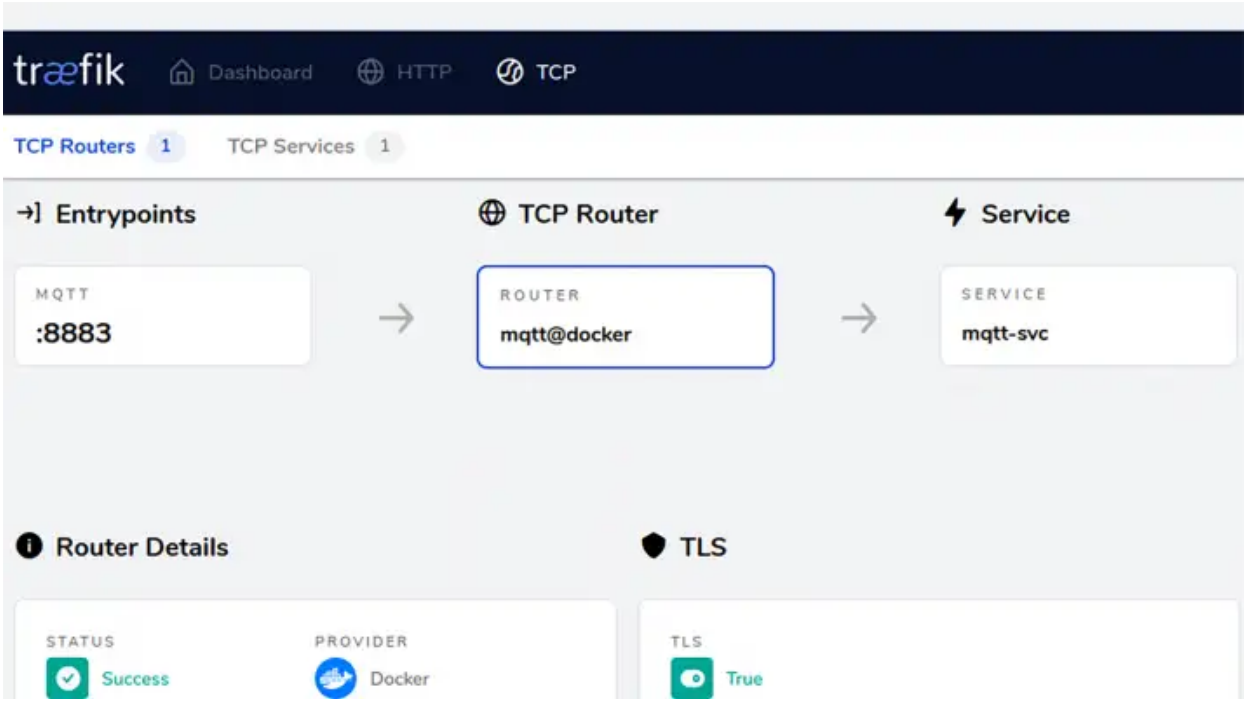
c't magazin für computer technik

TLS-Zertifikate: Træfik 2 für Container jenseits von HTTPS

| 18.02.2020 13:48 Uhr Merlin Schumacher



Wer viele verschlüsselte Dienste auf einem Server betreibt, hat oft mit Zertifikaten zu kämpfen. Træfik nimmt einem viel Arbeit ab, auch für TCP-Verbindungen.

Wenn Sie Træfik 1 für die Verwaltung von TLS-Zertifikaten und Routing einsetzen, müssen Sie spätestens dann die Zertifikate extrahieren, wenn ein Dienst nicht über HTTPS läuft. Dafür gibt es zwar zuverlässige Werkzeuge, aber am Ende liegen Zertifikatsdateien an verschiedenen Stellen im System – in Verzeichnissen, Container-Volumes oder gar Datenbanken.

Ist ein neues Zertifikat nötig, müssen Sie all die Daten aktualisieren und die zugehörigen Container oft neu starten. Das ist Fleißarbeit, die nicht nur nervt, sondern schnell zu Flüchtigkeitsfehlern führt. Mit Træfik 2 hat das ein Ende, denn der kann jedwede TCP-Verbindung nach außen per TLS absichern und kümmert sich vollautomatisch um die Zertifikatsverwaltung. Für den Dienst dahinter passiert das vollkommen transparent. Außerdem kann Træfik Anfragen je nach Hostname zu unterschiedlichen Diensten routen.

