

EMS Server With Custom Domain

For this guide, I will be using the domain element.io. I will set up EMS so that the Matrix usernames becomes `@someone:element.io`, and the Element client will be at <https://chat.element.io/>

From the guide at [Get Your Own EMS Server](#), I will be replacing the EMS hostname `ems-demo-staging.ems.host` with `element.ems.host`

Custom domains are only supported with Element Enterprise Cloud plans.

Prerequisites

- You own and control the domain you want to use
- If you do not have a website on the domain you want to use with your EMS server:
 - You can create a CNAME DNS record for the domain. Some providers call this ALIAS or CNAME Flattening when used on the root of the domain (domain root = `yourdomain.com`, not `something.yourdomain.com`)
- If you have a website on the domain you want to use with your EMS server:
 - Your website has HTTPS enabled using a valid certificate issued by a commonly recognized provider. For example, Comodo or LetsEncrypt.
 - You can serve plain-text JSON files at these exact paths
 - `https://yourdomain.com/.well-known/matrix/client`
 - `https://yourdomain.com/.well-known/matrix/server`
 - Note that these files do not and cannot have a file extension
 - You can add the header `Access-Control-Allow-Origin: *` to the client file on the web server

See also

- [FAQ: Can I use a subdomain instead of the root domain with my EMS server?](#)
- [FAQ: Can I use EMS-hosted well-knowns with the root of my domain?](#)

Setup

Some providers for DNS and website hosting providers need special configuration. See [Provider specific instructions](#) at the bottom for known solutions.

1. Follow steps 1 - 10 from [Get Your Own EMS Server](#)
2. On step 10 from [Get Your Own EMS Server](#), turn ON Custom DNS

Advanced options

These options are for advanced setups.

Custom DNS



3. In the Custom Homeserver domain field, enter element.io

Custom Homeserver domain

element.io

4. Create two files on your website according to the instructions given.
The path cannot be changed, but up to 30 redirects are supported.
While not required, you should add the header Content-Type application/json to both files.

1. https://element.io/.well-known/matrix/server

Well-Known documents

In order to prove that you own this domain and before you can use your hosted homeserver fully, you must complete domain verification.

Please create the following two .well-known documents:

```
GET https://element.io/.well-known/matrix/server
```

```
{
  "m.server": "element.ems.host:443"
}
```



Your custom **server homeserver Well-Known** document is not correctly set up.

Reason: Server well-known "m.server" key does not match element.ems.host:443

You can continue setting up your server. However, your custom homeserver federation will not work until Well-Known setup is complete.

[Check again](#)

```
{
  "m.server": "element.ems.host:443"
}
```

2. https://element.io/.well-known/matrix/client

```
GET https://element.io/.well-known/matrix/client
```

```
{
  "m.homeserver": {
    "base_url": "https://element.ems.host"
  },
  "m.identity_server": {
    "base_url": "https://vector.im"
  }
}
```

The `client` .well-known file needs the CORS header `Access-Control-Allow-Origin: *` enabled on your web server. Please see enable-cors.org for additional information.



Your custom `client` homeserver Well-Known document is not correctly set up.

Reason: Client well-known "m.homeserver base_url" does not match
`https://element.ems.host`

You can continue setting up your server. However, Element web clients will not be able to locate your custom homeserver using the domain `element.io` until client Well-Known setup is complete.

[Check again](#)

You need to enable the CORS header `Access-Control-Allow-Origin: *` on the web server for this file. See <https://enable-cors.org/> for instructions on how to do this. If you are using redirects, the CORS headers must be set on all steps/hops.

```
{
  "m.homeserver": {
    "base_url": "https://element.ems.host"
  },
  "m.identity_server": {
    "base_url": "https://vector.im"
  },
  "org.matrix.msc4143.rtc_foci": [
    {
```

```
    "type": "livekit",
    "livekit_service_url": "https://jwt.call.element.io"
  }
]
}
```

Optional Nginx-specific configuration

If your web server is running Nginx, you can set this in the Nginx config instead of creating actual files.

```
server {
    server_name element.io

    ...

    # Matrix well-known files
    location /.well-known/matrix/client {
        return 200
        '{"m.homeserver":{"base_url":"https://element.ems.host"},"m.identity_server":{"base_url":"https://vector.im"},"org.matrix.msc4143.rtc_foci":[{"type":"livekit","livekit_service_url":"https://jwt.call.element.io"}]}'
        add_header Content-Type application/json;
        add_header 'Access-Control-Allow-Origin' '*';
    }
    location /.well-known/matrix/server {
        return 200 '{"m.server": "element.ems.host:443"}';
        add_header Content-Type application/json;
    }
}
```

