Universidad de Buenos Aires

Facultades de Ciencias Económicas,

Cs. Exactas y Naturales e Ingeniería

Carrera de Maestría en Seguridad Informática

Trabajo Final de la Maestría

Zitram

Protocolo de intercambio de claves basado en la seudo inversa de Moore-Penrose

Autor:

Eladio José Cousiño Godoy

Tutor del Trabajo Final:

Hugo Scolnik

2019

Cohorte 2017

Declaración jurada de origen de los contenidos

“Por medio de la presente, el autor manifiesta conocer y aceptar el Reglamento de Trabajos Finales vigente y se hace responsable que la totalidad de los contenidos del presente documento son originales y de su creación exclusiva, o bien pertenecen a terceros u otras fuentes, que han sido adecuadamente referenciados y cuya inclusión no infringe la legislación Nacional e Internacional de Propiedad Intelectual”.

_____

Eladio José Cousiño Godoy

DNI: 95736769

Resumen

El presente trabajo recopila información referente a protocolos de intercambio de claves, enfocándose en el método de Diffie-Hellman y las vulnerabilidades de este, matrices y la seudo inversa de Moore-Penrose. Se inicia con una introducción a conceptos necesarios relacionados a criptografía y protocolos de intercambio de claves, luego se realiza una revisión de matrices y la seudo inversa de Moore-Penrose, posteriormente se propone un algoritmo denominado inicial, se realiza una evaluación de este, y a continuación se propone un método mejorado para mitigar las debilidades del propuesto inicialmente, el nombre asignado es Zitram. Finalmente se presentan conclusiones sobre el trabajo realizado, y se proponen mejoras para trabajos futuros.

Palabras clave: Criptografía, intercambio de claves, seudo inversas, forward secrecy.

Índice General

Agradecimientos

Al Dr. Hugo Scolnik por toda su ayuda y paciencia, sin las cuales no hubiese sido posible el presente trabajo.

A mi familia y amigos por todo el apoyo incondicional brindado durante esta maestría.

1. Cuerpo Introductorio

1.1. Área temática

El campo de trabajo está principalmente dirigido a la seguridad informática. Se abordarán temas más específicos como los protocolos de intercambio de claves y en menor grado matrices. El foco estará colocado en el proceso del acuerdo e intercambio de claves entre dos partes a través de un canal inseguro. Particularmente, se detallará un protocolo como alternativa a Diffie-Hellman para el acuerdo de claves.

1.2. Introducción

La creciente demanda de comunicación entre personas y dispositivos que se presenta en la actualidad exige al mismo tiempo mayor seguridad en las mismas. Es decir, que un tercero no autorizado a conocer la información que se está transmitiendo entre dos partes, no pueda acceder a ella.

La criptografía es la ciencia que se encarga de proteger la confidencialidad de la información transmitida ya sea por medios digitales o físicos.

Se define a la Criptografía como el estudio de técnicas matemáticas relacionadas a aspectos de la seguridad de la información como son la confidencialidad, integridad, autenticación y no repudio. Durante las edades ha sido un arte, muchos idearon diversas técnicas para conseguir los requerimientos de seguridad.

En el ámbito de la criptografía, se llama cifrado al proceso de convertir un texto plano, es decir, legible por cualquier entidad, en uno conformado por un conjunto símbolos sin sentido aparente, este se denomina texto cifrado y puede ser interpretado únicamente por las partes que posean la clave para descifrarlo.

Los algoritmos utilizados para la conversión de un texto plano a uno cifrado llevan el nombre de primitivas criptográficas.

Se denomina criptosistema al conjunto de primitivas criptográficas utilizadas para proveer confidencialidad a la información, y se clasifican típicamente en dos grupos: Criptografía Simétrica y Criptografía Asimétrica.

En la criptografía simétrica, se utiliza la misma clave para cifrar el mensaje como para descifrarlo. Uno de los mayores problemas de la criptografía simétrica es encontrar un método eficiente para acordar e intercambiar claves de forma segura. Este problema se conoce como el problema de la distribución de claves.

En el caso de la criptografía asimétrica, la clave para el cifrado no es la misma que para el descifrado. Razón por la cual existe una clave pública y una privada, si se cifra un mensaje con la clave pública, únicamente puede ser descifrada por el usuario que posea la clave privada adecuada. Por otra parte, si se cifra el mensaje con la clave privada, todos los que posean la clave pública pueden descifrar el mensaje [1].

En la actualidad uno de los algoritmos de intercambio de claves más extendido es el denominado Diffie-Hellman, el cual, basa su seguridad en la dificultad computacional de resolver el problema del logaritmo discreto. El problema principal de este sistema de intercambio de claves es el ataque denominado *man-in-the-middle*, esta metodología se describe con mayor detalle más adelante en este trabajo.

El fin de este trabajo es proveer un protocolo que pueda utilizarse en lugar de Diffie-Hellman para el intercambio seguro de claves entre dos partes utilizando como base la seudo inversa de Moore-Penrose.

1.3. Estado del arte

Diffie-Hellman fue desarrollado por Whitfield Diffie y Martin Hellman, es considerado el primer algoritmo criptográfico asimétrico de clave pública. Este protocolo permite a dos partes establecer una clave compartida a través de un canal inseguro, por ejemplo, internet. Es utilizado en protocolos como IPSec y SSH para generar la clave compartida.

Existen otros algoritmos basados en Diffie-Hellman, por ejemplo, Elgamal, desarrollado por Taher Elgamal en 1984, y MQV (Menezes-Qu-Vanstone), creado en 1995 y modificado en 1998, el cual incorpora autenticación al protocolo Diffie-Hellman.

Sin embargo, el más ampliamente utilizado sigue siendo Diffie-Hellman.

1.4. Hipótesis

Zitram es un protocolo de intercambio de claves criptográficamente seguro y, como tal, es una alternativa válida al protocolo Diffie-Hellman.

1.5. Objetivo

La meta del trabajo es proveer un protocolo de acuerdo de claves que sea capaz de reemplazar al protocolo Diffie-Hellman en la generación de claves seguras a través de un canal inseguro utilizando como base la seudo inversa de Moore-Penrose.

1.6. Alcance

En el presente trabajo se estudiarán la seudo inversa de Moore-Penrose, el protocolo Diffie-Hellman, los ataques a dicho protocolo poniendo foco en el denominado man-in-the-middle, y se generarán dos protocolos de intercambio de claves, uno basado exclusivamente en la seudo inversa y el segundo incorporando un protocolo adicional al primero para incrementar la seguridad de este y eliminar posibles vectores de ataque.

1.7. Metodología y Plan de actividades

Este trabajo se divide en tres partes: la primera parte está dedicada a los antecedentes que sientan las bases del trabajo. En la segunda se desarrollarán los protocolos de intercambio de claves y se realizarán pruebas para validar su robustez y eficiencia. Finalmente, en la tercera parte se elaborará un resumen del trabajo y se presentarán las conclusiones en base a los resultados de las pruebas. A continuación, se detalla el plan de actividades y el cronograma propuesto:

Primera parte: preliminares

- Actividad 1: Estudio de protocolos de intercambio de claves, enfocándose en el protocolo Diffie-Hellman.
- Actividad 2: Estudio de vulnerabilidades que afectan al protocolo Diffie-Hellman.
- Actividad 3: Estudio de la seudo inversa de Moore-Penrose.
- Actividad 4: Revisión histórica de criptografía basada en matrices.

Segunda parte: desarrollo de protocolos

- Actividad 1: Desarrollo del protocolo de intercambio de claves utilizando seudo inversas de Moore-Penrose.
- Actividad 2: Análisis de posibles vulnerabilidades del primer protocolo desarrollado.
- Actividad 3: Desarrollo del protocolo de intercambio de claves incorporando el primer protocolo a otro para incrementar la seguridad.
- Análisis de posibles vulnerabilidades del segundo protocolo desarrollado.

Tercera parte: conclusiones.

- Actividad 1: Resumir los resultados obtenidos de los análisis de los protocolos.
- Actividad 2: Elaborar conclusiones sobre los protocolos propuestos.

2. Preliminares

2.1. Fundamentos matemáticos

Los fundamentos matemáticos descritos en esta sección fueron extraídos de [2].

2.1.1. Teorema de la división, restos y la equivalencia modular

Dado un entero n, el conjunto Z de todos los enteros puede particionarse en dos subconjuntos: los que son múltiplos de n y los que no lo son. Gran parte de la teoría de números se basa en el refinamiento de esta partición obtenida

clasificando a los enteros que no son múltiplos de n de acuerdo con el resto que se obtiene al dividirlos por n.

Teorema 1: Para cualquier entero a y cualquier entero positivo n existen enteros únicos q y r tales que $0 \leq r < n$ y $a = qn + r$.

Dado un número real x se define a la función parte entera $[x]$ como al mayor entero que no supera a x.

El valor $q = [a/n]$ es el cociente de la división.

El valor $r = a \, mod \, n$ es el resto (o residuo) de la división. Por lo tanto, n/a si y sólo si $a \, mod \, n = 0$.

Por lo tanto,

$$a = [a/n]n + (a \, mod \, n) \tag{1}$$

O

$$a \, mod \, n = a - [a/n]n \tag{2}$$

Si $a \, mod \, n = b \, mod \, n$ se escribe $a \equiv b \, (mod \, n)$. Se dice en este caso que a es equivalente a b módulo n.

Como $a \equiv b \, (mod \, n)$ es una noción de equivalencia, el conjunto Z se particiona en n clases de equivalencia de acuerdo con sus restos al dividirlos por n. La clase de equivalencia módulo n que contiene a un entero a es: $[a]_n = \{a + kn : k \in Z\}$.

Escribir $a \in [b]_n$ equivale a $a \equiv b \, (mod \, n)$. El conjunto de todas esas clases de equivalencia es

$$Z_n = \{[a]_n : 0 \leq a \leq n - 1\} \tag{3}$$

En general se usa la definición

$$Z_n = \{0,1, \dots, n - 1\} \tag{4}$$

Naturalmente hay que tener en cuenta a las clases de equivalencia subyacentes. Por ejemplo, una referencia a -1 en Z_n debe entenderse como una referencia a $[n - 1]_n$ dado que $-1 \equiv n - 1 \, mod \, n$.

2.1.2. Divisores comunes y máximo común divisor (MCD)

Si d es un divisor de a y también un divisor de b, entonces se dice que d es un divisor común de a y b. Una propiedad importante de los divisores comunes es que:

$$d/a \text{ y } d/b \text{ implica } d/(a + b) \text{ y } d/(a - b) \tag{5}$$

Generalizando, se tiene que

$$d/a \text{ y } d/b \text{ implica } d/(ax + by) \tag{6}$$

También,

$$\text{Si } a/b \text{ entonces } |a| \leq |b| \text{ o } b = 0 \Longrightarrow a/b \text{ y } b/a \Longrightarrow a = \pm b \qquad (7)$$

El máximo común divisor de dos enteros a y b que no son simultáneamente nulos es el mayor de sus divisores comunes, y se denota por $MCD(a, b)$.

Teorema 6 (de la recursión): para cualquier entero no negativo a y cualquier entero positivo b

$$MCD\ (a, b) = MCD(b, a\ mod\ b)$$

2.1.3. Grupos finitos

Un grupo $(S,\oplus)$ es un conjunto S con una operación binaria $\oplus$ para la cual se verifican las siguientes propiedades:

1. Cerrado: Para todo $a, b\ \mathcal{E}\ S \Longrightarrow a \oplus b\ \mathcal{E}\ S$
2. Identidad: Existe un elemento neutro $e\ \mathcal{E}\ S$ tal que $a \oplus e = e\ \oplus a = a = a\ \forall\ a\ \in S$
3. Asociatividad: $\forall\ a, b, c\ \in S, (a \oplus b) \oplus c = a \oplus (b \oplus c)$
4. Inversa: $\forall\ a \in S\ \exists$ un único elemento $b \in S$ tal que $a \oplus b = b \oplus a = e$

Si un grupo satisface la propiedad conmutativa $a \oplus b = b \oplus a\ \forall\ a, b\ \in S$, se dice que es abeliano y si $|S| < \infty$ (donde $|S|$ denota la cardinalidad de S) se dice que el grupo es finito.

Se puede formar dos grupos abelianos finitos utilizando la suma y la multiplicación módulo n, donde n es un entero positivo, basados en las clases de equivalencia de los enteros módulo n. Para definir un grupo en Z_n se necesita operaciones binarias convenientes que se obtienen redefiniendo las operaciones ordinarias de suma y multiplicación. Si $a \equiv a'\ (mod\ n)$ y $b \equiv b'\ (mod\ n)$ entonces

$$a + b \equiv a' + b'\ (mod\ n)$$

$$ab \equiv a'b'\ (mod\ n)$$

De este modo se define la suma y la multiplicación módulo n, denotadas por $+_n$ y $._n$ mediante

$$[a]_n +_n [b]_n = [a + b]_n$$

$$[a]_n \cdot_n [b]_n = [ab]_n$$

Y análogamente la substracción

$$[a]_n -_n [b]_n = [a - b]_n$$

Usando esta definición de la suma módulo n se define el grupo aditivo módulo n como $(Z_n, +_n)$ recordando que $|Z_n| = n$.

Usando la definición de multiplicación módulo n se define el grupo multiplicativo módulo n como $(Z_n^*, \cdot_n)$ donde Z_n^* denota a los elementos de Z_n que son primos relativos con n, o sea

$$Z_n^* = \{[a]_n \in Z_n : MCD(a,b) = 1\}$$

Se puede ver entonces que $MCD(a,n) = 1 \Longrightarrow MCD(a+kn,n) = 1$ para todos los enteros k. Como $[a]_n = \{a + kn\,; k \in Z\}$ resulta que Z_n^* está bien definido. Por ejemplo, $Z_{15}^* = \{1,2,4,7,8,11,13,14\}$. La multiplicación módulo 15 es la operación del grupo; así $8.11 \equiv 13\ (mod\ 15)$ y la identidad para este grupo es el 1.

