

PKIaaS.io Documentation

Lyas Spiehler

Copyright © PKIaaS.io 2019-2024

Table of contents

1. Welcome to PKIaaS.io Documentation	5
2. Getting Started	6
2.1 Login	6
2.2 Create a Certificate Authority	6
2.2.1 CA Configuration Options	6
2.2.2 Switch CAs	6
2.3 Using Your New CA	6
2.4 Certificate Templates	6
2.4.1 Manage Templates	6
2.4.2 Create New Templates	6
2.5 Issue a Certificate	7
2.5.1 Public Certificate Request	7
2.5.2 Submit CSR	7
2.5.3 Create Pre-Approved Request	7
2.6 Approve Pending Certificate Requests	7
3. Certificate Authorities	8
3.1 Overview	8
3.1.1 Options	8
3.2 Create Certificate Authorities	9
3.2.1 Create a Root CA	9
3.2.2 Create an Intermediate/Subordinate CA	9
3.3 Edit Certificate Authorities	10
3.4 Delete Certificate Authorities	11
4. Certificate Templates	12
4.1 Overview	12
4.2 Create Templates	13
4.3 Edit Templates	14
4.3.1 Options	14
4.4 Delete Templates	16
4.5 Service URLs	17
5. ACME	18
5.1 Overview	18
5.2 Create an ACME Policy	19
5.3 Generate EAB Credentials	20

5.4 Register an ACME Account	21
5.4.1 ACME Clients	21
5.5 ACME Clients	22
5.5.1 Certbot	22
5.5.2 Certify the Web	23
5.5.3 Traefik	29
6. SCEP	30
6.1 Overview	30
6.1.1 Enabling SCEP	30
6.1.2 SCEP URL	30
6.1.3 SCEP Validation	30
6.2 Intune Integration	31
6.2.1 Prerequisites	31
6.2.2 Enabling Intune Integration	31
6.2.3 Create and assign SCEP certificate profiles in Intune	31
6.3 Microsoft NDES Compatibility	32
6.3.1 Enable Microsoft NDES Compatibility	32
6.3.2 Microsoft NDES Compatibility URL	32
6.3.3 Authentication	32
6.3.4 Restrict Access	32
6.4 Apple Configuration Profile	33
6.4.1 Prerequisites	33
6.4.2 Create an Apple Configuration Profile	33
6.4.3 Download and Install the Apple Configuration Profile	33
6.5 Static Passphrase	34
6.5.1 Enable Static Passphrase	34
6.6 SCEP Security	35
7. OCSP	36
7.1 Overview	36
7.2 OCSP Signing Certificate	36
7.3 OCSP Responder URL	36
7.4 OCSP in the AIA Extension	36
8. CRL	37
8.1 Overview	37
8.2 CRL Generation	37
8.3 CRL URL	37
8.4 CRL Distribution Point	37

9. Timestamping	38
9.1 Overview	38
9.2 Timestamping Service	38
9.3 More Information	38
10. Iot-HSM	39
10.1 Overview	39
10.1.1 Overview	39
10.2 Installation	40
10.3 Provisioning	41
11. Post-Quantum Cryptography	43
11.1 Overview	43
11.2 ML-DSA	44
11.2.1 Create a ML-DSA Certificate Authority	44
11.2.2 Generate ML-DSA Private Keys and CSRs	44
11.2.3 Issue a ML-DSA Certificate	44
12. Use Cases	45
12.1 Sample Use Case	45

1. Welcome to PKIaaS.io Documentation

Thank you for visiting the documentation for PKIaaS.io. This site is under construction. Contributors are welcome!

PKIaaS.io is here to make PKI functionality—usually limited to costly, complex platforms designed for enterprise customers—accessible to a broader audience. PKIaaS.io was designed to serve not only small to medium-sized businesses but also home lab users, developers, testers, and PKI enthusiasts of all kinds!

My name is Lyas Spiehler, and I was inspired to create PKIaaS.io out of a passion and curiosity for PKI, as well as a desire to fill the gap in accessible resources for learning about and easily generating x509 certificates. PKIaaS.io enables the creation of all types of certificates—including SMIME, code signing, TLS/SSL, document signing, and more—while offering comprehensive services you'd expect from a robust PKI platform, such as CRL, OCSP, SCEP, ACME, timestamping, certificate transparency (SCT), etc.

If you're new to PKIaaS.io, be sure to check out our [Getting Started](#) guide!

2. Getting Started

2.1 Login

Getting started with PKIaaS.io is easy! All you need to do is click "Manage PKI" at the top right of the main page and choose the login you'd like to use.

2.2 Create a Certificate Authority

All the certificates you'll create need to be signed by a certificate authority, so the first step is to create one. Navigating to any other page of the site before you've created a CA will redirect you back to a page forcing you to create one. If you're logging in for the first time, and you haven't created a CA before, you'll be asked to choose between a managed CA and a CA backed by an HSM. If you're new to PKI or you'd just like to take PKIaaS.io for a test drive, it would be best to choose the managed option to get going more quickly. Read more about these options [here](#).

Note: PKIaaS.io is a comprehensive PKI platform that requires the creation of a certificate authority to issue certificates. However, if your use case involves generating a simple self-signed certificate without the need for a certificate authority, a straightforward tool like [CertificateTools.com](#) may be a more convenient alternative.

2.2.1 CA Configuration Options

If you're still learning about PKI, feel free to use the default options and just update the name when creating your first CA. This is the easiest option to start issuing certificates and getting a feel for the site. When you're done choosing options, click "Create New CA". You'll then be redirected to the "Manage CAs" page and your new CA will appear in the drop down at the top of the page.

2.2.2 Switch CAs

If you create more than one CA, you'll have the option of switching between them by selecting the one you'd like to manage from the dropdown at the top of the PKI management page to the right of the navigation pane. All other pages on the site will show only the resources—templates, certificates, services, etc.—within the context of the selected CA.

2.3 Using Your New CA

Select the CA you want to work with from the dropdown at the top to the right of the navigation pane. This dropdown changes the scope of the site to manage the active CA. All actions performed—signing certificates, creating templates, creating ACME policies, etc.—will be done within the context of the selected CA.

2.4 Certificate Templates

2.4.1 Manage Templates

Certificate templates can be managed by navigating to **Certificate Templates -> Manage Templates** in the navigation pane on the left of the page. Certificate templates are used to enforce compliance of predefined standards for issuing certificates with your CA. These standards include key usages, subject attributes, basic constraints, and other certificate attributes. The use of templates is important to prevent accidental mis-issuance and to ensure that issued certificates are compliant with your security practices. All certificate signing on PKIaaS.io will be associated with a template.

2.4.2 Create New Templates

By default, a "Web Server" template is automatically built for every CA that is created. This template has a good out-of-the-box configuration for issuing TLS/SSL certificates for web servers, but new, custom certificate templates can be created by

navigating to **Certificate Templates -> Create Template**. Enter a name for the template and choose default settings by selecting an option from the "Starter Template" dropdown. Click "Create Template" to proceed to the configuration page, where the template options can be customized. See more information on the available options [here](#). Once satisfied with the settings, click "Save" to finalize the changes and return to the template management page.

2.5 Issue a Certificate

In the navigation page on the left, navigate to **Certificate Templates -> Manage Templates** and click on the template you'd like to use. From the menu, you can choose either, "Public Certificate Request", "Submit CSR", or "Create Pre-Approved Request." These options support various workflows for issuing certificates.

2.5.1 Public Certificate Request

If your workflow involves receiving certificate requests from users, the "Public Certificate Request" option allows you to send an email to users inviting them to easily submit a certificate request. It also shows the public URL that can be shared for others to submit a certificate request. This option allows users to submit a CSR or fill out a form containing the desired subject attributes and SANs. The latter option can be helpful for users that are not familiar with x509 certificates and includes an option to generate a private key automatically in the browser when the user claims their certificate after it is approved. To review and approve these requests, navigate to **X509 Certificates -> Pending Requests**.

