Chapter 4
Security Analysis of SDN WiFi Applications

David Artmann and Rahamatullah Khondoker

Abstract Mobile devices like smartphones, tablets and laptops demand highly-available and ubiquitous wireless networks, also named as Wireless Fidelity (WiFi) or Wireless Local Area Network (WLAN). The steadily rising amount of mobile devices implies new requirements claimed by administrators of enterprise wireless networks and owners of guest WiFi spots, such as the secure management of client authentication or the ability of load balancing. This work analyzes Odin, which solves the client association problem of wireless clients and OpenWiFi, a prototypical approach that separates authentication, access and accounting to raise the efficiency and lower the administrative effort for guest WiFi owners. Both technologies utilize SDN to regulate their objectives. This does not only bring benefits, but also implies new security aspects. Especially because SDN in WiFi is a young sector, developers need to make sure that their software ensures a proper security level. Subsequently, both technologies are evaluated by applying the threat modeling technique STRIDE. The decision on this framework is elucidated by comparing it against other possible alternatives. Our analysis reveals that both projects do not consider security at the beginning called security by design. Fortunately, Odin and OpenWiFi can be extended by suitable countermeasures to mitigate relevant threats. These are proposed in the respective subsection of their security analysis. Conclusively, optimization suggestions pertaining to both technologies are made.

Keywords Software Defined Networking (SDN) · Security · WiFi · STRIDE Odin · OpenWiFi · Security analysis

D. Artmann (✉)
Department of Computer Science, TU Darmstadt, Darmstadt, Germany
e-mail: artmann.david@gmail.com

R. Khondoker
Fraunhofer SIT, Darmstadt, Germany
e-mail: rahamatullah.khondoker@sit.fraunhofer.de; r.khondoker@yahoo.com

4.1 Introduction

With faster connection capabilities of multi-user-multiple-input and multiple-output (MU-MIMO), stronger performing hardware and steadily upcoming amount of communicating devices, wireless network infrastructures have become increasingly complex. Especially, the growth in mobile devices and upgrades of wireless connection standards like IEEE 802.11ac imply challenging requirements, such as the proper management of the association state of clients or offering comprehensible accounting and authentication of clients, which need to be accomplished by a new approach in networking.

Software Defined Networking (SDN) offers logically centralized control capabilities, an application programming interface (API) for network administrators to dynamically initialize, control, change and manage network behavior and a flow-based paradigm that is predestined for highly scalable wireless networks [10]. In detail, SDN decouples the data plane from the control plane, as visualized by control- and infrastructure layer in Fig. 4.1. OpenFlow, as utilized communication protocol between these layers exploits the fact that most Ethernet switches share a common set of functions. OpenFlow offers an interface to program the flow table in different switches and routers to dynamically add, edit and remove entries in the flow table [15]. The application layer residing on top of the architecture, consists of software that uses SDN communication services and utilizes an interface to the control layer via the northbound API. For example, the Odin Master, which will be introduced and explained as a central instance of the Odin Framework in Sect. 4.3, is such an application. The control plane is responsible for configuring the switch and routes, while the data plane forwards packets according to the decision of the control plane. Centralized instances, hosting the control layer logic, are called SDN controllers and take care of network route computation, configuration of network devices and

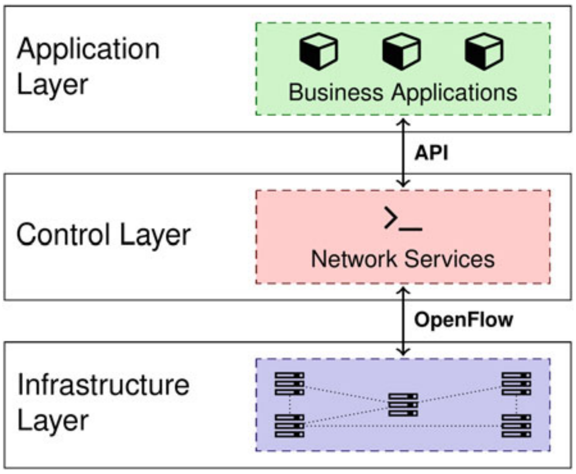

Fig. 4.1 Architecture of software defined networking (based on [14])

management of access control. Network elements (such as switches, routers) capable of packet switching and forwarding reside in the infrastructure layer.

