Arquitetura para gerenciamento de dispositivos através de assistentes virtuais comandados por voz

Honoré Vicente Cesário[1], Gustavo Girão[1], André Riker[2], Bruno L. Dalmazo[3], Roger Immich[1]

[1] Instituto Metrópole Digital
Universidade Federal do Rio Grande do Norte (UFRN)

[2]Faculdade de Computação
Universidade Federal do Pará (UFPA)

[3]Centro de Ciências Computacionais
Universidade Federal do Rio Grande (FURG)

honore.cesario@ufrn.br, girao@imd.ufrn.br,
afr@ufpa.br, dalmazo@furg.br, roger@imd.ufrn.br

Abstract. *Management and configuration of home networks has become increasingly complex. One of the reasons for this is due to the increase in the number of devices connected to your infrastructure. The great variety of manufacturers and lack of standardization in the available features can lead users to make wrong configurations. Performing these settings can be even more challenging for people who have a disability. In view of this, the present work proposes a layered architecture, where it will be possible, by means of voice or text commands and using natural language, that complex configurations are carried out, without the need for advanced technical knowledge regarding the infrastructure or specificities of the network device.*

Resumo. *O gerenciamento e configuração das redes domésticas tem se tornado cada vez mais complexa. Um dos motivos para isto é devido ao aumento no número de dispositivos conectados a sua infraestrutura. A grande variedade de fabricantes e falta de padronização nas funcionalidades disponibilizadas, podem levar os usuários a efetuar configurações equivocadas. Realizar essas configurações pode ser ainda mais desafiador para pessoas que apresentem algum tipo de deficiência. Diante disso, o presente trabalho propõe uma arquitetura em camadas, onde será possível mediante comandos por voz ou texto e utilizando linguagem natural, que configurações complexas sejam realizadas, sem que seja necessário conhecimento técnico avançado relativo a infraestrutura ou especificidades do dispositivo de rede.*

1. Introdução

De acordo com dados do IBGE, 84,4% da população brasileira vive em áreas urbanas [IBGE 2010]. Com isso, cresce a demanda por diversos serviços, como transportes, habitação e energia. Nesse contexto, devido a utilização cada vez maior de dispositivos IoT nas redes domésticas, estas podem contribuir para o provimento de mais informações relativas ao comportamento dos usuários, auxiliando no processo de construção de cidades mais inteligentes.

O processo de estruturação de uma rede doméstica sempre exigiu algum nível de conhecimento básico do usuário para sua instalação inicial. Realizar a instalação da rede doméstica e seus equipamentos tende a se tornar uma tarefa mais necessária nos próximos anos visto que, de acordo com pesquisa apresentada por [Titara 2021], 80% das pessoas de classe média pretendem tornar sua casa mais conectada. Além da tendência de

popularização das redes residenciais [Rodrigues et al. 2019], existe também a tendência de haver dispositivos com cada vez mais recursos [Immich et al. 2019]. Dessa forma, existe uma crescente exigência de habilidades e conhecimentos para configurar corretamente dispositivos de redes domésticas.

Configurações complexas feitas por usuários leigos em tecnologia é um cenário altamente propício a falhas de configuração. Além disso, existem inúmeras razões que impactam na dificuldade encontrada em realizar a configuração de equipamentos. Uma das principais, está relacionada às telas de instalação apresentadas aos usuários, onde cada fabricante utiliza uma interface distinta e com nomenclaturas técnicas diferentes para uma mesma função. Além disso, muitas vezes, apresentam uma interface completamente diferente de um modelo para outro [Fiorenza et al. 2021].

Um outro fator que pode aumentar a dificuldade na configuração das redes domésticas atinge uma relevante parcela da população, que são as pessoas que apresentam algum tipo de deficiência física. Dados do IBGE (Instituto Brasileiro de Geografia e Estatística), mostram que mais de 45 milhões de pessoas (23,9% da população) possuem algum tipo de deficiência, onde 36 milhões relatam ter algum tipo de deficiência visual [IBGE 2010]. Essas estatísticas reforçam a necessidade de novas propostas na literatura que promovam um ambiente inclusivo para pessoas com esse tipo de deficiência.

Outros aspectos também podem dificultar a configuração de redes. Isso ocorre quando são exigidas informações de baixo nível como endereçamento, protocolos utilizados e portas de comunicação para que uma regra de bloqueio de um serviço ou acesso seja desabilitado em um dispositivo, por exemplo. Devido ao número crescente de dispositivos IoT nas redes (além dos dispositivos já existentes como *notebooks* e *smartphones*), tanto corporativas como domésticas, o desafio de configurá-los só tende a aumentar com o passar do tempo.

Existem diversas soluções propostas na literatura para facilitar a configuração de redes domésticas, tornando essa tarefa mais intuitiva. Recentemente, o uso de linguagem natural tem emergido como um meio de facilitar a interação entre sistemas computacionais e o usuário. Nesse contexto, existem soluções que fazem uso do processamento de linguagem natural para disponibilizar aos usuários assistentes virtuais, tal como Alexa e Google Home Assistant, e assim permitir que usuário realize tarefas computacionais por meio de comandos de voz.

Para lidar com todas essas lacunas e tornar configurações de dispositivos de rede doméstica mais acessíveis e simples aos usuários sem conhecimentos técnicos avançados, esse trabalho propõe uma arquitetura em camadas, composta por módulos independentes, utilizando uma linguagem de intenções para padronizar os comandos para configuração de dispositivos gerenciáveis. Adicionalmente, um protótipo foi utilizado para validação da arquitetura.

Nesse sentido, para tornar as configurações menos complexas, é de grande importância que menos esforço seja empregado em cada dispositivo. Para tal, podemos instanciar funcionalidades de reconhecimento de comandos por voz em conjunto com o processamento de linguagem natural (PLN). A linguagem de intenções abstrai parâmetros específicos na solicitação expressa pelo usuário. Com isso, essas informações são correlacionados aos termos da linguagem de configuração do dispositivo para que os comandos correspondentes sejam gerados. Dessa forma, caso o dispositivo em questão seja substituído, por exemplo, basta que esses parâmetros sejam relacionados a linguagem de configuração desse novo dispositivo.

