SDN 2.0 실현을 위한 네트워크 가상화 플랫폼 핵심기술 및 서비스 연구

고려대학교

유 혁

2016년 11월 25일

목 차

I 과제 개요

사업 목적 및 필요성

사업 목적

- 국가 차원의 경제·사회적 중요성과 중장기적 파급효과가 큰 전략적 연구분야 지원을 통해 산업 활성화 지원
- 기초연구의 전략성 강화 및 목표 지향적 연구 활성화를 통해 국가 과학기술 경쟁력 제고 및 네트워크 산업에서의 세계 경쟁력 강화

필요성

- 미국, 유럽에서는 기존의 SDN을 발전시킨 SDN 2.0 기술 개발 추진
- SDN 2.0 - 개방화·자동화에 기반 가상 네트워크 인프라 실현이 목표
- 하지만 국내는 오픈플로우, NFV 기술에만 집중

→ 수년 내 세계 최고 기술과의 격차가 커질 수 있음

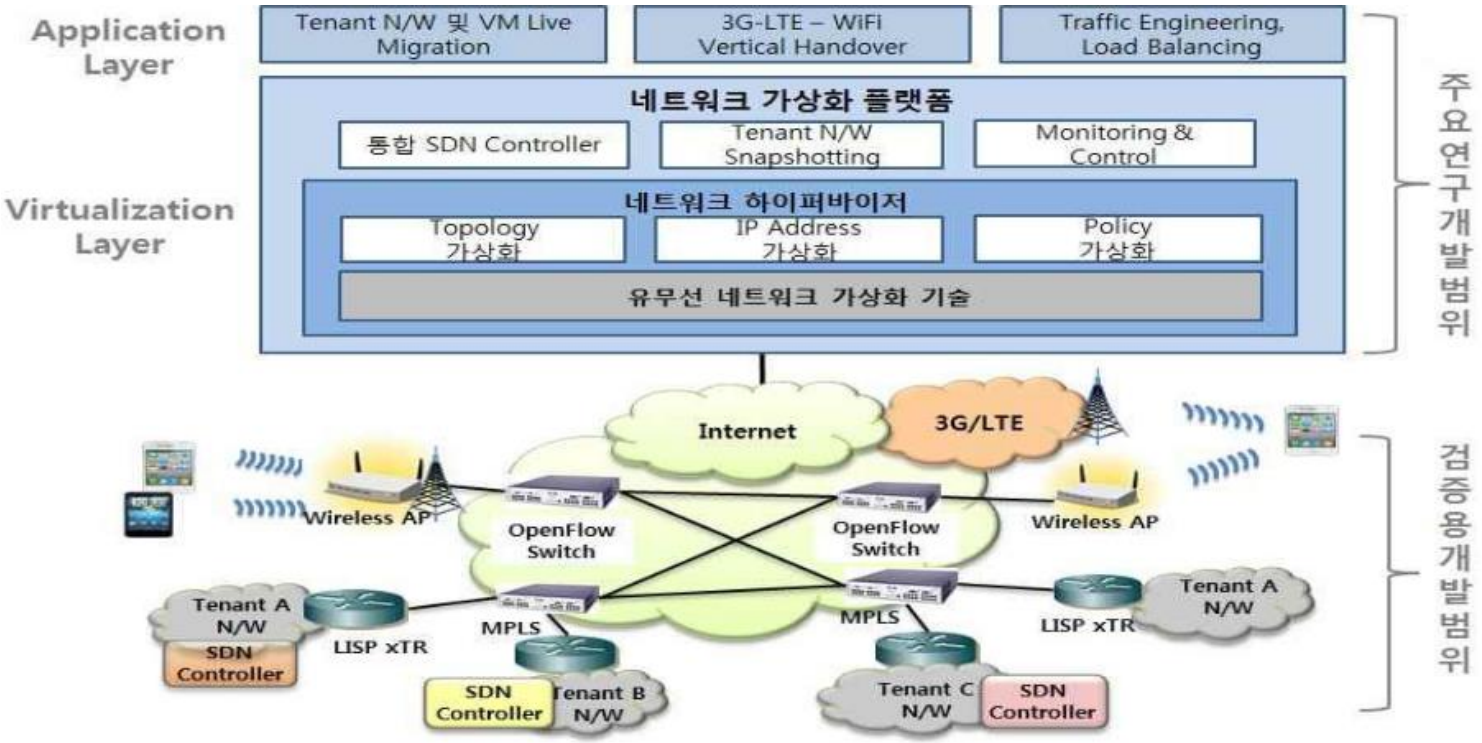
SDN 2.0의 핵심인 네트워크 하이퍼바이저 개발이 시급

최종 연구 목표

연구목표

SDN 네트워크 하이퍼바이저 및 가상화 플랫폼 개발과 캐리어용 인터페이스 확장 및 응용 서비스 기술 개발

- SDN 네트워크 하이퍼바이저 S/W 및 가상화 플랫폼 S/W
- SBI 프로토콜 및 네트워크 가상화 응용 서비스 S/W



연차별 추진계획

SDN 2.0 실현을 위한 네트워크 가상화 플랫폼 핵심기술 및 서비스 연구

연도	개발 내용
1차년도	• 세계 최고 수준의 네트워크 하이퍼바이저 설계 • 캐리어 네트워크 적용을 위한 컨트롤러 SBI 확장 기술 설계 • SDN 가상 네트워크 기반 트래픽 엔지니어링(TE) 알고리즘 설계 • 분산형 컨트롤러를 이용한 MPLS, LISP 프로토콜을 이용한 테스트베드 설계
2차년도	• 네트워크 하이퍼바이저 및 스위치 가상화 구현 • 확장 Southbound Interface(xSBI) 구현 • 트래픽 엔지니어링 구현 (네트워크 하이퍼바이저) • 오픈플로우 스위치 및 LISP xTR 테스트베드 구축
3차년도	• vSDN 컨트롤러 개발 • 트래픽 엔지니어링 구현 (가상 네트워크 컨트롤러) • vSDN 컨트롤러를 포함한 테스트베드 구축
4차년도	• 유무선 통합 테스트베드 구축 및 성능 측정 • 확장 SBI 기능 검증용 응용 서비스 개발 및 검증 • 네트워크 트래픽 및 장애 발생 모니터링 성능지표 검증 및 보완
5차년도	• 실제 네트워크 망인 KREONET에서 테스트망을 구축 및 시스템 안정화 • 유무선 복합 환경에서 기능 검증 및 보완