Teorema 14: $(Z_n^*, \cdot_n)$ es un grupo abeliano finito.

Según el teorema 6, para a, b, p enteros tales que $MCD(a,p) = MCD(b,p) = 1$ entonces $MCD(ab,p) = 1$. Esto implica que $(Z_n^*, \cdot_n)$ es cerrado.

La cardinalidad de Z_n^*, o sea $|Z_n^*|$ se denota por $\phi(n)$, conocida como la función de Euler, que satisface la ecuación

$$\phi(n) = n \prod_{p<n} \left(1 - \frac{1}{p}\right)$$

El orden de un grupo hace referencia al número de elementos de un grupo.

Si el orden de un grupo es un número finito, se dice que el grupo es finito.

Si el conjunto S cumple únicamente las propiedades de asociatividad e identidad, se dice que el conjunto es un monoide. [3]

2.1.4. Subgrupos

Si $(S, \oplus)$ es un grupo, $S' \subseteq S$, y $(S', \oplus)$ también es un grupo, entonces se dice que $(S', \oplus)$ es un subgrupo de $(S, \oplus)$. Por ejemplo, los enteros pares forman un subgrupo de los enteros con la operación de suma.

Teorema 14 (un subconjunto cerrado de un grupo finito es un subgrupo):

Si $(S, \oplus)$ es un grupo finito y S' es cualquier subconjunto de S tal que si $a, b \in S' \Longrightarrow a \oplus b \in S'$ entonces $(S', \oplus)$ es un subgrupo de $(S, \oplus)$.

2.1.5. Subgrupo generado por un elemento

El Teorema 14 da un modo de generar subgrupos de grupos finitos: elegir un elemento a y tomar todos los elementos que puedan obtenerse de él con la operación de grupo. Sea

$$a^{(k)} = a \oplus a \ldots \oplus a \text{ (k veces)}$$

Por ejemplo, si se toma $a = 2$ en Z_6 resulta $a^{(1)} = 2$, $a^2 = 4$, $a^3 = 6 \equiv 0$, $a^4 = 2$,…

En el grupo Z_n tenemos que $a^{(k)} = ka \; mod \; n$ y en el grupo Z_n^* es $a^{(k)} = \{a^k \; mod \; n\}$.

El subgrupo generado por a denotado por (a) o $(a, \oplus)$ se define por $(a) = \{a^{(k)}, \; k \geq 1\}$

Se dice que a genera (a) o que es un generador de (a). Como S es finito, (a) es subconjunto finito de S. La asociatividad de $\oplus$ implica que $a^{(i)} \oplus a^{(j)} = a^{(i+j)}$. (a) es cerrado, entonces por el Teorema 14, (a) es un subgrupo de S.

Corolario 19: si $(S, \oplus)$ es un grupo finito con identidad e, entonces $\forall a \in S, a^{(|S|)} = e$.

2.1.6. Raíz primitiva y generador del grupo

En cuanto a las potencias de $a \in Z_n^* \; mod \; n$. Comenzando con 0 se tiene que $a^0 \; mod \; n = 1$ y el i-ésimo valor es $a^i \; mod \; n = n - 1$. Por ejemplo, las potencias de $3 \; mod \; 7$ son:

i	0	1	2	3	4	5	6	7	8	9	10	11
3^i	1	3	2	6	4	5	1	3	2	6	4	5

En esta sección se usará la notación (a) para el subgrupo de Z_n^* generado por a y sea $ord_n(a)$ el orden de a en Z_n^*. Ahora se usará la definición de la función de Euler $\phi(n)$ para la cardinalidad de Z_n^* y se traduce el Corolario 19 en la notación de Z_n^* para obtener el Teorema de Euler y si además se lo especializa al caso Z_p^* resulta el Teorema de Fermat:

Teorema 30 (de Euler): para cualquier entero $n > 1$, $a^{\phi(n)} \equiv 1 \; (mod \; n) \; \forall \; a \in Z_n^*$.

Teorema 31 (de Fermat): si p es primo, $a^{p-1} \equiv 1 \; (mod \; p) \; \forall \; a \in Z_p^*$.

Esto se aplica a todo elemento en Z_p excepto el 0 puesto que $0 \notin Z_p^*$. Para todo $a \in Z_p$ se tiene que $a^p \equiv a \; (mod \; p)$.

Si $ord_n(g) = |Z_p^*|$ resulta que todo elemento de Z_p^* es una potencia de g, y en ese caso se dice que g es un generador o raíz primitiva de Z_p^*. Por ejemplo, 3 es un generador módulo 7. Si Z_p^* posee un generador, se dice que el grupo Z_p^* es cíclico. [4], [5]

2.1.7. Anillos

Si un conjunto R está equipado con dos operaciones binarias, la adición y la multiplicación, y cumple con los siguientes tres axiomas, se lo llama anillo.

Axioma 1: R es un grupo abeliano para la adición, esto significa que:

- Asociatividad: $(a+b)+c=a+(b+c)$ para todo $a,b,c \in R$.
- Conmutatividad: $a+b=b+a$ para todo $a,b \in R$.
- Identidad aditiva: existe un elemento $c \in R$ tal que $a+c=a \; \forall \; a \in R$
- Inversa aditiva: existe un elemento $-a \; \forall \; a \in R$ tal que $a+(-a)=0$

Axioma 2: R es un monoide para la multiplicación, esto significa que:

- Asociatividad: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ para todo $a,b,c \; \in R$
- Identidad multiplicativa: existe un elemento $c \in R$ tal que $a \cdot 1 = 1 \cdot a = a \; \forall \; a \in R$

Axioma 3: La multiplicación es distributiva con respecto a la adición, esto significa que:

- Distributiva por la izquierda: $a \cdot (b+c) = (a \cdot b) + (a \cdot c) \; \forall \; a,b,c \; \in R$
- Distributiva por la derecha: $(b+c) \cdot a = (b \cdot a) + (c \cdot a) \; \forall \; a,b,c \; \in R$

Si adicionalmente se cumple que:

$$a \cdot b = b \cdot a \;\; \forall \; a,b \; \in R$$

Es decir, la multiplicación es conmutativa, se dice que el anillo es conmutativo.

2.1.8. Notación de Landau

En el ámbito de ciencias de la computación, esta notación se utiliza para clasificar algoritmos de acuerdo con cómo su tiempo de ejecución o requerimientos de espacio aumentan con relación al aumento del tamaño de la entrada.

Un ejemplo de esta notación es utilizado más adelante en el trabajo, $O(\sqrt{n})$, esto significa que para una entrada de tamaño n, serán necesarias $\sqrt{n}$ operaciones. [6]–[9]

En la siguiente tabla, se observan algunos ejemplos de notación de Landau:

Tabla 1. Ejemplos de Notación de Landau. Basado en [10]

Notación	Nombre
$O(n)$	Orden lineal
$O(\log n)$	Orden logarítmico
$O(\sqrt{n})$	Orden sublineal
$O(n \log n)$	Orden lineal logarítmico
$O(n^2)$	Orden cuadrática
$O(n!)$	Orden factorial
$O(n^n)$	Orden potencial exponencial

2.1.9. Matrices e Inversas

Esta sección está basada fundamentalmente en [11].

Una matriz es un arreglo rectangular de números en filas y columnas. Por ejemplo:

$$A = \begin{bmatrix} -2 & 5 & 6 \\ 5 & 2 & 7 \end{bmatrix}$$

La matriz A tiene dos filas y tres columnas, esto indica que la matriz tiene dimensiones 2×3. Si la cantidad de filas es igual a la cantidad de columnas, se dice que la matriz es cuadrada.

Más adelante en el trabajo, se hace mención a las matrices circulantes, las cuales, son un tipo especial de matrices que están completamente definidas por una sola fila (la primera). Las demás filas son permutaciones cíclicas, es decir, cada fila después de la primera se genera rotando un elemento a la derecha con respecto a la anterior. Por ejemplo:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 2 & 3 & 1 \end{bmatrix}$$

Otro concepto importante que involucra a las matrices es el determinante. Es un valor escalar que puede ser computado a partir de los elementos de una matriz cuadrada y es denotado por $\det(A)$.

Para una matriz A de tamaño 2×2, el determinante se define como:

$$\det(A) = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

De manera similar, para una matriz A de tamaño 3×3, su determinante es:

$$\det(A) = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - ceg - bdi - afh$$

Para calcular el determinante de matrices de tamaño $n \times n$, se pueden utilizar la fórmula de Leibniz y la de Laplace [12]. Si una matriz tiene el determinante igual a cero, se dice que la matriz es singular.

Sea A una matriz cuadrada no singular, esta tiene una única inversa, denotada por A^{-1}, tal que:

$$AA^{-1} = A^{-1}A = I \tag{1}$$

Donde I es la matriz identidad. Las matrices y sus inversas tienen muchas otras propiedades las cuales no son relevantes para el desarrollo del trabajo, por lo tanto, no serán mencionadas.

Una matriz tiene una inversa si y sólo si es cuadrada y si es no singular, en otras palabras, si sus columnas (o filas) son linealmente independientes[1]. Sin

[1] Caso contrario, se dice que la matriz tiene rango deficiente.

embargo, la necesidad de algún tipo de inversa parcial de una matriz o incluso rectangular ha desencadenado el desarrollo de técnicas de generación de seudo inversas o inversas generalizadas.

Una seudo inversa es cualquier matriz X que satisface:

$$AXA = A \tag{2}$$

Si A fuese no singular, multiplicar ambos lados de la ecuación por A^{-1} resultaría en:

$$X = A^{-1}$$

Probablemente la aplicación más familiar de las matrices es la solución de sistemas de ecuaciones lineales. Sea

$$Ax = b \tag{3}$$

un sistema de ese tipo, donde b es un vector conocido y x es un vector desconocido. Si A es no singular, existe una única solución para x dada por:

$$x = A^{-1}b$$

En general, cuando A es singular o rectangular, pueden no existir soluciones o existir infinitas soluciones[2].

Se demostró en [13] que, si X es cualquier matriz que satisfaga (1), entonces (3) tiene solución si y sólo si

$$AXb = b$$

Donde la solución general es[3]

$$x = Xb + (I - XA)y \tag{4}$$

Donde y es arbitrario.

En 1955, Penrose demostró que, para cada matriz finita A (cuadrada o rectangular) de elementos reales o complejos, existe una matriz X que satisface las siguientes cuatro ecuaciones[4]

$$AXA = A \tag{5}$$

$$XAX = X \tag{6}$$

$$(AX)^* = AX \tag{7}$$

$$(XA)^* = XA \tag{8}$$

Donde A^* denota la conjugada transpuesta de A.

[2] Se da el caso de que tenga infinitas soluciones cuando la matriz tiene rango deficiente.
[3] Esta ecuación (4) es la que se utiliza más adelante para el desarrollo del protocolo.
[4] Estas ecuaciones son denominadas ecuaciones de Penrose.

Debido a que esta seudo inversa única ha sido estudiada previamente por E.H. Moore, se le conoce como la seudo inversa de Moore-Penrose, y es denotada por $A^{\dagger}$.

Si A es no singular, entonces $X = A^{-1}$ satisface las cuatro ecuaciones. Por consiguiente, la seudo inversa de Moore-Penrose de una matriz no singular es igual a la inversa ordinaria.

Si bien existen varios métodos para calcular la seudo inversa de Moore-Penrose, el desarrollo de los mismos no es relevante para el presente trabajo.

2.2. Seguridad de la Información

La seguridad de la información tiene como principales objetivos a la confidencialidad, integridad y disponibilidad. La confidencialidad se refiere a que la información no debe ser accedida por entidades no autorizadas. La integridad, a que la información no pueda ser alterada por entidades no autorizadas. Y la disponibilidad, a que la información pueda ser accedida cuando sea necesario.

En la actualidad, la conectividad crece a un ritmo muy acelerado, la información, prácticamente en su totalidad, es almacenada y transmitida en dispositivos que no pertenecen a los usuarios. Los avances en la computación distribuida han permitido que la disponibilidad de la información esté garantizada. Por lo tanto, existe una necesidad de satisfacer el resto de los objetivos de la seguridad de la información. Es aquí donde la criptografía juega un papel fundamental.

2.3. Criptografía

La criptografía es el estudio de métodos y algoritmos matemáticos relacionados con ciertos aspectos de la seguridad informática, como la confidencialidad, integridad, autenticación y el no repudio. Los conceptos de confidencialidad e integridad ya fueron descritos anteriormente, en cuanto a la autenticación, se refiere a la identificación de las partes que participan en la transmisión de la información. Y el no repudio consiste en prevenir que una entidad niegue acuerdos o acciones previas.

En el ámbito de la criptografía, se llama cifrado al proceso de convertir un texto plano en uno conformado por un conjunto de símbolos sin un sentido aparente, el cual puede ser interpretado únicamente por las partes que posean la clave. El proceso inverso se denomina descifrado.

El mecanismo utilizado para convertir un texto plano en uno cifrado, se denomina criptosistema. En otras palabras, se denomina criptosistema al conjunto de primitivas criptográficas utilizadas para proveer confidencialidad a la información, y se clasifican típicamente en: Criptografía Simétrica, Asimétrica y Funciones de Hashing. [1]

Un criptosistema puede ser definido en términos formales como una 5-tupla $(\mathrm{P}, \mathrm{C}, \mathrm{K}, \mathrm{E}, \mathrm{D})$ que satisface las siguientes condiciones (DIAPOSITIVA CRIPTO 1):

a) (P) es el conjunto finito de textos planos
b) (C) es el conjunto finito de textos cifrados
c) (K), el espacio de claves, es el conjunto finito de claves posibles
d) Para cada $k \in (K)$, existe una regla de cifrado $e_k \in (E)$ y una correspondiente regla de descifrado $d_k \in (D)$
e) Cada $e_k: (P) \to (C)$ y $d_k: (C) \to (P)$ son funciones tales que $d_k\big(e_k(x)\big) = x$ para cada texto plano $x \in (P)$

2.4. Criptoanálisis

El término criptoanálisis (del griego kryptós, que significa "escondido", y analýein, "aflojar" o "desatar") es el estudio de sistemas de información con el objetivo de analizar los aspectos ocultos de los sistemas. El criptoanálisis se utiliza para quebrar la seguridad criptográfica de los sistemas y obtener acceso al contenido de los mensajes cifrados, inclusive si las claves criptográficas son desconocidas.

