TTA Standard

정보통신단체표준(국문표준)

TTAK.KO-12.0273-Part1 개정일: 2018년 12월 xx일

HMAC 기반 키 유도 함수
- 제1부: 일반

HMAC-based Key Derivation Functions
- Part1: General

한국정보통신기술협회
Telecommunications Technology Association

표준초안 검토 위원회 정보보호기반 프로젝트그룹(PG501)

표준안 심의 위원회 정보보호 기술위원회(TC5)

	성명	소 속	직위	위원회 및 직위	표준번호
표준(과제) 제안	박제홍	NSR	책임연구원	정보보호기반 프로젝트그룹 위원	TTAK.KO-12.0273-Part1
표준 초안 작성자	박제홍	NSR	책임연구원	정보보호기반 프로젝트그룹 위원	TTAK.KO-12.0273-Part1
	김우환	NSR	책임연구원	-	TTAK.KO-12.0273-Part1
	한상윤	NSR	선임연구원	정보보호기반 프로젝트그룹 위원	TTAK.KO-12.0273
사무국 담당	박수정	TTA	책임	-	

본 표준 발간 이전에 접수된 지식재산권 확약서 정보는 본 표준의 '부록(지식재산권 확약서 정보)'에 명시하고 있으며, 이후 접수된 지식재산권 확약서는 TTA 웹사이트에서 확인할 수 있습니다.
본 표준과 관련하여 접수된 확약서 외의 지식재산권이 존재할 수 있습니다.

발행인 : 한국정보통신기술협회 회장
발행처 : 한국정보통신기술협회
13591, 경기도 성남시 분당구 분당로 47
Tel : 031-724-0114, Fax : 031-724-0109
발행일 : 2018.12

서 문

1 표준의 목적

키 유도 함수(Key Derivation Function)는 하나의 마스터 키로부터 다양한 용도의 암호 알고리즘 또는 다수의 개체가 사용할 개별 암호 키를 생성하는 용도로 사용된다. 이 표준은 해시 함수 기반 메시지 인증 코드 알고리즘인 HMAC을 사용하는 키 유도 함수를 제시하여, 효과적인 암호 키 관리를 가능하게 한다.

2 주요 내용 요약

이 표준은 해시 함수 기반 메시지 인증 코드 알고리즘인 HMAC을 의사 난수 함수(PRF, pseudo random function)로 사용하는 세 가지 키 유도 함수를 제시한다.

3 인용 표준과의 비교

3.1 인용 표준과의 관련성

이 표준은 NIST SP 800-108에 규정된 세 가지 키 유도 함수의 상세 규격을 준용하며, 기반 함수인 의사 난수 함수(PRF)를 HMAC으로 한정한다.

3.2 인용 표준과 본 표준의 비교표

TTAK.KO-12.0273-Part1	NIST SP 800-108	비고
1. 적용 범위	1. Introduction 2. Scope and Purpose	동일(번역)
2. 인용 표준	-	추가
3. 용어 정의	3. Definitions, Symbols and Abbreviations	동일(번역)
4. 약어		
5. 키 유도 함수	4. Pseudorandom Function (PRF) 5. Key Derivation Functions (KDF)	동일(번역)
-	6. Key Hierarchy	제외(일반 개념)
6. 안전성 고려사항	7. Security Considerations	동일(번역)

Preface

1 Purpose

For the purpose of generating individual cryptographic keys required for various cryptographic algorithms or for multiple entities, a key derivation function which derives such keys can be used. The standard provides techniques for the derivation of additional keys from a given master secret key using HMAC as an instance of a pseudorandom function, to make possible for efficient cryptographic key management.

2 Summary

The standard presents three key derivation functions using HMAC as a pseudorandom function.

3 Relationship to Reference Standards

The standard conforms to the detailed specifications of the three key derivation functions(KDFs) in NIST SP 800-108, but restricts the basic building block of such KDFs to HMAC.

목 차

HMAC 기반 키 유도 함수
- 제1부: 일반
(HMAC-based Key Derivation Functions
- Part1: General)

1 적용 범위

정보 시스템에서 유통되는 데이터에 대한 보호를 위해서는 다양한 암호 알고리즘의 적용이 필요하다. 데이터에 직접 적용되는 블록 암호(block cipher)나 메시지 인증 코드(message authentication code)와 같은 대칭 키 암호(symmetric key cryptography)가 제공하는 정보보호 서비스는, 암호 알고리즘에 사용되는 암호 키의 안전한 관리를 기반으로 성립한다.

