**IPTables/IPSec/Openswan**

1. **Descripcion del proyecto**

Este documento describe la instalación y configuración típica de IPTables e IPSec ( Openswan) en una sucursal , decimos típica porque la configuración puede variar dependiendo de los servicios que se necesiten tener habilitados, pero técnicamente la configuración seria igual para todas.

* 1. **Objetivos del proyecto**

El objetivo consiste en brindar una capa mas de protección a las comunicaciones y accesos a las sucursales, mediante IPtables e IPSec con lo cual podemos filtrar desde y a donde queremos conectarnos ya que podemos filtrar a que servicios queremos que se conecten y desde que IP o rago de IP’s permitiremos el acceso.Tambien se contempla la posibilidad de utilizar en conjunto con IPtables e IPSec para encriptar los datos a nivel de TCP/IP desde la línea de cajas a nuestro servidor.

* 1. **SOLUCION**

La solución consiste en instalar el paquete iptables, paquete el cual está diponible en todas las distribuciones de GNU/Linux y el paquete openswan.

**2.Conceptos Básicos**

**2.1 Netfilter http://www.netfilter.org/**

**Netfilter** es un framework disponible en el kernel de Linux que permite interceptar y manipular paquetes de red. Dicho framework permite realizar el manejo de paquetes en diferentes estados del procesamiento. Netfilter es también el nombre que recibe el proyecto que se encarga de ofrecer herramientas libres para firewalls basados en Linux.

El componente más popular construido sobre *Netfilter* es **iptables**, una herramienta de firewall que permite no solamente filtrar paquetes, sino también realizar traducción de direcciones de red (NAT) para IPV4 o mantener registros de logs. El proyecto Netfilter no sólo ofrece componentes disponibles como módulos del kernel sino que también ofrece herramientas de userspace.

**iptables** es el nombre de la herramienta de userspace mediante la cual el administrador puede definir políticas de filtrado del tráfico que circula por la red. El nombre ***iptables*** se utiliza frecuentemente de forma errónea para referirse a toda la infraestructura ofrecida por el proyecto Netfilter. Sin embargo, el proyecto ofrece otros subsistemas independientes de **iptables** tales como el ***connection tracking system***o sistema de seguimiento de conexiones, que permite encolar paquetes para que sean tratados desde el userspace.***iptables*** es un software disponible en prácticamente todas las distribuciones de Linux actuales.

**2.2 Conceptos Básicos de Funcionamiento**

Iptablesnos permite definir reglas acerca de qué hacer con los paquetes de red. Las reglas se agrupan en *cadenas*: cada cadena es una lista ordenada de reglas. Las cadenas se agrupan en *tablas*: cada tabla está asociada con un tipo diferente de procesamiento de paquetes.

Cada regla especifica qué paquetes la cumplen (*match*) y un *objetivo* que indica qué hacer con el paquete si éste cumple la regla. Cada paquete de red que llega a una computadora o que se envía desde una computadora recorre por lo menos una cadena y cada regla de esa cadena se comprueba con el paquete. Si la regla cumple con el datagrama, el recorrido se detiene y el destino de la regla dicta lo que se debe hacer con el paquete. Si el paquete alcanza el fin de una cadena predefinida sin haberse correspondido con ninguna regla de la cadena, la *política* de destino de la cadena dicta qué hacer con el paquete. Si el paquete alcanza el fin de una cadena definida por el usuario sin haber cumplido ninguna regla de la cadena o si la cadena definida por el usuario está vacía, el recorrido continúa en la cadena que hizo la llamada (lo que se denomina *implicit target RETURN* o RETORNO de destino implícito). Solo las cadenas predefinidas tienen políticas.

**2.2.1 Cadenas ( CHAINS )**

Hay tres tablas ya incorporadas, cada una de las cuales contiene ciertas cadenas predefinidas. Es posible crear nuevas tablas mediante módulos de extensión. El administrador puede crear y eliminar cadenas definidas por usuarios dentro de cualquier tabla. Inicialmente, todas las cadenas están vacías y tienen una política de destino que permite que todos los paquetes pasen sin ser bloqueados o alterados.

* filtertable (*Tabla de filtros*) — Esta tabla es la responsable del filtrado (es decir, de bloquear o permitir que un paquete continúe su camino). Todos los paquetes pasan a través de la tabla de filtros. Contiene las siguientes cadenas predefinidas y cualquier paquete pasará por una de ellas:
  + ***INPUT****chain* (Cadena de ENTRADA) — Todos los paquetes destinados a este sistema atraviesan esta cadena (y por esto se la llama algunas veces *LOCAL\_INPUT* o *ENTRADA\_LOCAL*)
  + ***OUTPUT****chain* (Cadena de SALIDA) — Todos los paquetes creados por este sistema atraviesan esta cadena (a la que también se la conoce como *LOCAL\_OUTPUT* o *SALIDA\_LOCAL*)
  + ***FORWARD****chain* (Cadena de REDIRECCIÓN) — Todos los paquetes que meramente pasan por este sistema para ser enrutados a su destino recorren esta cadena.

**Ejemplo de reglas en sucursal 02**

**Chain INPUT (policy DROP)**

target prot opt source destination

DROP all -- 0.0.0.0/0 0.0.0.0/0 state INVALID

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x12/0x12 state NEW

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x3F/0x00

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x03/0x03

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x06/0x06

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x3F/0x37

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x05/0x05

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x11/0x01

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x18/0x08

DROP tcp -- 0.0.0.0/0 0.0.0.0/0 tcp flags:0x30/0x20

ACCEPT all -- 127.0.0.1 127.0.0.1

ACCEPT all -- 129.2.255.253 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0 state RELATED,ESTABLISHED

ACCEPT icmp -- 0.0.0.0/0 129.2.255.253 icmp type 8 state NEW,RELATED,ESTABLISHED

ACCEPT icmp -- 0.0.0.0/0 129.2.255.253 icmp type 0 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 0.0.0.0/0 129.2.255.253 tcp dpt:445 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 0.0.0.0/0 129.2.255.253 tcp dpt:139 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 0.0.0.0/0 129.2.255.253 tcp dpt:515 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 0.0.0.0/0 129.2.255.253 tcp dpts:5900:5901 state NEW,RELATED,ESTABLISHED

ACCEPT udp -- 0.0.0.0/0 129.2.255.253 udp dpt:53 state NEW,ESTABLISHED

ACCEPT udp -- 0.0.0.0/0 129.2.255.253 udp dpt:161 state NEW,ESTABLISHED

ACCEPT udp -- 0.0.0.0/0 129.2.255.253 udp dpts:137:138 state NEW,ESTABLISHED

ACCEPT udp -- 0.0.0.0/0 129.2.255.253 udp dpt:873 state NEW,ESTABLISHED

ACCEPT udp -- 0.0.0.0/0 129.2.255.253 udp dpts:67:68 state NEW,ESTABLISHED

ACCEPT all -- 10.10.10.0/24 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 172.16.7.32 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 172.16.7.10 129.2.255.253 tcp dpts:9005:9007 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 172.16.7.10 129.2.255.253 tcp dpt:1524 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 172.16.7.10 129.2.255.253 tcp dpts:2370:2371 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.99 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.252 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.97 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.253 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.98 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.250 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.249 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.145 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.104 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.102 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.251 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.94 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT all -- 129.2.102.254 129.2.255.253 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 128.2.108.103 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 128.2.107.111 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 128.2.105.121 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 128.2.108.213 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.57.0/24 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.3.85 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 128.2.101.161 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 129.2.0.0/16 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 192.168.2.20 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 130.93.254.1 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.7.13 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.3.54 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.3.130 129.2.255.253 tcp dpt:1521 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.8.16 129.2.255.253 tcp dpt:22 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.7.39 129.2.255.253 tcp dpt:22 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.7.40 129.2.255.253 tcp dpt:22 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.7.13 129.2.255.253 tcp dpt:22 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.7.32 129.2.255.253 tcp dpt:22 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.3.61 129.2.255.253 tcp spt:25 dpts:1024:65535 state ESTABLISHED

