Lecture 6: Hash functions + intro to encryption

Posted on piazza.com

- Lab 3: due on Monday 2/11 at 11pm (re-download if you got it before Wed night)
- Week 3 reading (no reading for the discussion section)

Random functions R: $\{0,1\}^{in} \rightarrow \{0,1\}^{out}$

Case:	in < out	in = out (permutation)	in > out	in = ∞
One *public* function	Stream cipher ←	Codebook	→ Compression function →	Hash function/ random oracle
Family of functions indexed by *private* key		↓ Block cipher →	Message auth code	↗

Random functions R: $\{0,1\}^{in} \rightarrow \{0,1\}^{out}$

Case:	in < out	in = out (permutation)	in > out	in = ∞
One *public* function	Stream cipher	Codebook	Compression function	Hash function/ random oracle
Family of functions indexed by *private* key		Block cipher	Message auth code	

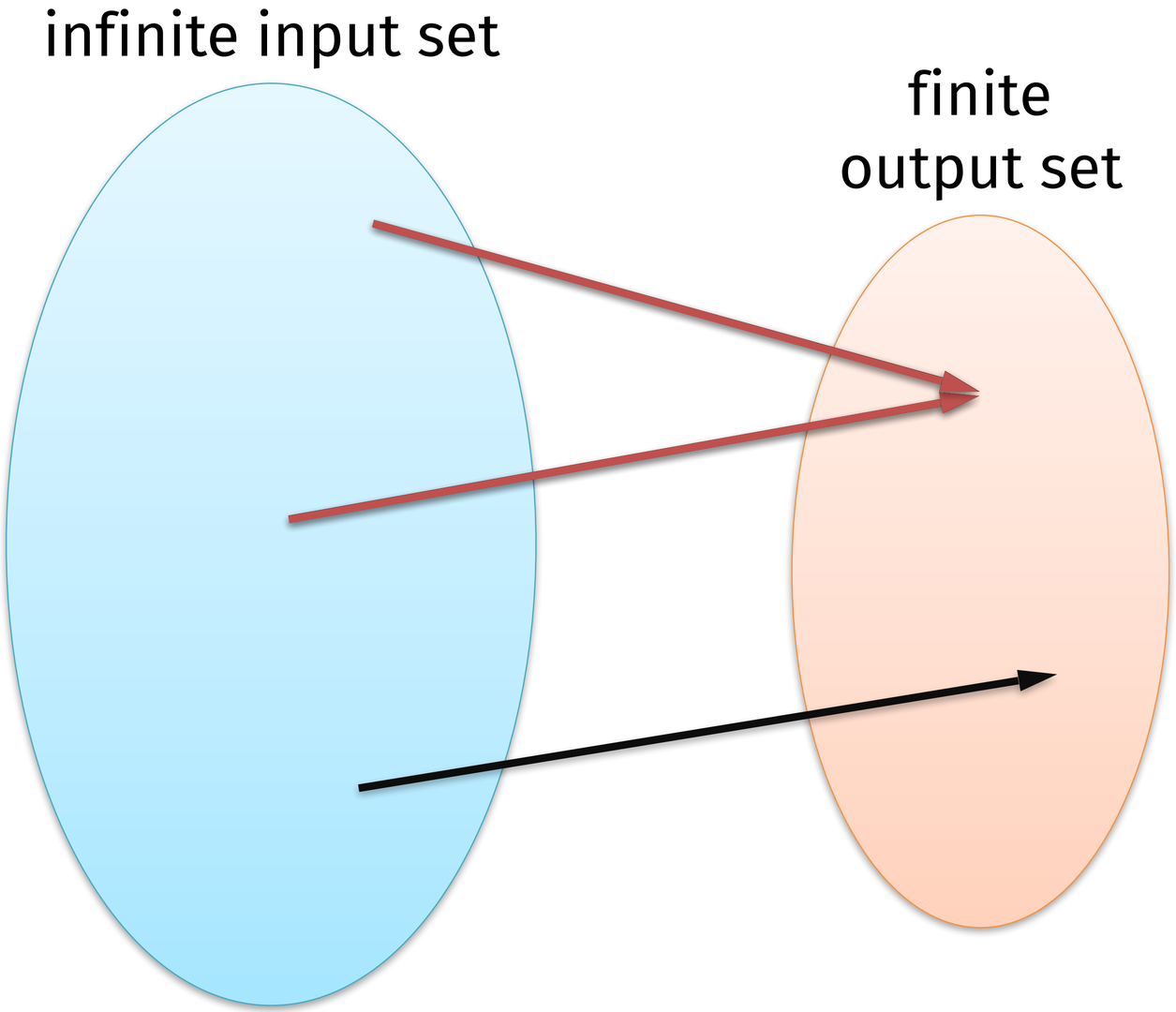
Random functions R: $\{0,1\}^{in} \rightarrow \{0,1\}^{out}$

Case:	in < out	in = out (permutation)	in > out	in = ∞
One *public* function	Stream cipher	Codebook	Compression function	Hash function/ random oracle
Family of functions indexed by *private* key		Block cipher → CMAC, OMAC	Message auth code	

Random functions R: $\{0,1\}^{in} \rightarrow \{0,1\}^{out}$

Case:	in < out	in = out (permutation)	in > out	in = ∞
One *public* function	Stream cipher	Codebook	Compression function → **Merkle-Dåmgard** → Hash function/ random oracle	
Family of functions indexed by *private* key		Block cipher →	Message auth code	**HMAC**

Hash function



- "Compresses" long messages into short digests
- Cannot invert!
- Difficult to find two messages with the same digest
- Runtime of brute force attack is given by birthday bound

