Universidade paulista
Faculdade de Ciências da Computação

CRIPTOGRAFIA

ABRAAO LUCAS QUEIROZ DA SILVA REZENDE – R086739

FELIPE DE OLIVEIRA SANTOS – R0533C0

FELIPE FERREIRA PICIN – R0822A6

FELIPE PEREIRA VESCIA – R1075I6

VINICIUS LOPES DE MORAIS - G005IJ0

SAO PAULO

2024

Universidade paulista
Faculdade de Ciências da Computação

CRIPTOGRAFIA

Trabalho para nota refente a materia de introducao a programacao estruturada

SAO PAULO

2024

ÍNDICE

Introdução

Com base no avanço da tecnologia, hardware, software e atualmente ganhando cada vez mais notoriedade e espaço no mercado, a inteligência artificial.

A cibersegurança tem sido cada vez mais afetada diretamente com esse desenvolvimento, sendo obrigada a acompanhar e reforçar suas stacks e métodos de criptografia dentro de suas atividades, como muitas empresas, instituições financeiras e governos estão adotando a automatização e aumentando o nível de atividades computacionais, desencadeando na máquina armazenando dados sensíveis e importantes de milhares de pessoas, ou seja, aumentando ainda mais a requisição por um sistema seguro e criptografado de ponta a ponta.

A consequência de uma internet insegura, é a fragilidade da segurança dos seus dados em sites ou navegações não criptografados, é a linha tênue dos seus dados protegidos ou expostos, isso para instituições é cada vez mais emergente, exposição de dados de usuários, informações governamentais e financeiras, é a decadência e a queda de uma organização como ao todo, tanto pública como privada.

Alinhado com a evolução tecnológica, a cibersegurança desenvolvendo novos métodos e algoritmos, é o caminho certo para um ambiente virtual seguro e criptografado.

CRIPTOGRAFIA SIMETRICA E ASSIMETRICA

A criptografia simétrica e criptografia assimétrica como os primeiros tópicos em uma introdução à criptografia se deve à importância fundamental desses conceitos. Eles formam a base de quase todas as aplicações práticas e teóricas em segurança da informação.

Criptografia simétrica

Criptografia simétrica é um método de proteção de informações em que o mesmo algoritmo e chave criptográfica são usados tanto para criptografar quanto para descriptografar os dados.

Características principais:

- **Chave única:** O remetente e o destinatário compartilham a mesma chave secreta.
- **Rápida e eficiente:** Geralmente, possui um desempenho mais rápido em comparação com a criptografia assimétrica, o que a torna adequada para grandes volumes de dados.
- **Segurança da chave:** A maior preocupação está na proteção da chave secreta. Caso ela seja interceptada ou descoberta, toda a comunicação fica comprometida.

Exemplos de algoritmos:

- **DES (Data Encryption Standard):** Um padrão mais antigo e atualmente considerado inseguro.
- **AES (Advanced Encryption Standard):** Muito usado hoje por ser rápido e altamente seguro.
- **3DES (Triple DES):** Uma versão mais segura do DES.
- **Blowfish e RC4:** Outras alternativas.

Aplicações:

- Transmissão de dados em redes locais.
- Arquivos criptografados.
- Conexões seguras, como em VPNs.

Por ser simples e eficiente, a criptografia simétrica é muito utilizada, mas requer um método seguro para o compartilhamento inicial da chave.

Criptografia assimétrica

Criptografia assimétrica: (ou criptografia de chave pública) é um método de criptografia que utiliza duas chaves diferentes para criptografar e descriptografar dados: uma chave pública e uma chave privada.

Características principais:

1. **Duas chaves diferentes:**

    a. **Chave pública:** Pode ser compartilhada livremente e usada para criptografar dados.
    b. **Chave privada:** É mantida em segredo e usada para descriptografar os dados criptografados com a chave pública correspondente.

2. **Segurança com base em problemas matemáticos:**
    a. A segurança da criptografia assimétrica depende de problemas matemáticos difíceis de resolver, como a **fatoração de grandes números primos** (utilizado no RSA) ou o **logaritmo discreto** (utilizado no algoritmo DH).

3. **Desempenho mais lento:**

    a. Comparado com a criptografia simétrica, a criptografia assimétrica tende a ser mais lenta, devido à complexidade das operações matemáticas necessárias para criptografar e descriptografar os dados.

4. **Usos principais:**

    a. **Autenticação e integridade:** Além de garantir a confidencialidade dos dados, a criptografia assimétrica também é utilizada para

autenticar a identidade do remetente e verificar a integridade da mensagem (por meio de assinaturas digitais).

b. **Troca de chaves segura:** A criptografia assimétrica é muitas vezes usada para trocar de forma segura uma chave simétrica, que então é usada para a criptografia de grandes volumes de dados (em sistemas como TLS/SSL).

