

SEMINÁRIO SEGURANÇA COMPUTACIONAL

Gerador e Verificador de
Assinaturas Virtuais

Lucas Rocha dos Santos - 211055325

Eduardo Marciano de Melo Meneses -
211055227

Gabriela de Oliveira Henriques -
211055254



GERAÇÃO E VERIFICAÇÃO DE ASSINATURAS

Geradores de Assinaturas

É uma ferramenta em que ocorre um processo para uma verificação de originalidade de uma mensagem.

Nesse processo, é utilizado um Hash para gerar um identificador de um arquivo.

O Hash é criptografado com uma chave privada do remetente, para assegurar que apenas ele pode encriptar a mensagem.

Verificadores de Assinaturas

O Verificador realiza o processo inverso de um gerador de assinaturas.

Ele recupera o Hash criado, usando o processo da descriptografia.

Com isso, o verificador realiza um Hash da mensagem recebida e compara o Hash passado com este criado, verificando a autenticidade do remetente.



RSA

Primeiro passo: encontrar dois números **p** e **q** que são primos, utilizando 1024 bits para cada um. Para gerar os primos foi utilizado o algoritmo Miller-Rabin. Nesse algoritmo é pego alguns números grandes e aleatórios. Esses números passam por testes de propriedades de números primos para ter certa garantia de que é um número primo. Ou seja, há um teste probabilístico.



Segundo passo: Com os dois números primos, é possível calcular $N = p \times q$. E também $\phi = (p-1) \times (q-1)$.

Com estes dois valores pode-se calcular o valor **d** e o valor **e**.

"**e**" é calculado encontrando um valor coprimo a ϕ .

"**d**" é inverso modular de "**e**" módulo ϕ .

Tendo os valores "**e**" e "**d**", temos as duas chaves sendo a chave pública (**e**, **n**) e a chave privada (**d**, **n**).

Terceiro passo: criptografar a mensagem. Esse processo é feito como $M^e \text{ mod } n$.

Quarto passo: para descriptografar, pegamos a mensagem cifrada (**C**) e utilizamos o processo de $C^d \text{ mod } n$.



FUNÇÃO HASH



SHA3

A função de hash utilizada foi o sha3, baseado no keccak e usamos duas variações de 512 bits e 256 bits.

Função Hash

Lembrando que a função Hash é um algoritmo que recebe uma entrada e comprime a mensagem seguindo algumas propriedades:

- Sem colisões – duas entradas diferentes precisam dar resultados diferentes.
- É impossível retornar para a entrada original depois que ela passa pela função.
- Possui saída com tamanho bem definido.
- Pseudoaleatoriedade.

Ser

OAEP

Técnica de Padding

É usado para inserir informações adicionais numa entrada para dificultar ataques de chosen plain-text.

Pseudo-Aleatoriedade

Como parte do padding, o OAEP utiliza de um número pseudo aleatório, adicionando uma difusão ao sistema.

Hash

OAEP utiliza funções Hash em seu funcionamento, neste caso foi utilizado sha3_512, ou seja, 512 bits.

Codificação do OAEP:

- a) O texto simples m é preenchido com k_1 zeros, anexando m a m' com comprimento de $n - k_0$ bits.
- b) r é convertido em uma string de $n - k_0$ bits por uma função de hash criptográfica G .
- c) $X = m' \oplus G(r)$.
- d) X é reduzido a k_0 bits por H .
- e) $Y = r \oplus H(X)$.
- f) O resultado do preenchimento é X e Y .

Decodificação do OAEP:

- a) r é recuperado por $r = Y \oplus H(X)$.
- b) m' é recuperado por $m' = X \oplus G(r)$.

Texto e imagem retirados e adaptados de: An Overview of RSA and OAEP Padding, Yutong Zhong

