



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2016-0111668
(43) 공개일자 2016년09월27일

(51) 국제특허분류(Int. Cl.)
H04L 29/06 (2006.01) H04L 29/08 (2006.01)
(52) CPC특허분류
H04L 69/166 (2013.01)
H04L 69/325 (2013.01)
(21) 출원번호 10-2015-0036581
(22) 출원일자 2015년03월17일
심사청구일자 없음

(71) 출원인
삼성전자주식회사
경기도 수원시 영통구 삼성로 129 (매탄동)
경희대학교 산학협력단
경기도 용인시 기흥구 덕영대로 1732 (서천동, 경희대학교 국제캠퍼스내)
(72) 발명자
이성원
경기도 성남시 분당구 중앙공원로 17 시범단지한양아파트 322동 502호
임한나
경기도 성남시 분당구 중앙공원로 54 시범단지우성아파트 220동 106호
(뒷면에 계속)
(74) 대리인
이전주

전체 청구항 수 : 총 20 항

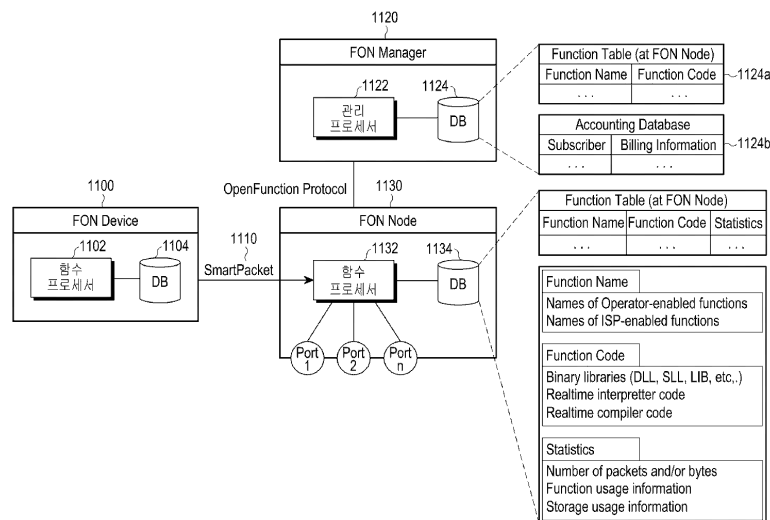
(54) 발명의 명칭 이동 통신 시스템에서 패킷 생성 방법 및 장치

(57) 요약

본 발명은 이동 통신 시스템에서 패킷 생성 방법 및 장치에 관한 것이다.

본 발명의 실시 예에 따른 방법은, 이동 통신 시스템에서의 디바이스에서 패킷 생성 방법에 있어서, 함수 기반의 패킷을 생성하는 과정; 및 상기 패킷을 타겟 노드로 전송하는 과정을 포함하고, 상기 패킷은 IP(internet protocol) 주소를 포함하지 않음을 특징으로 하는 한다.

대표도



(72) 발명자

김기정

경기도 수원시 영통구 청명로59번길 2, B02호

박중신

서울특별시 영등포구 도림로64가길 15-1

명세서

청구범위

청구항 1

이동 통신 시스템에서의 디바이스에서 패킷 생성 방법에 있어서,
네트워크가 제공하는 기능을 포함하는, 함수 기반의 패킷을 생성하는 과정; 및
상기 패킷을 타겟 노드로 전송하는 과정을 포함하고,
상기 패킷은 IP(internet protocol) 주소를 포함하지 않음을 특징으로 하는 디바이스에서 패킷 생성 방법.

청구항 2

제1항에 있어서,
상기 패킷은 함수 이름 및 파라미터를 포함하는 함수 호출 필드를 포함함을 특징으로 하는 디바이스에서 패킷 생성 방법.

청구항 3

제2항에 있어서,
상기 함수 이름은, 패킷 전달 관련 함수, 시스템 정보 조회 함수, 사용자 정보 조회 함수, 사용자 저장 공간 사용 관련 함수, 사업자 기능 수행 관련 함수 중 적어도 하나를 포함함을 특징으로 하는 디바이스에서 패킷 생성 방법.

청구항 4

제2항에 있어서,
상기 파라미터는 호출하고자 하는 함수의 입력 파라미터 및 출력 파라미터를 포함함을 특징으로 하는 디바이스에서 패킷 생성 방법.

청구항 5

제1항에 있어서,
함수 이름과 함수 코드를 포함하는 함수 테이블을 저장하는 과정을 더 포함함을 특징으로 하는 디바이스에서 패킷 생성 방법.

청구항 6

이동 통신 시스템에서의 디바이스에서 패킷 생성 장치에 있어서,
네트워크가 제공하는 기능을 포함하는, 함수 기반의 패킷을 생성하는 생성부; 및
상기 패킷을 타겟 노드로 전송하는 전송부를 포함하고,
상기 패킷은 IP(internet protocol) 주소를 포함하지 않음을 특징으로 하는 디바이스에서 패킷 생성 장치.

청구항 7

제6항에 있어서,

상기 패킷은 함수 이름 및 파라미터를 포함하는 함수 호출 필드를 포함함을 특징으로 하는 디바이스에서 패킷 생성 장치.

청구항 8

제7항에 있어서,

상기 함수 이름은,

패킷 전달 관련 함수, 시스템 정보 조회 함수, 사용자 정보 조회 함수, 사용자 저장 공간 사용 관련 함수, 사업자 기능 수행 관련 함수 중 적어도 하나를 포함함을 특징으로 하는 디바이스에서 패킷 생성 장치.

청구항 9

제7항에 있어서,

상기 파라미터는 호출하고자 하는 함수의 입력 파라미터 및 출력 파라미터를 포함함을 특징으로 하는 디바이스에서 패킷 생성 장치.

청구항 10

제6항에 있어서,

함수 이름과 함수 코드를 포함하는 함수 테이블을 저장하는 저장부를 더 포함함을 특징으로 하는 디바이스에서 패킷 생성 장치.

청구항 11

이동 통신 시스템에서의 네트워크에서 패킷 수신 방법에 있어서,

네트워크가 제공하는 기능을 포함하는, 함수 기반의 패킷을 수신하는 과정;

상기 패킷의 함수 테이블을 이용하여 함수를 확인하는 과정;

해당 함수를 실행하는 과정을 포함하고,

상기 패킷은 IP(internet protocol) 주소를 포함하지 않음을 특징으로 하는 네트워크에서 패킷 수신 방법.

청구항 12

제11항에 있어서,

상기 패킷은 함수 이름 및 파라미터를 포함하는 함수 호출 필드를 포함함을 특징으로 하는 네트워크에서 패킷 수신 방법.

청구항 13

제12항에 있어서,

상기 함수 이름은,

패킷 전달 관련 함수, 시스템 정보 조회 함수, 사용자 정보 조회 함수, 사용자 저장 공간 사용 관련 함수, 사업자 기능 수행 관련 함수 중 적어도 하나를 포함함을 특징으로 하는 네트워크에서 패킷 수신 방법.

청구항 14

제12항에 있어서,

상기 파라미터는 호출하고자 하는 함수의 입력 파라미터 및 출력 파라미터를 포함함을 특징으로 하는 네트워크에서 패킷 수신 방법.

청구항 15

제11항에 있어서,

함수 이름과 함수 코드를 포함하는 함수 테이블을 저장하는 과정을 더 포함함을 특징으로 하는 네트워크에서 패킷 수신 방법.

청구항 16

이동 통신 시스템에서의 네트워크에서 패킷 수신 장치에 있어서,

네트워크가 제공하는 기능을 포함하는, 함수 기반의 패킷을 수신하는 수신부; 및

상기 패킷의 함수 테이블을 이용하여 함수를 확인하고, 해당 함수를 실행하는 제어부를 포함하고,

상기 패킷은 IP(internet protocol) 주소를 포함하지 않음을 특징으로 하는 네트워크에서 패킷 수신 장치.

청구항 17

제16항에 있어서,

상기 패킷은 함수 이름 및 파라미터를 포함하는 함수 호출 필드를 포함함을 특징으로 하는 네트워크에서 패킷 수신 장치.

청구항 18

제17항에 있어서,

상기 함수 이름은, 패킷 전달 관련 함수, 시스템 정보 조회 함수, 사용자 정보 조회 함수, 사용자 저장 공간 사용 관련 함수, 사업자 기능 수행 관련 함수 중 적어도 하나를 포함함을 특징으로 하는 네트워크에서 패킷 수신 장치.

청구항 19

제17항에 있어서,

상기 파라미터는 호출하고자 하는 함수의 입력 파라미터 및 출력 파라미터를 포함함을 특징으로 하는 네트워크에서 패킷 수신 장치.

청구항 20

제16항에 있어서,

함수 이름과 함수 코드를 포함하는 함수 테이블을 저장하는 저장부를 더 포함함을 특징으로 하는 네트워크에서 패킷 수신 장치.

발명의 설명

기술 분야

[0001] 본 발명은 이동 통신 시스템에서 패킷 생성 방법 및 장치에 관한 것이다.

배경 기술

[0002] 아이피(IP : Internet Protocol) 네트워킹은 인터넷 기술의 비약적인 발전에 힘입어 급속한 발전을 이루고 있는 기술이다. 이러한 아이피 네트워킹에서는 각각 고정된 주소 즉, 아이피 주소(IP Address)를 가지고 각 사용자들 및 특정 서버가 동작하게 되며, 상기 주소를 근간으로 하여 라우팅(routing)이 이루어진다.

[0003] 한편, 이동통신 시스템에서도 단말에게 보다 많은 데이터를 제공하기 위해 여러 가지 방식들이 제안되고 있으며, 이러한 방식 중 하나로 이동 단말에 IP 주소를 할당하는 즉, 모바일 아이피(Mobile IP)라는 개념이 도입되기에 이르렀다.

[0004] 라우터는 IP 네트워킹을 지원함으로써, 글로벌 네트워크에서 종단의 단말 간 통신을 가능하게 한다. 하지만 IP 네트워킹에서의 IP 주소는 글로벌 네트워크에서 단말의 고유한 식별자로만 사용하기 때문에 단순 패킷 전송만을 지원할 뿐, 멀티미디어 및 상호교환적인 통신 서비스를 지원하기에 어려움이 있다. 더 나아가 다양한 응용서비스들이 만들어지는 환경에서 IP 주소만으로는 서비스를 인식할 수 없기 때문에 네트워크 안에서 부가 서비스를 제공하는 것이 불가능하다. 또 라우터는 제어 평면과 데이터 평면을 하나의 장치로 통합하고, 라우터 마다 서로 다른 제어 평면 정보를 가지면서 각자 독립적으로 동작하기 때문에, 새로운 네트워킹 프로토콜을 도입하기에 어려운 구조를 가지고 있다. 또한 라우터는 완전 분산 방식의 라우팅 알고리즘과 프로토콜을 사용하기 때문에, 망 운영자(Network Operator)가 라우팅 정책을 직접 정의하지 못하는 문제를 가지고 있다. 사용자(User)와 인터넷 서비스제공자(Internet Service Provider)도 라우팅 정책에 관여하지 못하며, 네트워크의 정적/동적 정보를 확인할 수도 없는 문제점이 있다.

발명의 내용

해결하려는 과제

[0005] 본 발명은 이동 통신 시스템에서 패킷 생성 방법 및 장치를 제공한다.

[0006] 본 발명은 사용자/인터넷 서비스 제공자가 원하는 기능을 직접 망에 적용하는 방법 및 장치를 제공한다.

과제의 해결 수단

[0007] 본 발명의 실시 예에 따른 방법은, 이동 통신 시스템에서의 디바이스에서 패킷 생성 방법에 있어서, 함수 기반의 패킷을 생성하는 과정; 및 상기 패킷을 타겟 노드로 전송하는 과정을 포함하고, 상기 패킷은 IP(internet protocol) 주소를 포함하지 않음을 특징으로 한다.

발명의 효과

[0008] 본 발명은 사용자/인터넷 서비스 제공자가 원하는 기능을 직접 망에 적용할 수 있다.

도면의 간단한 설명

[0009]

- 도 1은 라우터(Router)의 패킷 전송 방식을 지원하는 라우터 구조도;
- 도 2는 라우터(Router)의 패킷 전송 방식을 지원하는 IP 패킷 구조도;
- 도 3은 오픈 플로우의 패킷 전송 방식을 지원하는 라우터 구조도;
- 도 4는 오픈 플로우의 패킷 전송 방식을 지원하는 라우터에서의 IP 패킷 처리 과정을 도시한 흐름도;
- 도 5는 IP 어플리케이션(IP Application)의 IP 패킷 생성/전송 및 수신 과정을 나타낸 흐름도;
- 도 6은 도 5에서 설명한 IP 기반 어플리케이션의 소켓 제어 관련 프로그램 코드의 예시도;
- 도 7은 TCP/UDP Layer에서의 TCP/UDP 세그먼트 전송 과정을 나타낸 흐름도;
- 도 8은 IP Layer에서의 IP 패킷 전송 과정을 나타낸 흐름도;
- 도 9는 IP Layer에서의 IP 패킷 수신 과정을 나타낸 흐름도;
- 도 10은 TCP/UDP Layer에서의 TCP/UDP 세그먼트 수신 과정을 나타낸 흐름도;
- 도 11은 본 발명의 실시 예에 따른 패킷 생성을 위한 이동 통신 시스템 구조도;
- 도 12는 본 발명의 실시 예에 따른 스마트 패킷 구조도;
- 도 13 및 도 14는 본 발명이 적용 가능한 네트워크 구성의 예시도;
- 도 15는 본 발명의 실시 예에 따른 FON 디바이스의 세부 구조도;
- 도 16은 본 발명의 실시 예에 따른 FON 디바이스의 다른 세부 구조도;
- 도 17은 FON 어플리케이션에서의 스마트 패킷 생성/전송 및 수신 과정을 나타낸 흐름도;
- 도 18은 FON 디바이스에서 FON 기반 어플리케이션의 소켓 제어 관련 프로그램 코드의 예시도;
- 도 19는 FON Layer에서의 스마트 패킷 전송 과정을 흐름도;
- 도 20은 FON 디바이스의 DB에 저장되어 있는 함수 테이블의 구조도;
- 도 21은 FON Layer 에서의 스마트 패킷 수신 과정을 나타낸 흐름도;
- 도 22는 FON 노드에서의 스마트 패킷 수신 처리 과정을 나타낸 흐름도;
- 도 23은 FON 노드의 DB에 저장되어 있는 함수 테이블의 구조 및 예시도;
- 도 24는 함수 이용 정보 보고 메시지의 일 예를 도시한 포맷;
- 도 25는 FON 노드가 지원하는 기본 함수에 대한 리스트의 예시도;
- 도 26은 본 발명의 실시 예에 따른 함수 배포 메시지 구조도 및 예시도;
- 도 27은 본 발명의 실시 예에 따른 함수 삭제 메시지 구조도 및 예시도;
- 도 28은 본 발명의 실시 예에 따른 함수 추가 메시지 구조도 및 예시도;
- 도 29는 본 발명의 실시 예에 따른 회계 테이블 구조도;
- 도 30은 일반적인 4G/5G 이동통신 네트워크에서의 센서와 사용자 단말 간의 데이터 통신의 예시를 나타낸 시나리오;
- 도 31은 도 30의 전송 시나리오를 본 발명을 적용했을 때의 사용자 단말과 센서 간의 데이터 통신 과정을 나타낸 시나리오;
- 도 32는 일반적인 종단 간 패킷 전송 경로 정보 수집 예시도;
- 도 33은 본 발명의 실시 예에 따른 종단 간 패킷 전송 경로 정보 수집 시나리오;
- 도 34는 본 발명의 실시 예에 따른 생체모방 개미 집단 알고리즘을 활용한 패킷 전송 최적화 예시를 나타낸 도면;

도 35는 종래의 3rd-Party 웹 서비스 또는 미디어 서비스 가속화 예시도; 및

도 36은 본 발명의 실시 예에 따른 3rd-Party 웹 서비스 또는 미디어 서비스 가속화 예시도.

