



UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS CRATEÚS
CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ALFREDO HENRIQUE SAMPAIO RODRIGUES

ANÁLISE COMPARATIVA DA UTILIZAÇÃO DE TÉCNICAS DE SEGURANÇA NO
***OFFLOADING* COMPUTACIONAL**

CRATEÚS - CEARÁ

2022

ALFREDO HENRIQUE SAMPAIO RODRIGUES

**ANÁLISE COMPARATIVA DA UTILIZAÇÃO DE TÉCNICAS DE SEGURANÇA NO
OFFLOADING COMPUTACIONAL**

Trabalho de Conclusão de Curso apresentado à Universidade Federal do Ceará, Campus de Crateús, como requisito parcial à obtenção do título de bacharel em Ciência da Computação.

Orientador: Prof. Me. Francisco Anderson de Almada Gomes.

Coorientador: Prof. Me. Filipe Fernandes dos Santos Brasil de Matos.

CRATEÚS

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

R611a Rodrigues, Alfredo Henrique Sampaio Rodrigues.
Análise Comparativa da Utilização de Técnicas de Segurança no Offloading Computacional / Alfredo Henrique Sampaio Rodrigues Rodrigues. – 2022.
86 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Crateús, Curso de Ciência da Computação, Crateús, 2022.

Orientação: Prof. Me. Francisco Anderson de Almada Gomes.

Coorientação: Prof. Me. Filipe Fernandes dos Santos Brasil de Matos.

1. Dispositivos Móveis. 2. Privacidade de Dados. 3. Algoritmos de Criptografia. 4. Offloading. 5. Computação em Nuvem Móvel. I. Título.

CDD 004

ALFREDO HENRIQUE SAMPAIO RODRIGUES

**ANÁLISE COMPARATIVA DA UTILIZAÇÃO DE TÉCNICAS DE SEGURANÇA NO
OFFLOADING COMPUTACIONAL**

Trabalho de Conclusão de Curso apresentado à
Universidade Federal do Ceará, Campus de
Crateús, como requisito parcial à obtenção do
título de bacharel em Ciência da Computação.

Orientador: Prof. Me. Francisco Anderson de
Almada Gomes.

Coorientador: Prof. Me. Filipe Fernandes dos
Santos Brasil de Matos.

Aprovado em:

BANCA EXAMINADORA

Prof. Me. Francisco Anderson de Almada Gomes (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Me. Filipe Fernandes dos Santos Brasil de Matos (Coorientador)
Universidade Federal do Ceará (UFC)

Prof. Me. Maurício Moreira Neto
Centro Universitário Christus (UniChristus)

Prof. Me. Adriano Lima Cândido
Universidade Vale do Salgado (UniVS)

Dedico este trabalho aos meus pais.

AGRADECIMENTOS

Agradeço ao meu orientador e coorientador, aos meus pais, e as pessoas e amigos que durante esses anos de graduação me incentivaram, me apoiaram e me ajudaram de alguma forma.

“Se você não gosta do seu destino, não aceite.
Em vez disso, tenha a coragem de o mudar do
jeito que você quer que seja.”

(Uzumaki Naruto)

RESUMO

É notório que a área de dispositivos móveis está aumentando ao decorrer dos anos. O celular é responsável por aproximadamente metade do tráfego da web em todo mundo. No quarto trimestre de 2021, os dispositivos móveis (sem considerar o uso de *tablets*) geraram 54,4% do tráfego global para sites, um número que varia constantemente em torno dos 50% desde o início de 2017. Entretanto, esses dispositivos ainda possuem limitações em relação aos seus recursos computacionais (armazenamento, processamento) e energéticos. É por essas limitações que, dependendo do tipo de aplicação em execução em um dispositivo móvel, essa aplicação pode exigir recursos não disponíveis no dispositivo para que a execução de uma determinada ação ocorra de forma rápida e eficiente. Conseqüentemente, não causa uma boa experiência de usabilidade para o usuário dessas aplicações. Já existem formas para mitigar esse problema, uma delas é a Computação em Nuvem Móvel (MCC), que realiza a expansão de recursos do dispositivo para um ambiente remoto, preservando os recursos locais e enriquecendo a experiência do usuário. Entretanto, com essa operação de migrar um determinado processamento para um ambiente remoto, os dados são trafegados na rede sem proteção alguma e muitas aplicações possuem dados sensíveis que não podem ser expostos sem segurança. Logo, é preciso o uso de mecanismos de segurança para garantir a privacidade desses dados. Levando em consideração os problemas enfrentados, o desenvolvimento da pesquisa baseia-se no estudo de técnicas de segurança que garantem a privacidade aos dados dos usuários na migração dos dados. Essas técnicas são a criptografia homomórfica e esteganografia. Como resultado final, é possível concluir que a utilização do algoritmo de Esteganografia se mostrou mais eficiente quando comparado com o algoritmo de criptografia Homomórfica para a maioria dos casos de testes realizados na pesquisa, uma vez que para a operação de multiplicação de matrizes com dimensões de 250x250 por exemplo, utilizando a criptografia Homomórfica possui um tempo total gasto de 6,7 minutos, enquanto essa mesma operação utilizando a Esteganografia demora cerca de 1000 milissegundos.

Palavras-chave: Dispositivos Móveis, Privacidade de dados, Algoritmos de criptografia, *Offloading*, Computação em Nuvem Móvel (MCC)

ABSTRACT

Mobile accounts for approximately half of the web traffic worldwide. In the fourth trimester of 2021, mobile devices (not counting the use of tablets) generated 54,4% of global traffic to websites, a number that constantly varies around 50% since the beginning of 2017. The area of mobile devices has been increasing over the years. However, these devices still have limitations in their computing resources (storage, processing) and energy. It is for these limitations that, depending on the type of application running on a mobile device, this application may require resources not available on the device so that the execution of a particular action occurs quickly and efficiently, and consequently, does not cause a good experience of usability for the user of these applications. There are already ways to mitigate this problem, one of them is Mobile Cloud Computing (MCC), which expands device resources to a remote environment, preserving local resources and enriching the user experience. However, with this operation of migrating specific processing to a remote environment, data is transported on the network without any protection, and many applications have sensitive data that cannot be exposed without security. Therefore, it is necessary to use security mechanisms to guarantee the privacy of this data. Considering the problems showed, the research development is based on the study of security techniques that ensure the privacy of user data during data migration. These techniques are homomorphic encryption and steganography. As a final result, it is possible to conclude that the use of the Steganography algorithm proved to be more efficient when compared with the Homomorphic encryption algorithm for most of the test cases carried out in the research, since for the multiplication operation of matrices with dimensions of 250x250, for example, using Homomorphic encryption has a total time spent of 6.7 minutes, while this same operation using steganography takes about 1000 milliseconds.

Keywords: Mobile Devices, Data Privacy, Encryption Algorithms, Offloading, Mobile Cloud Computing (MCC)

LISTA DE ILUSTRAÇÕES

Figura 1 – Modelo analítico	30
Figura 2 – Arquitetura do CAOS	33
Figura 3 – Criptografia Simétrica	38
Figura 4 – Criptografia Assimétrica	40
Figura 5 – Criptografia não Homomórfica	43
Figura 6 – Criptografia Homomórfica	44
Figura 7 – Processo de Esteganografia	47
Figura 8 – Arquitetura do CAOS com suporte à criptografia	51
Figura 9 – <i>Offloading</i> de Computação de Imagem Protegida por Esteganografia	53
Figura 10 – Valor 13 ocultado em três tamanhos de blocos diferentes :	54
Figura 11 – Comparação de desempenho de recuperação em diferentes métodos.	56
Figura 12 – Consumo de energia em diferentes métodos de <i>offloading</i>	57
Figura 13 – Arquitetura Proposta	61
Figura 14 – Tela Inicial	62
Figura 15 – Capturas de tela da aplicação LoadBench	64
Figura 16 – Tempo total gasto com o <i>offloading</i> para Esteganografia com imagem maior	69
Figura 17 – Tempo total gasto com o <i>offloading</i> na Esteganografia com imagem menor	70
Figura 18 – Tempo total gasto com o <i>offloading</i> para a criptografia Homomórfica	71
Figura 19 – Tempo de Comunicação x Tempo de Execução x Tempo de Criptografia (Esteganografia com imagem maior)	74
Figura 20 – Tempo de Comunicação x Tempo de Execução x Tempo de Criptografia (Esteganografia com imagem menor)	75
Figura 21 – Tempo de Comunicação x Tempo de Execução x Tempo de Criptografia (Homomórfica)	77

LISTA DE TABELAS

Tabela 1 – Algoritmos Criptográficos Escolhidos	60
---	----

LISTA DE ABREVIATURAS E SIGLAS

AES	<i>Advanced Encryption Standard</i>
CAOS	<i>Context Acquisition and Offloading System</i>
DES	<i>Data Encryption Standard</i>
e.g.	<i>Por exemplo</i>
MCC	<i>Mobile Cloud Computing</i>
NIST	<i>National Institute of Standards and Technology</i>
RSA	<i>Rivest-Shamir-Adleman</i>

SUMÁRIO

1	INTRODUÇÃO	14
1.1	Contextualização	14
1.2	Justificativa	16
1.3	Objetivos	18
<i>1.3.1</i>	<i>Objetivo Geral</i>	<i>18</i>
<i>1.3.2</i>	<i>Objetivos Específicos</i>	<i>18</i>
1.4	Organização do Trabalho	18
2	FUNDAMENTAÇÃO TEÓRICA	20
2.1	Computação em Nuvem	20
2.2	Computação Móvel	22
<i>2.2.1</i>	<i>Dispositivos Móveis</i>	<i>22</i>
2.3	Mobile Cloud Computing (MCC)	26
<i>2.3.1</i>	<i>Offloading</i>	<i>27</i>
<i>2.3.2</i>	<i>Por quê fazer o offloading?</i>	<i>27</i>
<i>2.3.3</i>	<i>Onde fazer offloading?</i>	<i>28</i>
<i>2.3.4</i>	<i>Quando realizar offloading?</i>	<i>29</i>
<i>2.3.5</i>	<i>Fazer offloading de quê?</i>	<i>30</i>
<i>2.3.6</i>	<i>Como executar o offloading?</i>	<i>31</i>
2.4	CAOS: Context Acquisition and Offloading System	31
<i>2.4.1</i>	<i>Arquitetura do CAOS</i>	<i>32</i>
2.5	Segurança da Informação	35
2.6	Técnicas de Criptografia	37
<i>2.6.1</i>	<i>Criptografia Simétrica ou Chave Privada</i>	<i>38</i>
<i>2.6.2</i>	<i>Criptografia Assimétrica ou Chave Pública</i>	<i>40</i>
2.7	Técnicas de criptografia avançadas	42
<i>2.7.1</i>	<i>Criptografia Homomórfica</i>	<i>42</i>
<i>2.7.2</i>	<i>Algoritmos de criptografia parcialmente homomórfica</i>	<i>45</i>
<i>2.7.3</i>	<i>Esteganografia</i>	<i>46</i>
<i>2.7.4</i>	<i>Técnicas e algoritmos de esteganografia</i>	<i>49</i>
2.8	Considerações Finais	49

3	TRABALHOS RELACIONADOS	50
3.1	Estudo Comparativo do Impacto da Segurança em Ambientes de <i>Mobile Cloud Computing</i>	50
3.2	<i>Tradeoff between Energy Savings and Privacy Protection in Computation Offloading</i>	52
3.3	<i>Energy Savings in Privacy-Preserving Computation Offloading with Protection by Homomorphic Encryption</i>	55
3.4	Considerações finais	57
4	UTILIZANDO TÉCNICAS DE SEGURANÇA NO <i>OFFLOADING</i> COMPUTACIONAL EM AMBIENTES DE FOG COMPUTING	58
4.1	Metodologia	59
4.2	Arquitetura	60
4.3	Considerações finais	62
5	RESULTADOS	63
5.1	Aplicação	63
5.2	Descrição do Ambiente de Execução	64
5.3	Descrição do experimento	65
5.4	Resultados Obtidos	68
5.5	Considerações finais	78
6	CONCLUSÃO E TRABALHOS FUTUROS	80
6.1	Resultados Alcançados	80
6.2	Limitações encontradas	81
6.3	Trabalhos Futuros	82
	REFERÊNCIAS	83

1 INTRODUÇÃO

1.1 Contextualização

Devido a Tecnologia da Informação ser uma área em constante evolução, a todo momento, subáreas que fazem parte dela e que possuem seus conceitos aplicados na prática, tendem a sofrer atualizações para se adaptar a novos cenários. Isso acontece ao mesmo tempo em que ocorre o surgimento de novas subáreas que enriquecem ainda mais a área de Tecnologia da Informação por completo.

Dentre essa áreas, aquela relacionada aos dispositivos móveis está aumentando ao decorrer dos anos. De acordo com Brandao e Valim (2016), os *smartphones* são parte expressiva da rotina de pessoas das gerações Y¹ e Z². Ao compará-las, é possível perceber que para a geração Z, o celular é visto como uma ferramenta essencial e é basicamente como se fizesse parte de toda e qualquer pessoa dessa geração. Além disso, essa geração tende a ser mais dependente dos dispositivos móveis digitais do que a geração Y, embora seja sutil a diferença encontrada entre ambas.

Na pesquisa realizada por Clement (2022), o celular é responsável por aproximadamente metade do tráfego da web em todo mundo. No quarto trimestre de 2021, os dispositivos móveis (sem considerar o uso de *tablets*) geraram 54,4% do tráfego global para sites, um número que varia constantemente em torno dos 50% desde o início de 2017, e que foi superado consistentemente em 2020.

No estudo realizado pela Cisco (2020), até 2023, os dispositivos móveis globais terão um crescimento partindo de 8,8 bilhões em 2018, para 13,1 bilhões em 2023. A categoria de *smartphones* é a segunda categoria que crescerá mais nesse período. Além disso, a combinação dos recursos do dispositivo juntamente com a evolução da conectividade de redes móveis (3G, 4G, 5G) facilitará a experimentação e implementação de aplicativos bem mais avançados, contribuindo para o aumento do tráfego móvel global.

As atividades mais populares em todo o mundo realizadas pelos usuários de dispositivos móveis incluem: assistir a filmes ou vídeos *online*, usar *e-mail* e acessar mídias

¹ De 1980 a 1994

² De 1995 até os dias atuais

sociais (CLEMENT, 2022). Os aplicativos, juntamente com o acesso a internet, são a maneira mais popular de assistir a vídeos de qualquer lugar que o usuário esteja, além disso, os aplicativos de entretenimento mais baixados na ³*Apple Store* são ⁴*Netflix*, ⁵*Tencent Video* e ⁶*Amazon Prime Video*, todos esses são serviços de ⁷*Streaming* de dados. (CLEMENT, 2022)

Os aplicativos para dispositivos móveis devem gerar mais de 613 bilhões de dólares em receitas em 2025, com os jogos para dispositivos móveis representando a maior participação na receita entre todas as categorias de aplicativos (CECI, 2022). Em 2020, jogos e vídeos representaram as maiores fatias do mercado de conteúdo móvel do ano. Os setores de *ePublishing*⁸ e educação ainda possuem um mercado limitado para seu conteúdo móvel, apesar do aumento no uso de aplicativos trazido pela pandemia do COVID-19.

O uso de dispositivos móveis tem aumentado de forma muito rápida nos últimos anos, e a tendência é que cresça ainda mais. Porém, assim como é dito na pesquisa feita por Paula *et al.* (2013), devido a grande evolução desses dispositivos que passaram a possuir funções cada vez mais semelhantes a um computador, conseqüentemente ficaram cada vez mais expostos a diversos tipos de ataques que comprometem a segurança e privacidade de dados.

Ataques que permitem a um invasor conseguir o acesso a informações pessoais através de uma rede que não possui mecanismos para prover segurança aos dados trafegados, e partir disso, o atacante pode visualizar, por exemplo, mensagens trocadas e atividades realizadas por esses usuários em seus dispositivos móveis. Alguns dos ataques que envolvem descoberta de dados pessoais por parte de invasores estão listados a seguir (PAULA *et al.*, 2013):

- ***Eavesdropping***: Um invasor pode espionar e até interceptar dados importantes ou ligações telefônicas, caso a rede não utilize fortes métodos de segurança;
- ***Message forgery***: Um invasor pode alterar o conteúdo de mensagens em direção dupla (emissor e receptor) de forma que os mesmos nem percebam;
- ***Man in the middle attack***: Um ataque em que um invasor pode interceptar mensagens estando entre o dispositivo móvel e um ponto de acesso da rede;

³ <https://www.apple.com/store>

⁴ <https://www.netflix.com/br/>

⁵ <https://wetv.vip/>

⁶ <https://www.primevideo.com/>

⁷ <https://www.pcmag.com/encyclopedia/term/streaming-service>

⁸ <https://dictionary.cambridge.org/pt/dicionario/ingles/e-publishing>

1.2 Justificativa

Devido a popularidade do uso dos dispositivos móveis, a todo momento surgem várias novas aplicações que colaboram com as mais diversas áreas de conhecimento (*Por exemplo (e.g.), Saúde, Educação, Engenharia e Meio Ambiente*). Entretanto, os dispositivos móveis utilizados para a execução dessas aplicações poderão apresentar algumas limitações em relação aos seus recursos computacionais, assim como abordado em Trinta *et al.* (2020), tais como:

- Baixo poder de processamento de dados;
- Necessidade de economia energética;
- Baixo poder de armazenamento de dados;

É devido a essas limitações que, dependendo da aplicação a ser executada em um dispositivo móvel, essa aplicação pode necessitar de recursos não disponíveis pelo dispositivo para que a execução de uma determinada ação ocorra de forma rápida e eficiente, e conseqüentemente, não causa uma boa experiência de usabilidade para o usuário final.

Já existem formas para mitigar esse problema relacionado as limitações sobre os recursos computacionais dos dispositivos móveis. Uma delas é a Computação em Nuvem Móvel (*Mobile Cloud Computing ou MCC*), que realiza a expansão de recursos do dispositivo para um ambiente remoto, preservando os recursos locais do aparelho e enriquecendo a experiência do usuário (TRINTA *et al.*, 2020). Essa expansão é feita através do *offloading*, que se trata de uma técnica de migração de processamento/armazenamento do aparelho móvel para uma infraestrutura remota (SANTOS *et al.*, 2017).

No entanto, se não tomado o devido cuidado com essa operação de migrar um determinado processamento/armazenamento para um ambiente remoto, os dados, eventualmente, podem ser trafegados na rede sem proteção alguma. Além disso, muitas aplicações possuem dados sensíveis (*e.g., CPF, RG, dados bancários, resultados de exames médicos*) e que, não é interessante que sejam expostos sem privacidade, logo, é necessário utilizar mecanismos de segurança para garantir a proteção desses dados.

Um dos métodos que é bastante utilizado para proteção de dados para esses ambientes é através de criptografia de dados. A criptografia torna os dados ilegíveis, e partir desses dados

sem sentido, é possível realizar a recuperação dos dados originais. Normalmente isso é feito através da utilização de algoritmos de criptografia, que são basicamente programas de computador que implementam a ideia de um método de criptografia escolhido (BURNETT; PAINE, 2002).

Esses algoritmos normalmente recebem como entrada os dados legíveis e os transformam em dados ilegíveis, o contrário também pode ser feito. As técnicas criptográficas podem ser adaptadas de diferentes formas com o intuito de criar identidades digitais mais difíceis de serem identificadas. Caso algum invasor queira se fingir de uma outra pessoa, é necessário resolver o problema matemático por trás da técnica utilizada (BURNETT; PAINE, 2002).

Algumas das principais técnicas de criptografia abordadas pelo presente trabalho são: criptografia Homomórfica e Esteganografia. A primeira permite que após a encriptação de um determinado dado, seja possível realizar operações diretamente sobre o dado encriptado, enquanto outras técnicas (e.g., simétrica e assimétrica) possuem a necessidade de encriptar e decryptar o dado para executar ações sobre o mesmo. A segunda se trata de uma técnica de ocultação de dados através de mídias digitais (e.g., imagem, vídeo, áudio). De fato, o dado é embutido dentro de uma mídia e trafegado de forma oculta, o dado pode ser tanto texto puro quanto uma mídia digital.

O presente trabalho visa analisar o impacto da adição de segurança no processo de *offloading* computacional, através da utilização de algoritmos de criptografia de dados, através das técnicas de criptografia Homomórfica e técnicas de ocultação de dados através de Esteganografia.

Dito isso, o trabalho beneficia principalmente usuários e desenvolvedores de aplicações para dispositivos móveis. Devido ao fato que será possível garantir a integridade e confidencialidade na operação de *offloading*. Com isso, desenvolvedores terão uma base para a criação de aplicações que necessitam preservar os recursos locais do dispositivo móvel, com uma camada de segurança através do uso de algoritmos de criptografia de dados.

O desenvolvimento da pesquisa baseia-se no estudo de técnicas que garantem a privacidade aos dados dos usuários, estendendo o trabalho anterior de Silva (2019) e Gomes *et al.* (2020) que utilizam técnicas e algoritmos de criptografia simétrica e assimétrica e fazendo a adição das técnicas e algoritmos de criptografia Homomórfica e Esteganografia, como forma de expandir as possibilidades de técnicas de criptografia que podem ser utilizadas no processo *offloading*.

1.3 Objetivos

1.3.1 *Objetivo Geral*

Avaliar os impactos causados pela adição das técnicas de criptografia homomórfica e esteganografia em termos de tempo de processamento e selecionar uma estratégia que se mostra capaz de mitigar o problema da segurança nesse ambiente sem causar grandes perdas de desempenho.

1.3.2 *Objetivos Específicos*

- Analisar as estratégias de segurança na migração de dados;
- Analisar o impacto da segurança no desempenho de tarefas computacionais no *offloading*;
- Realizar um estudo comparativo entre as estratégias utilizadas.

1.4 Organização do Trabalho

Este trabalho está organizado em seis capítulos, conforme descrito a seguir:

- **Capítulo 2** - Pesquisa bibliográfica sobre o campo de interesse da pesquisa, a fim de compreender os conceitos e elementos fundamentais do trabalho: Computação em Nuvem, Computação Móvel e dispositivos móveis, Computação em Nuvem Móvel e conceitos de Segurança da Informação com foco em algoritmos de criptografia;
- **Capítulo 3** - Análise e comparação de pesquisas que estão relacionadas ao presente trabalho. Foram analisados trabalhos que propõem segurança em ambientes de *Mobile Cloud Computing* (MCC) e que possuem criptografia de dados como segurança desses ambientes;
- **Capítulo 4** - Apresentação da metodologia adotada para a realização da pesquisa, exibindo a distribuição das etapas adotadas, bem como da proposta;
- **Capítulo 5** - Apresentação dos experimentos que foram utilizados para avaliar as criptografias. Além disso, são apresentados os resultados encontrados no trabalho proposto.
- **Capítulo 6** - Por fim, são apresentadas as conclusões e os trabalhos futuros do

trabalho proposto.

2 FUNDAMENTAÇÃO TEÓRICA

Este Capítulo apresenta os conceitos, definições e principais características relacionados às áreas e subáreas que fundamentam a presente pesquisa e que são de suma importância para o entendimento da mesma. Serão abordados conceitos de Computação em Nuvem, Computação Móvel e dispositivos móveis, Computação em Nuvem Móvel e Segurança da Informação com foco em algoritmos de criptografia.

2.1 Computação em Nuvem

Devido o surgimento de diversos avanços tecnológicos nos últimos anos, sendo alguns deles relacionados, principalmente com a conexão com a Internet e suas altas taxas de velocidades, surgiram empresas que construíram infraestruturas de Tecnologia da Informação com alto poder de processamento e armazenamento, e que disponibilizaram esses recursos computacionais através da Internet, de forma que pessoas físicas e jurídicas pudessem fazer o uso desses serviços pagando a essas empresas valores acessíveis (VARELLA, 2019).

