JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 19, No. 8, pp. 1545-1554, Aug. 2018

네트워크 가상화 환경에서 끊김 없는 핸드오버를 위한 일반 AP 활용

이 형 봉[1] · 권 기 현[2*]
[1]강릉원주대학교 컴퓨터공학과
[2]강원대학교 전자정보통신공학부

Utilization of Legacy APs for Seamless Handover in a SDN Environment

Hyung-Bong Lee[1] · Ki-Hyeon Kwon[2*]

[1]Department of Computer Science & Engineering, Gangneung-Wonju National University, Wonju 25457, Korea

[2]Department of Electronics, Information & Communication Engineering, Kangwon National University, Samcheok 25913, Korea

[요 약]

무선 단말의 이동성을 지원하기 위해서는 하나의 AP (Access Point) 처리 영역에 적어도 2개 이상의 AP를 중복배치하여 통신영역을 이어야 한다. 이러한 무선랜 환경에서 무선 네트워크의 효과적인 활용과 사용자에게 지원되는 서비스의 극대화 관점에서 끊김 없는 핸드오버는 매우 중요한 논점 중의 하나이다. 한편, 최근 급속도로 부상하고 있는 네트워크 가상화는 네트워크 관리의 유연성, 세밀한 제어, 편의성 등을 혁명적으로 제고하고 있다. 네트워크 가상화는 원래 유선랜 위주의 데이터센터 내에서 스위치들 간의 통신 홉을 줄이거나 우회하는 플로우 라우팅 테이블을 실시간 제어함으로써 사용자 지연시간을 단축시키고 네트워크 견고성을 높인다. 이 연구에서는 유선랜 위주의 네트워크 가상화 플랫폼 OpenFlow를 일반 AP를 사용하는 밀집된 무선 네트워크 환경에 적용하여 디지털 콘텐츠의 스트리밍 서비스를 위한 끊김 없는 핸드오버를 실현하고 평가한다.

[Abstract]

In order to support the mobility of the wireless devices, at least two APs (Access Points) must be arranged in a single AP area to maintain communication area. In the WLAN (Wireless LAN) environment, seamless handover is one of the most important issues in terms of effective utilization of wireless networks and maximization of services for users. On the other hand, SDN (Software-Defined Networking), which is emerging rapidly in recent years, is revolutionizing network management in terms of flexibility, fine control, and convenience. SDN originally reduces latency time or increases network robustness by real-time flow table control reducing or bypassing paths between switches in LAN-based data centers. In this study, we apply OpenFlow, a SDN platform focused on wired LAN, to a dense WLAN environment using legacy APs to implement and evaluate seamless handover for streaming services of digital contents.

색인어 : 네트워크 가상화, 오픈 플로우, 핸드오버, 무선랜, 일반 AP

Key word : SDN, OpenFlow, Handover, WLAN, Legacy AP

http://dx.doi.org/10.9728/dcs.2018.19.8.1545

Received 28 July 2018; **Revised** 20 August 2018
Accepted 28 August 2018

***Corresponding Author; Ki-Hyeon Kwon**

Tel: +82-33-570-6400
E-mail: kweon@kangwon.ac.kr

Ⅰ. 서 론

무선 네트워크에서는 무선 매체 자체의 애매한 신뢰성으로 인해 상황 변화가 심하므로 유선 네트워크보다 월씬 더 능동적이고 즉각적인 대응이 필요하다. 여기서 능동적이면서 즉각적 대응이라 함은 네트워크 트래픽을 실시간 제어할 수 있어야 한다는 의미이다. 특히, 무선 네트워크 자원 요소들이 적절하게 위치하지 않는다면 AP 신호들의 간섭으로 호스트들의 이동성을 크게 저하시킬 수 있다. 무선 단말 사용자들은 대량의 디지털 콘텐츠에 접근하기 위해 더 높은 속도의 Wi-Fi를 원하고, 이를 위해서는 전파 신호 범위를 좁혀야 하는데, 이로 인해 작아진 셀은 신호간섭을 증가시킨다. 즉, 일반 WLAN 사용자는 제한된 통신 영역에서 다수의 Wi-Fi 신호와 높은 신호강도를 만나지만, 정작 데이터 전송 속도에는 만족하지 못한다. 이는 과도하게 많은 AP들의 신호 간섭으로 데이터 전송률이 낮아지기 때문이다. 이를테면 특정 공공 지역에 다수의 ISP (Internet Service Provider) 업체가 각각의 AP를 독자적으로 설치하면 신호 충돌로 인해 전송속도가 저하되고, 또 다시 이를 극복하기 위한 추가 인프라를 구축하는 악순환이 반복된다. 이런 환경에서는 벤더 간 무선 네트워크가 달라 상호 운영이 거의 불가능하고, 고유의 네트워크 프로토콜과 도구들이 복잡하기 때문에 관리자는 절대적으로 벤더에 의지할 수밖에 없다. SDN은 이와 같은 불합리를 해결할 수 있도록 네트워크를 가상화시켜 네트워크 관리의 유연성과 편리성을 제공하고, 특히 무선 네트워크 환경에서는 신호 측면과 데이터 전송 측면을 분리하여 부하 분산을 통해 균형을 잡을 수 있는 수단을 제공할 수 있다. 이 연구에서는 보편적인 AP들로 구성된 무선랜 환경에서 SDN 플랫폼인 OpenFlow를 이용하여 특정 AP에 집중된 트래픽을 분산시키기 위해 필요한 가장 기본 기능 중의 하나인 끊김 없는 핸드오버를 구현하고 평가한다. 이를 위해 Ⅱ 장에서 네트워크 가상화 관련 연구를 살펴보고, Ⅲ 장에서 일반 AP를 사용하여 OpenFlow 기반 무선랜 끊김 없는 핸드오버를 구현하며, Ⅳ 장에서 구현 결과를 평가한다. 그리고 마지막 Ⅴ장의 결론으로 이 논문을 맺는다.

