Enabling Secure Mobility with OpenFlow

Suneth Namal*, Ijaz Ahmad†, Andrei Gurtov‡ and Mika Ylianttila§
*†§Department of Communications, University of Oulu, Finland
‡Department of Computer Science and Engineering, Aalto University, Finland
Email: [*namal,†iahmad]@ee.oulu.fi,‡gurtov@hiit.fi,§mika.ylianttila@oulu.fi

***Abstract*—Software Defined Networking (SDN) and its one possible realization, OpenFlow, define the trends of future networks. However, the present OpenFlow architecture does not allow the switches to be mobile e.g., in a moving train as it would disrupt flow processing from network switches. We present OFHIP, an architecture that enables OpenFlow switches to change their IP addresses securely during mobility. OFHIP employs IPSec encapsulated security payload (ESP) in transport mode for protection against DoS, data origin authenticity, connectionless integrity, anti-replay protection, and limited traffic flow confidentiality. We demonstrate the benefits of OFHIP compared to present use of SSL in enabling mobility, reducing the connection latency and improving the resilience to known TCP-level attacks.**

I. Introduction

In SDN architecture, the control layer is logically centralized to a software-based controller which maintains the global view of the network. The controller allows the flexibility to configure, manage, secure, and to optimize network resources via automated and dynamic software programs. OpenFlow is one of the realizations of SDN that provides freedom to exploit new opportunities over the existing networking infrastructure. OpenFlow at the current stage does not support mobility [1]. This is a critical downfall that limits OpenFlow into fixed core-networks, clusters or data centers.

Robust multipath connectivity for fault tolerance is one of the critical issue inhabits in the current SDN architecture. This is even more crucial when the controller or the switch is mobile. In one hand, multiple channels between the switch and the controller can guarantee a certain level of assurance for seamless connectivity. On the other hand, multipath connectivity inspired by the efficient use of installed bandwidth and increased robustness by simultaneous use of potentially diversified paths that result to optimally utilize the network resources and increase redundancy.

Mobile base stations, mobile routers, switches, mobile clouds and server migration are some of the compelling reasons why mobility is insisted in software defined networks [2]. This would extend SDN benefits into mobile environments e.g., moving trains, buses, flights and other automobiles. The existing competing solutions with Mobile IP (MIP) still have problems related to "triangle routing" and drop of IP packets due to frequent handover when the host is away from Home Agent (HA). Mobility offers many dazzling opportunities that also bring with them some profound challenges related to security and privacy [3]. We argue that mobile IP has limitations against DoS, passive eavesdropping, insider attack, replay attack, tunnel spoofing and location privacy [4], [5]. Security in IP based networks is widely tackled with a common set of protocols composed of secure file transfer protocol (SFTP), secure socket layer (SSL), and transport layer security (TLS). OpenFlow enables an acceptable level of security with SSL or TLS though, it does not support mobility [6]. These limitations insist modifications to the current OpenFlow architecture. More specifically, below we describe the major problems of present OpenFlow version.

- **Flow processing :** Change of address would disrupt flow processing from network switches. Therefore, they require fast and regular updates to flow tables.
- **Secure session management :** Changing an IP address may also tear down active SSL/TCP sessions.
- **Secure handover :** Problem of mutual authentication and cannot support mobility and certificate exchange is not preferable for fast moving OpenFlow clients.
- **Flow rule management :** Change of IP address to solve latter issue causes additional overhead, since flow rules must be updated frequently.

This paper proposes a novel approach to handle OpenFlow based mobility with global identities introduced by the host identity protocol (HIP) layer. These identifiers are of the same format as IPv6 addresses. Furthermore, IPv6 is already proposed for the next version of OpenFlow and thus, applications could be easily adopted. In general, OFHIP addresses the security bottleneck of current OpenFlow version.

