Sprawozdanie
bezpieczeństwo protokołu MQTT
uwierzetylnianie i autoryzacja

Krzysztof Ruczkowski

2 maja 2020

Spis treści

1 Wprowadzenie **3**

2 Przebieg prac **3**
2.1 Stworzenie użytkowników 3
2.2 Konfiguracja uwierzytelniania 3
2.3 Test konfiguracji uwierzytelniania 3
2.4 Stworzenie pliku kontroli dostępu 3
2.5 Konfiguracja autoryzacji 4
2.6 Test konfiguracji autoryzacji 4

3 Podsumowanie **4**

1 Wprowadzenie

Sprawozdanie dotyczy labolatorium z bezpieczeństwa protokołu MQTT odnoszącego się do uwierzytelniania i autoryzacji.
Przetestowane na systemie Arch Linux.

2 Przebieg prac

2.1 Stworzenie użytkowników

Poniższe komendy wykonuję jako superuser, aby móc zapisywać do katalogu `/etc/mosquitto/`:

```
cd /etc/mosquitto/
mosquitto_passwd -c passwd.conf server
mosquitto_passwd -b passwd.conf client password
```

Plik `passwd.conf` zawiera loginy i zaszyfrowane hasła w formacie przypominającym plik `/etc/shadow`.

2.2 Konfiguracja uwierzytelniania

Po stworzeniu użytkowników należy edytować plik `mosquitto.conf` zgodnie z instrukcją. Należy dodać tam linijki uniemożliwiające połączenie bez loginu (`allow_anonymous false`) oraz wskazać plik z loginami i hasłami (`password_file /etc/mosquitto/passwd.conf`).

```
vim /etc/mosquitto/mosquitto.conf
```

Po konfiguracji nie trzeba uruchamiać ponownie komputera, wystarczy zrestartować serwis:

```
systemctl restart mosquitto
```

2.3 Test konfiguracji uwierzytelniania

Po konfiguracji pozostaje przetestować funkcjonalność przykładu, odpowiednio modyfikując pliki `sender.py` i `receiver.py`:

```
# dane dostępowe dla klienta
# analogicznie ustawiamy dane dostępowe serwera w receiver.py
client.username_pw_set(username='client', password='password')
```

Jeżeli nie dodamy uwierzytelnienia, nie będzie można wysyłać i odbierać wiadomości.

2.4 Stworzenie pliku kontroli dostępu

Poniższe komendy wykonuję jako superuser, aby móc zapisywać do katalogu `/etc/mosquitto/`:

```
cat > /etc/mosquitto/acfile.conf
# This only affects clients with username "server".
user server
topic server/name
topic worker/name

# This only affects clients with username "client".
user client
topic read server/name
topic worker/name
```

Działanie tego pliku jest opisane w sekcji `Default authentication and topic access control` w `mosquitto.conf`.

2.5 Konfiguracja autoryzacji

Po stworzeniu użytkowników należy edytować plik `mosquitto.conf` zgodnie z instrukcją. Należy dodać tam wskazać plik kontroli dostępu (`acl_file /etc/mosquitto/acfile.conf`).

```
vim /etc/mosquitto/mosquitto.conf
```

Po konfiguracji nie trzeba uruchamiać ponownie komputera, wystarczy zrestartować serwis:

```
systemctl restart mosquitto
```

2.6 Test konfiguracji autoryzacji

Po konfiguracji pozostaje przetestować funkcjonalność przykładu, odpowiednio modyfikując pliki `sender.py` i `receiver.py`:

```
# sender.py
from tkinter import messagebox

# (...)
def process_message(client, userdata, message):
    message_decoded = (str(message.payload.decode("utf-8")))
    messagebox.showinfo("Message from the Server", message_decoded)

def connect_to_broker():
    # (...)
    client.on_message = process_message
    client.loop_start()
    client.subscribe("server/name")
    # (...)

def disconnect_from_broker():
    # (...)
    client.loop_stop()
    # (...)

# receiver.py
# (...)
def create_main_window():
    # (...)
    hello_button = tkinter.Button(window, text="Hello from the server",
            command=lambda: client.publish("server/name", "Hello from the server"))
    # (...)
    hello_button.pack(side="right")
    # (...)
```

3 Podsumowanie

Uwierzytelnianie i autoryzacja w MQTT pozwala nam na bezpieczny przesył danych, aby uniemożliwić niepowołanym osobom na ich przechwycenie.

Można nasłuchiwać wszystkich wiadomości używając komendy `mosquitto_sub`, co przydaje się w debugowaniu:

```
mosquitto_sub -v -h MightyTos4 -t '#' -u server -P server_pass -p 8883 --cafile ca.crt
```