2015 The 5th International Workshop on Computer Science and Engineering
doi: 10.18178/wcse.2015.04.057

Method of Interaction Between Wireless and SDN Controllers to Manage a Wireless Network

Yuri Ushakov, Pert Polezhaev, Alexander Shukhman

Department of Mathematics, IT Center, Orenburg State University, Orenburg, 460018, Russian Federation

Keywords: computer network management, wideband, wireless networks.

Abstract. Wireless networks are widely used in many organizations. But administrators have problems with configuring and supporting wireless networks. Nowadays SDN protocols can significantly simplify network management but don't support specific features of wireless networks. Some projects have been developed to support wireless protocols as additions to OpenFlow. This paper describes the use of traditional managing methods for wireless network segments in combination with OpenFlow. SDNs are used to control traffic routing. The SDN controller can redirect some requests back to the wireless controller using REST API. Most wireless controllers support REST and can be used with this solution.

Introduction

In the past, wireless networks were used only for low-speed applications. Users didn't need broadband services or QoS. The present day technologies put forth the necessity for updating wireless infrastructure. But wireless protocols such as IEEE 802.11 developed in the 90s and 2000s don't provide flexible management and control plane. Some wireless protocols (CAPWAP, LWAPP and so on) have been developed to manage wireless infrastructure. But they cannot be used to fully control the traffic, arrange multicast in wireless networks, or to uniformly manage both wired and wireless infrastructure. WLAN provides easy connection to the network and high speeds up to 1700 Mbit/c (802.11ac). Control frames are exchanged between stations (STA) and the access point (AP) during transferring. In addition, the access points can share a certain context-sensitive information (a set of rules and parameters) or specific to a station. Mobile wireless networks allow to use many promising multimedia and peer applications (such as VoIP, 802.11 telephony, mobile video conferencing, chat). Unfortunately, the quality of network connection may be affected by volatility delay in handover to another access point. The measurements have shown that in addition to rather large delays themselves, they are significantly different for different configurations of stations and access points.

We propose the use of OpenFlow protocol for a unified control system. Such a system has modules for wireless networks controlled through REST or CAPWAP. Some wireless control systems require changes in firmware [1] for remote control. We assume the wireless infrastructure has already been installed and cannot be changed.

Control plane features

Modern managed access points can be controlled through both HTTP (REST) and CAPWAP (LWAPP, Netconf) methods. If a company already has a controller-based wireless network, the REST method is preferable. When OpenFlow is used for LAN management, then we can use the same controller to control the whole network (fig.1).

The OpenFlow controller acts as a network operation center (NOC) using OpenFlow to control LAN traffic, REST to control wireless controller and CAPWAP to control the managed APs.

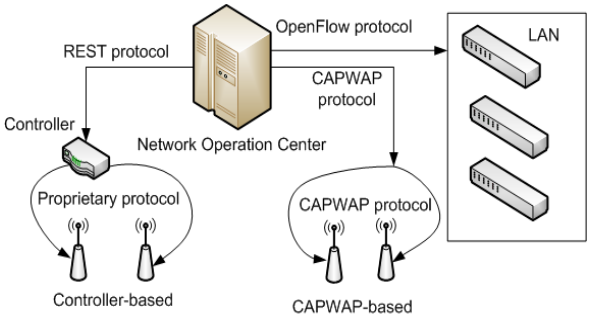

Fig.1: Hybrid control system

We used a third-party script for interception of connections between controllers and access points based on the man-in-the-middle technique. When devices try to connect, the script saves the socket data and selects an optimal controller to serve it. Then the script uses an address translation to masquerade the connection between the switch socket and associated controller socket.

When delay-sensitive traffic appears in the network, we need to rearrange client associations to optimize the load on access points. Some methods of forced hand-off don't work with CAPWAP: when clients look for SSID and make a decision about BSSID to connect to a new access point. Windows-based clients use only signal-preferred logic for connecting to competitive access points. Additionally, control frames on data-link layer are handled and generated within hardware and are not managed by CAPWAP. This protocol is only used to initiate de-association with an access point.

Advanced broadband services

The centrally controlled nature of SDN-based networks leads to some problems with scalability of the architecture. Those problems are scalability bottlenecks in the control platform and bottlenecks in the switches.

