Criptografía asimétrica ligera: seguridad de datos en dispositivos con pocos recursos computacionales

Miguel Morales-Sandoval
Cinvestav Tamaulipas
Cd. Victoria 87130, Mexico
miguel.morales@cinvestav.mx

Carlos Andres Lara-Nino*
Laboratoire Hubert Curien, CNRS
Saint-Etienne 42000, France
carlos.lara@univ-st-etienne.fr

Arturo Diaz-Perez
Cinvestav Guadalajara
Zapopan 45019, Mexico
adiaz@cinvestav.mx

RESUMEN

En este trabajo se presentan los resultados e impacto del trabajo de investigación desarrollado en el Cinvestav Unidad Tamaulipas sobre la creación de algoritmos y arquitecturas hardware para hacer frente al problema de garantizar servicios de seguridad de datos, como la confidencialidad, la integridad y la autenticación, en los datos que se producen, almacenan o transmiten entre dispositivos con pocos recursos computacionales. Un ejemplo de este tipo de dispositivos son los sensores inalámbricos, en el ámbito de las aplicaciones de e-salud, militares, o industriales. La estrategia de seguridad de datos se basa en el uso de algoritmos y protocolos criptográficos, los cuales pueden implementarse en software o en hardware. Este trabajo se enfoca en criptografía asimétrica, la cual está basada fuertemente en aritmética con operandos del orden de los 224 a los 2048 bits de longitud. Dada la naturaleza de los dispositivos que cuentan con pocas capacidades de cómputo y la alta demanda de recursos de los algoritmos de criptografía asimétrica, las soluciones de seguridad desarrolladas como parte del trabajo de investigación deben ser «ligeras», esto es, diseñadas con el objetivo de tener poco impacto en el dispositivo y en las aplicaciones donde se usan, afectando en la menor medida posible su desempeño al reducir el uso de los recursos de cómputo y el consumo de energía.

PALABRAS CLAVE

Cómputo móvil, Criptografía, Curvas elípticas, Internet de las Cosas, Seguridad ligera

Citar como:
Miguel Morales-Sandoval, Carlos Andres Lara-Nino, and Arturo Diaz-Perez. 2022. Criptografía asimétrica ligera: seguridad de datos en dispositivos con pocos recursos computacionales. En Miguel Morales Sandoval, José Gabriel Ramírez Torres y Javier Rubio Loyola (Eds.), *Ciencia e Ingeniería en Tecnologías Computacionales*, capítulo 2, 8 páginas. Cinvestav Unidad Tamaulipas, Ciudad Victoria, Mexico, ISBN: 978-607-9023-65-2.

1. INTRODUCCIÓN

La seguridad informática garantiza, entre otros, los servicios de autenticación, confidencialidad, integridad, y no repudio. Estos servicios son provistos principalmente por algoritmos y protocolos criptográficos. La Criptografía [1] es un área de estudio interdisciplinaria que involucra a las Ciencias Computacionales, las Matemáticas, la Teoría de Números, y la Complejidad Computacional.

1.1. Criptosistemas asimétricos

Un criptosistema es un conjunto de algoritmos que permiten implementar las operaciones de cifrado y descifrado de información. Cifrar es la acción de convertir una pieza de información que es entendible, por ejemplo, un mensaje de texto D, en algo totalmente incomprensible CT. Esto es, el cifrado transforma algo legible en algo ilegible. La transformación se realiza usando una *clave* (también conocida como *llave*). Los datos ilegibles pueden transformarse nuevamente en los datos originales (legibles) mediante el descifrado, usando nuevamente una clave. Estos procesos de transformación se muestran en la Fig. 2.

En un criptosistema asimétrico [2], la clave para cifrar es conocida por cualquier entidad (clave pública del receptor, K_{pub}), pero la clave para descifrar (clave privada k_{priv} del receptor) solo es conocida y usada por la entidad autorizada para acceder al contenido en su forma legible. Esto es, cualquiera puede enviar datos cifrados a una entidad particular usando la clave pública de dicha entidad receptora y solo el receptor puede descifrar la información usando su clave privada. Los criptosistemas son un conjunto de tres algoritmos:

1. Uno para generar una pareja de claves, pública y privada.
2. Otro para cifrar información usando una clave pública.
3. Y el tercero para descifrar información mediante la clave privada asociada a la clave pública usada durante el cifrado.

Los tres algoritmos son necesarios para completar un proceso donde se protege la información cifrándola y haciéndola incomprensible, excepto para las entidades autorizadas (las que poseen las claves privadas), que pueden descifrarla.

Los sistemas criptográficos asimétricos permiten implementar protocolos para el establecimiento de secretos compartidos, cifrado de datos bajo el concepto de sobres digitales, y autenticación mediante firmas digitales. Estos mecanismos han permitido proteger las comunicaciones al garantizar servicios de confidencialidad, integridad y autenticación robustos. Algunos ejemplos de aplicaciones donde actualmente se usan algoritmos y protocolos criptográficos para el intercambio seguro de datos incluyen la navegación en internet con el protocolo HTTPS, compras en línea usando el protocolo SET, y la transmisión de mensajes en dispositivos móviles (Whatsapp).

*Egresado del Programa de Doctorado en Ciencias en Ingeniería y Tecnologías Computacionales, en el Cinvestav Tamaulipas.

Jornadas de Divulgación - TopTamaulipas 2021, Noviembre 9–18, 2021, Ciudad Victoria, TAMPS

ISBN 978-607-9023-65-2.

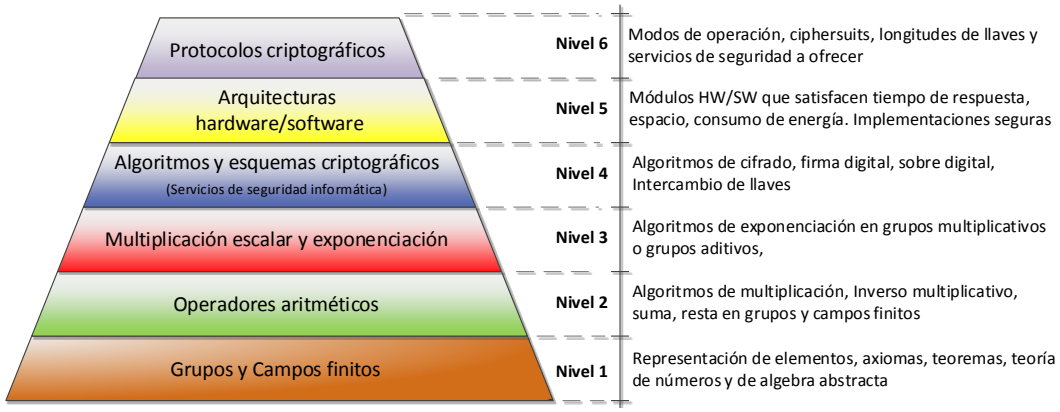

Figura 1: Distintos niveles de abstracción de mecanismos de seguridad informática basados en criptografía asimétrica.

