

Projeto II - CoAP

Alline Domingos e Natália Miranda

Engenharia de Telecomunicações

Instituto Federal de Santa Catarina

Dezembro de 2018

Introdução

O presente relatório tem como principal objetivo apresentar as especificações do protocolo Coap desenvolvido na disciplina de Protocolo por meio da linguagem python.

O CoAP (Constrained Application Protocol) é um protocolo de comunicação responsável por gerenciar a transferência de dados em redes de dispositivos com objetivo de prover frameworks para aplicações que manipulam recursos simples em redes de objetos interligados.

Sendo assim, baseado nas especificações descrita pela RFC 7252 uma versão simplificada do protocolo CoAP utilizando UDP como protocolo de transporte foi implementado neste trabalho.

Este relatório está dividido em 3 seções, onde na primeira seção apresenta-se as especificações da aplicação desenvolvida, em seguida na segunda seção apresenta-se a classe da aplicação, e por fim faz-se uma breve explicação sobre a utilização da aplicação desenvolvida.

1. Especificação

De acordo com a o RFC 7252 (IETF, 2014, p. 1) o protocolo foi projetado para aplicações M2M como, por exemplo, smart energy e automação residencial. E para isso ele define quatro tipos de mensagens: Confirmable, Non-confirmable, Acknowledgment e Reset (RFC 7252, 2014, p. 8).

- **Confirmable(CON):** Mensagens que precisam ser confirmadas e devem resultar em uma mensagem do tipo Acknowledgment ou Reset;
- **Non-confirmable(NON):** Mensagem que não necessita de confirmação, pois não carrega informações críticas.
- **Acknowledgment(ACK):** Mensagens que confirmam o recebimento de uma mensagem Confirmable.
- **Reset(RST):** indica que uma mensagem (CON ou NON) foi recebida, mas por falta de algum contexto ela não pôde ser processada.

Outras características do CoAP descritas na RFC 7252 são:

- Troca de mensagens assíncrona;
- Suporte à URI e Content-Type;
- Capacidades simples de proxy e caching;
- Mapeamento HTTP que permite que proxies possam prover acesso aos recursos do CoAP via HTTP de maneira uniforme;
- Interligação segura para Datagram Transport Layer Security (DTLS);

- Binding em User Datagram Protocol (UDP) com confiabilidade opcional suportando requisições tanto unicast quanto multicast;
- Suporte aos métodos GET, POST, PUT, DELETE.

2. Arquitetura Coap

O protocolo CoAP se encontra entre as camadas de aplicação e transporte sobre arquitetura TCP/IP, onde este criará duas subcamadas de Requisição/Resposta e Mensagem.



Figura 1 - Arquitetura CoAP¹

O protocolo utiliza um modelo de mensagens específico para troca de mensagens ponto-a-ponto por meio dos métodos get, post, put e delete utilizando um cabeçalho binário de comprimento fixo (4 bytes) que contém um formato padrão ilustrado pela figura 2 abaixo.

Ver	Tipo	TLK	Código	ID Mensagem
Token (opcional)				
Opções (opcional)				
Payload (opcional)				

Figura 2 - Datagrama Coap

Os campos que formam o cabeçalho do datagrama CoAP possuem objetivo e tamanho padrão definidos pela RFC que são:

- **Versão (ver):** Dois bits que indica o número correspondente à versão do CoAP;

¹Disponível em:

<http://wireengenharia.com.br/wp-content/uploads/2018/07/Figura-1-%E2%80%93-Posicao-do-protocolo-na-Arquitetura-TCP-IP.png>

- **Tipo (T):** Dois bits que indica qual é o tipo da mensagem enviada;
 0. Confirmable,
 1. Non-confirmable,
 2. Acknowledgment,
 3. Reset.
- **TKL:** Quatro bits que indicam a largura variável do token (até 8 bytes);
- **Código:** Oito bits que indicam o tipo da mensagem (método de requisição ou um código de resposta)
- **ID da mensagem:** Dezesesseis bits utilizados para detectar duplicação de mensagens.
- **Token:** campo utilizado para relacionar requisições e respostas.
- **Opções:** campo opcional para indicar opções específicas como caminho de uma informação ou host.
- **Payload:** campo opcional, destinado para armazenamento de carga útil.

