
CS 5430

Symmetric-key Encryption

Prof. Clarkson
Spring 2017

Review

- **First unit of course:** Introduction to security
- **Second:** Cryptography
- **Third through fifth:** The gold standard

Cryptography



P and Q prime

$$N = PQ$$

$$ED \equiv 1 \pmod{(P-1)(Q-1)}$$

$$C = M^E \pmod{N}$$

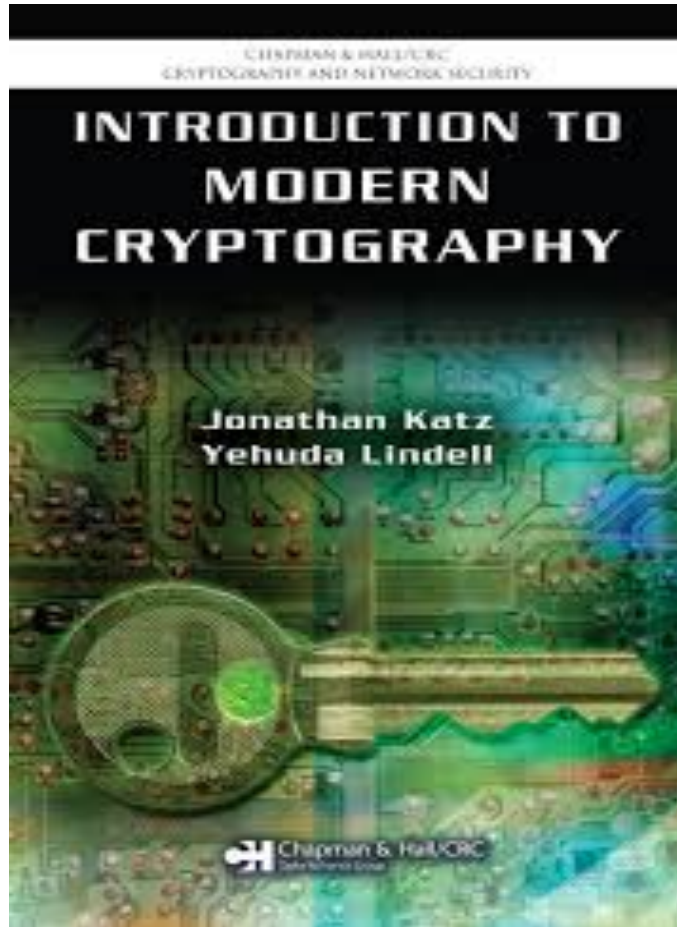
$$M = C^D \pmod{N}$$

Tenants of modern cryptography

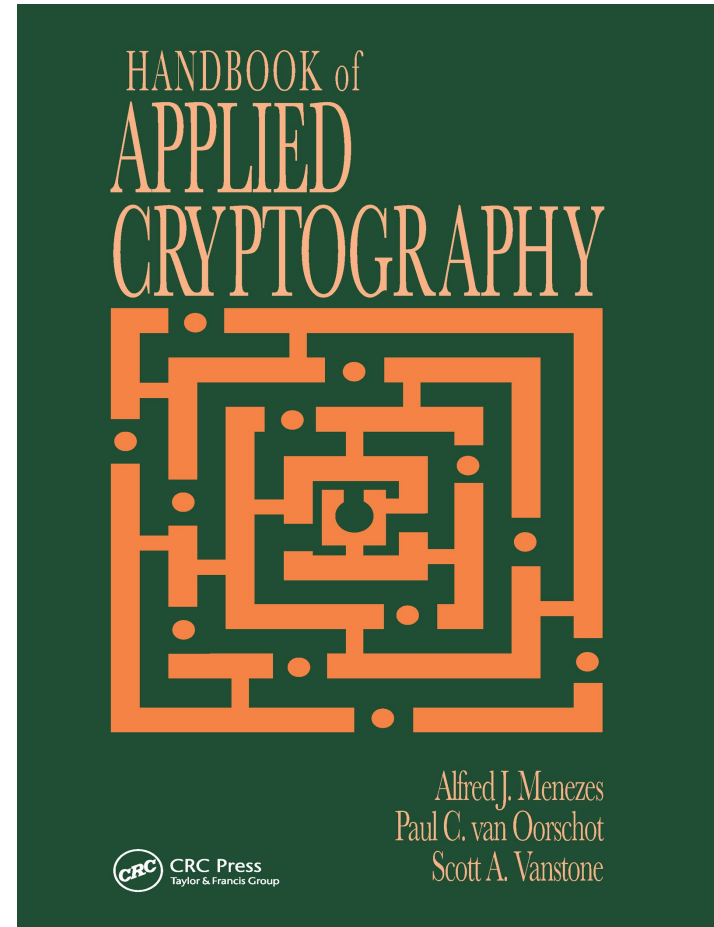
When inventing a cryptographic algorithm/protocol:

- Formulate a precise definition of security
- Provide a rigorous mathematical proof that the cryptographic algorithm/protocol satisfies the definition of security
- State any required assumptions in the proof, keeping them as minimal as possible

Cryptography



cf. CS 4830/6830

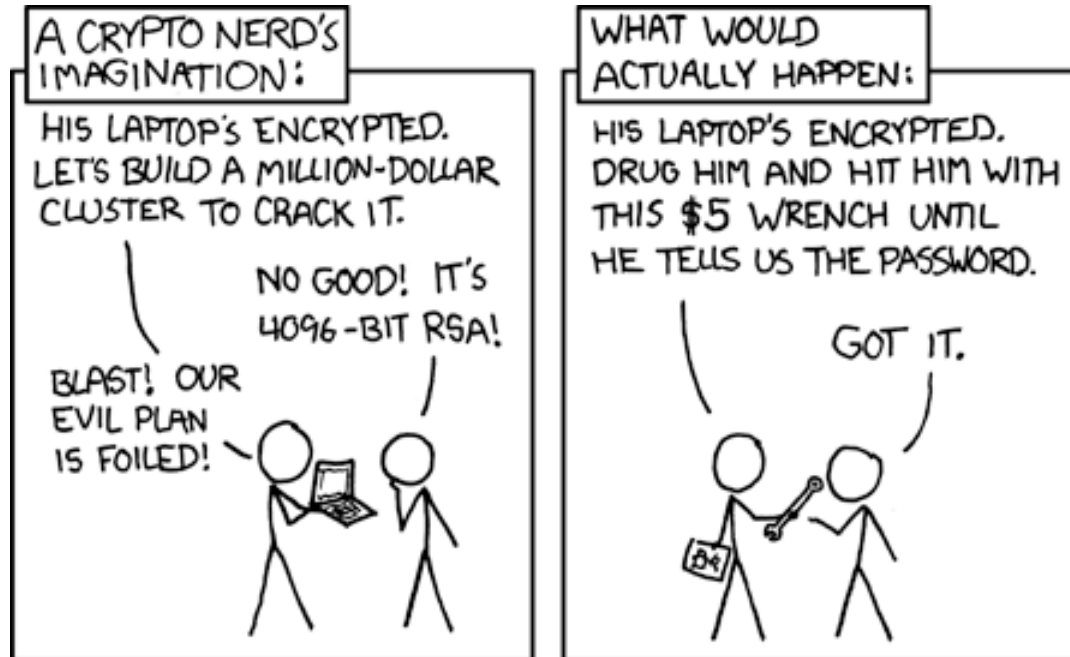


cf. CS 6832

Cryptography

It's a fun tool. But...

- Cryptography is not **the solution**



Cryptography

It's a fun tool. But...

- Cryptography is not **the solution**
- Cryptography is not **easy**
- Cryptography is not **cheap** (who cares?)

ENCRYPTION

Purpose of encryption

- **Threat:** attacker who controls the network
 - can read, modify, delete messages
 - in essence, the attacker *is* the network
 - *Dolev-Yao model* [1983]

Andrew Yao



b. 1946

Turing Award Winner 2000

In recognition of his fundamental contributions to the theory of computation, including the complexity-based theory of pseudorandom number generation, cryptography, and communication complexity.

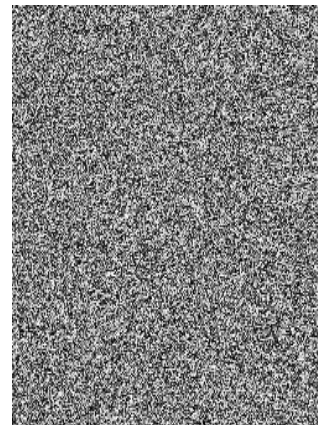
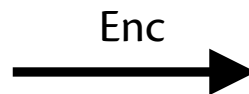
...Since its introduction, [the] “Dolev-Yao model” has been the starting point for most work done on symbolic security...

Purpose of encryption

- **Threat:** attacker who controls the network
 - can read, modify, delete messages
 - in essence, the attacker *is* the network
 - *Dolev-Yao model* [1983]
- **Harm:** messages containing secret information disclosed to attacker (violating confidentiality)
- **Vulnerability:** communication channel between sender and receiver can be read by other principals
- **Countermeasure:** encryption

Encryption algorithms

- $\text{Enc}(m; k)$: encrypt **message** (aka **plaintext** or **cleartext**) m under key k
- $\text{Dec}(c; k)$: decrypt **ciphertext** c with key k
 - note the semicolon



Protocol to exchange encrypted message

For principal A to send message m to principal B:

1. A computes ciphertext c by running $\text{Enc}(m; k)$.
2. A sends c to principal B.
3. B computes $\text{Dec}(c; k)$, recovering m .