Ⅱ. 네트워크 가상화 관련 연구

2-1 SDN (Software-Defined Networking)

SDN은 원래 PSTN (Public Switched Telephone Network)에서 프로비저닝 (네트워크 자원의 할당 및 배치)과 관리를 간소화시키기 위해 제어 평면과 데이터 평면을 분리하자는 취지에서 출발했고, 2004년경 IETF (Internet Engineering Task Force, 인터넷 표준화기구)에 의해 이 개념을 인터넷 네트워크 표준 도입을 시도했으나 위험 부담과 벤더들의 상이한 이해관계로 실패했고 [1], 이 후 스탠포드 대학 등 학계를 중심으로 연구가 시작되었고 현재는 유수의 네트워크 기업들이 지원하는 공개 프로젝트 형태로 진행되고 있다.

2-2 OpenFlow

OpenFlow는 SDN 표준 개방 플랫폼 중의 하나로 스탠포드 대학의 Ethane 프로젝트 [2]에 의해 2008년 1차 완성되었다. OpenFlow는 각 플로우 구성요소를 연관지어 실행하는 플로우 테이블 (flow table), OpenFlow 콘트롤러로부터의 명령 패킷을 보내고 받는 보안 채널 (secure channel), 그리고 OpenFlow 콘트롤러와 OpenFlow 가상 스위치 간의 통신을 지원하는 OpenFlow 프로토콜로로 구성된다. OpenFlow 가상 스위치는 OpenFlow 프로토콜을 이해하는 스위치이며, OpenFlow 콘트롤러와의 연결을 형성・유지하고, OpenFlow 콘트롤러의 제어를 받는다. Open Flow 가상 스위치는 다수의 네트워크 인터페이스로 구성되며, 보안 채널은 OpenFlow 콘트롤러와의 통신을 위한 TCP 기반의 보안 채널을 형성한다. 플로우 테이블은 VLAN (Virtual LAN), 이더넷, IP, TCP, 포트 등에 대한 관련 정보를 가지고 있으며, 이러한 정보들에 의해 플로우 동작이 결정되고, OpenFlow 프로토콜을 통해 정보들이 OpenFlow 콘트롤러로 전달된다 [2]. OpenFlow의 전체적인 구성도를 그림 1에 보였다.

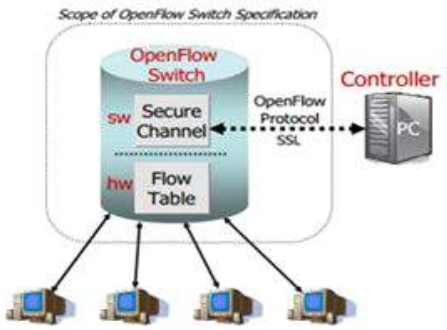

그림 1. OPenFlow의 전체 구성도
Fig. 1. Overall Block Diagram of the OpenFlow

이더넷 환경을 염두에 두고 시작된 OpenFlow를 무선랜에 적용하려는 시도는 주로 무선 노드 간 라우팅 정책 수립에 SDN을 적용하는 연구들이다. 논문 [3]-[5]는 주요 거점에 무선 중계 노드를 두어 멀티 홉 무선 통신으로 넓은 영역을 커버하는 무선 백본 즉, 무선 메시 네트워크 구현에 SDN을 활용하였다. 이와 성격이 다른 기업이나 학교, 병원 등 AP를 중심으로 밀집된 무선 통신 인프라에서의 SDN 적용 방안에 대한 연구는 최근에 시작되고 있다. 논문 [6]은 기존의 AP에는 중앙 콘트롤러와의 제어 통신 기능이 존재하지 않으므로 그림 2와 같이 인증, 어카운팅, 정책 설정, 채널 설정, 단말기 이동성 지원 등을 위해 중앙의 OpenFlow 콘트롤러와 통신할 수 있는 가벼운 가상 AP인 LVAP (Light Virtual Access Point) 구현 개념을 Odin으로 명명하여 제안하였다. 여기서 Odin은 SDN을 이용해 물리적 AP들

을 가상화시켜 주변의 무선 단말기에게는 오직 하나의 AP만을 보여주자는 것으로 이를 위해 특별히 목적에 부합하는 물리적 AP를 제조하여 사용하였다.

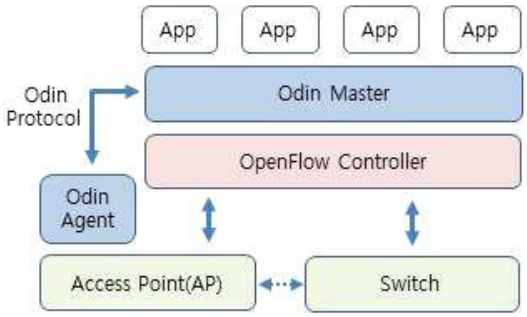

그림 2. LVAP (Odin)의 개념
Fig. 2. Concept of the LVAP (Odin)

