

Intercambio de claves de Diffie y Hellman

Adrián Racero Serrano
Juan Manuel Cardenosa Borrego

Índice

1	Introducción	2
2	Algoritmo	2
3	Difusión y confusión	4
4	Autenticación	4
5	Problemas de seguridad del intercambio de claves Diffie-Hellman	5
5.1	El ataque de Logjam	5
6	Variaciones del intercambio de claves Diffie-Hellman	6
6.1	Curva elíptica Diffie-Hellman	7
6.2	TLS	7
7	El intercambio de claves Diffie-Hellman y RSA	8
8	Computación cuántica en el intercambio de claves Diffie-Hellman	8
9	Conclusión	9
10	Bibliografía	10
11	Anexo	11
11.1	Código	11

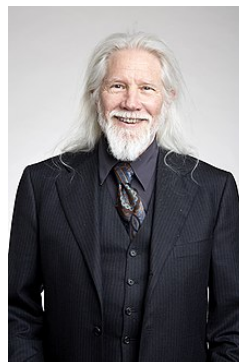
1 Introducción

En 1976, Whitfield Diffie y Martin Hellman crearon el algoritmo Diffie-Hellman, un protocolo de intercambio de claves que sigue siendo utilizado hoy en día. Su innovador trabajo en seguridad informática les valió el Premio A.M. Turing en 2015.

Este algoritmo posibilita que dos usuarios intercambien de manera confidencial una clave secreta, denominada K o de sesión, que luego utilizan para cifrar sus mensajes de manera simétrica.

A pesar de su aparente complejidad, el algoritmo Diffie-Hellman es bastante sencillo y se implementa con frecuencia en protocolos y aplicaciones de cifrado, como SSL, SSH o VPN. Su función principal es permitir que dos entidades (A y B) generen simultáneamente una clave compartida K_{AB} sin necesidad de enviarla a través del canal de comunicación.

No obstante, hay que tener en cuenta que el uso único de Diffie-Hellman puede llevar consigo varias vulnerabilidades, como puede ser la falta de autenticación o la aparición de los ordenadores cuánticos que podrían poner en peligro este algoritmo.



Whitfield Diffie



Martin Hellman

2 Algoritmo

1) A y B necesitan establecer y compartir valores comunes, como un valor q primo y una raíz primitiva α de q :

Para todo primo q existe un elemento $\alpha \in (\mathbb{Z}/q\mathbb{Z})^\times$ con $\text{ord}(\alpha) = q - 1 = \phi(q) = \#(\mathbb{Z}/q\mathbb{Z})^\times$. $(\mathbb{Z}/q\mathbb{Z})^\times$ es el grupo multiplicativo de las clases de equivalencia módulo q , es decir, que no tienen ningún factor primo en común con q :

$$(\mathbb{Z}/q\mathbb{Z})^\times = \{1, \alpha, \alpha^2, \dots, \alpha^{q-2}\}.$$

Este α se llama una **raíz primitiva** módulo q . Ejemplo: para $q = 5$, $\alpha = 3$ es una raíz primitiva módulo 5, ya que: $3^0 \bmod 5 = 1$, $3^1 \bmod 5 = 3$, $3^2 \bmod 5 = 4$ y $3^3 \bmod 5 = 2$.

2) Tanto A como B generan sus claves privadas ($X_{A/B}$) y públicas ($Y_{A/B}$) teniendo en cuenta que: $Y_A \equiv \alpha^{X_A} \pmod{q}$, donde $0 \leq X_A \leq (q-1)$, X_A es el logaritmo discreto de Y_A , y se representa $dlog_{\alpha,q}(Y_A)$.

3) Tanto A como B comparten sus respectivas claves públicas (Y_A/Y_B).