발명을 실시하기 위한 구체적인 내용

- [0010] 이하, 첨부된 도면을 참조하여 본 발명의 바람직한 실시 예들을 상세히 설명한다. 이때 첨부된 도면에서 동일한 구성 요소는 가능한 동일한 부호로 나타내고 있음에 유의하여야 한다. 또한 본 발명의 요지를 흐리게 할 수 있는 공지 기능 및 구성에 대한 상세한 설명은 생략할 것이다.
- [0011] 또한 이하에서 설명되는 본 명세서 및 청구범위에 사용된 용어나 단어는 통상적이거나 사전적인 의미로 한정해서 해석되어서는 아니되며, 발명자는 그 자신의 발명을 가장 최선의 방법으로 설명하기 위해 용어의 개념으로 적절하게 정의할 수 있다는 원칙에 입각하여 본 발명의 기술적 사상에 부합하는 의미와 개념으로 해석되어야만 한다.
- [0012] 본 출원에서 사용한 용어는 단지 특정한 실시 예를 설명하기 위해 사용된 것으로, 본 발명을 한정하려는 의도가 아니다. 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 출원에서, "포함하다" 또는 "가지다" 등의 용어는 명세서상에 기재된 특징, 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것이 존재함을 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.
- [0013] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가지고 있다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥 상 가지는 의미와 일치하는 의미를 가지는 것으로 해석되어야 하며, 본 출원에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.
- [0014] 본 발명의 실시 예는 롱 텀 에볼루션 (LTE: Long-Term Evolution, 이하 'LTE'라 칭하기로 한다) 이동 통신 시스템과, 롱 텀 에볼루션-어드밴스드 (LTE-A: Long-Term Evolution-Advanced, 이하 'LTE-A'라 칭하기로 한다) 이동 통신 시스템과, 고속 하향 링크 패킷 접속(high speed downlink packet access: HSDPA, 이하 'HSDPA'라 칭하기로 한다) 이동 통신 시스템과, 고속 상향 링크 패킷 접속(high speed uplink packet access: HSUPA, 이하 'HSUPA'라 칭하기로 한다) 이동 통신 시스템과, 3세대 프로젝트 파트너십 2(3rd generation project partnership 2: 3GPP2, 이하 '3GPP2'라 칭하기로 한다)의 고속 레이트 패킷 데이터(high rate packet data: HRPD, 이하 'HRPD'라 칭하기로 한다) 이동 통신 시스템과, 3GPP2의 광대역 부호 분할 다중 접속(WCDMA: Wideband Code Division Multiple Access, 이하 'WCDMA'라 칭하기로 한다) 이동 통신 시스템과, 3GPP2의 부호 분할 다중 접속(CDMA: Code Division Multiple Access, 이하 'CDMA'라 칭하기로 한다) 이동 통신 시스템과, 국제 전기 전자 기술자 협회(IEEE: Institute of Electrical and Electronics Engineers, 이하 'IEEE'라 칭하기로 한다) 802.16m 통신 시스템과, 진화된 패킷 시스템(EPS: Evolved Packet System, 이하 'EPS'라 칭하기로 한다)과, 모바일 인터넷 프로토콜(Mobile Internet Protocol: Mobile IP, 이하 'Mobile IP'라 칭하기로 한다) 시스템 등과 같은 다양한 통신 시스템들에 적용 가능한 물론이다.
- [0015] 패킷 라우팅 및 포워딩 기술로, 제어 평면(Control Plane)과 데이터 평면(Data Plane)이 통합되어 있는 라우터(Router)의 패킷 전송 방식과, 제어 평면과 데이터 평면이 분리된 형태를 가지는 오픈 플로우(openflow)의 패킷 전송 방식이 있다.
- [0016] 도 1은 라우터(Router)의 패킷 전송 방식을 지원하는 라우터 구조도이다.
- [0017] 라우터(100)는 IP 패킷에 대한 라우팅 및 포워딩을 제공하기 위해서 라우팅 프로세서(Routing Processor)(110), 포워딩 프로세서(120), DB(Database)(130)를 포함한다.
- [0018] 라우팅 프로세서(110)는 목적지 IP 주소에 따른 물리적 포트 정보(이하, "포트 정보"이라 칭함)를 포함하는 라우팅 테이블(140)을 생성한다.
- [0019] DB(130)는 생성된 라우팅 테이블을 저장한다.
- [0020] 포워딩 프로세서(120)는 IP 패킷을 수신하면, 해당 IP 패킷의 목적지 IP 주소에 해당하는 물리 포트를 식별하기 위해서 DB(130)에 저장되어 있는 라우팅 테이블(140)을 참조한다. 포워딩 프로세서(120)는 수신한 패킷의 목적

지 IP 주소가 라우팅 테이블에 존재하면, 목적지 IP 주소에 해당하는 물리 포트로 포워딩한다. 그러나, 수신한 패킷의 목적지 IP 주소에 대한 정보가 라우팅 테이블에 존재하지 않으면 포워딩 프로세서(120)는 디폴트 물리 포트로 포워딩한다.

- [0021] 도 2는 라우터(Router)의 패킷 전송 방식을 지원하는 IP 패킷 구조도이다.
- [0022] 도 2의 IP 패킷(200)은 트레일러(Trailer)(205), 페이로드(Payload)(207), 발신지 L3/L4 주소 필드(210), 목적지 L3/L4 주소 필드(220), 발신지 MAC 헤더(230), 목적지 MAC 헤더(240)를 포함한다. 발신지 L3/L4 주소 필드(210), 목적지 L3/L4 주소 필드(220)에는 IPv4/IPv6 주소(이하, "IP 주소"이라 칭함)(250), TCP/UDP 포트 넘버(260) 등을 포함한다.
- [0023] 도 3은 오픈 플로우의 패킷 전송 방식을 지원하는 라우터 구조도이다.
- [0024] 도 3의 라우터는 도 1의 라우터와 달리, 라우팅 프로세서(310)와 포워딩 프로세서(330)가 분리된다.
- [0025] 오픈 플로우 제어부(300)는 라우팅 프로세서(310), DB(312)를 포함한다.
- [0026] 오픈 플로우 스위치(320)는 포워딩 프로세서(330)와 DB(332)를 포함한다.
- [0027] 라우팅 프로세서(310)는 라우팅 테이블을 생성한다. DB(312)는 생성된 라우팅 테이블을 저장한다. 오픈 플로우 제어부(300)는 오픈 플로우 프로토콜(315)을 이용하여 패킷을 제어하기 위한 패킷 제어 정보를 오픈 플로우 스위치(320)로 전달한다. 패킷 제어 정보를 수신한 오픈 플로우 스위치(320)는 패킷 제어 정보를 Rule-Action 테이블(334)에 저장한다.
- [0028] 도 4는 오픈 플로우의 패킷 전송 방식을 지원하는 라우터에서의 IP 패킷 처리 과정을 도시한 흐름도이다.
- [0029] 포워딩 프로세서(330)는 401 단계에서 IP 패킷을 수신한다. 이후에 포워딩 프로세서(330)는 403 단계에서 수신한 IP 패킷의 IP 패킷 헤더를 확인한다. 포워딩 프로세서(330)는 IP 패킷 헤더를 통해 목적지/발신지 MAC 주소, 목적지/발신지 IP 주소, 목적지/발신지 TCP 포트 번호 등의 헤더 정보를 추출한다.
- [0030] 포워딩 프로세서(330)는 405 단계에서 추출된 헤더정보가 룰 액션 테이블(Rule-Action table)(334)에 작성되어 있는 Rule 정보와 일치하는지를 확인한다. 만약, 추출된 헤더정보와 Rule-Action 테이블(334)이 일치하지 않으면, 오픈 플로우 제어부(300)는 수신한 패킷을 포워딩 프로세서(330)로 포워딩하고, 오픈 플로우 제어부(300)의 결정에 따라 수신한 패킷을 특정 물리 포트로 포워딩하거나 폐기할 수 있다.
- [0031] 추출된 헤더정보와 Rule-Action 테이블(334)이 일치하면, 포워딩 프로세서(330)는 407 단계에서 Rule-Action 테이블(334)의 Action 타입을 확인/결정한다.
- [0032] 포워딩 프로세서(330)는 Action 정보(또는 타입)에 따라 플로우 제어 및 포워딩을 수행한다. Action 타입이 드랍인 경우, 포워딩 프로세서(330)는 409 단계에서 패킷을 드랍(drop)한다.
- [0033] Action 타입이 전달인 경우, 포워딩 프로세서(330)는 411 단계에서 단일 또는 복수 개의 물리 포트로 수신된 패킷을 포워딩한다.
- [0034] Action 타입이 보고인 경우, 포워딩 프로세서(330)는 413 단계에서 보고하기 위해 오픈 플로우 제어부(300)로 수신된 패킷을 포워딩한다.
- [0035] 상기에 기술한 라우팅 및 포워딩을 지원하는 네트워크에서의 종단 간 어플리케이션 데이터 통신에 대한 기술로, TCP/IP 기반의 소켓(Socket) 인터페이스를 이용한 어플리케이션 데이터 통신 기술이 있다. 이하에서는 TCP/IP 기반의 소켓(Socket) 인터페이스를 이용한 어플리케이션 데이터 통신 기술에 대해서 설명하기로 한다.
- [0036] 도 5는 IP 어플리케이션(IP Application)의 IP 패킷 생성/전송 및 수신 과정을 나타낸 흐름도이다.
- [0037] 도 5의 동작은 TCP/IP 프로토콜을 지원하는 단말에서 수행된다. IP 어플리케이션은 501 단계에서 socket() 함수를 호출하여 IP 통신을 위한 socket을 생성한다. IP 어플리케이션은 503 단계에서 단말 연결 함수(connection() 함수)를 호출하여 원격지 호스트로 연결을 시도한다.
- [0038] IP 어플리케이션은 505 단계에서 원격지 호스트로의 데이터를 전송하는 것인지 수신하는 것인지를 판단한다.
- [0039] 만약, 원격지 호스트로의 데이터를 전송하는 경우, IP 어플리케이션은 507 단계에서 write() 함수를 호출하여 TCP/UDP Layer로 데이터 전송을 요청한다. 511 단계에서와 같이, 더 이상 전송할 데이터가 없을 경우, IP 어플리케이션은 513 단계에서 close() 함수를 호출함으로써 원격지 호스트와 설정되어 있던 연결을 종료한다.

- [0040] 만약, 원격지 호스트로부터 데이터를 수신할 경우, IP 어플리케이션은 509 단계에서 read() 함수를 호출하여 TCP/UDP Layer에 원격지 호스트로부터 수신한 IP 패킷에서 IP 어플리케이션의 데이터 추출 및 수신을 요청한다. 이후, 511 단계에서와 같이, 더 이상 데이터를 수신하기를 원하지 않거나 더 이상 수신할 데이터가 없을 경우, IP 어플리케이션은 513 단계에서 close() 함수를 호출함으로써 원격지 호스트와 설정되어 있던 연결을 종료한다.
- [0041] 도 6은 도 5에서 설명한 IP 기반 어플리케이션의 소켓 제어 관련 프로그램 코드의 일 예를 나타낸 것이다.
- [0042] 참조번호 610은 도 5의 501 단계에서의 socket() 함수의 일 예를 나타낸다.
- [0043] 참조번호 620은 도 5의 503 단계에서의 연결 함수의 일 예를 나타낸다.
- [0044] 도 7은 TCP/UDP Layer에서의 TCP/UDP 세그먼트 전송 과정을 나타낸 흐름도이다.
- [0045] TCP/UDP Layer(Layer 4)는 701 단계에서 IP 어플리케이션으로부터 데이터를 수신한다.
- [0046] TCP/UDP Layer는 703 단계에서 전달 받은 데이터에 TCP/UDP 헤더를 추가하여 TCP/UDP 세그먼트를 생성한다. TCP/UDP Layer는 705 단계에서 생성된 세그먼트를 IP Layer(Layer 3)로 전달한다.
- [0047] 도 8은 IP Layer에서의 IP 패킷 전송 과정을 나타낸 흐름도이다.
- [0048] IP Layer는 801 단계에서 TCP/UDP Layer로부터 TCP/UDP 세그먼트를 수신한다.
- [0049] IP Layer는 803 단계에서 TCP/UDP 세그먼트에 IP 헤더를 추가하여 IP 패킷을 생성한다.
- [0050] IP Layer는 805 단계에서 생성된 IP 패킷을 Wi-Fi, LTE(Long Term Evolution) 같은 Layer 2로 전달한다.
- [0051] 도 9는 IP Layer에서의 IP 패킷 수신 과정을 나타낸 흐름도이다.
- [0052] IP Layer(Layer 3)는 901 단계에서 Wi-Fi, LTE와 같은 Layer 2로부터 원격지 호스트의 IP 패킷을 수신한다. 즉, IP Layer 는 Layer 2로부터 L2 프레임에서 프레임 헤더를 제거하여 IP 패킷을 추출한 것을 수신한다. IP 패킷을 전달 받은 IP Layer는 903 단계에서 IP 헤더를 제거하여 TCP/UDP 세그먼트를 추출하고, 905 단계에서 TCP/UDP Layer로 세그먼트를 전달한다.
- [0053] 도 10은 TCP/UDP Layer에서의 TCP/UDP 세그먼트 수신 과정을 나타낸 흐름도이다.
- [0054] TCP/UDP Layer는 1001 단계에서 IP Layer로부터 TCP/UDP 세그먼트를 수신한다. IP Layer로부터 TCP/UDP 세그먼트를 전달 받은 TCP/UDP Layer는 1003 단계에서 TCP/UDP 헤더를 제거하여 어플리케이션 데이터를 추출한다. TCP/UDP Layer는 1005 단계에서 IP 어플리케이션으로 어플리케이션 데이터를 전달한다.
- [0055] 라우터는 IP 네트워킹을 지원함으로써, 글로벌 네트워크에서 종단의 단말 간 통신을 가능하게 한다. 하지만 IP 네트워킹에서의 IP 주소는 글로벌 네트워크에서 단말의 고유한 식별자로만 사용하기 때문에 단순 패킷 전송만을 지원할 뿐, 멀티미디어 및 상호교환적인 통신 서비스를 지원하기에 어려움이 있다. 더 나아가 다양한 응용서비스들이 만들어지는 환경에서 IP 주소만으로는 서비스를 인식할 수 없기 때문에 네트워크 안에서 부가 서비스를 제공하는 것이 불가능하다. 또 라우터는 제어 평면과 데이터 평면을 하나의 장치로 통합하고, 라우터 마다 서로 다른 제어 평면 정보를 가지면서 각자 독립적으로 동작하기 때문에, 새로운 네트워킹 프로토콜을 도입하기에 어려운 구조를 가지고 있다. 라우터는 완전 분산 방식의 라우팅 알고리즘과 프로토콜을 사용하기 때문에, 망운영자(Network Operator)가 라우팅 정책을 직접 정의하지 못하는 문제를 가지고 있다. 사용자(User)와 인터넷서비스제공자(Internet Service Provider)도 라우팅 정책에 관여하지 못하며, 네트워크의 정적/동적 정보를 확인할 수도 없다.
- [0056] 종래의 라우터 기술의 문제점을 보완한 기술로 Software Defined Networking(SDN) 기술의 구현물인 오픈 플로우가 있다. 라우터는 완전 분산(Fully distributed)되어 있는 제어 평면 구조를 가지고 있는 반면, 오픈 플로우 는 중앙집중화(Centralized)된 제어 평면 구조를 가지고 있다. 또 라우터는 목적지 IP 주소만을 이용하여 포워딩할 물리 포트를 식별하여 패킷을 포워딩하는 반면, 오픈 플로우 는 목적지/발신지 MAC 주소, 목적지/발신지 IP 주소, 목적지/발신지 TCP 포트 번호 등 다양한 필드 정보를 이용한 패킷 포워딩을 하며, 뿐만 아니라 컨트롤러로 패킷 포워딩, 패킷 폐기 등을 수행할 수 있다. 이와 같은 오픈 플로우의 기술적 특성은 오픈 플로우 컨트롤러가 오픈 플로우 컨트롤러의 제어를 받는 오픈 플로우 스위치들로 패킷 제어 명령을 내림으로써, 패킷에 대한 플로우를 프로그래머블(Programmable)하게 제어할 수 있게 한다. 또 라우터와 달리, 목적지/발신지 MAC 주소, 목적지/발신지 IP 주소, 목적지/발신지 TCP 포트 번호 등 다양한 필드 정보를 이용하여 패킷을 포워딩하기 때문

에, 세밀한 플로우 제어가 가능하다.

- [0057] 종래 SDN/오픈 플로우 기술의 문제점 관련, 오픈 플로우는 라우터보다 세밀하고 프로그래머블한 플로우 제어를 제공하지만, IP 패킷에 대한 포워딩과 플로우 제어에만 한정된다. 그래서 오픈 플로우 또한 라우터와 마찬가지로 IP 프로토콜이 아닌 새로운 네트워크 계층 및 프로토콜을 제공할 수 없다. 구체적 말하자면, 망운영자/인터넷서비스제공자/사용자에 의한 네트워크 기능을 생성하고 수행할 수 있는 네트워크 프로그래머빌리티(Network Programmability)를 제공할 수 없다. 또 오픈 플로우는 망운영자에 의해서 플로우 제어만 가능할 뿐, 인터넷서비스제공자와 사용자에 의한 플로우 제어가 불가능하다. 오픈 플로우는 Rule-Action테이블에 작성되어 있는 플로우에 해당하는 패킷을 처리할 때마다 카운터를 증가시킬 뿐, 네트워크의 정적인 상태와 동적인 상태 정보를 수집할 수 없다. 더 나아가 오픈 플로우 스위치가 다수의 플로우를 제어하기 위해서는 복잡한 심층 패킷 분석(DPI: Deep Packet Inspection)을 요구하기 때문에 패킷 포워딩 측면에서 성능이 저하된다. 오픈 플로우 스위치가 Rule-Action테이블에 존재하지 않는 다수의 패킷 플로우를 수신할 경우에도 성능 저하 문제를 발생시킨다. Rule-Action테이블에 존재하지 않는 다수의 패킷 플로우를 수신한 오픈 플로우 스위치는 다수의 Packet-In 메시지를 생성하여 오픈 플로우 컨트롤러로 전송한다. 이때 잦은 Packet-In 메시지 전송이 오픈 플로우 스위치와 오픈 플로우 컨트롤러 사이의 인터페이스의 부하를 증가시켜, 오픈 플로우 컨트롤러에서 오픈 플로우 스위치로의 Packet-Out 메시지의 전송 지연이 발생한다. Packet-Out 메시지의 전송 지연은 오픈 플로우 스위치의 Rule-Action 테이블의 갱신의 지연으로 이어짐으로 패킷 포워딩 시간이 증가하게 된다.
- [0058] 먼저, 종래 라우터 기술의 문제점을 정리하면 다음과 같다.
- [0059] 첫째, IP 네트워크 프로토콜만 지원한다. IP는 40년된 프로토콜로서 '생존성'가 목적이다. 따라서, 미디어 및 인터랙티브 통신 서비스가 중심인 현재의 인터넷을 성능/기능 면에서 제대로 지원하지 못한다. 이는 IP 외에 새로운 네트워킹 프로토콜의 도입이 어렵기 때문이다.
- [0060] 둘째, 완전 분산 타입 제어 구조로서, 망운영자도 직접 라우팅 정책을 정의 못한다. 따라서, 사용자/인터넷서비스제공자는 라우팅 정책에 관여하지 못하는 것이 당연하며, 사용자/인터넷서비스제공자가 네트워크의 정적/동적 정보도 모르고 본인의 서비스를 구현하게 된다.
- [0061] 셋째, 패킷의 전송만 지원한다. 즉, 단순히 종단간 패킷 전달이 목적으로서, 서비스를 인식하여 부가 서비스를 제공하는 것이 불가능하며, 사용자/인터넷서비스제공자가 자신이 원하는 기능을 직접 망에 넣는 것은 당연하게 불가능하다.
- [0062] 다음으로, 종래 SDN/오픈 플로우의 문제점은 다음과 같다.
- [0063] 첫째, 종래 라우터 기술에서 해결하지 못한 문제로서, 망사업자(망운영자)가 패킷이 전달되는 경로를 결정하는 플로우(Flow)차원의 프로그래머빌리티는 지원하지만, 사용자/인터넷서비스제공자에 대해서 자신의 서비스에 속한 패킷의 경로를 결정하는 플로우(Flow)차원의 프로그래머빌리티는 지원하지 못한다.
- [0064] 둘째, 종래 라우터 기술에서 해결하지 못한 문제로서, 망사업자 및 사용자/인터넷서비스제공자가 자신이 원하는 기능을 네트워크 안에 구현하고 이를 서비스에서 활용하는 기술은 아직도 지원하지 못하고 있다.
- [0065] 셋째, 종래 라우터 기술에서 해결하지 못한 문제로서, 망사업자의 경우 네트워크의 동적/정적 상태를 구체적으로 파악하는 것이 어려우며, 사용자/인터넷서비스제공자의 경우 네트워크의 동적/정적 상태를 구체적으로 파악하는 것이 불가능하다.
- [0066] 넷째, SDN/오픈 플로우 기술의 등장으로 새롭게 발생한 문제로서, 패킷의 헤더 정보를 분석하기 위한 DPI 기능 구현의 부하가 기존 라우터 방식 대비 급격하게 증가한다.
- [0067] 다섯째, SDN/오픈 플로우 기술의 등장으로 새롭게 발생한 문제로서, 중앙집중화 된 SDN 컨트롤러와 오픈 플로우 스위치 간의 인터페이스 부하가 새롭게 발생하며, 네트워크 및 컨트롤러/스위치의 프로세싱 부하를 야기한다.
- [0068] 본 발명은 상기에 기술한 라우터와 SDN/오픈 플로우의 문제를 해결하기 위한 것이다. 본 발명은 망운영자/인터넷서비스제공자/사용자가 네트워크 장치에 새로운 네트워크 기능을 직접 쉽고 빠르게 도입할 수 있도록 하는 함수 기반 네트워크를 제안한다.
- [0069] 본 발명은 네트워크 기능에 대한 함수를 호출할 수 있는 스마트 패킷(SmartPacket)을 사용하여 네트워크 장치의 함수를 실행함으로써, IP 프로토콜뿐만 아니라 새로운 네트워크 기능 및 프로토콜을 제공하고 새로운 네트워크 서비스를 제공할 수 있다.