Exemplos de algoritmos:

- **RSA:** Um dos algoritmos mais populares de criptografia assimétrica, baseado na fatoração de números grandes.
- **ECC (Elliptic Curve Cryptography):** Usa curvas elípticas e oferece maior segurança com chaves menores, sendo uma alternativa mais eficiente ao RSA.
- **DSA (Digital Signature Algorithm):** Usado para criar assinaturas digitais e garantir a integridade e autenticidade das mensagens.

Vantagens da criptografia assimétrica:

- **Segurança sem necessidade de compartilhamento de chave secreta:** A chave pública pode ser distribuída abertamente, enquanto a chave privada nunca precisa ser compartilhada.
- **Autenticação:** A chave privada pode ser usada para assinar digitalmente mensagens, permitindo que o destinatário verifique a identidade do remetente e a integridade da mensagem.

Desvantagens:

- **Desempenho:** Como mencionado, a criptografia assimétrica tende a ser mais lenta do que a simétrica, o que pode ser um problema em ambientes que exigem alta performance, como transmissões em tempo real.
- **Tamanho das chaves:** Para manter a segurança, as chaves precisam ser maiores do que as usadas na criptografia simétrica, o que também afeta a eficiência.

Aplicações:

- **SSL/TLS:** Protocolos usados para criptografar a comunicação na web (como em HTTPS).
- **Assinaturas digitais:** Para garantir a autenticidade e integridade de documentos e transações digitais.
- **P2P e criptomoedas:** Sistemas de pagamento como o Bitcoin dependem de criptografia assimétrica para transações seguras.

Em resumo, a criptografia assimétrica permite um sistema de comunicação segura onde as partes podem trocar informações sem a necessidade de compartilhar uma chave secreta, utilizando algoritmos complexos para garantir a segurança e autenticidade das comunicações.

Algoritmo AES (Rijndael)

No dia 2 de Janeiro de 1997, o órgão oficial norte-americano NIST – National Institute of Standards in Technology (Instituto Nacional de Normas em Tecnologia) anunciou formalmente um plano para definir um algoritmo como o novo padrão para criptografia, convidando qualquer pessoa a desenvolver um algoritmo. Definido que o novo padrão seria conhecido como AES, foi aberta assim uma espécie de “concurso”, com a condição que o vencedor não teria quaisquer direitos quanto à propriedade intelectual do algoritmo selecionado. Os critérios para avaliação dos algoritmos que viessem a concorrer seriam: segurança (sem nenhuma fraqueza algorítmica), desempenho (rápido em várias plataformas) e tamanho (não poderia ocupar muito espaço nem utilizar muita memória) [03]. Várias pessoas físicas e jurídicas desenvolveram seus algoritmos, e no dia 20 de agosto de 1998, o NIST selecionou 15 candidatos. Vários dos 15 algoritmos originais não duraram muito tempo, pois em alguns foram descobertas fraquezas e em outros simplesmente eram muito grandes ou muito lentos, reduzindo a lista em apenas 5 algoritmos até agosto de 1999, sendo eles o MARS, RC6, Rijndael, Serpent e Twofish. No ano seguinte, estes algoritmos foram testados para decidir qual deles seria o vencedor. Finalmente, no dia 2 de outubro de 2000 [03], o NIST anunciou como grande vencedor o algoritmo Rijndael [08] [09], desenvolvido por dois pesquisadores belgas, Vincent Rijmen e Joan Daemen. 16 O Rijndael é um algoritmo de criptografia de bloco simétrico (mesma chave de cifragem e decifragem), com tamanho (quantidade de bits de trabalho) de bloco e de chave variáveis, podendo ser especificados independentemente para 128, 192 ou 256 bits [08], possui facilidade de implementação, propiciando uso em Smart Cards (cartões magnéticos utilizados em operações bancárias ou de compra eletrônica) e outros equipamentos que utilizam pouca memória RAM, além disso, utiliza poucos ciclos de processamento [03]. O código desse algoritmo é bem enxuto e não depende de nenhum outro tipo de componente criptográfico, como gerador de

números randômicos. Esse aspecto faz com que sua utilização apresente um nível de segurança superior [09]. No AES o resultado das diversas operações intermediárias realizadas é colocado em um lugar denominado “matriz de estado”. Uma matriz de estado pode ser definida por uma matriz retangular de bytes, onde esta matriz possui quatro linhas por um número Nb (número de blocos) de colunas que é igual ao tamanho do bloco (128, 192, 256) dividido por 32 [09].