$$E[\text{\# hash evals until collision}] \approx \sqrt{\frac{\pi}{2}L}$$

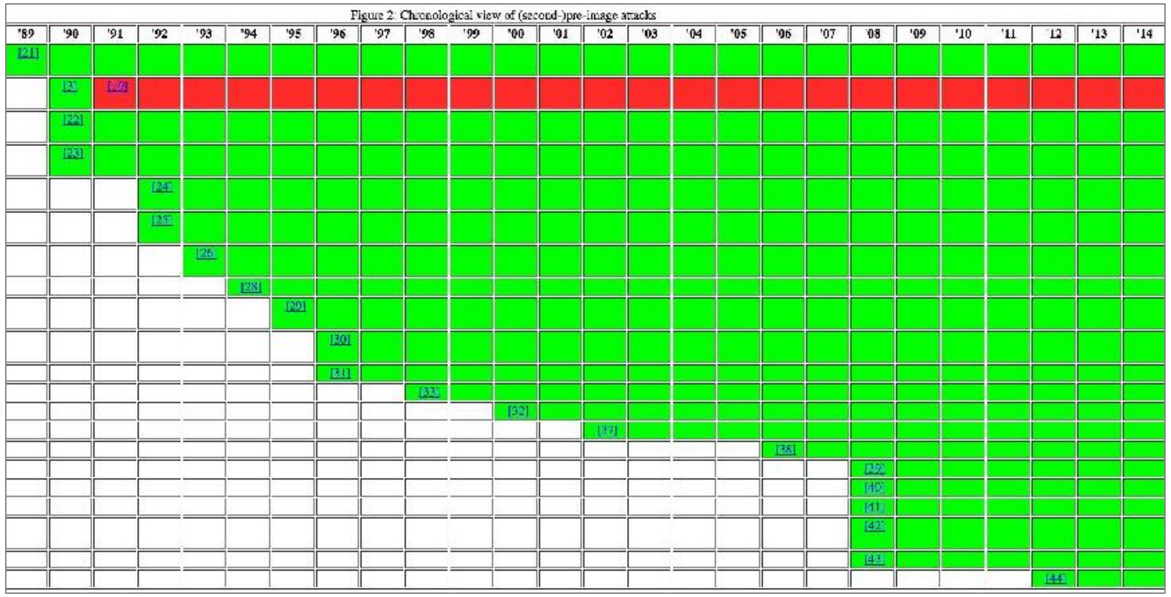
$$\approx 1.25\sqrt{L}$$

Hash function strength

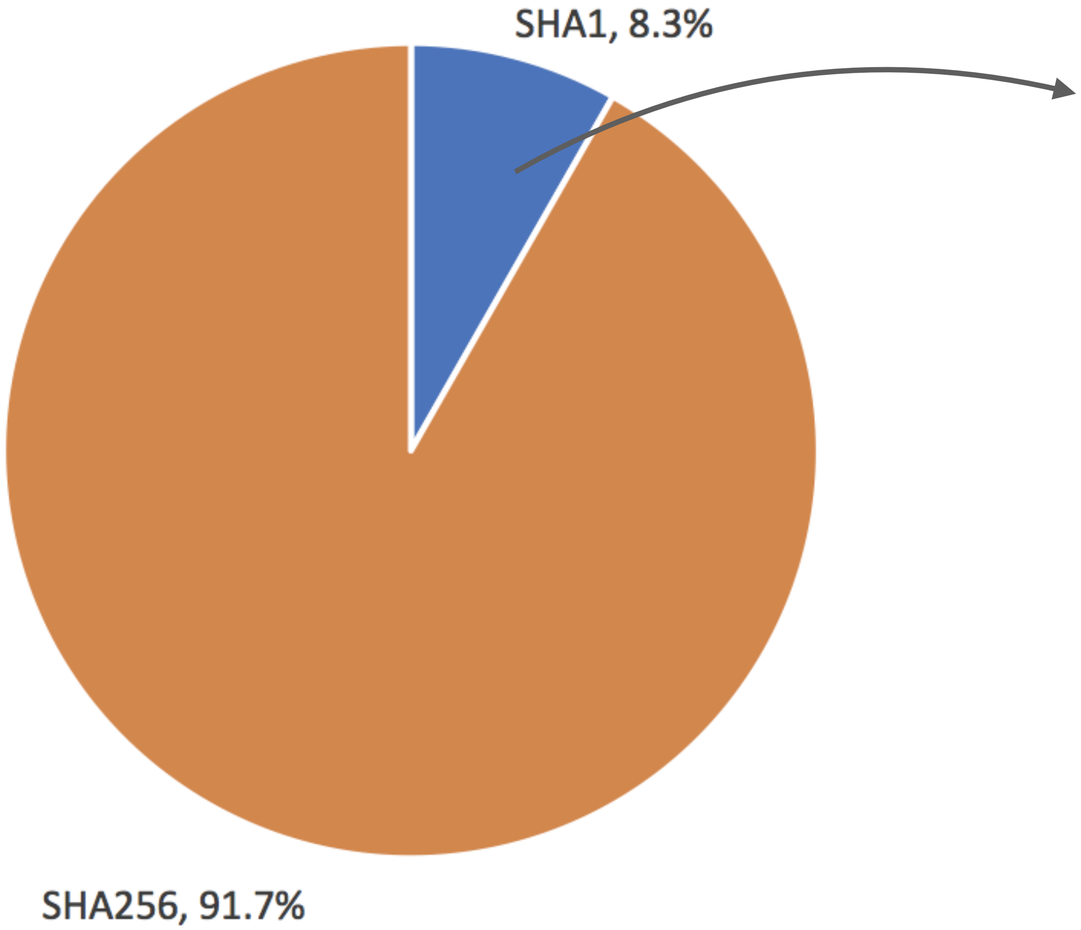
For all definitions, the adversary has the code of H: {0,1}* → {0,1}^{out}

- Preimage resistance: given **y** = **H**(**x**), tough to find any preimage **x'**
- 2nd preimage resistance: given **x**, tough to find new **x'** s.t. **H**(**x'**) = **H**(**x'**)
- Target collision resistance: same as CR, except **x** chosen before **H** known
- Collision resistance: given only **H**, difficult to find two different inputs **x** and **x'** s.t. **H**(**x'**) = **H**(**x'**) faster than a birthday bound search

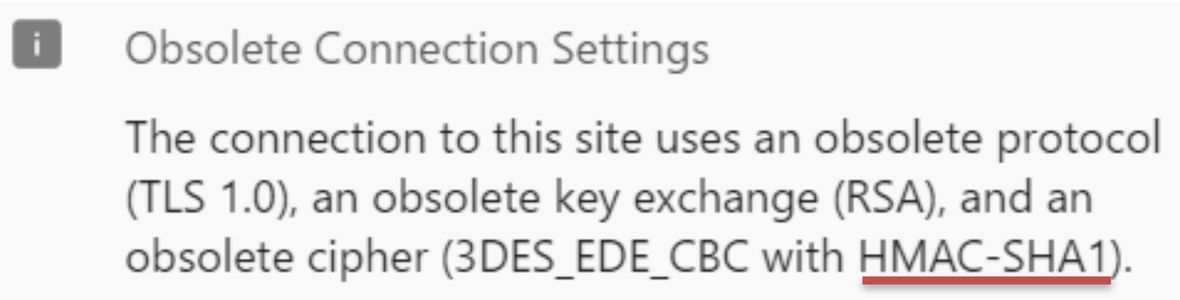
stronger



Hash functions used in TLS (2017)



bu.edu login page (2017)



Source: blog.cloudflare.com/how-expensive-is-crypto-anyway/

Uses of hash functions

Privacy

- Build block ciphers [Even-Mansour]
- Generate random numbers
- Protect passwords
- Forward secrecy of keys

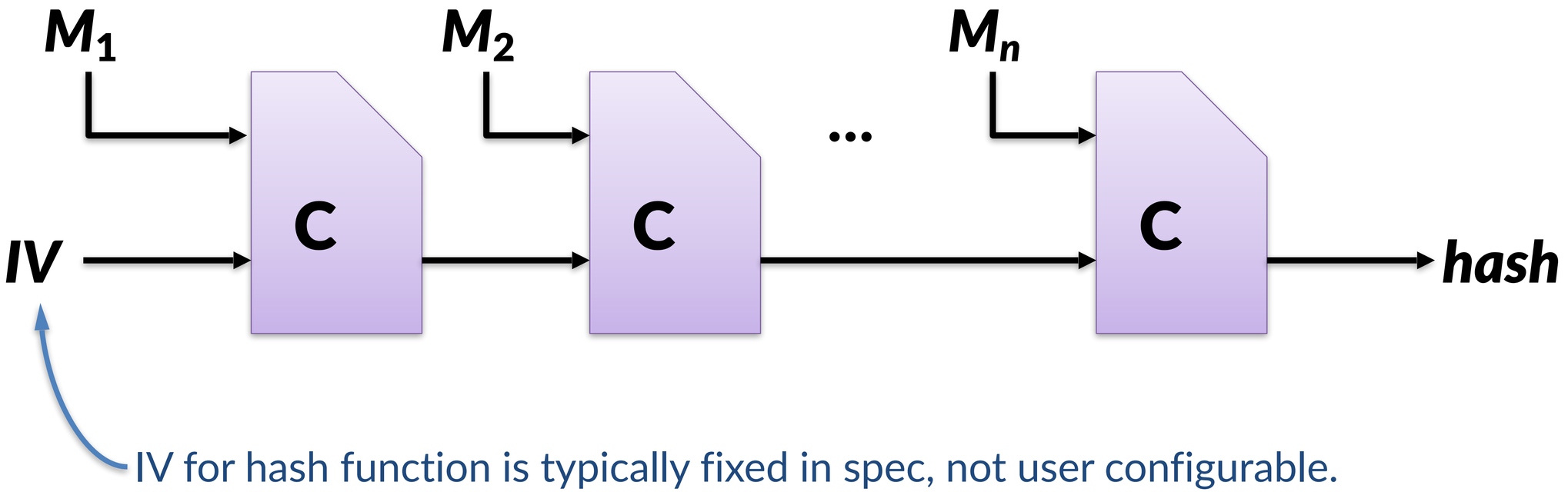
Authenticity

- Detect data modification
- Consensus via proof of work
- Digital signature and MAC (via HMAC)
- …

Merkle-Damgård paradigm

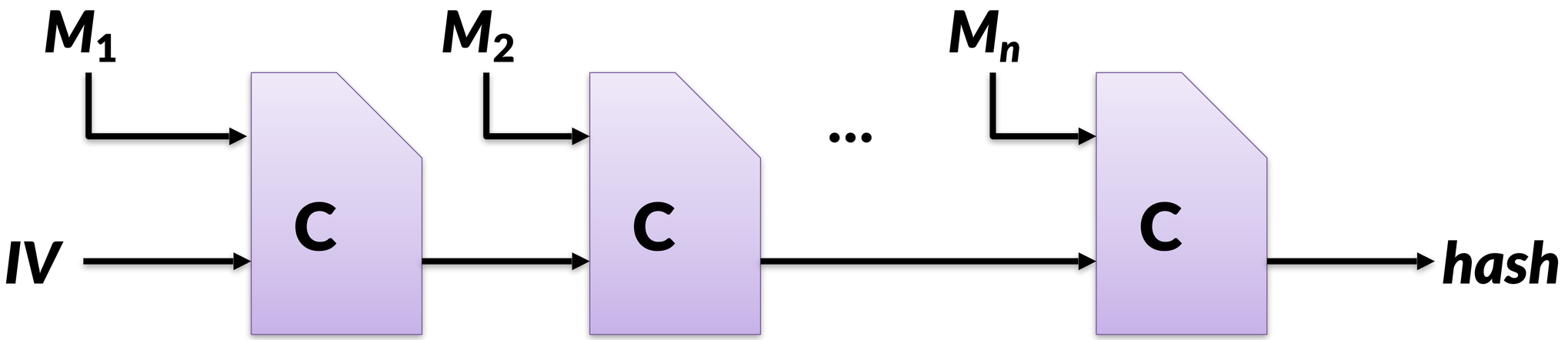
Build a variable-length input hash function from two primitives:

1. A fixed-length, *compressing* random-looking function
2. A mode of operation that iterates this function multiple times in a smart manner



Questions about Merkle-Damgård

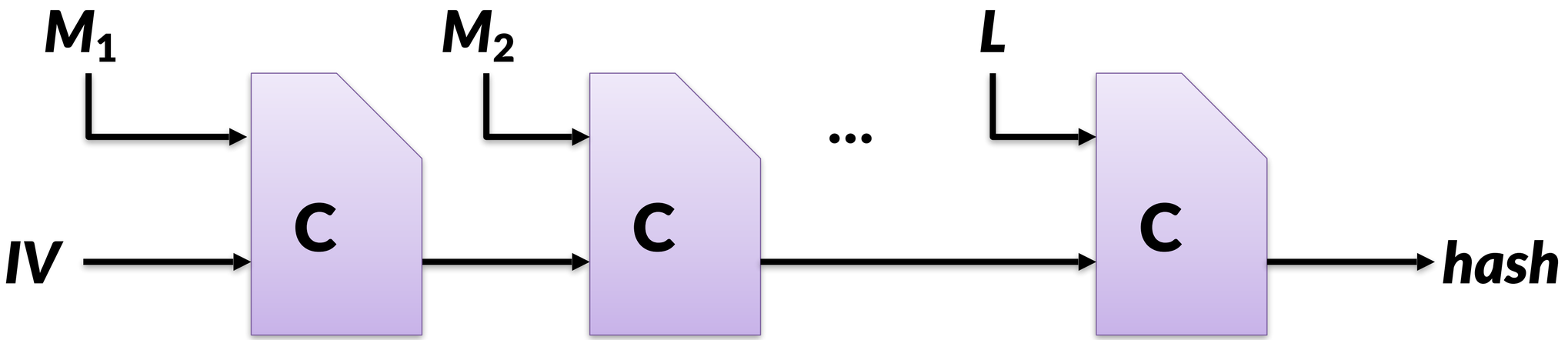
1. Vulnerable to length extension → HMAC adds a “finalization” step
2. How to pad messages in a sensible way?



Mihir Bellare's constraints for collision-free padding

1. Message M is always a prefix of pad(M)
2. If |M1| = |M2|, then |pad(M1)| = |pad(M2)|
3. If |M1| ≠ |M2|, then last block of pad(M1) ≠ last block of pad(M2)

One convention: write string length as the sole contents of final block

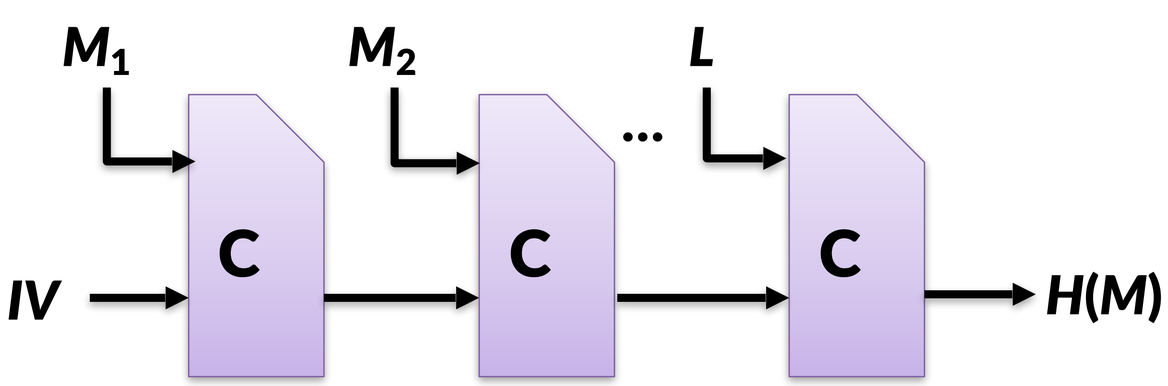


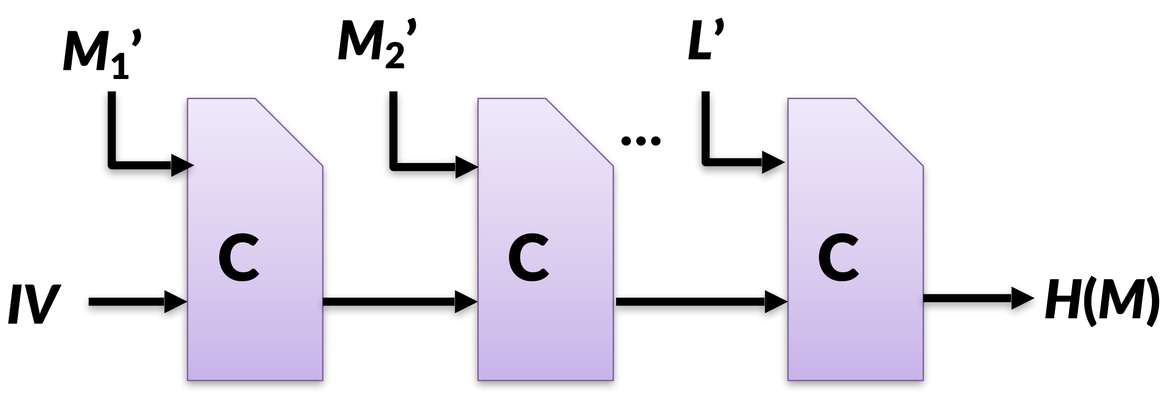
Why Bellare's padding suffices

Thm. If C: $\{0,1\}^{2\eta} \to \{0,1\}^{\eta}$ is collision-resistant, then H: $\{0,1\}^* \to \{0,1\}^{\eta}$ resulting from any fixed number of iterations of Merkle-Damgard is too.

Proof sketch. Show that collision in H $\Rightarrow$ collision in C.

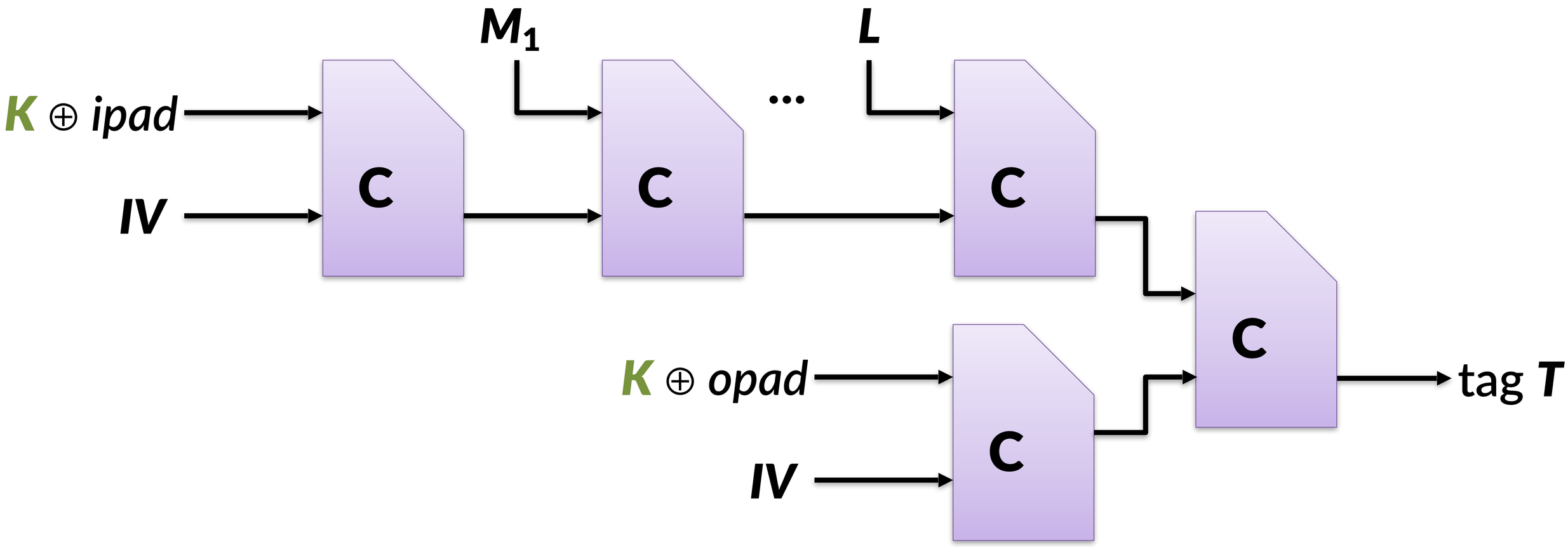
- If |M| ≠ |M'|: Collision in the final steps: C(something || L) and C(something || L')
- If |M| = |M'|: State after each C start out different, become identical somewhere!



