More on Cryptography
CS 236
On-Line MS Program
Networks and Systems Security

Peter Reiher

Outline

- Desirable characteristics of ciphers
- Stream and block ciphers
- Cryptographic modes
- Uses of cryptography
- Symmetric and asymmetric cryptography
- Digital signatures

Desirable Characteristics of Ciphers

- Well matched to requirements of application
  - Amount of secrecy required should match labor to achieve it
- Freedom from complexity
  - The more complex algorithms or key choices are, the worse

More Characteristics

- Simplicity of implementation
  - Seemingly more important for hand ciphering
  - But relates to probability of errors in computer implementations
- Errors should not propagate

Yet More Characteristics

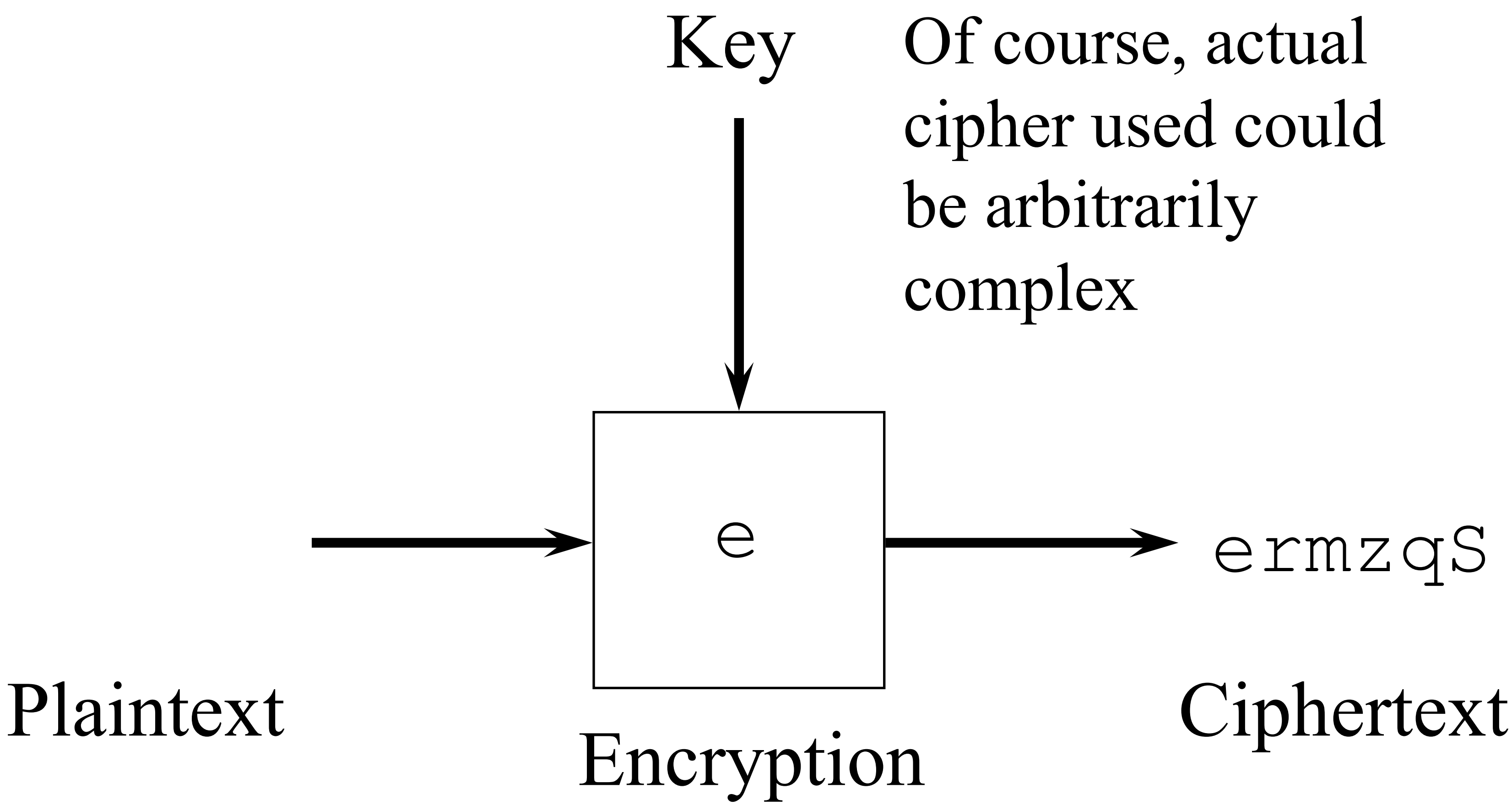
- Ciphertext size should be same as plaintext size
- Encryption should maximize *confusion*
  - Relation between plaintext and ciphertext should be complex
- Encryption should maximize *diffusion*
  - Plaintext information should be distributed throughout ciphertext

