# Abordagem para Controle de Acesso em IoT baseado em Edge Computing com foco em Confiabilidade

**Rogerio Lopes. V. Cesar**[1], **Antonio J. da S. Almeida**[1], **Márcio E. F. Maia**[1]

[1]PCOMP Programa de Pós-Graduação em Computação – Universidade Federal do Ceará (UFC)
Quixadá – CE – Brasil
Av. José de Freitas Queiroz, 5003, Quixadá - CE, 63902-580

{rogerionqnq,marcioefmaia}@gmail.com

***Resumo.*** *A Internet das Coisas permite a integração entre diversos dispositivos e serviços computacionais presentes na Internet. Em aplicações críticas, como e-health ou edifícios inteligentes, existem requisitos específicos que devem ser considerados, como confiabilidade e privacidade no acesso às informações, dadas as restrições computacionais inerentes aos dispositivos de borda. Neste artigo, propomos uma abordagem que busca melhorar a confiabilidade em ambientes IoT, por meio do uso de técnicas de controle de acesso utilizando uma arquitetura descentralizada baseada em edge computing. Assim, o objetivo é regionalizar e otimizar a troca de dados usando uma linguagem de consulta baseada em GraphQL para reduzir a busca excessiva de dados. Apresentamos uma análise de desempenho na qual comparamos a latência e a taxa de erros em solicitações de operações em ambientes cloud e edge computing.*

## 1. Introdução

A Internet das Coisas como características fundamentais a conexão e comunicação entre diferentes dispositivos inteligentes, em que estes vão desde de um pequeno sensor até máquinas mais robustas, em termos de recursos computacionais, como um smartphone ou dispositivo vestível.

Estima-se que até 2025 o número de dispositivos IoT conectados no mundo cresça para 21,5 bilhões [IoT Analytics Research 2018], a impactar na economia global com a movimentação de mais de 11 trilhões de dólares [Sowjanya and Nagaraju 2016].

Na Internet das Coisas, parte das aplicações possui como requisito fundamental o processamento de dados em tempo real [Liu et al. 2020]. Podemos visualizar essa característica em [Thiam et al. 2021] que realiza um estudo sobre a confiabilidade em aplicação IoT em agricultura e [Kumar et al. 2021] para monitorar a qualidade da água em lençóis freáticos, por exemplo. Em comum, essas aplicações capturam dados de situações do mundo real, para adquirir inteligência a partir das circunstâncias do cenário físico observado. A aquisição, processamento e entrega de resultados devem ser efetuados de forma confiável e segura, diante as restrições de tempo e recursos computacionais, respondendo aos eventos com segurança.

O controle de acesso é definido como a metodologia que lida com os direitos de acesso a dados e serviços, por um usuário ou dispositivo autorizado. Seu objetivo é gerenciar privilégios, de forma a limitar, permitir ou negar obtenção de serviços ou informações, garantindo a proteção dos dados [Bate et al. 2017]. Considerando a natureza

crítica dos sistemas IoT, a confiabilidade de elementos de segurança se torna primordial em qualquer projeto de aplicação.

Em engenharia de software, a confiabilidade representa a competência do sistema em manter a continuidade do serviço de forma correta [Avizienis et al. 2004]. Esse conceito está relacionado à diversos atributos, como disponibilidade, manutenibilidade e integridade.

Mesmo considerando o grande fluxo de dados nas proximidades dos dispositivos *(edge)*, concentrar os dados em um elemento central *(cloud)*, pode ser desnecessário e gerar alta latência [Kawaguchi and Bandai 2020]. Nesse sentido, a utilização de uma arquitetura descentralizada por meio de técnicas que melhorem a disponibilidade podem expor uma melhor confiabilidade.

Neste artigo, propomos uma infraestrutura para controle de acesso em IoT cujo objetivo é a melhoria na confiabilidade dos serviços. A proposta baseia-se em computação em borda e tecnologias de encaminhamento de mensagens sobre a linguagem de consultas baseadas em grafos (GraphQL), além de repositório de políticas de acesso baseadas em contexto.

O artigo segue organizado da seguinte forma: seção 2, que apresenta uma revisão da literatura; seção 3, na qual trata de trabalhos relacionados; seção 4, em que se expõe a abordagem proposta; seção 5 apresenta a avaliação e análise de experimentos; por fim, tem-se as considerações finais.

## 2. Revisão da Literatura

### 2.1. Controle de Acesso

O controle de acesso é definido como a metodologia que lida com os direitos de acesso a dados e serviços, por um usuário ou dispositivo autorizado. Seu objetivo é gerenciar privilégios, de forma a limitar, permitir ou negar obtenção de serviços ou informações, garantindo a proteção dos dados [Bate et al. 2017].

Para o desenvolvimento de mecanismos de controle de acesso, [Hu et al. 2013] define 3 principais componentes que podem ser abstraídos, sendo eles: **Políticas**, que são as regras impostas pelo ambiente no qual o controle de acesso agir; **Modelo**, responsável por traduzir as especificações das políticas a serem implementadas para os usuários; e **Mecanismo**, que traduz as operações entre o usuário e suas requisições baseadas na estrutura proposta pelas políticas e modelo.

### 2.2. *Confiabilidade em IoT*

Para [Pokorni 2019], o termo "confiabilidade" pode ser definido como a probabilidade de um determinado componente ou sistema satisfazer requisitos e padrões de desempenho, quando produz saídas corretas durante um determinado período de tempo. Nesse ponto de vista, a confiabilidade busca alcançar a qualidade do sistema, buscando evitar, através de prevenção ou tolerância, que aconteçam falhas ou erros no sistema.

