

# Configuration

---



- [leafnodes Configuration Block](#)
  - [TLS Block](#)
    - [Accepting side](#)
    - [Remote side](#)
  - [Authorization Block](#)
    - [Users Block](#)
  - [LeafNode remotes Entry Block](#)
    - [Signature Handler](#)
    - [Connecting using WebSocket protocol](#)
    - [tls Configuration Block](#)
- 

## leafnodes Configuration Block

The leaf node configuration block is used to configure incoming as well as outgoing leaf node connections. Most properties are for the configuration of incoming connections. The properties `remotes` and `reconnect` are for outgoing connections.

| Property               | Description                                                                                                                                             |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>host</code>      | Interface where the server will listen for incoming leafnode connections.                                                                               |
| <code>port</code>      | Port where the server will listen for incoming leafnode connections (default is 7422).                                                                  |
| <code>listen</code>    | Combines <code>host</code> and <code>port</code> as <code>&lt;host&gt;:&lt;port&gt;</code>                                                              |
| <code>tls</code>       | TLS configuration block (same as other nats-server <a href="#">tls configuration</a> ).                                                                 |
| <code>advertise</code> | Hostport <code>&lt;host&gt;:&lt;port&gt;</code> to advertise how this server can be contacted by leaf nodes. This is useful in cluster setups with NAT. |

| Property                   | Description                                                                                                                                                      |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>no_advertise</code>  | if <code>true</code> the server shouldn't be advertised to leaf nodes.                                                                                           |
| <code>authorization</code> | Authorization block. <a href="#">See Authorization Block section below</a> .                                                                                     |
| <code>remotes</code>       | List of <code>remote</code> entries specifying servers where leafnode client connection can be made.                                                             |
| <code>reconnect</code>     | Interval in seconds at which reconnect attempts to a remote server are made.                                                                                     |
| <code>compression</code>   | Configures compression of leafnode connections similar to <a href="#">cluster routes</a> . Defaults to <code>s2_auto</code> . See details <a href="#">here</a> . |

# TLS Block

As of NATS v2.10.0

The `tls` block for `leafnodes` configuration has an additional field enabling a *TLS-first handshake* between the remote and the accepting server.

| Property                     | Description                                                                                                                                         |
|------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>handshake_first</code> | If <code>true</code> on the accepting side, remote leafnodes are required to also have this setting configured in the <code>remotes</code> section. |

## Accepting side

Since Leafnodes can connect to a variety of servers, the ability to indicate if the TLS handshake should be done first is configured in 2 places. The *accepting* side is in the `tls` block of the `leafnodes` block.

```
leafnodes {
  port: 7422
  tls {
    handshake_first: true
    # other TLS fields...
  }
}
```

With the above configuration, an older server, or a server that does not have the remote configuration also configured with `handshake_first: true`, will fail to create a leafnode connection because the accepting-side server will initiate the TLS handshake while the soliciting side will wait for the INFO protocol to be received.

## Remote side

To indicate that a leafnode connection should perform the TLS handshake first, it needs to be configured in the remote configuration:

```
leafnodes {
  remotes [
    {
      urls: ["tls://example:7422"]
      tls: {
        handshake_first: true
        # other TLS fields...
      }
    }
  ]
}
```

If the remote is configured as such but the server it is connecting to does not have `handshake_first: true` configured, the connection will fail since the solicit side is performing a TLS handshake but will receive an INFO protocol in clear.

## Authorization Block

“ ”

A leaf node can authenticate against any user account on the hub (the incoming side of the connection), including those defined in the accounts themselves. This authorization block is therefore optional if appropriate account users already exist.

Whether configuring users in the account or in this dedicated authorization block is more convenient will depend on your deployment style.

| Property | Description |
|----------|-------------|
|----------|-------------|

|                       |                                                                                                                  |
|-----------------------|------------------------------------------------------------------------------------------------------------------|
| <code>user</code>     | Username for the leaf node connection.                                                                           |
| <code>password</code> | Password for the user entry.                                                                                     |
| <code>account</code>  | <a href="#">Account</a> this leaf node connection should be bound to.                                            |
| <code>timeout</code>  | Maximum number of seconds to wait for leaf node authentication.                                                  |
| <code>users</code>    | List of credentials and account to bind to leaf node connections. <a href="#">See User Block section below</a> . |

# Users Block

| Property              | Description                                                           |
|-----------------------|-----------------------------------------------------------------------|
| <code>user</code>     | Username for the leaf node connection.                                |
| <code>password</code> | Password for the user entry.                                          |
| <code>account</code>  | <a href="#">Account</a> this leaf node connection should be bound to. |

Here are some examples of using basic user/password authentication for leaf nodes (note while this is using accounts it is not using JWTs)

Singleton mode:

```
leafnodes {
  port: ...
  authorization {
    user: leaf
    password: secret
    account: TheAccount
  }
}
```

With above configuration, if a soliciting server creates a Leafnode connection with url: `nats://leaf:secret@host:port`, then the accepting server will bind the leafnode connection to the account "TheAccount". This account needs to exist otherwise the connection will be rejected.

Multi-users mode:

```
leafnodes {
  port: ...
  authorization {
    users = [
```

```

    {user: leaf1, password: secret, account: account1}
    {user: leaf2, password: secret, account: account2}
  ]
}
}

```

With the above, if a server connects using `leaf1:secret@host:port`, then the accepting server will bind the connection to account `account1`. If using `leaf2` user, then the accepting server will bind to connection to `account2`.

If username/password (either singleton or multi-users) is defined, then the connecting server **MUST** provide the proper credentials otherwise the connection will be rejected.

