(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2011년07월14일
(11) 등록번호 10-1048661
(24) 등록일자 2011년07월06일

(51) Int. Cl.
*H04K 1/00* (2006.01)
(21) 출원번호 10-2009-0006430
(22) 출원일자 2009년01월28일
심사청구일자 2009년01월28일
(65) 공개번호 10-2010-0087437
(43) 공개일자 2010년08월05일
(56) 선행기술조사문헌
KR1020020040103 A
KR1020050059346 A
US6154542 B
전체 청구항 수 : 총 11 항

(73) 특허권자
**한국과학기술원**
대전 유성구 구성동 373-1
(72) 발명자
**박기웅**
서울 노원구 월계4동 500-11 8/5
**박규호**
대전 유성구 구성동 한국과학기술원 6-3208
(74) 대리인
**김성호**

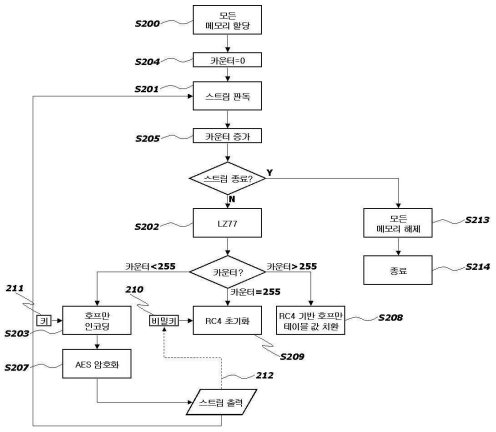
심사관 : 김현진

# (54) 데이터에 대한 압축 및 암호화 연산을 위한 방법, 장치 및 컴퓨터 판독 가능한 기록 매체

## (57) 요 약

데이터에 대한 압축 및 암호화 연산을 위한 방법, 장치 및 컴퓨터 판독 가능한 기록 매체가 개시된다. 본 발명에 따르면, 압축 연산 및 암호화 연산에 필요한 모든 메모리를 할당하고 이를 초기화하는 단계, 초기 출력 데이터에 대해 바이트 단위로 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘을 수행하는 단계, 초기 출력 데이터에 대한 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘이 완료된 후, 압축 알고리즘 및 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 수행하는 단계, 및 상기 메모리를 해제하는 단계를 포함하는 압축 및 암호화 연산 방법이 제공된다. 본 발명에 의하면, 압축 알고리즘 및 암호화 알고리즘이 결합되어 일괄적으로 수행됨으로써, 중복 연산이 제거되고 메모리 할당/해제에 따른 연산 오버헤드가 제거됨에 따라 효율적인 압축 및 암호화 연산이 수행될 수 있게 된다.

*대 표 도* - 도2a



## *특허청구의 범위*

**청구항 1**

압축 연산 및 암호화 연산에 필요한 모든 메모리를 할당하고 이를 초기화하는 단계,

초기 출력 데이터에 대해 바이트 단위로 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘을 수행하는 단계,

초기 출력 데이터에 대한 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘이 완료된 후, 압축 알고리즘 및 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 수행하는 단계, 및

상기 메모리를 해제하는 단계

를 포함하는 압축 및 암호화 연산 방법.

**청구항 2**

제1항에 있어서,

상기 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 수행하는 단계는,

(a) 상기 RC4 암호화 알고리즘 수행에 필요한 난수 열을 발생시키기 위한 비밀키 값을 초기화하는 단계,

(b) 상기 난수 열을 발생시키기 위해 RC4 암호화 알고리즘을 초기화하는 단계, 및

(c) 상기 난수 열을 발생시키고 이를 이용하여 상기 호프만 인코딩 알고리즘에 사용되는 내부 테이블의 값을 무작위적으로 치환하는 단계를 포함하는 압축 및 암호화 연산 방법.

**청구항 3**

제2항에 있어서,

상기 (a) 단계는,

상기 AES 암호화 알고리즘에 의해 암호화된 결과값을 이용하여 상기 비밀키 값을 초기화하는 단계를 포함하는 압축 및 암호화 연산 방법.

**청구항 4**

제2항에 있어서,

상기 (c) 단계는,

상기 발생된 난수 열과 상기 난수 열의 길이와 동일한 길이를 갖는 접두 부호 코드의 개수에 대해 모듈로(mod) 연산을 수행하는 단계,

상기 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산의 대상이 되는 입력 데이터에 따른 인코딩 데이터로부터 상기 모듈로 연산의 결과 값만큼 건너 뛴 위치의 데이터에 포함되는 접두 부호를 출력하는 단계,

상기 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산의 대상이 되는 입력 데이터에 따른 인코딩 데이터의 접미 부호, 및 상기 접미 부호와 동일한 길이를 갖는 난수 열에 대해 XOR 연산을 수행하여 그 결과 값을 출력하는 단계

를 포함하는 압축 및 암호화 연산 방법.

**청구항 5**

제1항에 있어서,

상기 압축 알고리즘은 LZ77 압축 알고리즘인 압축 및 암호화 연산 방법.

**청구항 6**

제1항에 있어서,

상기 압축 알고리즘은 BWT 변환 연산과 MTF 변환 연산이 순차적으로 수행되는 알고리즘인 압축 및 암호화 연산 방법.

**청구항 7**

호프만 인코딩 알고리즘과 RC4 암호화 알고리즘을 결합하여 수행하기 위한 방법으로서,

난수 발생기로부터 발생된 난수 열과 상기 난수 열의 길이와 동일한 길이를 갖는 접두 부호 코드의 개수에 대해 모듈로(mod) 연산을 수행하는 단계,

입력 데이터에 따른 인코딩 데이터로부터 상기 모듈로 연산의 결과 값만큼 건너 뛴 위치의 데이터에 포함되는 접두 부호를 출력하는 단계,

상기 입력 데이터에 따른 인코딩 데이터의 접미 부호, 및 상기 접미 부호와 동일한 길이를 갖는 난수 열에 대해 XOR 연산을 수행하여 그 결과 값을 출력하는 단계

를 포함하는 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 수행 방법.

**청구항 8**

압축 연산 및 암호화 연산에 필요한 모든 메모리를 할당, 초기화, 해제 및 관리하는 메모리 관리 모듈, 및

초기 출력 데이터에 대한 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘이 완료된 후, 압축 알고리즘 및 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 수행하는 호프만 인코딩/RC4 결합 모듈

을 포함하는 압축 및 암호화 연산 장치.

**청구항 9**

제8항에 있어서,

상기 메모리 관리 모듈은,

상기 압축 알고리즘에 사용되는 해쉬 테이블을 저장하기 위한 메모리를 가리키고 있는 제1 포인터,

상기 호프만 인코딩 알고리즘을 수행하기 위한 두 개의 트리를 가리키고 있는 제2 포인터,

상기 RC4 암호화 알고리즘을 수행하기 위한 키에 대한 값을 가리키고 있는 제3 포인터,

상기 압축 및 암호화 연산의 출력 값을 저장하기 위한 버퍼를 가리키고 있는 제4 포인터,

이전 압축 연산이 처리되고 있는 위치를 가리키기 위한 제5 포인터,

이전 암호화 연산이 처리되고 있는 위치를 가리키기 위한 제6 포인터,

상기 압축 알고리즘의 입력 버퍼를 가리키고 있는 제7 포인터, 및

상기 입력 버퍼에서 이전까지 압축 및 암호화가 처리된 위치를 가리키기 위한 제8 포인터

를 저장하는 디렉토리를 포함하는 압축 및 암호화 연산 장치.

