

Protocolo de Aplicação - Para Sensoriamento

Guilherme de Albuquerque
Instituto Federal de Santa Catarina
Campus São José

Marcus Vinicius Bunn
Instituto Federal de Santa Catarina
Campus São José

Vinicius Bandeira
Instituto Federal de Santa Catarina
Campus São José

20 de dezembro de 2016

1 Introdução

Este documento visa exemplificar e comentar a solução proposta pela equipe para o problema descrito na disciplina de Projetos e Protocolos do curso de Engenharia de Telecomunicações [1]. Sendo esta, gerar um protocolo de aplicação que gerenciase um sistema de sensoriamento e controle, para por exemplo, monitorar janelas, portas e iluminação de uma residência. Para tal, foi escolhido utilizar um modelo de Publisher e Subscriber [2]. Neste, existe um elemento central da rede, denominado Publisher, que é responsável por enviar mensagens aos demais elementos da rede que são os Subscribers. Assim, a figura 1 demonstra o modelo de comunicação do serviço. Nela, é possível observar que a comunicação só é feita do Publisher para os Subscribers e dos Subscriber para o Publisher, não existindo assim comunicação entre os Subscribers.

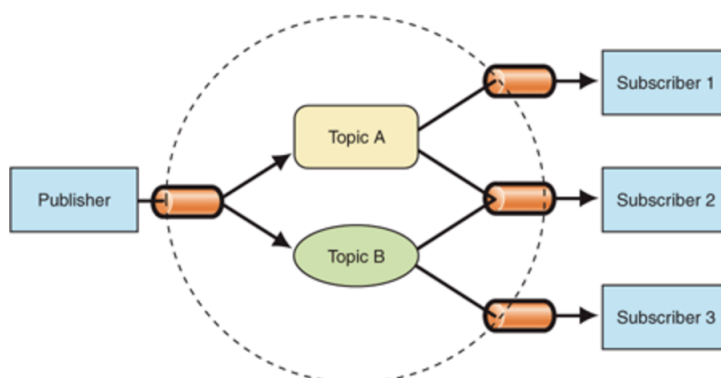


Figura 1 – Modelo de Comunicação Publish/Subscribe

Assim, será discutido o processo de criação de protocolo evidenciando os pontos chave escolhidos para desenvolvimento do projeto. Além disso, será apresentado um ambiente de testes utilizado para verificação projeto e será ilustrado um guia rápido de utilização do protocolo.

Por fim, a solução será apresentada em três sessões principais de onde o protocolo foi modelado, sendo elas intituladas: Tipos de Mensagem, Publisher e Subscriber. Além disso, em outras duas sessões a execução da solução será discutida, sendo elas : Testes, Resultados e Limitações do Protocolo. Por fim, um apêndice irá demonstrar um guia de utilização do protocolo.

2 Tipos de Mensagem

As mensagens foram modeladas utilizando a linguagem ANS.1 [3], que é uma abstração de mensagens para ambientes heterogêneos, e nela foram modeladas cinco tipos de mensagem que serão utilizadas pra realizar a comunicação do protocolo. Sendo elas, publish, subscribe, unsubscribe, notify e ACKsubs. Sendo que, para o desenvolvimento da aplicação, foi utilizado um compilador em C que traduzisse a implementação na linguagem ASN.1 para C [4]. Ainda mais, vale ressaltar que a codificação utilizada foi a DER.

A seguir, as mensagem serão apresentadas com o seu formato em ASN.1:

2.1 Publish

```
Publish ::= SEQUENCE {  
    subject OBJECT IDENTIFIER,  
    value BOOLEAN  
}
```

2.2 Subscribe

```
Subscribe ::= SEQUENCE {  
    subject OBJECT IDENTIFIER  
}
```

2.3 Unsubscribe

```
Unsubscribe ::= SEQUENCE {  
    subject OBJECT IDENTIFIER  
}
```

2.4 Notify

```
Notify ::= SEQUENCE {  
    subject OBJECT IDENTIFIER,  
    value BOOLEAN,  
    ip PrintableString(SIZE(1..256))  
}
```

2.5 ACKsubs

```
ACKsubs ::= SEQUENCE {  
    subject OBJECT IDENTIFIER,  
    value BOOLEAN  
}
```

3 Broker

O Broker é um servidor da rede, que funciona de forma passiva. Assim, ele só executa operações quando requisitado por um participante e fica ocioso no restante do tempo apenas aguardando comunicação entre as conexões. Seguindo, nele são armazenadas as conexões de todos os elementos da rede, separados pelos tópicos em que estão assinados. Ainda mais, cabe a ele a responsabilidade de controlar a assinatura dos Subscribers nos tópicos adicionando e retirando eles. Porém, somente quando tal operação for requisitada por um participante.