- [0070] 또한 본 발명은 망운영자/인터넷서비스제공자/사용자가 함수 실행정보를 포함하는 스마트 패킷을 이용하여 네트워크의 정적/동적 정보를 획득하게 하여, 망 관리를 용이하게 하는 것을 가능하게 한다.
- [0071] 도 11은 본 발명의 실시 예에 따른 패킷 생성을 위한 이동 통신 시스템 구조도이다.
- [0072] 도 11을 참조하면, 이동 통신 시스템은 FON 디바이스(Function-Oriented-Network Device, FON Device)(1100), FON 노드(Function-Oriented-Network Node, FON Node)(1130), 및 FON 매니저(Function-Oriented-Network Manager, FON Manager)(1120)를 포함한다.
- [0073] FON 디바이스(1100)는 함수 이름(Function Name)과 함수 코드(Function Code)를 가지고 있는 함수 테이블(Function Table)을 저장하는 DB(1104)와, 스마트 패킷(1110)을 생성하여 FON 노드로 전송하고, FON 노드로부터 스마트 패킷을 수신할 수 있는 함수 프로세서(Function Processor)(1102)를 포함한다. 스마트 패킷(1110)은 함수 기반의 패킷을 의미하고, IP 주소를 포함하지 않는다.
- [0074] 함수 프로세서(1102)는 FON 노드(1130)로부터 수신한 스마트 패킷(1110)의 함수 호출 필드에 명시되어 있는 함수를 실행할 수 있다. 또한 함수 프로세서(1102)는 FON 매니저(1120)로부터 함수 배포 메시지를 수신하고, 상기 함수 배포 메시지에 포함되어 있는 FON 매니저(1120)의 함수 테이블 정보를 FON 디바이스(1100)의 함수 테이블에 저장 및 업데이트를 수행한다. 함수 프로세서(1102)는 스마트 패킷(1110) 생성 시 DB(1104)에 저장되어 있는 함수 테이블을 이용하여 스마트 패킷(1110)의 검증을 수행할 수 있다.
- [0075] FON 디바이스에 대한 상세한 설명은 이후 도 15, 도 16의 설명에서 추가로 기술한다.
- [0076] FON 노드(1130)는 실행 가능한 함수 이름, 함수 코드, 함수 사용 횟수 또는 통계치(Statistics) 정보를 가지고 있는 함수 테이블을 저장하는 DB(1134)와, 스마트 패킷을 수신하고 함수 호출 필드의 정보를 추출하여 함수 이름에 해당하는 함수를 호출하여 실행하는 함수 프로세서(1132)로 구성된다. 함수 프로세서(1132)는 DB(1134)로부터 실행하고자 하는 함수 코드를 추출하여 동적 바인딩을 실시하고, 동적 바인딩된 함수를 실행하는 기능을 수행한다. 또한 함수 프로세서(1132)는 함수 실행 결과를 스마트 패킷의 페이로드에 명시할 수 있고, 다른 FON 디바이스 또는 FON 노드로 스마트 패킷을 전송할 수 있다. 함수 프로세서(1132)는 FON 매니저(1120)로부터 함수 배포 메시지를 수신하고, 상기 함수 배포 메시지에 포함되어 있는 FON 매니저(1120)의 함수 테이블 정보를 FON 노드(1130)의 함수 테이블에 저장 및 업데이트를 수행한다. 또 함수 프로세서(1132)는 FON 매니저로부터 수신한 함수 배포 메시지를 FON 디바이스로 전달함으로써, FON 매니저가 FON 디바이스로 함수 배포 메시지를 전송하는 것을 가능하게 한다. 함수 프로세서(1132)는 FON 디바이스로(1100)부터 함수 테이블에 존재하지 않는 함수 실행을 요청 받았을 때, FON 매니저(1120)로 함수 테이블에 존재하지 않는 함수 코드를 요청하고, FON 매니저(1120)로부터 함수 코드를 수신하는 기능을 수행할 수 있다. 함수 프로세서(1132)는 FON 매니저(1120)로부터 함수 삭제 메시지를 수신하고, 상기 함수 삭제 메시지에 포함되어 있는 함수 이름 정보와 함수 코드 정보를 함수 테이블에서 삭제하고 해당 함수에 대한 동적 바인딩 해제를 수행할 수 있다. 함수 프로세서(1132)는 FON 매니저(1120)로부터 함수 추가 메시지를 수신하고, 상기 함수 추가 메시지에 포함되어 있는 함수 이름 정보와 함수 코드 정보를 함수 테이블에 추가한다. 함수 프로세서(1132)는 DB(1134)에 저장되어 있는 함수 테이블을 관리하며, 사용자별 함수 이용 정보를 FON 매니저(1120)로 보고한다.
- [0077] FON 매니저(1120)는 실행 가능한 함수 이름과 함수 코드를 가지고 있는 함수 테이블(1124a)과 사용자(Subscriber) 정보 그리고 사용자 별 함수 이용에 따른 과금 정보(Billing Information)를 관리하는 회계 테이블(Accounting Table)(1124b)의 DB(1124)를 포함한다. 또한 FON 매니저(1120)는 함수 테이블(1124a)을 DB(1124)로부터 추출하여 FON 디바이스(1100) 또는 FON 노드(1130)로 전송하는 관리 프로세서(Management Processor)(1122)를 포함한다. 관리 프로세서(1122)는 FON 노드(1130)에서 실행할 수 있는 새로운 함수를 FON 매니저(1120)의 함수 테이블에 추가하고, 함수 추가 메시지를 FON 노드(1130)로 전송함으로써 FON 노드(1130)의 함수 테이블에 함수 이름 정보와 함수 코드 정보를 추가할 수 있다. 관리 프로세서(1122)는 FON 노드(1130)로 함수 삭제 메시지를 전송함으로써, FON 노드(1130)에 저장되어 있는 함수 테이블의 특정 함수 이름 정보와 함수 코드 정보를 삭제할 수 있다.
- [0078] 관리 프로세서(1122)는 FON 노드(1130)로부터 함수 코드 요청을 수신하고, DB(1124)의 함수 테이블(1124a)에서 함수 코드를 추출하여 전달할 수 있다. 관리 프로세서(1122)는 FON 노드(1130)로부터 사용자 별 함수 이용 정보를 수신하여 사용자 별 함수 이용에 대한 과금 정보를 산출하고, 해당 정보를 회계 테이블(1124b)에 저장/관리하는 역할을 수행한다.
- [0079] 관리 프로세서(1122)는 함수 배포 메시지를 FON 노드(1130)로 전송함으로써, FON 매니저(1120)의 함수 테이블을

FON 디바이스(1100) 또는 FON 노드(1130)로 배포하는 역할을 수행한다.

- [0080] FON 디바이스(1100)는 LTE, WiMAX, Wi-Fi, Bluetooth, ZigBee 등과 같은 무선 전송 기술과 Ethernet 등과 같은 유선 전송 기술을 사용하는 사용자 단말(User Equipment), 센서(Sensor), 데스크탑 컴퓨터, 태블릿(tablet) 컴퓨터, 노트북 등에 적용할 수 있다.
- [0081] FON 노드(1130)는 라우터, 스위치, Wi-Fi AP(Wi-Fi Access Point), eNB(evolved NodeB), PGW(Packet Data Network Gateway), SGW(Serving Gateway), HeNB(Home evolved NodeB) FGW(FemtoCell Gateway) 등에 적용할 수 있으며, 독립적인 장치로 존재하는 것도 가능하다.
- [0082] FON 매니저(1120)는 MME(Mobility Manager Entity), 오픈 플로우 제어부, SNMP 매니저(Simple Network Management Protocol Manager) 등에 적용할 수 있으며, 독립적인 장치로 존재하는 것도 가능하다.
- [0083] 도 12는 본 발명의 실시 예에 따른 스마트 패킷 구조도이다.
- [0084] 도 12의 스마트 패킷(1200)은 트레일러(Trailer)(1205), 페이로드(Payload)(1207), 함수 호출 필드(1210), 발신지 MAC 헤더(1230), 목적지 MAC 헤더(1240)를 포함한다. 함수 호출 필드(function invocation field)(1210)에는 IPv4/IPv6 주소(이하, "IP 주소"이라 칭함)(1250), TCP/UDP 포트 넘버(1260) 등을 포함할 수 있다. 즉, IP 프로토콜과 스택 구조를 고려하여 IP 네트워크와의 호환성을 제공하기 위해서 함수 호출 필드에 IPv4/IPv6 주소를 선택적으로 포함시킬 수 있다. IP 네트워크가 아닌 경우, IPv4/IPv6 주소를 포함하지 않는다.
- [0085] 함수 호출 필드(1210)는 호출하고자 하는 함수 이름 정보를 포함하며 복수 개의 함수 이름 정보를 포함할 수 있다. 또 호출하고자 하는 함수의 입력 파라미터와 출력 파라미터는 페이로드에 삽입될 수 있다.
- [0086] 도 13 및 도 14는 본 발명이 적용 가능한 네트워크 구성의 예시도이다.
- [0087] 도 13은 본 발명을 활용한 이동 통신 네트워크 구조이고, 도 14는 본 발명을 사용한 유무선 네트워크 구조를 나타낸 것이다.
- [0088] 도 13에서 나타낸 것과 같이, 센서(1312, 1314)와 사용자 단말(1310)에서 FON 디바이스가 적용되어 있다. 센서(1312, 1314)는 Bluetooth, ZigBee와 같은 전송 기술을 이용하여서 펌토셀 기지국(1316)과 통신하며, 사용자 단말(1310)은 LTE와 같은 전송 기술을 이용하여서 펌토셀 기지국(1316) 또는 eNB(1322)와 통신을 한다. 도 13에서의 펌토셀 기지국(1316)은 FON 노드가 적용되어 있고, 예컨대, Bluetooth/ZigBee/LTE/LTE-A 전송 기술을 모두 지원한다. eNB(1322)은 FON 노드가 적용되어 있고, 예컨대, LTE 또는 LTE-A 전송 기술을 지원한다.
- [0089] FGW(318), eNB(1322), SGW(serving gateway)(1324), PGW(packet data network gateway)(1326)는 FON 노드가 적용되어 있으며, MME(Mobile Mobility Entity)(1320)는 FON 매니저가 적용되어 있는 구조를 가지고 있다.
- [0090] 도 14에서는 Host 1(1410)에 FON 디바이스가 적용되어 있고, Wi-Fi와 같은 무선 전송 기술을 이용하여 Wi-Fi AP(1412)와 통신을 실시한다. Host 2(1424)는 Host 1(1410)과 동일하게 FON 디바이스가 적용되어 있고, Ethernet과 같은 유선 전송 기술을 이용하여 라우터 5(1422)와 통신을 한다. Wi-Fi AP(1412), 라우터 1~5(1414, 1416, 1418, 1420, 1422)는 FON 노드가 적용되어 있으며, FON 매니저(1426)는 상기에 설명한 것과 같이 독립적인 장치로 존재할 수 있다.
- [0091] 도 15는 본 발명의 실시 예에 따른 FON 디바이스의 세부 구조도를 나타낸다.
- [0092] 도 15는 FON 전용 디바이스인 경우의 세부 구조도이다.
- [0093] 도 15를 참조하면, FON 디바이스에는 MAC/PHY Layer(1520) 위에 함수 프로세서(FON processor)와 DB(1514)를 포함하는 FON Layer(Function-Oriented-Network Layer)(1510)가 위치하고, FON Layer (1510) 위에 FON 어플리케이션(Function-Oriented-Network Application)(1500)이 위치한다.
- [0094] FON 어플리케이션(1500)은 Socket 인터페이스를 이용하여 FON Layer(1510)로 FON 어플리케이션(1500)의 데이터를 전송할 수 있다.
- [0095] 도 16은 본 발명의 실시 예에 따른 FON 디바이스의 다른 세부 구조도를 나타낸다.
- [0096] 도 16은 FON 어플리케이션과 IP 어플리케이션을 동시에 지원하는 FON/IP 겸용 디바이스의 구조를 나타낸다.
- [0097] MAC/PHY Layer(1640) 위에 IP 기반 어플리케이션을 위한 IP Layer(1612)와 TCP/UDP Layer(1610)가 위치한다. IP Layer(1612), TCP/UDP Layer(1610)와 동등한 위치에 FON 프로세서(1632)와 DB(1634)를 포함하는 FON

Layer(1630)가 위치하게 된다.

- [0098] IP 기반 어플리케이션(1600)은 IP 기반 Socket 인터페이스를 이용하여 데이터를 전송한다. 반면에, FON 기반 어플리케이션(1620)은 FON 기반 Socket 인터페이스를 이용하여 데이터를 전송한다. FON 기반 Socket 인터페이스를 IP 기반의 기존 어플리케이션과의 호환성 및 개발 용이성을 위하여, IP 기반 Socket 인터페이스를 수정하여 구현할 수 있다.
- [0099] FON 디바이스에서의 IP 패킷 생성/전송 및 수신 과정은, 종래의 종단 간 어플리케이션 데이터 통신 방법과 동일하며, 도 5, 6, 7, 8, 9, 10에 나타나 있다. 이는 기존 서비스에 대해서는 호환성 차원에서 동일하게 지원하는 것이다.
- [0100] 도 17은 FON 어플리케이션에서의 스마트 패킷 생성/전송 및 수신 과정을 나타낸 흐름도이다.
- [0101] FON/IP 겸용 디바이스 또는 FON 전용 디바이스에서 스마트 패킷의 생성/전송 및 수신 시 수행된다. FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 1701 단계에서 socket() 함수를 호출하고, FON 통신을 위한 socket을 생성한다.
- [0102] FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 1703 단계에서 connection() 함수를 호출하여 FON 노드로 연결을 시도한다.
- [0103] FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 1705 단계에서 FON 노드로 데이터를 전송할 것인가 FON 노드로부터 데이터를 수신할 것인가를 결정한다.
- [0104] FON 노드로 데이터를 전송 할 경우, FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 1707 단계에서 spCreate() 함수를 호출하여 스마트 패킷을 생성한다. FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 스마트 패킷이 생성되면, 1709 단계에서 spAdd()함수를 호출하여 FON 노드에서 실행하고자 하는 함수를 스마트 패킷의 함수 호출 필드에 추가한다. 추가적으로 FON 노드에서 실행하고자 하는 함수를 스마트 패킷에 추가할 경우, spAdd()함수를 반복적으로 호출하여 FON 노드에서 실행하고자 하는 함수를 함수 호출 필드에 추가한다. 이와 같이, 여러 개의 함수를 넣어서 실행해 볼 수 있다. 여기서, 여러 가지의 함수의 일 예는 후술할 도 25를 참조하여 설명하기로 한다.
- [0105] 711 단계에서 함수 추가가 완료되면, FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 spTerminate() 함수를 호출하여 스마트 패킷 생성을 완료한다. 스마트 패킷 생성이 완료되면, write() 함수를 호출하여 FON Layer로 스마트 패킷 전송을 요청한다. 계속해서 전송할 데이터가 있을 경우, FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 spCreate(), spAdd(), spTerminate(), write() 함수를 반복적으로 호출하여 FON Layer로 데이터 전송을 요청한다. 1719 단계에서 더 이상 전송할 데이터가 없을 경우, FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 1721 단계에서 close() 함수를 호출함으로써 FON 노드와 설정되어 있던 연결을 종료한다.
- [0106] 1705 단계에서 FON 노드로부터 데이터를 수신하는 것으로 결정할 경우, FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 1717 단계에서 수신된 스마트 패킷의 페이로드를 확인하고, read() 함수를 호출하여 FON Layer에 FON 노드로부터 수신한 스마트 패킷에서 FON 어플리케이션의 데이터 추출 및 수신을 요청한다. 더 이상 데이터 수신을 원하지 않거나 더 이상 수신할 데이터가 없을 경우, FON/IP 겸용 디바이스 또는 FON 전용 디바이스는 close() 함수를 호출함으로써 FON 노드와 설정되어 있던 연결을 종료한다.
- [0107] 도 18은 FON 디바이스에서 FON 기반 어플리케이션의 소켓 제어 관련 프로그램 코드 예시를 나타낸 것이다.
- [0108] 참조번호 1800은 도 17의 1701 단계에서의 socket() 함수의 일 예를 나타낸다.
- [0109] 참조번호 1810은 도 17의 1703 단계에서의 연결 함수의 일 예를 나타낸다.
- [0110] 도 18은 도 6과 비교해 볼 때, 함수 기반(FON)의 소켓이며, 연결 함수에는 도 6에 비해, IP 주소(예컨대, 0.102.111.110)를 기재하였음을 알 수 있다.
- [0111] 도 19는 FON Layer에서의 스마트 패킷 전송 과정을 흐름도이다.
- [0112] FON Layer는 1901 단계에서 FON 어플리케이션으로부터 스마트 패킷을 수신한다.
- [0113] FON 어플리케이션으로부터 스마트 패킷을 수신하면, FON Layer는 1903 단계에서 부가적으로 FON Layer의 함수 프로세서가 FON 메시지 검증(Function-Oriented-Network Message Validation) 기능이 활성화 되어 있는지 확인한다. FON 메시지 검증은 FON 디바이스가 생성한 스마트 패킷이 FON 노드에서 오류 없이 네트워크 함수를 실행

할 수 있는지를 검사하는 것을 의미한다. FON 검증 메시지 기능이 활성화 되어 있지 않은 경우, FON Layer의 함수 프로세서는 1911 단계에서 스마트 패킷을 구성하고, 1913 단계에서 구성된 스마트 패킷을 Layer 2로 전달한다.

