Cryptographic algorithms

Prof. Bart Preneel

COSIC

Bart.Preneel(at)esat.DOTkuleuven.be

http://homes.esat.kuleuven.be/~preneel

February 2014

© Bart Preneel. All rights reserved

Outline

• 1. Cryptology: concepts and algorithms
  – symmetric algorithms for confidentiality
  – symmetric algorithms for data authentication
  – public-key cryptography
• 2. Cryptology: protocols
  – identification/entity authentication
  – key establishment
• 3. Public-Key Infrastructure fundamentals

Definitions

data entities

Confidentiality

encryption

Anonymity

Integrity

Identification

Availability

Authorisation

Non-repudiation of origin, receipt

Contract signing

Notarisation and Timestamping

4

Cryptology: basic principles

Symmetric cryptology: confidentiality

• old cipher systems:
  – transposition, substitution, rotor machines
• the opponent and her power
• the Vernam scheme
• DES and triple-DES
• AES
• RC4
Cryptographic Algorithms
Bart Preneel
February 2014

Old cipher systems (pre 1900)

- Caesar cipher: shift letters over k positions in the alphabet (k is the secret key)
  THIS IS THE CAESAR CIPHER
  WKLV LV WKH FDHVDU FLSKHU
- Julius Caesar never changed his key (k=3).

Cryptanalysis example:

<table>
<thead>
<tr>
<th>TIPGK</th>
<th>RERP JEEJ WLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>UJQN</td>
<td>BFSQ NASA XP</td>
</tr>
<tr>
<td>WNULP</td>
<td>WKHU BOEEO BJQ</td>
</tr>
<tr>
<td>ZOV</td>
<td>NXXIV PPFFP CRK</td>
</tr>
<tr>
<td>APWNR</td>
<td>YLIJW QQQOS DLS</td>
</tr>
<tr>
<td>BXQOS</td>
<td>EMOXX RRHRR ETM</td>
</tr>
<tr>
<td>CYPT ANALYSIS FUN</td>
<td></td>
</tr>
<tr>
<td>DSZQU</td>
<td>BOBMZ TJJT GVO</td>
</tr>
<tr>
<td>ETAVV</td>
<td>CPCNA UKUKU HWP</td>
</tr>
<tr>
<td>FUBSW</td>
<td>DQDOB VLVV IXQ</td>
</tr>
</tbody>
</table>

Plaintext? k = 17

Old cipher systems (pre 1900) (2)

- Substitutions
  - ABCDEFGHIJKLMNOPQRSTUVWXYZ
  - MZNJSRAXFQVDUCPP\textsuperscript{11}CRT
- Transpositions
  \textit{TRANS} ORI S
  \textit{POSIT} NOTIT
  \textit{IONS} OSANP

Security

- there are \( n! \) different substitutions on an alphabet with \( n \) letters
- there are \( n! \) different transpositions of \( n \) letters
- \( n=26: n! \approx 40329146112605635584000000 \approx 4 \cdot 10^{26} \) keys
- trying all possibilities at 1 nanosecond per key requires...
  \( 4 \cdot 10^{26} / (10^9 \cdot 10^5 \cdot 4 \cdot 10^3) = 10^{10} \) years

Assumptions on Eve (the opponent)

- A scheme is broken if Eve can deduce the key or obtain additional plaintext
- Eve can always \textit{try all keys} till “meaningful” plaintext appears: a brute force attack
  – solution: large key space
- Eve will try to find \textit{shortcut attacks} (faster than brute force)
  – history shows that designers are too optimistic about the security of their cryptosystems

Letter distributions

\begin{tikzpicture}
\begin{axis}[
    width=\textwidth,
    height=\textwidth,
    ybar,
    bar width=0.3cm,
    ylabel={Counts},
    xlabel={Letters},
    xtick=data,
    xticklabels={A,B,C,D,E,F,G,H,...,Z},
    ytick={0,2,4,6,8,10,12},
    yticklabels={0,2,4,6,8,10,12},
    legend pos=north west,
]
\addplot coordinates {
    (A,12) (B,8) (C,6) (D,8) (E,10) (F,12) (G,8) (H,6) (I,4) (J,2) (K,4) (L,6) (M,8) (N,10) (O,12) (P,8) (Q,6) (R,8) (S,10) (T,12) (U,8) (V,6) (W,4) (X,2) (Y,4) (Z,6)
};
\end{axis}
\end{tikzpicture}
Assumptions on Eve (the opponent)

