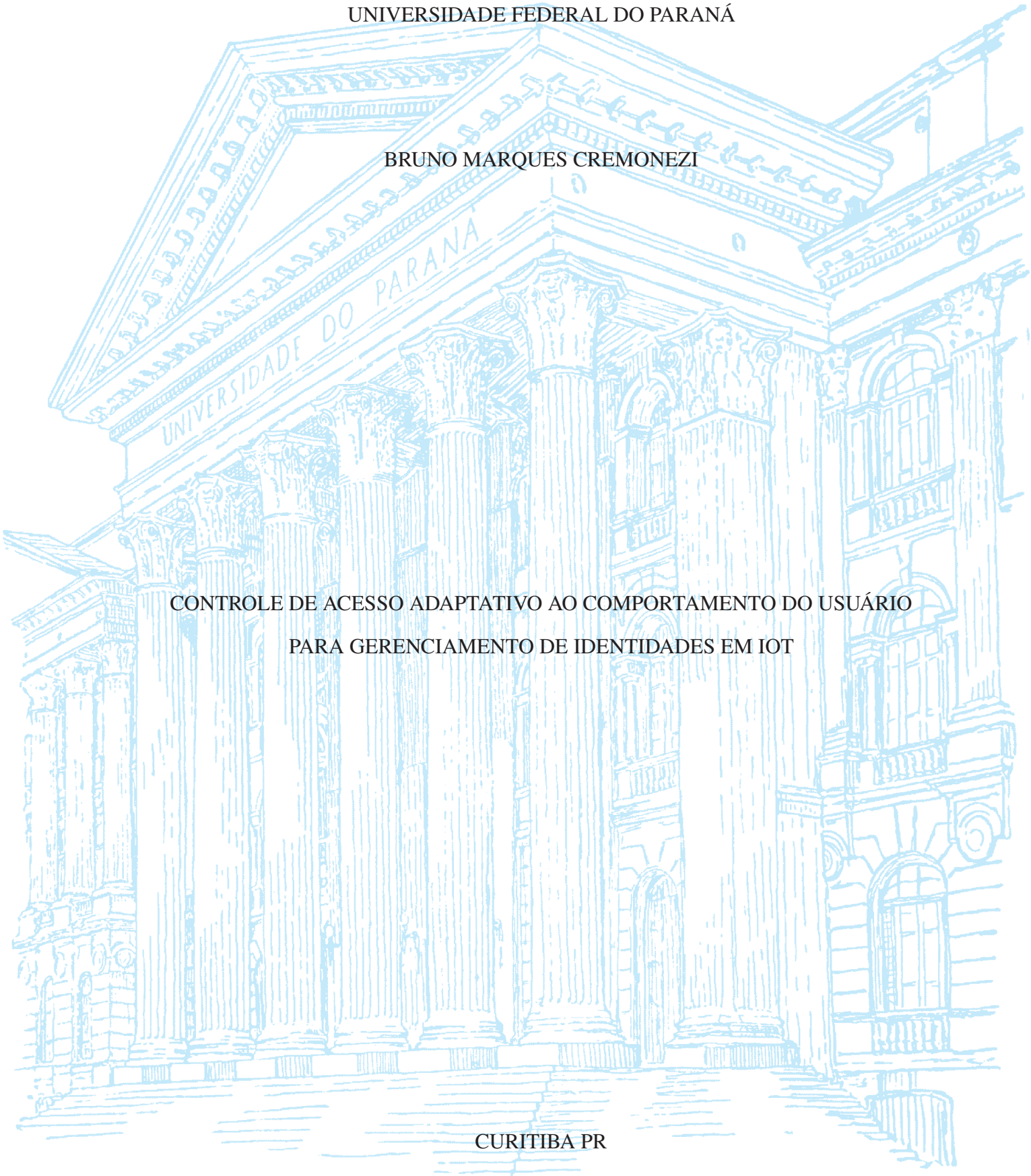
UNIVERSIDADE FEDERAL DO PARANÁ

BRUNO MARQUES CREMONEZI



CONTROLE DE ACESSO ADAPTATIVO AO COMPORTAMENTO DO USUÁRIO PARA GERENCIAMENTO DE IDENTIDADES EM IOT

CURITIBA PR

2023

BRUNO MARQUES CREMONEZI

CONTROLE DE ACESSO ADAPTATIVO AO COMPORTAMENTO DO USUÁRIO PARA GERENCIAMENTO DE IDENTIDADES EM IOT

Tese apresentada como requisito parcial à obtenção do grau de Doutor em Ciência da Computação no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Michele Nogueira Lima.

Coorientador: Alex Borges Vieira.

CURITIBA PR

2023

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA CIÊNCIA E TECNOLOGIA

Cremonezi, Bruno Marques
Controle de acesso adaptativo ao comportamento do usuário para gerenciamento de identidades em IoT. / Bruno Marques Cremonezi. – Curitiba, 2023.
1 recurso on-line : PDF.

Tese – (Doutorado) - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática.

Orientadora: Michele Nogueira Lima.
Coorientador: Alex Borges Vieira.

1. Internet das Coisas. 2. Computadores – Controle de acesso. 3. Identidade – Gestão. I. Universidade Federal do Paraná. II. Programa de Pós-Graduação em Informática. III. Lima, Michele Nogueira. IV. Vieira, Alex Borges. V. Título.

Bibliotecária: Roseny Rivelini Morciani CRB-9/1585

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da tese de Doutorado de **BRUNO MARQUES CREMONEZI** intitulada: **Controle de Acesso Adaptativo ao Comportamento do Usuário para Gerenciamento de Identidades em IoT**, sob orientação do Prof. Dr. MICHELE NOGUEIRA LIMA, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de doutor está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 15 de Setembro de 2023.

Assinatura Eletrônica
18/10/2023 12:32:17.0
MICHELE NOGUEIRA LIMA
Presidente da Banca Examinadora

Assinatura Eletrônica
18/10/2023 08:26:44.0
LEOBINO NASCIMENTO SAMPAIO
Avaliador Externo (UNIVERSIDADE FEDERAL DA BAHIA)

Assinatura Eletrônica
07/10/2023 02:04:54.0
WAGNER MACHADO NUNAN ZOLA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Assinatura Eletrônica
17/10/2023 19:16:05.0
ALDRI LUIZ DOS SANTOS
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Assinatura Eletrônica
06/10/2023 21:54:55.0
IGOR MONTEIRO MORAES
Avaliador Externo (UNIVERSIDADE FEDERAL FLUMINENSE)

Assinatura Eletrônica
17/10/2023 19:24:02.0
ALEX BORGES VIEIRA
Coorientador(a)

"Many fall in the face of chaos but not this one... not today"

AGRADECIMENTOS

Expresso minha profunda gratidão a todos que tornaram esta tese possível. Em primeiro lugar, à minha orientadora, Prof.ª Michele Nogueira, agradeço por me acolher em seu laboratório e compartilhar seu conhecimento. Ao meu grande amigo e coorientador, Prof. Alex Borges Vieira, minha imensa gratidão por todo apoio, paciência e disponibilidade, seja para esclarecer dúvidas técnicas ou oferecer conselhos. Sem você, esta tese não seria possível. Aos Profs. Edelberto Franco Silva e José Augusto Nacif, meu respeito e enorme gratidão. Além de professores, foram amigos e companheiros, sempre prontos a ensinar e a me apoiar nos momentos mais difíceis. Nunca conseguirei retribuir todo o suporte que recebi de vocês. Agradeço à Universidade Federal do Paraná pela oportunidade de realizar esta formação e à CAPES pelo suporte essencial ao desenvolvimento desta pesquisa. Aos amigos e colegas de laboratório, Agnaldo Batista, Gustavo Oliveira, Paulo Lenz, Benevid Silva, Carlos Pedroso, Igor Steuck, Yan Uehara, Cainã Passos, Anderson Neira, Andressa Vergut, Nelson Prates, Euclides, Bruno Schwengber, Uelinton Brezolin e Mateus Peloso, meu sincero agradecimento pela parceria ao longo desta jornada. Agradeço a toda a minha família, pela confiança e motivação, em especial, Flávia, Célio e Marise. Por fim, agradeço a todos que, com boa intenção, colaboraram direta ou indiretamente para a realização e finalização deste trabalho.

RESUMO

Os sistemas de Gestão de Identidades e Acessos (IAM) são fundamentais para garantir a autenticação e o controle de acesso na segurança de dados dos usuários e das aplicações na Internet das Coisas (IoT). A literatura sugere que um controle de acesso eficiente na IoT deve considerar os atributos das entidades envolvidas e criar políticas que definam os acessos permitidos e proibidos. O uso de atributos permite políticas mais precisas e granulares, reduzindo brechas de segurança e vazamentos de dados. Embora seja recomendável, o uso de atributos apresenta desafios, como a complexidade na gestão das políticas e o desempenho na obtenção dos atributos necessários para avaliá-las. Para um gerenciamento adequado, é essencial compreender o ambiente e prever o máximo de situações possíveis de acesso, traduzindo-as em políticas adequadas. Entretanto, a natureza dinâmica da IoT e a grande quantidade de objetos conectados tornam essa tarefa complexa, custosa e sujeita a erros. Modelos automatizados existem, mas são altamente complexos, gerando políticas difíceis de entender e modificar, transformando o controle de acesso em uma "caixa preta"não auditável, o que compromete sua aplicabilidade no dia a dia. A recuperação de atributos também exige atenção, pois as políticas dependem de atributos distribuídos em repositórios geograficamente distantes, o que pode gerar atrasos na avaliação e impactar o desempenho do controle de acesso. A hipótese deste trabalho é que o comportamento do usuário nos sistemas IAM pode ser utilizado para tornar o controle de acesso mais adaptado às necessidades da IoT, melhorando a escalabilidade, o desempenho e o gerenciamento. Dessa forma, são propostas uma arquitetura e métodos que utilizam esse comportamento para enfrentar os desafios da gestão de políticas e da recuperação de atributos. Primeiramente, apresenta-se um método para extrair políticas de acesso a partir dos registros do comportamento do usuário em uma aplicação. Esse método utiliza algoritmos de aprendizado supervisionado e genéticos para criar políticas legíveis e auditáveis, permitindo o gerenciamento humano. Os resultados indicaram que o método melhora o gerenciamento e a precisão das políticas em até 10% em comparação a técnicas do estado da arte. Além disso, propõe-se uma arquitetura para otimizar a recuperação de atributos, aplicando conceitos de cache e perfil de mobilidade. Essa abordagem replica proativamente os atributos mais próximos ao usuário, reduzindo em até 80% o número de saltos para obtê-los e diminuindo o custo de segurança comparado a técnicas tradicionais de cache. A solução equilibra a criação e a segurança na manutenção de novas réplicas na rede. Os resultados demonstram que o comportamento do usuário pode ser utilizado para aprimorar tanto o gerenciamento de políticas de acesso quanto a recuperação de atributos, validando a tese proposta.

Palavras-chave: Gestão de Identidades. Internet das Coisas. Controle de Acesso.

ABSTRACT

Identity and Access Management (IAM) systems are fundamental in ensuring appropriate authentication and access control for user data security and privacy in Internet of Things (IoT) applications. The literature suggests that efficient access control in IoT requires considering the attributes of the entities' and creating multiple policies to guide allowed and prohibited access in a specific application. By using these attributes, more precise and granular policies can be developed, which help prevent security breaches and data leaks. However, despite the recommendation to use attributes in developing access policies for IoT, it is crucial to understand the associated challenges, especially concerning the complexity of managing these attribute-based policies and the performance in obtaining the necessary attributes for their evaluation. Proper policy management presents a challenge that requires understanding the assessed environment and anticipating the most comprehensive range of possible access situations to translate them into correct policies. Due to the dynamic nature of IoT, along with a large number of connected objects and attributes associated with these objects, this task becomes complex, costly, and error-prone. It is important to note that there are models discussing policy management automation in the literature. However, these models are highly complex and generate difficult-to-understand policies, making the access control mechanism a non-auditable black box and impossible to be modified by humans in basic day-to-day management operations, such as updating and removing access policies. Regarding attribute recovery, policies are applied based on the attributes related to the entities involved in access, and problems such as obtaining or recovering attributes require attention. At this point, the large scale of IoT objects and users means that attributes are usually located in geographically distant repositories. Therefore, policy evaluation, which depends on these attributes, may suffer from delays impacting the access control mechanism's performance. This work hypothesizes that it is possible to use user behavior in IAM systems to make their access control mechanism more adapted to IoT's specific characteristics. Architectures and methods using user behavior are proposed to address the challenges faced by IoT access mechanisms: access policy management and attribute recovery performance. First, a method is presented to extract access policies through user behavior records in an application. This method uses supervised learning and genetic algorithms to create readable and auditable policies that humans can manage. The results showed that the method could assist in policy management and improve accuracy by up to 10% compared to similar state-of-the-art methods. This work also evaluated the challenge of attribute recovery in the IoT context. An architecture was proposed that uses cache concepts and mobility profiles, proactively replicating attributes used in access policies closer to the user. The approach considered the balance between creating and maintaining the security of new attribute replicas on the network. The results showed that it is possible to reduce the hops required to obtain attributes by up to 80%. Furthermore, the proposal reduces the security cost compared to traditional caching techniques. Therefore, the results demonstrate that user behavior can be used to improve access policy management and attribute recovery performance, validating the defined thesis.

Keywords: Identity Management. Internet of Things. Access Control.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÔNIMOS

IoT	Internet of Things (Internet das Coisas)
IAM	Identity and Access Management (Gestão de Identidades e Acessos)
ITU-T	ITU Telecommunication Standardization Sector
API	Application Programming Interface
AAA	Authentication Authorization and Audit (Autenticação, Autorização e Auditoria)
MFA	Multifactor Authentication (Autenticação Multifatorial)
DAC	Controle de Acesso Discricionário
MAC	Controle de Acesso Mandatório
RBAC	Controle de Acesso Baseado em Papéis
ABAC	Controle de Acesso Baseado em Atributos
XACML	*eXtensible Access Control Markup Language*
PEP	*Policy Enforcement Point*
PDP	*Policy Decision Point*
PIP	*Policy Information Point*
PAP	*Policy Administration Point*
DLT	*Distribuited Ledger Technology*
IdP	Provedor de Identidade
SP	Provedor de Serviço
SSO	*Single Sign-On*
BYOD	*Bring Your Own Device)*
ECG	Eletrocardiograma
EEG	Eletroencefalografia

ACL	*Access Control List*
PRISM	*Probability Risk and Impact System*
LCE	*Leave a Copy Everywhere*
LCD	*Leave a Copy Down*
LEAF	*Leave a Copy at Leaf*
KNN	*K-Nearest Neighbor*
FIFO	*First-in First Out*
LRU	*Least Recently Used*
RR	*Random Replacement*
SLRU	*Segmented Least Recently Used*
SPROT	*Predicted Segmented Least Recently Used*
HR	Taxa de Acerto
OHHR	Taxa de Acerto em um Salto
NAR	Número de réplicas de atributos

SUMÁRIO

1 INTRODUÇÃO

A Internet das Coisas (IoT, do inglês *Internet of Things*) é um conceito amplamente utilizado pelo mercado e academia. Conforme a literatura, o termo IoT geralmente se refere à integração de capacidades computacionais e de comunicação em diversos objetos do cotidiano, o que permite a coleta, transmissão e recepção de dados por meio de tecnologias de comunicação (Din et al., 2019; Laghari et al., 2021). Na década passada, no entanto, as tecnologias da IoT eram bastante limitadas e suas aplicações, na prática, estavam longe de se tornarem realidade. Usualmente, o que havia eram projetos de aplicações emergentes que utilizavam de forma combinada computadores, *smartphones* e um conjunto bastante limitado de dispositivos, como *tablets* e sensores, para realizar e automatizar algumas tarefas (Rao et al., 2012).

Hoje, essa realidade mudou. A IoT está cada vez mais real, suas aplicações estão cada vez mais complexas e ela já se expandiu para diversos outros dispositivos (Din et al., 2019; Laghari et al., 2021). Por exemplo, em nossas casas, as *smart* TVs coletam dados sobre nossos gostos pessoais e alimentam aplicações capazes de determinar as programações aconselháveis para nós e pessoas com gostos similares (Zhong e Yang, 2020); sensores e dispositivos de sistemas de climatização conectados são capazes de acessar a internet e se ajustarem automaticamente com base nas informações climáticas coletadas tanto da Internet quanto localmente através de diversos sensores (Song et al., 2017; Alaa et al., 2017). Na saúde, a IoT já se faz presente nos sistemas de monitoramento de saúde remotos (Moustafa et al., 2016), nos tradicionais dispositivos médicos, como bombas de insulina, as quais coletam informações de sensores espalhados pelo paciente para determinar a dosagem correta da próxima medicação (Gia et al., 2017) e até mesmo em cápsulas ingeridas com câmeras capazes de coletar e transferir imagens de todo o sistema digestivo de uma forma menos invasiva, se comparada com os métodos tradicionais da medicina (Pramanik et al., 2019). Por fim, na indústria, a IoT está presente desde dispositivos capazes de realizar a compra de produtos, quando eles se encontram em falta na casa do cliente, até em diversos sistemas, sensores e dispositivos para realizar a logística e entrega de produtos (Salah et al., 2020).

Enquanto os benefícios trazidos pela IoT aos seus usuários são inegáveis, a ideia de um mundo totalmente conectado traz consigo diversas preocupações, principalmente em relação à segurança e à privacidade dos dados dos usuários das aplicações IoT (Al-Turjman et al., 2022). Os dispositivos IoT não são dispositivos convencionais, como *laptops* e *smartphones*, que possuem diversas funcionalidades de segurança embutidas. Eles são heterogêneos, usualmente sem padronização e com capacidades computacionais limitadoras da aplicabilidade de medidas de segurança (Uganya et al., 2021). Além disso, devido à sua natureza muitas vezes ubíqua com seus usuários, os dispositivos IoT são responsáveis por coletar informações pessoais preciosas, como informações médicas, gostos pessoais, rotinas diárias e outras informações, o que atrai a atenção de usuários mal-intencionados (Kumar Jain e Gajrani, 2021). Logo, uma vez que está tudo conectado, a superfície de ataque para esse usuário mal-intencionado aumenta consideravelmente e um acesso indevido em um único dispositivo IoT pode expor em risco, não só aquele dispositivo, mas também todas as entidades do sistema IoT (Yadav et al., 2022). Por exemplo, os usuários mal-intencionados podem acessar um sensor de forma indevida e vazar informações médicas de um determinado paciente; podem utilizar esse acesso para um ataque mais elaborado em um controlador de glicose e causar um comportamento anormal na bomba de insulina, colocando a vida do usuário em risco; ou escalar esse acesso indevido em um único dispositivo para acessar o sistema médico e capturar informações pessoais não só daquele paciente, como também de todos os outros pacientes da aplicação médica; e assim por diante. Por essa razão, diversos órgãos e

projetos (OWASP IoT *Project* (Miessler, 2015), *IoT Security Foundation* (Badran, 2019), *IoT Security Compliance Framework* (Hosmer e Hosmer, 2018)) apontam a segurança e a privacidade da IoT como um dos principais desafios e tópicos de atenção para a próxima década. De acordo com projetos de pesquisa (OWASP (Miessler, 2015)) e empresas fortes na indústria (PingIdentity, Forgerock), os acessos indevidos às *Application Programming Interfaces* (APIs), bases de dados e aos dispositivos IoT são atualmente a principal preocupação para a segurança dos usuários e já são a maior razão dos vazamentos de dados que ocorrem atualmente.

Ainda que seja de extrema importância para a segurança dos dados dos usuários, o controle de acesso é um componente muitas vezes negligenciado, inexistente ou ineficaz para IoT (Zaidi et al., 2021; Xu et al., 2022; Fotiou et al., 2022). É importante observar que, para lidar bem com a IoT, é necessário o controle de acesso granular e total controle dos dispositivos IoT, ou seja, deve-se delimitar muito bem os usuários que podem acessá-los e quais funcionalidades podem utilizar (Zhou et al., 2018; Ravidas et al., 2019). Por exemplo, um termostato comercializado pela empresa Nest concede acesso a todas as funcionalidades disponíveis do dispositivo ou nada, ou seja, oferece um controle de acesso pouco granular para IoT e altamente indesejável (He et al., 2018). O produto *Homekit* da Apple também é outro sistema com pouca granularidade. Nele, os usuários são classificados em três categorias e cada uma possui um conjunto de ações que podem realizar sobre o dispositivo de forma fixa e imutável (Jia et al., 2021). Como consequência desses mecanismos não apropriados de autorização de acessos para IoT, já existem diversos relatos e estudos sobre ataques os quais violam a privacidade e a segurança da informação dos usuários de aplicações IoT. Por exemplo, em 2015, um acesso indevido de um usuário malicioso em uma babá eletrônica IoT permitiu que um estranho tivesse acesso às imagens do bebê e ao microfone do dispositivo (Stanislav e Beardsley, 2015). Recentemente, em 2022, acessos indevidos às câmeras de aspiradores de pó IoT foram realizados por usuários maliciosos, resultando em diversas fotos íntimas de seus usuários sendo expostas de forma indevida e inesperada na Internet (Krupp, 2023).

Para evitar essas situações, ou até piores, a solução está nos sistemas de gestão de identidade e acessos (IAM, do inglês Identity and Access Management). Por definição, em sua forma tradicional, um IAM é um conjunto de processos e tecnologias projetados para garantir a identidade de um usuário que acessa um sistema, assegurar a qualidade das informações relacionadas à sua identidade (como dados pessoais, localização, preferências, entre outros) e oferecer seus pilares fundamentais: autenticação, autorização (também conhecida como controle de acesso) e auditoria (Wangham et al., 2018). Historicamente, antes de a IoT se tornar uma tendência, os IAMs eram projetados para gerenciar e proteger o acesso de pessoas a aplicações, redes internas corporativas, ou mesmo para gerenciar o acesso de clientes, consumidores e parceiros a determinadas aplicações. Assim, pode-se afirmar que os IAMs eram *people-driven* (focados em pessoas) e limitados à proteção do acesso de usuários humanos a recursos computacionais. Com a chegada da IoT, o escopo do que um sistema IAM deve gerenciar se expandiu. Enquanto antes os sistemas IAM eram direcionados exclusivamente às pessoas, agora, com a IoT, os sistemas IAM precisam também gerenciar um vasto número de identidades atribuídas a dispositivos e serviços (os chamados usuários impessoais) e lidar com uma nova gama de acessos, cada vez mais complexos (Fan et al., 2020). Ou seja, os IAMs devem evoluir para ir além do conceito *people-driven*, incorporando mecanismos adaptados às demandas específicas da IoT.

É importante observar: como a IoT abrange uma ampla gama de aplicações com objetivos muitas vezes distintos, cada aplicação apresenta seu próprio conjunto de características, prioridades e requisitos específicos (Ravidas et al., 2019; Shakarami et al., 2022). Porém, de maneira geral, são determinados como requisitos-chave para aplicações em IoT a escalabilidade,

o gerenciamento, a interoperabilidade, a disponibilidade e o desempenho, variando de aplicação para aplicação a importância de cada requisito. Por exemplo, as aplicações clássicas IoT como cidades inteligentes, indústrias inteligentes, campus inteligentes e veículos automatizados operam com apertados requisitos de escala, gerenciamento e desempenho. Usualmente, nesse tipo de aplicação, temos uma escala enorme de dispositivos, diversas condições de acesso e novos dispositivos IoT podem se juntar ao ambiente a qualquer momento, requerendo uma arquitetura adaptativa capaz de suportar essas características e um sistema altamente escalável e gerenciável.

Os sistemas IAM são fundamentais para a proteção no contexto da IoT e herdam os requisitos das aplicações nas quais estão inseridos. Por exemplo, as funcionalidades desses sistemas devem ser desenvolvidas considerando a redução dos custos de comunicação, garantindo que aplicações como o monitoramento de pessoas e veículos autônomos operem adequadamente, atendendo à baixa latência exigida por essas aplicações. De forma semelhante, para aplicações médicas, além do desempenho, o foco principal costuma estar em alcançar alta disponibilidade e confiabilidade. Isso é essencial para que, em eventos críticos, como uma emergência médica grave, a aplicação consiga transmitir as informações necessárias para a equipe médica de maneira segura e sem falhas. Dessa forma, é importante destacar que não existe (e provavelmente não existirá) uma solução única na literatura capaz de atender a todas as necessidades. A IoT, devido à diversidade e criticidade de seus requisitos, varia amplamente entre as diferentes aplicações. Cada tipo de aplicação demanda mecanismos personalizados e projetados para atender a seus requisitos específicos. Neste manuscrito de tese, exploram-se os sistemas IAM voltados para a IoT com foco no controle de acesso, especialmente para aplicações que demandam escala, gerenciamento e desempenho. Para ilustrar essas aplicações, são investigados dois cenários específicos: uma indústria inteligente e um campus universitário inteligente. A escolha desses cenários foi motivada pela representatividade deles em relação aos requisitos desta tese.

1.1 MOTIVAÇÃO

Dentro dos conceitos de IAM, define-se que uma política de acesso é um conjunto de regras que regulam o controle de acesso a recursos computacionais e guiam o comportamento esperado de um usuário dentro de uma aplicação. Logo, essas políticas são fundamentais para garantir a segurança e a integridade dos dados, além de assegurar o uso adequado desses recursos. Por essa razão, o gerenciamento e a aplicação inadequados das políticas de acesso podem levar a graves problemas de segurança, como o vazamento de informações confidenciais e o comprometimento de não só um dispositivo, mas de todas as entidades presentes na IoT.

De maneira geral, a literatura atual aponta que, para ser efetivo para IoT, o método de controle de acesso (um dos principais métodos presentes nos IAMs) deve utilizar os atributos das entidades envolvidas para a tomada de decisão no controle de acesso. Em aplicações como indústrias inteligentes e campus inteligentes, existe uma infinidade de propostas na literatura que buscam oferecer controle de acesso com base em atributos para a criação e avaliação de políticas. No entanto, grande parte desses estudos se concentra apenas na aplicabilidade desses modelos em diferentes contextos, pouca atenção a desafios de gerenciamento e desempenho que surgem ao utilizar atributos para criar políticas e avaliar acessos. Para criar e gerenciar políticas de acesso de forma eficiente, é essencial compreender quais recursos são mais utilizados pelos usuários e dispositivos IoT, quais usuários têm acesso a esses recursos e as razões por trás desses acessos. Apenas com essas informações é possível refinar e gerenciar as políticas de acesso, garantindo que apenas o conjunto apropriado de usuários acesse os recursos necessários e autorizados. Embora a literatura reconheça a complexidade do gerenciamento das políticas de acesso — um desafio amplificado pelo volume e pela diversidade das interações na IoT — muitas das soluções

propostas são excessivamente complexas e inadequadas, tornando difícil para um administrador humano compreender as razões por trás das decisões tomadas, como a aprovação ou negação de um acesso específico. É importante notar que a auditoria, sendo um dos pilares dos sistemas IAM, exige que o gerenciamento de políticas seja transparente e explicável. No entanto, um dos problemas críticos identificados na literatura é a falta de auditabilidade em muitas técnicas. Usualmente observa-se técnicas 'caixas-pretas', que tornam a explicação das decisões difícil ou mesmo impossível, comprometendo a transparência necessária para auditorias eficazes e, consequentemente, a confiança no sistema. Portanto, oferecer soluções que combinem eficácia, auditabilidade e facilidade de gerenciamento continua sendo um desafio central no avanço das políticas de controle de acesso na IoT.

De maneira similar, pouca atenção está sendo dada para o desafio que pode surgir em relação à (i) latência e (ii) à consistência durante a recuperação dos atributos necessários para avaliar tais políticas. Vale ressaltar que, devido à vasta flexibilidade e ao elevado número de usuários IoT, os dados sobre os atributos, essenciais para avaliar uma política, frequentemente estão distribuídos em vários repositórios. Estes repositórios podem ser múltiplas nuvens computacionais situadas em locais geograficamente distantes, o que acontece por conta da capacidade de processamento e da escalabilidade da nuvem. Esse fato, por si só, introduz uma latência no momento de acessar esses atributos (i). Além disso, como os atributos geralmente refletem as características dos usuários e dos dispositivos, eles estão sempre sujeitos a mudanças. Por exemplo, o comportamento do usuário ao utilizar uma aplicação específica durante o dia pode fazer com que seus atributos sejam modificados, a localização de seus acessos possa variar, dentre outros aspectos. Logo, os atributos que não são atualizados frequentemente e permanecem não (ii) consistentes acabam sendo menos seguros do que aqueles atualizados em tempo real. Se permanecerem desatualizados, uma decisão de acesso incorreta pode ocorrer. Com isso, uma vez que a decisão de acesso pode ser afetada pelo atraso em recuperar os atributos nesses diferentes repositórios ou ser realizada de forma incorreta através de um atributo inconsistente em um determinado repositório, é essencial investigar técnicas capazes de entregar os atributos necessários para avaliar as políticas de acesso de forma que esses dois desafios sejam superados para garantir a segurança dos acessos sem afetar o desempenho e a escalabilidade necessária para as aplicações IoT.

É importante observar que todas as questões abordadas neste manuscrito de tese têm como ponto de convergência os padrões de comportamento do usuário no processo de controle de acesso. Neste trabalho, o interesse é em explorar quais os impactos desse comportamento em relação à escala, à gerência e ao desempenho da manutenção das políticas de acesso e da recuperação de atributos para a avaliação das políticas. A literatura evidencia que os registros de acesso dos sistemas de controle de acesso podem ser uma fonte valiosa de informação para adaptar e melhorar a eficiência do próprio mecanismo de controle de acesso; porém, existe uma lacuna de trabalhos que utilizam dessa informação para oferecer gerenciamento, escalabilidade e desempenho ao mecanismo de controle de acesso. De forma resumida, o comportamento do usuário localizado nesses registros - onde ocorreu um acesso, como ocorreu, por que ocorreu - pode fornecer *insights* sobre como o processo de controle de acesso ocorre, quais atributos mais utilizados, onde estão sendo utilizados etc. Portanto, essas informações podem ser capturadas via técnicas de aprendizado de máquina e utilizadas para ajustar as políticas de controle de acesso, melhorar o desempenho da recuperação de atributos e diversas outras funções que podem ser exploradas em trabalhos posteriores.

Apresentado esse contexto, a pergunta orientadora desta tese é: *Como a caracterização de padrões de comportamento dos usuários pode ser aplicada para tornar mais eficientes os IAM para IoT? A caracterização dos padrões passa pela análise de registros de acesso ao sistema de*

controle, como políticas mais utilizadas, localização dos acessos e atributos mais utilizados. Este trabalho investiga o uso de técnicas de aprendizado de máquina para identificar padrões de comportamento dos usuários a partir dos registros. Além disso, analisam-se os impactos dessa abordagem na eficiência do mecanismo de controle de acesso em relação ao gerenciamento, ao desempenho da manutenção das políticas de acesso e na recuperação de atributos para avaliação das políticas.

1.2 PROBLEMA

O problema tratado neste trabalho é definido da seguinte forma: *O rápido e contínuo crescimento da IoT impõe desafios significativos aos sistemas de IAM, uma vez que, atualmente, esses sistemas não conseguem assegurar o desempenho, o gerenciamento e a escalabilidade necessários para satisfazer as exigências de controle de acesso nas aplicações IoT. Isso inclui dificuldades em manipular e implementar políticas de acesso de forma gerenciável em ambientes IoT altamente dinâmicos e complexos, bem como problemas de desempenho e escalabilidade, como latência na avaliação de políticas de acesso em tempo real, especialmente no mecanismo responsável por recuperar os atributos necessários para a avaliação das políticas*. Há uma lacuna significativa na pesquisa sobre a utilização das informações sobre o comportamento do usuário para melhorar os mecanismos de IAM na IoT. Acredita-se que, através do comportamento do usuário, os sistemas IAM podem personalizar e adaptar seus componentes e políticas de acesso de acordo com os padrões de uso dos usuários. Logo, com esse comportamento, é possível antecipar as necessidades dos acessos e permitir que os sistemas IAM se preparem com antecedência, pré-carregando atributos ou facilitando processos de manutenção das políticas de acesso com base nos acessos prévios do usuário. Portanto, o problema de pesquisa tratado busca preencher essa lacuna e investigar a capacidade dos sistemas IAM de utilizar os dados de comportamento do usuário para adaptar e aprimorar o controle de acesso, diminuir a complexidade do gerenciamento de políticas de acesso e melhorar o desempenho do processo de recuperação e sincronização de atributos na avaliação das políticas. Ou seja, apesar de abranger aspectos como desempenho, escalabilidade e gerenciamento de políticas, é importante ressaltar que todos os problemas convergem em um único foco fundamental: como utilizar o comportamento do usuário para otimizar e adaptar o controle de acesso em ambientes IoT.

1.3 HIPÓTESE E OBJETIVOS

A hipótese central deste trabalho propõe que, ao analisar um amplo conjunto de dados sobre o comportamento do usuário no contexto do controle de acesso (incluindo políticas avaliadas, resultados de autorizações, localizações de acesso e atributos mais utilizados), é possível empregar técnicas de aprendizado para identificar e classificar padrões de comportamento. A partir desses padrões, os sistemas IAM podem adaptar especificamente o mecanismo de controle de acesso, de modo a simplificar o gerenciamento e a manutenção das políticas, otimizar o desempenho na recuperação e sincronização de atributos, além de fortalecer a segurança e a escalabilidade em ambientes IoT. Em outras palavras, embora essa proposta envolva várias etapas – como coleta e análise de dados, definição de padrões comportamentais e ajuste dos componentes de IAM – todas convergem para um único objetivo: tornar o método de controle de acesso mais eficiente e adequado às aplicações IoT. A compreensão do comportamento do usuário permite otimizar a gestão de atributos e políticas de acesso, resultando em um sistema IAM mais escalável e seguro.

Para validar a hipótese levantada e responder à pergunta orientadora deste trabalho, pretende-se caracterizar e modelar padrões típicos do comportamento sob a perspectiva do controle de acesso de usuários que acessam uma aplicação (i); entender o impacto e as vantagens de se utilizar esses comportamentos para criar e refatorar políticas de acesso (ii); analisar as compensações de custo-benefício em relação ao desempenho e à segurança de se utilizar os comportamentos para distribuir os atributos na rede (iii). Dessa forma, o objetivo geral deste trabalho é propor uma arquitetura e métodos que utilizem o comportamento do usuário para apoiar o mecanismo de autorização, tornando-o mais escalável, gerenciável e com melhor desempenho.

O objetivo geral descrito acima foi dividido em três objetivos de pesquisa específicos (OEs). Cada objetivo investiga um aspecto particular para preencher uma lacuna dos problemas que o controle de acesso enfrenta em aplicações IoT. Os objetivos de pesquisa específicos e uma breve descrição das hipóteses secundárias vêm a seguir:

1. **OE1 - Avaliar o comportamento do usuário para gerenciar políticas de acesso:** como nosso primeiro objetivo de pesquisa, nós investigamos diferentes abordagens que utilizam de métodos de aprendizado de máquina para solucionar o problema de gerenciamento de políticas em IoT. Logo, a pergunta chave que queremos responder é: *técnicas de aprendizado de máquina podem ser utilizadas como uma opção para auxiliar a manutenção de políticas de acesso de forma auditável?* Estudos presentes na literatura apontam que atualizações manuais, ou seja, feita por humanos em políticas de acesso é uma tarefa propensa a erros e é considerada hoje a maior culpada por vulnerabilidades de segurança. Para solucionar esse problema, diversos trabalhos apostam em abordagens de aprendizado de máquina, mas entregam soluções *black-box* que não oferecem legibilidade das políticas de acesso - o que as torna não auditáveis e fere um dos pilares básicos dos IAMs, a auditoria. Para ir em direção ao nosso OE1, em contraste com a literatura, pretendemos investigar técnicas as quais podemos facilmente compreender e extrair a lógica da política a que levou a uma decisão de acesso.

2. **OE2 - Modelar o comportamento de usuários em aplicações reais:** nosso segundo objetivo de pesquisa é focado na caracterização e modelo do comportamento típicos de usuários em uma aplicação sob a perspectiva do controle de acesso. Neste trabalho, os dados coletados são referentes a acessos que ocorrem em uma aplicação universitária. Especificamente, nosso objetivo é fornecer um modelo no qual diversas políticas e atributos do usuário são coletados para o desenvolvimento de pesquisa relacionadas ao controle de acesso - desde as políticas e atributos utilizados para determinar se um acesso é ou não válido até a data e a hora em que ocorreu a avaliação daquele acesso. Até o momento de divulgação desta tese, apenas alguns *datasets* são disponíveis publicamente para pesquisas e experimentos. Além disso, a qualidade dos dados ali presentes são questionados pela literatura. Usualmente, esses *datasets* são altamente desbalanceados ou contêm informações incompletas relacionadas ao controle de acesso. Sendo assim, nós acreditamos que esse *dataset* gerado por nós pode ser uma fonte de dados preciosa para trabalhos futuros. Esse *dataset* será utilizado para impulsionar os resultados obtidos no nosso OE3.

3. **OE3 - Avaliar o comportamento do usuário para distribuição de atributos em caches:** A partir das conclusões do OE1 e OE2, analisaremos como melhorar o desempenho da recuperação de atributos para a avaliação das políticas de acesso. Sendo assim, dado o problema de pesquisa, esse OE procura responder à seguinte pergunta: *o conceito de caching, aliado com a utilização do comportamento do usuário, é capaz*

de melhorar o desempenho da recuperação de atributos?. A literatura aponta que a utilização de caches pode melhorar significativamente o desempenho do método de controle de acesso. Porém, as soluções utilizam apenas algoritmos clássicos de cache e ignoram os problemas que a cache pode introduzir para a segurança. Note que, ao utilizar caches, problemas de consistência dos atributos são introduzidos, pois os atributos em cache utilizados para a decisão podem estar desatualizados em relação aos atributos armazenados no repositório de identidades. Portanto, ao utilizar caches, é necessário responder simultaneamente à seguinte pergunta: *Considerando que atributos inconsistentes geram decisões de acesso incorretas, pode-se criar um mecanismo para sincronização e revalidação de acessos?*. Para ir em direção ao OE3, pretende-se investigar o *trade-off* entre o desempenho ganho pela cache e os custos que as caches introduzem através de um mecanismo de segurança que realiza a sincronização e revalidação de acessos.

1.4 CONTRIBUIÇÕES

As contribuições desta tese são apresentadas a seguir:

- **Revisão bibliográfica sobre os conceitos e mecanismos da IAM e discussões sobre os desafios que a IoT introduziu a eles.** Em relação à primeira contribuição, a revisão da literatura incluiu uma compreensão geral dos principais conceitos e mecanismos utilizados nos sistemas IAM. Adicionalmente, com a apresentação dos conceitos de IoT e os impactos que ela gera nos sistemas de gestão de identidade, esta tese identifica diversas questões em aberto e oportunidades de pesquisa para o desenvolvimento de modelos, técnicas e sistemas para tornar os IAMs mais adequados para IoT.

- **Um método para mineração de políticas de acesso auditáveis com base em comportamentos.** Essa contribuição consiste em um método de mineração de políticas auditáveis com base em comportamentos de acesso que fornecem uma maneira automatizada para gerenciar e aprimorar as políticas de acesso. De maneira geral, o método proposto nessa contribuição pode ajudar a identificar padrões e tendências dos acessos, permitindo que as políticas sejam ajustadas para melhor atender às necessidades da aplicação. Diferentemente da literatura, nosso método oferece um resultado legível para humanos com o intuito de auxiliar a compreensão da aplicação e a auditoria do mecanismo de controle de acesso.

- **Caracterização de um modelo de controle de acesso executado em um campus universitário.** A coleta e a caracterização de acessos reais foram fundamentais em nossa pesquisa e contribuições relevantes desta tese. Como abordado pela literatura, os dados que existem de usuários, acessando uma aplicação, são usualmente privados ou com diversas informações ausentes. Logo, o conhecimento atual de como ocorrem os acessos é limitado, o que torna pesquisas na área uma tarefa desafiadora. O objetivo é criar um *dataset* que utiliza diversos dados e podem impactar no acesso e fornecer essas informações publicamente para novas pesquisas.