[0114] 만약, FON 검증 기능이 활성화 되어 있는 경우, FON Layer의 함수 프로세서는 1905 단계에서 FON Layer의 DB에 있는 함수 테이블을 이용하여 스마트 패킷이 FON 노드에서 오류 없이 네트워크 함수를 실행할 수 있는지 검사를 실시한다.

[0115] 도 20은 FON 디바이스의 DB에 저장되어 있는 함수 테이블의 구조를 나타낸다.

[0116] 함수 테이블의 구조(2000)는 예컨대, 함수 이름 필드와 함수 코드 필드로 구성된다. 함수 이름 필드에는 FON 노드에서 실행 가능한 함수의 이름을 포함하고, 함수 코드 필드에는 함수 이름에 대한 함수 소스 코드를 포함한다. 이때, FON 디바이스가 FON 노드 또는 다른 FON 디바이스의 요청으로 수행되어야 하는 함수는 소스 코드가 존재하지만, FON 디바이스가 FON 노드 또는 다른 FON 디바이스의 기능을 호출하는 경우는, 함수 이름만 존재하여, FON 메시지 생성 시 올바른 함수 호출인지를 검증하는 용도로 사용된다. FON Layer의 함수 프로세서는 1907 단계에서 FON 어플리케이션이 생성한 스마트 패킷의 함수 호출 필드에 작성되어 있는 함수가 함수 테이블에 존재하는지를 확인한다.

[0117] 함수 테이블에 스마트 패킷에 작성되어 있는 함수 이름이 존재하지 않을 경우, FON Layer의 함수 프로세서는 1915 단계에서 해당 스마트 패킷을 폐기하고, 1917 단계에서 FON 어플리케이션으로 에러 코드를 리턴한다.

[0118] 그러나 함수 테이블 안에 스마트 패킷에 작성되어 있는 함수 이름이 존재하는 경우, FON Layer의 함수 프로세서는 1909 단계에서 FON 노드에서 실행 가능한 함수로 판단할 수 있고, 스마트 패킷에 작성되어 있는 다음 함수 이름이 함수 테이블에 존재하는지에 대한 검사를 반복적으로 수행한다. FON Layer의 함수 프로세서는 스마트 패킷에 작성되어 있는 마지막 함수까지 함수 테이블에 존재하면, 11911 단계에서 마지막 함수까지 포함한 스마트 패킷을 생성하고, 1913 단계에서 생성한 스마트 패킷을 Layer 2로 전달한다.

[0119] 도 21은 FON Layer 에서의 스마트 패킷 수신 과정을 나타낸 흐름도이다.

[0120] FON Layer 은 2101 단계에서 Layer 2로부터 스마트 패킷을 수신한다.

[0121] FON Layer가 Layer 2부터 스마트 패킷을 전달 받으면, FON Layer의 함수 프로세서는 2103 단계에서 스마트 패킷의 함수 호출 필드를 추출하여 함수 이름을 탐색한다. FON Layer의 함수 프로세서는 2105 단계에서 탐색한 함수 이름이 PacketSend()인가를 판단한다. 만약, 탐색한 함수 이름이 PacketSend()가 아닐 경우, FON Layer의 함수 프로세서는 해당 함수를 실행한다.

[0122] 그러나 탐색한 함수 이름이 PacketSend()이면, FON Layer의 함수 프로세서는 2107 단계에서 스마트 패킷의 페이로드를 탐색한다. FON Layer의 함수 프로세서는 2111 단계에서 탐색한 페이로드에서 FON 어플리케이션 데이터를 추출하여 FON 어플리케이션으로 전달한다. FON Layer의 함수 프로세서는 2113 단계에서 스마트 패킷의 함수 호출 필드에 작성되어 있는 모든 함수에 대해서 이와 같은 과정을 반복하고, 마지막 함수까지 도달하면 스마트 패킷 수신 과정을 마치게 된다.

[0123] 도 22는 FON 노드에서의 스마트 패킷 수신 처리 과정을 나타낸 흐름도이다.

[0124] FON 노드의 함수 프로세서는 2201 단계에서 FON 디바이스 또는 FON 노드로부터 스마트 패킷을 수신한다.

[0125] FON 디바이스 또는 FON 노드로부터 스마트 패킷을 수신하면, FON 노드의 함수 프로세서는 2203 단계에서 수신된 스마트 패킷에서 스마트 패킷의 함수 호출 필드를 추출한다. 그리고 FON 노드의 함수 프로세서는 2205 단계에서 실행할 함수의 이름을 탐색한다. 이때, FON 노드의 함수 프로세서는 2207 단계에서 DB에 저장되어 있는 함수 테이블을 검색한다. FON 노드의 함수 프로세서는 2209 단계에서 함수 테이블로부터 함수 코드를 찾아서 실행한다. 함수 실행 방법은 다양한 방법이 존재할 수 있으며, FON 노드의 함수 프로세서는 동적 바인딩을 이용한 함수 실행을 기본으로 한다. FON 노드의 함수 프로세서는 DB에 저장되어 있는 함수 테이블에 존재하는 함수 코드들을 컴파일하여 실행 파일을 생성하고, 생성된 실행 파일들을 DB에 저장한다. FON 노드의 함수 프로세서는 스마트 패킷의 함수 호출 필드에 명시되어 있는 함수에 대한 실행 파일을 DB로부터 불러와 실행한다. 또 FON 노드의 함수 프로세서는 실행되지 않는 함수의 동적 바인딩을 해제할 수 있다.

[0126] 이후, FON 노드의 함수 프로세서는 2211 단계에서 실행이 완료된 함수에 대한 통계치를 업데이트하고, 2213 단계에서 모든 함수의 실행을 마쳤는가를 판단한다. 모든 함수의 실행을 마친 경우, FON 노드의 함수 프로세서는

동작을 종료한다. 그러나 모든 함수의 실행을 마치지 않은 경우, FON 노드의 함수 프로세서는 2205 단계로 귀환한다.

[0127] 도 23은 FON 노드의 DB에 저장되어 있는 함수 테이블의 구조 및 예시도이다.

[0128] 도 23을 참조하면, 함수 테이블(2300)은 함수 이름 필드, 함수 코드 필드, 통계 필드로 구성된다.

[0129] 함수 이름 필드에는 FON 노드에서 실행 가능한 함수의 이름을 포함한다.

[0130] 하고, 함수 코드 필드에는 함수 이름에 대한 함수 소스 코드를 포함한다.

[0131] 통계 필드는 함수의 호출 횟수를 포함한다.

[0132] 함수 실행이 완료되면, FON 노드의 함수 프로세서는 실행이 완료된 함수에 대한 함수 테이블의 통계 필드를 업데이트한다. 스마트 패킷에 명시되어 있는 모든 함수에 대해서 이와 같은 과정을 완료하면, FON 노드의 스마트 패킷 수신 처리 과정을 마치게 된다.

[0133] 추가적으로, FON 노드의 함수 프로세서는 스마트 패킷을 보낸 FON 디바이스를 식별하고, 함수 이용 정보를 FON 매니저로 전송한다. FON 디바이스를 식별하는 방법은 다양한 방법이 존재할 수 있으며, 4G/5G MAC 주소, Wi-Fi 주소, Bluetooth MAC 주소 등과 ISMI(International Mobile Subscriber Identity)와 같은 고유한 사용자 ID를 이용하여서 식별할 수 있다. 또 FON 디바이스가 호출한 함수 이름과 함수 호출 횟수를 산출한다.

[0134] 사용자 식별 정보와 함수 이용 정보는 도 24에 나타나 있는 함수 이용 정보 보고 메시지를 이용하여 FON 매니저로 전송한다.

[0135] 도 24는 함수 이용 정보 보고 메시지의 일 예를 도시한 포맷이다.

[0136] 함수 이용 정보 보고 메시지는 FON 디바이스에서 FON 메니저로 전송되는 메시지이다. 도 24를 통해 예컨대, ISMI를 이용하여 FON 디바이스를 식별함을 알 수 있다. 함수 이용 정보 보고 메시지는 함수 호출 정보를 포함한다. 함수 호출 정보를 추출하여 함수 호출 횟수에 따른 과금 비용을 산출할 수 있다.

[0137] 도 25는 FON 노드가 지원하는 기본 함수에 대한 리스트의 예시를 나타낸 도면이다.

[0138] 도 25는 도 17의 1709 단계에서 이용되는 함수의 일 예를 나타낸다.

[0139] FON 매니저로부터 함수를 다운 받아 FON 노드의 함수 테이블을 업데이트할 수 있고, 새로운 함수를 추가할 수도 있다. 또 FON 매니저로부터 함수 삭제 메시지를 수신하여, 함수 삭제 메시지에 해당하는 함수를 함수 테이블에서 삭제할 수도 있다. 본 발명에서는 이후 예시된 일부 함수의 구체적 동작을 기술하지만, 추가적인 발명으로 제시할 수 있다.

[0140] FON 매니저는 FON 매니저의 DB에 저장되어 있는 함수 테이블을 FON 디바이스와 FON 노드로 배포할 수 있는데, 배포 과정은 다음과 같다.

[0141] FON 매니저가 FON 디바이스 또는 FON 노드로 함수 테이블을 배포할 경우, FON 매니저의 관리 프로세서는 DB에 저장되어 있는 함수 테이블을 추출하여 함수 배포 메시지를 생성하여 FON 노드로 생성된 함수 배포 메시지를 전송함으로써, FON 노드로 FON 매니저의 함수 테이블 정보를 배포하게 된다.

[0142] 패킷 전달할 경우, 이용되는 함수는 PacketSend(), PacketInsertField(), PacketRemoveField(), PacketAddField(), PacketPort()를 포함한다.

[0143] PacketSend()는 패킷을 요청한 곳으로 전송하는데 필요한 함수이고, PacketInsertField()는 패킷의 특정 위치(필드)에 요청된 값을 삽입하는데 필요한 함수이고, PacketRemoveField()는 패킷의 특정 위치(필드)의 값을 제거하는데 필요한 함수이고, PacketAddField()는 패킷의 마지막 부분에 요청된 값을 삽입하는데 필요한 함수이고, PacketPort()는 패킷이 전송될 포트 번호(들)을 읽는데 필요한 함수이다.

[0144] 시스템 정보 조회 시에는 SystemRead() 함수를 이용하고, SystemRead() 함수는 FON 노드의 시스템 정보를 읽어 오는데 필요한 함수이다.

[0145] 사용자 정보 조회 시에는 SubscriberRRead() 함수를 이용하고, SubscriberRRead() 함수는 FON 노드에 접속한 사용자/단말 정보를 읽어오는데 필요한 함수이다.

[0146] 사용자 저장 공간을 사용시 필요한 함수는

StorageAvailability(),StorageCreate(),StorageRead(),StorageWrite(),StorageRemove() 등을 포함한다.

- [0147] StorageAvailability() 함수는 FON 노드에 사용자가 요청한 변수가 있는지를 확인하는데 필요한 함수이다.
- [0148] StorageCreate() 함수는 FON 노드에 사용자가 요청한 변수를 생성하는데 필요한 함수이다.
- [0149] StorageRead() 함수는 FON 노드에 사용자가 생성한 변수에 값을 읽는데 필요한 함수이다.
- [0150] StorageWrite() 함수는 FON 노드에 사용자가 생성한 변수에 값을 저장하는데 필요한 함수이다.
- [0151] StorageRemove() 함수는 FON 노드에 사용자가 요청한 변수를 제거하는데 필요한 함수이다.
- [0152] 사용자 기능 수행시 필요한 함수는 MathRandom() 함수와 MathGetMax() 함수를 포함한다.
- [0153] MathRandom() 함수는 입력 범위 내에서 Random 함수로 결과 값을 산출하는데 필요한 함수이다.
- [0154] MathGetMax() 함수는 입력 값을 비교하여 최대값을 리턴하는데 필요한 함수이다.
- [0155] 도 26은 본 발명의 실시 예에 따른 함수 배포 메시지 구조도 및 예시도이다.
- [0156] 함수 배포 메시지 구조도에서 메시지의 목적을 나타내는 명령(Command) 속성은 "Publication"속성값을 갖고, 함수 이름, 함수 코드, 입력 파라미터(input parameter), 출력 파라미터(output parameter) 정보와 배포하고자 하는 장치의 타겟(Target) 정보를 포함한다. 도 26에서의 타겟 정보는 FON 디바이스와 FON 노드가 된다.
- [0157] 도 26에 나타난 바와 같이, 작성된 함수 배포 메시지를 수신한 FON 노드는 Target 속성에 "FON Device"이라는 속성값이 추가로 포함되어 있을 경우, FON 노드와 통신하는 모든 FON 디바이스에게 함수 배포 메시지를 전송함으로써, FON 매니저가 FON 디바이스로 함수 테이블 정보를 배포하게 된다.
- [0158] FON 매니저는FON 노드에 배포된 함수들 중에서 특정 함수를 삭제하는 명령을 내릴 수 있는데, 함수 삭제 과정은 다음과 같다.
- [0159] 망운영자가 FON 노드의 함수 테이블에 존재하는 함수들 중에서 특정 함수를 삭제하고자 할 때, FON 매니저의 관리 프로세서는 함수 삭제 메시지를 생성하여 FON 노드로 전송함으로써, FON 노드에 존재하는 특정 함수를 삭제하게 된다.
- [0160] 도 27은 본 발명의 실시 예에 따른 함수 삭제 메시지 구조도 및 예시도를 나타낸다.
- [0161] 메시지의 목적을 나타내는 명령 속성은 "Delete"속성값을 갖고, 삭제하고자 하는 함수 이름, 함수 코드, 입력 파라미터, 출력 파라미터 정보와 함수 삭제를 수행할 장치의 타겟 정보를 포함한다. 타겟 정보는 예컨대, FON 디바이스와 FON 노드를 포함한다.
- [0162] FON 매니저는 함수 삭제 메시지를 FON 노드로 전송하고, 이를 수신한 FON 노드가 함수 삭제 메시지에 명시되어 있는 함수 정보를 함수 테이블에서 삭제함으로써, FON 매니저는 FON 노드의 함수 테이블에 존재하는 특정 함수를 삭제하게 된다.
- [0163] FON 매니저는 망운영자나 3rd-Party 사업자에 의해서 작성된 새로운 함수를 FON 노드로 전송하여 추가할 수 있는데, 새로운 함수 전송 및 추가 과정은 다음과 같다.
- [0164] 망운영자나 3rd-Party 사업자가 FON 노드로 새로운 함수를 추가하고자 할 때, FON 매니저의 관리 프로세서는 함수 추가 메시지를 생성하여 FON 노드로 전송함으로써, FON 노드의 함수 테이블에 새로운 함수를 추가할 수 있다.
- [0165] 도 28은 본 발명의 실시 예에 따른 함수 추가 메시지 구조도 및 예시도를 나타낸다.
- [0166] 메시지의 목적을 나타내는 명령 속성은 "Add"속성값을 갖고, 삭제하고자 하는 함수 이름, 함수 코드, 입력 파라미터, 출력 파라미터 정보와 함수 추가를 수행할 장치의 타겟 정보를 포함한다.
- [0167] 타겟 정보는 예컨대, FON 디바이스와 FON 노드를 포함한다.
- [0168] FON 매니저는 함수 추가 메시지를 FON 노드로 전송하고, 이를 수신한 FON 노드는 함수 추가 메시지에 명시되어 있는 함수 정보를 FON 노드의 함수 테이블에 추가함으로써, FON 매니저는 FON 노드의 함수 테이블에 새로운 함수를 추가하게 된다.
- [0169] FON 매니저는 FON 노드로부터 함수 이용 정보 메시지를 수신하면, 함수 이용에 따른 과금 정보를 관리하는데,

과금 정보 관리 과정은 다음과 같다.