**청구항 10**

제8항에 있어서,

상기 호프만 인코딩/RC4 결합 모듈은,

난수 열을 발생시키는 난수 열 발생기,

상기 난수 열 발생기로부터의 발생된 난수 열과 상기 난수 열의 길이와 동일한 길이를 갖는 접두 부호 코드의 개수에 대해 모듈로(mod) 연산을 수행하는 모듈로 연산부,

상기 호프만 인코딩/RC4 결합 모듈의 입력 데이터에 따른 인코딩 데이터로부터 상기 모듈로 연산의 결과 값만큼 건너 뛴 위치의 데이터에 포함되는 접두 부호를 출력하는 접두 부호 출력부,

상기 호프만 인코딩/RC4 결합 모듈의 입력 데이터에 따른 인코딩 데이터의 접미 부호, 및 상기 접미 부호와 동

일한 길이를 갖는 난수 열에 대해 XOR 연산을 수행하는 XOR 연산부,

상기 XOR 연산부의 출력값을 접미 부호로서 출력하는 접미 부호 출력부

를 포함하는 압축 및 암호화 연산 장치.

**청구항 11**

제1항 내지 제7항 중 어느 한 항에 따른 방법을 실행하기 위한 컴퓨터 프로그램을 기록하는 컴퓨터 판독 가능한 기록 매체.

# 명 세 서

## 발명의 상세한 설명

### 기 술 분 야

[0001] 본 발명은 데이터에 대한 압축 및 암호화 연산을 위한 방법, 장치 및 컴퓨터 판독 가능한 기록 매체에 관한 것으로, 보다 상세하게는, 압축 알고리즘과 암호화 알고리즘을 결합하여 수행함으로써 치환 연산의 중복성을 제거함과 동시에 메모리 할당 및 해제와 관련된 연산 수를 줄여 높은 연산 효율성을 제공할 수 있는 압축 및 암호화 연산을 위한 방법, 장치 및 컴퓨터 판독 가능한 기록 매체에 관한 것이다.

### 배 경 기 술

[0002] 최근 데이터 또는 정보의 홍수로 인해 데이터 처리 기술에 대한 중요성이 날로 커지고 있다.

[0003] 통신 및 자료 저장에 널리 사용되는 데이터 압축이란 소정 자료를 표현하는 데 사용된 비트 수를 줄이는 과정을 의미하는 것으로서, 정보를 관리하고 다루는 다양한 방법을 처리하는 수학 분야인 정보 이론의 가장 중요한 결과물 중의 하나이다. 자료 압축은 두 가지 과정을 수반하는데 그 중 하나는 크기를 줄이기 위해 자료가 압축 또는 부호화되는 과정이며, 다른 하나는 원래의 상태로 복원 또는 해독되는 과정이다.

[0004] 자료 압축이 가능한 이유는 모든 자료들이 엔트로피라는 정보화된 내용으로 표현될 수 있기 때문이다. 대부분의 자료가 데이터의 엔트로피가 최적이라고 암시하는 비트 수보다 더 많은 비트 수를 사용하여 표현되므로 압축이 가능한 것이다. 자료 압축 방법은 두 종류로 나뉘는데, 손실이 있는 것과 손실이 없는 것이다. 이 중 무손실 압축 알고리즘으로는 여러 가지 종류가 있으나, 현재 많이 사용되는 압축 알고리즘으로는 gzip 압축알고리즘과 bzip 압축 알고리즘이 있다. gzip 압축 알고리즘은 LZ77 압축 알고리즘과 호프만 인코딩을 결합한 압축알고리즘이고, bzip 압축 알고리즘은 BWT와 MTF 변환 후에 호프만 인코딩을 결합한 압축알고리즘이며, 파일의 압축 및 통신에 널리 적용되고 있다.

[0005] 한편, 데이터 암호화란 소정의 정보를 의미를 알 수 없는 형식(암호문)으로 변환하는 것을 의미하는 것으로서 암호문의 형태로 정보를 기억 장치에 저장하거나 통신 회선을 통해 전송함으로써 정보를 안전하게 보호할 수 있는 수단을 제공한다. 암호화는 암호 키(특정의 비트열)를 사용하여 정보를 암호문으로 변환하는 것이고, 복호화는 복호 키를 사용하여 암호화된 정보를 원래의 정보로 복원하는 것이다. 복호 키를 갖고 있지 않은 사람은 정보를 올바르게 복원할 수 없으므로, 복호 키가 제3자에게 알려지지 않는다면 암호화된 정보는 보호될 수 있게 된다. 암호 체계는 크게 대칭 암호 방식과 공개 키 암호 방식으로 분류된다. 이 중, 대칭 암호 방식은 암호화와 복호화에 동일한 키가 사용되고, 통신할 때에는 송신자와 수신자가 사전에 동일한 키를 비밀로 갖고 있을 필요가 있는 방식이다. 한편, 공개 키 암호 방식은 암호화와 복호화에 서로 다른 키를 사용하는데 암호 키는 공개하고 복호 키는 비밀로 하는 방식이다. 일반적으로 공개 키 암호 방식은 연산량이 대칭 암호 방식에 비하여 매우 크기 때문에 인증을 하는데 활용되고, 안전한 통신을 하는데 있어서는 대칭 암호 방식이을 보다 많이 활용되게 된다.

[0006] 대칭 암호 알고리즘은 다시 블록 암호 알고리즘과 스트림 암호 알고리즘으로 나뉜다. 이 중, 블록 암호 알고리즘은 블록 단위로 암호화를 수행하는 알고리즘으로서 여러 블록 암호 알고리즘 중 AES압축 알고리즘이 국제 표준으로 채택이 되었다. 한편, 스트림 암호 알고리즘은 랜덤 수열 발생기로 난수 열을 발생 시키고 평문을 일련의 비트열로 취급하여 한 번에 1bit씩(때로는 byte 단위로) XOR 연산을 행하여 암호화시키는 암호 시스템을 말하는 것으로서 스트림 암호 알고리즘은 블록 암호화 알고리즘보다 속도가 빠른 특징을 가지고 있다. 스트림 암호 알고리즘으로서는 여러 가지 방식 중 RC4 암호화 알고리즘이 널리 사용되고 있다.

[0007] 도 1은 종래 기술에 따른 데이터 압축 및 암호화 연산의 기본 흐름도를 나타내는 도면이다.

