

# WeOS Application Note 007: Setting up Site-to-Site SSL VPN/ OpenVPN in WeOS

This document describes how to setup a *site-to-site* OpenVPN with two nodes, an OpenVPN server (Alice) and an OpenVPN client (Bob) over the Internet.

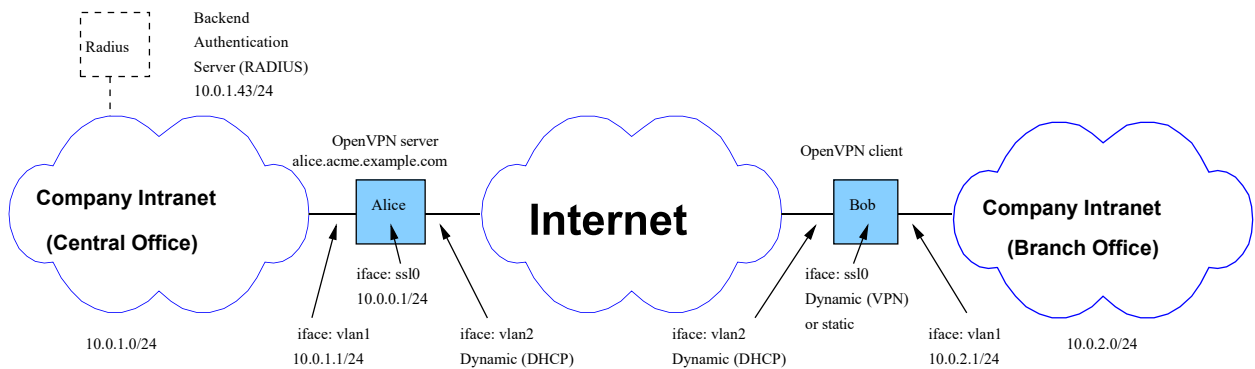


Figure 1 Site-to-site VPN example

## Introduction

The use case is to setup a site-to-site SSL VPN with two WeOS nodes, Alice (OpenVPN server) and Bob (OpenVPN client) over the Internet. To emulate the Internet, a single (auxiliary) WeOS unit is used, as shown in Figure 2. Still, the intention is that Alice and Bob could connect via regular ISPs as illustrated in Figure 1.

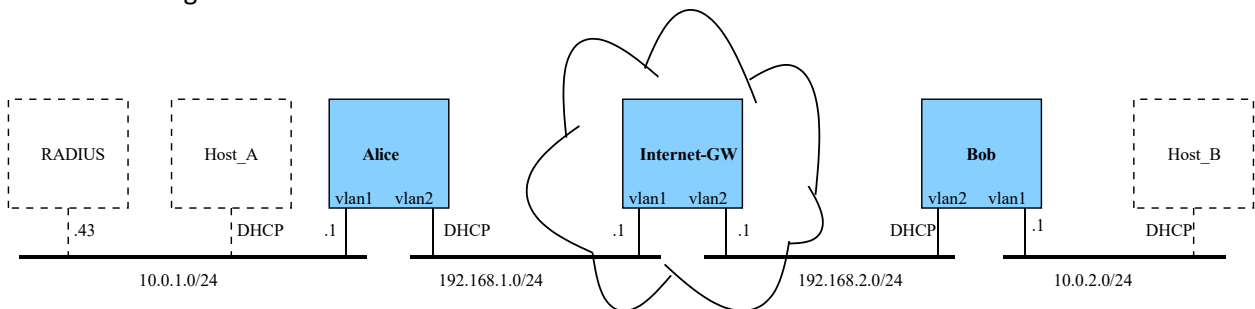


Figure 2 Application Note Network

- Alice:** Alice is an SSL VPN Server acting as VPN Gateway at a central office. Alice provides secure remote access to the central office network to Bob and other VPN clients/gateways. Additional information: Alice gets her upstream IP address dynamically (DHCP) from her ISP, and it is assumed that this address is publicly routable (if Alice is placed behind a NAT gateway, adequate port forwarding rules must be added to that NAT gateway). It is assumed that Alice uses DDNS to bind her acquired IP address to her domain name (here *alice.acme.example.com*).
- Bob:** Bob is an SSL VPN client acting as VPN Gateway at a branch office. Bob wishes to establish a VPN tunnel to the VPN Server Gateway (Alice). When the tunnel is up, units on

the branch office and central office networks can communicate securely with each other, as well as with the VPN Gateways (Alice and Bob).

Additional information: Bob gets his upstream IP address dynamically (DHCP) from his ISP. Here it is assumed that Bob's upstream IP address is "public/routable", although Bob could also reside behind a NAT Gateway.

In the simple case, Alice authenticates connecting SSL clients, such as Bob, by verifying his *certificate*; both Alice and Bob have certificates signed by a common certificate authority (CA). Similarly, Bob authenticates the server (Alice) by verifying her certificate.

As a complementary step, Alice can require Bob to provide additional username/password credentials, which she can verify in a local database or by invoking a backend server (typically RADIUS). This is shown later in this Application Note (see Appendix: Appendix B: Additional Authentication of Clients – Local database or RADIUS).

Alice and Bob will route traffic between their networks through the VPN tunnel, while other traffic is sent unencrypted towards the Internet as usual. To learn what networks reside at the central and branch office, two methods are provided:

- Dynamic routing: Let Alice and Bob run RIP or OSPF. This method handles topology changes automatically, thus is recommended when redundant gateways and tunnels are used between the branch and central office.
- Static routing: Alice and Bob can use static routing to reach each other's networks. Different options to achieve this will be shown depending if you wish to limit the configuration on the VPN client (Bob) or not.

For routing to work in a site-to-site use case, the VPN tunnel must be configured as a *layer-2* SSL tunnel (default is layer-3 tunnel).

When the tunnel is down, Alice and Bob risk to route traffic intended for the tunnel towards the Internet unencrypted. To ensure the traffic to peer network is dropped when the tunnel is down, Alice and Bob should configure static *blackhole routes* covering the address range of the company network.

## Configuring your WeOS unit via CLI

The configuration examples in this application note makes use of the WeOS CLI for almost all settings. If you prefer to use the Web interface for configuring your WeOS units, the information within this application note should still provide hints on what settings to configure via the Web interface.

For more information on how to access the CLI on a WeOS unit (console or remotely via SSH), please see chapter "General Maintenance" in the WeOS management guide, included in the WeOS release zip archive (<https://www.westermo.com>).

## Preparation Steps

### IP Plan

You need to prepare what IP address range to use for your sites as well as for the virtual subnet used by OpenVPN. In this example we allocate the private subnets from range 10.0.0.0/16:

- 10.0.0.0/24 is allocated for the OpenVPN virtual subnet. This is the subnet where the tunnel interface (ssl0) on Alice and Bob will have their IP address.
- 10.0.1.0/24 is the local network at Alice (Central Office)
- 10.0.2.0/24 is the local network at Bob (Branch Office)

If you wish to extend your site-to-site VPN with additional branch office networks, you could add VPN client gateways with subnets from the private range (10.0.3.0/24, 10.0.4.0/24, etc)

### Getting your WeOS units and software

In the Basic Setup you need two WeOS units (Alice and Bob) with Internet connection. The WeOS units must have SW Level “Extended”, e.g., RFI-2xx or Lynx-2xx series of products.

In the examples shown here we have an additional WeOS unit (Internet-GW in Figure 2) to emulate the Internet, but this is not needed if Alice and Bob have Internet connections. For more information on setting up this additional WeOS unit, see Appendix: Appendix A: (Optional) Preparation of auxiliary Internet Gateway.

The units used in this Application Note have WeOS-4.24.1 installed, but the configuration steps will likely work for other versions as well. It is recommended to install the latest WeOS “4.x.x” version available, see <https://www.westermo.com>.

### Generate Certificates and TLS Authentication Key

When using OpenVPN on WeOS units, authentication is based on certificates. To create certificates for Alice and Bob we recommend using the Easy-RSA scripts available as part of the OpenVPN distribution (<https://openvpn.net>). In addition, you should create a TLS authentication key; the same key is used both by Alice and Bob and the purpose is to mitigate any DDOS or port scanning attack against Alice. For this application note we will assume that you create the following certificate bundles and key:

- “alice.p12”: A PKCS#12 bundle holding VPN Server (Alice) certificate, private key, and her CA certificate.
- “bob.p12”: A PKCS#12 bundle holding VPN Client (Bob) certificate, private key, and his CA certificate (same CA certificate as used by Bob).
- “tlsauth.key”: An “OpenVPN” TLS Authentication key used by both Alice and Bob.

Appendix C: Creating Certificates for SSL VPNs using OpenVPN’s Easy-RSA tool contains hints on how you can create these files. (Furthermore, Westermo provides a separate Application Note on how to use the Easy-RSA scripts to create certificates and TLS authentication key. See “WeOS self-signed certificates” at <https://www.westermo.com>, “Support” => “Application Support” . )

### Sign up for a DDNS account

Dynamic DNS (DDNS): As your VPN gateways may use dynamic address assignment, use of a DDNS service comes in handy. WeOS supports several DDNS providers, see the “WeOS Management Guide” included in the WeOS Release Zip available from <https://www.westermo.com>.

DDNS is particularly useful to allow VPN clients to connect to the server (Alice) by use of her domain name (alice.acme.example.com).

In the network used in this application note (Figure 2), there is no DDNS server available. For the purpose of this making this application note self-contained, we let the Internet-GW act as DNS server with a static DNS entry for alice.acme.example.com. See also Appendix A: (Optional) Preparation of auxiliary Internet Gateway.