2-3 무선랜 핸드오버

CAPWAP (Control And Provisioning of Wireless Access Points) [7]은 그림 3과 같이 무선랜 AP 그룹을 중앙 콘트롤러에서 관리할 수 있는 표준 네트워크 프로토콜로서 네트워크 가상화에 중점을 두는 본격적인 SDN과는 약간 거리가 있다 [8]. 전통적인 WLAN 환경은 AS (Authentication Server), AP (Access Point), MS (Mobile Station) 등 크게 3 개의 컴포넌트로 구성되고, AP는 물리계층과 MAC 계층으로 구성된다. WTP (Wireless Termination Points)는 기존 Fat AP (자체 기능이 풍부한 AP)의 MAC 계층 기능 중 일부분과 물리계층의 기능을 Thin AP (기본 기능만 갖춘 AP)로 구현한 일종의 무선 안테나로 IEEE 802.11 물리 계층 기능을 담당하는 장치인데, CAPWAP의 AC (Access Controller)는 기존 Fat AP의 MAC 계층 기능을 중앙에서 통합 관리하기 위해 엔터프라이즈 무선랜 환경에서 도입한 물리적 구성 요소이다. 즉, CAPWAP는 엔터프라이즈 무선랜을 구성하고 있는 모든 AP의 MAC 기능을 중앙의 AC를 통해 직접 관리하도록 하는데, 이 때 AC는 IEEE 802.11의 인증 및 접근제어 부분을 담당한다. CAPWAP 무선랜 환경에서 AC를 통하여 티켓을 생성하고 분배하여 핸드오버를 지원하기 위해 WTP와 AC 사이에 사용되는 CAPWAP 프로토콜은 단지 티켓을 전송하는 역할만 담당하며, 호스트 (MS: Mobile Station)가 최초 인증과정을 거친 후 새로운 WTP로 핸드오버할 때 802.1x EAP-TLS (Extensible Authentication Protocol - Transport Layer Security, 인증 프레임워크 중 하나) 인증 과정 없이 곧바로 접속하여 서비스를 계속 이용할 수 있도록 함으로써 고속 핸드오버를 지원한다. CAPWAP에서 핸드오버 과정은 그림 4와 같다. 이 그림의 Ⓑ는, WTP1로부터 무선 네트워크 서비스를 제공받던 호스트가 새로운 WTP로 로밍하기 위해 Probe Request/Response 과정을 통해 이용 가능한 WTP2를 발견한 후 WTP2와 사용할 세션키를 새롭게 도출하는 과정인데, 이때 CAPWAP는 802.1x EAP-TLS 인증과정 없이 AC가 PMK (Pairwise Master Key)를 도출하여 AC와 호스트사이의 4-Way Handshake 과정을 바로 진행한다. 즉, 호스트는 접속하고 있던 WTP1과의 접속을 Disassociation 과정을 통해 해지한 후 최초 인증 시 도출했던 PMK0를 재사용하여 WTP2와의 Association 과정을 통해 4-Way Handshake 과정을 바로 실행하고 이를 통해 새로운 세션 키 PTK1을 도출한다. 시스코는 CAPWAP에 기반한 무선랜 핸드오버 가능 제품을 출시하고 있는데 독자적인 고유 AP를 개발하여 사용한다 [9].

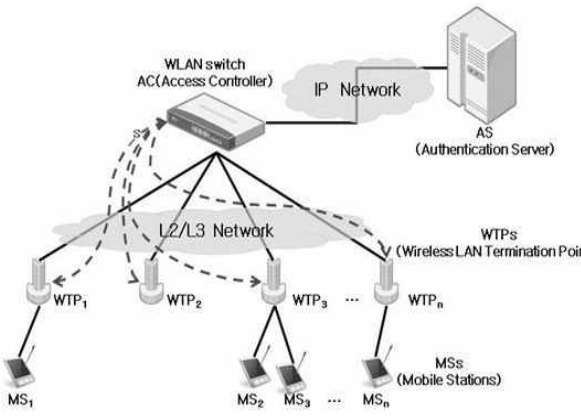

그림 3. CA{WA{의 전체 구성도
Fig. 3. Overall Block Diagram of the CAPWAP

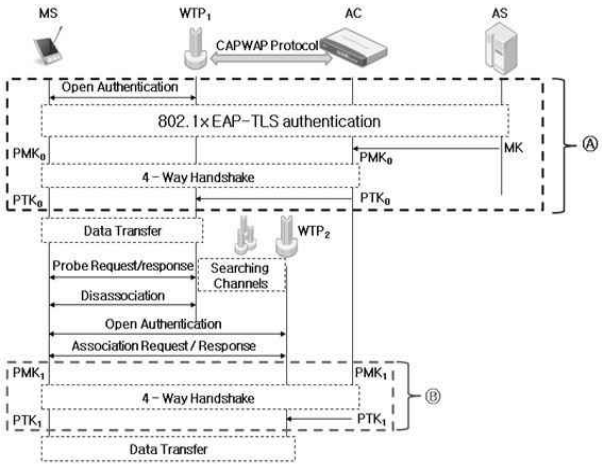

그림 4. CAPWAP의 인증 및 세션 키 도출 과정
Fig. 4. Authentication and Session Key Extraction Process of the CAPWAP

Ⅲ. OpenFlow 및 일반 AP 기반 끊김 없는 무선랜 핸드오버 구현

3-1 끊김 없는 무선랜 핸드오버 프로토콜 표준

안전하고 끊김 없는 802.11 무선랜 핸드오버를 위해서는 산

업 표준의 인프라 연동 모델인 ESS (Extended Service Set) 환경 하에서 신속한 BSS (Basic Service Set) 전이를 지원하는 IEEE 802.11r과, 무선 자원 관리 기능을 제공하는 IEEE 802.11k, AP를 제어하고 프로비저닝을 담당하는 중앙 콘트롤러, 기본 WLAN 기능을 담당하는 BSS, 그리고 이동 클라이언트 (무선 단말기)를 관리하는 AAA 서버 (Radius server) 등 크게 다섯 개의 요소가 필요한데 [10], 이들은 기본적으로 CAPWAP의 802.11 바인딩 (binding) 버전이다.

- IEEE 802.11r : 하나의 무선 통신 링크에 해당되는 BSS를 다른 AP로 전이하기 위한 안전 키 교환 프로토콜로서, 키 교환과 무선 자원 (AP) 할당을 병렬로 진행함으로써 로밍 시간을 단축한다 [11].
- IEEE 802.11k : 클라이언트 (무선 단말기)에 설치되어 AP를 중심으로 한 주변의 무선랜 자원 관리 체계 및 서비스 표준으로 클라이언트들이 가장 적합한 AP를 선택할 수 있는 정보를 제공한다 [12].
- 중앙 콘트롤러 : 전체 네트워크를 대상으로 BSS에 대한 정책 설정으로 가상 네트워크를 유지・관리한다.
- BSS : 하나의 AP와 해당 AP에 연결된 무선 단말기의 총칭으로 무선랜 통신 서비스의 기본 단위이다.
- AAA 서버 : 인증 (Authentication), 권한 부여 (Authorization), 계정 관리 (Accounting) 등 세 가지 기능을 제공하는 서버로서 요청・응답 형태로 동작한다.

