CACIC: Controle de Acesso Confiável Usando Enclaves a Dados em Nuvem da Internet das Coisas

Guilherme A. Thomaz[1], **Matheus B. Guerra**[1], **Matteo Sammarco**[2] **e Miguel Elias M. Campista**[1]

[1]Grupo de Teleinformática e Automação (GTA)
Universidade Federal do Rio de Janeiro (UFRJ)
[2]AXA

{guiaraujo,barreira,miguel}@gta.ufrj.br,
{matteo.sammarco}@axa.com

Resumo. *Os sensores da Internet das Coisas geram um enorme volume de informações sensíveis processadas em nuvem, como geolocalização, consumo de energia e sinais biomédicos. Os sistemas de controle de acesso convencionais falham em garantir que esses dados sejam acessados e modificados apenas por usuários autorizados em um cenário em que a nuvem é maliciosa. Este artigo propõe uma arquitetura para transmissão, armazenamento, processamento e disponibilização de dados sensíveis de Internet das Coisas em nuvens computacionais, utilizando ambientes de execução confiáveis. Diferente das arquiteturas convencionais, o cliente personaliza para quem os seus dados são disponibilizados, pois as publicações e consultas aos dados são processadas em regiões isoladas na memória chamadas de enclaves. A implementação do servidor seguro processa cerca de quinhentas requisições por segundo e o atraso inserido pelo processamento em enclaves de memória é igual a dezessete milissegundos. Os resultados revelam que a proposta é ágil e escalável, ao mesmo tempo que garante segurança, até mesmo quando o sistema operacional ou o administrador da nuvem são comprometidos.*

1. Introdução

Os dispositivos de Internet das Coisas (*Internet of Things* - IoT) como eletrodomésticos, relógios e fechaduras coletam informações como padrões de consumo de energia, sinais biomédicos e gravações de imagem e voz. As empresas utilizam esses dados para oferecer serviços que prometem mais conforto, saúde e segurança ao cliente. A execução de tais serviços, porém, exige o armazenamento e o processamento de grandes volumes de dados, justificando a transferência de ambas as tarefas para ambientes em nuvem. A computação em nuvem é conhecida por oferecer vantagens como baixo custo ao permitir que várias máquinas virtuais sejam instanciadas em uma mesma máquina física. Além disso, as nuvens oferecem alto desempenho, já que as máquinas virtuais podem alocar de forma dinâmica e eficiente recursos físicos em função da demanda [Othman and El-Mousa 2020].

Por um lado, as vantagens trazidas na execução de serviços da Internet das Coisas em nuvem facilitam o dia-a-dia dos clientes. Por outro lado, o emprego das nuvens levanta

Este trabalho foi realizado com recursos do CNPq, CAPES, FAPERJ e FAPESP (2015/24494-8, 2018/23292-0, 2015/24485-9, 2014/50937-1).

preocupações quanto à segurança dos dados, uma vez que podem colocar a privacidade dos clientes em risco ao revelar informações sensíveis. Pesquisas já demonstraram que sistemas comerciais de Internet das Coisas são vulneráveis a ataques que permitem que o atacante controle os dispositivos inteligentes de uma casa, como as fechaduras, por exemplo [Fernandes et al. 2016]. Sendo assim, garantir privacidade em um cenário em que os dados são processados em nuvem é um grande desafio, pois o cliente perde o controle sobre o que será feito e quem terá acesso aos seus próprios dados. Apesar dos esforços com novas legislações de proteção de dados (Lei Geral de Proteção de Dados – LGPD), uma empresa que age de forma maliciosa pode ainda obter vantagem comercial ou utilizar dados privados sem permissão de acesso. Mesmo considerando as arquiteturas atuais que empregam criptografia e autenticação, esses esquemas ainda podem falhar caso a própria empresa que controla a máquina tenha acesso direto aos dados.

Este artigo propõe o CACIC, uma arquitetura inovadora de Internet das Coisas na qual o cliente personaliza o que será feito com dados sensíveis enviados para nuvem e quem terá acesso a cada informação. Para tanto, o sistema utiliza enclaves, que são regiões isoladas de memória que impedem o acesso a dados tanto para leitura quanto para escrita por parte de qualquer componente do computador, mesmo aqueles com privilégios mais elevados como o sistema operacional e hipervisores. Na literatura corrente, os enclaves já são utilizados para proteger a coleta e agregação de dados, gerenciar chaves criptográficas e proteger bancos de dados [Yang et al. 2021, Li et al. 2019, Valadares et al. 2018, Priebe et al. 2018]. Nessas propostas, porém, o cliente não é capaz de decidir quais entidades possuem acesso aos seus dados e quais são impedidas de acessá-los. Nesse sentido, o CACIC contribui com o estado da arte ao garantir que as políticas de controle de acesso aos dados dos clientes são cumpridas pelo servidor em nuvem, mesmo quando ele age maliciosamente. Ademais, o CACIC se destaca de outras arquiteturas por garantir segurança em nuvem sem exigir um banco de dados, um tipo de sensor ou um protocolo de comunicação específico, facilitando a adoção em infraestruturas de Internet das Coisas já amplamente adotadas no mercado.

A implementação do servidor utiliza a tecnologia *Software Guard Extensions* (SGX) para realizar o processamento em enclaves, armazenar dados com integridade e confidencialidade e provar para o cliente que o sistema é seguro. Apenas os módulos que realizam processamentos críticos à segurança utilizam enclaves de memória, otimizando o desempenho. Os resultados da avaliação de desempenho revelam que a plataforma provê segurança no processamento dos dados em enclaves sem inserir um atraso grande no processamento das requisições. O sistema também é escalável, pois processa centenas de requisições por segundo, confirmando sua viabilidade em infraestruturas de Internet das Coisas.

O restante do artigo está organizado da seguinte forma. A Seção 2 descreve o cenário de Internet das Coisas em nuvem e o modelo de atacante. A Seção 3 apresenta a tecnologia de computação confiável por enclaves, que possibilita o processamento em regiões isoladas na memória. A Seção 4 detalha a arquitetura proposta, especificando os módulos, os protocolos e os formatos de mensagens desenvolvidos. A Seção 5 fornece os resultados da avaliação de desempenho da implementação. A Seção 6 apresenta o estado da arte e discute os trabalhos relacionados. Por fim, a Seção 7 conclui o artigo e apresenta direções para trabalhos futuros.