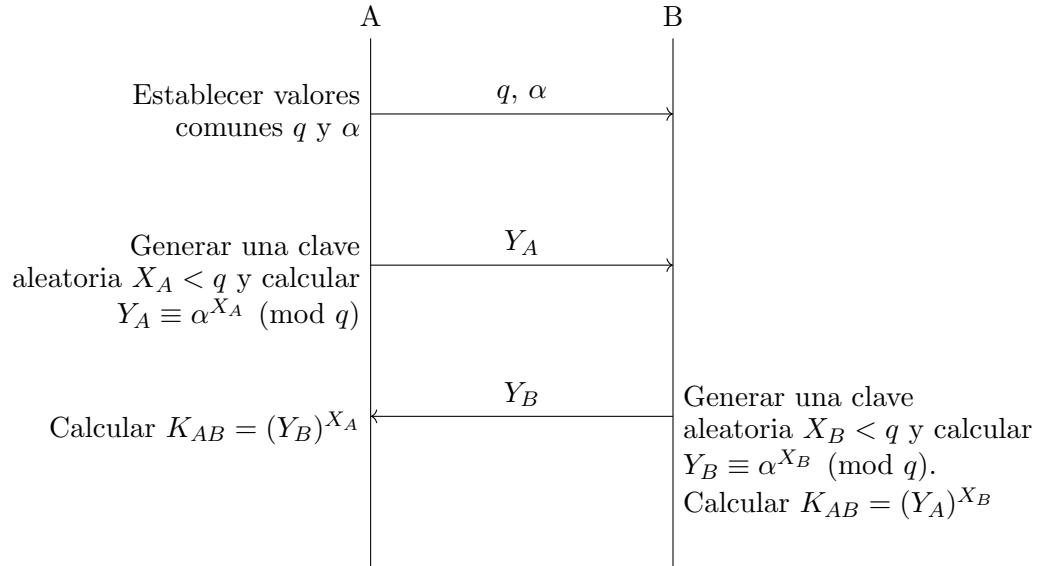
4) Tanto A como B generan la clave de sesión:

$$\begin{aligned} K_{AB} &= (Y_B)^{X_A} \pmod{q} \rightarrow K_{AB} = (\alpha^{X_B})^{X_A} \pmod{q} \\ K_{AB} &= (Y_A)^{X_B} \pmod{q} \rightarrow K_{AB} = (\alpha^{X_A})^{X_B} \pmod{q} \end{aligned}$$

A y B generan la misma clave secreta sin necesidad de compartir sus claves privadas:

$$K_{AB} = \alpha^{X_B X_A} \pmod{q} = \alpha^{X_A X_B} \pmod{q}$$

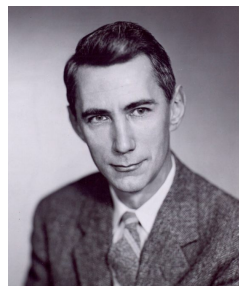
El siguiente diagrama de secuencia representa la interacción entre dos entidades A y B durante el intercambio de claves:



La efectividad del algoritmo depende de la dificultad de computar logaritmos discretos (mucho más complicado que calcular la exponenciación usada para transformar los números) y que es computacionalmente imposible deducir la clave privada del usuario U a partir de su clave pública. Por consecuencia, la clave secreta es el resultado de aplicar una función unidireccional al número secreto.

3 Difusión y confusión

Claude Shannon introdujo los conceptos de "confusión" y "difusión" como dos principios fundamentales en la teoría de la información y la criptografía. Aunque estos conceptos son fundamentales en el diseño de cifradores simétricos, como los utilizados en la criptografía de clave secreta, no se aplican directamente al intercambio de claves de Diffie-Hellman, que está más relacionado con la criptografía de clave pública y problemas matemáticos específicos.



Claude Shannon

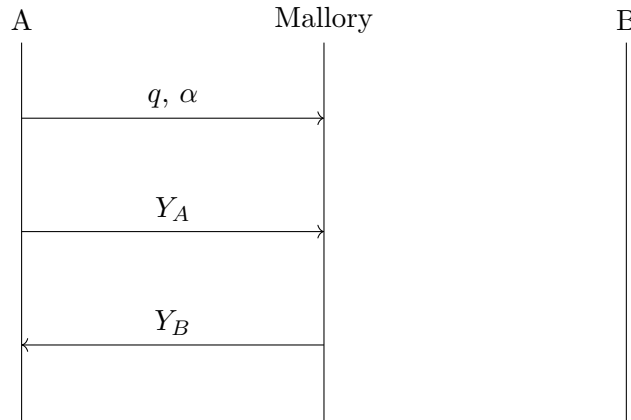
Estos conceptos se aplican una vez obtenida la clave secreta en el algoritmo, para mayor seguridad durante el intercambio de mensajes.