다섯 가지 요소들에 의해 끊김 없는 핸드오버가 이루어지는 과정은 아래와 같다 (그림 5).

① 무선 단말이 AP1와의 거리가 멀어지면, 주변 AP들에게 응답 요청 메시지를 보낸다.
② 주변 AP들은 단말에게 가까이 있음을 알리는 응답 메시지를 무선 단말에게 보낸다.
③ 무선 단말은 주변 AP들로부터의 메시지를 받아 신호 세기 등을 고려하여 가장 적합한 AP에게 접속을 요청하여 링크를 맺는다. 이 때, AP는 인증 절차를 따른다.
④ 마지막으로 무선 단말은 AP1과의 링크를 해제한다.

그림 5. 802.11의 핸드오버 과정
Fig. 5. Handover Process of 802.11k

3-2 OpenFlow 가상 스위치

□ 하드웨어 플랫폼

OpenFlow 가상 스위치는 논문 [13]의 LVSAP (Light Virtual Switch for AP) 개념을 표 1의 RB450G 보드에 적용하여 구현한다. RB450G 보드는 그림 6과 같이 시스템 연결 포트 (eth0)와 스위치 연결 포트 (eth0.1 ~ eth0.5)를 분리하여 제공한다.

표 1. RB450G 보드 사양
Table 1. Speification of RB450G Board

Items	Specification
Bootloader	RouterBoot
SOC	Atheros AR7161 MIPS 24K
CPU Speed	680Mhz (Max 800Mhz)
Flash-Chip	Hynix NAND 512MiB
Flash size	512M
RAM	256M
Ethernet	Atheros AR8316
Serial	1 Port

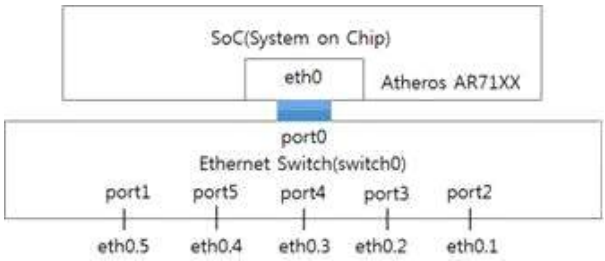

그림 6. RB450G 보드의 내부구조
Fig. 6. Internal Block Diagram of RB450G Board

□ OpenWrt 운영체제 빌드

무선랜 라우터 및 OpenFlow 가상 스위치를 탑재할 운영체제로 임베디드 리눅스 기반 OpenWrt (Open Wireless Router)를 이식・수정・구현하고, 그 주요 과정은 아래와 같다.

• 교차 컴파일을 위한 호스트 컴퓨터 환경 구축

PC에 Ubuntu ver.18.04 설치 후, 내려보낼 운영체제 이미지에 추가할 필수 모듈 패키지를 준비한다.

```
> sudo apt-get install build-essential subversion git-core
libncurses5-dev zlib1g-dev gawk flex quilt libssl-dev
xsltproc libxml-parser-perl unzip↵
```

• OpenWrt 소스 코드 준비 및 시스템 빌드

OpenWrt 커뮤니티에서 패키지를 다운받아 설정파일을 업데이트하고, 필요 패키지 선택 후 빌드한다.

```
> git clone git://git.openwrt.org/openwrt.git && cd
openwrt↵
>svn co -r r43011 svn://svn.openwrt.org/openwrt/trunk/ &&
cd trunk
> mv feeds.conf.default feeds.conf↵
> echo 'src-git openvswitch git://github.com/pichuang/open
  vwrt.git' >> feeds.conf↵
> ./scripts/feeds update -a↵
> ./scripts/feeds install -a↵
> make menuconfig↵
> make prereq↵
> make↵
```

□ OVS (Open vSwitch) 빌드

OVS 커뮤니티에서 패키지를 다운받아 설정파일을 업데이트하고, 필요 패키지 선택 후 가상스위치 전체 펌웨어를 빌드하여 내려보낼 운영체제 환경에 설치한다.

```
> ./scripts/feeds update -a↵
> ./scripts/feeds install -a↵
> echo 'src-git openvswitch git://github.com/pichuang/open
  vwrt.git' >> feeds.conf↵
> ./scripts/feeds update openvswitch↵
> ./scripts/feeds install -a -p openvswitch↵
> make menuconfig↵
> make kernel_menuconfig↵
> make↵
```

□ OpenWrt 및 OVS의 보드 설치 및 부팅

TFTP (Trivial File Transfer Protocol)를 이용하여 OpenWrt 및 OVS 패키지를 RB450G 보드에 설치하고 부팅시킨 다음 그림 7과 같이 OVS 커맨드로 확인한다.

그림 7. OpenFlow 가상 스위치 부팅 과정
Fig. 7. Booting Process of OpenFlow vSwitch

□ OpenWrt 네트워크 설정

그림 6의 내부 인터페이스에 따라 OpenWrt 내 네트워크 인터페이스를 그림8과 같이 설정한다. 이 설정의 주요 내용은 아래와 같다.