O restante desse trabalho foi divido conforme descrito abaixo. A Seção 2 apresenta os trabalhos relacionados. A arquitetura proposta é apresentada na Seção 3. A implementação e cenário de testes e resultados, respectivamente, nas Seções 4 e 5. Finalizando com as considerações finais na Seção 6.

2. Trabalhos relacionados

Na literatura existem várias abordagens com a finalidade de automatizar o acionamento de dispositivos e suas configurações. Nesse sentido, observa-se que soluções baseadas no uso do processamento da linguagem natural para configurações de dispositivos têm cada vez mais apelo da comunidade científica. Os trabalhos relacionados são apresentados em dois grupos: abordagens com monitoramento e acionamento de dispositivos por voz; e trabalhos baseados em linguagem de intenções para configurações.

2.1. Monitoramento e acionamento de dispositivos

Existem propostas que utilizam assistentes virtuais na função de interface conversacional. Entre estes, podemos citar como por exemplo a adoção do Google Home Assistant [Lago et al. 2021, Noruwana et al. 2020] e da Amazon Alexa [Austerjost et al. 2018]. A inclusão destes dispositivos busca possibilitar que a interação com o usuário ocorra de através de comandos por voz de forma natural. A arquitetura de *software* apresentada por [Longo et al. 2019], utiliza as redes sociais (como Facebook e Twitter), para que o usuário possa interagir com seus dispositivos inteligentes.

Smartphones ou dispositivos do tipo Raspberry são apresentados por [Rani et al. 2017, Alexakis et al. 2019, Kadali et al. 2020, Jivani et al. 2018, Hamdan et al. 2019], como dispositivos de entrada para recepção dos comandos, utilizando (em sua maioria) *chatbots* para interação com o usuário.

Trabalhos de monitoramento/acionamento, por apresentarem funcionalidades menos complexas, tem como característica a não utilização de uma camada de abstração utilizando uma linguagem de intenções. Na maioria dos casos, os trabalhos baseiam-se em assistentes de voz existentes (Google Home e Alexa, por exemplo), *chatbots* e regras usando o IFTTT (*If This Then That* - Se isso acontecer, então faça aquilo) para agendamento de eventos. A Tabela 1 resume as principais características de cada um dos trabalhos.

2.2. Configuração de dispositivos

Um processo de refinamento de intenções utilizando *chatbots*, baseado no *feedback* dos operadores de rede é proposto por [Jacobs et al. 2018]. Com isso, pretendem colaborar para o desenvolvimento de redes autônomas. O sistema Jingjing, apresentado por [Tian et al. 2019], permite que os operadores de rede expressem em uma linguagem de alto nível, nos comandos para atualização de suas ACLs (*Access Control Lists*), contribuindo com isso no aumento da eficiência e acerto nessas atualizações.

O *framework* P4I/O, foi desenvolvido por [Riftadi and Kuipers 2019], para redes baseadas em intenções e voltado a configuração de *switches*. Através dessa implementação, os comandos solicitados pelos usuários por meio de intenções são traduzidas para codificação P4 e as configurações enviadas para implementação nos *switches*. O protótipo apresentado por [Alsudais and Keller 2017], fornece uma camada intermediária de comunicação entre um usuário utilizando linguagem natural e diferentes sistemas de gerenciamento de redes. O sistema coleta e analisa tarefas comuns de gerenciamento em uma camada de abstração, encapsulando essas tarefas.

Tabela 1. Automação de acionamento/monitoramento

Autores	Entrada	Interface	Funcionalidades
[Lago et al. 2021]	Voz Texto	Google Home Assistant Aplicação Mobile	Acionamento Agendamento
[Longo et al. 2019]	Voz Texto	Redes sociais (Ex. Facebook, Twitter, Telegram)	Acionamento Agendamento
[Rani et al. 2017]	Voz	Aplicação Mobile	Acionamento
[Alexakis et al. 2019]	Voz Texto	Aplicação Web / Chatbot	Acionamento Monitoramento
[Noruwana et al. 2020]	Voz	Google Home Assistant Aplicação Mobile	Acionamento Monitoramento Alerta de intrusão
[Kadali et al. 2020]	Voz Texto	Telegram / Chatbot	Acionamento
[Jivani et al. 2018]	Voz	Raspberry Pi	Acionamento
[Hamdan et al. 2019]	Voz Texto	Raspberry Pi Aplicação Mobile	Acionamento
[Austerjost et al. 2018]	Voz	Amazon Alexa	Acionamento Leitura de dados

O *framework* Lumi, desenvolvido por [Jacobs et al. 2019], permite que operadores de rede possam expressar suas intenções de configuração em uma linguagem natural. Um *feedback* das ações realizadas é fornecido com o intuito de melhorar o aprendizado da solução com o passar do tempo.

A Tabela 2 apresenta os trabalhos que fazem uso de linguagem de intenções. Nestes, é possível perceber uma necessidade mais evidente na utilização de uma camada de abstração para proporcionar maior agilidade na alteração de configurações, redução na possibilidade de erros nas implementações de regras e rotas, além de diminuir a complexidade na manutenção dessas operações, por exemplo. Evitando com isso, que os usuários tenham que trabalhar com as especificidades de mais baixo nível, onde teriam que lidar com linguagens de programação específicas de diferentes fabricantes.

