Connecting to Thredd Guide

Version: 1.1

06 February 2026

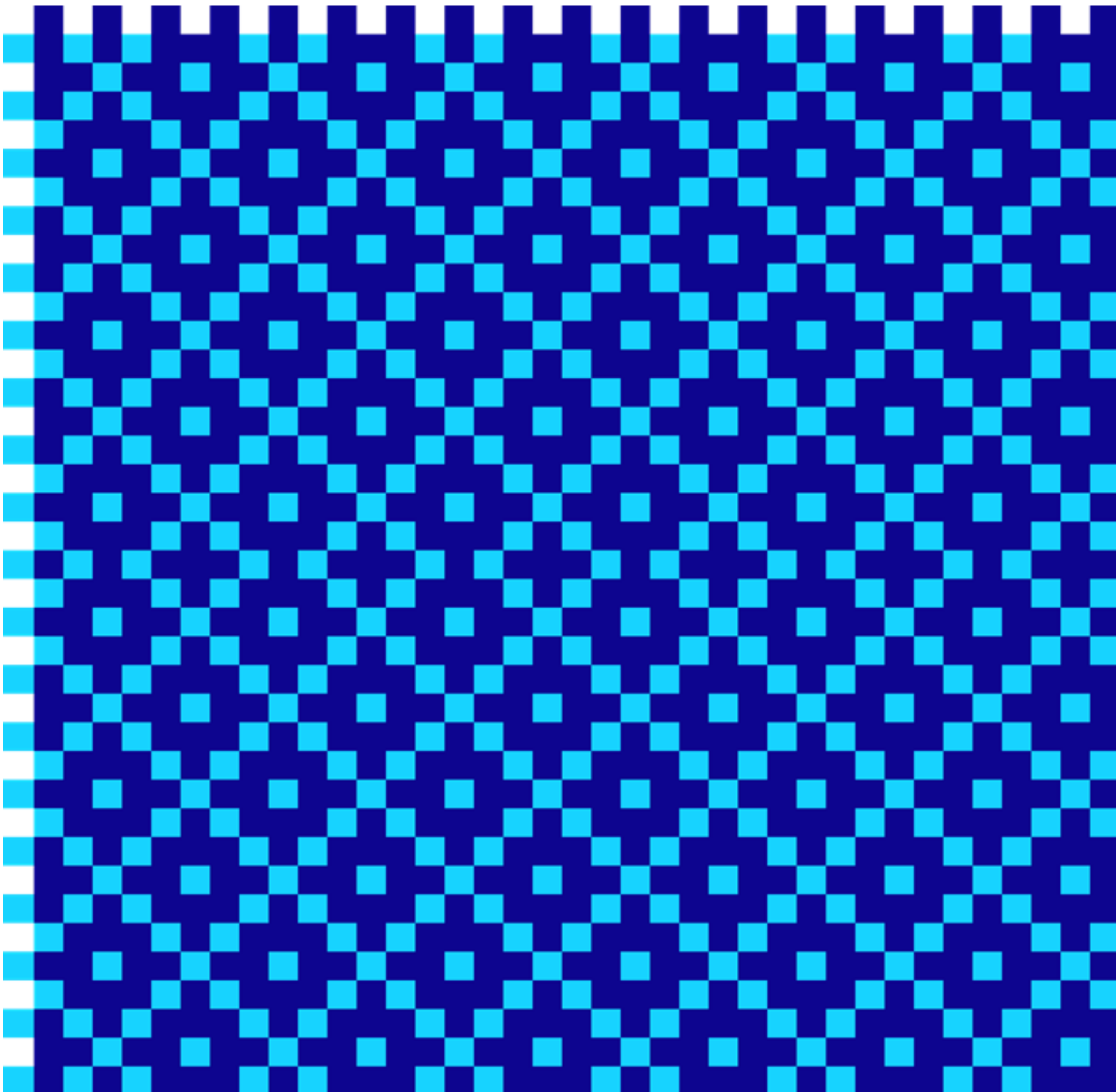
Publication number: CTG-1.1-2/6/2026

For the latest technical documentation, see the Documentation Portal.

Thredd, Kingsbourne House, 229-231 High Holborn, London, WC1V 7DA

Support Email: occ@thredd.com

Support Phone: +44 (0) 203 740 9682



Copyright

About this Guide

This guide is intended as a user guide, to provide information on connecting to Thredd services using AWS, VPN, and the Secure Connectivity Framework.

Target audience

This guide is aimed at developers and system integrators who need to set up secure connections to Thredd services. It provides information for security consultants and Chief Information Security Officers (CISOs) who need to assess Thredd's security infrastructure.

What's changed?

If you want to find out what's changed since the previous release, see the Document History section.

Terminology used in this Guide

Note: In this document, any terms in *italics* follow the standard (normative) terminology referenced in the OAuth 2.0 authorisation framework. For more information, see the IETF OAuth 2.0 Authorization Framework.

How to use this Guide

If you are new to Thredd and our security infrastructure, you should refer to the Secure Connectivity Framework and Setup Steps to understand about connecting to Thredd.

For the setup steps you need to perform, refer to the relevant sections.

Other Documentation

Refer to the table below for a list of other relevant documents that should be used together with this guide.

Document	Description
Key Concepts Guide	Provides an introduction to card payments and how describes how Thredd supports your card program.
Web Services Guide (SOAP)	Provides information on the available Thredd web services and fields in each web service.
Cards API	Explains how you can integrate with Thredd's Cards API which uses REST.
EHI Guide	Provides details of the Thredd External Host Interface (EHI).
Thredd Portal Guide	Explains how you can use the Thredd Portal guide for managing cards.
Smart Client Guide	Describes how to use the Thredd Smart Client to manage your account.
Card Transaction System Guide	Describes how to submit card test transactions in the UAT environment.

Tip: For the latest technical documentation, see the Documentation Portal.

Section 1: Getting Started

You should read this section if you want information on how to connect to our VPN and AWS options, or are new to the Secure Connectivity Framework and want to understand the basic principles.

Topics covered in this section:

- VPN and AWS
- Secure Connectivity Framework
- Setup Steps

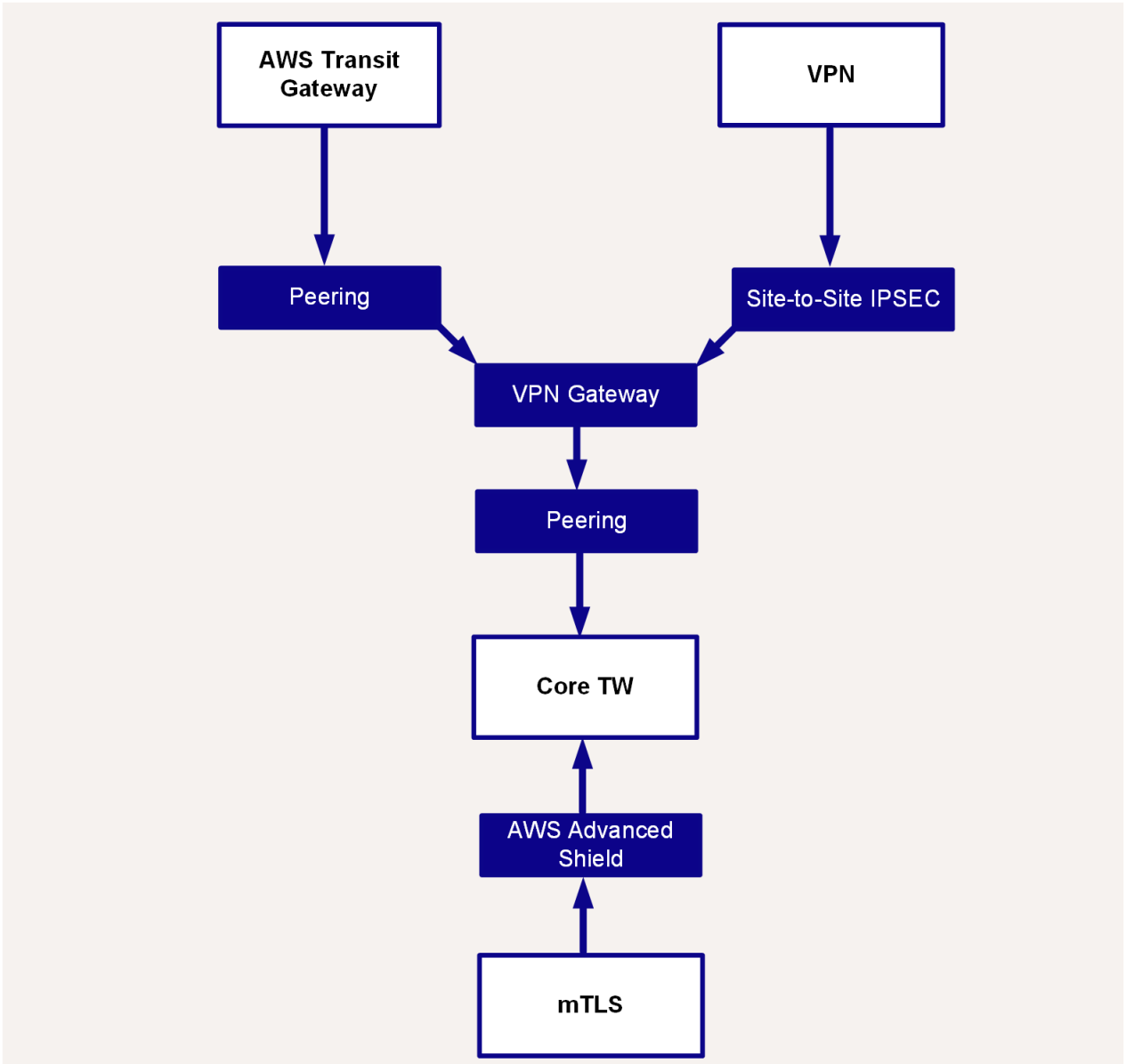
1.1 Connecting with AWS and VPN

This section details how to access the Thredd AWS environments. Thredd services are hosted in AWS. There are three supported integration options for clients:

- **IPSec Site-to-Site VPN** for clients using AWS and other Cloud and on-premise infrastructure.
- **AWS Transit Gateway Peering** for clients that host their Infrastructure in AWS.
- **Mutual TLS (mTLS)** for certificate-based authentication.

Note: For our VPN option you will be provided implementation documents which details how to connect to Thredd. For AWS, refer to the AWS documentation.

See the following diagram of the flow for each option.



Connecting using VPN

IMPORTANT: It is the responsibility of the client to ensure that any local changes are communicated to Thredd to maintain connectivity.

Internet Protocol Security (IPsec) VPN connections take place over public networks, but the data exchanged over the VPN is still private because it is encrypted. VPNs make it possible to securely access and exchange confidential data over shared network infrastructure. In this instance, the public Internet.

IPsec is a framework of open standards to ensure private and secure communications over Internet Protocol (IP) networks. Encapsulating Security Payload (ESP) and Authentication Header (AH) are the two IPsec security protocols used to provide these security services.

Thredd VPN Setup Information

Thredd enables a private connection from AWS for each client that sets up two VPNs per site. You must complete the Thredd VPN Connectivity Setup Form and share it with Thredd. When this has been received, Thredd sends over a configuration template for your VPN product.

Thredd typically only supports static routing for VPNs. However, in certain circumstances, Border Gateway Protocol (BGP) routing can be implemented. For AWS configurations, Thredd provides Transit Gateway (TGW) peering to allow you to connect directly to our environment and minimise latency. You must include the TGW ID when completing the Thredd VPN Connectivity Setup Form.

When a subnet has been provided, Thredd confirms that it is available. Note the following:

- An EHI endpoint must be a subnet.
- The maximum subnet is /22. Thredd recommends using /24.
- Subnet /20 is not permitted by Thredd.
- All services run through the same Thredd tunnel.

Where two VPNs are required for production access they must be split as follows:

- Web Services and EHI
- Thredd Portal

Connecting using AWS

IMPORTANT: It is the responsibility of the client to ensure that any local changes are communicated to Thredd to maintain connectivity.

AWS Transit Gateway Peeringis for clients that host their Infrastructure in Amazon Web Services (AWS). Internet Protocol Security (IPsec) is a framework of open standards to ensure private and secure communications over Internet Protocol (IP) networks. Encapsulating Security Payload (ESP) and Authentication Header (AH) are the two IPsec security protocols used to provide these security services.

IPsec VPN connections take place over public networks, but the data exchanged over the VPN is still private because it is encrypted. VPNs make it possible to securely access and exchange confidential data over shared network infrastructure. In this instance, the public Internet.

Thredd AWS Setup Information

Thredd enables a private connection from AWS for each client and sets up a transit gateway attachment per site. You must complete the Thredd VPN Connectivity Setup Form and share it with Thredd. When this has been received, Thredd sends over a configuration template for your VPN product.

Thredd only supports static routing for AWS Transit Gateway (TGW) attachments. When a subnet has been provided, Thredd confirms that it is available. Note the following:

- An EHI endpoint must be a subnet.
- The maximum subnet is /22. Thredd recommends using /24.
- Subnet /20 is not permitted by Thredd.
- All services run through the same Thredd tunnel.

Where two attachments are required for production access they must be split as follows:

- Web Services and EHI
- Thredd Portal

1.2 Secure Connectivity Framework

Thredd's Secure Connectivity Framework is the combination of several components which enables clients to access Thredd's resources securely over mTLS (Mutual Transport Level Security).

The main components are:

- Cloudentity
- Thredd CA
- mTLS Termination

Cloudentity

A Software as a Service (SaaS) capability which acts as the *Identity Provider (IDP)* for Thredd's interfaces (including Thredd CA and Thredd Portal) and as an *OAuth OpenID Provider (OP)* for the registration and management of customer applications, generation and validation of access tokens, and for the enforcement of access control policies.

The following figure illustrates Cloudentity and its relationship with other components.

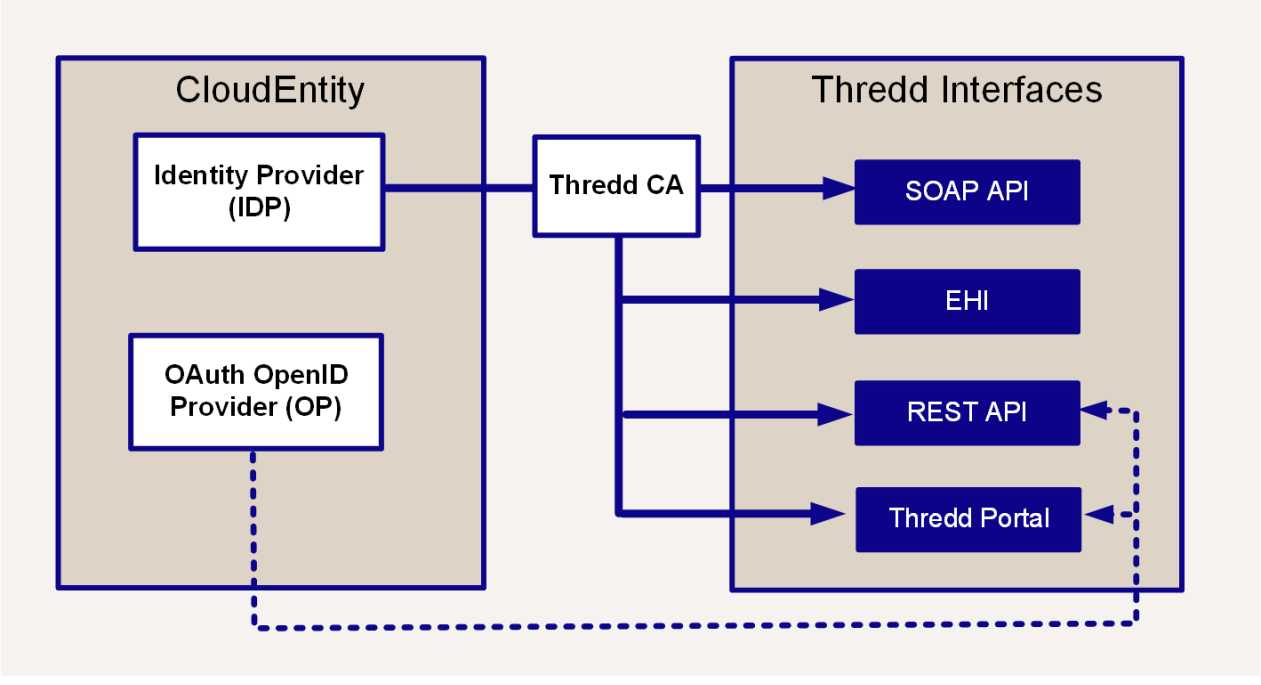


Figure 1: Secure Connectivity Framework Including Cloudentity, Thredd CA, and Various Thredd Services

Thredd CA

Thredd CA is Thredd's Certificate Authority for setting up and managing certificates to connect to various services. The certificates include:

- **Transport Certificates** – for establishing secure connections between resources.
- **Signing Certificates** – for the creation of signed messages, used for authentication of clients, and non-repudiation and authentication of notifications.

The following table shows the certificates that are needed for each Thredd application.

Thredd Application	Transport Certificate	Signing Certificate	Other Certificates	Cloud Entity
REST API	√	√	n/a	x
SOAP API	√	x	n/a	x
External Host Interface (EHI)	x (provided by Thredd)	x	Root and Issuing	x
Thredd Portal	x (pre-installed)	x	n/a	√
Smart Client	x (pre-installed)	x	n/a	x

Additional details

Thredd Application	Certificates Required
REST API	Transport Certificates and Signing Certificates.
SOAP	Transport Certificates.
External Host Interface (EHI)	Root and Issuing Certificates. These are used to verify Transport Certificates presented by Thredd.
Thredd Portal	Transport Certificates (pre-installed).
Smart Client	Transport Certificates (bundled in the installer).

mTLS Termination

mTLS Termination requires on-premise infrastructure for establishing *Trust Chains*. Trust Chains, which are used to prove that a certificate originates from a legitimate source, are established when clients present Thredd-issued Transport Certificates for connecting to protected resources. This is used exclusively in EHI.

1.3 Setup Steps

This page describes the setup steps for each of the services.

1.3.1 Before you Begin

You need to have been set up on Cloudentity and have obtained access to Thredd Certificate Authority. You can then follow the steps for connecting to individual Thredd services.

Setting Up SSO Using Your Provider

The Secure Connectivity Framework allows you to set up Single Sign-On (SSO) to access various Thredd services that use mTLS, for example Thredd Portal. This not mandatory but is recommended.

SSO allows:

- An enhanced user experience for users as it removes the hassle of remembering passwords.
- Companies to save time on maintenance.
- Reductions in overheads when managing accounts.

For more details, see Configuring SSO.

Set Up Cloudentity

- Thredd sets up Cloudentity for you to enable a Single Sign On journey by linking your IdP with Cloudentity. If you do not use an IdP, Cloudentity can act as the IdP.
- A Single Sign On journey is used to access Thredd Certificate Authority (CA) for the creation of certificates, as well when connecting to the Thredd Portal card management application. In both cases, there is at least one additional Admin user, who manages users. Once set up, your organisation is unlikely to need to engage with Thredd for integrating Cloudentity.
- Cloudentity is also used behind-the-scenes for managing access to the REST API as an *Authorisation Server*.

Set Up Thredd Certificate Authority (CA)

Thredd will provide access to the Thredd CA. Thredd adopts a self-service approach, which allows you to independently manage your certificates.

To request access to Thredd CA, please raise a support ticket.

1.3.2 Steps for Individual Thredd Applications

Secure connections are required to the following Thredd applications:

- SOAP API
- REST API
- External Host Interface (EHI)
- Thredd Portal
- Smart Client

SOAP API

Thredd's SOAP APIs are secured using mTLS. You will need to create Transport Certificates.

For more information, see Creating Client Transport Certificates for SOAP APIs.

REST API

Thredd's REST APIs are secured using mTLS. You should review the following information on how to set up your MTLS connection:

- If you are using Postman to test the Thredd REST APIs for your client application, you can configure Postman to use mTLS. Follow the steps for using Postman as provided on the Cards API Website: Authentication the Cards API with mTLS. You can also view the documentation from the latest Postman collection on the Cards API Website: Using Postman.
- For background details on the Communication Flow between Postman, Thredd Certificate Authority, and Cloudentity, refer to the Communication Flow for Connecting to the REST APIs.
- Before using Postman, you will need to generate Transport Certificates for your client application; see Client Application Certificates for REST APIs. You will also need to generate an obtain an Software Statement Assertion (SSA), which you use to connect to the REST APIs; see Generating and Obtaining a Software Statement Assertion (SSA).
- To help in using the steps in Postman for setting up access to the REST APIs, you should follow the guidance in Using DCR Endpoint Data.
- Postman allows you to generate and obtain SSAs. However, you can generate and obtain SSA using the Thredd CA interface and cURL; see Generating and Obtaining a Software Statement Assertion (SSA).

EHI

Follow the steps below for connecting to EHI:

1. Install Server and Client Certificates.
2. Download Root Certificates and Issuing Certificates. A Root Certificate identifies the Certificate Authority. An Issuing Certificate identifies the system's identity, for example, its public key.
3. Test Client and Server Certificates on your EHI endpoint for mTLS communication.

For more information, see Setting Up EHI with mTLS.

Thredd Portal

You will need to be set up with Cloudentity, enabling authentication using your own Identity Provider (IdP). If you do not use an IdP, Cloudentity can act as the IdP.

For more information, see Connecting to Thredd Portal.

Smart Client and the Card Transaction System (CTS)

- Smart Client Installation: run the Smart Client installer. The installer is bundled with Transport Certificates, which ensure that users in your organisation can connect over mTLS.
- CTS access: CTS can be accessed online. CTS users can use the same credentials that they use to access Smart Client in UAT (provided that CTS has been enabled).

For more information, see Connecting to Smart Client and CTS.

Configuring SSO

Follow these steps for configuring SSO:

- Configuring SSO with Okta (SAML)
- Configuring SSO with Google (SAML)
- Configuring SSO with Okta (OIDC)

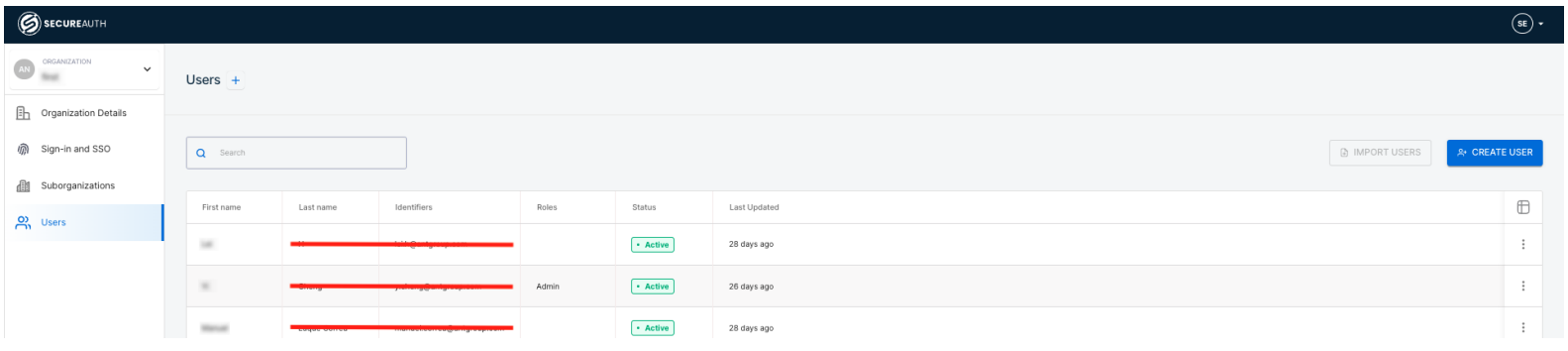
Other Services

Other services, including those for Fraud Transaction Monitoring and 3D-Secure, do not require you to set up secure connections via the Secure Connectivity Framework.

1.4 Manually Create Users

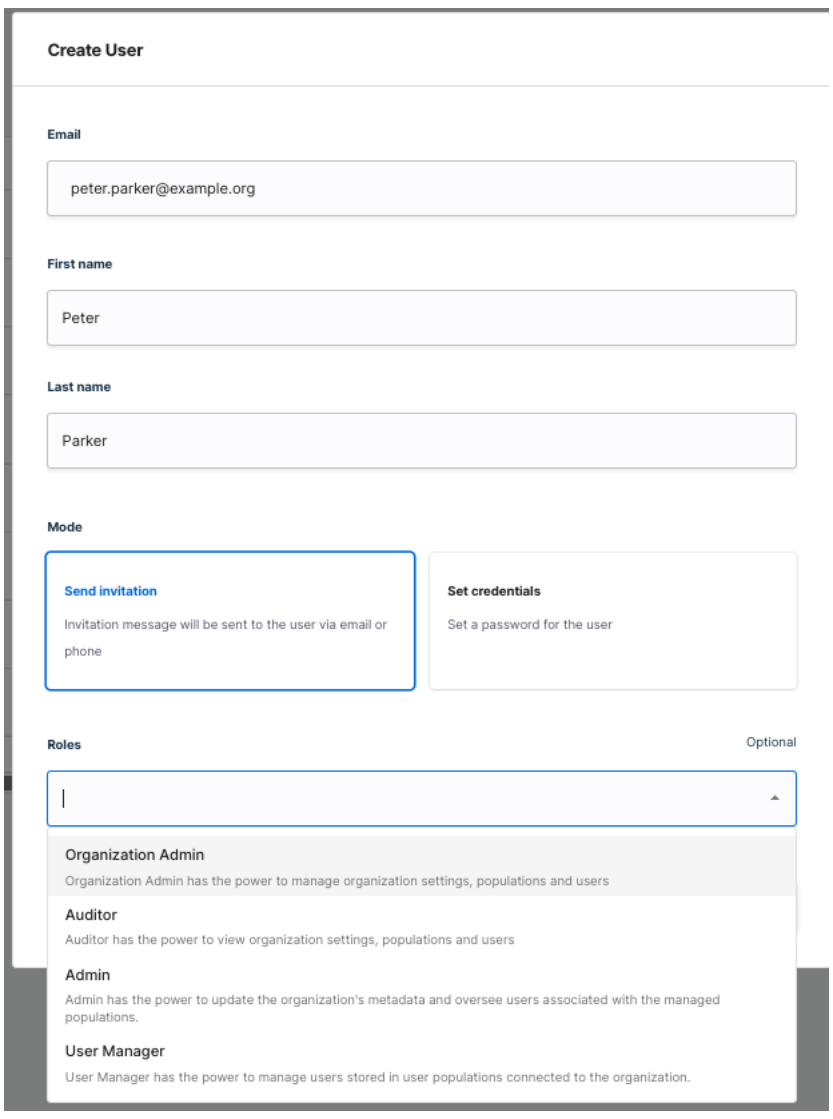
Users are created automatically using Single Sign-On after they first log in to any of the applications connected (for example, Raidiam Connect or Thredd Portal). For customers that do not have their own SSO compliant solution (such as Entra, Google, Okta), users must be added manually. This should be done by the customer user manager, which is usually the first administrator added for an organisation.

The customer user manager can add new users from the **Users** page in Cloudentity, which displays a list of their users and their current status.



To create a new user:

1. Click **Create User**. The **Create User** window opens.
2. Enter the details of the new user in the fields provided, ensuring that **Mode** is set to **Send Invitation**.



3. Select the role for the user from the **Roles** field. The roles that display depend on the role the administrator has.
4. Click **Save**.

The user will receive an email to activate their account, where they can then set a password. When this step has been completed, the user can log in to Thredd applications.

Assigning Restricted Codes to a User

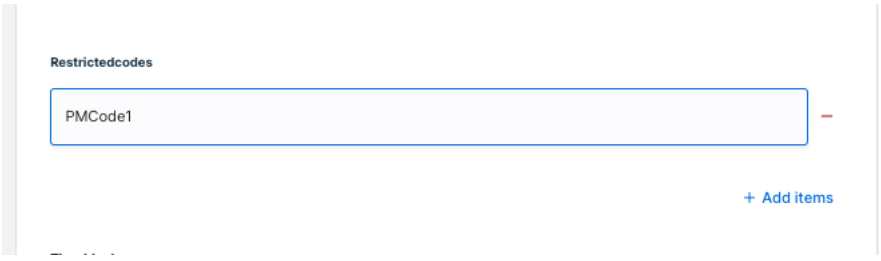
Some organisations have more than one Program Manager Code (pmcode). This code is used by Thredd Portal to control what data a user can access.

By default, every user in an organisation has access to all program manager codes assigned to that organisation. Administrators can restrict access to certain codes for specific users by adding to the user profile one or more restricted codes.

To add a restricted code to a user:

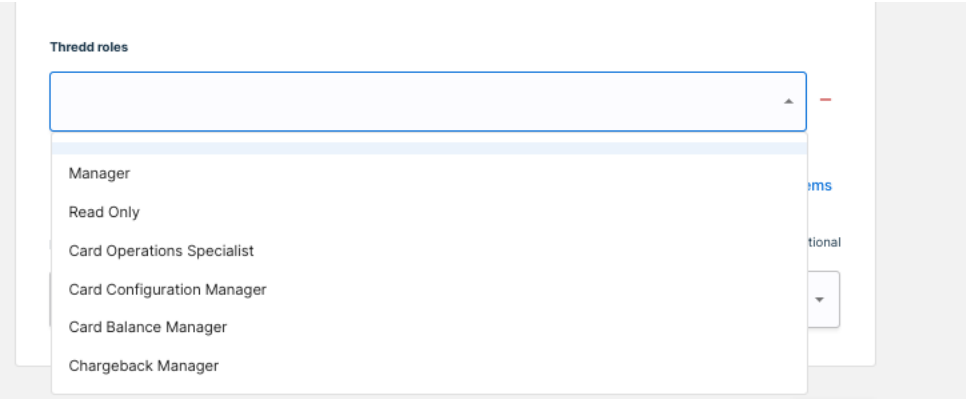
1. Open their User Profile.
2. Click **Add Items** in **Restrictedcodes**.
3. Enter the name of the pmcode you want to restrict for the user in the field provided.

In the example below, the user is restricted from using PMCode1.



Assigning Thredd Portal Roles to a User

You can add Thredd Portal roles to a user from their User Profile. Click the **Thredd roles** field to display the different roles available.



The following table describes what each role allows a user to do in Thredd Portal.

Thredd Role	Description
Manager	• Transaction Search & View • Remove auth • Card Search & View • Balance adjustment • Card Load/Unload • Change card status • PIN & CVC2 services • Edit cardholder details • Edit card configurations • Extend Thredd Expiry Date • Activate a Card • Balance Transfer
Read Only	• Card Search & View • Transaction Search & View
Card Operations Specialist	• Card Search & View • Transaction Search & View • Change card status • Activate a Card • Extend Thredd Expiry Date • PIN & CVC2 services • Remove auth
Card Configuration Manager	• Card Search & View • Transaction Search & View • Edit card configurations
Card Balance Manager	• Card Search & View • Transaction Search & View • Balance Transfer • Balance adjustment
Sales	• All actions available

Section 2: SOAP

You should read this section to understand setting up certificates for SOAP web services.

Topics covered in this section:

- Creating Organisation Transport Certificates for SOAP Web Services

2.1 Creating Organisation Transport Certificates for SOAP Web Services

This page describes how you create Transport Certificates for accessing Thredd's SOAP Web Services. As Thredd's SOAP Web Services are secured using Mutual Transport Layer Security (MTLS), your *client application* must present a trusted Transport Certificate for authentication. The steps described on this page for creating a certificate include those through the Thredd CA dashboard, and also via the command line interface.

2.1.1 Summary of Steps

The steps for obtaining a certificate are as follows:

1. Log in to Thredd CA.
2. Prepare the new certificate.
3. Set up the Certificate Signing Request (CSR) for the certificate.
4. Complete creation of the certificate.
5. Convert the certificate and key to the Public Key Cryptography Standard (PKCS#12) Syntax (for Windows only).

A CSR is a file that needs to be submitted to the Certificate Authority (CA) for generating a valid certificate.

Certificate Authority (CA)

A CA is responsible for generating and signing certificates, where Thredd uses its own CA.

Note: The following procedure ensures that you create a Transport Certificate from Thredd CA only. You must use certificates from this CA, as self-signed or third-party certificates result in connections being refused.

Storing Generated keys

After generating the private key and CSR on your systems, you must take steps to ensure keys are managed and stored securely.

2.1.2 Prerequisites

The following are prerequisites for creating Transport and Signing Certificates:

- There must be an account on Thredd CA, the application that is required for creating certificates.
- You must have installed OpenSSL on your machine for creating the CSRs.

To access Thredd CA, you must first complete Thredd's onboarding process. For details, contact your Implementation Manager or Account Manager.