Portanto, abaixo serão apresentados as operações que o Broker pode executar.

- Adicionar à um tópico: Quando o Broker recebe do Participante uma mensagem de Subscribe, ele então adiciona o endereço IP do mesmo em uma lista de IPs, fazendo o mesmo com o assunto no qual o nó deseja se inscrever.
- Retirar de um tópico: Quando o participante solicita um Unsubscribe, o Broker realiza a ação de percorrer a lista de IPs, apagando o referente ao da máquina que solicitou a ação.
- Notificar elementos: Ao receber um Publish de um nó, o Broker notifica os demais, através da mensagem de Notify, o estado e o IP do participante publicador. Nesse ponto o Broker também envia uma mensagem chamada ACKSubs para o publicador, apenas indicando a ele que seu publish foi realizado com sucesso.

4 Participante

Como já foi mencionado nas sessões anteriores o Subscriber é o membro ativo da rede. Ele é o responsável por iniciar uma conexão e realizar ações da aplicação, que serão tratadas pelo Broker. Assim, sempre que for necessária uma operação o Subscriber irá mandar uma requisição via mensagem ASN.1 na qual o broker irá tratar e responder ao Subscriber.

Portanto, a seguir serão apresentadas as operações que o Subscriber tem acesso:

- Inscrever em um tópico: Quando deseja se inscrever em um assunto, o Participante envia uma mensagem de Subscribe ao Broker.
- Remover de um tópico: Se for de interesse um Participante pode também se desvincular de um tópico através de uma mensagem de Unsubscribe enviada ao Broker.
- Publicar seu estado: É através de uma mensagem de Publish que um Participante informa aos outros assinantes do assunto o seu estado atual.
- Alterar seu estado: É permitido também ao Participante a troca de assunto, selecionando essa opção no seu menu de opções.

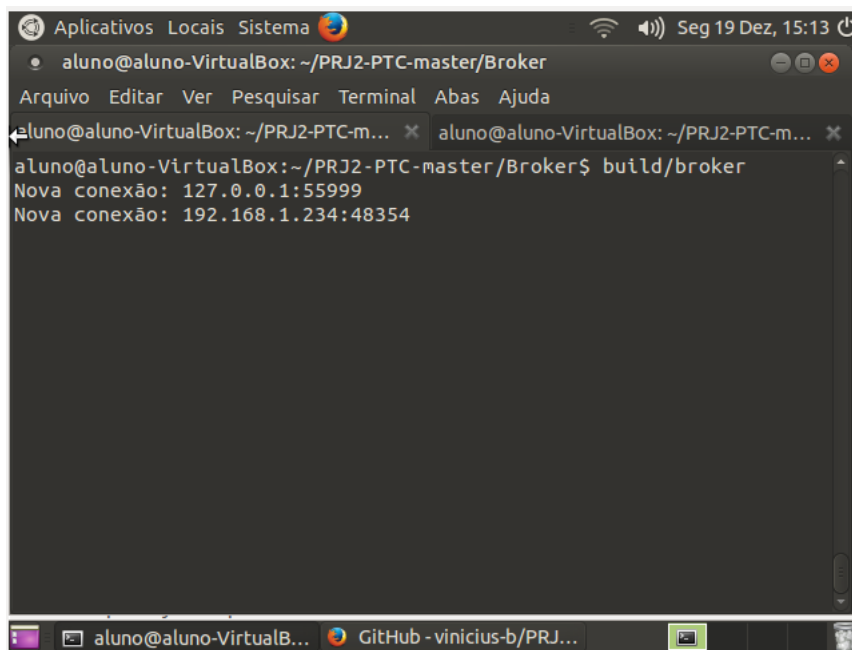
5 Testes e Resultados

Nesta sessão assume-se que o leitor é familiarizado com programação em C++ e operações básicas do sistema operacional Linux, e da solução apresentada neste documento para um protocolo de aplicação para uma rede de sensoriamento .

Sendo demonstrado os recursos do protocolo gerado, foi gerado um ambiente de testes para demonstrar a execução e o correto funcionamento do protocolo. Onde, a criação deste ambiente segue os critérios definidos no Apêndice A - Instruções de Uso.

5.1 Teste 1 - Conexão de Participantes

Na imagem abaixo vemos o comportamento do Broker ao receber conexões de dois participantes, um na máquina local (IP local - 127.0.0.1) e um em outra máquina.

