UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
CURSO DE ENGENHARIA MECATRÔNICA

FELIPE WILLIAN FRAPORTI

SEGURANÇA E ARMAZENAMENTO PERSISTENTE DE DADOS PARA INTERNET DAS COISAS: AVALIAÇÃO DE APLICAÇÃO NO RESTAURANTE UNIVERSITÁRIO DA UFSC

Joinville
2017

FELIPE WILLIAN FRAPORTI

SEGURANÇA E ARMAZENAMENTO PERSISTENTE DE DADOS PARA INTERNET DAS COISAS: AVALIAÇÃO DE APLICAÇÃO NO RESTAURANTE UNIVERSITÁRIO DA UFSC

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia Mecatrônica no curso de Engenharia Mecatrônica, da Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Orientador: Prof. Dr. Anderson Wedderhof Spengler

Joinville
2017

FELIPE WILLIAN FRAPORTI

SEGURANÇA E ARMAZENAMENTO PERSISTENTE DE DADOS PARA INTERNET DAS COISAS: AVALIAÇÃO DE APLICAÇÃO NO RESTAURANTE UNIVERSITÁRIO DA UFSC

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do título de Bacharel em Engenharia Mecatrônica, na Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Joinville, 30 de Novembro de 2017.

Prof. Dr. Diego Greff
Coordenador do Curso

Banca examinadora:

Prof. Dr. Anderson Wedderhof Spengler
Universidade Federal de Santa Catarina
Orientador

Prof. Dr. Gian Ricardo Berkenbrock
Universidade Federal de Santa Catarina

Prof. Dr. Rodrigo Moreira Bacurau
Universidade de Campinas

RESUMO

Motivado pela otimização e sistematização do controle de acesso do Restaurante Universitário da Universidade Federal de Santa Catarina, este trabalho tem como objetivo final, criar um protótipo que pode ser utilizado para autenticação e liberação de acesso de usuários no restaurante. Inicialmente é feita uma análise do problema e dos requisitos necessários para criação de uma possível solução e então, uma metodologia para execução dos testes de validação de requisitos é desenvolvida. Os testes executados têm como objetivo avaliar a viabilidade da solução nos seguintes aspectos: tempo de resposta do sistema, capacidade de armazenamento de informações e segurança dos dados de usuário. Como resultados, foi verificado que a implementação de tal solução seria viável tanto no viés técnico quanto financeiro.

Palavras-chave: internet das coisas, restaurante universitário, segurança dos dados.

ABSTRACT

Motivated by the optimization and systematization of the access control of the University Restaurant at the Federal University of Santa Catarina, this work's final goal is to create a prototype that can be used to authenticate and authorize user's access to the restaurant. Initially, an analysis of the problem and the necessary requirements for the creation of a solution is made and then, a methodology for the execution of the requirements validation tests is developed. The aim of the tests that are performed is to evaluate the viability of the solution in the following scope: system response time, information storage capacity and user data security. As a result, it was verified that the implementation of a solution would be viable both in technical and financial bias.

Keywords: internet of things, university restaurant, data security.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CI Circuito Integrado

eMote EPOSMote III

HTTP HyperText Transfer Protocol

HTTPS HyperText Transfer Protocol Secure

ICPEdu Infraestrutura de Chaves Públicas para Ensino e Pesquisa

IoT Internet of Things

IP Internet Protocol

ITU International Telecommunication Union

LISHA Laboratório de Integração Software/Hardware

RIFFS Reverse Indirect Flash File System

RU Restaurante Universitário

TCP Transmission Control Protocol

SUMÁRIO

1 INTRODUÇÃO

De acordo com Nordrum (2016) ao citar a previsão da empresa Gartner, que realiza pesquisas e análises na área de tecnologia estima-se que, em 2016, haviam 6,4 bilhões de dispositivos conectados à internet. Esta estimativa não leva em conta celulares, computadores e tablets. Gartner ainda prevê que em 2020 haverá aproximadamente 20,8 bilhões de dispositivos conectados. Se as previsões se mostrarem corretas, é seguro afirmar que a Internet das Coisas (IoT) está crescendo rapidamente. Este crescimento permitirá que muitos processos se tornem cada vez mais eficientes, simplificando as etapas envolvidas e possibilitando que dados coletados sejam analisados de forma mais sistemática, facilitando a tomada de decisões por líderes do segmento que contam com esta tecnologia.

Apesar de um futuro promissor, muitos desafios tecnológicos e de engenharia irão surgir. Alguns exemplos destes desafios são: a segurança dos dados transmitidos, fornecimento de energia para os bilhões de sensores e atuadores conectados, fornecimento de banda para transmissão de tantos dados e, por fim, a gestão e análise de todos os dados recebidos.

Se estes desafios forem superados, um mundo de possibilidades é aberto com aplicações que podem aumentar a eficiência produtiva e a qualidade de vida humana. Imagine a possibilidade de realização de um sensoriamento das lavouras para monitoramento da qualidade do solo, ou um acompanhamento constante de saúde do usuário para avaliação e exames médicos mais precisos, ou até mesmo a realização de uma conexão entre a indústria e o varejo para aumento da produtividade.

Este trabalho expõe uma solução para o projeto de modernização do Restaurante Universitário (RU) da Universidade Federal de Santa Catarina (UFSC) utilizando o conceito de IoT. Para que esta solução seja reconhecida, um sistema capaz de garantir segurança dos dados transmitidos contando com a capacidade de armazenamento persistente de informações deve ser concebido.

Como foi citado anteriormente, um dos impasses que deve ser superado é a garantia de proteção dos dados coletados. O protótipo deste trabalho, utiliza técnicas padrões para garantir a segurança dos dados transmitidos no projeto de modernização do RU e, dado que as restrições de projeto para sistemas embarcados se difere de sistemas computacionais, há dificuldades claras na implementação destes mecanismos de segurança em sistemas como este. Estas dificuldades estão relacionadas às limitações de memória dos sistemas embarcado, bem como sua capacidade de processamento limitada.

Outro ponto importante que deve ser avaliado é o armazenamento persistente de dados. Verifica-se que há uma limitação se comparado a outros sistemas compu-

tacionais. Esta limitação se deve não somente ao espaço disponível mas também às características construtivas das memórias não voláteis. Muitos dos microcontroladores utilizam a memória Flash como memória principal de armazenamento persistente. Esta memória é dividida em setores e possui uma vida útil limitada a quantidade de apagamentos de cada setor. Por conta disso, além da gestão do espaço na memória, a posição de escrita de novos dados deve ser avaliada.

Com a associação da conectividade e do armazenamento persistente de dados, o protótipo para ser utilizado na modernização do RU foi desenvolvido. A utilização deste protótipo seria bastante proveitosa, visto que atualmente não há um controle de acesso eficiente nos RUs da UFSC.

Para acessar o restaurante, apenas é verificado se o aluno possui uma carteirinha de identificação e um passe de acesso ao RU. Esta forma de controle pode não ser tão eficaz, pois não se sabe se a carteirinha apresentada é válida ou se o usuário ainda é membro da comunidade acadêmica e, apesar de não haver dados concretos de quantas pessoas fazem uso do restaurante sem autorização, ao especular que 5% das refeições servidas são para pessoas que não poderiam utilizar este serviço, apenas no RU Campus Trindade, houve em torno de 73 mil refeições servidas irregularmente ao longo do ano de 2016. Avaliando que cada refeição custa em aproximadamente R$ 9,00, o prejuízo em 2016 teria sido de R$ 657.000,00.

1.1 Objetivos

1.1.1 Objetivo Geral

Este trabalho busca avaliar estes dois aspectos de sistemas embarcados (transmissão de dados segura e armazenamento persistente de dados), e sua utilização no projeto de modernização do restaurante. Para que o objetivo seja alcançado, primeiramente é desenvolvido um *firmware* que seja capaz de realizar uma conexão segura com servidores na internet e então, um sistema de arquivos para gerenciamento da memória não volátil do microcontrolador é implementado.

Consequentemente, é possível realizar o julgamento da utilização destas tecnologias para implantação no restaurante. Com isso, o objetivo geral do trabalho consiste em desenvolver um protótipo funcional para ponderação e avaliação da utilização destes componentes na modernização do controle de acesso do RU.

1.1.2 Objetivos Específicos

Dentro deste escopo geral, há objetivos específicos que devem ser observados na validação dos experimentos. De forma genérica, estes objetivos consistem em verificar a capacidade de armazenamento do dispositivo, o tempo de execução da

tarefa de autenticação do usuário e a segurança durante a transmissão dos dados. Assim sendo, os objetivos específicos do trabalho podem ser enumerados como:

- Implementação de um sistema de arquivos responsável por gerenciar os dados armazenados no microcontrolador;
- Implementação de um *firmware* responsável por gerir a conectividade e garantir a segurança dos dados transmitidos;
- Desenvolvimento de um protótipo capaz de simular a utilização real do sistema para requisitos de projeto estabelecidos;
- Avaliação do tempo de resposta do sistema desde a solicitação de entrada no restaurante até a resposta recebida. Esta avaliação é feita para os casos *online* e *offline*;
- Análise da ocupação dos dados armazenados na memória flash.

2 CONTEXTUALIZAÇÃO

Em setembro de 1965 foram iniciadas as atividades do restaurante universitário da UFSC e, desde sua abertura, já foram servidas milhões de refeições. Com estas informações, já é possível reconhecer a importância e o tradicionalismo deste serviço.

Se forem analisados os dados disponíveis em Custo... (2016), constata-se que a mobilização de recursos para o fornecimento das refeições é bastante alta e, apesar da importância e da alta taxa de utilização deste serviço, é possível observar que os procedimentos necessários para que se faça uso dessa solução, são obsoletos. Com isso em mente, para que o problema seja solucionado, é importante fazer uma avaliação das etapas e processos necessários para que o usuário possa ter acesso ao restaurante. É importante notar que cada um dos restaurantes da UFSC possuem métodos diferentes para aquisição dos passes, e as etapas apresentadas aqui servem apenas como ilustração da ineficácia do processo, se referindo ao restaurante do campus de Joinville:

1. Emissão de GRU para realização do pagamento
2. Efetivação do pagamento no Banco do Brasil
3. Aquisição dos passes com o servidor técnico-administrativo da Universidade
4. Acesso ao restaurante por meio da entrega dos passes e apresentação da carteirinha universitária

Pode ser reconhecido que cada etapa do processo possui uma chance de erro e, ao combinar todas estas etapas, a chance de erro é acumulada, prejudicando a confiabilidade geral do sistema.

A primeira parte do processo consiste em emitir a Guia de Recolhimento da União (GRU) para a compra dos passes. Esta guia é emitida em nome de uma pessoa física associando a guia ao seu CPF, seu valor está relacionado à quantidade de passes que serão comprados. Apesar de ser um processo simples, há algumas fontes de erros e alguns pontos de ineficiência, como exemplo: é possível que o CPF do usuário seja inserido incorretamente, fazendo com que a compra seja descartada.

Após emitir a GRU, é necessário realizar a efetivação do pagamento, que deve ser feita exclusivamente no Banco do Brasil, o comprovante impresso do pagamento deve ser guardado para que seja apresentado ao responsável pela troca dos passes. Como este pagamento só pode ser realizado no Banco do Brasil, o usuário deverá possuir uma conta neste banco ou aguardar atendimento no caixa para que possa efetivar o pagamento, amplificando o tempo gasto no processo.

Ao fim destas etapas, o comprovante de pagamento da guia deve ser apresentado ao servidor técnico-administrativo responsável pela entrega dos passes, este servidor deve conferir os dados informados pelo usuário na emissão da guia, validando assim o *status* desta pessoa dentro da universidade. Apesar desse ser o procedimento correto, pode ser que exista um erro nessa fase por conta de falhas na verificação dos dados informados no momento de emissão da GRU ou até por erros cometidos pelo servidor no momento da validação como, por exemplo, não verificação da situação do usuário perante a universidade.

Por fim, a entrada no restaurante pode ser feita ao apresentar estes passes e a carteirinha de identificação da universidade para a pessoa responsável pelo RU. Teoricamente, esta pessoa também deveria fazer uma validação do usuário que está tentando entrar no restaurante conferindo os dados apresentados e validando sua situação na UFSC, porém, isto esporadicamente acontece dado o alto fluxo de pessoas no restaurante e o pouco tempo hábil para validação de todos usuários.

Com estes problemas apresentados, inicia-se a discussão de uma solução que poderia minimizar estes erros. É possível notar que os processos executados consistem na validação de dados e verificação de usuários. Desta forma, um sistema automatizado capaz de executar estas tarefas poderia minimizar parte dos problemas retratados. Para que esta solução seja efetuada, pode ser desenvolvido uma catraca conectada a internet juntamente com um sistema de armazenamento persistente de dados, utilizando uma conexão com a internet para validar o usuário em tempo real e verificar se este usuário pode acessar o restaurante. Com esta premissa, o armazenamento de dados pode ser utilizado para criação de uma redundância no sistema em caso de falhas de conexão com a internet.

Apesar desta catraca ser uma possível solução aos problemas gerais de acesso ao RU, há alguns problemas técnicos que podem ser produzidos por tal solução. Tanto a conexão com a internet quanto o sistema de armazenamento de dados possuem limitações e obstáculos a serem superados.

2.1 Problemas com a Memória Flash

Memórias flash são dispositivos de armazenamento não volátil, e são utilizadas para as mais diversas aplicações. Essas memórias são construídas a partir de uma tecnologia que oferece uma boa velocidade de leitura e escrita, aliando esta performance com uma alta confiabilidade e a característica de armazenamento persistente dos dados. Em contrapartida, esta tecnologia possui uma limitação quanto a vida útil e a densidade da memória que são relativamente reduzidas se comparadas aos outros tipos de tecnologia com as mesmas funcionalidades.

