3.3 블록 암호 – TEA

- TEA(Tiny Encryption Algorithm) 알고리즘
  - “아주 단순한 블록 암호 "
  - 64비트 블록 길이, 128비트 키 사용
  - 32비트 컴퓨터 구조를 가정해 작성되었으므로 모든 연산은 모듈로 232
  - 32번째 넘는 비트는 자동 생략된다는 의미
  - 매우 짧은 코드, CPU 자원 적게 소모 -> 임베디드 기기에서 유용
  - TEA는 아주 간단한 회전 함수를 사용하므로 AES와 완전 반대
  - 아주 간단한 회전 함수를 사용하는 TEA가 높은 보안을 달성하려면 회전의 수가 충분히 커야 함

3.3 블록 암호 – TEA

- TEA(Tiny Encryption Algorithm) 알고리즘
  - Feistel 구조 유사: 좌우 32비트로 나눠 연산
  - 각 라운드는 다음 연산으로 구성:
    - 덧셈 (mod 2^{32})
    - XOR
    - 비트 시프트 ($\ll, \gg$)
  - 32 라운드(64 Feistel step) 권장
  - Delta = 0x9E3779B9
    - 골든 레이쇼 관련 수
    - 키와 평문 패턴 상관관계 방지

표 3-6 TEA 암호화

$(K[0],K[1],K[2],K[3])$=128 비트 키
(L,R) = 평문(64비트 블록)
delta = 0x9e3779b9
sum = 0
for i = 1 to 32
    sum = sum + delta
    $L = L + (((R \ll 4) + K[0]) \oplus (R + \text{sum}) \oplus R \gg 5) + K[1]))$
    $R = R + (((L \ll 4) + K[2]) \oplus (L + \text{sum}) \oplus ((L \gg 5) + K[3]))$
next i
암호문=(L,R)

3.3 블록 암호 – TEA

■ TEA 복호화

표 3-7 TEA 복호화

$(K[0],K[1],K[2],K[3])$=128 비트 키

(L,R) = 암호문(64비트 블록)

delta = 0x9e3779b9

sum = delta $\ll 5$

for $i = 1$ to 32

$R = R - (((L \ll 4) + K[2]) \oplus (L + \text{sum}) \oplus ((L \gg 5) + K[3]))$

$L = L - (((R \ll 4) + K[0]) \oplus (R + \text{sum}) \oplus R \gg 5) + K[1]))$

sum = sum − delta

next i

평문=(L,R)

3.3 블록 암호 – TEA

TEA(Tiny Encryption Algorithm) 알고리즘 예시

- 가정:
  - L = 0x01234567, R = 0x89ABCDEF
  - K = (0x0, 0x1, 0x2, 0x3)
  - Δ = 0x9E3779B9
- 1라운드만 수행:
  - sum = Δ
  - L′ = 0x01234567 + ((0x89ABCDEF << 4) + 0x0) ⊕ (0x89ABCDEF + sum) ⊕ ((0x89ABCDEF >> 5) + 0x1)
  - R′ =

3.3 블록 암호 – TEA

TEA(Tiny Encryption Algorithm) 알고리즘

- 장점
  - 단순성: 구현 코드가 수십줄
  - 성능 : 32비트 연산기에서 빠름
  - 메모리 차지 거의 없음
  - 특허 X(누구나 사용 가능)
- 단점
  - 키 관련 약점: 키의 특정 조합에서 서로 다른 키가 같은 암호문을 생성(weak keys)
  - 관련 키 공격(Related-Key Attack) 가능
  - 현대 보안 표준 에서 사용 권장 x

3.3 블록 암호 – TEA

TEA 관련 키 공격

- 두 개의 TEA 메시지가 관련된 키로 암호화되어 있다는 것을 알게 되면 두 개의 평문은 복구 가능
- 그러나 무시해도 될 만큼 성공 확률이 매우 낮음

3.3 블록 암호 – 블록 암호 모드

■ ECB 모드

- 전자 코드북(ECB; Electronic CodeBook) 모드
- 원리: 각 블록을 독립적으로 같은 키로 암호화
  - $C_i = E(P_i, K), P_i = D(C_i, K)$
- 특징:
  - 구현이 간단, 병렬 처리 가능
  - 하지만 같은 평문 블록 -> 같은 암호문 블록 -> 패턴이 그대로 드러남
- 압축되지 않은 앨리스 이미지와 ECB 모드로 암호화된 앨리스 이미지