- **Uma arquitetura para a distribuição de atributos em caches.** Essa contribuição consiste em uma arquitetura que permita que os atributos utilizados sejam armazenados em caches distribuídas, o que pode melhorar, de maneira significativa, o desempenho da avaliação das políticas. O objetivo é utilizar essas caches para oferecer a escalabilidade

e a confiabilidade dos modelos de controle de acesso necessárias para IoT. Diferente da literatura, nesta tese é demonstrada, através do *dataset* coletado, como uma arquitetura real poderia ser estabelecida.

- **Uma técnica para distribuição de atributos em caches que utiliza características do acesso para uma melhor replicação dos atributos.** Para essa contribuição, propõe-se uma técnica que utiliza informações sobre o acesso, como tipo do recurso acessado, a localização do usuário e o momento do acesso, para determinar quais atributos de controle de acesso devem ser replicados em diferentes caches, ou seja, o objetivo é fazer com que os atributos mais relevantes e mais frequentemente acessados possam ser armazenados nas caches próximas dos acessos, enquanto os atributos menos relevantes podem ser armazenados em caches remotas. Ao utilizar o comportamento do usuário para tomar essa decisão de onde replicar, os resultados mostraram que essa técnica pode melhorar significativamente o desempenho da recuperação de atributos, sendo capaz de reduzir a latência da avaliação da política como um todo.

- **Uma técnica para atualização e sincronização de atributos e acessos.** Essa contribuição consiste em um mecanismo de sincronização de acessos que considera atributos mutáveis. O objetivo é criar um método formal para uma técnica de atualização e sincronização de atributos e acessos. O objetivo aqui é fornecer uma maneira eficaz e confiável para gerenciar os atributos mutáveis nas caches e estabelecer a consistência necessária, usando atualizações nas decisões de acesso nesse contexto. Para consolidar esses resultados, esta tese utiliza o *dataset* coletado.

1.5 PUBLICAÇÕES DERIVADAS DESTA TESE

A seguir, a lista das publicações derivadas direta ou indiretamente a partir deste trabalho de doutoramento.

- Cremonezi, B. M.; Vieira, A. B. ; Nacif, J. A. ; Nogueira, M. . **Identity Management for Internet of Things: Concepts, Challenges and Opportunities** submetido em Computer Communications, 2023.

- Cremonezi, B. M.; Vieira, A. B. ; Nacif, J. A. ; Silva, E. F. . **A Bi-directional Attribute Synchronization Mechanism for Access Control in IoT Environments** publicado em MobiCASE, 2022

- Cremonezi, B. M.; Gomes, A. R. ; Nacif, J. A. ; Nogueira, M. ; Silva, E. F. ; Vieira, A. B. . **Improving the attribute retrieval on ABAC using opportunistic caches for Fog-Based IoT Networks** publicado em Computer Networks, 2022.

- Gomes, A. R. ; Cremonezi, BRUNO MARQUES ; Nacif, J. A. ; Nogueira, M. ; Silva, E. F. ; Vieira, A. B. . **Uma Política de Cache de Identidades Multinível para Névoas Computacionais** publicado em Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais, 2021. (Best Paper)

- Gomes, A. R. ; Cremonezi, BRUNO MARQUES ; Nacif, J. A. ; Nogueira, M. ; Silva, E. F. ; Vieira, A. B. . **Opportunistic Attribute Caching: Improving the Efficiency of ABAC in Fog-Based IoT Networks.** publicado em IEEE International Conference on Communications, 2021.

- Cremonezi, B. M.; Vieira, A. B. ; Silva, E. F. ; Nacif, J. A. ; Nogueira, M. . **Um Método para Extração e Refinamento de Políticas de Acesso baseado em Árvore de Decisão e Algoritmo Genético.** publicado em Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2021.

- Bastos, L. L. ; Cremonezi, B. M. ; Santos, T. ; Rosário, D. ; Cerqueira, E. ; Santos, A. . **Smart Human Identification System Based on PPG and ECG Signals in Wearable Devices.** publicado em IEEE International Wireless Communications and Mobile Computing Conference, 2021.

- Cremonezi, B. M.; Vieira, A. B. ; Nacif, J. A. ; Lima, M. N. . **Um Sistema Multinível de Distribuição de Identidades em Névoas Computacionais.** publicado em Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2019.

- NAKAYAMA, F. ; LENZ, P. ; Cremonezi, B. M. ; Banou, S. ; Rosário, D. ; Chowdhury, K. ; Nogueira, M. ; Cerqueira, E. ; Santos, A. . **Autenticação Contínua e Segura Baseada em Sinais PPG e Comunicação Galvânica.** publicado em Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2019.

1.6 ORGANIZAÇÃO DO TEXTO

Este manuscrito de tese está dividido em cinco capítulos. O Capítulo 2 aborda os conceitos fundamentais da gestão de identidade, essenciais para entender o problema tratado, além de discutir as aplicações IoT e os desafios que elas impõem aos sistemas IAM. O Capítulo 3 apresenta a proposta de gerenciamento automatizado de políticas de acesso. No Capítulo 4, descreve-se a arquitetura sugerida e o repositório de identidades, discutindo a técnica de distribuição e sincronização de atributos e acessos entre diferentes repositórios. Por fim, o Capítulo 5 apresenta as considerações finais e fornece orientações para trabalhos futuros.

2 FUNDAMENTOS

Neste capítulo, será fornecida uma visão geral sobre o gerenciamento de identidades digitais em IoT, iniciando desde a definição básica do conceito de identidades até a descrição dos diversos mecanismos e modelos. O capítulo está organizado em duas partes. A primeira parte apresenta os fundamentos de IAM e é composta pela Seção 2.1, que introduz os conceitos de identidades digitais e sua importância na atualidade; pela Seção 2.2, em que discutem-se os IAMs e os seus componentes; e pela Seção 2.3, onde aborda-se os modelos de IAMs que podem ser adotados para garantir a segurança e o controle de acesso a recursos. Na segunda parte, este capítulo tem como foco dois aspectos fundamentais para esta tese: uma análise abrangente das aplicações e das tecnologias da IoT, e os desafios que essas aplicações introduziram aos componentes dos IAMs. A Seção 2.4 apresenta uma análise abrangente da IoT, explorando os conceitos e tecnologias que a tornam possível, além de discutir as promissoras aplicações da IoT em diversos setores da sociedade, como saúde, transporte e indústria. Também são abordadas arquiteturas e requisitos fundamentais para essas aplicações. A Seção 2.5, por sua vez, concentra-se nos desafios que a IoT trouxe aos componentes dos IAMs. Aqui, é demonstrado como a IoT torna mais desafiadores os requisitos de segurança e quais são os desafios dessas alterações para os componentes dos IAMs. Por fim, a Seção 3.5 apresenta um resumo dos pontos discutidos neste capítulo.

2.1 CONCEITOS DE IDENTIDADES DIGITAIS

O termo identidade é utilizado para referenciar uma representação de uma entidade, e essa representação ocorre através de um conjunto de reivindicações. Por exemplo, quando se pensa em pessoas, elas podem ser representadas através de reivindicações como nome, sobrenome, apelido, altura, peso e data de nascimento. Essas reivindicações são também chamadas de atributos e carregam elementos característicos da entidade a qual elas representam. É importante notar que cada identidade é exclusivamente associada a uma entidade, mas o contrário não é verdadeiro. Uma entidade pode possuir várias identidades, e cada identidade apresenta apenas atributos válidos e convenientes para um determinado contexto. Por exemplo, a identidade de uma pessoa para uma seguradora tende a ser diferente da identidade dessa mesma pessoa em um hospital. Para a seguradora, informações como saldo, transações anteriores e renda atual são valiosas e representam a pessoa naquele contexto. Para o hospital, por sua vez, deseja-se que cada identidade possua outras informações, como plano de saúde, internações passadas, tipo sanguíneo, doenças crônicas, etc. Essas múltiplas identidades são chamadas de identidades parciais e o conjunto de todas as identidades parciais é chamado de identidade global. Além disso, identidades podem ser permanentes ou temporárias para um determinado contexto, podendo em algumas ocasiões durar anos e em outras apenas segundos.

Embora o conceito de identidade seja frequentemente associado a pessoas, no mundo digital ele vai muito além disso, abrangendo dispositivos, serviços e aplicações. Por exemplo, na Figura 2.1, considera-se que Alex é um usuário humano de uma aplicação em um campus universitário. A identidade de Alex é formada por um conjunto de atributos relevantes que o representam nessa aplicação, como nome, profissão e departamento. Além dos usuários humanos, todos os dispositivos utilizados na aplicação também possuem suas próprias identidades, representadas por um conjunto distinto de atributos específicos para dispositivos, como modelo, endereço MAC, *firmware*, entre outros, conforme ilustrado na Figura.

Figura 2.1: Relação entre entidade, identidade e atributos

Para formalizar o conceito de identidade, este manuscrito de tese adota a definição proposta pela recomendação da ITU *Telecommunication Standardization Sector* (ITU-T). De acordo com essa recomendação, uma identidade digital é composta por três componentes principais: identificadores, atributos e credenciais. Os identificadores são uma sequência de dígitos, caracteres, símbolos ou qualquer outro tipo de dado que pode ser utilizado como um índice único dentro de um determinado contexto. Já os atributos correspondem a um conjunto de informações relacionadas à entidade representada por aquela identidade no contexto de uma aplicação. É importante observar que, embora o endereço MAC seja frequentemente utilizado como um identificador único de dispositivos, no exemplo ilustrado na Figura 2.1, ele é tratado apenas como um atributo da identidade do objeto na aplicação específica e não desempenha o papel de identificador daquela identidade. Por fim, as credenciais são o conjunto de dados que permitem a uma entidade provar que uma determinada identidade a representa. Em outras palavras, as credenciais estabelecem a ligação entre uma entidade e sua identidade por meio de um processo chamado autenticação, que será discutido nas próximas seções.

2.2 GESTÃO DE IDENTIDADES E ACESSOS

No mundo digital, diversos modelos são apresentados na literatura para definir um conjunto de regras e boas práticas destinadas a garantir a segurança das aplicações. Dentre esses modelos, um dos mais recentes e emergentes é o *Zero-Trust*. Criado para atender às necessidades modernas da computação, como acesso a *clouds* e trabalho remoto, o modelo *Zero-Trust* opera com base no princípio do "acesso com menor privilégio". Esse princípio estabelece que, em uma aplicação, as permissões devem ser concedidas de forma seletiva, permitindo que apenas um conjunto específico de usuários e dispositivos tenha acesso a determinados recursos computacionais, e somente pelo tempo estritamente necessário. Para cumprir esse modelo, um dos requisitos fundamentais é que "todos os acessos às informações e recursos computacionais sejam realizados por usuários e dispositivos devidamente autenticados e autorizados". Nesse contexto, a existência de um IAM robusto, capaz de identificar de maneira única cada entidade no sistema, torna-se indispensável. Sem essa base sólida, as operações básicas de autenticação e autorização tornam-se inviáveis no modelo *Zero-Trust*. Além disso, para garantir que todas as entidades possuam uma identidade bem definida, as aplicações geralmente adotam um ciclo de vida para as identidades, que especifica como elas são criadas, utilizadas, gerenciadas e descartadas. Hoje, a literatura busca encontrar soluções para as diversas etapas desse ciclo de vida e oferecer mecanismos que possibilitem a efetiva aplicação e sustentação do modelo *Zero-Trust*. A Figura 2.2 apresenta um ciclo de vida genérico para identidades, que pode ser aplicado em diversas aplicações. Este ciclo é composto por cinco fases principais: provisionamento,

propagação, utilização, manutenção e desprovisionamento. Cada uma dessas fases desempenha um papel crucial no ciclo de vida de uma identidade, garantindo a segurança e a eficiência das operações.

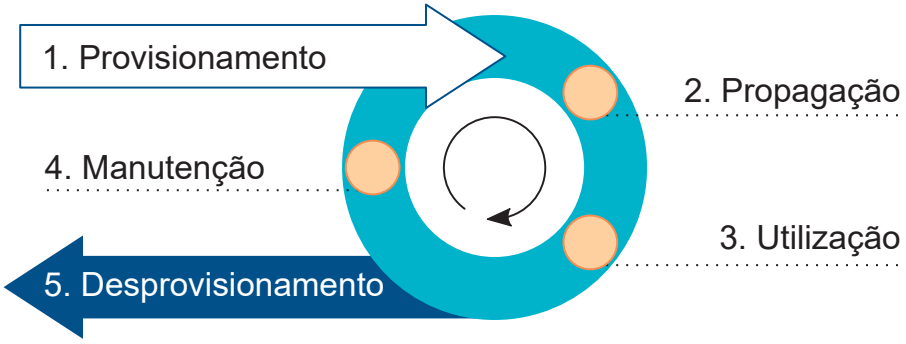


Figura 2.2: *Framework* genérico do ciclo de vida das identidades

Cada contexto de aplicação pode possuir seu próprio ciclo de vida para suas identidades. No entanto, planejar cada fase é essencial para construir uma arquitetura de identidade sólida para lidar com as aplicações. Resumidamente, uma identidade é criada quando, na fase de provisionamento, um usuário se registra no sistema. Após essa criação, o usuário está devidamente registrado e todas as informações do usuário são propagadas para um banco de dados (usualmente chamado de repositórios de identidades). Após essa propagação, as aplicações são capazes de utilizar essa identidade para determinar se um determinado usuário é legítimo e quais ações ele pode tomar. Neste trabalho, assume-se que as aplicações são capazes de utilizar a Internet ou qualquer infraestrutura de rede para realizar essa propagação. Ocasionalmente, algumas mudanças nessa identidade podem ocorrer; por exemplo, atributos extras podem ser adicionados, as credenciais podem mudar, etc. Nesses casos, é necessário realizar uma atualização na identidade, implicando que essa identidade necessita ser propagada novamente para evitar problemas de consistência da identidade. Por fim, quando um usuário sai do sistema, ou seja, sua identidade serviu ao seu propósito e não é mais necessária, ela é destruída (Toelen, 2008).

Provisionamento de Identidade: essa fase representa a criação de uma identidade, o que faz uma entidade a se tornar um usuário de uma aplicação. Nessa etapa, criam-se o identificador único de uma identidade, as credenciais e se coletam os atributos necessários do usuário, por exemplo, localização, e-mail, nome, ou qualquer informação relevante para uma determinada aplicação. O provisionamento pode ocorrer de duas formas: a primeira, um usuário, usualmente um humano, registra-se em uma aplicação ao enviar seus atributos e uma informação para ser utilizada como credencial. Nesse momento, a aplicação pode checar a validade, a autenticidade e a acurácia desses atributos e estabelecer uma conexão desse usuário com uma identidade. A segunda forma ocorre de maneira não explícita. Nesse caso, a identidade de um usuário é construída baseada na coleção de atributos indiretos (não explicitamente declarados, mas observados sob a perspectiva da aplicação) do usuário em um determinado contexto, por exemplo, um endereço de IP que pode ser rastreado até o provedor, que, por sua vez, pode ser associado ao usuário utilizador daquele IP (Ahson e Ilyas, 2008). Assim, quando a construção da identidade é realizada através da maneira indireta, o usuário, muitas vezes, não tem nem conhecimento da construção de sua identidade (Roussos et al., 2003).

Propagação da Identidade: determinadas aplicações necessitam de que frações da identidade sejam replicadas para outras aplicações ou sistemas. O objetivo dessa replicação é simples: algumas aplicações necessitam replicar sua identidade para obter um melhor desempenho, reduzir custos ou oferecer redundância para um mecanismo de defesa em casos de falhas no

sistema. Idealmente, essa propagação deve ocorrer após cada alteração em um atributo de uma determinada identidade e essa propagação deve ocorrer de modo confiável e rápido, com o intuito de evitar problemas de consistência (Windley, 2005).

Utilização da Identidade: essa é considerada a fase mais direta no ciclo de vida das identidades. Nessa fase, diversas aplicações e usuários utilizam as informações da identidade para verificar se um usuário é legítimo e para aplicar operações de autorização, que permitam a um determinado usuário realizar uma determinada ação naquela aplicação (Windley, 2005). Por exemplo, suponha que exista um usuário que deseja acessar um recurso computacional e, para acessar esse determinado recurso, deve-se utilizar a identidade do usuário que solicitou o acesso e checar se um atributo dessa identidade possui um determinado valor específico.

Manutenção da Identidade: usualmente, identidades não são dados estáticos, pois atributos e credenciais podem ser alterados no ciclo de vida das identidades. Por exemplo, uma característica base de um usuário pode se alterar ao longo do tempo ou algumas aplicações necessitam de novos atributos para se adequar às novas oportunidades de negócios. Independentemente do fator que motivou uma alteração na identidade, após realizada, é necessário que todas essas informações sejam propagadas para as outras aplicações e sistemas, para evitar, novamente, os problemas de consistência (Shakarami e Sandhu, 2019).

Desprovisionamento da Identidade: remover identidades ao fim do ciclo de vida é uma tarefa tão crucial quanto provê-las. O desprovisionamento consiste no processo de sinalizar para as aplicações quais usuários não são mais válidos. Em um primeiro momento, espera-se que o desprovisionamento remova por completo todos os dados relacionados à identidade desse usuário não válido. No entanto, remover essas informações podem gerar problemas de consistência para o processo de auditoria que monitora as atividades de cada usuário dentro da aplicação. Logo, uma das opções para desprovisionar uma identidade sem afetar o processo de auditoria consiste em desabilitar uma identidade. Nesse caso, as credenciais de uma identidade se tornam inválidas, o que faz com que o usuário não esteja mais associado com sua identidade; porém, as informações da identidade permanecem intactas. Assim, as aplicações minimizam os desafios na auditoria enquanto removem um usuário não válido. No entanto, os custos em manter essas informações podem sobrecarregar os sistemas, visto que o número de usuários não ativos pode crescer indefinidamente. Uma forma de lidar com ambos os problemas mutuamente consiste em utilizar uma forma híbrida de desprovisionamento. Ao combinar ambas as técnicas anteriores, o desprovisionamento híbrido remove as credenciais dos usuários não válidos e define uma data limite para a remoção do restante dos dados da identidade (Toelen, 2008).

2.2.1 Autenticação, Autorização e Auditoria

Autenticação, Autorização e Auditoria (AAA) são componentes fundamentais de qualquer IAM e fazem parte da etapa de utilização da identidade em seu ciclo de vida. Esses componentes são essenciais pois, através deles, os acessos são controlados e monitorados. Para ilustrá-los, a Figura 2.3 apresenta um esquema em que um usuário humano conectado em seu laptop deseja realizar um acesso em um determinado serviço oferecido por um recurso computacional (um dispositivo IoT, um servidor na nuvem, um banco de dados etc.). Note que, para realizar esse acesso, duas barreiras devem ser superadas: a barreira da autenticação - os usuários e dispositivos que realizam a requisição de acesso são identificados - e a barreira da autorização - as permissões dos usuários e dispositivos autenticados são verificadas para determinar se esse acesso pode ocorrer. Independentemente do resultado, a auditoria armazena todas as entradas e saídas, criando *logs* com todas as tentativas de acesso e seu resultado.

O componente de autenticação se baseia na ideia de que cada entidade no sistema possui uma informação específica distinguindo-a das demais. Essa informação é a credencial e, quando

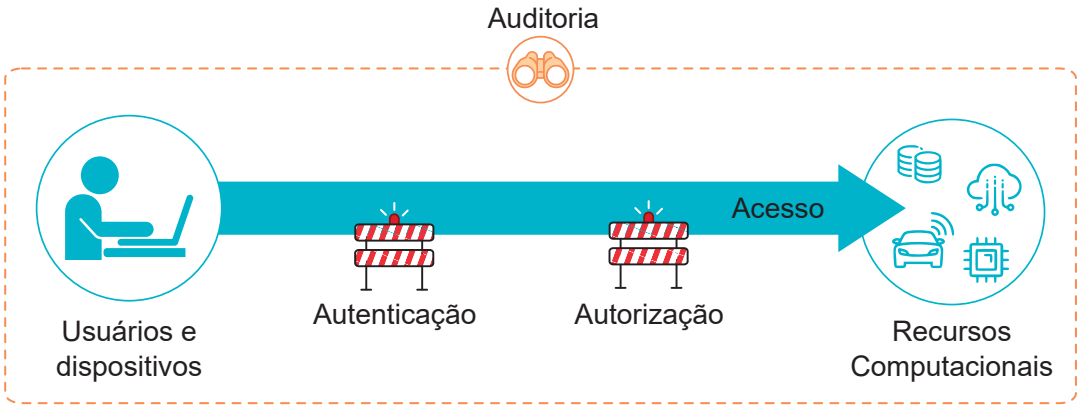


Figura 2.3: Autenticação, Autorização e Auditoria

apresentada durante a autenticação, prova que uma determinada entidade é a proprietária de uma identidade, ou seja, em um ambiente controlado e sem erros, uma entidade possui uma credencial conhecida só por ela e a oferece para o IAM. O IAM realiza o processo de autenticação através da verificação da credencial oferecida e conclui que uma entidade é autêntica e proprietária daquela uma identidade. Após concluído o processo de autenticação, para determinar se um acesso é válido ou não, inicia-se o processo de autorização. Visto que na grande maioria dos sistemas não é desejado que todas as entidades acessem todas as informações (por exemplo, um estagiário não deve possuir o mesmo acesso do que um CEO em uma empresa), a autorização é composta por um conjunto de regras o qual reflete as condições e políticas a que um acesso deve seguir para ser aceito. A forma como essas regras são construídas é determinada pelo método de autorização, que será apresentado nas subseções seguintes, mas usualmente envolvem ao menos um atributo de suas identidades. Por fim, a auditoria é o processo de registrar todas as atividades que um usuário exerceu enquanto estava no sistema (mesmo que não tenha sido autenticado e autorizado). Assim, na auditoria é rastreada todas as atividades e resultados como autenticações bem-sucedidas, autenticação falha, autorizações concedidas, autorizações negadas, recursos acessados etc.. Ainda que não seja o foco desta tese em determinar como realizar esta atividade, a auditoria é muito importante para os IAMs, pois, através dela, é possível rastrear eventos que causaram um incidente de segurança, por exemplo, ou até mesmo traçar um comportamento padrão de um determinado usuário e utilizar isso para revisar algum mecanismo presente no IAM.

2.2.1.1 Métodos de Autenticação

A autenticação consiste no processo de verificação se um usuário é proprietário de uma respectiva identidade através da comparação das credenciais (Torres et al., 2012). O tipo dessa credencial é chamado de fator de autenticação. Atualmente, o tipo de autenticação é determinado pelo fator utilizado. Dentre os diversos tipos de autenticação existentes, podemos classificá-los como: autenticação baseada no conhecimento, na propriedade, na característica e no contexto (Ometov et al., 2019). Note que o tipo da autenticação é determinado pelo fator de autenticação. Porém, é importante ressaltar que uma autenticação não está limitada a um único fator e pode ser realizada através da utilização de vários fatores. Nesses casos, quando a autenticação utiliza dois fatores, ela é chamada de autenticação bifatorial (2FA, do inglês *Two-factor Authentication*), e quando utiliza múltiplos fatores, ela é chamada de autenticação multifatorial (MFA, do inglês *Multifactor Authentication*) (Abhishek et al., 2013).

A **autenticação baseada no conhecimento:** essa autenticação requisita que o usuário apresente uma informação conhecida exclusivamente por ele. Usualmente, esse método de autenticação é referido como "autenticação baseada no que você conhece" e, devido à sua simplicidade, é amplamente utilizado atualmente (Velásquez et al., 2018). Para usuários humanos, o maior exemplo desse método de autenticação são utilizados os tradicionais nomes de usuário (amplamente conhecidos como *usernames*) e senha (do inglês, *passwords*), em que o usuário apresenta um identificador de identidade e uma senha, conhecida preferencialmente somente por ele (Abhishek et al., 2013). Apesar de sua popularidade, esse método de autenticação depende exclusivamente da capacidade de memorização do usuário, implicando que a senha não deve ser esquecida pelo usuário, pois, sem ela, esse usuário perde o acesso à sua identidade (Abhishek et al., 2013). Quando esse método de autenticação é aplicado para verificar a identidade do dispositivo na rede, existe a necessidade de o dispositivo armazenar essa senha na memória (Choi et al., 2018). Esse armazenamento implica que, em um cenário com um massivo número de dispositivos, gerenciar a senha de cada dispositivo individualmente é uma tarefa maçante, além de insegura, uma vez que a baixa capacidade computacional dos dispositivos pode implicar na ausência de mecanismos de segurança (Nawir et al., 2016).

A **autenticação baseada em propriedade:** essa autenticação é usualmente descrita como "autenticação baseada em algo que o usuário possui". Para usuários humanos, utilizam-se objetos específicos para provar uma identidade. Os exemplos mais comuns desses objetos são: dispositivos geradores de senha, cartões de identificação, *smart cards*, *tokens* e etc.. Essa categoria apresenta os mesmos problemas de segurança da anterior, pois, para se autenticar, é necessário ter a posse do objeto. No entanto, tal objeto pode ser danificado, perdido ou até mesmo roubado, o que resulta no usuário impossibilitado de acessar sua identidade (Nawir et al., 2016). Para dispositivos IoT, esse método pode ser representado por um segredo armazenado no dispositivo, por exemplo, uma chave criptográfica simétrica. Nesse caso, a capacidade de se comunicar com um terceiro já atua como uma prova de que o dispositivo possui o fator de propriedade (Choi et al., 2018).

A **autenticação baseada em característica:** também conhecida como autenticação biométrica, esse método é reconhecido hoje como uma das formas mais seguras de se autenticar, pois utiliza-se de fatores biológicos de um usuário (impressões digitais, reconhecimento de voz, íris, padrões de veias etc.) para a verificação de identidade (Wayman et al., 2005; Bhattacharyya et al., 2009; Merhav, 2019). De maneira geral, esse fator é beneficiado pelo fato de que os fatores biométricos são únicos para cada pessoa e não podem ser esquecidos ou perdidos. No entanto, como apontado por Bhattacharyya et al. (2009), grande parte desses fatores é constituída por informações públicas, o que torna esse tipo de autenticação propenso a fraudes. Impressões digitais, por exemplo, podem ser burladas através de uma réplica de silicone. Como resultado, essa credencial usualmente é aplicada em conjunção com as outras duas, com o intuito de fortalecer a garantia de identidade de um usuário (Bhattacharyya et al., 2009).

A **autenticação baseada em contexto:**essa autenticação é um fator adicional utilizado para aumentar a robustez dos outros tipos de autenticação (Habib e Leister, 2015). Nesse tipo, as informações contextuais como localização geográfica, horário e tecnologia de comunicação podem ser utilizadas em conjunto com os outros tipos de autenticação com o intuito de oferecer uma verificação de identidade mais precisa, pois possibilita a identificação de comportamentos suspeitos por parte do usuário. Por exemplo, um usuário que se autentica para acessar o serviço em duas posições geograficamente distantes em um curto período de tempo, usualmente é apontado como um acesso suspeito (Habib e Leister, 2015).

2.2.1.2 Métodos de Autorização

Um método de autorização está relacionado ao modelo de controle de acesso. De maneira geral, esse modelo determina um conjunto de requisitos pelos quais um usuário deve alcançar acesso ao serviço. Devido à infinidade de aplicações em computação, diversos modelos de controle de acesso foram sugeridos ao longo da história e que podem ser aproveitados para IoT. Dentre os principais modelos mais reconhecidos hoje, temos o controle de acesso discricionário, mandatório, baseado em papéis e baseado em atributos.

Controle de acesso discricionário (DAC, sigla do inglês *Discretionary Access control*) (Jin et al., 2012) é um método de controle de acesso desenvolvido inicialmente para sistemas operacionais, que foi transposto para o contexto de redes (Andaloussi et al., 2018). Nesse método de controle de acesso, um proprietário de um recurso (dispositivo IoT, por exemplo) especifica quais usuários podem acessá-lo e sob quais condições. Para isso, criam-se, dentro de cada recurso, diversas estruturas de dados contendo um conjunto de regras de acesso que guiará todo o processo de autorização (Sandhu e Munawer, 1998). Dentre os exemplos de regras de acesso, temos listas de controle de acesso, matrizes de acesso e tabelas de autorização que, de maneira geral, são estruturas capazes de armazenar um conjunto de permissões e condições de acesso de um usuário sob aquele recurso. Do ponto de vista de um único recurso, esse método permite um controle total ao proprietário em relação a quais usuários podem acessar o recurso, quais condições são necessárias e quais operações são válidas para cada usuário (Andaloussi et al., 2018). Porém, quando aplicada em uma grande quantidade de recursos, a falta de uma administração centralizada faz com que o processo de administração e auditoria de todos os recursos se torne uma tarefa com alto custo (Ubale Swapnaja et al., 2014; Andaloussi et al., 2018).

Controle de acesso mandatório (MAC, sigla do inglês de *Mandatory Access Control*) (Sandhu, 1993) é um método de autorização baseado em classificação de todas as entidades em um IAM. Nesse modelo, cada usuário e recurso possuem um rótulo de segurança que reflete na sensibilidade das informações a que ele é capaz de acessar ou gerar. De maneira simplória, definem-se níveis de segurança para os recursos e para os usuários, de modo que, para um usuário ter acesso a um determinado recurso, é necessário que seu nível de segurança seja maior ou igual ao nível de segurança do recurso. Ainda que bastante simples, para que atue corretamente, esse modelo necessita de duras restrições para realizar a rotulação das entidades (Sandhu, 1993). Por esse motivo, essa capacidade é permitida para apenas um pequeno conjunto de usuários, culminando em um método reconhecido como um dos mais inflexíveis de autorização. Devido a essa inflexibilidade, o método de autorização MAC é altamente custoso de se implementar e gerenciar, principalmente em ambientes dinâmicos que necessitam de mais flexibilidade. Por exemplo, em uma aplicação de IoT na saúde, é esperado que ocorram emergências em um determinado grupo de usuários (equipe de enfermagem, médicos e outros) e se necessite ter acesso a informações confidenciais de um paciente para estruturar formas de prover uma melhor resposta à emergência. No entanto, caso a aplicação utilize o MAC como forma de autorização, a mudança de rotulação de segurança deve ocorrer de maneira manual e está limitada a um pequeno grupo de usuários. Essa situação pode ocasionar um período no qual a equipe médica é totalmente impossibilitada de acessar tais recursos, o que pode pôr em risco a vida do paciente (Ubale Swapnaja et al., 2014).

Controle de Acesso baseado em papéis (RBAC, sigla do inglês de *Role Based Access Control*) (Ferraiolo et al., 2001) é um dos mais populares métodos de autorização. Nele, é definido um conjunto de papéis e cada papel possui um conjunto de operações e recursos válidos. Assim, cada usuário possui um papel que determina quais operações e recursos ele pode acessar. Usualmente, os papéis são organizados de maneira hierárquica: um papel é capaz de herdar a

capacidade de outros papéis de maneira similar ao MAC (Zhu et al., 2014). Pode-se dizer que no RBAC, as permissões de acesso estão atreladas estaticamente aos papéis definidos e não à identidade individual do usuário. Se a aplicação necessitar de um acesso granular por parte dos usuários, o número de papéis definidos pode crescer rapidamente, o que torna a gerência sobre os papéis potencialmente complexa para os IAMs (Ausanka-Crues, 2001).

Controle de acesso baseado em atributos (ABAC, sigla do inglês de *Attribute Based Access Control*) (Hu et al., 2013) é um método de autorização que, ao invés de papéis específicos, define-se um conjunto de condições (usualmente complexas) que um usuário deve possuir para acessar um determinado recurso. Assim, através de um conjunto de atributos e um conjunto de políticas de acesso, é possível definir o que cada usuário pode ou não acessar. Devido a essa granularidade oferecida, o ABAC é definido como um dos métodos de autenticação mais flexíveis (Coyne e Weil, 2013). No entanto, dado que tais atributos podem vir de diferentes fontes (por exemplo, o acesso depende de atributos de duas identidades distintas em repositórios diferentes) e mudam ao longo do tempo, manter a garantia da confiabilidade e consistência de que ambas as fontes estejam em conformidade com os atributos pode ser uma tarefa desafiadora (Shakarami e Sandhu, 2019).

Na literatura, o ABAC é atualmente um dos modelos mais recentes e promissores de controle de acesso. Sendo assim, diversos padrões e propostas são sugeridos para sua implementação. No entanto, dentre todos eles, o padrão XACML é reconhecido como um dos mais consolidados e, visto que foi desenvolvido explicitamente para se implementar o ABAC, apresenta uma série de sugestões e direções para uma implementação bem-sucedida. Nesse padrão, é definido desde uma forma em que políticas de acesso devem ser escritas até uma arquitetura básica de referência a ser seguida. Em relação a essa arquitetura, o padrão XACML apresenta quatro entidades com funções bem estabelecidas: *Policy Enforcement Point* (PEP), *Policy Decision Point* (PDP), *Policy Information Point* (PIP) e *Policy Administration Point* (PAP). O *Policy Enforcement Point* é o componente responsável por aplicar as decisões de acesso; o *Policy Decision Point* é o componente que avalia as solicitações de acesso em relação às políticas; o *Policy Administration Point* é o componente que atua como um repositório de políticas e oferece facilidades para o gerenciamento das políticas de acesso; por fim, o *Policy Information Point* é a fonte dos atributos necessários para a avaliação da política de acesso (Charaf et al., 2020). A Figura 2.4 ilustra as entidades apresentadas e a troca de mensagens que ocorre entre elas para determinar se um acesso é válido ou não.

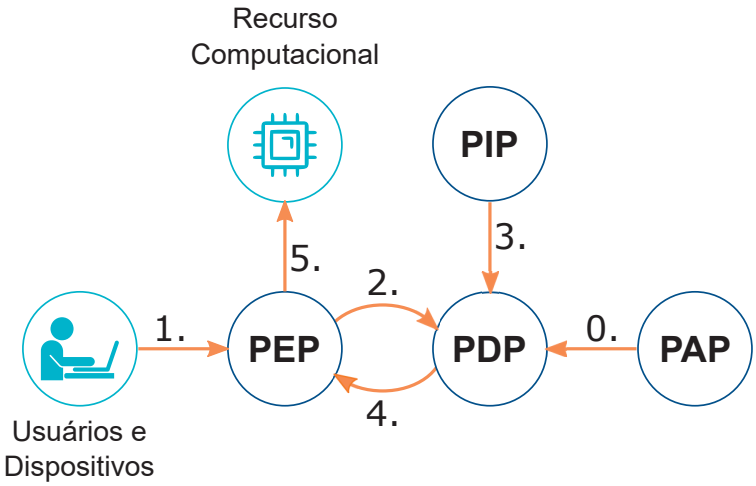


Figura 2.4: Arquitetura XACML

O PAP é a entidade responsável pela autoria e implantação das políticas de acesso nos PDPs (0). Assim , após implantadas as políticas, quando um usuário envia uma requisição de acesso para um recurso computacional (uma operação de leitura em um objeto, por exemplo), o PEP intercepta esse pedido (1). Com esse pedido interceptado, o PEP envia uma requisição de autorização para o PDP, para avaliar se esse acesso deve ou não ocorrer (2). Portanto, para determinar o destino desse acesso, o PDP deve realizar duas tarefas sequenciais: a primeira é encontrar uma política válida para essa situação de acesso (previamente implantada pelo PAP) e, após encontrada uma política, requisitar ao PIP os atributos necessários para avaliá-la (3). A decisão do PDP é encaminhada para o PEP (4), que, por sua vez, garante ou não o acesso ao recurso solicitado pelo usuário (5) (Ravidas et al., 2019).

2.3 MODELOS DE GESTÃO DE IDENTIDADE

Ao longo da história, diversas mudanças tecnológicas e de modelo de negócios impulsionaram evoluções sobre a forma como identidades são gerenciadas. Inicialmente, os IAMs eram modelos para implementar segurança em recursos de maneira completamente isolada. Assim, não havia diferença entre a entidade que provia o recurso e a entidade que fornecia as identidades (Benantar, 2006). No entanto, com o passar do tempo, esse modelo tornou-se datado para algumas aplicações e novos modelos de IAMs emergiram. Nesta seção, definem-se os cinco modelos mais populares de IAM: isolado, centralizado, federado, centrado no usuário e o modelo de identidade autossoberana. Os modelos isolados, centralizados, federados e centrados no usuário são classificados como modelos tradicionais de IAM e sempre apresentam três entidades: usuário, o provedor de identidade (IdP - responsável por armazenar a identidade e oferecer os métodos de autenticação) e o provedor de serviço (SP - responsável por oferecer os recursos). O modelo de identidade autossoberana, por sua vez, elimina essa tradicional arquitetura e dá controle total da identidade para os usuários. Através da tecnologia de registros globais distribuídos (DLT, sigla do inglês para *Distributed Ledger Technology*), esse modelo permite que os próprios usuários armazenem suas identidades e escolham com quais SPs vão compartilhar suas informações. A Figura 2.5 ilustra essa classificação, em que é mostrada toda a evolução dos modelos de IAM ao longo do tempo. Para cada modelo, são apresentadas as suas principais características, benefícios e limitações.

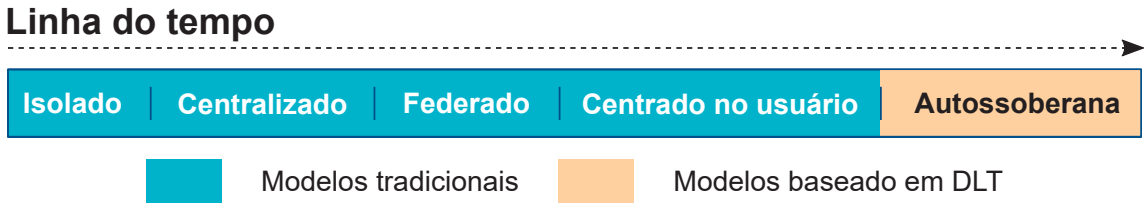


Figura 2.5: Modelos de sistemas de IAMs

Modelo Isolado: esse modelo, também conhecido como modelo silo, é amplamente adotado na Internet devido à sua fácil implementação. Nele, todo SP é também seu próprio IdP, uma vez que o SP possui todas as identidades de seus usuários e esse modelo torna a tarefa de realizar as operações de autenticação, autorização e auditoria bem simples. No entanto, à medida que o número de usuários cresce, esse modelo apresenta sérios problemas de escalabilidade, necessitando de estruturas cada vez mais robustas para suportar novos usuários. Sob o ponto de vista dos usuários, nos modelos isolados não existe uma forma de utilizar a mesma identidade

para acessar recursos diferentes, o que faz com que, para cada recurso que o usuário desejar acessar, seja necessário registrar uma identidade naquele SP (Stobert e Biddle, 2014).

Modelo Centralizado: nesse modelo, a identidade não está confinada apenas a um SP, mas também a uma rede de serviços em que há uma confiança em um IdP para armazenar as identidades e oferecer as operações de autenticação, autorização e auditoria. Assim, pode-se dizer que esse modelo consiste em um único IdP e vários SPs, os quais compartilham entre si as identidades dos usuários. Logo, uma vez que o usuário registra sua identidade, é possível estabelecer o conceito de autenticação única (SSO, sigla do inglês *Single Sign-On*), que permite ao usuário se autenticar uma única vez para obter acesso a todos os SPs, eliminando a necessidade de se autenticar para cada serviço acessado (L'Amrani et al., 2016). A desvantagem desse modelo é a existência do ponto único de falha, pois os atributos dos usuários ficam sob o poder de um único IdP e, caso comprometido, o funcionamento de todo o sistema também será comprometido.

