Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Arquitetura para Privacidade na Integração de Internet das Coisas e Computação em Nuvem

Luis Alberto Belem Pacheco

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Orientador
Prof. Dr. Eduardo Adilio Pelinson Alchieri

Brasília
2018

Ficha Catalográfica de Teses e Dissertações

Está página existe apenas para indicar onde a ficha catalográfica gerada para dissertações de mestrado e teses de doutorado defendidas na UnB. A Biblioteca Central é responsável pela ficha, mais informações nos sítios:

http://www.bce.unb.br
http://www.bce.unb.br/elaboracao-de-fichas-catalograficas-de-teses-e-dissertacoes

Esta página não deve ser inclusa na versão final do texto.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Arquitetura para Privacidade na Integração de Internet das Coisas e Computação em Nuvem

Luis Alberto Belem Pacheco

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Prof. Dr. Eduardo Adilio Pelinson Alchieri (Orientador)
CIC/UnB

Prof. Dr. Diego de Freitas Aranha
IC/UNICAMP

Prof. Dr. Priscila America Solis Mendez
CIC/UnB

Prof. Dr. Bruno L. Macchiavello Espinoza
Coordenador do Programa de Pós-graduação em Informática

Brasília, 15 de fevereiro de 2018

Dedicatória

Dedico este trabalho à minha filha, Luna Gonçalves Pacheco, que tem sido uma grande alegria na minha vida.

Agradecimentos

Agradeço aos profissionais da Universidade de Brasília, pelo apoio durante todo este trabalho, especialmente ao Professor Eduardo Alchieri, pelas orientações e conselhos, que tiveram grande importância para a obtenção dos resultados.

Resumo

Através da Internet das Coisas (IoT), uma imensa quantidade de dispositivos são conectados à internet, gerando uma grande quantidade de dados. Computação em nuvem é uma tecnologia adotada atualmente para processar, armazenar e prover controle de acesso a dados. Atualmente propõe-se a integração entre computação em nuvem e a Internet das Coisas, a essa integração chama-se *Cloud of Things - CoT*. Essa abordagem é especialmente útil para redes domésticas e pessoais, tais como automação residencial e assistência médica, pois facilita o acesso a informação pelos indivíduos. Apesar de trazer benefícios aos usuários, a integração desses dois conceitos tecnológicos introduz muitos desafios na área de segurança, já que a informação sai da esfera de controle do usuário ao ser enviada para a nuvem. Para que haja uma adoção em massa dessa tecnologia é importante que hajam protocolos e mecanismos para preservar a privacidade dos dados dos usuários ao armazenar seus dados na nuvem.

Neste contexto, este trabalho propõe uma arquitetura de privacidade em *Cloud of Things*, permitindo ao usuário o controle completo do acesso aos dados gerados pelos dispositivos de suas redes IoT e armazenados na nuvem. A arquitetura proposta provê um controle fino sob os dados, pois os protocolos e controles de privacidade são executados nos dispositivos e não na borda da rede pelo *gateway*, o qual também pode representar um ponto único de falha ou quebrar a segurança do sistema uma vez comprometido por um ataque bem sucedido.

Este trabalho também desenvolveu uma melhoria à arquitetura proposta, diminuindo a sobrecarga nos dispositivos IoT. Avaliações foram conduzidas por meio de uma análise analítica e experimental, utilizando o simulador de redes ns-3, ambas abordagens propostas foram discutidas e comparadas com outras arquiteturas. Os resultados obtidos indicam que mesmo dispositivos severamente limitados podem implementar mecanismos de segurança para prover privacidade na *Cloud of Things*.

Palavras-chave: privacidade, segurança, internet das coisas, computação em nuvem

Abstract

A large number of devices are connected to the internet through the Internet of Things (IoT) paradigm, resulting in a huge amount of produced data. Cloud computing is a computing paradigm currently adopted to process, store and provide access control to these data. This integration is called *Cloud of Things - CoT* and is useful in personal networks, like residential automation and health care, since it facilitates the access to the information. Although this integration brings benefits to the users, it introduces many security challenges since the information leaves the user control and is stored at the cloud providers. Particularly interesting, in order for these technologies to be adopted, it is important to provide protocols and mechanisms to preserve the users privacy when storing their data in the cloud.

In this context, this paper proposes an architecture for privacy in Cloud of Things, which allows the users to fully control the access to the data generated by the devices of their IoT networks and stored in the cloud. The proposed architecture enables a fine grained control over data, since the privacy protocols and controls are executed at the IoT devices instead of at the network border by a gateway, which also could represent a single point of failure or a component that could impair the security properties of the system once it is compromised by a successful attack.

This work also developed an enhancement for the proposed architecture, decreasing its overhead in IoT devices. Evaluations were conducted throught analytical analysis and experiments, using the ns-3 network simulator, both aproaches were discussed and compared with other architectures. Obtained results indicate that, even in severely constrained IoT devices, security mechanisms to provide privacy in Cloud of Things can be implemented.

Keywords: privacy, security, Internet of Things, cloud computing, Cloud of Things

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AES *Advanced Encryption Standard.*

BSS *Block Based Sharing Scheme.*

CoAP Constrained Application Protocol.

CoT Cloud of Things.

CP Configuração de Privacidade.

DES *Data Encryption Standard.*

DICE *DTLS In Constrained Environments.*

DTLS *Datagram Transport Layer Security.*

ECDH *Elliptic-curve Diffie–Hellman.*

EPROTECT *Enhanced PROTeCt.*

HTTP Hypertext Transfer Protocol.

IaaS *Infrastructure as a service.*

IEEE Instituto de Engenheiros Elétricos e Eletrônicos.

IETF *Internet Engineering Task Force.*

IoT Internet of Things.

KAC *Key-Aggregate Cryptosystem.*

MAC *Message Authentication Code.*

NIST *National Institute of Standards and Technology.*

PaaS *Platform as a service.*

PDL *Privacy Development Language.*

PEP *Privacy Enforcement Points.*

PKI *Public Key Infrastructure.*

PP Política de Privacidade.

PPF Políticas de Privacidade Flexíveis.

PROTECT ***P****rivacy a****R****quitecture for integrati****O****n of internet of* ***T****hings and* ***C****loud computing.*

RFID *Radio-Frequency Identification.*

RP *Relying Party.*

RSSF Redes de Sensores Sem Fio.

SaaS *Software as a service.*

SLA *Service Level Agreement.*

SOA *Service Oriented Architecture.*

TCP Transmission Control Protocol.

TLS *Transport Layer Security.*

TPC Terceira Parte Confiável.

UDP User Datagram Protocol.

UPECSI *User-driven Privacy Enforcement for Cloud-based Services in the IoT.*

Capítulo 1

Introdução

Este capítulo apresenta o trabalho realizado. Primeiramente uma motivação é explorada, abordando os conceitos e desafios da área estudada, em seguida os objetivos deste trabalho são elencados, por fim a organização do texto é apresentada.

1.1 Motivação

Os avanços nas tecnologias de miniaturização de componentes eletrônicos e nas tecnologias de comunicação sem fio possibilitaram o advento da Internet das Coisas (do inglês *Internet of Things (IoT)*). No paradigma IoT, os mais diversos itens do nosso cotidiano terão acesso à Internet, trazendo uma enorme gama de benefícios à população [2]. São previstas bilhões de "coisas" conectando-se à Internet para prover os mais diferentes tipos de informações aos usuários [3].

Os dispositivos de IoT podem estar presentes nos mais diversos meios, porém comumente uma rede IoT é implementada por meio de sensores. Redes de Sensores Sem Fio (RSSF) são compostas por diversos dispositivos de tamanho limitado que possuem uma unidade de processamento, um sensor que proporciona a interação com o mundo físico e uma antena para comunicação sem fio [4]. A Figura 1.1 mostra um exemplo de redes de sensores sem fio em que vários dispositivos enviam dados para uma unidade central. Os dispositivos das RSSFs geralmente possuem tamanho e fonte de energia limitados, dessa forma, seus componentes, incluindo a pilha de rede, precisam possuir um baixo custo de processamento.

RSSFs possibilitam diversos tipos de aplicações, este conceito surgiu com foco militar e aeroespacial, e hoje em dia abrange inúmeras áreas, tais como monitoramento e controle de atividades industriais, automatização residencial, monitoramento de saúde, mobilidade urbana, etc. A grande heterogeneidade de áreas abrangidas fazem das redes de sensores

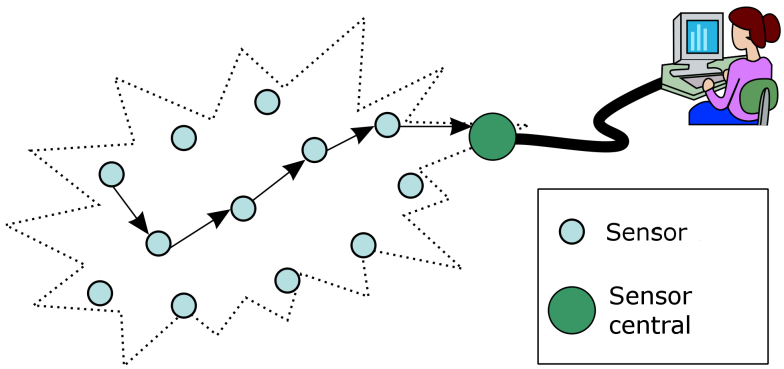


Figura 1.1: Exemplo de rede de sensores sem fio

sem fio perfeitas candidatas para possibilitar a Internet das Coisas, é previsto que RSSFs sejam grande parte das redes da Internet das Coisas.

Muitos protocolos, em todas as camadas da pilha, foram propostos para prover maior desempenho às redes de sensores. Nas camadas Física e de Enlace o padrão IEEE 802.15.4 [5] foi lançado em 2006 e possui diversas atualizações. Atualmente, é utilizado por grande parte das redes de sensores. Já protocolos das demais camadas possuem uma grande fragmentação, com diferentes propostas para diferentes aplicações, gerando a necessidade da realização de uma tradução quando a comunicação deve ocorrer com a Internet. Normalmente esta tradução é efetuada por um ponto de ligação (do inglês *gateway*), posicionado na borda da RSSF.

Atualmente grupos de trabalho de padronização da Internet tem realizado grande esforço para possibilitar a comunicação direta da rede de sensores com a Internet, utilizando protocolos da Internet (ou adaptações) nos sensores. Esta abordagem traz grande benefício para a implementação da Internet das Coisas, pois facilita o acesso direto à informação advinda do dispositivo pelo usuário [6]. Existem vários trabalhos em andamento cujo o objetivo é criar versões de protocolos já utilizados na Internet para que possam ser utilizados nas redes de sensores.

Lidar com a imensa quantidade de dados gerada pela IoT apresenta diversos desafios. As limitações tecnológicas da IoT (armazenamento, processamento, comunicação) podem ser mitigadas com a utilização de Computação em Nuvem. Estes modelos são complementares: a IoT produz uma imensa quantidade de dados, enquanto que a Computação em Nuvem é capaz de fornecer mecanismos para armazenamento e processamento destes dados. Esta combinação, aqui chamada de Cloud of Things (CoT), atualmente está sendo amplamente discutida [7, 8, 9, 10].

A computação em nuvem pode ampliar os recursos da IoT de diversas maneiras, visto que a nuvem é capaz de armazenar, processar e apresentar os dados gerados pelos dispositivos da Internet das Coisas. Aplicações podem utilizar os recursos virtualmente ilimitados

de plataformas de nuvem para processar e apresentar os dados da IoT aos usuários. Outras características da computação em nuvem também são de grande valia para a IoT, tais como a escalabilidade inerente da tecnologia e aumento da segurança, pois o controle de acesso à informação é realizado na nuvem, e essa dispõe de poder computacional adequado para essa tarefa, maior possibilidade de alcance, entre outros.

A Cloud of Things envolve a troca de informações privadas entre duas partes, portanto é necessário que os sistemas utilizados para propiciar esta arquitetura assegurem, dentre outras propriedades, a privacidade do usuário. Um sistema computacional deve assegurar, 3 propriedades para ser seguro [11]: confidencialidade, assegurando que apenas as partes interessadas terão acesso à informação; integridade, referente à proteção do dado contra mudanças indevidas; e disponibilidade, assegurando o acesso à informação quando necessário. A privacidade, contida na propriedade de confidencialidade, assegura o controle dos indivíduos em relação às suas informações, incluindo quem as coleta e armazena e a quem suas informações podem ser reveladas [12].Apesar dos imensos benefícios da integração entre IoT e Computação em Nuvem, a interação entre diferentes entidades aumenta a complexidade e a importunância do fornecimento de protocolos e mecanismos para assegurar as propriedades de segurança e, particularmente interessante no caso de redes pessoais na Internet das Coisas, preservar a privacidade dos usuários, já que seus dados serão armazenados na nuvem.

O *User-driven Privacy Enforcement for Cloud-based Services in the IoT* (UPECSI) [13] provê uma arquitetura abrangente para assegurar a privacidade do usuário em *Cloud of Things*, onde um ponto central da rede IoT é responsável por aplicar as medidas de segurança e enviar os dados à nuvem. A privacidade do usuário é assegurada por meio de uma linguagem de programação, onde os serviços de nuvem devem especificar como as informações do usuário serão utilizadas, e os usuários possuem a habilidade de habilitar ou desabilitar funções específicas do serviço de acordo com seus requisitos de privacidade.

Este trabalho busca aprimorar a segurança do UPECSI, transferindo as funções de aplicação dos mecanismos de segurança do ponto central para os próprios dispositivos IoT. Esta abordagem elimina o ponto único de falha e possibilita um controle mais granular dos dispositivos.

1.2 Objetivos

O objetivo geral deste trabalho é fornecer uma arquitetura abrangente para privacidade em *Cloud of Things*, que provê ao usuário pleno controle sob o acesso aos dados gerados pelos seus dispositivos pertencentes a Internet das Coisas e armazenados na nuvem. A arquitetura proposta deve permitir um controle fino sob os dados, visto que os mecanismos

de privacidade e controle são executados nos dispositivos ao invés de nos pontos de ligação, os quais podem representar um ponto único de falha ou componente com potencial de prejudicar as propriedades de segurança da rede uma vez comprometido por um ataque bem sucedido.

Os objetivos específicos, necessários para atingir o objetivo geral, são os seguintes:

1. Projetar uma arquitetura de *Cloud of Things* que assegure a privacidade do usuário;
2. Analisar e propor melhorias na arquitetura proposta;
3. Analisar o desempenho da arquiteturas propostas em diversos cenários, comparando-as com outras abordagens encontradas na literatura;

1.3 Organização do texto

O restante deste texto está organizado da seguinte forma. O Capítulo 2 apresenta uma introdução teórica aos principais conceitos relacionados a esse tema, que serão necessários para compreender as abordagens escolhidas no desenvolvimento da arquitetura proposta. O Capítulo 3 expõe os detalhes relacionados a integração entre Internet das Coisas e Computação em Nuvem e em seguida apresenta um estudo sobre o estado da arte referente ao assunto estudado por este trabalho. O Capítulo Seção 4 detalha a arquitetura proposta para privacidade em *Cloud of Things*, o aprimoramento realizado e também uma discussão referente aos mecanismos desenvolvidos. O Capítulo 5 apresenta uma análise analítica e experimental das propostas, comparando-as com o UPECSI e com uma abordagem sem nenhum tipo de segurança. Por fim, o Capítulo 6 conclui este trabalho, apresentando uma visão geral do que foi desenvolvido, as contribuições e trabalhos futuros.

Capítulo 2

Segurança Computacional

A interconexão entre duas tecnologias complexas, a Internet das Coisas e a Computação em Nuvem, gera novos desafios. Um dos aspectos mais importantes no surgimento de novas tecnologias é a segurança computacional, especialmente neste caso por tratar-se do armazenamento de informações (muitas vezes sensitivas) por terceiros (os provedores de nuvem). Este capítulo apresenta conceitos de segurança computacional que foram necessários para conceber a arquitetura proposta.

2.1 Propriedades de Segurança

Segundo o Instituto Nacional de Padrões e Tecnologia (do inglês *National Institute of Standards and Technology* (NIST)) dos Estados Unidos, segurança computacional é a proteção oferecida a um sistema de informação automatizado a fim de preservar a integridade, disponibilidade e confidencialidade dos recursos do sistema de informação (incluindo *hardware*, *software*, *firmware*, informações/dados e telecomunicações) [14]. Dessa forma, para que um sistema seja considerado seguro é necessário que atenda os três conceitos apresentados a seguir [11]:

1. Confidencialidade: apenas as partes autorizadas possuem acesso à informação;
2. Integridade: a informação não é modificada de forma indevida;
3. Disponibilidade: a informação é acessível quando necessário.

Outros conceitos podem ser utilizados para expandir a definição de segurança computacional. A ISO, organização não governamental destinada ao desenvolvimento e certificação de padrões, por meio da ISO 7498-2 [15], adiciona outros dois conceitos em sua definição de segurança computacional:

- Autenticidade: um subcomponente da integridade, assegura que a fonte da informação é confiável, ou seja, a mensagem é oriunda da fonte descrita;
- Irretratabilidade ou Não-repúdio: também parte da integridade, assegura a impossibilidade de negar a autoria ou recebimento de uma mensagem.

2.1.1 Confidencialidade

Confidencialidade significa assegurar que informações privadas ou confidenciais são acessíveis apenas as partes autorizadas. Este conceito envolve a privacidade das partes, uma vez que é necessário garantir que as mesmas possuam o controle necessário para definir quem terá acesso às suas informações [12].

Quanto mais complexa a tecnologia empregada na criação, processamento e apresentação de informações, mais partes são envolvidas e mais difícil se torna a garantia da confidencialidade.

Por exemplo, uma rede de sensores pessoal dirigida à automação residencial, que por sua vez se conecta à nuvem para facilitar o gerenciamento remoto pelo dono de tal rede. Neste caso são necessários vários mecanismos para assegurar que apenas as partes envolvidas, nesse caso o usuário e a nuvem, devem acessar a informação. Esses mecanismos devem abranger desde a geração e troca de mensagens locais na RSSF residencial, como a transferência e armazenamento dessa informação pela nuvem. Diferente da esfera local, onde o usuário possui controle sob seus dispositivos, a nuvem é controlada por uma unidade terceira, e dessa forma apresenta maiores desafios para a manutenção da confidencialidade. Tal confidencialidade estaria sendo violada se por acaso essa unidade terceira disponibilizasse os dados do usuário a uma outra parte não autorizada.

O conceito de confidencialidade muitas vezes pode apresentar dificuldades para ser definido, em certas circunstâcias pode-se ter dificuldades em identificar o dono da informação ou até mesmo as partes envolvidas, por exemplo, ao entrar em um *website* o computador descarrega todas as informações, desta forma essas informações podem ser consideradas do usuário ou do dono do *website*. Da mesma forma pode-se ter dificuldades em definir o que significa o acesso ao dado, por exemplo, muitas vezes apenas saber que o dado existe pode ser ainda mais relevante que o dado em si [16].

2.1.2 Integridade

Integridade significa manter o dado (ou sistema) protegido contra mudanças inadvertidas. A integridade do dado envolve seu conteúdo, a informação criada/enviada pela fonte é a mesma recebida pelo destinatário, e sua autenticidade (origem), o dado foi criado pelo seu

remetente [17]. A integridade de um sistema significa que o mesmo executa a sua função conforme a especificação, livre de mudanças inadvertidas.

Seguindo o exemplo utilizado na Seção 2.1.1, a comunicação não seria integra caso a nuvem não disponha de maneiras para assegurar que o dado enviado pelos sensores do usuário não foi alterado antes de ser recebido.

A integridade apresenta mais dificuldades em sua aplicação que a confiabilidade, para se assegurar a integridade de um dado recebido é necessário estar seguro de sua origem e de que não ouve mudança no dado durante seu envio. Muitas vezes para se assegurar estes dois itens é necessário que se tenha informações sobre a fonte do dado, o que envolve confiança nessa mesma fonte.

2.1.3 Disponibilidade

A disponibilidade de um sistema ou informação é assegurada quando o acesso a esse objeto é garantido. Em relação à segurança computacional, a disponibilidade é referente a tentativas maliciosas de negar o acesso a serviços e informações. O tempo de resposta de um sistema também deve ser considerado quando avaliada a disponibilidade do mesmo, ou seja, não só o sistema deve estar disponível, mas deve apresentar um tempo de resposta satisfatório.

Por exemplo, levando em conta um serviço de nuvem, que oferece acesso a dados de tráfego rodoviário a seus usuários. Este serviço estaria disponível apenas se, além de estar ativo, apresente tempos de resposta aceitáveis. Um ataque que afeta a disponibilidade do serviço poderia causar tanto a interrupção quanto um atraso significativo no tempo de resposta do serviço, o tornando não utilizável.

A disponibilidade pode ser um atributo difícil de ser garantido, já que serviços oferecidos pela Internet podem ser requisitados por qualquer usuário (mesmo que a requisição seja negada, o serviço processará a mesma e provavelmente enviará uma resposta de falha de acesso). Basta uma quantidade de requisições maior do que a suportada pelo sistema para que o mesmo apresente intermitência ou até mesmo cesse seu funcionamento. Para prevenir este tipo de ataque muitas vezes é utilizada redundância, ou seja, ao detectar aumento na carga do serviço servidores auxiliares são ativados.

2.2 Ameaças e Ataques

Ameaças são potenciais violações de segurança, que podem ou não se tornar reais, mas por possuir um potencial de acontecer devem ser consideradas ao se projetar mecanismos de segurança. Vulnerabilidades são falhas ou pontos fracos em um sistema que podem ser

exploradas, ou seja, ameaças são riscos que podem vir a explorar vulnerabilidades. As ações que acarretam a violação de segurança são chamadas ataques [18].

Tabela 2.1: Relação entre ameaças, ataques e propriedades de segurança.

Propriedades de Segurança	Ameaça	Ataque
Confidencialidade	Divulgação não autorizada	Exposição Interceptação Inferência Intrusão
Integridade	Fraude	Disfarce Falsificação Repúdio
Integridade e disponibilidade	Obstrução	Incapacitação Corrupção Obstrução
Integridade	Usurpação	Apropriação indevida Uso indevido

Ameaças estão relacionadas a perda de uma das propriedades da segurança computacional. De acordo com [19], a Tabela 2.1 apresenta a relação entre ameaças, ataques e propriedades de segurança relacionadas. As ameaças podem ser categorizadas em 4 classes:

- Divulgação não autorizada: ameaças à confidencialidade, consiste no acesso à informação protegida por entidades não autorizadas;
- Fraude: ameaça à integridade, ocorre quando uma entidade autorizada recebe falsos dados acreditando serem verdadeiros;
- Obstrução: ameaça à integridade e disponibilidade, é referente ao impedimento do correto funcionamento do sistema (incluindo parada completa do mesmo);
- Usurpação: ameaça à intridade, consiste no controle não autorizado de parte de um sistema.

Ataques violam a segurança computacional de um sistema por meio de ações inteligentes e deliberadas [18]. Ataques podem ser classificados como passivos e ativos. Ataques passivos consistem no monitoramento e escuta da comunicação do sistema, com o intuito de utilizar a informação coletada sem interferir nos recursos do sistema. Ataques ativos alteram o sistema, interferindo no seu funcionamento. Conforme apresentado a seguir, a RFC 4949 [18] relaciona as ameaças com os possíveis ataques.

2.2.1 Divulgação Não Autorizada

Ataques relacionados à divulgação não autorizada:

Exposição: dados sigilosos são expostos a entidades não autorizadas, pode ser intencional, quando um atacante expõe informações sensíveis a públicos não autorizados, ou não intencional, quando o mesmo acontece por falha no sistema.

Interceptação: o atacante obtém acesso à informação em uma comunicação da qual não faz parte. Os pacotes interceptados podem estar protegidos e não possibilitar o acesso a informação, mas caso não exita tal proteção o atacante terá acesso a informações sensíveis.

Inferência: por meio da análise de dados obtidos de forma autêntica o atacante obtenha informações não autorizadas. Por exemplo a análise dos padrões de tráfego de uma rede pode trazer informações sobre o conteúdo trafegado.

Intrusão: acesso à informação não autorizada por meio de acesso indevido a um sistema. Por exemplo, quando um atacante burla o controle de acesso de um sistema para obter informações sigilosas.

2.2.2 Fraude

Ataques relacionados à fraude:

Disfarce: o atacante se passa por uma entidade autorizada para ter acesso à informação. Por exemplo, o atacante obtém de alguma forma as credenciais de um usuário com acesso ao sistema e dessa forma é capaz de acessá-lo.

Falsificação: alterar ou substituir dados verdades por dados falsos. Por exemplo, um agente malicioso pode alterar seus dados bancários para obter mais recursos.

Repúdio: um usuário nega ter enviado ou ter recebido dados. O repúdio é possibilitado caso não hajam mecanismos para assegurar a origem ou destino da informação. Neste caso um atacante pode negar ter enviado mensagens maliciosas para uma vítima, por exemplo.

2.2.3 Obstrução

Ataques relacionados à obstrução:

Incapacitação: ataque à disponibilidade do sistema, pode ser realizado causando danos no hardware, na rede, ou no software do sistema. A forma mais comum é relacionada ao software, quando programas maliciosos são instalados no sistema e desabilitando parte ou todo do mesmo.

Corrupção: alterar a maneira como um sistema opera. Este ataque é referente à integridade, pois o sistema deixa de funcionar como esperado. Por exemplo um programa malicioso instalado no sistema pode prover acesso não autorizado de formas diferentes da provida pelo sistema.

