

Blockwise-Adaptive modes of operation (a tutorial)

Antoine Joux

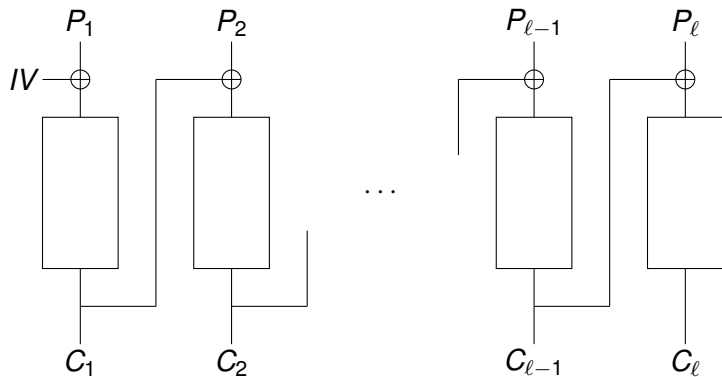
AfricaCrypt
June 24th, 2009

Modes of operation

- ▶ Block ciphers only encrypt blocks
- ▶ Need additional method for complete messages

⇒ Modes of operation

A typical mode: CBC encryption



Security for modes of operation

- ▶ Based on (chosen plaintext) distinguishers
 - ▶ First phase:
 - ▶ Distinguisher choose messages M_i and get encryptions C_i
 - ▶ Second phase:
 - ▶ Distinguisher send two messages $M^{(0)}$ and $M^{(1)}$ (same length)
 - ▶ Challenger sends back C an encryption of $M^{(b)}$
($b \leftarrow_R \{0, 1\}$)
 - ▶ Distinguisher returns a guess b'
 - ▶ Distinguisher considered successful when $b = b'$
- ▶ Let Pr be the probability of success.
- ▶ The advantage of the distinguisher is $|Pr - 1/2|$.

Example: Deterministic modes are insecure

- ▶ Deterministic modes:
 - ▶ Same message \Rightarrow Same encryption
- ▶ Define distinguisher
 - ▶ First phase:
 - ▶ Ask for C_0 : encryption of M_0
 - ▶ Second phase:
 - ▶ Send $M^{(0)} = M_0$ and $M^{(1)} \neq M_0$, receive C
 - ▶ If $C = C_0$, let $b' = 0$ else $b' = 1$
- ▶ We see that $Pr = 1$

CBC is secure up to birthday paradox

- ▶ Theorem of Bellare, Desai, Jokipi and Rogaway (1997)
 - ▶ Let E be a block cipher, then:

$$\mathbf{Adv}_{\text{CBC}(E)}(t, q) \leq 2 \cdot \mathbf{Adv}_E(t, q) + \frac{3q^2}{2^{n+1}}$$

considering adversaries running in time t and making q queries.

⇒ Security up to $q \ll 2^{n/2}$

Beyond the birthday paradox: how to attack CBC

- ▶ Need to use collisions, say $C_i = C_j$
- ▶ Recalling that $C_i = E(C_{i-1} \oplus P_i)$ we find:

$$E(C_{i-1} \oplus P_i) = E(C_{j-1} \oplus P_j)$$

- ▶ Thus, we learn that:

$$P_i \oplus P_j = C_{i-1} \oplus C_{j-1}.$$

Each collision leaks the XOR of two plaintext blocks

- ▶ Main cost: Storing $2^{n/2}$ blocks of message

Practical use of CBC

- ▶ For online applications or on low-end devices
 - ▶ Encryption is done one block at a time

⇒ Impact on security

- ▶ Bellare, Kohno, Namprempre 2002 & 2004 (SSH)
- ▶ J., Martinet, Valette 2002

Blockwise Adaptive Model

- ▶ To reflect the practice
- ▶ The adversary can submit messages block-by-block
 - ▶ Plaintext blocks can depend on previous ciphertext blocks
 - ▶ Even during the challenge step

Clearly yields stronger adversaries

A Blockwise Adaptive Distinguisher against CBC

- ▶ Define distinguisher
 - ▶ First phase:
 - ▶ Do nothing
 - ▶ Second phase:
 - ▶ Receive block 0 of ciphertext (IV)
 - ▶ Send $M_1^{(0)} = IV$ and $M_1^{(1)} = IV \oplus 1$, receive block C_1
 - ▶ Send $M_2^{(0)} = C_1$ and $M_2^{(1)} = C_1$, receive block C_2
 - ▶ If $C_1 = C_2$, let $b' = 0$ else $b' = 1$

- ▶ We see that $Pr = 1$

An easy fix: Delayed-CBC

- ▶ When M_i depends on C_{i-1}
 - ▶ We control input of E

- ▶ To avoid this, send C_{i-1} **after** receiving M_i

⇒ Enough to get a secure blockwise mode of operation

Delayed-CBC is secure up to birthday paradox

- ▶ Theorem of Fouque, Martinet and Poupard (2003)
 - ▶ Let E be a block cipher, then:

$$\mathbf{Adv}_{\text{D-CBC}(E)}(t, q) \leq 2 \cdot \mathbf{Adv}_E(t, q) + \frac{4q^2}{2^{n+1}}$$

considering **blockwise** adversaries running in time t and making q queries.

