

Element On-Premise Documentation

- Introduction to Element Server Suite
- Kubernetes Installations
- Kubernetes Installations - Quick Start
- Single Node Installations
- Single Node Installs: Storage and Backup Guidelines
- Using the Installer in an Air-Gapped Environment
- Troubleshooting
- Setting up Permalinks With the Installer
- Setting Up Well Known Delegation
- Setting up Delegated Authentication With the Installer
- Integrations and Add-Ons
 - Setting Up Jitsi and TURN With the Installer
 - Setting up Group Sync with the Installer
 - Setting up GitLab, GitHub, JIRA and Webhooks Integrations With the Installer
 - Setting up Adminbot and Auditbot
 - Setting Up Hydrogen
 - Setting up On-Premise Metrics
 - Setting Up the Telegram Bridge
 - Setting Up the Teams Bridge
 - Setting Up the IRC Bridge
 - Setting Up the SIP Bridge
 - Setting Up the XMPP Bridge
 - Setting up Location Sharing
 - Removing Legacy Integrations
- Support Policies
 - On-Premise Support Scope of Coverage
 - Single Node Scope of Coverage Addendum
- Archived Documentation Repository

- Documentation Covering Installer 2023-02.02 CLI Only.
 - Documentation Covering Installers From 2022.10.01 to 2023.02.01
 - Documentation Covering Installers From 2022.07.03 to 2022.09.05
 - Documentation covering v1 and installers prior to 2022-07.03
-
- Appendices
 - Appendix A: Preparing Element Server Suite PoC
-
- Migration from self-hosted to ESS On-Premise
 - Configuring Synapse workers
 - Setting up Delegated Authentication with LDAP on Windows AD
 - Setting up Delegated Authentication with OpenID on Microsoft Azure
 - Setting up Delegated Authentication with OpenID on Microsoft AD FS

Introduction to Element Server Suite

What is Element Server Suite?

Element Server Suite provides an enterprise-grade secure communications platform that can be either self-hosted by you or run fully managed in our Element Cloud. Element Server Suite includes the Element Matrix Server, which provides a host of security and privacy features, including:

- Built on the Matrix open communications standard.
- Provides end to end encrypted messaging, voice, and video through a consumer style messenger with the power of a collaboration tool.
- Delivers data sovereignty.
- Affords a high degree of flexibility that can be tailored to many use cases.
- Allows secure federation within a single organisation or across a supply chain or ecosystem.

and combines them with the following Element Server Extensions:

- Group Sync: Synchronize group data from your identity provider and map these into Element spaces.
- Adminbot: Give your server administrator the ability to be admin in any rooms on your homeserver.
- Auditbot: Have an auditable record of conversations conducted on your homeserver.
- Security and feature updates: Updates are easy to deploy and handled by our installer.
- Bridges: Bridge to IRC, XMPP, Telegram, Microsoft Teams, or SIP. More coming soon.

Further, we also offer Enterprise Support, giving you access to the experts in federated, secure communications giving you confidence to deploy our platform for your most critical secure communications needs.

Given the flexibility afforded by this platform, there are a number of moving parts to configure.

Prefer a fully managed deployment in Element's Cloud?

Element runs cloud based infrastructure built on Amazon Web Services for the purpose of hosting Element Server Suite for our customers. If you go with this option, we will manage setting up, configuring, and maintaining your Element Server Suite instance.

For more information, please see: [How to Get an EMS Server](#).

Deploying to Kubernetes or to a standalone server?

Element Enterprise On-Premise can be deployed both to Kubernetes (a lightweight container orchestration platform) or a standalone server. One key benefit of going with Kubernetes is that you can add more resources and nodes to a cluster as you need them where you are capped at one node with our standalone server. In the case of our standalone server installation, we deploy microk8s (a smaller lightweight distribution of Kubernetes), which we then use for deploying our application.

Hardware Requirements

In general, regardless of if you pick the standalone server or Kubernetes deployment, you will need a base level of hardware to support the application.

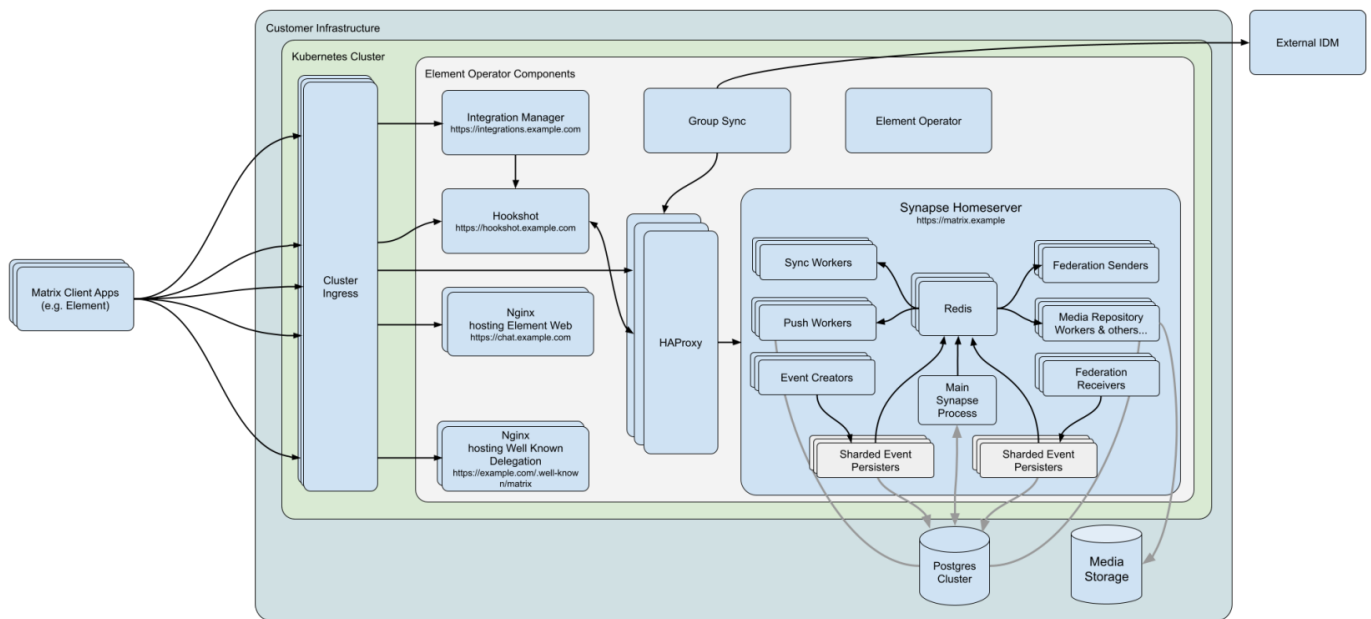
For scenarios that utilise open federation, Element recommends a **minimum** of 6 vCPUs/CPU's and 16GB ram for the host(s) running synapse pods.

For scenarios that utilise closed federation, Element recommends a **minimum** of 4 vCPUs/CPU's and 8GB ram for the host(s) running synapse pods.

On kubernetes installations, we default to 2 replicas of every service that you install. This can quickly increase the required number of CPUs, so be very mindful that right out of the box, you could need up to 16 vCPUs and adding more integrations and add-ons could increase this number even further.

Architecture

This document gives an overview of our secure communications platform architecture:



(Please click on the image to view it at 100%.)

Comprising our secure communications platform are the following components:

- synapse : The homeserver itself.
- element-web : The Element Web client.
- integrator: Our integration manager.
- synapse admin ui : Our Element Enterprise Administrator Dashboard.
- postgresql (Optional) : Our database. Only optional if you already have a separate PostgreSQL database, which is required for a multiple node setup.
- groupsync (Optional) : Our group sync software

- adminbot (Optional) : Our bot for admin tasks.
- auditbot (Optional) : Our bot that provides auditability.
- hookshot (Optional) : Our integrations with gitlab, github, jira, and custom webhooks.
- hydrogen (Optional) : A light weight alternative chat client.
- jitsi (Optional) : Our VoIP platform for group conferencing.
- coturn (Optional) : TURN server. Required if deploying VoIP.
- prometheus (Optional) : Provides metrics about the application and platform.
- grafana (Optional) : Graphs metrics to make them consumable.
- telegram bridge (Optional) : Bridge to connect Element to Telegram.
- teams bridge (Optional) : Bridge to connect Element to MS Teams.
- xmpp bridge (Optional) : Bridge to connect Element to XMPP.
- irc bridge (Optional) : Bridge to connect Element to IRC.
- sip bridge (Optional) : Bridge to connect Element to SIP.

For each of the components in this list (excluding postgresql, groupsync, adminbot, auditbot, and prometheus), you must provide a hostname on your network that meets this criteria:

- Fully resolvable to an IP address that is accessible from your clients.
- Signed PEM encoded certificates for the hostname in a crt/key pair. Certificates should be signed by an internet recognised authority, an internal to your company authority, or LetsEncrypt.

It is possible to deploy Element Enterprise On-Premise with self-signed certificates and without proper DNS in place, but this is not ideal as the mobile clients and federation do not work with self-signed certificates.

Information on how to use self-signed certificates and hostname mappings instead of DNS can be found in [How to Setup Local Host Resolution Without DNS](#)

In addition to hostnames for the above, you will also need a hostname and PEM encoded certificate key/cert pair for your base domain. If we were deploying a domain called example.com and wanted to deploy all of the software, we would have the following hostnames in our environment that needed to meet the above criteria:

- example.com (base domain)
- synapse.example.com (homeserver)
- element.example.com (element web)
- integrator.example.com (integration manager)
- admin.example.com (admin dashboard)
- hookshot.example.com (Our integrations)
- hydrogen.example.com (Our light weight chat client)
- jitsi.example.com (Our VoIP platform)
- coturn.example.com (Our TURN server)
- grafana.example.com (Our Grafana server)
- telegrambridge.example.com (Our Telegram Bridge)
- teamsbridge.example.com (Our Teams Bridge)

As mentioned above, this list excludes postgresql, groupsync, adminbot, auditbot, and prometheus.

Wildcard certificates do work with our application and it would be possible to have a certificate that validated *.example.com and example.com for the above scenario. It is key to do both the base domain and the wildcard in the same certificate in order for this to work.

Further, if you want to do voice or video across network boundaries (ie: between people not on the same local network), you will need a TURN server. If you already have one, you do not have to set up coturn. If you do not already have a TURN server, you will want to set up coturn (our installer can do this for you) and if your server is behind NAT, you will need to have an external IP in order for coturn to work.

Installation

Airgapped Environments

If you are going to be installing into an airgapped environment (one without internet connectivity), you will need to also download the airgapped installer, which is ~4GB of data that will need to be transferred to your airgapped environment. More information on this can be found in our airgapped installation documentation here:

<https://ems-docs.element.io/books/element-on-premise-documentation/page/using-the-installer-in-an-air-gapped-environment>

Software

To obtain our software, please visit our downloads page at: <https://ems.element.io/on-premise/download>

Kubernetes Application (Multiple Nodes)

For an installation into a kubernetes environment, make sure you have a Kubernetes platform deployed that you have access to and head over to [Kubernetes Installations](#)

Standalone (Single Node)

For a standalone installation, please note that we support these on the following platforms:

- Ubuntu Server 20.04
- Enterprise Linux 8 (RHEL, CentOS Stream, etc.)

Once you have a server with one of these installed, please head over to [Single Node Installations](#)

Kubernetes Installations

Overview

Our Installer can handle the installation of Element Enterprise into your existing production kubernetes (k8s) environment.

Prerequisites

Before beginning the installation, there are a few things that must be prepared to ensure a successful deployment and functioning installation.

Python environment

The installer needs python3, pip3 and python3-venv installed to run.

Kubectl environment

The installer uses your currently active `kubectl` context which can be determined with `kubectl config current-context` - make sure this is the correct context as all subsequent operations will be performed under this.

More information on configuring this can be found in the upstream kubectl docs

Be sure to `export K8S_AUTH_CONTEXT=<kube context name>` for the Installer if you need to use a context aside from your currently active one.

PostgreSQL

Before you can begin with the installation you must have a PostgreSQL database instance available. The installer does not manage databases itself.

The database you use **must** be set to a locale of `C` and use `UTF8` encoding - see <https://matrix-org.github.io/synapse/latest/postgres.html#set-up-database> for further details as they relate to Synapse. If the locale / encoding are incorrect, Synapse will fail to initialize the database and get stuck in a `CrashLoopBackoff` cycle.

Please make note of the database hostname, database name, user, and password as you will need these to begin the installation.

For testing and evaluation purposes, you can deploy PostgreSQL to k8s before you begin the installation process - see [Kubernetes Installations - Quick Start - Deploying PostgreSQL to Kubernetes](#) for more information.

Kubernetes Ingress Controller

The installer does not manage cluster Ingress capabilities since this is typically a cluster-wide concern - You must have this available prior to installation. Without a working Ingress Controller you will be unable to route traffic to your services without manual configuration.

If you do not have an Ingress Controller deployed please see [Kubernetes Installations - Quick Start - Deploying ingress-nginx to Kubernetes](#) for information on how to set up a bare-bones `ingress-nginx` installation to your cluster.

Use an existing Ingress Controller

If you have an Ingress Controller deployed already and it is set to the default class for the cluster, you shouldn't have to do anything else.

If you're unsure you can see which providers are available in your cluster with the following command:

```
$ kubectl get IngressClass
```

NAME	CONTROLLER	PARAMETERS	AGE
nginx	k8s.io/ingress-nginx	<none>	40d

And you can check to see whether an IngressClass is set to default using kubectl, for example:

```
$ kubectl describe IngressClass nginx
```

Name: nginx

Labels: app.kubernetes.io/component=controller
app.kubernetes.io/instance=ingress-nginx
app.kubernetes.io/managed-by=Helm
app.kubernetes.io/name=ingress-nginx
app.kubernetes.io/part-of=ingress-nginx
app.kubernetes.io/version=1.1.1
argocd.argoproj.io/instance=ingress-nginx
helm.sh/chart=ingress-nginx-4.0.17

Annotations: ingressclass.kubernetes.io/is-default-class: true

Controller: k8s.io/ingress-nginx

Events: <none>

In this example cluster there is only an `nginx` IngressClass and it is already default, but depending on the cluster you are deploying to this may be something you must manually set.

Migrating from our older installer

If you have previously used installer versions 2023-03.01 and earlier, you will need to run our migration script to convert your previous configuration to the new format that is used with our UI based installer. This script became available in 2023-03.02, so you must have at least that version or higher of the graphical installer for this to work.

NOTE: Before running the migration script, we highly recommend that you take a backup or snapshot of your working environment. While we have tested the migration script against several configurations at this point, we have not tested for all of the combinations of configuration that the previous installer allowed. We expect that migration will be a quick process for most customers, but in the event that something goes wrong, you'll want to be able to get back to a known good state through a backup or snapshot.

NB: If you are using group sync, you cannot presently migrate to the graphical installer. We are working to address the issues with migrating group sync and will remove this note once we have those addressed.

If you have not used our installer before, you may safely ignore this section.

To run the migration script, please do the following:

```
chmod +x ./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin
./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin --import ~/.element-
onpremise-config
```

Make sure to replace `~/.element-onpremise-config` with the path that your actual configuration exists in. Further, replace `YYYY-MM.VERSION` with the appropriate tag for the installer you downloaded.

Once the import has finished, the GUI will start and you will be able to browse to the installer at one of the provided URLs, much as if you had started the installer without doing a migration as detailed in the following section.

Beginning the Installation

Head to <https://ems.element.io/on-premise/download> and download the latest installer. The installer will be called `element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin`. You will take this file and copy it to the machine where you will be installing the Element Server Suite. Once you have this file on the machine in a directory accessible to your sudo-enabled user, you will run:

```
chmod +x ./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin
```

replacing the `YYYY-MM.VERSION` with the appropriate tag for the installer you downloaded.

If you have multiple kubernetes clusters configured in your kubeconfig, you will have to export the `K8S_AUTH_CONTEXT` variable before running the installer :

```
export K8S_AUTH_CONTEXT=<kube context name>
```

Once you have done this, you will run:

```
./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin
```

replacing the `YYYY-MM.VERSION` with the appropriate tag for the installer you downloaded, and this will start a web server with the installer loaded.

You will see a message similar to:

```
[user@element-demo ~]$ ./element-enterprise-graphical-installer-2023-02.02-gui.bin
```

```
Testing network...
```

```
Using self-signed certificate with SHA-256 fingerprint:
```

```
F3: 76: B3: 2E: 1B: B3: D2: 20: 3C: CD: D0: 72: A3: 5E: EC: 4F: BC: 3E: F5: 71: 37: 0B: D7: 68: 36: 2E: 2C: AA: 7A: F2: 83: 94
```

```
To start configuration open:
```

```
https://192.168.122.47:8443 or https://10.1.185.64:8443 or https://127.0.0.1:8443
```

At this point, you will need to open a web browser and browse to one of these IPs. You may need to open port 8443 in your firewall to be able to access this address from a different machine.

If you are unable to open port 8443 or you are having difficulty connecting from a different machine, you may want to try ssh port forwarding in which you would run:

```
ssh <host> -L 8443:127.0.0.1:8443
```

replacing host with the IP address or hostname of the machine that is running the installer. At this point, with ssh connected in this manner, you should be able to use the <https://127.0.0.1:8443> link as this will then forward that request to the installer box via ssh.

Upon loading this address for the first time, you may be greeted with a message informing you that your connection isn't private such as this:



Your connection isn't private

Attackers might be trying to steal your information from **192.168.122.47** (for example, passwords, messages, or credit cards).

NET::ERR_CERT_AUTHORITY_INVALID

Advanced

Go back

In this case, you'll need to click "Advanced" and then "Continue to (unsafe)" in order to view the installer. As the exact button names and links can vary between browsers, it would be hard for us to document them all, so you may have slightly different wording depending on your browser.

The Hosts Screen

The very first page that you come to is the host screen.



Host.

Install

Standalone Kubernetes Application

Skip Operator Setup

Default

Skip Updater Setup

Default

EMS Image Store

Your ems image store username / password

Username

382255a2-8d20-4a4a-aad7-2d3db34b7506

Token

.....



You will want to make sure that "Kubernetes Application" is selected. You can opt to skip the update setup or the operator setup, but unless you know why you are doing that, you should leave them alone.

The very next prompt that you come to is for an EMS Image Store Username and Token. These are provided to you by element as access tokens for our enterprise container registries. If you have lost your token, you can always generate a new token at <https://ems.element.io/on-premise/subscriptions>.

Namespaces

The namespace where to deploy the element applications

The namespace where to deploy the updater controller manager

The namespace where to deploy the updater controller manager

Connectivity

Connected Airgapped

Dockerhub

Optionally reduce rate limiting by providing your dockerhub credentials



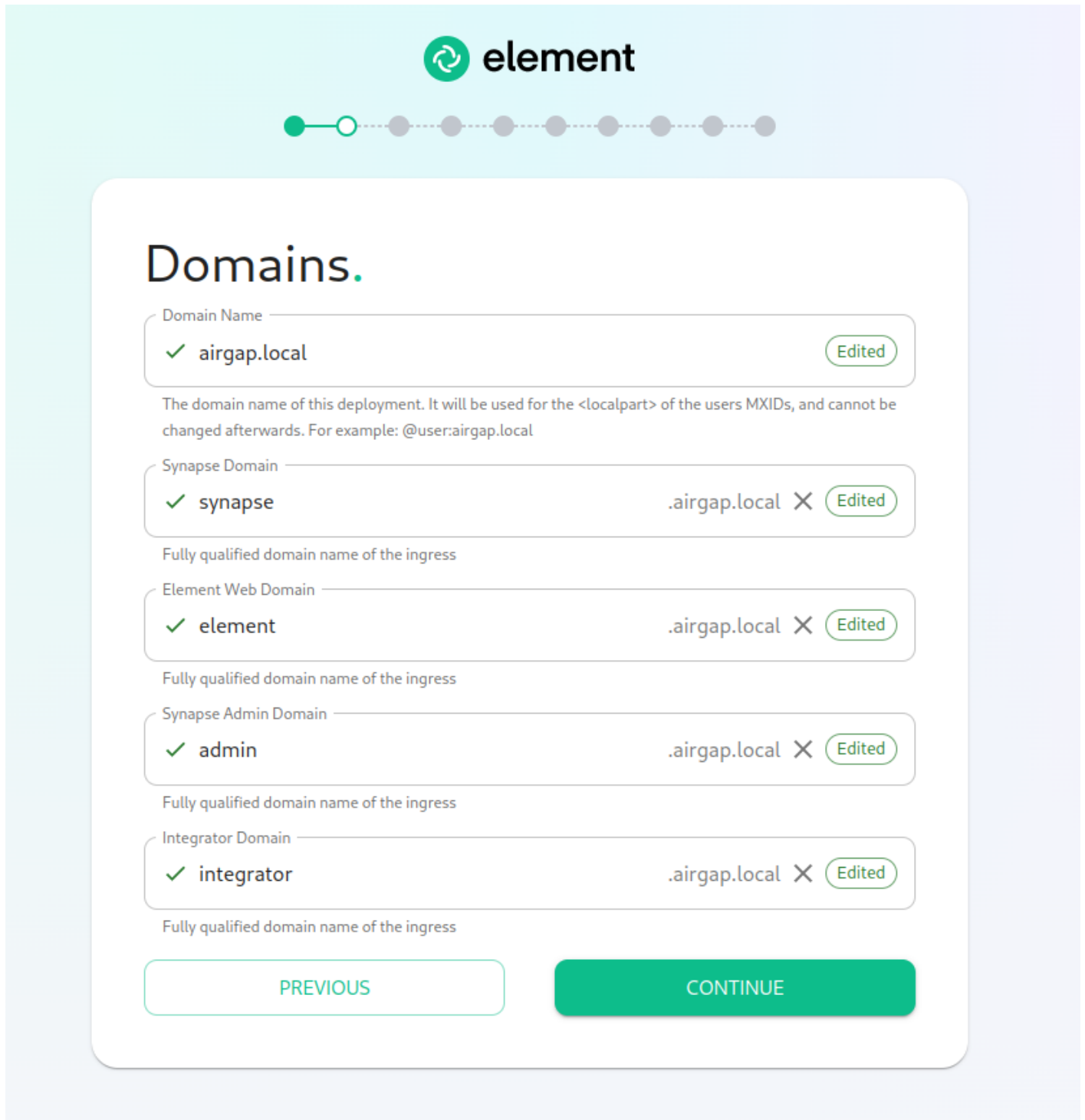
CONTINUE

Here, we find the ability to set the namespaces that the application will be deployed into.

The final options on the hostpage are related to connectivity. For this guide, we are assuming "Connected" and you can leave that be. If you are doing "Airgapped", you would pick airgapped at this point and then please see the section on airgapped installations.

You are presented with the option to provide docker hub credentials. These are optional, but if you do not provide them, you may be rate limited by Docker and this could cause issues pulling container images.

The Domains Screen



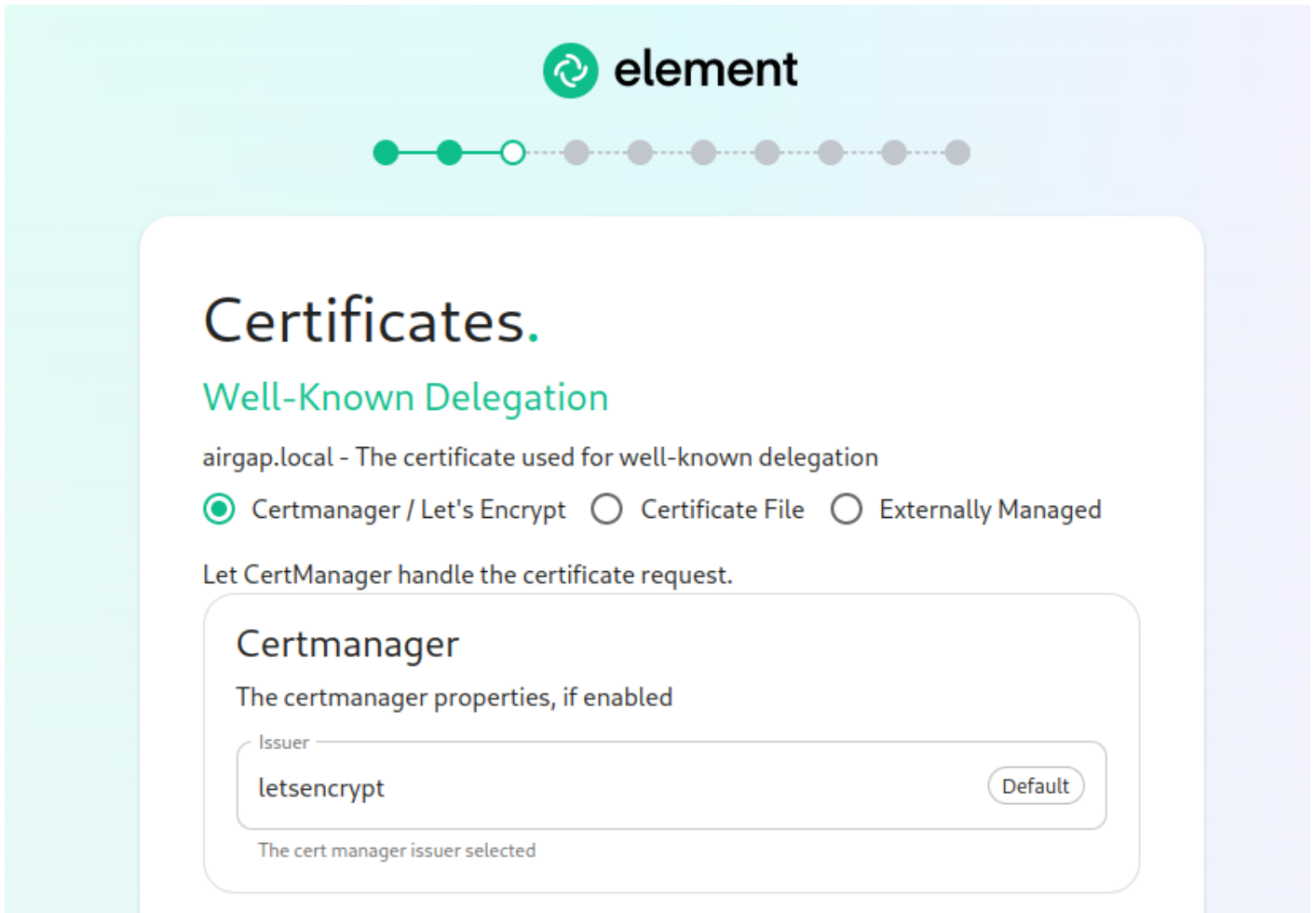
On this page, we get to specify the domains for our installation. In this example, we have a domain name of airgap.local and this would mean our MXIDs would look like @kabbott:airgap.local.

Our domain page has checking to ensure that the host names resolve. Once you get green checks across the board, you can click continue.

The Certificates Screen

On the Certificates screen, you will provide SSL certificate information for well-known delegation, Synapse, Element Web, Synapse Admin, and Integrator.

If you are using Let's Encrypt, then each of the sections should look like:



The screenshot shows the 'Certificates' configuration page in the Element interface. At the top, the 'element' logo is displayed with a progress indicator consisting of nine dots, where the first three are filled and the rest are empty. The main heading is 'Certificates.' followed by the sub-heading 'Well-Known Delegation'. Below this, the domain 'airgap.local' is listed with the description 'The certificate used for well-known delegation'. There are three radio button options: 'Certmanager / Let's Encrypt' (which is selected), 'Certificate File', and 'Externally Managed'. A note states 'Let CertManager handle the certificate request.' Below this is a 'Certmanager' section with the text 'The certmanager properties, if enabled'. It contains a text input field for 'Issuer' with the value 'letsencrypt' and a 'Default' button. A small note below the input field reads 'The cert manager issuer selected'.

If you are using certificate files, then you will see a screen like:



Certificates.

Well-Known Delegation

airgap.local - The certificate used for well-known delegation

Certmanager / Let's Encrypt Certificate File Externally Managed

Upload a certificate and its private key.

Certificate

Certificate file

Secrets / Well Known Delegation /

Well Known Delegation Certificate ▾

Certificate No file chosen
WellKnownDelegation Certificate

Secrets / Well Known Delegation /

Well Known Delegation Private Key ▾

Private key No file chosen
WellKnownDelegation Private Key

which allows you to upload a .crt and .key file for each host. These files must be in PEM encoding. Our installer does accept wildcard certificates.

Once you have completed the certificate section for each host on the page, you may click continue.

The Database Screen



Database.

PostgreSQL database name

PostgreSQL database host

PostgreSQL port



TLS settings to use for the Postgres connection

PostgreSQL username

/

/



The postgres password

As you must use an external PostgreSQL database with our kubernetes install, on this page, we provide the option to specify the database name, the database host name, the port to connect to, the SSL mode to use, and finally, the username and password to connect with. Once you have completed this section, you may click continue.

The Media Screen

On this page, you can specify the size of your synapse media volume. Please leave "Create New Volume" checked and specify the size of the volume that you wish to allocate. You must have this space available in `/data/element-deployment` or whatever you specified back on the hosts screen. If you wish to create a 50G volume, you would need to specify `50Gi` for the Volume size.

The Cluster Screen

Most deployments can ignore this, however, if you want to change any kubernetes cluster parameters, this is where to do it.

If you are in an environment where you have self-signed certificates, you will want to disable TLS verification, by clicking "Advanced" and then scrolling down and unchecking `Verify TLS`:

The screenshot shows the 'Verify TLS' configuration section. At the top, there is a checkbox labeled 'Verify TLS' which is currently unchecked. Below this, the section is titled 'TLS verification'. It contains two main panels. The first panel has a breadcrumb trail: 'Secrets / Global / CA.pem'. Below the breadcrumb, there is a 'Certificate authority' label, a 'Choose File' button, and the text 'No file chosen'. A green 'Edited' badge is present next to the text 'The CA to inject into the deployment.' The second panel also has a breadcrumb trail: 'Secrets / Global / Generic Shared Secret'. Below this, there is a text input field labeled 'Generic Shared Secret' containing a series of dots, and a toggle icon (an eye) to its right. Below the input field, there is a descriptive text: 'The generic shared secret to use as a seed for all internally-generated secrets'.

