Load-aware Hand-offs in Software Defined Wireless LANs

Anil Kumar Rangisetti, Baldaniya Hardik Bhopabhai, B Pradeep Kumar and Bheemarjuna Reddy Tamma
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad, India
Email: [cs12p1001, cs10b007, cs10b006, tbr]@iith.ac.in

***Abstract*—In Wireless Local Area Networks (WLANs), providing seamless mobility and balancing load among Access Points (APs) are challenging issues due to simple signal strength based association and hand-off mechanisms employed at wireless clients. Extensions to Software Defined Networking (SDN) framework for wireless networks could help to address theses issues in an efficient and cost-effective manner with a central view of WLAN at the SDN controller. In this work, we propose a novel load-aware hand-off algorithm for SDN based WLAN systems which considers traffic load of APs in addition to received signal strength at wireless clients to solve load imbalance among APs and offer seamless mobility. We implemented the proposed algorithm on a small-scale prototype testbed and obtained improved network throughput for mobile clients as well as static clients compared to legacy hand-off algorithms used in WLANs.**

I. Introduction

Now a days wireless radios based on IEEE 802.11 a/g/n standards are heavily being used for connecting wide-range of devices to the Internet, due to their capabilities in offering high throughput and support for relatively large number of users. In the view of supporting wide variety of wireless client devices and Access Points (APs), various Wi-Fi chip makers in the market developed wireless client's association and hand-off procedures mostly based on received signal strength indicator (RSSI) parameter. But, these RSSI based association decisions can create load imbalance and un-necessary hand-offs in enterprise WLAN deployments [1]. In order to address these issues, we can have a central controller having network state information like number of clients associated per AP, traffic load, average data rates, average RSSI values, operating modes and neighbouring APs. Various network solution providers are developing centralized architectures for solving these issues. As these architectures are typically proprietary in nature, their controller software/firmware are not extendible for incorporating new modules and services.

Software Defined Networking (SDN) architectures [2] are innovative, dynamic, easily manageable and scalable centralized architectures. In SDN, control and data planes are decoupled to simplify network management and control operations. Here, network control algorithms and its state are logically centralized so that the underlying network infrastructure can be represented as a network API for the applications. Innovative SDN evolution motivated wireless networking research community to focus on its applicability for wireless networks and creating programmable wireless network architectures. In [3], [4], the authors addressed some challenges faced while extending SDN for WLANs. As WLANs are needed to operate in dynamically changing operating environment, these architectures require network aware solutions for addressing hand-off, load-balance and other issues. In this work, we propose a novel load-aware hand-off algorithm for SDN based WLAN systems which considers traffic load of APs in addition to received signal strength at wireless clients to solve load imbalance among APs and offer seamless mobility.

II. Related Work and Motivation

OpenRoads [3] is the first successfully deployed programmable SDN-based WLAN architecture for testing various WLAN algorithms. It is a layer based architecture, namely a physical layer, a network virtualization/slicing layer and a controller layer. It discussed about how the OpenFlow [5] can be used by wireless researchers for performing hand-off between different wireless technologies. CloudMAC [6] is a data center based management architecture in which APs just forward MAC frames. Other functions such as processing MAC frames are implemented on standard servers of the data center. SDN architectures are also helpful for Wireless Mesh Networks (WMNs). OpenFlow based WMN architectures help in design of flexible packet routing algorithms for WMNs. In [7], the authors demonstrated usefulness of OpenFlow based WMN system for enabling client mobility to achieve fast handovers at low complexity and overhead.

ODIN is a programmable WLAN architecture [4] for simplifying association decisions of clients in enterprise WLANs. It provides a basic framework for designing various WLAN management algorithms at the SDN controller. Thanks to ODIN [4] architecture for its open source [8] support, in this work we extend it further for offering load-aware hand-offs in enterprise WLANs. The following are the changes that are made to ODIN framework in the proposed SDN based WLAN system:

- To have periodic network load knowledge at the central SDN based wireless controller, we introduce a new message called PUBLISH-AGENT-LOAD.
- As monitoring of clients associated with neighbor APs is needed, with the help of second radio available, each AP monitors traffic on other channels in the wireless medium and checks for sending hand-off initiation message to the controller.