We use a quasi-formal notation for protocols...

Protocol to exchange encrypted message

1. A: $c = \text{Enc}(m; k)$
2. A \rightarrow B: c
3. B: $m = \text{Dec}(c; k)$

Protocol narration:

- each step numbered
- each step is a computation by principal or a message between principals
- principals involved are identified as a prefix to each step
- (error handling relatively unspecified)

Cast of characters

- A = Alice
- B = Bob
- E = Eve (the passive eavesdropper)
- M = Mallory (the malicious and active attacker)
- T = Trent (trusted third party)
- ...



(origin of Alice and Bob: 1978 paper that introduced RSA encryption)

Cast of characters



I'VE DISCOVERED A WAY TO GET COMPUTER SCIENTISTS TO LISTEN TO ANY BORING STORY.

Shared key

- How did Alice and Bob come to share key k ?
 - maybe they met way in advance
 - maybe a trusted third party distributed the same key to both of them
 - better answers to come...
- But at some point, it was generated and shared
- Generation: **$k = \text{Gen}(\text{len})$**
 - len is the **length** of the key

Symmetric-key encryption scheme

Algorithms:

- $\text{Enc}(m; k)$: encrypt **message** (aka **plaintext** or **cleartext**) m under key k
- $\text{Dec}(c; k)$: decrypt **ciphertext** c with key k
- $\text{Gen}(\text{len})$: generate a **key** of length len

$(\text{Gen}, \text{Enc}, \text{Dec})$ is a symmetric-key **encryption scheme** aka **cryptosystem**

"Secure" encryption scheme?

Given ciphertext, cannot...

- **Determine key?**
 - Misses the point: we want to protect message secrecy
- **Determine plaintext?**
 - What if you could get 90% of plaintext?
- **Determine any character of plaintext?**
 - What if you could determine it's greater than 1000?
- **Determine any function of the plaintext!**
 - "Right" definition, but must be formulated carefully, and is stronger than some (many) real-world practical encryption schemes

Kerckhoffs' Principle

- Secrecy should depend upon the key remaining secret
- Secrecy should **not** depend upon the algorithm remaining secret
- Instance of Open Design
- Proprietary encryption schemes are to be avoided
 - Just google "proprietary encryption broken"

Perfect encryption

One-time pad:

- $\text{Gen}(\text{len})$ = uniformly random sequence of bits of length len
- $\text{Enc}(m; k) = \text{Dec}(m; k) = m \text{ XOR } k$
 - $\text{length}(m) = \text{length}(k)$

Security:

- Does reveal length of plaintext
- But nothing else!

Practicality:

- Keys must be long (as long as messages)
- Keys can never be reused, would reveal relationships
 - e.g., $(m_1 \text{ XOR } k) \text{ XOR } (m_2 \text{ XOR } k) = m_1 \text{ XOR } m_2$
- Distributing one-time use long keys is hard

REAL-WORLD ENCRYPTION

Block ciphers

- Encryption schemes that operate on fixed-size messages
- The fixed-size is a *block*
- Well-known examples:
 - DES
 - 3DES
 - AES

DES

- **DES (Data Encryption Standard)**
 - Block size: 64 bits
 - Key size: 56 bits
 - Designed by IBM in 1973-4, tweaked by the NSA, then became the US standard for encryption. International adoption followed.
- **3DES (Triple DES)**
 - Block size: 64 bits
 - Key size: 112 or 168 bits
 - Introduced in 1998, because 56 bit keys had become feasible to brute force.
 - 3DES is simply three DES encryptions with two different keys, for an effective 112 bit key; or with three different keys, for an effective 168 bit key.

AES

AES (Advanced Encryption Standard)

- Block size: 128 bits
- Key size: 128, 192, or 256 bits
- Public competition held by NIST, ending in 2001
- Now the US standard, approved by the NSA for Top Secret information
- Currently no practical attacks known

Breaking encryption schemes

- Assume that attack of concern is determining the key, given many ciphertext/plaintext pairs
- **Brute-force attack**: recover key by trying every possible key
 - e.g., AES-128, try all 2^{128} keys
- **Break** is an attack that recovers key in less work than brute-force
- Suppose best-known attack requires 2^X operations....then X is the **strength** aka **security level** of the encryption scheme
 - Best case is that strength = key length
 - As attacks are discovered, strength degrades
 - e.g., 3DES-168 has known attack that requires 2^{112} operations, reducing strength from 168 to 112

Key lengths

- Various recommendations for strength summarized at <https://www.keylength.com/en/>
- Based on:
 - known attacks
 - hardware capabilities
 - predicted advances
- Why not use highest strength possible?
Performance.

Upcoming events

- [Wed] A1 due, A2 out

If you think cryptography is the answer to your problem, then you don't know what your problem is. – Peter G. Neumann