Please bear in mind that disabling TLS verification and using self-signed certificates is not recommended for production deployments.

If your host names are not DNS resolvable, you need to use host aliases and this can be set up here. You will also click "Advanced" and scroll down to the "Host Aliases" section in "k8s". In here, you will click "Add Host Aliases" and then you will specify an IP and host names that resolve to that IP as such:

Host Aliases

The list of hosts aliases to configure on the pod spec. It should be avoid as much as possible to use this feature. Please prefer using a DNS entry to resolve your hostnames. This can be used as a workaround when entries cannot be resolved using DNS, for example for our automated testings.

IP *

192.168.122.47 Edited

An IP resolution to add to /etc/hosts

Hostnames

An hostname of the associated ip to add to /etc/hosts

airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

element.airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

synapse.airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

admin.airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

integrator.airgap.local Edited

[ADD MORE HOSTNAMES](#)

[ADD MORE HOST ALIASES](#)

Important: If you are not using OpenShift, you will need to set "Force UID GID" and "Set Sec Comp" to "Enable" under the section "Security Context" so that it looks like:

Security Context

Force UID GID

Enable

Edited ▼

Enable pod runAsUser and fsGroup in security context. Disable if it should not be used, in the case of openshift for example. Auto attempts to detect openshift automatically.

Set Sec Comp

Enable

Edited ▼

Enable RuntimeDefault pod seccomp. disable if it should not be used, in the case of openshift for example. Auto attempts to detect openshift automatically.

If you are using OpenShift, you should leave the values of "Force UID GID" and "Set Sec Comp" set to "Auto".

When you are finished with this page, you can click continue.

The Synapse Screen



Synapse.

This is a matrix homeserver.

Config

Accept Invites

Manual

Default

Whether to enable auto accept invites. Defaults to manual if not set

Max MAU Users

250

Default

Maximum number of Matrix Active Users

Registration

Closed

Default

Synapse registration

Secrets

/ Synapse

/ Admin Password

Admin Password

.....



Synapse admin user password

The first setting that you will see is whether you want to auto accept invites. The default of "Manual" will fit most use cases, but you are welcome to change this value.

The next setting is the maximum number of monthly active users (MAU) that you have purchased for your server. Your server will not allow you to go past this value. If you set this higher than your purchased MAU and you go over your purchased MAU, you will need to true up with Element to cover the cost of the unpaid users.

The next setting concerns registration. A server with open registration on the open internet can become a target, so we default to closed registration. You will notice that there is a setting called "Custom" and this requires explicit custom settings in the additional configuration section. Unless instructed by Element, you will not need the "Custom" option and should instead pick "Closed" or "Open" depending on your needs.

After this, you will see that the installer has picked an admin password for you. You will want to use the eye icon to view the password and copy this down as you will use this with the user `onprem-admin-donotdelete` to log into the admin panel after installation.

Telemetry

Telemetry properties

Enabled

True to enable telemetry

Default

Room

#element-telemetry Default

The telemetry room where to send telemetry

Advanced

Advanced

PREVIOUS CONTINUE

Continuing, we see telemetry. You should leave this enabled as you are required to report MAU to Element. In the event that you are installing into an environment without internet access, you may disable this so that it does not continue to try talking to Element. That said, you are still required to generate an MAU report at regular intervals and share that with Element.

For more information on the data that Element collects, please see: [What Telemetry Data is Collected by Element?](#)

Next, we have two advanced buttons. The top one is for synapse and gives you a text box to inject additional synapse configs (homeserver.yaml), access to the macaroon, registration shared secret, and signing key. Further, you can add and configure multiple synapse workers, external appservices, and federation. These topics are presently beyond the scope of this install guide, but we will write them up in due time. Further, you also have the ability to set a STUN shared secret and set TURN URIs for any existing TURN servers that you may have in your environment.

Additional

Additional config to inject

1

Secrets

/

Synapse

/

Macaroon

▼

Macaroon

.....



It should be generated randomly. It should never change.

Stun

Coturn configuration

HIDE

Secrets

/ Synapse

/ Stun Shared Secret

STUN Shared Secret

.....



It should be generated randomly and shared with the stun server.

Turn Uris

The stun servers

=



A stun server uri

ADD MORE TURN URIS

The second advanced button allows you to explicitly set any kubernetes cluster settings that you would like just for the synapse pods. Most users will be able to ignore this.

You can hit continue to go to the next screen.

The Element Web Screen

Most users will be able to simply click "Continue" here.

The Advanced section allows you to set any custom element web configurations you would like (config.json).

Config

Additional configuration

Element web additional configuration.

1

K8s

Force values for components of Element Web

SHOW

PREVIOUS

CONTINUE

A common custom configuration would be configuring permalinks for Element, which we have documented here: [Setting up Permalinks With the Installer](#)

Further, it provides access to the k8s section, allowing you to explicitly set any kubernetes cluster settings that you would like just for the element-web pod.

The Enterprise Admin Dashboard

Most users will be able to simply click "Continue" here. The Advanced section allows you to explicitly set any kubernetes cluster settings that you would like just for the synapse-admin-ui pod.

One word to note here is that if you are not using delegated authentication, then the initial username that an administrator will use to log into this dashboard post-installation is `onpremise-admin-donotdelete`. You can find the password for this user on the Synapse page in the "Admin Password" field.

If you are using delegated authentication, you will need to assign a user admin rights as detailed in this article: [How do I give a user admin rights when I am using delegated authentication and cannot log into the admin console?](#)

The Integrator Screen

The screenshot displays the 'Integrator' configuration page. At the top, it says 'Integrator.' followed by the subtitle 'Send messages to external services'. Below this is a 'Config' section with a checkbox for 'Enable Custom Widgets' (unchecked) and a sub-label 'Enable custom widgets in scalar-web'. A 'Default' button is present. The 'Jitsi Domain' field contains 'meet.element.io' with a 'Default' button. Below it, the 'Verify TLS' section has a dropdown menu set to 'Use Global Setting' with a 'Default' button and a downward arrow. The 'Log' section is titled 'Logging settings' and has a 'Level' dropdown set to 'Info' with a 'Default' button and a downward arrow. A sub-label reads 'The maximum level of log output'. At the bottom of the log section, there is a checkbox for 'Structured' (unchecked) and a sub-label 'Output logs in logstash format. Otherwise, logs are output in a console friendly format.' with a 'Default' button.

On this page, you can set up Integrator, the integrations manager.

The first option allows you to choose whether users can add custom widgets to their rooms with the integrator or not.

The next option allows you to specify which Jitsi instance the Jitsi widget will create conferences on.

The verify TLS option allows you to set this specifically for Integrator, regardless of what you set on the cluster screen.

The logging section allows you to set the log level and whether the output should be structured or not.

The Advanced section allows you to explicitly set any kubernetes cluster settings that you would like just for the integrator pods.

Click "Continue to go to the next screen".

The Integrations Screen

This screen is where you can install any available integrations.

Some of these integrations will have "YAML" next to them. When you see this designation, this integration requires making settings in YAML, much like the old installer. However, with this installer, these YAML files are pre-populated and often only involve a few changes.

If you do not see a "YAML" designation next to the integration then this means that will use regular GUI elements to configure this integration.

Over time, we will do the work required to move the integrations with "YAML" next to them to the new GUI format.

For specifics on configuring well known delegation, please see [Setting Up Well Known Delegation](#)

For specifics on setting up Delegated Authentication, please see [Setting up Delegated Authentication With the Installer](#)

For specifics on setting up Group Sync, please see [Setting up Group Sync with the Installer](#)

For specifics on setting up GitLab, GitHub, and JIRA integrations, please see [Setting up GitLab, GitHub, and JIRA Integrations With the Installer](#)

For specifics on setting up Adminbot and Auditbot, please see: [Setting up Adminbot and Auditbot](#)

For specifics on setting up Hydrogen, please see: [Setting Up Hydrogen](#)

For specifics on pointing your installation at an existing Jitsi instance, please see [Setting Up Jitsi and TURN With the Installer](#)

If you do not have an existing TURN server or Jitsi server, our installer can configure these for you by following the extra steps in [Setting Up Jitsi and TURN With the Installer](#)

For specifics on configuring the Teams Bridge, please see [Setting Up the Teams Bridge](#)

For specifics on configuring the Telegram Bridge, please see [Setting Up the Telegram Bridge](#)

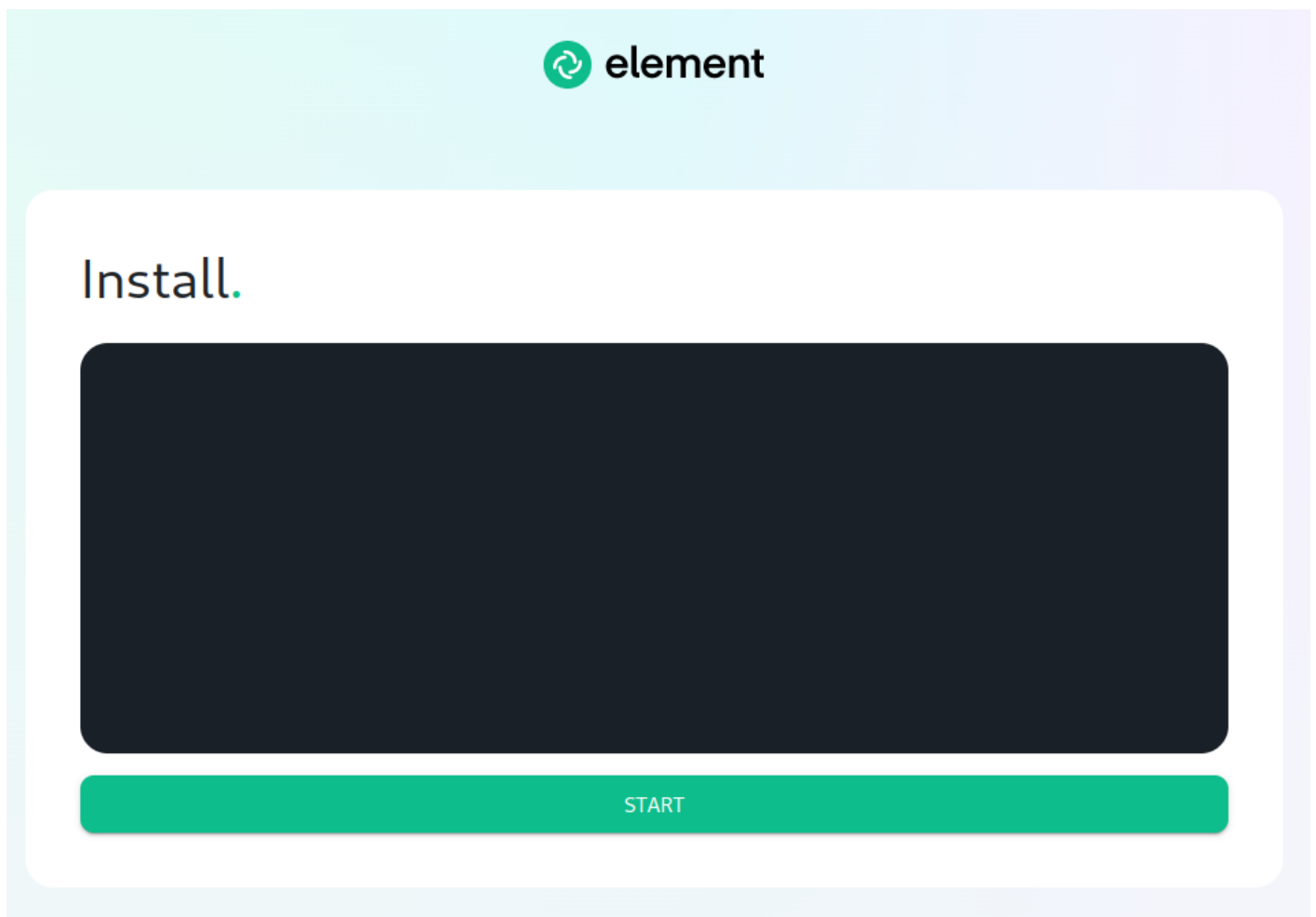
For specifics on configuring the IRC Bridge, please see [Setting Up the IRC Bridge](#)

For specifics on configuring the XMPP Bridge, please see [Setting Up the XMPP Bridge](#)

Once you have configured all of the integrations that you would like to configure, you can click "Continue" to head to the installation screen.

The Installation Screen

On the installation screen, you should see a blank console and a start button:



Click Start.

After a moment, you will notice the installer hang. If you go back to the prompt where you are running the installer, you will see that you are being asked for the sudo password:

Install.



```
[karll@airgap ~]$ ./element-enterprise-graphical-installer-2023-02.02-gui-rc1.bin
Testing network...

Using self-signed certificate with SHA-256 fingerprint:
    CB:8E:4A:75:80:32:D5:E0:A0:2C:90:A7:DF:F9:2F:9F:6D:14:F7:18:53:D0:C5:6C:20:D5:95:A8:1A:57:67:21

To start configuration open:
    https://192.168.122.47:8443 or https://10.1.185.64:8443 or https://127.0.0.1:8443
API resolved without sending a response for /api/logs, this may result in stalled requests.
[sudo] password for karll: █
```

Go ahead and enter the sudo password and the installation will continue.

On the very first time that you run the installer, you will be prompted to log out and back in again to allow Linux group membership changes to be refreshed. This means that you will need to issue a ctrl-C in the terminal running your installer and actually log all the way out of your Linux session, log back in, restart the installer, navigate back to the installer screen, click start again, and then re-enter your sudo password. You will only have

to perform this step once per server.

Verifying Your Installation

Once the installation has finished, it can take as much as 15 minutes on a first run for everything to be configured and set up. If you use:

```
kubectl get pods -n element-onprem
```

You should see similar output to:

NAME	READY	STATUS	RESTARTS	AGE
app-element-web-c5bd87777-rqr6s	1/1	Running	1	29m
server-well-known-8c6bd8447-wddtm	1/1	Running	1	29m
postgres-0	1/1	Running	1	40m
instance-synapse-main-0	1/1	Running	2	29m
instance-synapse-haproxy-5b4b55fc9c-hnlmp	1/1	Running	0	20m

Once the admin console is up and running:

```
first-element-deployment-synapse-admin-ui-564cbf5665-dn8nv 1/1 Running
1 (4h4m ago) 3d1h
```

and synapse:

```
first-element-deployment-synapse-redis-59548698df-gqkcq 1/1 Running
1 (4h4m ago) 3d2h
first-element-deployment-synapse-haproxy-7587dfd6f7-gp6wh 1/1 Running
2 (4h3m ago) 2d23h
first-element-deployment-synapse-appservice-0 1/1 Running
3 (4h3m ago) 3d
first-element-deployment-synapse-main-0 1/1 Running
0 3h19m
```

then you should be able to log in at your admin panel (in our case <https://admin.airgap.local/>) with the `onprem-admin-donotdelete` user and the password that was specified on the "Synapse" screen.

A word about Configuration Files

In the new installer, all configuration files are placed in the directory `.element-enterprise-server`. This can be found in your user's home directory. In this directory, you will find a subdirectory called `config` that contains the

actual configurations.

Running the Installer without the GUI

It is possible to run the installer without using the GUI provided that you have a valid set of configuration files in the `.element-enterprise-server/config` directory. Directions on how to do this are available at: <https://ems-docs.element.io/books/ems-knowledge-base/page/how-do-i-run-the-installer-without-using-the-gui>. Using this method, you could use the GUI as a configuration editor and then take the resulting configuration and modify it as needed for further installations.

This method also makes it possible to set things up once and then run future updates without having to use the GUI.

End-User Documentation

After completing the installation you can share our User Guide to help orient and onboard your users to Element!

Kubernetes Installations - Quick Start

For testing and evaluation purposes - Element cannot guarantee production readiness with these sample configurations.

Requires Helm installed locally

Deploying PostgreSQL to Kubernetes

If you do not have a database present, it is possible to deploy PostgreSQL to your Kubernetes cluster.

This is great for testing and *can* work great in a production environment, but only for those with a high degree of comfort with PostgreSQL as well as the tradeoffs involved with k8s-managed databases.

There are many different ways to do this depending on your organization's preferences - as long as it can create an instance / database with the required locale and encoding it will work just fine. For a simple non-production deployment, we will demonstrate deployment of the bitnami/postgresql into your cluster using Helm.

You can add the `bitnami` repo with a few commands:

```
helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update
helm search repo
bitnami/postgresql
```

~/Desktop

NAME	CHART VERSION	APP VERSION	
DESCRIPTION			
bitnami/postgresql	12.5.7	15.3.0	PostgreSQL (Postgres) is an open source object-...
bitnami/postgresql-ha	11.7.5	15.3.0	This PostgreSQL cluster solution includes the P...

Next, you'll need to create a `values.yaml` file to configure your PostgreSQL instance. This example is enough to get started, but please consult the chart's README and values.yaml for a list of full parameters and options.

```
auth:
  # This is the necessary configuration you will need for the Installer, minus the hostname
  database: "synapse"

  username: "synapse"
  password: "PleaseChangeMe! "
```



```

primary:
  initdb:
    # This ensures that the initial database will be created with the proper collation
    settings
    args: "--lc-collate=C --lc-ctype=C"

persistence:
  enabled: true
  # Set this value if you need to use a non-default StorageClass for your database's PVC
  # storageClass: ""
  size: 20Gi

# Optional - resource requests / requirements
# These are sufficient for a 10 - 20 user server
resources:
  requests:
    cpu: 500m
    memory: 512Mi
  limits:
    memory: 2Gi

```

This example `values.yaml` file is enough to get you started for testing purposes, but things such as TLS configuration, backups, HA and maintenance tasks are outside of the scope of the installer and this document.

Next, pick a namespace to deploy it to - this can be the same as the Installer's target namespace if you desire. For this example we'll use the `postgresql` namespace.

Then it's just a single Helm command to install:

```

# format:
# helm install --create-namespace -n <namespace> <helm-release-name> <repo/chart> -f <values
file> (-f <additional values file>)

helm install --create-namespace -n postgresql postgresql bitnami/postgresql -f values.yaml

```

Which should output something like this when it is successful:

```
-- snip --
```

PostgreSQL can be accessed via port 5432 on the following DNS names from within your cluster:

```
postgresql.postgresql.svc.cluster.local - Read/Write connection
-- snip --
```

This is telling us that `postgresql.postgresql.svc.cluster.local` will be our hostname for PostgreSQL connections, which is the remaining bit of configuration required for the Installer in addition to the database/username/password set in `values.yaml`. This will differ depending on what namespace you deploy to, so be sure to check everything over.

If needed, this output can be re-displayed with `helm get notes -n <namespace> <release name>`, which for this example would be `helm get notes -n postgresql postgresql`

Deploying ingress-nginx controller

Similar to the PostgreSQL quick start example, this requires Helm

The kubernetes/ingress-nginx chart is an easy way to get a cluster outfitted with Ingress capabilities.

In an environment where LoadBalancer services are handled transparently, such as in a simple test k3s environment with `svcLb` enabled there's a minimal amount of configuration.

This example `values.yaml` file will create an IngressClass named `nginx` that will be used by default for any `Ingress` objects in the cluster.

```
controller:
  ingressClassResource:
    name: nginx
    default: true
    enabled: true
```

However, depending on your cloud provider / vendor (i.e. AWS ALB, Google Cloud Load Balancing etc) the configuration for this can vary widely. There are several example configurations for many cloud providers in the chart's README

You can see what your resulting HTTP / HTTPS IP address for this ingress controller by examining the service it creates - for example, in my test environment I have an installed release of the `ingress-nginx` chart called `k3s` under the `ingress-nginx` namespace, so I can run the following:

```
# format:
# kubectl get service -n <namespace> <release-name>-ingress-nginx-controller
$ kubectl get service -n ingress-nginx k3s-ingress-nginx-controller
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	AGE
k3s-ingress-nginx-controller	LoadBalancer	10.43.254.210		

The value of `EXTERNAL-IP` will be the address that you'll need your DNS to point to (either locally via `/etc/hosts` or LAN / WAN DNS configuration) to access your installer-provisioned services.

Single Node Installations

Installing a Standalone Server

Overview

Our installer can handle the installation of environments in which only one server is available. This environment consists of a single server with a microk8s deployment in which we deploy our Element Server Suite to, resulting in a fully functioning version of our platform.

To get started with a standalone installation, there are several things that need to be considered and this guide will work through them:

- Operating System
- Postgresql Database
- TURN Server
- SSL Certificates
- Extra configuration items

Once these areas have been covered, you'll be ready to install your standalone server!

Operating System

To get started, we have tested on Ubuntu 20.04 and Red Hat Enterprise Linux 8.7 and suggest that you start there as well. For x86_64, you can grab an Ubuntu iso here:

<https://releases.ubuntu.com/20.04.3/ubuntu-20.04.3-live-server-amd64.iso>

or you can get Red Hat Enterprise Linux 8 with a Developer Subscription at:

https://access.redhat.com/downloads/content/479/ver=/rhel---8/8.7/x86_64/product-software

Note that future references in this document to `EL` reference Enterprise Linux.

Ubuntu Specific Directions

Make sure to select docker as a package option. Do set up ssh.

Once you log in, please run:

```
sudo apt-get update
sudo apt-get upgrade
```

The installer requires that you run it as a non-root user who has sudo permissions. Please make sure that you have a user who can use `sudo`. If you wanted to make a user called `element-demo` that can use `sudo`, the following commands (run as root) would achieve that:

```
useradd element-demo
gpasswd -a element-demo sudo
```

The installer also requires that your non-root user has a home directory in `/home`.

EL Specific directions

Make sure to select "Container Management" in the "Additional Software" section.

Once you log in, please run:

```
sudo yum update -y
sudo yum install podman-docker python39-pip python39-devel make gcc -y
sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm -y
sudo update-alternatives --config python3
```

On the `update-alternatives` command, if you see more than one option, select the option with a command string of `/usr/bin/python3.9`.

The installer requires that you run it as a non-root user who has sudo permissions. Please make sure that you have a user who can use `sudo`. If you wanted to make a user called `element-demo` that can use `sudo`, the following commands (run as root) would achieve that:

```
useradd element-demo
gpasswd -a element-demo wheel
```

The installer also requires that your non-root user has a home directory in `/home`.

Migrating from our older installer

If you have previously used installer versions 2023-03.01 and earlier, you will need to run our migration script to convert your previous configuration to the new format that is used with our UI based installer. This script became available in 2023-03.02, so you must have at least that version or higher of the graphical installer for this to work.

NOTE: Before running the migration script, we highly recommend that you take a backup or snapshot of your working environment. While we have tested the migration script against several configurations at this point, we have not tested for all of the combinations of configuration that the previous installer allowed. We expect that migration will be a quick process for most customers, but in the event that something goes wrong, you'll want to be able to get back to a known good state through a backup or snapshot.

NB: If you are using group sync, you cannot presently migrate to the graphical installer. We are working to address the issues with migrating group sync and will remove this note once we have those addressed.

If you have not used our installer before, you may safely ignore this section.

To run the migration script, please do the following:

```
chmod +x ./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin
./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin --import ~/.element-
onpremise-config
```

Make sure to replace `~/.element-onpremise-config` with the path that your actual configuration exists in. Further, replace `YYYY-MM.VERSION` with the appropriate tag for the installer you downloaded.

Once the import has finished, the GUI will start and you will be able to browse to the installer at one of the provided URLs, much as if you had started the installer without doing a migration as detailed in the following section.

Network Specifics

Element Enterprise On-Premise needs to bind and serve content over:

- Port 80 TCP
- Port 443 TCP
- Port 8443 TCP (Installer GUI)

microk8s needs to bind and serve content over:

- Port 16443 TCP
- Port 10250 TCP
- Port 10255 TCP
- Port 25000 TCP
- Port 12379 TCP
- Port 10257 TCP
- Port 10259 TCP
- Port 19001 TCP

For more information, see <https://microk8s.io/docs/ports>.

In a default Ubuntu installation, these ports are allowed through the firewall. You will need to ensure that these ports are passed through your firewall.

For EL instances with firewalld enabled, the installer will take care of opening these ports for you.

Further, you need to make sure that your host is able to access the following hosts on the internet:

- api.snapcraft.io
- *.snapcraftcontent.com
- gitlab.matrix.org
- gitlab-registry.matrix.org
- pypi.org

- docker.io
- *.docker.com
- get.helm.sh
- k8s.gcr.io
- cloud.google.com
- storage.googleapis.com

In addition, you will also need to make sure that your host can access your distributions' package repositories. As these hostnames can vary, it is beyond the scope of this documentation to enumerate them.

Network Proxies

We also cover the case where you need to use a proxy to access the internet. Please see this article for more information: [Configuring a microk8s Single Node Instance to Use a Network Proxy](#)

Postgresql Database

The installation requires that you have a postgresql database with a locale of C and UTF8 encoding set up. See <https://github.com/matrix-org/synapse/blob/develop/docs/postgres.md#set-up-database> for further details.

If you have this already, please make note of the database name, user, and password as you will need these to begin the installation.

If you do not already have a database, then the single node installer will set up PostgreSQL on your behalf.

Beginning the Installation

Head to <https://ems.element.io/on-premise/download> and download the latest installer. The installer will be called `element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin`. You will take this file and copy it to the machine where you will be installing the Element Server Suite. Once you have this file on the machine in a directory accessible to your sudo-enabled user, you will run:

```
chmod +x ./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin
```

replacing the `YYYY-MM.VERSION` with the appropriate tag for the installer you downloaded.

Once you have done this, you will run:

```
./element-enterprise-graphical-installer-YYYY-MM.VERSION-gui.bin
```

replacing the `YYYY-MM.VERSION` with the appropriate tag for the installer you downloaded, and this will start a web server with the installer loaded.

You will see a message similar to:

```
[user@element-demo ~]$ ./element-enterprise-graphical-installer-2023-02.02-gui.bin
Testing network...
```

Using self-signed certificate with SHA-256 fingerprint:

F3: 76: B3: 2E: 1B: B3: D2: 20: 3C: CD: D0: 72: A3: 5E: EC: 4F: BC: 3E: F5: 71: 37: 0B: D7: 68: 36: 2E: 2C: AA: 7A: F2: 83: 94

To start configuration open:

`https://192.168.122.47:8443` or `https://10.1.185.64:8443` or `https://127.0.0.1:8443`

At this point, you will need to open a web browser and browse to one of these IPs. You may need to open port 8443 in your firewall to be able to access this address from a different machine.

If you are unable to open port 8443 or you are having difficulty connecting from a different machine, you may want to try ssh port forwarding in which you would run:

```
ssh <host> -L 8443:127.0.0.1:8443
```

replacing host with the IP address or hostname of the machine that is running the installer. At this point, with ssh connected in this manner, you should be able to use the `https://127.0.0.1:8443` link as this will then forward that request to the installer box via ssh.

Upon loading this address for the first time, you may be greeted with a message informing you that your connection isn't private such as this:



Your connection isn't private

Attackers might be trying to steal your information from **192.168.122.47** (for example, passwords, messages, or credit cards).

NET::ERR_CERT_AUTHORITY_INVALID

Advanced

Go back

In this case, you'll need to click "Advanced" and then "Continue to (unsafe)" in order to view the installer. As the exact button names and links can vary between browsers, it would be hard for us to document them all, so you may have slightly different wording depending on your browser.

The Hosts Screen

The very first page that you come to is the host screen.

element

Host.
Install

Standalone Kubernetes Application

Cert Manager
Cert manager settings
 Setup Cert Manager Skip Cert Manager

EMS Image Store
Your ems image store username / password

Username
382255a2-8d20-4a4a-aad7-2d3db34b7506

Token
.....

You will want to make sure that "Standalone" is selected. If you are using LetsEncrypt for your certificates, you will want to make sure that you select "Setup Cert Manager" and enter an email address for LetsEncrypt to associate with your certificates. If you are using custom certificates or electing to manage SSL certificates yourself, then you will want to select "Skip Cert Manager".

The very next prompt that you come to is for an EMS Image Store Username and Token. These are provided to you by element as access tokens for our enterprise container registries. If you have lost your token, you can always generate a new token at <https://ems.element.io/on-premise/subscriptions>.

MicroK8s

The MicroK8s settings UI

Persistent Volumes Path

Default

The host path where to store the persistent volumes. They will be hosted as subfolders of this path.

Registry Size

Default

The size of the registry in Gi

DNS Resolvers

The list of DNS resolvers to use

=

A DNS Server IP

Default

⊖

=

A DNS Server IP

Default

⊖

[ADD MORE DNS RESOLVERS](#)

The next option that you have is for microk8s. By default, microk8s will set up persistent volumes in `/data/element-deployment` and will allow 20GB of space to do this. For most installations, this is fine and can be left alone, but if you'd like to customize those options, you can do that here.

Next, we have DNS resolvers. The default DNS resolvers are Google (8.8.8.8 and 8.8.4.4). If you need to use your company's DNS servers, please change these values appropriately.

Postgres in Cluster

Configure this if you want to automatically create postgres servers in your microk8s cluster.

Postgres in Cluster External PostgreSQL Server

Host Path *

/data/postgres

Default

The host path where to store the postgres dbs. They will be hosted as subfolders of this path.

Passwords Seed

.....



The passwords seeds to use.

Connectivity

Connected Airgapped

Dockerhub

Optionally reduce rate limiting by providing your dockerhub credentials

Username

Password

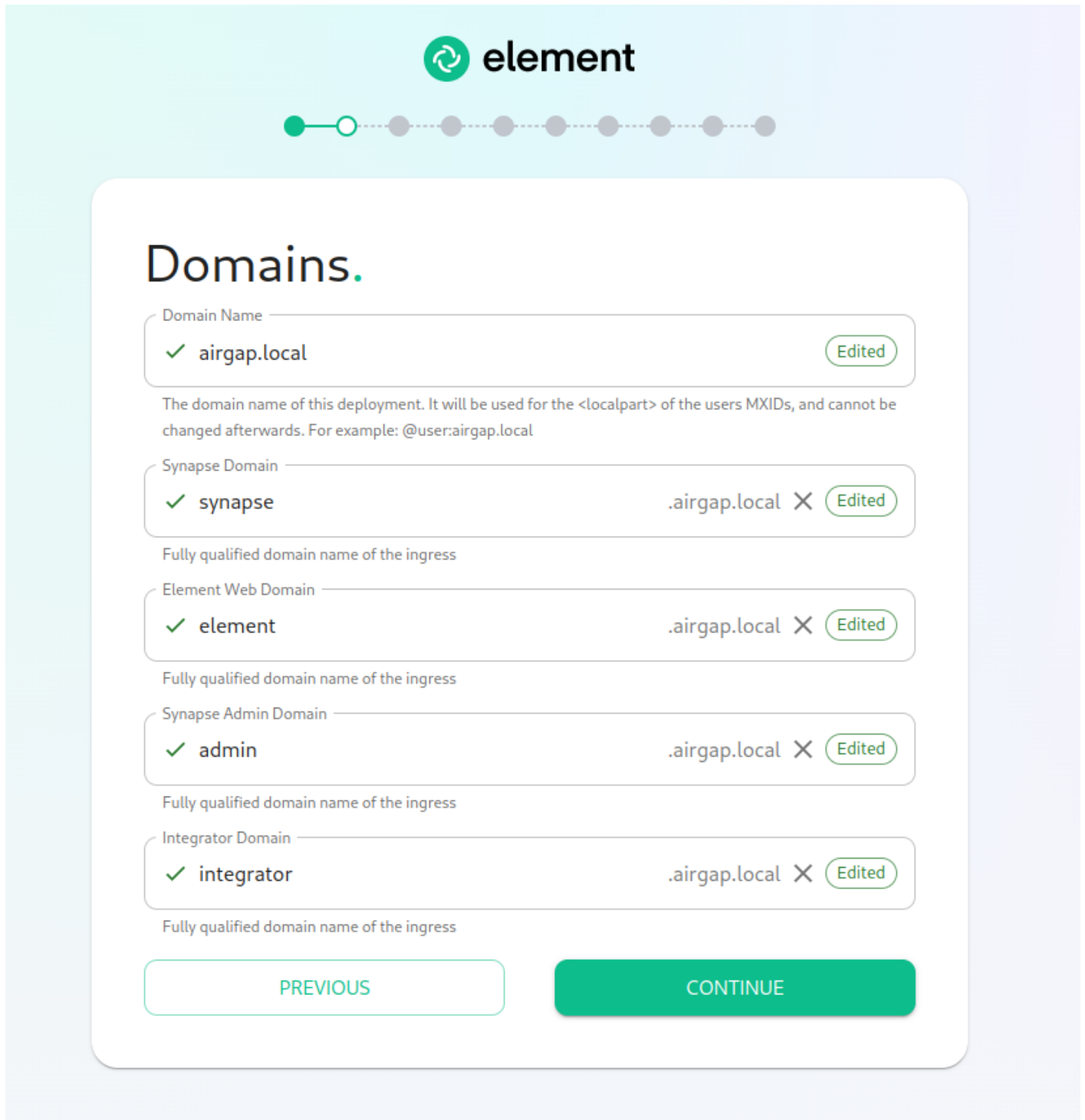


Next, we get the option to either have the installer install Postgres in your cluster or to use an external postgresql server. The Postgres in cluster option is only supported for our standalone installation and you should read our storage and backup guidelines for this configuration. At any rate, if you use the in cluster postgres, you will see that the installer defaults to /data/postgres and has generated a random password for your postgresql admin account. You can use the eye to see the password and you can certainly change this to whatever you'd like.

The final options on the hostpage are related to connectivity. For this guide, we are assuming "Connected" and you can leave that be. If you are doing "Airgapped", you would pick airgapped at this point and then please see the section on airgapped installations.

You are presented with the option to provide docker hub credentials. These are optional, but if you do not provide them, you may be rate limited by Docker and this could cause issues pulling container images.

The Domains Screen



On this page, we get to specify the domains for our installation. In this example, we have a domain name of airgap.local and this would mean our MXIDs would look like @kabbott:airgap.local.

Our domain page has checking to ensure that the host names resolve. Once you get green checks across the board, you can click continue.

The Certificates Screen

On the Certificates screen, you will provide SSL certificate information for well-known delegation, Synapse, Element Web, Synapse Admin, and Integrator.

2 options

Option 1: You already host a base domain `example.com` on a web server, then Well-Known Delegation should be set to `Externally Managed`.

Element clients need to be able to request `https://example.com/.well-known/matrix/client` to work properly.

The web server hosting the domain name should forward the requests to `.well-known/matrix/client` to the element enterprise server so that the wellKnownPod can serve it to the clients.

If that's not possible, the alternative is to copy the well known file directly on the example.com webserver. The wellKnownPod will still be present but won't be used by any system.

It cannot be set to `Certmanager / Letsencrypt`.

Option 2: You don't already host a base domain `example.com`, then the wellKnownPod hosts the well-known file and serves the base domain `example.com`.

You can choose those 3 different settings:

- `Certmanager / Letsencrypt`: the certificate for the base domain is signed by Letsencrypt
- `Certificate File`: the certificate is signed by your own CA or by a public CA (Verisign, Sectigo,..)
- `Existing TLS Certificate in the Cluster`: certificate already uploaded in a secret

If you are using Let's Encrypt, then each of the sections should look like:



Certificates.

Well-Known Delegation

airgap.local - The certificate used for well-known delegation

Certmanager / Let's Encrypt Certificate File Externally Managed

Let CertManager handle the certificate request.

Certmanager

The certmanager properties, if enabled

Issuer

letsencrypt

Default

The cert manager issuer selected

If you are using certificate files, then you will see a screen like:



Certificates.

Well-Known Delegation

airgap.local - The certificate used for well-known delegation

Certmanager / Let's Encrypt Certificate File Externally Managed

Upload a certificate and its private key.

Certificate

Certificate file

Secrets / Well Known Delegation /

Well Known Delegation Certificate ▾

Certificate No file chosen
WellKnownDelegation Certificate

Secrets / Well Known Delegation /

Well Known Delegation Private Key ▾

Private key No file chosen
WellKnownDelegation Private Key

which allows you to upload a .crt and .key file for each host. These files must be in PEM encoding. Our installer does accept wildcard certificates.

Once you have completed the certificate section for each host on the page, you may click continue.

The Database Screen

If you have elected to have the installer configure PostgreSQL for you, then you will not see this screen and can skip this section.

element

Database.

Database
PostgreSQL database name

Host
PostgreSQL database host

Port
5432 Default
PostgreSQL port

SSL Mode
Prefer Default
TLS settings to use for the Postgres connection

User
PostgreSQL username

Secrets / Synapse / Postgres Password

Postgresql Password
The postgres password

If you are using an external database, then you will see this page, where we provide the option to specify the database name, the database host name, the port to connect to, the SSL mode to use, and finally, the username and password to connect with.

If your database is installed on the same server where Element is installed, you have to enter the server public IP address since the container is not sharing the host network namespace. Entering 127.0.0.1 will resolve to the container itself and cause the installation failing.

Once you have completed this section, you may click continue.

The Media Screen

On this page, you can specify the size of your synapse media volume. Please leave "Create New Volume" checked and specify the size of the volume that you wish to allocate. You must have this space available in `/data/element-deployment` or whatever you specified back on the hosts screen. If you wish to create a 50G volume, you would need to specify `50Gi` for the Volume size.

The Cluster Screen

Most deployments can ignore this, however, if you want to change any microk8s cluster parameters, this is where to do it.

If you are in an environment where you have self-signed certificates, you will want to disable TLS verification, by clicking "Advanced" and then scrolling down and unchecking `Verify TLS`:

Verify TLS

TLS verification

Secrets / Global / CA.pem

Certificate authority Choose File No file chosen

Edited The CA to inject into the deployment.

Secrets / Global / Generic Shared Secret

Generic Shared Secret

The generic shared secret to use as a seed for all internally-generated secrets

Please bear in mind that disabling TLS verification and using self-signed certificates is not recommended for production deployments.

If your host names are not DNS resolvable, you need to use host aliases and this can be set up here. You will also click "Advanced" and scroll down to the "Host Aliases" section in "k8s". In here, you will click "Add Host Aliases" and then you will specify an IP and host names that resolve to that IP as such:

Host Aliases

The list of hosts aliases to configure on the pod spec. It should be avoid as much as possible to use this feature. Please prefer using an DNS entry to resolve your hostnames. This can be used as a workaround when entries cannot be resolved using DNS, for example for our automated testings.

IP *

192.168.122.47 Edited

An IP resolution to add to /etc/hosts

Hostnames

An hostname of the associated ip to add to /etc/hosts

airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

element.airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

synapse.airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

admin.airgap.local Edited

An hostname of the associated ip to add to /etc/hosts

integrator.airgap.local Edited

[ADD MORE HOSTNAMES](#)

[ADD MORE HOST ALIASES](#)

When you are finished with this page, you can click continue.

The Synapse Screen



Synapse.

This is a matrix homeserver.

Config

Accept Invites

Manual

Default

Whether to enable auto accept invites. Defaults to manual if not set

Max MAU Users

250

Default

Maximum number of Matrix Active Users

Registration

Closed

Default

Synapse registration

Secrets

/ Synapse

/ Admin Password

Admin Password

.....



Synapse admin user password

The first setting that you will see is whether you want to auto accept invites. The default of "Manual" will fit most use cases, but you are welcome to change this value.

The next setting is the maximum number of monthly active users (MAU) that you have purchased for your server. Your server will not allow you to go past this value. If you set this higher than your purchased MAU and you go over your purchased MAU, you will need to true up with Element to cover the cost of the unpaid users.

The next setting concerns registration. A server with open registration on the open internet can become a target, so we default to closed registration. You will notice that there is a setting called "Custom" and this requires explicit custom settings in the additional configuration section. Unless instructed by Element, you will not need the "Custom" option and should instead pick "Closed" or "Open" depending on your needs.

After this, you will see that the installer has picked an admin password for you. You will want to use the eye icon to view the password and copy this down as you will use this with the user `onprem-admin-donotdelete` to log into the admin panel after installation.

Telemetry

Telemetry properties

Enabled

True to enable telemetry

Default

Room

#element-telemetry Default

The telemetry room where to send telemetry

Advanced

Advanced

PREVIOUS CONTINUE

Continuing, we see telemetry. You should leave this enabled as you are required to report MAU to Element. In the event that you are installing into an environment without internet access, you may disable this so that it does not continue to try talking to Element. That said, you are still required to generate an MAU report at regular intervals and share that with Element.

For more information on the data that Element collects, please see: [What Telemetry Data is Collected by Element?](#)

Next, we have two advanced buttons. The top one is for synapse and gives you a text box to inject additional synapse configs (homeserver.yaml), access to the macaroon, registration shared secret, and signing key. Further, you can add and configure multiple synapse workers, external appservices, and federation. These topics are presently beyond the scope of this install guide, but we will write them up in due time. Further, you also have the ability to set a STUN shared secret and set TURN URIs for any existing TURN servers that you may have in your environment.

Additional

Additional config to inject

1

Secrets

/

Synapse

/

Macaroon

▼

Macaroon

.....



It should be generated randomly. It should never change.

Stun

Coturn configuration

HIDE

Secrets

/ Synapse

/ Stun Shared Secret

STUN Shared Secret

.....



It should be generated randomly and shared with the stun server.

Turn Uris

The stun servers

=



A stun server uri

ADD MORE TURN URIS

The second advanced button allows you to explicitly set any microk8s cluster settings that you would like just for the synapse pods. Most users will be able to ignore this.

You can hit continue to go to the next screen.

The Element Web Screen

Most users will be able to simply click "Continue" here.

The Advanced section allows you to set any custom element web configurations you would like (config.json).

Config

Additional configuration

Element web additional configuration.

1

K8s

Force values for components of Element Web

SHOW

PREVIOUS

CONTINUE

A common custom configuration would be configuring permalinks for Element, which we have documented here: [Setting up Permalinks With the Installer](#)

Further, it provides access to the k8s section, allowing you to explicitly set any microk8s cluster settings that you would like just for the element-web pod.

The Enterprise Admin Dashboard

Most users will be able to simply click "Continue" here. The Advanced section allows you to explicitly set any microk8s cluster settings that you would like just for the synapse-admin-ui pod.

One word to note here is that if you are not using delegated authentication, then the initial username that an administrator will use to log into this dashboard post-installation is `onpre-admin-donotdelete`. You can find the password for this user on the Synapse page in the "Admin Password" field.

If you are using delegated authentication, you will need to assign a user admin rights as detailed in this article: [How do I give a user admin rights when I am using delegated authentication and cannot log into the admin console?](#)

The Integrator Screen

The screenshot shows the 'Integrator' configuration page. At the top, it says 'Integrator.' followed by 'Send messages to external services'. Below this is a 'Config' section with a checkbox for 'Enable Custom Widgets' (unchecked) and a 'Default' button. Underneath is the 'Jitsi Domain' field with the value 'meet.element.io' and a 'Default' button. Below that is the 'Verify TLS' section with the option 'Use Global Setting' selected and a 'Default' dropdown. The 'Log' section is also visible, with 'Logging settings' and a 'Level' dropdown set to 'Info' (with a 'Default' dropdown). There is also a 'Structured' checkbox (unchecked) and a 'Default' button. The page has a light blue header and footer.

On this page, you can set up Integrator, the integrations manager.

The first option allows you to choose whether users can add custom widgets to their rooms with the integrator or not.

The next option allows you to specify which Jitsi instance the Jitsi widget will create conferences on.

The verify TLS option allows you to set this specifically for Integrator, regardless of what you set on the cluster screen.

The logging section allows you to set the log level and whether the output should be structured or not.

The Advanced section allows you to explicitly set any microk8s cluster settings that you would like just for the integrator pods.

Click "Continue to go to the next screen".

The Integrations Screen

This screen is where you can install any available integrations.

Some of these integrations will have "YAML" next to them. When you see this designation, this integration requires making settings in YAML, much like the old installer. However, with this installer, these YAML files are pre-populated and often only involve a few changes.

If you do not see a "YAML" designation next to the integration then this means that will use regular GUI elements to configure this integration.

Over time, we will do the work required to move the integrations with "YAML" next to them to the new GUI format.

For specifics on configuring well known delegation, please see [Setting Up Well Known Delegation](#)

For specifics on setting up Delegated Authentication, please see [Setting up Delegated Authentication With the Installer](#)

For specifics on setting up Group Sync, please see [Setting up Group Sync with the Installer](#)

For specifics on setting up GitLab, GitHub, and JIRA integrations, please see [Setting up GitLab, GitHub, and JIRA Integrations With the Installer](#)

For specifics on setting up Adminbot and Auditbot, please see: [Setting up Adminbot and Auditbot](#)

For specifics on setting up Hydrogen, please see: [Setting Up Hydrogen](#)

For specifics on pointing your installation at an existing Jitsi instance, please see [Setting Up Jitsi and TURN With the Installer](#)

If you do not have an existing TURN server or Jitsi server, our installer can configure these for you by following the extra steps in [Setting Up Jitsi and TURN With the Installer](#)

For specifics on configuring the Teams Bridge, please see [Setting Up the Teams Bridge](#)

For specifics on configuring the Telegram Bridge, please see [Setting Up the Telegram Bridge](#)

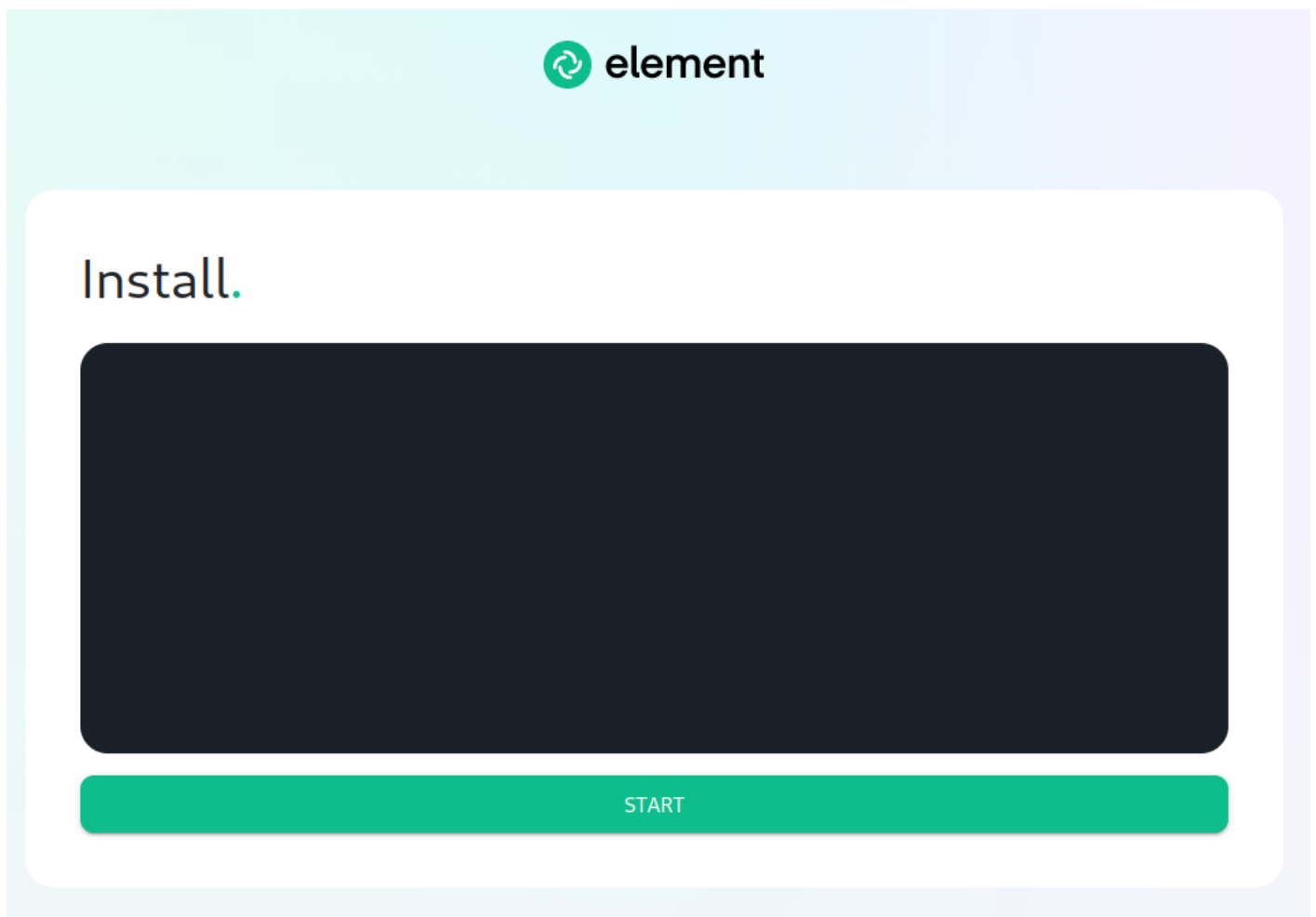
For specifics on configuring the IRC Bridge, please see [Setting Up the IRC Bridge](#)

For specifics on configuring the XMPP Bridge, please see [Setting Up the XMPP Bridge](#)

Once you have configured all of the integrations that you would like to configure, you can click "Continue" to head to the installation screen.

The Installation Screen

On the installation screen, you should see a blank console and a start button:



Click Start.

After a moment, you will notice the installer hang. If you go back to the prompt where you are running the installer, you will see that you are being asked for the sudo password:

Install.



```
[karll@airgap ~]$ ./element-enterprise-graphical-installer-2023-02.02-gui-rc1.bin
Testing network...

Using self-signed certificate with SHA-256 fingerprint:
    CB:8E:4A:75:80:32:D5:E0:A0:2C:90:A7:DF:F9:2F:9F:6D:14:F7:18:53:D0:C5:6C:20:D5:95:A8:1A:57:67:21

To start configuration open:
    https://192.168.122.47:8443 or https://10.1.185.64:8443 or https://127.0.0.1:8443
API resolved without sending a response for /api/logs, this may result in stalled requests.
[sudo] password for karll: 
```

Go ahead and enter the sudo password and the installation will continue.

On the very first time that you run the installer, you will be prompted to log out and back in again to allow Linux group membership changes to be refreshed. This means that you will need to issue a ctrl-C in the terminal running your installer and actually log all the way out of your Linux session, log back in, restart the installer, navigate back to the installer screen, click start again, and then re-enter your sudo password. You will only have

to perform this step once per server.

Verifying Your Installation

Once the installation has finished, it can take as much as 15 minutes on a first run for everything to be configured and set up. If you use:

```
kubectl get pods -n element-onprem
```

You should see similar output to:

NAME	READY	STATUS	RESTARTS	AGE
app-element-web-c5bd87777-rqr6s	1/1	Running	1	29m
server-well-known-8c6bd8447-wddtm	1/1	Running	1	29m
postgres-0	1/1	Running	1	40m
instance-synapse-main-0	1/1	Running	2	29m
instance-synapse-haproxy-5b4b55fc9c-hnlmp	1/1	Running	0	20m

Once the admin console is up and running:

```
first-element-deployment-synapse-admin-ui-564cbf5665-dn8nv 1/1 Running
1 (4h4m ago) 3d1h
```

and synapse:

```
first-element-deployment-synapse-redis-59548698df-gqkcq 1/1 Running
1 (4h4m ago) 3d2h
first-element-deployment-synapse-haproxy-7587dfd6f7-gp6wh 1/1 Running
2 (4h3m ago) 2d23h
first-element-deployment-synapse-appservice-0 1/1 Running
3 (4h3m ago) 3d
first-element-deployment-synapse-main-0 1/1 Running
0 3h19m
```

then you should be able to log in at your admin panel (in our case <https://admin.airgap.local/>) with the `onprem-admin-donotdelete` user and the password that was specified on the "Synapse" screen.

A word about Configuration Files

In the new installer, all configuration files are placed in the directory `.element-enterprise-server`. This can be found in your user's home directory. In this directory, you will find a subdirectory called `config` that contains the

actual configurations.

Running the Installer without the GUI

It is possible to run the installer without using the GUI provided that you have a valid set of configuration files in the `.element-enterprise-server/config` directory. Directions on how to do this are available at: <https://ems-docs.element.io/books/ems-knowledge-base/page/how-do-i-run-the-installer-without-using-the-gui>. Using this method, you could use the GUI as a configuration editor and then take the resulting configuration and modify it as needed for further installations.

This method also makes it possible to set things up once and then run future updates without having to use the GUI.

End-User Documentation

After completing the installation you can share our User Guide to help orient and onboard your users to Element!

Single Node Installs: Storage and Backup Guidelines

General storage recommendations for single-node instances

- `/mnt` (or a common root for all `<component>_data_path` variables) should be a distinct mount point
 - Ideally this would have an independent lifecycle from the server itself
 - Ideally this would be easily snapshot-able, either at a filesystem level or with the backing storage

Adminbot storage:

- Files stored with uid=10006/gid=10006, sample config uses `/mnt/data/adminbot` for single-node instances
 - The backing path for single node instances can be changed by setting `bot_data_path` in the `adminbot` config directory
- Storage space required is proportional to the number of user devices on the server. 1GB is sufficient for most servers
 - The size of the PVC can be changed by setting `bot_data_size` in the `adminbot` config directory

Auditbot storage:

- Files stored with uid=10006/gid=10006, sample config uses `/mnt/data/auditbot` for single-node instances
 - The backing path for single node instances can be changed by setting `bot_data_path` in the `auditbot` config directory
- Storage space required is proportional to the number of events tracked. 1GB is sufficient with the sample config `logfile_size` / `logfile_keep` values
 - The size of the PVC can be changed by setting `bot_data_size` in the `auditbot` config directory

Dimension storage :

- Main:
 - File stored with uid=10005/gid=1000, sample config uses `/mnt/dimension` for single-node instances
 - The backing path for single node instances can be changed by setting `bot_data_path` in the `dimension` config directory
 - Storage space is constant to store a single file. 10M is sufficient for every server
 - The size of the PVC can be changed by setting `bot_data_size` in the `dimension` config directory
- Postgres (in-cluster):

- Files stored with uid=999/gid=999, sample config does not specify a default path for single-node instances
 - The backing path for single node instances can be changed by setting `postgres_data_path` in the `dimension` config directory
- Storage space is proportional to the number of integration instances. 5GB is sufficient for most servers
 - The size of the PVC can be changed by setting `postgres_storage_size` in the `dimension` directory folder

Synapse storage:

- **Media:**
 - File stored with uid=10991/gid=10991, sample config uses `/mnt/data/synapse-media` for single-node instances
 - The backing path for single node instances can be changed by setting `media_host_data_path` in `parameters.yml`
 - Storage space required grows with the number and size of uploaded media. 50GB is used as a starting point for PoC but can easily be exceeded depending on your use-case
 - The size of the PVC can be changed by setting `media_size` in `parameters.yml`

Postgres (in-cluster) storage:

- Files stored with uid=999/gid=999, sample config uses `/mnt/data/synapse-postgres` for single-node instances
 - The backing path for single node instances can be changed by setting `postgres_data_path` in `parameters.yml`
- Storage space is proportional to the activity on the homeserver. 5GB is used as a starting point for PoC but can easily be exceeded depending on traffic
 - The size of the PVC can be changed by setting `postgres_storage_size` in `parameters.yml`

Backup Guidance:

- **Adminbot:**
 - Backups should be made by taking a snapshot of the PV (ideally) or rsyncing the backing directory to backup storage
- **Auditbot:**
 - Backups should be made by taking a snapshot of the PV (ideally) or rsyncing the backing directory to backup storage
- **Dimension:**
 - Backups should be made by taking a snapshot of the PV (ideally) or rsyncing the backing directory to backup storage
- **Synapse Media:**
 - Backups should be made by taking a snapshot of the PV (ideally) or rsyncing the backing directory to backup storage
- **Postgres (in-cluster):**
 - Backups should be made by `kubectl -n element-onprem exec -it postgres-synapse-0 -- sh -c 'pg_dump -U $POSTGRES_USER $POSTGRES_DB' > synapse_postgres_backup_$(date +%Y%m%d-%H%M%S).sql`
- **Postgres (external):**
 - Backup procedures as per your DBA
- **Configuration:**

- Please ensure that your entire configuration directory (that contains at least `parameters.yml` & `secrets.yml` but may also include other sub-directories & configuration files) is regularly backed up
 - The suggested configuration path in Element's documentation is `~/.element-onpremise-config` but could be anything. It is whatever directory you used with the installer.

Using the Installer in an Air-Gapped Environment

Defining Air-Gapped Environments

An air-gapped environment is any environment in which the running hosts will not have access to the greater internet. This proposes a situation in which these hosts are unable to get access to various needed bits of software from Element and also are unable to share telemetry data back with Element.

For some of these environments, they can be connected to the internet from time to time and updated during those connection periods. In other environments, the hosts are never connected to the internet and everything must be moved over sneaker net.

This guide will cover running the microk8s installer when only sneaker net is available as that is the most restrictive of these environments.

Preparing the media to sneaker net into the air gapped environment

You will need our airgapped dependencies .tar.gz file which you can get from Element:

- `element-enterprise-installer-airgapped-<version>-gui.tar.gz`

Running the installer in the air gapped environment

Extract the airgapped dependencies to a directory on the machine you are installing from. You obtain the following directories :

- `airgapped/pip`
- `airgapped/galaxy`
- `airgapped/snaps`
- `airgapped/containerd`
- `airgapped/images`

Your airgapped machine will still require access to airgapped linux repositories depending on your OS. If using Red Hat Enterprise Linux, you will also need access to the EPEL repository in your airgapped environment.

Connectivity

Connected Airgapped

Airgapped

Airgapped settings to automatically configure images

Local Registry

localhost:32000

Default

The local registry url to find airgapped images

Source Directory

The path to the airgapped directory to upload the images

CONTINUE

When using the installer, select "Airgapped" on the first hosts screen.

The Local Registry parameter should be left alone unless you have a separate custom registry that you would like to use.

For the Source directory, you need to specify the absolute path to the `airgapped` directory that was extracted from the tarball.

The installer will upload the images automatically to your local registry, and use these references to start the workloads.

If you are doing a kubernetes installation (instead of a single node installation), please note that once the image upload has been done, you will need to copy the `airgapped/images/images_digests.yml` file to the same path on the machine which will be used to render or deploy element services. Doing this, the new image digests will be used correctly in the kubernetes manifests used for deployment.

Troubleshooting

Introduction to Troubleshooting

Troubleshooting the Element Installer comes down to knowing a little bit about kubernetes and how to check the status of the various resources. This guide will walk you through some of the initial steps that you'll want to take when things are going wrong.

Known issues

Airgapped installation does not start

If you are using `element-enterprise-graphical-installer-2023-03.02-gui.bin` and `element-enterprise-installer-airgapped-2023-03.02-gui.tar.gz`. You might run into an error looking like this :

```
Looking in links: ./airgapped/pip
```

```
WARNING: Url './airgapped/pip' is ignored. It is either a non-existing path or lacks a specific scheme.
```

```
ERROR: Could not find a version that satisfies the requirement wheel (from versions: none)
```