5. Click [Check again](#) to verify that your [.well-known](#) files are configured correctly

```
GET https://element.io/.well-known/matrix/server
```

```
{  
  "m.server": "element.ems.host:443"  
}
```



Great. Your **server** Well-Known document is set up correctly!

```
GET https://element.io/.well-known/matrix/client
```

```
{  
  "m.homeserver": {  
    "base_url": "https://element.ems.host"  
  },  
  "m.identity_server": {  
    "base_url": "https://vector.im"  
  }  
}
```

The **client** .well-known file needs the CORS header **Access-Control-Allow-Origin: *** enabled on your web server. Please see enable-cors.org for additional information.



Great. Your **client** Well-Known document is set up correctly!

You can also verify your `.well-known` files from the command line

Note the lines `access-control-allow-origin: *` and `content-type: application/json`

1. On Mac or Linux, using the `terminal`

```
$ curl -i https://element.io/.well-known/matrix/client  
HTTP/2 200  
date: Fri, 31 Jul 2020 09:11:21 GMT  
content-type: application/json  
content-length: 129  
set-cookie: __cfduid=x...; expires=Sun, 30-Aug-20 09:11:21 GMT; path=/;  
domain=.element.io; HttpOnly; SameSite=Lax
```

```
access-control-allow-origin: *
cf-cache-status: DYNAMIC
cf-request-id: 0...
expect-ct: max-age=604800, report-uri="https://report-uri.cloudflare.com/cdn-
cgi/beacon/expect-ct"
server: cloudflare
cf-ray: 5...
```

```
{
  "m.homeserver": {
    "base_url": "https://element.ems.host"
  },
  "m.identity_server": {
    "base_url": "https://vector.im"
  },
  "org.matrix.msc4143.rtc_foci": [
    {
      "type": "livekit",
      "livekit_service_url": "https://jwt.call.element.io"
    }
  ]
}
```

```
$ curl -i https://element.io/.well-known/matrix/server
HTTP/2 200
date: Fri, 31 Jul 2020 09:11:25 GMT
content-type: application/json
content-length: 52
set-cookie: __cfduid=x...; expires=Sun, 30-Aug-20 09:11:25 GMT; path=/;
domain=.element.io; HttpOnly; SameSite=Lax
access-control-allow-origin: *
cf-cache-status: DYNAMIC
cf-request-id: 0...
expect-ct: max-age=604800, report-uri="https://report-uri.cloudflare.com/cdn-
cgi/beacon/expect-ct"
server: cloudflare
```

cf-ray: 5...

```
{  
  "m.server": "element.ams.host:443"  
}
```

2. On Windows, using PowerShell

```
PS C:\Users\twilight> Invoke-WebRequest -Uri https://element.io/.well-known/matrix/client
```

StatusCode : 200

StatusDescription : OK

```
Content      : {  
  "m.homeserver": {  
    "base_url": "https://element.ams.host"  
  },  
  "m.identity_server": {  
    "base_url": "https://vector.im"  
  },  
  "org.matrix.msc4143.rtc_foci": [  
    {  
      "type": "livekit",  
      "livekit_service_url": "https://jwt.call.element.io"  
    }  
  ]  
}
```

RawContent : HTTP/1.1 200 OK

Connection: keep-alive

Access-Control-Allow-Origin: *

CF-Cache-Status: DYNAMIC

cf-request-id: 0...

Expect-CT: max-age=604800, report-uri="https://repor...

Forms : {}

Headers : {[Connection, keep-alive], [Access-Control-Allow-Origin, *], [CF-Cache-Status, DYNAMIC], [cf-request-id, 0...]}...

Images : {}

```
InputFields    : {}
Links          : {}
ParsedHtml     : System.__ComObject
RawContentLength : 129
```

```
PS C:\Users\twilight> Invoke-WebRequest -Uri https://element.io/.well-known/matrix/server
```

```
StatusCode      : 200
StatusDescription : OK
Content         : {
                  "m.server": "element.ems.host:443"
                }
RawContent      : HTTP/1.1 200 OK
                  Connection: keep-alive
                  Access-Control-Allow-Origin: *
                  CF-Cache-Status: DYNAMIC
                  cf-request-id: 0...
                  Expect-CT: max-age=604800, report-uri="https://repor...
Forms           : {}
Headers         : {[Connection, keep-alive], [Access-Control-Allow-Origin, *], [CF-Cache-Status, DYNAMIC], [cf-request-id, 0...]}
Images          : {}
InputFields     : {}
Links           : {}
ParsedHtml      : System.__ComObject
RawContentLength : 52
```

6. You can continue without the `.well-known` files in place, but your server will have limited functionality until this is fixed
7. In the `Custom Client domain` field, enter `chat.element.io`. This can be any domain, except the same as `Custom Homeserver domain`

Custom Client domain

chat.element.io

Custom client DNS will not be set if this field is left blank.

