < [ { char 12 | char A char L char L } ] > \
    cmd_validity MACB < string > \
    cmd_validity MDTM < [ date nnnnnnnnnnnnnn[.n[n[n]]] ] string > \
    cmd_validity MODE < char ASBCZ > \
    cmd_validity PORT < host_port > \
    cmd_validity PROT < char CSEP > \
    cmd_validity STRU < char FRPO [ string ] > \
    cmd_validity TYPE < { char AE [ char NTC ] | char I | char L [ number ] } >
preprocessor ftp_telnet_protocol: ftp client default \
    max_resp_len 256 \
    bounce yes \
    ignore_telnet_erase_cmds yes \
    telnet_cmds yes


preprocessor smtp: ports { 25 465 587 691 } \
    inspection_type stateful \
    b64_decode_depth 0 \
    qp_decode_depth 0 \
    bitenc_decode_depth 0 \
    uu_decode_depth 0 \
    log_mailfrom \
    log_rcptto \
    log_filename \
    log_email_hdrs \
    normalize cmds \
    normalize_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM ESND ESOM ETRN
EVFY } \
```

```
    normalize_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET SAML SEND
SOML } \
    normalize_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT X-DRCP X-ERCP X-
EXCH50 } \
    normalize_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE
XSTA XTRN XUSR } \
    max_command_line_len 512 \
    max_header_line_len 1000 \
    max_response_line_len 512 \
    alt_max_command_line_len 260 { MAIL } \
    alt_max_command_line_len 300 { RCPT } \
    alt_max_command_line_len 500 { HELP HELO ETRN EHLO } \
    alt_max_command_line_len 255 { EXPN VRFY ATRN SIZE BDAT DEBUG EMAL ESAM ESND ESOM
EVFY IDENT NOOP RSET } \
    alt_max_command_line_len 246 { SEND SAML SOML AUTH TURN ETRN DATA RSET QUIT ONEX
QUEU STARTTLS TICK TIME TURNME VERB X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN
XLICENSE XQUE XSTA XTRN XUSR } \
    valid_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM ESND ESOM ETRN EVFY
} \
    valid_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET SAML SEND SOML
} \
    valid_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT X-DRCP X-ERCP X-EXCH50
} \
    valid_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA
XTRN XUSR } \
    xlink2state { enabled }

preprocessor ssh: server_ports { 22 } \
                  autodetect \
                  max_client_bytes 19600 \
                  max_encrypted_packets 20 \
                  max_server_version_len 100 \
                  enable_respoverflow enable_ssh1crc32 \
                  enable_srvoverflow enable_protomismatch


preprocessor dcerpc2: memcap 102400, events [co ]
preprocessor dcerpc2_server: default, policy WinXP, \
    detect [smb [139,445], tcp 135, udp 135, rpc-over-http-server 593], \
    autodetect [tcp 1025:, udp 1025:, rpc-over-http-server 1025:], \
    smb_max_chain 3, smb_invalid_shares ["C$", "D$", "ADMIN$"]

preprocessor dns: ports { 53 } enable_rdata_overflow

preprocessor ssl: ports { 443 465 563 636 989 992 993 994 995 5061 7801 7802 7900 7901
7902 7903 7904 7905 7906 7907 7908 7909 7910 7911 7912 7913 7914 7915 7916 7917 7918
7919 7920 }, trustservers, noinspect_encrypted

preprocessor sensitive_data: alert_threshold 25

preprocessor sip: max_sessions 40000, \
   ports { 5060 5061 5600 }, \
   methods { invite \
             cancel \
             ack \
             bye \
             register \
             options \
             refer \
             subscribe \
             update \
```

```
                join \
                info \
                message \
                notify \
                benotify \
                do \
                qauth \
                sprack \
                publish \
                service \
                unsubscribe \
                prack }, \
    max_uri_len 512, \
    max_call_id_len 80, \
    max_requestName_len 20, \
    max_from_len 256, \
    max_to_len 256, \
    max_via_len 1024, \
    max_contact_len 512, \
    max_content_len 2048

preprocessor imap: \
    ports { 143 } \
    b64_decode_depth 0 \
    qp_decode_depth 0 \
    bitenc_decode_depth 0 \
    uu_decode_depth 0

preprocessor pop: \
    ports { 110 } \
    b64_decode_depth 0 \
    qp_decode_depth 0 \
    bitenc_decode_depth 0 \
    uu_decode_depth 0

preprocessor modbus: ports { 502 }

preprocessor dnp3: ports { 20000 } \
    memcap 262144 \
    check_crc

preprocessor reputation: \
    memcap 500, \
    priority whitelist, \
    nested_ip inner, \
    whitelist $WHITE_LIST_PATH/white_list.rules, \
    blacklist $BLACK_LIST_PATH/black_list.rules

##################################################
#
#Configuración de plugins de salida.
#
##################################################

output log_tcpdump: /var/log/snort/snort.log
#output alert_csv: /var/log/alertas.csv default

# Estas son librerías necesarias de snort no se deben de eliminar.
include classification.config
include reference.config
##################################################
```

```
#
#Configuración de reglas.
#
##################################################

#include $RULE_COMMUNITY_PATH/community.rules

include $RULE_PATH/local.rules

include $RULE_PATH/file-identify.rules
include $RULE_PATH/app-detect.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/blacklist.rules
include $RULE_PATH/botnet-cnc.rules
include $RULE_PATH/browser-chrome.rules
include $RULE_PATH/browser-firefox.rules
include $RULE_PATH/browser-ie.rules
include $RULE_PATH/browser-other.rules
include $RULE_PATH/browser-plugins.rules
include $RULE_PATH/browser-webkit.rules
include $RULE_PATH/chat.rules
include $RULE_PATH/content-replace.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/experimental.rules
include $RULE_PATH/exploit-kit.rules
include $RULE_PATH/exploit.rules
include $RULE_PATH/file-executable.rules
include $RULE_PATH/file-flash.rules
include $RULE_PATH/file-image.rules
include $RULE_PATH/file-java.rules
include $RULE_PATH/file-multimedia.rules
include $RULE_PATH/file-office.rules
include $RULE_PATH/file-other.rules
include $RULE_PATH/file-pdf.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/icmp-info.rules
include $RULE_PATH/icmp.rules
include $RULE_PATH/imap.rules
include $RULE_PATH/indicator-compromise.rules
include $RULE_PATH/indicator-obfuscation.rules
include $RULE_PATH/indicator-scan.rules
include $RULE_PATH/indicator-shellcode.rules
include $RULE_PATH/info.rules
include $RULE_PATH/malware-backdoor.rules
include $RULE_PATH/malware-cnc.rules
include $RULE_PATH/malware-other.rules
include $RULE_PATH/malware-tools.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/multimedia.rules
include $RULE_PATH/mysql.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/nntp.rules
include $RULE_PATH/oracle.rules
include $RULE_PATH/os-linux.rules
include $RULE_PATH/os-mobile.rules
include $RULE_PATH/os-other.rules
```

