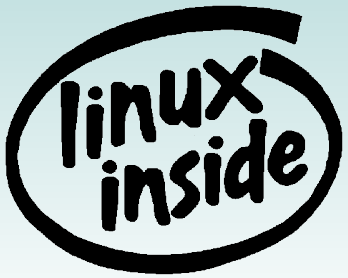
Firewall para PYME
y
WLAN segura

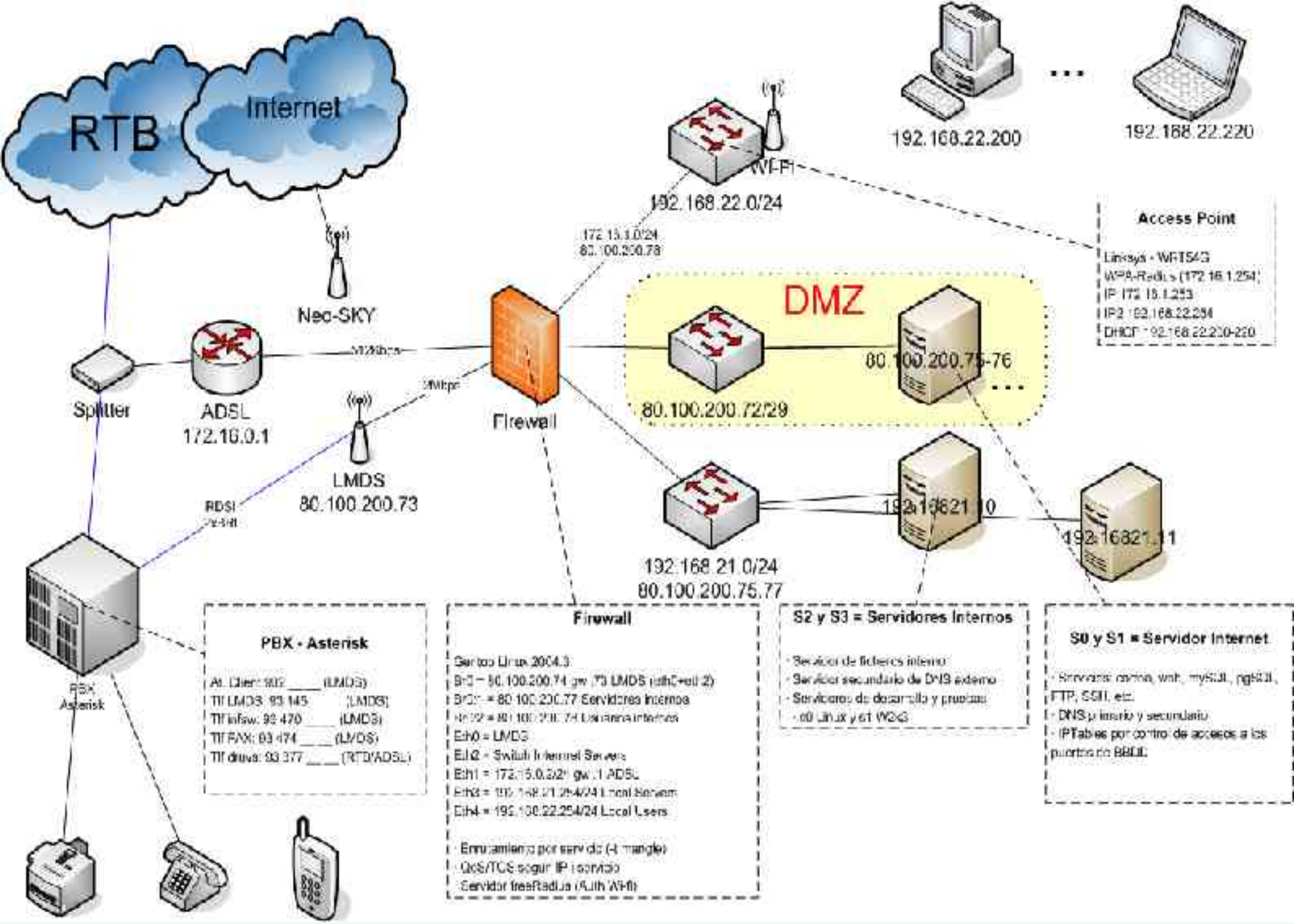
| Oriol Rius
| oriol@joor.net
| http://oriol.joor.net