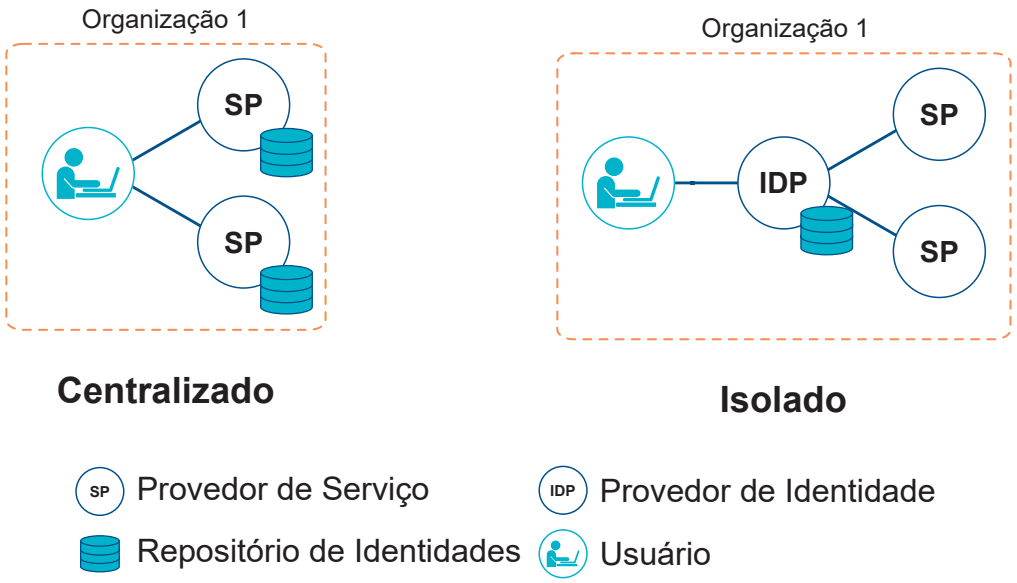


Figura 2.6: Modelo Isolado e Centralizado

Modelo Federado: o conceito de federação representa um relacionamento entre duas ou mais entidades que possuem um sistema IAM (Benantar, 2006). Portanto, no modelo federado, o armazenamento das identidades e os processos de autenticação e autorização são distribuídos entre vários IdPs presentes em vários domínios administrativos, como empresas, governos e outros. Cada domínio possui seu IdP e seus SPs. Ao estabelecer uma federação, os domínios chegam a um acordo, ou seja, estabelecem um padrão que permite a uma identidade emitida por um IdP de um determinado domínio o reconhecimento por SPs de outros domínios. Logo, o usuário não precisa lidar com diversas identidades e processos de autenticação, garantindo o conceito de autenticação única. No entanto, assim como no modelo centralizado, os atributos dos usuários ficam sob o poder dos IdPs (Nida et al., 2014).

Centrado no Usuário: esse modelo é o primeiro em que o usuário detém o controle sobre suas informações. Usualmente implementado com base em algum dos outros modelos descritos acima (sendo o modelo federado o mais comum), o usuário possui um dispositivo que armazena suas identidades e se autentica com diversos IdPs. Assim, esse dispositivo age como um seletor de IdPs, permitindo ao usuário se autenticar com diversos IdPs. Para controlar suas informações, cabe ao dispositivo realizar o trabalho de liberar os atributos de identidade do usuário para os SPs acessados pelo usuário. No entanto, ainda que possua mais controle sobre

sua identidade, os atributos dos usuários ainda permanecem sob o poder dos IdPs (Nida et al., 2014).

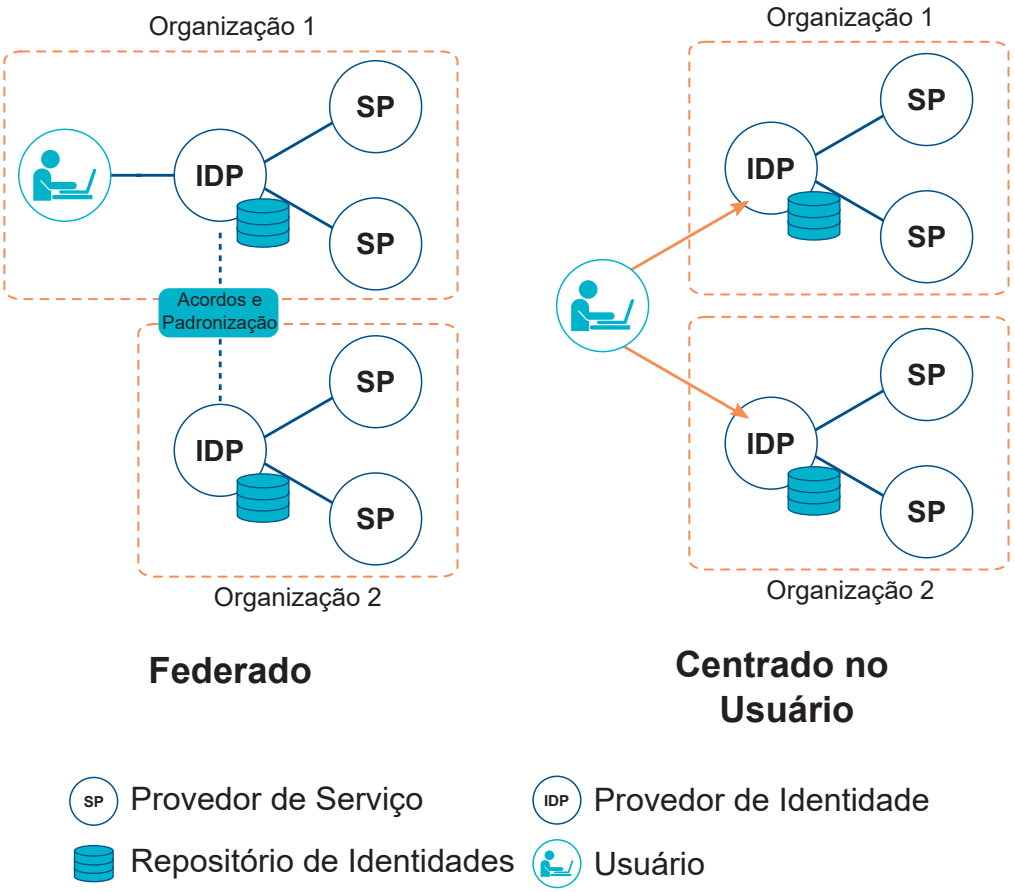


Figura 2.7: Modelo Federado e Centrado no usuário

Modelo com identidade autossoberana: esse modelo com identidade autossoberana refere-se a um sistema em que cada usuário tem total controle sobre seus dados. Isso significa que os usuários podem decidir quando, como e com quem compartilhar suas informações de identidade, sem depender de uma autoridade central para gerenciar ou validar essas ações. A tecnologia de registro distribuído (DLT, sigla do inglês *Distributed Ledger Technology*) é popular nesse contexto por oferecer características como imutabilidade, descentralização e auditoria, mas outros modelos também podem ser utilizados para implementar identidades autossoberanas. O modelo baseado em DLT funciona essencialmente como uma forma de banco de dados digital que é atualizado e mantido de forma independente por todos os membros participantes. Esse tipo de registro permite o armazenamento, distribuição e sincronização de diversos tipos de dados de maneira imutável, eliminando a necessidade de uma autoridade central para gerenciar ou transmitir os dados a todos os membros (Rauchs et al., 2018). Entre as DLTs mais populares utilizadas no modelo de identidade autossoberana, destaca-se a *blockchain*. Nesse sistema, os registros são organizados em blocos, que formam uma lista encadeada de registros. Cada bloco contém três partes principais: (1) os dados armazenados, juntamente com informações como data e hora da transação; (2) os dados do remetente, geralmente representados por uma "assinatura digital", uma técnica criptográfica que assegura a segurança e a integridade das informações; e (3) um identificador único gerado por uma função *hash*, calculado a partir de todas as operações realizadas nesse bloco. Quando um novo bloco é adicionado à *blockchain*, uma

nova operação de *hash* é realizada, combinando o *hash* final do bloco anterior com o *hash* gerado a partir das transações do novo bloco. Esse encadeamento torna todos os registros dependentes uns dos outros (Nakamoto et al., 2008), o que dificulta significativamente a modificação ou reversão de alterações nos registros (Alharby e Van Moorsel, 2017; Lo et al., 2017). Nesse contexto, as informações sobre a identidade são compartilhadas de forma segura utilizando a DLT. No entanto, por razões de segurança, armazenar atributos pessoais diretamente na DLT não é recomendado, pois esses registros são públicos e imutáveis (Tobin e Reed, 2016). Em vez disso, esse modelo emprega métodos como provas de conhecimento-zero (*zero-knowledge proofs*) para compartilhar um conjunto de reivindicações e atestados. Isso permite que uma identidade confirme informações específicas sem revelar os dados subjacentes (Tobin e Reed, 2016).

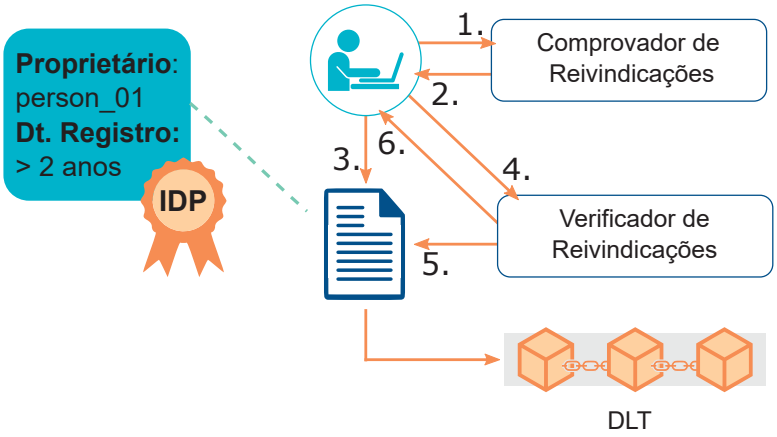


Figura 2.8: Modelo de identidade Autossoberana

A Figura 2.8, ilustra o modelo de IAM com identidade autossoberana. Nesse modelo, o IdP perde a função de armazenar as identidades do usuário e passa a ser apenas um verificador de reivindicações (termo do inglês para *Claim Verifier*). Por exemplo, nesta imagem define-se que para acessar um serviço (o SP nesse modelo é chamado de comprovador de reivindicações, do inglês *Claim Proofer*) é necessário que o usuário possua o $Atributo1 > X$. No entanto, ele só confia em reivindicações assinadas por um determinado verificador de reivindicações parceiro. Logo, o usuário cria uma reinvocação de que possui o $atributo1 > X$ e envia uma prova dessa reivindicação para o verificador de reivindicações (1). O verificador de reivindicações atesta a veracidade de tal informação e assina essa reivindicação, resultando em um atestado de veracidade que é retornado ao usuário (2). Logo, o usuário armazena essa reivindicação na DLT, tornando-a pública e imutável (3). Nesse ponto, ao solicitar o serviço para o comprovador de reivindicações, o usuário aponta para a reivindicação armazenada na DLT (4), na qual se confere o conteúdo da reivindicação e a assinatura (5). Caso esteja conforme os requisitos, o serviço é providenciado (6).

2.4 ANÁLISE ABRANGENTE DA IOT

Embora a IoT ofereça um cenário atraente para o desenvolvimento de novas aplicações e modelos de negócios, é importante abordar, com cautela, os requisitos de segurança que ela apresenta (Abomhara e Køien, 2014). Nesta seção, serão apresentadas algumas aplicações populares para IoT, além de uma arquitetura clássica para IoT que será assumida como base deste trabalho. Também serão discutidos os requisitos não funcionais necessários para lidar com as questões de segurança em aplicações IoT.

2.4.1 Aplicações IoT

A tecnologia IoT criou um mundo hiperconectado em que objetos físicos e virtuais interagem para compartilhar diversos tipos de dados (Tragos et al., 2015). Uma das características mais marcantes da IoT é a duração das interações entre suas entidades, que podem ser breves, por questões de milissegundos, ou durar anos. De maneira geral, as aplicações de IoT são altamente dinâmicas e utilizam interações humano-máquina e máquina-máquina para realizar diversas tarefas de forma automatizada e simplificada (Raza et al., 2021). Por isso, várias organizações de diferentes setores adotam a IoT em seus processos para simplificá-los, melhorá-los e automatizá-los (Said e Masud, 2013).

Casas inteligentes: As casas inteligentes estão cada vez mais integradas com aplicações IoT, o que traz conveniência e eficiência para os moradores (Kastner et al., 2017). Por exemplo, sistemas de iluminação inteligente ajustam automaticamente a intensidade da luz de acordo com a hora do dia ou a presença de pessoas, enquanto termostatos inteligentes aprendem os padrões de consumo de energia e otimizam a temperatura ambiente para conforto e economia (Jabbar et al., 2018). Além disso, dispositivos de segurança, como câmeras e sensores de movimento, monitoram constantemente a casa, garantindo a proteção dos moradores. Para que essas aplicações funcionem de maneira eficiente, é essencial que haja baixa latência na comunicação entre os dispositivos IoT; afinal, são necessárias respostas rápidas e precisas às necessidades do ambiente. Da mesma forma, políticas de acesso gerenciáveis são fundamentais para garantir a segurança dos dados, permitindo que apenas pessoas autorizadas possam interagir e controlar os dispositivos inteligentes ali presentes.

Cidades Inteligentes As cidades inteligentes estão cada vez mais no foco das aplicações IoT para melhorar a qualidade de vida dos cidadãos e otimizar a gestão de recursos urbanos (Syed et al., 2021). Por exemplo, sistemas de transporte inteligentes coletam e analisam dados em tempo real para otimizar o fluxo de tráfego, o que reduz congestionamentos e melhora a eficiência do transporte público. A iluminação pública inteligente se ajusta automaticamente de acordo com as condições ambientais e a presença de pedestres, o que gera energia e aumenta a segurança. Além disso, sensores de qualidade do ar e ruído ajudam a monitorar a poluição e contribuem para a implementação de políticas ambientais mais eficientes. Para que essas aplicações funcionem da maneira correta, é crucial garantir baixa latência na comunicação entre os dispositivos IoT; afinal, cidades são altamente dinâmicas e as respostas das aplicações devem ocorrer no mesmo ritmo. Da mesma forma, políticas de acesso gerenciáveis são essenciais para proteger os acessos aos dados dos cidadãos e garantir a segurança dos dados, pois é um ambiente com uma grande quantidade de dispositivos carregando consigo informações que podem ser valiosas sobre as rotinas das pessoas.

Veículos Inteligentes Os veículos inteligentes estão se tornando cada vez mais comuns e incorporam uma variedade de aplicações IoT para melhorar a segurança, a eficiência e o conforto dos passageiros. Por exemplo, sistemas de assistência ao motorista utilizam sensores e câmeras para monitorar o ambiente ao redor do veículo e fornecer alertas de colisão iminente ou assistência para manobras, como estacionamento automático (Vardhana et al., 2019). Além disso, os veículos conectados se comunicam com a infraestrutura de trânsito para otimizar rotas e evitar congestionamentos, melhorando a eficiência do tráfego (Kim et al., 2017). Os veículos elétricos inteligentes, por sua vez, ajustam o consumo de energia com base nas condições de direção e na necessidade de recarga, garantindo uma autonomia otimizada. Para que essas aplicações funcionem de maneira eficiente, é fundamental garantir baixa latência na comunicação entre os dispositivos IoT e os sistemas de controle do veículo, permitindo respostas rápidas e ações coordenadas em situações críticas de trânsito. Da mesma forma, políticas de acesso gerenciáveis são essenciais para proteger os dados dos usuários.

Aplicações de IoT na área da saúde As aplicações IoT na área da saúde revolucionaram o atendimento médico e a gestão de pacientes, melhorando a eficiência e a qualidade dos cuidados prestados. Por exemplo, dispositivos vestíveis, como *smartwatches* e monitores de glicose contínua, coletam dados em tempo real sobre a saúde dos pacientes, permitindo um monitoramento constante e alertas imediatos em casos de anormalidades (Dhanvijay e Patil, 2019). Além disso, sistemas de telemedicina conectam pacientes e profissionais de saúde remotamente, facilitando consultas e acompanhamento médico, independentemente da localização geográfica. Ainda, a gestão de medicamentos inteligente garante a administração correta de medicamentos em hospitais e instituições de saúde, reduzindo erros e melhorando a segurança do paciente. Para que essas aplicações funcionem de maneira eficiente, é crucial garantir baixa latência na comunicação entre os dispositivos IoT e os sistemas de gerenciamento de saúde, permitindo respostas rápidas e intervenções em tempo hábil em situações críticas. Da mesma forma, políticas de acesso gerenciáveis são fundamentais para proteger o acesso dos dados dos pacientes e garantir a segurança dos dados médicos, assegurando que apenas profissionais de saúde autorizados possam acessar e controlar as informações geradas pelos dispositivos IoT na área da saúde.

Industrias Inteligentes As indústrias inteligentes estão adotando aplicações IoT para otimizar processos de produção, aumentar a eficiência e reduzir custos operacionais. Por exemplo, sensores e dispositivos conectados são usados para monitorar em tempo real o desempenho e o estado de máquinas e equipamentos, possibilitando a manutenção preditiva e a redução do tempo de inatividade. Além disso, sistemas de automação avançados e robótica colaborativa permitem uma maior integração entre humanos e máquinas, aumentando a produtividade e a segurança no ambiente de trabalho (Chen et al., 2017; de Oliveira e Simões, 2017). A rastreabilidade e o controle de qualidade também são aprimorados por meio de etiquetas inteligentes e sistemas de inspeção automatizados, garantindo um alto padrão de qualidade dos produtos fabricados. Para que essas aplicações funcionem de maneira eficiente, é essencial garantir baixa latência na comunicação entre os dispositivos IoT e os sistemas de gerenciamento industrial, permitindo respostas rápidas e ações coordenadas em situações dinâmicas de produção. Da mesma forma, políticas de acesso gerenciáveis são cruciais para proteger a propriedade intelectual e garantir a segurança das informações, assegurando que apenas entidades autorizadas, como funcionários e prestadores de serviços, possam acessar e controlar os dados gerados pelos dispositivos IoT nas indústrias inteligentes.

2.4.2 Arquitetura básica de IoT

Como discutido anteriormente, a ideia principal da IoT consiste em ter vários objetos trocando informações entre si, coletando, transferindo e acessando dados. Portanto, na literatura, é possível encontrar várias propostas de arquiteturas para IoT. Geralmente, as propostas dividem a IoT em camadas, sendo que algumas camadas são bastante padronizadas entre elas. Na maioria dos casos, a camada superior é chamada de camada de aplicação, e a camada inferior é chamada de camada de objetos. A camada intermediária é a mais variada e às vezes pode ser dividida em duas ou mais subcamadas. A estrutura básica de uma arquitetura IoT é apresentada na Figura 2.9. Ela contém três camadas principais: a camada de objetos (inferior), uma ou mais camadas intermediárias e a camada de aplicação (superior). A seguir, estão as principais funcionalidades e recursos de cada camada.

Camada de Objetos: Na literatura, essa camada tem muitas nomenclaturas alternativas. Comumente, pode ser referida como a camada de percepção ou de hardware. No entanto, independentemente da terminologia, o objetivo dessa camada acaba sendo coletar diferentes dados do ambiente circundante, como localização, umidade, temperatura, movimento, velocidade e direção. Portanto, essa camada é composta principalmente por sensores que coletam dados

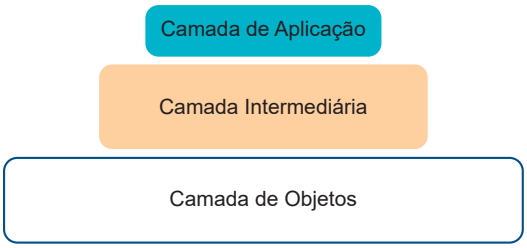


Figura 2.9: Arquitetura básica de IoT

do ambiente próximo e os enviam pela Internet. Algumas propostas assumem que essa camada é construída usando apenas sensores sem fio. No entanto, a maioria dos trabalhos recentes pressupõe que essa camada é composta de sensores e outros dispositivos, como atuadores, *tags* RFID e algumas redes de dispositivos (por exemplo, câmeras e telefones celulares). Uma das características mais significativas dessa camada é a presença de um grande conjunto de objetos físicos heterogêneos com diferentes condições de operação, capacidades e limitações.

Camada Intermediária: Como a camada de objetos contém muitos dispositivos heterogêneos, alguns deles não podem enviar os dados coletados diretamente pela Internet para a aplicação. Assim, os dados coletados devem ser primeiro manipulados, estruturados e enviados por outro dispositivo da camada intermediária com os recursos adequados para entregá-los corretamente. Portanto, o principal objetivo dessa camada é transmitir os dados produzidos na camada de objetos para o destino através da Internet. Embora essa camada possa ser chamada de camada de transmissão ou *gateway*, sua funcionalidade foi expandida nos dias de hoje. Além de coletar/enviar dados, vários trabalhos usam essa camada para registrar eventos e gerenciar dispositivos da camada de objetos. Normalmente, os dispositivos dessa camada são roteadores, *switches* e outros equipados com várias tecnologias de comunicação que vão desde *Bluetooth* até links via satélite.

Camada de Aplicação: É o destino final dos dados coletados. É a camada superior da arquitetura do IoT e é responsável por fornecer serviços ao usuário final e funcionalidades do sistema. Geralmente hospedada em servidores de nuvem, a camada de aplicação oferece uma maneira de controlar e se comunicar remotamente com todas as entidades do IoT conectadas e exibir os dados coletados em modelos, gráficos e fluxogramas.

2.4.3 Requisitos de gerais de Segurança e Integridade em IoT

Cada entidade presente na IoT é um ponto crítico que pode ser explorado para realizar vazamentos de dados e ataques maliciosos. Portanto, um dos principais desafios da IoT é garantir a segurança e a integridade em todos os milhares de acessos que podem ocorrer. A seguir, é apresentada uma lista de requisitos de segurança e integridade que qualquer IAM deve cumprir para garantir o sucesso das aplicações IoT em relação à segurança e à integridade de seus usuários e dispositivos.

Confidencialidade: A confidencialidade é um requisito de segurança em IoT que deve ser atendido, mas não é necessariamente obrigatório em cenários em que os dados utilizados são públicos. No entanto, ao observar as aplicações mais promissoras em IoT, nota-se que elas carregam dados sensíveis que não devem ser divulgados ou lidos por entidades não autorizadas na maioria das situações. Dados de pacientes em aplicações de saúde IoT, a rotina de uma pessoa em uma cidade inteligente e dados empresariais privados, por exemplo, são extremamente sensíveis e devem ser protegidos contra todo acesso não autorizado (Whitmore et al., 2015;

Balte et al., 2015). Sob esse contexto, a Autenticação e a Autorização são duas operações imprescindíveis para garantir a segurança das aplicações de IoT. De forma resumida, essas duas operações são responsáveis por garantir o acesso apenas de entidades identificadas e autorizadas a alguns recursos da IoT. Além disso, para tornar as aplicações ainda mais confidenciais, algumas aplicações precisam de um sistema adaptativo e robusto que use informações contextuais para aprimorar as operações de autenticação e autorização, permitindo o acesso somente em situações bem específicas (Kim e Lee, 2017).

Disponibilidade: O conceito de IoT gira em torno de um mundo onde tudo está conectado. Portanto, as aplicações de IoT precisam que todos os dados estejam disponíveis sempre que necessário e esses dados devem ser protegidos por mecanismos de segurança, ou seja, todas as entidades e dados de IoT devem ser acessíveis e disponíveis quando necessário, devem ser protegidos por mecanismos de segurança e proteção, e com latência mínima para não afetar os requisitos de desempenho das aplicações de IoT (Joshitta e Arockiam, 2016).

Integridade: A integridade é um dos requisitos mais críticos para várias aplicações de IoT. Embora esse requisito possa variar de acordo com a aplicação, um único dado inconsistente (por exemplo, dados de autorização não atualizados) ou a perda ou manipulação de qualquer dado enviado ou recebido por entidades de IoT podem potencialmente causar danos massivos a empresas e a pessoas (Joshitta e Arockiam, 2016).

Auditoria: Embora a auditoria em si não possa impedir ataques e garantir acesso seguro, ela é útil para aprimorar outras técnicas de segurança. Esse requisito cria a necessidade de registrar cada acesso que ocorre dentro de uma aplicação de IoT. Assim, quando coletamos dados com detalhes de cada acesso, essas informações podem fornecer *insights* para planejar novas políticas de segurança de rede. Em alguns casos, a auditoria também pode ajudar a verificar quando ocorre um incidente, mostrando o que aconteceu e quem foi o responsável (Whitmore et al., 2015).

2.5 DESAFIOS DA IOT NOS COMPONENTES DAS IAMS

Ao longo da última década, presenciamos o surgimento e o crescimento dos sistemas IAM. Diversos métodos de autenticação e autorização foram desenvolvidos pelos pesquisadores, o ciclo de vida das identidades progrediu até um patamar adequado naquele período, e outros obstáculos relacionados foram vencidos. Embora o panorama dos IAMs tenha se mantido praticamente inalterado na academia e no mercado, atualmente observamos que o surgimento e a expansão da IoT impulsionam os sistemas IAM a enfrentarem novos desafios, ao apresentar um conjunto totalmente inédito de questões. De fato, esse novo contexto estimulou pesquisas em diversos componentes da IAM e várias propostas surgiram para torná-los compatíveis com a IoT. É importante notar que, com o advento da IoT, diversas empresas e diversos pesquisadores levantaram as limitações dos atuais sistemas de IAM. Dentre elas, a principal limitação a ser superada pelos IAMs contemporâneos é a escalabilidade no gerenciamento de identidades. Em suma, os sistemas IAM convencionais foram concebidos para impor políticas de controle de acesso e gerenciar pessoas que utilizam serviços online, aplicativos e informações. Entretanto, esses sistemas não conseguem lidar com bilhões de entidades da IoT, suas respectivas identidades e os variados padrões de comunicação entre outras entidades, como dispositivos, indivíduos e aplicativos na rede. Portanto, o futuro dos IAMs depende, em primeiro lugar, da capacidade de escalar para o número elevado de identidades. No entanto, conforme apontado pelas pesquisas, lidar apenas com a escalabilidade não é suficiente para a IoT. A explosão de entidades de IoT também aumentou significativamente o volume e a complexidade dos acessos. Hoje, a IoT interconecta dispositivos, usuários, sistemas e plataformas em uma vasta rede, e o acesso vem

de diferentes lugares e entidades. Como resultado, a falta de opções de controle de acesso, a infraestrutura legada e as dependências de protocolos desatualizados são um enorme desafio no cenário atual de IAMs. Em resumo, algumas das oportunidades para a pesquisa em IAMs para IoT são descritas nas seguintes subseções.

2.5.1 Registro das Identidades

Nas aplicações de IoT, presume-se que os usuários normalmente possuem vários dispositivos de IoT com eles. Portanto, criar identidades digitais para cada dispositivo de IoT e atribuir as permissões corretas a essas identidades é uma tarefa complicada. Assim, um mecanismo eficaz de provisionamento de identidade é indispensável para os IAMs desenvolvidos para IoT. Atualmente, a maioria das propostas adota o provisionamento manual de identidades, em que um usuário humano identifica todos os seus dispositivos IoT no IAM antes de usá-los. Embora isso possa ser conveniente para pequenas aplicações, como casas inteligentes, esse provisionamento manual não é adequado se considerarmos aplicações maiores, como cidades inteligentes. Por exemplo, em Kumar et al. (2019), os autores propõem uma cidade inteligente com o conceito BYOD (*Bring Your Own Device*), permitindo que o usuário carregue qualquer dispositivo IoT e o conecte à rede da cidade. Nessa proposta, o autor argumenta que, embora o BYOD traga flexibilidade e praticidade para as aplicações de IoT, ele também traz um alto risco de ataques e acesso não autorizado na rede. Assim, para minimizar essa ameaça, todos os dispositivos devem ser identificados e estar fora de uma lista de negação. Se o usuário tiver apenas alguns dispositivos, o registro explícito e manual de seus dispositivos no IAM pode funcionar de forma direta. No entanto, se o número de dispositivos IoT aumentar, o registro direto e explícito se tornará uma tarefa ineficiente e custosa, o que afasta o interesse do usuário na aplicação.

2.5.2 Propagação, manutenção e remoção das identidades

À medida que as aplicações IoT se tornam maiores, o armazenamento, a manutenção e o cancelamento das identidades das entidades IoT se tornam mais complexos. Tradicionalmente, devido ao crescente número de usuários, dispositivos e aplicações, os dados de identidade de todas as entidades IoT podem ser mantidos em vários sistemas diferentes. Esses sistemas podem ser referidos como repositórios de identidade, e cada repositório pode conter diferentes tipos de dados de identidade. Por exemplo, alguns repositórios de identidade podem conter identidades humanas “tradicionais"dos sistemas de IAM legado, enquanto outros podem conter identidades de diversos dispositivos IoT. Alguns repositórios de identidade podem estar armazenados na nuvem, enquanto outros podem ser encontrados em dispositivos locais de armazenamento, como nós de *fog computing*. Independentemente disso, as diversas fontes de informações de identidade trazem uma série de desafios para obter eficientemente os dados das entidades em aplicações de IoT e manter a consistência desses dados em todos os diferentes repositórios que uma aplicação IoT pode possuir.

Para exemplificar, vamos supor uma aplicação de saúde sensível ao tempo, como monitoramento de eletrocardiograma (ECG) e eletroencefalografia (EEG), que requer autenticação e autorização contínuas de todos os pacientes, dispositivos de IoT de saúde e equipes médicas. Vamos assumir que as identidades dos pacientes e dispositivos de IoT são armazenados em um repositório de identidade específico da empresa de saúde, enquanto as equipes médicas estão em outro repositório de identidade localizado em outro local. Para esse caso, para fornecer as autenticações e autorizações necessárias, o IAM deve recuperar esses dados de identidade dos

dois repositórios e, sem um mecanismo de sincronização adequado, há a possibilidade de que os dados de identidade não sejam consistentes entre eles, podendo resultar em acessos indesejados.

Agora, aumentando esse cenário, como o número de pacientes, dispositivos e equipes médicas pode se multiplicar com a proliferação de aplicações de saúde, vamos adicionar mais complexidade e supor que esses repositórios de identidade foram migrados e estão na nuvem. Para esse caso, o IAM pode sofrer problemas de latência, o que impacta atividades fundamentais, como autenticações, comparações de atributos (por exemplo, determinar se um funcionário médico tem um determinado valor de atributo em sua identidade) e alterações de atributos (por exemplo, um usuário realizando modificações em um de seus atributos).

Além disso, como tantas informações estão espalhadas em diferentes armazenamentos de identidade, vamos supor que um membro da equipe médica possa deixar a organização, dispositivos IoT possam se tornar obsoletos e pacientes possam sair. Nesse caso, como tantas informações estão espalhadas em diferentes armazenamentos de identidade, a tarefa de garantir que todas essas identidades sejam excluídas ou alteradas é difícil e trabalhosa. Uma única identidade não revogada pode acessar informações confidenciais e aplicações, levando a vazamentos substanciais de dados e riscos para todos os usuários da aplicação.

2.5.3 Métodos de Autenticação

Métodos robustos de autenticação são necessários para estabelecer confiança entre as entidades de IoT conectadas. Sem eles, é impossível identificar uma entidade de IoT e, consequentemente, dar-lhe o acesso e as permissões adequadas para fazer o que precisa ser feito. No entanto, a autenticação é considerada uma das tarefas mais desafiadoras da IoT devido à grande variedade, volume e configurações de dispositivos IoT na rede. Em grande parte, a IoT é composta por dispositivos menores com capacidade de memória e microprocessadores limitados, o que restringe grande parte dos métodos de autenticação tradicionais e mais complexos. Por exemplo, a sobrecarga de comunicação de um método de autenticação é um fator crucial para a IoT, especialmente quando a aplicação espera dispositivos pequenos e simples (Ferrag et al., 2017); o número de mensagens trocadas deve ser mantido o mais baixo possível, e o tamanho das mensagens deve ser o menor possível devido à largura de banda restrita da tecnologia de comunicação sem fio. Adicionalmente, na maioria dos dispositivos IoT, a autenticação convencional baseada em senhas ou frases secretas não é viável, devido à ausência de interfaces ou interações humanas em grande parte desses aparelhos. Por isso, a tendência atual é utilizar o fator de posse para a IoT, o que faz com que a autenticação aconteça por meio de chaves embutidas nos dispositivos IoT e seus respectivos certificados PKI exclusivos.

Para fortalecer a segurança geral da autenticação para IoT, outra tendência atual é o uso do paradigma de autenticação contínua. Compatível com outros fatores de autenticação, esse paradigma tem como objetivo autenticar a legitimidade de uma entidade de IoT analisando seu perfil comportamental durante um acesso. Assim, por meio de uma análise contínua e em tempo real, a autenticação contínua atua como uma autenticação complementar que traduz o comportamento da entidade de IoT em pontuações de autenticação e pode exigir outro fator para comprovar sua identidade. Assim, após a autenticação tradicional (senhas, chaves integradas e certificados PKI), a autenticação contínua garante a autenticação da entidade IoT ao longo de todo o acesso. Desse modo, se um acesso estiver seguindo um caminho inédito (por exemplo, um usuário nunca acessou determinada informação crítica em um horário específico), a autenticação contínua detecta essa alteração e pode ser solicitado um fator adicional de autenticação para comprovar sua legitimidade.

2.5.4 Métodos de Autorização

Os métodos de autorização para IoT são amplamente estudados na literatura. De fato, diversas abordagens e estruturas de autorização abordam aplicações e desafios específicos da IoT. Em casos excepcionais, nos quais a aplicação IoT lida com um número pequeno de dispositivos conectados e não requer controle de autorização centralizado, como algumas aplicações de casas inteligentes, pode-se encontrar propostas que utilizam métodos simples, como a matriz de autorização. Geralmente, para essas situações, os autores adotam uma abordagem isolada para o IAM e estabelecem que o usuário proprietário do dispositivo IoT define os direitos de acesso dentro do próprio aparelho, tornando fácil a implementação e a utilização desse método. No entanto, embora eficaz para esses casos, apenas algumas aplicações não exigem controle de autorização centralizado, restringindo o uso das matrizes de autorização. Para aplicações que envolvem o uso de controle de autorização centralizado, a maioria das propostas na literatura acaba adaptando os métodos de autorização clássicos, como DAC, MAC, RBAC e ABAC. Embora a flexibilidade do DAC seja boa para IoT, não é restrita o suficiente para impor políticas de acesso, uma vez que a passagem de informações de um objeto para outro não é restrita, tornando-o inviável para várias aplicações. Por outro lado, o MAC restringe as políticas de acesso atribuindo aos sujeitos e aos objetos um rótulo de segurança, o que é inflexível para várias aplicações de IoT. O modelo RBAC é o segundo método de controle de acesso mais comumente utilizado para IAMs para IoT. Com ele, um administrador gera um grupo de funções, que expressam tarefas específicas e atribui uma lista de permissões a essas funções. Embora possa parecer natural determinar quais pessoas podem acessar quais serviços, com a introdução de IoT, definir todas as funções em todos os casos de uso é uma tarefa desafiadora e exaustiva, tornando o RBAC não tão adequado, pois carece de escalabilidade e flexibilidade.

Tabela 2.1: Métodos de autorização utilizados pela literatura

	Método		Método
Jindou et al. (2012)	RBAC	Lee et al. (2017)	DAC
Kim et al. (2012)	ABAC	Alshehri e Sandhu (2017)	ABAC / RBAC
Seitz et al. (2013)	ABAC	Ouaddah et al. (2017)	RBAC
Mahalle et al. (2013)	DAC	Tian et al. (2017)	DAC
Neisse et al. (2014)	ABAC	Islam et al. (2018)	ABAC
Salonikias et al. (2015)	ABAC	Sciancalepore et al. (2018)	ABAC
Barka et al. (2015)	RBAC	Bhatt e Sandhu (2020)	ABAC
Alshehri e Sandhu (2016)	ABAC	Bhatt et al. (2021)	ABAC
Ray et al. (2017)	ABAC	Ding e Ma (2021)	ABAC

O ABAC é definido como um dos métodos de autorização mais adequado para lidar com IoT (Ravidas et al., 2019). Em resumo, o ABAC pode lidar com um ambiente de controle de acesso complexo e mutável com políticas de acesso detalhadas expressas em termos dos atributos das entidades de IoT e do contexto. Consequentemente, estamos testemunhando a ampla adoção do ABAC, conforme apresentado na Tabela 2.1, com várias aplicações de IoT, implementando-as do zero ou migrando de outros modelos, como o RBAC, para o ABAC. No entanto, a natureza do IoT traz vários desafios em relação à sua implementação e ao seu gerenciamento (Ravidas et al., 2019). Ao ser construída do zero, uma quantidade considerável de políticas complexas para diversos casos de uso precisa ser formulada e gerenciada durante toda a vida útil da aplicação IoT. Caso sejam migradas de outros modelos, o administrador de rede deve converter várias políticas de acesso para o ABAC, um processo custoso e demorado. Dessa forma, independentemente da

origem das políticas de acesso, o desenvolvimento manual de políticas de ABAC é complexo e sujeito a erros humanos, resultando em acessos não autorizados e em vazamento de dados. Além disso, como as aplicações de IoT geralmente estão em constante evolução e mudança, a elaboração de políticas de acesso complexas torna-se uma tarefa infindável para os administradores de rede.

2.6 RESUMO

Neste capítulo, apresentamos em duas partes os conceitos básicos e necessários para entender o trabalho desenvolvido nos próximos capítulos. Na primeira parte, discutimos nas Seções 2.1, 2.2 e 2.3 os fundamentos da gestão de identidade e acesso, agrupados em quatro tópicos principais: (i) definimos o conceito geral de identidades digitais e sua importância; (ii) descrevemos o ciclo de vida das identidades na gestão de identidade e acesso - provisionamento, propagação, utilização, manutenção e desprovisionamento; (iii) abordamos os principais mecanismos de autenticação e autorização; e (iv) explicamos diferentes modelos de gestão de identidade que podem ser adotados para garantir a segurança e controle de acesso aos recursos. Adicionalmente, revisamos estudos anteriores que fornecem contexto e motivação para a análise de aplicações IoT e seu impacto na gestão de identidades.

Na segunda parte, apresentada nas Seções 2.4 e 2.5, discutimos estudos sobre a análise abrangente da IoT, abordando conceitos, tecnologias e aplicações promissoras em diversos setores da sociedade, como saúde, transporte e indústria. Observamos que estudos do estado da arte se concentraram principalmente em identificar as características dessas aplicações, bem como suas arquiteturas e requisitos fundamentais. Além disso, analisamos a literatura relacionada aos desafios da IoT nos sistemas IAM, discutindo como a IoT torna os requisitos de segurança mais desafiadores e quais são os desafios dessas mudanças para os componentes dos IAMs. Ainda que esta tese não tenha atacado todos os problemas apresentados da seção 2.5, este capítulo apresenta todos os fundamentos necessários para compreender o problema do desempenho na recuperação de atributos - abordado no Capítulo 4 e o problema de gerenciamento das políticas de acesso - abordado no Capítulo 3. Logo, esses aspectos discutidos neste capítulo fornecem a base para entender como a IoT impactou esses componentes e as propostas do mecanismo de mineração de políticas e distribuição de atributos (Capítulos 3 e 4).

3 GERENCIAMENTO AUTOMATIZADO DE POLÍTICAS DE ACESSO

Este capítulo aborda o OE1, um método de mineração de políticas, que se utiliza de registros de acesso e de aprendizado de máquina para extrair políticas de acesso precisas, sucintas e auditáveis. É importante notar que a mineração de políticas já é reconhecida como um mecanismo candidato popular para aliviar a complexidade do gerenciamento de políticas de acesso em IoT. No entanto, grande parte das técnicas oferecidas atualmente são uma "caixa-preta" não auditável, o que torna impossível para um ser humano verificar a corretude de uma regra. Consequentemente, essa característica contribui para que acessos indevidos ocorram e passem despercebidos pelo sistema. Intuitivamente, soluções que oferecem regras auditáveis e legíveis para humanos resolveriam esse problema. Porém, devido às técnicas utilizadas, poucas oferecem essa característica.

Dado os problemas mencionados anteriormente, pretendemos responder às seguintes perguntas neste capítulo:

- Quais são as técnicas utilizadas pela literatura para oferecer um método de mineração de políticas e por que elas não entregam um resultado satisfatório em relação à precisão e auditabilidade?
- Quais técnicas podem ser empregadas e como elas podem ser desenvolvidas para oferecer um mecanismo de mineração de políticas capaz de gerar regras de acesso precisas, sucintas e auditáveis?

A primeira questão deste capítulo é abordada na Seção 3.1, onde discutimos trabalhos relacionados à mineração de políticas de acesso, fornecendo uma visão geral de estudos anteriores e suas limitações. A segunda questão é tratada a partir da Seção 3.2, onde formalizamos o problema de mineração de políticas, destacando os desafios enfrentados e as capacidades que um mecanismo de mineração deve possuir. Na Seção 3.3, descrevemos os detalhes do mecanismo de mineração proposto. Na Seção 3.4, apresentamos a metodologia de avaliação e os resultados do mecanismo por meio de uma base de dados sintética e outra base de dados real. Por último, na Seção 3.5, enfatizaremos a importância dos resultados obtidos e faremos um resumo dos pontos discutidos neste capítulo.

