UNIVERSIDADE PAULISTA – UNIP

DESENVOLVIMENTO DE UM SISTEMA PARA PREVENÇÃO, ACOMPANHAMENTO E CONTROLE DE UMA EPIDEMIAS EM 5 CIDADES DA GRANDE SÃO PAULO

SÃO PAULO

2025

ANA

ANDRE LUIZ CIORFI FREITAS

FABRICIO ANDRADE

CASSIANO

GIANLUCCA DUARTE DA COSTA

GUSTAVO FRAGA

DESENVOLVIMENTO DE UM SISTEMA PARA PREVENÇÃO, ACOMPANHAMENTO E CONTROLE DE UMA EPIDEMIAS EM 5 CIDADES DA GRANDE SÃO PAULO

Artigo apresentado para matéria de Atividades Práticas Supervionadas (APS) da Universidade Paulista (UNIP), campus Anchieta São Paulo, sob orientação do professor: Marcos Junior.

SÃO PAULO

2025

RESUMO

Este trabalho tem por finalidade estudar uma das epidemias que afetam a região de São Paulo, com foco no mosquito transmissor da dengue. Serão abordadas informações relevantes sobre esse problema de saúde, incluindo dados estatísticos, evolução dos casos entre outros. Além disso, o projeto propõe o desenvolvimento de um programa em Java integrado a um banco de dados MySQL, com a finalidade de comparar e analisar informações sobre a doença nas cinco principais regiões da Grande São Paulo: São Bernardo do Campo, Santo André, São Caetano do Sul, Diadema e a capital São Paulo. A proposta visa oferecer uma ferramenta de visualização e análise que auxilie na compreensão da propagação da dengue nessas localidades.

Palavras-chave: Dengue. Saúde populacional. Java. Banco de dados. São Paulo.

ABSTRACT

This study aims to investigate one of the epidemics affecting the São Paulo region, focusing on the mosquito responsible for transmitting dengue. The research presents relevant information about this public health issue, including statistical data, case evolution, and other key aspects. Additionally, the project involves the development of a Java application integrated with a MySQL database, designed to compare and analyze dengue-related data across the five main areas of Greater São Paulo: São Bernardo do Campo, Santo André, São Caetano do Sul, Diadema, and the city of São Paulo. The proposed system seeks to provide a visualization and analysis tool to support a better understanding of dengue spread within these locations.

Keywords: Dengue. Population health. Java. Database.São Paulo.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1. OBJETIVO DO TRABALHO

O obejtivo deste trabalho, é investigar uma das principais epidemias que atingem a região metropolitana de São Paulo, com ênfase no Aedes aegypti, mosquito vetor da dengue. A pesquisa contempla a apresentação de dados relevantes sobre a doença, como estatísticas, histórico da disseminação e impacto nas comunidades. Complementarmente, será desenvolvido um sistema utilizando a linguagem Java, conectado a um banco de dados MySQL, com o intuito de coletar, comparar e analisar informações sobre a incidência da dengue nas cinco principais cidades da região: São Bernardo do Campo, Santo André, São Caetano do Sul, Diadema e São Paulo. O projeto busca disponibilizar uma ferramenta informativa que contribua para a visualização e interpretação da distribuição da doença nessas localidades.

2. INTRODUÇÃO

Desde o nascimento até o fim da vida, é inevitável que todo ser humano precise adotar uma série de cuidados e hábitos para preservar sua saúde. Vacinação, consumo de água potável, alimentação adequada e práticas de higiene são exemplos de atitudes fundamentais na prevenção de doenças. Seguindo essa mesma lógica, é possível destacar uma das preocupações recorrentes da saúde populacional no Brasil: a dengue.

Segundo o Ministério da Saúde (Brasil, 2024), a dengue é uma doença viral que faz parte do grupo das arboviroses, caracterizadas por serem transmitidas por insetos, como os mosquitos. No Brasil, o principal transmissor é a fêmea do *Aedes aegypti*, cujo nome significa “odioso do Egito”. Esse mosquito é o responsável por disseminar os quatro tipos conhecidos do vírus da dengue — DENV-1, DENV-2, DENV-3 e DENV-4 —, todos pertencentes à família *Flaviviridae* e ao gênero *Orthoflavivirus*. Cada sorotipo possui características genéticas próprias, o que dificulta o controle da doença. (BRASIL 2024).

Vale comentar que, a dengue não é uma doença recente no Brasil. Com base no livro “Dengue e teorias práticas”, registros históricos indicam que já no século XIX ocorreram epidemias em diversas cidades brasileiras, como no Rio de Janeiro em 1846, em São Paulo em 1852 e 1916, e em Niterói em 1923. Durante esse período, a doença era conhecida por diferentes nomes populares, como "polca", "patuleia", "urucubaca", "febre eruptiforme" e "febre quebra-ossos".Após um longo período, a dengue foi reintroduzida no Brasil entre 1981 e 1982, em Boa Vista, Roraima sendo a primeira epidemia registrada. Dessa forma, inicialmente confundida com rubéola, a doença foi confirmada como dengue, com a identificação dos vírus dengue 1 e dengue 4 pelos pesquisadores do Instituto Evandro Chagas (VALLE; PIMENTA; CUNHA, 2015).

A proliferação do mosquito é favorecida por diversos fatores, como a urbanização acelerada, o crescimento populacional sem planejamento, a falta de saneamento básico adequado e as condições climáticas. Esses elementos tornam o ambiente propício para a reprodução do vetor, principalmente nos meses mais quentes e chuvosos, entre outubro e maio, quando há um aumento significativo no número de casos e no risco de surtos da doença. (BRASIL 2024).

Ainda segundo o Ministério da Saúde, a dengue pode se manifestar de forma leve ou evoluir para quadros mais graves, sendo importante reconhecer os sinais e sintomas desde os primeiros dias. Os sintomas mais comuns incluem febre alta, dores de cabeça (especialmente atrás dos olhos), náuseas, sensação de cansaço extremo, dores nas articulações e o surgimento de manchas vermelhas na pele. Esses sinais geralmente caracterizam a fase inicial da doença, que pode durar de dois a sete dias.

No entanto, é fundamental estar atento aos chamados sinais de alerta, que podem indicar uma possível progressão para a forma grave da dengue. Entre eles estão dores abdominais intensas e persistentes, vômitos frequentes, tontura, dificuldade para respirar, sangramentos nas gengivas, nariz ou nas fezes, além de extremo cansaço ou irritabilidade. Esses sintomas exigem atenção médica imediata. Após a fase crítica da doença, o paciente pode iniciar a recuperação. No entanto, em alguns casos, a dengue evolui para formas mais severas, caracterizadas por extravasamento grave de plasma, hemorragias intensas ou falência de órgãos vitais, o que pode levar ao óbito se não houver intervenção médica adequada.

Por fim, o Ministério da Saúde (BRASIL, 2024), informa que a prevenção ainda é a forma mais eficaz de combater a dengue e outras arboviroses urbanas, como chikungunya e Zika. O controle do mosquito *Aedes aegypti* é fundamental e envolve medidas simples no cotidiano, como a eliminação de recipientes que acumulam água parada, vedação de caixas d’água, desobstrução de calhas e ralos, além do uso de telas nas janelas e repelentes em locais com transmissão ativa. A participação da população também é essencial, especialmente no apoio às ações desenvolvidas pelo Sistema Único de Saúde (SUS).Como reforço às estratégias de controle, a vacina contra a dengue passou a integrar o Calendário Nacional de Vacinação em fevereiro de 2024. Inicialmente, a campanha foi direcionada a 521 municípios, distribuídos em 37 regiões de saúde, conforme a capacidade de produção do imunizante. (BRASIL 2024).