ACCEPT udp -- 128.2.254.180 129.2.255.253 udp spt:53 dpts:1024:65535

ACCEPT udp -- 172.16.3.50 129.2.255.253 udp spt:53 dpts:1024:65535

ACCEPT udp -- 130.93.253.240 129.2.255.253 udp spt:53 dpts:1024:65535

ACCEPT udp -- 128.2.254.180 129.2.255.253 udp dpt:123

ACCEPT tcp -- 129.2.0.0/16 129.2.255.253 tcp dpt:80 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 129.2.22.0/24 129.2.255.253 tcp dpts:20:21 state NEW

ACCEPT tcp -- 128.2.101.72 129.2.255.253 tcp dpts:20:21 state NEW

ACCEPT tcp -- 172.16.57.0/24 129.2.255.253 tcp dpt:10101 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 129.2.102.0/24 129.2.255.253 tcp dpt:10101 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 129.2.102.0/24 129.2.255.253 tcp dpt:9001 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 172.16.57.0/24 129.2.255.253 tcp dpt:6580 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.57.0/24 129.2.255.253 tcp dpt:7373 state NEW,ESTABLISHED

AUDITA-TRAFICO all -- 0.0.0.0/0 0.0.0.0/0

**Chain FORWARD (policy DROP)**

**target prot opt source destination**

**Chain OUTPUT (policy ACCEPT)**

**target prot opt source destination**

ACCEPT all -- 0.0.0.0/0 0.0.0.0/0 state NEW,RELATED,ESTABLISHED

**Chain AUDITA-TRAFICO (1 references)**

target prot opt source destination

LOG all -- 0.0.0.0/0 129.2.255.253 limit: avg 10/sec burst 5 LOG flags 0 level 2 prefix `DENEGADO '

DROP all -- 0.0.0.0/0 0.0.0.0/0

**Se adjunta el archivo de configuración de ejemplo de la sucursal 02:**

****

****

**2.3 IPSecOpenSWANhttps://www.openswan.org/projects/openswan/**

Los protocolos de **IPsec** actúan en la capa de red, la capa 3 del modelo OSI. Otros protocolos de seguridad como SSL, TLS y SSH operan de la capa de transporte (capa 4 del modelo OSI). Esto hace que IPsec sea más flexible, ya que puede ser utilizado para proteger protocolos de la capa 4, incluyendo TCP y UDP, los protocolos de capa de transporte más usados. Una ventaja importante de IPsec frente a SSL y otros métodos que operan en capas superiores, es que para que una aplicación pueda usar IPsec no hay que hacer ningún cambio, mientras que para usar SSL y otros protocolos de niveles superiores, las aplicaciones tienen que modificar su código.

IPsecconsta de dos protocolos que han sido desarrollados para proporcionar seguridad a nivel de paquete, tanto para IPV4 como para IPV6:

* **AuthenticationHeader** (AH) proporciona integridad, autenticación si se eligen los algoritmos criptográficos apropiados.
* **Encapsulating Security Payload** (ESP) proporciona confidencialidad y la opción altamente recomendable de autenticación y protección de integridad.

**2.3.1 Instalacion**

Como root ejecutamos:

**yuminstallopenswan o rpm –Uvhopenswan-version.rpm**

**2.3.2 Configuración Servidor o Gateway ( LEFT )**

Openswan consta de varios archivos de configuración y varios tipos de configuración valga la redundancia. Si bien cuando vemos el archivo de configuración parece bastante simple, se probaron varias alternativas para ajustarnos lo mejor posible a nuestra necesidades lo cual llevó su tiempo y fue complejo en cierta medida. Uno de los puntos importantes y críticos a la hora de establecer una conexión con otro nodo es el método de autenticación. Openswan dispone de lo que se llaman claves publicas y privadas que se utilizan para realizar el intercambio de varios datos entre los nodos y para luego realizar la encriptación de los paquetes a travez de nuestra red.

Básicamente tenemos 3 formas de claves, PSK, RSA y Certificados SSL, la topología de nuestra red es sencilla ya que nos vamos a estar conectando entre servidores dentro de nuestra propia red y no vamos a estar formando un túnel a travez de internet por lo cual nos va a facilitar un poco la configuración.

Para nuestro caso vamos a utilizar claves de RSA , pública y privada respectivamente, un servidor va a oficiar de Gateway o servidor al cual se conectan el resto de los nodos, y los nodos restantes o clientes por llamarlos de alguna forma.

**2.3.2.2 Ajustes Previos**

Antes de iniciar el servicio debemos ajustar algunos parámetros del kernel para que correcto funcionamiento de Openswan, a continuación mostramos dichos parámetros los cuales pueden ser agragados a un script como ser rc.local en /etc/ para que sean ejecutados al inicio del sistema o con la utilidad sysct y agregarlos en /etc/sysctl.conf con el formato net.ipv4.conf.all.secure\_redirects = 1

**echo 0 > /proc/sys/net/ipv4/conf/eth0/secure\_redirects**

**echo 0 > /proc/sys/net/ipv4/conf/eth0/send\_redirects**

**echo 0 > /proc/sys/net/ipv4/conf/eth0/accept\_redirects**

**echo 0 > /proc/sys/net/ipv4/conf/eth0/send\_redirects**

**echo 0 >/proc/sys/net/ipv4/conf/all/send\_redirects**

**echo 0 >/proc/sys/net/ipv4/conf/default/send\_redirects**

**echo 0> /proc/sys/net/ipv4/conf/lo/send\_redirects**

**echo 0 > /proc/sys/net/ipv4/conf/all/accept\_redirects**

**echo 0 > /proc/sys/net/ipv4/conf/default/accept\_redirects**

**echo 0 > /proc/sys/net/ipv4/conf/lo/accept\_redirects**

**echo 0 > /proc/sys/net/ipv4/conf/eth0/accept\_redirects**

**net/ipv4/ip\_forward = 1**

**2.3.2.3 Creacion de la base NSS**

Una vez ajustado los parámetros del kernel procedemos a la creación de la base de NSS ( CertificateDatabase ) donde Ipsec va a guardar la clave pública y privada. Para ello necesitamos el paqutenss-tools instalado, verificamos si ya lo tenemos instalado en el sistema , de lo contrario habrá que instalarlo.

**[root@slnxapp3 ipsec.d]# rpm -qa|grepnss-tool**

**nss-tools-3.12.10-16.el6.x86\_64**

**[root@slnxapp3 ipsec.d]#**

Una vez instalado el paquete procedemos de la siguiente manera, la utilidad le va a pedir que introduzca un clave:

**[root@slnxapp3 ~]# certutil -N -d /etc/ipsec.d**

**Enter a password which will be used to encrypt your keys.**

**The password should be at least 8 characters long,**

**and should contain at least one non-alphabetic character.**

**Enter new password: coto-2012**

**Re-enter password:**

**2.3.2.4 Creación de claves RSA**

Ahora pasaremos a crear nuestro par de de claves, publica y privada, la publica es la que vamos a intercambiar con los clientes que se van a conectar a nuestra servidor.

**[root@slnxapp3 ipsec.d]# ipsecnewhostkey --configdir /etc/ipsec.d --output /etc/ipsec.d/hostkeys.secrets --bits 2048 --password coto-2012**

**Generated RSA key pair using the NSS database**

El comando anterior lo que hace es generar una clave que tiene 2048 bit y va a ser guardada en el archivo hostkeys.secrets, la extensión es importante Openswan maneja estos dos tipos de extensiones .conf y .secrets.

