27. Cifrado Asimétrico con Clave Pública y Privada

PARTE 1 — “IDENTIDAD CRIPTOGRÁFICA” (Par de claves)

FASE 1 — Crear carpeta de misión

Crear una identidad asimétrica (clave pública + clave privada)

- mkdir -p mision_canal_seguro
- cd mision_canal_seguro

```
ub@forense-ai:~$ mkdir -p mision_canal_seguro
ub@forense-ai:~$ cd mision_canal_seguro
ub@forense-ai:~/mision_canal_seguro$
```

FASE 2 — Generar clave privada del oficial Bob (2048 bits)

- openssl genrsa -out bob_privada.pem 2048

```
ub@forense-ai:~/mision_canal_seguro$ openssl genrsa -out bob_privada.pem 2048
```

Verificar clave privada

- openssl rsa -in bob_privada.pem -check -noout

```
ub@forense-ai:~/mision_canal_seguro$ openssl rsa -in bob_privada.pem -check -noout
RSA key ok
```

FASE 3 — Obtener la clave pública de Bob

- openssl rsa -in bob_privada.pem -pubout -out bob_publica.pem

```
ub@forense-ai:~/mision_canal_seguro$ openssl rsa -in bob_privada.pem -pubout -out bob_publica.pem
writing RSA key
```

Ver contenido:

- openssl rsa -pubin -in bob_publica.pem -text -noout

```
ub@forense-ai:~/mision_canal_seguro$ openssl rsa -pubin -in bob_publica.pem -text -noout
Public-Key: (2048 bit)
Modulus:
    00:ba:f0:f6:dd:94:63:8a:71:45:7d:40:8c:9a:f1:
    91:6d:3a:7c:57:c9:4a:af:57:c4:11:a0:3f:d5:23:
    c1:a2:ab:16:5b:ee:44:a0:c8:59:3e:3b:f0:34:70:
    eb:b5:bd:ef:7b:30:3a:ef:93:23:79:9c:b4:ab:d5:
    b2:cf:ee:ed:45:1a:7a:50:33:f3:92:bc:99:29:da:
    d5:08:7a:c0:06:ef:c6:6a:79:58:1a:73:9d:36:19:
    f6:43:d0:f9:0a:5a:18:0b:e1:02:50:f9:57:11:f6:
    8b:8b:12:2a:e6:6b:98:75:e0:c7:4f:76:5e:a3:ab:
    dd:e9:ab:e3:8f:df:95:15:72:4c:8e:c7:c9:50:ba:
    de:59:8e:fb:1f:62:89:d9:ac:70:76:1c:26:cf:cc:
    b6:39:66:3d:0c:a0:ea:b7:19:e1:34:8e:f5:fb:b7:
    5e:c8:7e:03:c4:b0:0f:ed:3d:80:7a:e4:ea:96:11:
    7e:29:c5:84:bd:c5:11:93:5c:9c:76:58:38:8e:d3:
    bc:69:ef:4c:54:30:bd:57:f5:74:8b:d3:c8:a4:a7:
    ff:e0:f7:fb:a1:9a:04:33:7e:5a:11:92:ac:c2:c7:
    96:f5:68:e0:e5:44:f8:32:a2:34:13:68:ce:48:bb:
    b1:55:78:d1:29:a5:20:db:2d:a4:ff:02:1a:7c:24:
    02:a7
Exponent: 65537 (0x10001)
```

Resultado esperado: los alumnos ven que la pública se puede inspeccionar sin revelar la privada.