대칭 키 암호 알고리즘의 운용에 필요한 암호 키는 키 설정 프로토콜(key establishment protocol)[4]이나 오프라인 경로를 통한 사전 분배(pre-shared) 등 다양한 방법을 통해 안전하게 공유될 수 있다. 그런데 데이터의 보호를 위해서 다양한 종류의 암호 알고리즘을 적용하는 경우, 암호 알고리즘의 용도에 따라 암호 키를 분리 적용하는 것이 일반적이다[6]. 그러므로 키 관리 정책에 따라 실제로는 한 개가 아닌 여러 개의 암호 키를 공유해야 할 필요가 발생할 수 있다.

다수의 암호 키를 효과적으로 공유하는 방법으로 사용하는 것이 키 유도 함수(KDF, key derivation function)이다. 키 설정 프로토콜이나 사전 공유 과정을 통해 공유하는 암호 키의 개수와 크기를 최소화하는 대신, 공유된 암호 키를 키 유도 함수(KDF)에 입력하여 필요한 개수만큼의 암호 키를 생성하는 것이다. 이와 유사한 경우로, 키 유도 함수(KDF)는 키 서버를 운용하는 시스템에서 다수의 개체(entity)에 분배하는 서로 다른 암호 키를 생성하는 용도로도 사용할 수 있다.

일부 키 설정 프로토콜은 자체적으로 키 유도 함수(KDF)를 정의하고 있다[5]. 이런 경우 암호 키 유도 과정은 압축 후 확장(extraction-then-expansion) 절차[5,7]를 준용할 수 있다. 압축 단계에서는 프로토콜에서 정의하는 키 유도 함수를 이용하여 마스터 키를 생성하고, 확장 단계에서는 이 마스터 키를 본 표준에서 제시하는 키 유도 함수에 적용하여 필요한 수의 암호 키를 생성한다.

2 인용 표준

NIST SP 800-108 (2009), Recommendation for Key Derivation Using Pseudorandom Functions (Revised).
(세 가지 키 유도 함수의 상세 규격을 준용하며, 기반 함수인 의사 난수 함수(PRF, pseudo random function)를 HMAC으로 한정함)

3 용어 정의

3.1 개체(entity)
개인(사람), 조직, 장치 또는 이들의 조합으로, 이 표준에서는 특정 프로세스를 실행하는 함수 단위를 가정

3.2 논스(nonce)
한 번 사용된 후 재사용되지 않는 것을 보장할 수 있는 값

3.3 메시지 인증 코드(MAC, message authentication code)
데이터에 대한 무결성 제공을 위해 사용되는 대칭 키 암호 알고리즘으로, 임의 길이 입력 데이터에 작용하여 고정된 길이의 인증 태그(authentication tag)를 생성

3.4 암호 키(cryptographic key)
암호 알고리즘에 의해 비밀 파라미터로 사용되는 비트열로, 명시된 길이의 랜덤 비트열이거나 랜덤 비트열과 계산상으로 구별 불가능한 같은 길이의 의사 난수 비트열

3.5 의사 난수 함수(PRF, pseudo random function)
주어진 정의역(domain)과 치역(range)에 대해 정의되는 모든 함수(function)의 집합에서 균등한 확률(uniform probability)에 따라 무작위로 선택된 함수(random function)와 구별이 어려우면서, 효율적으로 계산 가능한 함수

3.6 키 설정(key establishment)
둘 이상의 참여자에 의해 수행되는 과정으로, 그 결과로 얻어지는 키 요소(keying material)는 모든 참여자에 의해 공유

3.7 키 요소(key material)
요구 길이를 가지는 중첩되지 않는 각 부분(non-overlapping segments with the required lengths)이 대칭 키 암호 알고리즘의 암호 키로 사용될 수 있는 비트열

3.8 키 유도(key derivation)
하나의 암호 키로부터 키 요소(keying material)를 얻는 과정

3.9 키 유도 키(KDK, key derivation key)
다른 암호 키를 얻기 위해 키 유도 함수의 입력으로 사용되는 암호 키

3.10 파이프라인(pipeline)
일련의 순차적인 의사 난수 함수(PRF) 실행

3.11 해시 함수(hash function)
임의 길이의 메시지를 입력으로 받아 일정 길이의 출력값으로 압축하며, 다음 두 가지 성질을 가지는 암호 알고리즘
- 주어진 출력값에 대응하는 입력값을 찾는 것이 어려움
- 주어진 입력값에 대해, 같은 출력값을 가지는 다른 입력값을 찾는 것이 어려움

[출처] NIST SP 800-108

4 약어

HMAC	Keyed-hash Message Authentication Code
NIST	National Institute of Standards and Technology
NIST SP	NIST Special Publication

