Hochschule Konstanz
Fakultät Informatik

Introduction to IT Security

WIN+AIN

Hanno Langweg

06 Applications of Cryptography

Goals of cryptography

- Protection of data in transfer over insecure channel
- Protection of data in storage on untrusted media
- Confidentiality (prevent attacks)
- Integrity (detect attacks)
- Authenticity, origin of data (detect attacks)

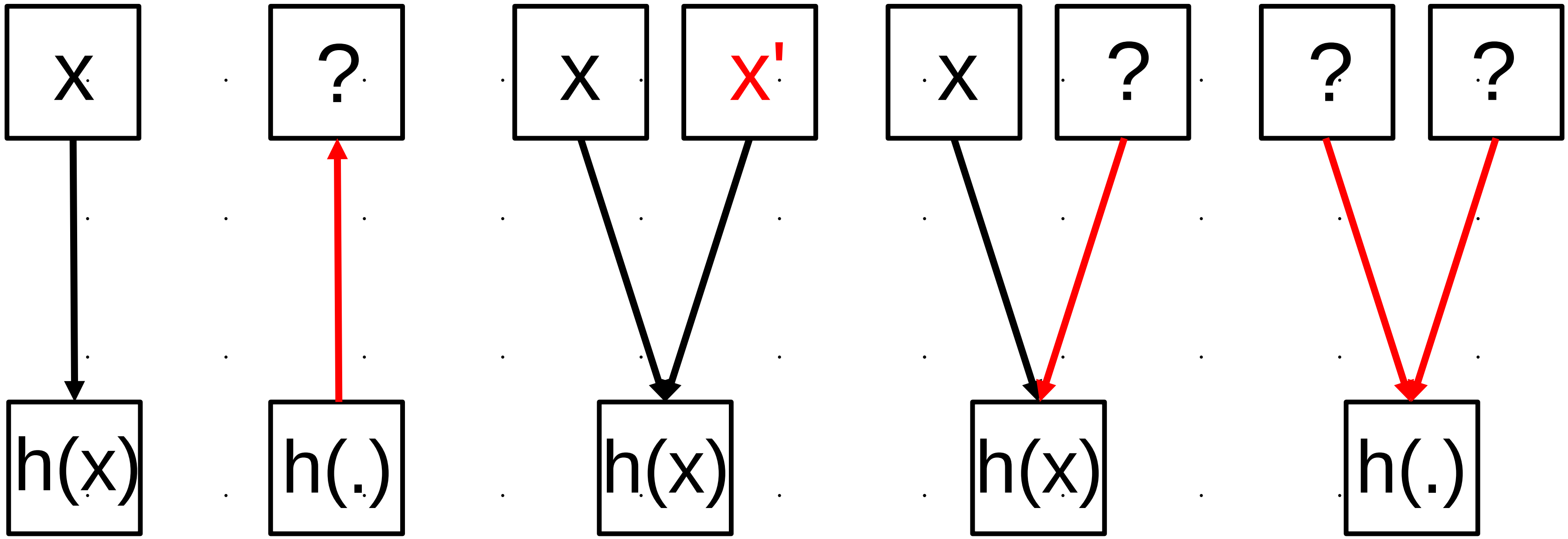
Hash functions

- Take input of **arbitrary length** and **map** it to output with **fixed length**, e.g. 512 bits

- Applications
  - File comparison
  - Integrity of messages, message authentication codes
  - Protection of passwords
  - Reduction of input to cryptographic algorithm
  - Digital signatures
  - Bitcoin
  - …

Hash functions

– Requirements for **one-way hash function** *h*
(easy to compute image, hard to compute source)
1. **Ease of computation**: given x, it is easy to compute h(x)
2. **Compression**: h maps inputs x of arbitrary bitlength to outputs h(x) of a fixed bitlength n
3. **One-way**: given a value y, it is computationally infeasible to find an input x so that h(x) = y
4. **Collision resistance**: it is computationally infeasible to find x and x' where x /= x' with h(x) = h(x')



Ease of computation

Pre-image resistance

Collision

Weak collision resistance (2nd pre-image resistance)

Strong collision resistance

Hash functions

- Frequently used hash functions
  - MD5: 128 bit digest
    Has been broken; no longer recommended for cryptography
    (But still good for e.g. fast file comparisons)
  - SHA-1 ("**S**ecure **H**ash **A**lgorithm"): 160 bit digest
    Attacks exist; replacement recommended
  - SHA-2 (SHA-256/384/512), RIPEMD-160:
    Still considered secure
  - SHA-3 as potential replacement for SHA-2 in case SHA-2
    turns out to be broken