- 공격자가 암호문으로부터 평문을 추정하는 것이 어렵지 않음
  - 같은 단어는 항상 같은 코드로 바뀌기 때문에 문장 구조가 드러남

(a) 앨리스

(b) ECB 모드로 암호화된 앨리스

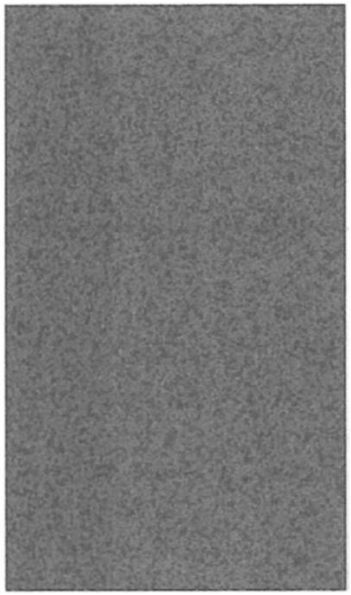
그림 3-3 앨리스는 ECB 모드를 싫어한다.

3.3 블록 암호 – 블록 암호 모드

■ CBC 모드

- 이전 암호문 블록을 현재 평문 블록과 XOR 후 암호화
  - $C_i = E(P_i \oplus C_{i-1}, K)$, $P_i = D(C_i, K) \oplus C_{i-1}$
- 암호 블록 연결(Cipher Block Chain) 모드
- 동일한 평문이 동일한 암호문을 생산하지 않는 것이 장점
- 특징
  - 같은 평문 블록이어도 앞 블록의 암호문에 따라 결과가 달라짐 -> 패턴 숨김
  - 직렬적 구조 -> 병렬화 어려움

(a) 앨리스



(b) CBC 모드로 암호화된 앨리스

그림 3-4 앨리스는 CBC 모드를 사랑한다.

3.3 블록 암호 – 블록 암호 모드

- CTR 모드
  - 임의 접근이 필요할 때 사용
- ECB, CBC, CTR 모드는 주로 사용하는 블록 암호
- 기밀성을 위한 블록 암호 방식
  - 데이터를 암호화해서 불안정한 채널에서 전송될 수 있도록 하는 것
  - 컴퓨터 하드드라이브처럼 불안정한 미디어에 저장되는 데이터를 암호화하는 것
  - 대칭키 암호는 두 문제 중 어느 것이든 해결하는 데 사용 가능

3.3 블록 암호 – 블록 암호 모드

■ CTR 모드

- 블록 암호로 부터 스트림 키를 만들어 평문과 XOR
  - $C_i = P_i \oplus E(IV + i, K), P_i = C_i \oplus E(IV + i, K)$
- 임의 접근이 필요할 때 사용

- ECB, CBC와 달리 블록 암호를 스트림 암호처럼 사용
- 병렬 처리 가능, 빠르고 효율적
- IV(Nonce) 관리가 매우 중요(재사용 시 치명적 보안 취약점)

- 파일 전송, 네트워크 암호화, 디스크 암호화에 많이 활용

3.3 블록 암호 – 블록 암호 모드

- ECB, CBC, CTR 모드는 주로 사용하는 블록 암호

- 기밀성을 위한 블록 암호 방식
  - 데이터를 암호화해서 불안정한 채널에서 전송될 수 있도록 하는 것
  - 컴퓨터 하드드라이브처럼 불안정한 미디어에 저장되는 데이터를 암호화 하는 것
  - 대칭키 암호는 두 문제 중 어느 것이든 해결하는 데 사용 가능

모드	장점	단점	예시/비유
ECB	단순, 병렬 처리 쉬움	패턴 노출, 보안 취약	그림 암호화 시 원본 윤곽 드러남
CBC	패턴 은닉, 보안 ↑	직렬 처리, IV 필요	도미노처럼 연쇄 효과
CTR	병렬 처리 가능, 빠름	IV 재사용 취약	일회용 번호표 방식