## Initial Configuration of your VPN Gateway Units

We start out by configuring initial configuration settings, such as hostname, VLANs, etc. on Alice and Bob. The example first shows the full example for the server (Alice), followed by a short explanation of what differs when configuring the corresponding settings on the client (Bob).

### Factory reset

For all WeOS units we start out with a factory default configuration. There are different ways to achieve a factory reset. From the CLI you can run “**copy factory-config running-config**” in Admin Exec context.

```
example:/#> cp factory-config running-config  
example:/#>
```

### Configuring System Name

First, we assign a hostname to Alice. It makes it simpler to see which unit we are managing.

```
example:/#> configure  
example:/config/#> system  
example:/config/system/#> hostname alice  
example:/config/system/#> leave  
alice:/#>
```

On Bob, the settings would be similar except that a different system hostname would be used:

```
example:/config/system/#> hostname bob
```

### Configuring Date and Time

Having a correct time on the VPN gateways is important. Without a correct system time, Alice and Bob may not be able to verify the certificates used for the OpenVPN tunnel.

It is recommended to use NTP client for time configuration to keep the time in sync. You could use an NTP server within your organisation or on the Internet, e.g., “**pool.ntp.org**”. Here we use the NTP server “**ntp.example.com**” run by our “Internet-GW” unit.

```
alice:/#> configure  
alice:/config/#> ntp  
alice:/config/ntp/#> server ntp.example.com  
alice:/config/ntp/server-ntp.example.com/#> leave  
alice:/#>
```

An alternative is to set the date manually, as shown below. The major drawback with this is that the system clock risks to be reset to “Unix 0 Time” (1970-01-01 00:00) if a unit is without power for a couple of days. Use of NTP would handle that automatically and is therefore recommended.

```
alice:/#> date 2018-12-06 12:04  
Thu Dec 6 12:04:00 UTC 2018  
alice:/#>
```

Date and time configuration can be done in the same manner on Bob as on Alice.

## Configuring VLANs

Let VLAN 2 be the *upstream* interface (WAN interface). Associate one port (here Ethernet port '1') to the upstream interface. That is, Ethernet port '1' becomes Alice WAN-port. Other ports will be associated with the default VLAN (VLAN 1), which will be used for the local (office) LAN.

```
alice:/#> configure
alice:/config/#> vlan 2
alice:/config/vlan-2/#> name wan-vlan
alice:/config/vlan-2/#> untagged eth 1
alice:/config/vlan-2/#> end
alice:/config/#> vlan 1
alice:/config/vlan-1/#> name office-vlan
alice:/config/vlan-1/#> leave
alice:/#>
```

VLAN configuration is done in the same manner on Bob as on Alice.

## Configuring IP settings

In this example, Alice uses DHCP on interface `vlan2` to get her WAN IP address and default gateway, from her ISP. (You may of course set the WAN address and default gateway manually, but that is not shown here.) We set the *admin distance* for our WAN to "1", thereby giving routes learnt via DHCP on `vlan2`, (e.g., default gateway) a high preference. Zeroconf addresses (169.254.x.x) are disabled, as there is no point using that on a WAN interface.

```
alice:/#> configure
alice:/config/#> iface vlan1
alice:/config/iface-vlan1/#> inet static
alice:/config/iface-vlan1/#> no address
alice:/config/iface-vlan1/#> address 10.0.1.1/24
alice:/config/iface-vlan1/#> distance 2
alice:/config/iface-vlan1/#> end
alice:/config/#> iface vlan2
alice:/config/iface-vlan2/#> inet dhcp
alice:/config/iface-vlan2/#> distance 1
alice:/config/iface-vlan2/#> no zeroconf
alice:/config/iface-vlan2/#> leave
alice:/#>
```

Bob would use a different IP address on his local interface (branch office network).

```
bob:/config/iface-vlan1/#> address 10.0.2.1/24
```

Connect your WAN port (here 'Eth 1' is associated with the WAN interface `vlan2`). Verify that you have received an address from your ISP by the "**show iface**" command. *If you intend to "emulate the Internet" as shown in Figure 2, this is a good time to setup your "Internet-GW" unit as described in Appendix: Appendix A: (Optional) Preparation of auxiliary Internet Gateway.*

```
alice:/#> show iface
Press Ctrl-C or Q(uit) to quit viewer, Space for next page, <CR> for next
line.

Interface Name      Oper   Address/Length      MTU    MAC/PtP Address
-----
lo                  UP     127.0.0.1/8         16436  N/A
vlan1              DOWN   10.0.1.1/24         1500   00:07:7c:00:34:61
vlan2              UP    192.168.1.10/24    1500   00:07:7c:00:34:63
-----

alice:/#>
```

### Configuring DDNS (hints)

In the test network used by this application note, there is no DDNS server, i.e. DDNS is not used. Still, below some *hints* are given on how you can configure DDNS on Alice. The exact configuration for your setup will differ as you may select a different *DDNS provider* (here “No-IP” is used, <https://www.noip.com/>) and you will certainly use another hostname for your VPN gateway. Use “**help provider**” in the DDNS context to see which providers are supported, select one and sign up for an account.

```
alice:/#> configure
alice:/config/#> ip
alice:/config/ip/#> ddns
alice:/config/ip/ddns/#> provider no-ip
alice:/config/ip/ddns/#> login MyAccountName MyAccountSecret
alice:/config/ip/ddns/#> hostname alice.acme.example.com
alice:/config/ip/ddns/#> show
Provider   : no-ip
SSL        : Disabled
Login      : MyAccountName
Password   : MyAccountSecret
Hostname   : alice.acme.example.com
Interval   : 600
alice:/config/ip/ddns/#> leave
alice:/#>
```

If you decide to run DDNS on Bob too (optional), he would use his own FQDN, e.g., `bob.acme.example.com`, when configuring his DDNS settings.

```
bob:/config/ip/ddns/#> hostname bob.acme.example.com
```

## OpenVPN Configuration

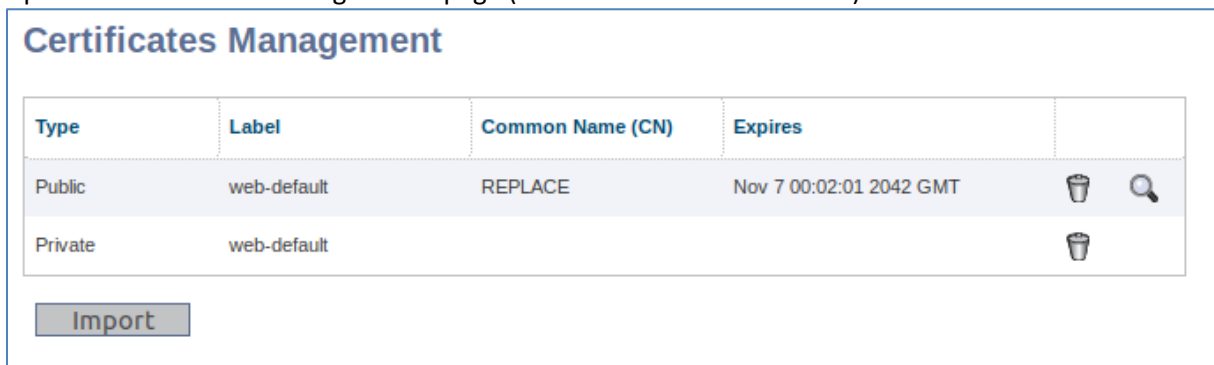
### Importing certificates and TLS Authentication key to the VPN Gateway

You can import certificates and TLS authentication key via Web or via the CLI.




#### Importing via Web interface

Simplest is to use Web-upload of your certificates TLS authentication key. Access Alice via HTTPS at her local address 10.0.1.1. The example below assumes you have created a PKCS12 certificate bundle for Alice named “alice.p12” (holding her certificate and private key, as well as the certificate of Alice’ and Bob’s common CA). (See Appendix C: Creating Certificates for SSL VPNs using OpenVPN’s Easy-RSA tool for hints on creating certificates.)

Open the “Certificate Management” page (Maintenance => Certificates)

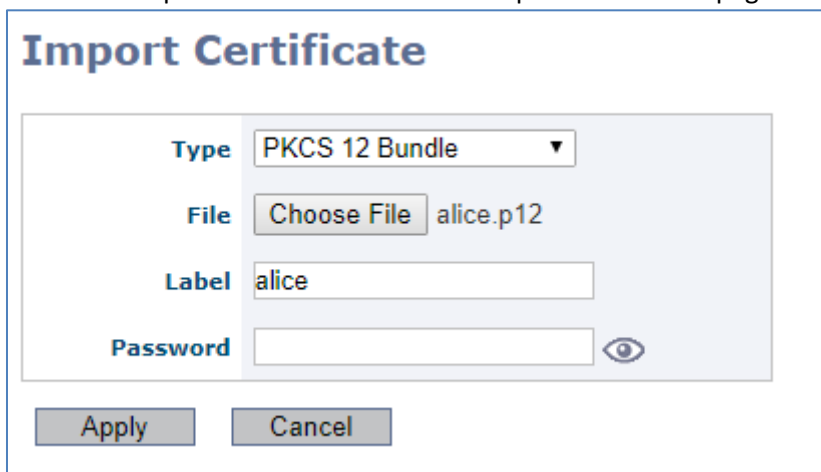


The screenshot shows the 'Certificates Management' page. It features a table with the following columns: Type, Label, Common Name (CN), Expires, and a set of icons (trash and search). The table contains two entries:

Type	Label	Common Name (CN)	Expires	
Public	web-default	REPLACE	Nov 7 00:02:01 2042 GMT	 
Private	web-default			

Below the table is an 'Import' button.

