# CROCUS: An objective approach for SDN controllers security assessment

Carlos Silva[1], Bruno Sousa[1], and João P. Vilela[2]

[1] University of Coimbra, CISUC, DEI
`carlosfelix@student.dei.uc.pt, bmsousa@dei.uc.pt`

[2] CRACS/INESCTEC, CISUC and Dep. of Computer Science, Faculty of Sciences, University of Porto, Porto, Portugal `jvilela@fc.up.pt`

**Abstract.** Software Defined Networking (SDN) facilitates the orchestration and configuration of network resources in a flexible and scalable form, where policies are managed by controller components that interact with network elements through multiple interfaces. The ubiquitous adoption of SDN leads to the availability of multiple SDN controllers, which have different characteristics in terms of performance and security support. SDN controllers are a common target in network attacks since their compromise leads to the capability of impairing the entire network. Thus, the choice of a SDN controller must be a meticulous process from early phases (design to production). CROCUS, herein proposed, provides a mechanism to enable an objective assessment of the security support of SDN controllers. CROCUS relies on the information provided by the Common Vulnerability Scoring System (CVSS) and considers security features derived from scenarios with stringent security requirements. Considering a vehicular communication scenario supported by multiple technologies, we narrow the selection of SDN controllers to OpenDayLight and ONOS choices. The results put in evidence that both controllers have security features relevant for demanding scenarios with ONOS excelling in some aspects.

**Keywords:** SDN · security · ONOS · OpenDayLight · DoS · MADM

## 1 Introduction

Cloud computing has transformed the way computing resources are provisioned. The consolidation of the paradigm provided a significant increase in the number of services available on the Internet and, consequently, in the number of users connected to the network. According to some researches [1], it is estimated that the number of devices connected to the Internet will be one trillion by the year 2025. Multiple technologies can be employed to enable the connection of heterogeneous devices (e.g. vehicles, smart lamps), and IoT objects in scenarios with stringent performance and security requirements (e.g. low latency, data encryption) [2].

The Software Defined Networking (SDN) paradigm facilitates the orchestration and configuration of network resources that can be placed at the edge and cloud side to fulfil scenarios' requirements. SDN Controllers are able to manage policies regarding network access, regarding services (to enable the chaining of service functions) in a flexible and scalable way. For such management the controllers have NorthBound interfaces to allow the connection from applications, SouthBound interfaces to allow the connection with network equipments using protocols like OpenFlow and NetConf. The East-West interfaces are employed for clustering purposes. SDN controllers like ONOS and OpenDayLight support features that are crucial to reduce security risks and to increase availability levels, such as the support of clustering to avoid Single Point of Failure (SPoF) [3–6]. Apart from these criteria, other aspects such as the scale of deployments and modularity also play a key role in the selection of suitable SDN controllers. For instance, controllers like Ryu or Pox, which are tailored for research purposes and are not seen as suitable choices for scenarios with stringent security requirements [7].

The ubiquitous adoption of SDN leads to the availability of multiple SDN controllers (e.g. ONOS, OpenDayLight, FloodLight, Ryu and Pox), which have different characteristics in terms of performance and security support. However, the increase in the number of aspects to be considered also generates an increase in the decision process complexity for choosing suitable controllers. The Multiple Attribute Decision Making (MADM) [8,9] is a multidisciplinary methodology that assists in the decision process when it is necessary to consider multiple attributes that should be maximized or minimized according to their degree of influence in the decision. In this regard, authors [10] propose feature based selection approaches to select SDN controllers, employing MADM mechanisms to compare diverse SDN controllers and enable an informed selection of SDN controllers, nonetheless the proposed approach does not includes security information and omits recent controllers like ONOS.

SDN controllers are a common target in network attacks since their compromise leads to the capability of impairing the entire network. Thus, the choice of a SDN controller must be a meticulous process from early phases, from design to production/deployment, without neglecting security aspects. Indeed, threat modelling and risk determination techniques are relevant to analyse different choices [11]. In spite of the availability of security analysis for SDN controllers [12], the employed threat modelling approaches like STRIDE [13] rely on subjective classifications, which may lead to biased or ineffective results. In addition, the modelling in such approaches is limited to the security features supported in SouthBound and NorthBound interfaces, taking out East-West interfaces required for clustering purposes.

The CROCUS methodology herein proposed provides a mechanism for objective assessment of the security support of SDN controllers. CROCUS relies on the information provided by the Common Vulnerability Scoring System (CVSS) [14], and also includes security features derived from scenarios with stringent security requirements and considered as mandatory in security studies [3, 4]. CROCUS assesses the security support of SDN controllers in an objective fashion by considering: i) the vulnerability information (CVSSv3), updated frequently and supporting the activities of Chief Information Security Officer (CISO); ii) the information of risk assessment approaches that enable the determination of the severity levels and probability of occurrence [11]; iii) the information of security features deemed as necessary for SDN controllers [3, 4], considering the application, control and data planes in SDN. CROCUS establishes a multi-step approach to consider the complexity of the multiple aspects in the decision process, through Methodical - a MADM approach which is available online[3], and has been employed by us in previous studies for an objective selection of choices for content migration and to enhance resilience support in cloud [15,16] In this work, considering a vehicular communication scenario supported by multiple technologies as case-study, we employ the proposed CROCUS methodology to narrow the selection of SDN controllers to OpenDayLight and ONOS. The obtained results put in evidence that both controllers have security features relevant for demanding scenarios, with ONOS standing out in some of the criteria.

This article is organised as follows: Section 2 discusses works that already exist in the literature, while section 3 presents the background with the main topics of the paper. The CROCUS mechanism is described in section 4, while section 5 describes the employment of CROCUS to enable the selection of SDN controllers in scenarios with multiple technologies. The final conclusions are presented in section 6.

## 2 Related Work

The choice of SDN controllers is commonly based on performance criteria [17]. As an example, authors [10] employ MADM mechanisms to compare OpenDayLight, FloodLight, Ryu and Pox controllers in order to select the one with the most appropriate feature set. The study does not includes security information and omits recent controllers like ONOS.