Asymmetric cryptography

Symmetric vs. asymmetric cryptosystems

- Symmetric cryptosystem
  - Both parties use **same (secret) key**
    Trusted channel needed to distribute key
  - **Fast**
- Asymmetric cryptosystem
  - Parties have a **public key** (for encryption) and a **private key** (for decryption)
    Public key can be announced in a public directory
  - **Slow**
- Can be combined ("hybrid")
  - Generate a **secret session key** for a symmetric cryptosystem
  - Use **asymmetric encryption** to **transmit session key**
  - Encrypt further **messages** with received **session key**

Key distribution

- Symmetric cryptosystem
  - n parties ➔ $\frac{n\ x\ (n-1)}{2}$ keys (1 key per pair of parties)
  - Distribute secret keys in advance over trustworthy channel
- Asymmetric cryptosystem
  - n parties ➔ $2\ x\ n$ keys (1 public, 1 private key per party)
  - Distribute only public keys in advance
  - Integrity+authenticity of public keys essential
- Hybrid cryptosystem
  - n parties ➔ $2\ x\ n$ keys (1 public, 1 private key per party)
  - Secret session key **generated when needed** (needs good random number generator)

Diffie and Hellman

- Proved public key cryptography was possible (but did not give a constructive example)
- Key exchange protocol
- Won Turing award in 2015: http://www.acm.org/awards/2015-turing

CRYPTOGRAPHY PIONEERS RECEIVE ACM A.M. TURING AWARD

Diffie and Hellman's Invention of Public-Key Cryptography and Digital Signatures Revolutionized Computer Security

ACM, the Association for Computing Machinery, today named Whitfield Diffie, former Chief Security Officer of Sun Microsystems and Martin E. Hellman, Professor Emeritus of Electrical Engineering at Stanford University, recipients of the 2015 ACM A.M. Turing Award for critical contributions to modern cryptography. The ability for two parties to use encryption to communicate privately over an otherwise insecure channel is fundamental for billions of people around the world. On a daily basis, individuals establish secure online connections with banks, e-commerce sites, email servers and the cloud. Diffie and Hellman's groundbreaking 1976 paper, "New Directions in Cryptography," introduced the ideas of public-key cryptography and digital signatures, which are the foundation for most regularly-used security protocols on the Internet today. The Diffie-Hellman Protocol protects daily Internet communications and trillions of dollars in financial transactions.

The ACM Turing Award, often referred to as the "Nobel Prize of Computing," carries a $1 million prize with financial support provided by Google, Inc. It is named for Alan M. Turing, the British mathematician who articulated the mathematical foundation and limits of computing and who was a key contributor to the Allied cryptoanalysis of the German Enigma cipher during World War II.

RSA

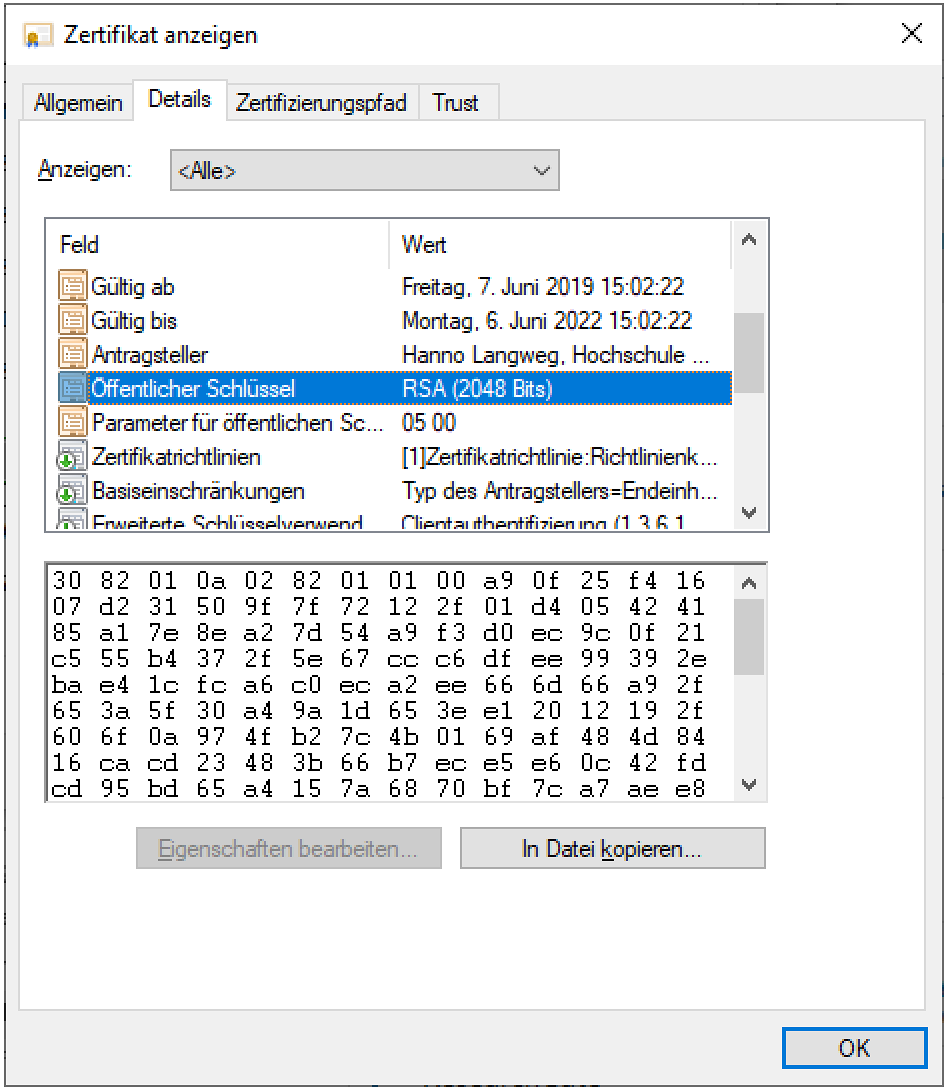
- Rivest, Shamir, Adleman (1977)
- Based on one-way function that is hard to invert
  - Factorization of (large) numbers into primes
  - Hard to determine prime factors
  - Easy to verify if factors known