978-1-4799-5041-6/14/$31.00 ©2014 IEEE

- In order to extend ODIN architecture to multi-channel WLAN environment, we used IEEE 802.11 Channel Switch Announcement (CSA) messages in beacon frames for reducing delay during hand-off between different channels.
- To address RSSI based clients association issues, we propose a novel load-aware hand-off algorithm which runs on the top of SDN based central controller as an application.

These changes are necessary for doing efficient and cost-effective hand-offs at the centralized controller.

III. Our Work

SDN based WLANs are employed to solve interference, loadbalance and hand-off issues with help of global network knowledge. Major goals of proposed SDN based WLAN architecture are as follows:

- Realize adaptive WLAN framework over existing enterprise WLAN systems.
- Simplify WLAN management with policy based management and virtualization of network resources.
- Implement efficient algorithms to achieve load balancing, seamless mobility and interference management in enterprise WLAN environments.
- Maintain compatibility with existing IEEE 802.11 WLAN protocols.

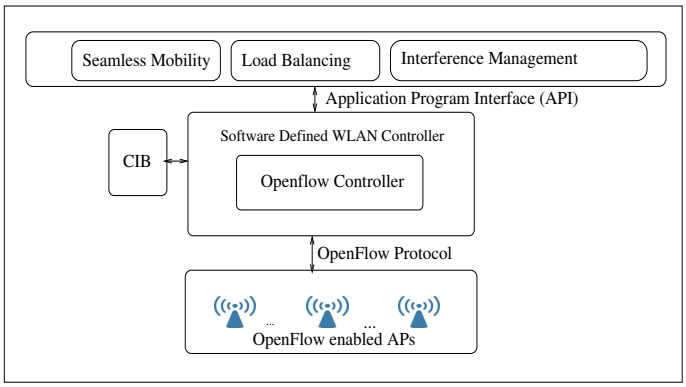

Fig. 1. Architecture of Software Defined Enterprise WLAN System

A. Framework Design

Fig. 1 shows the architecture of proposed OpenFlow enabled SDN framework for enterprise WLAN Systems. Software Defined WLAN Central Controller (CC) is the main component in this architecture. The CC is realized using *Floodlight* [9] controller. It monitors entire network traffic and events to optimally reconfigure the network elements in a flexible manner. All the necessary information regarding network state is maintained in a Central Information Base (CIB) at CC. The CC can be designed by extending the standard OpenFlow controller so that CC could have complete view of enterprise WLAN. Another important component in this architecture is the Assistive Agent (AA) click [13] based module installed on AP. We extend standard 802.11 AP with AA for communicating with CC all necessary parameters like load of AP, per client traffic statistics and other events (e.g, hand-off, load exceeded) related to WLAN operation. APs are integrated with *Open vSwitch* [10] to enable OpenFlow support and handle traffic flows. CC updates flows during hand-off with OpenFlow messages. We call these APs as OpenFlow enabled APs. We implemented the load-aware hand-off algorithm as a separate application in *Floodlight*. Following are the main challenges that we addressed in designing load-aware hand-off algorithm for SDN based WLAN system:

- How does CC decides hand-off event to be triggered from OpenFlow enabled APs?
- What is the definition of load at each AP in the WLAN system?
- How to provide network state information (AP load, clients traffic and signal strengths) to CC?
- What are the parameters to be considered for taking load-aware hand-off decisions?
- How to take care of ping-pong that can occur during hand-offs?
- If all major decisions are taken at CC, then how to prevent overloading of it?
- How to ensure clients can change their channels (if target AP is on some other channel) seamlessly during hand-off process?