Apart from the performance concerns, there is an increasing focus on security aspects of SDN controllers. In particular, authors [18] reveal that the advantages of SDN also brings security concerns due to the split between the control and data plane, and due to the higher exposure of attacks, since one entity manages all the roles for traffic forwarding in the network.

Security analysis of controllers, employing threat modelling approaches [12], consider the selection of SDN controllers mainly based on the information of SouthBound and NorthBound interfaces, disregarding East-West interfaces that are required for clustering purposes. The threats are considered as per the STRIDE threat modelling approach [13] which tends to be subjective regarding the classification of threats.

Authors also propose an analysis of the security in SDN controllers, focusing on the implemented functionalities [3, 4] and on threat models associated with Denial of Service attacks [19]. Authors specify the metrics that SDN controllers must support to mitigate such attacks. For instance, to avoid false master

[3] `https://github.com/bmsousa/MeTHODICAL`

nodes to control other SDN controllers, the messages exchanged between controllers must have security mechanisms (e.g., integrity, encryption) associated. Notwithstanding, no methodology is provided to enable an objective selection of controllers in such studies. The work of these authors is, however, employed in CROCUS to help define the “standard security features” that SDN controllers must support.

The main objective of this work is to propose a selection strategy for SDN controllers that, unlike previous works, considers security aspects from design to production phases using MADM methodology. With the possibility of customising the factors considered in the process, as well as its impact on the decision, the MADM methodology allows for the flexibility of the proposed strategies to encompass different security requirements.

## 3 Background

### 3.1 Software Defined Networks

The Software Defined Networks (SDN) paradigm allows the dynamic programming of the network infrastructure, due to the split of the data plane and the control plane. The latter has the role of managing the logic regarding the forwarding of data through all the devices in the network (e.g. hosts, switches, routers, etc). The SDN architecture considers three distinct planes, as suggested by the Open Network Foundation [20].

– **Application Plane** contains the applications that are responsible for the network management in real time. This plane includes applications that are responsible for the policies for traffic steering, load balancing, and security (e.g. Deep Packet Inspection). Such applications communicate with the SDN controller though specific APIs, available at the NorthBound Interface (NBI).
– **Control Plane** is responsible for the network management and sends to the controller requests to configure the behaviour of the data plane. For instance, data flows associated with a specific service (e.g. using as destination a certain port, 80 for HTTP service) are forwarded in a specific switch port, or also mirrored in another switch port for security analysis. This plane allows to manage the behaviour of the SDN controller, in particular on how the controller manages the flow policies. Another example is the usage of Intents, to express the desired behaviour regarding traffic steering, so that the controller is able to infer the necessary flow rules that are required for two entities to communicate [21].
– **Data Plane** abstracts the physical network components like switches that receive traffic and forward the traffic, firewalls which perform security functionalities. This plane is mainly concerned the forwarding process of data packets, which considers the information in the packets (e.g. MAC, IP addresses, etc) to forward traffic. When no rules exist for a given flow, the SDN controller is queried regarding the intended behaviour for that specific flow, if it should be discarded or forwarded.

### 3.2 Common Vulnerability Scoring System

The Common Vulnerability Scoring System (CVSS) [14] is an open framework to communicate the characteristics and severities of software. CVSS combines different categories of metrics into a score that varies in a $[0, 10]$ scale, where the first is classified as informative and the last as critical. The diverse categories of metrics include:

– **Base** contains information regarding the vulnerabilities that do not change with time, as happens with the temporal vulnerability categories.
– **Temporal** contains information regarding vulnerabilities that change over time due to events external to each vulnerability.
– **Environment** refer to the environment where the vulnerability can occur, thus including information regarding the required privileges to exploit a vulnerability.

The CVSS score is determined considering the metrics in the base category, which is required to determine the severity associated with a vulnerability. In addition, the base category considers two sub-types of metrics: the exploitability and impact, with the first related with the availability and easiness of exploring the vulnerabilities, while the impact is associated with the extent achieved with the successful exploitation of the vulnerability. Table 1 summarises the diverse metrics of the base category.

Table 1: CVSS Base metrics and values

| **Sub-type** | **Metric** | **Values** | **Description** |
|---|---|---|---|
| Exploitability | Attack Vector #**AV** | {1,2,3,4} | Value 4 refers to physical access to the vulnerable component, 1-refers to remote access. |
| | Attack Complexity #**AC** | {1,2} | Value 2 refers to high complexity while 1 refers to low complexity. |
| | Privileges Required #**PR** | {0,1,2} | Value 2 refers to high privilege (e.g root), 1 to low and 0 for none. |
| | User Interaction #**UI** | {0,1} | 0-No interaction, 1-Requires user interaction. |
| | Scope #**S** | {0,1} | 0-If only impacts the component, 1-If impacts other components beyond the security scope. |
| Impact | Confidentiality #**C** | {0,1,2} | 0-No loss of confidentiality, 2-Total loss of confidentiality. |
| | Availability #**A** | {0,1,2} | 0-Availability is not affected, 2- Successful denial of service. |
| | Integrity #**I** | {0,1,2} | 0-No impact on the integrity, 2- Total integrity loss. |
| | CVSS score **CVSS3** | $[0, 10]$ | Score that combines the metrics of the base category. |

The base metrics are required to establish the base score of CVSS. In addition, the scope metric allows to identify if a metric has impact on other component, for instance in Cross Site Scripting (XSS) vulnerabilities, the security of the Web servers needs to be compromised but it also impacts applications (e.g. browsers) running on end-user devices. The CVSS score is formulated according to the version of the CVSS. The most recent version is v3.1, but it does not introduces new metrics or metric values and changes in formulas when compared to v3.0 [22].

### 3.3 Multiple Attribute Decision Mechanism

Multiple Attribute Decision Mechanisms (MADM) have been employed in distinct domains as an approach to rank alternatives. For instance the Methodical algorithm [8] has been employed in scenarios for resource migration, path selection, QoS decision. In order to rank alternatives considering the MADM algorithms, the first step is to classify alternatives into two main categories/groups:

– **Benefits** includes all the metrics whose value should be maximised. For instance, values of #AV must be higher (physical access) in order to allow the full exploitation of vulnerability.
– **Costs** considers the metrics whose values must be minimised. For instance, values of the CVSS score must be close to zero, as they present lower probability of exploitation and reduced impact in the system and services.