- 물리 네트워크 인터페이스 "eth0"에 가상 네트워크 인터페이스 "lan"을 생성하고 고정 IP 주소 "192.168.1.10"을 부여한다.
- 물리 네트워크 인터페이스 "eth0"에 가상 네트워크 인터페이스 "wan"을 생성하고, IP 주소는 DHCP로 받는다.
- 여러 개의 VLAN을 지원하는 스위치 "switch0"를 생성한다.
- 스위치 "switch0"에 0~4의 다섯 개 VLAN을 생성하여 AP와의 연결을 지원한다.

```
config interface 'loopback'
        option ifname 'lo'
        option proto 'static'
        option ipaddr '127.0.0.1'
        option netmask '255.0.0.0'

config globals 'globals'
        option ula_prefix 'fda8:999b:2415::/48'

config interface 'lan'
        option ifname 'eth0'
        option force_link '1'
        option type 'bridge'
        option proto 'static'
        option ipaddr '192.168.1.10'
        option netmask '255.255.255.0'
        option ip6assign '60'

config interface 'wan'
        option ifname 'eth0'
        option proto 'dhcp'

config interface 'wan6'
        option ifname 'eth0'
        option proto 'dhcpv6'

config switch
        option name 'switch0'
        option reset '1'
        option enable_vlan '1'

config switch_vlan
        option device 'switch0'
        option vlan '1'
        option ports '0 1 2 3 4'
```

그림 8. OpenWrt의 네트워크 설정
Fig. 8. Network Configuration of OpenWrt

□ OVS 브릿지 설정

가상 스위치 보드 (RB450G)의 AP 연결 이더넷 인터페이스 eth0.1과 연결할 OVS 내 브릿지를 아래 스크립트와 같이 생성하고 대응시킨 후, 가상 스위치와 OpenFlow 컨트롤러의 통신을 개시하고, 그림 9와 같이 확인한다.

```
> ovs-vsctl add-br ovsbr0↵
> ovs-vsctl add-port ovsbr0 eth0.1↵
> ovs-vsctl set-controller ovsbr0 tcp:192.168.1.5↵
```

```
root@OpenWrt:~# ovs-vsctl show
d626a1b7-1c98-49b4-8592-c4aa277ccd1a
    Bridge "ovsbr0"
        Controller "tcp:192.168.1.5:6653"
            is_connected: true
        Port "eth1"
            Interface "eth0.1"
        Port "ovsbr0"
            Interface "ovsbr0"
                type: internal
root@OpenWrt:~#
```

그림 9. OVS 내 브릿지 설정 확인
Fig. 9. Confirmation of Bridge Configuration in OVS

3-3 AAA Server

AAA 서버는 끊김 없는 핸드오버를 위한 핵심 컴포넌트로서, Ububtu 18.04 운영체제 하에 표 2의 소프트웨어를 이식·구현한다. 그림 10은 이들 소프트웨어의 연동 체계도인데, 이중 인증을 위한 핵심 소프트웨어는 Free RADIUS [14]이다. 그림 11에 FreeRADIUS의 설치 및 인증 시험 과정을 보였다.

표 2. AAA 서버 구성 소프트웨어
Table 2. Software Items for AAA Server

Function	Software
Authentiticate	FreeRADIUS 2
DBMS	MYSQL 5.7
IP Management	DHCP
Log Management	Rsyslog -mysql
Web Server	Apache 2
DNS	Bind

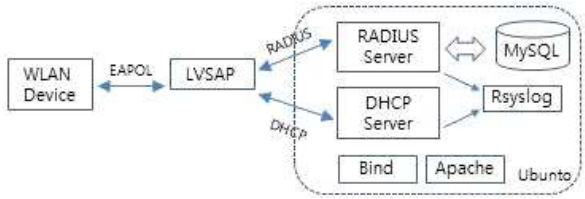

그림 10. AAA 서버 구성도
Fig. 10. Block Diagram of AAA Server

```
[패키지 설치]
neo@radius:~$ sudo apt-get install freeradius freeradius-mysql
freeradius-utils -y
[sudo] password for neo:
Reading package lists... Done
 :
[시험 계정 등록]
neo@radius:~$ sudo su -
root@radius:~# echo "testid01 Cleartext-Password := \"testpw\""
>> /etc/freeradius/users
root@radius:~# exit
neo@radius:~$

[사용자 인증 시험]
neo@radius:~$ radtest testid01 testpw localhost 0 testing123
Sending Access-Request of id 229 to 127.0.0.1 port 1812
        User-Name = "testid01"
        User-Password = "testpw"
        NAS-IP-Address = 127.0.1.1
        NAS-Port = 0
        Message-Authenticator = 0x00000000000000000000000000000000
rad_recv: Access-Accept packet from host 127.0.0.1 port 1812,
id=229, length=20
neo@radius:~$
```

그림 11. AAA 서버의 설치 및 시험 과정
Fig. 11. Installation and Test Process of AAA Server

3-4 OpenFlow 콘트롤러

OpenFlow 콘트롤러로는 ONOS (Open Network Operating System) 1.13.1 버전 [15]을 PC 리눅스 플랫폼 (Ubuntu 18.04)에 아래 스크립트와 같이 이식하여 구현한다. 그림 12에 ONOS 실행 모습을 보였다. 여기서 git clone은 해당 URL에 존재하는 패키지 환경을 로컬 시스템에 연결・구축한다.

```
> git clone https://gerrit.onosproject.org/onos/↵
> cd onos↵
> git checkout 1.13.1↵
> build/onos-buck build onos↵
> tar -xzfv buck-out/gen/tools/package/onos-package/onos.tar
  .gz -C $runtime_onos↵
> cd $runtime_onos↵
> bin/onos-service start↵
```