Una vez hecho eso podemos ver nuestra clave de la siguiente manera:

**[root@slnxapp3 ipsec.d]# ipsecshowhostkey --left**

**ipsecshowhostkeynss directory showhostkey: /etc/ipsec.d**

**# rsakeyAQOuvlFkO**

**leftrsasigkey=0sAQOuvlFkORA4+wGxdJRLcjE4FjJ7jP2QyUEa6KrzyEKaVJ+Yr0yM13rbGr1L/VLX/RURrhDRt0SpMd09KRFKOLMDo5OywbiizDFGw3RcgSa2G8srsG9bS6XjWMvThL7t6Vtuxb/Zi5BoGlMnaMq3OLRTL9aQS3dHFWF5iyLQIRqf9HfUyCeLLhHQuDX9W5zdLjJnLrH4ahhy6ZB4qeFDCpsFDNKoYPuJ6MrUWoqAkhyOGTohw/0xfgkCIIbiCMmlwL3DC7DfBjss1YLk4bMyhDQY6t3np7veTjr66ra5M1e9yx11iIqgAUYpLQVtRZr7vfcJlPFY8waQGdRfsIqD1Fil**

**[root@slnxapp3 ipsec.d]# ipsecshowhostkey --right**

**ipsecshowhostkeynss directory showhostkey: /etc/ipsec.d**

**# rsakeyAQOuvlFkO**

**rightrsasigkey=0sAQOuvlFkORA4+wGxdJRLcjE4FjJ7jP2QyUEa6KrzyEKaVJ+Yr0yM13rbGr1L/VLX/RURrhDRt0SpMd09KRFKOLMDo5OywbiizDFGw3RcgSa2G8srsG9bS6XjWMvThL7t6Vtuxb/Zi5BoGlMnaMq3OLRTL9aQS3dHFWF5iyLQIRqf9HfUyCeLLhHQuDX9W5zdLjJnLrH4ahhy6ZB4qeFDCpsFDNKoYPuJ6MrUWoqAkhyOGTohw/0xfgkCIIbiCMmlwL3DC7DfBjss1YLk4bMyhDQY6t3np7veTjr66ra5M1e9yx11iIqgAUYpLQVtRZr7vfcJlPFY8waQGdRfsIqD1FIl**

Openswanutilize el concepto de left/right en vez de cliente/servidor, básicamente podemos decir que cuando no referimos a left estamos hablando nosotros mismo, o sea el servidor o cliente que estamos configurando el cual puede ser una dirección de TCP/IP o un FQDN. Ya estamos casi listos para inciar el servicio, solo resta revisar los archivos de configuración. En /etc/ tenemos un archivo de configuración global que se llama ipsec.conf, lo mas importante a tener en cuenta es que la siguiente línea esté descomentada ya que por defecto viene deshabilitada y cuando iniciamos el servicio no nos va a cargar la configuración para nuestro host.

**[root@slnxapp3 ipsec.d]# cat /etc/ipsec.conf|grep include**

**include /etc/ipsec.d/\*.conf**

**2.3.2.5 Configuracion del archivo de conexión**

A continuación mostramos el archivo de configuración y explicamos los campos relacionados, en el mismo vemos dos conexiones definidas.

**conn sucursal:** es el nombre que le damos a la conexión

**left:** nuestra dirección de IP

**right:** dirección IP del cliente

**auto add:** significa que la conexión se va a establecer manualmente

**authby:** es el tipo de autenticación que usamos, en este caso claves RSA

**ike:** le indicamos que algoritmo de encriptación usamos para la fase1 autenticación

**esp:** le indicamos que algoritmo de encriptación usamos para la fase 2 negociación

**leftxauthserver:** decimos que nosotros somos un servidor de autenticación

**rightxauthclient:** decimos que right ( el cliente ) se va a autenticar con nosotros

**leftrsasigkey:** es nuestra clave publica que sacamos con ipsecshowhostkey --left

**rightrsasigkey:** es la clave publica del cliente que sacamos con ipsecshowhostkey –right

comando que ejecutamos en el cliente.

**[root@slnxapp3 ipsec.d]# cat slnxapp3.conf**

**conn sucursal**

**left=172.16.8.16**

**right=128.2.107.21**

**auto=add**

**#authby=secret**

**authby=rsasig**

**ike=aes256-sha1;modp2048**

**esp=aes256-sha1;modp2048**

**rightxauthclient=yes**

**leftxauthserver=yes**

**leftrsasigkey=0sAQOuvlFkORA4+wGxdJRLcjE4FjJ7jP2QyUEa6KrzyEKaVJ+Yr0yM13rbGr1L/VLX/RURrhDRt0SpMd09KRFKOLMDo5OywbiizDFGw3RcgSa2G8srsG9bS6XjWMvThL7t6Vtuxb/Zi5BoGlMnaMq3OLRTL9aQS3dHFWF5iyLQIRqf9HfUyCeLLhHQuDX9W5zdLjJnLrH4ahhy6ZB4qeFDCpsFDNKoYPuJ6MrUWoqAkhyOGTohw/0xfgkCIIbiCMmlwL3DC7DfBjss1YLk4bMyhDQY6t3np7veTjr66ra5M1e9yx11iIqgAUYpLQVtRZr7vfcJlPFY8waQGdRfsIqD1FIl**

**rightrsasigkey=0sAQPLYNPaDEI2W05NdX8YBVkaF7dVjpkNFuhkJbdMlGqnBcz11wP30aZqv6icuhE0zv7+bYf084XtK702hNAi0ws+eZElf8+hfbLo2lwi6ClmDIiDNiJ6ryUCvAN3QJKgn1nVWvNK5gBlmw6jXD00Bn2pDFkJ5sLvaxck1bEyBXb0RhMPm4otE809D0T7Cyae6gza3/BBj1HPs+6Q6MkmKU/qSJptRkaBSOBJCtFT/VHA6fTan2FykQ9EhpMpkHC0f4aGTFwGt/qB7lxf6AzrI4y36hPkfpuGR+kG+WQT/T+dykXPEPcKO7h0RtJruk8FdkvHREVTSNJaVQMWl2yyk521**

**conn slnxapp2**

**left=172.16.8.16**

**right=172.16.8.15**

**auto=add**

**#authby=secret**

**authby=rsasig**

**ike=aes256-sha1;modp2048**

**esp=aes256-sha1;modp2048**

**rightxauthclient=yes**

**leftxauthserver=yes**

**leftrsasigkey=0sAQOuvlFkORA4+wGxdJRLcjE4FjJ7jP2QyUEa6KrzyEKaVJ+Yr0yM13rbGr1L/VLX/RURrhDRt0SpMd09KRFKOLMDo5OywbiizDFGw3RcgSa2G8srsG9bS6XjWMvThL7t6Vtuxb/Zi5BoGlMnaMq3OLRTL9aQS3dHFWF5iyLQIRqf9HfUyCeLLhHQuDX9W5zdLjJnLrH4ahhy6ZB4qeFDCpsFDNKoYPuJ6MrUWoqAkhyOGTohw/0xfgkCIIbiCMmlwL3DC7DfBjss1YLk4bMyhDQY6t3np7veTjr66ra5M1e9yx11iIqgAUYpLQVtRZr7vfcJlPFY8waQGdRfsIqD1FIl**

**rightrsasigkey=0sAQPRbGsnPNwBLgBpsKRInn+xD78WgzhVZlioYn2zqB0lWQGgVKW4b2d+SamsWuPFbY2RMe/UCkWrIYIxEivBZQxmI5VhoB4Vtq84Vzz/V0D/Zzz5PgJScbK3zyaYxiEykKgq2PgYUaiY5lX8cVlKm18OGtTh8o0ljBVCwuCA0eK/MyS7AmOMcP3OBwhmGd5hP0N94E35SVDlUlSHXykKnzlpzNDkdfYKhPDwhcicGaDvF1i1NLdv3/bdgSHpCNnXWKRyVa7dypTE1SxiVaNPuLT3BSAPK1YOPhyqz97VuhfKGBaQuAkO3yRXm1FcNzwCefFvkGC8cxDw3Xr52BzR7f6h**

**2.3.2.6 Configuración del Autenticación**

Para darle un nivel mas de seguridad Openswan nos permite utilizar otro tipo de autenticación a la hora de conectarnos al servidor, esto lo podemos hacer con PAM o utilizando un archivo similar al /etc/passwd de linux donde guardamos los nombres de usuarios y las claves encriptadas, elegimos el archivo passwd por su sencillez y fácil configuración, para ello utilizamos la utilidad htpasswd que apache, ahora no solo vamos a tener el canal de comunicaciones encriptado, sino que también, ante un eventual compromiso del sistema remoto ( cliente ) si o si va a necesitar un usuario y clave para poder conectarse a nuestra red.