8. Create a CNAME DNS record with your DNS provider according to the instructions given `chat.element.io` CNAME `element.element.io`.

Custom Matrix Client CNAME Record

In order to prove that you own this domain and before you can use your (Element) Matrix client at this domain address, you must add a custom DNS record on your organisation's DNS servers.

This record will point your custom domain to your new hosted Element instance.

The CNAME record should have the following settings:

Alias	CNAME
chat.element.io.	element.element.io.

Note - The trailing '.' on the alias and CNAME (target) domains are important.

The full CNAME record (using the details from the table above) should be:

```
chat.element.io. CNAME element.element.io.
```



Your custom **Client DNS** record is not correctly set up.

You can continue setting up your server. However, your custom client DNS will not work until CNAME record setup is complete.

Once you have added your CNAME record, you will need to trigger a host rebuild (by clicking the 'Rebuild host' button from the host management page) in order to complete SSL certificate setup.

[Check again](#)

9. This shows how this is done with Cloudflare DNS. Depending on your DNS provider, this might be different. Consult the documentation for your provider. Note that Proxy must be turned off with Cloudflare.

DNS management for **element.io**

Search DNS Records



Search

Advanced

+ Add record

chat.element.io is an alias of **element.element.io**.

Type

CNAME

Name (required)

chat

Use @ for root

Target (required)

element.element.io

Proxy status



DNS only

TTL

Auto

Cancel

Save

10. Back on EMS, click [Check again](#). Note that sometimes it might take a while for your new DNS record to propagate. You can continue, but functionality will be limited. Check back with the Hosts tab on <https://ems.element.io/user/hosting> and click [Rebuild Host](#) once the DNS record is in place.



Great. Your domain CNAME record is set up correctly!

You can also verify the CNAME DNS record using the command line

1. On Mac or Linux, using the `terminal`

```
$ dig chat.element.io CNAME

; <<>> DiG 9.10.6 <<>> chat.element.io CNAME
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57888
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;chat.element.io.  IN CNAME

;; ANSWER SECTION:
chat.element.io. 299 IN CNAME element.element.io.

;; Query time: 32 msec
;; SERVER: 1.1.1.1#53(1.1.1.1)
;; WHEN: Fri Jul 31 10:21:56 BST 2020
;; MSG SIZE rcvd: 91
```

2. On Windows, using `PowerShell`

```
PS C:\Users\twilight> Resolve-DnsName -Name chat.element.io -Type CNAME
```

Name	Type	TTL	Section	NameHost
chat.element.io	CNAME	299	Answer	element.element.io

11. Continue from step 11 on [Get Your Own EMS Server](#)

Provider-specific instructions

GitHub Pages

If you are hosting your website with GitHub Pages, add this to the Jekyll config file `_config.yml`

```
include:  
  - .well-known
```

Microsoft Azure

If you are using Microsoft 365 / Azure to manage your domain or the rest of your infrastructure, please use the following instructions to host the `.well-known/matrix` URI (RFC 8615).

In this section, we will configure Azure to serve `https://yourdomain.com/.well-known/matrix/client` and `https://yourdomain.com/.well-known/matrix/client`.

The summary of steps is as follows:

- prepare json files
- create a Storage account then enable a Static website
- upload the json files to `.well-known/matrix/` in the `$web` container
- create a CDN and an endpoint for this container
- create a `CNAME` DNS entry for your custom domain, pointing to your CDN endpoint
- associate your custom domain to your CDN endpoint

Prepare client and server `.well-known` files locally

On your computer, prepare two plain text files called `client` and `server` (again, notice the lack of file extension such as `.txt`) which contain the following:

client

```
{  
  "m.homeserver": {  
    "base_url": "https://your-tenant.ams.host"  
  },  
}
```

```
"m.identity_server": {
  "base_url": "https://vector.im"
},
"org.matrix.msc4143.rtc_foci": [
  {
    "type": "livekit",
    "livekit_service_url": "https://jwt.call.your-tenant.io"
  }
]
}
```

Remember to replace `your-tenant` by the name of your EMS tenant.

server

```
{
  "m.homeserver": {
    "base_url": "https://your-tenant.ems.host:443"
  }
}
```

Remember to replace `your-tenant` by the name of your EMS tenant.