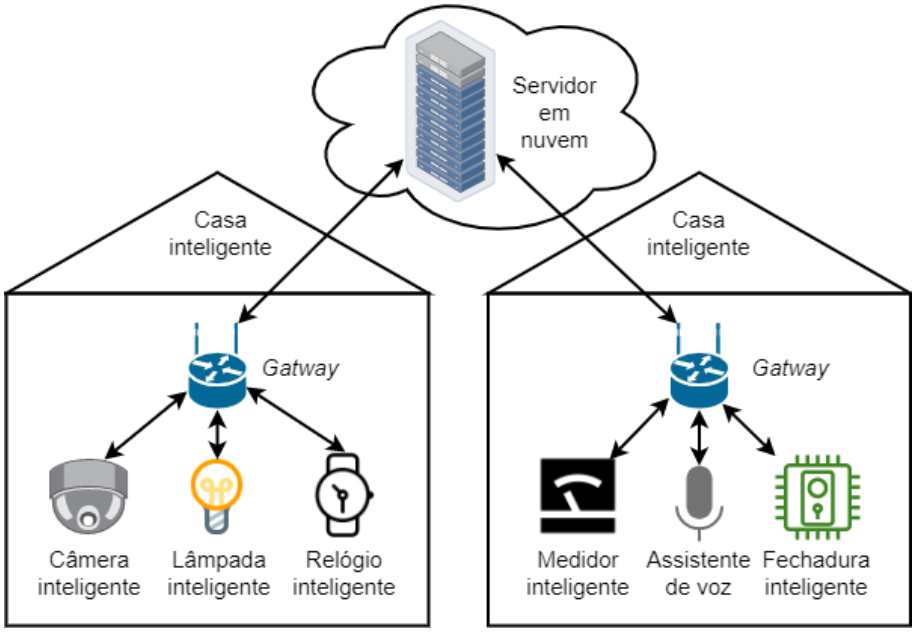

Figura 1. Cenário típico de casas inteligentes composto de dispositivos com restrição de recursos computacionais. Tais dispositivos se interconectam à nuvem computacional através de *gateways* locais. O acesso ao servidor em nuvem pode inserir risco à privacidade dos usuários caso dados sensíveis sejam usados indevidamente.

2. Cenário e Modelo de Atacante

A Figura 1 apresenta os componentes da infraestrutura de Internet das Coisas utilizada como referência neste trabalho. Considera-se um caso de uso de casas inteligentes (*smart homes*) em que sensores coletam dados de diferentes aplicações, como medidores elétricos, lâmpadas e fechaduras inteligentes [Ayoade et al. 2019]. Devido ao baixo poder computacional destes sensores, os pontos de acesso (*gateways*) pré-processam os dados coletados, formando a computação em borda (*edge computing*). Por fim, os dados são encaminhados para servidores hospedados em máquinas virtuais na nuvem, que aplicam processamentos mais custosos em um enorme volume de dados, como agregação e aprendizado de máquina [Souza et al. 2020]. A nuvem disponibiliza alguns dados para auxiliar na tomada de decisão de entidades, como empresas de energia elétrica interessadas em utilizar o consumo de energia para planejar em tempo real a distribuição de carga, por exemplo.

Uma vez que os dados são enviados para a nuvem, o usuário perde o controle do que será feito com suas informações caso os servidores não sejam confiáveis. Eibl *et al.* demonstraram que dados de consumo de energia elétrica utilizados por algoritmos em nuvem revelam informações privadas, como a quantidade de pessoas utilizando uma instalação em determinado momento [Eibl and Engel 2014]. Além disso, um agente malicioso pode se passar por um usuário autêntico e fabricar um dado que permita o acesso a uma fechadura eletrônica de uma casa inteligente. Estes exemplos ilustram que os servidores em nuvem devem atender aos seguintes requisitos de segurança:

1. **Confidencialidade**: os dados sensíveis não podem ser revelados para qualquer usuário.

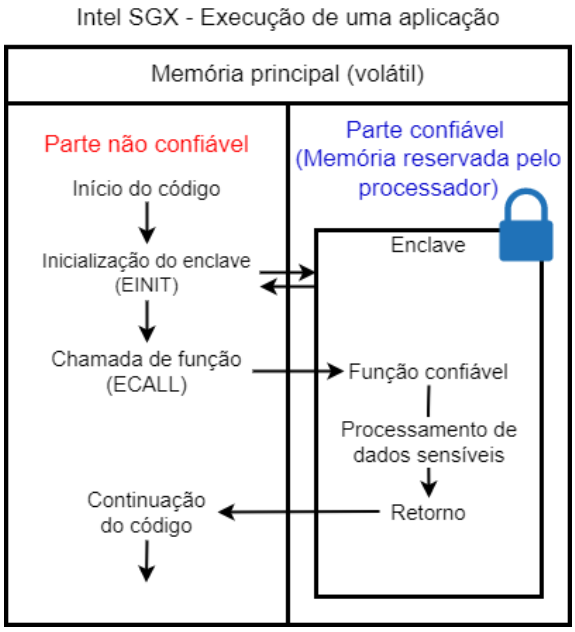

Figura 2. A aplicação inicializa um enclave de memória (instrução `EINIT`) e chama uma função confiável (instrução `ECALL`) que processa dados confidenciais em um ambiente isolado, com acesso protegido pelo processador.

2. **Integridade**: os dados não podem ser alterados por qualquer usuário.
3. **Autenticação**: os usuários devem se identificar para enviar e receber dados.
4. **Controle de Acesso**: os usuários devem determinar quem tem direito de acesso aos seus dados e o que pode ser feito com eles.

2.1. Modelo de atacante

Os sistemas de detecção de intrusão demonstram eficiência para impedir ataques de invasão de rede que possam comprometer a disponibilidade do sistema e a segurança dos dados [Guimarães et al. 2021]. Entretanto, garantir os requisitos de segurança quando a própria nuvem possui interesse em alterar ou vazar dados sensíveis ainda permanece um desafio. O modelo de atacante considerado para desenvolver a arquitetura apresentada neste artigo considera um agente malicioso que obtém acesso privilegiado ao servidor. O atacante pode ser um administrador de rede ou até mesmo um sistema operacional comprometido, capaz de ler e escrever qualquer arquivo e executar qualquer aplicação. Também é capaz de interceptar, ler, bloquear e fabricar pacotes de rede. O atacante não possui acesso à residência do cliente e, portanto, não possui acesso aos sensores e ao *gateway* da rede local [Valadares et al. 2018]. Também não é capaz de quebrar algoritmos criptográficos e nem inferir informações no interior do circuito integrado do processador.