```
include $RULE_PATH/os-solaris.rules
include $RULE_PATH/os-windows.rules
include $RULE_PATH/other-ids.rules
include $RULE_PATH/p2p.rules
include $RULE_PATH/phishing-spam.rules
include $RULE_PATH/policy-multimedia.rules
include $RULE_PATH/policy-other.rules
include $RULE_PATH/policy.rules
include $RULE_PATH/policy-social.rules
include $RULE_PATH/policy-spam.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules
include $RULE_PATH/protocol-dns.rules
include $RULE_PATH/protocol-finger.rules
include $RULE_PATH/protocol-ftp.rules
include $RULE_PATH/protocol-icmp.rules
include $RULE_PATH/protocol-imap.rules
include $RULE_PATH/protocol-nntp.rules
include $RULE_PATH/protocol-other.rules
include $RULE_PATH/protocol-pop.rules
include $RULE_PATH/protocol-rpc.rules
include $RULE_PATH/protocol-scada.rules
include $RULE_PATH/protocol-services.rules
include $RULE_PATH/protocol-snmp.rules
include $RULE_PATH/protocol-telnet.rules
include $RULE_PATH/protocol-tftp.rules
include $RULE_PATH/protocol-voip.rules
include $RULE_PATH/pua-adware.rules
include $RULE_PATH/pua-other.rules
include $RULE_PATH/pua-p2p.rules
include $RULE_PATH/pua-toolbars.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/scada.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/server-apache.rules
include $RULE_PATH/server-iis.rules
include $RULE_PATH/server-mail.rules
include $RULE_PATH/server-mssql.rules
include $RULE_PATH/server-mysql.rules
include $RULE_PATH/server-oracle.rules
include $RULE_PATH/server-other.rules
include $RULE_PATH/server-samba.rules
include $RULE_PATH/server-webapp.rules
include $RULE_PATH/shellcode.rules
include $RULE_PATH/smtp.rules
include $RULE_PATH/snmp.rules
include $RULE_PATH/specific-threats.rules
include $RULE_PATH/spyware-put.rules
include $RULE_PATH/sql.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/tftp.rules
include $RULE_PATH/virus.rules
include $RULE_PATH/voip.rules
include $RULE_PATH/web-activex.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-client.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-iis.rules
```

```
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-php.rules
include $RULE_PATH/x11.rules

include $PREPROC_RULE_PATH/preprocessor.rules
include $PREPROC_RULE_PATH/decoder.rules
include $PREPROC_RULE_PATH/sensitive-data.rules

#include $SO_RULE_PATH/browser-ie.rules
#include $SO_RULE_PATH/browser-other.rules
#include $SO_RULE_PATH/browser-plugins.rules
#include $SO_RULE_PATH/exploit-kit.rules
#include $SO_RULE_PATH/file-executable.rules
#include $SO_RULE_PATH/file-flash.rules
#include $SO_RULE_PATH/file-image.rules
#include $SO_RULE_PATH/file-java.rules
#include $SO_RULE_PATH/file-multimedia.rules
#include $SO_RULE_PATH/file-office.rules
#include $SO_RULE_PATH/file-other.rules
#include $SO_RULE_PATH/file-pdf.rules
#include $SO_RULE_PATH/indicator-shellcode.rules
#include $SO_RULE_PATH/malware-cnc.rules
#include $SO_RULE_PATH/malware-other.rules
#include $SO_RULE_PATH/netbios.rules
#include $SO_RULE_PATH/os-linux.rules
#include $SO_RULE_PATH/os-other.rules
#include $SO_RULE_PATH/os-windows.rules
#include $SO_RULE_PATH/policy-social.rules
#include $SO_RULE_PATH/protocol-dns.rules
#include $SO_RULE_PATH/protocol-icmp.rules
#include $SO_RULE_PATH/protocol-nntp.rules
#include $SO_RULE_PATH/protocol-other.rules
#include $SO_RULE_PATH/protocol-snmp.rules
#include $SO_RULE_PATH/protocol-voip.rules
#include $SO_RULE_PATH/pua-p2p.rules
#include $SO_RULE_PATH/server-apache.rules
#include $SO_RULE_PATH/server-iis.rules
#include $SO_RULE_PATH/server-mail.rules
#include $SO_RULE_PATH/server-mysql.rules
#include $SO_RULE_PATH/server-oracle.rules
#include $SO_RULE_PATH/server-other.rules
#include $SO_RULE_PATH/server-webapp.rules

#include $SO_RULE_PATH/bad-traffic.rules
#include $SO_RULE_PATH/browser-ie.rules
#include $SO_RULE_PATH/chat.rules
#include $SO_RULE_PATH/dos.rules
#include $SO_RULE_PATH/exploit.rules
#include $SO_RULE_PATH/file-flash.rules
#include $SO_RULE_PATH/icmp.rules
#include $SO_RULE_PATH/imap.rules
#include $SO_RULE_PATH/misc.rules
#include $SO_RULE_PATH/multimedia.rules
#include $SO_RULE_PATH/netbios.rules
#include $SO_RULE_PATH/nntp.rules
#include $SO_RULE_PATH/p2p.rules
#include $SO_RULE_PATH/smtp.rules
#include $SO_RULE_PATH/snmp.rules
#include $SO_RULE_PATH/specific-threats.rules
#include $SO_RULE_PATH/web-activex.rules
#include $SO_RULE_PATH/web-client.rules
```

```
#include $SO_RULE_PATH/web-iis.rules
#include $SO_RULE_PATH/web-misc.rules

include threshold.conf
[root@snort ~]#
```

Se puede revisar la configuración del archivo que hemos generado a través de SNORT de la siguiente forma:

```
[root@snort ~]# snort -c /etc/snort/snort.conf -T
Running in Test mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
.
.
.
++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
5590 Snort rules read
    5172 detection rules
    150 decoder rules
    268 preprocessor rules
5590 Option Chains linked into 238 Chain Headers
0 Dynamic rules
++++++++++++++++++++++++++++++++++++++++++++++++++

