Classical and Symmetric Crypto

Bryan Vanek

What exactly is Cryptography?

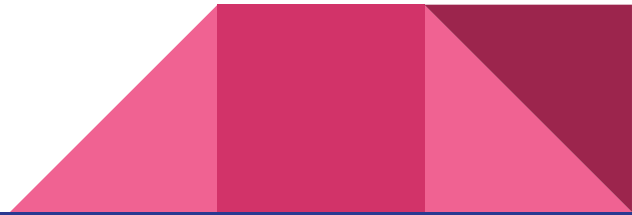
- Formally, it’s the practice/study of different ways to create secure communications between two different parties in the presence of an adversary
- You always pretend that there’s someone who can listen in on the communication (the ‘Alice & Bob problem)
- Has been around for thousands of years

Some Terminology

- Components of cryptography can be broken down into four categories:
  - **Ciphers:** a form of disguising a message. Somewhat similar to **obfuscation.** Provides *Prevention*.
  - **Encoding/Data representation:** a way to transform data into something easier to digest and interpret, but using a known, easily reversibly algorithm. Provides *Availability*.
  - **Encryption:** a way to transform data while keeping it secret from others. Usually involves the use of a key. Provides *Confidentiality*.
  - **Hashing:** a way to detect if data has been modified. Provides *Integrity*. (Will cover this if we have time)

Some More Terminology

- **Plaintext** - unencrypted information. Written in lowercase by convention.
  - You might also see this called “**in the clear**”
- **Ciphertext** - encrypted information. Written in UPPERCASE by convention
- Since we’re starting with classical ciphers, we’ll be dealing with literal text to start out, but the terms apply equally well to computer data, and it’s still called “plaintext” and “ciphertext”
- **Key** - the secret value that allows you to encrypt or decrypt something
- **Cryptanalysis** - breaking encryption



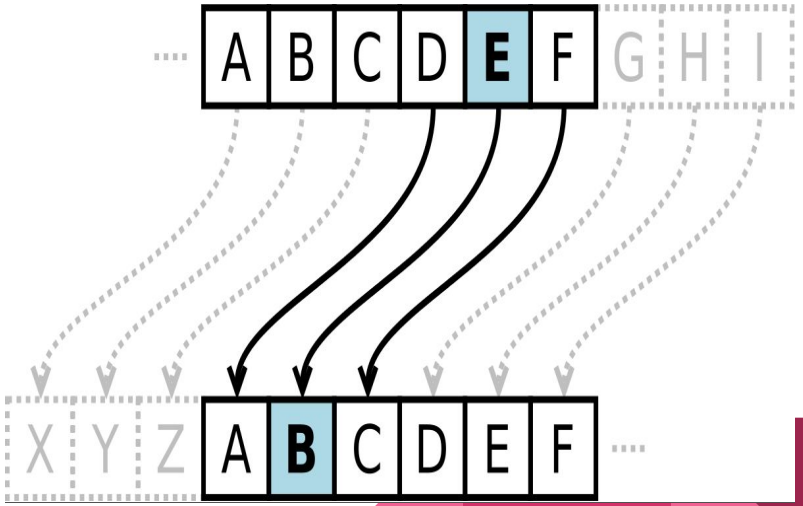
Ciphers

- **Monoalphabetic Cipher:** the Caesar Cipher
  - Used all the way back in 58 BC!
  - Involves the use of a **shift**
  - **ROT13** is a form of this
- Reeeeally easy to break
  - There are only 26 keys in the English alphabet
  - A cryptosystem that is to be used in the wild should effectively be impossible to crack
  - Keep in mind though that these are used to provide *prevention*, and not necessarily true security.

UMBC is a super cool school!

-- Apply a left shift of 3…

RJYZ fp x prmbo zlli pzelli!



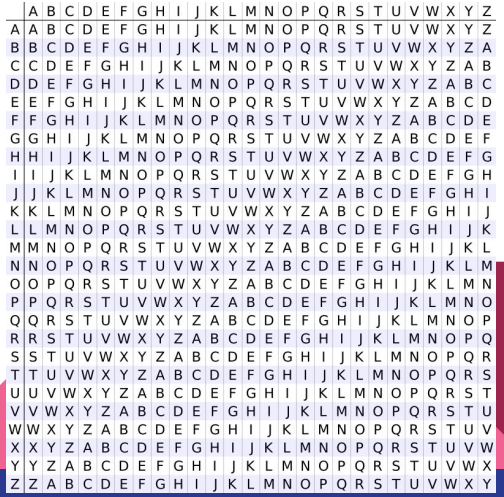
Ciphers

- **Polalphabetic ciphers**: Vigenere Cipher
  - Made in the 16th Century, a little harder to break
  - One of the first examples of using a key, even if it is a physical one…
  - Still uses a shift, but is based on the key now!