```
ERROR: No matching distribution found for wheel
```

The workaround for it is to copy the pip folder from the airgapped directory to `~/element-enterprise-server/installer/airgapped/pip`

install.sh problems

Sometimes there will be problems when running the `ansible-playbook` portion of the installer. When this happens, you can increase the verbosity of ansible logging by editing `.ansible.rc` in the installer directory and setting:

```
export ANSIBLE_DEBUG=true
export ANSIBLE_VERBOSE=4
```

and re-running the installer. This will generate quite verbose output, but that typically will help pinpoint what the actual problem with the installer is.

Problems post-installation

Checking Pod Status and Getting Logs

- In general, a well-functioning Element stack has at it's minimum the following containers (or pods in kubernetes language) running:

```
[user@element2 ~]$ kubectl get pods -n element-onprem
kubectl get pods -n element-onprem
```

NAME	READY	STATUS
first-element-deployment-element-web-6cc66f48c5-lvd7w 0 4d20h	1/1	Running
first-element-deployment-element-call-c9975d55b-dzjw2 0 4d20h	1/1	Running
integrator-postgres-0 0 4d20h	3/3	Running
synapse-postgres-0 0 4d20h	3/3	Running
first-element-deployment-integrator-59bcfc67c5-jkbm6 0 4d20h	3/3	Running
adminbot-admin-app-element-web-c9d456769-rpk9l 0 4d20h	1/1	Running
auditbot-admin-app-element-web-5859f54b4f-8lbng 0 4d20h	1/1	Running
first-element-deployment-synapse-redis-68f7bfbdc-why9m 0 4d20h	1/1	Running
first-element-deployment-synapse-haproxy-7f66f5fdf5-8sfkf 0 4d20h	1/1	Running
adminbot-pipe-0 0 4d20h	1/1	Running
auditbot-pipe-0 0 4d20h	1/1	Running
first-element-deployment-synapse-admin-ui-564bb5bb9f-87zb4 0 4d20h	1/1	Running
first-element-deployment-groupsnc-0 0 20h	1/1	Running
first-element-deployment-well-known-64d4cfd45f-l9kk 0 20h	1/1	Running

```

first-element-deployment-synapse-main-0          1/1    Running
0          20h
first-element-deployment-synapse-appservice-0    1/1    Running
0          20h

```

The above `kubectl get pods -n element-onprem` is the first place to start. You'll notice in the above, all of the pods are in the `Running` status and this indicates that all should be well. If the state is anything other than "Running" or "Creating", then you'll want to grab logs for those pods. To grab the logs for a pod, run:

```
kubectl logs -n element-onprem <pod name>
```

replacing `<pod name>` with the actual pod name. If we wanted to get the logs from synapse, the specific syntax would be:

```
kubectl logs -n element-onprem first-element-deployment-synapse-main-0
```

and this would generate logs similar to:

```

2022-05-03 17:46:33,333 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2887 - Dropped 0 items from caches
2022-05-03 17:46:33,375 - synapse.storage.databases.main.metrics - 471 - INFO -
generate_user_daily_visits-289 - Calling _generate_user_daily_visits
2022-05-03 17:46:58,424 - synapse.metrics._gc - 118 - INFO - sentinel - Collecting gc
1
2022-05-03 17:47:03,334 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2888 - Dropped 0 items from caches
2022-05-03 17:47:33,333 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2889 - Dropped 0 items from caches
2022-05-03 17:48:03,333 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2890 - Dropped 0 items from caches

```

- Again, for every pod not in the `Running` or `Creating` status, you'll want to use the above procedure to get the logs for Element to look at.
- If you don't have any pods in the `element-onprem` namespace as indicated by running the above command, then you should run:

```

[user@element2 ~]$ kubectl get pods -A
NAMESPACE          NAME                                READY   STATUS
RESTARTS   AGE
kube-system        calico-node-2lznr                  1/1     Running
0             8d
kube-system        calico-kube-controllers-c548999db-s5cjm 1/1     Running
0             8d
kube-system        coredns-5dbccd956f-glc8f           1/1     Running

```

0	8d			
kube-system		dashboard-metrics-scraper-6b6f796c8d-8x6p4	1/1	Running
0	8d			
ingress		nginx-ingress-microk8s-controller-w8lcn	1/1	Running
0	8d			
cert-manager		cert-manager-cainjector-6586bddc69-9xwkj	1/1	Running
0	8d			
kube-system		hostpath-provisioner-78cb89d65b-djfq5	1/1	Running
0	8d			
kube-system		kubernetes-dashboard-765646474b-5lhxp	1/1	Running
0	8d			
cert-manager		cert-manager-5bb9dd7d5d-cg9h8	1/1	Running
0	8d			
container-registry		registry-f69889b8c-zkhm5	1/1	Running
0	8d			
cert-manager		cert-manager-webhook-6fc8f4666b-9tmjb	1/1	Running
0	8d			
kube-system		metrics-server-5f8f64cb86-f876p	1/1	Running
0	8d			
jitsi		sysctl-jvb-vs9mn	1/1	Running
0	8d			
jitsi		shard-0-jicofo-7c5cd9fff5-qrzmk	1/1	Running
0	8d			
jitsi		shard-0-web-fdd565cd6-v49ps	1/1	Running
0	8d			
jitsi		shard-0-web-fdd565cd6-wmzpb	1/1	Running
0	8d			
jitsi		shard-0-prosody-6d466f5bc6-5qsbb	1/1	Running
0	8d			
jitsi		shard-0-jvb-0	1/2	Running
0	8d			
operator-onprem		element-operator-controller-manager-...	2/2	Running
0	4d			
updater-onprem		element-updater-controller-manager-...	2/2	Running
0	4d			
element-onprem		first-element-deployment-element-web-...	1/1	Running
0	4d			
element-onprem		first-element-deployment-element-call-...	1/1	Running
0	4d			

element-onprem-0	integrator-postgres-0	3/3	Running
element-onprem-0	synapse-postgres-0	3/3	Running
element-onprem-0	first-element-deployment-integrator-...	3/3	Running
element-onprem-0	adminbot-admin-app-element-web-...	1/1	Running
element-onprem-0	auditbot-admin-app-element-web-...	1/1	Running
element-onprem-0	first-element-deployment-synapse-redis-...	1/1	Running
element-onprem-0	first-element-deployment-synapse-haproxy-..	1/1	Running
element-onprem-0	adminbot-pipe-0	1/1	Running
element-onprem-0	auditbot-pipe-0	1/1	Running
element-onprem-0	first-element-deployment-synapse-admin-ui-.	1/1	Running
element-onprem-0	first-element-deployment-groupsinc-0	1/1	Running
element-onprem-0	first-element-deployment-well-known-...	1/1	Running
element-onprem-0	first-element-deployment-synapse-main-0	1/1	Running
element-onprem-0	first-element-deployment-synapse-appservice-0	1/1	Running

- This is the output from a healthy system, but if you have any of these pods not in the `Running` or `Creating` state, then please gather logs using the following syntax:

```
kubectl logs -n <namespace> <pod name>
```

- So to gather logs for the kubernetes ingress, you would run:

```
kubectl logs -n ingress nginx-ingress-microk8s-controller-w8lcn
```

and you would see logs similar to:

```

I0502 14:15:08.467258      6 leaderelection.go:248] attempting to acquire leader
lease ingress/ingress-controller-leader...
I0502 14:15:08.467587      6 controller.go:155] "Configuration changes detected,
backend reload required"
I0502 14:15:08.481539      6 leaderelection.go:258] successfully acquired lease
ingress/ingress-controller-leader
I0502 14:15:08.481656      6 status.go:84] "New leader elected" identity="nginx-
ingress-microk8s-controller-n6wmk"
I0502 14:15:08.515623      6 controller.go:172] "Backend successfully reloaded"
I0502 14:15:08.515681      6 controller.go:183] "Initial sync, sleeping for 1
second"
I0502 14:15:08.515705      6 event.go:282] Event(v1.ObjectReference{Kind: "Pod",
Namespace: "ingress", Name: "nginx-ingress-microk8s-controller-n6wmk", UID: "548d9478-
094e-4a19-ba61-284b60152b85", APIVersion: "v1", ResourceVersion: "524688",
FieldPath: ""}): type: 'Normal' reason: 'RELOAD' NGINX reload triggered due to a
change in configuration

```

Again, for all pods not in the `Running` or `Creating` state, please use the above method to get log data to send to Element.

Other Commands of Interest

Some other commands that may yield some interesting data while troubleshooting are:

- **Verify DNS names and IPs in certificates**

In the `certs` directory under the configuration directory, run:

```
for i in $(ls *cert); do echo $i && openssl x509 -in $i -noout -text | grep DNS; done
```

This will give you output similar to:

```

local.crt
          DNS: local, IP Address: 192.168.122.118, IP Address: 127.0.0.1
synapse2.local.crt
          DNS: synapse2.local, IP Address: 192.168.122.118, IP Address: 127.0.0.1

```

and this will allow you to verify that you have the right host names and IP addresses in your certificates.

- **Show hostname to IP mappings from within a pod**

Run:

```
kubectl exec -n element-onprem <pod_name> -- getent hosts
```

and you will see output similar to:


```

127.0.0.1      localhost
127.0.0.1      localhost ip6-localhost ip6-loopback
10.1.241.30    instance-hookshot-0.instance-hookshot.element-
onprem.svc.cluster.local instance-hookshot-0
192.168.122.5  ems.onprem.element.ems.onprem.hs.ems.onprem.adminbot.ems.onprem
auditbot.ems.onprem.integrator.ems.onprem.hookshot.ems.onprem.admin.ems.onprem
eleweb.ems.onprem

```

This will help you troubleshoot host resolution.

- **Show all persistent volumes and persistent volume claims for the `element-onprem` namespace:**

```
kubectl get pv -n element-onprem
```

This will give you output similar to:

NAME	CAPACITY	ACCESS MODES	RECLAIM
POLICY	STATUS	CLAIM	
STORAGECLASS	REASON	AGE	
pvc-fc3459f0-eb62-4afa-94ce-7b8f8105c6d1	20Gi	RWX	
Delete	Bound	container-registry/registry-claim	
microk8s-hostpath		8d	
integrator-postgres	5Gi	RWO	
Recycle	Bound	element-onprem/integrator-postgres	
microk8s-hostpath		8d	
synapse-postgres	5Gi	RWO	
Recycle	Bound	element-onprem/synapse-postgres	
microk8s-hostpath		8d	
hostpath-synapse-media	50Gi	RWO	
Recycle	Bound	element-onprem/first-element-deployment-synapse-media	
microk8s-hostpath		8d	
adminbot-bot-data	10M	RWO	
Recycle	Bound	element-onprem/adminbot-bot-data	
microk8s-hostpath		8d	
auditbot-bot-data	10M	RWO	
Recycle	Bound	element-onprem/auditbot-bot-data	
microk8s-hostpath		8d	

- ****Show the synapse configuration:****

For installers prior to 2022-05.06, use:

```
```bash
kubectl describe cm -n element-onprem first-element-deployment-synapse-shared
```

and this will return output similar to:

```
send_federation: True
start_pushers: True
turn_allow_guests: true
turn_shared_secret: n0t4ctuAllymatr1Xd0TorgSshar3d5ecret4obvIousreAsons
turn_uris:
- turns: turn.matrix.org?transport=udp
- turns: turn.matrix.org?transport=tcp
turn_user_lifetime: 86400000
```

For the 2022-05.06 installer and later, use:

```
kubectl -n element-onprem get secret synapse-secrets -o yaml 2>&1 | grep shared.yaml | awk -F
'shared.yaml:' '{print $2}' - | base64 -d
```

For the 2023-05.05 installer and later, use:

```
kubectl get secrets/first-element-deployment-synapse-secrets -n element-onprem -o yaml | grep
shared.yaml | awk '{ print $2}' | base64 -d
```

and you will get output similar to the above.

- **Show the Element Web configuration:**

```
kubectl describe cm -n element-onprem app-element-web
```

and this will return output similar to:

```
config.json:

{
 "default_server_config": {
 "m.homeserver": {
 "base_url": "https://synapse2.local",
 "server_name": "local"
 }
 },
 "dummy_end": "placeholder",
 "integrations_jitsi_widget_url": "https://dimension.element2.local/widgets/jitsi",
```

```
"integrations_rest_url": "https://dimension.element2.local/api/v1/scalar",
"integrations_ui_url": "https://dimension.element2.local/element",
"integrations_widgets_urls": [
 "https://dimension.element2.local/widgets"
]
}
```

- **Show the nginx configuration for Element Web: (If using nginx as your ingress controller in production or using the PoC installer.)**

```
kubectl describe cm -n element-onprem app-element-web-nginx
```

and this will return output similar to:

```
server {
 listen 8080;

 add_header X-Frame-Options SAMEORIGIN;
 add_header X-Content-Type-Options nosniff;
 add_header X-XSS-Protection "1; mode=block";
 add_header Content-Security-Policy "frame-ancestors 'self' ";
 add_header X-Robots-Tag "noindex, nofollow, noarchive, noimageindex";

 location / {
 root /usr/share/nginx/html;
 index index.html index.htm;

 charset utf-8;
 }
}
```

- **Check list of active kubernetes events:**

```
kubectl get events -A
```

You will see a list of events or the message `No resources found`.

- **Show the state of services in the `element-onprem` namespace:**

```
kubectl get services -n element-onprem
```

This should return output similar to:

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
PORT(S)	AGE		
postgres	ClusterIP	10.152.183.47	<none>

5432/TCP	6d23h			
app-element-web		ClusterIP	10.152.183.60	<none>
80/TCP	6d23h			
server-well-known		ClusterIP	10.152.183.185	<none>
80/TCP	6d23h			
instance-synapse-main-headless		ClusterIP	None	<none>
80/TCP	6d23h			
instance-synapse-main-0		ClusterIP	10.152.183.105	<none>
80/TCP, 9093/TCP, 9001/TCP	6d23h			
instance-synapse-haproxy		ClusterIP	10.152.183.78	<none>
80/TCP	6d23h			

- Show the status of the stateful sets in the `element-onprem` namespace:

```
kubectl get sts -n element-onprem
```

This should return output similar to:

NAME	READY	AGE
postgres	1/1	6d23h
instance-synapse-main	1/1	6d23h

- Show deployments in the `element-onprem` namespace:

```
kubectl get deploy -n element-onprem
```

This will return output similar to:

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
app-element-web	1/1	1	1	6d23h
server-well-known	1/1	1	1	6d23h
instance-synapse-haproxy	1/1	1	1	6d23h

- Show the status of all namespaces:

```
kubectl get namespaces
```

which will return output similar to:

NAME	STATUS	AGE
kube-system	Active	20d
kube-public	Active	20d
kube-node-lease	Active	20d
default	Active	20d
ingress	Active	6d23h

```
container-registry Active 6d23h
operator-onprem Active 6d23h
element-onprem Active 6d23h
```

- **View the MAU Settings in Synapse:**

```
kubectl get -n element-onprem secrets/synapse-secrets -o yaml | grep -i shared.yaml
-m 1| awk -F ': ' '{print $2}' - | base64 -d
```

which will return output similar to:

```
Local custom settings
mau_stats_only: true

limit_usage_by_mau: False
max_mau_value: 1000
mau_trial_days: 2

mau_appservice_trial_days:
 chatterbox: 0

enable_registration_token_3pid_bypass: true
```

- **Redeploy the microk8s setup**

It is possible to redeploy microk8s by running the following command as root:

```
snap remove microk8s
```

This command does remove all microk8s pods and related microk8s storage volumes. **Once this command has been run, you need to reboot your server** - otherwise you may have networking issues. Add `--purge` flag to remove the data if disk usage is a concern.

After the reboot, you can re-run the installer and have it re-deploy microk8s and Element Enterprise On-Premise for you.

## Node-based pods failing name resolution

```
05:03:45:601 ERROR [Pipeline] Unable to verify identity configuration for bot-auditbot:
Unknown errcode Unknown error
05:03:45:601 ERROR [Pipeline] Unable to verify identity. Stopping
matrix-pipe encountered an error and has stopped Error: getaddrinfo EAI_AGAIN
synapse.prod.ourdomain
 at GetAddrInfoReqWrap.onlookup [as oncomplete] (node:dns:84:26) {
 errno: -3001,
 code: 'EAI_AGAIN',
 syscall: 'getaddrinfo',
```

```
hostname: 'synapse.prod.ourdomain'
}
```

To see what Hosts are set, try:

```
kubectl exec -it -n element-onprem <pod name> getent hosts
```

So to do this on the adminbot-pipe-0 pod, it would look like:

```
kubectl exec -it -n element-onprem adminbot-pipe-0 getent hosts
```

and return output similar to:

```
127.0.0.1 localhost
127.0.0.1 localhost ip6-localhost ip6-loopback
10.1.241.27 adminbot-pipe-0
192.168.122.5 ems.onprem element.ems.onprem hs.ems.onprem adminbot.ems.onprem
auditbot.ems.onprem integrator.ems.onprem hookshot.ems.onprem admin.ems.onprem
eleweb.ems.onprem
```

## Node-based pods failing SSL

```
2023-02-06 15:42:04 ERROR: IrcBridge Failed to fetch roomlist from joined rooms: Error: unable
to verify the first certificate. Retrying
MatrixHttpClient (REQ-13) Error: unable to verify the first certificate
at TLSSocket.onConnectSecure (_tls_wrap.js:1515:34)
at TLSSocket.emit (events.js:400:28)
at TLSSocket.emit (domain.js:475:12)
at TLSSocket.finishInit (_tls_wrap.js:937:8),
at TLSWrap.ssl.onhandshakedone (_tls_wrap.js:709:12) {
 code: 'UNABLE TO VERIFY LEAF SIGNATURE'
}
```

Drop into a shell on the pod

```
kubectl exec -it -n element-onprem adminbot-pipe-0 -- /bin/sh
```

Check it's ability to send a request to the Synapse server

```
node

require("http")
```

```
request(https://synapse.server/)
```

## Default administrator

The installer creates a default administrator `onpre-admin-donotdelete`. The Synapse admin user password is defined under the synapse section in the installer.

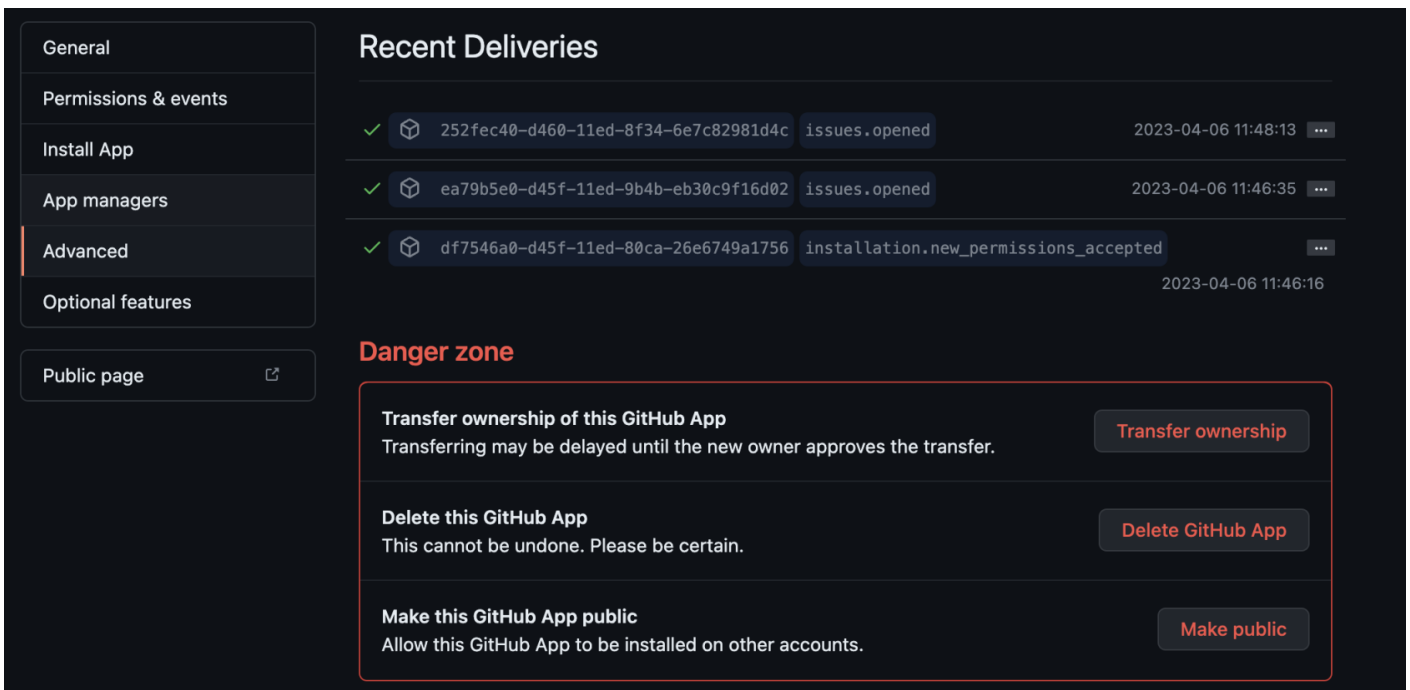
## Integration issues

### GitHub not sending events

You can trace webhook calls from your GitHub application under `Settings` / `developer settings` / `GitHub Apps`.

Select your GitHub App.

Click on `Advanced` and you should see queries/issues by your app under `Recent Deliveries`.



The screenshot shows the GitHub App settings interface. On the left is a sidebar with navigation options: General, Permissions & events, Install App, App managers, Advanced (highlighted), Optional features, and Public page. The main content area is titled 'Recent Deliveries' and contains a table of events:

Event Type	Timestamp
issues.opened	2023-04-06 11:48:13
issues.opened	2023-04-06 11:46:35
installation.new_permissions_accepted	2023-04-06 11:46:16

Below the table is a 'Danger zone' section with three actions:

- Transfer ownership of this GitHub App**: Transferring may be delayed until the new owner approves the transfer. [Transfer ownership](#)
- Delete this GitHub App**: This cannot be undone. Please be certain. [Delete GitHub App](#)
- Make this GitHub App public**: Allow this GitHub App to be installed on other accounts. [Make public](#)

## Updater and Operator in `ImagePullBackOff` state

Check EMS Image Store Username and Token

Check to see if you can pull the Docker image:

```
kubectl get pods -l app.kubernetes.io/instance=element-operator-controller-manager -n operator-onprem -o yaml | grep 'image:'
```

grab the entry like `image: gitlab-registry.matrix.org/ems-image-store/standard/kubernetes-operator@sha256:305c7ae51e3b3fbbeff8abf2454b47f86d676fa573ec13b45f8fa567dc02fcd1`

Should look like

```
microk8s.ctr image pull gitlab-registry.matrix.org/ems-image-store/standard/kubernetes-operator@sha256:305c7ae51e3b3fbbeff8abf2454b47f86d676fa573ec13b45f8fa567dc02fcd1 -u <EMS Image Store usenamer>: <EMS Image Store token>
```



# Setting up Permalinks With the Installer

## Element Extra Configurations

### Config

#### Additional configuration

Element web additional configuration.

1

### K8s

Force values for components of Element Web

SHOW

PREVIOUS

CONTINUE

Please go to the "Element Web" page of the installer, click on "Advanced" and add the following to "Additional Configuration":

```
{
 "permalinkPrefix": "https://<element fqdn>"
}
```

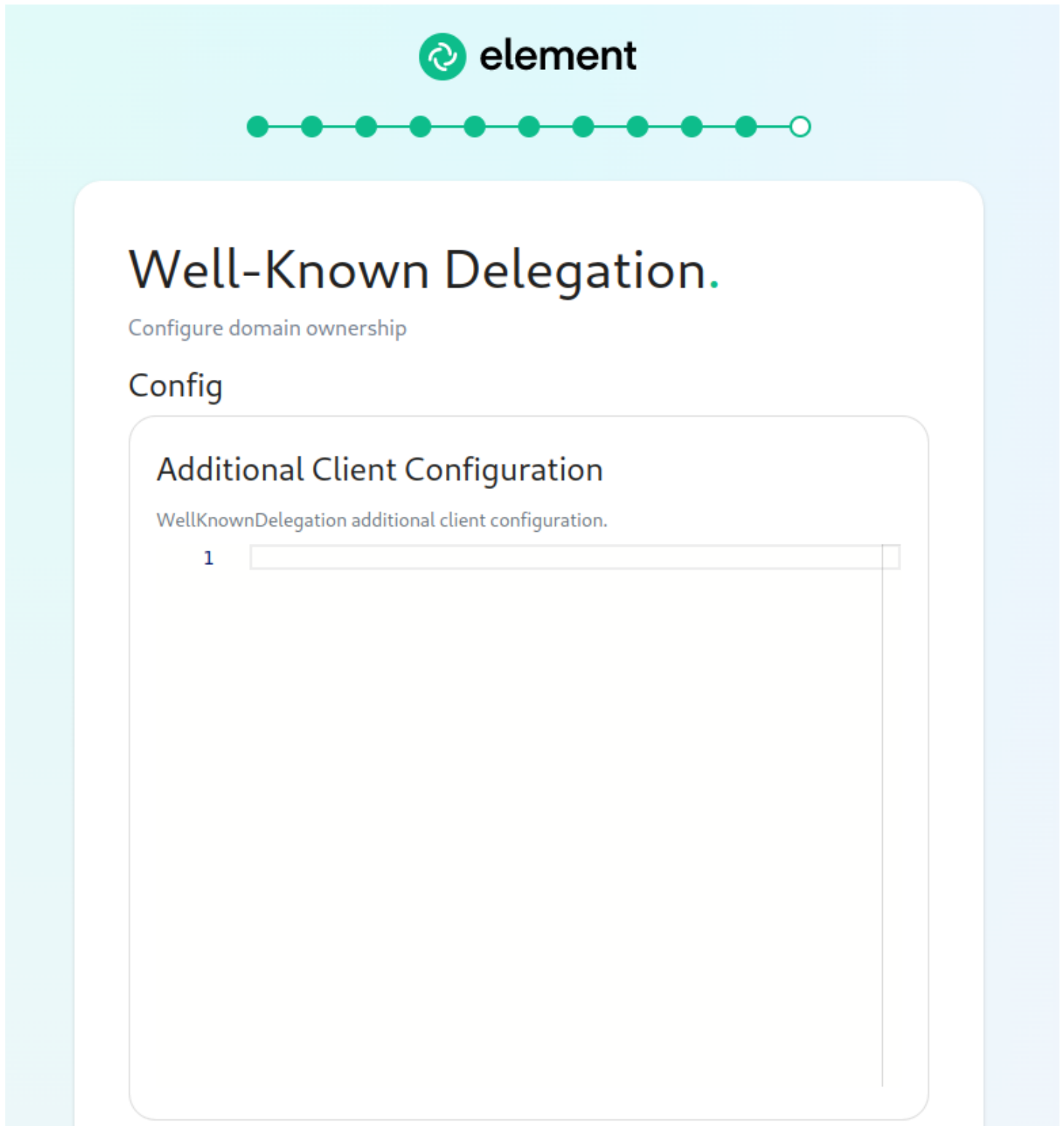
- Re-run the installer.

# Setting Up Well Known Delegation

## Well Known Delegation Configuration

From the Installer's Integrations page, click "Install" under "Well-Known Delegation".

Add any client configuration here:



The screenshot shows the Element installer's configuration page for Well-Known Delegation. At the top, the Element logo is displayed, followed by a progress indicator consisting of 11 green circles on a horizontal line, with the last circle being white. Below this, the page title "Well-Known Delegation." is shown in a large font, with the subtitle "Configure domain ownership" underneath. The "Config" section is active, and within it, the "Additional Client Configuration" section is visible. This section contains the text "WellKnownDelegation additional client configuration." and a single numbered input field (labeled "1") with a text area for configuration.

**element**

Well-Known Delegation.  
Configure domain ownership

Config

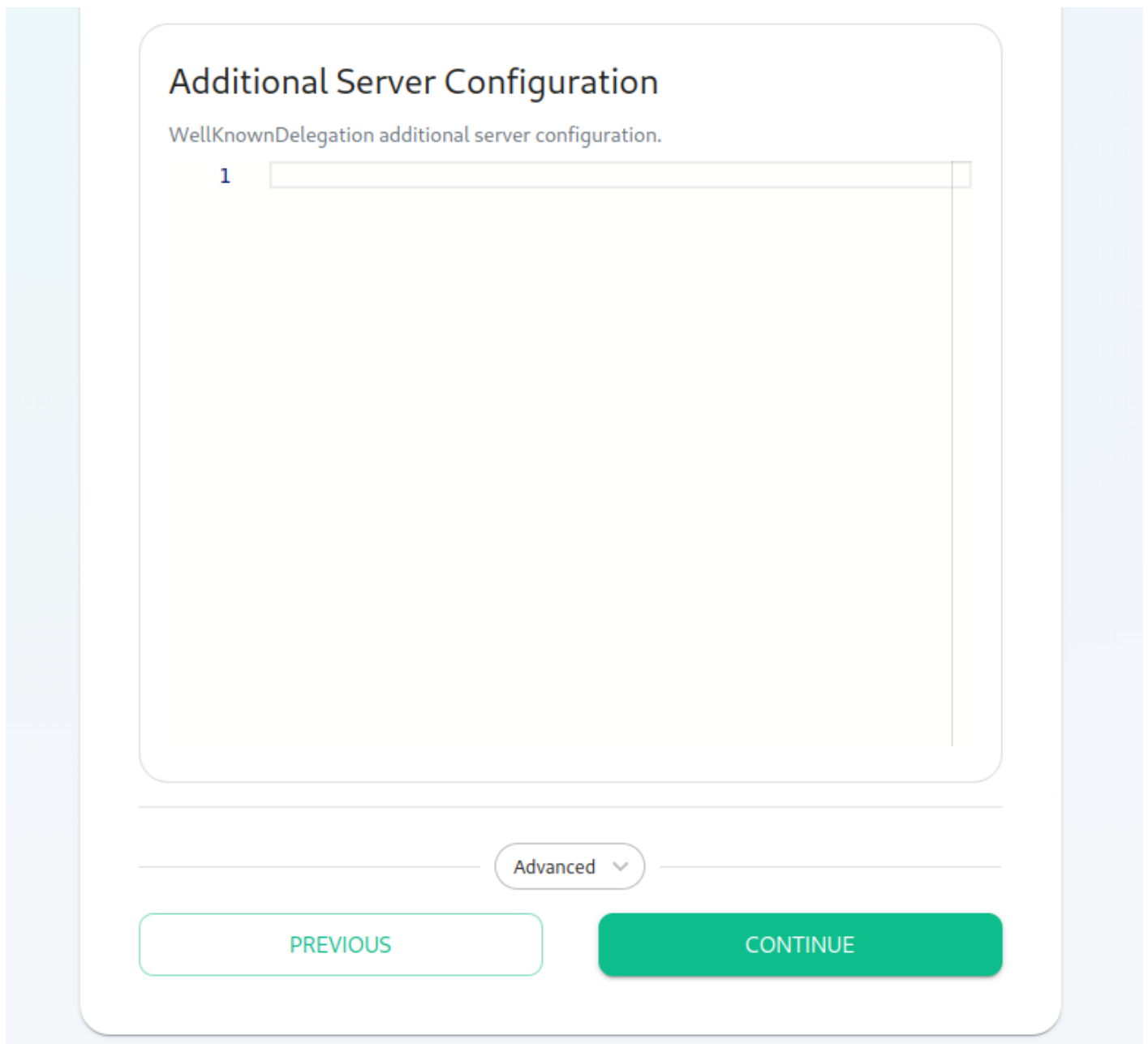
**Additional Client Configuration**  
WellKnownDelegation additional client configuration.