Obstrução: o sistema pode ser obstruído por meio do recursos que utiliza. Por exemplo a infraestrutura de rede, é possível obstruir um sistema sobrecarregando a rede que o mesmo utiliza. O sistema também pode ser obstruído simplesmente sobrecarregando-o, por meio de uma quantidade de solicitações maior do que o mesmo possa atender.

2.2.4 Usurpação

Ataques relacionados à usurpação:

Apropriação indevida: quando um atacante controla um sistema sem a devida permissão, muito comum em computadores pessoais, onde softwares maliciosos utilizam os recursos dos computadores para realizar ataques de negação de serviço.

Uso indevido: quando uma função do sistema é alterada para realizar tarefas prejudiciais ao próprio sistema, pode ser realizada por meio de software malicioso ou agentes que obtiveram acesso indevido.

Ameaças ubíquas são apresentadas em [17], onde oito dos tipos de ameaças mais utilizadas são descritas:

Bisbilhotar (do inglês *Snooping*): um tipo de revelação, consiste em ter acesso à informação não autorizada de forma passiva. Esse acesso pode se dar por meio de interceptação de uma mensagem, ou acesso direto ao sistema. Por se tratar de um ataque passivo não ocorre modificação ou destruição da informação.

Modificação: mudança de informações sem autorização. A mudança efetuada pode ter diferentes objetivos, cobrindo três classes de ameaças: fraude, quando a informação modificada gera alguma ação da entidade que a recebeu, dessa forma a ação não corresponde à informação verdadeira; roubo e obstrução podem ocorrer caso a informação modificada seja referente ao funcionamento do sistema, dessa forma seu

correto funcionamento pode ser prejudicado ou uma entidade não autorizada pode assumir o controle do sistema.

Falsificação: refere-se à personificação de uma entidade na comunicação. Por exemplo um usuário acessa o website de seu banco para efetuar transações, para tanto é necessário inserir informações de autenticação. É possível que um agente mal intencionado falsifique a página do banco e faça com que o usuário insira suas informações em uma página clone, fornecendo seus dados ao agente e não ao banco. Neste caso a falsificação se enquadra como fraude, pois o usuário recebe informações falsas. A falsificação também pode se enquadrar como roubo, que seria o caso do agente mal intencionado utilizar as informações de autenticação do usuário para acessar sua conta e realizar transações indevidas.

A personificação também pode ser uma funcionalidade desejada em um sistema, onde uma entidade delega à outra a autoridade de realizar ações como se fosse a primeira. Diferente da falsificação, onde uma entidade se passa por outra, na delegação a entidade autorizada se apresenta como ela mesma, mas possuindo autorização referente à outra. Por exemplo, um usuário pode fornecer autorização de leitura e escrita aos seus arquivos em uma plataforma de nuvem para que outro serviço os organize. Neste exemplo o serviço não estaria se passando pelo usuário, apenas portaria uma forma de provar a autorização fornecida pelo usuário.

Negação de origem: uma forma de fraude, ocorre quando é negada a origem de uma informação. Caso não hajam mecanismos para assegurar a origem da informação um agente malicioso pode negá-la. Para que não aconteça são necessários mecanismos relacionados a integridade. Por exemplo, em um sistema bancário, caso não hajam mecanismos para garantir a integridade do sistema, um usuário pode realizar transferências e em seguida negar que as realizou, como o banco não possui mecanismos para provar que a operação foi realizada pelo usuário ele será obrigado a retornar o dinheiro ao mesmo.

Negação de recebimento: assim como a negação de origem, também é uma forma de fraude, normalmente os mesmos mecanismos utilizados para prevenir a negação de origem também previnem a negação de recebimento. Utilizando o mesmo exemplo descrito anteriormente, o dono da conta de destino da transferência também poderia negar o recebimento da quantia, e dessa forma o usuário que realizou a transferência teria que realizá-la novamente.

Atraso: o atraso é referente a interrupção temporária de um serviço. O tempo de interrupção deve ser maior que o tempo de resposta habitual do sistema. O atraso

é considerado uma forma de roubo, pois é necessário o controle de elementos do sistema ou da rede para que um atraso seja efetuado. Ataques costumam utilizar o atraso como uma forma de auxílio para um outro fim. Por exemplo, pode-se gerar atraso em uma comunicação com o servidor primário para que o usuário tente uma comunicação com o servidor secundário, que por sua vez está sob controle do agente malicioso.

Negação de serviço: a consequência da negação de serviço é uma interrupção de longo termo, assim como o atraso, é uma forma de roubo. A negação de serviço pode acontecer na fonte, ou seja, o sistema é inibido de operar corretamente, no destino, quando o usuário é impossibilitado de se comunicar com o servidor, ou no meio de comunicação, quando a comunicação é impossibilitada ao longo do caminho. Tanto a negação de serviço como o atraso podem ocorrer de forma não intencional (não causada por um agente malicioso), por exemplo, um sistema pode não comportar a quantidade de usuários que o acessa simultaneamente, causando a princípio um atraso e por fim a negação de serviço.

2.3 Mecanismos de Segurança

Mecanismos de segurança são métodos, procedimentos e ferramentas utilizadas para garantir as propriedades de segurança. A função dos mecanismos é aplicar as propriedades de segurança computacional (integridade, confidencialidade e disponibilidade) ao sistema, dado ou comunicação em questão. Em segurança computacional existem três ferramentas fundamentais para a manutenção da segurança: autenticação, controle de acesso e criptografia [16].

2.3.1 Autenticação

Autenticação é o processo de atrelar uma identidade a um sujeito [17], ou seja, verificar a identidade alegada por uma entidade [18]. A autenticação é composta por dois passos:

- Identificação: declaração de identidade do sujeito, o usuário provê sua identificação ao sistema.
- Verificação: processo de validação da identificação provida pelo usuário.

Um exemplo de autenticação é o acesso à caixa de emails, o usuário primeiro deve fornecer seu endereço de email, este é o passo de identificação. Em seguida uma senha deve ser fornecida, a senha é utilizada pois o endereço de email é público. A senha é

necessária para que apenas o usuário seja capaz de realizar a autenticação à sua caixa de emails. A senha é utilizada para a verificação e o processo de autenticação é finalizado, caso as informações fornecidas pelo usuário sejam corretas o acesso à caixa de emails é liberado.

Existem três meios de autenticar a identidade de um usuário: algo que ele possui, algo que ele sabe ou algo que ele é.

Algo que o usuário possui: baseia-se em algum item que o usuário possua consigo, por exemplo smart-cards ou tokens, que comumente possuem uma identificação gravada. Também existem meios dinâmicos, onde ocorre uma interação entre o objeto que o usuário possui e o aparelho que realiza a autenticação, por exemplo pode existir um limite de acessos gravado no smart-card e a cada acesso este limite é decrescido.

Algo que o usuário sabe: este é o meio de autenticação mais utilizado atualmente. A utilização de senhas representa algo que o usuário sabe, e neste caso é importante que seja algo secreto. A utilização de senhas apresenta algumas dificuldades, é necessário preencher a senha sempre que a autenticação for realizada, o que pode ser inconveniente; caso a senha seja revelada a outra pessoa essa passa a ter acesso imediatamente; dependendo da implementação caso a senha seja perdida ou esquecida pode ser impossível recupera-la.

Um grande problema da utilização de senhas é a possibilidade de adivinhar a senha de um determinado usuário. Vários ataques à grandes bancos de dados de usuários e senhas foram realizados, e suas informações tornadas públicas [20, 21, 22]. O estudo dessas informações possibilitou o entendimento de como usuários escolhem suas senhas[23], tornando mais fácil adivinhar as senhas.

Este problema pode ser abordado por duas frentes, na implementação do sistema e na criação da senha pelo usuário. Referente ao sistema, atualmente é comum limitar o número de tentativas para realizar a autenticação, dessa forma quando o limite é alcançado uma outra forma de autenticação é necessária, como por exemplo responder perguntas cuja resposta foi previamente gravada pelo sistema. Referente ao usuário, é importante a criação de uma senha difícil de ser adivinhada, o que pode se tornar inconveniente.

Algo que o usuário é: biometria é uma propriedade biológica baseada em alguma característica física do corpo humano. Para ser utilizada como método de autenticação a propriedade biométrica utilizada deve ser única para cada indivíduo, impossibilitando a personificação. Propriedades comumente utilizadas são: digitais, geometria da mão, retina e iris e face. A utilização biometria necessita de dispositivos especiais

para realizar as leituras, o que pode ser inconveniente de acordo com o propósito do sistema.

A combinação de métodos de autenticação pode ser utilizada para aumentar o nível de segurança do sistema [24]. Chamada de autenticação multi fatores, essa abordagem também aumenta a inconveniência do sistema. Por exemplo, atualmente muitos sistemas utilizam senhas (algo que o usuário sabe) e o envio de uma outra palavra por SMS (algo que o usuário tem), apenas quando os dois métodos são apresentados corretamente o sistema valida a identidade do usuário.

Confiança e identidade em aplicações da Internet

Transações online entre organizações e indivíduos (ou entre diferentes organizações) requerem o compartilhamento de informações de identidade. Para maior eficiência e principalmente privacidade, apenas informações realmente necessárias são trocadas. Por exemplo, um sistema de e-commerce instalado em uma plataforma de nuvem requer apenas as informações estritamente necessárias de seus clientes para realizar a venda, o que normalmente envolve nome, endereço e, dependendo da forma de pagamento, cadastro de pessoa física.

A troca de informações entre as duas partes requer a existência de um nível de confiança entre ambas. Uma técnica amplamente aplicada para estabelecer tal confiança é a utilização de um Serviço provedor de identidade, o qual, por meio de termos de serviço, estabelece um elo de confiança entre as partes. A Parte Confiante (do inglês *Relying Party* (RP)) é a organização, e necessita de um certo grau de garantia de que as informações do usuário providas pelo Serviço provedor de identidade sejam verdadeiras. Já o Serviço provedor de identidade necessita que a RP utilize os dados fornecidos de acordo com o termo de serviço e as leis vigentes. Por fim, o usuário necessita de garantias que o RP e o Serviço provedor de identidade utilizem suas informações de acordo com suas preferências e privacidade.

O *OpenID* [25] é o padrão que implementa o processo acima descrito. Um padrão aberto, possibilita a autenticação de usuários utilizando um terceiro serviço, dessa forma o website não necessita implementar seu próprio sistema de autenticação, e o usuário pode utilizar uma única identificação em vários websites.

2.3.2 Controle de Acesso

Processos de controle de acesso regulam a utilização de recursos oferecidos por um sistema apenas à entidades autorizadas[18]. São três os elementos básicos do controle de acesso: sujeitos, objetos e direitos de acesso. Sujeitos são capazes de acessar objetos, normalmente são responsáveis pelas ações realizadas nos objetos a que possuem direitos de acesso.

Objetos são recursos cujo acesso é controlado. Autorizações descrevem como sujeitos podem acessar objetos, podem incluir leitura, escrita, execução, dentre outros.

No âmbito de uma plataforma de nuvem que provê serviços de armazenamento, sujeitos seriam os usuários que acessam o banco de dados, objetos são os bancos de dados e direitos de acesso são as permissões que cada usuário possui de como manusear o banco. Por exemplo, o usuário ALINE (sujeito) possui apenas permissão de leitura (direito de acesso) ao banco de dados VENDAS (objeto).

A aplicação do controle de acesso é uma tarefa simples, basta verificar, por exemplo em um banco de dados de autorização, se o sujeito tem direitos de acesso para executar a ação desejada ao objeto. A correta atribuição de direitos de acesso aos sujeitos requer maior esforço. Para tanto deve ser definida uma Política de Controle de Acesso, que descreve quem recebe determinados direitos de acesso. Políticas de Controle de Acesso geralmente podem ser categorizadas em:

- Controle de acesso discricionário: determinado pelo proprietário do objeto, o qual atribui direitos de acesso a outros sujeitos, podendo inclusive transferir a propriedade do objeto a outro sujeito. Por exemplo, são as permissões utilizadas em sistemas Linux, onde o proprietário dos arquivos designa por meio de comandos quem mais tem direitos de acesso sob seus arquivos.

- Controle de acesso obrigatório: determinado pelo sistema, o usuário não possui permissão para autorizar outro usuário a acessar determinado objeto. Utilizado para informações sensíveis, teve origem no âmbito militar. Sujeitos e objetos possuem rótulos de sensibilidade. Rótulos de sujeitos definem um determinado nível de confiança. Rótulos de objetos definem qual nível de segurança necessário para acessá-lo.

- Controle de acesso por papéis: determinado pelo papel que o usuário desempenha no sistema, direitos de acesso são concedidos a determinados papéis, e os usuários pertencentes a estes papéis herdam os direitos. Papéis, ou grupos, podem ser acumulados por usuários.

- Controle de acesso por atributos: determinado pelos atributos do usuário, o objeto a ser acessado e condições do ambiente. Este tipo de controle de acesso apresenta grande flexibilidade: usuários, objetos e ambiente possuem atributos, por meio da análise desses três fatores e da política de acesso do sistema, um mecanismo de controle de acesso declara se a ação pode ou não ser executada. Apresenta o maior custo dentre os tipos de controle de acesso, mas pode aplicar qualquer um dos tipos previamente expostos.

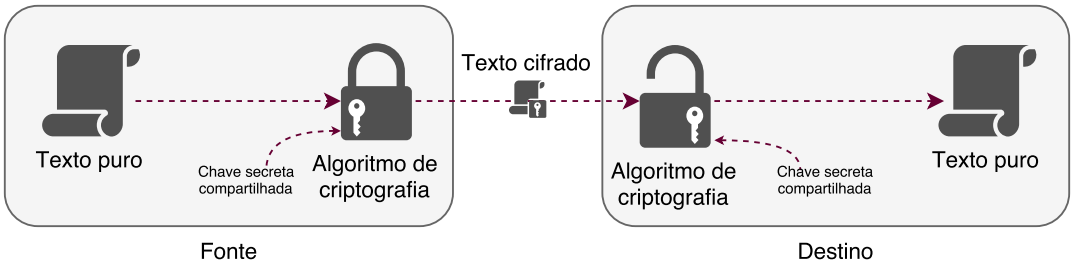


Figura 2.1: Modelo de criptografia simétrica.

2.3.3 Criptografia

Criptografia é uma ferramenta muito utilizada para prover confidencialidade em sistemas computacionais, e neste caso consiste na transformação de dados, ou seja, dados encriptados não podem ser entendidos por partes não autorizadas. A criptografia pode ser classificada em simétrica e assimétrica, a seguir são apresentados os dois tipos de criptografia, e por fim aplicações que utilizam criptografias são discutidas.

Criptografia simétrica

Criptografia simétrica utiliza uma única chave, onde ambas as partes a compartilham. A Figura 2.1 apresenta um modelo de criptografia simétrica, o texto é encriptado por um algoritmo que necessita de uma chave secreta. O receptor do texto encriptado utiliza a mesma chave secreta para decriptar o texto.

A segurança do processo está ligada principalmente ao algoritmo de criptografia e ao compartilhamento da chave secreta. O algoritmo precisa ser desenvolvido de forma que, de posse do texto puro e do texto encriptado, um atacante não consiga derivar a chave utilizada. Ambas as partes do processo devem obter a chave de forma segura e secreta, caso uma entidade não autorizada obtenha a chave qualquer criptografia realizada com a mesma perde sua confidencialidade.

O processo de encriptação pode ser comprometido por meio de criptoanálise e força bruta. A criptoanálise é baseada nas propriedades do algoritmo de criptografia, o objetivo é encontrar a chave ou o texto puro por meio de dedução baseada em análise de dados como pares de texto puro e encriptado. A força bruta tenta encontrar a chave por meio de exaustão, tentando todas as chaves possíveis, é importante notar que para que a força bruta seja bem sucedida é necessário ter o conhecimento de como identificar o texto puro, e esta tarefa deve ser automatizada.

Algoritmos de criptografia podem operar em bloco ou em fluxo. Criptografias de bloco processam o texto puro em pedaços (ou blocos) de tamanho predefinido, cada bloco é encriptado utilizando a mesma chave, por fim o texto encriptado consiste na concatenação dos blocos encriptado. Criptografias de fluxo processam o texto puro de forma continua,

um byte por vez (ou bit, ou qualquer outro elemento). A chave é utilizada como entrada em um gerador de dígitos pseudo-aleatório, que por sua vez é combinado ao texto puro por meio da operação lógica de disjunção exclusiva (XOR). Apesar de as criptografias de fluxo serem quase sempre mais rápidas, devido a sua simplicidade de implementação, as criptografias de bloco são amplamente mais utilizadas, pois suas chaves podem ser reutilizadas, facilitando seu gerenciamento.

A seguir são descritos os algoritmos de criptografia simétrica mais utilizados:

***Data Encryption Standard* (DES)** Uma criptografia de bloco, é utilizada como padrão pelo governo dos Estados Unidos desde 1977. Opera em blocos de 64 bits com chaves de 56 bits. Muitos trabalhos tentaram comprometer o algoritmo por meio de criptoanálise, no entanto ainda não houve nenhum relato de falha fatal no algoritmo [26]. Já o tamanho da chave do DES tornou-se um problema com o aumento da capacidade computacional dos equipamentos. Ataques de força bruta são muito eficazes contra o DES devido o tamanho de sua chave, um computador pessoal é capaz de adivinhar uma chave de 56 bits em 1 ano [27], enquanto supercomputadores podem realizar essa tarefa em apenas 1 hora [28]. Este problema foi resolvido com a criação do DES Triplo, onde o algoritmo do DES é repetido utilizando 2 ou 3 chaves, dessa forma o tamanho da chave passa a ser de 112 ou 168 bits, mas com segurança de 80 ou 112 bits, respectivamente.

***Advanced Encryption Standard* (AES)** Originalmente chamado de Rijndael [29], sucedeu o DES como padrão em 2005. Suporta chaves de 128, 192 e 256 bits. Assim como o DES, não há até hoje um ataque de criptoanálise que possa ser aplicado de forma prática. A motivação a substituição do Triplo DES não foi a segurança, mas sim eficiência e facilidade de implementação, dessa forma o AES oferece melhor desempenho em hardware e software [30]. O tamanho da chave do algoritmo torna ataques de força bruta impraticáveis com o poder computacional atual, uma chave de 128 bits demoraria $5,3 * 10^{17}$ anos para ser quebrada por um supercomputador [30].

Criptografia assimétrica

A criptografia assimétrica, também chamada de chave pública, baseia-se na utilização de duas chaves, uma pública e outra privada. O texto encriptado por uma chave pode apenas ser decriptado pela outra. A Figura 2.2 apresenta um exemplo de criptografia assimétrica, onde a chave pública é utilizada para encriptar texto puro e a privada para decriptar o texto encriptado. Neste caso se a chave privada é de posse apenas do destino e a chave

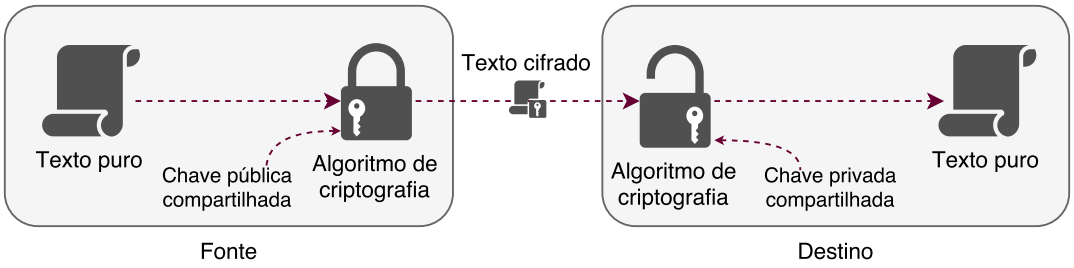


Figura 2.2: Modelo de criptografia assimétrica.

pública é de conhecimento compartilhado, apenas o destino será capaz de decriptar a mensagem, desta forma fornecendo confidencialidade à comunicação.

A criptografia de chave pública não substitui a criptografia simétrica, ambas são amplamente utilizadas em conjunto, na construção de protocolos de segurança para várias finalidades, as principais utilizações de criptografia assimétrica são assinaturas digitais e distribuição de chaves simétricas:

Assinatura digital Chaves assimétricas podem ser utilizadas para autenticação, por exemplo, a fonte assina a mensagem com sua chave privada, dessa forma é assegurada sua autenticidade, pois apenas a fonte possui tal chave. Note que neste caso não há confidencialidade, pois todos possuem a chave pública da fonte. Este exemplo também possui uma falha, uma entidade pode distribuir sua chave pública alegando qualquer identidade, para resolver este problema certificados são utilizados. Uma Autoridade Certificadora, que deve ser confiável perante a sociedade, é responsável por assegurar que a chave pública pertence à entidade que a possui. O protocolo X.509 [31] é amplamente utilizado para fornecer assinaturas digitais.

Troca de chaves simétricas A troca de mensagens confidenciais utilizando criptografia simétrica exige que as duas partes entrem em acordo a respeito de uma chave de forma confidencial. A criptografia de chave pública é amplamente utilizada para este processo. Por exemplo, basta que a fonte cifre uma chave simétrica com a chave pública do destino, dessa forma apenas o destino possuirá tal chave, e todas as mensagens podem ser encriptadas simetricamente. Este procedimento é amplamente utilizado por protocolos de segurança, como o *Transport Layer Security* (TLS) [32].

2.4 Considerações Finais

Este capítulo apresentou um apanhado dos principais conceitos sobre segurança computacional, como pode ser visto este tópico é extremamente abrangente e já foi muito estudado durante toda a história da computação. Os conceitos apresentados são necessários para o

entendimento da proposta deste trabalho, onde métodos de autenticação, de criptografia, dentre outros, são amplamente utilizados.

Capítulo 3

Cloud of Things: Integração de Internet das Coisas e Computação em Nuvem

Neste capítulo serão abordadas as tecnologias que possibilitam o surgimento da Cloud of Things (CoT): a Internet das Coisas e a Computação em Nuvem. Por fim, trabalhos relacionados a arquitetura proposta são apresentados, com destaque para o trabalho em que este estudo se baseia, o *User-driven Privacy Enforcement for Cloud-based Services in the IoT* (UPECSI) [13].

3.1 Internet das Coisas

O paradigma de Internet das Coisas (do inglês, Cloud of Things (CoT)) estabelece a conexão de vários objetos ("coisas") à Internet, tais como eletrodomésticos, sinais de transito, os mais diversos sensores, roupas, etc. É previsto que hajam bilhões de objetos conectados a Internet, fornecendo uma quantidade enorme de informação em diversas áreas de aplicação [33]. A Internet das Coisas terá um grande impacto na vida de seus usuários, a disponibilidade de informações antes inexistentes traz muitas possibilidades. Áreas como automação residencial, assistência na saúde e aprendizado virtual são exemplos de áreas domésticas onde a Internet das Coisas terá grande impacto. No campo de atividades comerciais, podemos listar automação industrial, logística, transporte inteligente, dentre outros [34].

O impacto advindo da Internet das Coisas é tamanho que o Conselho Nacional de Inteligência (do inglês, *National Inteligence Council*) dos Estados Unidos previu que essa será uma das 6 tecnologias com potencial impacto até 2025 [35].

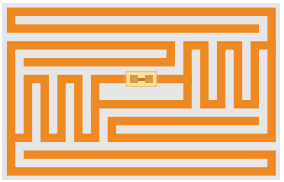
3.1.1 Tecnologias

A Internet das Coisas apresentará grande heterogeneidade quando se refere às tecnologias utilizadas pelos dispositivos, tanto em relação ao hardware quanto ao software que tais dispositivos irão empregar. A seguir são descritas as principais tecnologias que possibilitarão a Internet das Coisas.

Hardware

Muitas tecnologias de hardware que irão possibilitar a IoT já existem, são pequenos dispositivos de hardware que permitem a comunicação sem fios, normalmente com pouca ou nenhuma capacidade computacional. Identificador de Rádio Frequência (do inglês, *Radio-Frequency Identification* (RFID)) e Redes de Sensores Sem Fio (RSSF) são as duas principais tecnologias [36].

RFID é uma tecnologia de comunicação de pequena distância. A Figura 3.1 apresenta os dois componentes do RFID: etiquetas e leitores. As etiquetas possuem uma identificação única e podem ser anexadas a objetos ou até a pessoas, são objetos pequenos que podem ou não possuir poder computacional e fonte de energia, seus principais atrativos são o baixo custo e pequeno tamanho. Os leitores são responsáveis por ativar as etiquetas ao redor, energizando-as por meio de campos eletromagnéticos (no caso de etiquetas passivas, sem fonte de energia), através dos leitores é possível receber a identificação de todas as etiquetas RFID que se encontram na sua área de cobertura [37]. RFIDs podem ser utilizadas em diversas aplicações, de fato um objeto com uma etiqueta RFID passa a ser mapeado digitalmente, oferecendo possibilidades como melhor controle logístico e cartões de banco.



a) Etiqueta RFID

b) Leitor RFID

Figura 3.1: Exemplo de dispositivos RFID: etiquita (a) e leitor (b)

Redes de Sensores sem Fio, são formadas por pequenos dispositivos dotados de unidade de processamento, antena para comunicação sem fio, e possivelmente sensores que possibilitam a tradução de informações do meio físico para o meio digital. Muitas aplicações de RSSF exigem dispositivos baratos, pequenos e com fonte de energia limitada, dessa

forma tais dispositivos apresentam baixo poder computacional e capacidade de armazenamento. RSSFs possuem alta escalabilidade, permitindo cenários com grandes quantidades de dispositivos em uma só rede, tais como monitoramento de áreas e de clima, e pequenas redes como redes domésticas de automação e cuidado com a saúde [38].

