Caddy

- Caddyfile
  - Grundlegende Reverseproxyfunktion
  - Andere Caddy-Konfigurationen importieren
  - Vorhandenes Zertifikat in Caddy verwenden
  - Log-Datei für virtuellen Host in Caddy schreiben
  - IP-Bereiche für Zugriffe auf Caddy Server sperren
  - Automatische HTTP auf HTTPS Umleitung einrichten
- Administration
  - Plugins mit xCaddy installieren

Caddyfile

Caddyfile

Grundlegende Reverseproxyfunktion

Struktur ohne TLS

```
<FQDN> {
  reverse_proxy <ip-oder-hostname>:<port>
}
```

Struktur mit internen TLS Zertifikat

```
<FQDN> {
  reverse_proxy <ip-oder-hostname>:<port>
  tls internal
}
```

Andere Caddy-Konfigurationen importieren

Wir können, um die Konfiguration besser zu organisieren, einzelne Konfigurationsdateien für unsere virtuellen Hosts erstellen. Diese importieren wir dann mit einem Befehl in unsere **Caddyfile**.

Wir fügen einfach am Ende unserer **Caddyfile** den folgenden Befehl ein:

```
import /etc/caddy/conf/*.caddy
```

Es werden dann alle Dateien mit der `.caddy` Endung aus dem **conf-Verzeichnis** in die **Caddyfile** geladen. Das ermöglicht dann unsere Dateien besser zu organisieren.

Caddyfile

Vorhandenes Zertifikat in Caddy verwenden

Wir können für unseren **Caddy-Webserver** auch vorhandene **TLS-Zertifikate** verwenden. Dazu erstellen wir einen **virtuellen Host** und fügen dem die `tls` Option hinzu. Hinter der Option geben wir dann noch an, wo Caddy das **Zertifikat** und den **Zertifikatsschlüssel** findet.

```
www.website.de {
	reverse_proxy 192.168.10.2:80


	tls /opt/certificates/website.de/cert.pem /opt/certificates/website.de/key.pem


}
```

Caddyfile

Log-Datei für virtuellen Host in Caddy schreiben

Man kann in der Konfiguration des virtuellen Hosts im Caddy hinterlegen, ob eine **Log-Datei** geschrieben werden soll. Um diese dann mit dem **Prometheus Log Exporter** auszuwerten, soll die Log-Datei als **JSON** formatiert werden.

```
www.website.de {
	reverse_proxy 192.168.10.2

	tls /opt/certificates/website.de/cert.pem /opt/certificates/website.de/key.pem

	log {
		output file /var/log/caddy/website.access.log
		format json
	}
}
```

Caddyfile

IP-Bereiche für Zugriffe auf Caddy Server sperren

Um IP-Bereiche für den Zugriff auf einen virtuellen Host im **Caddy Webserver** zu blockieren, müssen wir in die Konfiguration der **Caddyfile** den Befehl `@blocked not remote ip <ip-subnet1> <ip-subnet2> <...>` verwenden.

Im Anschluss müssen wir nur den "Verweigerungstext" angeben. Schlussendlich kann die Konfiguration wie folgt aussehen:

```
www.website.de {
	reverse_proxy 192.168.10.2:80
	@blocked not remote_ip 192.168.20.0/24
	respond @blocked "Zugriff verweigert!" 403

	tls /opt/certificates/website.de/cert.pem /opt/certificates/website.de/key.pem

	log {
		output file /var/log/caddy/website.access.log
		format json
	}
}
```

Automatische HTTP auf HTTPS Umleitung einrichten

Einleitung

In diesem kurzen Artikel geht es darum, wie wir in unserer Caddyfile einrichten können, dass jeglicher HTTP-Verkehr automatisch auf HTTPS umgeleitet wird. Dazu müssen wir unsere Konfigurationsdatei nur minimal anpassen.

Durchführung

Im ersten Schritt öffnen wir unsere Caddyfile mit einem Editor unserer Wahl. Ich verwende immer `nano` dazu.

```
nano /etc/caddy/Caddyfile
```

Dort fügen wir folgenden Inhalt ein:

```
http:// {
        redir https://{host}{uri}
}
```

Sobald wir unsere Datei angepasst haben, lassen wir einmal Caddy überprüfen, ob unsere Konfiguration verwendbar für Caddy ist. Dazu führen wir den folgenden Befehl aus:

```
caddy validate --config /etc/caddy/Caddyfile
```

> **Info:** Wenn als Ausgabe "Valid Configuration" erscheint, haben wir alles richtig gemacht.

Zum Schluss aktivieren wir noch unsere angepasste Konfiguration:

```
systemctl reload caddy
```

Administration

Plugins mit xCaddy installieren

Einleitung

In diesem kurzen Artikel gehe ich drauf ein, wie man mit `xcaddy` weitere Plugins für den C**addy-Webserver** installieren kann, um den Webserver mit weiteren Funktionen auszustatten.

Dadurch können wir weitere Module hinzufügen, wie z.B. das DNS-Modul für ACME-Challenges mit einem Hetzner Nameserver, oder die NTLM Weiterleitung. Auf der folgenden Webseite kann man sich über die angebotenen Module informieren:

https://caddyserver.com/download

Plugins installieren

xcaddy installieren

> **Info:** Stelle bitte sicher das **golang** auf deinem Server installiert ist.

Im ersten Schritt müssen wir `xcaddy` auf unserem System installieren. Dazu führen wir den folgenden Befehl aus:

```
go install github.com/caddyserver/xcaddy/cmd/xcaddy@latest
```

Dadurch wird xcaddy auf dem System installiert. Durch die Ausführung des Befehls `xcaddy version` sollte uns, wenn alles richtig installiert ist, eine Versionsnummer ausgeben.

Plugins installieren

Um die Plugins zu installieren, müssen wir den folgenden Befehl verwenden:

```
xcaddy build --with github.com/caddy-plugin
```

Wenn wir mehrere Plugins auf einmal verwenden möchten, müssen wir den folgenden Befehl verwenden:

```
xcaddy build --with github.com/{plugin1} --with github.com/{plugin2} --with github.com/{plugin3}
```

Es wird in dem aktuellen Verzeichnis dann eine Datei `caddy` generiert. Diese müssen wir dann in das entsprechende Verzeichnis kopieren, indem sich die Datei befindet. In der Regel befindet sich die Datei unter: `/usr/local/bin/caddy`

Im Anschluss starten wir den Server einmal neu und dann sollten die Plugins aktiv sein. Überprüfen können wir die Module mit dem folgenden Befehl:

```
caddy list-modules
```