The rest of this paper is structured as follow. Section II, presents OpenFlow based connection establishment. Section III describes the proposed mobility architecture and scenario. Next, we present our implementation and results in Section IV. In Section V, we discuss the future research directions and some of the important findings. Finally, in Section VI, we conclude this paper.

II. OpenFlow based connection establishment

The present use of SSL over TCP introduces new security threats to known TCP-level attacks. There are several serious security flaws inherent in the protocol, regardless of the correctness of any implementation (e.g. SYN attacks, reset attack, sequence prediction attack, ICMP attacks and DoS attacks). The OpenFlow switches and routers connect to the remote control processor, namely the controller which handles the TCP requests. Therefore, an attacker will first attempt to break into the controller. Thus, any security weakness in the controller will allow attackers to penetrate into the network.

Though, OpenFlow specification mentions the applicability of user datagram protocol (UDP) with datagram transport layer security (DTLS), there are no detailed discussions of how it could be used. Certain TLS implementations do not currently check client certificates. In case, if a similar approach is used with OpenFlow, it could be vulnerable to any kind of man-in-the-middle attack. The specification does not either provide the certificate format or indicate what fields in the certificate are used for naming. OpenFlow refers the use of "TLS" without specifying a reference or a version number. This would result in non-interoperable implementations if different OpenFlow implementations use different versions.

III. Proposed Mobile Architecture and Scenario

In Table I, we have compared the HIP layer options with the existing SSL/TLS solution. In a nutshell, OFHIP solution is an integration of diet version of HIPv1 [7] layer with the existing OpenFlow version to replace the SSL/TLS based mutual authentication. As a result, we would get almost the same level of security with enhanced mobility support. It provides end-to-end encryption, mutual authentication and secure key exchange. The HIP layer identifies a host either by a host identifier (HI) or a host identity tag (HIT) defined in [7]. HI is the public key of an asymmetric key-pair which could be used as a local identity. However, it is not suitable to serve as a packet identifier, since the length can vary [8].

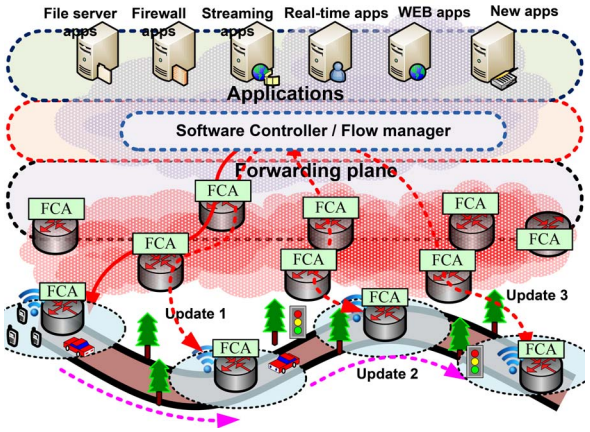

Fig. 1. SDN based mobile scenario.

In Fig. 1, we present SDN based mobility scenario where the switches or routers are configured on the fly. The flow control agents (FCA) on the mobile switches and routers are responsible for updating the software controller of the new location information for location based services. Besides that, the flow rules could be built on top the newly introduced cryptographic global identifiers which do not change over time. Thus, updating the flow rules due to mobility is no longer required. This would reduce the processing and control traffic overhead due to dynamic address configuration and thus, flow processing. Furthermore, it reduces overhead in the policy charging and rules function (PCRF) and improve the flexibility with QoS management and network configurations. Moreover, controller can keep records of HITs and authenticate them through controller signed ceertificates or using other techniques, such as DNSSEC and DHT-based verification.

The existing Mobile IP (MIP) solutions address the same problem in a different manner with the home address (HA) which always identifies the mobile node. The care-of address (CoA) associates the mobile node with its home address by providing information about the mobile node's current point of attachment. Mobile IP uses a registration mechanism to register the CoA with the HA. This process consumes a considerable amount of time as it is shown in [2]. Thus, we have proposed HIP layer "UPDATE" procedure to inform the peers of the new addresses. Fig. 2 presents our OFHIP solution which addresses mobility in OpenFlow.