Indice

Objetivo

1. Conectar nuestra PYME a internet de forma segura
2. Podemos gestionar el tráfico
3. Ahorrarnos el cableado de la empresa (WLAN)
4. Montar un entorno seguro
    4.1 perimetralmente firewall
    4.2 acceso a la red privada inalámbrica
5. Diponer de nuestros propios servidores de internet fuera de la LAN (DMZ)


RTB
Internet
Nec-SKY
Splitter
ADSL
172.16.0.1
LMDS
80.100.200.73
RDSI
Firewall
172.16.1.0/24
80.100.200.78
192.168.22.0/24
WI-FI
192.168.22.200
192.168.22.220
Access Point
Linksys - WRT54G
WPA-Radius (172.16.1.254)
IP: 172.16.1.253
IP2: 192.168.22.254
DHCP: 192.168.22.230-220
DMZ
80.100.200.72/29
80.100.200.75-76
192.168.21.0/24
80.100.200.75.77
192.168.21.10
192.168.21.11
PBX
Asterisk
PBX - Asterisk
Firewall
S2 y S3 = Servidores Internos
S0 y S1 = Servidor Internet

Funciones

1.Separar la DMZ de la LAN/WLAN
2.Enrutar tráfico según puerto destino
3.Evitar que los usuarios usen innecesariamente la conexión a internet principal.
4.Tráfico de usuarios por conexión a internet propia. (ADSL)
5. Servidores de internet, trabajan siempre por conexión principal. (LMDS)

Funciones II

6.No dejar pasar tráfico P2P
7.Centralizar gestión de Ips
8.Posibilidad de filtrar cualquier tipo de tráfico
9.Protejer servidores internos y de desarrollo
10.Servidor Radius con certificados para control de accesos de la WLAN.

Interfaces de Red

IP y máscaras de las interfaces del Firwall:

```
br0   inet addr:80.100.200.74  Bcast:80.100.200.79
Mask:255.255.255.248
br0:1 inet addr:80.100.200.77  Bcast:80.100.200.79
Mask:255.255.255.248
br0:2 inet addr:80.100.200.78  Bcast:80.100.200.79
Mask:255.255.255.248

eth0  NO IP
eth1  inet addr:172.16.0.2  Bcast:172.16.0.255
Mask:255.255.255.0
eth2  NO IP
eth3  inet addr:192.168.21.254  Bcast:192.168.21.255
Mask:255.255.255.0
eth4  inet addr:172.16.1.254  Bcast:172.16.1.255
Mask:255.255.255.0
```

Enrutamiento basico

```
f0 scripts # route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
80.100.200.72   0.0.0.0         255.255.255.248 U     0      0        0 br0
192.168.22.0    172.16.1.253    255.255.255.0   UG    0      0        0 eth4
192.168.21.0    0.0.0.0         255.255.255.0   U     0      0        0 eth3
172.16.0.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
172.16.1.0      0.0.0.0         255.255.255.0   U     0      0        0 eth4
127.0.0.0       127.0.0.1       255.0.0.0       UG    0      0        0 lo
0.0.0.0         80.100.200.73   0.0.0.0         UG    0      0        0 br0
```

Marcamos paquetes para enrutado

Con IPTABLES marcamos los paquetes que pasan por el firewall según nos convenga para:

- Enrutar según servicio
- QoS/ToS

Ejemplo:

```
iptables -A PREROUTING -t mangle -m mport -i eth4 -p tcp -s 192.168.22.0/24 \
                --dport 22 -j MARK --set-mark 10
```

Marcamos paquetes para enrutado

```
# Marca para tiempo real (RT)
MRT="10"
# Marca para servicios normals (MSERVEIS)
MSERVEIS="20"
# Marca para servicios de baja prioridad (MBAIXPRIO)
MBAIXAPRIO="30"

# Interficie a marcar els paquets
IF="eth4"
PROTOS="tcp udp"
# Xarxa d'usuaris
USERS_NET="192.168.22.0/24"

# Ports que han de funcionar a temps real
#         RDP  RA   PC-ANYWHE RTSP RTSP PPTP SIP  MSN/UDP
RT="22,53,3389,4899,5631,5632,554,7070,1723,5060,6801,6901"
#-------------------------------------------------------------------------
for TP in $PROTOS
do
        iptables -A PREROUTING -t mangle -m mport -i $IF -p $TP -s $USERS_NET \
                --dport $RT -j MARK --set-mark $MRT
done
```