5 키 유도 함수

5.1 개요

키 유도 함수(KDF, key derivation function)는 마스터 키와 부가 데이터를 입력으로 받아 암호 알고리즘에 직접 사용될 암호 키를 생성하는 함수이다. 키 유도 함수에 사용되는 마스터 키를 키 유도 키(KDK)라고 부르며, 난수 발생기 또는 키 설정 프로토콜을 통해 생성될 수 있다. 키 유도 키(KDK)가 키 설정 프로토콜을 통해 생성되는 경우, 키 유도 키(KDK)는 키 설정 절차의 결과인 비밀 키 요소(secret keying material)의 일부분(segment)이어야 한다. 여기에서 비밀 키 요소(secret keying material)는 공개키 키 공유 방식과 같은 키 설정 프로토콜의 수행 과정에서 계산된 대수적 형태(예를 들어 유한체 GF(p)에서의 Diffie-Hellman 값 g^{xy})의 공통키(shared secret value)가 아닌, 이를 키 유도 함수를 통해 가공한 랜덤 또는 의사 난수 비트열을 의미한다.

요구되는 길이를 가지면서 중첩되지 않도록 구분된 키 요소(non-overlapping segments with the required lengths)의 각 부분(segment)은 개별 암호 알고리즘의 암호 키로 사용

될 수 있다. 그러나 데이터를 수신하는 다른 개체들도 유도된 키 요소로부터 같은 암호 키를 얻기 위해서는, 키 요소를 처리하는 방법을 정의해야 한다. 예를 들어, 키 유도 함수를 통해 얻은 256 비트 키 요소에서 처음 128 비트는 메시지 인증 코드 알고리즘의 암호 키로 사용하고, 두 번째 128 비트는 암호화 알고리즘을 위한 암호 키로 사용하는 것과 같은 방법을 정의할 수 있다.

이 표준에서 정의하는 키 유도 함수는 응용에서 요구하는 키 요소의 길이에 따라 기반 의사 난수 함수(PRF)를 여러 번 호출할 수 있다. 의사 난수 함수(PRF)를 여러 번 호출하여 반복 사용하는 방법을 반복 모드(mode of iteration)라고 부르며, 이 표준에서는 카운터 모드(5.4절 참조), 피드백 모드(5.5절 참조), 그리고 더블-파이프라인 반복 모드(5.6절 참조)를 정의한다.

키 유도 함수는 의사 난수 함수(PRF)를 n번 반복하여 키 요소의 크기가 L 비트가 될 때까지 출력을 차례로 연접시킨다. 의사 난수 함수(PRF) 출력의 비트 길이를 h라 하고 $n = \lceil L/h \rceil$ 이라 하면, 각 반복 모드에서 n의 조건은 다음과 같다.

- 카운터 모드: $n < 2^r$ (r: 카운터의 비트열 표현에 필요한 비트 길이, $r \leq 32$),
- 피드백 모드, 더블-파이프라인 반복 모드 : $n < 2^{32}$.

의사 난수 함수(PRF)의 반복 과정에서, 키 유도 키(KDK) K_I는 의사 난수 함수(PRF)의 암호 키로 사용되고, 입력 데이터는 반복 변수와 고정 데이터 문자열로 구성된다. 반복 모드에 따라, 반복 변수는 카운터나 이전 단계에서의 의사 난수 함수(PRF) 출력값, 또는 이 둘의 결합이 될 수 있으며, 더블-파이프라인 반복 모드의 경우에 첫 번째 파이프라인으로부터의 출력이 될 수 있다. 그리고 의사 난수 함수(PRF) 입력 데이터의 고정 문자열은 Label, 구분 표식 0x00, Context, 그리고 $[L]_2$의 연접으로 구성된다. 필요에 따라서 고정된 입력 데이터 필드의 일부 요소는 생략할 수 있다.

5.2 기호

키 유도 함수의 규격을 설명하기 위해 사용하는 기호를 정리하면 다음과 같다.

