



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2019년03월08일
(11) 등록번호 10-1956408
(24) 등록일자 2019년03월04일

(51) 국제특허분류(Int. Cl.)
H04L 12/721 (2013.01) H04L 12/751 (2013.01)
(21) 출원번호 10-2014-7006124
(22) 출원일자(국제) 2012년07월26일
심사청구일자 2017년06월26일
(85) 번역문제출일자 2014년03월06일
(65) 공개번호 10-2014-0059795
(43) 공개일자 2014년05월16일
(86) 국제출원번호 PCT/IB2012/053833
(87) 국제공개번호 WO 2013/021304
국제공개일자 2013년02월14일
(30) 우선권주장
13/208,251 2011년08월11일 미국(US)
(56) 선행기술조사문헌
KR1020050089682 A
(뒷면에 계속)

(73) 특허권자
텔레호낙티에블라게트 엘엠 에릭슨(피유비엘)
스웨덴 스톡홀름 83 에스이-164
(72) 발명자
예다발리, 키란
미국 95135 캘리포니아주 산 호세 들라크루와 코
트 4280
베헤쉬티-자바레호, 네다
미국 95134 캘리포니아주 산 호세 유닛 1026 팔밀
라 디알. 3500
장, 잉
미국 94555 캘리포니아주 프리몬트 미드웨이 커먼
5474
(74) 대리인
장수길, 백만기

전체 청구항 수 : 총 16 항

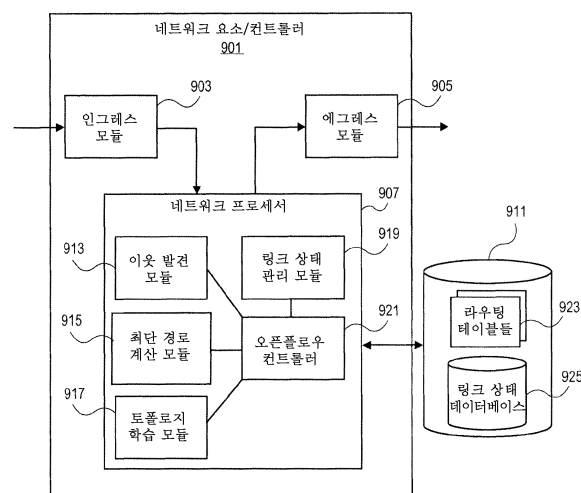
심사관 : 남기영

(54) 발명의 명칭 분할된 아키텍처 네트워크들에서 OSPF의 구현

(57) 요약

분할된 아키텍처 네트워크의 복수의 영역들 중 하나에 대해 복수의 컨트롤러들 중 하나로서 기능하는 네트워크 요소에서 방법이 구현된다. 컨트롤러는 분할된 아키텍처 네트워크의 영역에 대한 제어 플레인을 제공하고, 여기에서 컨트롤러는 분할된 아키텍처 네트워크의 영역에 대한 데이터 플레인을 제공하는 복수의 스위치들로부터 이격되어 있다. 컨트롤러는 제한된 인트라-영역 링크 코스트 데이터를, 분할된 아키텍처 네트워크의 다른 영역들의 다른 컨트롤러들에게, 그리고 분할된 아키텍처 네트워크를 포함하는 네트워크의 전통적인 라우터들에게 제공함으로써, 분할된 아키텍처 네트워크의 복수의 영역들에 걸쳐 최적화된 라우팅을 용이하게 한다. 제한된 인트라-영역 링크 코스트 데이터는 모든 내부 링크 코스트 데이터를 제공하지 않고 컨트롤러의 영역의 각 가능한 최단 경로 횡단에 대한 코스트들을 제공한다.

대표도 - 도9



(56) 선행기술조사문헌
KR1020050081780 A
US20030206528 A1
WO2010101705 A1
US20060092940 A1

명세서

청구범위

청구항 1

분할 아키텍처 네트워크(split architecture network)의 복수의 영역들 중 하나의 영역에 대한 복수의 컨트롤러들 중 하나의 컨트롤러로서 기능하는 네트워크 요소에서 구현되는 방법으로서 - 상기 컨트롤러는 상기 분할 아키텍처 네트워크의 상기 영역에 대해 제어 플레인(control plane)을 제공하고, 상기 컨트롤러는 상기 분할 아키텍처 네트워크의 영역에 대해 데이터 플레인(data plane)을 제공하는 복수의 스위치들로부터 원격에 있고, 상기 컨트롤러는, 제한된 인트라-영역 링크 코스트 데이터(limited intra-area link cost data)를 상기 분할 아키텍처 네트워크의 다른 영역들의 다른 컨트롤러들에 그리고 상기 분할 아키텍처 네트워크를 포함하는 네트워크의 전통적인 라우터들(traditional routers)에 제공함으로써, 상기 분할 아키텍처 네트워크의 상기 복수의 영역들에 걸쳐 최적화된 라우팅을 용이하게 하고, 상기 제한된 인트라-영역 링크 코스트 데이터는 모든 내부 링크 코스트 데이터를 제공하지 않고 상기 컨트롤러의 영역의 각각의 가능한 최단 경로 횡단(shortest path traversal)의 코스트들을 제공함 -;

상기 컨트롤러의 영역에서의 각각의 경계 스위치(border switch)를 포함하여 상기 분할 아키텍처 네트워크에서의 상기 컨트롤러의 영역의 토폴로지를 학습하는 단계 - 상기 컨트롤러의 영역에서의 각각의 경계 스위치는, 상기 컨트롤러의 영역을 상기 분할 아키텍처 네트워크의 다른 영역에 또는 상기 네트워크에서의 상기 전통적인 라우터들 중 하나의 라우터에 링크하는 적어도 하나의 외부 포트를 가짐 -;

상기 컨트롤러의 영역에서의 각각의 경계 스위치 쌍 사이의 최단 경로를 계산하는 단계;

각각의 경계 스위치 쌍 사이의 각각의 최단 경로의 코스트를 상기 컨트롤러의 라우팅 테이블에 저장하는 단계;

헬로우 프로토콜(hello protocol)을 이용하여 상기 분할 아키텍처 네트워크에서의 각각의 이웃 컨트롤러 또는 상기 네트워크에서의 전통적인 이웃 라우터를 식별하는 단계 - 각각의 이웃 컨트롤러는 상기 컨트롤러의 영역의 적어도 하나의 외부 포트를 통해 액세스가능한 상기 분할 아키텍처 네트워크의 다른 영역에서의 스위치들을 제어함 -;

각각의 이웃 컨트롤러 및 전통적인 이웃 라우터와 링크 상태 데이터베이스를 교환하는 단계 - 상기 링크 상태 데이터베이스는 각각의 경계 스위치 쌍 사이의 각각의 최단 경로의 코스트를 포함함 -;

상기 네트워크에 대한 최단 경로 트리를, 상기 컨트롤러를 상기 트리의 루트로 하여(with the controller as a root of the tree) 계산하는 단계; 및

상기 최단 경로 트리에 따라 포워딩을 구현하기 위해서, 상기 컨트롤러의 영역의 스위치들에서의 포워딩 테이블들을 업데이트하는 단계

를 포함하는 방법.

청구항 2

제1항에 있어서, 상기 각각의 이웃 컨트롤러를 식별하는 단계는,

상기 컨트롤러의 영역의 각각의 외부 포트 상에서 헬로우 패킷을 송신하는 단계; 및

상기 컨트롤러의 영역의 적어도 하나의 외부 포트를 통해 각각의 이웃 컨트롤러로부터 헬로우 패킷들을 수신하는 단계

를 더 포함하는 방법.

청구항 3

제1항에 있어서, 상기 링크 상태 데이터베이스를 교환하는 단계는 각각의 이웃 컨트롤러에 링크 상태 광고를 송신하는 단계를 더 포함하고, 상기 링크 상태 광고는 상기 컨트롤러의 각각의 외부 링크에 대한 코스트를 포함하는 방법.

청구항 4

제1항에 있어서, 상기 링크 상태 데이터베이스를 교환하는 단계는 각각의 이웃 컨트롤러에 링크 상태 광고를 송신하는 단계를 더 포함하고, 상기 링크 상태 광고는 상기 영역에서의 각각의 경계 스위치 쌍에 대한 최단 경로의 코스트를 포함하는 방법.

청구항 5

제1항에 있어서, 업데이트된 링크 상태 데이터를 각각의 이웃 컨트롤러에 광고하는 단계를 더 포함하는 방법.

청구항 6

제5항에 있어서, 상기 업데이트된 링크 상태 데이터를 광고하는 단계는 업데이트된 링크 상태 데이터를 갖는 링크 상태 광고 메시지를 각각의 이웃 컨트롤러에 송신하는 단계를 더 포함하는 방법.

청구항 7

제1항에 있어서, 상기 포워딩 테이블들을 업데이트하는 단계는 오픈플로우 프로토콜(OpenFlow Protocol)을 이용하여 상기 컨트롤러의 영역에서의 각각의 스위치의 포워딩 테이블들을 업데이트하는 단계를 더 포함하는 방법.

청구항 8

제1항에 있어서, 상기 컨트롤러에서 OSPF(Open Shortest Path First) 프로토콜 데이터 구조들을 초기화하는 단계를 더 포함하는 방법.

청구항 9

분할 아키텍처 네트워크의 복수의 영역들 중 하나의 영역에 대한 복수의 컨트롤러들 중 하나의 컨트롤러로서 기능하는 네트워크 요소로서 - 상기 컨트롤러는 상기 분할 아키텍처 네트워크의 상기 영역에 대해 제어 플레인을 제공하고, 상기 컨트롤러는 상기 분할 아키텍처 네트워크의 영역에 대해 데이터 플레인을 제공하는 복수의 스위치들로부터 원격에 있고, 상기 컨트롤러는, 제한된 인트라-영역 링크 코스트 데이터를 상기 분할 아키텍처 네트워크의 다른 영역들의 다른 컨트롤러들에 그리고 상기 분할 아키텍처 네트워크를 포함하는 네트워크의 전통적인 라우터들에 제공함으로써, 상기 분할 아키텍처 네트워크의 상기 복수의 영역들에 걸쳐 최적화된 라우팅을 용이하게 하고, 상기 제한된 인트라-영역 링크 코스트 데이터는 모든 내부 링크 코스트 데이터를 제공하지 않고 상기 컨트롤러의 영역의 각각의 가능한 최단 경로 횡단의 코스트들을 제공함 -,

네트워크를 통해 데이터를 수신하도록 구성된 인그레스 모듈(ingress module);

상기 네트워크를 통해 데이터를 송신하도록 구성된 에그레스 모듈(egress module);

상기 인그레스 모듈 및 상기 에그레스 모듈에 결합된 네트워크 프로세서 - 상기 네트워크 프로세서는, 컨트롤러 모듈, 토폴로지 학습 모듈, 최단 경로 계산 모듈, 이웃 발견 모듈 및 링크 상태 관리 모듈을 포함하는 모듈들의 세트를 실행하도록 구성됨 -; 및

상기 네트워크 프로세서에 통신가능하게 결합된 라우팅 테이블 저장 디바이스를 포함하고,

상기 토폴로지 학습 모듈은 상기 컨트롤러의 영역에서의 각각의 경계 스위치를 포함하여 상기 분할 아키텍처 네트워크에서의 상기 컨트롤러의 영역의 토폴로지를 결정하도록 구성되고, 각각의 경계 스위치는, 상기 컨트롤러의 영역을 상기 네트워크의 다른 영역에 또는 상기 네트워크에서의 전통적인 라우터에 링크하는 적어도 하나의 외부 포트를 갖고,

