Deploying 1E Client on macOS

Summary

Guidance for deploying 1E Client onto **macOS** devices, including installation and uninstallation. Only the Tachyon features of 1E Client are available on macOS.

On this page:

Requirements

Please review Design Considerations and Requirements pages. After installation please review the Verifying page.

For details of supported OS platforms please refer to Supported Platforms reference page.

Guidance provided below is for installation on macOS.

Please contact 1E if guidance is required for installation on other non-Windows OS and for Android.

1E Client does not have a license key. Even so, you must adhere to the terms of your license agreement.

Deployment choices

You must decide how you will configure the 1E Client and deploy to devices. For more information about configuring the 1E Client properties during and after installation, please refer to 1E Client configuration settings and installer properties.

Deploying the 1E Client is normally achieved using your existing software deployment tool.

Non-Windows installation account

To install the 1E Client on a non-Windows client the installation account must have privileges to run the **sudo** command.

Certificate files

Each Tachyon client device requires a **.pfx** certificate file. Please refer to Client certificates below, for steps to create and use the **.pfx** file.

The 1E Client for macOS offers two alternative methods for storage of the Tachyon client certificate and the public certificates for the chain of trust for the Tachyon Switch:

- SSL certificate file method
- Key Store method

If you have configured Tachyon Server to require client certificates, then each device requires a certificate with the following properties so the Tachyon client can be authenticated by the Tachyon Switch.

Client certificates must meet the following requirements:

- Issued by a trusted Certificate Authority (CA)
  - The certificate for the Root CA in the Certification Path must exist in the Trusted Root CA store of the client
  - If the issuing CA is not the Root CA then the certificate for the issuing CA and any intermediate CA in the Certification Path must exist in the Intermediate CA store of the client
  - If either of these CA certificates are different to those used by the Tachyon Web Server, they will need to be exported and imported on the Tachyon Web Server
  - Most organizations have automated distribution of these CA certificates to clients and servers, using Group Policy for example.
- Has at least the following Enhanced Key Usage
  - Client Authentication
- Has at least the following Key Usage
  - Digital Signature
  - Key encipherment
- Revocation information is included.
  - References at least one CRL Distribution point that uses HTTP.
- Has a Subject Name of type Common Name (**CN=<hostname>**) or Subject Alternative Name (**DNS Name=<hostname>**) where <hostname> depends on the type of device:

    - On domain-joined Windows PCs this must be the **hostname FQDN** of the computer, for example **W701.ACME.LOCAL**
    - On workgroup Windows PCs and non-Windows devices, this must be the hostname of the computer - as returned by the **hostname** command, for example on Windows PC this could be **W701**, and on a Mac this could be **MAC01.local**
- Has a private key
    - For workgroup Windows and non-Windows devices, this must be exportable

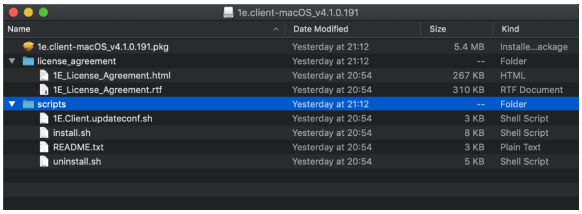
Preparation

The Windows and non-Windows versions of the 1E Client are available for download from the 1E Support Portal.

Installation source files for 1E Client for non-Windows are available in a zip file called **1EClient-Non-Windows.v5.1.x.x.zip**

Within the zip, the macOS 1E Client is provided as an Apple Disk Image file, such as **1e.client-macOS_v5.1.x.x.dmg**.

Once the **dmg** file is copied to macOS, double-click on the **dmg** file to automount it under **/Volumes** and display it in a Finder window, as shown opposite.



If you wish, you can script this using all or parts of the following example:

```
mkdir ~/Downloads/1e.client
cd ~/Downloads/1e.client
wget https://<<serverURL>>/1e.client-macOS_v5.1.x.x.dmg
hdiutil attach 1e.client-macOS_v5.1.x.x.dmg
```

The last line of the output of the **hdutil** command will show the location of the mount point, which is likely to be **/Volumes/image.1e.client-macOS_v5.1.x.x**

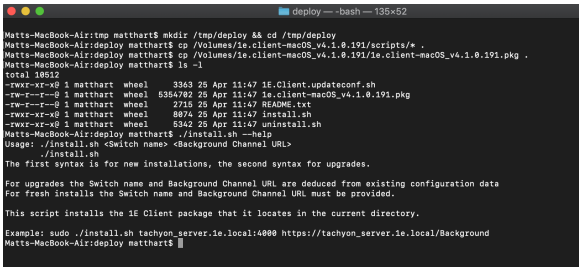
Installation

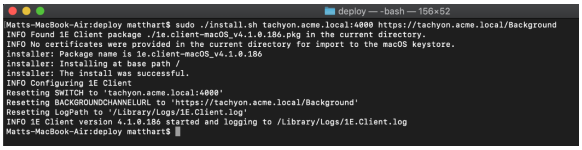
To install the macOS 1E Client perform **only one** of the following three options :