As limitações das memórias flash devem ser levadas em consideração no

desenvolvimento da solução apresentada neste trabalho. Como o espaço disponível para armazenamento é limitado, a forma em que os dados são escritos deve ser avaliada para que haja um melhor aproveitamento do baixo espaço disponível.

2.2 Problemas de Conectividade

A fundação da troca de dados na rede de computadores tem como base o protocolo HTTP. Este protocolo é inseguro e está sujeito à espionagem ou até alteração da informação transmitida, visto que os dados passam por inúmeros nós antes de chegar ao destino final. Para superar esta dificuldade deve ser adicionada uma camada de criptografia à pilha TCP/IP que realiza a transmissão dos pacotes de informação, esta camada deve ser capaz de prover um canal seguro de troca de informações entre dois pontos finais.

Apesar do problema de segurança ser resolvido com a implantação desta nova camada sobre a pilha TCP/IP, outras dificuldades podem vir a aparecer. Posto que muitos microcontroladores possuem uma memória reduzida e uma frequência de processamento relativamente baixa, a execução dos algoritmos de criptografia utilizados pela camada de segurança é limitada à capacidade de processamento e armazenamento dinâmico de dados do microcontrolador utilizado.

Mais adiante, é importante examinar como os elementos constituintes deste sistema são identificados e validados. Normalmente, isso é feito através de certificados digitais, que são autenticados por alguma instituição verificadora. Estes certificados são utilizados pela camada de segurança para validação do ponto de destino dos dados transmitidos. A dificuldade que aparece ao utilizar essa metodologia está relacionada ao tamanho dos certificados e sua forma armazenamento dentro dos microcontroladores, dada a capacidade de armazenamento limitada que estes dispositivos possuem. Além do mais, para garantir a segurança e consistência do sistema durante toda sua vida útil é importante que estes certificados sejam atualizados periodicamente, trazendo outros problemas relacionados com a geração e o armazenamento desses certificados.

3 REVISÃO TEÓRICA

Este capítulo apresenta um estudo das tecnologias que serão utilizadas na solução do problema. O sistema desenvolvido possui duas partes principais: conectividade com um servidor, e armazenamento persistente dos dados recebidos.

Para que seja o sistema funcione de forma adequada, é necessário que os usuários possuam uma validação de acesso em tempo real. Desta forma, deve haver uma confirmação do estado do usuário perante ao restaurante, esta validação é feita através de uma conexão com o servidor que é capaz de consentir o acesso deste usuário.

O primeiro componente do sistema a ser analisado é o acesso à internet visto que, a garantia de segurança dos dados transmitidos e velocidade de transmissão são características pretendidas neste projeto.

3.1 Conectividade

A internet que conhecemos hoje conecta centenas de milhões de dispositivos por todo o mundo com a finalidade de trocar de informações de forma rápida e consistente, de acordo com *International Telecommunication Union* (ITU), em 2016 haviam mais de 3 bilhões de pessoas conectadas a internet (ITU, 2017).

Para que esta rede cumpra seu objetivo, todos os membros devem estar interligados por meio de conexões, e a informação é transmitida de uma parte a outra através da comutação de pacotes. O sistema é dividido em vários componentes que são responsáveis por transmitir os diversos pacotes de informação entre um ponto e outro.

A fim de que a informação seja transmitida de forma estável e coerente entre os diversos subsistemas da rede, há dois protocolos que definem o formato em que os pacotes de informação são transmitidos. Estes protocolos são conhecidos como *Transmission Control Protocol* (TCP) e *Internet Protocol* (IP) que, simultaneamente são destacados como protocolo TCP/IP. O protocolo TCP provê uma entrega ordenada, confiável e sem erros de um conjunto de dados, já o protocolo IP especifica o formato dos pacotes de dados transmitidos entre os componentes da rede.

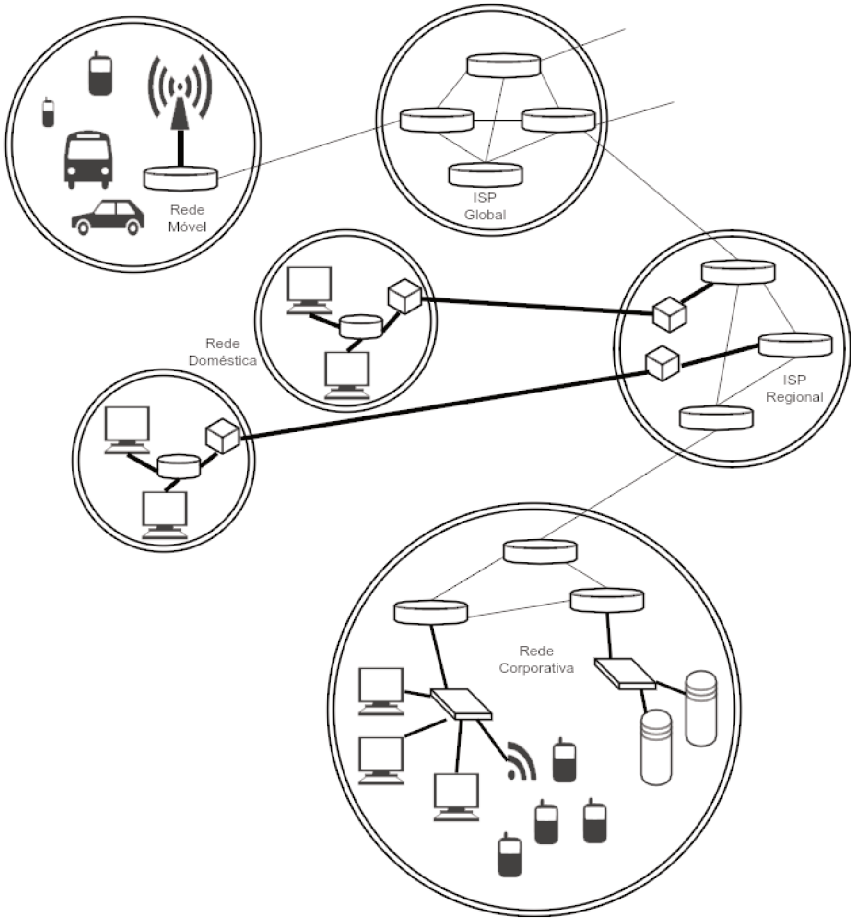
3.1.1 Sistemas Finais e a Borda da Internet

Os componentes de uma rede de computadores e a internet são discutidos em (KUROSE; ROSS, 2013). O autor define como borda da rede o grupo que contém os sistemas finais, incluindo computadores pessoais, servidores e computadores móveis.

Nos últimos anos, mais dispositivos não tradicionais estão sendo conectados à internet, estes sistemas também são conhecidos como hospedeiros pois abrigam aplicações capazes de transmitir informações. Ainda, de acordo com KUROSE; ROSS, os hospedeiros estão divididos em duas categorias: clientes e servidores. Sendo que os clientes normalmente podem ser computadores pessoais e celulares. Já os servidores são máquinas capazes de distribuir conteúdo para uma grande quantidade de usuários. Desta forma, o sistema em estudo neste trabalho é classificado como cliente, dado que há um consumo e geração dados em baixa quantidade, sempre recebendo o conteúdo de um servidor.

A partir disso, é necessário examinar como ocorre o acesso destes componentes à rede. Como ilustrado na Figura 1, os sistemas são conectados fisicamente por meio de um roteador de borda, este é o primeiro roteador encontrado em uma borda de rede e na imagem abaixo demonstra de forma destacada as conexões que fazem parte deste componente.

Figura 1 – Redes de Acesso.



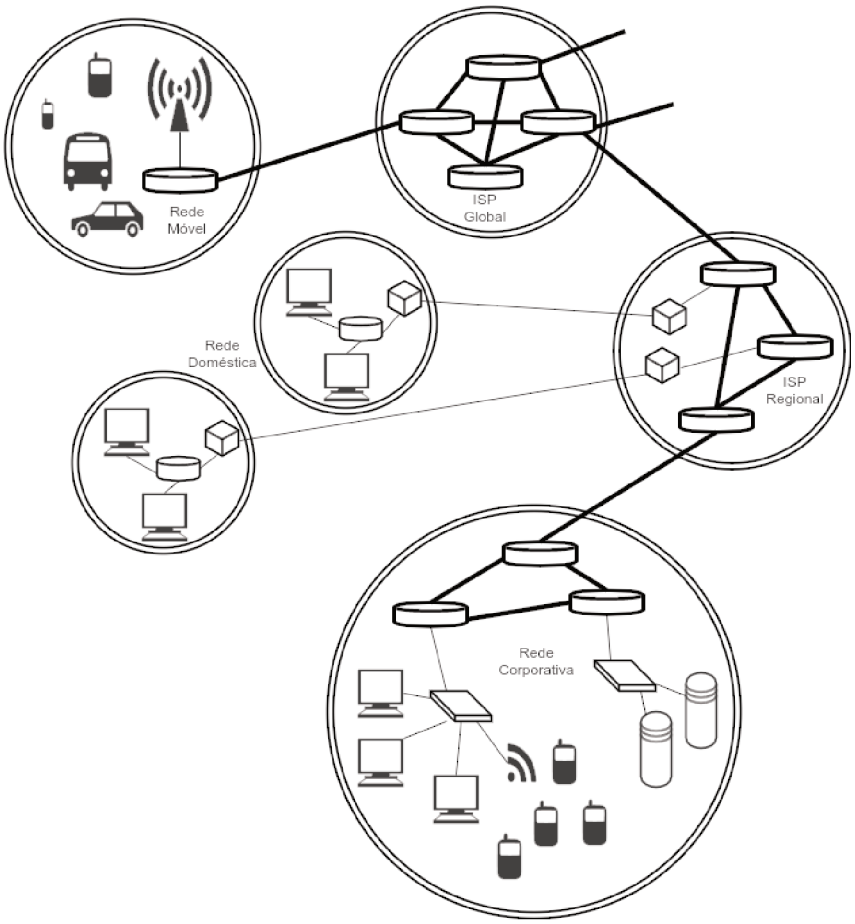
Fonte: Autor (2017)

A conexão entre o roteador de borda e os hospedeiros é normalmente feita através da tecnologia Ethernet ou WiFi. Já a conexão entre o roteador de borda e o núcleo da rede é feita por meio de cabos coaxiais, cabos DSL ou fibra óptica.

3.1.2 Núcleo da Internet

O núcleo da Internet tem como objetivo transmitir dados entre os sistemas finais (hospedeiros) que fazem parte da borda da internet. Este núcleo utiliza comutadores para encaminhar o sinal de um ponto a outro. Desta forma, o núcleo da internet pode ser considerado como sendo a malha que conecta todos os sistemas, a figura 2 destaca como é constituído o núcleo da internet.

Figura 2 – O Núcleo da Internet



Fonte: Autor (2017)

Os dados transmitidos são convertidos em pacotes que contém cabeçalhos e informações de controle de acordo com o protocolo IP. Para qualificar a duração da transmissão dos pacotes de dados é determinado o tempo de transmissão necessário para que o pacote de dados seja conduzido ao próximo roteador. Para avaliar

este tempo, ao considerar uma quantidade **N** de roteadores entre um transmissor e um receptor, sendo **L** o comprimento de cada pacote em bits, e **R** sendo a taxa de transmissão do canal em bits/segundo, é definido que:

$$t_{transmisso} = N\frac{L}{R}segundos \tag{3.1}$$

Além do tempo de transmissão, outros tempos devem ser contabilizados para qualificar a transmissão dos dados. Estes tempos estão relacionados à quantidade de desvios que o dado sofrerá, à sobrecarga da rede e, principalmente, à distância entre os dois sistemas finais, sendo que este último é conhecido como tempo de propagação. A performance de um sistema conectado a internet é diretamente afetada por estes tempos. Ao somar a duração de cada um destes tempos, é possível estimar o tempo total entre a saída de um dado do transmissor e a chegada do dado no receptor:

$$t_{total} = t_{processamento} + t_{fila} + t_{trans} + t_{prop} \tag{3.2}$$

Considera-se t_{fila} da equação 3.2, como sendo o tempo em que o pacote deve aguardar até que seja transmitido ao próximo link da rede, este tempo esta diretamente relacionado à sobrecarga e a quantidade de pacotes que devem ser transmitidos em um mesmo intervalo de tempo.

Além disso, $t_{processamento}$ pode ser considerado desprezível por ter seu valor na ordem dos microssegundos. Este tempo consiste na duração da leitura do cabeçalho do pacote para que o destino do pacote seja determinado e retransmitido.

Por fim, um último atraso poderia ser considerado na estimativa do tempo total de transmissão de um dado. Este atraso pode estar relacionado ao sistema final, levando em conta o tempo de processamento e as políticas de escalonamento deste sistema. Visto que sistemas embarcados normalmente consistem em aplicações dedicadas a uma tarefa específica, este atraso pode ser desconsiderado.