상기 컨트롤러 모듈은 상기 분할 아키텍처 네트워크에서의 상기 컨트롤러의 영역에 대한 제어 플레인 기능을 제공하도록 구성되고,

상기 최단 경로 계산 모듈은, 이웃 컨트롤러들 및 상기 전통적인 라우터들과 공유될, 상기 컨트롤러의 영역에서의 각각의 경계 스위치 사이의 최단 경로를 식별하고, 상기 네트워크에 대한 최단 경로 트리를, 상기 컨트롤러를 상기 트리의 루트로 하여 계산하도록 구성되고,

상기 이웃 발견 모듈은 헬로우 프로토콜을 이용하여 상기 네트워크에서의 각각의 이웃 컨트롤러 및 전통적인 라

우터들을 식별하도록 구성되고,

상기 링크 상태 관리 모듈은 상기 네트워크에서의 각각의 이웃 컨트롤러 및 전통적인 라우터들과 링크 상태 데이터베이스를 교환하도록 구성되고,

상기 링크 상태 데이터베이스는 상기 컨트롤러의 영역에서의 각각의 경계 스위치 쌍 사이의 각각의 최단 경로(route)의 코스트를 포함하며,

상기 라우팅 테이블 저장 디바이스는, 상기 컨트롤러의 영역에 대한 최단 경로 정보 및 상기 컨트롤러의 영역의 경계 스위치들 사이의 최단 경로 정보를 포함하여 상기 컨트롤러 모듈에 대한 라우팅 테이블들을 저장하도록 구성되는 네트워크 요소.

청구항 10

제9항에 있어서, 상기 이웃 발견 모듈은, 상기 영역의 각각의 외부 포트 상에서 헬로우 패킷을 송신하고, 상기 영역의 적어도 하나의 외부 포트를 통해 각각의 이웃 컨트롤러로부터 헬로우 패킷들을 수신하도록 구성되는 네트워크 요소.

청구항 11

제9항에 있어서, 상기 링크 상태 관리 모듈은 각각의 이웃 컨트롤러에 링크 상태 광고를 송신하도록 또한 구성되고, 상기 링크 상태 광고는 상기 컨트롤러의 각각의 외부 링크에 대한 코스트를 포함하는 네트워크 요소.

청구항 12

제9항에 있어서, 상기 링크 상태 관리 모듈은 각각의 이웃 컨트롤러에 링크 상태 광고를 송신하도록 또한 구성되고, 상기 링크 상태 광고는 상기 컨트롤러의 영역에서의 각각의 경계 스위치 쌍에 대한 최단 경로의 코스트를 포함하는 네트워크 요소.

청구항 13

제9항에 있어서, 상기 링크 상태 관리 모듈은 업데이트된 링크 상태 데이터를 각각의 이웃 컨트롤러에 광고하도록 또한 구성되는 네트워크 요소.

청구항 14

제9항에 있어서, 상기 링크 상태 관리 모듈은, 업데이트된 링크 상태 데이터를 갖는 링크 상태 광고 메시지를 각각의 이웃 컨트롤러에 송신하도록 또한 구성되는 네트워크 요소.

청구항 15

제9항에 있어서, 상기 컨트롤러 모듈은, 오픈폴로우 프로토콜을 이용하여 상기 컨트롤러의 영역에서의 각각의 스위치의 포워딩 테이블들을 업데이트하도록 또한 구성되는 네트워크 요소.

청구항 16

제9항에 있어서, 상기 최단 경로 계산 모듈은, 상기 컨트롤러에서 OSPF(Open Shortest Path First) 프로토콜 데이터 구조들을 초기화하도록 또한 구성되는 네트워크 요소.

발명의 설명

기술 분야

[0001] 본 발명의 실시예들은 분할된 아키텍처 네트워크들에서 경로 찾기 또는 라우팅에 관한 것이다. 특히, 본 발명의 실시예들은 분할된 아키텍처 네트워크 및 전통적인 라우터들을 포함하는 네트워크들에서 데이터를 라우팅하기 위한 OSPF(open shortest path first) 프로세스를 구현하기 위한 방법 및 시스템에 관한 것이다.

배경 기술

[0002] 분할된-아키텍처 네트워크 설계는 또한 제어 플레인(control plane) 및 포워딩 또는 데이터 플레인(data plan

e)으로도 지칭되는, 네트워크의 제어 및 포워딩 컴포넌트들 사이의 분리를 도입한다. 분할된 아키텍처 네트워크들은 캐리어-그레이트 네트워크, 모바일 백홀, 클라우드 컴퓨팅, 및 멀티레이어(L3 및 L2 및 L1, 광학 트랜스포트 네트워크(OTN), 파장 분할 멀티플렉싱(WDM)) 서포트의 액세스/어그리게이션 도메인 내에서 활용될 수 있고, 이들 모두는 네트워크 아키텍처의 구축하는 블록들 중에 있다.

[0003] 양쪽 포워딩(데이터) 및 제어 플레인들 모두를 동일한 박스 내에 통합하는 전통적인 네트워크 아키텍처와는 달리, 분할된 아키텍처는 이들 2개의 기능들을 디커플링하고, 포워딩 요소들(스위치들)과 상이한 물리적 로케이션들에 있을 수 있는 서버들(컨트롤러들) 상에서 제어 플레인을 운용한다. 분할된 아키텍처는 포워딩 플랫폼의 기능들 및 하드웨어를 단순화시키고, 네트워크의 지능 및 관리를, 스위치들을 감독하는 컨트롤러들의 세트에 집중시킨다. 전통적인 네트워크 아키텍처에서의 포워딩 및 제어 플레인들의 밀접한 커플링은 대개는 결과적으로 고도로 복잡화된 제어 플레인 및 복잡한 네트워크 관리로 나타난다. 이것은 새로운 네트워킹 디바이스들의 생성을 높은 비용이 들게 하고, 이들 디바이스들에서 잠재적인 전개를 위한 새로운 프로토콜들 및 기술에 대한 높은 진입 장벽을 생성한다. 라인 속도들, 포트 밀도들, 및 성능에 대한 급격한 개선에도 불구하고, 이들 특징들을 관리하기 위한 네트워크 제어 플레인 메커니즘들은 훨씬 더 느린 페이스로 진보되어 왔다.

[0004] 분할된 아키텍처 네트워크에서, 컨트롤러들은 스위치들로부터 정보를 수집하고, 적절한 포워딩 결정들을 계산하여 스위치들에게 분배한다. 컨트롤러들 및 스위치들은 제어 플레인 프로토콜을 이용하여 정보를 통신하고 교환한다. 그러한 프로토콜의 예는 오픈플로우로서, 스위치가 컨트롤러와 통신하는 오픈 및 표준 방법을 제공한다. 도 1은 스위치와 컨트롤러 사이의 오픈플로우 인터페이스의 개관의 다이어그램이다. 오픈플로우 컨트롤러는 보안 채널을 이용하여 오픈플로우 스위치와 통신하여, 포워딩 테이블(플로우 테이블)을 구성한다.

[0005] 오픈플로우 스위치 내의 포워딩 테이블은 패킷 헤더들의 필드들에 대한 매치들을 정의하는 규칙들, 하나의 규칙에 의해 정의된 매치를 검출할 때 실행되는 액션들, 및 데이터 플레인 내의 데이터 패킷들의 처리에 관한 통계들의 집합으로 구성된 엔트리들로 파플레이팅된다. 인커밍 데이터 패킷이 특정 규칙과 매칭하는 경우에, 데이터 패킷에 대한 연관된 액션들이 수행된다. 규칙은 프로토콜 스택 내에 수 개의 헤더들로부터의 핵심 필드들, 예를 들면 이더넷 MAC 어드레스, IP 어드레스, IP 프로토콜, TCP/UDP 포트 번호들뿐만 아니라 인커밍 포트 번호를 포함한다. 유사한 특성들을 가지고 있는 데이터 패킷들의 세트는 하나의 플로우로서 관리될 수 있다. 하나의 플로우는 데이터 패킷 내의 가용한 필드들의 임의의 개수 또는 조합을 이용하여 정의될 수 있다. 또한, 원하지 않는 필드들에 대한 와일드카드들을 이용함으로써, 가용한 필드들의 서브세트 상에서 매칭하는 규칙들을 제한시킬 수 있다.

[0006] 분할된 아키텍처의 제어 플레인 및 데이터 플레인의 디-커플링은 네트워크 제어 로직을 변형하는 태스크를 용이하게 하고, 개발자들이 널리 다양한 새로운 프로토콜들 및 관리 어플리케이션들을 구축할 수 있는 프로그램 인터페이스를 제공한다. 이러한 모델에서, 데이터 및 제어 플레인들은 독립적으로 발전되고 스케일링할 수 있는데 대해, 데이터 플레인 요소들의 코스트가 감소된다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0007] 분할된 아키텍처 네트워크의 복수의 영역들 중 하나에 대해 복수의 컨트롤러들 중 하나로서 기능하는 네트워크 요소에서 구현되는 방법이 기재된다. 컨트롤러는 분할된 아키텍처 네트워크의 영역에 대해 제어 플레인을 제공하고, 여기에서 컨트롤러는 분할된 아키텍처 네트워크의 영역에 대해 데이터 플레인을 제공하는 복수의 스위치들로부터 이격(remote)되어 있다. 컨트롤러는 제한된 인트라-영역 링크 코스트 데이터를, 분할된 아키텍처 네트워크의 다른 영역들의 다른 컨트롤러들에게, 그리고 분할된 아키텍처 네트워크를 포함하는 네트워크의 전통적인 라우터들에게 제공함으로써, 분할된 아키텍처 네트워크의 복수의 영역들에 걸쳐 최적화된 라우팅을 용이하게 한다. 제한된 인트라-영역 링크 코스트 데이터는 모든 내부 링크 코스트 데이터를 제공하지 않고 컨트롤러의 영역의 각 가능한 최단 경로 횡단(traversal)에 대한 코스트들을 제공한다. 방법은 컨트롤러의 영역 내의 각 경계 스위치를 포함하는 분할된 아키텍처 네트워크에서 컨트롤러의 영역의 토폴로지를 학습하는 단계를 포함하여 뒤따르는 단계들을 포함하고, 여기에서 컨트롤러의 영역 내의 각 경계 스위치는 컨트롤러의 영역을 분할된 아키텍처 네트워크의 또 하나의 영역에, 또는 네트워크 내의 전통적인 라우터들 중 하나에 링크하는 적어도 하

나의 외부 포트를 가지고 있다. 단계들은 컨트롤러의 영역 내에서 각 경계 스위치 쌍 사이의 최단 경로를 계산하는 단계를 포함한다. 단계들은 각 경계 스위치 쌍 사이의 각 최단 경로의 코스트를 컨트롤러의 라우팅 테이블 내에 저장하는 단계를 포함한다. 단계들은 헬로우 프로토콜을 이용하여 분할된 아키텍처 네트워크 내에서 각 이웃 컨트롤러, 또는 네트워크 내에서 이웃 전통적 라우터를 식별하는 단계를 포함하고, 여기에서, 각 이웃 컨트롤러는 컨트롤러의 영역의 적어도 하나의 외부 포트를 통해 액세스가능한 분할된 아키텍처 네트워크의 또 하나의 영역 내의 스위치들을 제어한다. 단계들은 각 이웃 컨트롤러와 링크 상태 데이터베이스를 교환하는 단계를 포함하고, 링크 상태 데이터베이스는 각 경계 스위치 쌍 사이의 각 최단 경로의 코스트를 포함한다. 단계들은 컨트롤러를 트리의 루트(root)로 하여 네트워크에 대한 최단 경로 트리를 계산하고, 컨트롤러의 영역의 스위치들에서 포워딩 테이블들을 업데이트하여 최단 경로 트리에 따라 포워딩을 구현하는 단계를 포함한다.