1

A sample client configuration might look like:

```
{
 "im.vector.riot.jitsi": {
 "preferredDomain": "jitsi.dev.local"
 }
}
```

Add any server configuration here:



**Additional Server Configuration**

WellKnownDelegation additional server configuration.

1

WellKnownDelegation

Advanced ▾

PREVIOUS CONTINUE

- Re-run the installer for the changes to take effect.

## Troubleshooting the Well Know config

The clients and servers will need to be able to access these configuration settings. You can check if everything is in place with curl. The following request is useful if your base domain is actually the same as your main webserver. This curl goes directly to the ingress of the kubernetes, which is implemented with nginx. Keeping the request header as "my.base.domain" allows nginx to route the request to the correct pod.

```
$ curl -X GET --header "HOST: my.base.domain" "https://matrix.my.base.domain/.well-known/matrix/client"
{
 "io.element.e2ee": {
 "default": false
 },
 "m.homeserver": {
 "base_url": "https://matrix.my.base.domain"
 }
}
```

The above shows a correctly setup well-known response, for the direct request to the cluster. In some setups there is a web server in front of your Element installation. In these cases the main web server should be implementing a reverse proxy for everything that is under `https://my.base.domain/.well-known/matrix/`. All these requests should be sent to `https://matrix.my.base.domain/.well-known/matrix/`. If the main web server would run Apache, the config would look like this :

```
ProxyPass /.well-known/matrix/ https://matrix.MYBASEDDOMAIN/.well-known/matrix/
ProxyPassReverse /.well-known/matrix/ https://matrix.MYBASEDDOMAIN/.well-known/matrix/
ProxyPreserveHost On
```

This is the check :

```
$ curl -X GET https://my.base.domain/.well-known/matrix/client
{
 "io.element.e2ee": {
 "default": false
 },
 "m.homeserver": {
 "base_url": "https://matrix.my.base.domain"
 }
}
```

You can check the ingress logs. Verify the request reaching the nginx and check for the correct path. Replace `$(XXXX)` with the actual name in your deployment ( `$ kubectl get pods -A` will reveal that name ).

```
$ kubectl logs nginx-ingress-microk8s-controller-{XXXX} -n ingress
```

```
...
```

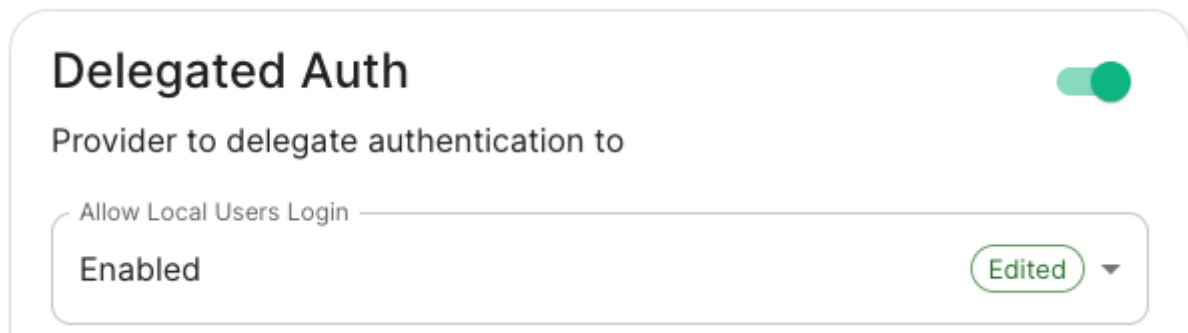
# Setting up Delegated Authentication With the Installer

## Delegated Authentication

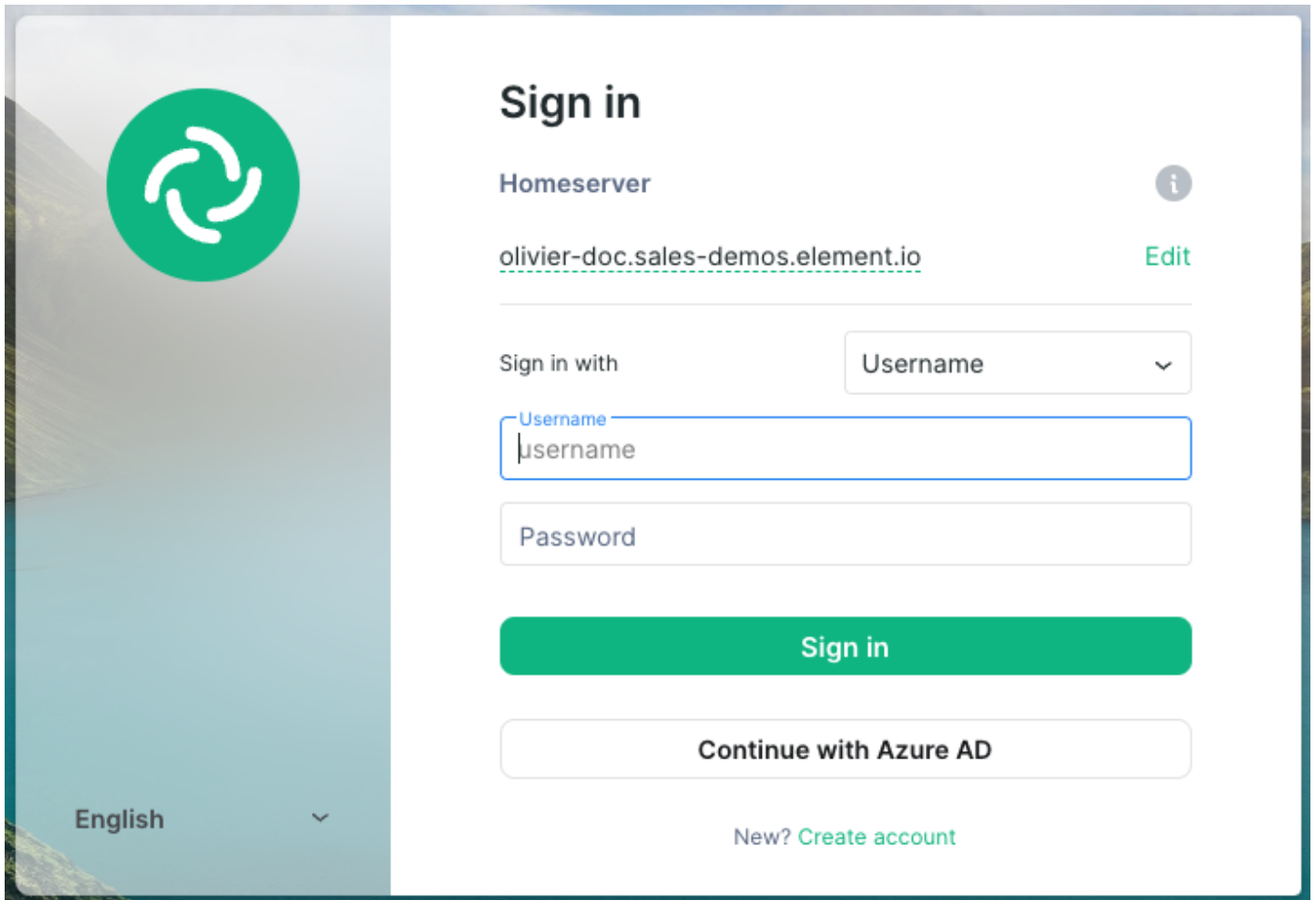
At present, we support delegating the authentication of users to the following provider interfaces:

- LDAP
- SAML
- OIDC
- CAS

When enabling Delegated Auth, you can still allow local users managed by Element to connect to the instance



When `Allow Local Users Login` is `Enabled`, you can both connect to your instance using your IDP and the local database.



Different options are offered by the installer and you can combine two or more options on the same instance like enabling SAML and OIDC delegated authentication.

Setting up Delegated Authentication with LDAP on Windows AD

Setting up Delegated Authentication with OpenID on Microsoft Azure

Setting up Delegated Authentication with OpenID on Microsoft AD FS

Note: We are rapidly working to expand and improve this documentation. For now, we are providing screenshots of working configurations, but in the future, we will better explain the options as well. If you do not see your provider listed below, please file a support ticket or reach out to your Element representative and we will work to get you connected and our documentation updated.



# Integrations and Add-Ons

# Setting Up Jitsi and TURN With the Installer

## Configure the Installer to install Jitsi and TURN

### Prerequisites

#### Firewall

You will have to open the following ports to your microk8s host to enable coturn and jitsi :

For jitsi :

- 30301/tcp
- 30300/udp

For coturn, allow the following ports :

- 3478/tcp
- 3478/udp
- 5349/tcp
- 5349/udp

You will also have to allow the following port range, depending on the settings you define in the installer (see below) :

- <coturn min port>-<coturn max port>/udp

#### DNS

The jitsi and coturn domain names must resolve to the VM access IP. You must not use `host_aliases` for these hosts to resolve to the private IP locally on your setup.

### Coturn

From the Installer's Integrations page, click "Install" under "Coturn".



# Coturn.

Cancel and return to Integrations

## Coturn.yml

Edited Default

```
1 shared_secret: # a shared secret, to generate randomly
2 coturn_fqdn: coturn.dev.local
3 min_port: 40000 # a port between 32769-65535
4 max_port: 40500 # max port, between 32769-65535
5 # external_ip: x.x.x.x # if coturn is behind NAT, you ne
6
```

PREVIOUS

CONTINUE

For the coturn.yml presented by the installer, edit the file and ensure the following values are set:

- `coturn_fqdn`: The access address to coturn. It should match something like `coturn.<fqdn.tld>`. **It must resolve to the public-facing IP of the VM.**
- `shared_secret`: A random value, you can generate it with `pwgen 32`
- `min_port`: The minimal UDP Port used by coturn for relaying UDP Packets, in range 32769-65535
- `max_port`: The maximum UDP Port used by coturn for relaying UDP Packets, in range 32769-65535

Further, for the `coturn_fqdn`, you will need to provide certificates for the installer outside of the GUI. Please find your `~/element-enterprise-server/config` directory and create a directory called `~/element-enterprise-server/config/legacy/certs` under which to put a `.crt/.key` PEM encoded certificate for this fqdn. If your fqdn was `coturn.airgap.local`, your filenames would need to be `coturn.airgap.local.crt` and `coturn.airgap.local.key`. You will need to have these certificate files in place before running the installer.

## Jitsi

From the Installer's Integrations page, click "Install" under "Jitsi".



# Jitsi.

Cancel and return to Integrations

## Jitsi.yml

Edited Default

```
1 jitsi_fqdn: jitsi.dev.local
2 jicofo_auth_password: # a secret internal password for jicofo auth
3 jicofo_component_secret: # a secret internal password for jicofo component
4 jvb_auth_password: # a secret internal password for jvb
5
6 app_name: "element-jitsi" # the hosted app name
7 helm_override_values: {} # if needed, to override helm values
8
9 timezone: Europe/Paris # The timezone in [TZ format](https://en.cppreference.com/w/cpp/string/basic.time_zone)
10
11 # stun_servers: needed if you don't setup coturn using the helm chart
12
```

PREVIOUS

CONTINUE

For the jitsi.yml presented by the installer, edit the file and ensure the following values are set:

- `jitsi_fqdn`: The access address to jitsi. It should match something like `jitsi.<fqdn.tld>`. **It must resolve to the public-facing IP of the VM.**
- `jicofo_auth_password`: # a secret internal password for jicofo auth
- `jicofo_component_secret`: # a secret internal password for jicofo component
- `jvb_auth_password`: # a secret internal password for jvb

- `helm_override_values`: {} # if needed, to override helm settings automatically set by the installer; For Helm values that can be overridden, see <https://vector-im.github.io/jitsi-helm/> For environment variables that can be passed in via Helm overrides, see <https://jitsi.github.io/handbook/docs/devops-guide/devops-guide-docker/>
- `timezone`: Europe/Paris # The timezone in TZ format
- `stun_servers`: Needed if you don't setup coturn using the installer. Should be a yaml list of server:port entries. Example:

```
stun_servers:
- ip: port
- ip: port
```

Further, for the `jitsi_fqdn`, you will need to provide .crt/.key PEM encoded certificates. These can be entered in the installer UI. If your fqdn was `jitsi.airgap.local`, your filenames would need to be `jitsi.airgap.local.crt` and `jitsi.airgap.local.key`. You will need to edit the file name field in the UI before pressing "Choose File" button when selecting the certificates.

If your network does not have any NAT, Jitsi cannot use the local coturn server to determine the IP it should advertise to the users. In this case, you might have issues with your calls and video. To workaround it, you can use the following configuration :

```
provide_node_address_as_public_ip: true

helm_override_values:
 jvb:
 extraEnvs:
 - name: JVB_ADVERTISE_IPS
 value: "public ip of jitsi"
 - name: JVB_ADVERTISE_PRIVATE_CANDIDATES
 value: "true"
```

## Element

## Config

### Additional configuration

Element web additional configuration.

1

### K8s

Force values for components of Element Web

SHOW

PREVIOUS

CONTINUE

Please go to the "Element Web" page of the installer, click on "Advanced" and add the following to "Additional Configuration":

```
{
 "jitsi": {
 "preferredDomain": "<jitsi_fqdn>"
 }
}
```

In the above text, you will want to replace `<jitsi_fqdn>` with the actual fqdn.

# Configure the installer to use an existing Jitsi instance

## Config

### Additional configuration

Element web additional configuration.

1

## K8s

Force values for components of Element Web

SHOW

PREVIOUS

CONTINUE

Please go to the "Element Web" page of the installer, click on "Advanced" and add the following to "Additional Configuration":

```
{
 "jitsi": {
```



```
 "preferredDomain": "your.jitsi.example.org"
 }
}
```

replacing `your.jitsi.example.org` with the hostname of your Jitsi server.

You will need to re-run the installer for this change to take effect.

## Configure the installer to use an existing Coturn instance

Follow the instructions here: <https://ems-docs.element.io/books/element-on-premise-documentation/page/single-node-installations#bkmrk-turn-server>

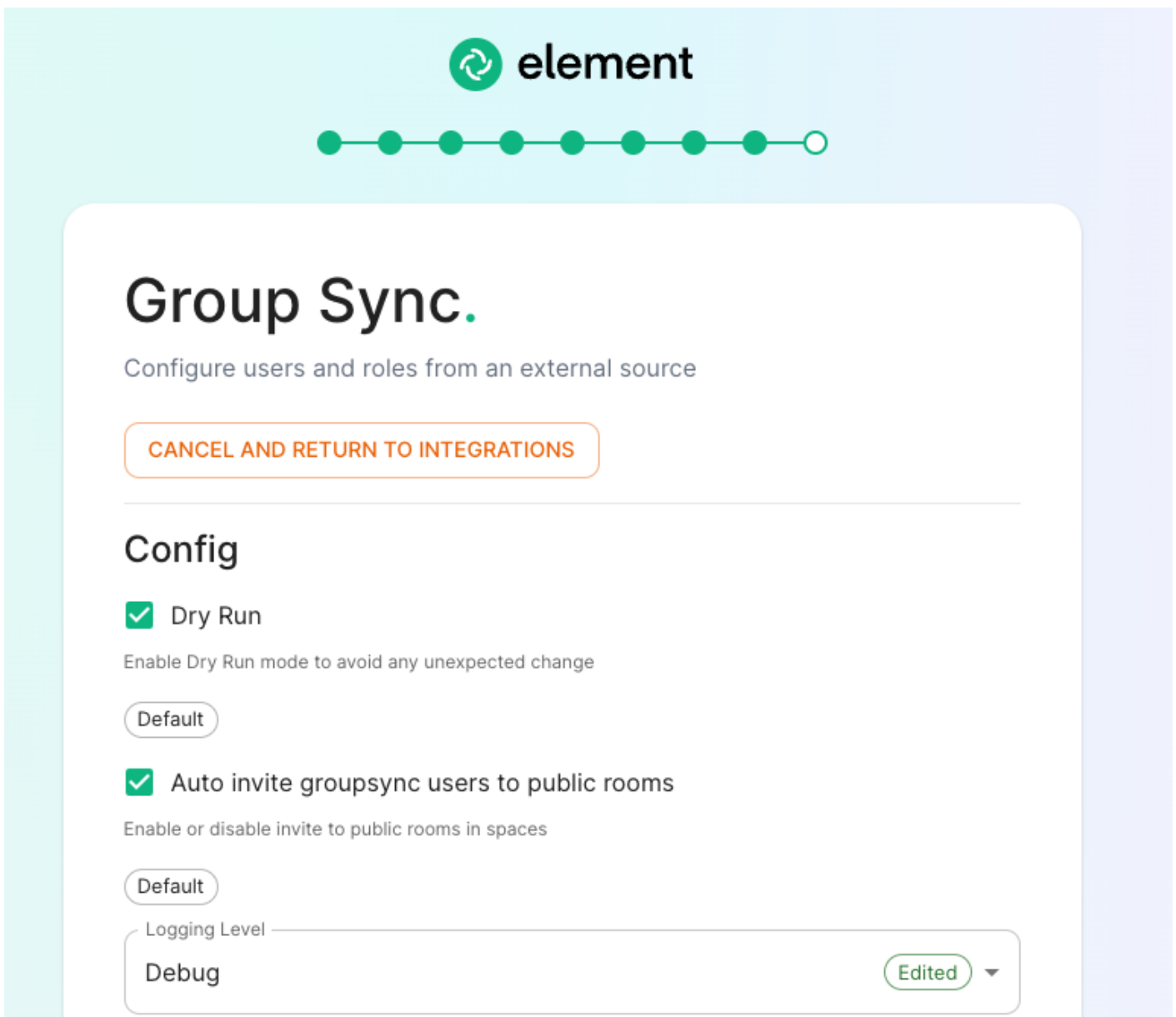
# Setting up Group Sync with the Installer

## What is Group Sync?

Group Sync allows you to use the ACLs from your identity infrastructure in order to set up permissions on Spaces and Rooms in the Element Ecosystem. Please note that the initial version we are providing only supports a single node, non-federated configuration.

## Configuring Group Sync

From the Installer's Integrations page, click "Install" under "Group Sync".



The screenshot shows the Element Group Sync configuration page. At the top, the Element logo is displayed with a progress indicator consisting of eight green circles, the last of which is white. Below the logo, the title "Group Sync." is prominently displayed, followed by the subtitle "Configure users and roles from an external source". A button labeled "CANCEL AND RETURN TO INTEGRATIONS" is positioned below the subtitle. The "Config" section contains two checked options: "Dry Run" (with a sub-note "Enable Dry Run mode to avoid any unexpected change") and "Auto invite groupsync users to public rooms" (with a sub-note "Enable or disable invite to public rooms in spaces"). Each option has a "Default" button. At the bottom, there is a "Logging Level" dropdown menu currently set to "Debug" and an "Edited" button with a downward arrow.

- Leaving `Dry Run` checked in combination with `Logging Level` set to `Debug` gives you the ability to visualize in the pod's log file what result group sync will produce without effectively creating spaces and potentially corrupting your database. Otherwise, uncheck `Dry Run` to create spaces according to your spaces mappings defined in the `Space mapping` section.
- `Auto invite groupsync users to public room` determines whether users will be automatically invited to rooms (default, public and space-joinable). Users will still get invited to spaces regardless of this setting.

# Configuring the source

## LDAP Servers

- You should create a LDAP account with read access.
- This account should use password authentication.

## Select source

LDAP  Azure (MSGraph)  SCIM

### LDAP

Mapping attribute for room name

name

The LDAP attribute to request space names

Mapping attribute for username

sAMAccountName

The LDAP attribute to request user id

LDAP Base DN

OU=Demo corp,DC=olivier,DC=sales-demos,DC=element,DC=io

The LDAP base DN

LDAP Bind DN

CN=gsync,CN=Users,DC=olivier,DC=sales-demos,DC=element,DC=i

The LDAP bind DN

Check interval in seconds

60

Default

the ldap check in seconds

LDAP Filter

An additional ldap filter

LDAP URI

ldap://3.75.187.130


The LDAP URI groupsync will use to request users

Secrets

/

Group Sync

/

LDAP Bind Password 

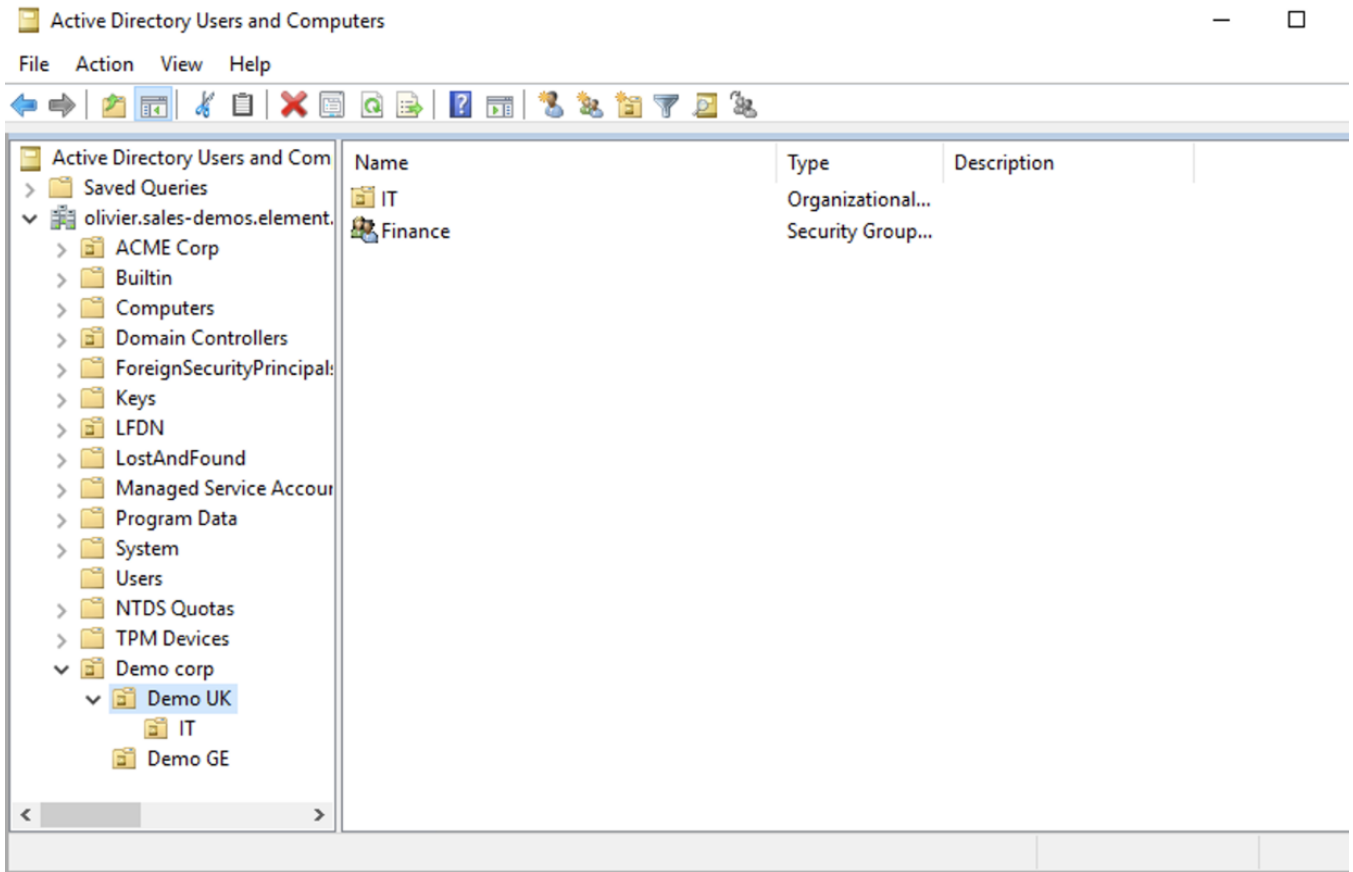
LDAP Password

.....



LDAP Bind password

- **LDAP Base DN** : the distinguished name of the root level Org Unit in your LDAP directory. In our example, **Demo Corp** is our root level, spaces are mapped against Org Units , but you can map a space against any object (groups, security groups,...) belonging to this root level.



The distinguished name can be displayed by selecting **View / Advanced Features** in the Active Directory console and then, right-clicking on the object, selecting **Properties / Attributes Editor**.

The DN is **OU=Demo corp, DC=olivier, DC=sales- demos, DC=element, DC=io**.

- **Mapping attribute for room name** : LDAP attribute used to give an internal ID to the space (visible when setting the log in debug mode)
- **Mapping attribute for username** : LDAP attribute like **sAMAccountName** used to map the localpart of the mxid against the value of this attribute.  
If **@bob: my- domain. org** is the mxid, **bob** is the localpart and groupsync expects to match this value in the LDAP attribute **sAMAccountName**.
- **LDAP Bind DN** : the distinguished name of the LDAP account with read access.
- **Check interval in seconds** : the frequency Group sync refreshes the space mapping in Element.
- **LDAP Filter** : an LDAP filter to filter out objects under the LDAP Base DN.
- **LDAP URI** : the URI of your LDAP server.
- **LDAP Bind Password** : the password of the LDAP account with read access.

## MS Graph (Azure AD)

- You need to create an **App registration**. You'll need the **Tenant ID** of the organization, the **Application (client ID)** and a secret generated from **Certificates & secrets** on the app.
- For the bridge to be able to operate correctly, navigate to **API permissions** and ensure it has access to **Group.Read.All**, **GroupMember.Read.All** and **User.Read.All**. Ensure that these are **Application permissions** (rather than **Delegated**).

- Remember to grant the admin consent for those.
- To use MSGraph source, select MSGraph as your source.
  - `msgraph_tenant_id` : This is the "Tenant ID" from your Azure Active Directory Overview
  - `msgraph_client_id` : Register your app in "App registrations". This will be its "Application (client) ID"
  - `msgraph_client_secret` : Go to "Certificates & secrets", and click on "New client secret". This will be the "Value" of the created secret (not the "Secret ID").

# Space Mapping

The space mapping mechanism allows us to configure spaces that Group Sync will maintain, beyond the ones that you can create manually.

It is optional – the configuration can be skipped but if you enable Group Sync, you have to edit the Space mapping by clicking on the `EDIT` button and rename the `(unnamed space)` to something meaningful.

### Space mapping

Toplevel

Add new space

Name

This is a toplevel space

---

Groups  Include all users in the directory in this space

Assign a group

---

Delete this space

Deleting a space is irreversible. Creating a new space with the same name is not equivalent to editing an existing one. Take care when removing spaces that have already been provisioned on your server.

DONE

`Include all users in the directory in this space` : all available users, regardless of group memberships join the space. This option is convenient when creating a common subspace shared between all users.

**Space mapping**

Cloud team

Business Systems team

Cloud Engineering Support

Customer Engineers

[Add new space](#)

Name

This is a subspace of **Cloud team**

Groups  Include all users in the directory in this space

External ID	Matrix Power Level
<input type="text" value="OU=Business Systems,OU=GC"/>	<input type="text" value="0"/>
<input type="text" value="CN=moderators,OU=Business"/>	<input type="text" value="50"/>

[Assign a group](#)

[Delete this space](#)

Deleting a space is irreversible. Creating a new space with the same name is not equivalent to editing an

[DONE](#)

When clicking on [Add new space](#), you can leave the space as a top level space or you can drag and drop this space onto an existing space, making this space a subspace of the existing space.

You can then map an external ID (the LDAP distinguished name) against a power level. Every user belonging to this external ID is granted the power level set in the interface. This external ID that can be any LDAP object like an OrgUnit, a Group or a Security Group

A power level 0 is a default user that can write messages, react to messages and delete his own messages.

A power level 50 is a moderator that can creates rooms, delete messages from members.


A power level 100 is an administrator but since GroupSync manages spaces, invitations to the rooms, it does not make sense to map a group against a power level 100.


Custom power levels other than 0 and 50 are not supported yet.

