Modern Ciphers

CMSC 23200/33250, Autumn 2018, Lecture 3

David Cash

University of Chicago

Plan for today

1. Very brief recap
2. OTP and issues with OTP
3. Shortening key-length: OTP with a stream cipher
4. Block ciphers

Historical Cipher: Substitution Cipher

Encrypt(K,m): Parse key K as a permutation π on {A,… Z}. Apply π to each character of m.

P: **ATTACKATDAWN**

K: π →

C: **ZKKZAMZKYZGT**

x	π(x)
A	Z
B	U
C	A
D	Y
E	R
F	E
G	X
H	B
I	D
J	C
K	M
L	Q
M	H
N	T
O	I
P	S
Q	V
R	N
S	P
T	K
U	O
V	F
W	G
X	W
Y	L
Z	J

How many keys?

$26! \approx 2^{88}$

9 million years to try all keys at rate of 1 trillion/sec

Cipher Example: One-Time Pad

Key K: Bitstring of length L

Plaintext M: Bitstring of length L

Encrypt(K,M): Output $K \oplus M$

Decrypt(K,C): Output $K \oplus C$

Example:

$$\begin{array}{r} 0101 \\ \oplus\ 1100 \\ \hline 1001 \end{array}$$

Correctly decrypts because

$$K \oplus C = K \oplus (K \oplus m) = (K \oplus K) \oplus m = m$$

Q: Is the one-time pad secure?

Bigger Q: What does "secure" even mean?

Evaluating Security of Crypto

<u>Kerckhoff's Principle</u>: Assume adversary knows your algorithms and implementation. The only thing it doesn't know is the key.

1. Quantify adversary goals

   Learn something about plaintext? Spoof a message?

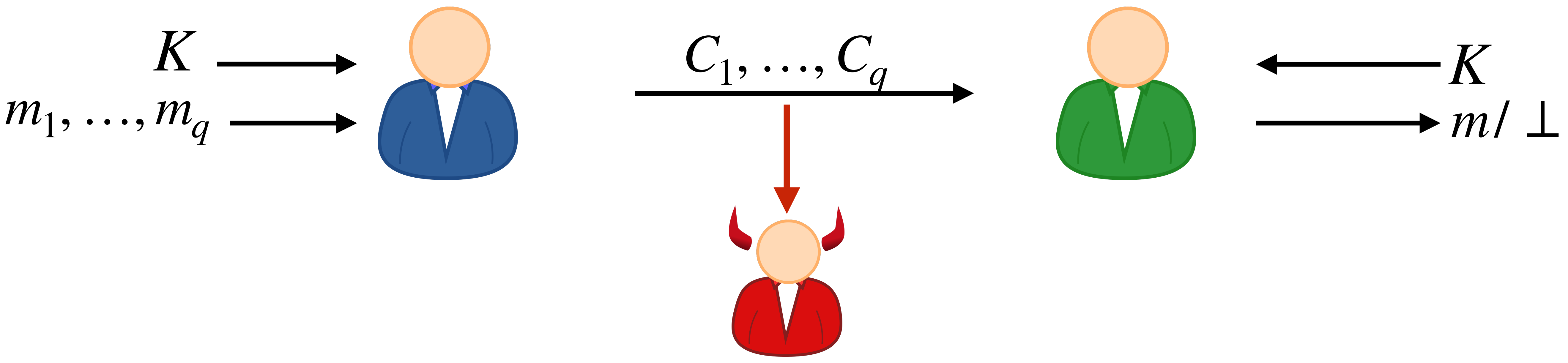
2. Quantify adversary capabilities

   View ciphertexts? Probe system with chosen inputs?

3. Quantify computational resources available to adversary

   Compute cycles? Memory?

Breaking Encryption - A Basic Game

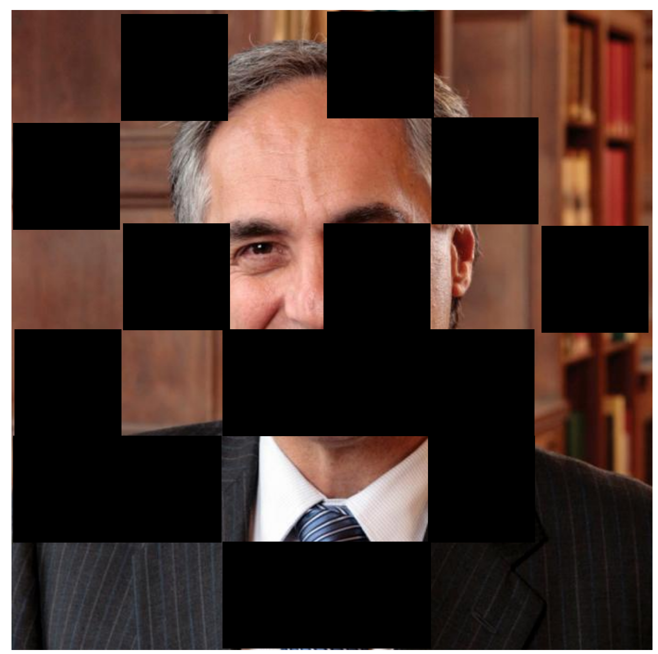


Ciphertext-only attack: The adversary sees ciphertexts and attempts to recover some useful information about plaintexts.

More attack settings in next lecture.

What is useful information?

- Recovering entire messages is useful
- But recovering **partial information** is also be useful



A lot of information is missing here.

But can we say who this is?

- Attacker may know large parts of plaintext already (e.g. formatting strings or application content). The attacker tries to obtain something it doesn't already know.

