

Mixed Authentication/Authorization Setup

Mixing both [nkeys](#) static config and [decentralized JWT Authentication/Authorization](#) is possible but needs some preparation in order to be able to do it.

The way this can be done is by **first** preparing a basic trusted operator setup that could be used in the future, and then base from that configuration to create the NKEYS static config using the same shared public nkeys for the accounts and then use clustering routes to bridge the two different auth setups during the transition.

For example, creating the following initial setup using [NSC](#):

```
nsc add account --name SYS
nsc add user --name sys
nsc add account --name A
nsc add user -a A --name test
nsc add account --name B
nsc add user -a B --name test
```

This will then generate something like the following:

```
nsc list accounts
```

Accounts	
Name	Public Key
A	ADFB2JXYTX0JEL6LNAXDREUGRX35B0LZI3B4PFFAC7IRPR30A4QNKBN2
B	ACWOMQA7PZTKJSBTR7BF6TBK3D776734PWHWDK07HFMQ0M5BI0YPSYZZ
SYS	ABKOWIYVTHNEK5HELPLAT2CF2CUPELIK4SZH2VCJHLFU22B5U2IIZU0

```
nsc list users -a A
```

Users	
Name	Public Key
test	UAP0K2P7EN3UFBL7SBJPQK3M3JMLALYRYKX5XWSVMVYK63ZMBHT0HVJR

We could use this configuration as the initial starting configuration for an nkeys config now, where all the NKEYS users public nkeys are explicitly listed (centralized auth model).

```
port = 4222

cluster {
  port = 6222

  # We will bridge two different servers with different auth models via routes
  # routes [ nats://127.0.0.1:6223 ]
}

system_account = ABK0WIYVTHNEK5HELPWLAT2CF2CUPELIK4SZH2VCJHLFU22B5U2IIZU0

accounts {
  # Account A
  ADFB2JXYTX0JEL6LNAXDREUGRX35B0LZI3B4PFFAC7IRPR30A4QNKBN2 {
    nkey: ADFB2JXYTX0JEL6LNAXDREUGRX35B0LZI3B4PFFAC7IRPR30A4QNKBN2
    users = [
      {nkey: "UAP0K2P7EN3UFBL7SBJPQK3M3JMLALYRYKX5XWSVMVYK63ZMBHT0HVJR" }
    ]
  }

  # Account B
  ACW0MQA7PZTKJSBTR7BF6TBK3D776734PWHWDK07HFMQ0M5BI0YPSYZZ {

  }

  # Account SYS
  ABK0WIYVTHNEK5HELPWLAT2CF2CUPELIK4SZH2VCJHLFU22B5U2IIZU0 {
  }
}
```

By using `nsc` it is possible to create a mem based resolver for the trusted operator setup:

```
nsc generate config --mem-resolver --sys-account SYS
```