3. Computação Confiável com Enclaves

Para lidar com a crescente ameaça de ataques virtuais e vazamento de dados, o Grupo de Computação Confiável (*Trusted Computing Group* - TCG) propôs o desenvolvimento de plataformas que fornecem segurança usando recursos de *hardware*. Dentre essas plataformas, a tecnologia de Extensões de Proteção de Software (*Software Guard Extensions* - SGX) se destaca por proteger dados e *softwares* mesmo se componentes com

alto nível de privilégio forem comprometidos ou forem contaminados por *malwares*. Essa tecnologia é adequada para o modelo de atacante apresentado, pois garante confidencialidade e integridade quando a BIOS, o sistema operacional, o hipervisor ou o administrador da máquina são maliciosos. O SGX usa instruções especiais dos processadores Intel para criar Ambientes de Execução Confiáveis (*Trusted Execution Environments* - TEEs), que processam dados e códigos em regiões isoladas na memória [Costan and Devadas 2016].

Os enclaves são regiões isoladas e encriptadas de memória, que só podem ser acessadas pelo proprietário do enclave utilizando instruções confiáveis. Essa divisão é feita para utilizar funcionalidades como chamadas de sistema (`syscall`) e acesso ao I/O, que só são disponíveis fora de enclaves. A Figura 2 mostra que apenas os dados críticos de uma aplicação são executados dentro da parte confiável, para reduzir a quantidade de informações armazenadas no enclave [Will et al. 2017]. A arquitetura, detalhada na próxima seção, utiliza enclaves para processar as consultas e as publicações de dados dos clientes em regiões de memória inacessíveis até mesmo para o administrador do sistema. Para armazenar os dados sensíveis de forma persistente, a infraestrutura do SGX oferece o processo de selagem. Neste processo, o conteúdo dos enclaves é criptografado por chaves que nunca saem da CPU e o resultado é transferido para o disco, de forma que só possa ser decriptado na mesma plataforma [Anati et al. 2013].

A execução de um procedimento de atestação remota é indispensável para a arquitetura proposta, pois o servidor que processa os dados em enclave precisa provar sua segurança e autenticidade para o cliente. Para isso, a Intel provisiona a cada processador uma chave de grupo chamada identificação de privacidade aprimorada (*Enhanced Privacy ID* - EPID), que só pode ser verificada ou revogada por uma entidade confiável chamada serviço de atestação da Intel (*Intel Attestation Service* - IAS). Na atestação, a CPU mede o *hash* do código de um enclave e envia um relatório assinado com a chave EPID (*quote*) para a parte interessada em verificar a segurança, que o encaminha para o servidor de atestação verificar a assinatura [Johnson et al. 2016]. A Intel oferece a possibilidade de *data center* implementarem seu próprio sistema de verificação de atestação, utilizando certificados provisionados pela Intel ao invés da chave EPID [Scarlata et al. 2018]. O protocolo de atestação se baseia em troca de chaves *Diffie-Hellman* de curva elíptica entre cliente e enclave para transmitir o relatório em um canal encriptado. Se a atestação for bem sucedida, o cliente envia os segredos por este canal encriptado, de modo que apenas o enclave consiga decriptá-los. A próxima seção detalha como essas prerrogativas de segurança do Intel SGX serão usadas no desenvolvimento do CACIC.

4. Arquitetura do CACIC

A Figura 3 ilustra a arquitetura, composta por clientes que enviam ou recebem dados de um servidor confiável em nuvem por meio de seus pontos de acesso. A comunicação utiliza o protocolo de transferência de hipertexto seguro (*Hyper Text Transfer Protocol Secure* - HTTPS) para garantir a autenticidade do servidor e do cliente e garantir a confidencialidade por meio de criptografia [Wang et al. 2017]. Entretanto, esta medida é insuficiente para atender os requisitos de segurança apresentados na Seção 2, pois o atacante obtém os dados uma vez que compromete a nuvem. Para tanto, o servidor possui um ambiente de execução confiável para realizar processamentos.

A Figura 4 detalha os procedimentos do sistema para publicar, processar e consul-

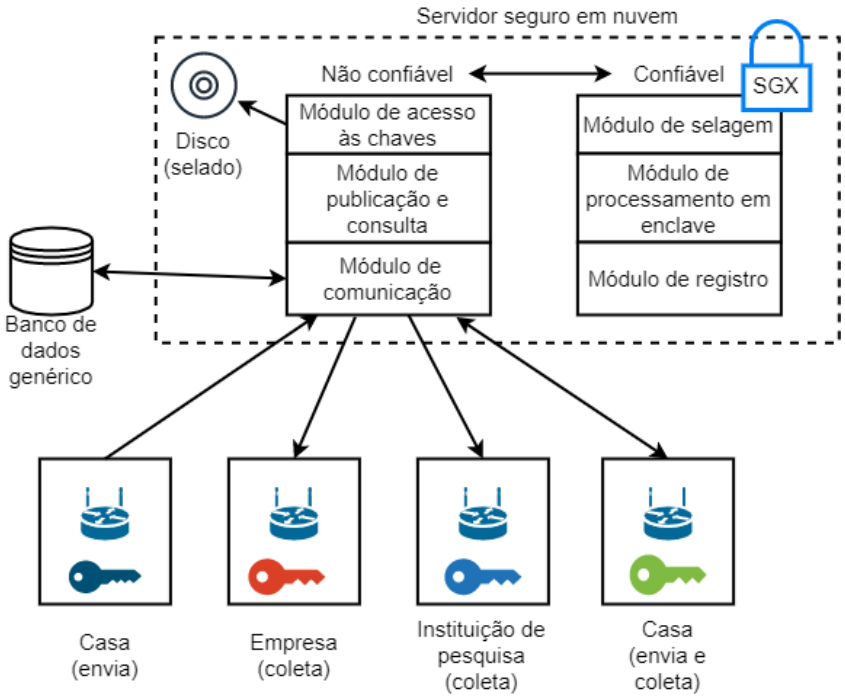

Figura 3. Os pontos de acesso dos clientes enviam e recebem dados encriptados com suas chaves de comunicação registradas. Os dados recebidos pela nuvem são decriptados, processados e encriptados em enclaves antes de serem publicados.