2.5.2 Submit CSR

The "Submit CSR" option provides an input field to paste the contents of a certificate signing request. To review and approve these requests, navigate to **X509 Certificates -> Pending Requests**.

2.5.3 Create Pre-Approved Request

The "Create Pre-Approved Request" feature enables a PKI administrator to pre-configure all the settings typically chosen during the approval process for a pending certificate request. The administrator can either redeem the certificate immediately or send an email to the user, inviting them to claim it. In the certificate claim workflow, the user can set a password to protect the PKCS#12 file, and a 2048-bit RSA private key will be generated in the browser. Alternatively, a CSR can be submitted, however, the subject attributes and SANs specified in it will be overridden by the pre-approved attributes in the signed certificate.

2.6 Approve Pending Certificate Requests

Certificates requested using the "Public Certificate Request" or "Submit CSR" options require approval. The requests pending approval can be found by navigating to **X509 Certificates -> Pending Requests**. Here you will find options to Approve, Sign, Decline, or Re-Send emails for various types of certificate requests. Not all options are available for all types of certificate request workflows. These options can be performed in bulk, or the requests can be selected individually where additional options are provided to override subject attributes, subject alternative names (SANs), and to customize the validity time of the certificate instead of taking the template default.

3. Certificate Authorities

3.1 Overview

A certificate authority (CA) serves as the foundation upon which all functionality on PKIaaS.io is built. Tasks like working with templates, issuing certificates, and managing ACME policies all operate within the scope of a CA. As such, the first step after signing into PKIaaS.io is creating a CA. Users will be automatically redirected to a setup page and prompted to create a CA before accessing other features of the site.

Users can choose between creating a managed CA or one backed by an HSM. With a managed CA, the private key is securely stored using AES-256 encryption, offering a streamlined setup without additional configuration. Alternatively, the HSM-backed option integrates with a [PKIaaS.io IoT-HSM](#), enabling key management via a YubiKey 5. In this setup, all signing operations are routed to the IoT-HSM, where they are processed by the YubiKey. For detailed guidance on creating a CA with IoT-HSM, refer to the [IoT-HSM documentation](#).

3.1.1 Options

Subject Attributes

When creating a certificate authority (CA), users are provided with a range of flexible options to tailor their setup. The first six fields customize the subject attributes of the CA, with only the "Certificate Authority Name" (common name) being mandatory.

Private Key

Next, users can configure the private key settings. Various types of private key types are supported including RSA, ECC, and CRYSTALS-dilithium. Here, users can also opt to use HSM (as previously mentioned) or import an existing private key.

Hash Algorithm, Validity, and Path Length

Further customization options include selecting the signature hash algorithm, specifying the path length, and setting the certificate authority's validity period. The signature algorithm chosen here determines how the CA's certificate is self-signed. Depending on the private key used, additional signature options may become available for signing certificates within the certificate templates after the CA is established. For more details on "Path Length," refer to RFC5280, which explains the "basic constraints" x509v3 attributes.

Key Usage

It is not recommended to make changes to the default "Key Usage" and "Extended Key Usage" options. Making changes could result in issues with your certificate authority.

Intermediate CA

Finally, the "CSR Only" option is designed for cases where the CA being created will function as an intermediate or subordinate certificate authority rather than a root CA. This option generates a Certificate Signing Request (CSR) that must be signed by another CA. The signing can be performed by an external CA (such as Microsoft or EJBCA) or another PKIaaS.io CA. Once the signed certificate is obtained, it must be imported before the CA can be activated and used.

3.2 Create Certificate Authorities

3.2.1 Create a Root CA

Users can create a root certificate authority by logging into PKIaaS.io and navigating to **Certificate Authorities -> Create New CA**. Fill out the form with the appropriate options for the desired CA. See [certificate authority overview](#) for documentation on all of the available options. When creating a root CA, it is necessary to leave the "CSR Only" option at the bottom on the form unchecked. Upon clicking "Create New CA," a new private key will be generated, and a self-signed, root certificate authority will be created and ready to issue certificates. See the [getting started](#) guide for more documentation about issuing certificates.

3.2.2 Create an Intermediate/Subordinate CA

To create an intermediate/subordinate certificate authority, ensure that the "CSR Only" option is selected on the **Certificate Authorities -> Create New CA** page when selecting the options for the desired CA. See [certificate authority overview](#) for documentation on all of the available options. Upon clicking "Create New CA," a new private key will be generated, and a CSR will be created. The browser will be redirected to a page containing the CSR where it can be copied to the clipboard and signed by another certificate authority.

Note: If at any time you need to navigate back to the CSR page, go to **Certificate Authorities -> Manage CAs**, click on the CA and select "Download/Copy CSR" from the menu.

Sign the CSR with a Root CA on PKIaaS.io

If a certificate authority not managed by PKIaaS.io will be used to sign the CSR, see the documentation for the CA on how to do so. The following instructions will detail how this can be done on a CA managed by PKIaaS.io.

While logged into PKIaaS.io, select the signing root CA from the dropdown at the top of the page. This will switch the context of the admin page to the selected CA. If a template for creating certificate authorities has not already been created, navigate to **Certificate Templates -> Create Template**. Enter "Intermediate CA" as the template name and choose "Certificate Authority" as the started template. The browser will be directed to a page where the template can optionally be customized. Click "Save" to finish creating the new template.

The browser should now be at the **Certificate Templates -> Manage Template** page. Click on the "Intermediate CA" template and select "Submit CSR" from the menu. Paste the CSR copied to the clipboard in a previous step and click submit. See the note above for instructions to navigate back to the CSR if needed.

The submitted CSR must now be approved in order to be signed. Navigate to **X509/Certificates -> Pending Requests**. Click on the pending certificate request for the new intermediate CA. A modal will appear where some certificate attributes can optionally be overridden. Click "Sign" to sign the submitted CSR and issue the certificate.

Finally navigate to **X509/Certificates -> Issued Certificates**. Click on the newly issued certificate and copy it to the clipboard. This certificate must be imported to finish creating the new intermediate CA. Navigate to **Certificate Authorities -> Manage CAs**, click on the new intermediate CA and choose "Import Signed Certificate" from the menu. Paste the signed certificate into the text area and click "Import Certificate." The intermediate CA is now ready to issue certificates. Select it from the dropdown at the top of the page to begin using it.

3.3 Edit Certificate Authorities

To edit a certificate authority, login to PKIaaS.io and navigate to **Certificate Templates -> Manage CAs** in the navigation pane on the left of the page. Click on a CA and select "Edit CA" from the menu. The CPS (certificate practice statement) can be customized here as well as the service certificate settings. The service certificate settings manage the key type, key size and hash algorithm used when creating certificates for services like time stamping and OCSP.

3.4 Delete Certificate Authorities

To delete a certificate authority, login to PKIaaS.io and navigate to **Certificate Templates -> Manage CAs** in the navigation pane on the left of the page. Click on a CA and select "Edit CA" from the menu. Click the red "Delete" button to delete the CA. This action is not reversible and will delete all certificates and templates associated with the CA.

4. Certificate Templates

4.1 Overview

Certificate templates are used to enforce compliance of predefined standards for issuing certificates with your certificate authority. These standards include key usages, subject attributes, basic constraints, and other certificate attributes. Certificate templates also serve as the entry point for many of the certificate issuance options on PKIaaS.io. Read more about these certificate issuance options on the [getting started page](#).

4.2 Create Templates

To create a new certificate template, login to PKIaaS.io and navigate to **Certificate Templates -> Create Template**. Enter a name for the template and choose default settings by selecting an option from the "Starter Template" dropdown. Click "Create Template" to proceed to the configuration page, where the template options can be customized. Once satisfied with the settings, click "Save" to finalize the changes and return to the template management page.

4.3 Edit Templates

To edit the settings for a certificate template, login to PKIaaS.io and navigate to **Certificate Templates -> Manage Templates** in the navigation pane on the left of the page. Click on a template and select "Edit Template".