- Cryptology = cryptography + cryptanalysis
- Eve knows the algorithm, except for the key (Kerckhoffs’s principle)
- increasing capability of Eve:
  - knows some information about the plaintext (e.g., in English)
  - knows part of the plaintext
  - can choose (part of) the plaintext and look at the ciphertext
  - can choose (part of) the ciphertext and look at the plaintext

New assumptions on Eve

- Eve may have access to side channels
  - timing attacks
  - simple power analysis
  - differential power analysis
  - acoustic attacks
  - electromagnetic interference
- Eve may launch (semi-)invasive attacks
  - differential fault analysis
  - probing of memory or bus

Side channel analysis

Cryptology + side channels

The Rotor machines (WW II)
Cryptographic Algorithms
Bart Preneel

Life cycle of a cryptographic algorithm

- Idea
- Mathematical analysis
- Publication
- Public evaluation
- HW/SW implementation
- Standardization
- Industrial products
- Take out of service

Vernam scheme

- 0 + 1 = 1
- 1 + 0 = 1
- 0 + 0 = 0
- 1 + 1 = 0

- Identical mathematical symbols can result in different electrical signals

Three approaches in cryptography

- **Information theoretic** security
  - Ciphertext only
  - Part of ciphertext only
  - Noisy version of ciphertext

- **System-based** or practical security
  - Also known as “prayer theoretic” security

- **Complexity theoretic** security:
  - Model of computation, definition, proof
  - Variant: Quantum cryptography

Synchronous Stream Cipher (SSC)

Exhaustive key search

- 2014: 2^{40} instructions is easy, 2^{60} is somewhat hard, 2^{80} is hard, 2^{128} is completely infeasible
  - 1 million machines with 16 cores and a clock speed of 4 GHz can do 2^{26} instructions per second or 2^{60} per year
  - Trying 1 key requires typically a few 100 instructions
- Moore’s “law”: Speed of computers doubles every 18 months: Key lengths need to grow in time
  - But adding 1 key bit doubles the work for the attacker

- Key length recommendations in 2014
  - < 70 bits: insecure
  - 80 bits: one year (but not for NSA)
  - 100 bits: 20 years
Exhaustive key search: multiple targets

- If one wants to recover 1 key out of $2^t$ keys, the cost to recover a key is $2^{k-t} < 2^k$.
- If one wants to recover all of $2^t$ keys with $t > k/3$ the cost per key can be reduced to $2^{k/3}$.
  - $2^t$ precomputation to fill a memory of size $2^{2k/3}$.
  - On-line cost per key: $2^{k/3}$ encryptions.
  - Known as time/memory tradeoff or “rainbow tables”.
- So depending on the circumstances, a 128-bit key can become an 85-bit key.

SSC: Specific properties

- Recipient needs to be synchronized with sender.
- No error-propagation:
  - Excellent for wireless communications.
- Key stream independent of data:
  - Key stream can be precomputed.
  - Particular model for cryptanalysis: attacker is not able to influence the state.

SSC: Avoid repeating key stream

- For a fixed key $K$ and initial value $IV$, the stream cipher output is a deterministic function of the state.
- A repetition of the state (for a given $K$, $IV$) leads to a repetition of the key stream and plaintext recovery (think of the problem of Vernam encryption with reused key):
  - Hence state needs to be large and next state function needs to guarantee a long period.
  - $IV$ can be used to generate a different key stream for every packet in a packet-oriented communication setting.
  - Old stream ciphers defined without $IV$ are problematic in such a setting.

Practical stream ciphers

- A5/1 (GSM) (64 or 54).
- E0 (Bluetooth) (128).
- RC4 (browser) (40-128).
- SNOW-3G (3GSM) (128).
- HC-128 (128).
- Trivium (80).