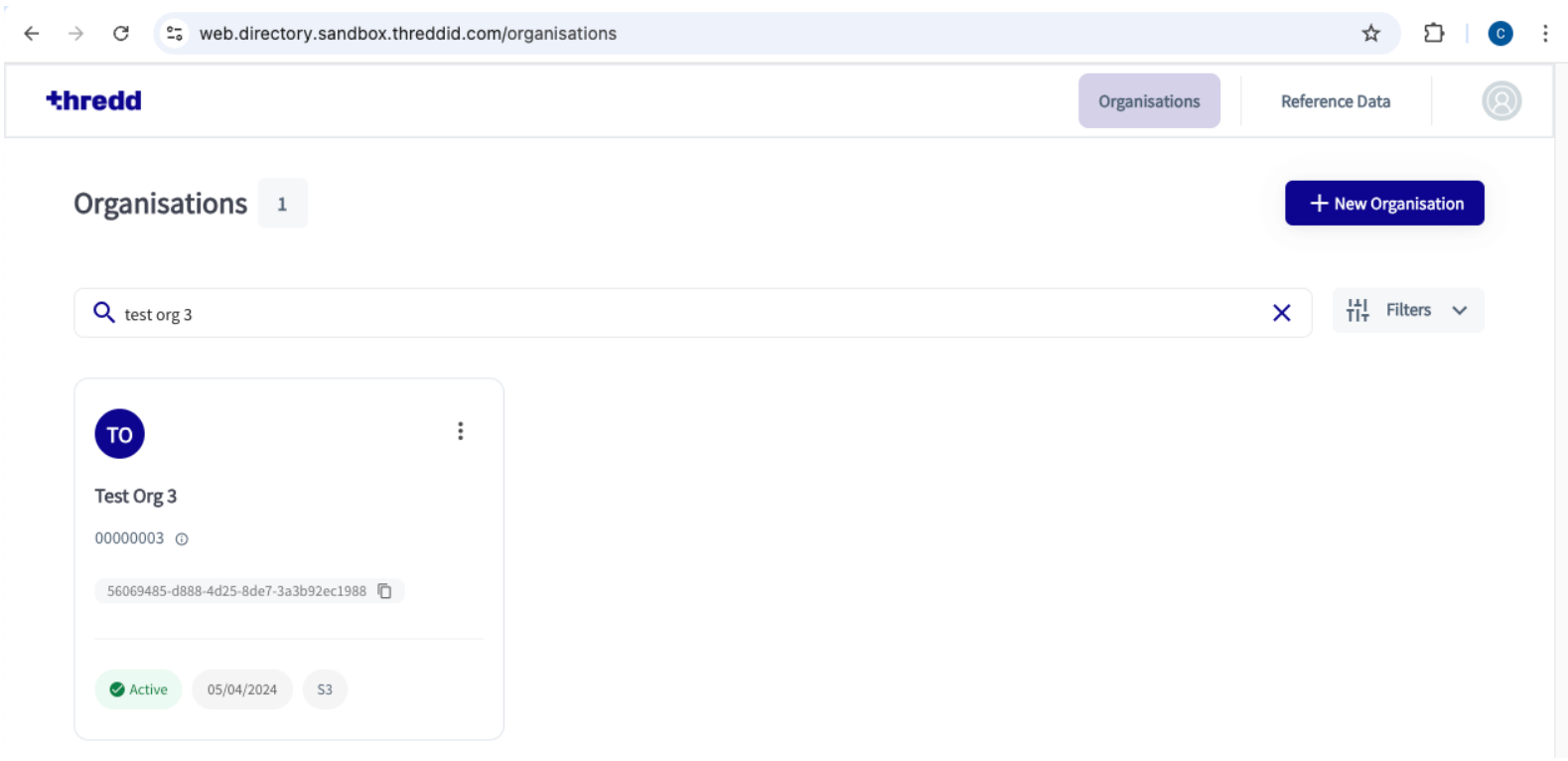
2.1.3 Steps in Setting Up a Certificate

Note that the screenshots below for Private Keys and CSRs contain example data only.

Step 1. Log in to Thredd CA

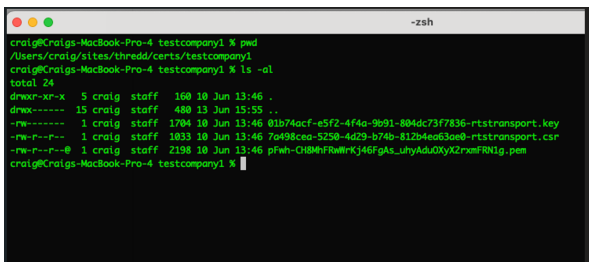
1. Log in to Thredd CA using these links:
    - Sandbox: https://web.directory.sandbox.threddid.com/
    - Production: https://web.directory.threddid.com/
2. In the Sign-in screen, enter your registered email address and click **Cloudentity SSO**.

3. Follow any other screen prompts or steps provided for your organisation. The Thredd CA dashboard appears with a list of organisations. An organisation appears as a tile as in the following example.

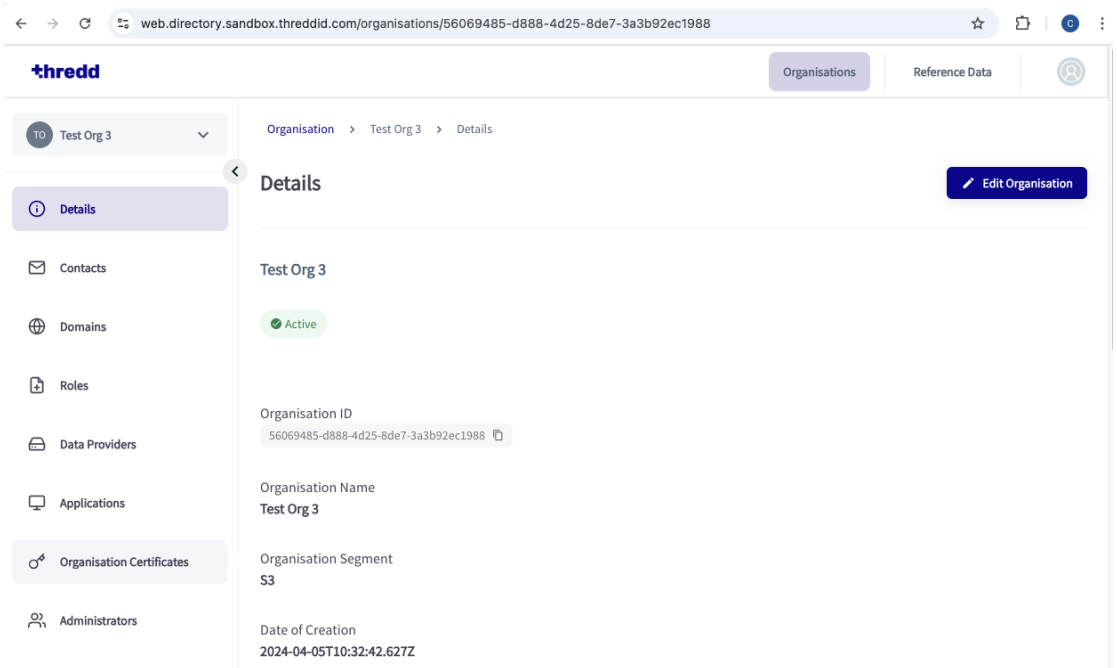


Step 2. Prepare the New Certificate

1. In a new terminal window, create an empty directory for making and receiving the private key and CSR. The following example shows a key pair and CSR that have been created in a directory.



2. In the Thredd CA dashboard, click on your organisation's tile.



3. In the Thredd CA dashboard, select **Organisation Certificates**.
4. Click **New Certificate**.
5. In the **New Certificate** window, select **RESOURCE SERVER TRANSPORT** in the **Select Certificate Type** dropdown list.

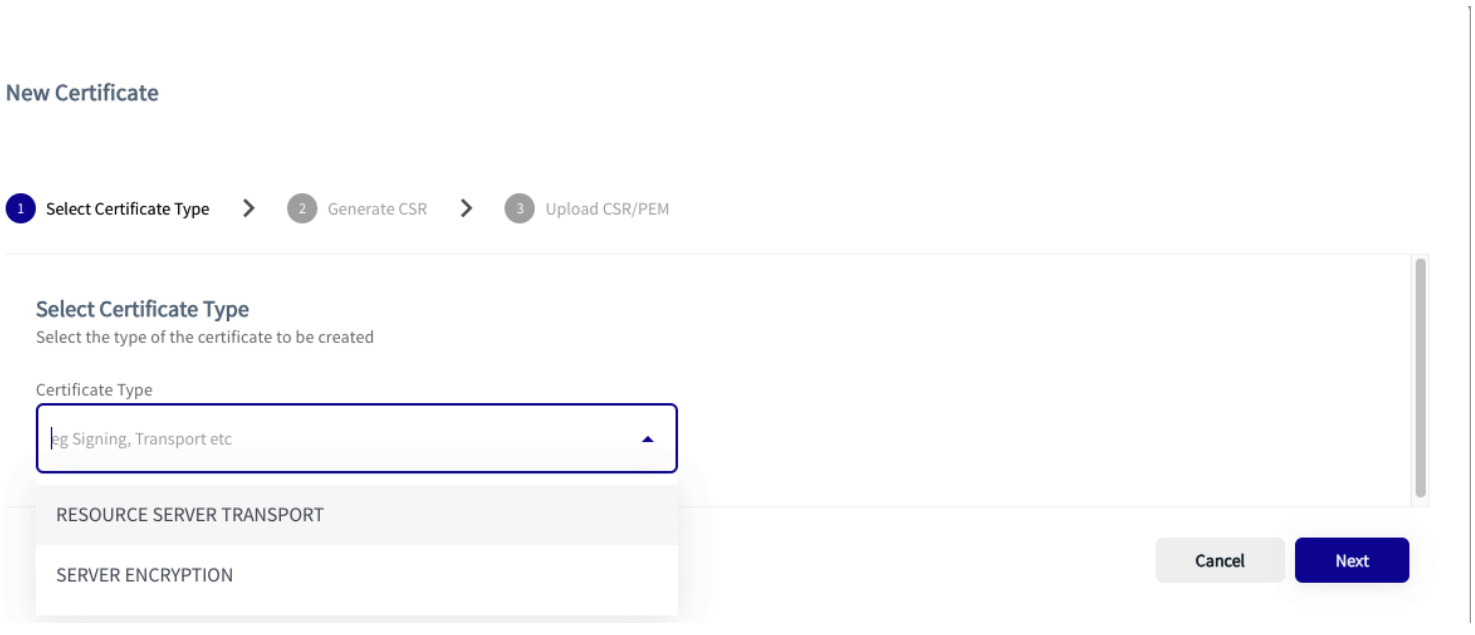


Figure 2: New Certificate Dialog box

6. Click **Next**.

Step 3: Set Up the CSR for the Certificate

1. Copy the generated CSR command and paste this into a terminal window.

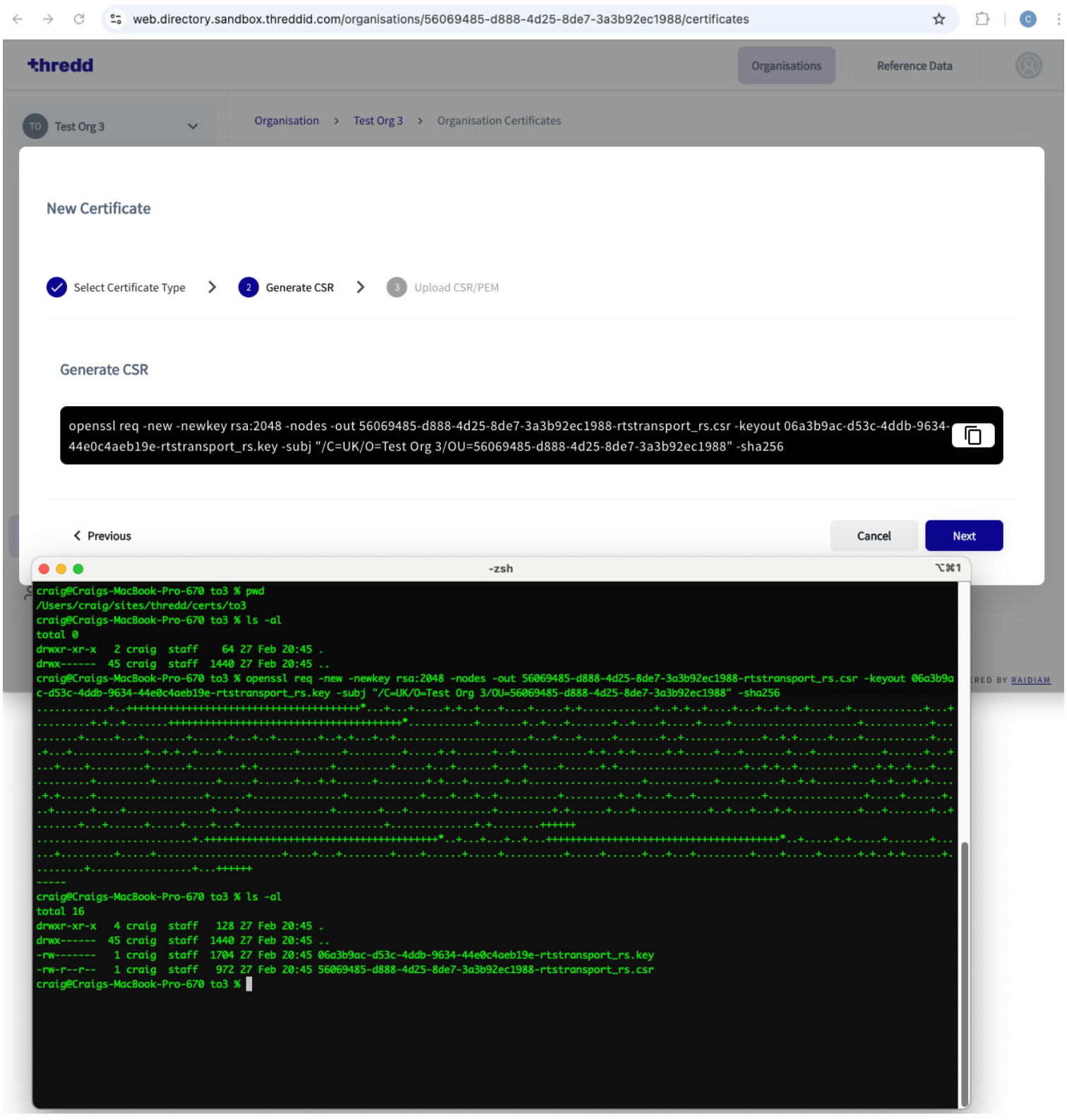
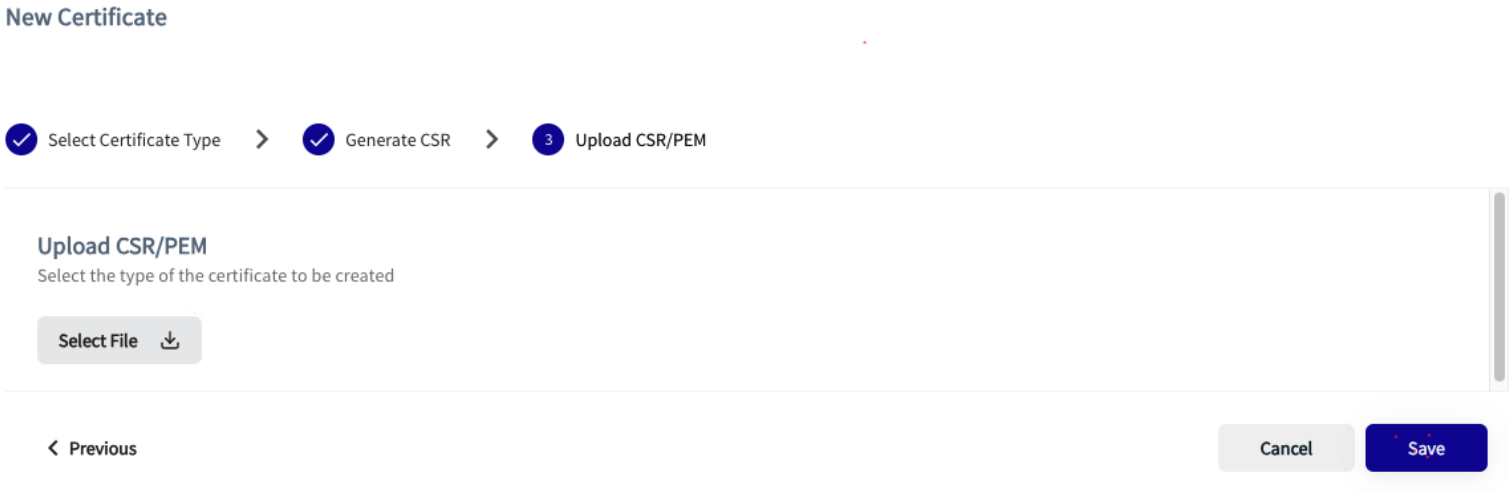


Figure 3: Terminal in the foreground and the New Certificates dialog box in the background

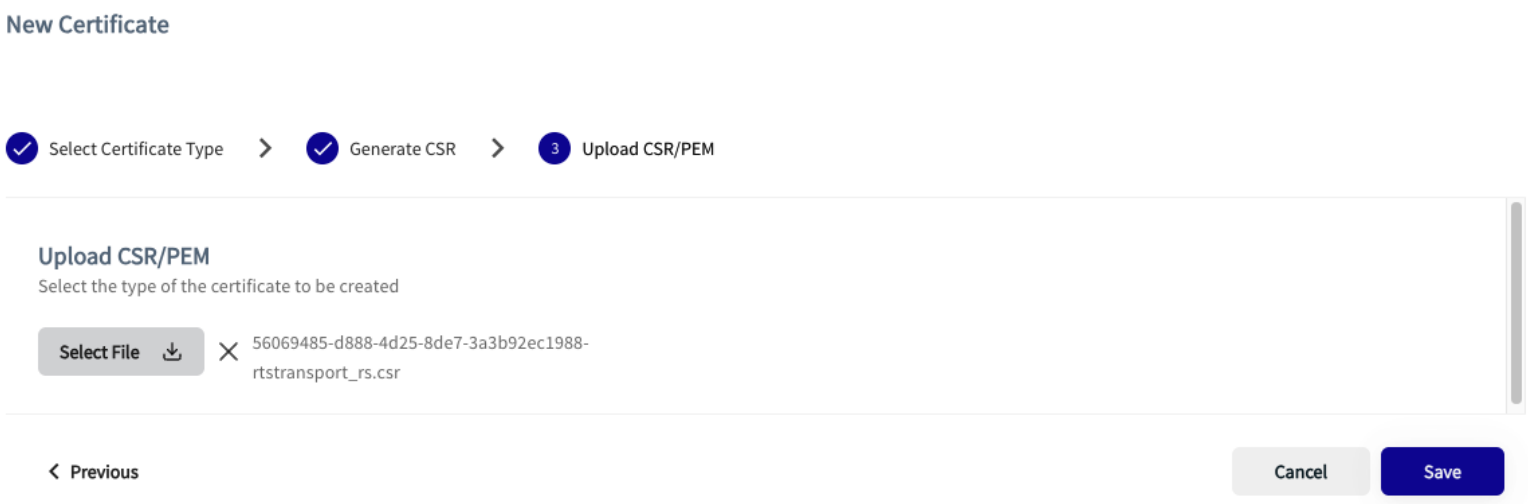
2. Click **Next**.
3. In the terminal window, press Enter to run the OpenSSL command. A Private Key and CSR generates in the background.
4. In Thredd CA, click **Select File**.



5. Browse to the CSR you just created. Then click **Open**.

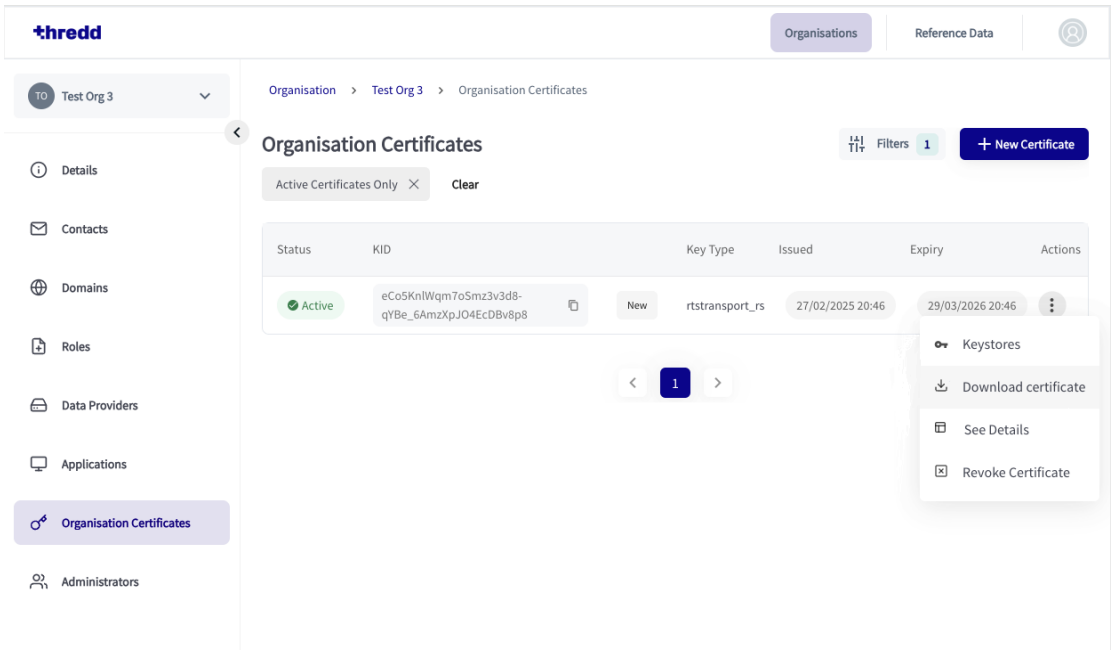
Step 4: Complete the Creation of Certificates

1. In the Thredd CA dashboard, click **Save** for the CSR you selected. This allows you to upload the CSR.



When uploaded, the certificate signing happens. This takes place in a few seconds. The newly created Transport Certificate then appears, and is ready for download.

2. On the three dots menu for the generated certificate, select **Download certificate**.



3. Click the download link on the CA UI to download the base 64 encoded X.509 certificate that was just generated.

When you've completed this process, a JSON Web Key Set (JWKS) endpoint is also created with public certificate details. JWKS is a JSON notation for sharing public keys which are used to verify the signature of a signed JSON web token (JWT).

Figure 4: JWKS Certificate

Step 5: Convert the Certificate and Key to PKCS#12 Syntax

If required, for example you are using Windows services, you can convert the public certificate and private key to the PKCS#12 syntax. For converting, you can use OpenSSL commands. In the PKCS#12 syntax, the files are in the .pfx format. For more details, see RFC 7292: PKCS #12: Personal Information Exchange Syntax v1.1.

Section 3: REST Applications

You should read this section to understand the setup steps for REST applications.

Topics covered in this section:

- Communication Flow
- Setting Up SSA
- Using DCR Endpoint Data

3.1 Communication Flow for Connecting to the REST APIs

This page describes the communication flow for accessing the Thredd REST APIs. There are two main areas in which you will need to set up your connection:

1. **Thredd Certificate Authority (CA) dashboard**. Log in to Thredd CA and create Transport and Signing Certificates. See Creating Client Application Certificates for REST APIs.
2. **A REST tool**. Configure your REST tool interface, for example Postman, to interact with Thredd CA and Cloudentity, so that you can access the API endpoints.

3.1.1 Individual Steps and Details

The following are the steps in the communication flow, where some are performed in Postman while others are done on Thredd Certificate Authority. The Thredd CA interface and Postman are both referred to here as a Data Consumer, where they are a customer of Thredd.

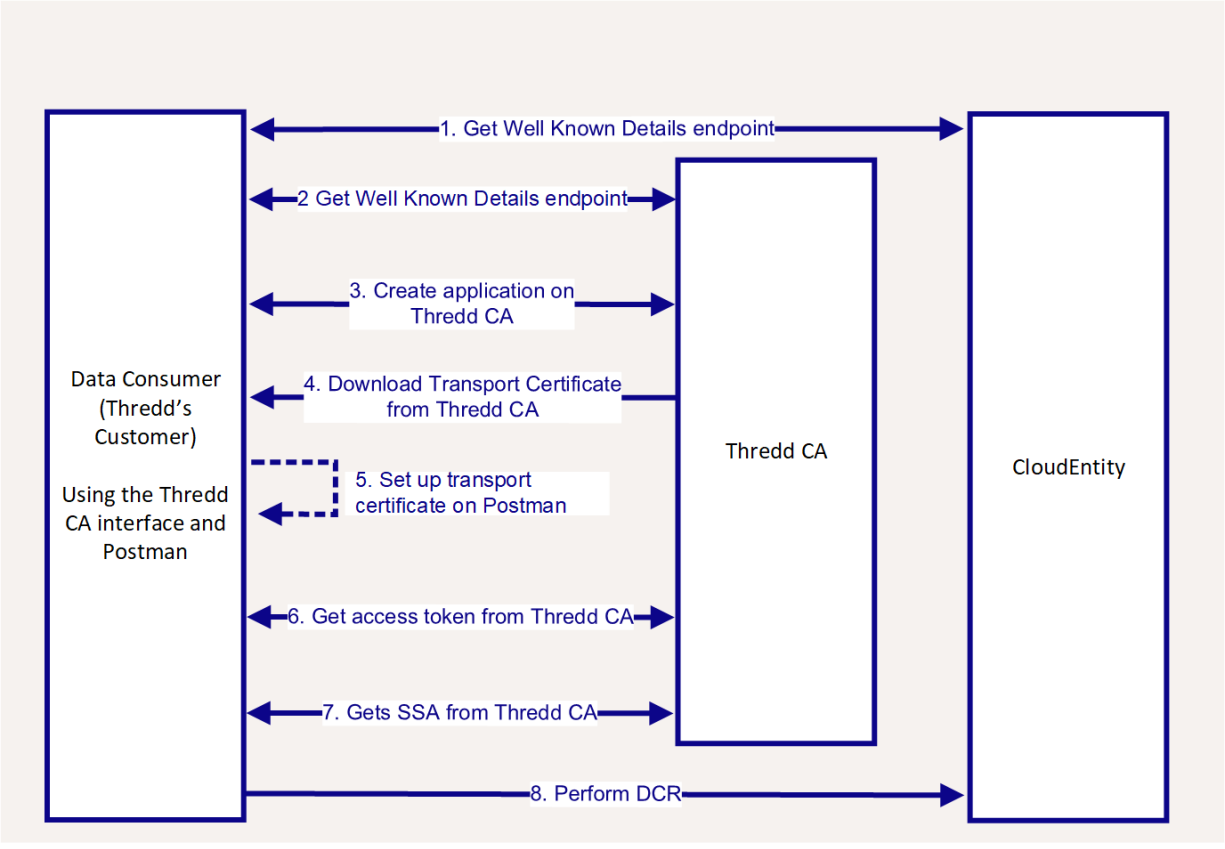


Figure 5: Individual Steps on Client Interactions with Thredd CA and Cloudentity

1. **Postman:** Gets Well Known endpoint from Cloudentity, which the client later uses to perform Dynamic Client Registration (DCR). A Well Known authorization server metadata endpoint provides a standardised way for clients to discover the necessary information to interact with an OAuth 2.0 or OpenID Connect server. You will need to set up the Get Well Known endpoint in Postman.
2. **Postman:** Gets Well Known Details endpoint which the client later uses to perform Dynamic Client Registration (DCR) from Thredd Certificate Authority. You will need to set this up in Postman.
3. **Thredd Certificate Authority:** Creates an Application on Thredd Certificate Authority.
4. **Thredd Certificate Authority:** Downloads the Transport Certificate from the user interface. You first create the Certificate Signing Request (CSR) and the Private Key, which Thredd CA then uses to generate the Transport Certificate. Fore more details on the steps for generating a certificate, see Creating OAuth 2.0 Client Application REST Transport Certificates.
5. **Postman:** Sets up a Transport Certificate on Postman that was created in Thredd Certificate Authority. The Transport Certificate is used to connect to the APIs.
6. **Postman:** Gets an access token from Thredd CA (Postman).
7. **Postman and Thredd Certificate Authority:** Get the Software Statement Assertion (SSA) from Thredd Certificate Authority. An SSA is a JSON web token for identity validation that is needed for Dynamic Client Registration. You will need to have created the SSA; see Generating and Obtaining an SSA.
8. **Postman:** Perform Dynamic Client Registration (DCR) by registering the OAuth Client on Cloudentity.

About DCR

DCR is a protocol that allows OAuth 2.0 and OpenID Connect clients (or client applications) to automatically register with an *Authorisation Server*, in this case Cloudentity. This is the final step that is required before accessing the REST APIs.

The following RFCs contain the definition of DCR:

- RFC 7591: OAuth 2.0 Dynamic Client Registration Protocol
- RFC 7592: OAuth 2.0 Dynamic Client Registration Management Protocol

Using Postman

For detailed steps on accessing the REST APIs through Postman, refer to the Cards API Website: Using Postman.

3.2 Creating Client Application Certificates for REST APIs

This page describes how you create Transport and Signing Certificates. Creating these certificates is an essential step for enabling OAuth 2.0 client applications that connect to Thredd's REST APIs. As Servers are secured using mTLS, your *client application* must present a trusted Transport Certificate when connecting to either the *Authorisation Server* or the *Resource Server* (which hosts the Thredd APIs). The steps described on this page for creating certificates include those through the Thredd CA dashboard, and also via the command line interface.

Summary of Steps

The steps for obtaining a certificate are as follows:

1. Log in to Thredd CA for your organisation. You will have access to at least one Organisation for your parent company.
2. Register a client application with Thredd's CA. This ensures that the CA recognises the client application before generating certificates.
3. Create a new Transport Certificate.

You will need to create a CSR (Certificate Signing Request) and a private key. A CSR is a file that needs to be submitted to the Certificate Authority (CA) for generating a valid certificate. Note that all screenshots for private keys and CSRs on this page show example data only.

Note: You need to create certificates before registering an application with the Authorisation Server. Registering an application with the Authorisation Server is part of Dynamic Client Registration. For details, on Dynamic Client Registration, refer to Dynamic Client Registration.

Certificate Types for the CA

This CA is responsible for generating and signing the following certificate types:

- Transport Certificates – used for client and server authentication.
- Signing Certificates – used for proving identity.

Note: The following procedure ensures that you create a Transport Certificate and Signing Certificate from the Thredd CA only. You must use certificates from this CA, as self-signed or third-party certificates result in connections being refused.

Storing Generated keys

After generating the private key and CSR on your system, you must take steps to ensure that the keys are managed and stored securely.

3.2.1 Prerequisites

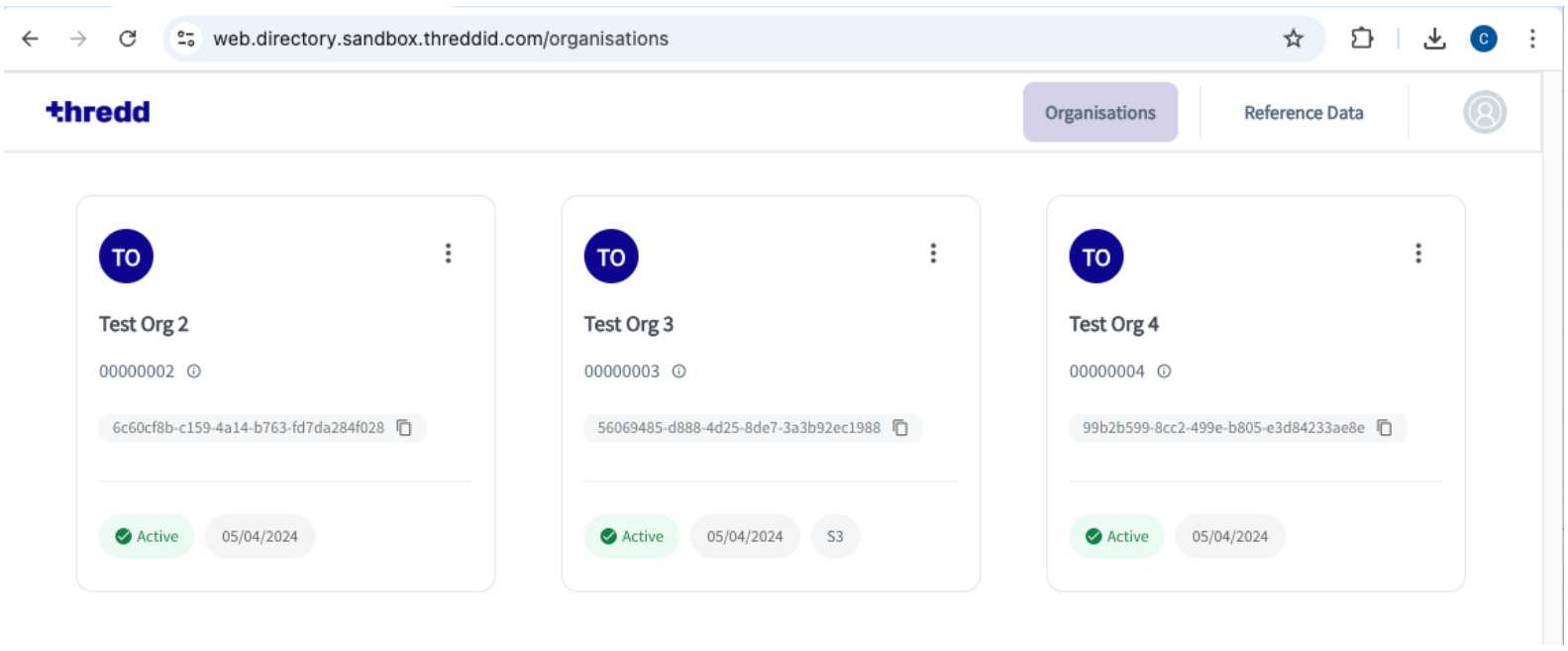
The following are prerequisites for creating Transport and Signing Certificates:

- There must be an account on Thredd CA, the application that is required for creating certificates.
- You must have installed OpenSSL on your machine for creating the CSRs.

