

Semana 4

Cifra de Ficheiro

Pretende-se cifrar o conteúdo de um ficheiro, assegurando a *confidencialidade* dos dados lá armazenados. Para tal iremos experimentar diferentes cifras, por forma a melhor percebermos as suas propriedades.

PROG: cfich_chacha20.py

Defina o programa cfich_chacha20.py que cifra um ficheiro usando a cifra sequencial [ChaCha20](#). O programa receba como argumentos:

- o tipo de operação a realizar: setup, enc ou dec
 - setup <fkey> cria ficheiro contendo uma chave apropriada para a cifra Chacha20 (com nome <fkey>)
 - enc <fich> <fkey> cifra ficheiro passado como argumento <fich>, usando a chave lida do ficheiro <fkey>. O criptograma resultante deverá ser gravado <fich>.enc (i.e. adiciona a extensão .enc ao nome do ficheiro de texto-limpo).
 - dec <fich> <fkey> decifra criptograma contido em <fich>, usando a chave lida do ficheiro <fkey>. Armazena o texto-limpo recuperado num ficheiro com nome <fich>.dec.

[!TIP] Note que o *NONCE* utilizado é requerido para decifrar o ficheiro. Deve por isso ser gravado juntamente com o criptograma.

QUESTÃO: Q2

Qual o impacto de se considerar um *NONCE* fixo (e.g. tudo 0)? Que implicações terá essa prática na segurança da cifra?

PROG: chacha20_int_attck.py

A cifra ChaCha20, por si só, não garante integridade dos dados. Por outro lado, por se tratar de uma **cifra sequencial síncrona**, não promove difusão da influência de troca de bits do criptograma.

O propósito do programa chacha20_int_attck.py é ilustrar como pode ser manipulada a informação cifrada pelo programa anterior - se soubermos um fragmento do conteúdo de uma dada posição do texto-limpo, podemos alterar essa informação. O programa chacha20_int_attck.py deve então receber os seguintes argumentos: <fctxt> <pos> <ptxtAtPos> <newPtxtAtPos>, sendo que <fctxt> é o nome do ficheiro contendo o criptograma; <pos> é a posição onde sabemos ter sido cifrado <ptxtAtPos>, e <newPtxtAtPos> é o que se pretende vir a obter quando se decifrar o ficheiro. O criptograma manipulado deve ser gravado no ficheiro com nome <fctxt>.attck.

PROG: cfich_aes_cbc.py e cfich_aes_ctr.py

Defina novas versões do programa que cifra ficheiros para utilizar a cifra por blocos [AES](#), nos modos [CBC](#) e [CTR](#).

[!NOTE] O modo CBC necessita que o texto limpo tenha um tamanho múltiplo do tamanho do bloco (16 byte, no caso do AES). Deve por isso usar [Padding](#).

QUESTÃO: Q3

Qual o impacto de utilizar o programa `chacha20_int_attck.py` nos criptogramas produzidos pelos programas `cfich_aes_cbc.py` e `cfich_aes_ctr.py`? Comente/justifique a resposta.

PROG: `pbenc_chacha20.py`

Armazenar segredos cryptográficas em ficheiros sem estarem devidamente protegidas **é uma má prática**. Em vez disso, deve-se:

1. Derivar os segredos a partir de uma *pass-phrase* com recurso a uma [Key Derivation Functions \(KDF\)](#) - o que se designa por **Password-Based Encryption**,
;
2. Armazenar em ficheiros devidamente protegidos (aka *keystore*), esta por sua vez recorrendo a *Password-Based Encryption* para a sua própria protecção.

Pretende-se assim alterar o programa `cfich_chacha20.py` para suportar *Password-Based Encryption*. Deixa portanto de existir o comando para gerar uma nova chave, e as opções `enc` e `dec` deixam de receber o nome do ficheiro da chave como argumento. Em vez disso, leem de `stdin` a *pass-phrase* que permitirá derivar a chave usada na cifra. Sugere-se a utilização da KDF [PBKDF2](#).