Stream and Block Ciphers

- Stream ciphers convert one symbol of plaintext immediately into one symbol of ciphertext
- Block ciphers work on a given sized chunk of data at a time

Stream Ciphers

Of course, actual cipher used could be arbitrarily complex



Advantages of Stream Ciphers

+ Speed of encryption and decryption
  - Each symbol encrypted as soon as it's available
+ Low error propagation
  - Errors affect only the symbol where the error occurred
    - Depending on *cryptographic mode*

Disadvantages of Stream Ciphers

- Low diffusion
  - Each symbol separately encrypted
  - Each ciphertext symbol only contains information about one plaintext symbol
- Susceptible to insertions and modifications
- Not good match for many common uses of cryptography
- Some disadvantages can be mitigated by use of proper cryptographic mode

Sample Stream Cipher: RC4

- Creates a changing key stream
  - Supposedly unpredictable
- XOR the next byte of the key stream with the next byte of text to encrypt
- XOR ciphertext byte with same key stream byte to decrypt
- Alter your key stream as you go along

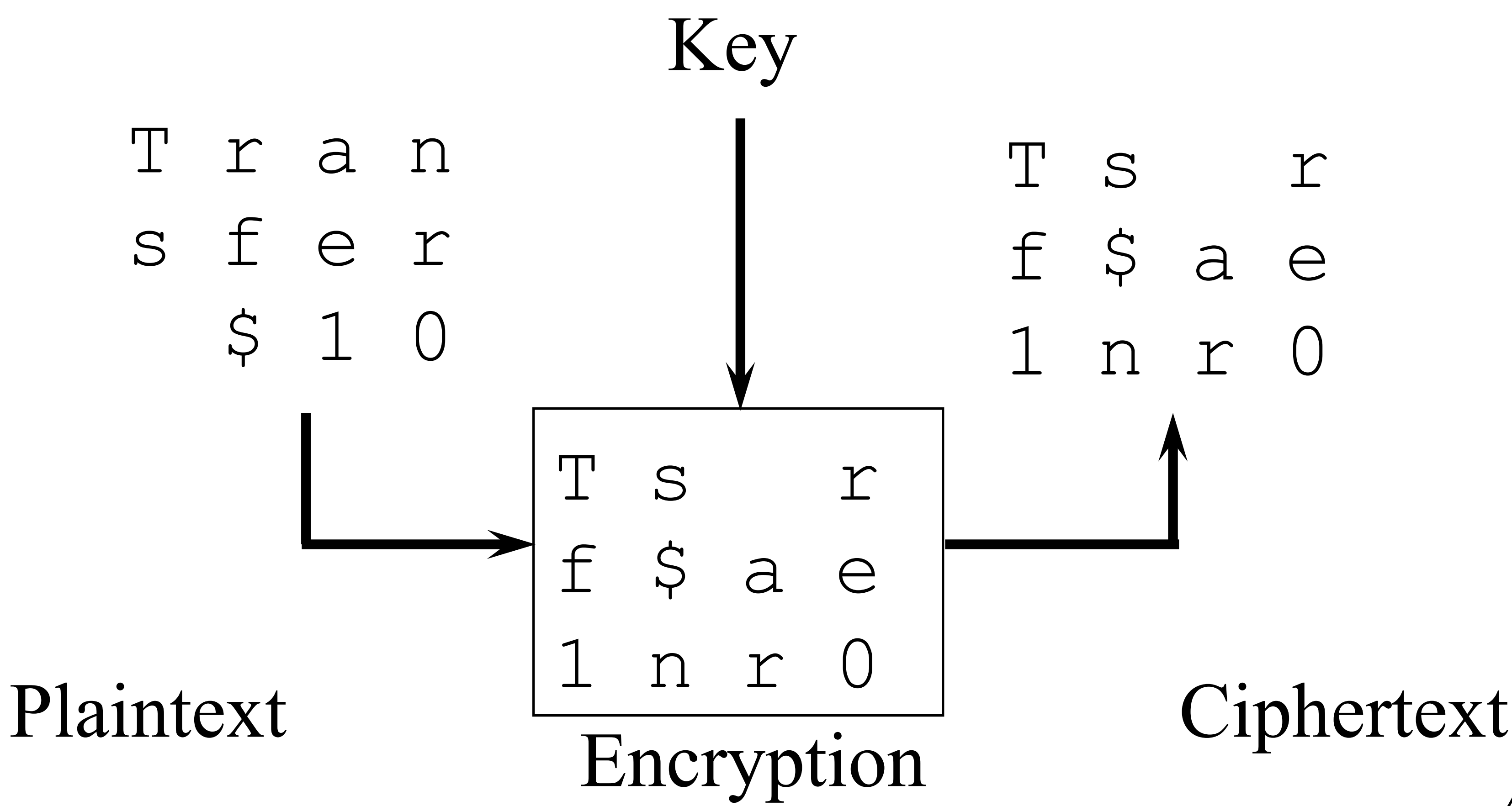
Creating an RC4 Key

- Fill an 256 byte array with 0-255
- Choose a key of 1-255 bytes
- Fill a second array with the key
  – Size of array depends on the key
- Use a simple operation based on the key to swap around bytes in the first array
- That produces the key stream you'll use
- Swap two array bytes each time you encrypt

Characteristics of RC4

- Around 10x faster than DES
- Significant cryptographic weakness in its initial key stream
  - Fixable by dropping the first few hundred of the keys
- Easy to use it wrong
  - Key reuse is a serious problem

Block Ciphers



Advantages of Block Ciphers

+ Good diffusion