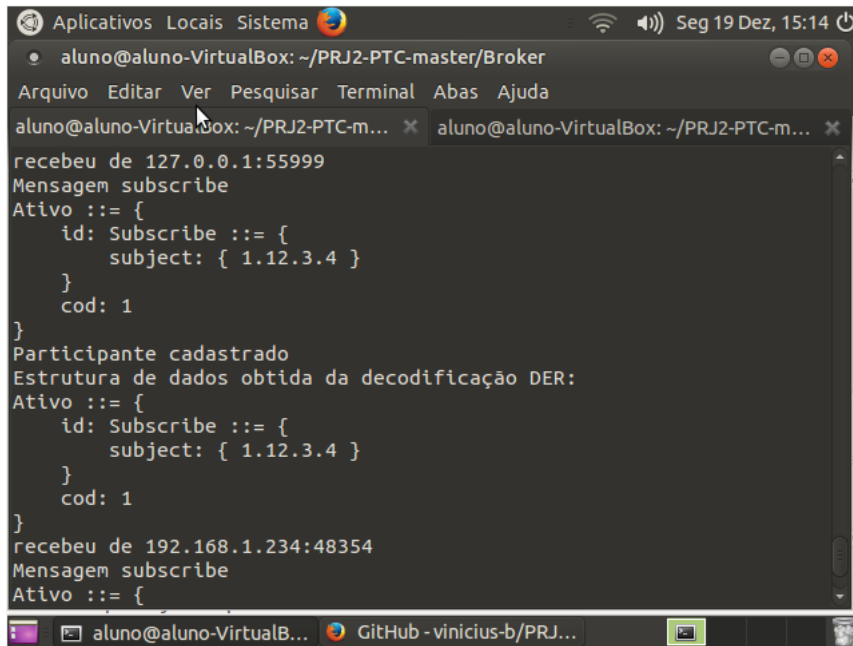


```
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Broker
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
aluno@aluno-VirtualBox:~/PRJ2-PTC-master/Broker$ build/broker
Nova conexão: 127.0.0.1:55999
Nova conexão: 192.168.1.234:48354
```

Figura 2 – Conexões realizadas no Broker

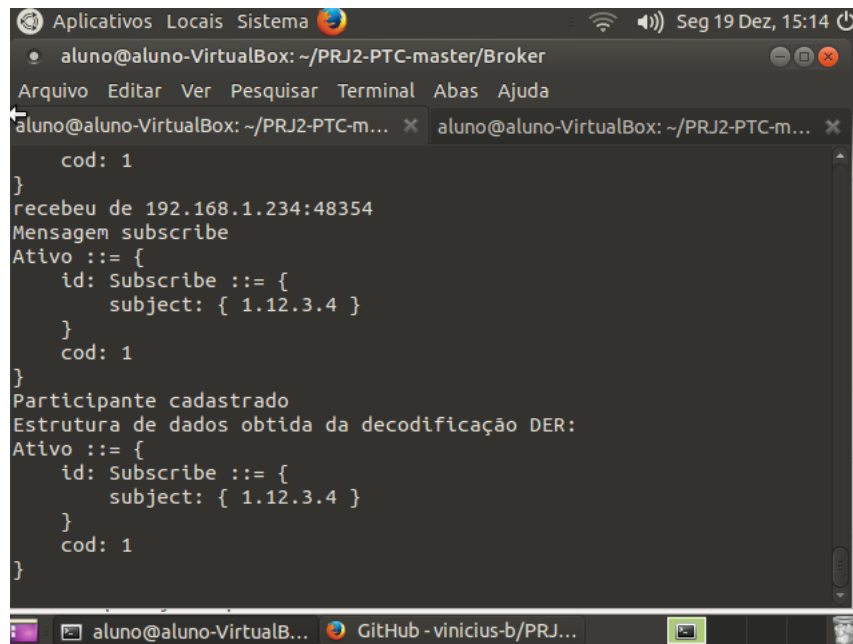
5.2 Teste 2 - Subscribe de Participantes

Depois de conectados os assinantes solicitam inscrição em um assunto através do menu de opções. O Broker então recebe as seguintes mensagens e os insere na lista de participantes.



```
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Broker
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
recebeu de 127.0.0.1:55999
Mensagem subscribe
Ativo ::= {
  id: Subscribe ::= {
    subject: { 1.12.3.4 }
  }
  cod: 1
}
Participante cadastrado
Estrutura de dados obtida da decodificação DER:
Ativo ::= {
  id: Subscribe ::= {
    subject: { 1.12.3.4 }
  }
  cod: 1
}
recebeu de 192.168.1.234:48354
Mensagem subscribe
Ativo ::= {
```

