SecuSpot: Toward Cloud-Assisted Secure Multi-Tenant WiFi HotSpot Infrastructures

Julius Schulz-Zander
Fraunhofer HHI, Germany
TU Berlin, Germany
julius@inet.tu-berlin.de

Raphael Lisicki
TU Berlin,
Germany
raphael@ralisi.de

Stefan Schmid
Aalborg University, Denmark
TU Berlin, Germany
schmiste@cs.aau.dk

Anja Feldmann
TU Berlin,
Germany
anja@inet.tu-berlin.de

ABSTRACT

Despite the increasing popularity of WiFi networks and the trend toward automated offloading of cellular traffic to WiFi (e.g., HotSpot 2.0), today's WiFi networks still provide a very poor *actual* coverage: a WiFi equipped device can typically connect to the Internet only through a very small fraction of the "available" access points. Accordingly, there is an enormous potential for multi-tenant WiFi hotspot architectures, which however also introduce more stringent requirements in terms of scalability and security. The latter is particularly critical, as HotSpots are often deployed in untrusted environments, *e.g.*, physically accessible Access Points deployed in the user's premises (e.g., FON) or cafes.

This paper proposes a Cloud-assisted multi-tenant and secure WiFi HotSpot infrastructure, called $SecuSpot$. SecuSpot is based on a modular access point and features interesting deployment flexibilities. These flexibilities can be exploited, e.g., to move security critical functions to the Cloud, and hence prevent eavesdropping even when deployed across *untrusted* Access Points. At the heart of SecuSpot lies a novel programmable wireless switch, the *wSwitch*. The *wSwitch* allows to (de-)multiplex the different tenants already on the HotSpot and to decouple essential security functions (association, authentication, and cryptography).

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network management; C.2.1 [**Network Architecture and Design**]: Wireless Communication

Keywords

Software-Defined Networking; Network Function Virtualization; IEEE 802.11; WiFi; Cloud; Wireless; Security

1. INTRODUCTION

Practically all portable end-devices today are WiFi enabled, and with the advent of the Internet-of-Things networks, WiFi is likely to extend to even more objects in the near future. WiFi is also increasingly attractive for mobile operators which aim to offload media traffic from cellular networks [20]. Several mobile operators plan massive WiFi hotspot deployments as part of the HotSpot 2.0 initiative: in a HotSpot 2.0 network, mobile devices automatically join a WiFi subscriber service whenever the user enters a Hotspot 2.0 area, to provide better bandwidth and services-on-demand to end-users and relieve carrier infrastructure of some traffic.

CAN'16, December 12 2016, Irvine, CA, USA

ISBN 978-1-4503-4673-3/16/12…$15.00

DOI: http://dx.doi.org/10.1145/3010079.3012015

1.1 The Problem and the Potential

However, while the popularity of WiFi is increasing rapidly, there exists a huge *demand-supply mismatch*: while there are billions of WiFi equipped devices and hundreds of millions of access points, today, a device can only get online through a very small percentage of these access points.

Accordingly, we see an enormous potential for virtualized WiFi networks supporting *multi-tenancy*: by virtualizing the network and opening the spare capacities, the *actually useful* WiFi coverage can be increased significantly. Indeed, we currently witness several initiatives to exploit these untapped resources, e.g., FON/WLAN-TO-GO services are offered to millions of users today.

1.2 The Challenge

The vision of multi-tenant WiFi networks as well as the resulting more stringent security requirements stand in stark contrast to today's inflexible WiFi architecture.

The first important challenge in multi-tenant WiFi networks regards efficiency: the wireless hop is known to be critical for the overall network performance, and can contribute non-negligible delay and jitter especially for high definition media [10, 17]. In particular, it should not only be simple to provide tenants access to a given hotspot infrastructure, but also to efficiently multiplex and demultiplex the tenant traffic: ideally, the traffic should be sent toward the tenant directly from the hotspot where it arrives. Moreover, it should be possible to decouple and (service-)differentiate between control and data plane traffic, and *to route the two traffic types along different paths*.