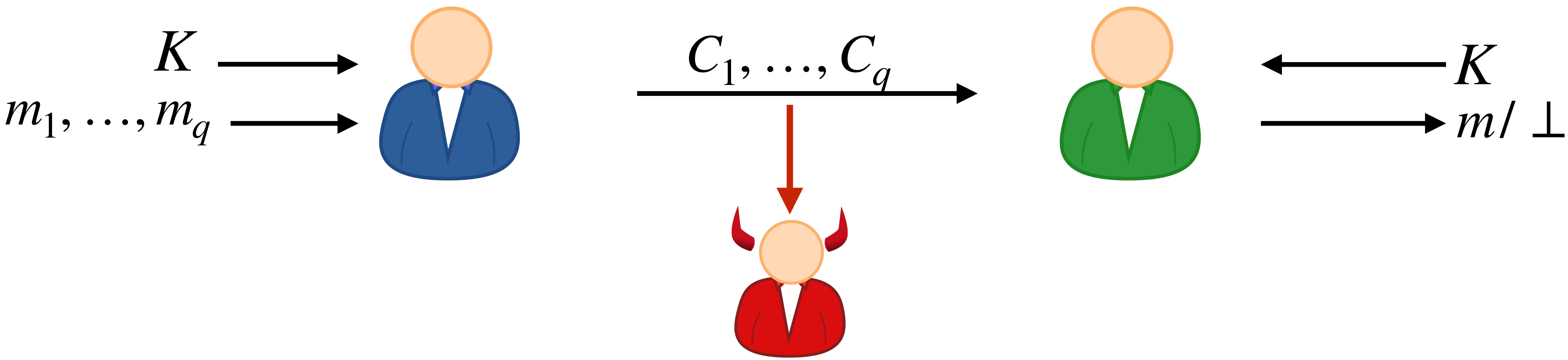
`M = http://site.com?password=`[redacted]

"Attacks" versus "Security"

An **attack** is successful as long as it recovers some useful information about plaintext.

Encryption should hide all possible partial information about plaintexts, since what is useful is situation-dependent.

Does an attack need to recover the key?



Full break: Adversary recovers K, decrypts all ciphertexts.

However: Clever attacker may compromise encryption without recovering the key.

Security of One-Time Pad

Claim: If adversary sees **only one** ciphertext under a random key, then any plaintext is equally likely, so it cannot recover any partial information besides plaintext length.

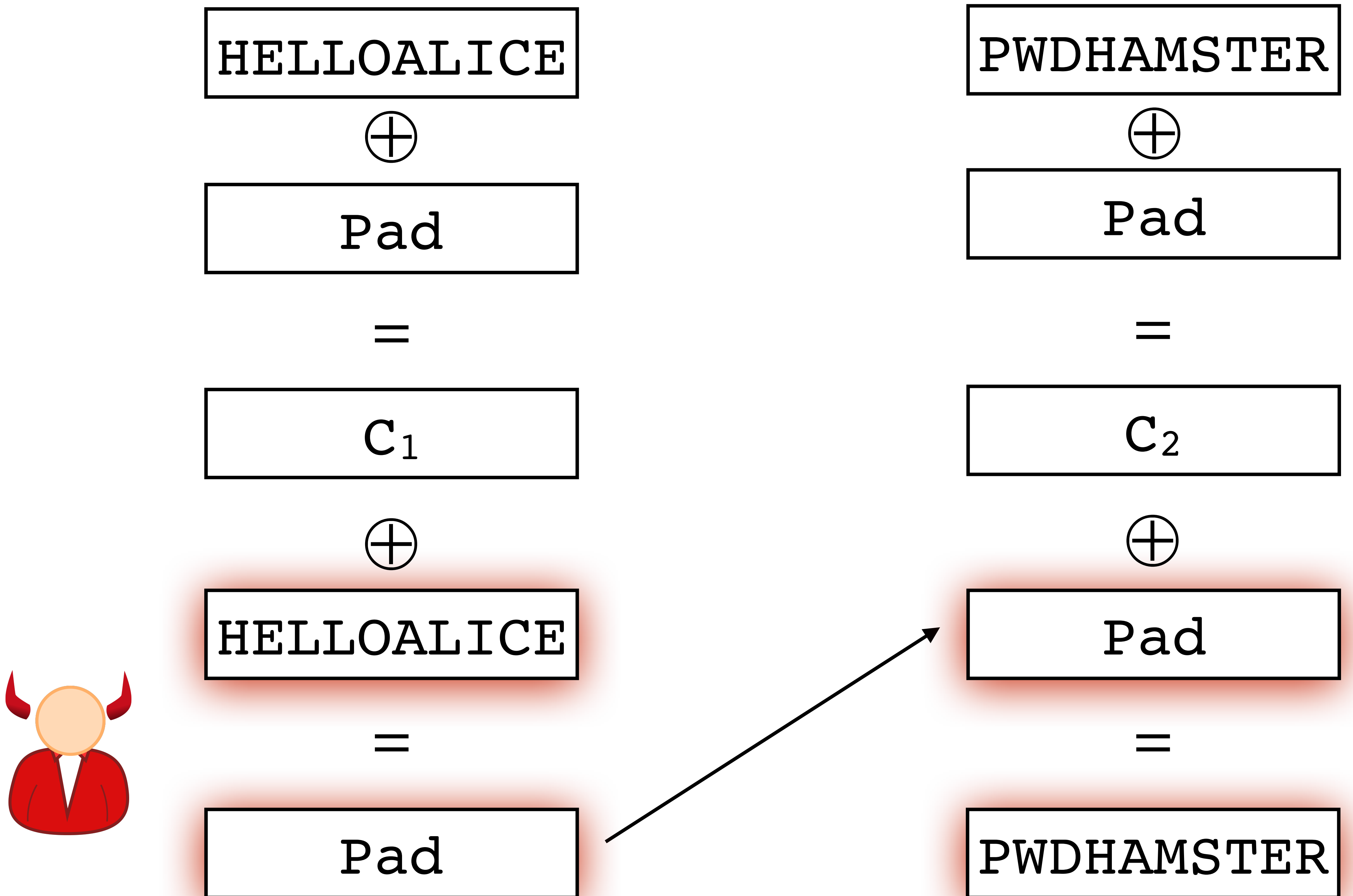
Ciphertext observed: `10111`
Possible plaintext: `00101`
⇒ Possible key: `10010`

1. Adversary goal: Learn partial information from plaintext
2. Adversary capability: Observe a single ciphertext
3. Adversary compute resources: Unlimited time/memory (!)