En adición al análisis matemático de los algoritmos criptográficos, el criptoanálisis incluye el estudio de ataques de canal lateral, los cuales no apuntan a las debilidades de los algoritmos criptográficos, en su lugar, explotan vulnerabilidades en la implementación de estos.[1], [14], [15]

2.5. Criptografía Simétrica

En la criptografía simétrica, se utiliza la misma clave para cifrar el mensaje como para descifrarlo, es decir que $e_k = d_k$. Uno de los mayores problemas de la criptografía simétrica es encontrar un método eficiente para acordar e intercambiar claves de forma segura. Este problema se conoce como el problema de la distribución de claves.

Supóngase que dos partes, Alice y Bob quieren comunicarse de forma segura utilizando criptografía simétrica. Para lograr esta confidencialidad, tanto Alice como Bob deberían establecer una clave secreta a través de un canal seguro para evitar que un tercero pueda tener acceso a los mensajes enviados. Posteriormente, si Alice desea enviar un mensaje $m \in M$ a Bob, ella debe computar $c = E_e(m)$ y transmite c a Bob. Al recibir c, Bob computa $D_d(c) = m$ y recupera el mensaje original m.

En la ilustración 1 se puede observar una comunicación entre dos partes utilizando un esquema de cifrado simétrico, con un canal seguro para el intercambio de claves.

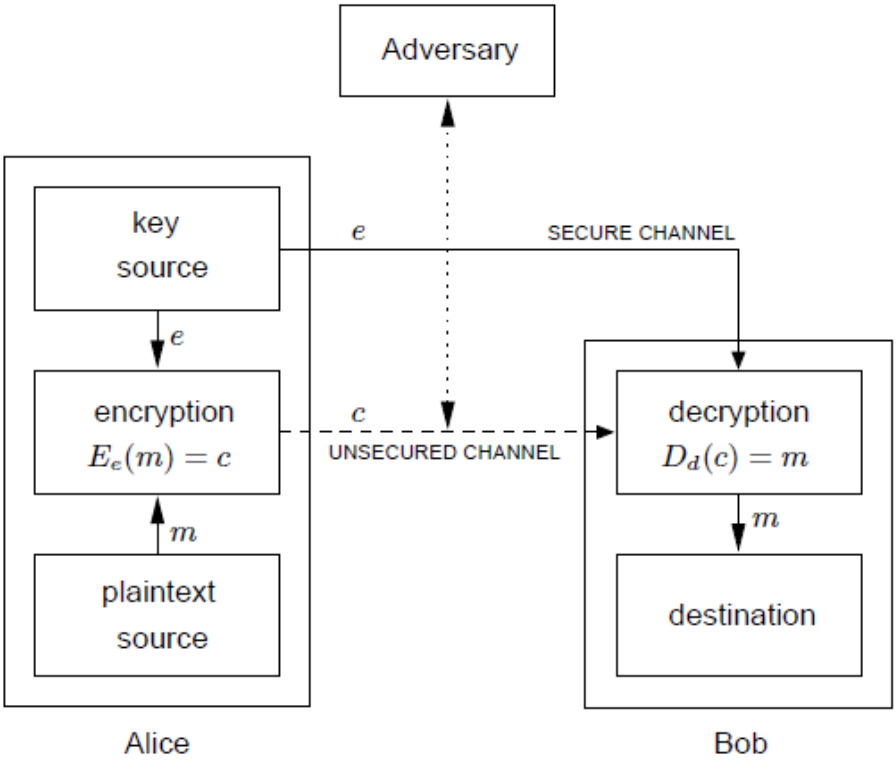

Ilustración 1. Criptografía Simétrica. Fuente: [1]

2.6. Criptografía Asimétrica

En el caso de la criptografía asimétrica, la clave para el cifrado no es la misma que para el descifrado, es decir que $e_k \neq d_k$. Razón por la cual existe una clave pública y una privada, si se cifra un mensaje con la clave pública, únicamente puede ser descifrado por el usuario que posea la clave privada adecuada. Por otra parte, si se cifra el mensaje con la clave privada, todos los que posean la clave pública pueden descifrar el mensaje.

Para que un esquema de cifrado pueda ser considerado de clave pública, por cada par de claves de cifrado/descifrado generado (e_k, d_k), una debe estar disponible públicamente, mientras que la otra debe mantenerse en secreto. Llevan de nombre de clave pública y privada respectivamente. Para que el esquema sea considerado seguro, debe ser computacionalmente inviable computar d_k a partir de e_k.

Supóngase nuevamente que dos partes, Alice y Bob quieren comunicarse de forma segura, esta vez utilizando criptografía asimétrica. En este caso, Bob genera un par de claves (e_k, d_k). Bob envía la clave de cifrado e_k a Alice a través de cualquier canal y almacena la clave de descifrado d_k en un lugar secreto y seguro. Entonces, cuando Alice desee enviar un mensaje m a Bob, utiliza la clave pública de este último para generar $c = E_c(m)$. Bob puede posteriormente descifrar el mensaje cifrado utilizando su clave privada d_k.

En la ilustración 2 se puede observar una comunicación entre dos partes utilizando criptografía asimétrica.

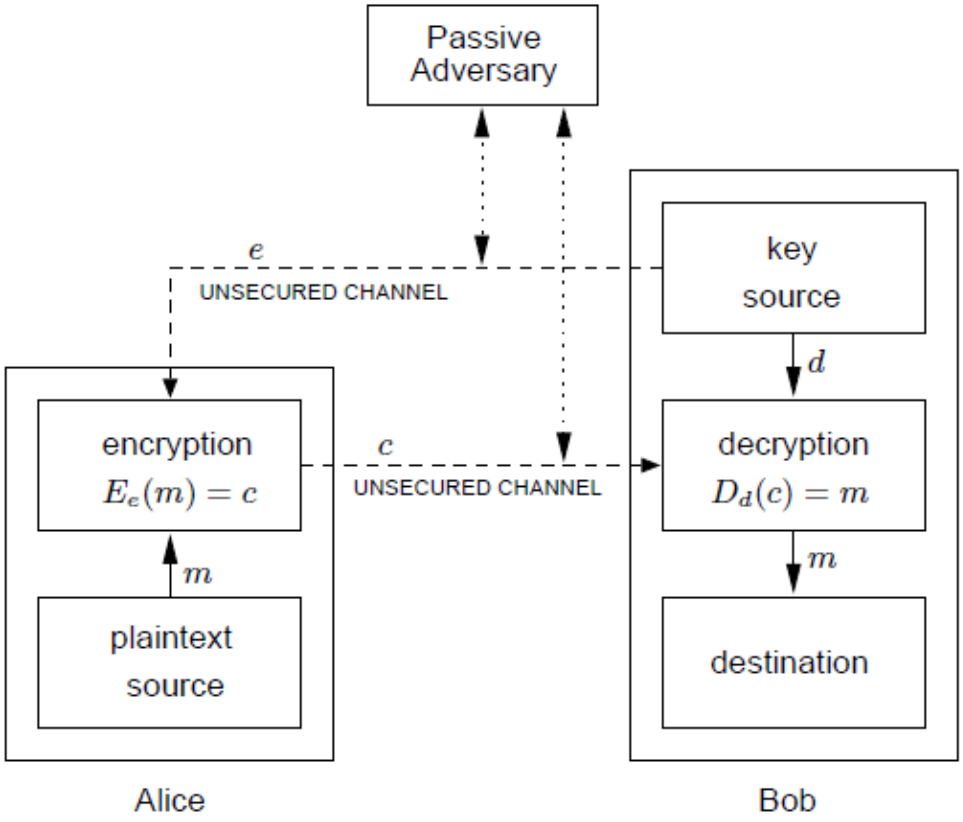

Ilustración 2. Criptografía Asimétrica. Fuente: [1]

2.7. Establecimiento de Claves

El establecimiento de claves es un proceso o protocolo mediante el cual un secreto compartido se vuelve disponible a dos o más partes, para su uso criptográfico. [1]

El establecimiento de claves puede ser dividido de forma general en dos tipos de protocolos:

- **Protocolo de Intercambio de Claves:** es una técnica de establecimiento de clave en la cual, un secreto compartido (clave), es derivado u obtenido por dos (o más) partes en función de la información proveída por cada una de ellas (idealmente), de tal modo que ninguna parte puede predeterminar el valor resultante.
- **Protocolo de transporte de claves:** es una técnica de establecimiento de clave en la cual una o más partes crean u obtienen un secreto compartido y lo transfiere de forma segura a las otras partes.

2.8. Man-in-the-middle

En términos de seguridad informática, un ataque del tipo *man-in-the-middle* (MITM) consiste en que un atacante intercepte, retransmita y altere, en secreto, las comunicaciones entre dos partes, las cuales no están conscientes del mismo y creen que se están comunicando directamente una con otra. Un ejemplo del ataque del tipo MITM es el espionaje activo, en el cual, el atacante genera conexiones independientes con las víctimas y retransmite mensajes entre ellos para hacerles creer que están hablando directamente en una conexión privada, cuando en realidad la totalidad de la conversación es controlada por el atacante.[1], [16]

Para ilustrar el funcionamiento de este tipo de ataques, se presenta el siguiente ejemplo:

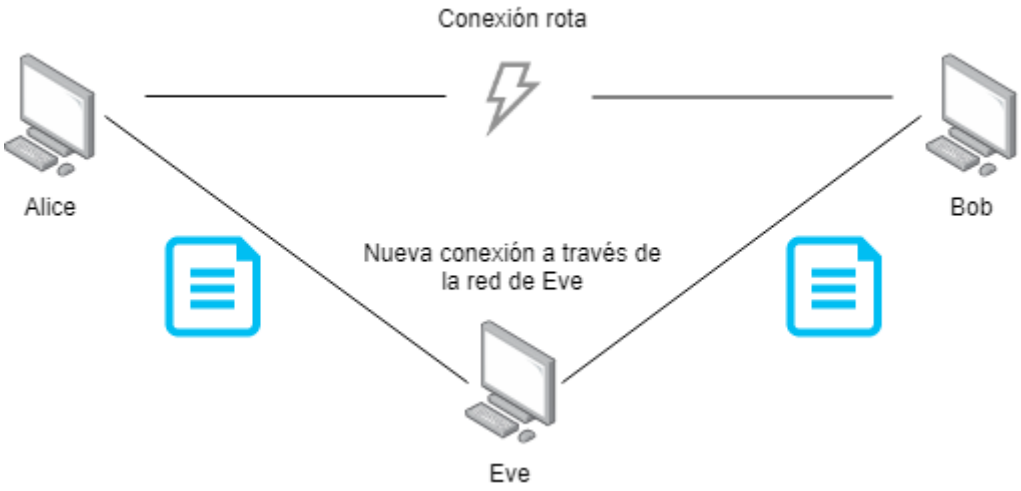

Ilustración 3. Man-in-the-middle. Fuente: Elaboración Propia.

2.9. Forward Secrecy

En criptografía, *Forward Secrecy*, también conocido como *Perfect Forward Secrecy*, es una característica de algunos protocolos[5] que asegura que las claves de sesión no serán comprometidas inclusive si la del servidor lo está. *Forward secrecy* protege las sesiones pasadas contra futuros compromisos mediante la generación de una única clave por cada sesión que inicia un usuario, el compromiso de una única clave de sesión afecta únicamente aquellos datos que fueron transferidos utilizándola, los demás datos no se ven afectados.

A continuación, se describe un ejemplo hipotético de un protocolo simple de mensajería instantánea que aplica *forward secrecy*:

1- Alice y Bob generan un par de claves públicas y privadas de larga duración, luego verifican el *fingerprint*[6] de las claves públicas en persona o a través de un

5 Por ejemplo, SSH, IPSec y el protocolo Signal, el cual implementa el algoritmo Double Ratchet para proveer Forward Secrecy.[1], [17], [18]

[6] En criptografía de clave pública, el fingerprint es una secuencia de bytes corta utilizado para identificar una clave pública de mayor longitud. Son creados al aplicar una función de hash a la clave pública.

canal seguro. La verificación establece la confianza de que Alice y Bob son realmente dueños de las claves.

2- Alice genera una clave de sesión de manera aleatoria, la cifra utilizando la clave pública de Bob y se la envía a Bob. Bob la descifra utilizando su clave privada. De esta forma queda establecida la clave de sesión.

3- Alice envía a Bob un mensaje, cifrándolo con un cifrador simétrico utilizando la clave de sesión negociada en el segundo paso.

4- Bob descifra el mensaje de Alice utilizando la clave negociada en el segundo paso.

5- El proceso se repite por cada mensaje nuevo, iniciando desde el segundo paso (e intercambiando los roles de emisor y receptor entre Alice y Bob según se requiera). El primer paso no se repite nunca.

2.10. El protocolo Diffie-Hellman (DH)

Publicado en 1976 y nombrado en honor a sus autores Whitfield Diffie y Martin Hellman, es el primer método de intercambio seguro de claves criptográficas a través de un canal inseguro, es decir, es el primer protocolo de clave pública. [19]

Tradicionalmente, para establecer una comunicación segura entre dos partes, se requería un intercambio de claves a través de un canal físico seguro, como por ejemplo listas de claves en papel transportados por un Courier confiable. El protocolo DH permite que dos partes, que no tienen conocimiento previo la una de la otra, establezcan un secreto compartido a través de un canal inseguro. Este secreto compartido puede ser utilizado posteriormente para cifrar la comunicación utilizando criptografía simétrica.

A continuación, se describen los pasos del protocolo:

Se supone que Alice y Bob quieren establecer una comunicación segura a través de una red abierta.