[0008] 먼저, 전송 또는 저장할 데이터가 발생하게 되면 압축을 위한 버퍼 및 치환 연산을 위한 테이블 생성을 위한 메모리 할당 연산이 수행된다(S100). 그 후, 정보의 입력 스트림으로부터 데이터를 읽어낸다(S101). 그 후, gzip 알고리즘을 수행하여 압축을 한다(S102). gzip 알고리즘은 RFC1897에 정의되어 있는 상용화된 압축 알고리즘으로서 LZ77 알고리즘을 수행(S102a)한 결과 값에 대해 호프만 인코딩 방식으로 압축을 행함으로써 이루어진다(S102b). 한 번의 gzip 알고리즘이 종료되면 그 결과 값, 즉, 압축의 결과를 버퍼로 출력한다(S103). 이와 같은 과정을 스트림이 모두 끝날 때까지 반복하여 수행한다. 과정 중 스트림이 모두 끝났는지를 판단한 후(S105), 스트림이 모두 끝난 것으로 판단되면, 메모리 할당을 했던 버퍼와 테이블에 대한 메모리를 해제한다(S109). 이것으로서 암호화 연산이 시작된다. 일반적으로 암호화 연산을 하는데 있어서는 대칭키 방식인 AES 또는 RC4 암호화 방식을 사용하나 도 1에서는 AES를 사용한다고 가정하였다. AES는 미국 정부 표준으로 지정된 블록 암호 형식으로서, 이전의 DES를 대체하며, 미국 표준 기술 연구소(NIST)가 5년의 표준화 과정을 거쳐 2001년 11월 26일에 연방 정보 처리 표준(FIPS 197)으로 발표하였다. 2002년 5월 26일부터 표준으로 효력을 발휘하기 시작했다. AES 알고리즘은 입력된 키를 확장시키는 키 확장 연산, 사용자로부터 입력한 데이터의 은닉을 위한 XOR 연산 및 SBOX를 이용한 치환연산, 쉬프트로우(Shift Rows), 믹스 컬럼(Mix Column)연산을 수행하며 이를 10회 반복적으로 수행하여 데이터 은닉을 수행하게 된다. 이러한 연산을 수행하기 위해 먼저 버퍼로부터 데이터를 읽어와(S104), SBOX 및 버퍼를 위한 메모리 할당 연산이 수행되어야 하며(S107), 상기 암호화 연산은 16Byte 단위로 수행되기 때문에 그 용량에 따라서 반복적인 연산이 수행되어야 한다. 즉, 버퍼가 끝났는지 여부를 판단하고(S108), 용량에 따른 버퍼가 끝날 때까지 연산을 반복적으로 수행하여야 한다. 이 과정 중에는 암호화 연산(S106) 및 메모리 할당 연산(S107)이 수행되게 되는데 이에 따라 연산 오버헤드가 계속적으로 가중된다. 또한, 상기 압축 알고리즘(S102) 및 암호화 알고리즘(S106)은 모두 수학적 연산(S102, S106) 및 메모리 할당/해제 연산(S100, S109, S107, S110)의 반복을 야기하기 때문에 시스템 오버헤드 역시 가중되어 시스템의 성능 저하를 야기시킬 수 있다는 문제점을 가지고 있다. 이러한 문제점을 해결하기 위해서는 보다 높은 성능을 갖는 프로세서를 갖추어야하는데 이에 따르면 비용적인 측면에서 매우 비효율적이게 된다.

[0009] 이처럼, 압축 알고리즘을 이용하게 되면 자료의 크기를 줄일 수 있어 효율적인 자료의 저장 및 통신을 할 수 있게 되고, 암호화 알고리즘을 이용하게 되면 자료의 안전한 저장 및 통신이 가능해지나, 이를 위해서는 많은 수학적 연산이 수행되어야 하며, 이 과정에서 이루어지는 중복된 치환연산 및 메모리 연산으로 인해 시스템의 성능 저하가 발생된다는 문제점이 있었다.

## *발명의 내용*

### *해결 하고자하는 과제*

[0010] 본 발명은 상술한 종래 기술의 문제점을 해결하는 것을 목적으로 한다.

[0011] 또한, 본 발명은 데이터의 압축 알고리즘과 암호화 알고리즘을 결합하여 동시에 수행함으로써 중복되는 치환연산이 제거됨에 따른 높은 연산 효율성을 제공하는 것을 그 목적으로 한다.

[0012] 한편, 본 발명의 다른 목적은 압축 알고리즘과 암호화 알고리즘을 결합하여 수행함에 있어서, 메모리의 할당, 초기화 및 해제와 관련된 연산을 일괄적으로 처리 및 관리 함으로써, 각 알고리즘의 시작과 종료 전후의 불필요한 메모리 할당 및 해제의 필요를 없애고, 불필요한 메모리간 복사 연산 또한 제거하며, 불필요한 반복적인 연산을 생략하여 단위시간당 압축 및 암호화의 처리량을 높일 수 있도록 하는 것이다.

### *과제 해결수단*

[0013] 상술한 목적을 달성하기 위한 본 발명의 일 실시예에 따르면, 압축 연산 및 암호화 연산에 필요한 모든 메모리를 할당하고 이를 초기화하는 단계, 초기 출력 데이터에 대해 바이트 단위로 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘을 수행하는 단계, 초기 출력 데이터에 대한 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘이 완료된 후, 압축 알고리즘 및 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 수행하는 단계, 및 상기 메모리를 해제하는 단계를 포함하는 압축 및 암호화 연산 방법이 제공된다.

[0014] 여기서, 상기 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 수행하는 단계는, (a) 상기 RC4 암호화 알고리즘 수행에 필요한 난수 열을 발생시키기 위한 비밀키 값을 초기화하는 단계, (b) 상기 난수 열을 발

생시키기 위해 RC4 암호화 알고리즘을 초기화하는 단계, 및 (c) 상기 난수 열을 발생시키고 이를 이용하여 상기 호프만 인코딩 알고리즘에 사용되는 내부 테이블의 값을 무작위적으로 치환하는 단계를 포함할 수도 있다.

[0015] 여기서, 상기 (a) 단계는, 상기 AES 암호화 알고리즘에 의해 암호화된 결과값을 이용하여 상기 비밀키 값을 초기화하는 단계를 포함할 수도 있다.

[0016] 여기서, 상기 (c) 단계는, 상기 발생된 난수 열과 상기 난수 열의 길이와 동일한 길이를 갖는 접두 부호 코드의 개수에 대해 모듈로(mod) 연산을 수행하는 단계, 상기 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산의 대상이 되는 입력 데이터에 따른 인코딩 데이터로부터 상기 모듈로 연산의 결과 값만큼 건너 뛴 위치의 데이터에 포함되는 접두 부호를 출력하는 단계, 상기 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산의 대상이 되는 입력 데이터에 따른 인코딩 데이터의 접미 부호, 및 상기 접미 부호와 동일한 길이를 갖는 난수 열에 대해 XOR 연산을 수행하여 그 결과 값을 출력하는 단계를 포함할 수도 있다.

[0017] 여기서, 상기 압축 알고리즘은 LZ77 압축 알고리즘일 수도 있다.

[0018] 여기서, 상기 압축 알고리즘은 BWT 변환 연산과 MTF 변환 연산이 순차적으로 수행되는 알고리즘일 수도 있다.

[0019] 본 발명의 다른 일 실시예에 따르면, 호프만 인코딩 알고리즘과 RC4 암호화 알고리즘을 결합하여 수행하기 위한 방법으로서, 난수 발생기로부터 발생된 난수 열과 상기 난수 열의 길이와 동일한 길이를 갖는 접두 부호 코드의 개수에 대해 모듈로(mod) 연산을 수행하는 단계, 입력 데이터에 따른 인코딩 데이터로부터 상기 모듈로 연산의 결과 값만큼 건너 뛴 위치의 데이터에 포함되는 접두 부호를 출력하는 단계, 상기 입력 데이터에 따른 인코딩 데이터의 접미 부호, 및 상기 접미 부호와 동일한 길이를 갖는 난수 열에 대해 XOR 연산을 수행하여 그 결과 값을 출력하는 단계를 포함하는 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 수행 방법이 제공된다.