3.1 TRABALHOS RELACIONADOS

Devido à escala massiva de políticas de acesso e ao alto número de entidades envolvidas nas aplicações IoT, listar todas as regras e políticas de acesso para determinar se um acesso deve ou não ser válido é uma tarefa quase impossível. Sendo assim, para facilitar a implementação e a manutenção dessas políticas de acesso, vários trabalhos da literatura se esforçam ao utilizar métodos de aprendizado de máquina para aliviar esse processo. De maneira geral, em todos os trabalhos, os autores treinam um modelo de aprendizado de máquina através de registros de acesso, históricos de acesso e comportamentos e utilizam esse modelo para tomar decisões.

No trabalho de Xu e Stoller (2014), os autores propuseram um método de mineração de políticas de acesso que possui como entrada um *log* de acesso contendo um conjunto de identificadores de usuários, os identificadores dos recursos acessados, as operações que foram realizadas naqueles recursos e um conjunto de atributos de cada usuário. Com base nesse *log*, os autores inicialmente geram uma *Access Control List* (ACL), que é composta por diversas tuplas

de usuário-permissão. De maneira resumida, cada tupla dessa lista indica que um identificador de usuário possui direito de realizar uma operação específica em um determinado recurso. Os autores apontam que essa lista já pode ser utilizada para criar políticas de acesso para o mecanismo de controle de acesso, gerando uma política trivial para cada tupla associando o identificador do usuário a um determinado recurso. Ainda que o conjunto de políticas criadas a partir do ACL cubra todas as situações de acesso presentes no *log*, as políticas resultantes são extremamente granulares e com generalização mínima, uma vez que são utilizadas apenas para determinar que um identificador de usuário possui permissão para acessar um determinado recurso. Para lidar com esse problema, os autores oferecem um método que generaliza as tuplas da ACL com o intuito de criar políticas candidatas que sejam capazes de generalizar e cobrir um maior número de usuários. Assim, o método proposto pega cada regra gerada e substitui o identificador do usuário por seus atributos. O objetivo dos autores é aumentar o processo de generalização da política e aumentar sua cobertura de forma que diversas tuplas da ACL possam ser satisfeitas por essa regra. Logo, quando se substitui os identificadores por seus atributos, as regras passam a ficar em função dos atributos e, consequentemente, mais generalizadas. Após definir políticas em relação aos atributos, o método proposto inicia um processo de união das políticas, excluindo políticas redundantes que já são cobertas por uma outra política e unindo as políticas únicas. Ainda que efetivo no processo de mineração de políticas de acesso, a abordagem apresentada é altamente baseada em uma heurística e gera regras altamente complicadas de se interpretar. Neste trabalho, os autores afirmam que seu método possui uma complexidade algorítmica cúbica em relação ao número de usuários únicos, o que impede sua execução em *log* com uma grande quantidade de usuários. Além disso, o processo de mineração apresentado foi testado exclusivamente em dados sintéticos, estipulados pelo próprio autor.

O trabalho de Medvet et al. (2015) foi inspirado no trabalho de Xu e Stoller (2014). Nele, os autores propuseram um método evolutivo para lidar com o problema de geração de políticas de acesso no mecanismo de controle de acesso. Neste trabalho, foi utilizada uma abordagem de dividir e conquistar pela qual a política de acesso é construída de maneira incremental. Em um primeiro momento, os autores seguiram de forma bem similar o método de (Xu e Stoller, 2014), pois realizaram um processo de transformação de um log de acesso em uma ACL trivial. Após a construção dessa ACL, o trabalho entra em um processo de repetição em que, em cada iteração, um conjunto de políticas aleatórias com base nos atributos dos usuários e recursos é criado. Para cada política, avalia-se o número de políticas triviais para as quais ela possui cobertura. Com todas as políticas avaliadas, o método seleciona a que cobre o maior número de casos e retira as políticas triviais cobertas da ACL. Repetido esse processo inúmeras vezes, o método finaliza no momento em que a ACL está completamente vazia. Como este trabalho seguiu a metodologia de avaliação definida por Xu e Stoller (2014), o método apresentado neste artigo também foi testado exclusivamente em dados sintéticos, o que, segundo os próprios autores, não é o ideal para o contexto de mineração de políticas.

O método apresentado por Iyer e Masoumzadeh (2018) se difere por ser capaz de minerar não só as políticas de acesso tradicionais que permitem o acesso, mas também políticas de acesso negativas de autorização, capazes de negar o acesso de usuários aos recursos de uma organização ou empresa. Para minerar esses dois tipos de políticas de acesso, os autores, ao invés de criar um algoritmo de extração de políticas, optaram por utilizar e estender um algoritmo de mineração de dados chamado *Probability Risk and Impact System* (PRISM). O algoritmo PRISM é um algoritmo iterativo que possui uma abordagem separar-e-conquistar para a obtenção de políticas corretas. Inicialmente, o algoritmo seleciona uma determinada classe da base de dados (classe dos acessos aceitos, por exemplo) e inicia uma regra vazia. Em seguida, ele avalia qual valor de um determinado atributo possui maior probabilidade de ocorrência naquela classe e

adiciona aquele atributo na política. Feito isso, ele divide essa base de dados em duas, uma base que possui todas as entradas com o valor selecionado e uma base que não possui. Para a base que possui o valor selecionado, ele repete o processo de forma a refinar a política criada e selecionar o atributo de maior probabilidade. Após repetir esse processo inúmeras vezes, em um determinado momento, não haverá classes diferentes na regra e o processo se encerrará. Para a base que não possui o valor selecionado, ele cria outra regra vazia e reinicia todo esse processo. Ainda que esse trabalho apresente regras "corretas" tanto para as políticas de acesso positivas quanto para as negativas, esse trabalho tende a gerar políticas especializadas (*overfitting*) para os dados de treinamento e não oferecer políticas que autorizem corretamente o conjunto independente de acessos para o teste.

3.2 PROBLEMA DE MINERAÇÃO DE POLÍTICAS

O ABAC é uma estratégia de autorização que estabelece permissões com base nos atributos de usuários e recursos. Sendo assim, uma política de controle de acesso é um conjunto de regras em que, para cada regra, há um conjunto de condições baseadas nos valores de atributos, permitindo ou não um determinado acesso (Hu et al., 2013). Para que um acesso ocorra, deve haver uma solicitação de acesso que demonstre o desejo de um usuário em realizar uma ação em um determinado recurso. Para que essa solicitação seja aceita, o usuário deve fazer uma solicitação de acesso e se identificar no sistema. Após a identificação do usuário, os valores de seus atributos e dos recursos são analisados em cima de uma política de acesso. Se os atributos dos usuários e recursos satisfizerem todas as condições de pelo menos uma regra da política de acesso, o acesso é permitido; caso contrário, o acesso é negado (Hu et al., 2013).

Neste trabalho, define-se que todas as políticas de acesso devem ser auditadas e, devido a esse fator, políticas de acesso extremamente grandes e complexas são indesejáveis. Além disso, devido a mudanças organizacionais ou até mesmo a erros humanos, a implementação e a manutenção dessas políticas de acesso podem ser imprecisas. Essas imprecisões podem resultar em dois tipos de erros: autorizações incorretas, nas quais um usuário, que não deveria ter acesso, conseguiu satisfazer pelo menos uma regra da política de acesso; e negações incorretas, nas quais um usuário, que deveria ter acesso, não consegue satisfazer nenhuma regra da política de acesso implementada. Tipicamente, a autorização incorreta (geralmente definida como precisão de autorização) é a mais problemática, já que pode ser usada como uma porta de entrada para vários ataques de segurança, como invasões de privacidade e escalonamento de privilégios. Uma negação incorreta (geralmente definida como revogação de autorização), normalmente, não apresenta um grande risco de segurança, já que o usuário pode simplesmente relatar seu erro ao administrador de recursos, que, por sua vez, pode corrigir ou adicionar uma exceção à política de acesso. No entanto, mesmo que não tenha um resultado problemático, essa manutenção de políticas e criação de exceções pode levar a mais erros humanos e políticas de acesso complexas com baixa auditoria (Cotrini et al., 2018).

Portanto, um registro de acesso contém informações sobre todas as solicitações de acesso feitas ao sistema durante um determinado período de tempo. Essas informações podem ser processadas para gerar estatísticas de acesso e caracterizar as suas ocorrências. Geralmente, cada linha desse registro é formada pela junção de uma decisão de acesso (acesso válido e inválido) com todos os atributos que influenciaram ou não essa decisão. Assim, a mineração de políticas consiste em determinar uma política de acesso para um registro de acesso, de modo que, para cada linha presente nesse registro, a política de acesso seja capaz de fornecer uma decisão de acesso correta. Portanto, é necessário identificar e definir um conjunto de valores de atributos que influenciam uma decisão de acesso. Assim, espera-se que, para cada registro, a política de

acesso resulte em acessos corretos. Como era de se esperar, projetar tais políticas de acesso com todas as possíveis combinações de atributos presentes no registro de acesso pode ser uma tarefa complexa. De acordo com Xu e Stoller (2014), esse problema pertence à classe NP-difícil, sendo uma variação do problema *Edge RMP* (Lu et al., 2008).

3.3 PROPOSTA

A Figura 3.1 ilustra o mecanismo de mineração de políticas de acesso proposto nesta tese. Esse mecanismo está dividido em 4 etapas: pré-processamento, extração, avaliação e refinamento.

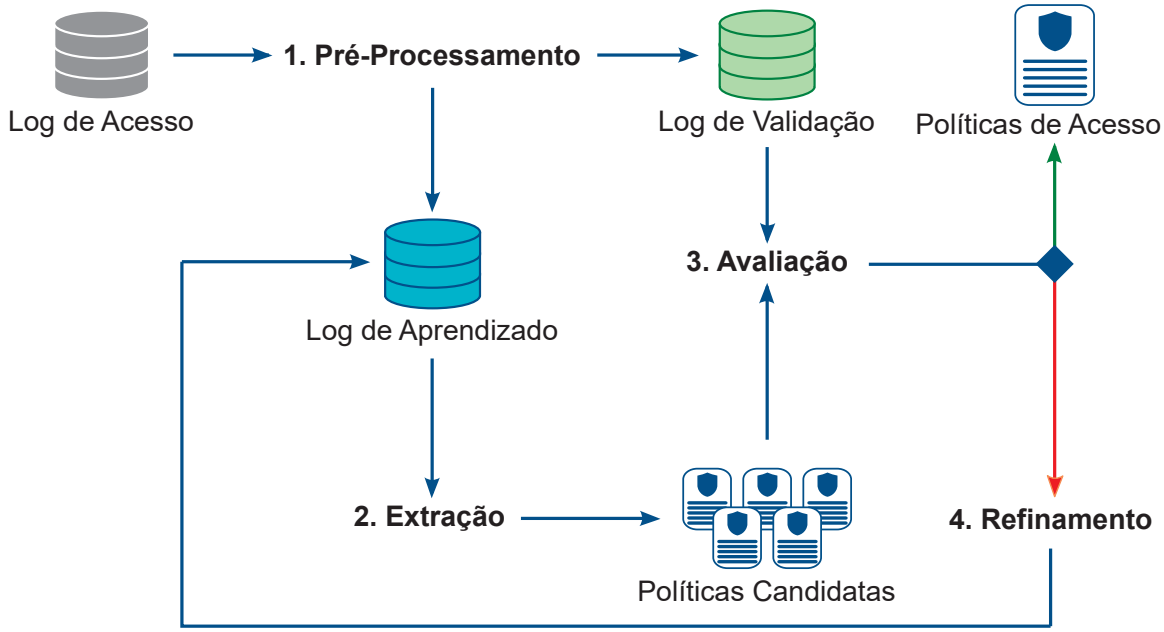


Figura 3.1: Visão geral do método de extração e refinamento de políticas

3.3.1 Pré-Processamento

O pré-processamento é um conjunto de subetapas que envolvem preparação, organização e estruturação do registro de acesso. Esta etapa precede a extração de política e está dividida em 5 subetapas: Entrada, Categorização, Complementação, Transformação e Divisão. Uma visão geral e exemplo do pré-processamento pode ser vista na Figura 3.2. É importante notar que, uma vez que a subetapa "entrada" possui grande simplicidade, sendo constituída apenas da leitura dos registros de acesso, seus detalhes serão ocultados na explicação das subetapas de pré-processamento.

Categorização: Nesta tese, todas as regras devem ser criadas tendo como base dados categóricos. Assim, o primeiro passo é pré-processar o registro de acesso de forma a converter todas as variáveis numéricas para variáveis categóricas. Como o mecanismo de controle de acesso se apropria de atributos dinâmicos como localização e horário, o nosso pré-processamento se inicia na transformação desses atributos de variáveis contínuas para variáveis discretas. Um determinado acesso pode possuir o atributo horário que define a exata hora em que o acesso ocorreu. Neste caso, acessos ocorridos entre 9h e 18h são transformados para uma variável

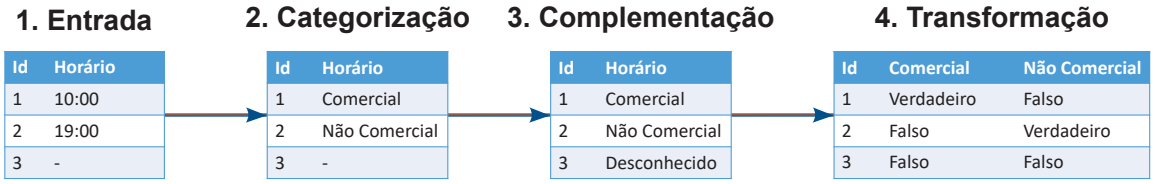


Figura 3.2: Exemplo das etapas do pré-processamento.

discreta como "comercial", enquanto acessos fora deste horário estão definidos como "Não comercial", como exemplificado na Figura 3.2 (2).

Complementação: Usualmente, registros de acesso podem apresentar informações faltantes. Logo, como exemplificado na Figura 3.2 (3), caso o registro apresente ausência de informação em alguma linha, o valor faltante é substituído por "Desconhecido" e esse valor será desconsiderado na execução.

Transformação: Após realizar a categorização e substituição dos valores faltantes, o pré-processamento realiza uma operação de transformação dos dados chamada *one-hot-encoding*. De maneira geral, essa transformação cria uma nova coluna com uma pergunta binária que indica a presença de cada possível valor dos atributos. Por exemplo, na Figura 3.2 (4), cada categoria do atributo horário foi atribuída em uma nova coluna. Sendo assim, criou-se a coluna "Comercial" e a coluna "Não Comercial". Note-se que, caso o valor do atributo seja comercial, a coluna comercial é preenchida com o valor "verdadeiro" e a coluna não comercial é preenchida com o valor "falso". Caso o valor seja não comercial, nota-se que ocorre o oposto. É importante ressaltar que, ao realizar essa transformação, é possível criar "perguntas" atômicas binárias, para cada coluna, as quais transmitem informações sobre o acesso. Por exemplo, na linha com identificador (id) 1 da Figura 3.2, podemos definir duas perguntas: "O horário de acesso é comercial?", "O horário de acesso é não comercial?". Por meio dessas perguntas, a próxima etapa do método de mineração de políticas (extração) gerará as políticas de acesso.

Divisão: A etapa da divisão é a etapa final do pré-processamento e consiste no particionamento do registro de acesso de treinamento em dois outros registros: registro de validação, que contém um subconjunto do registro de acesso que será utilizado para a avaliação da política de acesso, e o registro base de aprendizado, que contém o restante do registro de acesso e que será utilizado como base para extrair as políticas de acesso.

3.3.2 Extração

A etapa de extração de políticas recebe um registro de aprendizado pré-processado e entrega uma política de acesso condizente com tais dados. Para realizar essa tarefa, a extração foi dividida em outras 2 subetapas: inferência e interpretação. Nesta tese, a etapa de inferência se utiliza das árvores de decisão para extrair regras de classificação dos dados, e a interpretação realiza a transformação dessa árvore em uma política de acesso.

Inferência: Ao receber o registro de acesso pré-processado, as informações contidas nele estão na sua forma atômica, ou seja, definidas em uma função binária que estabelece ou não a presença de um determinado atributo em uma linha do registro. Para gerar as políticas de acesso positivas e negativas, optou-se por utilizar as árvores de decisão para classificar os acessos presentes no registro. Optou-se por esse modelo, pois as árvores de decisão são bastante estudadas na área de aprendizado de máquina e utilizadas para problemas de classificação (Umadevi e

Marseline, 2017). Logo, deseja-se criar um modelo de treinamento que pode ser utilizado para determinar se um acesso deve ser permitido.

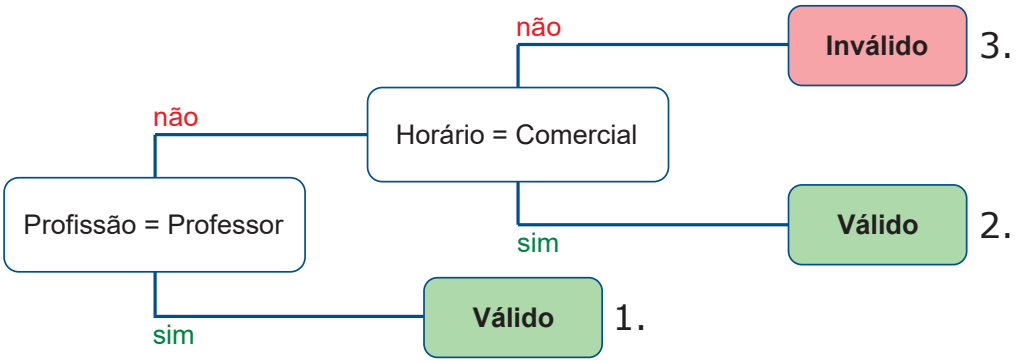


Figura 3.3: Exemplo de uma árvore de decisão gerada na sub-etapa de inferência

Interpretação: Uma vez que a árvore de decisão está construída, o processo de interpretação gera uma política de acesso através da união dos caminhos até as folhas de uma determinada classe. Por exemplo, na Figura 3.3, observa-se que as folhas 1 e 3 são da classe "Permitido" e a folha 2 é da classe "Negado". O processo de interpretação seleciona cada folha e gera o caminho até a raiz selecionando o nó superior e adicionando o valor do ramo. Quando o caminho até a raiz tem caminho maior que 1, expressão lógica "&&" é adicionada para cada novo salto até a raiz. Novamente utilizando a Figura 3.3 de exemplo, a folha 1 possui distância 1 da raiz e gera uma regra simples: "Se (Profissão=Professor)==Verdadeiro então Acesso permitido"; a folha 2, possui distância 2 e gera a seguinte regra "Se ((Profissão=Professor)==Falso && (Horário=Comercial)==Falso) então Acesso negado"; por fim, a folha 3, que possui distância 2, gera a seguinte regra "Se((Profissão=Professor)==Falso && (Horário=Comercial)==Verdadeiro) então Acesso permitido". Logo, ao utilizar a expressão lógica "|||" para realizar a união das folhas de uma mesma classe, temos que a política que permite o acesso é definida pela "folha 1 ||| folha 3" e a política que nega o acesso é definida pela "folha 2". Logo, a política resultante que permite o acesso é definida por "Se ((Profissão=Professor)==Verdadeiro) ||| ((Profissão=Professor)==Falso && (Horário=Comercial)==Verdadeiro) então Acesso permitido". Enquanto a política resultante que nega o acesso é definida por "Se ((Profissão=Professor)==Falso) && (Horário=Comercial)==Falso) então Acesso negado".

3.3.3 Avaliação

A etapa de avaliação se utiliza do registro de validação para testar a política de acesso gerada na etapa anterior. Para toda execução e cada política de acesso presente nos conjuntos de políticas de acesso candidatas, a avaliação testa todo registro de validação na política de acesso e define um valor de aptidão com base no tamanho da política de acesso, precisão e revocação. Essa aptidão possui como objetivo remover políticas de acesso exageradamente complexas ou políticas de acesso super permissivas. Como apresentado na sub-seção 4.1, define-se que o objetivo desta tese é apresentar regras de baixa complexidade e revocação, e essas regras devem possuir alta precisão. Assim, como esta tese possui 3 objetivos diferentes, foi definida uma soma ponderada entre esses objetivos com o intuito de transformá-los em um objetivo único. A métrica que definimos para unir esses objetivos é chamada de "Função de Aptidão Combinada de Precisão e Revogação" e está definida como: $Aptidão = A.(Precisão)^x + B.(Revogação)^y$

Nossa função de aptidão combina Precisão e Revogação de forma semelhante ao F1-Score, que utiliza a média harmônica dessas duas métricas. No entanto, diferentemente

do F1-Score, que atribui pesos iguais e não permite ajustes não-lineares, a função de aptidão proposta oferece maior flexibilidade ao introduzir pesos ajustáveis (A e B) e expoentes (x e y), permitindo priorizar métricas específicas de acordo com os objetivos do sistema. Ao assumir que A + B = 1, a métrica de aptidão definida varia entre 0 e 1, onde 0 representa políticas de acesso completamente inaptas e 1 políticas "perfeitas"para o registro de validação. Nessa fórmula, a Precisão avalia a proporção de decisões corretas, sendo essencial para medir a efetividade do modelo, enquanto a Revogação mede a capacidade de identificar todas as instâncias relevantes. Os pesos A e B ajustam a influência relativa de cada métrica, enquanto os expoentes x e y permitem modelar impactos não-lineares. Como a Precisão é um fator mais crítico, define-se que A > B e x > y, atribuindo maior peso à Precisão, mas aplicando maior punição a acessos inválidos em comparação a negações de acessos válidos. Em caso de empate na aptidão, utiliza-se como critério de desempate o tamanho da política de acesso, calculado como (quantidade de operadores && e ||) + 1.

3.3.4 Refinamento

A política de acesso gerada é altamente dependente do registro de aprendizado. Como esse registro de aprendizado contém apenas um conjunto de possibilidades de acesso, a política de acesso gerada por ele pode não ser a mais acurada. Assim, para remediar tal problema, a etapa de refinamento possui como objetivo injetar distorções intencionais no registro de aprendizado, com o intuito de se obter novas políticas de acesso candidatas. Visto que o espaço de busca para as novas políticas de acesso candidatas é grande, ao invés de realizar essas injeções de forma aleatória, optamos por realizá-las conforme uma heurística genética.

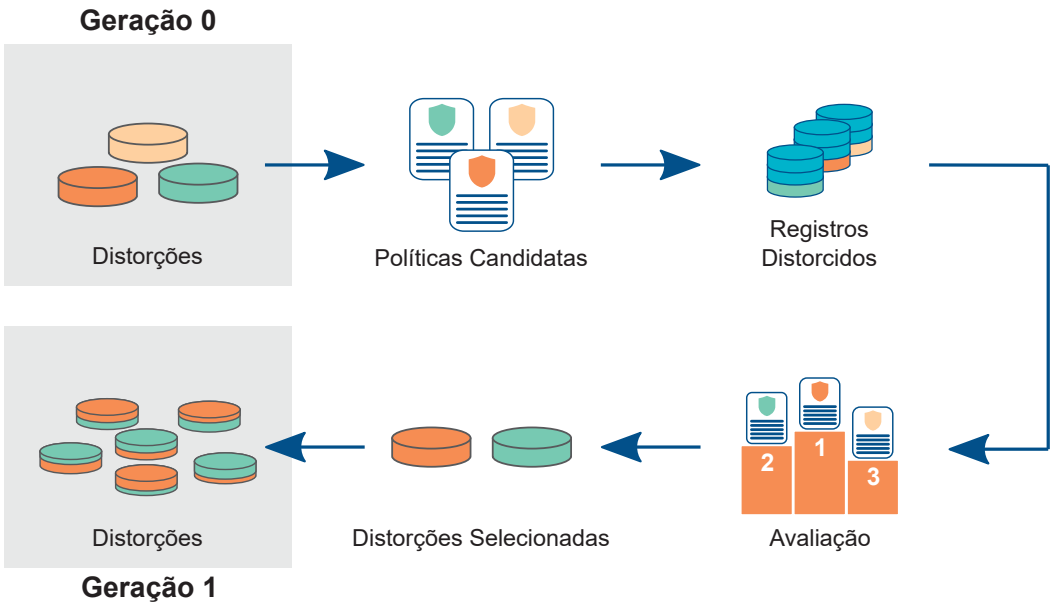


Figura 3.4: Exemplo das etapas do refinamento

Nesta tese, a fase de refinamento se apoia em um algoritmo genético. Em sua primeira etapa, é gerado um conjunto de acessos aleatórios (i), que serão agrupados para gerar um conjunto inicial de distorções (ii). Esse conjunto de distorções é chamado de geração 0 e cada elemento desse conjunto será injetado no registro base de aprendizado para se obter um registro distorcido. Por exemplo, na Figura 3.4 apresenta-se uma representação da fase de refinamento, observa-se 3 elementos na geração 0 (Figura 3.4a), de modo que cada um desses elementos será introduzido no

registro base de aprendizado e gerará um novo registro de aprendizado distorcido. É importante notar que esse registro distorcido é majoritariamente composto pelo registro de aprendizado disponibilizado na etapa de pré-processamento, apenas unido com um conjunto de entradas distorcidas. Nesta tese, estipulou-se que as distorções possuem 20% do tamanho de entradas do registro de aprendizado. Este percentual foi uma escolha prática que pode ser ajustada com base nos dados de entrada. Para cada registro distorcido, o processo de extração é executado, o que resulta em uma nova política de acesso (Figura 3.4b). Essa política de acesso então é avaliada, de forma que as distorções que geraram as políticas de acesso com os melhores valores de aptidão são selecionadas (Figura 3.4c) para gerar as distorções da próxima geração. Assim, com um conjunto de distorções que geraram melhores políticas de acesso, aplicam-se técnicas de cruzamento (iii), mutação (iv) e recombinação (v) para gerar uma nova geração de distorções, com base na geração anterior (Figura 3.4d). Esse processo se repete até um determinado número de gerações, ou quando as políticas de acesso conseguem atingir um nível aceitável na etapa de avaliação. A seguir, apresentamos detalhes sobre a geração de acessos aleatórios, geração do conjunto inicial de distorções, cruzamento, mutação e recombinação.

1. **Geração de Acessos Aleatórios:** Dado que o registro é composto por um conjunto de atributos e decisões de acesso, um acesso aleatório analisa os possíveis valores de cada atributo e gera uma decisão de acesso aleatória. Por exemplo, suponha um registro composto por acessos que possuem 2 atributos A e B. Suponha que o atributo A pode assumir os valores $\{a1, a2\}$, atributo B pode assumir os valores $\{b1, b2\}$ e a decisão de acesso assumir os valores $\{Negado, Permitido\}$. Um acesso aleatório é gerado ao selecionar um elemento aleatório de A, um elemento aleatório de B e uma decisão de acesso. Por exemplo, um conjunto de acessos aleatórios pode ser definido por $\{a1, b1, Negado\}$,$\{a1, b2, Permitido\}$,$\{a2, b1, Negado\}$,$\{a2, b2, Permitido\}$.

2. **Conjunto Inicial de Distorções:** Nesta tese, cada distorção representa um cromossomo. Esse cromossomo é modelado por meio de um vetor de tamanho N, em que $N > 1$, de forma que cada elemento desse vetor representa um acesso. O alelo de cada gene é representado por cada combinação de atributos e decisão de acesso possível de ocorrer no registro. Sendo assim, o conjunto inicial de distorções gera um conjunto de acessos aleatórios e os agrupa em diversos cromossomos.

3. **Cruzamento:** O cruzamento ocorre quando dois cromossomos são selecionados e combinados entre si. Para que isso ocorra, dado dois cromossomos A e B de tamanho N, um número aleatório i é selecionado, para $i \leq N$. Após definido o valor de i, o cruzamento une os elementos de 0 até i do vetor A com as posições $i + 1$ até N do vetor B e une os elementos de 0 até i do vetor B com as posições $i + 1$ até N do vetor A. Como resultado, dois novos vetores são gerados.

4. **Mutação:** Cada cromossomo possui um conjunto de genes, sendo cada gene uma decisão de acesso e um conjunto de atributos relacionados. Logo, a mutação seleciona um ou mais acessos de um cromossomo e substitui por outros gerados aleatoriamente.

5. **Recombinação**: Cada cromossomo é modelado em um vetor de tamanho N. O processo de recombinação seleciona um um número aleatório i, para $i \leq N$, e inverte a ordem do vetor. Ou seja, após definido o valor de i, os elementos entre $i + 1$ e N, passam a ser os de posição 0 e i, e vice-versa.

3.4 AVALIAÇÃO

Esta seção apresenta a avaliação do mecanismo de mineração de políticas de acesso descrita nesta tese. Primeiramente, descreve-se a metodologia de avaliação, incluindo o cenário considerado e as métricas para avaliar o seu desempenho. Em seguida, apresentam-se os resultados para várias combinações de estratégias de cache e políticas de substituição.

3.4.1 Metodologia

Nesta tese, vamos comparar o método proposto (A.Dec+Gen) com uma abordagem que utiliza árvore de decisão tradicional (A.Dec), similar ao trabalho (Bui e Stoller, 2020), porém aplicada para o mecanismo de controle de acesso. Os experimentos realizados para avaliar o desempenho de ambos os métodos foram realizados através de simulações computacionais. Os métodos foram desenvolvidos em *Python*, utilizando a biblioteca de aprendizado de máquina *sklearn* (Pedregosa et al., 2011). Os métodos foram testados em 2 bases de dados diferentes. A primeira base é constituída de dados sintéticos, sendo construída conforme as especificações de (Talukdar et al., 2017). A segunda base é constituída de dados reais da *Amazon*, disponibilizados na plataforma *Kaggle* (Amazon, 2013). Em cada base de dados, optamos por uma divisão de 80% de dados para treino e 20% de dados para teste. É importante notar que tanto a base de dados sintética quanto a base de dados da *Amazon* são desbalanceadas. Foram utilizadas três técnicas para lidar com desbalanceamento: nula (N), *oversampling* (O) e *undersampling* (U). A nula (N) consiste em executar ambos os métodos com os dados desbalanceados e ignorar o problema de desbalanceamento. O *oversampling* (O) consiste em replicar amostras aleatórias da classe minoritária, enquanto o *undersampling* (U) consiste em reduzir de forma aleatória os exemplos da classe majoritária. Nesta tese, o método proposto foi avaliado em máquinas de CPUs com 4 núcleos de 2.9 GHz e 8 GB de RAM. Em relação à árvore de decisão utilizada, o método proposto limita-se à sua altura em 6, para evitar árvores que gerem regras superespecializadas. Na parte evolutiva do método, o número máximo de gerações foi especificado em 200, sendo definido como o critério de parada o limite de gerações ou aptidão igual a 1.

3.4.2 Bases de Dados

A base de dados sintética foi dividida em 4 instâncias, sendo cada instância construída conforme as especificações de (Talukdar et al., 2017). Nessa base, deseja-se simular requisições de acesso que ocorreram de um conjunto de usuários sobre um determinado conjunto de objetos. Sendo assim, para a criação dessa base, geram-se 1000 usuários e 100 objetos. Cada usuário e objeto possuem 5 atributos únicos de forma que, para cada atributo, existem 5 valores diferentes. Quando uma requisição de acesso ocorre, essa requisição une todos os valores de atributos do usuário e todos os valores de atributos do objeto. Para determinar se esse acesso foi válido, foram geradas 30 regras aleatórias que contêm uma combinação de 5 valores de atributos (tanto do usuário quanto do objeto), que estipulam se o acesso é válido ou não. Uma vez estabelecidas as regras, cada usuário realiza uma requisição de acesso sobre cada objeto e, caso alguma regra seja satisfeita, o acesso é permitido; caso contrário, o acesso é negado. Para a construção dessa base, todas as n amostras de acesso permitidas foram selecionadas, enquanto somente um valor de n/5 amostras de acesso negadas foi selecionado. Ao todo, cerca de 1800 acessos foram selecionados para cada instância da base de dados sintética.

A base de dados disponibilizada pela *Amazon* é constituída de informações reais de dados coletados entre 2010 e 2011. Nela existem diversos acessos de usuários aos diferentes recursos da empresa, sendo todos esses acessos manualmente definidos por um administrador de

rede. Ao todo, essa base de dados possui 32 mil acessos de 12 mil usuários únicos em 7 mil recursos diferentes. Nessa base de dados, cada acesso possui 9 atributos relacionados com os usuários e os recursos. Essa base se destaca por ser uma base com grande desbalanceamento, possuindo cerca de 93% dos acessos permitidos. Para essa base de dados, a metodologia de separação das instâncias apresentada por Cotrini et al. (2018) foi seguida, de modo que todos os acessos que ocorreram sobre cada recurso foram agrupados, o que permitiu a criação de 7 mil instâncias de entrada, uma para cada recurso. Nesta tese, selecionamos as mesmas instâncias utilizadas no trabalho de referência.

3.4.3 Métricas de Avaliação

Para a avaliação de desempenho dos métodos, cada um foi mensurado, conforme as seguintes métricas, comumente utilizadas nesse contexto:

1. **Acurácia**: Fração de acessos corretamente classificados pela política de acesso em relação a todos os acessos (tanto acessos permitidos quanto negados);
2. **Precisão**: Fração de acerto de todos os acessos permitido, ou seja, dentre todas as classificações de acesso permitido que o modelo fez, quantas estão corretas;
3. **Revocação**: Fração de acerto dentre todos os acertos possíveis, ou seja, dentre todas as situações em que o acesso permitido é o valor esperado, quantas estão corretas;
4. **F1 score**: Média harmônica entre Precisão e Revocação;
5. **Taxa de Verdadeiro Negativo (TNR):** Proporção de acessos negados que devem ser genuinamente negados;
6. **Complexidade (Comp):** Somatório do tamanho de todas as regras da política de acesso.

3.4.4 Resultados

As Tabelas 3.1 e 3.2 sumarizam os resultados obtidos do método de comparação (A.Dec) e o método proposto (A.Dec+Gen) em relação ao dataset sintético e o da *Amazon*, respectivamente. Nelas, apresentamos cada instância gerada a partir dos datasets apresentados e mostramos o desempenho de ambos os métodos em relação às métricas apresentadas anteriormente. Nestas tabelas, ambos os métodos foram utilizados em conjunto com o balanceamento nulo (N), *oversampling* (O) e *undersampling* (U).

Na Tabela 3.1 é possível observar que a acurácia do método proposto é superior em todas as instâncias. Destaque para a instância Sintético#3, em que o método proposto, aplicado com a técnica de balanceamento *oversampling*, apresentou uma acurácia aproximadamente 10% superior em relação ao método de comparação. Sobre a precisão e revocação, ambas apresentaram resultados superiores ao método proposto. Consequentemente, o F1 *score* apresentou um melhor desempenho em nossa abordagem, ficando acima de 0.9 em todas as instâncias. Ter um valor alto nessa métrica é importante, pois nossa base de dados é bastante desbalanceada. Em situações como essa, um baixo valor do F1 *score* significa que a política de acesso criada está enviesada para aceitar mais acessos do que negar. Essa característica foi evidenciada na taxa de verdadeiro negativo (TNR), em que o método proposto foi capaz de aumentar a capacidade da política de acesso de negar acessos corretamente em todas as instâncias, exceto a primeira. Em relação à complexidade da política de acesso, é possível observar um movimento contrário, em que o método proposto apresentou regras mais complexas do que o comparado. Esse resultado foi

Instâncias	Técnica	Acurácia	Precisão	Revocação	F1	TNR	Comp.
Sintético#1	A.Dec (N)	0.818	0.868	0.900	0.884	0.544	**35**
	A.Dec (O)	0.832	0.915	0.862	0.888	**0.733**	58
	A.Dec (U)	0.795	0.904	0.820	0.860	0.711	72
	A.Dec+Gen (N)	0.882	0.904	**0.947**	**0.925**	0.667	74
	A.Dec+Gen (O)	0.858	0.912	0.902	0.907	0.711	60
	A.Dec+Gen (U)	**0.876**	**0.917**	0.922	0.920	0.722	59
Sintético#2	A.Dec (N)	0.846	0.887	0.925	0.905	0.543	57
	A.Dec (O)	0.860	0.900	0.927	0.913	0.602	58
	A.Dec (U)	0.857	0.905	0.916	0.910	0.626	**55**
	A.Dec+Gen (N)	0.875	0.882	**0.974**	0.925	0.496	89
	A.Dec+Gen (O)	**0.907**	**0.930**	0.955	**0.942**	**0.72**	70
	A.Dec+Gen (U)	0.863	0.910	0.919	0.914	0.650	59
Sintético#3	A.Dec (N)	0.793	0.812	**0.951**	0.876	0.264	83
	A.Dec (O)	0.759	0.934	0.740	0.826	0.826	**58**
	A.Dec (U)	0.802	0.910	0.825	0.865	0.727	59
	A.Dec+Gen (N)	0.873	0.927	0.907	0.917	0.760	69
	A.Dec+Gen (O)	**0.892**	0.918	0.945	**0.931**	0.716	73
	A.Dec+Gen (U)	0.826	**0.951**	0.816	0.879	**0.859**	77
Sintético#4	A.Dec (N)	0.812	0.850	0.919	0.883	0.457	64
	A.Dec (O)	0.809	0.845	0.921	0.882	0.437	64
	A.Dec (U)	0.798	0.874	0.861	0.868	0.586	**50**
	A.Dec+Gen (N)	0.867	**0.907**	0.921	0.914	**0.685**	84
	A.Dec+Gen (O)	**0.876**	0.874	**0.980**	**0.924**	0.526	84
	A.Dec+Gen (U)	0.846	0.893	0.909	0.901	0.636	73

Tabela 3.1: Dataset Sintético

esperado, uma vez que a complexidade não está diretamente introduzida na métrica de aptidão apresentada na Seção 3.3. Sendo assim, uma vez que a complexidade foi utilizada apenas como critério de desempate, nota-se que o método proposto priorizou a acurácia e a revocação.

A Tabela 3.2 mostra que o método proposto apresenta ganhos, mesmo quando utilizado em dados reais. Em relação à acurácia, o método proposto apresenta melhores resultados em todas as instâncias. No entanto, é importante destacar que a base de dados utilizada é extremamente desbalanceada, com a maioria dos acessos permitidos e uma minoria representando acessos negados. Esse desbalanceamento pode ter contribuído para os resultados elevados de acurácia, já que essa métrica tende a favorecer a classe de maior representatividade na base de dados. Em relação à precisão, embora o método proposto também tenha apresentado melhores resultados, os valores são novamente próximos aos do método comparado. Esse resultado reflete que o método proposto foi capaz de criar políticas capazes de identificar acessos que devem ser negados de forma mais precisa. Com relação à revocação, entretanto, notamos uma melhora em todos os casos, sendo a instância Rec.75078, a única que não apresentou a métrica de revocação perfeita. Consequentemente, o F1 *score* apresentou um melhor desempenho em nossa abordagem, ficando acima do método comparado em todas as instâncias. Quanto à taxa de verdadeiro negativo, é importante notar que o método proposto obteve resultados expressivos se contrastado com o método de comparação. O método de comparação assume como base uma árvore de decisão simples e gera políticas de acesso com base nessa informação. No método proposto, por outro lado, alguns dados sintéticos são adicionados para realizar a etapa de evolução. Consequentemente, o método proposto é capaz de extrapolar informações não apresentadas no *dataset* de entrada, gerando um conjunto de políticas de acesso mais complexas em relação ao método de comparação, porém com uma chance de, em um momento da evolução, gerar uma política de acesso correta a partir de dados não apresentados na entrada. Uma vez que esses

Instâncias	Técnica	Acurácia	Precisão	Revocação	F1	TNR	Comp.
Rec.25993	A.Dec (N)	0.911	0.957	0.949	0.953	0.167	**5**
	A.Dec (O)	0.935	0.958	0.974	0.966	0.167	**5**
	A.Dec (U)	0.919	0.957	0.957	0.957	0.167	**5**
	A.Dec+Gen (N)	**0.967**	**0.975**	0.991	**0.983**	**0.500**	14
	A.Dec+Gen (O)	0.959	0.959	**1**	0.979	0.167	10
	A.Dec+Gen (U)	0.951	0.959	0.991	0.975	0.167	7
Rec.4675	A.Dec (N)	0.988	0.992	0.996	0.994	0.333	**1**
	A.Dec (O)	0.988	0.992	0.996	0.994	0.333	2
	A.Dec (U)	0.984	0.992	0.992	0.992	0.333	**1**
	A.Dec+Gen (N)	0.988	0.992	0.996	0.994	0.333	5
	A.Dec+Gen (O)	**0.992**	0.992	**1**	**0.996**	0.333	12
	A.Dec+Gen (U)	0.988	**0.996**	0.992	0.994	**0.667**	3
Rec.75078	A.Dec (N)	0.968	0.984	0.984	0.984	0.333	3
	A.Dec (O)	**0.976**	0.984	**0.992**	**0.988**	0.333	3
	A.Dec (U)	0.944	0.991	0.951	0.971	**0.667**	**2**
	A.Dec+Gen (N)	**0.976**	0.984	**0.992**	**0.988**	0.333	6
	A.Dec+Gen (O)	**0.976**	0.984	**0.992**	**0.988**	0.333	10
	A.Dec+Gen (U)	0.968	**0.992**	0.975	0.983	**0.667**	**2**
Rec.79092	A.Dec (N)	0.979	**0.986**	0.993	0.989	**0.600**	**5**
	A.Dec (O)	0.959	0.979	0.979	0.979	0.400	**5**
	A.Dec (U)	0.945	0.978	0.965	0.971	0.400	10
	A.Dec+Gen (N)	**0.986**	**0.986**	**1**	**0.993**	**0.600**	10
	A.Dec+Gen (O)	**0.986**	**0.986**	**1**	**0.993**	**0.600**	13
	A.Dec+Gen (U)	0.945	0.978	0.965	0.971	0.400	12

Tabela 3.2: Dataset Amazon

dados conseguem gerar uma regra de negar um acesso corretamente e a base de dados possui poucos exemplos dessa negação, o método proposto foi capaz de manter essa regra na política de acesso gerada. Devido a essa característica, notamos que os nossos ganhos vêm acompanhados de um pequeno aumento na complexidade da regra, uma vez que as distorções inseridas acabaram sendo adicionadas na política de acesso. Porém, notamos que esse aumento não é exponencial e permanece na mesma ordem de grandeza do método comparado.