In order to perform load-aware hand-offs by addressing above issues, we introduce a few new messages in ODIN programmable WLAN architecture [4]. ODIN adapted Virtual Access Point (VAP) concept supported by Atheros chips for creating unique Basic Service Set Identification (BSSID) for each client. ODIN called these VAP as Light VAP (LVAP). During hand-offs, with LVAP the client could have consistent and continuous connection even when it is physically moved from one AP to other AP. It focused on reducing hand-off delay with ADD-LVAP and REMOVE-LVAP messages, but it has a major limitation that both APs need to operate on the same channel for the hand-off to work seamlessly. CC sends ADD-SUBSCRIPTION (*e.g.*, RSSI $>$ Signal Threshold (STHR)) message to APs so that APs could trigger hand-off events with PUBLISH messages. As with the same RSSI threshold based subscription message for all APs can lead to load unaware hand-offs, in this work, we introduce a unique hand-off subscription message for each AP. To efficiently handle these issues and to provide load-aware hand-offs for improving network performance PUBLISH-AGENT-LOAD, PUBLISH-CLIENT-LOAD and UPDATE-SUBSCRIPTION messages are introduced. The first message is for sending cumulative load of AP, next message is for sending a client's load (average RSSI, total number of packets, and average data rate) and last message is for updating hand-off subscription per AP according to its load. In order to prevent overloading of the CC these messages are generated periodically from APs. To address the issue of seamlessly handing over a client between APs configured on different Wi-Fi channels, we added

CSA element in beacon messages transmitted during hand-off process. In our work, connected AP is the one that sends out beacons with CSA (ElementId = 37 and length = 3 fields) element to the targeted client which is then moves to the new channel, but not connected AP unlike in 802.11 DFS standard. As shown in Fig. 2, CSA element consists of three fields: *channel switch mode, new channel and count*. *Channel Switch Mode* indicates whether client receiving CSA element needs to suspend its transmissions until it has changed to new channel (i.e., target AP) or not, *count* indicates maximum number of beacons with CSA that it may receive before changing to *new channel*. We configured *mode=1* and *count=3* to ensure that client should be able to change its radio to the new channel on or before receiving three beacons from the currently associated AP. We tested clients with Linux kernel 3.5 version, which is capable of processing CSA element to change current operating channel seamlessly. This solution does not require any client side driver modifications so it is more easily deployable solution. With these changes we could able to improve network performance without increasing hand-off delay. We have shown how these messages are helpful in realizing load-aware hand-offs with an example sequence diagram in Fig. 3.

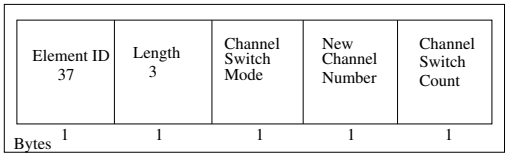

Fig. 2. Format of Channel Switch Announcement Element

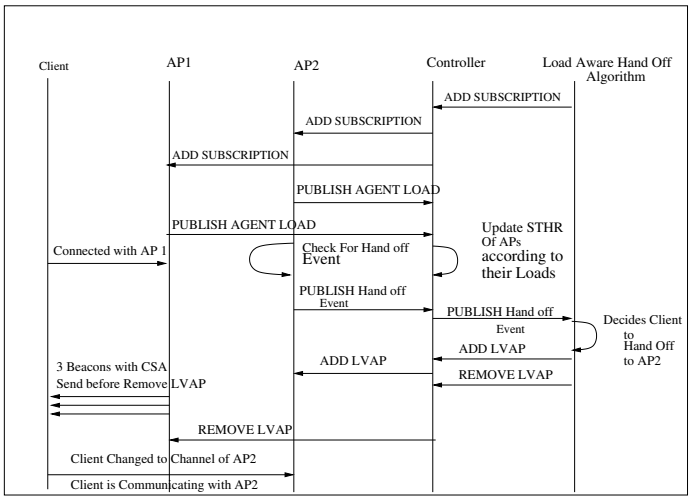

Fig. 3. Messages exchanged between APs, Controller and Load-Aware Hand-off Algorithm during client hand-off between APs

