



Instalação e configuração.

VPN (Virtual Private Network, ou rede virtual privada) é, como o nome sugere, uma rede virtual, criada para interligar duas redes distantes, através da Internet. Usar uma VPN permite que você compartilhe arquivos e usar aplicativos de produtividade e de gerenciamento, como se todos os micros estivessem conectados à mesma rede local. Você pode até mesmo imprimir em impressoras da rede remota, da mesma forma que faria com uma impressora local.

Antigamente, a única forma de interligar redes em dois locais diferentes era usar linhas de frame-relay. Para quem não é dessa época, uma linha de frame-relay oferece um link dedicado de 64 kbits entre dois pontos (a matriz e a filial de uma empresa, por exemplo), com um custo mensal fixo. Você pode combinar várias linhas frame-relay em uma única conexão, de acordo com a necessidade, o problema nesse caso é o preço. Atualmente, existem outras opções de conexões dedicadas e de conexões de alta disponibilidade, oferecidas pelas operadoras, mas elas ficam fora do orçamento da maioria das pequenas e médias empresas.

Usando uma VPN, você pode obter todos os benefícios de ter uma conexão dedicada entre dois pontos usando conexões via ADSL ou cabo, que são muito mais baratas. Dependendo do volume de uso da rede, você pode tanto utilizar a conexão que já tem quanto utilizar uma segunda conexão apenas para a VPN, evitando assim que o tráfego prejudique a navegação dos usuários. Além de interligar duas ou mais redes, a VPN pode ser acessada também por computadores (notebook) remotamente de qualquer local com internet.

Todos dados que trafegam através da VPN são criptografados, o que elimina o risco inerente à transmissão via Internet. Naturalmente, nenhuma VPN pode ser considerada "100% segura", já que sempre existe um pequeno risco de alguém conseguir obter acesso aos servidores, de forma a roubar as chaves de encriptação (por exemplo), mas, em uma VPN bem configurada, o risco é realmente muito pequeno. É muito mais provável que um funcionário descontente sabote a rede interna, ou envie informações sigilosas para fora, do que algum invasor consiga efetivamente obter acesso à rede via Internet.

Quem hoje usa vpn: Governo Federal, Estadual e algumas cidades, Exercito, Policia Federal, Bancos, Empresas Aéreas, Grandes Redes de Lojas e Magazines.

Nesse tópico, estudaremos como criar VPNs usando o OpenVPN.

Além de ser mais simples de configurar que a maioria das outras soluções de VPN, e bastante seguro e flexível, existe versões Linux e Windows, o que permite trabalhar com os dois sistemas junto na mesma rede.

Outras vantagens são que ele pode ser usado por clientes conectando através de uma conexão compartilhada via NAT (apenas o servidor precisa de portas abertas) e a boa tolerância contra conexões ruins, ou ao uso de conexões com IP dinâmico. A VPN pode ser configurada para ser restabelecido de forma automática em caso de interrupção na conexão, o que torna o link bastante confiável.

Com relação à segurança, o OpenVPN pode trabalhar de forma aberta sem verificação de senha ou utilizar chaves estáticas de 2048 bit(medio), ou certificados X509, onde a configuração é um pouco mais complexa, mas, em compensação, a segurança é muito maior (bem superior à da maioria das soluções comerciais). Isso permite que você escolha a melhor relação entre praticidade e segurança de acordo com a situação.

**Instalando o OpenVPN e criando uma VPN básica**

O primeiro passo é instalar o OpenVPN, tanto no cliente quanto no servidor. Nas distribuições derivadas do **Debian:**

# apt-get install openvpn

**OpenSuSE:**

Yast e, dentro da seção de gerenciamento de pacotes, procure pelo pacote openvpn.

Resultado: "openvpn"e "openvpn-devel". Basta selecionar os pacotes e concluir a instalação.