그림 11. AAA 서버의 설치 및 시험 과정
Fig. 11. Installation and Test Process of AAA Server

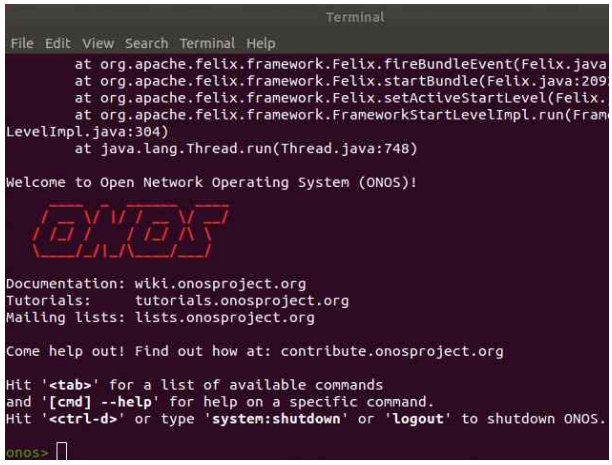

그림 12. ONOS (OpenFlow 콘트롤러) 실행
Fig. 12. Excution of ONOS (OpenFlow Controller)

3-5 핸드오버 어플리케이션

호스트 이동을 감지하여 플로우 테이블을 변경하는 “handover” 어플리케이션을 ONOS의 “Host Mobility” 어플리케이션을 이식하여 아래 스크립트와 같이 빌드하여 등록한다. 그림 13에 onos 서버에 번들된 “handover” 어플리케이션을 보였다.

```
> onos-create-app↵
> vi  pom.xml↵
> vi  src/main/java/org/myapp/handover.java↵
> mvn clean install↵
> onos-app localhost install target/org.myapp.handover-1.0.0.oar↵
[onos 콘솔에서]
onos> app activate org.myapp.handover↵
```

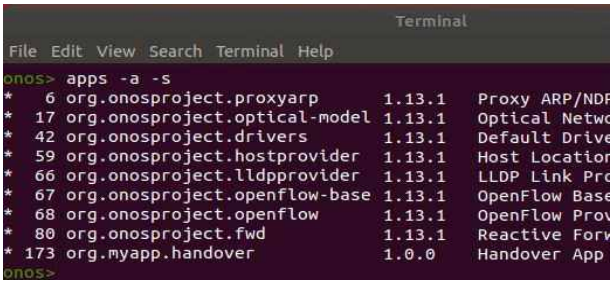

그림 13. 핸드오버 어플리케이션의 ONOS 서버 번들 확인
Fig. 13. Confirmation of Handover Application Bundling in ONOS Server

좀 더 구체적인 ONOS 어플리케이션 개발 과정은 아래와 같다.

- 어플리케이션 템플릿 생성
  - 어플리케이션 개발 디렉터리 (ONOS 소스 영역 밖)에서 “onos-create-app↵ ” 실행
  - groupId는 org.myapp, artiacrtId는 org.myapp.newapp, version은 1.0.0, onos-version은 1.13.1로 각각 입력하여 설정
  - pom.xml 파일과 src/main/java/org/myapp/newapp/AppComponent.java 소스파일이 생성됨
- $ONOS_HOME/apps 디렉터리에서 유사한 어플리케이션을 탐색하여 모든 소스파일을 newapp의 소스 디렉터리로 복사
- newapp의 pom.xml 파일 수정
  - onos.app.name은 org.myapp.newapp, onos.app.version은 1.0.0으로, 기타 onos.app,title 등을 수정
  - 탐색된 유사 app의 pom.xml과 newapp의 pom.xml 파일의 dependencies 파트를 비교하여 newapp의 pom.xml에 존재하지 않는 항목들을 복사. 복사 항목 중 version이 미지정된 것은 MAVEN Repositary (http://mavenrepositary.com)에서 검색하여 설정
  - Maven의 test 단계에서 실패하는 경우가 있으므로 src/test 디렉터리는 삭제
- newapp의 pom.xml 파일이 존재하는 디렉터리에서 “mvn clean install↵ ” 실행하여 어플리케이션 빌드

Ⅳ. 실험 및 평가

4-1 실험 환경 및 시나리오

무선랜 핸드오버를 실험하기 위해 그림 14와 같이 공유기를 통한 192.168.1.*의 내부 네트워크를 구성하고, 가상 스위치에 접근하는 무선 단말들은 DHCP로부터 192.168.3.*의 주소를 할당받아 무선 네트워크에 참여하도록 한다. 이 실험의 구체적인 상황 설정 및 전개는 아래와 같다.

- 두 개의 AP (ipTIME)을 다른 무선 네트워크가 미약한 건물 옥상에 25m 거리를 두고 배치한다.
- 802.11k가 탑재된 무선 단말기(Intel Wirelss Adapter)를 AP_0에 접근하여 무선 링크를 형성하고 3분 이상 길이의 유튜브 동영상에 접근한다.
- 무선 단말기를 AP_1 쪽으로 이동시켜 핸드오버를 유도한다.
- 무선 단말기의 동영상이 끊김없이 플레이되는지를 확인하다.
- 위 과정에 대한 주요 패킷을 Wireshark으로 캡쳐한다.

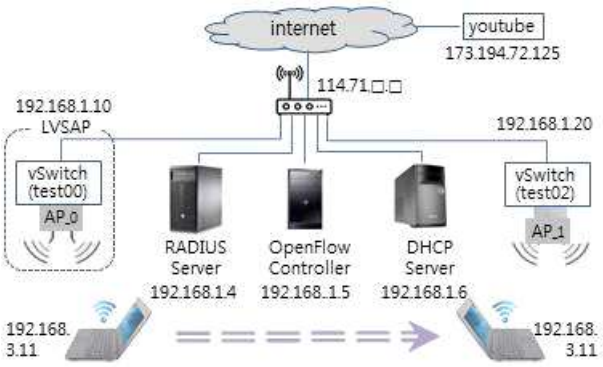

그림 14. 핸드오버 시험을 위한 실험환경
Fig. 14. Experiment Environment for Handover Test

4-2 실험 결과

AP_0를 통해서 동영상을 시청하다가 (그림 15 참조), 무선 단말기를 이동시켜 핸드오버 임계치에 다다르면 주변 AP를 스캐닝하고 (그림 16 참조), AP_1과 새롭게 링크 연결을 시도하여 핸드오버를 완료한 후 (그림 17 참조), 계속해서 동영상 스트리밍을 이어가는 모습 (그림 18 참조)으로부터 끊김 없는 핸드오버 기능이 작동하고 있음을 확인할 수 있다.