tar dados de forma segura. Inicialmente, o ponto de acesso do cliente, deve se registrar na plataforma (1) para compartilhar sua chave de comunicação simétrica (`CC`) com o servidor, utilizada para encriptar dados enviados e decriptar dados recebidos. O procedimento de geração da chave de comunicação pelo ponto de acesso não afeta a segurança da proposta, pois o modelo de atacante assume que o cliente é confiável. No registro, o ponto de acesso do cliente deve atestar se o servidor é confiável (2), como foi apresentado na Seção 3 e, se a atestação for bem sucedida, o ponto de acessa envia a chave para o enclave. O servidor armazena a chave de comunicação selada em disco, associada à chave pública do cliente, de modo que um atacante interessado em roubar dados dos clientes não terá a chave utilizada para decriptá-los. Ademais, este protocolo garante que o servidor não falsifique os códigos da plataforma confiável, pois o cliente compara a medida criptográfica do enclave enviada pelo servidor na atestação com um valor esperado. Todas as mensagens encriptadas possuem um código de autenticação de mensagens (*Message Authentication Code* - MAC), que permite auditar se elas foram alteradas.

Na publicação, o ponto de acesso do cliente recebe dados dos sensores (3) e monta uma mensagem (`M[publicação]`, 4) com os seguintes campos:

```
M[publicação]=[pub|nonce|tipo|tamanho|CC(dado|perm)].
```

O símbolo `|` simboliza a concatenação dos *bytes*. A chave pública (`pub`) identifica o cliente para o servidor, que localiza a chave de comunicação (`CC`) selada em disco. O tipo do dado identifica o dispositivo de origem, para que a plataforma aplique os processamentos adequados para cada caso. As permissões de acesso (`perm`) do cliente definem para o servidor quem pode acessar os dados. Além disso, as permissões são encriptadas

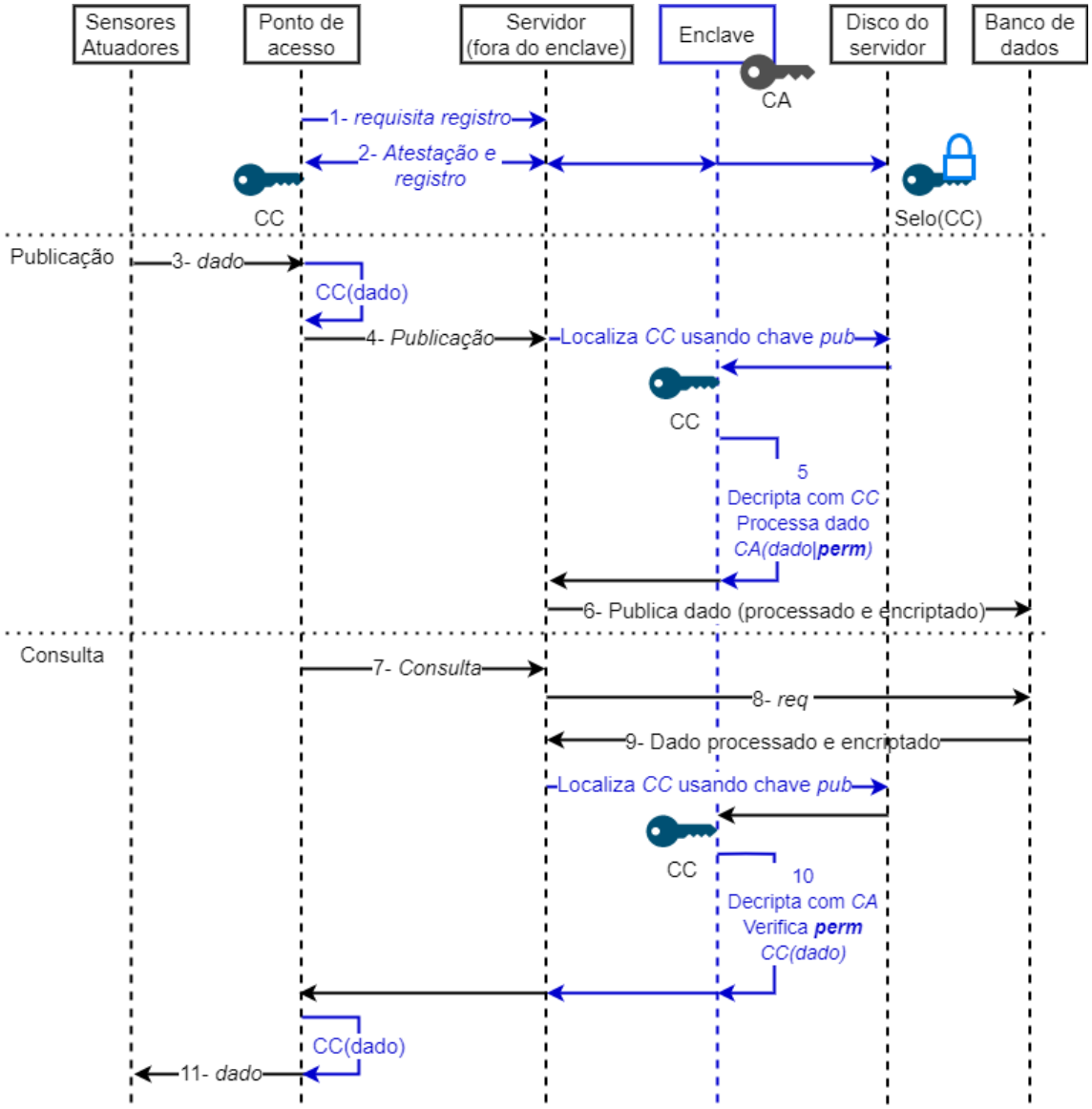

Figura 4. Os dados em trânsito são encriptados com a chave de comunicação (`CC`) e os dados armazenados são encriptados com a chave de armazenamento (`CA`). Apenas o enclave decripta e processa estes dados na nuvem. A cor azul destaca os mecanismos de segurança inseridos pelo CACIC. A personalização das permissões de acesso (`perm`) pelo cliente é uma inovação da proposta.

junto aos dados, por se tratarem de informações privadas. O `nonce` é enviado para evitar ataques de repetição (*replay*). Uma vantagem deste sistema está no fato de que ele não impõe nenhum requisito computacional ou formato de mensagem rígido para o sensor, pois é o ponto de acesso que estabelece a chave e encripta o dado, independente do formato da mensagem do sensor. A diversidade de dispositivos de Internet das Coisas em termos de protocolos e especificações de desempenho justifica esta escolha de projeto, tornando a proposta flexível e agnóstica ao dispositivo [HaddadPajouh et al. 2021].