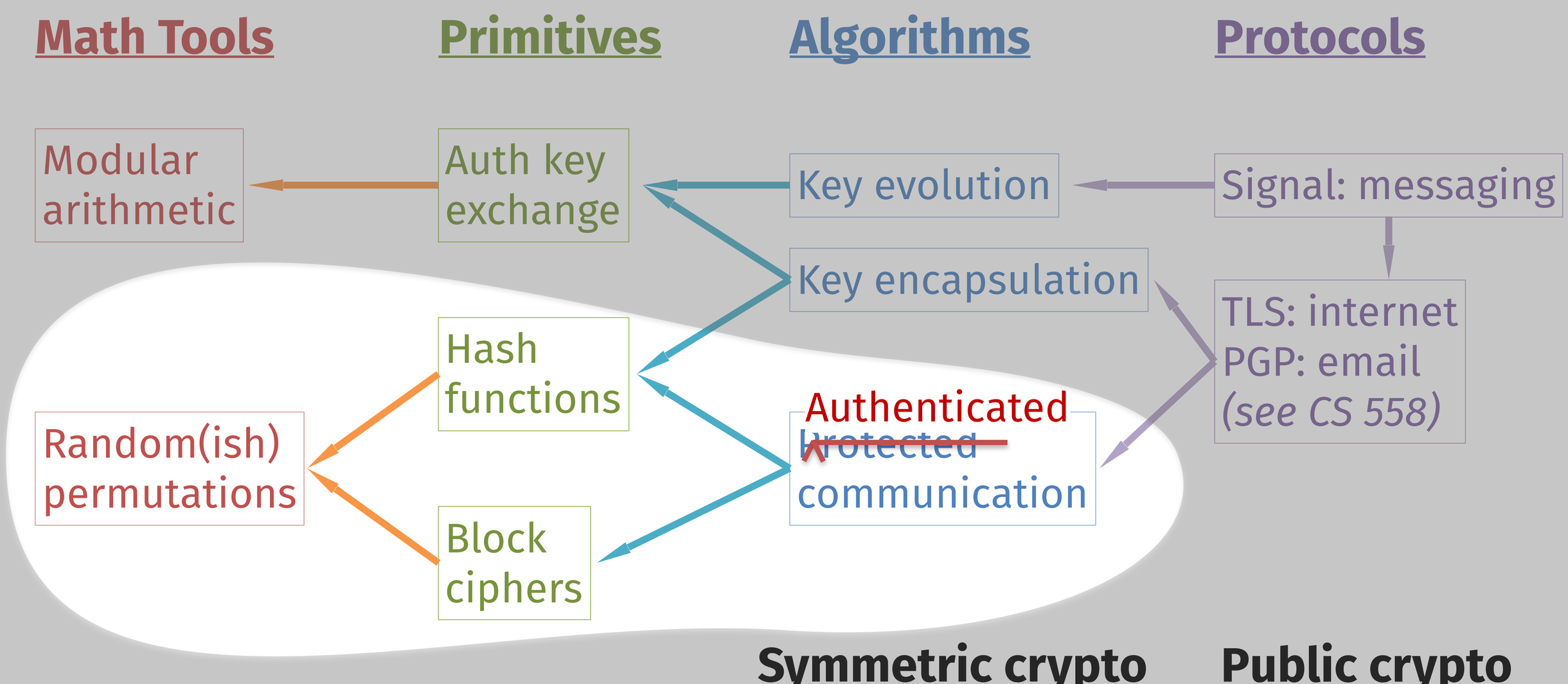


HMAC [Bellare Canetti Krawczyk 97]

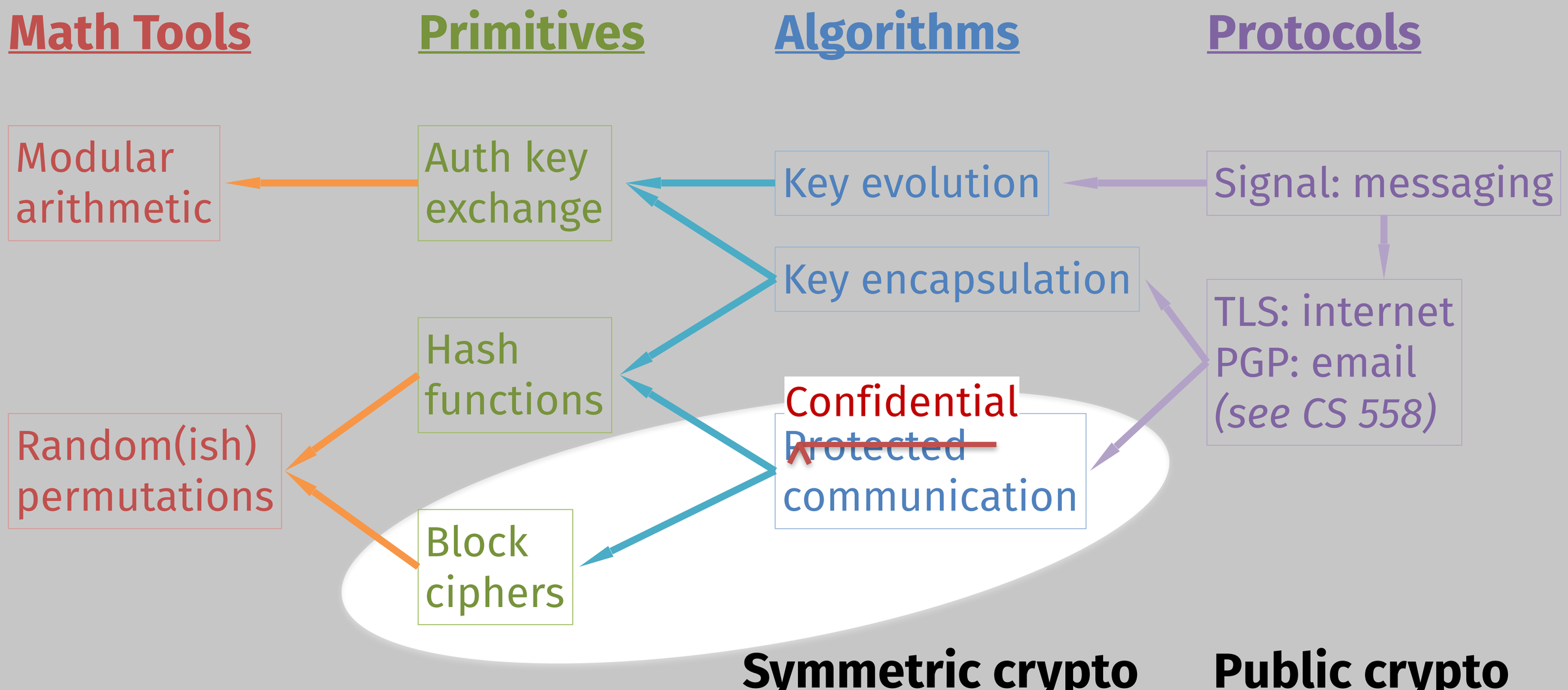
- Compute *H*(key || message)
- Finalize by applying *C* with (a function of) the key again



