Encrypting/Decrypting, con't

CS 161: Computer Security

Prof. Vern Paxson

TAs: Jethro Beekman, Mobin Javed,
Antonio Lupher, Paul Pearce
& Matthias Vallentin

http://inst.eecs.berkeley.edu/~cs161/

March 14, 2013

Announcements / Goals For Today

- It's prudent to *double-check* the summing used on your midterm scores
- In general, see course homepage for procedure to request re-grading
- For Project #1, be sure to read Neo's tweets
- Today:
  - Review main themes from last lecture
  - Stream ciphers: approximating one-time pads
  - Begin public-key cryptography: no shared keys!

Review of Where We're At

- Alice employs an Encryptor E to produce *ciphertext* from *plaintext*.
- Bob employs a Decryptor D to recover plaintext from ciphertext.

- So far, both E and D are configured using the same key **K**.
- **K** is a shared secret between Alice and Bob
  - Eavesdropper Eve doesn't know it (otherwise, **disaster!**)
- Use of same secret key for E and D ⇒ "symmetric key cryptography"

One-Time Pad

- For each message, use a new, independent key
- Key has same length as message; $C = M \oplus K$
- Provably secure … *if* we stick to these requirements
- Reuse of key = disaster
  - Eve can learn about *relationships* between messages
  - For known plaintext attacks, Eve can recover K and thus new message encrypted with its reuse

- *Not practical in most circumstances due to requiring huge volume of shared secret keys*

Block Ciphers

- Given a key, creates a seemingly random bijective mapping on bitstrings of length n

  $\{0, 1\}^n \rightarrow \{0, 1\}^n$ n is the *blocksize*

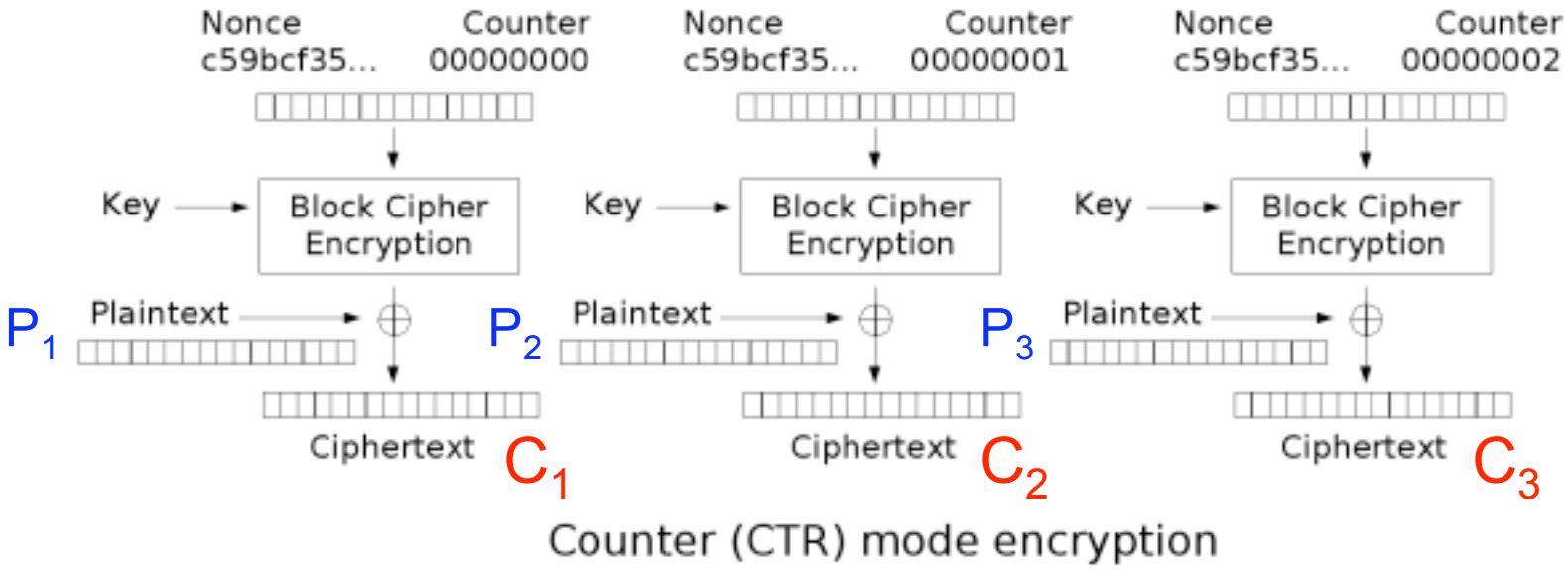
- A popular, solid block cipher: AES
  - Block size n = 128 bits
  - Keys can be 128/192/256 bits (all work fine)
  - As usual, AES defines both an encryptor and a decryptor function. They use the same key K.
- AES is very strong: no known significant flaws
  - Change single bit in key or plaintext ⇒ output appears completely different
- In general, we think of block ciphers as one form of "primitive" that we then build on