+-------------------[Rule Port Counts]---------------------------------------
|             tcp     udp    icmp      ip
|     src    1743      15       0       0
|     dst    2694     610       0       0
|     any     524       2       3       0
|      nc     436       0       0       0
|     s+d       1       1       0       0
+------------------------------------------------------------------------------
.
.
.
[ Port Based Pattern Matching Memory ]
+- [ Aho-Corasick Summary ] ------------------------------------
| Storage Format    : Full-Q
| Finite Automaton  : DFA
| Alphabet Size     : 256 Chars
| Sizeof State      : Variable (1,2,4 bytes)
| Instances         : 168
|     1 byte states : 158
|     2 byte states : 10
|     4 byte states : 0
| Characters        : 94500
| States            : 72607
| Transitions       : 7855830
| State Density     : 42.3%
| Patterns          : 5202
| Match States      : 5816
| Memory (MB)       : 37.48
|   Patterns        : 0.58
|   Match Lists     : 1.26
```

```
|   DFA
|     1 byte states : 0.97
|     2 byte states : 34.38
|     4 byte states : 0.00
+-------------------------------------------------------------------
[ Number of patterns truncated to 20 bytes: 319 ]

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.6.2 GRE (Build 77)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.4.0
           Using PCRE version: 7.8 2008-09-05
           Using ZLIB version: 1.2.3

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.1  <Build 1>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>

Snort successfully validated the configuration!
Snort exiting
[root@snort ~]#
```

Lo que nos indica que la configuración es correcta. Esto concluye la configuración de SNORT procederemos a arrancarlo.

2.5. Requisitos previos al arranque

Como ya hemos mencionado SNORT es un IDS muy versátil que tiene muchas posibilidades de ejecución. Al ser un proyecto que viene del software libre en realidad no cuenta con una interface gráfica y sus modos de ejecución son invocados desde la línea de comandos o SHELL.

Aunque no es el único método de ejecución con el que cuenta el IDS ya que también se puede correr en modo de Demonio o en segundo plano.

2.5.1. Interface de red en modo promiscuo.

Un dato importante acerca de cualquier IDS o Sniffer es que estos necesitan de un puerto espejo. Pero en el lado del IDS o del Sniffer es que requieren que la tarjeta de red estén en modo promiscuo.

A que se refiere este modo. Bueno como vimos en el capítulo de red las direcciones IP obedecen a segmentos específicos lo que permite hacer más prácticas las comunicaciones. El problema de esto es que las tarjetas de red al tener un segmento especifico y con el modo de comunicación que tienen si captan paquetes que no corresponden a su segmento o a broadcast del segmento al que pertenecen desechan el paquete.

Como bien sabemos los IDS deben de capturar la mayor cantidad de tráfico posible del segmento asignado para poder detectar las intrusiones. El modo promiscuo configura a la tarjeta para que no deseche los paquetes que no le corresponden y captura todo.

Para poder arrancar una tarjeta en modo promiscuo, en nuestro caso la tarjeta que está dedicada a ello que es la eth1 se puede hacer de la siguiente forma:

```
[root@snort ~]# ifdown eth1
[root@snort ~]# ifup eth1 promisc
[root@snort ~]#
```

Otr forma y la que nos servirá en un futuro para hacer que esta tarjeta de monitoreo siempre este en modo promiscuo es desde el archivo de configuración de la tarjeta de red.

2.5.2. Operadores de SNORT

Como mencionamos los comandos de SHELL son la base de la ejecución de SNORT para ello tenemos que conocer los parámetros que le dan su versatilidad.

Parámetro	Descripción
-v	Sniffer manda los logs a pantalla como son IPs y cabeceros
-l	Packet Logger manda los resultados a un archivo de log
-c	NIDS arranca a SNORT utilizando el archivo de configuración
-d	Incluye en el log a todas las cabeceras TCP, UDP e ICMP del paquete
-e	Incluye en el log las cabeceras de la capa de enlace
-b	Manda al log información de tipo binario
-L	Se usa para especificar el archivo de log binario
-r	Se usa para leer archivos de log binarios
-h	Se usa para especificar la red de HOME (sin necesidad de archivo snort.conf)

Parámetro	Descripción
-i	Indica la interface de red que se utilizará para monitoreo
(and/or/not) host	Operadores lógicos para operaciones diversas relacionadas con host
src net	Especifica la red de origen e ignora su contenido
(dst/src) port	Especifica puertos a ignorar

Esto nos da como resultado que SNORT puede correr en diferentes modos.

2.6. Modos de ejecución de SNORT

Los modos de ejecución de SNORT son:

- Modo Sniffer
- Modo Packet Logger
- Modo Sistema de Detección de Intrusos.
- Modo Demonio

2.6.1. Modo Sniffer

Es el más simple de ejecución y este solo entrega lo que captura en pantalla, independientemente de que tipo de tráfico que es. La forma de invocarlo es bajo los siguientes ejemplos:

```
[root@snort ~]# snort -v -i eth1
Running in packet dump mode

        --== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to passive.
Acquiring network traffic from "eth1".
Decoding Ethernet

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.6.2 GRE (Build 77)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.4.0
           Using PCRE version: 7.8 2008-09-05
           Using ZLIB version: 1.2.3

Commencing packet processing (pid=1982)
11/25-20:53:32.182009 133.1.10.35:52012 -> 239.255.255.250:1900
UDP TTL:1 TOS:0x0 ID:29820 IpLen:20 DgmLen:161
Len: 133
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+