Quando se trata de comunicação em rede, como é o caso de IoT, de um ponto de vista do usuário, a confiabilidade está relacionada à mínima interrupção do serviço e do ponto de vista do provedor do serviço é suportar ou tolerar falhas ou ataques sistemáticos, sem impactar diretamente na experiência do usuário. Para tanto, há múltiplas

possibilidade de utilização do termo "confiabilidade", como no caso de robustez da aplicação, resistência a problemas de segurança, autoadaptação e usabilidade a longo prazo [Maita 2020].

Para [Prasad et al. 2013], a confiabilidade é fator critico para uma comunicação eficiente dentro do paradigma IoT, pois atividades de detecção, processamento e transmissão de dados realizados de forma não confiável pode causar danos à todo o sistema como produção de relatórios de monitoramento incorretos, longos atrasos e até mesmo perca de dados. Esses efeitos resultariam em um desinteresse nas pessoas, de forma geral, inclusive na utilização de tecnologias que adotem o ambiente IoT.

O ambiente complexo em IoT requer uma visão de confiabilidade aplicada em cada camada sob diferentes aspectos, dados os diferentes requisitos das aplicações. Assim sendo, a arquitetura a qual é implantada o sistema influencia diretamente nos objetivos para qual a aplicação foi concebida. Há diferentes propostas de arquiteturas que visam principalmente confiabilidade, qualidade do serviço (QoS) e integridade dos dados [Maita 2020].

Inicialmente, a arquitetura das aplicações em IoT usada por pesquisadores foi a baseada na industria de comunicação móvel [Maita 2020], utilizando 3 camadas (*perception, network e application*), como exemplo a proposta por [Wu et al. 2010]. Para [Ravidas et al. 2019], a separação entre as camadas de *middleware* e comunicação permite um bom entendimento sobre o compartilhamento de recursos entre nós e usuários finais. Em seu estudo, há a adoção de arquitetura com 4 (*physical, network, middleware and application*). Analisando as tecnologias em *gateway* a partir da visão de IoT, [Zhong et al. 2015] apresenta também o modelo de 5 camadas (*perception, network access, network, application support e presentation*).

Considerando uma arquitetura em 3 camadas, [Kempf et al. 2011], aponta que, na camada *link*, a confiabilidade requer o uso da pilha TCP para transmissão entre aplicativos na internet e deve-se considerar problemas de percas de pacotes por congestionamento. Na camada de transmissão e roteamento, a comunicação deve preferir o protocolo UDP devido à restrições computacionais e de energia, porem é necessário tratar retransmissões. Na camada de aplicação, a indicação do correto funcionamento e estado operacional do aplicativo é importante para confiabilidade do sistema. Um outro elemento importante é o controle de acesso, cujo objetivo é gerenciar o acesso garantindo a continuidade e corretude da aplicação.

No estudo apresentado por [Xing 2020], os desafios relativos ao estudo de confiabilidade em IoT, são analisados sob a perspectiva de arquitetura em 4 camadas (percepção, comunicação, suporte e aplicação). O autor relata que a computação em borda ganha popularidade por melhorar o tempo de resposta (latência) em comparação ao processamento centralizado na nuvem. O trabalho condensa estudo sobre confiabilidade em IoT, apontando tópicos relevantes, categorizados por camadas (percepção, comunicação, suporte e aplicação).

Um modelo de avaliação de confiabilidade em sistemas IoT pode ser encontrado em [Thomas and Rad 2017], que se baseia em métricas fundamentais de qualidade para sistemas de alto desempenho, como no caso de sistemas IoT. Nesse sentido, a figura 1 mostra a relação entre essas métricas e a arquitetura em 4 camadas em sistemas IoT. A

**Usabilidade** está diretamente relacionada com à construção de uma interface para uso conveniente e satisfação do usuário. A **Dependabilidade** refere-se ao sistema funcionar de acordo com suas especificações, a relacionar concepção, projeto e objetivos. Um dos elementos que entregaria essa característica é o de redução de latência, que pode comprometer a entrega do serviço. A **Disponibilidade** é o parâmetro quantitativo que associa a probabilidade do sistema estar disponível para o uso em um determinado período. A tomada de decisão entre o dispositivo IoT e o usuário final deve ser feito em tempo prontamente acessível. A **Manutenabilidade** se refere à capacidade do sistema ser facilmente desacoplando, atualizado e consertado, conservando as funcionalidades do sistema sem obstrução.

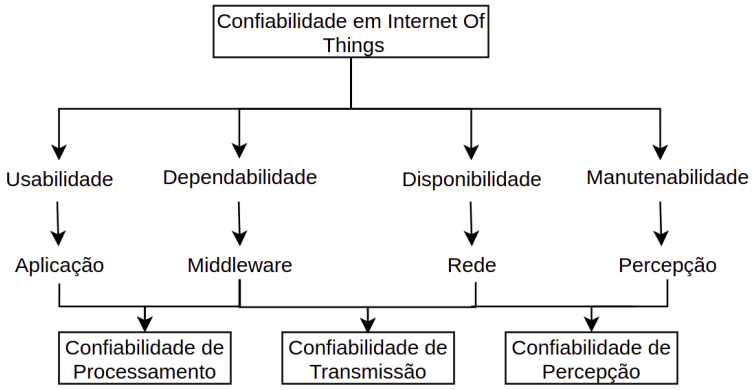


**Figura 1. Modelo de Avaliação de Confiabilidade em IoT - Fonte: [Thomas and Rad 2017] (Adaptado)**

## 2.3. Cloud e Edge Computing

O conceito de *Cloud Computing* está consolidado há alguns anos. Sua concepção é dada por tudo que esteja hospedado na Internet, sendo possível alocar recursos, serviços ou dados, e que estejam disponibilizados para o usuário, quando solicitado, podendo ainda compor serviços mais robustos [Biswas and Giaffreda 2014]. As principais características apresentadas pelos autores são: oferta de serviços sob demanda, acesso onipresente, *pool* de recursos e elasticidade.