Os métodos utilizados pelo protocolo para o acesso aos cliente aos recursos disponibilizados na URL pelo servidor são baseados no modelo REST como no protocolo HTTP e são:

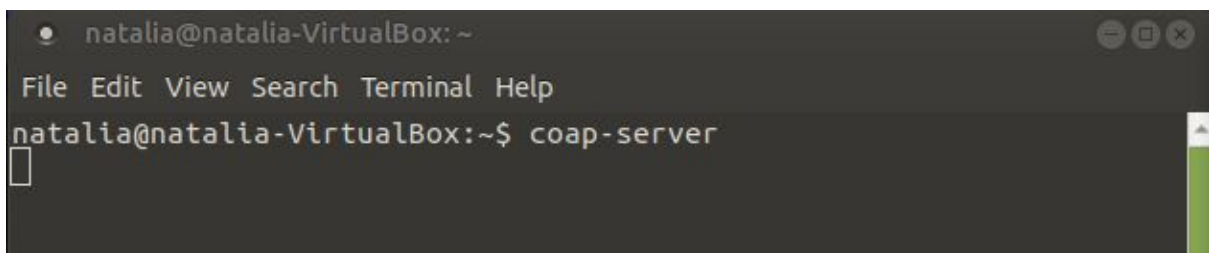
- **GET:** Solicita ao webserver a representação de um recurso.
- **POST:** Cria um recurso no webserver.
- **PUT:** Muda o estado de um recurso no webserver.
- **DELETE:** Remove ou altera o estado de um recurso.

3. Manual

Para utilizar o aplicação implementada os passos abaixo devem ser executados na sequência descrita:

3.1 Executar CoAP server:

A primeira etapa a ser executada para utilização da programa corresponde a execução do servidor CoAP que pode ser instalado por meio da biblioteca libcoap. A figura 3 abaixo traz o comando de execução que deve ser executado.



```
natalia@natalia-VirtualBox: ~
File Edit View Search Terminal Help
natalia@natalia-VirtualBox:~$ coap-server
```

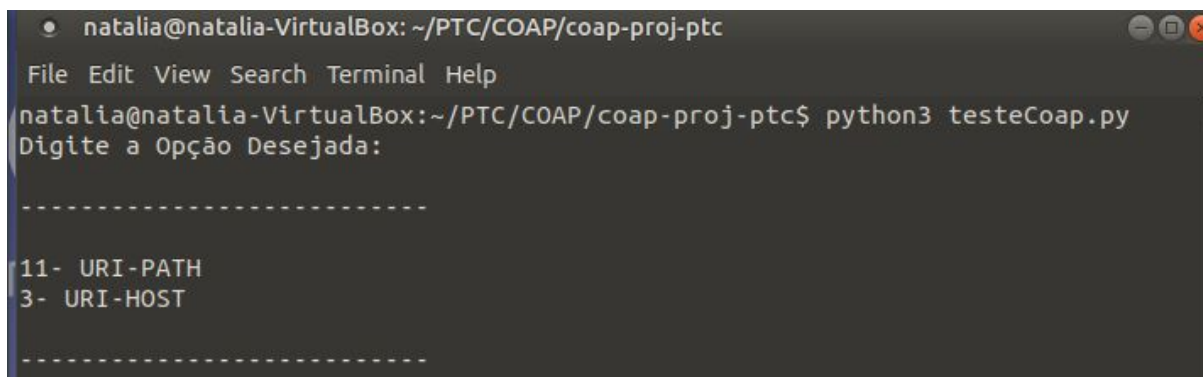
Figura 3 - Servidor Coap

3.2 Executar Wireshark:

Após executar o CoAP server você deve abrir o programa Wireshark em sua máquina e configurá-lo para capturar pacotes dentro de sua rede local (Loopback-lo).

3.3 Executar CoAP Cliente:

Com servidor CoAP rodando e com Wireshark capturando pacotes na rede local, a próxima etapa consiste na execução do programa. Para isso deve se rodar o linha de comando abaixo. Após isso deve se indicar a opção Coap desejada.