To access Thredd CA, you must first complete Thredd's onboarding process. For details, contact your Implementation Manager.

3.2.2 Step 1. Log In to Thredd CA

1. Log in to Thredd CA using the links below:
    - Sandbox: https://web.directory.sandbox.threddid.com/
    - Production: https://web.directory.threddid.com/
2. In the Sign-in screen, enter your registered email address and click **Cloudentity SSO**.
3. Follow any other screen prompts or steps provided for your organisation. The Thredd CA dashboard appears with a list of organisations.



3.2.3 Step 2. Register a Client Application

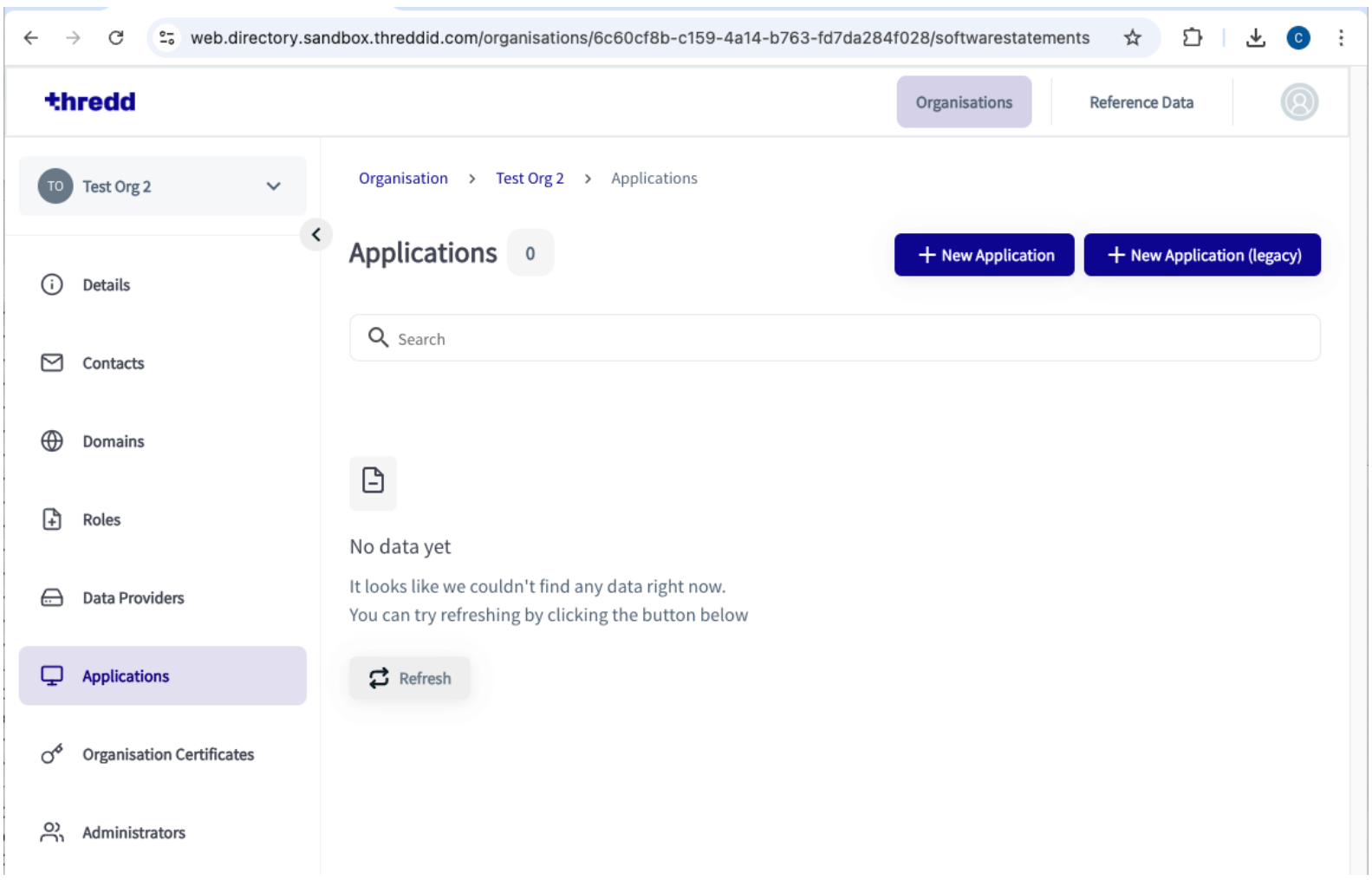
Before creating either a Transport Certificate or a Signing Certificate, you must register a *client application* with Thredd's CA. Using Thredd CA, you provide metadata associated with the application. The steps for registration are as follows:

1. Create the new client application.
2. Select roles associated with your client application.
3. Enter client details, which is any metadata associated with the client.
4. Enter user authentication settings.
5. Type in additional details

Note: The step for registering a client application does not register the application with the *Authorisation Server*. For information on how to register with an *Authorisation Server*, see Dynamic Client Registration.

Create a New Client Application

1. Select your organisation and then select **Applications** from the left-hand menu.

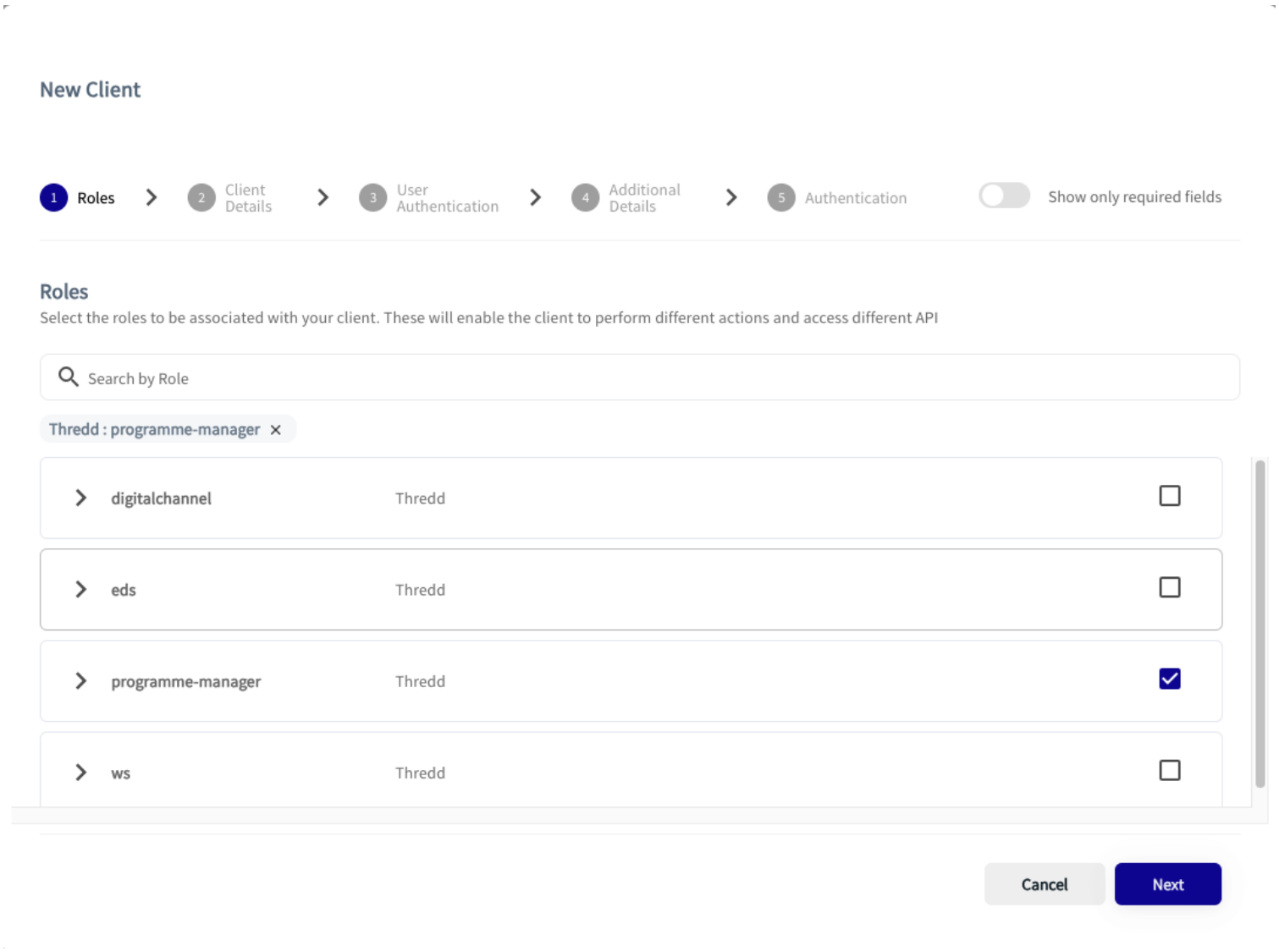


2. Click **New Application** (top-right of the screen). The **New Client** wizard appears.

Select Roles

The **New Client** shows the roles that are available to you. The organisation determines the available roles.

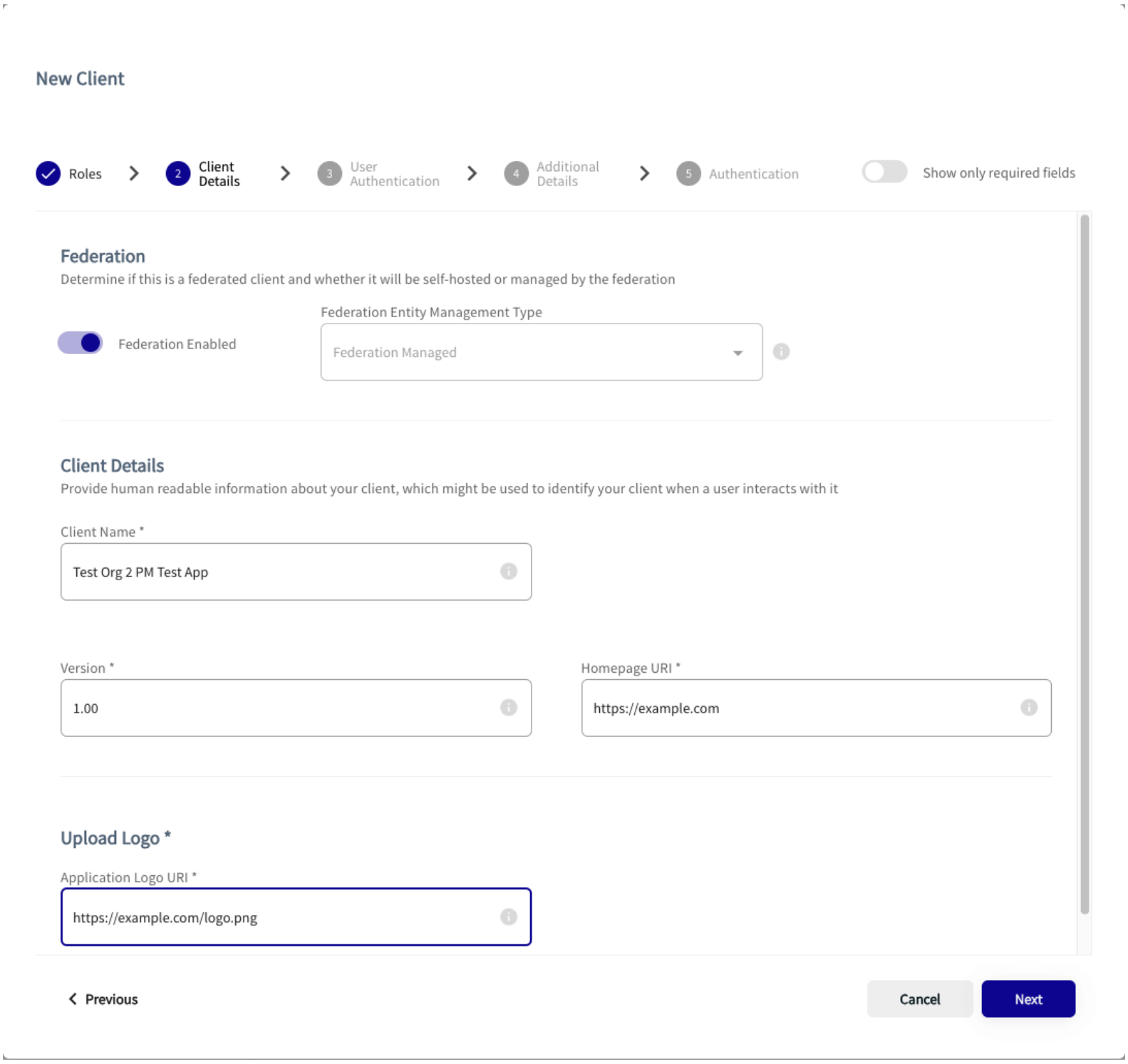
1. Tick a check box for the role you require. This example shows the selection of the **programme-manager** role.



2. Click **Next**.

Enter Client Details

1. Complete the form by entering details where applicable. The table below describes the fields.

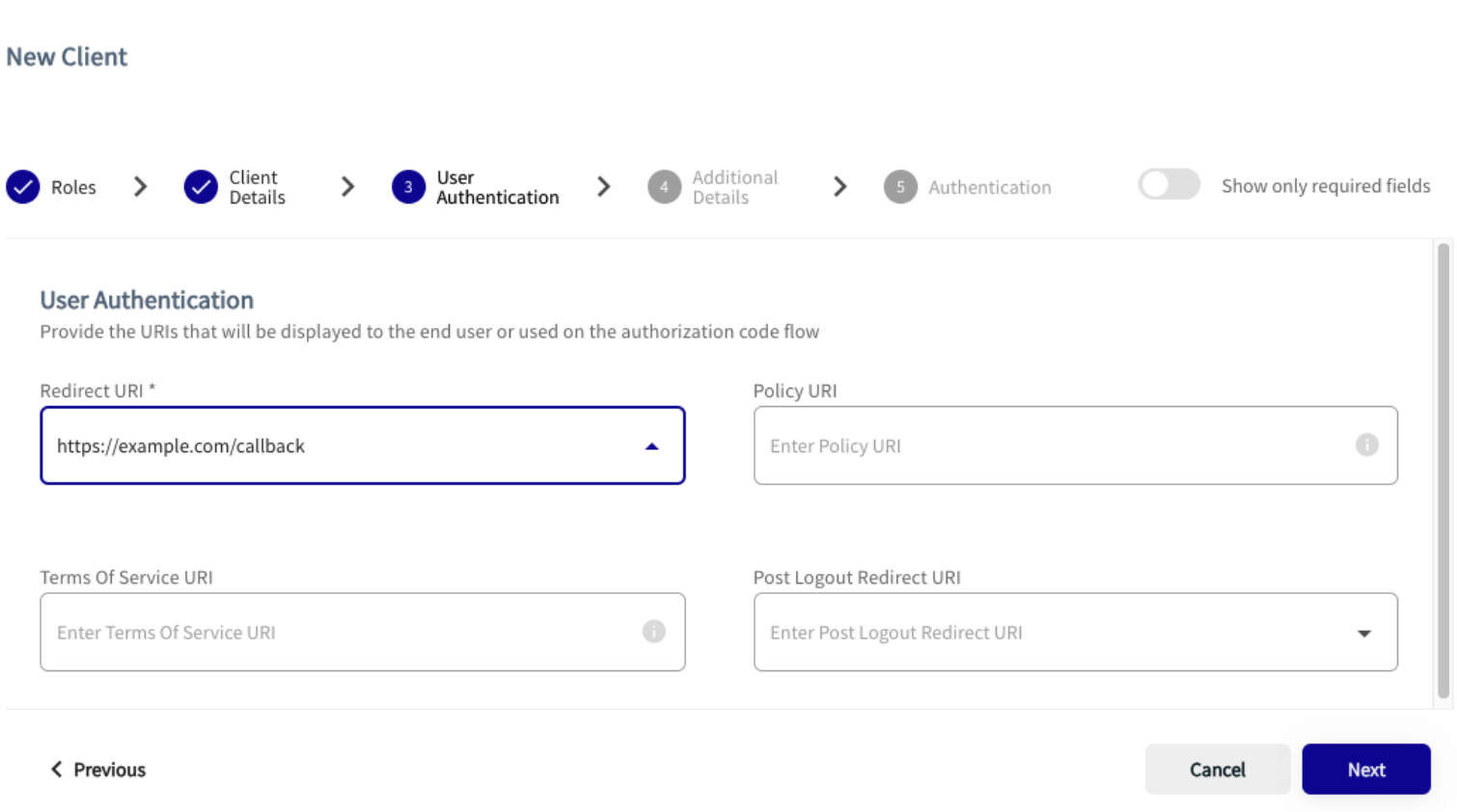


Field Name	Example Data	Notes
Federation Configuration	Federation Enabled	It is good practice to have federation enabled for all applications.
Client Name	Test Company App 1	The name of the intended OAuth client application. This setting is mandatory.
Version	1.0	Your chosen application version. This setting is mandatory.
Homepage URI	https://example.com	URI of the homepage. This setting is mandatory.
Upload Logo	https://example.com/logo.png	URI of the logo for the homepage. This must also include the file format of the logo. This setting is mandatory.

2. Click **Next**.

Enter User Authentication Settings

1. Complete the form by entering user authentication details where applicable. The following table describes the relevant fields.



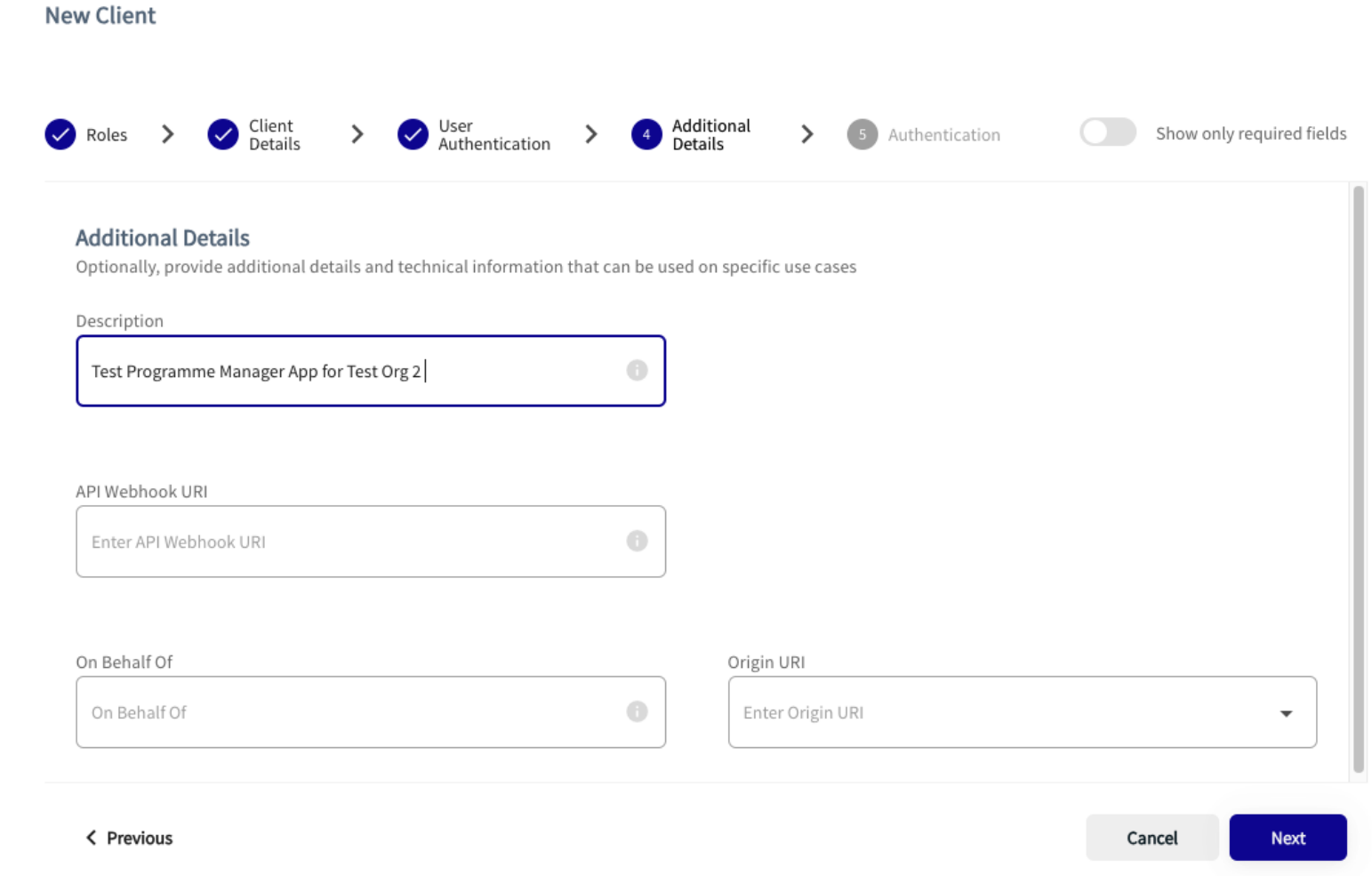
Note: You do not need to fill in the following settings on the page: Policy URL, Terms of Service URI, and the Post Logout Redirect URI.

Field Name	Example Data	Notes
Redirect URI *	https://example.com/callback	The callback URL any Confidential Client expects to be returned. This field is mandatory and you should enter the correct URI format.
Logo URI	https://example.com/logo.png	Company logo URL. This must also include the file format of the logo.

2. Click **Next**.

Type in Additional Details

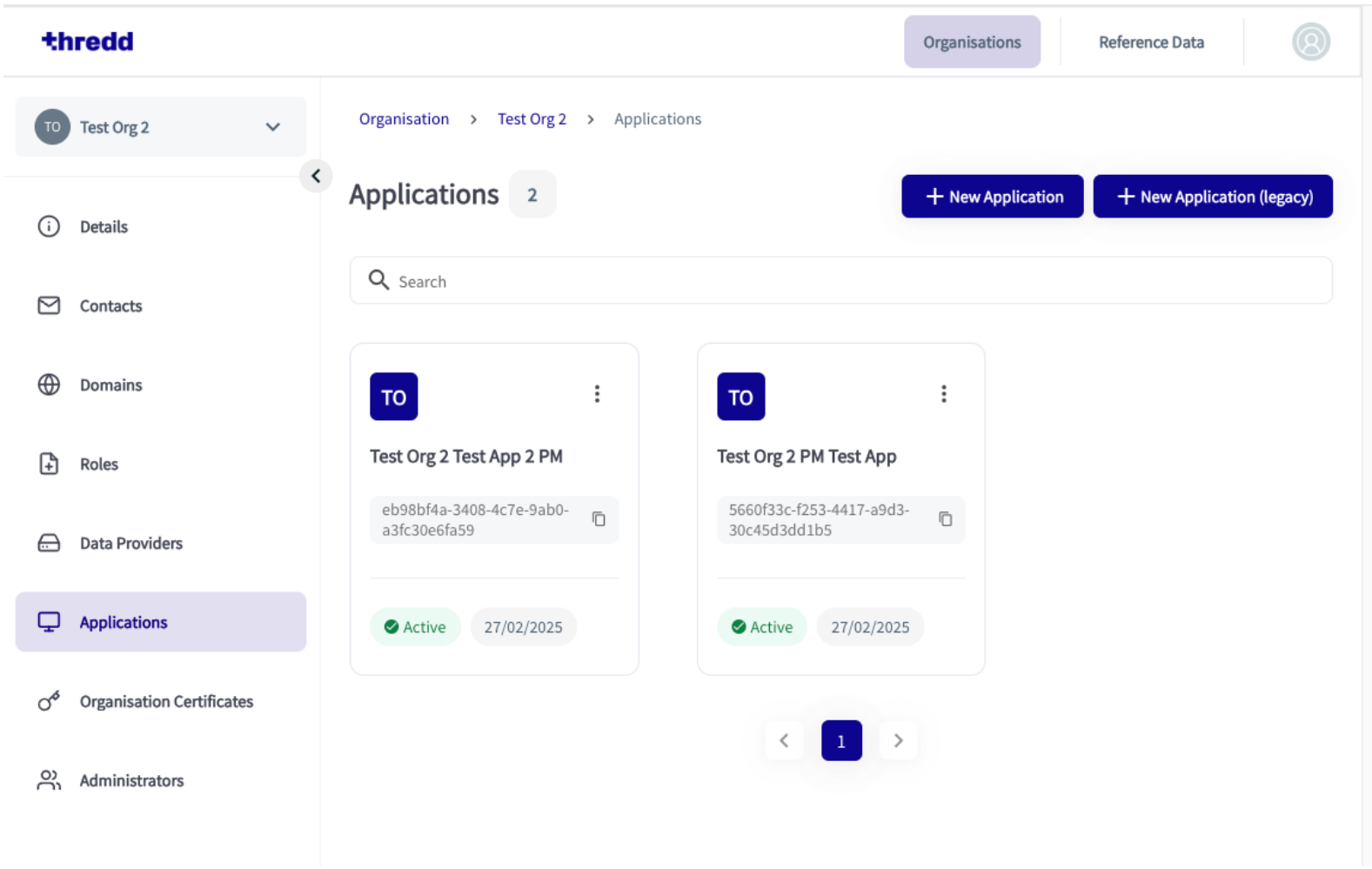
1. Complete the form by entering details where applicable. The following table describes the fields, all which are optional.



Field Name	Example Data	Notes
Description	Test Company App 1	Description of the app.
API Webhook URI	-	URI of the webhook.
On Behalf Of	-	On behalf of a particular user.
Origin URI	-	The origin URI.

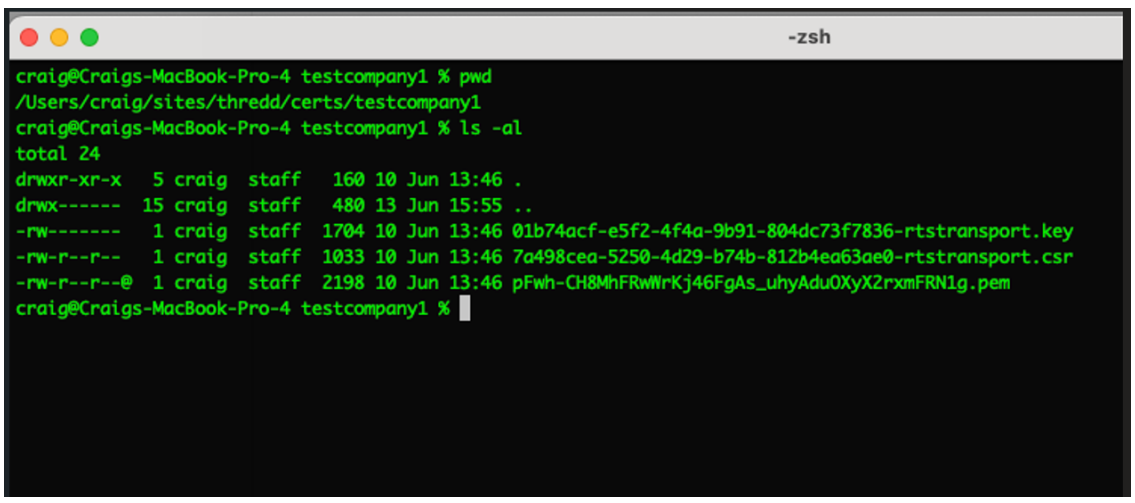
Note: Although not required, it is recommended that you enter a description.

2. Click **Next**.
3. Leave the following fields with their default values: **Token Signed Response Algorithm ID** and **Token Endpoint Authentication Method**. You should also leave **Require Signed Request Object** as enabled.
4. Click **Save**.
5. Ignore the step for adding an extra certificate.
6. Click **Done**. Your new Application appears as a tile on the **Applications** view.

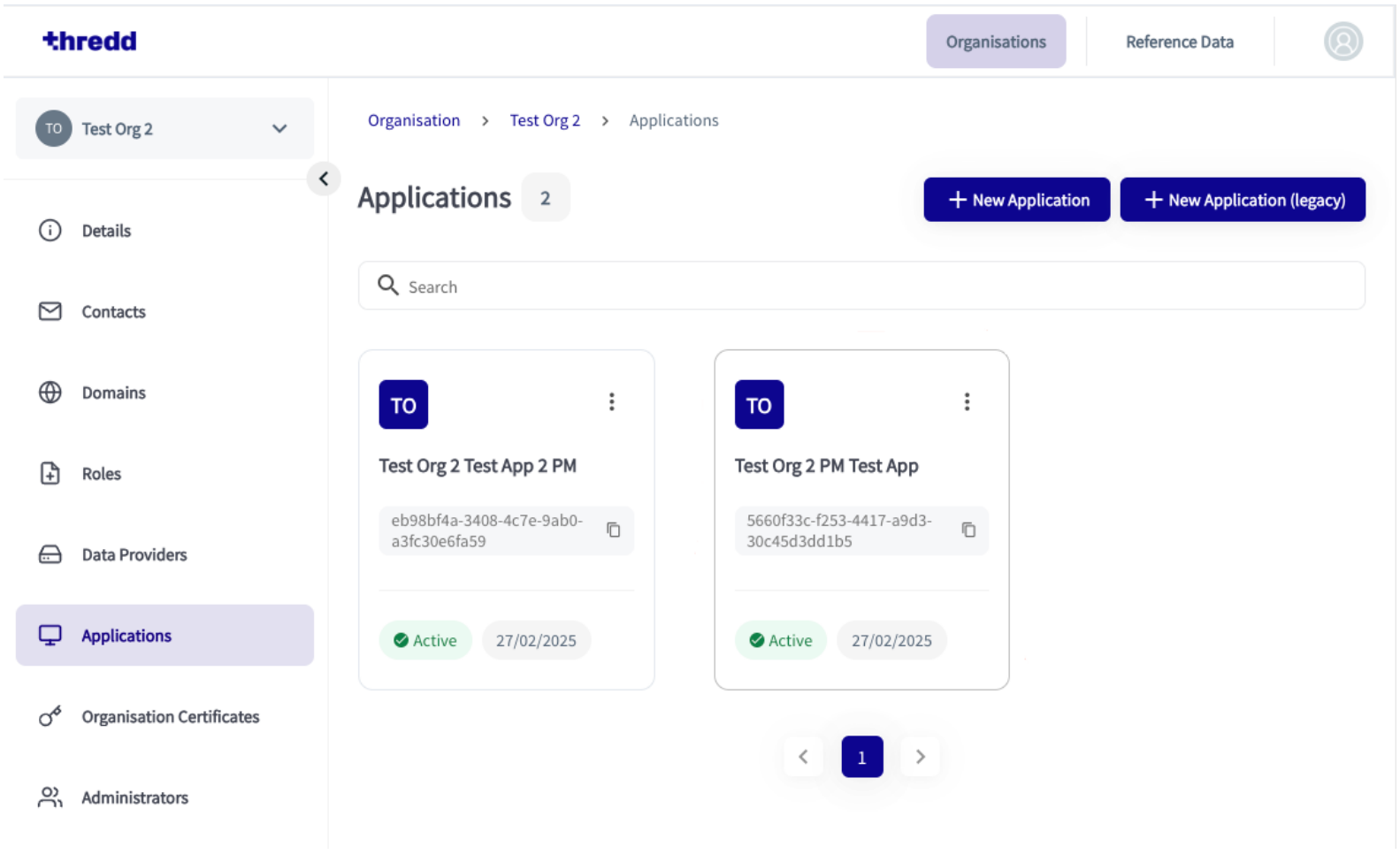


3.2.4 Step 3. Create a New Transport Certificate

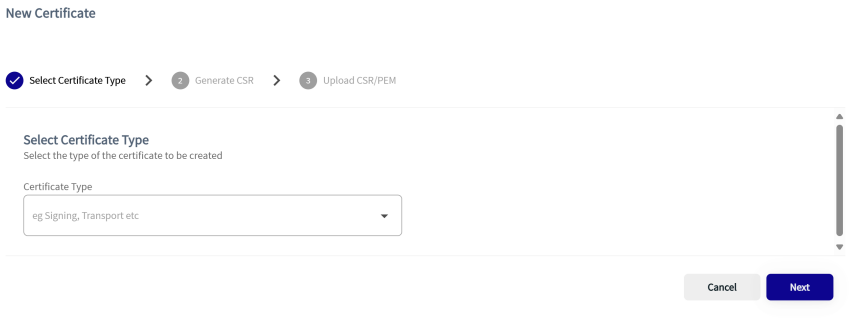
1. In a new terminal window, create an empty directory for storing the private key and CSR. The following example shows a key pair and CSR that have been created in a directory.