- **Confusión:** Implica hacer que la relación entre el texto cifrado y la clave sea compleja y difícil de entender. Esto se logra mediante operaciones no lineales y sustituciones, complicando la relación entre el texto cifrado y la clave.
- **Difusión:** Busca dispersar la influencia de un bit de entrada en muchos bits de salida. Pequeños cambios en la entrada deben propagarse a través del cifrado, distribuyendo la información en el texto cifrado. Es decir, cada carácter del texto cifrado ha de depender de diferentes partes de la clave.

4 Autenticación

Hoy en día, el método de intercambio de claves Diffie-Hellman raramente se emplea por sí solo, ya que carece de autenticación, lo que deja a los usuarios expuestos a posibles ataques de intermediarios. Al no contar con algún mecanismo de autenticación, los usuarios podrían establecer conexión con atacantes sin darse cuenta, creyendo que están comunicándose con una entidad de confianza.

En el siguiente diagrama observamos que una tercera persona (Mallory) está pichando el canal de comunicaciones haciéndose pasar por B, ya que no existe un mecanismo de autenticación mediante el cual A pueda comprobar con la persona que quiere comunicarse.



Por esta razón, el intercambio de claves Diffie-Hellman generalmente se implementa en conjunto con métodos de autenticación adicionales. Esto suele implicar el uso de certificados digitales y un algoritmo de clave pública, como RSA, para verificar la identidad de cada una de las partes involucradas.

5 Problemas de seguridad del intercambio de claves Diffie-Hellman

La eficacia del intercambio de claves Diffie-Hellman en términos de seguridad está definida por la forma en que se implementa y por la elección de los números involucrados. Por sí mismo, el método carece de la capacidad de autenticar a la otra parte en la comunicación. Sin embargo, en la práctica, se suelen emplear otros mecanismos para asegurar que la entidad en la otra extremidad de la conexión sea auténtica y no un impostor.

5.1 El ataque de Logjam

El intercambio de claves Diffie-Hellman se basó en la idea de que resolver el problema del logaritmo discreto sería una tarea complicada. Se tomó en consideración la eficacia del algoritmo de tamiz de campo numérico (algoritmo que se utiliza para identificar números primos en un rango específico).

En 1992, ya se sabía que, para un grupo dado G , tres de los cuatro pasos del algoritmo podrían calcularse de antemano. Al almacenar este progreso, el paso final podría calcularse en un tiempo corto. Este aspecto no causó mucha preocupación hasta que se descubrió que una parte considerable del tráfico de Internet utilizaba grupos de 1024 bits o menos. En 2015, un equipo académico realizó cálculos específicamente para el primo de 512 bits más comúnmente empleado en el intercambio de claves Diffie-Hellman en TLS.

Además, lograron debilitar el 80% de los servidores TLS que admitían DHE-EXPORT, por lo que estos servidores aceptarían un intercambio de claves Diffie-Hellman de grado de exportación de 512 bits para la conexión. En consecuencia, cada uno de estos servidores se vuelve vulnerable a un ataque por parte de un adversario con recursos suficientes.

Los investigadores llevaron sus resultados un paso más allá, estimando que un estado-nación podría comprometer un número primo de 1024 bits. Al romper el número primo de 1024 bits más usado, el equipo académico calculó que un adversario podría monitorizar el 18% de los sitios web HTTPS más populares, que ascienden a un millón.