4.3.1 Options

General

- **Enabled** - enables/disables the template
- **Allow unsolicited certificate web requests** - enables/disables the "Public Certificate Request" URL for the template
- **Allow the use of a valid certificate to authorize an automatic renewal request** - enables/disables certificate renewal authorization using a previously issued certificate using the renewal options found on the CA's CPS page
- **Automatically revoke old certificate** - revoke the old certificate used to authorize renewal
- **Require certificate to be within** - don't allow renewal using a certificate to authorize renewal unless it is within the specified number of days of expiration
- **Copy subject and SANs from original certificate during renewal** - use the same subject and subject alternative names from the old certificate used to authorize renewal
- **Signing Algorithm** - customize the hash algorithm used to sign certificates
- **Validity Period (Days)** - specify the default length of time the issued certificate will be valid—for manually issued certificates. This option can be overridden at the time of signing when the request is in **X509 Certificates -> Pending Requests**
- **Description** - provide a description for the template

Subject

To honor the subject attributes requested within the CSR, use the **Supplied by the CSR** option. Otherwise, the subject attributes in the CSR will be overridden by the defaults defined here.

Key Usage

Select the key usages and extended key usages to be enabled for certificates issued using this template. Custom comma-separated extended key usage OIDs can be entered into the **Custom EKU OIDs** field.

Basic Constraints

The Basic Constraints extension in an X509 certificate defines whether the certificate is a Certificate Authority (CA) or an end-entity (non-CA) certificate. Set the **Certificate Authority** option to yes if the template will be used to sign subordinate CAs (intermediate CAs) certificate authorities. Path length defines the maximum number of subordinate CAs that can exist below this certificate in the certificate hierarchy.

CRL / OCSP / AIA / CPS

- **Include AIA Extension in Certificate** - The Authority Information Access (AIA) attribute is an optional but commonly included extension in an X.509 certificate. It provides information about how to access important resources or services related to the certificate issuer. Use this option to enable or disable the AIA attribute in certificates issued using this template.
- **Custom URL** - Allows customization of the URL where the issuing CA's certificate (or chain) can be downloaded.
- **Include OCSP Extension in Certificate** - Enable or disable the OCSP Responder attribute within the AIA extension in certificates issued using this template.
- **Include CRL Extension in Certificate** - Enable or disable the CRL Distribution Point (CDP) attribute in certificates issued using this template.

- **Include CPS Extension in Certificate** - Enable or disable the Certificate Policies attribute in certificates issued using this template. This attribute contains the URL to the Certificate Practice Statement (CPS) and the policy identifier.
- **Policy** - Disable, select a predefined policy identifier, or enter a custom OID.

SCEP

Simple Certificate Enrollment Protocol is a protocol designed to simplify the process of managing and issuing digital certificates in a scalable and automated manner. It is commonly used in environments where devices like routers, printers, and mobile devices need to enroll for certificates to enable secure communications. The SCEP URL for the template can be found by navigating to **Certificate Templates -> Manage Templates**, next click on a template, and choose the **Show Template Service URLs** option. A modal will appear revealing the template service URLs including the SCEP URL.

- **Enable SCEP protocol** - Enable or disable SCEP for the template.
- **Enable SCEP Static Passphrase** - Enable or disable a static passphrase for SCEP services on the template. This may be required for SCEP processes that require the same passphrase to be used for multiple SCEP clients.
- **Enable Microsoft Intune/Endpoint Manager Integration** - Enable or disable Microsoft Intune SCEP integration for the template.
- **Strip Root from GetCACert Requests** - Exclude the root CA and return only the root in responses to SCEP GetCACert requests.
- **Enable Microsoft SCEP compatibility** - Enable or disable the Microsoft SCEP (NDES) compatibility. This option enabled PKIaaS.io to mimic the /CertSrv/mscep_admin/ URL on Microsoft NDES servers to generate "enrollment challenge passwords" for processes and workflows designed to work with this Microsoft NDES functionality. The URL for this page can be found by navigating to **Certificate Templates -> Manage Templates**, next click on a template, and choose the **SCEP Admin (Microsoft compatibility)** option to navigate to the URL in a new tab.
- **Enable Microsoft SCEP basic auth** - Enable or disable basic HTTP authentication for the Microsoft SCEP compatibility page. A username and password must be provided if this option is selected.
- **Enable IP ACL** - Enable or disable the IP-based access control list (firewall functionality) for access to SCEP requests, the Microsoft compatibility page (if enabled) or both. Enter a list of IPs allowed to access the resources.

CT Logs

For templates associated with root CAs or intermediate CAs that chain up to root CAs that require issued certificates to be submitted to certificate transparency logs, enter a comma-separated list of URLs where these certificates should be submitted here. The CT log must support logging for the CA associated with the template. This is something that must be coordinated with the owner of the CT log.

4.4 Delete Templates

To delete a certificate template, login to PKIaaS.io and navigate to **Certificate Templates -> Manage Templates** in the navigation pane on the left of the page. Click on a template and select "Edit Template". If the template has no active certificates, you can use the red Delete button to remove it. Note that all certificates issued under the template must either be expired or revoked, and the revocation reason cannot be "Certificate Hold".

4.5 Service URLs

The SCEP services URL, ACME directory URL, and a URL to allow users to make public certificate requests can all be found by navigating to **Certificate Templates -> Manage Templates** while logged into PKIaaS.io, and then by clicking one of the templates and selecting the **Show Template Service URLs** option.

5. ACME

5.1 Overview

The ACME (Automatic Certificate Management Environment) protocol is a standard for automating the process of obtaining, renewing, and managing SSL/TLS certificates. It was developed by the Internet Security Research Group (ISRG), the organization behind Let's Encrypt, and is specified in [RFC 8555](#). PKIaaS.io supports version 2 of the ACME protocol with HTTP-01 and DNS-01 challenges.

Note: PKIaaS.io requires use of an EAB (External Account Binding) to register an ACME account. Because of this requirement, only ACME clients that support EAB will work with the PKIaaS.io ACME service.

A few steps must be completed before requesting a certificate with an ACME client:

1. [Create an ACME Policy](#)
2. [Generate EAB Credentials](#)
3. [Register an ACME Account](#)

The ACME directory URL for all PKIaaS.io ACME requests is <https://acme-v02-api.pki.aa.io/directory>

5.2 Create an ACME Policy

An ACME policy associates ACME certificate requests with a specified certificate template and allows a comma-separated list of "pre-approved domains" to be supplied. Pre-approved domains will be automatically authorized and will not be required to do any ACME HTTP or DNS challenges. Wildcard domains are supported to pre-approve an entire root domain and all subdomains.

To create an ACME policy, login to PKIaaS.io and navigate to **ACME -> Policies**. Click "Create", enter a unique name for the new ACME policy, select which template should be used when signing certificates requested using this policy, optionally enter any pre-approved domains, and click "Create Policy."

Next, see [Generate EAB Credentials](#).

5.3 Generate EAB Credentials

An ACME policy must be created before EAB credentials can be generated. See [Create an ACME Policy](#) if this step hasn't been completed yet.

The use of EAB (external account binding) allows PKIaaS.io to associate ACME accounts with a specific user's PKIaaS.io account, CA, and ACME policy. To generate an EAB for registering a new ACME account, login to PKIaaS.io and navigate to **ACME -> Policies**. Click the gears in the "External Account Bindings" column of one of the ACME policies to navigate to the EAB management page for the policy and click "Create". Enter a unique name for the EAB. Enter the desired value for "Allowed Use Count". This value represents the number of times this EAB can be used to register unique ACME accounts. Lastly, enter the expiration date after which ACME new account registrations will no longer be authorized with this EAB and click "Create EAB".

When the new EAB is generated, a modal will appear providing the ACME directory URL, the EAB KID and HMAC Key, and some example commands to use the values provided to register a new ACME account using the certbot ACME client. These values can be used to register an ACME account with any ACME client that supports external account bindings. The KID and HMAC Key will only be displayed once, so be sure to copy them before closing the modal.