Figura 3 – Subscribe do Participante localizado na máquina local

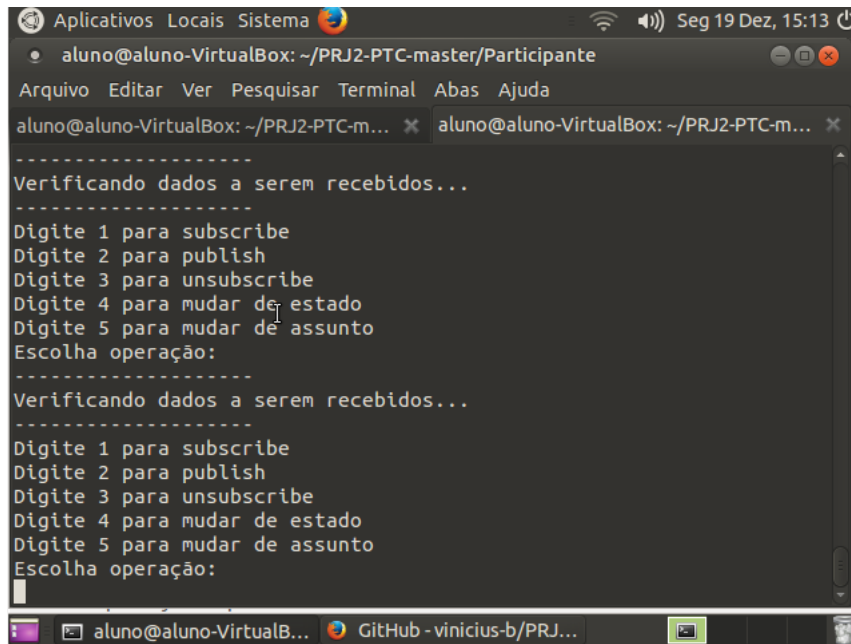


```
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Broker
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
cod: 1
}
recebeu de 192.168.1.234:48354
Mensagem subscribe
Ativo ::= {
  id: Subscribe ::= {
    subject: { 1.12.3.4 }
  }
  cod: 1
}
Participante cadastrado
Estrutura de dados obtida da decodificação DER:
Ativo ::= {
  id: Subscribe ::= {
    subject: { 1.12.3.4 }
  }
  cod: 1
}
}
```

Figura 4 – Subscribe do Participante localizado em outra máquina

5.3 Teste 3 - Menu de opções e recepção de dados

Na imagem abaixo observamos o comportamento do participante. Neste protocolo existe uma janela de 1 segundo em que cada participante verifica a existência de algum dado a ser recebido, caso não exista, um menu de opções aparece durante 5 segundos.

A terminal window titled 'aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Participante'. The window shows a menu with five options: 'Digite 1 para subscribe', 'Digite 2 para publish', 'Digite 3 para unsubscribe', 'Digite 4 para mudar de estado', and 'Digite 5 para mudar de assunto'. Below the menu, it says 'Escolha operação:'. The window is part of a desktop environment with a taskbar at the bottom showing icons for 'aluno@aluno-VirtualB...', 'GitHub - vinicius-b/PRJ...', and a file manager.

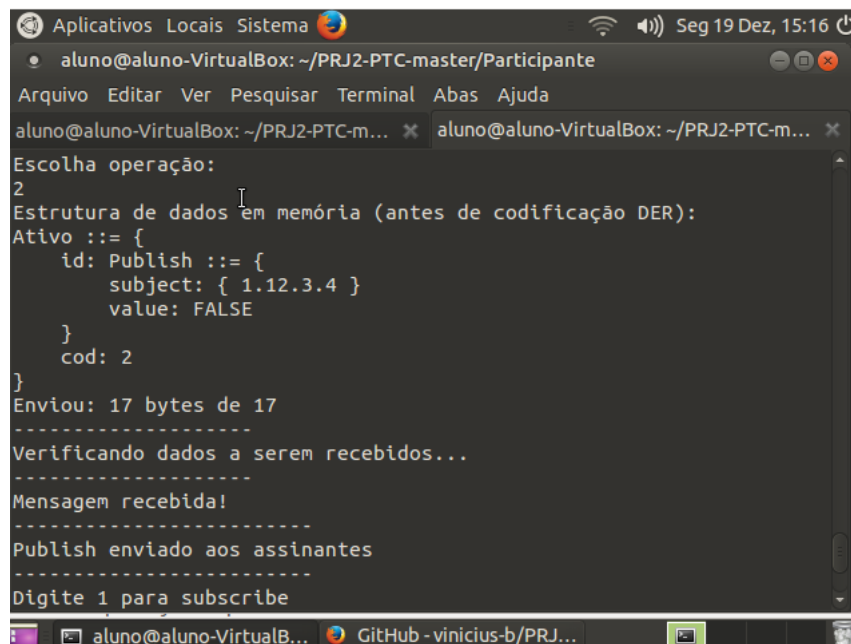
```
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Participante
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
-----
Verificando dados a serem recebidos...
-----
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
Digite 4 para mudar de estado
Digite 5 para mudar de assunto
Escolha operação:
-----
Verificando dados a serem recebidos...
-----
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
Digite 4 para mudar de estado
Digite 5 para mudar de assunto
Escolha operação:

```

Figura 5 – Comportamento de um participante

5.4 Teste 4 - Publish

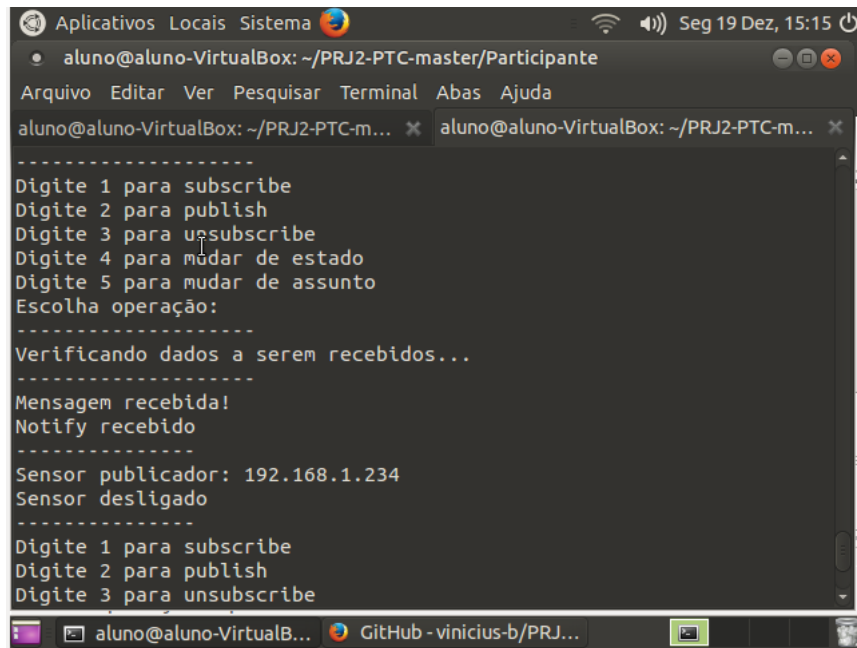
Em nosso protocolo uma mensagem de Publish possui um comportamento bem definido. O nó publicador envia um Publish e recebe uma mensagem indicando que sua mensagem foi enviada. Quando o Broker recebe o publish, ele notifica os demais nós assinantes daquele assunto com o estado e IP do nó publicador. Podemos ver nas imagens abaixo:

A terminal window titled 'aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Participante'. The window shows the user has selected option '2' (publish). It displays the data structure in memory: 'Ativo ::= { id: Publish ::= { subject: { 1.12.3.4 } value: FALSE } cod: 2 }'. It then shows 'Enviou: 17 bytes de 17', followed by 'Verificando dados a serem recebidos...', 'Mensagem recebida!', 'Publish enviado aos assinantes', and finally 'Digite 1 para subscribe'. The window is part of a desktop environment with a taskbar at the bottom showing icons for 'aluno@aluno-VirtualB...', 'GitHub - vinicius-b/PRJ...', and a file manager.

```
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Participante
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
Escolha operação:
2
Estrutura de dados em memória (antes de codificação DER):
Ativo ::= {
  id: Publish ::= {
    subject: { 1.12.3.4 }
    value: FALSE
  }
  cod: 2
}
Enviou: 17 bytes de 17
-----
Verificando dados a serem recebidos...
-----
Mensagem recebida!
-----
Publish enviado aos assinantes
-----
Digite 1 para subscribe

```

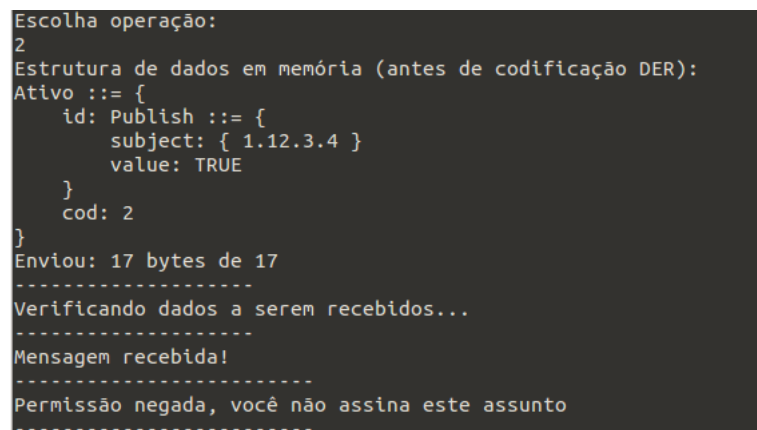
Figura 6 – Resultado de um Publish no nó publicador



```
Aplicativos Locais Sistema
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Participante
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
-----
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
Digite 4 para mudar de estado
Digite 5 para mudar de assunto
Escolha operação:
-----
Verificando dados a serem recebidos...
-----
Mensagem recebida!
Notify recebido
-----
Sensor publicador: 192.168.1.234
Sensor desligado
-----
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
```