An example configuration from the second node with the trusted operator setup could then be:

```
port = 4223

cluster {
    port = 6223
    routes [ nats://127.0.0.1:6222 ]
}

# debug = true
# trace = true

# Operator
operator = eyJ0eXAiOiJqd3QiLCJhbGciOiJlZDI1NTE5In0.eyJqdGkiOiJQNDJBScFTVVA0TUdGRU1EQzVCRVVGUkM1MLQ1M05OTzRIWkhRNEdETVkc0S0RZTFVRV0JBIiwiaWF0IjoxNTc0Mzc1OTE2LCJpc3MiOiJPQ09KSks5aSUNINKNHUU5LM1NRMzMVXTFpXWkpDUkRBtFJIWjZPVzQ0RFpZVVdNVVYzVlBSSEZSRClSm5hbWUiOiJLTyIsInN1YiI6Ik9DT0pKTlpJQ0g2Q0dRTkszu1EzNVdMWldasKnsSREFMUKhaNk9XNDREWllVv01VVjNXUFJIRlJEIiwidHlwZSI6Im9wZXJhdG9yIiwibmF0cyI6e3l9.pppa9-xhWXJLSCCtqj_dqlvXKR7WlVCh0cqoZ6lr8zg3WlWM8U0bnf6FHw_67-wRS7jj0n4PuA0P0MAJdE3pDA

system_account = ABKOWIYVTHNEK5HELPWLAT2CF2CUPELIK4SZH2VCJHLFU22B5U2IIIZUO

resolver = MEMORY

resolver_preload = {
    # Account "A"
    ADFB2JXYTX0JEL6LNAXDREUGRX35B0LZI3B4PFFAC7IRPR30A4QNKBN2:
    eyJ0eXAiOiJqd3QiLCJhbGciOiJlZDI1NTE5In0.eyJqdGkiOiIyQjNYNUJPQzQ3M05TU0hfWUVHTzJYRUhaTVNGRVB0RFBREJXWkJLRvdQVlg2TUlZU1JRIiwiaWF0IjoxNTc0Mzc1OTE2LCJpc3MiOiJPQ09KSks5aSUNINKNHUU5LM1NRMzMVXTFpXWkpDUkRBtFJIWjZPVzQ0RFpZVVdNVVYzVlBSSEZSRClSm5hbWUiOiJBIiwic3ViIjoIQURGQjJKWFlUWE9KRUw2TE5BWERSRVUHulgZNUJPTFPjJM0I0UEZGQU3SVJQUjNPQTRRTktCTjIiLCJ0eXBFIjoiyWNjb3VudCIsm5hdHMiOnsiZXhwbm3J0cyI6W3sibmFtZSI6InRlc3QiLCJzdWJqZWNOIjoidGVzdCIsm5R5cGUoiOiJzZXJ2aWNlIiwic2VydmVjZV9sYXRlbmN5Ijp7InNhbmhXSaW5nIjoxMDAsInRlc3VsdaHMiOiJsYXRlbmN5Lm9uLnRlc3QifX1dLCJsaWpdHMiOnsic3VicyI6LTEsm5bm4iOi0xLCJsZWFiIjotMSwiaWlwb3J0cyI6LTEsm5V4cG9ydHMiOi0xLCJkYXRhIjotMSwicGF5bG9hZCI6LTEsm5IndpbG
```

```
RjYXJkcyI6dHJlZX19fQ.zZBetgDN6nCFDVPwzF_124BPkc8amGPDnr0miKUa12xski5zskUI0Y00eIa1vTo0bkHIKTgM2
QDYpmXUQ0HnAQ
```

```
# Account "B"
```

```
ACW0MQA7PZTKJSBTR7BF6TBK3D776734PWHWDK07HFMQOM5BIOYPSYZZ:
```

```
eyJ0eXAiOiJqd3QiLCJhbGciOiJlZDI1NTE5In0.eyJqdGkiOiJBTfNFQkZGWDZMR0pQTLVMU1NXTDNTRTNISkk2WUZSWl
VKSDNLV0E1VE41WUtWRE5MVTJRIiwiawF0IjoXNTc0Mzc10TE2LCJpc3MiOiJJPQ09KSk5aSUNINKNHUU5LM1NRMzVXTFpX
WkpDUKRBTfJJIWjZPVzQ0RFpZVVdNVVYzV1BSSEZSRCIsIm5hbWUiOiJCIiwic3ViIjoIQUXNT01RQTdQWlRSLNCFVF13Qk
Y2VEJLM0Q3NzY3MzRQV0hXREtPN0hGTVFPTTVCSU9ZUFNZWloiLCJ0eXB1IjoIYWNjb3VudCIsIm5hdHMiOnsiaW1wb3J0
cyI6W3sibmFtZSI6InRlc3QiLCJzdWJqZWNoIjoIdGVzdCIsImFjY291bnQiOiJBREZCMkpYWVRYT0pFTDZMTkFYRFJFVU
dSWDM1Qk9MwkkzQjRQRkZBQzdJUlBSM09BNFF0S0J0MiIsInRvIjoIdGVzdCIsInR5cGUiOiJzZXJ2aW1lIn1dLCJsaW1p
dHMiOnsic3VicyI6LTesImNvb4iOi0xLCJsZWFiIjoIMSwiaW1wb3J0cyI6LTesImV4cG9ydHMiOi0xLCJkYXRhIjoIMSw
wicGF5bG9hZCI6LTesIndpbGRjYXJkcyI6dHJlZX19fQ.AnzziBwt5Tnphc2pr0NUU0pMpkAlJHvCPaag0GUtTYPCHKDp
hcJrwtAHi4v5N0I6npjoes0F0MlrfnHqidDAg
```

```
# Account "SYS"
```

```
ABK0WIYVTHNEK5HELPWLAT2CF2CUPELIK4SZH2VCJHLFU22B5U2IIZU0:
```

```
eyJ0eXAiOiJqd3QiLCJhbGciOiJlZDI1NTE5In0.eyJqdGkiOiI1WVUyWkc1UkRTSU1TN1pGVE1MU0NZQUtLVkVFWUpPUL
c0TDJPTLY3N1g1TLJZWkFGSkRRRIiwiawF0IjoXNTc0Mzc10TE2LCJpc3MiOiJJPQ09KSk5aSUNINKNHUU5LM1NRMzVXTFpX
WkpDUKRBTfJJIWjZPVzQ0RFpZVVdNVVYzV1BSSEZSRCIsIm5hbWUiOiJTWVMiLCJzdWUiOiJBQktPV0lZVlRITkVLNUhFTF
BXTEFUMkNGMkNVUEVMSUs0U1pIMlZDSkhMRlUyMkI1VTJJSVpVTyIsInR5cGUiOiJhY2NvdW50IiwibmF0cyI6eyJsaW1p
dHMiOnsic3VicyI6LTesImNvb4iOi0xLCJsZWFiIjoIMSwiaW1wb3J0cyI6LTesImV4cG9ydHMiOi0xLCJkYXRhIjoIMSw
wicGF5bG9hZCI6LTesIndpbGRjYXJkcyI6dHJlZX19fQ.5Fr04sZbWuFgRLuy7c1eQLUq_BQ4PNhIAN5A-
sRLkYWmvlc4c_Y4VfTbgl5zhNzCxfvj9SxT7ySgphup2BiRAA
}
```