You will be uploading these shortly.

Create a Storage account and Static website

Storage account

In the Azure Portal, [create a Storage account](#).



Create a storage account ...

Basics[Advanced](#)[Networking](#)[Data protection](#)[Encryption](#)[Tags](#)[Review](#)

Azure Storage is a Microsoft-managed service providing cloud storage that is highly available, secure, durable, scalable, and redundant. Azure Storage includes Azure Blobs (objects), Azure Data Lake Storage Gen2, Azure Files, Azure Queues, and Azure Tables. The cost of your storage account depends on the usage and the options you choose below. [Learn more about Azure storage accounts](#)

Project details

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription *

Azure subscription 1



Resource group *

well-known_group

[Create new](#)

Instance details

Storage account name ⓘ *

rfc8615demo

Region ⓘ *

(Europe) UK West

[Deploy to an edge zone](#)

Performance ⓘ *

**Standard:** Recommended for most scenarios (general-purpose v2 account)**Premium:** Recommended for scenarios that require low latency.

Redundancy ⓘ *

Geo-redundant storage (GRS)



Make read access to data available in the event of regional unavailability.

The name needs to be unique to Azure. `yourdomainwellknown` is an option that should work well in most scenarios.

Performance can be left to Standard.

Redundancy should be set to Geo-redundant storage (GRS) as the .well-known URI will be a core part of your EMS deployment.

You can leave all other options to their default or change them to fit your specific deployment scenario.

Finally, click create.

Static website

Once the Storage account is created, you will need to create a Static website in this Storage account. In the Storage account overview, choose "Static website", in the Data management section.

Home > rfc8615demo_1699629239459 | Overview >

rfc8615demo

Storage account

Search

«

Overview

Activity log

Tags

Diagnose and solve problems

Access Control (IAM)

Data migration

Events

Storage browser

Storage Mover

Data storage

Containers

File shares

Queues

Tables

Security + networking

Networking

Front Door and CDN

Access keys

Shared access signature

Encryption

Microsoft Defender for Cloud

Data management

Redundancy

Data protection

Object replication

Blob inventory

Static website

Lifecycle management

Azure search

Upload

Open in Explorer

Delete

Move

Refresh

Open in mobile

CLI / PS

Essentials

Resource group (move) : [well-known_group](#)

Location : ukwest

Primary/Secondary Location : Primary: UK West, Secondary: UK South

Subscription (move) : [Azure subscription 1](#)

Subscription ID : 3d4470a7-1fd3-4bae-b372-5b7fb11daa70

Disk state : Primary: Available, Secondary: Available

Tags (edit) : [Add tags](#)

Properties

Monitoring

Capabilities (7)

Recommendations (0)

Tutorials

Tools + SDKs

Blob service

Hierarchical namespace : Disabled

Default access tier : Hot

Blob anonymous access : Disabled

Blob soft delete : Enabled (7 days)

Container soft delete : Enabled (7 days)

Versioning : Disabled

Change feed : Disabled

NFS v3 : Disabled

Allow cross-tenant replication : Disabled

File service

Large file share : Disabled

Identity-based access : Not configured

Default share-level permissions : Disabled

Soft delete : Enabled (7 days)

Share capacity : 5 TiB

Queue service

CMK support : Disabled

Table service

CMK support : Disabled

You do not need to provide a specific Index document name or Error document path.

Enabling the Static website in your Storage account will automatically create a `$web` storage container to which you can upload the json text files which will be served at the .well-known URI.

Go to "Containers", in the "Data storage" section, to upload the `client` and `server` files you prepared earlier.

Click on the `$web` container, then chose "Upload", which will open a panel on the right.

When uploading the `client` and `server` files, make sure to open the "advanced" part of the upload panel and choose to upload to a specific folder: `.well-known/matrix/`

Upload blob



2 file(s) selected: server, client

Drag and drop files here or [Browse for files](#)

☐ Overwrite if files already exist

Advanced

Blob type ⓘ

Block blob

☒ Upload .vhd files as page blobs (recommended)

Block size ⓘ

4 MiB

Access tier ⓘ

Hot (Inferred)

Upload to folder

.well-known/matrix/

Blob index tags ⓘ

Key

Value

Encryption scope

☒ Use existing default container scope

☐ Choose an existing scope

Retention policy ⓘ

☐ No retention

☐ Choose custom retention period

Upload

Give feedback

Create a CDN, `CNAME` DNS entry for your custom domain and Custom domain name for the CDN endpoint

Creating a CDN is needed because Azure does not allow serving HTTPS over a custom domain using only a Storage account Static website. To do so, a CDN is necessary.