Although the usage of this technology enables innovative ways in networking, it also implicates security considerations which should be taken into account. If an attacker is able to get control of a SDN controller, he or she is also able to reconfigure the whole network, respectively all components that rely on the accurate functionality of the controller instance.

This paper analyses the security capabilities of two selected technologies that utilize SDN in their architecture to determine if they implement a proper security level. For this purpose, different threat modeling frameworks are introduced in the next section and compared. This validates the most qualified approach which is subsequently used for the security analysis of Odin and OpenWiFi.

The remainder of this work is organized as follows. To justify our decision of using STRIDE, several possible security analysis frameworks are presented and compared in Sect. 4.2. Afterwards, Odins technology is introduced in Sect. 4.3.1 followed by its security analysis in Sect. 4.3.2. Similarly for the second technology, OpenWiFi, at first its architecture is introduced in Sect. 4.4.1 followed by the security analysis of its architecture in Sect. 4.4.2. Conclusively, the results of both security analyses are used while comparing Odin and OpenWiFi in Sect. 4.5. Lastly, the suggestion of possible optimization steps which exploit existing synergies of both technologies concludes this work.

4.2 Methodology

This section introduces STRIDE [12] which is used to analyze the security of Odin and OpenWiFi and gives reasons on its decision.

STRIDE is a threat modeling method developed by Microsoft and is a part of its Secure Development Lifecycle. By classifying threats in categories, STRIDE enables to identify vulnerabilities and threats in analyzed systems and their software. There are several other threat modeling techniques like OCTAVE [5], PASTA [1] or TRIKE [25]. While the first one is a complex and heavyweight solution which focuses on organizational risk but not on technical risk, the second is an attack simulation methodology where users need to be aware of the definition and technical scope of the respective application. This requires knowledge of the source code and therefore limits this approach to developers. Lastly, the third technique focuses on the design phase because it is a requirements-centric approach and involves stakeholders. We decided to use STRIDE because it is lightweight and focuses on technical risk analysis. It is used to analyze and evaluate the security of Odin and OpenWiFi. STRIDE is predestined for this task because it does not assume any implementation details of the software to be evaluated. Additionally, it does not depend on a risk model which would in turn hinge on aspects like operational condition, usage environment and customers needs. Thus, it fits for evaluating the security of software in a prototypical state like Odin and OpenWiFi. STRIDE is based on a threat modeling methodology and Data Flow Diagrams (DFD). STRIDE is an acronym for the listed threats in

Table 4.1 Threats and security properties [12, Fig. 3]

Threat	Security property
Spoofing	Authentication
Tampering	Integrity
Repudiation	Non-repudiation
Information disclosure	Confidentiality
Denial of Service (DoS)	Availability
Elevation of privilege	Authorization

Table 4.2 Symbols used in DFDs and their threats [12, Fig. 4]

Name	**Symbol**	**Susceptible to**
Data Flow	→	Tampering, Information disclosure, DoS
Data Store	═	Tampering, Information disclosure, DoS
Process	○	Spoofing, Tampering, Repudiation, Information disclosure, DoS, Elevationofprivilege
Multi-process	◎	Spoofing, Tampering, Repudiation, Information disclosure, DoS, Elevation of privilege
Interactor	▭	Spoofing, Repudiation
Trust Boundary	- - - - -	

the following Table 4.1. Each of them is mapped by a security property which is necessary to be available to guard against these threats.

The basic procedure of STRIDE is to decompose a system in its components and show that each of the components is not susceptible to relevant threats. DFDs are used to visualize the interaction of components in the decomposed system. The diagram consists of standardized symbols which are shown in Table 4.2.

The one way arrow of a Data Flow represents data in motion e.g., over a network connection. A Data Store which is visualized by two parallel lines describes data at rest like files on the hard disk or databases. Processes as well as Multi-processes describe programs currently being executed. An Interactor, expressed by a rectangle, is used for endpoints in the system like web services, servers or people. Borders between untrusted and trusted elements are represented by Trust Boundaries.

Table 4.2 and the DFDs constitute a framework for investigating how the evaluated systems might fail. The following sections describe both, Odin and OpenWiFi. Afterwards, the respective security analysis is accomplished by analyzing the DFDs and the threat model.