A partir desse momento, surge o conceito de Computação em Nuvem (*Cloud Computing*) que, de acordo com (AUGUSTO *et al.*, 2016), originou-se quando aplicações passaram a necessitar de grandes centros de dados que suportem serviços da Web em larga escala. No momento da adoção deste modelo, foi possível obter uso eficiente, otimizado e flexível de recursos (e.g., processamento, armazenamento) de *hardware e software*.

Segundo o NIST (2011), a Computação em Nuvem é um modelo que possibilita acesso a rede de forma onipresente (ou seja, de qualquer lugar) e conveniente (no momento que desejar) sob demanda, a um conjunto compartilhado de recursos de computação (e.g., aplicações, equipamentos, dispositivos de armazenamento e processamento de dados) configuráveis que podem ser, a qualquer momento, alocados ou liberados sem a necessidade de uma interação direta com o provedor do serviço.

Ainda segundo o NIST (2011), existem características essenciais para um bom entendimento da arquitetura da Nuvem. Essas características estão listadas abaixo:

- **Autoatendimento sob demanda:** Um usuário dos serviços da Nuvem pode solicitar a

adição ou diminuição dos recursos computacionais sem a necessidade do contato com dono do provedor de serviço;

- **Amplo acesso à rede:** Os recursos da Nuvem são disponibilizados através da rede e podem ser consumidos por qualquer dispositivo (*e.g.*, celulares, *tablets*, *laptops*) com acesso a Internet;
- **Agrupamento de recursos:** Os vários recursos disponibilizados são reunidos para atender a diversos usuários da Nuvem. Esses recursos podem ser realocados dependendo da demanda, sem a necessidade dos usuários saberem a localização exata dos mesmos;
- **Elasticidade:** Os recursos podem aumentar ou diminuir rapidamente com o objetivo de atender melhor a demanda do usuário;
- **Serviço medido:** O controle em relação a utilização dos recursos é feito de forma automática, ou seja, o cliente paga o equivalente ao que ele realmente utilizou.

Empresas como a *Amazon*, *Google* e *Microsoft* possuem plataformas que disponibilizam diversos serviços de Computação em Nuvem. A *Amazon* possui a AWS (*Amazon Web Services*) que disponibiliza serviços como o EC2¹ (*Elastic Compute Cloud*), que possibilita aos desenvolvedores tenham o acesso a uma capacidade computacional e redimensionável de armazenamento e processamento para publicação de aplicativos. No caso da *Google*, a empresa tem como plataforma de Nuvem a *Google Cloud Platform*, que possui serviços como o *Cloud Sql*², que disponibiliza acesso a banco de dados relacionais (*e.g.*, *MySQL*, *PostgreSQL* e *SQLServer*) de qualquer lugar do mundo, e que realiza *backups* e replicações de forma automática, visando garantir disponibilidade, flexibilidade e desempenho do serviço.

Por sua vez, a *Microsoft* possui a *Microsoft Azure*, que possui serviços como o *Azure Data Share*³, que possibilita o controle do compartilhamento de dados de qualquer formato e tamanho e de várias fontes com outras organizações.

Desta forma, a Computação em Nuvem tem papel essencial no presente trabalho, pois é partir dos conceitos desta área que será possível ter um bom entendimento sobre o tópico de Computação em Nuvem Móvel. O próximo tópico será sobre Computação Móvel, em que serão abordados os conceitos para o melhor entendimento sobre esse paradigma e sobre dispositivos móveis.

¹ <https://aws.amazon.com/pt/ec2>

² <https://cloud.google.com/sql>

³ <https://azure.microsoft.com/pt-br/services/data-share>

2.2 Computação Móvel

Assim como já foi dito nos tópicos anteriores sobre as evoluções da tecnologia, a Computação Móvel (*Mobile Computing*) surgiu a partir dessas inovações. De acordo com Figueiredo e Nakamura (2003), a partir dos anos 90 houve um crescimento nas áreas de comunicação por redes de celular móvel, comunicação via satélite e redes locais sem fio. A globalização dessas tecnologias tornou possível o acesso remoto às informações, aplicações e serviços para os diversos tipos de usuários. Mas para uma melhor compreensão deste tópico, deve-se partir do conceito inicial de Mobilidade.

A Mobilidade pode ser definida como a capacidade de poder deslocar ou ser deslocado facilmente (LEE; SCHELL, 2005). No contexto da *Mobile Computing*, a mobilidade está relacionada com o uso, por parte das pessoas, de dispositivos móveis (*e.g., notebooks e smartphones*). Esses dispositivos devem ter o poder de realizar um conjunto de funções relacionadas a aplicações presentes nos mesmos, além de conectar-se à rede, receber dados e transmiti-los para usuários e aplicações. (LEE; SCHELL, 2005)

Para Figueiredo e Nakamura (2003), a *Mobile Computing* pode ser considerada como um paradigma da computação que permite aos usuários, independentes de seu ambiente e localização, ter acesso a serviços e informações, inclusive, em movimento. Tecnicamente falando, é um conceito que envolve processamento, mobilidade e comunicação sem fio, fazendo com que o acesso à informação seja de qualquer lugar e a qualquer momento, liberando o usuário de estar conectado a uma estrutura física e estática.

Dados os conceitos iniciais sobre a Computação Móvel, o próximo tópico fica responsável por tratar mais detalhadamente os conceitos de Dispositivos Móveis e suas características.

2.2.1 Dispositivos Móveis

A Computação Móvel se consolidou devido a grande adesão por parte das pessoas aos dispositivos móveis durante o decorrer dos anos, e existem dois motivos principais para esta alta adoção: primeiro, as aplicações que foram e ainda são criadas para esses dispositivos facilitam a vida das pessoas em relação a comunicação, mobilidade e lazer. Segundo, as características únicas e específicas desses aparelhos são propícias para a locomoção dos mesmos juntamente com seus usuários, logo, o acesso a essas aplicações se torna praticamente constante.

Provavelmente existirão dispositivos móveis que não possuem todas as características descritas abaixo, mas todas elas são essenciais para o maior proveito da Computação Móvel, assim como são apresentadas por Lee e Schell (2005) e Figueiredo e Nakamura (2003):

- **Portabilidade:** É a capacidade que esses dispositivos possuem de serem transportados facilmente para qualquer lugar. Atualmente, é possível perceber esse conceito aplicado na prática em *smartphones*, que normalmente são aparelhos com tamanhos reduzidos comparados com aparelhos mais antigos. E para que aparelhos tenham uma boa portabilidade, os fatores abaixo são de suma importância:
 - **Tamanho:** O dispositivo móvel precisa ter um tamanho reduzido, já que se ele for muito grande, poderá dificultar sua locomoção. Porém, não podem ser extremamente pequenos, pois isso pode dificultar o uso dos mesmos;
 - **Peso:** O peso é outro fator importantíssimo quando se trata de mobilidade. Atualmente os maiores dispositivos considerados móveis (*e.g., notebooks*) chegam a pesar alguns quilos, já os menores, chegam a pesar algumas gramas. Logo, é indiscutível que algo que pesa algumas gramas terá uma melhor mobilidade do que algo que pesa alguns quilos;
- **Usabilidade:** É um conceito que está relacionado com a facilidade que as pessoas tem ao utilizar algo para realizar uma determinada ação. Nesse contexto, se um dispositivo móvel for de fácil utilização para realizar tarefas de seu usuário, diz-se que ele tem uma boa usabilidade. Entretanto, a usabilidade de dispositivos móveis está diretamente relacionada às características de seus usuários, do ambiente em que se encontram e dos próprios aparelhos. Abaixo essas características estão descritas de uma forma mais detalhada:
 - **Usuário:** Fatores como conhecimento, tamanho, força e flexibilidade do usuário vão influenciar na usabilidade. Um *notebook* pode ser facilmente transportado por uma pessoa adulta, mas, pra uma criança, pode ser trabalhoso. Em geral, um adulto dedos maiores do que uma criança, logo um teclado muito pequeno pode não ser o ideal. É impossível saber o nível de conhecimento de todas as pessoas, logo, o dispositivo móvel deve ser o mais intuitivo possível, pois se o usuário sentir dificuldade em usá-lo, não será útil para ele;
 - **Ambiente:** O ambiente em que o usuário utiliza o dispositivo afeta diretamente na sua usabilidade. Por exemplo, para um garçom que trabalha muito tempo em pé anotando pedidos, é ideal para ele utilizar um dispositivo menor como um celular ou

tablet ao invés de um *notebook*. Dispositivos a prova d'água ou mais resistentes são os ideais para quem trabalha em ambientes arriscados, como profissionais de serviço de emergência;

- **Funcionalidade:** Os dispositivos móveis possuem diversas funcionalidades e servem para múltiplos propósitos. Essas funcionalidades são implementadas no dispositivo na forma de aplicação móvel, e cada dispositivo móvel, no geral, roda múltiplas aplicações móveis. Normalmente essas aplicações são classificadas nas duas categorias que estão descritas abaixo:
 - **Independentes:** São aplicações que funcionam sem precisar de contato com usuário ou outro sistema. (*e.g. relógio, jogos, calculadora*)
 - **Dependentes:** Esse conceito é contrário ao anterior. Nesse caso, as aplicações dessa categoria dependem de usuários ou de estarem conectadas a outros sistemas para funcionar. (*e.g. GPS, calendário, correio eletrônico*)
- **Conectividade:** A conectividade é basicamente a capacidade do dispositivo em conectar-se a uma infraestrutura de rede, que vai lhe proporcionar, na maioria das vezes, acesso a Internet, e a partir disso, o acesso a informações afim de realizar alguma tarefa. O dispositivo pode se conectar através de tecnologias de rede com fio e sem fio (*e.g., wireless*), mas para o contexto de Computação Móvel, a conexão *wireless* é a que faz mais sentido. Abaixo estão descritas alguns tipos de conexão de rede sem fio:
 - **Bluetooth:** Esse tipo de tecnologia permite que dispositivos possam se conectar de forma sem fio a vários dispositivos de telecomunicações ou eletrônicos (*e.g., impressoras, tablets e smartphones*). O *Bluetooth* usa tecnologia de rádio frequência, permitindo que dispositivos possam estar em qualquer posição para conexão, porém com uma limitação espacial de até 10 metros. Com ele é possível montar uma rede com até 8 dispositivos;
 - **Redes de celulares:** Esse tipo de infraestrutura é composta por um conjunto de células de rádio com tamanhos variados. Se um usuário de dispositivo móvel está dentro de uma célula, é possível receber e transmitir sinais de rádio. Se a cobertura do provedor de serviço for boa, o usuário pode mover-se tranquilamente sem perder a conexão;
 - **Redes locais sem fio:** O padrão 802.11 é o mais conhecido dessas redes, popularmente conhecido como *Wi-Fi*. Normalmente esse tipo de rede possui uma alta

velocidade, e hoje em dia está presente em quase todo local que possui acesso a Internet, principalmente locais públicos como restaurantes, bares, praças, aeroportos, dentre outros;

- **Redes de satélite:** Esse tipo de rede é outra alternativa sem fio para conectar dispositivos móveis. O uso mais comum dessa tecnologia é o GPS, que é um sistema de navegação que utiliza satélites na órbita da Terra que transmitem sinais. Dispositivos que possuem o GPS captam esses sinais e conseguem representar a posição onde o dispositivo se encontra.

Embora a combinação da Computação Móvel juntamente com o uso dos Dispositivos Móveis tenham várias vantagens, atualmente, existem demandas por parte de usuários, serviços e aplicações que fazem com que esses dispositivos tenham limitações computacionais, mesmo eles passando por diversas evoluções durante esses anos, afirma Gupta (2008).

Em Figueiredo e Nakamura (2003), é apresentado algumas dessas limitações. Primeiramente, em relação a redes sem fio, que possuem largura de banda limitada e possível perda de dados por causa de interferências ou pelo fato de desconexão com a rede devido a mobilidade. Outra limitação é em relação a segurança dos dados que são trafegados na rede, pois, nas redes sem fio, os dados são transmitidos através de um meio compartilhado por vários usuários e que, se não adotada nenhuma medida protetiva, coloca em risco a integridade e confidencialidade dos mesmos.

Em relação aos dispositivos móveis, uma limitação importante está relacionada ao consumo de energia, pois a fonte de energia desses dispositivos tendem a não ter uma boa duração devido a mobilidade e algumas aplicações exigem um poder de processamento mais alto, e conseqüentemente, consomem mais energia.

Em relação ao desenvolvimento de aplicações para esses dispositivos, deve ser levado em consideração essas limitações já citadas, mas além disso, na maioria das vezes, essas aplicações lidarão com dados pessoais dos usuários, logo, é necessário que as aplicações preservem a privacidade desses dados, visto que nenhum usuário vai se sentir seguro em disponibilizar seus dados para soluções que não preservem os mesmos.

Diante da contextualização sobre os conceitos que relacionam a Computação Móvel e os Dispositivos Móveis, percebeu-se que esses temas são bastante importantes para o presente

trabalho, porém ainda possuem problemas que precisam ser contornados. O próximo tópico é responsável por tratar sobre os conceitos do paradigma de Computação em Nuvem Móvel (*Mobile Cloud Computing* ou *MCC*).

2.3 *Mobile Cloud Computing* (MCC)

A MCC surgiu com objetivo de contornar os problemas presentes na Computação Móvel. Existem várias definições para esse paradigma. De acordo com Qureshi *et al.* (2011), a *Mobile Cloud Computing* é um paradigma da computação que faz o uso das vantagens presentes na Computação em Nuvem para resolver os problemas (*e.g., consumo de energia, processamento e armazenamento*) advindos da Computação Móvel, reunindo e integrando essas duas áreas.

Assim como Qureshi *et al.* (2011), Huang e Wu (2018) afirma que a MCC foi construída considerando os conceitos da Computação em Nuvem e Computação Móvel, com o objetivo de permitir que aplicativos que exigem recursos computacionais mais sofisticados possam ser executados em diferentes tipos de dispositivos móveis proporcionando uma boa experiência para os usuários.

Para Trinta *et al.* (2020), a MCC é um paradigma da computação que engloba três tecnologia diferentes: Computação em Nuvem, Computação Móvel e Redes sem fio. Trinta *et al.* (2020) afirma que o paradigma possui o objetivo de reduzir as limitações já citadas anteriormente dos dispositivos móveis através do acesso sem fio onipresente à recursos da Nuvem. Através disso é possível aumentar os recursos dos dispositivos móveis preservando os recursos locais, além de enriquecer a experiência dos seus usuários.

Sanaei *et al.* (2013) afirma que a *Mobile Cloud Computing* é uma rica área da Computação Móvel que faz proveito dos recursos elásticos e unificados de diversas Nuvens e de tecnologias de rede para disponibilizar funcionalidades, armazenamento e mobilidade de qualquer lugar e à qualquer momento, independente de ambientes ou plataformas e baseado no pagamento pela obtenção desses recursos conforme o seu uso.

Existem diversas técnicas de MCC que fazem com que essa tecnologia possa prover todas essas vantagens mencionadas anteriormente, e a mais importante delas para o presente trabalho é o *offloading*, que possibilita a transferência de computação dos dispositivos móveis com recursos escassos para máquinas mais poderosas computacionalmente (REGO *et al.*, 2017).

2.3.1 *Offloading*

O *offloading* (descarregamento) é o principal tópico de pesquisa da *Mobile Cloud Computing*, assim como é dito no trabalho de (FERNANDO *et al.*, 2013). Segundo a pesquisa desenvolvida por Santos *et al.* (2017), o *offloading* é a técnica popular utilizada para aumentar o desempenho e reduzir o gasto energético dos dispositivos móveis através da migração do processamento de dados para infraestruturas que possuem um maior poder computacional e de armazenamento.

É importante deixar claro que a operação de *offloading* é diferente da arquitetura tradicional de Cliente-Servidor. Enquanto na arquitetura Cliente-Servidor a transmissão/processamento de dados sempre é migrada para o servidor e só é possível em uma rede que possui conexão com a Internet, o *offloading* permite que o processamento de dados possa acontecer localmente caso não exista conexão com a Internet, ou caso o dispositivo móvel não tenha benefícios ao delegar a computação da tarefa para o servidor. (GOMES *et al.*, 2017)

No trabalho de Fernando *et al.* (2013), é mostrado que existem dois tipos de operação de *offloading*: de processamento e de dados. O *offloading* de processamento é uma operação computacional que delega o processamento que seria realizado no dispositivo móvel para outro ambiente de execução (*e.g.*, Nuvem, Névoa e outros dispositivos móveis), com o objetivo de prolongar a vida útil da bateria e aumentar a capacidade computacional. O *offloading* de dados tem por objetivo estender a capacidade de armazenamento do dispositivo móvel, delegando o armazenamento do dispositivo e enviando os dados para um ambiente com maior capacidade de armazenamento.

Em Trinta *et al.* (2020) é mostrado que existem questões importantes que devem ser levadas em conta para utilização dessa operação. Essas questões serão abordadas nos tópicos seguintes.

2.3.2 *Por quê fazer o offloading?*

Devido os dispositivos móveis possuírem essas limitações, os pesquisadores orientam o uso do *offloading* computacional para melhorar o desempenho dos aplicativos, economizar o consumo de bateria e executar aplicações que não seria possível devido essa falta de recursos. Segue as três principais questões relacionadas a esse tópico:

- **Melhor desempenho:** Quando o objetivo principal é melhorar o desempenho, é possível reduzir o tempo de processamento de aplicações;
- **Economia de energia:** Quando o objetivo é economizar energia, é possível reduzir o consumo energético dos dispositivos móveis;
- **Outros:** Quando energia e desempenho não são os principais motivos para a execução do *offloading* computacional. Em vez disso, o principal motivo pode ser melhorar a colaboração para realizar uma determinada tarefa, estender o armazenamento/capacidade ou reduzir custos monetários.

2.3.3 Onde fazer *offloading*?

Como já foi dito nos parágrafos anteriores, os dispositivos móveis fazem o uso de recursos remotos para melhorar seu desempenho. Dentre esses recursos remotos, os principais são: Nuvem pública, *cloudlets* e outros dispositivos móveis.

A Nuvem pública é normalmente a mais usada pelos dispositivos móveis para fazer proveito de seus recursos de elasticidade e conectividade. Os dispositivos podem se conectar na Nuvem através de redes móveis de celulares (*e.g.*, 3G, 4G e 5G) ou uma rede Wi-Fi. Aplicativos como *GoogleDocs* e *Gmail* são exemplos de aplicações que necessitam de Internet o tempo todo para utilizá-los.

No trabalho de Satyanarayanan (2001), os autores utilizavam servidores na vizinhança para lidar com a falta de recursos dos dispositivos móveis e melhorar a tempo de comunicação, que, futuramente, esses servidores passaram a se chamar *Cloudlets*. O conceito de *Cloudlet* é mais recente e surgiu no trabalho de Satyanarayanan *et al.* (2009) com o objetivo de utilizar equipamentos que compõem uma rede local, possibilitando que tais dispositivos possam atuar como servidores e atender as solicitações de *offloading* localmente, sem tornar necessário a comunicação do dispositivo móvel direto com a Nuvem. Assim, o processo de *offloading* se mantém próximo aos dispositivos e obtém vantagens como maiores taxas de velocidades e menores taxas de latência, visto que redes locais possuem essas vantagens em relação a conexões mais remotas como apontado pelo estudo feito por Costa *et al.* (2015).

Além dos dois recursos citados anteriormente, a técnica de *offloading* também pode ser realizada para outros dispositivos móveis. Nesse caso, um conjunto de dispositivos móveis

forma uma rede móvel e os que possuem maior poder computacional disponibilizam esses recursos para os dispositivos com menor desempenho (FERNANDO *et al.*, 2013).

2.3.4 *Quando realizar offloading?*

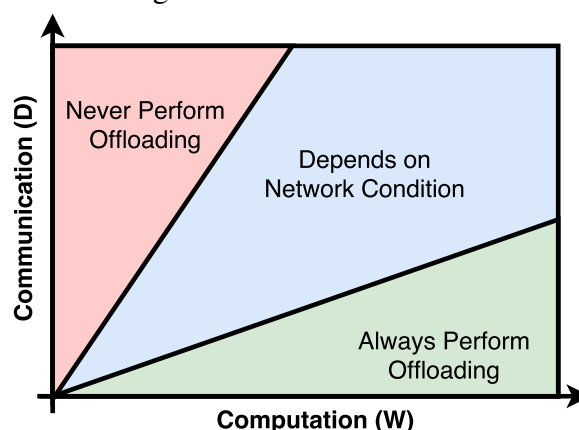
Até então, os trabalhos que existem em relação a esse questionamento não entraram em um consenso sobre quando realizar essa operação, isso vai depender muito do tipo aplicação e qual sua finalidade. Algumas aplicações podem priorizar a produtividade enquanto outras, a economia de energia, e isso acaba se tornando relativo e dependendo do objetivo principal da aplicação.

Em Kharbanda *et al.* (2012), é apresentado algumas abordagens para economizar a vida útil da bateria dos dispositivos móveis: mudar geração de semicondutores, fazer com que o partes da aplicação ou ela por completa possa entrar em repouso, executar as aplicações mais lentamente e, por fim, eliminar totalmente a computação do aparelho e delegá-la para outra infraestrutura.

No trabalho de Kumar *et al.* (2013) é apresentado um modelo analítico que visa responder a pergunta deste tópico em termos de melhoria no desempenho de dispositivos móveis. Esse modelo compara o tempo que uma tarefa leva para ser executada localmente no dispositivo móvel e o tempo para executar esse processamento de forma remota (*e.g.*, *Nuvem e Cloudlet*). A Figura 1 mostra o modelo feito pelo autor. A partir desse modelo, foi possível concluir as seguintes questões:

- **Recomendado fazer remotamente:** Se uma aplicação exige um processamento muito grande e a velocidade de transmissão dos dados for relativamente boa, a operação de *offloading* é recomendada;
- **Recomendado fazer localmente:** É o caso contrário ao anterior, se a aplicação não exige muito processamento e a conexão remota não é satisfatória, não é recomendado fazer o *offloading*;
- **Local ou remoto:** Se as duas questões anteriores estiverem em termos intermediários, ou seja, a aplicação não exige tanto processamento e a conexão possui uma comunicação razoável, fatores como o consumo de energia podem influenciar na decisão de realizar ou não o *offloading*.

Figura 1 – Modelo analítico



Fonte – (KUMAR *et al.*, 2013)

Para essa abordagem apresentada em Kumar *et al.* (2013), existe uma extensão proposta no trabalho de Rego (2016). Essa abordagem está relacionada com a utilização de dados históricos de processamento com o objetivo de criar árvores para auxiliar na decisão de realizar ou não a operação de *offloading*. O servidor fica responsável por criar as árvores de decisão e os dispositivos ficam responsáveis apenas por analisá-las, trazendo, como vantagem, a economia energética por parte dos dispositivos móveis.

2.3.5 Fazer *offloading* de quê?

Normalmente, caso a aplicação não esteja completamente hospedada em um servidor remoto, partes da mesma poderão ser migradas para o servidor. Segundo Liu *et al.* (2015), uma operação de particionamento pode ser utilizada para dividir a aplicação em componentes com diferentes níveis de granularidade. Existem vários estudos que tratam de quais partes da aplicação devem ser migradas ao servidor, mas, usualmente, as partes a seguir são as utilizadas:

- **Métodos:** Quando métodos ou funções da aplicação passam a ser acessíveis através procedimentos remotos ou chamadas direta do lado do servidor;
- **Componentes/Módulos:** Quando parte maiores da aplicação passam a ser acopladas do lado do servidor, como por exemplo, um grupo de classes;
- **Threads:** Quando *threads* podem ser migradas para o servidor remoto ou entre dispositivos móveis. Normalmente envolve alterações na máquina virtual do Android;
- **Aplicação inteira:** Nesse caso, são usadas técnicas de virtualização para executar clones dos dispositivos e todo o estado e execução da aplicação fica fora do dispositivo móvel.