Ⅱ 2차년도 추진 실적

2차년도 목표 및 내용

네트워크 하이퍼바이저 및 스위치 가상화 구현

- 네트워크 하이퍼바이저 주소 가상화 기능 개발
- 컨테이너 기반 가상 스위치 모델 개발

확장 SBI 구현

- 오픈플로우와 호환 가능한 벤더 독립적 LISP S/W 개발
- MPLS 지원을 위한 데이터평면 추상화 기술 개발

트래픽 엔지니어링

- 가상화된 SDN을 위한 네트워크 모니터링 플랫폼 개발
- 슬라이스 간 장애우회 알고리즘 개발

테스트베드 구축

- 하드웨어 기반 오픈플로우 스위치 실험 환경 구축
- 실험용 LISP 스위치 테스트베드 구축

네트워크 하이퍼바이저 및 스위치 가상화 구현 (1/6)

세부 목표 1-1.

- 네트워크 하이퍼바이저 주소 가상화 기능 개발

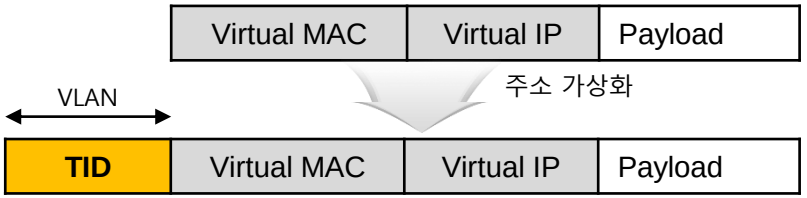
세부 내용

기존 주소 가상화 기법의 한계

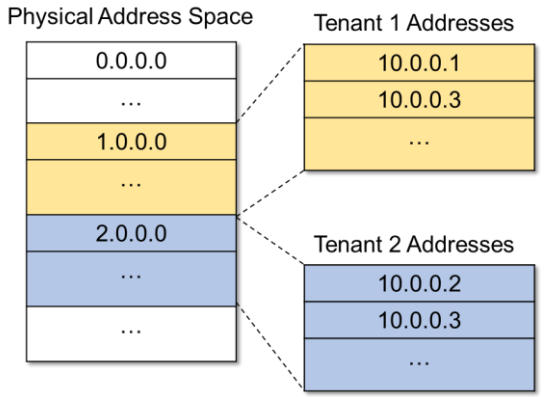
▪ **TID 기반 가상 주소 식별 [FlowN]**
- 중복되는 가상주소들을 Tenant Identifier(TID)로 식별
- TID는 VLAN 터널링을 통해 VLAN ID 필드(12bit)에 기재
- Ingress / egress 스위치에서 터널링 수행

▪ **1:1 주소 맵핑 [OpenVirteX (OVX)]**
- 가상 주소당 하나의 물리 주소를 1:1로 할당
- 물리 IP주소의 상위 8bit를 tenant identifier로 사용, 하위 24bit를 순차적으로 가상주소에 할당
- Ingress / egress 스위치에서 주소 rewriting 수행 (오픈플로우의 set action 이용)

▪ **한계**
- 가상 네트워크 수 제한 (TID 기반: 2^{12}개, 1:1 주소 맵핑: 2^{8}개)
- 한 가상 네트워크 내의 호스트 수 제한
- 각 가상 네트워크의 플로우 룰이 1:1로 물리 네트워크에 설치됨
  → TCAM 소모량 증가
  → Network state 변화 시 컨트롤 메시지 대량 발생
- 터널링 방식: 패킷 당 처리 오버헤드 증가
  → En/decapsulation, Packet fragmentation 등



<VLAN 터널링 기반 주소 가상화>



<OVX의 1:1 주소 맵핑>

네트워크 하이퍼바이저 및 스위치 가상화 구현 (2/6)

세부 목표 1-1.

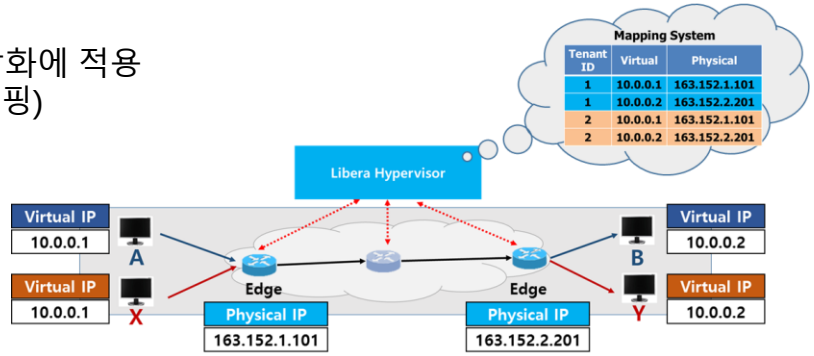
- 네트워크 하이퍼바이저 주소 가상화 기능 개발

세부 내용

Approach 1 – LISP-based one-to-many address mapping

- **동기:** 주소 맵핑 방식의 한계점인 가상 네트워크 수의 제한, 1:1 플로우 룰 설치로 인한 자원 소모를 해결

- **동작 방식**
  - LISP 프로토콜의 '내부망과 단말의 주소 분리 개념'을 주소 가상화에 적용
  - 호스트의 가상 주소를 인접 스위치의 물리 주소로 맵핑 (1:多 맵핑)
  - 인접 스위치에서 터널링 수행
- **장점**
  - 지원가능한 가상 네트워크 수 증가
  - 물리 네트워크 내 플로우 룰 엔트리 감소
  - Host migration 지원 용이 (Edge 스위치의 플로우 룰만 변경)
- **한계**
  - MAC 주소와 IP 각각 맵핑 필요
  - 터널링 오버헤드 존재