4.3 Odin

This section introduces Odin as a technology which solves the client association problem (is explained in the following section) by using SDN and analyses its security with the already envisaged threat modeling technique STRIDE.

4.3.1 Technology

Odin [24] was introduced as a Hot Topic of SDN workshops in August 2012 on the Special Interest Group on Data Communication (SIGCOMM). Suresh et al. designed this technology to overcome specific problems of wireless networks. In detail, the IEEE 802.11 standard allows the client to decide which access points (AP) it wants to be associated with [26, P.29], hence the infrastructure is not aware of this. Furthermore, the dynamic, broadcast and time-varying nature of the wireless medium in combination with the association state machine at the AP requires to keep track of state information changes constantly. Lastly, not only associated, but also interfering IEEE 802.11 devices need to be managed. As a fundamental element and to gain simplicity for programmers, Odin entails Light Virtual Access Points (LVAP). A LVAP constitutes an abstraction layer to separate the association state from the physical AP by virtualising it. This moves the association decision on the side of the infrastructure, enables programmers to handle several clients connected to one AP as logically isolated and gives the illusion of possessing its own AP to every client by assigning an unique Basic Service Set Identification (BSSID).

Figure 4.2 visualizes the architecture of Odin and is explained as follows. To target fully centralized deployments with a SDN controller, Odin sets one Master as an

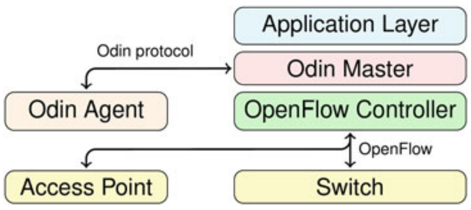

Fig. 4.2 The Master as OpenFlow application and the Odin Agents running on APs build the architecture of the Odin framework (based on [24, P.2])

OpenFlow application on top of a Floodlight SDN controller [21], communicating to Switches and APs. The Master has a global view of the network including connected clients and updates the forwarding tables on APs and Switches by using the OpenFlow protocol. Odin Agents, together with the Master implement a WiFi Split-Media Access Control (MAC) [4, P.7] which divides MAC functionalities between both parties. The Agents additionally contain the logic for LVAP handling. A Master controls Agents over a dedicated control channel via Transmission Control Protocol (TCP), which is established at boot time and allows it to add or remove LVAPs and query for statistics. Apps reside on top of the Odin Master which are allowed to inquire and examine those statistics. Each application is operated in a separate thread scheduled by the Master.

In the following, two figures and their respective explanation will contribute to a better comprehensibility of how Odin works. Given numerations in the figures will guide the reader through the process.

In Fig. 4.3, two clients are shown, Alice and Bob, connected to their exclusive LVAP. Those Light Virtual Access Points are hosted on Odin Agent 1 which runs on physical AP 1. This specific Agent as well as Agent 2, hosted on AP 2, are connected to the Odin Master via separate control channels. The use case of Figs. 4.3 and 4.4 implies a physical movement of Bobs client so that he has a stronger signal to AP 2. Going along from number 1 to 2 brings the mentioned signal strength variances on the APs. Going further along from number 2 to 3 the Odin Agent on the AP is queried for the signal strength of Bobs client by the Master. After the changed state is recognized, the Master decides to move Bobs LVAP in the custody of Agent 2.

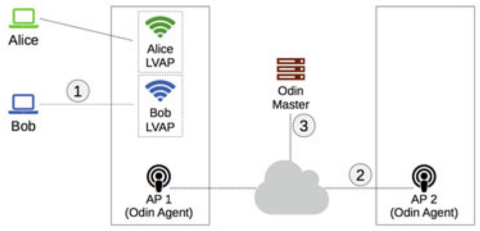

Fig. 4.3 LVAP migration because of client movement, part I

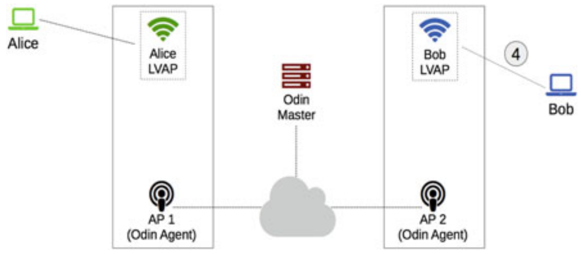

Fig. 4.4 LVAP migration because of client movement, part II

After the Master is informed as described before and is depicted in Fig. 4.3, it removes Bobs LVAP from AP 1 and adds it to AP 2, shown by number 4 of Fig. 4.4. If Bobs client has not been associated with the network before, a new LVAP on the AP which received the probe request would have been spawned, instead of moving it. Because Odin is fully transparent to the client, it does not witness this whole process.