2.3.6 Como executar o offloading?

Como é dito por Trinta *et al.* (2020), não há uma resposta concreta para a pergunta desse tópico, pois normalmente as soluções desenvolvidas são muito variadas e pensadas mais em relação aos tópicos que já foram explanados até então. Entretanto, Rego (2016) faz uma revisão da literatura e cria uma taxonomia para ajudar a responder essa questão. As categorias selecionadas estão detalhadas abaixo:

- **Anotação do Método:** Quando a solução de *offloading* suporta a granularidade de método;
- **Localização do Módulo de Decisão:** Está relacionada com a localização do módulo responsável pela decisão de realizar o *offloading*;
- **Abordagem do Módulo de Decisão:** Está relacionada com as abordagens e técnicas utilizadas pelo módulo de decisão;
- **Métricas Utilizadas para Decisão:** Diz respeito às métricas avaliadas no módulo de decisão do *offloading*;
- **Plataforma/Linguagem de Programação Suportada:** Diz respeito às plataformas móveis e linguagens de programação suportadas pela solução; e
- **Mecanismo de Descoberta:** Essa categoria leva em conta se a solução usa algum tipo de mecanismo de descoberta do ambiente remoto (*e.g.*, *cloudlet*).

Várias infraestruturas que oferecem suporte ao desenvolvimento de aplicações móveis com foco em *offloading* foram desenvolvidas (FERNANDO *et al.*, 2013; GUAN *et al.*, 2011; PARANJOTHI *et al.*, 2017; RAHIMI *et al.*, 2014; LIN *et al.*, 2020; DESHMUKH; SHAH, 2016; KHAN, 2015; XU *et al.*, 2018). A seguir, é apresentada uma infraestrutura desenvolvida no Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat)⁴ intitulada CAOS (*Context Acquisition and Offloading System*).

2.4 CAOS: *Context Acquisition and Offloading System*

No trabalho de Gomes *et al.* (2017), é dito que os dispositivos móveis (*e.g.*, *smartphones e tablets*), tiveram seus recursos computacionais melhorados ao decorrer dos anos devido a constante evolução de seus componentes de *hardware*, como o surgimento de novas gerações de processadores mais potentes, mais capacidade de armazenamento e melhores interfaces de rede.

⁴ <https://www.great.ufc.br/>

Além disso, segundo o autor, está cada vez mais indispensável o uso de dados contextuais dos usuários que são obtidos através de diversos sensores (*e.g.*, *GPS*, *acelerômetro* e *giroscópio*) instalados nos dispositivos móveis para melhorar a usabilidade e a experiência do usuário com as aplicações. Entretanto, devido as limitações já citadas dos dispositivos móveis e a complexidade de inferência de dados contextuais, os dispositivos ainda possuem recursos insuficientes para realizar algumas tarefas nesse contexto.

O *Context Acquisition and Offloading System* (CAOS) surgiu exatamente para ajudar no tratamento da complexidade dessas aplicações para os dispositivos móveis. Tanto Gomes *et al.* (2017) quanto Trinta *et al.* (2020) definem o CAOS como uma plataforma de software criada para auxiliar os desenvolvedores na criação de aplicações sensíveis ao contexto para a plataforma Android, fornecendo suporte a mecanismos de *offloading* computacional com o objetivo migrar dados contextuais ou o processamento dos mesmos para a Nuvem, visando obter melhorias em relação às deficiências presentes nos dispositivos móveis discutidas até então.

Como já foi dito, o CAOS é uma plataforma que surgiu para mitigar os problemas relacionados aos recursos escassos presentes nos dispositivos móveis, promovendo economia de energia e ganhos no desempenho do processamento de dados. Para tanto, essa plataforma possui uma arquitetura bastante importante e que será explicada no tópico seguinte.

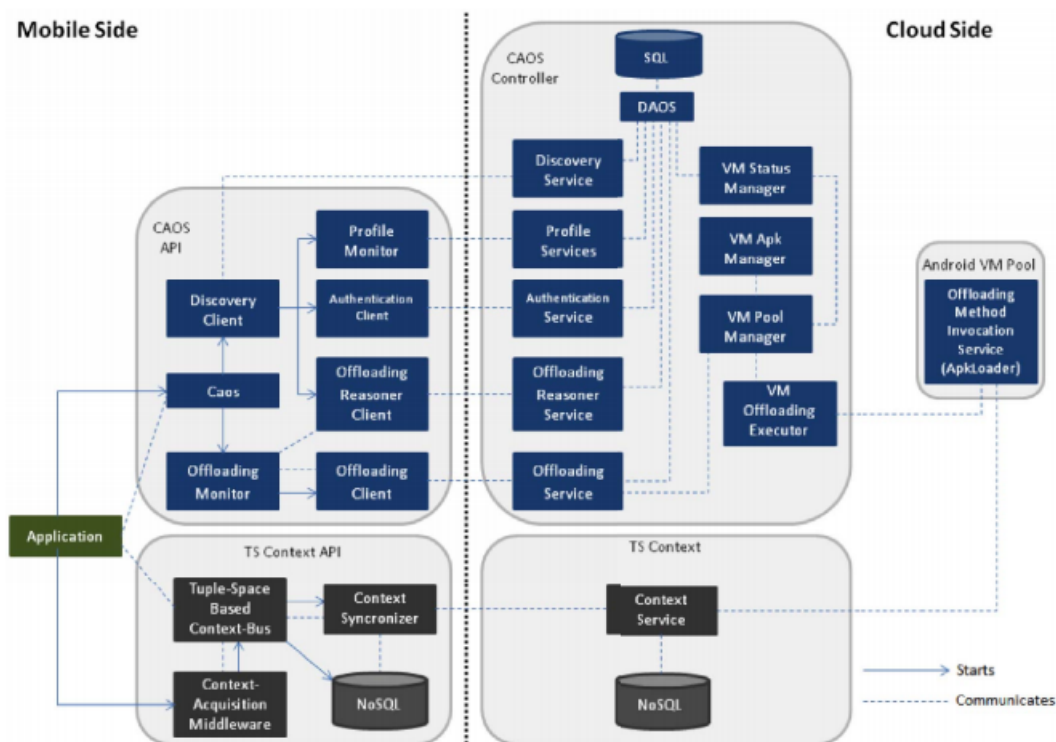
2.4.1 Arquitetura do CAOS

Gomes *et al.* (2017) e Trinta *et al.* (2020) mostram que a arquitetura utilizada por esse *framework* é basicamente uma arquitetura Cliente-Servidor tradicional. O lado cliente representa os dispositivos móveis que vão requisitar ações que serão processadas pelo lado servidor, ou melhor, a Nuvem. Os autores também utilizam a nomenclatura de lado Móvel (*Mobile Side*) e lado da Nuvem (*Cloud Side*), em que cada um desses lados são constituídos por um conjunto de componentes que funcionam em conjunto para realizar as operações citadas até então. A Figura 2 contém essa arquitetura e seu conjunto de componentes que serão explicados de forma mais detalhada.

O lado Móvel é formado pelos 10 componentes que estão descritos abaixo:

- **CAOS:** Responsável por sincronizar a inicialização do restante dos componentes pertencentes ao lado móvel;

Figura 2 – Arquitetura do CAOS



Fonte – (GOMES *et al.*, 2017)

- **Offloading Monitor:** Responsável pelo monitoramento da execução dos aplicativos e por interceptar métodos marcados com notações em Java toda vez que algum deles é chamado. Essa marcação pode ser feita pelo desenvolvedor para indicar que determinado método deve ser executado pelo CAOS;
- **Offloading Client:** Responsável por iniciar o processo de *offloading* do método e delegar o método e seus parâmetros para serem executados do lado do servidor na Nuvem;
- **Discovery Client:** Responsável por descobrir servidores executando de forma local (*e.g.*, *cloudlets*) através de mecanismos de comunicação baseado no protocolo UDP/Multicast;
- **Authentication Client:** Responsável pela autenticação do aplicativo e por enviar os dados do dispositivo móvel para o lado servidor a fim de manter uma lista de dispositivos conectados ao controlador específico do CAOS;
- **Profile Monitor:** Responsável por fazer o monitoramento do ambiente do dispositivo móvel (*e.g.*, *largura de banda da rede e latência, status de energia e memória*) e enviar esses dados para o Profile Service;
- **Offloading Reasoner Client:** Responsável por auxiliar a decisão de fazer ou não o *offloading* usando uma estrutura de dados para a tomada de decisão recebida de forma assíncrona

do *Offloading Reasoner Service*;

- **Context-Acquisition Middleware:** Responsável por gerenciar o ciclo de vida dos sensores baseados em software, ou seja, esse componente obtém as informações contextuais para o CAOS;
- **Tuple-Space Based Context-Bus:** Responsável pelo armazenamento e gerenciamento das informações contextuais em formato de tupla e fornece essas informações à aplicação;
- **Context Synchronizer:** Responsável por realizar a migração dos dados contextuais do dispositivo para a Nuvem.

Já o lado da Nuvem é formado pelos 11 componentes que estão descritos abaixo:

- **Discovery Service:** Responsável por fornecer informações corretas sobre os terminais para que os clientes possam acessar os serviços do CAOS;
- **Profile Services:** Possui um conjunto de serviços responsáveis por receber os dados que são monitorados dos dispositivos relacionados com qualidade da rede e tempo de execução dos métodos local dos métodos que podem ser transferidos com o objetivo de manter um histórico de métodos executados para montar a árvore de decisão na operação de *offloading*;
- **Authentication Service:** Responsável por salvar as informações do dispositivo e controlar quais dispositivo estão atualmente conectados no CAOS Controller;
- **Offloading Reasoner Service:** Responsável pelo processamento de dados do perfil do dispositivo móvel e pela criação da árvore de decisão;
- **Offloading Service:** Responsável por receber as solicitações advindas do *Offloading Client* redirecioná-las para o *VM Pool Manager*;
- **VM Pool Manager:** Responsável por fornecer um ambiente que redireciona solicitações de *offloading* para uma máquina virtual Android adequada, que é onde a execução acontece;
- **VM Apk Manager:** Responsável por enviar todos os arquivos APK para todas as máquinas virtuais Android acessíveis;
- **VM Status Manager:** Responsável por monitorar e manter informações (*e.g., o número atual de execuções de offloading*) sobre cada máquina virtual gerenciada pelo *VM Pool Manager* em um repositório;
- **VM Offloading Executor:** Responsável por solicitar a execução de *offloading* em máquinas virtuais Android que chama o componente *Offloading Method Invocation Service*;

- ***Offloading Method Invocation Service***: Responsável por executar o método requisitado para o *offloading*.
- ***Context Service***: Responsável por atuar como um repositório de contexto global que armazena todos os dados de informações de contexto enviados de todos os dispositivos móveis conectados aos Serviços do CAOS;

O CAOS inicialmente não possuía nenhum mecanismo de segurança para garantir a confidencialidade e integridade dos dados trafegados no processo de *offloading*. O trabalho de Silva (2019) e Gomes *et al.* (2020) consiste de uma análise do impacto da adição de segurança na operação de *offloading* utilizando o CAOS. O autor fornece uma extensão do framework CAOS com a adição de mais dois componentes que fazem o uso de algoritmos criptográficos para manter a segurança dos dados.

Um componente foi adicionado do lado Móvel da plataforma e outro foi adicionado do lado da Nuvem. Esses módulos fazem o uso de uma abordagem de criptografia híbrida, com a utilização de algoritmos de criptografia simétricos e assimétricos. Então, o trabalho do autor permite que utilizando essa nova arquitetura, o objeto de dados que será transferido no processo de *offloading* possa ter um mecanismo que preserve a privacidade dos dados do dispositivo móvel, e conseqüentemente, de seu usuário.

Com o trabalho desenvolvido pelo autor, foi possível prover segurança aos dados trafegados no processo de *offloading* utilizando o CAOS com a adição dos módulos citados, e além disso, foi constatado que o uso da criptografia tem uma influência quase nula em relação ao tempo de processamento da operação, e que o tamanho do dado que será trafegado possui influência maior em relação ao desempenho da operação de *offloading*.

2.5 Segurança da Informação

Com a evolução da Internet e das redes móveis, se tornou possível o surgimento de diversas novas tecnologias, o aprimoramento de áreas e tecnologias já existentes e o desenvolvimento de dispositivos cada vez mais modernos facilitando a vida das pessoas. No entanto, mesmo com todo esse avanço da computação, problemas relacionados com a segurança da informação e privacidade de dados ainda estão presentes principalmente nas redes e dispositivos móveis.

Bine *et al.* (2016) afirma que a portabilidade trazida pelas redes e dispositivos móveis foi satisfatória, mas juntamente com essas vantagens vieram diversas vulnerabilidades atreladas a aplicações e pessoas que fazem o uso dessas tecnologias. Além disso, o autor salienta que a segurança no contexto de dispositivos móveis está cada vez mais complexa, visto que a mesma informação pode estar ser salva e compartilhada em diferentes locais. (*e.g., e-mail, pendrives e dispositivos móveis*)

De acordo com Vijayarani e Tamilarasi (2011), o avanço da tecnologia permitiu que as organizações/empresas de diferentes ramos coletem e armazenem uma grande quantidade de dados pessoais sensíveis (*e.g., CPF e RG*). Além disso, algumas empresas liberam esses dados para análise em mineração de dados, e, com isso, colocam em risco a privacidade das informações dos usuários. Outro ponto é que, mesmo as aplicações comuns lidam com dados sensíveis das pessoas, logo, essas aplicações devem fazer o uso de mecanismos de segurança.

Nas seções anteriores, foi possível perceber limitações em relação a falta recursos computacionais por parte dos dispositivos móveis, e, como medida para contornar essa situação, foi apresentado a *Mobile Cloud Computing* (Seção 2.4) e a técnica de *offloading* (Seção 2.4.1). Porém, essa abordagem ainda apresenta problemas de segurança devido ao uso de redes sem fio na transmissão de dados do dispositivo móvel para a Nuvem. Segundo Andrade *et al.* (2008), a transferência desses dados de forma desprotegida faz com que os dados fiquem sujeitos a ataques como *Man in the Middle*, que consiste no roubo de informações de usuários por meio do sequestro de uma conexão TCP.

Para garantir que os dados das pessoas possam ser preservados, trabalhos na área de Segurança da Informação como os de Pinto e Gomes (2012) e Hussein *et al.* (2016), afirmam que para manter a privacidade desses dados, provedores na Nuvem ou empresas que possuem os mesmos, devem seguir os seguintes princípios de segurança:

- **Confidencialidade:** Sistemas não devem permitir o acesso de pessoas não autorizadas as informações manipuladas por ele;
- **Integridade:** Sistemas devem possuir a capacidade de que informações não possam ser alteradas de forma indevida, seja no momento que são armazenadas, trafegadas ou produzidas;

- **Disponibilidade:** Sistemas devem possuir a capacidade de informações estarem sempre disponíveis a qualquer momento para pessoas autorizadas;
- **Autenticidade:** Pessoas autorizadas são as únicas que podem enviar ou criar informações;
- **Não repúdio/Irretratibilidade:** Pessoas que participam de determinada ação não podem omitir sua participação nessa determinada ação;

Um dos métodos que é bastante utilizado para proteção de dados para esses ambientes em rede é através de criptografia de dados. A criptografia transforma dados legíveis em dados sem sentido, e a partir desses dados sem sentido, é possível realizar a recuperação dos dados originais. Normalmente isso é feito através da utilização de algoritmos de criptografia, que são basicamente programas de computador que implementam a ideia do método de criptografia escolhido (BURNETT; PAINE, 2002).

Silva (2019) define criptografia como uma técnica utilizada para garantir segurança na comunicação que funciona da seguinte forma: é utilizado uma função de encriptação que recebe uma chave e uma mensagem, a mensagem é transformada em um código criptografado utilizando como base de encriptação a chave fornecida. Se a mensagem encriptada for interceptada por algum atacante, não poderá ser compreendida pelo mesmo, e para ter acesso ao conteúdo original da mensagem é necessário que o receptor da mensagem tenha acesso a chave de criptografia e faça o processo de decifração.

Dentre os tipos de criptografia existentes, podemos considerar dois grupos: técnicas de criptografia mais tradicionais, como simétrica e assimétrica. E técnicas de criptografia mais avançadas, como criptografia homomórfica e estenografia. Esses conceitos serão tratados nos tópicos seguintes.

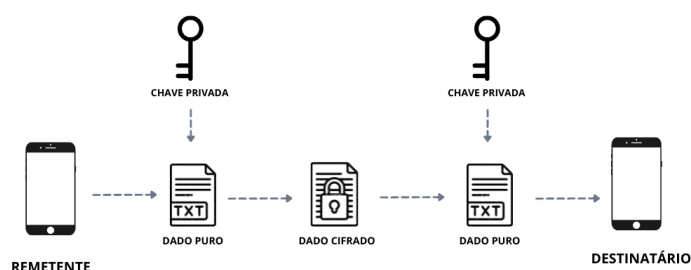
2.6 Técnicas de Criptografia

Os dois tipos de técnicas que serão apresentados a seguir são basicamente os modelos pioneiros quando se fala em criptografia, e foram a partir deles que as técnicas mais avançadas, que serão explanadas posteriormente, puderam surgir e possuir suas características únicas.

2.6.1 Criptografia Simétrica ou Chave Privada

Segundo Oliveira (2012), essa é a mais antiga técnica de criptografia e utiliza apenas uma chave tanto para encriptar quanto para decriptar dados. Essa técnica possui o seguinte funcionamento: para a troca de mensagens secretas, o remetente e destinatário compartilham de uma mesma chave (simétrica), e normalmente essa chave é um código ou uma senha que os dois usuários conhecem e devem manter em segredo para garantir a privacidade da troca de mensagens. Essa chave é o parâmetro que será passado para a função que vai transformar o texto puro em um texto cifrado (ilegível). A Figura 3 representa esse processo.

Figura 3 – Criptografia Simétrica



Fonte – Próprio Autor

Sua principal vantagem é a simplicidade por trás da sua implementação, é fácil de usar e possui um processamento rápido para a criptografia dos dados. Oliveira (2012) afirma que mesmo que as chaves utilizadas sejam complexas, o algoritmo continua tendo uma fácil elaboração. Entretanto, uma chave mais complexa não impede que a mesma seja interceptada no momento do compartilhamento entre usuários, porém, a utilização dessa técnica de criptografia é considerada importante, pois quanto mais simples o algoritmo, mais fácil será obter ganhos em relação a velocidade de processamento e facilidade de implementação, além de conseguir garantir a confidencialidade.

Ainda no trabalho de Oliveira (2012), o autor salienta que a principal desvantagem desse método é utilizar apenas uma chave para ciframento e deciframento. Como é preciso que essa chave seja compartilhada entre origem e destino, é necessário que esse compartilhamento seja feito através de um canal seguro antes mesmo da criação do canal que vai realizar a comunicação de mensagens criptografadas entre remetente e destinatário, pois a partir do momento que essa chave é interceptada, será possível ter acesso ao conteúdo original.

Outros problemas citados pelo autor são: a Criptografia Simétrica ou de Chave Privada não garante os princípios de autenticidade e não repúdio citados anteriormente. Outro problema está relacionado com o gerenciamento de chaves, pois a partir do momento que uma rede possui muitos usuários e cada par de usuário precisa de uma chave para a comunicação segura, o número de chaves cresce de forma quadrática em relação ao número de usuários, ou seja, supondo que uma rede possui n usuários, serão necessárias n^2 chaves para serem gerenciadas.

Abaixo estão descritos detalhadamente os principais algoritmos que utilizam esse método de criptografia (Oliveira (2012)):

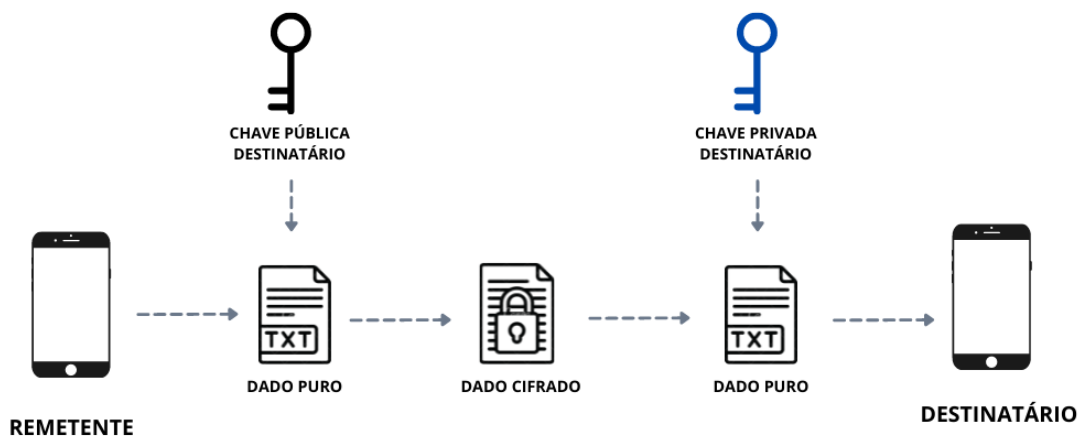
- **DES:** O *Data Encryption Standard* é um algoritmo simétrico e foi criado pela IBM em 1977. Possui um tamanho de chave de 56 bits (considerado pequeno), mas mesmo assim consegue realizar cerca de 72 quadrilhões de combinações. Após alguns anos, esse algoritmo foi quebrado através de um desafio lançado pelo *National Institute of Standards and Technology* (NIST);
- **3DES:** É considerado uma variação do *Data Encryption Standard* (DES), a diferença é que este utiliza três ciframentos sucessivos. É considerado um algoritmo muito lento, porém é seguro;
- **AES:** O *Data Encryption Standard* é um algoritmo de chave simétrica e que realiza cifração por bloco. Possui um tamanho fixo de bloco de 128 bits e uma chave que pode ser de 128, 192 ou 256 bits. Requer pouca memória, é fácil para utilizar e foi escolhido como o substituto do DES;
- **Blowfish:** Algoritmo desenvolvido por Bruce Schneier, que oferece a escolha, entre maior segurança ou desempenho através de chaves de tamanho variável. O autor aperfeiçoou o no *Twofish*;
- **Twofish:** O *Twofish* é uma chave simétrica que emprega a cifra de bloco de 128 bits, utilizando chaves de tamanhos variáveis, podendo ser de 128, 192 ou 256 bits. Ele realiza 16 interações durante a criptografia, sendo um algoritmo bastante veloz;
- **RC2:** um algoritmo voltado para utilização de criptografia em e-mails corporativos, além disso o mesmo possui tamanho de chave variável e foi projetado por Ron Rivest que também produziu outras variações, como o RC4, RC5 e RC6; e
- **CAST:** Da mesma maneira que o *Advanced Encryption Standard* (AES), este é um algoritmo que utiliza cifra por bloco, em que o bloco possui um tamanho de 64 bits e

possui um tamanho de chave variável de 40 a 128 bits. Além disso, realiza de 12 a 16 interações, ficando como padrão as 16 interações a partir de chaves que possuem no mínimo 80 bits.