[0008]

네트워크 요소는 분할된 아키텍처 네트워크의 복수의 영역들 중 하나에 대한 복수의 컨트롤러들 중 하나로서 기능한다. 컨트롤러는 분할된 아키텍처 네트워크의 영역에 대해 제어 플레인을 제공하고, 여기에서, 컨트롤러는 분할된 아키텍처 네트워크의 영역에 대해 데이터 플레인을 제공하는 복수의 스위치들로부터 이격되어 있다. 컨트롤러는 제한된 인트라-영역 링크 코스트 데이터를 분할된 아키텍처 네트워크의 다른 영역들의 다른 컨트롤러들에게, 그리고 분할된 아키텍처 네트워크를 포함하는 네트워크의 전통적인 라우터들에게 제공함으로써, 분할된 아키텍처 네트워크의 복수의 영역들에 걸쳐 최적화된 라우팅을 용이하게 한다. 제한된 인트라-영역 링크 코스트 데이터는 모든 내부 링크 코스트 데이터를 제공하지 않고 컨트롤러의 영역의 각 가능한 최단 경로 횡단의 코스트들을 제공한다. 네트워크 요소는 네트워크를 통해 데이터를 수신하도록 구성된 잉그레스(ingress) 모듈 및 네트워크를 통해 데이터를 송신하도록 구성된 에그레스(egress) 모듈을 포함한다. 네트워크 요소는 또한 잉그레스 모듈 및 에그레스 모듈에 결합된 네트워크 프로세서를 포함하고, 네트워크 프로세서는 컨트롤러 모듈, 토폴로지 학습 모듈, 최단 경로 계산 모듈, 이웃 발견 모듈 및 링크 상태 관리 모듈을 포함하는 모듈들의 세트를 실행하도록 구성된다. 토폴로지 학습 모듈은 컨트롤러의 영역 내의 각 경계 스위치를 포함하는 분할된 아키텍처 네트워크 내의 컨트롤러의 영역의 토폴로지를 결정하도록 구성되고, 여기에서 각 경계 스위치는 컨트롤러의 영역을, 네트워크의 또 하나의 영역 또는 네트워크 내의 전통적인 라우터에 링크하는 적어도 하나의 외부 포트를 가지고 있다. 컨트롤러 모듈은 분할된 아키텍처 네트워크에서 컨트롤러의 영역에 대해 제어 플레인 기능을 제공하도록 구성된다. 최단 경로 계산 모듈은 이웃 컨트롤러들 및 전통적인 라우터들과 공유될 컨트롤러의 영역 내의 각 경계 스위치 사이의 최단 경로를 식별하고 컨트롤러를 트리의 루트(root)로 하여 네트워크에 대한 최단 경로 트리를 계산하도록 구성된다. 이웃 발견 모듈은 헬로우 프로토콜을 이용하여 분할된 아키텍처 네트워크 내에서 각 이웃 컨트롤러를 식별하도록 구성되고, 링크 상태 관리 모듈은 각 이웃 컨트롤러와 링크 상태 데이터베이스를 교환하도록 구성된다. 링크 상태 데이터베이스는 컨트롤러의 영역의 각 경계 스위치 쌍 사이의 각 최단 루트(shortest route)의 코스트를 포함하고, 라우팅 테이블 저장 디바이스는 네트워크 프로세서에 통신 가능하게 결합된다. 라우팅 테이블 저장 디바이스는 컨트롤러의 영역에 대한 최단 경로 정보 및 컨트롤러의 영역의 경계 스위치들 사이의 최단 경로 정보를 포함하여 컨트롤러 모듈에 대한 라우팅 테이블들을 저장하도록 구성된다.

도면의 간단한 설명

[0009]

본 발명은 유사한 참조번호들이 유사한 구성요소들을 나타내는 첨부된 도면들의 그림들에서, 제한으로서가 아니라, 예로서 예시되어 있다. 유의할 점은, 본 개시에서 "하나의(an 또는 one)" 실시예에 대한 상이한 참조들은 반드시 동일한 실시예에 대한 것은 아니며, 그러한 참조들은 적어도 하나를 의미한다는 점이다. 또한, 특정 특징, 구조 또는 특성이 하나의 실시예와 관련하여 기재되는 경우에, 명시적으로 기재되어 있는지 여부에 관계없이 다른 실시예들과 관련하여 그러한 특징, 구조 또는 특성을 실행하는 것은 본 기술분야의 숙련자의 지식 내에 있는 것으로 사료된다.

도 1은 간단한 오픈플로우 네트워크의 아키텍처 예의 하나의 실시예의 다이어그램이다.

도 2는 OSPF(open shortest path first) 패킷 헤더의 다이어그램이다.

도 3은 OSPF 헬로우 패킷의 다이어그램이다.

도 4는 링크 상태 광고(LSA) 헤더 포맷의 다이어그램이다.

도 5는 라우터-LSA 메시지의 다이어그램이다.

도 6은 OSPF 외부 링크 코스트 정보가 교환되는 네트워크 예의 다이어그램이다.

도 7은 OSPF 내부 링크 코스트 정보가 교환되는 네트워크 예의 다이어그램이다.

도 8은 복수의 컨트롤러들을 가지는 예로 든 복수 영역 오픈플로우 네트워크의 하나의 실시예의 다이어그램이다.

도 9는 OSPF 프로세스를 구현하는 네트워크 요소의 하나의 실시예의 다이어그램이다.

도 10은 분할된 아키텍처 영역을 가지는 네트워크에서 OSPF 라우팅을 위한 프로세스의 하나의 실시예의 플로우 차트이다.

도 11은 OSPF 링크 상태 광고 헤더 포맷의 다이어그램이다.

도 12는 라우터 LSA 메시지의 다이어그램이다.

도 13은 분할된 아키텍처 OSPF 헬로우 메시지의 다이어그램이다.

도 14는 네트워크에서 헬로우 메시지를 분할된 아키텍처 컨트롤러로부터 전통적인 라우터에 전송하는 프로세스의 하나의 실시예의 다이어그램이다.

발명을 실시하기 위한 구체적인 내용

- [0010] 이하의 상세한 설명에서, 다수의 특정 세부사항들이 제시된다. 그러나, 본 발명의 실시예들은 이들 특정 세부사항없이도 실시될 수 있다는 것은 자명하다. 다른 예들에서, 공지된 회로들, 구조들 및 기술들은 본 설명의 이해를 모호하지 않게 하기 위해 상세하게 도시되어 있지 않았다. 그러나, 본 기술분야의 숙련자라면, 본 발명은 그러한 특정 세부사항들이 없이도 실시될 수 있다는 것을 잘 알고 있을 것이다. 본 기술분야의 통상의 기술자들이라면, 포함된 상세한 설명들로, 과도한 실험없이도 적절한 기능을 구현할 수 있을 것이다.
- [0011] 플로우 다이어그램들의 동작들은 도 7-9 및 도 14의 예로 든 실시예들을 참조하여 설명될 것이다. 그러나, 도 10의 플로우 다이어그램의 동작들은 도 7-9 및 도 14를 참조하여 설명된 것들 이외의 본 발명의 실시예들에 의해 수행될 수 있고 도 7-9 및 도 14를 참조하여 설명된 실시예들은 도 10의 흐름도를 참조하여 설명된 것들과 상이한 동작들을 수행할 수 있다.
- [0012] 도면들에 도시된 기술들은 하나 이상의 전자 디바이스들(예를 들면, 엔드 스테이션, 네트워크 요소, 서버 또는 유사한 전자 디바이스들) 상에 저장되어 실행되는 코드 및 데이터를 이용하여 구현될 수 있다. 그러한 전자 디바이스들은 비-일시적인 머신-판독가능 또는 컴퓨터-판독가능 저장 매체(예를 들면, 자기 디스크들, 광 디스크들, 랜덤 액세스 메모리, 판독 전용 메모리, 플래시 메모리 디바이스들, 및 상-변화 메모리(phase-change memory))와 같은 비-일시적인 머신-판독가능 또는 컴퓨터-판독가능 매체를 이용하여 코드 및 데이터를 저장하고 통신한다(내부적으로 및/또는 네트워크를 통해 다른 전자 디바이스들과). 뿐만 아니라, 그러한 전자 디바이스들은 통상적으로 하나 이상의 저장 디바이스들, 사용자 입력/출력 디바이스들(예를 들면, 키보드, 터치 스크린, 및/또는 디스플레이), 및 네트워크 접속들과 같은 하나 이상의 다른 컴포넌트들에 결합된 하나 이상의 프로세서들의 세트를 포함한다. 프로세서들의 세트 및 다른 컴포넌트들의 커풀링은 통상적으로 하나 이상의 버스들 및 브리지들(또한, 버스 컨트롤러들로도 지칭됨)을 통해서이다. 저장 디바이스들은 하나 이상의 비-일시적 머신-판독가능 또는 컴퓨터-판독가능 저장 매체 및 비-일시적 머신-판독가능 또는 컴퓨터-판독가능 통신 매체를 나타낸다. 그러므로, 주어진 전자 디바이스의 저장 디바이스는 통상적으로 그 전자 디바이스의 하나 이상의 프로세서들의 세트 상에서의 실행을 위해 코드 및/또는 데이터를 저장한다. 물론, 본 발명의 실시예의 하나 이상의 부분들은 소프트웨어, 펌웨어, 및/또는 하드웨어의 상이한 조합들을 이용하여 구현될 수 있다.
- [0013] 여기에 이용된 바와 같이, 네트워크 요소(예를 들면, 라우터, 스위치, 브리지, 또는 유사한 네트워킹 디바이스)는 네트워크 상의 다른 장비(예를 들면, 다른 네트워크 요소들, 엔드 스테이션들, 또는 유사한 네트워킹 디바이스들)를 통신가능하게 상호 접속시키는 하드웨어 및 소프트웨어를 포함하는 네트워킹 장비이다. 일부 네트워크 요소들은 복수의 네트워킹 기능들(예를 들면, 라우팅, 브리징, 스위칭, 레이어 2 어그리게이션, 세션 경계 제어, 멀티캐스팅, 및/또는 가입자 관리)에 대한 서포트를 제공하고, 및/또는 복수의 어플리케이션 서비스들(예를 들면, 데이터 수집)에 대한 서포트를 제공하는 "복수 서비스 네트워크 요소들"이다.
- [0014] **분할된-아키텍처 영역들**
- [0015] 단일 액세스/어그리게이션 네트워크는 복수의 전통적인 라우터들과 협력하여 작용하는 복수의 상이한 분할된-아키텍처 영역들로 구성될 수 있다. 여기에 이용된 바와 같이, 분할된-아키텍처 영역은 도메인에 유사한 분리된

라우팅을 가지는 분할된 아키텍처 네트워크의 섹션이다. 이것은 네트워크 강건성 또는 제어-플레인 스케일러빌리티를 위해 넓은 지리적 영역에 걸친 관리를 단순화하기 위해 수행될 수 있다. 각 분할된 아키텍처 영역은 분리된 컨트롤러에 의해 관리될 수 있다. 특정 어플리케이션에 따라, 이들 상이한 분할된-아키텍처 영역들의 컨트롤러들은 분할된 아키텍처 네트워크의 적절한 관리를 위해 일부 정보를 공유하고 교환할 필요가 있다.