1. A에서 B로 패킷 전송

Src Virtual IP	Dst Virtual IP
10.0.0.1	10.0.0.2

2. Ingress Edge에서 Encapsulation

Src Physical IP	Dst Physical IP	Tenant ID	Src Virtual IP	Dst Virtual IP
163.152.1.101	163.152.2.201	1	10.0.0.1	10.0.0.2

3. Egress Edge에서 Decapsulation

1. X에서 Y로 패킷 전송

Src Virtual IP	Dst Virtual IP
10.0.0.1	10.0.0.2

Src Physical IP	Dst Physical IP	Tenant ID	Src Virtual IP	Dst Virtual IP
163.152.1.101	163.152.2.201	2	10.0.0.1	10.0.0.2

네트워크 하이퍼바이저 및 스위치 가상화 구현 (3/6)

세부 목표 1-1.

- 네트워크 하이퍼바이저 주소 가상화 기능 개발

세부 내용

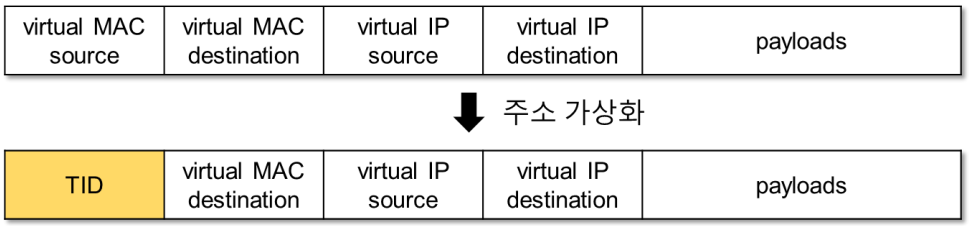
Approach 2 – Mapping-less address virtualization without tunneling

- **동기:** 터널링 없이 가상 네트워크의 확장성을 높이는 방식 고안

- **동작방식**
  - Ingress 스위치에서 패킷의 Source MAC 주소 필드에 TID 기재
    - 기존 Source MAC의 정보는 in_port 또는 source IP에 기재된 정보로 확인 가능
  - Egress 스위치에서 원래의 Source MAC 값 복원

- **장점**
  - 기존 VLAN 대비 월등한 가상 네트워크 수용 확장성 (2^{48}개의 가상 네트워크 지원 가능)
  - 터널링으로 인한 오버헤드 제거
  - MAC 주소 및 IP 주소 가상화 간단히 지원

- **한계**
  - 플로우 룰이 물리 네트워크에 1:1로 설치됨



네트워크 하이퍼바이저 및 스위치 가상화 구현 (4/6)

세부 목표 1-1.

- 네트워크 하이퍼바이저 주소 가상화 기능 개발

세부 내용

Approach 3 – Hop-by-hop forwarding

- **동기:** 연구 2를 기반으로, 자원 소모를 최소화하기 위한 플로우 룰의 1:多 맵핑 기법 연구
- **동작방식**
  - 가상 네트워크의 플로우 룰의 1:多로 맵핑을 위해, 가상 네트워크의 플로우 룰의 매칭 필드 조정, 액션 추가

스위치	Matching fields	추가하는 Action
Ingress	In_port, MAC pair, IP pair	Set {Source MAC=TID, Dest MAC = next hop}
Core	In_port, Destination MAC	Set {Destination MAC = next hop}
Egress	In_port, MAC pair, IP pair	Set {Source MAC=source host, Source Dest=destination host}

  - 단말의 주소가 필요한 지점에서는 TID와 IP pair까지 매칭
    - 코어 스위치에 설치되는 플로우 룰 중, 동일한 port로 들어왔으나, Output port가 다른 플로우 룰
- **장점**
  - 스위치에 패킷 도착 시 가상주소가 아닌 스위치의 물리 주소를 매치 → 스위치에 설치되는 플로우 룰 최소화
  - 코어 스위치의 플로우 룰 개수, 네트워크 하이퍼바이저와 스위치 사이의 컨트롤 채널 트래픽량 감소
- **한계**
  - 가상 네트워크 컨트롤러가 생성하는 플로우 룰을 네트워크 하이퍼바이저가 임의로 병합
  - 현재 가상 네트워크별 요구조건에 따라 플로우 룰의 맵핑 방식을 차등화하는 연구 진행 중

네트워크 하이퍼바이저 및 스위치 가상화 구현 (5/6)

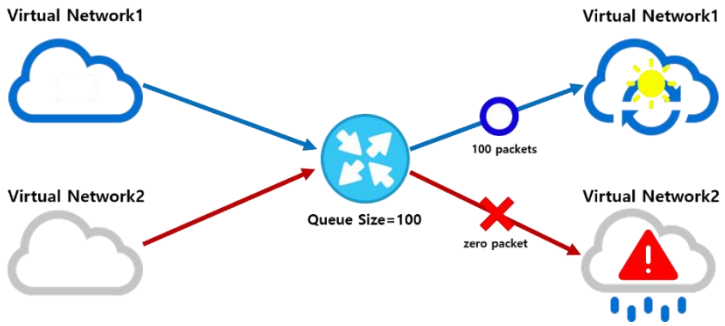
세부 목표 1-2.