Press the “Import” button to reach the “Import Certificate” page.



The screenshot shows the 'Import Certificate' page. It contains a form with the following fields:









- Type: PKCS 12 Bundle (dropdown menu)
- File: Choose File button, followed by the text 'alice.p12'
- Label: Text input field containing 'alice'
- Password: Text input field with an eye icon for visibility toggle

At the bottom of the form are 'Apply' and 'Cancel' buttons.

Select proper “Type” and “File”, and enter a suitable “Label” if necessary.



### Certificates Management

Type	Label	Common Name (CN)	Expires	
Public	server	server	Oct 9 10:37:34 2028 GMT	 
Public	web-default	REPLACE	Nov 7 00:02:01 2042 GMT	 
CA	server	ACME-CA	Oct 9 10:36:02 2028 GMT	 
Private	server			
Private	web-default			

Import

Next, we import the TLS authentication key, here named "tlsauth.key". Press the Import button again.

### Import Certificate

Type

File  tlsauth.key





Label

Apply

Cancel

Select proper "Type" and "File", and enter a suitable "Label" if necessary.

### Certificates Management

Type	Label	Common Name (CN)	Expires	
Public	alice	alice	Nov 20 18:12:17 2028 GMT	 
Public	web-default	REPLACE	Nov 7 00:02:01 2042 GMT	 
CA	alice	ACME-CA	Nov 20 18:09:49 2028 GMT	 
Private	alice			
Private	web-default			
OpenVPN	tlsauth			

Import

To import certificates and TLS authentication key on Bob, you would connect to Bob via HTTPS on his local address 10.0.2.1.

- Certificates: The same steps would be made on the VPN client (Bob), but with a PKCS12 file “bob.p12” holding his certificate and private key, as well as the certificate of Alice’ and Bob’s CA).
- TLS Authentication key: The same key (“tlsauth.key”) should be imported to Bob as imported to Alice.

### Importing via CLI

An alternative is import the certificates and TLS Authentication key via the CLI. The example below shows how to use the CLI “cert” command to import them via SCP from a host at 10.0.1.5 with a user “foo”.

```
alice:/#> cert import pkcs scp://foo@10.0.1.5/home/foo/alice.p12
Downloading alice.p12 from scp://foo...
foo@10.0.1.5's password:
alice.p12                               100% 3064      3.0KB/s   00:00
Importing certificate alice...
OK
alice:/#> cert import ovpn scp://foo@10.0.1.5/home/foo/tlsauth.key
Downloading tlsauth.key from scp://foo...
foo@10.0.1.5's password:
tlsauth.key                             100%  657      0.6KB/s   00:00
Importing certificate tlsauth...
OK
alice:/#> show cert
Type      Label           Common Name           Expires
-----
Pub       alice           web-default           Nov 20 18:12:17 2028 GMT
Pub       web-default    REPLACE               Nov  7 00:02:01 2042 GMT
CA        alice           ACME-CA               Nov 20 18:09:49 2028 GMT
Key       alice
Key       web-default
OVPN Key  tlsauth
alice:/#>
```

## OpenVPN Tunnel settings

We configure Alice as an OpenVPN server (mode server is default).

- To specify Alice certificate and her CA certificate we use the associated label from the certificate repository (see previous section).
- We also specify the use of TLS authentication, providing the label for the associated key. As Alice is server we specify key direction “0”.
- We specify AES-128-CBC as crypto for the data transfer phase (as of WeOS 4.24.1 the somewhat weaker BF-CBC is default).
- We add a pool of addresses from which Alice can automatically assign addresses to Bob or other connecting VPN clients.
- We specify the tunnel interface (ssl0) to be a layer-2 interface. This is required in a site-to-site VPN setup; otherwise the routing will not work.
- (Optional) We specify “client-to-client” to allow communication between connecting clients and client networks. In this example there is only one client (Bob), but if you add additional clients (Charlie, Dave, ...) and wish to enable communication between them, then setting “client-to-client” is needed.
- (Optional) We specify “vlan2” (the WAN interface) as outbound interface for the tunnel. This step could typically be skipped, as the default “auto” setting would result in vlan2 as outbound interface anyway.
- Other settings we leave with the defaults. If you wish stronger security you may select “AES-256-CBC” for encryption, and “SHA256” for integrity protection.

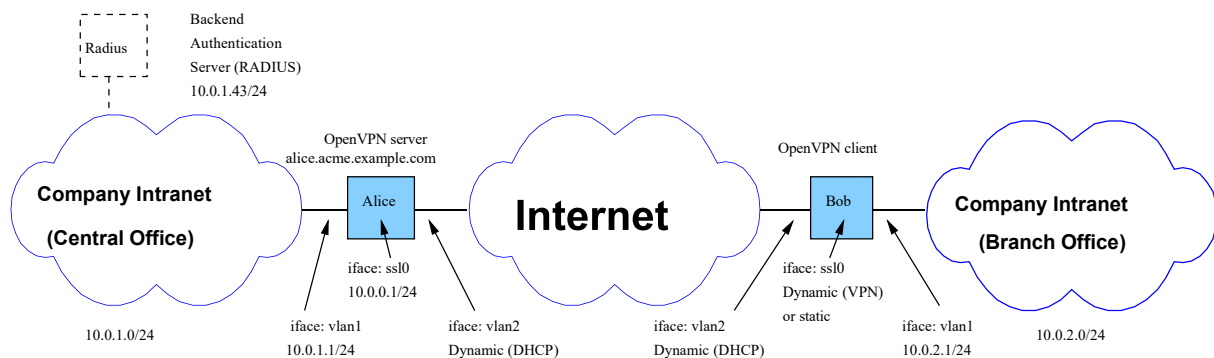


Figure 3 Configuration of OpenVPN tunnel settings at Server (Alice) and Client (Bob)

```
alice:/#> configure
alice:/config/#> tunnel
alice:/config/tunnel/#> ssl 0
alice:/config/tunnel/ssl-0/#> certificate alice
alice:/config/tunnel/ssl-0/#> ca-certificate alice
alice:/config/tunnel/ssl-0/#> tls-auth label tlsauth direction 0
alice:/config/tunnel/ssl-0/#> crypto aes-128-cbc
alice:/config/tunnel/ssl-0/#> outbound vlan2
alice:/config/tunnel/ssl-0/#> pool start 10.0.0.100 end 10.0.0.110 netmask
255.255.255.0
alice:/config/tunnel/ssl-0/#> type layer2
alice:/config/tunnel/ssl-0/#> client-to-client
alice:/config/tunnel/ssl-0/#> leave
alice:/#>
```

The tunnel configuration on Bob is shown below. The major differences are:

- Bob should be the client (the tunnel initiator), thus we set mode “no server”. Associated with this we define domain name of the server to connect to. In this sample setup this is “peer alice.acme.example.com”. **Note: In your setup you should modify the peer setting to match the domain name of your server.**
- We use label “bob” for Bob’s certificate and CA certificate settings.
- We set direction “1” for the tls-auth key.

```
bob:/#> configure
bob:/config/#> tunnel
bob:/config/tunnel/#> ssl 0
bob:/config/tunnel/ssl-0/#> no server
bob:/config/tunnel/ssl-0/#> certificate bob
bob:/config/tunnel/ssl-0/#> ca-certificate bob
bob:/config/tunnel/ssl-0/#> peer alice.acme.example.com
bob:/config/tunnel/ssl-0/#> tls-auth label tlsauth direction 1
bob:/config/tunnel/ssl-0/#> crypto aes-128-cbc
bob:/config/tunnel/ssl-0/#> outbound vlan2
bob:/config/tunnel/ssl-0/#> type layer2
bob:/config/tunnel/ssl-0/#> leave
bob:/#>
```

## IP Address of SSL interface

When creating an SSL tunnel in the previous section, Alice and Bob will get an interface (ssl0) connected to the virtual Open VPN subnet. The SSL server gateway (Alice) will almost always have a statically configured IP address on its SSL interface (ssl0). In this example we have also disabled management services except for SSH and HTTPS through the SSL Interface. SSH and HTTPS are optional to enable, i.e., you can skip the line “management https ssh”.

```
alice:/#> configure
alice:/config/#> iface ssl0
alice:/config/iface-ssl0/#> inet static
alice:/config/iface-ssl0/#> address 10.0.0.1/24
alice:/config/iface-ssl0/#> no management
alice:/config/iface-ssl0/#> management https ssh
alice:/config/iface-ssl0/#> leave
alice:/#>
```

The IP address of Bob’s SSL interface may be static or dynamic. If “dynamic” is used, the Bob will get his address assigned by Alice during tunnel establishment. We will see both methods (static and dynamic) for Bob’s SSL interface later. The example below shows how to set it to “dynamic” (this is the default for SSL interfaces), and we also limit the management services on this interface.

```
bob:/#> configure
bob:/config/#> iface ssl0
bob:/config/iface-ssl0/#> inet dynamic
bob:/config/iface-ssl0/#> no management
bob:/config/iface-ssl0/#> management https ssh
bob:/config/iface-ssl0/#> leave
bob:/#>
```

If you connect Alice and Bob to your ISPs, Bob should now be able to establish the VPN tunnel to Alice for test purposes. Remaining steps concerns settings for NAT, firewall, routing and hardening.

## Routing, NAT and Firewall settings

We need to configure routing rules to enable traffic to flow between Alice’s and Bob’s local networks through the VPN tunnel. In addition, Alice and Bob typically act as NAT Gateways and Firewalls towards the Internet.