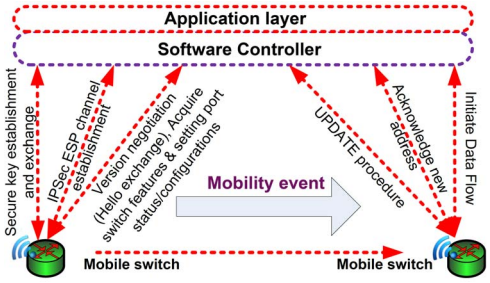

Fig. 2. OFHIP solution for secure connection establishment.

Mutual authentication in OpenFlow is a part of security assertion which is achieved with SSL or TLS. They operate at the transport layer in the OSI stack, and provide secured data transport for applications. It supports peer negotiation for algorithm selection, public key based exchange of secret session keys and X.509 certificates. However, they do not support mobility alone without the support of an underlying protocol (UDP, SCTP and etc). The modified version of TLS: DTLS can be used by applications directly or by tunneling to provide secure mobility [9]. However, with legacy IP protocols, the handover impact at the upper layers is still tight.

OFHIP uses IPSec encapsulating security payload (ESP) secure associations (SAs) that are bounded to HITs. Therefore, address reconfiguration would not have any impact on the higher-layer associations except the changes in network routing layer. The key-exchange in OFHIP is a cryptographic protocol that uses a randomly generated key encrypted by a Diffie-Hellman derived key in order to establish a pair of IPsec-ESP SAs between two entities: the initiator and the responder. The HIP layer in OFHIP solution maps arriving ESP packets to a HIT using the security parameter index (SPI) value in the packet and selects the source address and interface according to the SPI value set by ESP. After handover, data continues to flow inside of the ESP tunnel with the same SPI values but with a different IP address at the mobile node.

The initiator defines SPI value of the responder's outgoing SA, whereas the responder defines the SPI value of the initiators outgoing SA. To prevent replay attacks, we propose to use an incremental counter with a hash of the HIT. For secure mobility, rekeying may be necessary. Thus, new SAs

must be created by removing the old SAs. Once a host receives data on new SA, it can safely remove the old SA. The HIP layer uses AES-CBC for symmetric encryption and to provide CMAC for MACing functions while the session keys are encrypted with elliptic curve diffie-hellman (ECDH) keys.

The four-packet exchange makes OFHIP resilient to denial-of-service (DoS). The protocol transmits an EC Diffie-Hellman encrypted key in the 3^{rd} and 4^{th} packets, and authenticates the parties with those packets. The responder starts a puzzle exchange with the initiator in the 2^{nd} packet, and completes it in the 3^{rd} packet before the responder stores any state from the exchange. This model falls in the line of TLS and fairly equivalent to 802.11 master and pair-wise transient key, but handled in a single exchange.

IV. Implementation and Results

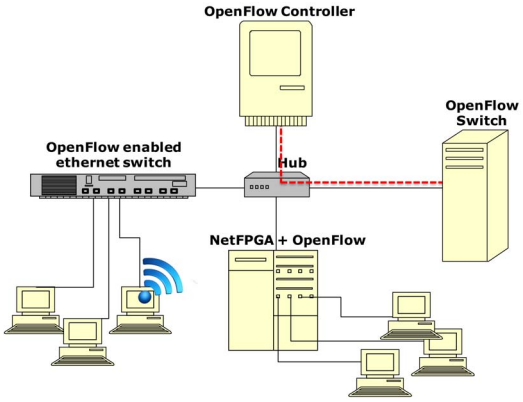

Fig. 3. OpenFlow experimental setup.

