**Convocatoria ordinaria - Parcial 2**

1. Responde razonadamente a las siguientes cuestiones: [5]
   1. En un esquema de cifrado simétrico, ¿qué papel juega el vector de inicialización (IV)? ¿Por qué es necesario? ¿Por qué es necesario guardarlo junto con el criptograma?
   2. ¿Qué gran problema de la criptografía de clave secreta solucionan los criptosistemas de clave pública? ¿Por qué y cómo?
   3. ¿Por qué se impone la condición $e \cdot d = 1$ mod(phi(n)) en la generación de claves RSA? ¿Qué ocurriría si no se impusiera dicha condición?

**Solución:**

1. El vector de inicialización (IV) usado en CBC proporciona la propiedad conocida como **seguridad semántica**, que permite obtener criptogramas diferentes para los mismos textos en claro. Es necesario, por tanto, para evitar que CBC tenga los mismos problemas que ECB, por ejemplo, con las imágenes. Es necesario guardarlo junto con el criptograma para que el receptor del mensaje pueda descifrarlo.
2. La gran ventaja de la criptografía asimétrica es que mejora enormemente el problema de la **distribución de claves**. Lo consigue dividiendo la clave simétrica clásica en dos partes, conocidas como clave pública y privada, de forma que solo es necesario mantener secreta ésta última. La distribución mejora porque la clave pública puede distribuirse libremente por un canal inseguro.
3. La condición equivale a que $e$ y $d$ son inversos multiplicativos. Esto significa que se puede asegurar que cada operación de cifrado puede "deshacerse.en el descifrado:

$$c = m^e \mod n \longrightarrow (m^e)^d = m^{ed} = m^1 = m$$

Si no se impusiera esta condición, podría darse el caso de que algunos criptogramas no recuperaran el texto en claro correspondiente y, por tanto, todo el esquema de cifrado no funcionaría adecuadamente.

2. Imagina que debes diseñar un sistema de almacenamiento seguro de contraseñas, que utiliza el esquema salt+hash. Una de las entradas de la base de datos es: [5]

$$estudiante13 : de28ab : 1b55f246d72fbd7dbbe744499b7c3893f24f4126$$

donde cada item está separado por dos puntos (:), y corresponden a usuario, salt y hash, respectivamente.

Con estos datos responde razonadamente a las siguientes preguntas:

1. ¿Qué tipo de función hash se ha utilizado en este sistema? ¿Por qué? ¿Qué puedes decir respecto a su seguridad?
2. En cuanto al valor salt, ¿te parece suficiente para garantizar la seguridad del sistema? Respecto a un sistema basado solo en hash (sin salt), ¿en qué factor incrementa la complejidad de un ataque por fuerza bruta esta longitud de salt?
3. Imagina ahora que el sistema debe almacenar las credenciales del usuario 'profesor12' con la contraseña 'grandino'. Utilizando este sistema y una función hash h() cualquiera, ¿cómo quedaría la entrada del usuario?

**Solución:**

1. Para tratar de determinar el tipo de hash, podemos tratar de contar la longitud de la salida que produce. En este caso, son 20 bytes (160 bits), luego solo puede ser SHA1, que es el único tipo de hash con esta longitud. Se considera ya un tipo de hash inseguro, puesto que ya se han encontrado colisiones en el mismo (aunque no todavía .ª la carta", como MD5).

2. El valor salt es de solo 3 bytes, cuando se recomiendan valores de hasta 16 bytes. No cuesta mucho su almacenamiento y el incremento en seguridad frente a ataques offline es importante. Puesto que disponemos de 3 bytes, la complejidad del ataque se incrementa en un factor $256^3 = 16 = 16777216$.
3. Lo primero importante es saber que **el salt no se debe reutilizar**, por lo que sería crear un nuevo valor (y no reutilizar el anterior). La entrada quedaría:

$$profesor12 : salt : h(salt + "grandino")$$

donde '+' es el símbolo de la concatenación.