**2.3.2.7 Creación del archivo passwd y usuarios**

Como el arhivo no existe debemos crearlo cuando ejecutamos la sentencia por primera vez, esto lo hacemos de la siguiente forma ubicados en el directorios /etc/ipsec.d , no puede ser otro directorio ya que ipsec busca el archivo passwd dentro de ese directorio:

**htpasswd -cd passwd usuario**

passwd es el nombre del archivo que vamos a crear, y usuario es el nombre del usuario que vamos a agregar. Por defecto cuanod creamos el archivo si nos fijamos el formato va a tener la siguiente estructura

**pmorales:$1$qM6.MvT9$/NlzixMu.ImZbq4GBihd7.**

pmorales es el nombre del usuario, el resto es la clave encriptada utilizando MD5 que es el formato que ipsec busca y el cual va a interpretar, lo que debemos hacer es modificar levemente la estructura del archivo para que openswan pueda interpretarlo y permitir que los usuarios definidos en el archivo puedan conectarse, el archivo debe quedar de la siguiente manera:

**pmorales:$1$qM6.MvT9$/NlzixMu.ImZbq4GBihd7.:sucursal**

Nótese que agregamos **:sucursal** esto le va a indicar a openswan que el usuario **pmorales** va a poder conectarse desde el cliente cuyo nombre de conexión es **sucursal**. Si necesitamos agregar otro usuario, como yà tenemos el archivo creado, ejecutamos el siguiente comando:

**htpasswdpasswd caja01**

Eso va a agregar el usuario **caja01** a nuestro archivo passwd, tal comando va a pedir que se le ingrese una clave, una vez proporcionada la misma, el archivo quedará de la siguiente manera:

**pmorales:$1$qM6.MvT9$/NlzixMu.ImZbq4GBihd7.:sucursal**

**caja01:mLHlUOgtF.QCk**

Nuevamente debemos modificar el formato del archivo para que openswan pueda interpretarlo, el mismo deberá quedar como se muestra a continuación:

**pmorales:$1$qM6.MvT9$/NlzixMu.ImZbq4GBihd7.:sucursal**

**caja01:mLHlUOgtF.QCk:slnxapp2**

Esto le indica a openswan que el usuario caja01 se puede conectar desde la conexion cuyo nombre es slnxapp2

**2.3.2.8 Iniciando el Servicio**

Una vez configurado nuestro servidor ya estamos en condiciones de inciar el servicio lo cual hacemos de la siguiente manera:

**[root@slnxapp3 ipsec.d]# service ipsec start**

**ipsec\_setup: Starting Openswan IPsec U2.6.32/K2.6.32-220.el6.x86\_64...**

**ipsec\_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips\_enabled**

**ipsec\_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips\_enabled**

Verificamos el estado:

**[root@slnxapp3 ipsec.d]# ipsecverify**

**Checking your system to see if IPsec got installed and started correctly:**

**Version check and ipsec on-path [OK]**

**Linux Openswan U2.6.32/K2.6.32-220.el6.x86\_64 (netkey)**

**Checking for IPsec support in kernel [OK]**

**SAref kernel support [N/A]**

**NETKEY: Testing for disabled ICMP send\_redirects [OK]**

**NETKEY detected, testing for disabled ICMP accept\_redirects [OK]**

**Checking that pluto is running [OK]**

**Pluto listening for IKE on udp 500 [OK]**

**Pluto listening for NAT-T on udp 4500 [OK]**

**Checking for 'ip' command [OK]**

**Checking /bin/sh is not /bin/dash [OK]**

**Checking for 'iptables' command [OK]**

**OpportunisticEncryptionSupport [DISABLED]**

Consultamos por el estado y observamos que tenemos dos conexiones activas, slxnapp2 y sucursal, slanxpp2 y sucursal indican el nombre de la conexión a la cual hace referencia el archivo de configuración, no necesariamente indica un nombre de host valido.

**[root@slnxapp3 ipsec.d]# ipsec auto –status**

**000 "slnxapp2": 172.16.8.16<172.16.8.16>[+XS+S=C]...172.16.8.15<172.16.8.15>[+XC+S=C]; unrouted; eroute owner: #0**

**000 "slnxapp2": myip=unset; hisip=unset;**

**000 "slnxapp2": ike\_life: 3600s; ipsec\_life: 28800s; rekey\_margin: 540s; rekey\_fuzz: 100%; keyingtries: 0**

**000 "slnxapp2": policy: RSASIG+ENCRYPT+TUNNEL+PFS+IKEv2ALLOW+SAREFTRACK; prio: 32,32; interface: eth0;**

**000 "slnxapp2": newest ISAKMP SA: #0; newest IPsec SA: #0;**

**000 "slnxapp2": IKE algorithms wanted: AES\_CBC(7)\_256-SHA1(2)\_000-MODP2048(14); flags=-strict**

**000 "slnxapp2": IKE algorithms found: AES\_CBC(7)\_256-SHA1(2)\_160-MODP2048(14)**

**000 "slnxapp2": ESP algorithms wanted: AES(12)\_256-SHA1(2)\_000; pfsgroup=MODP2048(14); flags=-strict**

**000 "slnxapp2": ESP algorithms loaded: AES(12)\_256-SHA1(2)\_160**

**000 "sucursal": 172.16.8.16<172.16.8.16>[+XS+S=C]...128.2.107.21<128.2.107.21>[+XC+S=C]; unrouted; eroute owner: #0**

**000 "sucursal": myip=unset; hisip=unset;**

**000 "sucursal": ike\_life: 3600s; ipsec\_life: 28800s; rekey\_margin: 540s; rekey\_fuzz: 100%; keyingtries: 0**

**000 "sucursal": policy: RSASIG+ENCRYPT+TUNNEL+PFS+IKEv2ALLOW+SAREFTRACK; prio: 32,32; interface: eth0;**

**000 "sucursal": newest ISAKMP SA: #0; newest IPsec SA: #0;**

**000 "sucursal": IKE algorithms wanted: AES\_CBC(7)\_256-SHA1(2)\_000-MODP2048(14); flags=-strict**

**000 "sucursal": IKE algorithms found: AES\_CBC(7)\_256-SHA1(2)\_160-MODP2048(14)**

**000 "sucursal": ESP algorithms wanted: AES(12)\_256-SHA1(2)\_000; pfsgroup=MODP2048(14); flags=-strict**

**000 "sucursal": ESP algorithmsloaded: AES(12)\_256-SHA1(2)\_160**

**2.4 Configuracion Cliente ( RIGHT )**

Para la configuración del cliente es igual que en el servidor, la creación de la base para NSS y la creación de claves es igual, mismos comandos, mismos parámetros mismos parámetros. Lo que hacemos a continuación es, copiamos y pegamos tal cual está la configuración en el servidor para ese cliente en el archivo de configuración del mismo, vemos un ejemplo a continuación:

**[root@suc666 ipsec.d]# cat suc666.conf**

**conn sucursal**

**left=172.16.8.16**

**right=128.2.107.21**

**auto=add**

**leftxauthserver=yes**

**rightxauthclient=yes**

**authby=rsasig**

**ike=aes256-sha1;modp2048**

**esp=aes256-sha1;modp2048**

**leftrsasigkey=0sAQOuvlFkORA4+wGxdJRLcjE4FjJ7jP2QyUEa6KrzyEKaVJ+Yr0yM13rbGr1L/VLX/RURrhDRt0SpMd09KRFKOLMDo5OywbiizDFGw3RcgSa2G8srsG9bS6XjWMvThL7t6Vtuxb/Zi5BoGlMnaMq3OLRTL9aQS3dHFWF5iyLQIRqf9HfUyCeLLhHQuDX9W5zdLjJnLrH4ahhy6ZB4qeFDCpsFDNKoYPuJ6MrUWoqAkhyOGTohw/0xfgkCIIbiCMmlwL3DC7DfBjss1YLk4bMyhDQY6t3np7veTjr66ra5M1e9yx11iIqgAUYpLQVtRZr7vfcJlPFY8waQGdRfsIqD1FIl**

**rightrsasigkey=0sAQPLYNPaDEI2W05NdX8YBVkaF7dVjpkNFuhkJbdMlGqnBcz11wP30aZqv6icuhE0zv7+bYf084XtK702hNAi0ws+eZElf8+hfbLo2lwi6ClmDIiDNiJ6ryUCvAN3QJKgn1nVWvNK5gBlmw6jXD00Bn2pDFkJ5sLvaxck1bEyBXb0RhMPm4otE809D0T7Cyae6gza3/BBj1HPs+6Q6MkmKU/qSJptRkaBSOBJCtFT/VHA6fTan2FykQ9EhpMpkHC0f4aGTFwGt/qB7lxf6AzrI4y36hPkfpuGR+kG+WQT/T+dykXPEPcKO7h0RtJruk8FdkvHREVTSNJaVQMWl2yyk521**

**2.4.1 Inicio del Servicio**

El servicio lo iniciamos al igual que en el servidor **serviceipsecstart**, podemos verificar que la configuración esté correcta con **ipsecverify**, recordemos que debemos modificar ciertos parámetros del kernel al igual que en el servidor, con la única diferencia que la conexión la debemos establecer manualmente ya que nos va a pedir usuario y clave como vemos a continuación:

Verificamos que la configuración esté bien, si falta configurar algún parámetro del kernel nos lo va a indicar como mostramos a continuación:

**[root@suc666 ipsec.d]# ipsec verify**

**Checking your system to see if IPsec got installed and started correctly:**

**Version check and ipsec on-path [OK]**

**Linux Openswan U2.6.32/K2.6.32-131.0.15.el6.i686 (netkey)**

**Checking for IPsec support in kernel [OK]**

**SAref kernel support [N/A]**

**NETKEY: Testing for disabled ICMP send\_redirects [FAILED]**

**Please disable /proc/sys/net/ipv4/conf/\*/send\_redirects**

**or NETKEY will cause the sending of bogus ICMP redirects!**

**NETKEY detected, testing for disabled ICMP accept\_redirects [FAILED]**

**Please disable /proc/sys/net/ipv4/conf/\*/accept\_redirects**

**or NETKEY will accept bogus ICMP redirects!**

**Testing against enforced SElinux mode [FAILED]**

**SElinux is running in 'enforced' mode.**

**If you encounter network related SElinux errors, especially when using KLIPS,**

**try disabling SElinux using:**

**echo "0" > /selinux/enforce (or edit /etc/sysconfig/selinux)**

**Checking that pluto is running [OK]**

**Pluto listening for IKE on udp 500 [OK]**

**Pluto listening for NAT-T on udp 4500 [OK]**

**Checking for 'ip' command [OK]**

**Checking /bin/sh is not /bin/dash [OK]**

**Checking for 'iptables' command [OK]**

**OpportunisticEncryptionSupport [DISABLED]**

Aplicamos los cambios y volvemos a correr la verificación:

**[root@suc666 ipsec.d]# ipsec verify**

**Checking your system to see if IPsec got installed and started correctly:**

**Version check and ipsec on-path [OK]**

**Linux Openswan U2.6.32/K2.6.32-131.0.15.el6.i686 (netkey)**

**Checking for IPsec support in kernel [OK]**

**SAref kernel support [N/A]**

**NETKEY: Testing for disabled ICMP send\_redirects [OK]**

**NETKEY detected, testing for disabled ICMP accept\_redirects [OK]**

**Testing against enforced SElinux mode [FAILED]**

**SElinux is running in 'enforced' mode.**

**If you encounter network related SElinux errors, especially when using KLIPS,**

**try disabling SElinux using:**

**echo "0" > /selinux/enforce (or edit /etc/sysconfig/selinux)**

**Checking that pluto is running [OK]**

**Pluto listening for IKE on udp 500 [OK]**

**Pluto listening for NAT-T on udp 4500 [OK]**

**Checking for 'ip' command [OK]**

**Checking /bin/sh is not /bin/dash [OK]**

**Checking for 'iptables' command [OK]**

**OpportunisticEncryptionSupport [DISABLED]**

Ahora podemos iniciar la conexión:

**[root@suc666 ipsec.d]# ipsec auto --up Sucursal**

**104 "sucursal" #1: STATE\_MAIN\_I1: initiate**

**003 "sucursal" #1: received Vendor ID payload [Openswan (this version) 2.6.32 ]**

**003 "sucursal" #1: received Vendor ID payload [Dead Peer Detection]**

**003 "sucursal" #1: received Vendor ID payload [XAUTH]**

**003 "sucursal" #1: received Vendor ID payload [RFC 3947] method set to=109**

**106 "sucursal" #1: STATE\_MAIN\_I2: sent MI2, expecting MR2**

**003 "sucursal" #1: NAT-Traversal: Result using RFC 3947 (NAT-Traversal): no NAT detected**

**108 "sucursal" #1: STATE\_MAIN\_I3: sent MI3, expecting MR3**

**003 "sucursal" #1: received Vendor ID payload [CAN-IKEv2]**

**004 "sucursal" #1: STATE\_MAIN\_I4: ISAKMP SA established {auth=OAKLEY\_RSA\_SIG cipher=aes\_256 prf=oakley\_sha group=modp2048}**

**041 "sucursal" #1: sucursal prompt for Username:**

**Enter username: pmorales**

**040 "sucursal" #1: sucursal prompt for Password:**

**Enter passphrase:**

**004 "sucursal" #1: STATE\_XAUTH\_I1: XAUTH client - awaiting CFG\_set**

**003 "sucursal" #1: discarding duplicate packet; already STATE\_XAUTH\_I1**

**004 "sucursal" #1: STATE\_XAUTH\_I1: XAUTH client - awaiting CFG\_set**

**117 "sucursal" #2: STATE\_QUICK\_I1: initiate**

**004 "sucursal" #2: STATE\_QUICK\_I2: sent QI2, IPsec SA established tunnel mode {ESP=>0x08c65125 <0xd5859087 xfrm=AES\_256-HMAC\_SHA1 NATOA=none NATD=none DPD=none}**

**[root@suc666 ipsec.d]#**

De este modo acabamos de iniciar una sesión encriptada entre suc66 y slnxapp3, si miramos los los del lado del servidor podemos ver como se estableció las misma:

**tail -f /var/log/secure**

**Jul 3 16:19:49 slnxapp3 pluto[23741]: "sucursal" #2: XAUTH: Sending XAUTH Login/Password Request**

**Jul 3 16:19:49 slnxapp3 pluto[23741]: "sucursal" #2: XAUTH: Sending Username/Password request (XAUTH\_R0)**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: XAUTH: User pmorales: Attempting to login**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: XAUTH: md5 authentication being called to authenticate user pmorales**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: XAUTH: password file (/etc/ipsec.d/passwd) open.**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: XAUTH: checking user(pmorales:sucursal)**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: XAUTH: User pmorales: Authentication Successful**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #2: XAUTH: xauth\_inR1(STF\_OK)**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #2: transition from state STATE\_XAUTH\_R1 to state STATE\_MAIN\_R3**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #2: STATE\_MAIN\_R3: sent MR3, ISAKMP SA established**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #2: the peer proposed: 172.16.8.16/32:0/0 -> 128.2.107.21/32:0/0**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #3: responding to Quick Mode proposal {msgid:a822f92e}**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #3: us: 172.16.8.16<172.16.8.16>[+XS+S=C]**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #3: them: 128.2.107.21<128.2.107.21>[+XC+S=C]**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #3: transition from state STATE\_QUICK\_R0 to state STATE\_QUICK\_R1**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #3: STATE\_QUICK\_R1: sent QR1, inbound IPsec SA installed, expecting QI2**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #3: transition from state STATE\_QUICK\_R1 to state STATE\_QUICK\_R2**

**Jul 3 16:20:27 slnxapp3 pluto[23741]: "sucursal" #3: STATE\_QUICK\_R2: IPsec SA established tunnel mode {ESP=>0xd5859087 <0x08c65125 xfrm=AES\_256-HMAC\_SHA1 NATOA=none NATD=none DPD=none}**

**2.4.2 Verificación de la conexión**

Mediante tcpdump realizamos una caputra de paquetes en donde vemos en efecto que la conexión estáencriptada, vemos paquetes ESP(spi ). Capturamos los paquetes que van con destino slnxapp3 y desde suc666 simplemente hicimos un ping.

**tcpdumpdst host 172.16.8.16**

**10:34:33.915258 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x6598c86a,seq=0x1), length 132**

**10:34:34.915921 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x6598c86a,seq=0x2), length 132**

**10:34:35.917804 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x6598c86a,seq=0x3), length 132**

**10:42:08.070789 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x1), length 100**

**10:42:08.072120 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x2), length 100**

**10:42:08.080910 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x3), length 100**

**10:42:08.081044 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x4), length 116**

**10:42:08.082407 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x5), length 884**

**10:42:08.084422 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x6), length 116**

**10:42:08.088843 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x7), length 244**

**10:42:08.135888 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x8), length 100**

**10:42:09.712357 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0x9), length 116**

**10:42:09.752494 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0xa), length 148**

**10:42:09.753278 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0xb), length 100**

**10:42:09.753378 IP 128.2.107.21 > slnxapp3.redcoto.com.ar: ESP(spi=0x7e9b5b31,seq=0xc), length 164**

**3 packets captured**

**6 packets received by filter**

**0 packets dropped by kernel**

**3.0 Respaldo de configuración**

Si por algún motivo debemos reemplazar la caja por otra ya sea porque tuvimos que reinstalar el sistema o reemplazarla para reparación, es bueno hacer una copia del directorio **/etc/ipsec.d/** y **/etc/ipsec.conf** en un lugar seguro, o mismo en el servidor al cual se conectan las cajas o en algún otro servidor en central,luego, simplemente restaurando el directorio podemos nuevamente restablecer la comunicación con el servidor.

**Revisiones.**

**Rev1**

Se define que la conexión de las cajas va a ser automática y no se requiere que se ingrese usuario y clave para la conexión, lo que se establece es que la caja no comunique si no se establece la conexión segura, esto se logra solo con iniciar el servicio ipsec, si está inactivo o falló por algún motivo no va a poder siquiera hacer ping al servidor en el cual el servicio si tiene que estar ejecutándose obviamente.

A continuación de muestra como queda la configuración del lado cliente ( LEFT ) o cajas y de lado servidor o RIGHT, o Gateway, lo que está comentado ( # ) es lo que se deshabilitó y se cambió auto=add por auto=start.

Cliente:

**conn sucursal**

**left=172.16.8.16**

**right=128.2.107.21**

**auto=start**

**#leftxauthserver=yes**

**#rightxauthclient=yes**

**authby=rsasig**

**ike=aes256-sha1;modp2048**

**esp=aes256-sha1;modp2048**

**leftrsasigkey=0sAQOuvlFkORA4+wGxdJRLcjE4FjJ7jP2QyUEa6KrzyEKaVJ+Yr0yM13rbGr1L/VLX/RURrhDRt0SpMd09KRFKOLMDo5OywbiizDFGw3RcgSa2G8srsG9bS6XjWMvThL7t6Vtuxb/Zi5BoGlMnaMq3OLRTL9aQS3dHFWF5iyLQIRqf9HfUyCeLLhHQuDX9W5zdLjJnLrH4ahhy6ZB4qeFDCpsFDNKoYPuJ6MrUWoqAkhyOGTohw/0xfgkCIIbiCMmlwL3DC7DfBjss1YLk4bMyhDQY6t3np7veTjr66ra5M1e9yx11iIqgAUYpLQVtRZr7vfcJlPFY8waQGdRfsIqD1FIl**

**rightrsasigkey=0sAQPLYNPaDEI2W05NdX8YBVkaF7dVjpkNFuhkJbdMlGqnBcz11wP30aZqv6icuhE0zv7+bYf084XtK702hNAi0ws+eZElf8+hfbLo2lwi6ClmDIiDNiJ6ryUCvAN3QJKgn1nVWvNK5gBlmw6jXD00Bn2pDFkJ5sLvaxck1bEyBXb0RhMPm4otE809D0T7Cyae6gza3/BBj1HPs+6Q6MkmKU/qSJptRkaBSOBJCtFT/VHA6fTan2FykQ9EhpMpkHC0f4aGTFwGt/qB7lxf6AzrI4y36hPkfpuGR+kG+WQT/T+dykXPEPcKO7h0RtJruk8FdkvHREVTSNJaVQMWl2yyk521**

Servidor:

**conn sucursal**

**left=172.16.8.16**

**right=128.2.107.21**

**auto=start**

**#authby=secret**

**authby=rsasig**

**ike=aes256-sha1;modp2048**

**esp=aes256-sha1;modp2048**

**#rightxauthclient=yes**

**#leftxauthserver=yes**

**leftrsasigkey=0sAQOuvlFkORA4+wGxdJRLcjE4FjJ7jP2QyUEa6KrzyEKaVJ+Yr0yM13rbGr1L/VLX/RURrhDRt0SpMd09KRFKOLMDo5OywbiizDFGw3RcgSa2G8srsG9bS6XjWMvThL7t6Vtuxb/Zi5BoGlMnaMq3OLRTL9aQS3dHFWF5iyLQIRqf9HfUyCeLLhHQuDX9W5zdLjJnLrH4ahhy6ZB4qeFDCpsFDNKoYPuJ6MrUWoqAkhyOGTohw/0xfgkCIIbiCMmlwL3DC7DfBjss1YLk4bMyhDQY6t3np7veTjr66ra5M1e9yx11iIqgAUYpLQVtRZr7vfcJlPFY8waQGdRfsIqD1FIl**

**rightrsasigkey=0sAQPLYNPaDEI2W05NdX8YBVkaF7dVjpkNFuhkJbdMlGqnBcz11wP30aZqv6icuhE0zv7+bYf084XtK702hNAi0ws+eZElf8+hfbLo2lwi6ClmDIiDNiJ6ryUCvAN3QJKgn1nVWvNK5gBlmw6jXD00Bn2pDFkJ5sLvaxck1bEyBXb0RhMPm4otE809D0T7Cyae6gza3/BBj1HPs+6Q6MkmKU/qSJptRkaBSOBJCtFT/VHA6fTan2FykQ9EhpMpkHC0f4aGTFwGt/qB7lxf6AzrI4y36hPkfpuGR+kG+WQT/T+dykXPEPcKO7h0RtJruk8FdkvHREVTSNJaVQMWl2yyk521**

**Rev2**

Se actualizó la configuración del firewall, se redefinen las reglas para hacerlas más compactas a pedido de Cristian Larrosa.