Note: After an ACME account is registered with an EAB, it is permanently associated with the policy the EAB was generated for. It is safe to delete an EAB after an ACME account had been registered with it.

Next, see [Register an ACME Account](#).

5.4 Register an ACME Account

An ACME EAB credential must be generated before an ACME account can be registered with PKIaaS.io. See [Generate EAB Credentials](#) if this step hasn't been completed yet.

In order to use the ACME protocol to request a certificate from PKIaaS.io, an ACME account must be first be registered using an EAB. See the links below for more information getting started with various ACME clients.

5.4.1 ACME Clients

- [certbot](#)
- [Certify the Web](#)
- [Traefik](#)

5.5 ACME Clients

5.5.1 Certbot

Be sure to have the EAB KID and HMAC Key ready for the steps below. See [Generate EAB Credentials](#) for more information.

All values that appear in ALL CAPS in the examples below must be customized with values specific to you.

Register an Account Only

```
certbot register --server https://acme-v02-api.pki.aa.io/directory --email YOUREMAIL@YOURDOMAIN.COM --agree-tos --eab-kid  
YOUREABKID --eab-hmac-key YOUREABHMACKEY
```

Register an Account and Order a Certificate for Apache

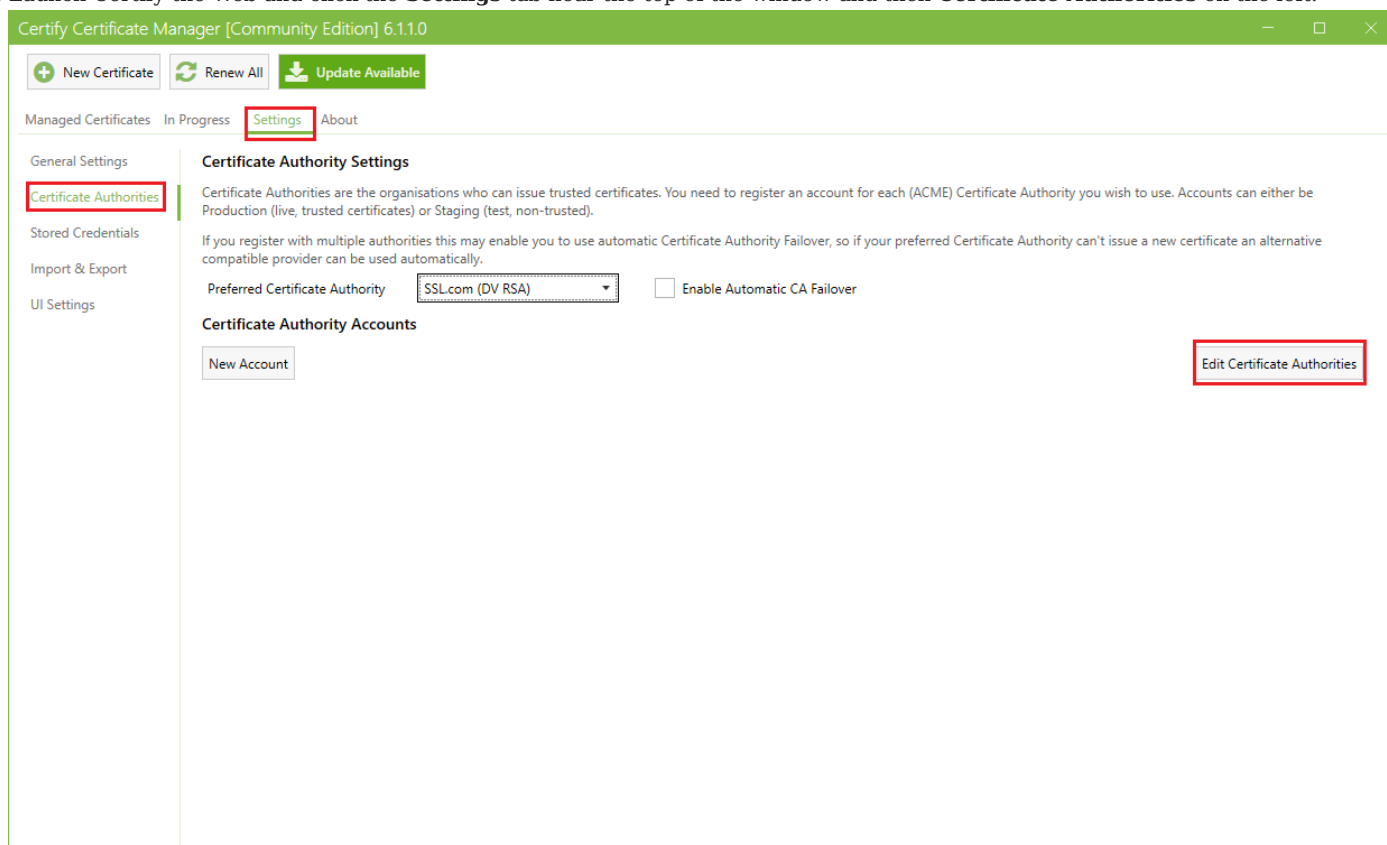
```
certbot --apache --server https://acme-v02-api.pki.aa.io/directory --email YOUREMAIL@YOURDOMAIN.COM --agree-tos -d MYDOMAIN.COM -  
d WWW.MYDOMAIN.COM --eab-kid YOUREABKID --eab-hmac-key YOUREABHMACKEY
```

See the [EFF certbot website](#) for more information about certbot.

5.5.2 Certify the Web

Be sure to have the EAB KID and HMAC Key ready for the steps below. See [Generate EAB Credentials](#) for more information.

1. Download Certify the Web from <https://certifytheweb.com/> and install it.
2. Launch Certify the Web and click the **Settings** tab near the top of the window and then **Certificate Authorities** on the left.



3. Click **Edit Certificate Authorities**.

- In the **Certificate Authority** field, leave "(New Certificate Authority)" selected.
- In the **Title** field, enter "PKIaaS.io" or something else to uniquely identify the new CA.
- In the **Production API** field, enter the PKIaaS.io ACME directory URL "https://acme-v02-api.pki.aa.s.io/directory".
- Optionally enter a **Description**. The **Staging API** field may be left blank.
- Enable all of the options under **Features**
- Click **Save**

Edit Certificate Authority

Certificate Authority: (New Certificate Authority)

Title: PKIaaS.io

Description: (Optional description)

Production API: https://acme-v02-api.pki.aa.s.io/directory

Staging API: (Url for the staging/test directory endpoint)

Features:

- Enabled
- Email Address Required
- Allow Untrusted TLS
- Allow Internal Hostnames

Buttons: Delete, Save, Cancel

4. Click the **New Account** button on the far left of the Edit Certificate Authorities button clicked in step 3.
 - From the **Certificate Authority** dropdown, select the CA title that was entered in step 5.
 - Enter the **Email Address** to be associated with the new ACME account.
 - Check **Yes, I Agree** to agree to the terms and conditions for the CA.

Edit ACME Account

Account Settings Advanced

To request certificates you need to register with each of the Certificate Authorities that you want to use.