2.6.2 Criptografia Assimétrica ou Chave Pública

Essa técnica de criptografia, diferentemente da anterior, faz o uso de um par de chaves diferentes (assimétricas) por usuário, uma pública e outra privada para realizar o procedimento de encriptar e decriptar dados, e além disso, essas chaves não precisam ser necessariamente senhas ou códigos, podendo se estender a arquivos digitais mais complexos (OLIVEIRA, 2012). A chave pública deve ficar disponível para qualquer pessoa que deseja estabelecer uma conexão segura. Já a chave privada, é necessário que apenas seu próprio usuário tenha conhecimento sobre a mesma, devido ao fato de que é através desta que o usuário poderá realizar o processo de decriptar a mensagem. A Figura 4 representa esse funcionamento.

Figura 4 – Criptografia Assimétrica



Fonte – Próprio Autor

Para entender o conceito, basta pensar num cadeado comum protegendo um determinado bem. A mensagem é este bem, e o cadeado, que pode ficar exposto, é a chave pública. Para que seja possível fazer a abertura do cadeado, é necessário que o usuário possua sua chave de abertura do mesmo, nesse caso, a chave privada, e assim poderá visualizar a mensagem (OLIVEIRA, 2012).

A principal vantagem dessa técnica é o ganho de segurança em relação à técnica anterior, uma vez que a chave privada não precisa e nem deve ser compartilhada com outras pessoas. Outra vantagem dessa criptografia é que qualquer pessoa poderá fazer o envio de mensagem secreta, pois a chave pública de qualquer usuário sempre estará disponível, dispensando o envio de chaves assim como foi apresentado no método *Criptografia Simétrica*.

Entretanto, a chave privada deve ser mantida de forma segura, pois no caso de alguém descobri-la, essa pessoa terá acesso a informação secreta que consta na mensagem, e consequentemente, o princípio de confidencialidade da mensagem será quebrado. Outra desvantagem que pode ser citada é que o tempo de processamento das mensagens utilizando a *Criptografia Assimétrica* é, na maioria das vezes, maior que o da *Criptografia Simétrica* fazendo com que seu uso seja limitado em determinados casos. Esse tempo de processamento maior acontece devido os algoritmos que utilizam essa técnica possuem uma grande complexidade empregada no reconhecimento das chaves.

Abaixo estão descritos, de forma detalhada, os principais algoritmos que utilizam esse método de criptografia (OLIVEIRA, 2012):

- **RSA:** É considerado o algoritmo de chave pública mais utilizado. Ele utiliza números primos para realização de suas operações. A princípio, o algoritmo consiste na facilidade de multiplicar dois números primos para obter um terceiro número, no entanto, o processo de recuperar os dois primos a partir daquele terceiro número é bastante difícil. Assim, quanto maiores forem esses números, maior a dificuldade para fatorá-los e, consequentemente, aumenta a segurança;
- **ElGamal:** É outro algoritmo de chave pública, assim como o *Rivest-Shamir-Adleman* (RSA), mas que a matemática utilizada por esses algoritmos são diferentes. Nesse caso, o ElGamal possui sua segurança baseado no problema de calcular logaritmos discretos, além de envolver a manipulação de grandes quantidades numéricas;
- **Diffie-Hellman:** Assim como o ElGamal, esse é outro algoritmo de chave pública que possui sua segurança baseado na dificuldade de resolver o problema de logaritmo discreto, além de ser considerado o algoritmo mais antigo em uso. Um fato interessante é que foram os criadores desse algoritmo que introduziram o conceito de criptografia de chave pública no ano de 1976; e

- **Curvas Elípticas:** Esse tipo de sistema criptográfico consiste na modificação de outros sistemas, como por exemplo, o que já foram citados anteriormente, trocando o domínio do conjunto de número finitos por o domínio das curvas elípticas. Mesmo utilizando chaves menores, esse algoritmo possui a capacidade de prover mais segurança que os anteriores. Porém, mesmo os algoritmos mais atuais desse tipo, em linhas gerais, ainda são mais demorados que o RSA.

Diante da contextualização sobre os conceitos de criptografia Simétrica e Assimétrica e de alguns dos seus principais algoritmos de criptografia, é notória a relevância que essas áreas possuem. O próximo tópico trata dos conceitos e técnicas de criptografia Homomórfica e Esteganografia.

2.7 Técnicas de criptografia avançadas

Os dois tipos de técnicas que serão apresentados a seguir são consideradas técnicas mais avançadas quando se trata de criptografia e ocultação de dados. A criptografia Homomórfica utilizou como base os conceitos trazidos pelas técnicas de criptografia Simétrica e Assimétrica. A Esteganografia é uma técnica de ocultação de dados criada a bastante tempo e utilizada pelas civilizações mais antigas para troca de informações de forma segura.

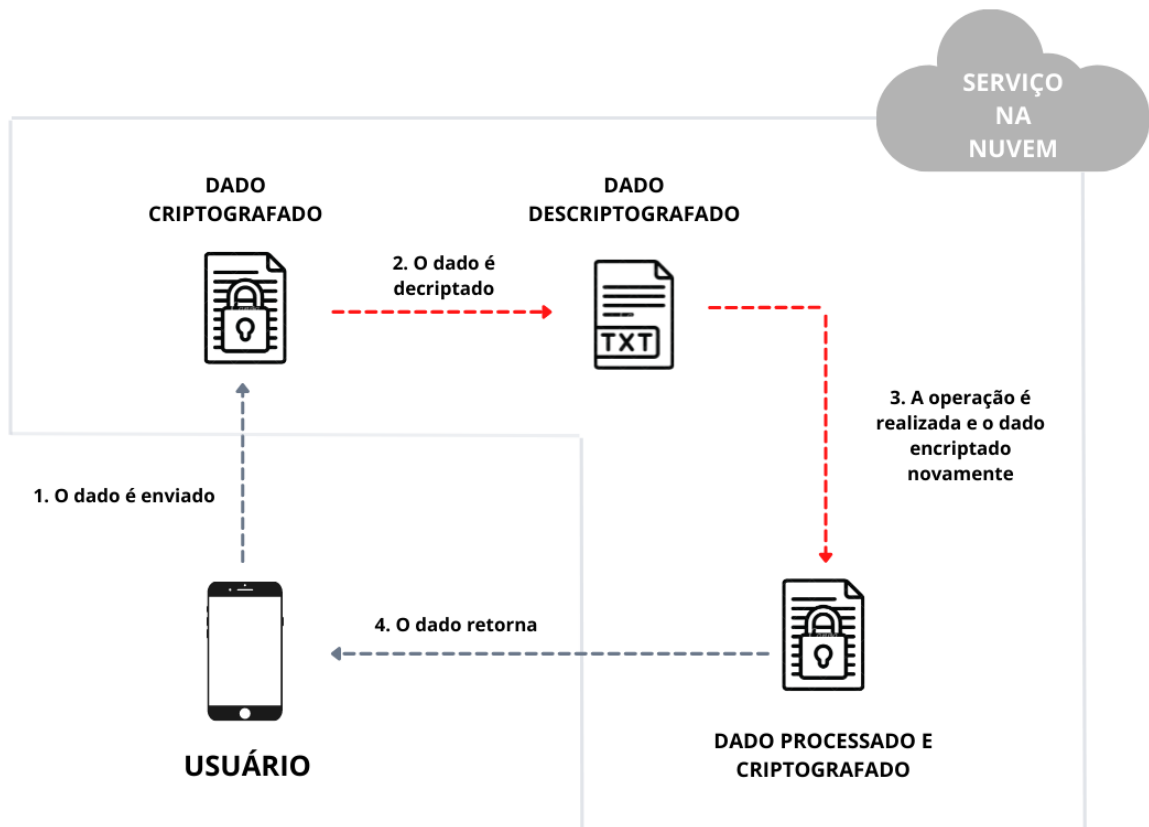
2.7.1 Criptografia Homomórfica

Os dois métodos (Chave Pública e Privada) que foram apresentados nos tópicos anteriores são utilizados para tentar prover segurança nos ambientes de *Cloud Computing*, no entanto, para realizar ações como processamento, armazenamento e manipulação de mensagens e arquivos, é necessário que o dado seja descriptografado (FILHO *et al.*, 2015), logo, o servidor em Nuvem possui acesso ao dado original, deixando o mesmo vulnerável.

O provimento de recursos sob demanda para o processamento e armazenamento massivo de dados está sujeito a: falhas de segurança, abusos com relação à privacidade, violação de direitos autorais, etc. Tais problemas são observados, principalmente, nos servidores que compõem a Nuvem, pois necessariamente os dados, que são acessados por esses servidores, precisam estar em formato original para que possa ser feito algum tipo de manipulação sobre os mesmos (FILHO *et al.*, 2015).

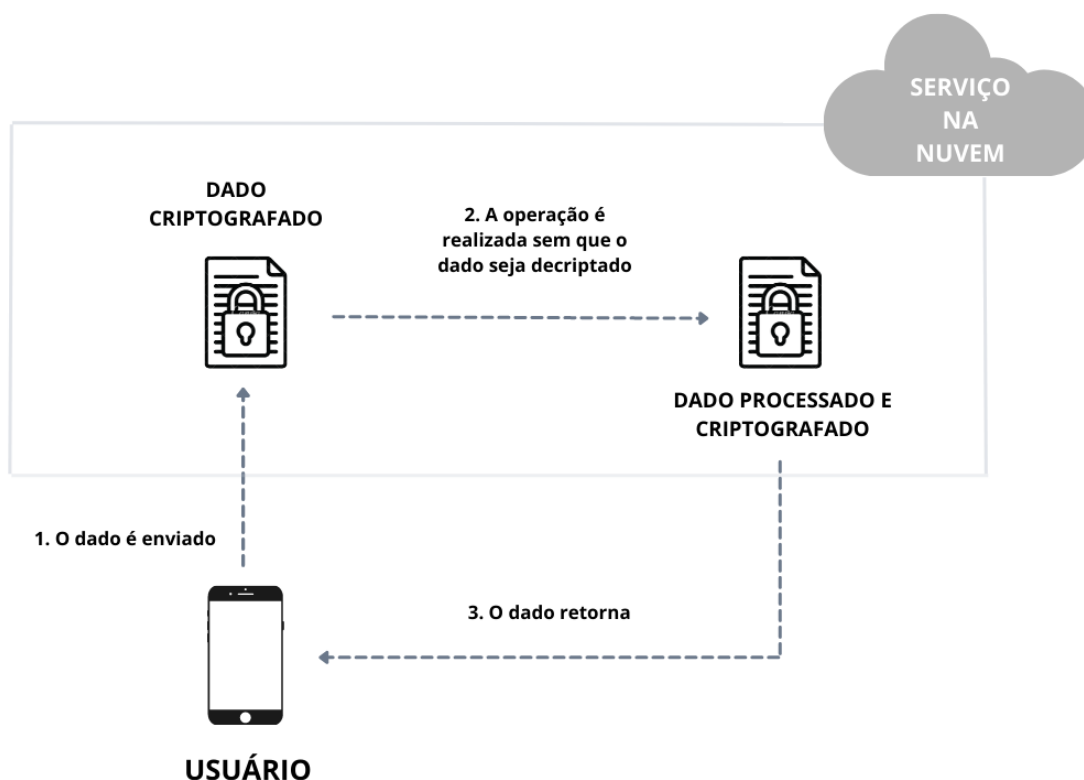
O método de Criptografia Homomórfica surge justamente com a ideia “contrária” a essa (representada pela Figura 5), possibilitando que um dado seja criptografado e enviado para o servidor, e mesmo assim, possa ser realizadas operações nesse dado sem necessidade de descriptografá-lo até que o dado retorne ao remetente, ou seja, não é preciso ter acesso ao texto original para fazer determinada manipulação. A Figura 6 representa esse processo.

Figura 5 – Criptografia não Homomórfica



Fonte – Próprio Autor

Figura 6 – Criptografia Homomórfica



Fonte – Próprio Autor

Além das operações encriptar e decriptar dados, essa técnica faz o uso de operações de adição e multiplicação. Para um entendimento melhor dessa técnica, é preciso partir primeiramente do conceito de *homomorfismo* aplicado a criptografia.

Nos trabalhos de Júnior (2013) e Junior (2018), o conceito de *homomorfismo* é definido pela composição por parte de uma função de encriptação F e dois conjuntos que podemos chamar C e E . O conjunto C é o conjunto que contém todas as possibilidades de mensagens em texto claro que serão utilizados para cifração. Já o conjunto E , é o conjunto de todos os textos cifrados possíveis para a operação de cifração. Além disso, "*" representa o conjunto das operações que podem ser aplicadas no texto simples, e "." representa o conjunto de operações que podem ser aplicadas no dado criptografado.

Logo, a função F que recebe como parâmetro uma mensagem pertencente ao conjunto C e essa função tem como saída uma cifração pertencente ao conjunto E ($F(m) = n$, com $m \in C$ e $n \in E$). Logo, para que o conceito de *homomorfismo* seja aplicado de forma correta, a seguinte condição matemática deve ser satisfeita:

$$F(m_1 * m_2) = F(m_1).F(m_2)$$

A partir dessa condição que deve ser satisfeita, é possível perceber que $F(m_1)$, $F(m_2)$ e $F(m_1 * m_2)$ pertencem ao conjunto E . A partir disso, uma operação é feita em cima dos dados originais sem a necessidade de decifrar o texto cifrado, e além disso, o resultado será um dado cifrado equivalente ao resultado original após a realização da operação desejada.

Os sistemas de Criptografia Homomórfica podem ser divididos em duas categorias: os **completamente homomórficos** e os **parcialmente homomórficos**. Junior (2018) afirma que um sistema completamente homomórfico é um sistema que possui suporte a qualquer número de operações de adição e multiplicação sobre dados. Esse primeiro tipo de sistema foi proposto por GENTRY (2009), mas, no entanto, o sistema proposto se mostrou ineficiente em relação ao tempo de processamento. Junior (2018) ainda afirma que, mesmo com novas técnicas criadas para esses sistemas, eles ainda continuam ineficientes comparados com outros sistemas da atualidade.

A maioria dos sistemas que são utilizados com propriedades homomórficas são os sistemas parcialmente homomórficos (JUNIOR, 2018). Esses tipo de sistemas, diferentemente do citado anteriormente, possui suporte a apenas uma das operações matemáticas também já citadas, o que faz com que eles sejam mais eficientes em termos de tempo de processamento. O sistema utiliza adição ou multiplicação para a criptografia e processamento dos dados. A maioria dos algoritmos que possuem as propriedades do homomorfismo são algoritmos de chave pública que serão abordados posteriormente, ou que já foram citados anteriormente.

Assim como foi mostrado nos algoritmos anteriores, logo abaixo estão listados alguns dos principais algoritmos que utilizam criptografia parcialmente homomórfica segundo a pesquisa feita no trabalho de Júnior (2013). Como até o exato momento não houve a criação de algoritmos totalmente homomórficos que sejam eficientes na prática, no presente trabalho serão mencionados apenas os que podem trazer um bom uso na prática.

2.7.2 Algoritmos de criptografia parcialmente homomórfica

- **Unpadded RSA:** É um criptosistema de chave pública com caráter determinístico. É considerado o pioneiro para o estudo de algoritmos homomórficos, pois foi a partir dele que as propriedades sobre homomorfismo puderam ser observadas. Porém, é considerado

um algoritmo extremamente frágil, apesar de ser baseado na dificuldade do logaritmo discreto e da fatoração dos naturais.

- **ElGamal:** É um criptosistema de chave pública e possui caráter probabilístico, diferentemente do algoritmo anterior. Além disso, suas características de homomorfismo estão relacionadas com sua maleabilidade. Logo, uma boa escolha de números primos e seus geradores é essencial para a segurança do algoritmo.
- **Goldwasser-Micali:** É outro algoritmo de chave pública, porém com foco no conjunto de números inteiros. Possui natureza probabilística e é considerado mais robusto que o algoritmo anterior. Este algoritmo trabalha diretamente com um conjunto específico de aspectos de Teoria dos Números.
- **Pailler:** É considerado o principal algoritmo de chave pública que utiliza homomorfismo à sua cifração, com caráter probabilístico assimétrico. A sua superioridade em quantidade de operações aplicáveis é responsável pela sua referência em toda pesquisa, tornando-se um algoritmo extremamente versátil e, conseqüentemente, muito utilizado nos diversos protocolos que envolvem o homomorfismo.
- **Damgård-Jurik:** É considerado uma generalização do criptosistema anterior, sendo assim, possui as mesmas características de segurança. A diferença é que este possui uma certa sofisticação para alguns casos específicos, e, devido a isso, a segurança do algoritmo melhora.
- **McEliece:** Também é um sistema de chave pública com caráter probabilístico, além de ser considerado o primeiro esquema a utilizar randomização no processo de cifração, diferenciando ele de todos os outros criptosistemas anteriores. É computacionalmente seguro contra um ataque quântico eficiente.

2.7.3 *Esteganografia*

Diferentemente das técnicas até aqui apresentadas, mas com o mesmo objetivo de prover segurança a dados, a esteganografia pode ser definida como a arte da escrita escondida. A palavra esteganografia é de origem grega, que significa o seguinte: **estegano** significa escondido/mascarado e **grafia** significa escrita. Essa área é composta por um conjunto de métodos e técnicas que foram utilizados e aperfeiçoados durante toda história (JULIO *et al.*, 2007)

Enquanto nas técnicas apresentadas até aqui é possível saber que existe uma mensagem, mas essa mensagem é ilegível, a esteganografia propõe a ideia de realmente esconder a mensagem de alguma forma, podendo ser através de mídias digitais como imagens ou vídeos, assim como ilustrado na Figura 7. Atualmente, os estudos nesta área estão focados na esteganografia digital, que consiste em um conjunto de técnicas e algoritmos capazes de permitir uma comunicação digital mais segura (ROCHA *et al.*, 2004).

Figura 7 – Processo de Esteganografia



Fonte – Próprio Autor

Para um melhor entendimento sobre o ocultamento de mensagens proposto por esse tipo de técnica, estão listados logo abaixo os estágios que a mensagem deve passar para se tornar segura, segundo Julio *et al.* (2007) e Rocha *et al.* (2004):

- **Dado embutido (*embedded data*):** É o dado que deve ser enviado de forma secreta, seja ele uma mensagem ou figura;
- **Mensagem de cobertura (*cover-message*):** Essa mensagem é responsável por mascarar o dado embutido. A mesma pode ser uma imagem, vídeo, áudio ou simplesmente texto;
- **Estego objeto (*stego objete*):** Esse objeto é resultante da inserção do dado embutido na mensagem de cobertura;
- **Estego chave (*stego key*):** Opcionalmente, essa chave pode ser utilizada para colocar dados do dado embutido na mensagem de cobertura;
- **Número de série digital (*fingerprinting*):** É representado por uma série de números no dado que será protegido com o objetivo de comprovar a autoria do documento.

Na pesquisa realizada por Julio *et al.* (2007), é mostrado que a esteganografia é dividida em em duas vertentes principais: a técnica e a linguística. A técnica está relacionada com o ocultamento da mensagem de forma física, assim como faziam os povos na antiguidade.

Um bom exemplo seria utilizar um pedaço de madeira para escrever algo e cobrir a informação com cera. Já a linguística se refere ao conjunto de técnicas que utilizam propriedades linguísticas (e.g., *spams, imagens e vídeos*).

Para que um sistema seja considerado esteganográfico, é imprescindível que os três requisitos listados abaixo sejam satisfeitos.

- **Segurança:** Para que a segurança possa ser garantida de forma não deixar rastros para possíveis ataques, o conteúdo deve ser oculto tanto perceptivelmente quanto por meios estatísticos. Entretanto, abordagens que venham a utilizar modelos estatísticos não se mostram eficientes em termos de recursos computacionais. Em relação a praticidade, um sistema pode ser considerado seguro, ou esteganograficamente forte, se não for possível descobrir que existe algum conteúdo oculto usando qualquer meio;
- **Carga útil:** A esteganografia deve permitir que uma carga significativa de informações possam ser embutidas, porém, dependendo do tipo de aplicação, deve-se verificar o que mais convém para seu contexto; e
- **Robustez:** É desejável que aplicações que utilizam a esteganografia tenham uma certa robustez, como por exemplo, possuir a capacidade de resistir a compressão de imagens devido ao motivo de que, na maioria das vezes, essas imagens precisam ser comprimidas antes de trafegarem na Internet.

Para essas técnicas que fazem o uso de meios digitais, os procedimentos mais comuns encontrados são através de utilização de imagens. Essas técnicas que utilizam imagens para a ocultação de dados, normalmente ocultam o dado inserindo-o no bit menos significativo da mídia, além de poder utilizar abordagens de filtragem e mascaramento, algoritmos de transformação, dentre outros. No trabalho de WAYNER (2002), é dito que o procedimento de inserção no bit menos significativo é uma das melhores abordagens quando se trata de esteganografia utilizando imagens, sendo possível ser aplicado com graus diferentes de sucesso.

Logo abaixo estão listadas algumas das principais técnicas utilizadas na esteganografia, de acordo com o trabalho de Julio *et al.* (2007):

2.7.4 Técnicas e algoritmos de esteganografia

- **LSB (*Least Significant Bit*):** Esse tipo de técnica pode ser aplicado a imagens, normalmente, que possuem 32 *bits* por *pixel*, e cada *pixel* é codificado em 4 *bytes*. A partir disso, pode ser feita a seleção de um *bit* (normalmente o menos significativo) para cada *byte* da imagem e que será o responsável por realizar o ocultamento do dado sem que haja alterações muito perceptíveis na imagem.
- **Filtragem e mascaramento:** Esse tipo de técnica é considerada mais robusta que a inserção LSB, devido as estego-imagens possuírem uma resistência maior a compressão e recorte, pois esta utiliza inserção no bit mais significativo das imagens, o que é uma ideia inversa a técnica de LSB. WAYNER (2002) afirma que técnicas de filtragem e mascaramento são mais fáceis para detecção.
- **Esteganografia em Vídeo:** A esteganografia aplicada em vídeo é bem parecida com a esteganografia aplicada em imagens, o que diferencia é que no vídeo, as informações são escondidas em cada *frame* do arquivo, ou seja, o vídeo pode ser considerado com uma sequência de imagens, e para cada imagem nessa sequência, uma informação pode ser escondida.
- **Espalhamento de espectro:** Nessa técnica, os dados escondidos são dispostos por toda a imagem de cobertura. Além disso, uma stego-chave é utilizada para selecionar randomicamente os canais de frequência pertencentes a imagem.

2.8 Considerações Finais

Esse Capítulo apresentou uma visão geral das principais áreas de conhecimento que este trabalho possui como base. Dentre as áreas é possível citar a Computação em Nuvem, Computação Móvel juntamente com dispositivos móveis e Segurança da Informação com o foco em algoritmos de criptografia. Foi exposto definições, conceitos mais básicos, características essenciais, limitações e desafios que essas áreas ainda possuem. Dessa forma, o próximo Capítulo é responsável por tratar sobre os trabalhos relacionados a presente pesquisa.

3 TRABALHOS RELACIONADOS

Este Capítulo apresenta os trabalhos relacionados à presente proposta. Estes trabalhos consistem em soluções para a segurança no processo de *offloading* computacional do dispositivo móvel para um ambiente remoto.

3.1 Estudo Comparativo do Impacto da Segurança em Ambientes de *Mobile Cloud Computing*

Silva (2019) apresenta um estudo analítico sobre o impacto causado no desempenho de aplicações para dispositivos móveis por meio de algoritmos de criptografia na operação de *offloading*, para tanto, uma métrica de tempo de processamento é utilizada. Desenvolvido em ambiente de MCC, faz a utilização de *framework* CAOS para ajudar na migração dos dados para a Nuvem.