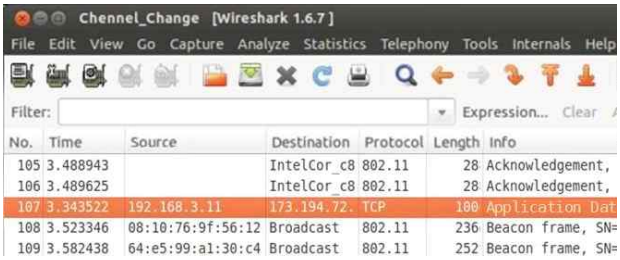

그림 15. 핸드오버 전 동영상 스트리밍 패킷
Fig. 15. Video Streaming Packet before Handover

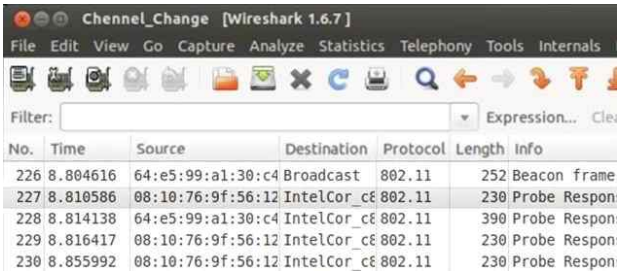

그림 16. 핸드오버를 위한 AP 스캐닝 패킷
Fig. 16. AP Scanning Packet for Handover

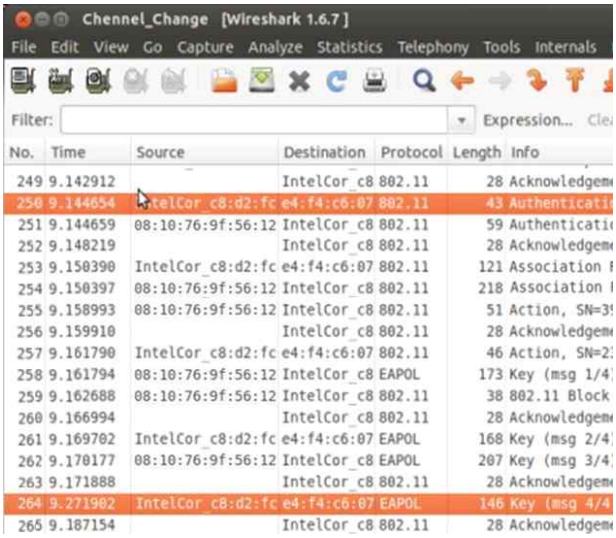

그림 17. 재접속을 위한 인증 및 키 교환 패킷
Fig. 17. Authentication and Key Exchange Packet for Association

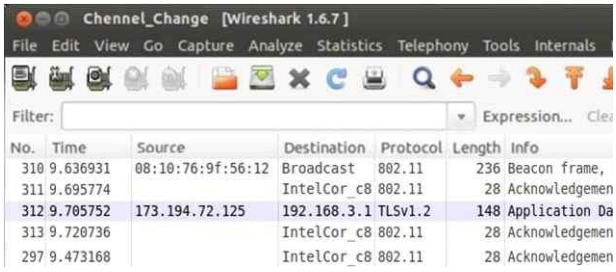

그림 18. 핸드오버 후 첫 동영상 스트리밍 패킷
Fig. 18. First Video Streaming Packet after Handover

4-3 성능 분석 및 평가

그림 15~18에 나타난 핸드오버 시간표를 요약하면 그림 19와 같고, 전체 지연시간은 832ms 이다. 이는 동영상과 같은 실시간 스트리밍 서비스를 위해 권고되는 핸드오버 최대 지연시간인 150ms [16]-[17]를 상당히 벗어날 뿐만 아니라, 이보다 더 나은 QoS를 위해 여러 개의 WLAN 카드를 활용하여 지연 시간을 34.7ms 까지 단축한 사례 [18]에 비춰보았을 때, 개선의 필요성이 강하게 제기되고 있다. 타당성이 높게 예상되는 개선 방향은 아래와 같다 (그림 20 참조).

- 무선 단말 주도 스캐닝 (active scanning)을 AP 주도 방식 (passive scanning)으로 전환하여 스캐닝 시간을 단축
- AP 재접속 시간과 플로우 테이블 업데이트 시간을 중복시켜 직렬처리 대신 병행처리 도입
- OpenFlow의 플로우 테이블 업데이트를 사후처리 (reactive update) 대신 사전처리 (proactive update) 도입

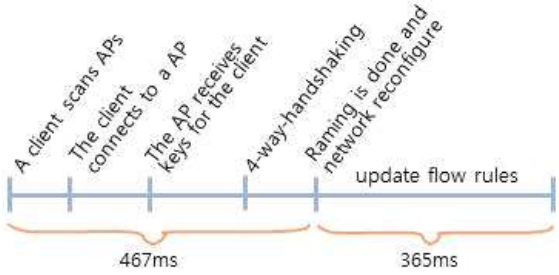

그림 19. 핸드오버 진행 타임 라인
Fig. 19. Time Line for Handover Progress

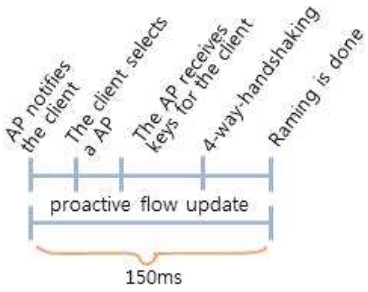

그림 20. 바람직한 핸드오버 진행 타임 라인
Fig. 20. Desirable Time Line for Handover Progress

Ⅴ. 결론 및 향후 계획

이 논문에서는 네트워크 가상화 플랫폼인 OpenFlow와 중저가형 일반 AP를 사용하여 무선랜을 구축하고 끊김 없는 핸드오버 기능을 구현하였다. 실험 결과 핸드오버 기능은 원할하게 이루어졌으나 핸드오버 지연시간이 길어 동영상과 같은 실시간 스트리밍 서비스를 위한 끊김 없는 핸드오버 관점에서는 미