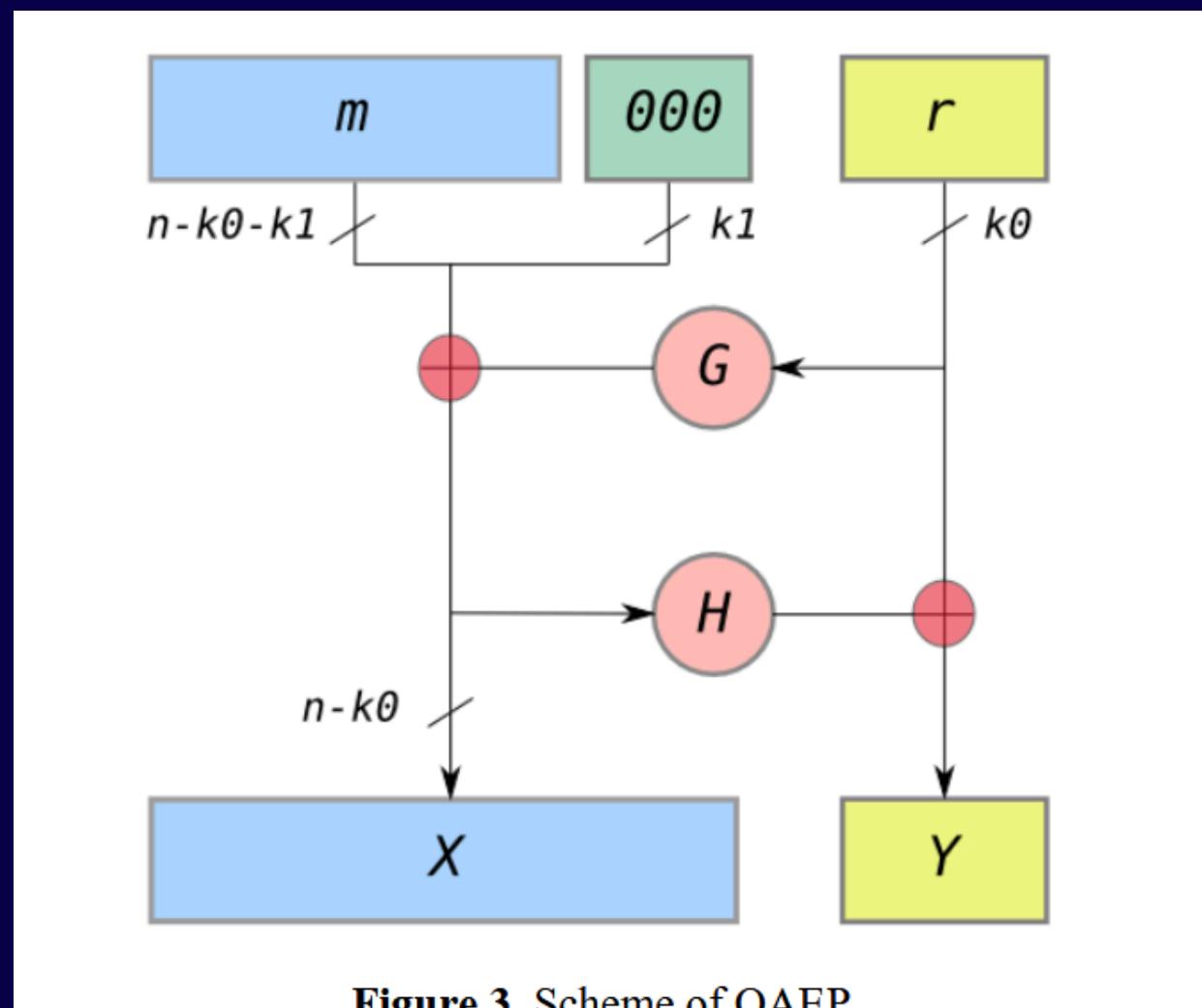


Figure 3. Scheme of OAEP

BASE 64

É uma base numérica utilizada para apresentação dos dados.

Ela utiliza 64 caracteres, sendo eles presentes no código na string:

"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"

Por exemplo, caso a mensagem fosse "2348", ela seria transformado para "ks".



JUNTANDO TODAS AS PARTES

Overview do processo

1. Tendo um arquivo, abrimos ele e lemos ele em binário. Esse será o plain_text.
2. É calculado o hash (h) da mensagem com a variável plain_text a partir do sha3 de 256 bits. Ou seja, $h = \text{sha3.hash256}(\text{plain_text})$.
3. Após isso, o hash h é encriptado utilizando o OAEP para o padding e, logo após isso, usando a encriptação do RSA (chamado de "e").
4. O plain_text é concatenado com o hash encriptado, formando a variável message_to_encrypt.
5. message_to_encrypt é convertido para base64, gerando a mensagem codificada.

Para decodificar a mensagem:

1. Primeiro a mensagem encriptada é convertida para retirar da base64.
2. O plain_text e o hash na mensagem recebida são separados. Com o plain_text recebido, é realizado um hash auxiliar para a parte de verificação.
3. O hash recebido é decriptado pelo algoritmo do RSA e OAEP, respectivamente.
4. É comparado o hash auxiliar calculado pelo destinatário com o hash recebido que foi decriptado no passo 3, validando se a mensagem, é autentica ou não.



OBRIGADA!



See You Next →