Software

IoT irá gerar uma quantidade imensa de dados, vindos de diferentes dispositivos. *Middleware* é uma camada de software entre os níveis tecnológicos e de aplicação, padronizando os dados vindos de diferentes plataformas. *Middlewares* serão utilizados na IoT para facilitar a agregação dos dados vindos de diferentes dispositivos [39]. Muitos trabalhos propõem *middlewares* projetados especialmente para a Internet das Coisas [40, 41]. Particularmente, a arquitetura dos *middlewares* propostos para IoT utiliza o esquema de orientação a serviço (do inglês, *Service Oriented Architecture* (SOA)). O modelo SOA dita que as funcionalidades de uma aplicação devem ser disponibilizadas na forma de serviços, acessados através de protocolos de comunicação padronizados, características que trazem benefícios à IoT.

A pilha de rede das RSSFs também está em franco desenvolvimento, a princípio muitos protocolos foram desenvolvidos para essas redes, voltados principalmente ao baixo consumo de energia [42, 43]. Atualmente um esforço é realizado para se padronizar a pilha de rede das RSSFs, especialmente trazendo protocolos já utilizados na Internet [44], possibilitando dessa forma a comunicação direta dos sensores com a Internet. Na camada de rede, o 6LoWPAN [45] efetua compressão e encapsulamento de cabeçalhos IPv6 [46] para que caibam em quadros do protocolo IEEE 802.15.4. Na camada de transporte, é indicada a utilização do protocolo UDP [47], que apesar de não oferecer funcionalidades de entrega confiável, ordenada e com checagem de erros como o TCP[48], apresenta um *overhead* muito menor. Apesar dos sensores poderem se comunicar diretamente com outros dispositivos da internet, um gateway ainda pode ser utilizado para efetuar operações muito custosas para os pequenos dispositivos.

3.1.2 Segurança

A Internet das Coisas traz grandes desafios na área de segurança, os pequenos dispositivos que se conectarão a Internet irão prover informações muitas vezes sensitivas, como por exemplo a localização de indivíduos e objetos pessoais. Portanto mecanismos de proteção para a comunicação de dispositivos IoT e a Internet são essenciais para possibilitar a adoção desse paradigma.

Como mencionado, a comunicação na Internet das Coisas acontece principalmente de forma sem fio. A utilização de um meio compartilhado necessita que toda comunicação sensitiva seja protegida, o que é alcançado por meio de criptografia. Entretanto, operações criptográficas e troca de chaves podem ser muito custosas para dispositivos da IoT, dessa forma é necessário que algoritmos de criptografia sejam mais rápidos e consumam menos energia [49]. Uma estratégia muito utilizada é a adoção de hardware especializado em criptografia, dessa forma tais operações são executadas em tempos praticáveis e seu consumo de energia é reduzido [50].

A quantidade e heterogeneidade dos dispositivos traz desafios para autenticação e autorização em IoT. Como já descrito, existem muitos métodos para se realizar tais tarefas, mas devido à complexidade das redes IoT tais métodos podem nem sempre ser aplicados. Alguns trabalhos tentam solucionar esses problemas, como [51], que utiliza uma terceira parte confiável que é responsável por realizar muitas tarefas em nome dos dispositivos.

É evidente que a difusão de dispositivos com capacidade de sensoriamento e comunicação com a Internet traz uma grande preocupação em relação à privacidade. A quantidade de diferentes tecnologias, assim como as diversas formas de armazenar e transmitir os dados gera um grande desafio à esta área. É de grande importância que os dados possuam identificação de propriedade e que sejam manipulados por terceiros de forma transparente, respeitando sempre os requerimentos do usuário. Neste quesito políticas de privacidade podem ser utilizadas nos dispositivos, aplicando as regras fornecidas por seus proprietários e prevenindo que dados sejam compartilhados com entidades não autorizadas [52].

3.2 Computação em Nuvem

A computação em nuvem pode ser descrita como um modelo que possibilita o acesso ubíquo a diversos recursos computacionais de forma flexível, tais como servidores, armazenamento, infraestrutura de rede, etc. Esse serviço deve ser fornecido de forma ágil e com o menor esforço gerencial possível [53]. Uma definição mais concisa é fornecida por [54]: computação em nuvem é uma forma especializada de computação distribuída que inclui modelos de utilização para fornecimento remoto de recursos escaláveis e mensuráveis.

A concepção da computação em nuvem foi motivada por alguns problemas rotineiramente encontrados na indústria, tais como planejamento de capacidade, redução de custos e agilidade organizacional [54]. Em muitas situações uma aplicação pode possuir raros picos de utilização que demandam recursos muitas vezes maiores que o comum, dessa forma é necessário que a infraestrutura disponível seja projetada para atender tais picos, levando a um aumento no custo de implantação. O aumento das soluções computacionais utilizadas por empresas gera também um aumento no custo necessário para implemen-

tar tais soluções, estes custos envolvem a contratação de mão-de-obra qualificada para manter a infraestrutura operacional, utilização de recursos como energia elétrica, aquisição de atualizações de software e hardware, dentre outros. Por fim, a implementação de hardware e software são processos que podem levar certo tempo, e muitas vezes esse tempo pode prejudicar a empresa, fazendo-a perder oportunidades de crescimento, dessa forma é desejado a maior responsividade possível em termos de implementação de recursos computacionais.

Atualmente a computação em nuvem representa um modelo amplamente utilizado, tanto em atividades comercias quanto pessoais. O fornecimento de recursos computacionais sob demanda pode reduzir dramaticamente o investimento necessário para a realização de diversas atividades comerciais, pois o custo de utilizar serviços de nuvem é comumente muito menor do que adquirir e manter a infraestrutura necessária para implementar tais recursos localmente [55]. Já na área pessoal a computação em nuvem trouxe aplicações convenientes aos usuários, tais como o armazenamento de dados online.

3.2.1 Entidades

A computação em nuvem envolve diversas entidades, de acordo com a forma e o tipo de serviço oferecido. O provedor de nuvem é a entidade que disponibiliza os recursos operacionais a serem utilizados pelos clientes. Clientes utilizam os serviços de nuvem disponibilizados pelo provedor. Muitas vezes a plataforma oferecida por provedores de nuvem possibilita que serviços de nuvem sejam de propriedade de terceiros. Normalmente as regras de utilização dos serviços e provedores, assim como as características do serviço prestado, são estabelecidas por acordos de nível de serviço (do inglês, *Service Level Agreement* (SLA)) [56].

3.2.2 Características

Um ambiente em nuvem precisa implementar atributos específicos para prover recursos computacionais escaláveis e mensuráveis de forma remota e eficiente. As seguintes características são comuns a maioria dos provedores de computação em nuvem: utilização sob demanda, acesso ubíquo, múltiplos inquilinos, elasticidade, uso mensurável e resiliência [54].

Utilização sob demanda: o cliente deve ter acesso, de forma unilateral, a mecanismos que possibilitem a utilização de recursos na nuvem. A utilização de tais recursos deve ser automática após sua configuração pelo cliente, fornecendo uma utilização sob demanda dos recursos oferecidos pelo provedor de nuvem.

Acesso ubíquo: um serviço de nuvem deve ser amplamente acessível. O acesso à nuvem deve ser possibilitado em diversos dispositivos, tais como computadores de mesa e *smartphones*, e tecnologias, como diferentes navegadores de Internet, sistemas operacionais, etc.

Múltiplos inquilinos (do inglês, *multi-tenancy*): requer que um provedor de nuvem possibilite a utilização de seus recursos (*hardware* e *software*) por múltiplos clientes, de forma que instâncias de diferentes usuários sejam isoladas. Esta característica também envolve o agrupamento de recursos, onde o provedor atribui seus recursos de forma dinâmica de acordo com a demanda.

Elasticidade: os recursos fornecidos pela nuvem devem ser automaticamente escaláveis de acordo com condições de tempo real ou pré-configuradas. Esta é uma das principais vantagens da computação em nuvem, pois está associada à redução de custos se comparada a implementação de infraestrutura local.

Uso mensurável: os recursos oferecidos pelo provedor devem ser mensuráveis, dessa forma os clientes possuem a habilidade de rastrear os recursos utilizados. Esta característica possibilita a cobrança apenas dos recursos utilizados pelo cliente.

Resiliência: o provedor deve implementar mecanismos para que os recursos oferecidos sejam resilientes a problemas de funcionamento. Provedores de nuvem devem possuir recursos redundantes, em diferentes localidades físicas, que passam a ser ativados (de forma automática) quando o recurso primário apresenta alguma deficiência.

3.2.3 Modelos de Serviço

O modelo de serviço de uma nuvem representa o tipo de recurso oferecido a seus clientes, são 3 os principais modelos de serviço: infraestrutura como serviço (do inglês, *Software as a service* (SaaS)), plataforma como serviço (do inglês, *Platform as a service* (PaaS)) e software como serviço (do inglês, *Infrastructure as a service* (IaaS)) [53]. No entanto muitos outros modelos são utilizados pelo mercado atualmente, sendo normalmente apenas uma especialização dos 3 modelos acima citados, como por exemplo o termo Banco de dados como Serviço, que é uma especialização do PaaS [54].

Infraestrutura como serviço: neste modelo de serviço o provedor de nuvem oferece recursos computacionais ligados a infraestrutura, tais como processamento, armazenamento, rede, etc. O cliente normalmente utiliza esse serviço para implantar softwares de sua escolha, ficando responsável por manter seu próprio sistema.

Plataforma como serviço: um provedor que oferece plataforma como serviço tipicamente fornece um ambiente pronto para uso, composto por recursos já instalados e configurados. Tais recursos são tipicamente ligados a ambientes prontos para a implantação de programas pelo cliente.

Software como serviço: fornece aplicações específicas que são executadas na infraestrutura em nuvem, tipicamente disponibilizado comercialmente. O cliente possui apenas controle sobre configurações limitadas da aplicação, sem disponibilização de opções referentes à administração do software.

3.2.4 Modelos de Implementação

O modelo de implementação de uma nuvem representa o tipo de ambiente referente principalmente à propriedade, tamanho e acesso. São quatro os modelos de implementação de uma nuvem: pública, comunitária, privada e híbrida [53].

Nuvem pública acessível ao público em geral, normalmente comercializada a um certo custo de acordo com as funcionalidades oferecidas e à quantidade de recursos utilizada pelo cliente;

Nuvem comunitária acessível à uma comunidade com interesses em comum, apenas usuários pertencentes a tal comunidade possuem acesso à plataforma, pode ser operada por uma ou mais entidades pertencentes a comunidade ou mesmo por empresas terceiras;

Nuvem privada de propriedade de uma única entidade, que opera e gerencia a infraestrutura em nuvem. Possibilita a descentralização dos recursos computacionais de uma organização sem que seus dados estejam de posse de terceiros;

Nuvem híbrida infraestrutura composta por duas ou mais dos outros tipos de nuvem, que continuam como entidades separadas mas propiciam a portabilidade de dados e aplicações.

3.2.5 Segurança

As características e modelos da computação em nuvem são possibilitados por diversas tecnologias, como virtualização, que introduzem ameaças e vulnerabilidades específicas em relação à computação tradicional [57].

Os modelos de serviços utilizados na nuvem são dependentes, aplicações SaaS são implantadas sob o PaaS, e o PaaS é dependente do IaaS. O mesmo ocorre com a segurança

da arquitetura, dessa forma vulnerabilidades no IaaS são propagadas aos outros modelos de serviço. Por exemplo, caso um atacante obtenha controle de uma plataforma na nuvem, muito provavelmente os serviços SaaS que utilizam tal plataforma também estarão comprometidos [58].

Relacionada à segurança, a privacidade é uma preocupação primária em computação em nuvem, pois os dados e aplicações do cliente passam a residir na nuvem, que é de propriedade do provedor. Desta forma existem riscos relacionados ligados a revelação de dados confidenciais, tais como informações financeiras, e informações pessoais.

Para prevenir o acesso aos dados pelo próprio provedor de nuvem estudos sugerem a utilização de criptografia homomórfica [59], onde é possível realizar operações nos dados cifrados, dessa forma o dado é armazenado de forma cifrada no provedor. A criptografia homomórfica atualmente pode ser aplicada para operações simples, como soma, mas para operações mais complexas exige um grande custo computacional, impossibilitando seu uso generalizado.

Políticas de privacidade ditam como o provedor deve lidar com os dados do cliente, sendo essencial que hajam garantias de que a política está sendo aplicada corretamente [60].

3.3 Cloud of Things

O modelo de serviços utilizado na computação em nuvem pode ser estendido para a integração com a Internet das Coisas, dessa forma as funcionalidades advindas da IoT podem ser oferecidas como serviço na nuvem. Por exemplo, sensores de temperatura espalhados pela cidade podem enviar seus dados para a nuvem, onde um serviço fornece, de maneira ubíqua e segura, acesso a essas informações para usuários em diversas plataformas.

A Figura 3.2 apresenta a arquitetura *Cloud of Things*, diversas redes IoT, homogêneas e heterogêneas, se comunicam com a nuvem, que armazena os dados e oferece serviços para processamento e apresentação. O usuário utiliza os serviços oferecidos pela nuvem para acessar informações relacionadas às redes IoT.

3.4 Trabalhos Relacionados

A seguir são apresentadas pesquisas relacionadas ao tema deste trabalho. O estado da arte em segurança e privacidade é apresentado para a Internet das Coisas, Computação em Nuvem e, por fim, *Cloud of Things*.

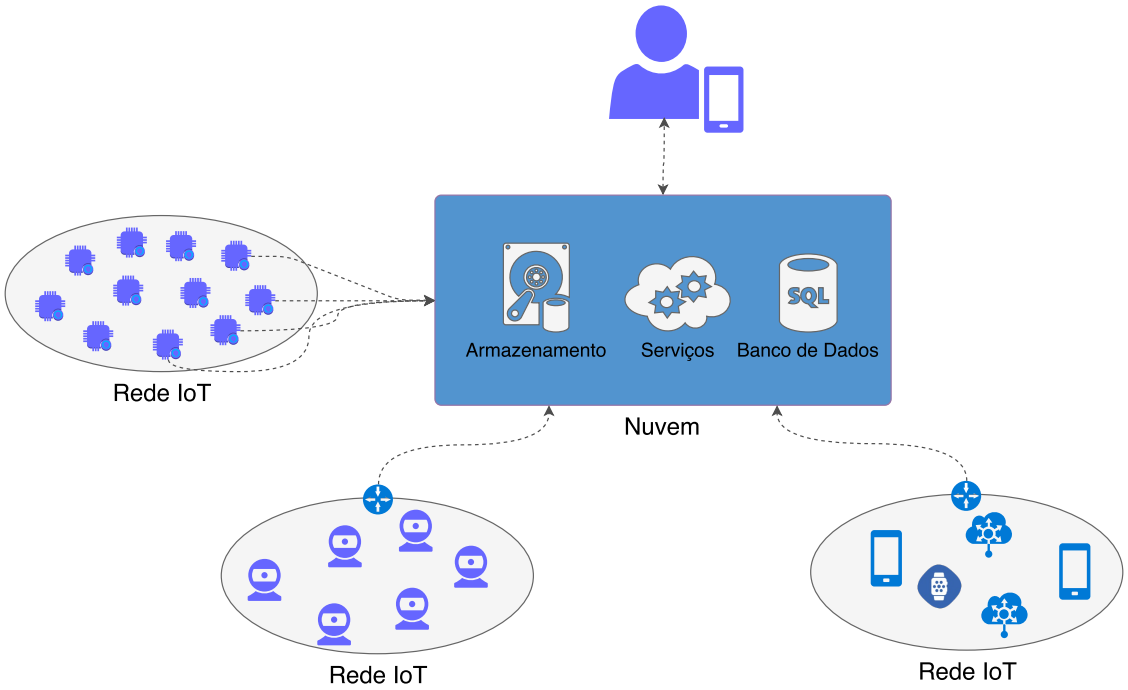


Figura 3.2: Arquitetura *Cloud of Things*

3.4.1 Segurança e Privacidade em IoT

A implementação de mecanismos de segurança na Internet das Coisas envolve muitos desafios, tais como a heterogeneidade e a limitação de recursos computacionais dos dispositivos. A Internet das Coisas irá gerar uma grande quantidade de informação, inclusive dados sensíveis, tais como dados pessoais e corporativos. Diversos trabalhos propõem protocolos e *frameworks* para prover segurança às comunicações dos dispositivos IoT.

Dorri *et al.* [61] apresenta uma arquitetura para segurança e privacidade em IoT baseada em *blockchain* [62]. *Blockchain* é uma estrutura de dados distribuída entre membros de uma rede [63]. Todas as comunicações são catalogadas em uma *blockchain* mineirada pelo *gateway*, dessa forma a integridade dos dados trafegados é garantida. A comunicação de dispositivos fora da rede local é realizada através de uma rede virtual privada (do inglês, *Virtual Private Network* (VPN)). Não são abordados os protocolos das camadas inferiores neste trabalho e nem a comunicação do *gateway* com a Internet. Pacheco *et al.*. [64] propõem um *framework* de segurança para a Internet das Coisas para casas e prédios inteligentes. Mecanismos de segurança são implementados no *gateway* para detectar anormalidades nos dispositivos através da análise dos dados enviados. De acordo com a classificação da anormalidade detectada uma ação é tomada, por exemplo, descarte do dado ou autenticação do dispositivo.

SecIoT [65] é um *framework* que fornece autenticação e controle de acesso. Através de uma unidade central (*gateway*) os dispositivos realizam a autenticação utilizando ca-

nais secundários (por exemplo, e-mail ou SMS), evitando a utilização de criptografia. O controle de acesso é definido pelo mapeamento entre papeis e regras de acesso.

Os trabalhos acima utilizam amplamente uma unidade central (ou *gateway*), tanto para executar os mecanismos de segurança quanto para realizar a comunicação das redes IoT com a Internet. Atualmente uma abordagem diferente está sendo discutida, onde os próprios dispositivos, quando dotados dos recursos computacionais necessários, realizam a comunicação direta com a Internet. Esta tendência pode ser notada pela intensa atividade de entidades padronizadoras, como o Instituto de Engenheiros Elétricos e Eletrônicos (IEEE) e o Grupo de Trabalho de Engenharia da Internet (do inglês, *Internet Engineering Task Force* (IETF)). Protocolos de comunicação e segurança estão sendo desenvolvidos tendo em conta as limitações dos dispositivos IoT e garantindo a interoperabilidade com padrões existentes da Internet [66].

Atualmente os padrões desenvolvidos ou ainda em desenvolvimento possibilitam a pilha de rede apresentada na Figura 3.3. O padrão IEEE 802.15.4 [67] define as camadas física e de enlace, responsáveis pela comunicação ponto a ponto entre dispositivos limitados. Este padrão foi primeiramente lançado em 2003, apresentando diversas atualizações e emendas, dessa forma atualmente é o padrão utilizado pela grande maioria dos dispositivos. Mecanismos de segurança são opcionais, utilizando criptografia simétrica (AES) com chaves de até 128 bits é possível obter confidencialidade, autenticidade e integridade. Não é abordada a distribuição e gerenciamento das chaves.

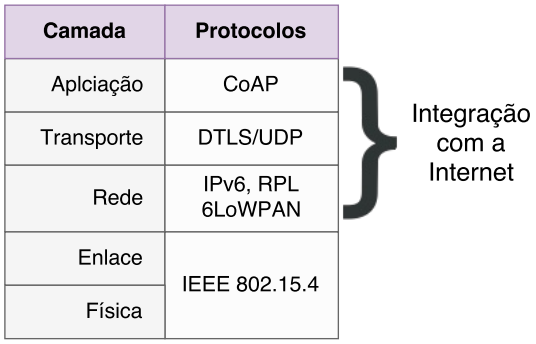


Figura 3.3: Pilha de rede da IoT.

O protocolo IPv6 [46] é o padrão responsável pelo endereçamento na camada de rede, seu espaço de endereçamento suporta bilhões de identificadores, possibilitando a identificação única de todos os dispositivos da IoT. Originalmente desenvolvido para uso da Internet, não foram consideradas as limitações dos dispositivos IoT, dessa forma uma adaptação é necessária. O 6LoWPAN [45] foi então concebido para realizar a compressão dos quadros IPv6, possibilitando sua utilização em redes IEEE 802.15.4. Atualmente não há mecanismos de segurança no 6LoWPAN. O protocolo Routing Protocol for Low power and Lossy Networks (RPL) [68] define o roteamento para dispositivos limitados.

Um *framework* adaptável é utilizado, desta forma o roteamento pode ser configurado de acordo com a aplicação da rede IoT. Esta abordagem foi utilizada para diminuir o consumo de recursos computacionais dos dispositivos. Através dos mesmos algoritmos de criptografia utilizados pelo IEEE 802.15.4, o RPL provê confidencialidade, autenticidade e integridade.

A limitação dos dispositivos IoT exige a utilização do protocolo *User Datagram Protocol* (UDP) [47], que apesar de não oferecer funcionalidades de entrega confiável, ordenada e com checagem de erros como o *Transmission Control Protocol* (TCP) [48], apresenta cabeçalhos menores e menor quantidade de mensagens de controle necessárias para o envio de uma mensagem. Na camada de aplicação o *Constrained Application Protocol* (CoAP) [69] é um protocolo de troca de mensagens para dispositivos e redes limitadas, visto que é muito similar ao *Hypertext Transfer Protocol* (HTTP) [70], a tradução entre os protocolos é de simples realização. O CoAP suporta mecanismos de segurança através do protocolo *Datagram Transport Layer Security* (DTLS) [71].O DTLS é uma adaptação do *Transport Layer Security* (TLS) [72] desenvolvido para fornecer segurança ao protocolo UDP, dessa forma confidencilidade, autenticação, integridade, não-repúdio e proteção contra ataques de repetição são providos. A adoção do DTLS em redes IoT ainda é uma questão aberta, algumas de suas funcionalidades impõe grande carga nos dispositivos, limitando a abrangência de sua utilização. Atualmente muitos trabalhos buscam adaptar o DTLS para redes limitadas, particularmente interessante, uma adaptação está sendo estudada pelo grupo de trabalho *DTLS In Constrained Environments* (DICE) [73].

A padronização dos protocolos de comunicação para a Internet das Coisas está em pleno desenvolvimento, com muitos desafios a serem abordados. Devido ao seu alto custo computacional, a utilização de criptografia (principalmente assimétrica) para prover segurança é um dos maiores desafios. Protocolos e mecanismos propostos buscam limitar a utilização de criptografia apenas ao extremamente necessário.

A implementação de mecanismos de segurança para dispositivos IoT apresenta diversos desafios, principalmente quando implementados para dispositivos de baixo poder computacional e com fonte de energia limitada. Kothmayr *et al.* [74] apresenta uma avaliação do protocolo DTLS [71], que implementa um canal seguro utilizando o UDP, o uso deste protocolo em dispositivos limitados está sendo ativamente estudado no momento. Resultados mostram que, apesar de promissor, este protocolo precisa de ajustes para que sua utilização se torne viável em dispositivos severamente limitados. Zhang *et. al* [1] avalia diversas implementações do sistema criptográfico AES, que é o sistema utilizado pela Internet. Uma plataforma bastante limitada foi utilizada, possuindo um micro controlador de apenas 8MHz, 128 KB de memória RAM e 512 KB de memória ROM. Atraso e energia foram avaliados, e os resultados mostram que, apesar do impacto não ser desprezível,

plataformas bastante limitadas podem utilizar este protocolo de criptografia simétrica.

3.4.2 Segurança e Privacidade em Computação em Nuvem

Devido a sua grande relevância para a adoção da computação em nuvem, segurança e privacidade são aspectos que vem sendo amplamente estudados [75, 76, 77, 78, 79]. Entretanto, ambos aspectos ainda possuem diversos desafios a serem explorados [80]. A seguir são apresentados alguns trabalhos que buscam prover privacidade no armazenamento de dados na nuvem e controle de acesso.

Sagumar *et al.* [81] propõe um algoritmo de criptografia simétrica simples e leve para armazenamento de dados encriptados na nuvem. Através de uma heurística que requer baixo processamento um texto puro é cifrado com duas chaves. O algoritmo é disponibilizado em forma de um serviço de nuvem, e as chaves não são reveladas ao provedor. No entanto o estudo não fornece uma análise criptográfica do algoritmo proposto, dessa forma a real segurança do mecanismo não é comprovada.

Key-Aggregate Cryptosystem (KAC) [82] fornece um sistema criptográfico assimétrico onde é possível derivar chaves privadas (responsáveis por decriptar o texto cifrado) compatíveis com apenas um sub conjunto de todos os dados cifrados com a chave privada principal. Ao cifrar um dado o seu proprietário deve indicar uma classe (ou índice) ao mesmo, dessa forma é possível criar uma chave capaz de decriptar apenas dados da classe respectiva. A abordagem proposta apresenta uma forma de fornecer controle de acesso através de chaves criptográficas, dessa forma apenas parte dos dados armazenados na nuvem podem ser acessados por um serviço. No entanto a utilização de criptografia assimétrica resulta em um custo computacional relativamente alto quando considerado que dispositivos limitados realizarão tal operação.

Block Based Sharing Scheme (BSS) [83] apresenta uma técnica de criptografia voltada a dispositivos móveis para armazenamento na nuvem. A chave criptográfica é gerada através de um algoritmo que tem como entrada a senha do usuário, que é então modificada e estendida. Os arquivos a serem armazenados na nuvem são divididos em blocos, cada bloco passa por uma operação de exclusão mútua (XOR) com a chave gerada previamente. A integridade dos dados pode ser verificada por meio do *hash* do nome do arquivo, senha criptográfica e tamanho do arquivo. A proposta apresentada visa fornecer um algoritmo leve para armazenamento criptográfico na nuvem, que de fato é alcançado pela utilização da operação XOR nos dados enviados. No entanto não há distinção entre as senhas utilizadas em cada arquivo, desta forma o detentor da senha tem acesso a todos os arquivos, não havendo possibilidade de restringir o acesso a apenas parte dos arquivos.

