Relatório do Trabalho.

Gerador/Verificador RSA

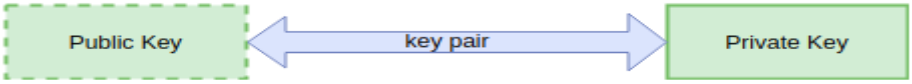
Heitor de Lima Belém – 16/0123950

02 de Dezembro de 2020 – 1º semestre

1. Introdução

A criptografia RSA (Rivest – Shamir – Adleman) é um dos algoritmos mais usados para criptografia segura de dados. É um algoritmo de criptografia assimétrico, que é apenas outra maneira de dizer “unilateral”. Nesse caso, é fácil para qualquer pessoa criptografar um dado, mas só é possível para alguém com a "chave" correta descriptografá-lo.

RSA funciona gerando uma chave pública e uma chave privada. As chaves públicas e privadas são geradas juntas e formam um par de chaves.



A chave pública pode ser usada para criptografar qualquer dado arbitrário, mas não pode descriptografá-lo.

A chave privada pode ser usada para descriptografar qualquer dado que foi criptografado por sua chave pública correspondente.

Isso significa que podemos dar nossa chave pública para quem quisermos. Eles podem criptografar todas as informações que desejam nos enviar, e a única maneira de acessar essas informações é usando nossa chave privada para descriptografá-las.

2. Descrição do trabalho

A primeira coisa que precisamos fazer é gerar o par de chaves pública e privada. Essas chaves são geradas aleatoriamente e serão usadas para todas as operações a seguir. Usamos a biblioteca padrão de criptografia do javascript (***crypto***) para gerar as chaves.

As variáveis *publicKey* e *privateKey* serão usadas para criptografia e descriptografia, respectivamente.

Criptografia / Descriptografia

Para fazer a encriptação de uma mensagem qualquer, foi usado o método *publicEncrypt*, que recebe alguns parâmetros:

1. Chave pública gerada (que ficou salva no arquivo em *./rsaKeys/public.pub*
2. O esquema de padding (foi utilizado o OAEP, como sugerido na especificação)
3. O algoritmo de hash SHA3
4. A mensagem a ser criptografada.

A única maneira de descriptografar os dados é usando a chave privada correspondente à chave pública com a qual eles foram criptografados. A biblioteca

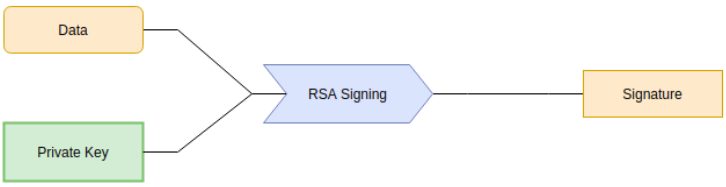
utilizada contém o método *privateDecrypt* que é usado para obter as informações originais dos dados criptografados. Esse método recebe como parâmetros:

1. Os dados criptografados
2. O algoritmo de hash usado na criptografia.
3. O esquema de padding usado.
4. A chave privada (somente o server tem conhecimento dela) correspondente à chave pública usada na encriptação.

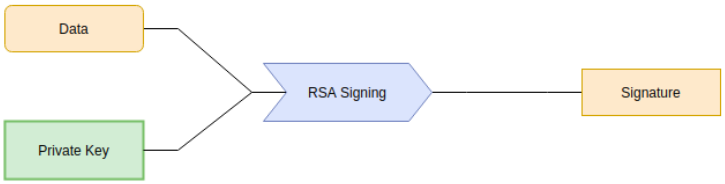
```
decrypt = (data) => {
  const decrypted = crypto.privateDecrypt(
    {
      key: fs.readFileSync('./rsaKeys/private.key'),
      padding: crypto.constants.RSA_PKCS1_OAEP_PADDING,
      oaepHash: "sha3-256",
      passphrase: "Top Secret"
    },
    data
  );
  fs.writeFileSync("./decryptedFiles/decryptedPlanText.txt", decrypted.toString("utf-8"));
  return decrypted;      You, a day ago • Add RSA first services
}
```

Assinatura / Verificação

As chaves RSA também são usadas para assinatura e verificação. A assinatura é diferente da criptografia, pois permite que você afirme a autenticidade, em vez da confidencialidade, ou seja, em vez de mascarar o conteúdo da mensagem original, um pedaço de dados é gerado a partir da mensagem, chamado de “assinatura”. Qualquer pessoa que tenha a assinatura, a mensagem e a chave pública pode usar a verificação RSA para se certificar de que a mensagem realmente veio da parte por quem a chave pública foi emitida. Se os dados ou assinatura não corresponderem, o processo de verificação falhará.



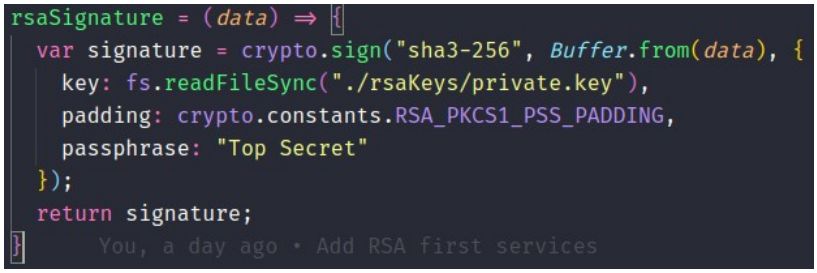
1. Assinatura RSA



2. Verificação RSA

Para fazer a assinatura e verificação das mensagens, foram usados os métodos *sign* e *verify* da biblioteca *crypto.* O método de assinatura recebe a mensagem que

queremos assinar, o algoritmo de hash e o esquema de padding usado, e gera uma assinatura na forma de bytes.



Já para a verificação, o método *verify* recebe o mesmo algoritmo de hash e padding usados na encriptação, além da assinatura e dos dados que queremos verificar de acordo com a assinatura e a chave pública. Retornando um *boolean* que identifica a autenticidade dos dados.

3. Conclusão

Neste trabalho foi visto como gerar chaves RSA públicas e privadas e como usá-las para criptografar, descriptografar, assinar e verificar dados arbitrários. Existem algumas limitações que devem ser consideradas antes de usar a encriptação RSA em seus dados, como o tamanho dos dados a serem criptografados em relação ao range de bits das chaves: a documentação do EncryptOEAP diz que a mensagem não deve ser maior do que o comprimento do módulo público menos duas vezes o comprimento do hash mais ou menos 2. Além de analisar o algoritmo de hash utilizado sobre os tipos de dados, se eles são críticos ou não na mensagem.