**Fedora, Red hat, CentOs:**

# yum install openvpn.

Pode ser necessário fazer antes a instalação do pacote **epel-release-6-8.noarch.rpm**

Presumindo que as duas máquinas estejam ligadas em rede e que não exista nenhum firewall no caminho, use o comando abaixo no cliente, especificando o endereço do servidor:

# openvpn --remote 192.168.1.1 --dev tun0 --ifconfig 10.0.0.2 10.0.0.1

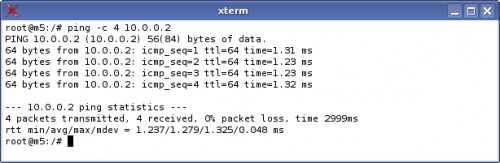
Faça o mesmo no servidor, especificando agora o endereço do cliente:

# openvpn --remote 192.168.1.202 --dev tun0 --ifconfig 10.0.0.1 10.0.0.2

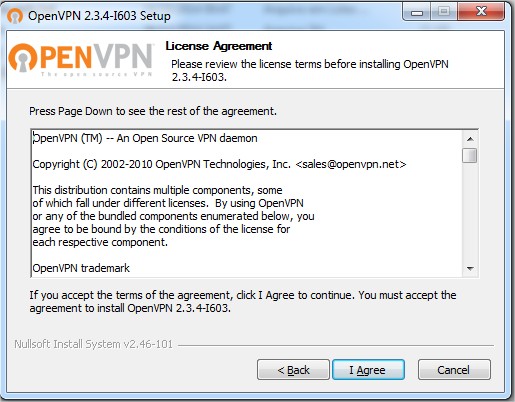
Rode o comando "ifconfig" no cliente e você verá que foi criada uma interface de rede virtual, a "tun0", com o endereço "10.0.0.1":



Servidor executando o ping no cliente.



Instalador para Windows, ultima versão é 2.3.4, ao final da instalação será instado um drive para criar o túnel de rede virtual para a conexão.



**Criando VPNs usando chaves estáticas**

A forma mais simples de configurar o OpenVPN é utilizar chaves estáticas, onde um arquivo contendo um algoritmo de encriptação é usado pelas duas partes através da VPN.

Nesse modo, você gera uma arquivo de texto no servidor, contendo a chave de encriptação e precisa apenas copiá-la para o cliente e fazer uma configuração rápida para criar a VPN.

O grande problema em utilizar uma chave estática é que o servidor suportará a conexão de um único cliente por VPN.

Para VPNs com vários clientes, ou em redes empresariais, onde a segurança é prioridade, é recomendável utilizar uma estrutura baseada em certificados X509, como veremos no tópico seguinte.

O primeiro passo é gerar o arquivo com a chave, no servidor. O arquivo deve ser gerado dentro do diretório "/etc/openvpn" (o diretório padrão de configuração), de forma que acessamos o diretório antes de executar o comando que gera a chave:

*# cd /etc/openvpn*

*# openvpn --genkey --secret senha.key*

Isso gerará o arquivo "senha.key", que contém a chave de encriptação que será usada para criar a conexão. Ele é um arquivo de texto simples, que contém uma chave de 2048 bits:



Este arquivo precisa estar no servidor e no cliente. E recomendável se colocar no seguinte diretório:  
/etc/openvpn/keys

**Servidor :**/etc/openvpn/server.conf  
*dev tun  
ifconfig 10.50.0.1 10.50.0.2  
secret senha.key*

**Cliente:**  
/etc/openvpn/client.conf  
*remote 192.168.1.254  
dev tun  
ifconfig 10.50.0.2 10.50.0.1  
secret senha.key*

A primeira linha do cliente se refere ao ip do servidor, lembre-se que em rede externa será necessário um ip fixo ou (DDNS/NO-IP) também temos que direcionar a porta padrão da vpn (1194 UDP) do modem para o servidor.

A openvpn trabalha como serviço, sendo assim:

**Iniciar**  *service openvpn start*

**Parar**  *service openvpn stop*

Podemos trocar a porta da VPN e o seu protocolo de comunicação, para isso temos que adicionar as seguintes linhas:

proto udp

port 1194

**Servidor : Cliente:***proto tcp-server proto tcp-client  
port 29999 port 29999*

Mas a VPN tem um desempenho e melhor em UDP. Isso devido às correções de erro e checagem de pacotes e até a retransmissão de pacotes que ocorrem no TCP.

**Acesso a rede do servidor (roteamento)**

Originalmente, o cliente terá acesso apenas ao próprio servidor. Para que ele tenha acesso aos demais hosts da rede, é necessário realizar alguns passos adicionais.