Essas características podem convergir com o ambiente para aplicações em IoT, a possibilitar a arquitetura de computação em Nuvem mais viável em alguns casos. No entanto o crescimento da utilização de dispositivos em ambiente IoT proporciona, consequentemente, o risco de aumento da largura da banda para comunicação. Além disso, também expõe riscos de segurança e privacidade [Alwarafy et al. 2020].

Para o modelo de costume de computação, as aplicações seguem um padrão onde a computação é processada no servidor em nuvem e executada nos dispositivos do usuário. Mas este mecanismo possui algumas desvantagens, como grande acúmulo de tarefas a serem processadas e alto atraso na transmissão de dados. Para tentar solucionar esses problemas, pode-se fazer o uso da *Edge Computing*, cujo objetivo é executar algumas tarefas de processamento e armazenamento de dados na borda da rede, possibilitando a implementação de tarefas o mais próximo dos usuários finais [Xue et al. 2020].

Além de características como consciência de localização, e acesso em tempo real, [Xue et al. 2020] cita algumas características chaves encontradas na arquitetura de *Edge Computing*, tais como a alta distribuição de processamento, baixa latência, alta largura de banda e proximidade com usuário.

## 3. Trabalhos Relacionados

Os trabalhos compreendem propostas de *frameworks* de autorização para acesso em ambientes IoT, apresentando arquitetura e infraestrutura de comunicação. Desta forma, trabalhos que investigam prioritariamente mecanismos de autenticação e/ou auditoria (*accountability*) não fazem parte deste escopo.

O trabalho de [Hernández-Ramos et al. 2015] apresenta a proposta de um *framework* de segurança IoT compatível com a arquitetura de referência ARM (*Architectural Reference Mode*) e analisa sua viabilidade em cenários de edifícios inteligentes. A base para o controle de autorização é composta por aspectos contextuais, integrando-os como componente fundamental para a estrutura implementada. Além disso, módulos que funcionam como gerenciador de níveis de confiança e gerenciamento de chaves e identidade são considerados na estrutura.

A proposta de [Bandara et al. 2016] consiste em um *framework* para controle de acesso em prédios inteligentes, como suplemento para API em edifícios inteligentes. Nesse caso, o gerenciador de segurança é considerado um terceiro sistema, diante da interação entre o usuário e a rede de sensores. O objetivo é facilitar a gerência da segurança, realizar cálculos complexos de segurança (suporte a ambientes com restrição computacional). Para garantir que apenas usuários autorizados possam acessar o dispositivo, este é forçado a solicitar ao gerente de segurança para avaliar solicitações.

[Alkhresheh et al. 2018] propõe, diante das técnicas tradicionais de especificações de políticas de controle de acesso, automatizar a geração destas, de acordo com o contexto, objetivando mais flexibilidade e adaptabilidade, diante dos ambientes de IoT dinâmicos.

Tendo em vista os problemas de segurança de dados provenientes do uso de tecnologias IoT em saúde, mais especificamente a necessidade de proteção de "coisas" inteligentes contra acessos não autorizados, [Pal et al. 2017] propõe uma arquitetura para controle de acesso que melhora o gerenciamento de políticas de forma a reduzí-las e prover um controle de acesso refinado para o domínio de *Smart Healthcare*. Em seu modelo, há uma abordagem híbrida em que são utilizados como base atributos, funções e recursos. Os atributos são utilizados para associar uma entidade a uma determinada função e também para avaliar permissões. Com base em atributos adicionais, os recursos especificados por políticas podem receber solicitações de acesso e a partir desta o acesso pode ser concedido ou não.

[Wang et al. 2019] Propõe o *framework* de controle de acesso distribuído baseado em atributos chamado ADAC (*Atribute-Based Distributed Access Control*), que associa a tecnologia *blockchain* e atributos de dispositivos IoT.

Podemos visualizar uma comparação entre os estudos mencionados nesse trabalho por meio da Tabela 3. Dentre as características, o presente trabalho obtém destaque em realizar estudo sobre a confiabilidade em IoT e a realização de análise de desempenho.

| Trabalho | Deploy | Modelo de Políticas | Domínio de Aplicação | Confiabilidade | Análise de Desempenho Estruturada |
|---|---|---|---|---|---|
| Hernández-Ramos et al. 2015 | Nuvem | Atributos | Prédios Inteligentes | Não | Não |
| Bandara et al. 2016 | Nuvem | Atributos | Prédios Inteligentes | Não | Não |
| Alkhresh et al. 2018 | Não especifica | Contexto | Prédios Inteligentes | Não | Não |
| Pal et al. 2017 | Borda | *Capability* | Saúde | Sim | Não |
| Wang et al. 2019 | Borda | Atributos | Não Específica | Não | Não |
| **Este Trabalho** | **Borda /Nuvem** | **Contexto** | **Prédios Inteligentes** | **Sim** | **sim** |

Além disso, o *deploy* tanto na nuvem quanto na borda da rede diferencia-se dos outros trabalhos. Em relação ao modelo de políticas, consideramos mais adequado a utilização de contexto como base para a produção de políticas, aproveitando o conceito de *edge computing*.