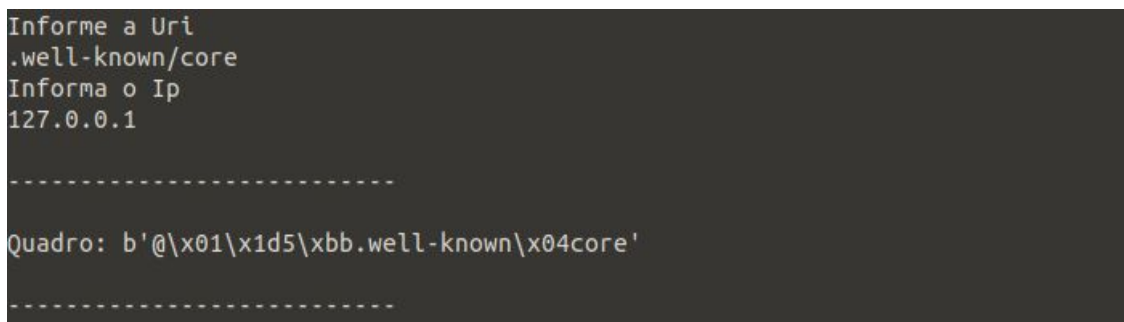
A terminal window titled 'natalia@natalia-VirtualBox: ~/PTC/COAP/coap-proj-ptc'. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The prompt is 'natalia@natalia-VirtualBox:~/PTC/COAP/coap-proj-ptc\$'. The user has entered 'python3 testeCoap.py'. The program outputs 'Digite a Opção Desejada:' followed by a dashed line. Below the line, it lists '11- URI-PATH' and '3- URI-HOST', followed by another dashed line.

```
natalia@natalia-VirtualBox: ~/PTC/COAP/coap-proj-ptc
File Edit View Search Terminal Help
natalia@natalia-VirtualBox:~/PTC/COAP/coap-proj-ptc$ python3 testeCoap.py
Digite a Opção Desejada:

-----
11- URI-PATH
3- URI-HOST
-----
```

Figura 4 - Cliente CoAP

Em seguida deve se indicar a uri e o endereço Ip local. Após executar esta ação o quadro de mensagem CoAP será gerado e enviado.

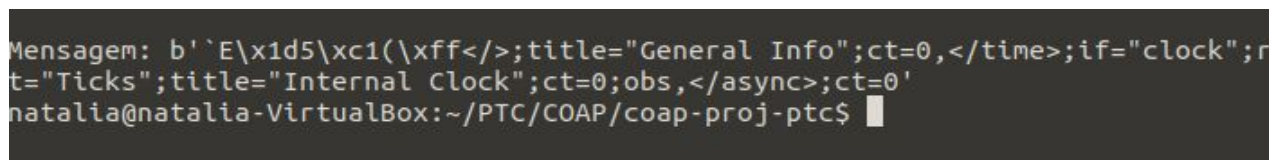
A terminal window showing the program's prompts and user input. It asks 'Informe a Uri' and '.well-known/core' is entered. Then it asks 'Informa o Ip' and '127.0.0.1' is entered. After a dashed line, it shows the generated CoAP message frame: 'Quadro: b'@\x01\x1d5\xbb.well-known\x04core''.

```
Informe a Uri
.well-known/core
Informa o Ip
127.0.0.1

-----
Quadro: b'@\x01\x1d5\xbb.well-known\x04core'
-----
```

Figura 5 - Cliente CoAP

Ao final uma confirmação será retornado pelo servidor indicado o recebimento da requisição.

A terminal window showing the received CoAP message. The message is displayed as 'Mensagem: b'\x01\x1d5\x01(\xff</>;title="General Info";ct=0,</time>;if="clock";r...' and continues on the next line with 't="Ticks";title="Internal Clock";ct=0;obs,</async>;ct=0'. The prompt 'natalia@natalia-VirtualBox:~/PTC/COAP/coap-proj-ptc\$' is visible at the bottom.

```
Mensagem: b'\x01\x1d5\x01(\xff</>;title="General Info";ct=0,</time>;if="clock";r
t="Ticks";title="Internal Clock";ct=0;obs,</async>;ct=0'
natalia@natalia-VirtualBox:~/PTC/COAP/coap-proj-ptc$
```

Figura 6 - Cliente CoAP

3.4 Confirmação:

No terminal após o envio do pacote uma mensagem de confirmação de recepção será retornado pelo servidor indicando que a mensagem de requisição foi recebida com sucesso. Este foi

[2] CoAP, Disponível em: <<http://wireengenharia.com.br/br/tag/coap/>>. Acesso: 16 Dez 2018.