In a nutshell, the OFHIP architecture is an attempt to introduce secure mobility with OpenFlow. In this work, we have developed the OpenFlow test-bed presented in Fig. 3. The test-bed consists of two DELL PowerEdge-R320 servers, a DELL PowerEdge-2900 server, a HP-6600 Ethernet switch and two laptops. The PowerEdge-2900 server is installed a "NetFPGA open platform for gigabit network" card developed by Stanford university [10]. This platform has four physical gigabit interfaces, internal memory, user defined network processing logic, and field programmable gate array (FPGA) that allows designing of routers and other networking hardware. We have installed CentOS kernel 2.6.18 to support the NetFPGA platform. This platform is loaded a compiled OpenFlow bit-file. The HP-6600 Ethernet switch is configured with the latest OpenFlow-enabled software version K.15.06.5080 [1].

We have used a laptop with an i5 CPU of 2.67GHz as the controller and another laptop of CPU 2.16GHz as a switch (both running on Linux kernel 2.6.35). The OpenFlow version 1.1.0 defines both non-SSL and SSL support that must be configured at the built. As it is presented in Fig 1, the secure connection establishment in OpenFlow has several phases. The HIP layer in OFHIP is responsible for mutual authentication and secure key exchange. However, this solution does not literally change the flow of OpenFlow based connection establishment. The HIP layer replaces the SSL based mutual authentication and connection establishment in OpenFlow.

The OpenFlow "HELLO" messages include the version, type, length and transaction-ID associated with the packets. The "Feature" request/reply messages check and agree on the switch/controller compatibility upon establishment of an OpenFlow channel. Followed by it, the SET-CONFIG messages could be used by the controller to set configuration parameters appropriately. In case, if it happens to close the connection due to version incompatibility or any other issue between the switch and the controller, the "CLOSE" message type in the HIP layer could be used.

First, we have investigated the options for OpenFlow based connection establishment between the switch and the controller. Fig. 4 presents both secure and insecure connection establishment with SSL and non-SSL options. In order to use SSL with OpenFlow, it is required to set-up a public-key infrastructure (PKI) which includes a pair of certificate authorities (CAs) for the controller and the switch. We have used a script to generate the PKI. Thereby, we have established the private keys and certificate authorities' certificates for the switch and the controller, and root certificates for their CAs.

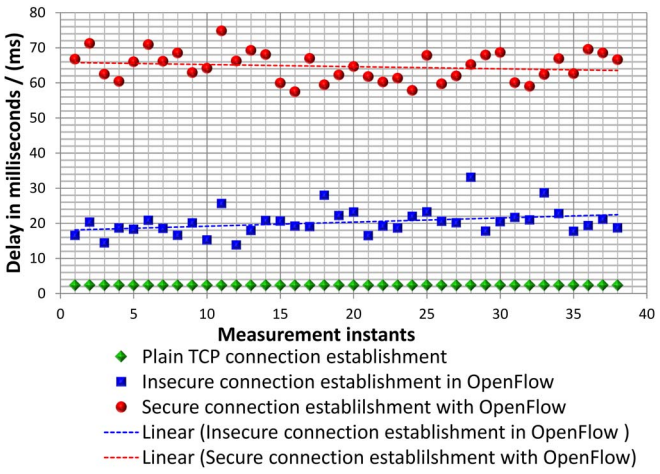

Fig. 4. Plain TCP and secure and insecure connection establishment delay with OpenFlow.

As it is depicted in Fig. 4, SSL exchange consumes a considerable amount of time compared to insecure connection establishment. The averaged delay for connection establishment with these options are measured to be 66 ms and 20 ms respectively. According to the results, the delay with SSL certificate option is almost as three times of the non-SSL option. The graph also presents linear regression of results. Plain TCP connection establishment phase is measured to be 2-3ms which could be almost neglected compared to that of the SSL-certificate exchange. The SSL based handshake in OpenFlow uses X.509 standard certificate type with RSA. Thus, this exchange is heavy and consumes relatively a long time compared to insecure session establishment over plain TCP. However, mobility as an essence in modern communi-

TABLE I
COMPARISON OF SECURITY AND MOBILITY FEATURES OF SSL/TLS, HIP-BEXV1, AND HIP-DEX.