Tabela 2. Automação de configurações

Autores	Dados aceitos	Interface	Funcionalidades
[Jacobs et al. 2018]	Texto	Aplicação Web / Chatbot	Configuração de redes
[Tian et al. 2019]	Texto	Programa Jingjing	Configuração de ACLs
[Riftadi and Kuipers 2019]	Texto	Programa P4	Configuração de redes
[Alsudais and Keller 2017]	Texto	Floodlight	Configuração de redes
[Jacobs et al. 2019]	Texto	Programa Lumi / Chatbot	Configuração de redes

Uma característica comum identificada entre os trabalhos analisados, está o fato de todos aceitarem como dados de entrada apenas a inserção de texto. Muito provavelmente tal fato deve-se à complexidade envolvida na adição de entrada por voz para determinados tipos de configurações, pois devido ao fato dos trabalhos apresentados nesta subseção apresentarem ênfase em ambientes corporativos (diferentemente da solução apresentada por este trabalho, que tem foco em redes domésticas), a implementação da voz como um dado de entrada pode ser limitado. Por outro lado, nas redes domésticas, a infraestrutura de comunicação utilizada é bastante simples quando comparada a existente nas empresas. Diante disso, o processo para inserção da voz como opção para entrada dos dados em um

ambiente doméstico torna-se menos complexo.

É importante destacar que, em determinadas situações e para determinados grupos de usuários (como os portadores de algum tipo de deficiência visual, por exemplo), o fato de existir a possibilidade de ser aceita a voz como dado de entrada, é extremamente importante, e em muitos casos, determinante para o uso ou não de uma ferramenta.

2.3. Lacunas em aberto

Nos trabalhos apresentados que eram voltados para o monitoramento e acionamento de dispositivos, percebeu-se a importância na escolha de um extrator de conceitos que tenha uma maior abrangência nas linguagens suportadas e que apresente uma base de dados ampla o suficiente. Com isso, não limitaria muito a forma que os usuários podem expressar seus comandos, tentando tornar o processo de envio da solicitação o mais próximo possível de uma comunicação natural.

Verificou-se também a necessidade de mecanismos que garantam uma usabilidade mínima de recursos em casos de falta de conexão com a Internet e questões relacionadas à segurança, onde os dispositivos apresentem algum mecanismo para que não permaneçam o tempo todo com o modo de escuta ativo, ouvindo os usuários, causando assim, problemas de privacidade. Além da possibilidade de implementação de algum mecanismo de autenticação para acesso à utilização dos serviços apenas por pessoas e dispositivos autorizados [Fernandes et al. 2020].

Os trabalhos propostos com foco na configuração de dispositivos exigem um grau de conhecimento técnico elevado, além de demandarem um conhecimento mínimo da arquitetura proposta e forma de funcionamento da solução. Esses fatores tendem a limitar a utilização da aplicação apenas a grupos específicos de usuários. Utilizam também uma base de dados pequena (com poucas sentenças de treinamento para o reconhecimento dos comandos enviados), contando com o *feedback* do usuário para aprimoramento da assertividade no reconhecimento das intenções. Com isso, prejudicam a usabilidade da solução e limitam seu uso apenas para usuários detentores de um maior conhecimento técnico.

Objetivando preencher algumas das lacunas mencionadas, a arquitetura proposta faz uso do reconhecimento de comandos através da voz em conjunto com uma linguagem de intenções. Garantindo comodidade ao usuário e uma camada de abstração para redução na complexidade das operações pretendidas, além de prover acessibilidade.

3. Proposta de uma arquitetura para gerenciamento de dispositivos

A arquitetura proposta tem como principal objetivo prover facilidade na configuração de dispositivos de diversos fabricantes, através de uma camada de abstração que permita ao usuário sua configuração sem necessariamente ter conhecimento de sua arquitetura, especificidades das interfaces de gerenciamento e nomenclaturas de funcionalidades, por exemplo, que são diferentes em cada fabricante. Adicionalmente propomos a implementação de uma camada para que as operações desejadas possam ser enviadas também através de comandos por voz.

A Figura 1 apresenta a arquitetura que é divida em seis camadas. Estas são flexíveis, possibilitando que outras tecnologias e protocolos possam ser utilizados de acordo com a necessidade do usuário. A coluna da esquerda apresenta o nome da camada e a coluna da direita as possibilidades de tecnologias a serem utilizadas para sua implementação.

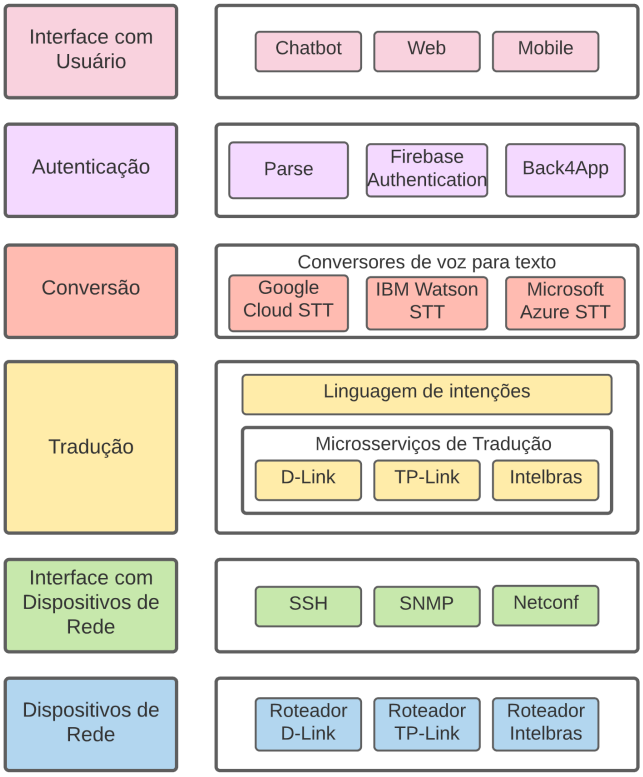

Figura 1. Arquitetura proposta

A arquitetura proposta visa permitir que configurações complexas possam ser efetuadas, utilizando uma linguagem de alto nível. Ela é formada por camadas compostas por módulos, onde módulos podem ser adicionados, caso novas funcionalidades necessitem ser implementadas. Podemos citar como um exemplo o acréscimo de um módulo de microsserviço de tradução de um outro fabricante. Dependendo da camada a ser adaptada, alguns ajustes serão necessários. Por exemplo, caso o dispositivo de rede seja substituído, o microsserviço de tradução com a sintaxe correspondente a esse novo dispositivo deverá ser adicionado à camada de tradução.

