# 12. Cryptography

Thomas Kister and Frank Stephan

April 10, 2014

## Secure Transmission

Bob sends a message to Alice. Nobody else should be able to read the message.

- Step 1: Bob translates the message into a coded version.
- Step 2: Bob sends the coded version to Alice.
- Step 3: Alice translates the coded version into the original message.

## Caesar Cypher

Each letter is replaced by the b$^{th}$ letter following it in alphabetical order. $b$ is the key. The key must be kept secret.

## Cyphering

$$c = Encr(m) = m + b \pmod{26}$$

## Decyphering

$$m = Decr(Encr(m)) = Encr^{-1}(c) = c - b \pmod{26}$$

## Random Numbers for Cryptography

The key, $b$, is chosen randomly.

## Example

$b = 2$: "BUZZ" becomes "DWBB"

## Code Breaking

"Doo Jdxo lv glylghg lqwr wkuhh sduwv, rqh ri zklfk wkh Ehojdh lqkdelw, wkh Dtxlwdql dqrwkhu, wkrvh zkr lq wkhlu rzq odqjxdjh duh fdoohg Fhowv, lq rxuv Jdxov, wkh wklug. Doo wkhvh gliihu iurp hdfk rwkhu lq odqjxdjh, fxvwrpv dqg odzv. "

What is the key?

## The Most Frequent Letters in English

E: 13%, T: 9%, A: 8%, O: 8%, I: 7%, N: 7%, S: 7%, H: 6%, R: 6%

## The Most Frequent Letters in the Text

H: 22%, D: 19%, W: 16%, L: 15%, K: 15%

### Code Breaking

"All Gaul is divided into three parts, one of which the Belgae inhabit, the Aquitani another, those who in their own language are called Celts, in ours Gauls, the third. All these differ from each other in language, customs and laws."

## XOR Cypher

Alice and Bob exchange a code-string. The message is encoded and decoded by flipping bits where the code string has 1s.

## Random Numbers for Cryptography

The code-string is chosen randomly.

## Cyphering

$$c = Encr(m) = m \oplus k$$

## Decyphering

$$m = Decr(Encr(m)) = c \oplus k$$

## Example

| | |
|---|---|
| `Message:` | `11010010001000010000010000001` |
| `Code-String:` | `10101100110010110101101100111` |
| `Coded Message:` | `01111110111010100101111100110` |
| | |
| `Coded Message:` | `01111110111010100101111100110` |
| `Code-String:` | `10101100110010110101101100111` |
| `Message:` | `11010010001000010000010000001` |

## Affine Cypher

Alice and Bob exchange two numbers $a$ and $b$. The message is encoded and decoded by applying an affine function to it.

## Cyphering

$$Encr(m) = a \times m + b \pmod{26}$$

## Decyphering

$$Decr(c) = \overline{a} \times (c - b) \pmod{26}$$

$\overline{a}$ is the modular multiplicative inverse of $a$ ($\overline{a} \times a = 1 \pmod{26}$).

## Random Numbers for Cryptography

$a$ and $b$ are chosen randomly, but $a$ must be coprime with the length of the alphabet (to ensure the existence of a modular multiplicative inverse).

## Example

"All Gaul is divided into three parts, one of which the Belgae inhabit, the Aquitani another, those who in their own language are called Celts, in ours Gauls, the third. All these differ from each other in language, customs and laws."

## $a = 5$, $b = 3$, $\bar{a} = 21$

"Dgg Hdzg rp srersxs rquv umkxx adkup, vqx vc jmrnm umx Ixghdx rqmdiru, umx Dfzrudqr dqvumxk, umvpx jmv rq umxrk vjq gdqhzdhx dkx ndggxs Nxgup, rq vzkp Hdzgp, umx umrks. Dgg umxpx srccxk ckvl xdnm vumxk rq gdqhzdhx, nzpuvlp dqs gdjp."

## Public Key Cryptography

In 1976, Whitfield Diffie and Martin Hellman proposed a new key exchange protocol.

## Public Key Cryptography

Encoding and decoding is done with different keys. Everyone can encode a message with the public key of the receiver. But only the private key of the receiver can decode a message. The encoding key can be made publicly available.

## Digital Signatures

A clear text message can be signed by the sender. A summary (hash value) of the message is encoded with the sender's private key. Everyone can decode the summary with the sender's public key, and verify it against the summary of the received message.

## Pretty Good Privacy

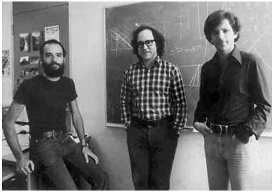
Some people have a public key for the PGP cryptography program on their web page.

## PGP

E582 94F2 E9A2 2748 6E8B 061B 31CC 528F D7FA 3F19

## RSA Cryptosystem

RSA was invented by Ellis, Cocks and Williamson for the British Government Communications Headquarters and by Rivest, Shamir and Adleman (RSA) in public open research.



## Main Idea

RSA is based on easy primality testing and hard factoring: it is easy to generate a public key and hard to compute the private key from the public key.