2. In the Thredd CA dashboard, click on the application tile you wish to create new certificates for. The second one below, Test Org 2 PM Test App, is an example.



4. Click the **App Certificates** tab.
5. Click the **New Certificates** button. The **New Certificates** window appears.



6. Choose **TRANSPORT** from the drop-down menu and click **Next**
7. Copy the generated CSR command and paste this into a terminal window. This action generates your private key and CSR.

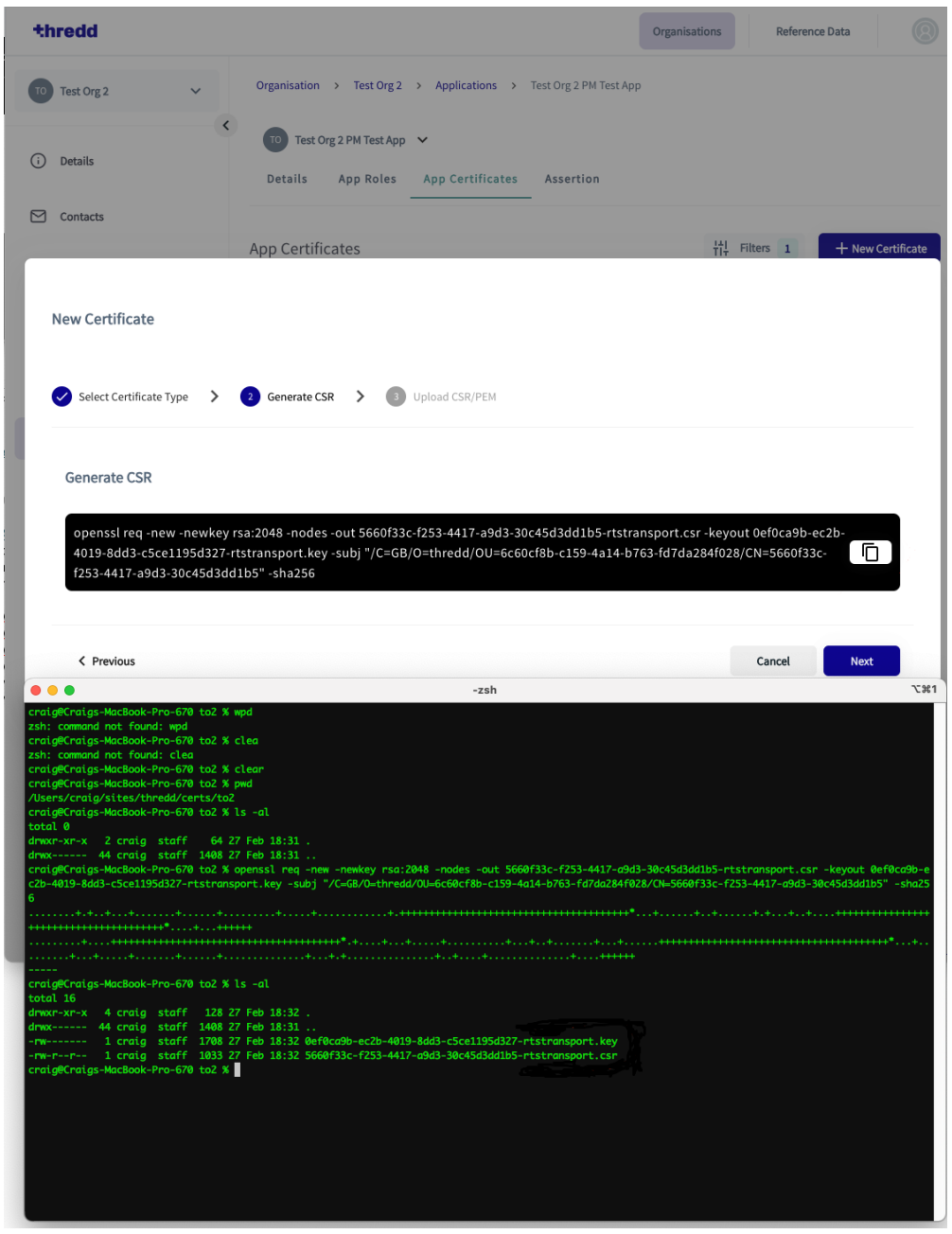
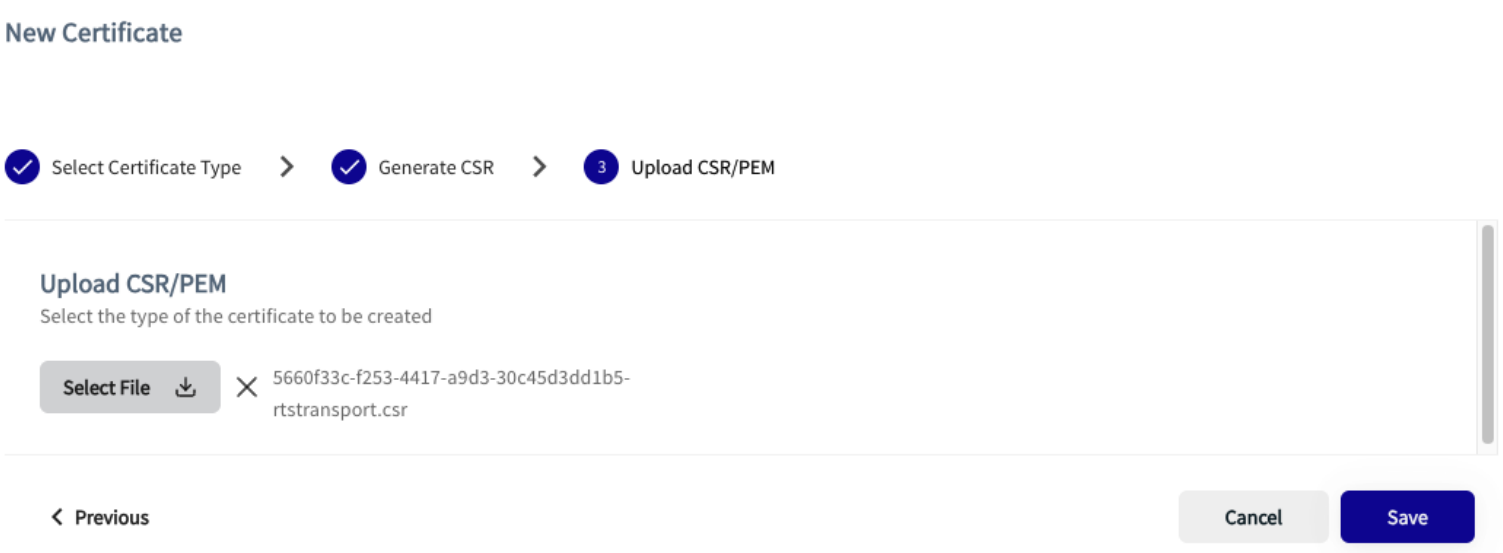
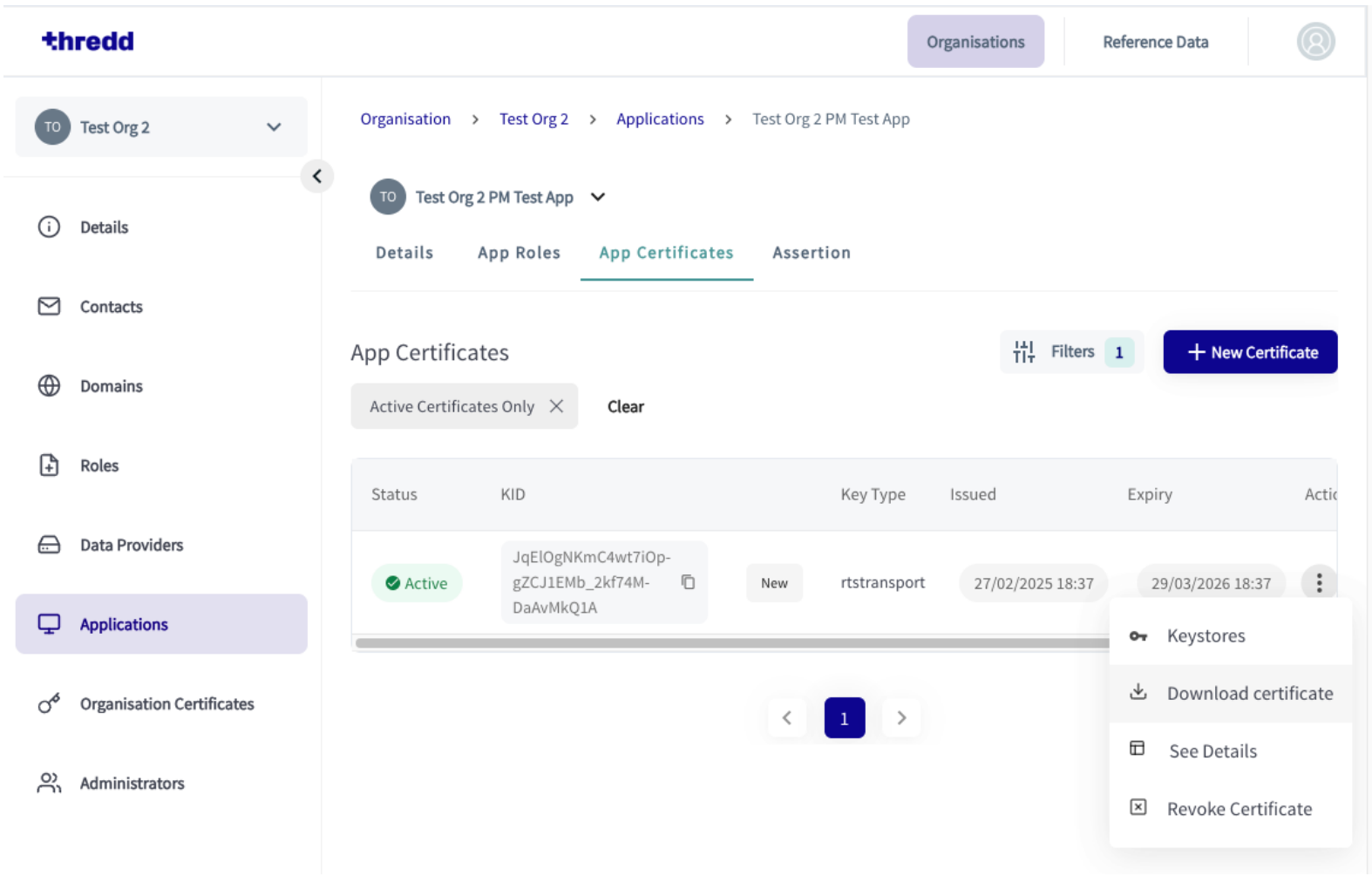


Figure 6: Terminal Window showing new command pasted into the prompt line

8. Click **Next** when the private key and CSR are generated, and select your CSR file for upload. Once selected, the CSR file appears on the screen.



9. Click **Save**. The signed x.509 certificate appears in your certificate list.



10. Download the certificate and move it to the same folder as the private key and CSR.

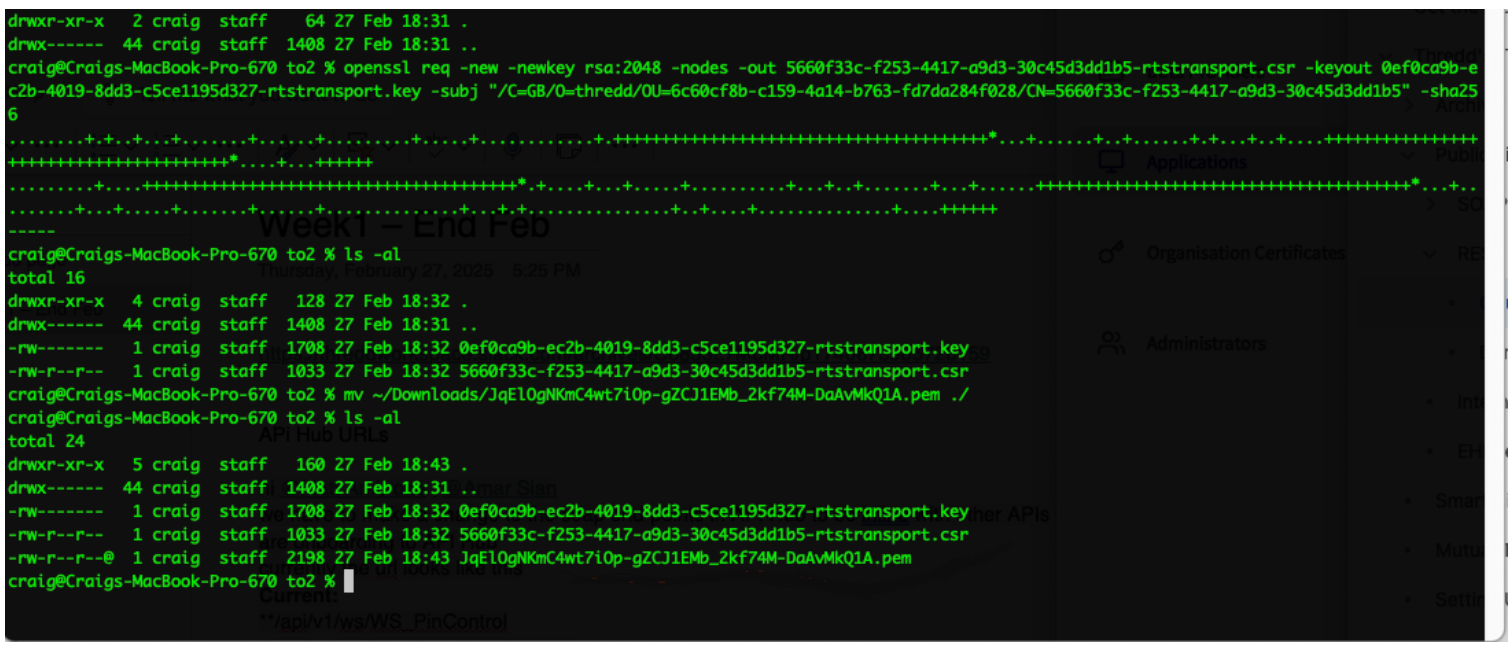


Figure 7: Open dialog box where you can select your CSR

You can now use the certificate as a Transport Certificate for any OAuth 2.0 client applications.

3.2.5 Step 4. Create a Signing Certificate

Client applications also require a *Signing Certificate* for signing assertions sent to the Authorisation Server. The private key for the signing certificate is used to sign client assertions for authentication as part of the token generation journey.

You need to repeat the steps described in Step 3. Create a New Transport Certificate. When selecting from the **Select Certificate Type** drop-down menu, select **Signing** in place of **Transport**.

3.3 Using Dynamic Client Registration (DCR) Endpoint Data

This page describes what you need to set for using the DCR endpoints. There are also guidelines on scopes that are required for the DCR endpoint. For details on the DCR endpoints in Postman, refer to the Cards API Website: Accessing Cards API with mTLS.

3.3.1 Adding Base URLs for DCR

For calling the DCR POST endpoint, you need to include the Base URLs within your REST tool, for example, Postman. The Base URLs are the hosts that accept the certificates. The Cards API, the API Hub, and Thredd CA hosts contain unique URLs, which can differ for UAT and Production. The Base URLs are as follows:

- **Cards API**: https://api.uat.threddpay.com/api/v1
- **API Hub**: https://uat-api.thredd.com
- **Thredd Certificate Authority**: https://matls-auth.directory.sandbox.threddid.com

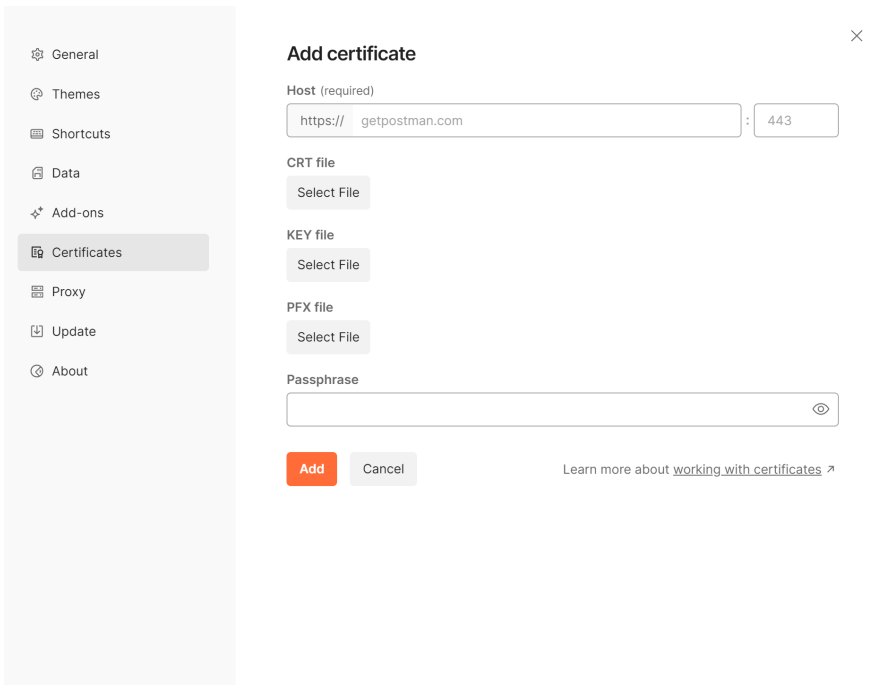
The following are used in Production.

- **API Hub**: https://api.thredd.com/ (for all PRD environments)
- **Cards API**: https://coreapi.threddpay.com

Adding Certificates to Hosts (Postman)

In Postman, you need to include the certificates with the individual hosts. Perform the following steps:

1. Click the gears icon and **Settings**.
2. Click **Certificates** from the left-hand menu.
3. Click **Add Certificate**.



4. Enter a host name.
5. Add CRT and Key files for the host.
6. Click the **Add** button.

For more details, refer to Cards API Website: Accessing the Cards API with mTLS.

3.3.2 Understanding Scopes in the DCR Endpoint

Depending on which Thredd Services you require, your Organisation will have been assigned a number of available Roles which can be chosen when you create your Application. The Scopes are what you can do for those Roles. The Scopes that you can register depend on the Roles your application is registered for. If you include more Scopes than are available for Roles, your application will not accept the new Scopes. This is because the Scopes you include depend on the Roles for the application.

Note: For more information on the different scopes, see Scopes.

For example, if you application includes the following "scopes":["bulkcard.read", "bulkcard.write" registered for the "bulk cards" role, then you cannot include additional Scopes in the request body. Adding a Scope such as "cards.write to the request body will not be accepted. The following shows an example of a DCR POST request.

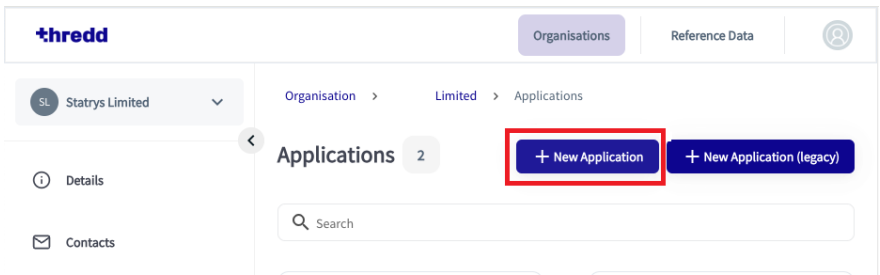

```
{
  "grant_types":[
    "client_credentials"
  ],
  "application_types":[
    "service",
    "dcr"
  ],
  "application_type":"service",
  "token_endpoint_auth_method":"private_key_jwt",
  "id_token_signed_response_alg":"RS256",
  "scopes":[
    "3ds.read",
    "bulkcard.read",
    "bulkcard.write",
    "cards.encrypted",
    "cards.read",
    "cards.sensitive",
    "cards.write",
    "cvv.read",
    "cvv.write",
    "digitalchannel",
    "fraud.read",
    "fraud.write",
    "pin.read",
    "pin.write"
  ],
  "tls_client_certificate_bound_access_tokens":true,
  "request_object_signing_alg":"RS256",
  "response_types":"token",
  "software_statement":"{{ssa}}"
}
```

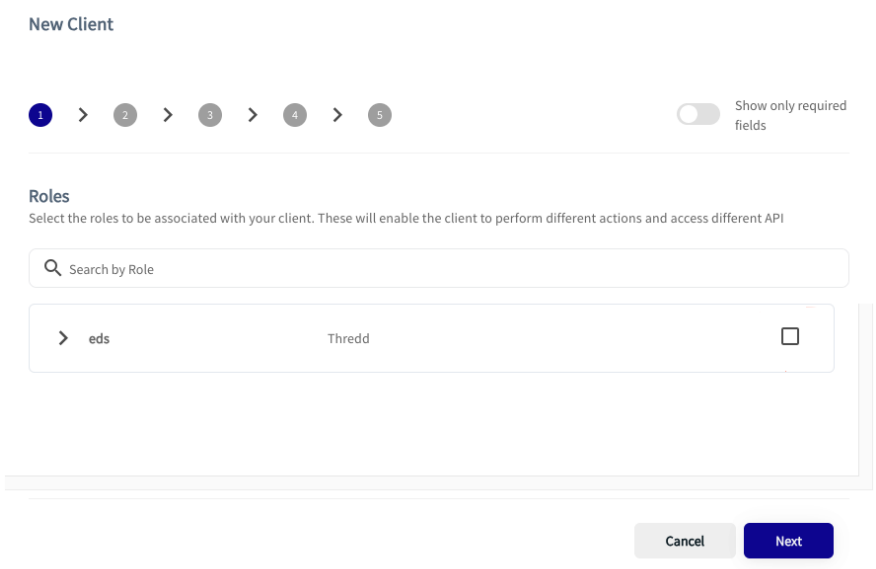

Setting a Role for a Scope

If your application requires a specific Scope, then you need to set the correct Role in Thredd Certificate Authority. When you have chosen the correct Role, you will see the Scope in the DCR request. The following example shows an application using the EDS Scopes of **eds.read** and **eds.write** when you select the **eds** Role in your application on the Thredd Certificate Authority.

For more details on the steps for creating REST certificates in the Thredd Certificate Authority dashboard, refer to Creating Client Application Certificates for REST APIs.

1. Click **New Application**.
   The Roles for the application appear.





2. Select the Role, in this case **eds** and click **Next**.
3. When you run the DCR request, you will see the eds scopes. See the example below.

```
{
  "grant_types":[
    "client_credentials"
  ],
  "application_types":[
    "service",
    "dcr"
  ],
  "application_type":"service",
  "token_endpoint_auth_method":"private_key_jwt",
  "id_token_signed_response_alg":"RS256",
  "scopes":[
    "3ds.read",
    "bulkcard.read",
    "bulkcard.write",
    "cards.encrypted",
    "cards.read",
    "cards.sensitive",
    "cards.write",
    "eds.read",
    "eds.write",
    "cvv.read",
    "cvv.write",
    "digitalchannel",
    "fraud.read",
    "fraud.write",
    "pin.read",
    "pin.write"
  ],
  "tls_client_certificate_bound_access_tokens":true,
  "request_object_signing_alg":"RS256",
  "response_types":"token",
  "software_statement":"{{ssa}}"
}
```

Updating Roles and Scopes

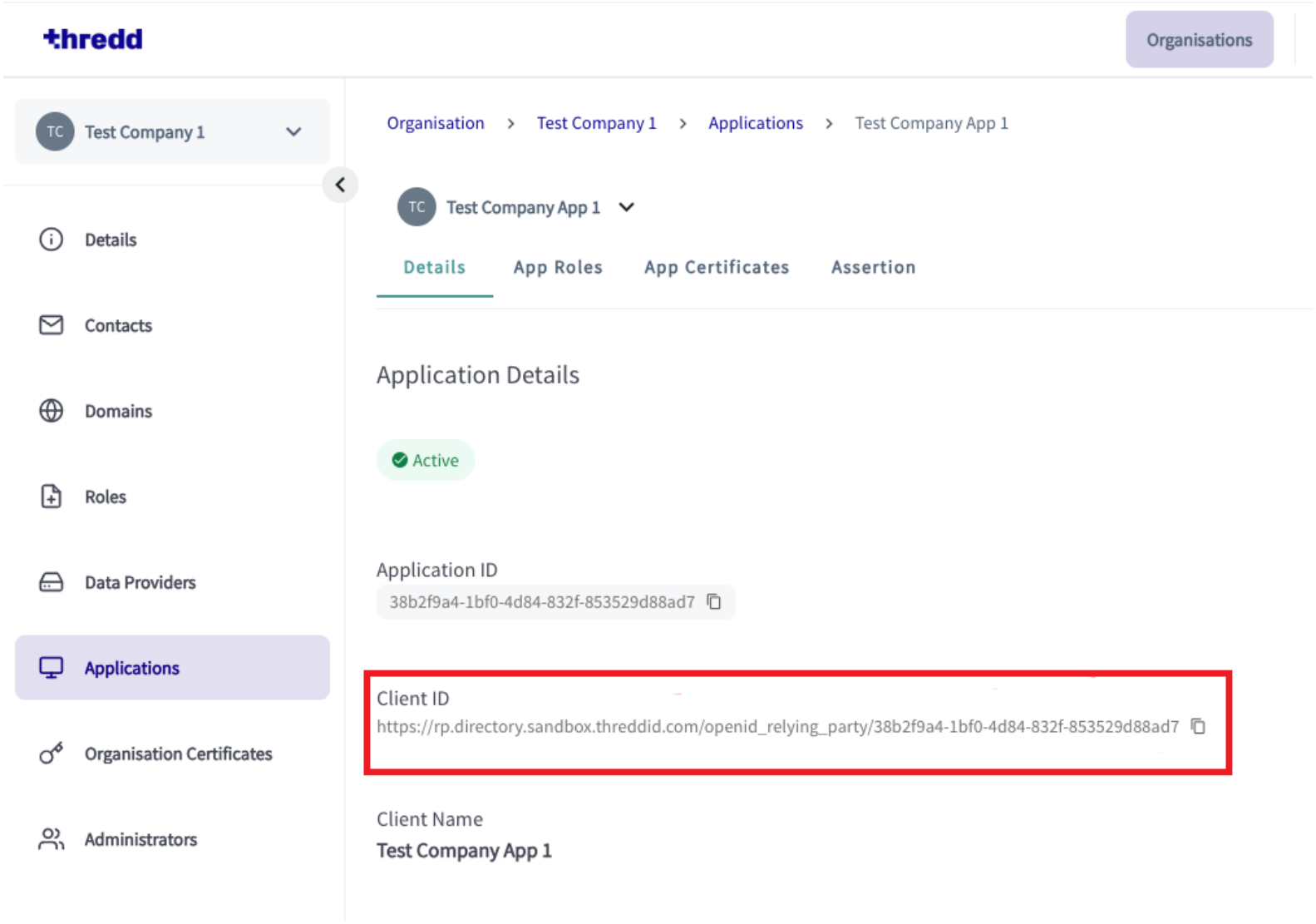
If required, you can update the Roles of an Application and the Scopes that your OAuth Client has registered. You may need new Scopes, for example, if a new service is added to Thredd and the organisation requires them. New Scopes may be part of an existing or new Role.
This depends on if you have access to the following variables: registration_access_token and the registration_client_uri (for details on these variables refer to Identifying and Storing the registration_access_token and registration_client_uri Data). To add Roles or Scopes there are a number of workflows to do this.

Note: For more details on the Postman steps refer to the Cards API Website: Access the Cards API with mTLS.

Note: Once you have performed a DCR POST or DCR PUT for adding or updating a client, contact Thredd so that we can update your Programme Manager (or Issuer) metadata related to the newly-registered or updated OAuth client.

Thredd Deletes a Client and Customer Creates a New One

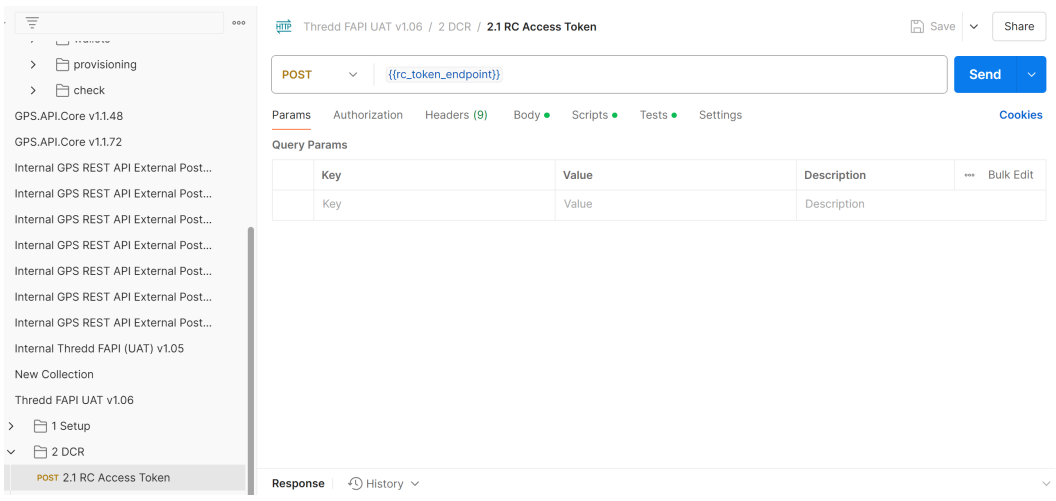
If you do not have access to the registration variables, you need to request Thredd to delete your existing OAuth Client. Once Thredd have deleted the OAuth Client, you can then create it again using the DCR POST request in Postman. Provide Thredd with the Client ID from the Thredd Certificate Authority Dashboard under **Applications > (Your Application) > Details**.



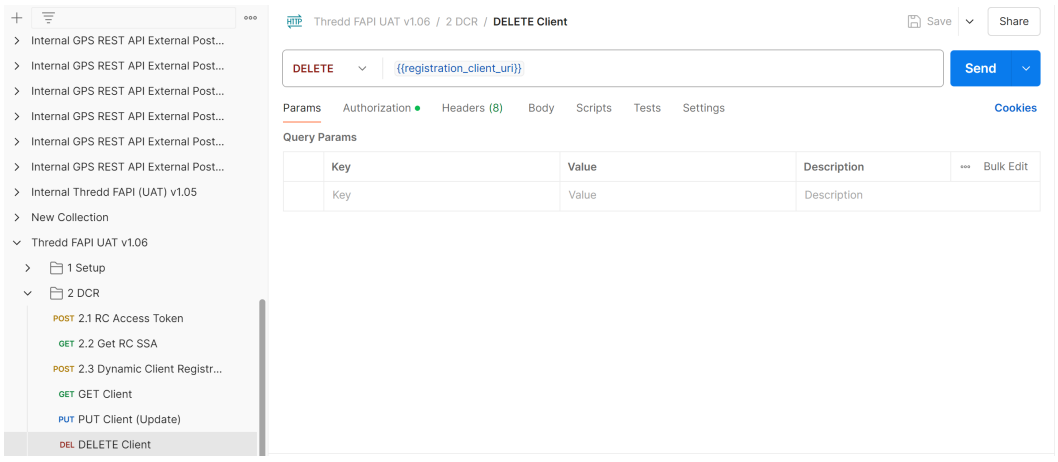
Customer Deletes an existing Client and Creates a New One

You should follow this method if you have access to the registration variables.

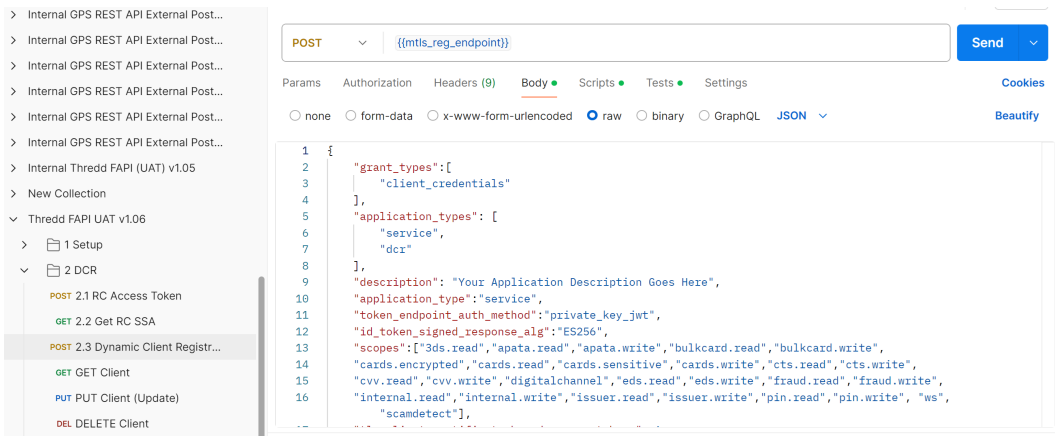
1. Use Postman to obtain a new access token.