Uma vez recebida a mensagem de publicação, o enclave do servidor recupera a chave de comunicação, decripta o dado e aplica um processamento, dependendo do tipo de dado (5). O desenvolvedor na nuvem deve programar as aplicações que processam os

dados dentro do enclave, para garantir a confidencialidade e a integridade em tempo de execução. Um exemplo de aplicação comum em redes elétricas inteligentes (*smart grids*) é a agregação. O enclave acessa os dados de consumo de energia elétrica de uma região armazenados no banco de dados, os decripta e calcula o consumo de energia total. Dessa forma, uma instituição utiliza esta informação para tomar decisões referentes ao planejamento de distribuição de energia, por exemplo, sem obter acesso aos dados individuais dos clientes [Silva et al. 2018]. Outras propostas utilizam criptografia homomórfica para este tipo de processamento. Porém, os enclaves se destacam por inserir um atraso de processamento muito menor, bem como possibilitar operações arbitrariamente complexas sobre os dados [Silva et al. 2018]. Antes da publicação no banco de dados, o dado é encriptado com uma chave de armazenamento (`CA`) acessível apenas pelo enclave, impedindo que qualquer outra entidade além do enclave recupere o dado em claro, mesmo que um atacante obtenha acesso de alto privilégio ao servidor.

Na consulta, o ponto de acesso do cliente envia uma mensagem (`M[consulta]`, 7) com o seguinte formato:

```
M[consulta]=[pub|nonce|tamanho|req],
```

onde `pub` é a chave pública que identifica o cliente e `req` é o comando utilizado para localizar e requisitar o dado no banco de dados. A arquitetura é agnóstica ao banco de dados, de modo que a mensagem de consulta possui um formato flexível de acordo com a aplicação. A requisição é encaminhada para o banco de dados (8) e a resposta é o dado encriptado (9). Em seguida, o enclave do servidor recupera a chave de comunicação (`CC`) do cliente, decripta o dado com a chave de armazenamento (`CA`) e verifica se as permissões de acesso (`perm`, armazenadas junto ao dado) autorizam o acesso ao dado por parte do cliente interessado (10). Em caso positivo, o enclave encripta o resultado com a chave `CC` e envia para o ponto de acesso. Por fim, o ponto de acesso decripta o dado recebido utilizando a chave `CC`, disponibilizando-o para o dispositivo interessado (11).

Um diferencial da arquitetura do CACIC está na possibilidade do cliente configurar no ponto de acesso as permissões de acesso. O ponto de acesso acrescenta o campo `perm` à mensagem de publicação (`M[publicação]`) para que o enclave impeça ou permita o acesso quando um cliente requisitar este dado. Um usuário pode impedir que os sinais biomédicos de seu relógio inteligente sejam disponibilizados para a fabricante do relógio, mas permitir que sejam acessados por uma instituição de pesquisa biomédica, por exemplo. Dessa forma, o cliente possui o controle completo sobre quem acessa seus dados, pois as requisições de acesso são processadas de forma confiável, mesmo que o servidor seja controlado por um agente malicioso. Isso só é possível graças à confiança que o cliente estabeleceu no ambiente de execução confiável do servidor.

5. Implementação e Resultados

Os objetivos da avaliação de desempenho são: i) verificar se a arquitetura é escalável e ii) avaliar o atraso de processamento inserido pelo ambiente de execução confiável. As propostas em Internet das Coisas em nuvem devem atender a um grande número de clientes ao mesmo tempo, tendo em vista a quantidade crescente de dispositivos. Esse desafio justifica a avaliação do desempenho em escala das propostas de segurança [Hou et al. 2016]. O servidor confiável foi implementado em um computador

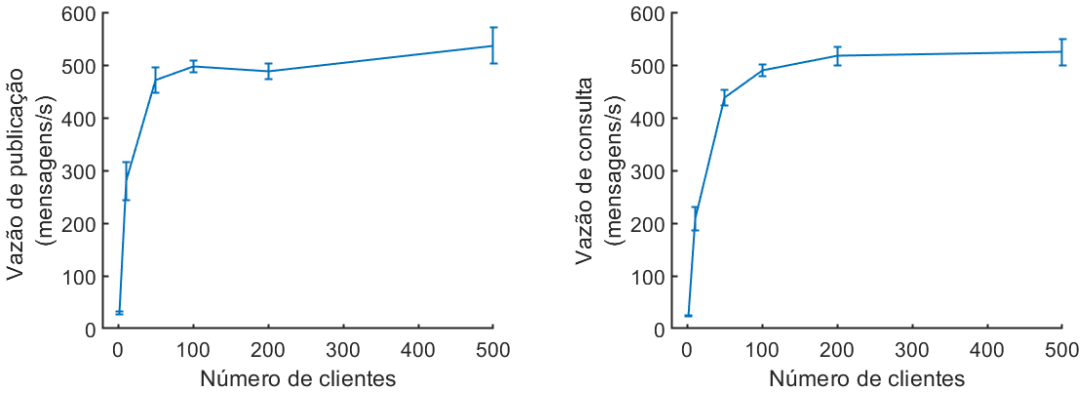

Figura 5. O sistema processa até 537 publicações por segundo (a) e 525 consultas por segundo (b), revelando que a proposta é escalável, pois processa paralelamente requisições de um grande número de clientes.

com Intel i9-10900 CPU 2.80 GHz com 32 GB RAM e 20 *threads*. Esta máquina simula requisições de publicação e consulta à dados fictícios no formato *Ultralight* 2.0, que é um padrão adotado pela plataforma FIWARE, para desenvolvimento de aplicações inteligentes, utilizada em outros trabalhos [Valadares et al. 2018, Araujo et al. 2019].[1] Um kit de desenvolvimento de software (*Software Development Kit* - SDK) para uso das instruções confiáveis do Intel SGX em linguagem C++ foi utilizado[2]. Os experimentos avaliam a quantidade de requisições que a plataforma processa por segundo, bem como o tempo dos procedimentos de publicação, consulta e registro. Os resultados possuem um intervalo de confiança de 95%.