[0016] OSPF(Open Shortest Path First) 라우팅

[0017] 전통적인 네트워크들 및 분할된 아키텍처 네트워크들 양쪽 모두는 스위치들과, 네트워크에 의해 서비스되는 다른 디바이스들 사이의 루트들을 계산해야 한다. OSPF(Open Shortest Path First)는 내부 게이트웨이 라우팅 프로토콜이다. OSPF(RFC 2328에 정의됨)는 라우터가 라우팅 도메인 내의 모든 노드들에게 그 이웃들의 링크-상태 정보를 브로드캐스팅하는 링크-상태 프로토콜이다. 이러한 정보를 이용하여, 모든 라우터는 도메인 내의 전체 네트워크의 토폴로지 맵을 구축한다. 각 라우터는 전체 네트워크 토폴로지를 반영하는 링크-상태 데이터베이스를 유지한다. 이러한 토폴로지 맵 및 링크 코스트 메트릭들에 기초하여, 라우터들은 다익스트라 알고리즘(Dijkstra's algorithm)을 이용하여 모든 다른 라우터들로의 최단 경로들을 결정한다. 이에 대해, 이러한 정보는 인터넷 프로토콜(IP) 패킷들의 포워딩에 이용되는 라우팅 테이블들을 생성하는데 이용된다.

[0018] OSPF는 관리의 용이함을 위해, 라우팅 도메인을, 영역 경계 라우터들(ABRs:Area Border Routers)에 의해 분리된 복수의 "영역들"로 분할하는 것을 가능하게 한다. 각 영역은 32 비트 식별자, 대개는 영역 내의 메인 라우터의 IP 어드레스에 의해 식별되고, 식별자 "0"은 OSPF 네트워크의 코어 또는 백본을 위해 예비된다(reserved). 링크 상태 정보의 브로드캐스트는 그 영역으로 제한되고 그 영역을 넘어 전송되지 않는다. OSPF 프로세스에서, 각 라우터(스위치)는 그것이 속하는 각 영역에 대해 OSPF 프로토콜 프로세스의 분리된 복사본을 운용한다. 하나의 라우터가 상이한 영역들에 속하는 복수의 인터페이스들을 구비한다면, 각 인터페이스에 하나씩, 프로세스의 복수의 복사본들을 운용한다. 부트-업 시에, 라우터의 OSPF 프로세스는 모든 라우팅 프로토콜 데이터 구조들을 초기화하고 더 낮은 레이어들로부터 그 라우터의 액티브 인터페이스들에 관한 정보를 획득한다. 그리고 나서, 라우터는 OSPF의 헬로우 프로토콜 패킷들을 이용하여 그 이웃들을 검출한다. 라우터는 헬로우 패킷들을 그 이웃들에게 송신하고 그 이웃들로부터 헬로우 패킷들을 수신한다. 브로드캐스트 및 점-대-점 네트워크 상에서, 헬로우 패킷들은 멀티캐스트 어드레스 224.0.0.5 상에서 송출된다. 비-브로드캐스트 네트워크들 상에서, 사용자 구성은 이웃들을 발견하도록 요구된다. 브로드캐스트 및 비-브로드캐스트 다중 액세스 네트워크들(NBMA) 상에서, 헬로우 프로토콜은 네트워크의 세그먼트에 대해 "지정된 라우터" 및 "백업 지정된 라우터"를 선택하는데 이용된다.

[0019] 이웃들의 검출 시에, 라우터는 그 새롭게 검출된 이웃들과 "어드제슨시들(adjacencies)"를 확립하려고 시도할 것이다. 어드제슨시들은 영역 내의 라우팅 정보의 분배를 결정한다. 라우팅 업데이트들은 단지 어드제슨시들 상에서만 송신되고 수신된다. 어드제슨시의 확립 시에, 라우터는 그 "링크-상태 데이터베이스"를 어드제슨시의 다른 엔드 상의 대응하는 라우터와 동기화시킬 것이다. 브로드캐스트 및 NBMA 네트워크들에 대해 지정된 라우터는 어느 라우터들이 인접하게(adjacent) 되는지를 결정한다.

[0020] 라우터들은 링크 상태 광고들(LSA)을 이용하여, "링크-상태"로도 지칭되는 그들의 상태를 주기적으로 광고한다. 링크-상태는 또한 라우터의 상태가 변경되는 경우에도 광고된다. LSA는 광고하는 라우터의 어드제슨시들을 포함한다. 라우터는 그 영역 전체에 대해 그 LSA를 플러딩한다. 플러딩(flooding) 알고리즘은 정보의 신뢰성을 보장하여, 그 영역 내의 모든 라우터들이 동일한 정확한 링크-상태 데이터베이스를 가지는 것을 확실하게 한다. 링크-상태 데이터베이스는 그 영역에 속하는 각 라우터에 의해 발원된 LSA들의 집합으로 구성된다.

[0021] 각 라우터는 이러한 데이터베이스를 이용하여, 그 자신을 루트(root)로 하여 최단 경로 트리를 계산한다. 이에 대해, 최단 경로 트리는 라우터의 라우팅 테이블을 생성하는데 이용된다. OSPF 메시지들은 TCP 또는 UDP와 같은 트랜스포트 레이어 프로토콜들의 이용없이, 송신되어 프로토콜 번호 89로 IP 패킷들에 직접 인캡슐레이팅된다. 추가적으로, OSPF는 그 자신의 에러 검출 및 정정을 이용한다.

[0022] OSPF는 5개의 상이한 패킷 타입들을 정의한다. 헬로우 패킷(Hello Packet): OSPF의 헬로우 프로토콜 패킷들은 이웃 관계들을 발견하고 유지하는데 이용된다. 데이터베이스 기술 패킷(Database Description Packet): 이들 패킷들은 어드제슨시들의 형성에 이용된다. 링크 상태 요구 패킷(Link State Request Packet): 이들 패킷들은 인접한 라우터들 사이에서 링크-상태 데이터베이스들을 다운로드하는데 이용된다. 링크 상태 업데이트 패킷(Link State Update Packet): 이들 패킷들은 OSPF의 신뢰성있는 업데이트 메커니즘에 이용된다. 단일 링크 상태 업데이트 패킷은 수 개의 라우터들의 LSA들을 포함할 수 있다. 링크 상태 승인 패킷(Link State Ack packet): 이들 패킷들은 OSPF의 신뢰성있는 업데이트 메커니즘에 대한 링크 상태 업데이트 패킷과 함께 이용된다.

다. OSPF 프로토콜 패킷들(헬로우 패킷들을 제외함)은 단지 어드레스시들을 통해서만 전송된다. 그러므로, 모든 OSPF 프로토콜 패킷들은 단일 인터넷 프로토콜(IP) 홉(hop)을 이동하고, 하나의 라우터의 IP 어드레스는 소스 어드레스로서, 다른 라우터의 IP 어드레스는 목적지 어드레스로서 가지고 있다.

[0023] RFC 2328에 정의된 OSPF는 5개의 상이한 타입들의 LSA들을 지정한다. 라우터-LSA: 이들 LSA들은 하나의 영역 내의 모든 라우터들에 의해 전송된다. 각 LSA는 하나의 영역으로의 라우터의 인터페이스들의 상태들을 포함한다. 라우터-LSA들은 단일 영역 전체를 통해서(throughout)만 플러딩된다. 네트워크-LSA: 이들 LSA들은 브로드캐스트 또는 NBMA 네트워크에 대해 지정된 라우터에 의해 발원된다. 이러한 LSA는 네트워크에 접속된 라우터들의 리스트를 포함한다. 라우터-LSA와 같이, 이러한 LSA는 또한 단일 영역 전체에 걸쳐서만 플러딩된다. 네트워크 요약 LSA: 이러한 LSA는 영역 경계 라우터(ABR)에 의해 발원되고 각 LSA는 영역 외부이지만 여전히 자율 시스템(AS:Autonomous System) 내부인 목적지 네트워크로의 루트를 기술한다. 바운더리 요약 LSA: 네트워크 요약 LSA와 같이, 이러한 LSA는 경계 영역 라우터(ABR)에 의해 발원되고, 이것은 AS 바운더리 라우터들로의 루트들을 기술한다. AS-외부-LSA: 이러한 LSA는 AS 바운더리 라우터들에 의해 발원되고, 이들은 AS 전체에 걸쳐 플러딩된다. 이들 LSA들의 각각은 또 하나의 AS의 목적지로의 루트를 기술한다. OSPF 패킷 헤더 포맷은 도 2에 도시된 바와 같다. 헬로우 패킷 포맷은 도 3에 도시되어 있다. LSA 헤더 포맷은 도 4에 도시되어 있다. 라우터 LSA 패킷 포맷은 도 5에 도시된 바와 같다.

[0024] **분할된-아키텍처 네트워크에서의 OSPF**

[0025] 분할된 아키텍처 네트워크에서, 모든 다른 제어 메시지들과 같은 라우팅 메시지들은 컨트롤러들 사이에서 교환된다. 분할된 아키텍처에서 OSPF를 구현하는 하나의 간단한 방법은 내부 코스트들을 무시하고 단지 외부 코스트들 - 즉 상이한 네트워크 영역들에 접속하는 링크들의 코스트들 -에 관한 정보만을 교환하는 것이다. 그렇게 함으로써, 전통적인 OSPF 메시지들은 컨트롤러들 사이에서 교환될 수 있고, 각 분할된 아키텍처 영역은 단일 노드로서 간주될 수 있다.

[0026] 도 6에 도시된 네트워크에서, 예를 들면, 컨트롤러 B는 그것이 10 유닛들의 코스트로 영역 C로의 링크를 가지고 있다고 알린다. 그러나, 이러한 정보에 기초하여 발견된 루트들은 차선책이다. 이러한 구현에서 내부 코스트들은 다른 컨트롤러들에게 제공되지 않기 때문에, 컨트롤러들은 실제 엔드-대-엔드 코스트들에 기초하여 결정을 수행할 수 없을 것이다. 이것은 하나의 영역 내의 내부 코스트들이 내부 경로 별로 상당히 가변되는 경우에 차선 루트들로 나타난다.

[0027] 하나의 대안 실시예에서, 차선 루트들의 상기 문제는 컨트롤러들 사이에 교환되는 정보에 내부 코스트들을 포함 시킴으로써 고칠 수 있다. 각 컨트롤러는 내부-링크 코스트를 계산하고 그 경계 스위치들의 임의의 2개 사이의 최단 경로(즉, 최소-비용 경로)를 구할 수 있다. 예를 들면, 다시 도 6을 참조하면, 컨트롤러 C가 그 자신의 영역과 컨트롤러 F의 영역 사이의 링크 코스트를 알리는 경우에, 경계 스위치들 S1 및 S2 사이의 최단-경로 코스트를 추가하고, 그것을 컨트롤러 B에 전송되는 코스트 정보에 추가한다. 유사하게, 컨트롤러 C는 스위치들 S2 및 S3 사이의 최단-경로 코스트를 영역들 C 및 F 사이의 코스트에 추가하고, 이러한 정보를 컨트롤러 D에 전송한다.