3.5 RESUMO

Os resultados indicaram que nossa proposta possui melhores resultados em relação a acurácia, precisão, revocação e taxa de verdadeiro negativo. Ainda que essa melhora ocorra em ambas as bases de dados comparadas, os ganhos se mostraram mais expressivos na base de dados sintética. Nesta tese, foi possível observar que em bases extremamente desbalanceadas, com poucos exemplos de acesso negado, ambos os métodos comparados possuíram dificuldades em identificar regras de negação. No entanto, como o método proposto possui uma característica evolutiva, ele foi capaz de gerar políticas que permitem negar acessos corretamente com base na injeção de distorções nas bases de dados. Porém, ainda que o método proposto apresentasse melhores resultados, ele introduziu distorções na política resultante, o que contribuiu para um aumento significativo na complexidade das políticas de acesso geradas. Sendo assim, destacamos a importância de aplicar técnicas mais sofisticadas tanto na geração quanto na simplificação das políticas, de forma a eliminar as distorções inseridas e garantir políticas de acesso mais claras e eficientes.

Portanto, neste capítulo, apresentamos os resultados alcançados, os quais estão alinhados com o OE1 - relacionados ao uso do comportamento do usuário para gerenciar políticas de acesso. Dessa forma, nossas principais contribuições neste capítulo podem ser resumidas em:

- Propusemos um método de mineração de políticas que emprega técnicas de árvore de decisão e algoritmos genéticos em registros de acesso para extrair políticas de acesso.
- Quando os conjuntos de aprendizado estão equilibrados, o método demonstrou ser capaz de extrair políticas precisas e auditáveis, apresentando uma repercussão negativa mínima em relação ao tamanho das políticas de acesso quando comparado a outros métodos da literatura.
- Em conjuntos de dados extremamente desequilibrados, tanto o método proposto quanto os encontrados na literatura enfrentaram dificuldades para fornecer resultados precisos

No próximo capítulo, abordamos nossos outros dois objetivos de pesquisa (OE2 e OE3), em que apresentaremos uma arquitetura e um mecanismo de distribuição de atributos para aumentar o desempenho da avaliação das políticas de acesso presentes na aplicação, inclusive das políticas geradas neste capítulo.

4 MECANISMO ADAPTATIVO E OPORTUNÍSTICO DE DISTRIBUIÇÃO DE ATRIBUTOS EM CACHES PARA IOT

Neste capítulo, serão abordados os OE2 e OE3, com a introdução de um repositório de atributos voltado para a IoT. Esse repositório é projetado para armazenar e replicar atributos de identidades digitais em caches distribuídas pela rede, com base no comportamento do usuário ao utilizar uma aplicação. No capítulo anterior, foi demonstrado que as regras podem ser mineradas de forma eficiente. Agora, o foco está no desempenho, especialmente na latência envolvida na avaliação das políticas de acesso. Para garantir eficiência na avaliação dessas políticas, é essencial que o ponto de decisão tenha acesso a todos os atributos necessários. Porém, ao considerar que esses atributos frequentemente estão localizados em nuvens computacionais distribuídas geograficamente, as aplicações IoT são impactadas por uma latência adicional significativa relacionada à recuperação desses dados.

Dado os problemas mencionados, pretendemos responder às seguintes perguntas neste capítulo:

- Quais são as técnicas utilizadas pela literatura para reduzir a latência no processo de recuperação de atributos?
- Como pode ser uma arquitetura distribuída de atributos e quais os impactos da utilização dessa nova arquitetura?
- Como seria o comportamento de um usuário de uma aplicação IoT real e como pode-se utilizar esse comportamento para otimizar a distribuição de atributos?

Dessa forma, iniciamos a primeira questão deste capítulo na Seção 4.1. Nessa seção, vários estudos prévios relacionados ao tema são apresentados, destacando propostas e abordagens destinadas a melhorar o desempenho do processo de avaliação das políticas de acesso. Em seguida, na Seção 4.2, começamos a tratar da segunda questão do capítulo. Aqui, apresentamos a arquitetura proposta para um repositório de atributos direcionado à IoT. Nesta parte, os principais componentes deste repositório e suas funcionalidades são discutidos. Os problemas que os repositórios de atributos para IoT enfrentam são formalizados na Seção 4.3, fornecendo respostas para nossa segunda questão. A terceira questão do capítulo começa na Seção 4.4 e se estende até o final do capítulo. Aqui, apresentamos a caracterização de uma aplicação IoT em um campus universitário. Essencialmente, apresentaremos um caso de estudo para demonstrar como o repositório de atributos para IoT proposto pode ser aplicado em um ambiente universitário real para aprimorar o desempenho do mecanismo de controle de acesso. A Seção 4.6 introduz os dois mecanismos centrais do repositório de atributos para IoT proposto: o mecanismo de distribuição de atributos e o mecanismo de sincronização de atributos. Esses mecanismos são fundamentais para garantir o desempenho do processo de autorização e a consistência dos atributos replicados nas caches de atributos. A análise formal do repositório de atributos para IoT proposto é realizada na Seção 4.7. Aqui, verificamos a consistência e a eficácia dos mecanismos apresentados. Na Seção 4.8, apresentamos os resultados da avaliação do repositório de atributos para IoT, demonstrando sua eficácia e desempenho no cenário do campus universitário. Finalmente, na Seção 4.9, destacamos a relevância dos resultados obtidos e fornecemos um resumo dos tópicos discutidos, concluindo este capítulo.

4.1 TRABALHOS RELACIONADOS

O ABAC é um método de autorização no qual políticas de acesso são escritas através dos atributos associados às identidades das entidades envolvidas no acesso. Amplamente reconhecido como o mecanismo de acesso mais eficiente no contexto da IoT, todas as entidades envolvidas no acesso possuem identidades e os atributos respectivos de suas identidades podem ser utilizados em suas políticas de acesso. Sendo assim, a recuperação dos atributos das identidades é uma etapa obrigatória para que uma política de acesso possa ser avaliada corretamente. Quando os atributos estão próximos da entidade que realiza a avaliação da política de acesso, o processo de recuperar os atributos não é considerado uma operação custosa. Porém, como introduzido por Hu et al. (2013), 2013), quando esses atributos vêm de múltiplas fontes externas, os problemas de desempenho podem surgir no mecanismo de controle de acesso.

De acordo com Hu et al. (2013), utilizar caches de atributos pode aliviar esse problema e tornar o processo de recuperação de atributos menos custoso para o processo de autorização, mas a distribuição desses atributos nas caches não pode ser realizada de forma negligente aos aspectos de segurança. Em seu trabalho, os autores apontaram que o problema de propagação de atributos difere de uma cache tradicional em vários aspectos. Dentre eles, é citado que uma cache tradicional armazena dados de aplicação, enquanto a propagação de atributos armazena informações de identidade das entidades presentes na aplicação, muitas vezes formadas por dados pessoais e sensíveis. Logo, além do desempenho, utilizar caches para armazenar os atributos de identidades também envolve questões de segurança, como garantir que os atributos das entidades sejam replicados de forma segura, atualizados em cada mudança e protegidos contra ataques quando armazenados. Logo, os autores concluíram que um equilíbrio deve ser encontrado entre replicar os atributos em cache para garantir desempenho e garantir a segurança desses atributos.

Nesta linha de pesquisa, Gómez et al. (2018) apresentou uma arquitetura de névoa-nuvem e demonstrou seus benefícios para o método de controle de acesso. Os autores se concentram na ineficiência da recuperação de atributos no mecanismo de controle de acesso e argumentam que, para reduzir o tempo de recuperação, deve-se fragmentar as identidades em identidades parciais com um subconjunto de atributos e utilizar os nós de névoa como dispositivos agregadores e de armazenamento. Sendo assim, uma vez que atuam como cache, esses nós da névoa oferecem os atributos de forma mais próxima ao local de onde as políticas são avaliadas, o que, consequentemente, reduz o tempo de recuperação dos atributos. Seus resultados mostraram que o uso dos nós de névoa para armazenar os atributos impacta positivamente o desempenho do método de controle de acesso. No entanto, os autores concluíram que os atributos devem ser colocados em pontos estratégicos da rede para atender às muitas solicitações de acesso e investigações posteriores são necessárias para o melhor posicionamento desses atributos.

Em Bhatt (2018), os autores apresentam um modelo de comunicação para o mecanismo de controle de acesso voltado para aplicações IoT que são executadas na nuvem. Nesse trabalho, os autores sugerem um novo fluxo de dados para aplicar políticas de acesso e novamente apontam que a névoa pode contribuir para a solução. De maneira geral, os autores demonstraram a necessidade de caches de atributos e mostraram como dispositivos de névoa IoT poderiam armazenar atributos para reduzir a latência de uma decisão de política. Embora mencionado como uma parte importante, o mecanismo de cache desse trabalho é simples e tratado de forma rasa. Quando os atributos são necessários, o primeiro passo é consultá-los na nuvem. Após sua utilização, esses atributos são armazenados em cache no nó que requisitou os atributos e usados em decisões de acesso subsequentes. De maneira geral, essa é uma abordagem clássica de cache chamada LEAF, em que o nó que requisita um arquivo é o nó que cria uma cópia do arquivo

solicitado para usos posteriores. Logo, o problema de cache de atributos é tratado como um problema convencional de cache e pouco foi discutido sobre a segurança dos atributos replicados.

No trabalho de (Castro et al., 2019) os autores apresentaram diversas modificações em uma arquitetura IoT para acelerar a avaliação de políticas no mecanismo de controle de acesso. Eles propuseram um sistema de controle de acesso híbrido, que avalia parcialmente as políticas de acesso nos dispositivos IoT, além de um mecanismo de cache de políticas de acesso. Os autores argumentaram que os *gateways*, presentes em grande parte das aplicações IoT, podem ser utilizados para mediar as comunicações entre os objetos. Logo, eles são excelentes candidatos para armazenar *tokens* de acesso e processar políticas de acesso devido ao seu poder de computação e de memória usualmente maior do que os demais dispositivos IoT da rede. Embora tenha apresentado resultados que demonstrem a eficiência de suas propostas, esse trabalho se focou na distribuição de políticas de acesso. Logo, pouco se foi discutido em relação a como alcançar os atributos para a avaliação de política e como revalidar os acessos caso uma política ou atributo sofra uma alteração após um acesso ser concedido.

Em Liu et al. (2020), os autores propuseram um método de recuperação de políticas de acesso para melhorar o desempenho da autorização em redes multimídia IoT. Semelhante ao trabalho anterior, ao invés de utilizar caches de atributos, os autores utilizaram a cache para armazenar as políticas de acesso. O diferencial desse trabalho é que o mecanismo de avaliação das políticas de acesso foi criado com base em um aprendizado de máquina. Sendo assim, as políticas de acesso eram interpretadas como uma árvore de decisão e fragmentos dessa árvore eram armazenados nas caches. Quando necessário, a recuperação de políticas e a decisão de políticas como um todo, ocorria através desses fragmentos da árvore distribuída nas caches. Os autores mostraram que o método proposto foi mais eficiente na avaliação de políticas de controle de acesso e que as caches aprimoraram a escalabilidade do mecanismo de controle de acesso. Porém, apesar de as caches de políticas de acesso serem eficientes, a recuperação dos atributos para avaliação dessas políticas de acesso ficou fora do escopo do trabalho.

Ravidas et al. (2020) é uma das abordagens mais diferentes em relação às demais. Em seu trabalho, os autores assumem que os atrasos para recuperar atributos irão ocorrer e, em alguns casos, o acesso aos próprios atributos pode estar indisponível. Sendo assim, com base nessa suposição, os autores criaram um mecanismo que permite tomar uma decisão sem possuir todos os atributos. Em sua abordagem, o atributo faltante é inferido através de um grafo de atributos que codifica o relacionamento dos acessos e a proximidade semântica das identidades para calcular sua similaridade. Assim, quando um atributo está ausente, é usada uma função de avaliação de política probabilística para calcular uma estimativa do atributo faltante e enviar para o ponto de decisão esse atributo inferido. Portanto, a política de decisão usa esse atributo inferido e reduz qualquer atraso que a recuperação de atributos acarretaria. No entanto, embora essa abordagem aumente o desempenho do mecanismo de controle de acesso, ela também introduz riscos de conceder ou negar acesso de forma incorreta e pouco se foi discutido sobre o que fazer quando esse atributo se tornar disponível novamente.

O trabalho de Siebach e Jess (2021) apresenta uma arquitetura de autorização chamada Abacus. Segundo os autores, ela foi desenvolvida com o objetivo de superar problemas inerentes em outras soluções de autorização de empresas e de código aberto. Em um dos mecanismos da arquitetura, os autores propõem uma cache de atributos localizada o mais próxima possível do mecanismo de decisão de políticas para reduzir a latência entre essas duas entidades e fornecer decisões com melhor desempenho global. No entanto, os autores não oferecem uma visão aprofundada desse componente de cache e apresentaram-no de forma simples e rasa.

Ainda que os trabalhos apresentados anteriormente tenham destacado a latência da autorização e a recuperação de atributos como um problema no mecanismo de autorização, apenas

Gomes et al. (2021a) o enfrentou diretamente e propôs uma visão distribuída do armazenamento de atributos. Em sua proposta, os autores criaram várias caches de atributos em vários pontos da rede e mostraram como diferentes estratégias de cache podem reduzir a latência da recuperação de atributos. Os autores mediram a qualidade de várias estratégias de cache tradicionais em um cenário completamente simulado e não destacaram as particularidades de se armazenar atributos em cache. Adicionalmente, por ser um cenário simulado, ainda que a literatura não apresente um *dataset* disponível, um cenário real traria a possibilidade de explorar outras estratégias de cache mais específicas para um determinado cenário.

References	**Network Model**	**Cache for**	**Cache Policies**	**Evaluation Method**	**Database**
Hu et al. (2013)	Not informed	Suggestion	N/A	N/A	N/A
Gómez et al. (2018)	Hierarchical	Attributes	Not informed	Simulation	Not informed
Liu et al. (2020)	Non-hierarchical	Policy decisions	Not informed	Simulation	Synthetic
Castro et al. (2019)	Hierarchical	Policy decisions	Not informed	Simulation	Synthetic
Ravidas et al. (2020)	Non-hierarchical	N/A	Not informed	Mathematical	N/A
Bhatt (2018)	Hierarchical	Attributes	LEAF, FIFO	Simulation	Synthetic
Siebach e Jess (2021)	Hierarchical	Attributes and Policies	LEAF, FIFO	Simulation	Synthetic
Gomes et al. (2021a)	Hierarchical	Attributes	**LCE, LCD, LRU, FIFO, RR, Probabilistic**	Simulation	**Real**
This	Hierarchical	Attributes	**LCE, LCD, LEAF, LRU, FIFO, RR, SLRU, SPROT**	Simulation	**Real**

Tabela 4.1: Resumo dos trabalhos relacionados

A Tabela 4.1 apresenta um resumo dos trabalhos mencionados nessa seção. A análise literária mostra várias abordagens para aumentar o desempenho do mecanismo de controle de acesso para IoT. Entre todos os trabalhos,Hu et al. (2013) é visto como um guia, já que contém quase todas as definições e considerações obrigatórias para mecanismos de controle de acesso que utilizam atributos. Em relação à estrutura de rede, existe uma tendência em relação ao uso de redes hierárquicas nuvem-névoa para IoT, exceto para Liu et al. (2020) e Ravidas et al. (2020), em que não há requisito para uma estrutura de rede. Ainda que não utilize o termo "cache" especificamente, Hu et al. (2013) apresenta diferentes tipos de caches para aumentar o desempenho do mecanismo de controle de acesso, como caches de decisão de política, caches de política e caches de atributos. Com exceção do trabalho de Ravidas et al. (2020), todos os trabalhos examinados aqui lidam com um tipo específico de cache. Trabalhos como Liu et al. (2020) e Castro et al. (2019) empregam caches para armazenar decisões e reduzir o tempo de decisão da política ao não reavaliar escolhas anteriores. Trabalhos como Bhatt (2018) usam caches para armazenar políticas, tornando a recuperação da política e, consequentemente, a decisão da política mais rápida. Por fim, trabalhos como Gómez et al. (2018), Bhatt (2018), Siebach e Jess (2021) e Gomes et al. (2021a) usam caches para reduzir o tempo de recuperação de atributos. Este manuscrito de tese pertence a este último conjunto de trabalhos. Nessa categoria de trabalhos, exceto pelo nosso trabalho anterior, a literatura analisada contém três trabalhos Gómez et al. (2018), Bhatt (2018), Siebach e Jess (2021), todos apresentando caches de atributos com abordagens simples e sem uma análise aprofundada. Suas simulações usaram apenas dados sintéticos, e a cache de atributos proposta é míope, sem uma estratégia por trás para problemas de cache padrão, como decisão de cache e substituição de cache. Ao contrário dos outros trabalhos relatados, a proposta desse manuscrito de tese se aprofunda no problema de cache de atributos. Inspirados em Ravidas et al. (2020), não abordamos o problema de cache de atributos com uma abordagem míope. Em vez disso, usamos previsões para ajudar essas abordagens míope a torná-las mais inteligentes e eficientes, melhorando a eficiência do mecanismo de controle de acesso e reduzindo o número de réplicas de atributos em cache. Assim,

nosso trabalho apresenta diferentes estratégias de cache e analisa seus resultados. Da mesma forma, em vez de usar dados sintéticos, usamos dados reais de uma aplicação em um campus para avaliar o desempenho de caches de atributos e diferentes estratégias de cache.

4.2 ARQUITETURA PROPOSTA

Neste manuscrito de tese, a arquitetura proposta é uma extensão da arquitetura XACML, com o objetivo de torná-la mais adequada aos requisitos de aplicações IoT, incluindo a necessidade de lidar com latência. A escolha por essa arquitetura foi motivada por sua consolidação no mercado e pelo fato de ter sido projetada especificamente para autorização, utilizando atributos na definição e avaliação de políticas de acesso. No contexto das aplicações IoT, que são o foco principal desta tese, várias pesquisas na literatura destacam a compatibilidade e flexibilidade do XACML para lidar com os requisitos de autorização em diferentes aplicações IoT. Além disso, a literatura indica que essa arquitetura é capaz de dar suporte a tecnologias de nuvem e névoa. Embora existam outras opções, como SecPal e Permis, esses padrões não têm o mesmo reconhecimento amplo quando comparados ao XACML (Atlam et al., 2018; Silva et al., 2018; Al-Hasnawi et al., 2019). Sendo assim, nesta seção, será apresentado um modelo da arquitetura proposta para um repositório de identidades operacional e em conformidade com a arquitetura XACML.

Usuários, atributos e políticas de Acesso: Na arquitetura proposta, define-se que apenas usuários podem requisitar acesso a um recurso. Porém, no contexto IoT, esse usuário pode ser uma pessoa, um serviço, um aplicativo ou até mesmo um dispositivo da rede. Independentemente da natureza do usuário, ele sempre terá uma identidade com um conjunto de atributos que o caracteriza. Nesta tese, os atributos abrangem um conjunto discreto de valores, e as políticas de acesso são elaboradas com base nesses atributos. Isso significa que a política define quais valores dos atributos um usuário deve possuir para acessar um recurso. Por exemplo, considere um usuário com o atributo “cargo" definido como “professor". Ao tentar executar uma operação de leitura em um recurso da universidade (um dispositivo IoT, por exemplo), esse usuário será autorizado somente se houver uma política no mecanismo de autorização da universidade que permita a professores realizar operações de leitura nesse recurso específico.

PIP, PAP e PDP: Na arquitetura proposta, a rede é organizada em uma topologia de árvore, na qual a raiz representa um serviço em nuvem (ou um conjunto de serviços externos em nuvem) que guarda os atributos de todos os usuários. De acordo com a arquitetura XACML, o nó raiz funciona como um PIP com capacidade suficiente para armazenar todas as informações sobre todos os usuários dessa aplicação. Os nós internos da árvore são os nós de névoa (roteadores, pontos de acesso, *switches*, etc), que podem atuar como PEP e PDP. Para facilitar a compreensão desta tese, esses nós são integrados, ou seja, desempenham as funções de PEP e PDP simultaneamente. Embora isso possa parecer impreciso, várias aplicações reais combinam PEP e PDP em uma única entidade para simplificar o sistema. Dessa forma, mesmo não sendo a abordagem mais escalável e extensível do XACML, essa unificação proporciona simplicidade sem comprometer a generalização do XACML. Além disso, como os nós de névoa também possuem capacidade de armazenamento, embora não seja tão ampla quanto a da nuvem, define-se que eles podem atuar como pequenos PIPs e armazenar um subconjunto dos atributos dos usuários para agilizar a avaliação da política.

Modelo de solicitação e resposta de atributos: A Figura 4.1 apresenta o modelo de solicitação e resposta de atributos durante a avaliação de uma política. O nó raiz r representa o servidor em nuvem, enquanto os outros nós (f_1, f_2, ..., f_n) são nós de névoa, como roteadores e *switches*. Neste exemplo, o nó f_n recebe uma solicitação de acesso e precisa avaliar se tal

solicitação é válida. No entanto, para realizar essa avaliação, os atributos necessários não estão disponíveis nele. Logo, para obter esses atributos, o nó f_n envia uma solicitação em direção à nuvem. Considerando que os nós de névoa podem atuar como PIP, se possuírem algum atributo solicitado em seu armazenamento, esse atributo é enviado diretamente da névoa. No exemplo apresentado, caso nenhum dos nós de névoa possua os atributos requisitados, a solicitação chega à nuvem, que retorna os atributos com uma latência correspondente à distância máxima d.

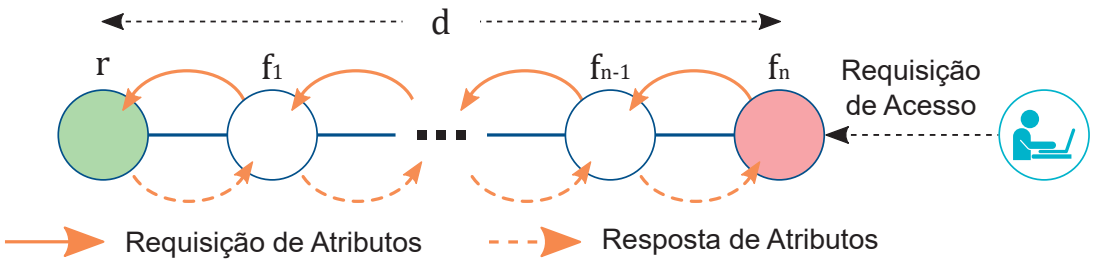


Figura 4.1: Estratégias reativas de cache de atributos

Essa arquitetura proposta se diferencia em relação ao modelo tradicional do XACML nos seguintes pontos: Enquanto o XACML convencional assume um PIP centralizado, a arquitetura proposta distribui atributos entre os nós de névoa, que atuam como pequenos PIPs locais. Essa descentralização tem o objetivo de criar uma estrutura de árvore, reduzir a necessidade de consultas frequentes à nuvem e minimizar a latência. A topologia em árvore e a consideração explícita da latência são adaptações propostas nesta tese para garantir a recuperação dos atributos em cenários IoT com a latência necessária para esse tipo de aplicação.

4.3 PROBLEMA DAS CACHES DE ATRIBUTOS

A Figura 4.2 ilustra a arquitetura que será abordada nesta tese. Sendo assim, o foco deste trabalho é uma aplicação IoT executada sobre um grande campus universitário. Esse campus é proprietário completo da aplicação IoT e possui um servidor em nuvem para armazenar todas as identidades dos usuários e dispositivos, ou seja, sob a perspectiva do método de autorização, a nuvem desempenha o papel de um PIP confiável para todas as requisições de atributos. Adicionalmente, essa aplicação também espera diversos usuários e dispositivos móveis distribuídos em uma grande área geográfica requisitando acesso aos múltiplos recursos oferecidos pelo campus. Logo, o método de autorização dessa aplicação deve ser capaz de lidar com um grande número de transações e oferecer uma baixa latência no processo de decisão.

Para dar suporte a esses requisitos e oferecer uma alta taxa de acessos, diminuir a latência do processo de decisão de um acesso é fundamental. Em um primeiro momento, o campus possuía diversos nós em névoa para interceptar os pedidos de acesso (PEPs) e tomar as decisões (PDPs). Porém, toda avaliação de política necessitava de uma requisição ao servidor de nuvem. Devido aos impactos que essa requisição adicionava no método de autorização, o campus disponibilizou diversos outros nós de névoa para atuar como PIP e reduzir a latência na disponibilização do atributos. É importante observar que, embora o foco desta tese seja essa aplicação específica, a solução proposta tem aplicabilidade em diversos outros cenários. Várias outras aplicações IoT, como as relacionadas à saúde (Ray et al., 2019) e cidades inteligentes (Hossain et al., 2018), utilizam nós de névoa e apresentam apertados requisitos de latência de comunicação, que podem ser diretamente afetados pelos atrasos causados pelo método de autorização.

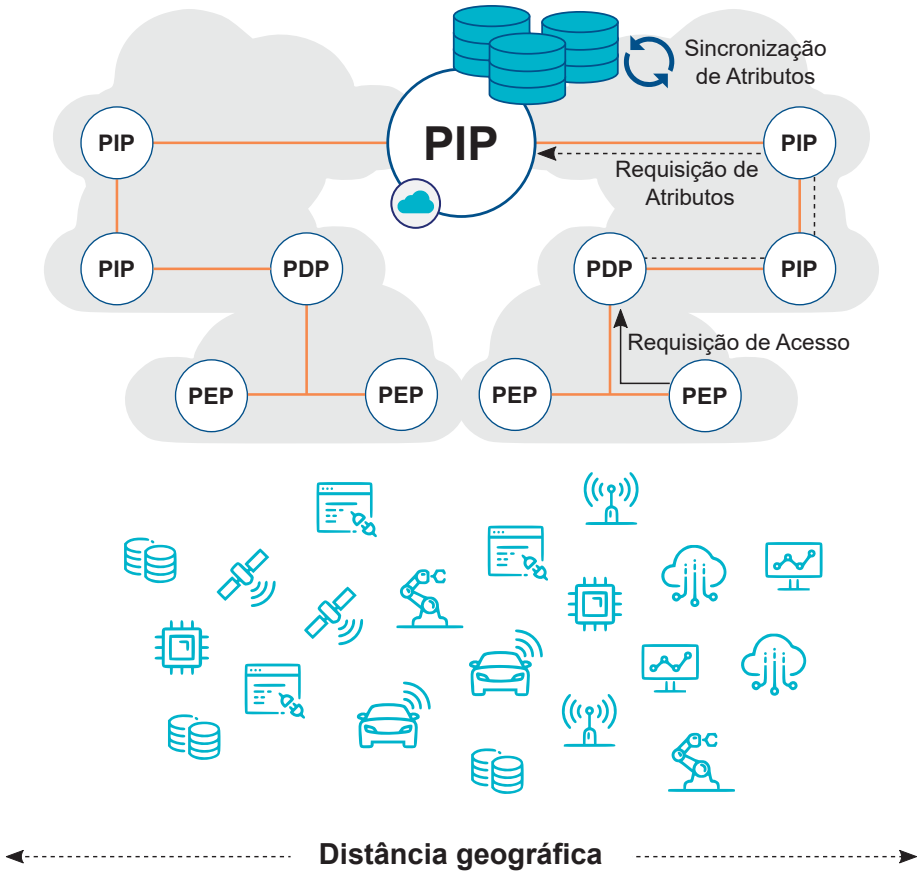


Figura 4.2: Arquitetura Proposta

Nesta tese, o conceito de nós de névoa segue o que foi definido no relatório estabelecido pela CISCO em 2015 (Systems, 2015). Neste relatório, uma série de dispositivos com potencial para se tornarem nós de névoa foi apresentada, incluindo roteadores, *switches*, pontos de acesso, entre outros. Portanto, esta tese considera que todos os roteadores e *switches* presentes na arquitetura possuem a capacidade de replicar, armazenar e manter de forma segura um subconjunto de atributos das identidades mais próximos ao ponto de decisão. Porém, ao adotar essa estratégia de criar caches de atributos, o PIP se fragmenta e o custo para manter esses atributos seguros e consistentes aumenta a cada nova réplica gerada. É importante destacar que uma nova réplica por si só não assegura melhora no desempenho da recuperação de atributos, é necessário que ela esteja próxima ao ponto de decisão no momento de avaliação de políticas. Ou seja, se durante uma solicitação de atributos os mesmos não forem encontrado, isso representa um *miss*, e essa requisição deve ser encaminhada para outro nó. Como consequência, a avaliação da política de acesso sofre impacto na latência para recuperar os atributos. Por exemplo, na Figura 4.1, o nó f_n é responsável pela avaliação de acesso e ocorreram diversos *misses* na solicitação de atributos até chegar à nuvem r. Como resultado, a latência é afetada pela distância d entre esses dois nós.

Consequentemente, o problema das caches de atributos consiste em encontrar um subconjunto de atributos para ser armazenado em cada nó de névoa, com o objetivo de minimizar o número de *misses* e, assim, reduzir a distância d. Considerando que esses acessos podem ocorrer de diferentes pontos e momentos, identificar quais atributos replicar, quando replicar e onde replicar se torna um desafio. Isso se dá especialmente ao considerar a necessidade de

encontrar um equilíbrio entre o benefício de uma réplica e o custo de mantê-la atualizada e segura, uma vez que, para cada nova replicação, o custo para manter a consistência do sistema aumenta.

4.4 CARACTERIZAÇÃO DE ACESSO EM UMA APLICAÇÃO IOT EM UM CAMPUS

Esta seção apresenta e analisa uma caracterização do conjunto de dados utilizado. Seu objetivo é compreender os efeitos da recuperação de atributos e como um método de cache pode melhorar o desempenho em um cenário de campus.

4.4.1 Metodologia

Para entender como o armazenamento em caches de atributos pode diminuir a latência na recuperação de atributos em um grande campus universitário, foi necessário realizar a coleta de dados reais de uma aplicação IoT. Especificamente, o objetivo central desta coleta foi capturar os padrões de acesso e mobilidade dos usuários que estão conectados nessa aplicação popular do campus universitário. Sendo assim, o conjunto de dados capturados consiste em informações dos acessos de alunos, funcionários administrativos e docentes em uma renomada universidade na América do Sul, com uma população diária de cerca de 20 mil usuários. É importante notar que, para entender o comportamento desses acessos, foi necessário monitorar o sistema de autorização da universidade por um período de quatro meses, em 108 pontos de acesso diferentes e distantes geograficamente. Esses dados foram obtidos através da análise de registros e incluíram, para cada requisição de acesso, o momento do evento, a identidade do usuário e os atributos necessários para avaliar essa solicitação de acesso. Em resumo, essa coleta considerou que as estações de trabalho com conexão cabeada em escritórios de pesquisa e pontos de acesso sem fio distribuídos pelo campus são as fontes das requisições de acesso. Ao total, foram monitorados 36 mil usuários únicos que efetuaram, ao todo, 9,6 milhões de solicitações de acesso durante esses quatro meses.

Na aplicação analisada, o mecanismo de autorização avalia cada pedido de acesso sempre que um usuário se conecta a um ponto de acesso (AP). Para isso, foram coletados registros desses pontos de acesso, que incluem informações sobre o momento de conexão e desconexão de cada usuário. Esses registros foram convertidos de texto simples para uma base de dados estruturada, contendo dados como o horário da requisição, a identificação do AP, a identificação do usuário e as regras de autorização utilizadas.

Entre os atributos coletados, destacam-se as operações de associação e desassociação, relacionadas ao padrão IEEE 802.11, além de respostas a requisições do Wi-Fi Protected Access (WPA), do Extensible Authentication Protocol (EAP) e de acessos negados. Neste trabalho, foram consideradas apenas as funções de associação e desassociação do usuário, com o objetivo de identificar os locais mais adequados para armazenar os objetos de atributos dos usuários, de forma que fiquem o mais próximo possível deles na infraestrutura de névoa. Para atender aos requisitos da Lei Geral de Proteção de Dados (LGPD), os dados nominais dos usuários foram anonimizados durante a etapa de conversão, tornando irreversível sua associação com os dados reais. Com os dados devidamente anonimizados e convertidos, foi possível extrair novas informações da base, como o tempo de permanência de cada usuário em um ponto de acesso (AP), o intervalo entre solicitações de acesso consecutivas em diferentes APs e a quantidade total de APs aos quais o usuário se conectou ao longo de quatro meses.

Com base nessas informações, foi realizado o rastreamento da duração de permanência dos usuários em cada ponto de acesso, o intervalo entre pedidos de acesso subsequentes em diferentes pontos de acesso e o número total de pontos de acesso utilizados pelos usuários ao longo de um dia. Durante a análise do comportamento de acesso e mobilidade, foram obtidos

alguns insights que demonstram como o armazenamento em cache de atributos pode acelerar a tomada de decisões na aplicação. Assim, ao longo do texto, alguns comentários correlacionam o comportamento dos usuários ao potencial das caches de atributos propostas.

4.4.2 Caracterização

O primeiro passo para entender o cenário é identificar o padrão de acesso e de mobilidade no campus. Portanto, nossos resultados se concentram em identificar quando, onde e como a maioria dos acessos ocorre. Para iniciar nossa análise, deve-se olhar para o comportamento de acesso de forma geral e observar os horários de pico de nossa aplicação. A Figura 4.3 mostra a porcentagem de solicitações de acesso que ocorrem em um dia útil aleatório no campus. O número de solicitações de acesso é maior durante o horário comercial, já que a maioria dos acessos ocorre entre 9h e 18h. Assim, como 80% das solicitações de acesso ocorrem durante essa janela de tempo, todas as análises realizadas consideram esses acessos.

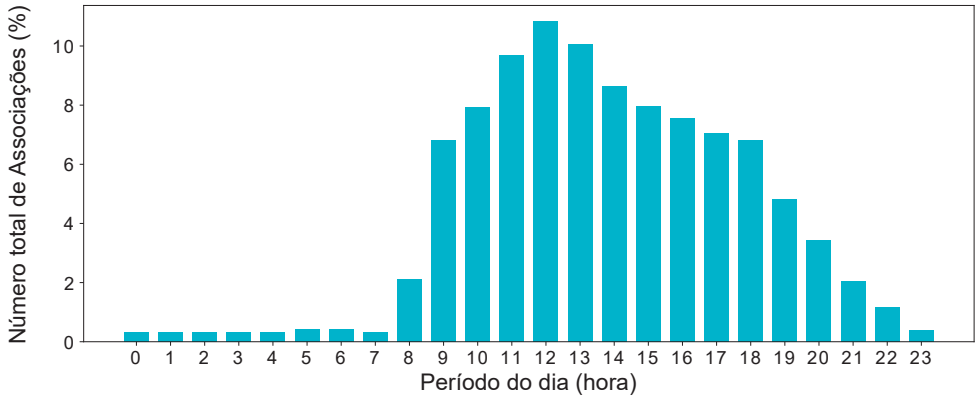


Figura 4.3: Comportamento dos Acessos

O próximo objetivo é determinar de onde e como essas solicitações de acesso vieram dentro desse intervalo de tempo. Portanto, foi inspecionado cada usuário individualmente e foi detectado que uma parte significativa dos usuários solicita acesso de locais diferentes ao longo do dia. Não há surpresa nesse padrão, uma vez que a maioria deles são estudantes que têm uma rotina diária bem definida, se deslocando entre aulas em vários locais espalhados pelo campus e outros pontos de interesse, como departamentos, bibliotecas e cafeterias.

Como os pontos de acesso cobrem quase toda a universidade, a análise desses dados permitiu encontrar um comportamento interessante em relação ao tempo de conexão. Geralmente, um único usuário conhece vários pontos de acesso em uma rota entre dois pontos de interesse. No entanto, os usuários tendem a permanecer conectados apenas por curtos períodos nesses pontos de acesso. Em média, cada ponto de acesso nessas rotas conhece cerca de 6 mil usuários, e esses usuários permanecem conectados por aproximadamente 5 minutos. Quando um usuário se desconecta, pode reconectar-se dentro de 3 minutos e geralmente usa um ponto de acesso adjacente, caracterizando padrões de rota e de mobilidade.

Esse comportamento é oposto para pontos de acesso mais próximos de salas de aula e outros pontos de interesse. No caso, esses pontos de acesso conhecem uma parte significativa dos usuários, já que são pontos de encontro para estudantes, professores e funcionários administrativos, mostrando tempos de conexão mais longos em horários específicos do dia. Por exemplo, a Figura 4.4 mostra o comportamento de acesso de dois pontos de acesso diferentes. O primeiro está em uma rota entre dois pontos de interesse e tem um comportamento mais estável, com um número constante de usuários conectados. Esse comportamento ocorre porque esse ponto de

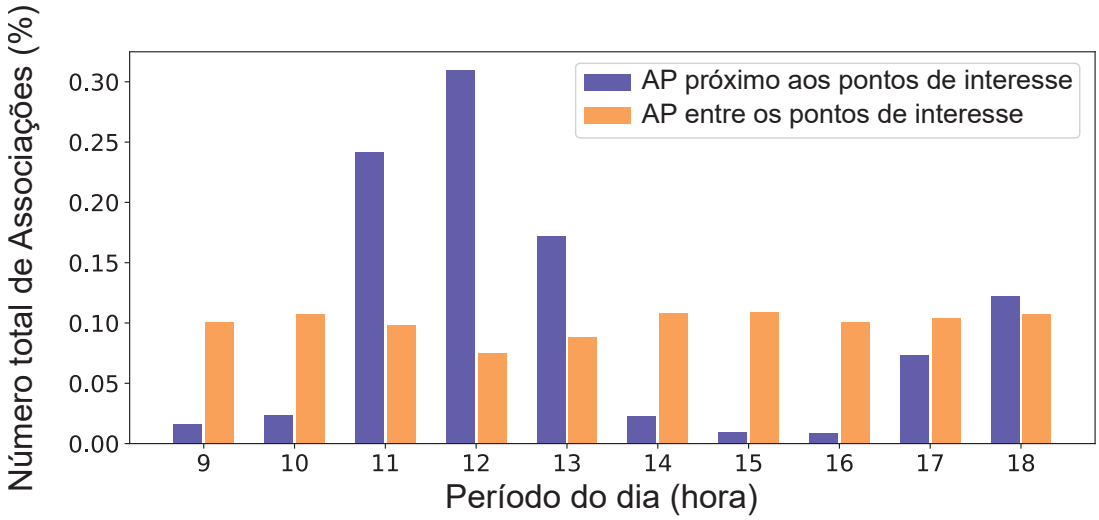


Figura 4.4: Comportamento dos Pontos de Acesso

acesso está localizado em uma rota entre pontos de interesse, constantemente utilizada, mas não como destino final da rota dos usuário. Para o segundo ponto de acesso, observamos que ele está localizado mais perto de uma cafeteria (um ponto de interesse), o que significa que esse ponto de acesso registra a atividade de vários usuários. Como resultado, seus picos de carga correspondem ao horário do almoço e do jantar. Esse mesmo comportamento aparece perto de salas de aula e departamentos em outros horários do dia.