- [0170] FON 노드로부터 도 24와 같은 함수 이용 정보 메시지를 수신하면, FON 매니저의 관리 프로세서는 DB에 저장되어 있는 회계 테이블을 관리한다.
- [0171] 도 29는 본 발명의 실시 예에 따른 회계 테이블 구조도를 나타낸다.
- [0172] FON 매니저의 관리 프로세서는 수신한 함수 이용 정보 보고 메시지에서 FON 노드 정보를 추출하여 사용자를 식별하고, 회계 테이블(2900)의 가입자 속성에 식별된 사용자 정보를 저장한다. 그리고 FON 매니저의 관리 프로세서는 함수 이용 정보 보고 메시지에서 함수 호출 정보를 추출하여 함수 호출 횟수에 따른 과금 비용을 산출한다. 산출된 과금 비용은 회계 테이블의 과금 정보 속성에 저장한다. 이와 같은 과정은 FON 노드로부터 함수 이용 정보 보고 메시지를 수신할 때마다 FON 매니저의 관리 프로세서에 의해 수행하여 사용자 별 네트워크 함수 이용에 따른 과금 정보를 관리하게 된다.
- [0173] 도 26, 도 27, 도 28은 본 발명의 예시를 위한 예로서, 다른 언어와 표시로도 구현은 가능하다.
- [0174] 본 발명을 활용한 시나리오의 예로서 본 발명에서는 다음과 같은 4가지 사례를 고려하였으며, 추가적인 함수 제안에 따른 신규 서비스도 가능하다.
- [0175] 첫째, 이기종 망 상에서의 D2D 기반 IoT 통신 환경의 패킷 전송 시나리오다.
- [0176] 도 30은 일반적인 4G/5G 이동통신 네트워크에서의 센서와 사용자 단말 간의 데이터 통신의 예시를 나타낸 시나리오이다.
- [0177] 웹토셀 기지국(3006)이 Bluetooth 전송 기술과 LTE 또는 5G 무선 전송 기술을 지원할 때, 동일한 웹토셀 기지국(3006)에 속해 있는 센서(3002)와 사용자 단말(3004)은 직접 통신을 하지 않고, PGW와 같은 IP Anchor(3008)와 DNS 서버(3010)를 거쳐, 센서(3002)와 사용자 단말(3004)간 IoT(Internet of Things) 통신을 실시한다. 먼저, 사용자 단말(3004)은 DNS 요청 메시지를 웹토셀 기지국(3006), IP Anchor 장치(3008)를 거쳐 DNS 서버(3010)로 전송한다. DNS 서버(3010)는 DNS 응답 메시지를 IP Anchor(3008), 웹토셀 기지국(3006)을 거쳐 사용자 단말(3004)로 전송한다. DNS 요청 및 응답 과정을 통해서, 사용자 단말(3004)은 센서의 IP 주소 정보를 획득하게 된다. 사용자 단말은 획득한 IP 주소를 이용하여 동일한 기지국 내에 있는 센서로 TCP 연결 설정을 시도한다. TCP SYNC 메시지는 웹토셀 기지국(3006), IP Anchor(3008)로 전송되고, IP Anchor(3008)는 센서(3002)가 속해 있는 웹토셀 기지국(3006)으로 TCP SYNC 메시지를 보낸다. 웹토셀 기지국(3006)은 TCP SYNC 메시지를 보낼 센서를 식별하고 Bluetooth 전송 기술을 이용하여 TCP SYNC 메시지를 해당 센서(3002)로 전송한다. TCP SYNC 메시지를 수신한 센서(3002)는 TCP SYNC 메시지에 대한 TCP ACK 메시지를 생성하여 웹토셀 기지국(3006)으로 전송한다. TCP ACK 메시지 수신한 웹토셀 기지국(3006)은 IP Anchor(3008)로 TCP ACK 메시지를 보내고, IP Anchor(3008)는 사용자 단말(3004)이 속해 있는 웹토셀 기지국(3006)으로 TCP ACK 메시지를 전송한다. 웹토셀 기지국(3006)은 TCP ACK 메시지를 보낼 사용자 단말(3004)을 식별하고 LTE 또는 5G 무선 전송 기술을 이용하여 TCP ACK 메시지를 해당 사용자 단말(3004)로 전송한다. 사용자 단말(3004)과 센서의 어플리케이션 데이터, TCP 연결 설정 종료 메시지 등의 전송 과정도, TCP SYNC, TCP ACK 전송 과정과 동일하게 수행된다. 이와 같은 종래의 전송 기술은 이동통신 네트워크 내에 시그널링을 증가시키고, IP Anchor(3008)와 DNS 서버(3010)의 부하와 코어 네트워크의 트래픽을 증가시킨다.
- [0178] 도 31은 도 30의 전송 시나리오를 본 발명을 적용했을 때의 사용자 단말과 센서 간의 데이터 통신 과정을 나타낸 시나리오이다.
- [0179] 사용자 단말은 3101 단계에서 스마트 패킷의 함수 호출 필드에 PacketSend() 함수를 명시하고 센서로 보낼 데이터와 센서의 Bluetooth MAC 주소를 페이로드에 담아 웹토셀 기지국으로 보낸다. 웹토셀 기지국은 FON 노드를 포함하고 있어, 스마트 패킷을 처리할 수 있다. FON 노드를 포함하는 웹토셀 기지국은 페이로드에서 PacketSend() 함수의 입력 파라미터로 사용되는 센서로 보낼 데이터와 센서의 Bluetooth MAC 주소를 추출하여 PacketSend() 함수의 입력 파라미터로 사용하고, 참조번호 3107과 같은 스마트 패킷의 함수 호출 필드에 명시되어 있는 PacketSend() 함수를 실행한다. PacketSend() 함수의 실행 결과로, 웹토셀 기지국이 3103 단계에서 LTE 또는 5G 전송 기술을 이용하여 수신한 사용자 단말의 스마트 패킷을 Bluetooth 전송 기술을 이용하여 센서로 전송한다. 추가적으로 웹토셀 기지국은 사용자 단말로 스마트 패킷이 정상적으로 처리되었음을 알릴 수 있다. 이와 같은 본 발명의 실시 예는 종래의 기술보다 이동통신 네트워크 내에서 시그널링을 감소시키고, IP Anchor와 DNS 서버 같은 장치의 부하를 분산시키고, 코어 네트워크에 발생하는 트래픽을 감소시킬 수 있다. 또한 본 발명의

실시 예는 무선 링크의 부하를 감소시킴으로써 무선 자원의 사용 효율을 증가시킬 수 있다.

[0180]

둘째, 종단 간 패킷 전송 경로 정보 수집 시나리오이다.

[0181]

도 32는 일반적인 종단 간 패킷 전송 경로 정보 수집 예시를 나타낸 것이다. 네트워크 장치는 스위치 또는 라우터 등과 같은 장비로 구성된다.

[0182]

일반적인 네트워크 구조에서는 DDoS(Distributed Denial of Service) 공격 등에 대비하기 위해서, 보안상의 이유로 네트워크 구조 및 경로 정보를 비공개로 설정한다. 이와 같은 일반적인 네트워크 구조는 사용자 및 망관리자 입장에서 네트워크의 상황과 문제점을 정확하게 파악하기 어렵다는 단점이 있다.

[0183]

도 33은 본 발명의 실시 예에 따른 종단 간 패킷 전송 경로 정보 수집 시나리오를 나타낸 것이다.

[0184]

네트워크 장치는 FON 노드로 구성된다. Host 1과 Host 2는 FON 디바이스로 구성된다. FON 노드를 포함하는 네트워크 장치들은 FON 매니저로부터 네트워크 모니터링 함수를 다운로드 받아 동적 바인딩을 실시하고, 사용자 또는 망관리자는 네트워크 모니터링 함수에 대한 실행 정보를 스마트 패킷에 포함시켜 FON 노드로 전송한다. 스마트 패킷을 수신한 FON 노드는 스마트 패킷에 명시되어 있는 네트워크 모니터링 함수를 실행하고, 측정된 네트워크 모니터링 정보를 DB에 저장하고, 사용자 또는 망관리자에게 측정된 네트워크 모니터링 정보를 전송한다. 이와 같은 본 발명의 동작 특성은, 사용자 및 관리 입장에서 네트워크의 상황과 문제점을 정확하게 파악하는 것을 가능하게 한다.

[0185]

셋째, 생체모방 개미 집단 알고리즘을 이용한 패킷 전송 최적화 시나리오다.

[0186]

도34는 본 발명의 실시 예에 따른 생체모방 개미 집단 알고리즘을 활용한 패킷 전송 최적화 예시를 나타낸 도면이다.

[0187]

Host 1과 Host 2는 FON 디바이스로 동작하고, 네트워크 장치들은 FON 노드로 동작한다. 네트워크 장치들은 FON 매니저로부터 생체모방 개미 집단 알고리즘 관련 함수들을 다운로드 받아 동적 바인딩을 수행한다. 이와 같은 상황에서 Host 1는 생체모방 개미 집단 알고리즘을 이용하여 Host 1에서 Host 2로의 데이터 전송 경로 중에서 최적화된 전송 경로를 결정하는 함수 실행 정보를 스마트 패킷의 함수 호출 필드에 작성하여 FON 노드가 동작하는 네트워크 장치로 전송한다. 스마트 패킷을 수신한 네트워크 노드는 스마트 패킷의 함수 호출 필드에 명시되어 있는 생체모방 개미 집단 알고리즘 함수들을 실행한다. 함수 실행이 완료되면, 스마트 패킷은 임의의 이웃한 네트워크 장치로 보내지고 해당 네트워크 장치에도 마찬가지로 스마트 패킷에 명시되어 있는 생체모방 개미 집단 알고리즘 함수들을 실행하게 된다. 이와 같은 스마트 패킷 전송 및 생체모방 개미 집단 알고리즘 함수 실행 과정이 반복적으로 수행되어, Host 2에 스마트 패킷이 도달하게 된다. 생체모방 개미 집단 알고리즘 함수들을 명시하고 있는 스마트 패킷이 계속해서 네트워크를 통과하게 되면, FON 노드가 동작하는 네트워크 장치에서는 생체모방 개미 알고리즘에 기반한 최적화된 전송 경로를 결정할 수 있게 된다. 본 발명을 네트워크에 적용할 경우, 다양한 네트워크 프로토콜 및 전송 기술을 도입하여 최적화된 전송 서비스를 제공하는 것이 가능하다.

[0188]

넷째, 3rd-Party 웹 서비스 또는 미디어 서비스 가속화 시나리오이다.

[0189]

도 35는 종래의 3rd-Party 웹 서비스 또는 미디어 서비스 가속화 예시도를 나타낸 것이다.

[0190]

사용자 단말은 웹 페이지를 구성하는 3개의 객체(object)를 서로 다른 HTTP 서버에 요청한다. 사용자 단말은 웹 페이지를 구성하는 3개의 객체에 대해서 HTTP 요청 메시지를 생성하여 기지국(BS)으로 전송한다. 사용자 단말의 HTTP 요청 메시지를 수신한 기지국은 서로 다른 HTTP 서버로 HTTP 요청 메시지를 전송한다. HTTP 요청 메시지는 이동 통신 네트워크를 거쳐 3rd-Party의 HTTP 서버 1, HTTP 서버 2, HTTP 서버 3에 도달하게 된다. HTTP 요청 메시지를 수신한 3rd-Party의 HTTP 서버 1, HTTP 서버 2, HTTP 서버 3은 웹 페이지를 구성하는 객체에 대한 HTTP 응답 메시지를 생성하여 사용자 단말로 전송한다. 3rd-Party의 HTTP 서버 1, HTTP 서버 2, HTTP 서버 3에서 전송된 HTTP 응답 메시지는 이동통신 네트워크를 거쳐 사용자 단말이 속한 기지국으로 전송되고, 기지국은 사용자 단말로 HTTP 응답 메시지를 전송하게 된다. 이와 같은 종래 기술의 웹 서비스 가속 방식은 인터넷에 3rd-Party 웹 서버를 배치하기 보다는 이동통신 네트워크 내에 웹 서버를 배치함으로써 이동통신 내에 있는 사용자 단말에게 가속화된 웹 서비스를 제공하는 방식이다. 이는 물리적으로 사용자 단말과 3rd-Party의 웹 서버 거리를 줄임으로써, 웹 서비스를 가속화할 뿐, 웹 서비스 자체를 가속화 하지는 못한다.

[0191]

도 36은 본 발명의 실시 예에 따른 3rd-Party 웹 서비스 또는 미디어 서비스 가속화 예시도를 나타낸다.

[0192]

사용자 단말은 FON 디바이스로 동작하고, 기지국은 FON 노드로 동작하고 있다.

- [0193] 기지국(FON 노드)는 FON 매니저로부터 3rd-Party가 제공한 웹 서비스 및 전송 최적화 기술을 구현한 함수를 다운로드 받아 DB에 저장하고 동적 바인딩하여 함수가 실행될 수 있도록 준비한다.
- [0194] 사용자 단말(FON 디바이스)는 3rd-Party의 웹 페이지를 요청하는 스마트 패킷을 생성하여 기지국으로 전송한다. 이때, 웹 페이지를 구성하는 서로 다른 3개의 객체에 대한 3개의 스마트 패킷을 생성하여 전송하는 것이 아니라, 1개의 스마트 패킷에 3개의 객체를 요청하도록 함수 호출 필드를 작성하여 기지국으로 전송한다. 상기 스마트 패킷을 수신한 기지국은 스마트 패킷에 명시되어 있는 함수를 실행하여 웹 페이지를 구성하는 3개의 객체에 대한 HTTP 요청 메시지를 생성하여 3rd-Party의 HTTP 서버 1, HTTP 서버 2, HTTP 서버 3로 전송한다. 또 기지국이 수신한 스마트 패킷에 코어 네트워크에서의 최적화된 전송 함수를 포함하고 있다면, 3rd-Party의 HTTP 서버로 HTTP 요청 메시지를 전송할 수 있게 된다. HTTP 요청 메시지를 수신한 3rd-Party의 HTTP 서버는 웹 페이지를 구성하는 객체에 대해서 HTTP 응답 메시지를 생성하여 기지국으로 전송한다. 3rd-Party의 HTTP 서버로부터 HTTP 응답 메시지를 수신한 기지국은 HTTP 응답 메시지에 포함되어 있는 객체가 멀티미디어 콘텐츠라면, 사용자 단말의 하드웨어 특성에 맞도록 콘텐츠를 압축하거나 크기를 줄이는 등의 콘텐츠 최적화를 수행하여 사용자 단말로 전송한다.
- [0195] 이와 같이, 본 발명의 실시 예는 무선 구간에서 전송되는 메시지의 개수를 줄여 무선 구간에서의 지연을 최소화하고, 3rd-Party의 콘텐츠를 최적화할 수 있으며, 코어 네트워크 등의 유선 구간에서 전송을 최적화함으로써, 3rd-Party 웹 서비스 또는 미디어 서비스를 가속하는 것이 가능하다.
- [0196] 본 발명의 효과는 다음과 같다.
- [0197] 첫째, IP 네트워크 프로토콜 외에 새로운 프로토콜의 도입이 가능하다. 네트워크에 새로운 네트워크 프로토콜의 기능을 함수로 구현하여 FON 노드들에 배포하고, 스마트 패킷을 통해서 기능을 호출함으로써, 망운영자/사용자/인터넷서비스제공자가 자신만의 새로운 네트워크 프로토콜을 만들고 망에 도입하는 것이 가능하다.
- [0198] 둘째, 망운영자와 사용자/인터넷서비스제공자가 직접 라우팅 정책을 정의할 수 있다. 직접 본인이 구현한 함수에 의하여 자신이 결정한 패킷 경로로 트래픽을 전송하는 것을 지원한다.
- [0199] 셋째, 패킷의 전송 외에 패킷이 요구하는 부가 기능을 FON 노드에서 지원할 수 있다.
- [0200] 종래 SDN/오픈 플로우 의 문제점에 대한 본 기술의 해결 사항은 다음과 같다.
- [0201] 첫째, 종래 라우터 기술에서 해결하지 못한 문제의 해결로서, 망사업자(망운영자) 및 사용자/인터넷서비스제공자가 패킷이 전달되는 경로를 결정하는 플로우차원의 프로그래머빌리티를 지원한다. 앞서 종래 라우터의 기술 문제점 해결에서 언급했듯이, 네트워크에 새로운 네트워크 프로토콜의 기능을 함수로 구현하여 FON 노드들에 배포하고, 스마트 패킷을 통해서 기능을 호출함으로써, 망운영자/사용자/인터넷서비스제공자가 자신만의 새로운 네트워크 프로토콜을 만들고 망에 도입하는 것이 가능하다. 또한, 직접 본인이 구현한 함수에 의하여 자신이 결정한 패킷 경로로 트래픽을 전송하는 것을 지원한다.
- [0202] 둘째, 종래 라우터 기술에서 해결하지 못한 문제의 해결로서, 망사업자 및 사용자/인터넷서비스제공자가 자신이 원하는 기능을 네트워크 안에 구현하고 이를 서비스에서 활용하는 기술이 가능하다. 앞서 종래 라우터의 기술 문제점 해결에서 언급했듯이, 패킷의 전송 외에 패킷이 요구하는 부가 기능을 FON 노드에서 지원할 수 있다.
- [0203] 셋째, 종래 라우터 기술에서 해결하지 못한 문제의 해결로서, 망사업자 및 사용자/인터넷서비스제공자가 네트워크의 동적/정적 상태를 구체적으로 파악하는 것이 가능하다. 본 기술의 활용 시나리오에서 나타난 것처럼, FON 노드의 부가 기능 구현을 통해서 네트워크의 동적/정적 상태를 스마트 패킷을 통해서 파악하는 것이 가능하다.
- [0204] 넷째, SDN/오픈 플로우 기술의 등장으로 새롭게 발생한 문제의 해결로서, 패킷의 헤더 정보를 분석하기 위한 DPI(Deep Packet Inspection) 기능의 구현이 단순화된다. 복잡한 패킷 구조를 일일이 분석하는 부하가 필요하지 않고, 단순하게 함수 호출 부분만 분석하는 수준으로 복잡도가 낮아진다.
- [0205] 다섯째, SDN/오픈 플로우 기술의 등장으로 새롭게 발생한 문제의 해결로서, 중앙집중화 된 SDN 컨트롤러와 오픈 플로우 스위치 간의 인터페이스 부하가 발생하지 않는다. 기존 SDN/오픈 플로우의 경우 컨트롤러만이 지능적인 처리를 하지만, FON 노드에서는 자체적으로 지능적인 패킷 처리가 가능하기에 FON 노드 매니저와의 인터페이스 부하가 SDN 대비 크지 않다.
- [0206] 한편 본 발명의 일 실시 예에 따른 이동 통신 시스템에서 패킷 생성 방법 및 장치 는 하드웨어, 소프트웨어 또는 하드웨어 및 소프트웨어의 조합의 형태로 실현 가능하다는 것을 알 수 있을 것이다. 이러한 임의의 소프트웨

어는 예를 들어, 삭제 가능 또는 재기록 가능 여부와 상관없이, ROM 등의 저장 장치와 같은 휘발성 또는 비휘발성 저장 장치, 또는 예를 들어, RAM, 메모리 칩, 장치 또는 집적 회로와 같은 메모리, 또는 예를 들어 CD, DVD, 자기 디스크 또는 자기 테이프 등과 같은 광학 또는 자기적으로 기록 가능함과 동시에 기계(예를 들어, 컴퓨터)로 읽을 수 있는 저장 매체에 저장될 수 있다. 본 발명의 일 실시 예에 따른 이동 통신 시스템에서 패킷 생성 방법은 제어부 및 메모리를 포함하는 컴퓨터 또는 휴대 단말에 의해 구현될 수 있고, 상기 메모리는 본 발명의 실시 예들을 구현하는 지시들을 포함하는 프로그램 또는 프로그램들을 저장하기에 적합한 기계로 읽을 수 있는 저장 매체의 한 예임을 알 수 있을 것이다.