PARTE 2 — “CIFRADO CON CLAVE PÚBLICA” (Confidencialidad)

FASE 4 — Crear mensaje secreto de Alice

- echo "Coordenadas del punto de encuentro: Sector 7G. Acceso nivel Alfa." > mensaje_alice.txt
- cat mensaje_alice.txt

```
ub@forense-ai:~/mision_canal_seguro$ echo "Coordenadas del punto de encuentro: Sector 7G. Acceso nivel
 Alfa." > mensaje_alice.txt
ub@forense-ai:~/mision_canal_seguro$ cat mensaje_alice.txt
Coordenadas del punto de encuentro: Sector 7G. Acceso nivel Alfa.
```

FASE 5 — Cifrar para Bob usando su clave pública

- openssl pkeyutl -encrypt -pubin -inkey bob_publica.pem -in mensaje_alice.txt -out mensaje_para_bob.bin

```
ub@forense-ai:~/mision_canal_seguro$ openssl pkeyutl -encrypt -pubin -inkey bob_publica.pem -in mensaj
e_alice.txt -out mensaje_para_bob.bin
```

Ver resultado del cifrado: datos binarios “ruido”.

- xxd mensaje_para_bob.bin | head

```
ub@forense-ai:~/mision_canal_seguro$ xxd mensaje_para_bob.bin | head
00000000: 5e05 f51e 321a edc8 599e f388 e532 b050  ^...2...Y....2.P
00000010: 2313 6130 d5b6 9f27 a221 9527 bcb0 b55d  #.a0...'.!.'...]
00000020: ada9 ee92 4790 9de9 738e d1e6 695b 3f7d  ....G...s...i[?}
00000030: 0c62 9fc0 f412 ae30 9235 99e8 7368 75aa  .b.....0.5..shu.
00000040: 1aa7 09be 089b eadc 5db7 1fe4 61b6 bc21  ........]...a..!
00000050: 400d 0fa9 0225 e775 2d04 f68f b8d6 aba4  @....%.u-.......
00000060: 2585 2fbf d5ca 52d2 8568 498a 754c bc84  %./...R..hI.uL..
00000070: d5c9 ff46 1220 18cb c295 2a04 88c0 0172  ...F. ....*....r
00000080: 20eb eb9a 8ef4 d502 6b7e 4afd b237 da2d   .......k~J..7.-
00000090: 0891 45a5 540e 8a55 7811 20d6 97cc df8e  ..E.T..Ux. .....
```

FASE 6 — Descifrar con la clave privada de Bob

El texto original vuelve intacto

- openssl pkeyutl -decrypt -inkey bob_privada.pem -in mensaje_para_bob.bin -out mensaje_descifrado.txt
- cat mensaje_descifrado.txt

```
ub@forense-ai:~/mision_canal_seguro$ openssl pkeyutl -decrypt -inkey bob_privada.pem -in mensaje_para_
bob.bin -out mensaje_descifrado.txt
ub@forense-ai:~/mision_canal_seguro$ cat mensaje_descifrado.txt
Coordenadas del punto de encuentro: Sector 7G. Acceso nivel Alfa.
```

PARTE 3 — “FIRMA DIGITAL” (Integridad + Autoría)

FASE 7 — Alice firma un mensaje

- openssl genrsa -out alice_privada.pem 2048
- openssl rsa -in alice_privada.pem -pubout -out alice_publica.pem

```
ub@forense-ai:~/mision_canal_seguro$ openssl genrsa -out alice_privada.pem 2048
ub@forense-ai:~/mision_canal_seguro$ openssl rsa -in alice_privada.pem -pubout -out alice_publica.pem
writing RSA key
```

FASE 8 — Crear el hash y firmarlo (firma con clave privada)

Crear archivo:

- echo "Orden de misión: activar escudos y mantener posición." > orden.txt

```
ub@forense-ai:~/mision_canal_seguro$ echo "Orden de misión: activar escudos y mantener posición." > orden.txt
```

Generar firma:

- openssl dgst -sha256 -sign alice_privada.pem -out orden.firma orden.txt

```
ub@forense-ai:~/mision_canal_seguro$ openssl dgst -sha256 -sign alice_privada.pem -out orden.firma orden.txt
```

FASE 9 — Verificar la firma con la clave pública

- openssl dgst -sha256 -verify alice_publica.pem -signature orden.firma orden.txt

```
ub@forense-ai:~/mision_canal_seguro$ openssl dgst -sha256 -verify alice_publica.pem -signature orden.firma orden.txt
Verified OK
```

FASE 10 — Simular sabotaje (modificación del mensaje)