Marcamos paquetes para enrutado

```
#-------------------------------------------------------------------------
#         RA canviats  VNC      X-Server  RTP/UDP    RTSP/UDP  MSN/TCP  MSN/UDP
RT_RANGS="55000:56000 5800:5910 6000:6010 16384:32767 6970:7170 6891:6901
2001:2120"
#-------------------------------------------------------------------------
for RT in $RT_RANGS
do
        for TP in $PROTOS
        do
                iptables -A PREROUTING -t mangle -i $IF -p $TP -s $USERS_NET \
                        --dport $RT -j MARK --set-mark $MRT
        done
done
#-------------------------------------------------------------------------
# Protocols que han de funcionar a temps real
RT_PROTO="50 51" # IPSec ESP,AH
#-------------------------------------------------------------------------
for P in $RT_PROTO
do
        iptables -A PREROUTING -t mangle -i $IF -p $P -s $USERS_NET -j MARK \
                        --set-mark $MRT
done
```

Marcamos paquetes para enrutado

```
#-----------------------------------------------------------------------
# Serveis amb prioritat normal
#SERVEIS="25,80,110,143,443,993,995,554,4080,873"
SERVEIS="80"
#-----------------------------------------------------------------------
for TP in $PROTOS
do
        iptables -A PREROUTING -t mangle -m mport -i $IF -p $TP -s $USERS_NET \
                --dport $SERVEIS -j MARK --set-mark $MSERVEIS
done

#-----------------------------------------------------------------------
SERVEIS_RANGS="8000:8088"
#-----------------------------------------------------------------------
for RT in $RT_RANGS
do
        for TP in $PROTOS
        do
                iptables -A PREROUTING -t mangle -i $IF -p $TP -s $USERS_NET \
                        --dport $SERVEIS_RANGS -j MARK --set-mark $MSERVEIS
        done
done
```

Marcamos paquetes para enrutado

```
#-----------------------------------------------------------------------
# Ports amb poca prioritat
BAIXAPRIO="20,21,115,873,161,163,1863"
#-----------------------------------------------------------------------
for TP in $PROTOS
do
        iptables -A PREROUTING -t mangle -m mport -i $IF -p $TP -s $USERS_NET \
                --dport $BAIXAPRIO -j MARK --set-mark $MBAIXAPRIO
done
#-----------------------------------------------------------------------
# Trafic P2P
iptables -A PREROUTING -t mangle -i $IF -m ipp2p --ipp2p -s 192.168.22.0/24 -j
MARK --set-mark $MBAIXAPRIO
```

Marcamos paquetes para enrutado (P2P)

· Marcamos paquetes con un módulo especial de IPTABLES el ipp2p
· Bloqueamos este tipo de tráfico sólo por el dia durante los dias laborables

```
# Creo una nova CHAIN anomenada DIA
iptables -N DIA
# Boqueijo trafic P2P en la nova CHAIN
iptables -A DIA -m ipp2p --ipp2p -s 192.168.22.0/24 -j DROP


# Estableixo quan s'han de complir les normes de la CHAIN anomenada DIA
IF="eth4"
DIES="Mon,Tue,Wed,Thu,Fri"
INICI="0800"
FINAL="2000"
iptables -A FORWARD -m time -i $IF -s 192.168.22.0/24 --days $DIES -timestart \
$INICI --timestop $FINAL -j DIA
```

Enrutado según las marcas

```
ADSL=121
LMDS=122

USUARIS_NET="192.168.22.0/24"
USUARIS_IP="172.16.1.254"
USUARIS_IF="eth4"
USUARIS_GW="172.16.1.253"

SINTRANET_NET="192.168.21.0/24"
SINTRANET_IP="192.168.21.254"
SINTRANET_IF="eth3"

ADSL_IP="172.16.0.2"
ADSL_GW="172.16.0.1"
ADSL_MASQ="255.255.255.0"
ADSL_NET="172.16.0.0/24"
ADSSL_IF="eth1"

LMDS_IP="80.100.200.74"
LMDS_GW="80.100.200.73"
LMDS_MASQ="255.255.255.248"
LMDS_IF="br0"
LMDS_NET="80.100.200.72/29"


INET="0.0.0.0/0"
```