- 컨테이너 기반 가상 스위치 모델 개발

세부 내용

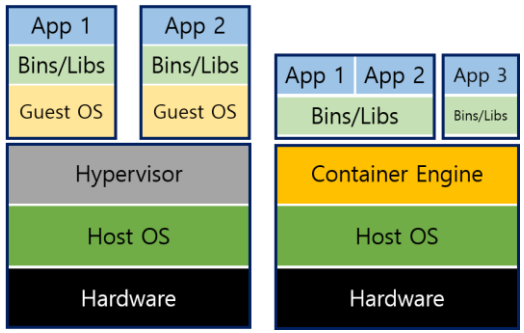
기존 가상 스위치 기법의 한계

- **가상 네트워크 간의 성능 간섭 현상**
  - 특정 가상 네트워크의 물리 자원 독점 사용 문제 존재
  - 가상 네트워크 1의 트래픽 증가 -> 물리 스위치 큐(queue)의 용량이 초과되어 가상 네트워크 2의 트래픽 처리 불가
- **독립된 가상 스위치 구조가 필요**
  - 물리 스위치 자원을 효율적으로 활용하고 각 가상 네트워크에 독립적 트래픽 처리를 제공해야 함
  - SDN 기반 네트워크 가상화 환경에 적합한 구조 필요



컨테이너 가상화 정의 및 필요성

- **컨테이너 가상화**
  - 각 컨테이너는 운영체제의 커널 위에 존재하는 사용자 공간 (userspace)에서 동작.
  - Cgroups을 이용하여 컨테이너 단위의 자원 할당(CPU, 메모리, 블록 I/O, 네트워크 등) 제공하여 자원 고립 제공
  - 기존 가상화 방식에 비해 경량화 (lightweight)
  - 100개 이상의 가상 네트워크를 지원하기 위해서는 경량화 된 가상화 환경이 필요



네트워크 하이퍼바이저 및 스위치 가상화 구현 (6/6)

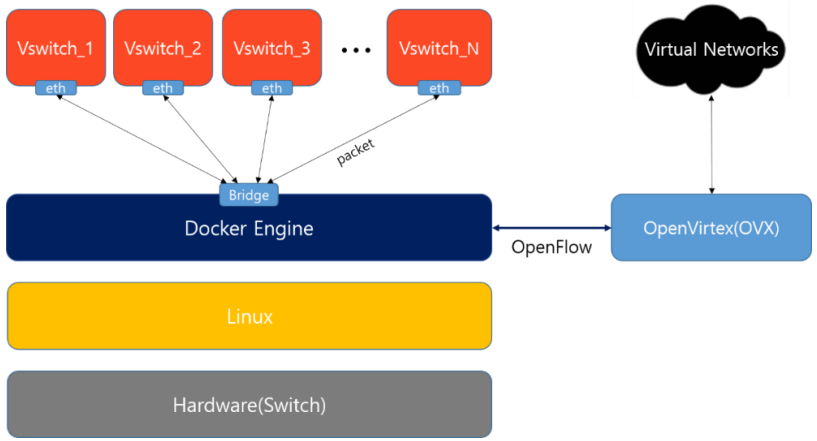
세부 목표 1-2.

- 컨테이너 기반 가상 스위치 모델 개발

세부 내용

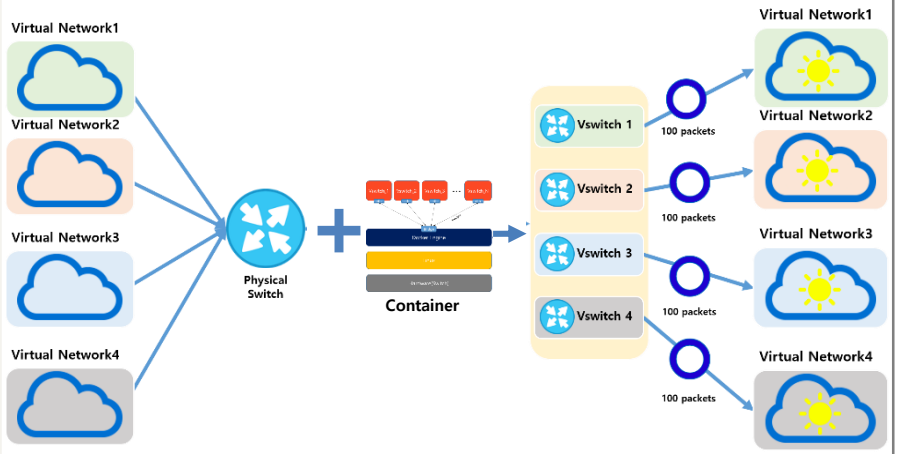
컨테이너 가상화 기반 스위치 모델

- **구조**
  - 각 컨테이너가 가상 스위치의 역할을 수행
  - 네트워크 하이퍼바이저와 오픈플로우 프로토콜로 통신함으로써 가상 네트워크 제어 가능
  - 여러 tenant가 가상 스위치를 각각 독립적으로 제어



가상 스위치 자원 고립

- **Cgroup을 이용한 가상 스위치 단위 자원 고립**
  - 가상 스위치 단위로 CPU, 메모리, 네트워크 자원 할당
  - 다른 가상 스위치에 영향을 받지 않고 안정적인 성능 제공
  - 성능 고립 뿐아니라, 각 가상 네트워크의 요구 사항에 따라 자원을 할당하는 성능 차등화를 제공할 수 있음



2차년도 목표 및 내용

네트워크 하이퍼바이저 및 스위치 가상화 구현

- 네트워크 하이퍼바이저 주소 가상화 기능 개발
- 컨테이너 기반 가상 스위치 모델 개발

확장 SBI 구현

- 오픈플로우와 호환 가능한 벤더 독립적 LISP S/W 개발
- MPLS 지원을 위한 데이터평면 추상화 기술 개발

트래픽 엔지니어링

- 가상화된 SDN을 위한 네트워크 모니터링 플랫폼 개발
- 슬라이스 간 장애우회 알고리즘 개발

테스트베드 구축

- 하드웨어 기반 오픈플로우 스위치 실험 환경 구축
- 실험용 LISP 스위치 테스트베드 구축

확장 SBI 구현 (1/4)

세부 목표 2-1.