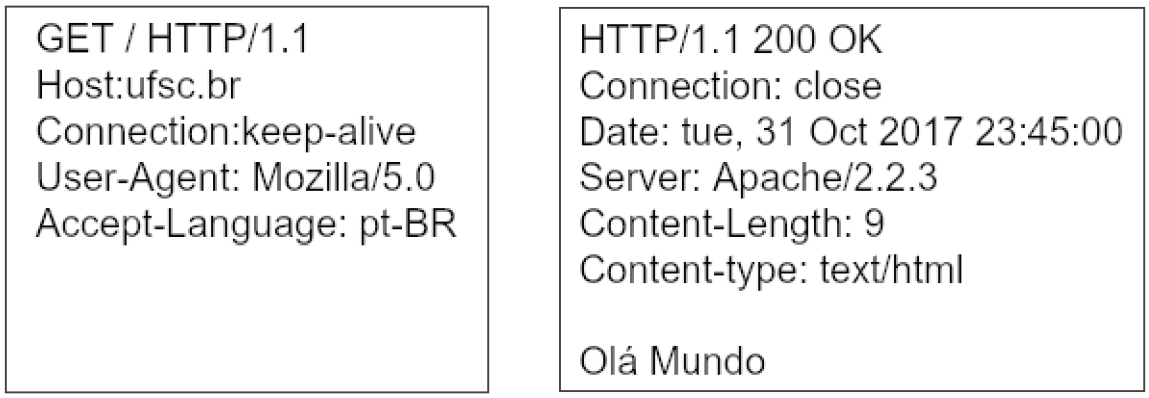
3.1.3 Transmissão Segura

Tendo esta análise em mente, já é possível avaliar que o tempo de transmissão é um ponto crítico do sistema. Também, tendo em vista que os dados transmitidos passam por diversos sistemas intermediários, é possível avaliar que a segurança se torna outro fator bastante relevante e crítico na transmissão de informações.

Desde a saída do dado do transmissor até a chegada do dado no receptor, há inúmeros canais e nós que conduzem a informação a ser transmitida. Se a informação não possuir nenhum encapsulamento de segurança, os dados que estão sendo propagados podem ser entendidos sem maiores dificuldades por qualquer programa de monitoramento de pacotes.

O protocolo *HyperText Transfer Protocol* (HTTP) é um dos protocolos mais

utilizados na internet hoje, este protocolo deve ser executado nos sistemas finais, cliente e hospedeiro. Com o HTTP, a estrutura das mensagens que são trocadas entre os sistemas finais é definida, havendo dois tipos de mensagens para este protocolo, de requisição e de resposta. A figura 3a é um exemplo de mensagem de requisição, já a figura 3b é um exemplo de mensagem de resposta.



(a) Requisição (b) Resposta

Figura 3 – Estruturas do protocolo HTTP

Fonte: Autor (2017)

Ao avaliar esta simples mensagem de requisição, é possível verificar que o conteúdo pode ser naturalmente lido por qualquer pessoa dada sua codificação em ASCII. Desta forma, ao transmitir a mensagem pela rede, a mensagem pode ser interceptada e interpretada facilmente por qualquer intermediário que possa vir a receber este pacote. Da mesma forma, o conteúdo da resposta é facilmente interpretado.

Com esta análise feita, é possível observar que o protocolo HTTP não adiciona segurança a transmissão de dados. De acordo com (HTTP...,), uma nova camada a conexão deve ser adicionada com o intuito de prover uma segurança a aplicações que possuem dados sensíveis. A união do HTTP com esta nova camada de segurança é conhecida como HTTP Secure (HTTPS). O HTTPS é geralmente utilizado para proteção de dados em transações *online* de dados.

3.2 Sistema de Arquivos

Muitas aplicações possuem a necessidade de armazenar uma grande quantidade de dados a longo prazo. Desta forma, o armazenamento de dados na memória virtual do processo em execução é inviabilizado. De acordo com Tanenbaum e Bos (2014), há 3 requisitos que são essenciais para armazenar informações a longo prazo, estes requisitos são:

- Possibilidade de armazenar uma grande quantidade de informações;
- Mesmo que a execução do processo termine, a informação deve permanecer intacta;
- Múltiplos processos devem ter acesso à essas informações.

O componente dos microcontroladores que é capaz de atender a estes requisitos é a memória flash. De acordo com Pereira (2004), esta memória tem sido amplamente utilizada por conta da sua alta densidade, peso leve, pequeno tempo de latência e baixo consumo de potência. Como ainda é uma tecnologia com um preço elevado em relação à outras tecnologias de armazenamento não volátil, sua utilização em computadores pessoais ou sistemas de grande porte é inviável, mas possui alta vantagem em sistemas embarcados.

Por suas características construtivas, as memórias Flash podem manter os dados consistentes por um longo período de tempo, mesmo sem receber nenhum tipo de energização. Porém, esta peculiaridade traz uma limitação para este componente, o número de apagamentos é limitado por conta do desgaste do dispositivo.

Além disso, é importante notar que estas memórias são divididas em setores, estes setores são blocos de dados que possuem um tamanho fixo. A operação de apagamento só pode ser realizada em um setor inteiro de uma só vez. O apagamento consiste em colocar todos os bits do setor em valor lógico "1", esta é a operação mais demorada da flash.

Com estas características em mente, é necessário realizar a implementação de um sistema capaz de abstrair estas particularidades e, ao mesmo tempo, atender os requisitos estabelecidos anteriormente. Desta forma, utiliza-se o conceito de arquivos e sistema de arquivos.

O sistema de arquivos que será utilizado deve ser escolhido levando em conta as singularidades e especificidades da grande maioria dos sistemas embarcados. Os microcontroladores possuem limitações em relação a sua capacidade de processamento e espaço de armazenamento disponível, sem contar que a vida útil das memórias não voláteis também é reduzida se comparada a sistemas computacionais.

Por conta disso, é implementado neste trabalho um sistema de arquivos que tem como proposta melhorar o desempenho dos coletores de lixo de memórias flash, além de economia na atualização das estruturas, processamento e aumento do tempo de vida útil da memória. Este sistema é apresentado por PEREIRA em Pereira (2004) e é intitulado Referse Indirect Flash File System (RIFFS).

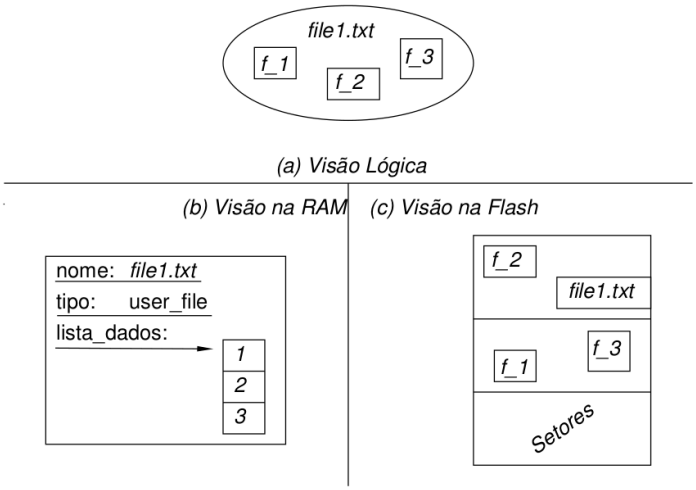
3.2.1 Reverse Indirect Flash File System

A principal motivação da criação do sistema RIFFS, foi simplificar as estruturas clássicas de sistemas de arquivos. O RIFFS implementa uma nova arquitetura no gerenciamento de diretórios chamada de árvore reversa, além de resgatar os conceitos de blocos lógicos de tamanho variado no gerenciamento dos arquivos. De acordo com Pereira (2004), as principais características definidas durante o desenvolvimento deste sistema são:

- Simplicidade das estruturas: ao simplificar as estruturas físicas de armazenamento da flash, foi possível perceber que além de economia de espaço, houve um aumento da vida útil dos setores.
- Arquivos possuindo todas as informações: as informações de entrada de diretório foram agrupadas com o descritor de arquivo.
- Navegabilidade: não há navegabilidade natural em uma árvore reversa, então a árvore montada na memória principal é uma árvore direta.
- Blocos Lógicos: por adotar a estrutura de blocos lógicos de tamanho variável para o armazenamento de dados, o mapeamento de dados é do tipo indexado.
- Fragmentação: não existe fragmentação externa, qualquer espaço pode ser alocado como um bloco de dados.
- Portabilidade: não há uso de bibliotecas externas e dependências, desta forma o sistema de arquivos pode ser compilado para qualquer plataforma.

Para este sistema, pode-se definir um arquivo como um conjunto de dados, que podem ser tanto de controle quanto de usuário. Os arquivos possuem um contexto, uma lista de blocos e um tipo, esta estrutura pode ser visualizada na Figura 4.

Figura 4 – Estrutura de um arquivo no RIFFS referente às visões: (a) lógica, (b) na RAM e (c) na Flash.



Fonte: Pereira (2004)

Na Figura 4 (a) a composição do arquivo no sistema RIFFS é abstraída e representada de forma simplificada, ilustrando o arquivo denominado "file.txt" que contém 3 blocos de dados. Já em 4 (b), é apresentado o formato do arquivo quando carregado na memória RAM. Por fim, em 4 (c) a disposição dos dados fisicamente na memória flash está representada.

Outro ponto importante que deve ser observado nesse sistema de arquivos é a disposição dos dados dentro de cada setor, dado que este componente é bastante relevante na construção do sistema. A Figura 5 demonstra como é feita a distribuição desses dados.

Figura 5 – Arquitetura interna de um setor do RIFFS.

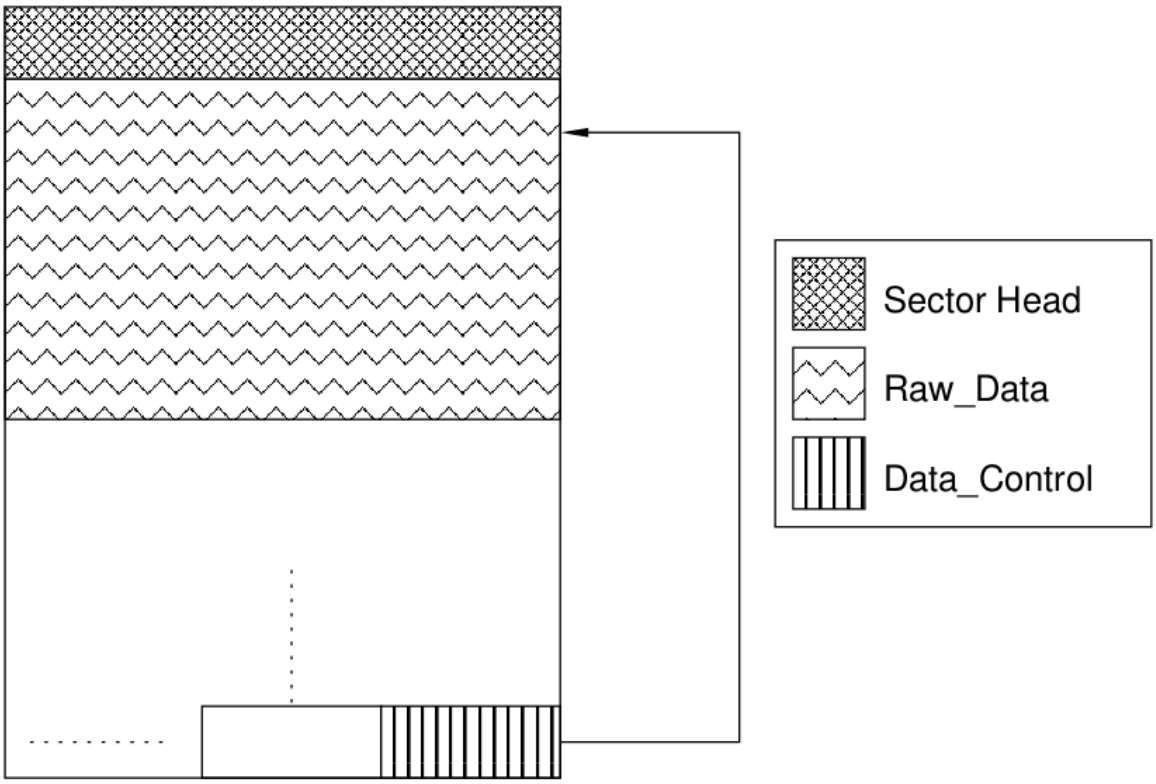


Figura 5.2: Arquitetura do Setor.

Fonte: Pereira (2004)

Cada setor possui um cabeçalho de 12 bytes ("Sector_Head"), este cabeçalho é localizado no endereço inicial de cada setor e é lido sempre que o sistema é iniciado. O cabeçalho possui algumas informações importantes, tais como:

- Número Mágico: define o setor como válido para utilização do sistema de arquivos RIFFS.
- Número de apagamentos: esta informação é bastante importante para que a vida útil da memória seja controlada, e a memória seja igualmente desgastada em todas suas regiões
- Tamanho do setor: com esta informação é possível definir onde a próxima escrita irá ocorrer, e quais são os limites de tamanho de dados que poderão ser inseridos na flash.

Ao final do setor, encontram-se as informações de controle de dados ("Data_Control"). Cada "Data_Control" possui 16 bytes, esta estrutura é adicionada à flash a cada novo conjunto de dados inseridos, ela contém informações importantes sobre a localização e tamanho destes dados. As informações contidas em "Data_Control" são:

- Deslocamento: este campo determina o endereço do dado na memória flash.
- Tamanho: o tamanho das informações inseridas
- Identificador: identificador único do dado
- Tipo: pode ser dado, setor, contexto de entrada ou contexto de log
- Versão: campo utilizado para ordenar os inúmeros pedaços de cada arquivo

Na inicialização do sistema de arquivos, todos os setores são percorridos para que ocorra uma localização de todos "Data_Control" e os arquivo do RIFFS sejam mapeados.