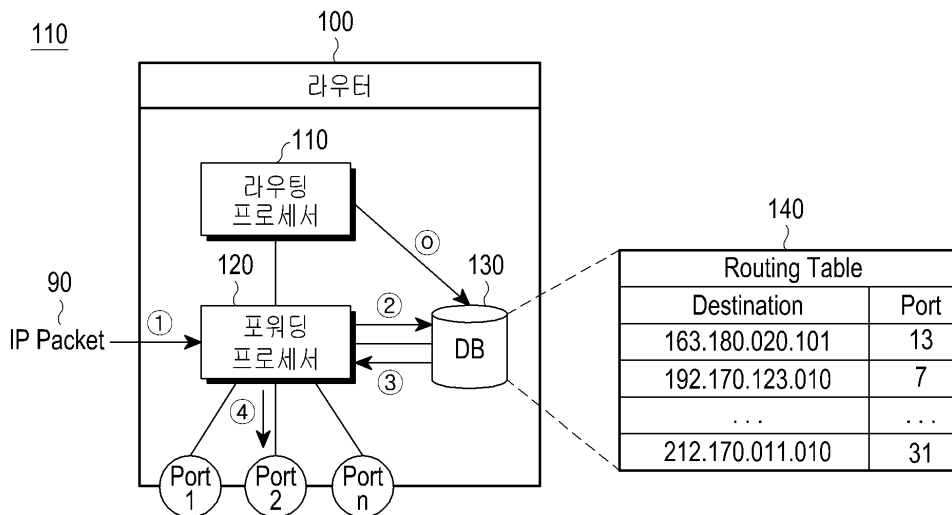
[0207] 따라서, 본 발명은 본 명세서의 임의의 청구항에 기재된 장치 또는 방법을 구현하기 위한 코드를 포함하는 프로그램 및 이러한 프로그램을 저장하는 기계(컴퓨터 등)로 읽을 수 있는 저장 매체를 포함한다. 또한, 이러한 프로그램은 유선 또는 무선 연결을 통해 전달되는 통신 신호와 같은 임의의 매체를 통해 전자적으로 이송될 수 있고, 본 발명은 이와 균등한 것을 적절하게 포함한다.

[0208] 또한 본 발명의 실시 예에 따른 이동 통신 시스템에서 패킷 생성 장치는 유선 또는 무선으로 연결되는 프로그램 제공 장치로부터 상기 프로그램을 수신하여 저장할 수 있다. 상기 프로그램 제공 장치는 상기 프로그램 처리 장치가 기 설정된 이동 통신 시스템에서 패킷 생성 방법을 수행하도록 하는 지시들을 포함하는 프로그램, 이동 통신 시스템에서 패킷 생성 방법에 필요한 정보 등을 저장하기 위한 메모리와, 상기 그래픽 처리 장치와의 유선 또는 무선 통신을 수행하기 위한 통신부와, 상기 그래픽 처리 장치의 요청 또는 자동으로 해당 프로그램을 상기 송수신 장치로 전송하는 제어부를 포함할 수 있다.

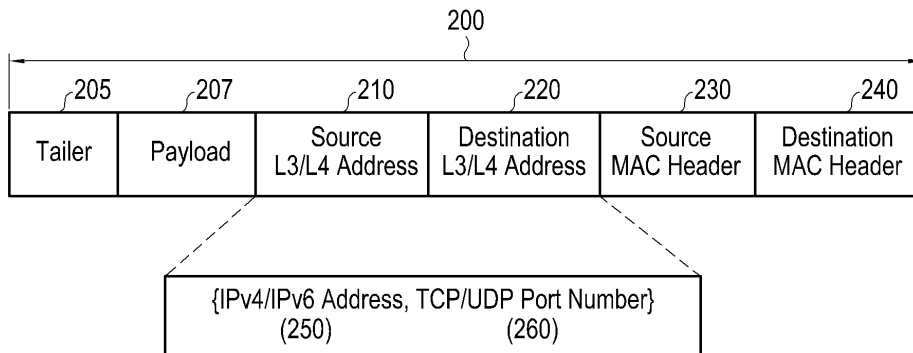
[0209] 한편 본 발명의 상세한 설명에서는 구체적인 실시 예에 관해 설명하였으나, 본 발명의 범위에서 벗어나지 않는 한도 내에서 여러 가지 변형이 가능함은 물론이다. 그러므로 본 발명의 범위는 설명된 실시 예에 국한되어 정해져서는 안되며 후술하는 특허청구의 범위뿐만 아니라 이 특허청구의 범위와 균등한 것들에 의해 정해져야 한다.