Wer bereits Erfahrung mit Træfik 2 hat, wird sich in das Konzept schnell hineinfinden. Es ähnelt dem Vorgehen bei HTTP-Verbindungen sehr. Für alle, für die Træfik 2 noch Neuland ist, **haben wir bereits eine Einführung verfasst [1]** . Darin werden Grundlagen wie die Einrichtung von Let's Encrypt und das HTTP-Routing erklärt. In diesem Artikel dient der MQTT-Broker Mosquitto als Beispieldienst. Er kann zwar selbst TLS anbieten, aber eben nur, wenn man ihm die richtigen Zertifikate zur Verfügung stellt. Eleganter ist es allerdings, wenn Træfik auch diesen Dienst absichert, wenn er sich ohnehin um die Verwaltung der Let's-Encrypt-Zertifikate kümmert. **Zum Artikel stellen wir ein Repository auf GitHub bereit [2]**. Dort sind alle hier beschriebenen Dateien enthalten.

Konfiguration

Das Beispiel-Repository geht davon aus, dass Sie eine laufende Docker-Installation mit aktuellem Docker-

Compose haben. Neben dem Compose-File gibt es noch zwei Konfigurationsdateien für Træefik namens static.yml und dynamic.yml. In seiner Standardkonfiguration horcht Træfik zunächst nur an den HTTP(S)-Ports, "Entrypoints" genannt. Damit er auch auf Anfragen für den MQTT-Port lauscht, müssen Sie den Entrypoint für den Port 8883 festlegen. Dafür bearbeiten Sie die Datei static.yml. Innerhalb des laufenden Docker-Containers ist sie dann für Træfik unter /etc/traefik/traefik.yml zu erreichen und dient als primäre Konfigurationsdatei. Erweitern Sie den Block `entryPoints` um einen Abschnitt für MQTT wie folgt:

```
entryPoints:
  web:
    address: ":80"
  web-secure:
    address: ":443"
  mqtt:
    address: ":8883"
```

Nun horcht Træfik an den Ports 80, 443 und 8883. Denken Sie daran, die Ports auch für den Træfik-Container nach außen zu öffnen, indem Sie die `docker-compose.yml` und Ihre Firewallregeln anpassen. Nach einem Neustart von Træfik sind die Änderungen angewendet.

[3]

Labels definieren

Damit Træfik über den neuen TCP-Dienst Bescheid weiß, müssen Sie noch ein paar Labels festlegen. Dafür bearbeiten Sie die Compose-Datei des Containers, etwa so:

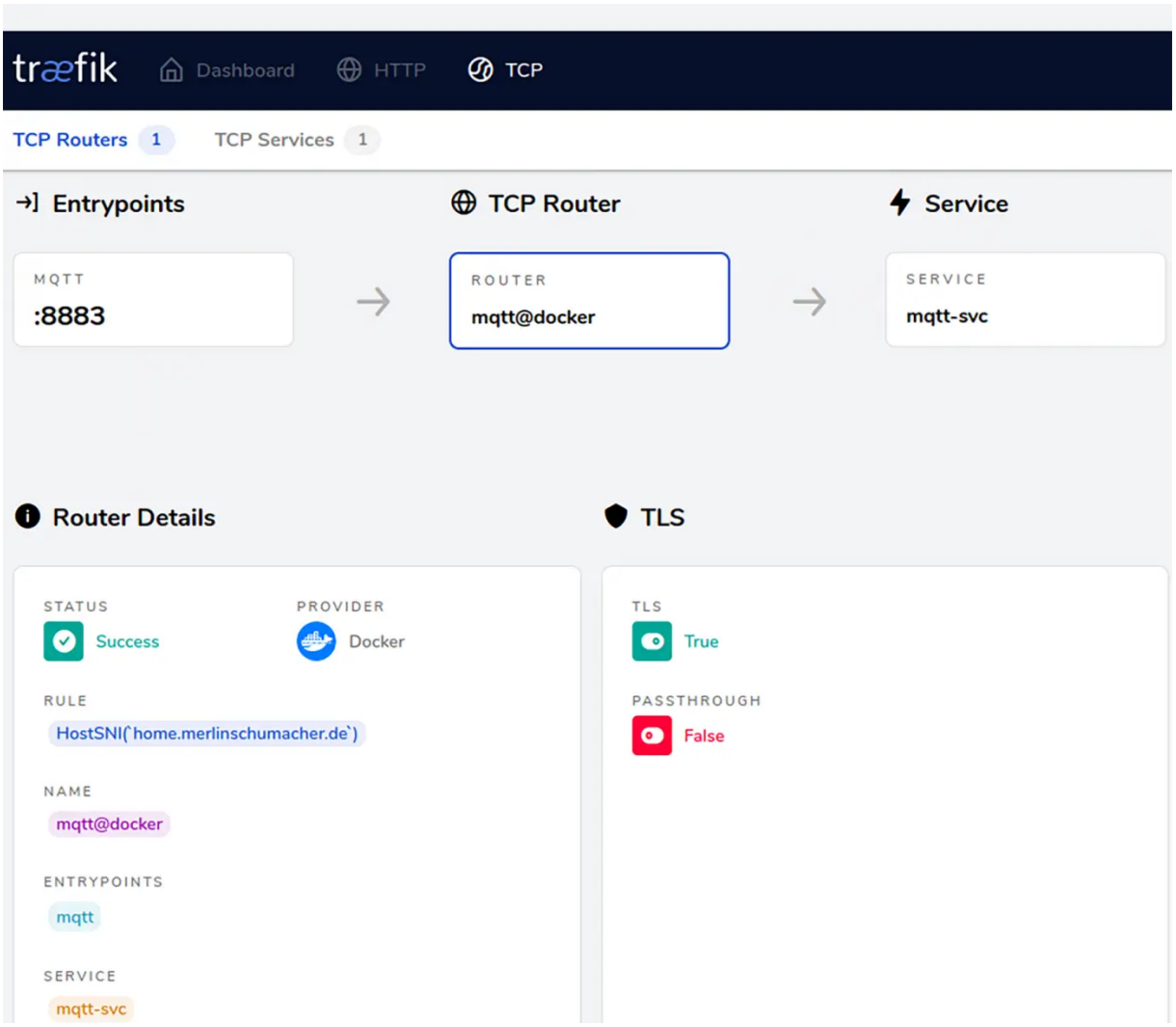
```
mqtt:
 image: "eclipse-mosquitto"
 labels:
 - "traefik.tcp.routers.mqtt.rule=HostSNI(`mqtt.example.org`)"
 - "traefik.tcp.routers.mqtt.entrypoints=mqtt"
 - "traefik.tcp.routers.mqtt.tls=true"
 - "traefik.tcp.routers.mqtt.tls.certResolver=default"
```

```
  - "traefik.tcp.routers.mqtt.service=mqtt-svc"
  - "traefik.tcp.services.mqtt-svc.loadbalancer.server.port=1883"
```

Das erste Label `traefik.tcp.routers.mqtt.rule` definiert den Hostnamen "mqtt.example.org". Bei Anfragen schicken moderne Clients immer den Hostnamen per SNI (Server Name Indication) mit. Der Router leitet die Anfrage dann passend weiter. Wollen Sie mehrere Dienste auf einem Server anbieten, müssen Sie unterschiedliche (Sub-)Domains nutzen.

Beim zweiten Label `traefik.tcp.routers.mqtt.entrypoints=mqtt` kommt der vorhin festgelegte Entrypoint zum Einsatz. `traefik.tcp.routers.mqtt.tls=true` bewirkt, dass der Dienst in den Genuss der TLS-Terminierung kommt. Fehlt die Variable oder ist nicht `true`, ist der Dienst auch nicht erreichbar, da Træfik ihn ignoriert. Via `traefik.tcp.routers.mqtt.tls.certResolver=default` erfährt Træfik, wo er das Zertifikat für diesen Dienst bestellen soll. In manchen Konstellationen, etwa bei Wildcard-Zertifikaten, ist das Fehlen unproblematisch, Um sich aber bei Konfigurationsänderungen Ärger zu ersparen, sollte man die Option setzen. `default` ist meist die richtige Wahl, außer man bezieht Zertifikate aus mehreren Quellen.