[0020] 본 발명의 또 다른 일 실시예에 따르면, 압축 연산 및 암호화 연산에 필요한 모든 메모리를 할당, 초기화, 해제 및 관리하는 메모리 관리 모듈, 및 초기 출력 데이터에 대한 압축 알고리즘, 호프만 인코딩 알고리즘 및 AES 암호화 알고리즘이 완료된 후, 압축 알고리즘 및 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 수행하는 호프만 인코딩/RC4 결합 모듈을 포함하는 압축 및 암호화 연산 장치가 제공된다.

[0021] 여기서, 상기 메모리 관리 모듈은, 상기 압축 알고리즘에 사용되는 해쉬 테이블을 저장하기 위한 메모리를 가리키고 있는 제1 포인터, 상기 호프만 인코딩 알고리즘을 수행하기 위한 두 개의 트리를 가리키고 있는 제2 포인터, 상기 RC4 암호화 알고리즘을 수행하기 위한 키에 대한 값을 가리키고 있는 제3 포인터, 상기 압축 및 암호화 연산의 출력 값을 저장하기 위한 버퍼를 가리키고 있는 제4 포인터, 이전 압축 연산이 처리되고 있는 위치를 가리키기 위한 제5 포인터, 이전 암호화 연산이 처리되고 있는 위치를 가리키기 위한 제6 포인터, 상기 압축 알고리즘의 입력 버퍼를 가리키고 있는 제7 포인터, 및 상기 입력 버퍼에서 이전까지 압축 및 암호화가 처리된 위치를 가리키기 위한 제8 포인터를 저장하는 디렉토리를 포함할 수도 있다.

[0022] 여기서, 상기 호프만 인코딩/RC4 결합 모듈은, 난수 열을 발생시키는 난수 열 발생기, 상기 난수 열 발생기로부터 발생된 난수 열과 상기 난수 열의 길이와 동일한 길이를 갖는 접두 부호 코드의 개수에 대해 모듈로(mod) 연산을 수행하는 모듈로 연산부, 상기 호프만 인코딩/RC4 결합 모듈의 입력 데이터에 따른 인코딩 데이터로부터 상기 모듈로 연산의 결과 값만큼 건너 뛴 위치의 데이터에 포함되는 접두 부호를 출력하는 접두 부호 출력부, 상기 호프만 인코딩/RC4 결합 모듈의 입력 데이터에 따른 인코딩 데이터의 접미 부호, 및 상기 접미 부호와 동일한 길이를 갖는 난수 열에 대해 XOR 연산을 수행하는 XOR 연산부, 상기 XOR 연산부의 출력값을 접미 부호로서 출력하는 접미 부호 출력부를 포함할 수도 있다.

[0023] 그리고, 본 발명의 또 다른 실시예에 따르면, 압축 알고리즘과 암호화 알고리즘을 결합하여 효율적인 연산 방법을 제공하기 위한 다른 방법 및 이러한 방법을 실행하기 위한 컴퓨터 프로그램을 기록하는 컴퓨터 판독 가능한 기록 매체가 제공될 수 있다.

## 효 과

[0024] 본 발명에 따르면, 데이터의 압축 알고리즘과 암호화 알고리즘이 결합되어 동시에 수행되기 때문에, 중복되는 치환 연산이 제거될 수 있으며, 이에 따라 높은 연산 효율성을 달성할 수 있다.

[0025] 한편, 본 발명에 따르면, 압축 알고리즘과 암호화 알고리즘을 결합하여 수행함에 있어서, 메모리의 할당, 초기화 및 해제와 관련된 연산이 일괄적으로 처리 및 관리되기 때문에, 각 알고리즘의 시작과 종료 전후의 불필요한 메모리 할당 및 해제의 필요가 없어지고, 불필요한 메모리간 복사 연산 또한 제거되며, 불필요한 반복적인 연산

이 생략되어 단위시간당 압축 및 암호화의 처리량이 높아질 수 있다.

## *발명의 실시를 위한 구체적인 내용*

[0026] 후술하는 본 발명에 대한 상세한 설명은, 본 발명이 실시될 수 있는 특정 실시예를 예시로서 도시하는 첨부 도면을 참조한다. 이들 실시예는 당업자가 본 발명을 실시할 수 있기에 충분하도록 상세히 설명된다. 본 발명의 다양한 실시예는 서로 다르지만 상호 배타적일 필요는 없음이 이해되어야 한다. 예를 들어, 여기에 기재되어 있는 특정 형상, 구조 및 특성은 일 실시예에 관련하여 본 발명의 정신 및 범위를 벗어나지 않으면서 다른 실시예로 구현될 수 있다. 또한, 각각의 개시된 실시예 내의 개별 구성요소의 위치 또는 배치는 본 발명의 정신 및 범위를 벗어나지 않으면서 변경될 수 있음이 이해되어야 한다. 따라서, 후술하는 상세한 설명은 한정적인 의미로서 취하려는 것이 아니며, 본 발명의 범위는, 적절하게 설명된다면, 그 청구항들이 주장하는 것과 균등한 모든 범위와 더불어 첨부된 청구항에 의해서만 한정된다. 도면에서 유사한 참조부호는 여러 측면에 걸쳐서 동일하거나 유사한 기능을 지칭한다.

[0027] 이하, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 본 발명을 용이하게 실시할 수 있도록 하기 위하여, 본 발명의 바람직한 실시예들에 관하여 첨부된 도면을 참조하여 상세히 설명하기로 한다.

[0028] 압축 및 암호화 연산의 결합

[0029] 도 2a 는 본 발명의 일 실시예에 따라 압축 연산 및 암호화 연산이 결합된 전체 연산의 흐름도를 나타내는 도면이다. 여기서, 초기 데이터의 출력은 255 byte인 것으로 가정된다.

[0030] 도 2a를 참조하면, 본 발명에 따른 압축 연산 및 암호화 연산 방법은 압축 및 암호화에 필요한 메모리를 각 알고리즘에서 할당 및 해제하지 않고, 압축 및 암호화 연산에 필요한 모든 메모리의 할당 및 초기화가 단계 S200에서 한꺼번에 수행되게 된다. 또한, 이는 압축 및 암호화 연산 메모리 관리 모듈에서 관리된다.

[0031] 전송 또는 저장할 데이터의 압축 및 암호화에 필요한 메모리의 할당 및 초기화가 이루어지게 되면(S200), 초기값 0으로 설정되는 내부 카운터를 할당하고(S204), 입력 스트림으로부터 데이터를 읽어온다(S201). 내부 카운터는 압축 및 암호화 연산이 수행된 횟수, 즉, 압축 및 암호화가 이루어진 바이트 크기를 카운트하기 위한 것이다.

[0032] 다음으로, 카운터를 하나 증가시키고(S205), 압축 알고리즘을 수행한다(S202). 압축 알고리즘은 LZ77 알고리즘(S202)과 호프만 인코딩 방식(S203)이 결합된 통상적인 gzip 알고리즘이 사용될 수 있다. 이러한 압축 알고리즘은 초기 데이터 출력, 즉, 255 byte에 대해 수행되게 되는데, 이 과정에서 L7ZZ 알고리즘(S202) 및 호프만 인코딩 방식(S203)이 수행된 이후, AES 암호화 알고리즘(S207)이 수행된다. 이 과정에서 gzip 알고리즘 대신 bzip 알고리즘이 사용될 수도 있는데, 이 때에는, 도 2b에 도시되는 바와 같이, LZ77 압축 알고리즘(S202)이 BWT 변환 연산(S216)과 MTF 변환 연산(S217)으로 대치되게 된다. 본 발명에 따르면, 단계 S200에서 이미 압축 및 암호화 연산에 필요한 모든 메모리의 할당 및 초기화가 완료되었고 이는 압축 및 암호화 연산 메모리 관리 모듈에서 관리되므로 메모리 할당 및 해제에 관련된 연산이 필요없어지게 된다.