Figura 1: Cartaz elaborado pelo Governo de SP e Centro Paula Souza sobre preveções contra dengue.

Fonte: SÃO PAULO.

Fonte: Fernandes Flávia, TechTudo 2022.

Porém, vale ressaltar que, uma vez quebrados ou invadidos, os sistemas criptográficos podem gerar inúmeros prejuízos. Segundo o site de tecnologia e notícias TECMUNDO, mantido pela NZN, e outras fontes, um exemplo de falha na segurança criptográfica ocorreu em julho de 2024, um pesquisador de segurança cibernética Ryan Castellucci conseguiu invadir o sistema da GivEnergy, uma empresa do Reino Unido que fornece energia virtual a cerca de 40 mil residências. Através de uma vulnerabilidade em uma chave criptográfica RSA de 512 bits, Castellucci obteve acesso à conta de administrador da plataforma de gerenciamento de energia da empresa. Ele descobriu essa falha ao inspecionar a API do sistema da GivEnergy, que estava integrado aos seus próprios painéis solares e sistema de armazenamento de bateria. Essa brecha de segurança permitiu que Castellucci controlasse uma rede de 60 mil sistemas conectados à nuvem da empresa, com uma capacidade de geração total de aproximadamente 200 MW. Além de conseguir acesso ao gerenciamento de energia, o pesquisador pôde visualizar dados pessoais de clientes, incluindo nomes, endereços, e-mails e números de telefone. Embora a invasão tenha demonstrado o potencial de manipulação dessas informações sensíveis, Castellucci optou por não as alterar, limitando-se a expor a vulnerabilidade da criptografia empregada. Posteriormente o erro foi encontrado e corrigido pela empresa, adotando outras medidas de segurança, como uma fortificação na chave criptografia para API. (TECMUNDO, 2024).

Diante disso é notável, o quanto a criptografia é importante e crucial para a defesa de informações e dados secretos, sendo cada vez mais importante na área de tecnologia e necessária em amplos locais e usos.

3. CRIPTOGRAFIA (conceitos gerais)

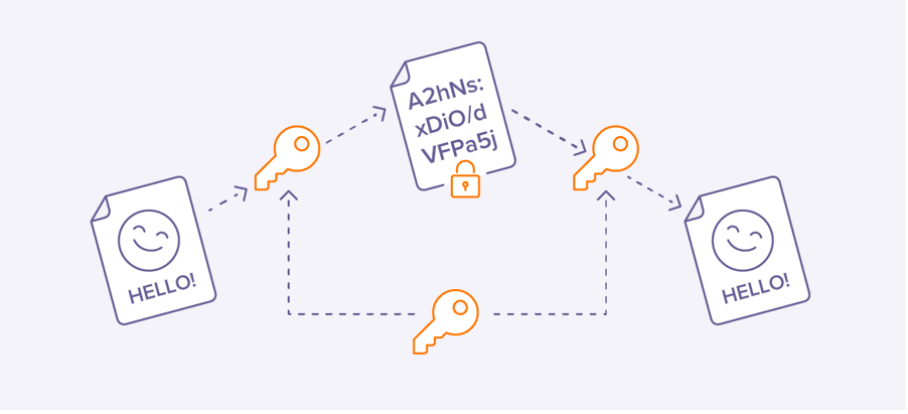
3.1 O que é criptografia

O termo "criptografia" tem suas raízes na Grécia Antiga, derivado das palavras "kryptós", que significa "oculto", e "gráphien", que significa "escrever". Assim, a criptografia pode ser entendida como a "arte de escrever de maneira oculta".

Segundo a empresa Kaspersky Lab, (que é uma empresa tecnológica russa especializada na produção de softwares de segurança à Internet, com distribuição de soluções para segurança da informação contra vírus, hackers, spam, entre outros). A criptografia consiste em transformar dados legíveis em um formato codificado, de modo que só possam ser acessados ou processados após serem devidamente descriptografados. Esse processo é um componente essencial para a proteção de informações, sendo uma das formas mais eficazes de impedir que dados de sistemas computacionais sejam interceptados e usados de maneira indevida por hackers ou afins.

A criptografia de dados é amplamente adotada tanto por indivíduos quanto por grandes empresas, garantindo a segurança de informações transmitidas entre navegadores e servidores. Isso abrange desde dados financeiros até detalhes pessoais sensíveis. Para implementar essa proteção, utilizam-se programas de criptografia, conhecidos como algoritmos de codificação, que criam sistemas projetados para serem extremamente difíceis de decifrar, exigindo um imenso poder de processamento para serem quebrados. (KASPERSKY)

Figura 2: Ilustração de criptografia.



Fonte: DEWITT, Derek. AVAST 2021.

3.2 Breve história da Criptografia

De acordo com informações do site oficial da IBM, material escrito por Josh Scheineider, a história da criptografia começou a ser usada por volta de 1900 a.C., no Egito Antigo, onde foram encontrados hieróglifos não padronizados em tumbas, sugerindo uma tentativa de cifrar informações. Mais tarde, em 1500 a.C., na Mesopotâmia, registros cifrados em tabletes de argila protegiam receitas comerciais de esmaltes cerâmicos. Em Esparta, por volta de 650 a.C., surgiu o uso da cítala, um dispositivo que embaralhava mensagens em couro, considerado um dos primeiros cifradores de transposição. Já em Roma, entre 100 e 44 a.C., Júlio César popularizou o "Cifra de César", um método simples de substituição, onde as letras eram deslocadas no alfabeto para cifrar mensagens. (SCHEINEIDER)

De acordo com Isabel Cristina Henriques Sales, mestranda do Programa Multinstitucional e Inter-Regional de Pós-Graduação em Ciências Contábeis (UnB, UFPB e UFRN), a Cifra de César, também chamada de código ou troca de César, é uma das técnicas mais simples e conhecidas de criptografia. Essa técnica consiste em uma cifra de substituição, onde cada letra de um texto é substituída por outra, com um deslocamento fixo de posições no alfabeto. Por exemplo, ao deslocar três posições, a letra A se torna D, a letra B se torna E, e assim sucessivamente. (SALES,2015)

De forma organizada, a Cifra de Cesar corresponde a está lógica resumidamente:

A	B	C	D	E	F	G	H	I
D	E	F	G	H	I	J	K	L

Com base nessas informações se aplicarmos esse deslocamento à frase "Programação de códigos", ela se transformaria em "surjudpdfdr gh frgljrv". O nome dessa técnica faz referência ao imperador Júlio César, que a usava para enviar mensagens seguras a seus generais.

Apesar de sua simplicidade, a Cifra de César é frequentemente utilizada em sistemas mais complexos, como a cifra de Vigenère, e ainda tem aplicações modernas, como no sistema ROT13. Entretanto, por ser uma cifra de substituição monoalfabética, ela é facilmente decifrável e não proporciona alta segurança. A criptografia por Cifra de César pode ser representada por aritmética modular, associando letras a números (A = 0, B = 1,..., Z = 25). Nesse formato, o processo de criptografia com deslocamento fixo pode ser descrito matematicamente.

Voltando a história, a criptografia na era Medieval no século IX, cita o matemático árabe Al-Kindi que introduziu a análise de frequência, uma técnica usada para decifrar cifras de substituição monoalfabéticas, sendo um marco na criptoanálise. Em 1467, Leon Battista Alberti desenvolveu cifras polialfabéticas, mais seguras do que as de um único alfabeto, o que o tornou uma figura central na criptografia medieval. Já no século XVI, Giovan Battista Bellaso publicou o método que mais tarde ficou conhecido como Cifra de Vigenère, erroneamente atribuída a Blaise de Vigenère, que também desenvolveu uma cifra de autochave em 1586.