Além das estruturas de cabeçalho do setor e de controle de dados, há também os dados crus ("Raw_Data"). É a informação inserida pelo usuário sem nenhum tipo de modificação. Como o "Data_Control" possui as informações de tamanho e de endereço de início dos dados crus, é possível ler a memória flash e retornar o resultado ao usuário.

4 PROJETO E ANÁLISE DE REQUISITOS

De acordo com (ZU, 2005), o projeto de sistemas inicia com a análise das restrições do problema e o levantamento de requisitos de projeto sendo que, para validação do cumprimento destes requisitos, alguns experimentos de teste devem ser realizados.

Como mencionado anteriormente, o projeto de modernização consiste em catracas conectadas a internet, de forma que seja possível automatizar o controle de acesso de usuários no restaurante universitário. Para o protótipo deste trabalho, quatro pontos devem ser avaliados e estudados: a conexão com a internet deve ser segura, o tempo de resposta deve ser baixo, o sistema deve ter um baixo custo, além de haver uma redundância para garantir sua robustez. Dessa forma, foram estabelecidos os requisitos que estão listados a seguir:

1. Estabelecimento de uma conexão segura para transferência de dados: quando dados são transmitidos da catraca para o servidor ou vice-versa, a interpretação destes dados deve ser infactível caso haja um farejador em algum ponto da rede.
2. Identificação de identidade da catraca ou do servidor: é importante que seja possível identificar o remetente e o destinatário das informações, para garantir que os dados transmitidos são confiáveis.
3. Armazenamento persistente de dados: este armazenamento é útil para garantir uma robustez no sistema em caso de falha da conexão.
4. Baixo tempo de resposta: dado o alto fluxo de usuários no restaurante, o sistema deve possuir um baixo tempo de resposta. O cálculo deste tempo é feito com base nos registros de utilização do restaurante nos anos passados.
5. Baixo custo: o custo do sistema deve ser baixo se comparado a soluções comerciais que oferecem dispositivos similares.

São feitos experimentos que buscam avaliar a viabilidade dos 4 primeiros requisitos e é esperado que ao final da execução dos experimentos, todos os requisitos sejam confirmados como alcançáveis.

4.1 Avaliação de Tempos de Resposta

Este experimento busca validar o requisito 4. Para isso, é realizada a simulação de utilização da catraca durante um período de uso. Para que o ensaio esteja mais

próximo da realidade, são elaboradas avaliações da quantidade de ciclos em que os testes devem ser efetuados e o tempo de resposta total que o sistema deve possuir.

No relatório de produção anual do RU, que está demonstrado em (CUSTO. . . , 2016), pode ser verificado que em 2016 foram servidos 1.491.135 refeições no campus da UFSC na Trindade em Florianópolis. Esta informação corresponde a oferta do serviço entre os meses de março e outubro, totalizando 245 dias de serviço. Desta forma:

$$N_{refeicoes/dia} = \frac{1.491.135}{245} \approx 6087 \frac{refeicoes}{dia} \tag{4.1}$$

Se for extrapolado que este número de refeições é servido apenas no período do almoço e há apenas uma catraca para liberação de acesso de usuários, tem-se que no intervalo de 3 horas são servidas as 6087 refeições. Assim sendo, a catraca deve ser capaz de suportar a tentativa de acesso de 6087 usuários durante este ciclo. Então:

$$N_{acessos/segundo} = \frac{6087}{10800} = 0.56 \frac{acessos}{segundo} \tag{4.2}$$

A partir da equação 4.2, é verificado que o sistema deve ser capaz de comportar um acesso de usuário a cada 1,8 segundos, aproximadamente. Por consequência, este tempo pode ser definido como tempo de resposta total do sistema entre o início da solicitação de entrada e a liberação do acesso. A Figura 6 ilustra de forma simplificada como é realizado este ensaio.

Neste ensaio, um servidor capaz de registrar dados de conexão foi criado. Quando há uma nova conexão feita pela catraca, os dados desta requisição são armazenados em um registro juntamente com a data e hora deste requirimento. Além disso, o corpo da mensagem possui informações gerais de acesso que estavam armazenadas na catraca, estas informações são: contador de acessos, tempo médio de resposta do sistema e último tempo de resposta do sistema.

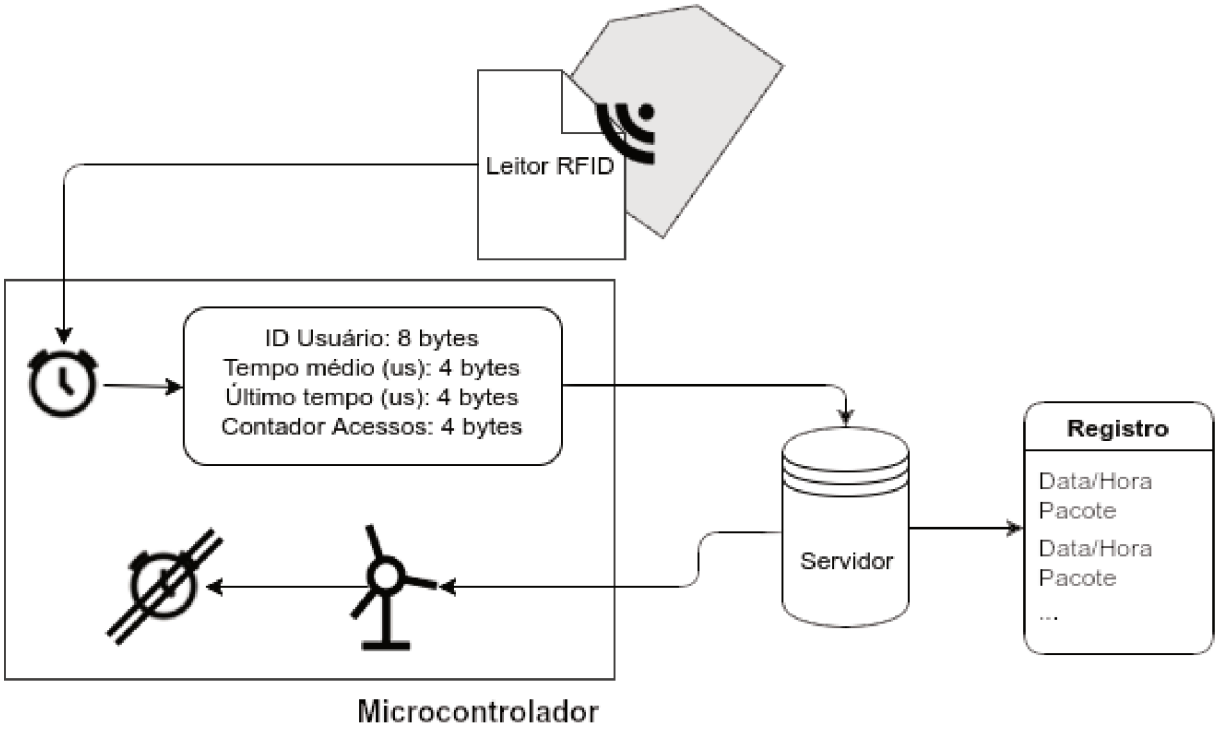
As etapas de execução do ensaio que estão expressas nesta figura podem ser enumeradas da seguinte forma:

1. Ao haver solicitação de acesso por meio da aproximação da carteirinha, o sistema inicia um cronômetro para avaliação do tempo de resposta

2. A catraca monta uma mensagem contendo o identificador do usuário que solicita acesso ao restaurante, o tempo de resposta médio do sistema, o tempo de resposta da última solicitação e a contagem de solicitações de acesso

3. Ao receber estas informações, o servidor armazena os dados em um registro e inicia a validação do identificador de usuário.

Figura 6 – Representação do Experimento



Fonte: Autor (2017)

4. Quando a validação foi completada, o servidor envia uma resposta à catraca, informando se o usuário tem o acesso permitido ou não.

5. De acordo com a resposta recebida, a catraca realiza a atuação e interrompe a contagem do cronômetro.

6. O valor de leitura atual é armazenado e a nova média do tempo de resposta total é calculada, estes valores são enviados na próxima comunicação estabelecida com o servidor.

Buscando realizar a automatização deste sistema para que seja possível realizar a execução de 6087 ciclos, a carteirinha foi acoplada a um servo motor controlado por um microcontrolador. Este servo motor executa uma ação que recorda a ação de aproximação e afastamento da carteirinha na chegada de um novo usuário. A cada iteração (movimento de aproximação e afastamento), o sistema entra em estado de espera por um período variável que vai de 1,8 a 5 segundos antes de iniciar a próxima iteração. Ao criar esta espera variável pode-se dizer que o sistema se aproxima mais da utilização real, visto que a solicitação de acesso ao restaurante não aconteceria em períodos idênticos, esta suposição é aceitável.

Como artifício para simulação dos diferentes tempos de aproximação e afastamento da carteirinha, o acionamento do servo motor é feito de uma forma atípica. Como o servo motor recebe como referência a posição do seu eixo em graus, o microcontrolador altera sua referência grau a grau, sendo que esta alteração está intercalada com uma espera aleatória de 1 a 10 milissegundos, com isto é garantido que a velocidade de aproximação seja variável.

Ao final dos 6087 ciclos, o sistema foi desligado e os dados armazenados no servidor são avaliados. O tempo de resposta total do sistema deve ser inferior ao estabelecido como requisito de projeto.

Além disso, é importante notar que neste experimento, todos os tempos relacionados à transmissão de informação na rede estão contabilizados, e que o tempo de propagação e o tempo de fila são considerados desprezíveis visto que o servidor e o protótipo estão na mesma localidade geográfica e estão utilizando uma conexão ponto a ponto. Em caso de replicação do ensaio, o tempo total de resposta pode aumentar de acordo com a distância física da conexão entre o servidor e o sistema, a quantidade de nós da rede e a sobrecarga do sistema.

4.2 Segurança da Conexão e Identificação das Partes

Para avaliar o primeiro e o segundo requisito, o protótipo se conecta com um servidor através de uma rede WiFi aberta em um computador equipado com um analisador de pacotes de rede, este analisador é um *software* bastante utilizado por analistas especialistas em redes de comunicação. O *software* em questão é conhecido como Wireshark, e pode ser encontrado em Wireshark (2017).

Ao se conectar na rede sem fio designada, o analisador é iniciado e começa a registrar todos os dados que trafegam por esta rede, ao filtrar o IP do transmissor como sendo o IP da catraca, é possível observar todas as etapas de comunicação do protocolo TCP/IP sendo executadas.

A catraca envia dados continuamente para o servidor e uma resposta deve ser recebida. Com isso, as etapas do protocolo e os dados transmitidos são verificados por meio do Wireshark. É esperado que a interpretação dos dados trafegados seja impraticável.

Além disso, é realizada a verificação da identidade da catraca ou do servidor quando é feita a troca de certificados entre o servidor e o cliente. Estes certificados devem conter informações específicas da organização em questão, e devem ser autenticados por autoridades verificadoras.

A emissão do certificado utilizado pelo protótipo é feita pela Infraestrutura de Chaves Públicas para Ensino e Pesquisa (ICPEdu), e pode ser feita por qualquer pessoa física que está relacionada a alguma instituição de ensino. O certificado possui

informações sobre o usuário que está realizando a conexão e a instituição que ele pertence, desta forma é possível saber quem é o emissor da requisição recebida pelo servidor.

Além disso, o servidor deve possuir um certificado de identificação que pode ser avaliado pela catraca durante o estabelecimento do *handshake*. Este certificado deve possuir as chaves de identidades correspondentes ao servidor que se quer conectar e caso estas informações sejam diferentes, a emissão de informações seja bloqueada.

É esperado que durante o estabelecimento do *handshake* de conexão, as informações dos certificados recebidos por ambas as partes sejam validadas, confirmando que os dados estão de acordo com as informações utilizadas na geração dos certificados.

4.3 Armazenamento Persistente de Dados

Em caso de falha da conexão ou queda de energia, a catraca deve continuar seu funcionamento para que o fluxo de pessoas no restaurante não seja interrompido. Por conta disso, informações de acesso dos usuários devem ser armazenadas de forma permanente.

Para simplificar e otimizar a utilização do armazenamento persistente de dados, um sistema de arquivos é utilizado. Este experimento busca qualificar a robustez do protótipo em casos de falha caso a liberação de acesso venha ocorrer de modo *offline*.

Além da avaliação de robustez, é possível considerar alguns tópicos importantes relacionados a utilização do RIFFS. É possível verificar o tempo de resposta de busca do identificador do usuário e também a capacidade de armazenamento do sistema.

Dado que o capítulo anterior demonstrou uma característica do RIFFS bastante importante para avaliação do espaço armazenado, é possível notar que para cada nova fração de arquivo adicionada na memória, dados de controle são adicionados ao final do setor utilizado. Desta forma, a capacidade de armazenamento é variável de acordo com a metodologia utilizada na escrita dos dados.

Se for considerado que o dispositivo possui um número de setores **S** e que cada setor possui **B** bytes disponíveis para utilização, ao descontar os bytes utilizados pelo cabeçalho do setor e seu respectivo controle, cada setor terá $B - 28$ bytes livres. Desta forma, o espaço disponível (E) em cada setor da memória seria:

$$E = (B - 28) \tag{4.3}$$

Apesar deste ser o espaço disponível real, ao adicionar informações na memória, E não diminui linearmente de acordo com a quantidade de bytes adicionados. Isto se dá por conta das informações de controle que são adicionadas. Sendo assim, a

forma em que a informação é inserida influencia no espaço disponível. Alguns exemplos e sugestões para inserção dos dados na memória estão enumerados a seguir:

1. Para cada usuário, pode ser criado um arquivo que possui como nome o identificador do usuário, contendo também as informações do tipo de usuário e a quantidade crédito que possui. Neste caso, cada usuário ocuparia 52 bytes do espaço disponível, sendo 32 bytes utilizados no arquivo criado, e 20 bytes utilizados nas informações armazenadas (dados do usuário + controle). Para esta situação, combinando a equação 4.3 com o espaço ocupado por cada usuário (52 bytes), seria possível armazenar informações de **N** usuários, sendo $N = \lfloor \frac{E}{52} \rfloor S$.

