T-79. 5501
Cryptology

Lecture 3, Feb 5, 2008
- Unicity distance, example
- Extended Euclidean Algorithm
- Chinese remainder theorem
- Euler’s totient function
- Euler’s theorem
- LFSR
Unicity distance - Example

AES Cryptosystem
\[ \mathcal{P} = \{0,1\}^{128}, \mathcal{C} = \{0,1\}^{128}, \mathcal{K} = \{0,1\}^{128} \]

Data \((L)\) to be encrypted: DES-keys, 64 bits each, each eighth bit is a parity check bit.

\[ H_L = 64 - 8 = 56; \ R_L = 1 - \frac{56}{64} = \frac{1}{8} \]

\[ n_0 \approx \log_2 |\mathcal{K}| / R_L \log_2 |\mathcal{P}| = \frac{128}{(128/8)} = 8 \]
Prime Numbers

Definition: An integer $p > 1$ is a prime if and only if its only positive integer divisors are 1 and $p$.

Fact: Any integer $a > 1$ has a unique representation as a product of its prime divisors

$$a = \prod_{i=1}^{t} p_i^{e_i} = p_1^{e_1} p_2^{e_2} \cdots p_t^{e_t}$$

where $p_1 < p_2 < \ldots < p_t$ and each $e_i$ is a positive integer.

Some first primes: 2,3,5,7,11,13,17,… For more primes, see: www.utm.edu/research/primes/

Example: Composite (non-prime) numbers and their factorisations:

$18 = 2 \cdot 3^2$, $27 = 3^3$, $42 = 2 \cdot 3 \cdot 7$, $84773093 = 8887 \cdot 9539$
Euclidean Algorithm

Given two positive integers and their representations as products of prime powers, it would be easy to extract from them the maximum set of common prime powers.

For example \( \gcd(18, 42) = \gcd(2 \cdot 3^2, 2 \cdot 3 \cdot 7) = 2 \cdot 3 = 6 \).

On the other hand, given just one (composite) integer, its factorization is hard to compute (in general).

*Euclidean Algorithm* is an efficient algorithm for finding the gcd of two integers. It is based on the following fact:

Let \( a > b \). Then \( \gcd(a,b) = \gcd(a \mod b, b) \).

Example: \( \gcd(42, 18) = \gcd(6, 18) = 6 \).

Example: \( \gcd(595, 408) = \gcd(187, 408) = \gcd(187, 34) = \gcd(17, 34) = 17 \).

Slowest case: Fibonacci sequence 1, 2, 3, 5, 8, 13, 21, ..., \( F_n = F_{n-1} + F_{n-2} \). For example it takes 5 iterations to compute \( \gcd(21, 13) \); in general it takes \( n-2 \) iterations to compute \( \gcd(F_n, F_{n-1}) \).
Extended Euclidean Algorithm for integers and computing a modular inverse

Fact: Given two positive integers \( a \) and \( b \) there exist integers \( u \) and \( v \) such that

\[ u \cdot a + v \cdot b = \gcd(a, b) \]

In particular, if \( \gcd(a, b) = 1 \), there exist positive integers \( u \) and \( v \) such that

\[ u \cdot a = 1 \pmod{b}, \text{ and } v \cdot b = 1 \pmod{a}. \]

The integers \( u \) and \( v \) can be computed using the Extended Euclidean Algorithm, which iteratively finds integers \( r_i, u_i \) and \( v_i \) such that

\[ r_0 = b, \quad r_1 = a; \quad u_0 = 0, \quad u_1 = 1; \quad v_0 = 1, \quad v_1 = 0 \]

and for \( i = 2,3,\ldots \) we compute \( q_i \) such that

\[ r_{i-2} = q_i \cdot r_{i-1} + r_i, \text{ where } 0 \leq r_i < r_{i-1}. \]

We set: \( u_i = u_{i-2} - q_i \cdot u_{i-1} \) and \( v_i = v_{i-2} - q_i \cdot v_{i-1} \). Then \( r_i = u_i \cdot a + v_i \cdot b \).

Let \( n \) be the index for which \( r_n > 0 \) and \( r_{n+1} = 0 \). Then

\[ r_n = \gcd(a, b) \text{ and } u_n = u \text{ and } v_n = v. \]
Extended Euclidean Algorithm: Example

\[\gcd(595, 408) = 17 = u \cdot 595 + v \cdot 408\]

<table>
<thead>
<tr>
<th>(i)</th>
<th>(q_i)</th>
<th>(r_i)</th>
<th>(u_i)</th>
<th>(v_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-</td>
<td>595</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>408</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>187</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>34</td>
<td>-2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>17</td>
<td>11</td>
<td>-16</td>
</tr>
</tbody>
</table>
Extended Euclidean Algorithm: Examples

\[ \gcd(595, 408) = 17 = 11 \cdot 595 + (-16) \cdot 408 \]
\[ = -397 \cdot 595 + 579 \cdot 408 \]

We get \[ 11 \cdot 595 = 17 \pmod{408} \]
and \[ 579 \cdot 408 = 17 \pmod{595} \]

If \( \gcd(a, b) = 1 \), this algorithm gives modular inverses.
Example: \[ 557 \cdot 797 = 1 \pmod{1047} \] that is
\[ 557 = 797^{-1} \pmod{1047} \]

If \( \gcd(a, b) = 1 \), the integers \( a \) and \( b \) are said to be coprime.
Galois Field

Given a binary polynomial \( f(x) \) of degree \( n \), consider a set of binary polynomials with degree less than \( n \). This set has \( 2^n \) polynomials. With polynomial arithmetic modulo \( f(x) \) this set is a ring.