  - Easier to make a set of encrypted characters depend on each other
+ Immunity to insertions
  - Encrypted text arrives in known lengths

Most common Internet crypto done with block ciphers

Disadvantages of Block Ciphers

- Slower
  - Need to wait for block of data before encryption/decryption starts
- Worse error propagation
  - Errors affect entire blocks

Cryptographic Modes

- Let’s say you have a bunch of data to encrypt
  – Using the same cipher and key
- How do you encrypt the entire set of data?
  – Given block ciphers have limited block size
  – And stream ciphers just keep going

The Basic Situation

J2?@=4l	Dor72m/	Sv&`>oo	Xl3lu*m
sS^0’sq	2ci;aE9	sS^0’sq	#rdL04,

Let’s say our block cipher has a block size of 7 characters and we use the same key for all

Now let’s encrypt

There’s something odd here . . .

Is this good? *Why did it happen?*

Another Problem With This Approach

What if these are transmissions representing deposits into bank accounts?

Xl3lu*m
#rdL04,

Insertion Attack!

Dor72m/
2ci;aE9

1840326	450
2201568	5000
3370259	8900
5610993	**1579**
6840924	2725
8436018	10

What if account 5610993 belongs to him?

So far, so good . . .

What Caused the Problems?

- Each block of data was independently encrypted
  - With the same key
- So two blocks with identical plaintext encrypt to the same ciphertext
- Not usually a good thing
- We used the wrong *cryptographic mode*
  - Electronic Codebook (ECB) Mode

Cryptographic Modes

- A cryptographic mode is a way of applying a particular cipher
  - Block or stream
- The same cipher can be used in different modes
  - But other things are altered a bit
- A cryptographic mode is a combination of cipher, key, and feedback
  - Plus some simple operations

So What Mode Should We Have Used?