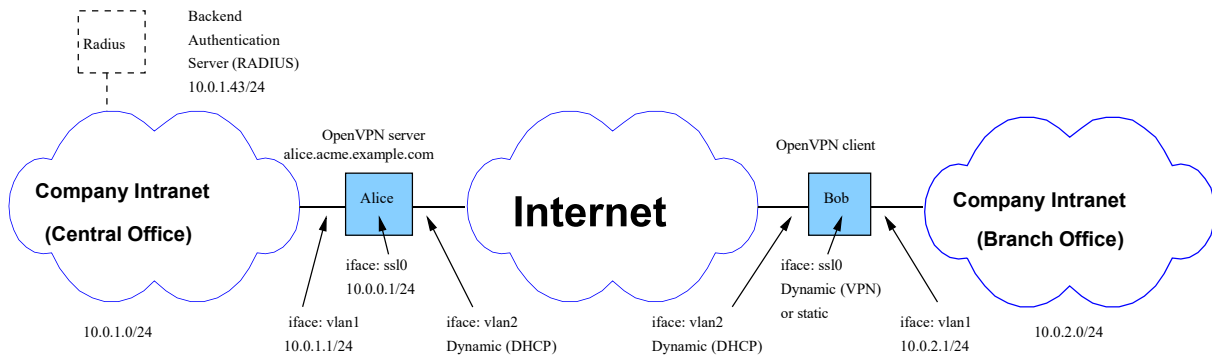


Figure 4 OpenVPN gateways (Alice and Bob) act as Firewall and NAT Gateways towards the Internet. Traffic between Central Office and Branch office shall be routed through the SSL tunnel.

## Configuring NAT and Firewall

Our VPN gateways (Alice and Bob) also act as NAT gateway (NATP/IP Masquerading) and Firewall towards the Internet.

- Enabling the firewall will block forwarding of traffic by default.
- We skip default filter rules when asked (“n” for No); it is ok to answer “y” (Yes), if you wish the gateway to respond to pings (ICMP Echo) on all existing interfaces.
- We configure the firewall to act as NATP (IP Masquerading) gateway with the WAN interface (vlan2) as outbound interface. The “addfilter” attribute allows traffic flows initiated “on the inside” to go out flow through the firewall.
- We add explicit firewall deny rules for port 53 (DNS) on the WAN interface (vlan2). The VPN gateway commonly act as a combined DHCP Server and DNS proxy to internal clients, although that is not shown in this application note. In that case the unit would by default act as DNS proxy on all interfaces, including the WAN interface (vlan2). Therefore, we should explicitly add the related deny rules for the WAN interface.
- We wish to allow traffic to flow between the local interface (here vlan1) and the tunnel interface (ssl0). Therefore, we add two related “filter allow” rules, one for each direction. In this example, all traffic between these interfaces is allowed. More fine grain allow filters could be used depending on your security policy.

```
alice:/#> configure
alice:/config/#> ip
alice:/config/ip/#> firewall
Activating firewall, type 'abort' to cancel.
Would you like a set of default filter rules for existing interfaces (y/N)? n
alice:/config/ip/firewall/#> nat type napt out vlan2 addfilter
alice:/config/ip/firewall/#> filter deny in vlan2 proto udp dport 53
alice:/config/ip/firewall/#> filter deny in vlan2 proto tcp dport 53
alice:/config/ip/firewall/#> filter allow in ssl0 out vlan1
alice:/config/ip/firewall/#> filter allow in vlan1 out ssl0
alice:/config/ip/firewall/#> leave
alice:/#>
```

At Bob, the same NAT and firewall filter rules can be used, as shown for Alice above.

## Routing

We have not yet added the routing configuration for the traffic to be routed through the VPN tunnel (this will be shown in three ways: with OSPF, with RIP, or with static routes). We should also ensure that private traffic is not routed unencrypted towards the Internet when the VPN tunnel is down.

### Avoiding that VPN traffic is routed outside the VPN tunnel

To avoid that traffic between your office networks are sent out unencrypted on the Internet when the VPN tunnel is down, we recommend that you add "blackhole" routes in your VPN gateways for your private network. In our example IP plan, we assume that traffic within IP range 10.0.0.0/16 is private and should be dropped when the tunnel is down, rather than being routed to the default gateway unencrypted.

```
alice:/#> configure
alice:/config/#> ip
alice:/config/ip/#> route 10.0.0.0/16 null0 200
alice:/config/ip/#> leave
alice:/#>
```

At Bob a blackhole route should be setup in the exact same way.

### Setting up routes for the VPN traffic: Alternative 1 with static routing

A simple way to setup routing use static routing. For static routing to work, both Alice and Bob must have static addresses on their respective ssl0 interface. Alice is the VPN Server gateway and has already been configured with a static address (10.0.0.1) on her ssl0 interface. Here we do the same for Bob, but with IP address 10.0.0.2.

```
bob:/#> configure
bob:/config/#> iface ssl0
bob:/config/iface-ssl0/#> inet static
bob:/config/iface-ssl0/#> address 10.0.0.2/24
```

Then we can add static routes both at Alice ...

```
alice:/#> configure
alice:/config/#> ip
alice:/config/ip/#> route 10.0.2.0/24 10.0.0.2
alice:/config/ip/#> leave
alice:/#>
```

... and on Bob:

```
bob:/#> configure
bob:/config/#> ip
bob:/config/ip/#> route 10.0.1.0/24 10.0.0.1
bob:/config/ip/#> leave
bob:/#>
```

## Setting up routes for the VPN traffic: Alternative 2 with dynamic routing (RIP or OSPF)

Another approach is to use dynamic routing protocols such as RIP or OSPF to exchange the routing information. This is relatively simple both on the server and client side, and it is a method that facilitates redundancy. That is, if you have added an extra VPN Gateway both at the central office (“Alice2”) and branch office (“Bob2”) to create a redundant VPN tunnel, then RIP (or OSPF) would automatically adapt the routing tables if your primary tunnels go down.

With RIP or OSPF, there is *no need* to ensure that Bob gets a fixed IP address. Instead Alice can assign an address from a pool. Thus, the configuration at Alice would look as follows (assuming RIP is used):

```
alice:/#> configure
alice:/config/#> tunnel
alice:/config/tunnel/#> ssl 0
alice:/config/tunnel/ssl-0/#> pool start 10.0.0.100 end 10.0.0.110
netmask 255.255.255.0
alice:/config/tunnel/ssl-0/#> end
alice:/config/tunnel/#> end
alice:/config/#> router
alice:/config/router/#> rip
alice:/config/router/rip/#> network vlan1
alice:/config/router/rip/#> network ssl0
alice:/config/router/rip/#> leave
alice:/#>
```

On Bob’s side, he should setup his ssl0 interface to get its IP address dynamically (from the pool) and enable RIP on the local interface and tunnel interface.

```
bob:/#> configure
bob:/config/#> iface ssl0
bob:/config/iface-ssl0/#> inet dynamic
bob:/config/iface-ssl0/#> end
bob:/config/#> router
bob:/config/router/#> rip
bob:/config/router/rip/#> network vlan1
bob:/config/router/rip/#> network ssl0
bob:/config/router/rip/#> leave
bob:/#>
```

If OSPF is preferred, the RIP configuration at Alice and Bob could be replaced by the following lines.

```
alice:/config/router/#> ospf
alice:/config/router/ospf/#> network 10.0.0.0/24 area 0
alice:/config/router/ospf/#> network 10.0.1.0/24 area 0
alice:/config/router/ospf/#> leave
alice:/#>
```

```
bob:/config/router/#> ospf
bob:/config/router/ospf/#> network 10.0.0.0/24 area 0
bob:/config/router/ospf/#> network 10.0.2.0/24 area 0
bob:/config/router/ospf/#> leave
bob:/#>
```

## Hardening

Here we assume that Alice and Bob will act as gateways towards the Internet. We have already configured Alice to act as firewall on her WAN interface (vlan2) in the previous section. We should also disable the management services available on the WAN interface, as enabled management services have implicit allow rules in the firewall.

```
alice:/#> configure
alice:/config/#> iface vlan2
alice:/config/iface-vlan2/#> no management
alice:/config/iface-vlan2/#> leave
alice:/#>
```

Alternatively, you could allow SSH and/or HTTPS management via the WAN interface. If you need to access your VPN gateway remotely, e.g., to perform management if the tunnel is down, enabling SSH or HTTPS via the WAN may be motivated, but this also depends on your local security policy.

```
alice:/#> configure
alice:/config/#> iface vlan2
alice:/config/iface-vlan2/#> no management
alice:/config/iface-vlan2/#> no ssh https
alice:/config/iface-vlan2/#> leave
alice:/#>
```

Similarly, management on the LAN interface (vlan1) should be limited or disabled, depending on your security policy. The following setting enables management using ssh, https and snmp on vlan1.

```
alice:/#> configure
alice:/config/#> iface vlan1
alice:/config/iface-vlan1/#> no management
alice:/config/iface-vlan1/#> no ssh https snmp
alice:/config/iface-vlan1/#> leave
alice:/#>
```

As always, you should ensure your admin account has a secure password. If you decide to enable any management service on the WAN interface, this becomes even more important!

```
alice:/#> configure
alice:/config/#> aaa
alice:/config/aaa/#> username admin MySecurePassword
alice:/config/aaa/#> leave
alice:/#>
```

There are additional steps you can take to hardening your unit. This guide is not intended to be exhaustive.