A5/1 stream cipher (GSM)

- Exhaustive key search: $2^{64}$ (or rather $2^{54}$).
  - Hardware 10K$ < 1$ minute ciphertext only.
- Search 2 smallest registers: $2^{45}$ steps.
- [BWS00] 1 minute on a PC.
  - 2 seconds of known plaintext.
  - $2^{48}$ precomputation, 146 GB storage.
- [BB05]: 10 minutes on a PC.
  - 3-4 minutes of ciphertext only.
- [Nohl-Paget’09]: Rainbow tables.
  - Seconds with a few frames of ciphertext only.
Bluetooth stream cipher

brute force: $2^{128}$ steps

[Lu+05] 24 known bits of 224 frames, $2^{38}$ computations, $2^{33}$ memory


• designed by Ron Rivest (MIT)
• leaked in 1994
• $S[0..255]$: secret table derived from user key $K$

\[
\begin{align*}
\text{for } &i=0 \text{ to } 255 \quad S[i] := i \\
\text{for } &j:=0 \\
\text{for } &i=0 \text{ to } 255 \\
\text{ swap } &S[i] \text{ and } S[j] \\
i :=0, & j:=0
\end{align*}
\]


Generate key stream which is added to plaintext

\[
\begin{align*}
&i:=i+1 \\
&j:=(j + S[i]) \mod 256 \\
&\text{swap } S[i] \text{ and } S[j] \\
&t:=(S[i] + S[j]) \mod 256 \\
&\text{output } S[t]
\end{align*}
\]

RC4: weaknesses

• was often used with 40-bit key
  – US export restrictions until Q4/2000
• best known general shortcut attack: $2^{241}$
  [Maximov-Khovratovich’09]
• weak keys and key setup (shuffle theory)
• large statistical deviations
  – bias of output bytes (sometimes very large)
  – can recover 220 out of 256 bytes of plaintexts after
    sending the same message 1 billion times (WPA/TLS)
• problem with resynchronization modes (WEP)

Block cipher

• large table: list $n$-bit ciphertext for each $n$-bit plaintext
  – if $n$ is large: very secure (codebook)
  – but for an $n$-bit block: $2^n$ values
  – impractical if $n \geq 32$
• alternative $n = 64$ or 128
  – simplify the implementation
  – repeat many simple operations

Block cipher (2)

• larger data units: 64…128 bits
• memoryless
• repeat simple operation (round) many times
Data Encryption Standard (1977)

- encrypts 64 plaintext bits under control of a 56-bit key
- 16 iterations of a relatively simple mapping
- FIPS: US government standard for sensitive but unclassified data
- worldwide de facto standard since early 80ies
- surrounded by controversy

Data Encryption Standard (DES)

The DES round function

DES S-box 1

Security of DES (56 bit key)

- PC: trying 1 DES key: 7.5 ns
- Trying all keys on 128 PCs: 1 month: $2^{27} \times 2^{16} \times 25 \times 2^{7} = 2^{55}$
- M. Wiener’s design (1993): 1,000,000 $ machine: 3 hours (in 2012: 3 seconds)

EFF Deep Crack (July 1998)
250,000 $ machine: 50 hours...

Federal Register, July 24, 2004

DEPARTMENT OF COMMERCE
National Institute of Standards and Technology
[Docket No. 040602169–4169–01]

Announcing Proposed Withdrawal of Federal Information Processing Standard (FIPS) 46–3, was evaluated pursuant to its scheduled review. At the conclusion of this review, NIST determined that the strength of the DES algorithm is no longer sufficient to adequately protect Federal government information. As a result, NIST proposes to withdraw FIPS 46–3, and the associated FIPS 74 and FIPS 81. Future use of DES by Federal agencies is to be permitted only as a component function of the Triple Data Encryption Algorithm (TDEA).
3-DES: NIST Spec. Pub. 800-67
(May 2004)
• two-key triple DES: until 2009
• three-key triple DES: until 2030

AES (Advanced Encryption Standard)
• open competition launched by US government (Sept. ‘97) to replace DES
• 22 contenders including IBM, RSA, Deutsche Telekom
• 128-bit block cipher with key of 128/192/256 bits
• as strong as triple-DES, but more efficient
• royalty-free

A machine that cracks a DES key in 1 second would take 149 trillion years to crack a 128-bit key

AES (2001)
• FIPS 197 published on December 2001 after 4-year open competition
  – other standards: ISO, IETF, IEEE 802.11,…
• fast adoption in the market
  – except for financial sector
  – NIST validation list: > 2700 implementations
    • http://csrc.nist.gov/groups/STM/cavp/documents/aes/aesval.html
• 2003: AES-128 also for secret information and AES-192/-256 for top secret information!