Obviously, security is another important dimension, especially in multi-tenant systems, but also in general. HotSpots are usually either deployed in *untrusted* environments such as in the user's premises (e.g., homes) or in publicly accessible locations, or they are deployed on *trusted* access points in secure places, *i.e.*, not physically accessible by the end-user. Most of today's WiFi HotSpots only provide *open* authentication schemes based on *capture pages* without encryption of the WiFi link. Only a few pro-

vide a secured wireless access based on 802.11u and the *HotSpot 2.0* initiative, *i.e.*, a client is authenticated based on the phone's SIM card. However, supporting encryption on the WiFi link requires a key to be installed on the physical access point. Furthermore, the predominant approach in enterprise environments is to tunnel (*e.g.*, via Generic Routing Encapsulation (GRE), Aruba's proprietary PAPI [2] (AP control and management) protocol, the standardized Control And Provisioning of Wireless Access Points (CAPWAP) [13] protocol, or Lightweight Access Point Protocol (LWAPP) [8]) all decrypted 802.11 traffic back to a centralized WiFi controller to provide features such as seamless mobility and service differentiation. Some WiFi architectures such as Meraki [4] or AeroHive [1] place more logic into the AP. However, in the typical Split-MAC architecture the encryption keys usually still reside on the physical access point. Thus, unfortunately, most of today's WiFi HotSpot architectures lack standard security mechanisms, and do not provide any means to achieve security in untrusted environments. This opens a host of vulnerabilities including, e.g., man-in-the-middle-attacks [7]. *In particular, most of today's hotspot architectures are subject to eavesdropping.* Clearly, in multi-tenant settings, these security issues are becoming even more critical.

1.3 Our Contributions

This paper presents SecuSpot, a first approach to realize secure Cloud-assisted multi-tenant WiFi HotSpot infrastructures. In particular, SecuSpot allows to outsource the control over the WiFi to the *Cloud*, while the data plane traffic can be forwarded along completely different paths. For example, encrypted wireless traffic can be routed directly at the WiFi AP to the operator's network accordingly. Our approach does not require encryption keys to be stored in the physical AP. Rather the keys can be stored securely in the Cloud of the respective tenant. In principle, SecuSpot removes all security critical functions from the Access Point.

At the heart of the SecuSpot architecture lies a novel programmable WiFi switch which may be of independent interest: the *wSwitch*. The *wSwitch* is deployed on the WiFi Hotspot and introduces flexibilities in how tenants are multiplexed and demultiplexed across the given infrastructure, as well as in how and where security critical information (such as keys) is stored. In particular, the *wSwitch* multiplexes clients based on (B)SSIDs (determined dynamically using a *probe request protocol*), and allows to decouple both the tenants as well as the control and data plane traffic already on the Access Point (*allowing to overcome routing inefficiencies*), without the need to store security critical key or perform expensive cryptographic operations on the Access Point (*improving security and performance*). Thus, in some sense, our approach can be seen as the wireless answer to FlowVisor [18], which performs slicing based on the packet header.

1.4 Organization

The remainder of this paper is organized as follows. Section 2 presents our architecture more generally, and Section 3 describes the *wSwitch* lying at the heart of our approach. Section 5 presents and evaluates different use cases. After reviewing related work in Section 6, we conclude our work in Section 7.

2. THE BIG PICTURE

One simple but static solution to enable multi-tenancy is to have a virtual wireless access point interfacing with unique SSIDs, *one for each tenant*. Depending on the SSID, the traffic is forwarded to the respective tenant-controller. This solution however comes with severe limitations: in this setting, encryption needs to be performed on the access point: potentially a vulnerable location, where storing keys is undesirable. Moreover, without a common interface, data and control traffic inherently needs to be forwarded along the same, statically pre-configured path: the tenant cannot reroute the traffic to a different place. This can result in a high load on the control plane. More importantly, it makes a more flexible network operation impossible, such as the dynamic up and down scaling of control resources.