Issues with One-Time Pad

1. Reusing a pad is insecure
2. One-Time Pad is *malleable*
3. One-Time Pad has a long key

Issue #1: Reusing a One-Time Pad is Insecure

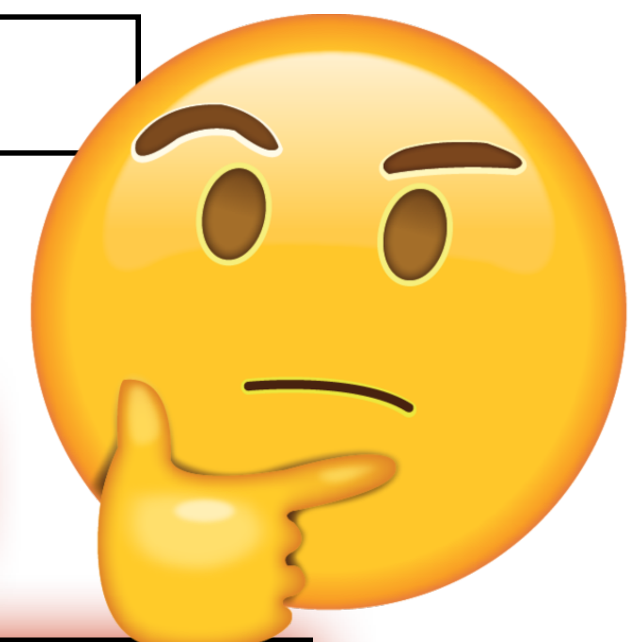


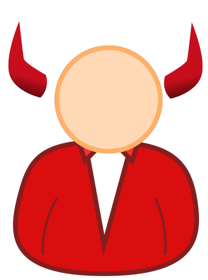
Issue #1: Reusing a One-Time Pad is Insecure

Has led to real attacks:

- Project Venona (1940s) attack by US on Soviet encryption
- MS Windows NT protocol PPTP
- WEP (old WiFi encryption protocol)
- Frequency table of x⊕y for English

= c_1 = c_2

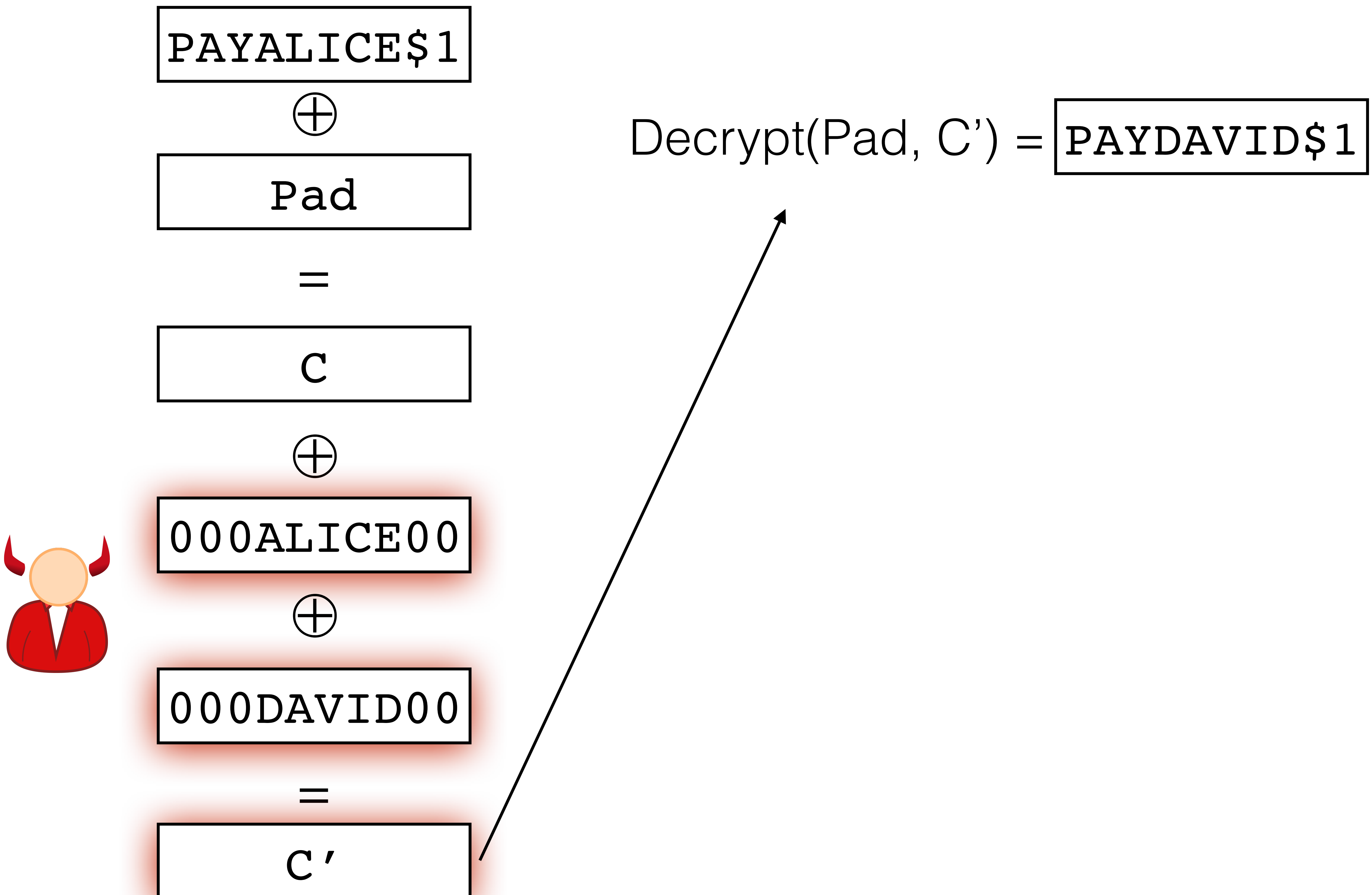




$c_1 \oplus c_2$

= S3CR3T1234 ⊕ 3L33THXRRR

Issue #2: One-Time Pad is *Malleable*



Issue #3: One-Time Pad Needs a Long Key

Can prove: Any cipher as secure as the OTP must have:

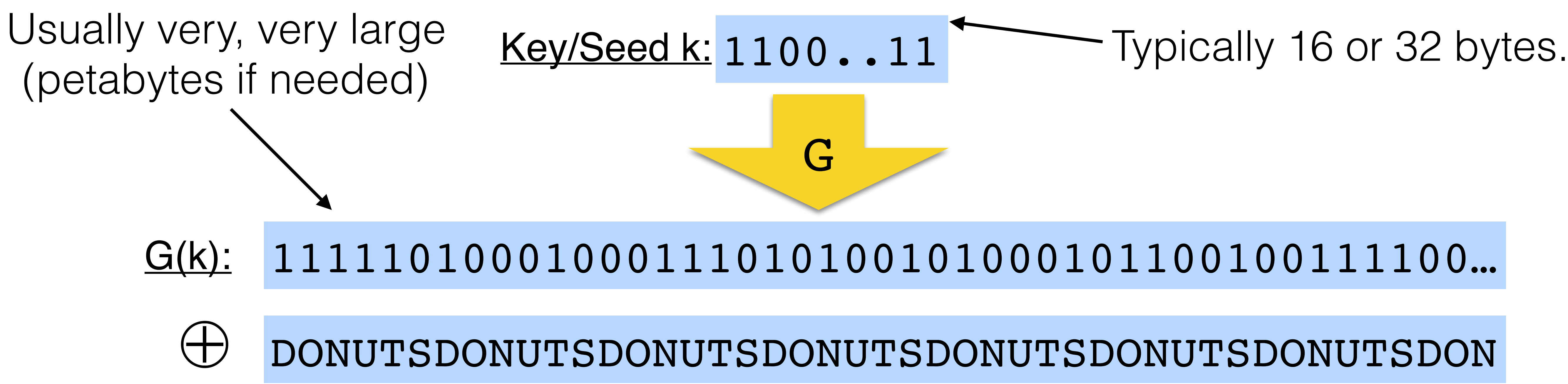
Key-length ≥ Plaintext-length

In practice: (covered in next few lectures):