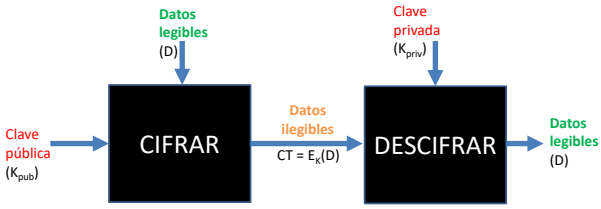

Figura 2: Procesos de cifrado (convertir texto legible a ilegible) y descifrado (convertir el texto ilegible del cifrado a texto legible nuevamente).

Los criptosistemas asimétricos se basan principalmente en la teoría de campos finitos y grupos (multiplicativos o aditivos), los cuales son estructuras algebraicas con propiedades especiales que permiten asegurar la seguridad de estos algoritmos. Así, los sistemas criptográficos basados en criptografía asimétrica quedan definidos en términos de operaciones matemáticas, por lo que las claves, los datos a cifrar, y los datos cifrados, son números o elementos de estas estructuras algebraicas.

En la Fig. 1 se muestra una perspectiva en capas de los diversos algoritmos que conforman la criptografía asimétrica. En nivel más bajo están las definiciones y tipos de representación de los elementos de los campos finitos y grupos. En la segunda capa se encuentran los algoritmos que permiten implementar operadores aritméticos en los campos finitos y en los grupos, como multiplicaciones, sumas, etc. En la capa 3 están las operaciones de grupo, fundamentales en la criptografía asimétrica. Estas operaciones son críticas porque por un lado definen la seguridad del sistema y por otro determinan qué tan eficiente es la solución de seguridad. Estas operaciones se usan, además de otras que son también aritméticas, en la capa 4 para implementar esquemas criptográficos, como algoritmos para cifrar información (que provee el servicio de confidencialidad). Los algoritmos criptográficos finalmente deben implementarse, o parte de ellos, en hardware o software. En la capa 5 se tienen las las estrategias de implementaciones más convenientes para que finalmente en la capa 6 se puedan desplegar los esquemas o protocolos que garantizan los servicios de seguridad en las aplicaciones de usuario.

La Fig. 1 permite dimensionar la complejidad computacional de los algoritmos criptográficos asimétricos, al requerir de diversos algoritmos, la mayoría está relacionado con operaciones aritméticas en las capas 1 y 2. Para éstas, actualmente se recomienda usar operandos con longitud entre 224 y 2048 bits, dependiendo del algoritmo criptográfico asimétrico empleado [3]. Estas longitudes se prevé que se incrementarán en los próximos años.

El cómputo intensivo que se realiza con los algoritmos en los niveles más bajos de la Fig. 1 afecta proporcionalmente el tiempo de ejecución de los algoritmos en los niveles más altos. Esto es, para que un usuario pueda contar con seguridad al momento de hacer una compra en línea usando su dispositivo móvil, y realizar dicho proceso en un tiempo razonable y con la menor penalización en el consumo de energía, se deben contar con algoritmos eficientes desde las capas más bajas.

1.2. Seguridad en los nuevos paradigmas de cómputo

Con la llegada del Cómputo Ubicuo y el Internet de las Cosas (IoT), considerado como la tercera era en la historia de la computación y las comunicaciones, la computación se realiza en cualquier parte y por cualquier dispositivo. Bajo este paradigma, los dispositivos de cómputo usados en estos entornos no son de altas prestaciones (como servidores o computadoras de escritorio), sino que generalmente son sistemas embebidos con recursos de cómputo restringidos (sensores inalámbricos, implantes médicos, tags RFID) que despliegan aplicaciones de alto impacto y con requerimientos indispensables de servicios de seguridad, por ejemplo, en escenarios de control de plantas nucleares, monitoreo médico de pacientes, sensado y actuación en automóviles, entre otras. El rango de aplicaciones donde se usan sistemas embebidos es muy amplio, tendencia que se visualiza a la alza a futuro [4].

La interacción e interconexión de los dispositivos embebidos en estas aplicaciones implica la adquisición, almacenamiento, procesamiento e intercambio de datos sensibles (por ejemplo, datos médicos en redes de área corporal), por lo que requieren tecnología de conectividad segura, no costosa, libre de fallas y confiable. Adicionalmente, bajo estas aplicaciones los sistemas embebidos

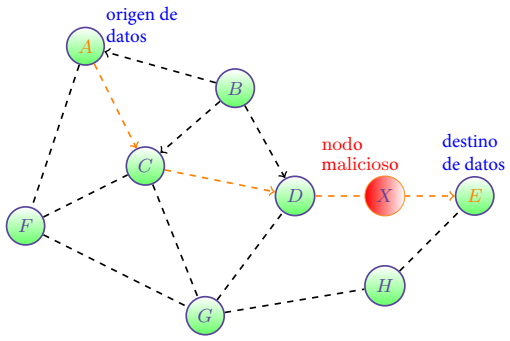

Figura 3: Modelo simple de una red de sensores. Los datos de origen deben protegerse de atacantes o nodos maliciosos presentes en la red.

operan en ambientes altamente hostiles e inseguros, lo que los hace susceptibles a un mayor número de amenazas.

Un modelo simple de los servicios de seguridad en una red de sensores se muestra en la Fig. 3. Los datos que se envían desde el nodo origen al nodo destino (por ejemplo, desde el teléfono donde se realiza una compra en línea hasta el servidor de la tienda) pueden ser interceptados, accedidos y usados de manera fraudulenta, mientras los datos atraviesan la red pública y sus nodos intermediarios que permiten la conexión entre el nodo emisor y en nodo receptor. Por ello, la seguridad es un requerimiento esencial en el IoT, donde diversos dispositivos de bajos recursos se conectan a Internet y sirven para realizar operaciones con datos sensibles. La criptografía asimétrica puede proveer los servicios de seguridad requeridos, tales como confidencialidad, integridad, y autenticación.