The MADM also provides flexibility to specify the importance of one criterium over another, through weights. In addition, categories can also be weighted, for example to give preference to the benefits category over the costs category. MADM algorithms are performed in several steps, including normalisation of values to apply weights, determination of ideal values in terms of benefits and costs, the distance of each alternative to the ideal values, and an aggregation of the distances in a score to allow the ranking of alternatives. A score close to zero represents that an alternative is closer to the ideal values, thus holds best values. A more detailed description can be found in [8].

## 4 CROCUS: SDN Controllers' Security Assessment Approach

This section describes the CROCUS approach which aims to enable the assessment of SDN controllers at design and production phases. The approach relies on well established security methodologies, like the Common Vulnerability Scoring System (CVSS) and on MADM to rank the vulnerabilities, and aims to be employed as a tool to those who require SDN mechanisms. CROCUS works in multiple steps in a closed loop and include: 1) Identification of SDN Controllers; 2) Information of vulnerabilities; 3) Rank vulnerabilities; 4) and finally perform selection of SDN controller. Such steps are further documented in the following subsections.

### 4.1 Step #1 - Identification of SDN Controllers

Besides the flexibility and management of the diverse SDN data planes, as described in section 3.1, the choice of SDN controllers can be related with the scenario, or with a specific purpose. From a security perspective, the choice can rely on controllers that can act in a cluster to avoid Single Point of Failures (SPoF) [23, 24]. The scale of deployments and the modularity also play a key role in the selection step, where SDN controllers like Ryu or Pox are more focused on research, and thus not identified as suitable choices [7]. The output of this step is the identification of possible SDN controllers - set of SDN controllers to deploy in a specific scenario.

### 4.2 Step #2 - Information of Vulnerabilities

Considering the input of the first step on the set of SDN controllers, the information of vulnerabilities can be queried using available information, like the one present in CVSS, as presented in section 3.2. The threat model can impact the collection of the vulnerabilities information, CROCUS does not instantiate to a particular threat model (e.g. DoS), as these are associated with the specificity of the scenario. As such, CROCUS proposes to include all the vulnerabilities, that may require physical access to be exploitable, or that map to insider threats (e.g., malicious administrators).

The CVSS version considered in CROCUS is CVSS v3, since the differences from v3.1 mainly rely on clarification aspects, and the vulnerabilities information conducted in the study case for SDN controllers mainly includes version v3. CVSS allows to determine the CVSS score, nonetheless, relevant information,

Table 2: CROCUS added metrics and values

| Metric | Values | Description |
|---|---|---|
| Status **#Stat** | {0,1} | 1-If solved, or 0- yet without a fix. |
| Update days **#UptDay** | $[0, \infty]$ | Number of days since identification and publishing of vulnerability and its resolution. |
| Actives at App plane **#AppPlane** | {0,1} | 1- If affects actives at the application plane. |
| Actives at Control plane **#CtrlPlane** | {0,1} | 1- If affects actives at the control plane. |
| Actives at Data plane **#DataPlane** | {0,1} | 1- If affects actives at the data plane. |
| Severity Level **#SevLevel** | $[1, 6]$ | 6- stands for extreme, while 1- is for negligible levels. |
| Probability of Occurrence **#Prob** | $[1, 6]$ | 6- stands for maximum, while 1- is for negligible probability. |
| Mitigation Measures **#MitMea** | {0,1,2} | 0- no measures, while 2- if for more than one measure. |

such as the state of resolution, the update date, as well as the impact in the diverse SDN planes (application, control, and data) needs to be considered for a complete and informed decision process. In this aspect, CROCUS proposes a set of additional metrics to fulfil this gap, as summarised in Table 2.

The status and update days metrics, which provide information regarding the correction of the vulnerability (in the form of software patches, new software versions) can be collected using the available information of vulnerabilities in the National Vulnerabilities Databases (NVD) [14]. It should be noticed that the update days does not necessarily mean that the vulnerability is solved, as it may include updates regarding available information, for instance to apply temporary configuration fixes in order to reduce the impact. Such metrics are also relevant to highlight the support of the community to correct and enhance features in a given SDN controller. For instance, SDN controllers without modifications in a period of one year might indicate low support from the community to add new functionalities to a SDN controller (e.g. add support for P4), or to correct identified vulnerabilities.

The severity level and probability of occurrence are introduced, considering the combination of vulnerability assessment and risk determination logic for enterprise scenarios and services [11]. Such reasoning allows one to assess the real impact of the vulnerabilities in enterprise scenarios or services relying on SDN controllers. In such context, the metrics regarding the actives of the diverse planes highlight which components are affected considering the SDN planes, as described in section 3.1. For instance, if a vulnerability only affects applications running on top of the controller, using the NBI interface, or affect either the control and data planes, either at the controller and/or switching equipments communicating through the SBI interface.

In a generic perspective, additional metrics like the number of mitigation measures are relevant, in particular when the vulnerability is not totally fixed. This mitigation measure accounts for configurations, documentation, procedures that can be performed to mitigate the impact of vulnerabilities. This metric has some similarities with the Remediation Level (RL) of the temporal metrics from CVSS. Nonetheless, CROCUS, considers the number of measures that can mitigate the impact of the vulnerability, while the RL metric only distinguishes if there are workarounds, temporary or official fixes. In addition, CROCUS does not consider Environmental Metrics, since they rely on customised CVSS scores, which are based on the subjective importance of users in organisations.

### 4.3 Step #3 - Rank Vulnerabilities

CROCUS ranks vulnerabilities using the Methodical algorithm [8], as outlined in section 3.3. Fig. 1 pictures the CROCUS metrics into the benefits and costs criteria categories for ranking. It also highlights the metrics proposed by CROCUS, extending the ones already provided by CVSS standards.