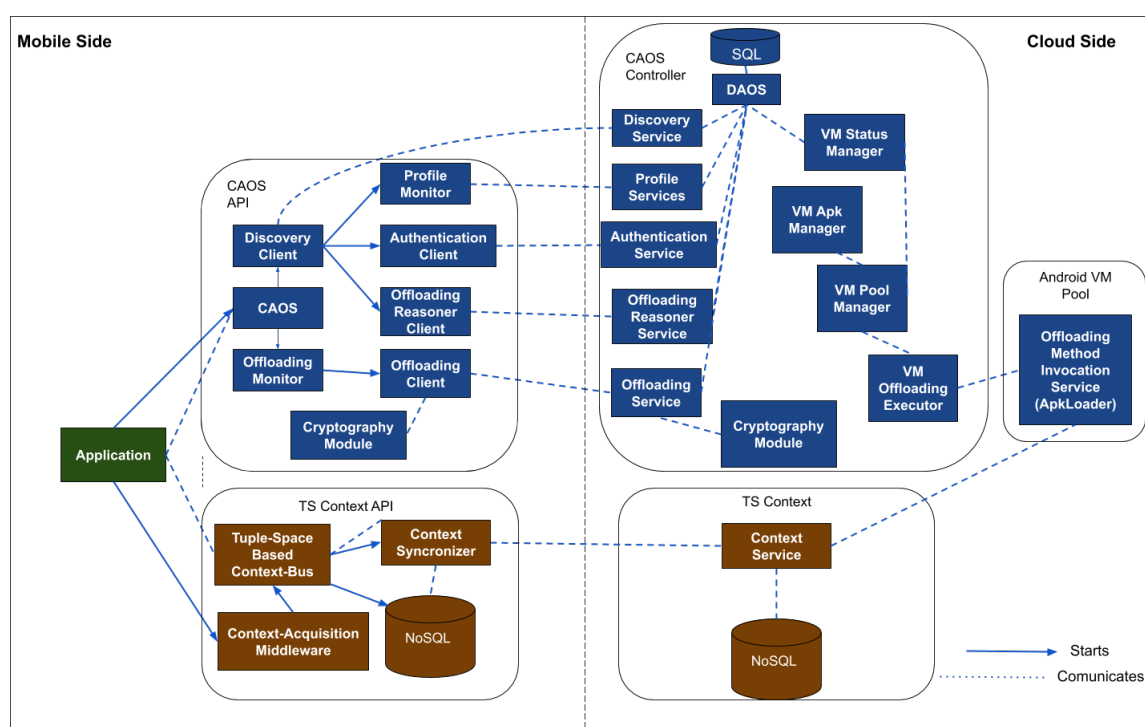
Com o objetivo de avaliar quais os impactos causados da adição de segurança (utilização de algoritmos de criptografia) em ambientes de *Mobile Cloud Computing*, um estudo comparativo foi realizado entre as estratégias utilizadas para a melhor que se adapta na resolução de problemas de segurança em MCC, mas que também não cause grandes perdas de desempenho nas operações.

A métrica de tempo de processamento utilizada está relacionada ao tempo total para a execução de uma determinada tarefa, o tempo total é composto pela diferença entre os tempos de envio da requisição e a recepção da resposta por parte do dispositivo móvel. Além da medição do tempo de processamento, é medido o tempo para encriptar e decriptar os dados, que é a soma do tempo gasto nessas operações, tanto do lado do dispositivo móvel, como do lado do servidor. Assim como, o tempo de *upload* e *download* da requisição e resposta. Todas as medições, realizadas por Silva (2019), foram feitas em milissegundos.

Como anteriormente mencionado, foi empregado o *framework* CAOS para atingir o objetivo do trabalho desenvolvido por Silva (2019), contudo o CAOS não possui nenhum mecanismo de segurança para garantir a confidencialidade e integridade dos dados trafegados no processo de *offloading*. Desta forma, um componente foi adicionado do lado móvel da plataforma e outro foi adicionado do lado da nuvem. Esses módulos fazem o uso de uma

abordagem de criptografia híbrida, com a utilização de algoritmos de criptografia simétricos e assimétricos. Então, o trabalho permite que, utilizando essa nova arquitetura, o objeto de dados em transferência no processo de *offloading* possa ser criptografado e, assim, a privacidade dos dados do dispositivo móvel e, conseqüentemente, de seu usuário seja preservada. A Figura 8 apresenta a nova arquitetura da plataforma após a adição desses módulos de segurança.

Figura 8 – Arquitetura do CAOS com suporte à criptografia



Fonte – (SILVA, 2019)

Além dessa modificação realizada no CAOS, uma aplicação de *Benchmarking* foi desenvolvida. Esse tipo de aplicação realiza operações computacionais mais complexas e possibilita medir o desempenho em relação ao tempo de execução que essas tarefas levam para serem realizadas. Nesse caso, a aplicação desenvolvida faz a ordenação de um vetor de elementos no intervalo de 0 a 1000 que são gerados aleatoriamente. Para o processo de ordenação são utilizados algoritmos de ordenação já conhecidos pela comunidade da computação, são eles: *MergeSort* e *QuickSort*.

Essa aplicação se utiliza da versão do CAOS modificada que oferece suporte a transmissão criptografada de objetos de dados. Além disso, o autor também realizou as medições de tempo e processamento utilizando a versão do CAOS anterior ao seu trabalho para fazer a comparação entre as duas versões.

Após todos os testes realizados, foi possível concluir que a utilização ou não dos algoritmos de criptografia no processo de *offloading* não foi um fator relevante no tempo gasto como todo, e que o fator decisivo para o todo o processo é o tempo de comunicação entre dispositivo móvel e Nuvem, que chega a representar 50% do tempo total gasto, logo é importantíssimo que essa conexão seja de qualidade.

Algoritmos como AES e ARC4 utilizados na pesquisa, obtiveram os melhores desempenho com um consumo de tempo total de processamento de 3%. Já algoritmos como Blowfish, DES e 3DES se mostraram menos efetivos em tempo de criptografia e a medida em que o objeto de dados crescia. Entretanto, como já foi dito, o tempo de decriptar e encriptar dados pelos algoritmos se torna insignificante comparado com o tempo de comunicação, logo, a nova versão do CAOS proposta pelo autor trás uma vantagem significativa para a segurança dos dados trafegados no processo de *offloading*.

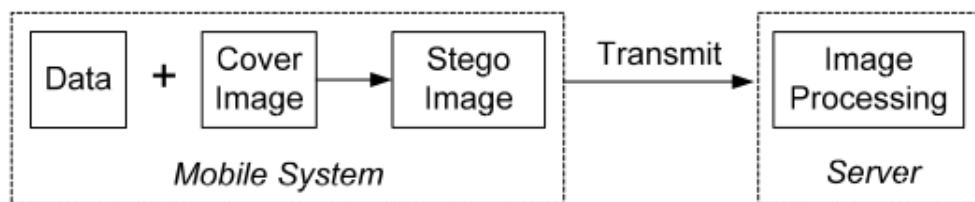
Fazendo a análise do trabalho citado, uma fragilidade notável é em relação a segurança no processo de *offloading*, pois quando os dados chegam no servidor, o mesmo precisa decriptar esses dados para poder realizar as operações necessárias, e, nesse momento, o servidor possui acesso a informação real, que muitas vezes podem ser informações pessoais que não deveriam ser de conhecimento do servidor.

3.2 Tradeoff between Energy Savings and Privacy Protection in Computation Offloading

Liu *et al.* (2010) apresentam uma implementação que utiliza técnicas de Esteganografia no processo de *offloading* computacional de imagens de dispositivos móveis para servidores remotos. Com o objetivo prover segurança aos dados e examinar o consumo energético gasto nesse processo, que é feito através da ocultamento do dado a ser enviado, ou seja, antes de enviar o dado, o mesmo é embutido dentro de uma imagem, e, além disso, o processamento do lado do servidor é feito no dado ocultado.

Os autores mencionam a utilização de técnicas mais tradicionais de esteganografia, como mostrado na Figura 9, em que os dados embutidos na imagem para ocultação precisam ser extraídos do lado do servidor. Assim, para que sejam feitas as operações necessárias, os dados que deveriam ser ocultos, são revelados no servidor. Os autores propõem um método de recuperação de imagens baseado no método de ocultação de dados por blocos.

Figura 9 – *Offloading* de Computação de Imagem Protegida por Esteganografia



Fonte – (LIU *et al.*, 2010)

A Figura 10 representa algumas formas de como dividir o valor 13 em diferentes blocos com diferentes dimensões. Esse método impõe a ideia de embutir dados em particionamentos (blocos) de um *pixel*, ao invés de ocultar o dado no *pixel* por completo, sendo isso uma medida tomada para minimizar erros. A *Cover Image* é dividida em blocos não sobrepostos com dimensões $M \times N$, assim como uma matriz, e para cada índice (linha-coluna) é ocultado um *pixel* de bloco. Supondo um valor 13 para ser embutido em uma quantidade de blocos 2×2 , 13 é dividido em $13 = 4 + 3 + 3 + 3$, em que três *pixels* são atribuídos ao valor 3 e um *pixel* é atribuído valor 4.

Além disso, é importante considerar que o método proposto pelos autores teve como base outros métodos relevantes, como o CBIR (*Content-based image retrieval*). Um método que converte imagens em representações numéricas que são chamadas recursos, e podemos considerar que esses recursos são basicamente utilizados para encontrar imagens semelhantes e que são extraídos do lado do servidor para que seja feita essa comparação de semelhança.

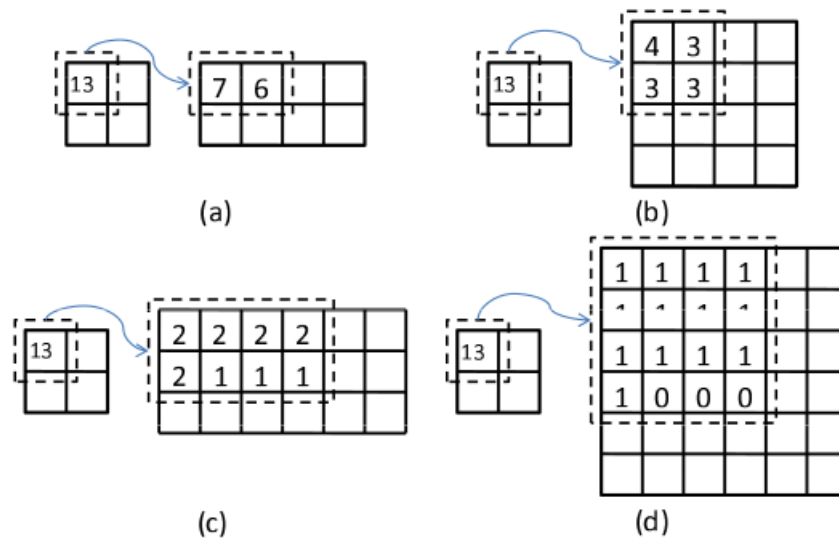
Para realizar a análise do método implementado, foram considerados os seguintes cenários:

- Executar o procedimento sem utilizar *offloading*;
- Utilizar o *offloading* para o processamento, porém sem proteção aos dados;
- Utilizar o *offloading* para o processamento juntamente com os dados protegidos;

A partir disso, foi constatado pelos os autores a seguintes afirmações:

- É possível alterar a quantidade de blocos escolhidos para manter um equilíbrio entre economia energética e segurança;
- Realizar o CBIR sem *offloading* consome mais energia do que todos os outros métodos, porém, oferece a melhor segurança, pois as imagens estão sempre no dispositivo móvel;

Figura 10 – Valor 13 ocultado em três tamanhos de blocos diferentes :



Fonte – (LIU *et al.*, 2010)

- Fazer o *offloading* com o CBIR diretamente sem ocultar os dados consome a menor quantidade de energia, mas não oferece nenhuma segurança;
- Ocultar imagens antes de fazer o *offloading* com CBIR faz com que possa ser alcançado diferentes níveis de segurança e consumo energético ao escolher diferentes tamanhos de bloco;
- Por fim, com um bloco maior, o nível de segurança aumenta, porém mais energia é consumida. Entretanto, ocultar dados em um bloco menor diminui a qualidade da imagem stego e aumenta o risco de perda de segurança

Então, com o método implementado pelos autores que utiliza uma técnica de esteganografia na operação de *offloading* com a comparação de imagens semelhantes, pode-se concluir que, a partir da utilização dessa técnica e da quantidade de blocos selecionados para a ocultação. É possível variar na quantidade de blocos utilizados de acordo com o objetivo de uso, pois é possível com a utilização de blocos maiores obter um nível maior de segurança, porém, o consumo energético também é elevado, e vice-versa.

Analisando o trabalho citado, foi possível notar que não avaliou-se o desempenho na técnica de *offloading* quando realizado através de diferentes técnicas de Esteganografia, e quanto isso influencia no tempo total da operação, a nível de processamento, comunicação e consumo de energia para todos os cenários apresentados.

3.3 *Energy Savings in Privacy-Preserving Computation Offloading with Protection by Homomorphic Encryption*

LIU Jibang; LU (2010) apresentaram uma abordagem para a economia de energia e privacidade de dados no processo de *offloading* para a operação de recuperação de imagens, mas nesse caso, os autores utilizam técnicas de criptografia homomórfica para prover a segurança dos dados trafegados entre dispositivo móvel e o servidor na Nuvem. Assim, se torna possível que as operações realizadas no servidor sejam feitas diretamente no dado criptografado, e posteriormente, sejam devolvidas para o dispositivo móvel com todas as operações necessárias realizadas sem que o servidor saiba de fato qual dado foi enviado para o mesmo.

Existem inúmeros algoritmos de recuperação de imagens, e para que os mesmos tenham suporte a privacidade de dados, é necessário que sejam feitas algumas alterações nesse algoritmo. Nesse caso, os autores consideraram dois algoritmos para serem utilizados no trabalho, são eles: ImgSeek e Gabor. Ambos são algoritmos de filtragem que consideram dois modelos de proteção de dados distintos: a Esteganografia e a criptografia Homomórfica.

ImgSeek pode ser realizado em dados esteganográficos com base no propriedade linear de extração de recursos (LIU *et al.*, 2010). Porém, ImgSeek assume que as imagens estão com as mesmas escalas e orientações. A filtragem Gabor tem melhor precisão do que ImgSeek pesquisando imagens semelhantes com objetos rotacionados, e foi a filtragem Gabor a adotada para este trabalho.

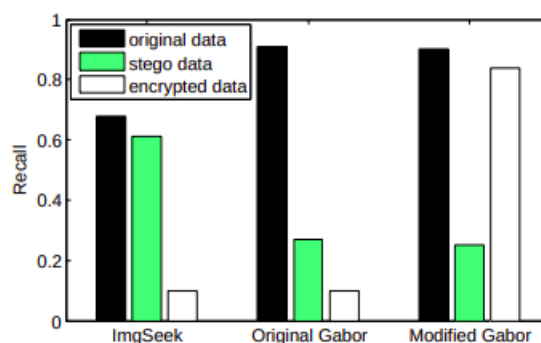
Em relação a privacidade dos dados, na solução proposta, a criptografia e a descryptografia são realizadas nos dispositivos móveis, sendo assim, o servidor só pode acessar os dados criptografados. Portanto, o invasor pode lançar ataques somente de texto cifrado e eles são a forma mais fraca de ataque. Outro fator importante é a chave utilizada para encriptação, pois quanto maior essa chave é, maior é a complexidade computacional para descobri-la. Os autores afirmam que, na abordagem desenvolvida por eles, encontrar o segredo de um número N (preferencialmente grande) é tão difícil quanto fazer a fatoração do mesmo.

Em relação ao desempenho, após comparações realizadas utilizando os algoritmos de ImgSeek, Gabor original e Gabor modificado pelos autores, temos que o ImgSeek possui desempenho inferior ao Gabor original utilizando a pesquisa em dados originais com algumas

rotações aplicadas. Quando os dados estão protegidos por esteganografia, o ImgSeek tem um velocidade de recuperação muito baixa, assim como o Gabor original utilizando esteganografia ou criptografia.

O Gabor modificado fornece desempenho próximo à pesquisa de dados desprotegidos e melhora muito o desempenho em dados criptografados. Como o método Gabor modificado funciona bem tanto com os dados originais quanto com os dados criptografados, o programa pode ser transferido para um servidor, independentemente se o usuário deseja aplicar privacidade nos dados originais, assim como mostra a Figura 11 a seguir:

Figura 11 – Comparação de desempenho de recuperação em diferentes métodos.



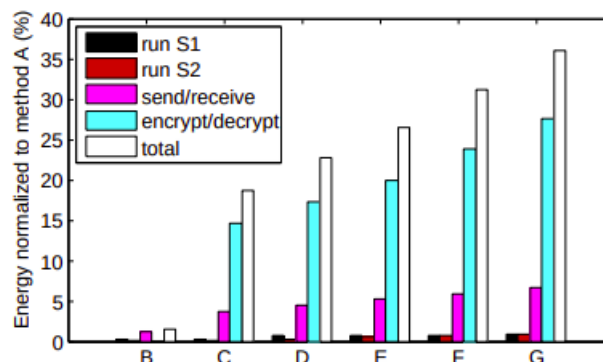
Fonte – (LIU JIBANG; LU, 2010)

Em relação ao consumo de energia, pode-se considerar os seguintes casos utilizados pelos autores e que podem ser visualizados na Figura 12:

- (A) - O programa de filtragem Gabor é executado diretamente no PDA (base para as comparações);
- (B) - O *offloading* do programa Gabor é feito para o servidor sem criptografia;
- (C) - Protege os dados usando criptografia homomórfica com chave de 64 bits;
- (D) - Protege os dados usando criptografia homomórfica com chave de 80 bits;
- (E) - Protege os dados usando criptografia homomórfica com chave de 96 bits;
- (F) - Protege os dados usando criptografia homomórfica com chave de 112 bits;
- (G) - Protege os dados usando criptografia homomórfica com chave de 128 bits;

O consumo de energia de todos esses métodos é inferior a 100% de (A), e além disso todos eles economizam energia no processo de *offloading*. O método (B) é o que consome menos energia, porém não possui proteção aos dados. O restante dos métodos possuem um consumo

Figura 12 – Consumo de energia em diferentes métodos de *offloading*.



Fonte – (LIU JIBANG; LU, 2010)

energético maior que (B) devido a adição de proteção aos dados. Outro fator importante é que conforme o tamanho da chave de criptografia cresce, o consumo energético aumenta, logo, um método baseado em um tamanho de chave maior sempre resulta em mais consumo de energia.

Analisando o trabalho citado, assim como o trabalho correlato anterior, foi possível notar que não foi realizada uma análise mais detalhada do impacto da criptografia a nível de processamento para os vários cenários apresentados pelos autores, e como a adição de segurança através dos algoritmos apresentados influencia no tempo de processamento total no processo de *offloading*.

3.4 Considerações finais

Este Capítulo apresentou os trabalhos correlatos ao presente trabalho, e, a partir disso, foi possível observar que os trabalhos de Liu *et al.* (2010) e LIU Jibang; LU (2010) apresentam problemas como a falta da análise dos impactos causados pela utilização da criptografia, e como isso influencia na aplicação como todo. Já o de Silva (2019) possui uma fragilidade que está relacionada com o servidor utilizado para processamento que possui acesso aos dados reais enviado para ele através do processo de *offloading*. Com o objetivo de prover uma solução capaz de realizar o *offloading* seguro do dispositivo móvel para o ambiente de Nuvem, mitigando os problemas encontrados nessa técnica, o próximo Capítulo apresenta a metodologia proposta do presente trabalho.

4 UTILIZANDO TÉCNICAS DE SEGURANÇA NO *OFFLOADING* COMPUTACIONAL EM AMBIENTES DE FOG COMPUTING

Os Capítulos anteriores deste trabalho apresentaram conceitos importantes para a compreensão deste trabalho, desde os fundamentos de infraestruturas como a Computação em Nuvem, Névoa e Móvel, até conceitos sobre Dispositivo Móveis, MCC (*Mobile Cloud Computing*), operação de *offloading* e conceitos de Segurança da Informação com foco em algoritmos de criptografia.

Além disso, foram expostos trabalhos relacionados a área de segurança no processo de *offloading* computacional, que, apesar de proporem técnicas que impedem a disseminação dos dados do dispositivo móvel para a Nuvem sem proteção, não avaliou-se o desempenho da operação de *offloading* quando realizado através das diferentes técnicas de Esteganografia e criptografia Homomórfica para todos os cenários apresentados.

É possível perceber problemas decorrentes das áreas de conhecimento do presente trabalho, como na área de Computação Móvel e Dispositivos Móveis (*e.g., fonte de energia, armazenamento e processamento limitados*), além de problemas de segurança trazidos pela MCC, e que são herdados da Computação em Nuvem e Móvel.

Dito isso, foi desenvolvido uma aplicação móvel que disponibiliza uma interface para que o usuário possa inserir todas os dados de entrada necessários para a execução da tarefa desejada. O processamento de desses dados será realizado na aplicação servidora hospedada em uma máquina com melhores recursos computacionais que o dispositivo móvel. Os algoritmos de criptografia serão implementados e disponibilizados aos usuários através módulos presentes nas aplicações mencionadas.

Por fim, será possível realizar uma análise para concluir quais foram os impactos causados no desempenho de aplicações para dispositivos móveis que utilizam a operação de *offloading* computacional após a adição desses algoritmos de segurança já citados.

Para medir o desempenho, serão utilizadas métricas de tempo de execução no processo de *offloading*, onde o tempo de execução é dado pela diferença entre o instante em que uma requisição foi feita pelo dispositivo móvel e o instante em que a resposta da requisição é retornada, sendo esse cálculo dado em milissegundos (ms).

Como foi dito aqui e em Seções anteriores, os dispositivos móveis possuem recursos escassos, então é totalmente necessário que seja feita essa análise para verificar a viabilidade da adição dos algoritmos de criptografia, e o quanto os mesmos podem influenciar no desempenho e tempo de processamento por parte dos dispositivos móveis, quando for necessário utilizar de segurança nessa transmissão. Os próximos tópicos tratam da metodologia que virá a ser utilizada no presente trabalho.

4.1 Metodologia

A presente pesquisa utilizará algoritmos com características semelhantes aos apresentados nos trabalhos relacionados 2 e 3, visto que os mesmos apresentam estruturas de software que tem como objetivo fornecer segurança na operação de *offloading* de dispositivos móveis para um ambiente remoto com maior poder de processamento. A Tabela 1 lista os algoritmos que serão implementados pelo presente trabalho.

Para o processo de escolha dos algoritmos informados na Tabela 1, foi considerado as características essenciais de cada um e a utilização feita pelos autores dos trabalhos relacionados de algoritmos semelhantes aos mesmos. Como o objetivo é a implementação dos algoritmos já citados anteriormente, foi feito a implementação de um algoritmo de cada técnica: Homomórfico e Esteganográfico.

O algoritmo de criptografia Homomórfica escolhido é considerado o principal algoritmo de chave pública que aplica o conceito de homomorfismo (JÚNIOR, 2013). Além disso, é muito utilizado em diversas pesquisas, e que se mostra mais eficiente em termos de tempo de processamento por utilizar uma criptografia parcialmente homomórfica (ver Seção 2.7.1). A partir disso, as operações que envolvem adições entre números podem ser realizadas usando os número totalmente cifrados, enquanto as operações que envolvem multiplicação, devem ser realizadas com um número cifrado e um número sem cifra.