	SSL/TLS	HIP-BEXv1	HIP-DEX
Mobility extendibility	Low (leverage mobility protocol)	High (Multihome extension)	High (Multihome extension)
Mutual authentication	High (SSL handshake protocol)	High (With four ways handshake)	High
Key exchange and encryption	High (RSA & CCM-AES)	High (Diffie-Hellman & RSA/DSA)	High (ECDH and AES encryption)
Signaling integrity	Medium (DSA)	High (HMAC - use HIP-gl or HIP-lg)	Low (external whitelisting)
Data integrity	Medium (symmetric-DES)	High (ESP-CBC)	High (AES-CCM)
Message authentication	HMAC (SHA-256)	HMAC (with SHA-1 or MD5)	CMAC
Replay attack resistance	High (with 3rd party)	High (with 3rd party)	High (with 3rd party)
Non-repudiation protection	Low (Standard TLS)	High (self certified)	High (self certified)
Man-in-the-Middle protection	Low	High (non-opportunistic)	Medium
Denial of Service attack	Medium	High (by packet design)	High (by packet design)
Forward secrecy	Low	High (with legacy applications)	Low (KeyWrap of random-keys)

cation does not perform well with connection oriented TCP. Therefore, these solutions that are already in OpenFlow would not meet the demanding high bandwidth and reduced latency.

In Fig. 5, we compare the delay in connection establishment with OpenFlow and OFHIP. As it is depicted from Fig. 5, IPSec connection establishment with OFHIP consumes around 44 ms whereas secure connection establishment with OpenFlow is about 66 ms. OFHIP delay consists of the delays associated with the wireless interface which encounters the wireless propagation delay and the 802.11 processing delay. These results here depict that the delay in secure connection establishment with OFHIP is almost close to that of the insecure connection establishment with OpenFlow which is measured through the wired interface. DTLS is proposed to

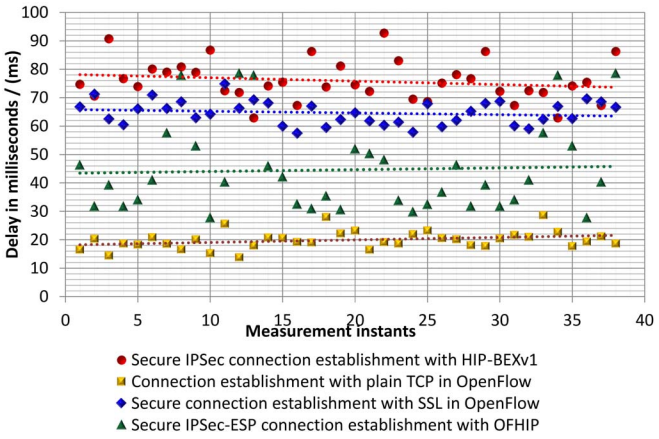

Fig. 5. Comparison of OFHIP with OpenFlow.

fill-up the mobility gap of OpenFlow. In one hand, it is deliberately designed to be similar to TLS as possible [9]. On the other hand, TLS cannot be directly used with datagram due to loss or reordering of packets. However, DTLS has the minimal changes to TLS to fix this problem. Since, DTLS is almost identical to TLS, we can expect same level of performance as it is with SSL/TLS. As a result, we believe the OFHIP solution has the high potential in terms of mobility, since the presented results are far below the requirements of widely used mobile applications. Next, we have measured the throughput characteristics of OFHIP. In order to analyze traffic characteristics, we install Jperf in the controller and the switch. Jperf is a graphical interface developed for Iperf. It allows easy configuration of parameters. Iperf is a testing tool which can create TCP and UDP traffic flows and measure throughput characteristics based on client/server mode [11]. We have investigated both TCP and UDP throughput of OFHIP. Fig. 6 presents throughput behavior of OpenFlow based plain TCP and secure OFHIP based ESP-IPSec channel over a 2Mbps limited bandwidth channel.

