Instituto Superior de Engenharia de Lisboa

Licenciatura/Mestrado em Engenharia Informática e de Computadores

Segurança Informática

Primeira série de exercícios, Semestre de Inverno de 11/12

Data de entrega: 25 de Outubro de 2011

---

1. Considere o contexto dos esquemas e primitivas criptográficas:

   1.1. Considere um novo modo de operação definido por:

   - Seja $x = x_1, \ldots, x_L$ a divisão nos blocos x_i do texto em claro x.
   - RV é um vector aleatório, com a dimensão do bloco, gerado por cada texto em claro x.
   - Seja $y_i = E(k)(x_i \oplus RV)$, para $i = 1, \ldots, L$, onde E é a operação de cifra, $\oplus$ denota o ou-exclusivo bit a bit.

   i. Defina o algoritmo de decifra para este modo de operação.

   ii. Compare este modo de operação com o modo CBC quanto a: *a)* possibilidade de padrões no texto em claro serem evidentes no texto cifrado, *b)* capacidade de paralelizar a cifra.

   1.2. Comente a seguinte frase, adaptada do livro *Computer Security*: "Stream ciphers can process plaintext in chuncks that are smaller than the block size of the cipher algorithm".

   1.3. Porque razão a arquitectura dos esquemas assimétricos de cifra prevê a adição de aleatoriedade no texto em claro?

2. Considere o contexto dos esquemas criptográficos e da *Java Cryptography Architecture* (JCA):

   2.1. A protecção de um *keystore* é feita usando um esquema MAC. Descreva em detalhe esta utilização e o tipo de protecção obtido.

   2.2. O método `doFinal` da *engine class* `Cipher` pode ser chamado repetidamente. Após cada chamada, a instância volta ao estado em ficou após a iniciação (método `init`). Assim sendo, qual a razão para a existência do método `update`?

3. Considere a versão simplificada do protocolo *Kerberos* apresentada em seguida

   1. $A \rightarrow T : A, B, N_A$
   2. $A \leftarrow T : ticket_B, E_{kAT}(k, N_A, L, B)$
   3. $A \rightarrow B : ticket_B, authenticator_A$
   4. $A \leftarrow B : E_k(T_A)$

   onde $ticket_B = E_{kBT}(k, A, L)$, $authenticator_A = E_k(A, T_A)$; L é a validade de $ticket_B$ e T_A é a marca temporal de A.

   3.1. Quais os mecanismos existentes neste protocolo para a protecção contra ataques de *replay*?

   3.2. Tendo em consideração que B recebe o bilhete de A, pode B autenticar-se como A perante uma terceira entidade?

   3.3. A utilização da cifra no bilhete pode ser substituída por um esquema MAC (*Message Authentication Code*)? Se sim, indique como.

   3.4. A utilização da cifra no $authenticator_A$ pode ser substituída por um esquema MAC (*Message Authentication Code*)? Se sim, indique como.

4. Considere os certificados definidos pela norma X.509 e o protocolo *Secure Socket Layer* (SSL).

   4.1. De que forma é obtida a chave autenticada do emissor de um certificado C?

   4.2. De que forma o *record protocol* evita ataques de *replay*.

   4.3. Considere que o servidor malicioso S_1 realiza uma instância do protocolo *handshake* com o cliente C. Como é que este protocolo impede que S_1 se possa autenticar como C perante um outro servidor S_2, nomeadamente através do reenvio das mensagens que C enviou para S_1.

4.4. As mensagens do sub-protocolo *handshake* são trocadas sobre um canal inseguro. Qual a técnica utilizada neste sub-protocolo para detectar ataques de *man-in-the-middle*, no qual o atacante consegue interceptar e modificar as mensagens trocadas entre cliente e servidor.

5. Realize, na plataforma Java, uma aplicação para geração de *hashs* cripográficos de ficheiros. A aplicação recebe na linha de comandos *i)* o nome da função de *hash* e *ii)* o ficheiro para o qual se quer obter o *hash*. O valor de hash é enviado para o *standard output*.

   5.1. Teste a sua aplicação usando certificados (ficheiros `.cer`) presentes no arquivo `certificates-and-keys.zip`, em anexo a este enunciado. Compare o resultado com os valores de *hash* apresentados pelo visualizador de certificados do sistema operativo (ou outro da sua confiança).

6. Realize, na plataforma Java, uma aplicação para a cifra e decifra de ficheiros. Os requisitos da aplicação são:

   - Operação de cifra:
     - A operação de cifra recebe a localização do ficheiro a cifrar e produz dois ficheiros: um com o resultado da cifra e outro com os meta-dados da cifra.
     - A operação de cifra é também parametrizada por: ficheiro com o certificado X.509 do destinatário; directoria contendo certificados de autoridades de certificação intermédias; e *key store* com as âncoras de confiança utilizadas na validação das cadeia de certificados.
     - Os meta-dados devem incluir o certificado utilizado, por forma a permitir a selecção automática da chave no destinatário.
     - A cifra do conteúdo do ficheiro deve usar mecanismos simétricos, sendo os mecanismos assimétricos usados para o transporte da chave simétrica.
   - Operação de decifra:
     - A operação de decifra recebe a localização do ficheiro cifrado e do ficheiro com os meta-dados, e produz um ficheiro com o resultado da decifra.
     - A operação de decifra é também parametrizada pelo *key store* contendo as chaves privadas a utilizar no processo de decifra.

   Utilize os certificados e chaves privadas presentes no arquivo `certificates-and-keys.zip`, em anexo a este enunciado.

   6.1. Apresente um gráfico com o tempo de execução das operações cifra e decifra para ficheiros com aproximadamente $100Kbytes$, $1Mbytes$, $10Mbytes$ e $100Mbytes$.

7. Configure uma instalação do servidor HTTP *Apache* para aceitar ligações HTTPS (HTTP sobre SSL) com os seguintes requisitos:

   - Autenticação de cliente obrigatória;
   - Aceitação, como *trust anchor*, do certificado auto-assinado presente no ficheiro `CA2.raiz.jks`.

   A cadeia de certificados do servidor e a sua chave privada estão presentes no ficheiro `localhost.pfx`.

   O arquivo `Apache2.2.zip`, presente em anexo a este enunciado, contém os ficheiros necessários à configuração e execução do servidor *Apache*. Não é necessário realizar qualquer instalação, bastando copiar as pastas e ficheiros para uma directoria local e executar o *deamon* `httpd`.

ISEL, 27 de Setembro de 2011.