Por fim o material da IBM escrito por Scheineider, informa que a criptografia moderna diz sobre que tal ciência na área militar se intensificou durante a Primeira Guerra Mundial, com as forças britânicas decifrando comunicações alemãs. Em 1917, Edward Hebern projetou a primeira máquina de criptografia rotativa, que automatizava a cifragem de mensagens. No pós-guerra, Arthur Scherbius desenvolveu a famosa máquina Enigma, que os alemães usaram extensivamente durante a Segunda Guerra Mundial, até ser decifrada por Alan Turing e sua equipe, em um esforço que marcou a criptoanálise moderna. Nos anos 1970, a IBM criou o DES (Data Encryption Standard), um dos primeiros padrões de criptografia modernos, embora logo fosse considerado inseguro devido ao avanço da tecnologia. Em 1976, o método de troca de chaves Diffie-Hellman revolucionou a criptografia com a introdução da chave assimétrica, eliminando a necessidade de uma chave secreta compartilhada. Um ano depois, o algoritmo RSA, desenvolvido por Rivest, Shamir e Adleman, trouxe uma nova forma de segurança baseada na fatoração de grandes números primos. Por fim, em 2001, o Advanced Encryption Standard (AES) substituiu o DES, sendo adotado por ser mais seguro e capaz de resistir aos desafios do poder computacional moderno. (SCHEINEIDER)

4. TÉCNICAS MAIS UTILIZADAS E CONHECIDAS

4.1 O que é uma chave na criptografia

Segundo a Cloudflare (que e é uma das maiores redes que operam na internet. As pessoas usam os serviços da Cloudflare com o objetivo de aumentar a segurança e o desempenho de seus sites e serviços) na criptografia, uma chave é um elemento fundamental que permite embaralhar os dados, transformando-os em uma sequência aparentemente aleatória. Em geral, uma chave é um valor longo composto de números e/ou letras. Quando aplicamos uma chave a um algoritmo de criptografia, o texto original, ou texto em claro, se transforma em um conjunto de dados codificados, tornando-o ilegível para quem não possui a chave correta. Contudo, com a chave adequada, os dados podem ser revertidos ao seu estado original. (CLOUDFLARE)

4.2 Chave Pública

Segundo o site oficial da IBM uma criptografia de chave pública, também conhecida como chave assimétrica, consiste em ser um par de chaves que estão matematicamente interligadas, uma mensagem criptografada com a primeira chave só pode ser decriptada com a segunda, e vice-versa.

Cada usuário de um sistema de chave pública possui um conjunto de duas chaves: uma chave privada, mantida em sigilo, e uma chave pública, que pode ser distribuída livremente. Qualquer pessoa pode usar a chave pública de outro usuário para enviar uma mensagem que só o destinatário poderá ler, pois apenas ele possui a chave privada correspondente. Essa abordagem permite a troca segura de informações, mesmo em redes sem segurança. Você pode criptografar uma mensagem para outra pessoa com a chave pública dela, e essa pessoa poderá decifrá-la com sua chave privada, protegendo o conteúdo durante a transmissão.

A criptografia de chave pública é particularmente útil devido a algumas de suas características:

1. **Segurança**: Cada participante precisa de apenas duas chaves — pública e privada. Somente a chave privada deve permanecer secreta, o que reduz o risco de roubo em comparação aos sistemas simétricos.

2. **Facilidade de distribuição**: As chaves públicas podem ser divulgadas sem comprometer a segurança, eliminando a necessidade de trocas secretas de chaves, como ocorre com sistemas de chaves simétricas.
3. **Desempenho assimétrico**: Como o nome sugere, a criptografia de chave pública é também chamada de criptografia assimétrica porque uma chave do par cifra a mensagem, enquanto a outra a decifra, ao contrário do que ocorre na criptografia simétrica.

No entanto, a criptografia de chave pública enfrenta o desafio de verificar a identidade do proprietário da chave. Sem uma forma confiável de validação, qualquer pessoa poderia criar um par de chaves e publicar a chave pública sob um nome falso. Sistemas de chave pública mitigam esse risco usando certificados digitais e assinaturas digitais, que garantem a autenticidade e a integridade das chaves. (IBM, 2022)

Vale ressaltar que, segundo Ronielton Rezende Oliveira, Gerente de Projetos (PMP, PRINCE2, CSM, MCTS, COBIT, ITIL) com formação acadêmica continuada (PhD, MSc, MBA) e experiência desde 1994 em empresas de diferentes portes e setores, alguns dos algoritmos de chave pública mais notáveis incluem o RSA, amplamente usado para comunicações seguras e conhecido por sua dependência na dificuldade de fatoração de números grandes, o ElGamal, que utiliza o problema do logaritmo discreto para manter a segurança e é popular em gerenciamento de chaves e o Diffie-Hellman, criado para estabelecer segredos compartilhados em comunicações abertas, embora sem fornecer ciframento direto. Além disso, algoritmos com Curvas Elípticas, como ElGamal e Diffie-Hellman, operam em um domínio matemático que permite chaves menores e segurança elevada, embora, em algumas situações, possam ser mais lentos do que o RSA.(OLIVEIRA,2012)

Figura 3: Ilustração de criptografia com chave pública.

Fonte: EVAL, 2019.

4.3 Chave Privada

A criptografia de chave privada é conhecida também como criptografia simétrica, uma vez que a mesma chave realiza tanto a criptografia quanto a descriptografia da mensagem. Segundo a IBM e seu artigo, sistemas de criptografia com chave privada utilizam uma única chave secreta que é compartilhada entre quem envia e quem recebe a mensagem. Ambos precisam ter essa chave: o remetente a usa para cifrar a mensagem, enquanto o destinatário a utiliza para decifrá-la. Para garantir a segurança da comunicação, a chave deve ser mantida confidencial por ambos.

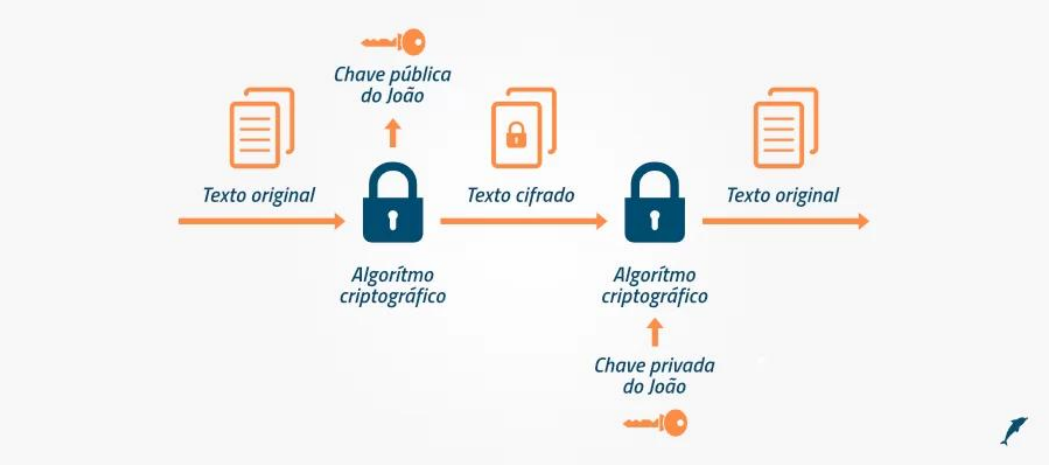
Esse tipo de criptografia apresenta algumas limitações para ser amplamente utilizado:

1. **Escalabilidade limitada**: Cada par de comunicantes precisa de uma chave exclusiva para troca segura de mensagens. O número de chaves necessárias cresce exponencialmente conforme mais usuários se comunicam.
2. **Risco na distribuição**: A chave deve ser compartilhada diretamente entre os participantes, o que a torna vulnerável a interceptações.
3. **Necessidade de acordo prévio**: A comunicação criptografada requer planejamento antecipado, pois é preciso trocar a chave de segurança antes que a primeira mensagem seja enviada. (IBM,2022).

