PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE CIÊNCIAS EXATAS E DA COMPUTAÇÃO
GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO



AUTOMAÇÃO RESIDENCIAL COM CRIPTOGRAFIA SIMÉTRICA COM CONTROLE VIA ANDROID

MANOEL MESSIAS DA SILVA JÚNIOR

GOIÂNIA
2018

MANOEL MESSIAS DA SILVA JUNIOR

AUTOMAÇÃO RESIDENCIAL COM CRIPTOGRAFIA SIMÉTRICA COM CONTROLE VIA ANDROID

Trabalho de Conclusão de Curso apresentado à Escola de Ciências Exatas e da Computação, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Engenharia de Computação.

Orientadora:

Profa. Ma. Ludmilla Reis Pinheiro dos Santos

Banca examinadora:

Profa. Ma. Mirian Sandra Rosa Gusmão

Prof. Dr. José Luiz Freitas Júnior

GOIÂNIA

2018

MANOEL MESSIAS DA SILVA JÚNIOR

AUTOMAÇÃO RESIDENCIAL COM CRIPTOGRAFIA SIMÉTRICA COM CONTROLE VIA ANDROID

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola de Ciências Exatas e da Computação, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Engenharia de Computação, em _____/_____/_____.

_____
Orientadora: Profa. Ma. Ludmilla Reis Pinheiro dos Santos

_____
Profa. Ma. Ludmilla Reis Pinheiro dos Santos
Coordenadora de Trabalho de Conclusão de Curso

GOIÂNIA

2018

A Deus pela minha vida e oportunidades.

Ao corpo docente da PUC.

Aos meus familiares, minha namorada e meus amigos.

AGRADECIMENTOS

A Deus pelo milagre da vida.

A Professora Ludmilla Reis Pinheiro dos Santos, orientadora acadêmica, pelo apoio e confiança depositada.

A minha mãe Maria das Virgens, pela paciência, amor, apoio e educação. Além dos sacrifícios diários buscando sempre o melhor para mim.

Ao professor José Luiz Freitas Júnior, pela confiança depositada e o incentivo.

Aos meus colegas pelo apoio.

A todos que direta ou indiretamente colaboraram para materialização deste trabalho.

“E enquanto seguimos pela estrada. Com nossas sombras maiores do que as nossas almas. Vemos uma senhora que todos nós conhecemos. Que emite uma luz branca e quer mostrar. Como tudo ainda vira ouro.”

Robert Plant

RESUMO

Este trabalho aborda conceitos e tecnologias que estão relacionados à segurança em automação residencial. Com estudos realizados em trabalhos relacionados, observou-se a pouca abordagem na área de segurança, e dessa forma sugeriu-se o tema deste trabalho incluindo a criptografia. Visando o baixo custo, foram selecionados aparelhos de uma residência que poderiam ser automatizados, são eles: ar condicionado, televisor e luminária. Para o controle do sistema, foi selecionado o sistema operacional Android, dessa forma desenvolveu-se um aplicativo na plataforma gratuita Android Studio. A comunicação utilizando criptografia entre o *smartphone* e a placa de desenvolvimento com recursos de Wi-Fi e *bluetooth* ESP32 utiliza *wireless*, que através de sockets, permitiu realizar a troca de dados e criar um protocolo próprio para a comunicação. Os resultados obtidos foram satisfatórios, ressaltando o baixo consumo de energia das placas, o preço total do projeto e a facilidade de utilizá-lo.

Palavras-Chave: *Automação Residencial, Android, Criptografia, ESP32.*

ABSTRACT

This project discusses concepts and technologies that are related to security in residential automation. With studies carried out in related works, it was observed the little approach in security, and in this way the theme of this work including the cryptography was suggested. Aiming at the low cost, were selected appliances of a residence that could be automated, they are: air conditioning, television and lamp. For the control of the system, the Android operating system was selected, this way an application was developed in the free platform Android Studio. The communication between the *smartphone* and the ESP32 card uses wireless, which through sockets, allowed to realize the exchange of data and to create a proper protocol for the communication. The results were satisfactory, highlighting the low energy consumption of the boards, the total price of the project and the ease of use.

Keywords*: Home Automation, Android, Cryptography, ESP32.*

LISTA DE ILUSTRAÇÕES

Figura 1 - Esquema VPN 21
Figura 2 - Desempenho das Implementações.....24
Figura 3 - Diagrama de blocos do sistema.....25
Figura 4 - Esquema geral do sistema.....27
Figura 5 - Diagrama de Comunicação.....30
Figura 6 - Esquema de ligação do sistema 31
Figura 7 - Diagrama de ligação do receptor infravermelho 35
Figura 8 - Diagrama de ligação do emissor infravermelho 37
Figura 9 - Diagrama de ligação da luminária.....38
Figura 10 - Diagrama de ligação da iluminação automática.....40
Figura 11 - Diagrama de ligação LM35. 41
Figura 12 - Tela de Login.....;.....43
Figura 13 - Tela inicial do aplicativo.....44
Figura 14 - Tela televisor do aplicativo.....45
Figura 15 - Tela ar condicionado do aplicativo.....46
Figura 16 - Tela iluminaria do aplicativo.....47
Figura 17 - Diagrama de comunicação entre placas e smartphone.....48
Figura 18 - Exemplo de protocolo 50
Figura 19 - Teste tela de login.....53
Figura 20 - Teste tela inicial. 54
Figura 21 - Testes comandos televisor.....54
Figura 22 - Teste comandos ar condicionado.....55
Figura 23 - Teste intensidades luminária.....56

LISTA DE TABELAS

Tabela 1 - Custos em Silício.....23
Tabela 2 - Potência Dissipada.....24
Tabela 3 - Comandos ar condicionado.....50
Tabela 4 - Comandos iluminação.....50
Tabela 5 - Comandos Televisor 51
Tabela 6 - Protocolos ESP32 – smartphone. 51
Tabela 7 - Preço dos Componentes.....56

LISTA DE QUADROS

Quadro 1 - Código fonte para receber os dados do infravermelho.....36
Quadro 2 - Teste Luminária 8 LEDs.....38
Quadro 3 - Código Teste para LM35.....41

LISTA DE SIGLAS

AES	*Advanced Encryption Standard*
ASC	American Standard Code for Information Interchange
API	*Application Programming Interface*
GE	*Generic Enablers*
GPIOs	*General Purpose Input/Output*
ECC	*Elliptic Curve Cryptography*
IDE	*Integrated Development Environment*
IoE	*Internet of Everything*
IoT	*Internet of Things*
IP	Internet Protocol
IR	*Infrared*
NDK	*Native Development Kit*
OWASP	*Open Web Application Security Project's*
PWM	*Pulse Width Modulation*
RAM	*Randon Access Memory*
PC	*Personal Computer*
PID	*Proportional-Integral-Derivative*
RGB	*Red Green Blue*
ROM	*Read-Only Memory*
RSA	*Rivest-Shamir-Adleman*
SHA	*Secure Hash Algorithm*
SRAM	*Static Random Access Memory*
TCP	*Transmission Control Protocol*
VPN	*Virtual Private Network*

SUMÁRIO

1 INTRODUÇÃO 15

1.1 OBJETIVO GERAL 16

1.2 OBJETIVOS ESPECÍFICOS 16

1.3 METODOLOGIA 16

1.4 ORGANIZAÇÃO 17

2 REFERENCIAL TEÓRICO 18

2.1 AUTOMAÇÃO RESIDENCIAL 18

2.2 SEGURANÇA NA INTERNET DAS COISAS 19

2.3 CRIPTOGRAFIA EM HARDWARE 19

2.4 TRABALHOS RELACIONADOS 20

2.4.1 Interconexão segura de dispositivos IoT à nuvem através de redes virtuais privadas (VPNs) 20

2.4.2 Análise Comparativa do Custo e do Desempenho de um Algoritmo de Criptografia para Sistemas Embarcados Explorando o Particionamento Hardware/Software 22

2.4.3 Automação Residencial de Baixo Custo por meio de Dispositivos Móveis com Sistema Operacional Android 25

2.4.4 Desenvolvimento de um sistema de automação residencial de baixo custo aliado ao conceito de internet das coisas (IoT) 26

2.5 DISCUSSÕES SOBRE OS TRABALHOS RELACIONADOS 28

3 PROPOSTA DE SOLUÇÃO 30

3.1 ESTRUTURA DO SISTEMA 30

3.2 COMPONENTES EM HARDWARE 32

3.2.1 Placa de desenvolvimento ESP32 e ESP8266 32

3.2.2 Infravermelho 33

3.2.3 LDR 33

3.2.4 LM35 33

3.3 COMPONENTES DE SOFTWARE 33

3.3.1 Arduino IDE 33

3.3.2 Android Studio 34

4 TESTES DE COMPONENTES 35

4.1 Testes Modulares de Hardware 35

4.1.1 Infravermelho receptor 35

4.1.2 Emissor Infravermelho 37

4.1.3 Luminária 37

4.1.4 LM35 40

4.2 Testes de Software 41

4.2.1 Teste Algoritmo de Criptografia AES ESP32 41

5 DESENVOLVIMENTO DO APLICATIVO E SEGURANÇA DA COMUNICAÇÃO 43

5.1 Desenvolvimento do aplicativo 43

5.1.1 Tela de Login 43

5.1.2 Tela Inicial 44

5.1.3 Tela Televisor 45

5.1.4 Tela Ar Condicionado 46

5.1.5 Tela Iluminação 47

5.2 Comunicação 47

5.3 Protocolos 49

5.3.1 Protocolos Android para ESP32 49

5.3.2 Protocolos ESP32 para smartphone 51

5.3.3 Protocolos ESP32 para ESP8266 52

6 RESULTADOS OBTIDOS 53

6.1 Testes de funcionamento do sistema 53

7 CONSIDERAÇÕES FINAIS 57

7.2 Conclusão 57

7.2 Trabalhos Futuros 57

REFERÊNCIAS 59

1 INTRODUÇÃO

Internet das coisas (IoT, do inglês *Internet of Things*) pode ser descrita como uma infraestrutura de rede global dinâmica, baseada em protocolos de comunicação em que coisas físicas e virtuais têm identidades, atributos físicos e personalidades virtuais, utilizando interfaces inteligentes e integradas às redes telemáticas (LEMOS, 2012). Pode ser citado como exemplo de aplicação de internet das coisas: um carro se comunica com o ar condicionado na residência do proprietário, informando o tempo estimado para chegada em casa, de acordo com o trânsito, com isso, o ar condicionado climatizará o ambiente de acordo com a preferência do usuário.

Estima-se que no ano de 2017, houveram em torno de 20 milhões de conexões inteligentes entre máquinas apenas no Brasil, em 2020 espera-se em torno de 43 milhões conexões inteligentes. Tal crescimento é acompanhado de grandes investimentos. A empresa SAP, anunciou que investirá 2 bilhões de euros até o ano de 2021 em soluções de IoT (ALBERTIN; ALBERTIN, 2017).

A empresa CISCO, prevê que entre os anos de 2013 a 2022, a internet de todas as coisas (IoE, do inglês *Internet of Everything*) que é um conceito que abrange a internet das coisas juntamente com pessoas, processos e dados, movimentará US$ 14,4 trilhões na área das industrias e empresas (BRADLEY, 2013).

Por se tratar de uma área que tende a crescer e utiliza redes públicas para o tráfego de informações, é evidente a necessidade de segurança. No entanto, grande parte das aplicações de IoT exigem sistemas embarcados, dificultando a implementação de segurança, pois podem possuir certas restrições devido as dimensões que o microcontrolador deve atender e temperatura dissipada. Com o intuito de fornecer mais segurança em tais aplicações, surgiram placas que contêm algoritmos de criptografia embutidos para a transmissão de dados, por exemplo, a placa ESP32 da empresa Espressif Systems.

Criptografar significa cifrar um texto, reproduzi-lo em código não conhecido, tornando-o, intencionalmente ininteligível para os que não devem ter acesso às informações (COSTA; FIGUEIREDO, 2010).

A criptografia pode ser classificada em: simétrica e assimétrica. Na criptografia simétrica, conhecida também como criptografia de chave privada, o processo de cifragem e decifragem são feitos com uma única chave, logo o remetente e o destinatário já devem ter conhecimento prévio da mesma.

Na criptografia assimétrica, também chamada de criptografia de chave pública, cada entidade possui um par de chaves, uma é pública e outra privada. Ao enviar uma mensagem, é necessário que o transmissor realize a criptografia com a chave pública do receptor, e assim apenas a chave privada do receptor pode decodificar a mensagem (MORENO, PEREIRA; CHIARAMONTE, 2005).

Com o intuito de explorar a área de segurança da internet das coisas, este trabalho propõe uma abordagem utilizando criptografia na comunicação entre um *smartphone* e uma placa de desenvolvimento ESP32.

1.1 Objetivo geral

Implementar um aplicativo usando o conceito de internet das coisas, para controlar equipamentos de uma residência, utilizando criptografia simétrica para garantir a segurança na comunicação entre o aplicativo e a placa ESP32.