While Sect. 4.3.1 introduced the technology of Odin the following section will document its security analysis by applying STRIDE.

4.3.2 Security Analysis

As mentioned in Sect. 4.2, a DFD is needed as for the security analysis using STRIDE. In the following, the DFD for the architecture of Odin is introduced. Afterwards, this DFD is used as a part of the threat modeling for the security analysis, targeting Odin.

Because the focus of this analysis lies on the purely Odin framework, the diagram in Fig. 4.5 entails the core of participating components: the Odin Master and an arbitrary number of Agents. Both are visualized as interactors because they are seen as end points in this model. Thus, the underlying OpenFlow controller, Access Points, and Switches as well as the Apps on top of the Master are omitted. In this figure the cardinality, as an originally foreign notation in DFDs, is introduced to point out that a Master can handle more than one Agent. Lastly, a Trust Boundary between Master and Agent reflects the inherently suspiciousness which is given by the fact that the Master has no opportunity to make sure an Agent has not been tampered with.

The adaption of STRIDE will be done from left to right with regard to Fig. 4.5, starting with the Odin Agent by stating the threats it is vulnerable to, appended with proper mitigation techniques.

Spoofing: In its present form, the implementation of the Odin Agent is not aware of a spoofed Master. This could be optimized by the use of mutual authentication, hence provide both sides with a spoofing protection like Kerberos [16].

Repudiation: In the scope of an Agent, repudiation means the inability of an Agent to proof that a specific Master has send the commands over its dedicated control channel. To avoid this lack of verifiability, both Master and Agent, should use digital signatures for signing all communicated data. As a result, an Odin Master as well

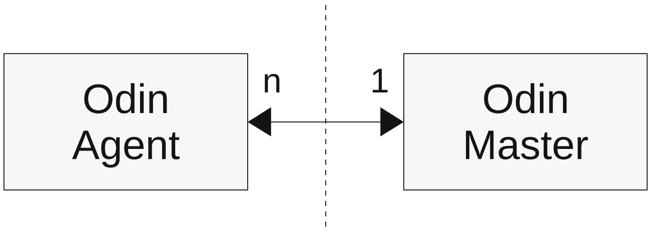

Fig. 4.5 The Data flow diagram related to Odin

as its Agents are able to validate the correctness of their counterpart. Administrators must be aware, that the introduction of digital signatures will cause a slight overhead due to the creation of signatures and their verification.

Further, the Data Flow between an Agent and its Master is analyzed.

Tampering: An attacker may manipulate the data in transit between Agent and Master. This means that the adversary is able to change a single bit in the message or add a whole payload to execute malicious code. A hindrance of this issue would be to secure the boot time established TCP/IP control channel via IPSec [13] Authentication Header on network layer or via Transport Layer Security (TLS) [7] on transport layer. Both techniques provide integrity checks, which would disclose modified packets.

Information disclosure: The data exchange between Agent and Master is not encrypted, thus an attacker is able to examine the traffic easily, because everything is sent as plaintext. An optimization regarding this lack of confidentiality is offered by IPSec. Utilizing its Encapsulating Security Payload mode, the whole traffic between Master and Agent can be encrypted. This enables to conceal the communication from any unauthorized individual and thus avoid the threat of information disclosure.

Denial of Service: This attack signifies that either a Master can not contact the Agents or vice versa. While analyzing this threat, the focus lies on the Master because it is a single point of failure and an inoperative Master would hinder the whole system to function. Whereas the unavailability of an Agent at worst causes some clients to be offline. Mitigating this attack could be done by using rate limiting or a load balancing system with high availability.

As a second interactor in the DFD of Fig. 4.5, STRIDE is also adapted on the Odin Master.

