Universidade Tecnológica Federal do Paraná (UTFPR)
Departamento Acadêmico de Informática (DAINF)

Introdução à Criptografia

Professor: Rodrigo Minetto (rodrigo.minetto@gmail.com)

Lista de exercícios

1) Codifique um algoritmo de cifra de fluxo **COM** a função XOR (funções de deciframento e ciframento). Para testar seu programa decifre o arquivo "**cifra_xor_a.txt**". Note que a sequência de chaves geradas deve ser idêntica para que o deciframento ocorra com sucesso. Assim, utilize o programa "**aleatorio.c**" para gerar as chaves e não utilize nenhuma semente. Ignore espaços e quebras de linha. As letras a, b, ..., z são representadas pelos números 0, 1, ... 25 (assim para cada caractere do texto lido faça: caractere - 'a'), também force os números aleatórios a ficarem no intervalo 0, 1, ... 25 (aleatorio() % 26).

2) Modifique o programa do exercício 1 de tal forma a criar uma cifra de fluxo **assíncrona**. Experimente adicionar, remover e alterar um caractere no texto cifrado. Descreva os efeitos dessas modificações na cifra OTP síncrona e assíncrona. Para testar seu programa decifre o arquivo "**cifra_xor_b.txt**". Veja o esquema da OTP assíncrona nos slides da aula. Note que o caractere cifrado (no intervalo de 0 a 25) é utilizado como semente para a função de geração de chaves.

3) A primeira vista, podemos pensar que um ataque por exaustão de chaves pode funcionar contra um sistema OTP. Por exemplo, suponha uma mensagem de 5 caracteres ASCII, representada por 40 bits, que foi cifrada utilizando um OTP de 40 bits. Explique exatamente porque um ataque por exaustão de chaves não vai funcionar contra esse ciframento, mesmo que os recursos computacionais sejam infinitos. Isto é um paradoxo pois é provado que a cifra OTP é incondicionalmente segura. Nota: você tem que resolver esse paradoxo. Respostas como: O OTP é incondicionalmente seguro e por isso um ataque por força bruta não vai funcionar não são válidas.

4) Suponha que Alice e Bob desejam trocar uma mensagem mas não possuem meios seguros para trocar uma chave. Então Alice tem a seguinte ideia:

- Alice faz $c_1 = m \oplus a$, e envia c_1 para Bob.
- Bob faz $c_2 = c_1 \oplus b$, e envia c_2 para Alice.
- Alice faz $c_3 = c_2 \oplus a$, e envia c_3 para Bob.
- Bob faz $m = c_3 \oplus b$, e recupera a mensagem m de Alice.

Esse protocolo para troca de mensagens é seguro? Explique.

5) Suponha uma cifra OTP semanticamente segura com espaço de chaves $K = \{0,1\}^\ell$. Um banco deseja quebrar a chave de deciframeno $k \in \{0,1\}^\ell$ em duas partes p_1 e p_2 de tal forma que ambas as partes são necessárias para a decifragem. A parte p_1 deve ser dada para um executivo e a parte p_2 a outro de tal forma que ambos precisam combinar suas partes para prosseguir com o deciframento.

O banco produz aleatoriamente $k_1 \in \{0,1\}^\ell$ e faz ${k_1}^* = k \oplus k_1$. Observe que $k = {k_1}^* \oplus k_1$. Assim, o banco pode dar k_1 para um executivo e ${k_1}^*$ para outro, e ambos devem estar presentes para o deciframento, pois cada chave individual não contém informação sobre a chave secreta k (note que cada parte da chave é um ciframento OTP de k).

Agora, suponha que o banco deseja quebrar k em três partes p_1, p_2 e p_3 de tal forma que duas partes permitam o deciframento de k. Isto assegura que mesmo que um executivo estiver doente, o deciframento pode prosseguir. Para realizar tal tarefa o banco produz aleatoriamente $(k_1, {k_1}^*)$ e $(k_2, {k_2}^*)$ conforme explicado anteriormente e que respeite $k_1 \oplus {k_1}^* = k_2 \oplus {k_2}^* = k$. Como o banco deve distribuir as partes, de tal maneira que duas partes qualquer possam decifrar utilizando o k, mas nenhuma parte sozinha consiga decifrar?

- (a) $p_1 = (k_1, k_2)$, $p_2 = ({k_1}^*, {k_2}^*)$, $p_3 = ({k_2}^*)$
- (b) $p_1 = (k_1, k_2)$, $p_2 = (k_2, {k_2}^*)$, $p_3 = ({k_2}^*)$
- (c) $p_1 = (k_1, k_2)$, $p_2 = ({k_1}^*, k_2)$, $p_3 = ({k_2}^*)$
- (d) $p_1 = (k_1, k_2)$, $p_2 = (k_1, k_2)$, $p_3 = ({k_2}^*)$
- (e) $p_1 = (k_1, k_2)$, $p_2 = ({k_1}^*)$, $p_3 = ({k_2}^*)$