1.2 Objetivos específicos

- Explorar o conceito de internet das coisas na automação residencial;
- Configurar a placa ESP32 para permitir a comunicação com: luminária, ar condicionado e televisor;
- Implementar um aplicativo na linguagem Java para o sistema operacional Android, que realizará comunicação criptografada com a placa ESP32.

1.3 Metodologia

- Estudo das funcionalidades das placas ESP32;
- Estudo sobre aplicações mobile utilizando o sistema operacional Android;
- Estudo de referenciais teóricos;
- Testes unitários dos componentes;
- Testes de integração dos componentes;
- Implementação de um aplicativo para o sistema operacional Android que permite controlar os seguintes dispositivos: luminária, ar condicionado e televisor de uma residência;

- Utilização do algoritmo de criptografia, para garantir a segurança na comunicação WiFi;
- Documentação.

1.4 Organização

Este capítulo apresentou uma breve introdução ao tema, os objetivos do trabalho e a metodologia usada.

O capítulo 2 apresenta os principais conceitos que estão diretamente ligados ao tema, o estado da arte e uma discussão sobre a mesma.

O capítulo 3 apresenta a proposta de solução, na qual especifica o comportamento do sistema e a forma de comunicação entre os componentes, além da descrição dos componentes de hardware e software.

O capítulo 4 apresenta como foi realizado os testes de componentes de hardware e software.

O capítulo 5 apresenta e descreve as telas do aplicativo Android. Também é mostrado a forma que foi proposto a segurança da informação e os protocolos utilizados na comunicação.

O capítulo 6 apresenta o teste de integração entre aplicativo Android com a placa ESP32 e os resultados obtidos.

O capítulo 7 apresenta as considerações finais e sugestões para trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo são apresentados os principais conceitos relacionados ao trabalho, são eles: automação residencial, segurança na internet das coisas e criptografia em hardware. Também são expostos trabalhos relacionados relevantes ao tema tratado.

2.1 Automação residencial

A automação residencial, também conhecida como domótica, teve seu conceito introduzido na França na década de 80, caracterizada por oferecer serviços de sistemas tecnológicos com a finalidade de atender necessidades básicas de segurança, conforto, comunicação e a gestão energética de uma residência (MACHADO, 2017).

Apesar de algumas empresas planejarem o desenvolvimento da microeletrônica para aplicações residenciais, a dificuldade de um sistema que funcione com um bom nível de robustez e ofereça uma interface simples para usuários, com pouco conhecimento técnico da área para gerir o sistema, fez com que a área de automação industrial se desenvolvesse mais rapidamente do que a área de automação residencial (BEGHINI, 2013).

Com a popularização dos computadores pessoais (PCs), os mesmos poderiam servir como uma central de automação. No entanto, devido a necessidade de manter o PC ligado, surgiu como desvantagem o consumo de energia (BEGHINI, 2013).

Para minimizar tal desvantagem, utiliza-se dispositivos embarcados com um maior poder de processamento, além de funcionalidades como acesso a rede de internet e *bluetooth* (BEGHINI, 2013).

Sendo assim, em uma residência existe inúmeros equipamentos que podem ser controlados por um *smartphone* ou um controle remoto, no entanto, a implementação de um sistema que automatiza uma residência pode ter um alto custo (WANZELER; FÜLBER; MERLIN, 2016).

2.2 Segurança na internet das coisas

Questionamentos sobre a segurança em dispositivos IoT, são abordados por empresas como a Symantec, Trend Micro e Kaspersky. A *Open Web Application Security Project's* (OWASP), em 2014, apresentou uma lista de vulnerabilidades que podem ser encontradas na internet das coisas, tais como: interface web insegura, autenticação e autorização insuficientes, serviços de rede inseguros e falta de criptografia nos dados (CONDEZ et al., 2017).

Para suprir a necessidade de segurança no mecanismo de transporte, pode ser utilizada uma *Virtual Private Network* (VPN), ou seja, uma rede privada criada através de uma rede pública que utiliza protocolos de tunelamento e criptografia (CONDEZ et al., 2017).

2.3 Criptografia em Hardware

Algoritmos computacionais executados em hardware possuem um melhor desempenho do que a implementações em software, porém, o custo de uma implementação integralmente em hardware se torna elevado. Assim, opções intermediárias sugerem o particionamento, ou seja, parte do algoritmo implementado em software e parte em hardware (COMPUTER ON THE BEACH, 2017).

Um hardware criptográfico pode ser utilizado para diminuir o fluxo de trabalho do processador, pois sistemas criptográficos podem exigir um custo computacional significativo (BURNETT; PAINE, 2002).

Aceleradores criptográficos compõe um hardware criptográfico, funcionando como co-processadores matemáticos, que implementam em hardware funções que normalmente seriam tratadas por um software. O fato das funções serem codificadas em hardware, possibilita que as realizações das tarefas sejam feitas mais rapidamente comparadas as funções codificadas em software (BURNETT; PAINE, 2002).

2.4 Trabalhos relacionados

Há diversos trabalhos que abordam a automação residencial juntamente com o conceito de IoT, e outros sobre criptografia em sistemas embarcados. Os próximos subitens apresentarão cinco trabalhos relacionados a estes temas.

2.4.1 Interconexão segura de dispositivos IoT à nuvem através de redes virtuais privadas (VPNs)

O artigo Interconexão segura de dispositivos IoT à nuvem através de redes virtuais privadas (VPNs), objetivou-se em criar uma conexão entre dispositivos IoT a plataforma em nuvem de forma segura, e para garantir a segurança fez-se o uso de VPNs. Dispositivos IoT, geralmente contam com uma baixa capacidade de armazenamento, por isso, em alguns casos, é necessário a transferência de dados coletados para estruturas computacionais com uma maior capacidade de processamento e armazenamento. Assim, a computação em nuvem tornou-se uma facilitadora da internet das coisas (CONDEZ et al., 2017).

Dessa forma, algumas plataformas em nuvem já estão preparadas para o desenvolvimento voltado a internet das coisas, como por exemplo, a Azure da Microsoft, Salesforce Thunder e Fiware (CONDEZ et al., 2017).

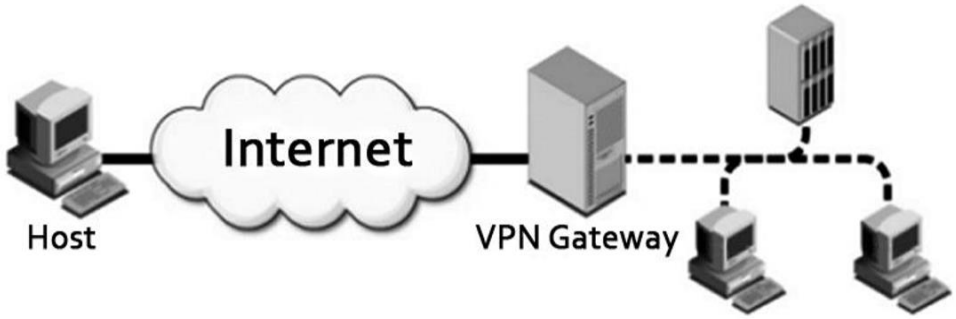
O Fiware é uma plataforma em nuvem, que pode ser descrito como “uma alternativa as soluções proprietárias já utilizadas em modelos para cidades inteligentes” além de “garantir a interoperabilidade e a criação de modelos padronizados de dados” (CONDEZ et al., 2017).

Por se tratar de uma plataforma de código aberto, é considerada OpenStack, sendo assim é empregada o uso de combinações de ferramentas para manutenção e construção de plataformas computacionais em nuvem. Tais ferramentas são conhecidas como *Generic Enablers* (GE). Dessa forma, a plataforma “possui uma gama de *Application Programming Interface* (API), GEs facilitadores que possibilitam a aplicação dos serviços de IoT e transporte de dados coletados para a estrutura” (CONDEZ et al., 2017).

Utilizando a plataforma Fiware, o trabalho conta com uma VPN para prover segurança entre host e servidor, com isso os dados ao serem submetidos à rede interna antes do servidor VPN, não possuirão criptografia. A Figura 1 esquematiza o

cenário onde a rede interna simbolizada pela linha tracejada que não possui criptografia. Enquanto a linha continua que conecta o servidor VPN com o host, possui criptografia (CONDEZ et al., 2017).

Figura 1 - Esquema VPN



Fonte: FASCI-TECH, 2017.

Para prover a segurança é utilizado o protocolo IPSec para o túnel de VPN garantindo segurança na camada IP, e os protocolos *Encapsulating Security* (ESP) e *Authentication Header* (AH) para prover segurança entre os pacotes (CONDEZ et al., 2017).

Foram realizados testes em ambientes simulados e reais. Para emular as redes nos testes em ambiente simulado, foi utilizado o software GNS3, enquanto as máquinas o *software Virtual Box*. Com uma conta na Fiware, foi instalada uma instancia do Sistema Operacional (SO) Ubuntu (CONDEZ et al., 2017).

Para teste do host, foi utilizada a placa de prototipagem Galileo Gen2, mas também é possível aplicar esta solução nas placas *Raspberry Pi* e Banana Pi. O sistema operacional utilizado no Galileo Gen 2, que é o Galileo Gentoo, pois possui módulos no kernel que suportam o IPsec (CONDEZ et al., 2017).

O trabalho sobre interconexão segura através do uso de VPN e utilizando o protocolo IPSec, conseguiu trazer um nível de segurança no transporte de dados dos dispositivos IoT à nuvem. Mas ainda é necessário incrementar camadas de proteção de dados coletados. Testes de desempenho não foram realizados, para avaliar o número de conexão criptografadas pelo servidor, também não foi feita nenhuma aplicação IoT no teste (CONDEZ et al., 2017).

2.4.2 Análise Comparativa do Custo e do Desempenho de um Algoritmo de Criptografia para Sistemas Embarcados Explorando o Particionamento Hardware/Software

Com o objetivo de comparar o desempenho de um algoritmo de criptografia em cenário de hardware, software e particionada, o trabalho “Análise Comparativa do Custo e do Desempenho de um Algoritmo de Criptografia para Sistemas Embarcados Explorando o Particionamento Hardware/Software” explora o custo e desempenho das soluções particionadas (COMPUTER ON THE BEACH, 2017).

Foram selecionados 20 algoritmos de criptografia simétrica, dos quais apenas um foi implementado. Na seleção do algoritmo, foi levado em consideração o ano de publicação, especificação, disponibilidade de implementação em software e hardware, aplicações em sistemas embarcados e criptografia leve que possibilite o uso em sistemas restritos. Assim foi feita a escolha do algoritmo SIMON. Tal algoritmo trabalha com duas funções básicas de geração de subchaves e a função de rodada que realiza a criptografia (COMPUTER ON THE BEACH, 2017).

Foram definidos três cenários diferentes para implementação do algoritmo, somente em hardware, somente em software e particionamento hardware/software. Foi utilizado um processador para leitura e escrita de mensagens, e um buffer para o armazenamento das mensagens cifradas. Dessa forma, foi feita a medida de latência no início do processo de criptografia quando é feita a escrita do buffer e terminada quando realiza a leitura.(COMPUTER ON THE BEACH, 2017).

O processador utilizado foi o LEON3, que é configurável e adéquo para sistemas embarcados, desta forma no primeiro cenário a criptografia foi feita toda em software, ou seja, o processador recebe o algoritmo de criptografia, e assim é responsável pela criptografia e armazenamento (COMPUTER ON THE BEACH, 2017).

O cenário da criptografia totalmente em hardware é feito com blocos em hardware que realiza todas as funções do algoritmo, e o processador LEON3 é responsável pelo armazenamento, transmissão e recepção de mensagens (COMPUTER ON THE BEACH, 2017).

Por último, no cenário do particionamento hardware/software, as funções mais custosas do algoritmo são executadas em hardware, e o LEON3 é responsável pelo armazenamento do texto claro, a criptografia parcial, e decriptografia parcial,

transmissão e recepção da mensagem intermediaria (COMPUTER ON THE BEACH, 2017).

Como resultado, as métricas avaliadas foram o custo em silício, o desempenho e a potência dissipada. Na Tabela 1 é exposto o valor do custo em silício das três implementações, levando em consideração que na aplicação em software não apresenta nenhum custo adicional ao sistema mínimo adotado (COMPUTER ON THE BEACH, 2017).