O algoritmo de Esteganografia escolhido é de autoria própria, e foi desenvolvido a partir dos estudos realizados das diferentes técnicas já desenvolvidas no contexto de ocultação de dados em mídias digitais. A técnica utilizada na presente pesquisa não esconde de fato a informação desejada dentro de uma imagem, no entanto, realiza uma operação de concatenação a nível de *byte* do dado desejado com a imagem utilizada para esconder esse dado.

Tabela 1 – Algoritmos Criptográficos Escolhidos

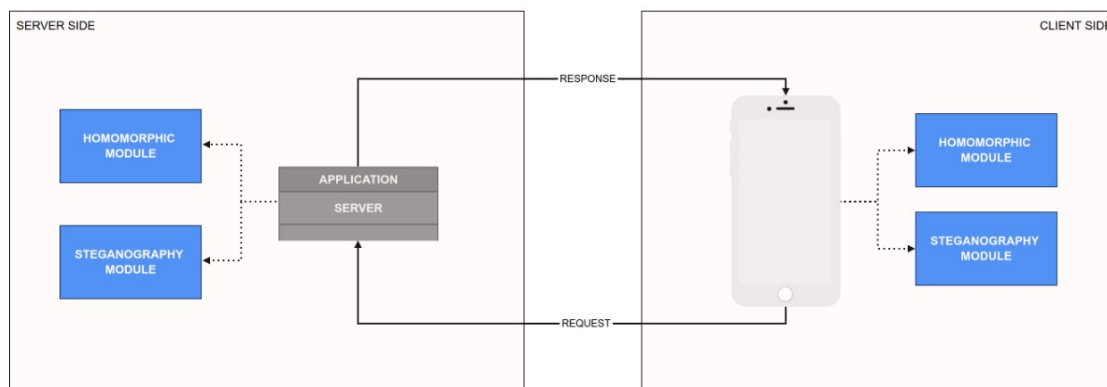
Algoritmo	Descrição	Classificação
Paillier	Trata-se do principal algoritmo de chave pública no qual se aplica Homomorfismo à sua cifração, por apresentar uma quantidade maior de operações aplicáveis. A sua superioridade em quantidade de operações aplicáveis é responsável pela sua referência em toda pesquisa, tornando-se um algoritmo extremamente versátil e, conseqüentemente, muito utilizado nos diversos protocolos que envolvem Homomorfismo.	Homomórfico
Próprio autor	Trata-se de um método que utiliza uma técnica semelhante ao <i>LSB (Least Significant Bit ou Bit Menos Significativo)</i> , mas que ao invés de embutir de fato uma informação dentro do bit menos significativo de uma imagem, tanto a imagem como a informação são convertidos a nível de <i>byte</i> para posteriormente ser aplicado uma operação de merge dos <i>bytes</i> da imagem com os <i>bytes</i> do dado a ser trafegado	Esteganográfico

Fonte – Próprio autor

4.2 Arquitetura

Visando alcançar o objetivo da presente pesquisa, como já foi dito anteriormente, foi realizado o desenvolvimento de dois módulos de criptografia: uma para algoritmos de criptografia Homomórfica e outro para Esteganografia. Esses módulos são responsáveis por garantir a segurança dos dados a serem trafegados no processo de *offloading* como um todo. Cada um dos módulos foi desenvolvida utilizando a linguagem de programação Java, além dos próprios algoritmos que estão presentes na Tabela 1 também possuem sua implementação utilizando essa mesma linguagem de programação. A Figura 13 apresenta a arquitetura proposta para o presente trabalho.

Figura 13 – Arquitetura Proposta



Fonte – Próprio Autor

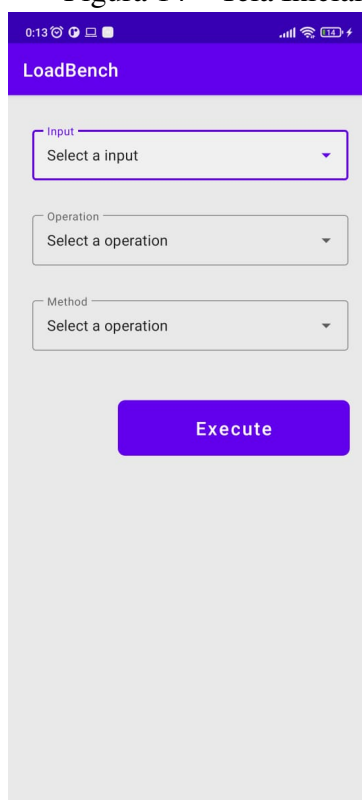
Para o uso dos recursos da criptografia parcialmente homomórfica, foi considerada a utilização de uma biblioteca externa intitulada de Javallier¹, que proporcionou para o presente trabalho um conjunto de métodos capazes de realizar a cifragem dos dados e executar operações de adição e multiplicação de números. Para os recursos da técnica de esteganografia, todo o processo foi desenvolvido pelo autor do presente trabalho, utilizando apenas recursos nativos da linguagem de programação Java e recursos da plataforma Android.

O processo de funcionamento e comunicação entre as aplicações funciona da seguinte forma: através da inicialização da aplicação móvel (exemplificada na Figura 14), será possível o usuário escolher quais os dados de entrada a serem utilizados e qual operação de proteção de dados é desejável (homomórfica ou esteganografia) e a partir disso, solicitar que seja feito a execução da operação de *offloading*. Se for estabelecido uma conexão com servidor remoto, é realizado a criptografia de todos os dados necessários com o objetivo de trafegar esses dados de forma segura.

A partir do momento que o servidor possui os dados criptografados, o mesmo pode realizar dois processos diferentes dependendo da técnica de criptografia utilizada para encriptar os dados: no caso da aplicação móvel utilizar a criptografia homomórfica, o servidor realiza o processamento necessário no próprio dado cifrado, obtendo como retorno um dado cifrado que será enviado para a aplicação móvel como resposta da solicitação, garantindo que essa resposta também será trafegada de forma segura.

¹ <https://github.com/n1analytics/javallier>

Figura 14 – Tela Inicial



The image shows a mobile application interface for 'LoadBench'. At the top, there is a status bar with the time '0:13' and various system icons. Below the status bar is a purple header with the text 'LoadBench'. The main content area is light gray and contains three dropdown menus. The first menu is labeled 'Input' and has the text 'Select a input'. The second menu is labeled 'Operation' and has the text 'Select a operation'. The third menu is labeled 'Method' and has the text 'Select a operation'. Below these menus is a blue button with the text 'Execute'.

Fonte – Próprio autor

No caso do dispositivo móvel utilizar a técnica de Esteganografia, a partir do momento que os servidor remoto possui os dados cifrados, o mesmo realiza um processo de decriptar apenas os dados necessários para realizar a operação desejada (desconsiderando os dados da imagem utilizada no processo de concatenação). Após isso, é realizado o processamento necessário no dado puro e o resultado é utilizado para realizar a operação de concatenação novamente, garantido que a resposta devolvida para a aplicação móvel, neste caso, também será trafegada de forma segura.

4.3 Considerações finais

Este Capítulo apresentou a proposta do presente trabalho que busca mitigar o problema da falta de segurança e privacidade de dados que é notável quando se utiliza o processo de migração de processamento/armazenamento de dados através da técnica de *offloading*. Para tanto, foi necessário o desenvolvimento de uma aplicação para dispositivos móveis responsável por implementar os algoritmos de criptografia já citados anteriormente e que executa tarefas computacionais complexas. Além disso, o lado servidor executa, a operação solicitada pela aplicação móvel, além de também implementar os algoritmos de criptografia citados.

5 RESULTADOS

Visando avaliar os impactos causados pela adição de técnicas de criptografia na operação de *offloading*, foi realizado o desenvolvimento de uma aplicação para dispositivos móveis chamada *LoadBench* e um servidor utilizado para realizar o processamento solicitado pela aplicação móvel¹. Estas aplicações foram utilizadas para a realização de testes, avaliando o tempo de processamento gasto com criptografia para a segurança dos dados.

Além disso, a aplicação possui a ideia de simular o cenário em que o usuário deseja executar uma tarefa que demanda muito processamento para o dispositivo móvel e que os dados em questão são sensíveis para serem expostos de qualquer forma, logo, necessita-se que seja possível garantir a privacidade dos mesmos.

A Seção 5.1 apresenta a aplicação *LoadBench*. A Seção 5.2 apresenta uma descrição do ambiente de execução dos testes. A Seção 5.3 apresenta uma descrição do experimento realizado. A Seção 5.4 apresenta os resultados obtidos com a realização dos testes.

5.1 Aplicação

LoadBench, é uma aplicação móvel que utiliza o conceito de *Benchmarking*, ou seja, uma aplicação que realiza operações computacionais complexas, onde é possível medir o desempenho relacionado ao tempo de processamento da operação solicitada/executada. *LoadBench* possui suporte para a operação de *offloading* com a utilização das técnicas de criptografia Homomórfica e de Esteganografia. A técnica de criptografia Homomórfica é utilizada a partir de uma biblioteca externa intitulada de *Javallier*², e a técnica de Esteganografia implementada com os recursos nativos da linguagem de programação Java.

As tarefas realizadas pela *LoadBench* são basicamente 3 operações matemáticas: soma de matrizes, multiplicação de matrizes e resolução da operação fatorial. Nas operações com matrizes, cada matriz é quadrada e os valores são do tipo *Integer* gerados de forma aleatória. As dimensões são 50x50, 100x100, 150x150 até 1000x1000. Para a operação de fatorial, foi escolhido, de forma empírica, como limiar inferior o número 50 e como limiar superior o número

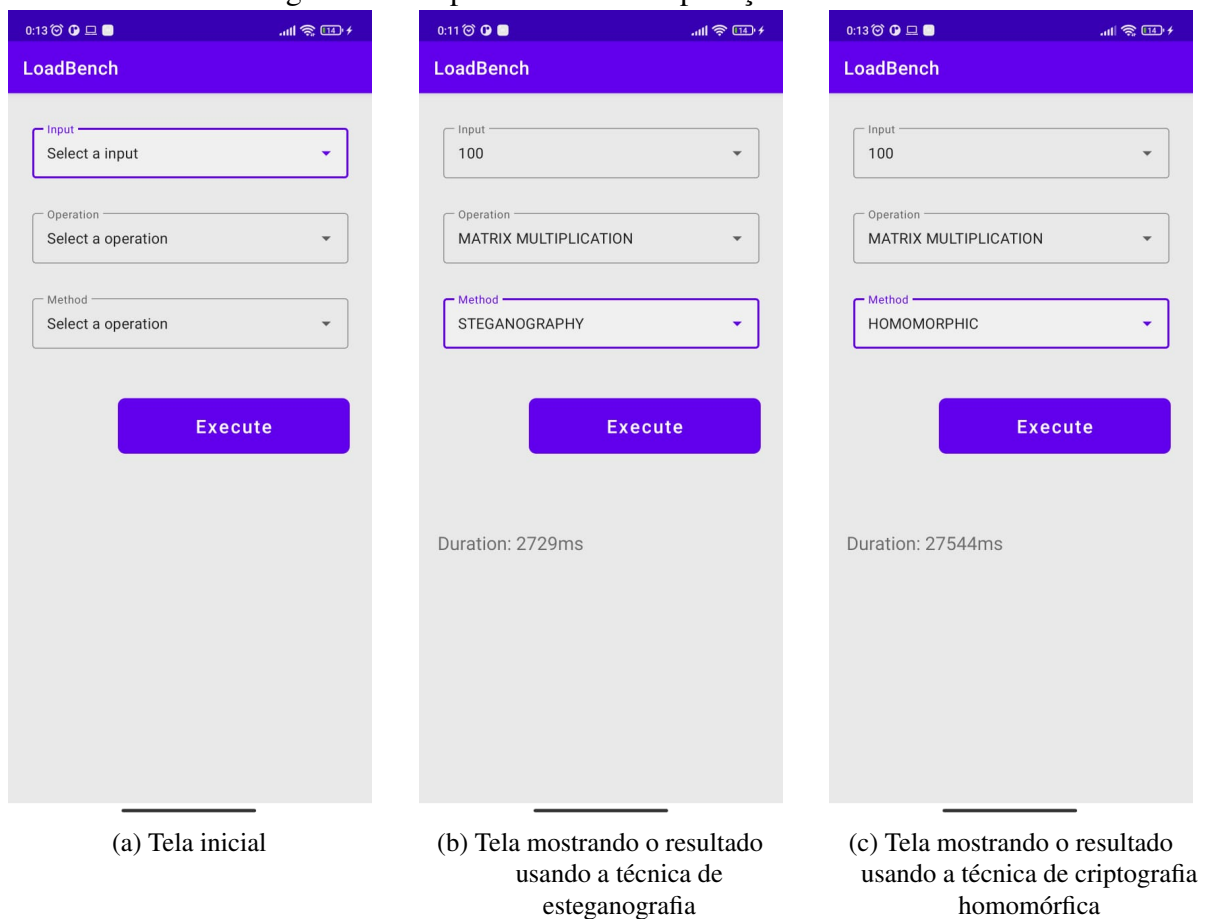
¹ <<https://github.com/henrique010/tcc-implementation>>

² <https://github.com/n1analytics/javallier>

200.

As Figuras 15a, 15b e 15c representam as capturas da tela da aplicação *LoadBench*, em que o usuário poderá selecionar um valor de entrada pré-definido, a operação matemática a ser utilizada (Multiplicação ou Adição de matrizes e Fatorial) e qual a técnica para a proteção dos dados será utilizada. Por fim, pressionando o botão EXECUTE, a geração de dados é feita do lado da aplicação móvel e o processo de executar o cálculo matemático é realizado no *cloudlet* com a técnica de *offloading* caso seja possível estabelecer a conexão com o servidor remoto, caso contrário, o processamento é realizado localmente.

Figura 15 – Capturas de tela da aplicação LoadBench



Fonte – Próprio autor

5.2 Descrição do Ambiente de Execução

Para o experimento foi utilizado um *smartphone* e um *laptop*. O *smartphone* possui as seguintes características:

- Xiaomi Redmi Note 10
- Processador de 64 *bits* Qualcomm Snapdragon 678 2.2Ghz Octa-Core
- 4GB de memória RAM
- 64GB de memória interna
- Android 11.0.0 (Google API 30)

O *laptop*, que neste experimento funciona como um *cloudlet*, tem as seguintes configurações:

- Acer Aspire A31542G
- Processador AMD Ryzen 5 3500u
- 8GB DDR4 de memória RAM
- 256 de SSD
- Placa de Vídeo Radeon 540x 2GB
- Ubuntu 20.04 LTS

5.3 Descrição do experimento

Para a realização do experimento, foi adotada a aplicação *LoadBench* para realizar a medição do desempenho dos algoritmos de criptografia quando utilizado a operação de *offloading*. Para a presente pesquisa, foi realizado a medição do tempo gasto com comunicação (*e.g.*, *upload e download*), tempo gasto com a execução da operação do lado do servidor e tempo total gasto com a operação de *offloading*. A medição do tempo de comunicação foi considerada, pois a operação de *offloading* é influenciado diretamente pela qualidade de rede entre o dispositivo móvel e servidor.

Para a medição do tempo total do *offloading*, foi realizado o cálculo a partir da marcação do instante em que a requisição realizada pela aplicação e o instante em que a aplicação obtém o resultado da execução oriunda do servidor.

O tempo gasto com *upload* foi definido como a diferença de tempo entre os momentos em que aplicação envia a solicitação ao servidor e esse a recebe. Já o *download* é o oposto, ou seja, é o período de tempo entre os momentos em que servidor envia a resposta e a aplicação a recebe. Já para o tempo gasto com a execução, foi considerado como o período entre o momento em que o servidor invoca o método a ser executado e esse devolve o resultado da computação ao

servidor.

Para a implementação da técnica de criptografia Homomórfica não foi considerada a medição do tempo de encriptar e decriptar os dados, devido a natureza de como as operações podem ser realizadas nessa técnica, ou seja, é possível realizar o processamento no próprio dado cifrado e o resultado dessa operação continua sendo um dado cifrado referente ao valor real, caso a operação fosse realizada nos dados sem criptografia. Além disso, o algoritmo de criptografia utilizado pela biblioteca *Javallier* implementa o criptossistema de *Paillier*, ou seja, é um criptossistema que utiliza criptografia parcialmente homomórfica.

Com a biblioteca *Javallier*³, as operações que envolvem adição de números, o processamento é realizado com os dois número completamente cifrados. Já para operações que envolvem multiplicação, o processamento deve ser feito com apenas um número totalmente cifrado, enquanto o outro número deve estar de forma completamente clara. Além disso, independente do uso das duas operações citadas, após ser realizado o processamento dos dados, sempre é obtido como resultado final da operação um dado cifrado.

Considerando o cenário de adição e multiplicação de matrizes, para a operação de adição de matrizes, é necessário que os valores sejam totalmente cifrados. Para a multiplicação de matrizes, é necessário que apenas uma matriz tenha seus dados totalmente cifrados, enquanto a outra matriz deve possuir seus dados sem nenhum tipo de cifra. Todo o processo de geração e encriptação das matrizes é realizado no dispositivo móvel. O objeto de dados a ser trafegado é composto pelas duas matrizes e a chave pública gerada pelo cliente, que no caso é necessária para que o servidor consiga realizar as operações de adição e multiplicação de números.

Para a implementação da técnica de Esteganografia, foi desenvolvido um algoritmo a partir dos estudos realizados das diferentes técnicas já desenvolvidas até aqui no contexto de ocultação de dados. A ideia do *LSB* (Least Significant Bit ou Bit Menos Significativo), por exemplo, é utilizar o *bit* menos significativo de uma imagem para embutir a informação desejada e, assim, poder trafegar a informação sem que seja possível saber que existem dados embutidos nessa imagem e assim evitar que algum atacante possa ter acesso aos dados.

A técnica utilizada na presente pesquisa não esconde de fato a informação desejada dentro do bit menos significativo de uma imagem, no entanto, realiza uma operação de

³ <https://github.com/n1analytics/javallier>

concatenação do dado desejado com a imagem utilizada para esconder esse dado. O processo ocorre da seguinte forma: a informação escolhida para ser trafegada é convertida para *bytes*, o mesmo acontece para a imagem selecionada para participar do processo. A partir disso, é realizada a concatenação, obtendo como resultado um novo conjunto de bytes (informação desejada + imagem).

Esse novo conjunto de *bytes* é o que de fato será trafegado no processo de *offloading*. Para que o servidor consiga extrair apenas a informação necessária para que possa ser realizado o processamento sobre o dado, é necessário que o servidor tenha conhecimento de qual imagem foi utilizada na operação de concatenação. Com o servidor possuindo conhecimento de qual imagem foi utilizada, é possível extrair somente os *bytes* da informação desejada, e, a partir disso, realizar as operações necessárias.

Considerando o cenário de adição e multiplicação de matrizes, para os dois tipos de operações matemáticas, é realizado basicamente o processo informado anteriormente. As duas matrizes utilizadas são convertidas em *bytes* e concatenadas à imagem para serem trafegadas (matriz1 + imagem e matriz2 + imagem).

No lado do servidor, o mesmo faz a extração apenas dos dados necessários para as duas matrizes, converte esses dados para o tipo em que possa realizar as operações desejadas, realiza a operação necessária (adição ou multiplicação) e obtém, como retorno, a matriz resultante da operação realizada. Após isso, os dados da matriz resultante são convertidos para *bytes*, assim como a imagem selecionada, é realizada a operação de concatenação e o resultado é devolvido para o dispositivo móvel.

Para a realização do experimento, foram escolhidos, de forma empírica 8 intervalos de dimensões de linhas e colunas para as matrizes. Inicialmente, começando com dimensões que variam sempre em cinquenta na diferença de linhas e colunas, como 50x50, 100x100, 150x150, 200x200 e 250x250 para a criptografia Homomórfica e a Esteganografia. Posteriormente, foi utilizado dimensões de linhas e colunas a partir de 500x500, seguido por 750x750 e 1000x1000. Neste segundo caso, apenas na Esteganografia foi possível a utilização de matrizes com dimensões 1000x1000, devido a Homomórfica possuir limitações relacionadas à memória para encriptação de matrizes com dimensões 1000x1000.

De acordo com o Teorema Central do Limite (TCL), quando se tem uma amostra

suficientemente grande, a distribuição de probabilidade da média amostral pode ser aproximada por uma distribuição normal. Essa aproximação é válida a partir de 30 médias amostrais. Para as dimensões que variam de 50x50 até 250x250 foi realizado a execução de testes cinquenta vezes, excluindo os dez melhores e dez piores resultados em termos de tempo de processamento, ficando como resultado amostral os trinta valores restantes.

Para as dimensões de 500x500, 750x750 e 1000x1000 foi realizado a execução de testes considerando apenas três execuções, devido às operações que utilizam criptografia Homomórfica se mostrarem muito custosas, possuindo tempo de execução variando de 53,47 minutos a 3,15 horas, sendo assim, tornando inviável a realização de testes com cinquenta repetições. No entanto, foi escolhido esse número (três) de repetições para obter ao menos dados iniciais de como possivelmente as operações se comportam em termos de tempo de processamento utilizando essas dimensões, mesmo para esse número reduzido de repetições. Todos os dados referentes a esses experimentos foram coletados e dispostos em planilhas para futura análise dos resultados.

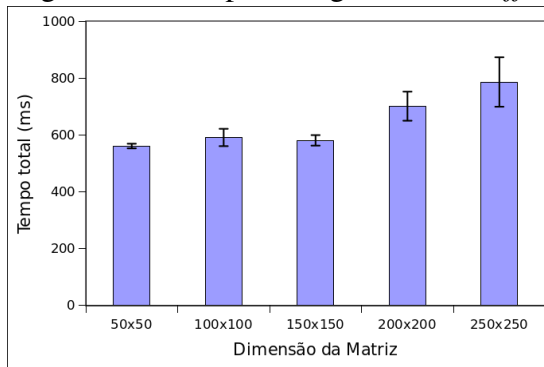
5.4 Resultados Obtidos

Para as duas técnicas utilizadas no presente trabalho, que consistem na criptografia Homomórfica e na técnica de Esteganografia, foi realizada a coleta das métricas já informadas anteriormente (e.g., custo com a criptografia dos dados, custo com a comunicação e custo com tempo total do processo de *offloading*) e realizado uma média aritmética das 30 execuções no caso das entradas que variam entre 50x50 a 250x250, e uma média aritmética das 3 execuções no caso das entradas que variam entre 500x500 a 1000x1000. Foi considerado um intervalo de confiança de 95% e o *software* Gnumeric⁴ para realizar a análise e a criação dos gráficos que serão expostos em seguida.

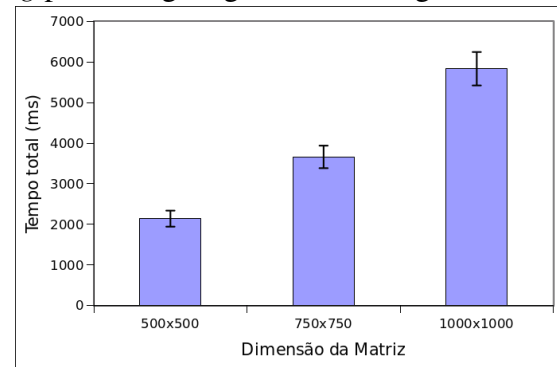
As Figuras 16 e 17 apresentam a média do tempo total gasto com o processo de *offloading* para o algoritmo de Esteganografia levando em consideração a utilização de duas imagens com tamanho diferentes (uma imagem maior com tamanho de 1.246.202 *bytes* e uma imagem menor com um tamanho de 6.103 *bytes*). A escolha de diferentes tamanhos de imagem tem como objetivo avaliar o comportamento da criptografia quando submetida a imagens menores e maiores.