A(i)	더블 파이프라인 반복 모드의 첫 번째 파이프라인에서 i 번째 반복의 출력
A ‖ B	비트열 A와 B의 병합
Context	키 요소의 생성과 관련된 정보(생성에 참여한 개체 또는 생성된 키 요소를 사용하는 개체의 식별자(ID), 그리고 선택적으로 키 유도 과정에서 사용되는 키 공유 논스 등을 포함)를 포함하는 비트열
h	의사 난수 함수(PRF) 출력의 비트 길이
H()	해시 함수

i	각 반복을 표시하는 카운터로, 각 반복 계산 단계에서 의사 난수 함수(PRF) 입력으로 사용될 때는 길이가 r인 비트열로 표현됨
IV	피드백 모드에서 최초 의사 난수 함수(PRF) 계산 시 초기값으로 사용되는 비트열로 값이 없는 경우도 허용
K(i)	i 번째 반복 실행한 의사 난수 함수(PRF)의 출력
K_I	키 요소 K_O를 생성하는데 사용되는 키 유도 키(KDK)
K_O	키 유도 키(KDK) K_I와 다른 데이터로부터 얻어지는 키 요소
L	키 요소 K_O의 비트 길이로, 키 유도 함수의 입력으로 사용될 때는 비트열로 표현됨
Label	키 요소의 생성 목적을 표시하는 문자열로 비트열로 인코딩되어 키 유도 함수의 입력으로 사용됨
n	키 요소를 생성하기 위해 필요한 의사 난수 함수(PRF) 반복 횟수
PRF(s, x)	씨드 s와 데이터 x를 입력받는 의사 난수 함수(PRF)
r	32와 같거나 작은 정수로, 키 유도 함수에서 각 의사 난수 함수(PRF) 반복 처리 카운터 i에 대한 이진 표현의 비트 길이
|S|	비트열 S의 비트 길이
$[T]_2$	입력값 T에 대해 함수, 알고리즘, 또는 프로토콜에서 명시하는 길이로 표현된 비트열
w	키 유도 키의 비트 길이를 표시하는 정수
{X}	데이터 X가 키 유도 함수에 대한 선택적 입력임을 표시
⌈X⌉	X와 같거나 큰 가장 작은 정수. X의 올림
∅	빈 비트열. 비트열 A에 대하여, ∅ ‖ A = A ‖ ∅ = A

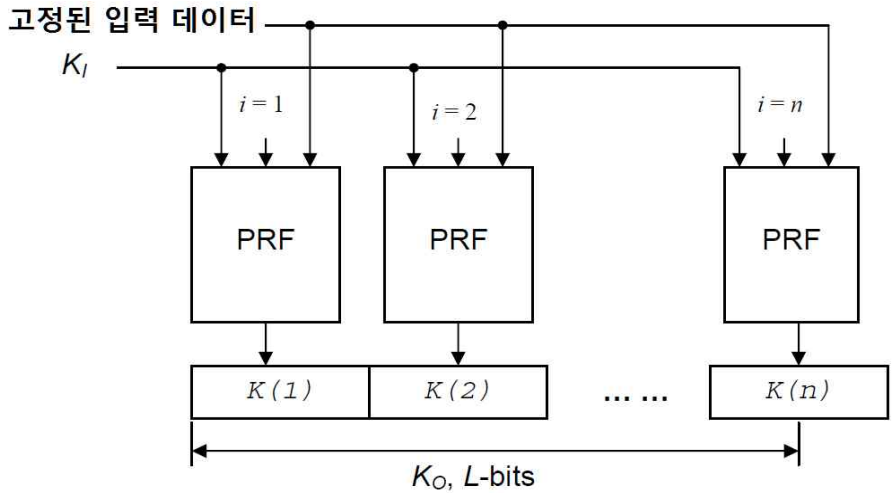
5.3 의사 난수 함수

의사 난수 함수(PRF)는 키 유도 함수를 정의하는 핵심 요소이다. 일반적으로 의사 난수 함수족 $\{PRF(s, x) \mid s \in S\}$는 씨드라 불리는 색인 s와 입력 변수 x를 가진 다항식 시간 계산 가능 함수로 구성된다. s가 색인 집합 S로부터 무작위로 선택되고 외부로 알려지지 않았을 때, PRF(s, x)는 같은 입출력 범위를 가지는 랜덤 함수와 계산상으로 구별 불가능(computationally indistinguishable)하다.

암호 키 K_I를 의사 난수 함수(PRF)의 씨드로 사용할 때, 의사 난수 함수(PRF)의 출력은 키 요소로 사용될 수 있다. 이 표준에서는 해시 함수 기반의 메시지 인증 코드 알고리즘인 HMAC을 의사 난수 함수(PRF)로 간주한다. HMAC을 이용한 키 유도 함수에 대해, 키 유도 키(KDK) K_I는 길이가 $|K_I|$인 모든 비트열의 집합에서 균일한 확률 분포에 따라 무작위로 선택된 것으로 가정한다.

5.4 카운터 모드를 이용한 키 유도 함수

카운터 모드를 이용한 키 유도 함수의 출력은 의사 난수 함수(PRF)의 입력값에 포함된 카운터를 순차적으로 증가시키며 얻은 의사 난수 함수(PRF) 출력값을 순서대로 연접하여 얻는다. 카운터 모드를 이용한 키 유도 함수의 동작 방식은 (그림 5-1)과 같다.