Spoofing: Although a spoofed client is prevented by the support of Wi-Fi Protected Access 2 [24, P.4], in the current implementation of Odin, a Master can not make sure that the Agent it is currently talking to is not spoofed. This could be circumvented by using a proper authentication mechanism between a Master and its Agents like the network authentication scheme RADIUS [22].

Repudiation: This threat implies that a Master can not proof, a specific Agent has communicated to it. One could claim that repudiation could be prevented by the fact, that the framework works on data link layer, i.e. with MAC addresses and thus the entity behind it can be identified. But unfortunately, MAC addresses can be spoofed. A solution would be processing each Address Resolution Protocol (ARP)-request and permitting only valid ones. Additionally, an Intrusion Detection System like Snort [23] could be used to support the validation process by monitoring for ARP-spoofing.

On a final note the following table 4.3 summarizes the antecedent analysis of the Odin framework. A *X* denotes that a threat could be mitigated by the suggested techniques, mentioned in the respective analysis of each threat.

Table 4.3 STRIDE threat matrix of Odin Framework

Type	Component	Threats					
		S	T	R	I	D	E
Interactors	Odin Agent	*X*		*X*			
	Odin Master	*X*		*X*			
Data flows	Agent ↔ Master		*X*		*X*	*X*	

A *X* denotes that a threat could be mitigated by the suggested techniques, mentioned in the respective analysis of each threat

4.4 OpenWiFi

After presenting the technology and the security analysis of Odin, the second part of this work introduces the architecture of OpenWiFi. Afterwards, the threat modeling technique STRIDE is applied on this prototypical technology as well.

4.4.1 Technology

Common guest WiFi systems such as CoovaChilli [6] typically are implemented as triple-A services and therefore unite access, authentication and accounting. Yap et al. introduced OpenWiFi as a prototype of separating a triple-A service into its single participating components, to reduce complexity and costs for guest WiFi owners. Additionally, the user gets relieved from the burden of remembering many different credentials for the various guest WiFi spots.

In detail, the inventors of this architecture suggest to delegate the authentication to third party service providers like Google or Facebook [27, P.4]. Those are able to handle the authentication by using techniques like OAuth2 [11] or OpenID [17]. The reason of choosing such well known authentication providers lies in their user amount, hence the probability is high that a guest WiFi user already has an account for the specific provider. Furthermore, access is provided by one or more APs which optimally support multiple SSIDs. This allows the guest WiFi provider to present several distinct WiFi networks to the user or host a private one in parallel. Additionally, if desired, the same SSID can be provided to the client across all networks. Lastly, a separate accounting service is suggested to enable responsibility delegation and billing.

The following example architecture represents the idea of OpenWiFi and is based on a given example of the inventors [27, P.5]. As in Sect. 4.3.1, a numeration is provided within Fig. 4.6 which will support the explanation.

Before going along the numeration and illuminate the given use case, each participating component deserves an introduction: going bottom up, two wireless clients, Home and Guest Device, can be seen whereby the focus lies on the latter. Surrounded by a rectangle, the guest WiFi owners OpenFlow enabled AP, a TP-Link

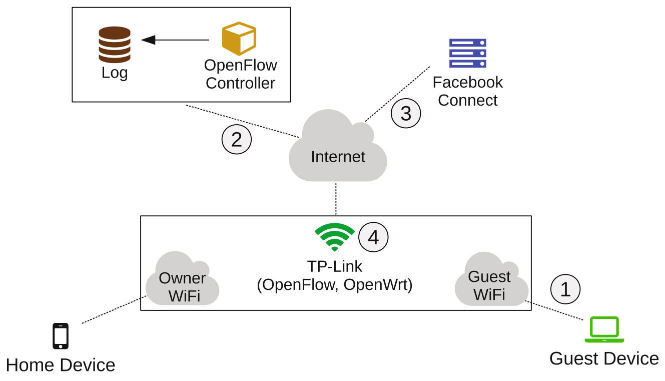

Fig. 4.6 Exemplary OpenWiFi architecture

TL-WR1043ND, hosts two SSIDs, Owner WiFi and Guest WiFi, to which the clients are connected. This AP runs on OpenWrt [18], revision backfire, which in turn runs the OpenFlow port Pantou [19] and is responsible for providing information used for access-control, redirection and accounting by the OpenFlow controller. It is enabled to offer these services, because the controller has been supplied with logic to make use of the given controls and statistics provided by the OpenFlow software switch, thus can be used as accounting service. Because Pantou seems to be a deprecated software, an exemplary technology in this case would be Open vSwitch [20]. Additionally, the OpenFlow controller logs authentication and flows to a SQLite database. Lastly, Facebook Connect is used as external authentication service provider (ASP).