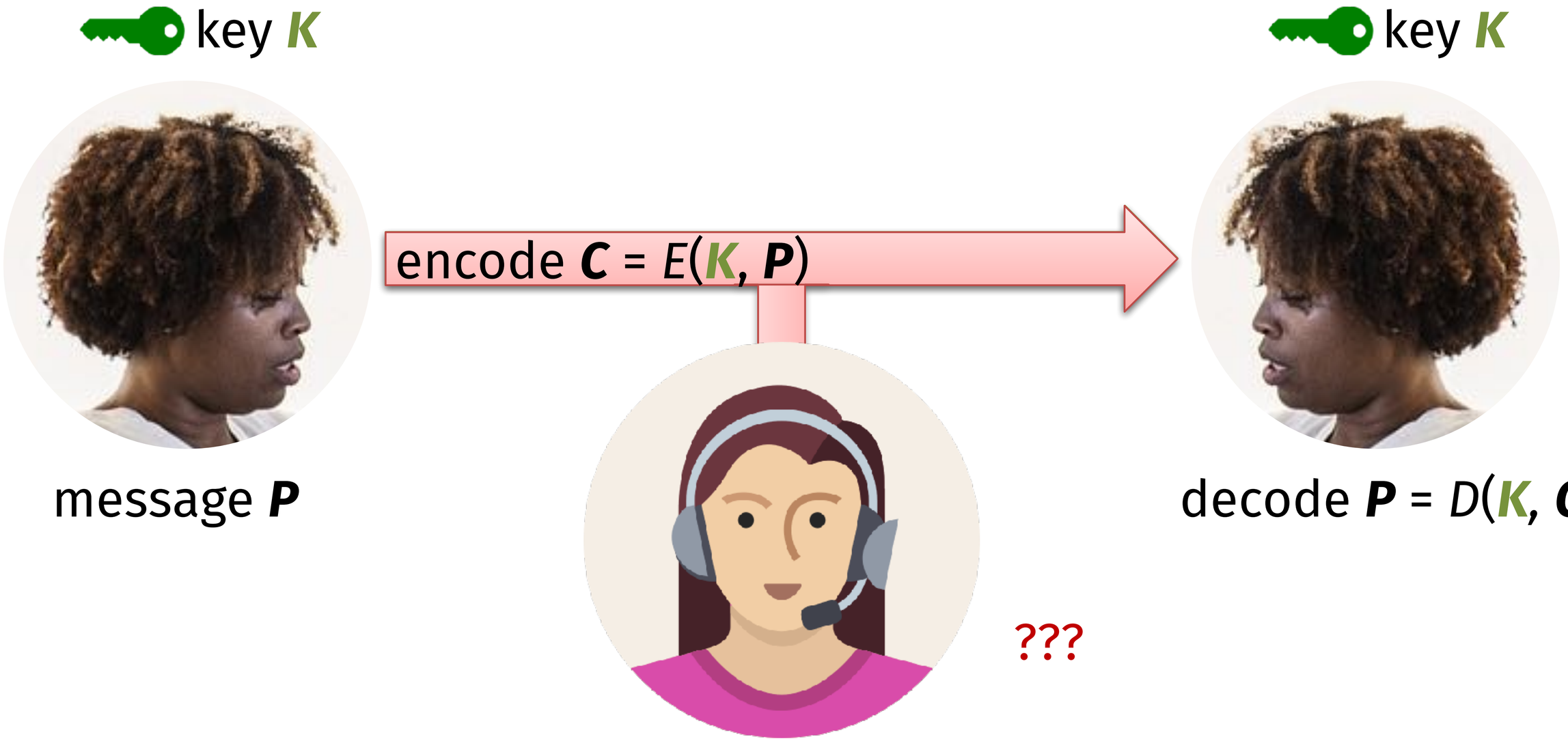
Our progress so far



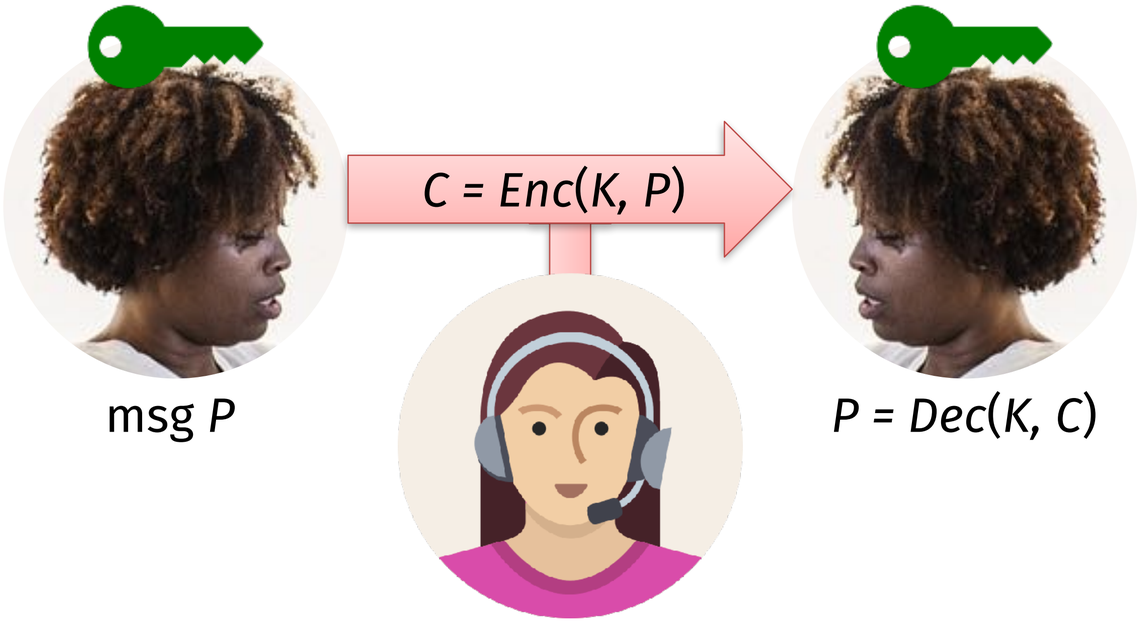
Today: Encryption via enciphering



Part 1: Protecting data at rest



Alice's integrity + confidentiality goals



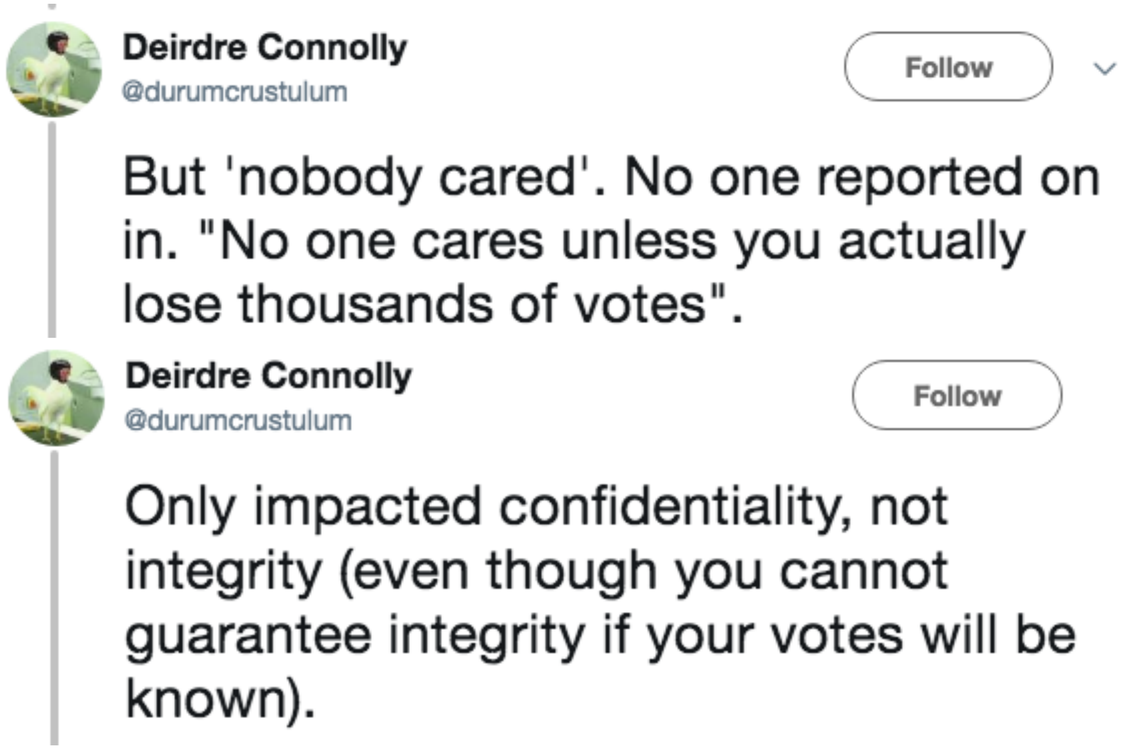
- *Data authenticity*:
  if Eve tampers with *C*, then Alice can detect the change
- *Entity authenticity*:
  future Alice knows that she previously created *C*
- ***Privacy*: Eve cannot learn *P***

Warning!

Confidentiality xor authenticity is not possible. If you don't have both, often you don't have either.