3.4 무결성

■ 기밀성과 무결성

- 기밀성은 비인가자가 읽는 행위를 방지하는 것
- 무결성은 비인가자가 쓰는 행위를 탐지하는 것

■ 메시지 인증 코드(MAC; Message Authentication Code)

- 데이터 무결성을 보장하기 위해 블록 암호 사용
- MAC 계산 공식

$$C_0 = E(P_0 \oplus \mathrm{IV}, K),\ C_1 = E(P_1 \oplus C_0, K), \ldots,$$

$$C_{N-1} = E(P_{N-1} \oplus C_{N-2}, \mathrm{K}) = \mathrm{MAC}$$

3.4 무결성

■ MAC

- MAC는 CBC 암호화에서 거의 언제나 마지막 블록까지 전파된다는 점을 이용
- 이것이 MAC가 무결성을 제공할 수 있는 특징
- 효율성을 위해 데이터를 CBC 모드로 단 한 번 암호화 후 기밀성과 무결성 보호를 둘 다 가지는 것이 유용함

3.5 양자 컴퓨터와 대칭 암호

■ 양자 컴퓨팅

- 강력한 컴퓨터를 만들기 위해 양자역학의 특징을 이용하는 것
- 선택 문제에 대한 엄청난 성능 증가 가능
- 양자 컴퓨팅을 위한 특별한 알고리즘이 필요

■ 양자 컴퓨터와 대칭 암호 보안

- 일반 대칭 암호 공격에 사용 가능한 최선의 양자 알고리즘은 1996년에 로브 그로버(Lov Grover)가 개발한 알고리즘
- 양자 컴퓨팅은 공개키 암호 시스템에 위협적

3.5 양자 컴퓨터와 대칭 암호

■ 양자 컴퓨터란?

- 기존 디지털 컴퓨터: 비트를 사용 -> 0 또는 1의 두가지 값만 가짐
- 양자 컴퓨터: 큐비트를 사용 -> 0과 1을 동시에 표현할 수 있음
- 큐비트끼리 얽히면, 다수 큐비트가 동시에 다양한 계산을 병렬적으로 수행 가능
- 강력한 컴퓨터를 만들기 위해 양자역학의 특징을 이용하는 것
- 선택 문제에 대한 엄청난 성능 증가 가능
- 양자 컴퓨팅을 위한 특별한 알고리즘이 필요

3.5 양자 컴퓨터와 대칭 암호

■ 양자 컴퓨터와 대칭 암호 보안

- 일반 대칭 암호 공격에 사용 가능한 최선의 양자 알고리즘은 1996년에 로브 그로버(Lov Grover)가 개발한 알고리즘
- 대칭 암호(블록 암호, 스트림 암호)에는 상대적으로 덜 위협적
- 양자 컴퓨팅은 공개키 암호 시스템에 위협적

3.5 양자 컴퓨터와 대칭 암호

■ Grover 알고리즘

- 기존 컴퓨터에서 n비트 키 탐색: 2^n번 시도 필요
- 양자 컴퓨터에서 Grover 알고리즘 적용: $2^{\frac{n}{2}}$ 번 시도로 줄어듬

- DES(키 길이 56비트):
  - 전통: $2^{56} \approx 7.2 \times 10^{16}$ 번 시도
  - 양자: $2^{56} \approx 7.2 \times 10^{8}$ 번 시도 -> 훨씬 더 빨리 깨짐
- AES-128
  - 전통: 2^{128}
  - 양자: 2^{64} → 슈퍼컴퓨터 + 양자 기술 결합시 이론적 가능성 존재

■ 대응 방안

- 대칭 암호의 키 길이를 늘이면 양자 공격에 충분히 안전해짐
  - AES-256 -> 양자 공격에 대해서도 안전 2^{128}

3.6 요약

■ 대칭키 암호

- 스트림 암호와 블록 암호
- 스트림 암호는 실용성을 위해 보안성이 다소 약화된 일회성 암호체계를 일반화한 것으로 A5/1, RC4가 스트림 암호임
- 블록 암호는 고전 코드북의 전자화된 형태로 DES, AES, TEA 블록 암호 해당
- 블록 암호를 사용한 다양한 모드로 ECB 모드, CBC 모드, CTR 모드가 있으며 CBC 모드에 기반한 블록 암호가 데이터 무결성을 제공
- 인증 프로토콜에 유용