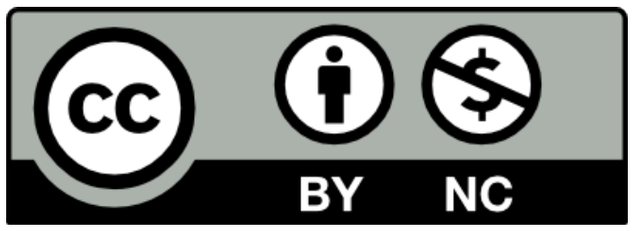
A *Very* Brief Intro to Cryptography

Steven M. Bellovin
https://www.cs.columbia.edu/~smb



Warning

- Cryptography is a subtle, mathematical discipline
- Everything I’m going to talk about is oversimplified—treating all of this in even moderate depth would take a full college-level course
- Getting crypto right is very hard; even experts make mistakes—which is why open standards, developed by people who know what they’re doing and available for critique by others, are so important

Algorithms and Keys

- To encrypt, you need an *algorithm;* every communication needs its own *key* (today, a very large number)

- But everyone can use the same (often open) algorithm

- The security of the communication rests entirely in the secrecy of the keys (Kerckhoffs, 1883)

The Caesar Cipher

A	B	C	…	X	Y	Z
↓	↓	↓		↓	↓	↓
D	E	F	…	A	B	C

- Supposedly used by Julius Caesar
- Replace each letter with one three later in the alphabet
- In this case, the key is *3*

Encrypting and Decrypting

Encryption

- Generate a message
- Select the proper key for this recipient
- Apply the encryption algorithm, using this key

Decryption

- Receive the message
- Select the proper key
- Apply the decryption algorithm, using this key
- Read the message

The sender and recipient have to agree on the key to use!
They have to share it in advance! They have to protect it!

Public Key Cryptography

- Conventional ("symmetric") cryptography: the two parties share the same key
- Public key ("asymmetric") cryptography: each party has a *public* encryption key and a *private* decryption key
- The public key can be published; anyone can use it to send the owner a secret message
- Analogy: one key locks a door; a different one unlocks it
- The private key is kept confidential; knowledge of it is necessary to read that message
- *No more need to share keys ahead of time!*
  - Web encryption (HTTPS) is based on public key cryptography

Who Owns the Key?

- Suppose that Alice wants to send a message to Bob.
  - How does she find Bob's public key?
  - How does she know it's *really* Bob's key, and not some enemy's?

- A trusted party, commonly called a *certificate authority* (CA), vouches for who owns which key
  - (This is done using more complex cryptography)
  - HTTPS relies on a set of well-known CAs for web encryption
  - Browsers or operating systems have a list of CAs they—and therefore users—trust

Using Cryptography

- Using cryptography properly is *hard*
- Doing anything requires a *cryptographic protocol*—a stylized set of messages to properly set up the communication
- Correct protocol design is extremely difficult—the first published protocol was five messages long, but had a flaw that went unnoticed for *17 years*
- If you get the protocol wrong, keys can leak—but cryptographic security depends entirely on protecting the keys

Back Doors in Encryption

- Many governments want access to cryptographic keys
- Typically, this would be done by changing the cryptographic protocols
- But cryptographic protocols are very hard to get right even without this extra requirement!
- Such protocol changes carry a significant risk of insecurity—even apart from other issues, e.g., which governments should have such access

Questions?

Quantum Cryptography

- Quantum cryptography relies on a fundamental property of physics: for certain physical properties, you cannot measure them without changing some other property, and you can't measure both simultaneously
- This—plus some conventional cryptography!—prevents eavesdropping, because the attacker doesn't know what to measure. Security is guaranteed by the laws of physics!
- But…
  - Conventional public key cryptography is still needed to set up the quantum keys (and for other aspects)
  - It's unclear if quantum-encrypted messages can *ever* pass through routers
  - The security guarantees apply to ideal devices—but real-world devices aren't perfect and often leak enough extra information to compromise secrecy
- I do not expect quantum cryptography to *ever* be useful in the real world

Quantum Computing

- In the quantum world, a system can be in multiple states simultaneously
- This can drastically speed up attacks on cryptosystems
- Symmetric algorithms are at some risk, but they were designed with enough security margin that there isn't a problem
- All deployed public key systems are very vulnerable. The US government is currently conducting an open, public competition for a replacement algorithm; expect it in 2–3 years

Major Encryption Standards

- AES (NIST)—General encryption. Winner of an open competition
- RSA (de facto)—The first public key algorithm; being replaced by elliptic curve
- TLS (IETF)—Used for HTTPS, sending and receiving email, general connection security
- IPsec (IETF)—Virtual Private Networks
- WPA2 (IEEE)—WiFi security
- S/MIME (IETF)—Email encryption
- WEP (IEEE)—WiFi security. No longer used; it was a badly broken scheme that has long since been cracked
- Phone encryption (Apple, Google, vendors)
- Signal (de factor)—Phone calls and text messages; adopted by Facebook for WhatsApp

Major Threats

- Buggy code—hack into an endpoint
- Errors in protocol design
  - Possibly sabotaged protocols, e.g., a random number generator apparently designed by the NSA and unwittingly adopted by NIST…
- Usage errors
- Quantum computers—but not a threat for a fair number of years; worry only if your secrets need to be protected for decades

Not a threat: flaws in the basic, low-level algorithms such as AES and RSA. Even the academic community has very deep knowledge of cryptosystem design.