- Use stream cipher: Encrypt(K,m) = G(K)⊕m
- Add authentication tag
- Use nonces to encrypt multiple messages

Tool to address key-length of OTP: Stream Ciphers

<u>Stream cipher syntax:</u> Algorithm G that takes one input and produces an very long bit-string as output.



Use G(seed) in place of pad.
Still malleable and still one-time, but key is shorter.

Stream Cipher Security Goal (Sketch)

<u>Security goal:</u> When **`k`** is random and unknown, **`G(k)`** should "look" random.

… even to an adversary spending a lot of computation.

Much stronger requirement that "passes statistical tests".

<u>Brute force attack:</u> Given **`y=G(k)`**, try all possible **`k`** and see if you get the string **`y`**.

<u>Clarified goal:</u> When **`k`** is random and unknown, **`G(k)`** should "look" random to anyone with less computational power needed for a brute force attack.

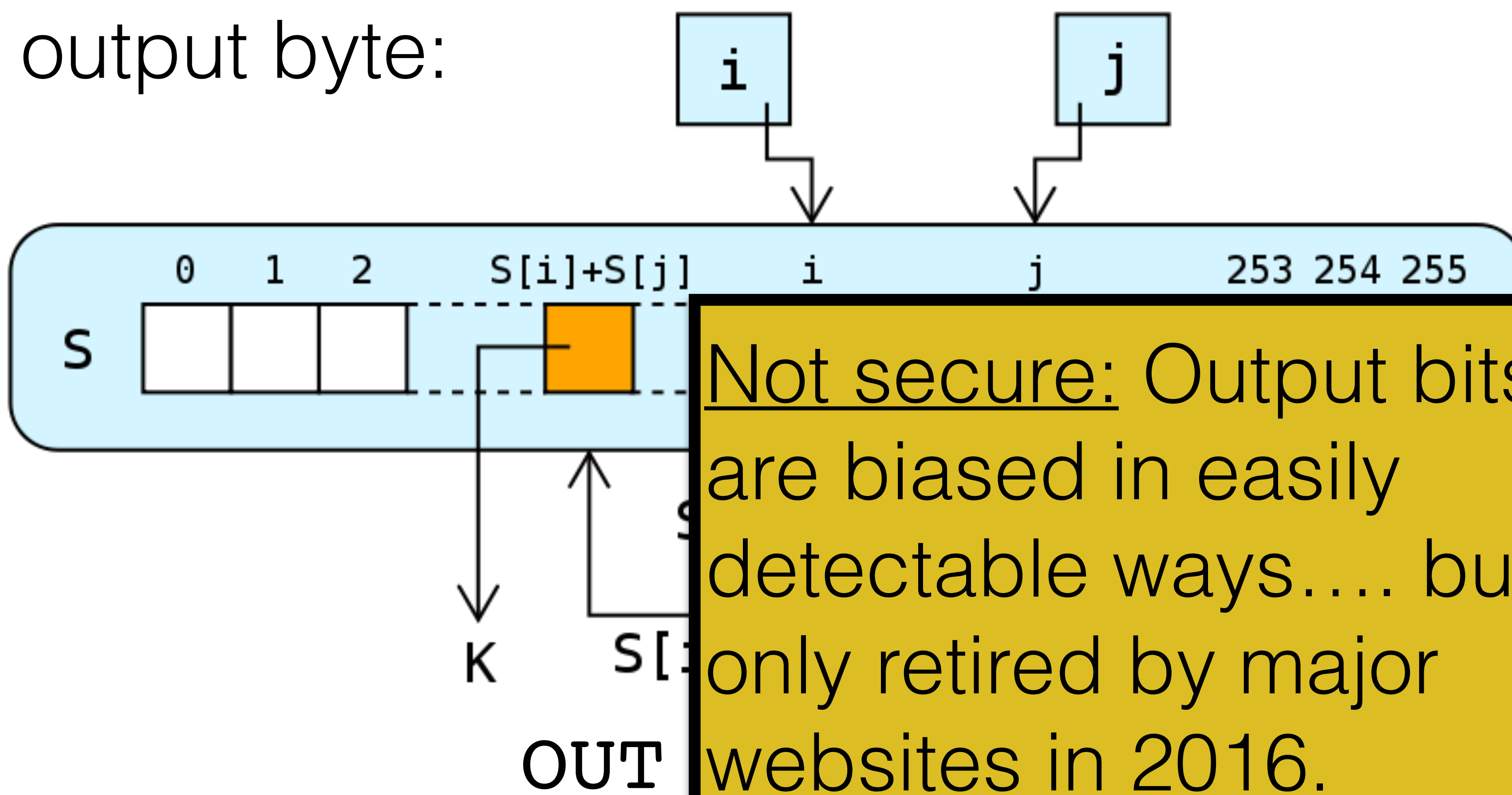
(keylength = 256 is considered strong now)

Example Stream Cipher: RC4

Warning: Broken

<u>Internal state:</u> Array `S` of 256 bytes and ptrs `i, j`

To compute next output byte:



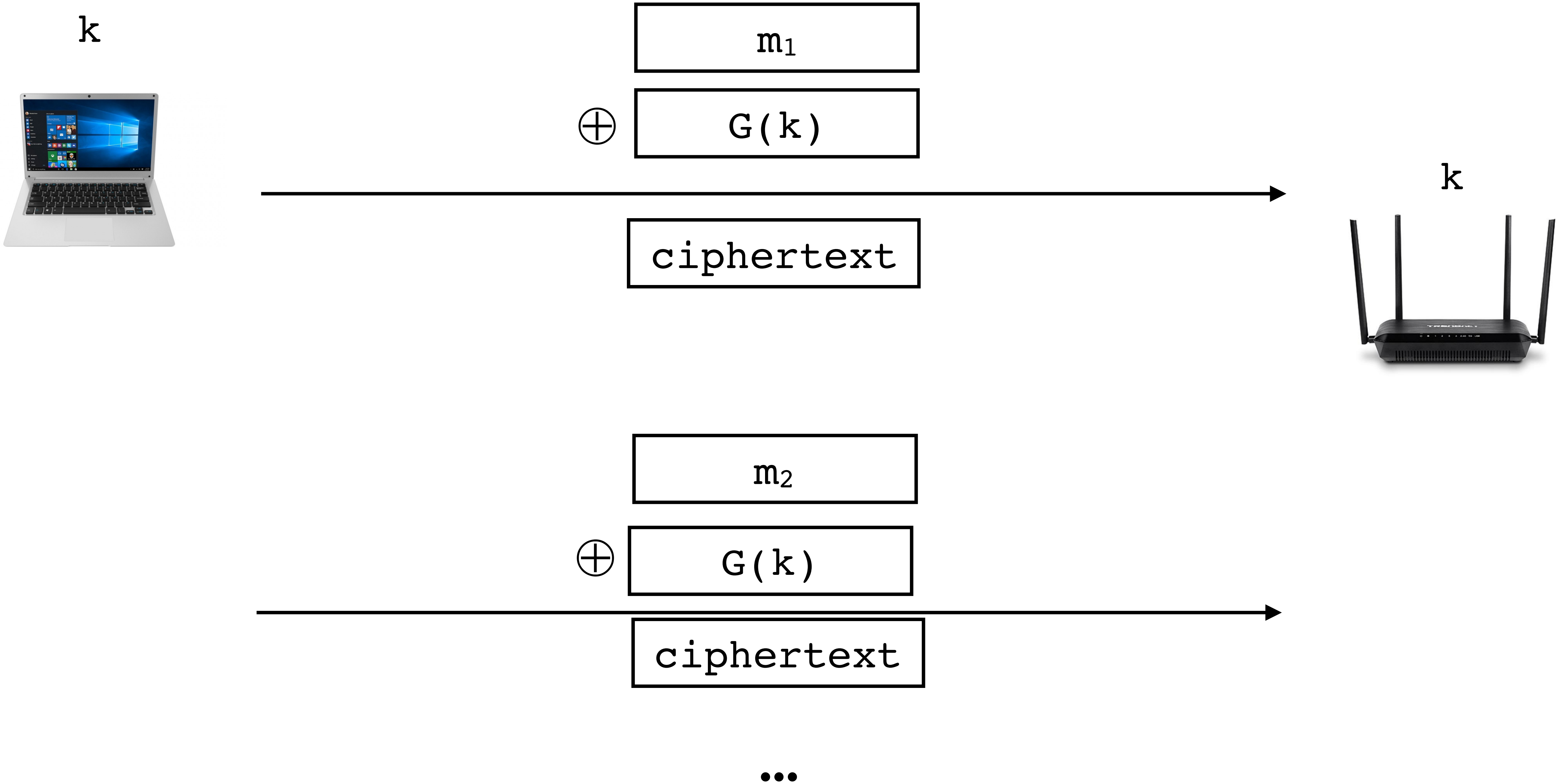
<u>Not secure:</u> Output bits are biased in easily detectable ways…. but only retired by major websites in 2016.

<u>Replacement:</u> Salsa20/ ChaCha, or AES-based methods to be discussed

<u>Then:</u>

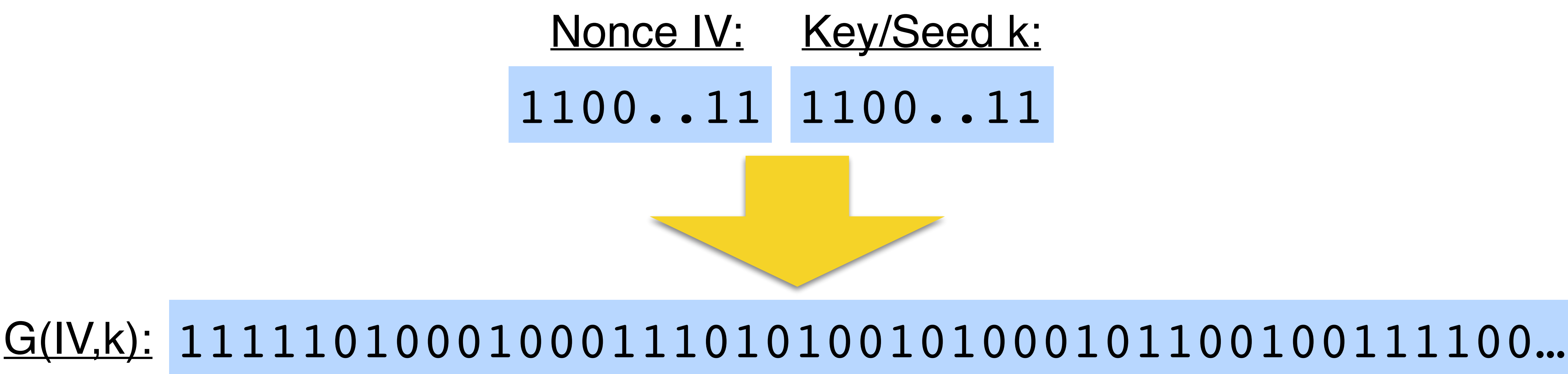
- `i=i+1 mod256`
- `j=j+S[i] mod256`
- `swap S[i] and S[j]`