After presenting the components and their roles in the OpenWiFi system, which builds on the cooperation of those, the given use case is described by going along the numeration of Fig. 4.6. Number 1 represents the initial login of a guest device to the WLAN network Guest WiFi. It is supplied with an IP address assigned via DHCP and marked as unauthenticated. While the client is labeled with this status, only its ARP, DNS and DHCP traffic is permitted [27, P.4]. Number 2, as the next step, states the redirection of a user after opening the web browser to the landing page of the accounting service. This is realized by hijacking the HTTP traffic and performing an HTTP redirect with client error code 403 [8, P.59]. In this case the user has only one choice with the mentioned authentication service (AS) of Facebook showing up on the appeared landing page. Number 3 represents clicking on the button for Facebook Connect. Subsequently, the users traffic gets forwarded to the authentication site of Facebook for entering credentials. While this happens, OpenWiFi marks the user with a new status, pending authentication. After credentials are submitted, identity is established and a valid authentication is assured, the user is asked for permission to reveal information to OpenWiFi. Confirming this dialog, the AS returns an access token to OpenWiFi and the user gets forwarded to an approval page. At this point, the user is labeled as authenticated. Lastly number 4 implies the usage of the returned access token. It enables to retrieve the users identity to associate corresponding traffic for accounting purpose.

After introducing the technology of OpenWiFi, the next Sect. 4.4.2 utilizes STRIDE to document its security analysis.

4.4.2 Security Analysis

As seen in the exemplary OpenWiFi architecture, the core components of this technology are an OpenFlow controller, a central Access Point and the external third party ASP, like Facebook Connect. Because each of the components is seen as end point in the model, these elements are visualized as interactors in Fig. 4.7. The WLAN clients as well as the network clouds are omitted since the focus lies on the core of OpenWiFi. Because each component lives on its own and does not directly belong to another, they are separated by Trust Boundaries.

With regard to Fig. 4.7, the analysis will go along the elements from left to right starting with the OpenFlow controller.

Spoofing: If an attacker is able to spoof an AP to the controller, he or she is also able to fake provided information about access control or redirection. OpenWiFi does not mitigate this threat by default. Prevention is constituted by ensuring authentication, e.g. by using certificates to sign the communicated data between Access Point and OpenFlow controller.

Repudiation: To assure non-repudiation to the controller, OpenWiFi has to implement digital signatures or the usage of timestamps to prevent deniability. If spoofing is already prevented by the usage of certificates, one can exploit this by utilizing existing certificates, assuming that they can be used for digital signatures.

Continuing with the next component, the Data Flow between OpenFlow controller and Access Point is analyzed.

Tampering: The possibility of an attacker to modify sent data between the controller and AP among others enables the attacker to tamper with accounting information and thus bypass upcoming payments. Because the OpenFlow protocol is used, which lies on top of the transport layer and uses TLS [15], the Data Flow is protected against tampering.

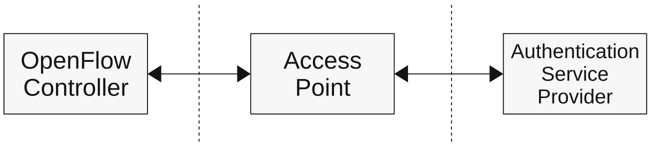

Fig. 4.7 The Data flow diagram related to OpenWiFi

Information disclosure: If TLS is used for encapsulating the traffic, any unauthorized person or program is unable to get access to the plain text behind encrypted traffic.

Denial of Service: In this context, the unavailability of a link between both parties, OpenFlow controller and Access Point, means no access control, redirection or accounting. As a consequence, the OpenWiFi architecture would be inoperative because the system can not survive without the OpenFlow controller or the Access Point. A possible mitigation, likewise described in Sect. 4.3.2, is high availability by using more than one device per side or the usage of Quality of Service by favoring the traffic between participating entities and throttling the remaining.