1- Alice y Bob acuerdan un número primo[7] p y un generador α del grupo $\mathbb{Z}_p^*$ $(2 \leq \alpha \leq p-2)$

2- Mensajes del protocolo

$A \rightarrow B : \alpha^x \bmod p$ (1)

$A \leftarrow B : \alpha^y \bmod p$ (2)

3- Acciones del protocolo. Realizar los siguientes pasos cada vez que se requiera una clave compartida.

[7] Un entero a > 1 tal que sus únicos divisores son 1 y a se denomina primo.
Un número es primo si es mayor a 1 y no puede ser escrito como producto de otros dos números naturales más pequeños [20], [21]. Los números naturales son todos los enteros positivos incluyendo el cero.

(a) Alice elige un secreto aleatorio x, $1 \leq x \leq p-2$, y envía a Bob el mensaje (1).
(b) Bob elige un secreto aleatorio y, $1 \leq y \leq p-2$, y envía a Alice el mensaje (2).
(c) Bob recibe α^x y computa la clave compartida $K = (\alpha^x)^y$ mod p.
(d) Bob recibe α^y y computa la clave compartida $K = (\alpha^y)^x$ mod p.

A modo de clarificar el funcionamiento del protocolo, se presenta el siguiente ejemplo:

Alice y Bob acuerdan públicamente utilizar el módulo $p = 23$ y el generador $\alpha = 5$, el cual es raíz primitiva del grupo $\mathbb{Z}_{23}^*$

Alice	**Bob**
Elige un secreto $x = 4$	
Calcula $A = \alpha^x$ mod p y envía A a Bob $A = 5^4 \ mod \ 23 = 4$	
	Elige un secreto $y = 3$
	Calcula $B = \alpha^y$ mod p y envía B a Alice $B = 5^3 \ mod \ 23 = 10$
Calcula $s = B^x \ mod \ p$ $s = 10^4 \ mod \ 23 = 18$	Calcula $s = A^y \ mod \ p$ $s = 4^3 \ mod \ 23 = 18$

Alice y Bob ahora tienen un secreto compartido, el número 18.

Han llegado al mismo valor numérico porque dado un grupo módulo p, se cumple que: $A^y \ mod \ p = \alpha^{xy} \ mod \ p = \alpha^{yx} \ mod \ p = B^x \ mod \ p$ [22]

2.11. Problema del logaritmo discreto

Sea G un grupo cíclico de orden n. Sea α un generador de G, y $\beta \in G$. El logaritmo discreto de β en base α, denotado por $\log_\alpha \beta$, es el único entero x, $0 \leq x \leq n-1$, tal que $\beta = \alpha^x$.

A modo de aclaración, se propone el siguiente ejemplo: sea $p = 97$. Entonces, $\mathbb{Z}_{97}^*$ es un grupo cíclico de orden $n = 96$. Un generador de $\mathbb{Z}_{97}^*$ es $\alpha = 5$. Debido a que $5^{32} \equiv 35 \ (mod 97)$, $\log_5 35 = 32$ en $\mathbb{Z}_{97}^*$.

El problema del logaritmo discreto (DLP), entonces, es el siguiente: dado un primo p, un generador α de $\mathbb{Z}_p^*$, y un elemento $\beta \in \mathbb{Z}_p^*$, encontrar el entero x, $0 \leq x \leq p-2$, tal que $\alpha^x \equiv \beta \ (mod \ p)$.

2.12. Problema de Diffie-Hellman

El problema de Diffie-Hellman (DHP) consiste en lo siguiente: dado un primo p, un generador α de $\mathbb{Z}_p^*$, y elementos $\alpha^a \ mod \ p$ y $\alpha^b \ mod \ p$, encontrar $\alpha^{ab} \ mod \ p$. Como se puede observar, el DHP está ligado al DLP, es decir, al encontrar un

algoritmo que resuelva el DLP eficientemente, consecuentemente, también se podría utilizar para resolver el DHP.

Este problema es actualmente considerado difícil de resolver porque si bien existen algunos algoritmos que pueden ser utilizados para hallar la solución del DLP, ninguno de ellos es suficientemente eficiente. Estos son los algoritmos utilizados en la actualidad para encontrar la solución al DLP:

- **Búsqueda exhaustiva:** consiste en calcular sucesivamente $a^0, a^1, a^2, \ldots$ hasta obtener B. Este método requiere $O(n)$ multiplicaciones, donde n es el orden de a, y por lo tanto este método es ineficiente si n es muy grande.

- ***Baby-step giant-step*:** este algoritmo es una variación de la búsqueda exhaustiva en el cual se utiliza espacio en memoria para almacenar precálculos en una tabla, lo cual se traduce en un menor tiempo de computación de B. Es decir, se compensa el espacio de memoria con la reducción del tiempo de ejecución.

Este algoritmo requiere $O(\sqrt{n})$ elementos del grupo. La tabla de precálculos requiere $O(\sqrt{n})$ multiplicaciones para ser construida, y $O(\sqrt{\mathrm{nlog}\, n})$ comparaciones para ser ordenada. Una vez construida la tabla, el cálculo final toma $O(\sqrt{n})$ multiplicaciones y $O(\sqrt{n})$ búsquedas en la tabla. Bajo la asumpción de que una multiplicación en el grupo toma más tiempo que $\log n$ comparaciones, el tiempo de ejecución del algoritmo puede ser $O(\sqrt{n})$.

- **Pollard's rho:** es un algoritmo aleatorio para computar logaritmos discretos con un tiempo de ejecución esperado similar al algoritmo *Baby-step giant-step*, sin embargo, el espacio requerido para el almacenamiento de precálculos es mínimo en comparación. Por esta razón es ampliamente preferido en lugar del algoritmo *Baby-step giant-step* para problemas de interés práctico.

Sea n un número primo y G un grupo de orden n. Asúmase que la función $F: G \to G$ definida por la ecuación:

$$x_{i+1} = f(x_i) = \begin{cases} \beta \cdot x_i, & if\ x_i \in S_1, \\ x_i^2, & if\ x_i \in S_2, \\ \alpha \cdot x_i, & if\ x_i \in S_3, \end{cases}$$

se comporta como una función aleatoria. En consecuencia, el tiempo de ejecución esperado del algoritmo Pollard's rho para algoritmos discretos en G es $O(\sqrt{n})$ operaciones de grupo.

- **Pohlig-Hellman:** este algoritmo utiliza la factorización del orden n del grupo G. Sea $n = p_1^{e_1} p_2^{e_2} \ldots p_r^{e_r}$ la factorización prima de n. Si $x = \log_\alpha \beta$, entonces el enfoque consiste en determinar $x_i = x \bmod p_i^{e_i}$ para $1 \le i \le r$, y entonces utilizar el algoritmo de Gauss para recuperar $x \bmod n$. Cada entero x_i es determinado por la computación de dígitos $l_0, l_1, \ldots, l_{e_i - 1}$ a la vez de su representación p_i-aria: $x_i = l_0 + l_1 p_i + \cdots + l_{e_i - 1} p_i^{e_i - 1}$, donde $0 \le l_j \le p_i - 1$.

Dada la factorización de n, el tiempo de ejecución del algoritmo Pohlig-Hellman es $O(\sum_{i=1}^{r} e_i(\ln n + \sqrt{p_i}))$

- *Index-calculus*: este algoritmo es uno de los mejores métodos conocidos para computar logaritmos discretos. La técnica empleada no puede ser aplicada a todos los grupos, pero cuando puede ser aplicada, usualmente el tiempo de computación es subexponencial.

El algoritmo requiere la selección de un subconjunto S relativamente pequeño de elementos de G, llamado *factor base*, de modo tal que una fracción de elementos de G pueda ser expresada eficientemente como productos de elementos de S. El algoritmo procede a precomputar una base de datos que contiene los logaritmos de todos los elementos de S, y luego la utiliza cada vez que el logaritmo de un elemento particular del grupo es requerido.

2.13. El Algoritmo Double Ratchet

El algoritmo Double Ratchet es utilizado por dos partes para intercambiar mensajes cifrados basados en una clave secreta compartida. Típicamente las partes utilizan un protocolo de acuerdo de clave (Por ejemplo, X3DH[8]) para acordar una clave secreta. A continuación, las partes utilizan Double Ratchet para enviar y recibir sus mensajes cifrados.

Las partes derivan nuevas claves por cada mensaje en Double Ratchet de modo que las claves antiguas no pueden ser calculadas a partir de las nuevas. Las partes también envían valores públicos de Diffie-Hellman adjuntos a sus mensajes. El resultado de los cálculos de Diffie-Hellman es combinado en las claves derivadas, de esta forma, las claves nuevas no pueden ser calculadas a partir de las antiguas. Estas propiedades proveen protección a los mensajes antiguos y nuevos en caso del compromiso de alguna de las claves.

2.13.1. Cadenas de KDF

Las cadenas de KDF son un concepto principal en el algoritmo Double Ratchet.

Se definen como una función criptográfica que toma un secreto, una clave KDF aleatoria y datos de entrada y retorna datos de salida. Los datos de salida son indistinguibles de datos aleatorios siempre que no se conozca la clave.

A continuación, se presenta un diagrama para el esclarecimiento del proceso, una clave KDF y algunos datos se utilizan como entrada de la función KDF, la salida se divide en dos partes, una se utiliza de entrada para la siguiente KDF y la otra como clave para el cifrado del mensaje.

[8] X3DH o Extended Triple Diffie-Hellman es un protocolo de acuerdo de clave que establece un secreto compartido entre dos partes que se autentican mutuamente utilizando claves públicas. [23]

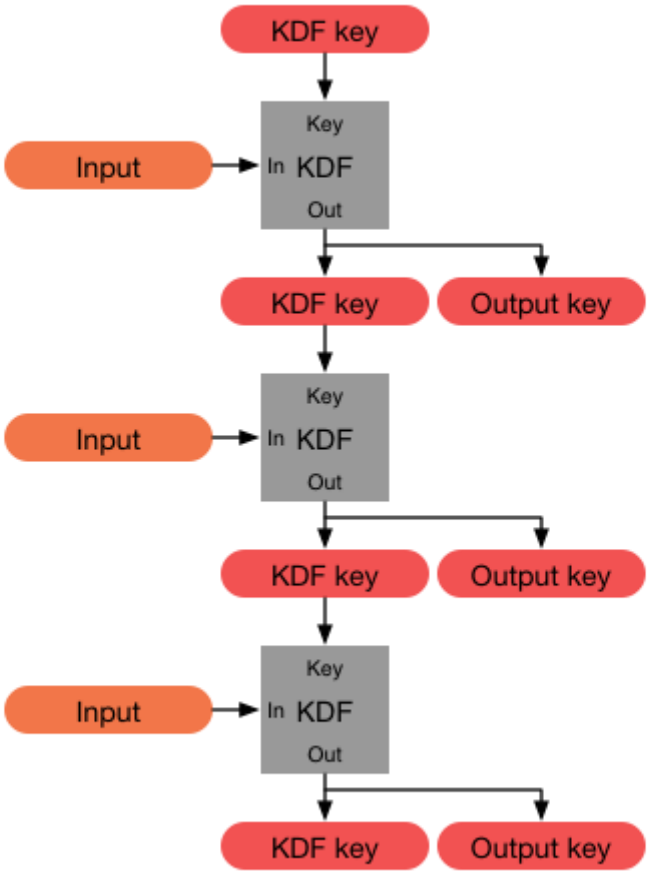

Ilustración 4. Cadena KDF. Fuente:[24]

Una cadena KDF tiene las siguientes propiedades:

- Resiliencia: Las claves de salida aparentan aleatoriedad para un adversario sin conocimiento de las claves KDF. Esto es verdad inclusive si el adversario puede controlar los datos de entrada de la función KDF.
- Forward Security: Las claves de salida del pasado aparentan aleatoriedad para un adversario que consigue la clave KDF en algún momento del tiempo.
- Break-in recovery: Las claves de salida futuras aparentan aleatoriedad para un adversario que consigue la clave KDF en algún momento del tiempo, siempre y cuando se les haya añadido suficiente entropía a los datos de entrada.

En una sesión entre Alice y Bob, cada uno de ellos almacena una clave KDF para tres cadenas: la cadena raíz, la cadena de envío y la cadena de recepción (La cadena de envío de Alice es igual a la cadena de recepción de Bob y viceversa)

Con cada intercambio de mensajes, Alice y Bob, también intercambian nuevas claves públicas Diffie-Hellman, y los secretos de salida Diffie-Hellman se

convierten en entradas de la cadena raíz. Esto se denomina el Ratchet Diffie-Hellman.

Las cadenas de envío y recepción avanzan con cada mensaje enviado y recibido. Sus salidas son utilizadas para cifrar y descifrar mensajes. Esto se denomina el Ratchet de clave simétrica. [24]

2.13.2. El ratchet Diffie-Hellman

Si un atacante roba las claves de las cadenas de envío y recepción de una de las partes, el atacante puede computar todas las claves futuras y descifrar los mensajes futuros. Para prevenir esto, el algoritmo Double Ratchet combina el ratchet de clave simétrica con un ratchet Diffie-Hellman el cual actualiza las claves de las cadenas basadas en salidas de Diffie-Hellman.

Para implementar este ratchet, cada parte genera un par de claves Diffie-Hellman (una clave privada y una pública) las cuales se convierten en sus pares de claves de ratchet actuales. Cada mensaje de cualquiera de las partes inicia con un encabezado que contiene la clave pública actual del ratchet. Cuando una nueva clave pública de ratchet es recibida desde una parte remota, se realiza un paso en el ratchet Diffie-Hellman, esto reemplaza el par de claves actual del ratchet de la parte local con un nuevo par de claves.

Este comportamiento, en el cual las partes toman turnos para reemplazar los pares de claves de los ratchets, permite que, si un atacante compromete una de las partes, pueda conocer el valor de la clave privada actual del ratchet, pero esa clave privada será eventualmente reemplazada por una que no haya sido comprometida.