B. Parameters considering for load-aware hand-off decision making

Hand-off algorithm with fixed STHR on all APs can cause load imbalance in WLANs, limiting coverage area of AP even when it is lightly loaded, delaying hand-off decision for a client moving from heavily loaded AP to lightly loaded AP, and vice versa. To overcome these issues, the proposed load-aware algorithm considers load of the AP in addition to RSSI values. Load of an AP is defined with a parameter called Traffic Intensity (TI) [11]. This parameter measures utilization of channel resources in a range from 0 to 1, where 0 indicates channel is idle and 1 indicates channel is fully occupied. TI is defined as the amount of time the AP is busy (AP-BUSY-TIME) with transmission or reception of N frames in a given time period T (in seconds) as given in Eqn (1). Its calculation includes all Data, Management, and Control including duplicate and non-header error-prone MAC frames. For each frame, we calculate its duration (d_i) by its frame-length and PHY data rate. APs send their TI(t) values to the CC in PUBLISH-AGENT-LOAD messages.

$$\text{TI(t)} = \frac{\sum_{i=1}^{N} d_i}{T} \tag{1}$$

For each AP, its effective TI is calculated at the CC as given below in Eqn (2),

$$\text{CTI(t)} = 0.9 * \text{TI(t)} + 0.1 * \text{CTI(t-1)} \tag{2}$$

In rest of this paper, we referred the terms load and TI interchangeably. We have given 90% weightage to current time period changes in TI. Load-aware algorithm uses two parameters to take hand-off decisions: RSSI and TI. We do not change any client side driver modifications and they still try to associate to APs based on RSSI values, so the main requirement of AP is to report to the CC when a client is moving out of its range. In order to reflect AP decision based on both RSSI and load levels, we map RSSI values against load levels (*low, medium and high*). To do this mapping, we need to analyze RSSI variations in WLAN deployment environment. To analyse these RSSI variations, we conducted a simple experiment with the testbed shown in Fig. 4. We

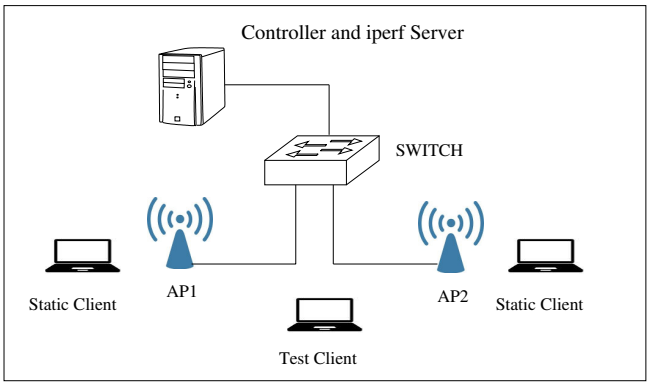

Fig. 4. Architecture of the Testbed

need to determine minimum and maximum RSSI that can be configured in the test environment. To accomplish this, we placed a test client very close to an AP and slowly moved that client away from it to find out where it is getting disconnected (i.e., trying to re-associate with some other AP). In our lab

environment, the minimum RSSI is 185 where the client is trying to re-associate and the maximum RSSI is 235 when the client is very nearer to the AP. In order to identify maximum possible variations in RSSI values, we conducted a simple experiment as follows:

- Place a Static Client (SC) in the middle of two APs.
- Then SC runs *iperf* TCP for the 180 seconds duration. We measured max and min RSSI values experienced by it over the duration of the flow.
- RSSI values are averaged over one sec time interval.
- Then by using EWMA with weightage of α = 0.6, 0.8 and 1 (α indicates weightage given to averaged RSSI values over current one sec and (1-α) indicates weightage given to historical RSSI estimate), we calculated RSSI estimates.