2. Da mesma forma que demonstrado no item anterior, para cada usuário um arquivo seria criado. Porém, não haveria informações contidas neste arquivo, supondo que na memória apenas estariam registrados os usuários que podem utilizar o restaurante durante um período de serviço. Sendo assim, apenas 32 bytes seriam utilizados e a memória poderia $N = \lfloor \frac{E}{32} \rfloor S$ usuários.

3. Dado que a criação de arquivos demanda uma maior ocupação dos dados, poderia ser criado um número pequeno de arquivos que conteriam uma lista com os identificadores de usuários (8 bytes), o tipo e o valor de crédito que possui (4 bytes). Esta metodologia lembra um sistema de paginação, e as páginas poderiam ser divididas de acordo com alguma regra baseada no identificador de usuário. Neste caso, para um número **P** de páginas por setor, a equação 4.3 seria variada como $E^* = (B - 28 - 48P)$ e o número máximo de usuários seria $N = \lfloor \frac{E^*}{12} \rfloor S$.

A partir dos exemplo listados, é possível verificar que a metodologia 3 seria a melhor opção de armazenamento se for priorizada a capacidade de armazenamento. Porém, para cada nova solicitação de acesso, uma busca deveria ser feita na lista e este tempo de resposta pode não ser aceitável para o pior caso desta busca.

Em contrapartida, as duas primeiras formas de armazenamento se tornam interessante se o tempo de resposta for priorizado, já que os identificadores de usuário fazem parte do nome do arquivo e são armazenados na memória RAM durante a inicialização do RIFFS. Com isso a busca seria de fato mais rápida, porém a capacidade de armazenamento de usuários seria diminuída além de existir a possibilidade de a memória RAM não conseguir suportar a quantidade de informações demandada pela aplicação.

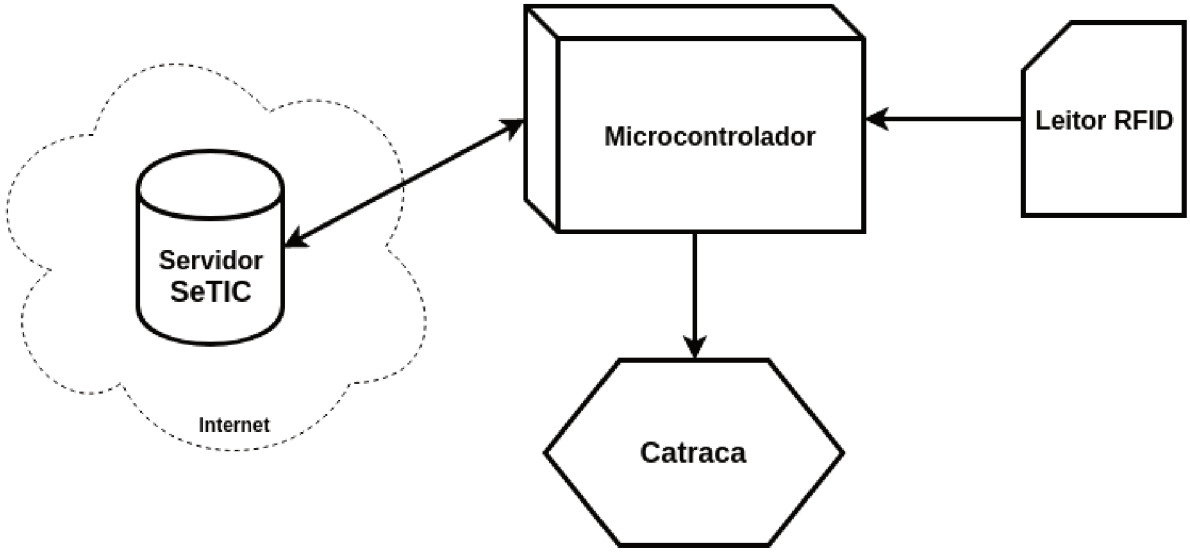
Por fim, o experimento realizado busca validar a análise do espaço de armazenamento, bem como os tempos de resposta da metodologia que for considerada mais adequada para o sistema em questão. Primeiramente o protótipo é conectado a uma rede sem fio controlada e então faz *download* de identificadores de usuários que poderiam utilizar o restaurante. O protótipo inicia sua execução normal mas sem

nenhuma conexão com a internet disponível, fazendo com que o sistema passe a trabalhar *offline*. É avaliado então, a capacidade de armazenamento e o tempo de resposta para esta situação de operação.

5 IMPLEMENTAÇÃO

Este capítulo apresenta como foi realizada a implementação do protótipo utilizado na autenticação de usuários do restaurante universitário. A primeira etapa foi realizar uma análise dos componentes utilizados e então, iniciou-se a implementação efetivamente.

Figura 7 – Modelo Preliminar do Protótipo.



Fonte: Autor

Na Figura 7 foi apresentado o modelo preliminar do protótipo estudado, com os componentes necessários para que a solução seja implementada. Apesar da imagem conter um componente representando a catraca, este trabalho não avalia sua utilização. Desta forma, o foco deste estudo é o modelo representado por um microcontrolador conectado a um leitor RFID e à internet. Apesar de simplificado, é possível verificar que com os elementos demonstrados na figura, o sistema é capaz de cumprir os requisitos de projeto.

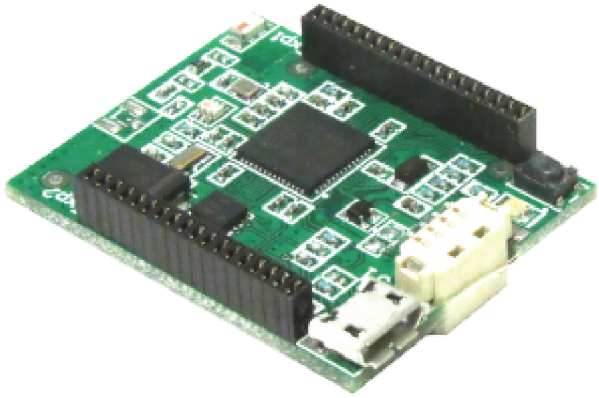
5.1 Componentes utilizados

Como um dos requisitos de projeto é o baixo custo do sistema, a seleção de componentes é feita levando esta premissa em consideração.

5.1.1 EPOSMote III

O EPOSMote III (eMote) é uma plataforma de monitoramento e atuação que foi desenvolvida pelo Laboratório de Integração Software/Hardware (LISHA) da UFSC, e pode ser visualizado na figura 8. Esta plataforma possui o circuito integrado (CI) CC2538 que é constituído por um sistema ARM Cortex-M3 e uma memória flash de 256 KB, (CC2538. . . ,).

Figura 8 – EPOSMote III



Fonte: (LISHA, 2017)

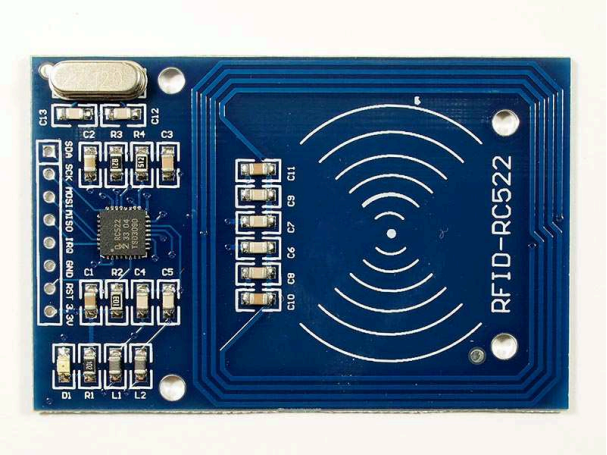
Além disso, o CI conta com interfaces SPI, UART, pinos de entrada ou saída para uso geral e conversores analógico-digital de 12 bits, fazendo com que o microcontrolador esteja apto a cumprir os requisitos do sistema.

O custo de montagem desta plataforma em baixa escala é de aproximadamente US$ 28,00. Apesar de ser mais caro que muitos outros sistemas que poderiam ser utilizadas para a mesma aplicação, devido ao fato de que ela é desenvolvida pela UFSC, a facilidade de aquisição e de customizações de seus componentes faz com que a utilização desta plataforma se torne atrativa.

5.1.2 MFRC-522

A leitura da carteirinha do usuário é feita por meio da tecnologia RFID utilizando o módulo MFRC-522, Figura 9. Este módulo possui um baixo custo (pode ser comprado por menos de R$ 20,00) e entre suas características, está a frequência de leitura de 13,56 MHz que é compatível com a frequência da carteirinha da UFSC, tensão de alimentação de 3,3 V e transmissão de dados via SPI.

Figura 9 – Módulo RFID MRFC-522



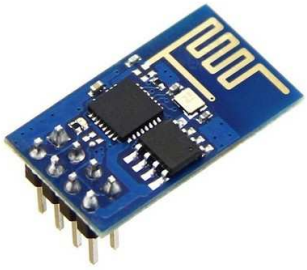
Fonte: (HOTMCU, 2017)

Quando o usuário aproxima sua carteirinha, o módulo recupera o identificador do usuário gravado e envia este identificador ao microcontrolador através do barramento SPI. Uma das desvantagens deste módulo é o fato de não gerar interrupções quando há aproximação de uma carteirinha, forçando assim a utilização de um mecanismo de *polling* periódico para verificação da presença de uma carteirinha.

5.1.3 Módulo WiFi

O módulo WiFi utilizado neste protótipo (figura 10) conta com o circuito integrado ESP8266 que tem a pilha do protocolo TCP/IP integrada, abrindo a possibilidade de conectar o dispositivo com redes WiFi, (ESPRESSIF. . . ,). Além disso, o ESP8266 possui a capacidade de lidar com conexões seguras utilizando a camada TLS sobre a pilha TCP/IP. Seu custo é de aproximadamente R$ 15,00.

Figura 10 – Módulo WiFi - ESP01



Fonte: (STUDIO, 2017)

Este módulo possui uma interface de comunicação UART, para troca de dados com o microcontrolador e sua tensão de alimentação é de 3,3 V. O CI possui uma memória flash interna de 64 kB, o que é suficiente para armazenar informações de certificados e outros dados importantes sobre a rede conectada.

É importante notar que este módulo tem um demanda de corrente muito maior do que o microcontrolador pode oferecer e desta forma, uma fonte externa para fornecimento de energia deve ser utilizada.

5.1.4 Servidor de Testes

Para o servidor de testes foi utilizado o *framework* Node.js. Este *framework* é de código livre e roda em diversas plataformas (Windows, Linux, Unix, Mac OS).

A diferença entre o Node.js e outras tecnologias de servidor é que ao invés de possuir várias *threads* bloqueantes que são executadas na abertura de cada conexão, ele possui apenas uma *thread* não bloqueante que trata cada conexão instantaneamente, fazendo com que o projeto possa ser altamente escalável. Além disso, esta plataforma foi desenvolvida baseando-se na premissa de desenvolvimento rápido e simplificado, como a análise do servidor não é o foco deste trabalho, a utilização desta plataforma se mostrou adequada.

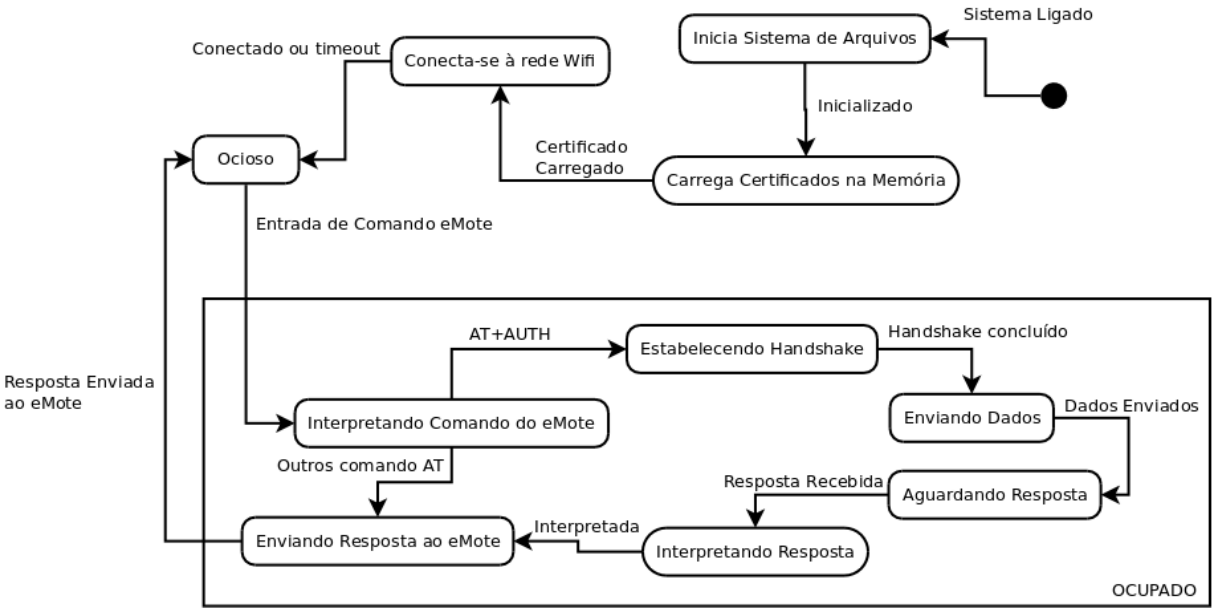
5.2 Implementação da Conexão Segura

Toda a lógica de segurança e estabelecimento da conexão com a internet é feita dentro do módulo ESP8266. Como dito anteriormente, este módulo é capaz de estabelecer conexões seguras por meio de redes WiFi, cumprindo assim, o requisito de segurança do sistema.