Modifica el archivo:

- echo "Orden de misión: desactivar escudos y abrir bahía de carga." > orden.txt

```
ub@forense-ai:~/mision_canal_seguro$ echo "Orden de misión: desactivar escudos y abrir bahía de carga." > orden.txt
```

Volver a verificar:

- openssl dgst -sha256 -verify alice_publica.pem -signature orden.firma orden.txt

```
ub@forense-ai:~/mision_canal_seguro$ openssl dgst -sha256 -verify alice_publica.pem -signature orden.firma orden.txt
Verification failure
4097D4B2147F0000:error:02000068:rsa routines:ossl_rsa_verify:bad signature:../crypto/rsa/rsa_sign.c:430:
4097D4B2147F0000:error:1C880004:Provider routines:rsa_verify:RSA lib:../providers/implementations/signature/rsa_sig.c:774:
```

PARTE 4 — “CERTIFICADOS” (Confianza en la clave pública)

FASE 11 — Crear la CA de la Flota Estelar

Generar clave privada CA:

- openssl genrsa -out starfleet_ca_privada.pem 4096

```
ub@forense-ai:~/mision_canal_seguro$ openssl genrsa -out starfleet_ca_privada.pem 4096
```

Generar un certificado autofirmado:

- openssl req -x509 -new -key starfleet_ca_privada.pem -sha256 -days 365 -out starfleet_ca.crt \
  -subj "/C=ES/O=Starfleet Academy/OU=Security Division/CN=Starfleet Root CA"

```
ub@forense-ai:~/mision_canal_seguro$ openssl req -x509 -new -key starfleet_ca_privada.pem -sha256 -days 365 -out starfleet_c
a.crt \
-subj "/C=ES/O=Starfleet Academy/OU=Security Division/CN=Starfleet Root CA"
```

Ver certificado:

- openssl x509 -in starfleet_ca.crt -text -noout | head -n40

```
ub@forense-ai:~/mision_canal_seguro$ openssl x509 -in starfleet_ca.crt -text -noout | head -n40
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            4d:9c:09:a7:bb:93:59:46:b1:16:30:36:43:fa:f7:9d:07:84:c1:ee
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = ES, O = Starfleet Academy, OU = Security Division, CN = Starfleet Root CA
        Validity
            Not Before: Feb 17 21:14:10 2026 GMT
            Not After : Feb 17 21:14:10 2027 GMT
        Subject: C = ES, O = Starfleet Academy, OU = Security Division, CN = Starfleet Root CA
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:d6:c7:5d:68:0f:6e:74:7d:6a:5d:7e:4d:c4:8f:
                    16:a5:02:f1:ed:91:89:fe:04:d2:d6:50:51:14:aa:
                    56:a4:9a:ea:c1:1e:32:eb:4d:04:44:5b:07:7c:76:
                    8a:ab:c0:56:f4:e9:3b:e0:77:d0:90:ef:a3:35:6b:
                    48:db:08:96:ee:77:fa:ea:b0:d0:27:04:46:66:0e:
                    33:77:fe:c3:3a:a1:85:6c:db:d4:a2:12:f1:34:be:
                    44:19:3d:1f:97:9c:53:93:ff:43:11:a6:f9:43:a7:
                    fa:55:ac:1c:62:ef:22:63:11:04:d9:85:2d:49:24:
                    67:b7:62:3f:58:70:cd:32:8e:f3:f0:2f:66:78:8f:
                    2b:3d:92:5e:b2:7b:5d:1f:31:0c:2b:70:5d:30:af:
                    fe:7f:aa:e5:7b:a7:f0:e7:78:94:08:46:84:d5:54:
                    ba:70:44:a5:11:4f:c3:63:19:a3:3f:a9:32:0f:b6:
                    ba:8e:a5:69:7d:8f:ce:62:df:16:0a:63:bc:8d:ef:
                    22:99:89:c5:2f:69:08:89:3a:ea:a4:e9:d9:7d:28:
                    02:78:de:50:d2:77:14:72:62:9f:11:fc:1f:6c:28:
                    ab:9c:f1:24:fa:ec:c6:03:08:99:11:05:2e:e3:d4:
                    89:4f:38:dc:fa:32:19:89:83:dd:e7:8b:ad:15:5a:
                    a7:88:35:43:7f:2d:fb:de:b2:b8:a9:20:52:99:3c:
                    69:76:32:ac:ef:20:b9:e7:b1:06:41:39:e9:7d:81:
                    02:86:6b:c4:eb:8b:82:2c:ea:4d:d4:5b:6e:ae:1d:
                    b9:8a:eb:d9:79:61:35:5a:e1:f5:33:f3:1a:b1:ed:
                    43:42:d0:ba:19:24:41:42:bb:97:96:a4:8d:d8:4b:
                    1d:f4:65:a4:26:2d:00:99:8d:f6:5d:cf:38:10:e4:
                    7e:cf:00:d3:3c:ea:07:2d:78:fc:2c:05:1d:2e:7e:
                    a2:81:cb:ed:4f:fb:74:e2:d4:42:c7:e3:ec:af:8d:
ub@forense-ai:~/mision_canal_seguro$
```