Tabela 1 - Custos em Silício

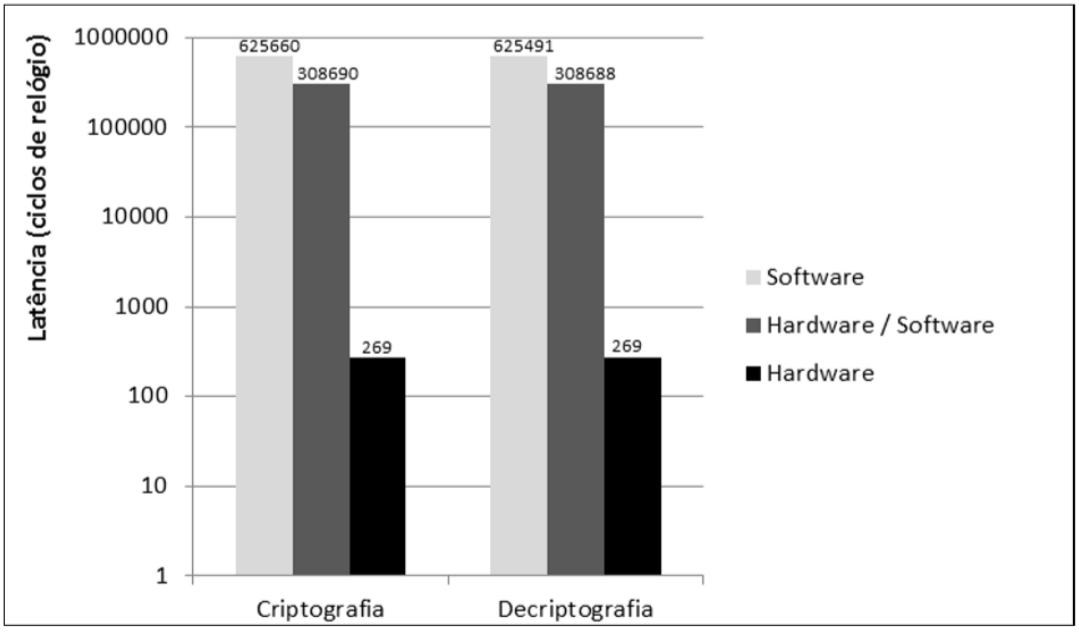
Cenário	**LUTs**	**FFs**
Software	5506	2122
Software / Hardware	6329 (+15%)	2872 (+35,3%)
Hardware	6862 (+24,3%)	3090 (+45,6%)

Fonte: COMPUTER ON THE BEACH, 2017.

O custo da implementação particionada apresentou aumento de 15% em *Look-up Tables* (LUTs) e 35.3% *em Flip-Flops* (FFs). Enquanto a implementação totalmente em hardware apresentou aumento 24,3% em LUTs e 45,6% em FFs (COMPUTER ON THE BEACH, 2017).

O desempenho foi medido pela latência, provida do momento da detecção de uma transação pelo barramento e cessada quando a mensagem termina de percorrer. A solução particionada é aproximadamente 2 vezes mais rápida que em software, enquanto a totalmente em hardware é 2.326 vezes mais rápida. O gráfico da Figura 2 apresenta os resultados do desempenho das três implementações (COMPUTER ON THE BEACH, 2017).

Figura 2 - Desempenho das Implementações



Fonte: COMPUTER ON THE BEACH, 2017.

Por último a potência dissipada, onde a implementação em sofware se mostrou um coeficiente de dissipação de 273,76mW, enquanto a implementação particionada 294,59mW, valor 7,6% maior comparada à implementação em software. A implementação em hardware gastou 296,01mW, com uma porcentagem de 8,12% maior comparada a implementação totalmente em software, tais valores estão apresentados na Tabela 2 (COMPUTER ON THE BEACH, 2017).

Tabela 2 - Potência Dissipada.

Cenário	Potência Dissipada (mW)	Latência (ms)	Energia (mJ)
Software	273,76	19,248	5,269
Hardware/Software	294,59 (+7,6%)	9,498 (-50,75%)	2,798 (-46,9%)
Hardware	296,01 (+8,12%)	0,008 (-99,96%)	0,002 (-99,96%)

Fonte: COMPUTER ON THE BEACH, 2017.

Pode-se concluir a viabilidade de algoritmos de criptografia em implementação via hardware em relação ao desempenho, em software em relação ao custo e na implementação particionada o equilíbrio de custo e desempenho.

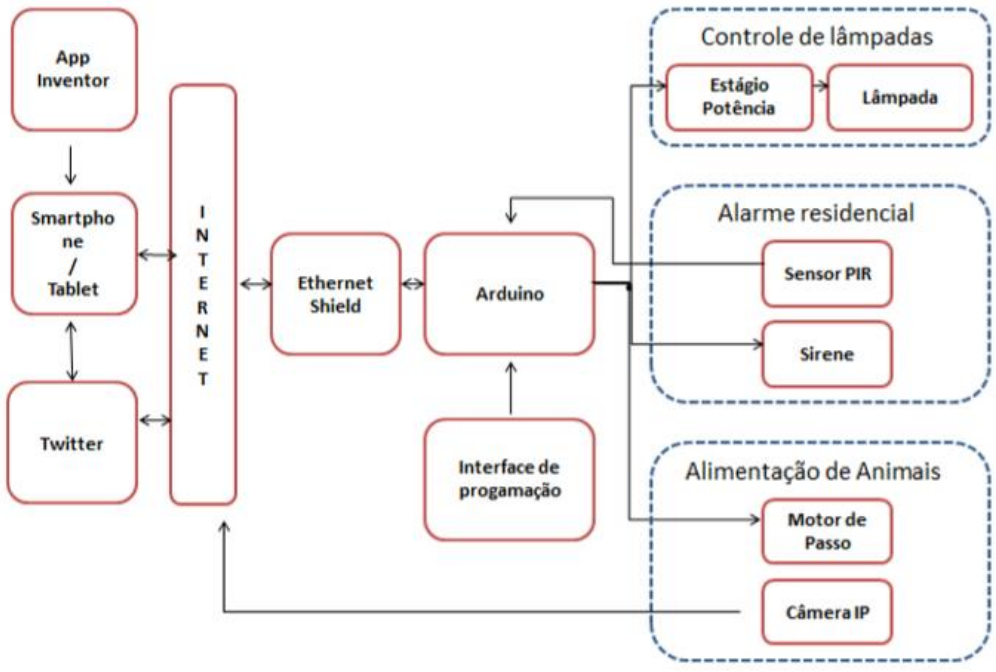
2.4.3 Automação Residencial de Baixo Custo por meio de Dispositivos Móveis com Sistema Operacional Android

O trabalho “Automação Residencial de Baixo Custo Por Meio de Dispositivos Móveis Com Sistema Operacional Android” teve como objetivo coordenar processos residenciais do cotidiano, o trabalho selecionou três aplicações a serem controladas através de um *smartphone*, são elas: controle de lâmpadas, alarme e alimentação de animais de estimação (BEGHINI, 2013).

O diagrama de blocos apresentado na Figura 3, mostra o funcionamento do sistema, controlado por um Arduino UNO, que pode: (a) acionar o motor de passo, responsável pela alimentação de animais; (b) acionar a sirene, caso receba dados indicando que o sensor de presença tenha sido ativado; (c) controlar a lâmpada (BEGHINI, 2013).

Para ativar tais funcionalidades, é necessário que o Arduino receba algum comando via internet, através de um aplicativo desenvolvido para o sistema. O aplicativo também receberá dados de uma câmera no ambiente de alimentação dos animais, sendo que não foi implementado (BEGHINI, 2013).

Figura 3 - Diagrama de blocos do sistema.



Fonte: BEGHINI, 2013.

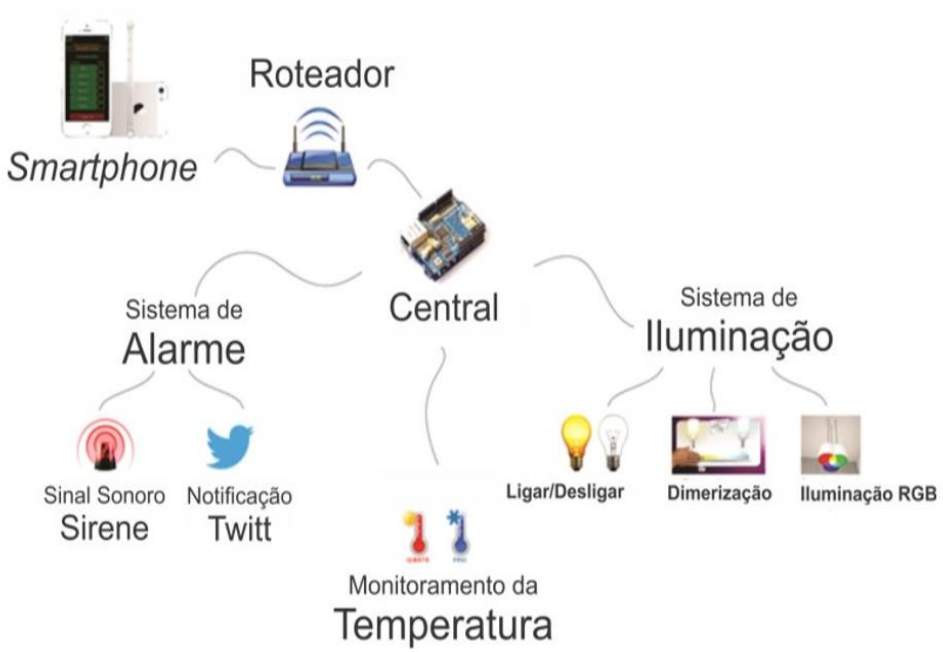
Para a construção do aplicativo, foi utilizado o *App Inventor,* que é uma ferramenta que possui uma interface gráfica de programação, com isso possibilita a usuários sem grande conhecimento em programação possa desenvolver aplicativos para sistemas operacionais Android (BEGHINI, 2013).

O trabalho apresentou um baixo custo, além de apresentar ferramentas fáceis de utilização como por exemplo o Arduino Uno e o App Inventor. No entanto, devido a utilização do Arduino, o mesmo apresenta um baixo poder de processamento e poucas portas de saídas disponíveis (BEGHINI, 2013).

2.4.4 Desenvolvimento de um sistema de automação residencial de baixo custo aliado ao conceito de internet das coisas (IoT)

Com objetivo de desenvolver um sistema de automação residencial de baixo custo, o artigo “Desenvolvimento de um sistema de automação residencial de baixo custo aliado ao conceito de internet das coisas (IoT)”, propôs a automatização de lâmpadas, onde as mesmas podem: ser ligadas ou desligadas, ter a iluminação dimerizável e um controle de iluminação RGB (red-green-blue), implementação do monitoramento de temperatura e um sistema de alarme residencial que se baseia num sensor de movimento que ao ser disparado aciona uma buzina e envia uma notificação via twitter ao usuário da residência. Todo o sistema é controlado via *smartphone* que é necessário estar conectado à rede interna da residência. Através de uma maquete residencial foi implementado o projeto prático. A Figura 4 apresenta o esquema geral do sistema (WANZELER; FÜLBER; MERLIN, 2016).

Figura 4 - Esquema geral do sistema.



Fonte: WANZELER; FÜLBER; MERLIN, 2016.

O sistema contém os seguintes equipamentos: Ethernet Shield, módulo relé 5 volts, sensor de temperatura (LM35), sensor de luminosidade (LDR), sensor de movimento (PIR), roteador 150 Mbps, *smartphone,* lâmpadas e Arduino Uno (WANZELER; FÜLBER; MERLIN, 2016).

O custo do material total utilizado no sistema foi de R$ 446,00, enquanto projetos de automação que podem ser contratados de empresas podem variar de 10 a 50 mil reais já implementados (WANZELER; FÜLBER; MERLIN, 2016).

Foi desenvolvida uma aplicação para *smartphone* no editor *touchOSC*, que controla a iluminação, recebe dados da temperatura e aciona o alarme. Tal editor não exige grande experiência em programação, além de ser viável para desenvolver aplicações moveis (WANZELER; FÜLBER; MERLIN, 2016).

Pode-se concluir que a ferramenta utilizada como central da automação, o Arduino Uno, mostrou-se de fácil implementação além do custo benefício, porém é limitado na questão de portas e capacidade de processamento (WANZELER; FÜLBER; MERLIN, 2016).

2.5 Discussões sobre os trabalhos relacionados

O trabalho **"***Interconexão segura de dispostivos IoT à nuvem através de redes virtuais privadas (VPNs)",* não deixou evidente se é necessário a utilização de uma placa de prototipagem que possibilita o uso de um Sistema Operacional, pois devido a aplicação exigir módulos em um kernel que suportam IPsec, microcontroladores como arduino podem se tornar inviáveis para realização da comunicação segura utilizando o protocolo IPSec entre host e servidor.

Não há bibliotecas da plataforma FIWARE, que tenha algum mecanismo de segurança no processo de transporte de dados coletados. No entanto, a plataforma possui g*eneric enablers* (GEs) como a *KeyRock,* que oferece suporte à autenticação de usuários e dispositivos, o GE Wilma Proxy, que oferece suporte a funções de Proxy, dentre outras (FIWARE, 2018)

"Análise Comparativa do Custo e do Desempenho de um Algoritmo de Criptografia para Sistemas Embarcados Explorando o Particionamento Hardware/Software" evidencia a superioridade de desempenho de algoritmos implementados em hardware em relação às implementações em software. No entanto, o trabalho deixa claro que para realizar a implementação via hardware o custo pode se tornar maior, mas esse pode ser reduzido caso o microcontrolador ofereça aceleradores criptográficos, por exemplo o ESP32.

Os trabalhos "Automação residencial de baixo custo por meio de dispositivos móveis com sistema operacional *android* e desenvolvimento de um sistema de automação residencial de baixo custo aliado ao conceito de internet das coisas (iot)" se assemelham quanto às tecnologias utilizadas, como por exemplo Arduino e relé. Outro fator que coincide, é a forma de controle, via wireless, com uma diferença que no primeiro é necessário que o sistema esteja conectado à internet, enquanto no segundo é necessário apenas que o sistema esteja ligado à rede interna. Devido a tal funcionalidade, caso haja algum problema na rede os sistemas se tornam indisponíveis.

