



PRÁCTICA: CRIPTOGRAFÍA

Objetivo:

Contestar un conjunto de preguntas / ejercicios relacionados con la materia aprendida en el curso.

Detalles:

En esta práctica el alumno aplicará las técnicas y utilizará las diferentes herramientas vistas durante el módulo. Para aquellos ejercicios para los que se haya usado código, se entregará el código junto con el documento de respuestas.

Cualquier password que sea necesaria tendrá un valor 123456.

Evaluación

Es obligatorio la entrega de un informe para considerar como APTA la práctica. Este informe ha de contener:

- Los enunciados seguido de las respuestas justificadas.
- En el caso de que se hayan usado comandos / herramientas también se deben nombrar y explicar los pasos realizados.

El código escrito para la resolución de los problemas se entrega en archivos separados junto al informe.

Se va a valorar el proceso de razonamiento aunque no se llegue a resolver completamente los problemas. Si el código no funciona, pero se explica detalladamente la intención se valorará positivamente.



Ejercicios:

1. Tenemos un sistema que usa claves de 16 bytes. Por razones de seguridad vamos a proteger la clave de tal forma que ninguna persona tenga acceso directamente a la clave. Por ello, vamos a realizar un proceso de disociación de la misma, en el cuál tendremos, una clave fija en código, la cual, sólo el desarrollador tendrá acceso, y otra parte en un fichero de propiedades que rellenará el Key Manager. La clave final se generará por código, realizando un XOR entre la que se encuentra en el properties y en el código.

La clave fija en código es A1EF2ABFE1AAEEFF, mientras que en desarrollo sabemos que la clave final (en memoria) es B1AA12BA21AABB12. ¿Qué valor ha puesto el Key Manager en properties para forzar dicha clave final?

La clave fija, recordemos es A1EF2ABFE1AAEEFF, mientras que en producción sabemos que la parte dinámica que se modifica en los ficheros de propiedades es B98A15BA31AE BB22. ¿Qué clave será con la que se trabaje en memoria?

2. Dada la clave con etiqueta “cifrado-sim-aes-256” que contiene el keystore. El iv estará compuesto por ceros binarios (“00”). Se requiere obtener el dato en claro correspondiente al siguiente dato cifrado:

```
zcFJxR1fzaBj+gVWFRAah1N2wv+G2P01ifrKejICaGpQkPnZMiexn3WXlGYX5WnNIosy  
KfkNKG9GGSgG1awaZg==
```

Para este caso, se ha usado un AES/CBC/PKCS7. Si lo desciframos, ¿qué obtenemos? ¿Qué ocurre si decidimos cambiar el padding a x923 en el descifrado? ¿Cuánto padding se ha añadido en el cifrado? Como truco, estudiar el resultado en hexadecimal, cuando lo imprimas en consola introducir un string para diferenciar el final, como “|*****|”.

Se valorará positivamente, obtener el dato de la clave desde el keystore mediante codificación en Python (u otro lenguaje).

3. Se requiere cifrar el texto “Este curso es de lo mejor que podemos encontrar en el mercado”. La clave para ello, tiene la etiqueta en el keyStore “cifrado-sim-chacha-256”. El nonce “9Yccn/f5nJJhAt2S”.

¿Cómo podríamos mejorar de forma sencilla el sistema, de tal forma, que no sólo garanticemos la confidencialidad sino, además, la integridad del mismo? Demuestra, tu propuesta por código, así como añade los datos necesarios para evaluar tu solución.



4. Tenemos el siguiente jwt, cuya clave es “KeepCoding”.

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c3VhcmlvIjoiRmVsaXBIIiJvZHIJcdT  
AwZWRndWV6Iiwicm9sIjoiaXNOY3JtYWwifQ.-  
KiAA8cjkamjwRUiNVHgGeJU8k2wiErdxQP_iFXumM8
```

¿Qué algoritmo de firma hemos realizado?

¿Cuál es el body del jwt?

Un hacker está enviando a nuestro sistema el siguiente jwt:

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c3VhcmlvIjoiRmVsaXBIIiJvZHIJcdT  
AwZWRndWV6Iiwicm9sIjoiaXNBZG1pbiJ9.-  
KiAA8cjkamjwRUiNVHgGeJU8k2wiErdxQP_iFXumM8
```

¿Qué está intentando realizar?

¿Qué ocurre si intentamos validarlo con pyjwt?

5. El siguiente hash se corresponde con un SHA3 Keccak del texto “En KeepCoding aprendemos cómo protegernos con criptografía”.

```
bced1be95fbd85d2ffce9c85434d79aa26f24ce82fbd4439517ea3f072d56fe
```

¿Qué tipo de SHA3 hemos generado?

Y si hacemos un SHA2, y obtenemos el siguiente resultado:

```
4cec5a9f85dcc5c4c6ccb603d124cf1cdc6dfe836459551a1044f4f2908aa5d63739506f646883  
3d77c07cfd69c488823b8d858283f1d05877120e8c5351c833
```

¿Qué hash hemos realizado?

Genera ahora un SHA3 Keccak de 256 bits con el siguiente texto: “En KeepCoding aprendemos cómo protegernos con criptografía.” ¿Qué propiedad destacarías del hash, atendiendo a los resultados anteriores?

6. Calcula el hmac-256 (usando la clave contenida en el KeyStore) del siguiente texto:

Siempre existe más de una forma de hacerlo, y más de una solución válida.

Se debe evidenciar la respuesta. Cuidado si se usan herramientas fuera de los lenguajes de programación, por las codificaciones es mejor trabajar en hexadecimal.