Além disso, segundo Ronielton Rezende Oliveira, destaca diversos exemplos desta técnica de criptografia, o AES (Advanced Encryption Standard), escolhido pelo NIST como padrão para substituir o DES, com blocos de 128 bits e chaves de 128, 192 ou 256 bits. Esse algoritmo é conhecido por sua rapidez e eficiência em software e hardware. Já o DES (Data Encryption Standard), embora popular no passado, tornou-se vulnerável devido ao tamanho reduzido da chave (56 bits), sendo aprimorado no 3DES, que realiza três camadas de ciframento para aumentar a segurança, embora seja mais lento. O IDEA (International Data Encryption Algorithm), desenvolvido para o mercado financeiro e PGP (um dos programas de e-mail seguro mais populares), utiliza uma chave fixa de 128 bits e é notado por seu desempenho superior ao DES em software. O Blowfish oferece segurança flexível, com chaves variáveis entre 32 e 448 bits, enquanto sua versão aprimorada, Twofish, utiliza blocos de 128 bits e chaves de até 256 bits, sendo rápida e disponível sem restrições de patente.

Outros algoritmos notáveis incluem o RC2, com chave variável de 8 a 1024 bits, ideal para criptografia de e-mails, e o CAST-128, que possui chaves de 40 a 128 bits e é comumente usados em aplicações que exigem segurança confiável e baixo custo computacional.

Figura 4: Ilustração de criptografia com chave privada.



Fonte: EVAL, 2019.

4.4 Certificado digital

Segundo Cristian Thiago Moecke, ex-CTO da BRy Tecnologia e atualmente na Griaule, o certificado digital é uma ferramenta crucial no ambiente digital, atuando como uma identidade eletrônica que verifica e autêntica a identidade de indivíduos e empresas. Baseado na criptografia de chaves assimétricas, cada certificado contém uma chave pública e uma chave privada, essenciais para assinaturas digitais. Emitidos por Autoridades Certificadoras (AC) confiáveis, os certificados garantem a validade das informações, que incluem o nome do titular e a assinatura digital da AC. No Brasil, o Instituto Nacional de Tecnologia da Informação (ITI) gerencia a Infraestrutura de Chaves Públicas Brasileira (ICP-Brasil), que assegura a segurança do sistema de certificação. A certificação digital é fundamental para a confiança em transações eletrônicas, sendo amplamente utilizada para assinatura de contratos e autenticação em sistemas, especialmente com o aumento da digitalização nos negócios e na administração pública. (THIAGO MOECKE, 2018)

De acordo com pesquisas de dois artigos da empresa ValidCertificadora, presente em mais de 15 países e especializada dentro do grupo Valid, dedicada especificamente à área de certificação digital, e TOTVS que é uma empresa brasileira de tecnologia e software com sede em São Paulo, reconhecida como líder em soluções para empresas e com uma ampla gama de sistemas e serviços financeiros com mais de 70 mil clientes, atende 12 setores da economia, os certificados digitais no Brasil são classificados em três principais grupos: A, S e T, conforme sua função e nível de segurança.

Os certificados digitais são divididos em categorias que atendem a diferentes necessidades de segurança e autenticidade em transações eletrônicas. Entre eles, os certificados do tipo A são fundamentais para autenticar documentos e transações, conferindo-lhes validade jurídica. O **Certificado Digital A1**, com validade de um ano, é armazenado diretamente no computador do usuário e ideal para transações frequentes, já que permite assinaturas rápidas e dispensa o uso de hardware específico. O **Certificado Digital A2**, com validade de dois anos, é armazenado em mídias como discos rígidos, oferecendo segurança prática para assinaturas digitais. Já o **Certificado Digital A3** é guardado em dispositivos externos, como smart cards ou tokens USB, com validade de um a três anos, proporcionando alta segurança para setores que lidam com informações sensíveis, pois mantém as chaves fora do computador.

Similar ao A3, o **Certificado Digital A4** usa chaves de 2048 bits, sendo recomendado para aplicações que requerem confidencialidade e integridade dos dados. Por outro lado, os certificados do tipo S asseguram o sigilo das transações por meio de criptografia. Classificados nas variações de S1 a S4, eles diferem no tamanho da chave, validade e opções de armazenamento, permitindo que as organizações escolham o nível de segurança que melhor se adapta às suas necessidades. Por fim, o Certificado do tipo T, conhecido como "certificado de tempo" ou “timestamp”, é utilizado para registrar a data e hora exatas de criação ou assinatura de um documento digital. Esse tipo de certificação é essencial para garantir a integridade e autenticidade dos documentos ao longo do tempo, funcionando como uma prova de que uma transação ocorreu em um momento específico. (VALIDCERTIFICADORA, 2024; VALIDCERTIFICADORA, 2024; TOTVS, 2020; TOTVS, 2024)

4.5 Assinatura digital

De acordo com Carlos Francisco Tatara, CTO da Bry e graduado em Ciências da Computação pela UFSC, a assinatura digital é uma técnica que formaliza documentos digitais, garantindo segurança e integridade, especialmente em arquivos como PDFs. Dedicado há duas décadas à segurança da informação, ele explica que essa técnica utiliza chaves criptográficas em certificados digitais, identificando os signatários e assegurando a validade jurídica dos documentos. Com o crescimento da digitalização das informações, a autenticação desses dados tornou-se essencial, assim como acontece com as assinaturas manuscritas. A criação da ICP-Brasil em 2001 conferiu validade jurídica aos documentos eletrônicos, permitindo que, quando autenticados por assinatura digital, substituam completamente os papéis. Ele explica que entre as principais vantagens estão a otimização de processos, a conformidade jurídica e compliance, a redução de custos e a promoção da sustentabilidade. Destacando que a assinatura digital requer o uso de um certificado digital enquanto a assinatura eletrônica utiliza métodos de identificar a pessoa signatária.

A Lei n° 14.063, de setembro de 2020, regulamentou assinaturas eletrônicas em interações com entidades públicas, introduzindo novas classificações para facilitar a transformação digital. Antes, a adoção era dificultada pelos custos, já que os órgãos públicos precisavam usar certificados digitais ICP-Brasil.

A **assinatura eletrônica simples** identifica o signatário ao vincular informações a dados eletrônicos, utilizando formulários ou endereço IP. Sua segurança é básica e a validade jurídica depende do consentimento das partes, não sendo recomendada para transações que exigem maior segurança. Já a **Assinatura eletrônica avançada** é uma forma de assinatura digital que oferece alto nível de segurança e autenticidade, utilizando certificados digitais (não necessariamente da ICP-Brasil) e criptografia avançada. Ela pode incluir métodos adicionais, como biometria, tokens e PINs, permitindo detectar alterações no documento após a assinatura. Isso é essencial para transações que exigem alta segurança, como contratos financeiros e documentos legais. Por fim, a **Assinatura eletrônica qualificada**, ou assinatura digital, é validada por certificados digitais da ICP-Brasil emitidos por Autoridades Certificadoras (AC).

Ela utiliza chaves criptográficas assimétricas para comprovar a identidade do usuário, conferindo à assinatura a mesma validade jurídica de uma assinatura manual. Dentre os tipos de assinatura digital, a qualificada possui o sistema de segurança mais complexo. (FRANCISCO TATARA, 2024)

4.6 Hashing (funções hash)

Segundo Sávio Vale, graduado em Engenharia Mecânica pelo Instituto Federal do Piauí (IFPI), a Função Hash é um algoritmo matemático utilizado na criptografia, que transforma dados, como arquivos e senhas, em uma sequência alfanumérica de comprimento fixo. Por exemplo, o hash da palavra "voitto" gerado pela função MD5 é 494009d6ad36e1caa1b05e7cc98ab48f. Essa técnica é essencial para resumir informações, verificar a integridade de arquivos e garantir a segurança de dados armazenados em servidores. O hash é complexo e indecifrável, atuando como uma proteção contra invasões, especialmente em transações digitais e criptomoedas.