- Disabling unused management services is always a good habit. In the example below, we disable “ipconfig” service, but you could also consider disabling SNMP. They are already blocked on the WAN interface (“no management”), but here we disable the “ipconfig” service fully.
- We can also disable LLDP and IGMP signalling on the WAN port (here eth 1) and WAN VLAN (VLAN 2) respectively.



```
alice:/#> configure
alice:/config/#> no ipconfig
alice:/config/#> lldp
alice:/config/lldp/#> no port eth 1
alice:/config/lldp/#> end
alice:/config/#> vlan 2
alice:/config/vlan-2/#> no igmp
alice:/config/vlan-2/#> leave
alice:/#>
```

## Store the configuration

If you have not done that already, you should store the running configuration to startup-configuration.

```
alice:/#> cp running startup
alice:/#>
```

The same command should be run on Bob to store his configuration.

## Verifying Successful Tunnel Establishment

Once Bob has established the VPN tunnel to Alice they should be able to Verify VPN status (established). There are multiple methods.

- *Inspect VPN LED*: At the client (Bob), check the VPN LED turns green at Bob. This indicates the tunnel is successfully established.  
Note: The VPN LED at the SSL server (Alice) will turn green as soon as she is able to receive connections, i.e., irrespective if a connection is established or not. Thus, a green LED at the Alice is not a guarantee that a tunnel is up.
- *Inspect VPN Status in CLI*: Inspect the VPN status of the tunnel (at Alice and Bob) via CLI command for:
  - *SSL tunnel status*: Commands: In “Admin Exec” context, use “**show tunnel ssl**” (overall) or “**show tunnel ssl 0**” (details). In the detailed status information at Alice you should be able to see information about the connected client (Bob).
  - *SSL Interface status*: Command “**show iface ssl0**”, status should be *Up*.
- *Inspect VPN status in Web*:
  - SSL tunnel status information is found at path “Status => VPN & Tunnel => SSL”
  - SSL interface status information is found at path “Status => Interface”

## Troubleshooting

In case your VPN tunnels does not come up, the “messages log” file includes logging information relevant for SSL VPNs.

- In CLI: In “Admin Exec” context, use “**show log**” or “**follow log**” to find inspect the messages file.
- In Web: You find logs under path “Maintenance => View Log”, then select the “messages” file.

Common errors:

- Incorrect clock on unit. If the clock of Alice or Bob is not within the valid time for the used certificates, the tunnel will not come up or the service not even start. If a WeOS unit has been powerless for a long time; it will then start with “Unix 0 time” (1970-01-01). Configure the unit to use NTP client (recommended) or set the time manually.
- Use of layer-3 SSL tunnels in site-to-site use case. Configure the tunnel as “layer-2” instead.
- Bob not able to resolve IP address of the server (Alice). The problem could be that Bob’s DNS server is down or unreachable. If Alice is using Dynamic DNS (DDNS), the problem may relate to issues in the DDNS updates.
- Loss of connectivity between Alice and Bob. Bob will not be able to initiate (or re-initiate) the tunnel. Troubleshoot connectivity with “ping”, “traceroute” or other means. To mitigate the situation, you should use a redundant server at central office and branch office to setup a redundant tunnel. OSPF or RIP is recommended to handle routing dynamically.
- Inconsistent tunnel settings on VPN client and server, e.g., configuration of different encryption protocol at client and server side.

Example log error information. Here Bob is unable to resolve the address of the VPN server Alice (“peer alice.acme.example.com), i.e. this problem is related to DNS.

```
Nov 25 16:14:06 bob openvpn[2226]: RESOLVE: Cannot resolve host  
address: alice.acme.example.com: Name or service not known
```

## Appendixes

### Appendix A: (Optional) Preparation of auxiliary Internet Gateway

If you wish to test this application note as shown in Figure 2, you could use a separate WeOS unit to mimic the Internet-GW. SW level “extended” is needed it must be able to forward IP packets. The unit will act as DNS server to allow Bob to resolve “alice.acme.example.com” to the address Alice is assigned. It will also act as DHCP server and NTP server.

#### Factory reset

We start out with a factory default configuration. There are different ways to achieve a factory reset. From the CLI you can run “copy factory-config running-config” in Admin Exec context.

```
example:/#> cp factory-config running-config
example:/#>
```

#### Configure system name

We give this unit the hostname “internet-gw”.

```
example:/#> config
example:/config/#> system
example:/config/system/#> hostname internet-gw
example:/config/system/#> leave
internet-gw:/#>
```

#### Configure VLANs

The unit has one VLAN (VLAN 1). We create another VLAN (VLAN 2) with one port (here port “5”) where Bob can connect. Alice will connect to VLAN 1; any of the other ports will work.

```
internet-gw:/#> config
internet-gw:/config/#> vlan 2
internet-gw:/config/vlan-2/#> untagged 5
internet-gw:/config/vlan-2/#> leave
internet-gw:/#>
```

#### Configure IP settings

```
internet-gw:/#> config
internet-gw:/config/#> iface vlan1
internet-gw:/config/iface-vlan1/#> inet static
internet-gw:/config/iface-vlan1/#> no address
internet-gw:/config/iface-vlan1/#> address 192.168.1.1/24
internet-gw:/config/iface-vlan1/#> end
internet-gw:/config/#> iface vlan2
internet-gw:/config/iface-vlan2/#> address 192.168.2.1/24
internet-gw:/config/iface-vlan2/#> leave
internet-gw:/#>
```

## Configure DHCP Server

The “Internet-GW” unit will mimic Alice’ and Bob’s ISPs, and assign IP address via DHCP. To ensure Alice gets a specific address (here 192.168.1.10), the pool on subnet 192.168.1.0 is limited to only contain that address.

```
internet-gw:/#> config
internet-gw:/config/#> dhcp-server
internet-gw:/config/dhcp-server/#> subnet 192.168.1.0/24
internet-gw:/config/dhcp-server/subnet-192.168.1.0/#> pool 192.168.1.10 1
internet-gw:/config/dhcp-server/subnet-192.168.1.0/#> end
internet-gw:/config/dhcp-server/#> subnet 192.168.2.0
internet-gw:/config/dhcp-server/subnet-192.168.2.0/#> leave
internet-gw:/#>
```

## Configure static DNS entries

When acting as DHCP server, the unit will also act as DNS (proxy) server. By adding static DNS entries (host entries), it can also respond to additional DNS requests for “alice.acme.example.com” and “ntp.example.com”.

```
internet-gw:/#> config
internet-gw:/config/#> ip
internet-gw:/config/ip/#> host alice.acme.example.com 192.168.1.10
internet-gw:/config/ip/#> host ntp.example.com 192.168.1.1
internet-gw:/config/ip/#> leave
internet-gw:/#>
```

## Configure unit as NTP server

The WeOS unit can act as NTP server (tech preview as of WeOS 4.24.1). In this case it will be a standalone NTP server, using the local clock as reference (not recommended except for test purposes like this). As part of this, we set the current time by use of the “date” command.

```
internet-gw:/#> date
Sun Jan 18 03:03:38 UTC 1970
internet-gw:/#> date 2018-12-06 12:04
Thu Dec 6 12:04:00 UTC 2018
internet-gw:/#> configure
internet-gw:/config/#> ntp
internet-gw:/config/ntp/#> listen
internet-gw:/config/ntp/#> leave
internet-gw:/#>
```

## Store the configuration

As we are done with the configuration we store the running configuration to startup-configuration.

```
internet-gw:/#> cp running startup
internet-gw:/#>
```

## Appendix B: Additional Authentication of Clients – Local database or RADIUS

As part of the tunnel establishment, the server (Alice) and client (Bob) will authenticate each other using the certificates issued by their common Certificate Authority. OpenVPN also enables the server to conduct an additional authentication of the client, where Alice requires Bob to present username/password credentials after the certificates are validated.

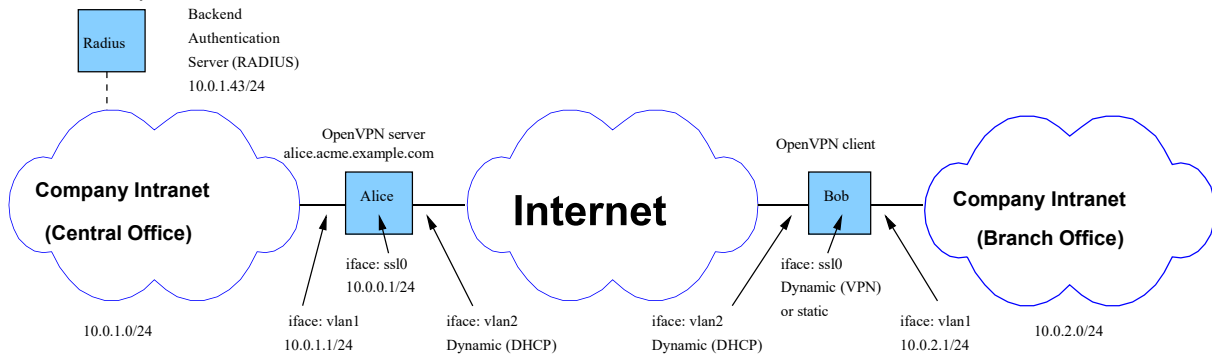


Figure 5 Additional authentication of client (Bob) by local database list (Alice) or backend authentication server (RADIUS).

We start by looking at the required configuration on the client Bob, i.e., how Bob should fill out his credentials. After that we look at two different methods for Alice to verify Bob's credentials; either by storing credentials in a local database at Alice, or by forwarding the credential handshake messages to a backend authentication server. In that example we show a rudimentary setup with a backend RADIUS server, but TACACS+ would also be an option.