## 4. Abordagem Proposta

O modelo de controle de acesso para uma aplicação em IoT, deve atender aos requisitos de acesso refinado aos recursos, adaptação ao ambiente e confiabilidade de entrega do serviço solicitado. Esses requisitos guiaram o desenvolvimento da proposta. Assim, introduziremos alguns conceitos para a abordagem, seguidos de estudo de caso, modelo de políticas, arquitetura e implementação.

### 4.1. Definições

Como estrutura tradicional, um mecanismo de controle de acesso certifica que um **sujeito**, por meio de permissões (políticas) para cada tipo de **operação** a ser realizada em um **objeto**. Neste trabalho, podemos definir: **sujeito** tido como as aplicações de propósito geral ou específico para processamento de dados no ambiente; **operação** como sendo as ações de baseadas em aplicações em dados não estruturados, como criação e leitura de arquivos ou serviços baseados em web (*query/mutation*), além de mecanismo de *publish/subscribe*; **objetos** que são dados ou séries históricas de dados produzidos por elementos no ambiente.

### 4.2. Modelo de Políticas

Para o modelo de controle de acesso que ofereça flexibilidade, ciência de contexto e dinamicidade diante do cenário complexo de IoT, foram definidas para este trabalho políticas baseadas em contexto. Assim, os dados coletados, armazenados ou processados no ambiente devem ser disponibilizados para usuários somente por usuários autenticados e autorizados quando necessários dependendo da dinâmica do contexto apresentado.

Os atributos de contexto são utilizados para compor as expressões que definem os modelos das políticas. As expressões **(Ex)** associam e comparam os atributos **(A)**, dos elementos envonvidos (**S - Sujeito e O - Objeto )** e seus valores, na requisição de acesso, por intermédio de operadores relacionais **($op_r$)** ( “ $>$ ”, “ $<$ ”, “ $>$ ”, “ $\leq$ ”, “ $\geq$ ”, “ $=$ ”, “ $\neq$ ”). As expressões seguem a estrutura:

$$EX_{\mathrm{i}} = A(S, O, P)_{\mathrm{i}} < \boldsymbol{op_r} > \text{“}valor\text{”} \tag{1}$$

De acordo com as necessidades de restrições de acesso, a composição de expressões contextuais definem as políticas **P** de controle de acesso, por meio de operadores lógicos **$op_l$** (“ $\wedge$ ”, “ $\vee$ ”, “$\neg$”), resultando em permissão ou proibição.

$$P = Ex_{\mathrm{i}} < \boldsymbol{op_l} > Ex_{(\mathrm{i}+1)} < \boldsymbol{op_l} > Ex_{(\mathrm{i}+2)} < \boldsymbol{op_l} > \ \ldots Ex_{(\mathrm{n}-1)} < \boldsymbol{op_l} > Ex_{\mathrm{n}} \tag{2}$$

### 4.3. Arquitetura

A infraestrutura da proposta obedece algumas questões que serão discutidas nas próximas seções. Assim, a referência para o desenvolvimento de aplicações pode ser explanada.

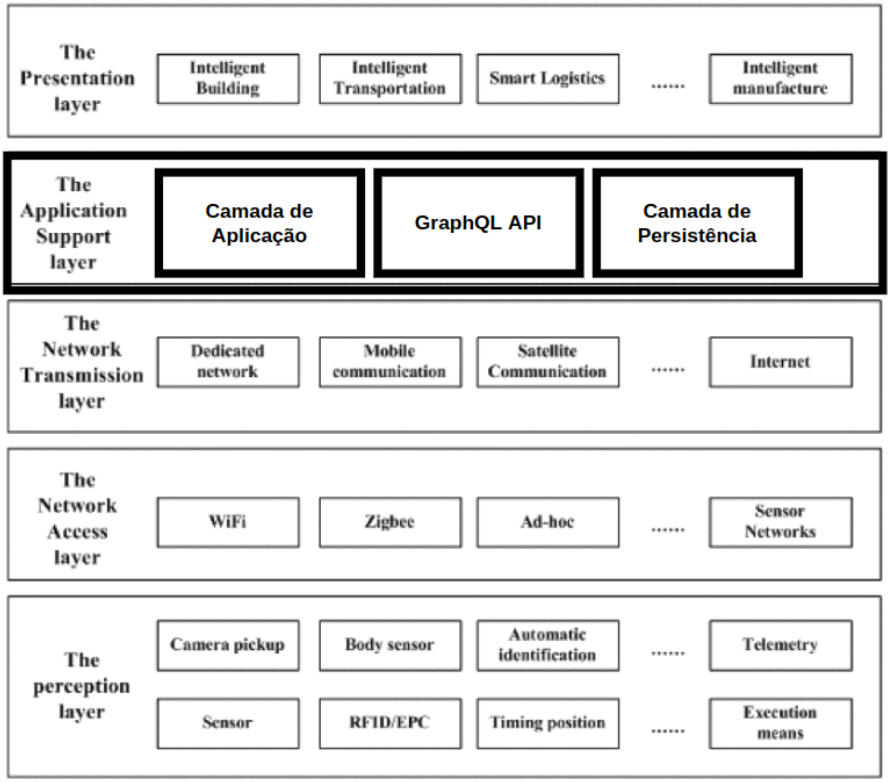


**Figura 2. Localização dos componentes e serviços em alto nível propostos nesse trabalho - Fonte: Próprio autor**

Os elementos apresentados na figura 2 têm intuito de apresentar uma referência para modelos de aplicações a serem desenvolvidas utilizando a abordagem mostrada neste trabalho. Na figura, referenciamos todos os componentes (Camadas de Aplicação e Persistência e GraphQL API) na camada de aplicação, evidenciando a possibilidade ou necessidade de produção de outros componentes que possam compor a pilha.