Enrutado según las marcas

```
# Reglas de enrutado según marcas:
ip rule add fwmark 10 lookup lmds
ip rule add fwmark 20 lookup adsl
ip rule add fwmark 30 lookup adsl

# Definimos tabla de enrutado para ADSL
ip route add $LMDS_NET dev $LMDS_IF src $LMDS_IP table $ADSL
ip route add $USUARIS_NET dev $USUARIS_IF src $USUARIS_IP via $USUARIS_GW table
$ADSL
ip route add $ADSL_NET dev $ADSL_IF src $ADSL_IP table $ADSL
ip route add $SINTRANET_NET dev $SINTRANET_IF src $SINTRANET_IP table $ADSL
ip route add default via $ADSL_GW table $ADSL

# Definimos tabla de enrutado para LMDS:
ip route add $ADSL_NET dev $ADSL_IF src $ADSL_IP table $LMDS
ip route add $USUARIS_NET dev $USUARIS_IF src $USUARIS_IP via $USUARIS_GW table
$LMDS
ip route add $LMDS_NET dev $LMDS_IF src $LMDS_IP table $LMDS
ip route add $SINTRANET_NET dev $SINTRANET_IF src $SINTRANET_IP table $LMDS
ip route add default via $LMDS_GW table $LMDS

# Forzamos el caché
ip route flush cache
```

Tablas de enrutado v2 (enrutado avanzado)

```
f0 scripts # ip route show table lmds
80.100.200.72/29 dev br0  scope link  src 80.100.200.74
192.168.22.0/24 via 172.16.1.253 dev eth4  src 172.16.1.254
192.168.21.0/24 dev eth3  scope link  src 192.168.21.254
default via 80.100.200.73 dev br0


f0 scripts # ip route show table adsl
80.100.200.72/29 dev br0  scope link  src 80.100.200.74
192.168.22.0/24 via 172.16.1.253 dev eth4  src 172.16.1.254
192.168.21.0/24 dev eth3  scope link  src 192.168.21.254
default via 172.16.0.1 dev eth1


f0 scripts # ip route
80.100.200.72/29 dev br0  proto kernel  scope link  src 80.100.200.74
192.168.22.0/24 via 172.16.1.253 dev eth4
192.168.21.0/24 dev eth3  proto kernel  scope link  src 192.168.21.254
172.16.0.0/24 dev eth1  proto kernel  scope link  src 172.16.0.2
172.16.1.0/24 dev eth4  proto kernel  scope link  src 172.16.1.254
127.0.0.0/8 via 127.0.0.1 dev lo  scope link
default via 80.100.200.73 dev br0
```

SNAT
(para salir a internet)

· Definimos con que IP pública se navega por internet.

· Los usuarios navegan por internet con la IP .78:

```
iptables -t nat -A POSTROUTING -s 192.168.22.0/24 -d ! 80.100.200.72/29 -o br0
-j SNAT --to-source 80.100.200.78
```

· Los servidores de desarrollo salen a internet enmascarados como la IP .77:

```
iptables -t nat -A POSTROUTING -s 192.168.21.0/24 -d ! 80.100.200.72/29 -o br0
-j SNAT --to-source 80.100.200.77
```

· Esquema final:.

```
80.100.200.77 -> Xarxa: 192.168.21.0/24 -> Servidores Intranet
80.100.200.78 -> Xarxa: 192.168.22.0/24 -> Usuarios
```

DNAT
(redirigir puertos)

```
# Redirecciones de puertos para SERVIDORS INTERNOS (192.168.21.10)
PORTS_TCP="80 22 53 21 20 8080 5432"
PORTS_UDP="53 21 20"
for P in $PORTS_TCP
do
        iptables -t nat -A PREROUTING -p tcp -i br0 -d 80.100.200.77 --dport $P
-j DNAT --to 192.168.21.10:$P
done

for P in $PORTS_UDP
do
        iptables -t nat -A PREROUTING -p udp -i br0 -d 80.100.200.77 --dport $P
-j DNAT --to 192.168.21.10:$P
done
```