– Prof. Matthew Green, Johns Hopkins

Lack of confidentiality can impact integrity



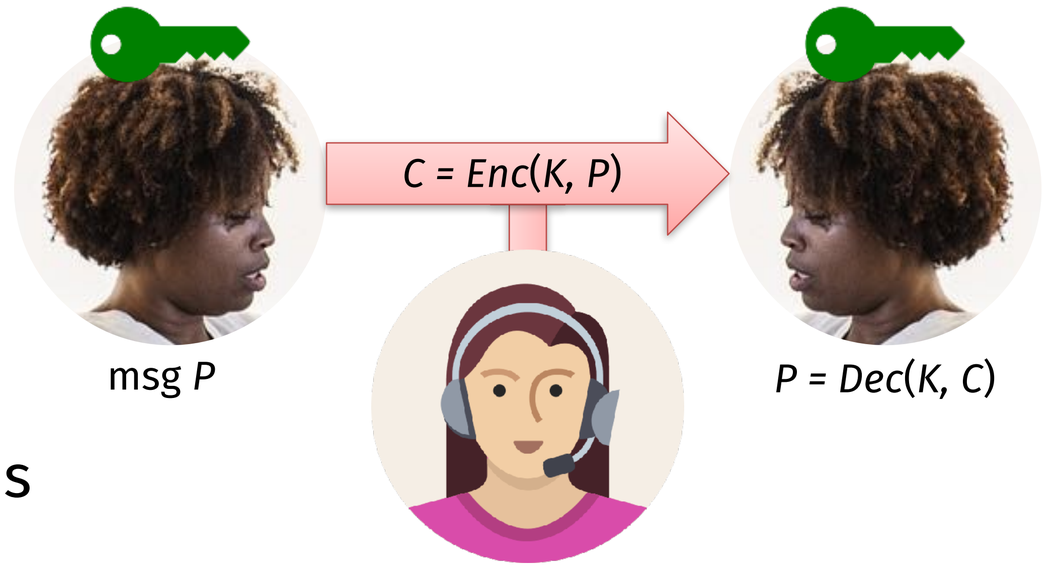
Lack of integrity can impact confidentiality

If you have to perform any cryptographic operation before verifying the MAC on a message you've received, it will somehow inevitably lead to doom!

– Moxie Marlinspike,
"Cryptographic Doom Principle"

Nevertheless, let's press onward…

Recall Auguste Kerchkoffs' principles

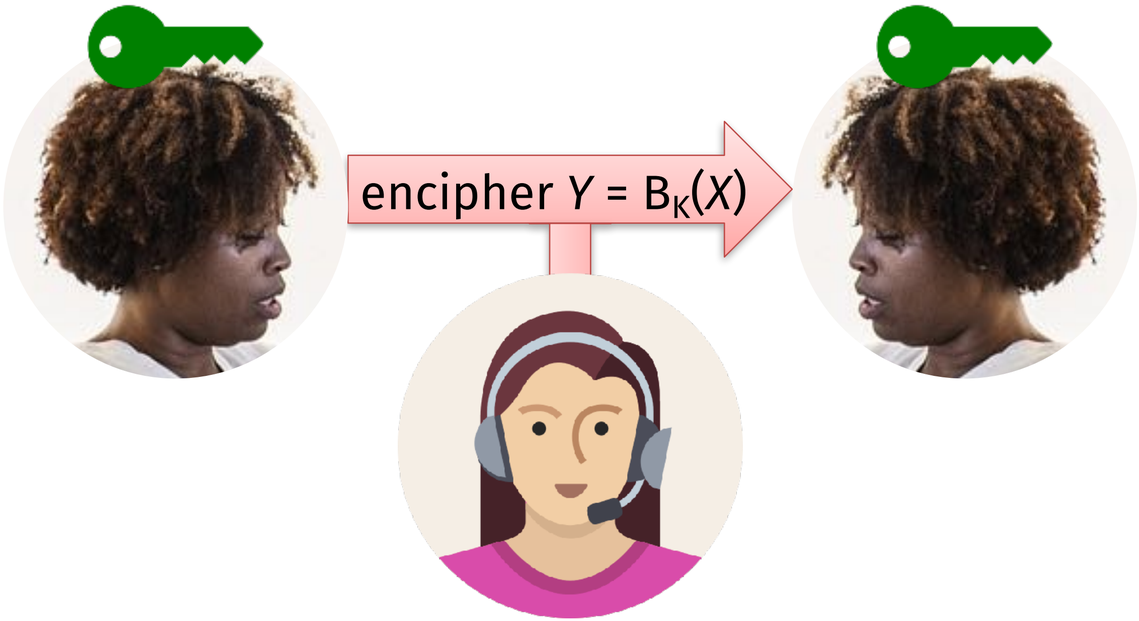


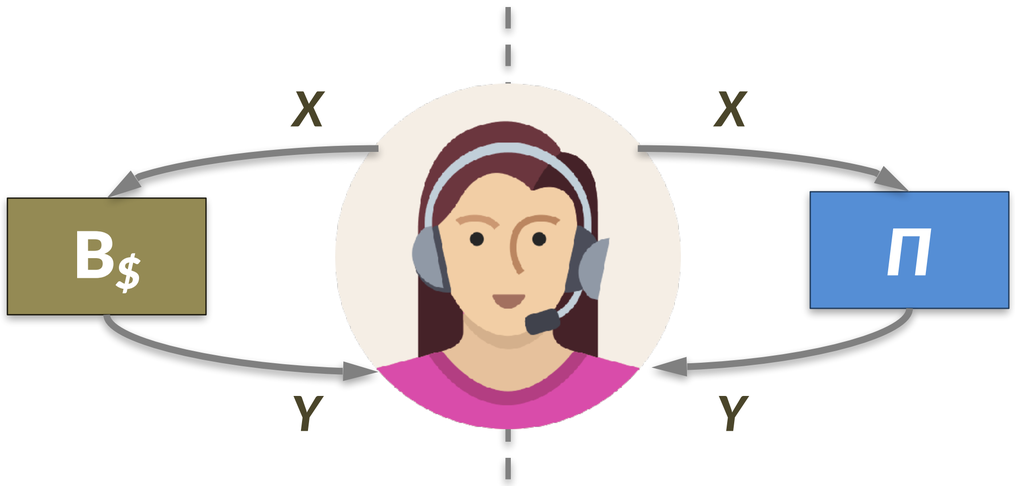
1. The system must be practically, if not mathematically, ***indecipherable***
2. It should ***not require secrecy***, and it should not be a problem if it falls into enemy hands
3. It must be possible to communicate and ***remember the key*** without using written notes, and correspondents must be able to ***change or modify it at will***

   …

6. The system must be easy to use and should ***not be stressful to use*** or require its users to know and comply with a long list of rules