- Key generation
- Key distribution
- Encryption
- Decryption

RSA key generation

- Bob chooses two large prime numbers p and q (large, i.e. p * q > 3,000 bits; BSI 2019)
- Bob computes $n = p * q$
- Bob finds a value e with
  - $1 < e < n$ (recommended: $e > 2^{16}$, e.g. 65537 = 0x10001)
  - $GCD(e, (p-1)*(q-1)) = 1$ $\varphi(p*q) = (p-1)*(q-1)$
- Bob finds a value d with
  - $(e * d) \bmod ((p-1)*(q-1)) = 1$
- Bob now has two keys
  - d is called the **private key**
  - (n, e) is called the **public key**

My RSA public key



RSA key generation - example

- Bob chooses two prime numbers p = 19 and q = 31
- Bob computes n = p * q = 19 * 31 = 589
- Bob finds a value e = 49 with
  - 1 < e < n = 589 (Note: 589 << 2^{16} in this example)
  - GCD(e, (p-1)*(q-1)) = GCD(49, 18*30) = 1
- Bob now finds a value d with
  - (e * d) mod ((p-1)*(q-1)) = 1
  - By help of **Extended Euclidean Algorithm**
  - d = 1069

RSA key generation - example

- $p = 19$, $q = 31$, $n = 589$, $e = 49$, $d = 1069$
- Bob now has two keys
  - $d = 1069$ is called the private key
  - $(n, e) = (589, 49)$ is called the public key

- Remember: in reality, numbers would be larger, i.e. $n > 2^{3000}$, $e > 2^{16}$, $d > 2^{500}$

RSA key distribution

- Bob posts the public key (589, 49) to a public directory
  - Phone book (remember, RSA was invented in 1970s)
  - Newspaper (printed, distributed, large quantities)
  - Homepage
  - Key server, LDAP directory etc.
  - …

RSA encryption

- Alice wants to send a message m_{plain} = ”A” to Bob
  - ”A” = 65 (ASCII, UTF-8)
  - Encrypted message $m_{enc} = (m_{plain})^e \bmod n =$
    $65^{49} \bmod 589 = 198$
  - Remember: $a^{(x+y)} = a^x * a^y$
    $49_{10} = 110001_2$
    $65^{49} = \mathbf{65^{32}} * \mathbf{65^{16}} * \mathbf{65^{1}} = 102 * 524 * 65 \bmod 589 = 198$

  - $65^2 \bmod 589 = 4225 \bmod 589 = 102$
  - $65^4 = 65^2 * 65^2 = 102 * 102 \bmod 589 = 391$
  - $65^8 = 65^4 * 65^4 = 391 * 391 \bmod 589 = 330$
  - $\mathbf{65^{16}} = 65^8 * 65^8 = 330 * 330 \bmod 589 = \mathbf{524}$
  - $\mathbf{65^{32}} = 65^{16} * 65^{16} = 524 * 524 \bmod 589 = \mathbf{102}$

RSA encryption

– Alice sends $m_{enc} = 198$ to Bob.