Zu jedem Træfik-Router gehört auch ein Service. Welcher das ist, bestimmt `traefik.tcp.routers.mqtt.service`. Die Service-Definition für `mqtt-svc` ist das letzte Label. Dort geben Sie den Port des Dienstes an, der im Container läuft. Bei MQTT ist das 1883. Anschließend starten oder aktualisieren Sie den Container mit `docker-compose up -d`. Haben Sie das Træfik-Dashboard aktiviert, sollten Sie im Abschnitt "TCP" sowohl den frisch konfigurierten Router als auch den Service sehen.



In Træfiks Dashboard kann man sehen, ob das Routing und die TLS-Terminierung funktionieren.

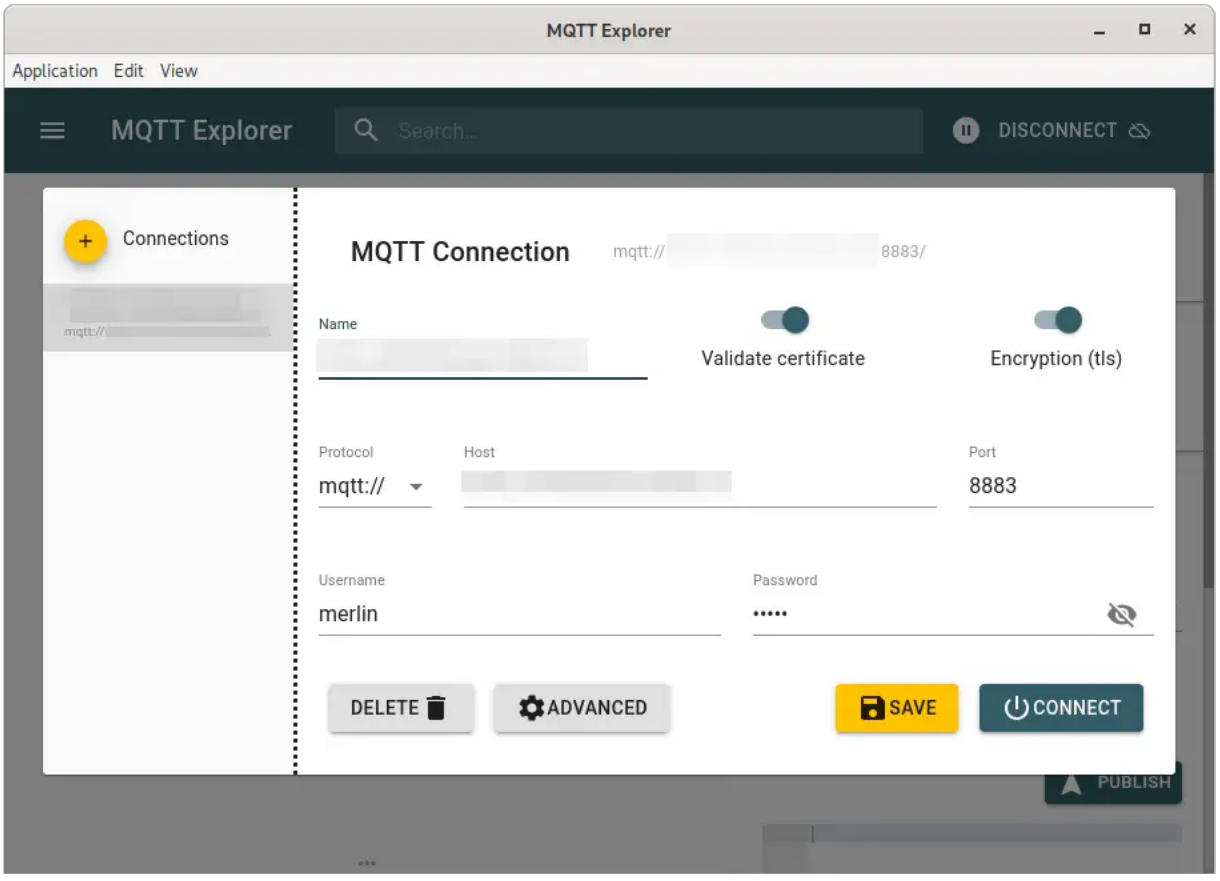
Damit Træfik einen TCP-Dienst routet, muss man nur eine Handvoll Labels vergeben.