FASE 12 — Alice solicita un certificado (CSR)

CSR = Certificate Signing Request (petición de certificado).

- openssl req -new -key alice_privada.pem -out alice.csr \
  -subj "/C=ES/O=Starfleet Academy/OU=Communications/CN=Alice"

```
ub@forense-ai:~/mision_canal_seguro$ openssl req -new -key alice_privada.pem -out alice.csr \
-subj "/C=ES/O=Starfleet Academy/OU=Communications/CN=Alice"
```

FASE 13 — La CA firma el certificado de Alice

- openssl x509 -req -in alice.csr -CA starfleet_ca.crt -CAkey starfleet_ca_privada.pem \
  -CAcreateserial -out alice_cert.crt -days 365 -sha256

```
ub@forense-ai:~/mision_canal_seguro$ openssl x509 -req -in alice.csr -CA starfleet_ca.crt -CAkey starfleet_ca_privada.pem \
-CAcreateserial -out alice_cert.crt -days 365 -sha256
Certificate request self-signature ok
subject=C = ES, O = Starfleet Academy, OU = Communications, CN = Alice
```

Ver contenido:

- openssl x509 -in alice_cert.crt -text-noout | head -n60

```
ub@forense-ai:~/mision_canal_seguro$ openssl x509 -in alice_cert.crt -text-noout | head -n60
-----BEGIN CERTIFICATE-----
MIIEOjCCAiICFBtjRIdWckY3kvdytheYVaGJ3zMKMA0GCSqGSIb3DQEBCwUAMGEx
CzAJBgNVBAYTAkVTMRowGAYDVQQKDBFTdGFyZmxlZXQgQWNhZGVteTEaMBgGA1UE
CwwRU2VjdXJpdHkgRGl2aXNpb24xGjAYBgNVBAMMEVN0YXJmbGVldCBSb290IENB
MB4XDTI2MDIxNzIxMTgwM1oXDTI3MDIxNzIxMTgwM1owUjELMAkGA1UEBhMCRVMx
GjAYBgNVBAoMEVN0YXJmbGVldCBBY2FkZW15MRcwFQYDVQQLDA5Db21tdW5pY2F0
aW9uczEOMAwGA1UEAwwFQWxpY2UwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEK
AoIBAQC8Hf9zCBcy0VpNDaeuCgxRCYJHlhsDD1VTtavnOHpVFSWIlFmY/BhAm+MS
slwVS4NpkAXcwNklwJaE5WUdWiXx817RwzqXw2mY5sr/8s3jvW1QJjxt5sjF4uhe
O+JGKrOd3BYVVbD9D8AQ5JwDzCR4MOAya4WRZO3EwaH4DqLHLf0vFfIQBp6wUDKM
BQUwCa5THI5KI+LTj5zC/o2qSBxbCXmiy8+6biqyPMzcG2/BBjbVdxN5V+SQfEmD
xqqBzkB9G2RXl8URBTmASGH0nbZxM45AtSAiqiVELWHcpupByQNga9TX166gWxbP
7olLFQ4jP9G+CtMgG9R/pfORnTKRAgMBAAEwDQYJKoZIhvcNAQELBQADggIBALVi
Z1b7jvkroyyqPCrbdcni3gVhumaZJibBnjPiShRlHlpxrIxhtQ29oKrum3QylspI
GKHRRMRgUG7vWVUiCricMKghlE7kedSwo3c413oZUhi5GPoc0a0GyT0qBjr+OJvI
Ffa5SQNK9sJO58KLBaoaB8FZcxjyGsMOVGE+7bAn200bUT0BeiwHejdfhQA963YY
zc/on14alkBmKbt/QgqfV4IXR1t/QvQ0O8HMTAQyQY3LVBKvhlmx0z48HzTOWwhN
vNPW6Vza6NgjbvkXs1RR6vOsBpJpRhJ0qrW+RaOiHDqzku4VZHkTkSErllWBG4VF
MLL1HVBxEswZnjIaMldIeJBWDD1tL4W3jO7pXOSfo5mZyaEzW+4K7OX/liihuPP+
zASRfOxpjbXTMln241BhRK6tJc86MYdvVOaPn8MT7580YFoazooNGftq1K9NffJd
