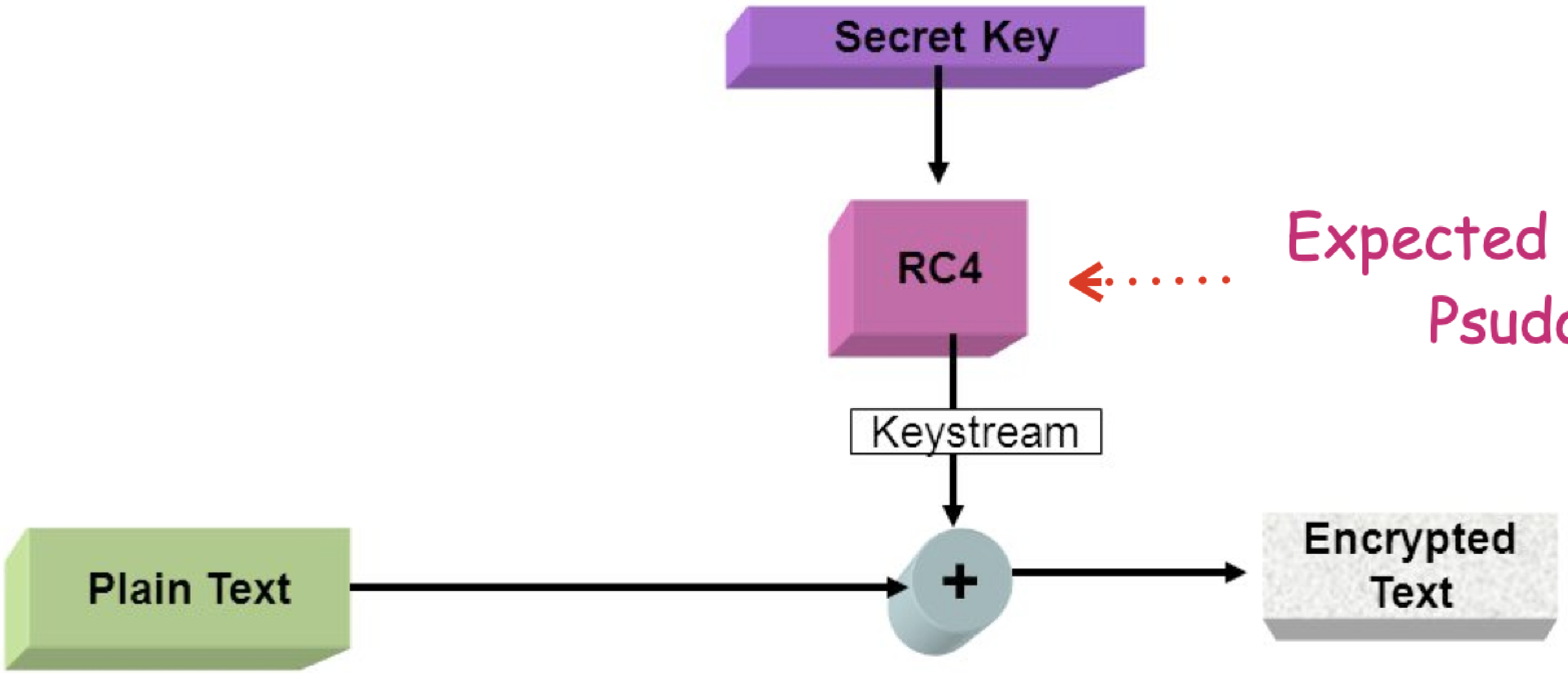
Cryptanalysis of RC4

RC4 Block Diagram

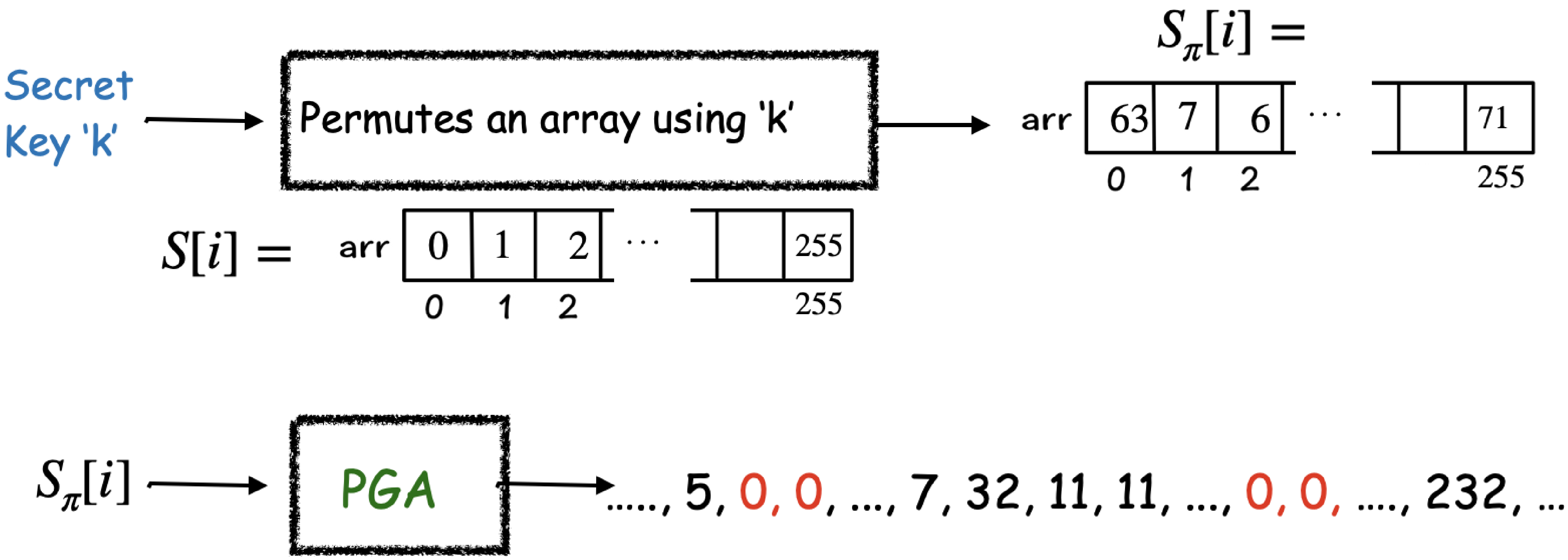


Expected to act like Psudo-RNG

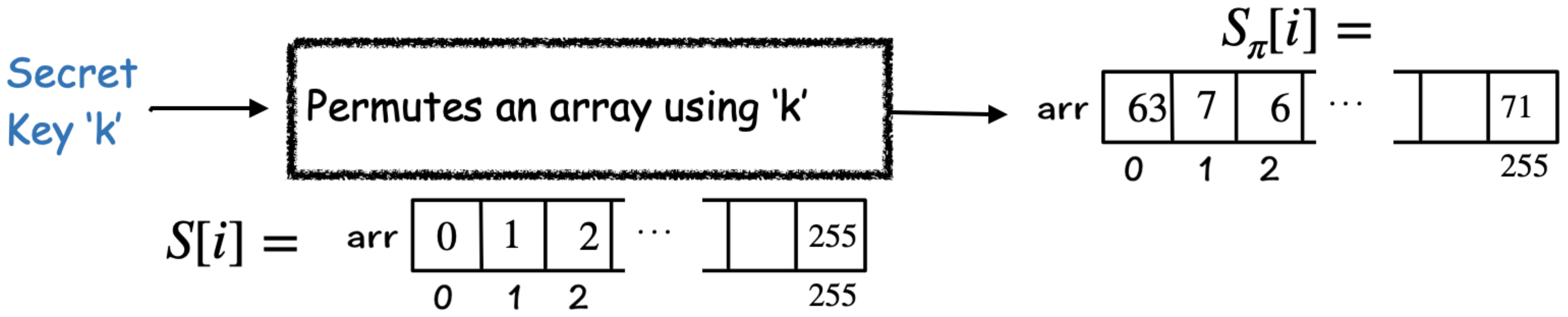
Image Credit: shorturl.at/syBNU

How it generate Psuedo-Random Number?

- Key Scheduling algorithm initiated by a secret key.
- Pseudo Random number generation algorithms (PGA).



Working of the Key Scheduling algorithms?



$$
\begin{array}{l}
\text{input: string of bytes } s \\
\text{for } i \leftarrow 0 \text{ to } 255 \text{ do:} \quad S[i] \leftarrow i \\
j \leftarrow 0 \\
\text{for } i \leftarrow 0 \text{ to } 255 \text{ do} \\
\quad k \leftarrow s\big[i \bmod |s|\big] \quad // \quad \textit{extract one byte from seed} \\
\quad j \leftarrow \big(\, j + S[i] + k \,\big) \bmod 256 \\
\quad \text{swap}(S[i], S[j])
\end{array}
$$

Psuedo Code: Boneh & Shoup

Pseudo Random number generating algorithm

$S_\pi[i] \longrightarrow \boxed{\text{PGA}} \longrightarrow$ ....., 5, 0, 0, ..., 7, 32, 11, 11, ..., 0, 0, ..., 232, ...