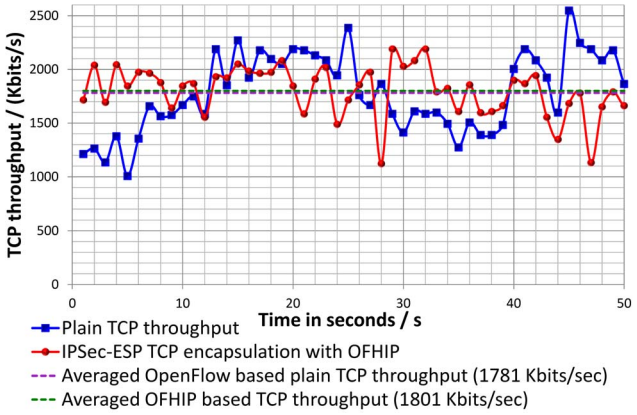

Fig. 6. TCP throughput (window size 83.5KBytes)

.

We have used local scope identifiers (LSIs) with Jperf for local communication in OFHIP. Therefore, sessions do not terminate even after IP renewal. However, the observations have verified that throughput drops to zero during handover since TCP is connection oriented. Thus, it is clear that TCP data transmission using ESP encapsulation over OFHIP cannot be used with mobile switches though; it has almost the same throughput as plain TCP in a fixed network.

In Fig 7, we have compared the plain UDP throughput and IP/ESP encapsulated UDP throughput of OFHIP. Using Ipref, the wireless bandwidth is limited to 1 Mbps. We have investigated the throughput characteristics of mobility events with a script that automatically changes the IP address of the device. The HIP layer handles address update and informs the recipient of the new address. After acquiring a new address,

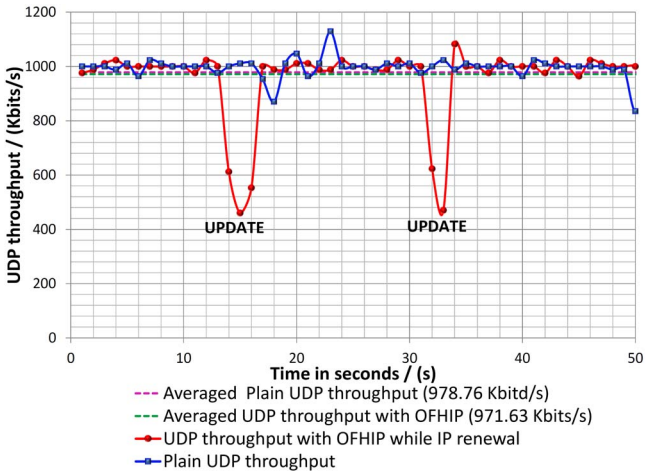

Fig. 7. UDP throughput (MTU-1470 byte, Buffer-160KBytes).

UDP traffic is redirected according to the address update procedure implemented in the HIP layer.

Therefore, throughput remains almost stable even with the mobility events which we have scripted. Fig. 7 illustrates two such instants where the throughput suddenly drops due to the scripted mobility events. The measured UDP encapsulated IPsec ESP throughput is almost similar to the unencrypted UDP throughput. Thus, OFHIP uses UDP encapsulation of IP traffic with mobile applications. It must be stated here that the actual packet loss during handover is the time of HIP layer "UPDATE" procedure call. Hence, multi-homing is a promising solution to avoid packet-loss during handover. Moreover, handover buffers are capable of avoiding any such packet loss that could be happening during handover. Since, HIP layer uses a separate identity other than the IP address, it can also enable multi-homing as a counterpart.