O algoritmo hash transforma dados de entrada em saídas padronizadas de tamanho fixo, variando entre 128 e 512 bits, facilitando o manejo de grandes volumes de informação. Apesar de parecer que o hash da palavra "voitto" não resume muita informação, essa mesma função pode compactar textos longos, como um artigo de 2000 palavras. O processo de criação de uma função hash é intrincado e requer um entendimento profundo de matemática e programação.

As principais características de uma função hash são: a saída de tamanho fixo, a eficiência de operação e a determinística, onde entradas idênticas sempre geram saídas equivalentes. Os algoritmos de hash mais populares atualmente incluem o Message Digest (MD), que se concentra na verificação da integridade de arquivos, com versões como MD2, MD4 e MD5. Outra função importante é a Secure Hash Function (SHA), empregada na transmissão de dados entre servidor e cliente. O RIPEMD é uma versão melhorada das funções MD, gerando saídas de 160 bits, enquanto as do MD têm 128 bits. Por fim, o Whirlpool é um algoritmo de código livre desenvolvido por dois professores, um brasileiro e outro belga, e é utilizado por organizações como a ISO e a IEC. (VALE, 2024)

5. DISSERTAÇÃO

A criptografia é um campo fundamental na segurança digital, garantindo que informações sensíveis permaneçam protegidas contra acessos não autorizados. O código mostra uma abordagem de criptografia utilizando Curvas Elípticas (ECC), uma técnica moderna e muito eficiente para criptografar e descriptografar mensagens. A criptografia de Curvas Elípticas tem se destacado devido à sua robustez e eficiência, utilizando operações matemáticas em curvas elípticas sobre campos finitos para garantir a segurança das comunicações. A seguir, serão abordados aspectos como a estruturação, conceitos e fundamentação do programa, visando esclarecer melhor seu funcionamento e os princípios que o embasam. Serão discutidos os benefícios da técnica utilizada em comparação com métodos anteriores, destacando as melhorias em termos de eficiência e segurança.

Além disso, exploraremos as aplicações práticas que utilizam essa técnica, mostrando como ela é amplamente aplicada em diferentes contextos de segurança digital. Uma análise comparativa entre essa técnica e outras abordagens modernas será apresentada, permitindo uma visão mais abrangente das opções disponíveis no campo da criptografia. Por fim, também serão mencionadas possíveis vulnerabilidades e falhas associadas à técnica, oferecendo uma visão realista sobre suas limitações e ressaltando a importância de complementar o programa com outras medidas de segurança para garantir a proteção dos dados.

5.1 Estruturação, conceitos e fundamentação

O código executa a criptografia com Curvas Elípticas utilizando uma curva definida pela equação y^2 = x^3 + ax + b mod p, onde a, b e p são parâmetros definidos no código, sendo p = 97 um número primo. O código realiza a verificação se um ponto está na curva e implementa operações de adição de pontos na curva, o que é fundamental para as operações de geração de chaves e multiplicação escalar. No ECC, as operações essenciais são a adição de pontos e a multiplicação escalar, a multiplicação escalar é usada para gerar a chave pública a partir da chave privada, e a função de adição de pontos permite calcular a soma de pontos na curva, essencial para o processo de criptografia. A

Criptografia de Curvas Elípticas (ECC) é baseada em operações matemáticas em curvas elípticas sobre campos finitos. A principal operação é o ponto de multiplicação, quando um ponto na curva é multiplicado por um número inteiro para gerar outro ponto. Ela se baseia na dificuldade de resolver o problema do logaritmo discreto em curvas elípticas, o que torna a criptografia bastante segura mesmo com chaves menores.

Existem protocolos como o ECDH (Elliptic Curve Diffie-Hellman) que permitem a troca segura de chaves entre partes, enquanto algoritmos como o ECDSA (Elliptic Curve Digital Signature Algorithm) são usados para gerar assinaturas digitais, garantindo a integridade e autenticidade das mensagens.ECC é mais eficiente que RSA em termos de tamanho da chave, oferecendo o mesmo nível de segurança com chaves menores.

O código inicial que implementa RSA (Rivest-Shamir-Adleman) é um exemplo clássico de criptografia assimétrica, um método que utiliza um par de chaves para proteger dados. A chave pública é divulgada, permitindo que qualquer pessoa a use para criptografar mensagens que somente o detentor da chave privada pode descriptografar. A segurança do RSA baseia-se em problemas matemáticos complexos, como a fatoração de números primos grandes, tornando-o computacionalmente difícil para os invasores. A implementação inicial do RSA envolve cálculos de exponenciação modular, que são seguras, mas exigem recursos de processamento significativos, especialmente para grandes volumes de dados.

5.2 Benefícios em relação às técnicas anteriores

A criptografia de Curvas Elípticas (ECC) oferece vários benefícios em comparação com técnicas de criptografia tradicionais, como o RSA. Um dos principais benefícios é a capacidade de oferecer o mesmo nível de segurança com chaves menores. Por exemplo, uma chave de 256 bits em ECC oferece o mesmo nível de segurança que uma chave de 3072 bits em RSA. Isso resulta em um desempenho muito mais eficiente, que consome menos recursos computacionais e menos energia. O fato de ECC necessitar de chaves menores para alcançar o mesmo nível de segurança que RSA pode ser vantajoso para plataformas com restrições de largura de banda e com capacidade limitada, como dispositivos móveis e sistemas embarcados.

Além disso, o ECC é mais eficiente em termos de tempo de processamento, o que torna o código mais rápido e escalável. A geração de chaves e as operações de criptografia podem ser feitas de forma mais rápida e com menos uso de memória em comparação com outras técnicas, como RSA.

5.3 Aplicações que fazem/fizeram uso da técnica

A criptografia baseada em Curvas Elípticas (ECC) tem se tornado cada vez mais popular na implementação de conexões seguras em protocolos como HTTPS. Isso acontece principalmente devido à sua maior eficiência em comparação com outros algoritmos tradicionais, como RSA. ECC oferece a mesma segurança com chaves menores, o que reduz o uso de recursos computacionais, tornando as conexões mais rápidas e eficientes, especialmente em ambientes de alta demanda, como sites que lidam com grandes volumes de tráfego.

De acordo, SSL.com (empresa de tecnologia que desempenha um papel crucial na estratégia de segurança cibernética em camadas de uma organização. Como fornecedor de identidade digital e serviços confiáveis, a empresa oferece certificados digitais publicamente reconhecidos, serviços de assinatura em nuvem e soluções de infraestrutura de chave pública (PKI). Suas soluções são utilizadas por empresas e governos em mais de 180 países para proteger redes internas, comunicações com clientes, plataformas de comércio eletrônico e serviços web.

A criptografia de curva elíptica (ECC) possui diversas aplicações práticas, como:

1. Comunicação segura: Implementada em protocolos como TLS/SSL para garantir comunicação web criptografada.
2. Assinaturas digitais: Utilizada em esquemas como o ECDSA (Elliptic Curve Digital Signature Algorithm).
3. Criptomoeda: Base para geração de chaves e assinaturas digitais em sistemas como Bitcoin e Ethereum.
4. Dispositivos móveis: Ideal para dispositivos com recursos limitados, devido à sua eficiência.