O primeiro experimento avalia a quantidade de publicações de dados que o servidor é capaz de processar por segundo, que inclui as etapas de 3 a 6 indicadas na Figura 4. O número de clientes simultâneos que enviam uma requisição varia entre 1, 10, 50, 100, 200 e 500. Em cada caso, mede-se o tempo que o servidor leva até publicar todos os dados recebidos. Nenhum outro processamento nos dados foi realizado, pois o objetivo deste experimento é verificar isoladamente o desempenho da publicação. A Figura 5(a) mostra que a vazão de publicação aumenta rapidamente com o número de clientes, pois o sistema processa as requisições em paralelo. Para dez clientes, a vazão é aproximadamente dez vezes maior que a vazão no caso de um cliente, apontando um crescimento aproximadamente linear. A medida que o número de clientes aumenta, a inclinação do gráfico diminui, até que a vazão se estabiliza em um valor de máximo, que representa o limite da capacidade de processamento do servidor. As mensagens passam a demorar mais tempo para serem processadas quando o servidor atinge sua capacidade de paralelismo, pois são inseridas em uma fila. A limitação no desempenho de aplicações com SGX se deve principalmente aos procedimentos de inicialização, entrada e saída de enclaves, como demonstrado por outros trabalhos [Gjerdrum et al. 2017]. Ainda assim, a proposta é escalável, atingindo um pico de 537 publicações por segundo. O segundo experimento avalia a vazão de consulta de dados que inclui as etapas 7 a 11 da Figura 4, no mesmo cenário proposto para o primeiro experimento. A Figura 5(b) revela um comportamento

[1]https://fiware-iotagent-ul.readthedocs.io/en/latest/

[2]Repositório do projeto disponível em: https://github.com/GTA-UFRJ-team/TACIoT

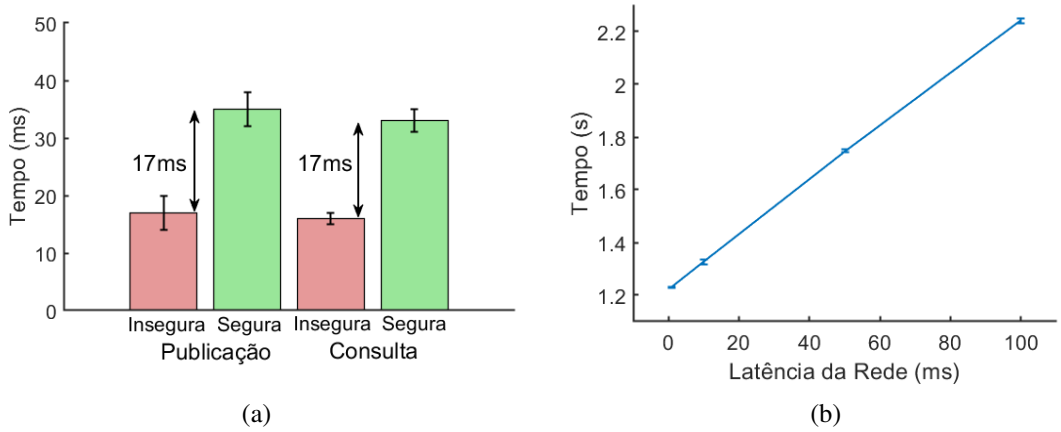

Figura 6. O ambiente confiável insere um atraso de 17ms no tempo de processamento (a). Um aumento de 0,1s na latência da rede se reflete em um aumento de 1s no tempo de registro (b).

similar ao primeiro experimento, atingindo um pico de 525 requisições por segundo. O sistema apresenta um desempenho otimizado, pois o servidor realiza apenas uma entrada e saída de enclave por publicação ou consulta, como demonstrado na Seção 4.

O terceiro experimento avalia o atraso que o ambiente de execução confiável insere no tempo de processamento das requisições de publicação e consulta. Em um primeiro cenário, mede-se o tempo que o servidor demora para publicar ou consultar um dado sem realizar os processamentos em enclave. Em um segundo cenário, mede-se o tempo de publicação e consulta utilizando o ambiente de execução confiável. Por fim, a diferença entre as duas medidas é o tempo que o esquema de segurança proposto acrescenta no processamento das requisições. A Figura 6(a) apresenta os tempos de processamento das requisições no primeiro cenário (sem computação confiável) na cor vermelha e os tempos de processamentos no segundo cenário (com computação confiável) na cor verde. O gráfico revela que os mecanismos de segurança inserem apenas 17 milissegundos no tempo de processamento das requisições. Esse atraso se deve ao procedimento de inicialização do enclave na memória. Uma vez que ele foi inicializado, o tempo para processar as requisições dentro do enclave foi desprezível. Esse atraso não degrada o desempenho das aplicações de Internet das Coisas em nuvem, pois o tempo para processar dados de dispositivos comerciais pode variar desde dezenas de milissegundos até alguns segundos, como mostrado por outros trabalhos [Ayoade et al. 2019]. Esses resultados revelam que os mecanismos de segurança baseados em ambientes de execução confiáveis para Internet das Coisas não oferecem uma sobrecarga perceptível para o usuário final interessado em consultar e publicar dados em nuvem.