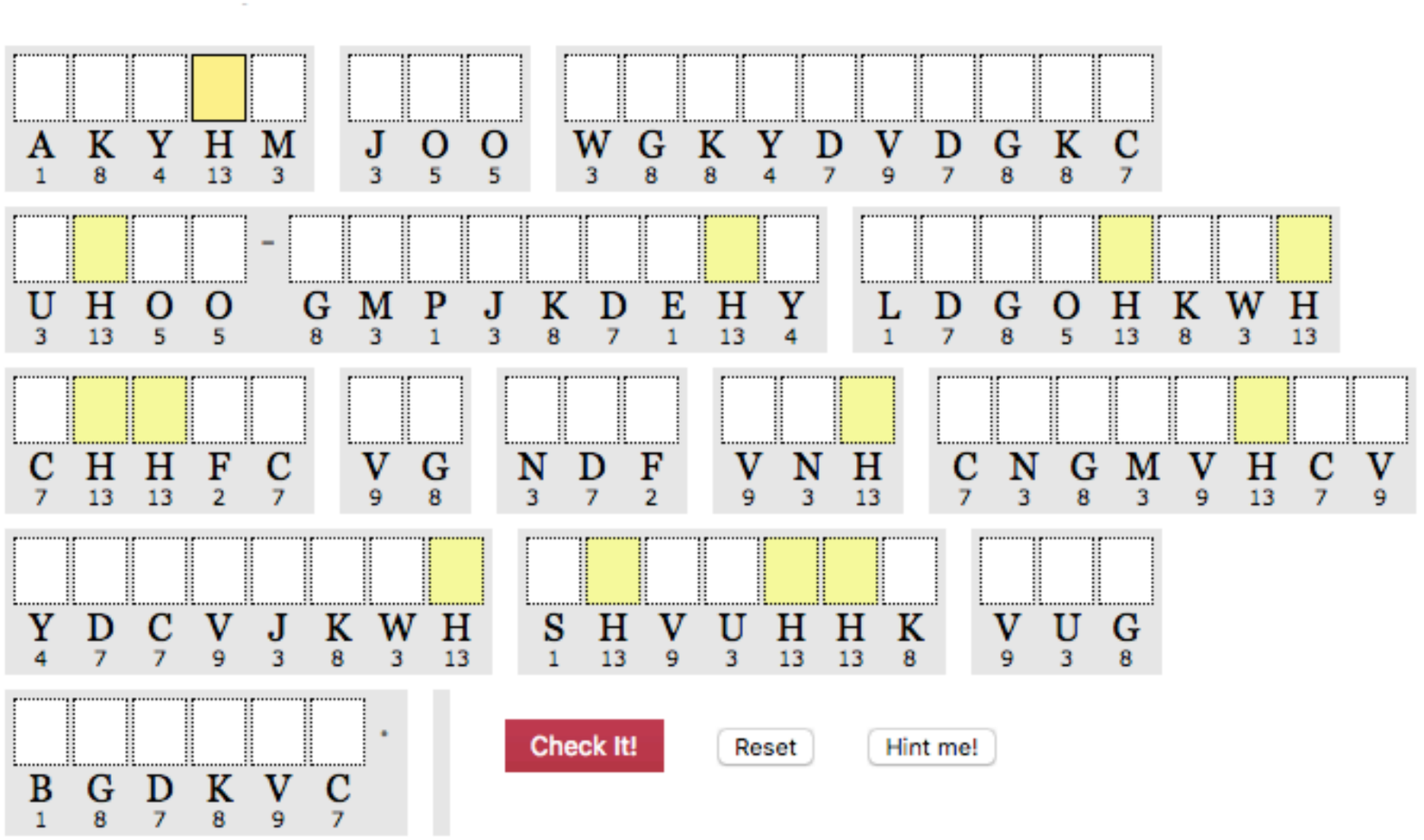
Does a cipher give us confidential communication?





- First reaction: Yes! The cipher transforms *X* in a *confusing* way
- Full answer: Only works if Alice never enciphers the same block twice, and Eve knows that Alice wouldn't do this. Otherwise, Alice must prevent Eve from learning *frequency* of each block

cryptograms.org



Stats

A - 1
B - 1
C - 7
D - 7
E - 1
F - 2
G - 8
H - 13
J - 3
K - 8
L - 1
M - 3
N - 3
O - 5
P - 1
S - 1
U - 3
V - 9
W - 3
Y - 4

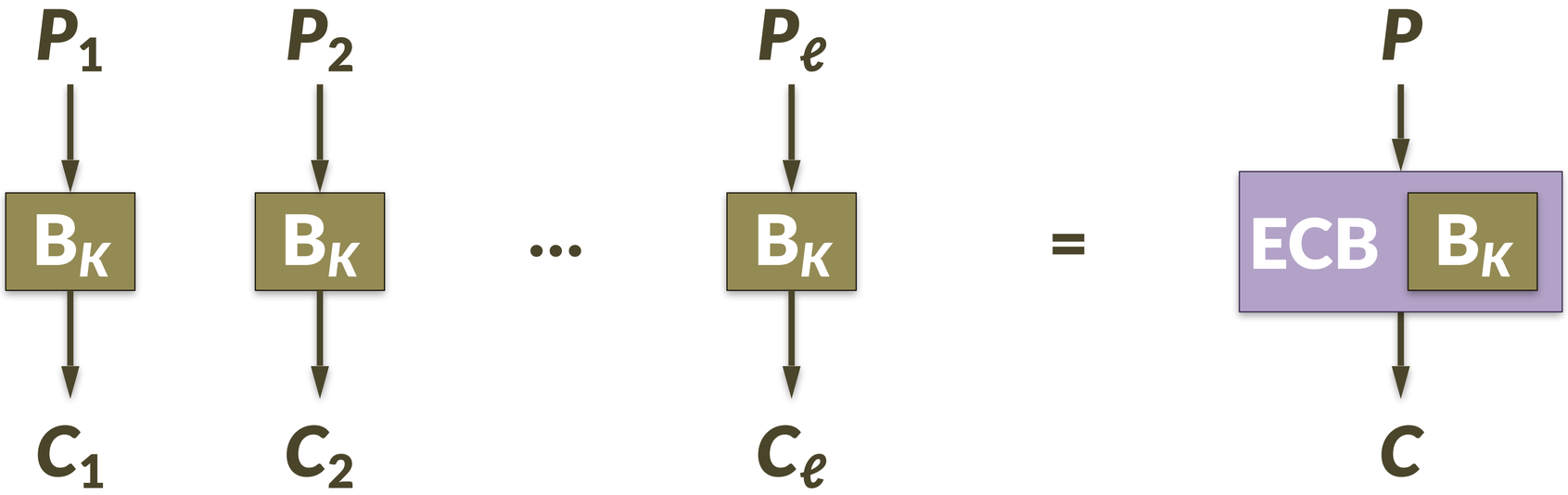
Supporting longer messages

Questions

- What happens with multiple ***P***s?
- How about a long ***P***?
- What exactly is our goal here?
  How can we prove confidentiality?

Idea

- If each block cipher call introduces confusion, simply apply piecemeal
- This is ***Electronic Codebook (ECB)*** mode



ECB mode ⇒ sad Linux penguins

Raw image of
Linux penguin

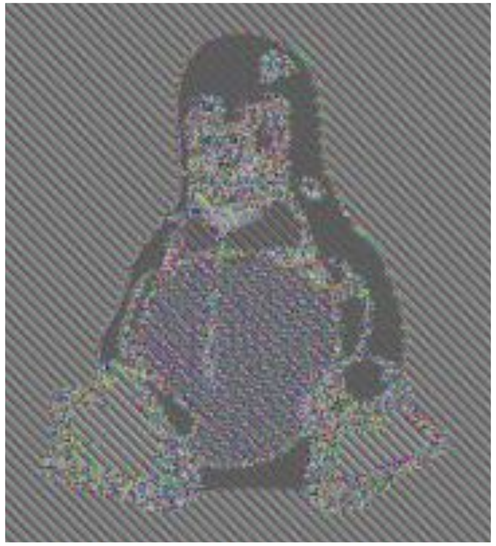
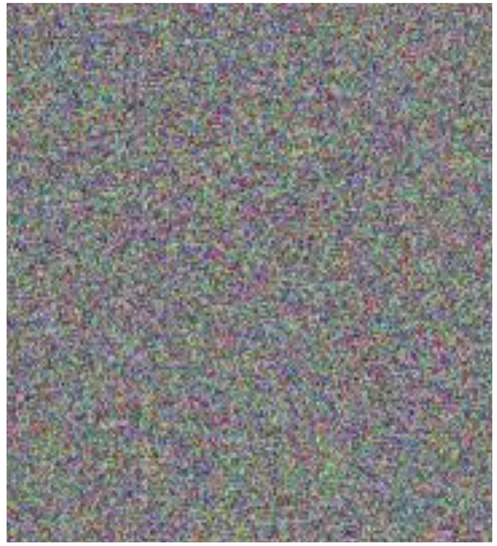


Image after
ECB mode



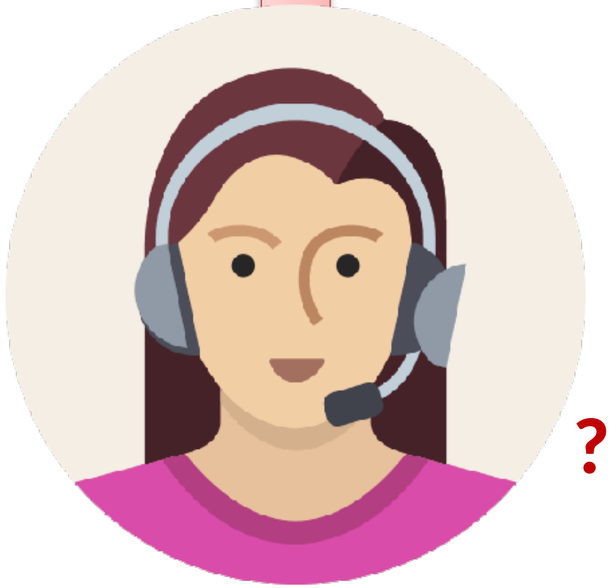
What we want
encryption to do

What if message blocks *don't* repeat?

key K

private data
$P_1, P_2, \ldots P_\ell$

encode $C_i = B_K(P_i)$



???

