Teoría de Números y Criptografía

F. J. Lobillo

2020/2021

Parte II

Criptografía y Curvas Elípticas

Índice general

Capítulo 2

Criptografía simétrica

2.1 Cifrado y secreto

La primera tarea de la criptografía es proporcionar confidencialidad mediante métodos de cifrado.

Dados conjuntos

- $\mathcal{M}$ el conjunto se los mensajes, textos en claro o *plaintexts*,
- $\mathcal{C}$ el conjunto de los criptogramas o *cyphertexts*,
- $\mathcal{K} \subseteq \mathcal{K}_p \times \mathcal{K}_s$ el espacio de claves o *key space*,

un cripsosistema viene definido por dos aplicaciones

$$\begin{aligned} E &: \mathcal{K}_p \times \mathcal{M} \to \mathcal{C}, \\ D &: \mathcal{K}_s \times \mathcal{C} \to \mathcal{M}, \end{aligned}$$

tales que para cualquier clave $k_p \in \mathcal{K}_p$, existe una clave $k_s \in \mathcal{K}_s$ de manera que dado cualquier mensaje $m \in \mathcal{M}$,

$$D(k_s, E(k_p, m)) = m. \tag{2.1}$$

Fijadas claves $k_p \in \mathcal{K}_p$ y su correspondiente $k_s \in \mathcal{K}_s$, se suele utilizar la notación

$$E_{k_p} : \mathcal{M} \to \mathcal{C}, \left[E_{k_p}(m) = E(k_p, m)\right]$$
$$D_{k_s} : \mathcal{C} \to \mathcal{M}, [D_{k_s}(c) = D(k_s, c)]$$

para las funciones de cifrado y descifrado. La propiedad (2.1) se transforma en

$$D_{k_s}(E_{k_p}(m)) = m.$$

En la criptografía clásica, también llamada simétrica, se tiene que $\mathcal{K}_p = \mathcal{K}_s$ y $k_s = k_p = k \in \mathcal{K}$, o al menos hay métodos eficientes para conocer k_s a partir de k_p y viceversa. En la criptografía asimétrica, también llamada de clave pública, no se conocen métodos eficientes para conocer k_s a partir de k_p.

2.2 Objetivos de la criptografía

Confidencialidad La información solo puede ser accesible por las entidades autorizadas.

Integridad La información no ha sido alterada en el envío.

Autenticidad La información proviene de quien afirma haberla enviado.

No repudio El emisario de una información no puede a posteriori negar que ha realizado tal envío.

2.3 Ataques

El criptoanálisis es la disciplina encargada de tratar de averiguar el mensaje o la clave empleada. Desde el punto de vista de la seguridad se parte del conocido como *Principio de Kerckhoffs*, que establece que el adversario conoce todos los detalles del criptosistema excepto la clave empleada. Es decir, la seguridad debe recaer en el secreto de la clave empleada para descifrar.

Los posibles ataques se clasifican como sigue:

Criptograma El adversario conoce el criptograma. Esta situación siempre se da.

Mensaje conocido El atacante conoce parejas mensaje/criptograma cifradas con una misma clave.

Mensaje escogido El atacante puede generar criptogramas para mensajes de su elección. Una vez obtenidas dichas parejas, trata de averiguar el mensaje correspondiente a un criptograma desconocido.

Mensaje escogido-adaptativo El atacante no solo puede generar parejas mensaje/criptograma a su elección, sino que puede hacerlo tantas veces como quiera realizando los análisis que considere oportunos entre medias.

Criptograma escogido y escogido-adaptativo Similar a los anteriores pero partiendo del criptograma, es decir, tiene acceso a descifrar los criptogramas que desee, inicialmente o a lo largo del proceso. Evidentemente este ataque busca la clave.

2.4 Seguridad probable

El primer intento de formalizar el concepto de seguridad de un criptosistema se debe a C. E. Shannon. Define lo que se conoce como un cifrado *perfectamente secreto*, aquél que resiste cualquier ataque al criptograma. Es decir, el criptograma no aporta ninguna información sobre el mensaje, incluso bajo la hipótesis de que el atacante dispone de capacidad de cálculo y tiempo ilimitados. El cifrado de Vernam es el más conocido entre los perfectamente secretos. Si asumimos que nuestro mensaje es una cadena de bits, que podemos identificar con $\mathbb{Z}_2$, la clave va a ser una cadena aleatoria del mismo tamaño, y el cifrado consiste en realizar la suma módulo 2 de cada bit del mensaje con cada bit de la clave[1]. El descifrado consiste en realizar la suma del criptograma con la misma clave.

Para utilizar este cifrado hay que tener en cuenta que transferir la clave cuesta el mismo trabajo que transmitir el mensaje. Si hay un canal seguro para la clave, se puede emplear el mismo canal para el mensaje. Suelen distribuirse las claves previamente, que deben ser almacenadas. Además, cada clave debe usarse una sola vez, ya que

$$\mathtt{c} = \mathtt{m} \oplus \mathtt{k} \text{ y } \mathtt{c}' = \mathtt{m}' \oplus \mathtt{k} \Rightarrow \mathtt{c} \oplus \mathtt{c}' = \mathtt{m} \oplus \mathtt{m}',$$

por lo que perdemos la aleatoriedad.

La dificultad de usar cifrados perfectamente secretos ha conducido a analizar la seguridad desde puntos de vista mas laxos. Concretamente

[1] La suma en $\mathbb{Z}_2 = \mathbb{B}$ suele representarse en el mundo de la informática mediante el símbolo $\oplus$ o mediante `XOR`.

se abandona la hipótesis de que el atacante tiene capacidad de cálculo ilimitada. Se pasa a considerar criptosistemas que resisten ataques *factibles*. Un ataque factible es aquel que puede realizarse mediante un algoritmo eficiente. El punto de vista más aceptado establece que los algoritmos polinomiales son eficientes, mientras que los no polinomiales no lo son.

La Teoría de Números viene siendo empleada desde los años 70 del siglo pasado como fuente de problemas que mezclan algoritmos eficientes con otros que no lo son.

2.5 Criptografía simétrica

Un criptosistema simétrico, como hemos dicho antes, viene determinado por dos aplicaciones

$$E : \mathcal{K} \times \mathcal{M} \to \mathcal{C},$$
$$D : \mathcal{K} \times \mathcal{C} \to \mathcal{M},$$

tales que para cualquier clave $k \in \mathcal{K}$ y cualquier mensaje $m \in \mathcal{M}$,

$$D(k, E(k, m)) = m. \quad (2.2)$$

Por supuesto, fijada $k \in \mathcal{K}$ se suele utilizar la notación

$$E_k : \mathcal{M} \to \mathcal{C},$$
$$D_k : \mathcal{C} \to \mathcal{M},$$

para las funciones de cifrado y descifrado. La propiedad (2.2) se transforma en

$$D_k(E_k(m)) = m.$$

La criptografía simétrica moderna es criptografía digital. Los conjuntos $\mathcal{M}$, $\mathcal{C}$ y $\mathcal{K}$ son cadenas de bits, los primeros $\mathcal{M} = \mathcal{C} = \mathbb{B}^*$ y el espacio de claves $\mathcal{K} = \mathbb{B}^K$.

2.6 Cifrados de flujo

Sean $\mathcal{M} = \mathcal{C} = \mathbb{B}^*$ y $\mathcal{K} = \mathbb{B}^K$ para cierto $K \in \mathbb{N}$. Sea

$$f : \mathcal{K} \times \mathcal{M} \to \mathbb{B}^*$$

una aplicación tal que $f(k, m)_i = f(k, m_{[0,i-1]})_i$ donde $m_{[0,i-1]} = m_0 \cdots m_{i-1}$. El cifrado de flujo asociado a f es

$$E_k(m) = (m_i \oplus f(k, m_{[0,i-1]})_i)_{i \geq 0}$$

El cifrado de Vernam es un cifrado de flujo en el que f no depende del mensaje y genera una sucesión aleatoria.

Los más famosos son

- RC4
- A5/1, A5/2
- Portfolio eSTREAM
- Cualquier cifrado de bloque en modo feedback.

Cifrados de flujo síncronos. La sucesión criptográfica se genera independientemente del mensaje y del criptograma, es decir,

$$f : \mathcal{K} \to \mathbb{B}^*$$

Satisface las ecuaciones:

$$\begin{aligned} \sigma_{i+1} &= g(\sigma_i, k), \\ z_i &= f(\sigma_i, k), \\ c_i &= z_i \oplus m_i, \end{aligned}$$

donde σ_0 es el estado inicial, k es la clave, f es la función *siguiente estado*, f produce la sucesión criptográfica z_i que es sumada (XOR) con mensaje m_i para generar el criptograma c_i.

Cifrados de flujo autosincronizables. La sucesión criptográfica se genera a partir de la clave y de una cantidad fija de bits en el criptograma, matemáticamente

$$\begin{aligned} \sigma_i &= (c_{i-t}, c_{i-t+1}, \ldots, c_{i-1}), \\ z_i &= f(\sigma_i, k) \\ c_i &= z_i \oplus m_i, \end{aligned}$$

donde $\sigma_0 = (c_{-t}, c_{-t+1}, \ldots, c_{-1})$ es el estado inicial, k es la clave, f es la función que genera la sucesión criptográfica z_i que es sumada (XOR) con el mensaje m_i para generar el criptograma c_i.

2.7 Cifrados de bloque

Son criptosistemas en los que mensajes, criptogramas y claves están limitados a cadenas de una longitud fija, formalmente,

$$\mathsf{E} : \mathbb{B}^{\mathsf{K}} \times \mathbb{B}^{\mathsf{N}} \to \mathbb{B}^{\mathsf{N}},$$
$$\mathsf{D} : \mathbb{B}^{\mathsf{K}} \times \mathbb{B}^{\mathsf{N}} \to \mathbb{B}^{\mathsf{N}},$$

o para cada clave $\mathsf{k} \in \mathbb{B}^{\mathsf{K}}$,

$$\mathsf{E}_{\mathsf{k}} : \mathbb{B}^{\mathsf{N}} \to \mathbb{B}^{\mathsf{N}},$$
$$\mathsf{D}_{\mathsf{k}} : \mathbb{B}^{\mathsf{N}} \to \mathbb{B}^{\mathsf{N}}.$$

N se conoce como el tamaño del bloque y a K como el tamaño de la clave.

Algunos de los más conocidos son

- DES
- IDEA
- Blowfish
- AES (Rijndael)

2.7.1 *Modos de operación*

Cómo se extiende E de $\mathbb{B}^{\mathsf{N}}$ a $\mathbb{B}^*$ es competencia de los modos de operación. Estos modos dependen del tamaño del bloque, no de la clave.

Electronic CodeBook.

2.1 (ECBencrypt). **Input:** $m \in \mathbb{B}^*$
Output: $c \in \mathbb{B}^*$
Dividimos $m = m_{[1]} \cdots m_{[l]}$ con $m_{[i]} \in \mathbb{B}^N$.
for $i = 1, \ldots, l$ **do**
$c_{[i]} = E_k(m_{[i]})$.
return $c_{[1]} \cdots c_{[l]}$

2.2 (ECBdecrypt). **Input:** $c \in \mathbb{B}^*$
Output: $m \in \mathbb{B}^*$
Dividimos $c = c_{[1]} \cdots c_{[l]}$ con $c_{[i]} \in \mathbb{B}^N$.
for $i = 1, \ldots, l$ **do**
$m_{[i]} = D_k(c_{[i]})$.
return $m_{[1]} \cdots m_{[l]}$

Cipher-Block Chaining.

2.3 (CBCencrypt). **Input:** $m \in \mathbb{B}^*$
Output: $c \in \mathbb{B}^*$
$c_{[0]} \in \mathbb{B}^N$ {Es lo que se conoce como IV}
dividimos $m = m_{[1]} \cdots m_{[l]}$ con $m_{[i]} \in \mathbb{B}^N$.
for $i = 1, \ldots, l$ **do**
$c_{[i]} = E_k(m_{[i]} \oplus c_{[i-1]})$.
return $c_{[0]} \cdots c_{[l]}$

2.4 (CBCdecrypt). **Input:** $c \in \mathbb{B}^*$
Output: $m \in \mathbb{B}^*$
dividimos $c = c_{[0]} \cdots c_{[l]}$ con $c_{[i]} \in \mathbb{B}^N$.
for $i = 1, \ldots, l$ **do**

$m_{[i]} = D_k(c_{[i]}) \oplus c_{[i-1]}$.
return $m_{[1]} \cdots m_{[l]}$

Cipher FeedBack.

2.5 (CFBencrypt)**. Input:** $m \in \mathbb{B}^*$, $1 \leq r \leq N$
Output: $c \in \mathbb{B}^*$
$x_{[1]} \in \mathbb{B}^N$ {Es lo que se conoce como IV}
dividimos $m = m_{[1]} \cdots m_{[l]}$ con $m_{[i]} \in \mathbb{B}^r$.
for $i = 1, \ldots, l$ **do**
$c_{[i]} = m_{[i]} \oplus \mathrm{msb}_r(E_k(x_{[i]}))$.
$x_{[i+1]} = \mathrm{lsb}_{N-r}(x_{[i]}) \| c_{[i]}$
return $c_{[1]} \cdots c_{[l]}$

2.6 (CFBdecrypt)**. Input:** $c \in \mathbb{B}^*$, $1 \leq r \leq N$, $x_{[1]} \in \mathbb{B}^N$
Output: $m \in \mathbb{B}^*$
dividimos $c = c_{[1]} \cdots c_{[l]}$ con $c_{[i]} \in \mathbb{B}^r$.
for $i = 1, \ldots, l$ **do**
$m_{[i]} = c_{[i]} \oplus \mathrm{msb}_r(E_k(x_{[i]}))$.
$x_{[i+1]} = \mathrm{lsb}_{N-r}(x_{[i]}) \| c_{[i]}$
return $m_{[1]} \cdots m_{[l]}$

Output FeedBack.

2.7 (OFBencrypt)**. Input:** $m \in \mathbb{B}^*$
Output: $c \in \mathbb{B}^*$
$x_{[0]} \in \mathbb{B}^N$ {Es lo que se conoce como IV}
dividimos $m = m_{[1]} \cdots m_{[l]}$ con $m_{[i]} \in \mathbb{B}^N$.
for $i = 1, \ldots, l$ **do**

$x_{[i]} = E_k(x_{[i-1]})$
$c_{[i]} = m_{[i]} \oplus x_{[i]}$.
return $c_{[1]} \cdots c_{[l]}$

2.8 (OFBdecrypt). **Input:** $c \in \mathbb{B}^*$, $x_{[0]} \in \mathbb{B}^N$
Output: $m \in \mathbb{B}^*$
dividimos $c = c_{[1]} \cdots c_{[l]}$ con $c_{[i]} \in \mathbb{B}^N$.
for $i = 1, \ldots, l$ **do**
$x_{[i]} = E_k(x_{[i-1]})$
$m_{[i]} = c_{[i]} \oplus x_{[i]}$.
return $m_{[1]} \cdots m_{[l]}$

2.8

Apéndice: Sistemas de numeración

Como hemos destacado, en el mundo digital tanto los mensajes como las claves se identifican con cadenas de bits, que a su vez pueden identificarse con números enteros escritos en binario. Vamos a representar un número binario mediante una cadena de dígitos binarios iniciada por 0b. Así

$$0b11010011 = 1 + 2 + 16 + 64 + 128 = 211.$$

Como las cadenas binarias son especialmente largas, se suele usar la base 8 (octal) o la base 16 (hexadecimal). Así

$$211 = 0b11010011 = \texttt{0xD3} = 0o323$$

Ejercicio 2.1 (MiniAES)**.** MiniAES fue publicado en el año 2002 por R. C-W Phan, y es una versión reducida de AES que conserva su estructura pero permite hacer los cálculos de forma más comprensible para mejorar el entendimiento académico. La versión aquí propuesta difiere de la original en la función de sustitución.

En primer lugar se trabaja con el cuerpo de 16 elementos en lugar de con el cuerpo de 256 elementos. Dicho cuerpo se representa mediante $\mathbb{F}_{16} = \mathbb{Z}_2(\xi)_{\xi^4+\xi+1}$, es decir, la suma es bit a bit pero el producto es el producto polinomial módulo $\xi^4 + \xi + 1$. Los elementos de $\mathbb{F}_{16}$ se representan como cadenas de 4 bits o como un dígito hexadecimal, según el esquema

$$1011 = \texttt{0xB} = \xi^3 + \xi + 1,$$

es decir, identificamos $\mathbb{F}_{16} \cong \mathbb{F}_2^4$ a través de la base $\{\xi^3, \xi^2, \xi, 1\}$. La función de sustitución aquí empleada $\gamma : \mathbb{F}_{16} \to \mathbb{F}_{16}$ es la siguiente. Dado $x_3x_2x_1x_0 \in \mathbb{F}_2^4$, denotamos

$$\mathrm{inv}(x_3x_2x_1x_0) = \begin{cases} (x_3x_2x_1x_0)^{-1} & \text{si } x_3x_2x_1x_0 \neq 0000, \\ 0000 & \text{si } x_3x_2x_1x_0 = 0000. \end{cases}$$

Nuestra función es

$$\gamma(x_3x_2x_1x_0) = \mathrm{inv}(x_3x_2x_1x_0) \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} + (0011).$$

El bloque básico sobre el que actúa Mini AES es un bloque de 16 bits que se representa como una matriz 2×2 con coeficientes en $\mathbb{F}_{16}$ rellena por columnas, es decir,

$$\begin{bmatrix} a_0 & a_2 \\ a_1 & a_3 \end{bmatrix}$$

La función de sustitución `Sub` es la extensión de γ a cada valor de la matriz, es decir,

$$\gamma \begin{bmatrix} a_0 & a_2 \\ a_1 & a_3 \end{bmatrix} = \begin{bmatrix} \gamma(a_0) & \gamma(a_2) \\ \gamma(a_1) & \gamma(a_3) \end{bmatrix}.$$

La función `ShiftRow`, representada por π, realiza un desplazamiento en la última fila del bloque, es decir,

$$\pi \begin{bmatrix} a_0 & a_2 \\ a_1 & a_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_2 \\ a_3 & a_1 \end{bmatrix}$$

La función `MixColumn`, representada por θ, actúa mediante la siguiente descripción,

$$\theta \begin{bmatrix} a_0 & a_2 \\ a_1 & a_3 \end{bmatrix} = \begin{bmatrix} 0011 & 0010 \\ 0010 & 0011 \end{bmatrix} \begin{bmatrix} a_0 & a_2 \\ a_1 & a_3 \end{bmatrix}.$$

Por último la clave se añade de la misma forma que en el AES general, mediante la suma lógica XOR, y se representa mediante σ_{K_i}. Mini AES consta de dos rondas más una ronda cero al igual que AES. La última ronda no incluye `MixColumn`, así tenemos la siguiente descripción:

$$E_K = \sigma_{K_2} \circ \pi \circ \gamma \circ \sigma_{K_1} \circ \theta \circ \pi \circ \gamma \circ \sigma_{K_0}$$

La claves de ronda se obtienen

K_0	$w_0 = k_0$ $w_1 = k_1$ $w_2 = k_2$ $w_3 = k_3$
K_1	$w_4 = w_0 \oplus \gamma(w_3) \oplus 0001$ $w_5 = w_1 \oplus w_4$ $w_6 = w_2 \oplus w_5$ $w_7 = w_3 \oplus w_6$
K_2	$w_8 = w_4 \oplus \gamma(w_7) \oplus 0010$ $w_9 = w_5 \oplus w_8$ $w_{10} = w_6 \oplus w_9$ $w_{11} = w_7 \oplus w_{10}$

donde $K = (k_0, k_1, k_2, k_3)$ es la clave de 16 bits.

1. Calcula explícitamente la función γ.

2. Calcula γ', θ', π', K'_0, K'_1, K'_2 tales que

$$D_K = \sigma_{K'_2} \circ \pi' \circ \gamma' \circ \sigma_{K'_1} \circ \theta' \circ \pi' \circ \gamma' \circ \sigma_{K'_0}.$$

3. Calcula $c = E_{dni}(\texttt{0xA136})$ donde dni es el número de tu DNI módulo 65536 en binario.

4. Calcula $D_{dni}(c)$.

5. Calcula $E_{dni}(\texttt{0xA036})$. Compara el resultado con c.

Ejercicio 2.2. Una de las herramientas más empleadas en el diseño de cifrados de flujo son los LFSR (Registros lineales retroalimentados de desplazamiento). Un LFSR de orden t (homogéneo) viene dado por una aplicación lineal

$$f : \mathbb{F}_q^t \to \mathbb{F}_q, \quad [(x_0, \dots, x_{t-1}) \mapsto k_0 x_0 + \cdots + k_{t-1} x_{t-1}]$$

para ciertos $k_0, \dots, k_{t-1} \in \mathbb{F}_q$, que consideramos como la clave del criptosistema. Dado un estado inicial $(z_0, \dots, z_{t-1}) \in \mathbb{F}_q$, el vector de inicialización IV, se construye la sucesión

$$z_n = f(z_{n-t}, z_{n-t+1}, \dots, z_{n-1}).$$

La función de cifrado es

$$E_{k_0,\dots,k_{t-1}}(m_0 m_1 \cdots) = c_0 c_1 \cdots$$

donde

$$c_i = m_i + z_i.$$

En este ejercicio estamos considerando como alfabeto $\mathbb{F}_q$ en lugar del usual $\mathbb{B} = \mathbb{F}_2$. Supongamos que $q = 2^5$ y vemos $\mathbb{F}_{2^5} = \mathbb{F}_2(\xi)_{\xi^5+\xi^2+1}$. Identificamos los elementos de $\mathbb{F}_{2^5} = \mathbb{F}_2^5$, es decir, cada elemento se representa como una cadena de 5 bits según el esquema

$$10110 = \xi^3 + \xi^2 + 1.$$

Codificamos caracteres mediante 5 bits, el 00000 corresponde al espacio, los números del 00001 al 11011 son los 27 caracteres del alfabeto latino incluyendo la letra Ñ, los números del 11100 al 11111 son los signos de puntuación `. , ; :` respectivamente. Este sistema nos permite convertir los caracteres a una cadena de bits y de cadena de bits a lista de caracteres.

Partimos de una clave $\mathsf{key} = (\mathsf{k}_0, \mathsf{k}_1, \mathsf{k}_2, \mathsf{k}_3) \in \mathbb{F}_{2^5}^4$. Ciframos un cierto texto, y obtenemos el criptograma

`.LJMJ,RYRQVSNNQTFDWH.Ñ UAQTGI;Ñ.L`,

donde los cuatro primeros caracteres se corresponden con el IV. Sabiendo que firmo mis mensajes con la cadena `JAVIER.`, descifra el mensaje.

Ejercicio 2.3. Demostrar que un cifrado por bloques en modo CFB y OFB es un cifrado de flujo. ¿De qué tipo?

Ejercicios de evaluación de Criptosistemas simétricos

Ejercicio. Consideremos el cifrado por bloques MiniAES descrito en el ejercicio 2.1.

1. Calcula $\mathsf{E}_{\mathtt{dni}}(\mathtt{0x01234567})$ usando el modo CBC e IV $= \mathtt{0x0001}$.

2. Calcula $\mathsf{E}_{\mathtt{dni}}(\mathtt{0x01234567})$ usando el modo CFB, $\mathtt{r} = \mathsf{11}$, y vector de inicialización IV $= \mathtt{0x0001}$.

Bibliografía

[1] Hans Delfs and Helmut Knebl. *Introduction to Cryptography*. Information Security and Cryptography. Springer-Verlag Berlin Heidelberg, 2015.

[2] Andreas Enge. *Elliptic curves and their applications to cryptography. An Introduction.* Kluwer Academic Publishers, 1999.

[3] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford University Press, fourth edition, 1960.

[4] Nathan Jacobson. *Basic Algebra: I.* W.H. Freeman & Company, second edition, 1985.

[5] Neal Koblitz. *A Course in Number Theory and Cryptography*, volume 114 of *Graduate Texts in Mathematics*. Springer-Verlag New York, 2 edition, 1994.

[6] National Institute of Standards and Technology (NIST). *Digital Signature Standard (DSS)*, July 2013.

[7] Harald Niederreiter and Arne Winterhof. *Applied Number Theory*. Springer International Publishing, 2015.

[8] Nigel P. Smart. *Cryptography Made Simple*. Information Security and Cryptography. Springer International Publishing, 2016.

[9] Joachim von zur Gathen. *CryptoSchool*. Springer-Verlag Berlin Heidelberg, 2015.