Nesse cenário, com esses dois tipos de pontos de acesso, surge os *insights* de duas estratégias de cache: armazenar os atributos nos pontos de interesse ou tentar prever o movimento do usuário e carregar seus atributos conforme seu deslocamento. A primeira abordagem é mais simples, pois permite reagir ao padrão de acesso e posicionar os atributos mais próximos dos pontos de acesso em locais de interesse por todo o campus. No entanto, embora essa estratégia possa funcionar durante grande parte do dia (afinal usuários costumam ficar bastante tempo nos pontos de interesse), ela pode ficar um passo atrás quando o usuário se desloca no campus, já que os atributos necessários para a avaliação da política ficam disponíveis somente após a movimentação do usuário acontecer. Isso significa que os atributos estão constantemente atrasados em relação ao movimento do usuário e não podem ser usados para melhorar o desempenho dessa solicitação de acesso. Portanto, a segunda abordagem emprega o padrão de movimento do usuário e antecipa seus deslocamentos futuros. Em seguida, replica proativamente os dados do usuário nos caches. Apesar de parecer mais eficiente, o desempenho dessa estratégia se deteriora quando os usuários exibem mobilidade não determinística.

Nesse sentido, foi caracterizado a diversidade de padrões de mobilidade encontrados no campus universitário. Por exemplo, a Figura 4.5 ilustra quantos pontos de acesso um usuário se conecta ao longo de sua vida útil. A seguir, utiliza-se essa distribuição para traçar perfis de mobilidade e definir o quão custoso é prever sua mobilidade.

De acordo com a Figura 4.5, cerca de 10% dos usuários do campus conectam-se apenas a um único ponto de acesso. Em outras palavras, esses usuários praticamente não apresentam um padrão de mobilidade e sua localização futura é fácil de determinar. Por esse motivo, rotula-se os usuários desse grupo como “usuários fixos” e eles são excluídos da análise, uma vez que o problema de armazenamento em cache de atributos é trivial para eles. Assim, após excluir os usuários fixos, o primeiro quartil conecta-se a 2 a 4 pontos de acesso. Portanto, os usuários desse grupo apresentam um padrão de mobilidade pequeno, que é um padrão direto que ajuda na abordagem proativa. Por esse motivo, o método rotulou esses usuários de “baixa mobilidade".

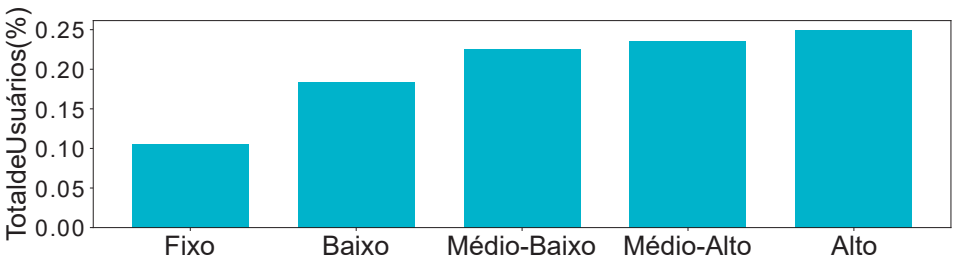


Figura 4.5: Perfil de Mobilidade dos Usuários.

O segundo quartil da figura apresenta usuários que se conectam a uma média de 5 a 12 pontos de acesso. Esses usuários se movem pelo campus. No entanto, na maioria dos casos, os usuários desse grupo ainda têm um caminho organizado e fácil de conhecer, o que também pode favorecer a abordagem proativa. Por essa razão, eles são classificados como usuários de "mobilidade média-baixa". O terceiro quartil tem usuários com movimentos mais confusos e difíceis de determinar. Logo, eles são classificados como "usuários de mobilidade média-alta". Nesse caso, os usuários se conectam, em média, a 13 a 28 pontos de acesso. Note que esse padrão de movimento é mais aleatório do que os apresentados anteriormente, o que pode tornar a abordagem de cache proativa mais difícil. Finalmente, o último quartil abrange os usuários que apresentam um padrão de acesso quase caótico, ou seja, conectando-se a pelo menos 29 pontos de acesso. Para esse caso, fica claro que a abordagem de cache proativa deve depender de um algoritmo de previsão de mobilidade altamente complexo e preciso para atender a esses usuários e entregar os atributos de forma eficiente.

4.5 MECANISMOS DE DISTRIBUIÇÃO DE ATRIBUTOS

Esta seção apresenta o mecanismo oportunístico de distribuição de atributos em caches. O método proposto é oportunístico pois, cada requisição de acesso, é uma oportunidade para replicar atributos e se adaptar ao comportamento do usuário. Consequentemente, nosso mecanismo visa minimizar o tempo de recuperação dos mesmos em solicitações de acesso subsequentes. O mecanismo proposto possui duas abordagens que se complementam para replicar os atributos nos nós de névoa: a abordagem reativa de distribuição de atributos e a abordagem proativa de distribuição de atributos.

4.5.1 Mecanismo Reativo de Distribuição de Atributos

A abordagem que contém o mecanismo reativo de distribuição de atributos utiliza a resposta de uma solicitação de atributos para criar e armazenar as réplicas dos atributos solicitados. Conforme ilustrado na Figura 4.6, suponha que uma solicitação de acesso ocorra no nó de névoa f_n e os atributos foram encontrados somente em r. Nesse caso, todo nó de névoa presente no percurso da resposta da requisição de atributos $(f_1, ..., f_{n-1}, f_n)$ tem a oportunidade de replicar esses atributos e armazená-los. Para determinar quando utilizar essa resposta para replicar, esta tese explora três estratégias distintas, são elas: ***Leave a Copy Everywhere*** (LCE), ***Leaf a Copy Down*** (LCD) e ***Leave a Copy at Leaf*** (LEAF).

Leave a Copy Everywhere é considerada a estratégia mais ambiciosa das três. Nela, todos os nós no caminho entre o nó solicitante f_n e o nó que possui os atributos r replicam o atributo e armazenam uma cópia em seu PIP. Logo, em relação ao número de cópias, para cada solicitação de atributos são criadas na ordem de $O(n)$ cópias, ou seja, conforme ilustrado na Figura 4.6,

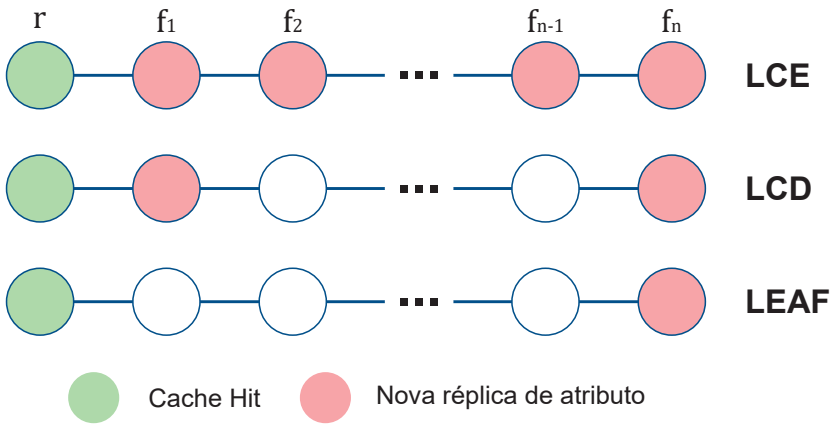


Figura 4.6: Estratégias reativas de cache de atributos

todos os nós (f_1, ..., f_{n-1}, f_n) criam e armazenam uma cópia dos atributos respondidos em sua memória. Portanto, uma vez que essa estratégia sempre gera uma cópia do atributo solicitado em todos os nós, ela é considerada a estratégia mais expansiva e cria bastante redundância no sistema. Embora essa redundância possa parecer ótima para diminuir a latência para encontrar os atributos necessários para a tomada de decisão, ela aumenta consideravelmente o custo de manter todos esses atributos seguros e sincronizados. Adicionalmente, visto que a capacidade de armazenamento desses nós é limitada, gerar réplicas de atributos de forma direta pode gerar uma maior rotatividade na cache e fazer com que o atributo não esteja no local mais adequado para reduzir o d quando for necessário.

Leave a Copy Down é uma estratégia na qual as réplicas dos atributos se movem de forma gradual em direção ao local da requisição de atributos. Nessa estratégia, durante a resposta do parceiro r ao solicitante f_n, apenas o primeiro nó em seu caminho cria uma réplica e armazena os atributos. Essa estratégia possui ordem $O(1)$ réplicas de atributos e, conforme apresentado na Figura 4.6, apenas o nó f_1 replica os atributos necessários em seu PIP. Se esses atributos necessários forem solicitados novamente em uma solicitação subsequente de atributos e esses atributos estiverem na cache do nó f_1, apenas o nó f_2 os replicará. Essa estratégia é considerada mais conservadora e implica que os atributos se movam gradualmente em direção ao solicitante f_n, um passo de cada vez para cada solicitação subsequente de atributos. Assim, nessa estratégia, enquanto os atributos mais utilizados permanecem mais próximos do usuário (o que diminui a distância d para solicitações subsequentes), os menos necessários permanecem distantes, o que implica em mais espaço para os nós mais próximos armazenarem os atributos mais populares sem tanta rotatividade, o que pode reduzir a latência do processo de requisição de atributos.

Leave a Copy at Leaf é considerada a mais direta das três. Essa estratégia também possui ordem $O(1)$ réplicas de atributos e adota uma estratégia de “tudo ou nada", ou seja, as caches fornecem, para cada solicitação, ou o melhor ou o pior desempenho na resposta da requisição de atributos. Isso ocorre pois, durante a resposta do nó com os atributos requisitados r ao solicitante f_n, apenas o último nó em seu caminho replica os atributos. Nesse sentido, na Figura 4.6, apenas o nó f_n, que é o nó solicitante, replica os atributos necessários no PIP. Se os atributos necessários ainda estiverem em cache para uma subsequente solicitação de atributos, essa estratégia fornecerá o melhor desempenho; caso contrário, será necessário recorrer à nuvem.

4.5.2 Mecanismo Proativo de Distribuição de Atributos

Esta tese considera que os usuários podem solicitar acesso aos recursos de diferentes locais ao longo do tempo. Logo, o mecanismo proativo de distribuição de atributos também tem o objetivo de encontrar um subconjunto de atributos a serem replicados e armazenados em diferentes nós da névoa para minimizar o número de *misses* na operação de decisão de política. Em experimentos preliminares, foi observado que o método reativo por si só não era eficiente e sempre ficava um passo atrás quando o usuário se movia dentro do campus. Por exemplo, a Figura 4.7 apresenta a trajetória do usuário que sai de um ponto de interesse f_1 e caminha em direção ao ponto de interesse fn. Para esse exemplo, o resultado da operação de recuperação de atributos à medida em que ele se move é apresentado para cada instante de tempo t_n. Nesse experimento preliminar, caso o nó não possuísse o atributo necessário, o resultado era caracterizado como *miss*, enquanto que, se o atributo estivesse presente, o resultado era caracterizado como *hit*. É importante destacar que foi observado que a mudança do usuário de um ponto de acesso para outro resultava em um miss, e, consequentemente, gerava uma latência no processo de decisão.

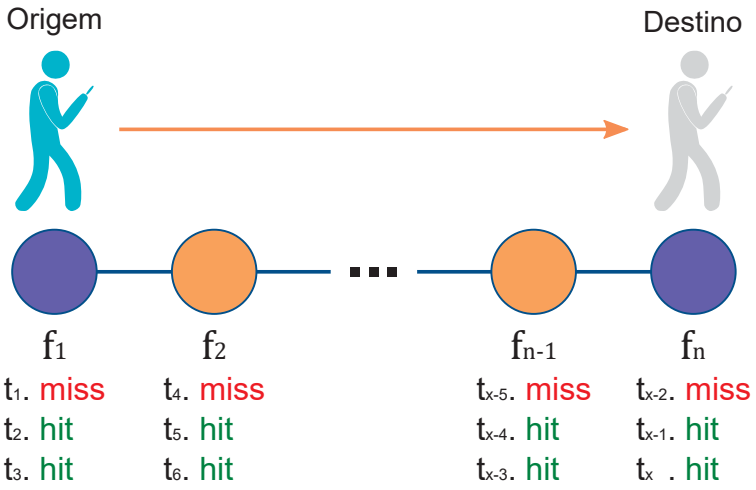


Figura 4.7: Comportamento da estratégia somente reativa

Para evitar essa situação, foi necessário criar um método proativo que pudesse antecipar as necessidades de acesso do usuário e armazenar previamente os atributos relevantes em diferentes nós da névoa, de modo a garantir um tempo de resposta mais rápido na operação de decisão de política. Portanto, ao contrário do mecanismo reativo de distribuição de atributos, a estratégia proativa visa prever onde ocorrerá o próximo acesso para replicar proativamente atributos nesse local, reduzindo a taxa de *miss*. Portanto, para alcançar esse objetivo, um algoritmo de previsão de mobilidade eficaz e preciso é essencial para determinar a posição da solicitação subsequente e replicar proativamente os atributos.

Na literatura, a previsão de mobilidade ainda é um tópico em aberto. Várias propostas visam criar algoritmos de previsão de mobilidade para a mobilidade humana em diferentes contextos, variando de universidades a cuidados de saúde. No entanto, o desempenho preditivo desses algoritmos na literatura varia consideravelmente; alguns oferecem precisão de mais de 90%, enquanto outros entregam menos de 40%. Portanto, uma vez que a previsão de mobilidade não é o foco deste trabalho e o desempenho pode variar amplamente, o mecanismo proativo de distribuição de atributos proposto pressupõe um oráculo que conhece o padrão de mobilidade do usuário e está ciente da localização da solicitação de acesso atual e da próxima. Assim,

nesta tese, o oráculo serve como um substituto para um algoritmo de previsão de mobilidade. É importante deixar claro que esta suposição foi feita para simplificar a modelagem e a análise para um entendimento inicial dos benefícios potenciais do mecanismo proativo. Por exemplo, em vez de usar abordagens complexas, como cadeias de Markov e redes neurais para prever a mobilidade e melhorar a latência de nossa abordagem, o oráculo as substitui como se fosse uma caixa preta que fornece a previsão correta de onde ocorrerão os acessos. Porém, dado que o desempenho das previsões de mobilidade varia na literatura, replicamos essa suposição através da construção de dois preditores de mobilidade. O primeiro é este oráculo perfeito que entrega a previsão com 100% de precisão e o segundo é uma implementação de um simples *K-nearest neighbor* (KNN) (Guo et al., 2003) aplicado à mobilidade humana, semelhante ao apresentado por (Hastari et al., 2021) com apenas 45% de precisão. O KNN, nesse caso, serve como um limite inferior do método proposto.

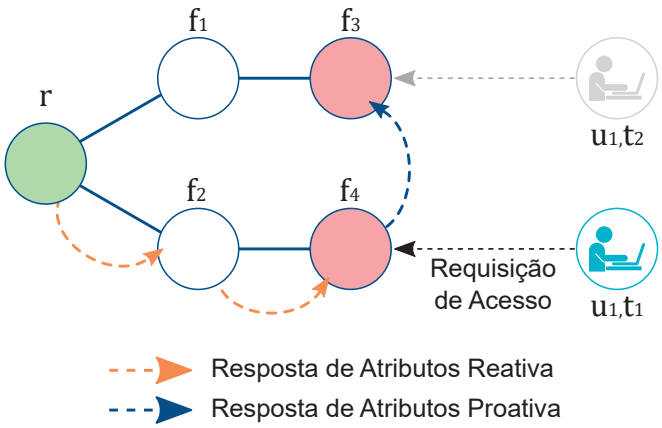


Figura 4.8: Estratégia proativa de cache de atributos

Para exemplificar essa abordagem, considere uma rede com quatro nós de névoa (f_1, f_2, f_3, f_4), um servidor de nuvem no nó raiz r e um usuário u_1, conforme mostrado na Figura 4.8. Supondo que, no tempo t_1, o usuário u_1 solicite acesso em f_3, o pedido de atributo flui do requisitante f_n para o parceiro r e retorna, permitindo que todos os nós no caminho repliquem esses atributos. No entanto, uma vez que no instante de tempo t_2, esse mesmo usuário solicita acesso ao nó de névoa f_4, a proposta replica proativamente os atributos para esse nó de névoa, caso ele não contenha esse atributo. Assim, no instante de tempo t_2, é possível fazer a solicitação de acesso (e a eventual solicitação subsequente de atributo que ela pode originar) com $d = 0$.

4.5.3 Mecanismo de Substituição de Atributos

Cada nó da névoa possui uma quantidade limitada de atributos que suporta armazenar. Logo, em algum momento durante a execução da aplicação, esses nós podem ficar cheios. Supondo que exista a necessidade de armazenar novos atributos nesse nó, já que a cache está cheia, é necessário descartar algum ou alguns atributos para ceder espaço para os novos atributos. Essa ação é chamada de substituição de atributos em cache e, nesta tese, cinco políticas diferentes para determinar qual atributo descartar foram exploradas, são elas: *First-in First Out* (FIFO), *Least Recently Used* (LRU), *Random Replacement* (RR), *Segmented Least Recently Used* (SLRU) e *Predicted Segmented Least Recently Used* (SPROT). É importante ressaltar que, conforme ilustrado na Figura 4.9, todos os algoritmos utilizados possuem a mesma complexidade; $O(1)$ para inserir um novo atributo, $O(1)$ para remover um atributo da cache, e $O(n)$ para buscar um atributo.

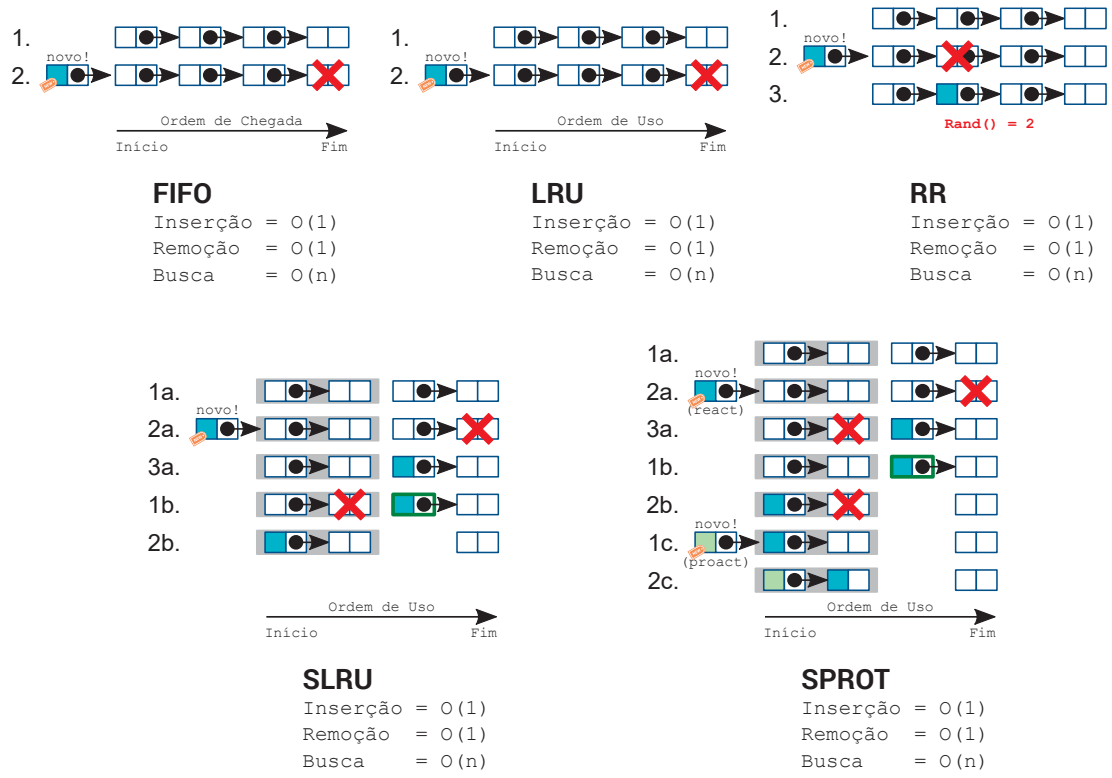


Figura 4.9: Estratégias de substituição de atributos

A primeira política de substituição de cache é a **First In First Out (FIFO)**, em que o atributo mais antigo armazenado está localizado na parte final da cache e o mais recente está situado no início (FIFO 1.). Assim, quando um novo atributo chega (FIFO 2.), descarta-se diretamente o atributo final e inserimos o novo atributo no início; ambas as operações têm a complexidade de algoritmo de $O(1)$.

A segunda política de substituição de cache é a política **Least Recently Used (LRU)**, em que o atributo mais recentemente usado está localizado no início da cache e o menos recentemente usado está localizado no final (LRU 1.). Logo, para abrir espaço para um novo atributo, descarta-se o atributo no final, que é o menos recentemente usado, e inserimos o novo atributo no início da cache (LRU 2.); nesse caso, ambas as operações têm a mesma complexidade $O(1)$.

A terceira política é a **Random Replacement (RR)**, que descarta um atributo aleatoriamente. Nesse caso, mesmo sem uma ordenação direta da cache, um atributo aleatório é descartado (RR 1. e 2.), e o novo atributo ocupa seu lugar (RR 3.). Como ambas as operações ocorrem diretamente por meio de uma escolha aleatória, a substituição aleatória tem o mesmo $O(1)$ para excluir um atributo e $O(1)$ para incluir o novo.

A quarta política é a **Segmented Least Recently Used (SLRU)**, que é uma variação do clássico LRU. Essa política de substituição separa a cache em dois segmentos: o segmento de oportunidade e o segmento protegido (SLRU 1a.). Ambos os segmentos são classificados pelo menor uso, o que significa que o início é o atributo mais recentemente utilizado e o final é o atributo menos recentemente utilizado. O segmento de oportunidade armazena todos os novos atributos e, se esse segmento ficar cheio, segue o padrão LRU padrão, excluindo o atributo final para dar lugar ao novo (SLRU 2a. e 3a.). No entanto, se algum atributo do segmento de oportunidade for acessado, ele é promovido para a parte protegida, tornando-o mais difícil de

ser descartado (SLRU 1b. e 2b.). Quando o segmento protegido fica cheio e outro atributo é promovido, o SLRU se comporta como a política LRU padrão. Como ainda estamos usando as mesmas operações de exclusão e inserção do LRU, a complexidade do SLRU é $O(1)$ para inserir e $O(1)$ para excluir um atributo.

A última política é a **Predicted Segmented Least Recently Used (SPROT)**, que é uma pequena variação do SLRU. A cache também é dividida em segmentos de oportunidade e protegido, a ordenação por utilização é aplicada a ambos os segmentos (SPROT 1a.). No entanto, no SPROT, o comportamento da substituição muda se o novo atributo vier das estratégias reativas ou proativas de replicação de atributo. Para cada novo atributo que é criado da abordagem reativa, o SPROT age como um simples SLRU, excluindo o atributo final e armazenando o novo no segmento de oportunidade (SPROT 2a. e 3a.). Se algum atributo do segmento de oportunidade for acessado, esse atributo será promovido ao segmento protegido (SPROT 1b. e 2b.). Se esse novo atributo vier da abordagem proativa, por outro lado, ele é inserido diretamente no segmento protegido. Em outras palavras, se não houver espaço no segmento protegido, o atributo final desse segmento é descartado e o novo atributo é inserido (SPROT 1c. e 2c.).

4.6 MECANISMO DE SINCRONIZAÇÃO DE ATRIBUTOS

Nesta seção, apresenta-se o mecanismo de sincronização bidirecional de atributos. Basicamente, discute-se como mapear e sincronizar todos os atributos dos usuários em múltiplas caches presentes em *switches*, roteadores e pontos de acesso. Para isso, para cada atributo presente no PIP, o mecanismo de sincronização mapeia a sua localização e a de suas réplicas, mantendo um controle rígido sobre o atual estado de consistência do sistema e, consequentemente, reduzindo a exposição dos PDPs aos atributos desatualizados. De maneira geral, ainda que o subconjunto dos atributos do PIP esteja replicado em múltiplas caches na rede disposta em uma topologia de árvore, nosso método é capaz de oferecer uma sincronização dos atributos em toda a rede para permitir um ambiente consistente. É importante notar que, nesta tese, é considerada a implementação do mecanismo de sincronização bidirecional de atributos em dispositivos mais robustos. Logo, é considerado que esses dispositivos irão possuir maior capacidade de processamento, memória e fontes de energia estáveis. Em cenários onde os dispositivos forem de baixo custo ou sensores com capacidade limitada, a abordagem proposta nessa seção pode ser impraticável.

4.6.1 Mapeamento de Atributos

Em nosso modelo de sistema, os atributos e suas cópias são distribuídos em um PIP apoiado por caches em uma rede com topologia de árvore. Para tornar os atributos consistentes, assume-se que o nó raiz da árvore tem a função de controlador dos atributos. Essa entidade é chamada de gerenciador de atributos nesta tese e sua função é não só oferecer atributos, mas também monitorar suas cópias espalhadas pela rede e mantê-las atualizadas. Conforme apresentado anteriormente, a raiz contém uma base de dados atualizada com todos os atributos e oferece uma visão global e combinada de todos os atributos. Cada atributo presente nesse gerenciador possui um valor e um número de versão. Para ilustrar sua função, a Figura 4.10 apresenta um exemplo de sua utilidade. Note que a cache da direita (1) possui a identidade com identificador “user01” e essa identidade tem o atributo “nome” com o valor “Alex” na versão 0. Da mesma forma, a cache da esquerda (2) também possui a identidade com identificador “user01”, mas em vez de “nome”, essa identidade tem o atributo “cargo” com o valor “Professor” na versão 0. Como várias caches não têm visão completa dos usuários e atributos podem estar faltando em

outros, o gerenciador de atributos permite uma visão geral do usuário. Portanto, o gerenciador de atributos possui os atributos de ambas as identidades e cria uma identidade completa com os atributos “nome"e “cargo", também na versão 0. Além disso, o gerenciador de atributos cria uma base que aponta em quais caches a identidade está armazenada. Se essa identidade for removida da cache ou adicionada a outras, essas caches devem enviar uma mensagem de atualização do mapa de atributos para o gerenciador.

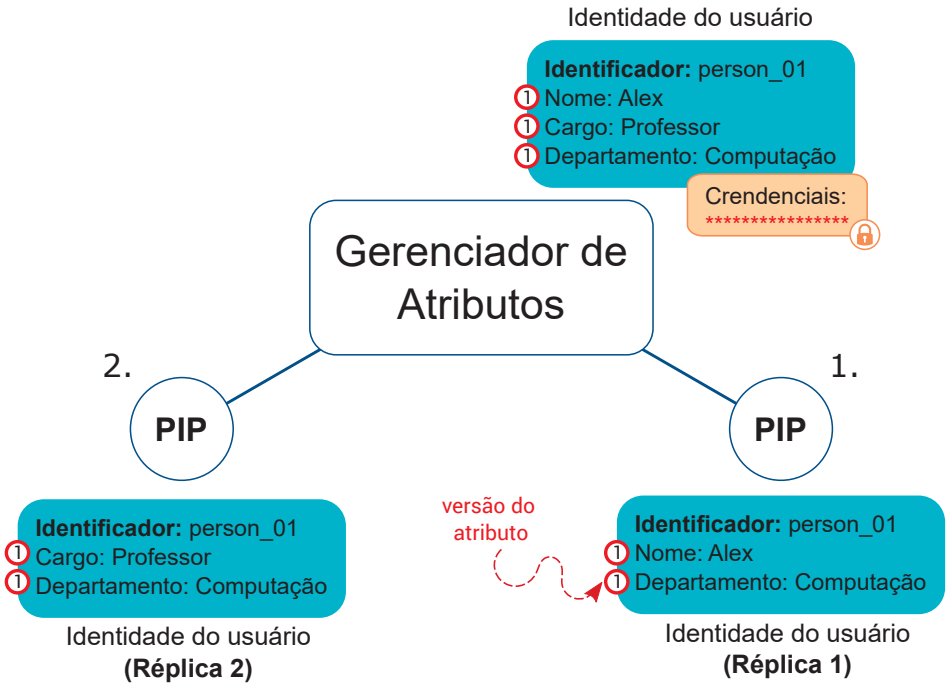


Figura 4.10: Mapeamento de Atributos

4.6.2 Sincronização de Atributos

Como os atributos são mutáveis, qualquer cache pode atualizá-los. No entanto, para refletir essa mudança em outras caches, é necessário sincronizá-los. Nesta tese, propõe-se um mecanismo de sincronização bidirecional de atributos. O mecanismo recebe esse nome porque, inicialmente, a atualização é enviada para o gerenciador de atributos (atualização ocorre “para cima") e o gerenciador, com base em seu mapa de atributos, atualiza todas as *caches* (atualização ocorre “para baixo"). Para exemplificar esse processo, a Figura 4.11 ilustra uma atualização de atributos. Suponha que o usuário “user01"tem sua identidade replicada em ambas as caches (1 e 2). Suponha que, devido a alguma mudança operacional, o cargo desse usuário mudou de “professor"para “pesquisador"e essa mudança ocorreu na cache 2 (a). A cache 2 atualiza a sversão do atributo (de 0 para 1) encaminha essa mudança para o gerenciador de atributos (b), que atualiza a visão global do sistema (c). Após atualizar a visão global, ele busca em seu mapa de atributos em quais caches esse atributo foi armazenado e os atualiza (d).

É importante notar que o processo de sincronização ocorre por meio de uma operação básica. Existe uma cache que é a fonte da mudança (ou seja, o PIP ou a cache em que um atributo foi alterado) e outra que é alvo para sincronizar a mudança (o PIP ou a cache em que a mudança será propagada). Note que o mapa serve como guia para essa operação. Toda *cache* possui um mapa que aponta para o gerenciador de atributos e o gerenciador, por sua vez, possui um mapa com todas as *caches* em que aquele atributo está presente. Caso o gerenciador de atributos receba

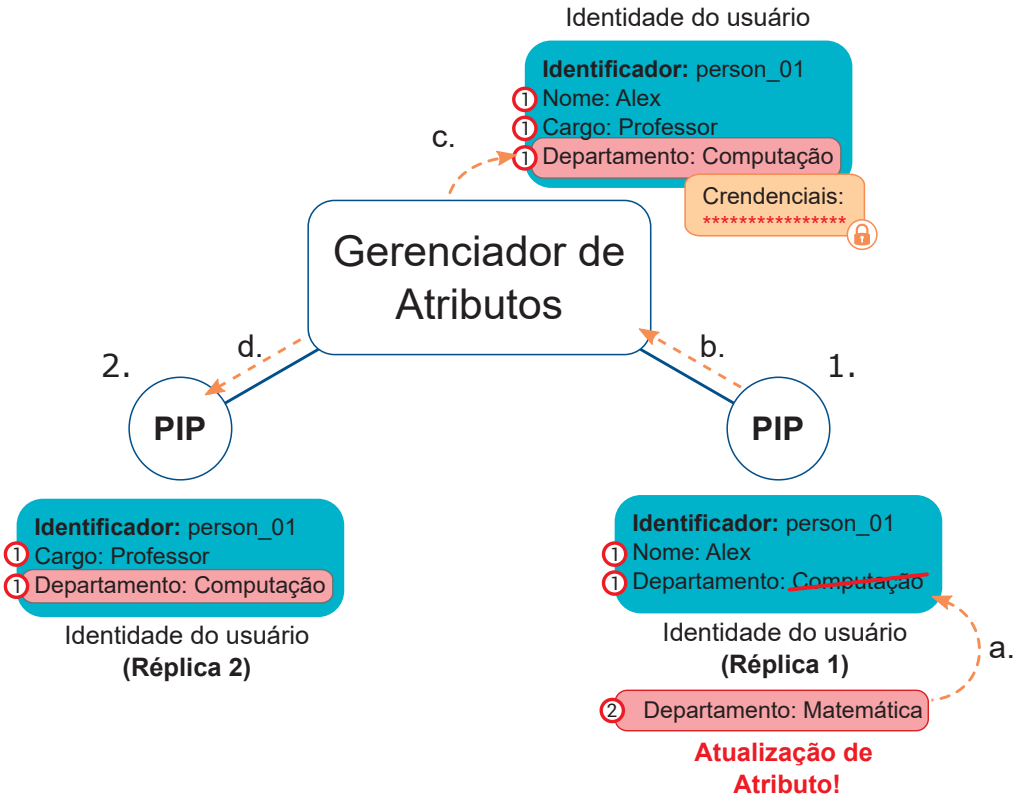


Figura 4.11: Sincronização de Atributos

duas ou mais modificações com a mesma versão mas valores diferentes, determina-se como um conflito e um operador humano deve determinar qual a versão correta da modificação.

Nesta tese, o mecanismo de sincronização dos atributos pressupõe que o gerenciador de atributos é capaz de fornecer um mapeamento preciso da localização de todos os atributos. Assim, desde que não ocorram falhas na rede nem dificuldades no envio de mensagens, o nosso método garante que os atributos no PIP estarão sincronizados nas caches e, consequentemente, nos pontos de decisão para tomada de decisão e revalidação de acessos. No entanto, não há garantia de que esses atributos não tenham sido modificados ou atacados nas caches. Esse cenário é considerado fora do escopo do nosso trabalho.

4.6.3 Revalidação de Acessos

Nesta tese assume-se que quando um PDP toma uma decisão de acesso, ele mantém um histórico dessa permissão (P), que apresenta qual regra de acesso foi atendida e, consequentemente, quais atributos foram utilizados para tomar a decisão ($P = a1, a2, ...$). Portanto, após uma alteração de atributos, o PIP anuncia para o PDP uma mudança e os acessos são reavaliados. As permissões possuem três estados neste trabalho: válido, desconhecido e revogado. Assim, em vez de avaliar a política de acesso como um todo, em um primeiro momento, apenas a permissão é avaliada.

Ao considerar a mutabilidade dos atributos, a Figura 4.12 ilustra como ocorre a mudança do estado das permissões quando o ponto de decisão recebe uma mensagem informando que um determinado atributo foi atualizado. Se o atributo modificado não é usado na permissão de acesso, a permissão permanecerá **válida**. Se o atributo é utilizado, mas a decisão da regra de acesso não é alterada, a permissão também permanece **válida**. Se o atributo afeta a permissão de acesso a ponto de tornar a regra que concedeu a permissão inválida, a permissão se tornará

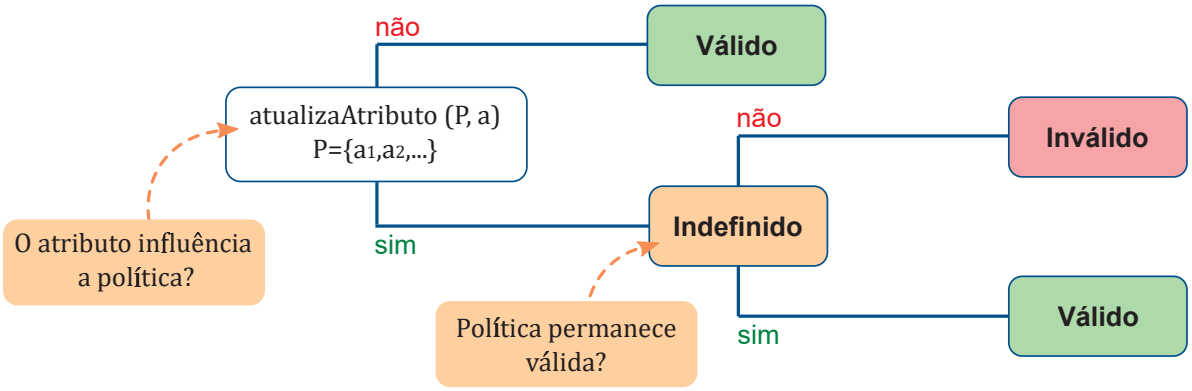


Figura 4.12: Revalidação de Acesso

desconhecida, e toda a política de acesso deve ser reavaliada usando o novo atributo. Se alguma outra regra da política de acesso for satisfeita, a permissão volta ao estado **válida** e é atualizada com a nova regra que satisfaz. Se nenhuma regra for satisfeita, ocorre uma **revogação** de acesso, que é comunicada imediatamente ao PEP, que suspende o acesso do usuário ao recurso.

4.7 ANÁLISE FORMAL

Esta seção apresenta a avaliação da cache de atributos proposta. Inicialmente, para avaliar os mecanismos de distribuição e os mecanismos de sincronização bidirecional de atributos, escolheu-se realizar uma verificação formal por meio de autômatos temporizados. Conceitualmente, esses autômatos são uma generalização dos autômatos finitos para um domínio de tempo contínuo, possuindo, além das tradicionais transições e estados, um número finito de variáveis reais chamadas relógios, cujos valores crescem com uma derivada 1 em relação à passagem do tempo. Cada transição do autômato pode ser restrita aos valores do relógio e só pode ocorrer caso uma determinada condição seja satisfeita. De forma geral, exceto pela operação de zerar, não há operações de modificação dos relógios.

4.7.1 Modelos

Nesta tese, considera-se uma aplicação IoT na qual um usuário solicita acesso a vários dispositivos, como termostatos, por exemplo, em diferentes locais (Shakarami, 2022). Para ilustrar esse cenário, considere a Figura 4.13. Suponha que Alex é um professor recém-contratado, localizado próximo ao PEP (1). Sua identidade possui três atributos: função, confiança e localização. Em um instante de tempo t_1, ele começou a trabalhar, e seu nível de confiança foi associado ao valor mais baixo, por exemplo, 1. Nessa aplicação, existe uma política de acesso que nega aos usuários todas as operações sobre dispositivos IoT se seu nível de confiança for igual a 1. Após algum tempo, um usuário superior de Alex atualiza seu nível de confiança para o valor 2 na base PIP (1). Nesse momento, Alex realiza um acesso no instante de tempo t_2 e é permitido, por exemplo, ligar/desligar o dispositivo. Agora, suponha que Alex mude sua localização para o PEP (2). Se o atributo de atualização de nível tiver sido sincronizado com o PIP (2), Alex poderá realizar um acesso no instante t_3; caso contrário, o acesso será negado incorretamente. Neste trabalho, enumeram-se as condições em que esse acesso é incorretamente negado. Essa situação é modelada a seguir.