[0033] 한편, 255 byte에 대한 압축 알고리즘이 수행되었는지 여부는 압축 및 암호화가 이루어진 바이트 수를 카운트하는 카운터의 값으로부터 파악될 수 있다. 만약, 카운터의 값이 255 보다 크게 되면, 즉, 255 byte에 대한 압축 및 암호화 연산이 완료되면 이전과 같이 L7ZZ 알고리즘(S202)에 의해 압축이 수행되긴 하지만 호프만 인코딩 부분에 있어서 암호화 알고리즘인 RC4 와 결합되게 된다.

[0034] RC4 알고리즘은 RSA 데이터 시큐리티(Data Security)사의 론 리베스트(Ron Rivest)에 의해 개발된 스트림 암호 시스템으로 외부에서 제공된 키에 따라 임의 길이의 유사 난수 열을 발생하고 이를 평문과 XOR연산을 통하여 암호화를 수행하게 되는 알고리즘이다. RC4 알고리즘은 암호화 연산 수행에 있어서 AES 알고리즘과 비교하여 극히 적은 연산만을 요구한다.

[0035] RC4 암호화 연산을 하는데 있어서는 난수 열을 발생시키기 위한 RC4 초기화 단계(S209)가 수행된다. 원래는 호프만 인코딩의 결과값과 RC4의 난수 발생기로부터의 난수 열을 XOR연산시키게 되나, 본 발명에 있어서는 RC4의 난수 발생을 위한 비밀키 값(210)을 통상 사용되는 128bit의 키(211)로 입력하는 것이 아니라, AES 암호화 알고리즘을 통해 암호화된(S207) 255 byte(=2040bit)의 결과 값(212)을 이용하여 초기화 시키고, 난수 발생기로부터의 값을 이용하여 내부 호프만 테이블의 값을 무작위적으로 치환하는 연산을 호프만 인코딩/RC4 결합 모듈에서 수행하게 된다. 즉, 호프만 인코딩/RC4 결합 모듈에서 호프만 인코딩과 RC4 알고리즘이 결합된 연산이 수행된다(S208). 이에 따라, 원래의 암호화 안전성을 떨어뜨리지 않으면서도, 기존 방법에 비하여 연산적으로 매우

효율적인 방법으로 압축 연산과 암호화 연산이 동시에 수행될 수 있게 된다. 호프만 인코딩 방식과 RC4 알고리즘의 결합 방식에 대해서는 후에 도 3 및 도 4를 참조하여 상세히 설명하기로 한다.

[0036] 모든 입력 데이터의 압축 및 암호화 연산의 처리가 완료된 후에는 압축 및 암호화 연산 메모리 관리 모듈에서 메모리에 대한 해제를 수행한다(S213). 그 후, 모든 연산이 종료되게 된다(S214). 본 발명의 압축 및 암호화 알고리즘은 RC4 암호화 알고리즘과 호프만 인코딩 알고리즘을 결합함으로써, 중복되는 수학적 연산을 효율적으로 줄이게 된다.

[0037] 호프만 인코딩 알고리즘과 RC4 알고리즘의 결합

[0038] 이하에서는, 도 2a의 압축 및 암호화 연산 방법에 있어서, 단계 S208에서 수행되는 호프만 인코딩 알고리즘과 RC4 알고리즘이 결합되는 방식을 설명한다. 이를 위해 먼저 통상적인 호프만 인코딩 알고리즘과 RC4 알고리즘에 대해 설명하기로 한다.

[0039] 호프만 인코딩 알고리즘과 RC4 알고리즘

[0040] 도 3은 통상적인 방법에 따른 호프만 인코딩 알고리즘과 RC4 알고리즘을 설명하는 도면이다. 먼저, 도 3a는 통상적인 호프만 인코딩 알고리즘(300, 301)을 설명하기 위한 도면이고, 도 3b는 통상적인 RC4 알고리즘(302)을 설명하기 위한 도면이다.

[0041] 도 3a를 참조하면, 호프만 인코딩 알고리즘(300, 301)은 문자들의 빈도로부터 접두 부호(303; 어떤 한 문자에 대한 부호가 다른 부호들의 접두어가 되지 않는 부호)를 만들어 내는 알고리즘으로, 작은 빈도의 문자일수록 더 긴 부호를 쓰고 높은 빈도의 문자일수록 더 짧은 부호를 쓴다. 호프만 인코딩 알고리즘(300, 301)은 주어진 빈도에 대해서 항상 최적의 접두 부호(303)를 만들어 내며, 호프만 인코딩 알고리즘을 위한 부호화 테이블은 문자(304)와 매칭 길이(305)에 대한 인코딩을 위한 테이블 및 거리(306)를 표현하기 위한 테이블로 구성될 수 있다. 전술하였던 gzip 알고리즘은 LZ77 알고리즘과 호프만 인코딩 알고리즘으로 이루어지는데 LZ77 알고리즘의 출력은 문자(304), 매칭 길이(305) 및 거리(306)로 나뉘어 지며, "문자(304)와 매칭길이(305)"를 표현하는 호프만 트리(300), 거리(306)를 표현하는 호프만 트리(301) 2 개로 구성될 수 있다. 이 때, 데이터 255 byte의 값은 문자(304)에 매칭이 되며, 길이에 대한 인코딩은 길이(305)에 매칭되어 표현된다. 각 테이블의 값(307)은 인코딩이 되는 값을 의미하며, 각각의 엔트리에는 접미 부호(308)가 붙어있어 같은 접두 부호(303)를 갖고 있더라도 여러 문자로 인코딩이 될 수 있는 것이다. 예를 들어, 매칭 길이(305) 부분의 세 번째 열(309)은 길이 "33" 내지 길이 "41" 까지 해당되고 접미 부호 3 bit(310)에 의하여 길이 "33" 내지 길이 "41" 중 하나가 선택된다고 가정하는 경우, 매칭 길이에 대한 정보 "35"가 입력되면 접두 부호는 "0000011"(309)이 되고 해당 열은 3 bit의 접미 부호(310)에 의해 8개의 길이 정보로 표현되기 때문에 해당하는 접미부호 "010"(310)가 입력이 되어 인코딩이 완성된다. 또한, 같은 원리로 거리에 대한 호프만 트리(301)에서도 네 번째 열(311)이 길이 "250" 내지 길이 "257" 까지 해당되고 접미 부호 3 bit(312)에 의하여 길이 "250" 내지 길이 "257" 중 하나가 선택된다고 가정하면, 거리에 대한 정보 "250"이 입력될 때 이는 네번째 열(311)에 해당되고 해당 열은 3 bit의 접미 부호(312)로 구성되며, "000"로 인코딩 되는 것이다. 호프만 트리를 생성하는 데에 있어서는 문자의 출현 빈도에 따라서 유동적인 비트길이를 할당하게 되므로, 적게 나오는 문자일수록 더 긴 부호를 쓰고 많이 나오는 문자일수록 더 짧은 부호를 쓰게 되어 LZ77의 출력을 더욱 압축할 수 있게 되는 것이다.

