1.1

$Y_i = D(k)(x_i) \oplus RV$

1.2

a)

Ao contrário do CBC, padrões no texto claro poderão ser evidentes neste algoritmo, devido ao facto de não haver interdependência entre os blocos. O RV é sempre o mesmo para todo x_i.

Neste modo de operação os padrões no texto claro poderam ser evidentes porque blocos(x_i) iguais geram blocos cifrados(y_i) iguais. Isto não acontece no modo CBC porque a cifra nos blocos(x_i) é feita usando a informação cifrada do bloco anterior, ou o vetor inicial no caso do primeiro, o que faz com que blocos iguais geram blocos cifrados diferentes.

b)

Em termos de paralelização, o CBC não é paralelizável pois não é possível correr vários blocos da cifra paralelamente, porque necessitam do valor cifrado do bloco anterior, ou do vetor inicial no caso do primeiro. Contrariamente, na cifra definida neste exercício nenhum dos seus blocos depende da realização prévia de outro bloco.

2

Esta abordagem foi escolhida pois o custo computacional deste método é menor em comparação a um sistema totalmente assimétrico, garantindo maior segurança da chave simétrica e deixando de existir a necessidade de enviar a chave simétrica através de um canal seguro.

Para decifrar a mensagem, primeiro utiliza-se a chave privada do recetor para decifrar a chave encriptada, isto gera a session key que é depois usada para a decifra da cifra para a obter a mensagem.

3.1

Antes de gerar a assinatura é necessário definir as chaves privadas e publicas, estas são definidas utilizando as funções de iniciação initSign, que recebe por parâmetro a chave privada, e initVerify, que recebe por parâmetro a chave publica.

Para definir os dados a assinalar utiliza-se a função update e, depois de estar tudo definido é utilizada a função sign() para gerar a assinatura, o retorno da função.

Utilizando a chave definida na função initSign() e os dados colocados pela função update() é gerado uma assinatura que depois pode ser verificada.

3.2

No caso onde a função de hash está comprometida, isto é, onde é computacionalmente fazível, dado x, obter $x' \neq x$ tal que $MD5(x') = MD5(x)$, implica que existe a possibilidade de uma mensagem ser alterada na transmissão e ser considerada legitima quando for verificada, pois o resultado da função de hash MD5 é o mesmo para duas mensagens diferentes, causando assim problemas de autenticidade.

4.1

A chave necessária para validar a assinatura de um certificado não está presente em nenhum certificado com a exceção do certificado raiz. Isto dá-se devido ao facto de que para validar um certificado é necessário ir ao certificado anterior, o certificado a partir do qual o novo foi criado.

Como o certificado raiz não tem nenhum precedente, ele tem de conter a chave para se auto certificar.

4.2

Os certificados X.509 necessitam de uma chave pública e uma chave privada para garantir a sua autenticidade.

Se fosse usado um esquema MAC qualquer pessoa poderia criar certificados falsos que seriam reconhecidos como autenticados.

4.3

Os ficheiros ".cer" contem apenas a chave pública isto faz com que possam ser transmitidos por canais inseguros.

Os ficheiros ".pfx" contem tanto a chave pública como a privada por isso a sua transmissão tem de ser, estritamente, por canais seguros.

5 – Neste exercício decidimos que ao executar o programa fosse necessário passar por parâmetro o nome da função de *hash* e o ficheiro para o qual queres obter o *hash*, por esta ordem, depois de verificar-mos se os parâmetros existem fazemos o que é pedido e retornamos o valor do *hash* do ficheiro.

6 – Neste exercício decidimos que só é possível receber os parâmetros depois de começar o programa, isto é não são aceites, nem vistos, o que for passado por argumento. Utilizando a interface podemos então fazer o que nos é pedido no exercício, usando os algoritmos AES em modo GCM sem padding para cifra simétrica e RSA para cifra assimétrica. Foram escolhidos estes algoritmos pois foram falados em aula e pareceram adequados ao problema.