Sin embargo, los algoritmos y esquemas de criptografía asimétrica existentes difícilmente pueden ser usados en los nuevos modelos de computación ubicua, principalmente por la enorme disparidad entre el costo computacional que estos algoritmos criptográficos demandan y los bajos recursos de cómputo que se tienen disponibles en los sistemas embebidos para ejecutarlos.

A fin de proveer servicios de seguridad robustos en sistemas embebidos, como los basados en criptografía asimétrica, es necesario desarrollar nuevos algoritmos que se adapten a las restricciones de recursos de cómputo inherentes en sistemas embebidos (por ejemplo en nodos sensores). De igual forma, es necesario desarrollar las correspondientes arquitecturas de cómputo hardware/software que tengan el más bajo impacto en el dispositivo y en las aplicaciones. Esto es, se requiere contar con arquitecturas de hardware o software que exhiban el menor consumo energía, operen con pocos recursos de memoria y poder de cómputo, tengan la mejor eficiencia posible, y garanticen los niveles de seguridad y tiempos de respuesta requeridos. Sin embargo, comprometer todos estos requerimientos es un reto.

1.3. Objetivo y estructura de este documento

En este documento se describen los principales resultados en la línea de investigación de criptografía ligera, destacando el diseño de nuevos algoritmos o sus arquitecturas en hardware para realizar las operaciones más demandantes en el Nivel 2 y 3 en la Fig. 1, bajo un enfoque de cómputo ligero, donde se requieran pocos recursos computacionales para su ejecución sin sacrificar desempeño y nivel de seguridad. Estos nuevos operadores aritméticos son los bloques de construcción para proveer servicios de seguridad mediante esquemas de criptografía asimétrica (Nivel 4).

Con los nuevos algoritmos de operadores aritméticos en estructuras algebraicas se crearon arquitecturas de cómputo hardware/software ligeras (Nivel 5), que fueran lo más compactas posible y demandaran menor consumo de energía. Esto permitiría contar con protocolos de seguridad (Nivel 6) para proveer servicios de confidencialidad, autenticación e integridad robustos basados en criptografía asimétrica en aplicaciones de Cómputo Ubicuo e Internet de las cosas, donde las plataformas de cómputo subyacentes cuentan con capacidades de cómputo mucho menores que computadoras de escritorio o servidores.

La estructura del resto del documento es la siguiente: En la Sección 2 se describen las estructuras algebraicas donde se sustentan los algoritmos de criptografía simétrica, particularmente, los operadores aritméticos más relevantes en campos finitos. En la Sección 3 se describen los operadores en grupos que son los que se usan en construcciones criptográficas, descritas en la Sección 4. En la Sección 5 se presentan a manera de resumen las principales aportaciones que en el Cinvestav Tamaulipas se ha hecho al campo de la criptografía asimétrica ligera. Finalmente, la Sección 6 concluye este trabajo y describe las áreas de estudio en curso y futuras en relación con la criptografía asimétrica ligera.

2. CAMPOS FINITOS

Un campo es un conjunto de elementos que tiene definidas dos operaciones sobre sus elementos: + y ×. Ejemplos de éstos son los números enteros, los números reales y los números complejos. En criptografía, se requiere del uso de campos finitos, esto es, un conjunto de elementos fijo, pero para los cuales las reglas que conocemos sobre los campos siguen aplicando. Ejemplo de estas reglas son la existencia del elemento neutro por cada operador, la propiedad de cerradura, las propiedades conmutativa y asociativa, la existencia de inversos, etc.

En el caso de un campo finito, es difícil pensar cómo se puede mantener, por ejemplo, la propiedad de cerradura. Ésta establece que el resultado de operar dos elementos es nuevamente un elemento del campo. Considere por ejemplo el caso del conjunto $\{0, 1, 2, 3, 4\}$. Aunque el conjunto es finito (se pueden contar sus elementos), ¿cómo puede ser que por ejemplo, $4 + 3 = 2$ sea nuevamente un elemento del conjunto?. Es el mismo caso para $4 \times 3 = 2$. Lo anterior demuestra que las matemáticas como tradicionalmente las conocemos no aplican en el caso de los campos finitos.

El campo primo es uno de los campos finitos más usados en criptografía asimétrica. Se define como $Z_p = \{0, 1, \cdots, p-1\}$, siendo p un número primo, esto es, un número que solo puede ser dividido por 1 o por el mismo. Ejemplos de campos primos son Z_2, Z_3, Z_7, Z_{11}, etc. El primero es el conocido campo binario, definido como $Z_2 = \{0, 1\}$. Las operaciones definidas en Z_p son $(+ \mod p)$ y $(\times \mod p)$. El operador módulo se refiere al residuo de una división. Considere por ejemplo el conjunto $Z_3 = \{0, 1, 2\}$. Los siguientes son ejemplos las operaciones permitidas en Z_3.

- $(2 + 3) \mod 3 = 5 \mod 3 = 2$.

- $(2 \times 3) \mod 3 = 6 \mod 3 = 0$.

En el primer caso, los valores se suman y el resultado se divide entre $p = 3$. El valor del residuo es el resultado final de la operación. Lo mismo ocurre en el segundo caso, solo que ahora se realiza la multiplicación, el resultado se divide entre $p = 3$ y el residuo que es 0 en este caso es el resultado final de la operación. Cualquier residuo estará siempre en el rango $[0, p-1]$, lo que asegura que el campo sea finito, esto es, el resultado de alguna de las dos operaciones siempre será un elemento del mismo conjunto.

En Z_p, '0' es el elemento neutro del operador $(+ \mod p)$ mientras que '1' es el elemento neutro del operador $(\times \mod p)$. Por consiguiente, para cada elemento z en Z_p, existe un elemento $-z$ y z^{-1}, tal que $(z + (-z) \mod p) = 0$ y $(z \times z^{-1} \mod p)$. Llamamos a $-z$ y z^{-1} el inverso aditivo y multiplicativo, respectivamente, de z.