- Cipher Block Chaining (CBC) mode might be better
- Ties together a group of related encrypted blocks
- Hides that two blocks are identical
- Foils insertion attacks

Cipher Block Chaining Mode

- Adds feedback into encryption process
- The encrypted version of the previous block is used to encrypt this block
- For block X+1, XOR the plaintext with the ciphertext of block X
  - Then encrypt the result
- Each block's encryption depends on all previous blocks' contents
- Decryption is similar

What About the First Block?

- If we send the same first block in two messages with the same key,
  - Won't it be encrypted the same way?
- Might easily happen with message headers or standardized file formats
- CBC as described would encrypt the first block of the same message sent twice the same way both times

Initialization Vectors

- A technique used with CBC
  - And other crypto modes
  - Abbreviated IV
- Ensures that encryption results are always unique
  - Even for duplicate message using the same key
- XOR a random string with the first block
  - $plaintext \oplus IV$
  - Then do CBC for subsequent blocks

Encrypting With An IV

First block of message

1	1	0	1	0	0	0	1

Initialization vector

0	1	0	0	1	1	0	0

XOR IV and message

0	0	1	1	0	1	1	1

Encrypt msg and send IV plus message

Second block of message

0	0	0	1	1	0	0	0

Use previous msg for CBC

Apply CBC

1	0	0	1	1	1	1	0

Encrypt and send second block of msg

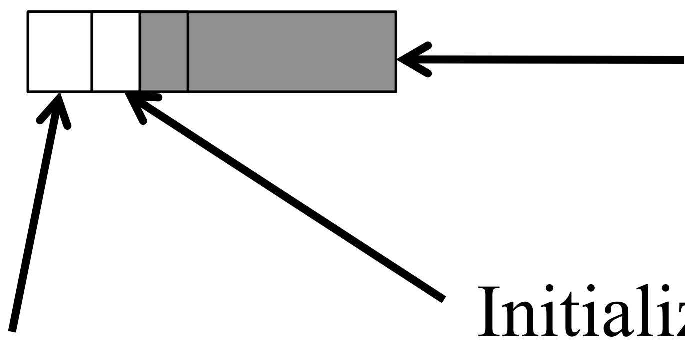
No need to also send 1st block again

How To Decrypt With Initialization Vectors?

- First block received decrypts to

  $P = plaintext \oplus IV$
- $plaintext = P \oplus IV$
- No problem if receiver knows *IV*
  - Typically, *IV* is sent in the message
- Subsequent blocks use standard CBC
  - So can be decrypted that way

An Example of IV Decryption



And XOR with the plaintext IV

1	1	0	1	0	0	0	1

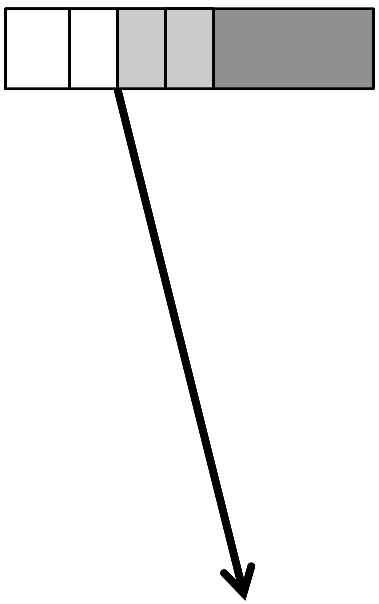
0	1	0	0	1	1	0	0

1	0	0	1	1	1	0	1

The message probably contains multiple encrypted blocks

Now decrypt the message

For Subsequent Blocks



Use previous ciphertext block instead of IV

And XOR with the previous ciphertext block

0	0	0	1	1	0	0	0

0	0	1	1	0	1	1	1

1	1	0	0	0	0	1	1

Now decrypt the message

Some Important Crypto Modes

- Electronic codebook mode (ECB)
- Cipher block chaining mode (CBC)
- Cipher-feedback mode (CFB) and Output-feedback mode (OFB)

  Both convert block to stream cipher