Fact: If \( f(x) \) is irreducible, then this set with 2-ary (binary) polynomial arithmetic is a field denoted by \( GF(2^n) \).

In particular, every nonzero polynomial has a multiplicative inverse modulo \( f(x) \). We can compute a multiplicative inverse of a polynomial using the Extended Euclidean Algorithm.

The next slide presents the Extended Euclidean Algorithm for integers. It works exactly the same way for polynomials.
Extended Euclidean Algorithm for polynomials

Example

Example: Compute the multiplicative inverse of $x^2$ modulo $x^4 + x + 1$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$q_i$</th>
<th>$r_i$</th>
<th>$u_i$</th>
<th>$v_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>$x^4 + x + 1$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>$x^2$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>$x^2$</td>
<td>$x + 1$</td>
<td>$x^2$</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>$x$</td>
<td>$x$</td>
<td>$x^3 + 1$</td>
<td>$x$</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>$x^3 + x^2 + 1$</td>
<td>$x + 1$</td>
</tr>
</tbody>
</table>
Extended Euclidean Algorithm for polynomials

Example cont’d

So we get

\[ u_4 \cdot x^2 + v_4 \cdot (x^4 + x + 1) = (x^3 + x^2 + 1)x^2 + (x + 1)(x^4 + x + 1) = 1 = r_4 \]

from where the multiplicative inverse of \( x^2 \mod x^4 + x + 1 \) is equal to \( x^3 + x^2 + 1 \).
Chinese Remainder Theorem (two moduli)

Problem: Assume $m_1$ and $m_2$ are coprime. Given $x_1$ and $x_2$, how to find $0 \leq x < m_1 m_2$ such that

$x = x_1 \mod m_1$

$x = x_2 \mod m_2$

Solution: Use the Extended Euclidean Algorithm to find $u$ and $v$ such that $u \cdot m_1 + v \cdot m_2 = 1$. Then

$x = x \cdot (u \cdot m_1 + v \cdot m_2) = x \cdot u \cdot m_1 + x \cdot v \cdot m_2$

$= (x_2 + r \cdot m_2) \cdot u \cdot m_1 + (x_1 + s \cdot m_1) \cdot v \cdot m_2.$

It follows that

$x = x \mod (m_1 m_2) = (x_2 \cdot u \cdot m_1 + x_1 \cdot v \cdot m_2) \mod (m_1 m_2)$
Chinese Remainder Theorem (general case)

Theorem: Assume \( m_1, m_2, \ldots, m_t \) are mutually coprime. Denote \( M = m_1 \cdot m_2 \cdot \ldots \cdot m_t \). Given \( x_1, x_2, \ldots, x_t \) there exists a unique \( x \), \( 0 \leq x < M \), such that

\[
\begin{align*}
x &= x_1 \mod m_1 \\
x &= x_2 \mod m_2 \\
&\quad \vdots \\
x &= x_t \mod m_t
\end{align*}
\]

\( x \) can be computed as

\[
x = (x_1 \cdot u_1 \cdot M_1 + x_2 \cdot u_2 \cdot M_2 + \ldots + x_t \cdot u_t \cdot M_t) \mod M,
\]

where \( M_i = (m_1 \cdot m_2 \cdot \ldots \cdot m_t)/m_i \) and \( u_i = M_i^{-1} \mod m_i \).

Proof: For the case \( t = 2 \) see previous page. The general case was presented on the whiteboard, see also textbook.
Chinese Remainder Theorem: Example

Let \( m_1 = 7 \), \( m_2 = 11 \), \( m_3 = 13 \). Then \( M = 1001 \).

**Problem:** Compute \( x \), \( 0 \leq x \leq 1000 \) such that

- \( x = 5 \mod 7 \)
- \( x = 3 \mod 11 \)
- \( x = 10 \mod 13 \)

**Solution:**

\[
M_1 = m_2 m_3 = 143; \quad M_2 = m_1 m_3 = 91; \quad M_3 = m_1 m_2 = 77
\]

\[
u_1 = M_1^{-1} \mod m_1 = 143^{-1} \mod 7 = 3^{-1} \mod 7 = 5; \quad \text{similarly}
\]

\[
u_2 = M_2^{-1} \mod m_2 = 3^{-1} \mod 11 = 4; \quad u_3 = (-1)^{-1} \mod 13 = -1.
\]

Then

\[
x = ( 5 \cdot 5 \cdot 143 + 3 \cdot 4 \cdot 91 +10 \cdot (-1) \cdot 77) \mod 1001 = 894
\]
Euler’s Totient Function $\phi(n)$

Definition: Let $n > 1$ be integer. Then we set

$$\phi(n) = \#\{ a \mid 0 < a < n, \gcd(a,n) = 1 \},$$

that is, $\phi(n)$ is the number of positive integers less than $n$ which are coprime with $n$.