Pad reuse can still happen with stream ciphers



Addressing pad reuse: Stream cipher with a nonce

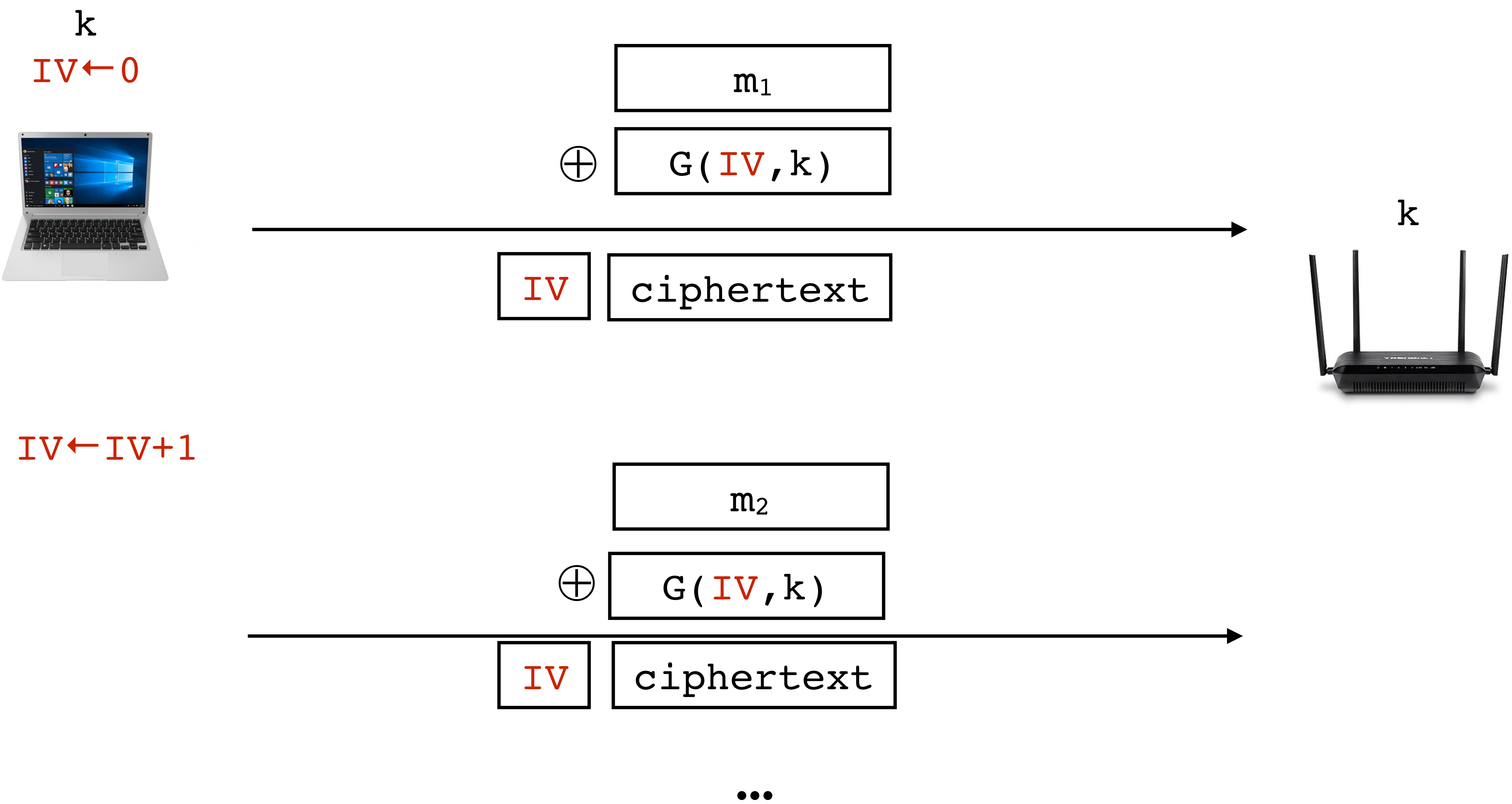
<u>Stream cipher with a nonce:</u> Algorithm G that takes **two inputs** and produces an very long bit-string as output.



- “nonce” = “number once”.
- Usually denoted IV = “initialization vector”

<u>Security goal:</u> When `k` is random and unknown, `G(IV,k)` should “look” random and independent for each value of `IV`.

Solution 1: Stream cipher with a nonce

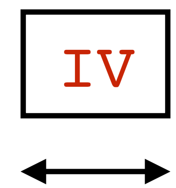


- If nonce repeats, then pad repeats

Example of Pad Re-use: WEP

Warning: Broken

<u>IEEE 802.11b WEP:</u> WiFi security standard ’97-‘03



IV is 24-bit wide counter

- Repeats after 2^{24} frames (≈16 million)
- IV is often set to zero on power cycle

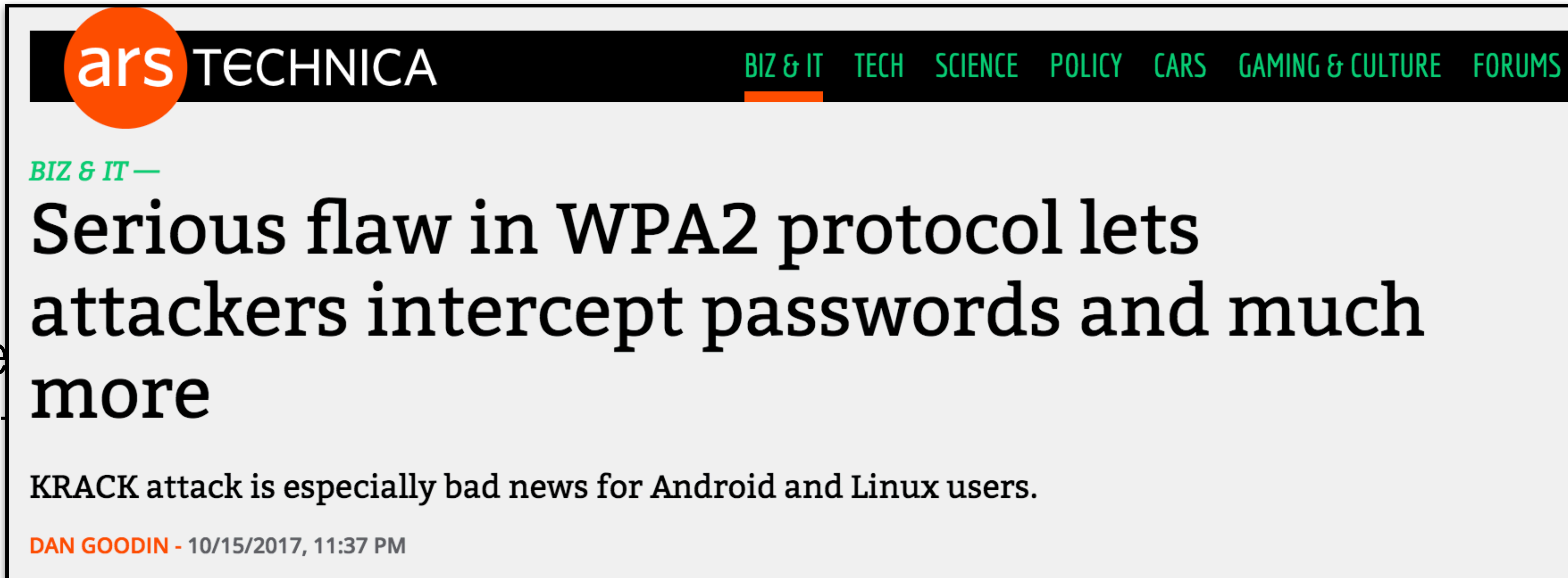
<u>Solutions: (WPA2 replacement)</u>

- Larger IV space, or force rekeying more often
- Set IV to combination of packet number, address, etc