Broadband services include some usual traffic flows (file sharing, web-casting), multicast flows (IPTV, VOD) and delay-sensitive applications (VOIP, VOD). Calculating QoS parameters we should consider traffic features. An AP originally is not meant to provide QoS except traffic priority management. When traffic is started, the controller calculates QoS parameters, routes, and wireless options to determine the optimal path. Additionally, the controller makes a decision which AP should be used for different types of traffic (fig. 2). The decision is made according to the method of cooperative games under uncertainty with limited resources [2].

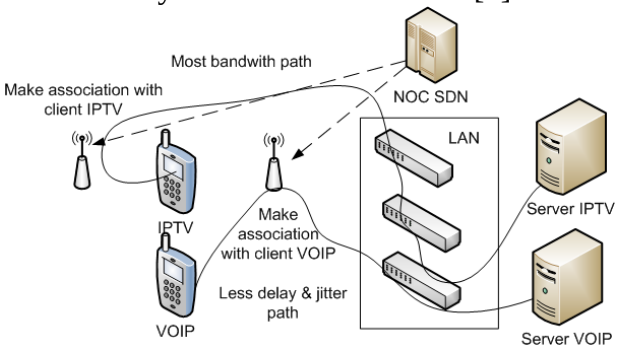

Fig. 2. Adaptive routing with dynamic AP association

The adaptive path solution requires some information about the current network load and state. In the article [3] the method of Shortest Span First (SSF) was developed to control and dynamically balance traffic flows. This method allows to choose routes according to the traffic features. The VOIP traffic is routed by paths with minimal delays, VOD is routed by minimal loaded paths. Wireless segments don't need additional tuning because they are controlled by the single SDN controller.

Our test bed consisted of some dual-band IEEE 802.11n APs and some wireless clients (notebooks with USB network card, tablets, smartphones). When we covered wireless network in a big room, all the clients tried to connect from the same point of the room. Then the clients began to send traffic and packet loss ratio was about 6%. When the number of connections reached the limit

value for the AP, the clients lost the service: any collision on the channel led to stopping the rest of the traffic for this channel in the AP's coverage area.

The transfer of control process depends on the sequence of state messages exchanged between the access points and the stations at the physical level.

For IEEE 802.11 networks that do not have an access control mechanism, there is little difference between a full-fledged new connection and the transfer of control. On the other hand, the delay of transfer of control will be greater than the delay of associating added to the delay of connecting to access points.

There is a problem with multicast routing and multicast tree expanding. When we have IPTV enabled and want to use OpenFlow, we have to calculate several trees to switching multicast traffic. The controller should consider the multicast load to calculate QoS parameters.

Adaptive routing principles

A network performance depends on the throughput of switches and routers handling the traffic within the network segment. Calculating a route with the use of standard protocols (OSPF, IS-IS, BGP) is not always effective for reducing the total delay when mobile wireless clients move. Quality of Service (QoS) requirements to network jitter and response time complicate the dynamic routing. Transfer of high-priority audio and video streams requires to reserve a bandwidth between specified pairs of nodes.

When some network nodes fail, the topology changes or the load increases greatly a routing protocol should reallocate any resources and bandwidth reserves. It is necessary to use routing methods which do not require a complete conversion of communication paths in a dynamically changing network structure. We used the network controller and storage for additional information concerning the network configuration to make the decision on the network topology.

To solve the problem of routing, the model of the network as a weighted connected directed graph $G = (V, L, W)$ is commonly used, where V is the set of N vertices, L is set of links between the vertices (arcs), W is the set of weights of arcs. Path is defined as a sequence of ports of network devices.

In this regard, the vertices of the graph G must associate with the ports used for network devices. Arcs correspond to the physical connections between the ports. The weight of an arc depends on the priority code point (PCP) in one of the queues associated with the port. Any queue is associated with the minimum bandwidth and the maximum delay for the port the queue at.

So the weight of each arc l_{ij} is determined as a dynamically changing vector consisting of three components: $W(l_{ij})=(B_{ij},b_{ij}(t),lat_{ij}(t))$, where B_{ij} is the maximum capacity of a network connection; $b_{ij}(t)$ is the residual capacity at the time t; $lat_{ij}(t)$ is the delay at the time t.

The information about the residual capacity and the value of the delay depend on the load of the communication link. The total delay for the entire route is calculated as the sum of delays along arcs, i.e. the delay is a metric parameter. At the same time, the capacity of the route is determined by the minimum bandwidth arc. Dijkstra's algorithm involves the repetition of the calculations for each change of network configuration.

For the case of a directed graph G a modification of the algorithm is required.