O quarto experimento avalia o tempo de registro de um cliente. Mede-se o tempo desde o cliente requisitar o registro na plataforma até a chave de comunicação ser armazenada em disco no servidor. Devido à troca de mensagens para atestar o servidor, o tempo do procedimento depende muito da latência de comunicação entre cliente e servidor. Para verificar esta influência, o programa do cliente foi executado na mesma máquina e a latência foi simulada através de um comando de *delay*. A Figura 6(b) mostra uma relação linear entre a latência da rede e o tempo de registro. Isto se justifica pois a relação entre o tempo de registro (T_{reg}) e a latência (L) é modelada com a Equação 1, onde

$m = 10$ é o número de mensagens, e T_{proc} é o tempo de processamento local, tanto no cliente quanto no servidor. O tempo de registro é aceitável, considerando que este procedimento será feito uma única vez para o servidor se atestar e cadastrar o ponto de acesso. Ademais, este tempo se deve ao procedimento padrão de atestação da Intel.

$$T_{reg} = mL + T_{proc} \tag{1}$$

6. Trabalhos relacionados

A computação confiável é frequentemente apresentada na literatura como uma solução para proteger os dados sensíveis gerados pelos sensores inteligentes quando os usuários que controlam os sistemas podem ser maliciosos. Yang *et al.* implementam uma plataforma de computação confiável personalizada em sensores industriais para garantir a confiança dos dados publicados em um cenário em que os dispositivos podem ser comprometidos [Yang et al. 2021]. A proposta se serve de corrente de blocos (*blockchain*) para manter um registro transparente, auditável e distribuído dos dados, bem como executar processamentos mais complexos sobre estes dados com contratos inteligentes [Thomaz et al. 2021]. O artigo não lida com a segurança de dados enviados para a nuvem e os dados publicados no livro-razão distribuído são públicos, fazendo com que o sistema não atenda ao requisito de confidencialidade.

As arquiteturas modernas de Internet das Coisas se servem do poder computacional e de armazenamento da nuvem para armazenar e processar os dados, exigindo que as propostas em segurança ofereçam confidencialidade e integridade nestes ambientes sem confiança [Valadares et al. 2021]. Priebe *et al.* propõem a execução de um motor de banco de dados dentro de enclaves de memória utilizando o Intel SGX [Priebe et al. 2018]. O sistema é promissor para a arquitetura de Internet das Coisas, na qual sensores publicam e consultam dados em nuvem, pois as requisições são processadas em um ambiente isolado do administrador e do sistema operacional. A proposta se concentra no armazenamento dos dados e não é uma solução completa para Internet das Coisas. Em um cenário em que a nuvem é maliciosa, o atacante obtém acesso aos dados processados na memória principal antes de serem publicados.

Li *et al.* realizam a agregação de dados de medidores elétricos inteligentes, tarifação dinâmica e previsão de consumo dentro de enclaves de memória [Li et al. 2019]. Esses processamentos exigem o acesso a dados de diversos clientes e são proibitivos para dispositivos com baixo poder computacional, obrigando o cliente a entregar seus dados para um servidor remoto. A arquitetura emprega o Intel SGX no ponto de acesso do cliente e no servidor remoto. Ademais, ela se concentra na publicação segura dos dados para a empresa de energia elétrica processá-los remotamente e não implementa um sistema de controle de acesso para clientes autorizados consultarem os dados armazenados. Os ambientes de execução confiáveis não são as únicas ferramentas para garantir segurança enquanto os dados são processados. Silva *et al.* comparam o tempo de agregação de medidas de consumo de energia elétrica utilizando computação confiável com Intel SGX e computação homomórfica [Silva et al. 2018]. Os dois mecanismos de agregação garantem privacidade em tempo de processamento. Porém, a computação confiável apresentou um desempenho dez mil vezes melhor, confirmando que os ambientes de execução confiáveis são soluções que, além de seguras, não degradam o desempenho.

Valadares *et al.* expandem a arquitetura da plataforma de aplicações inteligentes FIWARE para fornecer dados sensíveis de sensores para usuários autenticados [Valadares et al. 2018]. Os autores se servem de enclaves em nuvem para armazenar as chaves de encriptação dos dados gerados por produtores e decriptação dos dados consultados por consumidores. A arquitetura não processa os dados em nuvem, exigindo que o consumidor possua um processador com suporte ao SGX e se ateste para receber a chave de decriptação dos dados e processá-los. A plataforma em nuvem implementa um servidor de publicação-inscrição (*publish-subscribe*), responsável apenas por disseminar os dados para os consumidores autenticados e gerenciar as chaves. A plataforma não permite que o produtor personalize que é feito com seus dados. Ayoade *et al.* propõem computação confiável para processar os dados de Internet das Coisas de diversas empresas em um ambiente compartilhado em nuvem (*middleware*). O artigo destaca que, apesar do alto custo-benefício dos serviços em nuvem, vulnerabilidades nesses ambientes permitem com que uma empresa obtenha acesso a segredos industriais e dados sensíveis de clientes. Os autores isolam o processamento dos dados em enclaves e disponibilizam os dados de um dispositivo apenas para a empresa que o fabricou. O sistema utiliza enclaves no ponto de acesso, obrigando o cliente a se atestar a cada publicação de dado, além de realizar saídas e entradas em enclaves com mais frequência. Dessa forma, apesar de assumir que o *gateway* pode ser malicioso, a plataforma oferece uma sobrecarga no desempenho maior que a apresentada pelo CACIC.

Ao contrário dos artigos citados anteriormente, este artigo propõe um sistema ágil para o usuário determinar como seus dados de Internet das Coisas serão utilizados em nuvem, mesmo quando entidades com alto nível de privilégio são maliciosas. A arquitetura é flexível e atende ao cenário diverso de Internet das Coisas, pois não exige um banco de dados específico, tendo em vista que as requisições são sempre pré-processadas em enclaves. O sistema também é agnóstico ao sensor, pois não especifica formatos predefinidos de mensagem, funcionalidades específicas ou requisitos mínimos de desempenho dos dispositivos. Outro diferencial é que o sistema aplica processamentos arbitrariamente complexos sobre os dados, o que não acontece em propostas baseadas em criptografia homomórfica, por exemplo.