Para facilitar o entendimento, as camadas serão apresentadas seguindo metodologia Top-Down. Na camada de ***interface com usuário*** são apresentadas as tecnologias para comunicação que podem ser utilizadas pelo usuário para solicitar a configuração dos dispositivos, através de uma linguagem natural (de alto nível). De acordo com a opção escolhida, a comunicação pode ocorrer através de uma *interface web*, aplicação para *mobile*, ou até mesmo via *chatbot*.

O objetivo da camada de ***autenticação*** é prover o controle de acesso necessário, garantindo que as alterações efetuadas sejam realizadas por quem efetivamente possui o privilégio devido. Dependendo da complexidade das alterações que poderão vir a ser providas pela solução, níveis de permissão podem ser estabelecidos, para que determinadas configurações possam ser realizadas apenas por usuários com níveis de privilégios diferenciados. Por exemplo, na configuração de uma residência, as regras para bloqueio/liberação para acesso à Internet pelos dispositivos, exigiria um grau de permissão mais elevado (nível de acesso para configuração) e exclusivo para os pais, de forma a impedir o controle dessa configuração por crianças.

Na camada de ***conversão***, caso a entrada dos dados feita pelo usuário ocorra via mensagem de voz, seu conteúdo será convertido em texto para posteriormente ser tratado

pela camada de *tradução*. Algumas das ferramentas comerciais disponíveis mais conhecidas para realizar a conversão dos comandos de voz para texto, são: Cloud Speech To Text (Google), *Watson Speech To Text* (IBM) e *Azure Speech To Text* (Microsoft).

A camada de ***tradução*** apresenta o módulo de *linguagem de intenções*, que tem a função de realizar a conversão entre a linguagem natural e uma linguagem intermediária, fazendo a ponte com a linguagem de mais baixo nível dos equipamentos e apresentando para o módulo que segue logo abaixo na arquitetura, as intenções do usuário. Ainda nessa camada, o módulo de *microsserviços de tradução* é responsável pela tradução das informações repassadas pelo módulo anterior, dentro da camada, para a codificação de comandos específicos de acordo com cada modelo de equipamento.

O objetivo da camada de ***interface com dispositivos de rede***, é ligar a camada de *tradução* com a camada de *dispositivos de rede*. As informações produzidas pelos microsserviços de tradução são enviadas para o equipamento, utilizando para isso algum dos protocolos de comunicação apresentados. A escolha desse protocolo pode variar de acordo com as características do dispositivo, como por exemplo, do seu suporte (ou falta dele) a algum protocolo de comunicação específico.

Por último, na camada ***dispositivos de rede***, estão presentes os dispositivos que receberão os comandos solicitados pelos usuários. Os módulos presentes nessa camada devem ter seus microsserviços de tradução correspondentes implementados para que possa ser feita a validação da arquitetura proposta. Os dispositivos de rede em questão podem ser quaisquer equipamentos gerenciáveis que forneçam suporte a recebimento de configurações por linha de comandos.

4. Implementação

Foi projetado e implementado um protótipo para avaliação e viabilidade da arquitetura proposta. Como a arquitetura é modular e flexível, diversas outras possibilidades de ferramentas e tecnologias podem ser utilizadas, em substituição as que serão propostas, sem impactar no resultado final. O fluxo dos dados para o refinamento das intenções solicitados pelos usuários e as tecnologias e ferramentas que foram utilizadas para sua implementação, podem ser observados na Figura 2.

A aplicação responsável pela interface conversacional com o usuário foi desenvolvida com o *framework* Flutter[1]. Ao entrar na aplicação, para ter acesso ao *chatbot*, o usuário deverá realizar a autenticação no banco de dados que está hospedado no *Firebase Authenticator*.

No momento da interação com o *chatbot*, o protótipo proporciona ao usuário a possibilidade de envio de suas solicitações de configuração por comandos textuais ou por voz. A interação permanecerá ativa até que sejam coletadas todas as informações necessárias para que a intenção do usuário seja identificada.

Após confirmação com o usuário que a intenção identificada pelo *chatbot* está correta, evitando com isso possíveis problemas relativos a erros na execução da solicitação, a interação é finalizada confirmando que o comando será executado. Estas informações são enviadas via *webhook*, para uma API do tipo REST, que será responsável pela tradução das intenções para a linguagem de configuração de baixo nível do dispositivo de rede.

Em seguida, após o comando correspondente a intenção do usuário estar devidamente estruturado, este deverá ser enviado para o dispositivo. Esse procedimento é