The output of the rank step is an ordered score *VSco*, where lower values are the most favourable, as they are close to the ideal solutions that Methodical internally considers. The list of vulnerabilities identified in the previous steps

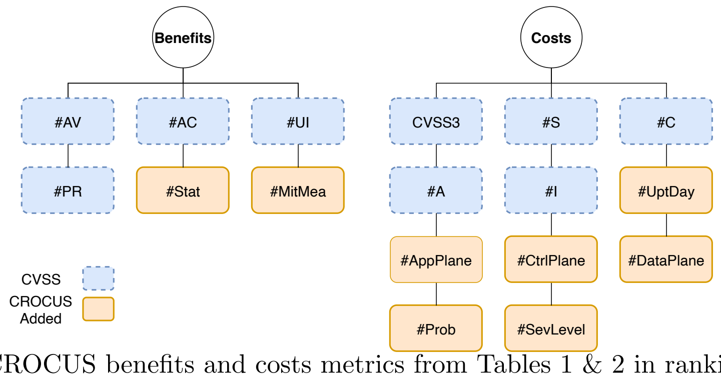

Fig. 1: CROCUS benefits and costs metrics from Tables 1 & 2 in ranking step.

are ranked considering the multiple attributes, which derived from CVSS and introduced in CROCUS.

In CROCUS, each vulnerability of an SDN controller represents an alternative (in the terminology of MADM), and through the application of the ranking step, an ordered list of vulnerabilities is obtained, requiring a final step to filter/select the SDN controller.

## 4.4 Step #4 - Selection of SDN Controller

The final selection of the SDN controller relies on a heuristic approach that combines the vulnerabilities of the diverse SDN controllers ($VSo$), the solvability ratio ($rCri$), and other relevant security features ($SeSo$) that are considered in three categories: Design, Interface and Security Services, as summarised in Table 3. The security features are evaluated in a three-level scale, where 0-no support, 1-partial support and 2-full support. Such classifications rely on the official and available documentation of the controllers. The provided features rely on the features that are reported in the literature as being mandatory or highly recommended for SDN controllers in enterprise environments [3–6]. Such features are then ranked using the Methodical algorithm.

The solvability ratio - $rCri$ corresponds to a composite metric assessing the reaction that the community/entity responsible for a SDN controller has to solve critical vulnerabilities, in a time period. $rCri_c$, regarding controller $c$, is determined as per Eq. 1 and considers the average of days to solve critical vulnerabilities - $uptDayCri_c$, which has a certain number of critical vulnerabilities - $nVulCri_c$. A critical vulnerability has a CVSS score above 4 (i.e. CVSS3 $\geq$ 4). The $TOTuptDayCri$ corresponds to the sum of the averages days to solve critical vulnerabilities in all $N$ controllers.

$$rCri_c = \frac{\frac{\sum uptDayCri_c}{nVulCri_c}}{TOTuptDayCri}, with\ c \in [1, N] \tag{1}$$

Lower values of $rCri$ are more interesting, since they represent that a vulnerability takes less time to be solved/addressed.

Table 3: CROCUS functional metrics per category with values {0,1,2}

| C. | Metric | Description |
|---|---|---|
| Design | Security Resources #**RS** | Mechanisms to protect networks resources. |
| | Policy Conflicts resolution #**IRP** | Schemes to ensure that policies to not introduce opposite behaviours. |
| | Multiple instances #**IMC** | Clustering support to avoid SPoF. |
| | Protection of Inter-cluster msgs #**PIC** | In cluster mode ensure that the information of master is verified [5]. |
| | Secure Storage #**AS** | Information is stored in a secure way and with integrity verification schemes. |
| Interface | Secure communication interface #**CCS** | Communications with the SDN controller are secured (i.e. TLS). |
| | GUI/REST API security #**API** | Interfaces exposing the SDN controller, in particular the NBI are secured (e.g. TLS). |
| Services | IDPS integration #**IDPS** | Integration with Intrusion Detection Prevention Systems is performed in a seamless mode. |
| | AAA support #**AAA** | Resources' usage requires authentication. |
| | Resource Monitoring #**MR** | Resources are monitored (topology changes). |
| | Logs and audit #**AuD** | Information of SDN planes is logged. |

The $CROCUS_c$, for controller $c$, corresponds to the result of the heuristic enabling the objective selection of the SDN controller. As per Eq. 2, $CROCUS_c$ combines the solvability ratio - $rCri$, the vulnerabilities score - $VSo$, and the score of the security functions - $SeSo$ employing a utility function with a weighted sum of these metrics. The average of vulnerabilities score - $VSo$ considers the number of vulnerabilities identified $nV_c$, while the average of the score of security functionalities - $SeSo$ is determined considering the number of secure functionalities $nS_c$. Weights for the solvability metric $wR$, for the score of the vulnerabilities $wV$ and for the score of security functionalities $wS$ are configurable to allow modelling user preferences, with $wR + wV + wS = 1$.

$$CROCUS_c = rCri_c * wR + \frac{\sum VSo_c}{nV_c} * wV + \frac{\sum SeSo_c}{nS_c} * wS \tag{2}$$

The goal is to achieve lower values of $CROCUS_c$, as they represent more efficient solvability ratios and scores close to the ideal values.

## 5 A Case Study: SDN Controller for 5G Networks

This section describes a use case with the selection of a SDN controller in heterogeneous networks.

### 5.1 Scenario

A scenario consisting of heterogeneous technologies can be considered as a study use case [2]. Such technologies are employed to allow vehicles to communicate with the infrastructure. The mmWave or fiber can be employed to allow the connection between infrastructure nodes (e.g. Road Side Units - RSUs), while vehicles can communicate with the infrastructure (e.g. RSU) using 5G radio. The infrastructure can also support other services, like multimedia services with caching mechanisms or served through content delivery networks, to enhance the quality of video. In both scenarios, SDN is employed to facilitate the management of network policies, to enable the chaining of service functions (SFCs) and to facilitate the interconnection with orchestration platforms for VNFs.

In such scenario, performance metrics like Round-Trip-Time (RTT), throughput, bandwidth and burst rate are considered by the literature [25]. Nonetheless, the security features summarised in Table 3 are also relevant. For instance, the support of clustering, the support of secure resources (i.e. validate and enforce use of privileges of applications), the support for resource monitoring and the support for policy conflicts resolution are features that enhance the security in aforementioned scenarios. Given such security constraints, the choice of SDN controllers narrows to ONOS and OpenDayLight [4, 25]. SDN controllers like Ryu or Pox are not considered as feasible controllers [7, 24] since they lack support for clustering or do not have support for demanding scenarios.