In the following, the Access Point as second interactor of the diagram is examined.

Spoofing: An AP is vulnerable to faked opposites in two ways. First, by a spoofed OpenFlow controller which could lead to false access controls or other unauthorized decisions. And second, in form of a spoofed authentication service, which could lead to faked identities. Both scenarios can be mitigated by the usage of mutual authentication techniques like provided by the authentication scheme Kerberos.

Repudiation: If certificates are already in use, then the administrator of OpenWiFi is able to take advantage of this by also using them for digital signatures to assure non-repudiation. If not, than additionally to certificates, the usage of timestamps would be a countermeasure for deniability.

As a fourth component, STRIDE is also adapted on the Data Flow between an Access Point and the Authentication Service Provider.

Tampering: If an adversary is able to modify the data in transit between the AP and ASP, he is also able to fake or manipulate the authentication by using another identity. This can be mitigated by the usage of HTTP over TLS (HTTPS) which uses keyed message authentication code to verify the integrity of data [7, P.13].

Information disclosure: When the connection between AP and ASP is based on HTTPS, their link is secured against exposure of any data which is transported over the channel, as long as the underlying TLS is configured with a proper cipher suite that securely encrypts the traffic.

Denial of Service: This threat needs to be split up in two parts. The first one consists of outsourcing the ASP. So in the majority of cases, the OpenWiFi architecture is unable to affect it in any way which implies its configuration and protection against DoS of any kind. Hence, OpenWiFi depends on the proper security mechanisms of the ASP and has no influence on it. It is only able to control the AP-side by offering several devices and backups for failover.

To wrap up the security analysis of OpenWiFi, we apply STRIDE on the ASPs interactor side.

Spoofing: As mentioned earlier, most probably, OpenWiFi does not influence the ASP. Thus, the latter must be protected from a spoofed AP on its side of the relation. Although the provider is not part of the native OpenWiFi system, a proper security

Table 4.4 STRIDE threat matrix of OpenWiFi

Type	Component	Threats					
		S	T	R	I	D	E
Interactors	OF controller	*X*		*X*			
	AP	*X*		*X*			
	ASP	*X*		*X*			
Data flows	OF controller ↔ AP		✓		✓	*X*	
	AP ↔ ASP		*X*		*X*	*X*	

A *X* denotes the suggestion of countermeasures and a ✓ states that OpenWiFi mitigates the threat by default. OF stands for OpenFlow

of this external part also contributes to a better one of OpenWiFi. A solution for both sides is the usage of a common certification authority which both sides trust. Hence, they can use the distributed certificates for the protection against spoofing.

Repudiation: If the AP and ASP are already using certificates, then they can rely on this to protect against deniability.

The following Table 4.4 closes up the security analysis of the prototypical approach OpenWiFi and gives an overview of the components and the respective threats they are susceptible for. A *X* denotes the suggestion of mitigation mechanisms and a ✓ states that OpenWiFi mitigates the threat by default.

4.5 Comparison and Conclusion

The vigorous surge of mobile devices, similar to mobiles predecessor, the telephone, has revolutionized communication. Their demand for a flexible and manageable network administrating approach has led to software like Odin and OpenWiFi.

Whereas the former is a framework which uses SDN to establish the central maintainability of wireless clients and their AP association, OpenWiFi is an approach to separate authentication, access and accounting to simplify the process of providing guest WiFi systems.

To facilitate a proper security analysis, we validated the most fitting technique in Sect. 4.2 and applied the elected candidate, STRIDE, to Odin and OpenWiFi.

Although both technologies have been invented for different purposes, they could be used to complement each other. To go into detail: with Odin already in use as a base of the wireless network, OpenWiFi can be built on top of it. This is possible because both technologies rely on an OpenFlow controller which could be used as synergy, assuming compatible hardware is used. As a result, it would be even more efficient and comfortable for the guest WiFi owner to offer different SSIDs or manage load balancing. Keeping this idea in mind, the administrator could also benefit of the merged security features which have to be realized when building up such a system.

Because of their prototypical state both technologies do not essentially involve security in their design yet. But as presented in Sects. 4.3.2 and 4.4.2, they fortunately entail opportunities to add security.