If no username/password is provided, it is still possible to provide the account the connection should be associated with:

```

leafnodes {
  port: ...
  authorization {
    account: TheAccount
  }
}

```

With the above, a connection without credentials will be bound to the account "TheAccount".

If other form of credentials are used (jwt, nkey or other), then the server will attempt to authenticate and if successful associate to the account for that specific user. If the user authentication fails (wrong password, no such user, etc..) the connection will be also rejected.

## LeafNode `remotes` Entry Block

| Property                  | Description                                                                                                                                                 |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>url</code>          | Leafnode URL (URL protocol should be <code>nats-leaf</code> ).                                                                                              |
| <code>urls</code>         | Leafnode URL array. Supports multiple URLs for discovery, e.g., <code>urls: [ "nats-leaf://host1:7422", "nats-leaf://host2:7422" ]</code>                   |
| <code>no_randomize</code> | IF true, always try connecting in sequence on the URLs in the list. Default behavior if to shuffle the URLs and start connection attempts with a random URL |

| Property                        | Description                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>account</code>            | <a href="#">Account</a> name or JWT public key identifying the local account to bind to this remote server. Any traffic locally on this account will be forwarded to the remote server.                                                                                                                                                                                                             |
| <code>deny_imports</code>       | List of subjects which will not be imported over this leaf node connection. Subscriptions to those subjects will not be propagated to the hub.                                                                                                                                                                                                                                                      |
| <code>deny_exports</code>       | List of subjects which will not be exported over this leaf node connection. Subscriptions to those subjects will not be propagated into the leaf node.                                                                                                                                                                                                                                              |
| <code>credentials</code>        | Credential file for connecting to the leafnode server.                                                                                                                                                                                                                                                                                                                                              |
| <code>nkey</code>               | Nkey used for connecting to the leafnode server.                                                                                                                                                                                                                                                                                                                                                    |
| <code>tls</code>                | A <a href="#">TLS configuration</a> block. Leafnode client will use specified TLS certificates when connecting/authenticating.                                                                                                                                                                                                                                                                      |
| <code>ws_compression</code>     | If connecting with <a href="#">Websocket</a> protocol, this boolean ( <code>true</code> or <code>false</code> ) indicates to the remote server that it wishes to use compression. The default is <code>false</code> .                                                                                                                                                                               |
| <code>ws_no_masking</code>      | If connecting with <a href="#">Websocket</a> protocol, this boolean indicates to the remote server that it wishes not to mask outbound WebSocket frames. The default is <code>false</code> , which means that outbound frames will be masked.                                                                                                                                                       |
| <code>compression</code>        | Configures compression of leafnode connections similar to <a href="#">cluster routes</a> . Defaults to <code>s2_auto</code> . See details <a href="#">here</a> .                                                                                                                                                                                                                                    |
| <code>hub</code>                | Default is false. If set to true the roles of leaf node and hub will be reversed. This allows the hub to initiate a leaf node connection to the leaf.                                                                                                                                                                                                                                               |
| <code>first_info_timeout</code> | Default <code>1s</code> . The first information sent back by the hub (incoming side) will be the server metadata. The client will only wait <code>first_info_timeout</code> before giving up. This is useful if there is the change that the port on the other side is not a NATS server or not a port accepting leaf node connection. In this case the client would wait forever for the metadata. |

## Signature Handler

As of NATS Server v.2.9.0, for users embedding the NATS Server, it is possible to replace the use of the credentials file by a signature callback which will sign the `nonce` and provide the JWT in the `CONNECT` protocol. The `RemoteLeafOpts` has a new field:

```
SignatureCB  SignatureHandler
```

The callback definition is:

```
// SignatureHandler is used to sign a nonce from the server while
// authenticating with Nkeys. The callback should sign the nonce and
// return the JWT and the raw signature.
type SignatureHandler func([]byte) (string, []byte, error)
```

And example of how to use it can be found [here](#)

## Connecting using WebSocket protocol

Since NATS 2.2.0, Leaf nodes support outbound WebSocket connections by specifying `ws` as the scheme component of the remote server URLs:

```
leafnodes {
  remotes [
    {urls: ["ws://hostname1:443", "ws://hostname2:443"]}
  ]
}
```

Note that if a URL has the `ws` scheme, all URLs the list must be `ws`. You cannot mix and match. Therefore this would be considered an invalid configuration:

```
remotes [
  # Invalid configuration that will prevent the server from starting
  {urls: ["ws://hostname1:443", "nats://hostname2:7422"]}
]
```

Note that the decision to make a TLS connection is not based on `wss://` (as opposed to `ws://`) but instead in the presence of a TLS configuration in the `leafnodes{}` or the specific remote configuration block.

To configure Websocket in the remote server, check the [Websocket](#) section.

## tls Configuration Block

| Property               | Description                     |
|------------------------|---------------------------------|
| <code>cert_file</code> | TLS certificate file.           |
| <code>key_file</code>  | TLS certificate key file.       |
| <code>ca_file</code>   | TLS certificate authority file. |

| Property                       | Description                                                                                                                      |
|--------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <code>insecure</code>          | Skip certificate verification.                                                                                                   |
| <code>verify</code>            | If <code>true</code> , require and verify client certificates.                                                                   |
| <code>verify_and_map</code>    | If <code>true</code> , require and verify client certificates and use values map certificate values for authentication purposes. |
| <code>cipher_suites</code>     | When set, only the specified TLS cipher suites will be allowed. Values must match golang version used to build the server.       |
| <code>curve_preferences</code> | List of TLS cypher curves to use in order.                                                                                       |
| <code>timeout</code>           | TLS handshake timeout in fractional seconds.                                                                                     |

📄 #3

📄 cocobook-sync 📄

2026-10-28 11:13:52 UTC

📄 cocobook-sync 📄

2026-10-29 01:42:38 UTC