As propostas aqui apresentadas indicam a utilização de algoritmos de criptografia simples para o armazenamento de dados cifrados na nuvem. Esta abordagem, apesar

de fornecer uma alternativa de baixo consumo de recursos computacionais, não possui a alta confiabilidade provida por algoritmos de criptografia simétrica padrões, como o AES, que já passou por diversos estudos de criptoanálise sem ser comprometido. Portanto é possível identificar uma relação de custo-benefício referente à técnica de criptografia a ser utilizada e a confidencialidade obtida.

3.4.3 Segurança e Privacidade em CoT

O conceito de *Cloud of Things* envolve a integração de vários elementos, aumentando a complexidade das soluções de segurança que precisam ser adotadas [84, 10]. Em redes IoT domésticas, o armazenamento dos dados do usuário na nuvem traz um problema relacionado a privacidade, pois a partir deste momento os dados estão sob controle de uma entidade diferente, e portanto podem ser utilizados para fins que o usuário não aprova [8].

A integração de redes IoT com a nuvem é estudada a algum tempo, a maioria das contribuições tem foco em desenvolver métodos e ferramentas para coletar e armazenar uma grande quantidade de dados, controlar acesso aos dados coletados e gerenciar os dispositivos IoT. *OpenIoT* [85] é um *middleware* de código aberto projetado para facilitar a construção e gerenciamento de redes IoT. Este *middleware* fornece meios de gerenciar dados coletados de diferentes tipos de dispositivos, por meio de uma abstração de sensor virtual. Os dados coletados dos dispositivos são transmitidos para nuvem, onde podem ser explorados com o auxílio de anotações semânticas, que fornecem informações a respeito dos dados coletados. *OpenIoT* também fornece um conjunto de serviços de nuvem que proporcionam diversas funcionalidades aos usuários, tais como visualização e monitoramento da plataforma IoT. Este projeto não define protocolos de comunicação específicos entre seus componentes, mas abstrai a comunicação, possibilitando a utilização de diversos protocolos. A comunicação segura pode ser alcançada em todas as etapas da comunicação, no entanto a privacidade dos usuários não é foco do projeto, já que os dados são acessíveis pela plataforma de nuvem.

Xively [86] é uma plataforma de nuvem proprietária que gerencia dados gerados por dispositivos IoT. Usuários associam seus dispositivos à plataforma e então podem utilizar diversos serviços disponíveis para consumir os dados enviados. Uma API também é disponibilizada para a construção de serviços específicos. Os dispositivos IoT utilizam os protocolos MQTT [87] e TLS [32] para uma conexão segura com os servidores do Xively, bibliotecas em diversas linguagens de programação estão disponíveis para a implementação do MQTT, mas apenas em C e Python para o TLS. No entanto, estes dois protocolos são amplamente utilizados, e bibliotecas de ambos costumam estar disponíveis para plataformas IoT, com implementações otimizadas. Os dados das redes IoT são ar-

mazenados em uma base de dados de séries temporais (do inglês, *time series database*), um tipo de banco de dados otimizado para lidar com dados numéricos ligados a um determinado tempo. No entanto, usuários podem enviar seus dados para bancos de dados externos, aumentando a privacidade da solução. A plataforma *Xively* fornece ferramentas avançadas para gerenciar os dispositivos IoT e os dados recebidos, entretanto os mecanismos de segurança utilizados não são otimizados para dispositivos severamente limitados, principalmente com fonte de energia limitada, o que limita a gama de aplicação de seus serviços. Em relação a privacidade, apesar de usuários poderem armazenar seus dados em bancos de dados externos, esta abordagem tem duas desvantagens: usuários precisam ter o conhecimento necessário para implantar e gerenciar uma solução de banco de dados, e, apesar dos dados serem armazenados externamente, eles ainda passam pelos servidores da *Xively*, o que significa que existe a possibilidade de a plataforma de nuvem ter acesso aos dados.

Os trabalhos acima mencionados oferecem plataformas de nuvem para gerenciar dispositivos IoT e seus dados, provendo meios para utilizar os dados coletados em serviços já disponíveis e APIs para construir serviços customizados. Esta abordagem vem sendo amplamente utilizada pela indústria, pois apresenta diversos benefícios, tais como a facilidade de gerenciar os dispositivos IoT de um único lugar, e uma maneira de consumir os dados coletados com serviços já disponíveis. Entretanto, como já mencionado, o armazenamento de dados privados aos usuários na nuvem cria um problema de privacidade, onde seus dados ficam expostos a diversas entidades (plataforma de nuvem e serviços). O projeto *SensorCloud* [88] aborda esta problemática por meio de uma plataforma de nuvem onde apenas serviços que consomem os dados possuem acesso aos mesmos. Uma arquitetura baseada em camadas é implementada, redes IoT (ou Redes de Sensores sem Fio) conectam-se com a nuvem por meio de Pontos de Confiança (do inglês, *Trust Points*), que são responsáveis por manter uma comunicação segura com a nuvem. Os Pontos de Confiança encriptam os dados gerados pelos sensores com uma chave secreta, e os enviam dessa forma para a plataforma de nuvem, que os armazena encriptados. Os serviços de nuvem a serem utilizados recebem a chave secreta, dessa forma apenas entidades autorizadas possuem acesso aos dados, provendo maior privacidade ao usuário. A comunicação entre os dispositivos das redes IoT e o Ponto de Confiança não é abordada pelo *SensorCloud*, apenas a comunicação entre o Ponto de Confiança e a Plataforma e Serviços de nuvem. O *SensorCloud* aplica-se a uma arquitetura onde usuários possuem redes IoT que se comunicam com um Ponto de Confiança, o qual possui mais recursos computacionais e também está sob controle do usuário. A utilização de serviços de nuvem ocorre por meio de uma vinculação entre o Ponto de Confiança e a plataforma de nuvem, a qual armazena os dados dos dispositivos e fornece uma loja onde os usuários podem escolher serviços

que consumirão seus dados. Ao enviar os dados para a nuvem, o Ponto de Confiança encripta os dados com uma chave simétrica, que por sua vez é encriptada pelas chaves públicas de todos os serviços autorizados a acessar o respectivo dado, por fim, as chaves encriptadas são armazenadas em um repositório de chaves na plataforma de nuvem. Esta abordagem impossibilita o acesso aos dados pela plataforma de nuvem, possibilitando um nível alto de privacidade, contudo é necessária a utilização de outro dispositivo (o Ponto de Confiança), aumentando o custo dos usuários.

O *User-driven Privacy Enforcement for Cloud-based Services in the IoT* (UPECSI) [13] estende o SensorCloud, implementando uma solução voltada a privacidade que abrange desde o processo de desenvolvimento de um serviço em nuvem até o usuário. Uma Linguagem de Desenvolvimento de Privacidade (do inglês, *Privacy Development Language* (PDL)) foi desenvolvida para facilitar o desenvolvimento de serviços de nuvem com privacidade. A utilização da PDL permite ao desenvolvedor fornecer informações detalhadas sobre quais dados e como eles são utilizados pela aplicação. Esta funcionalidade é então utilizada pelo usuário para habilitar (ou não) certas funções do serviço de acordo com suas preferências. A segurança entre as redes IoT do usuário e a nuvem é realizada por um Ponto de Aplicação de Privacidade (do inglês, *Privacy Enforcement Points* (PEP)), entidade análoga ao Ponto de Confiança no *SensorCloud*, mas com algumas outras responsabilidades. O PEP é um dispositivo na borda da rede de sensores que possui capacidade computacional e fonte de energia ilimitada, dessa forma é capaz de executar as tarefas necessárias para garantir a segurança e privacidade dos dados do usuário. Esta arquitetura permite ao usuário alterar a Política de Privacidade de acordo com suas preferências, fornecendo um alto nível de controle sob os dados armazenados na nuvem.

Não apenas o acesso aos dados é requisito de privacidade, a maneira como o provedor e os serviços de nuvem manuseia os dados do usuário também devem ser levados em conta. Por exemplo, deve ser possível estipular um tempo de vida para o dado armazenado, ou até mesmo a localização de armazenamento do dado [89]. A arquitetura UPECSI utiliza Anotações de Manuseio de Dados (do inglês, *Data Handling Annotations*) [90] para aplicar os requisitos de privacidade, tais anotações são enviadas com os dados para o provedor de nuvem, indicando como eles devem ser manuseados. Como mencionado anteriormente, a PDL também gera dados que possibilitam auditoria e monitoramento dos serviços de nuvem.

3.5 Considerações Finais

Este capítulo apresentou o estado da arte em Internet das Coisas, Computação em Nuvem e *Cloud of Things*. Como pode ser notado, estes são tópicos de grande interesse da

comunidade acadêmica, com muitos trabalhos abordando os desafios de cada área. Através dos trabalhos relacionados apresentados é possível notar que a integração entre a Internet das Coisas e a Computação em Nuvem ainda é recente, com muitas áreas em aberto, especialmente a privacidade. A arquitetura UPECSI, apesar de fornecer uma abordagem abrangente, se baseia na utilização de uma unidade central como responsável pelos dados da rede IoT. Esta abordagem apresenta algumas limitações, por exemplo, a introdução de um único ponto de falha na rede. No capítulo seguinte será proposta uma adaptação dos mecanismos de comunicação do UPECSI, a fim de remover este ponto único de falha e aumentar a segurança da arquitetura.

Capítulo 4

Arquitetura para Privacidade em Cloud of Things

Este capítulo apresenta a arquitetura desenvolvida durante este trabalho, primeiramente, na introdução, o contexto da proposta é apresentado, em seguida uma descrição geral é realizada, explicando como a arquitetura foi desenvolvida. Após a descrição geral ocorre o detalhamento da arquitetura, exibindo os protocolos desenvolvidos. Por fim, uma discussão cobrindo vários aspectos da arquitetura proposta é apresentada.

4.1 Introdução

Este trabalho modifica a arquitetura UPECSI [13] introduzindo mecanismos e protocolos necessários para possibilitar a comunicação direta entre os dispositivos das redes IoT com a nuvem. Para que isso seja possível, as tarefas relacionadas a segurança e privacidade, de responsabilidade do Ponto de Aplicação de Privacidade, são transferidas aos dispositivos. A transferência de tais tarefas é possibilitada pela utilização de uma pilha de rede nos dispositivos formada por protocolos da Internet, portanto este trabalho também propõe a remoção do PEP. Contudo, um dispositivo de borda ainda pode ser utilizado para tradução de protocolos ou outras tarefas que não necessitam de privilégios sob os dados.

A abordagem proposta acarreta pelo menos as seguintes vantagens: *(1)* melhora a tolerância a falhas da rede pois remove o dispositivo de borda, o qual propicia um ponto único de falha, da arquitetura; *(2)* melhora a segurança da rede pois remove um componente responsável por todas as tarefas relacionadas à segurança e que, consequentemente, pode quebrar as propriedades de segurança do sistema uma vez que seja comprometido por um ataque bem sucedido; e *(3)* a execução da aplicação de mecanismos de segurança e privacidade nos dispositivos possibilita um controle fino sob os dados, pois cada dispositivo pode adotar uma política de segurança diferente.

4.2 Descrição Geral da Arquitetura

Este trabalho propõe uma arquitetura voltada a privacidade para a integração entre a Internet das Coisas e a Computação em Nuvem. Os protocolos de comunicação propostos tem por objetivo possibilitar a comunicação direta e segura entre os dispositivos da Internet das Coisas com a nuvem. Uma análise discursiva é realizada a respeito das características dos protocolos propostos, envolvendo sua eficiência, segurança e viabilidade tendo em consideração as características limitadas dos dispositivos IoT.

Este trabalho foi realizado em duas etapas, primeiramente, foi proposta uma arquitetura inicial, chamada de *PROTeCt* (do inglês, ***P**rivacy a**R**quitecture for integrati**O**n of internet of **T**hings and **C**loud computing*), em seguida foram propostas melhorias para esta arquitetura, e o trabalho resultante foi chamado de E-PROTeCt (do inglês, *Enhanced PROTeCt*).

A validação da arquitetura proposta ocorre de 2 formas: *(1)* uma análise analítica é realizada, derivando formulas referentes a quantidade de dados encriptados e dados enviados; e *(2)* uma análise experimental, onde simulações de diversos cenários são realizadas e os resultados comparados com a arquitetura UPECSI e uma abordagem sem segurança.

4.3 PROTeCt

A arquitetura UPECSI envolve vários aspectos de um sistema, na integração entre a Internet das Coisas e a Computação em Nuvem, desde o desenvolvimento de serviços da nuvem até os protocolos de comunicação entre a borda das redes IoT e o provedor de nuvem. No entanto, não são mencionados os protocolos referentes a comunicação interna das redes IoT. O PROTeCt tem por objetivo modificar o UPECSI para abranger todos os dispositivos da arquitetura, introduzindo os mecanismos necessários para implementar uma comunicação segura entre os dispositivos IoT e a plataforma de nuvem.

Como já mencionado na introdução, o PROTeCt vantagens em comparação ao UPECSI, referentes ao aprimoramento da tolerância a falhas do sistema, melhora na segurança geral da rede e possibilidade de um controle fino dos dispositivos IoT. A abordagem do PROTeCt também traz desvantagens, principalmente relacionada à carga introduzida nos dispositivos, que agora são responsáveis por executar tarefas adicionais, um dos objetivos deste trabalho é quantificar esta carga. Como mencionado anteriormente, os *gateways* ainda podem existir, mas no PROTeCt eles não aplicam medidas de segurança nos dados.

A Figura 4.1 apresenta uma visão geral do PROTeCt, onde um usuário pode ter diversas redes IoT sob seu controle. De forma geral, o usuário vincula os dispositivos de suas redes ao provedor de nuvem, para que os dados gerados por eles seja armazenado

na nuvem. Os serviços de nuvem com autorização podem então processar os dados e apresenta-los ao usuário. Uma Terceira Parte Confiável (TPC) é responsável por auditar e monitorar os serviços de nuvem, prover políticas de privacidade padrão (para usuários iniciantes) e participar de alguns mecanismos de segurança e privacidade envolvendo a comunicação dos dispositivos com a nuvem.

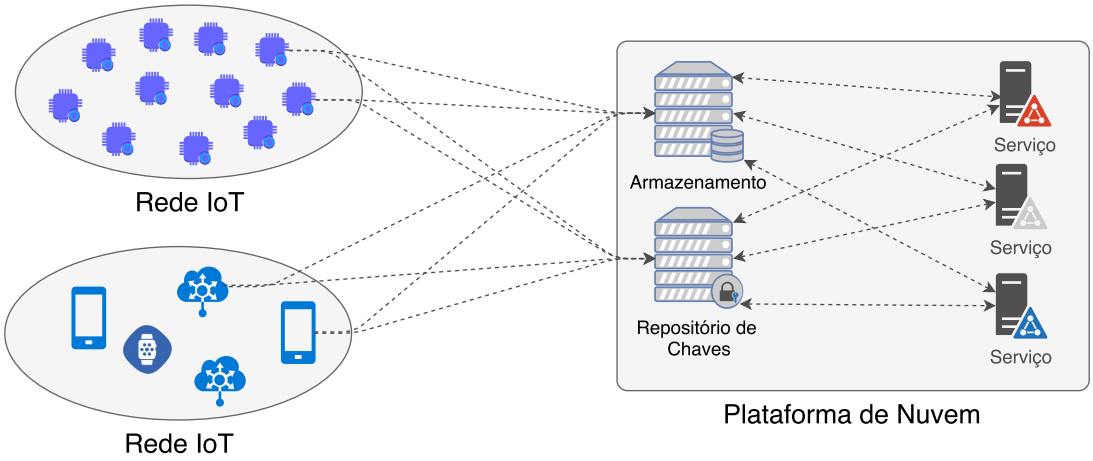


Figura 4.1: Arquitetura para Cloud of Things

Os dispositivos IoT armazenam os dados por eles gerados na nuvem de acordo com a Política de Privacidade. A aplicação da política possibilita que dados: *(1)* não sejam enviados para a nuvem, *(2)* sejam enviados cifrados, ou até mesmo *(3)* sejam enviados sem criptografia. Após o envio, os requisitos de privacidade são de responsabilidade dos serviços de nuvem, pois os mesmos possuem as chaves necessárias para acessar os dados.

O restante deste capítulo descreve os mecanismos de comunicação definidos pela arquitetura proposta. Primeiramente são descritos o processo de vinculação entre os dispositivos IoT e o provedor de nuvem (Seção 4.3.1) e a aplicação da política de privacidade definida pelo usuário (Seção 4.3.2). Em seguida são apresentados os mecanismos utilizados para assegurar a aplicação da política de privacidade tanto pelos dispositivos quanto pela nuvem (Seção 4.3.3). Por fim, políticas de privacidade flexíveis são apresentadas (Seção 4.3.4), as quais pode ser utilizadas para alterar as configurações de privacidade de forma dinâmica, de acordo com parâmetros pré definidos.

4.3.1 Vinculação dos Dispositivos IoT ao Provedor de Nuvem

Assim como geralmente ocorre em provedores de nuvem privados, primeiramente o usuário precisa cadastrar uma conta para então ter acesso aos serviços. Após o processo de registro os usuários podem vincular seus dispositivos para então armazenar os dados gerados. O processo de vinculação é realizado com a utilização do protocolo OAuth 2.0 [91], um

protocolo de código aberto para autorização segura. Após o processo de vinculação, os dispositivos podem enviar os dados gerados para a nuvem sem possuir as credenciais do usuário.

O protocolo OAuth 2.0 exige que a comunicação entre as partes seja realizada com um canal seguro para que o mesmo proporcione autorização segura. Na Internet, o protocolo responsável por prover um canal seguro é o TLS, o qual é utilizado em conjunto com o TCP. O protocolo TCP proporciona diversas funcionalidades ao canal, como controle de fluxo e ordenação das mensagens, mas a implementação destas funcionalidades adicionam uma alta carga à comunicação, tornando a utilização deste protocolo inviável para dispositivos severamente limitados. Atualmente esta área está sendo ativamente estudada pela comunidade acadêmica e diversas propostas de procolos de transporte seguros já foram realizadas. Particularmente relevante, uma adaptação do DTLS [73] está sendo estudada pelo grupo de trabalho DICE. O intuito do PROTeCt é a utilização de protocolos padronizados, para que seja possível a comunicação direta dos dispositivos IoT com a Internet, portanto este trabalho não define um protocolo específico para implementar a segurança da camada de transporte.

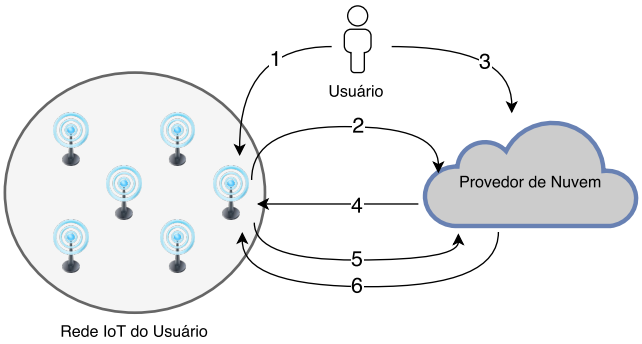


Figura 4.2: Processo de vinculação.

O usuário pode vincular um único dispositivo ou múltiplos dispositivos por vez. O processo de vinculação de múltiplos dispositivos pode ser realizado apenas com a utilização de um dispositivo de borda, dessa forma todos os dispositivos IoT compartilham a mesma identificação e o usuário não será capaz de aplicar uma política de privacidade diferente a cada dispositivo. Na nossa proposta, é realizada a vinculação de um dispositivo por vez, utilizando o processo definido como Concessão de Código de Autorização (do inglês *Authorization Code Grant*) do protocolo OAuth, este tipo de permissão necessita da intervenção do usuário apenas na fase de configuração. Após receber a autorização, os dispositivos podem enviar dados para a nuvem sem utilizar as credenciais do usuário.

A Figura 4.2 apresenta o processo de vinculação para um dispositivo, a qual possui os seguintes passos:

1. O usuário inicia o processo como o *Resource Owner* acessando a página web de vinculação do dispositivo;
2. O dispositivo solicita o Código de Autorização para a nuvem;
3. O usuário é então redirecionado a uma página para inserir suas credenciais;
4. Após receber as credenciais do usuário a plataforma de nuvem envia o Código de Autorização ao dispositivo;
5. De posse do Código de Autorização, o dispositivo solicita um Token de Acesso;
6. A plataforma de nuvem envia o Token de Acesso.

Ao finalizar o processo o dispositivo possui um Token de Acesso que precisa ser anexado a todas as mensagens enviadas à nuvem. O provedor de nuvem autoriza o recebimento e armazenamento dos dados de acordo com o Token de Acesso. O processo de vinculação para múltiplos dispositivos é o mesmo, mas é executado pelo dispositivo de borda, que, ao receber o Token de Acesso, o envia a todos os dispositivos da rede. Como todos os dispositivos possuem o mesmo token, não há distinção de dispositivos pelo provedor de nuvem. Após vincular todos os dispositivos o usuário também pode criar redes lógicas para facilitar o gerenciamento.

4.3.2 Aplicação da Política de Privacidade

Uma Política de Privacidade (PP) define se um dado pode ser armazenado na nuvem, como ele será armazenado (encriptado, texto puro, ou até mesmo não armazenado) e o que pode ser feito com este dado por um serviço de nuvem. Diferentemente de políticas de privacidade comuns, onde o usuário tem apenas a opção de aceitar ou rejeitar ela por completo, no PROTeCt o usuário tem o poder de alterar a sua política de privacidade com o tempo. Esta abordagem permite ao usuário assumir controle real sob os dados que envia para a nuvem, pois tem conhecimento e controle sob as operações realizadas com seus dados pelos serviços autorizados. Serviços de nuvem fornecem ao usuário uma interface onde é possível analisar quais são os dados utilizados pelo serviço, e para quais propósitos, os usuários então podem habilitar ou desabilitar funções específicas do serviço de modo a restringir o acesso a determinados dados de seus dispositivos. Por exemplo, um usuário pode desabilitar a utilização de dados de localização por um determinado serviço se assim desejar.

Assim como no processo de vinculação, no UPECSI este procedimento é realizado pelo dispositivo de borda, e os usuários atualizam as políticas de privacidade por este dispositivo. No PROTeCt uma Terceira Parte Confiável é utilizada para auxiliar no

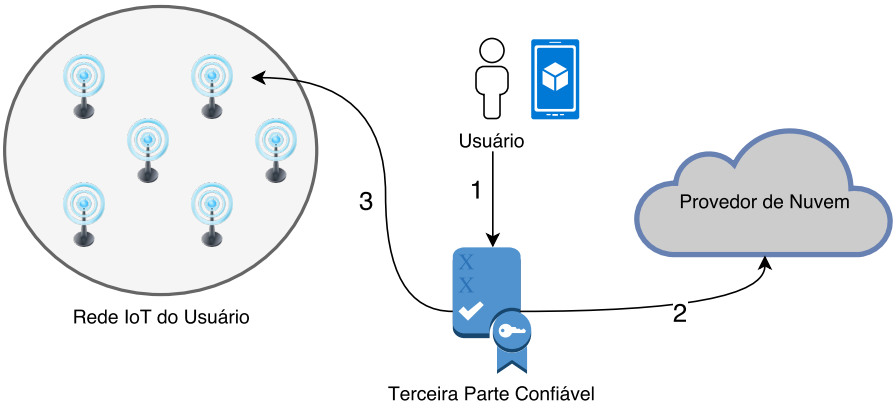


Figura 4.3: Atualização da Política de Privacidade.

processo de atualização das políticas de privacidade (por exemplo, auditar a PP), o usuário pode atualizar uma única PP e enviar a todos os dispositivos de suas redes de uma vez.

A Política de Privacidade é aplicada toda vez que um dispositivo IoT envia dados para a nuvem, tal aplicação é realizada pelo próprio dispositivo. Quando o usuário autoriza um novo serviço de nuvem, a política de privacidade associada a este serviço é enviada para a TPC e em seguida a todos os dispositivos do usuário. A Figura 4.3 apresenta o processo de atualização de uma PP:

1. O usuário acessa a Política de Privacidade na Terceira Parte Confiável por meio de uma interface (navegador, smartphone, etc) e realiza as configurações desejadas;
2. A TPC envia a PP atualizada para o serviço de nuvem respectivo;
3. A TPC gera uma Configuração de Privacidade (CP) e envia aos dispositivos do usuário;

4.3.3 Armazenamento Privado

Após vincular os dispositivos ao provedor de nuvem e configurar as Políticas de Privacidade referentes aos serviços utilizados, é necessário garantir que os dados vindos dos dispositivos IoT são armazenados e acessados apenas por entidades autorizadas. Este objetivo é atingido por um mecanismo de armazenamento privativo baseado em criptografia simétrica. Assim como no processo de vinculação, a comunicação entre os dispositivos IoT e a nuvem deve ser segura, o que é garantido por um protocolo da camada de transporte. Os dados sensíveis são armazenados cifrados na nuvem (dados não sensíveis podem ser armazenados sem criptografia), dessa forma é prevenido o acesso ao dado até mesmo pelo provedor de nuvem. As chaves utilizadas para decriptar os dados são armazenadas em um repositório de chaves também na nuvem (como projetado por [92]).

Antes do envio de uma mensagem, o dispositivo IoT deve filtra-la de acordo com a Configuração de Privacidade, dessa forma decidindo se o dado deve deixar a esfera de controle do usuário. O dado é então cifrado com um algoritmo de criptografia simétrica. A chave utilizada por tal algoritmo é então cifrada com a chave pública dos serviços autorizado a acessar o respectivo dado. Caso haja mais de um serviço utilizando o mesmo dado, a chave simétrica deve ser cifrada uma vez para cada serviço. O dado é armazenado apenas uma vez no provedor, independente do número de serviços que o acessarão.