key K

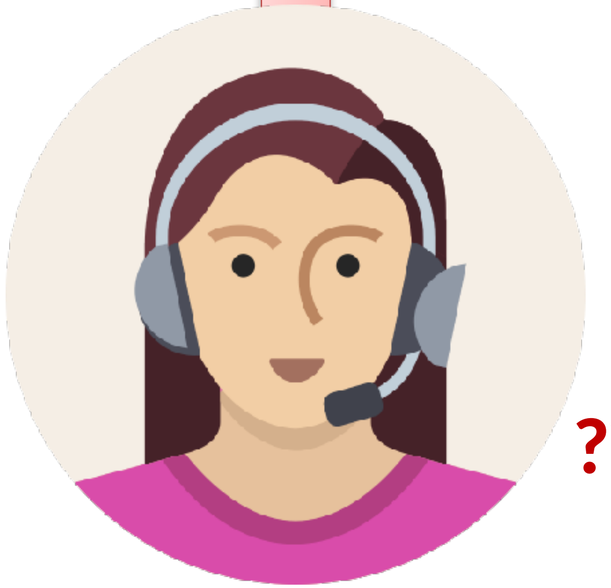
decode $P_i = B_K^{-1}(C_i)$

What if message blocks *don't* repeat?

key K

private data
$P_1, P_2, \ldots P_\ell$

encode $C_i = \Pi(P_i)$



???

key K

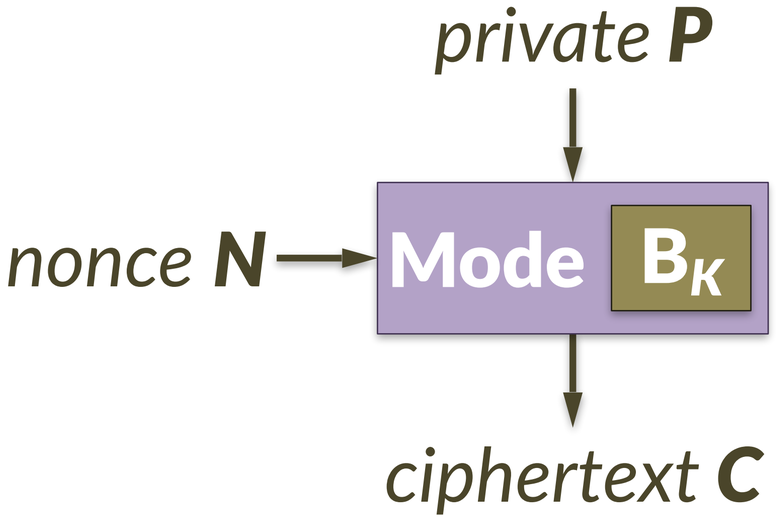
decode $P_i = \Pi^{-1}(C_i)$

Question: *How do we guarantee that message blocks don't repeat?*

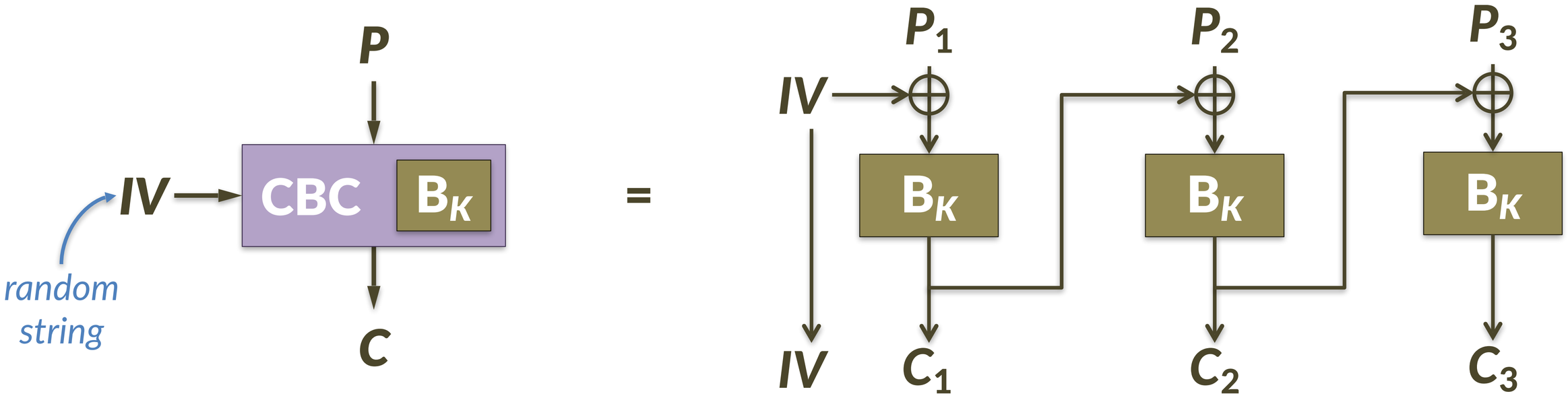
Lessons learned

- **Randomness matters:** We can confuse Eve! Just need to design a mode of operation that guarantees each enciphered block is unique.

- **Definitions matter:** Our argument leveraged the concept that a block cipher “looks like” a random permutation from Eve’s point of view.



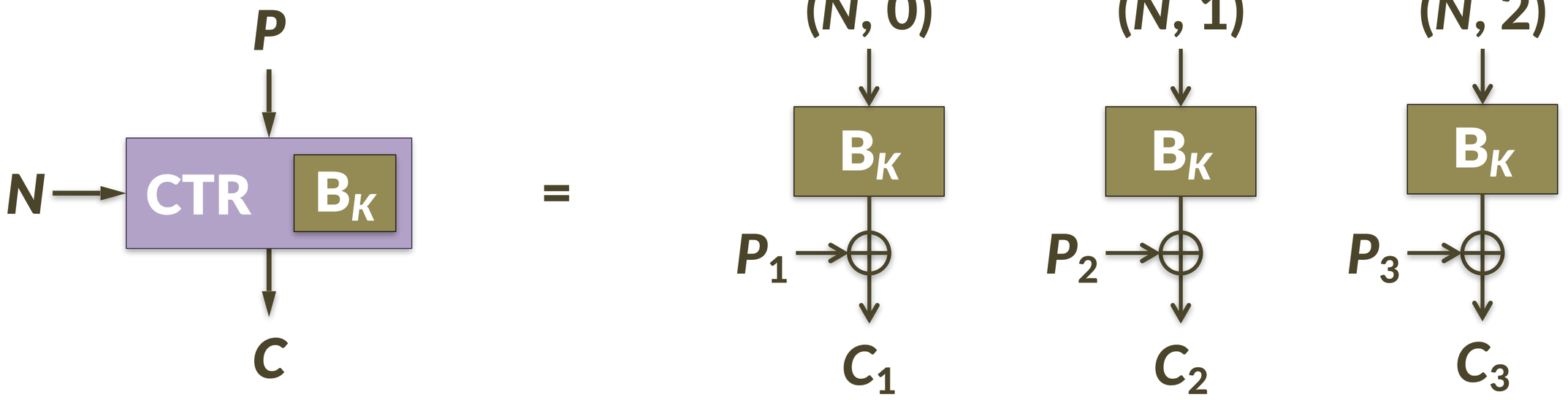
Cipher block chaining (CBC) mode



Two differences from CBC-MAC:

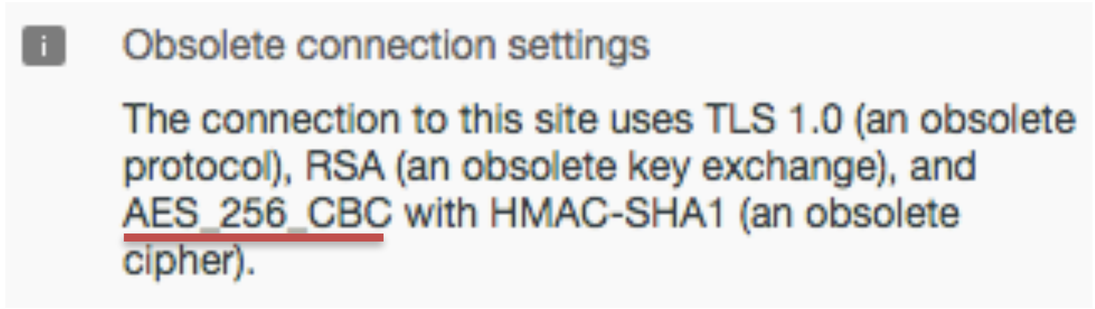
1. All blocks are output
2. First block is protected by a public, randomly chosen *initialization vector*

Counter (CTR) mode



Encryption in practice

bu.edu homepage (2017)



www.amazon.com