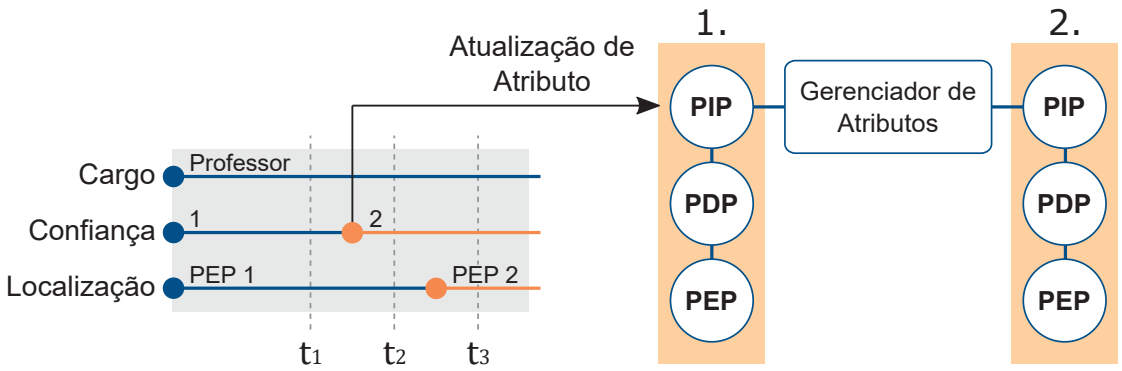


Figura 4.13: Cenário de avaliação

Para todas as comunicações que ocorrem entre os usuários e as entidades do XACML, modelamos o envio (_SENT) e o recebimento das mensagens (_RECEIVED) por meio de um autômato que modela o canal de comunicação (Figura 4.14). Resumidamente, para cada mensagem enviada (_SENT), o canal de comunicação implica em um atraso de comunicação que faz com que um determinado instante de tempo t seja passado até que a mensagem seja recebida pela outra entidade (_RECEIVED). Por exemplo, em nosso canal, quando um pedido de acesso é enviado para o PEP, o canal sincroniza essa mensagem (ACCESS_REQUEST_SENT[e]?), aguarda um instante de tempo t, e só então encaminha essa mensagem para o PEP através da mensagem de sincronização (ACCESS_REQUEST_RECEIVED[channel]!). Observe que todas as mensagens apresentadas na arquitetura XACML estão modeladas nesse canal por meio das sincronizações ACCESS_REQUEST, DECISION_REQUEST, ATTRIBUTE_REQUEST, ACCESS_RESPONSE, DECISION_RESPONSE e ATTRIBUTE_RESPONSE. Além disso, a sincronização dos atributos atualizados em um determinado PIP com o gerenciador de atributos está modelada na mensagem GA_SYNC_ATTRIBUTE.

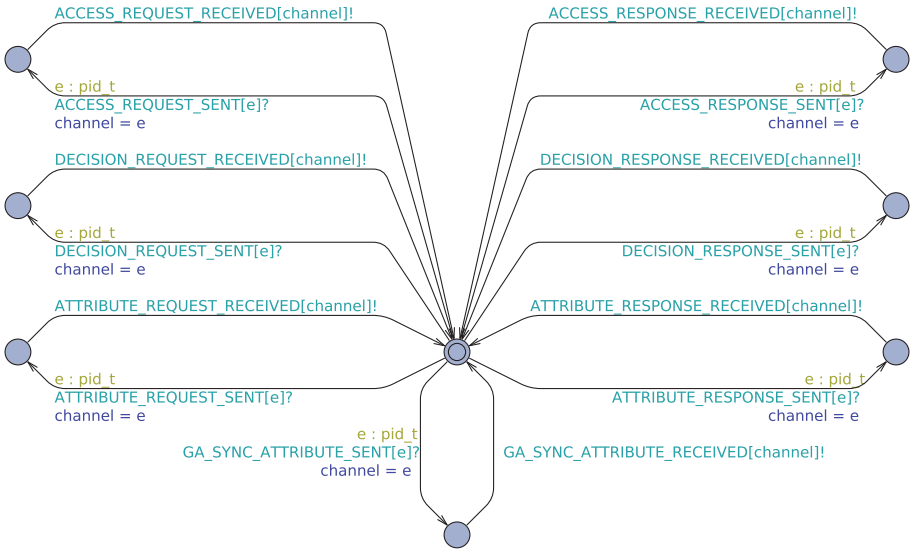


Figura 4.14: Autômato - Canal de comunicação

A Figura 4.15 ilustra o autômato que modela o PEP. Ao interceptar um pedido de acesso do cliente (ACCESS_REQUEST_RECEIVED), o autômato envia para o PDP uma requisição de decisão (DECISION_REQUEST_SENT) e aguarda o PDP responder. Após a resposta ser

recebida (DECISION_RESPONSE_RECEIVED), o PEP interpreta a resposta do PDP e a envia para o cliente (ACCESS_RESPONSE_SENT).

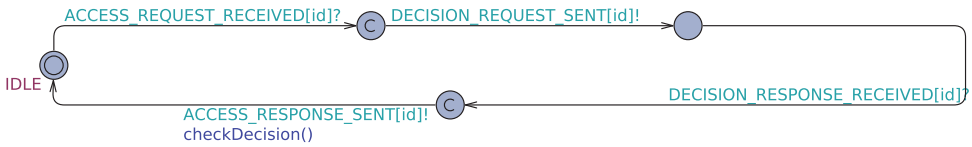


Figura 4.15: Autômato - PEP

A Figura 4.16 ilustra o autômato que modela o PDP. Ao receber um pedido de decisão do PEP (DECISION_REQUEST_RECEIVED), o autômato envia para o PIP uma requisição de atributos (ATTRIBUTE_REQUEST_SENT) e aguarda o PIP responder. Após receber os atributos pedidos do PIP (ATTRIBUTE_RESPONSE_RECEIVED), o PDP avalia a política de acesso e envia sua decisão para o PEP (DECISION_RESPONSE_SENT). É importante notar que, durante seu funcionamento, o PDP pode ser requisitado para reavaliar um acesso após uma mudança em um determinado atributo, atualizando os acessos não afetados e revogando acessos incorretos (CHECK_ACCESS_PDP), conforme apresentado na Seção 4.3.

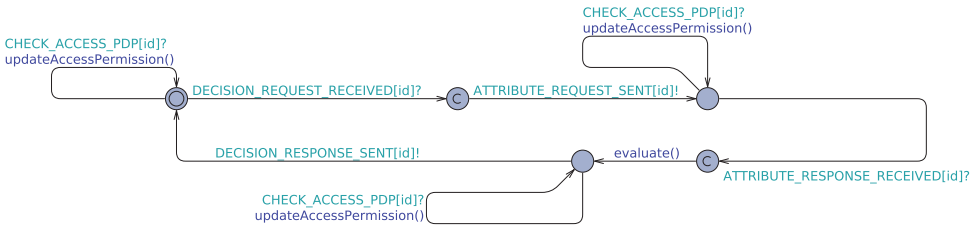


Figura 4.16: Autômato - PDP

A Figura 4.17 ilustra o autômato que modela o PIP. Nele, sua principal função consiste em aguardar uma requisição de atributos (ATTRIBUTE_REQUEST_RECEIVED) e respondê-la adequadamente (ATTRIBUTE_RESPONSE_SENT). Nesta tese, como o atributo é mutável, sua mudança ocorre diretamente no PIP (PIP_PLEASE_CHANGE_ATTRIBUTE) e desencadeia o envio de uma mensagem para o gerenciador de atributos para iniciar o processo de sincronização. É importante notar que, quando o PIP recebe uma mensagem de sincronização do gerenciador de atributos (SYNC_PIP), ele envia ao PDP um pedido de reavaliação dos acessos (CHECK_ACCESS_PDP).

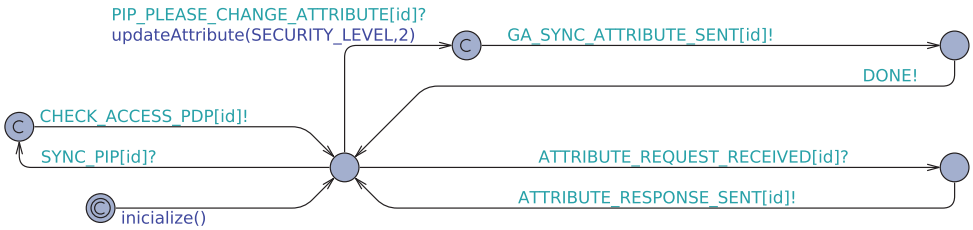


Figura 4.17: Autômato - PIP

Finalmente, para completar os modelos, a Figura 4.18 apresenta o autômato que representa o gerenciador de atributos. De forma geral, sua principal função é aguardar pedidos de sincronização de atributos (GA_SYNC_ATTRIBUTE_RECEIVED) por parte de um PIP e enviar a requisição para os demais PIPs em que o atributo está mapeado (SYNC_PIP).

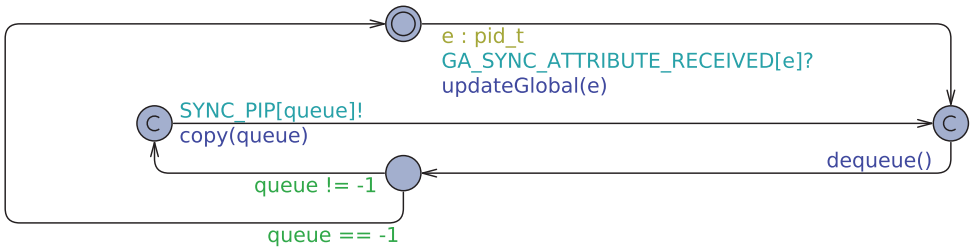


Figura 4.18: Autômato - Gerenciador de Atributos

4.7.2 Verificação Formal dos Modelos

Para iniciar a verificação formal do nosso modelo, foram definidas três propriedades básicas a serem alcançadas. Em um primeiro momento, desejamos verificar se o modelo está livre de paradas e correto. Para isso, utilizamos a seguinte expressão: $E <> deadlock$. Nos nossos testes, o limite de tempo foi de 30 minutos e não foi encontrado nenhum *deadlock*. Embora isso não prove que o modelo esteja livre de *deadlocks*, é um indicativo de que o modelo está correto. A nossa segunda expressão foi utilizada para verificar a existência de acessos indevidos no instante $t3$. Para isso, utilizamos a seguinte expressão: $E <> PEP[2] \cdot invalidAuthorization()$. Traduzindo-a: “Existe uma trajetória em que existe um acesso inválido no PEP 2?". Nesse caso, o simulador indicou a existência de acessos concedidos de forma incorreta, em que o processo de sincronização ocorre após a requisição de acesso no PEP 2. Para verificar se o nosso método revoga esses acessos corretamente, utilizamos a terceira expressão: $E <> PEP[2] \cdot invalidAuthorization() \;\&\&\; CENARIO \cdot FIM$. Nesse caso, o simulador indicou essa expressão como falsa. Isso significa que, embora existam acessos que possam ter sido concedidos de forma indevida, em algum momento, nosso método é capaz de revogá-los e, ao final da verificação, não há nenhum acesso inválido. Com isso, podemos afirmar que o nosso método é capaz de reconhecer um acesso indevido em algum momento e revogá-lo corretamente.

4.8 AVALIAÇÃO

Esta seção apresenta a avaliação do mecanismo de distribuição de atributos descrita nesta tese. Primeiramente, descreve-se a metodologia de avaliação, incluindo o cenário considerado e as métricas para avaliar o seu desempenho. Em seguida, apresentam-se os resultados para várias combinações de estratégias de cache e políticas de substituição.

4.8.1 Metodologia

A avaliação do mecanismo oportunístico de distribuição de atributos foi realizada por meio de simulações em um conjunto de dados de quatro meses que criamos para caracterizar o campus. Portanto, construímos um simulador de rede baseado em eventos que utiliza todos os registros de dados como entrada e imita todos os eventos que ocorrem em ordem direta. Nosso simulador é capaz de simular uma topologia de rede de pontos de acesso baseada em névoa e vários usuários solicitando acesso a uma aplicação específica em um determinado ponto de acesso e horário. Para todos os pontos de acesso foram instaladas uma estratégia proativa, uma estratégia reativa e uma política de substituição de cache.

A topologia de rede simulada segue uma árvore hierárquica, semelhante a um campus universitário. Em particular, as simulações consideram diferentes topologias de rede em forma de árvore hierárquica, em que variam a altura da árvore de 2 a 7. As topologias também variam com a altura de cada camada. Por exemplo, a camada inferior da rede apresenta 108 pontos

de acesso geograficamente distribuídos pela universidade em todas as topologias. A raiz da topologia de rede em forma de árvore, ou seja, a camada superior da rede, representa um servidor de nuvem privada. Todos os níveis intermediários restantes da rede, ou seja, as camadas entre a raiz da árvore e a camada inferior, apresentam 62 *switches* que interconectam toda a rede do campus. Esse cenário segue um relatório da Cisco que define o conceito de computação em névoa como dispositivos com capacidades de computação, armazenamento e rede embutidos, essenciais para facilitar a execução de aplicativos de IoT (Systems, 2015). De acordo com o relatório, um conjunto de candidatos potenciais são dispositivos de névoa, incluindo roteadores, *switches*, pontos de acesso sem fio, etc. (Systems, 2015). A Figura 4.19 ilustra isso e mostra duas topologias em uma escala menor. Ambas têm o mesmo número de *switches* e pontos de acesso. No entanto, a organização dos *switches* é diferente, implicando duas árvores com alturas diferentes, mas com o mesmo número de folhas. Enquanto o primeiro (1) tem uma altura de dois e é mais amplo, o segundo (2) tem uma altura de três e é mais estreito quando comparado ao primeiro. Portanto, este trabalho segue a mesma ideia, mas com 108 pontos de acesso e 62 *switches*.

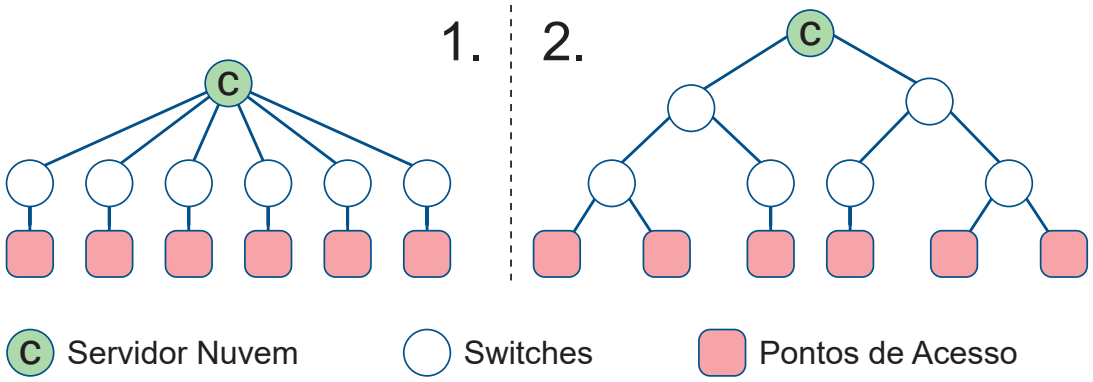


Figura 4.19: Exemplo de Topologias de Rede.

O servidor de nuvem privada armazena um único atributo para todos os 36 mil usuários exclusivos, enquanto os *switches* e pontos de acesso (nós de névoa) podem armazenar uma fração desse número. Assim, as simulações apresentaram que a capacidade de armazenamento dos nós de névoa também variará. Como desejamos avaliar um cenário com abundância e escassez de recursos, decidimos variar a capacidade de armazenamento desses nós de névoa de 0,1% a 1% do número total de atributos, o que corresponde a 36 e 360 atributos por nó de névoa, respectivamente.

A cada solicitação que um cliente realiza, avaliamos o desempenho da cache usando diferentes combinações de estratégias reativas e políticas de substituição. Portanto, as simulações consideram os cenários usando todas as estratégias de replicação (LCE, LCD e LEAF) com todas as políticas de substituição (LRU, FIFO, RR, SLRU, SPROT). Consideramos apenas a política SPROT em conjunto com nossa nova estratégia de cache de atributo oportunista pró-ativa. No entanto, como essa estratégia de cache pró-ativa depende muito da precisão do preditor de mobilidade, as simulações comparam essa política de substituição com dois preditores de mobilidade diferentes: o primeiro é uma implementação do *K-nearest neighbor* (KNN) (Guo et al., 2003) aplicado à mobilidade humana, semelhante ao apresentado por Hastari et al. (2021), em que obtemos uma precisão de apenas cerca de 45%, e o segundo preditor de mobilidade é um oráculo perfeito com 100% de precisão. Finalmente, é essencial apontar que, em Bhatt

(2018); Siebach e Jess (2021), os autores estão usando a combinação de LEAF e FIFO, ou seja, um mecanismo de cache reativo baseado em políticas.

As avaliações consideram as seguintes métricas de desempenho: Taxa de Acerto (HR), Taxa de Acerto em um Salto (OHHR) e o número de réplicas de atributos (NAR). A Taxa de Acerto é uma métrica para determinar quantas solicitações de atributos as caches conseguiram atender com sucesso, em comparação com quantas solicitações elas receberam. Portanto, essa métrica indica o quão bons são os atributos replicados localizados nos nós de *fog*. A Equação 4.1 formaliza a definição da métrica HR.

$$HR = \frac{número\ de\ cache\ hits}{número\ de\ cache\ hits\ +\ número\ de\ caches\ misses} \tag{4.1}$$

A OHHR é uma métrica para determinar quantas solicitações de atributos as caches mais próximos do usuário (ou seja, aqueles com um salto) conseguiram atender em comparação com quantas solicitações eles receberam. Essa métrica mostra quantas solicitações de atributos são respondidas instantaneamente pelos nós de *fog*. A Equação 4.2 determina como calcular a métrica de Taxa de Acerto em um Salto.

$$OHHR = \frac{número\ de\ cache\ hits\ de\ salto\ único}{número\ de\ cache\ hits\ +\ número\ de\ caches\ misses} \tag{4.2}$$

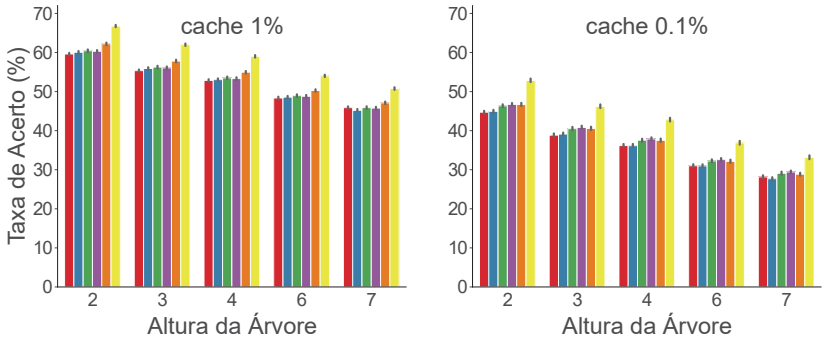
Por fim, para calcular o NAR, as simulações armazenam todos os atributos replicados em todos os nós de *fog* e consideram que cada nova réplica aumenta o custo de mantê-los seguros e atualizados. É importante ressaltar que, mesmo que a réplica de atributo seja descartada por uma política de substituição, as simulações ainda consideram essa réplica porque, em algum momento, houve a necessidade de manter esses atributos em cache seguros e atualizados.

4.8.2 Resultados Numéricos

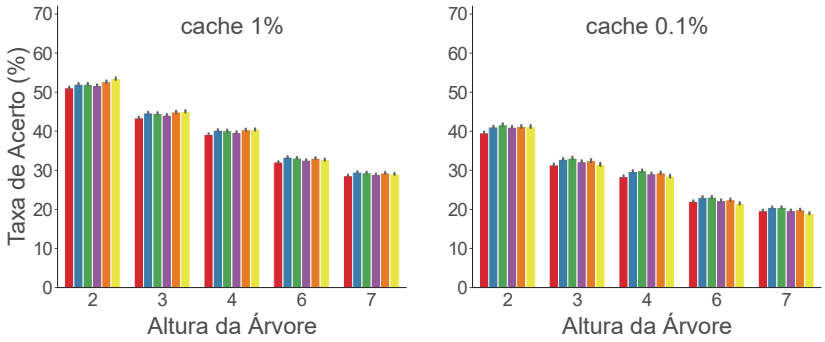
A Figura 4.20 apresenta a HR para todas as combinações de estratégias de cache. As Figuras 4.21 e 4.22 seguem a mesma metodologia de apresentar os resultados para o OHHR e o NAR, respectivamente. Cada par de gráficos representa uma estratégia reativa diferente, sendo o primeiro par (a) com a estratégia LCE, o segundo (b) com a estratégia LCD e o último (c) com a estratégia LEAF. Para cada par, os gráficos à esquerda apresentam nós de *fog* com mais recursos (1% do número total de atributos), e os gráficos à direita mostram nós de fog com capacidade de armazenamento mais limitada (0,1% do número total de atributos). Além disso, cada gráfico apresenta todas as políticas de substituição de cache em colunas, e cada conjunto de seis colunas se refere a topologias de rede diferentes. Os conjuntos à esquerda representam árvores menores e mais amplas, e os da direita referem-se a topologias de rede mais profundas e mais finas. Este trabalho considera políticas de substituição de cache tradicionais como RR, FIFO, LRU, SLRU e dois SPROTs diferentes: o SPROT tradicional e o SPROT100. Ambos SPROT e SPROT100 são a mesma política de substituição SPROT apresentada na Seção 4.5. No entanto, o SPROT usa o KNN com uma precisão de 45%, e o SPROT100 usa o oráculo perfeito com 100% de precisão.

A Figura 4.20 apresenta as estratégias reativas oportunísticas que são os fatores mais relevantes que afetam o índice de acerto. Por exemplo, ao comparar os resultados apresentados pelo LCE (a), LCD (b) e LEAF (c), o LCE dá cerca de 10% mais acertos nos nós de *fog* quando usado em conjunto com uma política de substituição. Esse resultado é esperado, uma vez que o LCE (a) é o que cria a alta quantidade de réplicas, aumentando a chance de ocorrência de um acerto nos nós de *fog*. As estratégias LCD (b) e LEAF (c) dão quase o mesmo número de acertos nos nós de *fog* quando usadas com as políticas de substituição RR, FIFO, LRU e SLRU. No entanto, ao aplicar a política de substituição SPROT, o índice de acerto aumenta 5% quando o

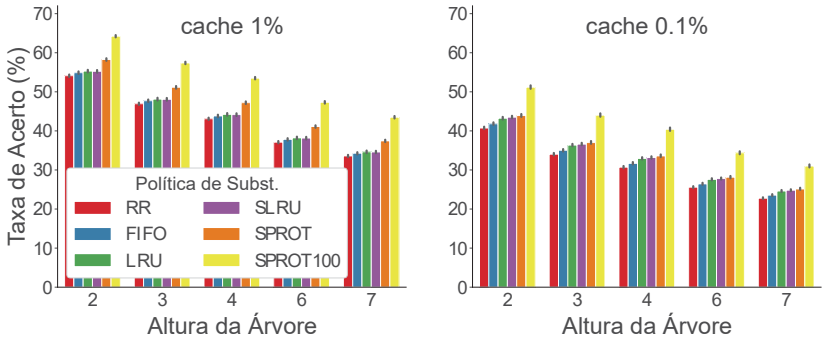
algoritmo de previsão de mobilidade é a nossa implementação KNN e aumenta em 10% quando o oráculo é perfeito. Esse resultado mostra que a estratégia LEAF não sobrecarrega as caches, dando mais oportunidades para o oráculo funcionar e aumentando a efetividade da estratégia. A estratégia LEAF combinada com a política de substituição SPROT dá resultados quase iguais ao LCE combinado com SPROT. Portanto, quanto mais preciso for o algoritmo de previsão de mobilidade empregado (representado por nossa implementação KNN e oráculo perfeito), mais efeito SPROT tem sobre os índices de acerto.



(a) LCE



(b) LCD

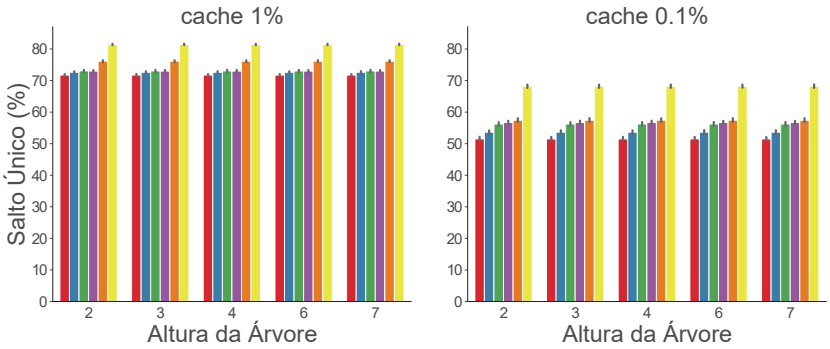


(c) LEAF

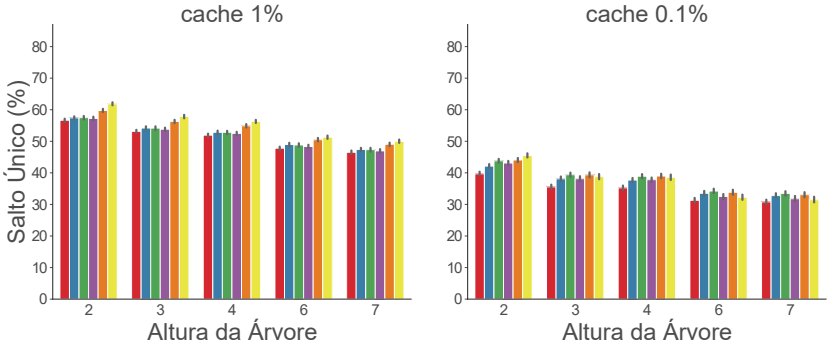
Figura 4.20: Hit Ratio por estratégia reativa

O tamanho da cache segue um padrão bem definido que mostra que o desempenho aumenta quando temos um cache com uma capacidade de armazenamento maior. Em geral, quando temos um cache com 1% do número total de atributos dos usuários, ele apresenta 10% a mais de desempenho do que a cache com o tamanho de 0,1%. Isso ocorre porque quanto maior o tamanho, maior o número de atributos armazenados e melhores as chances de encontrar o atributo demandado, sugerindo que investir em nós de *fog* com maior capacidade de armazenamento pode aumentar o desempenho da recuperação de atributos.

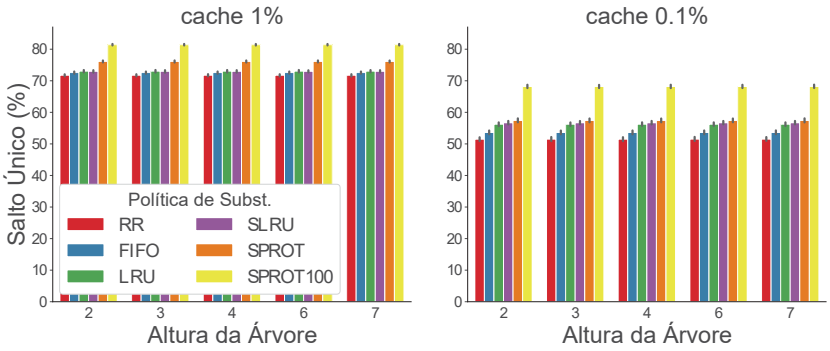
A topologia de rede também é um fator que impacta em todos os casos. O desempenho do mecanismo de cache oportunista diminui à medida que a árvore fica mais profunda. Esse resultado também pode ser esperado, uma vez que, como mostrado na Figura 4.19, o número de folhas permanece o mesmo para cada arquitetura, o que significa que as árvores mais profundas têm mais folhas conectadas por roteador, criando gargalos e uma disputa por atributos nesses nós.



(a) LCE



(b) LCD



(c) LEAF

Figura 4.21: Salto único por estratégia reativa

A Figura 4.21 apresenta o OHHR para cada combinação de estratégias reativas de armazenamento em cache, políticas de substituição, tamanhos de cache e topologias de rede. As estratégias de cache LCE (a) e LEAF (c) apresentam melhor desempenho do que a estratégia LCD (b) para todas as combinações equivalentes de estratégias de cache e políticas de substituição. Por exemplo, tanto LCE (a) quanto LEAF (c) reduzem até 80% no número de saltos para alcançar os atributos quando a cache tem o tamanho de 1% do número total de atributos e 60% quando as caches têm o tamanho de apenas 0,1% do número total de atributos. Esses resultados são qualitativamente iguais, independentemente da política de substituição e topologia de rede. Por outro lado, para a estratégia LCD (b), observamos que o tamanho da cache afeta o desempenho,

mas a topologia da rede também o afeta. Isso ocorre porque o LCD adota uma abordagem mais conservadora, descendo lentamente com os atributos. Portanto, é possível observar que quanto maior a altura da árvore, pior é o resultado dessa estratégia. Ao comparar LCE (a) e LEAF (c), observa-se que ambas têm estratégias fixas, colocando atributos em toda a rede e nas folhas, respectivamente. Consequentemente, ambas as estratégias apresentam resultados semelhantes e sem impacto em relação à altura da árvore.

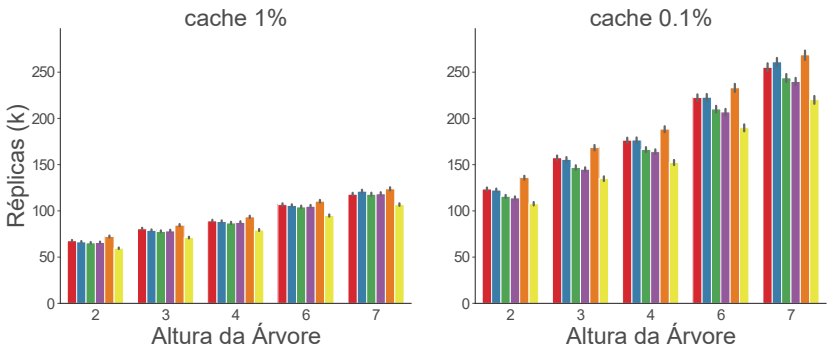
As políticas de substituição de cache tradicionais, como RR, FIFO, LRU, SLRU, apresentam resultados semelhantes para todos os casos das estratégias LCE (a) e LEAF (c) considerando caches do mesmo tamanho. A estratégia de substituição SPROT com LCE (a) e LEAF (c) apresenta melhor desempenho do que as políticas de substituição tradicionais. Usando um oráculo perfeito (SPROT100), essa estratégia teve um desempenho 10% melhor do que a taxa de acerto de um único salto. Esses resultados mostram que a proatividade dessas estratégias de substituição coloca os atributos nos lugares certos, aumentando a eficácia da cache e resultando em mais acertos de um único salto. No caso do LCD (b), as estratégias SPROT e SPROT100 não melhoram o desempenho.

A Figura 4.22 mostra o número de réplicas de atributos, distinguindo a estratégia reativa, política de substituição, topologia e tamanho de cache utilizados. Por exemplo, observando a estratégia reativa, pode-se observar que a LCE (a) cria mais réplicas do que as estratégias LEAF (c) e LCD (b) ao usar a mesma política de substituição e tamanho de cache. Esse resultado é esperado, uma vez que a LCE (a) cria uma cópia em cada nó, enquanto a LCD (b) e LEAF (c) criam apenas uma por solicitação de atributo, justificando seu menor número de réplicas.

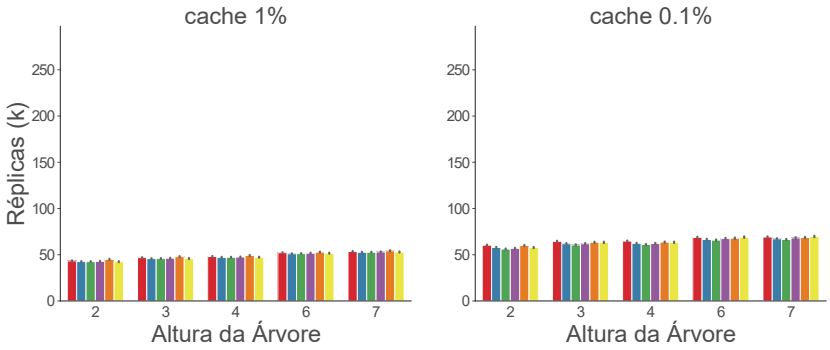
As políticas de substituição de cache SPROT e SPROT100 apresentam comportamentos distintos e interessantes. Enquanto as políticas tradicionais de substituição, como RR, FIFO, LRU e SLRU, têm resultados semelhantes, o SPROT100 aumenta o número de réplicas quando combinado com LCD (b) e LEAF (c). No entanto, quando usado em conjunto com LCE (a), as réplicas diminuem, independentemente do tamanho da cache e da altura da árvore. Esse comportamento ocorre porque, quando o mecanismo cria proativamente uma réplica em LCE (a), essa réplica é colocada no lugar certo, diminuindo a necessidade de novas réplicas e, ainda, quando as estratégias criam apenas algumas réplicas, como LCD (b) e LEAF (c), fazem uma réplica adicional no local certo, acelerando a descida dos atributos ao custo de aumentar o número de réplicas. Quando o SPROT usa nossa implementação de KNN, observamos que ele apenas aumenta o número de réplicas sem nenhum benefício aparente. Ainda, como pode ser visto, mesmo aumentando o número de réplicas com ou sem um oráculo perfeito, a estratégia proativa, juntamente com LEAF, custa muito menos, com apenas 10% menos de Hit-Ratio e um One-hop Hit Ratio semelhante.

O uso de tamanhos de cache maiores pode resultar em menos réplicas, já que haverá menos substituições. Esse efeito ocorre para todas as estratégias reativas. No entanto, é mais evidente para a LCE (a), em que caches de 0,1% criam duas vezes mais réplicas do que as caches de 1%. Além disso, enquanto as outras estratégias mantêm um número linear de réplicas de atributos para todas as alturas de árvore, a estratégia LCE aumenta o número de réplicas de atributos com base na altura da árvore. Ao adotar a LCE, árvores mais altas criam mais réplicas do que aquelas com alturas menores.

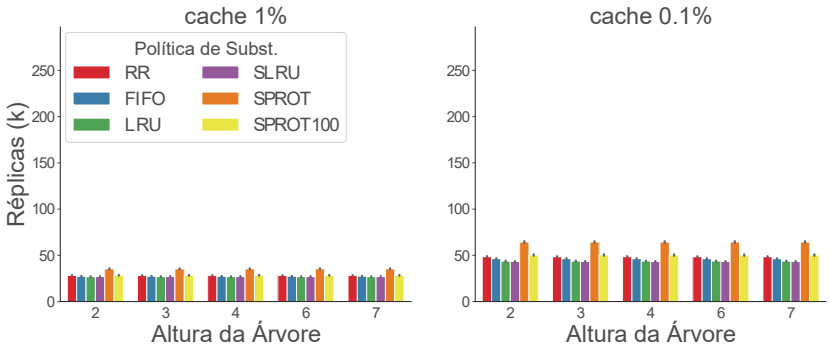
De acordo com os resultados anteriores, a estratégia reativa LEAF (c), combinada com a política de substituição SPROT100, apresenta o melhor custo-benefício de todas as combinações. Essa combinação proporciona uma redução de 10% nas taxas de acerto e de acerto em um salto, mas custa menos do que a LCE (a). Por exemplo, como mostrado na Figura 4.20, a LCE (a) e a LEAF (c), com caches de tamanho 0,1% e árvores de altura 7, utilizadas em conjunto com a política de substituição SPROT100, apresentam os melhores resultados. Nesse caso, a LCE



(a) LCE



(b) LCD



(c) LEAF

Figura 4.22: Réplicas por estratégia reativa

tem uma taxa de acerto cerca de 10% melhor. Por outro lado, como mostrado na Figura 4.22, o cenário exato custa quase oito vezes mais para a LCE (a) do que para a LEAF (c).

4.9 RESUMO

Os nossos resultados evidenciam que o método que propomos melhora o desempenho do mecanismo de controle de acesso. Por exemplo, pode reduzir em 80% o número de acessos à nuvem, uma vez que os dispositivos IoT encontram rapidamente réplicas dos atributos na neblina. Os resultados também mostram que a precisão do algoritmo de predição de mobilidade desempenha um papel crítico no desempenho do sistema de cache. Por exemplo, o método KNN apresenta uma taxa de solicitações à nuvem semelhante à de um oráculo perfeito, mas o número de réplicas dos atributos pode aumentar consideravelmente. Portanto, concluímos que o método remove a centralidade PDP e PIP, aumenta a eficiência da rede e reduz problemas de atraso que as decisões de política podem causar em aplicações IoT, baseadas na neblina, sem aumentar

o custo para manter a segurança e a frescura dos atributos em cache. Nesse sentido, como a precisão do algoritmo de predição de mobilidade é crítica para o nosso método, destacamos a necessidade de explorar novas estratégias reativas e oportunísticas para lidar com previsões errôneas, removendo réplicas de atributos desnecessárias.

Sendo assim, neste capítulo apresentamos os resultados obtidos que vão ao encontro dos OE2 e OE3 - ambos relacionados com a utilização do comportamento do usuário para a distribuição de atributos. Sendo assim, pode-se resumir nossas principais contribuições neste capítulo:

- Coletamos e analisamos dados de acesso a uma aplicação IoT popular em um vasto campus universitário durante 4 meses, o que possibilitou compreender os comportamentos de acesso e mobilidade dos usuários nessa aplicação.
- Sugerimos uma arquitetura que viabiliza a distribuição de atributos em caches, empregando a tecnologia de computação em névoa para a criação dessas caches.
- Apresentamos um método que combina abordagens reativas e proativas para a distribuição de atributos, onde a abordagem reativa se baseia em estratégias de cache tradicionais, enquanto a estratégia proativa leva em conta o comportamento do usuário para melhor posicionar as réplicas.
- Desenvolvemos e validamos um mecanismo de sincronização e revalidação de acessos e atributos que diminui a exposição do PDP a atributos desatualizados.

O próximo capítulo conclui esta tese e aponta para as direções de pesquisas futuras.

5 CONCLUSÕES

Neste capítulo, são apresentadas as principais contribuições desta tese. Além disso, esse capítulo também aponta uma discussão sobre as direções futuras de pesquisa. Primeiramente, demonstramos as realizações desta dissertação na Seção 5.1 e dividimos essa seção em uma subseção para cada objetivo de pesquisa, cada uma com um breve resumo do objetivo e dos resultados obtidos. Em seguida, é apresentada uma discussão mais ampla sobre as direções futuras de pesquisa e as questões abertas de pesquisa na Seção 5.2.

5.1 PRINCIPAIS CONTRIBUIÇÕES

Retomando ao Capítulo 1, o objetivo desta tese era desenvolver novos modelos, arquiteturas, métodos e ferramentas para apoiar o mecanismo de autorização, tornando-o mais **escalável**, **gerenciável** e com melhor **desempenho**. Sendo assim, o objetivo desta dissertação foi dividido na entrega dos três objetivos específicos de pesquisa:

- OE1 - Utilizar o comportamento do usuário para gerenciar políticas de acesso
- OE2 - Modelar o comportamento de usuários em aplicações reais
- OE3 - Utilizar o comportamento do usuário para distribuição de atributos em caches

A seguir, é apresentado os resultados obtidos para cada objetivo de pesquisa.

5.1.1 OE1 - Utilizar o comportamento do usuário para gerenciar políticas de acesso

Em nosso OE1, propusemos um método que emprega o comportamento do usuário para gerenciar políticas de acesso de forma auditável e compreensível para humanos. Após avaliar a literatura sobre métodos capazes de gerar políticas auditáveis, optamos por utilizar árvores de decisão e algoritmos genéticos. As bases de dados utilizadas em nosso estudo consistem em um conjunto sintético e uma base real fornecida pela *Amazon*. A base sintética apresenta uma distribuição de acessos aceitos e negados de 80-20, enquanto a base da *Amazon* é desbalanceada, com uma distribuição de 93-7. Nesse contexto, nossas principais contribuições são:

- Um método de mineração de políticas que utiliza árvores de decisão e algoritmos genéticos para extrair políticas de acesso a partir de registros. Essa abordagem combina a legibilidade das árvores de decisão e remove sua dependência do registro através de distorções intencionais no aprendizado, visando expandir o espaço de busca e encontrar políticas mais otimizadas;
- Demonstramos que, quando os conjuntos de aprendizado estão equilibrados, nosso método é capaz de extrair políticas precisas e auditáveis com um impacto negativo mínimo no tamanho das políticas de acesso em comparação com outros métodos da literatura. No entanto, em conjuntos de dados extremamente desequilibrados, tanto o método proposto quanto os encontrados na literatura enfrentaram dificuldades para fornecer resultados precisos.

Essas contribuições foram utilizadas na publicação (Cremonezi et al., 2021).