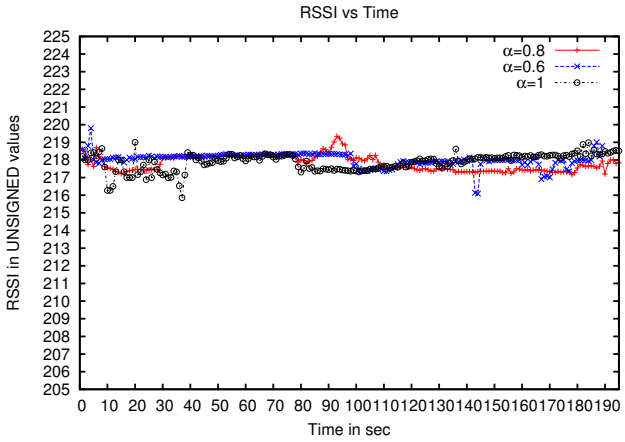

Fig. 5. RSSI variations with α = 0.6, 0.8, 1

The variations in RSSI values are shown in Fig. 5. With α=0.8 compared to 0.6 and 1, RSSI variations are stabilized in our lab test environment. From the experiment results and graphs, we determined the maximum variation in RSSI values as [4 to 7]. In order to ignore these variations at any second, we used difference between RSSI threshold values of two load levels as 15 and same value is also used for hysteresis setting to avoid ping-pong effects during hand-off of a client from one AP to another.

C. Load-aware Hand-off Algorithm

The Algorithm1 describes the proposed load-aware hand-off algorithm running on the CC.

The initial STHR is fixed at 185 for all APs in the WLAN system. It's the value below which a client cannot be served by any AP. The TI value for each AP is updated regularly at the CC, which helps in updating STHR according to APs' load. At CC, updated STHR value of the AP is used for detecting triggering of hand-off events. As we are changing STHR dynamically based on load levels of APs, there is a possibility of ping-pong effect between the APs. In our algorithm first condition-1 ensures that client is handing-off to less loaded AP and at the same time signal strength experienced with respect to less loaded AP is good enough for

Algorithm 1 Load-aware Hand-off Algorithm

1: CC sends initial hand-off subscription (RSSI $>$ STHR) messages to APs in WLAN system
2: CC gets updated loads statistics (TI) of APs periodically
3: **loop**
4: CC updates STHR value to be matched against APs' load levels
When Client starts moving to the overlapping region of APs then hand-off decision is taken as follows
5: **if** (($Load_{connectedAP}$ - $Load_{competingAP}$) $> HT_{LOAD}$) and ($RSSI_{connectedAP} < (RSSI_{competingAP} + HT_{RSSI1})$) **then**
6: Hand-off the client to the competing AP
7: **else if** (($RSSI_{competingAP} > (RSSI_{connectedAP} + HT_{RSSI2})$) and ($Load_{competingAP} < (Load_{connectedAP} + HT_{LOAD})$)) **then**
8: Hand-off the client to the competing AP
9: **else**
10: No Hand-Off
11: **end if**
12: **end loop**

TABLE I
TEST SETUP FOR THE EXPERIMENTS

Number of APs	2 (Alix3d3 boards)
AP Wi-fi Cards	2 (Ath9k 802.11bgn)
APs' operating mode	802.11g
Operating System on AP	Openwrt
Other tools on AP	Click2.0.1 and Open vSwitch1.9.0
Number of Clients	3
Linux Kernel version of Clients	3.5
Controller Software	Floodlight
Initial STHR	185
HT_{LOAD}	30%
HT_{RSSI1}	10
HT_{RSSI2}	15
Channels Tested	3, 9 of 2.4GHz
α	0.6, 0.8 and 0.9
Time Interval (T)	5 seconds

providing better QoS. Second condition-2 ensures that even when load difference between APs is below load hysteresis margin, if a client is experiencing better signal with less loaded AP then hand-off that client to that AP. In order to avoid the ping-pong effects, we are using two RSSI based hysteresis thresholds (HT_{RSSI1} and HT_{RSSI2}) and one load based hysteresis threshold (HT_{LOAD}). With the help of HT_{RSSI}, we can guard client against moving between APs due to rapid fluctuations in RSSI values. In order to avoid ping-pong effect due to sudden changes to load differences after client hand-off between APs, we make use of the HT_{LOAD}. The two RSSI HTs are required for avoiding boundary-case ping-pong effects. HT_{RSSI1} should be always less than HT_{RSSI2} to ensure the same.