En el siguiente diagrama se puede visualizar un resumen del proceso:

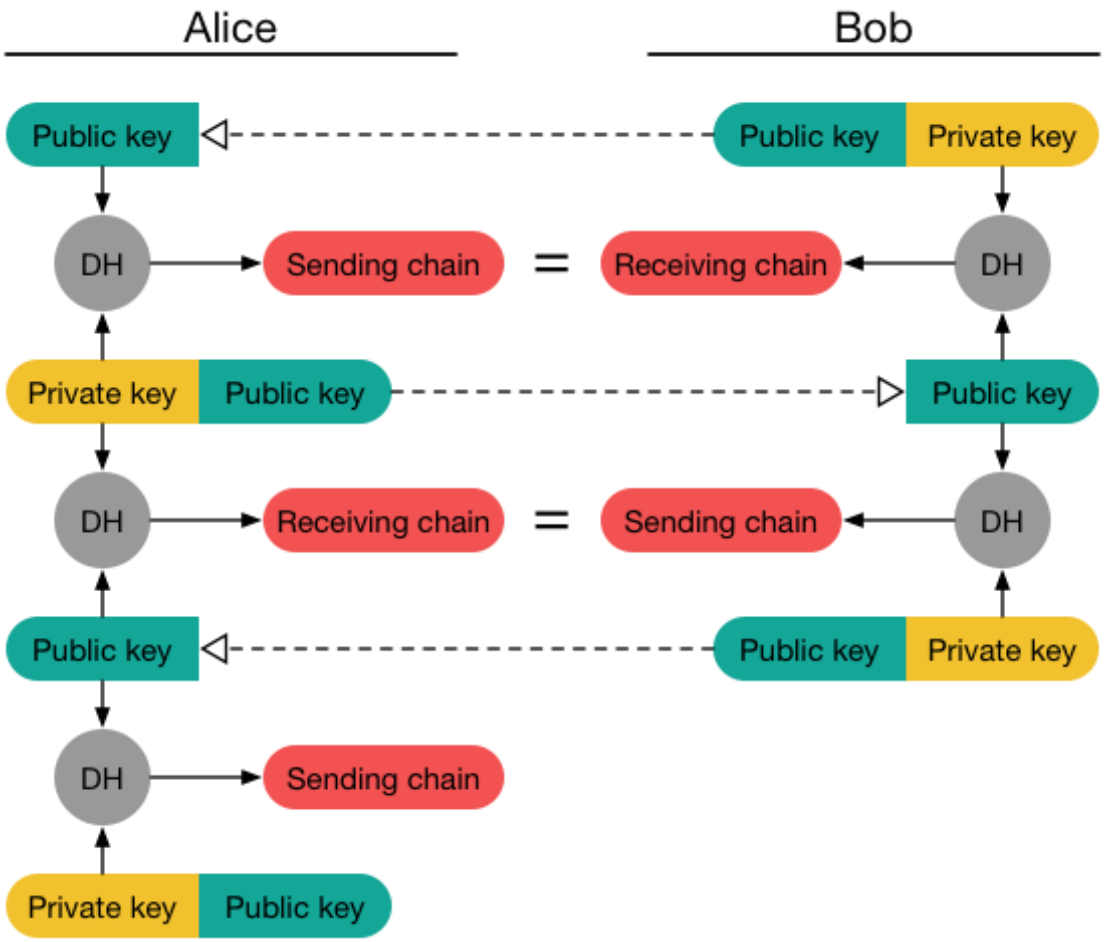

Ilustración 5. Diffie-Hellman Ratchet. Fuente:[24]

El diagrama de arriba es una simplificación, en lugar de tomar las claves de las cadenas directamente de las salidas de Diffie-Hellman, estas últimas son utilizadas como entrada de una KDF en una cadena raíz, y las salidas de la cadena raíz son utilizadas como claves de las cadenas de envío y recepción. La utilización de una KDF incrementa la resiliencia y la recuperación ante compromisos.

Al combinar el ratchet de clave simétrica y el Diffie-Hellman se obtiene el algoritmo Double Ratchet:

- Cuando un mensaje es enviado o recibido, se avanza un paso en el ratchet de clave simétrica, aplicándolo a la cadena de envío o recepción para derivar la clave del mensaje.
- Cuando una clave pública de ratchet es recibida, se avanza un paso en el ratchet de Diffie-Hellman para reemplazar a las claves de las cadenas.

3. El protocolo inicial

El protocolo inicial supone que dos usuarios, Alice y Bob, pretenden cifrar sus comunicaciones. Este acuerdo puede ser unidireccional, desde un usuario hacia el otro, y consiste en que ambos usuarios cuenten con una matriz A que sea igual para ambos.

También se propone un protocolo de renovación de clave, que puede ser ejecutado en caso de que el secreto compartido se considere comprometido[9].

Pasos que seguir para llegar a un secreto compartido entre Alice y Bob:

1- Supóngase que Alice y Bob acuerdan una matriz A, el secreto compartido, a través de un canal abierto, como internet[10].
2- Alice y Bob acuerdan la transformación lineal que, aplicada a una de las filas o columnas, hace que la matriz A tenga rango deficiente.
3- Bob envía a Alice los vectores z y b.
4- Alice aplica la transformación lineal a la matriz A y procede a resolver la ecuación (4)
5- Bob aplica la transformación lineal a la matriz A y procede a resolver la ecuación (4)
6- Alice y Bob ahora comparten el mismo secreto compartido x.

A modo de clarificar el funcionamiento del protocolo de intercambio de claves, se presenta el siguiente ejemplo:

Alice	**Bob**
Acuerdan la matriz $A^{3\times3}$	
Acuerdan que: $a_{11} = a_{12} * b_{11} + a_{13} * b_{12}$, $a_{21} = a_{22} * b_{11} + a_{23} * b_{12}$ y $a_{31} = a_{32} * b_{11} + a_{33} * b_{12}$	
	Envía z y b a Alice
Alice calcula a_{11}, a_{12} y a_{13}	Bob calcula a_{11}, a_{12} y a_{13}
Alice calcula: $x = A^{\dagger}b + (I - A^{\dagger}A)z$ $x = A^{\dagger}(b - Az) + z$	Bob calcula: $x = A^{\dagger}b + (I - A^{\dagger}A)z$ $x = A^{\dagger}(b - Az) + z$

Alice y Bob ahora poseen el mismo secreto compartido $x = A^{\dagger}(b - Az) + z$.

Ahora pueden utilizar este secreto compartido como clave o pueden usarlo de entrada para algún algoritmo de derivación de clave, la salida de dicho algoritmo sería la clave de cifrado de un protocolo de clave simétrica como el AES[11].

[9] Se considera que una clave o un secreto está comprometido cuando un tercero no autorizado tiene conocimiento de este.
[10] En la siguiente sección se explica por qué no es seguro.
[11] Advanced Encryption Standard

En caso de que Alice o Bob consideren que su secreto compartido x pueda estar comprometido, pueden iniciar el protocolo de renovación de clave, este consiste en que una de las dos partes genere de forma aleatoria los vectores z, b y los envíe a la otra. Luego, se realiza el cálculo de los elementos a_{11}, a_{12} y a_{13} y se vuelve a generar el secreto compartido x.

A modo de clarificar el funcionamiento del protocolo de renovación de clave, se presenta el siguiente ejemplo en el que Bob considera que el secreto compartido fue comprometido:

Alice	**Bob**
	Genera z, b y los envía a Alice
Alice calcula a_{11}, a_{12} y a_{13}	Bob calcula a_{11}, a_{12} y a_{13}
Alice calcula: $x = A^{\dagger}b + (I - A^{\dagger}A)z$ $x = A^{\dagger}(b - Az) + z$	Bob calcula: $x = A^{\dagger}b + (I - A^{\dagger}A)z$ $x = A^{\dagger}(b - Az) + z$

Al igual que en el caso anterior, se puede usar este secreto compartido como clave o como entrada para algún algoritmo de derivación de clave. Esta capacidad de generar nuevas claves sin tener que reiniciar el protocolo está inspirada en el funcionamiento del algoritmo *double-ratchet* del protocolo Signal, visto anteriormente en este trabajo.

3.1. Seguridad del protocolo

Si Alice y Bob logran acordar la matriz A de forma segura, la atacante Eve, deberá resolver un sistema de ecuaciones lineales subdeterminado, es decir, un sistema en el que se tienen menos ecuaciones que variables desconocidas. Este tipo de sistemas puede tener infinitas soluciones, por lo tanto, un ataque por fuerza bruta de parte de Eve se vuelve computacionalmente imposible.

No obstante, Alice y Bob se enfrentan al problema de distribución de claves. Si ellos se encuentran separados de modo que no pueden tener contacto directo o no poseen un canal seguro de transmisión para establecer la matriz inicial A, como es el caso en el ejemplo provisto en el punto anterior, Eve puede llevar a cabo un ataque del tipo *man-in-the-middle*, generar el mismo secreto x y desencriptar la comunicación se vuelve trivial.

Mediante el *forward secrecy* es posible que cada mensaje que sea enviado entre Alice y Bob sea cifrado con una clave diferente, incrementando la dificultad de éxito para el ataque de Eve.

Al mismo tiempo, permite incluirlo en la categoría de “protocolos con auto reparación”, los cuales reciben este nombre porque si una clave es comprometida, el atacante tendrá acceso únicamente al mensaje que fue cifrado utilizando esa clave.

A continuación, en el esquema, se puede observar lo descrito anteriormente. Bob genera un mensaje M1 y su clave correspondiente K1, cifra el mensaje utilizando la clave, de esta forma obtiene C1, y envía C1 a Alice. Alice utiliza su clave K1 para descifrar C1 y obtener así M1. El proceso se repite cuando Alice genera un mensaje de respuesta para Bob.

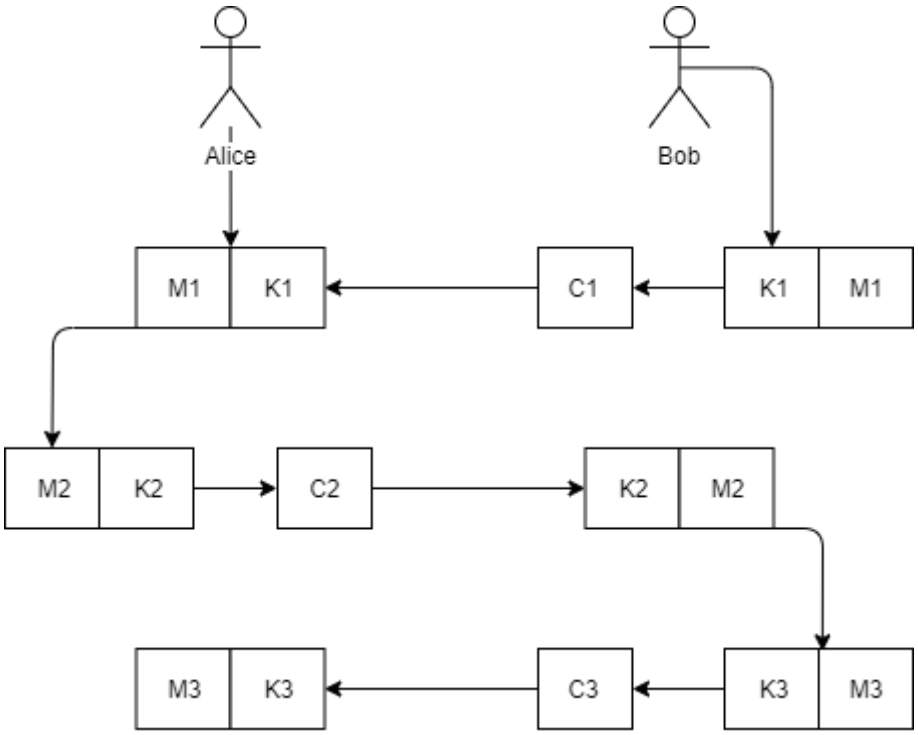

Ilustración 6. Protocolo inicial. Fuente: Elaboración propia.

Ahora bien, en el ejemplo provisto anteriormente, Alice y Bob no cuentan con un canal seguro para la transmisión de la matriz inicial A, por consiguiente, la seguridad del protocolo es nula. Esto significa que para que el protocolo sea realmente seguro, se requiera extenderlo, añadiéndole una capa de intercambio de secreto inicial que pueda ser utilizada para establecer la matriz inicial A de manera segura a través de un canal inseguro.

Las matrices llevan siendo utilizadas en la criptografía desde hace ya bastante tiempo, siendo el cifrador de Hill uno de los métodos más conocidos. Inclusive existen algoritmos de clave pública que hacen uso de esta estructura matemática.

Durante la elaboración del trabajo, se revisaron varios de estos algoritmos, entre los cuales se pueden destacar los siguientes[12][13]:

[12] Los protocolos no tienen nombre por lo cual se nombra a sus autores.

[13] Los algoritmos se describen en alto nivel, para profundizar en la matemática detrás de los mismos, referirse a los documentos referenciados.

3.1.1. Mukesh Kumar Singh en [25]

El autor propone un protocolo de intercambio de claves y un algoritmo de cifrado utilizando multiplicación de matrices sobre un anillo conmutativo.

. Tiene ciertas similitudes con el protocolo Diffie-Hellman y funciona de la siguiente manera.

Supóngase que dos partes, Alice y Bob, necesitan establecer una clave. Acuerdan una matriz inicial G, esta matriz es pública. Alice genera dos matrices circulantes aleatorias A_1 y A_2 , calcula el valor público $P_1 = A_1 \cdot G \cdot A_2$ y lo envía a Bob.

Bob, por su parte, genera dos matrices circulantes aleatorias B_1 y B_1 , calcula el valor público $P_2 = B_1 \cdot G \cdot B_2$ y lo envía a Alice.

Alice tiene $P_2 = B_1 \cdot G \cdot B_2$, A_1 y A_2 . Alice calcula

$$S = A_1 \cdot P_2 \cdot A_2 = A_1 \cdot B_1 \cdot G \cdot B_2 \cdot A_2 = A_1 \cdot B_1 \cdot G \cdot A_2 \cdot B_2$$

Esto es posible debido a que la multiplicación de matrices circulantes es conmutativa.