DNAT
(redirigir puertos)

- Redirección de puertos para PCs de usuarios
- No es recomanable usar esta prácitca, ya que insecuriza la red
- Mejor usar VPNs

per ejemplo:

```
iptables -t nat -A PREROUTING -p tcp -i br0 -d 80.100.200.78 --dport 22 -j DNAT
--to 192.168.22.111:22
```

FireWalling

```
# Filtrando puertos de las IPs de los servidores de internet (DMZ)
IPS="80.100.200.75 80.100.200.76"
PROTOS="tcp udp"
SERVEIS_TCP="25,80,53,110,5432,21,20,22"
SERVEIS_TCP_RANGS="8000:8100"
SERVEIS_UDP="53,161,163"
IPT="iptables -A FORWARD -i br0"


for IP in $IPS
do
        # Filtros TCP
        $IPT -m mport -p tcp -d $IP --dport $SERVEIS_TCP -j ACCEPT
        # Filtros a RANGOS de puertos TCP
        for P in $SERVEIS_TCP_RANGS
        do
                $IPT -p tcp -d $IP --dport $P -j ACCEPT
        done
        # Filtros UDP
        $IPT -m mport -p udp -d $IP --dport $SERVEIS_UDP -j ACCEPT
        # Filtros a RANGOS de puerts UDP
        #for P in $SERVEIS_UDP_RANGS
        #do
                #$IPT -p udp -d $IP --dport $P -j ACCEPT
        #done
```

FireWalling

```
        # Permitir que las conexiones establecidas/relacionadas funcionen
(ex.FTP)
        iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

        # Permitimos acceso de las redes internas a Internet
        $IPT -s 192.168.22.0/24 -d $IP -j ACCEPT
        $IPT -s 192.168.21.0/24 -d $IP -j ACCEPT
        $IPT -s $IP -d 192.168.22.0/24 -j ACCEPT
        $IPT -s $IP -d 192.168.21.0/24 -j ACCEPT

        # Permitimos tráfico ICMP
        $IPT -p icmp -s $IP -j ACCEPT
        $IPT -p icmp -d $IP -j ACCEPT

        # Permitimos acceso a internet de los servidores
        $IPT -s $IP -j ACCEPT
        $IPT -p tcp -d $IP --dport 1024: ! --syn -j ACCEPT
        $IPT -p udp -d $IP --dport 1024: -j ACCEPT

        # Default policy: DROP!!!
        $IPT -d $IP -j DROP

done
```

Acceso por WLAN seguro

* Instalar freeRADIUS al firewall: emerge freeRADIUS
* Configuramos el freeRADIUS para trabajar con EAP/TLS
* Generamos los certificados (PKI)
* Comprobamos que el freeRADIUS funciona correctamente
* Configuramos el AP (Linksys WRT54G)
* Configuramos los clientes Windows XP con SP2
* Configuramos los clientes con Linux

Basado en la siguiente documentación:

· Català: http://oriol.joor.net/blog/?itemid=1574

· English: http://oriol.joor.net/blog/?itemid=1631

Qué es EAP-TLS?

· EAP-TLS es un estándard abierto de la IETF
· Soportado por muchos fabricantes de wireless.
· Buen nivel de seguridad ya que el TLS se considera el sucesor del SSL, o sea, que es más seguro que este.
· Se usa PKI para la conexión con RADIUS.
· Es un poco difícil poner en marcha el sistema PKI.
· Vale la pena porque nos da un gran nivel de seguridad.
· En algunos casos no es trivial configurar los clientes.