Ainda sobre os trabalhos de BEGHINI (2013) e WANZELER; FÜLBER; MERLIN (2016) fica evidente o conforto que traz ao usuário, pois não é necessário a proximidade a um interruptor para acionamento dos equipamentos automatizados.

Por fim, nenhum dos trabalhos apresentaram uma forma segura de comunicação entre o *smartphone* e a central, com isso o sistema se torna vulnerável

a ataques, que dependendo de qual equipamento automatizado pode ocasionar um grande prejuízo ao usuário.

Com base no exposto, o capítulo 3 apresenta a proposta de solução deste trabalho.

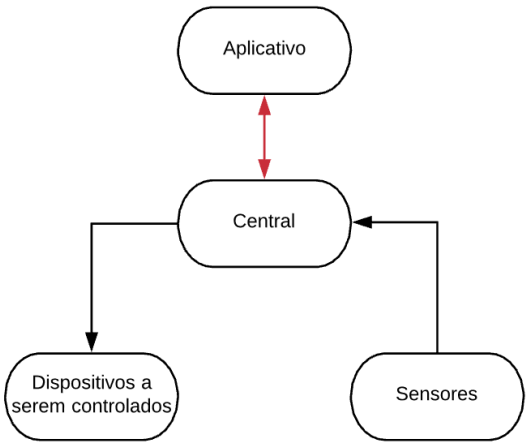
3 PROPOSTA DE SOLUÇÃO

Este capítulo apresenta a proposta de solução deste trabalho considerando os trabalhos e artigos apresentados no referencial teórico. Também é descrito as principiais tecnologias do sistema.

3.1 Estrutura do sistema

O sistema é elaborado considerando o requisito de garantia de segurança. O diagrama apresentado na Figura 5, mostra a comunicação do sistema, onde o aplicativo para *smartphones* ou *tablets*, se comunicará com a central que será composta pela placa ESP32, a comunicação poderá ser feita via *wifi, com ambos os* aparelhos na mesma rede interna. A troca de mensagens entre o aplicativo e a central será criptografada com o algoritmo de chave simétrica *Advanced Encryption Standard (*AES). A motivação para escolha do algoritmo foi devido ser um padrão de criptografia simétrica e ao suporte de aceleradores criptográficos presentes na placa ESP32.

Figura 5 - Diagrama de Comunicação.



Fonte: Elaborado pelo autor.

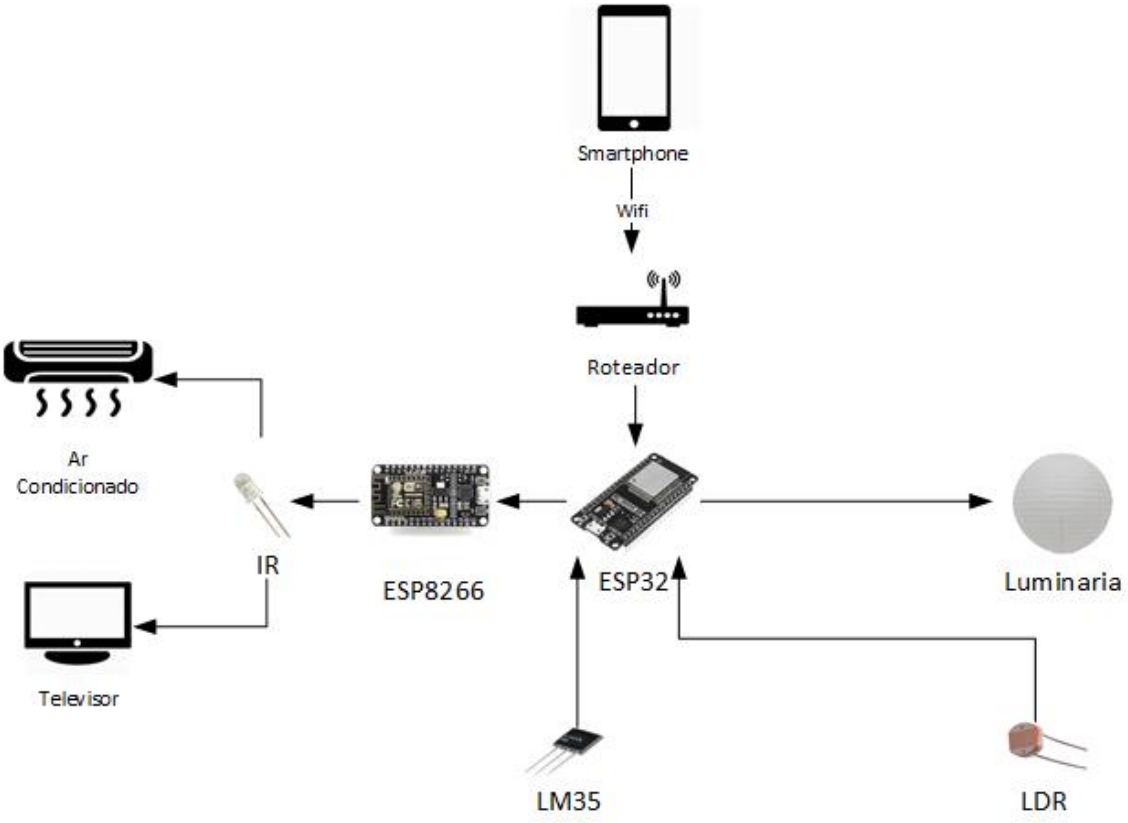
A central também será responsável por controlar dispositivos presentes na residência, que no caso deste trabalho, são eles: lâmpada, televisor e ar condicionado. A central receberá dados de sensores que indicarão a luminosidade no ambiente e temperatura.

O aplicativo tem como função: (a) informa comandos para a central, que servirá de controle remoto para o televisor e o aparelho de ar condicionado; (b) na

luminosidade, o usuário será informado do status de uma lâmpada e poderá alterá-la caso deseje, ou escolher a taxa de luminosidade para o ambiente. Para ajustar a taxa de luminosidade, o sistema controlará o brilho da luminária, caso o usuário deseje uma luminosidade maior durante o dia, a luminária terá intensidade menor ou desligada, ao passar do dia o sistema automaticamente aumenta a intensidade do brilho da luminária para garantir a mesma luminosidade que o usuário optou.

A Figura 6, apresenta o esquema de ligações do sistema onde por meio do aplicativo instalado em um *smartphone* envia comandos via wifi para o roteador que encaminhará os comandos para a ESP32. Comandos para o televisor e ar condicionado serão encaminhados para a placa ESP8266, que através de um emissor de infravermelho (IR) a placa poderá controlar o ar condicionado e televisor. Utilizando de um sensor de temperatura (LM35), será feito a medição do nível do calor em determinado cômodo da residência. E por fim, obter dados de um LDR que representam a taxa de iluminação do ambiente.

Figura 6 - Esquema de ligação do sistema



Fonte: Elaborado pelo autor.

A placa de desenvolvimento ESP32 conta com Aceleradores de hardware para os algoritmos AES, S*ecure Hash Algorithm* (SHA*), Rivest-Shamir-Adleman* (RSA) e *Elliptic Curve Cryptography* (ECC). Dos quais, o ECC e RSA são algoritmos de chave pública, o SHA é um algoritmo de função *hash*, e o AES é um algoritmo de bloco de chave privada.

O fato de algoritmos de chave pública exigirem um maior custo computacional e como consequência gastarem um maior tempo de execução, as funções *hash* têm como objetivo calcular para uma *string* (de cumprimento variável) um valor digital de tamanho fixo (chamado valor de *hash*) (BARROSO, 2016).

Foi eleito o algoritmo AES para realizar a criptografia entre a placa ESP32 e o *Smartphone,* no qual garante a segurança necessária para troca de mensagens, o modo de encriptação utilizado foi o *Cipher-Block Chaining* (CBC), que utiliza um vetor de inicialização no momento de criptografar e decriptografar os dados. Tal vetor garante que blocos cifrados, por mais que tenham os mesmos valores não resultem em blocos iguais.

3.2 Componentes em Hardware

Esta seção apresenta os principais componentes de hardware utilizados neste trabalho.

3.2.1 Placa de desenvolvimento ESP32 e ESP8266

Um microcontrolador pode ser descrito como um dispositivo digital que contém dentro de um mesmo invólucro, um microprocessador e outros dispositivos, que contém entradas e saídas digitais, conversores analógicos digitais, além de memórias R*ead-Only Memory* (ROM) e R*andom-Access Memory* (RAM) (SENAI et al., 2015). A placa de desenvolvimento ESP8266, conta com um microcontrolador ESP8266 da empresa Espressif Systems, possui Wi-Fi, 17 *General Purpose Input/Output* (GPIOs) com funcionalidades como P*ulse Width Modulation* PWM e *InfraRed* (IR), 160KBytes de *Static Random Access Memory* (SRAM), dentre outros detalhes técnicos.

Seu sucessor o ESP32, conta com Wi-Fi, *Bluetooth*, 36 GPIOs com resolução de 12 bits, no entanto não conta com funcionalidade IR, 512KBytes de SRAM, dentre outras características.

3.2.2 Infravermelho

A tecnologia de infravermelho atualmente, está presente em boa parte de controles remotos, sejam de televisor, ar condicionado, portões elétricos, dentre outros. A tecnologia utiliza um emissor de luz infravermelha no controle, e um receptor no aparelho, dessa forma, é possível que o controle envie sinais para o aparelho receptor.

3.2.3 LDR

O sensor de luminosidade *Light Dependent Resistor* (LDR) é um resistor de resistência variável, onde a resistência é inversamente proporcional a quantidade de luz sobre sua superfície, assim quanto mais luminosidade menor será sua resistência.

3.2.4 LM35

O sensor LM35 é um sensor de temperatura e apresenta uma saída de tensão linear que é proporcional a temperatura quando alimentado por uma fonte de 4V a 20V. Para cada grau célsius de temperatura, há um aumento de 10mV em sua tensão de saída, operando na faixa e -55 a 150°C (BOUFLEUR et al., 2014)

3.3 Componentes de Software

Esta seção apresenta os principais componentes de software utilizados neste trabalho.

3.3.1 Arduino IDE

O software *Arduino Integrated Development Environment (IDE)* é uma interface de programação de software livre, sendo a linguagem utilizada baseada na linguagem C. O software permite a escrita de um conjunto de instruções que são compiladas e o upload para a placa conectada. Os programas escritos na interface são conhecidos como *sketches.* A interface do Arduino IDE, também permite uma comunicação serial

entre o computador e a placa conectada. Através de um monitor serial é possível enviar comandos e visualizar dados na tela do PC. Projetada inicialmente para criar e carregar códigos para o Arduino, atualmente é possível utilizá-la em inúmeras outras placas, por exemplo a ESP32 e ESP8266 (MCROBERTS, 2011).

3.3.2 Android Studio

O *Android Studio* tem uma estrutura própria para desenvolvimento de aplicativos para o sistema operacional Android. Desenvolvido pela Google contém um poderoso editor de código, utiliza nativamente o *Gradle,* no qual é utilizado para automatizar funções que fazem parte do processo de implementação, além de emulador, ferramentas Lint para capturar o desempenho, usabilidade, compatibilidade de versão, suporte para C++ e NDK, suporte integrado para *Google Cloud Platform* (ANDROID STUDIO, 2018).

O capítulo 4 apresenta os testes de hardware e software utilizados neste trabalho.

4 TESTES DE COMPONENTES

Este capítulo apresenta os testes de cada componente contido no sistema, juntamente com seus diagramas e resultados obtidos.

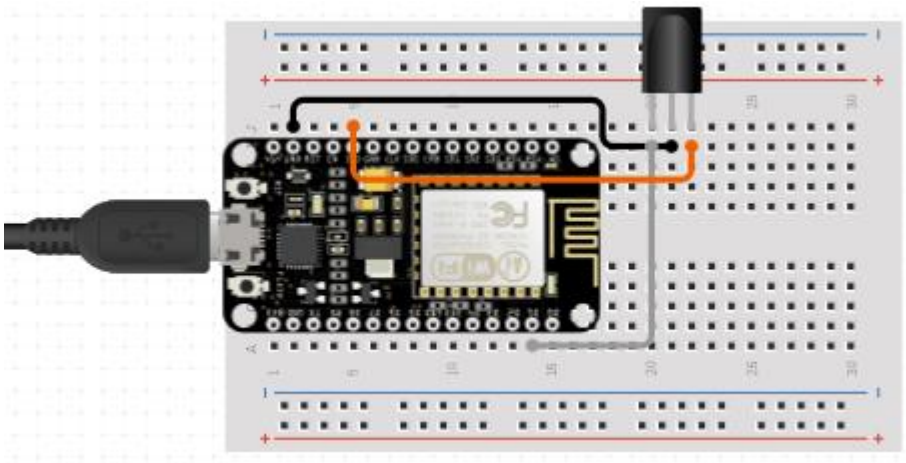
4.1 Testes Modulares de Hardware

As próximas subseções relatam os testes modulares dos componentes de hardware.