References

1. Morana MM, Ucedavelez T (2015, May) Application threat modeling
2. Bansal M, Mehlman J, Katti S, Levis P (2012) OpenRadio: a programmable wireless dataplane. HotSDN12. Helsinki, Finland
3. Baskett P, Shang Y, Zeng W, Guttersohn B (2013, March) SDNAN: Software-defined networking in ad hoc networks of smartphones. 10th consumer communications and networking conference (CCNC). Las Vegas, Nevada
4. Calhoun PR, Montemurro MP, Stanley D (2009, March) RFC 5416: control and provisioning of wireless access points (CAPWAP) protocol binding for IEEE 802.11. Network Working Group
5. Caralli RA, Stevens JF, Young LR, Wilson WR (2007, May) Introducing OCTAVE allegro: improving the information security risk assessment process. CarnegieMellon University
6. CoovaChilli Project. https://coova.github.io/CoovaChilli/. Accessed: December 2016
7. Dierks T, Rescorla E (2008, August) RFC 5246: The Transport Layer Security (TLS) protocol Version 1.2. Network Working Group
8. Fielding R, Reschke J (2014, June) RFC 7231: Hypertext Transfer Protocol (HTTP/1.1): semantics and content. Internet Engineering Task Force (IETF)
9. Gudipati A, Perry D, Li LE, Katti S (2013) SoftRAN: Software Defined Radio Access Network. HotSDN13. Hong Kong, China
10. Haleplidis E, Pentikousis K, Denazis S, Salim JH, Meyer D, Koufopavlou O (2015, January) RFC 7426: Software-Defined Networking (SDN): layers and architecture terminology. Internet Research Task Force (IRTF)
11. Hardt D (2012, October) The OAuth 2.0 authorization framework. Internet Engineering Task Force (IETF)
12. Hernan S, Lambert S, Ostwald T, Shostack A (2006) Uncover security design flaws using the STRIDE approach. Available: https://msdn.microsoft.com/magazine/msdn-magazine-issues. November 2006. Accessed: November, 11th 2016
13. Kent S, Seo K (2005, December) RFC 4301: Security Architecture for the Internet Protocol. Network Working Group
14. Kolias C, Ahlawat S, Ashton C, Cohn M, Manning S, Nathan S (2013, September) Openflow-enabled mobile and wireless networks. Open Network Foundation
15. McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, Shenker S, Turner J (2008, March) OpenFlow: Enabling Innovation in Campus Networks. ACM SIGCOMM Comput Commun Rev 38(2):69–74. https://doi.org/10.1145/1355734.1355746
16. Neuman C, Yu T, Hartman S, Raeburn K (2008, July) RFC 4120: the Kerberos network authentication service (V5). Network Working Group
17. *OpenID Project*. https://openid.net/. Accessed: January 2017
18. OpenWrt Project. https://openwrt.org/. Accessed: December 2016
19. Pantou implementation. https://github.com/CPqD/openflow-openwrt. Accessed: December 2016
20. Pfaff B, Pettit J, Koponen T, Amidon K, Casado M, Shenker S (2009, October) Extending networking into the virtualization layer. 8th ACM Workshop on Hot Topics in Networks. New York City, NY
21. Project Floodlight. http://www.projectfloodlight.org/floodlight/. Accessed: December 2016

22. Rigney C, Willens S, Rubens AC, Simpson WA (2000, June) RFC 2865: remote authentication dial in user service (RADIUS). Network Working Group
23. Snort Project. https://snort.org/. Accessed: December 2016
24. Suresh L, Schulz-Zander J, Merz R, Feldmann A, Vazao T (2012) Towards Programmable Enterprise WLANS with Odin. Helsinki, Finland
25. Saitta P, Larcom B, Eddington M (2005, July) Trike v.1 methodology documentation. Octotrike
26. Yang L, Zerfos P, Sadot E (2005, June) RFC 4118: architecture taxonomy for control and provisioning of wireless access points (CAPWAP). The Internet Society
27. Yap KK, Yiakoumis Y, Kobayashi M, Katti S, Parulkar G, McKeown N (2011, July) Separating Authentication, Access and Accounting: A Case Study with OpenWiFi. Stanford University