freeRADIUS con EAP/TLS

Solo hay que configurar dos ficheros el resto pueden estar vacios:

radiusd.conf: la mayor parte del fichero la podemos dejar por defecto, sólo debemos configurar la parte EAP y la parte TLS. Y las configuraciones generales que nos interesen.
http://oriol.joor.net/article_fitxers/1574/radiusd.conf

clients.conf: definimos quien se puede conectar a nuestro servidor Radius. O sea, el AP y asignamos una pre-shared key para securizar el enlace entre RADIUS y AP.
http://oriol.joor.net/article_fitxers/1574/clients.conf

freeRADIUS con EAP/TLS

Parte más importante de la configuración de radiusd.conf:

```
eap {
    default_eap_type = tls
    timer_expire     = 60
    ignore_unknown_eap_types = no

    tls {
        private_key_password = algunacontrasena
        private_key_file = /etc/raddb/certs/cert-srv.pem
        certificate_file = /etc/raddb/certs/cert-srv.pem
        CA_file = /etc/raddb/certs/demoCA/cacert.pem
        dh_file = /etc/raddb/certs/dh
        random_file = /etc/raddb/certs/random
        fragment_size = 1024
        include_length = yes
    }
}
```

Generamos certificados (PKI)

· Generamos certificados para el servidor y para el cliente.
· Necesitamos OpenSSL
· Hace falta tener el CA.pl en el PATH del sistema. Este fichero se incluye en los fuentes del OpenSSL. Pero no se copia automáticamente en ninguna carpeta que este en el PATH.
· Script para generar los certificados automáticamente.
http://oriol.joor.net/article_fitxers/1574/certs.tar.gz

Generamos certificados (PKI)

· Creamos un nuevo directorio descomprimimos el script para generar los certificados en ese directorio y ejecutamos ./certs.sh
· Ahora se ha creado un directorio que se llama certs/ con los certificados en su interior.
· Copiamos ese contenido a /etc/raddb/certs.
· Configuramos adecuadamente los parámetros de radiusd.conf para acceder a los certificados.
· Para lanzar el demonio depurando su salida hay que lanzarlo así: radiusd -X

Configuramos el AP

· En el ejemplo vemos como se configura el Linksys WRT54G con el firmware de HyperWRT:

TKIP (Temporal Key Integrity Protocol): definido en 802.11i para sustituir WEP sin cambiar el hardware. Usa RC4 igual que WEP pero para cada paquete usando su única clave de cifrado.

Configurando clientes para la WLAN

· Manual para configurar clientes WXP con SP2:

http://oriol.joor.net/article_fitxers/1574/WXP-WAP-EAPTLS.pd
esta en catalán, pero este documento se basa en:
http://oriol.joor.net/article_fitxers/1574/EAPTLS.pdf

·Configurar clientes en Linux con el aplicativo WPA_Supplicant:
http://oriol.joor.net/blog/?itemid=1579
este manual esta en catalán, y no sólo eseña a configurar el WPA supllicant sinó también en NDISWRAPPER para tarjetas sin driver nativo en linux. Como es mi caso la: Broadcom BCM4306 802.11b/g

Configurar WPA_Supplicant

- Fichero de configuración /etc/wpa_supplicant.conf:

```
network={
    ssid="ssiddelared"
    proto=WPA
    key_mgmt=WPA-EAP
    pairwise=TKIP
    group=TKIP
    eap=TLS
    identity="exemple@exemple.com"
    ca_cert="/etc/cert/root.pem"
    client_cert="/etc/cert/root.pem"
    private_key="/etc/cert/root.p12"
    private_key_passwd="whatever"
}
```

WPA_Supplicant

· Lanzando el cliente en modo depuración:

```
wpa_supplicant -dd -iwlan0 -c/etc/wpa_supplicant.conf
-Dndiswrapper
```

· Lanzando el cliente:

```
wpa_supplicant -iwlan0 -c/etc/wpa_supplicant.conf
-Dndiswrapper
```

· En Gentoo hay un fichero de configuración para estos parámetros:
/etc/conf.d/wpa_supplicant y luego podemos lanzar el servicio como demonio de sistema: /etc/init.d/wpa_supplicant start

Conclusiones

· Antes de empezar a diseñar una red hay que pensar un buen rato
· No hay que tener prisa en hacer las cosas
· Es muy importante ser paciente
· Con una buena base y unas buenas referencias todos podemos montar un buen firewall y una WLAN segura
· No conozco sistemas más sencillos que garanticen un nivel de seguridad suficiente para una empresa con el software/hardware de WLAN actual
· A pesar de que los sistemas son mejorables mi experiencia me dice que el sistema descrito es suficiente en muchos entornos.

DUDAS & PREGUNTAS

MUCHAS GRACIAS

Oriol Rius | oriol@joor.net | http://oriol.joor.net