4.1.1 Infravermelho receptor

Para clonar as teclas do televisor e do ar condicionado, foi montado o diagrama de ligação do receptor infravermelho, apresentado na Figura 7, o qual é composto pela placa de desenvolvimento ESP32 e um receptor infravermelho, que objetiva capturar e armazenar o código infravermelho dos controles remoto do televisor e do ar condicionado.

Figura 7 - Diagrama de ligação do receptor infravermelho



Fonte: Elaborado pelo autor.

A placa utilizada para receber os sinais foi a ESP8266. Os dados brutos recebidos por cada tecla do controle remoto do televisor, são todos de tamanho 32 bits, podendo ser armazenados em um hexadecimal de 8 dígitos, enquanto o do ar condicionado varia podendo chegar até 141 bits. Outra característica do controle do

ar condicionado testado, é que para cada temperatura é enviado um código diferente. O código utilizado para receber os dados é apresentado no Quadro 1.

Quadro 1 - Código fonte para receber os dados do infravermelho

```
#include <IRremoteESP8266.h>
#include <IRrecv.h>
#include <IRutils.h>
uint16_t RECV_PIN = 14;
IRrecv irrecv(RECV_PIN);
decode_results results;

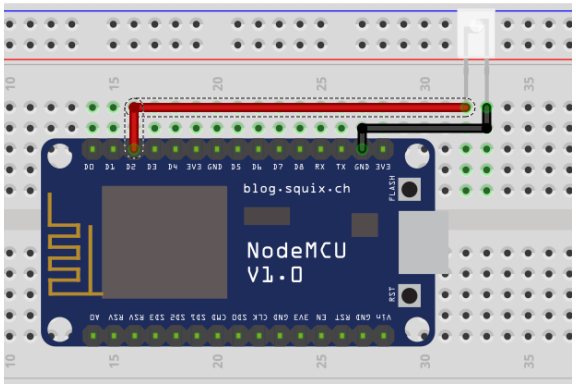
void setup() {
	Serial.begin(115200);
	irrecv.enableIRIn();
}
void dump(decode_results *results) {
	uint16_t count = results->rawlen;
	serialPrintUint64(results->value, 16);
	Serial.print(" (");
	Serial.print(results->bits, DEC);
	Serial.println(" bits)");
	Serial.print("SINAL BRUTO (");
	Serial.print(count, DEC);
	Serial.print("): {");
	for (uint16_t i = 1; i < count; i++) {
		if (i % 100 == 0)
			yield();
		if (i & 1) {
			Serial.print(results->rawbuf[i] * RAWTICK, DEC);
		}
		else {
			Serial.print(", ");
			Serial.print((uint32_t) results->rawbuf[i] * RAWTICK, DEC);
		}
	}
	Serial.println("};");
}
void loop() {
	if (irrecv.decode(&results)) {
		dump(&results);
		Serial.println("SINAL RECEBIDO");
		irrecv.resume();
	}
}
```

Fonte: Elaborado pelo autor.

4.1.2 Emissor Infravermelho

Para realizar a emissão dos dados infravermelhos, foi montado o circuito apresentado no diagrama de ligação do emissor de infravermelho, Figura 8.

Figura 8 - Diagrama de ligação do emissor infravermelho



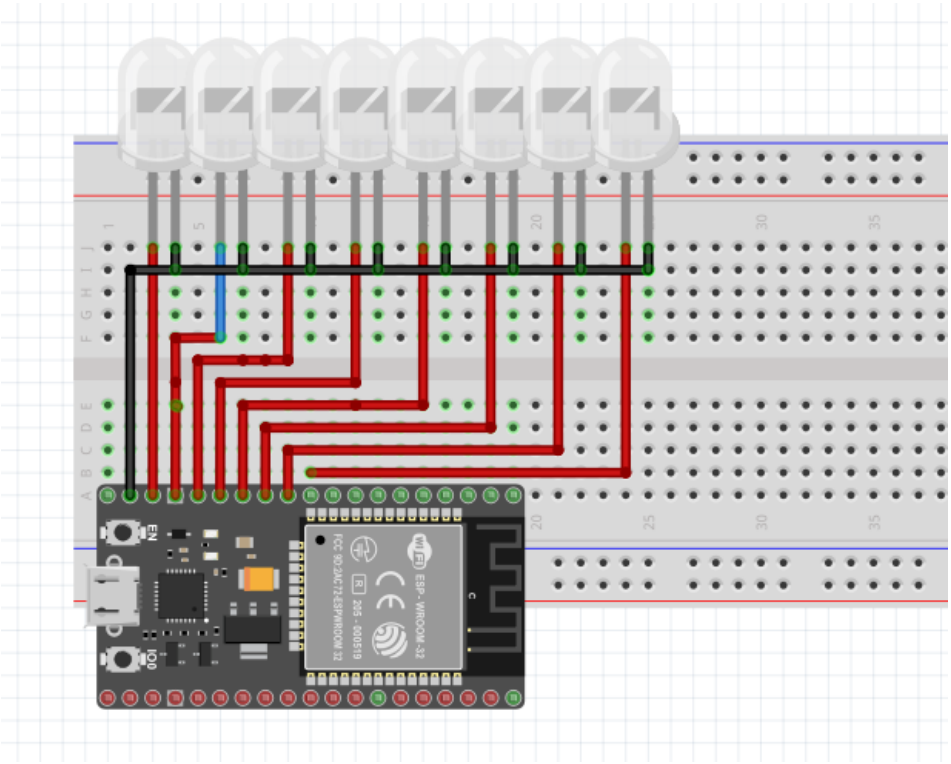
Fonte: Elaborado pelo autor.

O envio de sinais é feito de duas formas, sendo uma para o televisor e outra para o ar condicionado. Para o televisor, é enviado o hexadecimal já obtido no teste apresentado na seção 4.1.1 Infravermelho receptor. Já para o ar condicionado é necessário o envio do sinal através de um vetor do tamanho dos dados recebidos, tal vetor é conhecido pelo nome *raw*.

4.1.3 Luminária

Na luminária, foram realizados dois testes. O primeiro para verificar se a placa suportava a saída com oito LEDs, conforme o diagrama de ligação da luminária, Figura 9.

Figura 9 - Diagrama de ligação da luminária.



Fonte: Elaborado pelo autor.

Como mostra na Figura 9, não houve necessidade de resistor nos LEDs, devido a tensão e corrente fornecida pela placa ser menor do que o limite dos LEDs. O código utilizado para realizar o teste, pode ser observado no Quadro 2.

Quadro 2 - Teste Luminária 8 LEDs

```
void setup(){
pinMode(13, OUTPUT); // conFigura pino como saída
pinMode(12, OUTPUT); // conFigura pino como saída
pinMode(14, OUTPUT); // conFigura pino como saída
pinMode(27, OUTPUT); // conFigura pino como saída
pinMode(26, OUTPUT); // conFigura pino como saída
pinMode(25, OUTPUT); // conFigura pino como saída
pinMode(33, OUTPUT); // conFigura pino como saída
pinMode(32, OUTPUT); // conFigura pino como saída
ledcAttachPin(13, 0);//Atribuimos o pino 13 ao canal 0.
ledcAttachPin(12, 1);//Atribuimos o pino 12 ao canal 1.
ledcAttachPin(14, 2);//Atribuimos o pino 14 ao canal 2.
ledcAttachPin(27, 3);//Atribuimos o pino 27 ao canal 3.
ledcAttachPin(26, 4);//Atribuimos o pino 26 ao canal 4.
ledcAttachPin(25, 5);//Atribuimos o pino 25 ao canal 5.
ledcAttachPin(33, 6);//Atribuimos o pino 33 ao canal 6.
ledcAttachPin(32, 7);//Atribuimos o pino 32 ao canal 7.
ledcSetup(0, 1000, 10);//Can.0/fre.1000Hz/resolução 10b
ledcSetup(1, 1000, 10);
```

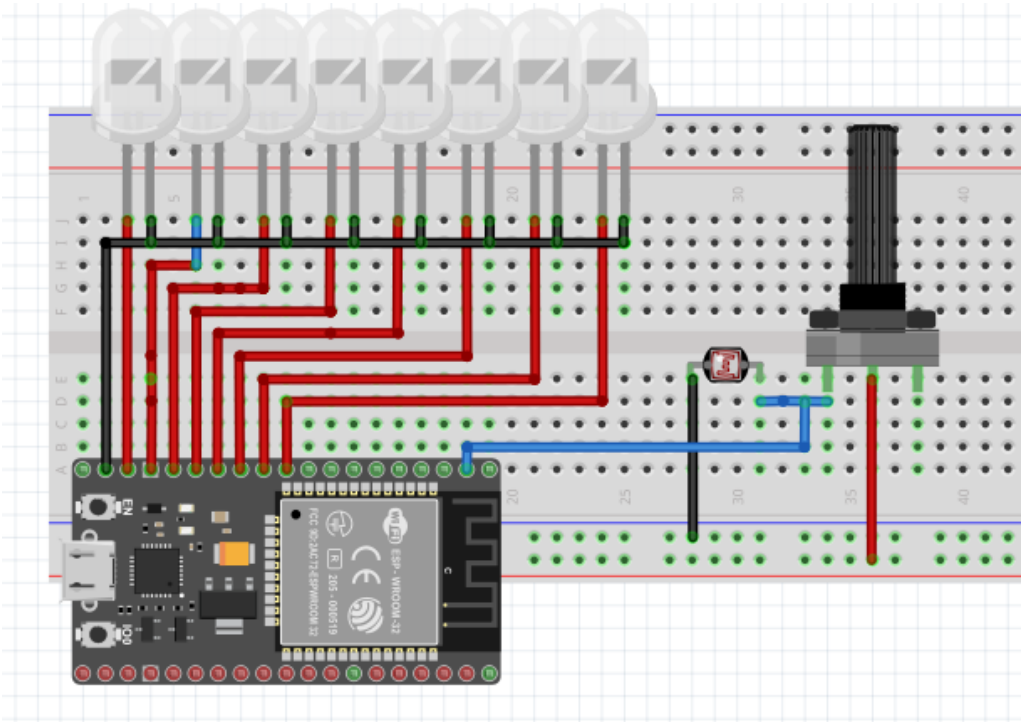
```
ledcSetup(2, 1000, 10);
ledcSetup(3, 1000, 10);
ledcSetup(4, 1000, 10);
ledcSetup(5, 1000, 10);
ledcSetup(6, 1000, 10);
ledcSetup(7, 1000, 10);
}

void loop(){
        for(int i=0;i<1024;i++){
                ledcWrite(0, i);
                ledcWrite(1, i);
                ledcWrite(2, i);
                ledcWrite(3, i);
                ledcWrite(4, i);
                ledcWrite(5, i);
                ledcWrite(6, i);
                ledcWrite(7, i);
                ledcWrite(8, i);
                ledcWrite(9, i);
                delay(2);
        }
        delay (5000); //aguarda 5s
        for(int i=1024;i>0;i--) {
                ledcWrite(0, i);
                ledcWrite(1, i);
                ledcWrite(2, i);
                ledcWrite(3, i);
                ledcWrite(4, i);
                ledcWrite(5, i);
                ledcWrite(6, i);
                ledcWrite(7, i);
                ledcWrite(8, i);
                ledcWrite(9, i);
        delay(2);
        }
    delay(1000); //aguarda 1s
   }
```

Fonte: Elaborado pelo autor.

O segundo teste da luminária, apresentado no diagrama de ligação da iluminação automática, Figura 10, foi em conjunto ao LDR para captar a iluminação do ambiente, e um potenciômetro com resistência máxima de 100kΩ para calibrar o sistema, tal teste foi realizado esquematizar a iluminação automática do sistema.

Figura 10 - Diagrama de ligação da iluminação automática.



Fonte: Elaborado pelo autor.

Para calcular o brilho dos LEDs de acordo com a luminosidade do local, foi utilizado um controlador Proporcional Integral Derivativo (PID), no qual o erro deve ser minimizado pela proporção, zerado pela integral e ajustar a velocidade da resposta transitória através do termo derivativo.

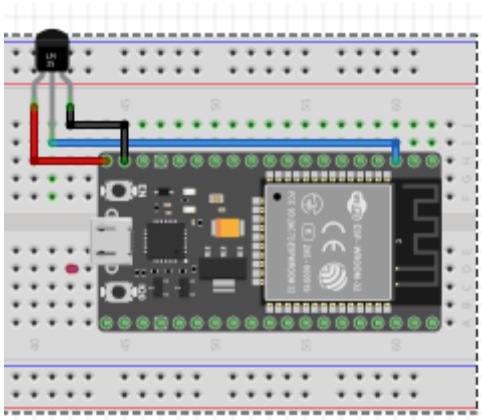
A escolha do algoritmo PID foi visando implementações futuras, com mais aparelhos para iluminação e mais dados obtidos de sensores.

O programa foi escrito prevendo a implementação final, utilizando a conexão com WiFi. Foi observado que os pinos 15 e 2 da placa ESP32, quando conectado ao WiFi, não faziam a leitura dos dados, e assim a conexão foi feita pela porta 32 que funcionou adequadamente.