## Users allowed un every GrupSync room

## Users allowed in every GroupSync room

Optionally configures a list of users to allow in any groupsync-managed room



 @adminbot.\*

[ADD MORE USERS ALLOWED IN EVERY GROUPOSYNC ROOM](#)

A list of userid patterns that will not get kicked from rooms even if they don't belong to them according to LDAP.

This is useful for things like auditbot if Audibot has been enabled.

Patterns listed here will be wrapped in ^ and \$ before matching.

## Defaults Rooms



# Default Rooms

A list of rooms to configure by default in all spaces

## Room Properties

A room to configure by default in all spaces - The room properties



Name

General topics

Edited

The room name

[ADD MORE DEFAULT ROOMS](#)

A list of rooms added to every space

Integrations and Add-Ons

# Setting up GitLab, GitHub, JIRA and Webhooks Integrations With the Installer

In Element Server Suite, our GitLab, GitHub, and JIRA extensions are provided by the hookshot package. This documentation explains how to configure hookshot.

## Configuring Hookshot with the Installer

From the Installer's Integrations page, click "Install" under "Hookshot: Github, Gitlab, Jira, and Custom Webhooks."



# Hookshot: Github, Gitlab, Jira, and Custom Webhooks.

Forward messages from external sources into rooms as they occur

[Cancel and return to Integrations](#)

Logging Level  Default ▾

Verify TLS  Default ▾

TLS Verification

Secrets / Hookshot / Hookshot Pass Key ▾

**Hookshot Password key**  No file chosen  
**Edited** You can generate it using : `openssl genpkey -out passkey.pem -outform PEM -algorithm RSA -pkeyopt rsa_keygen`

Secrets / Hookshot / Provisioning Secret ▾

Hookshot provisioning secret  Edited

This password should be generated randomly.

On the first screen here, we can set the logging level and a hookshot specific verify tls setting. Most users can leave these alone.

To use hookshot, you will need to generate a hookshot password key, when can be done by running the following command on a Linux command line:

```
openssl genpkey -out passkey.pem -outform PEM -algorithm RSA -pkeyopt rsa_keygen_bits:4096
```

which will generate output similar to this:

```
.....
.....++++
.....++++
```

Once this has finished, you will have a file called passkey.pem that can use to upload as the "Hookshot Password key".

If you wish to change the hookshot provisioning secret, you can, but you can also leave this alone as it is randomly generated by the installer.

## Bot

### The hookshot bot

Avatar

mxc://

Default

The hookshot bot avatar mxc url

Display Name

Hookshot Bot

Default

The hookshot bot display name

Username

hookshot

Default

The hookshot bot username

## Widgets

### The hookshot widgets settings

SHOW

## GitLab

Gitlab hooks



## Jira



## Webhooks

Configuration of hookshot generic webhooks



Next, we get to a set of settings that allow us to make changes to the Hookshot bot's appearance.

There is also a button to show widget settings, which brings up these options:

# Widgets

The hookshot widgets settings

HIDE

## Add on Invite

Add widgets on invite

Default

## Add to Admin Rooms

Add widgets to admin rooms

Default

Title

Hookshot Configuration

Default

The hookshot widget title

## Disallowed IP Ranges

Which IP ranges should be disallowed when resolving homeserver IP addresses (for security reasons). Unless you know what you are doing, it is recommended to not change this.

<input type="checkbox"/>	<input type="text" value="An IP range, ipv4 or ipv6 format"/>	<input type="text" value="192.168.122.0/24"/>	<input type="text" value="Default"/>
<input type="checkbox"/>	<input type="text" value="An IP range, ipv4 or ipv6 format"/>	<input type="text" value="127.0.0.0/8"/>	<input type="text" value="Default"/>
<input type="checkbox"/>	<input type="text" value="An IP range, ipv4 or ipv6 format"/>	<input type="text" value="10.0.0.0/8"/>	<input type="text" value="Default"/>
<input type="checkbox"/>	<input type="text" value="An IP range, ipv4 or ipv6 format"/>	<input type="text" value="172.16.0.0/12"/>	<input type="text" value="Default"/>

In this form, we have the ability to control how widgets are incorporated into rooms (the defaults are usually fine) and to set a list of Disallowed IP ranges wherein widgets will not load if the homeserver IP falls in the range. If your homeservers IP falls in any of these ranges, you will want to remove that range so that the widgets will load!

Next, we have the option to enable Gitlab, which shows us the following settings:

# GitLab

Gitlab hooks

Secrets / Hookshot / GitLab Webhook Secret

Gitlab webhook secret

..... Edited

This password should be generated and shared with gitlab on the webhook page.

## Gitlab instances Rooms

Name \*

Gitlab instance name

URL \*

Gitlab instance URL

ADD MORE GITLAB INSTANCES ROOMS

The webhook secret is randomly generated and does not need to be changed. You can also add Gitlab instances by specifying an instance name and pasting the URL.

Next, we have the option to enable Jira, which shows us the following settings:

The screenshot shows a configuration page for Jira. At the top, the word "Jira" is displayed with a green toggle switch to its right. Below this, there are two main sections. The first section is for the OAuth Client ID, with a text input field containing "OAuth Client ID \*". Below the field is the label "Jira OAuth client id". The second section is for the Jira OAuth client secret, with a breadcrumb trail "Secrets / Hookshot / Jira OAuth Client Secret" and a dropdown arrow. Below this is a text input field containing "Jira OAuth client secret" and an eye icon. Below the field is the instruction "This secret should be copied from the OAuth secret in Jira settings." The third section is for the Jira Webhook Secret, with a breadcrumb trail "Secrets / Hookshot / Jira Webhook Secret" and a dropdown arrow. Below this is a text input field containing "Jira webhook secret" and a series of dots representing a password. To the right of the field is an eye icon and a green "Edited" button. Below the field is the instruction "This password should be generated and shared with Jira on the webhook page."

In here, we can specify the OAuth Client ID and the OAuth client secret to connect to Jira. To obtain this information, please follow these steps:

The JIRA service currently only supports atlassian.com (JIRA SaaS) when handling user authentication. Support for on-prem deployments is hoping to land soon.

- You'll first need to head to <https://developer.atlassian.com/console/myapps/create-3lo-app/> to create a "OAuth 2.0 (3LO)" integration.
- Once named and created, you will need to:
- Enable the User REST, JIRA Platform REST and User Identity APIs under Permissions.
- Use rotating tokens under Authorisation.
- Set a callback url. This will be the public URL to hookshot with a path of `/jira/oauth`.
- Copy the client ID and Secret from Settings

Once you've set these, you'll notice that a webhook secret has been randomly generated for you. You can leave this alone or edit it if you desire.

Next, let's look at configuring Webhooks:



## Webhooks



### Configuration of hookshot generic webhooks

Allow JS Transformation Functions

To allow JS Transformations functions

Default

Enabled

Enable or disable generic webhooks

Default

User ID Prefix

webhooks user id prefixes

You can set whether or not webhooks are enabled and whether they allow JS Transformation functions. It is good to leave these enabled per the defaults. You can also specify the user id prefix for the creation of custom webhooks. If you set this to `webhook_` then each new webhook will appear in a room with a username starting with `webhook_`.

Next, let's look at configuring Github:

# GitHub



## Configuration of hookshot github integration

Auth ID \*

GitHub application auth id

OAuth Client ID \*

GitHub OAuth client id

Secrets

/

Hookshot

/

GitHub Key File



**GitHub  
application key  
file**

Choose File

No file chosen

Edited

It can be generated by clicking "Generate a private key" under the Private keys section on the GitHub app page

Secrets

/

Hookshot

/

GitHub OAuth Client Secret



GitHub OAuth client secret



The OAuth Client secret of the github app page.

Secrets

/

Hookshot

/

GitHub Webhook Secret



GitHub webhook secret



It is the Webhook secret on the GitHub App page.

This bridge requires a GitHub App. You will need to create one. Once you have created this, you'll be able to fill in the Auth ID and OAuth Client ID. You will also need to generate a "GitHub application key file" to upload this. Further, you will need to specify a "GitHub OAuth client secret" and a "GitHub webhook secret", both of which will appear on your newly created GitHub app page.

## Default Options

The default options to apply to github hooks

Command Prefix

Choose the prefix to use when sending commands to the bot. Ideally starts with "!" lgh

Hotlink command prefix

Send a link to an issue/PR in the room when a user mentions a prefix followed by a number

Share issue room link on new issues

When new issues are created, provide a Matrix alias link to the issue room

Default

### Enable Hooks

Enable notifications for some event types

ADD ENABLE HOOKS

### Excluding Labels

Never notify on issues matching these label names

ADD EXCLUDING LABELS

### Ignore Hooks

Choose to exclude notifications for some event types

ADD IGNORE HOOKS

On this screen, we have the option to change how we call the bot and other minor settings. We also have the ability to select which hooks we provide notifications for, what labels we wish to exclude, and then which hooks we will ignore completely.

## Including Labels

Only notify on issues matching these label name

ADD INCLUDING LABELS

## New issues options

Configuration options for new issues

### Labels

Automatically set these labels on issues created via commands

ADD LABELS

## PR Diff

Show a diff in the room when a PR is created, subject to limits

Enabled

Enable the PR diff

Default

Max Lines

0

Default

Max number of lines to display in the room

Now we have the ability to add a list of labels that we want to match. This has the impact of the integration only notifying you of issues with a specific set of labels.

We then have the ability to add a list of labels that all newly created issues through the bot should be labeled with.

Then we have the ability to enable showing diffs in the room when a PR is created.

## Workflow Run

Configuration options for workflow run results

### Matching Branch

Only report workflow runs if it matches this regex.

### Excluding Workflows

Never report workflow runs with a matching workflow name.

[ADD EXCLUDING WORKFLOWS](#)

### Including Workflows

Only report workflow runs with a matching workflow name.

[ADD INCLUDING WORKFLOWS](#)

Moving along, we can configure how workflow run results are configured in the bot, including matching specific workflows and including or excluding specific workflows.

## Finishing Configuration

You further have the ability to click "Advanced" and set any kubernetes specific settings for how this pod is run. Once you have set everything up on this page, you can click "Continue" to go back to the Integrations page.

When you have finished running the installer and the hookshot pod is up and running, there are some configurations to handle in the Element client itself in the rooms that you wish the integration to be present.

As an admin, you will need to invite the hookshot bot into a room. The name can be found in the installer configuration under the username field in the "Bot" section.

Once you have invited the bot into the room, you can use the "Add widgets, bridges, & bots" functionality to add the "Hookshot Configuration" widget to the room and finish the setup.

# Setting up Adminbot and Auditbot

## Overview

Adminbot allows for an Element Administrator to become admin in any existing room or space on a managed homeserver. This enables you to delete rooms for which the room administrator has left your company and other useful administration actions.

Auditbot allows you to have the ability to export any communications in any room that the auditbot is a member of, even if encryption is in use. This is important in enabling you to handle compliance requirements that require chat histories be obtainable.

## On using Admin Bot and Audit Bot

Currently, we deploy a special version of Element Web to allow you to log in as the adminbot and auditbot. Given this, please do not make changes to widgets in rooms while logged in as the adminbot or the auditbot. The special Element Web does not have any custom settings that you have applied to the main Element Web that your users use and as such, you can cause problems for yourself by working with widgets as the adminbot and auditbot. In the future, we are working to provide custom interfaces for these bots.

## Configuring Admin Bot

From the Installer's Integrations page, click "Install" under "Admin Bot"

For the adminbot.yml presented by the installer, edit the file and ensure the following values are set:

```
bot_backup_phrase: adminsecret
bot_data_path: /mnt/data/adminbot
bot_data_size: 10M

enable_dm_admin: false

join_local_rooms_only: true
access_elementweb_fqdn: adminbot.airgap.local
```

Let's discuss them:

- **bot\_backup\_phrase:** This is the security phrase that will guard access to your encryption keys. Do NOT share this phrase with anyone. This is required.
- **bot\_data\_path:** This is the directory where the bot's data will be stored. If you need to change the path, please do, but for most cases, you can leave this alone. **If you are deploying to Kubernetes,**

### you need to comment this out!

- **bot\_data\_size**: In most cases, you can leave this at 10M, but it does put a limit on the amount of data that can be written by the bot to the path.
- **enable\_dm\_admin**: This defaults to `false` and that behavior means that adminbot **will not** join DMs. If you want full control of DMs, simply set this to `true`.
- **join\_local\_rooms\_only**: This defaults to `true` and that behavior means that adminbot will only join rooms on your local homeserver.
- **access\_elementweb\_fqdn**: You should set this to a hostname that is resolvable in your environment which will host a special instance of Element Web for logging in. This hostname will need a crt/key PEM encoded key pair and these files will need to be stored in `~/.element-enterprise-server/config/legacy/certs` prior to running the installer. In the above example, we have the hostname of `adminbot.airgap.local`. This means that the installer expects to find `adminbot.airgap.local.crt` and `adminbot.airgap.local.key` in the `~/.element-enterprise-server/config/legacy/certs` directory. If you are using Let's Encrypt, you do not need to add these files.
- **verify\_tls** : Optional. If doing a POC with self-signed certificates, set this to 0. Defaults to 1.

## Configuring Audit Bot

From the Installer's Integrations page, click "Install" under "Audit Bot"

For the `auditbot.yml` presented by the installer, edit the file and ensure the following values are set:

```
bot_backup_phrase: auditsecret
bot_data_path: /mnt/data/auditbot
bot_data_size: 10M

join_local_rooms_only: true
enable_dm_audit: false
access_elementweb_fqdn: auditbot.airgap.local

optional : the S3 bucket where to store the audit logs
#s3_bucket:
#s3_access_key_id:
#s3_secret_access_key:
#s3_key_prefix:
#s3_region:
#s3_endpoint:

optional : the local logfile settings. Used if s3 bucket is not enabled.
logfile_size: 1M
logfile_keep: 3
```

Let's discuss them:

- **bot\_backup\_phrase**: This is the security phrase that will guard access to your encryption keys. Do NOT share this phrase with anyone. This is required.
- **bot\_data\_path**: This is the directory where the bot's data will be stored. If you need to change the path, please do, but for most cases, you can leave this alone. **If you are deploying to Kubernetes, you need to comment this out!**
- **bot\_data\_size**: In most cases, you can leave this at 10M, but it does put a limit on the amount of data that can be written by the bot to the path.
- **join\_local\_rooms\_only**: This defaults to `true` and that behavior means that adminbot will only join rooms on your local homeserver.
- **enable\_dm\_admin**: This defaults to `false` and that behavior means that adminbot **will not** join DMs. If you want full control of DMs, simply set this to `true`.
- **access\_elementweb\_fqdn**: You should set this to a hostname that is resolvable in your environment which will host a special instance of Element Web for logging in. This hostname will need a crt/key PEM encoded key pair and these files will need to be stored in `~/.element-enterprise-server/config/legacy/certs` prior to running the installer. In the above example, we have the hostname of `auditbot.airgap.local`. This means that the installer expects to find `auditbot.airgap.local.crt` and `auditbot.airgap.local.key` in the `~/.element-enterprise-server/config/legacy/certs` directory. If you are using Let's Encrypt, you do not need to add these files.
- **verify\_tls** : Optional. If doing a POC with self-signed certificates, set this to 0. Defaults to 1.

# Adminbot Federation

## On the central admin bot server

Complete the values for the provided central.yml in the installer interface. Here is an explanation of the parameters:

- `adminbot_fqdn` : The FQDN which will be targeted by remote federated servers as the central audit server
- `remote_federated_homeservers` : A list containing every remote audited server. It contains the following variables :
  - `matrix_server` : URL of the synapse server
  - `domain_name` : Domain name from parameters.yaml (the server name part of the users mxid)
  - If the server is managed by the installer :
    - `generic_shared_secret` : The generic shared secret to get from secrets.yaml
    - `adminuser_token` : The token from the admin user, to get via `kubectl get synapseusers/adminuser-donotdelete -n element-onprem -o yaml`. It's the value of the field `status.accessToken`.
  - If the server is not managed by the installer :
    - `as_token` : The as token configured on the remote appservice file on the remote server.
    - `hs_token` : The as token configured on the remote appservice file on the remote server.
    - `adminuser_token` : An access token to an user which is server admin.

## On the remote admin bot server

Complete the access.yml file in the installer interface by providing the fqdn of the central admin bot server.

- `central_adminbot_fqdn` : The value of `adminbot_fqdn` on the central audit bot server



# Auditbot Federation

## On the central auditbot server

Complete the values for the provided central.yml in the installer interface. Here is an explanation of the parameters:

- `auditbot_fqdn` : The FQDN which will be targeted by remote federated servers as the central audit server
- `remote_federated_homeservers` : A list containing every remote audited server. It contains the following variables :
  - `matrix_server` : URL of the synapse server
  - `domain_name` : Domain name from parameters.yaml (the server name part of the users mxid)
  - If the server is managed by the installer :
    - `generic_shared_secret` : The generic shared secret to get from secrets.yaml
    - `adminuser_token` : The token from the admin user, to get via `kubectl get synapseusers/adminuser-donotdelete -n element-onprem -o yaml`. It's the value of the field `status.accessToken`.
  - If the server is not managed by the installer :
    - `as_token` : The as token configured on the remote appservice file on the remote server.
    - `hs_token` : The as token configured on the remote appservice file on the remote server.
    - `adminuser_token` : An access token to an user which is server admin.

## On the remote audit bot server

Complete the access.yml file in the installer interface by providing the fqdn of the central audit bot server.

- `central_auditbot_fqdn` : The value of `auditbot_fqdn` on the central audit bot server

# Setting Up Hydrogen

## Configuring Hydrogen

From the Installer's Integrations page, click "Install" under "Hydrogen".

For the hydrogen.yml presented by the installer, edit the file and ensure the following values are set:

- `hydrogen_fqdn` is the FQDN that will be used for accessing hydrogen. It must have a PEM formatted SSL certificate as mentioned in the introduction. The crt/key pair must be in the `CONFIG_DIRECTORY/certs` directory.
- `extra_config` is extra json config that should be injected into the hydrogen client configuration.

You will need to re-run the installer after making these changes for them to take effect.

# Setting up On-Premise Metrics

## Setting up prometheus and grafana

From the Installer's Integrations page, click "Install" under "Prometheus"

For the provided prom.yml, see the following descriptions of the parameters:

- If you want to write prometheus data to a remote prometheus instance, please define these 4 variables :
  - `remote_write_url` : The URL of the endpoint to which to push remote writes
  - `remote_write_external_labels` : The labels to add to your data, to identify the writes from this cluster
  - `remote_write_username` : The username to use to push the writes
  - `remote_write_password` : The password to use to push the writes
- You can configure which prometheus components you want to deploy :
- `deploy_prometheus` : `true` to deploy prometheus
- `deploy_node_exporter` : requires prometheus deployment. Set to `true` to gather data about the k8s nodes.
- `deploy_kube_control_plane_monitoring` : requires prometheus deployment. Set to `true` to gather data about the kube control plane.
- `deploy_kube_state_metrics` : requires prometheus deployment. Set to `true` to gather data about kube metrics.
- `deploy_element_service_monitors` : Set to `true` to create `ServiceMonitor` resources into the K8S cluster. Set it to `true` if you want to monitor your element services stack using prometheus.
- You can choose to deploy grafana on the cluster :
  - `deploy_grafana` : `true`
  - `grafana_fqdn` : The FQDN of the grafana application
  - `grafana_data_path` : `/mnt/data/grafana`
  - `grafana_data_size` : 1G
- If doing a POC with self-signed certificates:
  - `verify_tls` : Optional. If doing a POC with self-signed certificates, set this to 0. Defaults to 1.

For the specified `grafana_fqdn`, you will need to provide a crt/key PEM encoded key pair in `~/element-enterprise-server/config/legacy/certs` prior to running the installer. If our hostname were `metrics.airgap.local`, the installer will expect to find `metrics.airgap.local.crt` and `metrics.airgap.local.key` in the `~/element-enterprise-server/config/legacy/certs` directory. If you are using Let's Encrypt, you do not need to add these files.

After running the installer, open the FQDN of Grafana. The initial login user is `admin` and password is `admin`. You'll be required to set a new password, please define one secured and keep it in a safe place. ~

# Setting Up the Telegram Bridge

## Configuring Telegram bridge

### On Telegram platform

- Login to [my.telegram.org](https://my.telegram.org) to get a telegram app ID and hash (get from [here](#)). You should use a phone number associated to your company.

## Basic config

From the Installer's Integrations page, click "Install" under "Telegram Bridge".

For the provided telegram.yml file, please see the following options:

- `postgres_create_in_cluster`: `true` to create the postgres db into the k8s cluster. On a standalone deployment, it is necessary to define the `postgres_data_path`.
- `postgres_fqdn`: The fqdn of the postgres server. If using `postgres_create_in_cluster`, you can choose the name of the workload.
- `postgres_data_path`: `"/mnt/data/telegram-postgres"`
- `postgres_port`: `5432`
- `postgres_user`: The user to connect to the db.
- `postgres_db`: The name of the db.
- `postgres_password`: A password to connect to the db.
- `telegram_fqdn`: The FQDN of the bridge for communicating with Telegram and using public login user interface.
- `max_users`: Max number of users enabled on the bridge.
- `bot_username`: The username of the bot for users to manage their bridge connectivity.
- `bot_display_name`: The display name of the bot.
- `bot_avatar`: An mx content URL to the bot avatar.
- `admins`: The list of admins of the bridge.
- `enable_encryption`: `true` to allow e2e encryption in bridge.
- `enable_encryption_by_default`: `true` to enable by default e2e encryption on every chat created by the bridge.
- `enable_public_portal`: `true` to give the possibility to users to login using the bridge portal UI.
- `telegram_api_id`: The telegram API ID you got from telegram platform.
- `telegram_api_hash`: The telegram api hash you got from telegram platform.

For the specified `telegram_fqdn`, you will need to provide a crt/key PEM encoded key pair in `~/element-enterprise-server/config/legacy/certs` prior to running the installer. If our hostname were `telegram.airgap.local`, the installer will expect to find `telegram.airgap.local.crt` and `telegram.airgap.local.key` in the `~/element-enterprise-server/config/legacy/certs` directory. If you are using Let's Encrypt, you do not need to add these files.`

You will need to re-run the installer after making changes for these to take effect.

# Usage

- Talk to the telegram bot to login to the bridge. See Telegram Bridge starting at "Bridge Telegram to your Element account". Instead of addressing the bot as that document explains, use "@bot\_username:domain" as per your setup.

# Setting Up the Teams Bridge

## Configuring Teams Bridge

### Register with Microsoft Azure

You will first need to generate an "Application" to serve connect your Teams bridge with Microsoft.

- Connect to Azure on [https://portal.azure.com/#blade/Microsoft\\_AAD\\_IAM/ActiveDirectoryMenuBlade/Overview](https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/Overview) to go to the Active Directory.
- Go to "Register an application screen" and register an application.
- Supported account types can be what fits your needs, but do not select "Personal Microsoft accounts"
- **Redirect URI** must be `https://<teams_fqdn>/authenticate`. You must use the type `Desktop and Mobile apps`. You don't need to check any of suggested redirection URIs.
- You should be taken to a general configuration page. Click Certificates & secrets
- Generate a **Client Secret** and copy the resulting value. The value will be your `teams_client_secret`.

## Permissions

You will need to set some API permissions.

For each of the list below click Add permission > Microsoft Graph > and then set the **Delegated permissions**.

- ChannelMessage.Read.All - Delegated
- ChannelMessage.Send - Delegated
- ChatMessage.Read - Delegated
- ChatMessage.Send - Delegated
- ChatMember.Read - Delegated
- ChatMember.ReadWrite - Delegated
- Group.ReadWrite.All - Delegated
- offline\_access - Delegated
- profile - Delegated
- Team.ReadBasic.All - Delegated
- User.Read - Delegated
- User.Read.All - Delegated

For each of the list below click Add permission > Microsoft Graph > and then set the **Application permissions**:

- ChannelMember.Read.All - Application
- ChannelMessage.Read.All - Application
- Chat.Create - Application
- Chat.Read.All - Application
- Chat.ReadBasic.All - Application
- Chat.ReadWrite.All - Application
- ChatMember.Read.All - Application
- ChatMember.ReadWrite.All - Application

- ChatMessage.Read.All - Application
- Group.Create - Application
- Group.Read.All - Application
- Group.ReadWrite.All - Application
- GroupMember.Read.All - Application
- GroupMember.ReadWrite.All - Application
- User.Read.All - Application

Once you are done, click **Grant admin consent**

- Go to Overview
- Copy the "Application (client) ID" as your `teams_client_id` in the config
- Copy the "Directory (tenant) ID" as the `teams_tenant_id` in the config.

## Setting up the bot user

The bridge requires a Teams user to be registered as a "bot" to send messages on behalf of Matrix users. You just need to allocate one user from the Teams interface to do this.

- First, you must go to the Azure Active Directory page.
- Click users.
- Click New user.
- Ensure **Create user** is selected.
- Enter a User name ex. "matrixbridge".
- Enter a Name ex. "Matrix Bridge".
- Enter an Initial password.
- Create the user.
- Optionally, set more profile details like an avatar.
- You will now need to log in as this new bot user to set a permanent password (Teams requires you to reset the password on login).
- After logging in you should be prompted to set a new password.
- Enter the bot username and password into config under `teams_bot_username` and `teams_bot_password`

## Getting the groupId

The groupId can be found by opening Teams, clicking ... on a team, and clicking "Get link to team". The groupId is included in the URL `12345678- abcd- efgh- ijkl- lmnopqrstuvw` in this example.

```
https://teams.microsoft.com/l/team/19%3XXX%40thread.tacv2/conversations?groupId=12345678- abcd-efgh-ijkl- lmnopqrstuvw&tenantId=87654321- dcba- hgfe- lkji- zyxwutsrqpo
```

## On the hosting machine

### Generate teams registration keys

```
openssl genrsa -out teams.key 1024
openssl req -new -x509 -key teams.key -out teams.crt -days 365
```

These keys need to be placed in `~/element-enterprise-server/config/legacy/certs/teams` on the machine that you are running the installer on.

## Configure Teams Bridge

From the Installer's Integrations page, click "Install" under "Microsoft Teams Bridge"

For the provided teams.yml, please see the following documentation of the parameters:

```
teams_client_id: # teams app client id
teams_client_secret: # teams app secret
teams_tenant_id: # teams app tenant id
teams_bot_username: # teams bot username
teams_bot_password: # teams bot password
teams_cert_file: teams.crt
teams_cert_private: teams.key
teams_fqdn: <teams bridge fqdn>
teams_bridged_groups:
- group_id: 218b0bfe-05d3-4a63-8323-846d189f1dc1 #change me
 properties:
 autoCreateRooms:
 public: true
 powerLevelContent:
 users:
 "@alice:example.com": 100 # This will add <alice> account as admin
 "@teams-bot:example.com": 100 # the Teams bot mxid
 <bot_sender_localpart>: <domain_name>
 autoCreateSpace: true
 limits:
 maxChannels: 25
 maxTeamsUsers: 25
repeat "- group_id:" section above for each Team you want to bridge

bot_display_name: Teams Bridge Bot
bot_sender_localpart: teams-bot
enable_welcome_room: true
welcome_room_text: |
```



Welcome, your Element host is configured to bridge to a Teams instance.

This means that Microsoft Teams messages will appear on your Element account and you can send messages in Element rooms to have them appear on teams.

To allow Element to access your Teams account, please say `login` and follow the steps to get connected. Once you are connected, you can open the  Explore Rooms dialog to find your Teams rooms.

```
namespaces_prefix_user: OPTIONAL: default to _teams_
```

```
namespaces_prefix_aliases: OPTIONAL: default to teams_
```

- For each Bridged Group, you will need to set a `group_id` and some properties found in the config sample.

You will need to re-run the installer for changes to take effect.

# Setting Up the IRC Bridge

## Overview

The IRC bridge allows you to bridge IRC servers into your Element server.

From the Installer's Integrations page, click "Install" under "IRC Bridge"

Edit the provided bridge.yml based on the following documentation:

- `key_file: irc-passkey.pem` To generate the `irc-passkey.pem` file, please run the following in the `~/element-enterprise-server/legacy/certs/` directory: `openssl genpkey -out passkey.pem -outform PEM -algorithm RSA -pkeyopt rsa_keygen_bits:2048`
- `postgres_fqdn: ircbridge-postgres` Use `ircbridge-postgres` if using `postgres-create-in-cluster` otherwise point this at your external database.
- `postgres_user: ircbridge` Leave this if you are using `postgres-create-in-cluster`.
- `postgres_db: ircbridge` Leave this if you are using `postgres-create-in-cluster`.
- `postgres_password: postgres_password` Set this to either your password for the user connecting to an existing database, or if using `postgres_create_in_cluster`, set this to a new password with `pwgen 32 1`.
- `# postgres_create_in_cluster: true` # uncomment if you want the installer to install postgresql for you. Not supported with the multi-node installer, where you must use an external postgres.
- `postgres_port` Can be used to specify a non-standard port. 5432 is used if not specified. Optional
- `postgres_sslmode` Can be used to specify the sslmode for the Postgres connection. Options are 'disable', 'no-verify' or 'verify-full'. 'disable' is used if not specified. Optional
- Now specify a list of Matrix IDs that have admin access to the IRC bridge such as:

```
admins:
- "@adminuser: dev.local"
- "@adminuser2: dev2.local"
```

- `enable_presence: true` This determines if presence is presented to IRC or not.
- `drop_matrix_messages_after_seconds: 0`
- `bot_username: "ircbridgebot"` The name of the bot.
- `enable_ident: false` Whether or not to enable IRC ident.
- `ident_port_type: # HostPort or NodePort` Required if enabling ident.
- `ident_port_number: 10230` Required if enabling ident.
- `logging_level: info` Set the default logging level of the bridge.
- `enable_provisioning: true`

Next, we have the provisioning rules section, which will make sure that rooms are not bridged if a match is made on these rules. This is useful for preventing bad actors on Matrix from flooding IRC. This section looks like:

```
provisioning_rules:
The bridge checks the joined members of a propective room and checks to see
if any users matching these regex sets are in the room. `exempt` users never
```

```
match, and will be ignored. If any user matches `conflict`, the room will not
be allowed to be bridged until the user is removed. Both sets take a regular expression.
userIds:
 exempt:
 # These users never conflict, even if matching
 - "@doubleagent:badguys.com"
 conflict:
 # These users will deny a room from being bridged.
 - "@*:badguys.com"
provisioning_room_limit: 50
```

- `rmau_limit: 100` Set this to the maximum number of remote monthly active users that you would like to allow in a bridged IRC room.
- `users_prefix: "irc_"` Set a user prefix for irc users.
- `alias_prefix: "irc_"` Set an alias prefix for irc users.
- `address: irc.someserver.net` The address of the irc server to bridge. Now for the above IRC server, we have a set of parameters that can be set:
- `name: "Server Name"` The server name to show on the bridge. Now below that, you'll see the botConfig with these parameters:
- `enabled: true` Leave this on.
- `nick: "MatrixBot"` Nick of the bridge bot
- `username: "matrixbot"` Username of the bridge bot.
- `password: "some_password"` Password of the bridge bot. Generate this with `pwgen 32 1`

For other settings that can also be applied to this config file, please see: <https://github.com/matrix-org/matrix-appservice-irc/blob/develop/config.sample.yaml#L52>

You will need to re-run the installer for changes to take effect.