A chave simétrica é atualizada periodicamente, prevenindo que novos serviços acessem dados armazenados previamente à sua autorização, ou que serviços com autorização cancelada continuem a acessar os dados do usuário. As chaves antigas continuam no repositório de chaves, possibilitando o acesso a dados já armazenados por novos serviços.

Como pode ser visto, os dispositivos IoT executam diversas operações criptográficas. No caso de dispositivos limitados, com fonte de energia restrita, operações criptográficas podem diminuir significativamente o tempo de vida da rede. Duas abordagens são propostas para mitigar este problema, as chaves criptográficas são gerenciadas pelos os dispositivos IoT (Seção 4.3.3) ou pela Terceira Parte Confiável (Seção 4.3.3). A seguir tais abordagens são apresentadas e suas vantagens e desvantagens discutidas.

Gerenciamento das Chaves pelos Dispositivos IoT

A Figura 4.4 apresenta a abordagem onde o próprio dispositivo é responsável por criar as chaves simétricas, cifrá-las com as chaves públicas dos serviços e enviá-las para o repositório de chaves no provedor de nuvem.

Esta abordagem é proposta para dispositivos que possuam capacidade computacional e energética suficientes para executar as seguintes tarefas (Figura 4.4) sem impactar significativamente seu tempo de vida:

1. Receber as chaves públicas dos serviços pelo provedor de nuvem;
2. Criar periodicamente chaves simétricas;
3. Cifrar as chaves simétricas com as chaves públicas dos serviços autorizados;
4. Enviar as chaves simétricas cifradas ao repositório de chaves;
5. Enviar dados cifrados ao provedor de nuvem.

Gerenciamento das Chaves por uma Terceira Parte Confiável

A Figura 4.5 apresenta a segunda abordagem, onde uma Terceira Parte Confiável é responsável pela maioria das tarefas relacionadas ao controle de acesso dos dados, atenuando

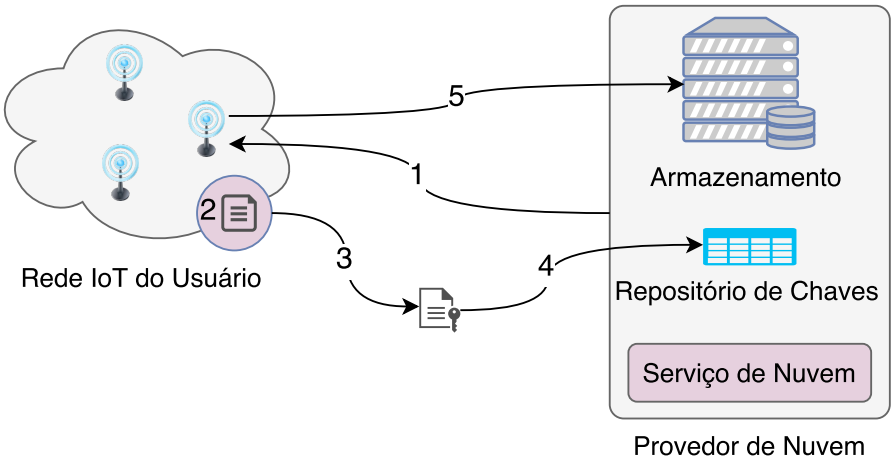


Figura 4.4: Gerenciamento das Chaves pelos Dispositivos IoT.

a carga nos dispositivos IoT. A TPC cria as chaves simétricas, recebe as chaves públicas dos serviços, cifra as chaves simétricas e às envia ao repositório de chaves. As chaves simétricas também são enviadas aos dispositivos.

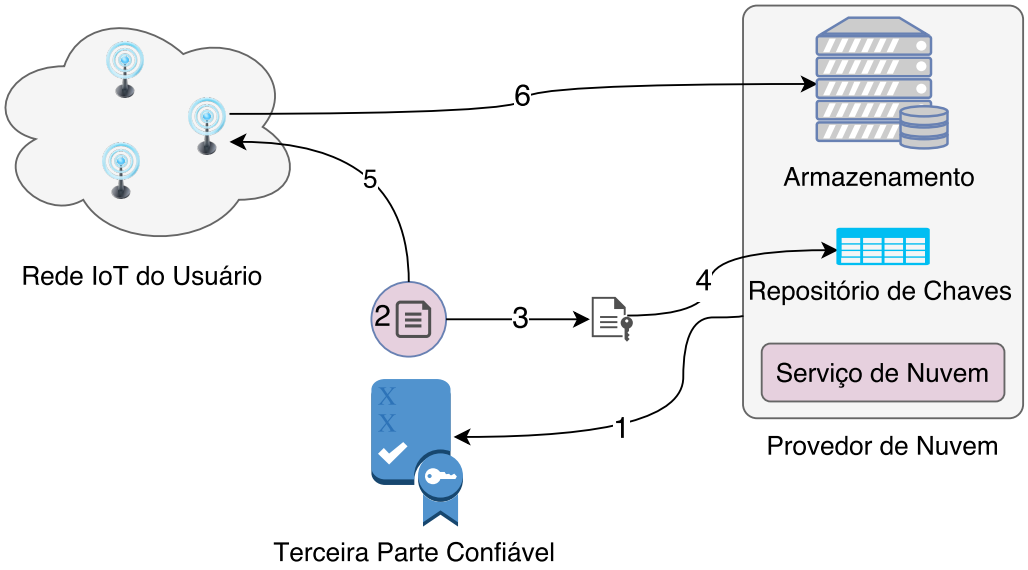


Figura 4.5: Gerenciamento das Chaves por uma TPC.

Esta abordagem é vantajosa para dispositivos limitados, onde as tarefas migradas à TPC diminuiriam o atraso na comunicação e aumentariam a expectativa de vida do aparelho. Nesta abordagem são executados os seguintes passos:

1. **TPC**: Recebe as chaves públicas dos serviços pelo provedor de nuvem;
2. **TPC**: Cria periodicamente chaves simétricas;
3. **TPC**: Cifra as chaves simétricas com as chaves públicas dos serviços autorizados;

4. **TPC**: Envia as chaves simétricas cifradas ao repositório de chaves;
5. **TPC/Dispositivo IoT**: Envia (de forma segura) as chaves simétricas aos dispositivos IoT;
6. **Dispositivo IoT**: Envia dados cifrados ao provedor de nuvem.

4.3.4 Políticas de Privacidade Flexíveis

Os mecanismos de controle de acesso aos dados propostos neste trabalho possibilitam que apenas entidades previamente autorizadas tenham acesso aos dados do usuário na nuvem. Entretanto, em situações excepcionais, pode ser conveniente diminuir os requisitos de privacidade para possibilitar o acesso aos dados por novas entidades. Por exemplo, quando uma rede de monitoramento de saúde detecta uma emergência, pode ser mais vantajoso ao usuário que vários médicos hospitais acesso aos seus dados, maximizando a chance de um atendimento rápido.

Este trabalho propõe a utilização de Políticas de Privacidade Flexíveis (PPF) a fim de possibilitar o cenário descrito acima. PPFs são Políticas de Privacidade secundárias, ativadas quando determinado evento é detectado. O usuário precisa primeiramente configurar os parâmetros que ativarão a PPF, que podem ser relacionados aos dados gerados pelas suas redes IoT ou até mesmo por eventos externos. PPFs são criadas da mesma maneira que Políticas de Privacidade comuns (Seção 4.3.2).

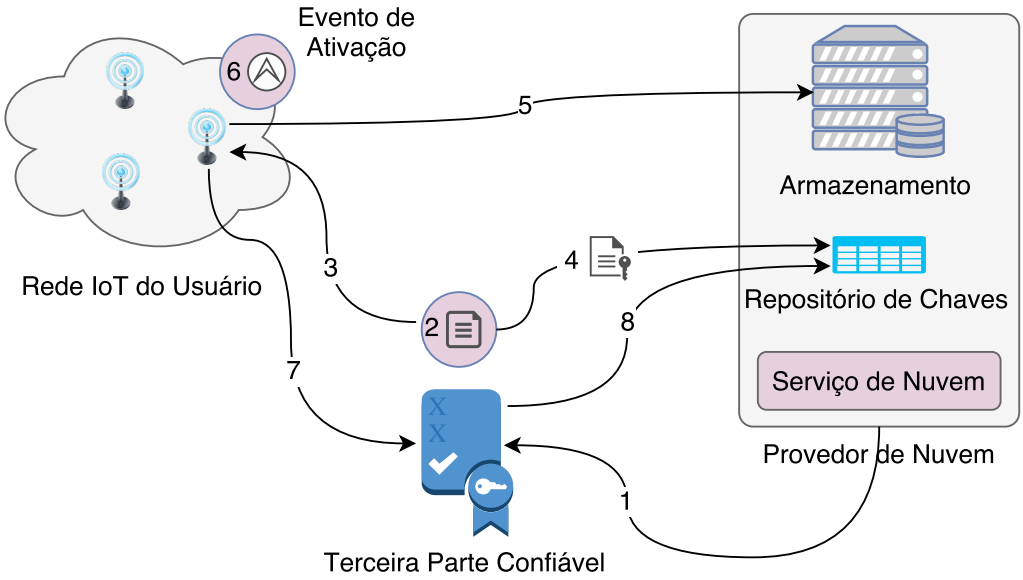


Figura 4.6: Ativação de uma Política de Privacidade Flexível.

A Figura 4.6 apresenta o processo de ativação de uma PPF. O processo é similar ao controle de acesso de dados gerenciado pela TPC, mas neste caso as chaves públicas

utilizadas para cifrar as chaves simétricas são geradas pela própria TPC e os serviços de nuvem podem acessar tais dados apenas quando a PPF é ativada. O processo de ativação de uma PPF possui os seguintes passos:

1. A TPC recebe as chaves públicas dos serviços pelo provedor de nuvem;
2. A TPC cria periodicamente as chaves criptográficas:
   (a) Chaves simétricas, que são utilizadas pelos dispositivos para encriptar os dados (as mesmas utilizadas para as Políticas de Privacidade comuns);
   (b) Um par de chaves pública/privada para cada serviço que receberá acesso quando a PPF for ativada, estes pares ficam de posse da TPC;
3. A TPC envia de forma segura as chaves simétricas para os dispositivos IoT;
4. A TPC envia de forma segura as chaves simétricas cifradas com as chaves públicas para o repositório de chaves;
5. O dispositivo IoT envia dados cifrados com a chave simétrica para a nuvem;
6. O dispositivo IoT detecta um evento que ativa a PPF;
7. O dispositivo IoT alerta a TPC;
8. A TPC envia, por meio do repositório de chaves, a chave privada (gerada no passo 2.2) para o serviço de nuvem que receberá acesso aos dados, esta chave privada é cifrada com a chave pública do serviço (recebida no passo 1).

Após a desativação de uma PPF, o par de chaves assimétricas referentes ao serviço que teve autorização temporária deve ser recriado, e as novas chaves simétricas geradas a partir deste momento devem ser cifradas com o novo par, revogando o acesso temporário deste serviço. O processo de Políticas de Privacidade Flexíveis é altamente dependente da TPC, pois os dispositivos IoT são intrinsecamente não confiáveis, e muitas vezes o evento que ativará a PPF pode ser referente a falha dos dispositivos.

4.4 Enhanced PROTeCt

O *Enhanced PROTeCt* (E-PROTeCt) aumenta o desempenho do PROTeCt por diminuir os custos associados ao processamento de operações criptográficas e de transmissão de dados dos dispositivos IoT. O E-PROTeCt tem foco apenas na etapa de transmissão de dados, já que esta etapa ocorre com maior frequência e tem maior impacto no desempenho e tempo de vida da rede, diferentemente das etapas de vinculação e atualização das

políticas de privacidade, que ocorrem de forma eventual. Como descrito anteriormente, o PROTeCt exige a utilização de um protocolo de segurança na camada de transporte para realizar o envio dos dados de forma segura, e como ocorre a criptografia dos dados na camada de aplicação (pelo processo de armazenamento privado - Seção 4.3.3), os dados são encriptados mais uma vez na camada de transporte.

O processo de transmissão de dados exige um canal seguro entre os dispositivos IoT e a plataforma de nuvem para que ações mal intencionadas sejam evitadas (por exemplo, prevenir que entidades mal intencionadas personifiquem um dispositivo do usuário utilizando o *Access Token* e enviar dados falsos para a plataforma de nuvem). O E-PROTeCt propõe um mecanismo de segurança implementado na camada de aplicação para proteger a comunicação entre dispositivos IoT e a plataforma de nuvem. O E-PROTeCt assume que existe uma chave secreta compartilhada entre cada dispositivo e a plataforma de nuvem, este compartilhamento pode ocorrer durante a fase de vinculação, quando um protocolo de canal seguro é utilizado.

No PROTeCt, ao transmitir um determinado dado, o dispositivo IoT deve anexar o *Access Token*, o qual credencia o dispositivo em nome do usuário, à mensagem. A Figura 4.7 exibe as operações de criptografia e formato da mensagem de uma transmissão utilizando um protocolo de segurança na camada de transporte, neste caso o DTLS é utilizado como referência, o campo MAC refere-se ao Código de Autenticação da Mensagem (do inglês, *Message Authentication Code* (MAC)) [71]. Os dados gerados pelos dispositivos IoT são encriptados duas vezes: a *Criptografia 1* é realizada na camada de aplicação pelo mecanismo de armazenamento privado; a *Criptografia 2* é realizada pelo protocolo de segurança da camada de transporte. Note que os dados gerado pelos dispositivos IoT são encriptados primeiro na camada de aplicação e em seguida novamente pelo protocolo DTLS, junto com o MAC e o Access Token.

O E-PROTeCt reduz a quantidade de dados que precisam ser encriptados para que a transmissão seja realizada de forma segura. Consequentemente, a quantidade de dados que precisam ser transmitidos também diminui. A criptografia realizada na camada de aplicação sempre será necessária, pois é assim que os dados são armazenados na plataforma de nuvem, de forma que apenas serviços de nuvem autorizados acessem os dados. Portanto, o método proposto pelo E-PROTeCt é aplicado na camada de aplicação, removendo a necessidade de utilização de um protocolo de segurança na camada de transporte. Para proteger a mensagem por completo é necessário apenas associar o *Access Token* com os dados já encriptados, prevenindo a personificação pelo simples fato de observar a mensagem trafegando na rede e enviando dados falsos com o *Access Token* capturado. O E-PROTeCt propõe a transmissão de um *hash* dos dados encriptados, o qual é encriptado pela chave secreta (simétrica) compartilhada entre o dispositivo e a plataforma de nuvem.

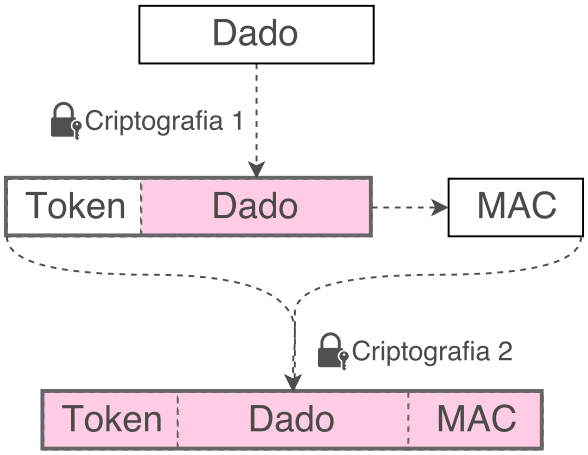


Figura 4.7: Operações criptográficas realizadas quando um protocolo de segurança da camada de transporte é utilizado. Caixas cinzas e brancas representam dados encriptados e puros, respectivamente.

Nesta abordagem, mesmo de posse do *Access Token*, um atacante não é capaz de fabricar uma mensagem falsa, pois não possui a chave secreta para encriptar o *hash* da mensagem.

A Figura 4.8 apresenta as operações criptográficas e o formatado da mensagem propostos pelo E-PROTeCt, onde os dados a serem armazenados na nuvem são encriptados apenas uma vez (*Criptografia 1*), e então a chave simétrica compartilhada é utilizada para encriptar o *hash* da mensagem já encriptada (*Criptografia 2*). Por fim, o *Access Token* é anexado à mensagem e a mesma é transmitida. A plataforma de nuvem, ao receber a mensagem, verifica sua autenticidade realizando os seguintes passos: *(1)* identifica a chave secreta referente ao *Access Token* recebido; *(2)* decripta o *hash* recebido; e *(3)*, compara o *hash* obtido na etapa 2 com o *hash* realizado na parte referente aos dados recebidos. É importante ressaltar que o *hash* é obtido da mensagem encriptada e a plataforma de nuvem continua sem ter acesso aos dados do usuário.

4.5 Discussões

Esta seção apresenta uma ampla discussão a respeito de aspectos relacionados com a arquitetura apresentada na seção anterior. Primeiramente são discutidos aspectos relacionados à segurança da comunicação sem-fio para dispositivos limitados, e como eles se comparam a tecnologias utilizadas em dispositivos com maior poder de processamento, como o PEP. Em seguia são discutidos os mecanismos propostos, apresentando suas limitações e benefícios.

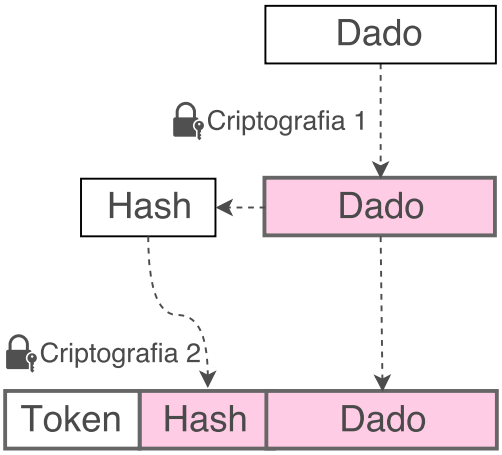


Figura 4.8: Operações criptográficas necessárias quando utilizando uma chave secreta compartilhada com a plataforma de nuvem. Caixas cinzas e brancas representam dados encriptados e puros, respectivamente.

4.5.1 Comunicação Direta dos Dispositivos com a Internet

Um dos objetivos deste trabalho é possibilitar a comunicação direta entre dispositivos IoT severamente limitados e a Internet, e para tanto é necessário que os dispositivos utilizem protocolos compatíveis com os já utilizados hoje na Internet. A arquitetura em que este trabalho é baseado (UPECSI [13]) não define os protocolos de comunicação utilizados entre os dispositivos IoT e o dispositivo de borda, o E-PROTeCt define estes protocolos e transfere as responsabilidades do PEP para os dispositivos IoT.

Apesar deste trabalho definir protocolos para a camada de aplicação, uma discussão a respeito de protocolos de segurança para a camada de transporte projetos para dispositivos limitados é pertinente, já que propomos não utiliza-los. A segurança nas camadas mais baixas (Camada de Rede e Camada de Enlace) de dispositivos limitados apresenta diversos desafios, mas existem protocolos bastante maduros e consolidados: o IEEE 802.15.4 para a camada de enlace, o 6LoWPAN para endereçamento com IPv6 e o RPL [93] para roteamento na camada de rede são exemplos.

A implementação de mecanismos de segurança para dispositivos IoT severamente limitados possui diversos desafios. A carga adicional necessária para realizar operações criptográficas e a sobrecarga relacionada a quantidade e tamanho das mensagens necessárias para prover a segurança pode aumentar a latência e diminuir o tempo de vida dos dispositivos (no caso de não haver fonte de energia ilimitada). Protocolos de segurança da Internet, como o TLS, utilizam Infraestrutura de Chave Pública (do inglês, *Public Key Infrastructure* (PKI)) [94] para trocar chaves simétricas de forma secreta, e com estas chaves simétricas as mensagens são protegidas, muito parecido com a bordagem utilizada neste trabalho. A primeira etapa, relacionada à troca de chaves, é especialmente custosa,

tanto em termos de bytes trocados como em termos de carga de processamento, por isso muitos trabalhos assumem que as chaves foram previamente carregadas nos dispositivos.

O TLS pode utilizar diversos algoritmos para a troca de chave, confidencialidade e autenticidade. O algoritmo para troca de chaves mais utilizado é o RSA [95], o qual acarreta uma grande sobrecarga para dispositivos limitados. Portanto estudos atuais sugerem a utilização de criptografia de curva-elíptica, como o *Elliptic-curve Diffie–Hellman* (ECDH) [96], que consume menos recursos que o RSA para processadores de 8 bits [97]. Plataformas IoT muitas vezes incluem um *hardware* específico para efetuar operações criptográficas [98], dessa forma o desempenho dessas operações aumenta consideravelmente. Apesar de haver diversas propostas de protocolos de canal seguro para dispositivos limitados [99, 100, 73], o *Internet Engineering Task Force* (IETF) ainda está trabalhando em um padrão.

Já que a intenção deste trabalho é possibilitar a comunicação direta entre dispositivos IoT severamente limitados e a Internet, o que significa a utilização de protolos definidos pelas entidades padronizadoras, estre tabalho não implementa um novo protocolo de camada de transporte ou utiliza um já proposto por outros trabalhos. O PROTeCt assume a utilização de um canal seguro, e sua análise utiliza o DTLS como implementação, enquanto o E-PROTeCt implementa comunicação segura diretamente na camada de aplicação, reduzindo a desvantagem introduzida por estes mecanismos.

4.5.2 Arquitetura Proposta

Esta seção discute os protocolos propostos neste trabalho, primeiramente é realizada uma discussão a cerca do protocolo de vinculação, indicando seus benefícios e possíveis dificuldades de implementação em dispositivos limitados. Em seguida, são discutidos aspectos a respeito de como as Políticas de Privacidade são aplicadas pelos dispositivos IoT, em relação às suas características limitadas. O processo de armazenamento privado também é discutido, este é o processo executado toda vez que um dispositivo envia dados para a nuvem, portanto a sobrecarga introduzida pelo mesmo é analisada em detalhes. Por fim, as Políticas de Privacidade Flexíveis são discutidas.

Vinculação

O processo de vinculação ocorre apenas quando o usuário registra os seus dispositivos à plataforma de nuvem, ou seja, este processo possivelmente ocorrerá apenas uma vez para cada dispositivo. Portanto, a carga adicionada aos dispositivos pode ser negligenciada. Caso o usuário possua uma grande quantidade de dispositivos, este processo pode se tornar inconveniente e exaustivo, em situações como esta, a vinculação pode ocorrer por meio de

um dispositivo auxiliar, que personifica todos os outros dispositivos, e, ao obter o *Access Token* o redireciona a todos os dispositivos. Esse recurso agiliza a etapa de vinculação da rede, mas diminui a granularidade de controle do usuário, pois todos os dispositivos serão obrigatoriamente vinculados ao mesmo provedor.

O protolo OAuth 2.0 exige a utilização de um canal seguro entre o dispositivo e o provedor, pois não dispõe de recursos para manter a confidencialidade das mensagens trocadas. Caso o processo de vinculação seja realizado por cada dispositivo, é necessário que eles disponham de uma interface de acesso ao usuário. Levando em conta sua natureza limitada, é importante que a implementação dessa etapa seja realizada de forma otimizada.

Aplicação da Política de Privacidade

A aplicação da Política de Privacidade é realizada da mesma forma que no UPECSI, transferindo o processo do *gateway* ao dispositivo. Uma diferença é a utilização da Terceira Parte Confiável para auxiliar na alteração da PP, que fornece uma interface ao usuário. Esta abordagem é realizada para previnir que o serviço de nuvem possa alterar a PP sem consentimento do usuário. O processo de atualização da PP não ocasiona uma carga considerável aos dispositivos, já que não deve acontecer com frequência. O procedimento de aplicação da PP é realizado por cada dispositivo, o qual, de posse da PP, deve filtrar todas as mensagens a serem enviadas ao provedor. Como esta filtragem não exige grande processamento, sua implementação também não deve ocasionar aumento significativo da carga no dispositivo.

Armazenamento Privado

A privacidade do usuário é garantida por meio de criptografia. O dado é armazenado no provedor de forma encriptada, e apenas os serviços autorizados possuem a chave para decriptá-lo. Este mecanismo permite que apenas entidades previamente autorizadas tenham acesso ao dado. Como já discutido, operações criptográficas podem ser custosas para dispositivos IoT. Embora os dispositivos IoT precisem encriptar os dados, algumas operações podem ser transferidas para a Terceira Parte Confiável, aliviando a carga nos dispositivos. Nesta abordagem o gerenciamento das chaves é realizado pela TPC, e os dispositivos precisam apenas receber as chaves e cifrar os dados. Esta abordagem apresenta uma alternativa, no entanto a utilização da abordagem seguinte é recomendada sempre que possível, pois não insere um novo ponto único de falha no processo.

Os próprios dispositivos podem ser responsáveis por criar as chaves simétricas periodicamente e enviá-las aos serviços autorizados, sem o intermédio da Terceira Parte Confiável. Este método introduz maior carga aos dispositivos, que ficam responsáveis por tarefas de alto custo computacional: criação de chaves simétricas, encriptação das chaves

simétricas com as chaves públicas dos serviços e envio das chaves encriptadas aos serviços. Vale notar que a encriptação e o envio é realizada uma vez para cada serviço sendo utilizado, e esse processo repete-se periodicamente, deve-se ter cautela ao utilizar tal método, pois caso o período de renovação das chaves seja muito pequeno os dispositivos podem ter uma sobrecarrega. A vantagem desta abordagem é a remoção da Terceira Parte Confiável do processo, tornando o protocolo mais seguro pois diminui a superfície de ataque. Este método é indicado para redes IoT de moderada capacidade computacional e de grande disponibilidade de energia.