O primeiro é ativar o roteamento de pacotes no servidor, de forma que ele passe a encaminhar os pacotes recebidos do cliente para a interface de rede local. Mesmo que o servidor possua uma única placa de rede, a VPN é vista pelo sistema como uma interface separada, a VPN e a rede local são duas placas de rede separadas. (ethx e tun)

O roteamento é feito usando dois comandos (executados no servidor):

***echo 1 > /proc/sys/net/ipv4/ip\_forward***

***iptables -t nat -s 10.50.0.2 -A POSTROUTING -o eth0 -j MASQUERADE***

O primeiro, ativa o roteamento de pacotes no Kernel, e o segundo orienta o sistema a encaminhar os pacotes recebidos a partir do endereço IP do cliente 10.50.0.2, para a interface de rede local. Estes dois comandos precisam ser executado junto com a inicialização do sistema **(/etc/rc.local)** para que sejam executados a cada boot.

O próximo passo é no cliente.

**route add -net 192.168.1.0 netmask 255.255.255.0 gw 10.50.0.1 dev tun0**

Adicionar uma rota de saída, assim quando for fazer acesso a algum IP da faixa **“192.168.1.0”**, que esta do outro lado da vpn (rede do servidor), o nosso micro vai fazer o acesso através do túnel.

Obs: Lembre-se que cada rede *“cliente”* e *“servidor”* tem de ter faixa de *“IP”* diferente, sendo assim o nosso *“servidor” 192.168.1.254* e o *“cliente” 192.168.2.21*

**Parâmetros**

**Keepalive** faz com que o servidor e o cliente monitorem a conexão, enviando pings periódicos, e reinicie a VPN automaticamente caso não recebam resposta dentro de um período determinado (segundos).  
 Esta opção é importante, afinal as conexões de internet são instáveis.

***Ex: keepalive 10 120***

“keepalive” comando, “10” intervalo entre cada ping, “120” timeout (se a vpn não tiver resposta do ping após esse tempo ela é reiniciada automaticamente.

**Inactive,** especifica um tempo maximo (segundos) para tentativas de re-conexão, após esse tempo o cliente desiste da conexão automaticamente.

***Ex:inactive 3600***

**Comp-lzo,** ativa a compactação dos dados transmitidos através do túnel (é necessário que o pacote "lzo" esteja instalado).Isso vai ajudar a ficar mais rápida a sua comunicação. Esta opção deve ser incluída tanto no servidor quanto no cliente.

***Ex:comp-lzo***

**Persistkey** e **Persisttun**. Elas fazem com que o vpn mantenha a interface tun aberta e as chaves carregadas quando é reiniciado, o que torna mais rápida a restauração da conexão.

***Ex:  
persist-key  
persist-tun***

**Float,** que permite que o túnel continue aberto mesmo que o endereço IP da outra máquina mude. Em situações normais, a mudança de IP faria com que a conexão fosse encerrada e o túnel fosse interrompido até que fosse reiniciado (seja manualmente ou seja pelo uso da opção keepalive). Essa opção deve ser incluída tanto na configuração do servidor quanto na do cliente:

***Ex:float***

**Server**, esse comando substitui o “*ifconfig 10.50.0.1 10.50.0.2”*, sendo assim vamos apenas dizer qual será o “ip” e a mascara de rede do servidor, o próprio servidor vai dar os endereços de ip as estações assim que elas forem se conectando a “vpn”, lembrando que o 1 ip e do servidor 10.50.0.1

**Ex: server 10.50.0.0 255.255.255.0**

* Somente no servidor.

Obs: nesse caso o cliente precisa ter “**client”** no lugar do “*ifconfig 10.50.0.2 10.50.0.1”*

***Ex: client***

* Somente nas estações.

**Push "route ip masc”,** criando uma rota automaticamente, já fornecida pelo servidor da vpn, assim eliminamos o comando “**route add -net 192.168.1.0 netmask 255.255.255.0 gw 10.50.0.1 dev tun0”** que vimos anteriormente.

**Ex: push "route 192.168.1.0 255.255.255.0"**

* Somente no servidor.