Figura 7 – Notify recebido pelos demais nós

É importante ressaltar que quando um nó tenta enviar um Publish para um assunto no qual ele não está inscrito, o Broker envia uma mensagem indicando ao mesmo que ele não assina o assunto, como mostrado abaixo:

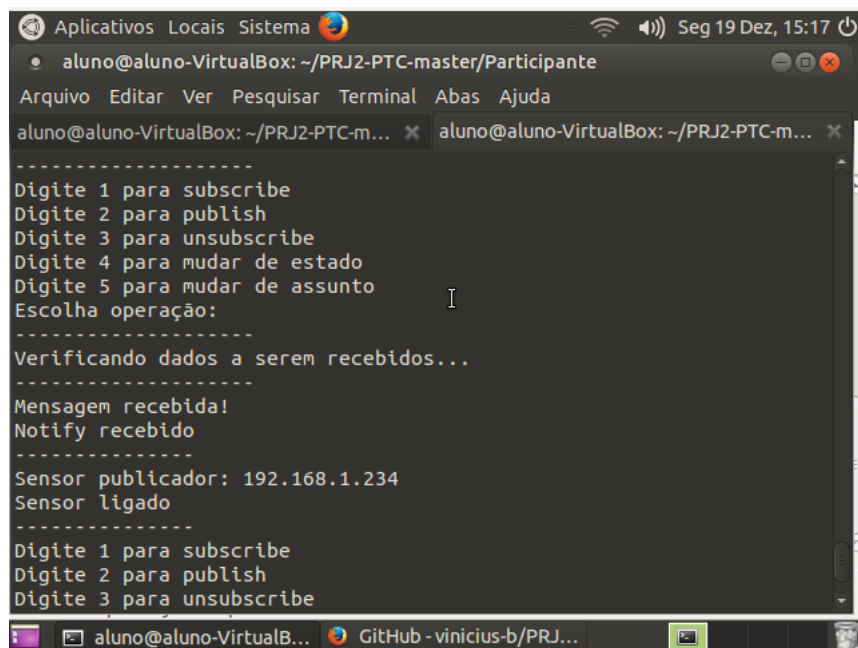


```
Escolha operação:
2
Estrutura de dados em memória (antes de codificação DER):
Ativo ::= {
  id: Publish ::= {
    subject: { 1.12.3.4 }
    value: TRUE
  }
  cod: 2
}
Enviou: 17 bytes de 17
-----
Verificando dados a serem recebidos...
-----
Mensagem recebida!
-----
Permissão negada, você não assina este assunto
-----
```

Figura 8 – Publish negado

5.5 Teste 5 - Mudança de estado

Existe uma opção no menu do participante que habilita a troca de estado do mesmo, sendo assim depois dessa mudança pode-se dar um publish com a situação atualizada do sensor, como mostrado abaixo.

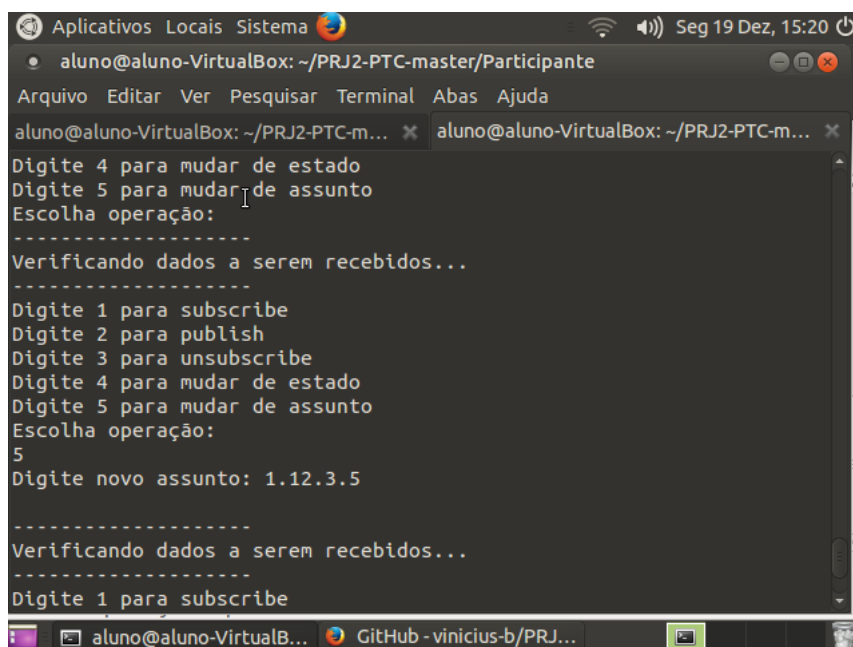


```
Aplicativos Locais Sistema
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Participante
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
-----
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
Digite 4 para mudar de estado
Digite 5 para mudar de assunto
Escolha operação:
-----
Verificando dados a serem recebidos...
-----
Mensagem recebida!
Notify recebido
-----
Sensor publicador: 192.168.1.234
Sensor ligado
-----
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
```