Quando utilizando o PROTeCt, em ambas as abordagens o dado será cifrado duas vezes, uma na camada de transporte, por um protocolo que garanta um canal seguro, e outra na camada de aplicação, para armazenar o dado de forma privada. O E-PROTeCt foi concebido para mitigar a carga introduzida por este processo, de forma muito parecida ao DTLS, mas evitando encriptar a parte dos dados duas vezes, ele reduz a quantidade de dados encriptados durante o envio. No entanto, o protocolo implementado no E-PROTeCt não realiza proteção contra inserção duplicada de dados. Caso um atacante tenha acesso a um pacote já enviado, o mesmo pode realizar um novo envio deste mesmo pacote, resultando em seu armazenamento duplicado. Este comportamento pode ser considerado uma troca entre segurança e desempenho.

Políticas de Privacidade Flexíveis

Políticas de Privacidade Flexíveis foram propostas para aumentar a flexibilidade da arquitetura. Este protocolo possibilita ao usuário definir limites, para que serviços de nuvem sejam habilitados ou desabilitados. Este mecanismo é especialmente útil em situações excepcionais, onde pode ser mais vantajoso abdicar da privacidade.

Uma vez que o principal propósito das Políticas de Privacidade Flexíveis são situações excepcionais, pode ser o caso em que o próprio dispositivo IoT deixe de funcionar, desta forma o acionamento da PPF é possível apenas por um agente externo. Para assegurar o correto funcionamento em uma situação como esta a TPC é responsável pela aplicação da PPF, gerenciando as chaves e autorizando os serviços quando necessário. Este método torna o protocolo mais confiável e introduz pouca carga aos dispositivos, pois apenas uma mensagem é enviada do dispositivo à TPC para ativação da PPF.

4.6 Considerações Finais

Este capítulo apresentou a arquitetura desenvolvida neste trabalho. A arquitetura para privacidade na integração entre a Internet das Coisas e a computação em nuvem foi desenvolvida e discutida, todas suas etapas foram apresentadas, e, onde pertinente, compara-

ções com o UPECSI foram realizadas. O próximo capítulo apresenta um análise analítica e experimental das soluções propostas.

Capítulo 5

Resultados

Este capítulo apresenta a avaliação da arquiteturas propostas. As arquitetura são avaliadas em duas etapas, primeiramente é realizada uma análise analítica, onde é estudada a quantidade de operações criptográficas e a quantidade de *bytes* que devem ser enviados dos dispositivos IoT para o *gateway*, e deste para a nuvem. Note que nesta análise o PROTeCt e o E-PROTeCt utilizam o *gateway* para encaminhamento de dados. Por fim, uma análise experimental é efetuada, nesta etapa uma simulação foi realizada, e resultados referentes à vazão, latência e consumo de energia foram obtidos e analisados.

Todas as análises são relativas apenas ao processo de Armazenamento Privado(Seção 4.3.3), pois esta é a etapa que ocorre com frequência, enquanto as outras ocorrem de forma esporádica. Em ambas as etapas comparações são realizadas entre os seguintes itens:

- **Sem Segurança**: dados são enviados sem qualquer proteção à nuvem;
- **UPECSI**: os dados são enviados sem segurança até o PEP, onde os dados são encriptados e enviados à nuvem (de acordo com [13]);
- **PROTeCt**: dados são enviados de acordo com a arquitetura proposta, utilizando o DTLS como protocolo de segurança para a camada de transporte;
- **E-PROTeCt**: dados são enviados de acordo com a arquitetura proposta, utilizando o mecanismo de proteção na camada de aplicação.

5.1 Análise Analítica

Estão seção apresenta uma análise analítica acerca dos custos, em termos de criptografia e quantidade de dados enviados, para prover privacidade na integração de IoT e computação em nuvem. A fim de prover segurança dentro da rede IoT, as abordagens PROTeCt e E-PROTeCt caracterizam-se por introduzir maior necessidade criptográfica e de envio

de dados pelos dispositivos de IoT e por isso estas métricas foram escolhidas para esta análise.

5.1.1 Operações Criptográficas Executadas pelos Dispositivos IoT e o *Gateway*

A Tabela 5.1 apresenta uma análise referente à quantidade de operações criptográficas realizadas em cada abordagem, tanto nos dispositivos IoT quanto no *gateway*. A seguinte nomenclatura é utilizada: $C(u)$ – custo para encriptar u *bytes*; $T(j)$ – tamanho resultante da criptografia de j *bytes*; x – tamanho dos dados gerados pelos dispositivos de IoT que devem ser enviados ao provedor; y – tamanho do *token* de acesso usado para identificar o dispositivo de IoT; z – tamanho do *hash* da cifra de x.

Tabela 5.1: Custo criptográfico nos dispositivos de IoT e no *gateway*, considerando dados de x *bytes*, um *token* de acesso de y *bytes* e um *hash* de z *bytes*.

	Criptografia nos dispositivos de IoT	Criptografia no *Gateway*
Sem Segurança	—	—
UPECSI	—	$C(x) + C(T(x) + y + z + 4)$
PROTeCt	$C(x) + C(T(x) + y + z + 4)$	—
E-PROTeCt	$C(x) + C(z)$	—

Obviamente, a abordagem Sem Segurança não executa operações criptográficas. A UPECSI não adiciona custo criptográfico nos dispositivos de IoT visto que esta comunicação não é segura, enquanto o *gateway* é o responsável pela criptografia dos dados ($C(x)$) e envio ao provedor por meio de um canal seguro ($C(T(x)+y+z+4)$). Por outro lado, o PROTeCt transfere este custo para os dispositivos de IoT, fornecendo segurança na rede interna e possibilitando um controle de acesso aos dados por dispositivo. Por fim, o E-PROTeCt primeiramente encripta os dados ($C(x)$) para então encriptar seu *hash* ($C(z)$). Existe ainda um *overhead* de 4 *bytes* na camada de aplicação, referente à utilização de um protocolo para representação dos dados, que não precisam ser encriptados quando não usamos um canal seguro.

Todas as abordagens exigem duas operações criptográficas, mas para diferentes quantidade de dados. Na UPECSI e PROTeCt, a quantidade de dados das duas operações depende do tamanho dos dados gerados pelos dispositivos de IoT (x), enquanto que na E-PROTeCt apenas uma das operações é afetada por este valor, pois $C(z)$ é sempre constante. Além disso, esta última abordagem acaba com a necessidade de se encriptar os dados duas vezes, reduzindo a quantidade de dados a serem encriptados.

5.1.2 Quantidade de Dados Transmitidos

Uma análise referente à quantidade de dados enviados também é realizada, pois em redes severamente limitadas mesmo a economia de poucos *bytes* pode ocasionar um aumento significativo no desempenho e tempo de vida dos dispositivos, já que normalmente a quantidade de dados enviados é muito pequena.

Neste caso, é importante ressaltar que o algoritmo de criptografia AES em modo CBC, comumente usado para criptografia simétrica, utiliza blocos de 128 *bits* (16 *bytes*). Por exemplo, a criptografia de 10 *bytes* resulta em um bloco de 16 *bytes*. A utilização de um algoritmo de *padding* se faz necessária para que o destino identifique o tamanho real da mensagem, e por isso um bloco na realidade pode conter apenas 15 *bytes* de dados reais, ou seja, a criptografia de 16 *bytes* resulta em 2 blocos, totalizando 32 *bytes*.

A Tabela 5.2 apresenta a sobrecarga de dados a serem enviados em cada abordagem. Em redes IoT muitas vezes a quantidade de dados enviados é pequena, e o *padding* pode impactar o desempenho da aplicação. Em relação aos dados enviados, as abordagens PROTeCt e E-PROTeCt apresentam maior quantidade de dados enviado em relação à UPECSI, este é o custo necessário para implementação de mecanismos de segurança diretamente no dispositivo de IoT.

Tabela 5.2: Quantidade de dados enviados nas comunicações, considerando dados de x *bytes*, um *token* de acesso de y *bytes* e um *hash* de z *bytes*.

	Dispositivos de IoT → *Gateway*	*Gateway* → Provedor de Nuvem
Sem Segurança	$x + 4$	$x + y + 4$
UPECSI	$x + 4$	$T(T(x) + y + z + 4)$
PROTeCt	$T(T(x) + y + z + 4)$	$T(T(x) + y + z + 4)$
E-PROTeCt	$T(x) + T(z) + y + 4$	$T(x) + T(z) + y + 4$

A Figura 5.1 apresenta um gráfico mostrando a sobrecarga da quantidade de dados de cada abordagem, neste caso são utilizados 10 *bytes* para o *hash*, 10 *bytes* para o *Access Token*, 20 *bytes* para o MAC e uma sobrecarga de 52 *bytes* relacionada aos cabeçalhos das camadas inferiores. A linha horizontal marca o tamanho máximo do quadro definido pelo padrão IEEE 802.15.4, que são 127 *bytes*, qualquer quadro acima desse tamanho deve ser fragmentado. Como pode ser visto, as abordagens propostas por este trabalho introduzem uma sobrecarga significativa no que tange o tamanho das mensagens, limitando a quantidade de dados úteis a serem enviadas em um único quadro. Ambos UPECSI e Sem Segurança possuem a mesma sobrecarga, relativa aos 4 *bytes* do protocolo de aplicação. Quando utilizando as abordagens seguras, o tamanho máximo de carga útil sem que ocorra fragmentação é de 15 *bytes* para o PROTeCt e 31 *bytes* para o E-PROTeCt, e nas abordagens sem segurança o valor é de 71 *bytes*.

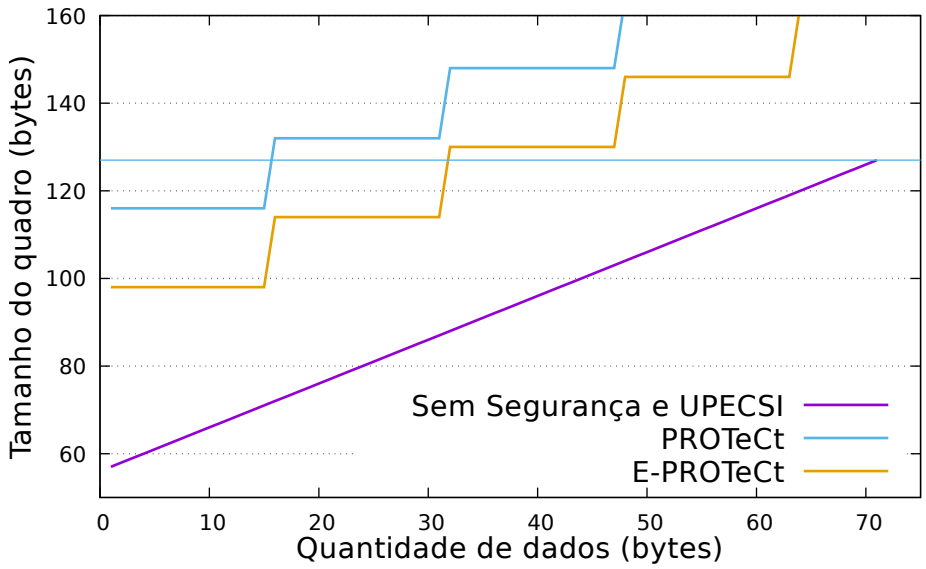


Figura 5.1: Sobrecarga relacionada ao tamanho do quadro para cada abordagem.

5.2 Avaliação Experimental

A seguir são apresentados os resultados obtidos através das simulações, primeiramente os ambientes simulados são descritos, em seguida uma análise de desempenho da rede é realizada. Posteriormente, o impacto das propostas na latência é discutido e, por fim, o impacto no tempo de vida é apresentado.

5.2.1 Configurações do Experimento

A avaliação experimental deste trabalho é realizada por meio de experimentos executados no simulador ns-3. As simulações contemplam as quatro abordagens descritas anteriormente. Com o intuito de fornecer a maior quantidade de informações possível, diversos cenários foram simulados, variando a quantidade de dispositivos IoT na rede e a frequência de mensagens geradas por eles. A pilha de protocolos utilizados pelos dispositivos IoT é reproduzida de acordo com a Figura 3.3, em todas as camadas padrões para dispositivos limitados foram escolhidos.

As camadas Física e de Enlace são definidas pelo padrão IEEE 802.15.4 [67], que opera na faixa de frequência de 2,4 GHz, possui velocidade máxima de 250 kbps e tamanho do quadro de até 127 *bytes*. Endereçamento de rede é possibilitado pelo protocolo IPv6 [101], que é utilizado em conjunto com o 6LoWPAN [45] para que este protocolo se torne viável para dispositivos limitados. Na camada de transporte o protocolo UDP [102] é utilizado, e no PROTeCt o DTLS [71] foi escolhido para prover um canal seguro. COAP [103] é o padrão utilizado para representar os dados na camada de aplicação, o qual introduz

uma sobrecarga de 4 *bytes* nos cabeçalhos. Em conjunto, as camadas Física, de Enlace e de Rede somam uma sobrecarga de 52 *bytes*, restando apenas 75 *bytes* paras as camadas superiores.

Todas as simulações foram executadas com uma carga útil de 10 *bytes*, o suficiente para representar diversos valores (por exemplo, a temperatura de um comodo ou o estado de uma lâmpada). A frequência em que os dados são gerados pelos dispositivos é variada em 5 segundos, 15 segundos, 30 segundos, 1 minuto, 2 minutos, 3 minutos, 4 minutos e 5 minutos. A quantidade de dispositivos na rede IoT é variada em 4, 8, 16, 32, 64, 128 e 160. Os dispositivos IoT são conectados a um dispositivo de borda, que realiza o encaminhamento para a plataforma de nuvem, o enlace entre o dispositivo de borda e a plataforma de nuvem possui uma latência de 80 ms, simulando a natureza não confiável da Internet. A Figura 5.2 apresenta o cenário de simulação com 20 dispositivos, círculos azuis representam dispositivos IoT, o círculo verde representa o dispositivo de borda e o círculo laranja representa a plataforma de nuvem. Todos os dispositivos IoT estão no raio de alcance do dispositivo de borda, ou seja, todas as comunicações dentro da rede IoT ocorrem em um único salto. Não há interferência externa no ambiente, apenas possíveis colisões entre os dispositivos podem prejudicar uma comunicação já ativa.

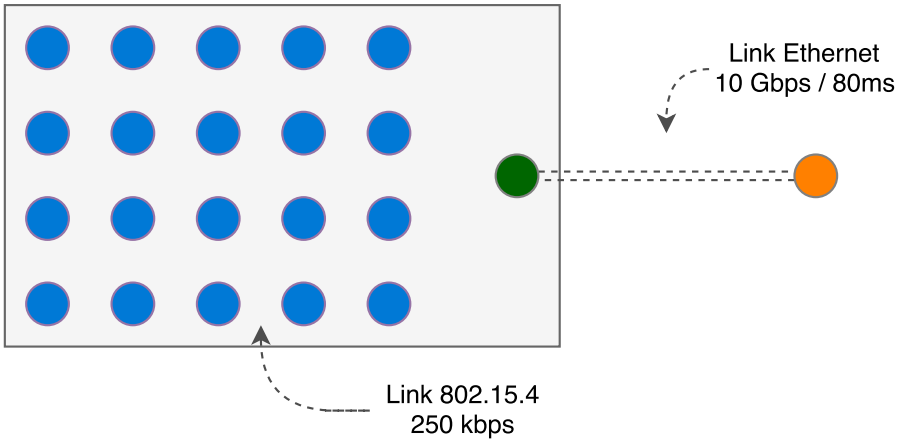


Figura 5.2: Cenário de simulação com 20 dispositivos IoT.

O consumo de energia e a latência referente à execução das operações criptográficas pelos dispositivos foi realizada por meio das medições realizadas por Zhang *et al.* [1]. Este trabalho avaliou diversas implementações do AES na plataforma Micaz Mote, que possui um micro controlador ATmega128L de 8 bits, 128 KB de memória RAM, 512 KM de memória ROM, um transceptor de radiofrequência CC2420 [104] de 2,4 GZ compatível com o padrão IEEE 802.15.4. O consumo de energia referente ao envio e recebimento dos dados é referente ao *datasheet* do CC2420 [104]. A latência das operações criptográficas executadas pelo *gateway* foi obtida do estudo realizado por Fisher *et al.* [105], que avalia a execução do AES na plataforma Raspberry Pi Modelo A [106], uma plataforma aberta

que possui um processador ARM de 700 MHz e 256 MB de memória RAM. A Tabela 5.3 apresenta os valores de latência e consumo de corrente relacionados à execução do AES nas plataformas mencionadas acima e o consumo relacionado ao transceptor dos dispositivos IoT.

Tabela 5.3: Latência (ms) and corrente consumida (mA) dos dispositivos quando encriptando 16 *bytes* de dados utilizando o AES-128 [1]; e consumo de corrente do rádio dos dispositivos IoT para transmissão e recepção.

Item	Valor
Criptografia (latência)	0.3506 ms
Criptografia (corrente)	25.501 mAJ
Transmissão (TX)	17.4 mAJ
Recepção (RX)	18.8 mAJ

As próximas subseções apresentam uma análise de latência e tempo de vida dos dispositivos. A latência é relativa à diferença entre o momento em que o dado é gerado no dispositivo IoT e o momento em que ele é recebido pela plataforma de nuvem, os resultados mostram a média da latência de todos os pacotes transmitidos durante a simulação, além do desvio padrão desta média. O tempo de vida de um dispositivo é referente à vida útil de uma bateria de 26000 mAh. O consumo de energia é relativo à soma das energias consumidas pelas operações criptográficas, transmissão e recepção dos dados nos dispositivos. Por este motivo, o tempo de vida obtido é maior que na realidade, pois os dispositivos executam várias outras tarefas que também consomem energia, como por exemplo os processos do Sistema Operacional. As chaves criptográficas são geradas apenas uma vez (esta tarefa consome uma quantidade considerável de energia, e leva alguns milissegundos para ser concluída), ou seja, as chaves não mudam durante a simulação.

5.2.2 Análise de Saturação da Rede

Primeiramente é realizada uma análise referente à saturação da rede, permitindo identificar quais dos cenários simulados não são comportados pela rede IoT, independente das implementações a serem avaliadas. A Figura 5.3 exibe a razão entre a vazão alcançada e a vazão gerada, para a implementação sem segurança (com 64 dispositivos), mostrando em que cenários a rede não possui capacidade para comportar o tráfego gerado. Como pode ser visto, quando a vazão gerada ultrapassa a marca de 1000 bps a vazão da rede cai abruptamente, pois os pacotes gerados não conseguem ser enviados, congestionando a rede. Os cenários em que este fato ocorre são: todas as simulações com 160 dispositivos e as simulações com 64 e 128 e pacotes gerados a cada 5 segundos.

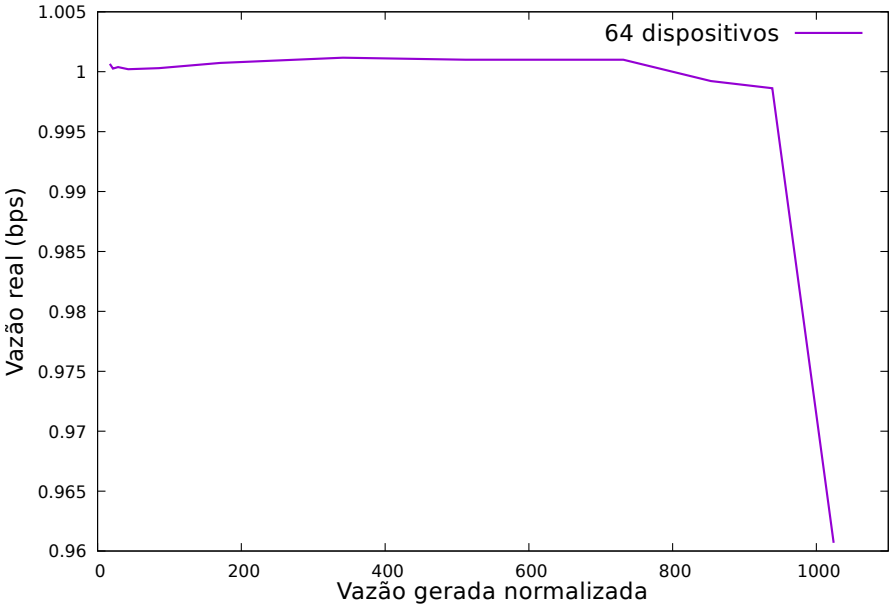
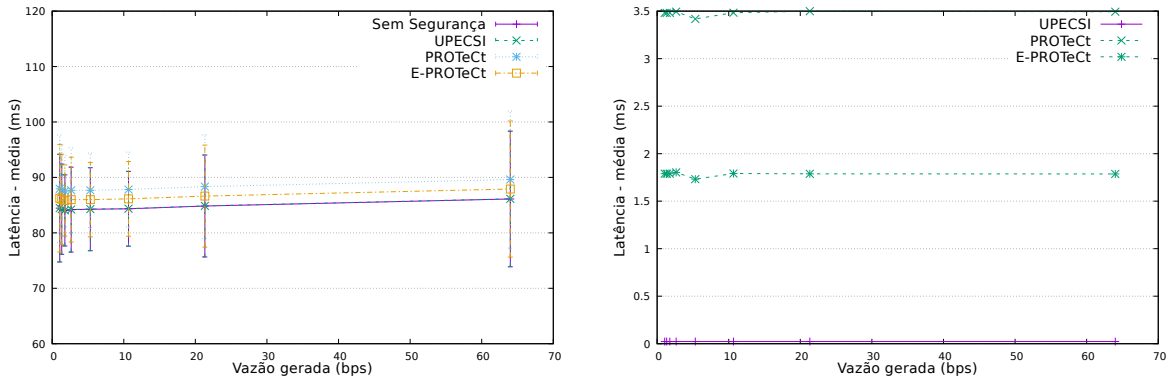


Figura 5.3: Razão entre a vazão real da rede e a vazão gerada em redes sem segurança para 64 dispositivos.

5.2.3 Análise da Latência pela Vazão

As Figuras 5.4 a 5.10 apresentam a análise da latência média pela vazão gerada em simulações com diferentes quantidades de dispositivos. O primeiro gráfico de cada figura representa a latência média pela vazão gerada das 4 abordagens simuladas, enquanto o segundo representa a diferença entre a latência média das abordagens UPECSI, PROTeCt e E-PROTeCt em relação a simulação sem segurança.

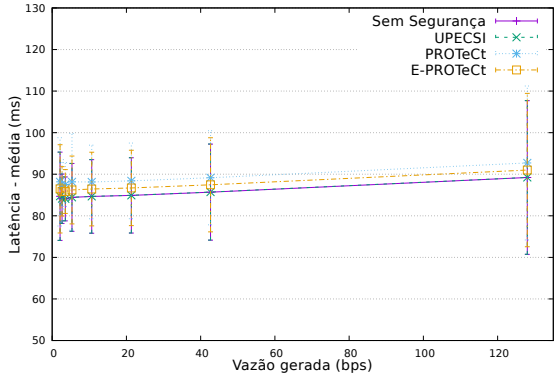
O primeiro gráfico das figuras deixa claro que, em relação ao atraso de 80 ms do enlace *gateway*-nuvem, o atraso da rede IoT tem um impacto bem menor no desempenho



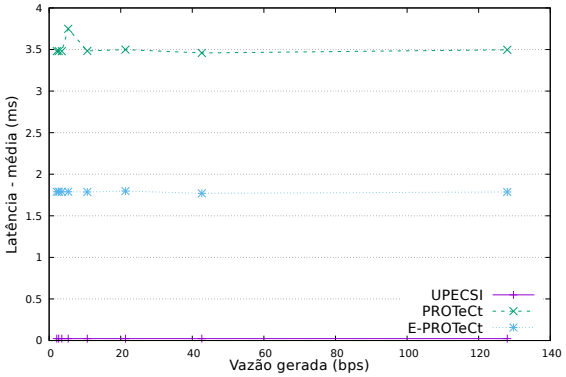
(a) Latência média de cada abordagem pela vazão gerada.

(b) Latência média tomando a abordagem sem segurança como parâmetro.

Figura 5.4: Análise da latência para uma rede IoT com 4 dispositivos.

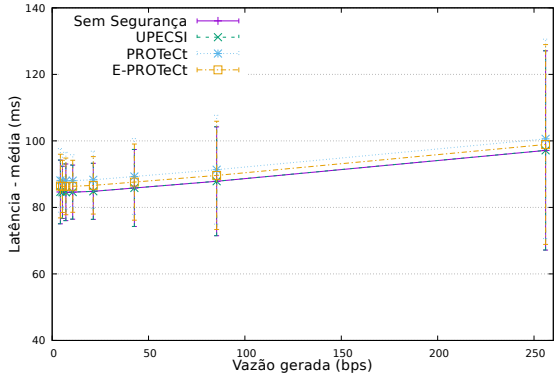


(a) Latência média de cada abordagem pela vazão gerada.

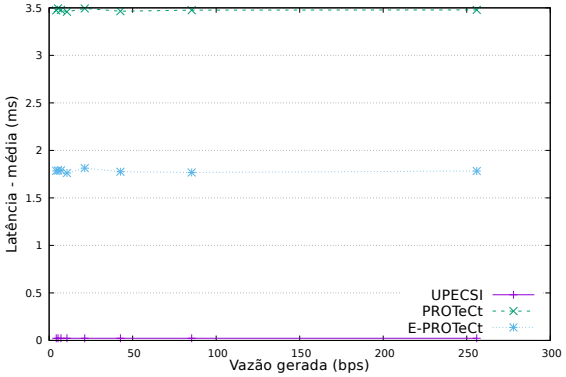


(b) Latência média tomando a abordagem sem segurança como parâmetro.