[0042] 한편, 도 3b를 참조하면, RC4 알고리즘(302)은 RSA 데이터 시큐리티(Data Security)사의 론 리베스트(Ron Rivest)에 의해 개발된 스트림 암호 시스템으로서 외부에서 제공된 키(313)에 따라 임의 길이의 유사 난수 열을 발생시키고(314), 이를 평문과 XOR 연산 수행하여(315) 암호화를 완료하게 되는 알고리즘이다.

[0043] 호프만 인코딩 알고리즘과 RC4 알고리즘의 결합

[0044] 도 4 는 본 발명의 일 실시예에 따른 호프만 인코딩 알고리즘과 RC4 알고리즘의 결합 방법을 설명하기 위한 도면이다.

[0045] 도 3a를 참조하여 설명한 기존 호프만 인코딩 알고리즘(300, 301)에서는 입력 데이터가 입력되면 호프만 트리에 따라 특정 비트열로 인코딩이 된다. 그러나, 도 4를 참조하면, 본 발명의 호프만 인코딩 알고리즘은 RC4 알고리즘과 결합된 방식이다. 먼저, 입력된 데이터(410)에 따른 인코딩 데이터(400)가 정해지면, 난수 발생기(401)에 의해 생성된 8 bit의 정보를 판독하고 이와 같은 비트 길이를 갖는 접두 부호 코드(402)의 개수와 모듈로(mod) 연산(411)을 수행한다. 그 후, 그 결과 값(403) 만큼 인코딩 데이터(400)로부터 건너뛰어 얻은 접두 부호(404)를 인코딩 값으로 출력한다. 그 결과 값이 도면부호 408로 나타내어지는 값이다. 한편, 접미 부호에 대한 인코딩은 다음과 같은 방식으로 이루어진다. 먼저 접미 부호 길이(405) 만큼의 난수 열(406)을 판독하고

이를 상기 접미 부호와 함께 XOR 연산(407)을 취한다. 그 결과 값(409)이 인코딩 값으로서 출력되게 된다.

[0046] 본 발명의 방식에 따르면 모든 호프만 인코딩 열에 대해 XOR 연산을 취하지 않게 되고 호프만 인코딩 알고리즘과 치환 연산의 수행이 동시에 가능해지게 되므로 연산 효율성이 높아지며, 접미 부호에 대해서는 XOR 연산을 취함으로써 보안성이 더욱 높아지게 된다.

[0047] 압축 및 암호화 연산 메모리 관리 모듈

[0048] 이하에서는, 압축 및 암호화 연산 메모리 관리 모듈의 구성에 대해 설명하기로 한다.

[0049] 도 5는 본 발명의 일 실시예에 따른 압축 및 암호화 연산 메모리 관리 모듈의 구성을 나타내는 도면이다.

[0050] 전술한 바와 같이, 종래의 방식에 따라 압축 연산과 암호화 연산을 순차적으로 수행할 경우에는 압축 연산에 필요한 메모리를 할당 및 해제하고, 이와 별도로 암호화 연산에 필요한 메모리를 할당 및 해제하게 되는데 이는 불필요한 메모리간 복사 연산을 수반하게 되며, 할당 해제에 있어서도 불필요한 반복적인 연산을 수반하게 되는 문제가 있었다.

[0051] 본 발명에서는 이러한 문제점을 해결하기 위해 압축 연산과 암호화 연산에 필요한 모든 메모리를 압축 및 암호화 연산 메모리 관리 모듈에서 일괄적으로 관리하도록 하였다. 이에 따르면, 메모리 할당 및 해제에 따른 연산 오버헤드를 줄여 단위 시간당 처리 속도를 더욱 높일 수 있다.

[0052] 도 5에 도시되는 바와 같이, 본 발명의 압축 및 암호화 연산 메모리 관리 모듈은 메모리 테이블 디렉토리(500)와 메모리 청크(501; Chunk)로 구성된다.

[0053] 또다시, 메모리 테이블 디렉토리(500)는 LZ77 알고리즘에 사용되는 해쉬 테이블(502)을 저장하기 위한 메모리를 가리키고 있는 포인터(503), 호프만 인코딩 알고리즘을 위한 두 개의 트리(504, 505)를 가리키고 있는 포인터(506, 507), RC4 알고리즘을 수행하기 위해 필요한 키에 대한 값(508)을 가리키고 있는 포인터(509), 압축 및 암호화 연산의 출력 값을 저장하기 위한 버퍼(510)를 가리키고 있는 포인터(511), 이전 압축 연산이 처리되고 있는 위치를 가리키기 위한 포인터(512), 이전 암호화 연산이 처리되고 있는 위치를 가리키기 위한 포인터(513), LZ77 알고리즘의 입력 버퍼(514)를 가리키고 있는 포인터(515), 입력 버퍼에서 이전까지 압축 및 암호화가 처리된 위치를 가리키기 위한 포인터(516)를 포함한다.

[0054] 본 발명의 압축 및 암호화 연산 메모리 관리 모듈에 따르면, 각 알고리즘의 시작과 종료 전후에 수행되는 불필요한 메모리 할당 및 해제가 제거되게 되어, 불필요한 메모리간 복사 연산이 제거되며, 할당 해제에 있어서도 불필요한 반복적인 연산이 제거되는 효과를 얻을 수 있게 된다. 또한, 이에 따라 메모리 할당 및 해제에 따른 연산 오버헤드를 줄여 단위 시간당 처리 속도를 더욱 높일 수 있게 된다.

[0055] 이상에서 본 발명이 구체적인 구성요소 등과 같은 특정 사항들과 한정된 실시예 및 도면에 의해 설명되었으나, 이는 본 발명의 보다 전반적인 이해를 돕기 위해서 제공된 것일 뿐, 본 발명이 상기 실시예들에 한정되는 것은 아니며, 본 발명이 속하는 기술분야에서 통상적인 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형을 꾀할 수 있다.

[0056] 따라서, 본 발명의 사상은 상기 설명된 실시예에 국한되어 정해져서는 아니되며, 후술하는 특허청구범위뿐만 아니라 이 특허청구범위와 균등하게 또는 등가적으로 변형된 모든 것들은 본 발명의 사상의 범주에 속한다고 할 것이다.

## *도면의 간단한 설명*

[0057] 도 1은 종래 기술에 따른 압축 및 암호화 연산 방법을 설명하는 흐름도이다.

[0058] 도 2a 및 도 2b는 본 발명의 일 실시예에 따른 압축 및 암호화 연산 방법을 설명하는 흐름도이다.

[0059] 도 3a는 통상적인 호프만 인코딩 알고리즘을 설명하기 위한 도면이다.