도면

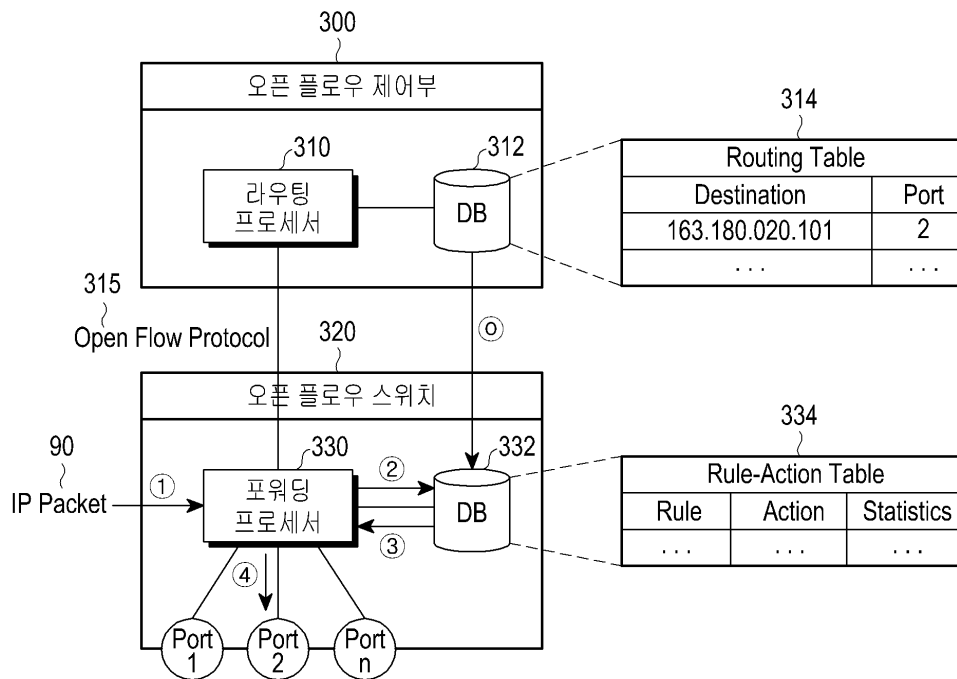
도면1



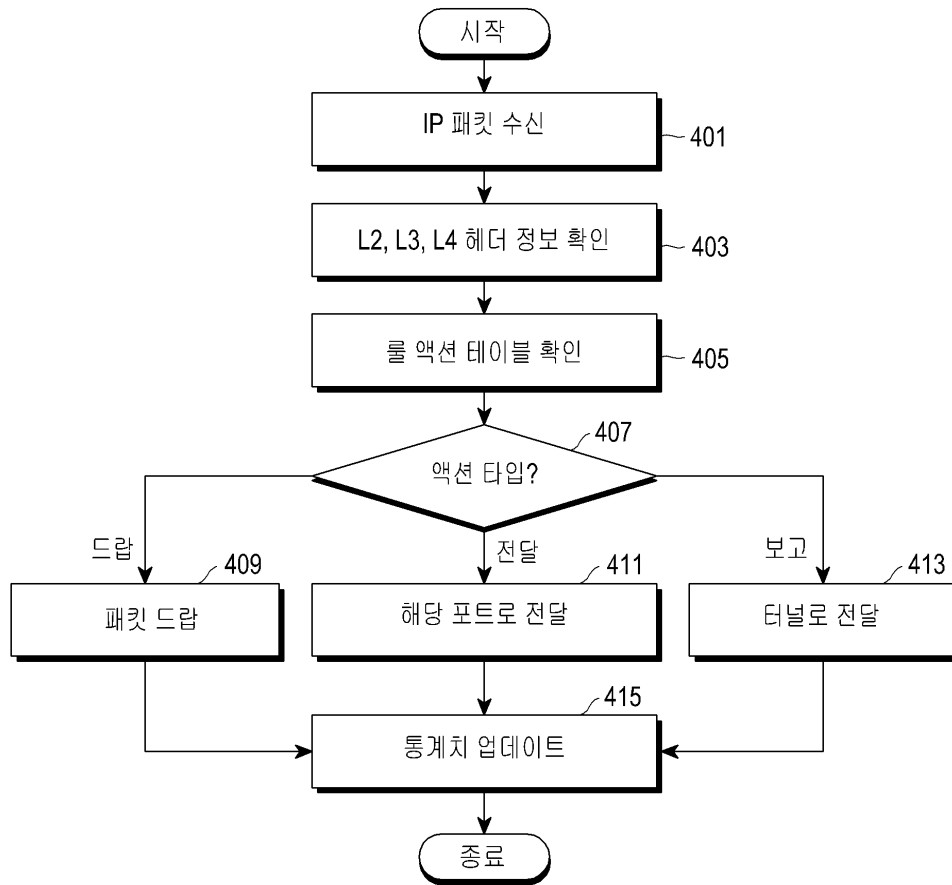
도면2



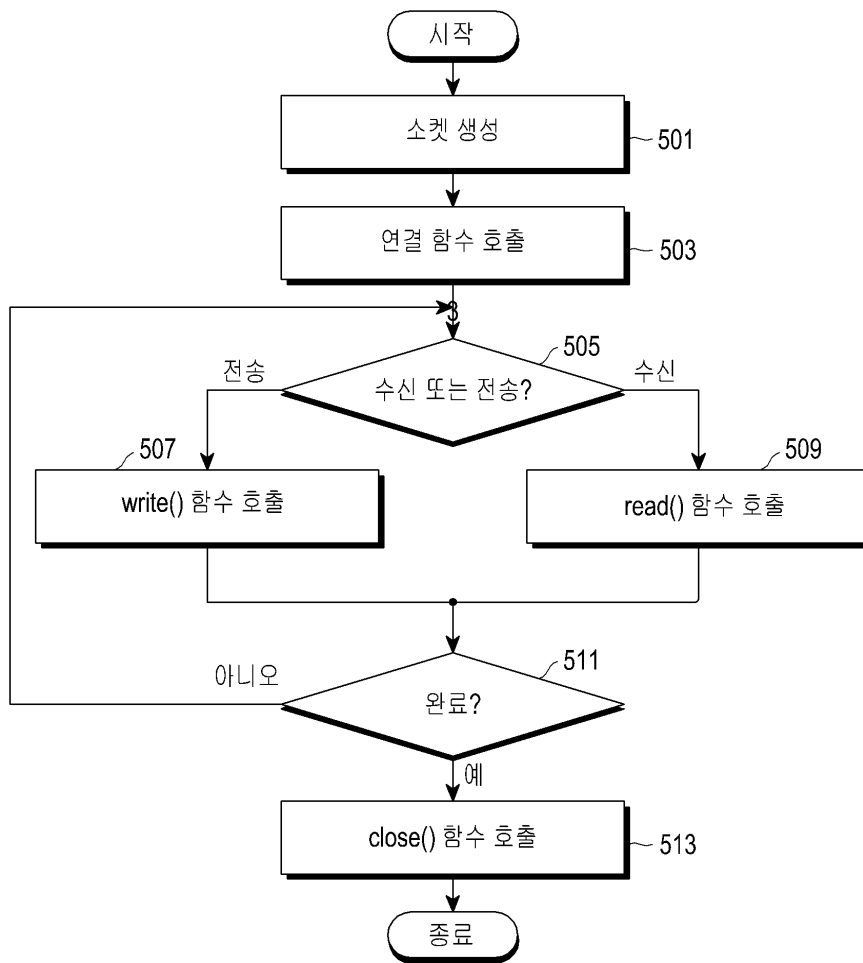
도면3



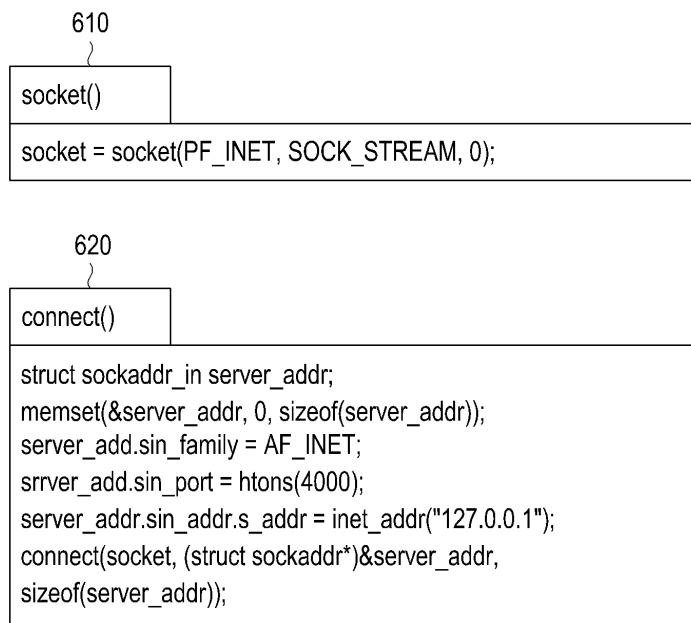
도면4



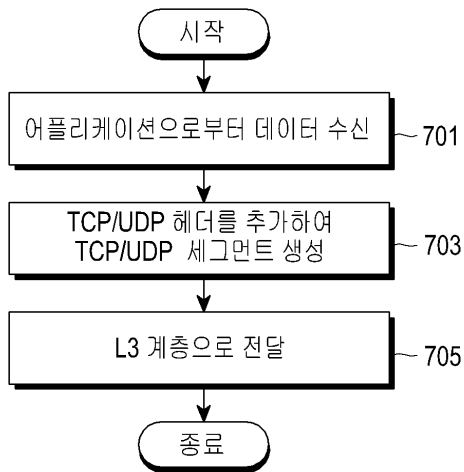
도면5



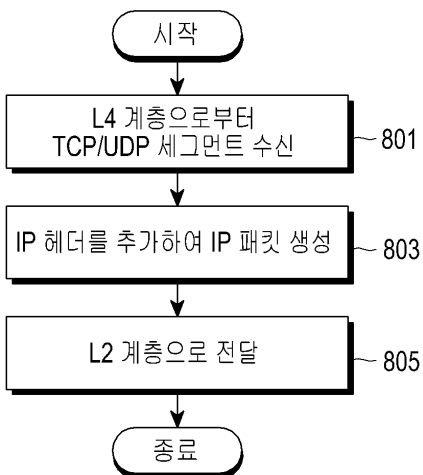
도면6



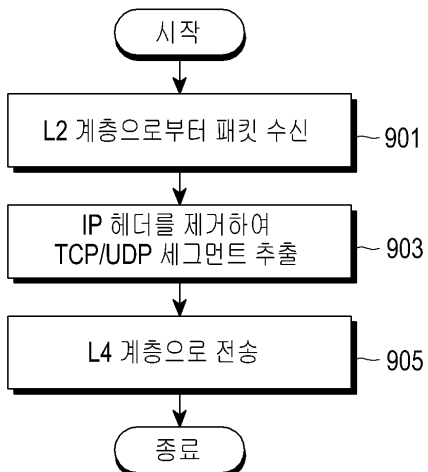
도면7



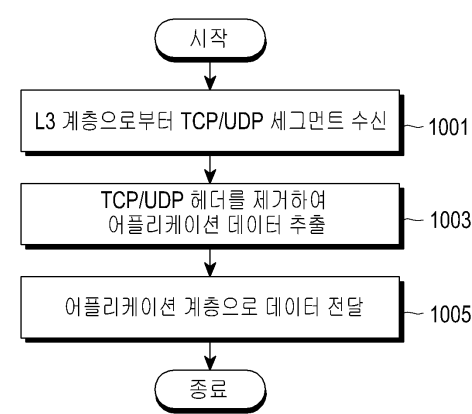
도면8



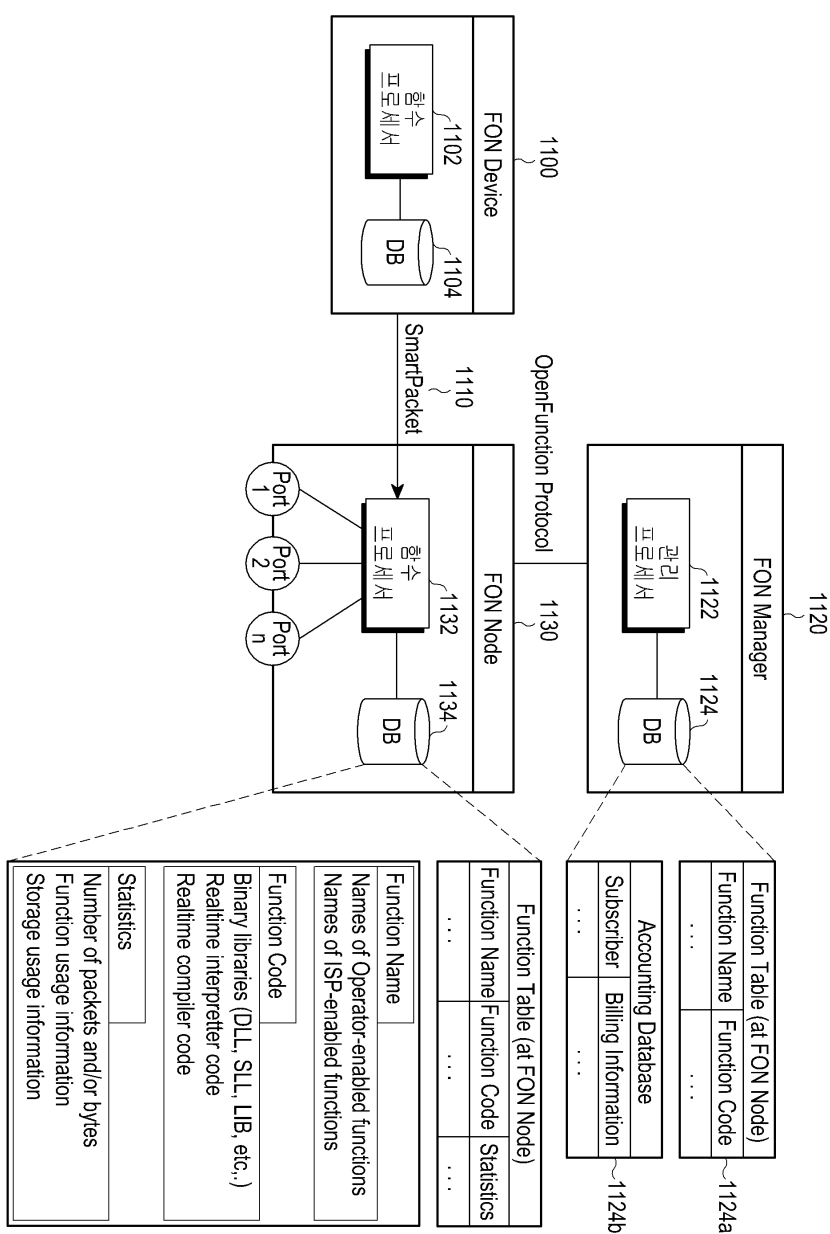
도면9



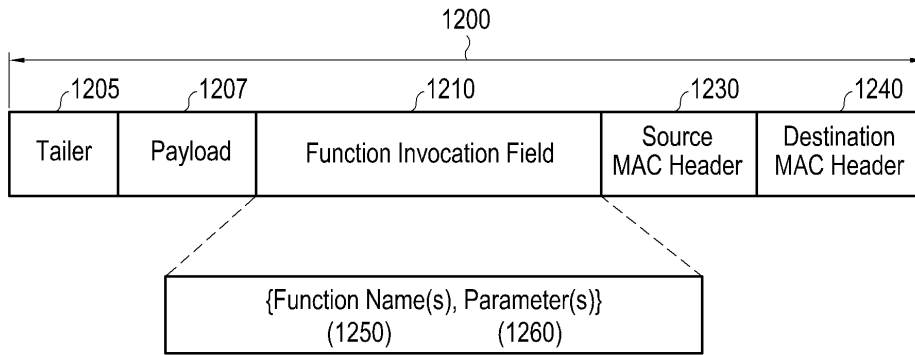
도면10



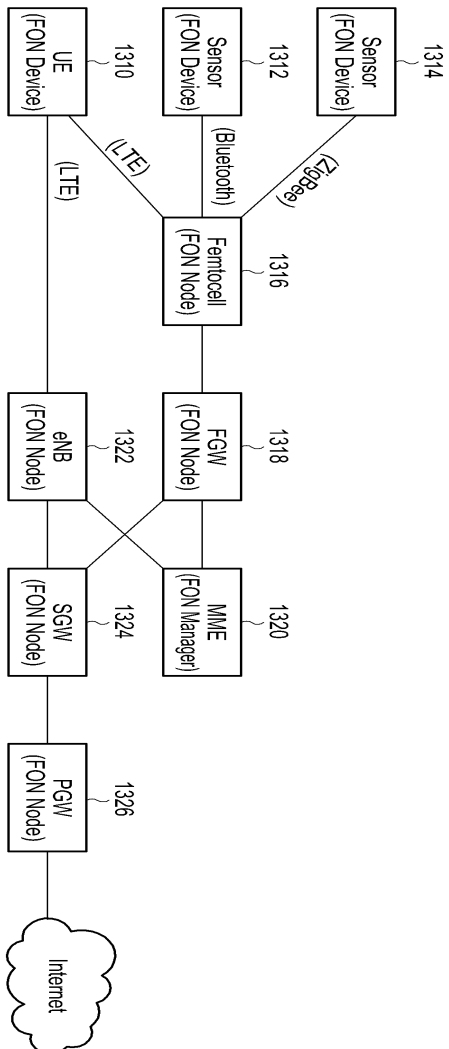
도면11



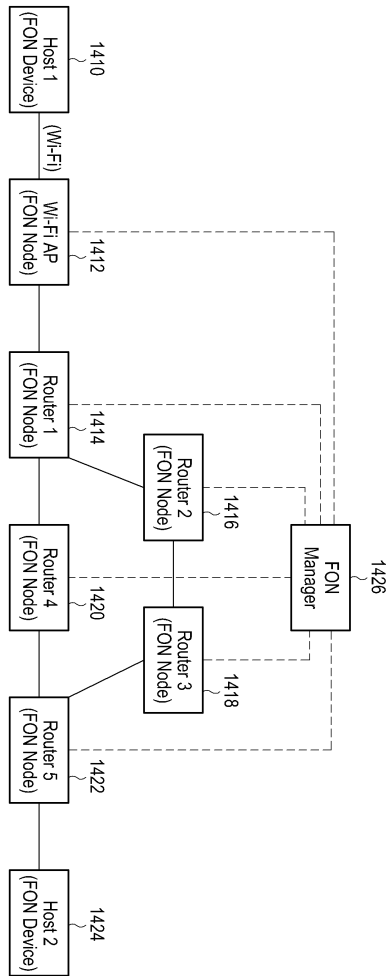
도면12



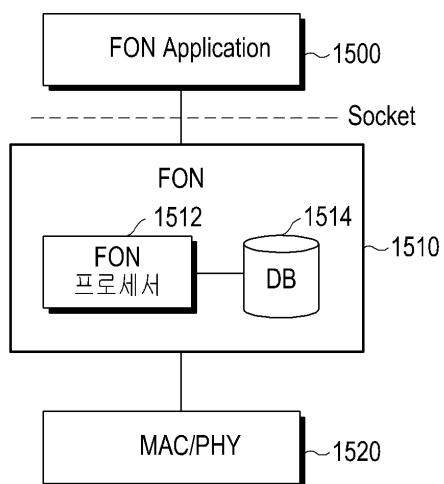
도면13



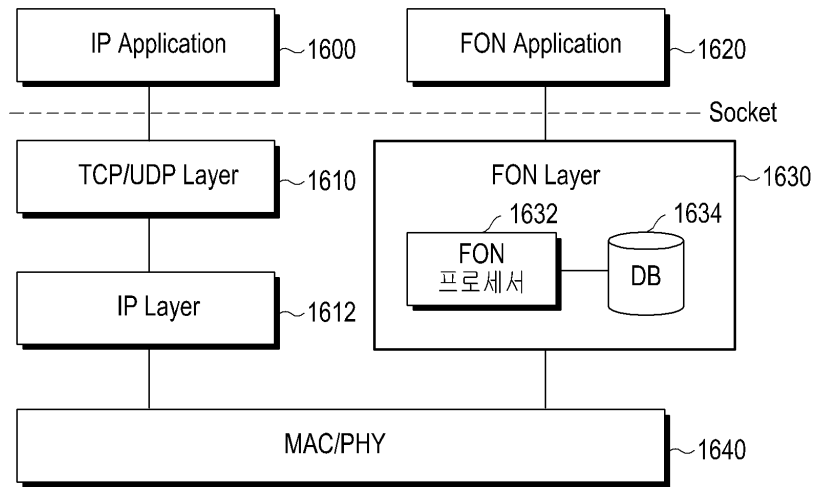
도면14



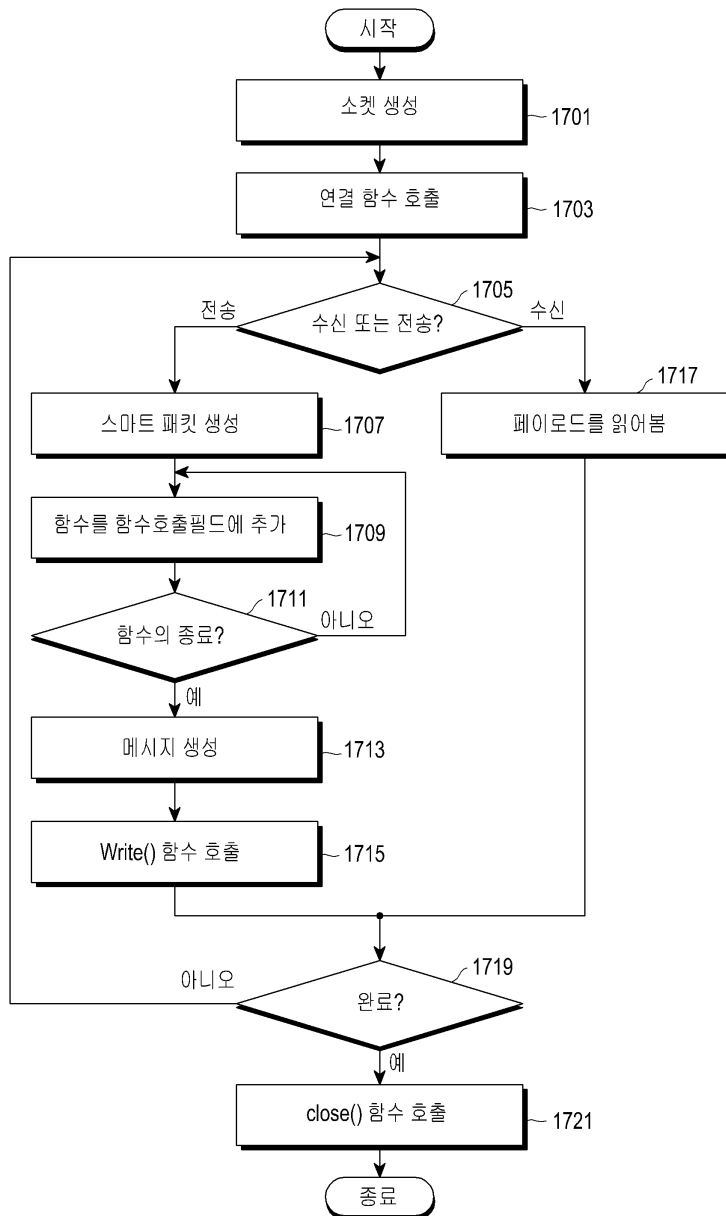
도면15



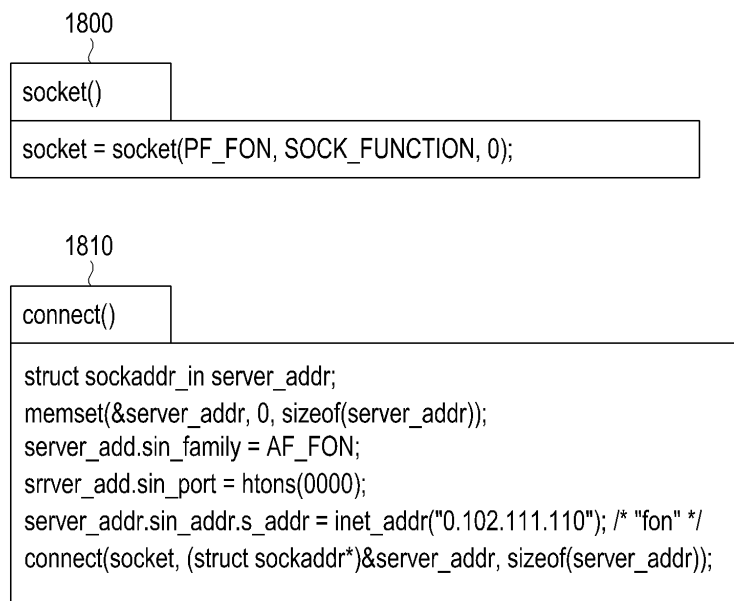
도면16



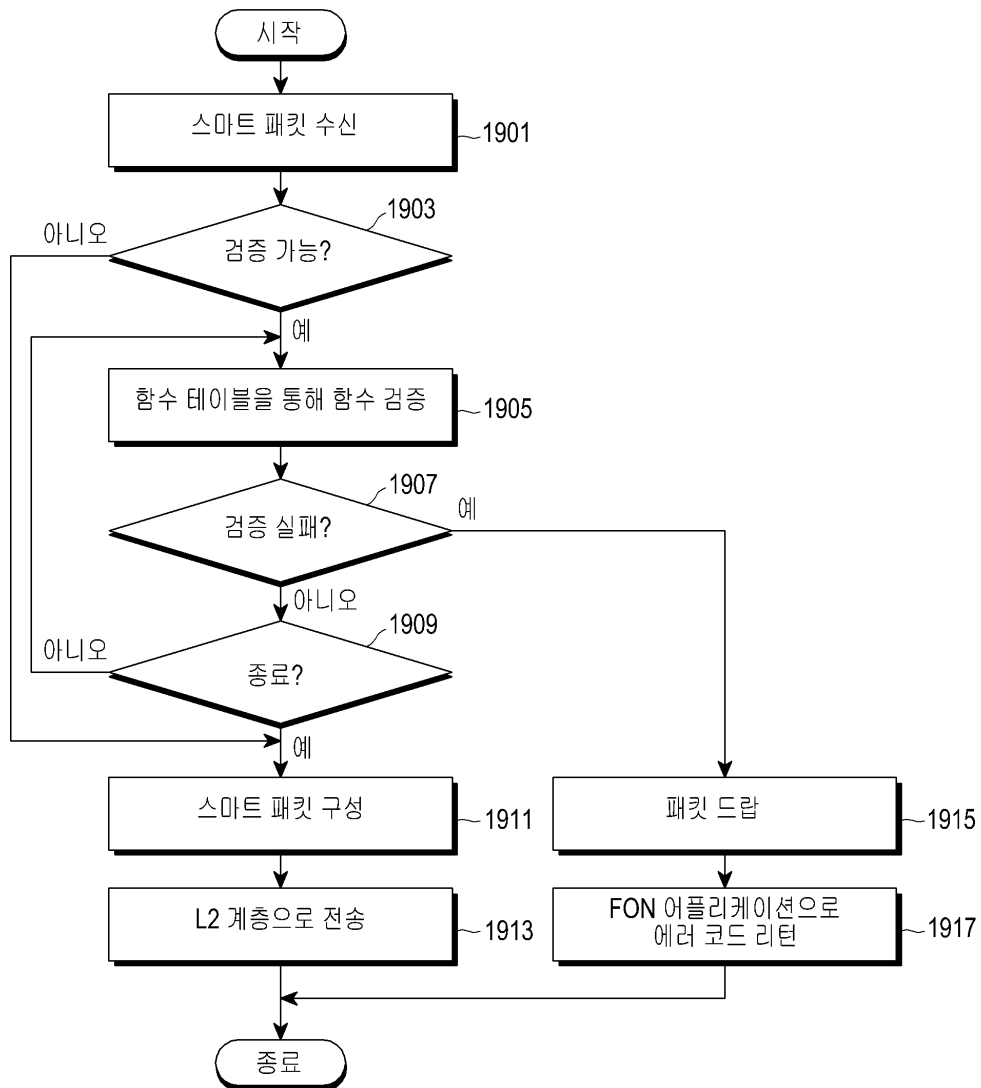
도면17



도면18



도면19

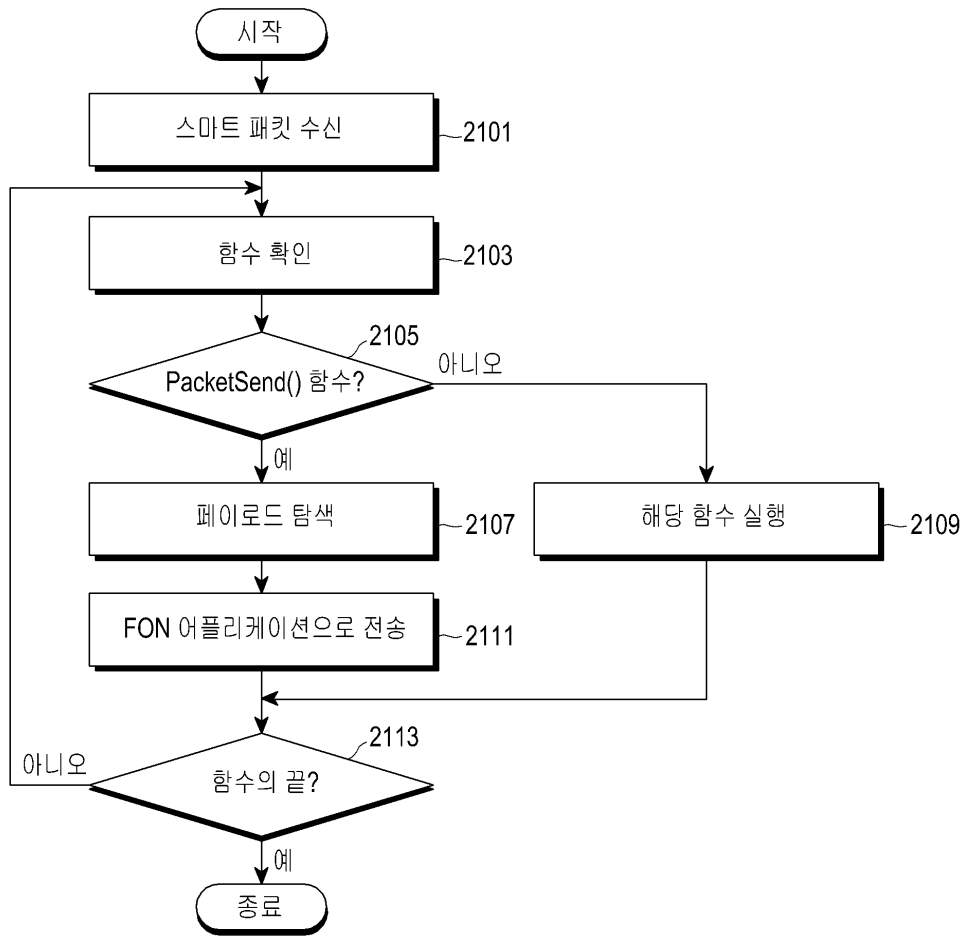


도면20

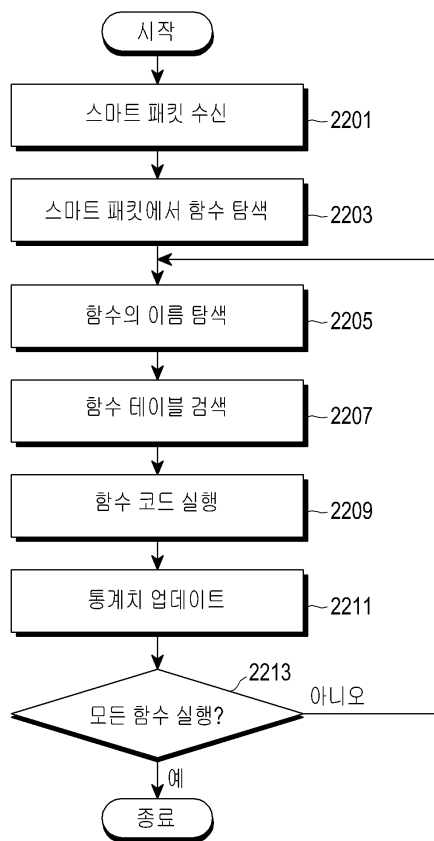
2000

Function Table	
Function Name	Function Code
PacketSend ()	{ ... }
NewFunction 1()	{ ... }
NewFunction 2()	—
NewFunction 3()	—
NewFunction 4()	—

