

Services



- [Adding a Public Service Export](#)
- [Importing a Service](#)
 - [Pushing the changes to the nats servers](#)
 - [Testing the Service](#)
- [Securing Services](#)
 - [Creating a Private Service Export](#)
 - [Generating an Activation Token](#)
- [Importing a Private Service](#)
 - [Testing the Private Service](#)

To share services that other accounts can reach via request reply, you have to *Export* a *Service*. *Services* are associated with the account performing the replies and are advertised in the exporting accounts' JWT.

Adding a Public Service Export

To add a service to your account:

```
nsc add export --name help --subject help --service
```

```
[ OK ] added public service export "help"
```

To review the service export:

```
nsc describe account
```

Account Details	
-----------------	--

Name	A
Account ID	ADETPT36WBIBUKM3IBCVM4A5YUSDXFEJPW4M6GGVBYCBW7RRNFTV5NGE
Issuer ID	0AFEEYZSYYVI4FXLRXJTM32PQEI3RG0WZJT7Y3YFM4HB7ACPE4RTJPG
Issued	2019-12-04 18:20:42 UTC
Expires	
Max Connections	Unlimited
Max Leaf Node Connections	Unlimited
Max Data	Unlimited
Max Exports	Unlimited
Max Imports	Unlimited
Max Msg Payload	Unlimited
Max Subscriptions	Unlimited
Exports Allows Wildcards	True
Imports	None

Exports					
Name	Type	Subject	Public	Revocations	Tracking
help	Service	help	Yes	0	-

Importing a Service

Importing a service enables you to send requests to the remote *Account*. To import a Service, you have to create an *Import*. To create an import you need to know:

- The exporting account's public key
- The subject the service is listening on
- You can map the service's subject to a different subject
- Self-imports are not valid; you can only import services from other accounts.

To learn how to inspect a JWT from an account server, [check this article](#).

First let's create a second account to import the service into:

```
nsc add account B
```

```
[ OK ] generated and stored account key
"AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H"
[ OK ] added account "B"
```

Add the import of the subject 'help'

```
nsc add import --src-account ADETPT36WBIBUKM3IBCV4A5YUSDJFEJPW4M6GGVBYCBW7RRNFTV5NGE --
remote-subject help --service
```

```
[ OK ] added service import "help"
```

Verifying our work:

```
nsc describe account
```

Account Details	
Name	B
Account ID	AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H
Issuer ID	0AFEEYZSYYVI4FXLRXJTMM32PQEI3RG0WZJT7Y3YFM4HB7ACPE4RTJPG
Issued	2019-12-04 20:12:42 UTC
Expires	
Max Connections	Unlimited
Max Leaf Node Connections	Unlimited
Max Data	Unlimited
Max Exports	Unlimited
Max Imports	Unlimited
Max Msg Payload	Unlimited
Max Subscriptions	Unlimited
Exports Allows Wildcards	True
Exports	None
Imports	

Name	Type	Remote	Local/Prefix	Expires	From Account	Public
help	Service	help	help		A	Yes

Let's also add a user to make requests from the service:

```
nsc add user b
```

```
[ OK ] generated and stored user key
"UDKNTNEL5YD66U2FZZ2B3WX2PLJFKEFHAPJ3NWJBFF44PT76Y2RAVFVE"
[ OK ] generated user creds file "~/.nkeys/creds/0/B/b.creds"
[ OK ] added user "b" to account "B"
```

Pushing the changes to the nats servers

If your nats servers are configured to use the built-in NATS resolver, remember that you need to 'push' any account changes you may have done (locally) using `nsc add` to the servers for those changes to take effect.

e.g. `nsc push -i` or `nsc push -a B -u nats://localhost`

Testing the Service

To test the service, we can install the ['nats'](#) CLI tool:

Set up a process to handle the request. This process will run from account 'A' using user 'U':

```
nats reply --creds ~/.nkeys/creds/0/A/U.creds help "I will help"
```

Remember you can also do:

```
nsc reply --account A --user U help "I will help"
```

Send the request:

```
nats request --creds ~/.nkeys/creds/0/B/b.creds help me
```