Especificação Galois Field (GF 28) Todos os bytes no algoritmo AES são interpretados como sendo elementos de um campo finito que podem ser representados por uma descrição polinomial, do tipo b7x 7 + b6x 6 + b5x 5 + b4x 4 + b3x 3 + b2x 2 + b1x+ b0, ou seja, o byte 57H {binário 01010111} pode ser representado pelo polinômio x6 + x4 + x2 + x + 1. Desta forma se torna possível toda uma descrição matemática de funções como adição, multiplicação e outras tratando-se agora o byte como polinômio e efetuando tais operações, como operações polinomiais. A representação polinomial de Galois Field possui uma representação mais fácil e permite uma representação numérica quando ao realizar operações polinomiais o resultado produz um outro polinômio de ordem maior do que os operandos originais. Desta maneira a teoria aplicada (GF 28) na dedução matemática usa do recurso de um polinômio irredutível (ou módulo), “x4 + 1”, com o qual se pode manter o resultado final com o mesmo índice dos operandos originais, reduzindo o resultado a partir deste polinômio [08] [09].

CONFIDENCIALIDADE DIGITAL: A FORÇA DO AES NA PROTEÇÃO DE DADOS

O aumento de casos de vazamento de dados sensíveis relatados nos últimos anos no Brasil trouxe a necessidade de uma proteção para estes. Uma solução para este novo problema, é a utilização de algoritmos de criptografia. O objetivo da criptografia é proteger os dados, de forma que apenas o emissor e o receptor consigam decifrá-los. Essa é uma necessidade antiga, que vem sendo aperfeiçoada

ao longo dos tempos. Vivendo em mundo altamente conectado, a criptografia é indispensável para garantir a confidencialidade dos dados inseridos nos meios digitais. Os aplicativos que são utilizados diariamente empregam criptografia para garantir que nossos dados sensíveis sejam protegidos contra vazamentos e uso indevido. A criptografia também está presente nas redes sem fio, as senhas cadastradas são criptografadas a fim de permitir a conexão somente a quem inserir a senha correta. A partir de testes realizados com alguns algoritmos de criptografia simétrica, conclui-se que a utilização do algoritmo AES (Advanced Encryption Standard) se torna mais viável do que os outros algoritmos testados, estes resultados foram obtidos a partir de análises de comparação do desempenho que estes algoritmos levam para criptografar um bloco de tamanho específico. O algoritmo AES é um algoritmo de cifra em bloco que suporta uma chave de 128, 192 ou 256 bits. O número de rodadas do sistema varia de acordo com o tamanho da chave, esse número pode ser de 10, 12 ou 14 rodadas respectivamente para os valores citados anteriormente. A estrutura do algoritmo utiliza quatro estágios: AddRoundKey, SubBytes, ShiftRows e MixColumns. O processo de cifragem inicia e finaliza com o estágio AddRoundKey, pois é o único que faz a utilização da chave, ou seja o processo só é reversível com o conhecimento da chave, trazendo mais segurança para o dado criptografado. SubBytes faz substituições com uma “caixa-S”, MixColumn realiza um embaralhamento de colunas e o ShiftRows irá fazer uma permutação. Busca-se conscientizar a importância da criptografia para a privacidade e interoperabilidade dos dados sensíveis que inserimos nas mais diversas plataformas online que é acessado diariamente, bem como apresentar o algoritmo AES, como uma melhor opção dentre os algoritmos de criptografia simétrica testados.

O AES (Advanced Encryption Standard) é um algoritmo de criptografia simétrica amplamente usado para proteger dados sensíveis. Ele funciona aplicando uma série de operações matemáticas sobre blocos de dados fixos, transformando informações legíveis (texto plano) em um formato incompreensível (texto cifrado). A segurança do AES depende de sua chave, que deve ser mantida em segredo e usada tanto para criptografar quanto para descriptografar os dados.

Como o AES funciona?

O AES é baseado em um tipo de criptografia chamado simétrica, onde a mesma chave é usada para ambos os processos. Ele opera em blocos de 128 bits (16 bytes) e permite chaves de três tamanhos diferentes: 128, 192 ou 256 bits. A escolha do tamanho da chave determina a segurança e o desempenho do algoritmo: quanto maior a chave, mais seguro é o processo, mas isso também exige mais recursos computacionais.

Etapas do AES

O processo de criptografia no AES é dividido em rodadas. O número de rodadas varia conforme o tamanho da chave:

- AES-128: 10 rodadas.
- AES-192: 12 rodadas.
- AES-256: 14 rodadas.