11/25-20:53:45.480910 62.128.100.43:443 -> 133.1.10.72:55369
TCP TTL:62 TOS:0x0 ID:25966 IpLen:20 DgmLen:52 DF
***A**** Seq: 0x52E6260A  Ack: 0x90AC4332  Win: 0x16A0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 587744100 441819749
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

```
11/25-20:53:45.657327 133.1.10.88:5353 -> 224.0.0.251:5353
UDP TTL:255 TOS:0x0 ID:2187 IpLen:20 DgmLen:228
Len: 200
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/25-20:53:46.657434 133.1.10.88:5353 -> 224.0.0.251:5353
UDP TTL:255 TOS:0x0 ID:2188 IpLen:20 DgmLen:228
Len: 200
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

^C*** Caught Int-Signal
11/25-20:53:48.651782 133.1.10.88:5353 -> 224.0.0.251:5353
UDP TTL:255 TOS:0x0 ID:2189 IpLen:20 DgmLen:228
Len: 200
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

===============================================================================
Run time for packet processing was 16.817305 seconds
Snort processed 57 packets.
Snort ran for 0 days 0 hours 0 minutes 16 seconds
   Pkts/sec:            3
===============================================================================
Memory usage summary:
  Total non-mmapped bytes (arena):       782336
  Bytes in mapped regions (hblkhd):      12640256
  Total allocated space (uordblks):      668880
  Total free space (fordblks):           113456
  Topmost releasable block (keepcost):   97104
===============================================================================
Packet I/O Totals:
   Received:           57
   Analyzed:           57 (100.000%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            0 (  0.000%)
   Injected:            0
===============================================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:           57 (100.000%)
       VLAN:            0 (  0.000%)
        IP4:           40 ( 70.175%)
       Frag:            0 (  0.000%)
       ICMP:            0 (  0.000%)
        UDP:           11 ( 19.298%)
        TCP:           29 ( 50.877%)
        IP6:            0 (  0.000%)
    IP6 Ext:            0 (  0.000%)
   IP6 Opts:            0 (  0.000%)
      Frag6:            0 (  0.000%)
      ICMP6:            0 (  0.000%)
       UDP6:            0 (  0.000%)
       TCP6:            0 (  0.000%)
     Teredo:            0 (  0.000%)
    ICMP-IP:            0 (  0.000%)
    IP4/IP4:            0 (  0.000%)
    IP4/IP6:            0 (  0.000%)
    IP6/IP4:            0 (  0.000%)
    IP6/IP6:            0 (  0.000%)
        GRE:            0 (  0.000%)
    GRE Eth:            0 (  0.000%)
   GRE VLAN:            0 (  0.000%)
```

```
    GRE IP4:            0 (  0.000%)
    GRE IP6:            0 (  0.000%)
GRE IP6 Ext:            0 (  0.000%)
   GRE PPTP:            0 (  0.000%)
    GRE ARP:            0 (  0.000%)
    GRE IPX:            0 (  0.000%)
   GRE Loop:            0 (  0.000%)
       MPLS:            0 (  0.000%)
        ARP:            0 (  0.000%)
        IPX:            0 (  0.000%)
   Eth Loop:            0 (  0.000%)
   Eth Disc:            0 (  0.000%)
   IP4 Disc:            0 (  0.000%)
   IP6 Disc:            0 (  0.000%)
   TCP Disc:            0 (  0.000%)
   UDP Disc:            0 (  0.000%)
  ICMP Disc:            0 (  0.000%)
All Discard:            0 (  0.000%)
      Other:           17 ( 29.825%)
Bad Chk Sum:            0 (  0.000%)
    Bad TTL:            0 (  0.000%)
     S5 G 1:            0 (  0.000%)
     S5 G 2:            0 (  0.000%)
      Total:           57
===============================================================================
Snort exiting
[root@snort ~]#
```

2.6.2. Modo Packet Logger

En este modo sigue sin distinguir tráfico y la diferencia es que entrega los paquetes dentro de un archivo.

```
[root@snort ~]# snort -b -L snort.log
```

2.6.3. Modo Sistema de Detección de Intrusos.

Este sin duda es el que nos interesa, este se usa corriendo SNORT a través del archivo snort.conf este es el modo que necesitamos para poder declarar que estamos haciendo detección de intrusos, es aquí donde usaremos el poder de las reglas y haremos funcionar a SNORT.

Un ejemplo muy claro y ejecutándolo directamente en la consola:

```
[root@snort ~]# snort -d -l /var/log/snort -c /etc/snort/snort.conf -i eth1
```