The service receives the request:

```
Received on [help]: 'me'
```

And the response is received by the requestor:

```
Received [_INBOX.v6KAX0v1bu87k49hbg3dgn.StIGJF0D] : 'I will help'
```

Or more simply:

```
nsc reply --account A --user U help "I will help"
nsc req --account B --user b help me
```

```
published request: [help] : 'me'
received reply: [_INBOX.GCJltVq1wRSb5FoJrJ6SE9.w8utbBXR] : 'I will help'
```

Securing Services

If you want to create a service that is only accessible to accounts you designate you can create a *private* service. The export will be visible in your account, but subscribing accounts will require an authorization token that must be created by you and generated specifically for the requesting account. The authorization token is simply a JWT signed by your account where you authorize the client account to import your service.

Creating a Private Service Export

```
nsc add export --subject "private.help.*" --private --service --account A
```

```
[ OK ] added private service export "private.help.*"
```

As before, we declared an export, but this time we added the `--private` flag. The other thing to note is that the subject for the request has a wildcard. This enables the account to map specific subjects to specifically authorized accounts.

```
nsc describe account A
```

Account Details	
Name	A

Account ID	ADETP36WBIBUKM3IBCVM4A5YUSDXFEJPW4M6GGVBYCBW7RRNFTV5NGE
Issuer ID	0AFEEYZSYYVI4FXLRXJTM32PQEI3RG0WZJT7Y3YFM4HB7ACPE4RTJPG
Issued	2019-12-04 20:19:19 UTC
Expires	
Max Connections	Unlimited
Max Leaf Node Connections	Unlimited
Max Data	Unlimited
Max Exports	Unlimited
Max Imports	Unlimited
Max Msg Payload	Unlimited
Max Subscriptions	Unlimited
Exports Allows Wildcards	True
Imports	None

Exports					
Name	Type	Subject	Public	Revocations	Tracking
help	Service	help	Yes	0	-
private.help.*	Service	private.help.*	No	0	-

Generating an Activation Token

For the foreign account to *import* a private service and be able to send requests, you have to generate an activation token. The activation token in addition to granting permission to the account allows you to subset the service's subject:

To generate a token, you'll need to know the public key of the account importing the service. We can easily find the public key for account B by running:

```
nsc list keys --account B
```

Keys			

Entity	Key	Signing Key	Stored
0	0AFEYYZSYVVI4FXLRXJTM32PQEI3RGOWZJT7Y3YFM4HB7ACPE4RTJPG		*
B	AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H		*
b	UDKNTNEL5YD66U2FZZ2B3WX2PLJFKEFHAPJ3NWJBFF44PT76Y2RAVFVE		*

```
nsc generate activation --account A --target-account
AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H --subject
private.help.AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H -o /tmp/activation.jwt
```

```
[ OK ] generated "private.help.*" activation for account
"AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H"
[ OK ] wrote account description to "/tmp/activation.jwt"
```

The command took the account that has the export ('A'), the public key of account B, the subject where requests from account B will be handled, and an output file where the token can be stored. The subject for the export allows the service to handle all requests coming in on private.help.*, but account B can only request from a specific subject.

For completeness, the contents of the JWT file looks like this:

```
cat /tmp/activation.jwt
```

```
-----BEGIN NATS ACTIVATION JWT-----
eyJ0eXAiOiJqd3QiLCJhbGciOiJlZDI1NTE5In0.eyJqdGkiOiJUS01LNEFHT1p0VERDTERGUk9QTl1NM0hHUVRDTEJTUK
tNQXxXWTVSUUhFVEVNNE1VTDdB1iwiawWF0IjoXNTc1NDkxNjEwLCJpc3MiOiJBREVVUUFQZnldCSUJVS00zSUJDVWk00QTVZ
VVNEWEZFS1BXNE02R0dWQlLDQlc3U1J0RlRWNU5HRSIIm5hbWUiOiJwcm12YXRlLmhlbHAuQUFNNDZFM1lGNVdPWlNFNV
d0WVdITjNZWU1TVlpuPU0k2WEHURjJRNjRFQ1BYU0ZRWlJPSk1QMkgiLCJzdWIiOiJBQU00NkUzWUY1V09aU0U1V05ZV0h0
M1lZSVNWWk9TSTZYSFRGMlE2NEVDUFhTRlFaUk9KTVAySCIsInR5cGUiOiJhY3RpdmF0aW9uIiwibmF0cyI6eyJzdWJqZW
N0IjoicHJpdmF0ZS5oZWxwLkFBTTQ2RTNZRjVXT1pTRTVXLTlXSE4zWVlJU1ZaT1NJNlhIVEYyUTY0RUNQWFNGUVpST0pN
UDJIIiwidHlwZSI6InNlcnZpY2UifX0.4tFx_1UzPUwbV8wFNIJsQYu91K9hZaGRLE10n0phfHGetvMPv1384KC-
1AiNdhAp0bSDFosdDcpjryD0QxaDCQ
-----END NATS ACTIVATION JWT-----
```

