Stroll through Cryptography

Josef Pieprzyk

Data61, CSIRO, Sydney, Australia

Road Map

Private-Key Cryptography

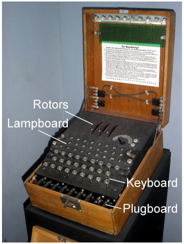
Public-Key Cryptography

Multiparty Computations

Early Cryptography

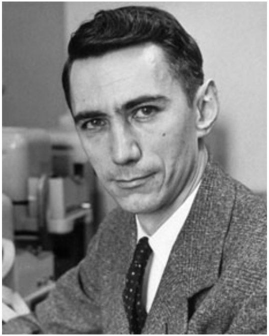
- Ancient ciphers - Caesar cipher

- Military cryptography (Enigma, Purple) - an example of early encryption machines



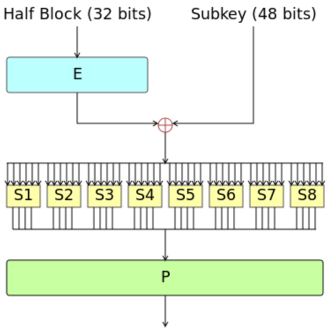
Theory of Secrecy Systems – Modern Cryptography

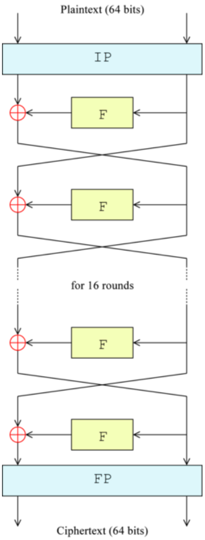
- Developed by Claude Shannon in the late 1940s in famous paper 'A Communication Theory of Secrecy Systems'
- Concept of Ideal Cipher (OTP)
- Design of secure cipher from insecure components (SP network)



Modern Private-Key Cryptography – DES

- Data Encryption Standard (1975) – NIST Standard (IBM)
- Feistel structure, 4×6 eight S-boxes
- 56-bit keys

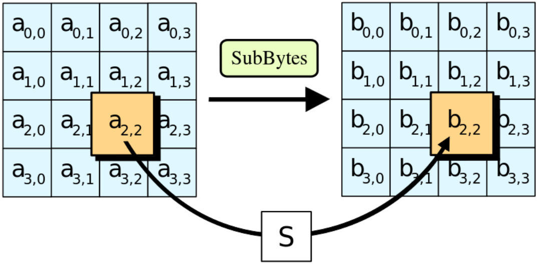




Modern Private-Key Cryptography – AES

Advanced Encryption Standard

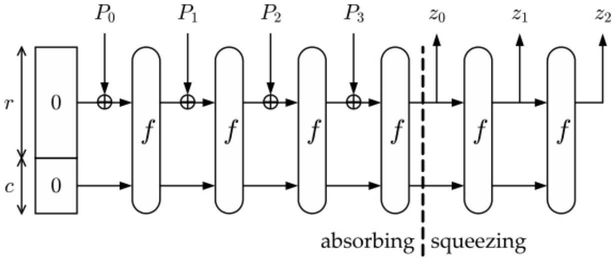
- Public AES competition announced by NIST in 1997
- Finalists: Rijndael, Serpent, Twofish, RC6, MARS
- Winner - Rijndael (Vincent Rijmen and Joan Daemen) - 2001
- SP network structure, 8×8 S-box



Hashing – SHA3

Secure Hash Algorithm Standard

- Cryptanalysis by Xiaoyung Wang
- SHA3 Competition - NIST 2007
- Finalists: Blake, Grøstl, JH, Keccak and Skein
- Winner - Keccak, 2012 (Guido Bertoni, Joan Daemen, Michael Peeters and Gilles Van Assche)
- Sponge structure



Authenticated Encryption – CAESAR (2014 – 2017)

Authenticated Encryption Competition
(Daniel Bernstein)

- Lightweight applications – Ascon and ACORN
- High-performance applications – AEGIS and OCB
- Defense in depth – Deoxys-II and COLM