Figura 5.5: Análise da latência para uma rede IoT com 8 dispositivos.

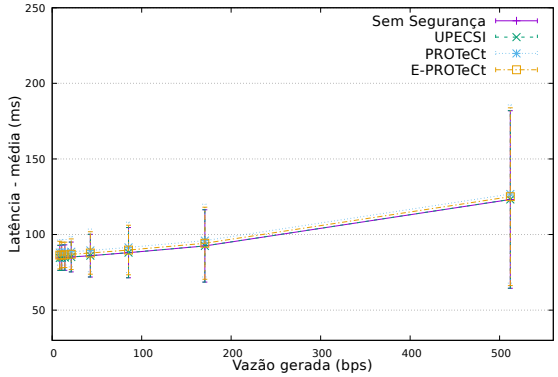


(a) Latência média de cada abordagem pela vazão gerada.

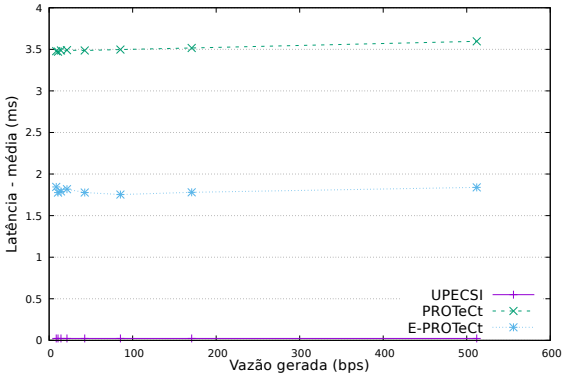


(b) Latência média tomando a abordagem sem segurança como parâmetro.

Figura 5.6: Análise da latência para uma rede IoT com 16 dispositivos.

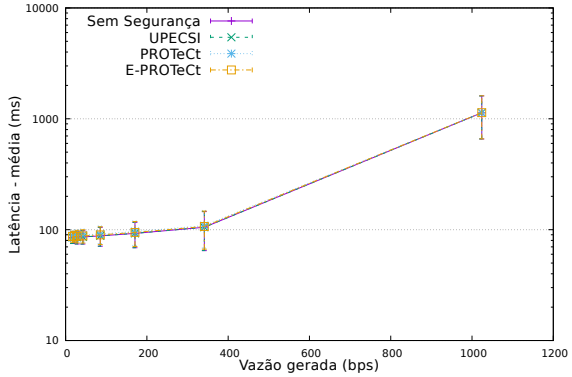


(a) Latência média de cada abordagem pela vazão gerada.

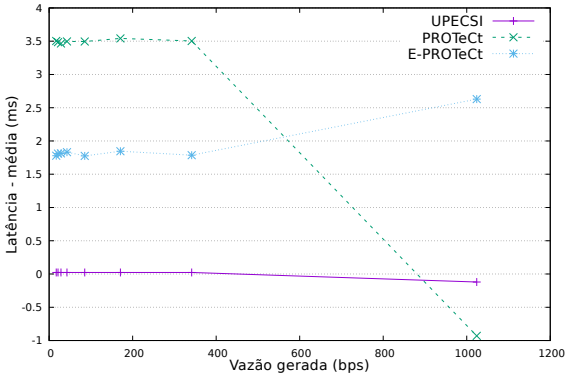


(b) Latência média tomando a abordagem sem segurança como parâmetro.

Figura 5.7: Análise da latência para uma rede IoT com 32 dispositivos.

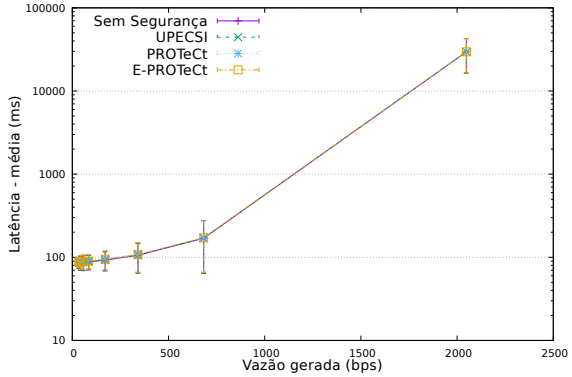


(a) Latência média de cada abordagem pela vazão gerada.

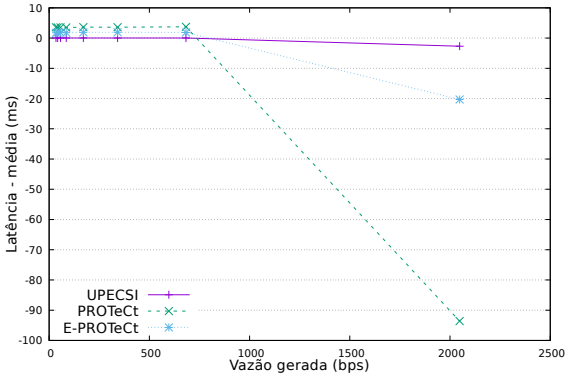


(b) Latência média tomando a abordagem sem segurança como parâmetro.

Figura 5.8: Análise da latência para uma rede IoT com 64 dispositivos.

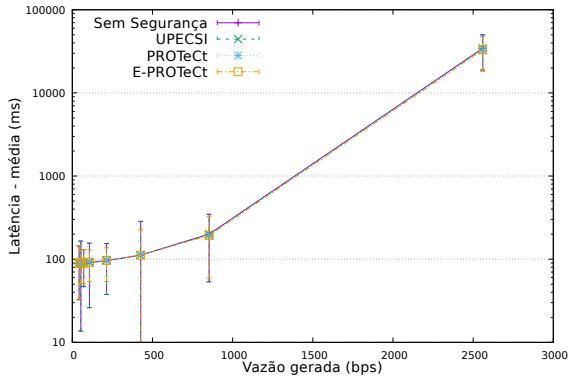


(a) Latência média de cada abordagem pela vazão gerada.

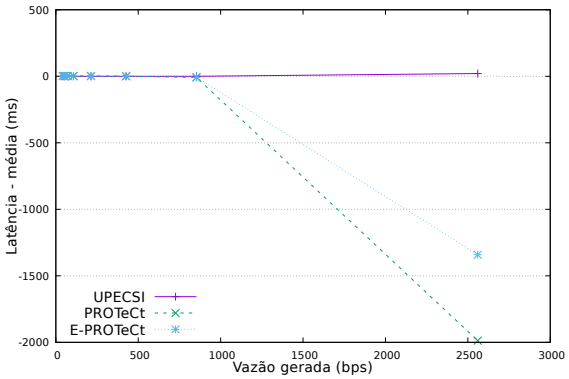


(b) Latência média tomando a abordagem sem segurança como parâmetro.

Figura 5.9: Análise da latência para uma rede IoT com 128 dispositivos.



(a) Latência média de cada abordagem pela vazão gerada.



(b) Latência média tomando a abordagem sem segurança como parâmetro.

Figura 5.10: Análise da latência para uma rede IoT com 160 dispositivos.

da rede toda. Também é possível notar que a partir dos limites encontrados na análise de saturação da rede tanto a latência média quanto seu desvio padrão aumentam consideravelmente em todas as abordagens, o que indica que as mesmas não alteram a capacidade da rede.

O segundo gráfico das figuras exibe a diferença entre as abordagens, neste gráfico é possível avaliar a sobrecarga das abordagens analisadas com maior facilidade. A abordagem UPECSI possui diferença negligenciável, que é referente ao processamento das mensagens no PEP, que executa as tarefas necessárias para proteger a comunicação entre ele e a nuvem. Pode-se notar que, mesmo em uma rede com 160 dispositivos o PEP não é sobrecarregado, ou seja, o dispositivo escolhido tem poder suficiente para lidar com as redes simuladas. A diferença da latência média entre as abordagens PROTeCt e E-PROTeCt e a simulação sem segurança alguma é, em média, de 3,5 ms e 1,75 ms, respectivamente. Considerando apenas as simulações onde não ocorre saturação da rede, as diferenças são quase constantes, independentemente da quantidade de dispositivos ou vazão gerada. A redução da sobrecarga da latência entre as abordagens PROTeCt e E-PROTeCt é, em média, de 50%, fornecendo indícios de que a implementação de mecanismos de segurança na camada de aplicação oferece grande vantagem em relação à utilização de um protocolo de segurança na camada de transporte.

Redes IoT de dispositivos limitados possuem baixa frequência de geração de dados, dessa forma, uma sobrecarga de 1,75 ms na latência média pode ser considerada completamente viável, fornecendo um excelente custo-benefício pela implementação de segurança na rede.

5.2.4 Análise do Tempo de Vida pela Vazão

As Figuras 5.11 a 5.17 apresentam uma análise referente ao tempo de vida dos dispositivos em relação à vazão gerada, em redes de diversos tamanhos. O primeiro gráfico de cada figura exite o tempo de vida, em anos, de acordo com o aumento da vazão gerada na reed IoT, enquanto o segundo apresenta a razão entre o tempo de vida das abordagens UPECSI, PROTeCt e E-PROTeCt em relação a simulação sem segurança. Para o tempo de vida foi escolhida a análise da razão pois, neste caso, o atraso do enlace *gateway*-nuvem não afeta este parâmetro, dessa forma a razão do tempo re vida representa exatamente a sobrecarga introduzida pelas abordagens analisadas.

O consumo de energia resultante das simulações é referente a dois fatores: o consumo do rádio dos dispositivos e o consumo das operações de criptografia. O primeiro ocorre em todas as abordagens, enquanto o segundo ocorre apenas nas abordagens propostas, que protegem os dados já nos dispositivos IoT. O consumo do rádio é sempre maior que o das operações criptográficas, e a diferença entre as abordagens propostas e as simulações

sem segurança aumentam de forma linear, portanto o impacto do consumo das operações critográficas na diferença entre as abordagens é maior nas simulações de menor vazão.

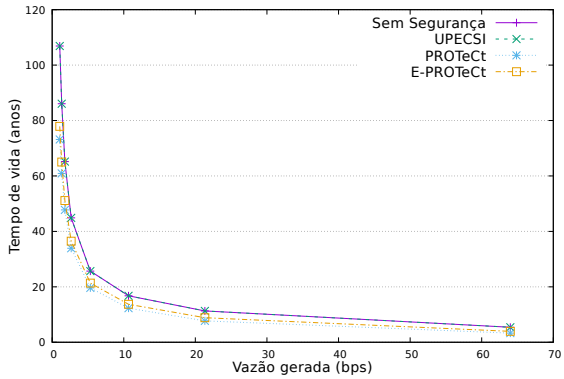
De acordo com o primeiro gráfico das figuras, o tempo de vida dos dispositivos diminui exponencialmente em relação a vazão gerada, indicando que, em dispositivos com fonte de energia limitada, é importante manter uma taxa de envio de dados baixa. A queda no tempo de vida também é inversamente proporcional ao aumento da quantidade de dispositivos na rede, pois como o meio é compartilhado todos os dispositivos consomem energia recebendo os pacotes enviados pelos demais, para então verificar o endereço de destino e descartar pacotes. Vale ressaltar que o alto tempo de vida observado nos experimentos de baixa vazão se devem pois vários fatores relativos ao gasto de energia dos dispositivos não estão sendo levados em conta.

A razão do tempo de vida em relação à abordagem sem segurança, indicada no segundo gráfico de cada figura, exibe de forma clara a sobrecarga introduzida por cada abordagem. A UPECSI não apresenta qualquer sobrecarga, pois, assim como na análise da latência, a rede IoT se comporta da mesma forma que nas simulações sem segurança. Já as abordagens propostas por este trabalho, PROTeCt e E-PROTeCt apresentam redução média do tempo de vida de 78,20 % e 84,49 %, respectivamente. Pode-se notar que a redução no tempo de vida aumenta conforme a quantidade de pacotes trafegados e de dispositivos na rede também aumentam, pois os dispositivos gastam energia recebendo os pacotes das outras comunicações, amplificando os efeitos da sobrecarga introduzida pelas abordagens.

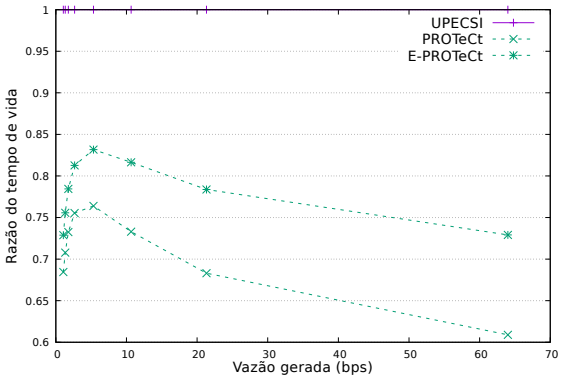
A abordagem E-PROTeCt apresenta uma menor redução no tempo de vida dos dispositivos, se comparada à PROTeCt. A razão do tempo de vida em relação à simulação sem segurança apresenta uma diferença absoluta entre as duas abordagens que varia de 0,04 % a 12,02 %, resultado em um ganho que varia entre 5,18 % e 19,74 %. Estes valores indicam os ganhos no aumento do tempo de vida dos dispositivos quando utilizando mecanismos de segurança na camada de aplicação. Estes valores, embora menos expressivos que as reduções na latência, indicam que a abordagem do E-PROTeCt traz benefícios reais em relação à PROTeCt.

5.3 Considerações Finais

Este capítulo apresentou a avaliação das arquiteturas propostas. A análise analítica mostrou o aumento na sobrecarga dos dispositivos, e pode ser visto que a abordagem do E-PROTeCt de fato traz um ganho significativo de desempenho, principalmente quando levamos em conta o *padding* do algoritmo de criptografia. A avaliação experimental explorou diversos cenários, mostrando os limites de uma rede IoT que utiliza os protocolos

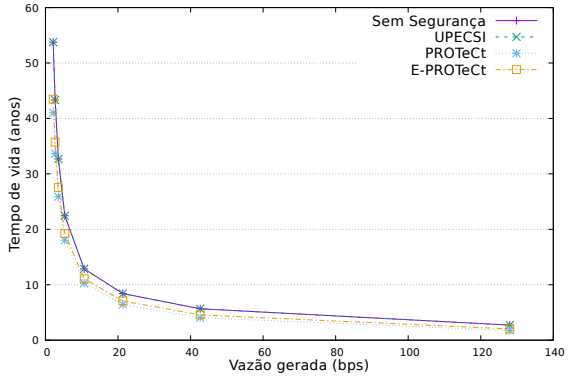


(a) Tempo de vida, em anos, de cada abordagem pela vazão gerada.

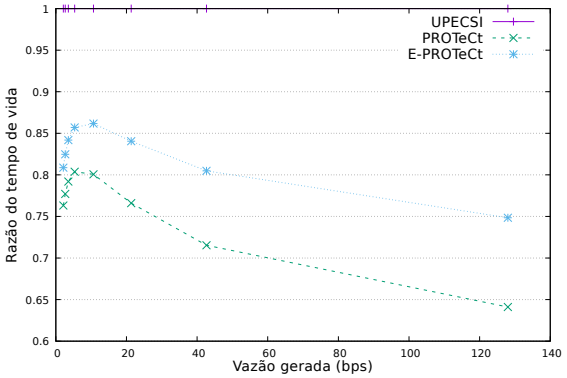


(b) Taxa de redução do tempo de vida em relação à abordagem sem segurança.

Figura 5.11: Análise do tempo de vida pela vazão gerada em uma rede IoT com 4 dispositivos.

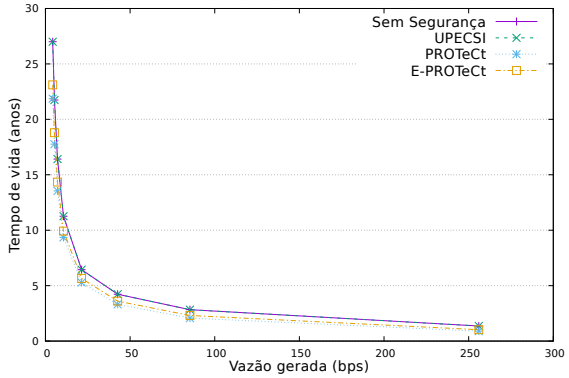


(a) Tempo de vida, em anos, de cada abordagem pela vazão gerada.

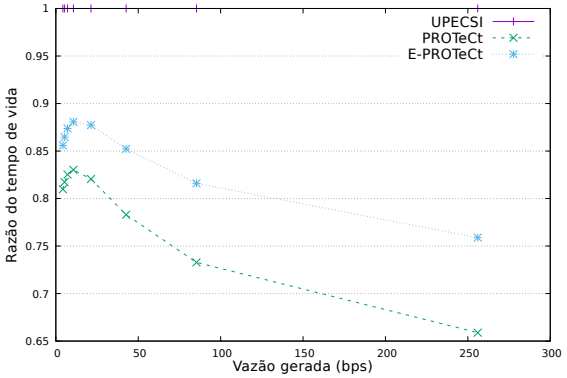


(b) Taxa de redução do tempo de vida em relação à abordagem sem segurança.

Figura 5.12: Análise do tempo de vida pela vazão gerada em uma rede IoT com 8 dispositivos.

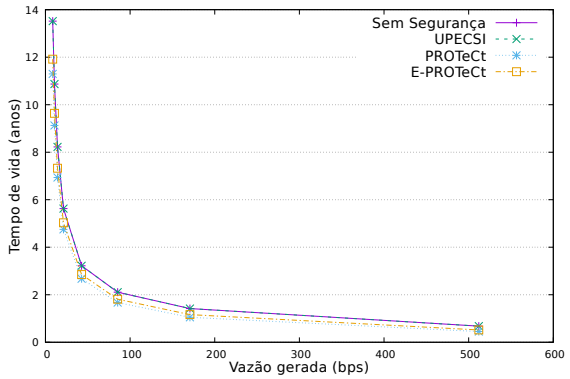


(a) Tempo de vida, em anos, de cada abordagem pela vazão gerada.

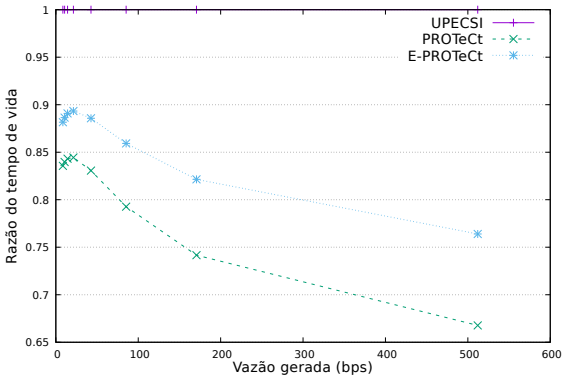


(b) Taxa de redução do tempo de vida em relação à abordagem sem segurança.

Figura 5.13: Análise do tempo de vida pela vazão gerada em uma rede IoT com 16 dispositivos.

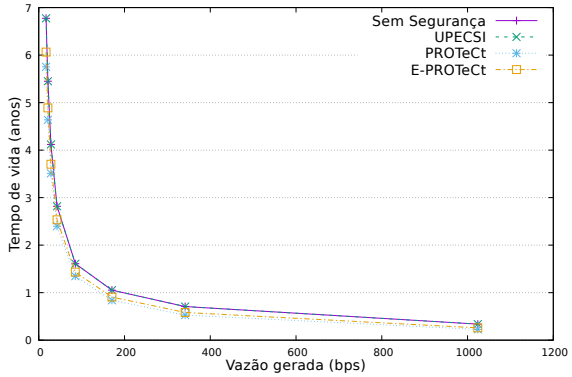


(a) Tempo de vida, em anos, de cada abordagem pela vazão gerada.

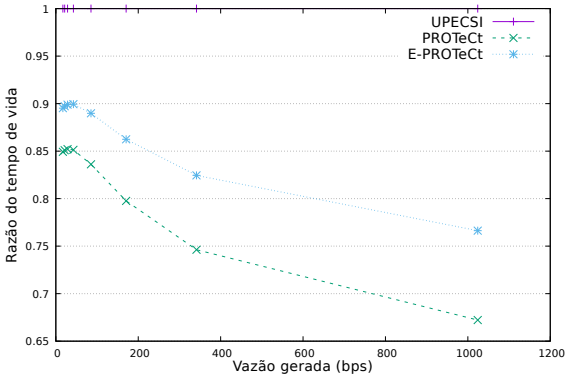


(b) Taxa de redução do tempo de vida em relação à abordagem sem segurança.

Figura 5.14: Análise do tempo de vida pela vazão gerada em uma rede IoT com 32 dispositivos.

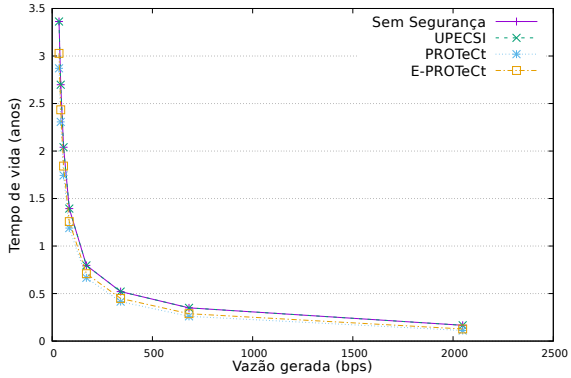


(a) Tempo de vida, em anos, de cada abordagem pela vazão gerada.

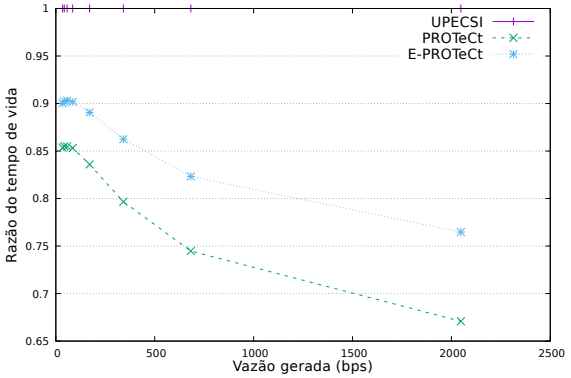


(b) Taxa de redução do tempo de vida em relação à abordagem sem segurança.

Figura 5.15: Análise do tempo de vida pela vazão gerada em uma rede IoT com 64 dispositivos.

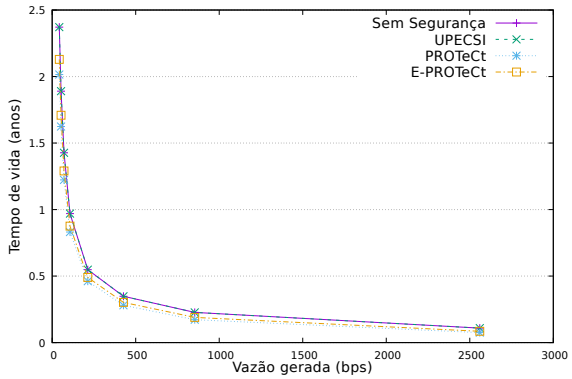


(a) Tempo de vida, em anos, de cada abordagem pela vazão gerada.

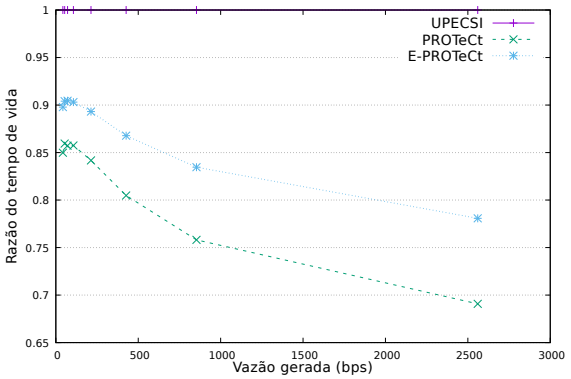


(b) Taxa de redução do tempo de vida em relação à abordagem sem segurança.

Figura 5.16: Análise do tempo de vida pela vazão gerada em uma rede IoT com 128 dispositivos.



(a) Tempo de vida, em anos, de cada abordagem pela vazão gerada.



(b) Taxa de redução do tempo de vida em relação à abordagem sem segurança.

Figura 5.17: Análise do tempo de vida pela vazão gerada em uma rede IoT com 160 dispositivos.

citados, pode-se perceber que, apesar de sobrecarga visível, a aplicação de mecanismos de segurança nos próprios dispositivos é completamente factível.

Capítulo 6

Conclusão

Este capítulo apresenta uma conclusão ao trabalho realizado, primeiramente uma introdução é apresentada, em seguida, na visão geral, um resumo do trabalho desenvolvido é apresentado. Uma revisão dos objetivos e contribuições é disposta, apresentando os principais resultados obtidos. Por fim, os trabalhos futuros são apresentados, elucidando possíveis tarefas a serem realizadas futuramente.

6.1 Visão geral

A integração da Internet das Coisas com a Computação em Nuvem, apesar de ser uma abordagem amplamente utilizada, oferce diversos desafios. A Internet das Coisas é formada por dispositivos de diferentes características, desde dispositivos severamente limitados, com baixo poder computacional, baixo alcance de comunicação e, muitas vezes, fonte de energia limitada, a dispositivos de capacidade computacional considerável e sem limitações de energia. Esta heterogeneidade traz um desafio no que tange a comunicação segura de dispositivos limitados com a Internet. Nesta área, é preciso desenvolver protocolos que possam ser executados por estes dispositivos de menor capacidade e, ao mesmo tempo, sejam compatíveis com os protocolos utilizados atualmente na Internet.