$i \leftarrow 0, \quad j \leftarrow 0$

repeat

$\qquad i \leftarrow (i+1) \bmod 256$

$\qquad j \leftarrow (j + S[i]) \bmod 256$

$\qquad \text{swap}(S[i], S[j])$

$\qquad \text{output} \quad S\big[\ (S[i] + S[j]) \bmod 256\ \big]$

forever

Psuedo Code: Boneh & Shoup

Timeline of cryptanalysis of RC4: Single byte biases

$$S_\pi[i] \longrightarrow \boxed{\text{PGA}} \longrightarrow \begin{array}{l} z_1,\ z_2,\ z_3,\ \ldots . z_r, \ldots \\ 5, 0, 19, \ldots, 7, 32, 11, 11, \ldots, \ldots, 232, \ldots \end{array}$$

- [Mantin-Samir 2001]: $P[z_2 = 0] \approx \frac{2}{n} \neq \frac{1}{n}[!]$
- [Mironov 2002]: Described distribution of z_1 *output*.
- [Maitra-Paul-SenGupta]: If

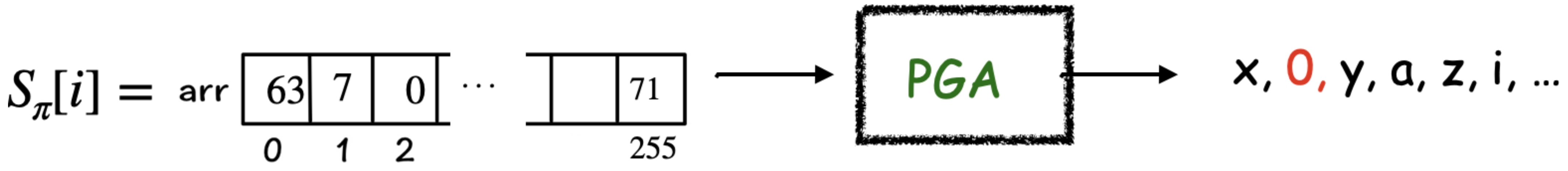
  $3 \leq r \leq 255,\ P[z_r = 0] = \frac{1}{n} + \frac{c_r}{n^2};\ 0.242811 \leq c_r \leq 1.337057$
- [SenGupta-Maitra-Paul-Sarkar 2011]:

$$P[z_l = 256 - l] \geq \frac{1}{n} + \frac{1}{n^2};\ l = Keylength$$

Martin- Samir Lemma (2001):

- Deterministic output of PNRG:

  $S[2] = 0$ *and* $S[1] \neq 2$ *always produces* $z_2 = 0$ [!]