⁴ <http://www.gnumeric.org/>

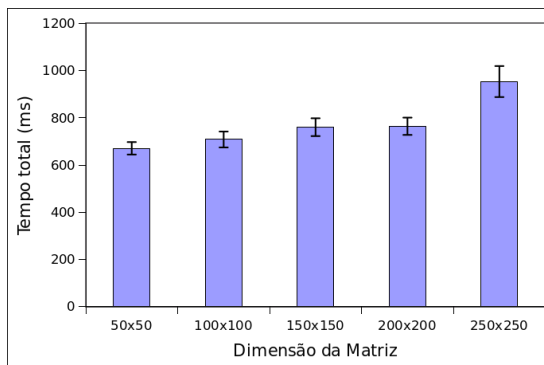
Figura 16 – Tempo total gasto com o *offloading* para Esteganografia com imagem maior



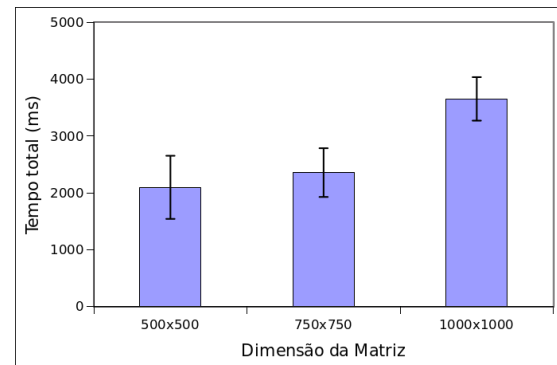
(a) Adição de matriz - 50x50 a 250x250



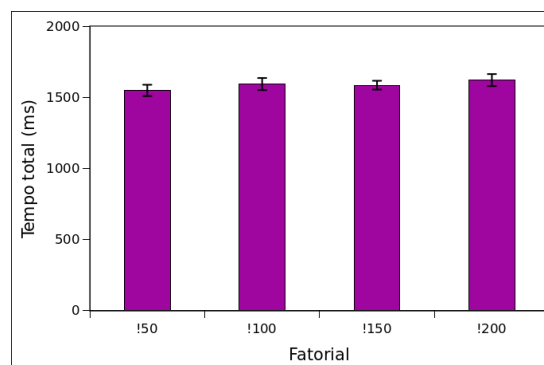
(b) Adição de matriz - 500x500 a 1000x1000



(c) Multiplicação de matriz - 50x50 a 250x250



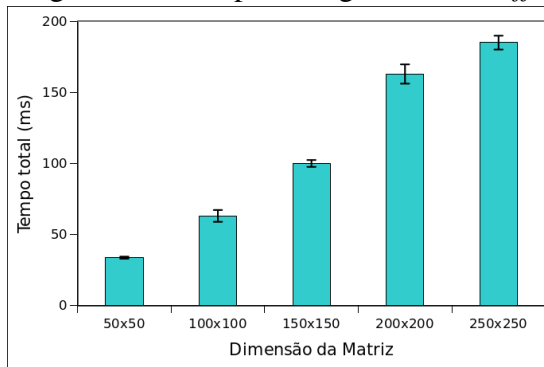
(d) Multiplicação de matriz - 500x500 a 1000x1000



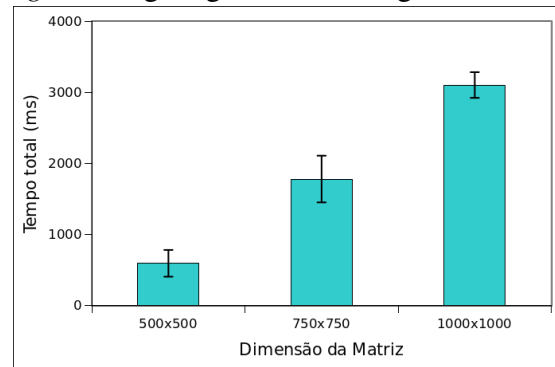
(e) Fatorial - 50 a 200

Fonte – Próprio autor

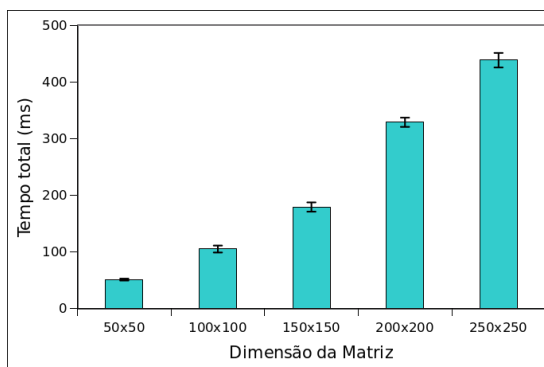
Figura 17 – Tempo total gasto com o *offloading* na Esteganografia com imagem menor



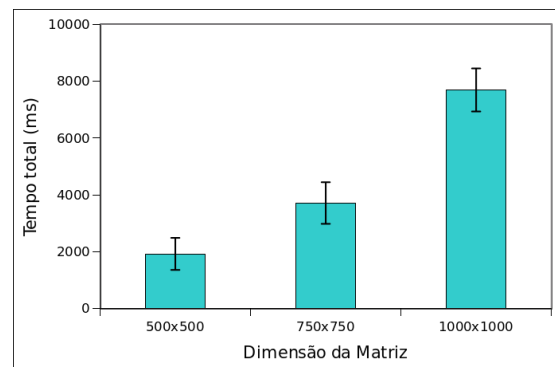
(a) Adição de matriz - 50x50 a 250x250



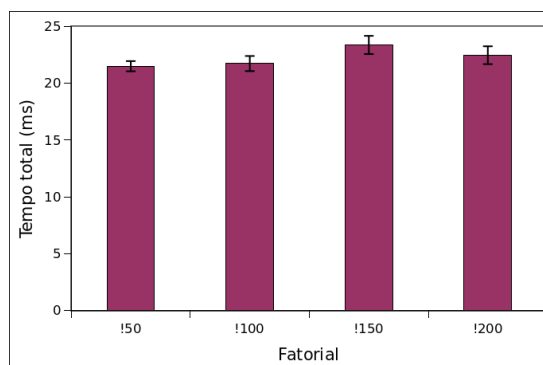
(b) Adição de matriz - 500x500 a 1000x1000



(c) Multiplicação de matriz - 50x50 a 250x250



(d) Multiplicação de matriz - 500x500 a 1000x1000

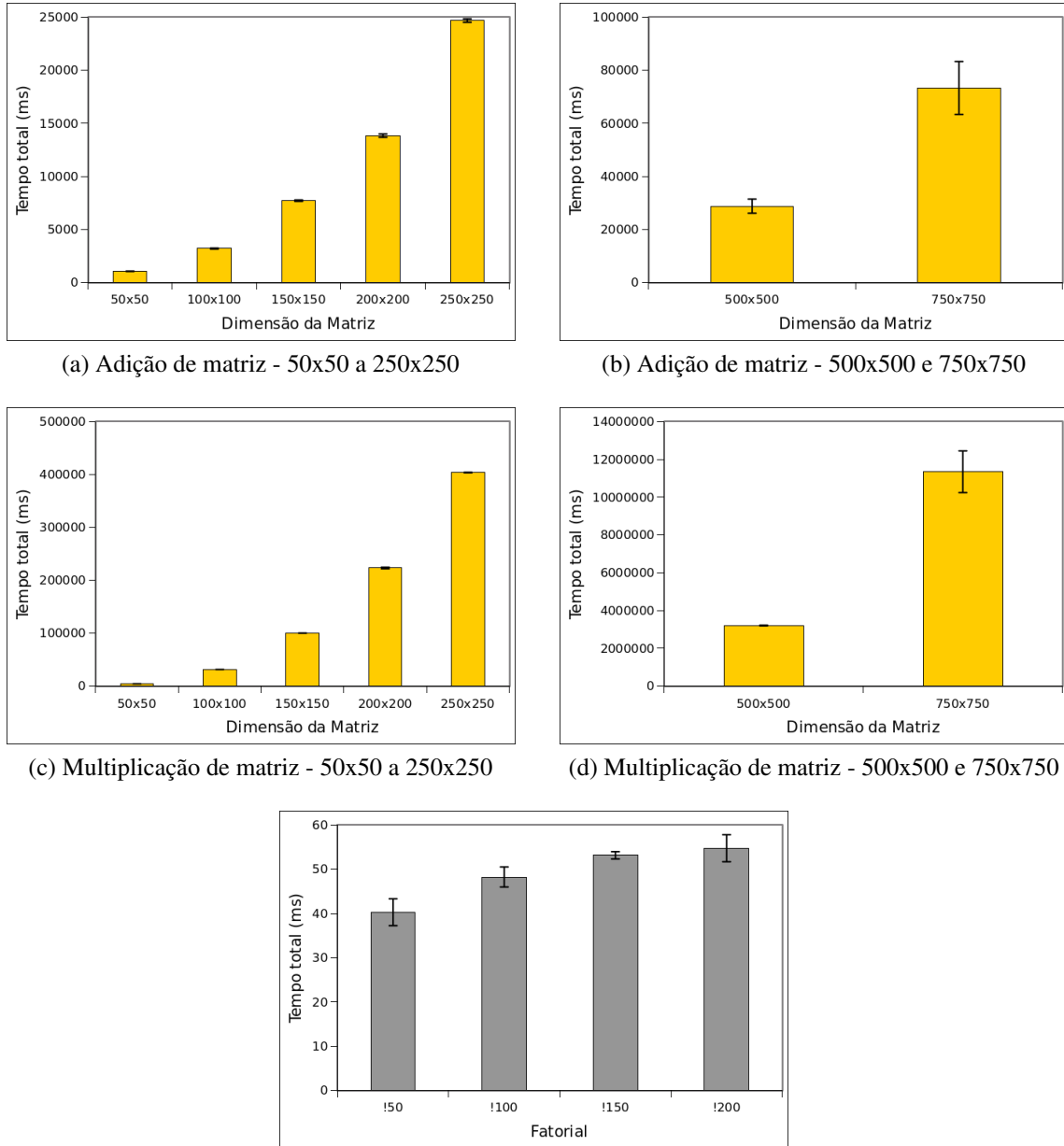


(e) Fatorial - 50 a 200

Fonte – Próprio autor

Logo em seguida, a Figura 18 apresenta a média do tempo total gasto com o processo de *offloading* para o algoritmo de criptografia Homomórfica.

Figura 18 – Tempo total gasto com o *offloading* para a criptografia Homomórfica



(e) Fatorial - 50 a 200

Fonte – Próprio autor

A partir dos gráficos presentes nas figuras mencionadas anteriormente, é possível perceber que o algoritmo que emprega a técnica de Esteganografia se mostra bem mais eficiente que o algoritmo utilizado para criptografia Homomórfica em termos gerais. Considerando a utilização das diferentes imagens para a Esteganografia, percebe-se que a utilização da imagem com tamanho maior gasta um pouco mais de tempo quando comparado ao uso da imagem menor,

o que já era esperado, pois quanto maior a imagem utilizada no processo, maior será o objeto a ser trafegado na operação de *offloading*. Isso também é aplicável para a medida em que as dimensões das matrizes ou o fatorial aumentam.

Considerando a operação de adição de matrizes, nota-se que, em ambos os casos da Esteganografia, a operação como um todo é realizada de forma bem rápida tanto para as dimensões que variam de 50x50 a 250x250 quanto para as dimensões que variam entre 500x500 a 1000x1000. O mesmo pode ser considerado para a operação de multiplicação de matrizes, que se torna ligeiramente mais custosa do que a operação de adição, entretanto, ainda apresenta um gasto médio estável considerando todas as variações utilizadas de linhas e colunas.

Para a operação de fatorial, o custo gasto com a operação de *offloading* como todo também apresenta tempo de processamento bem rápido para as variações de entrada de 50 a 200. Além disso, é perceptível que para os cenários que utilizam a imagem maior para realizar a operação mostram-se um pouco mais lentos, o que, assim como aconteceu com as operações de adição e multiplicação de matrizes, é um cenário já esperado.

A Figura 18 apresenta os dados de média do tempo total gasto com a operação de *offloading* para o algoritmo de criptografia Homomórfica. A partir disso, nota-se que os valores gerados nos testes, principalmente para as dimensões que variam entre 50x50 e 250x250, são valores tão próximos entre si que a representação do intervalo de confiança é quase imperceptível no gráfico, com uma melhor visualização quando considera-se dimensões de 500x500 e 750x750. Além disso, para a criptografia Homomórfica, não foi utilizada a dimensão de 1000x1000 devido a limitações relacionadas a consumo de memória.

Considerando a operação de adição de matrizes, é possível notar que para quase todas as dimensões utilizadas, o algoritmo de criptografia Homomórfica mostra-se bem mais custoso quando comparado com o de Esteganografia a medida em que as dimensões aumentam, com um ponto de atenção para a dimensão de 750x750 devido a mesma possuir uma média próxima a 1 minuto de execução, o que já não é considerado um tempo ideal para o cenário de aplicações para dispositivos móveis.

No entanto, a diferença maior se encontra quando é levado em consideração a multiplicação de matrizes, pois, a partir do momento em que o número de linhas e colunas aumentam, o tempo gasto na operação aumenta bastante, fazendo com o tempo de processamento,

considerando a dimensão de 250x250, tenha um custo médio de 6,7 minutos (403669,3 ms). Mas o problema maior está quando considera-se a dimensão de 500x500 e 750x750, ambas com custo médio respectivamente de 53,47 minutos (3208734,66 ms) e 3,15 horas (11346165,33 ms).

Para a operação de fatorial utilizando o algoritmo de criptografia Homomórfica é possível perceber que, diferente das duas operações anteriores, o mesmo demonstra um gasto médio estável considerando todas as variações de entrada, o que indica uma eficiência para a operação em questão. Ele é ligeiramente mais custoso do que a operação de fatorial que utiliza o algoritmo de Esteganografia, mas que, em termos gerais, possui um tempo de processamento relativamente rápido.

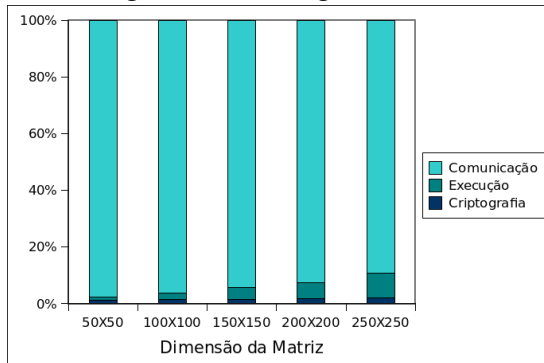
As Figuras 19, 20 e 21 apresentam o tempo total gasto com a criptografia, o tempo total gasto da execução do método matemático e o tempo total gasto com a comunicação no *offloading* utilizando os algoritmos de Esteganografia (com maior e menor imagem) e criptografia Homomórfica respectivamente. Para cada uma das figuras, foi descontado, além das métricas citadas, o tempo gasto com os *overheads* do processo como todo.

Analisando as Figuras 19 e 20, é possível notar que o tempo gasto com a comunicação se mostra bem mais influente quando utiliza-se uma imagem com um maior tamanho. Já para o cenário com uma menor imagem, é possível notar que o tempo gasto com execução da operação matemática e o tempo gasto com a criptografia se mostram mais influentes, exatamente pelo motivo do tempo de comunicação ser menor, devido ao uso de uma imagem com dimensões menores no processo.

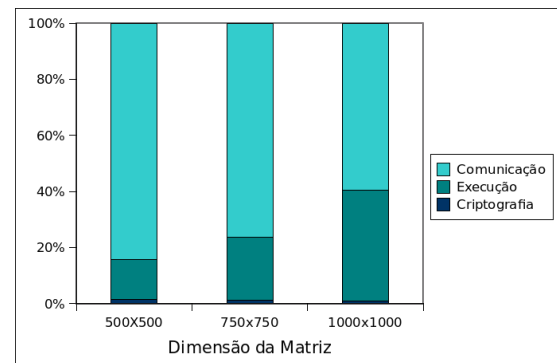
No entanto, de acordo com os gráficos, à medida em que as entradas aumentam para cada uma das operações matemáticas, o tempo gasto com a execução da operação começa a possuir uma influência maior no tempo total gasto no processo. Além disso, enquanto é possível notar, o tempo de execução se mostra mais influente no tempo total gasto, o inverso ocorre para o tempo de criptografia, pois, a medida que as entradas aumentam, o tempo gasto com criptografia se mostra cada vez menos influente.

Em relação ao tempo gasto com a criptografia, levando em consideração as duas operações matemáticas de matrizes, é perceptível que, para as dimensões que variam entre 50x50 a 250x250, o tempo gasto com a criptografia mostra uma influência quase que constante, porém muito pequena quando comparada com o tempo gasto com comunicação e execução.

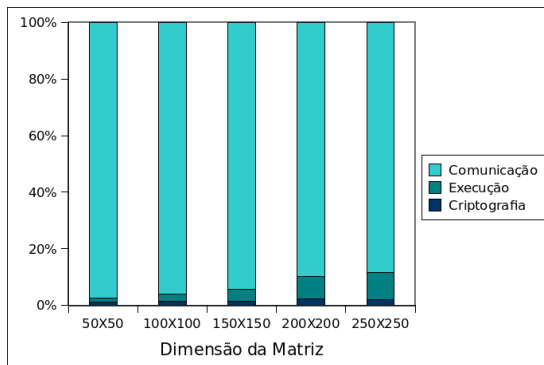
Figura 19 – Tempo de Comunicação x Tempo de Execução x Tempo de Criptografia (Esteganografia com imagem maior)



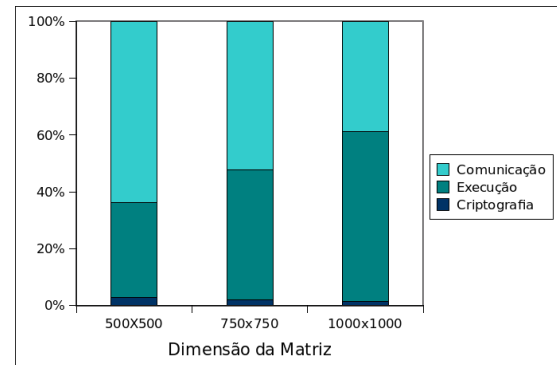
(a) Adição de matriz - 50x50 a 250x250



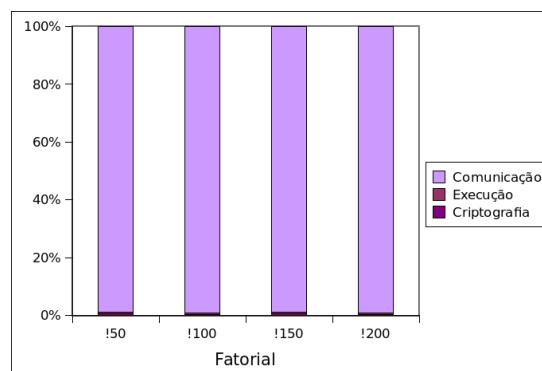
(b) Adição de matriz - 500x500 a 1000x1000



(c) Multiplicação de matriz - 50x50 a 250x250



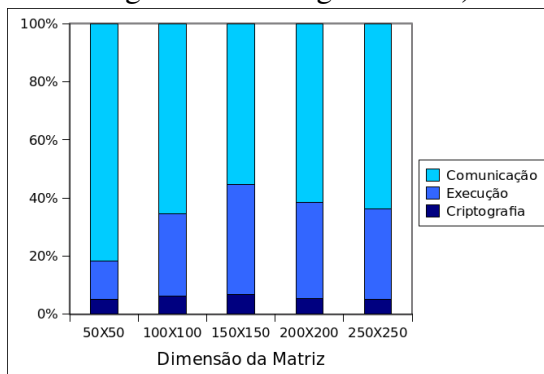
(d) Multiplicação de matriz - 500x500 a 1000x1000



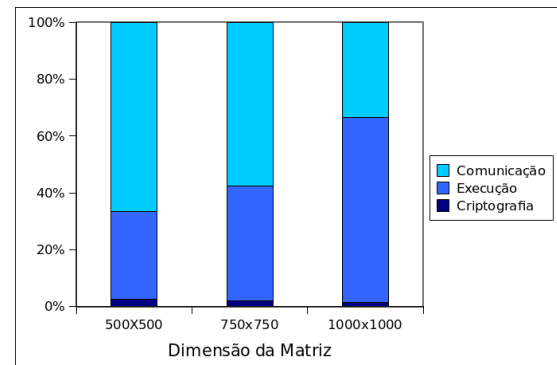
(e) Fatorial - 50 a 200

Fonte – Próprio autor

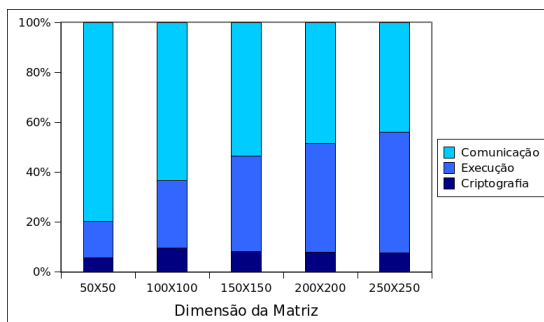
Figura 20 – Tempo de Comunicação x Tempo de Execução x Tempo de Criptografia (Esteganografia com imagem menor)



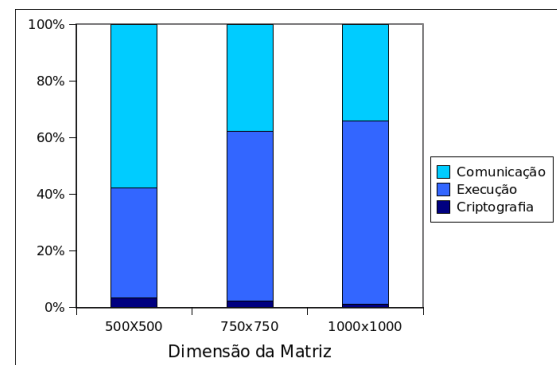
(a) Adição de matriz - 50x50 a 250x250



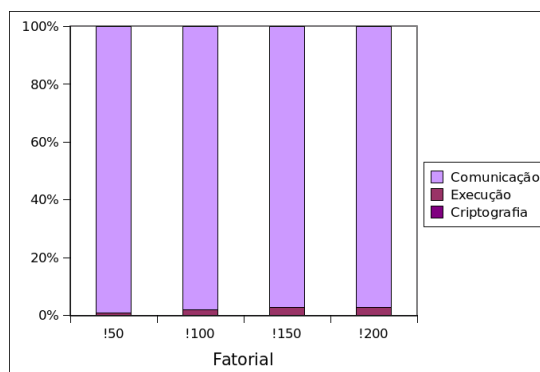
(b) Adição de matriz - 500x500 a 1000x1000



(c) Multiplicação de matriz - 50x50 a 250x250



(d) Multiplicação de matriz - 500x500 a 1000x1000



(e) Fatorial - 50 a 200

Fonte – Próprio autor

É possível perceber uma influência maior quando considera-se o cenário utilizando uma imagem menor. No entanto, considerando as operações que utilizam dimensões que variam entre 500x500 a 1000x1000, a influência do tempo gasto com a criptografia se mostra cada vez menor, chegando a percentuais próximos a 1% de influência. Analisando os cenários com a operação de Fatorial, é possível perceber que o tempo gasto com a criptografia é basicamente nulo, mostrando uma influência muito pequena para ambos os cenários com imagens que possuem o comprimento diferente.

Analisando a Figura 21 e considerando a operação de adição de matrizes, é perceptível que, a medida em que as dimensões das matrizes aumentam, o tempo de execução é ligeiramente dominado pelo tempo gasto com a comunicação, ou seja, o tempo de execução se torna cada vez menos influente no processo de *offloading* como um todo.