Block Cipher Modes

- Problem with using block ciphers: what if message M not the same size as n (blocksize)?
  - If M smaller than n, we pad it
    - Don't worry about how to choose the padding or figure out the exact length (these aren't hard)
  - If M larger than n, need multiple instances of block cipher
- A block cipher mode = procedure to use multiple instances of block cipher to encrypt a single msg
- Modes are tricky! We want to make sure they don't **leak** information:
  - Not give away parts of message that are related
  - Not give away relationships between different messages

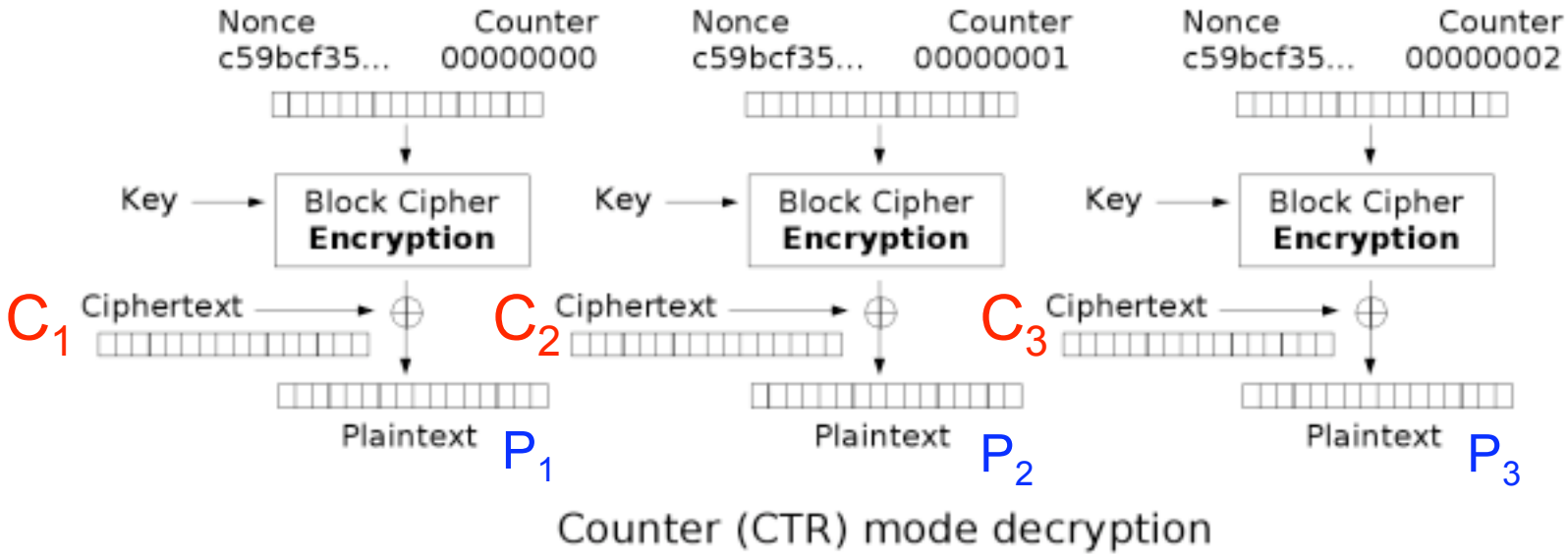
Block Cipher Modes, con't

- Simplest mode (ECB) **fails** (recall penguin image)
- To mask relationships between different blocks of M, ensure each block is "seeded" with something (very-)hard-to-predict
  - Lecture mentioned "intermingling"; that's *one* way to do it
- To mask relationships between different messages, seed each use of a block cipher mode w/ unique #
  - Initialization Vector (IV)
  - A type of nonce: a value used only once (single context)
- Important: IV is not sensitive!
  - Alice sends (IV, C) to Bob. *Eve can see IV*.
- This led us to CBC = *Cipher Block Chaining* mode



Counter (CTR) mode encryption

(Nonce = Same as IV)



Counter (CTR) mode decryption

(Note, uses block cipher's *encryption* functionality, not decryption)