IA012 - Tarefa 6

Rodrigo Seiji Piubeli Hirao (186837)

16 de dezembro de 2021

01) A afirmação está incorreta, pois o DH é usado não para criptografar ou assinar, mas sim para gerar uma chave simétrica nova por meio de uma conversa sem que haja troca dessa chave.

02) Devem ser concordados em um corpo **p**, uma mensagem inicial α e enviados as chaves públicas geradas a partir das chaves privadas $k_{pu} = \alpha^{k_{pr}} \mod p$

03) O algoritmo DH serve para gerar chaves usando funções de mão única, sem que de fato seja enviada a chave de um recipiente a outro. Assim, ele é um algoritmo que usa de métodos criptográficos para uma troca de chaves simétricas.

04) O DH depende da segurança do método criptográfico de chave pública usado, sendo que este envia as chaves públicas calculadas a partir de uma chave privada. Sua segurança então, baseia-se no fato da chave privada não poder ser calculada a partir da chave pública, assim deve se ter cuidado de não expor ou reutilizar a chave privada, tanto quanto usar um algoritmo seguro com um tamanho considerado seguro para tal.

05) O problema do logaritmo discreto é o problema para se calcular o logaritmo de uma função com módulo, o que se prova muito difícil, o que torna o LDG genérico, é o fato da exponenciação (a função inversa ao logaritmo) não ser um ciclo de multiplicações, mas sim um ciclo de operações genéricas, por exemplo podemos definir a operação como uma somatória, logo $a^3 = a + a + a$, ou como qualquer operação, eis o nome logaritmo discreto generalizado.

06) Dado o grupo com a as matrizes 2x2 mod n, podemos definir a operação básica como a mupliplicação matricial

$$\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \cdot \begin{bmatrix} b_1 & b_2 \\ b_3 & b_4 \end{bmatrix} = \begin{bmatrix} a_1b_1 + a_2b_3 & a_1b_2 + a_2b_4 \\ a_3b_1 + a_4b_3 & a_3b_2 + a_4b_4 \end{bmatrix}$$

Podemos definir a exponenciação como a multiplicação 2 vezes

$$\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}^n = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \cdot \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \cdots \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \cdot \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$$

O que satisfaz a igualdade

$$\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}^{ab} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}^{ba}$$

E é uma operação de difícil reversão, até porque ela depende de uma exponencial normal no primeiro elemento, que já é difícil no RSA. Logo ela pode ser usada no DH.

07) Se fosse usado o grupo Z_p com uma operação de soma teríamos que

$$a^n \mod p = a \cdot a \ldots a \cdot a \mod p = a + a \ldots a + a \mod p = na \mod p$$

E como temos que

$$a^n \equiv c \mod p$$

Podemos descobrir o expoente **n** apenas calculando o inverso de **a** no grupo Z_p, tal que

$$n \equiv ca^{-1} (\mod p)$$

Já a curva elíptica possui uma operação muito mais imprevisível fazendo o logaritmo discreto ser muito mais difícil que apenas calcular o inverso da mensagem.

08) ele possuirá o módulo **p**, a base **g**, as chaves públicas $K_{puA} = g^{K_{prA}} \mod p$, $K_{puB} = g^{K_{prB}} \mod p$ e deverá descobrir $K_s = g^{K_{prA}K_{prB}}$, assim tendo que descobrir uma das chaves públicas K_{prA} ou K_{prB}, o que é o problema difícil do Logaritmo Discreto, tornando um ataque inviável. Porém caso não seja implementado corretamente podem haver problemas, como:

- **Chave menor que 2048 bits -** É possível fazer um ataque de peneira para descobrir os fatores, que é um golpe custoso, porém, se a chave tiver 1024 bits ou menos é possível pré calcular alguns valores antes, tornando o algoritmo viável.
- **Chave reutilizada -** Se a mesma chave for reutilizada é possível perceber que uma mensagem foi reenviada, pois sua cifra será a mesma
- **Chave não aleatória -** Se houver um padrão para geração da chave é possível descobrir depois de algumas tentativas

09) Os 2 algoritmos são iguais até o ponto em que geram a chave comum aos participantes $K_S = K_M = g^{ab} \mod p$, sendo que no ElGamal a chave pública é chamada de chave efêmera e após o cálculo da chave simétrica o algoritmo de ElGamal contém um passo a mais usando a chave como máscara com $y = x \cdot K_M \mod p$ para envio e decriptografação para autenticação da mensagem do outro lado.

Outro ponto é que o ElGamal agrupa o envio de mensagens, precisando apenas de 2 comunicações sendo enviado **g**, **p** e $K_{puB} = \beta$ na primeira comunicação e $K_{puA} = K_E$ e **y** na segunda.

10) Temos que

$$K_M = g^{ab} \mod p$$
$$y = x \cdot g^{ab} \mod p$$

Logo se o **a** for o mesmo em 2 comunicações temos que

$$\begin{cases} y_1 = x_1 \cdot g^{ab} \mod p \\ y_2 = x_2 \cdot g^{ab} \mod p \end{cases} \tag{1}$$

Pois b será sempre o mesmo, quem deve mudar é **a**, logo

$$\frac{y_1}{y_2} \equiv \frac{x_1 \cdot g^{ab}}{x_2 \cdot g^{ab}} \equiv \frac{x_1}{x_2} \mod p$$

Assim, sabendo apenas uma mensagem x_1, é possível descobrir outra mensagem x_2 apenas com

$$x_2 \equiv \frac{y_2}{y_1} x_1 \mod p$$