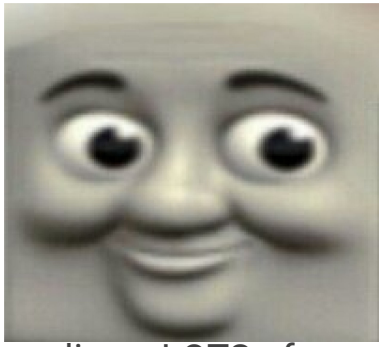
UMBC is a super cool school!

-- using a keyword ‘retriever’
-- retriever = ‘17 4 19 17 8 4 21 4 17’
-- keyword is 9 letters long; apply each of these shifts and repeat!

LQUT qw v wlgik twsg wtyshc!



Ciphers



- **Transpositional ciphers:** Railfence cipher
  - Performing clever transformations of the alphabet/sentence using shapes or lines. LOTS of these exist!
  - In this case, we take a sentence and turn it into a fence or rail tracks, then squish it back together!
  - You would only need to figure out the **rails** and **offset**

UMBC is a super cool school!

-- 3 rails and an offset of 2

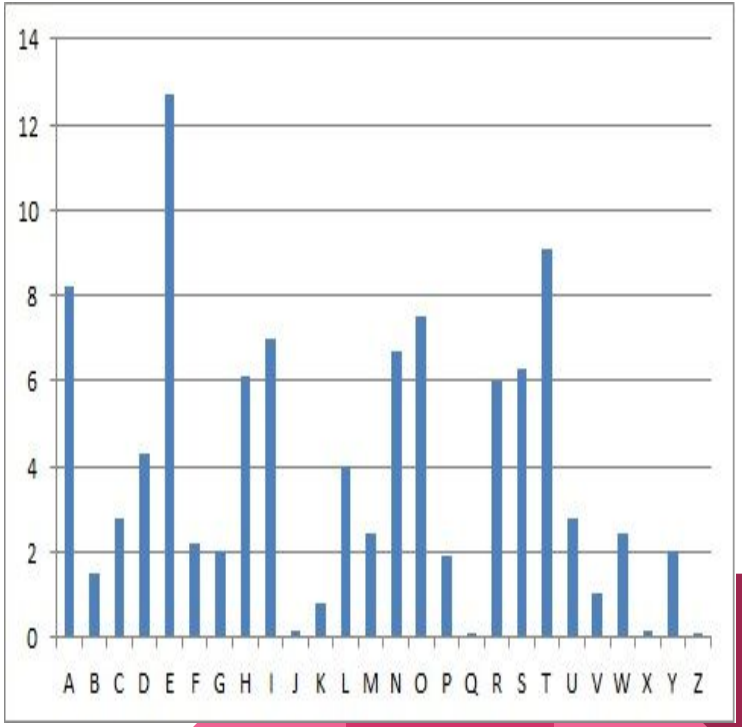
```
B   s   s   r   o   c   l
 M C i     u e     o l s h o !
U       a   p   c       o
```

-- We then go back across the rows and make our new message!

BssroclMCi ue olsho!U apc o

Problems with using ciphers alone

- NOT real cryptography
- Many forms of cryptanalysis exist to break these
  - If the keyspace is too small, you can always just brute force it
  - Since we're dealing with a known language, both monoalphabetic and polyalphabetic ciphers can be busted with a technique known as frequency analysis
- If someone knows the key (or even part of it), you're in trouble!