Create a CDN endpoint

Go back to your Storage account's main view and choose "Front Door and CDN" in the Security + Networking section, to create a CDN endpoint.

Home > rfc8615demo_1699629239459 | Overview > rfc8615demo

rfc8615demo | Front Door and CDN ☆ ...
Storage account

Search << Azure Front Door and Azure CDN are CDN (Content Delivery Network) services designed to send audio, video, images, and other files faster and more reliably.

- Overview
- Activity log
- Tags
- Diagnose and solve problems
- Access Control (IAM)
- Data migration
- Events
- Storage browser
- Storage Mover

Data storage

- Containers
- File shares
- Queues
- Tables

Security + networking

- Networking
- Front Door and CDN**
- Access keys
- Shared access signature
- Encryption
- Microsoft Defender for Cloud

Endpoints

Click on your endpoint below to manage and configure different features of your existing CDN endpoints. [Learn more about adding custom domains](#)

Host name	Profile name	Service type
-----------	--------------	--------------

New Endpoint

Service type

☐ Azure Front Door (Recommended)

☒ Azure CDN

Create new/use existing profile

☒ Create new

☐ Use existing

Profile name *

rfc8615demo

Endpoint name *

rfc8615demo

Endpoint host name

rfc8615demo.azureedge.net

Origin host name *

rfc8615demo.z35.web.core.windows.net (Static website) ▾

Pricing tier *

Standard Microsoft ▾

[Learn more](#)

Query string caching behavior *

▾

Service type: Azure CDN is sufficient, the more advanced features of Azure Front Door are not necessary here.

If you need to create a new profile, you may call the Profile name and Endpoint name as you wish. For consistency, it is suggested to call them `yourdomainwellknown` as previously.

For Origin host name, pick your Static website.

For Query string caching behaviour, pick Ignore Query String, although this setting is not important in our context.

Once your endpoint is deployed, go to the resource, then go to your endpoint's further settings by clicking on it. Make note of its hostname (`rfc8615demo.azureedge.net` in our example) as you will need it shortly.

Home >

rfc8615demo Front Door and CDN profile

Search << + Endpoint Purge Move Delete

Overview

- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Settings

- Properties
- Quickstart
- Identity
- Locks

Monitoring

- Alerts
- Metrics
- Diagnostic settings
- Logs

Essentials

Resource group (move) : [well-known_group](#)

Status : Active

Subscription (move) : [Azure subscription 1](#)

Subscription ID : 3d4470a7-1fd3-4bae-b372-5b7fb11daa70

Endpoints

Hostname	↑↓ Status	↑↓ Protocol
rfc8615demo.azureedge.net	Running	HTTP, HTTPS

Create CNAME DNS entry for your custom domain

You will now create a DNS entry for your custom domain, a CNAME pointing to your Azure CDN endpoint's hostname. How to do so exactly will depend on who hosts your DNS servers. The specifics of this are beyond this documentation, but the following general information should be sufficient.

In your DNS provider's admin panel, add a DNS entry with the following details:

- Type: CNAME
- Domain: your custom domain, such as yourdomain.com
- Target: your Azure CDN Endpoint's hostname, such as rfc8615demo.azureedge.net in our example

Once created, this DNS entry may take some time to propagate, but in most cases will be picked up quickly by Azure, as needed in the next step.

Associate your custom domain with your CDN Endpoint

In the endpoint's settings, choose "+ custom domain" to start adding your custom domain. A panel will open on the right.

Enter your custom domain: yourdomain.com and finally, click add.

If the CNAME entry can be seen by Azure, after a few minutes your custom domain should be associated with your CDN Endpoint.

Final result

https://yourdomain.com/.well-known/matrix/client and https://yourdomain.com/.well-known/matrix/server are now served over HTTPS by Azure.

You should now have the following resources in your Azure account:

- Storage account
- Front Door and CDN Profile
- Endpoint

Resources

- <https://learn.microsoft.com/en-us/azure/cdn/cdn-create-a-storage-account-with-cdn>
- <https://learn.microsoft.com/en-us/azure/cdn/cdn-storage-custom-domain-https>

Revision #24

Created 18 April 2022 14:19:41 by Karl Abbott

Updated 27 March 2025 14:48:58 by twilight