## Connecting to the Bridge

1. From Element, send a DM to the bridge bot `/msg @ircbridgebot:element.local` where `element.local` is your server's domain name
2. Send the bridge `!whois` to see if you can get logged in to the IRC network

# Setting Up the SIP Bridge

## Configuring SIP bridge

### Basic config

From the Installer's Integrations page, click "Install" under "SIP Bridge"

For the provided sipbridge.yml, please see the following documentation:

- ``postgres_create_in_cluster``: ``true`` to create the postgres db into the k8s cluster. On a standalone deployment, it is necessary to define the ``postgres_data_path``.
- ``postgres_fqdn``: The fqdn of the postgres server. If using ``postgres_create_in_cluster``, you can choose the name of the workload.
- ``postgres_data_path``: `"/mnt/data/sipbridge-postgres"`
- ``postgres_port``: 5432
- ``postgres_user``: The user to connect to the db.
- ``postgres_db``: The name of the db.
- ``postgres_password``: A password to connect to the db.
- ``port_type``: ``HostPort`` or ``NodePort`` depending on which kind of deployment you want to use. On standalone deployment, we advise you to use ``HostPort`` mode.
- ``port``: The port on which to configure the SIP protocol. On ``NodePort`` mode, it should be in kubernetes range:
- ``enable_tcp``: ``true`` to enable TCP SIP.
- ``pstn_gateway``: The hostname of the PSTN Gateway.
- ``external_address``: The external address of the SIP Bridge
- ``proxy`` : The address of the SIP Proxy
- ``user_agent``: A user agent for the sip bridge.
- ``user_avatar``: An MXC url to the sip bridge avatar. Don't define it if you have not uploaded any avatar.
- ``encryption_key``: A 32 character long secret used for encryption. Generate this with ``pwgen 32 1``

# Setting Up the XMPP Bridge

## Configuring the XMPP Bridge

The XMPP bridge relies on the xmpp "component" feature. It is an equivalent of matrix application services. You need to configure an XMPP Component on an XMPP Server that the bridge will use to bridge matrix and xmpp user.

## On the hosting machine

From the Installer's Integrations page, click "Install" under "XMPP Bridge".

For the provided xmpp.yml, please use the following documentation to configure the bridge:

- `xmpp_service`: XMPP Address of the service endpoint.
- `xmpp_domain`: The XMPP FQDN with the External Component subdomain (i.e. `element.xmpp.example.com`)
- `bot_username`: The xmpp bot username on matrix
- `alias_prefix`: The prefix for bridged aliases
- `user_prefix`: The prefix for bridged users
- `enable_portals_gateway`: `true` to enable portals.
- `xmpp_component_password`: Xmpp component password
- `postgres_create_in_cluster`: `true` if you want the installer to automatically set up postgres. Requires `postgres_data_path` if using this.
- `postgres_fqdn`: PostgreSQL server fqdn or ip
- `postgres_user`: PostgreSQL username
- `postgres_db`: PostgreSQL database
- `postgres_port`: PostgreSQL port, default to 5432
- `postgres_password`: PostgreSQL password
- `postgres_create_in_cluster`: Whether or not to create the postgres as a k8s statefulset
- Re-run the installer

In all the examples below the following are set:

- The `domain_name` is your homeserver domain ( the part after : in your MXID ) : `element.local`
- XMPP Server FQDN: `xmpp.example.com`
- XMPP External Component/`xmpp_domain`: `element.xmpp.example.com`

## Prosody Example

If you are configuring prosody, you need the following component configuration (for the sample xmpp server, `element.xmpp.example.com`):

```
Component "element.xmpp.example.com"
 ssl = {
 certificate = "/etc/prosody/certs/tls.crt";
 key = "/etc/prosody/certs/tls.key";
 }
 component_secret = "eeb8choosaim3oothaeGh0aequiop4ji"
```

And then with that configured, you would pass the following into `xmpp.yml`:

```
xmpp_service: xmpp://xmpp.example.com:5347
xmpp_domain: "element.xmpp.example.com" # external component subdomain
xmpp_component_password: eeb8choosaim3oothaeGh0aequiop4ji # xmpp component password
```

Note: We've used `pwgen 32 1` to generate the `component_secret`.

## Joining an XMPP Room

Once you have the XMPP bridge up, you need to map an XMPP room to a Matrix ID. For example, if the room on XMPP is named: `#iwotevo@conference.xmpp.lab.element.com`

(conference is the fqdn of the component's hosting rooms on our xmpp test instance)

then on Matrix, you would join:

```
#_xmpp_iwotevo_conference.xmpp.example.com:element.local
```

The command to do that from within the Element client would be: (assuming your homeserver domain is example.com)

```
/join #_xmpp_iwotevo_conference.xmpp.example.com:element.local
```

## Joining a Matrix room from XMPP

If the Element/Matrix room is public you should be able to query the room list at the external component server address (Ex: element.xmpp.example.com)

The Matrix room at alias `#roomname:element.local` maps to `#roomname#element.local@element.xmpp.example.com` on the XMPP server xmpp.example.com if your `xmpp_domain: element.xmpp.example.com`

Element		XMPP
---------	--	------

<b>#roomname</b> :element.local (native Matrix room)	?	<b>#roomname</b> #element.local@element.xmpp.example.com (bridged into XMPP)
<b>#_xmpp_roomname</b> _conference.xmpp.example.com:element.local (bridged into Matrix/Element)	?	<b>#roomname</b> @conference.xmpp.example.com (native XMPP room)

# Setting up Location Sharing

## Overview

The ability to send a location share, whether static or live, is available without any additional configuration.

However, when *receiving* a location share, in order to display it on a map, the client must have access to a tile server. If it does not, the location will be displayed as text with coordinates.

By default, location sharing uses a MapTiler instance and API key that is sourced and paid for by Element. This is provided free, primarily for personal EMS users and those on Matrix.org.

If no alternate tileserver is configured either on the HomeServer or client then the mobile and desktop applications will fall back to Element's MapTiler instance. Self-hosted instances of Element Web will not fall back, and will show an error message.

## Using Element's MapTiler instance

Customers should be advised that our MapTiler instance is not intended for commercial use, it does not come with any uptime or support SLA, we are not under any contractual obligation to provide it or continue to provide it, and for the most robust privacy customers should either source their own cloud-based tileserver or self-host one on-premises.

However, if they wish to use our instance with Element Web for testing, demonstration or POC purposes, they can configure the `map_style_url` by adding extra configurations in the advanced section of the Element Web page in the installer:

```
{
 "map_style_url":
 "https://api.maptiler.com/maps/streets/style.json?key=fU3v1MsMn4Jb6dnEIFsx"
}
```

## Using a different tileserver

If the customer sources an alternate tileserver, whether from MapTiler or elsewhere, you should enter the tileserver URL in the `extra_client` section of the Well-Known Delegation Integration accessed from the Integrations page in the Installer:



```
{
 ... other info ...
 "m.tile_server": {
 "map_style_url": "http://mytileserver.example.com/style.json"
 }
}
```

## Self-hosting a tileserver

Customers can also host their own tileserver if they wish to dedicate the resources to doing so. Detailed information on how to do so is available [here](#).

# Removing Legacy Integrations

Today, if you remove a Yaml integration's config, its components will not be removed from the cluster automatically. You will also need to manually remove the custom resources from the Kubernetes cluster.

More details to come soon.

# Support Policies

# On-Premise Support Scope of Coverage

For Element Enterprise On-Premise, we support the following:

- Installation and Operation (Configuring the Installer, Debugging Issues)
- Synapse Usage/Configuration/Prioritised Bug Fixes
- Element Web Usage/Configuration/Prioritised Bug Fixes
- Integrations
  - Delegated Auth (e.g. SAML/LDAP) (Add-on)
  - Group Sync (LDAP, AD Graph API, SCIM supported) (Add-on)
  - Github / Gitlab
  - JIRA
  - Webhooks
  - Jitsi
  - Chatterbox (Add-on)
  - Adminbot (Add-on)
  - Auditbot (Add-on)

For Element On-Premise, we support the following:

- Installation and Operation (Configuring the Installer, Debugging Issues)
- Synapse Usage/Configuration/Prioritised Bug Fixes
- Element Web Usage/Configuration/Prioritised Bug Fixes
- Integrations
  - Github / Gitlab
  - JIRA
  - Webhooks
  - Jitsi

The following items are **not** included in support coverage:

- General Infrastructure Assistance
- K8s Assistance
- Operating System Support
- Postgresql Database Support

For single node setups, the following also applies:

- Element does not support deployment to a microk8s that was not installed by our installer.
- Element does not provide a backup solution.
- Element does not provide support for any underlying storage.

For kubernetes deployments, the following also applies:

- Element does not support deploying the installer created postgresql in a kubernetes environment.
- Element requires that you deploy postgresql separately in a kubernetes environment, external to your Element deployment.

# Single Node Scope of Coverage Addendum

- List of what is supported for this setup exists.
  - Element supports the installation of microk8s using our installer on all installer supported platforms.
  - Element supports upgrading microk8s using our installer on all installer supported platforms.
  - Element supports the installation, configuration, and maintenance of our on-premise software delivered by the installer running on microk8s.
  - Element provides diagnosis and bug fixes for our on-premise software delivered by the installer running on microk8s.
- List of what is not supported in this setup exists.
  - Element does not support the underlying operating system.
  - Element does not support deployment to microk8s that was installed separately from our installer.
  - Element does not provide a backup solution.
  - Element does not provide support for any underlying storage.
- Create a checklist of what makes a single node production workload.
  - RHEL 8+ or Ubuntu 20.04+
  - microk8s installed and running.
  - Customer provided backup solution in place.
  - Customer managed storage in place.
  - synapse, haproxy, and element-web all running.
  - Optionally, dimension, adminbot, auditbot, group sync, and hookshot may be running as well.

# Archived Documentation Repository

Archived Documentation Repository

# Documentation Covering Installer 2023-02.02 CLI Only.

element-on-premise-documentation (2).pdf

Archived Documentation Repository

# Documentation Covering Installers From 2022.10.01 to 2023.02.01

element-on-premise-documentation.pdf



Archived Documentation Repository

# Documentation Covering Installers From 2022.07.03 to 2022.09.05

element-on-premise-documentation-0703-0905.pdf

Archived Documentation Repository

# Documentation covering v1 and installers prior to 2022-07.03

element-on-premise-documentation-july28-2022.pdf

# Appendices

# Appendix A: Preparing Element Server Suite PoC

Please reach out our Element Sales Team if you want to run a Proof of Concept for Element Server Suite.

**Note** This guide is for running Proof of Concepts. We don't aim to show every feature here, we want to get you up and running most quickly. This guide is focusing on connected standalone installations currently. There are scenarios currently not covered by this guide. Installing into airgapped / disconnected environments, or testing our Cloud Based offering.

A Proof-of-Concept is done in preparation of a subscription sale with the goal of demonstrating the required capabilities.

## Overview

1. Create an account on element.io
2. Communication via matrix space
3. PoC preparation
  - 3.1 Preparation of the VM
  - 3.2 DNS names & certificates for the endpoints
  - 3.3 Matrix IDs & Well Known delegation
  - 3.4 Authentication & Postgres DB

## 1. Create an account on element.io

Please create an account on element.io. We will enable this account as part of the PoC process and grant you access to the Enterprise Server Suite software packages.

## 2. Communication via matrix room

The account team will create a matrix room to improve communication and invite you. We will need your Matrix ID (MXID) for this. If you don't already have a MXID, you can create one here by signing up. This will create an account on matrix.org, you can authenticate via serverial identity providers. Send the MXID to the account team, so they can add you to the room. You could use the Element Web Client that you used to create the account or install one of the Element Mobile apps from the App or Playstore.

## 3. PoC preparation

Element Server Suite can be installed in a Kubernetes Cluster or as a standalone installation on top of an Operating System (RHEL 8 or Ubuntu 20.04). Most Proof-of-Concept installations will select the Standalone Installation on top of a VM which we recommend for speed and ease of operation.

Click here for an overview of the Element Server Suite. Here is the link detailing the single node installation.

## 3.1 Preparation of the VM

Please set up a VM with **8 vCPUs** and **32GB RAM** and **100 GB Storage**. If this sounds like a lot of resources to you, the requirements do in fact vary and could be scaled down later if required. Install Ubuntu 20.04 LTS or RHEL8. Update the system to the latest available patches and create a user to be used for maintaining the Element Server Suite. See our documentation for this step here.

## 3.2 DNS Names and Certificates

You need to select a base domain for the Server. This can differ from the base domain of the matrix IDs but is often the same. Read more about this in the section on Matrix IDs and Well Known delegation below.

You have chosen eng.acme.com. The following DNS entries must be prepared and point to the external IP of the VM.

This results in the following hostnames for you :

- eng.acme.com (base domain - might already exist )
- matrix.eng.acme.com (the synapse homeserver)
- element.eng.acme.com (element web)
- admin.eng.acme.com (admin dashboard)

Optional for Monitoring and Integrations :

- grafana.eng.acme.com (Our Grafana server)
- hookshot.eng.acme.com (Our integrations)
- integrator.eng.acme.com (integration manager)

Optional for Video Chat with Jitsi :

- jitsi.eng.acme.com (Our VoIP platform)
- coturn.eng.acme.com (Our TURN server)

Optional for the Admin Backdoor functionality :

- roomadmin.eng.acme.com

We require certificates for these hostnames to enable SSL/TLS encryption. The quick and easy way is to use the embedded letsencrypt. This is only available if you are in a connected environment. You can provide and use your own certificates.

## 3.3 Matrix IDs & Well Know delegation

Matrix IDs have the following format :

**@USER:SERVER**

In our example case the matrix server is `matrix.eng.acme.com`. If a user Tom Maier has a username **tmaier** in your LDAP, this would lead to an MXID `@tmaier:matrix.eng.acme.com`. This is often not desired as we like to keep the MXIDs short. It is more elegant to drop the "matrix" host name from the MXIDs. Tom's MXID would look like this `@tmaier:eng.acme.com`.

In order to be able to offer matrix IDs with the base domain, we recommend setting up a reverse proxy on `eng.acme.com`, which forwards `https://eng.acme.com/.well-known/matrix/` to the matrix/synapse server on `https://matrix.eng.acme.com/.well-known/matrix`. Or you shorten the hostname part of your MXIDs even more to `acme.com`, this would require you to put the reverse proxy onto `acme.com`.

The configuration on your Apache WebServer should be similar to this :

```
ProxyPass /.well-known/matrix/ https://matrix.eng.acme.com/.well-known/matrix/
ProxyPassReverse /.well-known/matrix/ https://matrix.eng.acme.com/.well-known/matrix/
ProxyPreserveHost On
```

More about well-known and MXIDs can be found in our Upstream Documentation [here](#) and [here](#). Further configurations can be made using the well-known mechanism. An example is documented [here](#).

## 3.4 Authentication and Postgres DB

The quickest setup is using local authentication and users only. This is what we recommend for a first step in a Proof-of-Concept situation. User accounts are created in the local Postgresql DB through our Admin UI or through API scripts for automation in this case. We support many mechanisms for Authentication like LDAP, SAML2 and OIDC. We recommend to configure these as a 2nd step only if required.

You have the option to use an internal or external Postgres DB. We do recommend to use the internal Postgres DB for Proof-of-Concept installations. The internal Postgres DB is only available when you are opting for the Standalone Installation on top of an Operating System. You will need an external Postgres DB when installing into an existing Kubernetes cluster.

## Checklist before starting the installation

Please prepare the above items before starting the installation. Make sure you have :

- created and communicated your MXID to the Element Sales Team
- registered an account on `element.io`
- created and prepared your vm / machine with enough resources
- created DNS entries
- decided on letsencrypt / created host certificates for your hostnames
- installed the reverse proxy on the webserver of your MXID URL e.g. `eng.acme.com` or `acme.com`

Don't hesitate to reach out to your Element Sales Team for support. We are here to guide you.

# Migration from self-hosted to ESS On-Premise

## Notes

Before starting with this guide, please contact EMS support from <https://ems.element.io/support> or by emailing [ems-support@element.io](mailto:ems-support@element.io)

- Except where specified, you should be able to just copy-paste each command in succession.
- Please do not change any file names anywhere.

## Preparation

This section outlines what you should do ahead of the migration in order to ensure the migration goes as quickly as possible and without issues.

- At the latest 48 hours before your migration is scheduled, set the TTL on any DNS records that need to be updated to the lowest allowed value.
- Upgrade your Synapse to the same version as EMS is running. Generally this will be the latest stable release. [https://element.ems.host/\\_matrix/federation/v1/version](https://element.ems.host/_matrix/federation/v1/version) is a good indicator, but confirm version with your EMS contact.
  - This is not required, but if your Synapse version is not the same as the EMS version, your migration will take longer.
- Check the size of your database:
  - PostgreSQL: Connect to your database and issue the command `\l+`
  - SQLite: `ls -lah /path/to/homeserver.db`
- Check the size of your media repository and report to your EMS contact.
  - Synapse Media Store: `du -hs /path/to/synapse/media_store/`
  - Matrix Media Repo: <https://github.com/turt2live/matrix-media-repo/blob/master/docs/admin.md#per-server-usage>
- If you are using SQLite instead of PostgreSQL, you should port your database to PostgreSQL by following this guide before dumping your database

## SSH to your matrix server

You might want to run everything in a `tmux` or a `screen` session to avoid disruption in case of a lost SSH connection.

# Upgrade Synapse to the same version EES is running

Follow <https://matrix-org.github.io/synapse/latest/upgrade.html>

Start Synapse, make sure it's happy Stop Synapse

## Create a folder to store everything

```
mkdir -p /tmp/synapse_export
cd /tmp/synapse_export
```

The guide from here on assumes your current working directory is `/tmp/synapse_export`.

## Set restrictive permissions on the folder

If you are working as root: (otherwise set restrictive permissions as needed):

```
chmod 000 /tmp/synapse_export
```

## Copy Synapse config

Copy the following files and send to EMS Support:

- Your Synapse configuration file (usually `homeserver.yaml`)
- Your message signing key.
  - This is stored in a separate file. See the Synapse config file [`homeserver.yaml`] for the path. The variable is `signing_key_path` [https://github.com/matrix-org/synapse/blob/v1.32.2/docs/sample\\_config.yaml#L1526-L1528](https://github.com/matrix-org/synapse/blob/v1.32.2/docs/sample_config.yaml#L1526-L1528)

## Stop Synapse

### **DO NOT START IT AGAIN AFTER THIS**

Doing so can cause issues with federation and inconsistent data for your users.

While you wait for the database to export or files to transfer, you should edit or create the well-known files and DNS records to point to your new EES host. This can take a while to update so should be done as soon as possible in order to ensure your server will function properly when the migration is complete.



# Database export

## PostgreSQL

### Dump, compress

Replace:

- `<dbhost>` (ip or fqdn for your database server)
- `<dbusername>` (username for your synapse database)
- `<dbname>` (the name of the database for synapse)

```
pg_dump -0 -h <dbhost> -U <dbusername> -d <dbname> | gzip > synapse_db_export.sql.gz
```

## Setup new host

[LINK TO ON\_PREM SETUP DOCS]

On the new host set the

grab `macaroon_secret_key` from `homeserver.yaml` and place it in the "Secrets \ Synapse \ Macaroon"

## Import DB

Enter a bash shell on the Synapse postgres container:

```
kubectl exec -it -n element-onprem synapse-postgres-0 --container postgres -- /bin/bash
```

```
psql -U synapse_user synapse on postgres container shell
```

**THE FOLLOWING COMMAND WILL ERASE THE EXISTING SYNAPSE DATABASE WITHOUT WARNING OR CONFIRMATION. PLEASE ENSURE THAT IT IS THE CORRECT DB AND THERE IS NO PRODUCTION DATA ON IT**

```
DO $$ DECLARE
r RECORD;
BEGIN
 FOR r IN (SELECT tablename FROM pg_tables WHERE schemaname = current_schema()) LOOP
 EXECUTE 'DROP TABLE ' || quote_ident(r.tablename) || ' CASCADE';
 END LOOP;
END $$;
```

```
DROP sequence cache_invalidation_stream_seq;
DROP sequence state_group_id_seq;
DROP sequence user_id_seq;
DROP sequence account_data_sequence;
DROP sequence application_services_txn_id_seq;
DROP sequence device_inbox_sequence;
DROP sequence event_auth_chain_id;
DROP sequence events_backfill_stream_seq;
DROP sequence events_stream_seq;
DROP sequence presence_stream_sequence;
DROP sequence receipts_sequence;
DROP sequence un_partial_stated_event_stream_sequence;
DROP sequence un_partial_stated_room_stream_sequence;
```

\q to quit

on host:

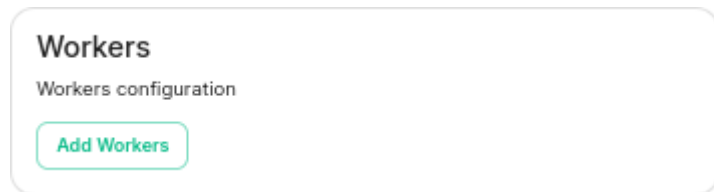
```
gzip -d synapse_export.sql.gz
sudo cp synapse_export.sql /data/postgres/synapse/
```

on pod: `cd /var/lib/postgresql/data`

```
psql -U synapse_user synapse < var/lib/postgresql/data/synapse_export.sql
```

# Configuring Synapse workers

From the Installer's Synapse page, scroll down to Synapse workers view.



Click on Add Workers

## Workers

Workers configuration

### Additional

Arbitrary extra config to inject into the Synapse worker configuration as a YAML string

1

Instances

1



Default

Number of instances of this worker type

Type

Default

Type of worker being configured

### Resources

Kubernetes resources to allocate to each instance.

#### Limits

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manager-compute-resources-container/>

Memory  Default

Name

Add to Limits

#### Requests

Requests describes the minimum amount of compute resources required. More info: <https://kubernetes.io/docs/concepts/configuration/manager-compute-resources-container/>

CPU  Default

Memory  Default

Name

Add to Requests

Add more Workers

You have to select a Worker Type. Here are the workers which can be useful to you :

1. Pushers : If you experience slowness with notifications sending to clients
2. Client-Reader : If you experience slowness when clients login and sync their chat rooms
3. Synchrotron : If you experience slowness when rooms are active
4. Federation-x : If you are working in a federated setup, you might want to dedicate federation to workers.

If you are experiencing resources congestion, you can try to reduce the resources requested by each worker. Be aware that

- if the node gets full of memory, it will try to kill containers which are consuming more than what they requested
- if a container consumes more than its memory limit, it will be automatically killed by the node, even if there is free memory left.

You will need to re-run the installer after making these changes for them to take effect.

# Setting up Delegated Authentication with LDAP on Windows AD

In the installer, set the following fields:

- **Base** : the distinguished name of the root level Org Unit in your LDAP directory. The distinguished name can be displayed by selecting **View / Advanced Features** in the Active Directory console and then, right-clicking on the object, selecting **Properties / Attributes Editor** .

Base \*  
OU=LFDN,DC=olivier,DC=sales-demos,DC=elemen  
The LDAP Base search

Bind Dn \*  
CN=Administrator,CN=Users,DC=olivier,DC=sales-  
The LDAP Bind DN

Filter  
The LDAP Filter

Uri \*  
ldap://3.75.187.130  
The ldap URI, usually your domain controller

Secrets / Synapse / LDAP Bind Password

LDAP Bind Password  
Password used for ldap bind dn

- **Base Dn** : the distinguished name of the LDAP account with read access.
- **Filter** : an LDAP filter to filter out objects under the LDAP Base DN.
- **Uri** : the URI of your LDAP server.
- **LDAP Bind Password** : the password of the LDAP account with read access.

## Attributes

### LDAP to Synapse attribute mapping

Mail

**mail**

The ldap attribute mapped to synapse mail

Name

**cn**

The ldap attribute mapped to synapse name

UID

**sAMAccountName**

The ldap attribute mapped to synapse uid

# Setting up Delegated Authentication with OpenID on Microsoft Azure

Before setting up the installer, you have to configure Microsoft Azure Active Directory.

## Set up Microsoft Azure Active Directory

- You need to create an `App registration`.
- You have to select `Redirect URI (optional)` and set it to `https://matrix.your-domain.com/_synapse/client/oidc/callback`

### Register an application ...

The user-facing display name for this application (this can be changed later).

#### Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (olivier-doc only - Single tenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- Personal Microsoft accounts only

[Help me choose...](#)

#### Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Register an app you're working on here. Integrate gallery apps and other apps from outside your organization by adding from [Enterprise applications](#).

For the bridge to be able to operate correctly, navigate to API permissions, add Microsoft Graph APIs, choose Delegated Permissions and add

- openid
- profile

Remember to grant the admin consent for those.

To setup the installer, you'll need

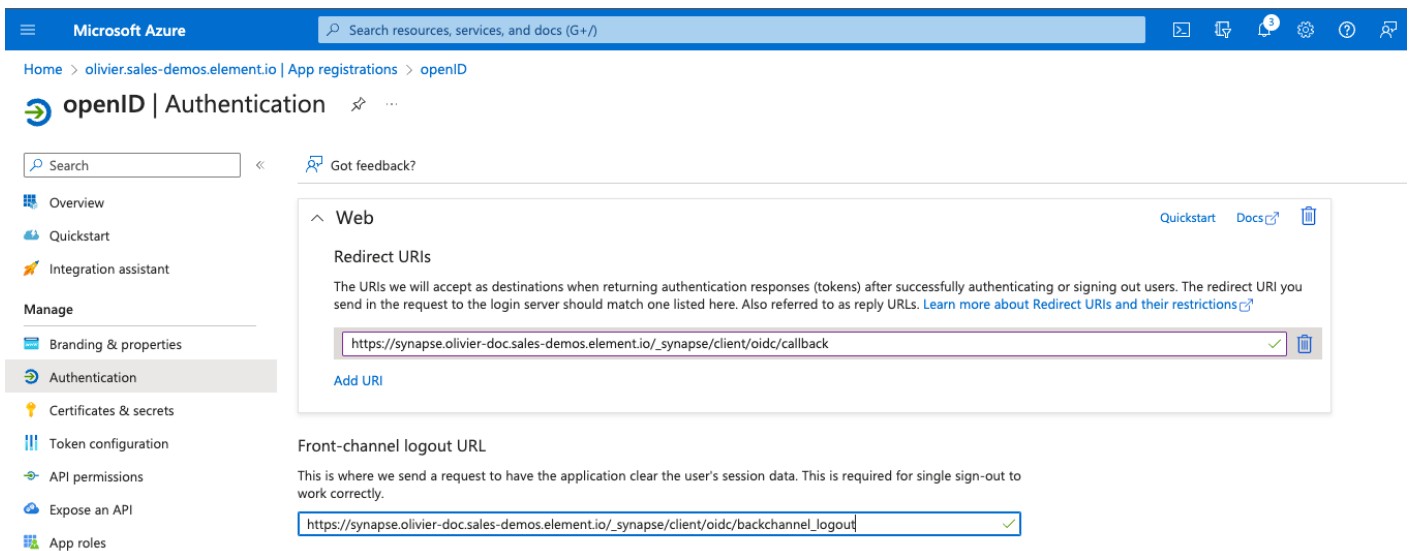
- the `Application (client) ID`
- the `Directory (tenant) ID`
- a secret generated from `Certificates & secrets` on the app.