RSA decryption

- Bob receives $m_{enc} = 198$
- Bob computes $m_{dec} = (m_{enc})^d \bmod n$
  - $m_{dec} = (m_{enc})^d = 198^{1069} \bmod 589 = 65$

  - $1069_{10} = 10000101101_2$
  - $198^2 = 330$, $\mathbf{198^4} = 330^2 = 524$
  - $\mathbf{198^8} = 524^2 = 102$, $198^{16} = 102^2 = 391$
  - $\mathbf{198^{32}} = 391^2 = 330$, $198^{64} = 330^2 = 524$
  - $198^{128} = 524^2 = 102$, $198^{256} = 102^2 = 391$
  - $198^{512} = 391^2 = 330$, $\mathbf{198^{1024}} = 330^2 = 524$
  - $198^{1069} = \mathbf{198^{1024}} * \mathbf{198^{32}} * \mathbf{198^8} * \mathbf{198^4} * \mathbf{198^1} = 65$

Security of RSA

- There is **no known fast way** to factor large numbers into prime factors
- Algorithms have been developed to **solve factorization** for numbers **up to several hundred bits**
  - 2009: 768 bit RSA modulus (Kleinjung et al.)
    ~ 2,000 years of 2.2 GHz AMD Opteron
    https://eprint.iacr.org/2010/006.pdf
  - 2014: 1,199 bit (limited to 2^{1199}-1, not arbitrary number)
- Key lengths of **3,000 bits** are considered **strong enough for the foreseeable future**
- Quantum computers may be able to solve facorization faster in the future (if ever)

Quantum computers

- Quantum computers may be able to solve factorization faster in the future (if ever)
- But (as of 2019):
  - Shor's algorithm for factorization needs 5 registers with 2,000 qbits each to break RSA with 2,048 bit keys
  - For every "dirty" qbit you need 100 times as many qbits to get a "clean" bit
  - Hence, a quantum computer would need **1 million qbits** to break RSA with 2,048 bit keys

INTEL'S 49-QUBIT PROCESSOR

During his keynote at CES 2018 in January, Intel CEO Brian Krzanich unveiled our 49-qubit superconducting quantum test chip, code-named "**Tangle Lake.**" The 3-inch by 3-inch chip and its package is now in the hands of Intel's quantum research partner QuTech in the Netherlands for testing at low temperatures. Quantum computing is heralded for its potential to tackle problems that today's conventional computers can't handle. Scientists and industries are looking to quantum computing to speed advancements in areas like chemistry or drug development, financial modeling, and even climate forecasting.

https://newsroom.intel.com/news/future-quantum-computing-counted-qubits/#gs.ieeqt8

Elliptic curve cryptography (ECC)

- Asymmetric cryptosystem
- Based on difficulty of solving the discrete logarithm problem on elliptic curves
- Same level of security as RSA
- Smaller key lengths (100s instead of 1,000s bits)
- Less computationally expensive, good for e.g. smart cards

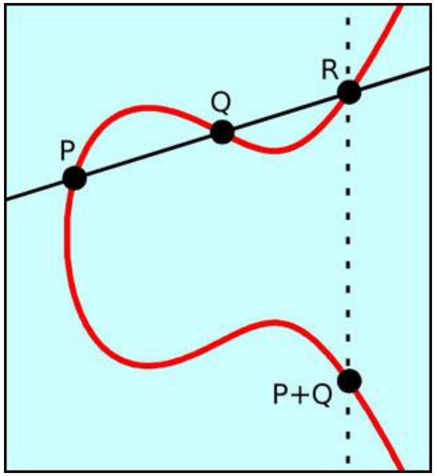


Image by Andrew Sutherland

Electronic signatures

Electronic signatures

- Proof of authenticity and integrity of data
- Legal recognition on same level as handwritten signature

- Usually two steps:

1. Computation of message digest with hash function
   (useful for compression, not necessary for signature)
2. **Signing** of message digest using **asymmetric** cryptosystem
   (would also work with symmetric cryptosystem; asymmetric cryptosystem also supports non-repudiation)

- *Signatory* keeps *private key* secret (often using e.g. a smart card)
- *Recipient* can verify signature with *public* key

Applications of electronic signatures

- Email integrity and authenticity (e.g. S/MIME, PGP)
- Submission of tax returns
- Electronic invoices
- Communication between lawyers and courts
- Emission trading
- Certificates of origin (cross-border transports)
- E-BAföG
- …

Summary

- One-way **hash functions:** Ease of computation, compression, one-way, collision resistance
- Asymmetric cryptography
  - RSA based on difficulty of factorisation of large numbers into primes
  - Encryption/decryption without key agreement in advance
  - Signing and signature verification with different keys
    ➔ non-repudiation
- Designing a cryptosystem is hard
  - **Use existing algorithms and implementations**