Certificate Authority PKIaaS.io

Email Address MYEMAIL@MYDOMAIN.COM

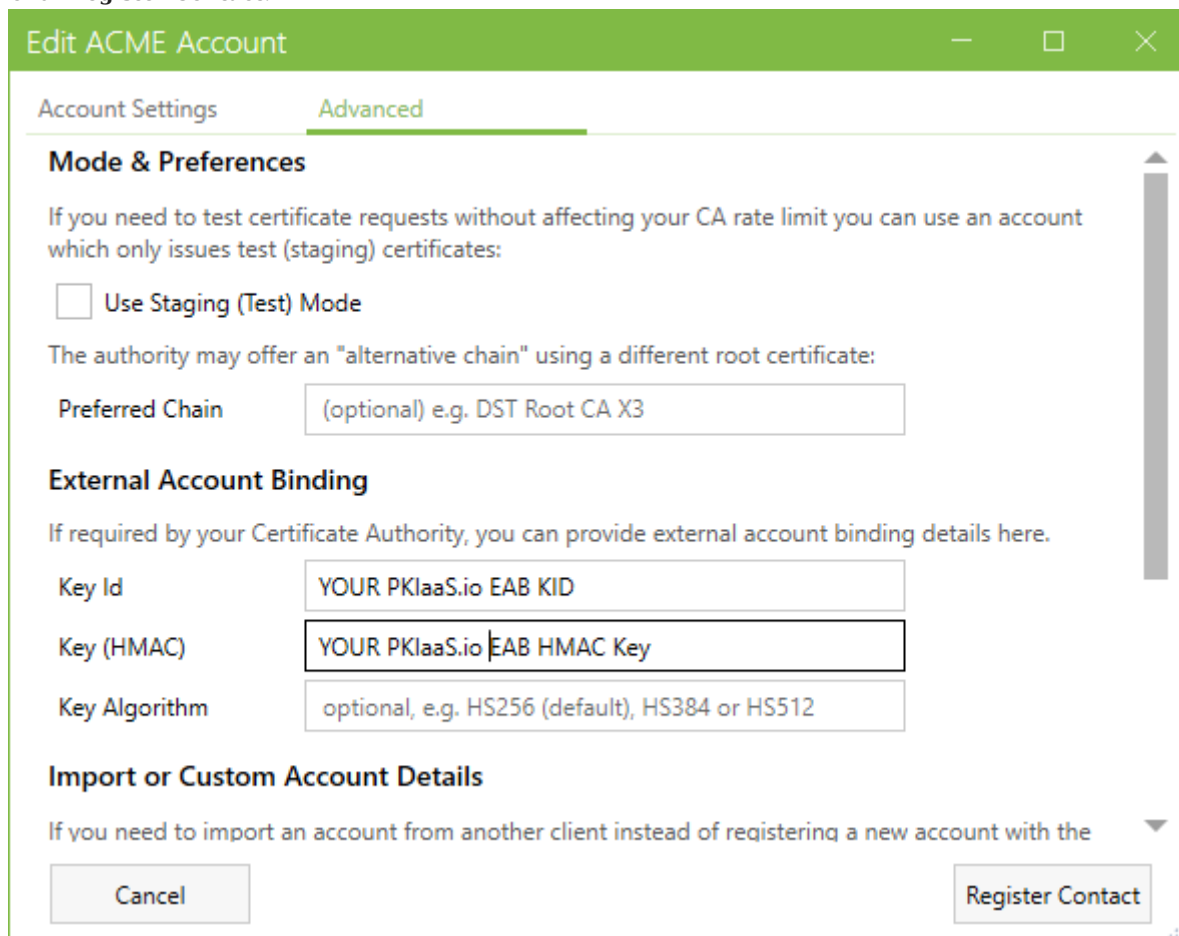
The email address provided may be used to notify you of upcoming certificate renewals if required. Invalid email addresses will be rejected by the Certificate Authority.

To proceed, confirm that you agree to the current terms and conditions for this Certificate Authority. Yes, I Agree

Cancel Register Contact

5. Click **Advanced** to enter the EAB credentials.

- Enter the **Key Id** and **Key (HMAC)** using the "EAB KID" and "EAB HMAC Key" provided when generating the EAB credentials.
- Click **Register Contact**.

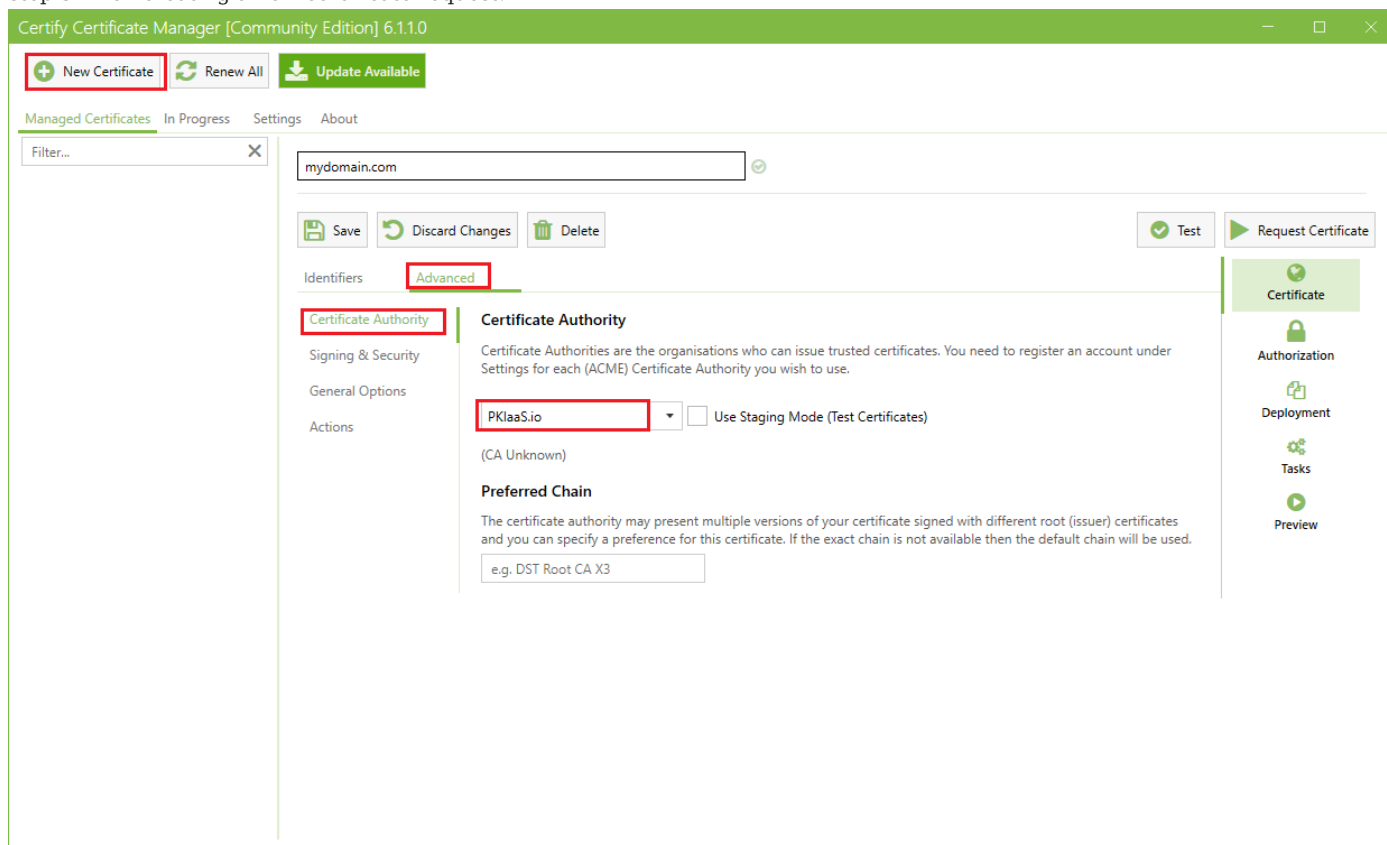


The screenshot shows a window titled "Edit ACME Account" with a green header bar. Below the header, there are two tabs: "Account Settings" and "Advanced", with "Advanced" being the active tab. The "Advanced" tab contains the following sections:

- Mode & Preferences**
 - A text block: "If you need to test certificate requests without affecting your CA rate limit you can use an account which only issues test (staging) certificates:"
 - A checkbox labeled "Use Staging (Test) Mode" which is currently unchecked.
 - A text block: "The authority may offer an 'alternative chain' using a different root certificate:"
 - A text input field labeled "Preferred Chain" with the placeholder text "(optional) e.g. DST Root CA X3".
- External Account Binding**
 - A text block: "If required by your Certificate Authority, you can provide external account binding details here."
 - A text input field labeled "Key Id" with the placeholder text "YOUR PKIaaS.io EAB KID".
 - A text input field labeled "Key (HMAC)" with the placeholder text "YOUR PKIaaS.io EAB HMAC Key".
 - A text input field labeled "Key Algorithm" with the placeholder text "optional, e.g. HS256 (default), HS384 or HS512".
- Import or Custom Account Details**
 - A text block: "If you need to import an account from another client instead of registering a new account with the".

