

Encryption at Rest



- [TPM \(Windows only\)](#)
 - [Changing encryption settings](#)
 - [Enabling with existing data](#)
 - [Disabling or changing the key](#)
 - [Changing the cipher](#)
 - [Performance considerations](#)
-

Supported since NATS server version 2.3.0

TPM is supported on Windows since NATS Server version 2.11.0

“ ”

Note, that although encryption at rest by the NATS server is fully supported, we recommend using file system encryption where available.

File system encryption, in particular when provided by Cloud hosted services, is optimized for throughput, does not place a burden on the NATS server and removes the need for secret management from the NATS installation.

The NATS server can be configured to encrypt message blocks which includes message headers and payloads. Other metadata files are encrypted as well, such as the stream metadata file and consumer metadata files.

Two choices of ciphers are currently supported:

- `chachapoly` - [ChaCha20-Poly1305](#)
- `aes` - [AES-GCM](#)

Enabling encryption is done through the `jetstream` [configuration block](#) on the server.

```
jetstream : {  
  cipher: chachapoly  
  key : "6dYfBV0zzEkR3vxZCNjxmnVh/aIqgid1"  
}
```

It is recommended to provide the encryption key through an environment variable at runtime, such as `$JS_KEY`, so it will not be persisted in a file.

```
jetstream : {  
  cipher: chachapoly  
  key: $JS_KEY  
}
```

The variable can be exported in the environment or passed when the server starts up.

```
JS_KEY="mykey" nats-server -c js.conf
```

TPM (Windows only)

```
jetstream {  
  store_dir: nats  
  max_file_store: 10G  
  tpm {  
    keys_file: "keys"  
    encryption_password: "pwd"  
  }  
}
```

Property	Description	Default	Version
<code>keys_file</code>	Specifies the file where encryption keys are stored. This option is required, otherwise TPM will not be active. If the file does NOT EXIST, a new key will be dynamically created and stored in the <code>pcr</code>	required	2.11.0

Property	Description	Default	Version
<code>encryption_password</code>	Password used for decrypting data in the keys file. OR, the password used to seal the dynamically created key in the TPM store.	required	2.11.0
<code>srk_password</code>	The Storage Root Key (SRK) password is used to access the TPM's storage root key. The srk password is optional in TPM 2.0.	not set	2.11.0
<code>pcr</code>	Platform Configuration Registers (PCRs). 0-16 are reserved. Pick a value from 17 to 23.	22	2.11.0
<code>cipher</code>	<code>chacha</code> / <code>chachapoly</code> or <code>aes</code> .	<code>chachapoly</code>	2.11.0

Changing encryption settings

Enabling with existing data

Enabling encryption on a server with existing data is supported. Do note that existing unencrypted message blocks will not be re-encrypted, however any new blocks that are stored *will* be encrypted going forward.

If it is desired to encrypt the existing blocks, the stream can be backed up and restored (which decrypts on backup and then re-encrypts when restoring it).

Disabling or changing the key

If encryption was enabled on the server and the server is restarted with a different key or disabled all together, the server will fail to decrypt messages when attempting to load them from the store. If this happens, you'll see log messages like the following:

```
Error decrypting our stream metafile: chacha20poly1305: message authentication failed
```

Note, that this will impact JetStream functionality, but the server will still support core NATS functionality.

Changing the cipher

It is possible to change the `cipher`, however the same key must be used. The server will properly encrypt new message blocks with the new cipher and decrypt existing messages blocks with the existing cipher.

Performance considerations

Performance considerations: As expected, encryption is likely to decrease performance, but by how much is hard to define. In some performance tests on a MacbookPro 2.8 GHz Intel Core i7 with SSD, we have observed as little as 1% decrease to more than 30%. In addition to CPU cycles required for encryption, the encrypted files may be larger, which results in more data being stored or read.

📦 #1

📦 cocobook-sync 📦 2026-10-28 11:14:42 UTC

📦 cocobook-sync 📦 2026-10-28 11:14:43 UTC