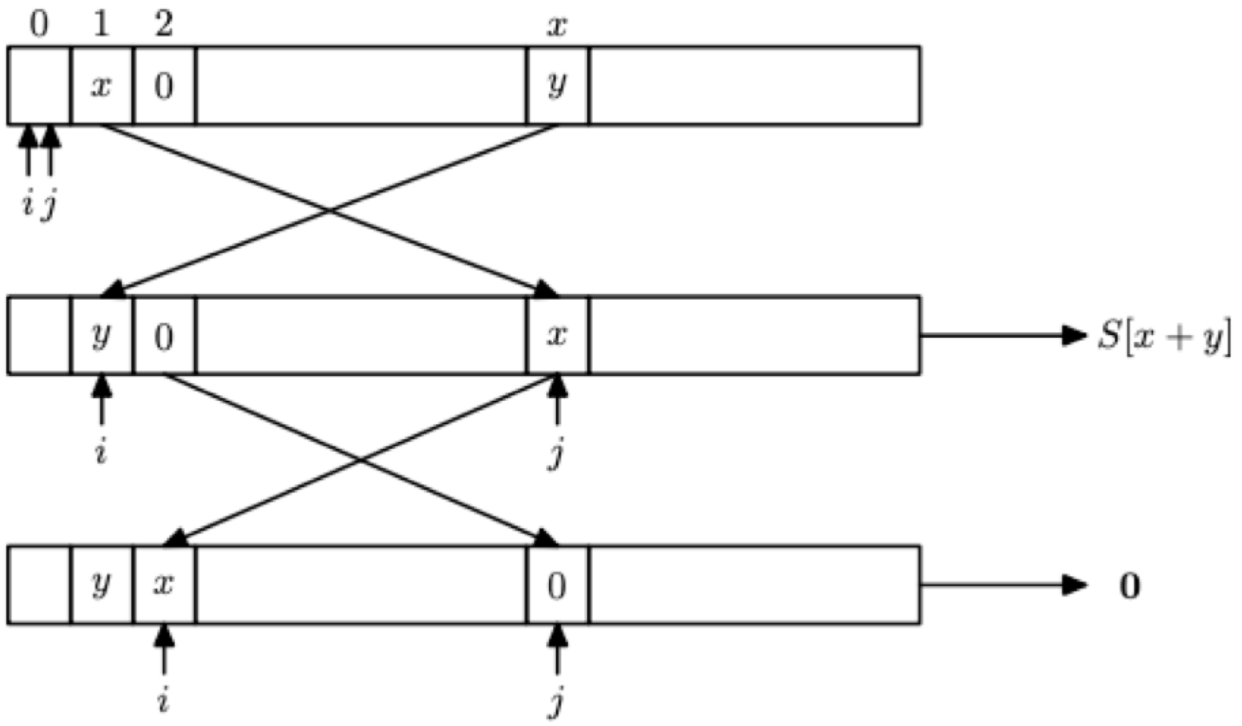
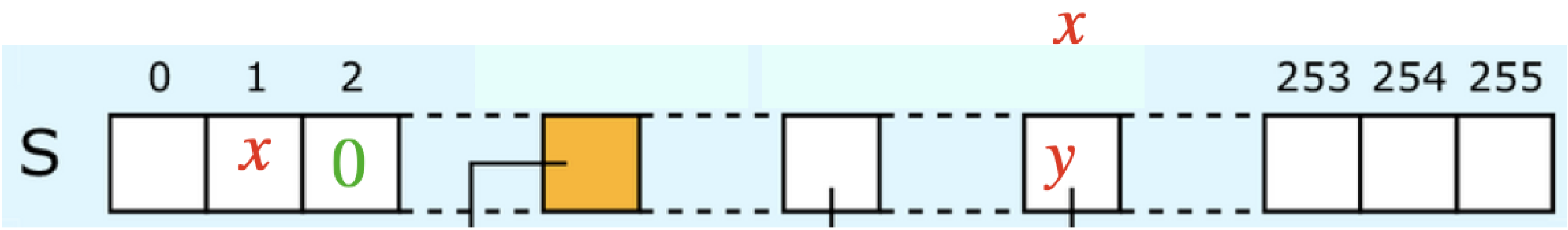


Image: Boneh & Shoup



$Event:$

$S[2] = 0 \ and \ S[1] \neq 2$

$\sim Event:$

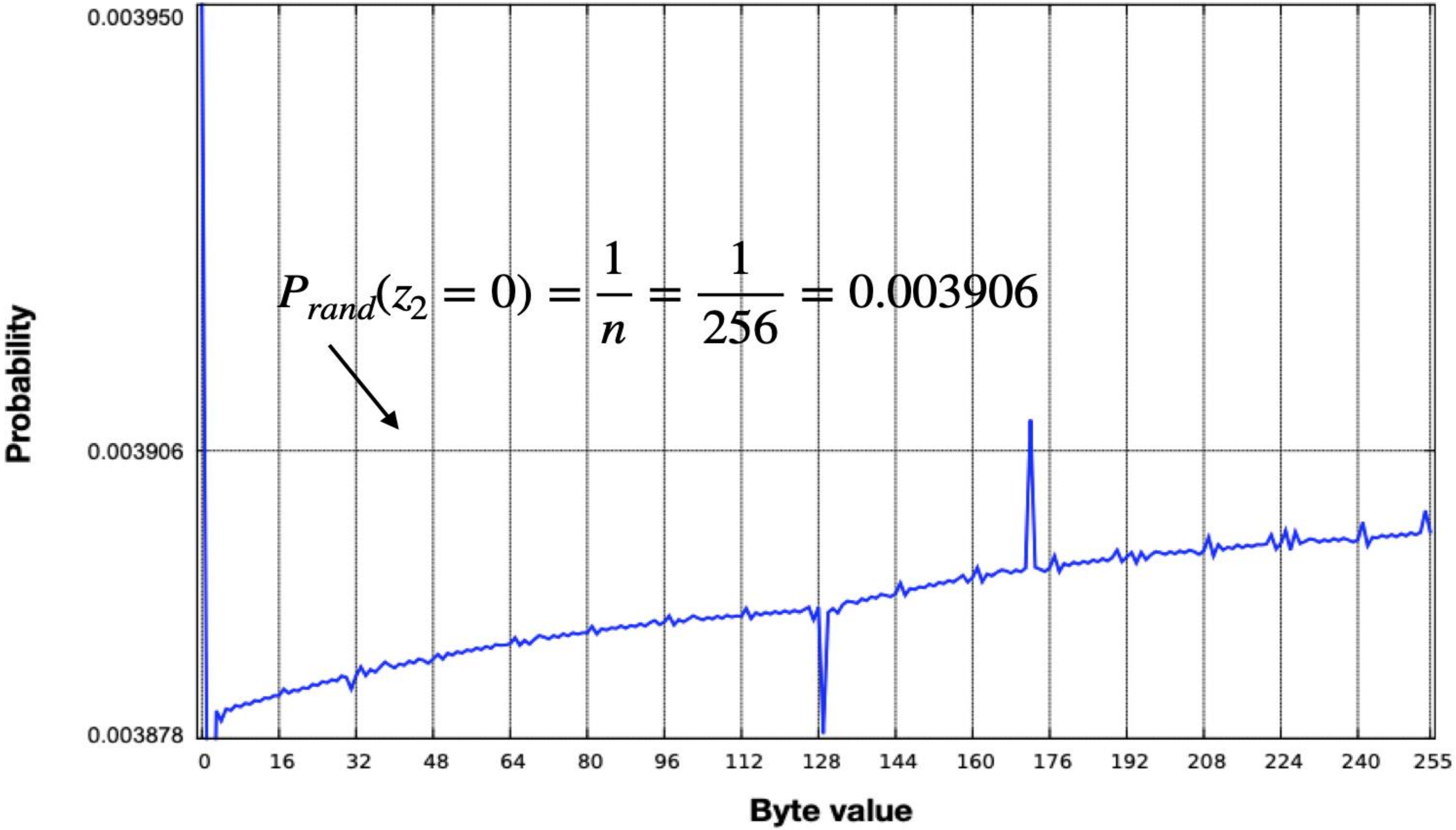
$P(Event) = \frac{1}{n} \times (1 - \frac{1}{n})$

