Semana 4

Cifra de Ficheiro

Pretende-se cifrar o conteÃºdo de um ficheiro, assegurando a *confidencialidade* dos dados lÃ¡ 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 operaÃ§Ã£o 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 deverÃ¡ ser gravado `<fich>.enc` (i.e. adiciona a extensÃ£o `.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 Ã© requerido para decifrar o ficheiro. Deve por isso ser gravado juntamente com o criptograma.

QUESTÃƒO: Q2

Qual o impacto de se considerar um *NONCE* fixo (e.g. tudo `0`)? Que implicaÃ§Ãµes terÃ¡ essa prÃ¡tica na seguranÃ§a da cifra?

PROG: `chacha20_int_attck.py`

A cifra `ChaCha20`, por si sÃ³, nÃ£o garante integridade dos dados. Por outro lado, por se tratar de uma **cifra sequencial sÃncrona**, nÃ£o promove difusÃ£o da influÃªncia de troca de bits do criptograma.

O propÃ³sito do programa `chacha20_int_attck.py` Ã© ilustrar como pode ser manipulada a informaÃ§Ã£o cifrada pelo programa anterior -- se soubermos um fragmento do conteÃºdo de uma dada posiÃ§Ã£o do texto-limpo, podemos alterar essa informaÃ§Ã£o. O programa `chacha20_int_attck.py`deve entÃ£o receber os seguintes argumentos: `<fctxt> <pos> <ptxtAtPos> <newPtxtAtPos>`, sendo que `<fctxt>` Ã© o nome do ficheiro contendo o criptograma; `<pos>` Ã© a posiÃ§Ã£o onde sabemos ter sido cifrado `<ptxtAtPos>`, e `<newPtxtAtPos>` Ã© 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 versÃµes 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 mÃºltiplo do tamanho do bloco (16 byte, no caso do AES). Deve por isso usar Padding.

QUESTÃƒO: 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 cryptogrÃ¡ficas em ficheiros sem estarem devidamente protegidas **Ã© uma mÃ¡ prÃ¡tica**. 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 prÃ³pria protecÃ§Ã£o.

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 opÃ§Ãµes `enc` e `dec` deixam de receber o nome do ficheiro da chave como argumento. Em vez disso, leem de `stdin` a *pass-phrase* que permitirÃ¡ derivar a chave usada na cifra. Sugere-se a utilizaÃ§Ã£o da KDF `PBKDF2`.