[1]Flutter: https://docs.flutter.dev/

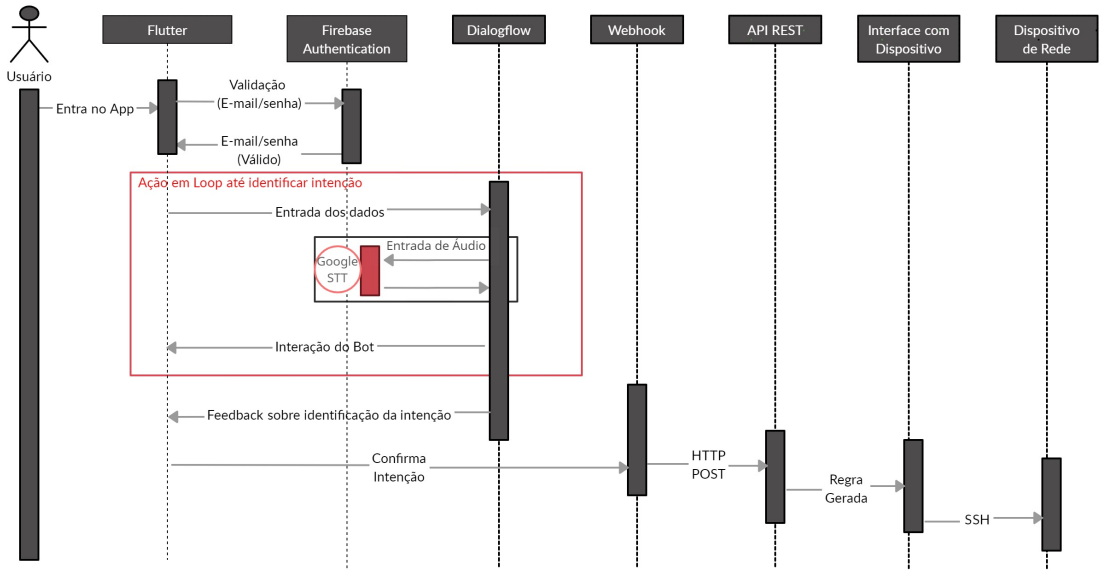

Figura 2. Diagrama de Sequência

realizado através do uso de conectores. Para o protótipo em questão será utilizado um conector SSH para envio dos comandos para o roteador com o *firmware* do DD-WRT. Para implementação do conector SSH que permite a conexão com o roteador para execução dos comandos foi utilizada a biblioteca Python, Paramiko[2]. Essa biblioteca estabelece as conexões via SSH, utilizando para isso informações de *login* e senha, IP e porta de conexão com o dispositivo de rede.

É importante ressaltar que cada roteador e/ou dispositivo tem uma sintaxe para execução de comandos e *firmware* próprios. O processo de tradução das intenções para comandos que serão recepcionados pelo dispositivo é diretamente dependente desses fatores. Roteadores com o *firmware* DD-WRT, utilizam o IPTables como aplicação padrão de *firewall*. O IPTables é um programa de código aberto utilizado para implementar filtragem de pacotes mediante análise de conjuntos de regras definidas pelo usuário [Netfilter 2021]. A sintaxe de um comando IPTables é basicamente formada da maneira apresentada abaixo [Andreasson et al. 2001].

iptables [-t table] command [match] [target/jump]

Após o comando inicial *iptables*, é inserida a *table* (que são compostas por *chains*), seguido pelo *match* e sendo finalizado com o *target*. A Tabela 3 apresenta as *chains* de uma das tabelas do *firewall* IPTables. As *matches* especificam as condições que devem ser atendidas para execução de determinado comando. Alguns tipos mais utilizados são apresentados na Tabela 4. Por último, *targets* que utilizados para definir a ação que o comando deve executar no pacote, sendo demonstrado algumas opções de uso na Tabela 5.

No protótipo apresentado, dispositivos IoT e também os dispositivos convencionais (que podemos chamar de dispositivos não IoT) como *notebooks* e *smartphones*, por exemplo, também poderão ser objetos de uso/controle pelos comandos gerados. Na Tabela 6 são apresentados os requisitos necessários para que o comando possa ser gerado em

[2]Paramiko: https://www.paramiko.org/

Tabela 3. *Chains*

Termo	Função
INPUT	Entrada de dados
OUTPUT	Saída de dados
FORWARD	Regras do Firewall para host na rede

Tabela 4. *Matches*

Termo	Função
-L	Lista Regras
-F	Apaga Regras
-s	IP de Origem
-d	IP de Destino
-o	Interface de origem do pacote
-j	Ação a ser realizada
-p	Protocolo do pacote
–dport	Porta de destino
–sport	Porta de origem

Tabela 5. *Targets*

Termo	Função
DROP	Bloqueia os pacotes (não avisa a origem)
REJECT	Rejeita os pacotes (avisa a origem)
ACCEPT	Libera passagem dos pacotes
DNAT	Reescreve o IP de destino do pacote
SNAT	Reescreve o IP de origem do pacote

sua completude. Além disso, também são apresentadas as intenções esperadas, onde estão presentes alguns exemplos de termos que podem ser empregados pelo usuário para fazer uma solicitação e a sintaxe correspondente (baseada em um *firewall* que utilize IPTables como solução para filtragem de pacotes IP).

Tabela 6. Intenção de bloqueio em dispositivo IoT

Requisitos	Intenções esperadas	Sintaxe
host	fechadura, trava	-s 192.168.0.170/24
location	sala, entrada da casa	
service	externo, exterior	-m multiport –dports 80,443
traffic	bloquear, negar, não permitir	-j REJECT

Um exemplo real de uma interação correspondente ao apresentado na Tabela 6, com uma fechadura inteligente, onde o usuário solicitaria após sua chegada em casa, por exemplo, que não fosse permitido que essa fechadura pudesse ter sua abertura acionada externamente. Dessa forma as informações abaixo que estão em negrito, identificadas como entidades pelo *Dialogflow*, seriam extraídas do contexto e traduzidas para a linguagem de baixo nível do dispositivo. O comando gerado seria o resultado esperado para a solicitação realizada pelo usuário.

Comando de voz:

"***Não permita*** *o acesso* ***externo*** *a* ***fechadura*** *da* ***sala***"

Comando gerado:

iptables -I FORWARD -p tcp -s 192.168.0.170/24 -m multiport –dports 80,443 -j REJECT

A Tabela 7, de forma semelhante ao já apresentado anteriormente, apresenta os

requisitos, as interações relacionadas aos requisitos exigidos e a sintaxe correspondente para execução de uma regra para bloqueio da navegação em um dispositivo não IoT.

Tabela 7. Intenção de bloqueio na navegação em dispositivos não IoT

Requisitos	Intenções esperadas	Sintaxe
host	pc, computador	-s 192.168.0.180/24
location	sala, quarto	
service	Internet, navegação,	-m multiport –dports 21,80,443
traffic	bloquear, não permitir, negar	-j REJECT

Uma forma de interação do usuário com o sistema, poderia ser o usuário fazendo a seguinte solicitação para bloquear o acesso do computador da sala à Internet. Onde a saída correspondente após extração e tratamento dos dados informados pelo usuário é apresentada abaixo.

Comando de voz:

"Por favor, ***bloquear*** *o acesso a* ***Internet*** *no* ***computador*** *da* ***sala****"*

Comando gerado:

iptables -I FORWARD -p tcp -s 192.168.0.180/24 -m multiport –dports 21,80,443 -j REJECT

Após isso, o comando gerado pelo tradutor será encaminhado para conector suportado pelo dispositivo para seu envio e execução.