## 5.2 SDN controllers

The SDN controllers considered in the study case include ONOS and OpenDayLight for the study period between the years 2014 and 2020.

Table 4: OpenDayLight vulnerabilities

| **CVE-id** | #Stat | Pub.Date | Upd.Date | #uptDay | **CVSS3** | #AV | #AC | #PR | #S | #UI | #C | #I | #A | #MitMea | #AppPlane | #CtrlPlane | #DataPlane | #SevLevel | #Prob |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CVE-2018-10898 | 1 | 30/07/18 | 09/10/19 | 436 | 3 | 2 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 1 | 0 | 1 | 0 | 3 | 3 |
| CVE-2018-1132 | 1 | 20/06/18 | 09/10/19 | 476 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 1 | 0 | 1 | 0 | 3 | 3 |
| CVE-2018-1078 | 1 | 16/03/18 | 09/10/19 | 572 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 0 | 1 | 0 | 3 | 3 |
| CVE-2017-1000411 | 1 | 31/01/18 | 03/10/19 | 610 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 2 | 3 |
| CVE-2017-1000406 | 1 | 30/11/17 | 20/12/17 | 20 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 2 | 3 |
| CVE-2015-1778 | 1 | 27/06/17 | 05/07/17 | 8 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 1 | 1 | 0 | 3 | 3 |
| CVE-2014-8149 | 1 | 27/06/17 | 03/07/17 | 6 | 3 | 1 | 1 | 1 | 1 | 0 | 2 | 2 | 2 | 2 | 1 | 1 | 0 | 3 | 3 |
| CVE-2017-1000361 | 1 | 24/04/17 | 03/10/19 | 892 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 1 | 0 | 2 | 3 |
| CVE-2017-1000357 | 1 | 24/04/17 | 02/10/19 | 891 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 1 | 0 | 2 | 3 |
| CVE-2016-4970 | 1 | 13/04/17 | 14/02/21 | 1403 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 1 | 0 | 2 | 3 |
| CVE-2015-1612 | 1 | 04/04/17 | 11/04/17 | 7 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 2 | 0 | 1 | 1 | 2 | 3 |
| CVE-2015-1611 | 1 | 04/04/17 | 11/04/17 | 7 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 2 | 0 | 1 | 1 | 2 | 3 |

$uptDayCri_{ODL} = 1056/3 = 352$ $rCri_{ODL} = \frac{352}{352+155.54} \approx 69.35\%$

Table 4 summarises the vulnerabilities of ODL in the study period, illustrating a total of 12 vulnerabilities with high and critical risk. The ODL vulnerabilities are reported considering the VSS and CROCUS proposed metrics (recall Table 1 and Table 2).

Table 5: ONOS vulnerabilities

| **CVE-id** | #Stat | Pub.Date | Upd.Date | #uptDay | **CVSS3** | #AV | #AC | #PR | #S | #UI | #C | #I | #A | #MitMea | #AppPlane | #CtrlPlane | #DataPlane | #SevLevel | #Prob |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CVE-2020-35604 | 1 | 21/12/20 | 25/12/20 | 4 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 0 | 3 | 3 |
| CVE-2019-16302 | 0 | 20/02/20 | 25/02/20 | 5 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 1 | 0 | 2 | 2 |
| CVE-2019-16301 | 0 | 20/02/20 | 25/02/20 | 5 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 1 | 0 | 2 | 2 |
| CVE-2019-16300 | 0 | 20/02/20 | 25/02/20 | 5 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 1 | 0 | 2 | 2 |
| CVE-2019-16299 | 0 | 20/02/20 | 25/02/20 | 5 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 1 | 0 | 2 | 2 |
| CVE-2019-16298 | 0 | 20/02/20 | 25/02/20 | 5 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 1 | 0 | 2 | 2 |
| CVE-2019-16297 | 0 | 20/02/20 | 25/02/20 | 5 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 1 | 0 | 2 | 2 |
| CVE-2019-11189 | 0 | 20/02/20 | 28/02/20 | 8 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 0 | 1 | 1 | 2 | 3 |
| CVE-2020-8495 | 1 | 30/01/20 | 06/02/20 | 7 | 3 | 1 | 2 | 1 | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 3 | 2 |
| CVE-2020-8494 | 1 | 30/01/20 | 06/02/20 | 7 | 3 | 1 | 1 | 1 | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 3 | 2 |
| CVE-2019-18418 | 0 | 24/10/19 | 29/10/19 | 5 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 0 | 1 | 1 | 0 | 3 | 2 |
| CVE-2019-12587 | 1 | 04/09/19 | 24/08/20 | 355 | 3 | 2 | 1 | 0 | 1 | 0 | 2 | 2 | 0 | 1 | 0 | 1 | 1 | 2 | 3 |
| CVE-2019-15571 | 1 | 26/08/19 | 03/09/19 | 8 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 0 | 3 | 3 |
| CVE-2019-1010234 | 1 | 22/07/19 | 25/07/19 | 3 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 0 | 3 | 3 |
| CVE-2019-1010245 | 1 | 19/07/19 | 25/07/19 | 6 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 1 | 1 | 0 | 3 | 2 |
| CVE-2019-13624 | 1 | 16/07/19 | 19/07/19 | 3 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 1 | 1 | 0 | 3 | 2 |
| CVE-2018-15868 | 1 | 21/06/19 | 24/06/19 | 3 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 0 | 3 | 2 |
| CVE-2018-1000616 | 1 | 09/07/18 | 04/09/18 | 57 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 0 | 1 | 0 | 3 | 2 |
| CVE-2018-1000614 | 1 | 09/07/18 | 04/09/18 | 57 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 0 | 1 | 0 | 3 | 2 |
| CVE-2018-11316 | 1 | 03/07/18 | 11/09/18 | 70 | 4 | 1 | 1 | 0 | 2 | 1 | 2 | 2 | 2 | 1 | 0 | 1 | 1 | 3 | 4 |
| CVE-2018-11314 | 1 | 03/07/18 | 11/09/18 | 70 | 4 | 1 | 1 | 0 | 2 | 1 | 2 | 2 | 2 | 1 | 0 | 1 | 1 | 3 | 4 |
| CVE-2018-1000155 | 1 | 24/05/18 | 03/10/19 | 497 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 0 | 1 | 1 | 3 | 3 |
| CVE-2014-8129 | 1 | 01/03/18 | 06/04/18 | 36 | 3 | 1 | 1 | 0 | 1 | 1 | 2 | 2 | 2 | 1 | 0 | 1 | 0 | 3 | 2 |
| CVE-2018-5452 | 1 | 07/03/17 | 18/09/20 | 926 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 1 | 1 | 3 | 2 |
| CVE-2017-13763 | 1 | 29/08/17 | 03/10/18 | 765 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 2 | 0 | 1 | 1 | 2 | 2 |
| CVE-2015-7516 | 1 | 24/08/14 | 30/08/17 | 1102 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 2 | 0 | 1 | 1 | 3 | 2 |
| CVE-2017-1000081 | 1 | 17/07/17 | 07/12/20 | 1239 | 4 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 1 | 1 | 0 | 3 | 3 |
| CVE-2017-1000080 | 1 | 17/07/17 | 07/12/20 | 1239 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 2 | 1 | 1 | 0 | 3 | 2 |
| CVE-2017-1000079 | 1 | 17/07/17 | 07/12/20 | 1239 | 3 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 2 | 1 | 1 | 0 | 3 | 3 |