[0028] 이러한 대안 실시예에는 또한 문제들을 발생시킨다. 이러한 실시예는 동일한 링크에 대해 복수(그리고 가능하게는 상이한) 링크 코스트 메시지들을 전송하는 것을 요구한다. 예를 들면, 영역 F에 도달하는데 컨트롤러 C에 의해 통보되는 코스트는 이러한 정보가 컨트롤러 B에 전송되는 경우와 컨트롤러 D에 전송되는 경우가 상이한데, 이는 이들이 각각 상이한 내부-경로 코스트(예를 들면, S1과 S2 사이의 내부 최단-경로 코스트가 S2와 S3 사이의 코스트와 상이하다면, 컨트롤러 C는 그 자신의 영역과 컨트롤러 F의 영역 사이의 2개의 상이한 링크 코스트들을 통보할 것이다)에 기초하기 때문이다. 결국, 이들 2개의 반대되는 메시지들은 다른 컨트롤러들에 의해 수신될 것이고, 이는 이들을 2개의 분리된 메시지들로서 해석하기보다는 제1 수신된 메시지에 대한 업데이트로서 제2 메시지를 해석할 것이다.

[0029] 예를 들면, OSPF 프로토콜을 변형하여 복수의 코스트들을 동일한 링크로 돌리는 것을 허용함으로써, 이러한 복수-코스트 문제를 해결하려고 시도하는 추가 실시예들에 대해, 분할된 아키텍처에서 컨트롤러들에 의해 발견된 최단 경로는 최적의 것이 아닐 수 있다. 예를 들면, 도 7에 도시된 시나리오에서, 각 링크 상의 숫자들이 어느 방향으로든 링크들의 코스트들을 보여주는 경우에, 차선 루트들이 결정될 것이다. 영역 A에서 영역 F로의 최단 경로를 계산할 때, 이하의 단계들이 취해질 것이다. (1) C는 B 및 D에 통보한다: F에 도달하는데 코스트 2(1+1), (2) B는 A에 통보한다: C에 도달하는데 코스트 11(10+1), (3) D는 B에 통보한다: C에 도달하는데 코스트 21(1+20)(유의할 점은 영역 D 내의 경계 스위치들 S4 및 S5 사이에 단지 하나의 내부 경로가 있다는 점)

다), 및 (4) B는 A에게 통보한다: C에 도달하는데 코스트 2(1+1).

- [0030] 컨트롤러 A가 상기 정보에 기초하여 영역 F로의 최단 경로를 계산하는 경우에, 14인 그 최소 코스트로 인해 다른 경로들에 비해 경로 A-B-C-F를 선택한다. 그러나, A와 F 사이의 최적 라우팅은, 도 7에 도시된 모든 코스트 정보에 기초하여, A-B-D-E-D-C-F로서, 단지 10의 코스트이다. 컨트롤러 A는 컨트롤러들 사이에 교환되는 정보가 그러한 선택에 충분하지 않으므로 이러한 경로를 선택할 수 없다. 그러므로, 최적 라우팅 솔루션을 위해, 컨트롤러들 사이에 교환되는 OSPF 메시지들은 이하에 더 설명되는 바와 같이, 내부-링크 코스트에 관한 추가 정보를 전송하는 것이 필요하다.
- [0031] **추상화된 인트라-영역 경로 코스트들을 가지는 OSPF**
- [0032] 본 발명의 실시예들은 종래 기술의 단점들을 회피하기 위한 방법 및 시스템을 제공한다. 종래 기술 및 상기 제시된 바와 같은 분할된-아키텍처 네트워크에서의 OSPF의 간단한 구현들은 차선 라우팅 솔루션들을 제공하고, 여기에서 최단 경로는 항상 정확하게 결정되는 것이 아니고, 및/또는 과도한 양의 정보가 불충분하거나 스케일링하지 않은 루트들을 식별하기 위해 제공된다.
- [0033] 본 발명의 실시예들은 종래 기술의 이들 단점들을 극복한다. 본 발명의 실시예들은 경계 스위치들의 각 쌍 사이의 인트라-영역 경로 코스트들을 코스트 값과 연관된 직접적인 링크로서 추상화한다. 이러한 솔루션은 최적 경로들을 제공하고, 효율적으로 실행될 수 있고, 스케일링가능하며, 비-분할된-아키텍처들의 종래의 라우터들과 후방 호환가능하다(backward compatible).
- [0034] 하나의 실시예에서, OSPF 라우팅 프로토콜은 분할된 아키텍처에서 컨트롤러들 사이에서 구현되어, 네트워크의 임의의 2개의 포워딩 요소들 사이에서 최적(최단) 경로들을 확립한다. 실시예는 인트라-영역 경로 코스트 및 인터-영역 링크 코스트 양쪽 모두의 필요한 정보를 전체 멀티-영역 분할된 아키텍처 네트워크에 노출한다. 상기 언급된 바와 같이, 실시예들은 최적합성, 효율, 스케일러빌리티, 및 후방 호환성을 제공한다.
- [0035] 라우팅 프로토콜은 각 분할된-아키텍처 컨트롤러가 임의의 다른 목적지에 도달하는 최적 경로를 개별적으로 계산하도록 허용하는 충분하고 정확한 정보를 제공한다. 이것은 전통적인 인트라-도메인 라우팅 프로토콜(OSPF, IS-IS)이 전통적인 네트워크들에서 제공하는 최적화 속성이다. 그러나, 최적 경로의 정의는 분할된-아키텍처 네트워크들에서의 이용을 위해 약간 변경된다. 전통적인 네트워크에서, 그것은 최소 인터-라우터 코스트들을 가지는 경로이다. 분할된-아키텍처 네트워크 컨텍스트에서, 최적 경로는 최소 인터-라우터 코스트 플러스 인트라-라우터 코스트(또는 인트라 SA-영역 코스트)를 가지는 경로이다.
- [0036] 분할된-아키텍처 OSPF(SA-OSPF) 라우팅 프로토콜의 실시예들은 각 라우터가 최단 경로 결정을 독립적으로 계산할 수 있게 한다. OSPF는 빠른 속도로 수렴한다. 환언하면, 분할된-아키텍처 OSPF 프로토콜은 전통적인 네트워크 OSPF에 비해 추가적인 컨버전스 오버헤드를 유발시키지 않는다. 분할된-아키텍처 OSPF 프로토콜은 수백 개의 스위치들을 가지는 큰 네트워크로 스케일링한다. 스케일러빌리티는 교환되는 다수의 메시지들의 오버헤드 및 컨트롤러들 상의 저장 요구조건 양쪽 모두에 의해 계량화될 수 있다. 점진적 전개는 네트워크 프로토콜들에 대한 임의의 새로운 제안들의 선정에 핵심적이다. 분할된-아키텍처 OSPF 프로토콜은 종래의 라우터들과 후방 호환가능하다. 이러한 속성은 실제 네트워크 환경에서 그 잠재적인 이용을 증가시킨다.
- [0037] 분할된-아키텍처 OSPF의 실시예들은 경계 스위치들의 임의의 쌍들 사이의 인트라-영역 경로 코스트를 코스트 값과 연관된 직접 링크로서 추상화한다. 이러한 기능을 용이하게 하는 분할된-아키텍처 OSPF의 수 개의 양태들이 있다. 인트라-영역 코스트는 후방 호환성을 보장하도록 종래 OSPF 메시지들의 형태로 임베디드된다. 인트라-영역 코스트는 인터-영역 코스트와는 별개로, 모든 영역들의 모든 컨트롤러들에게 전파된다. 이것은 최단 경로 계산의 최적합성을 보장한다. 이는 각 컨트롤러가 전체 네트워크의 완전한 픽처, 즉, 인터-영역 및 인트라-영역 모두를 가지는 것을 보장한다. 그러한 충분한 영역을 가진 상태에서, 최단 경로를 계산하는 것은 전통적인 OSPF에서와 같이, 다익스트라 알고리즘을 이용하여 용이하게 수행될 수 있다.
- [0038] 분할된-아키텍처 OSPF는 외부 컨트롤러들에게 어느 내부 정보가 전파되어야 하는지가 주의깊게 관리된다는 점에서 스케일링가능하다. 하나의 순진한 접근법은 분할된-아키텍처 스위치들의 임의의 쌍들 사이에서 내부 코스트들을 전송하는 것일 것이다. 그러나, 이것은 스케일러빌리티 이슈를 유발할 것이다. 더구나, 이러한 정보의 대부분은 외부 컨트롤러 상에서 수행하는 결정에 유용하지 않다. 분할된-아키텍처 OSPF는 경계 분할된-아키텍처 스위치들의 임의의 쌍의 전체 집계된(total aggregated) 코스트만을 전파한다.
- [0039] 도 8은 분할된-아키텍처 네트워크의 하나의 예로 든 실시예의 다이어그램이다. 예로 든 분할된-아키텍처 네트워크는 분리된 분할된-아키텍처 영역들(SAs, 801A-C)로 분할된다. 각 영역(801A-C)은 스위치들의 세트를 포함

한다. 동일한 영역 내의 모든 스위치들은 단일 논리적 컨트롤러(803A-C)에 의해 제어된다. 하나의 실시예에서, SA들은 주 컨트롤러, 및 리던던시 목적을 위한 백업 컨트롤러들의 세트로서 구현될 수 있다.

[0040] 각 SA 내의 스위치들은 분할된 아키텍처 네트워크의 데이터 플레인을 구현할 수 있는 임의의 타입의 라우터, 스위치 또는 유사한 네트워킹 디바이스일 수 있다. 스위치들은 경계 분할된-아키텍처 스위치들 및 내부 분할된-아키텍처 스위치들을 포함할 수 있다. 경계 분할된-아키텍처 스위치는 상이한 SA 영역들의 또 하나의 스위치에 접속하는 인터페이스를 가지고서 분할된-아키텍처 기능들을 지원한다. 경계 분할된-아키텍처 스위치는 통상적으로 단일 SA 영역의 컨트롤러에 의해 제어된다. 다른 실시예들에서, 경계 분할된-아키텍처 스위치는 복수의 SA들 내에 있을 수 있고, 각각의 SA 컨트롤러에 의해 제어되는 인터페이스들을 가지고 있다. 내부 분할된-아키텍처 스위치는 분할된-아키텍처 프로토콜들을 지원한다. 이는 그 영역 내의 컨트롤러에 의해 제어된다. 모든 그 이웃들은 동일한 SA 영역 내에 있다.

[0041] 스위치들은 링크들의 세트를 통해 서로와 통신 상태에 있다. 이들 링크들은 유선 또는 무선 통신 매체들 및 그 임의의 조합을 포함하는 임의의 타입의 통신 매체일 수 있다. 링크들은 내부 링크들 또는 외부 링크들로 카테고리화될 수 있다. 내부 링크는 SA 영역 내의 2개의 스위치들 사이의 링크들이고, 이들 스위치들은 경계 스위치들 또는 내부 SA 영역 스위치 중 어느 하나일 수 있으며, 이들은 동일한 SA 영역에 속한다. 외부 링크들은 상이한 SA 영역들에 속하는 2개의 SA 스위치들 사이의 링크들이다. 이 경우에, SA 스위치들 양쪽 모두는 경계 SA 스위치들이다.