## Cyphering

Choose at random two distinct large (circa the square of a googol) prime numbers $p$ and $q$. Compute $n = p \times q$. Compute $f = (p-1) \times (q-1)$. Choose at random a number $e$ less than $f$ and coprime to $f$. Compute the modular multiplicative inverse $d$ of $e$ for $f$.

$$c = Encr(m) = m^e \pmod{n}$$

## Decyphering

$$m = Decr(c) = c^d \pmod{n}$$

## Random Numbers for Cryptography

$p$ and $q$ and $e$ are chosen randomly but $p$ and $q$ are large primes and $e$ is less than and coprime to $p \times q$.

## RSA

- Choose prime numbers $p$ and $q$: $p = 11$ and $q = 13$
- Compute giving $n = p \times q$: $n = 143$
- Compute $f = (p - 1) \times (q - 1)$: $f = 120$
- Choose $e$ less than $f$, and coprime to $f$: $e = 7$
- Compute the modular multiplicative inverse $d$ of $e$ for $f$: $d = 103$
- The public key is $(n, e) = (143, 7)$
- The encryption function is $Encr(m) = c = m^e \pmod{n}$.
- The private key is $(n, d) = (143, 103)$
- The decryption function is $Decr(m) = m = c^d \pmod{n}$.

## RSA

- Public Key: $(n, e) = (143, 7)$.
- Message (between 0 and $n - 1$):
  $m = 1010_{base2} = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 8$
- Encryption: $Encr(m) = c = m^e \pmod{n} = 8^7 mod(143) = 57$.
- Private Key: is $(n, d) = (143, 103)$.
- Decryption: $Decr(m) = m = c^d \pmod{n} = 57^{103} \pmod{143} = 8$.

## Example

| Message | Encr(m) | Decr(m) |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 2 | 128 | 63 |
| 3 | 42 | 16 |
| 28 | 63 | 128 |
| 63 | 2 | 28 |
| 128 | 28 | 2 |
| 142 | 142 | 142 |

### How to Compute $e$ Efficiently

The modular inverse can be computed in $O(\log(n)^2)$.

### How to Encrypt and Decrypt Efficiently?

$m^e \pmod{n}$ can be done in $O(\log(e))$.

### How to Find $(n, d)$ if we Know $(n, e)$?

Intractable (hopefully!).

## Factorization of a 768-bit RSA modulus

"On December 12, 2009, we factored the 768-bit, 232-digit number RSA-768 by the number field sieve. [...] This result is a record for factoring general integers. Factoring a 1024-bit RSA modulus would be about a thousand times harder, and a 768-bit RSA modulus is several thousands times harder to factor than a 512-bit one. Because the first factorization of a 512-bit RSA modulus was reported only a decade ago it is not unreasonable to expect that 1024-bit RSA moduli can be factored well within-the next decade [...]. Thus, it would be prudent to phase out usage of 1024-bit RSA within the next three to four years."
Kleinjung et al.

## Communication Protocol

- Bob sends message $m$ to Alice.
- Bob codes $m$ with his decryption algorithm and then with Alice's encryption algorithm:

$$c = Encr_A(Decr_B(m)).$$

- Bob transmits $c$ to Alice.
- Alice decodes $c$ with her decryption algorithm and then encrypts it with Bob's public encryption algorithm in order to make it readable:

$$m = Encr_B(Decr_A(c)).$$

Why does Bob apply $Decr_B$ to his message?

## Generating Random Numbers

- Ancient Egyptians, Indians and Chinese gambled with dice 5000 years ago.
- In 1927 L. Tippett, a British statistician, published a table of 41,600 random numbers.

## Generating Random Numbers

- Hardware random number generators use physical phenomena such as thermal electronic noise (built-in Intel Pentium) and radioactive decay.
- Software Random numbers generators generate pseudo-random numbers: sequences of numbers as a function of seed.
- George Marsaglia, an American mathematician, made 4.8 billion random bits available.
- Kolmogorov, a Russian mathematician, defined random data as data that cannot be generated by a program shorter than the data itself.

## The Middle Square Method

- The method was invented by J. Von Neumann, a German mathematician, in 1948.
- Take a number of $k$ digits, the seed, square it and take the middle $k$ digits.

```
function middlesquare4digits(seed, n) {
  for (i = 0; i < n; i++) {
    seed = seed * seed;
    seed = Math.floor(seed / 100);
    seed = seed % 10000;
    document.write("Random number between 0 and 9999: "
        + seed + "<br>");
  }
}
```

```
seed = 1234
Random number between 0 and 9999: 5227
Random number between 0 and 9999: 3215
Random number between 0 and 9999: 3362
Random number between 0 and 9999: 3030
Random number between 0 and 9999: 1808
Random number between 0 and 9999: 2688
Random number between 0 and 9999: 2253
Random number between 0 and 9999: 760
Random number between 0 and 9999: 5776
```

```
seed = 2041
Random number between 0 and 9999: 1656
Random number between 0 and 9999: 7423
Random number between 0 and 9999: 1009
Random number between 0 and 9999: 180
Random number between 0 and 9999: 324
Random number between 0 and 9999: 1049
Random number between 0 and 9999: 1004
Random number between 0 and 9999: 80
Random number between 0 and 9999: 64
Random number between 0 and 9999: 40
Random number between 0 and 9999: 16
Random number between 0 and 9999: 2
Random number between 0 and 9999: 0
```

## Attribution