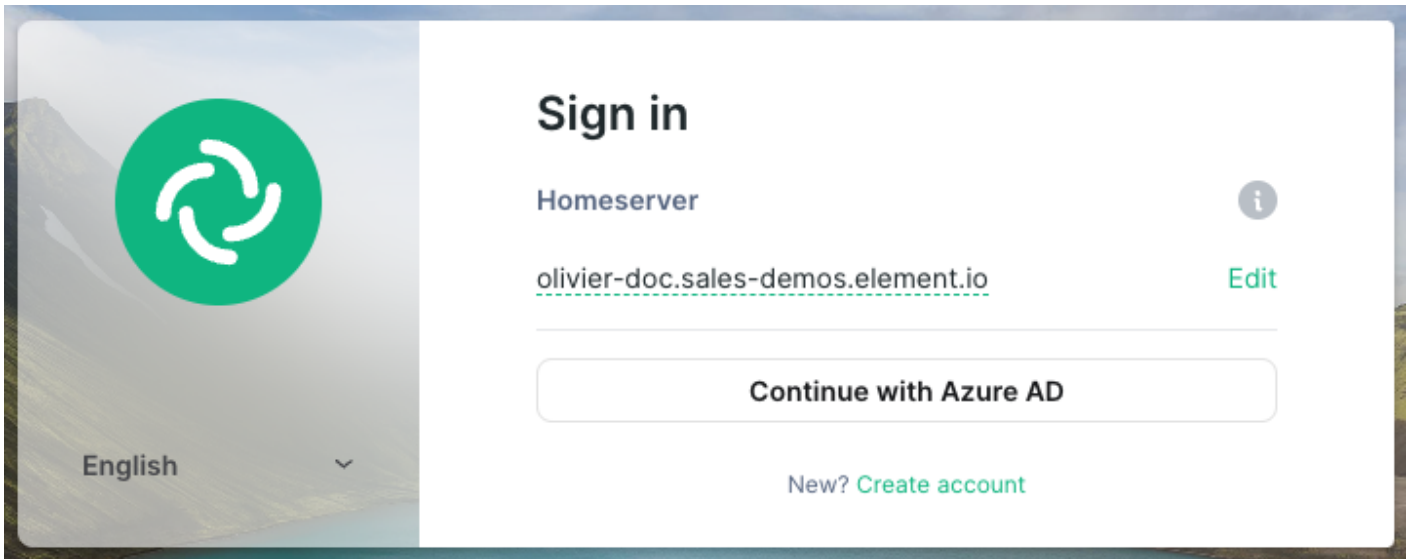
# Configure the installer

Add an OIDC provider in the 'Synapse' configuration after enabling `Delegated Auth` and set the following fields in the installer:

- `Allow Existing Users`: if checked, it allows a user logging in via OIDC to match a pre-existing account instead of failing. This could be used if switching from password logins to OIDC.
- `Authorization Endpoint`: the oauth2 authorization endpoint. Required if provider discovery is disabled.  
`https://login.microsoftonline.com/<Directory (tenant) ID>/oauth2/v2.0/authorize`
- `Backchannel Logout Enabled`: Synapse supports receiving OpenID Connect Back-Channel Logout notifications. This lets the OpenID Connect Provider notify Synapse when a user logs out, so that Synapse can end that user session. This property has to be set to `https://your-domain/_synapse/client/oidc/backchannel_logout` in your identity provider



- `Client Auth Method`: auth method to use when exchanging the token. Set it to `Client Secret Post` or any method supported by your Idp
- `Client ID`: your `Application (client) ID`
- `Discover`: enable/disable the use of the OIDC discovery mechanism to discover endpoints
- `Idp Brand`: an optional brand for this identity provider, allowing clients to style the login flow according to the identity provider in question
- `Idp ID`: a string identifying your identity provider in your configuration
- `Idp Name`: A user-facing name for this identity provider, which is used to offer the user a choice of login mechanisms in the Element UI. In the screenshot below, `Idp Name` is set to `Azure AD`



- **Issuer**: the OIDC issuer. Used to validate tokens and (if discovery is enabled) to discover the provider's endpoints  
[https://login.microsoftonline.com/<Directory \(tenant\) ID>/v2.0](https://login.microsoftonline.com/<Directory (tenant) ID>/v2.0)
- **Token Endpoint**: the oauth2 authorization endpoint. Required if provider discovery is disabled.
- **Client Secret**: your secret value defined under "Certificates and secrets"

**openID | Certificates & secrets** ✨ ...

Search << Got feedback?

Overview  
Quickstart  
Integration assistant

**Manage**

- Branding & properties
- Authentication
- Certificates & secrets**
- Token configuration
- API permissions
- Expose an API
- App roles
- Owners

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable location (using an HTTPS scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Application registration certificates, secrets and federated credentials can be found in the tabs below.

Certificates (0) **Client secrets (1)** Federated credentials (0)

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.


+ New client secret


Description	Expires	Value	Secret ID
Password uploaded on Wed ...	5/2/2025	G0P8Q~-5Yxf3wPcXuYllpJy...	fb62af90-740e-498c-af03-...

- **Scopes**: add every scope on a different line
  - The openid scope is required which translates to the Sign you in permission in the consent UI
  - You might also include other scopes in this request for requesting consent.

## Scopes

A list of scopes requested during the authorization process.

 Standard scopes include openid, profile, email.  [Edited](#)

 Standard scopes include openid, profile, email.  [Edited](#)

[ADD MORE SCOPES](#)

- User Mapping Provider: Configuration for how attributes returned from a OIDC provider are mapped onto a matrix user.

## User Mapping Provider

### Subject Template

The claim used to identify the subject of the ID token.

### Picture Template

The template used to generate the user's profile picture URL in Matrix.

### Localpart Template

The template used to generate the local part of the user's Matrix ID.

Localpart Template \*

```
{{ user.preferred_username.split('@')[0] }}
```

The template used to generate the local part of the user's Matrix ID.

### Display Name Template

The template used to generate the user's display name in Matrix.

Display Name Template \*

```
{{ user.name }}
```

The template used to generate the user's display name in Matrix.

### Email Template

The template used to generate the user's email address in Matrix.

- **Localpart Template**: Jinja2 template for the localpart of the MXID. Set it to `{{ user.preferred_username.split('@')[0] }}` for Azure AD
- **Display Name Template**: Jinja2 template for the display name to set on first login. If unset, no displayname will be set. Set it to `{{ user.name }}` for Azure AD

Other configurations are documented [here](#).

# Setting up Delegated Authentication with OpenID on Microsoft AD FS

## Install Microsoft AD FS

Before starting the installation, make sure:

- your Windows computer name is correct since you won't be able to change it after having installed AD FS
- you configured your server with a static IP address
- your server joined a domain and your domain is defined under Server Manager > Local server
- you can resolve your server FQDN like computername.my-domain.com

You can find a checklist here.

Steps to follow:

- Install AD CS (Certificate Server) to issue valid certificates for AD FS. AD CS provides a platform for issuing and managing public key infrastructure [PKI] certificates.
- Install AD FS (Federation Server)

## Install AD CS

You need to install the AD CS Server Role.

- Follow this guide.

## Obtain and Configure an SSL Certificate for AD FS

Before installing AD FS, you are required to generate a certificate for your federation service. The SSL certificate is used for securing communications between federation servers and clients.

- Follow this guide.
- Additionally, this guide provides more details on how to create a certificate template.

## Install AD FS

You need to install the AD FS Role Service.

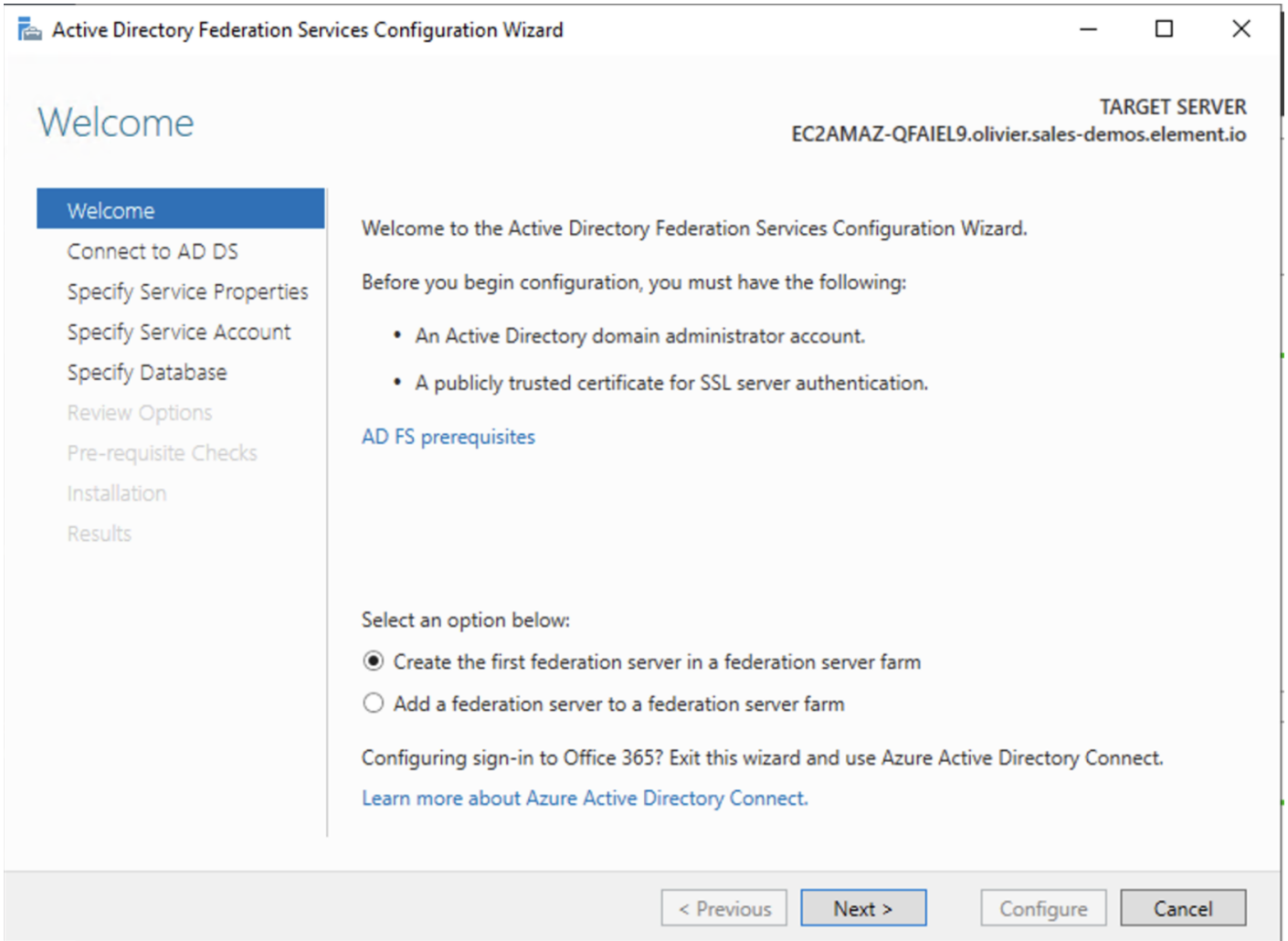
- Follow this guide.

## Configure the federation service

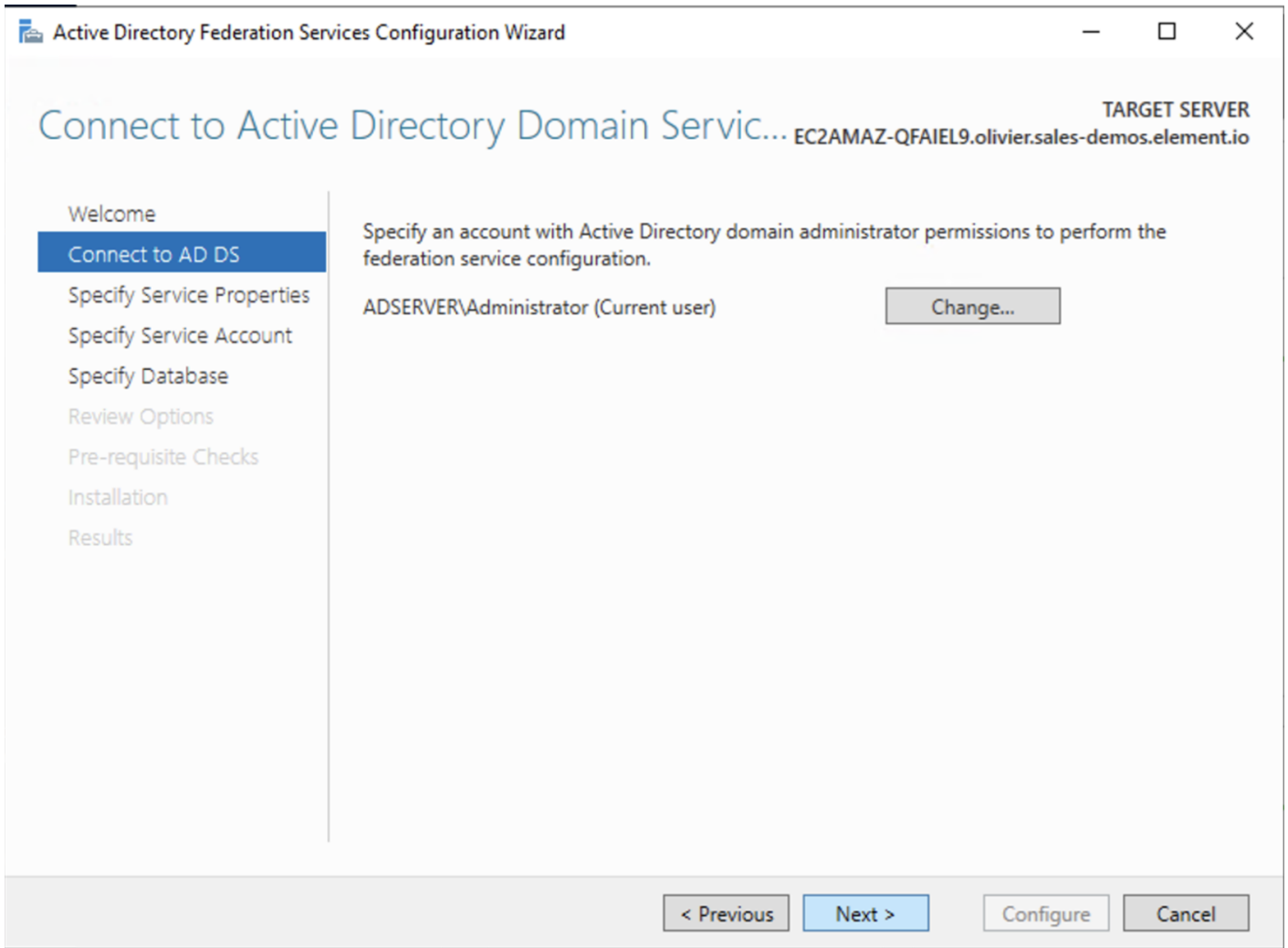
AD FS is installed but not configured.

- Click on `Configure the federation service on this server` under `Post-deployment configuration` in the `Server Manager`.

- Ensure `Create the first federation server in a federation server farm` and is selected



- Click `Next`



- Select the SSL Certificate and set a Federation Service Display Name

Active Directory Federation Services Configuration Wizard

Specify Service Properties

TARGET SERVER  
EC2AMAZ-QFAIEL9.olivier.sales-demos.element.io

Welcome  
Connect to AD DS  
Specify Service Properties  
Specify Service Account  
Specify Database  
Review Options  
Pre-requisite Checks  
Installation  
Results

SSL Certificate: ec2amaz-qfaiel9.olivier.sales-der Import...

View

Federation Service Name: ec2amaz-qfaiel9.olivier.sales-der  
Example: fs.contoso.com

Federation Service Display Name: ACME Corp  
Users will see the display name at sign in.  
Example: Contoso Corporation

< Previous Next > Configure Cancel

- On the Specify Service Account page, you can either Create a Group Managed Service Account (gMSA) or Specify an existing Service or gMSA Account



## Specify Service Account

TARGET SERVER  
EC2AMAZ-QFAIEL9.olivier.sales-demos.element.io

Welcome

Connect to AD DS

Specify Service Properties

Specify Service Account

Specify Database

Review Options

Pre-requisite Checks

Installation

Results

Specify a domain user account or group Managed Service Account.

 Create a Group Managed Service AccountAccount Name: ADSERVER\  Use an existing domain user account or group Managed Service AccountAccount Name: ADSERVER\Adminis...  Account Password: 

&lt; Previous

Next &gt;

Configure

Cancel

- Choose your database

Active Directory Federation Services Configuration Wizard

TARGET SERVER  
EC2AMAZ-QFAIEL9.olivier.sales-demos.element.io

## Specify Configuration Database

Welcome

Connect to AD DS

Specify Service Properties

Specify Service Account

**Specify Database**

Review Options

Pre-requisite Checks

Installation

Results

Specify a database to store the Active Directory Federation Service configuration data.

Create a database on this server using Windows Internal Database.

Specify the location of a SQL Server database.

Database Host Name:

Database Instance:

*To use the default instance, leave this field blank.*

< Previous   Next >   Configure   Cancel

- Review Options , check prerequisites are completed and click on **Configure**
- Restart the server

## Add AD FS as an OpenID Connect identity provider

To enable sign-in for users with an AD FS account, create an Application Group in your AD FS.  
To create an Application Group, follow these steps:

- In **Server Manager** , select **Tools** , and then select **AD FS Management**
- In AD FS Management, right-click on **Application Groups** and select **Add Application Group**
- On the Application Group Wizard **Welcome** screen
  - Enter the Name of your application
  - Under **Standalone applications** section, select **Server application** and click **Next**

Add Application Group Wizard ✕

### Welcome

**Steps**

- Welcome
- Server application
- Configure Application Credentials
- Summary
- Complete

Name:

Description:

Template:

Client-Server applications

- Native application accessing a web API
- Server application accessing a web API
- Web browser accessing a web application

Standalone applications

- Native application
- Server application**
- Web API

- Enter `https://<matrix domain>/_synapse/client/oidc/callback` in Redirect URI: field, click `Add`, save the `Client Identifier` somewhere, you will need it when setting up Element and click `Next` (e.g. `https://matrix.domain.com/_synapse/client/oidc/callback`)

**Server application****Steps**

- Welcome
- Server application
- Configure Application Credentials
- Summary
- Complete

Name:

Synapse server - Server application

Client Identifier:

Redirect URI:

Example: https://Contoso.com

Add

[https://matrix.olivier-openid-ads.sales-demos.element.io/\\_synapse/client/oidc/callback](https://matrix.olivier-openid-ads.sales-demos.element.io/_synapse/client/oidc/callback)

Remove

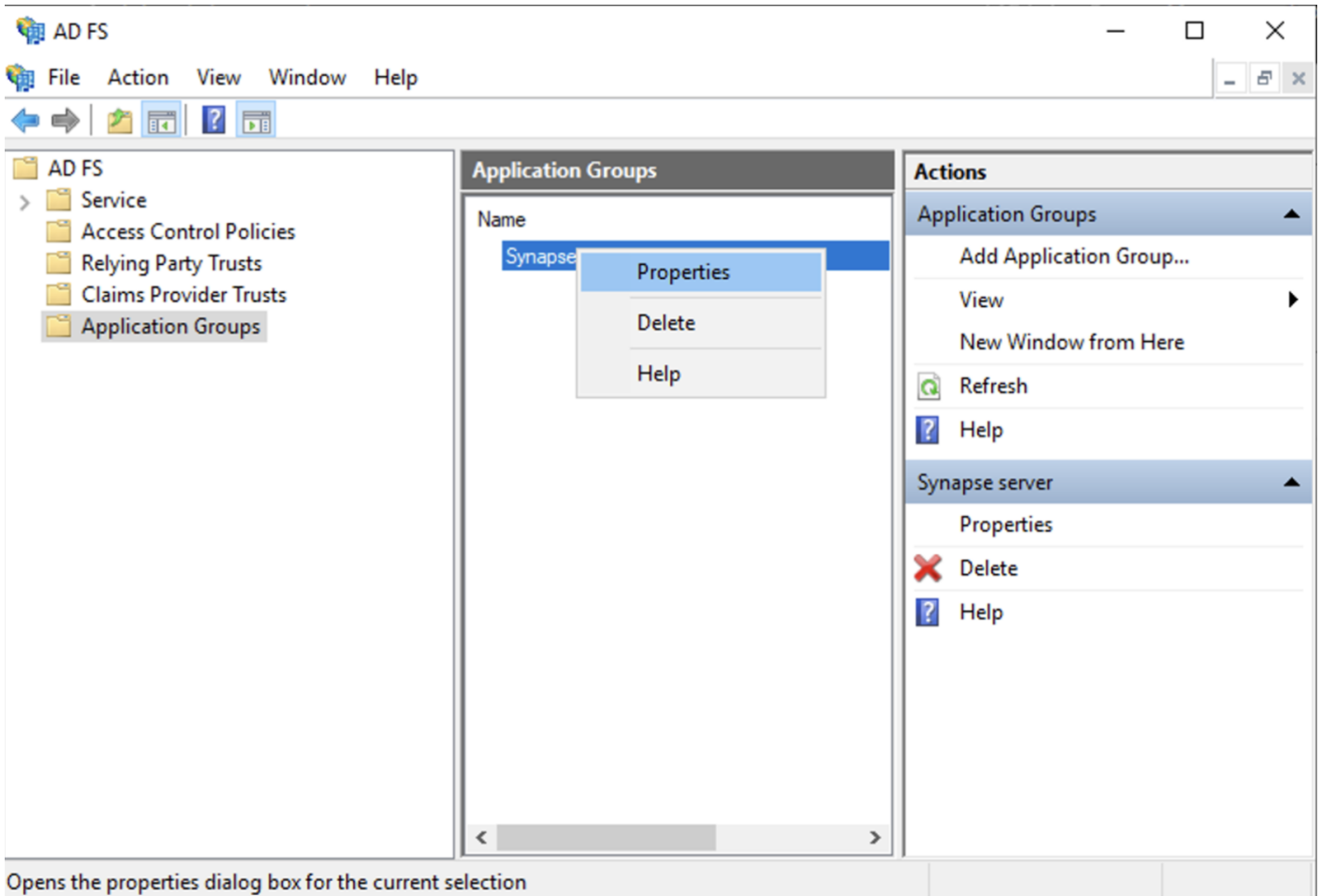
Description:

&lt; Previous

Next &gt;

Cancel

- Select  Generate a shared secret checkbox and make a note of the generated Secret and press  (Secret needs to be added in the Element Installer GUI in a later step)
- Right click on the created Application Group and select `Properties`



Opens the properties dialog box for the current selection

- Select `Add application...` button.
- Select `Web API`
- In the `Identifier` field, type in the `client_id` you saved before and click `Next`

Add a new application to Synapse server

### Configure Web API

**Steps**

- Welcome
- Configure Web API
- Apply Access Control Policy
- Configure Application Permissions
- Summary
- Complete

Name: Synapse server - Web API

Identifier: Example: https://Contoso.com

e011a4c6-d295-4080-b666-9047be465bf3

Description:

< Previous **Next >** Cancel

- Select `Permit everyone` and click `Next`
- Under Permitted scopes: select `openid` and `profile` and click `Next`

Add a new application to Synapse server X

## Configure Application Permissions

**Steps**

- Welcome
- Configure Web API
- Apply Access Control Policy
- Configure Application Permissions
- Summary
- Complete

Configure permissions to enable client applications to access this Web API.

Client application (caller):

Name	Description
Synapse server - Server application	

Permitted scopes:

Scope Name	Description
<input type="checkbox"/> allatclaims	Requests the access token claims in the identity token.
<input type="checkbox"/> aza	Scope allows broker client to request primary refresh token.
<input type="checkbox"/> email	Request the email claim for the signed in user.
<input type="checkbox"/> logon_cert	The logon_cert scope allows an application to request logon...
<input checked="" type="checkbox"/> openid	Request use of the OpenID Connect authorization protocol.
<input checked="" type="checkbox"/> profile	Request profile related claims for the signed in user.
<input type="checkbox"/> user_imperso...	Request permission for the application to access the resour...
<input type="checkbox"/> von_cert	The von_cert scope allows an application to request VPN ...

- On `Summary` page, click `Next`
- Click `Close` and then `OK`

## Export Domain Trusted Root Certificate

- Run `mmc.exe`
- Add the `Certificates` snap-in
  - File/Add snap-in for `Certificates`, `Computer account`
- Under `Trusted Root Certification Authorities / Certificates`, select your DC cert
- Right click and select `All Tasks / Export...` and export as `Base-64 encoded X 509 (.CER)`
- Copy file to local machine

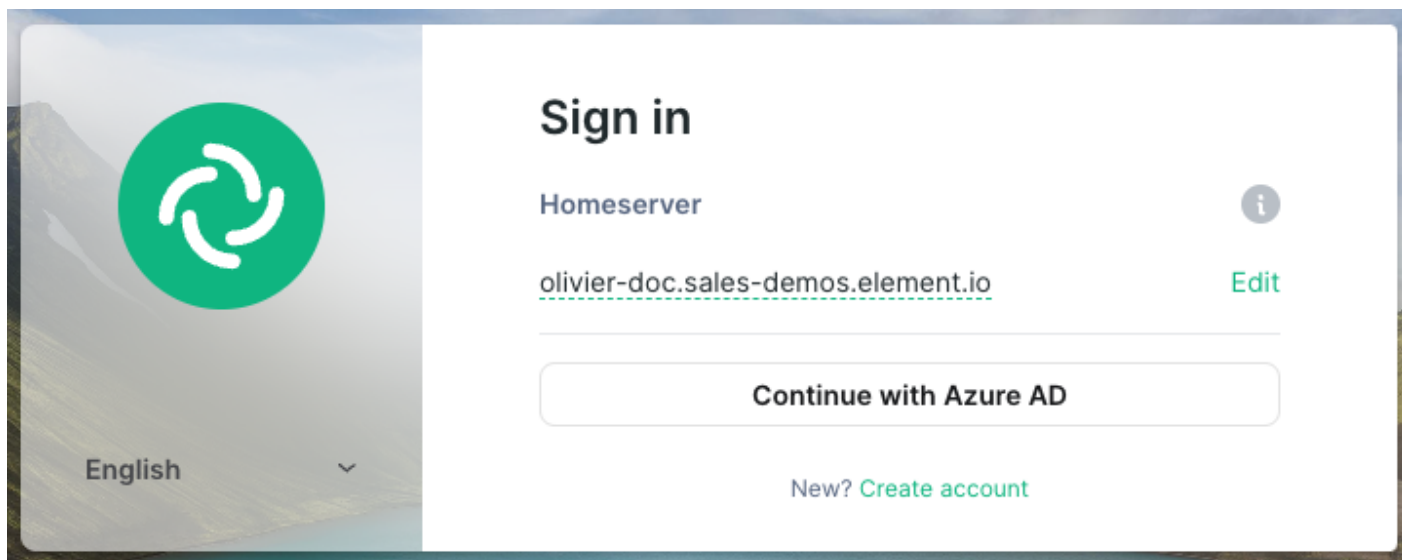
## Configure the installer

Add an OIDC provider in the 'Synapse' configuration after enabling `Delegated Auth` and set the following fields in the installer:

- `Allow Existing Users`: if checked, it allows a user logging in via OIDC to match a pre-existing account instead of failing. This could be used if switching from password logins to OIDC.
- `Authorization Endpoint`: the oauth2 authorization endpoint. Required if provider discovery is disabled.

https://login.microsoftonline.com/<Directory (tenant) ID>/oauth2/v2.0/authorize

- **Backchannel Logout Enabled**: Synapse supports receiving OpenID Connect Back-Channel Logout notifications. This lets the OpenID Connect Provider notify Synapse when a user logs out, so that Synapse can end that user session.
- **Client Auth Method**: auth method to use when exchanging the token. Set it to **Client Secret Basic** or any method supported by your Idp
- **Client ID**: the **Client ID** you saved before
- **Discover**: enable/disable the use of the OIDC discovery mechanism to discover endpoints
- **Idp Brand**: an optional brand for this identity provider, allowing clients to style the login flow according to the identity provider in question
- **Idp ID**: a string identifying your identity provider in your configuration
- **Idp Name**: A user-facing name for this identity provider, which is used to offer the user a choice of login mechanisms in the Element UI. In the screenshot below, **Idp Name** is set to **Azure AD**

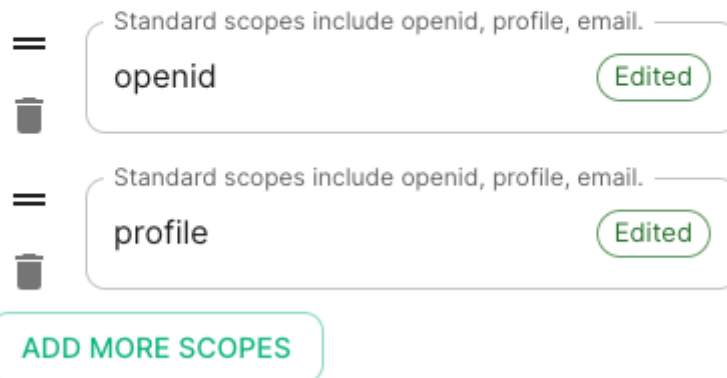


- **Issuer**: the OIDC issuer. Used to validate tokens and (if discovery is enabled) to discover the provider's endpoints `https://<your-ads.domain.com>/ads/`
- **Token Endpoint**: the oauth2 authorization endpoint. Required if provider discovery is disabled.
- **Client Secret**: your client secret you saved before.
- **Scopes**: add every scope on a different line
  - The openid scope is required which translates to the Sign you in permission in the consent UI
  - You might also include other scopes in this request for requesting consent.



## Scopes

A list of scopes requested during the authorization process.



The screenshot shows a configuration interface for scopes. At the top, there is a title 'Scopes' and a subtitle 'A list of scopes requested during the authorization process.' Below this, there are two entries for scopes. Each entry consists of a list icon (three horizontal lines) and a trash icon (a small square with a diagonal line) on the left. To the right of these icons is a rounded rectangular box containing the scope name. Above the scope name, there is a small text label 'Standard scopes include openid, profile, email.' and a minus sign. The first entry has the scope name 'openid' and a green 'Edited' button to its right. The second entry has the scope name 'profile' and a green 'Edited' button to its right. At the bottom of the interface, there is a green button with the text 'ADD MORE SCOPES'.

- User Mapping Provider: Configuration for how attributes returned from a OIDC provider are mapped onto a matrix user.
  - Localpart Template: Jinja2 template for the localpart of the MXID. Set it to `{{ user.upn.split('@')[0] }}` for AD FS

Other configurations are documented here.