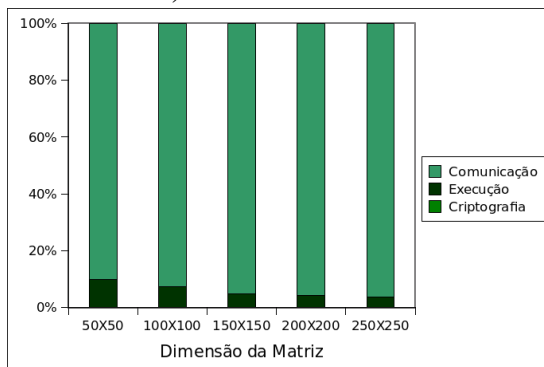
Entretanto, acontece exatamente o contrário quando considera-se a operação de multiplicação de matrizes, ficando perceptível que, nesse caso, a medida em que as dimensões das matrizes aumentam, o tempo de execução domina quase que por completo o tempo gasto com a comunicação, ou seja, o tempo gasto com a comunicação se torna cada vez menos influente no processo como todo.

Considerando a operação de Fatorial, é possível perceber que essa operação possui um comportamento similar a operação de multiplicação de matrizes, ou seja, à medida em que a entrada cresce, o tempo gasto com a execução vai dominando o tempo gasto com a comunicação. O diferencial do Fatorial para as operações com matrizes é o tempo gasto no processo como todo, que mostra uma eficiência bem maior quando realizado essa comparação em termos gerais.

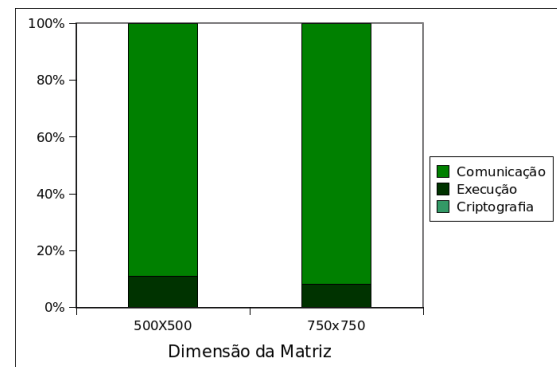
A partir de tudo que foi exposto até aqui, é possível concluir que independente da utilização de uma imagem com tamanho maior ou menor, o tempo gasto com a criptografia utilizando a técnica de Esteganografia não possui tanto impacto no processo de *offloading* como um todo, pois o impacto maior se mostra presente no tempo gasto com comunicação e na execução da operação, sendo métricas que vão variar seu nível de performance dependendo das condições do ambiente em que os testes estão sendo executados.

Já para o cenário da criptografia Homomórfica, foi possível perceber três diferentes situações que estão ligadas diretamente com a complexidade do problema a ser resolvido:

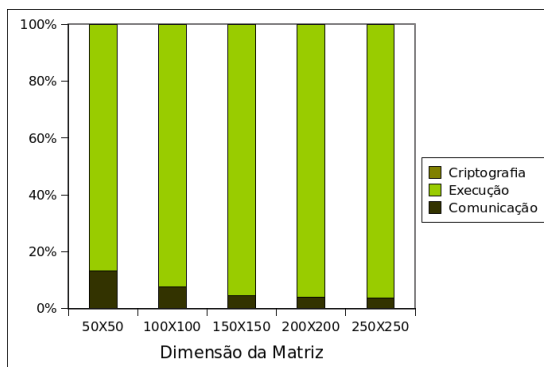
Figura 21 – Tempo de Comunicação x Tempo de Execução x Tempo de Criptografia (Homomórfica)



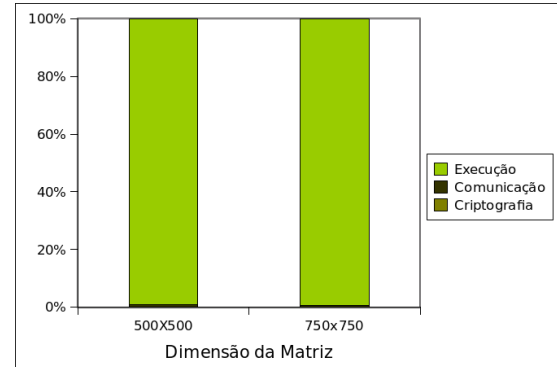
(a) Adição de matriz - 50x50 a 250x250



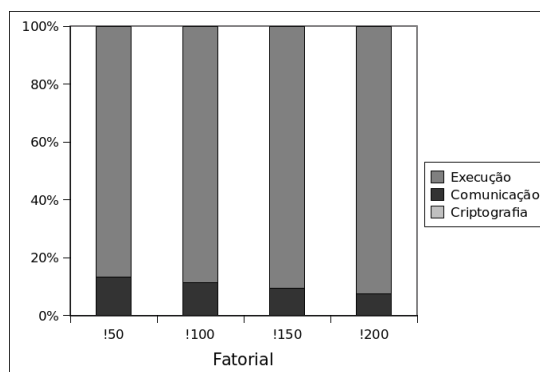
(b) Adição de matriz - 500x500 e 750x750



(c) Multiplicação de matriz - 50x50 a 250x250



(d) Multiplicação de matriz - 500x500 e 750x750



(e) Fatorial - 50 a 200

Fonte – Próprio autor

- Problema com complexidade linear (**Fatorial**): apresentou um custo total rápido e eficiente com comunicação e execução da operação matemática para todas as entradas utilizadas;
- Problema com complexidade quadrática (**Adição de matrizes**): apresentou o tempo gasto com a comunicação como fator determinante para a ineficiência do processo como todo, enquanto o tempo gasto com a operação matemática chegou a ter, no máximo um percentual próximo a 15%;
- Problema com complexidade cúbica (**Multiplicação de matrizes**): apresentou o tempo gasto com a operação matemática como fator determinante para a ineficiência do processo como todo, que, em determinados cenários, demorou até horas para ser resolvido. Enquanto o tempo gasto com a comunicação foi algo próximo ao que foi gasto na operação de adição, mas que foi dominado completamente pelo tempo gasto com a operação matemática;

Logo, pode-se considerar que utilizar o algoritmo de Esteganografia trará o benefício da privacidade dos dados, pois o tempo total gasto com a criptografia poderá ser considerado quase irrelevante, devido as outras métricas possuírem uma influência bem maior no tempo total gasto na operação como todo. E utilizar a técnica de criptografia Homomórfica trará benefícios relacionados à segurança dos dados, porém, caso o problema a ser resolvido tenha uma complexidade quadrática ou cúbica, será um processo muito ineficiente à medida em que o objeto de dados a ser trafegado possui um tamanho cada vez maior. Caso o problema a ser resolvido tenha complexidade linear, será um processo eficiente considerando o cenário de aplicações para dispositivos móveis.

5.5 Considerações finais

Este Capítulo apresentou as informações relacionadas aos resultados obtidos pelo presente trabalho. Foi apresentada a aplicação móvel proposta, a descrição do ambiente de execução dos testes e a descrição do experimento realizado. Cada cenário de teste considerou o impacto causado pela adição de algoritmos de criptografia na operação de *offloading*. A análise dos resultados foi realizada usando as comparações entre tempo gasto com comunicação, tempo gasto com a execução da operação matemática, tempo gasto com a criptografia dos dados e o tempo gasto no geral para cada uma das duas técnicas de segurança utilizadas. A partir disso, notou-se que os testes realizados utilizando o algoritmo de Esteganografia se mostraram bem mais eficiente quando comparado com o algoritmo de criptografia Homomórfica, uma vez que,

para a Esteganografia, o tempo gasto não apresentou uma grande variância com o crescimento do objeto de dados. O que não ocorreu para a Homomórfica no cenário de problemas que serão resolvidos em tempo quadrático ou cúbico, pois, para esse algoritmo, a medida em que o objeto de dados cresce, o tempo gasto da operação como um todo apresenta um crescimento muito acelerado, o que torna a utilização dessa técnica ineficiente para esse contexto.

6 CONCLUSÃO E TRABALHOS FUTUROS

Este capítulo resume o que foi discutido e desenvolvido neste trabalho de conclusão de curso. A Seção 6.1 apresenta os resultados alcançados com a realização da proposta, enquanto na Seção 6.2 é apresentada as principais limitações encontradas na solução. Por fim, a última seção apresenta os trabalhos futuros.

6.1 Resultados Alcançados

A presente pesquisa apresentou o Capítulo de Fundamentação Teórica onde foi exposto informações essenciais de todas as áreas de conhecimento que estão presentes no trabalho em questão e que são fundamentais para o entendimento da pesquisa como um todo. Dentre esses conceitos, pode-se citar a Computação em Nuvem e Móvel, bem como (*Mobile Cloud Computing*), com foco na operação de *offloading* e além disso, conceitos sobre Dispositivo Móveis e conceitos de Segurança da Informação com foco em algoritmos de criptografia.

Foi apresentado os trabalhos relacionados a área de segurança no processo de *offloading* computacional, que apesar de proporem técnicas que impedem a disseminação dos dados do dispositivo móvel para a nuvem sem proteção, apresentam uma falta de análise do impacto real no processamento das requisições. Logo, fez-se necessário compreender esses cenários para selecionar as melhores alternativas para o trabalho que pudessem mitigar os problemas encontrados.

A metodologia utilizada na realização da pesquisa apresentou as etapas a serem seguidas para a construção do trabalho, iniciando pelo estudo bibliográfico para selecionar os algoritmos de Criptografia Homomórfica e de Esteganografia. Posteriormente, a definição da arquitetura das aplicações desenvolvidas e a comunicação entre as mesmas, além da definição da métrica de desempenho utilizada para medir o impacto causado no processo *offloading* após a adição dos algoritmos de criptografia no processo.

O presente trabalho apresentou uma implementação de uma arquitetura cliente-servidor com o foco em prover segurança com uso de algoritmos de criptografia Homomórfica e Esteganografia para a operação de *offloading* utilizada no cenário de aplicações para dispositivos móveis. A partir disso, abre-se a possibilidade para que desenvolvedores da plataforma *Android*

e da linguagem de programação Java possam utilizar os recursos do *offloading* com segurança a partir dos algoritmos de criptografia implementados.

Além disso, no Capítulo de Resultados foi apresentado a realização de todos os testes e análise de dados gerados. A partir disso, foi possível concluir que a utilização do algoritmo de Esteganografia se mostrou mais eficiente quando comparado com o algoritmo de criptografia Homomórfica em termos de tempo de *offloading*, uma vez que para a Esteganografia o tempo gasto não apresentou uma grande variância com o crescimento do objeto de dados, e em termos gerais, o custo com o processamento total foi baixo.

Esse resultado foi diferente do que não ocorreu para a Homomórfica, mais precisamente no cenário de problemas que serão resolvidos em tempo quadrático ou cúbico, pois para esse algoritmo, a medida em que o objeto de dados cresce, o tempo gasto da operação como um todo apresenta um crescimento muito acelerado, o que torna a utilização dessa técnica ineficiente para esse contexto. Entretanto, para o cenário de problemas que podem ser resolvidos em tempo linear, o custo com o tempo total de processamento foi baixo e se mostrou eficiente.

6.2 Limitações encontradas

Ao decorrer da construção do presente trabalho foram encontradas algumas dificuldades responsáveis por limitar algumas execuções e a utilização de algumas ferramentas que inicialmente foram propostas no trabalho de conclusão I.

A primeira limitação que pode ser citada foi que, não foi possível implementar os algoritmos de criptografia Homomórfica e Esteganografia diretamente no módulo de segurança desenvolvido por (SILVA, 2019) no *framework CAOS*, e ,a partir disso, realizar a análise do impacto trazido pela adição desses algoritmos utilizando o CAOS. Esta limitação foi devido, principalmente, a ocupações extra curriculares do presente autor que ocasionaram em pouco tempo para o entendimento e utilização do *framework*.

Outra limitação encontrada foi para a execução das operações de adição e multiplicação utilizando a criptografia Homomórfica para matrizes com dimensões de 1000x1000, uma vez que a aplicação móvel não conseguiu realizar a criação das matrizes com essas dimensões para que fosse possível solicitar a execução por parte da aplicação servidor, devido a falta de recursos relacionados a memória do dispositivo quando utilizado essa técnica.

6.3 Trabalhos Futuros

Como trabalhos futuros, propõe-se algumas melhorias no trabalho, tais como:

- Implementar os algoritmos de criptografia Homomórfica e Esteganografia utilizando o módulo de segurança do CAOS;
- Realizar uma análise comparativa do tempo gasto no processo de *offloading* utilizando os algoritmos já implementados no módulo do CAOS com os algoritmos de criptografia Homomórfica e Esteganografia;
- Implementar uma Anotação Java, que receberá parâmetros (*e.g.*, nível de segurança, desempenho exigido) e, a partir deles, definir o algoritmo criptográfico que será utilizado.
- Realizar uma análise do consumo energético do *offloading* ao ser realizado através da solução proposta.

REFERÊNCIAS

- ANDRADE, L. P.; SOARES, D. N.; COUTINHO, M. M.; ABELÉM, A. J. G. Análise das vulnerabilidades de segurança existentes nas redes locais sem fio: Um estudo de caso do projeto wlaca. **Universidade Federal do Pará, Belém**, <http://www.lprad.ufpa.br/~margalho/wdeec/tcc.pdf>, v. 15, 2008.
- AUGUSTO *et al.* Computação em névoa: Conceitos, aplicações e desafios. Sociedade Brasileira de Computação, 2016.
- BINE; KUK, J.; NEUMANN, J. Estudo de segurança em dispositivos móveis. **Revista Científica Semana Acadêmica Unicentro – PR. Ed.96, vol.1.**, 2016.
- BRANDAO, E.; VALIM, P. Dependência mobile: a relação da nova geração com os gadgets móveis digitais. **white paper**, 2016.
- BURNETT, S.; PAINE, S. Criptografia e segurança: o guia oficial rsa. Gulf Professional Publishing, 2002.
- CECI, L. **Number of mobile app downloads worldwide from 2016 to 2021**. 2022. Disponível em: <<https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/>>.
- CISCO. Cisco annual internet report (2018-2023). **white paper**, 2020.
- CLEMENT, J. **Percentage of mobile device website traffic worldwide from 1st quarter 2015 to 4th quarter 2021**. 2022. Disponível em: <<https://www.statista.com/statistics/277125/share-of-website-traffic-coming-from-mobile-devices/>>.
- COSTA *et al.* Mpos: A multiplatform offloading system. Proceedings of the 30th Annual ACM Symposium on Applied Computing, 2015.
- DESHMUKH, S.; SHAH, R. Computation offloading frameworks in mobile cloud computing: a survey. In: IEEE. **2016 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)**. [S.l.], 2016. p. 1–5.
- FERNANDO, N.; LOKE, S. W.; RAHAYU, W. Mobile cloud computing: A survey. **Future generation computer systems**, Elsevier, v. 29, n. 1, p. 84–106, 2013.
- FIGUEIREDO, C.; NAKAMURA, E. F. Computação móvel: Novas oportunidades e novos desafios. Research Gate, 2003.
- FILHO, J. G.; SILVA, G. P. da; MICELI, C. Compressão e otimização de chaves públicas usando algoritmo genético em criptografia completamente homomórfica. **XV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais-SBSEG**, p. 09–12, 2015.
- GENTRY, C. Fully homomorphic encryption using ideal lattices. **Proceedings of the forty-first annual ACM symposium on Theory of computing**, p. 169–178, 2009.
- GOMES *et al.* Um serviço de offloading de dados contextuais com suporte à privacidade. 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC). [S.l.], 2017. v. 1, p. 957–966, 2017.

- GOMES, F. A.; REGO, P. A.; ROCHA, L.; SOUZA, J. N. de; TRINTA, F. Chaos: A context acquisition and offloading system. In: IEEE. **2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)**. [S.l.], 2017. v. 1, p. 957–966.
- GOMES, F. A.; REGO, P. A.; TRINTA, F. A. M.; VIANA, W.; SILVA, F. A.; MACÊDO, J. A. de; SOUZA, J. N. de. A study about the impact of encryption support on a mobile cloud computing framework. In: **CLOSER**. [S.l.: s.n.], 2020. p. 400–407.
- GUAN, L.; KE, X.; SONG, M.; SONG, J. A survey of research on mobile cloud computing. In: IEEE. **2011 10th IEEE/ACIS International Conference on Computer and Information Science**. [S.l.], 2011. p. 387–392.
- GUPTA, A. K. Challenges of mobile computing. In: **Proceedings of 2nd National Conference on Challenges and Opportunities in Information Technology (COIT-2008)**. Mandi Gobindgarh, India: RIMT-IET. [S.l.: s.n.], 2008. p. 86–90.
- HUANG, D.; WU, H. Mobile cloud computing taxonomy. In: **Mobile Cloud Computing**. [S.l.]: Elsevier, 2018. p. 5–29.
- HUSSEIN, N. H.; KHALID, A.; KHANFAR, K. A survey of cryptography cloud storage techniques. **International Journal of Computer Science and Mobile Computing**, v. 5, n. 2, p. 186–191, 2016.
- JULIO, E. P.; BRAZIL, W. G.; ALBUQUERQUE, C. Esteganografia e suas aplicações. **Livro de Minicursos do SBSEG**. Rio de Janeiro: Sociedade Brasileira de Computação, v. 7, p. 54–102, 2007.
- JUNIOR, J. L. G. Aplicabilidade de criptografia homomórfica. 2018.
- JÚNIOR, N. B. Criptografia homomórfica. 2013.
- KHAN, M. A. A survey of computation offloading strategies for performance improvement of applications running on mobile devices. **Journal of Network and Computer Applications**, Elsevier, v. 56, p. 28–40, 2015.
- KHARBANDA, H.; KRISHNAN, M.; CAMPBELL, R. H. Synergy: A middleware for energy conservation in mobile devices. p. 54–62, 2012.
- KUMAR, K.; LIU, J.; LU, Y.-H.; BHARGAVA, B. A survey of computation offloading for mobile systems. **Mobile networks and Applications**, Springer, v. 18, n. 1, p. 129–140, 2013.
- LEE, H. S. V.; SCHELL, R. Aplicações móveis: arquitetura, projeto e desenvolvimento. Person Education, Inc..sob o selo Prentice Hall PTR, 2005.
- LIN, H.; ZEADALLY, S.; CHEN, Z.; LABIOD, H.; WANG, L. A survey on computation offloading modeling for edge computing. **Journal of Network and Computer Applications**, Elsevier, p. 102781, 2020.
- LIU, J.; AHMED, E.; SHIRAZ, M.; GANI, A.; BUYYA, R.; QURESHI, A. Application partitioning algorithms in mobile cloud computing: Taxonomy, review and future directions. **Journal of Network and Computer Applications**, Elsevier, v. 48, p. 99–117, 2015.
- LIU, J.; KUMAR, K.; LU, Y.-H. Tradeoff between energy savings and privacy protection in computation offloading. p. 213–218, 2010.

LIU JIBANG; LU, Y.-H. Energy savings in privacy-preserving computation offloading with protection by homomorphic encryption. Proceedings of the 2010 international conference on Power aware computing and systems, HotPower, p. 1–7, 2010.

NIST. The nist definition of cloud computing. Computer Security Division, Information Technology Laboratory, National . . . , 2011.

OLIVEIRA, R. R. Criptografia simétrica e assimétrica-os principais algoritmos de cifragem. **Segurança Digital [Revista online]**, v. 31, p. 11–15, 2012.

PARANJOTHI, A.; KHAN, M. S.; NIJIM, M. *et al.* Survey on three components of mobile cloud computing: offloading, distribution and privacy. **Journal of Computer and Communications**, Scientific Research Publishing, v. 5, n. 06, p. 1, 2017.

PAULA, A. S. de; BRESSAN, M. A.; ABE, T. **SEGURANÇA EM REDES MÓVEIS**. Tese (Doutorado) — Pós Graduação em Redes e Segurança de Sistemas Pontifícia Universidade Católica do Paraná, 2013.

PINTO, P. M. T. L. N.; GOMES, A. R. L. Segurança na conectividade wifi em dispositivos móveis: Estudo de caso do iphone. **e-xacta**, v. 4, n. 3, 2012.

QURESHI, S. S.; AHMAD, T.; RAFIQUE, K. *et al.* Mobile cloud computing as future for mobile applications-implementation methods and challenging issues. In: IEEE. **2011 IEEE International Conference on Cloud Computing and Intelligence Systems**. [S.l.], 2011. p. 467–471.

RAHIMI, M. R.; REN, J.; LIU, C. H.; VASILAKOS, A. V.; VENKATASUBRAMANIAN, N. Mobile cloud computing: A survey, state of art and future directions. **Mobile Networks and Applications**, Springer, v. 19, n. 2, p. 133–143, 2014.

REGO, P. A.; COSTA, P. B.; COUTINHO, E. F.; ROCHA, L. S.; TRINTA, F. A.; SOUZA, J. N. de. Performing computation offloading on multiple platforms. **Computer Communications**, Elsevier, v. 105, p. 1–13, 2017.

REGO, P. A. L. **Applying Smart Decisions, Adaptive Monitoring and Mobility Support for Enhancing Offloading Systems**. Tese (Doutorado) — Universidade Federal do Ceará, 2016.

ROCHA, A. *et al.* Compressão e otimização de chaves públicas usando algoritmo genético em criptografia completamente homomórfica. Webmedia & LA-Web-Joint Conference 2004, 2004.

SANAEI, Z.; ABOLFAZLI, S.; GANI, A.; BUYYA, R. Heterogeneity in mobile cloud computing: taxonomy and open challenges. **IEEE Communications Surveys & Tutorials**, IEEE, v. 16, n. 1, p. 369–392, 2013.

SANTOS, G. B. dos; REGO, P. A.; TRINTA, F. Uma proposta de solução para offloading de métodos entre dispositivos móveis. In: SBC. **Anais Estendidos do XXIII Simpósio Brasileiro de Sistemas Multimídia e Web**. [S.l.], 2017. p. 76–81.

SATYANARAYANAN, M. . Pervasive computing: vision and challenges. *Personal Communications*, IEEE, 8(4):10–17, 2001.

SATYANARAYANAN, M.; BAHL, P.; CACERES, R.; DAVIES, N. The case for vm-based cloudlets in mobile computing. **IEEE pervasive Computing**, IEEE, v. 8, n. 4, p. 14–23, 2009.

SILVA, W. R. d. Estudo comparativo do impacto da segurança em ambientes de mobile cloud computing. 2019.

TRINTA, F.; REGO, P. A.; GOMES, F.; ROCHA, L.; VIANA, W.; SOUZA, J. N. de. Using mobile cloud computing for developing context-aware multimedia applications. In: **Special Topics in Multimedia, IoT and Web Technologies**. [S.l.]: Springer, 2020. p. 51–89.

VARELLA, W. Arquitetura de solução de computação em nuvem. Editora Senac São Paulo, 2019.

VIJAYARANI, S.; TAMILARASI, A. An efficient masking technique for sensitive data protection. In: IEEE. **2011 International Conference on Recent Trends in Information Technology (ICRTIT)**. [S.l.], 2011. p. 1245–1249.

WAYNER, P. Disappearing cryptography: Information hiding: Steganography and watermarking (2nd edition). Morgan Kaufmann Publishers Inc., 2002.

XU, D.; LI, Y.; CHEN, X.; LI, J.; HUI, P.; CHEN, S.; CROWCROFT, J. A survey of opportunistic offloading. **IEEE Communications Surveys & Tutorials**, IEEE, v. 20, n. 3, p. 2198–2236, 2018.