La existencia del elemento neutro y de inversos para cada operación hace a Z_p una estructura algebraica completa, donde se pueden definir operaciones matemáticas que son las construcciones básicas de los criptosistemas asimétricos.

De acuerdo con la Fig. 1, un reto a superar es implementar eficientemente los operadores $(+ \mod p)$ y $(\times \mod p)$, considerando que Z_p es relativamente grande. Actualmente, para criptosistemas como RSA, DH y DSA, p debe ser usado un número primo de unos 2,048 bits mientras que para ECC, p debe ser al menos de 224 bits. No obstante, la operación que realmente impacta en el desempeño de las capas superiores de la Fig. 1, es la multiplicación en grupos y en campos finitos.

2.1. Multiplicación en Z_p

Cualquier número entero (como los elementos en Z_p) pueden representarse en base binaria como se muestra en la Eq. 1. De esta forma, un número a puede verse como una cadena de m bits $(a_{m-1}, a_{m-2}, \ldots + a_0)$.

$$a = (a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \ldots + a_0) \tag{1}$$

Tomando la representación de la Eq. 1, la multiplicación $(a \times b)$ mód p puede realizarse mediante a sumas acumulativas de b consigo mismo, más corrimientos a la izquierda por la multiplicación de 2. Esto implica realizar m iteraciones, una para cada valor a_i. Una estrategia que se puede utilizar para mejorar el desempeño de estos algoritmos es hacer uso de recursos adicionales de hardware para procesar el operando sobre el que se itera de forma más rápida. Este tipo de algoritmos reciben el descriptivo de "basado en dígitos", que significa que en lugar de procesar un bit a la vez del operando a de acuerdo con la Eq. 1, se procesan d bits a la vez, es decir un dígito, lo que produce un cálculo más rápido pues el número de iteraciones baja de m a m/d.

La investigación sobre multiplicadores eficientes en Z_p es muy amplia, y el cuerpo del conocimiento en dicha área puede revisarse en distintas fuentes, por ejemplo, en [5].

2.2. Inversión

Otra operación de campo finito que generalmente es muy costosa es la división, la cual se calcula mediante el cálculo de un inverso multiplicativo (inversión) y una multiplicación, tal como se muestra en la Eq. 2.

$$\frac{a}{b} = ab^{-1}. \tag{2}$$

Por tanto, el primer paso es el cálculo del elemento inverso multiplicativo de b, esto es $b^{-1} \in Z_p$ tal que $b \times b^{-1} = 1$. En 1985, Wang demostró que todo elemento $b \in Z_p$ distinto de cero tiene un único inverso multiplicativo b^{-1}.

Aunque las operaciones para calcular un inverso multiplicativo no son sencillas, existen algoritmos que permiten resolverlas de forma eficiente. Una de ellas es usando el pequeño teorema de Fermat (FLT), el cual establece que dado un número primo p y b un entero que satisfacen $\gcd(b, p) = 1$, se tiene que $b^p \equiv b \mod p$. Entonces, para toda $b \neq 0 \in Z_p$, se puede calcular b^{-1} como b^{p-2}, esto es, mediante multiplicaciones acumulativas de b consigo mismo, $p - 2$ veces.

3. GRUPOS

Existen dos grupos principales que se usan en criptografía asimétrica: el grupo multiplicativo, denotado como Z_p^* y el grupo aditivo, el más común, denotado como $E(Z_p)$.

3.1. Grupo multiplicativo Z_p^*

La diferencia entre Z_p (un campo finito) y Z_p^* (un grupo) radica en que el primero tiene dos operaciones definidas, como se comentó anteriormente, y el segundo solo tiene una.

La única operación definida en Z_p^* es $(\times \mod p)$, y por tanto, solo existe un elemento neutro y el concepto de inverso multiplicativo: todo lo relacionado con $(+ \mod p)$ que se tiene en Z_p desaparece. Por ello, ahora $Z_p^* = \{1, 2, \cdots, p-1\}$. Un grupo debe cumplir varias propiedades, y como en el caso de los campos finitos, debe cumplir la propiedad de cerradura: toda operación en Z_p^* produce un elemento en Z_p^*.

Dado que solo se tiene una operación en Z_p^*, mediante ella y un elemento especial llamado generador, se puede obtener cualquier elemento de Z_p^*. Esto es, dado g en Z_p^*, g es un generador si para todo z en Z_p^*, existe un entero positivo a tal que $z = g^a \mod p$, siendo $g^a = (g \times g \times \cdots \times g) \mod p$. La operación $g^a \mod p$, llamada *exponenciación*, generalmente solo se indica como g^a, y como se puede ver, consiste en aplicar la operación de grupo $(\times \mod p)$ sobre el generador a veces. Dado que Z_p^* es finito, existe un número finito de valores a que se pueden usar para generar todos los elementos de Z_p^* a partir de g, que coincide con el número de elementos de Z_p^*, esto es, $a \in [1, p-1]$.

La exponenciación por tanto se puede implementar mediante un multiplicador en Z_p de manera iterativa. Sin embargo, esto no ocurre en la práctica, ya que sería totalmente ineficiente. Lo anterior debido a que en aplicaciones reales de criptografía asimétrica, Z_p^* es muy grande, siendo p de más de 2048 bits. Esto implicaría tomar un valor de a en el rango $[1, 2^{2048}]$, lo cual generaría un número de iteraciones en la multiplicación acumulativa extremadamente grande. Hasta la fecha, aún con la computadora más potente, es infactible por ejemplo, recorrer todos los valores de un contador de 112 bits; mucho más complicado es tratar de recorrer todos los valores de un contador de 2048 bits. Es por ello que se ha realizado un trabajo de investigación, y aún se sigue haciendo, para obtener

los mejores algoritmos y arquitecturas de cómputo que permitan realizar la operación de grupo, la exponenciación en este caso, de manera eficiente. Para referencia, el lector puede referirse a [6] para más detalles sobre algoritmos de exponenciación en Z_p^*.

3.2. Grupo aditivo $E(Z_p)$

En el caso de $E(Z_p)$, lo que se tiene es un conjunto de puntos (x, y) que satisfacen una ecuación $f(x, y)$, generalmente cúbica en la variable x. Tanto x como y son elementos de Z_p y $f(x, y)$ define un gráfico conocido como *curva elíptica*. Un punto $P_i(x_i, y_i)$ pertenecerá a $E(Z_p)$ si satisface la ecuación que define a la curva elíptica, por ejemplo, $y^2 = x^3 + ax + b$, con a y b en Z_p. En la Fig. 4 se muestra un ejemplo el conjunto de puntos (x, y) de la curva elíptica definida por la ecuación $y^2 = x^3 + x + 1$ sobre el el campo Z_{13}. Nótese en esta figura la simetría de la curva en relación con el eje horizontal.