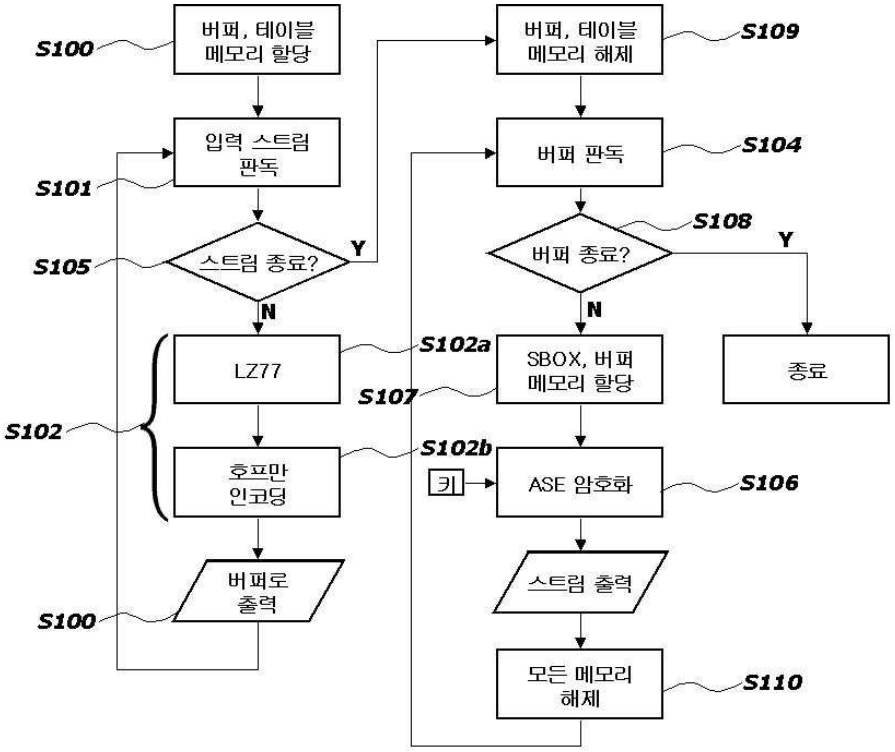
[0060] 도 3b는 통상적인 RC4 암호화 알고리즘을 설명하기 위한 도면이다.

[0061] 도 4는 본 발명의 일 실시예에 따른 호프만 인코딩 알고리즘/RC4 암호화 알고리즘 결합 연산을 설명하는 흐름도이다.

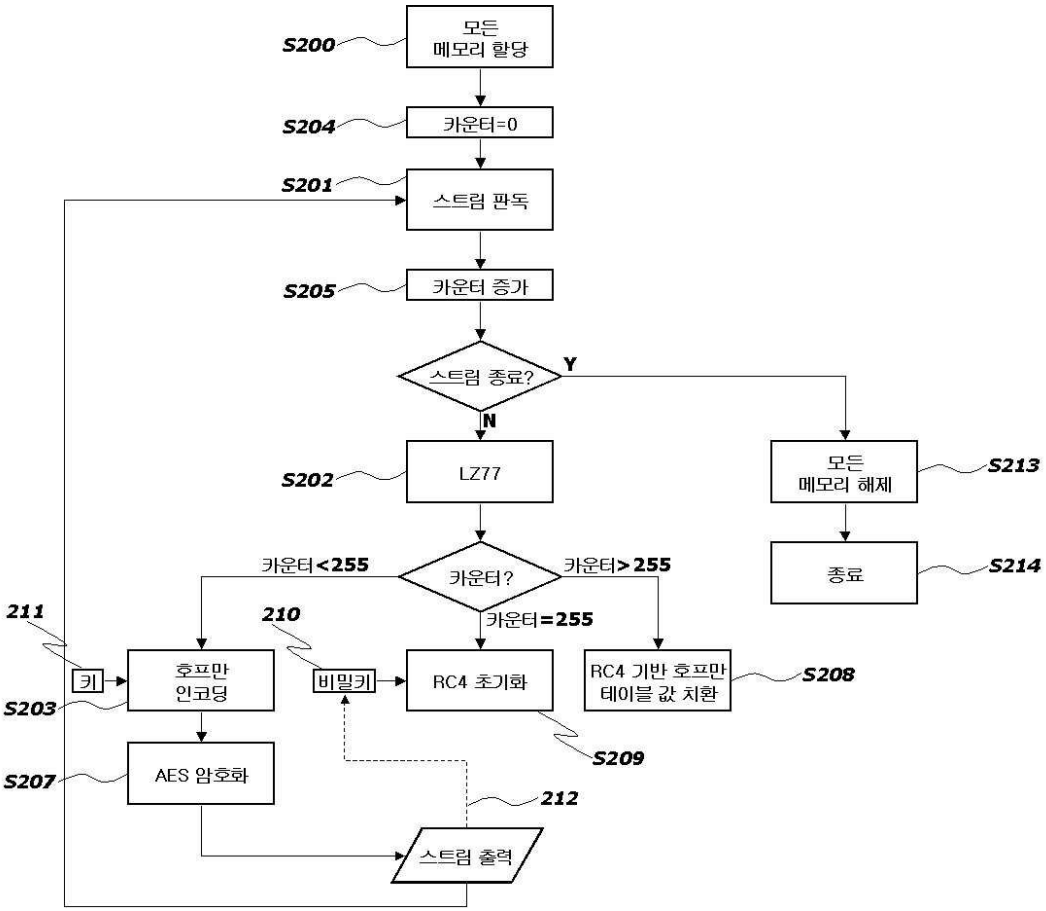
[0062] 도 5는 본 발명의 일 실시예에 따른 압축 및 암호화 연산 메모리 관리 모듈의 구성을 나타내는 도면이다.

## 도면

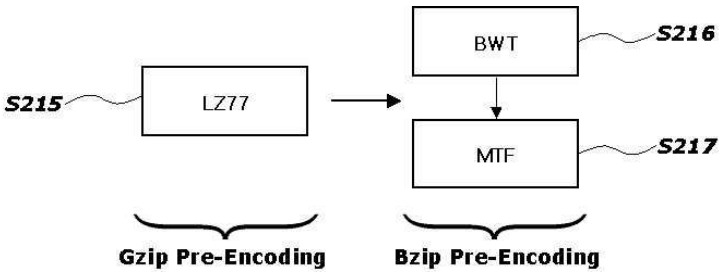
### 도면1



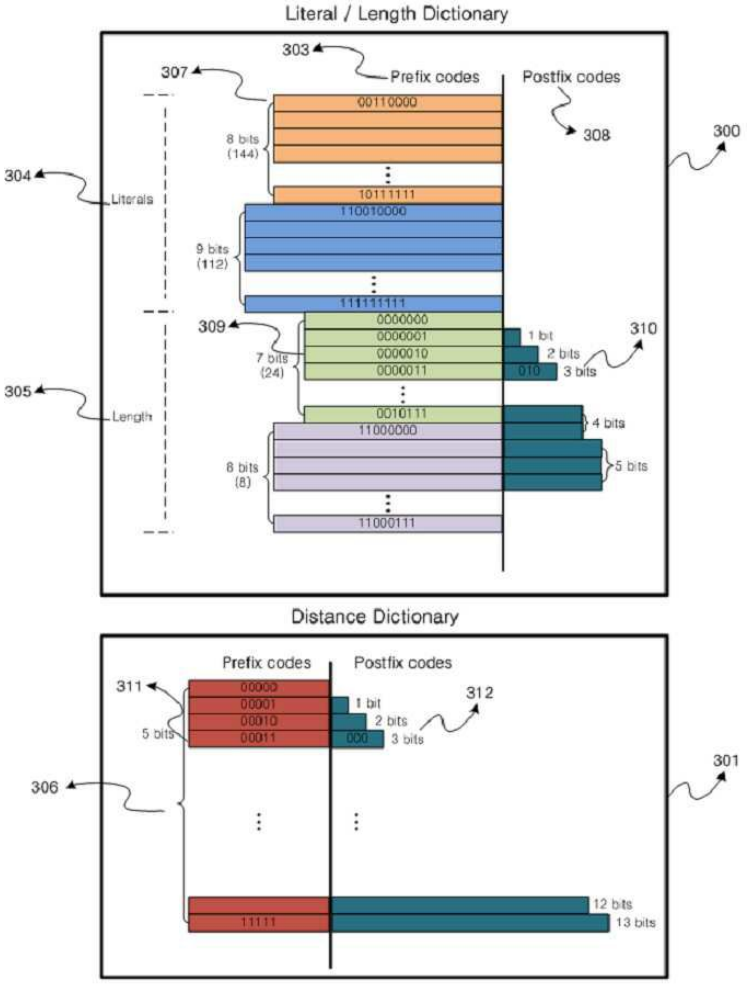
도면2a


S200
모든 메모리 할당
S204
카운터=0
S201
스트림 판독
S205
카운터 증가
스트림 종료?
Y
N
S202
LZ77
모든 메모리 해제
S213
종료
S214
카운터?
카운터<255
카운터>255
카운터=255
211
210
키
호프만 인코딩
비밀키
RC4 초기화
RC4 기반 호프만 테이블 값 치환
S208
S203
S209
AES 암호화
S207
212
스트림 출력