2. Run the DCR Delete request in Postman.



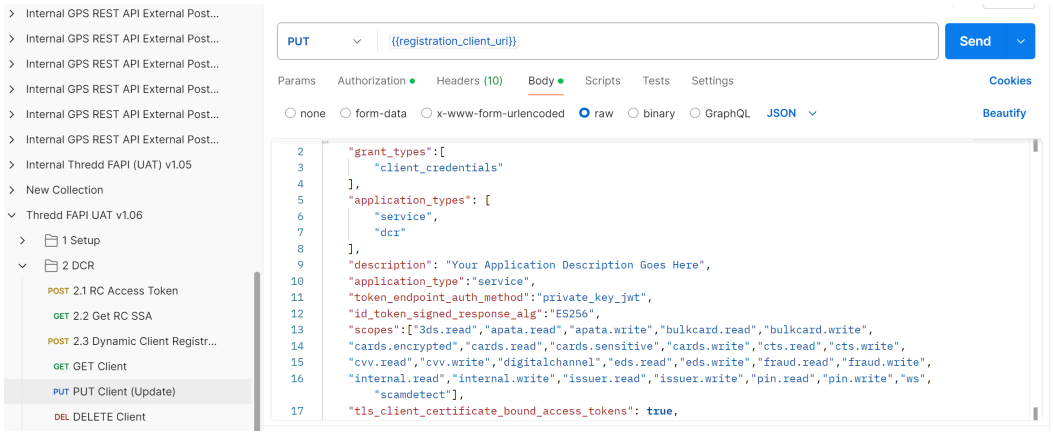
3. Run the SSA request in Postman.
4. Run the DCR POST request.



Customer Updates the Client Using the PUT Method

To update the OAuth Client, you can use the DCR PUT method. This is provided that you have access to the registration variables.

Run the DCR PUT request.



3.3.3 Identifying and Storing the registration_access_token and registration_client_uri Data

The registration_access_token and the registration_client_uri settings are returned in the response of the DCR endpoint. You must store these settings securely in order to use them again.

The following example response includes:

"registration_access_token": "PKW64FAixrjhxewTH9lR26o7m6_GvbL6RKWv0x5mIM.dWPLOrdXw7yGu2R1HAWlx8dSr3Jf-thb_zNFEGSOUNg"

and

"registration_client_uri": "https://auth.uat.threddid.com/confidential-clients/oauth2/register/https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4".


```
{
  "client_name":"Test Org 4",
  "description":"",
  "client_uri":"",
  "logo_uri":"https://example.com/logo.png",
  "policy_uri":"",
  "tos_uri":"",
```

```
  "organisation_id":"",
  "client_id":"https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4",
  "application_type":"service",
  "application_types":[
    "service",
    "dcr"
  ],
  "redirect_uris":[
    "https://example.com/cb"
  ],
  "grant_types":[
    "client_credentials"
  ],
  "response_types":[
    "token"
  ],
  "scope":"3ds.read bulkcard.read bulkcard.write cards.encrypted cards.read cards.sensitive cards.write cvv.read cvv.write digit-
alchannel fraud.read fraud.write internal.read internal.write issuer.read issuer.write pin.read pin.write",
  "scopes":[
    "3ds.read",
    "bulkcard.read",
    "bulkcard.write",
    "cards.encrypted",
    "cards.read",
    "cards.sensitive",
    "cards.write",
    "cvv.read",
    "cvv.write",
    "digitalchannel",
    "fraud.read",
    "fraud.write",
    "internal.read",
    "internal.write",
    "issuer.read",
    "issuer.write",
    "pin.read",
    "pin.write"
  ],
  "audience":[
    "https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4"
  ],
  "token_endpoint_auth_method":"private_key_jwt",
  "revocation_endpoint_auth_method":"private_key_jwt",
  "introspection_endpoint_auth_method":"private_key_jwt",
  "token_exchange":{
    "actor_claims":null
  },
  "token_endpoint_auth_signing_alg":"",
  "jwks":{
    "keys":[
    ]
  },
  "jwks_uri":"https://keystore.directory.sandbox.threddid.com/99b2b599-8cc2-499e-b805-e3d84233ae8e/b9f04639-cdeb-425b-a310-
450d3d293fd4/application.jwks",
  "request_object_signing_alg":"RS256",
  "request_object_encryption_alg":"",
  "request_object_encryption_enc":"",
  "request_uris":[

  ],
  "client_id_issued_at":1723556609,
  "created_at":"2024-08-13T13:43:29.619067912Z",
  "updated_at":"2024-08-13T13:43:29.619067912Z",
  "client_secret_expires_at":0,
  "sector_identifier_uri":"https://keystore.directory.sandbox.threddid.com/99b2b599-8cc2-499e-b805-e3d84233ae8e/b9f04639-cdeb-
425b-a310-450d3d293fd4/redirect_uris.json",
  "userinfo_signed_response_alg":"none",
  "id_token_signed_response_alg":"ES256",
  "id_token_encrypted_response_alg":"",
  "id_token_encrypted_response_enc":"",
  "tls_client_certificate_bound_access_tokens":true,
```

```
  "tls_client_auth_subject_dn":"",
  "tls_client_auth_san_dns":"",
  "tls_client_auth_san_uri":"",
  "tls_client_auth_san_ip":"",
  "tls_client_auth_san_email":"",
  "privacy":{
    "scopes":null
  },
  "subject_type":"pairwise",
  "backchannel_token_delivery_mode":"",
  "backchannel_client_notification_endpoint":"",
  "backchannel_authentication_request_signing_alg":"",
  "backchannel_user_code_parameter":false,
  "require_pushed_authorization_requests":false,
  "authorization_signed_response_alg":"ES256",
  "authorization_encrypted_response_alg":"",
  "authorization_encrypted_response_enc":"",
  "dpop_bound_access_tokens":false,
  "authorization_details_types":null,
  "post_logout_redirect_uris":[
  ],
  "app_url":"",
  "backchannel_logout_uri":"",
  "backchannel_logout_session_required":false,
  "client_secret":"xT4SDSlIp9J3aLMIHMtQ99Uok6onqaQJEiQ0Hp2QBXc",
  "hashed_secret":"c2d-
c89f-
d3581d6222a7a143f173a4696feee7773f4156ada52c241eea25b30990f-
b35f2-
faa42105f1e51895d-
d7ced75f0847659f6396f-
bc977cf71d73-
bef7a1a7f93830e871ae9f-
b767318b0c545ee9543dc8ae94044cfc8a9b26bcf8bad53e3ba00ef889d1d8338dfdbdedd1aed382b51a86cf5f08d21d4cbf6f23d66141c50",
  "software_id":"b9f04639-cdeb-425b-a310-450d3d293fd4",
  "software_version":"1.00",
  "software_statement":"eyJraWQiOiJz-
aWduZXIiLCJ0eXAiOiJKV1QiLCJh-
bGciOiJQUzI1NiJ9.eyJzb2Z0d2FyZV9zdGF0ZW1lb-
nRfcm9sZXMiOlt-
dLCJvcm-
dfandrc190cmFuc3BvcnRf-
dXJpI-
joi-
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL3RyYW5zcG9y-
dC5qd2tzIi-
wiYXBpX3dlYm-
hvb2t-
fdXJp-
cyI6W10sIm9yZ19zdGF0dXMiOiJBY3Rp-
dmUiLCJs-
b2d-
vX3Vy-
aSI6Im-
h0dHBzOi8vZXh-
hbXBsZS5jb20vbG9nby5wbm-
ciLCJy-
b2xl-
cyI6W10sIm-
lzcyI6InRocmVkZCBzYW5kYm94IFNTQSBp-
c3N1ZXIiLCJvcm-
dfandrc190cmFuc3BvcnR-
faW5hY3Rp-
dmVf-
dXJpI-
joi-
```

```
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL2luYWN0aXZlL3RyYW5zcG9y-
dC5qd2tzIi-
wiY2x-
pZW50X2lkI-
joi-
aHR0cHM6Ly9ycC5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vb3Bl-
bmlkX3Jl-
bHlpb-
mdfcGFy-
dHk-
vYjlmMDQ2Mzk-
tY2RlYi00MjViLWEzMTAtNDUwZDNkMjkzZmQ0Ii-
wiY2x-
pZW50X2Rl-
c2Ny-
aXB0aW9uI-
joiVGVzdCBPcm-
cgNCBp-
cyBhIEN1c3RvbWVyIG9mIFRocmVkZCIsIm1vZGUiOiJMaXZlIi-
wic29m-
dHd-
hcmV-
faWQiOiJiOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQiLCJzb2Z0d2FyZV92ZXJz-
aW9uI-
joiMS4wMCIsIm9yZ19uYW1lI-
joiVGVzdCBPcm-
cgNCIsInNvZnR3YXJlX2ZsYWdzIjp7fSwib3JpZ2luX3Vy-
aXMiOlt-
dLCJjbGllb-
nRf-
bmFtZSI6IlRl-
c3QgT3JnIDQiLCJpYXQiOjE3MjM1NTYzNTAsIm-
p3a3Nf-
dHJhb-
nNw-
b3J0X3Vy-
aSI6Im-
h0dHBzOi8va2V5c3RvcmUuZGlyZWN0b3J5LnNhb-
mRib3gudGhyZWRkaWQuY29tLzk5YjJiNTk5LThjYzItNDk5ZS1iODA1LWUzZDg0MjMzYWU4ZS9iOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQvdHJhb-
nNw-
b3J0Lm-
p3a3MiLCJvcm-
dhbm-
lzYXRp-
b25f-
dGFncyI6W10sInN1Ym-
plY3Rf-
dHl-
wZSI6InBhaXJ3aXNlIi-
wicmVkaXJlY3Rf-
dXJp-
cyI6WyJodHRw-
czovL2V4YW1w-
bGUuY29tL2NiIl0sInNlY3Rvcl9pZGVudGl-
maWVyX3Vy-
aSI6Im-
h0dHBzOi8va2V5c3RvcmUuZGlyZWN0b3J5LnNhb-
mRib3gudGhyZWRkaWQuY29tLzk5YjJiNTk5LThjYzItNDk5ZS1iODA1LWUzZDg0MjMzYWU4ZS9iOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQvcmVkaXJ-
lY3Rf-
dXJpcy5qc29uIi-
wib3JnX2p3a3N-
faW5hY3Rp-
dmVf-
dXJpI-
joi-
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL2luYWN0aXZlL2Fw-
cGx-
pY2F0aW9uLm-
p3a3MiLCJvcm-
dhbm-
lzYXRp-
```

```
cGx-
pY2F0aW9uLm-
p3a3MiLCJqd2tzX2luYWN0aXZlX3Vy-
aSI6Im-
h0dHBzOi8va2V5c3RvcmUuZGlyZWN0b3J5LnNhb-
mRib3gudGhyZWRkaWQuY29tLzk5YjJiNTk5LThjYzItNDk5ZS1iODA1LWUzZDg0MjMzYWU4ZS9iOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQvaW5hY3R-
pdmUvYXBw-
bGljYXRp-
b24uandrcyIsIm9yZ19udW1iZXIiOiIwMDAwMDAwNCIsIm-
p3a3Nf-
dXJpI-
joi-
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL2I5ZjA0NjM5LWNkZWItNDI1Yi1hMzEwLTQ1MGQzZDI5M2ZkNC9h-
cHBsaWNhdGlvbi5qd2tzIiwic3RhdHVzIjoiQWN0aXZlIiwib3JnYW5pc2F0aW9uX2NvbXBldGVudF9hdXRob3JpdHlfY2xhaW1zIjpbXX0.N5wPwnVFcramX7BAYOWn_
1tWcLxSRNaIxst-ZK_p29he5mN1_hdFF6BoVMG0Chh-dV_FL4-luQlg7Qn3EMRCb3RWp8U7pcj0lmFSDcUyvlQpoB5hqrXgjPEOkIC7uqJuTDjwakj0NFtJDtK_l_FVp-
PA2ZsDJanwCxsnxxMxVFlxFPwCYk8jBTNE84zx092a4-Tj4VfE4e5S5HA7if8sR7PiXvAMWs1jBrUQ9enWacfu_Xno_-kywtrkAeR4fi-MGNeA4Ik-
bYbh8LyhMjg3XXun87BoL8-b_An5fX5y-XS8VnENg3NFU3D8soQ05OKlXnIDT2Dp7v1lFlNYrV1jO9w",
  "dynamically_registered":true,
  "registration_access_token":"PKW64FAixrjhxewTH9lR26o7m6_GvbL6RKWv0x5mIM.dWPLOrdXw7yGu2R1HAWlx8dSr3Jf-thb_zNFEGSOUNg",
  "registration_client_uri":"https://auth.uat.threddid.com/confidential-cli-
ents/oauth2/register/https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4",
  "registration_access_token_expires_in":0
}
```

Section 4: EHI Setup

You should read this section to understand the setup steps for EHI.

Topics covered in this section:

- Setting Up EHI with mTLS

4.1 Setting Up EHI with mTLS

This section describes how to set up the External Host Interface (EHI) to communicate with Thredd using Mutual Transport Level Security (mTLS). This involves:

1. Setting up both server and client certificates for the mTLS connection.
2. Testing the certificates to prove the establishment of the mTLS connection.

To find out where mTLS exists in the network infrastructure between Thredd and your EHI endpoint, refer to background information in Message Architecture Between Thredd and EHI. For information on boosting security in the mTLS setup, see Additional Security Controls.

4.1.1 Overview of mTLS in EHI

mTLS is used where both parties must prove their identities to each other for authentication, in this case Thredd and yourself, the Client. The server and the client present their respective certificates and private keys to verify who they claim to be. The high-level communication flow is as follows:

1. The client (Thredd) connects to the server (you, the Client).
2. The server (you) presents your certificate.
3. The client (Thredd) verifies your certificate.
4. The client (Thredd) presents their certificate.
5. The server (you) verifies the client's certificate.

Once both certificates are verified, the server grants access. The client and server then exchange information over the secure connection.

mTLS relies on a system of digitally signed certificates issued by a trusted third party called a certificate authority (CA). The CA proves the authenticity of the public key and the identity of the server presenting the key. In addition to the server certificate, you also needs to have root and leaf certificates for validation. This ensures that a Chain of Trust is established. For a more detailed description on mTLS, refer to the descriptions on the Cloudflare website.

4.1.2 Set Up Certificates on the Server

You first need to set up the server certificates on your EHI server. The EHI server validates the client certificates that Thredd sends to you.

1. Obtain a Server Certificate from a Certificate of Authority vendor, such as Verizon or Digicert or Amazon Web Services.
2. Install certificates on your EHI listening endpoint. The certificates are as follows:

- **Server or Leaf Certificate** – this certificate is issued to individual servers by a CA. This certificate provides the “leaf” of a hierarchical tree of authority that’s traceable to the trusted root certificate.
- **Intermediate or Issuing Certificate** – this is a digital representation of a computer's identity in the PKI system, containing the public encryption key of the system, the purpose of the certificate, the identity of the issuing Certificate Authority, the validity period, and the bearer's name and digital signature.
- **Root or CA Certificate** – this enables an organisation to be their own Certificate Authority. This certificate also contains the public key. Typically, the certificate is installed on the server.

4.1.3 Store the Client Certificates

As Thredd sends you a Client Certificate, your server verifies the incoming certificate by matching it against the information held in Thredd's Certificate Authority’s Root and Issuing Certificates. You need to store the CA's Root and Issuing Certificates on your mTLS termination point. The steps for storing depend on your CA Vendor. For example, the steps may differ if your vendor is Alibaba Cloud or AWS.

Note: The mTLS termination point is part of the infrastructure that manages the mTLS communication, and is where the certificates are stored.

The following are the Sandbox Certificates for UAT.

Certificate Type	URL
Root	https://crl.pki.sandbox.threddid.com/root-ca.pem
Issuing	https://crl.pki.sandbox.threddid.com/issuer-ca.pem

The following are the Production Certificates

Certificate Type	URL
Root	https://crl.pki.threddid.com/root-ca.pem
Issuing	https://crl.pki.threddid.com/issuer-ca.pem

4.1.4 Testing the EHI Endpoint

Once you have set up your EHI endpoint with mTLS, you should test the endpoint with the online SSLLabs tool and OpenSSL. This is to ensure that you can successfully communicate with EHI over mTLS. Once you have completed testing, provide the EHI endpoint to Thredd.

Note: You must not provide the EHI endpoint if you have not completed testing.

Test Using SSLLabs

1. Go to the URL of the tool: https://www.ssllabs.com/ssltest/
2. Enter the URL to be tested in the SSL Labs test screen test page. The results appear similar to the following:

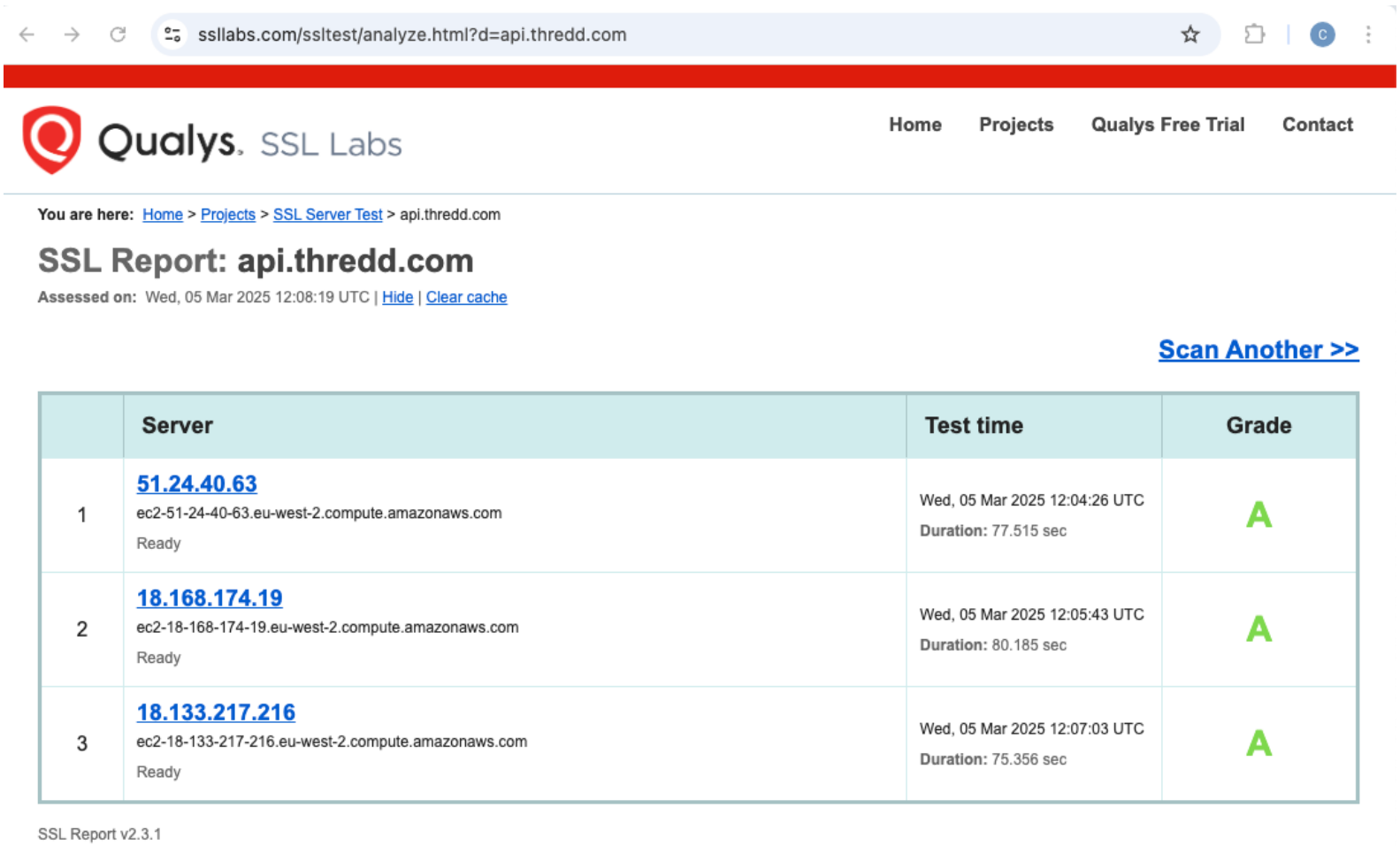


Figure 8: Passing Tests on SSL Labs

An “A” Grade results in the test passing. However, “B” will not result in a pass, and can indicate that the problem is due to a missing an Immediate or Root certificate on your server.

Test Using OpenSSL

You run the following command that triggers the TLS Handshake for the communication between server and the client. You receive responses for the server and client certificates.

```
openssl s_client -connect ehi.yourdomain.com:443
```

Server Certificate Result

The results for the server certificate appears as follows:

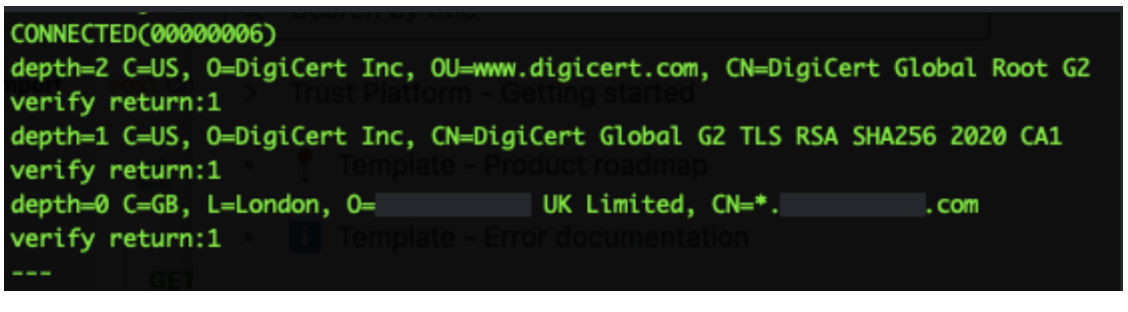


Figure 9: Server Certificate Result Including Depth Settings

If `Depth 0,1,2` all show `verify return:1` this indicates that the EHI servers trust the TLS endpoint/Server Certificate. There could be an issue with server certificate result, if the results appear differently, for example, there is no `verify return:1` for the depth settings. The Depth settings mean the following:

Depth Setting	Description
Depth 0 = Root	The Root certificate has been sent
Depth 1 = Intermediate:	The Leaf certificate has been sent.
Depth 2= Root:	The Root certificate has been sent.

You should configure your server's TLS setup so that it sends the Server Certificate and the Intermediate Certificate. If the server sends only the Server certificate the Chain of Trust is not complete, which results in the mTLS not being set up between the server and the client.

Client Certificate Result

For validating the Client Certificate, certificates used by Thredd appear under `Acceptable client certificate CA names`. An issue may exist if the `Acceptable client certificate CA names` section is empty, where there Thredd's CAs are not listed.

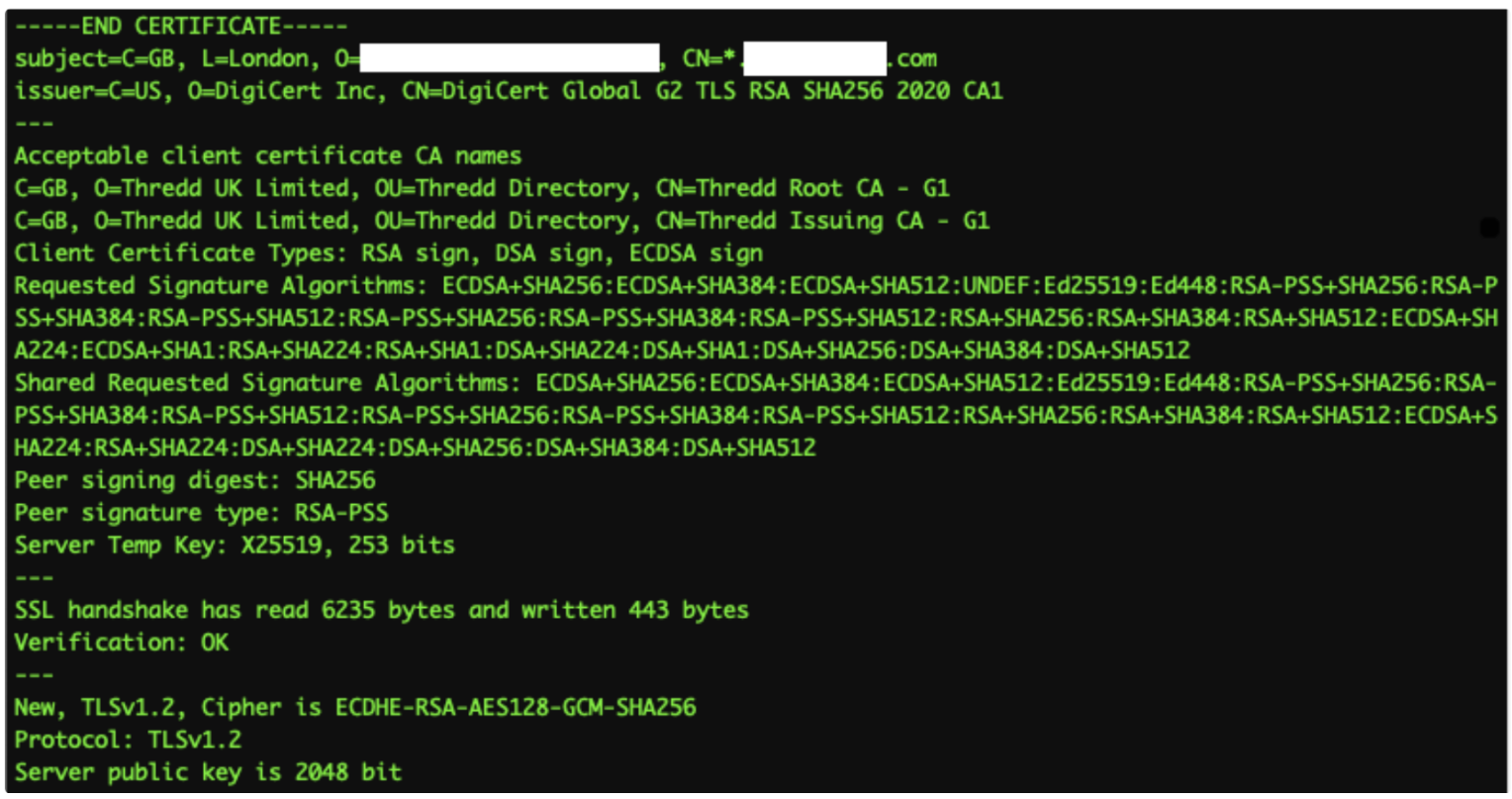


Figure 10: Client Certificate Response

Note: Listing the CA authority is optional. If this is empty, it may not mean that the mutual part of the certificate validation has failed.

Note: You may need to check the EHI listener endpoint configuration if the following has occurred:

- If the SSL Labs test does not give an A Grade Response
- the Server Certificate does not show `verify return:1` at all depths
- the Acceptable client certificate CA names are not sent

Testing Requests Without a Certificate

For additional testing, you can test a request to the EHI endpoint that does not contain a certificate. Using cURL, you see a 400 or 403 response as in the following example.

```
# Request
curl -v https://api.yourdomain.com/ehim/endpoint/api
# Partial Response
<html>
<head><title>400 No required SSL certificate was sent</title></head>
<body>
<center><h1>400 Bad Request</h1></center>
<center>No required SSL certificate was sent</center>
```

Testing Requests With a Certificate

You can also test a request to the EHI endpoint that contains a certificate. Using cURL, you see a 200 response as in the following example.

```
# Request
curl --cert ./exampleClientCert.pem --key ./exampleClientCert.key \
     -X POST \
     -H "Content-Type: application/json" \
     -d '{"foo": "bar"}' \
     -v https://api.yourdomain.com/ehim/endpoint/api
# Partial Response
200 OK
```

4.1.5 Message Architecture Between Thredd and the Customer EHI

The following shows the message architecture between Thredd and Thredd and your EHI components (the EHI Customer). The EHI listener endpoint and the mTLS termination point are part of your EHI components.

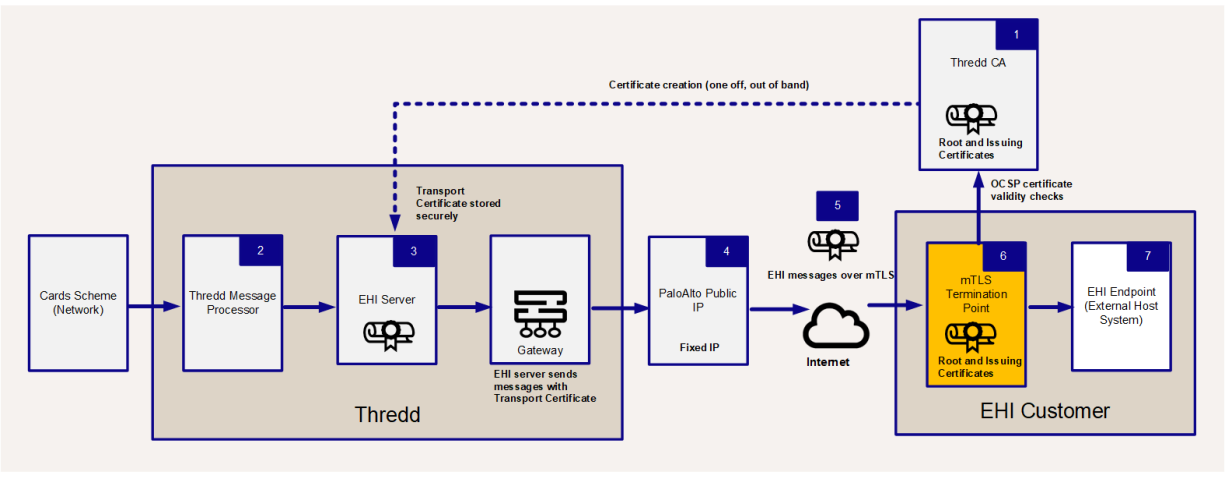


Figure 11: Message Architecture Between Thredd and EHI

1. Thredd uses the *Certificate Authority*, Thredd Certificate Authority, to create a Transport Certificate to enable communication later over mTLS.
2. Thredd's Message Processor sends and processes EHI messages that are received from the Cards Schemes (Networks) to Thredd's EHI servers.
3. Thredd sends the EHI messages via a gateway and adds the Transport Certificate.
4. EHI requests travel across the Internet with the associated Transport Certificate attached. Note that Thredd's outgoing firewall has a fixed IP address.
5. The Transport Certificate is presented to the customer's incoming mTLS termination point during the connection handshake.
6. Your systems validate the Transport Certificate by checking it against the CA chain of trust (Root and Issuing Certificates).
7. When the mTLS handshake is complete, you receive EHI messages on your EHI endpoint (External Host System).

4.1.6 Adding Security Controls

You can choose to implement additional optional controls to enhance mTLS security. These include the:

- **IP Address Allow List** – Thredd EHI messages are sent from a firewall with a fixed IP address. Optionally, you can add this IP address to your allowlist.
- **Certificate Pinning** – While our mutual communication is secured by our Transport Certificate, which is only trusted if it has been issued by our CA, you can choose to also implement Certificate Pinning. Certificate Pinning blocks attempted requests made with the incorrect certificates. For more information, see Certificate and Public Key Pinning | OWASP Foundation.

Section 5: Other Thredd Applications

You should read this section to understand the setup steps for other Thredd applications, including Smart Client and Thredd Portal.

Topics covered in this section:

- Smart Client and CTS
- Thredd Portal

5.1 Connecting to Smart Client and CTS

5.1.1 Smart Client Installation

Complete the following steps to connect to Smart Client:

1. Ensure that the laptop you are using has an internet connection.
2. Go to a Thredd-hosted page for downloading the Smart Client (SC) installer.
3. Run the install program.

For UAT, the download links are:

- Smart Client (card processor): https://sc-uat.thredd.net/
- PAN Finder: https://pf-uat.thredd.net/

5.1.2 CTS Access

CTS can be accessed online. CTS users can use the same credentials as are used to access Smart Client in UAT (provided that CTS has been enabled).

You can access CTS at the following link: https://cts-uat.globalprocessing.net:54340/

5.2 Connecting to Thredd Portal

You must first be set up on Cloudentity before you can access Thredd Portal.

Thredd Portal functions as a *Confidential Client*, where Thredd's own application infrastructure undertakes authentication and authorisation activity on behalf of the user.

The following is a summary of steps we run:

- Thredd adds an Admin user for your organisation to Cloudentity.
- We help to set up Single Sign-On SSO with your organisation.

Your Admin user is then able to perform the following:

- Add an organisation and assign roles to the organisation.
- Add Thredd Portal users to Cloudentity.
- Add roles in Cloudentity.