5. Cenário de testes e resultados obtidos

Com o objetivo de validar o funcionamento do protótipo, um laboratório virtual de testes foi montado. Foi utilizado o *software* de virtualização da Oracle, VirtualBox[3]. A forma que a infraestrutura está disposta no laboratório de testes pode ser observada na Figura 3. O *host* que foi utilizado para montar o laboratório é uma máquina com o SO Windows 10, equipado com um processador Core i3-8130, 12 GB de memória RAM e um SSD de 240GB. Essa máquina é responsável por prover a infraestrutura para armazenamento das máquinas virtuais (VMs).

A comunicação do usuário com o protótipo desenvolvido ocorre através de uma aplicação *mobile*. O *chatbot* ***A****utomating configura****T****ion of* ***E****ndpoints over* ***N****etwork* ***A****mbiences* (ATENA), realiza através do Dialogflow a interação com o usuário. Um exemplo de comunicação utilizando o *chatbot* pode ser observado na Figura 4.

Para que as informações possam ser enviadas do Dialogflow para o servidor que realizará o refinamento das intenções é necessário que o servidor permita conexão externa pela porta HTTPS (443). Para isso, foi utilizada a ferramenta Ngrok[4], que permite o acesso externo a serviços locais através de túneis seguros. Dessa forma, toda troca de informações com o *chatbot*, intermediadas pelo Dialogflow, podem ser enviadas para o servidor. Esse servidor será responsável pelo refinamento das intenções solicitadas pelo usuário.

Em um primeiro momento as informações enviadas do Dialogflow para o servidor são, em sua grande maioria, desnecessárias para que a intenção do usuário seja identificada. Na Figura 5, podem ser observado os dados enviados sem que haja nenhum

[3]VirtualBox - https://www.virtualbox.org/

[4]Ngrok - https://ngrok.com/

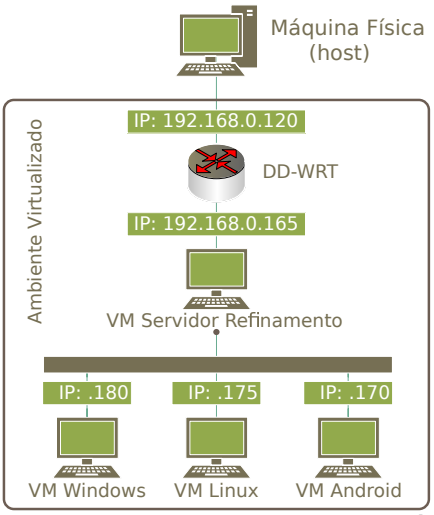

Figura 3. Infraestrutura do Laboratório

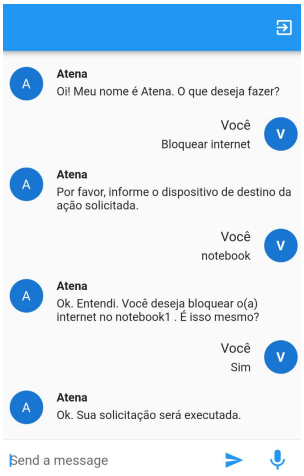

Figura 4. *Chatbot* Atena

filtro para tratamento dos dados. Percebe-se que diversos parâmetros podem ser suprimidos para que apenas a intenção do usuário seja visualizada. Após realizar um filtro nas informações relevantes, a Figura 6, mostra o resultado da saída obtido, onde são apresentados apenas os parâmetros necessários para que o comando solicitado pelo usuário possa ser executado.

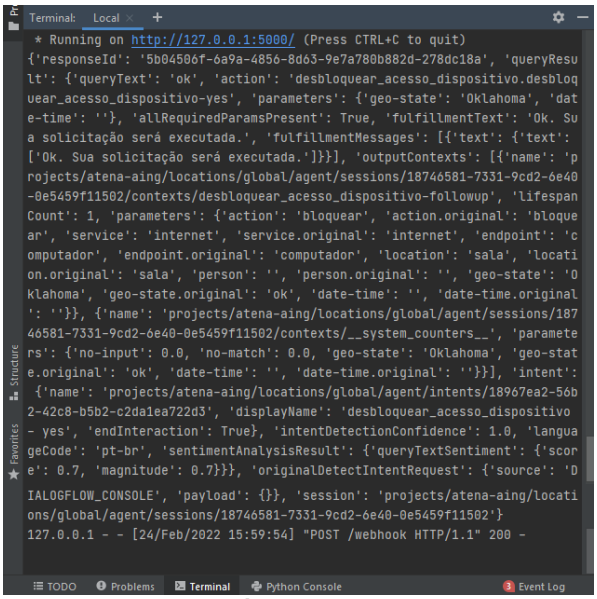

Figura 5. Saída de dados original

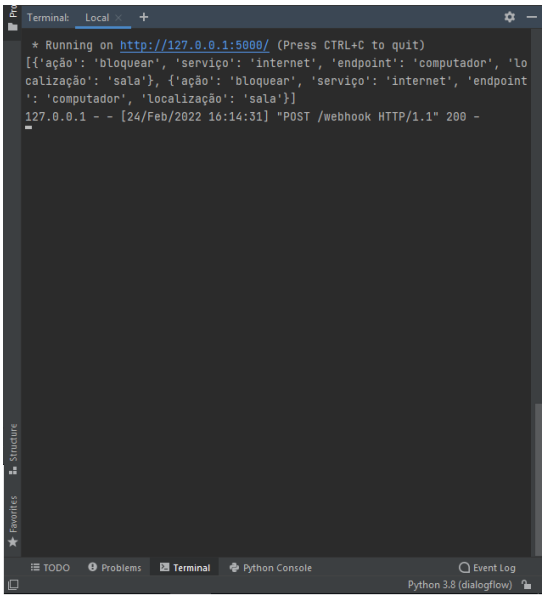

Figura 6. Saída de dados filtrada

Baseado nos parâmetros reconhecidos na etapa anterior, a intenção do usuário é identificada e o comando correspondente ao solicitado é executado. Para monitoramento do estado da conexão, possibilitando que a aplicação da regra de bloqueio seja visualizada, será utilizado o programa Wireshark[5].