- 오픈플로우와 호환 가능한 벤더 독립적 LISP S/W 개발

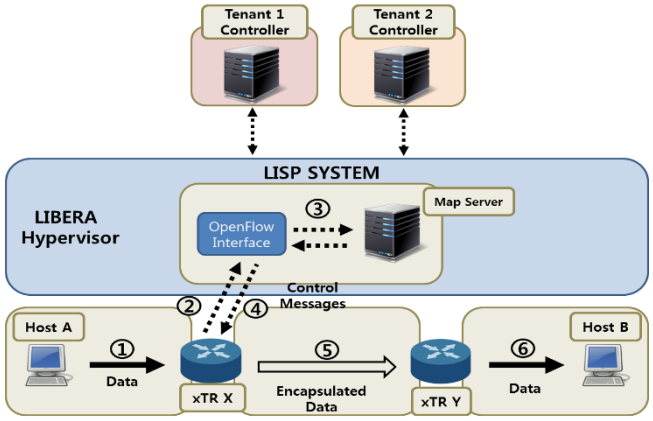
세부 내용

기존 네트워크 하이퍼바이저에서의 LISP 지원

- **LISP (Locator Identifier Separation Protocol)**
  - 식별자(identifier)와 위치자(locator)를 분리하는 방식의 새로운 인터넷 아키텍쳐
  - 라우팅과 어드레싱의 확장성 보장
- **LISP 지원 기능의 필요성**
  - 라우팅테이블 확장성 저하: 노드가 증가함에 따라 라우팅 테이블 크기가 기하급수적으로 증가
  - Live migration의 경우, IP주소가 변환되면 네트워크 경로를 재탐색 해야 하므로 네트워크 지연시간 증가
- **벤더 독립적 LISP 지원의 필요성**
  - 기존 LISP 구조는 특정 벤더 (Cisco)가 제공하는 하드웨어와 소프트웨어를 사용해야 하므로 제어평면과 데이터 평면의 분리가 불가능하고 벤더에 종속적
  - SDN 환경에서의 LISP 지원을 위해 오픈플로우 프로토콜과 호환 가능한 벤더 독립적 LISP S/W가 필요함

오픈플로우 기반 LISP 지원 기능 개발

- **제어 평면 (리베라 하이퍼바이저)**
  - 가상 네트워크를 지원하는 매핑서버 구축: RLOC 주소 할당 및 매핑정보 저장 및 관리를 위한 매핑서버 구현
  - LISP encap/decap을 위한 오픈플로우 Action 구현
- **데이터 평면 (OpenvSwitch)**
  - LISP 패킷 형태로 encap/decap 기능 추가



확장 SBI 구현 (2/4)

세부 목표 2-1.

- 오픈플로우와 호환 가능한 벤더 독립적 LISP S/W 개발

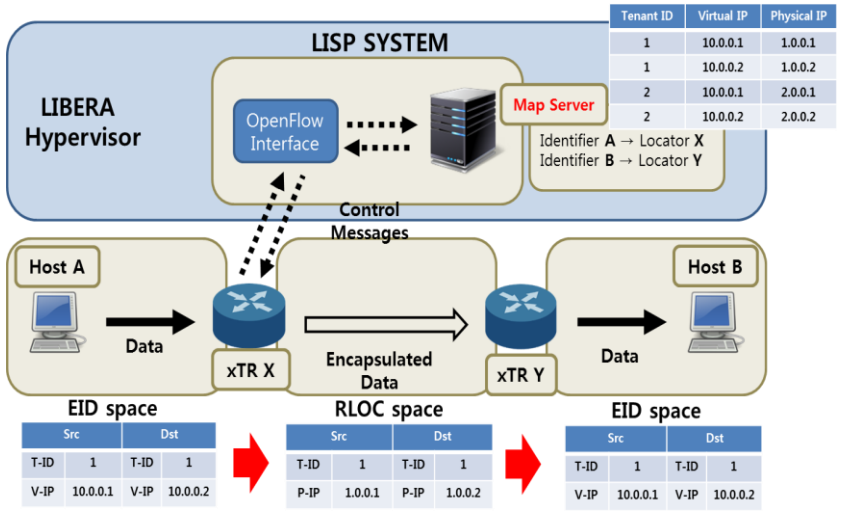
세부 내용

제어 평면 (리베라 하이퍼바이저)의 LISP 지원기능 구현

- **네트워크 하이퍼바이저 내 매핑서버 (MS) 구현**
  - 가상 네트워크 환경에서 서로 다른 tenant가 동일한 가상 주소를 가질 때, 이를 tenant ID를 통해 구별하도록 매핑서버 설계 및 구현
  - PacketIn의 정보를 분석, Map-register/request 기능을 함으로써 MS에 새로운 매핑정보 저장/탐색이 가능

- **오픈플로우 테이블 동기화 기능 개발**
  - Live Migration의 경우, 노드의 가상 주소는 동일하게 유지하지만 매핑되는 스위치의 주소가 변하기 때문에 매핑 정보 업데이트가 필요함
  - 네트워크 하이퍼바이저와 네트워크 스위치의 매핑 정보를 업데이트 하도록 오픈플로우 기능 확장

리베라 하이퍼바이저에서의 LISP 동작 흐름도



확장 SBI 구현 (3/4)

세부 목표 2-1.