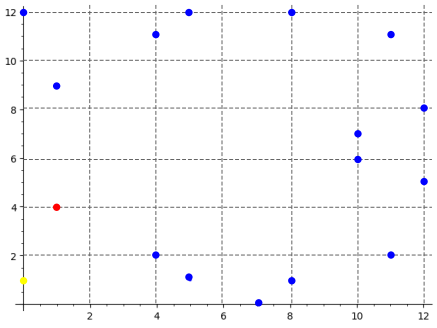

Figura 4: Ejemplo de una curva elíptica dada por la ecuación $y^2 = x^3 + x + 1 \pmod{13}$.

$E(Z_p)$ es un grupo aditivo, esto es, la operación de grupo es la suma de puntos. Esta operación la denotaremos como $\oplus$. Si en $E(Z_p)$ existe un elemento P, tal que todo elemento Q en $E(Z_p)$ puede obtenerse a partir de una sucesión de $a - 1$ operaciones de grupo sobre P, esto es $Q = P \oplus P \oplus \cdots \oplus P$ (sumar P a-veces), entonces se dice que P es un generador de $E(Z_p)$. La operación $P \oplus P \oplus \cdots \oplus P$ se representa como aP y se denomina *multiplicación escalar*: es una operación análoga a la exponenciación en el grupo multiplicativo. Si se asume que $E(Z_p)$ tiene un número finito de elementos, n, los posibles valores de a para generar todos los elementos de la curva elíptica a partir de P es el rango [1,n-1]. En la Fig. 5 se ejemplifica el concepto de generador. Para que $E(Z_p)$ sea un grupo aditivo, se introduce un punto especial llamado punto en el infinito, denotado por $\mathcal{O}$, que funge como el elemento neutro y permite la definición de inversos aditivos en la curva elíptica.

El orden de una curva elíptica es su número de elementos, n. Para curvas especiales dicho orden es un numero primo, y generalmente, son esas curvas elípticas las que se usan para construir esquemas criptográficos. Al igual que en la exponenciación, la multiplicación escalar es una operación muy demandante computacionalmente. Gran parte del trabajo de investigación en esquemas criptográficos basados en curvas elípticas consiste en el diseño e implementación

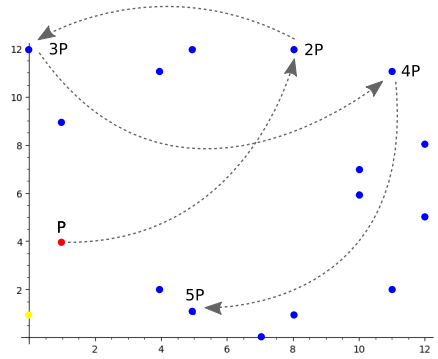

Figura 5: Comportamiento de un generador en un grupo de curva elíptica.

de algoritmos y arquitecturas hardware o software para la multiplicación escalar. En [7], el lector puede encontrar más detalles sobre el trabajo relacionado con la implementación eficiente de aP.

4. OPERACIONES CRIPTOGRÁFICAS

En 1976, Diffie y Hellman [2] propusieron un protocolo fundamentado en la teoría de grupos, particularmente, usando el grupo multiplicativo Z_p^*, para que dos entidades pudieran acordar un secreto compartido de manera segura. Este trabajo sentó las bases de la criptografía asimétrica. Más adelante, en 1987 Koblitz y Miller hicieron lo conducente pero con el grupo $E(Z_p)$ [8]. De manera generalizada, los sistemas criptográficos basados en grupos pueden ser definidos usando un grupo genérico G, pudiendo ser éste el multiplicativo o el aditivo; asumiendo que cada uno tiene una operación de grupo $\diamond$, un generador g, y de orden n.

4.1. Protocolo Diffie-Hellman (DH)

Este protocolo se creó para poder establecer un secreto compartido entre dos entidades a través de un canal de comunicación inseguro, a partir de los datos públicos (G, g, n). Este trabajo significó un hito en la historia de la criptografía e hizo merecedores a los inventores, Withfield Diffie y Martin Hellman, al premio Alan Turing, el máximo galardón que se otorga en el campo de las ciencias en computación.

El procedimiento consiste de los siguientes pasos entre dos entidades A y B.

1. A selecciona un número aleatorio en el intervalo $[1, n - 1]$. Sea este el número a. Este número solo lo conoce A.
2. B hace los mismo, selecciona un número aleatorio en el intervalo $[1, n - 1]$. Sea este el número b. Este número solo lo conoce B.
3. A realiza la operación $z_a = g^a$ y envía z_a a B.
4. B realiza la operación $z_b = g^b$ y envía z_b a A.
5. Cuando A recibe z_b, realiza la operación $(z_b)^a$.
6. Cuando B recibe z_a, realiza la operación $(z_a)^b$.

Los pasos 5 y 6 son tales que se cumple la Eq. 3.

$$s = \underbrace{(z_a)^b}_{\text{Calculado por } B} = (g^a)^b = (g^{ab}) = (g^b)^a = \underbrace{(z_b)^a}_{\text{Calculado por } A} \quad (3)$$

El valor s sirve como valor secreto, que nadie más que A y B conocen y que pueden usar para intercambiar información de forma segura, por ejemplo, cifrando los datos con dicho secreto compartido. El cifrado, a manera de analogía, funciona como un candado convencional, solo el que tiene la llave puede abrir el candado y acceder a lo que dicho candado protege. La clave en este caso será s, que solo la poseen A y B.

4.2. Seguridad de la criptografía asimétrica

En el caso del protocolo DH, un atacante podría intentar resolver el siguiente problema: Dados $\{g, g^a, g^b\}$, calcular $g^{ab} = s$. A este problema se le llama el *problema computacional Diffie Hellman* (PCDH). Se ha demostrado que la única forma de resolver este problema es resolviendo el problema del logaritmo discreto (PLD).

El PLD se enuncia de la siguiente forma: Dados $\{G, g, n\}$, siendo G un grupo (aditivo o multiplicativo), con generador g y de orden n, y dado cualquier elemento z en G, hallar el número entero a tal que $z = g^a$. Si pudiera resolverse el PLD, se podrían obtener a y b a partir de z_a y z_b; con estos valores el atacante podría calcular g^{ab} y por tanto obtener el secreto s. Sin embargo, hasta el momento de escribir este artículo, no existe un algoritmo eficiente que permita resolver el PLD, de ahí que el protocolo DH se considera seguro. Para garantizar que el PLD es infactible de abordarse, aún con el poder de cómputo más reciente, es necesario que G sea suficientemente grande. Para protocolos DH basados en grupos multiplicativos (Z_p^*), el tamaño del grupo debe estar en el orden de 2^{2048}, es decir, p debe ser un número primo de al menos 2048 bits. Para el caso de grupos aditivos $E(Z_p)$, el tamaño n del grupo debe ser del orden de 2^{224}, al menos.

4.3. Cifrado asimétrico

En el cifrado asimétrico, cada entidad cuenta con dos llaves relacionadas entre sí: una llave privada k y una llave pública $f(k)$. La primera no debe compartirse con nadie; la segunda se obtiene a partir de la primera mediante una función f, y cualquiera puede tener acceso a ella. La seguridad en este caso radica en que no será posible obtener k a partir de $f(k)$. Lo anterior se consigue usando una función f que es fácil de calcular, pero su operación inversa es muy difícil. Esto debe ser así para que en efecto, dado $f(k)$ no se pueda realizar $f^{-1}(f(k)) = k$.

El trabajo de Diffie y Hellman en la propuesta de su protocolo de establecimiento de llaves sentó las bases para crear cifradores asimétricos. Usando f como una exponenciación en Z_p^* (lo mismo y análogo para la multiplicación escalar en $E(Z_p)$), y debido a la dificultad del PLD asociada a cada operación (inversa), las llaves privada y pública pueden generarse por una entidad A de la siguiente forma:

1. Para Z_p^* con generador g y orden $n = p$:
   - Seleccionar a en [1, p-1]
   - Calcular g^a
   - La llave privada es a; la llave pública es g^a.
2. Para $E(Z_p)$ con generador P y orden n:
   - Seleccionar a en [1, n-1]
   - Calcular aP
   - La llave privada es a; la llave pública es aP.

Si el grupo multiplicativo o aditivo es suficientemente grande, el PLD es infactible de resolverse y por tanto, no podría obtenerse la llave privada a partir de la llave pública.

En el cifrado asimétrico, intervienen dos entidades: el que cifra (entidad A) y el que descifra (Entidad B). Para realizar las operaciones de cifrado, la entidad de interés es B, ya que será la única autorizada para descifrar y acceder a una pieza de información D. Primero B deberá generar sus llaves $\{k_b, f(k_B)\}$. Una entidad A realiza el cifrado asimétrico de una pieza de información D para enviarla a B de la siguiente forma:

1. Obtiene la llave pública de B.
2. Cifra D usando $f(k_B)$ para obtener los datos cifrados CT_B. Esto se denota como $CT_B = CIFRAR(D, f(k_B))$.
3. Envía CT_B a B.

Por su parte, B y solo B podrá acceder a D usando su llave privada k_B. Esta es la razón por la que las llaves deben estar relacionadas. El descifrado se realiza de la siguiente forma:

1. B accede a su clave privada k_B, de manera segura. Por ejemplo, almacenada en un dispositivo que solo B puede acceder o que está protegido por un mecanismo robusto.
2. Descifra CT_B usando k_B para obtener D. Esto se denota como $D = DESCIFRAR(CT_B, K_B) = DESCIFRAR(CIFRAR(D, f(k_B)), k_B) = D$.

La implementación de las funciones *CIFRAR* y *DESCIFRAR* involucran diversas operaciones matemáticas, definidas ya sea Z_p, en Z_p^*, o en $E(Z_p)$, que dominan el tiempo de ejecución de dichas operaciones. Una descripción detallada de los algoritmos de cifrado y descifrado asimétricos más populares puede consultarse en [1].

4.4. Firmas digitales

El cifrado asimétrico descrito en la sección anterior garantiza el servicio de confidencialidad entre A y B, pero no la autenticación. Esto es, no es posible garantizar que los datos provienen de A, ya que no se ha usado ningún elemento vinculante de A en el proceso de seguridad de los datos. Para ello, se usa el concepto de firma digital, que es el análogo a las firmas manuscritas. Una firma digital se implementa de manera muy simple en el concepto. La realización de un algoritmo de firma digital, sin embargo, requiere más elementos de seguridad como son las funciones de huella digital (funciones hash) y los certificados digitales. Estos últimos son componentes que asocian una llave pública con la identidad de algo o alguien. Por ejemplo, una llave pública que es un número en Z_p^* o un punto (x, y) en $E(Z_p)$ debe asociarse a un nombre, a un RFC, a una CURP o a algún elemento que indique quién es el propietario de esa llave pública.

En este caso, la operación de generación de firma digital corresponde al emisor y no al receptor del mensaje. Por ello, el uso de llaves se invierte: ahora el emisor A cifra (o más bien dicho firma) una pieza de información D usando su llave privada k_A, y la entidad B puede descifrar, o más bien, verificar si la firma es válida usando la llave pública de A, $f(k_A)$. Si la firma resulta válida, se autentica

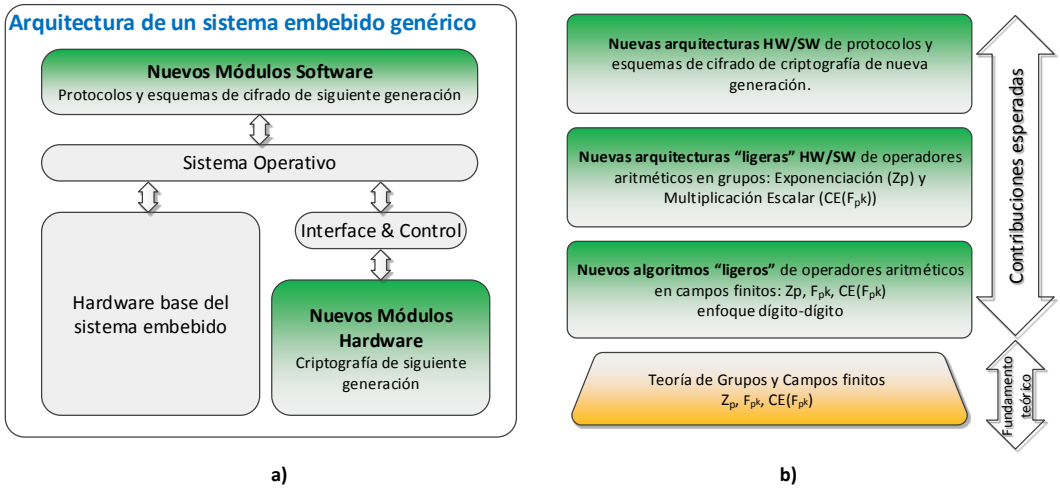

Figura 6: a) Sistema embebido con componentes Hardware/Software. b) Módulos de operadores, algoritmos, esquemas y arquitecturas de criptografía asimétrica ligera de siguiente generación que pueden incorporarse en a).

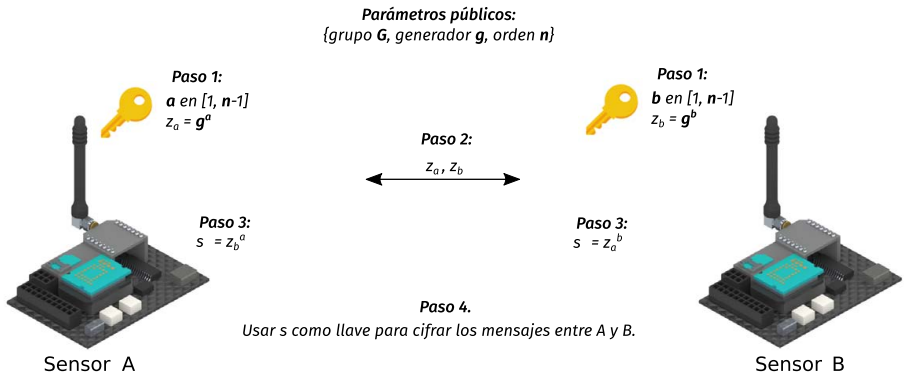

Figura 7: Establecimiento de llaves entre dos nodos sensores (sistemas embebidos). Note que el par de llaves $\{a, z_a\}$ o $\{b, z_b\}$ puede también ser usado en el cifrado asimétrico o la generación de firmas digitales.

el origen e integridad de D. Si en el trascurso del envío de D y de su firma a B se modifica D ya sea intencional o accidentalmente, o si la verificación de la firma se realiza con otra llave pública que no es la del emisor, la validación de la firma fallará.

Los pasos siguientes resumen las operaciones de firmar y verificar una firma digital.

A firma el mensaje D de la siguiente forma:

1. Calcula el valor hash de D. $h = HASH(D)$.
2. Obtiene la firma $\sigma_{\{A,D\}} = CIFRAR(h, k_A)$.
3. Transmite $\{\sigma_{\{A,D\}}, D\}$.

B verifica la firma $\sigma_{\{A,D\}}$ del mensaje D de la siguiente forma:

1. Obtiene la clave pública $f(k_A)$ de A.
2. Calcula nuevamente el valor hash de D. $h' = HASH(D)$.
3. Descifra el valor hash original $h = DESCIFRAR(\sigma_{\{A,D\}}, f(k_A))$.
4. La firma es válida si y solo si $h = h'$.

5. PRINCIPALES APORTACIONES

Desarrollar nuevos algoritmos, protocolos y esquemas de seguridad basados en criptografía asimétrica, la cual está basada en operadores aritméticos en los grupos en Z_p^*, y $E(Z_p)$, así como en campos finitos Z_p, para dispositivos con pocos recursos computacionales es un reto. Por un lado, los algoritmos de criptografía asimétrica son costosos en términos de operaciones de cómputo; por el otro, se tienen pocos recursos para desplegarlos.

En la investigación desarrollada, los algoritmos y protocolos creados se conciben como módulos hardware/software que pueden integrarse a la arquitectura computacional de un sistema embebido, tal como se muestra en la Figura 6. El trabajo que se ha realizado en el Cinvestav Tamaulipas sobre criptografía asimétrica ligera comprende:

1. Algoritmos y arquitecturas hardware de operadores aritméticos en campos finitos Z_p (Niveles 2 y 5 de la Fig. 1),

usando nuevos enfoques de procesamiento de datos y particularmente sobre el operador de multiplicación. Destacan los multiplicadores digito-serial y dígito-dígito. Estos multiplicadores se han reportado en foros internacionales como [5, 9, 10].

2. Algoritmos y arquitecturas hardware de operadores aritméticos en grupos multiplicativos Z_p^* y aditivos $E(Z_p)$ (Niveles 3 y 5 de la Fig. 1), usando nuevos enfoques de procesamiento de datos. La operación de grupo se implementa reutilizando el operador de multiplicación, por ejemplo, en [11]. Un campo finito de particular interés para ECC en hardware es el campo de extensión Z_{2^m}. Sobre este caso, se han reportado arquitecturas hardware compactas y con reducción de energía, por ejemplo en [12, 13].
3. Arquitecturas hardware/software de protocolos de criptografía asimétrica (Niveles 5 y 4 de la Fig. 1), en particular para establecimiento de secretos compartidos en redes de sistemas embebidos. Un ejemplo es el protocolo ligero para establecimiento de claves en redes de sensors [14, 15] e implementaciones hardware/software para el establecimiento de secretos compartidos, por ejemplo, éstas se han reportado en [11, 15].

Los desarrollos realizados se han enfocado en las capas bajas de la Fig.1 , dado que son las que tienen impacto en las aplicaciones de las capas superiores. Para el desarrollo de los algoritmos, protocolos y arquitecturas hardware/software, se han usado FPGAs, que son dispositivos programables que pueden implementar prácticamente cualquier algoritmo en hardware, explotando el paralelismo y obteniendo diseños que pueden ser integrados con un microprocesador e incluso con otros módulos de entrada y salida, como una antena, una conexión a internet por cable o salida de video. De esta manera, los FPGAs permiten desplegar soluciones como la descrita en la Fig. 6).

La Fig. 7 muestra un ejemplo de cómo dos dispositivos emulados con tarjetas FPGA pueden ejecutar un protocolo DH para establecer primero un secreto compartido, o pueden ejecutar el algoritmos de generación de llaves pública y privada. Con estas llaves, se pueden implementar los procesos de cifrado/descifrado asimétrico y de autenticación con firmas digitales.

6. CONCLUSIONES Y TRABAJO FUTURO

La criptografía asimétrica, cuya seguridad está basada en el PLD, ha sido usada de manera exitosa hasta nuestros días. Por tanto, los protocolos de intercambio de llaves, de sobres digitales o de firmas digitales, se mantienen seguros. Sin embargo, la potencial aparición de una computadora cuántica ha motivado el desarrollo de una nueva línea de investigación en criptografía, llamada *criptografía postcuántica* (PQC) [16]. Lo anterior debido a que se ha demostrado que el PLD podría ser fácilmente resuelto con ayuda de estos equipos, llevando a la inutilización y vulneración de los esquemas criptográficos asimétricos conocidos hasta ahora.

En el Cinvestav Tamaulipas se está realizando investigación sobre PQC [17], en el entorno ligero, ya que en caso de que las computadoras cuánticas se desarrollen en los próximos años, será necesario contar con nuevos algoritmos PQC y sus arquitecturas hardware/software que puedan usarse de manera eficiente en entornos de cómputo restrictivo, como en los sistemas embebidos o el IoT, los cuales son el modelo de cómputo a prevalecer en el corto y mediano plazo.

AGRADECIMIENTOS

Gran parte del trabajo de investigación sobre criptografía asimétrica ligera ha sido realizado con el apoyo del Fondo Sectorial de Investigación para la Educación, Ciencia Básica SEP-CONACyT, proyecto número 281565, titulado *Desarrollo de nuevos algoritmos y arquitecturas de cómputo para criptografía ligera*. El proyecto estuvo a cargo del Dr. Miguel Morales Sandoval.

REFERENCIAS

[1] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 2001.
[2] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.
[3] Elaine Barker. Recommendation for key management. Standard, National Institute of Standards and Technology, Gaithersburg, MD, 5 2020.
[4] Rishika Mehta, Jyoti Sahni, and Kavita Khanna. Internet of things: Vision, applications and challenges. *Procedia Computer Science*, 132:1263–1269, 2018. International Conference on Computational Intelligence and Data Science.
[5] Miguel Morales-Sandoval and Arturo Diaz-Perez. A compact FPGA-based montgomery multiplier over prime fields. In *Proceedings of the 23rd ACM international conference on Great lakes symposium on VLSI*, GLSVLSI '13, pages 245–250, New York, NY, USA, 2013.
[6] Christof Paar. Exponentiation algorithms. In Henk C. A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security*, pages 434–436. Springer US, Boston, MA, 2011.
[7] Matheus F. De Oliveira and Marco Aurélio Henriques. A secure and efficient method for scalar multiplication on supersingular elliptic curves over binary fields. In *Proceedings of the 16th International Conference on Information Security - Volume 7807*, ISC 2013, page 407–416, Berlin, Heidelberg, 2013. Springer-Verlag.
[8] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177): 203–209, January 1987.
[9] Miguel Morales-Sandoval and Arturo Diaz-Perez. Scalable GF(p) Montgomery multiplier based on a digit-digit computation approach. *IET Computers and Digital Techniques*, 10(3):102–109, 2016. doi: 10.1049/iet-cdt.2015.0055.
[10] Luis Rodriguez-Flores, Miguel Morales-Sandoval, Rene Cumplido, Claudia Feregrino-Uribe, and Ignacio Algredo-Badillo. Compact FPGA hardware architecture for public key encryption in embedded devices. *PLoS One*, 13(1):1 – 21, 2018. doi: 10.1371/journal.pone.0190939.
[11] Miguel Morales-Sandoval, Luis Armando Rodriguez-Flores, Rene Cumplido, Jose Juan Garcia-Hernandez, Claudia Feregrino, and Ignacio Algredo-Badillo. A compact FPGA-based accelerator for curve-based cryptography in Wireless Sensor Networks. *Journal of Sensors, special issue: Recent Advances in Security and Privacy for Wireless Sensor Networks*, 2021:1–13, 2021. doi: 10.1155/2021/8860413.
[12] Carlos Andres Lara-Nino, Arturo Diaz-Perez, and Miguel Morales-Sandoval. Energy/area-efficient scalar multiplication with binary Edwards curves for the IoT. *Sensors, special issue "Privacy and Security for Resource Constrained IoT Devices and Networks"*, 19(3):1 – 35, 2019. doi: 10.3390/s19030720.
[13] Carlos Andres Lara-Nino, Arturo Diaz-Perez, and Miguel Morales-Sandoval. Lightweight elliptic curve cryptography accelerator for internet of things applications. *Ad Hoc Networks*, 103:102159, 2020. doi: 10.1016/j.adhoc.2020.102159.
[14] Carlos Andres Lara-Nino, Miguel Morales-Sandoval, and Arturo Diaz-Perez. Lightweight key establishment for WSNs. In *2019 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 1–8, University of Victoria, Canada, August 2019. doi: 10.1109/PACRIM47961.2019.8985101.
[15] Carlos Andres Lara-Nino, Arturo Diaz-Perez, and Miguel Morales-Sandoval. Key-Establishment Protocols for Constrained Cyber-Physical Systems. In Ali Ismail Awad, Steven Furnell, Marcin Paprzycki, and Sudhir Kumar Sharma, editors, *Security in Cyber-Physical Systems: Foundations and Applications*, pages 39–65. Springer International Publishing, Cham, 2021. ISBN 978-3-030-67361-1. doi: 10.1007/978-3-030-67361-1_2.
[16] Daniel J. Bernstein and Tanja Lange. Post-quantum cryptography. *Nat.*, 549 (7671):188–194, 2017. doi: 10.1038/nature23461.
[17] Carlos Andres Lara-Nino, Arturo Diaz-Perez, and Miguel Morales-Sandoval. Post-Quantum Cryptography on Wireless Sensor Networks: Challenges and Opportunities. In *Integration of WSNs into Internet of Things: A Security Perspective*, pages 81–99. CRC Press, 1st edition, 2021. ISBN 978-0-3676-2019-6. doi: 10.1201/9781003107521.