We denote the tree of shortest paths to all vertices of V from the initial vertex v_s as T_g. We select a set of arcs L_t included in the tree T_g and a set of other arcs L_r. Under certain conditions, the arc l_{ij} from L_r should be included in L_r instead of the arc l_{kp}. Such a replacement of the pair of arcs is called a transition and is denoted $l_{ij} \leftrightarrow l_{kp}$. The idea of the method of effective adaptive routing is based on the following fact. When the weight of an arc from the shortest-path tree changes or the weight of an arc involved in transition to the arc from the shortest-path tree, it is necessary to process the lists of the optimal routes and replace the arcs whose weight changed.

Co-existence multi-vendor hardware

Some managed access points with CAPWAP support can be reassigned with OpenFlow CAPWAP add-on without any changes in firmware. But many vendor-based controllers with CAPWAP support (Cisco, Meraki, Aruba, Juniper, Ubiquiti) use private keys and encryption to manage the access points. In a CAPWAP environment, a lightweight access point discovers a controller by using CAPWAP discovery mechanisms and then sends a CAPWAP join request to the controller. The controller sends a CAPWAP join response to the access point allowing to join the controller [4]. When the access point joins the controller, the controller manages its configuration, control transactions, and data transactions. We can use this discovery mechanism to get control. There are three methods of discovery, when we can intercept request: from neighbors (when all the APs are made by same vendor), from DHCP options 43 and 60 [5], from DNS request (for example, Cisco Air using CISCO-LWAPP-CONTROLLER.localdomain to DNS discovery).

By default in a CAPWAP environment, the APs will tunnel all of their traffic back to the controller and then out. We can use tunnel-free method to send traffic and process it on switches and the SDN controller.

For other APs, we can use asynchrony REST logging to get topology changes and to monitor client moving. When the controller processes the entire information it can calculate the optimal routing with any combination of REST and CAPWAP controllers and APs.

When the network consists of multi-vendor managed APs we need to consider all vendor-specific features of communications between APs and the controller. For example, Zyxel uses the modified CAPWAP protocol in his controller software, as well as Cisco controllers don`t support some APs of third-party vendors. Trapeze controller by Juniper is not compatible with any third-party APs. UniFi controllers use Ethernet-over-IP multicasting to discover their APs. But all the mentioned solutions are declared as CAPWAP-compatible by vendors. Therefore, we need to use specific modules to communicate with APs by different vendors.

The best solution will be to group devices by different vendors into isolated subnets or VLANs and to use unique modules to manage each subnet. When we use SDN it is possible to do it without any changes to physical scheme. The SDN controller forms isolated segments based on vendor-specific information or manual settings. All the specific requests are transmitted immediately to WLAN controllers. CAPWAP Split MAC technology [4] provide basic features to manage MAC layer (manage BSSID, SSID mask and others) and is implemented on all the CAPWAP-enabled devices by any vendor. The SDN controller can use information from tunnel-free CAPWAP traffic directly.

For executing a routing algorithm, the OpenFlow controller needs the information about wireless client placement on physical ports of switches. The controller doesn’t need the information about access points as intermediate devices. Each access point is viewed as a host and the requirements for traffic isolation to their MAC address are obtained from external sources (wireless network support modules).

Implementation

The wireless network control module can be implemented separately from OpenFlow modules but must be controlled through web to unify the access. We use Floodlight as an OpenFlow controller and well documented Floodlight REST API to control OpenFlow entries and paths. The wireless module uses “Static Flow Pusher” API to create isolated subnets associated with different vendors.

The CAPWAP protocol defines two modes of operation: Split and Local MAC. There is no stable Split MAC implementation in open source products. We use the unchanged version of openCAPWAP controller (AC).

We use three external controllers to create the test bed: Cisco vWLC 7.5, Juniper WLC800 and UniFI controller. For Cisco WLC controller we can use REST and SSH CLI [5]. But REST for Cisco WLC is not well documented, unlike CLI commands. We use RSH tool and certificate-based authentication for interaction with this controller. For UniFi controller [6] and Juniper controller [7] there are REST API examples of basic control tasks. We can use JSON and REST together to work directly with the controller. Our modules can be used separately from the control system since we

don`t use sockets, internal software connections and other proprietary methods of interaction. Any new module can be added to the controller without changes except a wireless control module.

When an administrator wants to create a managed domain they should select a vendor from the list, define where APs and the controller are located. If the administrator doesn't know where APs are located they can use OpenFlow filters [8] to define AP MAC.