[0042] 링크 상태 광고들(LSA들, 805)은 분할된-아키텍처 OSPF를 구현하는 예로 든 LSA들의 세트이다. 각 LSA는 경계 스위치들의 쌍, 및 이들 2개의 경계 스위치들 사이의 SA 영역을 횡단하는 연관된 코스트들의 추상화이다. 이들 LSA들은 각 컨트롤러에 의해 생성되어, 인접하는 컨트롤러들에게 송신된다. 예로 든 LSA 세트(805)는 컨트롤러(803B) 및 SA 영역(801B)에 대한 LSA들의 세트이다.

[0043] 도 9는 컨트롤러를 구현하는 네트워크 요소의 하나의 실시예의 다이어그램이다. 하나의 실시예에서, 컨트롤러(901)는 라우터, 스위치 또는 유사한 네트워킹 디바이스이다. 컨트롤러(901)는 인그레스 모듈(903), 에그레스 모듈(905), 네트워크 프로세서(907), 및 저장 디바이스(911)를 포함할 수 있다. 인그레스 모듈(903)은 물리적 및 링크 레벨에서 인커밍 데이터 트래픽을 핸들링하고, 추가 처리를 위해 이러한 데이터를 네트워크 프로세서에 제공한다. 유사하게, 에그레스 모듈(905)은 물리적 및 링크 레벨에서 아웃고잉 데이터 트래픽을 핸들링하여 이를 접속된 네트워크를 통해 다른 디바이스들에게 송신한다. 이들 2개의 모듈들은 네트워크를 통해 다른 디바이스들과의 통신을 가능하게 하도록 함께 기능한다.

[0044] 네트워크 프로세서(907)는 네트워크의 데이터 플레인을 관리하는 네트워크의 제어 플레인에 관련된 각각의 기능들을 포함하는 네트워크 요소의 기능들을 실행하는 처리 디바이스 또는 처리 디바이스들의 세트이다. 네트워크 프로세서(907)는 이웃 발견 모듈(913), OSPF 모듈(915), 토폴로지 학습 모듈(917), 링크 상태 관리 모듈(919) 및 오픈플로우 컨트롤러(921)와 같은 컨트롤러 모듈들을 포함하는 모듈들의 세트를 실행시킬 수 있다.

[0045] 뿐만 아니라, 네트워크 프로세서(907)는 저장 디바이스(911) 내에 저장된 데이터에 액세스할 수 있다. 저장 디바이스(911)에 저장된 데이터는 라우팅 테이블들(923) 및 링크 상태 데이터베이스(925)를 포함할 수 있다. 다른 실시예들에서, 저장 디바이스(911)는 임의의 개수의 분리된 로컬 또는 분산된 저장 디바이스들 및 이들 디바이스들에 걸친 저장된 데이터의 임의의 배열을 포함할 수 있다. 네트워크 프로세서에 의해 실행되는 다른 모듈들은 저장 디바이스(911)로부터 로딩되거나 거기에 저장될 수 있다.

[0046] 이웃 발견 모듈(913)은 네트워크에서 이웃하는 컨트롤러들의 각각에 관한 정보를 획득하여 컨트롤러에 의해 관리되는 SA 영역의 스위치들 사이의 적절한 통신 및 그들의 구성을 가능하게 하도록, 헬로우 프로토콜 또는 유사한 프로토콜을 이용하여 네트워크 내의 다른 디바이스들과 통신하기 위한 프로토콜들을 관리할 수 있다. 임의의 헬로우 프로토콜 또는 프로세스는 SA 영역에 대한 인접하는 컨트롤러들 및 스위치들을 식별하는데 이용될 수 있다.

[0047] 토폴로지 학습 모듈(917)은 이웃 발견 모듈(913)에 의해 수집된 정보를 활용하여, 컨트롤러가 동작하는 네트워크의 토폴로지를 결정한다. 이러한 토폴로지 정보는 OSPF 모듈(915)에 의해 활용되어, 네트워크를 통한 최적 루트들을 계산한다. 토폴로지 정보는 또한 링크 상태 관리 모듈(919)에 의해 활용되어, 네트워크 내의 링크 코스트들을 추적하고 결정한다.

[0048] OSPF 모듈(917)은 소스 또는 발원하는 디바이스로부터 네트워크 내의 목적지 디바이스 사이의 최적 루트들을 계산한다. OSPF 모듈(917)은 라우팅 정보를 라우팅 테이블들(923)의 세트에 저장할 수 있다. OSPF 모듈(917)은

토폴로지 학습 모듈(917)에 의해 생성된 토폴로지 정보를 이용하여, 네트워크 요소들 사이의 루트들을 계산할 수 있다. 일부 실시예들에서, OSPF 모듈(917)은 각각의 다른 디바이스로의 루트들을 확립하도록, 루트(root)에서 컨트롤러를 가지고서 네트워크에 대한 스패닝(spanning) 트리를 계산할 수 있다.

[0049] 링크 상태 관리 모듈(919)은 링크 상태 데이터베이스(925) 내의 네트워크 또는 네트워크의 영역에 대한 링크 상태 정보를 관리한다. 링크 상태 관리 모듈(919)은 또한 컨트롤러의 SA 영역 내뿐만 아니라 컨트롤러와 다른 컨트롤러들 사이의 접속들에 관한 정보를 제공하는 인접하는 컨트롤러들에게 유포되는 링크 상태 광고들을 생성할 수 있다. 이러한 정보는 네트워크의 다른 컨트롤러들에게 전송되는 각 경계 스위치 쌍에 대한 링크 상태 광고들의 세트로서 패키징될 수 있다.

[0050] 컨트롤러(921)는 SA 영역의 제어 플레인을 관리하는 임의의 타입의 분할된-아키텍처 컨트롤러일 수 있다. 컨트롤러는 분할된 아키텍처 네트워크를 관리하기 위한 오픈플로우 프로토콜 또는 유사한 프로토콜을 구현할 수 있다. 컨트롤러 모듈(921)은 SA 영역 내의 스위치들과 통신하여 데이터 플레인 상에서 패킷들의 포워딩을 구성한다. 컨트롤러(921)는 또한 다른 컨트롤러들과 통신하여, 이웃 정보, 링크 상태 광고들을 교환하고 유사한 정보를 피어들에게 제공한다.

[0051] 도 10은 분할된-아키텍처 네트워크에서 OSPF를 구현하고 지원하는 컨트롤러의 동작의 하나의 실시예의 플로우차트이다. 프로세스는 명료성을 위해 컨트롤러에 의해 실행된다는 측면에서 기재되어 있지만, 프로세스는 특정 컨트롤러 모듈(예를 들면, 오픈플로우 컨트롤러)과 함께 컨트롤러의 컴포넌트(예를 들면, OSPF 모듈)에 의해 실행될 수 있다. 프로세스는 컨트롤러가 활성화되거나 리셋되는 때에 초기화될 수 있다. 다른 실시예들에서, 프로세스는 연속적이거나 주기적이다. 컨트롤러는 OSPF 라우팅 프로토콜 관련 데이터 구조들과 같은 데이터를 저장하고 조작하는데 활용되는 데이터 구조들의 세트를 초기화할 수 있다(블록 1001).

[0052] 컨트롤러는 경계 스위치들인 SA 영역 내의 이들 스위치들을 식별하는 것을 포함하여 그 할당된 SA 영역의 토폴로지를 학습한다(블록 1003). 토폴로지는 다른 컨트롤러들 및 전통적인 라우터들과의 링크 상태 광고들 또는 링크 상태 데이터베이스들의 교환을 통해, 그리고 유사한 메커니즘들을 통해 학습될 수 있다. 상이한 SA 영역들 내의 컨트롤러들은 SA-OSPF 프로토콜과 통신할 수 있다. 교환되는 메시지들은 헬로우 메시지들, 데이터베이스 기술(Database description), 링크 상태 요구, 링크 상태 업데이트, 및 링크 상태 승인(Link State Acknowledgement)을 포함하며, 전통적인 OSPF 메시지들과 유사하다. 특히, 도 8에서, 링크 상태 업데이트 메시지(805)가 도시되어 있다. 단일 링크 상태 업데이트 패킷은 수 개의 링크 상태 광고들(LSAs)을 포함할 수 있다. SA-OSPF의 기본 라우팅 알고리즘의 분리된 복사본이 각 영역에서 운용된다. 복수의 영역들로의 인터페이스들을 가지는 라우터들은 알고리즘의 복수의 복사본들을 운용한다. 이러한 데이터는 SA 영역, 및 SA 영역이 위치하는 더 넓은 네트워크의 토폴로지 맵으로 컴파일링된다.

[0053] 컨트롤러(예를 들면, 컨트롤러의 OSPF 모듈)는 컨트롤러의 SA 영역 내의 경계 스위치들의 각 쌍 사이의 최단 경로를 계산한다(블록 1005). 이것은 스위치들 사이의 다른 인트라-영역 루트들을 계산하는 것과 함께 수행될 수 있다. 그렇지만, 경계 스위치 쌍 코스트들은 이들 컨트롤러들 및 전통적인 라우터들이 컨트롤러의 SA 영역을 횡단하는 최적 경로들을 결정할 수 있도록 다른 컨트롤러들 및 전통적인 라우터들과 공유해야 하는 것이다. 최단 경로는 SA 영역의 학습된 토폴로지를 이용하여 계산된다. 하나의 실시예에서, 다익스트라의 알고리즘이 경계 스위치들의 임의의 쌍들 사이의 최단 경로를 계산하는데 이용된다. 그리고나서, 각 쌍에 대한 최단 경로의 계산된 코스트들은 토폴로지 데이터 또는 OSPF 라우팅 데이터와 함께 저장된다(블록 1007).

[0054] 그리고나서, 컨트롤러는 이웃 컨트롤러들을 식별한다(블록 1009). 하나의 실시예에서, 컨트롤러는 SA-OSPF의 헬로우 프로토콜을 이용하여 이웃들을 획득할 수 있다. 컨트롤러는 헬로우 패킷들을 그 이웃하는 영역의 또 하나의 컨트롤러에게, 또는 이웃하는 전통적인 라우터에 전송하고, 이에 대해 다른 컨트롤러들 및 전통적인 라우터들로부터 헬로우 패킷들을 수신한다(블록 1111). 브로드캐스트 및 점대점 네트워크들 상에서, 컨트롤러는 그 헬로우 패킷들을 멀티캐스트 어드레스에 전송함으로써, 그 이웃하는 영역들 내의 컨트롤러들 또는 전통적인 라우터들을 다이나믹하게 검출한다. 비-브로드캐스트 네트워크들 상에서, 일부 구성 정보는 이웃하는 컨트롤러들을 발견하도록 제공될 수 있다.