### OpenVPN client (Bob): Configuration of username/password credentials

The username/password credentials are configured via the "identity" command, available for VPN clients within the SSL configuration context.

```
bob:/#> configure
bob:/config/#> tunnel ssl 0
bob:/config/tunnel/ssl-0/#> identity bob password builder
bob:/config/tunnel/ssl-0/#> leave
bob:/#>
```

### OpenVPN server (Alice): Verifying credentials using a local database

The server can configure a local database holding VPN client credentials. In this case we only have one client (Bob), but you should repeat the "username" command for every connecting VPN client.

```
alice:/#> configure
alice:/config/#> aaa
alice:/config/aaa/#> local-db 1
alice:/config/aaa/local-db-1/#> description openvpn-users
alice:/config/aaa/local-db-1/#> username bob builder
alice:/config/aaa/local-db-1/#> leave
alice:/#>
```

Once the local database is created you can instruct the VPN Server to use it as an additional authentication step for connecting VPN clients. This is done by the “aaa-method” setting available for VPN servers in the SSL VPN configuration context.

```
alice:/#> configure
alice:/config/#> tunnel ssl 0
Alice:/config/tunnel/ssl-0/#> aaa-method local-db 1
Alice:/config/tunnel/ssl-0/#> leave
alice:/#>
```

### OpenVPN server (Alice): Verifying credentials using a backend RADIUS server

In the test setup in Figure 5 there is a RADIUS server located at address 10.0.1.43 in Alice’ local network. We will not provide a description of how to setup a RADIUS server, but give some rudimentary hints on how to get a Linux Server with FreeRADIUS (<https://freeradius.org>) configured for this specific test. Two files (“clients.conf” and “users”) are affected:

In /etc/freeradius/clients.conf (security between server Alice and the RADIUS server):

```
client 10.0.1.1 {
    shortname = 10.0.1.1
    secret    = str4wb3rry
    nastype   = other
}
```

In /etc/freeradius/users (credentials of OpenVPN clients):

```
bob          Cleartext-password := "builder"
```

On the VPN server Alice, you first need to configure settings related to the RADIUS server (the default type for remote-server is RADIUS).

```
alice:/#> configure
alice:/config/#> aaa
alice:/config/aaa/#> remote-server 1
alice:/config/aaa/remote-server-1/#> address 10.0.1.43
alice:/config/aaa/remote-server-1/#> password str4wb3rry
alice:/config/aaa/remote-server-1/#> leave
alice:/#>
```

Once the RADIUS server settings are configured you can instruct the VPN Server to use it as an additional authentication step for connecting VPN clients. This is done by the “aaa-method” setting available for VPN servers in the SSL VPN configuration context.

```
alice:/#> configure
alice:/config/#> tunnel ssl 0
alice:/config/tunnel/ssl-0/#> aaa-method remote-server 1
alice:/config/tunnel/ssl-0/#> leave
alice:/#>
```

## Appendix C: Creating Certificates for SSL VPNs using OpenVPN’s Easy-RSA tool

Objective:

- Create certificates and keys to be used for OpenVPN/SSL VPN
- VPN Server certificate (Alice) should include Server Extension (while) VPN client certificates (Bob, Charlie and Dave) should not.
- Create PKCS bundles (Certificate, Key, CA Certificate) for easy import into WeOS units
- Create TLS Authentication key

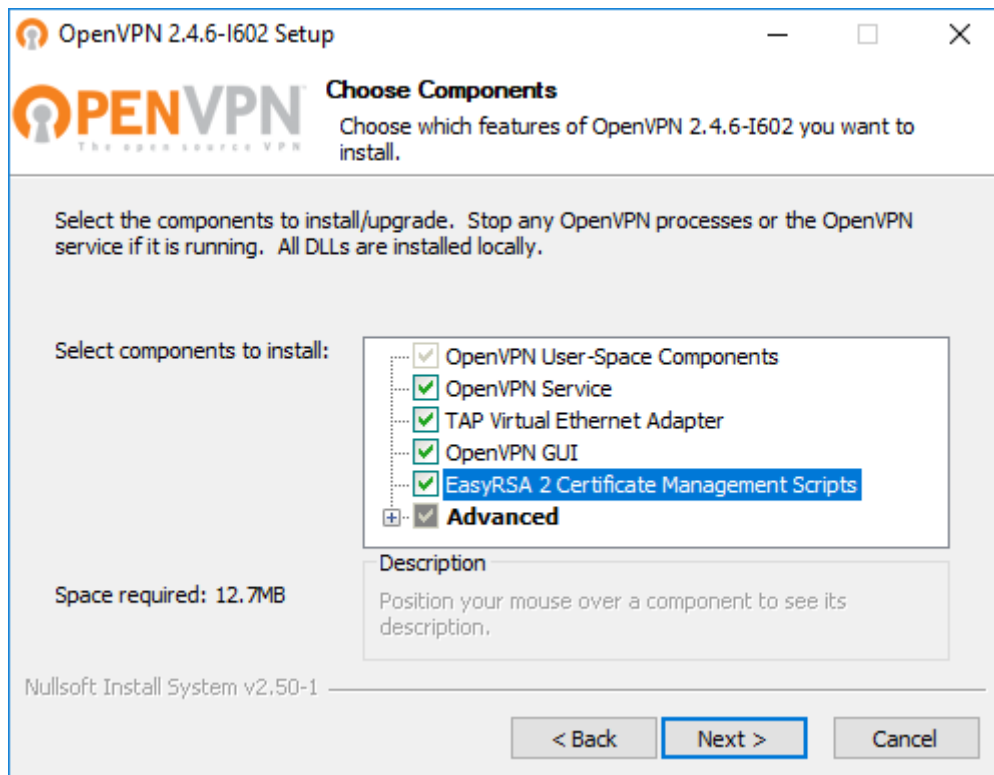
To achieve this, we will create a rudimentary Certificate Authority (CA) based on the OpenVPN Easy-RSA tool. Here we will use a Windows 10 PC, but OpenVPN/Easy-RSA is available on other platforms too, including MAC and Linux.

The main source of information for OpenVPN and Easy-RSA is [www.openvpn.net](http://www.openvpn.net), and in particular their community documentation. You are strongly recommended to visit that documentation for more complete information on maintaining your own CA. The guide in this appendix is provided “as is” with the intention to assist you in creating the needed certificates and keys.

### Installing OpenVPN and Easy-RSA

The Easy-RSA tool for Windows is distributed as part of the OpenVPN software bundle. Download it from [www.openvpn.net](http://www.openvpn.net). In the “Community” tab and “Downloads” you find the “Windows Installer”.

When running the windows installer, make sure to add the Easy-RSA tool.



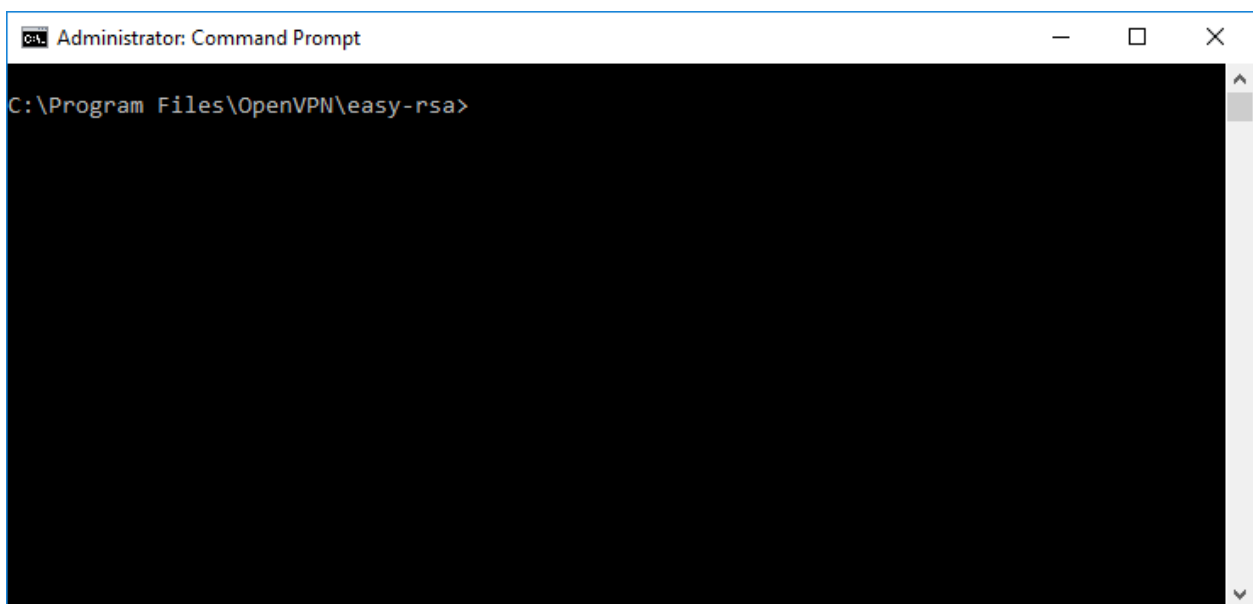
## Setting up CA

The example in this section is based on the guideline *Setting up your own Certificate Authority (CA)*<sup>1</sup> found at [www.openvpn.net](http://www.openvpn.net) (tab “Community” => “Documentation”)

The aim is to create PKCS (.p12) certificate bundles for the VPN server (Alice) and clients (Bob, Charlie and Dave). We will first create a CA, which can then be used to sign the user certificates for Alice, Bob, Charlie and Dave.

### Preparation

The Easy-RSA scripts are found in a subdirectory of the OpenVPN installation. Open a Windows Command Prompt, and change directory to “C:\Program Files\OpenVPN\easy-rsa”. **Note:** You may need to open the Command prompt as “Administrator” in order to run the scripts.