5. Internet das Coisas (IoT): Aplicada na proteção da comunicação entre dispositivos IoT com baixa capacidade de processamento e memória.
6. Governo e forças armadas: Adotada pela NSA para proteger informações sensíveis e confidenciais. (SSL.com, 2024)

5.4 Discussão comparativa entre esta técnica e outras em alta

ECC oferece um desempenho superior em termos de tempo de processamento e eficiência. Para o mesmo nível de segurança, ECC requer muito menos poder computacional, o que a torna uma escolha preferida para dispositivos móveis e sistemas com recursos limitados. Segundo SSL.com, a criptografia de curva elíptica (ECC) e o RSA são dois métodos amplamente utilizados em criptografia de chave pública. Embora o RSA tenha sido um padrão confiável por muitos anos, o ECC oferece uma alternativa com segurança equivalente, porém utilizando chaves menores, o que o torna mais eficiente para aplicações modernas. (SSL.com, 2024)

A tabela a seguir ilustra as diferenças nos tamanhos de chave para níveis equivalentes de segurança:

Figura 5: Comparação de bits entre ECC e RSA.

Nível de segurança (bits)	Tamanho da chave ECC	Tamanho da chave RSA
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	521	15360

Fonte: SSL.com

Ambas as técnicas oferecem uma segurança forte, mas ECC tem uma vantagem em termos de eficiência porque usa chaves menores. Enquanto a segurança do RSA

depende da dificuldade de fatorar números grandes, a do ECC se baseia na dificuldade de resolver o problema do logaritmo discreto, que é mais difícil de ser atacado com chaves menores.

5.5 Vulnerabilidades e falhas

Embora o ECC seja uma técnica altamente eficiente e segura, ela não está imune a vulnerabilidades, como

1. Escolha de Parâmetros de Curva: O código usa parâmetros predefinidos como a = 2, b = 3 e p = 97, e isso pode não ser seguro em uma aplicação real. A escolha inadequada de parâmetros pode tornar o sistema vulnerável a ataques.
2. Ataques com poder computacional: Mesmo que seja mais eficiente, o ECC não está imune a ataques com poder computacional avançado, como os que podem ser realizados com computadores quânticos, que poderiam quebrar a segurança das técnicas baseadas em logaritmos discretos.
3. Gerenciamento de Chaves: Em um sistema real, o gerenciamento inadequado de uma chave privada de um usuário pode comprometer toda a segurança. No código, a chave privada é gerada aleatoriamente, mas em sistemas reais o processo de geração e armazenamento de chaves exige maior cuidado.

5.6 Melhorias propostas e/ou implemenetadas

O código RSA inicial exigia um processo mais detalhado para garantir segurança e autenticidade. Cada caractere da mensagem era transformado em um número, que passava por cálculos de exponenciação modular complexas usando um par de chaves: uma chave pública e uma chave privada. Isso não só garantia que apenas o destinatário com a chave privada correta pudesse decifrar a mensagem, mas também criava um método de autenticação confiável.

Com o uso da chave privada, o remetente podia confirmar sua identidade, o que é essencial em ambientes onde a confiança é crucial. Esse modelo faz do RSA uma ferramenta poderosa e indispensável para estabelecer canais de comunicação seguros,

principalmente em aplicações como e-commerce, transferências bancárias e qualquer troca de informações confidenciais.

As melhorias em ECC vêm principalmente na adoção de curvas seguras e a constante verificação da integridade dos parâmetros. O uso de ECC está se expandindo para mais aplicações, principalmente em dispositivos móveis e IoT. Curvas seguras são menos suscetíveis a ataques específicos, como o ataque de Logaritmo Discreto no contexto de curvas elípticas. A implementação de curvas Edwards, por exemplo, oferece maior resistência a ataques e otimiza o desempenho, sendo ideal para uso em ambientes onde a segurança é crítica e os recursos são limitados, como em dispositivos móveis e IoT.

Validação de Parâmetros de Curva: Para impedir ataques, é essencial validar constantemente os parâmetros da curva antes de qualquer operação criptográfica. A verificação dos pontos da curva e da validade dos valores de entrada evita que sejam inseridos valores inválidos ou maliciosos, que poderiam comprometer a segurança. ECC oferece uma vantagem única em comparação a outras criptografias, como RSA, por proporcionar alta segurança com chaves muito menores. Isso reduz o uso de memória e processamento, o que é perfeito para dispositivos móveis e IoT com pequenos números de recursos.

6. PROJETO DO PROGRAMA

Segundo a empresa DigiCert, Inc. (empresa de segurança digital localizada em Lehi, Utah, oferece infraestrutura de chave pública (PKI) e validação para emissão de certificados digitais ou TLS/SSL. Atuando como autoridade certificadora (CA) e terceiro confiável, a empresa desempenha um papel essencial na segurança digital.) a criptografia de curva elíptica (ECC) utiliza a estrutura algébrica de curvas elípticas em campos finitos. Ela baseia-se na dificuldade prática de determinar o logaritmo discreto de um elemento aleatório de uma curva elíptica associado a um ponto base conhecido publicamente.

Esse método criptográfico foi proposto independentemente por Neal Koblitz e Victor S. Miller em 1985, e os algoritmos ECC começaram a ser amplamente adotados em 2004. Em comparação ao RSA, a ECC oferece a vantagem de chaves menores, que

proporcionam maior segurança e maior velocidade. No entanto, sua desvantagem é a falta de compatibilidade de alguns serviços e aplicativos com certificados TLS/SSL baseados em ECC. (DIGICERT)

Objetivo

Este projeto visa desenvolver e demonstrar o funcionamento de um sistema de criptografia baseado em Curvas Elípticas (ECC), utilizando Python para a implementação prática. O sistema inclui:

1. Geração de chaves pública e privada;
2. Funções de criptografia e descriptografia com uso de curva elíptica;
3. Uso de um ponto base arbitrário e uma curva definida no campo finito.

O foco do projeto é oferecer uma implementação educacional, simplificada e funcional do ECC, destacando sua segurança e aplicabilidade em cenários reais.

Justificativa

A Criptografia de Curvas Elípticas é amplamente reconhecida por sua eficiência em segurança com chaves menores em comparação com métodos tradicionais, como RSA. Este projeto simplifica a ECC para facilitar o aprendizado e demonstra como os conceitos básicos podem ser aplicados. Além disso, explora o uso prático em dispositivos de recursos limitados, enfatizando o potencial da ECC.

Metodologia

Fase 1: Configuração Inicial e Definição de Parâmetros

1. Estudo inicial sobre o funcionamento de Curvas Elípticas.
2. Definição da curva: y^2 = x^3 + ax + b mod p, com parâmetros:
   - a=2a, b=3, p=97 (campo finito);
   - Ponto base arbitrário P=(2,22).
3. Configuração do ambiente Python para desenvolvimento.

Fase 2: Geração de Chaves

1. Implementação da geração da chave privada como um número aleatório no intervalo [1,255].
2. Cálculo da chave pública pela multiplicação escalar da chave privada com o ponto base P.

Fase 3: Criptografia e Descriptografia

1. Criptografia:
   - Cada caractere da mensagem é criptografado com a chave pública.
   - Um "nonce" aleatório é gerado para cada caractere, aumentando a segurança.
   - A cifra combina o caractere da mensagem, o "nonce" e o valor x da chave pública.

2. Descriptografia:
   - Utiliza a chave privada e o "nonce" para recuperar o caractere original, garantindo integridade e segurança.

Fase 4: Validação e Testes

1. Testar a criptografia e descriptografia para mensagens de até 128 caracteres.
2. Avaliar o desempenho do sistema em termos de tempo de geração de chaves e criptografia.
3. Garantir que o sistema seja robusto contra erros de entrada e ataques básicos.

Resultados Esperados

1. Um sistema funcional de criptografia que utiliza Curvas Elípticas:
   - Geração de pares de chaves pública e privada.
   - Criptografia e descriptografia de mensagens.