(그림 5-1) 카운터 모드를 이용한 키 유도 함수

카운터 모드를 이용한 키 유도 함수 구체적인 절차는 알고리즘 1과 같다.

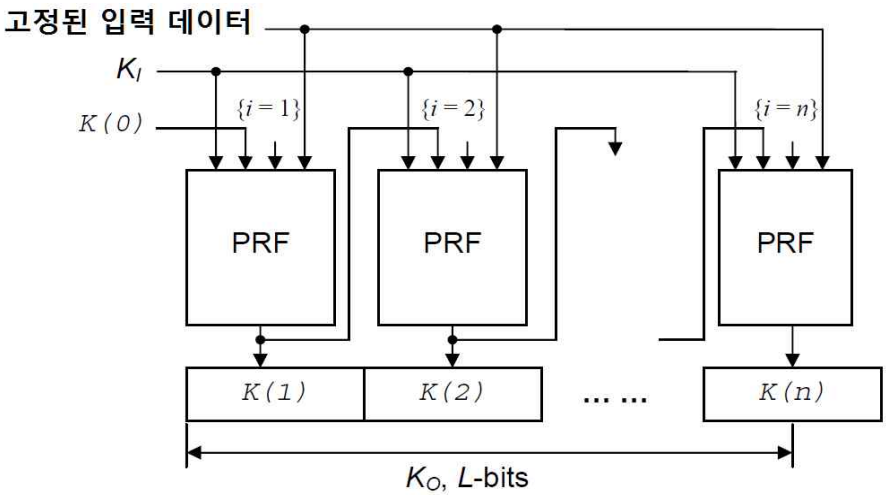
알고리즘 1 카운터 모드를 이용한 키 유도 함수	
고정된 값: 1. h - 의사 난수 함수(PRF) 출력의 비트 길이. 2. r - 카운터에 대한 이진 표현의 길이.	
입력: K_I (키 유도 키), Label, Context, L	
출력: 키 요소 K_O	
1:	$n \leftarrow \lceil L/h \rceil$
2:	**if** $(n > (2^r - 1))$ **then**
3:	**return** ERROR_FLAG
4:	**end if**
5:	result(0) $\leftarrow \varnothing$
6:	**for** i = 1 **to** n **do**
7:	$K(i) \leftarrow HMAC(K_I, [i]_2 \| Label \| 0x00 \| Context \| [L]_2)$
8:	$result(i) \leftarrow result(i-1) \| K(i)$
9:	**end do**
10:	$K_O \leftarrow$ **leftmost**(result(n), L)
11:	**return** K_O

각 반복 과정에서, 의사 난수 함수(PRF)의 고정 입력 데이터는 비트열 Label‖0x00‖Context‖$[L]_2$ 이다. 순차별로 증가하는 카운터 $[i]_2$는 반복 변수이고, r 비트 길이의 비트열로 표현된다.

카운터 모드를 이용한 키 유도 함수에서 반복 횟수 n의 상한은 $2^r - 1$이다. 따라서 HMAC의 입력을 구성하는 카운터 $[i]_2$는 특정 키 유도 함수 동작 과정에서 반복되지 않는다. 이러한 제한에 의해 키 요소의 길이 L의 상한은 $(2^r - 1)h$로 정해진다.

5.5 피드백 모드를 이용한 키 유도 함수

피드백 모드를 이용한 키 유도 함수의 출력은 이전 단계의 의사 난수 함수(PRF) 출력값과 (선택적으로) 카운터를 의사 난수 함수(PRF) 입력값으로 활용하여 얻은 출력값을 차례로 연접하여 얻는다. 피드백 모드를 이용한 키 유도 함수의 동작 방식은 (그림 5-2)와 같다.



(그림 5-2) 피드백 모드를 이용한 키 유도 함수

피드백 모드를 이용한 키 유도 함수의 구체적인 절차는 알고리즘 2와 같다. ($L \leq h$, IV = ∅, 그리고 카운터가 사용될 때, 피드백 모드는 카운터 모드와 같은 출력값을 생성한다.)

각 반복 과정에서, 의사 난수 함수(PRF)의 고정 입력 데이터는 비트열 Label‖0x00‖Context‖$[L]_2$ 이다. 이전 의사 난수 함수(PRF) 출력 K(i - 1)과 순차별로 증가하는 카운터 $[i]_2$는 반복 변수이다.