At the bottom of the dialog, there are two buttons: "Cancel" on the left and "Register Contact" on the right.

6. The ACME account is now registered and ready to requests certificates from PKIaaS.io. Be sure to select the new CA created in step 3 when creating a new certificate request.



See Certify the Web's [Certificate Manager Documentation](#) for more information about Certify The Web.

5.5.3 Traefik

Be sure to have the EAB KID and HMAC Key ready for the steps below. See [Generate EAB Credentials](#) for more information.

All values that appear in ALL CAPS in the examples below must be customized with values specific to you.

1. Create a docker-compose.yml on your server with the following content:

```

1  version: "3.3"
2
3  services:
4
5    traefik:
6      image: "traefik:v3.2"
7      container_name: "traefik"
8      command:
9        #- "--log.level=DEBUG"
10       - "--api.insecure=true"
11       - "--providers.docker=true"
12       - "--providers.docker.exposedbydefault=false"
13       - "--entryPoints.web.address=:80"
14       - "--entryPoints.websecure.address=:443"
15       - "--certificatesresolvers.myresolver.acme.httpchallenge=true"
16       - "--certificatesresolvers.myresolver.acme.httpchallenge.entrypoint=web"
17       - "--certificatesresolvers.myresolver.acme.caserver=https://acme-v02-api.pki.aa.io/directory"
18       - "--certificatesresolvers.myresolver.acme.eab.kid=YOUREABKID"
19       - "--certificatesresolvers.myresolver.acme.eab.hmacencoded=YOUREABHMACKEY"
20       - "--certificatesresolvers.myresolver.acme.email=YOUREMAIL@YOURDOMAIN.COM"
21       - "--certificatesresolvers.myresolver.acme.storage=/letsencrypt/acme.json"
22     ports:
23       - "80:80"
24       - "443:443"
25       - "8080:8080"
26     volumes:
27       - "/letsencrypt:/letsencrypt"
28       - "/var/run/docker.sock:/var/run/docker.sock:ro"
29
30    whoami:
31      image: "traefik/whoami"
32      container_name: "simple-service"
33      labels:
34        - "traefik.enable=true"
35        - "traefik.http.routers.whoami.rule=Host(`whoami.example.com`)"
36        - "traefik.http.routers.whoami.entrypoints=websecure"
37        - "traefik.http.routers.whoami.tls.certresolver=myresolver"

```

2. Create a "letsencrypt" directory in the same directory you created the docker-compose.yml file.
3. Replace the values for **certificatesresolvers.myresolver.acme.eab.kid**, **certificatesresolvers.myresolver.acme.eab.hmacencoded**, and **certificatesresolvers.myresolver.acme.email** with your own correct values.
4. Replace whoami.example.com by your own domain within the traefik.http.routers.whoami.rule label of the whoami service.
5. Optionally uncomment the following line if you want to test/debug:

```

#- "--log.level=DEBUG"

```

6. Run `docker-compose up -d` within the folder where you created the previous file.
7. Wait a few moments for the certificate request to complete, and visit `https://your_own_domain` to confirm everything worked as expected.

See the [Traefik website](#) for more information about Traefik.

6. SCEP

6.1 Overview

SCEP (Simple Certificate Enrollment Protocol) is a long-established, open source protocol that automates the request and issuance of digital certificates from a certificate authority over HTTP. SCEP is often used by mobile device management (MDM) solutions to issue certificates to managed devices. It is also often used by network devices, such as routers and switches, to obtain certificates for secure communication.

6.1.1 Enabling SCEP

PKIaaS.io supports SCEP for certificate enrollment, but it is not enabled by default. SCEP can be enabled and disabled on a per-template basis. To enable SCEP for a template log in to PKIaaS.io and navigate to **Certificate Templates -> Manage Templates**. Click on the template you wish to enable SCEP for, then click "Edit Template" and select the "SCEP" tab. From here, you can enable SCEP and configure the SCEP settings for the template.

6.1.2 SCEP URL

The SCEP URL is unique to each template. To find the SCEP URL for a template, navigate to **Certificate Templates -> Manage Templates**, click on the template you wish to use and select "Show Template Service URLs" from the menu. This is the URL that devices will use to communicate with PKIaaS.io to request a certificate from the template.

6.1.3 SCEP Validation

When a device requests a certificate via SCEP, PKIaaS.io will validate the request before issuing the certificate. Multiple validation methods are supported. See the links below for more information on each.

1. [Static Passphrase](#)
2. [Microsoft NDES Compatibility](#)
3. [Intune Integration](#)

6.2 Intune Integration

PKIaaS.io natively supports SCEP integration with Microsoft Intune allowing certificates to be validated and issued to devices managed by Intune.

6.2.1 Prerequisites

Before the Intune integration can be configured, communication must be authorized between PKIaaS.io and Intune. Follow Microsoft's documentation to configure the necessary permissions and create an application registration in Azure AD. <https://learn.microsoft.com/en-us/mem/intune/protect/certificate-authority-add-scep-overview>

After following the Microsoft documentation, it is required to also add the "Application.Read.All" permission to the application registration created in Azure AD in the previous step for "Azure Active Directory Graph". This permission is required to allow PKIaaS.io to query the graph API for the ScepRequestValidationFEService service principal endpoint. Follow the steps below:

1. Navigate to the azure app created in the first step
2. Go to API permissions
3. Click "Add a permission"
4. Select the "APIs my organization uses" tab
5. Search for "Windows"
6. Select "Windows Azure Active Directory"
7. Select "Application permissions"
8. Select "Application.Read.All"
9. Finally, grant admin consent to this permission

6.2.2 Enabling Intune Integration

To enable Intune integration, login to [PKIaaS.io](#), and navigate to **Certificate Templates -> Manage Templates**, click on the template you wish to enable Microsoft Intune integration for, then click "Edit Template" and select the "SCEP" tab. Check the "Enable Microsoft Intune/Endpoint Manager Integration" checkbox. A form will appear requiring the following information:

- **Tenant ID:** the tenant ID for your Azure tenant (refer to step 6 from the [Microsoft Intune documentation](#))
- **Application (client) ID:** the application (client) ID for the application registration created previously in Azure AD (refer to step 4 from the [Microsoft Intune documentation](#))
- **Client Secret:** the client secret for the application registration created previously in Azure AD (refer to step 5 from the [Microsoft Intune documentation](#))

Finally, click "Save Template" to enable the integration.

6.2.3 Create and assign SCEP certificate profiles in Intune

After the integration is enabled, the remainder of the configuration is done in Intune. Follow the steps in the <https://learn.microsoft.com/en-us/mem/intune/protect/certificates-profile-scep> to create and assign SCEP certificate profiles in Intune. Be sure to have the template SCEP URL handy while configuring the SCEP certificate profile. See [SCEP Overview](#) for help finding the URL for the template.

6.3 Microsoft NDES Compatibility

PKIaaS.io offers Microsoft NDES compatibility to support third-party applications and processes built to work with Microsoft NDES. The feature can be enabled on a per-template basis and can be used to generate one-time challenge passwords for certificate requests from the template. The challenge passwords expire after 60 minutes.

6.3.1 Enable Microsoft NDES Compatibility

To enable Microsoft NDES compatibility for a template, navigate to **Certificate Templates -> Manage Templates**, click on the template you wish to enable Microsoft NDES compatibility for, then click "Edit Template" and select the "SCEP" tab. Check the "Enable Microsoft SCEP Compatibility" checkbox and save the template.