- 오픈플로우와 호환 가능한 벤더 독립적 LISP S/W 개발

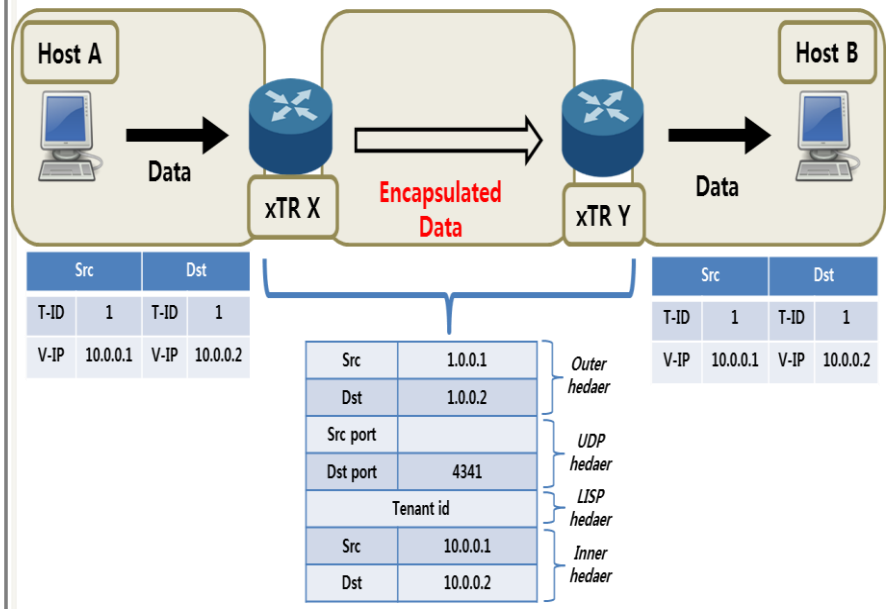
세부 내용

데이터평면 (OpenvSwitch)의 LISP 지원기능 구현

- **Encap/Decap action 추가**
  - Non-LISP packet을 LISP의 형태로 라우팅 하도록 설계
  - Outer IP 헤더와, LISP UDP 헤더의 encapsulation/de-capsulation 기능 구현
  - LISP Header 하위 32비트에 Tenant ID 값을 저장하여 tenant 식별

- **추후 진행 일정**
  - OpenvSwitch Flow table에 tenant ID 필드 추가(~2016.12 월 완료 예정)
  - LISP 표준에 명시된 기능 구현
  - - Migration 이 진행되는 과정에서 발생할 수 있는 packet loss를 줄이기 위한 packet forwarding 기술 개발 계획 (ex. 스위치상의 Buffer 를 이용하여 저장-전달)

OpenvSwitch 에서의 LISP 패킷 처리 과정



확장 SBI 구현 (4/4)

세부 목표 2-2.

- MPLS 지원을 위한 데이터평면 추상화 기술 개발

세부 내용

기존 SDN 1.0의 한계

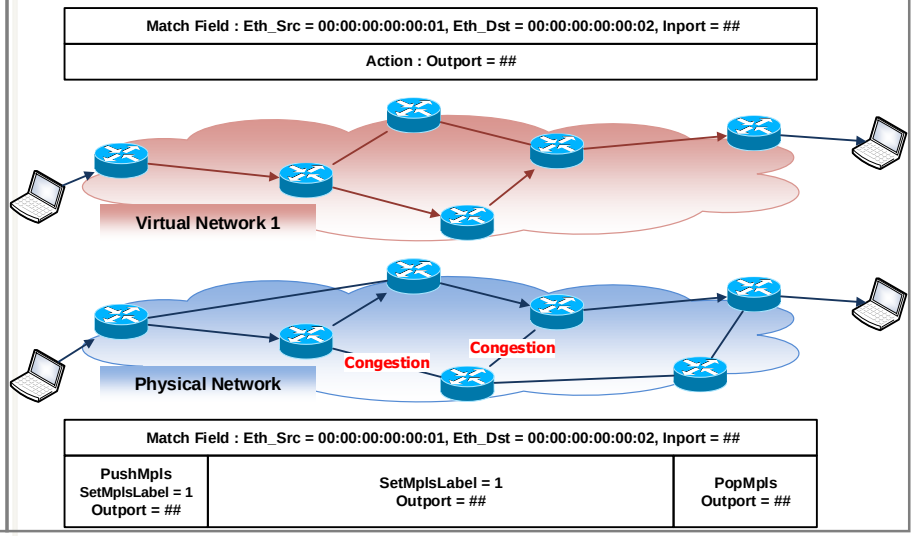
- **Multi-protocol label switch (MPLS)**
  - 패킷의 짧은 라벨을 식별하여 경로를 정하는 라우팅 기법
  - 네트워크 자원을 효율적으로 활용, 빠른 트래픽 엔지니어링 제공
- **MPLS와 같은 Legacy 네트워크 프로토콜 미지원**
  - 기존 네트워크와의 호환성 결여
  - 캐리어급 네트워크 지원이 어려움

MPLS 프로토콜을 지원하는 네트워크 하이퍼바이저 개발

- **제어 평면 (리베라 하이퍼바이저)에서의 MPLS 지원**
  - Openflowj 라이브러리를 활용한 버전 업그레이드
  - 가상 네트워크 Flow 단위 MPLS 라벨 할당
    - ✓ 가상 네트워크의 호스트 MAC 주소로 Flow 구분
    - ✓ Flow ID는 Source MAC, Destination MAC 주소를 기준으로 Index를 증가 시켜 생성
- **데이터평면 (OpenvSwitch)에서의 MPLS 지원**
  - 오픈플로우 v.1.3 의 확장된 matching field 활용

