TLS: How does it Work? & More

Automated Certification via Let’s Encrypt

> For automated TLS certificate management, Traefik integrates with Let’s Encrypt. See detailed instructions in this Let’s Encrypt Post.

> If you want to Setup your Own TLS-Challenge go check out this Guide: Docker-compose with Let's Encrypt: TLS Challenge

What is TLS and How Does It Work?

TLS (Transport Layer Security) is a cryptographic protocol designed to provide secure communication over a network. It is the successor to SSL and ensures that data transmitted between a client and server is encrypted and authenticated, protecting it from eavesdropping and tampering.

When a client (such as a browser) connects to a server via HTTPS, the following steps occur:

- **Handshake:** The client and server exchange information to establish a secure connection, agreeing on encryption protocols and verifying identities using certificates.
- **Encryption:** Once the handshake is complete, all subsequent communication is encrypted to protect the data from being intercepted.
- **Data Integrity:** TLS ensures that the transmitted data has not been altered during transfer.

TLS is crucial for securing sensitive information like login credentials, credit card numbers, and personal data. Traefik makes it easy to manage TLS certificates, either through Let’s Encrypt or user-defined certificates.

More about TLS-Options for Traefik

User-Defined Certificates

To add or remove TLS certificates dynamically, define them in the `tls.certificates` section of the dynamic configuration:

File (YAML)

```
# Dynamic configuration
tls:
  certificates:
    - certFile: /path/to/domain.cert
      keyFile: /path/to/domain.key
    - certFile: /path/to/other-domain.cert
      keyFile: /path/to/other-domain.key
```

File (TOML)

Note: In Kubernetes, certificates must be provided as secrets instead of using the file provider.

Certificates Stores

In Traefik, certificates are grouped in certificate stores:

File (YAML)

```
# Dynamic configuration
tls:
  stores:
    default: {}
```

By default, all certificates are stored in the `default` store. Any additional store definitions are ignored.

File (YAML): Specifying Certificate Stores

```yaml
# Dynamic configuration
tls:
  certificates:
    - certFile: /path/to/domain.cert
      keyFile: /path/to/domain.key
      stores:
        - default
    - certFile: /path/to/other-domain.cert
      keyFile: /path/to/other-domain.key
```

Default Certificate

Traefik can use a default certificate for connections without SNI or matching domains. Define the default certificate in a TLS store:

File (YAML)

```yaml
# Dynamic configuration
tls:
  stores:
    default:
      defaultCertificate:
        certFile: /path/to/cert.crt
        keyFile: /path/to/cert.key
```

ACME Default Certificate: Traefik can also generate a default certificate using an ACME provider:

File (YAML)

```yaml
# Dynamic configuration
tls:
  stores:
    default:
      defaultGeneratedCert:
        resolver: myresolver
        domain:
          main: example.org
          sans:
            - foo.example.org
        - bar.example.org
```

TLS Options

The TLS options allow you to configure parameters of the TLS connection:

Default TLS Option

```
# Dynamic configuration
tls:
  options:
    default:
      minVersion: VersionTLS12
```

Minimum and Maximum TLS Version

```
# Dynamic configuration
tls:
  options:
    default:
      minVersion: VersionTLS12
      maxVersion: VersionTLS13
```

Cipher Suites

```
# Dynamic configuration
tls:
  options:
    default:
      cipherSuites:
        - TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
```

For more information, refer to the official Traefik documentation.

Revision #9
Created 12 September 2024 14:09:30 by aeoneros
Updated 11 February 2025 09:11:40 by aeoneros