For more details, refer to Thredd Portal Guide: Getting Started.

Section 6: Appendices

You should read this section to understand the setup steps for EHI.

Topics covered in this section:

- Setting Up SSA

  Configuring SSO with Google (SAML)

  Configuring SSO with Okta (SAML)

  Configuring SSO with Okta (OIDC)

6.1 Generating and Obtaining a Software Statement Assertion (SSA)

Before using Postman, you need to generate an obtain a Software Statement Assertion (SSA) for your application, which you will use to connect to the REST APIs. You can use the following guidance to achieve this.

An SSA is a signed JWT token with information about the Application that is presented to Cloudentity when performing DCR (the last step before accessing the REST APIs).

Overview

- First, generate an SSA for your application in the Thredd CA dashboard.
- Once you have generated the SSA, obtain the SSA through either Postman or an API-based method for connecting to Thredd CA (as this page describes). For details of the steps in Postman, see Accessing the Cards API with mTLS.

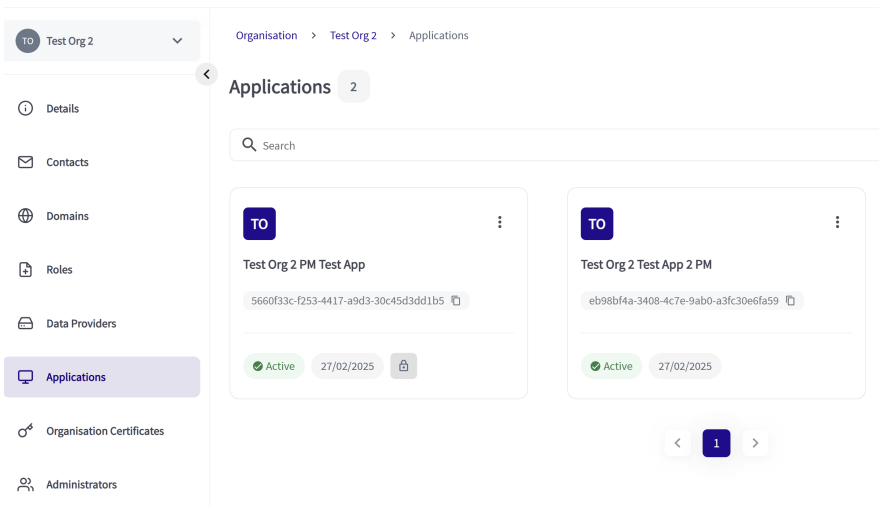
Prerequisites

- You must have completed Thredd's onboarding process, set up SSO and Cloudentity, and have an account in the Thredd CA dashboard. These steps are summarised in Setup Steps.
- You must have registered your application in Thredd CA and generated Transport Certificates for it. For a guide, see Creating Client Application Certificates for REST APIs.

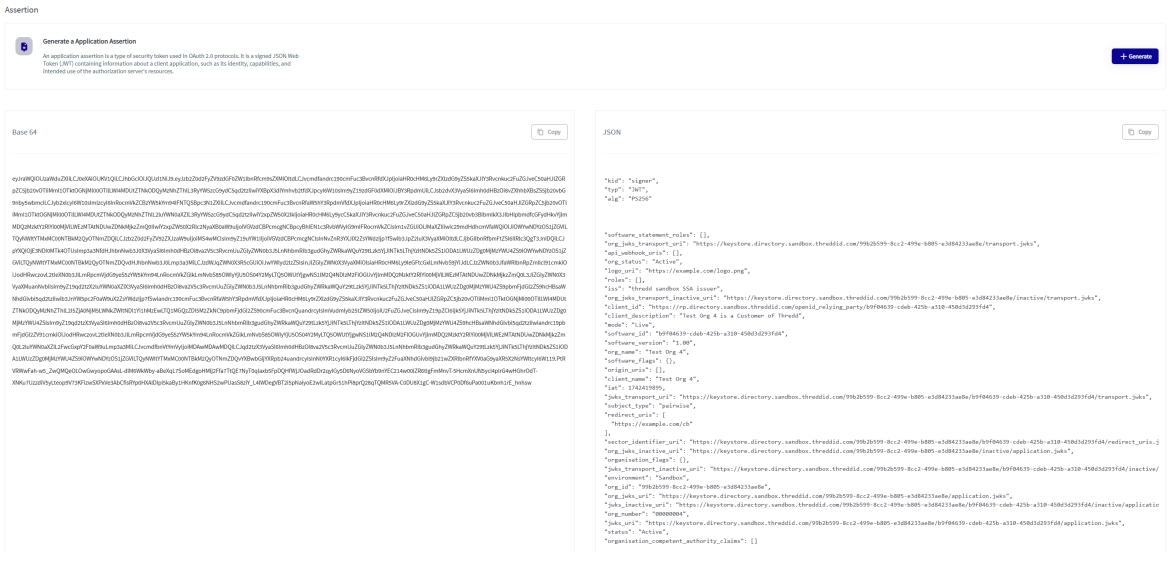
6.1.1 Generating an SSA in Thredd CA

You can generate an SSA for your Application under the **Applications > Application Assertion** section for your Organisation.

1. From the list of Organisations, click on the relevant tile.
2. Click **Applications**.



3. Select the Application that you want to generate an assertion for under **Applications**.
4. Click the **Assertion** tab.
5. Click the **Generate** button. A message appears followed by the Assertion.



6.1.2 Obtaining an SSA Using API Calls

You can use API calls from outside of Thredd CA in order to request the SSA, which you generated on Thredd CA. This includes the command for acquiring an access token for connecting to Thredd CA, and the command for requesting an SSA. The command for acquiring an access token executes a Client Credentials Grant, which initiates a request to Thredd CA over mTLS.

You need to have also created a Transport Certificate for accessing Thredd CA (see Creating Client Application Certificates for REST APIs for more details). You will need to prepare the Client ID and the SSA details.

Preparing the Client ID Details

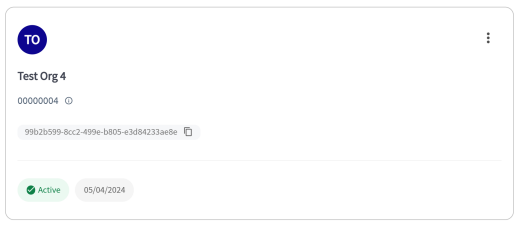
You will need to get the client_id URL which includes the URL of Thredd CA and the Client ID.

The following is an example:

client_id=https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4.

You can find the Client ID on the Organisations page in Thredd CA.

1. Log in to Thredd CA using these links:
    - Sandbox: https://web.directory.sandbox.threddid.com/
    - Production: https://web.directory.threddid.com/
2. In the Sign-in screen, enter your registered email address and click CloudEntity SSO.
3. Locate the organisation tile and copy the Client ID.



Preparing SSA Details

You need to prepare details of the SSA for the SSA URL.

The following is an example: client_id=https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4

This includes:

- **{{rc_mtls_base_url}}**: This is the variable for the Base URL for Thredd CA
- **{{org_id}}**: The Organisation ID
- **SSA ID**: This is the ID of the SSA, which is the same as the Application ID

Entering Commands to Obtain an SSA

Enter the following command for acquiring an access token. This includes the URL of the Thredd CA in location.

```
curl --location 'https://matls-auth.directory.sandbox.threddid.com/token' \
                --header 'Content-Type: application/x-www-form-urlencoded' \
                --data-urlencode 'scope=directory:software' \
                --data-urlencode 'client_id=https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-
a310-450d3d293fd4' \
                --data-urlencode 'grant_type=client_credentials'
```

Type in the following command to request an SSA. This includes the SSA URL.

```
curl --location 'https://matls-auth.directory.sandbox.threddid.com//organisations/99b2b599-8cc2-499e-b805-e3d84233ae8e/soft-
warestatements/b9f04639-cdeb
                -425b-a310-450d3d293fd4/assertion' \
            --header 'Authorization: Bearer <access_token>'
```

The response you see is as follows. This includes the SSA in Base 64 format and the metadata associated with the SSA.

```
{
  "client_name":"Test Org 4",
  "description":"",
  "client_uri":"",
  "logo_uri":"https://example.com/logo.png",
  "policy_uri":"",
  "tos_uri":"",
  "organisation_id":"",
```

```
  "client_id":"https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4",
  "application_type":"service",
  "application_types":[
    "service",
    "dcr"
  ],
  "redirect_uris":[
    "https://example.com/cb"
  ],
  "grant_types":[
    "client_credentials"
  ],
  "response_types":[
    "token"
  ],
  "scope":"3ds.read bulkcard.read bulkcard.write cards.encrypted cards.read cards.sensitive cards.write cvv.read cvv.write digit-
alchannel fraud.read fraud.write internal.read internal.write issuer.read issuer.write pin.read pin.write",
  "scopes":[
    "3ds.read",
    "bulkcard.read",
    "bulkcard.write",
    "cards.encrypted",
    "cards.read",
    "cards.sensitive",
    "cards.write",
    "cvv.read",
    "cvv.write",
    "digitalchannel",
    "fraud.read",
    "fraud.write",
    "internal.read",
    "internal.write",
    "issuer.read",
    "issuer.write",
    "pin.read",
    "pin.write"
  ],
  "audience":[
    "https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4"
  ],
  "token_endpoint_auth_method":"private_key_jwt",
  "revocation_endpoint_auth_method":"private_key_jwt",
  "introspection_endpoint_auth_method":"private_key_jwt",
  "token_exchange":{
    "actor_claims":null
  },
  "token_endpoint_auth_signing_alg":"",
  "jwks":{
    "keys":[
      ]
  },
  "jwks_uri":"https://keystore.directory.sandbox.threddid.com/99b2b599-8cc2-499e-b805-e3d84233ae8e/b9f04639-cdeb-425b-a310-
450d3d293fd4/application.jwks",
  "request_object_signing_alg":"RS256",
  "request_object_encryption_alg":"",
  "request_object_encryption_enc":"",
  "request_uris":[

  ],
  "client_id_issued_at":1723556609,
  "created_at":"2024-08-13T13:43:29.619067912Z",
  "updated_at":"2024-08-13T13:43:29.619067912Z",
  "client_secret_expires_at":0,
  "sector_identifier_uri":"https://keystore.directory.sandbox.threddid.com/99b2b599-8cc2-499e-b805-e3d84233ae8e/b9f04639-cdeb-
425b-a310-450d3d293fd4/redirect_uris.json",
  "userinfo_signed_response_alg":"none",
  "id_token_signed_response_alg":"ES256",
  "id_token_encrypted_response_alg":"",
  "id_token_encrypted_response_enc":"",
  "tls_client_certificate_bound_access_tokens":true,
```

```
  "tls_client_auth_subject_dn":"",
  "tls_client_auth_san_dns":"",
  "tls_client_auth_san_uri":"",
  "tls_client_auth_san_ip":"",
  "tls_client_auth_san_email":"",
  "privacy":{
    "scopes":null
  },
  "subject_type":"pairwise",
  "backchannel_token_delivery_mode":"",
  "backchannel_client_notification_endpoint":"",
  "backchannel_authentication_request_signing_alg":"",
  "backchannel_user_code_parameter":false,
  "require_pushed_authorization_requests":false,
  "authorization_signed_response_alg":"ES256",
  "authorization_encrypted_response_alg":"",
  "authorization_encrypted_response_enc":"",
  "dpop_bound_access_tokens":false,
  "authorization_details_types":null,
  "post_logout_redirect_uris":[

  ],
  "app_url":"",
  "backchannel_logout_uri":"",
  "backchannel_logout_session_required":false,
  "client_secret":"xT4SDSlIp9J3aLMIHMtQ99Uok6onqaQJEiQ0Hp2QBXc",
  "hashed_secret":"c2d-
c89f-
d3581d6222a7a143f173a4696feee7773f4156ada52c241eea25b30990f-
b35f2-
faa42105f1e51895d-
d7ced75f0847659f6396f-
bc977cf71d73-
bef7a1a7f93830e871ae9f-
b767318b0c545ee9543dc8ae94044cfc8a9b26bcf8bad53e3ba00ef889d1d8338dfdbdedd1aed382b51a86cf5f08d21d4cbf6f23d66141c50",
  "software_id":"b9f04639-cdeb-425b-a310-450d3d293fd4",
  "software_version":"1.00",
  "software_statement":"eyJraWQiOiJz-
aWduZXIiLCJ0eXAiOiJKV1QiLCJh-
bGciOiJQUzI1NiJ9.eyJzb2Z0d2FyZV9zdGF0ZW1lb-
nRfcm9sZXMiOlt-
dLCJvcm-
dfandrc190cmFuc3BvcnRf-
dXJpI-
joi-
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL3RyYW5zcG9y-
dC5qd2tzIi-
wiYXBpX3dlYm-
hvb2t-
fdXJp-
cyI6W10sIm9yZ19zdGF0dXMiOiJBY3Rp-
dmUiLCJs-
b2d-
vX3Vy-
aSI6Im-
h0dHBzOi8vZXh-
hbXBsZS5jb20vbG9nby5wbm-
ciLCJy-
b2xl-
cyI6W10sIm-
lzcyI6InRocmVkZCBzYW5kYm94IFNTQSBp-
c3N1ZXIiLCJvcm-
dfandrc190cmFuc3BvcnR-
faW5hY3Rp-
dmVf-
dXJpI-
```

```
joi-
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL2luYWN0aXZlL3RyYW5zcG9y-
dC5qd2tzIi-
wiY2x-
pZW50X2lkI-
joi-
aHR0cHM6Ly9ycC5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vb3Bl-
bmlkX3Jl-
bHlpb-
mdfcGFy-
dHk-
vYjlmMDQ2Mzk-
tY2RlYi00MjViLWEzMTAtNDUwZDNkMjkzZmQ0Ii-
wiY2x-
pZW50X2Rl-
c2Ny-
aXB0aW9uI-
joiVGVzdCBPcm-
cgNCBp-
cyBhIEN1c3RvbWVyIG9mIFRocmVkZCIsIm1vZGUiOiJMaXZlIi-
wic29m-
dHd-
hcmV-
faWQiOiJiOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQiLCJzb2Z0d2FyZV92ZXJz-
aW9uI-
joiMS4wMCIsIm9yZ19uYW1lI-
joiVGVzdCBPcm-
cgNCIsInNvZnR3YXJlX2ZsYWdzIjp7fSwib3JpZ2luX3Vy-
aXMiOlt-
dLCJjbGllb-
nRf-
bmFtZSI6IlRl-
c3QgT3JnIDQiLCJpYXQiOjE3MjM1NTYzNTAsIm-
p3a3Nf-
dHJhb-
nNw-
b3J0X3Vy-
aSI6Im-
h0dHBzOi8va2V5c3RvcmUuZGlyZWN0b3J5LnNhb-
mRib3gudGhyZWRkaWQuY29tLzk5YjJiNTk5LThjYzItNDk5ZS1iODA1LWUzZDg0MjMzYWU4ZS9iOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQvdHJhb-
nNw-
b3J0Lm-
p3a3MiLCJvcm-
dhbm-
lzYXRp-
b25f-
dGFncyI6W10sInN1Ym-
plY3Rf-
dHl-
wZSI6InBhaXJ3aXNlIi-
wicmVkaXJlY3Rf-
dXJp-
cyI6WyJodHRw-
czovL2V4YW1w-
bGUuY29tL2NiIl0sInNlY3Rvcl9pZGVudGl-
maWVyX3Vy-
aSI6Im-
h0dHBzOi8va2V5c3RvcmUuZGlyZWN0b3J5LnNhb-
mRib3gudGhyZWRkaWQuY29tLzk5YjJiNTk5LThjYzItNDk5ZS1iODA1LWUzZDg0MjMzYWU4ZS9iOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQvcmVkaXJ-
lY3Rf-
dXJpcy5qc29uIi-
wib3JnX2p3a3N-
faW5hY3Rp-
dmVf-
dXJpI-
joi-
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL2luYWN0aXZlL2Fw-
cGx-
pY2F0aW9uLm-
p3a3MiLCJvcm-
dhbm-
```

```
hZThlL2Fw-
cGx-
pY2F0aW9uLm-
p3a3MiLCJqd2tzX2luYWN0aXZlX3Vy-
aSI6Im-
h0dHBzOi8va2V5c3RvcmUuZGlyZWN0b3J5LnNhb-
mRib3gudGhyZWRkaWQuY29tLzk5YjJiNTk5LThjYzItNDk5ZS1iODA1LWUzZDg0MjMzYWU4ZS9iOWYwNDYzOS1jZGViLTQyNWItYTMxMC00NTBkM2QyOTNmZDQvaW5hY3R-
pdmUvYXBw-
bGljYXRp-
b24uandrcyIsIm9yZ19udW1iZXIiOiIwMDAwMDAwNCIsIm-
p3a3Nf-
dXJpI-
joi-
aHR0cHM6Ly9rZXlzdG9yZS5kaXJlY3Rvcnkuc2FuZGJveC50aHJlZGRpZC5jb20vOTliMmI1OTk-
tOGNjMi00OTllLWI4MDUtZTNkODQyMzNhZThlL2I5ZjA0NjM5LWNkZWItNDI1Yi1hMzEwLTQ1MGQzZDI5M2ZkNC9h-
cHBsaWNhdGlvbi5qd2tzIiwic3RhdHVzIjoiQWN0aXZlIiwib3JnYW5pc2F0aW9uX2NvbXBldGVudF9hdXRob3JpdHlfY2xhaW1zIjpbXX0.N5wPwnVFcramX7BAYOWn_
1tWcLxSRNaIxst-ZK_p29he5mN1_hdFF6BoVMG0Chh-dV_FL4-luQlg7Qn3EMRCb3RWp8U7pcj0lmFSDcUyvlQpoB5hqrXgjPEOkIC7uqJuTDjwakj0NFtJDtK_l_FVp-
PA2ZsDJanwCxsnxxMxVFlxFPwCYk8jBTNE84zx092a4-Tj4VfE4e5S5HA7if8sR7PiXvAMWs1jBrUQ9enWacfu_Xno_-kywtrkAeR4fi-MGNeA4Ik-
bYbh8LyhMjg3XXun87BoL8-b_An5fX5y-XS8VnENg3NFU3D8soQ05OKlXnIDT2Dp7v1lFlNYrV1jO9w",
  "dynamically_registered":true,
  "registration_access_token":"PKW64FAixrjhxewTH9lR26o7m6_GvbL6RKWv0x5mIM.dWPLOrdXw7yGu2R1HAWlx8dSr3Jf-thb_zNFEGSOUNg",
  "registration_client_uri":"https://auth.uat.threddid.com/confidential-cli-
ents/oauth2/register/https://rp.directory.sandbox.threddid.com/openid_relying_party/b9f04639-cdeb-425b-a310-450d3d293fd4",
  "registration_access_token_expires_in":0
}
```

6.2 Configuring SSO with Okta (SAML)

If your organisation uses Okta as an IdP, you can use Okta for configuring SSO for accessing various Thredd services, for example, Thredd Portal. This page describes the steps for using the 2.0 version of Security Assertion Markup Language (SAML) protocol for setting up SSO.

As a client, you would already have an account on the Okta Administration Console.

Note: Setting up SSO is not mandatory, but is recommended.

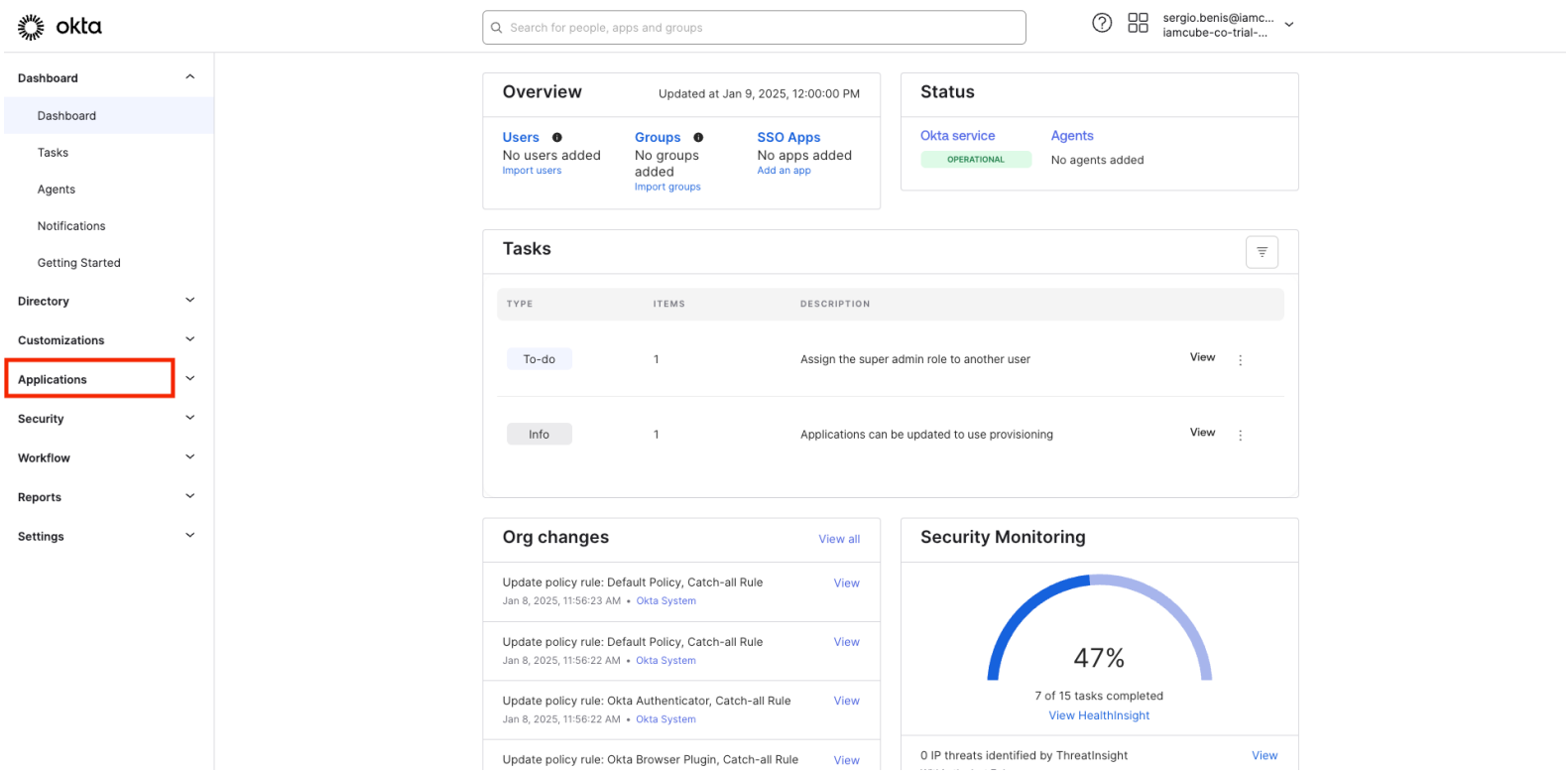
6.2.1 Summary of Steps

The steps involve:

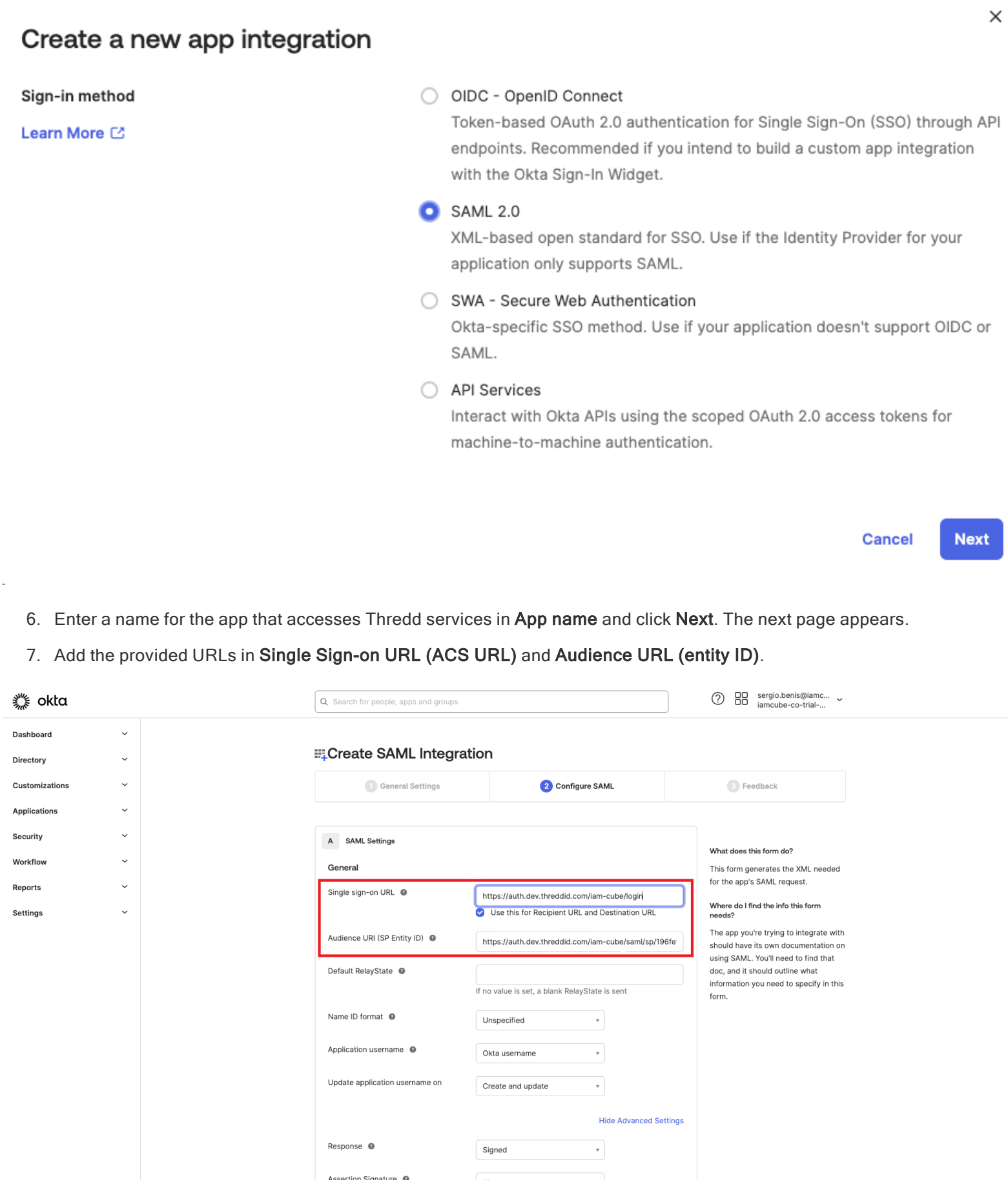
- Creating a SAML app for your SSO connection to Thredd services.
- Adding configurations from Thredd including the SSO URLs and the Entity ID.
- Mapping fields associated with the users defined by Okta with those used by your app.
- Sharing the Metadata URL with Thredd.

6.2.2 Configure SSO

1. Log in to the Okta Administration console.
2. From the left-hand menu, select **Applications**.

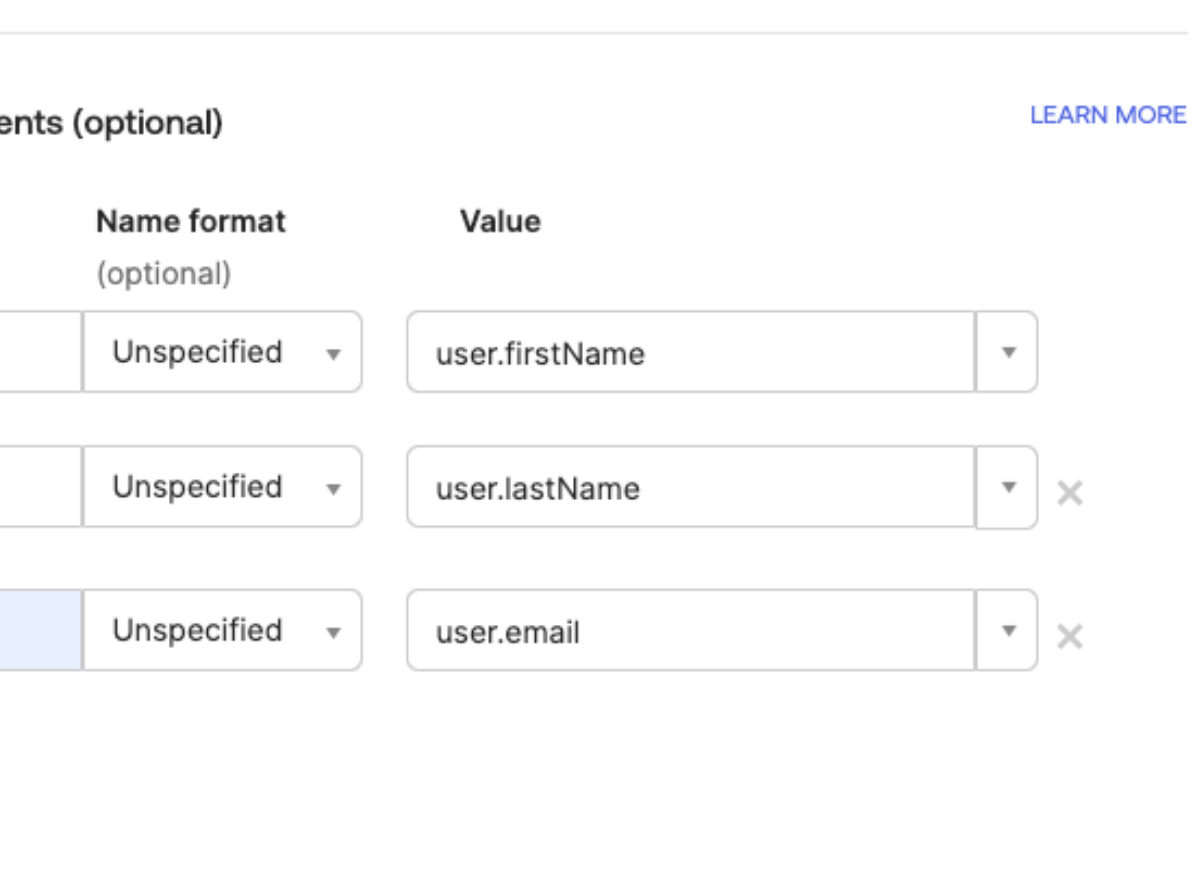


3. Click on **Create Application**.
4. Select **SAML 2.0** and click **Next**. The next page appears.



6. Enter a name for the app that accesses Thredd services in **App name** and click **Next**. The next page appears.
7. Add the provided URLs in **Single Sign-on URL (ACS URL)** and **Audience URL (entity ID)**.

8. Scroll down on the same page and configure the following Attribute Statements:
    a. Enter an attribute name in the **Name** column.
    b. Select a value in the **Value** column.
    c. To add another entry, click the **Add Another** button.
    d. Repeat steps a and b.



9. Click **Next**.
10. In the displayed page, select **This is an internal app that we created** and click **Finish**.
11. In the displayed **Metadata Data** details that appear, share the Metadata URL with Thredd.

You can then assign the application to the users or groups who will be using the Thredd services.

6.3 Configuring SSO with Google (SAML)

If your organisation uses Google, you can configure Google as an IdP provider to provide SSO access to various Thredd services. For example, you can use SSO to access Thredd Services, such as Thredd Portal. This page describes the steps for using the 2.0 version of Security Assertion Markup Language (SAML) protocol for setting up SSO.

As a client, you would already have an account on the Google Admin Console.

Note: Setting up SSO is not mandatory, but is recommended.