4.1.4 LM35

A Figura 11 apresenta o diagrama de ligação do teste do LM35 para medição de temperatura utilizando o LM35.

Figura 11 - Diagrama de ligação LM35.



Fonte: Elaborado pelo autor.

O código utilizado para testar o LM35, pode ser observado no Quadro 3.

Quadro 3 - Código Teste para LM35.

```
int LM35 = 39; // Define o pino que lera a saída do LM35
float temperatura; // Variável que armazenará a temperatura
void setup() {
Serial.begin(115200); // inicializa a comunicação serial
}
void loop() {
 leitura = analogRead(LM35);
 temperatura = (leitura*3/(2048))*100;
Serial.print("Temperatura: ");
Serial.println(temperatura);
delay(2000);
}
```

Fonte: Elaborado pelo autor.

4.2 Testes de Software

Esta seção apresenta os testes realizados para verificar o funcionamento dos softwares desenvolvidos.

4.2.1 Teste Algoritmo de Criptografia AES ESP32

.

O código teste realiza a criptografia e decriptografia, além de colocar o resultado na base 64, onde variam de A-Z, a-z, ‘0-9’, “/” e “+”. É de extrema importância trabalhar na base 64 ou em hexadecimal, pois na comunicação entre placa e

smartphone há caracteres que indicam final de transmissão, e como na criptografia resulta sempre em caracteres diversificados, há possibilidade de que algum deles coincida com o caractere de final de transmissão.

Foi observado que ao trabalhar em hexadecimal a conversão era mais rápida, e assim o programa final trabalhou com a conversão para hexadecimal ao invés de base 64.

O capítulo 5 apresenta o desenvolvimento do aplicativo e como a segurança foi implementada na comunicação.

5 DESENVOLVIMENTO DO APLICATIVO E SEGURANÇA DA COMUNICAÇÃO

Nesse capítulo é apresentado as telas do aplicativo juntamente com as funcionalidades das mesmas e o protocolo implementado para comunicação.

5.1 Desenvolvimento do aplicativo

Esta seção apresenta as telas do aplicativo com suas respectivas funcionalidades.

5.1.1 Tela de Login

Ao abrir o aplicativo o usuário deve digitar as credenciais de usuário e senha, como apresentado na Figura 12.

Figura 12 – Tela de Login



Fonte: Elaborado pelo autor.

A senha do usuário será utilizada para criptografar as mensagens que serão enviadas para a placa ESP32, a mesma deve atender ao critério de ter 16 bytes, pois o algoritmo de criptografia implementado, AES, trabalha com senhas de 128 bits. O usuário permanece logado enquanto o aplicativo estiver aberto.

O aplicativo envia para a placa ESP32 o nome de usuário e a senha criptografados. O software do ESP32 ao decriptografar compara os dados recebidos com o usuário e senha pré-definidos, caso a informação esteja coerente, a placa ESP32 criptografa uma resposta e envia para o aplicativo que abre a TELA INICIAL.

As cores utilizadas no aplicativo foram tons de azul que fazem contraste ao fundo de tela, que conta com um degrade circular cinza que é utilizado nas demais telas do aplicativo.

5.1.2 Tela Inicial

A tela inicial do contém três botões, cada um redireciona para a tela de controle do equipamento escolhido. A Figura 13 apresenta a tela Inicial.

Figura 13 - Tela inicial do aplicativo.

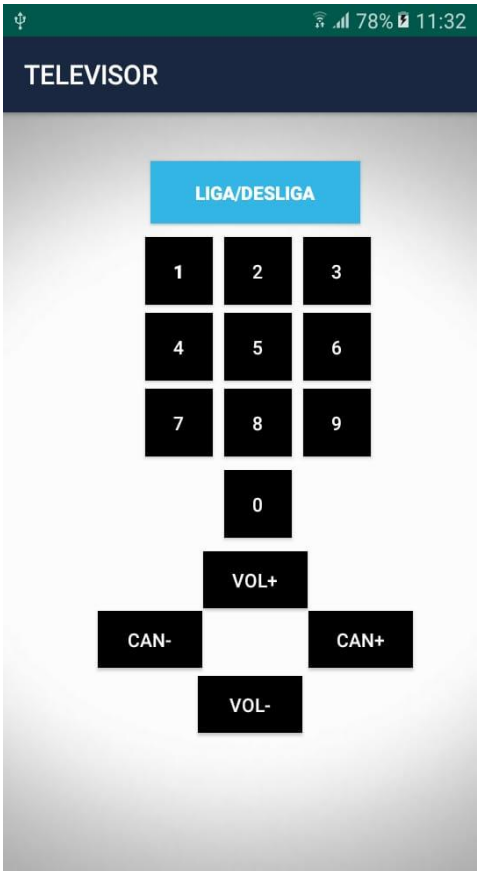


Fonte: Elaborado pelo autor.

5.1.3 Tela Televisor

A tela responsável por enviar comandos para o televisor está ilustrada na Figura 14. Foi implementado apenas os botões mais utilizados no controle do televisor, são eles: Liga/desliga, números 0 a 9, aumentar volume, diminuir volume, aumentar canal e diminuir canal.

Figura 124 - Tela televisor do aplicativo.



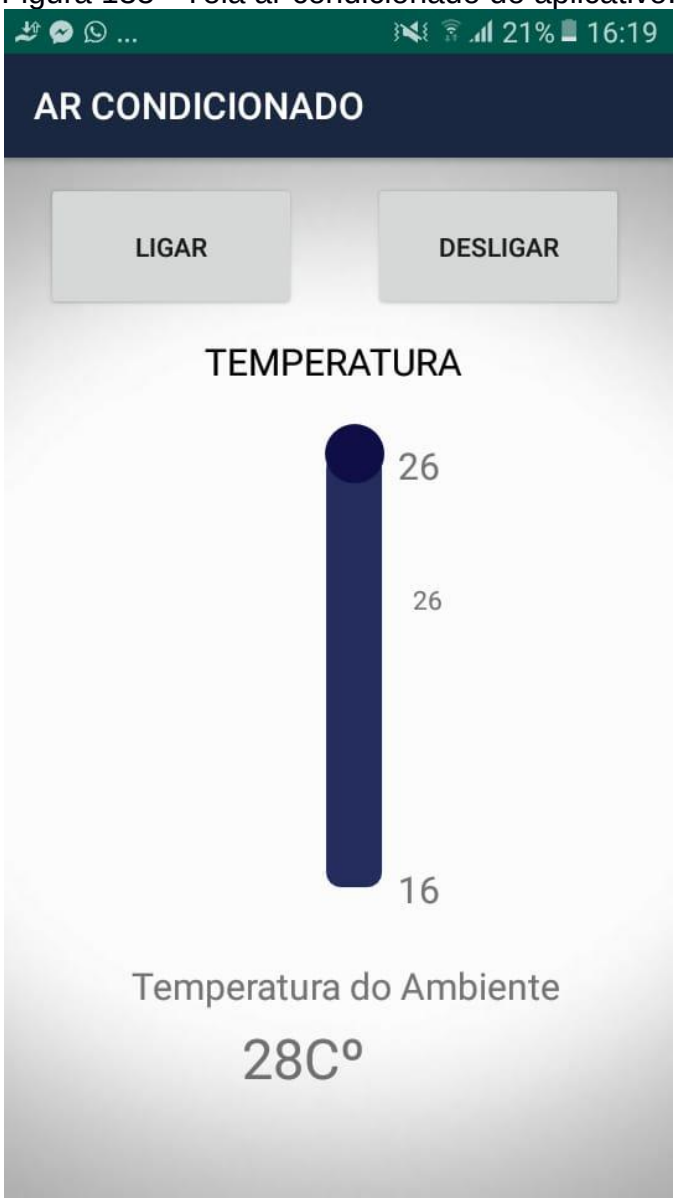
Fonte: Elaborado pelo autor.

No botão LIGA/DESLIGA foi utilizado o azul com a mesma tonalidade dos ícones da tela inicial. A cor das demais teclas é preta, para que junto com a posição das teclas, se assemelhe ao controle remoto de um televisor.

5.1.4 Tela Ar Condicionado

Na tela AR CONDICIONADO, Figura 15, há dois botões para ligar e desligar o aparelho de ar condicionado. Foram utilizados dois botões visto que o comando responsável para ligar o ar é diferente do comando para desligar. A tela também possui uma barra de rolagem que pode variar entre os valores de 16 a 26, com o objetivo de escolha de temperatura através da mesma. Por fim é apresentada no final da tela, a temperatura atual do ambiente que é obtida através de um sensor LM35 ligado a placa ESP32.

Figura 135 - Tela ar condicionado do aplicativo.



Fonte: Elaborado pelo autor.

5.1.5 Tela Iluminação

A tela de iluminação, Figura 16, conta com uma barra de rolagem na qual o usuário pode definir a intensidade da luz da luminária, podendo variar de 0 a 100. A ativação da luminosidade automática, cria um *thread* que lê o valor de tensão numa porta da placa ESP32, onde há um divisor de tensão entre o LDR e um potenciômetro. O potenciômetro é utilizado para criar a possibilidade de calibragem, já que a resistência do LDR está diretamente ligada à luminosidade do local. O *setpoint* da luminosidade pode ser escolhido através da barra de rolagem da luminária. A ativação de luminosidade automática pode ser desativada caso o botão seja pressionado novamente.

Assim como as telas do ar condicionado e login, a tela controle de iluminação faz uma requisição para a placa ESP32, como resposta há dados sobre a intensidade de luz atual na luminária.

Figura 16 - Tela iluminaria do aplicativo.



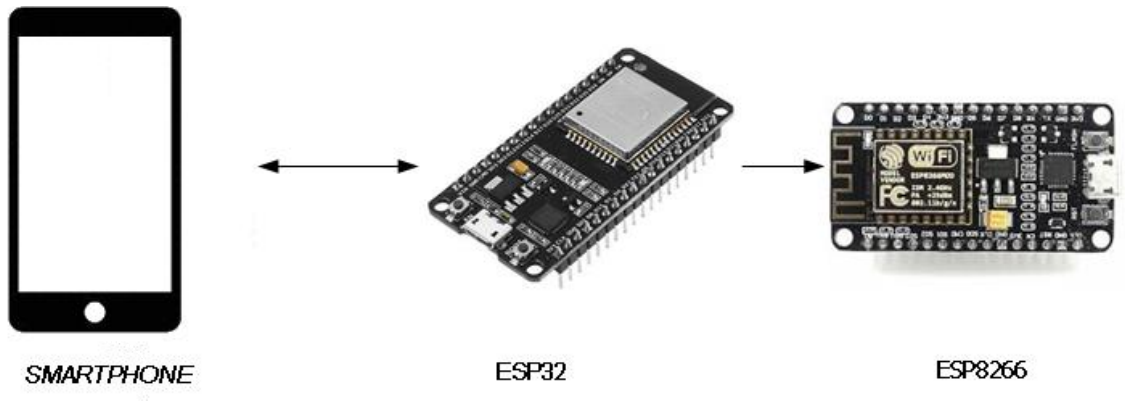
Fonte: Elaborado pelo autor.

5.2 Comunicação

O sistema funciona de forma que comandos provindos do celular são executados pela placa de desenvolvimento ESP32, e caso seja algum comando para

o televisor ou o aparelho de ar condicionado, são enviados para o ESP8266. A comunicação entre estes três componentes é feita através de WiFi, utilizando sockets *Transmission Control Protocol* (TCP) para troca de mensagens. O diagrama da Figura 17 ilustra tal comunicação.

Figura 17 - Diagrama de comunicação entre placas e *smartphone.*



Fonte: Elaborado pelo autor.

Foi criado um protocolo para envio das mensagens do aplicativo android para placa ESP32, onde o primeiro caractere indica o aparelho a ser controlado e o segundo o comando, tais protocolos estão descritos na seção 5.3. Ao clicar em algum comando no *smartphone*: (a) cria-se o protocolo com um caractere representando o aparelho a ser controlado e o comando que o aparelho deve executar; (b) realiza feito um preenchimento, pois o algoritmo de criptografia AES utiliza uma cifra de blocos de 16 Bytes; (c) aplica-se um processo de criptografia, conversão do resultado para hexadecimal e inserção do caractere ‘\r’ que indica o final da transmissão; (d) abre-se a comunicação com a placa ESP32, envia-se o protocolo e encerrado a conexão.

A placa ESP32 recebe os dados até encontrar o caractere ‘\r’, e em seguida encerra a conexão, para que caso outro *smartphone* tente conexão, a comunicação esteja livre. Depois de encerrar, o comando é convertido para base *American Standard Code for Information Interchange* (ASCII) decriptografado e o aparelho a ser controlado executa o comando.

A comunicação entre ESP32 e *smarphne* é feita de forma semelhante, com criptografia e conversão para a base hexadecimal, no entanto não há padronização dos comandos a serem enviados, ou seja, podem variar de tamanho.

A comunicação entre as placas ESP32 e ESP826 é feita por WiFi, no entanto, não apresenta criptografia. Um mecanismo de segurança proposto, foi que a placa ESP8266 só execute comando se o comando estiver sido enviado de um endereço de IP restrito o qual foi configurado no ESP32.