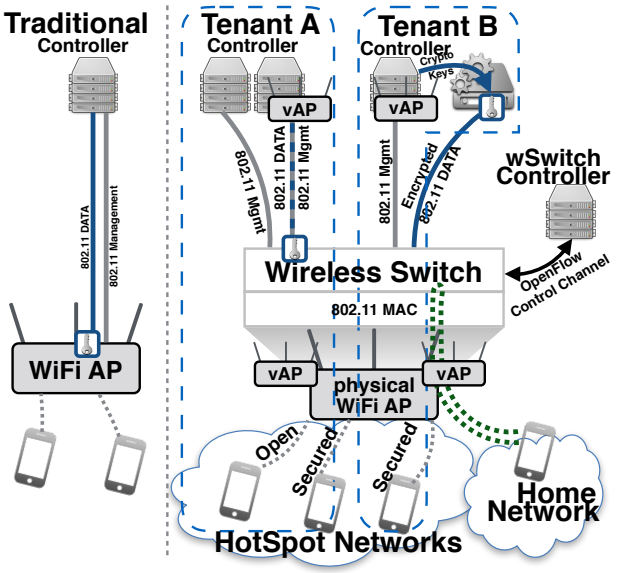

Figure 1: The *Wireless Switch*: The *wSwitch* supports matching on the BSSID, SSID, IEEE 802.11 Type and Subtype. This facilitates forwarding of flows to the respective tenant, *e.g.*, all traffic which is directed to a particular access point identified by a unique BSSID. The *wSwitch* allows to forward all 802.11 management frames to a remote location (in particular, the Cloud) while data can be forwarded to another network entity/middlebox, *e.g.*, a secured network function handling en/decryption of traffic or an Internet gateway. In a traditional setup, e.g., based on CAPWAP, all traffic is usually sent to just one location. A virtual AP (vAP) can also run directly on the AP, *e.g.*, in a home network scenario.

At the heart of our more flexible approach lies a programmable wireless switch, henceforth simply called the *wSwitch* (see Figure 1). In the spirit of OpenFlow switches, the *wSwitch* is based on a *match-action paradigm*. However, the paradigm is adapted to meet the whistles and bells of the WiFi: the programmable wireless switch combines matching on 802.11 frames and OpenFlow headers with actions for forwarding such as tunneling/encapsulation. For example, *wSwitch* allows to forward all 802.11 management frames to one or more remote locations while data can be forwarded to another network entity/middlebox. For instance, if a client performs an active scan without specifying the network name (a certain SSID) the packets are duplicated by the wireless switch and are forwarded to all tenants. On the other hand, if the SSID or BSSID (MAC address of the virtual AP) is specified, only the tenant who hosts the virtual AP will receive the packets.

With these concepts in mind, we are now ready to present our solution in detail. Our virtual APs can be configured to either only handle the management or also the datapath. Accordingly, this allows to separate the control and the data plane. For instance, the encrypted data frames can either be fully handled in the Cloud, or in the data plane using network processors (i.e., en/decryption in hardware). Specifically, the Cloud-assisted architecture reduces network traffic and hence allows to scale the system.

In particular, our approach provides multi-tenancy: several virtual access points can be deployed on the same physical network.

Moreover, $SecuSpot$ provides scalability: several virtual APs and WiFi middlebox instances can be realized per tenant. Our controller handles the forwarding rules in the *wSwitch* and *OpenFlow* wired switch, *e.g.*, forwards the frames to the proper tenant or virtual access point instance.

Transmission rate and power control are handled locally at the first hop, by the access point. Thus, the remote controller needs to push the client's state to the local access point (see Figure 2). However, the crypto key is not installed on the access point but kept by the virtual AP instance in the operator's premises. Thus, the connection is not vulnerable to eavesdropping.

The management of the physical access points and, accordingly, control of the wireless switches is performed by the operator of the physical network. The virtual APs are managed by one or more controller instances, managed by the network operator running on general purpose computing hardware, hosted on-premise. The *wireless switch* allows to steer flows to the actual virtual AP in the Cloud based on the BSSID.

3. THE HEART: THE WIRELESS SWITCH

At the heart of SecuSpot lies a novel wireless switch, the *wSwitch*. The *wireless switch* supports matching on the BSSID, SSID, IEEE 802.11 Type and Subtype. Note, the SSID is only available in a few frames. Moreover, the wireless switch leverages the actions provided by the *Open vSwitch* such as tunneling (packet encapsulation) and packet forwarding. This enables multi-tenancy by forwarding IEEE 802.11 frames to endpoints in the Cloud or middleboxes: The *wSwitch* can classify, multiplex and demultiplex traffic directly on the access point. Moreover, data and control traffic can be forwarded independently: control traffic can for example be steered to the Cloud while data plane traffic is directly routed to the tenant. There is no need to store the crypto keys on the access point: rather, the traffic can be tunneled to a secure location where the keys are stored.

In more details, to connect via a multi-tenant access point, clients *actively scan* their neighborhood, sending *probe request frames*. These frames are forwarded to the registered tenants and local authorizers, *e.g.*, the *hostapd* which performs the MAC SubLayer Management Entity (MLME) together with Linux's *mac80211* wireless subsystem. After receiving the probe request frames, the virtual access point instances respond with a probe response frame to become potential candidate access points for the client to associate with. The virtual access point instances can be located on-premises (in the operator's Cloud) or located directly on the physical AP, *e.g.*, in case of a home network.