Figura 9 – Mudança de estado de um nó

5.6 Teste 6 - Troca de assunto

O menu do participante também possibilita a troca de assunto para um outro que esteja previamente cadastrado como assunto válido, como mostrado na figura abaixo:



```
Aplicativos Locais Sistema
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Participante
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
-----
Digite 4 para mudar de estado
Digite 5 para mudar de assunto
Escolha operação:
-----
Verificando dados a serem recebidos...
-----
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
Digite 4 para mudar de estado
Digite 5 para mudar de assunto
Escolha operação:
5
Digite novo assunto: 1.12.3.5
-----
Verificando dados a serem recebidos...
-----
Digite 1 para subscribe
```

Figura 10 – Mudança de assunto

Porém se o nó tentar mudar para um assunto que não está previamente cadastrado ele recebe uma mensagem indicando a invalidade da ação:


```

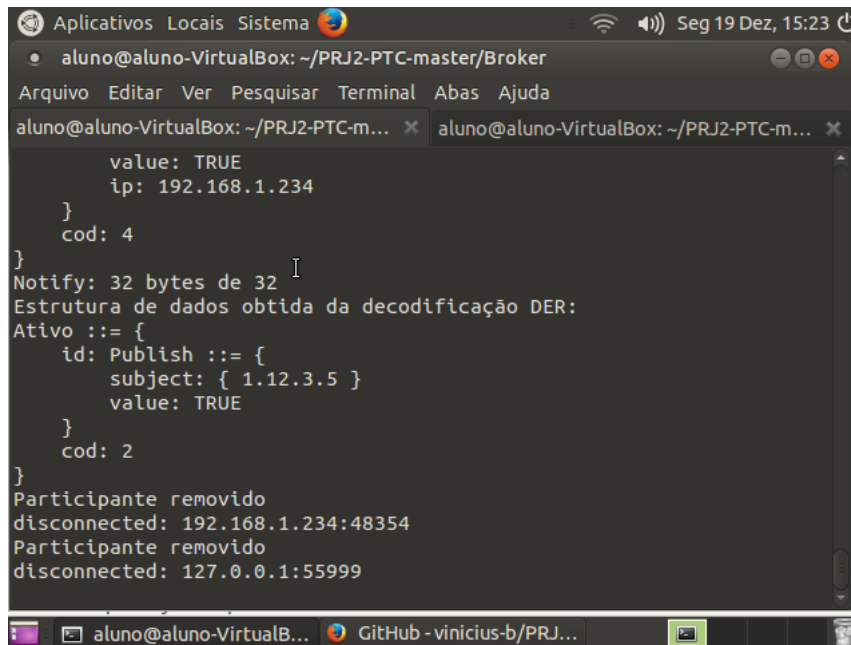
Digite 1 para subscribe
Digite 2 para publish
Digite 3 para unsubscribe
Digite 4 para mudar de estado
Digite 5 para mudar de assunto
Escolha operação:
5
Digite novo assunto: 1.12.3.12
Assunto inválido

```

Figura 11 – Assunto inválido

5.7 Teste 7 - Desconexão dos participantes

Quando uma desconexão acontece o participante é eliminado da lista de participantes cadastrados num determinado assunto. Isso acontece no Broker, que exibe uma mensagem confirmando a exclusão dos mesmos, como mostrado na figura abaixo:



```

Aplicativos Locais Sistema
aluno@aluno-VirtualBox: ~/PRJ2-PTC-master/Broker
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x aluno@aluno-VirtualBox: ~/PRJ2-PTC-m... x
    value: TRUE
    ip: 192.168.1.234
  }
  cod: 4
}
Notify: 32 bytes de 32
Estrutura de dados obtida da decodificação DER:
Ativo ::= {
  id: Publish ::= {
    subject: { 1.12.3.5 }
    value: TRUE
  }
  cod: 2
}
Participante removido
disconnected: 192.168.1.234:48354
Participante removido
disconnected: 127.0.0.1:55999

```

Figura 12 – Desconexão de participantes

6 Limitações do Protocolo

A limitação mais evidente de nosso protocolo é a questão de sincronização, no Participante, entre tempo de recepção e de realização de ação. Em nossa implementação o participante possui janelas bem definidas de tempo para realizar ambas ações, sendo de 1 segundo para receber dados e 5 segundos para escolher a operação a ser feita. Em casos extremos esta limitação de janelas de tempo pode gerar erros.