$uptDayCri_{ONOS} = 2022/13 \approx 155.54$ $rCri_{ONOS} = \frac{155.54}{352+155.54} \approx 30.65\%$

It should be noticed that vulnerabilities below CVSS3 exist for the study period, but they have even omitted in the CROCUS evaluation, as they are not relevant for a final decision, and also to avoid introducing more bias in the selection process. The average update time for the critical vulnerabilities relies $\approx 352 days$, which is high when compared with the average update time of ONOS. Indeed, one can observe that the vulnerabilities of ODL are lower when compared to ONOS, but the solvability ratio is higher $rCri_{ODL} \approx 69.35\%$. In addition, ODL has also the vulnerability with high risk that took more time to be solved, when compared to ONOS. The average resolution time for the vulnerabilities of ODL relies ≈ 444 days and none of the vulnerabilities found in the study period are in the status open or without information.

Table 5 depicts 30 security vulnerabilities for the study period with ≈ 26.67% with the status to be solved or without additional information. For instance, the CVE-2019-16302 has not yet available a solution, only mitigation measures [26]. ONOS has a total of 13 critical vulnerabilities, and in the same line of ODL, vulnerabilities with risk below high exist but are not included in the evaluation of CROCUS. The average time for vulnerability resolution relies in values ≈ 266.76 *days* which is lower than the one observed in ODL. Indeed the solvability ratio is lower in ONOS $rCri_{ONOS} \approx 30.65\%$, which means that ONOS has an active development process and that new features are being incorporated, since the vulnerabilities more recent when compared to ODL (after the year 2019). OpenDayLight does not disclose vulnerabilities information publicly since the end of 2018.

Another aspect refers to the type of vulnerabilities, which are reported as medium risk, and they result from events and interactions between components (e.g. bugs) of the ONOS controller. The main issue, is that these kind of vulnerabilities are harder to detect, since they require a deep knowledge of the controller and its internals. But on another perspective such kind of vulnerabilities are more interesting to attackers as they exploit is difficult but is also harder to detect. As stated, the identification of the vulnerabilities, the possible mitigation measures, through the analysis of available documentation, is one the contributions of this paper, since the results of such analysis are included in formulation of the $CROCUS_c$ objective selection.

### 5.3 Ranking Vulnerabilities

This section presents the results of applying the Methodical algorithm with different weights sets. To enable the comparison of the proposed approaches a set of weights has been considered, as summarised in Table 6. The *uniform* set considers the same relevance for the criteria within the respective category, while the *SDN* puts more emphasis on the set of SDN priorities the metrics related with SDN (e.g. #AppPlane, #CtrlPlane and #DataPlane) and the resolution of vulnerabilities (i.e. #Stat and #MitMea metrics). The *CVSS* and *CROCUS* sets aim to intensify the associated metrics, with the former putting emphasis on the CVSS standards (e.g. CVSS3 score), while the later mainly considered the metrics introduced in the CROCUS approach.

CROCUS also considers the possibility of establishing more preference to the benefits or costs metrics categories/groups. The *equalGrp* establishes the same importance (50%) for benefits and costs categories, while the *benefGrp* puts

Table 6: Sets of metrics weights

| Weight Set in (%) | Benefits | | | | | | Costs | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | #UI | #AV | #PR | #AC | #Stat | #MitMea | #S | #C | #I | #A | CVSS3 | #SevLevel | #Prob | #AppPlane | #CtrlPlane | #DataPlane | #UptDays |
| uniform | 16,67 | 16,67 | 16,67 | 16,67 | 16,67 | 16,67 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 |
| SDN | 5,00 | 10,00 | 10,00 | 25,00 | 25,00 | 25,00 | 5,00 | 5,00 | 5,00 | 5,00 | 5,00 | 5,00 | 5,00 | 20,00 | 20,00 | 20,00 | 5,00 |
| CVSS | 20,00 | 20,00 | 20,00 | 20,00 | 10,00 | 10,00 | 15,00 | 15,00 | 15,00 | 15,00 | 20,00 | 5,00 | 5,00 | 2,50 | 2,50 | 2,50 | 2,50 |
| CROCUS | 5,00 | 5,00 | 5,00 | 5,00 | 40,00 | 40,00 | 2,50 | 2,50 | 2,50 | 2,50 | 2,50 | 15,00 | 15,00 | 15,00 | 15,00 | 15,00 | 12,50 |

emphasis on the benefits category with 75% for benefits. The *costsGrp* places 75% of relevance in costs metric category.