Such virtual isolation in comparison with traditional VLANs or VXLANs uses only some flows to create an isolated subnet while all other traffic is transmitted without limitation.

Our wireless control module is used only to change a configuration or get the information on the current state from controllers. All the controllers work independently from this module. For interacting with a system module by REST our module puts a request to the required controller and gets a response to interpret it. If the OpenFlow controller fails, the wireless network segment will continue to work separately from the wired network segment. Static entities in isolated subnets are independent from the OpenFlow controller.

When we use traffic-specific routing for different traffic types users can connect to different access points in the same area. The wireless control module has the entire information from all the access points by different vendors and selects the best AP for the preferred traffic type. For example, a client using VoIP would be re-associated with the AP having the least number of clients or the least delay in wired segment. Dynamic tracking of user traffic allows to avoid interference of traffic with opposite QoS requirements on the same AP (for example, VoIP and video casting). The adaptive routing module is used to calculate the optimal path between any source and destination with QoS prerequisites.

We don't implement an adaptive routing module as Floodlight module because it can reduce flexibility of this scheme. We plan to develop such a module in java for processing wireless traffic on Northbridge level of the controller to reach better performance and less delay in installing a new flow into the OpenFlow tables. Then we plan to exclude some controllers and reassign APs to the CAPWAP controller, but neither method of CAPWAP discovery can associate APs with our controller.

All the vendor-locked controllers use proprietary add-ons to CAPWAP protocol. APs by any vendors can be controlled stably through the REST protocol. If there are no CAPWAP APs in the network we don't need to use the CAPWAP controller.

A list of possible control and monitoring features depends on the controller capabilities by REST protocol or CLI. We have implemented AP channel monitoring, client association and dynamic re-association with a preferred AP. All the authentication features are implemented through vendor-based controllers.

Conclusion

So, the SDN technology can be applied not only in wired networks. The best practice is to use SDN for managing vendor-specific traffic locally and wireless traffic globally. Today most vendors develop SDN products. But, none of the wireless products use SDN as the basic technology, so the above-described approach can soon be used in networks.

The combination of CAPWAP, traditional vendor-specific managing methods to manage access points and wired SDN infrastructure allows to unify the management of the entire network. The REST technology for module interconnections helps to expand this managing method to any wireless controllers. Using adaptive routing can reduce the delays and the infrastructure load without any topology changes.

Acknowledgements

The research work was supported by the Russian Foundation for Basic Research (grants 13-01-97050, 14-07-97034, and 15-07-06071), the President of the Russian Federation (grant for young scientists SP-2179.2015.5) and the Foundation for Assistance of Development of Small Businesses in the Field of Science and Technology (grant UMNIK 2628GU1/2014)

References

[1] Suresh L., Schulz-Zander J., Merz R., Feldmann A., Vazao T. Towards Programmable Enterprise WLANs with Odin. HotSDN '12, Aug 2012.

[2] Vlatskaya I.V., Nesterenko M.Yu., Polezhaev P.N. Decision support models under uncertainty which reduced to cooperative games. Vestnik of Orenburg state university, 2012.

[3] Kim W., Sharma P., Lee J., Banerjee S., Tourrilhes J., etc. Automated and Scalable QoS Control for Network Convergence. INM/WREN '10, 2010.

[4] Bernaschi M., Cacace F., Iannello G., Velucci M., Vollero V. OpenCAPWAP: An Open Source CAPWAP Implementation for the Management and Configuration of WiFi Hot-Spots. Computer Networks, 53, p. 213-230, 2009.

[5] Cisco Wireless Controller Configuration Guide, Release 7.4. http://www.cisco.com/c/en/us/td/docs/wireless/controller/74/configuration/guides/consolidated/b_cg74_CONSOLIDATED.pdf. 2013.

[6] Borg J. UniFi API. https://github.com/calmh/unifi-api/ README.md. 2014.

[7] Juniper WLC Protocol Guide. https://github.com/Juniper/py-jnpr-wlc/blob/master/docs/wlc_protocol_guide.md. 2013

[8] Kanui B., Parraga J. Floodlight REST API. http://www.openflowhub.org/display/floodlightcontroller/Floodlight+REST+API. 2014.

[9] Peterfi J. Wrapper for Hostapd. https://github.com/ iosifpeterfi/openCAPWAP-OpenWRT/blob/master/hostapd_wrapper/ README.rst. 2013.