7. Trabajamos en una empresa de desarrollo que tiene una aplicación web, la cual requiere un login y trabajar con passwords. Nos preguntan qué mecanismo de almacenamiento de las mismas proponemos.

Tras realizar un análisis, el analista de seguridad propone un hash SHA-1. Su responsable, le indica que es una mala opción. ¿Por qué crees que es una mala opción?

Después de meditarlo, propone almacenarlo con un SHA-256, y su responsable le pregunta si no lo va a fortalecer de alguna forma. ¿Qué se te ocurre?

Parece que el responsable se ha quedado conforme, tras mejorar la propuesta del SHA-256, no obstante, hay margen de mejora. ¿Qué propondrías?

8. Tenemos la siguiente API REST, muy simple.

Request:

Post /movimientos

Campo	Tipo	Requiere Confidencialidad	Observaciones
idUsuario	Number	N	Identificador
Usuario	String	S	Nombre y Apellidos
Tarjeta	Number	S	

Petición de ejemplo que se desea enviar:

```
{"idUsuario":1,"usuario":"José Manuel Barrio Barrio","tarjeta":4231212345676891}
```

Response:

Campo	Tipo	Requiere Confidencialidad	Observaciones
idUsuario	Number	N	Identificador
movTarjeta	Array	S	Formato del ejemplo
Saldo	Number	S	Tendra formato 12300 para indicar 123.00



Moneda	String	N	EUR, DOLLAR
--------	--------	---	----------------

```
{
  "idUsuario": 1,
  "movTarjeta": [{
    "id": 1,
    "comercio": "Comercio Juan",
    "importe": 5000
  }, {
    "id": 2,
    "comercio": "Rest Paquito",
    "importe": 6000
  }],
  "Moneda": "EUR",
  "Saldo": 23400
}
```

Como se puede ver en el API, tenemos ciertos parámetros que deben mantenerse confidenciales. Así mismo, nos gustaría que nadie nos modificase el mensaje sin que nos enterásemos. Se requiere una redefinición de dicha API para garantizar la integridad y la confidencialidad de los mensajes. Se debe asumir que el sistema end to end no usa TLS entre todos los puntos.

¿Qué algoritmos usarías? ¿Serías capaz de hacer un ejemplo en Python de cómo resolverlo?

(Importante, de cara a calcular cualquier cosa, se deben quitar retornos de carro, espacios intermedios, usar las mismas comillas, etc)

9. El responsable de Raúl, Pedro, ha enviado este mensaje a RRHH:

Se debe ascender inmediatamente a Raúl. Es necesario mejorarle sus condiciones económicas un 20% para que se quede con nosotros.

Lo acompaña del siguiente fichero de firma PGP (MensajeRespoDeRaulARRHH.txt.sig). Nosotros, que pertenecemos a RRHH vamos al directorio a recuperar la clave para verificarlo. Tendremos los ficheros Pedro-priv.txt y Pedro-publ.txt, con las claves privada y pública.

Las claves de los ficheros de RRHH son RRHH-priv.txt y RRHH-publ.txt que también se tendrán disponibles.

Se requiere verificar la misma, y evidenciar dicha prueba.

Así mismo, se requiere firmar el siguiente mensaje con la clave correspondiente de las anteriores, simulando que eres personal de RRHH.



Viendo su perfil en el mercado, hemos decidido ascenderle y mejorarle un 25% su salario. Saludos.

Por último, cifra el siguiente mensaje tanto con la clave pública de RRHH como la de Pedro y adjunta el fichero con la práctica.

Estamos todos de acuerdo, el ascenso será el mes que viene.

10. Nuestra compañía tiene un contrato con una empresa que nos da un servicio de almacenamiento de información de videollamadas. Para lo cual, la misma nos envía la clave simétrica de cada videollamada cifrada usando un RSA-OAEP. El hash que usa el algoritmo interno es un SHA-256.

El texto cifrado es el siguiente:

```
7edee3ec0b808c440078d63ee65b17e85f0c1adbc0da1b7fa842f24fb06b332c1560
38062d9daa8ccfe83bace1dca475cfb7757f1f6446840044fe698a631fe882e1a6fc
00a2de30025e9dcc76e74f9d9d721e9664a6319eaa59dc9011bfc624d2a63eb0e449
ed4471ff06c9a303465d0a50ae0a8e5418a1d12e9392faaaf9d4046aa16e424ae1e2
6844bcf4abc4f8413961396f2ef9ffcd432928d428c2a23fb85b497d89190e3cfa49
6b6016cd32e816336cad7784989af89ff853a3acd796813eade65ca3a10bbf58c621
5fdf26ce061d19b39670481d03b51bb0eccc926c9d6e9cb05ba56082a899f9aa72f9
4c158e56335c5594fcc7f8f301ac1e15a938
```

Las claves pública y privada las tenemos en los ficheros clave-rsa-oaep-publ.pem y clave-rsa-oaep-priv.pem.

Si has recuperado la clave, vuelve a cifrarla con el mismo algoritmo. ¿Por qué son diferentes los textos cifrados?

11. Nos debemos comunicar con una empresa, para lo cual, hemos decidido usar un algoritmo como el AES/GCM en la comunicación. Nuestro sistema, usa los siguientes datos en cada comunicación con el tercero:

Key: E2CFF885901B3449E9C448BA5B948A8C4EE322152B3F1ACFA0148FB3A426DB72
Nonce: 9Yccn/f5nJJhAt2S

¿Qué estamos haciendo mal?