A Figura 7 exibe o estado da conexão no momento em que a mesma é bloqueada. Primeiramente, é possível observar como é estabelecida uma conexão do protocolo TCP (1), conhecida como *Three-way Handshake*. Nesse momento a regra de bloqueio não está

[5]Wireshark: https://www.wireshark.org/

ativa e a comunicação acontece normalmente. Esse mecanismo de comunicação funciona da seguinte forma: a máquina virtual (cliente) envia um pacote de sincronização SYN, o *site* acessado (servidor) retorna com um pacote SYN+ACK e o cliente retorna um pacote ACK, estabelecendo a comunicação. No momento em que o roteador (IP: 192.168.0.120)

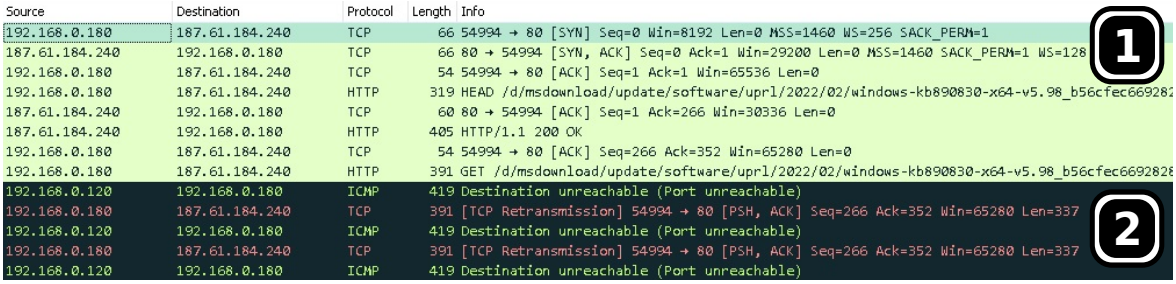

Figura 7. Estado da conexão durante aplicação de restrições

aplica a regra de bloqueio solicitada (Etapa 2 da figura), o acesso à porta HTTP é negado e a navegação interrompida. Importante destacar que, o comando enviado bloqueia apenas portas específicas para cada tipo de solicitação, como foi exemplificado na Tabela 7. Dessa forma, dentro da infraestrutura da rede doméstica, o acesso ao dispositivo permanece disponível.

Para avaliar a usabilidade do protótipo, 80 (oitenta) solicitações contemplando regras de bloqueio e desbloqueio de acesso a Internet, além de filtro e remoção de filtro de acesso a domínios específicos foram executadas. No primeiro teste foram executadas 40 (quarenta) solicitações com apenas 50% dos parâmetros necessários para identificar a intenção do usuário. O restante dos parâmetros foram obtidos mediante interação com o protótipo e seus resultados podem ser observados na Figura 8. No segundo teste foram executadas mais 40 (quarenta) solicitações, desta vez contando já na primeira interação com o *chatbot*, com 100% dos parâmetros necessários para a descoberta da intenção. Os resultados da avaliação podem ser visto na Figura 9.

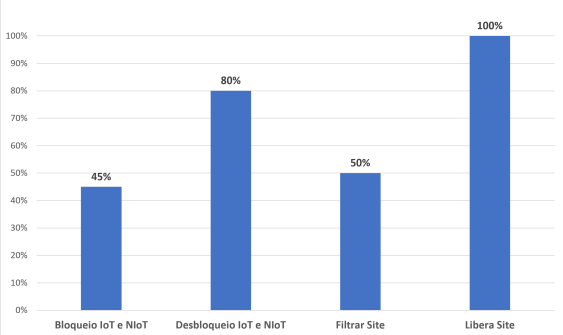

Figura 8. Assertividade com 50% dos parâmetros

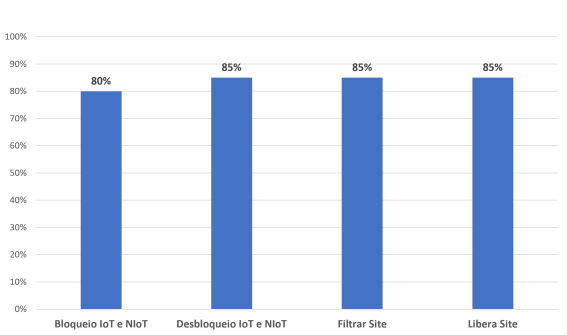

Figura 9. Assertividade com 100% dos parâmetros

Observa-se nos resultados que a assertividade média nos testes realizados com 50% dos parâmetros foi de 68,75%, com uma variação (±25,94%) na assertividade, devido ao baixo índice de acerto em algumas das regras. Quando avaliamos os testes realizados com 100% dos parâmetros, a assertividade média foi de 83,75% de intenções identificadas corretamente. Além disso, apresentou um desvio padrão de apenas ±2,5%, o que demonstra que quando utilizados todos os parâmetros, a variação na assertividade apresentada pelo *chatbot* é mínima.

6. Conclusão

Este trabalho apresentou uma arquitetura base para desenvolvimento de soluções de configuração de dispositivos de rede doméstica através do comando de voz. O aumento no número de dispositivos nas redes domésticas aliado com a exigência cada vez maior em questões relacionadas à segurança e a necessidade em proporcionar um aumento na acessibilidade, indicam que é cada vez mais necessário que sejam disponibilizadas aos usuários ferramentas que possibilitem que configurações mais complexas possam ser realizadas através de uma linguagem natural.