5.1.2 OE2 - Modelar o comportamento de usuários em aplicações reais

No OE2, nos concentramos em caracterizar e modelar o comportamento do usuário e a carga de trabalho associados na recuperação de atributos em um amplo campus universitário. Para isso, coletamos dados reais de uma aplicação IoT, com o objetivo de capturar os padrões de acesso e mobilidade dos usuários em um grande campus universitário na América do Sul. Nosso conjunto de dados inclui políticas de acesso para estudantes, equipe administrativa e professores, com uma população diária de aproximadamente 20 mil pessoas. Rastreamos o mecanismo de autorização da universidade durante quatro meses em diversos pontos de acesso e verificamos os registros que continham informações como horário do evento, identidade do usuário e atributos necessários para cada solicitação de acesso. Nossas principais contribuições em relação à modelagem do comportamento de usuários em aplicações reais podem ser resumidas da seguinte forma:

- Caracterizamos os diversos pontos de acesso e observamos que os usuários tendem a se conectar por curtos períodos em pontos de acesso ao longo de suas rotas, enquanto passam mais tempo conectados em pontos de acesso próximos a salas de aula e a outros pontos de interesse. Essa abordagem permite a identificação de padrões de rota e mobilidade, bem como o comportamento dos usuários em relação ao tempo de conexão e à localização dos pontos de acesso;
- Com base nos padrões de mobilidade em um campus universitário, propusemos uma categorização dos usuários. Observamos que cerca de 75% dos usuários exibem comportamento de média-alta e alta mobilidade, ou seja, concluímos que esses usuários representam um desafio maior para a oferta de um armazenamento em cache de atributos eficiente.

Essas contribuições foram utilizadas em três publicações (Gomes et al., 2021b,a; Cremonezi et al., 2022b)

5.1.3 OE3 - Utilizar o comportamento do usuário para distribuição de atributos em caches

No OE3, focamos em analisar como aprimorar o desempenho da recuperação de atributos para a avaliação das políticas de acesso. Sendo assim, para satisfazer esse OE, nossa hipótese envolveu o uso de conceitos de *caching* para armazenar atributos próximos ao usuário. No entanto, ao longo de nosso trabalho, enfrentamos questões importantes que não poderiam ser negligenciadas. A princípio, replicar atributos em várias caches parecia muito atraente para melhorar o desempenho da recuperação de atributos. Porém, a cada réplica criada, percebemos que os custos relacionados à manutenção da consistência e à segurança dos atributos aumentavam. Assim, identificamos a necessidade de aplicar técnicas mais adequadas para a replicação desses atributos. Encontrar técnicas que equilibrem desempenho e segurança é essencial para que os IAMs possam oferecer controle de acesso com desempenho adequado para IoT. Contudo, embora alguns estudos anteriores mencionassem essa investigação, poucos se aprofundaram no tema. Entre os que o fizeram, abordaram o problema como se fosse um problema de cache tradicional e aplicaram técnicas clássicas de cache, o que, considerando a natureza das identidades, não é apropriado.

Para preencher essa lacuna, o comportamento do usuário analisado no OE2 proporcionou *insights* para o desenvolvimento de técnicas mais adequadas, permitindo que o método de distribuição de atributos seja mais eficiente e equilibre a replicação e a segurança dos atributos. Portanto, nossas principais contribuições em relação à replicação de atributos em caches, utilizando o comportamento do usuário, podem ser resumidas da seguinte forma:

- Propusemos uma arquitetura que possibilita a distribuição de atributos em caches. Essa arquitetura utiliza da tecnologia de computação em névoa para estabelecer essas caches. Adicionalmente, apresentamos um método que se utiliza de abordagens reativas e proativas para a distribuição de atributos. A abordagem reativa é baseada em estratégias de cache convencionais, enquanto a estratégia proativa utiliza o comportamento do usuário para posicionar as réplicas de maneira mais eficiente;
- Empregamos um subconjunto dos dados coletados no OE2 para avaliar o mecanismo oportunístico de distribuição de atributos. Nossos resultados indicam que a combinação das abordagens reativas e proativas é vantajosa, proporcionando melhorias de até 10% nas taxas de acerto e acerto em um salto em comparação com as estratégias de cache convencionais. Observou-se que as melhorias mais expressivas ocorrem ao unir a estratégia reativa LEAF à política de substituição SPROT100. Especificamente, constatamos que essa combinação se torna atraente devido à proximidade dos resultados em relação à melhor estratégia de cache tradicional (aproximadamente 10% mais requisições na nuvem do que o LCE), porém com um custo de manutenção da consistência e segurança dos atributos quase oito vezes menor.

Essas contribuições foram apresentadas em cinco publicações (Cremonezi et al., 2019; Gomes et al., 2021b,a; Cremonezi et al., 2022b,a)

5.1.4 Contribuição Final

Após alcançar nossos três objetivos específicos, a hipótese proposta nesta tese se mostrou verdadeira: ao utilizar um conjunto de dados que inclui informações sobre o comportamento do usuário em uma aplicação sob a perspectiva do mecanismo de controle de acesso, como políticas avaliadas, resultado das autorizações, localizações dos acessos e atributos mais utilizados, foi possível aplicar diversas técnicas de análise desses comportamentos para aprimorar os componentes e mecanismos dos sistemas IAM. Consequentemente, as melhorias alcançadas nesta tese têm o potencial de reduzir a complexidade do gerenciamento e manutenção das políticas de acesso e aprimorar o desempenho do processo de recuperação e sincronização de atributos para a avaliação das políticas de acesso. Portanto, ao adquirir uma compreensão sólida do comportamento do usuário no contexto do controle de acesso, conseguimos utilizar essas informações para otimizar o gerenciamento de atributos e políticas de acesso, o que tornou o método de controle de acesso mais eficiente e permitiu que os sistemas IAM se tornem mais eficazes e adequados para lidar com as necessidades das aplicações IoT analisadas.

5.2 TRABALHOS FUTUROS

Nesta seção, abordaremos as direções futuras de pesquisa que pretendemos seguir. Essas novas direções de pesquisa são organizadas novamente com base nos três OE adotados nesta dissertação. Discutiremos cada objetivo de pesquisa e as possíveis lacunas que podem ser investigadas em trabalhos futuros relacionados a eles.

5.2.1 OE1 - Utilizar o comportamento do usuário para gerenciar políticas de acesso

Em relação ao OE1, pretendemos explorar duas possíveis direções para aprimorar o método proposto: (i) implementar técnicas que possam lidar com conjuntos de dados extremamente desbalanceados ou (ii) adotar abordagens alternativas para gerar políticas compreensíveis e eliminar distorções presentes na política resultante.

No que diz respeito à direção (i), na literatura, existem algoritmos de aprendizado de máquina projetados especificamente para tratar de bases desbalanceadas, como o *Support Vector Machine* (SVM) e o *Balanced Random Forest*. Entretanto, tais métodos não são facilmente interpretáveis, sendo crucial aplicar técnicas de interpretabilidade ao utilizá-los na mineração de políticas de acesso. Por exemplo, as *Random Forests* são geralmente consideradas de difícil interpretação, uma vez que elas combinam várias árvores de decisão em uma espécie de votação para chegar à decisão final. Nesse sentido, estratégias de interpretabilidade encontradas na literatura, como selecionar árvores de decisão representativas da *Random Forest*, onde algumas árvores são escolhidas para capturar o comportamento médio da floresta, e técnicas de explicação, como o LIME (*Local Interpretable Model-agnostic Explanations*), que criam regras interpretáveis para as previsões de qualquer modelo, podem ser extremamente úteis neste contexto e devem ser empregadas.

Caso prossiga para a direção (ii), técnicas avançadas de refinamento de árvores de decisão encontradas na literatura podem ser úteis. Por exemplo, a poda das árvores de decisão pode ser aprimorada para remover distorções que foram inseridas. Similarmente, técnicas como a validação cruzada durante o treinamento podem ajudar na escolha da melhor árvore de decisão a partir do conjunto de políticas candidatas. Além disso, a combinação desses e outras abordagens de refinamento e seleção de árvores de decisão pode resultar em políticas de acesso mais eficientes e compreensíveis.

5.2.2 OE2 - Modelar o comportamento de usuários em aplicações reais

Para futuros trabalhos relacionados ao OE2, diversas direções podem ser exploradas, levando em consideração a caracterização e a modelagem do comportamento típico dos usuários em diferentes aplicações sob a perspectiva do controle de acesso. Nesta tese, focamos em uma aplicação universitária, entretanto, no contexto da IoT, uma ampla variedade de aplicações pode ser investigada, incluindo as casas inteligentes, os veículos inteligentes e outros. Percebemos nesta pesquisa que trabalhos relacionados ao controle de acesso requerem, no mínimo, um conjunto de dados de uma aplicação real para determinar sua eficácia e validade. Logo, destacamos a importância de coletar e analisar dados de diferentes domínios e aplicações para desenvolver novos trabalhos e analisar, por exemplo, o impacto da utilização das caches de atributos em outras aplicações.

5.2.3 OE3 - Utilizar o comportamento do usuário para distribuição de atributos em caches

Por fim, imaginamos quatro direções para trabalhos futuros em relação ao mecanismo oportunístico de distribuição de atributos em caches entregue no OE3. Primeiramente, vamos considerar a abordagem reativa de distribuição de atributos. Nesta tese, o mecanismo reativo de distribuição de atributos utiliza as clássicas estratégias LCE, LCD e LEAF. Cada uma com suas limitações e vantagens. Conforme apresentado, mesmo que uma estratégia seja adequada em um determinado cenário, ela pode não ser adequada para todos os cenários. Sendo assim, um próximo passo de investigação é explorar outras abordagens de distribuição de atributos em caches. Uma abordagem híbrida ou adaptativa em relação ao número de réplicas pode ser uma estratégia válida que merece atenção. Por exemplo, enquanto o custo estiver abaixo de um determinado limiar, pode-se utilizar estratégias mais expansivas, como a LCE. Caso o custo atinja um determinado patamar, utiliza-se a estratégia LEAF. Essa abordagem híbrida aproveita as vantagens de cada estratégia individual, dependendo dos custos associados à replicação. Para implementar tal estratégia, basta monitorar o mecanismo de sincronização de atributos e alimentar um sistema que determina qual estratégia aplicar e quando mudar de uma estratégia para outra.

Em segundo lugar, observamos que a precisão do algoritmo de predição de mobilidade desempenha um papel crítico no mecanismo proativo de distribuição de atributos. Para ajudar a eliminar réplicas de atributos desnecessárias e melhorar a eficiência geral do mecanismo, além de aprimorar a acurácia da predição, abordagens inovadoras, como a de (Ravidas et al., 2020), podem ser úteis. Por exemplo, pode-se desenvolver um método capaz de determinar uma incerteza nas previsões de mobilidade e, ao invés de criar uma réplica do atributo, utiliza-se um atributo "virtual" calculado com base em estimativas relacionadas aos acessos anteriores. Essa abordagem considera as informações disponíveis, como padrões de acesso e comportamento dos usuários, para gerar atributos virtuais que representem os dados reais com uma margem de erro aceitável e não geraria um custo adicional ao erro. No entanto, ao adotar essa direção, é necessário ter cautela em relação à acurácia dos atributos "virtuais" para garantir a confiabilidade do sistema. Para isso, pode-se considerar novamente o uso de técnicas de aprendizado de máquina para aprimorar a qualidade das estimativas.

Em terceiro lugar, considerando o mecanismo de substituição de atributos, percebemos que a escolha da política de substituição adequada pode ter um impacto significativo no desempenho e na eficiência do mecanismo de replicação de atributos. Assim, várias outras estratégias, não abordadas nesta tese, devem ser exploradas para verificar se são mais adequadas. Por exemplo, a política de *Adaptive Replacement Cache* (ARC) é reconhecida na literatura como uma alternativa ao LRU e FIFO, que se adapta às mudanças nos padrões de acesso. A ARC mantém informações sobre os itens recentemente evitados e usados e ajusta dinamicamente qual estratégia utilizar. De maneira similar, *Clock with Adaptive Replacement* (CAR) é uma estratégia que utiliza uma lista dupla de itens recentemente acessados (similar aos utilizados nesta tese) e um ponteiro para simular um relógio. Com base nesses elementos, a CAR pode tomar decisões mais precisas sobre quais itens substituir, levando em consideração tanto a "idade"da réplica quanto a sua frequência de acesso. Nesse caso, ambos os métodos, ARC e CAR, podem ser úteis para dar continuidade a esta tese. Ao avaliar diferentes políticas de substituição, é importante analisar o desempenho delas em diversos cenários e aplicações, identificando as abordagens mais adequadas para cada situação e melhorando a eficiência geral do mecanismo de replicação de atributos como um todo.

Finalmente, nossa quarta e última direção futura se dirige ao mecanismo de sincronização de atributos. Atualmente, utilizamos o gerenciador de atributos para coordenar todas as atualizações e sincronizações entre os atributos, ou seja, o gerenciador de atributos é responsável por propagar essas alterações para todas as caches relevantes que possuam um determinado atributo. A vantagem desse método é a simplicidade na implementação, mas atualmente ele representa um ponto único de falha. Nesse caso, para oferecer melhor escalabilidade e segurança, uma abordagem distribuída desse gerenciador de atributos é necessária.

REFERÊNCIAS

Abhishek, K., Roshan, S., Kumar, P. e Ranjan, R. (2013). A comprehensive study on multifactor authentication schemes. Em *Advances in Computing and Information Technology*, páginas 561–568. Springer.

Abomhara, M. e Køien, G. M. (2014). Security and privacy in the internet of things: Current status and open issues. Em *2014 international conference on privacy and security in mobile systems (PRISMS)*, páginas 1–8. IEEE.

Ahson, S. A. e Ilyas, M. (2008). *SIP handbook: services, technologies, and security of Session Initiation Protocol*. CRC Press.

Al-Hasnawi, A., Carr, S. M. e Gupta, A. (2019). Fog-based local and remote policy enforcement for preserving data privacy in the internet of things. *Internet of Things*, 7:100069.

Al-Turjman, F., Zahmatkesh, H. e Shahroze, R. (2022). An overview of security and privacy in smart cities' iot communications. *Transactions on Emerging Telecommunications Technologies*, 33(3):e3677.

Alaa, M., Zaidan, A. A., Zaidan, B. B., Talal, M. e Kiah, M. L. M. (2017). A review of smart home applications based on internet of things. *Journal of Network and Computer Applications*, 97:48–65.

Alharby, M. e Van Moorsel, A. (2017). Blockchain-based smart contracts: A systematic mapping study. Em *International Conference on Cloud Computing, Big Data and Blockchain*. IEEE.

Alshehri, A. e Sandhu, R. (2016). Access control models for cloud-enabled internet of things: A proposed architecture and research agenda. Em *2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC)*, páginas 530–538. IEEE.

Alshehri, A. e Sandhu, R. (2017). Access control models for virtual object communication in cloud-enabled iot. Em *2017 IEEE International Conference on Information Reuse and Integration (IRI)*, páginas 16–25. IEEE.

Amazon (2013). Amazon.com - Employee Access Challenge. `https://www.kaggle.com/c/amazon-employee-access-challenge/`. [Último acesso em 21/02/2021].

Andaloussi, Y., El Ouadghiri, M. D., Maurel, Y., Bonnin, J.-M. e Chaoui, H. (2018). Access control in iot environments: Feasible scenarios. Em *Procedia computer science*, páginas 1031–1036. Elsevier.

Atlam, H. F., Alassafi, M. O., Alenezi, A., Walters, R. J. e Wills, G. B. (2018). XACML for Building Access Control Policies in Internet of Things. Em *International Conference on Internet of Things, Big Data and Security*, páginas 253–260.

Ausanka-Crues, R. (2001). Methods for access control: advances and limitations. Em *Harvey Mudd College*, páginas 20–34.

Badran, H. (2019). Iot security and consumer trust. Em *Proceedings of the 20th Annual International Conference on Digital Government Research*, páginas 133–140.

Balte, A., Kashid, A. e Patil, B. (2015). Security issues in internet of things (iot): A survey. *International Journal of Advanced Research in Computer Science and Software Engineering*, 5(4).

Barka, E., Mathew, S. S. e Atif, Y. (2015). Securing the web of things with role-based access control. Em *International Conference on Codes, Cryptology, and Information Security*, páginas 14–26. Springer.

Behrmann, G., David, A. e Larsen, K. G. (2004). A tutorial on uppaal. *Formal Methods for the Design of Real-Time Systems: International School on Formal Methods for the Design of Computer, Communication, and Software Systems, Bertinora, Italy, September 13-18, 2004, Revised Lectures*, páginas 200–236.

Benantar, M. (2006). Introduction to identity-management models. Em *Access Control Systems: Security, Identity Management and Trust Models*, páginas 40–72. Springer.

Bhatt, S. (2018). *Attribute-Based Access and Communication Control Models for Cloud and Cloud-Enabled Internet of Things*. Tese de doutorado, The University of Texas at San Antonio.

Bhatt, S., Pham, T. K., Gupta, M., Benson, J., Park, J. e Sandhu, R. (2021). Attribute-based access control for aws internet of things and secure industries of the future. *IEEE Access*.

Bhatt, S. e Sandhu, R. (2020). Abac-cc: Attribute-based access control and communication control for internet of things. Em *Proceedings of the 25th ACM Symposium on Access Control Models and Technologies*, páginas 203–212.

Bhattacharyya, D., Ranjan, R., Alisherov, F., Choi, M. et al. (2009). Biometric authentication: A review. Em *International Journal of u-and e-Service, Science and Technology*, páginas 13–28.

Bui, T. e Stoller, S. D. (2020). Learning attribute-based and relationship-based access control policies with unknown values. Em *Int. Conf. on Information Systems Security*, páginas 23–44.

Castro, T. O., Caitité, V. G., Macedo, D. F. e dos Santos, A. L. (2019). Casa-iot: Scalable and context-aware iot access control supporting multiple users. *International Journal of Network Management*, 29(5):e2084.

Charaf, L. A., ALIHAMIDI, I., ADDAIM, A. e Abdessalam, A. (2020). A distributed xacml based access control architecture for iot systems. Em *2020 1st International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, páginas 1–5. IEEE.

Chen, B., Wan, J., Shu, L., Li, P., Mukherjee, M. e Yin, B. (2017). Smart factory of industry 4.0: Key technologies, application case, and challenges. Em *IEEE Access*, páginas 6505–6519. IEEE.

Choi, D., Seo, S.-H., Oh, Y.-S. e Kang, Y. (2018). Two-factor fuzzy commitment for unmanned iot devices security. Em *IEEE Internet of Things Journal*, páginas 335–348. IEEE.

Cotrini, C., Weghorn, T. e Basin, D. (2018). Mining abac rules from sparse logs. Em *IEEE European Symposium on Security and Privacy (EuroS&P)*, páginas 31–46.

Coyne, E. e Weil, T. R. (2013). Abac and rbac: scalable, flexible, and auditable access management. Em *IT Professional*, páginas 14–16. IEEE.

Cremonezi, B., Gomes Filho, A. R., Silva, E. F., Nacif, J. A. M., Vieira, A. B. e Nogueira, M. (2022a). Improving the attribute retrieval on abac using opportunistic caches for fog-based iot networks. *Computer Networks*, 213:109000.

Cremonezi, B., Rocha, L., Vieira, A., Oliveira, A. e Silva, E. (2022b). *MobiCASE 2022*. MobiCASE.

Cremonezi, B., Vieira, A., Nacif, J., Nogueira, M. e Santos, A. (2019). Um sistema multinível de distribuição de identidades em névoas computacionais. Em *Anais do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, páginas 543–555. SBC.

Cremonezi, B., Vieira, A., Nacif, J., Silva, E. F. e Nogueira, M. (2021). Um método para extração e refinamento de políticas de acesso baseado em árvore de decisão e algoritmo genético. Em *Anais do XXXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, páginas 686–699. SBC.

de Oliveira, F. T. e Simões, W. L. (2017). A indústria 4.0 e a produção no contexto dos estudantes da engenharia. Em *Simpósio de engenharia de produção*.

Dhanvijay, M. M. e Patil, S. C. (2019). Internet of things: A survey of enabling technologies in healthcare and its applications. *Computer Networks*, 153:113–131.

Din, I. U., Almogren, A., Guizani, M. e Zuair, M. (2019). A decade of internet of things: Analysis in the light of healthcare applications. *Ieee Access*, 7:89967–89979.

Ding, S. e Ma, M. (2021). An attribute-based access control mechanism for blockchain-enabled internet of vehicles. Em *Advances in Computer, Communication and Computational Sciences*, páginas 905–915. Springer.

Fan, X., Chai, Q., Xu, L. e Guo, D. (2020). Diam-iot: A decentralized identity and access management framework for internet of things. Em *Proceedings of the 2nd ACM International Symposium on Blockchain and Secure Critical Infrastructure*, páginas 186–191.

Ferrag, M. A., Maglaras, L. A., Janicke, H., Jiang, J. e Shu, L. (2017). Authentication protocols for internet of things: a comprehensive survey. *Security and Communication Networks*, 2017.

Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R. e Chandramouli, R. (2001). Proposed nist standard for role-based access control. Em *ACM Transactions on Information and System Security*, páginas 224–274. ACM.

Fotiou, N., Siris, V. A., Polyzos, G. C., Kortesniemi, Y. e Lagutin, D. (2022). Capabilities-based access control for iot devices using verifiable credentials. Em *2022 IEEE Security and Privacy Workshops (SPW)*, páginas 222–228. IEEE.

Gia, T. N., Ali, M., Dhaou, I. B., Rahmani, A. M., Westerlund, T., Liljeberg, P. e Tenhunen, H. (2017). Iot-based continuous glucose monitoring system: A feasibility study. *Procedia Computer Science*, 109:327–334.

Gomes, A. R., Cremonezi, B., Nacif, J. A. M., Nogueira, M., Silva, E. F. e Vieira, A. B. (2021a). Opportunistic attribute caching: Improving the efficiency of abac in fog-based iot networks. Em *ICC 2021-IEEE International Conference on Communications*, páginas 1–6. IEEE.

Gomes, A. R., Cremonezi, B., Silva, E. F., Vieira, A. B., Nacif, J. e Nogueira, M. (2021b). Uma política de cache de identidades multinível para névoas computacionais. Em *Anais do XXI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, páginas 15–28. SBC.

Gómez, A., Masip-Bruin, X., Marin-Tordera, E., Kahvazadeh, S. e Garcia, J. (2018). A resource identity management strategy for combined fog-to-cloud systems. Em *2018 IEEE 19th International Symposium on"A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*, páginas 01–06. IEEE.

Guo, G., Wang, H., Bell, D., Bi, Y. e Greer, K. (2003). Knn model-based approach in classification. Em *On the Move to Meaningful Internet Systems*. Springer.

Habib, K. e Leister, W. (2015). Context-aware authentication for the internet of things. Em *Elev. Int. Conf. Auton. Auton. Syst. fined*, páginas 134–139.

Hastari, R. R., Yuliana, M. e Kristalina, P. (2021). Students trajectory pattern finding scheme based on rssi geolocation as a part of smart campus. Em *2021 International Electronics Symposium (IES)*, páginas 337–342. IEEE.

He, W., Golla, M., Padhi, R., Ofek, J., Dürmuth, M., Fernandes, E. e Ur, B. (2018). Rethinking access control and authentication for the home internet of things (iot). Em *USENIX Security Symposium*, páginas 255–272.

Hosmer, C. e Hosmer, C. (2018). Iot vulnerabilities. *Defending IoT Infrastructures with the Raspberry Pi: Monitoring and Detecting Nefarious Behavior in Real Time*, páginas 1–15.

Hossain, S. A., Rahman, M. A. e Hossain, M. A. (2018). Edge computing framework for enabling situation awareness in iot based smart city. *Journal of Parallel and Distributed Computing*, 122:226–237.

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). Em *NIST special publication*. Citeseer.

Islam, S. R., Hossain, M., Hasan, R. e Duong, T. Q. (2018). A conceptual framework for an iot-based health assistant and its authorization model. Em *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, páginas 616–621. IEEE.

Iyer, P. e Masoumzadeh, A. (2018). Mining positive and negative attribute-based access control policy rules. Em *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, páginas 161–172.

Jabbar, W. A., Alsibai, M. H., Amran, N. S. S. e Mahayadin, S. K. (2018). Design and implementation of iot-based automation system for smart home. Em *2018 International Symposium on Networks, Computers and Communications (ISNCC)*, páginas 1–6. IEEE.

Jia, Y., Yuan, B., Xing, L., Zhao, D., Zhang, Y., Wang, X., Liu, Y., Zheng, K., Crnjak, P., Zhang, Y. et al. (2021). Who's in control? on security risks of disjointed iot device management channels. Em *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, páginas 1289–1305.

Jin, X., Krishnan, R. e Sandhu, R. (2012). A unified attribute-based access control model covering dac, mac and rbac. Em *IFIP Annual Conference on Data and Applications Security and Privacy*, páginas 41–55. Springer.

Jindou, J., Xiaofeng, Q. e Cheng, C. (2012). Access control method for web of things based on role and sns. Em *2012 IEEE 12th International Conference on Computer and Information Technology*, páginas 316–321. IEEE.

Joshitta, R. S. M. e Arockiam, L. (2016). Security in iot environment: a survey. *International Journal of Information Technology and Mechanical Engineering*, 2(7):1–8.

Kastner, W., Krammer, L. e Fernbach, A. (2017). State of the art in smart homes and buildings. Em *Industrial Communication Technology Handbook*, páginas 55–1. CRC Press.

Kim, H. e Lee, E. A. (2017). Authentication and authorization for the internet of things. *IT Professional*, 19(5):27–33.

Kim, J. E., Boulos, G., Yackovich, J., Barth, T., Beckel, C. e Mosse, D. (2012). Seamless integration of heterogeneous devices and access control in smart homes. Em *2012 Eighth International Conference on Intelligent Environments*, páginas 206–213. IEEE.

Kim, Y., Oh, H. e Kang, S. (2017). Proof of concept of home iot connected vehicles. Em *Sensors*, páginas 1289–1305. Multidisciplinary Digital Publishing Institute.

Krupp, A. F. (2023). Privacy is not dead: Expressively using law to push back against corporate deregulators and meaningfully protect data privacy rights. *Georgia Law Review*, 57(2):10.

Kumar, P., Lin, Y., Bai, G., Paverd, A., Dong, J. S. e Martin, A. (2019). Smart grid metering networks: A survey on security, privacy and open research issues. *IEEE Communications Surveys & Tutorials*, 21(3):2886–2927.

Kumar Jain, V. e Gajrani, J. (2021). Iot security: A survey of issues, attacks and defences. Em *Intelligent Learning for Computer Vision: Proceedings of Congress on Intelligent Systems 2020*, páginas 219–236. Springer.

Laghari, A. A., Wu, K., Laghari, R. A., Ali, M. e Khan, A. A. (2021). A review and state of art of internet of things (iot). *Archives of Computational Methods in Engineering*, páginas 1–19.

L'Amrani, H., Berroukech, B. E., El Idrissi, Y. E. B. e Ajhoun, R. (2016). Identity management systems: Laws of identity for models 7 evaluation. Em *IEEE International Colloquium on Information Science and Technology*, páginas 736–740. IEEE.

Lee, S., Choi, J., Kim, J., Cho, B., Lee, S., Kim, H. e Kim, J. (2017). Fact: Functionality-centric access control system for iot programming frameworks. Em *Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies*, páginas 43–54.

Liu, M., Yang, C., Li, H. e Zhang, Y. (2020). An efficient attribute-based access control (abac) policy retrieval method based on attribute and value levels in multimedia networks. *Sensors*, 20(6):1741.

Lo, S. K., Xu, X., Chiam, Y. K. e Lu, Q. (2017). Evaluating suitability of applying blockchain. Em *International Conference on Engineering of Complex Computer Systems*, páginas 158–161. IEEE.

Lu, H., Vaidya, J. e Atluri, V. (2008). Optimal boolean matrix decomposition: Application to role engineering. Em *IEEE 24th Int. Conf. on Data Engineering*, páginas 297–306.

Mahalle, P. N., Anggorojati, B., Prasad, N. R. e Prasad, R. (2013). Identity authentication and capability based access control (iacac) for the internet of things. *Journal of Cyber Security and Mobility*, 1(4):309–348.

Medvet, E., Bartoli, A., Carminati, B. e Ferrari, E. (2015). Evolutionary inference of attribute-based access control policies. Em *International Conference on Evolutionary Multi-Criterion Optimization*, páginas 351–365. Springer.

Merhav, N. (2019). False-accept/false-reject trade-offs for ensembles of biometric authentication systems. Em *IEEE Transactions on Information Theory*, páginas 4997–5006. IEEE.

Miessler, D. (2015). Securing the internet of things: Mapping attack surface areas using the owasp iot top 10. Em *RSA conference*, volume 2015.

Moustafa, H., Schooler, E. M., Shen, G. e Kamath, S. (2016). Remote monitoring and medical devices control in ehealth. Em *2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, páginas 1–8. IEEE.

Nakamoto, S. et al. (2008). Bitcoin: A peer-to-peer electronic cash system. *Working Paper*.

Nawir, M., Amir, A., Yaakob, N. e Lynn, O. B. (2016). Internet of things (iot): Taxonomy of security attacks. Em *2016 3rd International Conference on Electronic Design (ICED)*, páginas 321–326. IEEE.

Neisse, R., Steri, G. e Baldini, G. (2014). Enforcement of security policy rules for the internet of things. Em *2014 IEEE 10th international conference on wireless and mobile computing, networking and communications (WiMob)*, páginas 165–172. IEEE.

Nida, P., Dhiman, H. e Hussain, S. (2014). A survey on identity and access management in cloud computing. Em *Int. J. Eng. Res. Technol.*

Ometov, A., Petrov, V., Bezzateev, S., Andreev, S., Koucheryavy, Y. e Gerla, M. (2019). Challenges of multi-factor authentication for securing advanced iot applications. *IEEE Network*, 33(2):82–88.

Ouaddah, A., Mousannif, H., Abou Elkalam, A. e Ouahman, A. A. (2017). Access control in the internet of things: Big challenges and new opportunities. *Computer Networks*, 112:237–262.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Pramanik, P. K. D., Upadhyaya, B. K., Pal, S. e Pal, T. (2019). Internet of things, smart sensors, and pervasive systems: Enabling connected and pervasive healthcare. Em *Healthcare data analytics and management*, páginas 1–58. Elsevier.

Rao, B. P., Saluia, P., Sharma, N., Mittal, A. e Sharma, S. V. (2012). Cloud computing for internet of things & sensing based applications. Em *2012 Sixth International Conference on Sensing Technology (ICST)*, páginas 374–380. IEEE.

Rauchs, M., Glidden, A., Gordon, B., Pieters, G. C., Recanatini, M., Rostand, F., Vagneur, K. e Zhang, B. Z. (2018). Distributed ledger technology systems: a conceptual framework. *Available at SSRN 3230013*.

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

Ravidas, S., Ray, I. e Zannone, N. (2020). Handling incomplete information in policy evaluation using attribute similarity. Em *2020 Second IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, páginas 79–88. IEEE.

Ray, I., Alangot, B., Nair, S. e Achuthan, K. (2017). Using attribute-based access control for remote healthcare monitoring. Em *2017 Fourth International Conference on Software Defined Systems (SDS)*, páginas 137–142. IEEE.

Ray, P. P., Dash, D. e De, D. (2019). Edge computing for internet of things: A survey, e-healthcare case study and future direction. *Journal of Network and Computer Applications*, 140:1–22.

Raza, M. S., Zongsheng, T. e Muslam, M. M. A. (2021). A review of human-to-machine and machine-to-machine approaches for internet of things. Em *2020 International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, páginas 1–5. IEEE.

Roussos, G., Peterson, D. e Patel, U. (2003). Mobile identity management: An enacted view. Em *International Journal of Electronic Commerce*, páginas 81–100. Taylor & Francis.

Said, O. e Masud, M. (2013). Towards internet of things: Survey and future vision. Em *International Journal of Computer Networks*, páginas 1–17.

Salah, K., Alfalasi, A., Alfalasi, M., Alharmoudi, M., Alzaabi, M., Alzyeodi, A. e Ahmad, R. W. (2020). Iot-enabled shipping container with environmental monitoring and location tracking. Em *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*, páginas 1–6. IEEE.

Salonikias, S., Mavridis, I. e Gritzalis, D. (2015). Access control issues in utilizing fog computing for transport infrastructure. Em *International Conference on Critical Information Infrastructures Security*, páginas 15–26. Springer.

Sandhu, R. e Munawer, Q. (1998). How to do discretionary access control using roles. Em *Proceedings of the third ACM workshop on Role-based access control*, páginas 47–54. ACM.

Sandhu, R. S. (1993). Lattice-based access control models. Em *Computer*, páginas 9–19. IEEE.

Sciancalepore, S., Piro, G., Tedeschi, P., Boggia, G. e Bianchi, G. (2018). Multi-domain access rights composition in federated iot platforms. Em *EWSN*, páginas 290–295.

Seitz, L., Selander, G. e Gehrmann, C. (2013). Authorization framework for the internet-of-things. Em *2013 IEEE 14th International Symposium on"A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*, páginas 1–6. IEEE.

Shakarami, M. (2022). *Operation and Administration of Access Control in IoT Environments*. Tese de doutorado, The University of Texas at San Antonio.

Shakarami, M., Benson, J. e Sandhu, R. (2022). Blockchain-based administration of access in smart home iot. Em *Proceedings of the 2022 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems*, páginas 57–66.

Shakarami, M. e Sandhu, R. (2019). Safety and consistency of subject attributes for attribute-based pre-authorization systems. Em *National Cyber Summit*, páginas 248–263. Springer.

Siebach e Jess, J. A. (2021). *The Abacus: A New Approach to Authorization*. Tese de doutorado, Brigham Young University.

Silva, E. F., Muchaluat-Saade, D. C. e Fernandes, N. C. (2018). Across: A generic framework for attribute-based access control with distributed policies for virtual organizations. *Future Generation Computer Systems*, 78:1–17.

Song, W., Feng, N., Tian, Y. e Fong, S. (2017). An iot-based smart controlling system of air conditioner for high energy efficiency. Em *2017 IEEE International Conference 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)*, páginas 442–449. IEEE.

Stanislav, M. e Beardsley, T. (2015). Hacking iot: A case study on baby monitor exposures and vulnerabilities. *Rapid7 Report*.

Stobert, E. e Biddle, R. (2014). The password life cycle: user behaviour in managing passwords. Em *10th Symposium On Usable Privacy and Security ({SOUPS} 2014)*, páginas 243–255.

Syed, A. S., Sierra-Sosa, D., Kumar, A. e Elmaghraby, A. (2021). Iot in smart cities: A survey of technologies, practices and challenges. *Smart Cities*, 4(2):429–475.

Systems, C. (2015). Fog computing and the internet of things: Extend the cloud to where the things are. *White Paper*.

Talukdar, T., Batra, G., Vaidya, J., Atluri, V. e Sural, S. (2017). Efficient bottom-up mining of attribute based access control policies. Em *Int. Conf. on Collaboration and Internet Computing*.

Tian, Y., Zhang, N., Lin, Y.-H., Wang, X., Ur, B., Guo, X. e Tague, P. (2017). Smartauth: User-centered authorization for the internet of things. Em *26th {USENIX} Security Symposium ({USENIX} Security 17)*, páginas 361–378.

Tobin, A. e Reed, D. (2016). The inevitable rise of self-sovereign identity. Em *The Sovrin Foundation*.

Toelen, O. (2008). Identity and access management. Dissertação de Mestrado, Eindhoven University of Technology.

Torres, J., Macedo, R., Nogueira, M. e Pujolle, G. (2012). Identity management requirements in future internet. Em *Anais do XII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, páginas 317–323, Porto Alegre, RS, Brasil. SBC.

Tragos, E. Z., Pöhls, H. C., Staudemeyer, R. C., Slamanig, D., Kapovits, A., Suppan, S., Fragkiadakis, A., Baldini, G., Neisse, R., Langendörfer, P. et al. (2015). Securing the internet of things—security and privacy in a hyperconnected world. *Building the hyperconnected society-internet of things research and innovation value chains, ecosystems and markets. River Publishers Series of Communications*, páginas 189–219.

Ubale Swapnaja, A., Modani Dattatray, G. e Apte Sulabha, S. (2014). Analysis of dac mac rbac access control based models for security. Em *International Journal of Computer Applications*, páginas 6–13. Foundation of Computer Science.

Uganya, G., Radhika e Vijayaraj, N. (2021). A survey on internet of things: Applications, recent issues, attacks, and security mechanisms. *Journal of Circuits, Systems and Computers*, 30(05):2130006.

Umadevi, S. e Marseline, K. J. (2017). A survey on data mining classification algorithms. Em *Int. Conf. on Signal Processing and Communication*, páginas 264–268.

Vardhana, M., Arunkumar, N., Abdulhay, E. e Vishnuprasad, P. (2019). Iot based real time trafic control using cloud computing. Em *Cluster Computing*, páginas 2495–2504. Springer.

Velásquez, I., Caro, A. e Rodríguez, A. (2018). Authentication schemes and methods: A systematic literature review. Em *Information and Software Technology*, volume 94, páginas 30–37. Elsevier.

Wangham, M. S., Marins, A., Ferraz, C. A., da Silva, C. E., Saade, D. C., Silva, E. F., de Mello, E. R., de Oliveira, F. B., Seixas, F. L., Oliveira, L. B. et al. (2018). O futuro da gestão de identidades digitais. Em *Anais Estendidos do XVIII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, páginas 146–166. SBC.

Wayman, J., Jain, A., Maltoni, D. e Maio, D. (2005). An introduction to biometric authentication systems. Em *Biometric Systems*, páginas 1–20. Springer.

Whitmore, A., Agarwal, A. e Da Xu, L. (2015). The internet of things—a survey of topics and trends. Em *Information Systems Frontiers*, páginas 261–274. Springer.

Windley, P. J. (2005). *Digital Identity: Unmasking identity management architecture (IMA)*. "O'Reilly Media, Inc.".

Xu, W., Kong, L., Wang, N. e Liu, J. (2022). Optimization of attribute-based access control policy with priority filtering. Em *2022 IEEE 4th International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, páginas 1045–1052. IEEE.

Xu, Z. e Stoller, S. D. (2014). Mining attribute-based access control policies. *IEEE Transactions on Dependable and Secure Computing*, 12(5):533–545.

Yadav, C. S., Singh, J., Yadav, A., Pattanayak, H. S., Kumar, R., Khan, A. A., Haq, M. A., Alhussen, A. e Alharby, S. (2022). Malware analysis in iot & android systems with defensive mechanism. *Electronics*, 11(15):2354.

Zaidi, S. Y. A., Shah, M. A., Khattak, H. A., Maple, C., Rauf, H. T., El-Sherbeeny, A. M. e El-Meligy, M. A. (2021). An attribute-based access control for iot using blockchain and smart contracts. *Sustainability*, 13(19):10556.

Zhong, B. e Yang, F. (2020). From entertainment device to iot terminal: Smart tv helps define the future living in smart home. Em *Handbook of Research on Managerial Practices and Disruptive Innovation in Asia*, páginas 128–146. IGI Global.

Zhou, Q., Elbadry, M., Ye, F. e Yang, Y. (2018). Heracles: Scalable, fine-grained access control for internet-of-things in enterprise environments. Em *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, páginas 1772–1780. IEEE.

Zhu, Y., Huang, D., Hu, C.-J. e Wang, X. (2014). From rbac to abac: constructing flexible data access control for cloud storage services. Em *IEEE Transactions on Services Computing*, páginas 601–616. IEEE.

APÊNDICE A – A FERRAMENTA UPPAAL

A.1 DESCRIÇÃO

A ferramenta UPPAAL (Behrmann et al., 2004) permite uma modelagem de sistemas através da definição de vários autômatos básicos em um editor. Além disso, a ferramenta possui um simulador de trajetórias do sistema e um módulo de verificação automática de propriedades. O módulo de verificação se utiliza de algoritmos e estruturas de dados disponíveis na literatura para realizar um *model-checking* do sistema, que é um produto cartesiano dos autômatos básicos, contra propriedades expressas em um subconjunto da lógica TCTL (*Timed Computation Tree Logic*).

Na lógica TCTL, conceitualmente, o quantificador A denota "para toda trajetória", enquanto o quantificador E denota "existe uma trajetória". Para uma análise, esses quantificadores devem ser combinados com os quantificadores $<>$ e $[]$, que denotam, respectivamente, "em algum estado da trajetória (*eventually*)" e "em todos os estados da trajetória". Sendo assim, a ferramental UPPAAL consegue apresentar, no seu simulador, um exemplo e um contraexemplo de uma determinada expressão φ quando uma propriedade do tipo $E <> \varphi$ é verdadeira (exemplo), ou quando uma propriedade do tipo $A[]\varphi$ é falsa (contraexemplo). Para o presente trabalho, utiliza-se apenas as propriedades básicas de alcançabilidade expressas pelo predicado $E <> \varphi$, que denota a existência de uma trajetória em que a fórmula φ se torna válida em algum momento futuro.