Example of Pad Re-use: WEP

Warning: Broken

IEEE 802.11b WEP: WiFi security standard ’97-‘03



ars TECHNICA

BIZ & IT | TECH | SCIENCE | POLICY | CARS | GAMING & CULTURE | FORUMS

BIZ & IT —

Serious flaw in WPA2 protocol lets attackers intercept passwords and much more

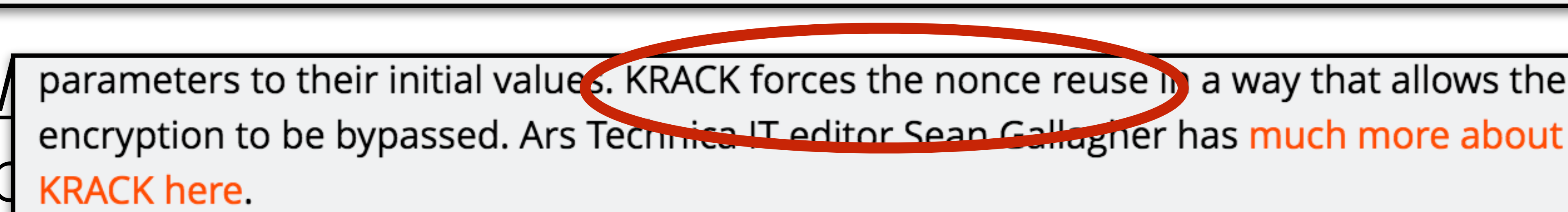
KRACK attack is especially bad news for Android and Linux users.

DAN GOODIN - 10/15/2017, 11:37 PM

- Re
- Of

Solutions: (W

- Larger IV sp



parameters to their initial values. KRACK forces the nonce reuse in a way that allows the encryption to be bypassed. Ars Technica IT editor Sean Gallagher has much more about KRACK here.

- Set IV to combination of packet number, address, etc

Issues with One-Time Pad

1. Reusing a pad is insecure ✓ *Use unique nonces*
2. One-Time Pad is *malleable*
3. One-Time Pad has a long key ✓ *Use stream cipher with sort key*

More difficult to address; We will return to this later.

Next Up: Blockciphers

Blockciphers are a ubiquitous crypto tool applied to many different problems.

Informal definition: A blockcipher is essentially a substitution cipher with a very large alphabet and a very compact key. Require that efficient algorithms for forward and backward directions.

Typical parameters:
Alphabet = $\{0,1\}^{128}$
Key length = 16 bytes.

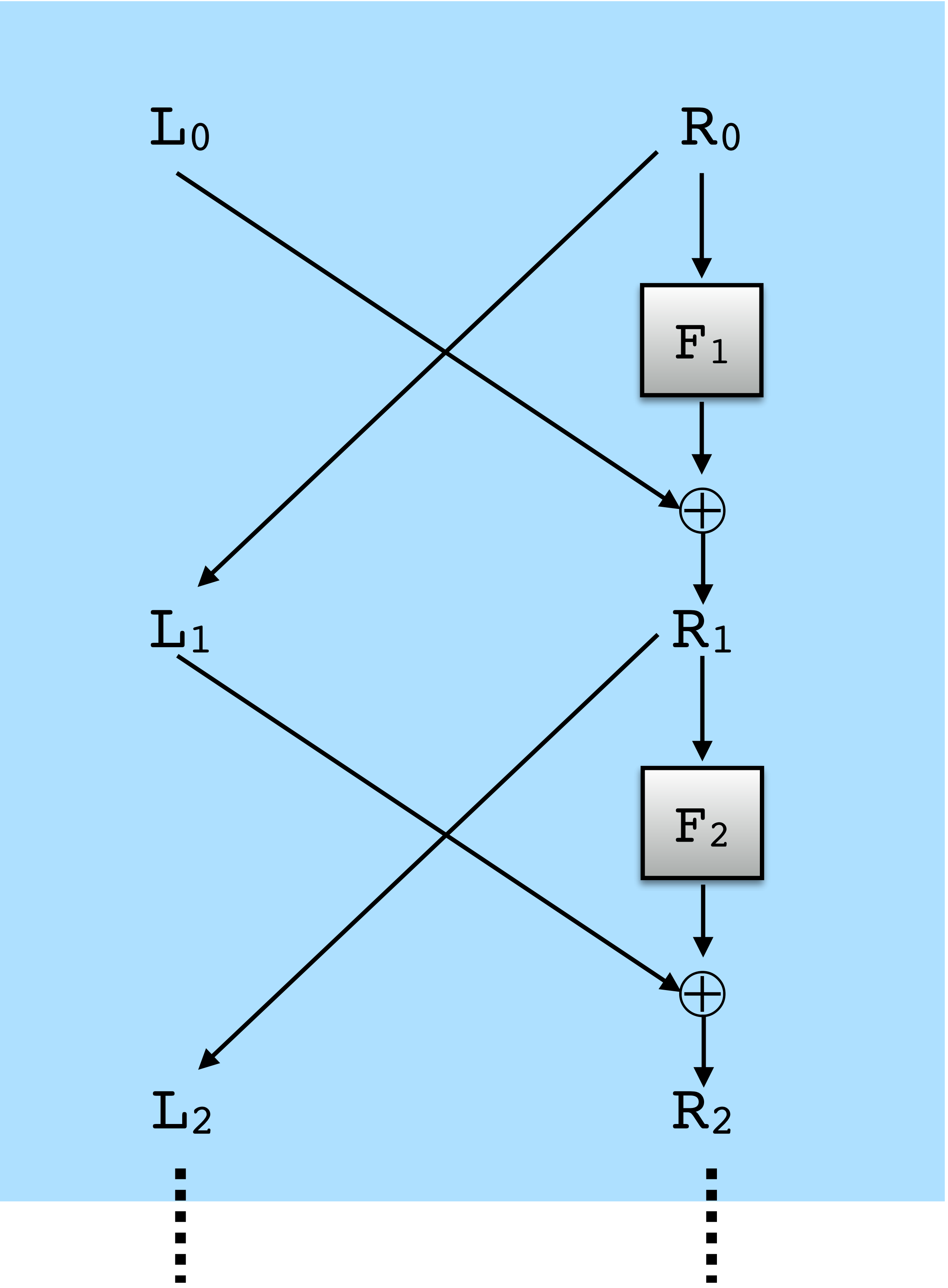
Plan: Build many higher-level protocols from a good blockchiper.