h4stQ80bP9cbtQMnt/qhYLmBie23nbKD7ZQ+L/630XaRbcDiztdTsKnhIrhbR/z0
bM4QY+WSUdFfUfwQXWK7kIN/HQ43mv00wcoOxzic2FvNf4h+8jhxcTIKQm7wxhw+
jrIWoh4snWbAdr+WZCTG7lDLDf/Fsdkn6h9uEEscrAUeKInCmBxrGBXodPyE6eWL
6G6xkK+2JAJcEZ81r3oWZFoJbnDcYzA3kVKyA05A
-----END CERTIFICATE-----
```

FASE 14 — Verificar que el certificado de Alice “cuelga” de la CA

- openssl verify -CAfile starfleet_ca.crt alice_cert.crt

```
ub@forense-ai:~/mision_canal_seguro$ openssl verify -CAfile starfleet_ca.crt alice_cert.crt
alice_cert.crt: OK
```

La creación de este resultado indica que se ha realizado correctamente: alice_cert.crt: OK

PARTE 5 — “HÍBRIDO” (Cómo funciona Internet de verdad)

Diferencia cifrado vs firma

- El objetivo del cifrado es garantizar la confidencialidad: el remitente cifra el mensaje utilizando la clave pública del destinatario, y solo la clave privada correspondiente puede descifrarlo, lo que impide que terceros lean el contenido.

- El objetivo de las firmas digitales es garantizar la integridad y la autenticidad de la identidad: el remitente firma el mensaje utilizando su clave privada, y cualquiera puede verificar la firma utilizando su clave pública, lo que confirma que el mensaje no ha sido manipulado y que realmente procede de ese remitente.

Por qué se necesitan certificados

- Porque una clave pública por sí sola no puede demostrar su propiedad, los atacantes pueden falsificar claves públicas para suplantar a otros. Los certificados digitales basados en el estándar X.509 vinculan la información de identidad a la clave pública y están firmados por una autoridad de certificación (CA) de confianza, lo que establece una cadena de confianza. Al verificar la firma del certificado, los usuarios pueden confirmar que la clave pública pertenece realmente a la entidad correspondiente.

Por qué TLS (Transport Layer Security) es híbrido

Porque el cifrado asimétrico (como RSA o Diffie-Hellman), aunque es seguro, es computacionalmente lento y no es adecuado para cifrar grandes volúmenes de datos, por el contrario, el cifrado simétrico es rápido y eficiente, pero requiere que ambas partes compartan la misma clave. En consecuencia, TLS emplea primero el cifrado asimétrico para establecer de forma segura una clave de sesión y, a continuación, utiliza el cifrado simétrico para proteger los datos de comunicación posteriores, equilibrando así la seguridad y el rendimiento.