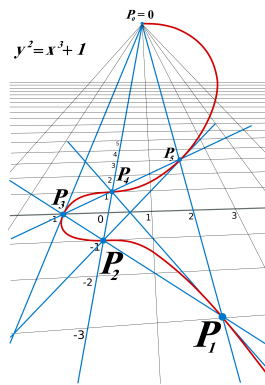
Además, indicaron que permitiría al adversario descifrar las conexiones del 66% de los servidores VPN y del 26% de los servidores SSH. En una nota posterior del informe, los académicos plantearon la posibilidad de que la NSA ya tenga estas capacidades.

A pesar de esta vulnerabilidad, el intercambio de claves Diffie-Hellman puede seguir siendo seguro si se implementa correctamente. Utilizando una clave de 2048 bits, el ataque Logjam no es efectivo. Además, los navegadores están protegidos contra este tipo de ataque.

6 Variaciones del intercambio de claves Diffie-Hellman

El intercambio de claves Diffie-Hellman se puede implementar de distintas maneras y también proporciona la base para varios otros algoritmos. Algunas de estas implementaciones proporcionan autorización, confidencialidad, técnicas criptográficas varias, como el *Perfect forward secrecy*, ...

6.1 Curva elíptica Diffie-Hellman



Curva elíptica

Curva elíptica Diffie-Hellman aprovecha la estructura algebraica de las curvas elípticas, lo que permite que su implementación alcance niveles similares de seguridad con tamaños de clave más pequeños. Las claves de curva elíptica de 224 bits proporcionan el mismo nivel de seguridad que las claves RSA de 2048 bits. Esto mejora la eficiencia de conmutación y reduce los requisitos de almacenamiento.

Diffie-Hellman de curva elíptica funciona de manera similar al intercambio de claves estándar Diffie-Hellman, excepto por la longitud de clave más pequeña y su naturaleza basada en curva elíptica.

6.2 TLS

TLS, un protocolo utilizado para proteger gran parte de Internet, puede utilizar los intercambios Diffie-Hellman de tres formas diferentes: anónima, estática y efímera. En la práctica, sólo debería implementarse la ley Diffie-Hellman efímera, ya que otras opciones tienen problemas de seguridad.

- **Diffie-Hellman anónimo:** no utiliza ninguna autenticación y la deja vulnerable a los ataques *man-in-the-middle*. No debe usarse ni implementarse.
- **Diffie-Hellman estático:** utiliza certificados para autenticar el servidor. No autentica al cliente de forma predeterminada, ni proporciona *Perfect forward secrecy*.
- **Diffie-Hellman efímero:** se considera la implementación más segura porque proporciona un secreto directo perfecto. Se suele combinar con un algoritmo como DSA o RSA para autenticar a una o ambas partes en la conexión. Esta versión utiliza diferentes pares de claves cada vez que se ejecuta el protocolo. Esto le da a la conexión un *Perfect forward secrecy*, porque incluso si una clave se ve comprometida en el futuro, no se puede usar para descifrar todos los mensajes pasados.

7 El intercambio de claves Diffie-Hellman y RSA

El intercambio de claves Diffie-Hellman se complementa a menudo con algoritmos como RSA para proporcionar autenticación en conexiones seguras. RSA, aunque capaz de cifrar mensajes, se utiliza principalmente para autenticar a las partes mediante certificados digitales verificados por autoridades de certificación. Este proceso implica la firma de mensajes con claves privadas y la verificación con las claves públicas correspondientes en los certificados.

A pesar de la autenticación lograda por RSA, su uso exclusivo para cifrar todas las comunicaciones sería ineficiente. Por lo tanto, muchos protocolos de seguridad optan por combinar el intercambio de claves Diffie-Hellman para llegar a un secreto compartido eficientemente. Este secreto, a menudo utilizado como clave simétrica compartida, se emplea en algoritmos de cifrado simétrico como AES para la transmisión segura de datos entre las partes autenticadas.

Este enfoque es más eficiente que depender únicamente de RSA para todo el intercambio, ya que el cifrado de clave simétrica es más rápido. Además, RSA presenta desventajas adicionales, como la necesidad de relleno para seguridad y la falta de secreto directo perfecto en comparación con el intercambio efímero de Diffie-Hellman.