\Rightarrow Security up to $q \ll 2^{n/2}$

- ▶ Essentially the same as the security of usual CBC

Beyond the birthday paradox: attacking Delayed-CBC

- ▶ Once again need collisions . . .

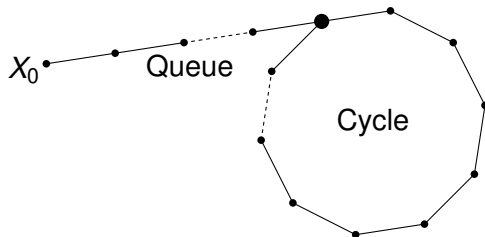
Let's first look at collision search

Generic collision algorithms

- ▶ Collision within random sets
 - ▶ Needs to store the set \Rightarrow Large memory
- ▶ Collision on outputs of a random function
 - ▶ No real need to store the set, the function is doing it !
 - ▶ Memory efficient methods exist

Cycle finding methods

- ▶ Search collisions without memory
- ▶ Apply cycle finding on $X_{n+1} = F(X_n)$ using:
 - ▶ Brent
 - ▶ Floyd
 - ▶ Nivasch



Nivasch's method

- ▶ Compute $X_{n+1} = F(X_n)$
- ▶ Store list of smallest values seen so far:

$$L_0 < L_1 < \dots < L_k$$

- ▶ Store as pairs: (L_i, n_i)
- ▶ If $L_i < X_n < L_{i+1}$, then $(L_{i+1}, n_{i+1}) \leftarrow (X_n, n)$
Delete end of L
- ▶ If $L_i = X_n$, cycle of length $n - n_i$

Cycle entrance \Rightarrow Collision

Beyond the birthday paradox: attacking Delayed-CBC

- ▶ Once again need collisions ...
- ▶ But, we can now use Nivasch's
- ▶ Let us encrypt a message defined by:
 - ▶ Let P_1 be a random value
 - ▶ Let $P_i = P_{i-1} \oplus C_{i-2}$
 - ▶ Define $Z_i = E^{-1}(C_i) = P_{i-1} \oplus C_{i-2} \oplus C_{i-1}$

- ▶ With this choice:

$$\begin{aligned}Z_{i+1} &= P_i \oplus C_{i-1} \oplus C_i \\ &= P_{i-1} \oplus C_{i-2} \oplus C_{i-1} \oplus E(Z_i) \\ &= Z_i \oplus E(Z_i)\end{aligned}$$

Nivasch's algorithm against Delayed-CBC

- ▶ Store list of smallest Z values seen so far:

$$Z^{(0)} < Z^{(1)} < \dots < Z^{(k)}$$

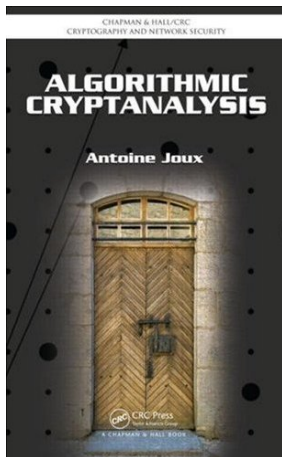
- ▶ Store as triples $(Z^{(t)}, P^{(t)}, E(Z^{(t)}))$
 - ▶ Note that if $Z^{(t)} = Z_i$, then $E(Z^{(t)}) = C_i$
 - ▶ For $P^{(t)}$, use P_i

Memoryless Distinguisher against Delayed-CBC (beyond the birthday bound)

- ▶ Second phase:
 - ▶ First, send random value as P_1 and receive C_0 (IV)
 - ▶ Initialize empty Nivasch's stack
 - ▶ Loop ($M_i^{(0)} = M_i^{(1)} = P_i$):
 - ▶ Let $P_i = P_{i-1} \oplus C_{i-2}$
 - ▶ Send P_i and receive C_{i-1}
 - ▶ Compute $Z_i = E^{-1}(C_i) = P_{i-1} \oplus C_{i-2} \oplus C_{i-1}$
 - ▶ Search/store (Z_i, P_i, \cdot) in Nivasch's stack (**break** if found)
 - ▶ (If needed) update Z_{i-1} in stack as $(Z_{i-1}, P_{i-1}, C_{i-1})$
 - ▶ Pick pair $(Z_i, P_{\text{ref}}, C_{\text{ref}})$ from Nivasch's stack
 - ▶ Send $M_{i+1}^{(0)} = P_{\text{ref}}$ and $M_{i+1}^{(1)} = P_{\text{ref}} \oplus 1$
 - ▶ Receive block C_{i+1}
 - ▶ If $C_{i+1} = C_{\text{ref}}$, let $b' = 0$ else $b' = 1$

We see that $Pr = 1$

Conclusion



<http://www.joux.biz/algcrypt>