1. Código-fonte documentado e de fácil entendimento, adequado para fins educacionais.

2. Relatório final com:

- Explicação do processo de implementação;
- Exemplos de uso;
- Análise de resultados e considerações sobre segurança.

7. RELATÓRIO COM AS LINHAS DE CÓDIGO DO PROGRAMA

Importar a biblioteca "random"

```
import random
```

A biblioteca "random" é usada para gerar números aleatórios. No código, gera valores aleatórios para chaves privadas e "nonces" (valores únicos para cifragem de cada caractere).

Parâmetros da curva elíptica

```
a = 2
b = 3
p = 97  # Campo finito
```

Esses parâmetros definem a equação da curva elíptica, y^2 = x^3 + ax + b mod p, onde "a", "b" são constantes da curva e "p" define um campo finito (número primo que limita o conjunto de pontos na curva).

Função para verificar se um ponto está na curva

```
def ponto_na_curva(x, y):
    return (y**2) % p == (x**3 + a * x + b) % p
```

Essa função verifica se um ponto (x, y) satisfaz a equação da curva elíptica, ou

seja, se o ponto está na curva. Se a igualdade for verdadeira, o ponto é válido.

Função de adição de pontos na curva elíptica

def add_pontos(P, Q):

Esta função realiza a adição de dois pontos, "P" e "Q", na curva elíptica:

1. Caso P == Q: Realiza a "duplicação" de ponto usando a fórmula de derivada para tangência de ponto.
2. Caso P != Q: Calcula a inclinação (m) entre os dois pontos e utiliza a fórmula de adição para obter o novo ponto (x_r, y_r).
3. A adição de pontos em curvas elípticas é uma operação essencial para criptografia em ECC.

Função para multiplicação escalar de um ponto

def multiplicacao_escalar(k, P):

Essa função multiplica um ponto "P" por um escalar "k" usando adições sucessivas de "P". Isso permite calcular o produto escalar k * P para gerar uma chave pública.A implementação usa uma técnica chamada "multiplicação rápida" para economizar operações, duplicando o ponto "P" em cada iteração.

Gerar chaves privadas e públicas

def gerar_chaves():

Essa função gera uma chave privada aleatória (número entre 1 e 255) e calcula a chave pública usando a multiplicação escalar com o ponto base P = (2, 22). A chave pública é um ponto na curva elíptica, gerado a partir da chave privada.

Função para criptografar uma mensagem

def cifra_com_chave_publica(mensagem, chave_publica):

Essa função criptografa uma mensagem usando a chave pública:
Para cada caractere na mensagem:
Gera um "nonce" aleatório.
Calcula o valor criptografado do caractere somando seu valor ASCII, o "nonce" e a coordenada "x" da chave pública.

Armazena o "nonce" e o valor criptografado em uma lista.

Função para descriptografar a mensagem

def descriptografar_mensagem(mensagem_criptografada, chave_privada):

Essa função reverte a criptografia usando a chave privada:
Com a chave privada, recria a chave pública.
Para cada par (nonce, char_criptografado):
Recupera o caractere original subtraindo "nonce" e "x" da chave pública do caractere criptografado.

Função para obter uma mensagem do usuário

def obter_mensagem():

Essa função limita a entrada do usuário a 128 caracteres, garantindo que a mensagem não exceda esse limite.

Função principal

def main():

A função "main()" organiza o fluxo do programa:

1. Solicita ao usuário a escolha entre criptografar ou descriptografar:

Criptografia:

Recebe a mensagem do usuário.

Gera as chaves pública e privada.

Criptografa a mensagem e exibe o resultado.

Descriptografia:

Recebe a mensagem criptografada e a chave privada.

Descriptografa a mensagem e exibe a mensagem original.

Por fim, "main()" é chamada para executar o programa.

```
import random

# Parâmetros da curva elíptica
a = 2
b = 3
p = 97  # Campo finito

# Função para verificar se um ponto está na curva
def ponto_na_curva(x, y):
    return (y**2) % p == (x**3 + a * x + b) % p

# Função de adição de pontos em uma curva elíptica
def add_pontos(P, Q):
    if P == Q:
        # Fórmula de duplicação de ponto
        m = (3 * P[0]**2 + a) * pow(2 * P[1], p - 2, p) % p
    else:
        # Fórmula de adição de dois pontos diferentes
        m = (Q[1] - P[1]) * pow(Q[0] - P[0], p - 2, p) % p

    x_r = (m**2 - P[0] - Q[0]) % p
    y_r = (m * (P[0] - x_r) - P[1]) % p

    return (x_r, y_r)

# Função para multiplicação escalar de um ponto
def multiplicacao_escalar(k, P):
    Q = None
    for i in range(256):  # Trabalhando com uma chave privada de 1 a 255
```

```
        if (k >> i) & 1:
            if Q is None:
                Q = P
            else:
                Q = add_pontos(Q, P)
        P = add_pontos(P, P)
    return Q

# Gerar chaves privadas e públicas
def gerar_chaves():
    chave_privada = random.randint(1, 255)  # Chave privada no intervalo de
1 a 255
    P = (2, 22)  # Ponto base arbitrário na curva
    chave_publica = multiplicacao_escalar(chave_privada, P)  #
Multiplicação escalar para gerar chave pública
    return chave_privada, chave_publica

# Função para criptografar mensagem (simulação)
def cifra_com_chave_publica(mensagem, chave_publica):
    mensagem_criptografada = []
    for char in mensagem:
        nonce = random.randint(1, 255)  # Gera um nonce aleatório para cada
caractere
        # 'Nonce' é um número aleatório usado para misturar a cifra
        x, y = chave_publica  # Pega a chave pública
        char_criptografado = (ord(char) + nonce + x) % 256  # Usa o valor
de x da chave pública para criptografar
        mensagem_criptografada.append((nonce, char_criptografado))  #
Armazena nonce e caractere cifrado
    return mensagem_criptografada

# Função de descriptografia utilizando chave privada e nonce
def descriptografar_mensagem(mensagem_criptografada, chave_privada):
    mensagem_original = []
    P = (2, 22)  # Ponto base na curva
    chave_publica = multiplicacao_escalar(chave_privada, P)
    for (nonce, char_criptografado) in mensagem_criptografada:
        # Recupera o valor original do caractere subtraindo a chave pública
        x, y = chave_publica  # Chave pública gerada pela chave privada
        char_original = chr((char_criptografado - nonce - x) % 256)
        mensagem_original.append(char_original)
    return ''.join(mensagem_original)

# Função para limitar a entrada a 128 caracteres
def obter_mensagem():
    mensagem = input("Digite a mensagem para criptografar (limite de 128
caracteres): ")
    # Limitar a mensagem a 128 caracteres
    if len(mensagem) > 128:
```

```
        print("Mensagem muito longa! A mensagem será limitada a 128
caracteres.")
        mensagem = mensagem[:128]
    return mensagem

# Função principal
def main():
    escolha = int(input("Escolha a operação (1 para criptografar, 2 para
descriptografar): "))

    if escolha == 1:
        mensagem = obter_mensagem()  # Obtém a mensagem com limite de 128
caracteres
        chave_privada, chave_publica = gerar_chaves()  # Gera chaves
aleatórias
        mensagem_criptografada = cifra_com_chave_publica(mensagem,
chave_publica)
        print(f"Mensagem criptografada: {mensagem_criptografada}")
        print(f"Chave privada (para descriptografar): {chave_privada}")
        print(f"Chave pública (para uso compartilhado): {chave_publica}")

    elif escolha == 2:
        # Solicita a entrada da mensagem criptografada e a chave privada
        mensagem_criptografada_str = input("Digite a mensagem criptografada
(formato lista de tuplas): ")
        chave_privada_str = input("Digite a chave privada para
descriptografar: ")

        # Tenta converter os inputs para os tipos corretos
        try:
            mensagem_criptografada = eval(mensagem_criptografada_str)
            chave_privada = int(chave_privada_str)

            # Descriptografa a mensagem
            mensagem_original =
descriptografar_mensagem(mensagem_criptografada, chave_privada)
            print(f"Mensagem original: {mensagem_original}")

        except Exception as e:
            print(f"Erro: {e}")

    else:
        print("Escolha inválida!")

# Rodar a função principal
main()
```

8. REFERÊNCIA BIBLIOGRÁFICA

BRASIL. Ministério da Saúde. Dengue. Brasília: Ministério da Saúde, 2024. Disponível em: https://www.gov.br/saude/pt-br/assuntos/saude-de-a-a-z/d/dengue. Acesso em: 20 abr. 2025.