When decoded it looks like this:

```
nsc describe jwt -f /tmp/activation.jwt
```

```
Activation
```

Name	private.help.AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H
Account ID	AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H
Issuer ID	ADETPT36WBIBUKM3IBCVM4A5YUSDXFEJPW4M6GGVBYCBW7RRNFTV5NGE
Issued	2019-12-04 20:33:30 UTC
Expires	
Hash ID	DD6BZKI2LTQKAJYD5GTSI40FUG72KD2BF74NFVLUN047PR40X64Q=====
Import Type	Service
Import Subject	private.help.AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H
Max Messages	Unlimited
Max Msg Payload	Unlimited
Network Src	Any
Time	Any

The token can be shared directly with the client account.

“ If you manage many tokens for many accounts, you may want to host activation tokens on a web server and share the URL with the account. The benefit to the hosted approach is that any updates to the token would be available to the importing account whenever their account is updated, provided the URL you host them in is stable. When using a JWT account server, the tokens can be stored right on the server and shared by an URL that is printed when the token is generated.

Importing a Private Service

Importing a private service is more natural than a public one because the activation token stores all the necessary details. Again, the token can be an actual file path or a remote URL.

```
nsc add import --account B -u /tmp/activation.jwt --local-subject private.help --name
private.help
```

```
[ OK ] added service import
"private.help.AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H"
```

Describe account B

nsc describe account B

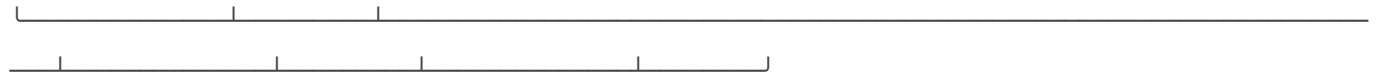
Account Details			
Name	B		
Account ID	AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H		
Issuer ID	0AFEEYZSYYVI4FXLRXJTMM32PQEI3RGOWZJT7Y3YFM4HB7ACPE4RTJPG		
Issued	2019-12-04 20:38:06 UTC		
Expires			
Max Connections	Unlimited		
Max Leaf Node Connections	Unlimited		
Max Data	Unlimited		
Max Exports	Unlimited		
Max Imports	Unlimited		
Max Msg Payload	Unlimited		
Max Subscriptions	Unlimited		
Exports Allows Wildcards	True		
Exports	None		

Imports

</

```
private.help.AAM46E3YF5W0ZSE5WNYWHN3YYISVZ0SI6XHTF2Q64ECPXSFQZR0JMP2H | private.help
```

```
| | A | No |
```



When importing a service, you can specify the local subject you want to use to make requests. The local subject in this case is `private.help`. However when the request is forwarded by NATS, the request is sent on the remote subject.

Testing the Private Service

Testing a private service is no different than a public one:

```
nsc reply --account A --user U "private.help.*" "help is here"
```

```
nsc req --account B --user b private.help help_me
```

```
published request: [private.help] : 'help_me'
```

```
received reply: [_INBOX.3MhS0iCHfq08wU11x59bHB.jpE2jvEj] : 'help is here'
```

#1

□ cocobook-sync ## 2026-10-28 11:10:52 UTC

□ cocobook-sync ## 2026-10-28 11:10:52 UTC