Estima-se que a Internet das Coisas será composta por bilhões de dispositivos, os quais irão gerar um volume imenso de dados. A utilização da Computação em Nuvem para armazenar e controlar o acesso a esses dados é de extrema utilidade, no entanto estes dados passam a estar sob controle da organização responsável pela paltaforma de nuvem. Este trabalho teve como objetivo assegurar que os requisitos do usuário sejam respeitados quando integrando suas redes de Internet das Coisas com a nuvem.

A arquitetura desenvolvida foi baseada na *User-driven Privacy Enforcement for Cloud-based Services in the IoT* (UPECSI), trabalho que fornece uma arquitetura integrada para prover privacidade na utilização da nuvem por usuários detentores de redes IoT.

A privacidade do usuário é assegurada por meio de armazenamento criptografado, onde apenas as entidades que devem acessar os dados possuem as chaves criptográficas para realizar esta operação. No entanto o UPECSI exige a utilização de um ponto central na rede do usuário, que fica responsável por criptografar as mensagens e enviá-las para a nuvem. Esta abordagem traz algumas desvantagens, tais como a inserção de um ponto único de falha, que, caso comprometido, não apenas pode obstruir o funcionamento da rede toda, como também pode revelar os dados de todos os dispositivos, já que o mesmo é responsável por protege-los.

6.2 Revisão dos objetivos e contribuições da dissertação

Este trabalho teve por objetivo fornecer uma arquitetura abrangente para privacidade na integração da Internet das Coisas com a computação em nuvem. Com o intuito de mitigar os problemas encontrados no UPECSI, foi proposta uma abordagem em que os próprios dispositivos IoT são responsáveis por aplicar as políticas de privacidade em seus dados. Na arquitetura proposta, os dispositivos de borda não possuem acesso aos dados, pois os mesmo são criptografados na origem, eliminando o ponto único de falha encontrado no UPECSI. Esta abordagem traz outros benefícios, a comunicação direta entre os dispositivos IoT e a nuvem possibilitam um maior controle dos mesmos, pois cada dispositivo é alcançável, e não representado por uma entidade única.

A arquitetura desenvolvida por este trabalho foi chamada de PROTeCt (do inglês, ***P****rivacy a****R****quitecture for integrati****O****n of internet of* ***T****hings and* ***C****loud computing*). Esta arquitetura foi aperfeiçoada, onde foi introduzido um mecanismo de proteção da comunicação diretamente na camada de aplicação, com o intuito de melhorar o desempenho da arquitetura, este aperfeiçoamento foi chamado de E-PROTeCt (do inglês, *Extended PROTeCt*).

A avaliação das arquiteturas propostas foi realizada por meio de simulações executadas no simulador *ns-3*, onde os resultados foram comparados com o UPECSI e com uma aplicação sem qualquer tipo de segurança. Os protocolos de comunicação foram desenvolvidos para serem suportados mesmo por dispositivos extremamente limitados, portanto o desempenho da arquitetura foi o alvo principal das análises realizadas. Foram simulados diversos cenários, variando a quantidade de dispositivos na rede e o tráfego gerado por eles. Os dispositivos utilizados são extremamente limitados, possuem CPU de 8 bits, memória RAM de 128 KB e memória ROM de 512 KB.

Os resultados obtidos mostraram que a sobrecarga introduzida pelos protocolos propostos, apesar de não serem negligenciáveis, não diminuem significativamente o desempe-

nho da rede, tanto em relação a latência das mensagens quanto em relação ao tempo de vida dos dispositivos. Este último parâmetro tem por objetivo mostrar que a arquitetura proposta é viável inclusive para dispositivos sem fonte de energia ilimitada (alimentados por bateria).

Partes deste trabalho foram publicadas durante seu desenvolvimento. Um trabalho preliminar, estudando a segurança de redes IoT, foi publicado no *IEEE 16th International Symposium on Network Computing and Applications (NCA)*, em 2016 [107]; uma primeira versão da arquitetura foi apresentada no *19th International Conference on Enterprise Information (ICEIS)*, em 2017 [108]; melhorias na arquitetura foram publicadas no *XVIII WTF 2017 Workshop de Testes e Tolerância a Falhas*, em 2017; e, por fim, uma versão final, com análise mais aprofundada, foi publicada no NCA de 2017 [109].

6.3 Trabalhos Futuros

Por fim, como trabalhos futuros, pode-se estudar a viabilidade de algoritmos de criptografia de fluxo e o AES em modo GCM, os quais aparentam necessitar de menos recursos (no caso da criptografia de fluxo) e não possuem a desvantagem do *padding*, eliminando o crescimento do tamanho dos pacotes enviados. Pode-se também implementar a arquitetura proposta em plataformas IoT, o que possibilitará um estudo mais aprofundado referente a sobrecarga nos dispositivos.

Referências

[1] Zhang, Fan, Reiner Dojen e Tom Coffey: *Comparative performance and energy consumption analysis of different aes implementations on a wireless sensor network node*. International Journal of Sensor Networks, 10(4):192–201, 2011. xiii, 30, 57, 58

[2] Chui, Michael, Markus Löffler e Roger Roberts: *The internet of things*. McKinsey Quarterly, 2(2010):1–9, 2010. 1

[3] Sundmaeker, Harald, Patrick Guillemin, Peter Friess e Sylvie Woelfflé: *Vision and challenges for realising the internet of things*. Cluster of European Research Projects on the Internet of Things, European Commision, 2010. 1

[4] Akyildiz, Ian F e Mehmet Can Vuran: *Wireless sensor networks*, volume 4. John Wiley & Sons, 2010. 1

[5] Group, WPAN Working: *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*. IEEE Standard for Information Technology, 2006. 2

[6] Granjal, Jorge, Edmundo Monteiro e Jorge Sá Silva: *Security in the integration of low-power wireless sensor networks with the internet: A survey*. Ad Hoc Networks, 24:264–287, 2015. 2

[7] Lee, Kevin, David Murray, Danny Hughes e Wouter Joosen: *Extending sensor networks into the cloud using amazon web services*. Em *IEEE International Conference on Networked Embedded Systems for Enterprise Applications*, páginas 1–7, 2010. 2

[8] Aazam, Mohammad, Imran Khan, Aymen Abdullah Alsaffar e Eui Nam Huh: *Cloud of things: Integrating internet of things and cloud computing and the issues involved*. Em *Proceedings of 11th International Bhurban Conference on Applied Sciences & Technology*, páginas 414–419, 2014. 2, 32

[9] Fox, Geoffrey C, Supun Kamburugamuve e Ryan D Hartman: *Architecture and measured characteristics of a cloud based internet of things*. Em *International Conference on Collaboration Technologies and Systems*, 2012. 2

[10] Botta, Alessio, Walter de Donato, Valerio Persico e Antonio Pescapé: *Integration of cloud computing and internet of things: a survey*. Future Generation Computer Systems, 56:684–700, 2016. 2, 32

[11] Avizienis, Algirdas, J C Laprie, Brian Randell e Carl Landwehr: *Basic concepts and taxonomy of dependable and secure computing*. IEEE transactions on dependable and secure computing, 1(1):11–33, 2004. 3, 5

[12] Stallings, William: *Cryptography and network security: principles and practices*. Pearson Education India, 2006. 3, 6

[13] Henze, Martin, Lars Hermerschmidt, Daniel Kerpen, Roger Häußling, Bernhard Rumpe e Klaus Wehrle: *A comprehensive approach to privacy in the cloud-based internet of things*. Future Generation Computer Systems, 56:701–718, 2016. 3, 20, 34, 36, 48, 53

[14] Guttman, Barbara e Edward A Roback: *An introduction to computer security: the NIST handbook*. DIANE Publishing, 1995. 5

[15] ISO, IS: *7498-2. information processing systems open systems interconnection basic reference model-part 2: Security architecture*. ISO Geneva, Switzerland, 1989. 5

[16] Charles P. Pfleeger, Shari Lawrence Pfleeger, Jonathan Margulies: *Security in Computing*. Prentice Hall, 5ª edição, 2015, ISBN 0134085043,9780134085043. 6, 12

[17] Matt, Bishop *et al.*: *Introduction to computer security*. Pearson Education India, 2006. 7, 10, 12

[18] SHIREY, R: *Internet security glossary, version 2*, 2007. 8, 12, 14

[19] Shirey, Dr. Rob: *Security architecture for internet protocols: A guide for protocol designs and standards*. Internet-draft draft-irtf-psrg-secarch-sect1-00, Internet Engineering Task Force, novembro 1994. `https://tools.ietf.org/html/draft-irtf-psrg-secarch-sect1-00`. 8

[20] Goodin, Dan: *10,000 hotmail passwords mysteriously leaked to web*. The Register, nov 2009. `http://www.theregister.co.uk/2009/10/05/hotmail_passwords_leaked/`. 13

[21] Leswing, Kif: *Yahoo confirms major breach — and it could be the largest hack of all time*. Business Insider, sep 2016. `http://uk.businessinsider.com/yahoo-hack-by-state-sponsored-actor-biggest-of-all-time-2016-9?r=US&IR=T`. 13

[22] Greenberg, Andy: *Hack brief: 412m accounts breached on friendfinder sex sites*. Wired, nov 2016. `https://www.wired.com/2016/11/hack-brief-412m-accounts-breached-friendfinder-sex-sites/`. 13

[23] Malone, David e Kevin Maher: *Investigating the distribution of password choices*. Em *Proceedings of the 21st international conference on World Wide Web*, páginas 301–310. ACM, 2012. 13

[24] O'Gorman, Lawrence: *Comparing passwords, tokens, and biometrics for user authentication*. Proceedings of the IEEE, 91(12):2021–2040, 2003. 14

[25] Recordon, David e Drummond Reed: *Openid 2.0: a platform for user-centric identity management*. Em *Proceedings of the second ACM workshop on Digital identity management*, páginas 11–16. ACM, 2006. 14

[26] Biham, Eli e Adi Shamir: *Differential cryptanalysis of the data encryption standard*. Springer Science & Business Media, 2012. 17

[27] Basu, Abhi, Abhilasha Bhargav-Spantzel e George Pappas: *Intel® aes-ni performance testing over full disk encryption*, maio 2012. 17

[28] Arora, Mohit: *How secure is aes against brute force attacks*. EE Times, 5(7), 2012. 17

[29] Daemen, Joan e Vincent Rijmen: *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013. 17

[30] Schneier, Bruce, John Kelsey, Doug Whiting, David Wagner, Chris Hall, Niels Ferguson, Tadayoshi Kohno e Mike Stay: *The twofish team's final comments on aes selection*. AES round, 2, 2000. 17

[31] Galperin, Slava, Stefan Santesson, Michael Myers, Ambarish Malpani e Carlisle Adams: *X. 509 internet public key infrastructure online certificate status protocol-ocsp*. 2013. 18

[32] Dierks, Tim: *The transport layer security (tls) protocol version 1.2*. RFC 5246, IETF Trust, 2008. 18, 32

[33] Leal, Bernardo, Luigi Atzori, Daniel Giusto, Antonio Iera e Giacomo Morabito: *The Internet of Things: 20th Tyrrhenian Workshop on Digital Communications*. Springer-Verlag New York, 2010, ISBN 1441916733,9781441916730. 20

[34] Atzori, Luigi, Antonio Iera e Giacomo Morabito: *The internet of things: A survey*. Computer networks, 54(15):2787–2805, 2010. 20

[35] Council, N: *Six technologies with potential impacts on us interests out to 2025*. Disruptive Civil Technologies2008, 2008. 20

[36] Gubbi, Jayavardhana, Rajkumar Buyya, Slaven Marusic e Marimuthu Palaniswami: *Internet of things (iot): A vision, architectural elements, and future directions*. Future generation computer systems, 29(7):1645–1660, 2013. 21

[37] Ahson, Syed A e Mohammad Ilyas: *RFID handbook: applications, technology, security, and privacy*. CRC press, 2008. 21

[38] Potdar, Vidyasagar, Atif Sharif e Elizabeth Chang: *Wireless sensor networks: A survey*. Em *Advanced Information Networking and Applications Workshops, 2009. WAINA'09. International Conference on*, páginas 636–641. IEEE, 2009. 22

[39] Whitmore, Andrew, Anurag Agarwal e Li Da Xu: *The internet of things—a survey of topics and trends*. Information Systems Frontiers, 17(2):261–274, 2015. 22

[40] Aberer, Karl, Manfred Hauswirth e Ali Salehi: *Middleware support for the internet of things*. Proceedings of 5. GI/ITG KuVS Fachgespraech—Drahtlose Sensornetze, páginas 15–19, 2006. 22

[41] Gómez-Goiri, Aitor e Diego López-de Ipiña: *A triple space-based semantic distributed middleware for internet of things*. Em *International Conference on Web Engineering*, páginas 447–458. Springer, 2010. 22

[42] Buettner, Michael, Gary V Yee, Eric Anderson e Richard Han: *X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks*. Em *Proceedings of the 4th international conference on Embedded networked sensor systems*, páginas 307–320. ACM, 2006. 22

[43] Yu, Yan, Ramesh Govindan e Deborah Estrin: *Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks*. 2001. 22

[44] Sheng, Zhengguo, Shusen Yang, Yifan Yu, Athanasios Vasilakos, Julie Mccann e Kin Leung: *A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities*. IEEE Wireless Communications, 20(6):91–98, 2013. 22

[45] Kushalnagar, Nandakishore, Gabriel Montenegro e Christian Schumacher: *Ipv6 over low-power wireless personal area networks (6lowpans): overview, assumptions, problem statement, and goals*. Relatório Técnico, IETF Trust, 2007. 22, 29, 56

[46] Deering, Stephen E: *Internet protocol, version 6 (ipv6) specification*. RFC 2460, The Internet Society, 1998. 22, 29

[47] Postel, Jon: *User datagram protocol (udp)*. RFC 768, IETF, 1980. 22, 30

[48] Postel, Jon: *Transmission control protocol (tcp)*. RFC 793, Defense Advanced Research Projects Agency, 1981. 22, 30

[49] Bandyopadhyay, Debasis e Jaydip Sen: *Internet of things: Applications and challenges in technology and standardization*. Wireless Personal Communications, 58(1):49–69, 2011. 23

[50] Bertoni, Guido, Luca Breveglieri e Matteo Venturi: *Power aware design of an elliptic curve coprocessor for 8 bit platforms*. Em *Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on*, páginas 5–pp. IEEE, 2006. 23

[51] Liu, Jing, Yang Xiao e CL Philip Chen: *Authentication and access control in the internet of things*. Em *Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*, páginas 588–592. IEEE, 2012. 23

[52] Stankovic, John A: *Research directions for the internet of things*. IEEE Internet of Things Journal, 1(1):3–9, 2014. 23

[53] Mell, Peter, Tim Grance *et al.*: *The nist definition of cloud computing*. 2011. 23, 25, 26

[54] Erl, Thomas, Ricardo Puttini e Zaigham Mahmood: *Cloud computing: concepts, technology & architecture*. Pearson Education, 2013. 23, 24, 25

[55] Alford, Ted e Gwen Morton: *The economics of cloud computing*. Booz Allen Hamilton, 2009. 24

[56] Patel, Pankesh, Ajith H Ranabahu e Amit P Sheth: *Service level agreement in cloud computing*. 2009. 24

[57] Ryan, Mark D: *Cloud computing security: The scientific challenge, and a survey of solutions*. Journal of Systems and Software, 86(9):2263–2268, 2013. 26

[58] Ali, Mazhar, Samee U Khan e Athanasios V Vasilakos: *Security in cloud computing: Opportunities and challenges*. Information Sciences, 305:357–383, 2015. 27

[59] Gentry, Craig *et al.*: *Fully homomorphic encryption using ideal lattices*. Em *STOC*, volume 9, páginas 169–178, 2009. 27

[60] Betge-Brezetz, Stephane, Guy Bertrand Kamga, Marie Pascale Dupont e Aoues Guesmi: *End-to-end privacy policy enforcement in cloud infrastructure*. Em *Cloud Networking (CloudNet), 2013 IEEE 2nd International Conference on*, páginas 25–32. IEEE, 2013. 27

[61] Dorri, Ali, S Kanhere, Raja Jurdak e Praveen Gauravaram: *Blockchain for iot security and privacy: The case study of a smart home*. Em *proceedings of the 2nd IEEE Workshop on security, privacy, and trust in the Internet of things (PERCOM), Hawaii, USA*, 2017. 28

[62] Nakamoto, Satoshi: *Bitcoin: A peer-to-peer electronic cash system*. 2009. 28

[63] Christidis, Konstantinos e Michael Devetsikiotis: *Blockchains and smart contracts for the internet of things*. IEEE Access, 4:2292–2303, 2016. 28

[64] Pacheco, Jesus e Salim Hariri: *Iot security framework for smart cyber infrastructures*. Em *Foundations and Applications of Self* Systems, IEEE International Workshops on*, páginas 242–247. IEEE, 2016. 28

[65] Huang, Xin, Paul Craig, Hangyu Lin e Zheng Yan: *Seciot: a security framework for the internet of things*. Security and Communication Networks, 2015. 28

[66] Granjal, Jorge, Edmundo Monteiro e Jorge Sá Silva: *Security for the internet of things: a survey of existing protocols and open research issues*. IEEE Communications Surveys & Tutorials, 17(3):1294–1312, 2015. 29

[67] Group, IEEE P802.15 Working: *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*. IEEE Standard for Information Technology, 2015. 29, 56

[68] Thubert, P, A Brandt, J Hui, R Kelsey, P Levis, K Pister, R Struik, JP Vasseur e R Alexander: *Rpl: Ipv6 routing protocol for low power and lossy networks*. RFC 6550, 2012. 29

[69] Shelby, Zach, Klaus Hartke e Carsten Bormann: *The constrained application protocol (coap)*. RFC 7275, Internet Engineering Task Force (IETF), 2014. 30

[70] Fielding, R e J Reschke: *Hypertext transfer orotocol (http/1.1): Message syntax and routing*, 2014. 30

[71] McGrew, David e Eric Rescorla: *Datagram transport layer security (dtls) extension to establish keys for secure real-time transport protocol (srtp)*. 2010. 30, 46, 56

[72] Dierks, Tim: *The transport layer security (tls) protocol version 1.2*. 2008. 30

[73] Tschofenig, H e T Fossati: *Transport layer security (tls)/datagram transport layer security (dtls) profiles for the internet of things*. RFC 7925, 2016. 30, 39, 49

[74] Kothmayr, Thomas, Corinna Schmitt, Wen Hu, Michael Brünig e Georg Carle: *Dtls based security and two-way authentication for the internet of things*. Ad Hoc Networks, 11(8):2710–2723, 2013. 30

[75] Zissis, Dimitrios e Dimitrios Lekkas: *Addressing cloud computing security issues*. Future Generation computer systems, 28(3):583–592, 2012. 31

[76] Takabi, Hassan, James BD Joshi e Gail Joon Ahn: *Security and privacy challenges in cloud computing environments*. IEEE Security & Privacy, 8(6):24–31, 2010. 31

[77] Jansen, Wayne e Timothy Grance: *Sp 800-144. guidelines on security and privacy in public cloud computing*. 2011. 31

[78] Group, Top Threats Working *et al.*: *The notorious nine: cloud computing top threats in 2013*. Cloud Security Alliance, 2013. 31

[79] Catteddu, Daniele: *Cloud computing: benefits, risks and recommendations for information security*. Em *Web application security*, páginas 17–17. Springer, 2010. 31

[80] Islam, Tariqul, D Manivannan e Sherali Zeadally: *A classification and characterization of security threats in cloud computing*. Int. J. Next-Gener. Comput, 7(1), 2016. 31

[81] Sugumar, Ramalingam e Sharmila Banu Sheik Imam: *Symmetric encryption algorithm to secure outsourced data in public cloud storage*. Indian Journal of Science and Technology, 8(23):1, 2015. 31

[82] Chu, Cheng Kang, Sherman SM Chow, Wen Guey Tzeng, Jianying Zhou e Robert H Deng: *Key-aggregate cryptosystem for scalable data sharing in cloud storage*. IEEE Transactions on Parallel and Distributed Systems, 25(2):468–477, 2014. 31

[83] Khan, Abdul Nasir, ML Mat Kiah, Mazhar Ali, Sajjad A Madani, Shahaboddin Shamshirband *et al.*: *Bss: block-based sharing scheme for secure data storage services in mobile cloud environment.* The Journal of Supercomputing, 70(2):946–976, 2014. 31

[84] Roman, Rodrigo, Jianying Zhou e Javier Lopez: *On the features and challenges of security and privacy in distributed internet of things.* Computer Networks, 57(10):2266–2279, 2013. 32

[85] Soldatos, John, Nikos Kefalakis, Manfred Hauswirth, Martin Serrano, Jean Paul Calbimonte, Mehdi Riahi, Karl Aberer, Prem Prakash Jayaraman, Arkady Zaslavsky, Ivana Podnar Žarko *et al.*: *Openiot: Open source internet-of-things in the cloud.* Em *Interoperability and open-source solutions for the internet of things*, páginas 13–25. Springer, 2015. 32

[86] LogMeIn, I: *Xively.* `http://www.xively.com`, 2015. 32

[87] Hunkeler, Urs, Hong Linh Truong e Andy Stanford-Clark: *Mqtt-s—a publish/subscribe protocol for wireless sensor networks.* Em *Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on*, páginas 791–798. IEEE, 2008. 32

[88] Eggert, Michael, Roger Häußling, Martin Henze, Lars Hermerschmidt, René Hummen, Daniel Kerpen, Antonio Navarro Pérez, Bernhard Rumpe, Dirk Thißen e Klaus Wehrle: *Sensorcloud: Towards the interdisciplinary development of a trustworthy platform for globally interconnected sensors and actuators.* Em *Trusted Cloud Computing*. 2014. 33

[89] Henze, Martin, Marcel Großfengels, Maik Koprowski e Klaus Wehrle: *Towards data handling requirements-aware cloud computing.* Em *IEEE International Conference on Cloud Computing Technology and Science*, 2013. 34

[90] Henze, Martin, René Hummen e Klaus Wehrle: *The cloud needs cross-layer data handling annotations.* Em *IEEE Security and Privacy Workshops (SPW)*, 2013. 34

[91] Hardt, Dick: *The oauth 2.0 authorization framework.* RFC 6749, 2012. 38

[92] Henze, Martin, René Hummen, Roman Matzutt e Klaus Wehrle: *A trust point-based security architecture for sensor data in the cloud.* Em *Trusted Cloud Computing*, páginas 77–106. Springer, 2014. 41

[93] Winter, Tim: *Rpl: Ipv6 routing protocol for low-power and lossy networks.* RFC 6550, 2012. 48

[94] Salomaa, Arto: *Public-key cryptography.* Springer Science & Business Media, 2013. 48

[95] Rivest, Ronald L, Adi Shamir e Leonard Adleman: *A method for obtaining digital signatures and public-key cryptosystems.* Communications of the ACM, 21(2):120–126, 1978. 49

[96] Hankerson, Darrel, Alfred J Menezes e Scott Vanstone: *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006. 49

[97] Gura, Nils, Arun Patel, Arvinderpal Wander, Hans Eberle e Sheueling Chang Shantz: *Comparing elliptic curve cryptography and rsa on 8-bit cpus*. Em *International Workshop on Cryptographic Hardware and Embedded Systems*, páginas 119–132, 2004. 49

[98] Hu, Wen, Peter Corke, Wen Chan Shih e Leslie Overs: *secfleck: A public key technology platform for wireless sensor networks*. Em *European Conference on Wireless Sensor Networks*, páginas 296–311, 2009. 49

[99] Sethi, Mohit, Jari Arkko e Ari Keränen: *End-to-end security for sleepy smart object networks*. Em *IEEE Conference on Local Computer Networks Workshops*, 2012. 49

[100] Hummen, René, Hossein Shafagh, Shahid Raza, Thiemo Voig e Klaus Wehrle: *Delegation-based authentication and authorization for the ip-based internet of things*. Em *Annual IEEE International Conference on Sensing, Communication, and Networking*, 2014. 49

[101] Deering, Stephen E: *Internet protocol, version 6 (ipv6) specification*. RFC 2460, 1998. 56

[102] Postel, Jon: *User datagram protocol (udp)*. RFC 768, 1980. 56

[103] Shelby, Zach, Klaus Hartke e Carsten Bormann: *The constrained application protocol (coap)*. RFC 7252, 2014. 56

[104] Instruments, Texas: *CC2420: Single-chip 2.4 ghz ieee 802.15.4 compliant and zigbee ready RF transceiver*. `http://www.ti.com/product/CC2420`, 2017. 57

[105] Fisher, Roy, Lehlogonolo Ledwaba, Gerhard Hancke e Carel Kruger: *Open hardware: A role to play in wireless sensor networks?* Sensors, 15(3):6818–6844, 2015. 57

[106] Pi, Raspberry: *Raspberry pi*, 2013. 57

[107] Pacheco, Luis Alberto B, Joao JC Gondim, Priscila A Solis Barreto e Eduardo Alchieri: *Evaluation of distributed denial of service threat in the internet of things*. Em *Network Computing and Applications (NCA), 2016 IEEE 15th International Symposium on*, páginas 89–92. IEEE, 2016. 69

[108] Pacheco, Luis, Eduardo Alchieri e Priscila Solis: *Architecture for privacy in cloud of things*. Em *Proceedings of the 19th International Conference on Enterprise Information Systems - Volume 2: ICEIS,*, páginas 487–494. INSTICC, SciTePress, 2017, ISBN 978-989-758-248-6. 69

[109] Pacheco, Luis Alberto B, Eduardo Alchieri e Priscila A Solis Barreto: *Enhancing and evaluating an architecture for privacy in the integration of internet of things and cloud computing*. Em *Network Computing and Applications (NCA), 2017 IEEE 16th International Symposium on*, páginas 1–8. IEEE, 2017. 69