O sistema de controle de acesso dessa proposta é baseado em contexto e utiliza a tecnologia *edge computing*. O gerenciamento das requisições é executado por meio da linguagem de consulta GraphQL. A figura 3 mostra os 3 elementos principais que consistem o modelo: *cliente*, *Server Edge* e End Devices.

O **cliente** é a entidade que possui e manipula o dispositivo/aplicação cliente e que envia a requisição de acesso a um determinado recurso dentro do ambiente IoT. De acordo com a linguagem de consulta GraphQL, a requisição pode ser de dois tipos: *query* que realiza uma consulta de dados ou *mutation* que executa ações de alteração nos dados (por exemplo inserção, atualização ou exclusão). Na arquitetura apresentada, esse tipo de tecnologia pode também pode efetuar conexões do tipo *websocket* por meio de *subscriptions e publishs*, às quais permitem a comunicação bidirecional, notificações em alterações e percepção em tempo real, com objetivo de gerenciar a disponibilidade dos recursos.

O ***Server Edge*** é o dispositivo responsável por receber as requisições e prepará-las para acesso aos recursos. Essa entidade é composta por 2 camadas: A **camada de aplicação**, que gerencia as requisições em um *endpoint - GraphQL API*, sendo responsável por preparar e aplicar as políticas de acesso, e a **Camada de Persistência**, que armazena as políticas de acesso. A utilização do GraphQL determina um único ponto para o acesso aos recursos, fornecendo todas as informações necessárias para as aplicações cliente. Essa configuração sustenta também a redução de ocorrências de *overfetching* (quando há uma consulta de dados além dos necessários), e *underfetching* (quando a requisição retorna menos dados que o essencial), características essas presentes em arquiteturas do tipo REST.

Os **End-Devices** representam os dispositivos IoT podendo ser sensores, ou atuadores ou recursos possíveis de acesso para acesso ao cliente.

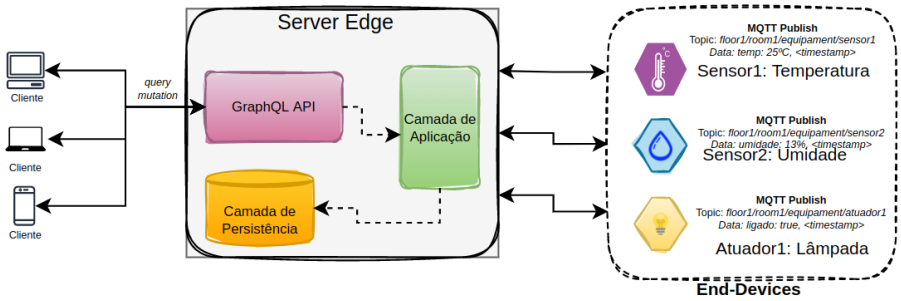


**Figura 3. Componentes da outra mais arquitetura proposta. Fonte: (Próprio Autor)**

## 4.4. Implementação

A implementação é baseada na arquitetura proposta na figura 3. A construção do *server edge* é feita em linguagem JavaScript no ambiente de execução *nodejs*[1]. A estrutura para a utilização do *schema* GraphQL é apoiada na biblioteca *graphql-yoga*[2].

Para a camada de persistência, foi usado o MongoDB[3]. Este, por sua vez, é um

[1]https://nodejs.org/
[2]https://github.com/dotansimha/graphql-yoga
[3]https://www.mongodb.com/

banco de dados multiplataforma orientado a documentos que oferece alto desempenho e fácil escalabilidade.

Para o controle de acesso, os dados do contexto são instanciados segundo um modelo contendo, para realização do estudo, 10 atributos de contexto ( *temperature, hour, day, point, role, localization, age, connectivity, battery* e *device*), além da indicação do ponto que a política referencia (*point*), que são comparados com as políticas cadastradas no banco de dados. Em nosso estudo, consideramos que o usuário/objeto já esteja, previamente, autenticado por mecanismo adequado e são analisadas as políticas para autorização da operação. O algoritmo 1 mostra o fluxo da interação. Ao receber a requisição, é instanciado o contexto e logo após, há a consulta de políticas no repositório. Considerando os 9 elemntos de contexto citados anteriormente, o algoritmo compara o item da política, com o item do contexto. Caso obtenha paridade ou temha um item não cadastrado (*null*), a variável *count*, é incrementada. Esta última é verificada em seu valor, e caso obtenha-se 9, o ponto (*point*) é adicionado à lista de pontos permitidos (*allPointsAllowed)*.