IV. Experimental Setup and Performance Results

We have conducted the following experiments on an experimental testbed to show that the proposed load-aware hand-off algorithm improves the network throughput in enterprise

WLAN environments. The performance of proposed algorithm is compared with that of Fixed RSSI algorithm. The *iperf* [12] TCP tests are conducted with the testbed shown in the Fig. 4. Details of testbed are given in Table. I. The load offered by clients is generated using *iperf* TCP client/server tools. The testbed comprised of two OpenFlow enabled APs, each fitted with two Wi-Fi radios. These APs ran click based Wi-Fi agents. These APs and server are connected to the same wired network. The server ran, *iperf* and Floodlight based Controller (CC) with load-aware hand-off module.

Experiment 1: Aim of this experiment is to show that how load-aware hand-off algorithm can provide improved network throughput to static clients which are in overlapping region of APs. This test setup involved 2 APs and 3 clients. This experiment ran for a total duration of 240 sec. Initially, AP1 has two clients connected to it, one static client (SC1) placed near AP1 and one Test Client (TC) placed at a position where RSSI values experienced by it are approximately equal with respect to both APs. Another Static Client (SC2) is connected to AP2 by placing it nearer to AP2. This experiment is conducted twice and averaged values for every 10 sec are reported in Fig. 6 and Fig. 7. At time t=0 sec, *iperf* TCP sessions with duration of 240 sec and 120 sec are started in the SCs connected to AP1 and AP2, respectively. Later at t=60 sec, *iperf* TCP session with 180 sec duration is started in the TC. Now SC1 and TC are sharing bandwidth equally (as shown in Fig. 6 and Fig. 7). The load of AP2 is lowered after SC2 completed its TCP session at t=120 sec. Load-aware hand-off algorithm by detecting this load imbalance, handed-over (at t=123 sec) TC to AP2 (as shown in Fig. 7), resulting in improvement in the average network throughput from 4.5 Mbps to 9 Mbps during 130 to 240 sec. Moreover, the average throughput of SC1 with AP1 also experienced an increase from 4 Mbps to 9 Mbps. But in the case of legacy fixed RSSI hand-off scheme, the TC is still connected to AP1 (as shown in Fig. 6), due to which entire network throughput suffered. Compared to Fixed RSSI algorithm, proposed load-aware hand-off algorithm resulted in improvement of network throughput by 9 Mbps.

Experiment 2: WLANs operating in same channel environment could suffer from lot of interference, which can decrease entire network throughput. So in this experiment we show how our solution seamlessly handed-off clients between APs operating on orthogonal Wi-Fi channels, which was not considered in ODIN [4]. As we discussed in Section III, CSA message embedded in Wi-Fi beacon is used for changing client's channel seamlessly during hand-off. This solution does not require any client side modifications except that clients' MLME should be able to process beacon having CSA element. In this experiment TC is initially connected with AP1 and it started *iperf* a TCP session with 120 secs duration. In first 60 secs, TC physically moved from AP1 to AP2 and in the remaining 60 sec it moved back from AP2 to AP1. As shown in Fig. 8, in the first part of the experiment hand-off occurred at t=40 sec and in the next part it occurred at t=90 sec (approximately). There are two key points to be observed in

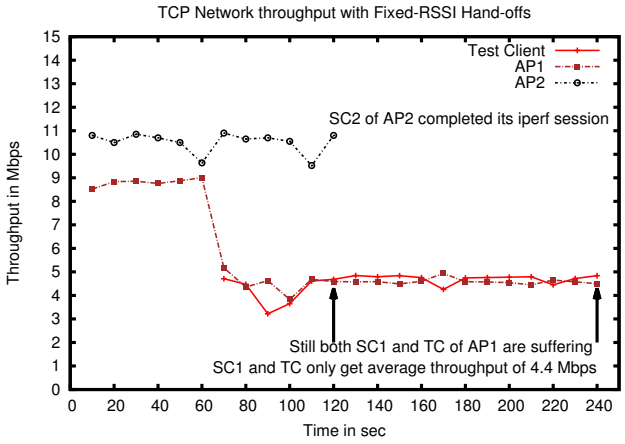

Fig. 6. Network throughput when APs load change dynamically for Fixed RSSI hand-off algorithm