AES (2001)
• security:
  – algebraic attacks of [Courtois+02] not effective
  – side channel attacks: cache attacks on unprotected implementations
• speed:
  – software: 7.6 cycles/byte [Käser-Schwabe’09]
  – hardware: Intel provides AES instruction (Westmere/Sandy Bridge, 2010/2011) at 0.75 cycles/byte for decryption – AMD one year behind

[Shamir ’07] AES may well be the last block cipher
Encryption limitations

- Ciphertext becomes random string: “normal” crypto does not encrypt a credit card number into a (valid) credit card number
- Typically does not hide the length of the plaintext (unless randomized padding)
- Does **not** hide existence of plaintext (requires steganography)
- Does **not** hide that Alice is talking to Bob (requires traffic confidentiality, e.g. TOR)

Data authentication: the problem

- encryption provides confidentiality:
  - prevents Eve from learning information on the cleartext/plaintext
  - but does not protect against modifications (active eavesdropping)
- Bob wants to know:
  - the **source** of the information (data origin)
  - that the information has not been **modified**
  - (optionally) **timeliness** and **sequence**
- data authentication is typically more complex than data confidentiality

Symmetric cryptology: data authentication

- the problem
- hash functions without a key
  - MDC: Manipulation Detection Codes
- hash functions with a secret key
  - MAC: Message Authentication Codes

Data authentication: MAC algorithms

- Replace protection of authenticity of (long) message by protection of secrecy of (short) key
- Add MAC to the plaintext

MAC algorithms

- typical MAC lengths: 32..96 bits
  - Forgery attacks: $2^m$ steps with $m$ the MAC length in bits
- typical key lengths: (56)..112..160 bits
  - Exhaustive key search: $2^k$ steps with $k$ the key length in bits
- birthday attacks: security level smaller than expected
MAC algorithms

- Banking: CBC-MAC based on triple-DES
- Internet: HMAC and CBC-MAC based on AES
- Information theoretic secure MAC algorithms (authentication codes): GMAC/UMAC
  - Highly efficient
  - Rather long keys (some)
  - Part of the key refreshed per message

CBC-MAC based on AES

P1

 AES

 P2

 AES

 P3

 AES

C1

AES

C2

AES

C3

security level: \(2^{64}\)

Select leftmost 64 bits

Data authentication: MDC

- MDC (manipulation detection code)
- Protect short hash value rather than long text

This is an input to a cryptographic hash function. The input is a very long string, that is reduced by the hash function to a string of fixed length. There are additional security conditions: it should be very hard to find an input hashing to a given value (a preimage) or to find two colliding inputs (a collision).

Data authentication: MDC

- n-bit result
  - preimage resistance: for given \(y\), hard to find input \(x\) such that \(h(x) = y\) (2\(^n\) operations)
  - \(2^{nd}\) preimage resistance: hard to find \(x' \neq x\) such that \(h(x') = h(x)\) (2\(^n\) operations)
  - Collision resistance: hard to find \((x,x')\) with \(x' \neq x\) such that \(h(x') = h(x)\) (2\(^n\) operations)

Important hash algorithms

- MD5
  - \(2^{nd}\) preimage 2\(^{128}\) steps (improved to 2\(^{123}\) steps)
  - Collisions 2\(^{64}\) steps
  - Shortcut: Aug. '04: 2\(^{39}\) steps; '09: 2\(^{29}\) steps

- SHA-1:
  - \(2^{nd}\) preimage 2\(^{160}\) steps
  - Collisions 2\(^{80}\) steps
  - 1.6 MS for 1 year in 2014
  - Shortcut: Aug. '05: 2\(^{69}\) steps

- SHA-2 family (2002)

- SHA-3 family (2013) – Keccak (Belgian design)
  - \(2^{nd}\) preimage 2\(^{256}\) .. 2\(^{512}\) steps
  - Collisions 2\(^{128}\) .. 2\(^{384}\) steps
NIST’s Modes of Operation for AES