**ifconfig-pool-persist**  Ele faz com que o OpenVPN passe a armazenar uma lista com os endereços IP usados por cada cliente da VPN e faça o possível para atribuir sempre os mesmos endereços em futuras conexões. Isso não impede que os endereços mudem, mas torna as mudanças muito menos freqüentes. Especificando um arquivo de texto onde a lista será salva, como em:

**Ex: ifconfig-pool-persist /etc/openvpn/ipp.txt**

* Somente no servidor.

Com essa regra já temos a rota pronta para a rede do servidor, mas para que o cliente aceite o comando de rota nos precisamos adicionar o seguinte comando no cliente **“pull” (puxar)**, além da rota ele vai trazer o endereço IP da maquina qual o servidor vai liberar (10.0.50.2)

**Ex: pull**

* Somente nas estações.

**Remote-cert-tls server**, que faz com que os clientes verifiquem o certificado do servidor no momento da conexão. Ela é mais uma opção destinada a proteger os clientes contra ataques man-in-the-middle. Esta opção é adicionada apenas na configuração dos clientes:

**Ex: remote-cert-tls server**

* Somente nas estações.

**Shaper**, controle de limite da banda total de saída usada pela VPN a um determinado volume de tráfego. Se você usa um link ADSL com 2 mb (ou seja, 256 kbytes) de upload, você poderia restringir o uso de banda pela VPN a 200 kbytes, por exemplo, de forma a deixar pelo menos uma pequena porção da banda reservada para outros usos.

A opção **"shaper**" pode ser incluída na configuração do servidor (para que seja aplicada ao tráfego de saída somado de todos os clientes conectados a ele), na configuração dos clientes (limitando assim o tráfego de saída permitido por cada um) ou em ambos. Basta especificar o limite desejado, em bytes, como em:

**Ex: shaper 204800**

* Somente no servidor.

**Max-clients**, você pode também limitar o número de clientes simultâneos que serão aceitos pelo servidor usando a opção "max-clients", especificando o número de clientes desejados.

**Ex: max-clients 100**

* Somente no servidor.

**duplicate-cn**, permite varias conexões com o mesmo certificado.

* Somente no servidor.

**Remote-random,** em um servidor com várias conexões de internet, ou no caso de uma rede com vários servidores de VPN (onde os clientes podem se conectar a qualquer um dos servidores para obter acesso à rede), você pode criar um sistema simples de redundância e de balanceamento de carga especificando os endereços de todos os servidores na configuração dos clientes (criando várias linhas "remote") e adicionando a opção "remote-random".

Ex:

**remote srv.no-ip.org**

**remote srv.dyndns.org**

**remote srv.com.br**

**remote-random**

Isso faz com que o cliente escolha aleatoriamente entre os três endereços especificados a cada conexão, tentando os outros dois caso o primeiro esteja inacessível. Como cada cliente escolherá um servidor diferente a cada conexão, a carga acabará sendo dividida igualmente entre os servidores.

**CRIANDO CERTIFICADOS**

Para criar os certificados, utilizaremos o easy-rsa, um conjunto de scripts incluídos no pacote do OpenVPN. Na maioria das distribuições, os scripts estão disponíveis dentro da pasta "/usr/share/doc/openvpn/examples/easy-rsa". Se a pasta não existir na sua instalação, use os comandos abaixo para localizar a pasta:

# updatedb

# locate easy-rsa

**Ou faça download em** [**ftp://200.146.119.59/OpenVpn/easy-rsa.arj**](ftp://200.146.119.59/OpenVpn/easy-rsa.arj)

No CentOs é usada a pasta "/usr/share/doc/openvpn-2.0.9/easy-rsa/" (onde o 2.0.9 é a versão do pacote).

Temos que copiar a pasta para o local de trabalho, com o seguinte comando:

***cp –a /usr/shared/doc/ openvpn-2.0.9/easy-rsa/ /etc/openvpn/easy-rsa/***

ou descompactar o arquivo easy-rsa.arj

***arj x easy-rsa.arj /etc/openvpn/easy-rsa -y***

A partir daí, toda a configuração será feita dentro do novo diretório:

*cd /etc/openvpn/easy-rsa*

Comece editando o arquivo "vars" usando seu editor de textos preferido:

*vi vars*

No final do arquivo, você encontra um conjunto de parâmetros usados para gerar as chaves (país, estado, cidade, empresa, etc.), que devem ser editados, como em:

export KEY\_COUNTRY=BR

export KEY\_PROVINCE=PR

export KEY\_CITY="Maringa"

export KEY\_ORG="SG Sistemas"

export KEY\_EMAIL="eu@sgsistemas.com.br"

Agora vamos aos comando para criar os arquivos de chaves.