알고리즘 2 피드백 모드를 이용한 키 유도 함수

고정된 값:

1. h - 의사 난수 함수(PRF) 출력의 비트 길이
2. r - 카운터에 대한 이진 표현의 길이 (r은 카운터가 입력으로 사용될 때만 명시)

입력: K_I (키 유도 키), Label, Context, IV, L

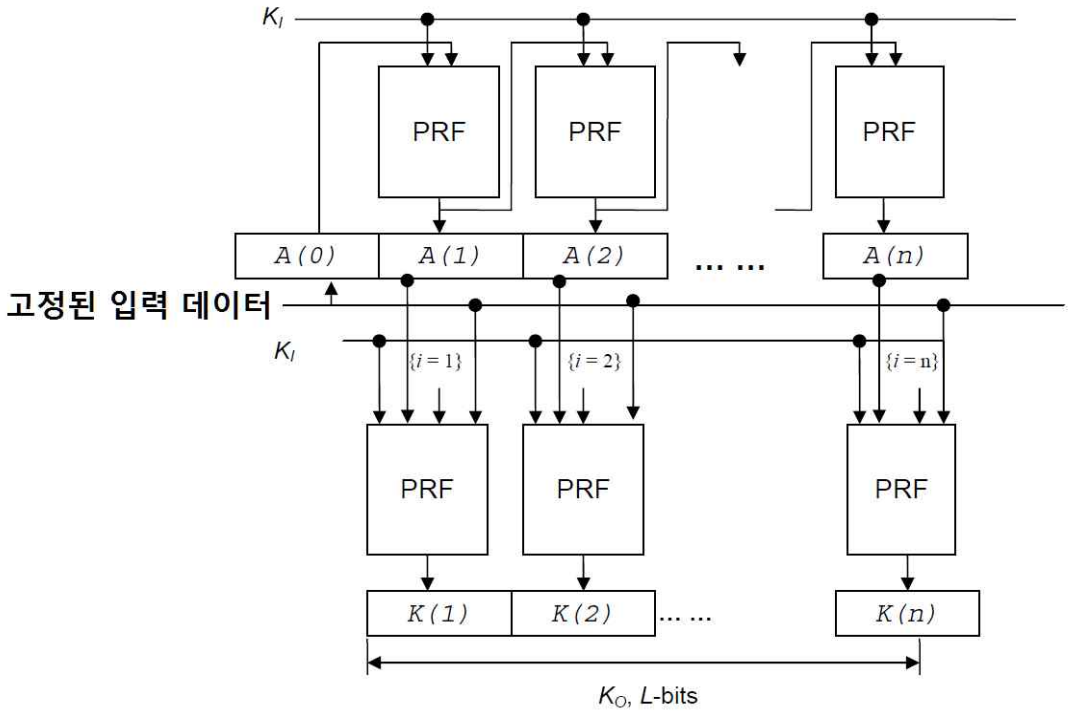
출력: 키 요소 K_O

```
n ← ⌈L/h⌉
if (n > (2^32 - 1)) then
    return ERROR_FLAG
end if
result(0) ← ∅
K(0) ← ∅
for i = 1 to n do
    K(i) ← HMAC(K_I, K(i - 1) {|| [i]_2} || Label || 0x00 || Context || [L]_2)
    result(i) ← result(i - 1) || K(i)
end do
K_O ← leftmost(result(n), L)
return K_O
```

5.6 더블-파이프라인 반복 모드를 이용한 키 유도 함수

카운터 모드(5.4절) 또는 피드백 모드(5.5절)를 이용한 키 유도 함수에서 의사 난수 함수(PRF)는 단일 파이프라인에서 반복 호출된다. 더블-파이프라인 반복 모드는 의사 난수 함수(PRF)를 두 개의 파이프라인에서 반복하도록 정의한다. 첫 번째 파이프라인에서, 비밀 값 A(i)의 수열이 생성되고, 각각의 블록은 두 번째 파이프라인에서 해당하는 순번의 의사 난수 함수(PRF)에 대한 입력값으로 사용된다. 더블-파이프라인 모드를 이용한 키 유도 함수의 동작 방식은 (그림 5-3)과 같고, 구체적인 절차는 알고리즘 3과 같다.

첫 번째 반복 파이프라인은 $A(0) = Label \| 0x00 \| Context \| [L]_2$를 초기값으로 하는 피드백 모드로 동작한다. 두 번째 파이프라인은 A(i)와 선택적으로 카운터 $[i]_2$를 반복 변수로 의사 난수 함수(PRF)에 입력하여 출력 K(i)를 생성한다.