1. Use the **install.sh** script provided under the scripts directory, to install and configure the client, as shown in the pictures opposite:
    - Copy the script to a separate directory together with the package you wish to use for installation or upgrade
    - Invoke **install.sh** script to install, configure and start the 1E Client, giving the Switch host and port as the first parameter and the Background Channel URL as the second:

    ```
    sudo ./install.sh tachyon.acme.local:4000 https://tachyon.acme.local/Background
    ```

    > ⚠ This method will start the client, but connection to the Tachyon Switch will fail unless the client has the necessary certificates, as described in Client certificates below.





2. Install the package file from a command window to install using defaults. The **-target** is the volume mount point, not the path in the file system at which it will be installed.

    ```
    sudo installer -pkg /path/to/pkg/1e.client-macOS_v5.1.x.x.pkg -target /
    ```

3. Double-click on the package (.pkg) file within the Finder window to install using defaults.

For options 2 and 3, you will need to reconfigure the 1E Client either using the **1E.Client.updateconf.sh** script or by editing the **/etc/1E/Client/1E.Client.conf** file, as described in Reconfiguring 1E Client on macOS below.

You will also need to provide the necessary client certificates, as described in Client certificates below, and then start the client.

After installation, you can unmount the **dmg** using the following command:

```
hdiutil eject /Volumes/image.1e.client-macOS_v5.1.
x.x
```

Reconfiguration

You will need to use the **1E.Client.updateconf.sh** script if you want to reconfigure the client, or you installed 1E Client using installation options 2 or 3 above.

The **pkg** installation package for macOS includes a bash script called **1E.Client.updateconf.sh**. The configuration properties for the Switch and Background Channel are mandatory, therefore assuming they are on the same Tachyon Server which has a DNS Name FQDN **tachyon.acme.local** t hen the post-installation command-line would be similar to the following but all on one command line. The second Switch and Background Channel can be removed if a DMZ server is not used. Single-quotes avoids escape characters and are necessary to allow use of the ; character.

```
sudo sh '/Library/Application Support/1E/Client/1E.Client.updateconf.sh'
'/Library/Application Support/1E/Client/1E.Client.conf'
SWITCH='tachyon.acme.local:4000;tachyon.acme.com:4000'
BACKGROUNDCHANNELURL='https://tachyon.acme.local:443/Background/;https://tachyon.acme.com:443/Background/'
```

Please refer to 1E Client configuration settings and installer properties for a list of other configuration properties that can be appended to the above command-line.

Client certificates

The 1E Client for macOS offers two alternative methods for storage of the Tachyon client certificate and the public certificates for the chain of trust for the Tachyon Switch:

- SSL certificate file method
- Key Store method

First, you need to issue a certificate and create the **.pfx** file.

Using OpenSSL to create the .pfx file

Each non-Windows devices requires its own certificate. Below is a guide for using a Microsoft CA to issue a certificate (which is the same for Windows computers), then exporting it and using OpenSSL to prepare it before installing it on the non-Windows device.

First, you will need to have created a new Certificate template on your Certificate Authority by making a duplicate of either the Computer or Workstation template and configuring it with at least the following properties:

- **General** - use a suitable name such as **Tachyon Devices** and validity period
- **Request Handling** - Allow private keys to be exported
- **Subject Name** - Allow information to be supplied in the certificate request, rather than being built from Active Directory information
- **Extensions** - Application Policies should contain only Client Authentication
- **Security** - ensure relevant users and computers will be able to request certificates.

Once the new template is created on the CA, issue it.

Using the issued template, request a certificate for a target device, and export it in **.pfx** form and remember the password.

The target device requires a copy of the basic **cacert.pem** and the **.pfx** file with its password removed. You can do this using the following steps. Use the relevant OpenSSL version for the OS. OpenSSL is normally available by default on Linux and Mac devices. If you want to follow these steps on Windows you will need to download the open source version appropriate to your OS.