$P(\sim Event) = 1 - P(Event)$

$P(z_2 = 0) = 1$

$P(z_2 = 0) = \frac{1}{n}$

$Conclusion: \ P[z_2 = 0] = 1 \times P(Event) \ + \ \frac{1}{n} \times P(\sim Event) \approx \frac{2}{n}$



Notice spike in probability at byte value zero .

Conclusion : Zero is more likely to appear as z_2 .

Image: AlFardan et.al (2013)

[Mironov 2002]: Described distribution of z_1 *output*.

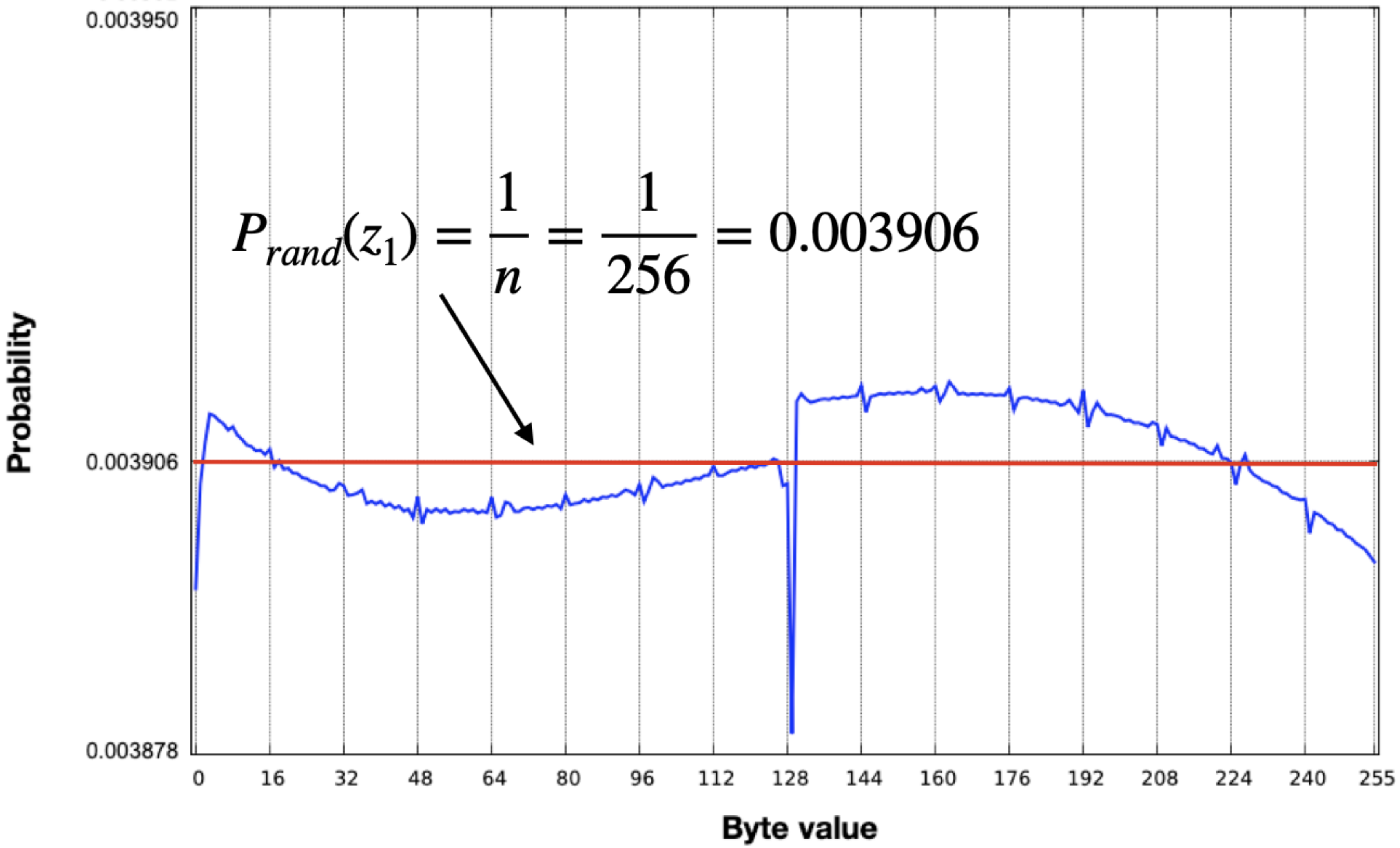


Image: AlFardan et.al (2013)

Can it be used to launch any crypto attack targeting us (general public)?

- Several browser in late 90's used RC4 to encrypt data.
- An usual data transmitted as: $E_{RC4}(Cookie \,||\, client\ request)$
- Cookie could be sensitive data too !!!