**Algoritmo 1:** Análise de autorização

```
Entrada: operação, ponto a ser acessado
Recebimento de requisição ;
Instanciação do contexto ;
allPointsAllowed = [] ;
//lista de pontos permitidos para a requisição. Consulta políticas no BD ;
count = 0;
repeat
    if politica.item[n] == contexto.item[n] or politica.item[n] == null then
        count++;
        if (count == 9) then
            //extrai a referência do ponto da política
            //insere na estrutura de pontos permitidos
            allPointsAllowed.push(point) ;
until todas as políticas;
```

## 5. Avaliação

Para a avaliação, realizamos uma análise de desempenho seguindo os pontos definidos por [Bukh 1992]. Dessa forma, buscamos moderar viés que por ventura a estrutura do estudo possa conter. Na figura 4, especificamos os pontos da abordagem:

Para tal, em ambiente de *cloud*, foi utilizada a plataforma heroku [4]. Em ambiente de *edge*, um Raspberry Pi 4 Model B, dispositivo que conta com um processador Broadcom 2711 de 64 bits, arquitetura Cortex A72, 1.5 GHz e 4 GB LPDDR4 SDRAM [5].

### 5.1. Experimento

O objetivo principal do estudo é efetuar uma comparação para um modelo de controle de acesso baseado em contexto com foco em confiabilidade em ambiente IoT, quando implementado em *cloud* e em *edge computing*. Nesse sentido, busca-se realizar uma verificação

[4]https://www.heroku.com/

[5]https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

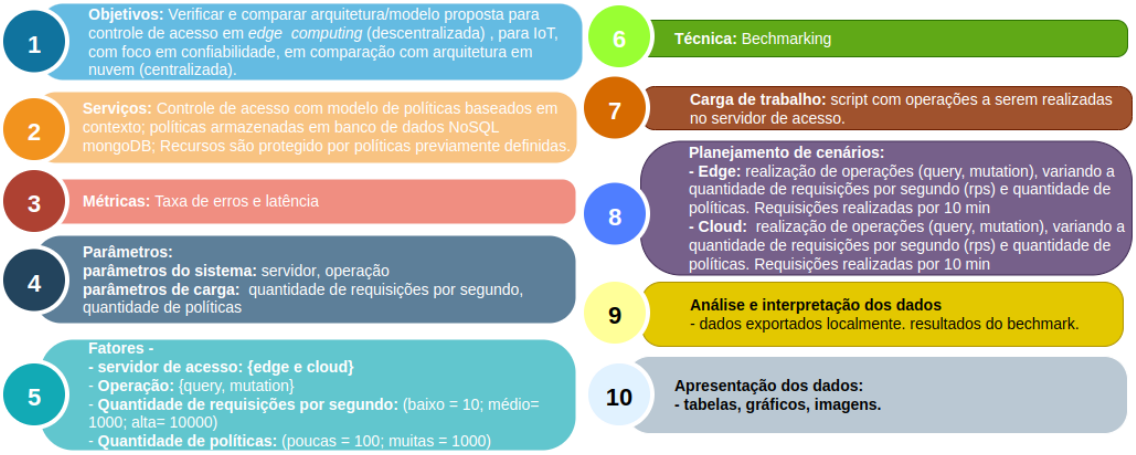


**Figura 4. Análise de desempenho sistematizada [Bukh 1992] - Fonte: (Próprio Autor)**

de parâmetros que são importantes para estudo de confiabilidade em aplicações desse tipo. Assim, como visto na figura 4, fatores como requisições por segundo e quantidade de políticas armazenadas, foram fatores que variaram durante o estudo, considerando um intervalo de tempo de 10 minutos.

A utilização de um *benchmark* foi necessária, sendo escolhido o *GraphQLBench*[6]. Essa ferramenta possui a característica de ser versátil e ideal para carregamento de testes em serviços GraphQL.

A execução dos experimentos foi introduzida por um cenário onde, em cada ambiente (*cloud* e *edge*) , as rps (requisições por segundo) iniciam em 10, com uma quantidade de políticas baixa (100). A cada execução do benchmark, a quantidade de rps são alteradas (de 10 para 1.000 e depois para 100.000), sendo realizada a execução em cenário também para muitas políticas (1000) cadastradas. A execução das requisições foram feitas para operação de *query* (consulta) e *mutation* (atualização, criação e exclusão).

## 5.2. Resultados

Com o objetivo de avaliar a latência das requisições de acesso em ambiente de borda e nuvem, a figura 5 mostra a média de latência para os diferentes cenários em questão. Houve variação em quantidade de políticas e requisições por segundo. O *benchmark* envia as requisições e realiza o cálculo da latência para a resposta à requisição, diante do cenário. Verificou-se que, na maioria dos casos, a média da latência em *cloud* foi maior que em *edge*. Foi observado também que a quantidade de políticas cadastradas pouco influenciou no contexto da avaliação. Foi verificado também que nos cenários onde a quantidade de políticas é 10000 e as rps é 10 e 1000, a latência em edge foi, em média maior. Em tese, atribui-se o fenômeno à uma baixa quantidade de experimentos realizados, necessitando assim de uma maior e melhor análise estatística.

O cenário foi visto, quando a operação foi *mutation*. Em média, a latência em relação à requisições da operação, para ambas as situações de políticas (poucas e muitas), foi menor quando o serviço foi oferecido na borda da rede (figura 6).