*source vars # Carega as variáveis do arquivo vars*

*./clean-all # Limpa qualquer sobra da configuração anterior*

*./build-ca #Gera certificado raiz com SSL.*

Isso criará a pasta "/etc/openvpn/easy-rsa/keys", contendo os seguintes arquivos:

ca.crt ca.key index.txt serial

O arquivo "ca.crt" contém o certificado raiz. Ele jamais deve sair do servidor.

Depois de criar o certificado raiz, podemos passar para o para o passo seguinte, que é gerar os certificados para o servidor e para os clientes.

***Certificado do servidor***, usamos o script "build-key-server", especificando como parâmetro o nome do arquivo que será gerado ("servidor" no exemplo).

*./build-key-server servidor*

Obs: Vai aparecer uma opção para colocar uma senha como opção adicional de segurança, passar em branco. Caso contrario vai pedir a senha toda vez que fizer a conexão.

***Certificado do cliente,*** usando o script "build-key". Você pode gerar chaves para vários clientes em seqüência, especificando nomes diferentes, como em:

./build-key filial

./build-key filial2

Obs: Pode usar a mesma chave para acesso de todos.

***Certificado build-dh***, ele gera os parâmetros Diffie Hellman, que reforçam a segurança, permitindo que o servidor e os clientes troquem as chaves (de forma segura) no momento da conexão, confirmando suas identidades:

./build-dh

O processo de geração das chaves gera um conjunto de arquivos ".csr". Estes arquivos correspondem às requisições para a assinatura das chaves, que é feita durante o processo de criação. Eles podem ser removidos no final do processo:

rm \*.csr

Caso esteja curioso, estes scripts nada mais fazem do que executar os comandos correspondentes do OpenSSL, usando os parâmetros definidos no arquivo "vars.

Obs: sua chave padrão tem validade de 10 anos ou seja 3650 dias. Se a sua VPN ainda estiver em uso até lá, não se esqueça de trocar as chaves antes do 3650° dia, caso contrário sua VPN parará de funcionar após completar o décimo aniversário.

Agora fazer a copia das chaves para o servidor e os clientes:

***Servidor:***

"ca.crt", "servidor.crt", "servidor.key" e “dh1024.pem”

cd /etc/openvpn/easy-rsa/keys

mkdir /etc/openvpn/keys

cp -a ca.crt servidor.crt servidor.key dh1024.pem /etc/openvpn/keys/

***Cliente:***

"ca.crt", "dh1024.pem" , “filial.crt” e “filial.key”

Nos clientes Linux, crie o diretório "/etc/openvpn/keys".

Windows “C:\Program Files (x86)\OpenVPN\config\keys”

Obs: no windows o arquivos de configuração que seria o ***client.conf*** passa a ser ***client.ovpn.***

Exemplo final de configuração da OpenVPN.

**Configuração do Servidor.**

server 10.50.0.0 255.255.255.0

*dev tun*

*proto udp*

*port 29999*

*keepalive 10 120*

*comp-lzo*

*persist-key*

*persist-tun*

*float*

dh keys/dh1024.pem

ca keys/ca.crt

cert keys/servidor.crt

key keys/servidor.key

ifconfig-pool-persist ipp.txt

push "route 10.0.0.0 255.255.255.0"

duplicate-cn

status openvpn-status.log

log openvpn.log

verb 3

Configuração cliente é praticamente a mesma, com a adição da linha **"remote"**, que especifica o endereço do servidor e a mudança na linha **"ifconfig"**, onde a ordem dos endereços são trocados:

client

*remote 10.0.0.100 1194*

*dev tun*

*proto udp*

*port 29999*

*keepalive 10 120*

*comp-lzo*

*persist-key*

*persist-tun*

*float*

dh keys/dh1024.pem

ca keys/ca.crt

cert keys/filial.crt

key keys/filial.key

pull

remote-cert-tls server

verb 3