Even though they have different authorization mechanisms, these two servers are able to route account messages because they share the same NKEY.

We have created at least one user, in this case with creds:

```
-----BEGIN NATS USER JWT-----
```

```
eyJ0eXAiOiJqd3QiLCJhbGciOiJlZDI1NTE5In0.eyJqdGkiOiJNRkM3V1E1N0hKUE9aWUV0TEhVRTZTWVFVQTDVKTURWSk
xIQzJRTkpYNUVJS0RGR1U1REhRIiwiawF0IjoXNTc0Mzc10TE2LCJpc3MiOiJBREZCMkpYWVRYT0pFTDZMTkFYRFJFVUdS
WDM1Qk9MwkkzQjRQRkZBQzdJUlBSM09BNFF0S0J0MiIsIm5hbWUiOiJ0ZXN0Iiwic3ViIjoIQUVFT0syUDdFTjNVRk1JMN1
NCSlBR5zNM0pNTEFMWVJZS1g1WfdTVk1WWUs2M1pNqkhUT0hWSLIiLCJ0eXB1IjoIdXNlciIsIm5hdHMiOnsic3VicyI6LTesImNvb4iOi0xLCJsZWFiIjoIMSwiaW1wb3J0cyI6LTesImV4cG9ydHMiOi0xLCJkYXRhIjoIMSw
wicGF5bG9hZCI6LTesIndpbGRjYXJkcyI6dHJlZX19fQ.MSU2aUIBK1iUsg7h52LLrEfTwVMF_wB3HDq75ECskxSyyDDMTk9_3957UtQF-
3yoGCIhK0kwjzX8C-WXnLADw
```

```
-----END NATS USER JWT-----
```

```
***** IMPORTANT *****  
NKEY Seed printed below can be used to sign and prove identity.  
NKEYs are sensitive and should be treated as secrets.  
-----BEGIN USER NKEY SEED-----  
SUANVBWRHHFMGHNIT6UJHPN2TGVBIILE7VPVNEQ7DGCJ26ZD2V3KAHT4M  
-----END USER NKEY SEED-----  
*****
```

And this same user is able to connect to either one of the servers (bound to 4222 and 4223 respectively):

Subscriber Service:

```
package main  
  
import (  
    "log"  
  
    "github.com/nats-io/nats.go"  
)  
  
func main() {  
    opts := make([]nats.Option, 0)  
  
    // Extract public nkey from seed  
    //  
    // Public:  UAP0K2P7EN3UFBL7SBJPQK3M3JMLALYRYKX5XWSVMVYK63ZMBHT0HVJR  
    // Private: SUANVBWRHHFMGHNIT6UJHPN2TGVBIILE7VPVNEQ7DGCJ26ZD2V3KAHT4M  
    //  
    nkey, err := nats.NkeyOptionFromSeed("path/to/seed.nkey")  
    if err != nil {  
        log.Fatal(err)  
    }  
    opts = append(opts, nkey)  
    nc, err := nats.Connect("127.0.0.1:4222", opts...)  
    if err != nil {  
        log.Fatal(err)  
    }  
    nc.Subscribe("test", func(m *nats.Msg){  
        log.Printf("[Received] %q, replying... \n", string(m.Data))  
        m.Respond([]byte("pong from nkeys based server"))  
    })  
}
```

```
    })

    select {}

}
```

Requestor:

```
package main

import (
    "log"
    "time"

    "github.com/nats-io/nats.go"
)

func main() {
    nc, err := nats.Connect("127.0.0.1:4223", nats.UserCredentials("path/to/user.creds"))
    if err != nil {
        log.Fatal(err)
    }

    for range time.NewTicker(1 * time.Second).C {
        resp, err := nc.Request("test", []byte("test"), 1*time.Second)
        if err != nil {
            log.Println("[Error]", err)
            continue
        }
        log.Println("[Received]", string(resp.Data))
    }
}
```

👤 #3

📄 cocobook-sync 👤 2026-10-28 11:14:12 UTC
📄 cocobook-sync 👤 2026-10-29 01:42:39 UTC