[6]https://github.com/hasura/graphql-bench

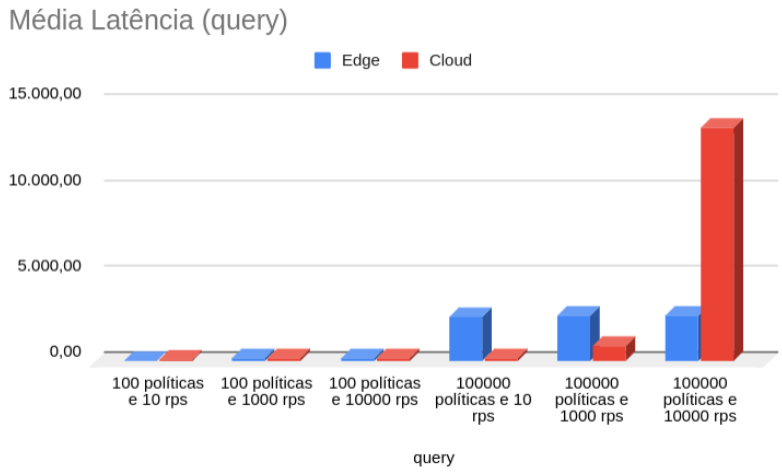


**Figura 5. Média latência em Query - Fonte: (Próprio Autor)**

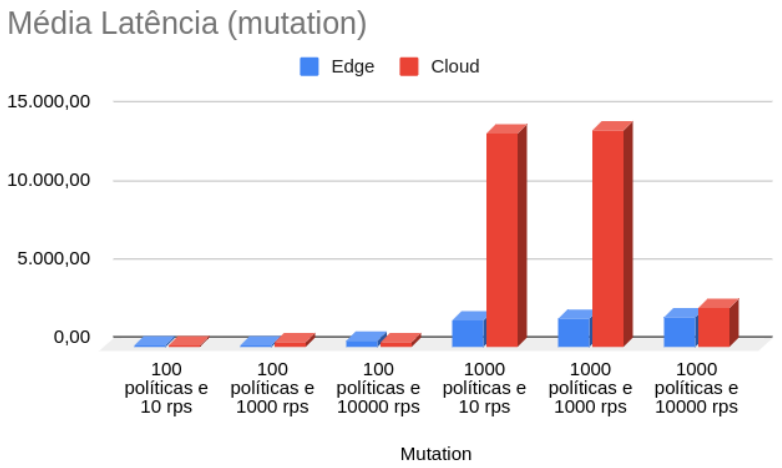


**Figura 6. Média latência em Mutation - Fonte: (Próprio Autor)**

Foi analisado o pior valor registrado de latência e feita a comparação em cada cenário e apresentada a taxa de variação borda/nuvem. A tabela 1, Mostra esse dados, podendo-se observar que, na maioria dos cenários, o pior valor de latência acontece na nuvem, tendo inclusive o maior valor registrado (30.207,87 ms), para o cenário de muitas políticas e alta taxa de requisições por segundo. Percebeu-se também uma grande discrepância entre os valores de latência para o mesmo cenário, mas para operações diferentes, o que pode indicar a necessidade de outro design de experimento para confirmar hipóteses aqui em estudo.

A tabela 2 mostra a taxa de erros medidas durante os experimentos. Foi verificado que, no ambiente em nuvem, houve maior taxa de erros, principalmente em nos cenários com maior quantidade de políticas. Não foram registrados erros para o ambiente em borda.

**Tabela 1. Pior latência, em milissegundos, registrada e variação de *Edge* em relação a *Cloud***

| Cenários | Query | | | Mutation | | |
|---|---|---|---|---|---|---|
| | *edge* | *cloud* | *edge/cloud* | *edge* | *cloud* | *edge/cloud* |
| 100 pol./10 rps | 2.619,56 | 366,81 | 714,15% | 488,30 | 441,20 | 110,68% |
| 100 pol./1000 rps | 817,48 | 756,09 | 108,12% | 327,16 | 996,69 | 32,82% |
| 100 pol./10000 rps | 824,97 | 7219,59 | 11,43% | 1.115,30 | 874,50 | 127,54% |
| 1000 pol./10 rps | 4294,14 | 673,1 | 637,96% | 2.717,37 | 30.201,94 | 9,00% |
| 1000 pol./1000 rps | 4119,17 | 24985,14 | 16,48% | 2.985,19 | 30.206,84 | 9,88% |
| 1000 pol./10000 rps | 4470,1 | 30151,15 | 14,83% | 2.969,37 | 30.207,87 | 9,83% |

**Tabela 2. Taxa de Erros em requisições para os cenários em estudo.**

| Cenários | Query | | Mutation | |
|---|---|---|---|---|
| | *edge* | *cloud* | *edge* | *cloud* |
| 100 políticas e 10 rps | 0,00% | 0,00% | 0,00% | 0,00% |
| 100 políticas e 1000 rps | 0,00% | 0,00% | 0,00% | 0,00% |
| 100 políticas e 10000 rps | 0,00% | 0,00% | 0,00% | 0,00% |
| 1000 políticas e 10 rps | 0,00% | 0,57% | 0,00% | 0,00% |
| 1000 políticas e 1000 rps | 0,00% | 0,46% | 0,00% | 0,00% |
| 1000 políticas e 10000 rps | 0,00% | 0,34% | 0,00% | 0,23% |

## 6. Considerações Finais

Neste artigo, propomos um estudo de análise de desempenho para um controle de acesso em IoT com políticas baseadas em contexto e foco em confiabilidade. Além disso, tecnologias que propiciassem mais confiabilidade, no âmbito de disponibilidade, como mecanismo de consulta de dados GraphQL, banco de dados NoSQL e baixa complexidade de modelo de políticas, foram considerados. O estudo contou com uma comparação entre arquiteturas centralizada (*cloud computing*) e descentralizada (*edge computing*). A partir do desenho e execução dos experimentos, verificou-se mais confiabilidade na camada de aplicação, por meio de uma menor taxa de erros e menores, em média, valores de latência no ambiente em borda da rede. Como trabalho futuro, propomos a implementação de um *framework* para controle de acesso em *IoT* e um desenho de experimentos que possibilite uma maior quantidade de dados estatísticos, além de analisar a confiabilidade ponta-a-ponta.