[0055] 다음으로, 컨트롤러는 그 새롭게 획득된 이웃하는 컨트롤러들의 일부와 어드레스시들을 형성하려고 시도할 것이다. 링크-상태 데이터베이스들은 인접하는 컨트롤러들과 LSA 메시지들을 교환함으로써(블록 1115) 인접하는 컨트롤러들의 쌍들 사이에서 동기화된다(블록 1113). 각 LSA 메시지는 그 전체 외부 링크들의 코스트를 포함한다. 컨트롤러는 특정 이웃에게, 이 이웃에게 접속하는 링크들의 코스트를 전송할 뿐만 아니라, 링크의 코스트를 모든 이웃들에게 전송한다. 외부 링크들의 코스트를 송출할 뿐만 아니라, 컨트롤러는 또한 내부 링크

들의 코스트를 송출할 것이다. LSA 메시지는 2개의 관련된 인터페이스 ID들과 함께, 2개의 경계 스위치들의 SA 스위치 ID들을 이용하여 구성될 수 있다. 컨트롤러는 링크 상태로도 불리는데, 그 SA 영역의 상태를 주기적으로 광고할 수 있다. 링크 상태는 또한 영역(컨트롤러)의 상태가 변경되는 경우에 광고된다. 영역의 어드레스시들은 그 LSA들의 콘텐츠에서 반영된다. 어드레스시들과 링크 상태 사이의 이러한 관계는 프로토콜이 오류 및 성능 변경들을 시의적절한 방식으로 검출할 수 있게 한다.

[0056] LSA들은 복수의 SA 영역들에 걸쳐 플러딩된다. 플러딩 프로세스는 신뢰성있고, 전체 네트워크의 모든 컨트롤러들이 정확하게 동일한 링크-상태 데이터베이스를 가지는 것을 보장한다. 이러한 데이터베이스는 인터-영역 토폴로지 및 인트라-영역 토폴로지 모두를 포함하여 각 영역에 의해 발원되는 LSA들의 집합으로 구성된다.

[0057] 이러한 링크 상태 데이터베이스로부터, 각 컨트롤러는 그 자신을 루트(root)로 하여, 전체 네트워크에 대한 최단 경로 트리를 계산한다(블록 1117). 이에 대해, 이러한 최단 경로 트리는 OSPF 프로토콜에 대한 라우팅 테이블을 산출한다. 이러한 라우팅 정보에 기초하여, 컨트롤러는 그 SA 영역 내의 스위치들을 프로그래밍하여, 계산된 최단 경로를 이용하여 SA 영역을 통해 데이터 패킷들을 라우팅한다(블록 1119). 스위치들의 업데이트는 보안 채널을 통해, 또는 유사한 프로세스를 이용하여 제어 플레인 프로토콜(예를 들면, 오픈플로우)을 이용하여 달성될 수 있다.

[0058] 네트워크 업데이트된 링크 상태에서 변경들이 발생함에 따라, 광고들이 컨트롤러에 의해 전송될 수 있다(블록 1121). LSA 메시지들은 인접하는 컨트롤러들에 전송되어, 이웃하는 컨트롤러들이 그들 라우팅을 필요한 대로 업데이트할 수 있도록, 이들에게 컨트롤러의 SA 영역 내의 변경들을 통지한다.

[0059] 분할된-아키텍처 OSPF는 전통적인 라우터가 네트워크 내의 다른 라우터들에게 단일 라우터로 보인다는 점에서 전통적인 OSPF와 상이하다. 그러나, SA 영역은 네트워크의 다른 영역들/라우터들에게 N개의 라우터들로 보일 것이고, 여기에서 N은 영역의 경계 분할된-아키텍처 스위치들의 개수이다. 차이점은, 전통적인 OSPF는 영역당 단일 라우터 상에서 단일 프로세스를 운용하고 단일 라우터로 보이지만, SA-OSPF는 SA 영역의 단일 컨트롤러 상에서 단일 프로세스를 운용하고 다른 컨트롤러들에게 N개의 라우터들로 보인다는 점이다.

[0060] 하나의 실시예에서, 단일 SA-OSPF 프로세스는 전통적인 라우터의 OSPF 프로세스에 의해 전송된 매 헬로우 패킷에 대해, N개의 경계 SA 스위치들의 각각을 나타내는 N개의 상이한 헬로우 패킷을 전송한다. 이들 헬로우 패킷들은 다른 라우터들/SA 영역들에 접속된 인터페이스들 상에서 전송된다. 유사하게, 단일 SA-OSPF 프로세스는 전통적인 라우터의 OSPF 프로세스에 의해 전송된 매 라우터-LSA 패킷에 대해, N개의 경계 SA 스위치들의 각각을 나타내는 N개의 상이한 라우터-LSA 패킷들을 전송한다. 이들 LSA 패킷들은 다른 라우터들/SA 영역들에 접속된 인터페이스들 상에서 전송된다.

[0061] 요약하면, SA 영역에 대한 컨트롤러는 그것이 N개의 OSPF 컨트롤러 프로세스들을 가지고 있는 것처럼 OSPF를 관리하고, 여기에서 N은 SA 영역의 경계 분할된-아키텍처 스위치들의 개수이다. 이것은 접속된 전통적인 라우터들 상에서 운용되는 OSPF와의 후방 호환성을 보장한다. 그러므로, 컨트롤러는 N개의 헬로우 패킷들, N개의 LSA 메시지들, 등을 송출한다. 전통적인 네트워크들의 동일한 개수의 라우터들에 대해, 분할된-아키텍처 네트워크들에서, SA-OSPF는 전통적인 OSPF에 비해 더 적은 개수의 프로토콜 패킷들을 이용한다. 절감들은 동일한 SA 영역에 속하는 경계 SA 스위치들 사이에서 OSPF 프로토콜 패킷들을 전송할 필요가 없다는 점에서 나온다.

[0062] **LSA 메시지들**

[0063] 도 11은 OSPF 링크 상태 광고 헤더 포맷의 다이어그램이다. SA-OSPF가 후방 호환가능하도록 보장하기 위해, 표준 OSPF 메시지 포맷과 동일한 포맷(즉, 동일한 필드들의 시퀀스 및 필드들의 길이)이 활용된다. 그러나, 일부 필드들의 값들의 할당이 상이할 수 있다.

[0064] 도 12는 SA-OSPF LSA 메시지, 구체적으로는 라우터-LSA 타입 메시지의 다이어그램이다. 라우터-LSA는 하나의 영역 내의 각 스위치에 의해 생성된다. 이것은 그 영역 내의 영역의 인터페이스들의 상태를 기술한다. 이러한 메시지는 모든 컨트롤러들에 의해 발원된다. 이러한 LSA는 멀티-도메인 SA 네트워크에 대한 라우터의 인터페이스들의 수집된 상태들을 기술한다. 이는 모든 컨트롤러들 전체에 걸쳐 플러딩된다.

[0065] 하나의 실시예에서, 라우터-LSA 메시지는 메시지(도 12에 도시됨)의 4개의 필드들: 셀프 스위치(Self Switch) ID, 셀프 인터페이스(Self Interface) ID, 이웃 인터페이스 ID, 및 이웃 스위치 ID에서 전통적인 라우터-LSA 메시지와 상이하다. 스위치 ID는 분할된 아키텍처에서 각 스위치에 할당된 고유한 32-비트 식별자이다. 이러한 ID는 멀티-영역 SA 도메인의 모든 스위치들 간에 고유해야 한다. 하나의 실시예에서, 스위치의 최고 IP 어드레스가 스위치 ID로서 이용된다. 셀프 스위치 ID는 광고하는 스위치의 식별자이고, 이웃 스위치 ID는 이웃하는

스위치의 식별자이다. 인터페이스 ID는 스위치의 각 인터페이스에 할당된 32-비트 식별자이다. 이러한 식별자는 단지 단일 스위치 내에서 고유할 필요가 있다. 셀프 인터페이스 ID는 광고하는 스위치의 인터페이스 ID이고, 이웃 스위치 ID는 이웃하는 인터페이스(이웃하는 스위치의)의 인터페이스 ID이다.

[0066] **헬로우 메시지들의 핸들링**

[0067] OSPF 헬로우 프로토콜은 라우터가 이웃하는 라우터들과 어드제슨시를 확립하고 유지할 수 있게 하는 메커니즘이다. 인접하는 라우터들은 헬로우 메시지들을 교환한다. 브로드캐스트 및 점-대-점 네트워크들에서, 라우터는 헬로우 메시지들을 멀티캐스트 어드레스에 전송함으로써 그 이웃하는 라우터들을 다이내믹하게 검출할 수 있다. 헬로우 패킷들을 수신할 때, 라우터는 이웃들 사이에서 어드제슨시들을 형성한다. 링크-상태 데이터베이스들은 라우팅 업데이트들뿐만 아니라 인접하는 라우터들의 쌍들 사이에서 동기화된다. 어드제슨시가 확립된 이후에, 라우터는 그것이 살아있다는 것을 나타내도록 헬로우 메시지들을 여전히 주기적으로 교환할 필요가 있다. 요약하면, 헬로우 메시지의 목적은 이웃하는 관계들을 발견하고 유지하는 것이다.

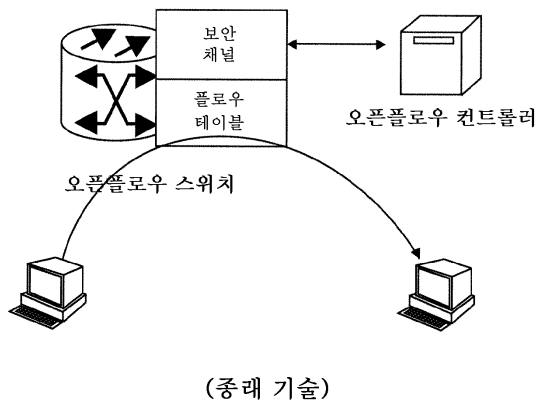
[0068] 도 13은 SA-OSPF의 헬로우 메시지의 다이어그램이다. SA-OSPF 프로토콜에서, 헬로우 메시지는 변경되지 않을 수 있다. 그러나, 분할된-아키텍처 컨트롤러는 이웃하는 라우터들과, 그것이 접속하는 분할된 아키텍처 스위치 사이의 어드제슨시들을 유지하도록 추가적인 헬로우 메시지들을 구성할 필요가 있다.

[0069] 도 14는 분할된-아키텍처 컨트롤러와 전통적인 종래의 라우터 사이에서 교환되는 헬로우 메시지의 다이어그램이다. SA-OSPF에서, 컨트롤러는 우선 경계 스위치들 대신에 OSPF 헬로우 메시지를 구성한다. 그것은 스위치 ID의 필드에서 경계 스위치의 최고 IP 어드레스를 이용한다. 패킷은 우선 스위치 S1에 주입되고, 그리고 나서 에그레스 링크 상의 인접하는 라우터들에 전송된다.

[0070] 그러므로, 분할된-아키텍처 네트워크에서 OSPF를 구현하기 위한 방법, 시스템 및 장치가 기재된다. 상기 설명은 예시적인 것으로 제한하려는 것이 아니라는 것은 자명하다. 다수의 다른 실시예들은 상기 기재물을 읽고 이해하는 경우에 본 기술분야의 숙련자들에게 명백할 것이다. 그러므로, 본 발명의 범주는 그러한 청구항들이 권리를 주는 등가물들의 전체 범주와 함께, 첨부된 청구항들을 참조하여 결정되어야 한다.