흡한 결과로 나타났다. 그러나 핸드오버 지연시간을 단축시킬 수 있는 연구 방향이 구체적이고, 일부 벤더가 독점 공급하는 AP 가격의 1/5에 지나지 않는 일반 AP를 활용했으며, 무엇보다도 오픈 소프트웨어 기반 네트워크 가상화 플랫폼을 완성했다는 점에서 의의가 크다. 이어지는 연구에서 AP 주도 스캐닝, AP 재접속 과정과 플로우 테이블 업데이트 과정의 병행처리, 그리고 플로우 테이블의 사전처리 방법을 개발하여 핸드오버 지연시간을 150ms 이내로 단축시킬 예정인다.

참고문헌

[1] Wikipedia SDN, https://en.wikipedia.org/wiki/Software-defined_networking

[2] N. McKeown, T. Anderson, H. Balakrishnan et al., "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Computer Communication Review archive*, Vol. 38, Issue 2, pp. 69-74, April 2008.

[3] S. Lee, S. Chung, ""Performance Improvement via Flow-based Routing on OpenFlow-based Wireless Mesh Networks," *Journal of KIISE: Information Networking*, Vol. 40, No. 4, pp. 187-199, August 2013.

[4] A. Detti, C. Pisa, S. Salsano, N. B-Melazzi, "Wireless Mesh Software Defined Networks (wmSDN)," *Proceeding of IEEE Conference on Wireless and Mobile Computing*, pp. 89-95, October, 2013.

[5] W. Kim, S. Chung, H. Choi, and M. Do, "Contents Routing in the OpenFlow-based Wireless Mesh Network Environment," *Journal of KIISE: Information Networking*, Vol. 41, No. 10, pp. 810-823, October 2014.

[6] L. S. Puthalath, "Programming the Enterprise WLAN: An SDN Approach," Master's Thesis, Technical University of Lisboa, pp. 1-86, 2012.

[7] IETF RFC5415, Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification [Internet]. Available: https://tools.ietf.org/html/rfc5415.

[8] Wikipedia CAPWAP, https://en.wikipedia.org/wiki/CAPWAP

[9] Cicso Inc., Cisco Product Quick Reference Guide [Internet], Available: http://www.cisco.com/c/en/us/qrg/index.html

[10] IEEE, IEEE 802.11-2016 Standards [Internet], Available: http://standards.ieee.org/findstds/standard/802.11-2016.html

[11] K-H. Chi, C-C. Tseng, Y-H. Tsai, "Fast Handoff among IEEE 802.11r Mobility Domains," *Journal of Infromation Science and Engineering*, Vol 26, No. 4, pp. 1345-1362, July 2010.

[12] H. Robers, Mobile Devices and Wi-Fi [Internet], Available: https://community.arubanetworks.com/aruba/attachments/aruba/WLAN-Pro-Conf-EU-2014/1/1/WLANPro_EU_MobileDevices%2520v1.0-airheads.pdf

[13] H. B. Lee, J. H. Park, "Implementation of an OpenFlow-based Access Point Virtual Switch for Monitoring and Virtualization of Legacy Wireless LAN," Journal of The Korea Society of Computer and Information, Vol. 21, No. 1, pp. 65-72, 2016

[14] FreeRADIUS, FreeRADIUS Documentation, Available: https://freeradius.org/documentation

[15] ONOS, ONOS release, Available: https://downloads.onosproject.org/release

[16] J. H. Choi, S. W. Min, "Design and Implementation of a Novel Fast Handoff Algorithm for Streaming Service in Wireless LANs," *The Journal of Korea Information and Communication Society*, Vol. 36, No. 1, pp. 1-7, January, 2011.

[17] M. L. Tetarwal, A. Kuntal, P. Karmakar, "A Review on Handoff Latency Reducing Techniques in IEEE 802.11 WLAN," *Proceeding of National Seminar on Recent Advances in Wireless Networks and Communications(NWNC-2014)*, No. 2, pp. 22-28, 2014.

[18] P. Dely, A. Kassler, L. Chow, N. Bambos, N. Bayer, H. Einsiedler, et. al, "A software-defined networking approach for handover management with real-time video in WLANS," *J. Mod. Transport*, Vol. 21, No. 1, pp. 58-65, 2013.

이형봉 (Hyung-Bong Lee)

1984년 : 서울대학교 계산통계학과(학사)
1986년 : 서울대학교 대학원 계산통계학과(석사)
2000년 : 강원대학교 대학원 컴퓨터과학과(박사)

1986년 ~ 1994년: LG전자 컴퓨터연구소
1994년 ~ 1999년: 한국디지털(주)
2004년 ~ 현 재: 강릉원주대학교 컴퓨터공학과 교수
※관심분야 : 무선 통신 (Wireless Networks), 센서 네트워크 (Sensor Networks), 임베디드 시스템 (Embedded Systems), 사물 인터넷 (IOT) 등

권기현 (Ki-Hyeon Kwon)

1993년 : 강원대학교 컴퓨터과학과(학사)
1995년 : 강원대학교 대학원 컴퓨터과학과(석사)
2000년 : 강원대학교 대학원 컴퓨터과학과(박사)

1998년 ~ 2002년: 동원대학 인터넷정보과 교수
2002년~ 현 재: 강원대학교 정보통신공학과 교수
※관심분야 : 패턴 인식 (Pattern Recognition), 이미지 처리 (Image Processing), 사물 인터넷 (IOT) 등