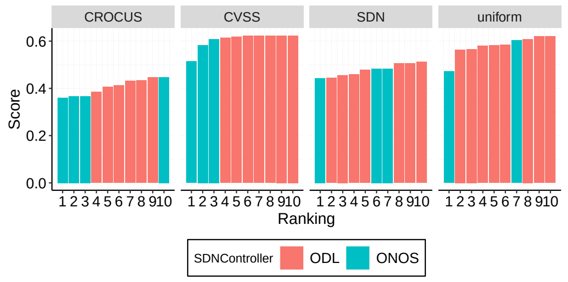

Fig. 2: TopTen per weight set and *equalGrp* weight category

Fig. 2 depicts the topTen ranking of the vulnerabilities per the set of metrics weights. The vulnerabilities associated with ONOS for all the weight sets are always placed in the first place. All the weight sets rank the *CVE-2014-8129* in first place, due its low probability and reduced impact in diverse SDN planes. Although not pictured, the weight category/group does not affect the ranking in terms of placing the vulnerabilities of ONOS in first place.

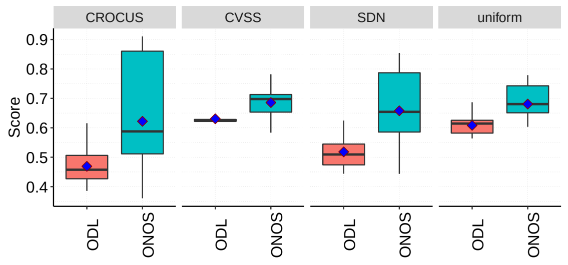

Fig. 3: Variation of vulnerability score per controller and weight set

Fig. 3 depicts the variation of vulnerability scores per controller and per set of metrics weight. The scores associated with ONOS tend to haver a higher variation, in particular in the cases with the weight set defining extreme values in the relevance of some criteria. The *CROCUS* weight set introduces higher variation, where ONOS has vulnerabilities close to the ideal (values near zero) but also more distant from the ideal solution (close to one). The difference between ONOS and ODL is also patent in such case, with the an higher variation in the mean values (represented as diamond points), with values above 0.1. The variation in ONOS is also associated with the number of vulnerabilities which is more than the double when compared to ODL.

Such results also put in evidence, that the weights to rank vulnerabilities of SDN controllers must be set to put emphasis on the metrics that are associated with the CVSS score (#AV, #I, #A, etc).

## 5.4 Selection of SDN Controller

Table 7 contains the values of the functional metrics for ODL and ONOS controllers that were determined considering Table 3 and available documentation.

Table 7: Values of functional metrics for ODL and ONOS controllers

| Controller | Design | | | | | Interface | | Service | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | #RS | #RP | #MC | #PIC | #AS | #CCS | #API | #IDS | #AAA | #MR | AuD |
| ODL | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 1 | 2 |
| ONOS | 1 | 2 | 2 | 0 | 1 | 2 | 2 | 1 | 1 | 2 | 2 |

ODL has a framework to support AAA, thus leading to the maximum classification in #RS, #AAA and #AS metrics. ONOS includes security features like the security mode, but some of these features are not well documented, being unclear the support for secure storage #AS. The Defense4All project of ODL facilitates the integration with IDPS, while ONOS does not provide documentation to perform such integration, thus the value 0 in the #IDS metric. On

Table 8: Weights of functional metrics for ODL and ONOS controllers

| Weight Set | Design | | | | | Interface | | Service | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| (%) | #RS | #RP | #MC | #PIC | #AS | #CCS | #API | #IDS | #AAA | #MR | AuD |
| uniform | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 | 9,09 |
| Design | 17,50 | 17,50 | 17,50 | 17,50 | 15,00 | 2,50 | 2,50 | 2,50 | 2,50 | 2,50 | 2,50 |
| Interface | 5,00 | 5,00 | 5,00 | 5,00 | 2,50 | 30,00 | 30,00 | 5,00 | 5,00 | 5,00 | 5,00 |
| Service | 5,00 | 2,50 | 2,50 | 2,50 | 2,50 | 2,50 | 2,50 | 20,00 | 20,00 | 20,00 | 20,00 |

the other hand, ONOS provides support for multiple monitoring and audit solutions, while ODL only documents one approach. All the security functionalities are considered as belonging to the benefits category/group, since they provide a clear advantage in terms of security.

The weights of the security functionalities are also considered in diverse sets, as summarised in Table 8, where the *Design* puts emphasis on the design metrics, the *Interface* gives more preference to the security metrics in the controller interfaces, and the *Service* prioritises the security metrics associated with monitoring and audit support. Fig. 4 outlines the best performance of ODL regarding

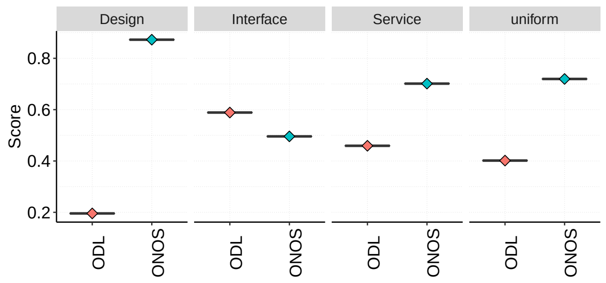

Fig. 4: Variation of score per controller and weight set

the security functionalities, in the majority of the weight sets. ONOS only surpasses ODL in the *Interface* weight set due to stronger security mechanisms for REST APIs.

The results discussed so far only focus on particular scores or metrics, considering the different sets of weights. The CROCUS aggregation score, as per Eq. 2 aggregates the solvability ratio of critical vulnerabilities - $rCric$ , the average vulnerabilities score - $VSo$ and the average score of security functionalities - $SeSo$. Fig. 5 illustrates the CROCUS score for the ODL and ONOS controllers.

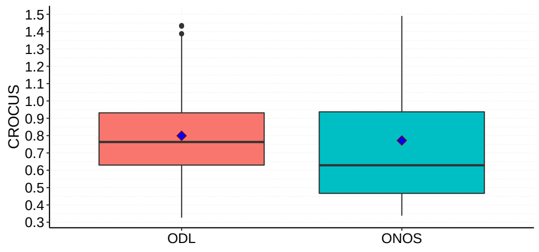

Fig. 5: CROCUS

CROCUS highlights that both controllers have the same security performance, if considering the mean values, represented as blue diamonds, the minimum values and the third quartile (75th percentile). The values for ODL and ONOS controllers are similar with low differences between them. The ONOS controller has higher values for CROCUS ($\approx$ 1.5), while ODL has around $\approx$ 1.4 (with some outliers). Nonetheless, the variation between the minimum value and the first quartile (25 th percentile) is lower for ONOS ($\approx 0.45$, while ODL has $\approx 0.65$), which holds the tendency of ONOS to have values near zero. Thus, the best values for CROCUS metric, herein proposed. CROCUS assesses ONOS as the most suitable choice the SDN controller.