When the client starts the association process with an access point, the switch forwards the frames only to the particular access point instance. Here, the access point instance is identified by a unique BSSID (the MAC address of the AP). For instance, the virtual access points of a tenant could be identified by the *Organizationally Unique Identifier* (OUI), *i.e.*, the first three octets or upper 24 bits (two bits are used to signal the usage of unicast/multicast or global/local) of the MAC address. Hence, tenants can announce their own unique virtual access points.

Before a client can form an association with an AP, it needs to send an authentication frame to the virtual AP. Depending on the authentication scheme, this can involve the exchange of several authentication frames. After the client performed the authentication, it can continue with the association process to form a virtual link between the AP and itself. Specifically, the client can be authenticated at several APs, but can only be associated with one AP.

In case of a local virtual AP, the frame is handled by the local authenticator, *i.e.*, the *hostapd* instance running directly on the physical AP. However, in case of a locally deployed authenticator, the keys are usually stored on the physical access point, which could make the system subject to eavesdropping: the traffic will be decrypted at the AP. For instance, WiFi hotspots deployed in the user's premises are usually considered untrusted, since an attacker could compromise the system system by gaining physical control over the AP. Thus, the key material should not be stored on a hotspot deployed in an untrusted environment.

In the case of a virtual access point running in the operator's networks, the frames are encapsulated and forwarded by the *wSwitch* to the remote site. After the authentication succeeds with the client, the crypto keys are stored on premises in the remote virtual AP instance or network entity/middlebox. Thus, in case of a virtual access point hosted in the operator's network, no encryption keys are stored locally on the physical AP. Eventually, the crypto keys are stored in a more secure place which prevents eavesdropping at the physical AP. Accordingly, all encrypted 802.11 data frames from the client are decrypted at the remote site and vice versa. However, it is also possible to directly perform the decryption of the 802.11 traffic at the physical AP if necessary, and to switch the 802.11 frames or plain Ethernet frames locally.

4. IMPLEMENTATION DETAILS

The $SecuSpot$ WiFi controller is realized within an OpenFlow controller [3] and runs on a virtual machine or container. It in-turn runs atop general purpose computing hardware such as x86-based servers. The controller is realized as a Java Application and extends the OpenFlow protocol with basic WiFi management functionality as part of the *experimental* extension.

The wireless agent on the nodes is realized as a *C/C++*-based agent which registers as an OpenFlow-switch with WiFi capabilities. Our agent exposes an interface for the management of the *wireless switch*, WiFi features, and OpenFlow switch. The latter is used to steer flow directly at the access point and to forward the WiFi traffic to the Cloud/middleboxes accordingly. The management of the Wi-Fi is done through Linux's *netlink* interface.

The operating system on the WiFi APs is based on LEDE (formerly known as OpenWrt). It runs a novel *ovsd* [5] daemon which provides the interface and control logic for handling of the *Open vSwitch*, which is an OpenFlow-enabled replacement for the regular Linux bridge. The ovsd interacts with the LEDE networking environment through the *netifd*.

Moreover, we have extended Linux's standard 802.11 management daemon *hostapd* to provide an interface via LEDE's *ubus* message bus system. *ubus*-enabled programs can subscribe and listen to events and execute calls. Specifically, we have extended the *hostapd* service to provide the means to control the WiFi part including creating virtual access points, handling of client state, and client association management.

5. USE CASES AND EVALUATION

This section first discusses some use cases and then reports on first evaluation results.

5.1 Use Cases

To provide some ideas on how SecuSpot can be used and deployed, we discuss three examples.

Case 1: Cloud-assisted virtual WiFi.

In our first use case, we consider a WiFi hotspot deployed in an untrusted environment, where all traffic from a mobile client is for-

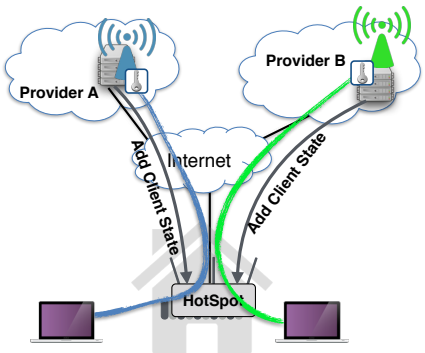

Figure 2: Case 1: Virtual WiFi AP in the Cloud (Cloud-assisted HotSpots). Two tenants (Provider A and B) provide WiFi hotspot services via virtual access points hosted in the Cloud, where all user traffic is decrypted.