Se ejecutara SNORT de forma muy similar a la anterior con la diferencia de que nos dará un numero de proceso y si damos una típica cancelación de proceso ctrl + c nos devolverá algo como lo siguiente y nos retornará al SHELL

```
Acquiring network traffic from "eth1".
Reload thread starting...
Reload thread started, thread 0x7fe770287700 (35426)
Decoding Ethernet

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.6.2 GRE (Build 77)
```

```
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.4.0
           Using PCRE version: 7.8 2008-09-05
           Using ZLIB version: 1.2.3

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.1  <Build 1>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
Commencing packet processing (pid=35425)
^C*** Caught Int-Signal
===============================================================================
Run time for packet processing was 264.240256 seconds
Snort processed 4276 packets.
Snort ran for 0 days 0 hours 4 minutes 24 seconds
   Pkts/min:         1069
   Pkts/sec:           16
===============================================================================
Memory usage summary:
  Total non-mmapped bytes (arena):       202338304
  Bytes in mapped regions (hblkhd):      12640256
  Total allocated space (uordblks):      75015744
  Total free space (fordblks):           127322560
  Topmost releasable block (keepcost):   1024
===============================================================================
Packet I/O Totals:
   Received:         4275
   Analyzed:         4276 (100.023%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            0 (  0.000%)
   Injected:            0
===============================================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:         4286 (100.000%)
       VLAN:            0 (  0.000%)
        IP4:         3938 ( 91.881%)
       Frag:            0 (  0.000%)
       ICMP:            0 (  0.000%)
        UDP:          879 ( 20.509%)
        TCP:         2958 ( 69.015%)
        IP6:           15 (  0.350%)
    IP6 Ext:           15 (  0.350%)
   IP6 Opts:            0 (  0.000%)
      Frag6:            0 (  0.000%)
      ICMP6:            0 (  0.000%)
       UDP6:           15 (  0.350%)
       TCP6:            0 (  0.000%)
```

```
      Teredo:            0 (  0.000%)
     ICMP-IP:            0 (  0.000%)
     IP4/IP4:            0 (  0.000%)
     IP4/IP6:            0 (  0.000%)
     IP6/IP4:            0 (  0.000%)
     IP6/IP6:            0 (  0.000%)
         GRE:            0 (  0.000%)
     GRE Eth:            0 (  0.000%)
    GRE VLAN:            0 (  0.000%)
     GRE IP4:            0 (  0.000%)
     GRE IP6:            0 (  0.000%)
GRE IP6 Ext:             0 (  0.000%)
    GRE PPTP:            0 (  0.000%)
     GRE ARP:            0 (  0.000%)
     GRE IPX:            0 (  0.000%)
    GRE Loop:            0 (  0.000%)
        MPLS:            0 (  0.000%)
         ARP:           70 (  1.633%)
         IPX:            0 (  0.000%)
    Eth Loop:            0 (  0.000%)
    Eth Disc:            0 (  0.000%)
    IP4 Disc:           98 (  2.287%)
    IP6 Disc:            0 (  0.000%)
    TCP Disc:            0 (  0.000%)
    UDP Disc:            0 (  0.000%)
   ICMP Disc:            0 (  0.000%)
All Discard:            98 (  2.287%)
      Other:           266 (  6.206%)
Bad Chk Sum:             1 (  0.023%)
    Bad TTL:             0 (  0.000%)
     S5 G 1:             5 (  0.117%)
     S5 G 2:             5 (  0.117%)
      Total:          4286
===============================================================================
Action Stats:
     Alerts:            33 (  0.770%)
     Logged:            33 (  0.770%)
     Passed:             0 (  0.000%)
Limits:
      Match:             0
      Queue:             0
        Log:             0
      Event:             0
      Alert:             0
Verdicts:
      Allow:          2670 ( 62.456%)
      Block:             0 (  0.000%)
    Replace:             0 (  0.000%)
  Whitelist:          1606 ( 37.567%)
  Blacklist:             0 (  0.000%)
     Ignore:             0 (  0.000%)
===============================================================================
Frag3 statistics:
        Total Fragments: 0
      Frags Reassembled: 0
               Discards: 0
          Memory Faults: 0
               Timeouts: 0
               Overlaps: 0
              Anomalies: 0
                 Alerts: 0
```

```
                   Drops: 0
     FragTrackers Added: 0
    FragTrackers Dumped: 0
FragTrackers Auto Freed: 0
    Frag Nodes Inserted: 0
     Frag Nodes Deleted: 0
===============================================================================
Stream5 statistics:
            Total sessions: 207
              TCP sessions: 75
              UDP sessions: 132
             ICMP sessions: 0
               IP sessions: 0
                TCP Prunes: 0
                UDP Prunes: 0
               ICMP Prunes: 0
                 IP Prunes: 0
TCP StreamTrackers Created: 75
TCP StreamTrackers Deleted: 75
              TCP Timeouts: 0
              TCP Overlaps: 11
       TCP Segments Queued: 519
     TCP Segments Released: 519
       TCP Rebuilt Packets: 319
         TCP Segments Used: 462
              TCP Discards: 21
                  TCP Gaps: 66
      UDP Sessions Created: 132
      UDP Sessions Deleted: 132
              UDP Timeouts: 0
              UDP Discards: 0
                    Events: 30
           Internal Events: 0
           TCP Port Filter
                  Filtered: 0
                 Inspected: 0
                   Tracked: 2948
           UDP Port Filter
                  Filtered: 0
                 Inspected: 295
                   Tracked: 132
===============================================================================
HTTP Inspect - encodings (Note: stream-reassembled packets included):
    POST methods:                         0
    GET methods:                          66
    HTTP Request Headers extracted:       66
    HTTP Request Cookies extracted:       5
    Post parameters extracted:            0
    HTTP response Headers extracted:      29
    HTTP Response Cookies extracted:      0
    Unicode:                              2
    Double unicode:                       0
    Non-ASCII representable:              0
    Directory traversals:                 0
    Extra slashes ("//"):                 6
    Self-referencing paths ("./"):        0
    HTTP Response Gzip packets extracted: 9
    Gzip Compressed Data Processed:       28555.00
    Gzip Decompressed Data Processed:     73517.00
    Total packets processed:              824
===============================================================================
```

```
SMTP Preprocessor Statistics
  Total sessions                                   : 0
  Max concurrent sessions                          : 0
===============================================================================
dcerpc2 Preprocessor Statistics
  Total sessions: 0
===============================================================================
SSL Preprocessor:
   SSL packets decoded: 503
          Client Hello: 90
          Server Hello: 72
           Certificate: 10
           Server Done: 155
   Client Key Exchange: 28
   Server Key Exchange: 3
         Change Cipher: 180
              Finished: 0
    Client Application: 109
    Server Application: 49
                 Alert: 4
  Unrecognized records: 155
  Completed handshakes: 0
        Bad handshakes: 0
      Sessions ignored: 49
    Detection disabled: 2
===============================================================================
SIP Preprocessor Statistics
  Total sessions: 0
===============================================================================
Reputation Preprocessor Statistics
  Total Memory Allocated: 0
===============================================================================
Snort exiting
[root@snort ~]#
```

2.6.4. Modo Demonio

En modo demonio, se debe de declarar SNORT como un proceso que arranque al momento de que el sistema operativo arranque y sea supervisado. Esto se logra creando un usuario y creando un script.

Crearemos el usuario y dándoles propiedad en la carpeta de /etc/snort de la siguiente forma:

```
[root@snort ~]# useradd snort -d /var/log/snort/ -s /sbin/nologin -c SNORT_IDS
useradd: warning: the home directory already exists.
Not copying any file from skel directory into it.
[root@snort ~]# cd /etc/snort/
[root@snort snort]# chown -R snort:snort *
[root@snort snort]#
```