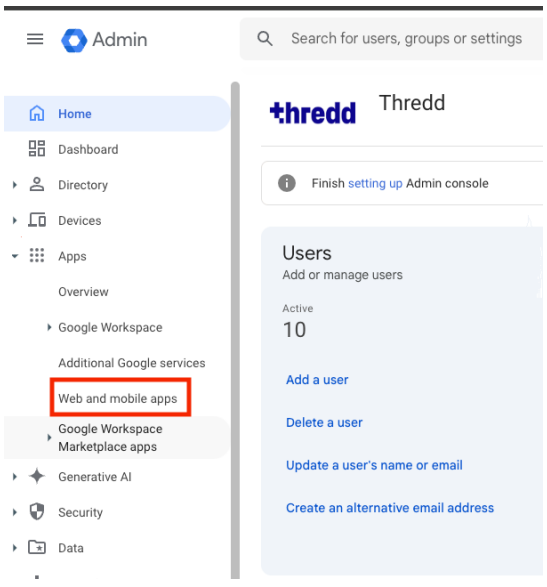
6.3.1 Summary of Steps

The steps involve:

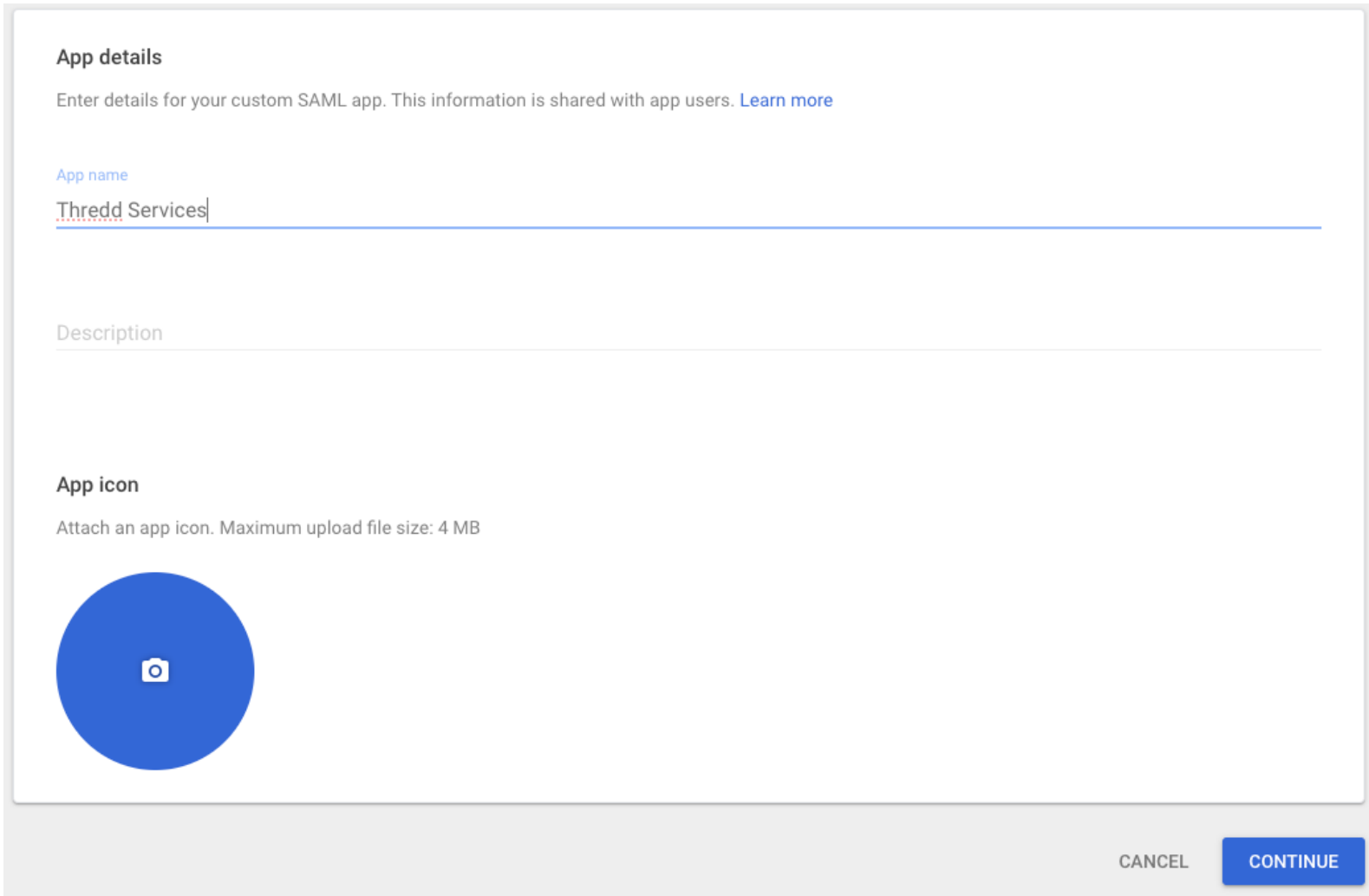
- Creating a SAML app for your SSO connection to Thredd services.
- Choosing either to download IdP metadata or to add configurations from Thredd. If you add configurations from Thredd, you include the SSO URL and the Entity ID.
- Mapping fields associated with the users defined by Google to those used by your app.
- Assigning access permission on your app.

6.3.2 Configuring SSO

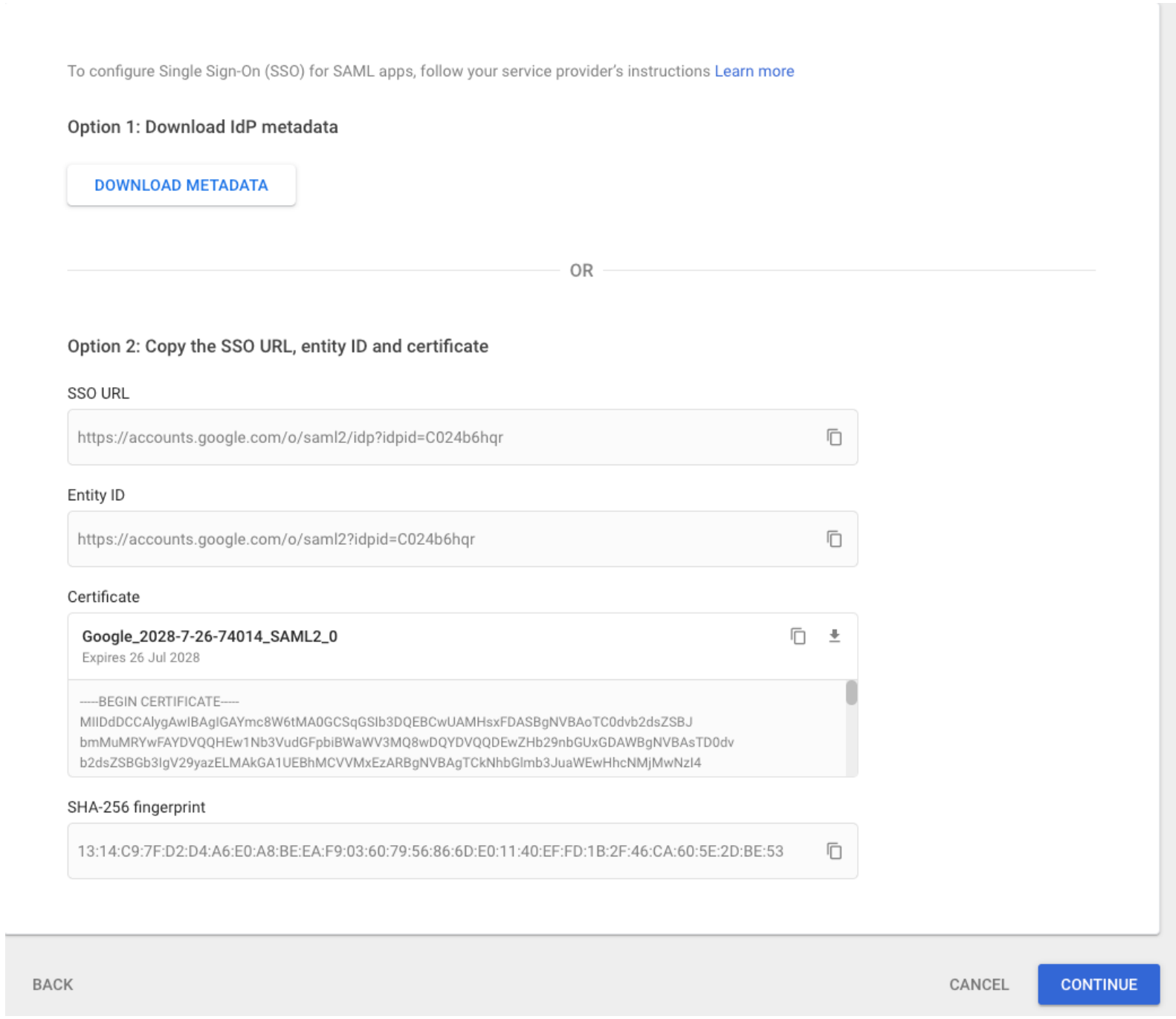
1. Log in to the Google Admin console.
2. Select **Apps** > **Web and mobile apps**.



3. Click on **Add app** and select **Add custom SAML app**.
4. Enter a name for the app that accesses Thredd services in **App name** and click **Continue**. The next page appears.

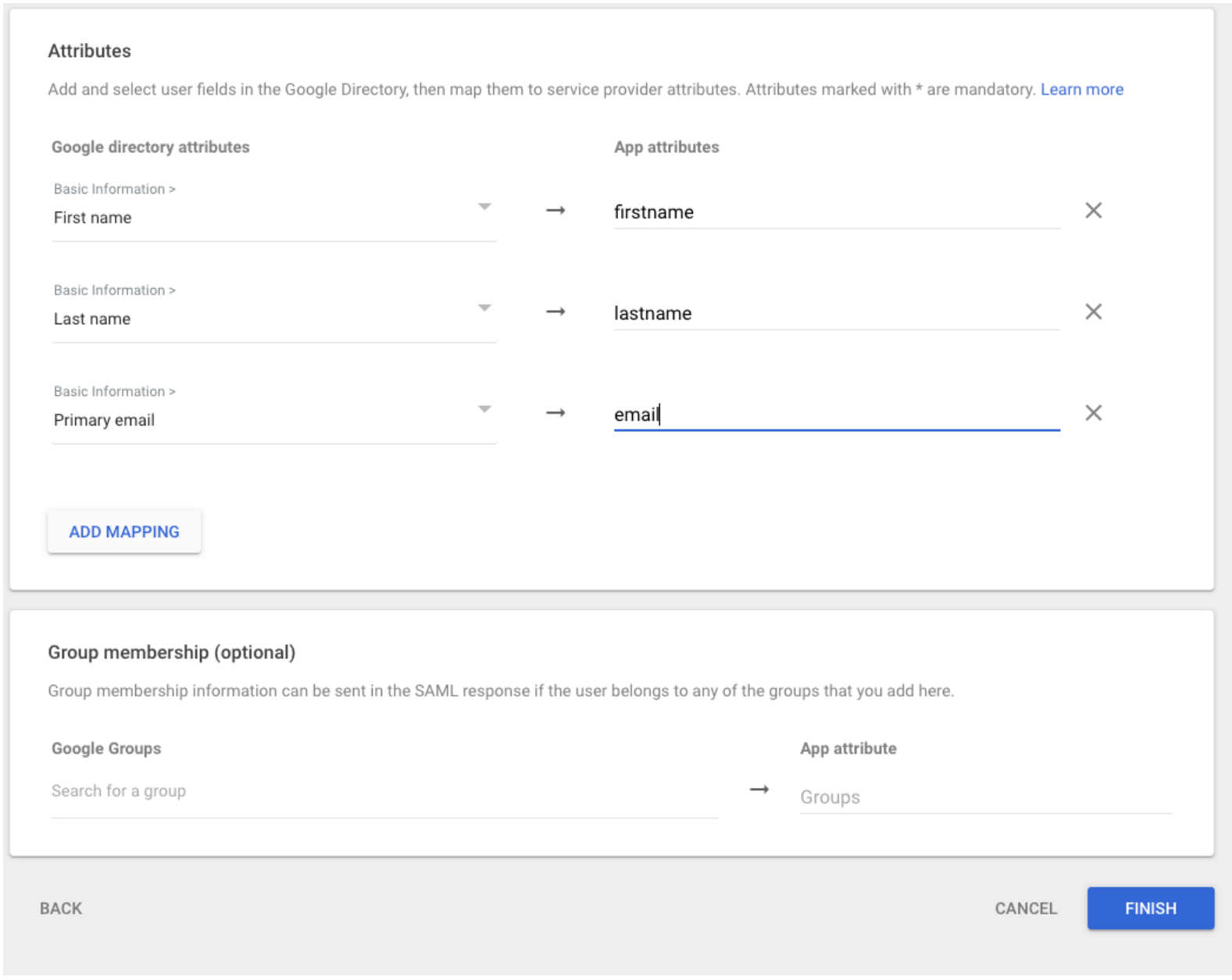


5. To download the metadata, click **Download Metadata** and save the file. Then share the file with Thredd. Once you have completed this, go to step 7.
6. To include entity ID and URL details:
    a. Add the URL in **SSO URL**.
    b. Add in the Entity ID in **Entity ID**. A certificate and a SHA-256 fingerprint appear. These are generated automatically on the console.



7. Click **Continue**.
8. Configure attribute mapping.
    a. Click the **Add Mapping** button.
    b. Select a group in **Google directory attributes** and choose a group in **App attributes**.
    c. To add another entry, click the **Add Mapping** button again and repeat the step for choosing both attributes.

The following shows an example.

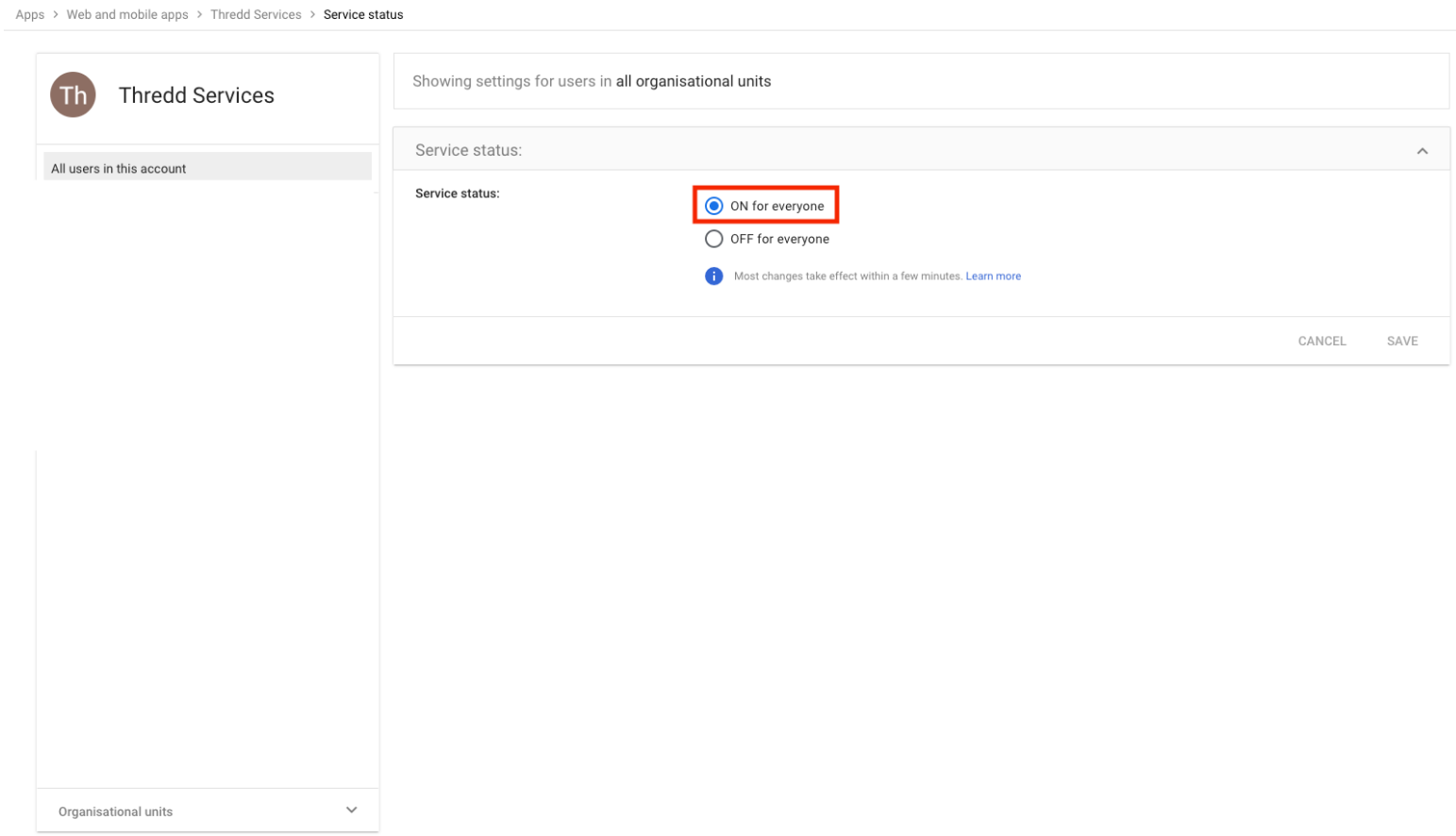


9. Click **Finish**.
10. Once completed, select access permission options (see the following procedure).

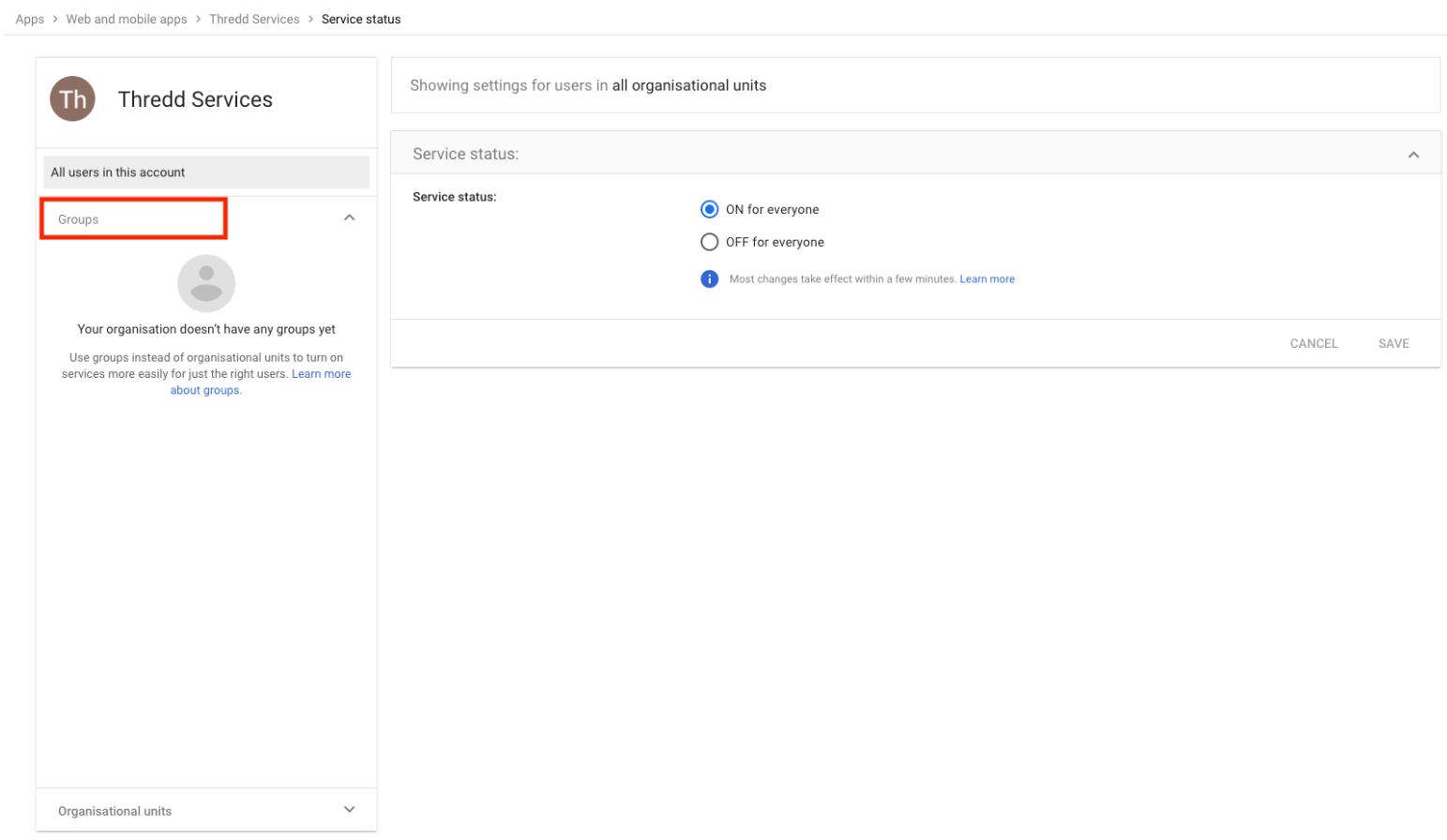
Setting Access Permission Options

You can set access permission options for the app based on anyone who holds a Google account, membership of specific Google groups, and organisational units. An organisational unit is a named organisation within Google.

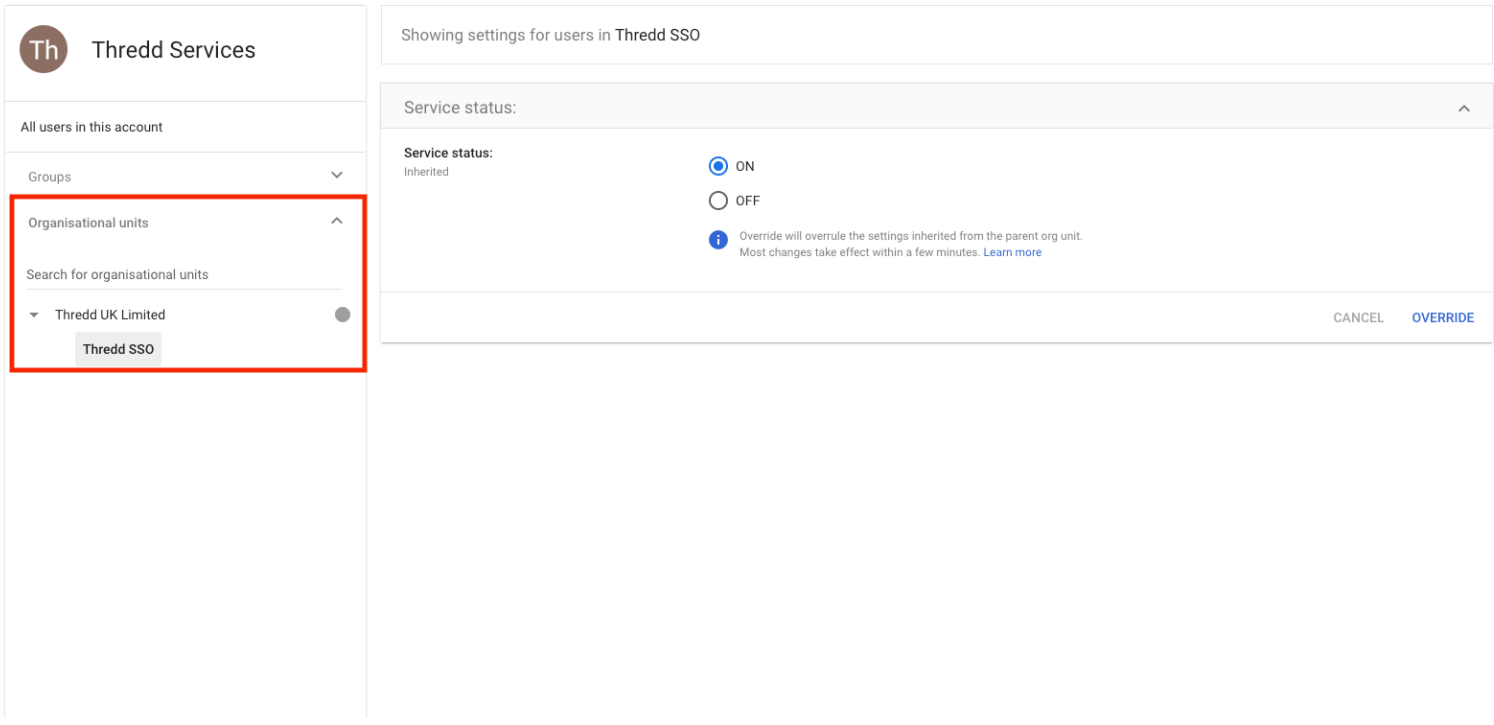
1. To provide access to anyone who holds a corporate Google account, select **All users in this account** on the left hand menu. Then choose **ON for everyone** in the main screen.



2. To provide access to members of a selected group:
    a. Select **Groups** on the left hand menu.
    b. Select a group.
    c. Select **ON for everyone** in the main screen.



3. To provide access to members of specific Organisational units:
    a. Select **Organisational units** on the left hand menu.
    b. Select an organisational unit.
    c. Select **ON for everyone** in the main screen.


Th
Thredd Services
All users in this account
Groups
Organisational units
Search for organisational units
Thredd UK Limited
Thredd SSO
Showing settings for users in Thredd SSO
Service status:
Service status:
Inherited
ON
OFF
Override will overrule the settings inherited from the parent org unit.
Most changes take effect within a few minutes. Learn more
CANCEL
OVERRIDE

6.4 Configuring SSO with Okta (OIDC)

If your organisation uses Okta, you can configure Okta as an IdP provider to provide SSO access to various Thredd services. For example, you can use SSO to access Thredd Services such as Thredd Portal. This page describes the steps for using the OpenID Connect (OIDC) protocol for setting up SSO.

Note: Setting up SSO is not mandatory, but is recommended.

6.4.1 Overview

The steps involve:

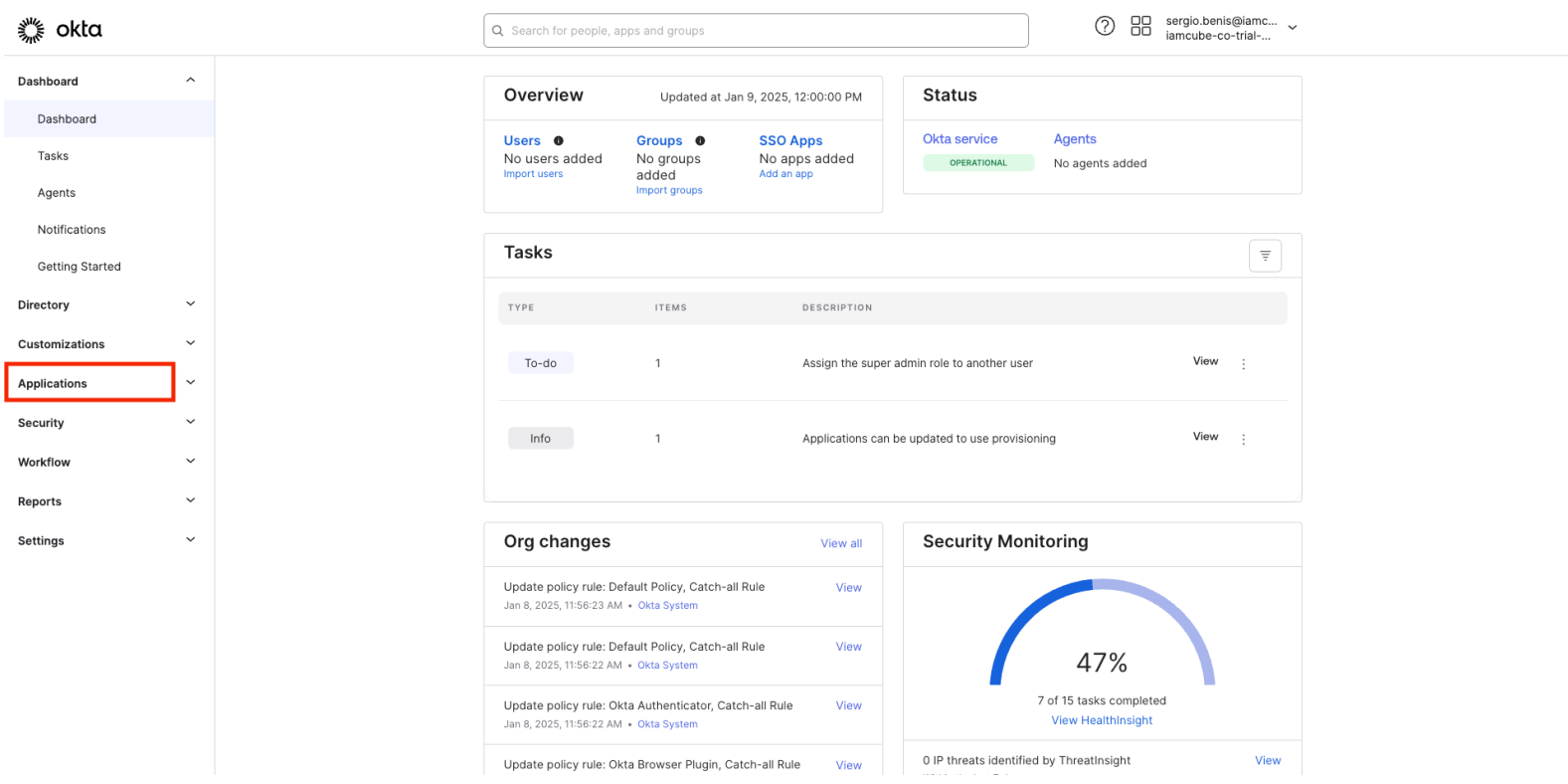
- Creating an app and app integration.
- Setting URL and refresh token settings.
- Specifying your access control requirement.
- Sharing authentication details with Thredd, through either the Client ID/Client Secret method or the private_key_jwt authentication method.

  Note: Thredd recommends using the private_key_jwt authentication method.

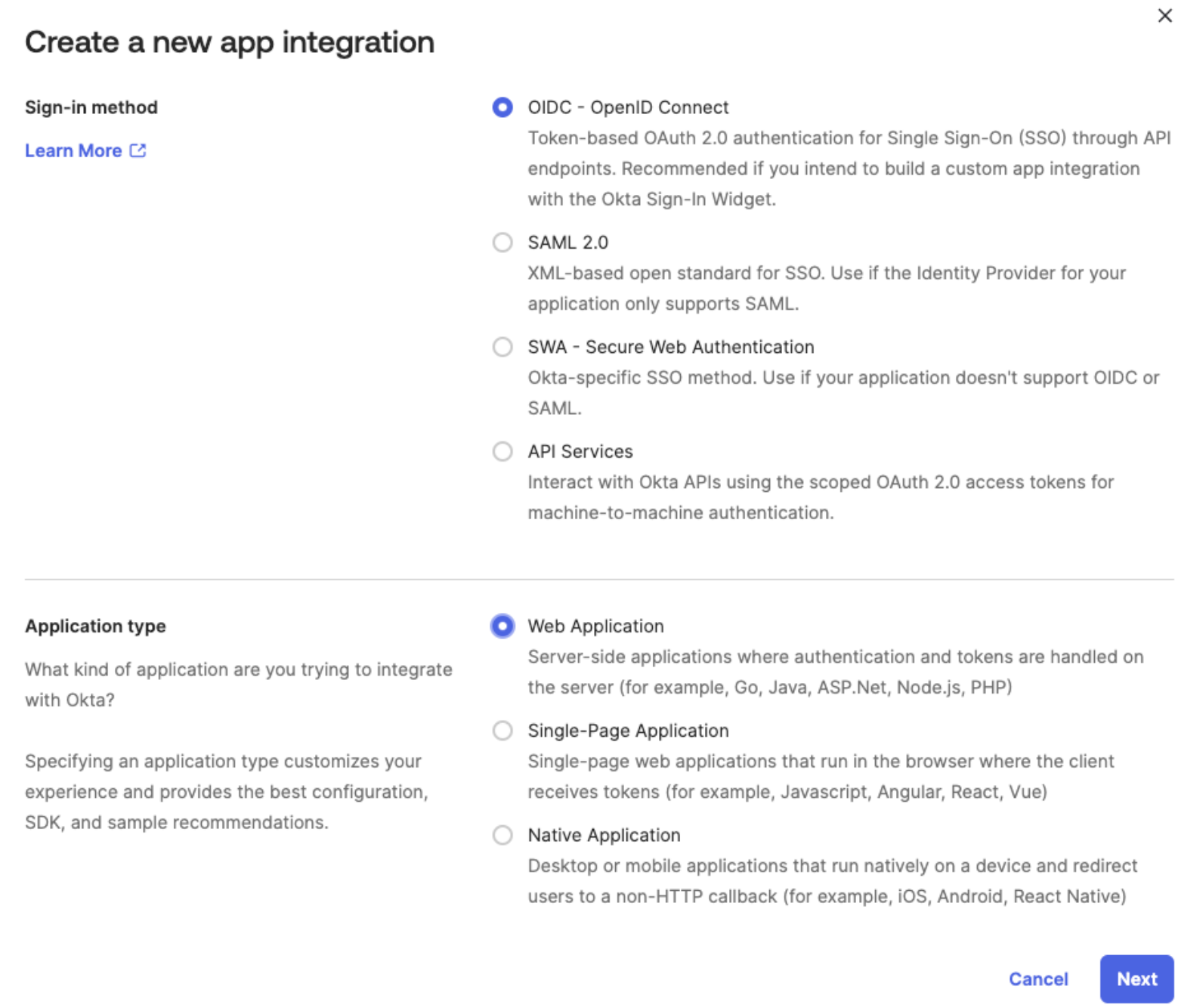
Thredd will provide you with a Sign-in Redirect URI for creating a web application integration.