V. Discussion

With OFHIP, authentication is much faster compared to legacy cryptographic protocols due to elliptic curve cryptography (ECC) based key generation that offers the same level of security with relatively small key sizes. Despite that, the flow-tables with the new cryptographic identifiers can simplify mobility management, since it is not necessary to update the flow-tables when the flows are built on top of those identifiers.Therefore, the matching rules can be made globally unique and available for global mobility management. Furthermore, it allows redefining data-path identifiers by using these cryptographic identifiers. Moreover, we can prove concatenation of two globally unique identifiers can again produce unique identity which does not collide with another identifier. Thus, we propose that it could be passed through a hash function to generate the 64-bit cryptographic data-path identifiers. Otherwise, 128 bits lengthy identifiers could be used for global level flow processing and identification.

Commercialization of OpenFlow requires guaranteed connectivity between the switch and the controller which could be realized with robust multipath connectivity. Equal Cost Multipath (ECMP) or BGP Multi-Exit Discriminators (MEDs) addresses this problem though they are lacking the required flexibility and scalability. The HIP layer in OFHIP enables multihoming and thus allows to configure multiple identities. Inspired by multihoming, OFHIP could be an enabler for multipath connectivity to the OpenFlow controller. Ultimately, it will end-up utilizing the network resources optimally and enhance data-rate besides the improved capability for fault tolerance and redundancy management.

VI. Conclusion

This paper is an attempt to introduce mobility into legacy OpenFlow based switches and to improve resilience against the known TCP-level attacks. OFHIP introduces a cryptographic name space which is identical to the IPv6 address space. They are cryptographically globally unique and ideal for developing mobile applications, since they do not change with mobility events. The results depict that the handover latency in OFHIP is well below the limits of delay requirements of mobile applications. In comparison of the number of SAs, OpenFlow supports up to 15 secure associations per second, whereas OFHIP supports upto 22 secure associations per second. This is an improvement by 147% compared to legacy SSL connection latency. We have shown that OFHIP has better throughput compared to TCP backhaul throughput and stands very well against the scripted mobility events with UDP encapsulated IPSec-ESP. In a nutshell, the results prove benefits of OFHIP compared to present use of SSL in terms of mobility, reduced connection latency and improved security.

References

[1] ONF. (2013) Open Networking Foundation. [Online]. Available: https://www.opennetworking.org/

[2] S. Namal, M. Liyanage, and A. Gurtov, "Realization of Mobile Femtocells: Operational and Protocol Requirements," *Wireless Personal Communications*, pp. 1–26, 2012.

[3] S. Namal, J. Pellikka, and A. Gurtov, "Secure and multihomed vehicular emtocells," in *75th Vehicular Technology Conference (VTC Spring)*. IEEE, 2012, pp. 1–5.

[4] 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.

[5] S. Namal, K. Georgantas, and A. Gurtov, "Lightweight authentication and key management on 802.11 with Elliptic Curve Cryptography," *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*, pp. 1830–1835, 2013.

[6] B. Boughzala, R. Ben Ali, M. Lemay, Y. Lemieux, and O. Cherkaoui, "OpenFlow supporting inter-domain virtual machine migration," in *8th International Conference on Wireless and Optical Communications Networks (WOCN)*. IEEE, 2011, pp. 1–7.

[7] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson, "RFC 5201: Host Identity Protocol," *Network Working Group*, 2008.

[8] A. Gurtov, "Host Identity Protocol (HIP): Towards the Secure Mobile Internet," *Wiley Publishing*, 2008.

[9] E. Rescorla and N. Modadugu, "RFC 4347: Datagram transport layer security," 2006.

[10] J. Naous, D. Erickson, G. A. Covington, G. Appenzeller, and N. McKeown, "Implementing an OpenFlow switch on the NetFPGA platform," in *Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*. ACM, 2008, pp. 1–9.

[11] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "Iperf: The TCP/UDP bandwidth measurement tool," *http://dast.nlanr.net/Projects*, 2005.