Usecase: MPLS 프로토콜을 이용한 트래픽 엔지니어링

- 가상 네트워크의 컨트롤러의 명령을 네트워크 하이퍼바이저에서 MPLS 프로토콜 명령으로 변환
- 물리 네트워크에서는 MPLS 라벨링을 통해 빠른 TE 가능



2차년도 목표 및 내용

네트워크 하이퍼바이저 및 스위치 가상화 구현

- 네트워크 하이퍼바이저 주소 가상화 기능 개발
- 컨테이너 기반 가상 스위치 모델 개발

확장 SBI 구현

- 오픈플로우와 호환 가능한 벤더 독립적 LISP S/W 개발
- MPLS 지원을 위한 데이터평면 추상화 기술 개발

트래픽 엔지니어링

- 가상화된 SDN을 위한 네트워크 모니터링 플랫폼 개발
- 슬라이스 간 장애우회 알고리즘 개발

테스트베드 구축

- 하드웨어 기반 오픈플로우 스위치 실험 환경 구축
- 실험용 LISP 스위치 테스트베드 구축

트래픽 엔지니어링 구현 (1/2)

세부 목표 3-1.

- 가상화된 SDN을 위한 네트워크 모니터링 플랫폼 개발

세부 내용

기존 SDN 기반 모니터링의 한계

- 기존 SDN에서의 네트워크 모니터링
  - 컨트롤러 부하 경감을 위한 모니터링 최적화에 집중 (DevoFlow, FlowSense, OpenSketch 등)
- 가상화 상황에서의 모니터링은 연구된 바 없음
- OpenVirteX: 정적 주기로 전체 물리 네트워크 모니터링, 가상 네트워크 요청 시 보유한 모니터 정보만 전달
  - 정확성 문제 발생: 플로우 테이블 consistency 체크를 하는 컨트롤러의 경우(예- ONOS) 오작동 초래

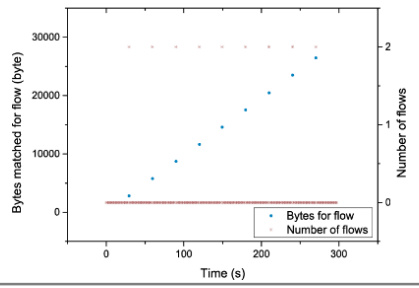
Flo-v: Low overhead network monitoring in network hypervisor

- Pre-monitoring (from OVX): 모니터링 Delay를 줄이기 위해 가상 네트워크 요청 전에 모니터링 수행
- Selective monitoring: 가상 네트워크가 요청하는 리소스만 반복적으로 모니터
- Adaptive monitoring: 사전 모니터링의 주기를 각 가상 네트워크의 요청을 반영하여 조절

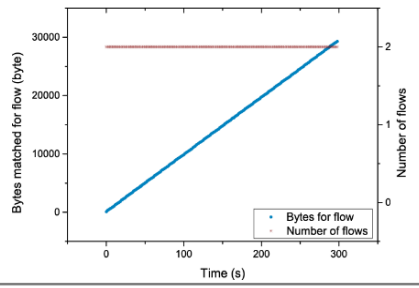
평가

- 300번의 연속적인 Ping 트래픽이 존재하는 가상 네트워크 모니터링 결과
  - 기존 OVX: 고정된 모니터링 주기인 30초마다 트래픽을 관찰하기 때문에 30초 사이의 트래픽은 가상 네트워크에서 감지하지 못함
  - 제안한 기법은 정상적인 트래픽의 관찰 가능
- 오버헤드: 기존 OVX에 비해 0.7ms의 지연 증가
  - 지연 감소를 위해, 모니터링 주기 변경에 따른 timer resetting 정책 최적화 예정

<기존 OVX>



<Flo-v>



트래픽 엔지니어링 구현 (2/2)

세부 목표 3-2.

- 슬라이스 간 장애우회 알고리즘 개발

세부 내용

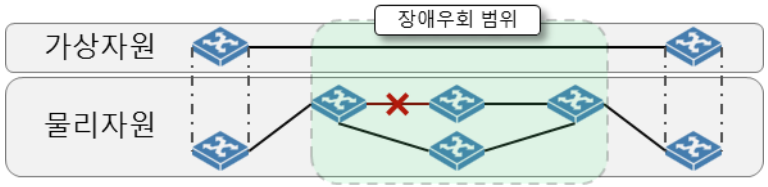
기존 장애우회 기술 분석

▪ 슬라이스 간 장애우회

- 네트워크 하이퍼바이저는 물리 자원과 가상 자원(슬라이스)의 맵핑을 관리함
- 결함이 발생한 물리자원의 맵핑을 조정하여, 가상 자원의 결함을 방지하는 기술

▪ 가상화된 SDN에서의 실패 대응 및 한계

- 관련 연구 및 논문 없음
- OpenVirteX의 Resilience 기능: Link reconstruction 제공
- 한계점
  - 가상 링크에 맵핑된 내부 자원에서만 장애우회



- 네트워크 동적 반영 불가 (Utilization 한계)
- 대체경로 정적 계산 및 개수 제한 (확장성 없음)