6.3.2 Microsoft NDES Compatibility URL

After enabling Microsoft NDES compatibility, the URL to access the Microsoft compatible UI can be found by navigating to **Certificate Templates -> Manage Templates**, clicking on a template, and selecting "SCEP Admin (Microsoft Compatibility)" from the menu. This URL is unique to each template and can be used to generate one-time passcodes for certificate requests from the template.

6.3.3 Authentication

Access to the Microsoft NDES Compatibility URL is automatically authenticated using the same credentials used to log in to PKIaaS.io if there is an active session. In order to access the URL without an active session—this may be required by third-party applications designed to integrate with Microsoft NDES—basic authentication can be enabled by checking the "Enable Microsoft Basic Authentication" checkbox in the "SCEP" tab of the template configuration. Enter a username and password to use for basic authentication and save the template.

6.3.4 Restrict Access

To restrict access to the Microsoft NDES Compatibility URL to specific IP addresses, see [SCEP Security](#).

6.4 Apple Configuration Profile

PKIaaS.io supports a feature that allows Apple devices running iOS or MacOS to request a certificate via the SCEP protocol without the need for mobile device management. This works by creating an Apple configuration profile and emailing it to an Apple user to download and install on their device. The profile contains the SCEP URL and other configuration settings that the device uses to request a certificate from PKIaaS.io.

6.4.1 Prerequisites

SCEP must first be enabled on the template that will be used to issue certificates to Apple devices. Follow the steps in the [SCEP Overview](#) to enable SCEP on the template.

6.4.2 Create an Apple Configuration Profile

To create an Apple configuration profile to request a certificate via SCEP, login to [PKIaaS.io](#), and navigate to **Certificate Templates -> Manage Templates**, click on the template you wish to issue the certificate from, and click "Create Pre-Approved Request". Fill out the form following the below instructions:

1. Enter the desired subject alternative names, common names and other subject attributes
2. Choose the "Apple Device" option
3. Enter the email address of the Apple user that will receive the profile
4. Click "Submit"

6.4.3 Download and Install the Apple Configuration Profile

After submitting the form, an email will be sent to the Apple user with a link to download the configuration profile. The user can then download and install the profile on their device and the device will automatically request a certificate from PKIaaS.io using the SCEP protocol. See the Documentation from apple in the links below for more information on how to install the profile on an Apple device after it has been downloaded.

- [Install a configuration profile on iOS, iPadOS, and visionOS devices](#)
- [Install a configuration profile on macOS devices](#)

6.5 Static Passphrase

PKIaaS.io supports the use of a static passphrase as a validation method for SCEP certificate requests, however, using a static password is generally considered a security risk due to its potential for unauthorized access if compromised. For this reason, we recommend using a more secure validation method, such as Microsoft NDES Compatibility or Intune Integration, whenever possible.

6.5.1 Enable Static Passphrase

Within the SCEP tab of the template configuration, check the "Enable SCEP Static Passphrase" checkbox. Click the input that says "Click to Generate" to generate a passphrase. This passphrase will not be visible after the template is saved, so be sure to copy it to a secure location before saving.

Note: The passphrase can be updated in the future by clicking the "Click to Generate" input again. This will generate a new passphrase and invalidate the old one. If ever it is clicked by mistake, be sure to click "Cancel" in the template editor to keep the original passphrase.

6.6 SCEP Security

SCEP access can be restricted to specific IP addresses. To restrict access to a specific IP address, navigate to **Certificate Templates -> Manage Templates**, click on the template you wish to restrict access to and select the "SCEP" tab. Check the "Enable IP ACL" checkbox and enter the IP address you wish to allow SCEP requests from in the "Allowed IPs" field. Multiple IP addresses can be entered, separated by commas. Using the "Limit access to" drop down, you may choose whether to limit access to SCEP Requests, the Microsoft compatibility UI or both.

7. OCSP

7.1 Overview

OCSP (Online Certificate Status Protocol) is a network protocol used in Public Key Infrastructure (PKI) to check the revocation status of digital certificates. It allows clients, like web browsers, to verify whether a certificate is still valid or has been revoked without requiring a full download of a Certificate Revocation List (CRL).

7.2 OCSP Signing Certificate

By default, PKIaaS.io generates a certificate with the "OCSP Signing" extended key usage (EKU) attribute for each CA when the CA is created. The certificate will appear in **X509 Certificates -> Issued Certificates** with the common name "PKIaaS.io OCSP Responder". This certificate is used to sign OCSP responses for the CA. When the certificate expires or is manually revoked, a new OCSP signing certificate will be generated on demand during the next OCSP request.

7.3 OCSP Responder URL

To find the OCSP responder URL along with many other service URLs, login to PKIaaS.io, and navigate to **Certificate Authorities -> Manage CAs**. Click on the desired CA, and Select "Show CA Service URLs". A valid OCSP request must be sent to the OCSP responder URL to receive a signed OCSP response. Tools like the [OCSP Checker](#) can be used to return the OCSP response for a certificate.

7.4 OCSP in the AIA Extension

By default, the OCSP URL is also encoded in issued certificates in the Authority Information Access (AIA) extension. This can be changed within a certificate template by navigating to **Certificate Templates -> Manage Templates**, clicking on a template, and selecting "Edit Template". The OCSP URL can be removed from certificates issues using the template by unchecking "Include OCSP Extension in Certificate" in the **CRL / OCSP / AIA / CPS** section of the template settings.

8. CRL

8.1 Overview

A CRL (Certificate Revocation List) is a list published by a Certificate Authority (CA) that contains the serial numbers of digital certificates that have been revoked before their scheduled expiration. This list is part of Public Key Infrastructure (PKI) and serves as a mechanism to help entities verify whether a certificate is still valid.

8.2 CRL Generation

PKIaaS.io automatically generates CRLs on demand when a HTTP request is received to download the CRL. CRLs are cached and valid for a period of 30 days from the time of generation. When the CRL expires or another certificate is revoked, the cache is deleted, and a new CRL is generated on demand during the next HTTP request to download the CRL.

8.3 CRL URL

To find the CRL URL along with many other service URLs, login to PKIaaS.io, and navigate to **Certificate Authorities -> Manage CAs**. Click on the desired CA, and Select "Show CA Service URLs". Click the CRL URL to download the CRL.

8.4 CRL Distribution Point

By default, the CRL URL is also encoded in issued certificates in the CRL Distribution Points (CDP) extension. This can be changed within a certificate template by navigating to **Certificate Templates -> Manage Templates**, clicking on a template, and selecting "Edit Template". The CRL URL can be removed from certificates issues using the template by unchecking "Include CRL Extension in Certificate" in the **CRL / OCSP / AIA / CPS** section of the template settings.

9. Timestamping

9.1 Overview

Timestamping is the process of adding a trusted and verifiable time and date to a piece of digital data. This establishes proof that the data existed or was signed at a specific moment in time and has not been altered since. It ensures that signed data remains verifiable and trustworthy long after the initial signing event, even after the signing certificate has expired.

9.2 Timestamping Service

PKIaaS.io provides a timestamping service that allows users to request a trusted timestamp for code signing, PDF document signing, etc. Both the RFC 3161 and Authenticode timestamping protocols are supported. The timestamping service URL is unique to each CA and can be found by logging into [PKIaaS.io](https://www.pki-aaS.io) and navigating to **Certificate Authorities -> Manage CAs**. Click on the desired CA, and select "Show CA Service URLs".

9.3 More Information

See the links below for more information about various tools for timestamping documents and code.

- <https://www.digicert.com/kb/code-signing/digicert-certificate-utility-to-sign-code.htm>
- <https://globaltrust.eu/en/adobe-acrobat-reader-dc-use-certificate-timestamp-server/>
- <https://learn.microsoft.com/en-us/windows/win32/seccrypto/signtool>

10. Iot-HSM

10.1 Overview

See the [installation documentation](#) to get started with IoT-HSM.