Cada rodada consiste em quatro etapas principais, que trabalham juntas para embaralhar os dados e garantir a segurança:

1. SubBytes: Cada byte do bloco de dados é substituído por outro, com base em uma tabela fixa (S-Box), introduzindo complexidade e dificultando ataques baseados em padrões.

2. ShiftRows: As linhas da matriz de dados são reorganizadas por deslocamentos cíclicos, embaralhando ainda mais a posição dos bytes.

3. MixColumns: As colunas da matriz são misturadas por operações matemáticas, espalhando os bits entre os bytes para reforçar a dispersão das informações.

4. AddRoundKey: O bloco de dados é combinado com uma parte da chave por meio de uma operação XOR, garantindo que a chave esteja integrada ao processo.

Na última rodada, o passo MixColumns é omitido para que a descriptografia seja mais eficiente.

Aplicação prática do AES

O AES é amplamente utilizado em diferentes contextos do dia a dia, principalmente para proteger dados digitais. Alguns exemplos incluem:

1. Comunicação segura: O AES é parte de protocolos como HTTPS, que protege dados transmitidos pela internet, e VPNs, que garantem conexões seguras.

2. Criptografia de dispositivos: Smartphones, laptops e discos rígidos usam AES para proteger arquivos e sistemas inteiros contra acessos não autorizados.

3. Redes Wi-Fi: O AES é uma base dos padrões de segurança modernos, como WPA2 e WPA3, usados para proteger redes sem fio.

4. Armazenamento na nuvem: Dados armazenados em serviços como Google Drive e Dropbox são frequentemente protegidos com AES.

5. Criptomoedas e blockchain: Muitos sistemas de blockchain utilizam o AES para proteger carteiras e transações.

Conclusão

O AES combina eficiência, segurança e flexibilidade, o que o torna uma escolha confiável para proteger informações sensíveis. Sua popularidade se deve ao fato de ser robusto contra ataques conhecidos e escalável, podendo ser usado em dispositivos móveis, sistemas empresariais e até mesmo em operações governamentais críticas. A prática de implementá-lo corretamente depende de gerenciar bem as chaves e garantir que o algoritmo seja usado em ambientes seguros.

```python
# Função de chave para garantir que a chave tenha 16 caracteres
def ajusta_chave(chave):
    return chave.ljust(16, ' ').encode('utf-8')[:16]

# Função de XOR para simular a nossas cifragem
def cifra_xor(mensagem, chave):
    chave = ajusta_chave(chave)
    # Converte a mensagem e a chave para uma lista de bytes
    mensagem_bytes = mensagem.encode('utf-8')
    resultado = bytearray()

    # XOR entre cada byte da mensagem e da chave que iremos colocar
    for i in range(len(mensagem_bytes)):
        resultado.append(mensagem_bytes[i] ^ chave[i % len(chave)])  # A chave é repetida se for menor do que o que precisamos

    return bytes(resultado)

# Função para decifrar, que é o mesmo processo de cifrar
def decifra_xor(mensagem_cifrada, chave):
    chave = ajusta_chave(chave)
    resultado = bytearray()

    # XOR entre cada byte da mensagem cifrada e da chave que iremos colocar
    for i in range(len(mensagem_cifrada)):
        resultado.append(mensagem_cifrada[i] ^ chave[i % len(chave)])  # A chave é repetida se for menor do que o que precisamos

    return bytes(resultado).decode('utf-8')

# Função principal onde podemos escrever a mensagem e a chave
def main():
    mensagem = input("Digite a mensagem que deseja cifrar: ")
    chave = input("Digite a chave (até 16 caracteres): ")
```



```python
    # Cifra a mensagem
    mensagem_cifrada = cifra_xor(mensagem, chave)
    print(f"Mensagem cifrada: {mensagem_cifrada.hex()}")  # Vai exibir a mensagem cifrada

    # Decifra a mensagem que escrevemos
    mensagem_decifrada = decifra_xor(mensagem_cifrada, chave)
    print(f"Mensagem decifrada: {mensagem_decifrada}")

# Chama a função principal
if __name__ == "__main__":
    main()
```

BIBLIOGRAFIA

IEEE XPLORE. Cybersegurança. Disponível em <https://ieeexplore.ieee.org/abstract/document/1201257> acesso em 10/11/2024

NIST. Cybersegurança. Disponível em <https://www.nist.gov/cybersecurity> acesso em 13/11/2024

Stack Exchange. Criptografia. Disponível em <https://crypto.stackexchange.com> acesso em 13/11/2024

MCTIC. Repositório. Algoritmo de Criptografia. Disponível em <https://repositorio.mctic.gov.br/bitstream/mctic/3478/3/2008_alessandro_campos_dissertacao.pdf> acesso em 11/11/2024