Ahora solo resta crear el script del demonio. Para este caso tome un demonio que plantea SNORT en su manual de instalación y le hice unas cuantas modificaciones.

```
[root@snort init.d]# cat snort
#!/bin/bash
#
# snort Start up the SNORT Intrusion Detection System daemon
#
# chkconfig: 2345 55 25
# description: SNORT is a Open Source Intrusion Detection System
```

```
# This service starts up the snort daemon.
#
# processname: snort
# pidfile: /var/run/snort_eth0.pid

### BEGIN INIT INFO
# Provides: snort
# Required-Start: $local_fs $network $syslog
# Required-Stop: $local_fs $syslog
# Should-Start: $syslog
# Should-Stop: $network $syslog
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: Start up the SNORT Intrusion Detection System daemon
# Description: SNORT is an application for Open Source Intrusion Detection.
# This service starts up the Snort IDS daemon.
### END INIT INFO

# source function library
. /etc/rc.d/init.d/functions

# pull in sysconfig settings
[ -f /etc/sysconfig/snort ] && . /etc/sysconfig/snort

RETVAL=0
prog="snort"
lockfile=/var/lock/subsys/$prog

# Some functions to make the below more readable
SNORTD=/usr/sbin/snort
#OPTIONS="-A fast -b -d -D -i eth0 -u snort -g snort -c /etc/snort/snort.conf -l
/var/log/snort"
#PID_FILE=/var/run/snort_eth0.pid

# Convert the /etc/sysconfig/snort settings to something snort can
# use on the startup line.

if [ "$ALERTMODE"X = "X" ]; then
      ALERTMODE=""
else
      ALERTMODE="-A $ALERTMODE"
fi

if [ "$USER"X = "X" ]; then
      USER="snort"
fi

if [ "$GROUP"X = "X" ]; then
      GROUP="snort"
fi

if [ "$BINARY_LOG"X = "1X" ]; then
      BINARY_LOG="-b"
else
      BINARY_LOG=""
fi

if [ "$LINK_LAYER"X = "1X" ]; then
      LINK_LAYER="-e"
else
      LINK_LAYER=""
```

```
fi

if [ "$CONF"X = "X" ]; then
      CONF="-c /etc/snort/snort.conf"
else
      CONF="-c $CONF"
fi

if [ "$INTERFACE"X = "X" ]; then
      INTERFACE="-i eth0"
      PID_FILE="/var/run/snort_eth0.pid"
else
      PID_FILE="/var/run/snort_$INTERFACE.pid"
      INTERFACE="-i $INTERFACE"
fi

if [ "$DUMP_APP"X = "1X" ]; then
      DUMP_APP="-d"
else
      DUMP_APP=""
fi

if [ "$NO_PACKET_LOG"X = "1X" ]; then
      NO_PACKET_LOG="-N"
else
      NO_PACKET_LOG=""
fi

if [ "$PRINT_INTERFACE"X = "1X" ]; then
      PRINT_INTERFACE="-I"
else
      PRINT_INTERFACE=""
fi

if [ "$PASS_FIRST"X = "1X" ]; then
      PASS_FIRST="-o"
else
      PASS_FIRST=""
fi

if [ "$LOGDIR"X = "X" ]; then
      LOGDIR=/var/log/snort
fi

# These are used by the 'stats' option
if [ "$SYSLOG"X = "X" ]; then
      SYSLOG=/var/log/messages
fi

if [ "$SECS"X = "X" ]; then
      SECS=5
fi

if [ ! "$BPFFILE"X = "X" ]; then
      BPFFILE="-F $BPFFILE"
fi

runlevel=$(set -- $(runlevel); eval "echo \$$#" )

start()
{
```

```
      [ -x $SNORTD ] || exit 5
      echo -n $"Taking out of corral $prog: "
      daemon --pidfile="/var/run/snort_eth1.pid" snort -d -D -i eth1 -u snort -g snort
-c /etc/snort/snort.conf -l /var/log/snort $BPF && succsess || failure
#daemon --pidfile=$PID_FILE $SNORTD $ALERTMODE $BINARY_LOG $LINK_LAYER $NO_PACKET_LOG
$DUMP_APP -D $PRINT_INTERFACE $INTERFACE -u $USER -g $GROUP $CONF -l $LOGDIR
$PASS_FIRST $BPFFILE $BPF && success || failure

      RETVAL=$?

      [ $RETVAL = 0 ] && touch $lockfile
      echo
      return $RETVAL
}

stop()
{
      echo -n $"Making bacon: $prog: "
      killproc $SNORTD
      if [ -e $PID_FILE ]; then
            chown -R $USER:$GROUP /var/run/snort_$INTERFACE.* && rm -f
/var/run/snort_$INTERFACE.pi*
      fi
      RETVAL=$?

      # if we are in halt or reboot runlevel kill all running sessions
      # so the TCP connections are closed cleanly
      if [ "x$runlevel" = x0 -o "x$runlevel" = x6 ] ; then
            trap '' TERM
            killall $prog 2>/dev/null
            trap TERM
      fi
      [ $RETVAL -eq 0 ] && rm -f $lockfile
      echo
      return $RETVAL
}

restart() {
      stop
      start
}

rh_status() {
      echo -n $"Porks are sniffing at: "
      status -p $PID_FILE $SNORTD
}

rh_status_q() {
      rh_status >/dev/null 2>&1
}

case "$1" in
      start)
            rh_status_q && exit 0
            start
            ;;
      stop)
            if ! rh_status_q; then
                  rm -f $lockfile
                  exit 0
            fi
```

```
            stop
            ;;
      restart)
            restart
            ;;
      status)
            rh_status
            RETVAL=$?
            if [ $RETVAL -eq 3 -a -f $lockfile ] ; then
                  RETVAL=2
            fi
            ;;
      *)
            echo $"Some farmers use this: $0 {start|stop|restart|status}"
            RETVAL=2
esac
exit $RETVAL
[root@snort init.d]#
```

Una vez hecho esto, hay que hacer el vínculo simbólico al script y darle permisos de ejecución.

```
[root@snort Demonio]# cp snort /etc/init.d/snort
[root@snort Demonio]# cd /etc/init.d/
[root@snort init.d]# ln -s /usr/local/bin/snort snort
[root@snort init.d]# chmod 700 snort
[root@snort init.d]# vi /etc/sysconfig/snort
[root@snort init.d]#
```

Ahora tenemos que dar permisos al usuario snort y lo hacemos de la siguiente manera.

```
[root@snort ~]# cd /etc/sysconfig/
[root@snort sysconfig]# chmod 700 snort
[root@snort sysconfig]# chown snort:snort snort
[root@snort sysconfig]# cd /usr/local/lib
[root@snort lib]# chown -R snort:snort snort*
[root@snort lib]# chown -R snort:snort snort_dynamic*
[root@snort lib]# chown -R 700 snort*
[root@snort lib]# cd /etc/sysconfig/
```

Por ultimo debemos de crear un archivo de configuración en la carpeta /etc/sysconfig donde podremos colocar las opciones para que SNORT arranque. Si observamos el script, en este archivo podremos especificar todas las opciones con las que cuenta SNORT para correr en modo demonio. Por ahora solo dejaré una configuración básica adaptada a las necesidades de la implementación de la siguiente forma:

```
[root@snort init.d]# cd /etc/sysconfig/
[root@snort sysconfig]# cat snort
# /etc/sysconfig/snort
# $Id: snort.sysconfig,v 1.8 2003/09/19 05:18:12 dwittenb Exp $
#### General Configuration
INTERFACE=eth1
CONF=/etc/snort/snort.conf
USER=snort
GROUP=snort
PASS_FIRST=0
#### Logging & Alerting
[root@snort sysconfig]#
```

Finalmente podemos confirmar si la configuración es correcta mediante el uso del comando service.

```
[root@snort sysconfig]# service snort start
Taking out of corral snort:                                [  OK  ]
[root@snort sysconfig]#
```

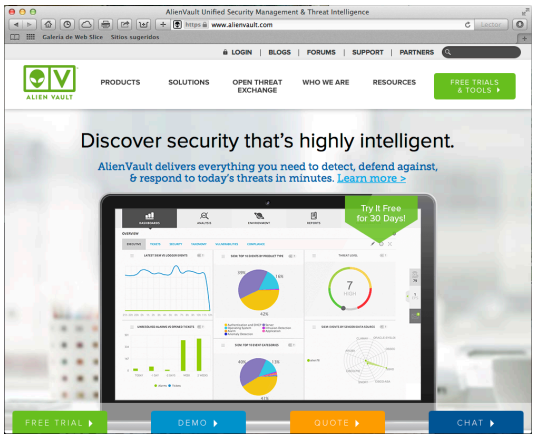
Este es el final de la puesta en marcha de SNORT y también de la parte técnica. Ahora entraremos en el mundo de la respuesta a incidentes que abordaremos de una forma muy superficial.

2.7. Respuesta a incidentes.

Algunas metodologías abordan de forma muy seria la respuesta a incidentes en el caso de esta escuela no es necesario hacer todo un proceso de tickets para poder responder a incidentes.

En los ambientes corporativos los IDS como SNORT se complementan con un correlacionador de eventos o SGSI. Estos elementos son parte de la era del "Big Data" en donde se tienen muchos datos y se analizan de forma inteligente a modo que se tenga la menor cantidad de alertas o falsos positivos, además de dar la capacidad de reaccionar de modo inteligente a posibles ataques de forma eficiente.

Un ejemplo que tiene integrado a SNORT como herramienta de detección de intrusos es USM de Alien Vault www.alienvault.com



Ellos aparte de darle un aspecto más grafico hacen la tarea de levantar tickets en caso de algún incidente.

Para nuestro caso es importante recalcar que no se cuenta con presupuesto por lo que se hace una pequeña capacitación a la persona responsable de sistemas de la escuela acerca de las alertas y esta entrega un reporte de incidentes una vez al mes a su dirección. Que no es otra cosa que un Excel con el listado de alertas encontradas. Abajo un ejemplo:

Fecha de Alerta	Fecha de solución	Nombre del incidente:	Dispositivo que lo ha detectado:	Origen(es) involucrado:	Destino(s) involucrados:	Puerto(s) destino:	Número de eventos que se hayan detectado:	Estatus
08/08/2014 01:44:20	08/08/2014 01:44:21	INDICATOR-OBFUSCATION select concat statement - possible sql injection obfuscation-BLOCKED	snort	96.44.189.101	192.168.1.73	80	1	Falso Positivo
08/08/2014 01:49:24	08/08/2014 01:49:24	SQL waitfor delay function - possible SQL injection attempt-BLOCKED	snort	96.47.226.22	192.168.1.73	80	1	Falso Positivo
08/08/2014 13:05:01	08/08/2014 14:18:46	Anomalous Port Activity	snort	218.77.79.43 116.10.191.173 116.10.191.213 116.10.191.198 116.10.191.199	192.168.x.x 148.223.91.x	22	1179	Resuelto
08/08/2014 14:07:05	08/08/2014 14:23:29	Anomalous Port Activity	snort	218.77.79.43 116.10.191.173 116.10.191.213 116.10.191.198 116.10.191.199	192.168.x.x 148.223.91.x	22	259	Resuelto
20/08/2014 18:56:12	20/08/2014 19:14:06	SERVER-OTHER Adobe Coldfusion viewexample.cfm access	snort	189.208.243.184 IP Address 189.208.243.184 (Websites Lookup) Reverse DNS wimax-cpe-189-208-243-184.mexdf.static.axtel.... ASN AS6503 ASN Owner Axtel, S.A.B. de C.V. ISP AXTEL Continent North America Country Code Flag (MX) Mexico Latitude / Longitude 25.6667 / -100.4	192.168.1.73	80	1	Resuelto

Capítulo 4

4. Conclusiones

Como hemos podido observar en este trabajo. No hacen falta grandes capitales para implementar soluciones de seguridad que puedan adaptarse a ambientes pequeños y escolares.

La implementación hecha en este colegio, nos deja ver que en cualquier lugar se pueden existen amenazas que pueden tener un riesgo muy alto para negocios medianos que pueden impactar de forma muy significativa su operación.

El uso de esta tecnología puede ayudar a reducir este riesgo hasta el punto de mitigarlo.

Los IDS no son herramientas que estén alejadas de la vida cotidiana, tampoco son elementos exclusivos de grandes industrias. El software libre, muchas veces despreciado, es la solución para negocios que no gozan del capital de las grandes empresas.

Los ambientes escolares, tienen una capacidad de re implementación muy dinámica comparada con las grandes empresas, lo que los vuelve altamente reactivos en caso de incidentes, además de que les brinda la facilidad de adoptar nuevas tecnologías en tiempos cortos.

El tiempo de implementación de este IDS fue relativamente corto. No así en ambientes corporativos que tienen que hacer procesos de hasta uno o dos años para realizar implementaciones tan complejas como las que involucran los IDS.

El aprendizaje técnico acerca de que es lo que sucede en la red, es importante a todos los niveles de organizaciones, ya que les permite brindar soluciones que dan como respuesta mayor seguridad para los alumnos y para el negocio.

La experiencia brindada a través de esta implementación hace visible la capacidad de implementaciones sencillas y baratas que solucionan grandes problemas.

Referencias y Apéndices

R

Referencias bibliográficas

Fly Team, Requirenmets Analysys from Busines View to Architecture, Editor Prentice hall PTR, 23 august 2002, ISBN 0-13-028228-6 ,page 496

Newcower Erick, Lomow Greg, Understending SOA with Web Services, Editor Addison Wesley Professional, December 14, 2004, ISBN 0-18086-0, Page 480

Krafzig Dirk, Banke Karl, Slama Dirk, Enterprise SOA: Service Oriented Architecture Best Practice, Editor Prentice Hall PTR, Nov 9 2004, ISBN o-13-146575-9, Page 408

Steplent Albin, The Art of Software Architecture: Design Methods and Techniques, Editor John Wiley & Sons ,2003, ISBN 047-122886-9, Page 312

Pallab Saha, Handbook of Enterprise Systems Architecture in Practice, Editor Britisa Calaloving Publication Data, 2007, ISBN 978-1-59904-189-6, Page 499-

McGovern james, Sims Oliver, Enterprise Services Oriented Architectures Concepts, Chanllenges, Recommendations, Editor Springer 2005, ISBN 10 1-4020-3704-4 page 434.

Albanese jason, sonnenreich Wes, Neywork Security Ilustrated, Mc graw Hill, 2004, ISBN 0-07-143355-4, page 409.

Gress Syn, Albanese Jason, Managing Cisco network Security second Edition, Editor Cisco Press, 2002, ISBN 1-913836-56-6
Page 786.

Bornett David, Groth David, Mc Bee Jim, Cabling The Complete Guide to Network Wiring Third Edition, Editor Sybex, ISBN 0-2821-4331-8, page 733-

Odom Wendell, McDonald Rick, Routers and Routing Basics CCNA 2 Companion Guide, Editor CISCO PRESS, 2006, ISBN 1-58713-166-8,page 473.

Referencias de Internet

Manual de desarrolladores de SNORT
http://manual.snort.org
(document web)
30 septiembre de 2014

System Administration, Networking, and Security Institute,
https://www2.sans.org/
(Document Web)
1 noviembre 2007

Internet Engineering Task Force DNS Extensions Working Group, Last Modified: 2007-07-25
http://www.ietf.org/html.charters/dnsext-charter.html
(Documento Web)
3 noviembre 2007

APÉNDICES

A: Glosario de Siglas

BGP
Protocolo de Entrada Limite

DHCP
Protocolo de Configuración Dinámica de Host

DNS
Servidor de Nombres de Servicio

FTP
Protocolo de transferencia de Archivos

HTTP

ISO
Organización Internacional de Normalización

ICMP
Protocolo de Acceso de Mensaje de Internet

IDS
Sistema de Detección de Intrusos

IT
Tecnología de Información

LAN
Red de Área Local

NTP
Protocolo de Adaptador de Red

NTU
Unidad de Terminación de Equipo

OSPF
Protocolo de Enrutamiento Jerárquico de Pasarela Interior

OMG
Grupo d e Gestión de Objetos

RAS
Servicios de Acceso Remoto

RIP
Protocolo de Información Directa

SGSI
Sistema de Gestión de la Seguridad de Información

SNMP
Protocolo de Administración Simple de Red

SOA
Arquitectura Orientada a Servicios

SONA
Arquitectura de Red Orientada a Servicios

SSH
Shell Seguro en Unix

SSL
Socket de la Capa de Seguridad protocolo de la capa de transporte

TCP
Protocolo de Control de Transmisión

V.35
Estándar de UIT-T que describe en producto síncrono

VPN
Red Virtual Privada

B: Glosario de Términos

Broadcast
Broadcast, en castellano difusión, es un modo de transmisión de información donde un nodo emisor envía información a una multitud de nodos receptores de manera simultánea, sin necesidad de reproducir la misma transmisión nodo por nodo.

Fast Ethernet
Cualquiera de las especificaciones de Ethernet de 100-Mbps.

Heurística
Se denomina heurística a la capacidad de un sistema para realizar de forma inmediata innovaciones positivas para sus fines. La capacidad heurística es un rasgo característico de los humanos, desde cuyo punto de vista puede describirse como *el arte y la ciencia del descubrimiento y de la invención* o de resolver problemas mediante la creatividad y el pensamiento lateral o pensamiento divergente.

HyperTerminal
Es un programa que usted puede utilizar para conectar Windows a otros ordenadores, sitios Telnet, tablero sistemas.

Infraestructura
Conjunto de tecnologías que componen la plataforma para el manejo de tecnologías de la información.

Interfaz
La interfaz es el lugar de la interacción, el espacio donde se desarrollan los intercambios.

Internetworking
Término general utilizado para referirse a la industria que ha surgido en torno de la cuestión de la conexión de redes entre sí. El término se puede referir a productos, procedimientos y tecnologías.

Loopback
El dispositivo de red loopback es un interfaz de red virtual que siempre representa al propio dispositivo independientemente de la dirección IP que se le haya asignado. El valor en IPv4 es 127.0.0.1. y ::1 para el caso de IPv6

Se utiliza en tareas de diagnóstico de conectividad y validez del protocolo de comunicación, así como para indicar que el destino del puntero o URL es el mismo host.

Metodología
La metodología es parte de una ciencia que estudia los métodos que en ella se emplea.

Negocio

Específicamente, negocio puede referirse a entidades individuales de la economía. En algunas jurisdicciones legales, tales entidades son reguladas por la ley para conducir las operaciones a favor de empresarios. Un negocio industrial es referido comúnmente como una industria

Nodo

En informática, un nodo es "Punto de intersección o unión de varios elementos que confluyen en el mismo lugar". Ejemplo: en una red de ordenadores cada una de las máquinas es un nodo, y si la red es Internet, cada servidor constituye también un nodo.

Organización
Las organizaciones son sistemas diseñados para lograr metas y objetivos por medio de los recursos humanos, tecnológicos y metodológicos. Están compuestas por subsistemas interrelacionados que cumplen funciones especializadas.

Organizar
Proceso de poner a dos o más personas a trabajar juntas, de manera estructurada, para alcanzar una meta concreta o un conjunto de metas.

Patch panel
Parche panel o un parche bahía es un panel, típicamente conectores, que alberga conexiones del cable.

Proceso
Es un conjunto de actividades o eventos que se realizan o suceden con un determinado fin.

Red de área local
LAN es la abreviatura de **Local Area Network** (Red de Área Local o simplemente Red Local). Una red local es la interconexión de varios ordenadores y periféricos. Su extensión está limitada físicamente a un edificio o a un entorno de unos pocos kilómetros. Su aplicación más extendida es la interconexión de ordenadores personales y estaciones de trabajo en oficinas, fábricas, etc; para compartir recursos e intercambiar datos y aplicaciones. En definitiva, permite que dos o más máquinas se comuniquen.

RFC 2196
Es una guía sobre el establecimiento de la seguridad informática políticas y procedimientos para los sitios que tienen sistemas a través de Internet (sin embargo, la información también debe ser útil a los lugares aún no conectadas a la Internet). La guía enumera las cuestiones y factores que un sitio debe tener en cuenta al configurar sus propias políticas. Se formula una serie de recomendaciones y proporciona discusiones de las esferas pertinentes.

Servicio
En economía y en marketing (mercadotecnia) un servicio es un conjunto de actividades que buscan responder a una o más necesidades de un cliente. Se define un marco en donde las actividades se desarrollarán con la idea de fijar una expectativa en el resultado de éstas. Es el equivalente no material de un bien. La presentación de un servicio no resulta en posesión, y así es como un servicio se diferencia de proveer un bien físico.

Troubleshooting
Procedimiento para solucionar problemas, solución de problemas, localización de problemas, localización de fallas, localización de averías

Usabilidad
Efectividad, eficiencia y satisfacción con la que un grupo específico de usuarios realiza una serie de tareas en un ambiente en particular.