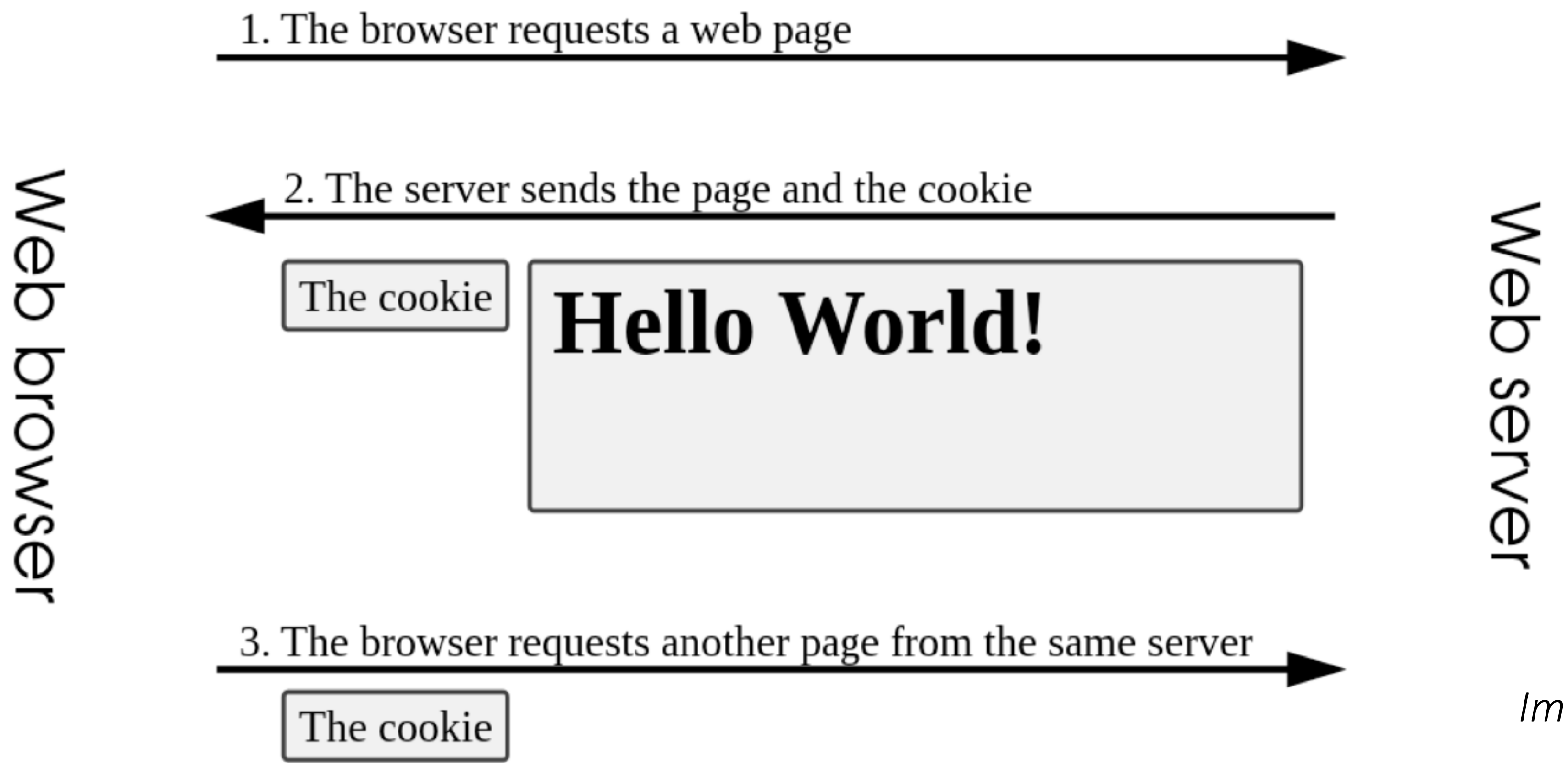


Image: WIkipedia HTTP cookies

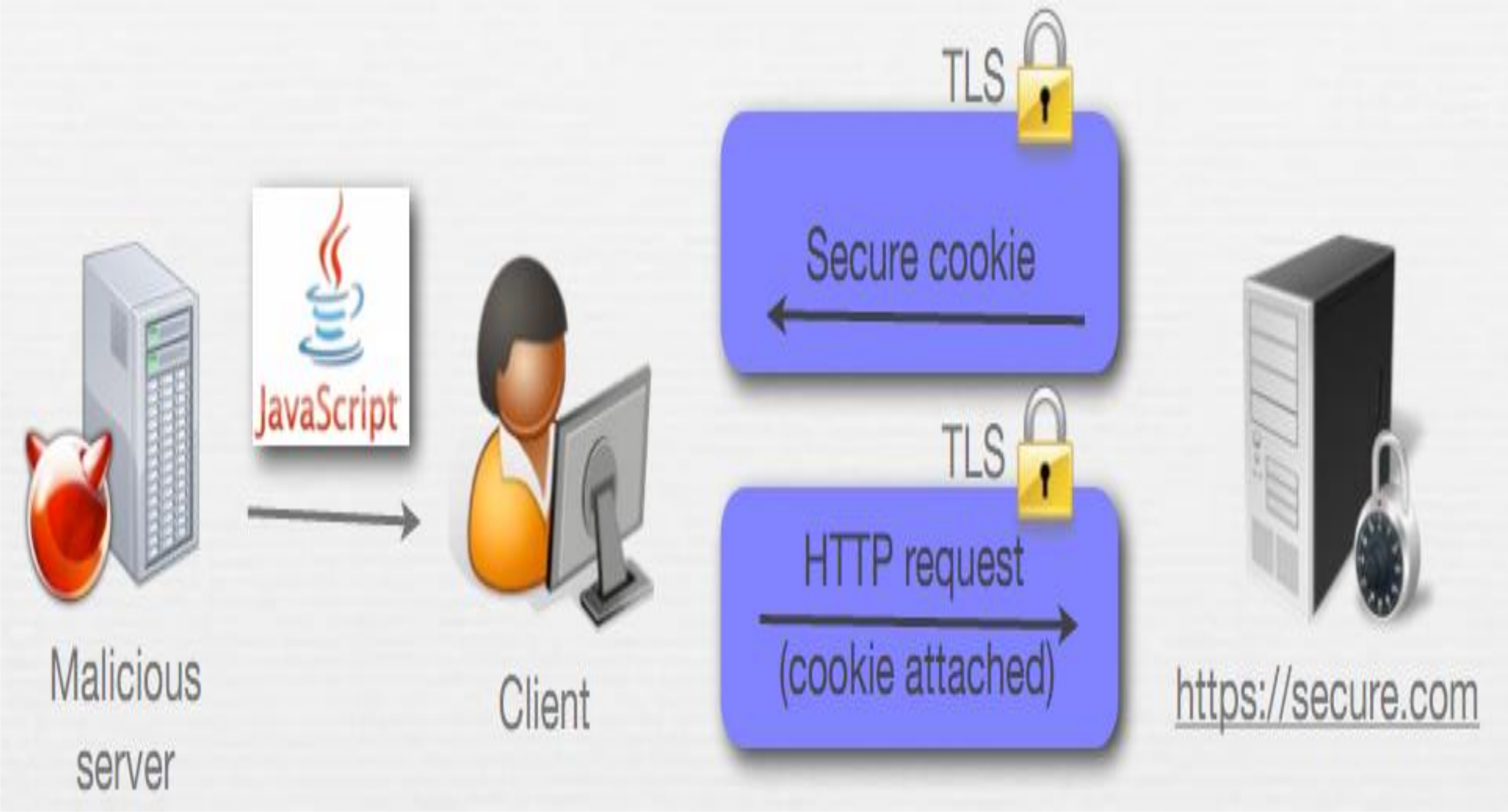


Image Credit: www.isg.rhul.ac.uk/tls/

Our Cookies at risk!!!

- It has been shown that of a single plaintext under 2^{30} random keys can help to extract first 128 bytes of the plaintext.

$$
\begin{aligned}
E_{RC4,\,k_1}(P_1) &= C_1 \\
E_{RC4,\,k_2}(P_1) &= C_2 \\
E_{RC4,\,k_3}(P_1) &= C_3 \\
&\dots\dots \\
2^{30}\ &times\ (!)
\end{aligned}
$$

- Several cookies are usually embedded in the first a few hundred bytes of the plaintext.
- A brief explanation on the next slide.

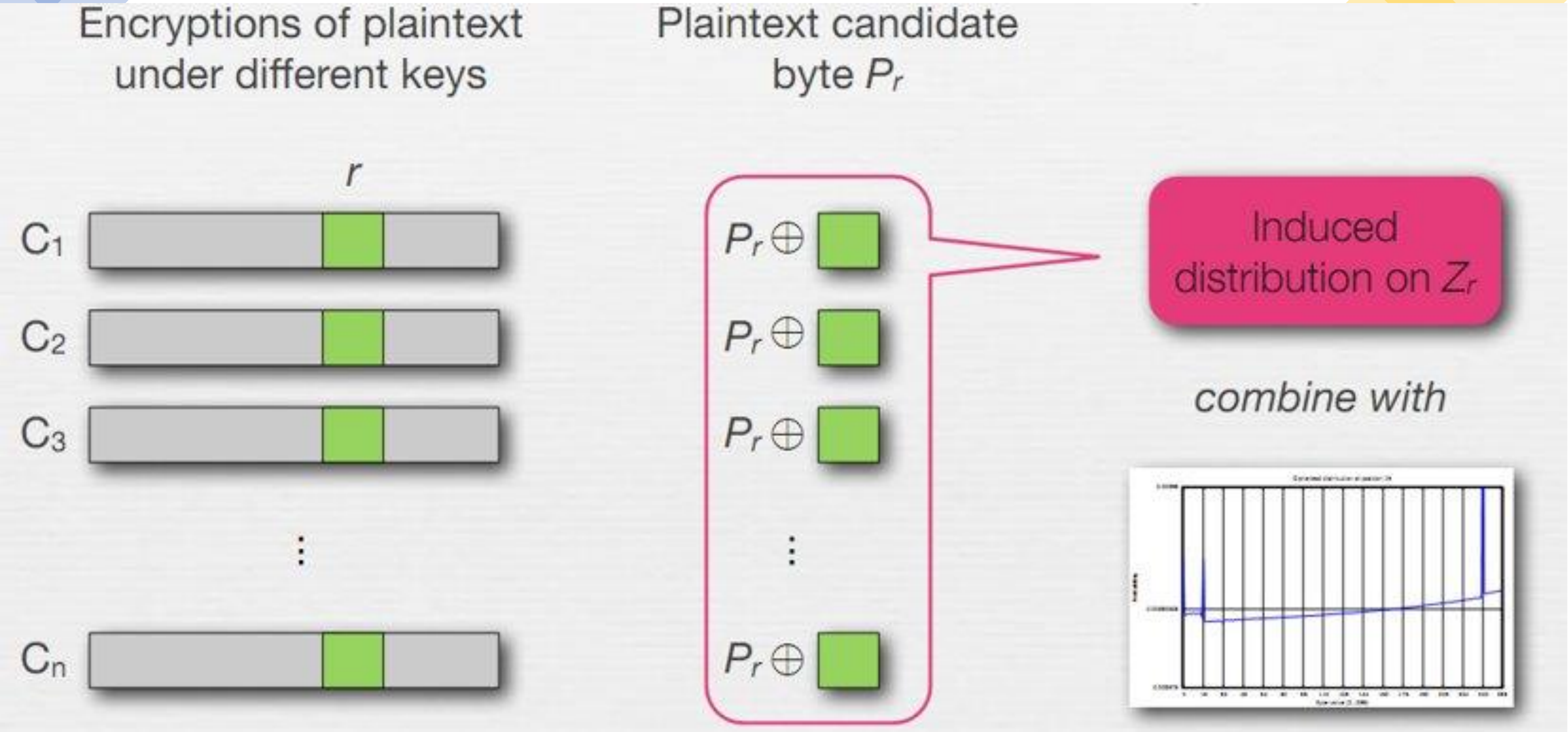


Image: AlFardan et.al (2013)

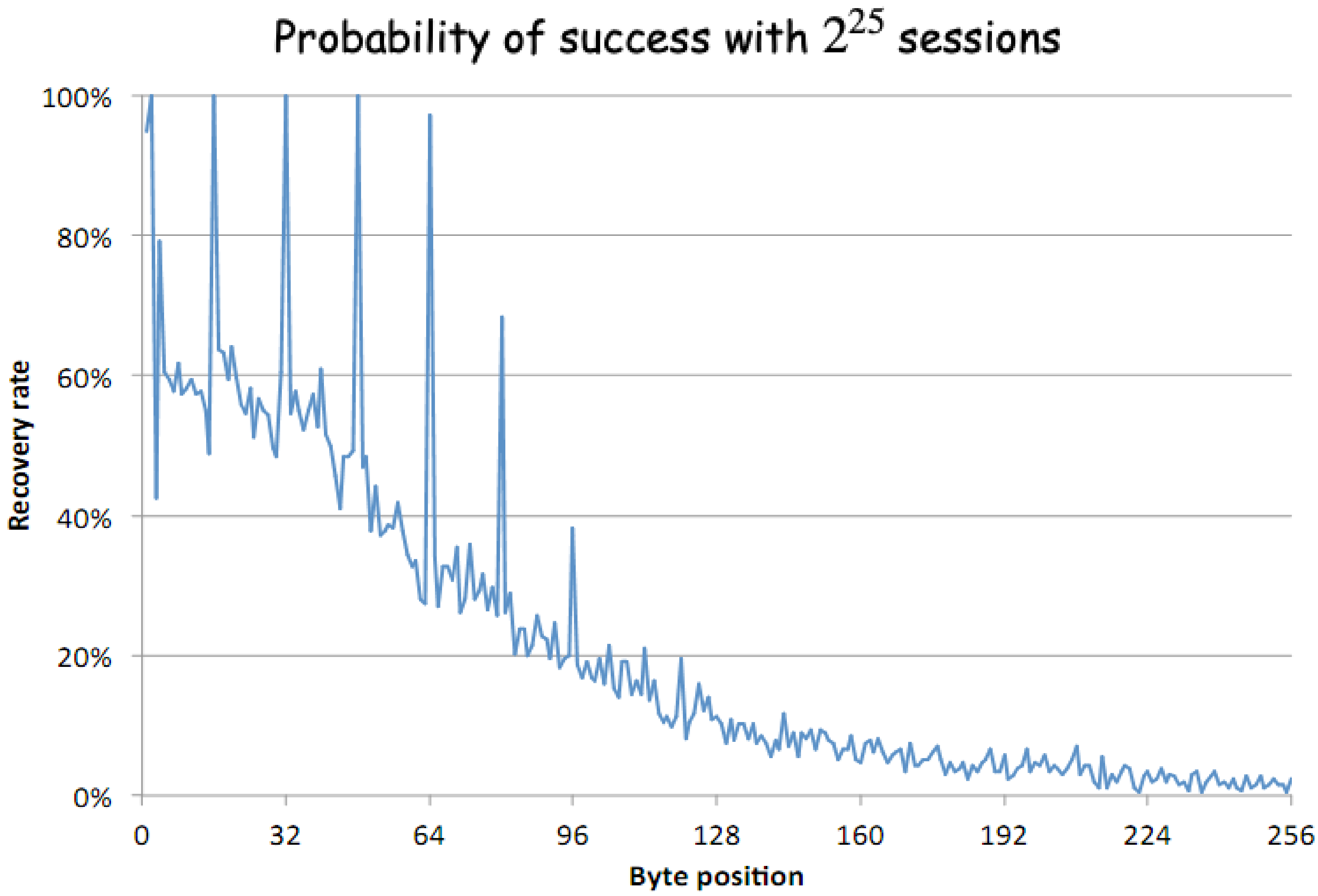


Image Credit: www.isg.rhul.ac.uk/tls/

Probability of success with 2^{30} sessions

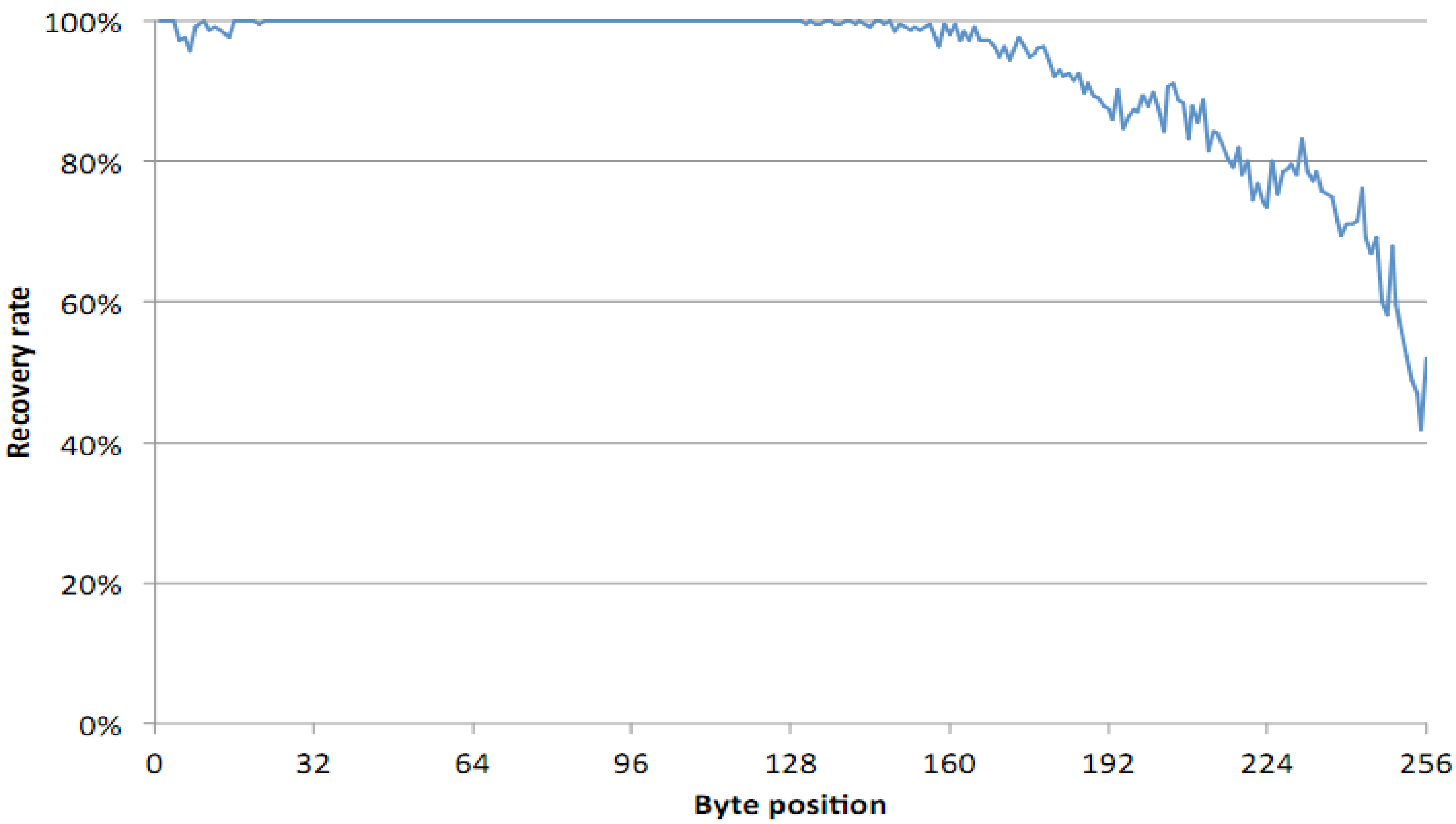


Image Credit: www.isg.rhul.ac.uk/tls/

Cryptanalysis of RC4: consecutive bytes biases

$$\ldots, z_r, z_{r+1}, \ldots, \ldots, z_n, z_{n+1}, \ldots$$

$$S_\pi[i] \longrightarrow \boxed{\text{PGA}} \longrightarrow \ldots, 5, 0, 0, \ldots, 7, 32, 11, 11, \ldots, 0, 0, \ldots, 232, \ldots$$

- Fluhrer-McGrew identified biases for consecutive bytes.
- They occurs more frequently than we expect from an ideal random string of same length.
- It can be used to launch cryptographic attacks.

Bias in RC4 Stream generator

> **Let a RC4 output → 5, 0, 0, 23, ..., 7, 32, 11, 11, ..., 0, 0 ...**

> **Let a ideal random generator → 83, 7, 69, 0, 0, ..., 42, 4, 13, ..., 27, 8 ...**

Fluher- Martin Theorem:

Lemma $\boxed{}$ **(Fluhrer-McGrew).** *Suppose RC4 is initialized with a random state T in ST_{RC4}. Let (z_1, z_2) be the first two bytes output by RC4 when started in state T. Then*

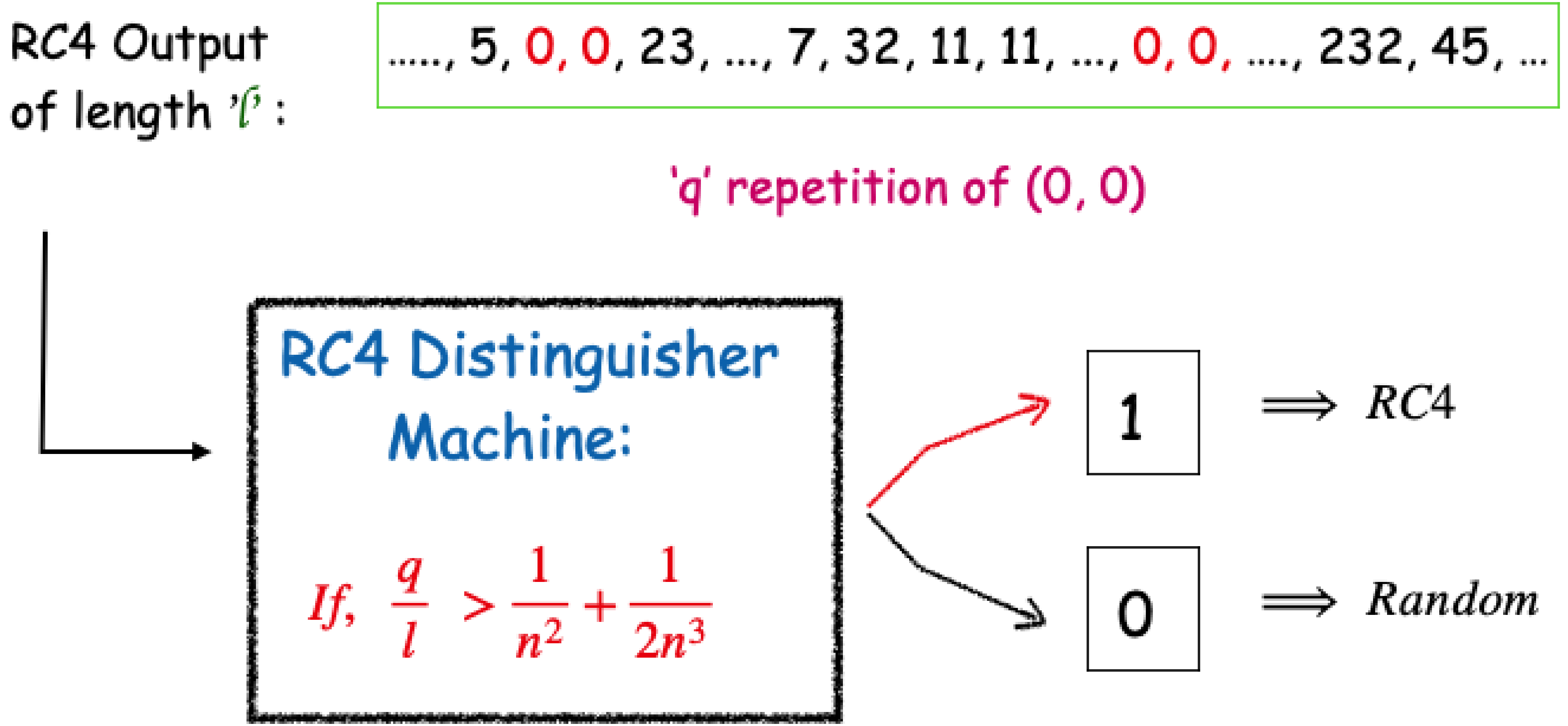
$$i \neq n-1 \implies \Pr[(z_1, z_2) = (0,0)] \geq (1/n^2) \cdot (1 + (1/n))$$

$$i \neq 0, 1 \implies \Pr[(z_1, z_2) = (0,1)] \geq (1/n^2) \cdot (1 + (1/n))$$

In ideal random string→ Probability of occurance of (......, x, y, ..., ...)

$$= P[x] \times P[y] = \frac{1}{n} \times \frac{1}{n} = \frac{1}{n^2}$$

Result: Boneh & Shoup



RC4 Output of length 'l' :
....., 5, 0, 0, 23, ..., 7, 32, 11, 11, ..., 0, 0, ..., 232, 45, ...
'q' repetition of (0, 0)
RC4 Distinguisher Machine:
If, $\frac{q}{l} > \frac{1}{n^2} + \frac{1}{2n^3}$
1
0
$\Longrightarrow$ RC4
$\Longrightarrow$ Random

Distinguisher (say, D) advantage as function of output stream 'l':

$$\ell = 2^{14} \text{ bytes:} \qquad \text{PRGadv}[D, RC4] \geq 2^{-8}$$
$$\ell = 2^{34} \text{ bytes:} \qquad \text{PRGadv}[D, RC4] \geq 0.5$$

What if one uses other anomalous digraphs along with (0, 0) and (0,1) ?

Then, Distinguisher advantage as function of 'l':

$$l = 2^{30.6} : \quad PRGadv[D, RC4] \geq 0.8$$

Result: Boneh & Shoup

Appendix:

We have written a python program to **manually** verify the Martin-Shamir Lemma (2001) for case:

$$S[2] = 0 \; and \; S[1] \neq 2 \implies P[z_2 = 0] = 1$$

https://colab.research.google.com/drive/1cEblhoQ9gVXtGIqXSgHfACx9Rqb9ruqL?usp=sharing

Thank You