6.4.2 Configure SSO

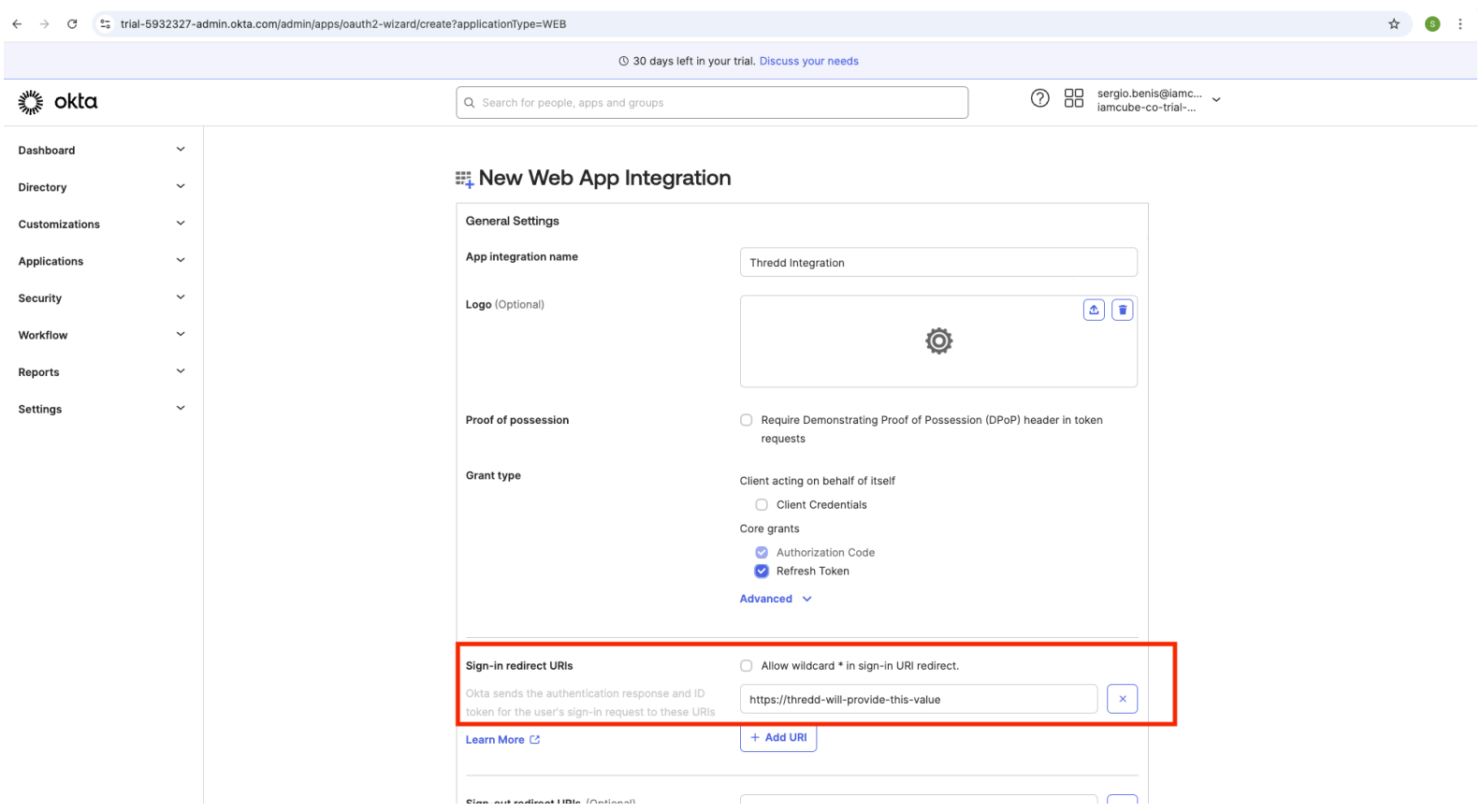
1. Log in to the Okta Administration console.
2. Select **Applications** from the left-hand menu.



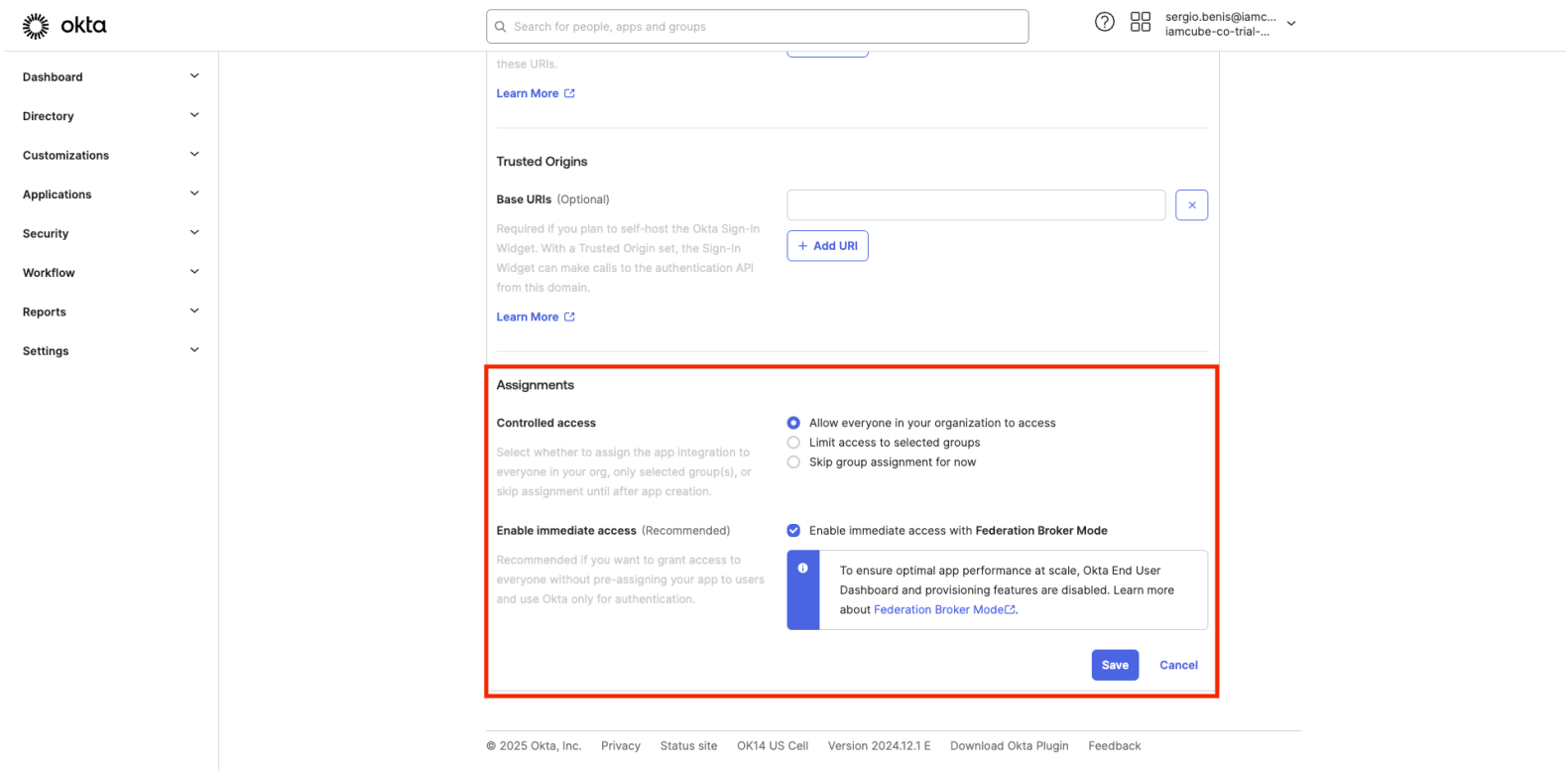
3. Click **Create Application**.
4. In Create a new application integration, select **OIDC - OpenID Connect** and **Web Application.**



5. Click **Next.**
6. Enter a name for your application integration in **Application integration name**.
7. Select the **Refresh Token** check box.
8. Enter the URL value of the sign-in redirect URIs in the **Sign-in redirect URIs** section.



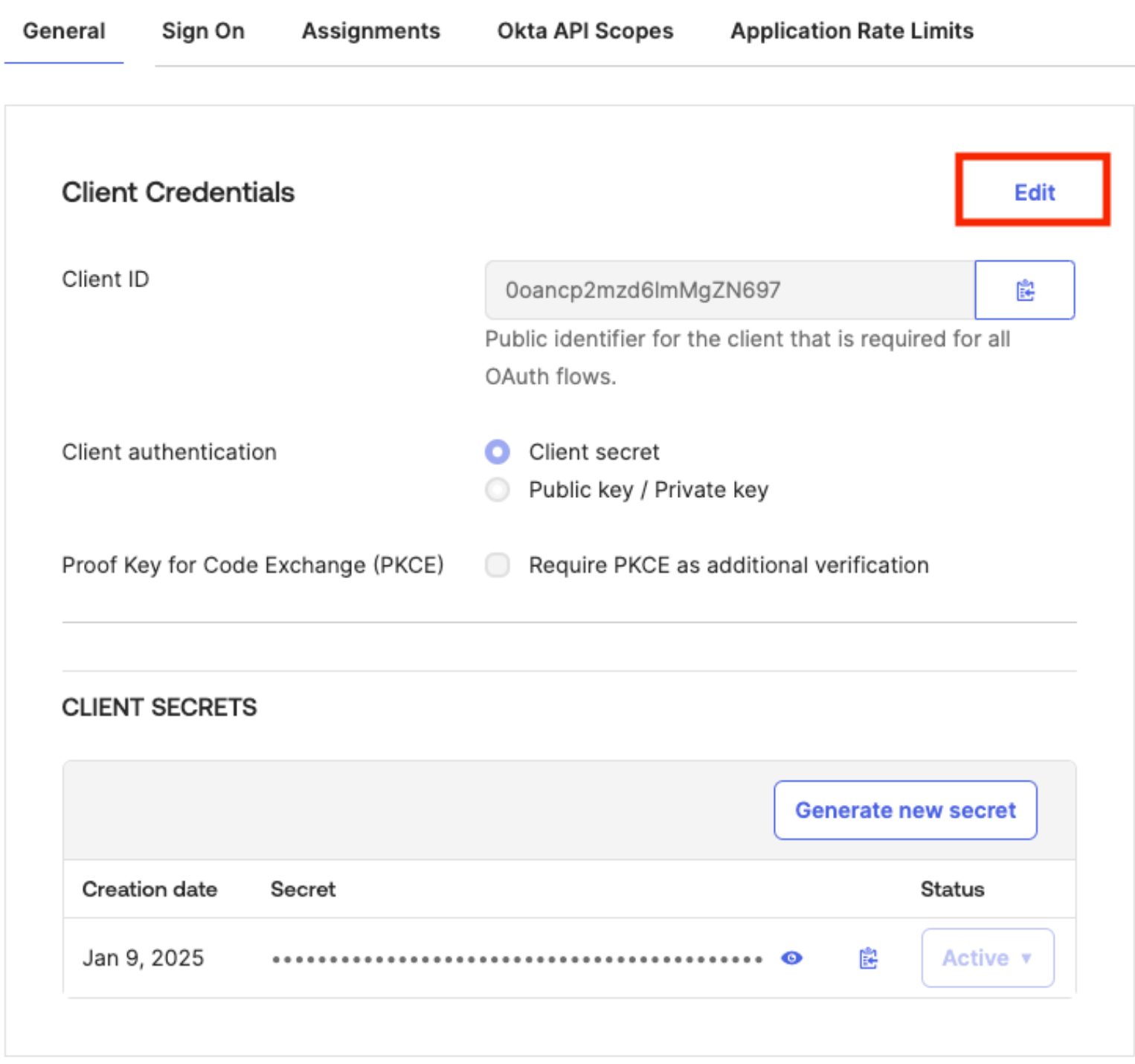
9. Select how you want to control access to Thredd applications from your organisation. The example below shows that access is available to all users in your organisation, and where there is immediate access.



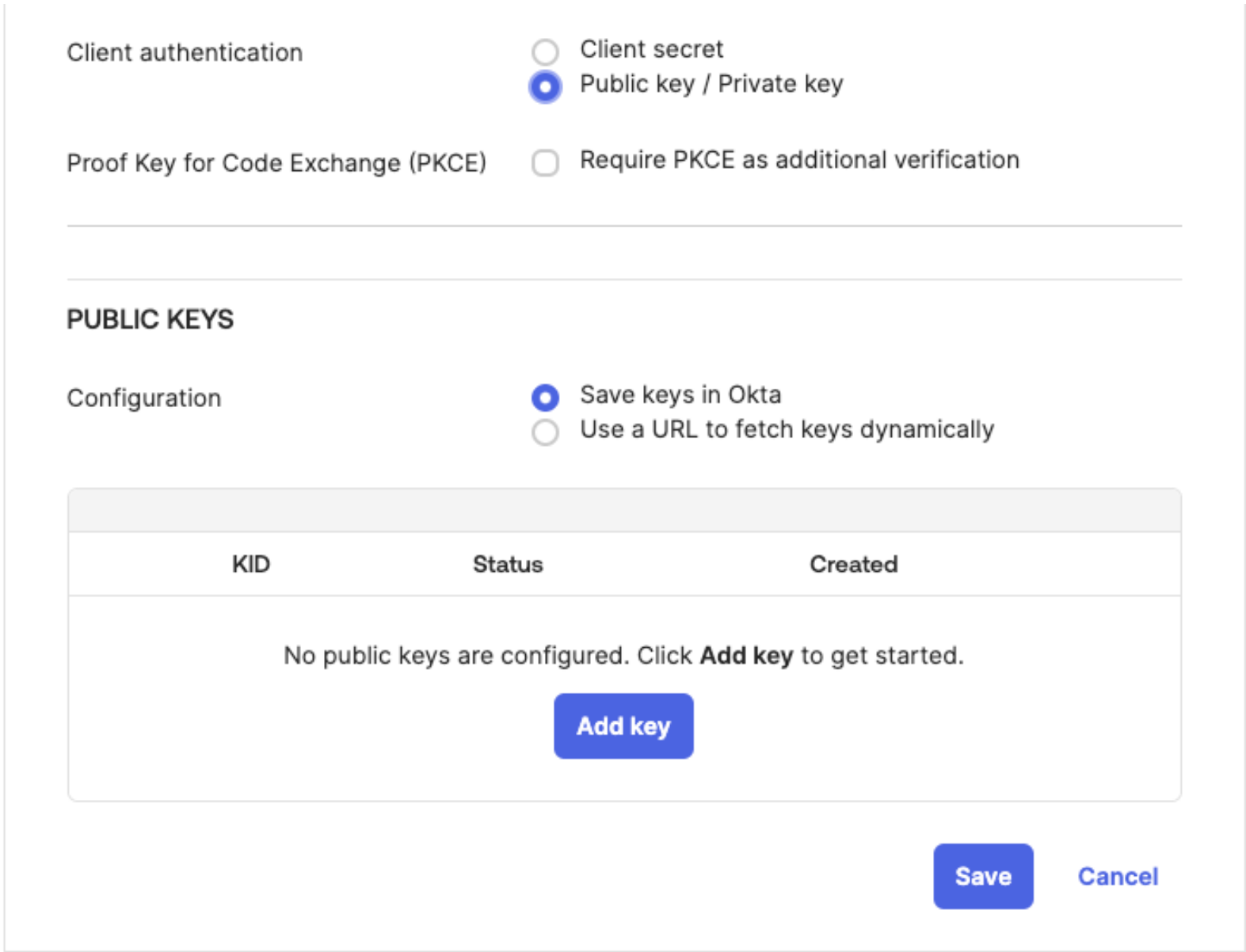
10. Share details with Thredd.
    - If you are using a Client ID/Client secret, share this detail with Thredd using your preferred secure method.
    - If you prefer Thredd's recommended authetication method of `private_key_jwt`, perform the steps below for sharing the private key.

Share the Private Key `private_key_jwt`

1. Select the application that you have just created under **General**.
2. Click **Edit** next to **Client Credentials**.

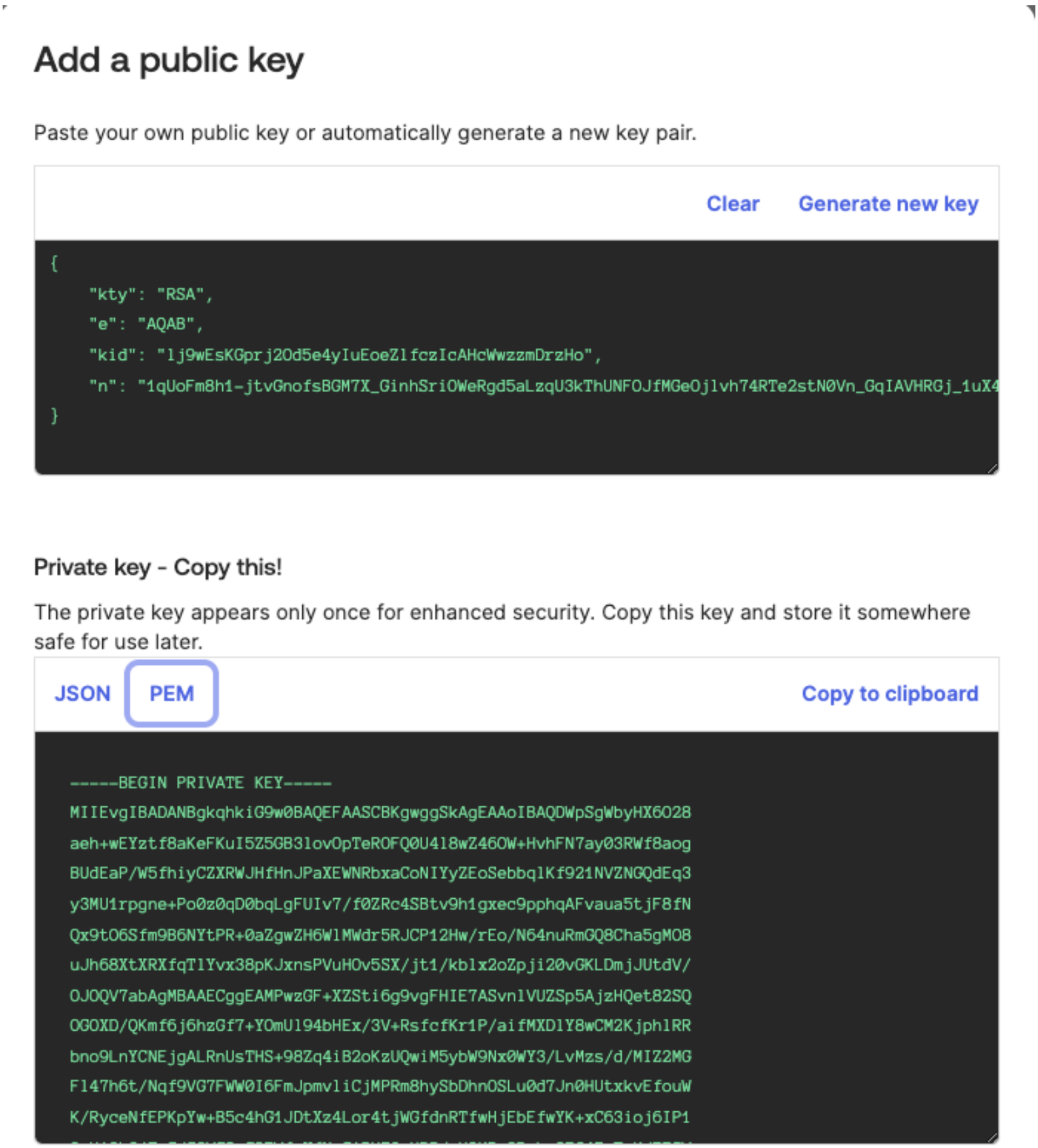


3. Select **Public Key / Private Key**, the page updates to show the public key configuration options:



4. Click **Add key**. The **Add a public key** screen appears.
5. Select **Generate new key**.
6. When the private key is shown, select PEM.
7. Copy the value to your clipboard and save it.

You will need to share the private key with Thredd. The following shows the on-screen instructions that appear for generating and copying a private key.


Add a public key
Paste your own public key or automatically generate a new key pair.
Clear
Generate new key
{
"kty": "RSA",
"e": "AQAB",
"kid": "lj9wEsKGprj2Od5e4yIuEoeZlfczIcAHcWwzzmDrzHo",
"n": "1qUoFm8h1-jtvGnofsBGM7X_GinhSriOWeRgd5aLzqU3kThUNFOJfMGeOjlvh74RTe2stN0Vn_GqIAVHRGj_1uX4
}
Private key - Copy this!
The private key appears only once for enhanced security. Copy this key and store it somewhere safe for use later.
JSON
PEM
Copy to clipboard
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDWpSgWbyHX6O28
aeh+wEYztf8aKeFKuI5Z5GB3lovOpTeROFQ0U4l8wZ46OW+HvhFN7ay03RWf8aog
BUdEaP/W5fhiyCZXRWJHfHnJPaXEWNRbxaCoNIYyZEoSebbqlKf921NVZNGQdEq3
y3MU1rpgne+Po0z0qD0bqLgFUIv7/f0ZRc4SBtv9h1gxec9pphqAFvaua5tjF8fN
Qx9tO6Sfm9B6NYtPR+0aZgwZH6WlMWdr5RJCP12Hw/rEo/N64nuRmGQ8Cha5gMO8
uJh68XtXRXfqTlYvx38pKJxnsPVuHOv5SX/jt1/kblx2oZpji20vGKLDmjJUtdV/
OJOQV7abAgMBAAECggEAMPwzGF+XZSti6g9vgFHIE7ASvnlVUZSp5AjzHQet82SQ
OGOXD/QKmf6j6hzGf7+YOmUl94bHEx/3V+RsfcfKr1P/aifMXDlY8wCM2KjphlRR
bno9LnYCNEjgALRnUsTHS+98Zq4iB2oKzUQwiM5ybW9Nx0WY3/LvMzs/d/MIZ2MG
Fl47h6t/Nqf9VG7FWW0I6FmJpmvliCjMPRm8hySbDhnOSLu0d7Jn0HUtxkvEfouW
K/RyceNfEPKpYw+B5c4hG1JDtXz4Lor4tjWGfdnRTfwHjEbEfwYK+xC63ioj6IP1
Done
Cancel

8. Click **Done**.

Scopes

The following is a list of scopes that can be returned when using the DCR Endpoint.

Scope Name	Description
cards.read	Enables you to use GET endpoints to return information on cards.
cards.write	Enables you to use the POST, UPDATE and PATCH endpoints to create and update card information.
pin.read	Enables you to use the Retrieve PIN endpoint.
pin.write	Enables you to use the Set PIN and Unblock PIN endpoint.
cvv.read	Enables you to use the Retrieve CVV endpoint.
cvv.write	Enables you to use the Unblock CVV endpoint.
bulkcard.read	Enables you to use the Get Bulk Card Progress endpoint.
bulkcard.write	Enables you to use the Create Bulk Card endpoint.
cards.encrypted	Enables you to use the Get Encrypted Data endpoint.
cards.sensitive	Enables you to use the Get Full PAN endpoint.
3ds.read	Enables you to use the Get 3DS Configuration endpoint.
ads.read	Enables you to use the GET endpoints for Visa Alias Directory.
ads.write	Enables you to use the POST, DELETE and PUT endpoints for Visa Alias Directory.

General FAQs

Below is a detailed FAQ focusing on the Secure Connectivity Framework and its components, including mTLS, Cloudentity, Thredd CA, and related topics.

Overview

What is the Secure Connectivity Framework?

The Secure Connectivity Framework is a combination of components that enable secure access to Thredd's resources using a common identity store. These include Cloudentity, Thredd CA, and mTLS termination. It ensures secure communication and access control through features like mTLS, OAuth, and certificate-based authentication.

What is mTLS and how is it used, in the Secure Connectivity Framework?

mTLS (Mutual Transport Layer Security) is a security protocol used to establish trust between clients and servers. In the Secure Connectivity Framework:

- Transport Certificates issued by Thredd CA are used to establish secure connections.
- mTLS Termination requires on-premise infrastructure to establish Trust Chains, ensuring that certificates originate from legitimate sources.
- It is used in the External Host Interface (EHI) setup, and works in the background for the SOAP and REST API setups.

What is the role of Cloudentity in the Secure Connectivity Framework?

Cloudentity is a Software as a Service (SaaS) capability that acts as:

- **Identity Provider (IDP):** For Thredd's interfaces, including Thredd CA and Thredd Portal.
- **OAuth OpenID Provider (OP):** For registering and managing customer applications, generating and validating access tokens, and enforcing access control policies.
- **Policy Decision Point (PDP):** To allow or deny access to protected Thredd resources based on policies that check attributes like Access Token Claims, mTLS Certificates, and User Roles.

Can I still use or request a VPN setup?

No, VPN setups are not supported. You must connect securely to Thredd using the Secure Connectivity Framework.

Certificates

What is Thredd CA, and what certificates does it provide?

Thredd CA is Thredd's Certificate Authority responsible for creating and managing certificates. It provides:

- **Transport Certificates:** For secure connections between resources.
- **Signing Certificates:** For private_key_jwt authentication, signed messages, and non-repudiation.
- **Root and Issuing Certificates:** Used to verify Transport Certificates in EHI setups.

What certificates are required for different Thredd applications?

The certificates required for various Thredd applications are as follows:

- **REST API:** Requires both Transport and Signing Certificates.
- **SOAP API:** Requires only Transport Certificates.
- **External Host Interface (EHI):** Requires Root and Issuing Certificates.
- **Thredd Portal:** Pre-installed Transport Certificates.
- **Smart Client:** Bundled Transport Certificates in the installer.

Single Sign-On

How does SSO work in the Secure Connectivity Framework?

SSO capabilities are provided through Cloudentity, allowing federated authentication. This enables users to authenticate once and gain access to multiple Thredd resources without needing to log in separately for each service. You can also configure your own IdP provider for SSO.

Dynamic Client Registration (DCR)

What is DCR?

DCR is a process supported by Cloudentity for registering client applications dynamically. It simplifies the onboarding of new applications by automating the registration process.

External Host Interface

How do I set up mTLS on EHI?

Step 1: Set up Certificates

You need to configure both server and client certificates to enable mTLS.

1. Set up Server Certificates:
    a. Obtain a Server Certificate from a trusted Certificate Authority (CA) vendor, such as Verizon, Digicert, or Amazon Web Services.
    b. Install the following certificates on your EHI listening endpoint:
        - Server or Leaf Certificate: Issued to individual servers by a CA. This certificate is the "leaf" of the hierarchical tree of trust.
        - Intermediate or Issuing Certificate: Represents the CA that issued the server certificate.
        - Root Certificate: The trusted root of the CA chain.
2. Set up Client Certificates. Thredd will present its Transport Certificate during the connection handshake. Your system must validate this certificate against the CA chain of trust (Root and Issuing Certificates).

Step 2: Test the mTLS Connection

1. Test the connection to ensure the mTLS handshake is successful.
2. Verify that your system can receive and process EHI messages securely.

Step 3: Add Optional Security Enhancements

To further secure your mTLS setup, consider implementing the following:

1. IP Address Allow List: Add Thredd's fixed IP address to your allow list to ensure only authorized traffic is accepted.
2. Certificate Pinning: Implement certificate pinning to block requests made with incorrect certificates.

Postman Setup

How do I use Postman to test the Secure Connectivity Framework?

Prerequisites

- Install Postman on your system.
- Create an application in Thredd CA.

Main Steps

Step 1: Ensure that you have the necessary Transport Certificates (CSR and Private Key) generated and signed by the Thredd Certificate Authority (CA).

Step 2: Add variables from Thredd CA to the variables section in the Postman collection. These include:

- The **Organisation ID** (UUID)
- The **Application ID** (UUID)
- The **Signing Certificate** (KID)
- The **Client ID**

Step 3: Add certificates for mTLS:

1. Open Postman and click on the Settings (the gear icon in the top-right corner).
2. Go to the **Certificates** tab and click **Add Certificate**.
3. For each required host, add the following details:
    - **Host:** Enter the host URL (for example, matls-auth.directory.sandbox.threddid.com).
    - **CRT file:** Upload the PEM file (certificate).
    - **KEY file:** Upload the private key file.
4. Repeat this process for all required hosts:
    - matls-auth.directory.sandbox.threddid.com
    - uat-thredd.mtls.eu.authz.cloudentity.io
    - api.uat.threddpay.com
    - api-uat.thredd.com
5. Save the configuration.

Step 4: Set the following variables in the **Variables** tab:

- client_id
- kid
- private_key
- ssa

Step 5: Call initial setup endpoints, including the Cloudentity Well-Known and Raidiam Connect Well-Known endpoints.

Step 6: Run DCR through the DCR endpoints. This includes getting the Radium Connect SSA and running DCR itself.

Step 7: Generate an Access Token:

1. In Postman, expand the **Authorisation** folder in the API Postman Collection.
2. Locate the **Get Access Token** endpoint.
3. The body of the request should be automatically generated.
4. Click **Send** to call the token endpoint.
5. The access token is returned in the response and stored in the variables of the Postman Collection.

Step 8: Use the Access Token:

Use the generated access token to authenticate and call other endpoints in the Cards API Postman Collection.

More Information

Where can I find more information?

For more details, refer to the following resources:

- Accessing the Cards API with mTLS (Cards API Website)
- External Interface (EHI) Guide
- Secure Connectivity Framework Product Sheet

Glossary

This page provides a list of glossary terms used in this guide.

C

Card Scheme (Network)

Card network, such as Discover, MasterCard, or Visa, responsible for managing transactions over the network and for arbitration of any disputes.

Certificate Authority

A Certificate Authority is an entity that validates the identities of entities (such as individuals, organizations, or websites) and binds them to cryptographic key pairs through the issuance of digital certificates.

Cloudentity

A service that provides identity, authorization, and open banking solutions to help organizations deliver secure digital transformation. Cloudentity is the Identity Provider (IdP) for Raidiam CA and Thredd Portal, and is also an OAuth OpenID Provider (OP) for the REST API.

Confidential Client

A client that can maintain the confidentiality of their credentials (e.g., client implemented on a secure server with restricted access to the client credentials), or capable of secure client authentication using other means. For more details, refer to RFC 6749 for the OAuth 2.0 Authorization Framework.

E

External Host Interface (EHI)

The external system to which Thredd sends real-time transaction-related data. The Program Manager uses their external host system to hold details of the balance on the cards in their programme and perform transaction-related services, such as payment authorisation, transaction matching and reconciliation.

I

Identity Provider (IDP)

An IDP is a system entity that creates, maintains, and manages identity information for principals and also provides authentication services for relying on applications within a federation or distributed network.

M

Mutual Transport Layer Security (mTLS)

mTLS is a method for mutual authentication that ensures the parties at each end of a network connection are who they claim to be by verifying that they both have the correct private key. The information within their respective TLS certificates provides additional verification.

O

OAuth OpenID Provider

An OAuth OpenID Provider (OP) is an entity that has implemented the OpenID Connect and OAuth 2.0 protocols. OPs can also be referred to by the role it plays, such as: a security token service, an identity provider (IDP), or an authorization server. In the Thredd Platform, the OP enforces access control policies in Cloudentity.

P

Program Manager

A customer who manages a card program. The program manager can create branded cards, load funds and provide other card or banking services to their end customers.

S

Scope

The permissions related to the Roles your application is registered for. Each scope relates to specific endpoiints. For example, the cards.encrypted scope gives permission to use the Get Encrypted Data endpoint.

Secure Connectivity Framework

The Secure Connectivity Framework is an umbrella set of rules and standards for identity management, verification, and assurance within a sector. The framework establishes common principles, definitions, and Open Standards for data sharing to create the foundations of a trusted data-sharing ecosystem

Smart Client

Smart Client is a user interface for programme managers to manage their account on the Thredd Platform. Smart Client is installed as a desktop application.

T

Thredd CA

Thredd CA (Certificate Authority) acts as the Certificate Authority for issuing certificates. It also includes a dashboard interface where you can create applications for your organisation, as well as assertions.

Transport Certificate

A Transport Certificate (or TLS Certificate) is a data file that contains important information for verifying a server's or device's identity, including the public key, a statement of who issued the certificate (TLS certificates are issued by a Certificate Authority), and the certificate's expiration date.

Document History

This section provides details of what has changed since the previous document release.

Version	Date	Reason	Revised by
Version 1.1	06/02/2026	Added new Scopes appendix, which describes each of the different scopes returned from the DCR Endpoint.	JB
	23/01/2026	Added section on how to connect to Thredd using AWS and VPN. See Connecting with AWS and VPN.	JB
	23/01/2026	Added a prerequisites list in the Generating and Obtaining a Software Statement Assertion (SSA) section and linked it with related information elsewhere in the guide.	ER
	14/08/2025	Added section on how to manually create users, including how to assign restricted codes and Thredd Portal roles. See Manually Create Users.	JB
	05/06/2025	New FAQ section. See FAQs.	KD
Version 1.0	25/03/2025	First version	WS

Contact Us

Please contact us if you have queries relating to this document. Our contact details are provided below.

Thredd UK Ltd.

Company registration number 09926803

Support Email: occ@thredd.com

Telephone: +44 (0) 203 740 9682

Our Head Office

Kingsbourne House

229-231 High Holborn

London

WC1V 7DA

Technical Publications

If you want to contact our technical publications team directly, for queries or feedback related to this guide, you can email us at: docs@thredd.com.