SÃO PAULO (Estado). Centro Paula Souza. Campanha contra a dengue. São Paulo: Centro Paula Souza, 2025. Disponível em: https://www.cps.sp.gov.br/campanha-contra-a-dengue/. Acesso em: 21 abr. 2025.

VALLE, Denise; PIMENTA, Denise Nacif; CUNHA, Rivaldo Venâncio da (Orgs.). Dengue: teorias e práticas. Rio de Janeiro: Editora Fiocruz, 2015

9. FICHA DE ATIVIDADES PRÀTICAS SUPERVISIODAS

UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: Fernando C. Quinto TURMA: P RA: R0

CURSO: Ciência da computação CAMPUS: Limeira SEMESTRE: 1º TURNO:

CÓDIGO DA ATIVIDADE: 7607 SEMESTRE: 1º ANO GRADE: 2024

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)
26/3/24	Reunião em grupo	3	Fernando C. Quinto	3
5/4/24	Início do documento	4	Fernando C. Quinto	4
10/4/24	Conferindo requisitos do pequeno P. Rubins	3	Fernando C. Quinto	3
26/4/24	Conferindo requisitos do pequeno Lista	3	Fernando C. Quinto	3
3/5/24	Conferindo requisitos do pequeno Billing	3	Fernando C. Quinto	3
4/5/24	Conferindo requisitos do pequeno grupo	4	Fernando C. Quinto	4
10/5/24	Formatação do pequeno e relatório bibliográfico	5	Fernando C. Quinto	5
21/5/24	Organização de mapas mapa e site	6	Fernando C. Quinto	6

UNIP
UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRATICAS SUPERVISIONADAS - APS

NOME: Gustavo Braga TURMA: 01/CC1P39 RA: R1029H5

CURSO: 13701 Ciência da computação CAMPUS: Sorocaba SEMESTRE: 1º TURNO: Noturno

CÓDIGO DA ATIVIDADE: 76B7 SEMESTRE: 1º Semestre ANO GRADE: 2024

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
26/03	Reunião em grupo	3 horas	Gustavo Braga	03.	
05/04	Início do documento	3 horas	Gustavo Braga	03	
10/04	Pesquisas sobre poluição hídrica	4 horas	Gustavo Braga	04	
13/04	Pesquisas sobre poluição hídrica	4 horas	Gustavo Braga	04	
18/04	Fim das pesquisas de poluição hídrica	3 horas	Gustavo Braga	03	
19/04	Formatação das pesquisas para ABNT	3 horas	Gustavo Braga	03	
26/04	Pesquisas sobre Rio Tietê	4 horas	Gustavo Braga	04	
28/04	Pesquisas poluição do Rio Tietê	3 horas	Gustavo Braga	03	
02/05	Fim das pesquisas do Rio Tietê	2 horas	Gustavo Braga	02	
03/05	Pesquisa Represa Billings	3 horas	Gustavo Braga	03	
04/05	Pesquisa da Represa Billings e Guarapiranga	4 horas	Gustavo Braga	04	
05/05	Fim das pesquisas da Represa Billings	3 horas	Gustavo Braga	03	
06/05	Fim das pesquisas da Represa Guarapiranga	3 horas	Gustavo Braga	03	
10/05	Formatação e organização ABNT	2 horas	Gustavo Braga	02	
12/05	Organização de textos para o site	3 horas	Gustavo Braga	03	
21/05	Organização do sumário e lista	3 horas	Gustavo Braga	03	
23/05	Formatação e organização final	3 horas	Gustavo Braga	03	

TOTAL DE HORAS ATRIBUÍDAS: 53

AVALIAÇÃO: _____
Aprovado ou Reprovado

NOTA: _____

DATA: ___/___/_____

_____
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP
UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: Gustavo Yudi Watanabe TURMA: 01/CCLP39 RA: R080CFL

CURSO: Ciência da Computação CAMPUS: Anchieta SEMESTRE: 1º TURNO: Noturno

CÓDIGO DA ATIVIDADE: 76B7 SEMESTRE: 1º Semestre ANO GRADE: 2024

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
26/03	Reunião do grupo	3horas	Gustavo Yudi	03	
06/04	Pesquisa geral	3horas	Gustavo Yudi	03	
08/04	Resumo e abstract	2horas	Gustavo Yudi	02	
10/04	Pesquisa sobre poluição hídrica	3horas	Gustavo Yudi	03	
11/04	Pesquisa das principais causas da poluição	3horas	Gustavo Yudi	03	
13/04	Textos das principais causas	4horas	Gustavo Yudi	04	
12/04	Pesquisa das principais consequências	2horas	Gustavo Yudi	02	
12/04	Textos das principais consequências	2horas	Gustavo Yudi	02	
15/04	Pesquisa das soluções para a poluição	3horas	Gustavo Yudi	03	
15/09	Texto das soluções	4horas	Gustavo Yudi	04	
21/04	Pesquisa das leis	3horas	Gustavo Yudi	03	
21/04	Texto das leis	2horas	Gustavo Yudi	02	
23/04	Formatação do sumário	2horas	Gustavo Yudi	02	
07/05	Textos para o site	2horas	Gustavo Yudi	02	
09/05	Busca das imagens para o site e artefatos	1hora	Gustavo Yudi	01	
14/05	Projeto do Site	2horas	Gustavo Yudi	02	

(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.

TOTAL DE HORAS ATRIBUÍDAS: 27

AVALIAÇÃO: _____
Aprovado ou Reprovado

NOTA: _____

DATA: ____/____/_____

_____
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP
UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRATICAS SUPERVISIONADAS - APS

NOME: Kayky Santos de Sima TURMA: P RA: R103267

CURSO: Ciência da Computação CAMPUS: Anchieta SEMESTRE: 1º TURNO: Noturno

CÓDIGO DA ATIVIDADE: 7687 SEMESTRE: 1º Semestre ANO GRADE: 2024

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
26/03/24	Reunião em grupo	3	Kayky Santos de Sima	3	
06/05/24	Protótipo inicial do site com reunião de ideias	3	Kayky Santos de Sima	3	
08/05/24	Estudos do bootstrap para inclusão no site	3	Kayky Santos de Sima	3	
10/05/24	Inclusão do bootstrap e decisão do design	3	Kayky Santos de Sima	3	
11/05/24	Criação da página de introdução, conteúdo e imagens	8	Kayky Santos de Sima	8	
12/05/24	Correção inicial, verificação de links	6	Kayky Santos de Sima	6	
12/05/24	Verificação de erros e desenvolvimento de páginas	9	Kayky Santos de Sima	9	
13/05/24	Edição de novas imagens e otimização responsividade	4	Kayky Santos de Sima	4	

TOTAL DE HORAS ATRIBUÍDAS: 39 horas

AVALIAÇÃO: _____
Aprovado ou Reprovado

NOTA: _____

DATA: ____/____/_____

_____
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO