2550 Intro to cybersecurity

L9: Crypto PRG

abhi shelat

One-time pad

PROBLEMS:

KEY IS AS LONG AS THE MESSAGE.

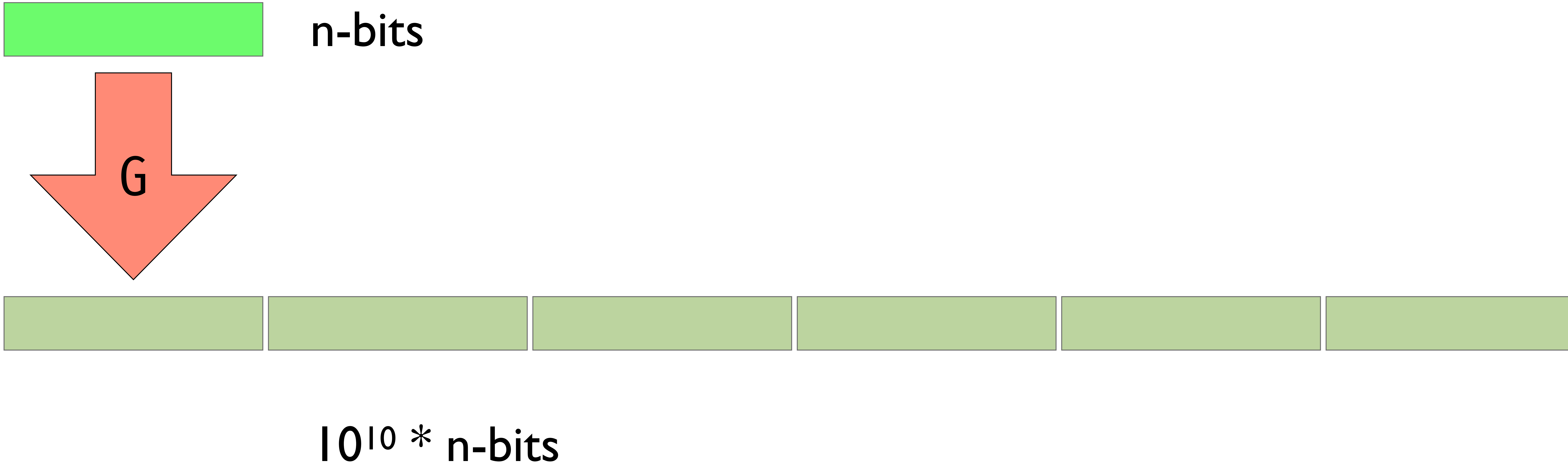
REQUIRED FOR PERFECT SECURITY.

$$
\begin{aligned}
\mathcal{M} &= \{0,1\}^n \\
\mathcal{K} &= \{0,1\}^n \\
\mathsf{Gen} &= k = k_1 k_2 \ldots k_n \leftarrow \{0,1\}^n \\
Enc_k(m_1 m_2 \ldots m_n) &= c_1 c_2 \ldots c_n \text{ where } c_i = m_i \oplus k_i \\
Dec_k(c_1 c_2 \ldots c_n) &= m_1 m_2 \ldots m_n \text{ where } m_i = c_i \oplus k_i
\end{aligned}
$$

Goal: Symmetric encryption with a "short" key that works for **1** arbitrarily long message

Tradeoff: Must settle for weaker security (not perfect)

Goal: One key to a long key



Perfect secrecy

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{M}, \mathcal{K})$

is said to be PERFECTLY SECRET if

$$\forall m_1, m_2 \in \mathcal{M}, \forall c$$

$$\Pr[k \leftarrow \mathsf{Gen} \ : \ \mathsf{Enc}_k(m_1) = c]$$

$$=$$

$$\Pr[k \leftarrow \mathsf{Gen} \ : \ \mathsf{Enc}_k(m_2) = c]$$

Perfect secrecy

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{M}, \mathcal{K})$

is said to be PERFECTLY SECRET if

$$\forall m_1, m_2 \in \mathcal{M}, \forall c$$

$$\Pr[k \leftarrow \mathsf{Gen} \ : \ \mathsf{Enc}_k(m_1) = c]$$

$$=$$

$$\Pr[k \leftarrow \mathsf{Gen} \ : \ \mathsf{Enc}_k(m_2) = c]$$

Indistinguishable secrecy

$$\Pr[k \leftarrow \mathsf{Gen} \ : \ \mathsf{Enc}_k(m_1) = c]$$

$$\sim$$

$$\Pr[k \leftarrow \mathsf{Gen} \ : \ \mathsf{Enc}_k(m_2) = c]$$

"So close that no efficient computer can distinguish"

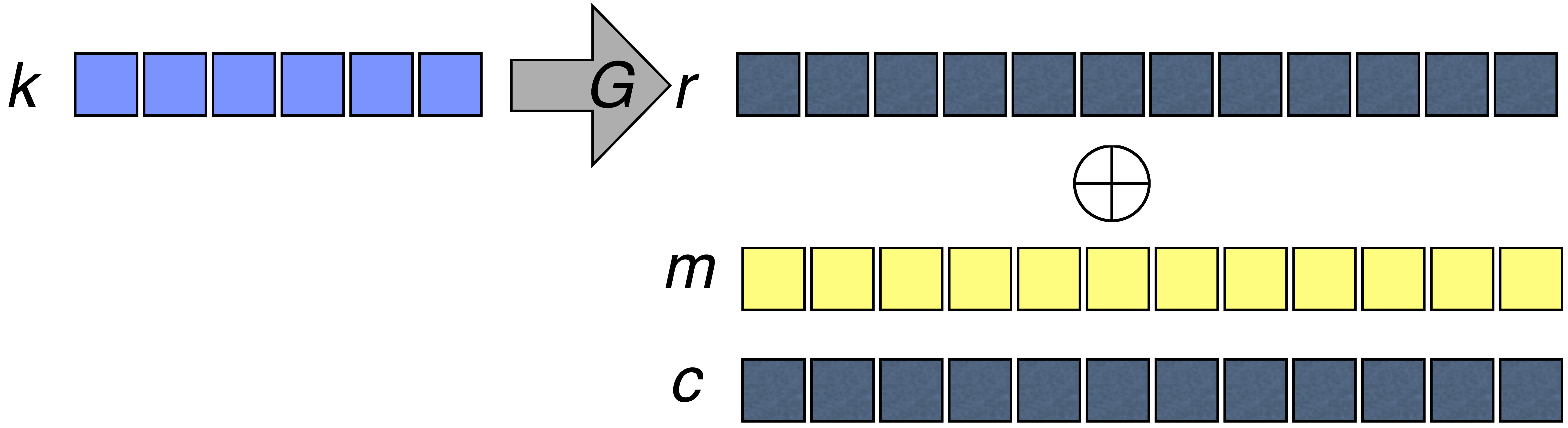
This is the idea behind a stream cipher.

An encryption scheme

Gen(1^n) $k \leftarrow U_{n/2}$ (key generation)

Enc$_k$(*m*) $r \leftarrow G(k)$ $|r| = |m|$ (encryption)

output $m \oplus r$



Stream cipher

Gen: pick an n-bit binary string k

Enc(k,m): Output G(k) + m

Dec(k,c): Output G(k) + c

what security properties are needed for this to work?

$$
\begin{aligned}
\mathcal{M} &= \{0,1\}^n \\
\mathcal{K} &= \{0,1\}^n \\
\mathsf{Gen} &= k = k_1 k_2 \ldots k_n \leftarrow \{0,1\}^n \\
Enc_k(m_1 m_2 \ldots m_n) &= c_1 c_2 \ldots c_n \text{ where } c_i = m_i \oplus k_i \\
Dec_k(c_1 c_2 \ldots c_n) &= m_1 m_2 \ldots m_n \text{ where } m_i = c_i \oplus k_i
\end{aligned}
$$

$$U_n$$

One time pad needed keys from uniform distribution on strings of len n

what security properties are needed for this to work?

"Same # of 0s as 1s?"

Vigenere cipher

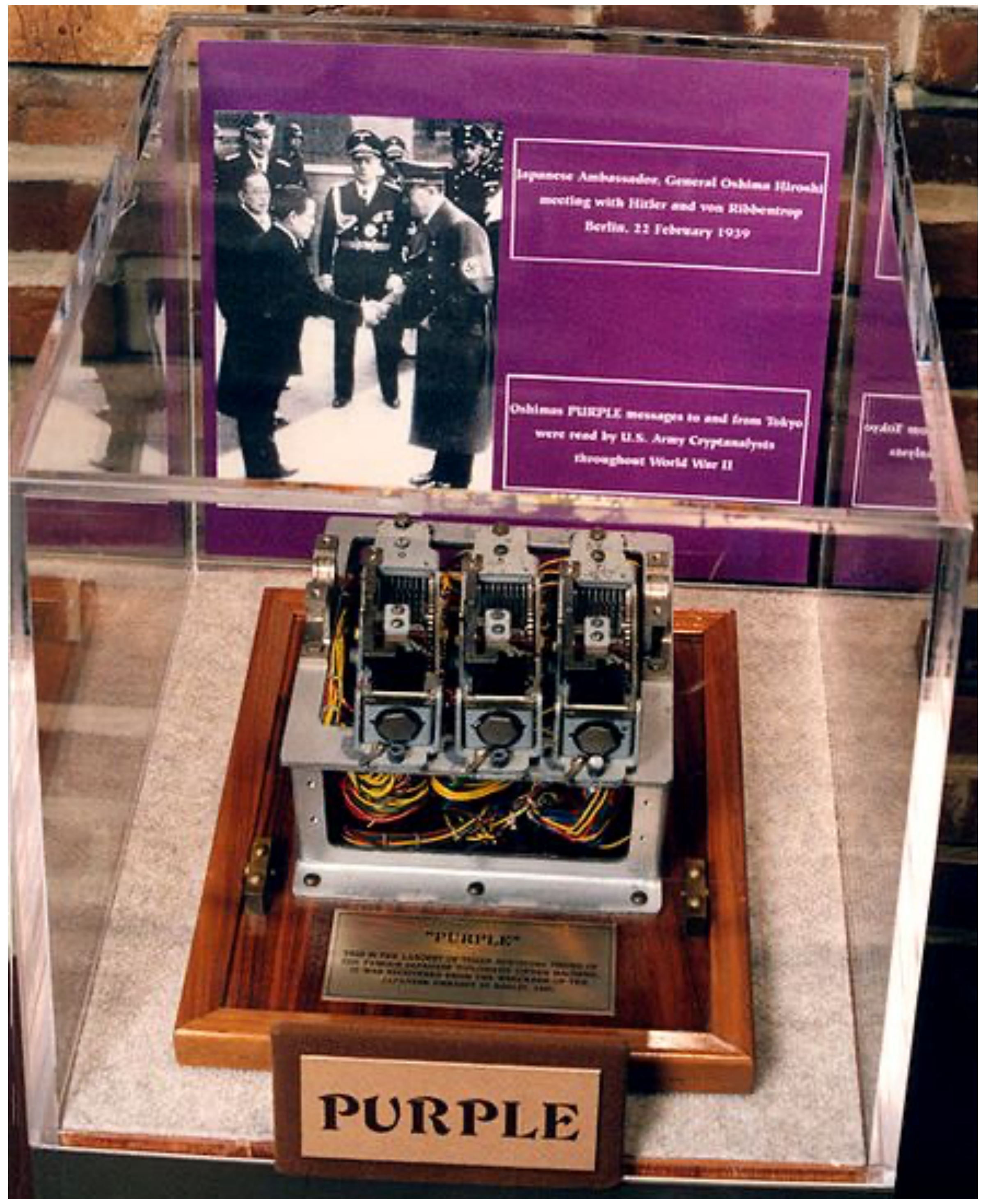
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

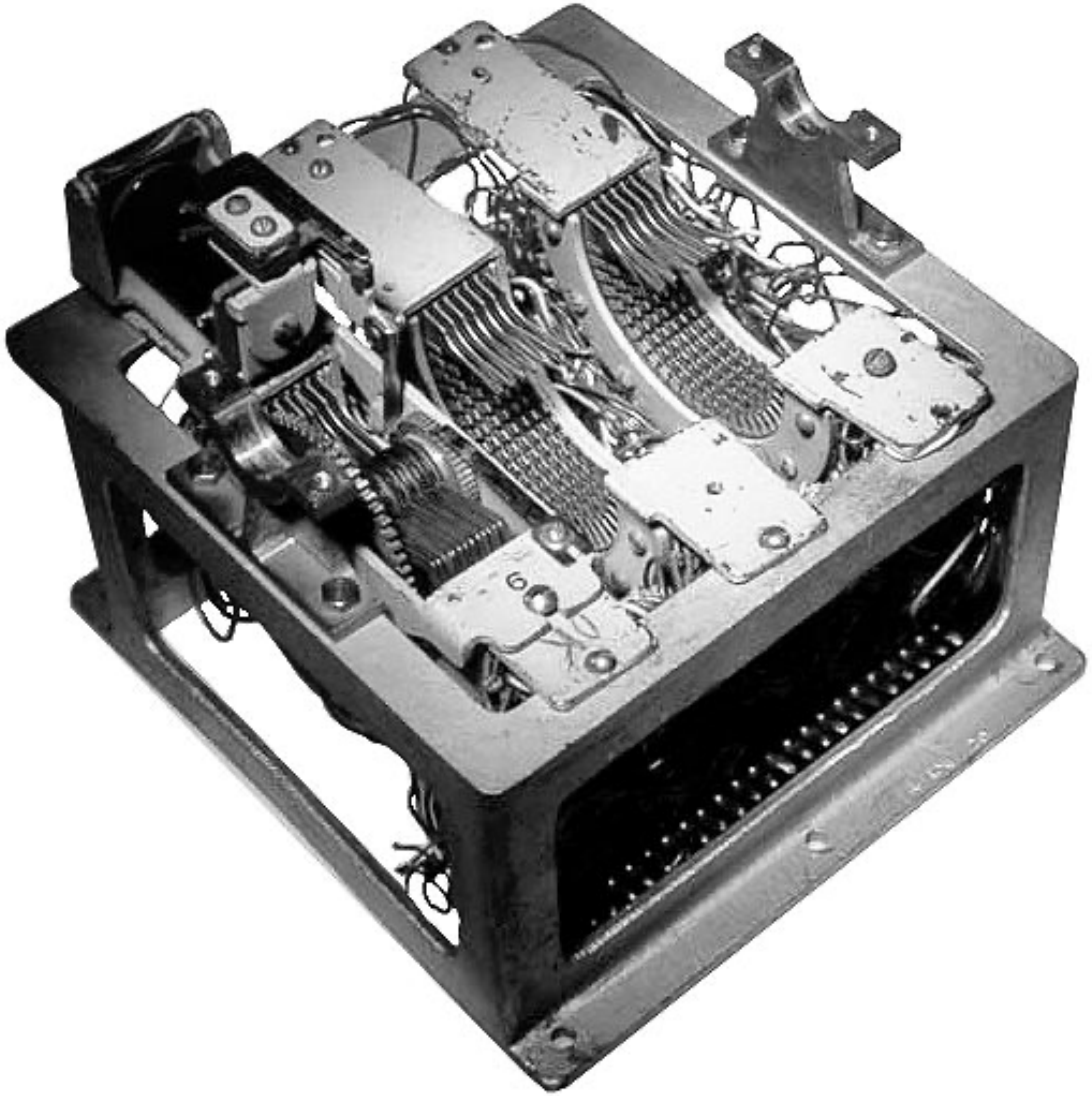
MSG: T H E M O D E R N S T U D Y O F . . .

KEY: A B H I A B H I A B H I A B H I A B H I A B H I

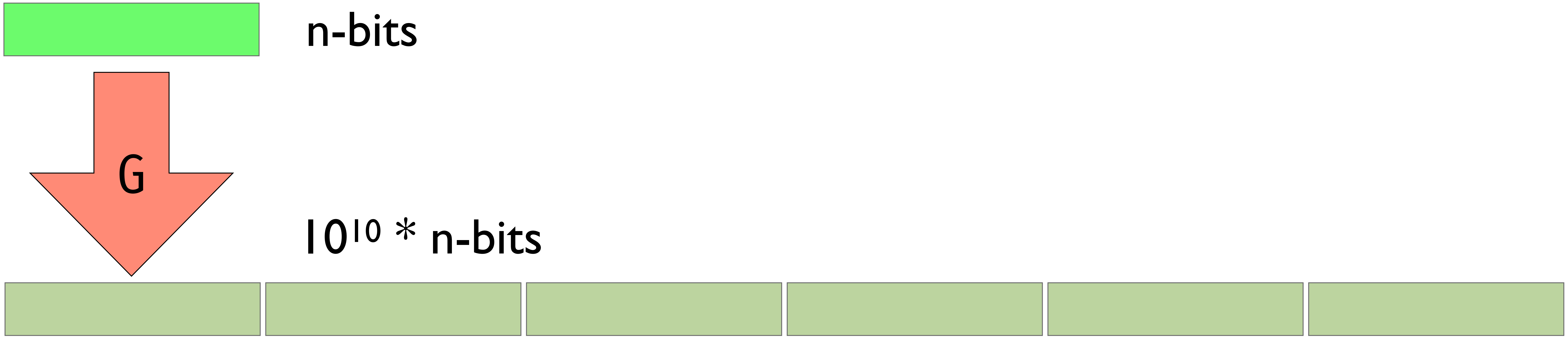
ciphertext: T I L U O E L Z N T A C D Z V N . . .

Other examples





Enigma



should "appear" to be the same as a random string $\{0,1\}^{10^{10}n}$

$$U_{10^{10}n}$$

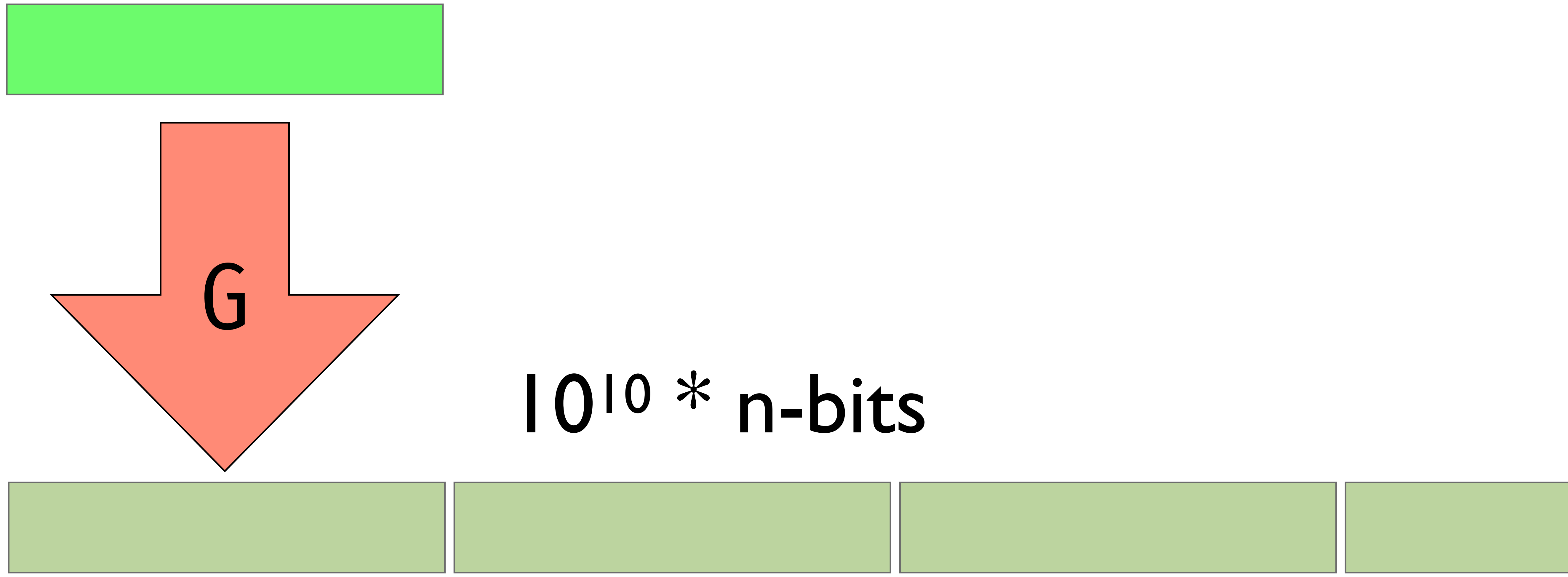
what does it mean for a process G that produces keys to be pseudo-random?

“Computational Indistinguishability” provides a precise way of formulating pseudo-randomness

Truly random

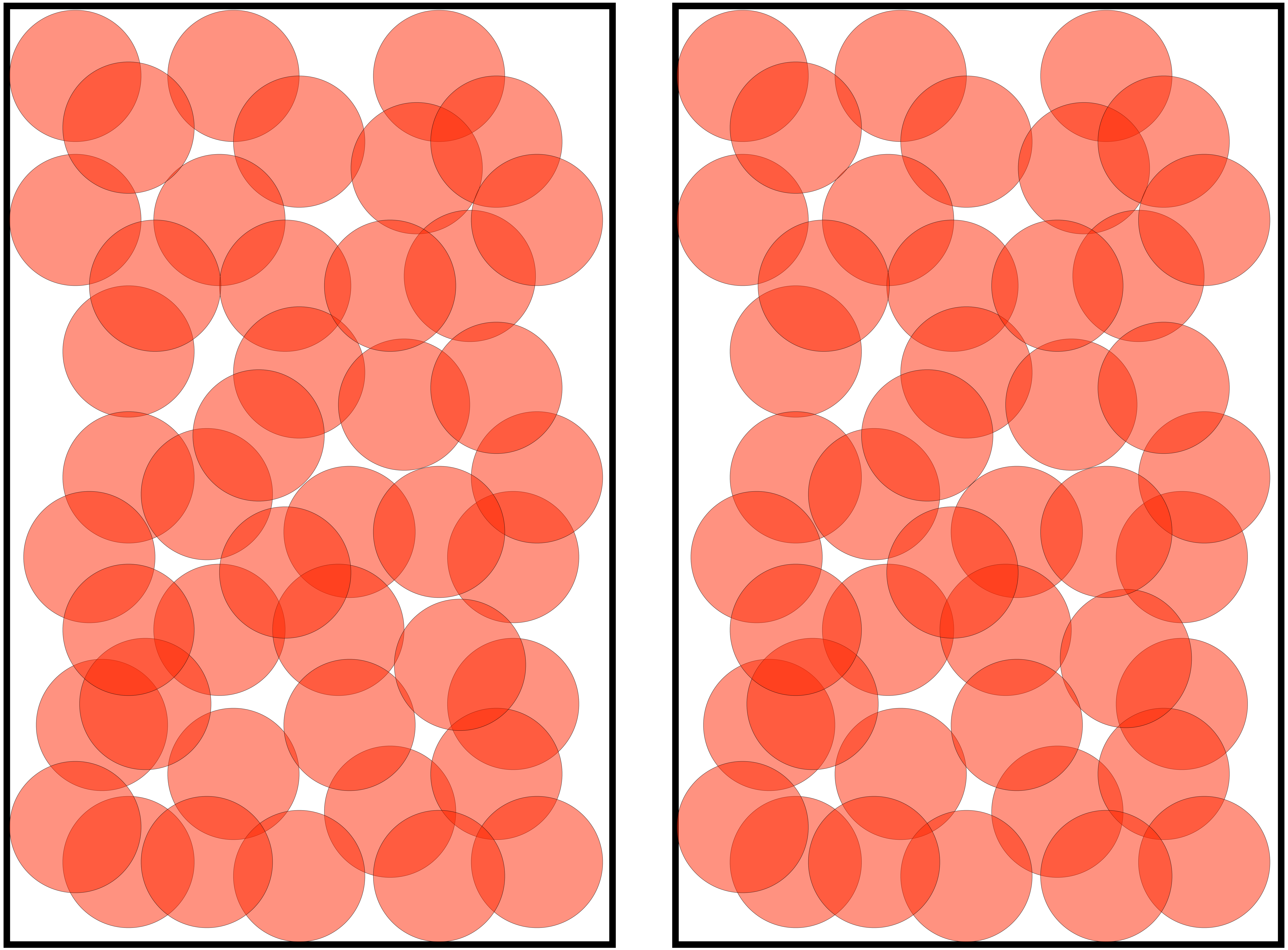
$$U_n$$

Pseudo-randomness



next slide has 2 pics

are they the same
or different?



same or different?

twice the time.

same or different?

lesson:

Ability to answer correctly...

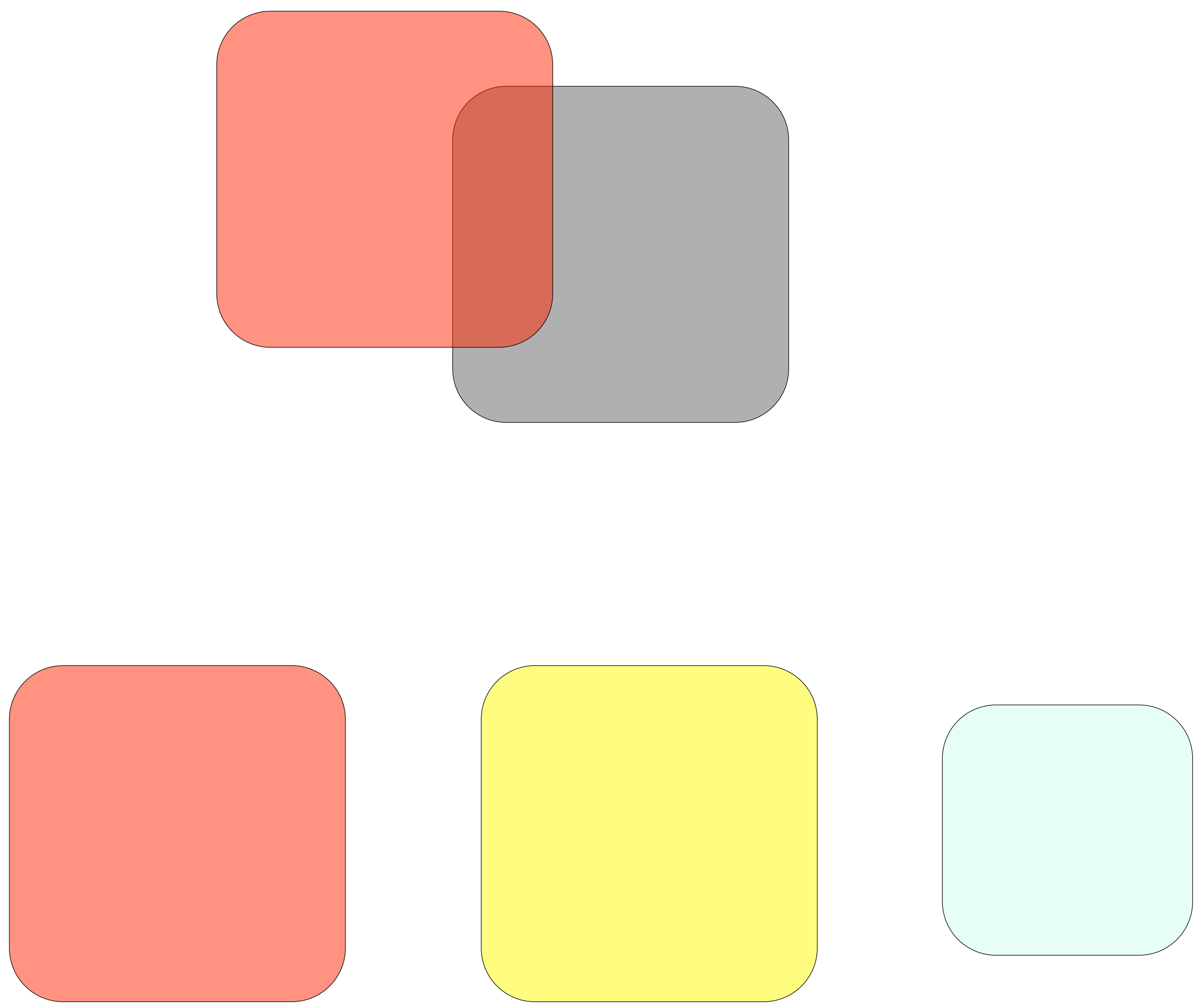
http://i.telegraph.co.uk/telegraph/multimedia/archive/01553/toss_1553792c.jpg

NEW PROBLEM:

consider all drawings consisting of boxes.

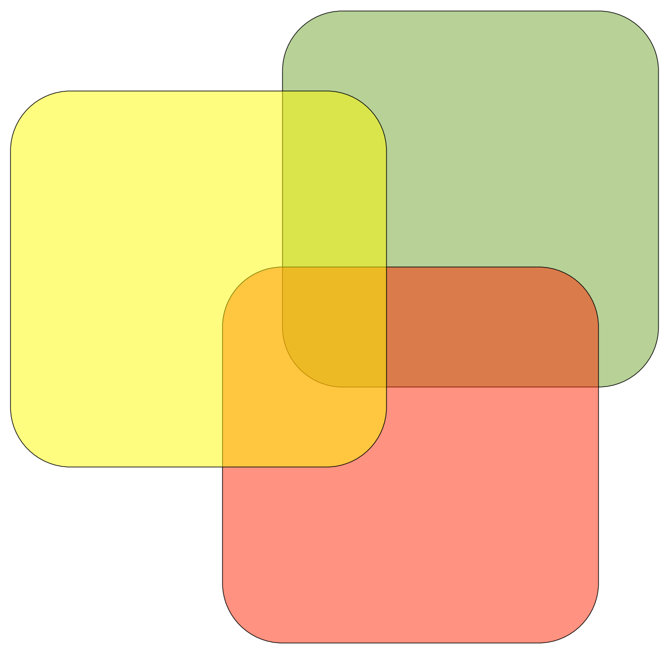
evens

# of boxes that overlap another box is even



odds

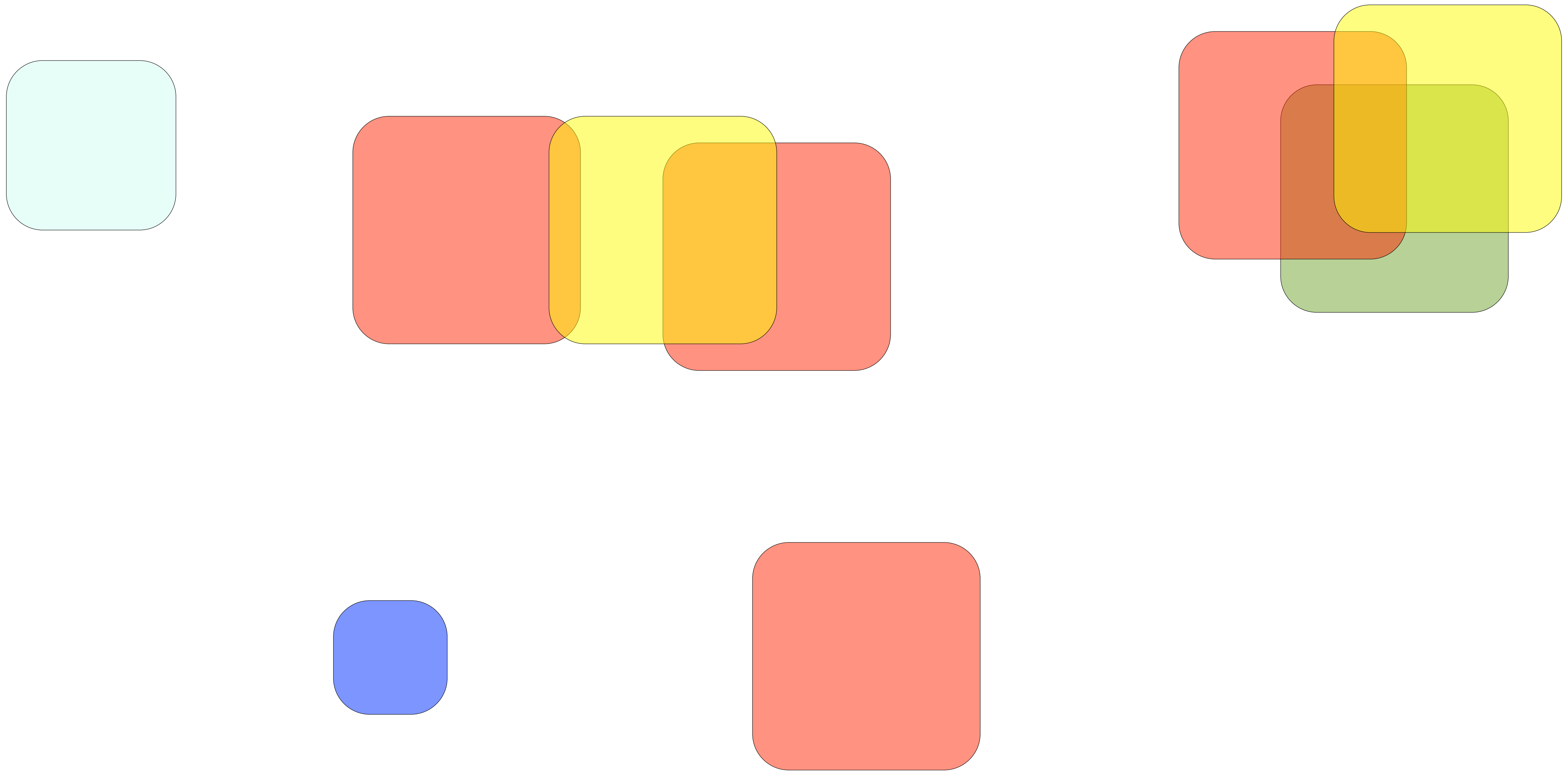
of ... is odd

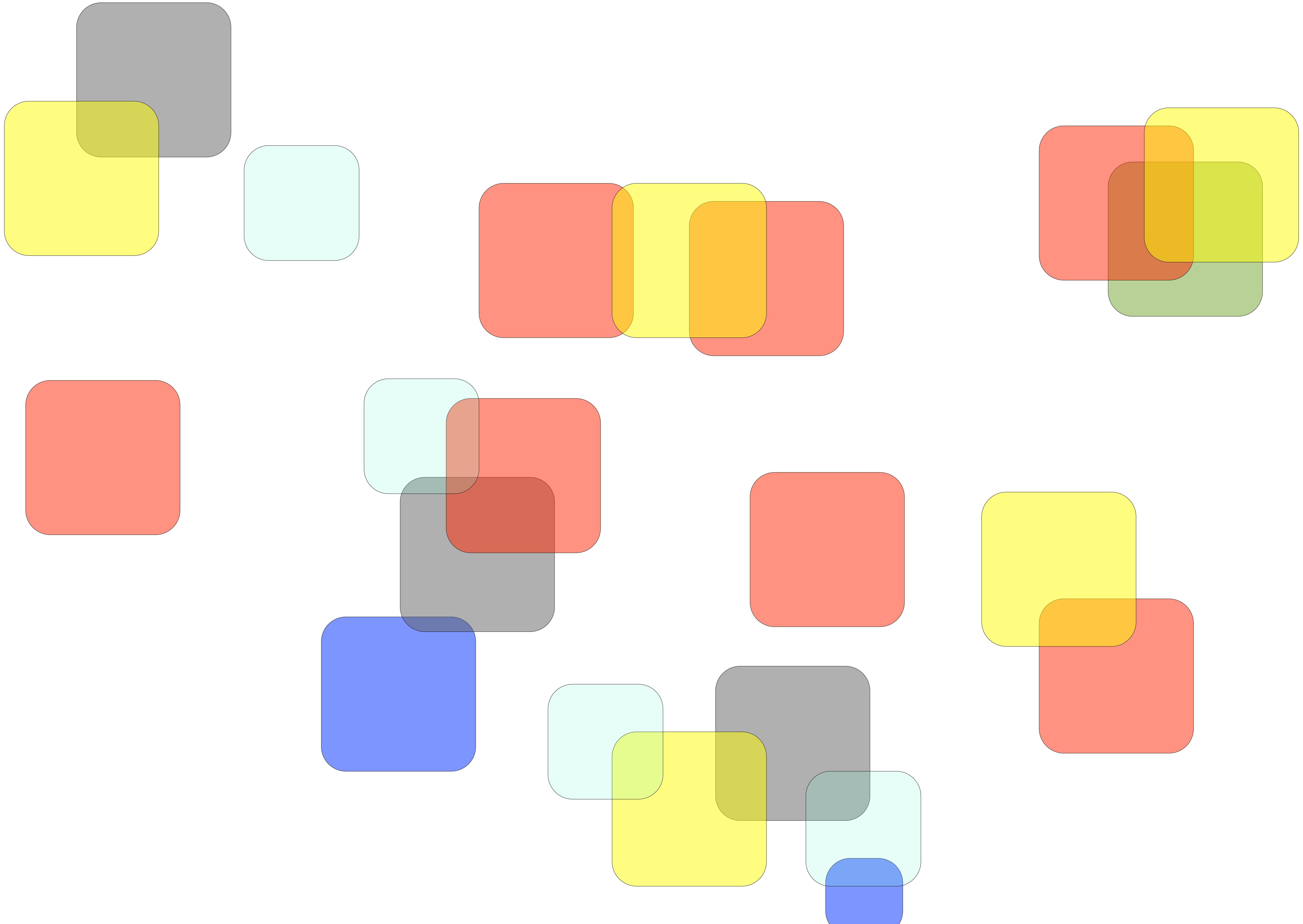


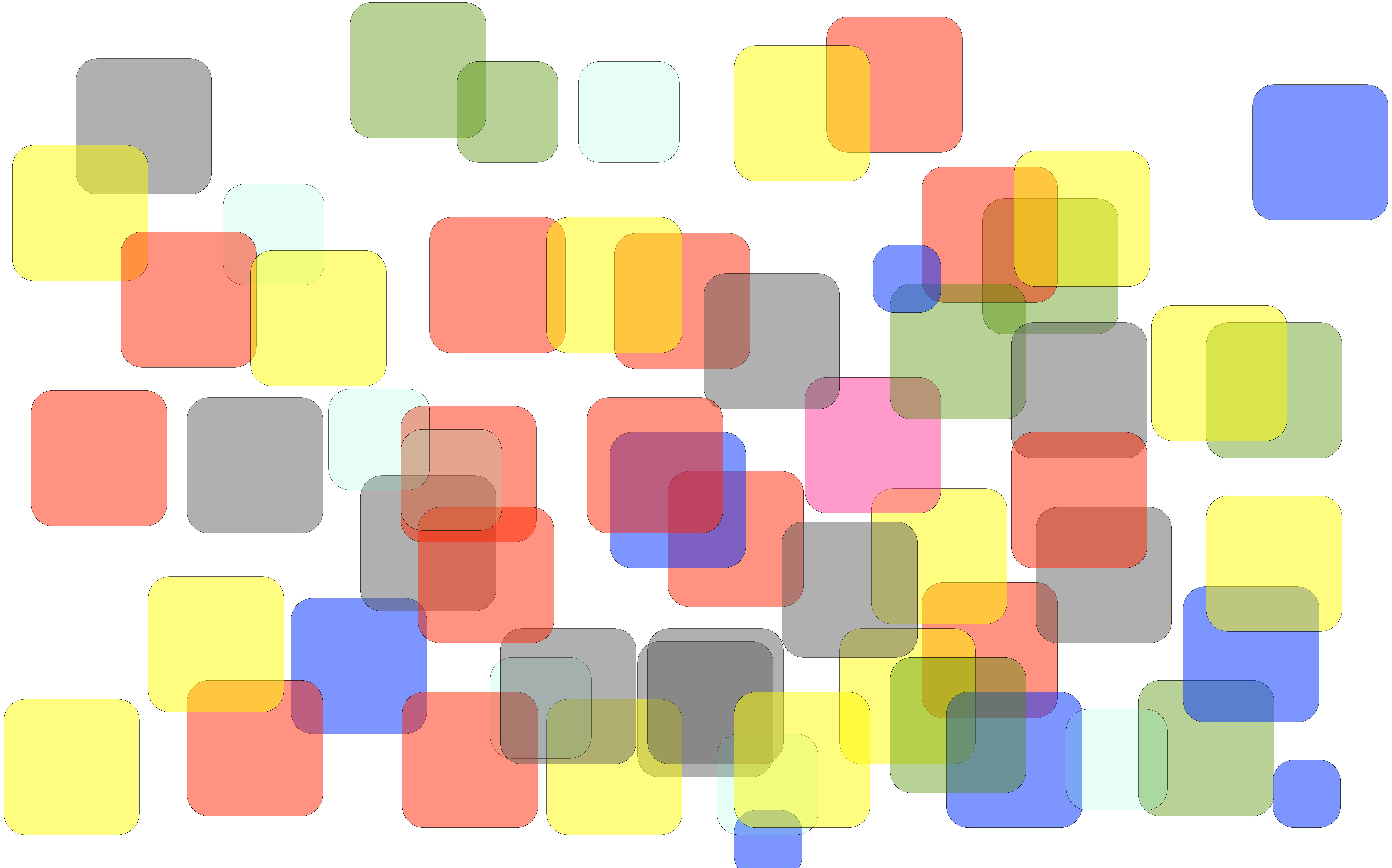
GAME:

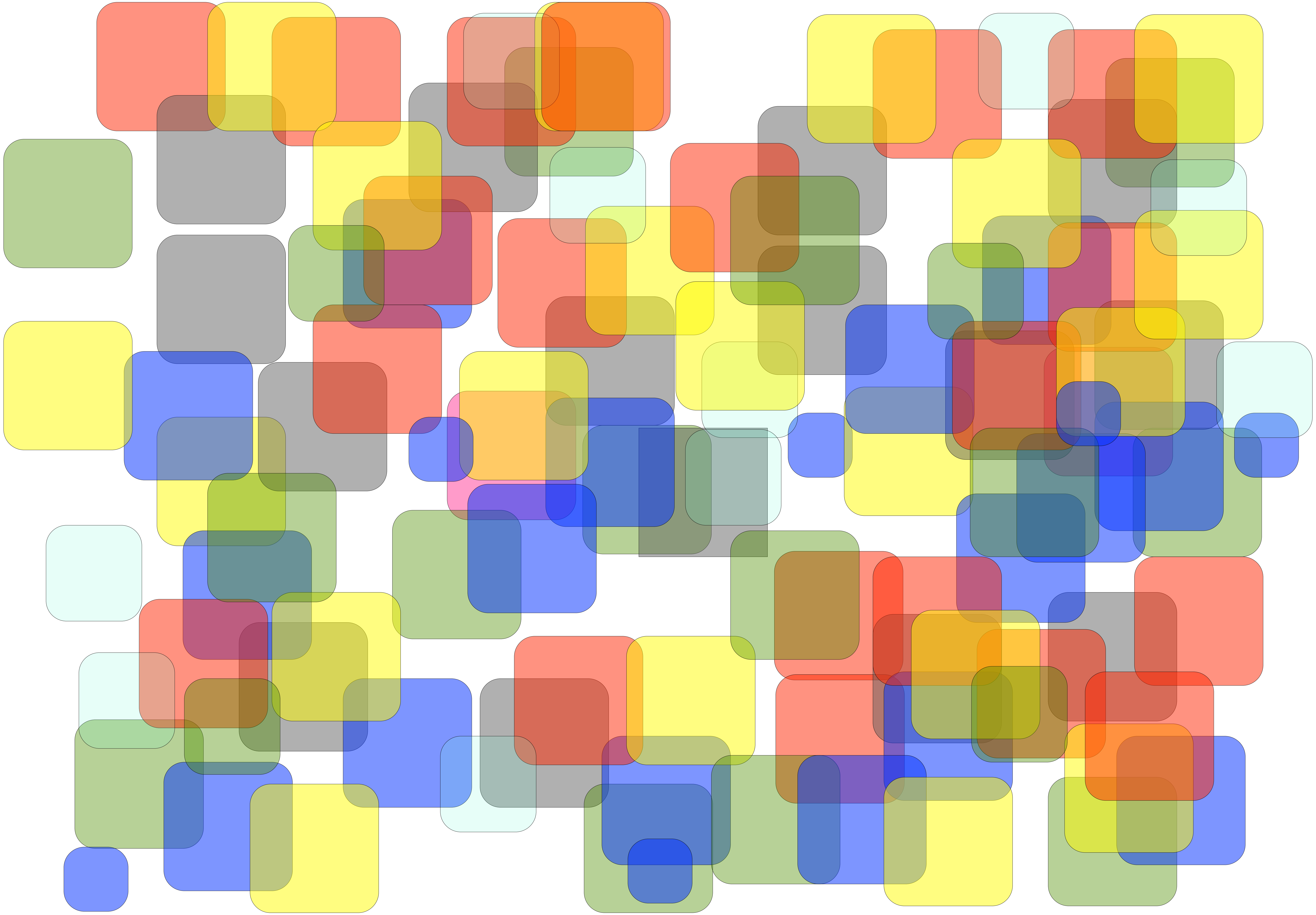
I will pick a sample from either evens or odds, and you will have to guess which one.

READY?









This game is parameterized by its size: i.e, # of boxes.

This game is parameterized by its size: i.e, # of boxes.

evens_n odds_n

As the game size increases, it becomes intractable (for a human) to distinguish b/w
evens and odd

Two ensembles are computationally indistinguishable if it becomes progressively harder for any computer to distinguish the two.

Two ensembles are comp. indistinguishable

$$\{X_n\}_{n \in N} \approx \{Y_n\}_{n \in N}$$

Two ensembles are comp. indistinguishable

$$\{X_n\}_{n \in N} \approx \{Y_n\}_{n \in N}$$

if for all non-uniform p.p.t. alg D,
there exists a negligible function $\epsilon(n)$
such that for all n

Two ensembles are comp. indistinguishable

$$\{X_n\}_{n \in N} \approx \{Y_n\}_{n \in N}$$

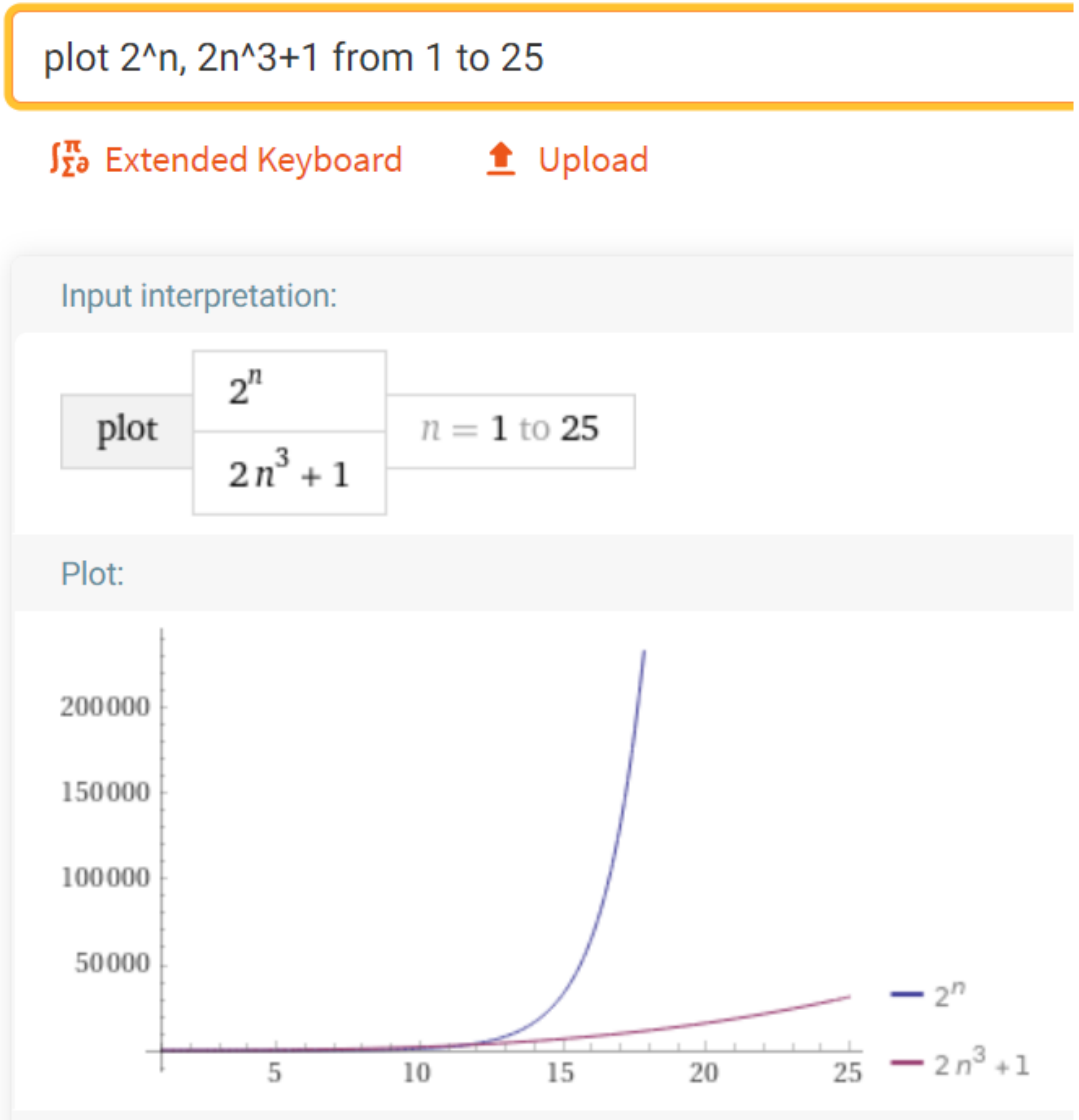
if for all non-uniform p.p.t. alg D,
there exists a negligible function $\epsilon(n)$
such that for all n

$$\left| \Pr\left[t \leftarrow X_n, D(t) = 1\right] - \Pr\left[t \leftarrow Y_n, D(t) = 1\right] \right| \leq \epsilon(n).$$

Polynomial vs. Exponential

- Consider the functions $f(n) = 2n^3 + 1$ and $g(n) = 2^n$
- Which function is "bigger"?

11	2663	2048
12	3457	4096
13	4395	8192
14	5489	16384
20	16001	1,048,576
30	54001	1,073,741,824
35	85751	34,359,738,368



Polynomial vs. Exponential

Polynomial vs. Exponential

- A function is negligible if it approaches 0 faster than any inverse polynomial

Polynomial vs. Exponential

- A function is negligible if it approaches 0 faster than any inverse polynomial
- **Definition:** A function $f\colon \mathbb{N} \to \mathbb{R}$ is a negligible function if for any positive polynomial $p(\cdot)$ there exists N such that for all $n > N$ it holds that

$$f(n) < \frac{1}{p(n)}$$

Polynomial vs. Exponential

- A function is negligible if it approaches 0 faster than any inverse polynomial
- **Definition:** A function $f : \mathbb{N} \to \mathbb{R}$ is a negligible function if for any positive polynomial $p(\cdot)$ there exists N such that for all $n > N$ it holds that

$$f(n) < \frac{1}{p(n)}$$

- For example: 2^{-n}, $2^{-\sqrt{n}}$ and $2^{-\log^2(n)}$ are negligible functions

Polynomial vs. Exponential

- A function is negligible if it approaches 0 faster than any inverse polynomial
- **Definition:** A function $f : \mathbb{N} \rightarrow \mathbb{R}$ is a negligible function if for any positive polynomial $p(\cdot)$ there exists N such that for all $n > N$ it holds that

$$f(n) < \frac{1}{p(n)}$$

- For example: 2^{-n}, $2^{-\sqrt{n}}$ and $2^{-\log^2(n)}$ are negligible functions
- $1/2$, $1/\log^2(n)$ and $1/n^5$ are non-negligible functions

pseudo-random

An algorithm $\{G\}$ is said to be

pseudo-random

pseudo-random

$$\{k \leftarrow \{0,1\}^n : G(k)\}_n \overset{\text{if}}{\approx} \{U_\ell\}_{\ell(n)}$$

"The output of the PRG, when evaluated on seeds of increasing length"

"Truly uniform strings of the same length as the output of the PRG"

An algorithm $\{G\}$ is said to be

pseudo-random

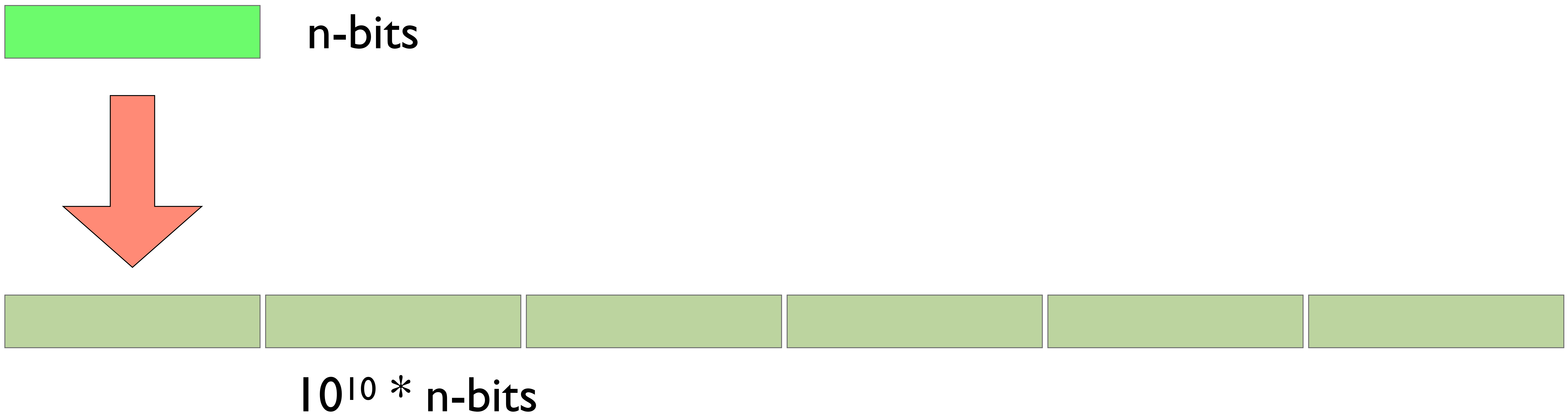
if

$$\{k \leftarrow \{0,1\}^n : G(k)\}_n \approx \{U_\ell\}_{\ell(n)}$$

"The output of the PRG, when evaluated on seeds of increasing length"

"Truly uniform strings of the same length as the output of the PRG"

Original goal



Pseudo-random generator

A family of functions $G : \{0,1\}^n \to \{0,1\}^m$

is a pseudo-random generator if

Pseudo-random generator

A family of functions $G : \{0,1\}^n \rightarrow \{0,1\}^m$

is a pseudo-random generator if

G can be computed in p.p.t.

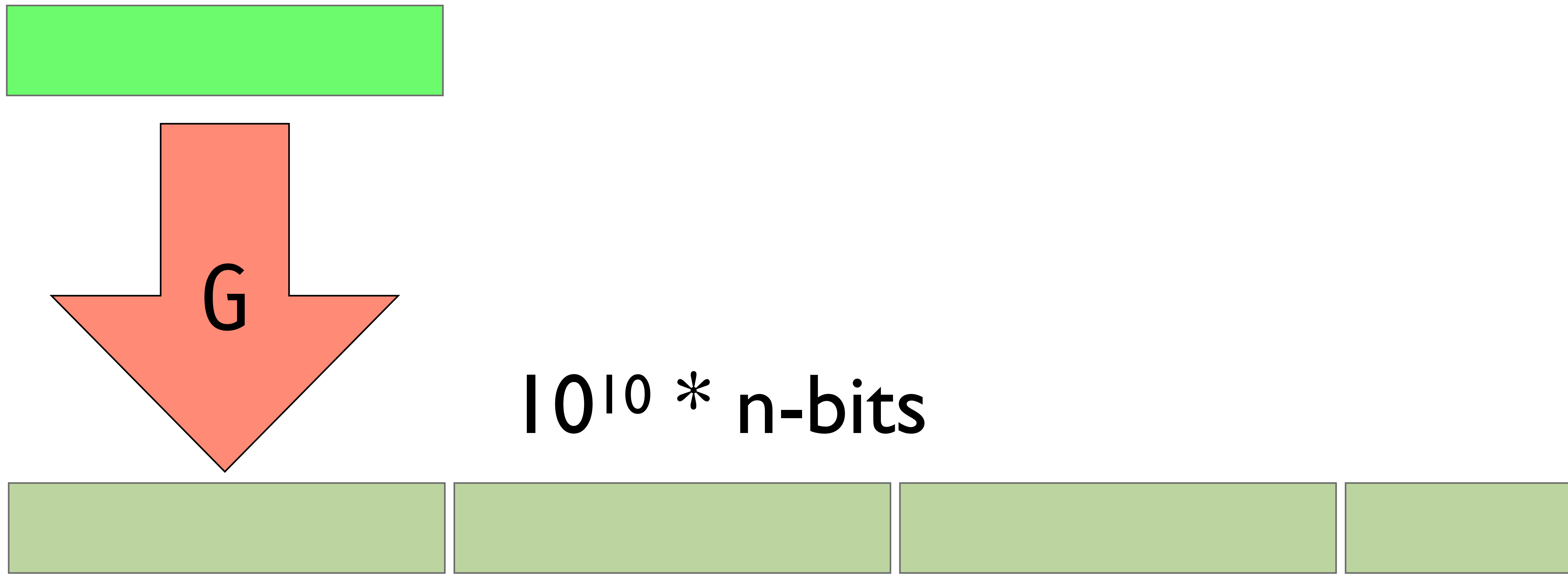
$|G(x)| > \ell(|x|)$ for some $\ell(y) > y$

$\{x \leftarrow U_n \ : \ G(x)\}_{n\in\mathbb{N}}$ is pseudo-random

Truly random

$$U_n$$

Pseudo-randomness



The same notion of indistinguishability helps us define security for symmetric encryption.

Perfect secrecy

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{M}, \mathcal{K})$

is said to be PERFECTLY SECRET if

$$\forall m_1, m_2 \in \mathcal{M}, \forall c$$

$$\Pr[k \leftarrow \mathsf{Gen} \;:\; \mathsf{Enc}_k(m_1) = c]$$

$$=$$

$$\Pr[k \leftarrow \mathsf{Gen} \;:\; \mathsf{Enc}_k(m_2) = c]$$

Perfect secrecy

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{M}, \mathcal{K})$

is said to be PERFECTLY SECRET if

$$\forall m_1, m_2 \in \mathcal{M}, \forall c$$

$$\Pr[k \leftarrow \mathsf{Gen} \; : \; \mathsf{Enc}_k(m_1) = c]$$

$$=$$

$$\Pr[k \leftarrow \mathsf{Gen} \; : \; \mathsf{Enc}_k(m_2) = c]$$

Indistinguishable secrecy

$$\Pr[k \leftarrow \mathsf{Gen} \; : \; \mathsf{Enc}_k(m_1) = c]$$

$$\sim$$

$$\Pr[k \leftarrow \mathsf{Gen} \; : \; \mathsf{Enc}_k(m_2) = c]$$

“So close that no efficient computer can distinguish”

computational secrecy

$$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{M}, \mathcal{K})$$

is said to be computationally secure if

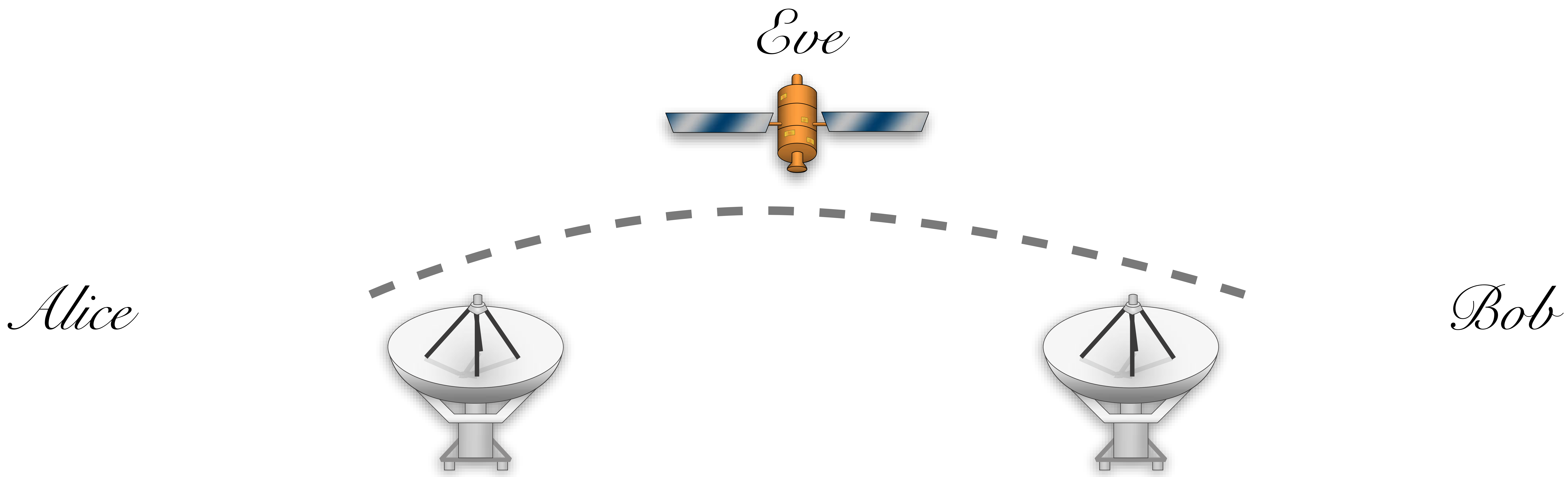
$$\forall m_1, m_2 \in \mathcal{M} \text{ s.t. } |m_1| = |m_2|, \forall c$$

$$\{k \leftarrow \mathrm{Gen}(1^n) \;:\; Enc_k(m_1)\}$$

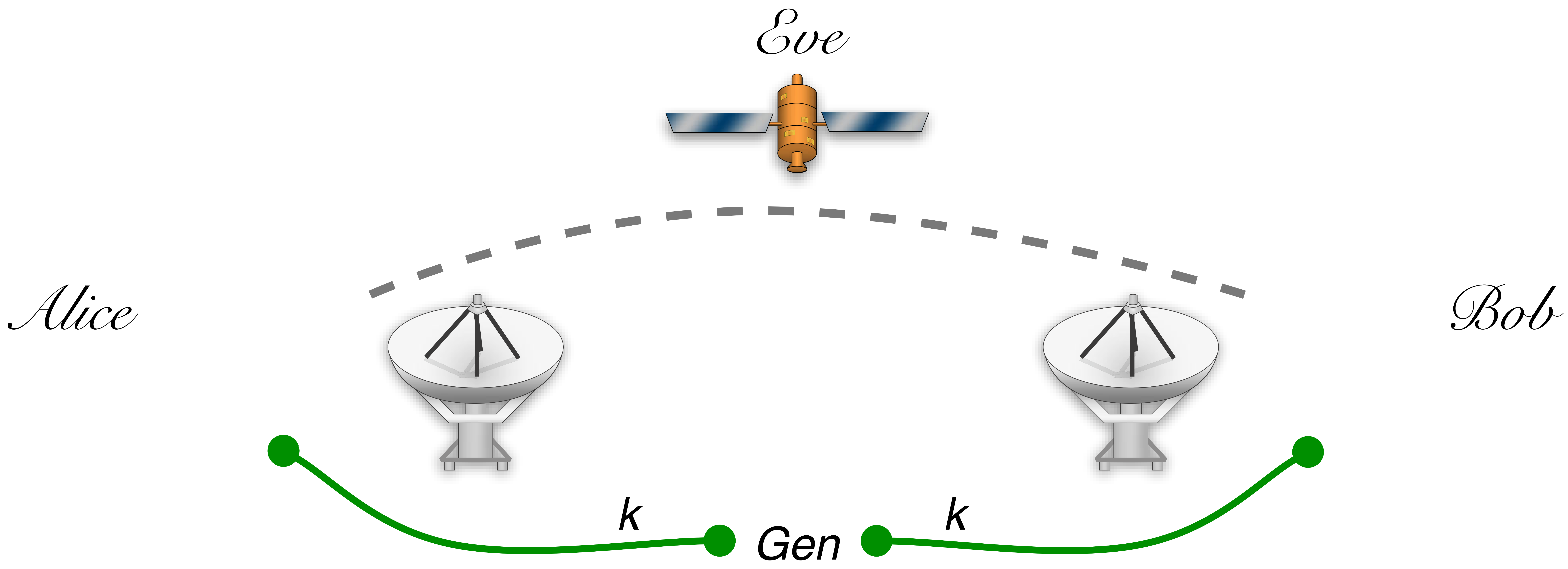
$$\approx$$

$$\{k \leftarrow \mathrm{Gen}(1^n) \;:\; Enc_k(m_2)\}$$

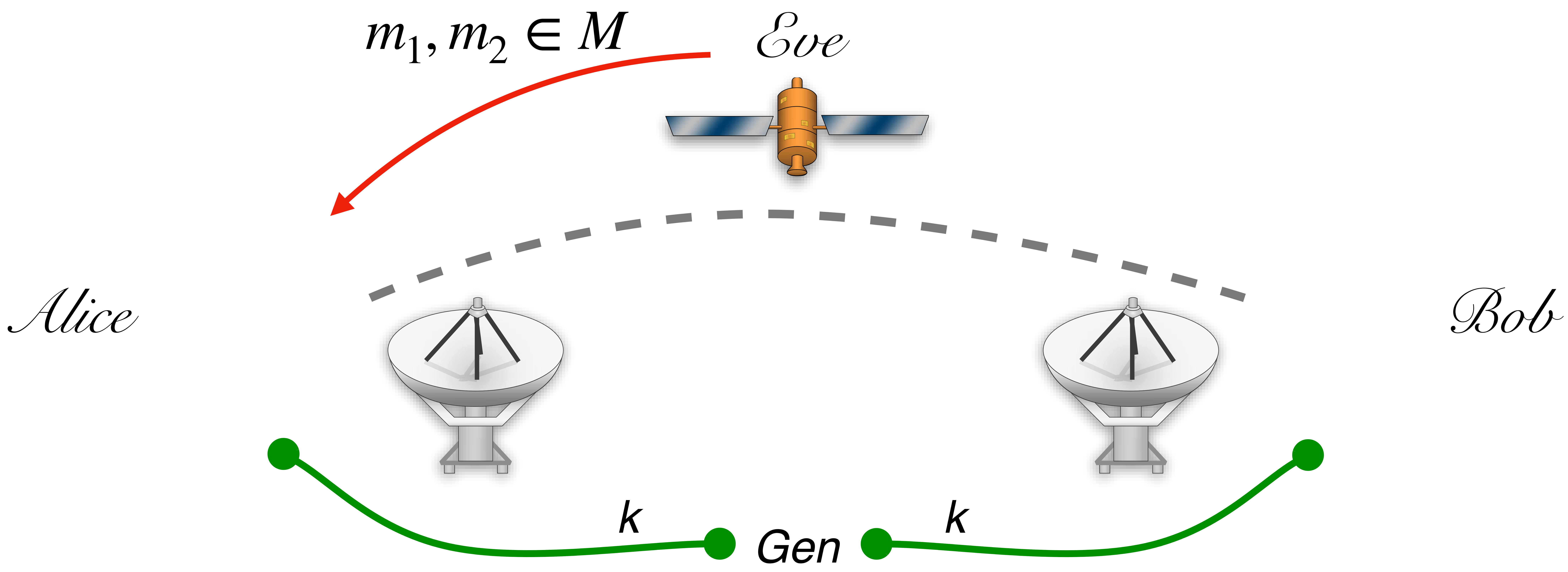
Simple security game for Enc



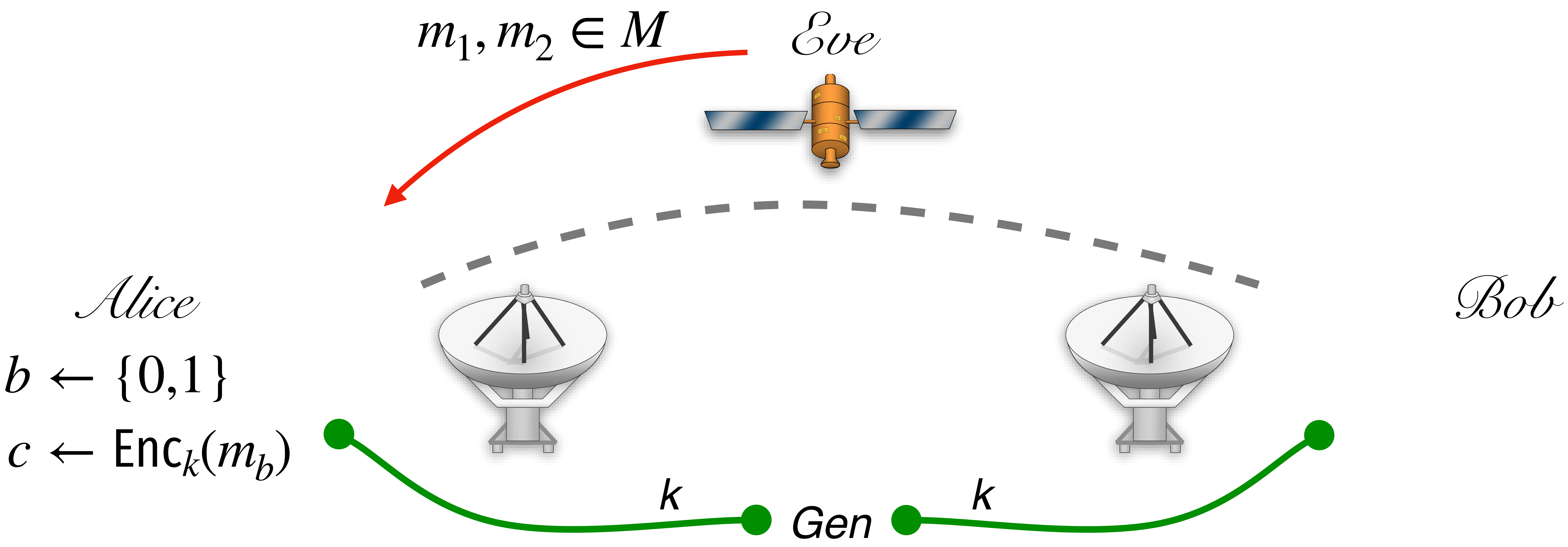
Simple security game for Enc



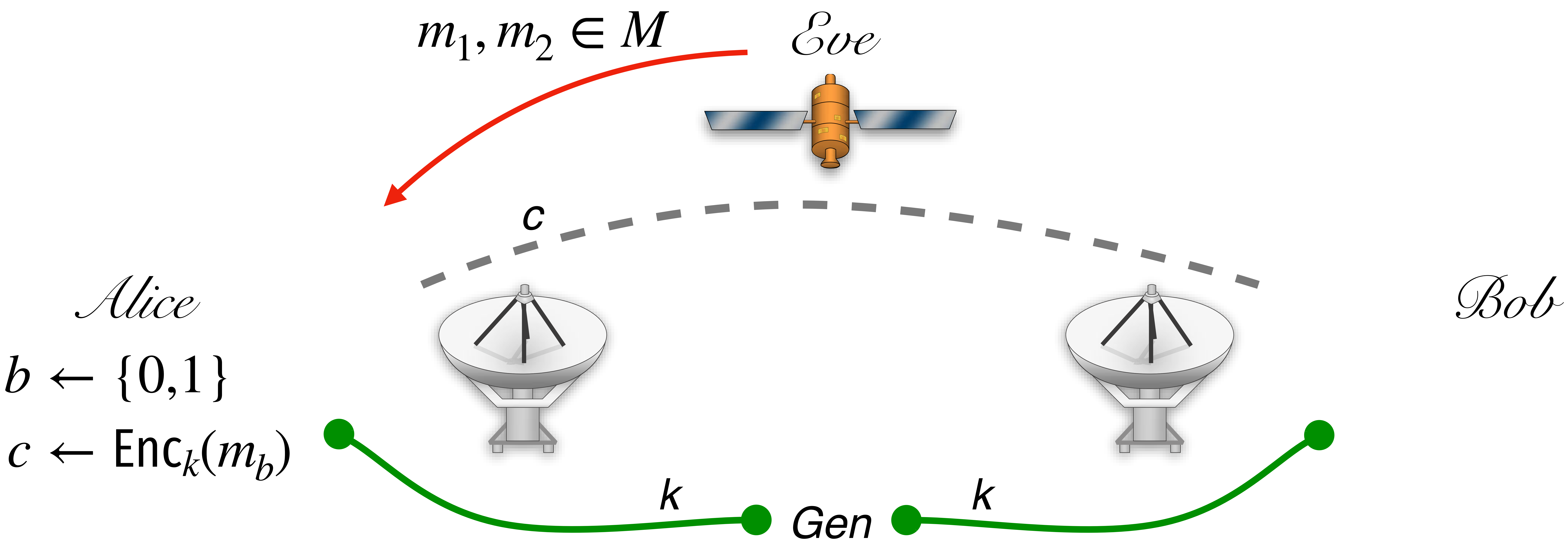
Simple security game for Enc



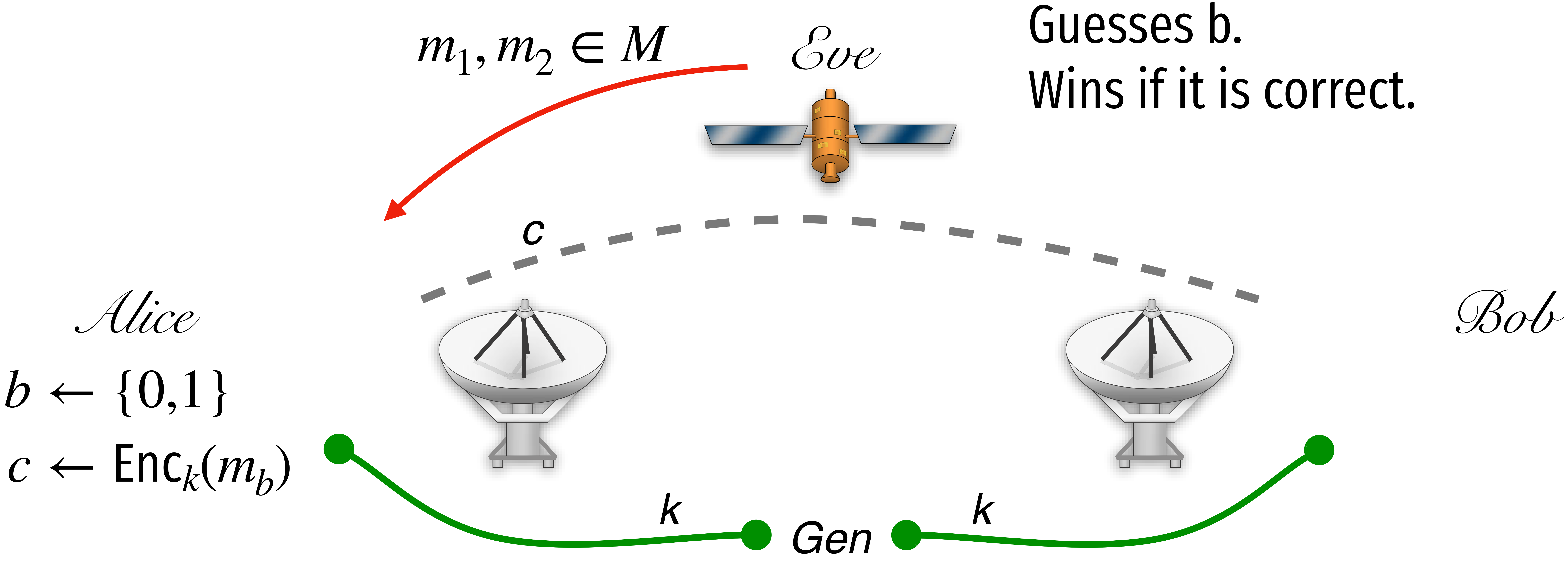
Simple security game for Enc



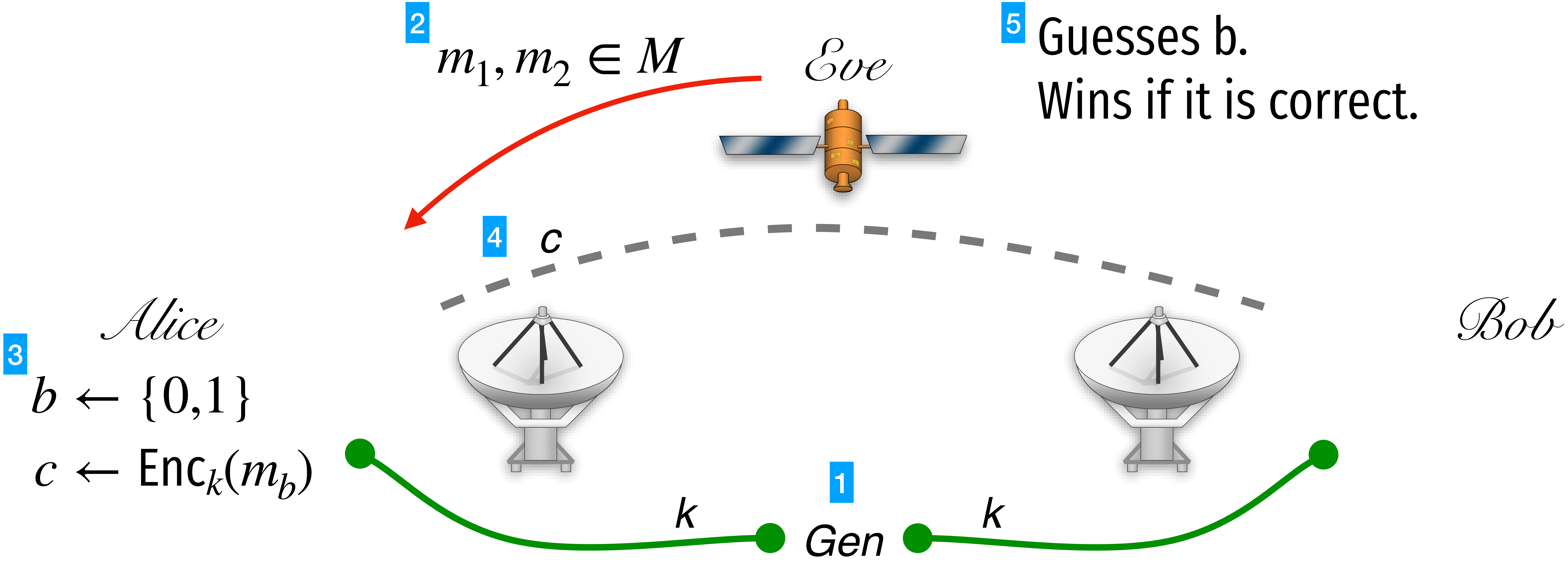
Simple security game for Enc



Simple security game for Enc



Simple security game for Enc



Given a secure PRG,
then (Gen, Enc, Dec) described earlier is secure in this game.

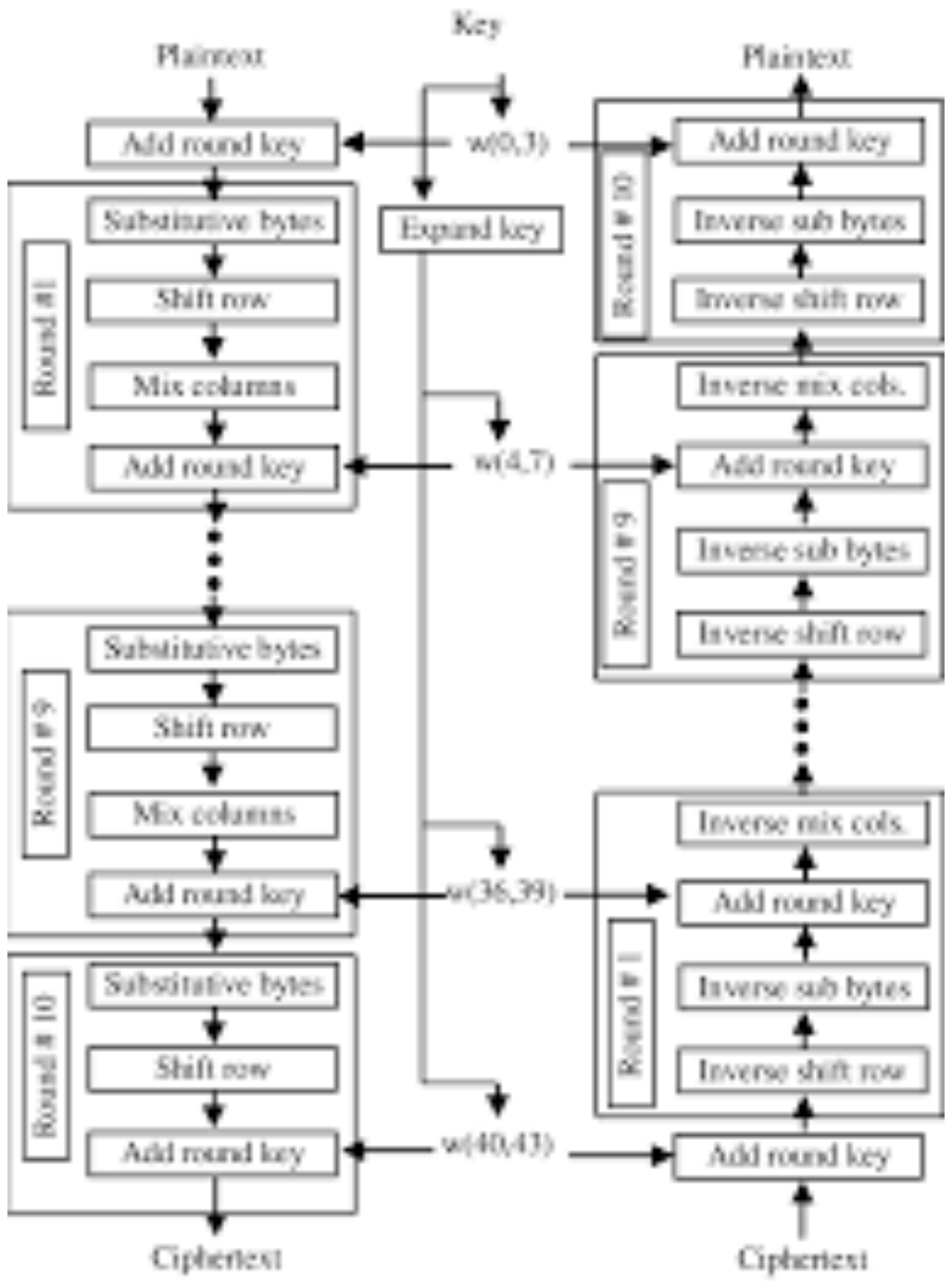
How can we build pseudo-random generators and symmetric encryption?

Two ways to build PRGS + Symmetric Enc

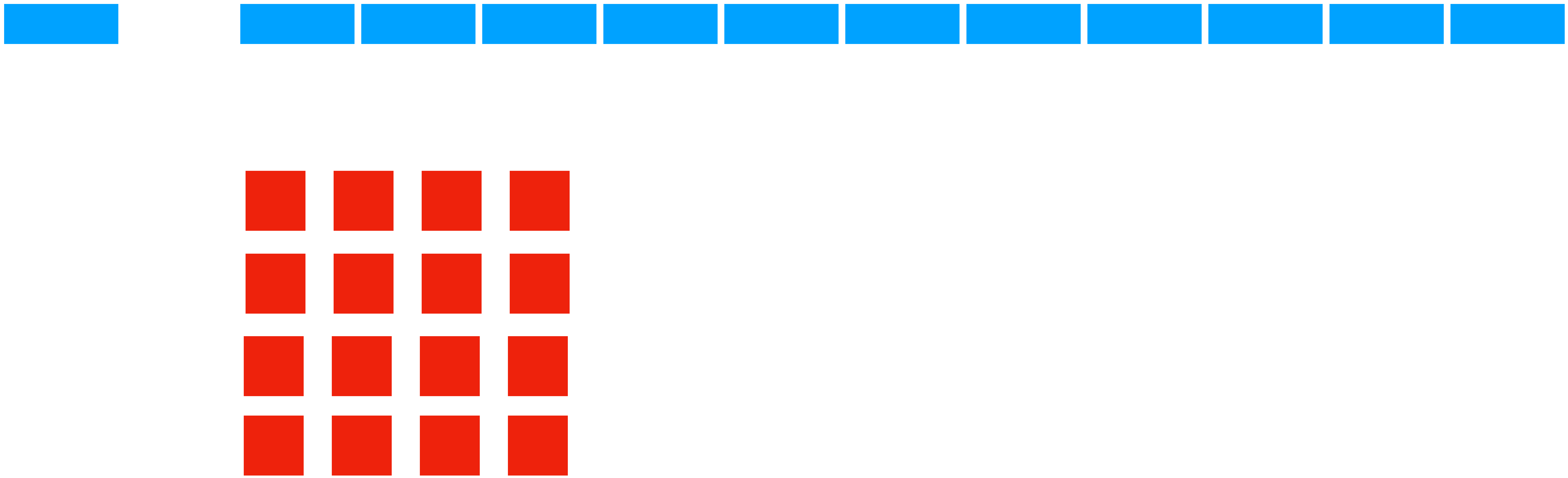
Principled

Heuristic

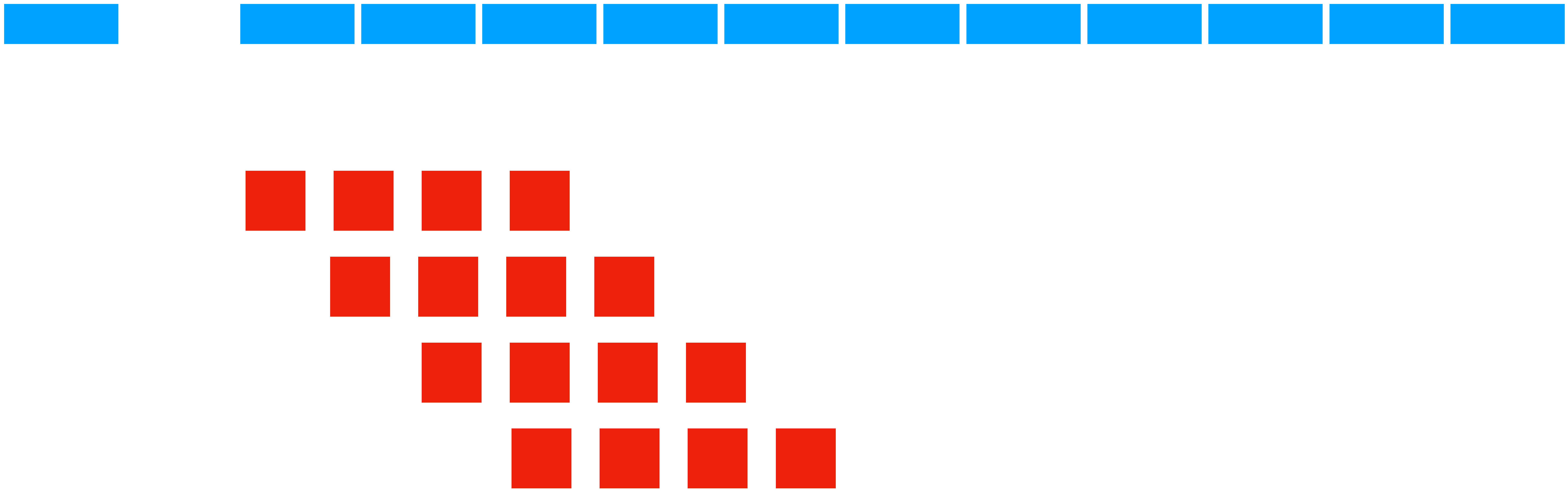
Modern version: AES



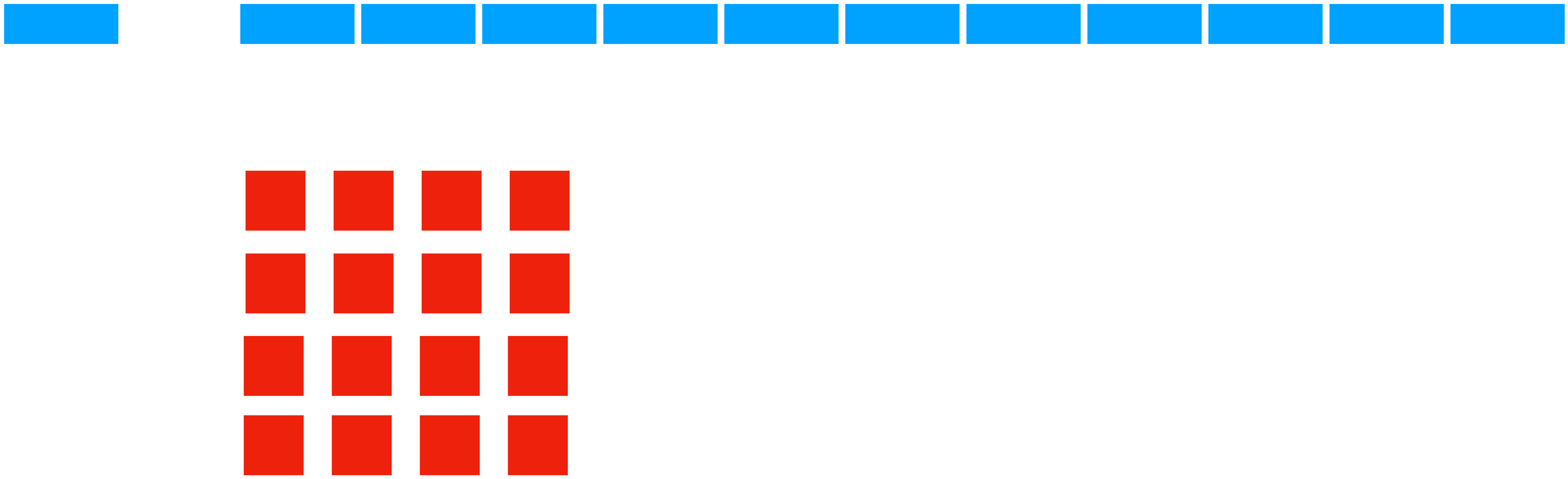
AES(k,m)



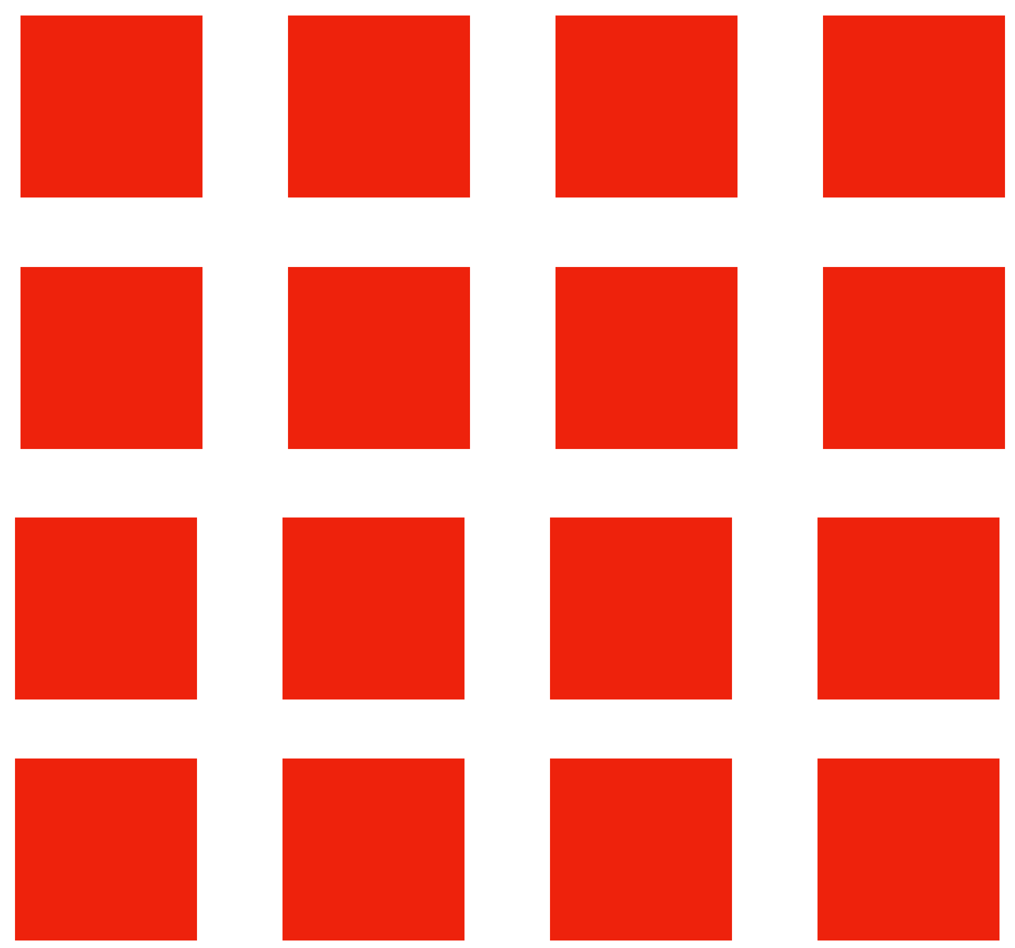
AES(k,m)



AES(k,m)



AES(k,m)



Add round key 1 into m

For i=1...9:

- SubBytes: apply a map to all bytes
- ShiftRows: permute the bytes
- MixColumns: permute columns
- AddRoundKey i+1

SubBytes: apply a map to all bytes

ShiftRows: permute the bytes

AddRoundKey i+1

Main security comes from s-box

AES S-Box

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

The column is determined by the least significant nibble, and the row by the most significant nibble. For example, the value `0x9a` is converted into `0xb8`.

AES is very fast.

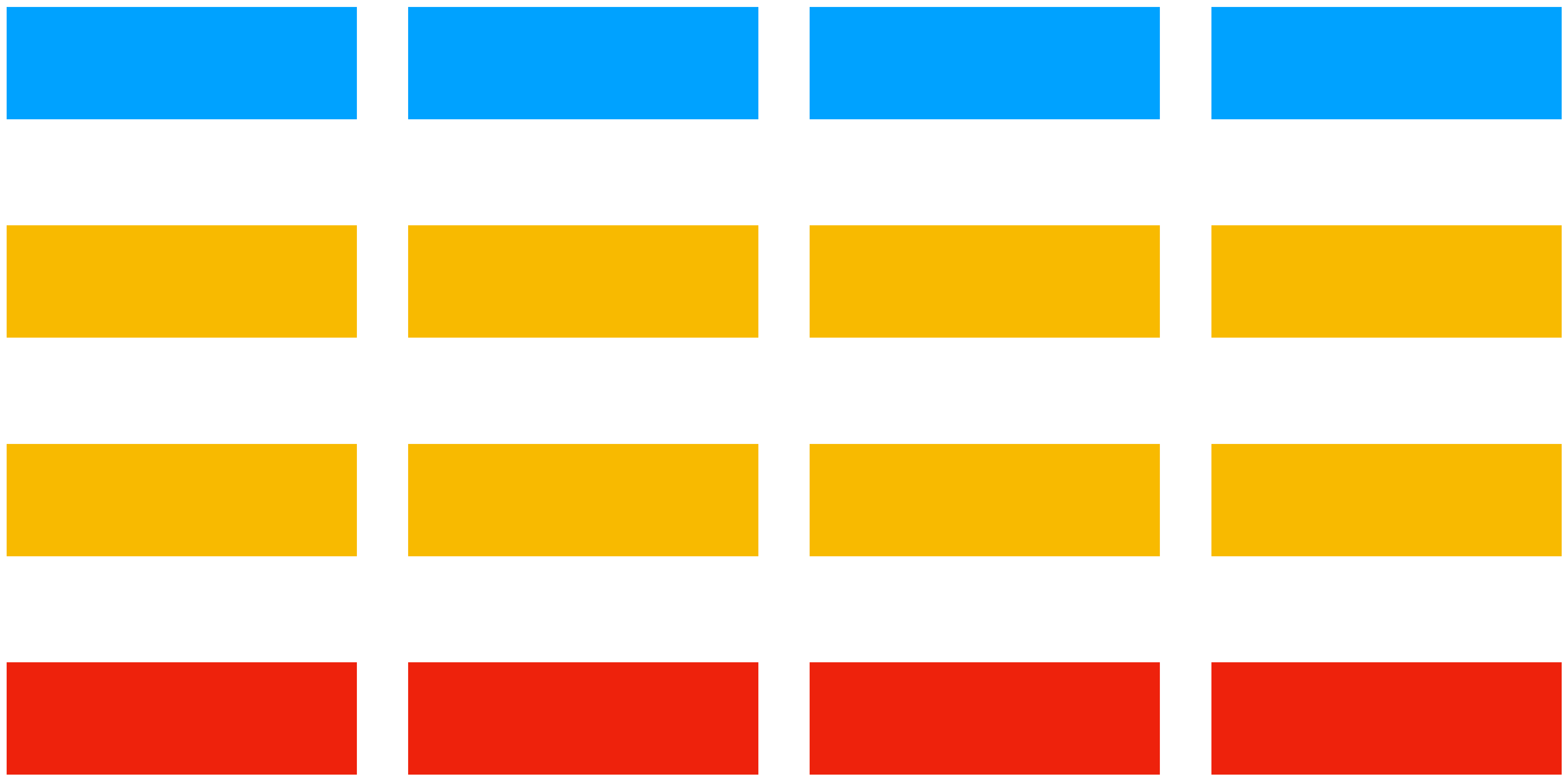
https://calomel.org/aesni_ssl_performance.html

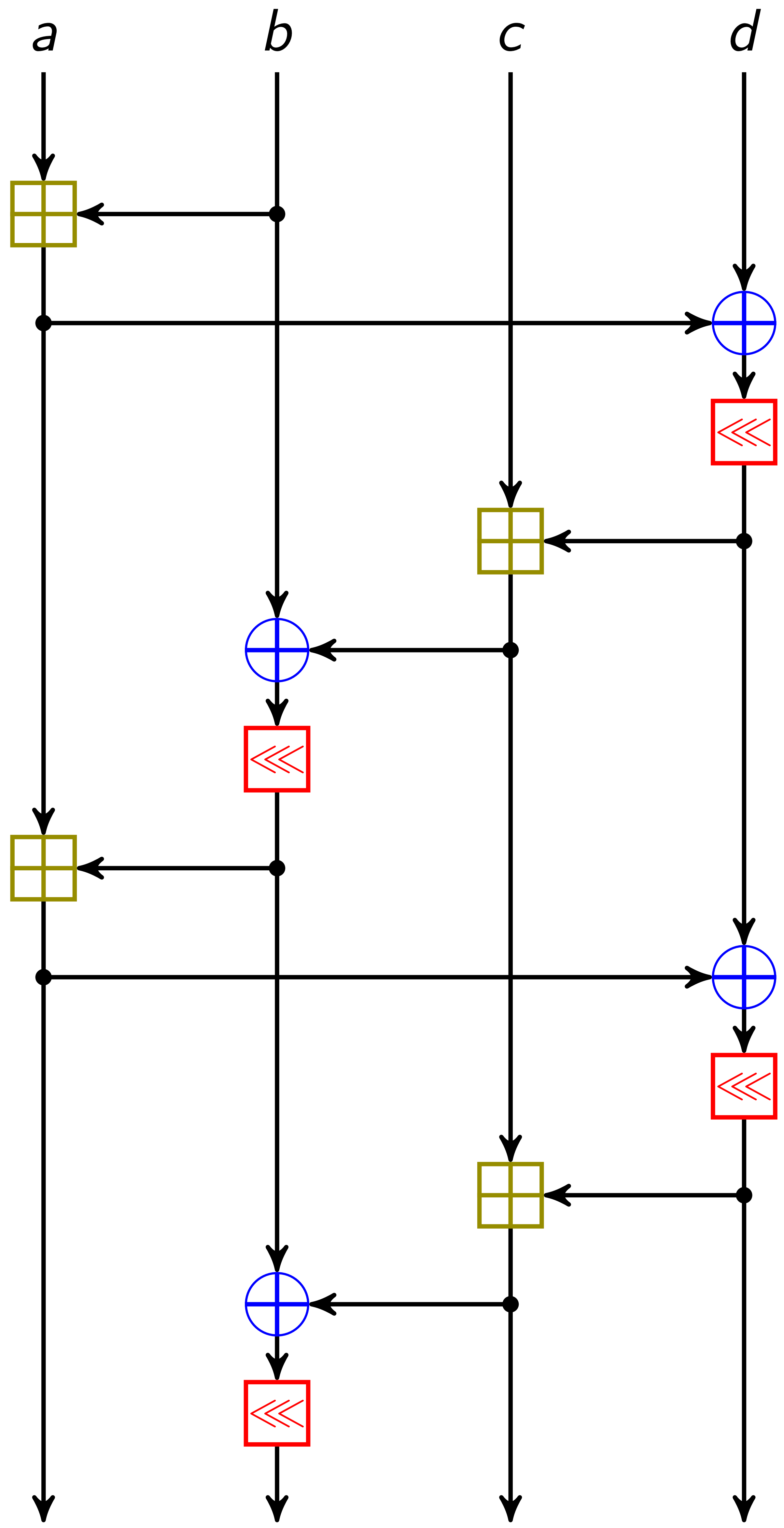
Cipher Performance per CPU core

AES Performance per CPU core for TLS v1.2 Ciphers
(Higher is Better, Speeds in Megabytes per Second)

	ChaCha20	AES-128-GCM	AES-256-GCM	AES-128-CBC	AES-256-CBC	Total Score
AMD Ryzen 7 1800X	573	3006	2642	1513	1101	= 8835
Intel W-2125	565	2808	2426	1698	1235	= 8732
Intel i7-6700	585	2607	2251	1561	1131	= 8135
AMD EPYC 7551	355	2213	1962	1114	811	= 6455
Intel i5-6500	410	1729	1520	1078	783	= 5520
Intel i7-4750HQ	369	1556	1353	688	499	= 4465
AMD FX 8350	367	1453	1278	716	514	= 4328
AMD FX 8150	347	1441	1273	716	515	= 4292
Intel E5-2650 v4	404	1479	1286	652	468	= 4289
Intel i7-2700K	382	1353	1212	763	552	= 4262
Intel i7-3840QM	373	1279	1143	725	520	= 4040
Intel i5-2500K	358	1274	1140	728	522	= 4022
AMD FX 6100	326	1344	1186	671	481	= 4008
AMD A10-7850K	321	1303	1176	685	499	= 3984
AMD A8-7600 Kaveri	306	1246	1108	648	470	= 3778
Intel E5-2640 v3	303	1286	1126	585	419	= 3719
AMD Opteron 6380	293	1203	1063	589	423	= 3571
AMD Opteron 6378	282	1138	986	561	406	= 3373
AMD Opteron 6274	232	1054	926	524	376	= 3112
Intel Xeon E5-2630	247	962	864	541	394	= 3008
Intel Xeon E5645	262	817	717	727	524	= 3047

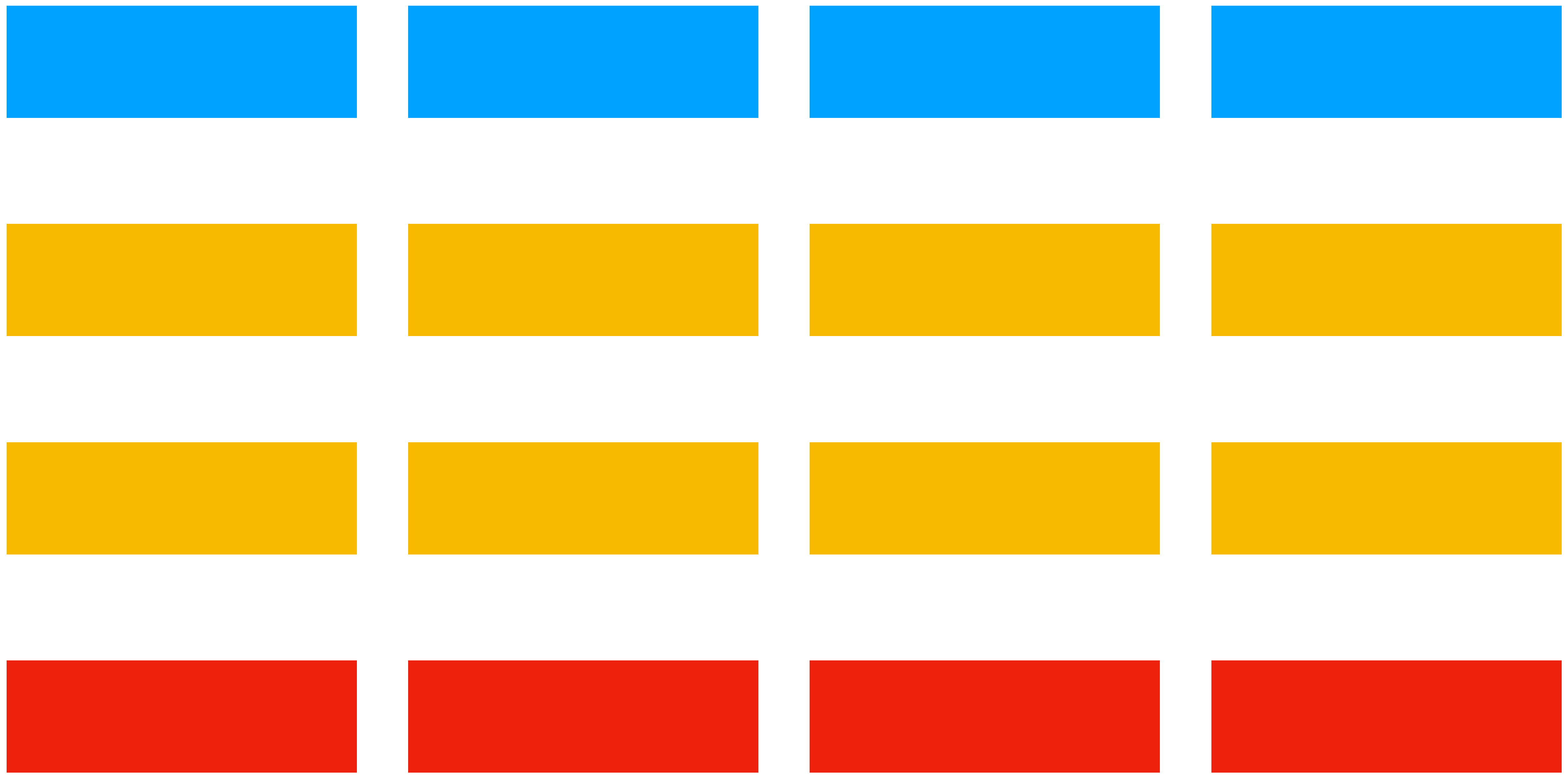
Efficiency: chacha20 (a stream cipher)




a
b
c
d

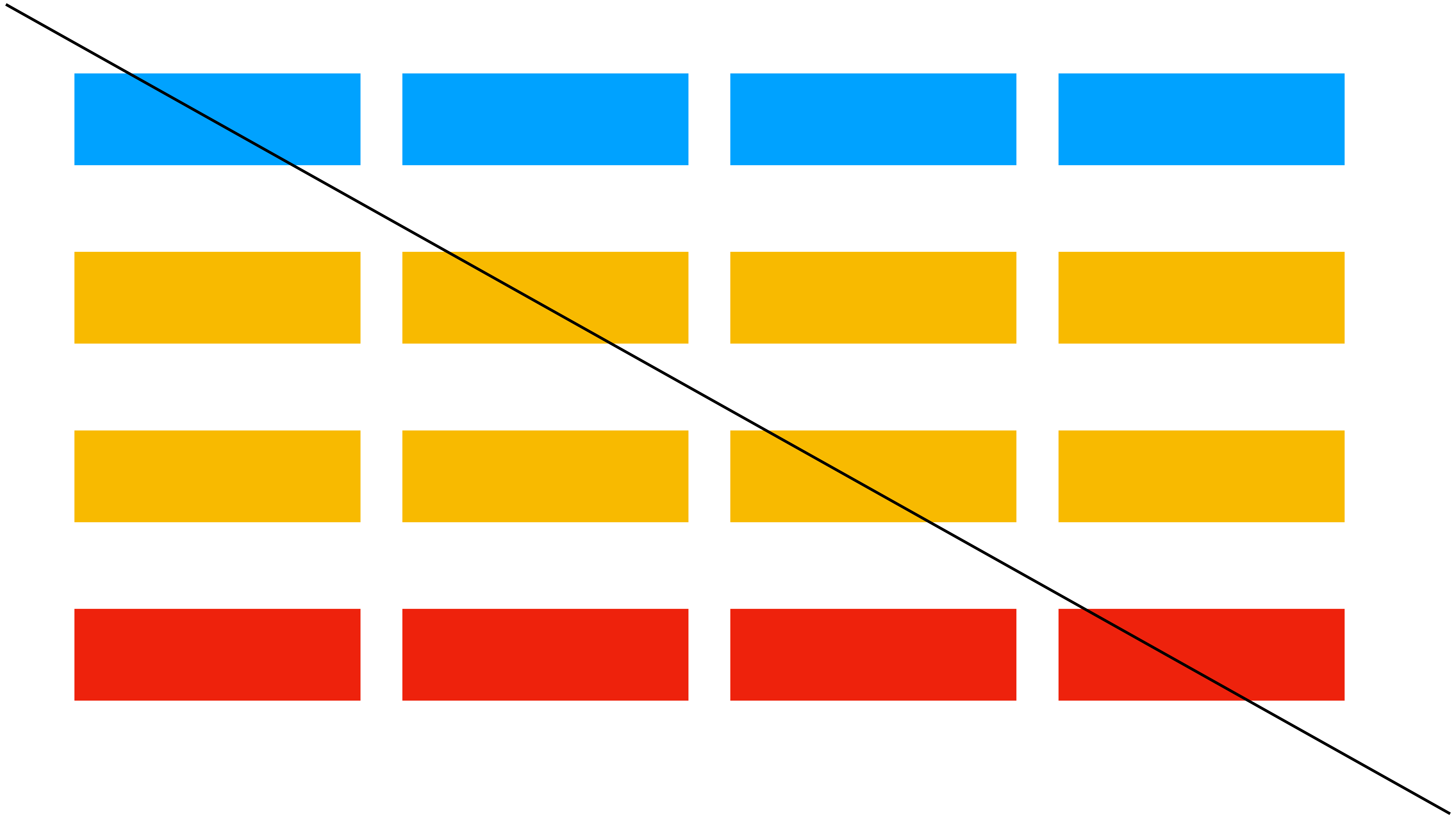
Efficiency: chacha20 (a stream cipher)

Columns



Efficiency: chacha20 (a stream cipher)

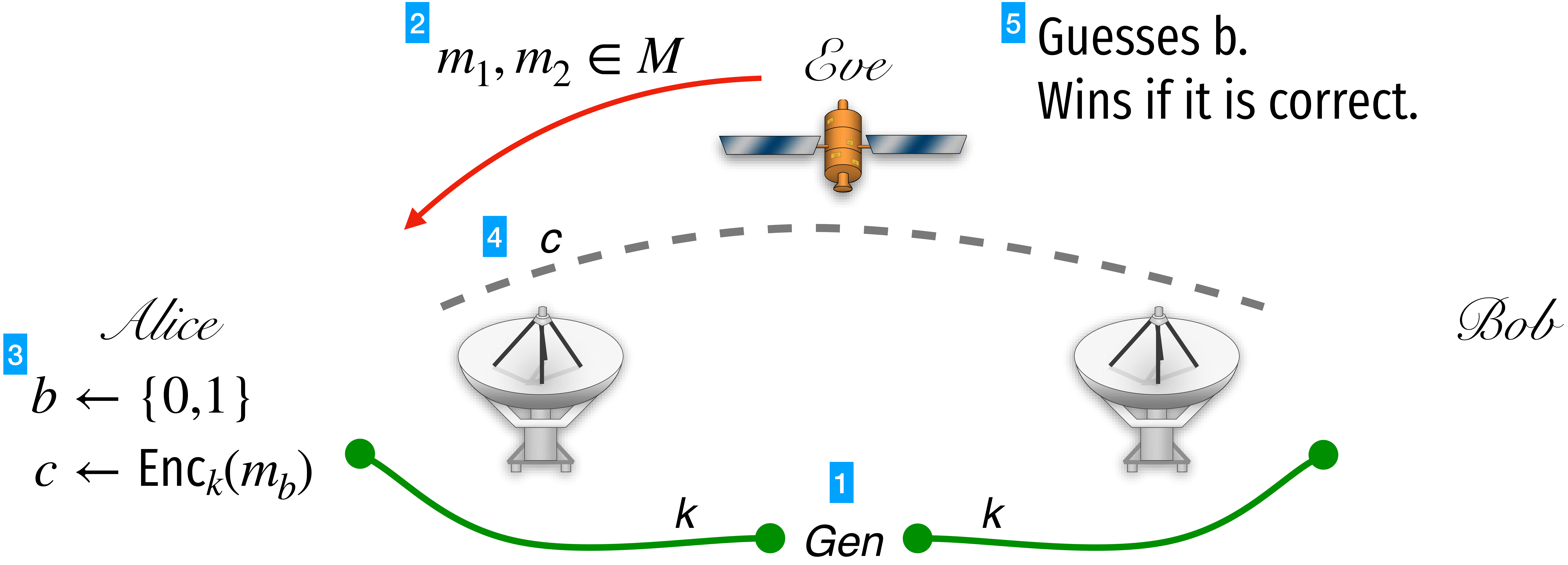
Diagonals

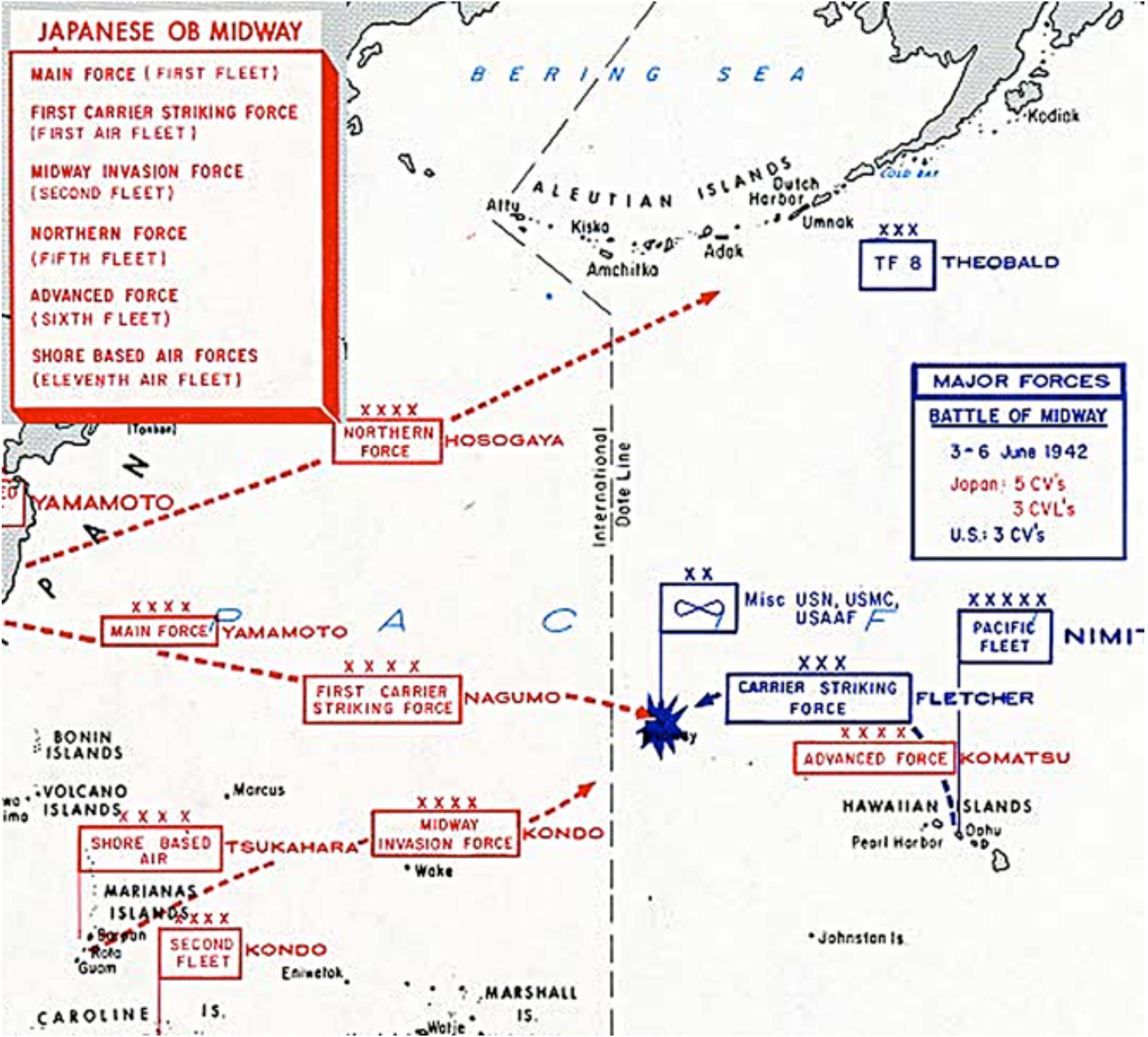


Chacha20

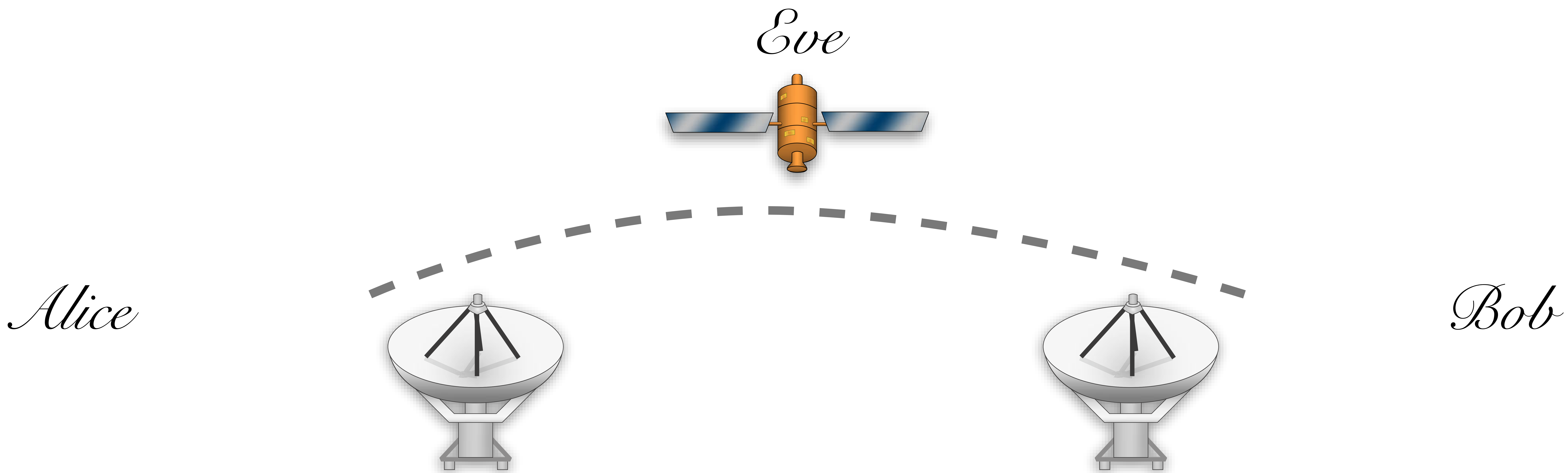
```
void chacha_block(uint32_t out[16], uint32_t const in[16])
{
  int i;
  uint32_t x[16];

  for (i = 0; i < 16; ++i)
    x[i] = in[i];
  // 10 loops × 2 rounds/loop = 20 rounds
  for (i = 0; i < ROUNDS; i += 2) {
    // Odd round
    QR(x[0], x[4], x[ 8], x[12]); // column 0
    QR(x[1], x[5], x[ 9], x[13]); // column 1
    QR(x[2], x[6], x[10], x[14]); // column 2
    QR(x[3], x[7], x[11], x[15]); // column 3
    // Even round
    QR(x[0], x[5], x[10], x[15]); // diagonal 1 (main diagonal)
    QR(x[1], x[6], x[11], x[12]); // diagonal 2
    QR(x[2], x[7], x[ 8], x[13]); // diagonal 3
    QR(x[3], x[4], x[ 9], x[14]); // diagonal 4
  }
  for (i = 0; i < 16; ++i)
    out[i] = x[i] + in[i];
}
```

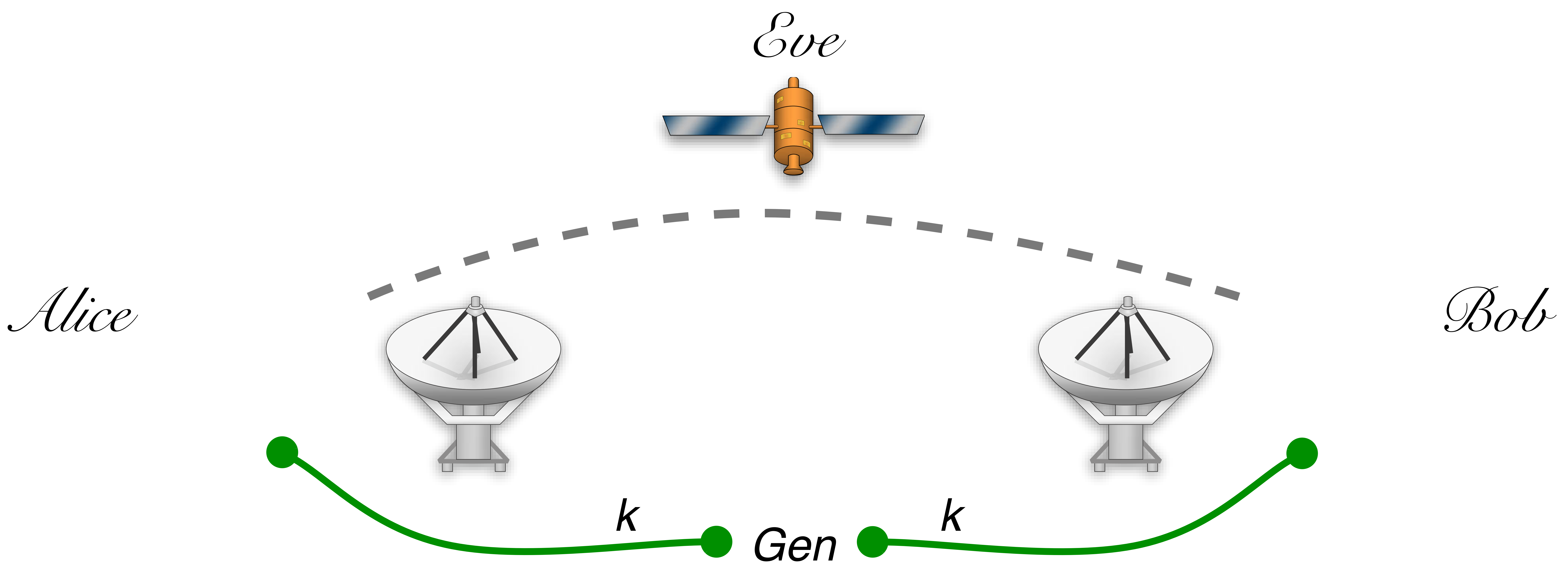
Is this game strong enough to capture all feasible attacks?



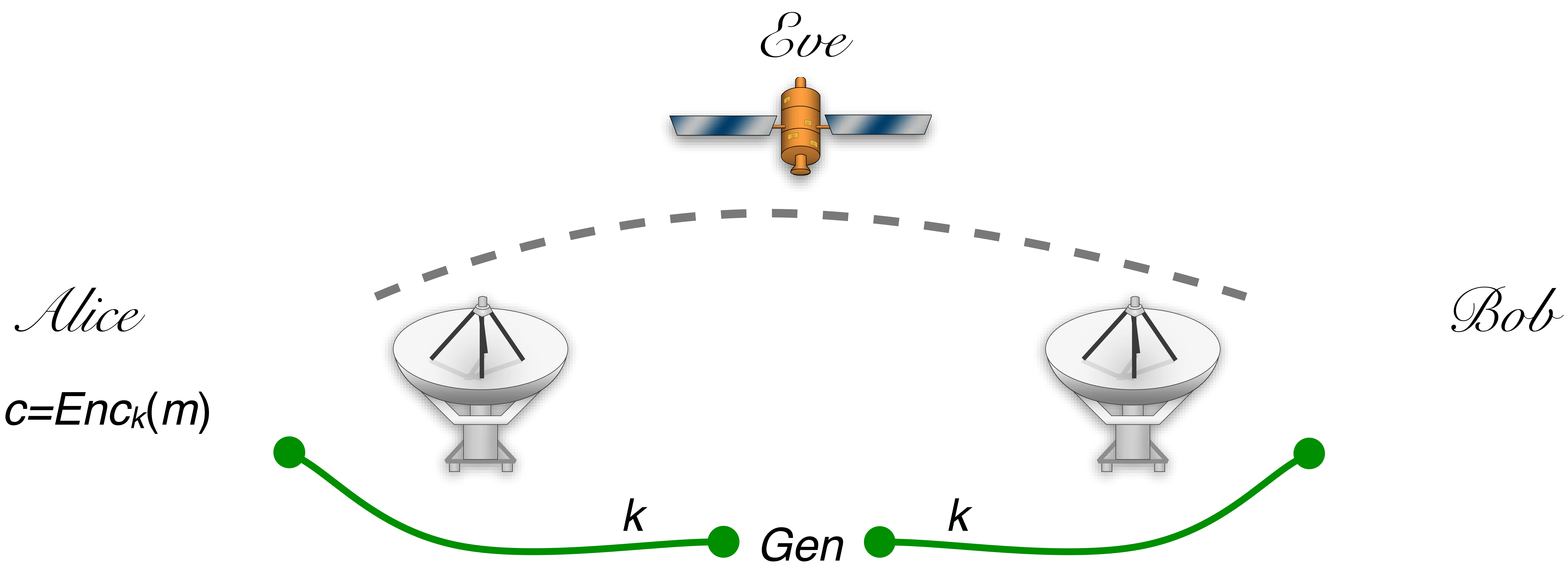


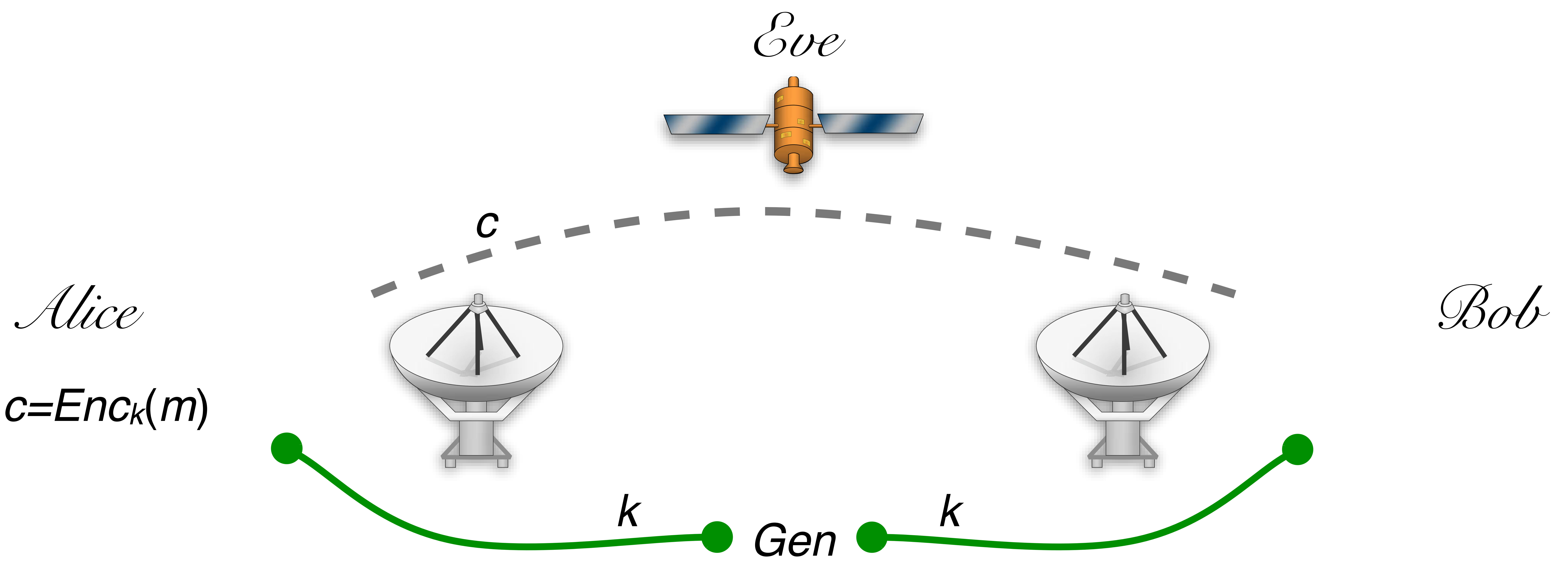
JAPANESE OB MIDWAY
MAIN FORCE (FIRST FLEET)
FIRST CARRIER STRIKING FORCE (FIRST AIR FLEET)
MIDWAY INVASION FORCE (SECOND FLEET)
NORTHERN FORCE (FIFTH FLEET)
ADVANCED FORCE (SIXTH FLEET)
SHORE BASED AIR FORCES (ELEVENTH AIR FLEET)
BERING SEA
Kodiak
ALEUTIAN ISLANDS
Dutch Harbor
Attu
Kiska
Umnak
Adak
Amchitka
TF 8 THEOBALD
NORTHERN FORCE HOSOGAYA
YAMAMOTO
International Date Line
MAJOR FORCES
BATTLE OF MIDWAY
3-6 June 1942
Japan: 5 CV's
3 CVL's
U.S.: 3 CV's
Misc USN, USMC, USAAF
PACIFIC FLEET NIMIT
MAIN FORCE YAMAMOTO
PACIFIC
FIRST CARRIER STRIKING FORCE NAGUMO
CARRIER STRIKING FORCE FLETCHER
ADVANCED FORCE KOMATSU
BONIN ISLANDS
VOLCANO ISLANDS
Marcus
HAWAIIAN ISLANDS
Oahu
Pearl Harbor
SHORE BASED AIR TSUKAHARA
MIDWAY INVASION FORCE KONDO
Wake
MARIANAS ISLANDS
SECOND FLEET KONDO
Guam
Johnston Is
Eniwetok
MARSHALL IS.
CAROLINE IS.
Wotje


Eve
Alice
Bob

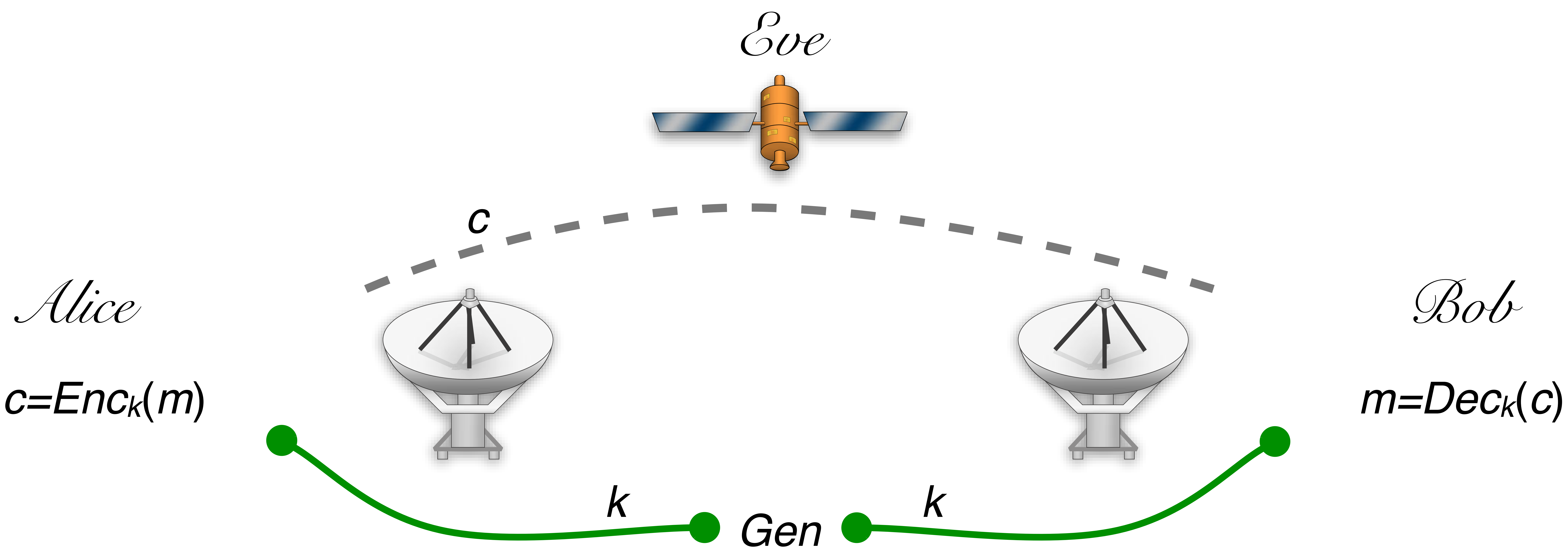


Eve
Alice
Bob
k
Gen
k

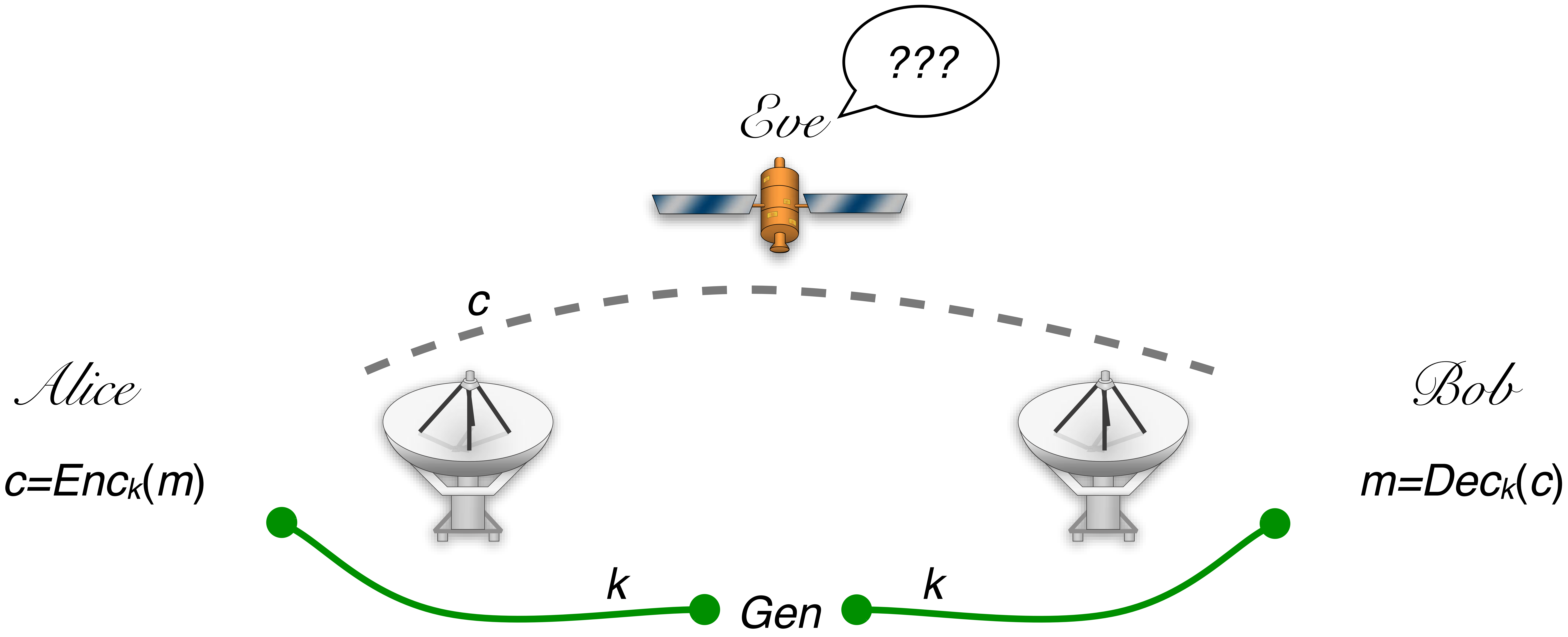

Eve
Alice
Bob
c=Enc_k(m)
k
Gen
k



Eve
c
Alice
Bob
c=Enc_k(m)
k
Gen
k



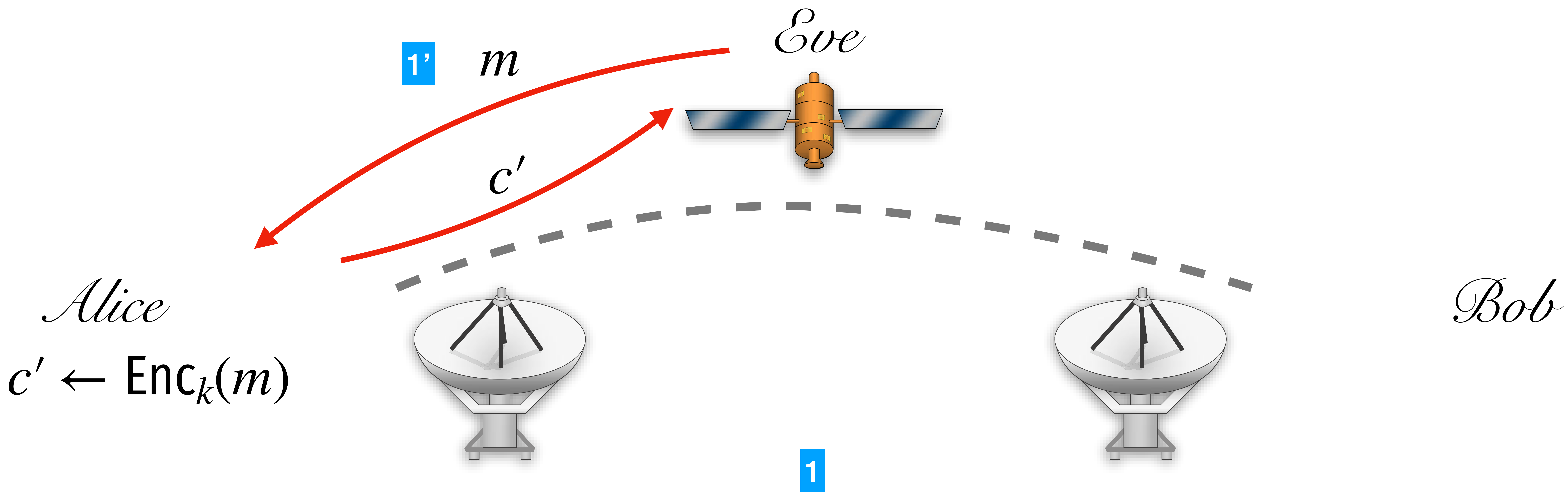
Eve
c
Alice
Bob
c=Enck(m)
m=Deck(c)
k
k
Gen



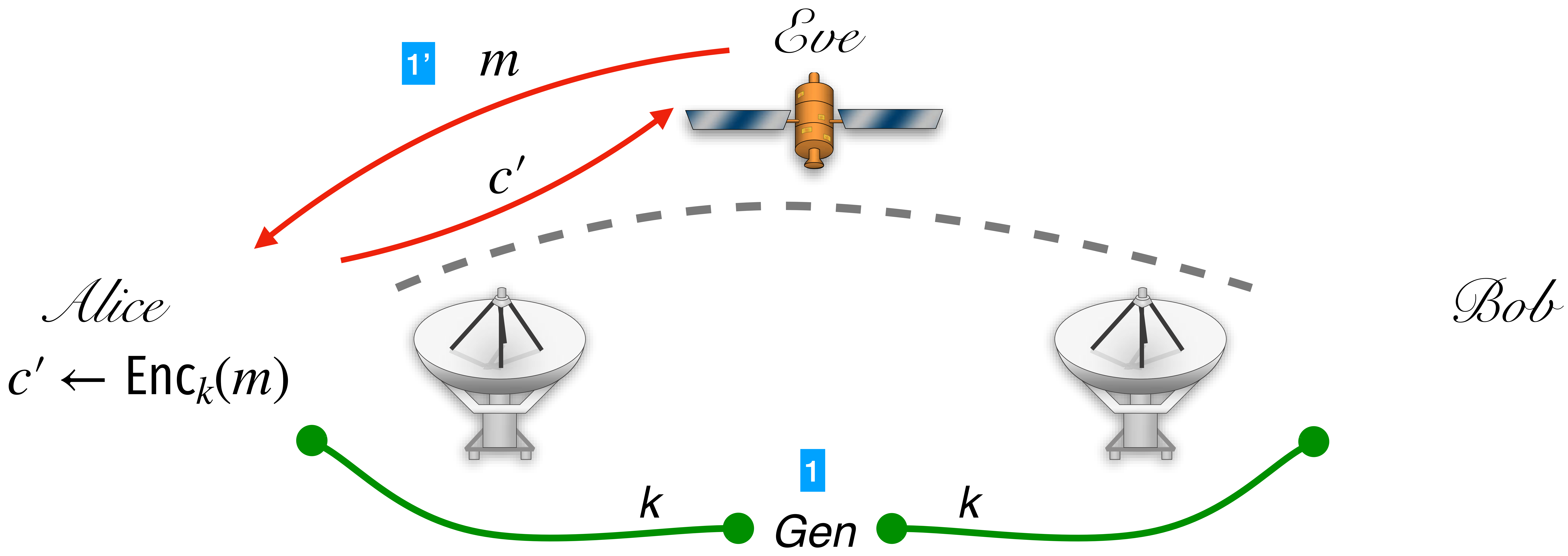
???
Eve
c
Alice
Bob
c=Enc$_k$(m)
m=Dec$_k$(c)
k
Gen
k

IND-CPA attack for Symmetric Enc

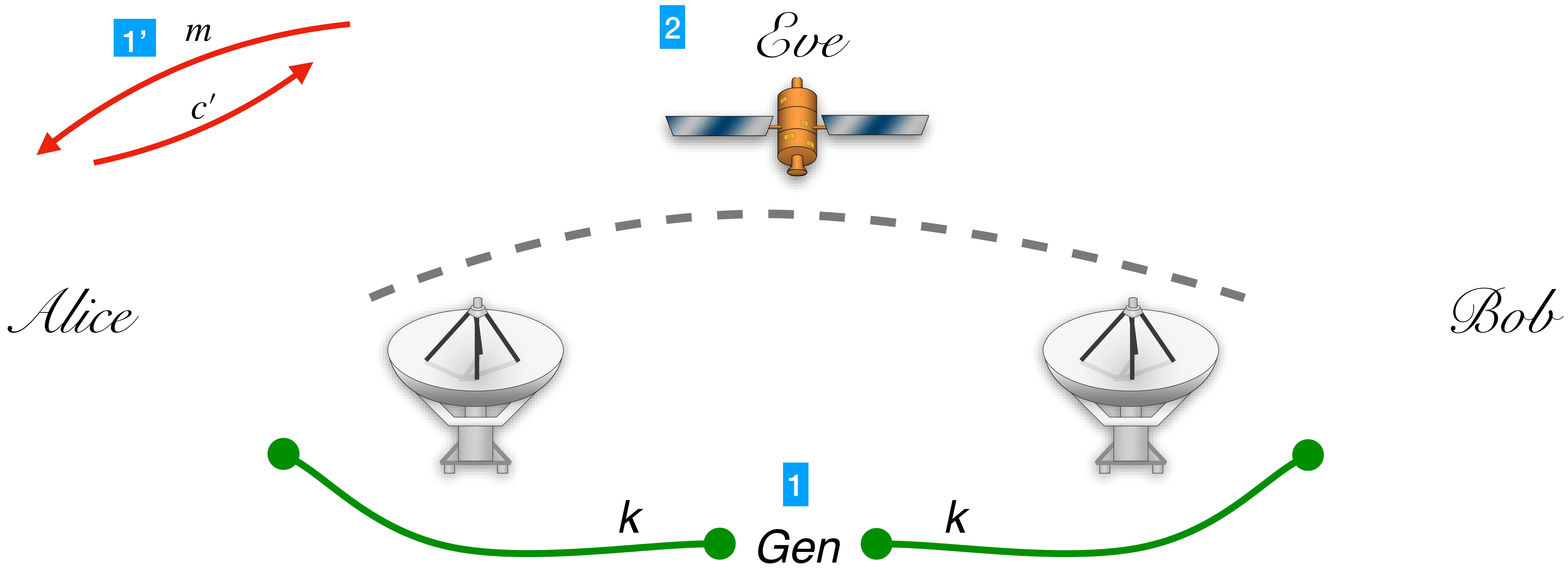
IND-CPA attack for Symmetric Enc



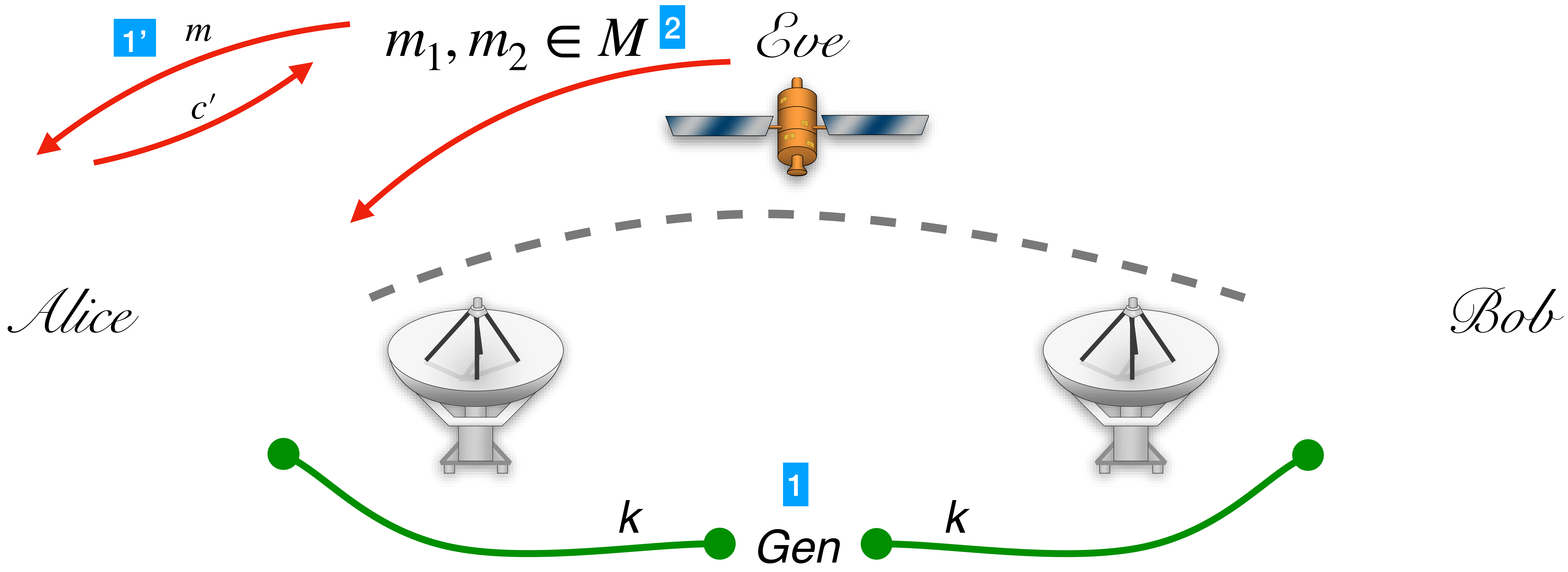
IND-CPA attack for Symmetric Enc



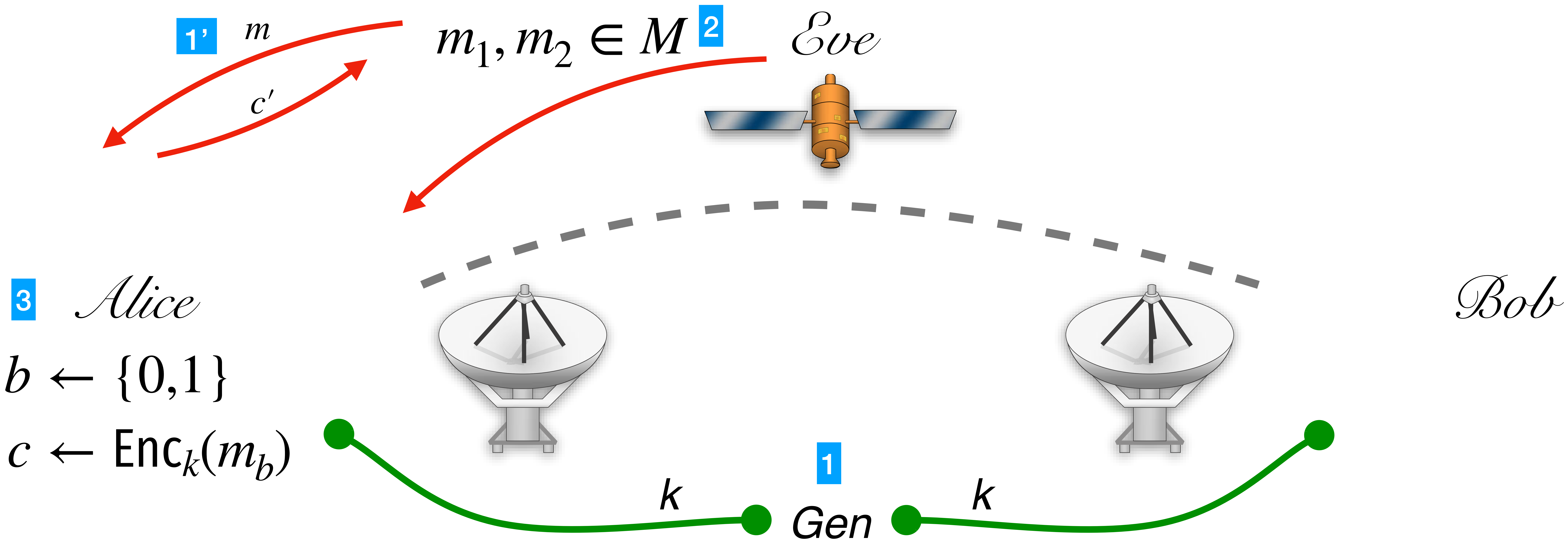
IND-CPA attack for Symmetric Enc



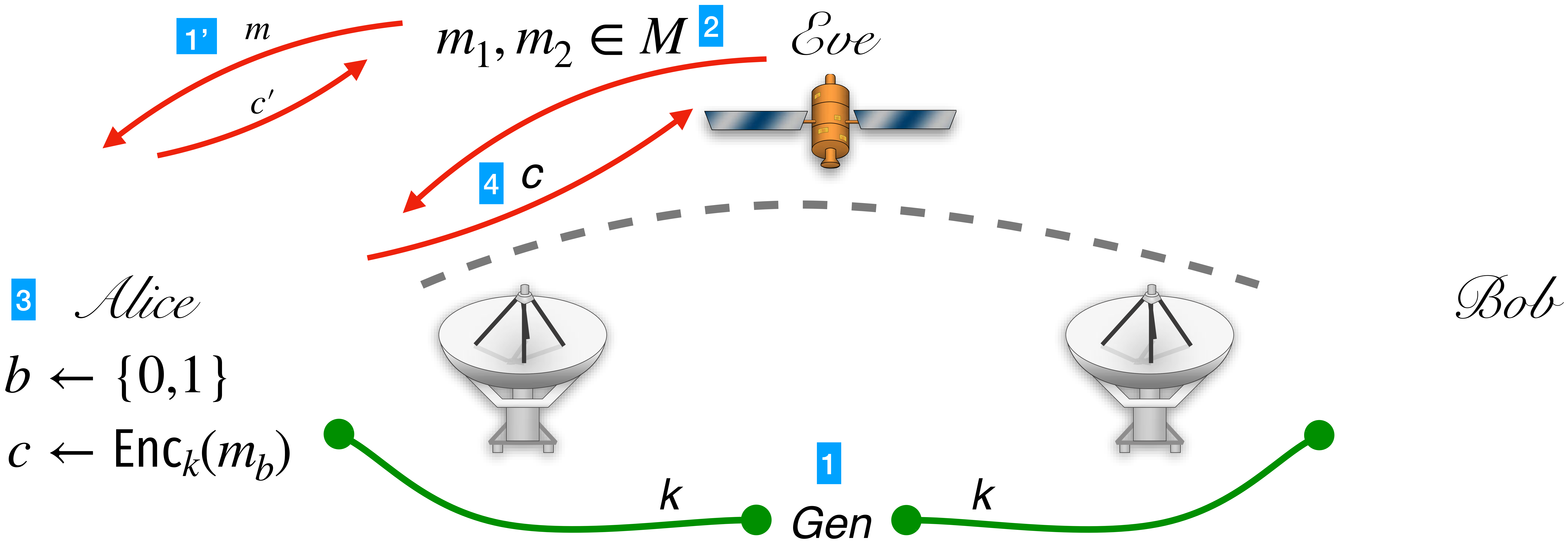
IND-CPA attack for Symmetric Enc



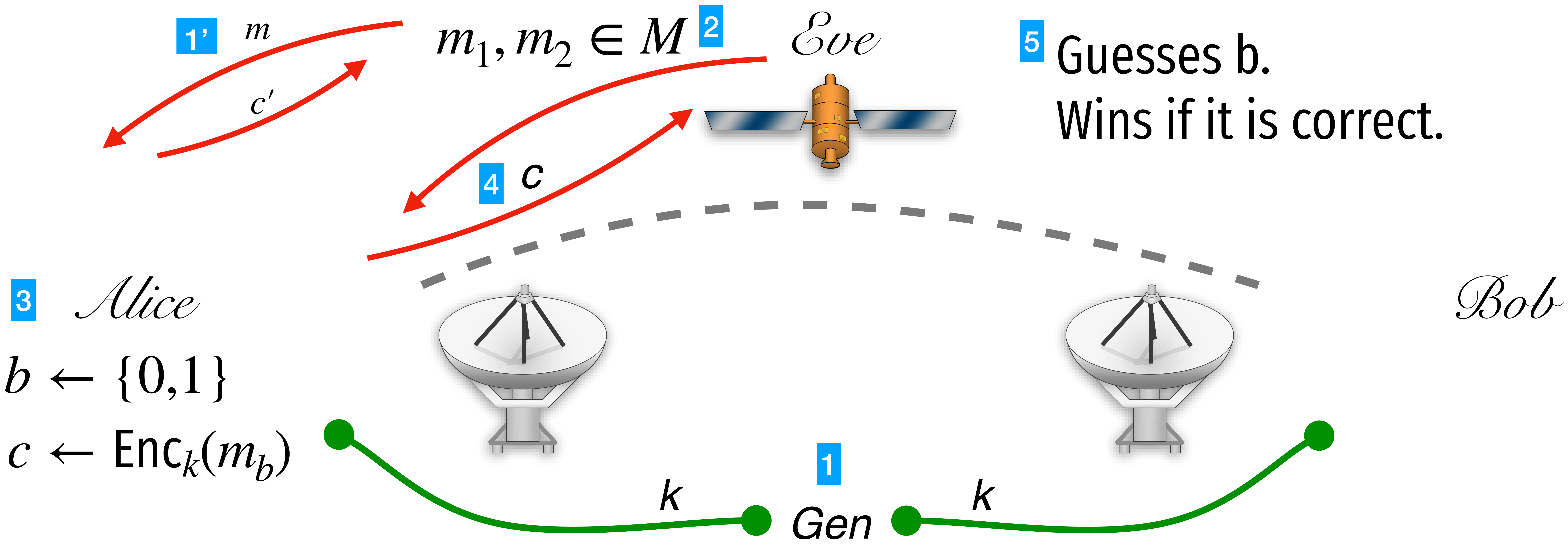
IND-CPA attack for Symmetric Enc



IND-CPA attack for Symmetric Enc



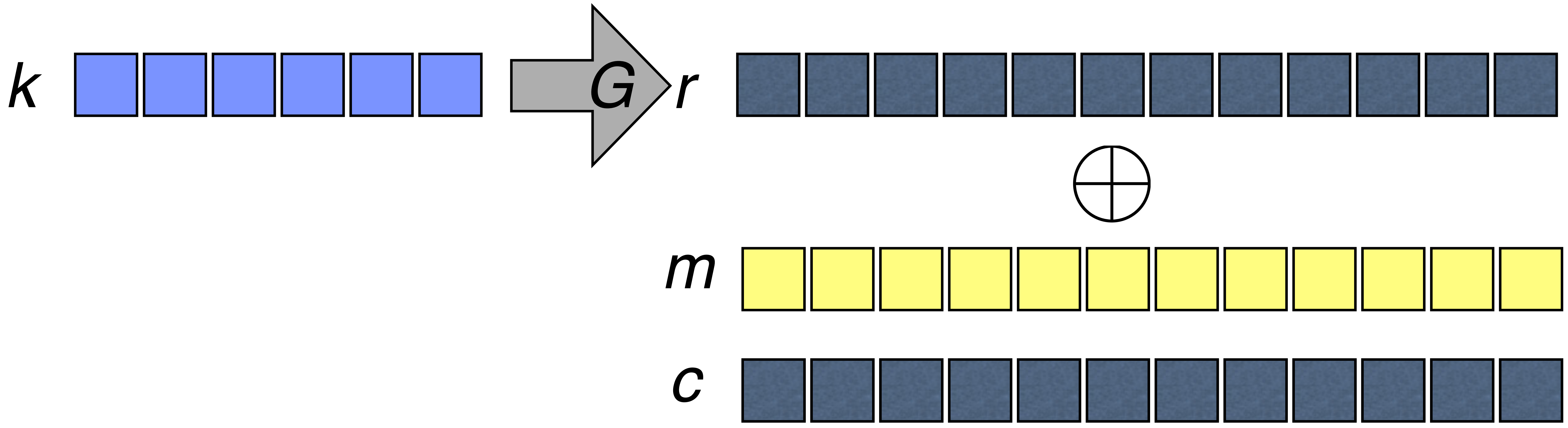
IND-CPA attack for Symmetric Enc



Our construction can satisfy this notion if both Alice and Bob maintain a counter of how much random tape they have used.

Enc$_k$(m) $\quad r \leftarrow G(k) \quad |r| = n$ (encryption)

Dec$_k$(c) $\quad$ output $\quad m \oplus r$ (decryption)



Theorem: If One-way functions exist, Then IND-CPA secure symmetric encryption exists.

Goal: Symmetric encryption with a "short" key that works for **1** arbitrarily long message

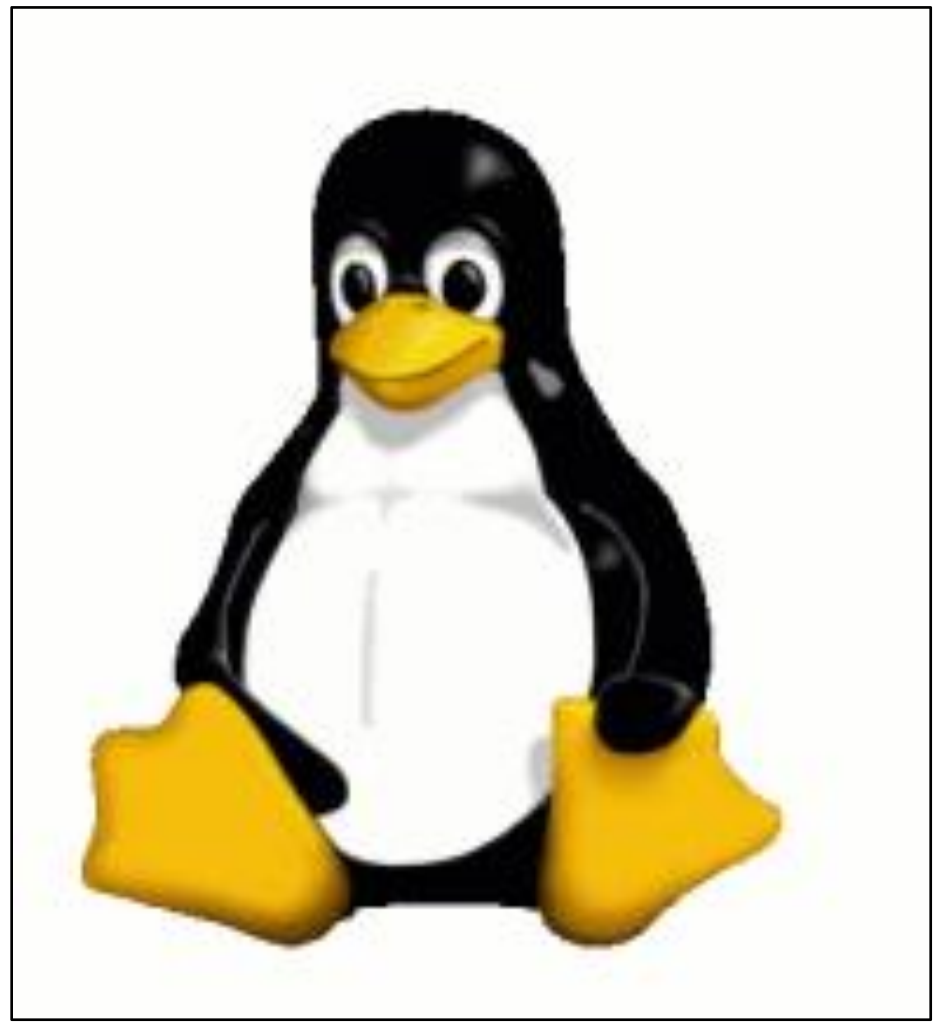
What about many messages?

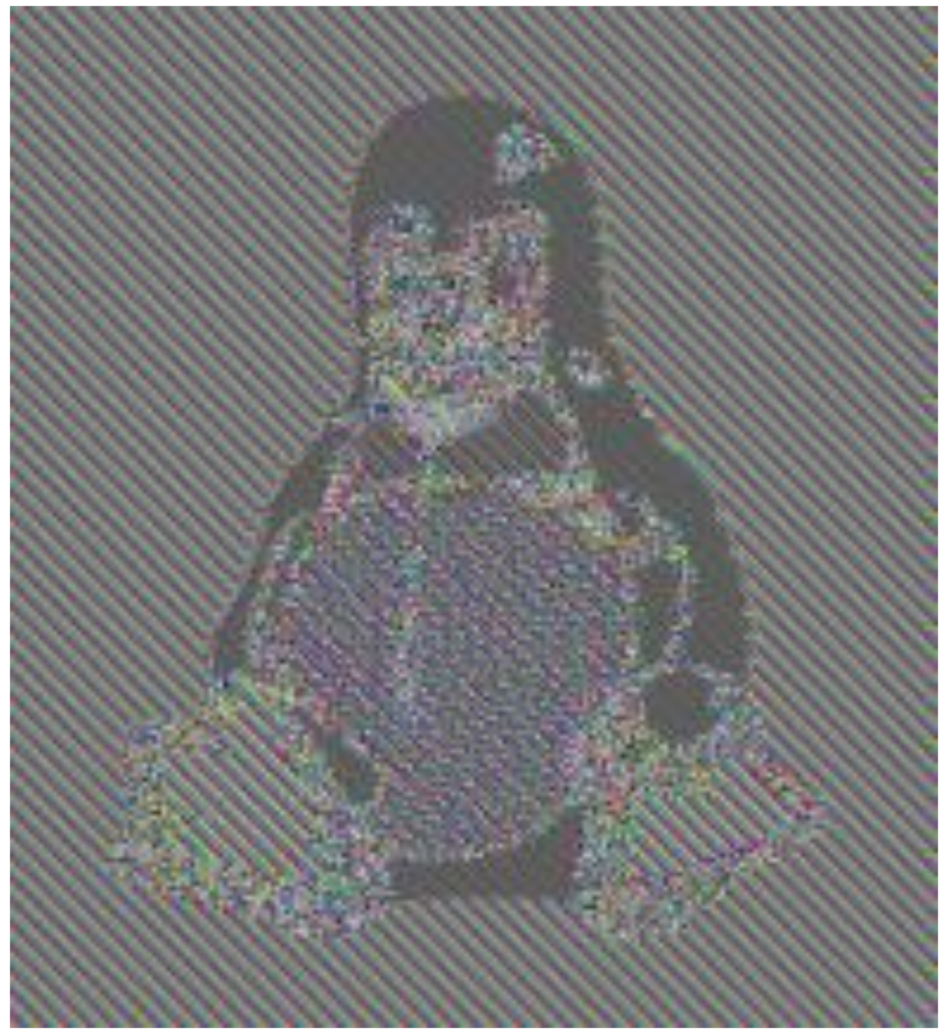
Handling many messages the wrong way

Electronic CodeBook (ECB) mode:

$$\mathrm{Enc}_k(m_1 \cdots m_\ell) = \Big(F_k(m_1), F_k(m_2), \ldots, F_k(m_\ell) \Big)$$

Original image





ECB mode encryption

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

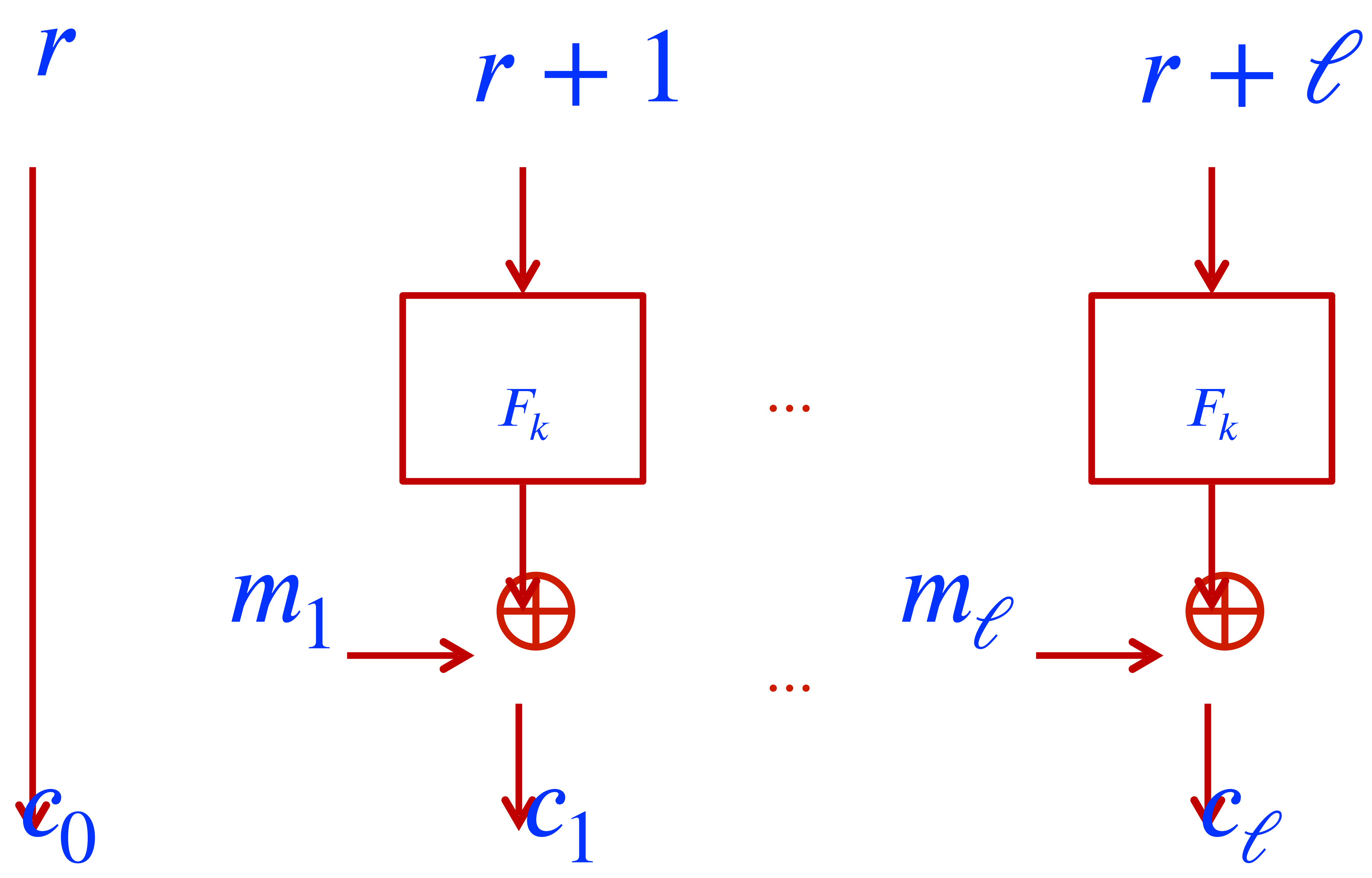
Modes of Operation: AES-CTR

Modes of Operation: AES-CTR

$$\mathrm{Enc}_k\left(m_1 \cdots m_\ell; r\right)$$
$$= \left(r, F_k(r+1) \oplus m_1, F_k(r+2) \oplus m_2, \ldots, F_k\left(r+\ell\right) \oplus m_\ell\right)$$

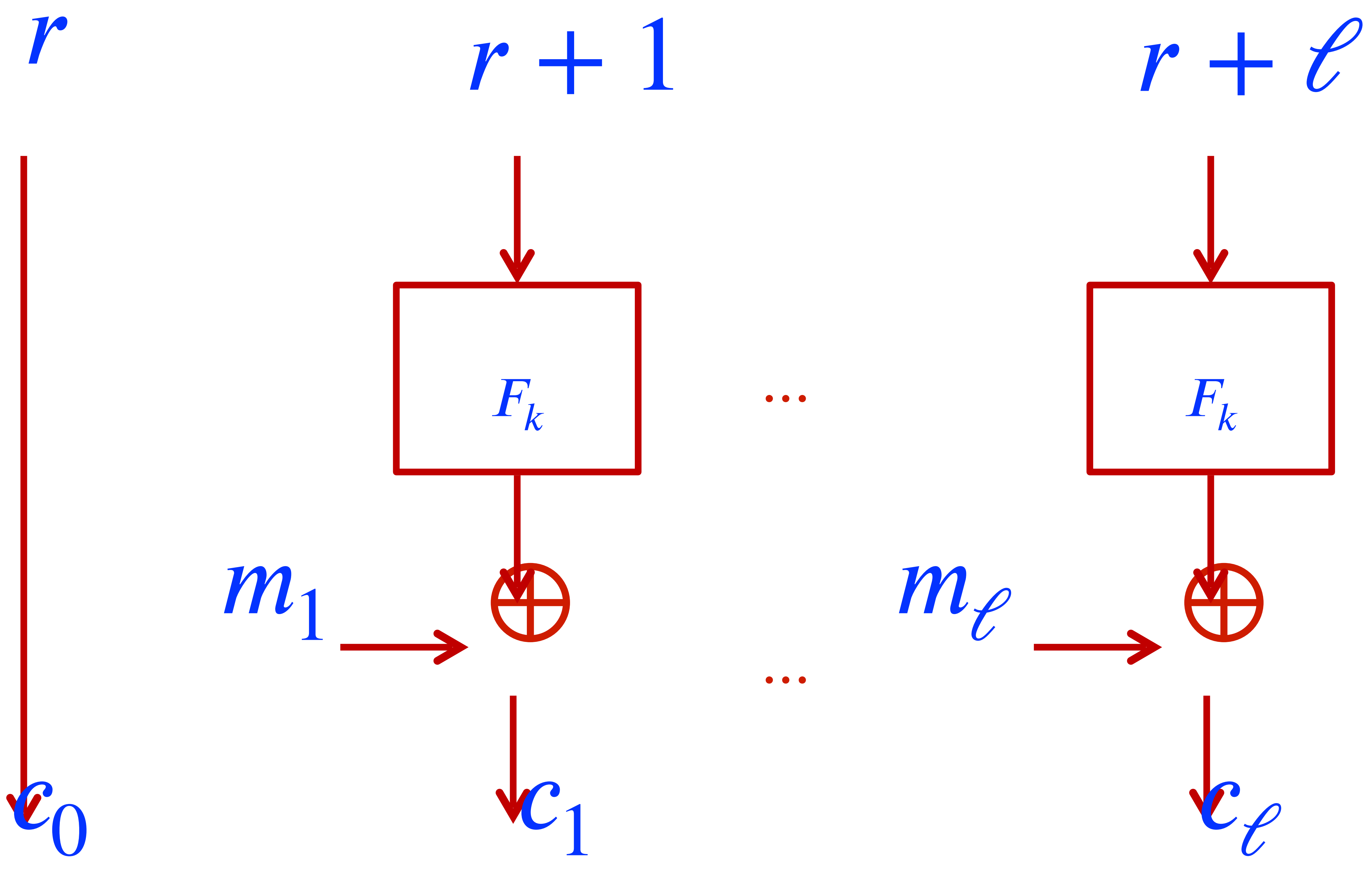
Modes of Operation: AES-CTR

$$\mathrm{Enc}_k\left(m_1 \cdots m_\ell; r\right)$$
$$= \left(r, F_k(r+1) \oplus m_1, F_k(r+2) \oplus m_2, \ldots, F_k\left(r+\ell\right) \oplus m_\ell\right)$$



Modes of Operation: AES-CTR

$$\mathrm{Enc}_k(m_1 \cdots m_\ell; r)$$
$$= \left(r, F_k(r+1) \oplus m_1, F_k(r+2) \oplus m_2, \ldots, F_k(r+\ell) \oplus m_\ell\right)$$



Ciphertext expansion is just one block

AES-CTR is also IND-CPA secure when nonce r is chosen uniquely for each encryption.