10.1.1 Overview

IoT-HSM is a lightweight application designed to enable a persistent connection between one or more YubiKeys and PKIaaS.io. Once deployed and provisioned for a Certificate Authority (CA), it securely forwards all signing requests for the CA from PKIaaS.io to the IoT-HSM, where the connected YubiKey handles the signing. To ensure robust security, all communications with the IoT-HSM are digitally signed with end-to-end encryption via S/MIME.

█

10.2 Installation

To get started with IoT-HSM, it is recommend to deploy a Long-Term Support (LTS) version of [Ubuntu Server](#). IoT-HSM has been tested on Ubuntu 20.04, 22.04, and 24.04. Once Ubuntu is installed and updated, login to a terminal shell, and execute the following commands as the root user. You can access a root shell by entering `sudo su -` at the terminal.

```
apt update
apt -y install curl
curl -sL https://raw.githubusercontent.com/lspiebler/iot-hsm/master/scripts/setup_ubuntu2004.sh | bash -
```

After the script completes, it will be prompt to set a new password twice—this password will be used for the admin account on the IoT-HSM web interface. Once configured, open a browser and navigate to the IP address of the Ubuntu endpoint, prefixed with `https://`. For example: `https://192.168.1.5`.

See the [provisioning documentation](#) to provision a certificate authority for use with IoT-HSM.

10.3 Provisioning

If you haven't yet deployed IoT-HSM, see the [installation documentation](#) to get started.

Create a CA with an HSM Key on PKIaaS.io

In order for a certificate authority on PKIaaS.io to delegate signing operations to a YubiKey managed by IoT-HSM, it must first be provisioned to do so. The first step in this process is to select the "HSM" option as the "Key Type" when creating a new certificate authority on the **Certificate Authorities -> Create new CA** page.

Retrieve HSM Provisioning Certificate from PKIaaS.io

After clicking "Create New CA," the browser will navigate to a page showing the "HSM Provisioning Certificate." This will be used as a template to configure the new CA on one of the YubiKey slots.

Note: If at any time you need to navigate back to this page, go to **Certificate Authorities -> Manage CAs**, click on the CA and select "Download HSM Provisioning Certificate" from the menu.

Prepare the Private Key on the IoT-HSM

Login to the IoT-HSM. From the "HSM Options..." dropdown, choose "Provision a New Slot." This will navigate to the provisioning wizard. Choose the slot to provision and a key option. The "Generate" option (recommended) will generate a new private key on the YubiKey slot using the "Key Type" option provided. The "Import" option allows an existing private key to be imported to the YubiKey slot. This option should only be used when the private key has been handled with extreme care and is ideal under circumstances when it is important to have a backup of the private key or if multiple IoT-HSMs will be provisioned with the same private key for redundancy.

Import the HSM Provisioning Certificate to the IoT-HSM

SELF-SIGNED ROOT CA

After preparing the private key, the next page in the provisioning wizard is where the "HSM Provisioning Certificate" generated in a previous step will be submitted. If a self-signed root CA is being generated, leave the first option selected, paste the provisioning certificate into the text area, and click submit.

INTERMEDIATE CA

If this CA will be an intermediate CA, select the option to generate a CSR, paste the provisioning certificate into the text area, and click submit. A modal will appear with a CSR that will need to be signed by the root CA issuing the certificate for this intermediate. Finally, select the last option in the dropdown and paste the certificate signed by the root using the provided CSR and click submit.

Note: If at any time you need to navigate back to this page, go to the IoT-HSM home page, select the slot being provisioned from the "Configured Slots..." drop down, and click "Complete Provisioning."

Import the New Certificate from the IoT-HSM to PKIaaS.io

Navigate to the IoT-HSM home page. Select the newly provisioned slot from the "Configured Slots..." drop down and click "View Certificate." Copy the certificate to the clipboard and on PKIaaS.io, navigate to **Certificate Authorities -> Manage CAs**, click on the CA being provisioned, and select "Import Signed Certificate." Paste the certificate copied from the IoT-HSM into the text area and click "Import Certificate." A prompt will appear instructing to restart the IoT-HSM device. On the IoT-HSM home page, click "Reload" in the navigation bar at the top. Click "Import Certificate" again on the PKIaaS.io import signed CA page. If the imported certificate is an intermediate CA and PKIaaS.io is unable to download the certificate chain, a modal will appear prompting for the chain, including all intermediates and the root, to be supplied before the import can be completed.

Provisioning Complete

When the steps have all be completed successfully, all signing requests for the provisioned CA will be delegated to the YubiKey on the IoT-HSM. Signing operations will fail if the IoT-HSM is not online with the YubiKey inserted.

11. Post-Quantum Cryptography

11.1 Overview

Post-quantum cryptography (PQC) refers to cryptographic algorithms designed to remain secure against the potential threats posed by quantum computers. Quantum computers have the potential to break many of the cryptographic algorithms that are currently in use, including RSA and ECC. As a result, the National Institute of Standards and Technology (NIST) has been working to standardize post-quantum cryptographic algorithms.

On August 13, 2024, NIST finalized two digital signature standards that can be used to create quantum-safe X.509 certificates: ML-DSA, as defined in [FIPS 204](#), and SLH-DSA, as defined in [FIPS 205](#). PKIaaS.io currently supports ML-DSA. Support for SLH-DSA is planned for a future release.

- [ML-DSA](#)

11.2 ML-DSA

11.2.1 Create a ML-DSA Certificate Authority

Login to [PKIaaS.io](https://pkiaas.io), and navigate to **Certificate Authorities -> Create New CA**. From the **Key Type** dropdown, select "ML-DSA". In the **Parameter Set** drop down, "65" (security level 2) will be selected by default. A different parameter set may be selected if desired to customize the security level of the private key for the CA. See more information about ML-DSA security levels below. Refer to [Certificate Authorities Overview](#) for more information on the available options in the new CA form. After the form has been completed, click **Create New CA** to create the ML-DSA CA.

Security Levels (Parameter Sets)

ML-DSA offers parameter sets that meet three security levels: ML-DSA-44, ML-DSA-65, and ML-DSA-87. ML-DSA-44 is intended to meet NIST's level 2 security category, ML-DSA-65 is intended to meet level 3, and ML-DSA-87 is intended to meet level 5. See <https://www.ietf.org/archive/id/draft-salter-lamps-cms-ml-dsa-00.html> for more information.

- **Security Level 2 (ML-DSA-44)** - Attacks on algorithms in the level 2 category are intended to be at least as hard as performing a collision search on SHA256.
- **Security Level 3 (ML-DSA-65)** - Attacks on algorithms in the level 3 category are intended to be at least as hard as performing an exhaustive key search on AES192.
- **Security Level 5 (ML-DSA-87)** - Attacks on algorithms in the level 5 category are intended to be at least as hard as performing an exhaustive key search on AES256.

11.2.2 Generate ML-DSA Private Keys and CSRs

In order to issue a quantum-safe ML-DSA certificate, a CSR must be generated with a ML-DSA private key. This can be done using tools like OpenSSL with the liboqs provider or online at <https://certificatetools.com/> by selecting one of the "Generate ML-DSA-(44|65|87) Private Key (liboqs)" options from the **Private Key** dropdown.

Note: In production, it is recommended to generate the private key and CSR on a secure system that is not connected to the internet.

11.2.3 Issue a ML-DSA Certificate

Navigate to **Certificate Templates -> Manage Templates**, and click on the desired template. From the menu select **Submit CSR**. Paste the contents of a base64 certificate signing request (CSR) generated with a ML-DSA private key and click **Submit**. Navigate to **X509 Certificates -> Pending Requests** to review and approve the request. Click on the pending request to view the details, optionally override requested attributes or validity, and click **Sign** to issue the certificate. Navigate to **X509 Certificates -> Issued Certificates** to view and download the issued certificate.

12. Use Cases

12.1 Sample Use Case

Sample Use Case