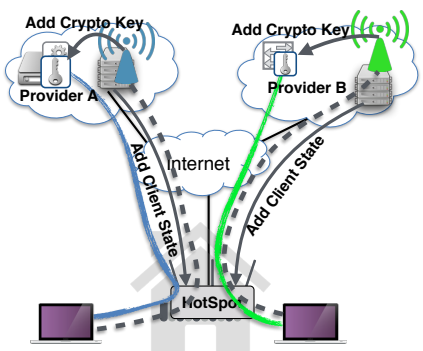

Figure 3: Case 2: Offloading of crypto functions into the datapath (Separation of 802.11 data and management). All data traffic is handled independently of the management traffic.

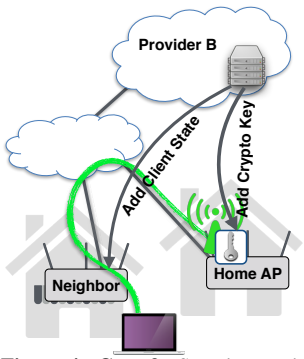

Figure 4: Case 3: Seamless mobility domain in a dense home network environment. Virtual WiFi AP in the remote home. En/Decryption is still performed at the user's home AP.

warded to a Cloud-based controller. Figure 2 depicts the scenario, where two tenants (Provider A and B) provide WiFi hotspot services via virtual access points hosted in the Cloud, where all user traffic is decrypted. This however, comes at the risk of overloading the operator's controller. Thus, reducing the load on the controller is paramount.

Case 2: Crypto into the Datapath (Separation of Data and Management).

In our second use case, we consider the approach of splitting the control and the data plane related traffic, *i.e.*, all data traffic is handled independently of the management traffic. Figure 3 shows the scenario, where the data traffic is decrypted in the datapath by specific network equipment/middleboxes. For instance, an ISP could deploy the decryption key in a BRAS or DSLAM. Alternatively, in the case of FTTB, the key could be also deployed in an aggregation box in the basement of a building.

Case 3: Cloud-assisted Seamless Secure Mobility Between Dense Home Networks.

Finally, we investigate the performance of a seamless mobility domain in a dense home network environment [16, 21], where a client can leverage neighboring physical access points to achieve high physical data rates at the wireless link. Here, all the security related features are still performed at the user's home AP to prevent eavesdropping. In contrast to solutions such as CAPWAP, the data is not decrypted directly at the first hop AP. By moving the client's association state around, the client does not need to re-authenticate when moving between physical APs. This allows seamless handovers between physical APs while keeping the security level high. This is particularly useful for Voice-over-WLAN (VoWLAN) where data frames are sent every 20 ms and which requires a roaming latency of less than 100 ms, to prevent the call disruptions when roaming to a new AP. Figure 4 depicts the example use case, where a client can exploit higher physical data rates by leveraging nearby neighboring APs. The traffic is forwarded by the wireless switch to the home AP, where the traffic is decrypted.

Note that while 802.11r (*aka*, *Fast BSS Transition*) already tries to minimize the delay when voice clients transition from one AP to another, the security related functions are still executed at the first hop AP. In a nutshell, 802.11r allows to establish security and QoS states at the target AP before or during a re-association. However, not all commodity end-devices today support 802.11r and some of the older devices have issues of parsing the new information which is carried in beacons and probe frames. Thus, 802.11r cannot be guaranteed in environments with a large device diversity as they exist for example in BYOD and home networks. However, the big advantage of 802.11r lies in the fact that no communication to the RADIUS Server is necessary with WPA2 enterprise security. This can dramatically reduce the handover delay, since the RADIUS server does not necessarily needs to be deployed in the same LAN. Thus, this feature is more useful in enterprise environments. In this setup all traffic is sent to the former AP which performs the security mechanisms.

5.2 Evaluation

We conducted a first performance analysis using our prototype implementation.

Testbed.

We use several access points from our WiFi testbed, which is located in an office building. The access points are x86-based AMD Geode PC Engines Alix 2d3 boards equipped with an Atheros 802.11n AR9280 WiFi card. The card can operate in the 2.4 GHz and 5 GHz spectrum.