For prime $p$, $\phi(p) = p - 1$. We set $\phi(1) = 1$.

For a prime power $p^e$, we have $\phi(p^e) = p^{e-1}(p -1)$.

The multiplicative property holds:

$$\phi(m \times n) = \phi(m) \times \phi(n), \text{ for all } m, n, \gcd(m,n) = 1 .$$

Now Euler’s totient function can be computed for any integer using its prime factorisation.

Example: $\phi(18) = \phi(2 \times 3^2) = \phi(2) \times \phi(3^2) = (2-1) \times (3-1)3^1 = 6$, that is, the number of invertible (coprime with 18) numbers modulo 18 is equal to 6. They are: 1, 5, 7, 11, 13, 17.
Euler’s Theorem

\[ Z_n^* = \{ a \mid 0 < a < n, \gcd(a, n) = 1 \}, \text{and} \# Z_n^* = \phi(n) \]

**Euler’s Theorem**: For any integers \( n \) and \( a \) such that \( a \neq 0 \) and \( \gcd(a, n) = 1 \) the following holds:

\[ a^{\phi(n)} \equiv 1 \pmod{n} \]

**Fermat’s Theorem**: For a prime \( p \) and any integer \( a \) such that \( a \neq 0 \) and \( a \) is not a multiple of \( p \) the following holds:

\[ a^{p-1} \equiv 1 \pmod{p} \]
Linear Feedback Shift Registers

A binary linear feedback shift register (LFSR) is the following device

where the \( i \)\(^{th} \) tap constant \( c_i = 1 \), if the switch connected, and \( c_i = 0 \) if it is open. The contents of the register \( z_0, z_1, z_2, z_3, \ldots, z_{m-1} \) are binary values. Given this state of the device the output is \( z_0 \) and the new contents are \( z_1, z_2, z_3, \ldots, z_{m-1}, z_m \), where \( z_m \) is computed using the recursion equation

\[
z_m = c_0 z_0 + c_1 z_1 + c_2 z_2 + c_3 z_3 + \ldots + c_{m-1} z_{m-1}\]

The sum is computed \textit{modulo} 2. As this process is iterated, the LFSR outputs a binary sequence \( z_0, z_1, z_2, z_3, \ldots, z_{m-1}, z_m, \ldots \). Then the terms of this sequence satisfy the linear recursion relation

16
LFSR: The first examples

\[ z_{k+m} = c_0 z_k + c_1 z_{k+1} + c_2 z_{k+2} + c_3 z_{k+3} + \ldots + c_{m-1} z_{k+m-1} \]

for all \( k = 0, 1, 2, \ldots \)

Examples 1.

a) \( z_i = 0, \; i = 0, 1, 2, \ldots \) shortest LFSR: \[ \quad \text{(no contents, length = 0)} \]

b) \( z_i = 1, \; i = 0, 1, 2, \ldots \) shortest LFSR: \[ \quad \text{(length } m = 1) \]

c) sequence 010101... ; shortest LFSR: \[ \quad \text{(length } m = 2) \]

\[ z_0 = 0, \; z_1 = 1, \; z_{k+2} = z_k, \; k = 0, 1, 2, \ldots \]

d) sequence 0000001000000010... LFSR: \[ \quad \]
LFSR: Connection polynomial

The polynomial over $\mathbb{Z}_2$

$$f(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + \ldots + c_{m-1} x^{m-1} + x^m$$

is called the connection polynomial of the LFSR with taps $c_0, c_1, \ldots, c_{m-1}$.

Given $f(x) = c_0 + c_1 x + \ldots + c_{m-1} x^{m-1} + x^m$, of degree $m$, we denote by $f^*(x)$ the reciprocal polynomial of $f$, defined as follows:

$$f^*(x) = x^m f(x^{-1}) = c_0 x^m + c_1 x^{m-1} + c_2 x^{m-2} + \ldots + c_{m-1} x + 1.$$

It has the following properties:

1. $\deg f^*(x) \leq \deg f(x)$, and $\deg f^*(x) = \deg f(x)$ if and only if $c_0 = 1$.

2. Let $h(x) = f(x)g(x)$. Then $h^*(x) = f^*(x)g^*(x)$.

The set of sequences generated by the LFSR with connection polynomial $f(x)$ is denoted by $\Omega(f)$:

$$\Omega(f) = \{S = (z_i) | z_i \in \mathbb{Z}_2; z_{k+m} = c_0 z_k + c_1 z_{k+1} + \ldots + c_{m-1} z_{k+m-1}, k = 0, 1, \ldots \}.$$