- ECB/CBC/CFB/OFB + CTR (Dec 01)
- MAC algorithm: CMAC (May 05)
- Authenticated encryption:
  - CCM: CTR + CBC-MAC
  - GCM: Galois Counter Mode

Issues:
- associated data
- parallelizable
- on-line
- provable security

Concrete recommendations

- AES-128 in CCM mode
  - CCM = CTR mode + CBC-MAC
  - change key after 2^{40} blocks
- Stream ciphers (better performance)
  - hardware: SNOW-3G or Trivium
  - software: HC-128
- CAESAR: open competition from 2013-2017 will come up with better solutions
  - http://competitions.cr.yp.to/caesar.html

Public-key cryptology

- the problem
- public-key encryption
- digital signatures
- an example: RSA
- advantages of public-key cryptology

Limitation of symmetric cryptology

- Reduce security of information to security of keys
- But: how to establish these secret keys?
  - cumbersome and expensive
  - or risky: all keys in 1 place
- Do we really need to establish secret keys?

Public key cryptography: encryption

Public key cryptography: digital signature
A public-key distribution protocol: Diffie-Hellman

• Before: Alice and Bob have never met and share no secrets; they know a public system parameter $\alpha$

  
  \[
  \begin{align*}
  \text{generate } x & \Rightarrow \alpha^x \\
  \text{compute } \alpha^x & \Rightarrow \alpha^y \\
  \text{compute } k = \alpha^y & = (\alpha^x)^y
  \end{align*}
  \]

• After: Alice and Bob share a short term key $k$
  – Eve cannot compute $k$: in several mathematical structures it is hard to derive $x$ from $\alpha^x$
  (this is known as the discrete logarithm problem)

RSA (‘78)

• choose 2 “large” prime numbers $p$ and $q$
• modulus $n = p \cdot q$
• compute $\lambda(n) = \text{lcm}(p-1,q-1)$
• choose $e$ relatively prime w.r.t. $\lambda(n)$
• compute $d = e^{-1} \mod \lambda(n)$

• public key = $(e,n)$
• private key = $d$ of $(p,q)$

• encryption: $c = m^e \mod n$
• decryption: $m = c^d \mod n$

The security of RSA is based on the “fact” that it is easy to generate two large primes, but that it is hard to factor their product

Factorisation records

2009: 768 bits or 232 digits

<table>
<thead>
<tr>
<th>Size (digits)</th>
<th>Effort (log)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 digit ~3.3 bits</td>
<td></td>
</tr>
<tr>
<td>768 bits</td>
<td>2009</td>
</tr>
<tr>
<td>512 bits</td>
<td>2009</td>
</tr>
</tbody>
</table>

Advantages of public key cryptology

• Reduce protection of information to protection of authenticity of public keys
• Confidentiality without establishing secret keys
  – extremely useful in an open environment
• Data authentication without shared secret keys: digital signature
  – sender and receiver have different capability
  – third party can resolve dispute between sender and receiver
Disadvantages of public key cryptology

- Calculations in software or hardware **two to three orders of magnitude** slower than symmetric algorithms
- Longer keys: 1024 bits rather than 56...128 bits
- What if factoring is easy?

Crypto software libraries

http://ece.gmu.edu/crypto_resources/web_resources/libraries.htm

**C/C++/C#**
- Botan (C++)
- Cryptlib (C)
- Cryptopp (C++)
- CydSSL (C) embedded
- GnuTLS (C)
- Libgcrypt (C++)
- MatrixSSL (C++) embedded
- MirACL (binaries)
- OpenSSL (C++)
- PolarSSL (C)

**Java**
- SunJCA/JCE
- BouncyCastle (BC, C#)
- CryptixCrypto (until '05)
- EspreSSL
- FlexiProvider
- GNU Crypto
- IAIK
- Java SSL
- RSA JSafe

Reading material

- B. Preneel, Modern cryptology: an introduction.
  - This text corresponds more or less to the second half of these slides
  - It covers in more detail how block ciphers are used in practice, and explains how DES works.
  - It does not cover identification, key management and application to network security.

Selected books on cryptology

- Other authors: Johannes Buchmann, Serge Vaudenay

Books on network security and more

- IACR (International Association for Cryptologic Research): www.iacr.org