1. First, extract the certificate:

```
openssl pkcs12 -clcerts -nokeys -in <YourPKCSFile>.pfx -out certificate.crt
```

2. Second, the CA key:

```
openssl pkcs12 -cacerts -nokeys -in <YourPKCSFile>.pfx -out ca-cert.ca
```

3. Now, the private key:

```
openssl pkcs12 -nocerts -in <YourPKCSFile>.pfx -out private.key -passout pass:TemporaryPassword
```

4. Remove the passphrase:

```
openssl rsa -in private.key -out new.key -passin pass:TemporaryPassword
```

5. Put things together for the new PKCS-File (on Windows, **type** can be used instead of **cat**):

```
cat new.key > PEM.pem
cat certificate.crt >> PEM.pem
cat ca-cert.ca >> PEM.pem
```

6. And create the new **.pfx** file, when prompted for a password ensure that you enter an empty password (that is press enter when prompted for the password and confirmation without entering any text):

```
openssl pkcs12 -export -nodes -CAfile ca-cert.ca -in PEM.pem -out Tachyon.pfx
```

Now you have a new PKCS12 key file without passphrase on the private key part. This **Tachyon.pfx** file and the **cacert.pem** file, must be placed in one of the following locations - depending on the OS. These are hidden folders.

SSL certificate file method

This method is identical to that used for other non-Windows 1E Clients.

The client certificate file (**Tachyon.pfx**), and Certificate Authority (CA) certificate(s) for the Switch certificate (**cacert.pem** file) are stored in the hidden directory: **/Library/Application Support/1E/Client/.sslcerts**

If the client certificate (**Tachyon.pfx**) uses the same certificate trust chain as the Tachyon Switch, then **cacert.pem** is optional. This is because the client will have already cached the public certificates when it parses **Tachyon.pfx**.

If the client certificate (**Tachyon.pfx**) uses a different certificate trust chain from the Tachyon Switch, then **cacert.pem** is always required.

Key Store method

The client certificate, and Certificate Authority (CA) certificate(s) for the Switch certificate, are stored within the **macOS Key Store**.

a. Obtain and import Certificate Authority (CA) certificates

You can skip this step if the Certificate Authority (CA) certificates used by the client and the Switch are the same, and are included in the client certificate.

If the client certificate does not contain the CA certificates for itself, *or* the CA certificates used by the Switch, then the individual CA certificates will need to be obtained and imported into the macOS Key Store as follows:

1. Obtain Certificate Authority (CA) certificates as individual **.cer** files
    a. They must be exported individually because when presented with a bundle of certificates the Keychain Access UI only imports the last one it encounters.
    b. If you have a **cacert.pem** then you can use a text editor of your choice to extract each certificate as a separate **.cer** file.
    c. Use Microsoft's Manage Computer Certificates (MMC) to export as a **.cer** file using either DER encoded binary X.509 or base-64 encoded X.509 format.
2. Copy the certificate .cer file(s) onto the macOS device.
3. Open Finder and double click on each .cer file in turn to import each certificate into the macOS Keychain store. Or use File Import Items... directly from Keychain Access.
4. Open Keychain Access; the certificates should appear under the login keychain.
5. Move (by dragging) each newly imported certificate to the System keychain, to ensure that it is trusted by all users and local system processes including the Tachyon Agent which will run on this macOS machine.

6. Starting with the root CA certificate, if it is shown as not trusted, double-click to open. At the top left, above the Details section, expand the Trust section and ensure that When using this certificate Always Trust is selected and save the changes.

If you experience problems importing certificates using the Keychain Access app, for example if it reports error -25294 and `CSSM_CODE_MEMORY_ERROR`, an alternative way of importing public certificates and trusting them is to use the security command line tool. For example:

```
sudo security add-trusted-cert -d -r trustRoot -k /Library/Keychains/System.keychain <certificate.cer>
```

b. Create a client certificate for each macOS device

Each macOS device requires its own client certificate. Below is a guide for using a Microsoft Certificate Authority (CA) to issue a client certificate, then exporting it ready for installation on the device.

First, create a new Certificate template on your Certificate Authority by making a duplicate of either the Computer or Workstation template and configuring it with at least the following properties:

- General - use a suitable name such as **Tachyon Devices** and validity period.
- Request Handling - Allow private keys to be exported.
- Subject Name - Allow information to be supplied in the certificate request, rather than being built from Active Directory information.
- Extensions - Application Policies should contain only Client Authentication.
- Security - ensure relevant users and computers will be able to request certificates.

Please refer to Tachyon client requirements: Client Certificates for more information on client certificate requirements.

Once the new template is created on the Certificate Authority (CA), issue it.

Using the issued template, request a certificate for a target device, and export it in .pfx form and remember the password. The certificate and associated private key should be exported, together with all extended properties except *Include all certificates in the certification path if possible and Enable certificate privacy*.

c. Install a client certificate on each macOS device

To install a client certificate on each macOS device follow the instructions below.