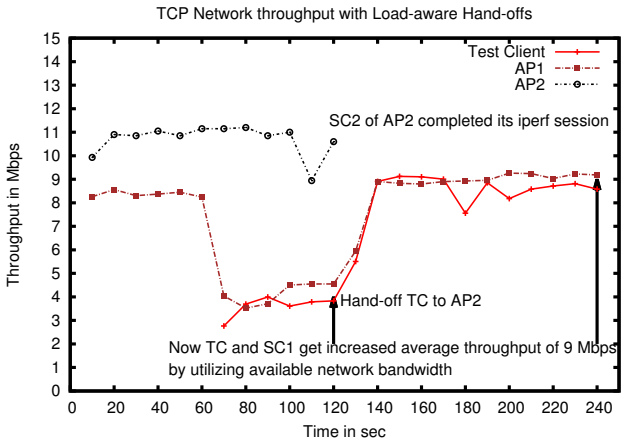

Fig. 7. Network throughput when APs load change dynamically for Load-aware hand-off algorithm

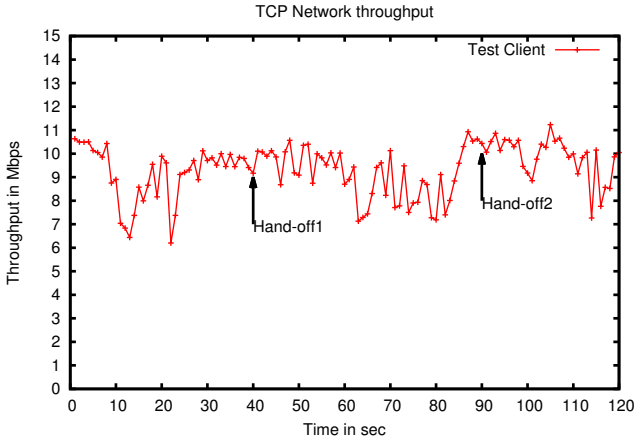

Fig. 8. Test Client Throughput when it moves between two APs configured with channels 3 and 9 respectively

Fig. 8. One, there is no sudden throughput drop at the time of hand-off and second, there are no ping-pong effects for the

TC. Fig. 8 clearly shows that there was no throughput drop at any hand-off point in the entire duration of experiment.

Experiment 3: Main goal of this experiment is to show that load-aware algorithm takes efficient decisions when there are lightly loaded APs in overlapping serving region or when all serving APs are equally loaded. This experiment is conducted with two APs and two clients. Initially both clients are connected to AP1 by placing them near to AP1. This causes AP1 is heavily loaded when compared to AP2. This experiment is repeated twice and averaged values are plotted for every 10 secs. At t=0 sec, on both clients *iperf* TCP session is started for 180 secs duration. Later one of the clients (called as TC) started physically moving towards AP2 for 90 secs. During this movement, TC is handed-off to the lightly loaded AP2 by load-aware algorithm between t=20 sec and t=23 sec (as shown in Fig. 10) but with Fixed-RSSI hand-off algorithm TC is handed-off to the lightly loaded AP2 between t=50 sec and t=53 sec (as shown in Fig. 10). From the graphs, it can be observed that in the duration t=20 sec to t=50 sec, early hand-off by load-aware algorithm improved throughput for both TC (increased average of 3.8 Mbps as shown in Fig. 10) and static client (increased average of 4.5 Mbps as shown in Fig. 9) when compared to fixed-RSSI algorithm. In next 90 seconds of the experiment, both APs are equally loaded, so when TC started moving backward to AP1, it is handed-off to AP1 between t=130 sec and t=133 sec by both the algorithms. When both APs are equally loaded, as the load-ware algorithm also considers better RSSI for hand-off decision, the hand-off occurred at around same time for both the algorithms. From the graphs (as shown in Fig. 9 and Fig. 10), we can observe that both the algorithms are performing equally well when the APs are loaded equally.