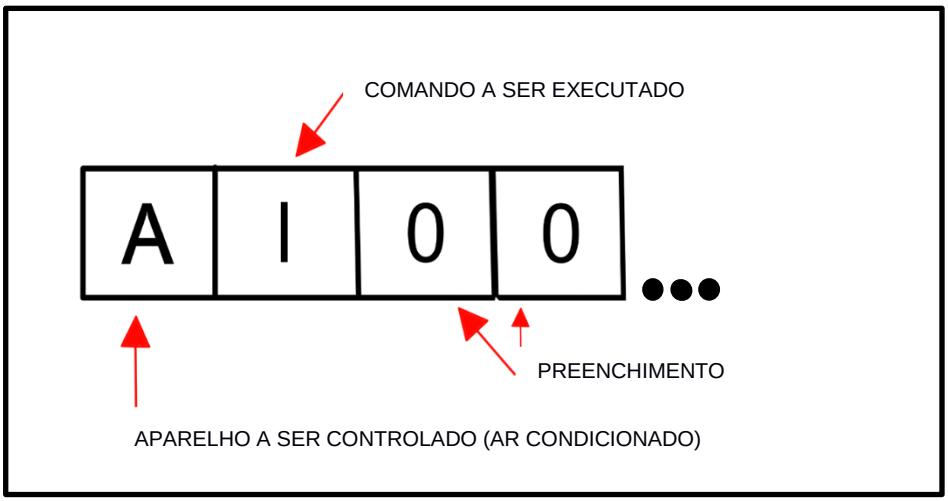
5.3 Protocolos

Nesta seção é apresentado os protocolos de comunicação entre *smartphone*, placa ESP32 e placa ESP8266.

5.3.1 Protocolos Android para ESP32

Para cada informação que chega à placa ESP32, o primeiro caractere indica o que a mesma deve executar. As possibilidades de comando são: ‘l’ indica que há uma solicitação de login; ‘i’ indica uma requisição para receber informações para montar a tela de iluminação; ‘a’ indica uma requisição para receber informações para montar tela de comandos do ar condicionado; ‘A’ indica um comando que deverá ser direcionado ao aparelho de ar condicionado; ‘T’ indica um comando que deverá ser direcionado ao televisor e ‘L’ indicando o novo valor de intensidade para luminária ou o *setpoint* do algoritmo PID, caso a opção de iluminação automática esteja ativada. Ainda é feito o preenchimento até que o protocolo tenha um tamanho de 16 bytes, tal preenchimento é feito por números 0. A Figura 18 ilustra um exemplo de protocolo.

Figura 148 - Exemplo de protocolo



Fonte: Elaborado pelo autor.

As tabelas 3, 4 e 5 descrevem os protocolos para controlar os aparelhos de ar condicionado, luminária e televisor, respectivamente.

Tabela 3 - Comandos ar condicionado

Comandos AR	**Protocolo**
Ligar	Al
Desligar	Ad
16ºC	Aq
17ºC	Aw
18ºC	Ae
19ºC	Ar
20ºC	At
21ºC	Ay
22ºC	Au
23ºC	Ai
24ºC	Ao
25ºC	Ap
26ºC	Aa
26ºC	As

Fonte: Elaborado pelo autor.

Tabela 4 - Comandos iluminação

Comandos Luminária	**Protocolo**
Intensidade 0-100	I(VALOR)

Fonte: Elaborado pelo autor.

Tabela 5 - Comandos Televisor

Comandos TV	Protocolo
Ligar	TL
Canal-	TQ
Canal+	TW
volume-	TE
volume+	TR
Mudo	TT
0	T0
1	T1
2	T2
3	T3
4	T4
5	T5
6	T6
7	T7
8	T8
9	T9

Fonte: Elaborado pelo autor

Os protocolos responsáveis para fazer solicitação de abertura de tela para a placa ESP32 são compostos unicamente de um caractere e o preenchimento.

5.3.2 Protocolos ESP32 para smartphone

Os protocolos que a placa ESP32 envia para o *smartphone* são de tamanhos variáveis, mas seguem a mesma lógica dos protocolos do *smartphone* para a placa ESP32. O primeiro caractere representa qual tela deverá ser aberta, enquanto os demais informam os dados em que a tela receberá, exceto pela tela de login, que só abrirá uma nova tela caso as credenciais estejam certas. A Tabela 6 ilustra o formato dos protocolos.

Tabela 6 - Protocolos ESP32 – *smartphone.*

Comandos	Protocolo
Abrir tela do ar condicionado	a(valor de temperatura)
Abrir tela da iluminação	i(valor da intensidade de luz da luminária)

Credenciais da tela de login corretas	lo
Credenciais da tela de login erradas	le

Fonte: Elaborado pelo autor.

5.3.3 Protocolos ESP32 para ESP8266

Quando a placa ESP32 recebe um comando para controlar o televisor ou ar condicionado, ela encaminha o protocolo sem a criptografia para a placa ESP8266, dessa sendo os protocolos de comunicação entre as duas placas estão descritos anteriormente nas Tabelas 3 e 5.

O capítulo 6 apresenta os resultados obtidos com este trabalho.

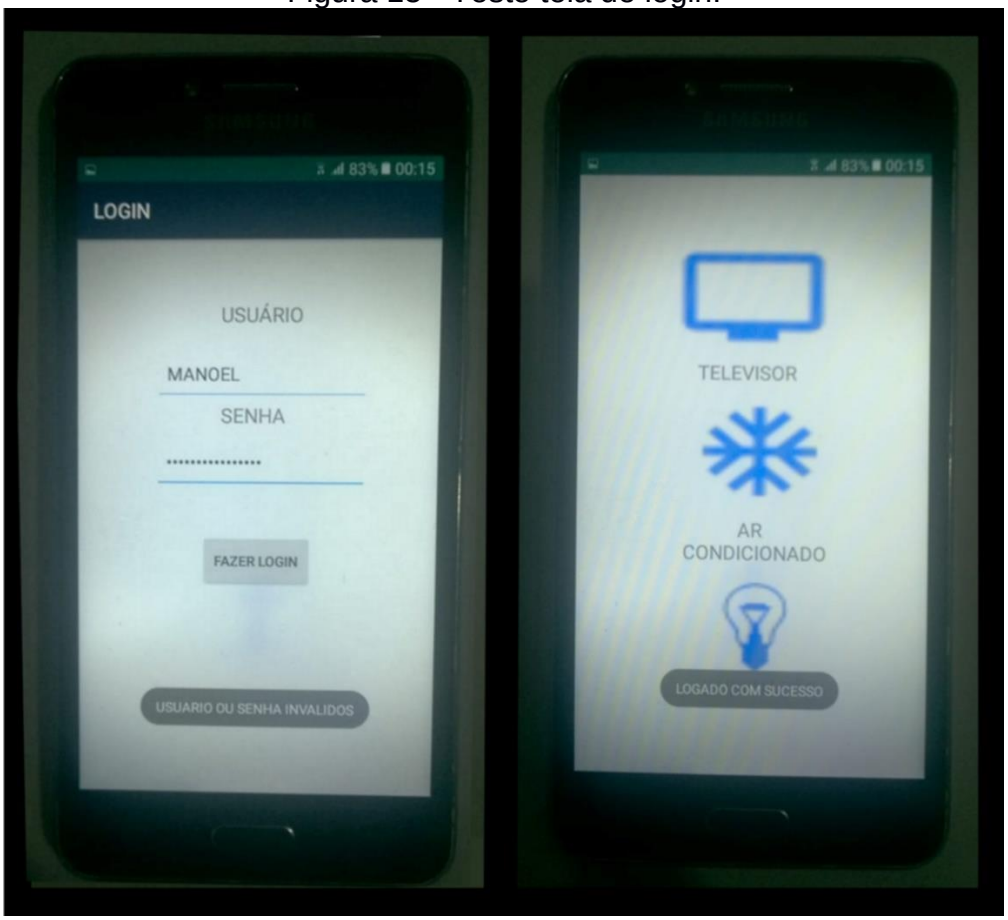
6 RESULTADOS OBTIDOS

Este capítulo apresenta os resultados obtidos da implementação final e os testes de integração do sistema.

6.1 Testes de funcionamento do sistema

Para utilização do sistema, é necessário que o usuário faça login. A Figura 19 apresenta o login sendo feito, onde a imagem à esquerda mostra o comportamento do sistema quando as credenciais estão erradas, e à direita quando estão corretas.

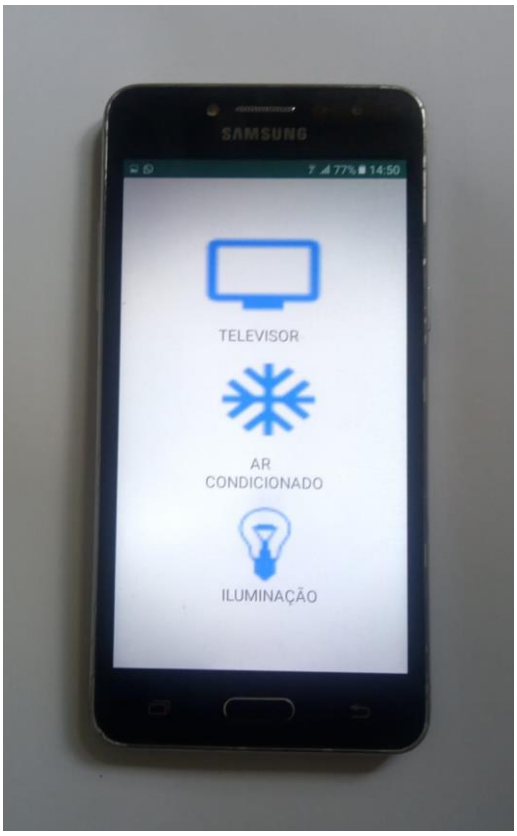
Figura 15 - Teste tela de login.



Fonte: Elaborado pelo autor.

A Figura 20 apresenta a tela inicial, onde o usuário deve escolher qual aparelho será controlado.

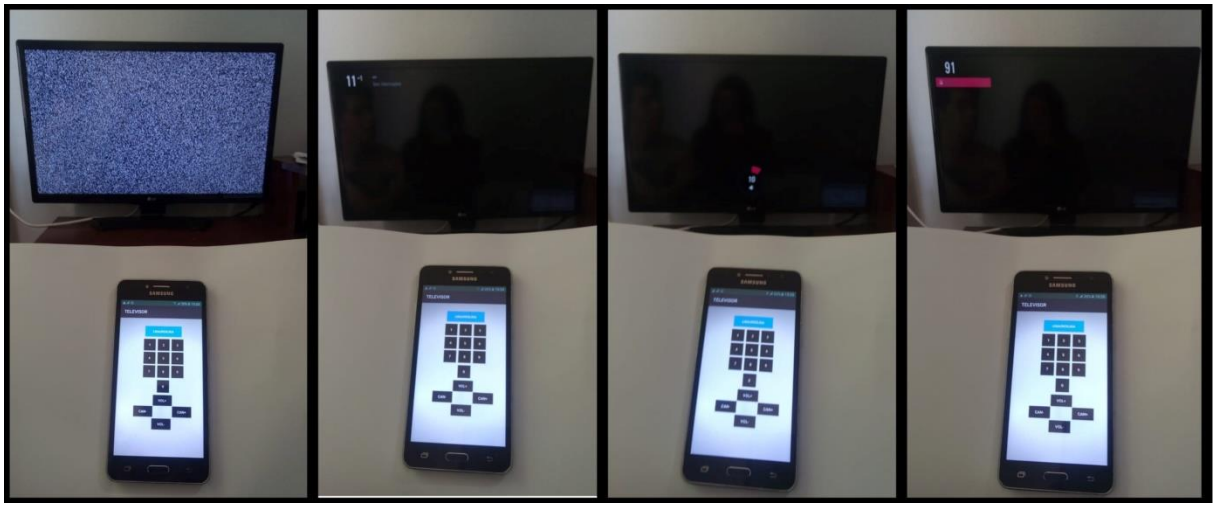
Figura 16 - Teste tela inicial.



Fonte: Elaborado pelo autor.

Ao clicar no ícone do televisor, é aberta a tela TELEVISOR para realizar o teste do controle no mesmo. A Figura 21 apresenta os resultados de alguns dos comandos testados, são eles: LIGAR, CANAL +, VOLUME +, números ‘9’ e ‘1’.