Encoding and Data Representation

- This also is NOT Cryptography!!
- Rather, we view this as ways to represent data so that computers and other applications can better interpret and digest it!
- Some examples:
  - **ASCII:** UMBC is a super cool school!
  - **Decimal:** 85 77 66 67 32 105 115 32 97 32 115 117 112 101 114 32 99 111 111 108 32 115 99 104 111 111 108 33
  - **Hexadecimal:** 55 4d 42 43 20 69 73 20 61 20 73 75 70 65 72 20 63 6f 6f 6c 20 73 63 68 6f 6f 6c 21
  - **Binary:** 01010101 01001101 01000010 01000011 00100000 01101001 01110011 00100000 01100001 00100000 01110011 01110101 01110000 01100101 01110010 00100000 01100011 01101111 01101111 01101100 00100000 01110011 01100011 01101000 01101111 01101111 01101100 00100001

Encoding and Data Representation

- Data can also be condensed for accessibility purposes.
  - **Base64 Encoding:** Converts data into ASCII. Based off of a 64 character table (A-Za-z0-9+/)
  - UMBC is a super cool school: VU1CQyBpcyBhIHN1cGVyIGNvb2wgc2Nob29sIQ==
  - *** The equal signs are important giveaways! **
- It can also be completely changed!
  - **Morse code:** ..- -- -... -.-. / .. ... / .- / ... ..- .--. . .-. / -.-. --- --- .-.. / ... -.-. .... --- --- .-..
- **The important takeaway here:** most forms of encoding have some kind of giveaway in them that can let other people know how to get the original message
  - Remember that this provides *Availability!*

Real Encryption

- True encryption involves one and/or two things:
  - Something that cannot be replicated
  - Something that only the parties involved have
- Simplest (but not necessarily usable) example: One-time pad
  - Performing random operations and (ultimately) a shift on each individual letter in a message
  - Simplest form of this is called a **one-time pad cipher**.
  - This is sort of like a polyalphabetic cipher!

UMBC is a super cool school!

-- say I want to give this to Zack and only do it once
-- I create a psuedorandom ‘pad’ that is then destroyed and never used again.
-- There are 22 characters to be shifted
-- My pad: 19-12-6-15-18-21-5-10-23-6-22-9-6-15-5-8-19-3-13-21-23-17

MXGQ zm e bquzz hcss ketikb!

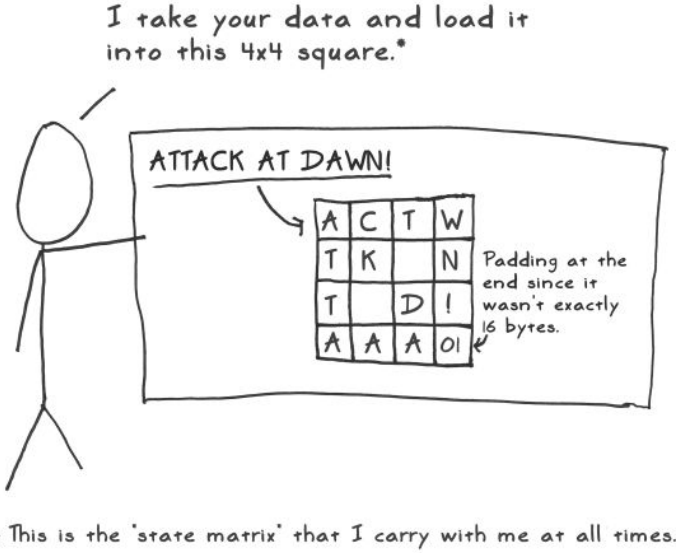
BUT...is it truly random if I’m the one that makes the key?

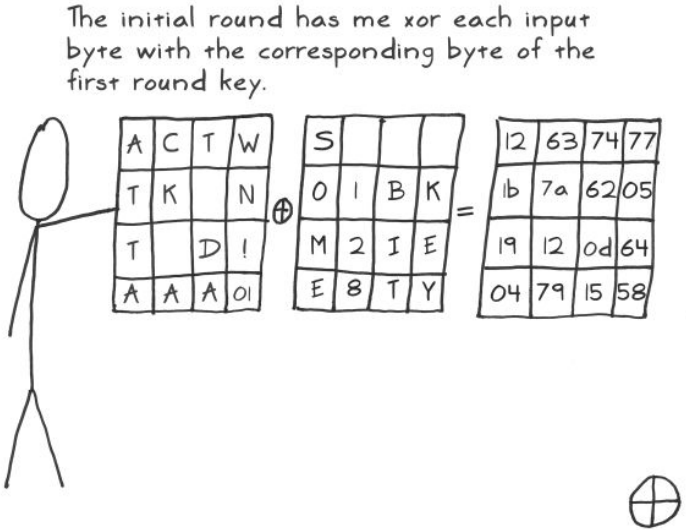
Symmetric Encryption

- Starting to get more in the modern era
- Concept here is that the same key is used to encrypt and decrypt the message
-