A arquitetura proposta mostrou-se viável após os testes realizados com o protótipo desenvolvido, onde apresentou em um dos cenários testados uma assertividade média superior a 83%. O *chatbot* recebe as solicitações do usuário, por voz ou texto (em linguagem natural), identifica sua intenção e a converte para a linguagem de baixo nível do dispositivo de rede, enviando o comando correspondente para execução, tudo isso de forma transparente para o usuário. A ferramenta apresentou um excelente resultado, sendo capaz de suportar vários comandos, como por exemplo: configuração de regras de bloqueio de acesso de dispositivos a Internet ou portas de comunicação específicas; e bloqueio de domínios.

Como trabalhos futuros, pretendemos adicionar na arquitetura uma camada responsável pela descoberta e registro de forma automática de novos dispositivos que sejam conectados à rede. Além disso, um funcionalidade de *rollback* pode ser implementada para possibilitar que o último comando solicitado pelo usuário seja desfeito. Outras funcionalidades que serão exploradas, são alinhadas aos aprimoramentos de garantia da privacidade no tratamento de uso das informações coletadas e segurança dos dados dos usuários.

Referências

Alexakis, G., Panagiotakis, S., Fragkakis, A., Markakis, E., and Vassilakis, K. (2019). Control of smart home operations using natural language processing, voice recognition and iot technologies in a multi-tier architecture. *Designs*, 3(3).

Alsudais, A. and Keller, E. (2017). Hey network, can you understand me? In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 193–198.

Andreasson, O. et al. (2001). Iptables tutorial 1.2. 2. *Copyright© 2001–2006 Oskar Andreasson, GNU Free Documentation License*.

Austerjost, J., Porr, M., Riedel, N., Geier, D., Becker, T., Scheper, T., Marquard, D., Lindner, P., and Beutel, S. (2018). Introducing a virtual assistant to the lab: A voice user interface for the intuitive control of laboratory instruments. *SLAS Technology*, 23:476 – 482.

Fernandes, R., Paz, G., Kretuz, D., Mansilha, R., Jenuario, T., and Immich, R. (2020). S3as: uma solução de autenticação e autorização através de aplicativos de smartphones. *Revista Eletrônica Argentina-Brasil de Tecnologias da Informação e da Comunicação*, 3(1).

Fiorenza, M., Kreutz, D., Mansilha, R., Macedo, D., Feitosa, E., and Immich, R. (2021). Representação e aplicação de políticas de segurança em firewalls de redes híbridas. In *Anais do XXXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 490–503, Porto Alegre, RS, Brasil. SBC.

Hamdan, O., Shanableh, H., Zaki, I., Al-Ali, A. R., and Shanableh, T. (2019). Iot-based interactive dual mode smart home automation. In *2019 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–2.

IBGE (2010). Censo demográfico 2010. Disponível em: https://biblioteca.ibge.gov.br/visualizacao/periodicos/94/cd_2010_religiao_deficiencia.pdf. Acesso em: 25 de jun. de 2021.

Immich, R., Villas, L., Bittencourt, L., and Madeira, E. (2019). Multi-tier edge-to-cloud architecture for adaptive video delivery. In *2019 7th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 23–30.

Jacobs, A., Pfitscher, R. J., Ferreira, R., and Granville, L. (2018). Refining network intents for self-driving networks. *Proceedings of the Afternoon Workshop on Self-Driving Networks*.

Jacobs, A. S., Pfitscher, R. J., Ribeiro, R. H., Ferreira, R. A., Granville, L. Z., and Rao, S. G. (2019). Deploying natural language intents with lumi. In *Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos*, SIGCOMM Posters and Demos '19, page 82–84, New York, NY, USA. Association for Computing Machinery.

Jivani, F. D., Malvankar, M., and Shankarmani, R. (2018). A voice controlled smart home solution with a centralized management framework implemented using ai and nlp. In *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*, pages 1–5.

Kadali, B., Prasad, N., Kudav, P., and Deshpande, M. (2020). Home automation using chatbot and voice assistant. *ITM Web of Conferences*, 32:01002.

Lago, A. S., Dias, J. P., and Ferreira, H. S. (2021). Managing non-trivial internet-of-things systems with conversational assistants: A prototype and a feasibility experiment. *Journal of Computational Science*, 51:101324.

Longo, C. F., Santoro, C., and Santoro, F. F. (2019). Meaning extraction in a domotic assistant agent interacting by means of natural language. In *2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pages 21–26.

Netfilter (2021). O projeto netfilter.org "iptables". Disponível em: https://www.netfilter.org/projects/iptables/index.html. Acesso em: 07 de out. 2021.

Noruwana, N. C., Owolawi, P. A., and Mapayi, T. (2020). Interactive iot-based speech-controlled home automation system. In *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, pages 1–8.

Rani, P. J., Bakthakumar, J., Kumaar, B. P., Kumaar, U. P., and Kumar, S. (2017). Voice controlled home automation system using natural language processing (nlp) and internet of things (iot). In *2017 Third International Conference on Science Technology Engineering Management (ICONSTEM)*, pages 368–373.

Riftadi, M. and Kuipers, F. (2019). P4i/o: Intent-based networking with p4. In *2019 IEEE Conference on Network Softwarization (NetSoft)*, pages 438–443.

Rodrigues, D. O., Santos, F. A., Filho, G. P. R., Akabane, A. T., Cabral, R., Immich, R., Jr., W. L., da Cunha, F. D., Guidoni, D. L., Silva, T. H., do Rosário, D., Cerqueira, E., Loureiro, A. A. F., and Villas, L. A. (2019). Computação urbana da teoria à prática: Fundamentos, aplicações e desafios. *CoRR*, abs/1912.05662.

Tian, B., Zhang, X., Zhai, E., Liu, H. H., Ye, Q., Wang, C., Wu, X., Ji, Z., Sang, Y., Zhang, M., Yu, D., Tian, C., Zheng, H., and Zhao, B. Y. (2019). Safely and automatically updating in-network acl configurations with intent language. In *Proceedings of the ACM Special Interest Group on Data Communication*, SIGCOMM '19, page 214–226, New York, NY, USA. Association for Computing Machinery.

Titara, D. (2021). Casa conectada é tendência mundial. *Revista Eletrolar News*, 22(143):54–70.