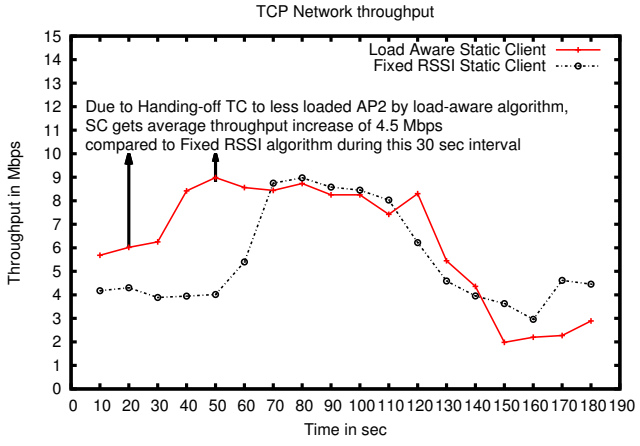

Fig. 9. Network throughput of Static client when it is moving between low load and high load APs with respect to Fixed RSSI and Load-aware hand-off algorithms

V. Conclusions and Futurework

In our work, we implemented the proposed load-aware hand-off algorithm on a Software Defined WLANs. We conducted experiments on a small scale testbed to test performance of the proposed algorithm. Experiment 1 and 3 showed that both static and mobile clients in the overlapping region of APs can associate with less loaded AP automatically and utilize the available network bandwidth, by using the proposed algorithm. In contrast to ODIN, our proposed architecture can be deployed in multi channel WLAN environment and still achieve seamless hand-off with an advantage of controlled interference in network. Future work includes thoroughly testing proposed SDN based WLAN framework and Load-aware hand-off algorithm on a production scale enterprise WLAN environment.

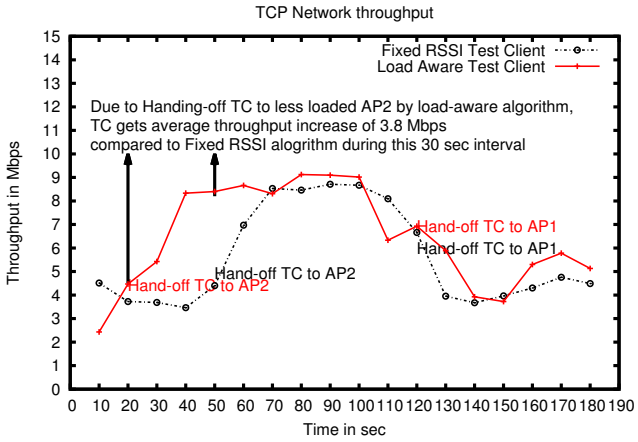

Fig. 10. Network throughput of Test client when it is moving between low load and high load APs with respect to Fixed RSSI and Load-aware hand-off algorithms

References

[1] D. Gong and Y. Yang, "Ap association in 802.11 n wlans with heterogeneous clients," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 1440–1448.

[2] "IBM Software Defined Networking," www.ibm.com/systems/networking/solutions/sdn.html.

[3] K.-K. Yap, M. Kobayashi, R. Sherwood, T.-Y. Huang, M. Chan, N. Handigol, and N. McKeown, "Openroads: Empowering research in mobile networks," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 125–126, 2010.

[4] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao, "Towards programmable enterprise wlans with odin," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 115–120.

[5] "OpenFlow," https://www.opennetworking.org/sdn-resources/onf-specifications/openflow.

[6] J. Vestin, P. Dely, A. Kassler, N. Bayer, H. Einsiedler, and C. Peylo, "Cloudmac: towards software defined wlans," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 16, no. 4, pp. 42–45, 2013.

[7] P. Dely, A. Kassler, and N. Bayer, "Openflow for wireless mesh networks," in *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*. IEEE, 2011, pp. 1–6.

[8] "ODIN git source," https://github.com/lalithsuresh/odin.

[9] "Floodlight OpenFlow Controller," http://www.projectfloodlight.org.

[10] "Open vSwitch," http://openvswitch.org.

[11] B. R. Tamma, N. Baldo, B. Manoj, and R. R. Rao, "Multi-channel wireless traffic sensing and characterization for cognitive networking," in *Communications, 2009. ICC'09. IEEE International Conference on*. IEEE, 2009, pp. 1–5.

[12] "Iperf," https://iperf.fr/.

[13] "Click Modular Router," www.read.cs.ucla.edu/click/click.