## Referências


Alkhresheh, A., Elgazzar, K., and Hassanein, H. S. (2018). Context-aware automatic access policy specification for iot environments. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pages 793–799. IEEE.

Alwarafy, A., Al-Thelaya, K. A., Abdallah, M., Schneider, J., and Hamdi, M. (2020). A survey on security and privacy issues in edge computing-assisted internet of things. *IEEE Internet of Things Journal*.

Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE transactions on dependable and secure computing*, 1(1):11–33.

Bandara, S., Yashiro, T., Koshizuka, N., and Sakamura, K. (2016). Access control framework for api-enabled devices in smart buildings. In *2016 22nd Asia-Pacific Conference on Communications (APCC)*, pages 210–217. IEEE.

Bate, K. O., Kumar, N., and Khatri, S. K. (2017). Framework for authentication and access control in iot. In *2017 2nd International Conference on Telecommunication and Networks (TEL-NET)*, pages 1–6. IEEE.

Biswas, A. R. and Giaffreda, R. (2014). Iot and cloud convergence: Opportunities and challenges. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, pages 375–376. IEEE.

Bukh, P. N. D. (1992). The art of computer systems performance analysis, techniques for experimental design, measurement, simulation and modeling.

Hernández-Ramos, J. L., Moreno, M. V., Bernabé, J. B., Carrillo, D. G., and Skarmeta, A. F. (2015). Safir: Secure access framework for iot-enabled services on smart buildings. *Journal of Computer and System Sciences*, 81(8):1452–1463.

Hu, V. C., Ferraiolo, D., Kuhn, R., Friedman, A. R., Lang, A. J., Cogdell, M. M., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K., et al. (2013). Guide to attribute based access control (abac) definition and considerations (draft). *NIST special publication*, 800(162).

IoT Analytics Research (2018). tate of the iot 2018: Number of iot devices now at 7b -market accelerating. Disponível em: `https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b`. Acesso em: 04 Março 2021.

Kawaguchi, R. and Bandai, M. (2020). Edge based mqtt broker architecture for geographical iot applications. In *2020 International Conference on Information Networking (ICOIN)*, pages 232–235. IEEE.

Kempf, J., Arkko, J., Beheshti, N., and Yedavalli, K. (2011). Thoughts on reliability in the internet of things. In *Interconnecting smart objects with the Internet workshop*, volume 1, pages 1–4.

Kumar, G. R., Kishore, D., Kumar, G. V. M. G., Avila, J., Thenmozhi, K., Amirtharaja, R., and Praveenkumar, P. (2021). Waste contamination in water—a real-time water quality monitoring system using iot. In *2021 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–4. IEEE.

Liu, Y., Peng, M., Shou, G., Chen, Y., and Chen, S. (2020). Toward edge intelligence: Multiaccess edge computing for 5g and internet of things. *IEEE Internet of Things Journal*, 7(8):6722–6747.

Maita, S. L. S. (2020). *New Models of Reliability in the New Generation of Internet of Things*. PhD thesis, 00500:: Universidade de Coimbra.

Pal, S., Hitchens, M., Varadharajan, V., and Rabehaja, T. (2017). On design of a fine-grained access control architecture for securing iot-enabled smart healthcare systems. In *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 432–441.

Pokorni, S. J. (2019). Reliability and availability of the internet of things. *Vojnotehnički glasnik*, 67(3):588–600.

Prasad, S. S., Kumar, C., et al. (2013). A green and reliable internet of things. *Communications and Network*, 5(1):44–48.

Ravidas, S., Lekidis, A., Paci, F., and Zannone, N. (2019). Access control in internet-of-things: A survey. *Journal of Network and Computer Applications*, 144:79–101.

Sowjanya, G. and Nagaraju, S. (2016). Design and implementation of door access control and security system based on iot. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 2, pages 1–4. IEEE.

Thiam, F., Mbaye, M., and Wyglinski, A. M. (2021). Generic reliability analysis model of iot: Agriculture use case. In *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, pages 1–5. IEEE.

Thomas, M. O. and Rad, B. B. (2017). Reliability evaluation metrics for internet of things, car tracking system: a review. *Int. J. Inf. Technol. Comput. Sci.(IJITCS)*, 9(2):1–10.

Wang, P., Yue, Y., Sun, W., and Liu, J. (2019). An attribute-based distributed access control for blockchain-enabled iot. In *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–6. IEEE.

Wu, M., Lu, T.-J., Ling, F.-Y., Sun, J., and Du, H.-Y. (2010). Research on the architecture of internet of things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)*, volume 5, pages V5–484. IEEE.

Xing, L. (2020). Reliability in internet of things: Current status and future perspectives. *IEEE Internet of Things Journal*, 7(8):6704–6721.

Xue, H., Huang, B., Qin, M., Zhou, H., and Yang, H. (2020). Edge computing for internet of things: A survey. In *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, pages 755–760. IEEE.

Zhong, C.-L., Zhu, Z., and Huang, R.-G. (2015). Study on the iot architecture and gateway technology. In *2015 14th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*, pages 196–199. IEEE.