도면21



도면22



도면23

2300

Function Table		
Function Name	Function Code	Statistics
PacketSend ()	{ ... }	100
SystemRead ()	{ ... }	200

도면24

2400

```

<FON Device Info>
  <Wi-Fi MAC Address>
    11:22:33:44:55:66
  </Wi-Fi MAC Address>
  <IMSI>
    450051012341234
  </IMSI>
</FON Device Info>

<Function Invocation Info>
  <Function 1>
    <Invocating Number> 2 </Invocating Number>
  </Function 1>
  <Function 2>
    <Invocating Number> 3 </Invocating Number>
  </Function 2>
</Function Invocation Info>
  
```

도면25

2500

	함수 이름	함수 기능
패킷 전달 (Transportation)	PacketSend()	패킷을 요청된 곳으로 전송함
	PacketInsertField()	패킷의 특정 위치(필드)에 요청된 값을 삽입함
	PacketRemoveField()	패킷의 특정 위치(필드)에 값을 제거함
	PacketAddField()	패킷의 마지막 부분에 요청된 값을 삽입함
	PacketPorts()	패킷이 전송될 포트 번호(들)를 읽음
시스템정보 조회 (System Info Retrieval)	SystemRead()	FON 노드의 시스템 정보를 읽음
사용자정보 조회 (Subscriber Info Retrieval)	SubscriberRead()	FON 노드에 접속한 사용자/단말 정보를 읽음
사용자 저장공간 사용 (Storage)	StorageAvailability()	FON 노드에 사용자가 요청한 변수가 있는지 확인함
	StorageCreate()	FON 노드에 사용자가 요청한 변수를 생성함
	StorageRead()	FON 노드에 사용자가 생성한 변수의 값을 읽음
	StorageWrite()	FON 노드에 사용자가 생성한 변수에 값을 저장함
	StorageRemove()	FON 노드에 사용자가 요청한 변수를 제거함
사업자 기능 수행 (Logic Execution)	MathRandom()	입력 범위내에서 Random 함수로 결과 값을 산출함
	MathGetMax()	입력 값을 비교하여 최대값을 리턴함

도면26

2600

```

<Command>
  Publication
</Command>

<Target>
  FON Device, FON Node
</Target>

<Function Table>
  <Function 1>
    <name> A </name>
    <input parameter> int </input parameter>
    <output parameter> void </output parameter>
    <Function Code>
      A(int a1)
      {
        Print(a1);
      }
    </Function Code>
  </Function 1>
  <Function 2>
    <name> B </name>
    <input parameter> char </input parameter>
    <output parameter> void </output parameter>
    <Function Code>
      B(char b1)
      {
        Print(b1);
      }
    </Function Code>
  </Function 2>
</Function Table>
  
```

도면27

2700

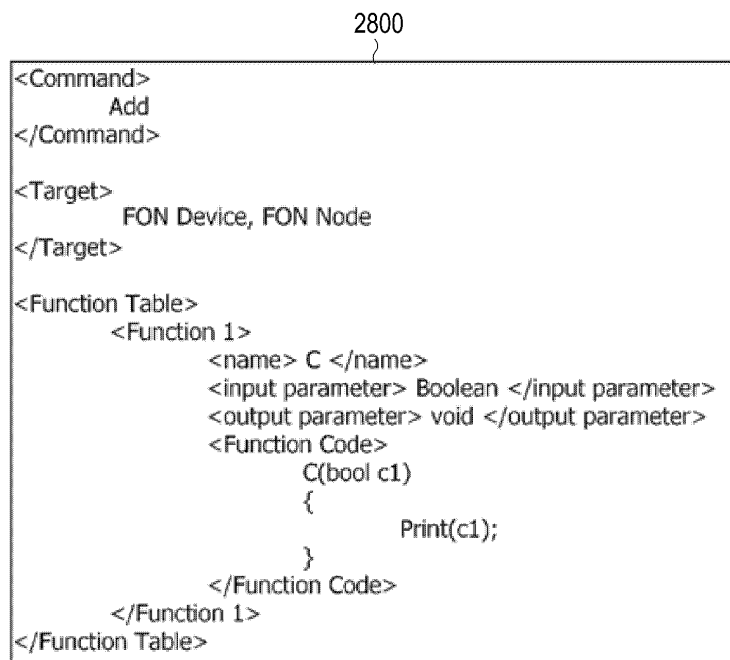
```

<Command>
  Delete
</Command>

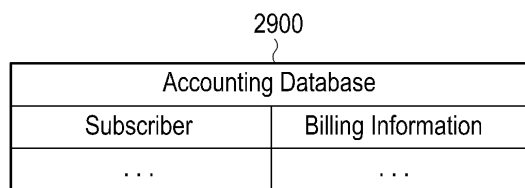
<Target>
  FON Device, FON Node
</Target>

<Function Table>
  <Function 1>
    <name> A </name>
    <input parameter> int </input parameter>
    <output parameter> void </output parameter>
    <Function Code>
      A(int a1)
      {
        Print(a1);
      }
    </Function Code>
  </Function 1>
</Function Table>
  
```

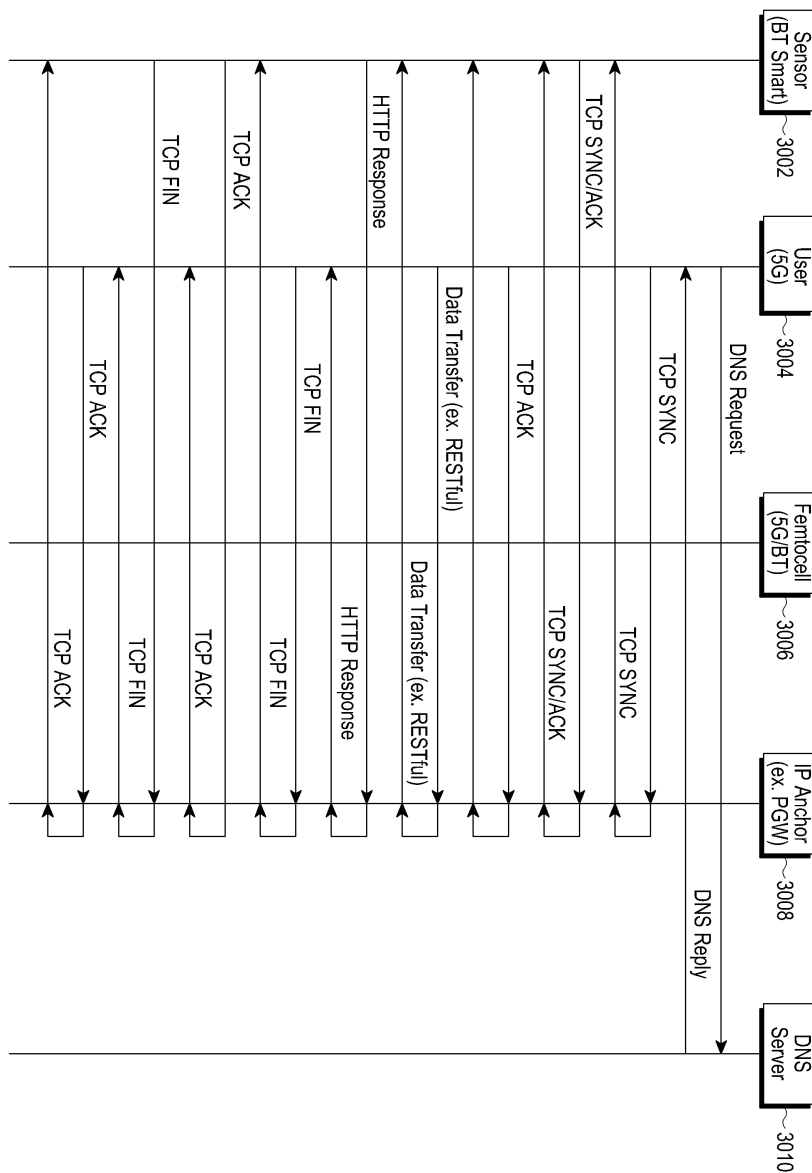
도면28



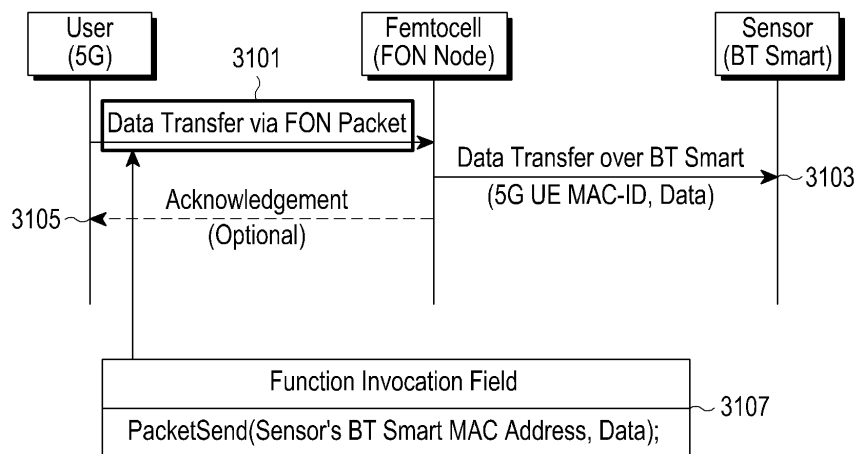
도면29



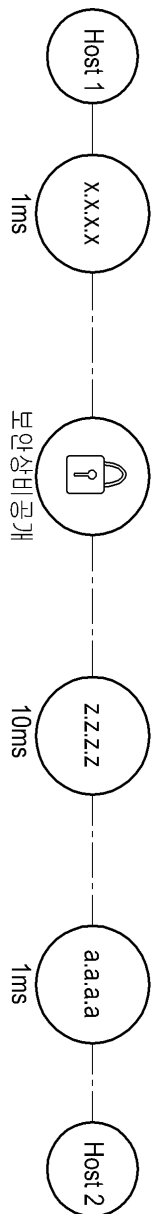
도면30



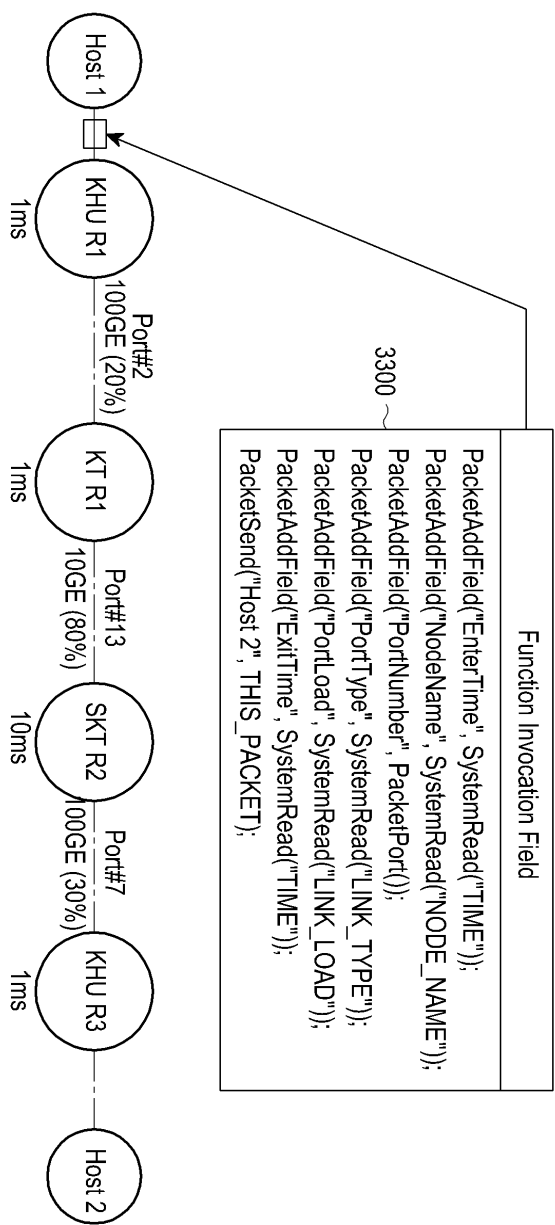
도면31



도면32



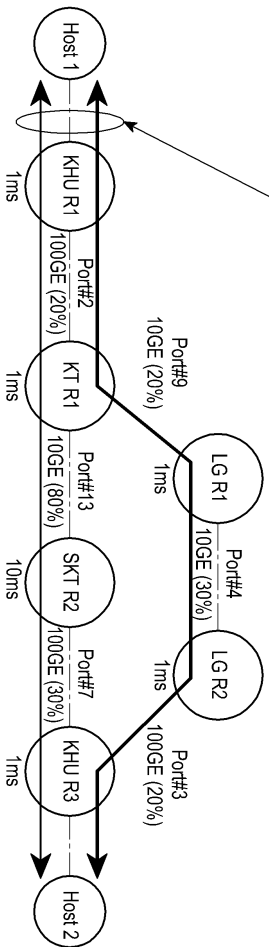
도면33



Function Invocation Field

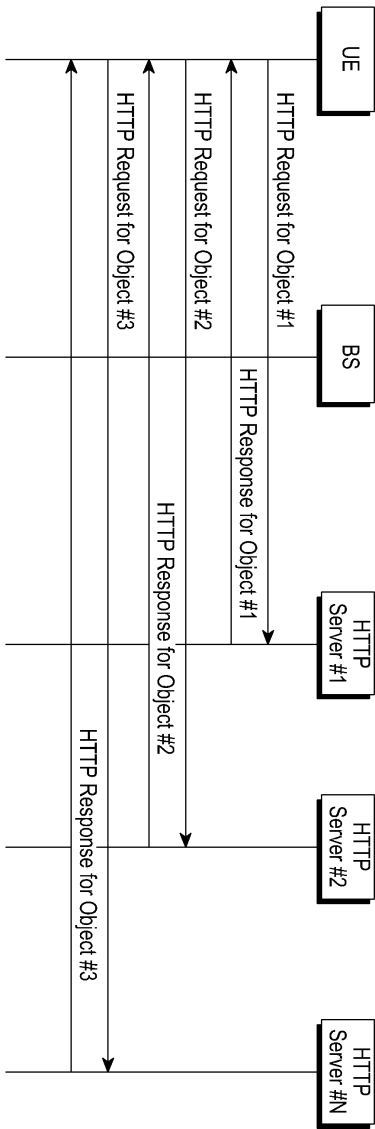
```
TEMP_ENABLE = StorageAvailability("MY_Pheromone");
if(TEMP_ENABLE == FALSE):
    TEMP_PORTS = PacketPort("Host 2");
    StorageCreate("MY_Pheromone", TEMP_PORTS);
    StorageWrite("MY_Pheromone", ALL, 0);
endif;

TEMP_RANDOM = MathRandom(0,99);
if(TEMP_RANDOM > 70):
    TEMP_CURRENT_PHEROMONE = StorageRead(lMY_Pheromone,
    MathGetMax("MY_Pheromone"));
    StorageWrite(lMY_Pheromone, RANDOM, TEMP_CURRENT_PHEROMONE + 1);
    PacketSend("Host 2", THIS_PACKET, MathGetMax("MY_Pheromone"));
elseif:
    TEMP_RANDOM = MathRandom(0, TEMP_PORTS);
    TEMP_CURRENT_PHEROMONE = StorageRead(lMY_Pheromone, RANDOM);
    StorageWrite(lMY_Pheromone, RANDOM, TEMP_CURRENT_PHEROMONE + 1);
    PacketSend("Host 2", THIS_PACKET, TEMP_RANDOM);
endif;
```



도면34

도면35



도면36