1. Copy the client certificate onto the macOS device.
2. Open Keychain Access and from its File menu select **Import Items,** then navigate to the certificate. Double-click it and enter its password when prompted, then the client certificate and its associated private key should appear under the login keychain. Alternatively double-click on the **.pfx** file in the Finder.
3. Copy the client certificate and its associated private key separately under the System keychain.
4. Allow access to the private key for the Tachyon Agent by following these instructions:
5. Open the Keychain Access app, either by pressing command-space and searching for keychain or from the Finder app, top left select Go > Utilities and double click to invoke the Keychain Access app.
6. Locate the private key of the Tachyon Agent within the System keychain.
7. For macOS Sierra a private key will appear under its associated public certificate.
8. For other macOS versions the private key will not appear under its associated public certificate and you will need to locate it using the Keychain Access app.
9. Ensure the private key is present in the System keychain.
10. Double click the private key and select the Access Control tab.
11. Ensure **Confirm before allowing access** is selected and then add the Tachyon Agent binary to the **Always allow access by these applications**; adding the 1E Client by its path name, which is typically /Library/Application Support/1E/Client/1E.Client

> The Keychain Access app cannot be guaranteed to always refresh its view following, for example, an update of access control permission. However, after exiting the app, killing the process `/Applications/Utilities/Keychain Access.app/Contents/MacOS/Keychain Access` and then restarting the app, will ensure the Keychain Access view will correctly display the contents of the Key Store.

An alternative approach to using the Keychain Access app is to import the **.pfx** file using the command-line security command:

```
sudo security import <download location>/<macOSpfx>.pfx -k /Library/Keychains/System.keychain -t agg -f
pkcs12 -P <password> -T '/Library/Application Support/1E/Client/1E.Client'
```

This command imports a client certificate and the associated private key (termed **an identity** on macOS) from a **.pfx** file to a <keychain> as an aggregated type in PKCS #12 format using the specified password and giving access to the Tachyon client.

> When upgrading the 1E Client, by default the new version will not automatically be granted access to the private key in the Key Store. You can either repeat the import process during upgrade, or avoid the overhead by using the command-line security command with the import -A flag instead of -T . Please note that Apple do not recommend this approach as it is considered less secure. For more information on the security command please refer to the macOS page: https://ss64.com/osx/security-export.html.

If you do not grant correct access to the client certificate's private key, and Tachyon client debug logging is configured the Tachyon client will report that it cannot obtain the client certificate's private key because the user name or passphrase entered is not correct.

Starting the 1E Client

To load and start the 1E Client use the command:

```
sudo launchctl load -w /Library/LaunchDaemons/com.1e.pkg.1E.Client.plist
```

If it is necessary to stop the 1E Client use the command:

```
sudo launchctl unload /Library/LaunchDaemons/com.1e.pkg.1E.Client.plist
```

If the 1E Client should fail to start correctly please check:

- **sudo launchctl list | grep -i 1e** to confirm the service is actually running. It should appear as **com.1e.pkg.1E.Client**
- **/Library/Logs/1E.Client.Daemon.log** shows any service start errors
- **tail -f /Library/Logs/1E.Client.log** shows the current operation of the 1E Client
- If necessary raise the **/Library/Application Support/1E/Client/1E.Client.conf** to **LoggingLevel=debug** using a suitable text editor such as vi and then restart the 1E Client.

Uninstallation

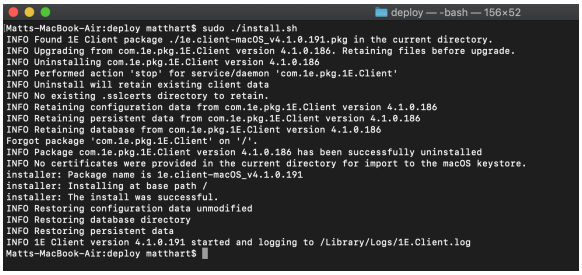
To uninstall the 1E Client from a macOS device invoke the `uninstall.sh` script from within a sudo bash script. `uninstall.sh` is available within the scripts directory of the Apple Disk Image (dmg) file of the macOS 1E client.

Alternatively, enter the following commands inside a sudo bash shell:

```
bash$ cd /Library
bash$ pkgutil --only-files --files com.1e.pkg.1E.Client | tr '\n' '\0' | xargs -n 1 -0 rm
bash$ pkgutil --forget com.1e.pkg.1E.Client
```

Upgrading

The **install.sh** script can be used for upgrading as well as installing. The bash script will deduce parameters from the existing configuration so no parameters are necessary, as shown in the picture opposite.



When upgrading the 1E Client, by default the new version will not automatically be granted access to the private key in the Key Store. You can either repeat the import process during upgrade, or avoid the overhead by using the command-line security command with the import -A flag instead of -T . Please note that Apple do not recommend this approach as it is considered less secure. For more information on the security command please refer to the macOS page: https://ss64.com/osx/security-export.html.

If you do not grant correct access to the client certificate's private key, and Tachyon client debug logging is configured the Tachyon client will report that it cannot obtain the client certificate's private key because the user name or passphrase entered is not correct.