Este módulo conta com um *firmware* que permite sua utilização por meio de comandos AT. O problema com a utilização deste *firmware* está relacionado com o estabelecimento da conexão segura, que não está implementado da forma que é estabelecida neste projeto, não enviando o certificado de identificação ao servidor. Sendo assim, um novo *firmware* para o ESP8266 foi escrito de forma que os requisitos do projeto fossem atingidos de forma mais eficiente.

O *firmware* interpreta comandos AT recebidos via UART, estes são utilizados para configuração e envio de dados por meio de requisições POST ou GET. Além disso, é possível transmitir e atualizar o certificado de identificação da catraca por meio dessa interface. O diagrama que ilustra o funcionamento deste *firmware* está ilustrado na Figura 11.

Figura 11 – Diagrama de Estados do Firmware utilizado no ESP01



Fonte: Autor (2017)

Ao iniciar o ESP8266, o sistema de arquivos deste dispositivo é iniciado e então o certificado que será utilizado na formação da conexão é carregado na memória RAM. Com esta etapa concluída, o módulo se conecta a última rede WiFi utilizada visto que esta informação é armazenada na sua memória interna, então o componente entra no estado ocioso aguardando algum comando do eMote.

Neste diagrama, apenas o funcionamento do comando utilizado para envio de dados ao servidor é demonstrado (AT+AUTH) enquanto que outros comandos disponíveis estão colocados na tabela do Apêndice A. É importante notar que quando o ESP8266 está processando algum comando, ele se encontra no estado ocupado, descartando assim quaisquer outros comandos que possam ser recebidos.

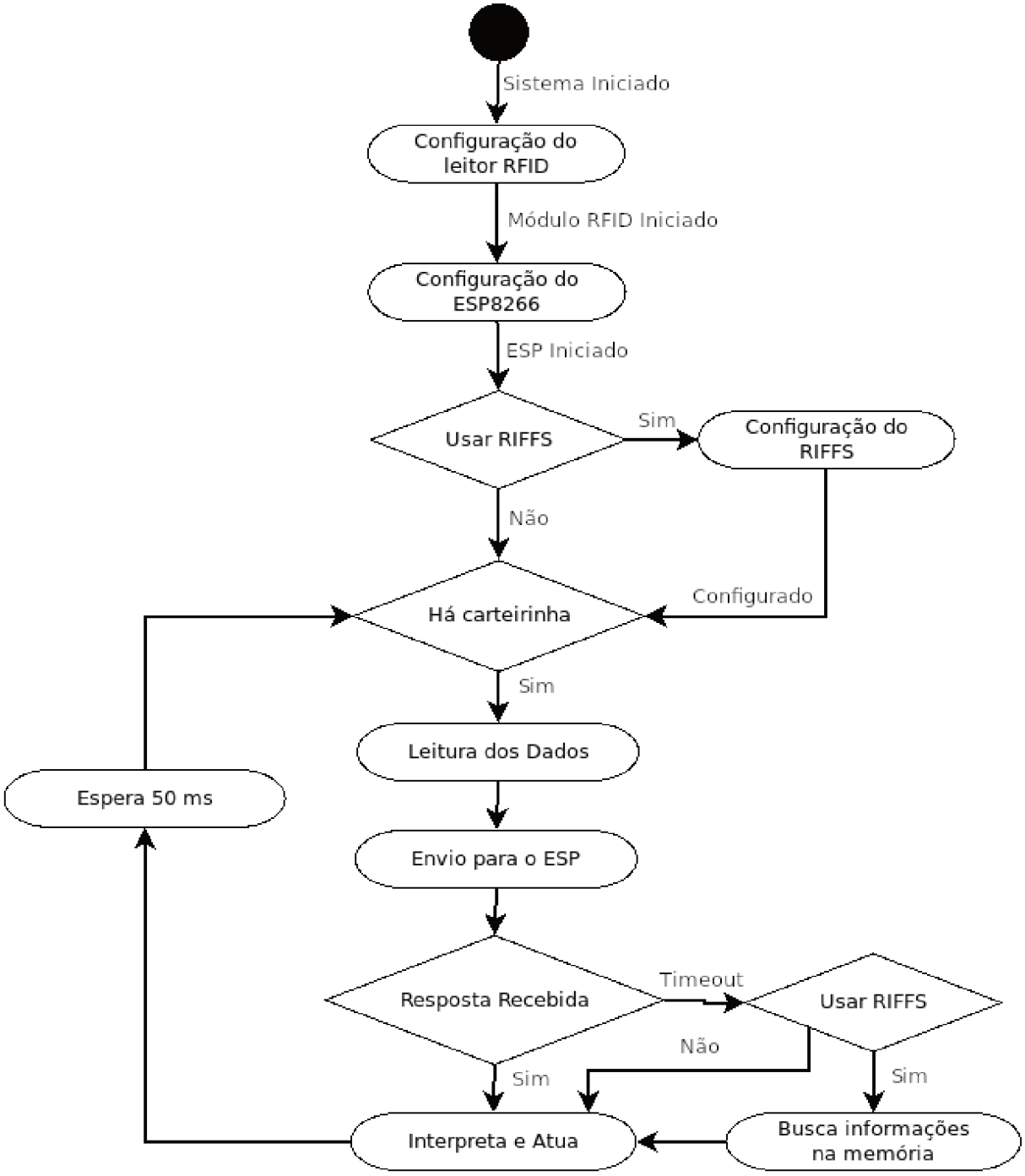
5.3 Implementação do controle no eMote

O EPOSMote é provido de um sistema operacional desenvolvido para sistemas embarcados, conhecido como EPOS. Este componente tem o papel de controlar todo o sistema, servindo de interface entre o leitor da carteirinha, o atuador na catraca e a conexão com o servidor.

Por meio de sua interface UART, o eMote se comunica com o ESP8266, enquanto que a transação de dados com o leitor de carteirinha é dada pela interface SPI. O controle da catraca pode ser feito utilizando um pino de propósito geral que aciona o relé responsável pela liberação do movimento. A tabela que lista as conexões do eMote

com os outros componentes pode ser encontrada no apêndice B. O funcionamento geral do sistema pode ser verificado no diagrama da Figura 12.

Figura 12 – Fluxograma de funcionamento do protótipo



Fonte: Autor (2017)

De forma geral, quando é verificado que há aproximação da carteirinha, o sistema realiza a leitura destes dados e inicia o processo de autenticação com o

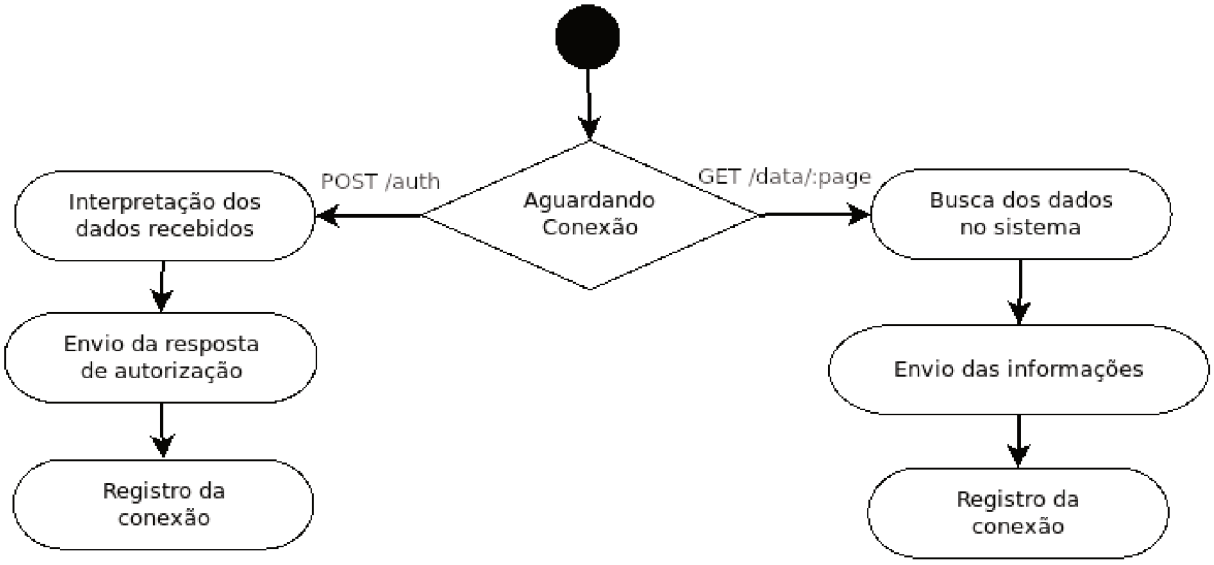
servidor, caso não possua resposta dentro de um intervalo de tempo, uma busca interna nos arquivos armazenados é realizada.

É importante notar que a forma de gravação das informações de usuários na memória não volátil impacta na metodologia utilizada para efetuação da leitura dos dados. No Capítulo 4 foram sugeridos 3 métodos distintos para implantação do algorítimo responsável pelo armazenamento desses dados mas, neste caso, apenas o terceiro método foi implementado, o método de armazenamento por paginação. Nota-se que nenhum critério para definição de páginas foi desenvolvido já que os experimentos são feitos para o pior caso e não há necessidade de definição deste critério neste estudo. Por fim, o motivo de escolha dessa metodologia se deve ao fato de que não há necessidade de carregamento de uma grande quantidade de informações na memória RAM durante a etapa de inicialização do RIFFS, tornando esta metodologia atrativa visto que sua utilização não dependerá da quantidade de memória RAM disponível no sistema.

5.4 Implementação do Servidor

O servidor foi implementado utilizando o *framework* NodeJS. Primeiramente a conexão é aberta na porta 443, que é utilizada como padrão para representação de conexões seguras. Quando há uma nova requisição, o servidor verifica a rota e o método da requisição e então executa a ação correspondente à rota e ao método. O diagrama da Figura 13 demonstra como é o funcionamento desse sistema.

Figura 13 – Diagrama de funcionamento do servidor



Fonte: Autor (2017)

Esta figura ilustra que foram criadas duas rotas, uma com um método GET e outra com um método POST. A primeira é utilizada para downloads dos identificadores que podem ter acesso ao restaurante, já a segunda rota faz a validação em tempo real se um usuário pode realizar o acesso.

Sempre que uma das rotas é acessada, as informações da conexão são gravadas em um arquivo de texto para que depois os resultados desse experimento sejam avaliados. A rota que é utilizada para o ensaio de tempos de resposta deve receber no corpo da requisição algumas estatísticas que estão armazenadas no eMote, estas informações são: contagem do número de acessos, tempo médio de resposta das solicitações de acesso e tempo de resposta da última solicitação de acesso.

6 RESULTADOS

Após a integração dos componentes demonstrados no Capítulo 5 e execução dos testes estabelecidos no Capítulo 4, foi iniciada a avaliação dos resultados e análise do cumprimento dos requisitos de projeto. É esperado que para cada requisito estabelecido, os testes demonstrem que a utilização deste sistema juntamente com as tecnologias utilizadas é viável, validando que o projeto está pronto para ter sua aplicação iniciada. Este capítulo expressa primeiramente os resultados obtidos na avaliação de tempos de resposta, então os resultados relacionados à segurança e identificação e armazenamento de dados são apresentados.

6.1 Tempos de Resposta

Como foi explicado no Capítulo 4, a execução desse ensaio foi feita de duas formas distintas:

1. A aproximação e o afastamento foram executados de forma constante, ou seja, para todas as iterações o tempo de espera entre repetições e a velocidade do movimento eram os mesmos.