Podemos citar como limitação também a questão de mudança de assunto. Para que um participante assine um outro assunto ele precisa "mudar de assunto" e em seguida realizar um Subscribe, sendo que a partir desse momento ele só envia Publishes para o novo assunto, mesmo estando inscrito em dois. Para enviar ao assunto anterior o participante precisa digitar a opção "mudar de assunto" no seu menu, retornando ao assunto anterior.

Nosso protocolo também faz diferenciação apenas por endereço de IPs, ou seja, IPs iguais não recebem Notify do Broker. Isso é um problema em casos reais onde sensores podem estar ligados a um switch e são reconhecidos com o mesmo IP para rede externa, somente diferenciando a porta.

7 Apêndice A - Instruções de Uso

Nesta sessão assume-se que o leitor é familiarizado com programação em c++ e operações básicas do sistema operacional Linux, e da solução apresentada neste documento para um protocolo de comunicação sem fio.

Assim sendo, será exemplificado uma situação de teste, que deverá ser repetida para uma futura execução do protocolo.

7.1 Montagem do cenário

Para realizar o teste deve-se ter ao menos dois computadores ou máquinas virtuais com IPs diferentes. No teste feito para esse relatório utilizamos duas máquinas virtuais.

Após fazer o download da API localizada em <https://github.com/vinicius-b/PRJ2-PTC> deve-se descompactar os arquivos em um diretório e fazer as seguintes operações:

7.2 Broker

Deve-se entrar na pasta broker e realizar os seguintes comandos:

```
$ rm -rf build
```

```
$ sudo ./install.sh
```

```
$ make build
```

```
$ make all
```

Para executar o Broker deve-se então digitar:

```
$ build/broker
```

A partir desse momento o Broker entra em estado Idle, aguardando por conexões como mostrado abaixo:

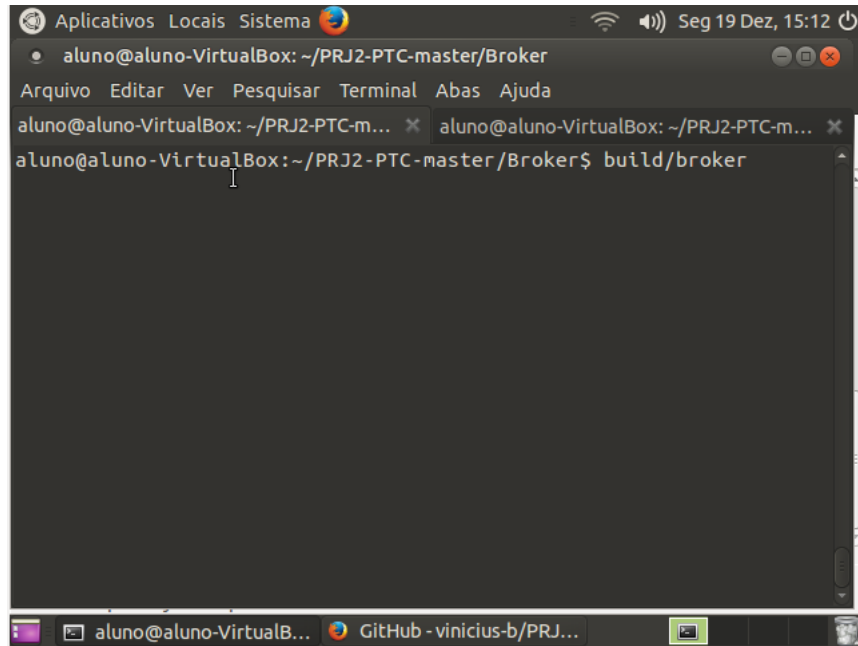


Figura 13 – Broker aguardando conexões

7.3 Participante

Deve-se entrar na pasta Participante e realizar os seguintes comandos:

```
$ rm -rf build
```

```
$ sudo ./install.sh
```

```
$ make build
```

```
$ make all
```

Para executar o Participante deve-se então digitar:

```
$ build/part
```

A partir desse momento o participante assume o comportamento descrito nas sessões anteriores, alternando entre menu de opções e espera por recepção de mensagens.

Referências

- [1] “Projeto de protocolos IFSC - SJ.” <http://wiki.sj.ifsc.edu.br/wiki/index.php/PTC-2016-2>. Acessado: 2016-11-20. 1
- [2] “Publish/Subscribe.” <https://msdn.microsoft.com/en-us/library/ff649664.aspx>. Acessado: 2016-12-10. 1
- [3] “Introduction to ASN.1.” <http://www.itu.int/en/ITU-T/asn1/Pages/introduction.aspx>. Acessado: 2016-12-10. 2
- [4] “Compilador ASN1.” http://wiki.sj.ifsc.edu.br/wiki/index.php/Compilador_ASN1. Acessado: 2016-12-10. 2