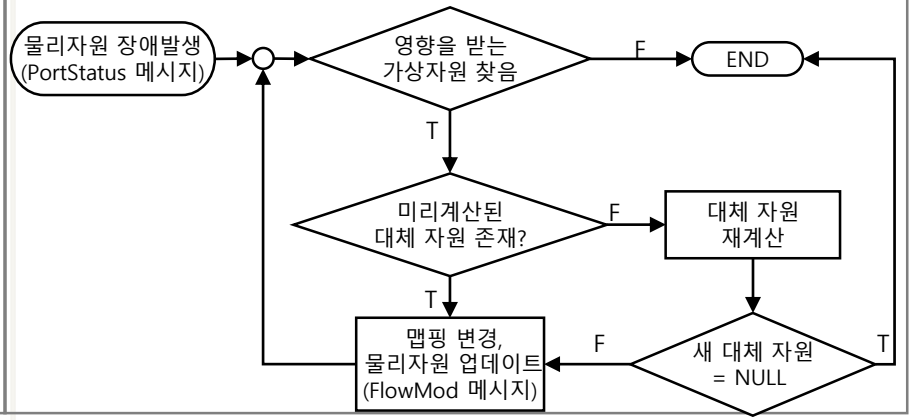
Seamless failover for Slices

▪ 정의

- 초기 자원 맵핑의 경계를 넘어, 포트-링크-스위치 수준의 모든 물리 실패에 대응하는 장애우회 기법

▪ 동작 방식

- 슬라이스 재맵핑의 범주 확장 (가상 자원에 맵핑된 내부 물리 자원 → 전체 네트워크)
- 라우팅 알고리즘 개선 (Network state 반영)



2차년도 목표 및 내용

네트워크 하이퍼바이저 및 스위치 가상화 구현

- 네트워크 하이퍼바이저 주소 가상화 기능 개발
- 컨테이너 기반 가상 스위치 모델 개발

확장 SBI 구현

- 오픈플로우와 호환 가능한 벤더 독립적 LISP S/W 개발
- MPLS 지원을 위한 데이터평면 추상화 기술 개발

트래픽 엔지니어링

- 가상화된 SDN을 위한 네트워크 모니터링 플랫폼 개발
- 슬라이스 간 장애우회 알고리즘 개발

테스트베드 구축

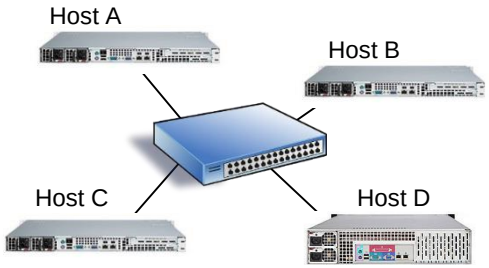
- 하드웨어 기반 오픈플로우 스위치 실험 환경 구축
- 실험용 LISP 스위치 테스트베드 구축

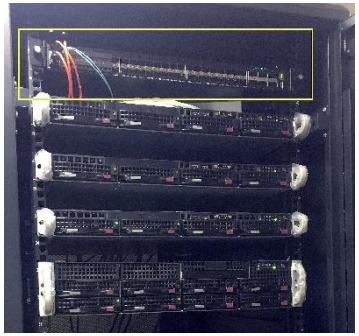
스위치 기반 테스트베드 구현

세부 목표 4-1.

- 하드웨어 기반 오픈플로우 실험 환경 구축

- **오픈플로우 v1.3 지원 델 10G 스위치 S4048 스위치 구입**
- **Rack 스케일의 노드 및 ToR 스위치로 구성된 실험환경 구축**
- **소프트웨어 기반 스위치 및 하드웨어 기반 오픈플로우 스위치를 이용한 다양한 테스트 가능**

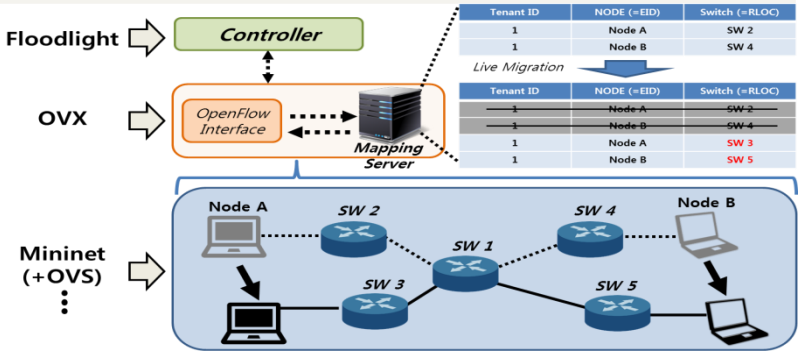




세부 목표 4-2.

- 실험용 LISP 스위치 테스트베드 구축

- **2차년도 구현 및 개발한 내용을 바탕으로, 실제 노드 간 LISP 기반 통신이 제대로 이루어 지는지에 대한 테스트베드 설계**
- **새로운 node에 대한 Mapping 정보 저장 및 기존 node들에 대한 Mapping 정보 업데이트 동작 테스트**
- **Migration이 이루어 지는 상황에서 node간 통신 연결 테스트**



Ⅲ 향후 계획

향후 계획

- **연구 결과물의 오픈 소스화**
  - 네트워크 하이퍼바이저 주소 가상화의 결과 소스 코드를 OpenVirtex 오픈소스에 Contribution 예정
  - 지속적인 코드 Contribution을 통해 메인 스트림 적용이 목표

- **가상 네트워크를 고려한 트래픽 엔지니어링 기술 연구**
  - 기 개발된 프로토타입 최적화 및 고도화
  - 확장된 MPLS SBI를 활용한 MPLS-TE 기술의 적용 (Load balancing, Fast-failover 등)
  - 가상 네트워크별 서비스 요구조건 보장을 위한 QoS 관련 연구

- **스위치 가상화 기법 확장을 통한 고속 패킷 처리 기법 개발**
  - 기존 컨테이너 가상화 환경에서 OpenvSwitch를 사용한 패킷 (64B) 포워딩의 경우 line rate (10Gbps) 를 만족하지 못함
  - 패킷 처리를 위한 메모리 할당 및 패킷 정보 관리를 최적화한 시스템 개발 중

감사합니다.

– Q&A –