Bob, por su parte, tiene $P_1 = A_1 \cdot G \cdot A_2$, B_1 y B_1 . Bob calcula

$$S = B_1 \cdot P_1 \cdot B_2 = B_1 \cdot A_1 \cdot G \cdot A_2 \cdot B_2 = A_1 \cdot B_1 \cdot G \cdot A_2 \cdot B_2$$

Por lo tanto, ambas partes tienen $S = A_1 \cdot B_1 \cdot G \cdot A_2 \cdot B_1$ como clave secreta. Para la elección de A_1 , A_2 y B_1 , B_2 se debe tener en cuenta que deben conmutar entre sí, pero no con G.

A continuación, se ilustra el protocolo para un mejor entendimiento.

Alice	**Bob**
Acuerdan la matriz $G = \begin{bmatrix} 7 & 2 \\ 23 & 3 \end{bmatrix}$ tal que $\det G_n = 0$	
Genera dos matrices circulantes aleatorias, $A_1 = \begin{bmatrix} 13 & 11 \\ 11 & 13 \end{bmatrix}$ y $A_2 = \begin{bmatrix} 15 & 17 \\ 17 & 15 \end{bmatrix}$, no deben conmutar con G	
Calcula $P_1 = A_1 \cdot G \cdot A_2 = \begin{bmatrix} 3 & 13 \\ 27 & 27 \end{bmatrix}$	
Envía P_1 a Bob	
	Genera dos matrices circulantes aleatorias, $B_1 = \begin{bmatrix} 17 & 2 \\ 2 & 17 \end{bmatrix}$ y $B_2 = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}$, no deben conmutar con G
	Calcula $P_2 = B_1 \cdot G \cdot B_2 = \begin{bmatrix} 15 & 30 \\ 30 & 30 \end{bmatrix}$
	Envía P_2 a Alice

Calcula $S = A_1 \cdot P_2 \cdot A_2 = A_1 \cdot B_1 \cdot G \cdot B_2 \cdot A_2 = A_1 \cdot B_1 \cdot G \cdot A_2 \cdot B_2 = \begin{bmatrix} 25 & 20 \\ 20 & 5 \end{bmatrix}$	
	Calcula $S = B_1 \cdot P_1 \cdot B_2 = B_1 \cdot A_1 \cdot G \cdot A_2 \cdot B_2 = A_1 \cdot B_1 \cdot G \cdot A_2 \cdot B_2 = \begin{bmatrix} 25 & 20 \\ 20 & 5 \end{bmatrix}$

Nótese que todas las multiplicaciones de matrices utilizadas en el ejemplo son modulo $n = 35$. Se puede observar que tanto S_1 como S_2 tienen el mismo valor, por lo tanto, Alice y Bob ahora comparten el mismo secreto.

De acuerdo con el autor, la seguridad de este algoritmo se basa en la dificultad de resolver un sistema de ecuaciones polinomiales multivariadas. Este problema es considerado NP-Hard en cualquier campo.

3.1.2. Rafael Alvarez et al. en [26]

El autor propone una aplicación de matrices bloque al protocolo de intercambio de claves de Diffie-Hellman.

Dado un número primo p y $r, s \in \mathbb{N}$, se denota por $Mat_{r\times s}(\mathbb{Z}_p)$ las matrices de tamaño $r \times s$, con elementos en $\mathbb{Z}_p$, y por $GL_r(\mathbb{Z}_p)$ y $GL_s(\mathbb{Z}_p)$ las matrices invertibles de tamaño $r \times r$ y $s \times s$. Se define:

$$\theta = \left\{ \begin{bmatrix} A & X \\ 0 & B \end{bmatrix}, A \in GL_r(\mathbb{Z}_p), B \in GL_s(\mathbb{Z}_p), X \in Mat_{r\times s}(\mathbb{Z}_p) \right\}$$

El proceso para establecer un secreto compartido entre dos partes, nuevamente Alice y Bob, consiste en lo siguiente:

1. Alice y Bob acuerdan $p \in \mathbb{Z}$, $M_1 = \begin{bmatrix} A_1 & X_1 \\ 0 & B_1 \end{bmatrix} \in \theta$, con orden m_1 y $M_2 = \begin{bmatrix} A_2 & X_2 \\ 0 & B_2 \end{bmatrix} \in \theta$, con orden m_2.
2. Alice genera aleatoriamente dos claves privadas r y s tal que $1 \leq r \leq m_1$, $1 \leq s \leq m_2$ y computa
$$C = M_1^r \cdot M_2^s$$
y publica este valor.
3. Bob genera aleatoriamente dos claves privadas v y w tal que $1 \leq v \leq m_1$, $1 \leq w \leq m_2$ y computa
y publica este valor
$$F = M_1^v \cdot M_2^w$$
$$D = M_1^v \cdot C \cdot M_2^w$$
$$= M_1^v \cdot M_1^r \cdot M_2^s \cdot M_2^w$$

$= M_1^{v+r} \cdot M_2^{s+w}$

$= M_1^{r+v} \cdot M_2^{w+s}$

$= M_1^{r} \cdot M_1^{v} \cdot M_2^{w} \cdot M_2^{s}$

y publica este valor.

4. Alice calcula M_1^{-r} , M_2^{-s} y

$$D = M_1^{-r} \cdot D \cdot M_2^{-s}$$
$$= M_1^{-r} \cdot M_1^{r} \cdot M_1^{v} \cdot M_2^{w} \cdot M_2^{s} \cdot M_2^{-s}$$
$$= M_1^{v} \cdot M_2^{w}$$

5. Las claves públicas de Alice y Bob son respectivamente C y D.

De esta forma, el secreto compartido entre Alice y Bob es la matriz F. Un atacante podría conocer p y M , pero para obtener el secreto compartido deberá resolver un problema de complejidad similar a la del problema del logaritmo discreto.

3.1.3. Eligijus Sakalauskas en [27]

El autor propone una función de potenciación de matrices (MPF[14]) para la construcción de primitivas criptográficas. Una MPF es una acción de dos matrices que potencian, a la derecha e izquierda, una matriz base. Las ecuaciones de inversión de MPF, correspondientes al problema MPF, son derivados y tienen estructura similar a las ecuaciones del problema cuadrático multivariante (MQ[15]). De acuerdo con el autor, parecería ser que los problemas MPF son más complicados, debido a que las ecuaciones no están definidas sobre el campo, sino que están representadas como acciones de izquierda a derecha de dos matrices definidas sobre el casi-semi anillo infinito en la matriz definida sobre un semigrupo infinito, aditivo y no conmutativo.

Sin embargo, el autor deja en claro que se requiere más investigación para determinar si efectivamente el problema MPF cuenta con la dificultad supuesta.

3.1.4. Adinarayana Reddy K en [28]

El autor propone una versión modificada del cifrador de Hill basado en matrices circulantes, donde una matriz circulante de orden primo es compartida como un secreto compartido y una matriz G no singular es elegida clave pública de forma que el determinante de la matriz coeficiente G_n es cero. El algoritmo funciona de la siguiente manera:

[14] Por sus siglas en inglés Matrix Power Function.
[15] Por sus siglas en inglés Multivariate Problem.

1- Supóngase que dos partes, Alice y Bob, desean establecer un secreto compartido.

2- Alice y Bob seleccionan una matriz G no singular de tamaño $n \times n$ en $GF(P)$ como clave pública, tal que $\det(G_c) = 0$.

3- Alice selecciona una matriz circulante prima A de tamaño $n \times n$ en $GF(P)$ como clave secreta.

4- Alice calcula la clave $K = A \cdot G \cdot A^{-1}\ mod\ P$ y se la envía a Bob.

Cifrado:

- M_i es el i-esimo bloque de texto plano de tamaño n
- C_i es el i-esimo bloque de texto cifrado
- $C_i = K \cdot M_i + V_i \cdot T\ mod\ P$, donde V_i es la i-esima fila de la matriz circulante A

Descifrado:

- Calcular $K^{-1} = A \cdot G^{-1} \cdot A^{-1}\ mod\ P$
- $M_i = K^{-1} \cdot (C_i - V_i \cdot T)\ mod\ P$

Aquí, V es la primera fila de la matriz circulante prima A. Por cada bloque de texto plano a ser encriptado, se usa un vector diferente V por rotación. Esto elimina el ataque conocido como "texto plano conocido"[16]. Al mismo tiempo previene los ataques del tipo "solo texto cifrado"[17] debido a que el módulo es un número primo y el de tipo "texto plano elegido"[18].

La seguridad del algoritmo radica en la dificultad de resolver ecuaciones polinómicas multivariables, por ejemplo, $K = A \cdot G \cdot A^{-1}\ mod\ P$. Este es un problema del tipo NP-Difícil, debido a que resolver el módulo es difícil cuando se utiliza un número primo muy grande.

[16] KPA por sus siglas en inglés (Known-plaintext attack) es una metodología de ataques donde el atacante tiene acceso a tanto el texto plano como al texto cifrado. Estos pueden ser utilizados para descubrir más información secreta como claves y libros de códigos. Un caso ejemplo bien conocido de esta metodología es el utilizado para descifrar mensajes cifrados con la máquina Enigma durante la segunda guerra mundial, en el cual los mensajes de los alemanes concluían siempre con el texto "Heil Hitler". [29]

[17] COA por sus siglas en inglés (Ciphertext-only attack) es una metodología de ataques donde se asume que el atacante tiene acceso únicamente a los textos cifrados. Sin embargo, el atacante puede poseer cierto conocimiento sobre el texto plano, por ejemplo, el lenguaje en el que está escrito y por lo tanto la distribución estadística de los caracteres. [30]

[18] CPA por sus siglas en inglés (Chosen-plaintext attack) es una metodología de ataques donde se presume que el atacante puede obtener textos cifrados a partir de textos planos arbitrarios. El objetivo de este ataque es obtener información que reduzca la seguridad del esquema de cifrado. Los algoritmos de cifrado modernos se esfuerzan por proveer seguridad semántica, es decir, que el texto cifrado sea indistinguible bajo el ataque de texto plano elegido, por lo tanto, ser inmunes a este tipo de ataques por diseño.

3.1.5. Augmented Hill Cipher [31]

El autor demuestra que el protocolo de Reddy *et al.* es vulnerable a los ataques del tipo KPA y CPA, y al mismo tiempo propone una mejora al cifrador de Hill original. El proceso de cifrado es el siguiente:

Al inicio de cada sesión, las partes deben acordar tres claves secretas diferentes y aleatorias K_0, K_1 y K_2. Cada una de estas claves secretas es una matriz invertible $m \times m$ en Z_n donde n es un entero. Los enteros m y n son parámetros de seguridad.

El texto plano a ser transferido debe ser dividido en $P_0, P_1, P_2, P_3, \dots, P_N$ donde P_i es una matriz $m \times m$ sobre un entero n e $i = 0,1,2,3, \dots, N$. Cada uno de los P_i es considerado como un bloque de texto plano de longitud L, donde $L = m^2 \times \lceil \log_2 n \rceil$ bits. Si la longitud del texto plano no es múltiplo de L, se deben agregar bits de manera aleatoria[19] al final del mismo.

Las matrices P_i son cargadas, inicialmente, columna por columna, con el texto plano, por ejemplo, el primer grupo de $\lceil \log_2 n \rceil$ bits son cargados en la celda de la columna 1-fila 1, el segundo grupo de $\lceil \log_2 n \rceil$ bits son cargados en la celda de la columna 2-fila 2, y así sucesivamente. Luego, las matrices de texto cifrado $C_0, C_1, C_2, C_3, \dots, C_N$, de tamaño $m \times m$, pueden ser computadas de la siguiente manera:

$$C_i' = \left(K_{i \bmod 3} P_i + K_{(i+1) \bmod 3}\right) mod\ n, i = 0,1,2, \dots, N$$

$$C_i = \begin{cases} C_i' \oplus K_2 & si\ i = 0 \\ C_i' \oplus C_{i-1}' & si\ i = 1,2, \dots, N \end{cases}$$

Para el descifrado, el texto plano P_i puede ser computado a partir de su correspondiente texto cifrado C_i de la siguiente manera:

$$C_i' = \begin{cases} C_i \oplus K_2 & si\ i = 0 \\ C_i \oplus C_{i-1} & si\ i = 1,2, \dots, N \end{cases}$$

$$P_i = \left[K_{i \bmod 3}^{-1} * (C_i' - K_{(i+1) \bmod 3})\right] mod\ n, , i = 0,1,2, \dots, N$$

Como se mencionó con anterioridad en esta sección, m y n son parámetros de seguridad, lo cual significa que la seguridad del algoritmo depende directamente de los valores de m y n. Si se eligen m y n con valores muy grandes, entonces el sistema será más seguro pero, al mismo tiempo, perderá eficiencia, es decir, el proceso se ralentiza.

El autor declara que, si bien el espacio de claves es mucho mayor a la del cifrador de Hill original, la complejidad del algoritmo es similar a la de otras variantes del mismo cifrador.

3.1.6. Pedro Hecht en [32]

El autor propone un protocolo de intercambio de claves basado en las siguientes definiciones y convenciones:

[19] Los bits agregados llevan el nombre de *padding*.

- Espacios M_8 y P_8

El espacio no conmutativo elegido es el grupo general lineal[20] $GL(8, Z_{251})$ también conocido en criptografía como el grupo de matrices de Hill. El módulo 251 es el mayor primo representable en 8 bits. El espacio $GL(8, Z_{251})$ se define como M_8. La elección del orden 8 obedece a que ofrece una aceptable solución de compromiso entre requerimientos de memoria, complejidad computacional y seguridad criptográfica, tal como se define más adelante.

Para obtener matrices aleatorias $M \in_R M_8$ se procede a generar 64 enteros al azar y uniformemente distribuidos (simbolizado $\in_R$) en Z_{251} y se acepta si $\det(M)$ no fuese nulo. Dado que el módulo es primo, el hecho que la matriz no sea singular asegura su inversibilidad. Basándose en dicho argumento, las matrices así generadas serán aleatorias y uniformemente distribuidas en el espacio M_8.