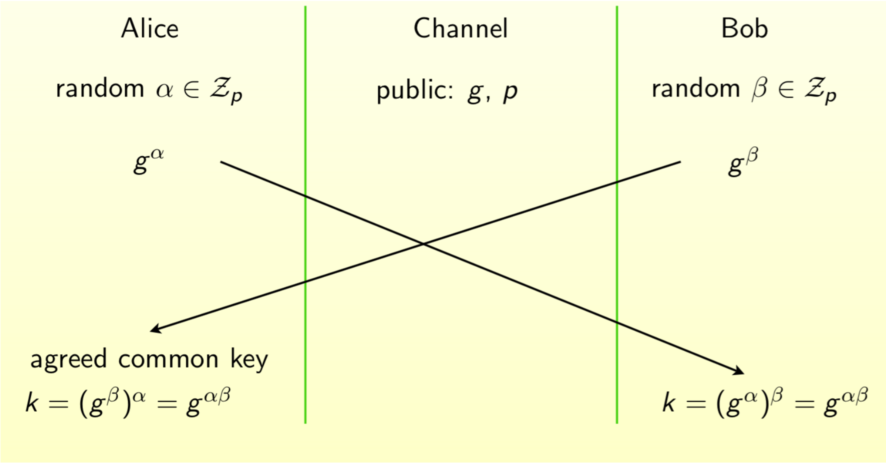
Road Map

Private-Key Cryptography

Public-Key Cryptography

Multiparty Computations

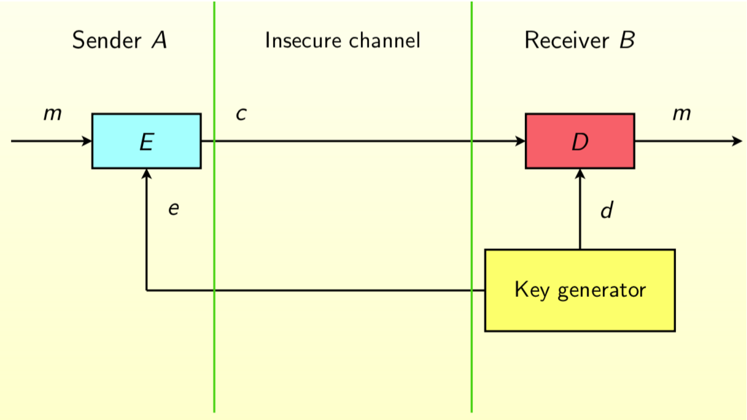
Diffie-Hellman Key Agreement (1976)



El Gamal Cryptosystem (1984)

Alice	Channel	Bob
random $s \in \mathcal{Z}_p$	public; g, g^k, p	secret k
$c_2 = g^s$	$\longrightarrow$	$(g^s)^k = g^{sk}$
$c_1 = m(g^k)^s$	$\longrightarrow$	$m = c_1.g^{-sk}$

Rivest-Shamir-Adleman Cryptosystem (1978)



public key $e \in \mathcal{Z}_N$ random
$(\gcd(e, \varphi(N)) = 1)$
$d = e^{-1} \mod (p-1)(q-1)$

Encryption $c = m^e \pmod{N}$
Decryption $m = c^d \pmod{N}$

Pairing-based Cryptography

- Pairing invented by Menezes Okamoto and Vanstone (1993) – an attack on elliptic curve logarithms
- Definition:
  Given two abelian groups G_1, G_2 and a cyclic group G_3 of order n, then a pairing is a map

  $$e : G_1 \times G_2 \to G_3$$

  with the following properties:
  - ⋆ bilinearity

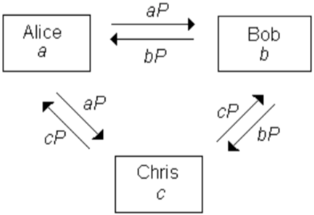
  $$\begin{aligned} e(P+P', Q) &= e(P, Q) \cdot e(P', Q) \\ e(P, Q+Q') &= e(P, Q) \cdot e(P, Q') \end{aligned}$$

  - ⋆ non-degeneracy

  $$\begin{aligned} \forall_{P \neq 0; P \in G_1} \exists_{Q \in G_2} \quad & e(P, Q) \neq 1 \\ \forall_{Q \neq 0; Q \in G_2} \exists_{P \in G_1} \quad & e(P, Q) \neq 1 \end{aligned}$$