7. Conclusões

Este artigo propôs uma nova arquitetura para proteger o acesso aos dados de Internet das Coisas transmitidos, processados e armazenados em nuvem. O artigo considerou um modelo de atacante com controle quase total sobre o sistema, como um administrador na nuvem interessado em obter alguma vantagem financeira, por exemplo. Nesse cenário, os esquemas convencionais de criptografia e detecção de intrusão falham. A arquitetura detalhada no artigo utiliza ambientes de execução confiáveis para solucionar esses problemas e atender aos rígidos requisitos de confidencialidade, integridade, autenticação e controle de acesso, mesmo quando o atacante na nuvem possui um alto nível de privilégio. O sistema é flexível, pois não depende dos dispositivos e das arquiteturas específicas utilizadas pelas plataformas de Internet das Coisas empresariais, o que contribui para sua implementação comercial. A avaliação de desempenho do sistema demonstrou que ele é ágil, pois os mecanismos de segurança inserem um tempo pequeno no processamento das requisições de publicação e acesso. Ademais, o sistema processa mais de quinhentas requisições por segundo, demonstrando que, além de segura, a proposta é escalável.

Como trabalhos futuros, planeja-se propor uma extensão da arquitetura para proteger os dados nos sensores e no ponto de acesso. Também é prevista a implementação de um sistema para que os usuários da plataforma publiquem seus próprios códigos de processamento de dados a serem protegidos no enclave em nuvem. Pretende-se também avaliar o desempenho dos enclaves para processamentos mais complexos, como aprendizado de máquina, por exemplo. Outra direção futura de pesquisa bastante promissora é a implementação de arquiteturas de processamento e armazenamento distribuído em enclaves.

Referências

Anati, I., Gueron, S., Johnson, S., and Scarlata, V. (2013). Innovative technology for cpu based attestation and sealing. In *Proceedings of the 2nd international workshop on hardware and architectural support for security and privacy*, volume 13. ACM New York, NY, USA.

Araujo, V., Mitra, K., Saguna, S., and Åhlund, C. (2019). Performance evaluation of fiware: A cloud-based iot platform for smart cities. *Journal of Parallel and Distributed Computing*, 132:250–261.

Ayoade, G., El-Ghamry, A., Karande, V., Khan, L., Alrahmawy, M., and Rashad, M. Z. (2019). Secure data processing for iot middleware systems. *The Journal of Supercomputing*, 75(8):4684–4709.

Costan, V. and Devadas, S. (2016). Intel sgx explained. *Cryptology ePrint Archive*.

Eibl, G. and Engel, D. (2014). Influence of data granularity on nonintrusive appliance load monitoring. In *Proceedings of the 2nd ACM workshop on Information hiding and multimedia security*, pages 147–151.

Fernandes, E., Jung, J., and Prakash, A. (2016). Security analysis of emerging smart home applications. In *2016 IEEE symposium on security and privacy (SP)*, pages 636–654. IEEE.

Gjerdrum, A. T., Pettersen, R., Johansen, H. D., and Johansen, D. (2017). Performance of trusted computing in cloud infrastructures with intel SGX. In *CLOSER*, pages 668–675.

Guimarães, L. C., Rebello, G. A. F., Camilo, G. F., de Souza, L. A. C., and Duarte, O. C. (2021). A threat monitoring system for intelligent data analytics of network traffic. *Annals of Telecommunications*, pages 1–16.

HaddadPajouh, H., Dehghantanha, A., Parizi, R. M., Aledhari, M., and Karimipour, H. (2021). A survey on internet of things security: Requirements, challenges, and solutions. *Internet of Things*, 14:100129.

Hou, L., Zhao, S., Xiong, X., Zheng, K., Chatzimisios, P., Hossain, M. S., and Xiang, W. (2016). Internet of things cloud: Architecture and implementation. *IEEE Communications Magazine*, 54(12):32–39.

Johnson, S., Scarlata, V., Rozas, C., Brickell, E., and Mckeen, F. (2016). Intel software guard extensions: EPID provisioning and attestation services. *White Paper*, 1(1-10):119.

Li, S., Xue, K., Wei, D. S., Yue, H., Yu, N., and Hong, P. (2019). Secgrid: A secure and efficient sgx-enabled smart grid system with rich functionalities. *IEEE Transactions on Information Forensics and Security*, 15:1318–1330.

Othman, M. M. and El-Mousa, A. (2020). Internet of things amp; cloud computing internet of things as a service approach. In *2020 11th International Conference on Information and Communication Systems (ICICS)*, pages 318–323.

Priebe, C., Vaswani, K., and Costa, M. (2018). Enclavedb: A secure database using sgx. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 264–278. IEEE.

Scarlata, V., Johnson, S., Beaney, J., and Zmijewski, P. (2018). Supporting third party attestation for intel sgx with intel data center attestation primitives. *White paper*.

Silva, L. V., Barbosa, P., Marinho, R., and Brito, A. (2018). Security and privacy aware data aggregation on cloud computing. *Journal of Internet Services and Applications*, 9(1):1–13.

Souza, L. A. C., Antonio F. Rebello, G., Camilo, G. F., Guimarães, L. C. B., and Duarte, O. C. M. B. (2020). DFedForest: Decentralized Federated Forest. In *2020 IEEE Blockchain*, pages 90–97.

Thomaz, G. A., Camilo, G. F., de Souza, L. A. C., and Duarte, O. C. M. (2021). Uma análise comparativa da arquitetura e desempenho de plataformas de corrente de blocos permissionadas para contratos inteligentes. In *Anais do IV Workshop em Blockchain: Teoria, Tecnologias e Aplicações*, pages 114–127. SBC.

Valadares, D. C. G., da Silva, M. S. L., Brito, A. E. M., and Salvador, E. M. (2018). Achieving data dissemination with security using fiware and intel software guard extensions (sgx). In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–7. IEEE.

Valadares, D. C. G., Will, N. C., Spohn, M. A., de Souza Santos, D. F., Perkusich, A., and Gorgonio, K. C. (2021). Trusted execution environments for cloud/fog-based internet of things applications. In *CLOSER*, pages 111–121.

Wang, J., Hong, Z., Zhang, Y., and Jin, Y. (2017). Enabling security-enhanced attestation with intel sgx for remote terminal and iot. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(1):88–96.

Will, N. C., Condé, R. C., and Maziero, C. A. (2017). Mecanismos de segurança baseados em hardware: uma introdução à arquitetura intel sgx. *Minicursos do XVII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais. Brasilia, DF, BR: Sociedade Brasileira de Computação*, pages 49–98.

Yang, Q., Wang, H., Wu, X., Wang, T., Zhang, S., and Liu, N. (2021). Secure blockchain platform for industrial iot with trusted computing hardware. *arXiv preprint arXiv:2110.15161*.