Prüfen

Ob der Dienst nun verschlüsselt arbeitet, können Sie mit einem zum Dienst passenden Client oder mit OpenSSL überprüfen:

```
openssl s_client -connect mqtt.example.org:8883
```

OpenSSL stellt eine verschlüsselte Verbindung zu dem Dienst her und gibt dabei allerhand Informationen zu der Verbindung und den genutzten Zertifikaten aus. Nun können Sie direkt per Tastatureingabe Daten an den Dienst senden. Sofern der irgendwas damit anfangen kann, wird er antworten. Wird das Gespräch von einem "Bad Gateway" jäh beendet, klappt das Routing nicht und Træfik wusste nicht wohin mit den Daten. Mit Strg+C beenden Sie `openssl`. Für den Test des MQTT-Dienstes dahinter empfiehlt sich das Programm MQTT-Explorer. In dessen Einstellungen lässt sich festlegen, ob es die Zertifikate von TLS-Verbindungen überprüft oder nicht. Dadurch kann man Probleme mit der Zertifikatsbeschaffung diagnostizieren, weil der Broker erreichbar ist, auch wenn Træfik sein ungültiges Standardzertifikat ausliefert.



Der MQTT-Explorer ist ein hilfreiches Werkzeug beim Testen von MQTT-Verbindungen.

Abhärten

Wer sicherstellen möchte, dass die TLS-Konfiguration von Træfik den aktuellen Sicherheitsstandards entspricht, muss sich noch die Datei dynamic.yml vorknöpfen.:

```
tls:
```

```
options:
  default:
    minVersion: VersionTLS12
    sniStrict: true
    cipherSuites:
      - "TLS_ECDHE_ECDSA_WITH..."
      - "TLS_ECDHE_RSA_WITH_..."
```

Dort ist es ratsam, mindestens die TLS-Version 1.2 einzufordern (`minVersion`), Anfragen ohne SNI abzulehnen (`sniStrict`) und die Liste der Cipher-Suites einzuschränken, um gebrochene Verschlüsselungsverfahren auszuschließen. Eine gute Vorkonfiguration kann man **mit Mozillas SSL Configuration Generator erzeugen [4]**, der TLS-Konfigurationen für fast 20 verschiedene Server generieren kann. Einer davon ist Træfik. Leider spuckt der Generator nur Konfigurationsvorlagen in TOML (Tom's Obvious Minimal Language) statt YAML aus. Man muss also genau hinschauen, wo die Schlüssel aus Mozillas Vorschlag in der dynamic.yml landen müssen.

In den Einstellungen können Sie drei Stufen wählen: Modern, Intermedia und Old. Sie sollten Sie die Option "Intermediate" wählen, denn sie ist ein guter Kompromiss zwischen Sicherheit und Kompatibilität. Die von Mozilla für den HTTPS-Router (`http.routers.router-secure.tls`) vorgeschlagene Option `intermediate` aktiviert allerhand überholte Cipher-Suites. Lassen Sie sie weg. Es reicht, die TLS-Optionen sowie die HSTS-Middleware zu aktivieren. Anschließend sollte Ihr Server in den **SSL-Tests von Qualys [5]** oder **Immuniweb [6]** hohe Noten erreichen. Es lohnt sich, alle paar Monate mal ein Auge auf Mozillas Generator zu werfen und die Optionen und Cipher-Liste zu aktualisieren. (**mls [7]**)

URL dieses Artikels:

https://www.heise.de/-4658800

Links in diesem Artikel:

[1] https://www.heise.de/ratgeber/Eingehenden-HTTP-Verkehr-mit-Traefik-routen-4483710.html

[2] https://github.com/jamct/traefik-example/tree/master/06_tcp

[3] https://www.heise.de/ct/

[4] https://ssl-config.mozilla.org/

[5] https://www.ssllabs.com/ssltest/

[6] https://www.immuniweb.com/ssl/

[7] mailto:mls@ct.de