(그림 5-3) 더블-파이프라인 반복 모드를 이용한 키 유도 함수

알고리즘 3 더블-파이프라인 반복 모드를 이용한 키 유도 함수
고정된 값: 1. h - 의사 난수 함수(PRF) 출력의 비트 길이 2. r - 카운터에 대한 이진 표현의 길이 (r은 카운터가 입력으로 사용될 때만 명시) 입력: K_I (키 유도 키), Label, Context, L 출력: 키 요소 K_O

```
1:  n ← ⌈L/h⌉
2:  if (n > (2^32 − 1)) then
3:      return ERROR_FLAG
4:  end if
5:  result(0) ← ∅
6:  A(0) ← Label || 0x00 || Context || [L]2
7:  for i = 1 to n do
8:      A(i) ← HMAC(KI, A(i − 1))
9:      K(i) ← HMAC(KI, A(i) {|| [i]2} || Label || 0x00 || Context || [L]2)
10:     result(i) ← result(i − 1) || K(i)
11: end do
12: KO ← leftmost(result(n), L)
13: return KO
```

6 안전성 고려사항

6.1 암호학적 안전성

키 유도 키(KDK) K_I가 키 유도 함수의 유일한 비밀 요소라는 가정하에, 키 유도 함수의 안전성 수준(security strength)은 키 유도 함수의 출력과 같은 길이의 균등하게 분포된 비트열(truly uniformly distributed bit string of the same length)을 구별하는데 필요한 작업의 양으로 측정될 수 있다. 이러한 작업의 양은 주어진 키 유도 함수 출력의 부분(segment)으로부터 K_I 또는 유도된 키 요소의 다른 부분을 찾는 작업의 양을 넘지 않는다. 키 유도 키 K_I를 제외한 키 유도 함수의 입력 데이터와 대응하는 출력 데이터를 이용할 수 있을 때, 모든 가능한 K_I 후보에 대한 전수 조사(= 2^w 번의 키 유도 함수 실행, $w = |K_I|$)를 통해 K_I를 찾을 수 있다.

6.2 키 유도 키의 길이

HMAC은 임의 길이의 키 유도 키를 사용할 수 있다. 키 유도 키의 길이가 기반 해시 함수의 블록 길이보다 클 경우, HMAC의 동작 절차에 의해 키 유도 키는 h 비트로 압축되기 때문에 2^h 번의 HMAC 계산을 통해 해시된 키 유도 키를 찾을 수 있다. 따라서 키 유도 키가 일정 크기를 넘어서더라도 키 유도 함수의 안전성 수준은 높아지지 않는다.

6.3 키 요소의 암호 키 변환

키 유도 함수를 통해 유도되는 암호 키의 길이는 해당 암호 키를 사용하는 암호 알고리즘과 안전성 수준에 의해 결정된다. 키 유도 함수를 사용하는 응용으로부터 부여되는 제한사항이 없는 경우, 다음의 제약사항에 따라 유도된 키 요소에서 요구되는 길이를 가지는 어느 부분(segment)이나 암호 키로 설정될 수 있다.

- 키 요소로부터 여러 개의 암호 키를 확보해야 할 때는 키 요소에서 서로 겹치지 않는 부분들(non-overlapping segments)이 선택되어야 한다.

따라서 키 요소의 비트 길이 L은 요구되는 암호 키 길이의 총합 이상이어야 한다.

6.4 입력 데이터 변환

키 유도 함수의 입력 데이터는 Label, Context, 키 요소의 길이 등의 요소로 구성된다. 그리고 이러한 각각의 입력 데이터 구성 요소는 비트열로 변환된다. 이때 사용되는 변환 방법은 모든 가능한 입력 정보의 집합과 비트열의 집합 사이에 일대일 대응 관계를 정의할 수 있어야 한다. 그리고 다른 입력 데이터 구성 요소는 특정 순서로 조립되어야 한다.

변환 방법은 결합된 입력 정보를 고유한 이진 문자열로 모호하지 않게 변환하도록 설계되어야 한다.

6.5 키 분리

키 요소로부터 유도되는 암호 키의 일부가 노출되더라도 다른 암호 키의 안전성 수준을 저하시키지 않아야 한다. 이러한 의미의 키 분리(key separation)는 다음의 두 가지 상황에 대해 다른 접근 방식으로 달성될 수 있다.