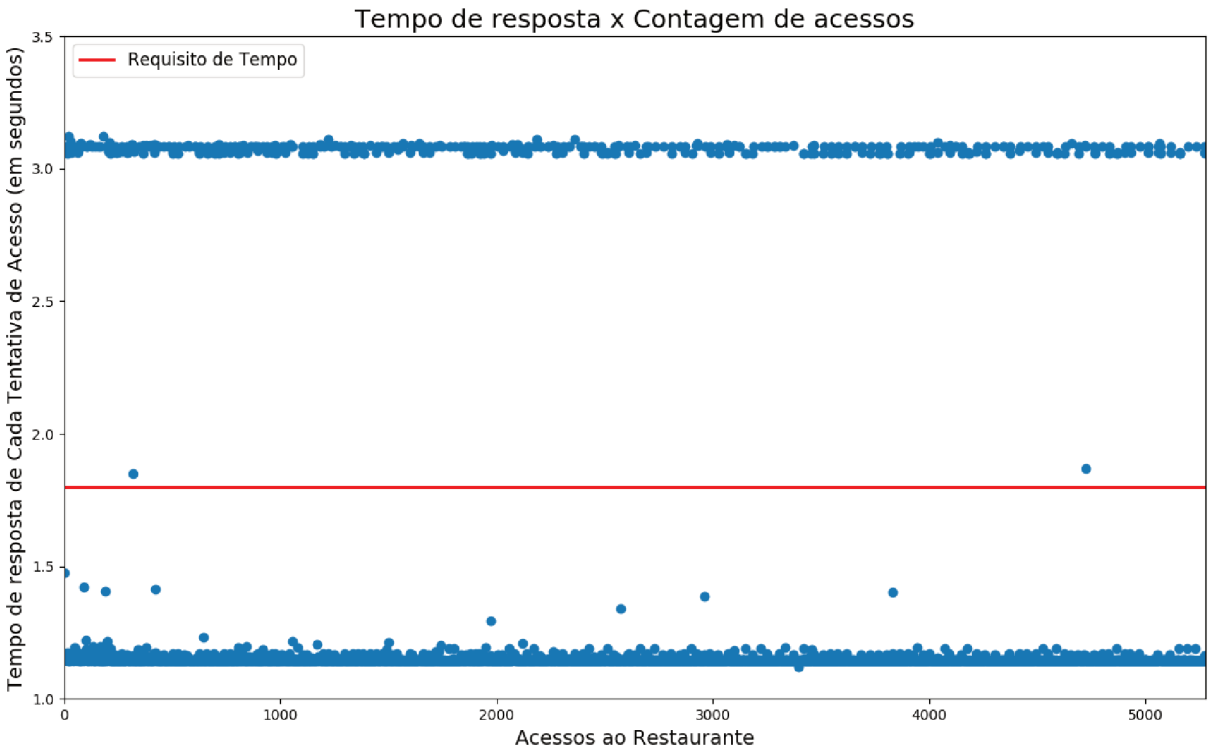
2. Um tempo aleatório de espera entre 1,8 segundos e 5 segundos foi definido entre as repetições e, além disso, a velocidade de aproximação da carteirinha é variada de forma aleatória a cada iteração.

Desta forma, foi possível verificar a execução do sistema para uma situação regular e constante, bem como uma situação irregular que se aproxima da utilização real. Foi constatado que para as duas execuções do teste, o sistema teve um tempo de resposta inferior a 1,5 segundos, garantindo assim o requisito de tempo estabelecido no projeto. A Figura 14 demonstra a distribuição dos tempos de resposta para cada acesso durante a execução do primeiro ensaio.

Pode ser verificado nesse gráfico que em algumas tentativas de acesso, o requisito de tempo não foi cumprido. Além disso nota-se que esses pontos, na sua grande maioria, se agrupam em torno do valor de 3,1 segundos. É possível afirmar que este agrupamento está relacionado com o *timeout* implementado, caso o sistema não receba nenhuma resposta em 3 segundos, aquela tentativa de acesso é descartada e uma nova tentativa é esperada. Além disso, foi verificado que 396 tentativas de acesso não cumpriram o requisito de tempo, representando 7% da quantidade total de acessos.

Outro ponto a considerar é que, apesar de haver 6100 iterações neste experimento, apenas 5272 foram contabilizadas. Fazendo com que o número de tentativas de acessos não contabilizados seja de aproximadamente, 13,57%. A hipótese sugerida

Figura 14 – Distribuição dos Tempos de Resposta para iteração constante



Fonte: Autor (2017)

para esta situação de falha de leituras, é que há limitações no módulo RFID selecionado, não sendo possível realizar leituras quando o movimento de aproximação ocorre muito rapidamente. Esta hipótese deve ser avaliada e verificada em ensaios futuros, mas é importante notar que este resultado não tem um valor efetivo para o caso da utilização real do sistema, dado que em uma utilização real, os usuários devem aguardar uma resposta efetiva do sistema antes de realizar uma nova tentativa de acesso, e está situação não foi representada neste experimento.

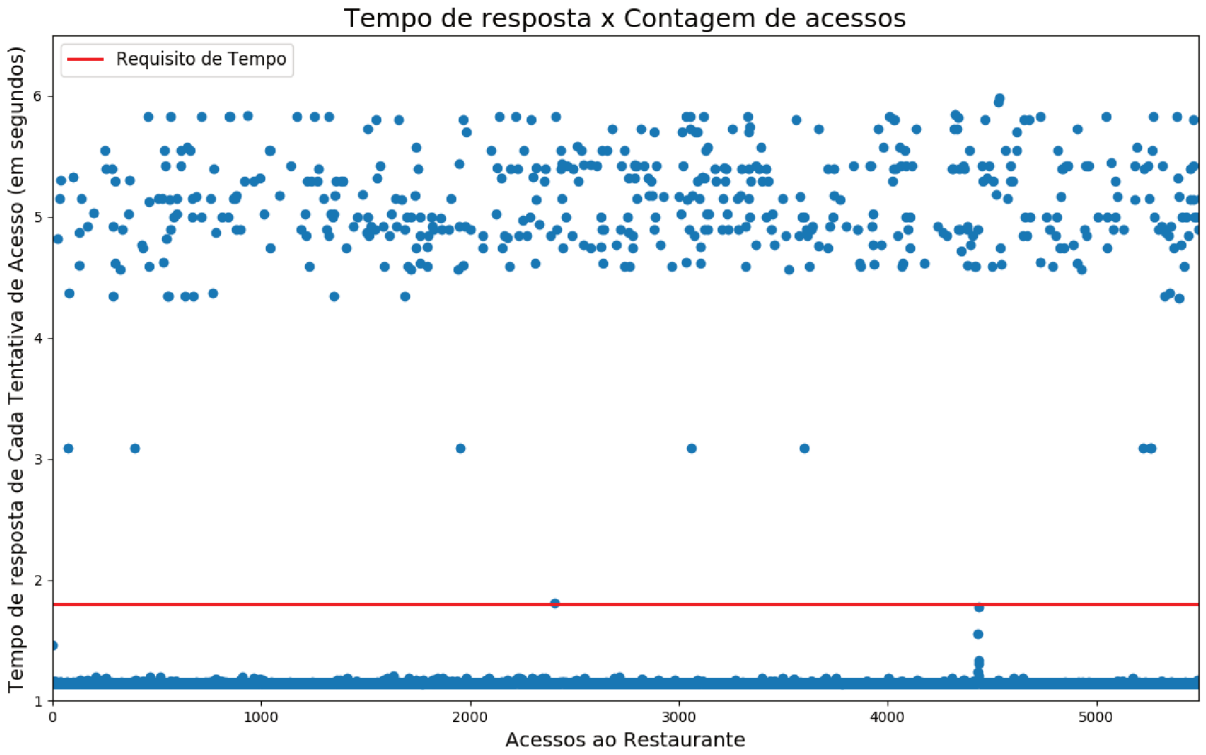
Se for analisado que este requisito de tempo foi extrapolado de tal forma que o sistema seja elevado a pontos críticos, ao considerar que há mais que uma catraca em cada restaurante, o requisito de tempo definido, é aumentado proporcionalmente ao aumento do número de catracas: com duas catracas, o requisito de tempo para cada acesso pode ser duplicado, e a demanda de capacidade ainda seria atendida. Para o caso de implantação de duas catracas no RU, o requisito de tempo poderia ser elevado em duas vezes (caso apenas a quantidade de usuários verificados for avaliada), sendo assim, ainda seria possível servir as 6 mil refeições durante um período de serviço.

Para o segundo ensaio, pode-se verificar um resultado um pouco diferente se comparado ao primeiro, esta diferença está relacionada ao maior número de acessos que não cumpriram o requisito de tempo e ao tempo médio de resposta. Diferentemente

do primeiro teste, esta execução não conta com um *timeout*, desta forma, foi possível apurar o tempo real de resposta das tentativas de acesso que extrapolaram o requisito de tempo.

É importante verificar que a porcentagem de acessos que não atingiram este requisito teve um aumento insignificativo em relação ao primeiro ensaio, neste experimento esta quantidade foi de 480 usuários, representando 8% da quantidade total de acessos. O gráfico do experimento está apresentado na Figura 15.

Figura 15 – Distribuição dos Tempos de Resposta para iteração variável



Fonte: Autor (2017)

Dado que o valor médio de todas as amostras é de 1,49 segundos, pode-se considerar que o requisito foi cumprido, porém ainda resta um questionamento sobre o aumento súbito deste tempo de resposta para algumas iterações. Sendo que a comunicação se dá ponto a ponto, não deveria haver sobrecargas na rede ou diminuição na banda do canal, então algumas hipóteses para causa destas falhas devem ser levantadas: existência de erros na implementação do *firmware* ou na implementação do controlador, falhas da comunicação e transmissão de dados entre ESP8266 e EPOSMote e falhas na implementação do servidor de registro.

Para finalizar esta avaliação, a Tabela 1 apresenta de forma comparativa os resultados obtidos para o primeiro e segundo ensaio. Como não há diferenças

expressivas entre os resultados de ambos ensaios nestas condições, valida-se a estabilidade do sistema em diferentes condições de operação.

Tabela 1 – Tabela Comparativa - Ensaio Constante e Variável

Parâmetro	**Primeiro Ensaio**	**Segundo Ensaio**
Número de Iterações	6100	6100
Número de Leituras	5272	5489
Tempo de Resposta Médio	1,29 s	1,49 s
Quantidade Acima do Requisito	396 (7%)	480 (8%)
Desvio Padrão	0,51	1,12

6.2 Segurança e Identificação

Este ensaio é bastante simples se comparado aos outros dois experimentos. Avalia-se a segurança dos dados que são enviados ao servidor. O protótipo foi iniciado de forma que o *download* de informações de usuários fosse realizado. Porém, a propagação do sinal WiFi utilizado neste teste é feita por meio de um computador com a capacidade de monitoramento de pacotes de informação que passam por ele. Desta forma, este computador é simulado como um nó do núcleo ou da borda da rede.

Ao avaliar a transmissão de um pacote TCP, é esperado que os passos de transferência de dados estabelecidos por este protocolo sejam verificados e que, principalmente, a informação contida neste pacote não possa ser interpretada. Ao executar o teste, foi verificado um resultado satisfatório que pode ser verificado na figura 16.

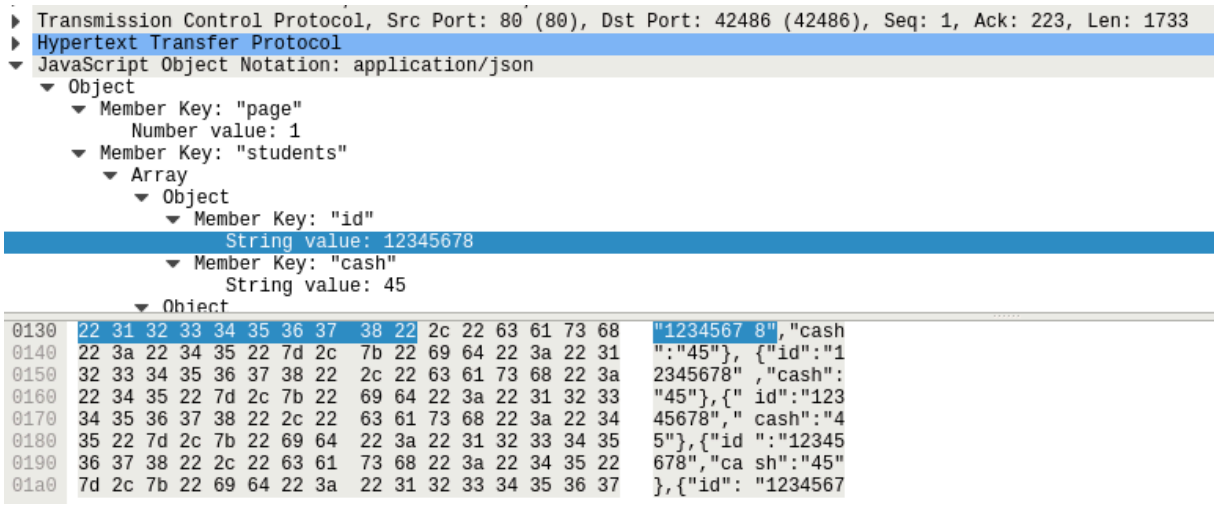
Figura 16 – Monitoramento dos Pacotes Transmitidos do Servidor a Catraca - Utilizando HTTPS

Fonte: Autor (2017)

Assim sendo, percebe-se que a execução do protocolo TCP com a camada SSL ocorre de forma adequada, fazendo com que todos os dados transmitidos entre os sistemas finais estejam ilegíveis. Para fins de comparação, a Figura 17 demonstra

a execução do mesmo teste, mas de forma que a camada SSL não seja utilizada. É esperado que por conta disso, os dados transferidos sejam legíveis e facilmente interpretados.

Figura 17 – Monitoramento dos Pacotes Transmitidos do Servidor a Catraca - Utilizando HTTP



Fonte: Autor (2017)

É possível verificar com esse teste que os dados privados da aplicação podem ser interpretados facilmente por um monitorador que pacotes. Neste caso, o servidor envia à catraca informações de identificadores de usuários e a quantidade de crédito para acesso ao restaurante que eles possuem.

Com isso, pode-se averiguar que dada a implementação realizada, a segurança dos dados transmitidos é validada, garantindo assim que o requisito de segurança na transmissão de dados do projeto seja cumprido para as situações avaliadas.