We use a server with a Core i7 and 8 GB RAM for the controller, and an identical server for the datapath encryption as shown in Figure 3. We capture the traffic as soon as we reach the steady state. Each experiment run was at least 30 seconds. For the dense home network case (Figure 4), we can leverage the hardware crypto features from the AMD Geode processor. We have conducted several runs for each setup.

Case 1.

First, we evaluate the performance of running the virtual access points on premises in a remote location. Note, all 802.11 management and data frames are encapsulated and forwarded to the Cloud. In order to prevent fragmentation on the lower networking layers, we set the MTU of the link and signal the TCP MSS accordingly. We compare our results against the base-line *Standard* case, where all functions are performed directly on the AP.

Figure 6 shows the throughput performance on the downlink. We observe, that in *Use Case 1*, one can achieve almost as much throughput as in the *Standard* case. However, the maximum throughput is slightly lower when moving all 802.11 frames to the Cloud or middlebox in the datapath, *i.e.*, the bandwidth decreases due to the limitation of the uplink to 100 $Mbit/s$ and the additional tunneling overhead. This is also the case on the uplink as indicated in Figure 7. However, we observe no significant performance impact with our solution.

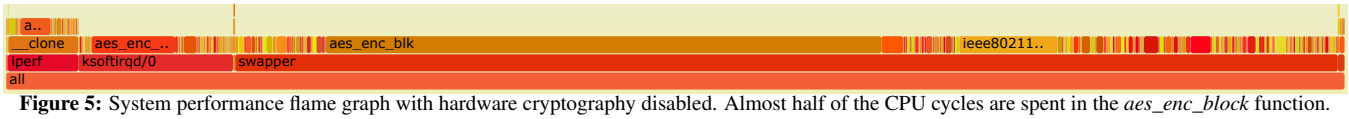

Figure 5: System performance flame graph with hardware cryptography disabled. Almost half of the CPU cycles are spent in the *aes_enc_block* function.

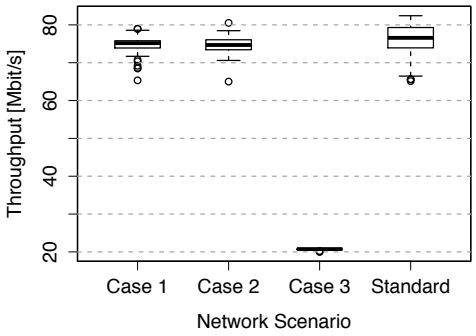

Figure 6: Downlink performance of an 802.11n link with cryptography enabled. Moving en/decryption to the Cloud has little performance implications.

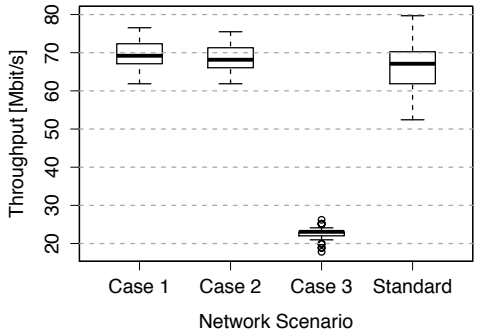

Figure 7: Uplink performance of an 802.11n link with cryptography enabled. Moving en/decryption to the Cloud has little performance implications.

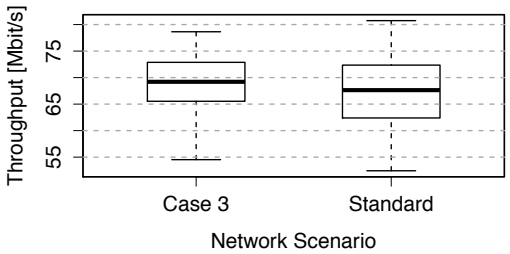

Figure 8: Comparison of the uplink performance without encryption. Case 3 could benefit from faster hardware crypto engines in today's APs.

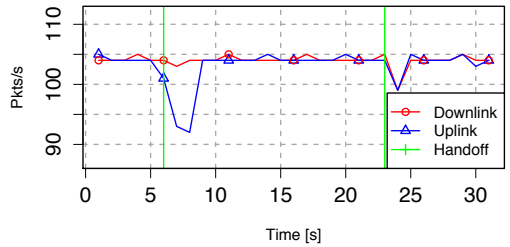

Figure 9: Impact of virtual AP migration on a voice call. We observe almost no packet loss during handovers, *i.e.*, basically virtual AP migrations.