Now: Two example blockciphers, DES and AES.

Data Encryption Standard (DES)

- Originally a designed by IBM
- Parameters adjusted by NSA
- NIST Standard in 1976
  - Block length n = 64
  - Key length k = 56

Parses input block into 32-bit chunks and applies 16 rounds of a “Feistel Network”



DES is Broken

Warning: Broken

Attack	Complexity	Year
Biham&Shamir	2^{47} encrypted blocks	1992
DESCHALL	41 days	1997
EFF Deepcrack	4.5 days	1998
EFF Deepcrack	22 hours	1999

- 3DES ("Triple DES") is still used by banks
- 3DES encrypts three times (so key length is 118)
- 3DES is not known to be broken but should be avoided

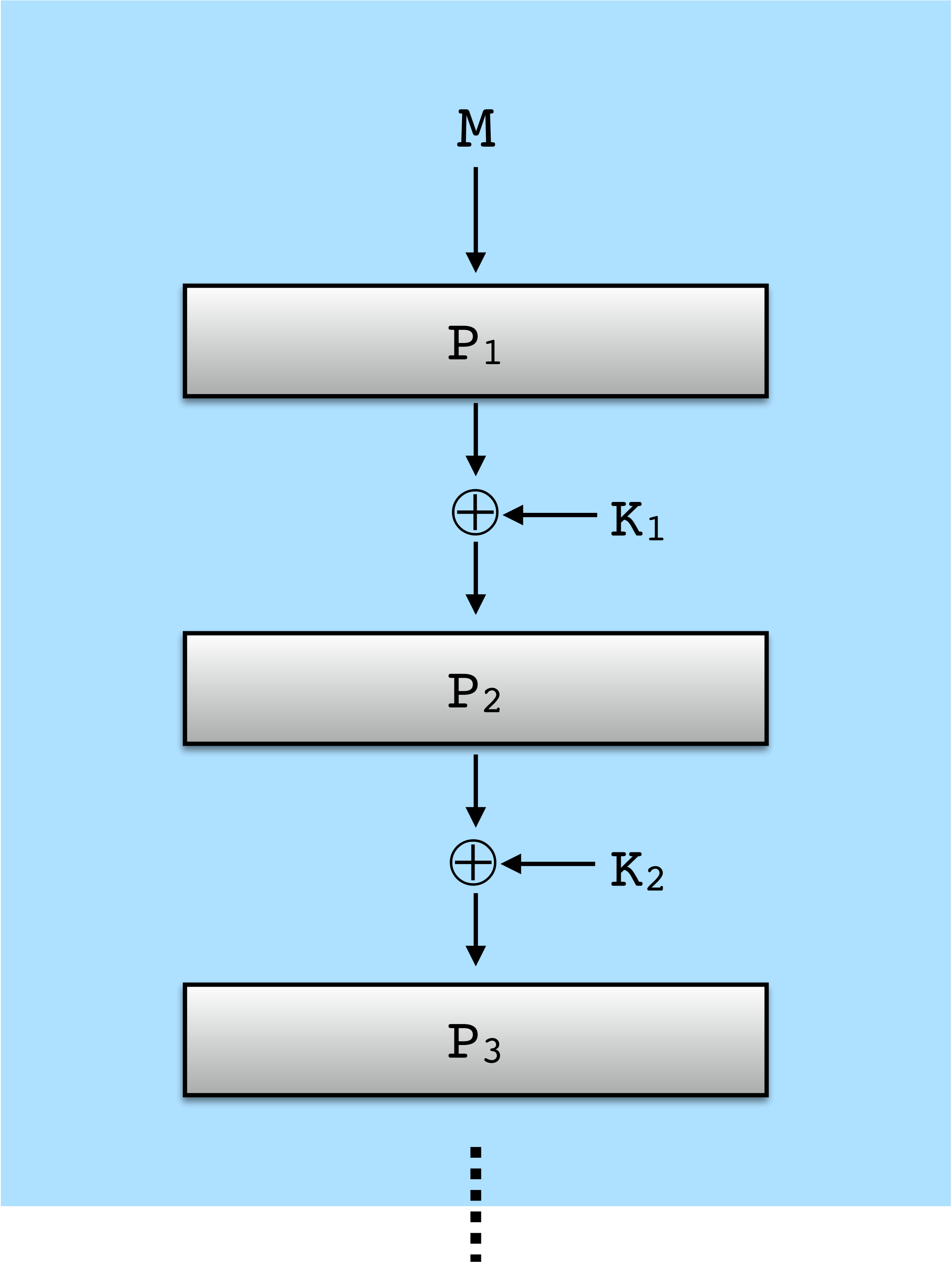
Advanced Encryption Standard (AES)

- NIST ran competition to replace DES starting in 1997
- Several submissions, *Rijndael* chosen and standardized
- AES is now the gold standard blockcipher
- Very fast; Intel chips even have AES instructions

Advanced Encryption Standard (AES)

- Due to Rijmen and Daemen
  - Block length n = 128
  - Key length k = 128,192,256

- Different structure from DES.
- 10 rounds of “substitution-permutation”



AES is not (know to be) broken

Attack	Complexity	Year
Bogdanov et al.	$\approx 2^{126.1}$	2011

- Compare to trying all keys: $2^{126.1} \approx 2^{128}/4$
- Always prefer AES for a blockcipher if setting can support it (i.e. everything except low-power hardware)

Brief Aside: Computational Strength Today

# Steps	Who can do that many?
2^{56}	Strong computer with GPUs
2^{80}	All computers on Bitcoin network in a few days
2^{128}	Very large quantum computer*
2^{192}	Nobody?
2^{256}	Nobody?

*Not directly comparable but this is an estimate of equivalent power. Quantum computers are most effective against public-key crypto, but they also speed up attacks on symmeric-key crypto. (More next week.)