Virtual Private Networking

Linux Systems Security

Virtual Private Network

- Provides a method of extending access to one or more internal hosts or networks, using a public network
- Enables the use of firewalled internal services (e.g. DNS, web, file sharing, authentication) without having to configure the firewall to provide access to each service
- Enables the use of LAN protocols securely with remote clients using encryption to prevent eavesdropping and detect tampering
- Can be used by bad actors to exfiltrate data and bypass firewall restrictions

VPN Implementations

- Various protocols for VPNs have been defined
- IPsec IETF RFC 6434, designed for and with IPv6 but works with IPv4 with some problems, provides security at the transport level, requires routable IPs
- SSL/TLS VPNs (e.g. OpenVPN, SoftEther VPN), simplified universal offshoot of IPSEC that uses TLS
- DTLS (e.g. Cisco AnyConnect VPN, OpenConnect VPN), similar to TLS but run over UDP and designed for streaming
- MPPE/PPTP (e.g. Windows feature), Microsoft product, old, vulnerable, replaced by L2TP
- SSH (e.g. OpenSSH tunnels), alternative to OpenVPN/OpenConnect VPN that doesn't require root but is worse than other solutions in every other way
- Wireguard, proposed replacement for all other VPNs, it's great, it's amazing, it's fairly new, it ignores all the hard stuff so you have to solve those things outside of wireguard

OpenVPN

- VPN implementation with commercial and community versions, community version is under GPL
- Certificate and key based, can also use a pre-shared private key for connection establishment and/or login/password
- PAM can be added to OpenVPN configurations to support multi-factor authentication (e.g. shared secret key plus login/password)
- Client software available for all popular platforms
- Server version available for many platforms, also available as a commercial VM VHD and in commercial cloud-aware configurations from openvpn.net

OpenVPN Package

- Ubuntu and Debian repositories have openvpn community edition packages, but are behind the release schedule (not good in the security world)
- Add the openvpn repositories to your machine to install current software from the openvpn.net site, install easy-rsa if you want to use it to set up certificates for openvpn use (see https://community.openvpn.net/openvpn/wiki/OpenvpnSoftwareRepos for instructions on adding repos for openvpn to ubuntu)
- Packages do set up to establish vpn services at boot based on conf files

OpenVPN Certificate Authority

- Certificates are used to authenticate the openvpn server and client to each other, 2-way TLS authentication
- Genuine certificates and a real CA can be used as long as the client and server both use the same CA, but additional authentication factors (e.g. tls-auth, login/password) should be used
- A private CA can be generated and used by the server to sign certificates, servers and clients signed by that CA can trust each other, this is the preferred method
- easy-rsa is a package of scripts designed to simplify the management of this limited function CA
- see https://help.ubuntu.com/lts/serverguide/certificates-and-security.html for an alternate way to set up a certificate authority for internal use

Creating A Private CA

- Install easy-rsa
  - apt install easy-rsa
- Set up your private CA
  - make-cadir /etc/openvpn/easy-rsa
- Initialize your PKI database
  - cd /etc/openvpn/easy-rsa; ./easyrsa init-pki
- Create the CA key and certificate
  - cd /etc/openvpn/easy-rsa; ./easyrsa build-ca

Creating VPN Server Files

- Use the easyrsa script to create the keys and signed certificate for the VPN server
  - cd /etc/openvpn/easy-rsa ;./build-server-full *servername*
- Use openssl to create the DH parameters (required) and also a tls auth key (optional) for the VPN server
  - openssl dhparam -out /etc/openvpn/dh2048.pem 2048
  - openssl --genkey --secret /etc/openvpn/ta.key
- Copy the vpn server's files to the openvpn directory
  - cd /etc/openvpn
  - ln pki/ca.crt
  - ln pki/issued/*servername*.crt
  - ln pki/private/*servername*.key

OpenVPN Server Configuration

- Copy the example server.conf file from the /usr/share/doc/openvpn/examples directory and modify it
  - gzip -d </usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz >/etc/openvpn/*servername*.conf
- Modify the new server config file to specify the interface to listen on as well as the names of the server key/cert/dh/ta files already generated
- Enable ip forwarding in the kernel
  - edit /etc/sysctl.conf to set ip_forward
  - sysctl -w net.ipv4.ip_forward=1
- Be sure to forward your UDP port as necessary through firewalls, and allow it through your iptables firewall if you are using one

Additional OpenVPN Server Options

- Choose a local address for the server to listen on or leave it at the default of all addresses
- The default port of 1194 can be changed if desired, UDP is the preferred protocol for performance reasons
- Set up tunnel or tap device as desired, with matching server or server-bridge if you want to route the external host to a LAN or other networks
- Add any push options for routing or dhcp as required

OpenVPN Service Control

- Test your configuration using
  - openvpn --script-security 2 --config server.conf
- Once the test is successful, run the server automatically using
  - systemctl start openvpn@servername
- daemon log messages get sent to syslog and are placed in /var/log/syslog by default
  - grep ovpn- /var/log/syslog
- There is also a status file continuously updated in /var/log/openvpn/openvpn-status.log

OpenVPN Client Configuration

- Each client requires a key, a certificate signed by your private CA, the tls-auth shared key file, and the CA certificate from your private CA
- You can generate the client certificate and key on the server trivially by using the easy-rsa scripts
  - cd /etc/openvpn/easy-rsa; ./build-client-full *clientname*
- Install openvpn on the client host
- Securely transfer the necessary files (ca.crt, ta.key, *clientname*.crt, *clientname*.key) to the client
- You could also generate a csr on the client, import it to the private CA, and sign it with the private CA instead of building the whole thing on the private CA

OpenVPN Client Configuration

- Copy the example client.conf file from the /usr/share/doc/openvpn/examples directory and modify it
  - cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf /etc/openvpn/*client*.conf
- Modify the new client config file to specify the server to connect to, as well as the names of the server key/cert/dh/ta files already generated
- If the client is not running Linux, create a client.ovpn by adding <ca><key><cert><tls-auth> and key-direction 1 sections to the client.conf file and send that to the client
- Once the test is successful, run the client automatically using
  - systemctl start openvpn@client
- The logs are the same for the client as the server

Adding a Login/Password to OpenVPN Authentication

- Add plugin /usr/lib/openvpn/openvpn-plugin-auth-**pam**.so login to the server.conf file
- The server can also use client-cert-not-required and remove tls-auth if desired, although these actions reduce your protection against fraudulent authentication
- The client.conf must use auth-user-pass in order to request a login from the user
- You can also add smart card support with the PKCS#11 libraries

Other VPN Solutions

- OpenConnect http://www.infradead.org/ocserv/
- SSH VPN https://help.ubuntu.com/community/SSH_VPN
- Hamachi http://logmein.com