Case 2.

Next, we evaluate the separation of the 802.11 management and data frames. Here, all management traffic is sent to a remote authenticator and all data traffic is handled by a crypto box in the datapath. We observe that the throughput is nearly the same compared to the previous case. Figure 6 and Figure 7 show that the throughput is close to 70 $Mbit/s$ and again close to the Standard case. The advantage of this case is to handle functions such as traffic en/decryption, en/decapsulation, and de-duplication directly in data plane instead of moving everything to operator's data center.

For instance, today's general purpose computing hardware deployed in the datapath can easily handle the traffic of hundreds of access points. Specifically, Intel's DPDK greatly boosts packet processing performance and throughput. Moreover, recent CPUs can easily handle several hundreds of gigabit of AES (block chain) traffic per second.

Case 3.

Finally, we evaluate the scenario of the seamless mobility domain in a dense home network environment, where a client can leverage neighboring physical access points to achieve high physical data rates at the wireless link. Here, all the security related features are still performed at the user's home AP to prevent eavesdropping. All traffic is forwarded to the user's home AP, *i.e.*, the AP running in the user's premises. This case is similar to the first case, however, the performance of the CPUs and SoCs of commodity home APs is by far lower compared to standard server computing power.

We see a performance drop by a factor of four down to $20Mbit/s$ (see Figure 6 and Figure 7). However, this is due to the limited crypto performance of the embedded CPU. Accordingly, we evaluated the performance without encryption. Figure 8 indicates that the performance without encryption is identical to the *Standard* case, *i.e.*, where all 802.11 frames are directly handled on the AP. Thus, we conclude, that the performance could be equal to our *Case 1* if the AP's CPU/SoC would provide a faster hardware crypto engine.

Moreover, this case also enables non-disruptive handovers between neighboring physical APs while keeping the security level high. This is particularly useful for Voice-over-WLAN (VoWLAN). Figure 9 shows the impact of handovers on a VoIP call, *i.e.*, we observe almost no packet loss during handovers.

Note, today's modern CPUs have a built-in crypto engine to realize tasks such as AES cryptography in hardware. While modern Intel CPU can easily achieve hundreds of gigabits per second, the performance of the embedded CPU AMD Geode LX 500 is fairly limited to just roughly 30 megabits per second. However, modern System-on-Chips (SoCs) provide much higher crypto performance [1]. Figure 5 shows the CPU cycles spent in *aes_enc_blk* when performing encryption in software. This indicates that there can be a huge performance gain when using hardware crypto of modern CPUs (*e.g.*, AES-NI of today's Intel CPUs).

6. RELATED WORK

The security risks introduced by today's hotspot infrastructures have already been reported in several reports [7]. Some vendors in the carrier wireless networks space try to mitigate security related problems by separately encrypting backhaul connections with IPSec or similar VPN protocols. This however leaves a weak link:

[1] https://wiki.openwrt.org/doc/howto/benchmark.openssl

the cleartext data can still be accessed and modified within the access point itself [7].

To the best of our knowledge, our work is the first to consider the design of a more flexible and secure multi-tenant WiFi hotspot architecture. However, our work builds upon several recent results. In particular, our prototype builds upon the LVAP abstraction of Odin [16] as well as the LegoFi vision of a more modular WiFi [15]. Moreover, while we set the focus in this work on the multi-tenancy aspects, we did not consider how to efficiently and/or fairly share the available bandwidth and wireless spectrum among tenants. There are several existing expedient solutions to the airtime fairness [11, 19] and service differentiation problem [14]. These, however, are orthogonal to our approach and can easily employed together with the *wSwitch* architecture.

Perhaps the closest work to ours are *Anyfi* [6] and CloudMAC [9]. Anyfi encrypts traffic end-to-end, from the mobile device to the Controller, using the standard IEEE 802.11i AES or TKIP encryption. However, it inherently enforces the bundling of all tenant (control and data plane) traffic, to be steered to the Cloud: this is not only inflexible but also also introduces a high load in the control plane. As far as we know, the Anyfi throughput is limited due to the handling of packet forwarding in userspace [6].