En lugar de RSA, el intercambio de claves Diffie-Hellman puede combinarse con algoritmos como el Estándar de firma digital (DSS) para proporcionar autenticación, intercambio de claves, confidencialidad e integridad de datos, eliminando la necesidad de RSA en determinadas situaciones. Este enfoque flexible se adapta a las necesidades específicas de seguridad en las conexiones.

8 Computación cuántica en el intercambio de claves Diffie-Hellman

La computación cuántica, una disciplina en constante evolución, plantea desafíos significativos en el ámbito de la criptografía. Los detalles exactos de su funcionamiento son complejos, pero se espera que los ordenadores cuánticos tengan la capacidad de resolver problemas que actualmente son demasiado difíciles para los ordenadores clásicos, lo que abre nuevas posibilidades y riesgos.

La amenaza principal se encuentra en los algoritmos cuánticos, como el algoritmo de Grover, que podrían acelerar los ataques contra sistemas de cifrado que utilizan claves simétricas. Aunque este riesgo puede mitigarse duplicando el tamaño de la clave, la preocupación se centra en el algoritmo de Shor, que afectaría directamente a la criptografía de clave pública.

Los algoritmos de clave pública comunes, como Diffie-Hellman, se basan en la complejidad de problemas como el logaritmo discreto y la factorización de enteros. Con el avance de las computadoras cuánticas, se espera que el algoritmo de Shor simplifique la resolución de estos problemas, poniendo en peligro la seguridad de los sistemas criptográficos que dependen de ellos.

En particular, el intercambio de claves Diffie-Hellman se sustenta en la dificultad práctica de resolver el problema del logaritmo discreto con la tecnología actual. Aunque no hay una línea de tiempo precisa sobre cuándo la computación cuántica representará una amenaza seria para este protocolo, se están desarrollando sustitutos y alternativas en la criptografía para anticipar y contrarrestar estos posibles riesgos. La seguridad de la criptografía de clave pública, fundamental para proteger las comunicaciones, es el foco principal de estos desafíos que los criptógrafos están abordando activamente.

9 Conclusión

En conclusión, el intercambio de claves de Diffie-Hellman ha demostrado ser una piedra angular en la construcción de comunicaciones seguras en entornos digitales. A lo largo de décadas, este protocolo ha ofrecido una solución eficaz para el desafío fundamental de establecer claves compartidas en un canal inseguro.

La gran característica del intercambio de claves de Diffie-Hellman radica en su capacidad para garantizar la confidencialidad de las claves compartidas, incluso en un escenario donde un atacante puede interceptar la comunicación. Su enfoque basado en la complejidad computacional de los logaritmos discretos ha resistido la prueba del tiempo, proporcionando una base sólida para numerosos protocolos de seguridad.

Sin embargo, es crucial reconocer las limitaciones inherentes del Diffie-Hellman, especialmente en lo que respecta a la autenticación. En el mundo real, su implementación a menudo se combina con certificados digitales y algoritmos de clave pública para abordar la falta de verificación de la identidad de las partes involucradas. Esta adaptación ha permitido que el Diffie-Hellman se integre con éxito en protocolos como TLS/SSL, proporcionando una seguridad más completa y autenticada.

A medida que la ciberseguridad evoluciona, el intercambio de claves de Diffie-Hellman enfrentará desafíos continuos, desde la amenaza potencial de la computación cuántica hasta la necesidad constante de mejorar los protocolos de autenticación. Sin embargo, su impacto duradero en la seguridad de la información subraya su importancia como un pilar confiable en la construcción de sistemas de comunicación seguros y confiables. En un panorama digital en constante cambio, el Diffie-Hellman sigue siendo un contribuyente fundamental a la garantía de la privacidad y la integridad de las comunicaciones en línea.