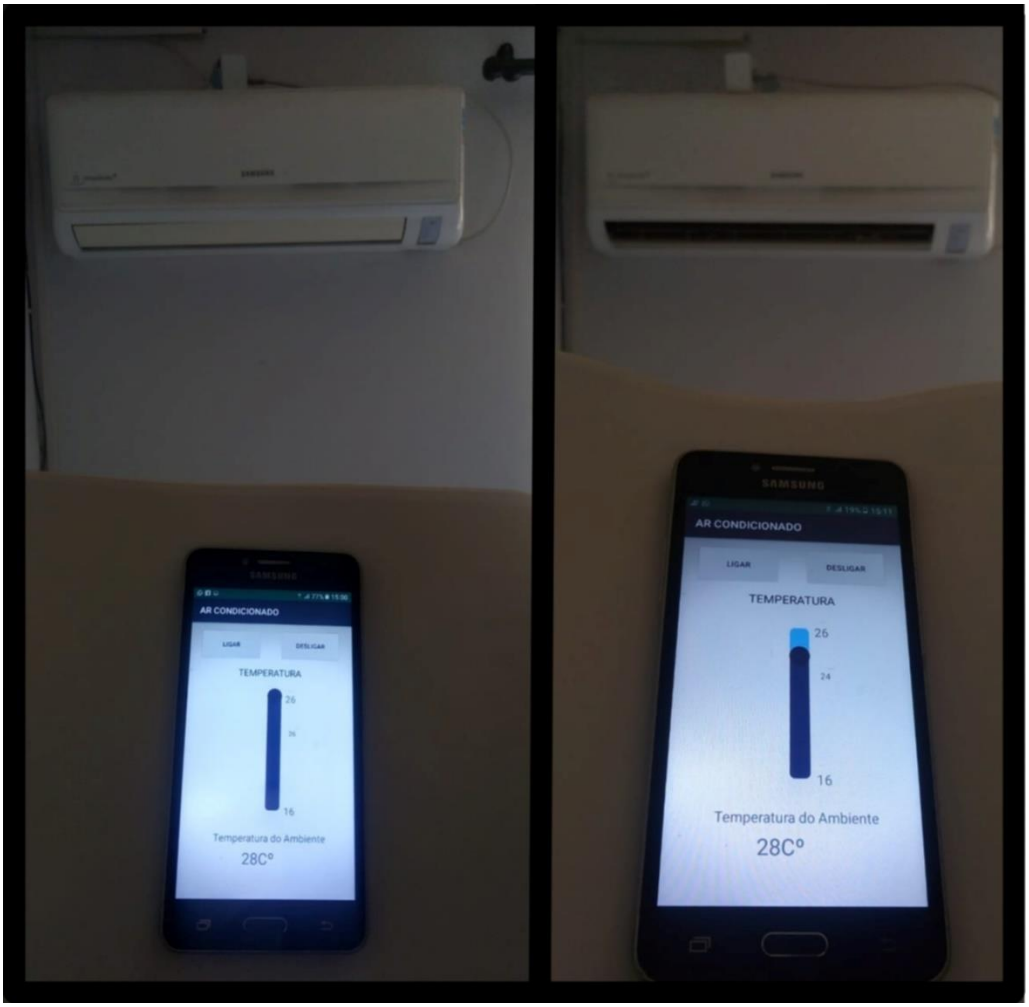
Figura 21 - Testes comandos televisor.



Fonte: Elaborado pelo autor.

O segundo teste consiste em testar o aparelho de ar condicionado. A Figura 22 demonstra o funcionamento do botão ligar da tela AR CONDICIONADO.

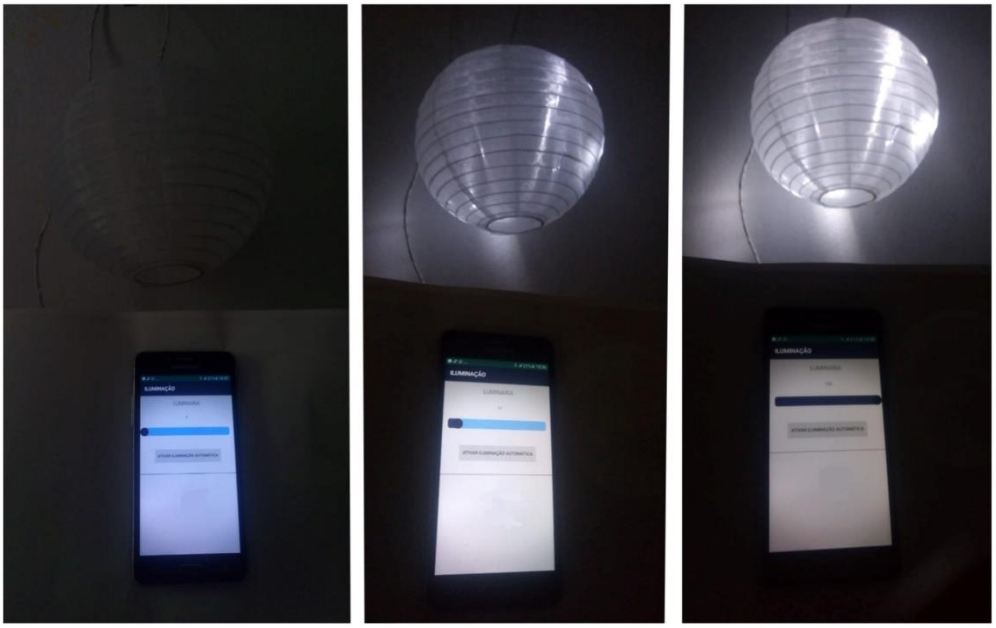
Figura 22 - Teste comandos ar condicionado.



Fonte: Elaborado pelo autor.

O terceiro teste foi da luminária, onde foi testado o acionamento da mesma, desligada, pouca intensidade, e intensidade máxima. A figura 23 apresenta os resultados do teste.

Figura 23 - Teste intensidades luminária.



Fonte: Elaborado pelo autor.

A Tabela 7 apresenta os custos dos equipamentos utilizados para realização do trabalho.

Tabela 7 - Preço dos Componentes.

Quantidade	Descrição do Equipamento	Preço Unitário	Preço Total
1X	ESP32	R$ 43,00	R$ 43,00
1X	ESP8266	R$ 25,00	R$ 25,00
1X	Potenciômetro 100k	R$ 1,80	R$1,80
5X	LDR	R$ 0,50	R$ 2,50
1X	LM35	R$ 4,90	R$ 4,90
5X	LED emissor infravermelho	R$ 1,15	R$ 5,75
2X	LED receptor infravermelho	R$ 2,25	R$ 4,50
10X	LED alto brilho branco	R$ 0,65	R$ 6,25
2X	Armação Luminária	R$ 10,00	R$ 20,00
		TOTAL:	**R$ 113,70**

Fonte: Elaborado pelo autor.

O capítulo 7 apresenta as considerações finais deste trabalho.

7 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as considerações finais do trabalho e as sugestões de trabalhos futuros.

7.2 Conclusão

Perante ao problema de segurança em soluções de automação residencial, este trabalho objetivou proporcionar uma comunicação segura. Apresentou-se como proposta de solução um sistema com *hardware* e *software* de baixo custo, capaz de controlar um aparelho de ar condicionado, televisor e luminária, utilizando o conceito de internet das coisas

A interface com o usuário desenvolvida foi um aplicativo para o sistema operacional Android, que se comunica com a placa ESP32 através de uma rede WiFi presente na residência. Através do algoritmo de criptografia simétrico AES, foi estabelecido uma comunicação segura entre a placa ESP32 e *smartphone*, que permitiu controlar os aparelhos a serem automatizados.

Estudos sobre a placa ESP32 e de aplicações mobile que utilizam o sistema operacional Android, foram essenciais para implementação do algoritmo de criptografia. Testes unitários foram realizados para compreender o comportamento dos hardwares, softwares e funcionalidades utilizadas, por exemplo: emissor infravermelho, PWM, AES, dentre outros.

Após os testes unitários, realizou-se a integração dos componentes modelando o sistema. Os resultados obtidos foram satisfatórios, ressaltando o baixo consumo de energia das placas, o preço total do projeto e a facilidade de utilizá-lo.

As principais dificuldades do projeto, foram a integração de duas plataformas distintas: Android e ESP32, e inicialmente dificuldade de se trabalhar com a linguagem de programação JAVA.

7.2 Trabalhos Futuros

Sugestão para trabalhos futuros:

- Adicionar mais componentes na residência para serem controlados;

- Desenvolver uma placa controladora especifica para o projeto, diminuindo o preço de hardware;
- Implementação de uma rede neural para reconhecimento dos padrões de usuários, para que o controle seja de forma autônoma;
- Incrementar uma plataforma em nuvem com objetivo de ser um banco de dados, que armazenara aparelhos que foram controlados, o dia e a hora.

8 REFERÊNCIAS

ALBERTIN, A. L.; ALBERTIN, R. M. de M. **A INTERNET DAS COISAS IRÁ MUITO ALÉM DAS COISAS.** *GV-executivo*, v. 16, n. 2, março-abril, p.12-17, 2017.

ANDROID STUDIO, **Conheça o Android Studio.** 2018. Disponível em: <https://developer.android.com/studio/intro/>. Acesso em: 12 nov. 2018.

BARROSO, Bruno Lemos. **CRIPTOGRAFIA: Aspectos Teóricos e Proposta de Autenticação Utilizando Dados Cadastrais do Usuário.** 2016. 107 f. Monografia (Especialização) - Curso de Gerência de Redes de Computadores, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2016.

BEGHINI, Lucas Bragazza. **Automação residencial de baixo custo por meio de dispositivos móveis com sistema operacional Android.** 2013. 76 f. TCC (Graduação) - Curso de Engenharia Elétrica Com ênfase em Eletrônica, Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2013. Disponível em: <http://www.tcc.sc.usp.br/tce/disponiveis/18/180450/tce-04022014-152853/?&lang=br>. Acesso em: 02 de maio de 2018

BERTOL, Gustavo de Carvalho. **Sistemas Fotovoltaicos de Geração de Energia: Comparação de Desempenho entre um Sistema com Rastreamento Solar e um Sistema Estático.** 2012. 58 f. TCC (Graduação) - Curso de Engenharia Elétrica, Universidade Estadual Paulista, Bauru, 2012.

BOUFLEUR, Andréia Cardoso et al. CONSTRUÇÃO DE UM PROTÓTIPO DE UM TANQUE PARA PRODUÇÃO DE CHÁ COM CONTROLE DE TEMPERATURA E DE NÍVEL. **Revista de Engenharia e Tecnologia**, Ponta Grossa, v. 6, n. 3, p.196-206, dez. 2014. Disponível em: <http://revistas2.uepg.br/index.php/ret/article/view/11583/209209209558>. Acesso em: 12 nov. 2018.

BRADLEY, J. et al. (2013a) Internet of Everything (IoE) Value Index. White Paper. Cisco Systems, San Jose -CA. Disponível em: < https://www.cisco.com/c/dam/en_us/about/business-insights/docs/ioe-value-index-whitepaper.pdf >. Acesso em: 09 de nov. de 2018.

BURNETT, Steve; PAINE, Stephen. **Criptografia e Segurança O guia Oficial RSA.** Rio de Janeiro: Elsevier, 2002. 373 p

COMPUTER ON THE BEACH, 8., 2017, Florianópolis. **Analise Comparativa do Custo e do Desempenho de um Algoritmo de Criptografia para Sistemas Embarcados Explorando o Particionamento Hardware/Software** Computer On The Beach, 2017. 10 p. Disponível em: <https://siaiap32.univali.br/seer/index.php/acotb/article/view/10584>. Acesso em: 02 de maio de 2018.

CONDEZ, Bruna Tchordach et al. Interconexão segura de dispositivos IoT à nuvem através de redes virtuais privadas (VPNs). **Fasci-tech**, São Caetano do Sul, v. 1, n. 12, p.18-31, abr. 2017. Disponível em: <https://www.fatecsaocaetano.edu.br/fascitech/index.php/fascitech/article/view/107/110>. Acesso em: 2 maio 2018.

COSTA, Celso; FIGUEIREDO, Luiz Manoel. **Introdução à Criptografia.** Rio de Janeiro: Fundação Cecierj / Consórcio Cederj, 2010. 100 p.

FIWARE. **Security considerations.** Disponível em: <https://fiware-orion.readthedocs.io/en/master/user/security/index.html>. Acesso em: 10 nov. 2018.

LEMOS, André. **A comunicação das coisas. Internet das Coisas e Teoria Ator-Rede. Etiquetas de radiofrequência em uniformes escolares na Bahia**. Cyber Arte Cultura. A trama das Redes. Salvador, 2013, p 18-47.

MACHADO, Wictor Junqueira. **AUTOMAÇÃO RESIDENCIAL COM CONTROLE POR MEIO DE APLICATIVO ANDROID.**2017. 103 f. TCC (Graduação) - Curso de CiÊncia da Computação, Pontifícia Universidade Católica de Goiás, Goiânia, 2017.

MCROBERTS, Michael. **Arduino Básico.** São Paulo: Novatec, 2011. 456 p.

MORENO, Edward David; PEREIRA, Fabio Dacêncio; CHIARAMONTE, Rodolfo Barros. **Criptografia em Software e Hardware.** Novatec, 2005. 294 p.

SENAI et al. **Sistemas Digitais.** São Paulo: Senai-sp, 2015. 140 p.

WANZELER, Tiago; FÜLBER, Heleno; MERLIN, Bruno. **Desenvolvimento de um sistema de automação residencial de baixo custo aliado ao conceito de Internet das Coisas (IoT).** In: SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES E PROCESSAMENTO DE SINAIS, 34, 2016, Santarém. p. 40 - 44. Disponível em: <http://www.sbrt.org.br/sbrt2016/sessoes/st1-4.php#08>. Acesso em: 02 de maio de 2018.