Todo grupo no conmutativo posee subgrupos que sí lo son. Para los grupos matriciales existe una condición necesaria y suficiente que expresa que dos matrices no singulares a y b conmutan $(a.b = b.a)$ si comparten una misma base ortonormal, es decir poseen la misma matriz de autovectores. Esta condición puede ser explotada para generar matrices aleatorias de Hill orden $n\ (mod\ p)$ que conmutan usando el algoritmo de la Tabla 1. Se define P_8 como el subgrupo conmutativo de M_8 que haya sido generado a partir de una determinada matriz P de autovectores. Queda claro que $P_8 \subset M_8 \equiv GL(8, Z_{251})$.

Para obtener matrices privadas $A \in_R P_8$ se emplea una tercera parte de confianza (TPC) que procede primero a elegir y luego a publicar una matriz P al azar en M_8 y que oficiará de matriz de autovectores. Para obtener matrices diagonales[21] D al azar, se busca al azar un vector de autovalores uniformemente distribuido en Z_{251} con valores diferentes de a pares y no nulos $\Lambda = (\lambda_1, \dots, \lambda_8)$. En base a ese vector se genera la matriz diagonal D. Para obtener potencias en P_8, se define (arbitrariamente) una cota superior z. Ese entero debe brindar suficiente seguridad para combinaciones aleatorias de dos potencias $(m, n) \in_R [2, z]^2$ sin afectar sensiblemente el tiempo de cómputo de las potencias.

- **Función trampa de una vía**

Se opta por hacer uso del problema de la descomposición simétrica generalizada (GSDP), el cual consiste en lo siguiente:

Sea G un grupo no conmutativo y S un subgrupo abeliano:

Dados:

$$(x, y) \in G \times G, S \subset G, (m, n) \in Z,$$

[20] El grupo general lineal de grado n es el conjunto de matrices invertibles de tamaño $n \times n$, y la operación de multiplicación de matrices.

[21] Una matriz diagonal es una matriz cuadrada en la cual, todos los elementos son cero, con excepción de los de la diagonal principal.

Hallar:

$$z \in S \; tal \; que \; y = z^m x z^n$$

El autor declara que no queda claro que se puedan atacar variantes del problema por medio de técnicas conocidas. Por lo tanto, conjetura, aunque no demuestra, que el único ataque disponible es el de fuerza bruta por exploración sistemática del espacio de elementos. Por eso se incluye al problema GSDP como instancia de la clase de complejidad temporal NP.

- **Algoritmo para el cómputo de matrices inversas**

Para invertir matrices no singulares en M_8 se puede utilizar el algoritmo que se presenta en la Tabla 1.

$A^{-1}(mod\ p) = A^{-1}\ Det[A]\ Det^{-1}[A]\ (mod\ p)$

Tabla 1. Algoritmo de inversión de Matrices de Hill de orden n y módulo p.

Supóngase que Alice quiere transmitir a Bob una clave secreta K. El protocolo de transporte de claves consiste en los siguientes pasos:

1. Preparación

Parámetros	Alice	Bob
TPC (Tercer Parte de Confianza)	Define y publica $P \in_R P_8$	
Clave secreta a transportar	$K \in M_8$	
Claves Privadas	$D_A = (\lambda_1 \dots \lambda_8)$ (todos $\neq y > 0$) $D_A \in_R M_8$ $A = PD_AP^{-1}$	$D_B = (\lambda_1 \dots \lambda_8)$ (todos $\neq y > 0$) $D_B \in_R M_8$ $B = PD_BP^{-1}$

2. Alice elige $(k_1, k_2) \in_R [2, z]^2$ y luego genera y envía a Bob $T_A = A^{k_1} K A^{k_2}$
3. Bob elige $(r_1, r_2) \in_R [2, z]^2$ y luego genera y envía a Alice $T_B = B^{r_1} K B^{r_2}$
4. Alice calcula $S = A^{-k_1} T_B A^{-k_2}$
5. Bob calcula $K = B^{-r_1} S B^{-r_2}$
6. Validación

$$\begin{aligned} K &= B^{-r_1} S B^{-r_2} \\ &= (B^{-r_1} B^{r_1}) K (B^{-r_2} B^{r_2}), donde \\ S &= A^{-k_1} T_B A^{-k_2} \\ &= A^{-k_1} (B^{r_1} T_A B^{r_2}) A^{-k_2} \\ &= A^{-k_1} B^{r_1} (A^{k_1} K A^{k_2}) B^{r_2} A^{-k_2} \\ &= (A^{-k_1} A^{k_1}) B^{r_1} K B^{r_2} (A^{k_2} A^{-k_2}) \\ &= B^{r_1} K B^{r_2} \end{aligned}$$

Se puede observar que en ningún momento se comprometen las claves privadas a menos que se resuelva GSDP.

Para quebrar el protocolo se necesita deducir la clave privada de la entidad atacada. Dado que la clave privada depende de la generación de una matriz diagonal de orden ocho de elementos diferentes y no nulos, se puede computar que el cardinal del subgrupo conmutativo es:

$$|P_8| = 249.248.246.245.244.243.242 = 13190481178699144320 \approx 10^{19} \approx 2^{64}$$

Con los recursos computacionales vigentes resulta impracticable la exploración sistemática de este espacio. Si a esto se suma que la resolución del problema GSDP es, según lo conjeturado por el autor, de complejidad temporal NP, se puede deducir una seguridad criptográfica mínima de 64 bits para el protocolo.

4. Protocolo Final Propuesto

Como se mencionó previamente en el trabajo, la falta de un método de acuerdo de clave inicial implica precariedad en la seguridad del protocolo inicial propuesto. Para el proceso de mejora, se revisaron algunos algoritmos de intercambio de claves, de los cuales, se seleccionó el protocolo propuesto por Pedro Hecht. Esta elección se debe tanto a la simplicidad de implementación como a su eficiencia en consumo de recursos, lo cual permitirá la implementación en plataformas computacionalmente pobres.

No obstante, el algoritmo puede ser acoplado a otros en con el objetivo de incrementar la seguridad de estos, cualquiera que sirva para establecer la matriz inicial, sirve para ser acoplado al propuesto inicialmente.

Entonces, supóngase ahora que Alice y Bob quieren comunicarse de forma segura:

1. Preparación

Parámetros	Alice	Bob
TPC (Tercer Parte de Confianza)	Define y publica $P \in_R P_8$	
Clave secreta a transportar	$K \in M_8$	
Claves Privadas	$D_A = (\lambda_1 \dots \lambda_8)$ (todos $\neq y > 0$) $D_A \in_R M_8$ $A = PD_AP^{-1}$	$D_B = (\lambda_1 \dots \lambda_8)$ (todos $\neq y > 0$) $D_B \in_R M_8$ $B = PD_BP^{-1}$
Transformaciones Lineales	Arbitrarias	

2. Alice elige $(k_1, k_2) \in_R [2, z]^2$ y luego genera y envía a Bob $T_A = A^{k_1} K A^{k_2}$
3. Bob elige $(r_1, r_2) \in_R [2, z]^2$ y luego genera y envía a Alice $T_B = B^{r_1} K B^{r_2}$
4. Alice calcula $S = A^{-k_1} T_B A^{-k_2}$

5. Bob calcula $K = B^{-r_1} S B^{-r_2}$
6. Bob envía a Alice los vectores z y b.
7. Alice aplica las transformaciones lineales a la matriz A y procede a resolver la ecuación $x = A^{\dagger}b + (I - A^{\dagger}A)z$
8. Bob aplica las transformaciones lineales a la matriz A y procede a resolver la ecuación $x = A^{\dagger}b + (I - A^{\dagger}A)z$
9. Alice y Bob ahora comparten el mismo secreto compartido x.

A modo de aclarar el funcionamiento del protocolo, se presenta el siguiente ejemplo:

Alice	**Bob**
Generan la matriz secreta $A^{3\times3}$ utilizando el protocolo de Pedro Hecht.	
Acuerdan las transformaciones lineales: $a_{11} = a_{12} * b_{11} + a_{13} * b_{12}$, $a_{21} = a_{22} * b_{11} + a_{23} * b_{12}$ y $a_{31} = a_{32} * b_{11} + a_{33} * b_{12}$	
	Envía z y b a Alice
Alice calcula a_{11}, a_{12} y a_{13}	Bob calcula a_{11}, a_{12} y a_{13}
Alice calcula: $x = A^{\dagger}b + (I - A^{\dagger}A)z$ $x = A^{\dagger}(b - Az) + z$	Bob calcula: $x = A^{\dagger}b + (I - A^{\dagger}A)z$ $x = A^{\dagger}(b - Az) + z$

Alice y Bob ahora pueden utilizar el secreto compartido como clave de un algoritmo simétrico, por ejemplo, el AES. La matriz A es generada por sesión, una vez acabada la sesión, debe ser eliminada, al igual que el resto de las claves de mensajes. Como parte del protocolo, cada vez que se envíe un mensaje de un punto a otro, se deben enviar los vectores z, b de esta forma, cada mensaje de la red será cifrado con una clave diferente. Y en caso de que una parte desconfíe de la seguridad de las claves en un momento dado, puede solicitar el reinicio del protocolo para generar una matriz A completamente nueva. Esto evita que todos los mensajes sean comprometidos si un atacante descubre una clave, a lo sumo, el atacante será capaz de descifrar los mensajes de una sola sesión.

Al añadir el protocolo propuesto en el presente trabajo al de Hecht, se puede observar una mejoría en dos aspectos fundamentales:

Consumo de ancho de banda

Inicialmente, el protocolo propuesto por Hecht requiere dos transmisiones a través de la red para que ambas partes puedan acordar claves (T_A y T_B). Esto significa que para n claves, las dos partes van a realizar $2n$ transmisiones.

Cuando se agrega Zitram al protocolo, el acuerdo inicial requiere tres transmisiones, sin embargo, la generación de claves posterior, una sola (los vectores z, b). Entonces, para n claves, las dos partes necesitarán realizar $n + 2$

transmisiones por la red. Por lo tanto, se reduce casi a la mitad el consumo de recursos de red.

Incremento de seguridad de la clave:

El protocolo de Hecht genera claves a partir de un espacio limitado, si bien es cierto que explorar el espacio es computacionalmente inviable en la actualidad, es teóricamente posible. Al agregar Zitram, se elimina esta posibilidad porque no se trabaja en un espacio limitado para la generación de claves.

5. Conclusiones

El protocolo propuesto brinda una opción novedosa y segura, como alternativa, al protocolo de intercambio de claves más difundido en la actualidad, Diffie-Hellman. La posibilidad de acoplar el algoritmo inicial a otros algoritmos le concede una resiliencia notable al momento de realizar implementaciones. Conforme se inventen nuevos métodos de compartir matrices a través de un canal inseguro, se puede recurrir a agregarle seguridad con el protocolo propuesto.

La capacidad de generar nuevas claves para cada mensaje enviado por la red sin necesidad de reiniciar todo el protocolo permite un ahorro apreciable en ciclos de procesador y latencia de la red. Al mismo tiempo, esta característica provee al algoritmo de un mecanismo de "auto reparación", es decir que, si una clave es comprometida, el atacante no podrá descifrar más que un mensaje, en otras palabras, el algoritmo propuesto implementa el *forward secrecy*.

Tanto Diffie-Hellman como otros protocolos de intercambio de claves (incluido el utilizado como base para el protocolo final propuesto) realizan sus operaciones en un grupo, es decir, existe un número limitado de valores posibles para las claves, si bien este grupo puede ser muy grande en algunos casos, esta limitación en la cantidad de elementos permite que los ataques del tipo búsqueda exhaustiva o fuerza bruta (como los que fueron descritos en el presente trabajo), sean viables porque en el peor de los casos le tomaría n intentos hallar la clave n, sin embargo, cuando se tienen infinitos elementos posibles, como es el caso del protocolo final propuesto, la búsqueda exhaustiva se vuelve inviable, porque en el peor de los casos, le puede tomar un tiempo indefinido hallar la respuesta correcta.

Si bien en el presente trabajo no se realizó un criptoanálisis exhaustivo del protocolo, se demostró que a pesar de ser susceptible a ser atacado de la misma forma que puede ser atacado Diffie-Hellman, es decir, utilizando el ataque *man-in-the-middle*, el protocolo puede ser adaptado fácilmente para mitigar esta vulnerabilidad, incorporándolo a otros para incrementar su seguridad. Se propuso un ejemplo exitoso agregándole Zitram al protocolo propuesto por Hecht se redujo considerablemente la carga en la red requerida para generar claves y se eliminó la posibilidad de realizar un ataque de búsqueda exhaustiva sobre el mismo.

5.1. Prueba de hipótesis

Tal como se la presentó en el capítulo introductorio, la hipótesis de este trabajo es la siguiente:

Zitram es un protocolo de intercambio de claves criptográficamente seguro y, como tal, es una alternativa válida al protocolo Diffie-Hellman.

Para validar dicha hipótesis se procederá a desmenuzar las frases que componen dicha afirmación:

“Zitram es un protocolo de intercambio de claves ...” Zitram fue concebido como una técnica en la cual, dos partes pueden obtener un secreto compartido a partir de datos proveídos por las partes y sin la necesidad de transmitirlo a través de la red.