10 Bibliografía

<https://cadadr.org/teoria-de-numeros-basica/hoja-6.pdf>

<https://ciberseguridad.com/guias/recursos/intercambio-claves-diffie-hellman/>

<https://www.computerweekly.com/es/definicion/Intercambio-de-claves-Diffie-Hellman-intercambio-de-claves-exponencial>

11 Anexo

11.1 Código

La clase **DiffieHellman** replica el funcionamiento del algoritmo:

```
1 from funciones_auxiliares import *
2
3 class DiffieHellman :
4
5     def __init__(self, q, alfa):
6
7         if not(es_primo(q)):
8             print("ERROR: q no es primo")
9             exit(-1)
10
11         if not(es_raiz_primitiva(alfa, q)):
12             print("ERROR: alfa no es raiz primitiva de
13                 q")
14             exit(-1)
15
16         self.__q__ = q
17         self.__alfa__ = alfa
18
19         self.__generarClaves__()
20
21     def __generarClaves__(self):
22         #  $X < q$ 
23         self.__clavePrivada__ = numero_aleatorio(self.
24             __q__)
25         #  $Y = \text{alfa}^X \text{ mod } q$ 
26         self.__clavePublica__ = ( self.__alfa__ **
27             self.__clavePrivada__ ) % self.__q__
28
29     def compartir_clave_publica(self):
30         return self.__clavePublica__
31
32     def generar_clave_secreta(self, clavePublicaB):
33         #  $K = Yb^Xa \text{ mod } q$ 
34         self.__claveSecreta__ = ( clavePublicaB **
35             self.__clavePrivada__ ) % self.__q__
36
37     # Este metodo solo se utiliza para comprobar
38     # que ambas claves secretas son iguales.
39     # Las claves secretas no se deben compartir.
40     def compartir_clave_secreta(self):
```

```

37         return self.__claveSecreta__
38
39     # Cifrar un mensaje usando XOR y la clave secreta
40     def cifrar_xor(self, mensaje):
41         cifrado = [char ^ self.__claveSecreta__ for
42                     char in mensaje.encode()]
43         return bytes(cifrado)
44
45     # Descifrar un mensaje usando XOR y la clave
46     secreta
47     def descifrar_xor(self, cifrado):
48         mensaje = ''.join([chr(char ^ self.
49                             __claveSecreta__) for char in cifrado])
50         return mensaje

```

Prueba para comprobar el funcionamiento del algoritmo:

```

1  from algoritmo import *
2
3  print("—— INICIALIZAR VALORES ——")
4  print("q = 153")
5  print("alfa = 3")
6  print()
7  DH1 = DiffieHellman(353,3)
8  DH2 = DiffieHellman(353,3)
9
10 print("—— GENERAR CLAVE SECRETA ——")
11 DH1.generar_clave_secreta(DH2.compartir_clave_publica
12                             ())
13 print("Clave de DH1: " + str(DH1.
14                               compartir_clave_secreta()))
15 DH2.generar_clave_secreta(DH1.compartir_clave_publica
16                             ())
17 print("Clave de DH2: " + str(DH2.
18                               compartir_clave_secreta()))
19 print()
20
21 print("—— CIFRADO/DESCIFRADO ——")
22 mensaje = "Diffie Hellman"
23 cifrado = DH1.cifrar_xor(mensaje)
24 print("DH1 cifrando...")
25 print(mensaje + " -> " + str(cifrado))
26 mensaje_descifrado = DH2.descifrar_xor(cifrado)
27 print("DH2 descifrando...")
28 print(str(cifrado) + " -> " + mensaje_descifrado)

```

Salida de la prueba:

———— INICIALIZAR VALORES ————

q = 153

alfa = 3

———— GENERAR CLAVE SECRETA ————

Clave de DH1: 231

Clave de DH2: 231

———— CIFRADO/DESCIFRADO ————

DH1 cifrando...

Diffie Hellman -> b'\xa3\xe\x81\x81\xe\x82\xc7\xaf
\x82\x8b\x8b\x8a\x86\x89'

DH2 descifrando...

b'\xa3\xe\x81\x81\xe\x82\xc7\xaf
\x82\x8b\x8b\x8a\x86\x89' -> Diffie Hellman