Moreover, CloudMAC [9] entirely offloads the non-realtime MAC layer processing to the Cloud. Similarly, due to the integration with OpenFlow CloudMAC provides a certain level of flexibility. However, it neither proposes the notion of a wireless switch to achieve full flexibility when handling IEEE 802.11 frames, nor does it allow handling of 802.11 frames in the data plane by middleboxes. Moreover, like Anyfi it depends on forwarding in userspace, which limits the performance. Since it relies on the Click Modular Router [12] it is limited to the older 802.11abg standards.

7. CONCLUSION

We understand our work as a first step toward a modern WiFi network which supports important multi-tenancy use cases while providing a high degree of security, by introducing flexibilities on where cryptographic operations are performed. Our first results are promising: our experiments show a high performance, using strong cryptography and standard Intel hardware only. The multi-Gbps rates are likely to increase further when using specialized AES crypto hardware. In our future work, we will incorporate existing airtime fairness and service differentiation approaches in our prototype.

8. ACKNOWLEDGMENTS

Julius Schulz-Zander was supported by the DFG project Gottfried Wilhelm Leibniz-Preis 2011 FKZ FE 570/4-1. Stefan Schmid was supported by the Danish VILLUM Foundation Project ReNet.

9. REFERENCES

[1] AeroHive Networks. http://www.aerohive.com.
[2] Aruba Networks. http://www.arubanetworks.com/.
[3] Floodlight. http://floodlight.openflowhub.org/.
[4] Meraki. http://www.meraki.com/.
[5] ovsd. https://github.com/berlin-open-wireless-lab/ovsd.
[6] Anyfi.net. Anyfi documentation. In *http://anyfi.net/documentation*, 2016.
[7] P. Bright. Insecure vodafone femtocells allow eavesdropping, call fraud. *Ars Technica*, 2011.
[8] P. Calhoun, R. Suri, N. Cam-Winget, M. Williams, S. H. B. O'Hara, and S. Kelly. Lightweight Access Point Protocol. RFC 5412, 2010.
[9] P. Dely, J. Vestin, A. Kassler, N. Bayer, H. Einsiedler, and C. Peylo. Cloudmac: An openflow based architecture for 802.11 mac layer processing in the cloud. In *2012 IEEE Globecom Workshops*, pages 186–191, Dec 2012.
[10] Y. J. Gwon, J. Kempf, R. Dendukuri, and R. Jain. VoIPv6 over IEEE 802.11b Wireless LAN. In *Proc. WiNMee*, 2005.
[11] T. Høiland-Jørgensen. Airtime fairness with mac80211 and ath9k. In *https://lists.bufferbloat.net/pipermail/make-wifi-fast/2016-June/000747.html*, 2016.
[12] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router. *ToCS 2000*.
[13] D. S. P. Calhoun, M. Montemurro. Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification. RFC 5415, 2009.
[14] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, and A. Feldmann. Opensdwn: Programmatic control over home and enterprise wifi. In *Proc. ACM Sigcomm Symposium on SDN Research (SOSR)*, 2015.
[15] J. Schulz-Zander, S. Schmid, J. Kempf, R. Riggio, and A. Feldmann. Legofi the wifi building blocks! the case for a modular wifi architecture. In *Proc. ACM MOBICOM Workshop on Mobility in the Evolving Internet Architecture (MobiArch)*, 2016.
[16] J. Schulz-Zander, L. Suresh, N. Sarrar, A. Feldmann, T. Hühn, and R. Merz. Programmatic Orchestration of WiFi Networks. In *Proc. USENIX ATC '14*.
[17] S. Sen, N. K. Madabhushi, and S. Banerjee. Scalable WiFi Media Delivery through Adaptive Broadcasts. In *Proc. NSDI*, 2010.
[18] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar. Can the production network be the testbed? In *Proc. 9th USENIX Conference on Operating Systems Design and Implementation (OSDI)*, pages 1–6, 2010.
[19] G. Tan and J. Guttag. Time-based fairness improves performance in multi-rate wlans. In *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, ATEC '04, pages 23–23, Berkeley, CA, USA, 2004. USENIX Association.
[20] S. Taylor, A. Young, and A. Noronha. What do consumers want from wi-fi? *Insights from Cisco Internet Business Solutions Group (IBSG) Consumer Research*, 2012.
[21] Y. Yiakoumis, M. Bansal, A. Covington, J. van Reijendam, S. Katti, and N. McKeown. BeHop: A Testbed for Dense WiFi Networks. In *Proc. WiNTECH '14*.