Run the “init-config” script:

```
C:\Program Files\OpenVPN\easy-rsa>init-config
```

Now edit the vars file (called vars.bat on Windows) and set the KEY\_COUNTRY, KEY\_PROVINCE, KEY\_CITY, KEY\_ORG, and KEY\_EMAIL parameters. Don’t leave any of these parameters blank. You can use notepad to edit the file.

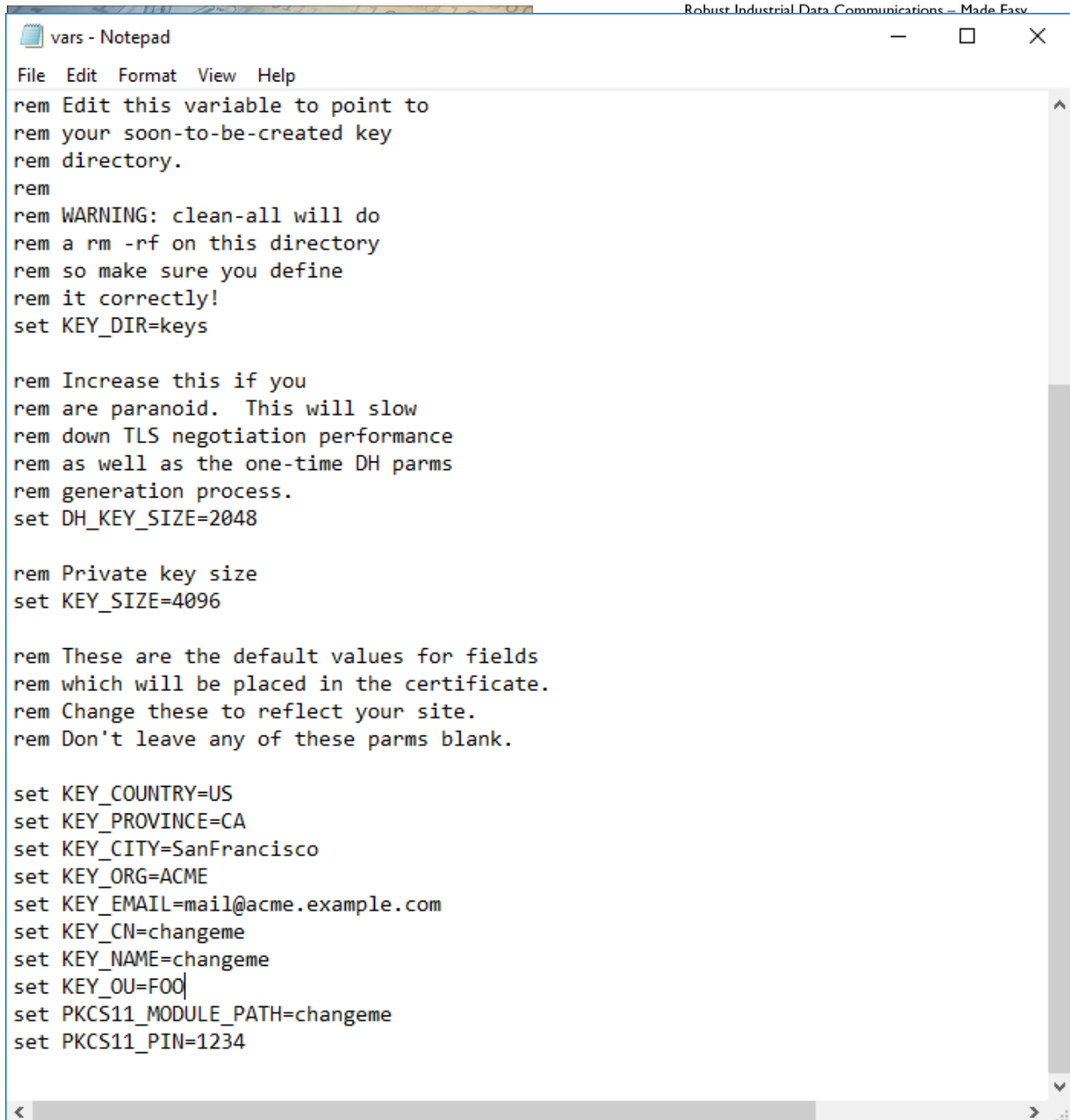
```
C:\Program Files\OpenVPN\easy-rsa>notepad vars.bat
```

In this example most of the default values were used, only KEY\_ORG, KEY\_EMAIL, and KEY\_OU got new values (ACME, [mail@acme.example.com](mailto:mail@acme.example.com) and FOO).

---

<sup>1</sup> <https://openvpn.net/community-resources/setting-up-your-own-certificate-authority-ca/> Accessed November 2018





```
vars - Notepad
File Edit Format View Help
rem Edit this variable to point to
rem your soon-to-be-created key
rem directory.
rem
rem WARNING: clean-all will do
rem a rm -rf on this directory
rem so make sure you define
rem it correctly!
set KEY_DIR=keys

rem Increase this if you
rem are paranoid. This will slow
rem down TLS negotiation performance
rem as well as the one-time DH parms
rem generation process.
set DH_KEY_SIZE=2048

rem Private key size
set KEY_SIZE=4096

rem These are the default values for fields
rem which will be placed in the certificate.
rem Change these to reflect your site.
rem Don't leave any of these parms blank.

set KEY_COUNTRY=US
set KEY_PROVINCE=CA
set KEY_CITY=SanFrancisco
set KEY_ORG=ACME
set KEY_EMAIL=mail@acme.example.com
set KEY_CN=changeme
set KEY_NAME=changeme
set KEY_OU=FOO
set PKCS11_MODULE_PATH=changeme
set PKCS11_PIN=1234
```

Then run the “vars” and “clean-all” scripts.

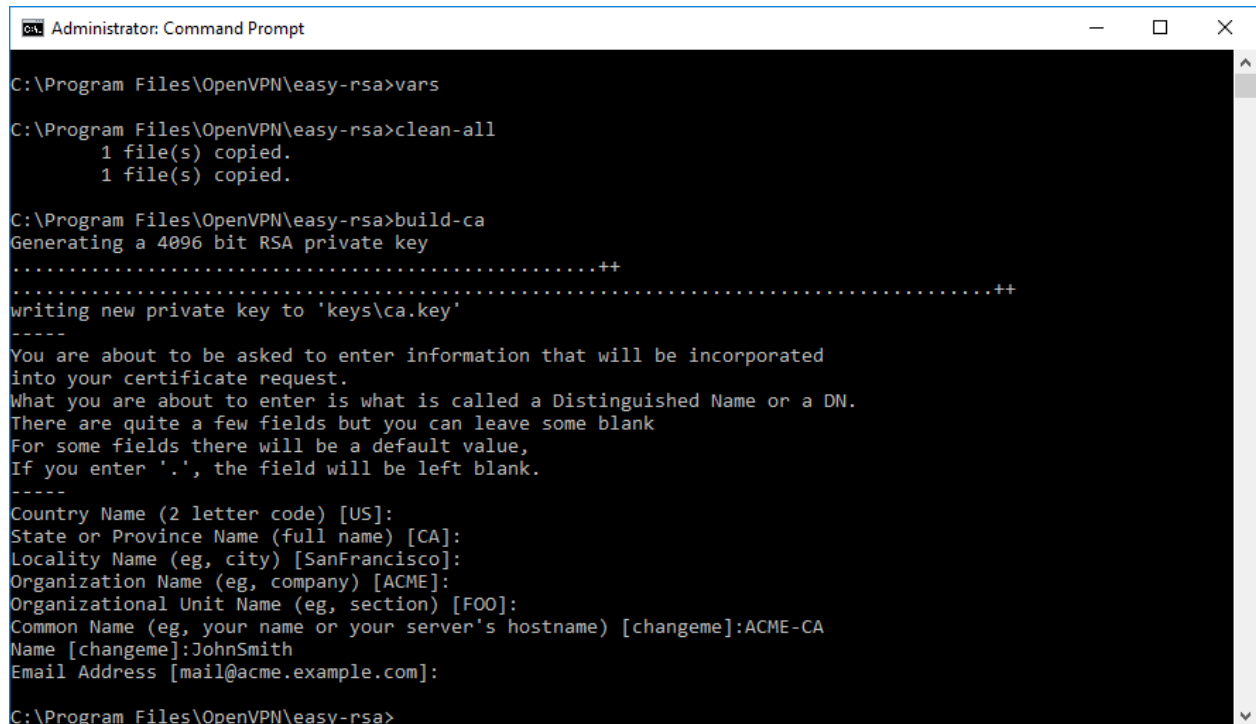
```
C:\Program Files\OpenVPN\easy-rsa>vars
C:\Program Files\OpenVPN\easy-rsa>clean-all
The system cannot find the file specified.
    1 file(s) copied.
    1 file(s) copied.
```

### Create CA and generate CA certificate

To generate the CA certificate, run the “build-ca” script

```
C:\Program Files\OpenVPN\easy-rsa> build-ca
```

Most default values from the “vars” file were used. The “Common Name” and “Name” settings were modified.



```
Administrator: Command Prompt
C:\Program Files\OpenVPN\easy-rsa>vars
C:\Program Files\OpenVPN\easy-rsa>clean-all
    1 file(s) copied.
    1 file(s) copied.
C:\Program Files\OpenVPN\easy-rsa>build-ca
Generating a 4096 bit RSA private key
.....++
.....++
writing new private key to 'keys/ca.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [US]:
State or Province Name (full name) [CA]:
Locality Name (eg, city) [SanFrancisco]:
Organization Name (eg, company) [ACME]:
Organizational Unit Name (eg, section) [FOO]:
Common Name (eg, your name or your server's hostname) [changeme]:ACME-CA
Name [changeme]:JohnSmith
Email Address [mail@acme.example.com]:
C:\Program Files\OpenVPN\easy-rsa>
```

## Generating Server and Client certificates

### Generating the Server Certificate

Use the “build-key-server” script to create the certificate for the VPN server (Alice). This ensures the certificate includes “TLS Web Server Authentication” attribute.

```
C:\Program Files\OpenVPN\easy-rsa> build-key-server alice
```

Most parameters can use defaults. When *Common Name* is queried, a unique name is required (here “alice” was used). Two other queries require positive responses (y), “Sign the certificate? [y/n]” and “1 out of 1 certificate requests certified, commit? [y/n]”.