Three-Party Diffie-Hellman (Joux 2000)

- Alice $\rightarrow$ { Bob, Chris }: $a \cdot P$
- Bob $\rightarrow$ { Alice, Chris }: $b \cdot P$
- Chris $\rightarrow$ { Alice, Bob }: $c \cdot P$



- Alice computes $K = e(b \cdot P, c \cdot P)^a = e(P, P)^{abc}$
- Bob computes $K = e(a \cdot P, c \cdot P)^b = e(P, P)^{abc}$
- Chris computes $K = e(a \cdot P, b \cdot P)^C = e(P, P)^{abc}$

Identity-Based Encryption (Boneh/Franklin 2001)

Problems with PKC:

- Public-key encryption requires the senders to use AUTHENTIC public keys of receivers
- Need for TA that distributes certificates of public keys (PKI)

Solution – IBE

- TA – single public key + generates receivers' decryption keys
- Sender (Alice) uses public key of Bob $K_B = H(ID_B)$
- Receiver (Bob) gets decryption key D_B from TA

Certificateless Public-Key Cryptography

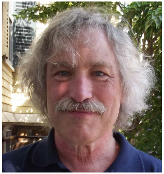
Characteristics of IBE

- Senders do not need certificates
- TA generates decryption keys
- Key escrow problem
- Revocation could be a problem

Better – Certificateless (Public-key) Encryption (CE) - Al-Riyami/Paterson 2003

- Senders do not need certificates
- No key escrow problem

NTRU Public-Key Encryption



- NTRU – Nth degree TRUncated polynomial ring
- Invented in 1995 by Hoffstein, Pipher, and Silverman
- The design went through few iterations
- Variant `pNTRUEncrypt` is IND-CPA secure assuming hardness of worse-case problems in ideal lattices (2011, Stehlé and Steinfeld)
- Variant `NTRUCCA` is IND-CCA2 secure assuming hardness of worse-case problems in ideal lattices (2012, Steinfeld et al)

Post-Quantum Cryptography Competition – NIST 2016-2020

Type	PKE/KEM	Signature
Lattice-based	CRYSTALS-KYBER NTRU SABER	CRYSTALS-DILITHIUM FALCON
Code-based	McEliece	
Multivariate		Rainbow

Homomorphic Encryption

- How to secure data in the cloud?
- How to protect privacy if you outsource your computations?
- Homomorphic Encryption - (1978, Rivest, Adleman and Dertouzos)
- How could it work?

  additive homomorphism

$$E(m_1 + m_2) = E(m_1) + E(m_2)$$

  multiplicative homomorphisms

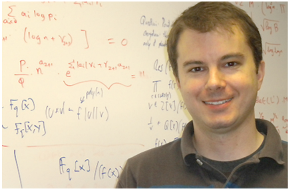
$$E(m_1 \cdot m_2) = E(m_1) \cdot E(m_2)$$

- Early homomorphic encryptions:

  Goldwasser-Micali Encryption (1982)

  Paillier cryptosystem (1999)

Fully Homomorphic Encryption



- Fully Homomorphic Encryption (FHE) - Craig Gentry (2009)
- Allows to evaluate a circuit (with addition and multiplication operations) such that

$$f(E(x_1), \ldots, E(x_n)) = E(f(x_1, \ldots, x_n))$$

- Encryption uses lattices and is very slow
- There FHE with much better efficiency

Road Map

Multiparty Computations

Assume that

- there is a collection of participants

  $\{P_1, P_2, \ldots, P_n\}$ and a function $Y = F(x_1, \ldots, x_n)$

- each participant

  P_i holds a private input x_i for $i = 1, \ldots, n$

- MPC protocol allows participants to evaluate the function F in such a way that at the end of the protocol

  all participants learn Y and
  their inputs remain private

Classical Solutions

- Yao, 1982 – the concept of secure MPC – Millionaire Problem
- Goldreich, Micali and Wigderson, 1987 – solution with computational security
- Ben-Or, Goldwasser, and Wigderson and independently Chaum, Crepeau, and Damgård, 1988 – solutions with unconditional security

MPC Applications

- Money without Trusted Authority (Bit Coin)
- Collaborative Data Mining
- Lacation-based Services
- Secure Cloud Services
- Electronic Elections

Thank You