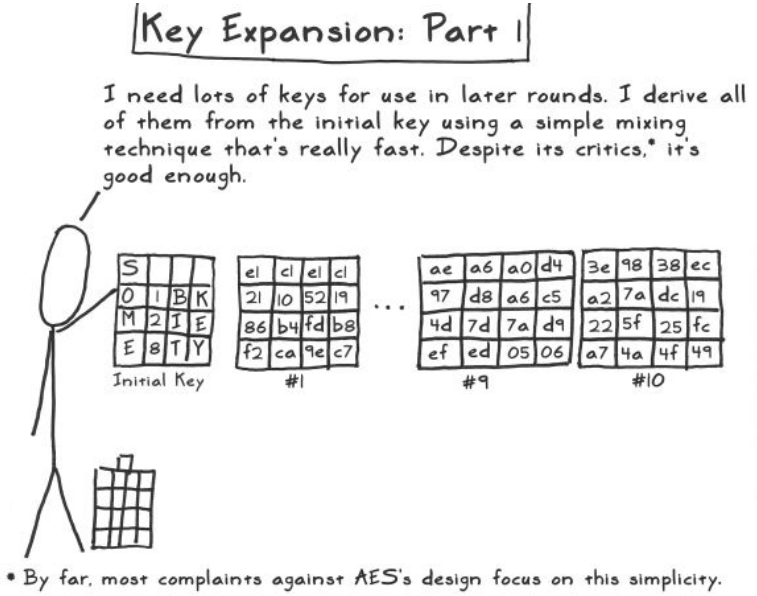
Symmetric Encryption

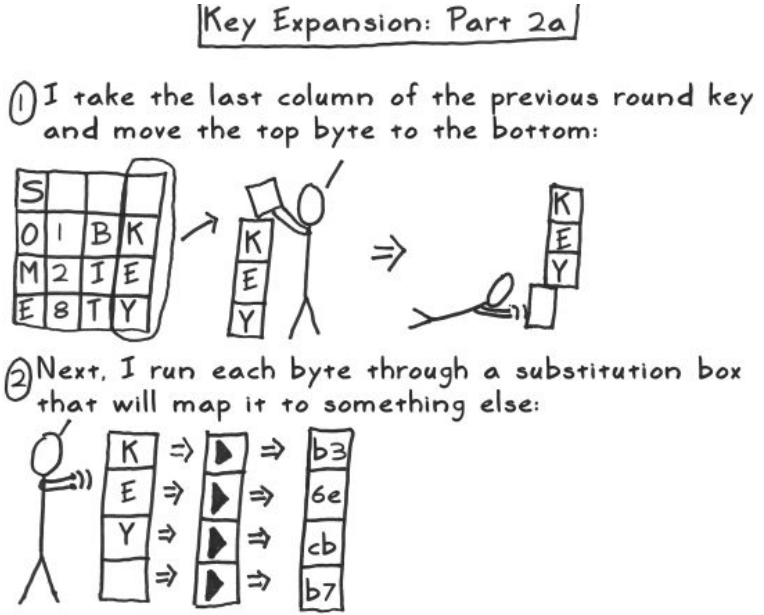
- Classic example: AES (used today)
  - Predecessors to this were DES and Triple DES, but those were defeated
- The way AES works:



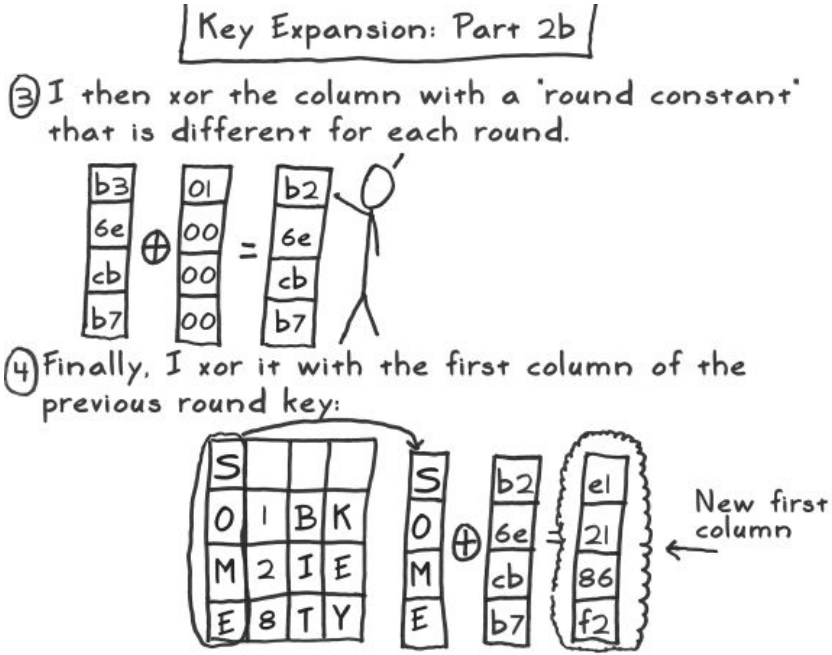


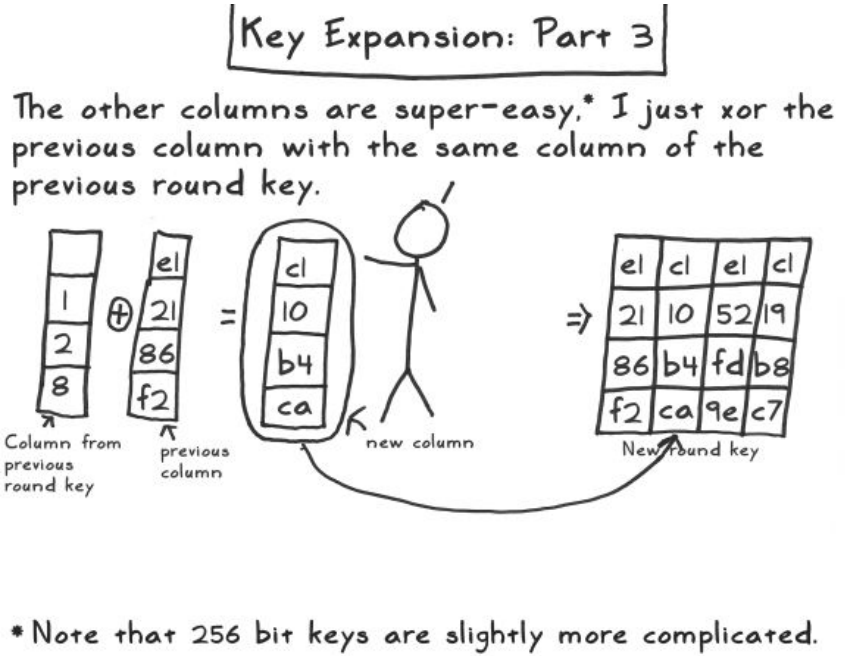
AES in a few pictures





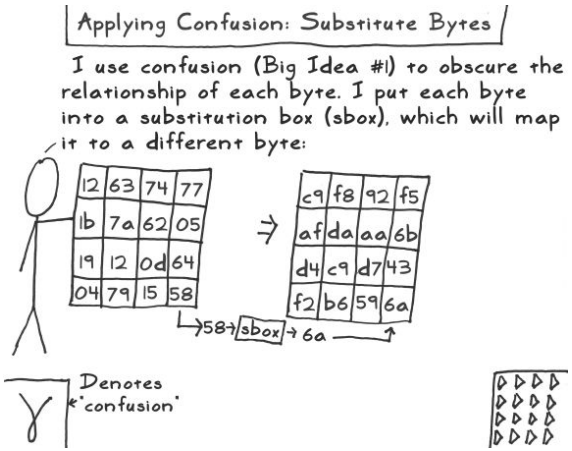
AES in a few pictures

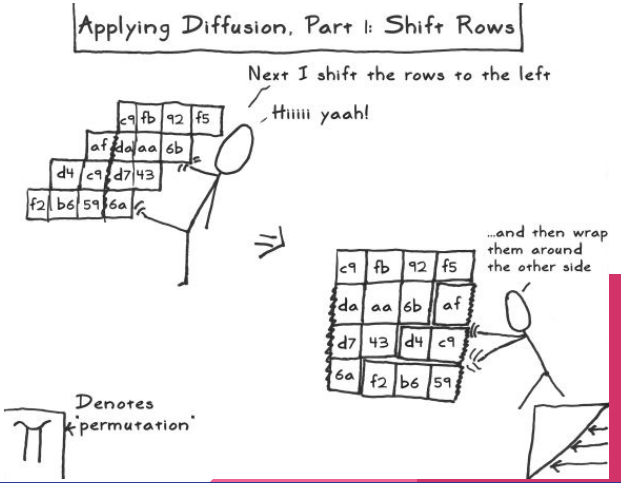




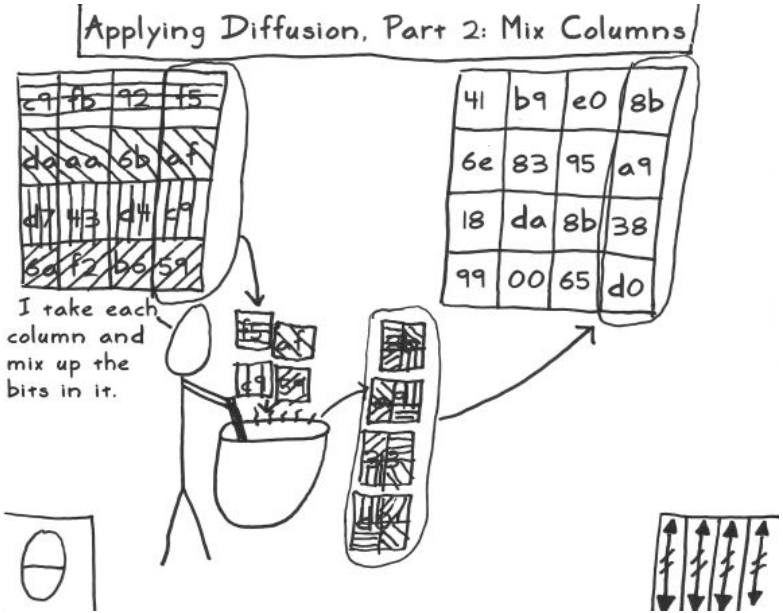
The second half of AES

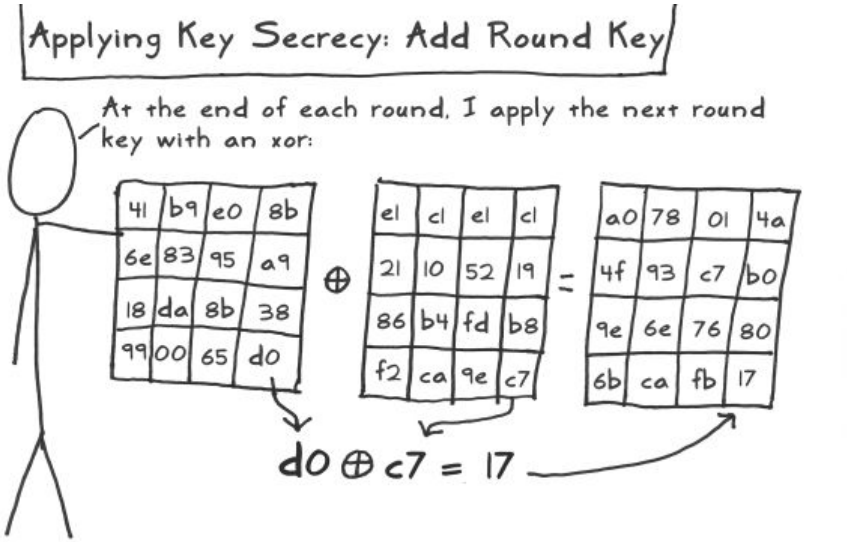
- The key expansion is then performed multiple times to each series of blocks based on the size of the key length, and
- Furthermore, the bytes are obfuscated and the rows are shifted to introduce both **confusion, diffusion, and key secrecy** into the algorithm (or chaos, disbursement, and ownership security as I like to think of it)





The second half of AES





Benefits of Symmetric Encryption

- Symmetric key ciphers are valuable because:
  - It is relatively inexpensive to produce a strong key for these ciphers (it's fast!)
  - The keys tend to be much smaller for the level of protection they afford (not a lot of space)
  - The algorithms are relatively inexpensive to process (it's even faster!)
- But all this is meaningless if the exchange can't retain the privacy of the key.
  - This usually means that the secret key must be encrypted in a different key, and the recipient must already have the key that will be needed to decrypt the encrypted secret-key.
  - Can lead to a never-ending dependency on another key.
- So while it's useful when you want quick security, you'll need to use something else for more sensitive communications/data.