## 6 Conclusions

CROCUS has been employed in a scenario considering multiple technologies and stringent requirements in terms of performance and security, to enable an objective selection of SDN controllers. CROCUS can also assist CISO and other security managers in the decision process of selecting the most suitable SDN controllers, focusing on security aspects, without disregarding performance constraints. CROCUS is simple to be applied in design, production phases and aggregates information publicly available. In particular, information regarding the vulnerabilities affecting controllers, as well their mitigation measures.

Our next steps include the integration of CROCUS in SDN controllers to enhance security support in real-time by enabling the configuration of multiple instances in the clustering process and the deployment of security policies to mitigate such kind of attacks.

## Acknowledgements

This work was funded by the European Regional Development Fund (FEDER), through the Regional Operational Programme of Centre (CENTRO 2020) of the Portugal 2020 framework and FCT under the MIT Portugal Program [Project SNOB-5G with Nr. 045929 (CENTRO-01-0247-FEDER-045929)]

## References

1. M. Taneja and A. Davy, "Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm," in *IFIP/IEEE IM*. IEEE, 2017.
2. A. Cohen, H. Esfahanizadeh, B. Sousa, J. P. Vilela, M. Luís, D. Raposo, F. Michel, S. Sargento, and M. Médard, "Bringing network coding into SDN: A case-study for highly meshed heterogeneous communications," *CoRR*, vol. abs/2010.00343, 2020.
3. S. Scott-Hayward, "Design and deployment of secure, robust, and resilient SDN controllers," in *IEEE NetSoft*, 2015.
4. M. P. Singh and A. Bhandari, "New-flow based DDoS attacks in SDN: Taxonomy, rationales, and research challenges," *Comp. Comm.*, vol. 154, no. February, 2020.
5. A. R. Abdou, C. van Oorschot, and T. Wan, "A framework and comparative analysis of control plane security of SDN and conventional networks," *arXiv*, 2017.
6. S. Yoon, S. Shin, P. Porras, V. Yegneswaran, H. Kang, M. Fong, B. O'Connor, and T. Vachuska, "A Security-Mode for Carrier-Grade SDN Controllers," in *ACM ACSAC*. ACM, dec 2017.
7. L. Mamushiane, A. Lysko, and S. Dlamini, "A comparative evaluation of the performance of popular SDN controllers," *IFIP Wireless Days*, April 2018.

8. B. Sousa, K. Pentikousis, and M. Curado, "Methodical: Towards the next generation of multihomed applications," *Computer Networks*, vol. 65, 2014.
9. S. Baghla and S. Bansal, "VIKOR MADM Based Optimization Method For Vertical Handover In Heterogeneous Networks," *Advances in Systems Science and Applications*, vol. 18, no. 3, 2018.
10. R. Khondoker, A. Zaalouk, R. Marx, and K. Bayarou, "Feature-based comparison and selection of Software Defined Networking controllers," in *WCCAIS*, 2014.
11. M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating Security and Systems Engineering.* Wiley, 2005.
12. R. K. Arbettu, R. Khondoker, K. Bayarou, and F. Weber, "Security analysis of OpenDaylight, ONOS, Rosemary and Ryu SDN controllers," in *17th Networks Symposium*, 2016.
13. Microsoft, "The STRIDE Threat Model," 2009.
14. NIST ITL National Vulnerability Database, "Common Vulnerability Scoring System Calculator," https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator.
15. M. C. Araújo, B. Sousa, M. Curado, and L. F. Bittencourt, "CMFog: Proactive Content Migration Using Markov Chain and MADM in Fog Computing," in *2020 IEEE/ACM UCC*, 2020.
16. B. Sousa, K. Pentikousis, and M. Curado, "Optimizing quality of resilience in the cloud," in *2014 IEEE Global Communications Conference*, 2014.
17. L. Zhu, M. M. Karim, K. Sharif, C. Xu, F. Li, X. Du, and M. Guizani, "Sdn controllers: A comprehensive analysis and performance evaluation study," *ACM Comput. Surv.*, vol. 53, no. 6, Dec. 2020.
18. L. Schehlmann, S. Abt, and H. Baier, "Blessing or curse? Revisiting security aspects of Software-Defined Networking," in *CNSM*, 2014.
19. Y. Xu and Y. Liu, "DDoS Attack Detection under SDN Context," in *IEEE INFOCOM 2016.* IEEE Press, 2016.
20. O. N. F. (ONF), "SDN Architecture 1.0 Overview," November 2014.
21. D. Sanvito, D. Moro, M. Gulli, I. Filippini, A. Capone, and A. Campanella, "ONOS Intent Monitor and Reroute service: enabling plug&play routing logic," in *2018 4th IEEE NetSoft.* IEEE, jun 2018.
22. FIRST, "Common vulnerability scoring system version 3.1: User guide," https://www.first.org/cvss/user-guide, 2021.
23. B. Martini, M. Gharbaoui, D. Adami, P. Castoldi, and S. Giordano, "Experimenting SDN and Cloud Orchestration in Virtualized Testing Facilities: Performance Results and Comparison," *IEEE TNSM*, vol. 16, no. 3, 2019.
24. S. Hamid, N. Zakaria, and J. Ahmed, "ReCSDN: Resilient Controller for Software Defined Networks," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 8, 2017.
25. S. Badotra and S. N. Panda, "Evaluation and comparison of OpenDayLight and open networking operating system in software-defined networking," *Cluster Computing*, 2019.
26. B. E. U. et al., "Automated Discovery of Cross-Plane Event-Based Vulnerabilities in Software-Defined Networking," in *NDSS Symposium.* Internet Society, February 2020.