1. 하나의 키 요소에서 여러 개의 암호 키를 유도하는 경우, 키 요소에서 서로 겹치지 않는 부분들(non-overlapping segments)을 개별 암호 키로 설정한다.
2. 하나의 키 유도 키를 특정 키 유도 함수에 적용하여 생성되는 여러 개의 키 요소 사이의 암호학적인 독립성을 보장하기 위해, 각각의 키 유도 함수에 다른 데이터를 입력한다. 입력 데이터의 차이를 보장할 수 있는 정보의 예는 다음과 같다.
   - Label: 다른 목적을 위해 키 요소를 생성하는 경우
   - Context: 키 요소를 공유하는 개체가 다르거나 키 유도 함수를 호출하는 응용이 논스와 같은 구분 정보를 사용하는 경우
   - 세션 식별값: 개별 세션에 독립적으로 사용할 키 요소를 생성하는 경우

6.6 Context 결속

키 유도 함수로부터 생성되는 키 요소는 입력 데이터에 포함된 정보에 결속되어야 한다. 키 요소의 생성과 관련된 정보(생성에 참여한 개체 또는 생성된 키 요소를 사용하는 개체의 식별자(ID), 그리고 선택적으로 키 유도 과정에서 사용되는 키 공유 논스 등을 포함)는 입력 데이터의 Context에 포함된다.

Context 결속(context binding)은 유도된 암호 키를 사용하는 응용의 안전성 수준을 높이는 것은 아니지만, 암호 키 유도에 관련된 모든 개체가 누가, 언제, 어떻게 암호 키를 사용하는지에 대해 공유하고 있음을 보장함으로써 프로토콜의 오작동을 감지하는 방법을 제시할 수는 있다.

부 록 Ⅰ-1

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

지식재산권 확약서 정보

해당 사항 없음

※ 본 표준이 발간된 후 접수된 확약서가 있을 수 있으니, TTA 웹사이트에서 확인하시기 바랍니다.

부 록 Ⅰ-2

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

시험인증 관련 사항

Ⅰ-2.1 시험인증 대상 여부

해당 사항 없음

Ⅰ-2.2 시험표준 제정 현황

해당 사항 없음

부 록 Ⅰ-3

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

본 표준의 연계(family) 표준

Ⅰ-3.1 TTAK.KO-12.0273-Part2

이 표준에서 제시하는 키 유도 함수의 기반 해시 함수로 SHA-2[2]를 사용할 경우의 참조 구현값을 제시함

Ⅰ-3.2 TTAK.KO-12.0273-Part3

이 표준에서 제시하는 키 유도 함수의 기반 해시 함수로 LSH[1]를 사용할 경우의 참조 구현값을 제시함

Ⅰ-3.3 TTAK.KO-12.0274-Part1

패스워드(또는 패스프레이즈)를 입력으로 하여 암호 키를 생성하는 키 유도 함수의 상세 규격을 제시함

Ⅰ-3.4 TTAK.KO-12.0272

블록 암호 기반 메시지 인증 코드인 CMAC을 의사 난수 함수로 사용하는 세 가지 키 유도 함수의 상세 규격을 정의하고, 국내 개발 주요 블록 암호 알고리즘(SEED, ARIA, HIGHT, LEA)을 CMAC의 기반 함수로 사용하는 경우의 참조 구현값을 제시함

부 록 Ⅰ-4

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

참고 문헌

[1] TTA TTAK.KO-12.0276, “해시 함수 LSH”.

[2] NIST FIPS PUB 180-4, “Secure Hash Standard (SHS)”, 2015. 8.

[3] NIST FIPS PUB 198-1, “The Keyed-Hash Message Authentication Code (HMAC)”, 2008. 7.

[4] NIST SP 800-56A, “Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography (Revision 3)”, 2018. 4.

[5] NIST SP 800-56C, “Recommendation for Key-Derivation Methods in Key-Establishment Scheme (Revision 1)”, 2018. 4.

[6] NIST SP 800-57 Part 1, “Recommendation for Key Management - Part 1: General (Revision 4)”, 2016. 1.

[7] Y. Dodis, R. Gennaro, J. Hastad, H. Krawczyk, and T. Rabin. Randomness extraction and key derivation using the CBC, Cascade, and HMAC modes. Advances in Cryptology - CRYPTO 2004. Lecture Notes in Computer Science, vol. 3152, p. 494-510. Springer-Verlag, 2004.

부 록 Ⅰ-5

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

영문표준 해설서

해당 사항 없음

부 록 Ⅰ-6

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

표준의 이력

판수	채택일	표준번호	내용	담당 위원회
제1판	2015.12.16	제정 TTAK.KO-12.0273	-	정보보호기반 PG (PG501)
제2판	2018.12.xx	개정 TTAK.KO-12.0273-Part1	개별 해시 함수를 이용한 참조 구현값을 연계 표준으로 분리	정보보호기반 PG (PG501)