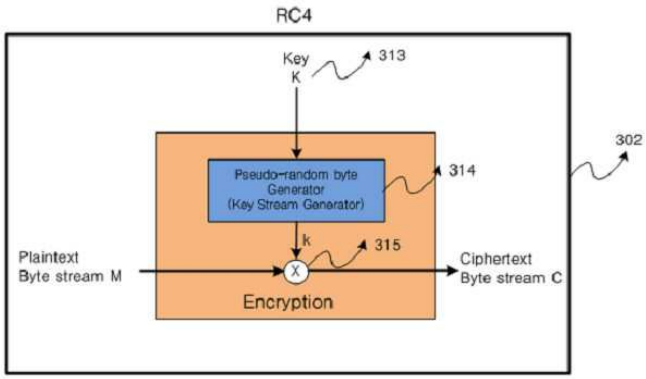
도면2b


S215
LZ77
BWT
S216
MTF
S217
Gzip Pre-Encoding
Bzip Pre-Encoding

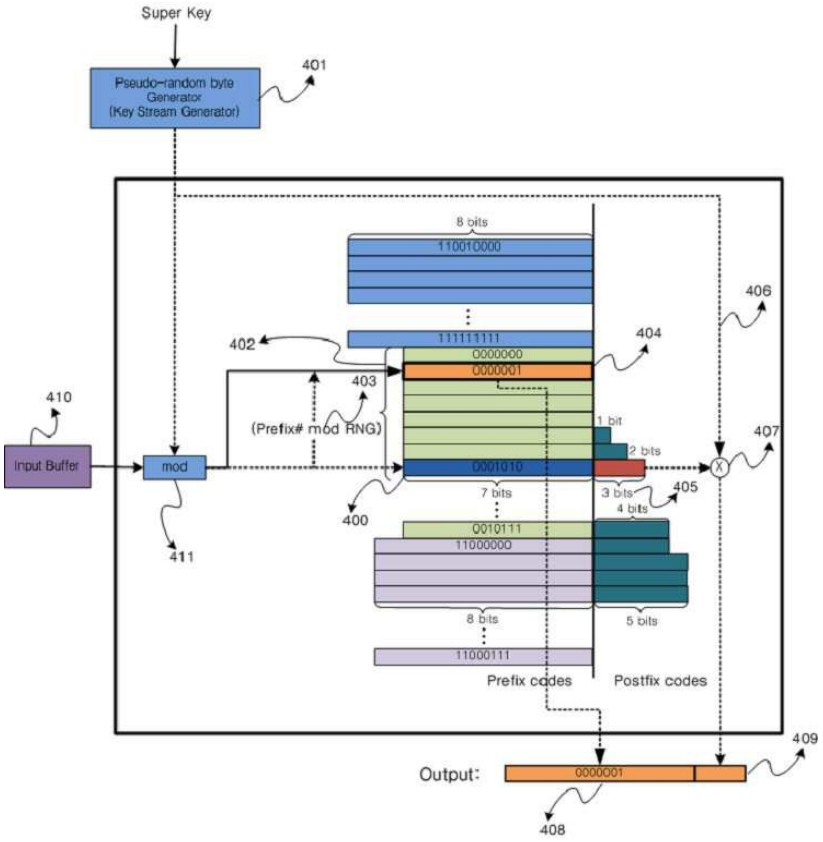
도면3a


Literal / Length Dictionary
303
Prefix codes
Postfix codes
307
308
300
00110000
8 bits
(144)
10111111
304
Literals
110010000
9 bits
(112)
111111111
309
0000000
0000001
0000010
0000011
7 bits
(24)
1 bit
2 bits
3 bits
310
305
Length
0010111
11000000
4 bits
5 bits
8 bits
(8)
11000111
Distance Dictionary
Prefix codes
Postfix codes
311
00000
00001
00010
00011
5 bits
1 bit
2 bits
3 bits
312
306
301
11111
12 bits
13 bits

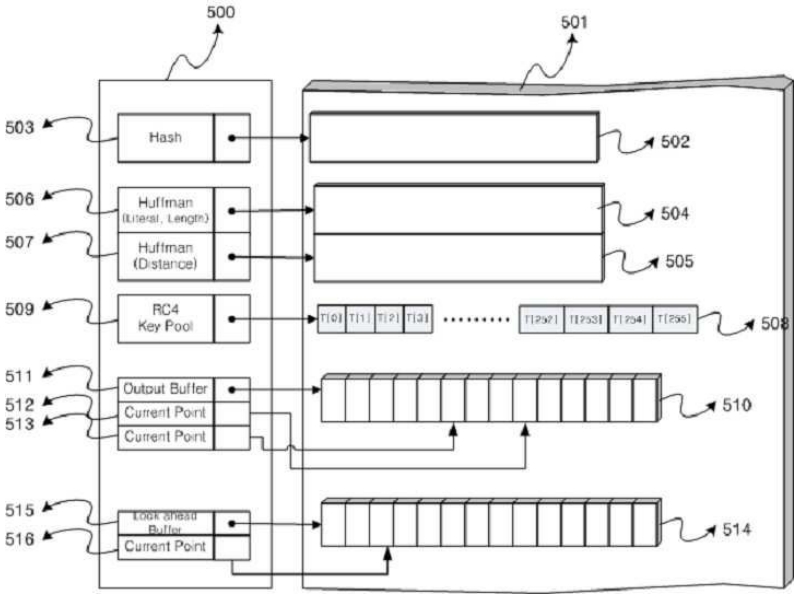
도면3b


RC4
Key
K
313
302
Pseudo-random byte
Generator
(Key Stream Generator)
314
k
315
Plaintext
Byte stream M
Ciphertext
Byte stream C
Encryption

도면4


Super Key
Pseudo-random byte Generator (Key Stream Generator)
401
8 bits
110010000
406
111111111
404
402
0000000
0000001
403
410
(Prefix# mod RNG)
1 bit
407
Input Buffer
mod
2 bits
0001010
7 bits
3 bits
4 bits
405
400
0010111
11000000
411
8 bits
5 bits
11000111
Prefix codes
Postfix codes
409
Output:
0000001
408

도면5


500
501
503
Hash
502
506
Huffman (Literal, Length)
504
507
Huffman (Distance)
505
509
RC4 Key Pool
T[0] T[1] T[2] T[3] ..... T[252] T[253] T[254] T[255]
508
511
Output Buffer
512
Current Point
510
513
Current Point
515
Look ahead Buffer
514
516
Current Point