### Generating the Client Certificate(s)

For Bob (and the other clients) we can use the “build-key-pkcs12” script

```
C:\Program Files\OpenVPN\easy-rsa>build-key-pkcs12 bob
```

This creates a certificate (bob.crt) and private key (bob.key), as well as a PKCS12 bundle. The bundle contains Bob’s certificate and private key, and also the certificate of his CA (ca.crt).

The same script can then be run for additional clients (Charlie and Dave).

### Creating a PKCS (.p12) bundle for the server Alice (optional)

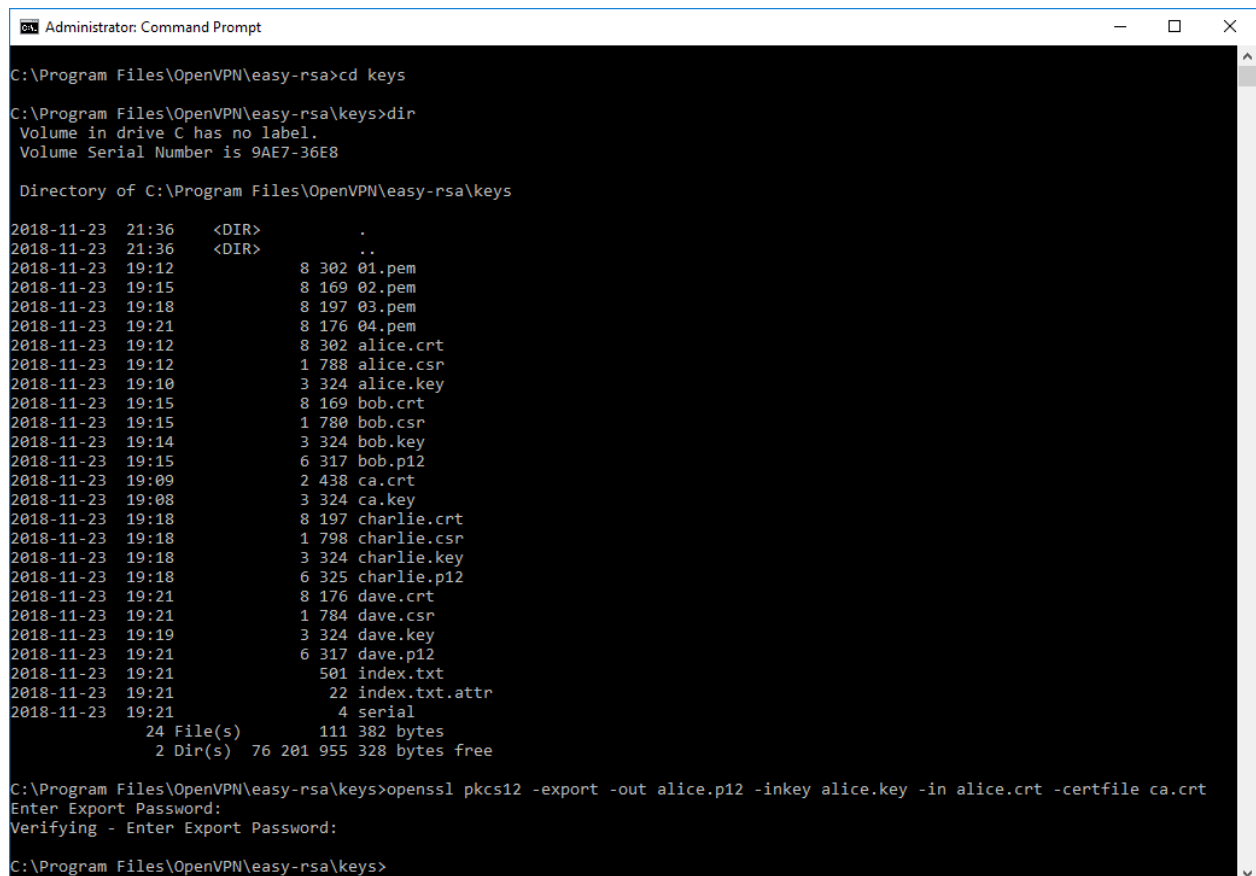
Alice certificate (alice.crt) and private key (alice.key) reside in the “keys” subdirectory. To simplify the import to the WeOS unit (Alice), we would like to create a PKCS12 bundle for Alice (as already done for Bob, Charlie and Dave).

1. Change directory to the “keys” directory. There you can find the files necessary to create the PKCS12 bundle for the server.

```
C:\Program Files\OpenVPN\easy-rsa\keys>cd keys
```

2. Then run the following “openssl” command to create the PKCS12 bundle.

```
C:\Program Files\OpenVPN\easy-rsa\keys>openssl pkcs12 -export -out alice.p12 -inkey  
alice.key -in alice.crt -certfiles ca.crt
```



```
Administrator: Command Prompt
C:\Program Files\OpenVPN\easy-rsa>cd keys
C:\Program Files\OpenVPN\easy-rsa\keys>dir
Volume in drive C has no label.
Volume Serial Number is 9AE7-36E8

Directory of C:\Program Files\OpenVPN\easy-rsa\keys

2018-11-23  21:36    <DIR>          .
2018-11-23  21:36    <DIR>          ..
2018-11-23  19:12             8 302 01.pem
2018-11-23  19:15             8 169 02.pem
2018-11-23  19:18             8 197 03.pem
2018-11-23  19:21             8 176 04.pem
2018-11-23  19:12             8 302 alice.crt
2018-11-23  19:12             1 788 alice.csr
2018-11-23  19:10             3 324 alice.key
2018-11-23  19:15             8 169 bob.crt
2018-11-23  19:15             1 780 bob.csr
2018-11-23  19:14             3 324 bob.key
2018-11-23  19:15             6 317 bob.p12
2018-11-23  19:09             2 438 ca.crt
2018-11-23  19:08             3 324 ca.key
2018-11-23  19:18             8 197 charlie.crt
2018-11-23  19:18             1 798 charlie.csr
2018-11-23  19:18             3 324 charlie.key
2018-11-23  19:18             6 325 charlie.p12
2018-11-23  19:21             8 176 dave.crt
2018-11-23  19:21             1 784 dave.csr
2018-11-23  19:19             3 324 dave.key
2018-11-23  19:21             6 317 dave.p12
2018-11-23  19:21             501 index.txt
2018-11-23  19:21             22 index.txt.attr
2018-11-23  19:21             4 serial
                24 File(s)          111 382 bytes
                2 Dir(s)   76 201 955 328 bytes free

C:\Program Files\OpenVPN\easy-rsa\keys>openssl pkcs12 -export -out alice.p12 -inkey alice.key -in alice.crt -certfile ca.crt
Enter Export Password:
Verifying - Enter Export Password:

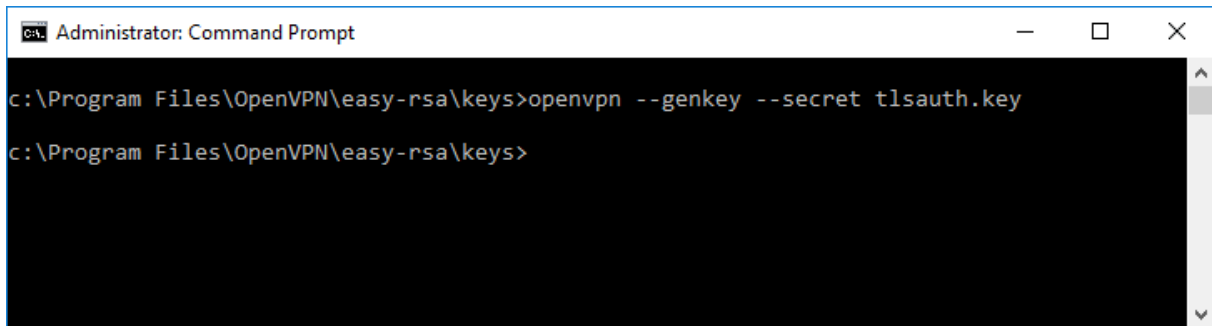
C:\Program Files\OpenVPN\easy-rsa\keys>
```

### Creating TLS authentication key

To create the TLS authentication key, the “openvpn” command can be used.

```
openvpn --genkey --secret tlsauth.key
```

The key file (here “tlsauth.key”) is created in the current directory. In the example below this is run in the “keys” directory.



```
Administrator: Command Prompt
c:\Program Files\OpenVPN\easy-rsa\keys>openvpn --genkey --secret tlsauth.key
c:\Program Files\OpenVPN\easy-rsa\keys>
```

If the openvpn executable is not found, you probably need to update the search path by running the following below. (This is usually done by the “vars” script used when setting up the CA.)

```
set "PATH=%PATH%;C:\Program Files\OpenVPN\bin"
```

## Revision history for version 1.0

Revision	Rev by	Revision Note	Date
00	JOV	First version	2019-03-18
01			
02			