**Rev 3**

Se actualizó la configuración del firewall, se configura el logueo para que envíe como kern.crit al archivo /var/log/iptables.log y al servidor slnxapp3, con esta configuración solo enviamos las alertas referentes a iptables y no es resto de los mensajes delkernel ahorrando trafico no necesario.

#define KERN\_EMERG "<0>" /\* system is unusable \*/

#define KERN\_ALERT "<1>" /\* action must be taken immediately \*/

#define KERN\_CRIT "<2>" /\* critical conditions \*/

#define KERN\_ERR "<3>" /\* error conditions \*/

#define KERN\_WARNING "<4>" /\* warning conditions \*/

#define KERN\_NOTICE "<5>" /\* normal but significant condition \*/

#define KERN\_INFO "<6>" /\* informational \*/

#define KERN\_DEBUG "<7>" /\* debug-level messages \*/

**$IPTABLES -A AUDITA-TRAFICO -d $D\_INTERNA \**

**-m limit --limit 10/s -j LOG \**

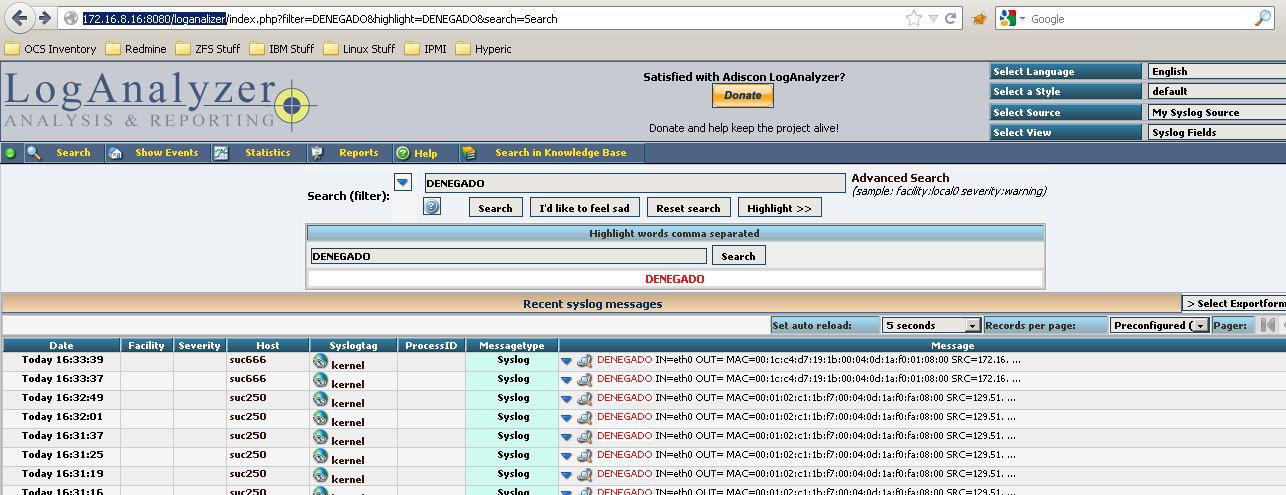
**--log-prefix "DENEGADO " --log-level 2**

En el cliente hay que modificar syslog.conf para que quede como se muestar a continuación:

**kern.crit /var/log/iptables.log**

**kern.crit @172.16.8.16**

De esta manera guardamos una copiar local en el servidor, el log lo podemos ver utilizando la herramienta LogAnalizer accediendo a la siguiente url: <http://172.16.8.16:8080/loganalizer> la imagen a continuación es una captura de la pantalla del software mostrando la imagen casi en tiempo real, ya que podemos hacer que refresque cada 5 segundos que es el tiempo mínimo que nos permite. El software está instalado en el servidor slnxapp3 en /var/www/html/loganalizer, la configuración es sencilla y el archivo de configuración se llama config.php.



**Rev 4**

Recordar cargar el moduloip\_conntrack\_ftp, o bien lo podemos hacer ejecutandomodprobeip\_conntrack\_ftp o agregarlo en /etc/rc.modules, si no existe el archivo hay que crearlo y darle permisos de ejecución , la línea que hay que agregar es la siguiente:

**suc02:/etc>cat rc.modules**

**modprobeip\_conntrack\_ftp**

**Rev 5**

Se agregó una tarea en el cron que contrala que el firewall esté activo:

**\*/10 \* \* \* \* /tecnol/util/check\_iptables.sh > /dev/null 2>&1**

**Rev 6**

Arhivos que se agregaron que conformar el firewall:

**accept.rules**

Tiene las reglas con ACCEPT en todas las cadenas, este lo generamos manualmente con iptables-save>accept.rules.

**bajafirewall.sh**

Baja el firewall

**check\_iptables.sh**

Script que chequea desde el cron que las reglas esten activas

**check\_log2.messages.seek**

Lo genera el script check\_log2.pl , guarda una posición de ultima búsqueda en el archivo messages.

**check\_log2.pl**

Chequea si ocurre alguna ocurrencia de la palabra DENEGADO ( ver Rev 3 ) en el archivo /var/log/messages y envia un mail de notificación si encuentra ocurrencia.

**firewall.sh**

El firewall en si.

**utils.pm**

**utils.sh**

Archivos usados por el script chec\_log2.pl

**Rev7**

Estas son las variables que relevantes para cada sucursal que hay que modificar y podes copiar a otra y que funcione:

**D\_INTERNA='129.X.255.253'**

**MISUBRED='129.103.0.0/255.255.0.0'**

**MISUBRED1='129.103.255.0/255.255.255.0'**

**# Servicio de Maquinas Zona E!**

**NTSZEXXX='129.X.250.250'**

**#IP de las cajas**

**REDCAJAS1='129.X.102.0/24'**

**REDFTP='129.103.22.0/24'**

**# Coto Digital.**

**NTSXXX='129.X.253.253'**

Si tenemos alertas del firewall que tal IP no se puede conectar al puerto 1521/26, la función que otorga permisos a dicho puerto el Activar\_Oracle, si definimos una variable con una IP o rango de IP’s hay que agregarla acá

Activar\_Oracle(){

echo '- Politica [ Habilito Oracle ]'

for server in $DBAS $NTSDBA $MAYUDA $NTSWEB8 $MISUBRED1 $NTSINTRANET \

$PUCARA $GRIDDBA $NTSZE129 $NTSANDES $NTSSQL11 \

$GDM $AROBERTO $MMONTERO $NTS129 $OSIRIS $LAURA \

$ALANDUN $NTSNORTE $NTSWEB7 $FFALLARDI $DWH \

$SEGINFORMACION $MULTISTORE $NTSPAMPA **$VARIABLE-NUEVA**

do

for i in $LISTENER

do

$IPTABLES -A INPUT -p tcp -s $server -d $D\_INTERNA \

--dport $i -m state --state NEW,ESTABLISHED -j ACCEPT

done

done

}

**Rev 8**

Descripciòn de las reglas pedido por la auditoria ejemplo sucursal 107:

**Chain INPUT (policy DROP)**

**target prot opt source destination**

**Por defecto deniego paquetes mal formados y que no cumplen con el standard TCP/IP.**

DROP all -- anywhere anywhere state INVALID

DROP tcp -- anywhere anywhere tcp flags:SYN,ACK/SYN,ACK state NEW

DROP tcp -- anywhere anywhere tcp flags:FIN,SYN,RST,PSH,ACK,URG/NONE

DROP tcp -- anywhere anywhere tcp flags:FIN,SYN/FIN,SYN

DROP tcp -- anywhere anywhere tcp flags:SYN,RST/SYN,RST

DROP tcp -- anywhere anywhere tcp flags:FIN,SYN,RST,PSH,ACK,URG/FIN,SYN,RST,ACK,URG

DROP tcp -- anywhere anywhere tcp flags:FIN,RST/FIN,RST

DROP tcp -- anywhere anywhere tcp flags:FIN,ACK/FIN

DROP tcp -- anywhere anywhere tcp flags:PSH,ACK/PSH

DROP tcp -- anywhere anywhere tcp flags:ACK,URG/URG

DROP all -- ntsalfred2.redcoto.com.ar suc107

**Acepto paquetes internos, comunicación entre el poropio servidor.**

ACCEPT all -- loopback loopback

ACCEPT all -- suc107 suc107 state NEW,RELATED,ESTABLISHED

ACCEPT all -- nodo2 anywhere

ACCEPT all -- 129.107.255.250 anywhere

**Acepto paquetes que cumplan los siguientes estados, no significa que acepto toda comunicación de cualquier lado.**

ACCEPT all -- anywhere anywhere state RELATED,ESTABLISHED

**Permito hacer y responder el ping.**

ACCEPT icmp -- anywhere suc107 icmp echo-request state NEW,RELATED,ESTABLISHED

ACCEPT icmp -- anywhere suc107 icmp echo-reply state NEW,RELATED,ESTABLISHED

**Permito consultas HTTP.**

ACCEPT tcp -- anywhere suc107 tcp dpt:http state NEW,RELATED,ESTABLISHED

**Permito conexiones a la base Oracle.**

ACCEPT tcp -- anywhere suc107 tcp dpt:1521 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- anywhere suc107 tcp dpt:ingreslock state NEW,RELATED,ESTABLISHED

**Habilito servicios de Impresión y SAMBA.**

ACCEPT tcp -- anywhere suc107 tcp dpt:microsoft-ds state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- anywhere suc107 tcp dpt:netbios-ssn state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- anywhere suc107 tcp dpt:printer state NEW,RELATED,ESTABLISHED

**Acepto VNC.**

ACCEPT tcp -- anywhere suc107 tcp dpts:5900:5901 state NEW,RELATED,ESTABLISHED

**Permito consulta recursos compartidos SAMBA.**

ACCEPT udp -- anywhere suc107 udp dpts:netbios-ns:netbios-dgm state NEW,ESTABLISHED

**Habilito Rsync.**

ACCEPT udp -- anywhere suc107 udp dpt:rsync state NEW,ESTABLISHED

**Habilito DHCP**

ACCEPT udp -- anywhere suc107 udp dpts:bootps:bootpc state NEW,ESTABLISHED

**Acepto todo desde los siguientes servidores.**

ACCEPT all -- online1.redcoto.com.ar suc107 state NEW,RELATED,ESTABLISHED

ACCEPT all -- online2.redcoto.com.ar suc107 state NEW,RELATED,ESTABLISHED

ACCEPT all -- sf000 suc107 state NEW,RELATED,ESTABLISHED

**Permito el servicio Control-M**

ACCEPT tcp -- ctm01.redcoto.com.ar suc107 tcp dpts:9005:9007 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- ctm01.redcoto.com.ar suc107 tcp dpt:ingreslock state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- ctm01.redcoto.com.ar suc107 tcp dpts:2370:2371 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 129.107.253.253 suc107 tcp dpts:9005:9007 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 129.107.253.253 suc107 tcp dpt:ingreslock state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 129.107.253.253 suc107 tcp dpts:2370:2371 state NEW,RELATED,ESTABLISHED

**Acepto todo el tráfico desde las cajas.**

ACCEPT all -- 129.107.102.0/24 suc107 state NEW,ESTABLISHED

ACCEPT all -- 129.107.103.0/24 suc107 state NEW,ESTABLISHED

**Acepto conexiones por secure shell SSH.**

ACCEPT tcp -- slnxapp3.redcoto.com.ar suc107 tcp dpt:ssh state NEW,ESTABLISHED

ACCEPT tcp -- adm01.redcoto.com.ar suc107 tcp dpt:ssh state NEW,ESTABLISHED

ACCEPT tcp -- sp9.redcoto.com.ar suc107 tcp dpt:ssh state NEW,ESTABLISHED

ACCEPT tcp -- gdm.redcoto.com.ar suc107 tcp dpt:ssh state NEW,ESTABLISHED

ACCEPT tcp -- PSP1.redcoto.com.ar suc107 tcp dpt:ssh state NEW,ESTABLISHED

**Permito consultas de correo.**

ACCEPT tcp -- relayinterno suc107 tcp spt:smtp dpts:1024:65535 state ESTABLISHED

**Habilito consultas DNS.**

ACCEPT udp -- dc08001.redcoto.com.ar suc107 udp spt:domain dpts:1024:65535

ACCEPT udp -- dc08000.redcoto.com.ar suc107 udp spt:domain dpts:1024:65535

**Permito sincronizar la hora.**

ACCEPT tcp -- dc08001.redcoto.com.ar suc107 tcp dpt:ntp state NEW

ACCEPT tcp -- dc08000.redcoto.com.ar suc107 tcp dpt:ntp state NEW

**Permito FTP para las cámaras.**

ACCEPT tcp -- 129.107.22.0/24 suc107 tcp dpts:ftp-data:ftp state NEW

ACCEPT tcp -- 128.2.101.72 suc107 tcp dpts:ftp-data:ftp state NEW

**Permito monitoreo de nagios.**

ACCEPT tcp -- slnxapp3.redcoto.com.ar suc107 tcp dpt:nrpe state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- slnxapp3.redcoto.com.ar suc107 tcp dpt:snmp state NEW,RELATED,ESTABLISHED

**Permito consultas servicio Ticket Status.**

ACCEPT tcp -- 172.16.0.0/16 suc107 tcp dpt:10101 state NEW,RELATED,ESTABLISHED

ACCEPT tcp -- 129.107.255.0/24 suc107 tcp dpt:10101 state NEW,RELATED,ESTABLISHED

**Habilito comunicación con el servidor de etiquetas.**

ACCEPT tcp -- 129.107.255.251 suc107 tcp dpts:ftp-data:ftp state NEW,RELATED,ESTABLISHED

**Habilito consultar servicio multistore, Mesa de Ayuda/Chariano.**

ACCEPT tcp -- 172.16.0.0/16 suc107 tcp dpt:6580 state NEW,ESTABLISHED

ACCEPT tcp -- 172.16.0.0/16 suc107 tcp dpt:7373 state NEW,ESTABLISHED

ACCEPT tcp -- 129.199.0.0/24 suc107 tcp dpt:6580 state NEW,ESTABLISHED

ACCEPT tcp -- 129.199.0.0/24 suc107 tcp dpt:7373 state NEW,ESTABLISHED

**Regla para auditar el firewall /var/log/iptables.log**

AUDITA-TRAFICO all -- anywhere anywhere

Chain FORWARD (policy DROP)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

ACCEPT all -- anywhere anywhere state NEW,RELATED,ESTABLISHED

Chain AUDITA-TRAFICO (1 references)

target prot opt source destination

LOG all -- anywhere suc107 limit: avg 10/sec burst 5 LOG level crit prefix `DENEGADO '

DROP all -- anywhere anywhere