도면

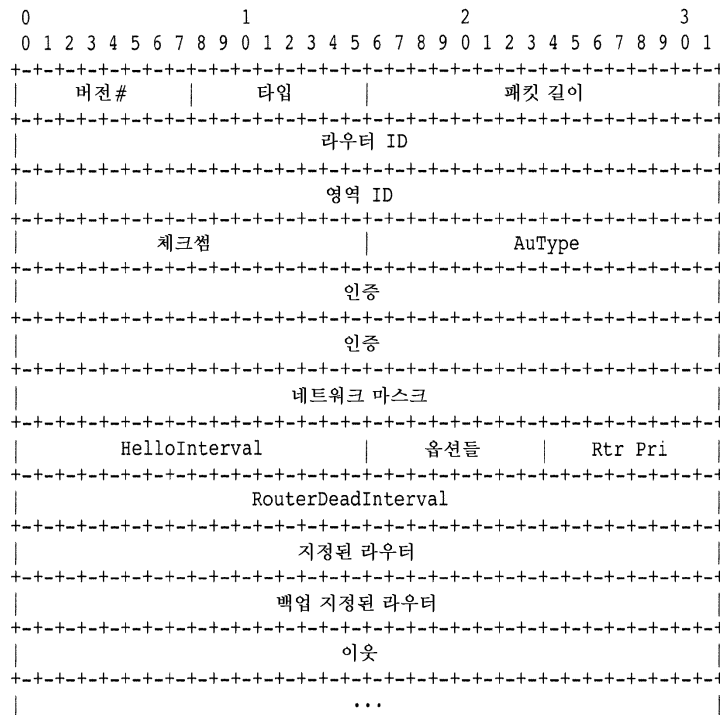
도면1



도면2



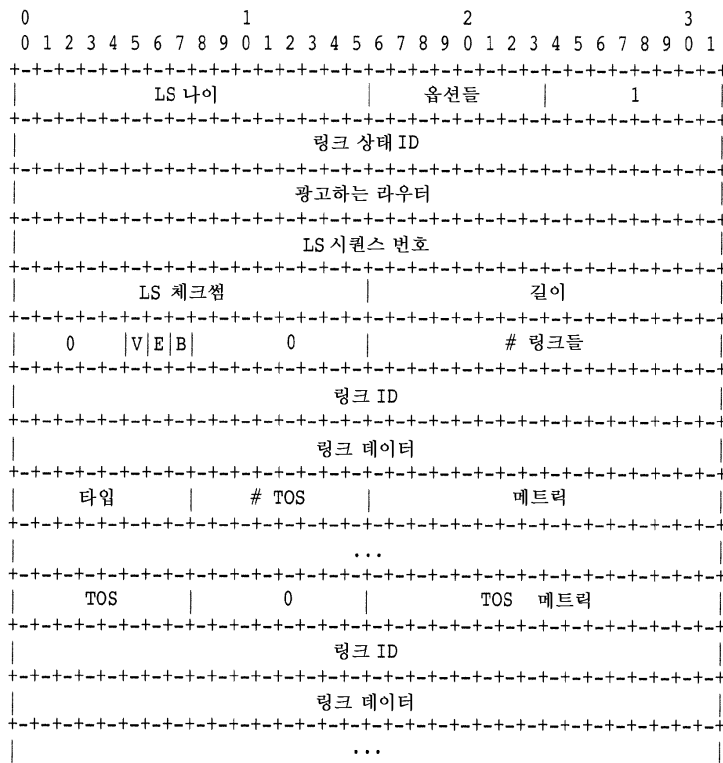
도면3



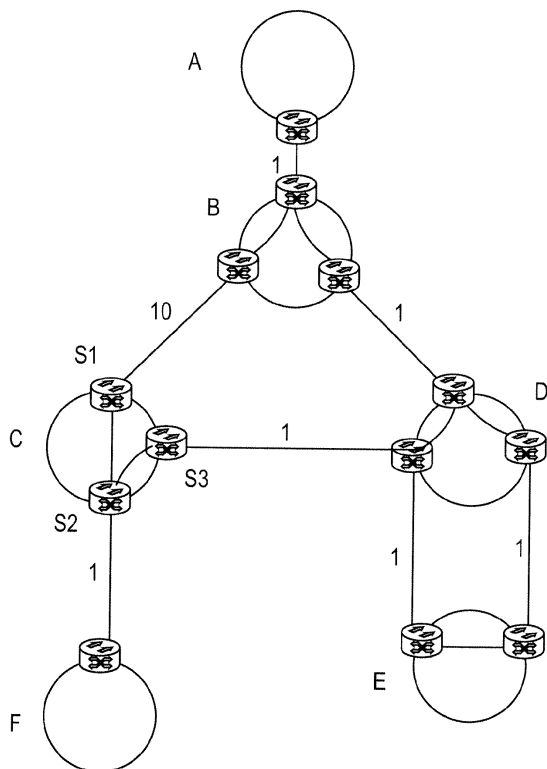
도면4



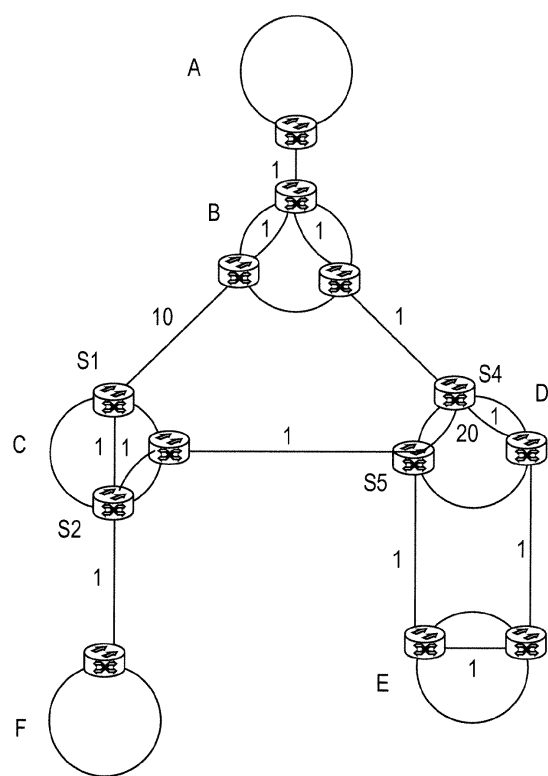
도면5



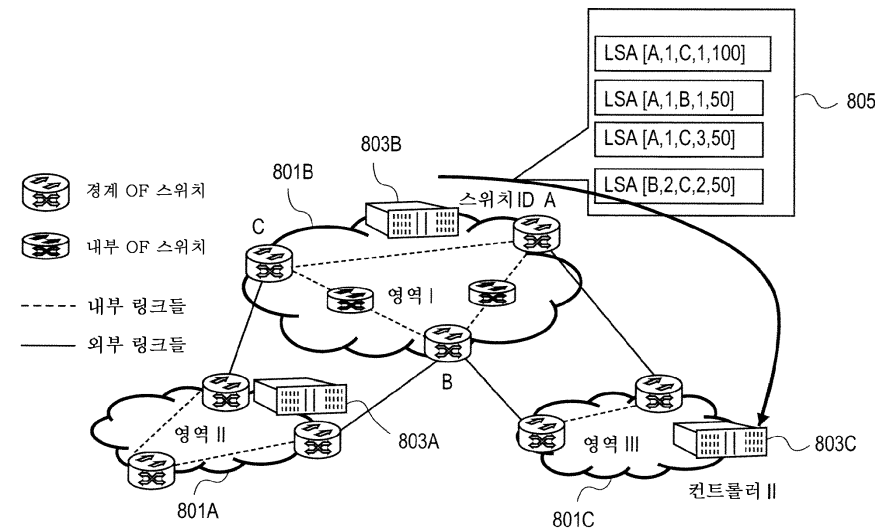
도면6



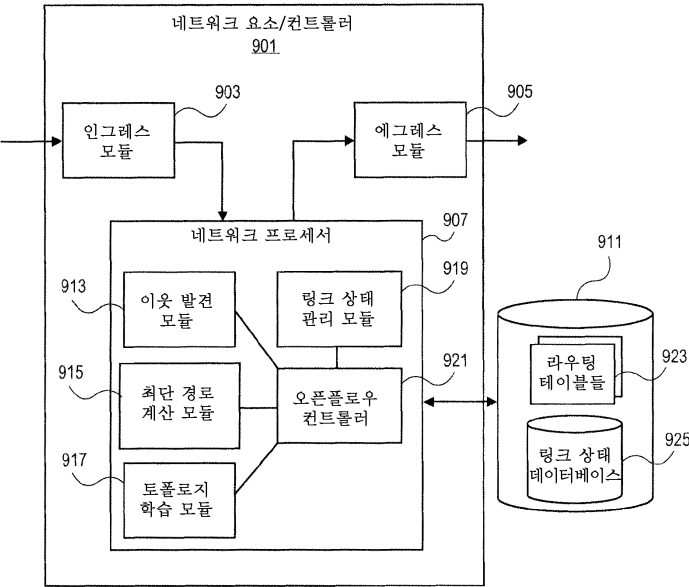
도면7



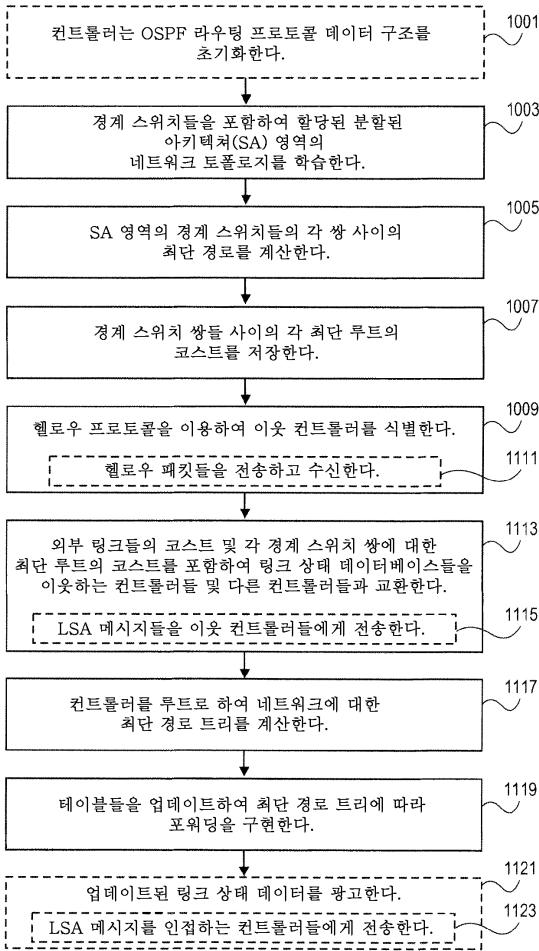
도면8



도면9



도면10



도면11

비트/바이트	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31														
0	LS 나이																LS 타입																													
32	링크 상태 ID																																													
64	광고하는 라우터																																													
96	LS 시퀀스 번호																																													
128	LS 체크섬																길이																													
160	"LS 타입"에 따라 컨텐츠의 세부사항들이 가변될 것이다.																																													
192																																														
-																																														
-																																														
-	...																																													

도면12

비트/바이트	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	LS 나이															0	0	1	1													
32	링크 상태 ID																															
64	셀프 스위치 ID																															
96	LS 스위치 번호																															
128	LS 체크섬															길이																
160	0	N	x	V	E	B											옵션들															
192	타입															0	메트릭															
224	셀프 인터페이스 ID																															
256	이웃 인터페이스 ID																															
288	이웃 스위치 ID																															
320	...																															
-	타입															0	메트릭															
-	인터페이스 ID																															
-	이웃 인터페이스 ID																															
-	이웃 라우터 ID																															
-	...																															

도면13



도면14