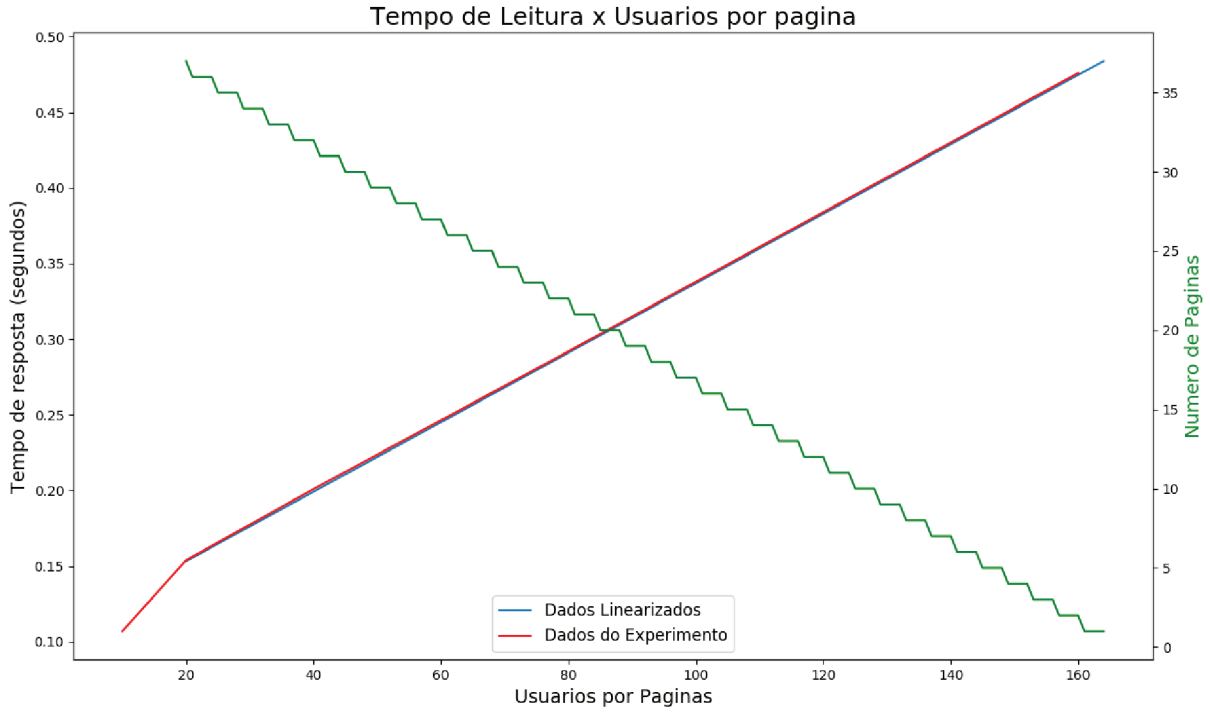
6.3 Armazenamento de Dados

Ao iniciar o sistema, é realizado o *download* dos identificadores dos usuários que podem fazer uso do restaurante e a execução do protótipo é iniciada de forma *offline*. Desta maneira, quando houver solicitação de acesso, o sistema deve realizar uma busca pelo identificador do usuário na memória e caso for encontrado, o acesso do usuário é validado.

Algumas análises interessantes podem ser feitas com o resultado deste experimento. Primeiramente, é possível avaliar a capacidade de armazenamento do microcontrolador com o RIFFS. Das diferentes metodologias para realização do armazenamento de dados, a estrutura escolhida foi a de paginação e, para esta estrutura, a

capacidade de armazenamento é inversamente proporcional ao número de páginas, sendo esse, inversamente proporcional ao tempo de leitura total. Analiticamente é verificado que se o número de páginas for diminuído, a capacidade de armazenamento aumenta, mas o tempo de leitura também aumenta. E o experimento resultante desta análise está demonstrado na Figura 18.

Figura 18 – Tempos de Leitura da página x Capacidade de Usuários



Fonte: Autor (2017)

A partir do gráfico, é possível confirmar que a análise feita está coerente. É importante notar que para o caso de uma página por setor da memória Flash, a capacidade máxima de armazenamento seria de 164 usuários por setor, totalizando 15416 usuários, se contabilizado todos os setores disponíveis para utilização. Mesmo que o tempo de resposta para esta situação seja maior, este tempo é de aproximadamente 0,5 segundos, sendo menor que o tempo de resposta de uma aplicação *online*.

Isto posto, é seguro afirmar que a utilização de um sistema de arquivos para armazenamento das informações é viável em casos onde o tempo de resposta deve ser priorizado, pois a mínima capacidade de atendimento do restaurante seria mantida. É necessário, porém, avaliar se a capacidade de armazenamento comporta a quantidade de usuários máxima que poderiam fazer parte da aplicação.

6.4 Considerações Finais

Por fim, dado que o custo é um dos requisitos do projeto, sua avaliação é apresentada com o valor médio de cada componente utilizado na concepção desse protótipo, esta avaliação não leva em conta o custo das horas de desenvolvimento.

Os valores são referentes a confecção em baixa escala com componentes adquiridos no mercado local, uma média entre os 3 fornecedores que oferecem o menor preço é obtida, com exceção do EPOSMote III que não é vendido no varejo. Na tabela 2 é apresentado o custo de cada componente utilizado.

Tabela 2 – Tabela de Custo para Confecção do Protótipo

Componente	**Custo**
MFRC522	R$ 12,60
ESP-01	R$ 11,80
EPOSMote	US$ 28,00 / R$ 91,60
Catraca	R$ 1900,00
Total	*R$ 116,00*

Pode-se notar que o custo do sistema de controle de acessos não é elevado, especialmente se for comparado a possível economia financeira que a automatização do sistema poderá trazer. Além disso, se for considerado o valor agregado pelo acréscimo de uma catraca ao sistema, o valor desembolsado para construção do componente de controle é ínfimo.

É importante constatar também, que apesar da utilização do sistema de arquivos ser viável, a discussão realizada neste trabalho levou em conta apenas a utilização de uma única catraca, situação que não ocorre na aplicação real. Desta forma, ao utilizar múltiplas catracas, as informações armazenadas em cada catraca devem ser sincronizadas, trazendo desafios técnicos e possíveis problemas. Para contornar esta situação, seria interessante que todo o processo de autorização de acesso seja localizado em um único ponto do sistema de forma que as catracas constituam uma rede de atuadores, sem nenhum papel de avaliar a situação do usuário perante o restaurante, apenas recebendo dados e atuando de forma comandada.

Esta dificuldade relacionada a centralização dos dados pelas múltiplas catracas não deve ser relevante no caso em que cada catraca autentica o usuário diretamente com um servidor, dado que as informações estariam concentradas neste servidor. O problema desta abordagem está relacionado com a robustez do sistema, em caso de falhas de comunicação ou queda de energia, cada catraca deve possuir informações dos usuários para que o funcionamento do restaurante não seja interrompido, regressando ao obstáculo relacionado com a sincronização das informações.

Há algumas tecnologias sendo pesquisadas e que poderiam ser utilizadas para

solucionar este problema. No caso da implantação de um *gateway* para centralizar a informação e controlar as diversas catracas, uma rede de comunicação que atenda os requisitos de tempo e segurança do projeto deve ser criada. Desta forma, uma sugestão para trabalhos futuros seria estudar a criação desta rede por meio da utilização do protocolo TSTP (RESNER; FRöHLICH, 2016). Este protocolo possui sincronização geográfica e temporal e tem como alguns pilares de sua fundação a eficiência energética, segurança dos dados e temporização. Dado que o requisito de tempo foi cumprido de forma satisfatória com uma margem de segurança, a adição de mais uma camada para sincronização de dados no sistema pode ser considerada.

7 CONCLUSÃO

Ao examinar os resultados obtidos, verifica-se que, dado os componentes escolhidos e a implementação realizada, o sistema de automatização do controle de acesso no RU da UFSC pode ser implantado de forma que a demanda do projeto seja atendida.

Considerando que os testes de tempo de resposta avaliaram situações críticas de operação bem como situações regulares de execução, foi verificado que o protótipo é capaz de estabelecer uma conexão segura com um servidor, e responder adequadamente ao requisito de tempo estabelecido. Além disso, a análise do projeto constatou que no caso de utilização do sistema de arquivos para mais de uma catraca, um mecanismo de sincronia dos dados deve ser implementado a fim de que as informações se mantenham consistentes.

Ao final deste trabalho, observou-se que muitos outros tópicos poderiam ser analisados na tentativa de obter resultados superiores e construir um sistema mais eficiente. Uma avaliação interessante para futuros trabalhos seria o refinamento da implementação para que mecanismos multi-tarefas sejam utilizados, dado que o sistema operacional utilizado nesta implementação dá suporte a estes mecanismos.

Por fim pode-se dizer que, tecnicamente, o sistema apresentado poderia ser implantado em uma aplicação real, se correções e aperfeiçoamentos na implementação forem realizados. De qualquer maneira, a implementação de uma solução de baixo custo que utiliza componentes simples e populares trouxe um resultado concreto de que a modernização do RU da UFSC pode ser realizada de forma adequada e efetiva, tornando todo o processo de controle de acesso do restaurante mais confiável e eficaz.

REFERÊNCIAS

CC2538 Powerful Wireless Microcontroller System-On-Chip for 2.4-GHz IEEE 802.15.4, 6LoWPAN, and ZigBee® Applications. Disponível em: <http://www.ti.com/lit/ds/symlink/cc2538.pdf>.

CUSTO de Produção - Restaurante Universitário Campus Trindade. [S.l.], 2016. Disponível em: <http://ru.ufsc.br/files/2015/10/Relat%C3%B3rio-2016-Trindade.pdf>.

ESPRESSIF SMART CONNECTIVITY PLATFORM: ESP8266. Disponível em: <https://nurdspace.nl/images/e/e0/ESP8266_Specifications_English.pdf>.

HOTMCU. **MRFC-522**. 2017. (Acessado em 20/10/2017). Disponível em: <http://www.hotmcu.com/mifare-1356mhz-rc522-rfid-card-reader-module-p-84.html>.

HTTP Over TLS. (Accessado em 20/11/2017). Disponível em: <https://tools.ietf.org/html/rfc2818>.

ITU. **Measuring the Information Society Report 2016**. 2017. (Acessado em 20/10/2017). Disponível em: <http://www.itu.int/en/ITU-D/Statistics/Pages/publications/mis2016.aspx>.

KUROSE, J. F.; ROSS, K. W. **Computer Networking - A Top-Down Approach**. [S.l.]: Pearson, 2013.

LISHA. **EPOSMote III**. 2017. (Acessado em 20/10/2017). Disponível em: <http://epos.lisha.ufsc.br/EPOSMote+III>.

NORDRUM, A. **Popular Internet of Things Forecast of 50 Billion Devices by 2020 Is Outdated**. 2016. (Accessado em 30/09/2017). Disponível em: <https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated>.

PEREIRA, M. T. **RIFFS: Um Sistema de Arquivos para Memórias Flash baseado em Árvores Reversas**. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Fevereiro 2004.

RESNER, D.; FRöHLICH, A. A. Tstp mac: A foundation for the trustful space-time protocol. 11 2016. Disponível em: <http://www.lisha.ufsc.br/pub/Resner_EUC_2016.pdf>.

STUDIO, S. **ESP-01**. 2017. (Acessado em 20/10/2017). Disponível em: <https://statics3.seeedstudio.com/images/product/WiFi+Serial+Transceiver+Module.jpg>.

TANENBAUM, A. S.; BOS, H. **Modern Operating Systems**. [S.l.]: Pearson, 2014.

WIRESHARK. 2017. (Acessado em 01/11/2017). Disponível em: <https://www.wireshark.org/>.

ZU, H. **Software Design Methodology: From Principles to Architectural Systems**. [S.l.]: Butterworth-Heinemann, 2005.

APÊNDICE A – TABELA DE COMANDOS - ESP8266

Tabela 3 – Tabela de Comandos AT para o firmware desenvolvido no ESP8266

Comando	Descrição	Resposta Esperada
AT+SYSTEMCHECK	Verifica o estado do ESP8266	CONNECTION=(OK or FAIL) FILESYSTEM=(OK or FAIL)
AT+SYSTEMREADY	Verifica se o ESP8266 está pronto	(OK or FAIL)
AT+RESPONSETIME	Recupera o tempo de resposta do servidor na última requisição	OK=(tempo em milisegundos)
AT+GETHOST	Verifica o endereço de host em que as requisições são enviadas	OK=(nome do host)
AT+GETROUTE	Verifica a rota em que as requisições estão sendo enviadas	OK=(nome da rota)
AT+GETPORT	Recupera a porta utilizada para estabelecer a conexão com o servidor	OK=(número da porta)
AT+SETHOST	Altera o endereço padrão do host	OK
AT+SETROUTE	Altera a rota padrão	OK
AT+SETPORT	Altera a porta padrão do servidor	OK
AT+SETSSID	Altera o nome da rede WiFi utilizada	OK
AT+SETPASSWORD	Altera a senha da rede WiFi utilizada	OK
AT+CONNECTWIFI	Se conecta a rede WiFi estabelecida	OK
AT+GETTIMESTAMP	Recupera o valor de timestamp atual	OK=(unixtimestamp)
AT+GETHEAPSIZE	Recupera o espaço disponível na pilha	OK=(tamanho da pilha em bytes)
AT+AUTH	Realiza uma requisição POST no endereço e rota estabelecidos	OK=(resposta do servidor)
AT+GET	Realiza uma requisição GET no endereço e rota estabelecidos	OK=(resposta do servidor)

APÊNDICE B – TABELA DE CONEXÕES

Tabela 4 – Tabela de Conexões do Protótipo

Pino no eMote	Descrição
A4	Master In Slave Out - SPI do Módulo RFID
A5	Mater Out Slave In - SPI do Módulo RFID
B5	Slave Select - SPI do Módulo RFID
C6	Reset - SPI do Módulo RFID
D5	TX - Transmissão UART para o módulo ESP
D6	RX - Transmissão UART para o módulo ESP