“... criptográficamente seguro” Si bien el algoritmo principal de Zitram es vulnerable al ataque del tipo MITM, este puede ser utilizado en conjunto a otros algoritmos que sean capaces de inicializar la matriz base del algoritmo. La seguridad de Zitram, en consecuencia, depende de la seguridad del algoritmo que inicializa la matriz base. Esta flexibilidad de acoplar el algoritmo a otros permite el incremento de la seguridad conforme se desarrollen nuevos métodos de intercambio de claves. La seguridad de la versión propuesta al final de este protocolo

“... es una alternativa válida al protocolo Diffie-Hellman” anteriormente la capacidad de cómputo y lentitud en la transmisión a través de de la red representaban parámetros muy críticos a tener en cuenta al momento de elegir un protocolo u otro, si bien en la actualidad siguen siendo parámetros importantes, la velocidad de procesamiento y transmisión de datos se incrementó drásticamente y sigue creciendo a ritmo acelerado. Es cierto que la multiplicación de matrices requiere más ciclos de procesamiento que una potencia o una multiplicación de enteros, sin embargo, la velocidad de generación de claves utilizando Zitram se adecua a los estándares actuales. Existen incontables alternativas a Diffie-Hellman, algunas más seguras que otras, la seguridad, flexibilidad de integración y velocidad de resolución de Zitram cumplen con los requerimientos para ser considerado una alternativa válida.

5.2. Mejoras Futuras y posible profundización

Tal como se mencionó en el apartado del protocolo inicial, la principal falencia o debilidad del protocolo es su necesidad de un acuerdo inicial, este acuerdo inicial debe ser mantenido en secreto para garantizar la seguridad de la comunicación, si cayese en manos de un tercero, este tendría acceso total a la comunicación. Se podría desarrollar un método para transferir este acuerdo inicial de forma segura, sin la necesidad de darlo a conocer públicamente.

Al mismo tiempo, se podría analizar otras alternativas a la propuesta en el presente trabajo para realizar este acuerdo inicial, no es novedad que la computación avanza a una velocidad considerable, nuevas tecnologías, primitivas criptográficas y, sobre todo, ataques emergen cada día, y la resiliencia del protocolo de ser incorporado a otros permite que su implementación sea sencilla.

Por otra parte, otras líneas de mejora o profundización importantes serían el criptoanálisis en profundidad del protocolo, invertir mayor esfuerzo en analizar la seguridad computacional de los secretos generados, la entropía de los mismos utilizando varias veces la matriz inicial, el nivel de seguridad alcanzado por diferentes tamaños de matrices y vectores, una clave de una longitud determinada generada mediante el protocolo, ¿posee mayor seguridad que una clave de la misma longitud generada mediante otro protocolo?

6. Bibliografía

[1] A. J. Menezes, P. C. Van Oorschot, y S. A. Vanstone, *Handbook of applied cryptography*. Boca Raton: CRC Press, 1997.

[2] H. Scolnik, «Fundamentos matemáticos del método RSA». Universidad de Buenos Aires, mar-2011.

[3] J. M. Howie, *Fundamentals of semigroup theory*. Oxford : New York: Clarendon ; Oxford University Press, 1995.

[4] E. W. Weisstein, «Group Order», *Wolfram Mathworld*. [En línea]. Disponible en: http://mathworld.wolfram.com/GroupOrder.html. [Accedido: 25-nov-2019].

[5] E. W. Weisstein, «Cyclic Group», *Wolfram Mathworld*. [En línea]. Disponible en: http://mathworld.wolfram.com/GroupOrder.html. [Accedido: 25-nov-2019].

[6] G. Swamypillai, «When can an algorithm have square root(n) time complexity?», *StackOverflow*. [En línea]. Disponible en: https://stackoverflow.com/questions/33194931/when-can-an-algorithm-have-square-rootn-time-complexity. [Accedido: 20-nov-2019].

[7] D. Du y K.-I. Ko, *Theory of computational complexity*. New York: Wiley, 2000.

[8] A. Zygmund, *Trigonometric series*, 3rd ed. [Cambridge, UK ; New York: Cambridge University Press, 2002.

[9] T. H. Cormen y T. H. Cormen, Eds., *Introduction to algorithms*, 2nd ed. Cambridge, Mass: MIT Press, 2001.

[10] «Time Complexity», *Wikipedia*. [En línea]. Disponible en: https://en.wikipedia.org/wiki/Time_complexity#Table_of_common_time_complexities. [Accedido: 24-nov-2019].

[11] A. Ben-Israel y T. N. E. Greville, *Generalized inverses: theory and applications*, 2nd ed. New York: Springer, 2003.

[12] R. A. Horn y C. R. Johnson, *Matrix analysis*, Second edition, Corrected reprint. New York, NY: Cambridge University Press, 2017.

[13] R. Penrose, «A generalized inverse for matrices», *Math. Proc. Camb. Phil. Soc.*, vol. 51, n.º 3, pp. 406-413, jul. 1955, doi: 10.1017/S0305004100030401.

[14] D. Kahn, *The Codebreakers: the Comprehensive History of Secret Communication from Ancient Times to the Internet.* New York: Scribner, 1996.

[15] «Cryptanalysis». [En línea]. Disponible en: https://en.wikipedia.org/wiki/Cryptanalysis. [Accedido: 17-mar-2019].

[16] S. Malenkovich, «¿QUÉ ES UN ATAQUE MAN-IN-THE-MIDDLE?», 04-oct-2013. [En línea]. Disponible en: https://latam.kaspersky.com/blog/que-es-un-ataque-man-in-the-middle/469/. [Accedido: 25-mar-2019].

[17] S. W. Jung y S. Jung, «Secure Password Authentication for Distributed Computing», en *Computational Intelligence and Security*, vol. 4456, Y. Wang, Y. Cheung, y H. Liu, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 491-501.

[18] «Forward Secrecy», *Wikipedia*. [En línea]. Disponible en: https://en.wikipedia.org/wiki/Forward_secrecy. [Accedido: 10-abr-2019].

[19] W. Diffie y M. Hellman, «New directions in cryptography», *IEEE Transactions on Information Theory*, vol. 22, n.º 6, pp. 644-654, nov. 1976, doi: 10.1109/TIT.1976.1055638.
[20] A. Gardiner, *The Mathematical Olympiad handbook: an introduction to problem solving based on the first 32 British mathematical olympiads 1965-1996*. Oxford ; New York: Oxford University Press, 1997.
[21] V. Neale, *Closing the gap: the quest to understand prime numbers*. Oxford: Oxford University Press, 2017.
[22] F. Garzia, *Handbook of communications security*. Southampton ; Boston: WIT Press, 2013.
[23] M. Marlinspike, «The X3DH Key Agreement Protocol», *Signal.org*, 11-abr-2016. [En línea]. Disponible en: https://signal.org/docs/specifications/x3dh/. [Accedido: 06-jun-2019].
[24] M. Marlinspike, «The Double Ratchet Algorithm», *Signal.org*, 20-nov-2016. [En línea]. Disponible en: https://signal.org/docs/specifications/doubleratchet/. [Accedido: 06-jun-2019].
[25] M. K. Singh, «Public key cryptography with matrices», en *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, West Point, NY, USA, 2004, pp. 146-152, doi: 10.1109/IAW.2004.1437810.
[26] WSEAS International Conferences, *Proceedings of the WSEAS international conferences EED, MECHANICS, E-ACTIVITIES, ISP, CIMMACS, ELECTROSCIENCE 2007: Tenerife, Spain, december 14-16, 2007.* S. l.: WSEAS, 2007.
[27] E. Sakalauskas, «Enhanced Matrix Power Function for Cryptographic Primitive Construction», *Symmetry*, vol. 10, n.º 2, p. 43, feb. 2018, doi: 10.3390/sym10020043.
[28] A. Pal, Ed., *2nd International Conference on Computer, Communication, Control and Information Technology (C3IT-2012): Hooghly, West Bengal, India, 25 - 26 February 2012*. Red Hook, NY: Curran, 2013.
[29] Wikipedia, «Known-Plaintext Attack». [En línea]. Disponible en: https://en.wikipedia.org/wiki/Known-plaintext_attack. [Accedido: 12-feb-2019].
[30] Wikipedia, «Ciphertext-Only Attack». [En línea]. Disponible en: https://en.wikipedia.org/wiki/Ciphertext-only_attack. [Accedido: 12-feb-2019].
[31] A. A. ElHabshy, «Augmented Hill Cipher», *International Journal of Network Security*, vol. 21, n.º 5, pp. 812-818, sep. 2019, doi: 10.6633/IJNS.201909 21(5).13.
[32] P. Hecht, «A Post-Quantum Set of Compact Asymmetric Protocols using a General Linear Group», p. 7.
[33] Wikipedia, «Key Exchange». 12-oct-2019.
[34] «What is Key Management? a CISO Perspective», *Cryptomathic*. .
[35] K. J. Kim, Ed., *Information Science and Applications*, vol. 339. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.
[36] B. Murgante *et al.*, Eds., *Computational Science and Its Applications – ICCSA 2013*, vol. 7974. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.
[37] R. Thomas, «Safety in numbers», *Plus Magazine*. .
[38] L. Harn y H.-Y. Lin, «Authenticated key agreement without using one-way hash functions», *Electronics Letters*, vol. 37, n.º 10, p. 629, 2001.

[39] M. Dehkordi, «Identity – Based Multiple Key Agreement Scheme», *KSII Transactions on Internet and Information Systems*, 2011.
[40] F. Bao, R. H. Deng, y H. Zhu, «Variations of Diffie-Hellman Problem», en *Information and Communications Security*, vol. 2836, S. Qing, D. Gollmann, y J. Zhou, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 301-312.
[41] E. Barker, L. Chen, A. Roginsky, y A. Vassilev, «Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography», *NIST Special Publication*, p. 139.
[42] E. Barker, M. Smid, D. Branstad, y S. Chokhani, «A Framework for Designing Cryptographic Key Management Systems», National Institute of Standards and Technology, NIST SP 800-130, ago. 2013.
[43] A. Beletsky, A. Beletsky, y R. Kandyba, «Matrix Analogues of the Diffie-Hellman Protocol», p. 8.
[44] J. B. Nelson, «THE DIFFIE-HELLMAN KEY EXCHANGE IN MATRICES OVER A FIELD AND A RING», p. 30.
[45] M. Eftekhari, «A Diffie–Hellman key exchange protocol using matrices over noncommutative rings», *Groups - Complexity - Cryptology*, vol. 4, n.º 1, ene. 2012.
[46] L. Harn y H.-Y. Lin, «Authenticated key agreement without using one-way hash functions», *Electron. Lett.*, vol. 37, n.º 10, p. 629, 2001.
[47] C.-C. Lee, C.-T. Li, S.-T. Chiu, y Y.-M. Lai, «A new three-party-authenticated key agreement scheme based on chaotic maps without password table», *Nonlinear Dyn*, vol. 79, n.º 4, pp. 2485-2495, mar. 2015.
[48] M. Cao, D. Chen, Z. Yuan, Z. Qin, y C. Lou, «A lightweight key distribution scheme for secure D2D communication», en *2018 International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, Tangier, 2018, pp. 1-8.
[49] M. Bellare y P. Rogaway, «Introduction to Modern Cryptography», p. 283.
[50] D. Boneh, «The Decision Diffie-Hellman problem», en *Algorithmic Number Theory*, vol. 1423, J. P. Buhler, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 48-63.
[51] F. Bao, R. H. Deng, y H. Zhu, «Variations of Diffie-Hellman Problem», en *Information and Communications Security*, vol. 2836, S. Qing, D. Gollmann, y J. Zhou, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 301-312.
[52] J. H. Cheon, «Security Analysis of the Strong Diffie-Hellman Problem», en *Advances in Cryptology - EUROCRYPT 2006*, vol. 4004, S. Vaudenay, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1-11.
[53] D. Harkins y D. Carrel, «The Internet Key Exchange (IKE)», RFC Editor, RFC2409, nov. 1998.
[54] E. Rescorla, «Diffie-Hellman Key Agreement Method», RFC Editor, RFC2631, jun. 1999.
[55] H. H. Kilinc y T. Yanik, «A Survey of SIP Authentication and Key Agreement Schemes», *IEEE Commun. Surv. Tutorials*, vol. 16, n.º 2, pp. 1005-1023, 2014.
[56] I. Anshel, M. Anshel, B. Fisher, y D. Goldfeld, «New Key Agreement Protocols in Braid Group Cryptography», en *Topics in Cryptology — CT-RSA 2001*, vol. 2020, D. Naccache, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 13-27.
[57] T. Leighton y S. Micali, «Secret-Key Agreement without Public-Key Cryptography (Extended Abstract)», p. 24.

[58] B. Murgante *et al.*, Eds., *Computational science and its applications - ICCSA 2013: 13th international conference, Ho Chi Minh City, Vietnam, June 24-27, 2013; proceedings. Pt. 4: ...* Berlin: Springer, 2013.
[59] W. H. Press, Ed., *FORTRAN numerical recipes*, 2nd ed. Cambridge [England] ; New York: Cambridge University Press, 1996.
[60] V. N. Katsikis, D. Pappas, y A. Petralias, «An improved method for the computation of the Moore-Penrose inverse matrix», *arXiv:1102.1845 [math]*, feb. 2011.
[61] M. James, «The generalised inverse», *Math. Gaz.*, vol. 62, n.º 420, pp. 109-114, jun. 1978.
[62] K. J. Kim, *Information science and applications*. 2015.
[63] C.-K. Wu y E. Dawson, «Generalized inverses in public key cryptosystem design», p. 12.
[64] E. Sakalauskas, «Enhanced Matrix Power Function for Cryptographic Primitive Construction», *Symmetry*, vol. 10, n.º 2, p. 43, feb. 2018.
[65] K. A. Reddy, B. Vishnuvardhan, Madhuviswanatham, y A. V. N. Krishna, «A Modified Hill Cipher Based on Circulant Matrices», *Procedia Technology*, vol. 4, pp. 114-118, 2012.
[66] E. Oswald, M. Fischlin, y EUROCRYPT, Eds., *Advances in cryptology - EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015; proceedings. Pt. 2: ...* Heidelberg: Springer, 2015.
[67] R. Bhaskar, K. Chandrasekaran, S. Lokam, P. L. Montgomery, y Y. Yacobi, «AN OBSERVATION ABOUT VARIATIONS OF THE DIFFIE-HELLMAN ASSUMPTION», p. 7.
[68] L. Buttyán y I. Vajda, *Kriptográfia és alkalmazásai*. Budapest: Typotex, 2005.