La sécurité dans le SDN

Entre innovation et carences de sécurité

Aghiles DJOUDI
Raouf KERKOUCHE
Adlène AIT BELKACEM
Lyssia BENMESSAOUD
Asma DJEKHABA

***Abstract*— Le SDN a connu ces 10 dernières années un intérêt sans précédent, surtout dans un contexte où les réseaux traditionnels montrent encore davantage leur faiblesse. Toutefois, cette architecture innovante n'est pas sans risque et elle est même sujette à de nombreuses menaces de sécurité. Notre article traitera des différentes menaces et défis à relever, il proposera aussi différentes solutions pour contrer ces menaces.**

I. INTRODUCTION

La limitation des architectures réseaux traditionnelles devient forte, les réseaux actuels ne peuvent intégrer une nouvelle solution sans effectuer d'étude de performances, ces derniers ne supportent pas un passage à l'échelle. A titre d'exemple, si l'on veut intégrer une nouvelle application dans une infrastructure réseau, il faut au préalable faire une étude sur la bande passante résiduelle, dans le but de ne pas altérer les performances et la stabilité du réseau. Ainsi, le temps d'introduction au marché de nouvelles solutions est beaucoup trop long et les techniques de provisionnement ne sont pas assez rapides, la nouvelle technologie réseau défini par logiciel (SDN) a émergé pour pallier à ces limitations, Le marché s'est rapidement approprié cette technologie comme l'ensemble de solutions/architectures permettant de supprimer les frontières existantes entre les mondes des applications et du réseau. Comme avec tout grand changement d'infrastructure, il faut avoir conscience que le réseau ne se défendra pas tout seul, quelques solutions de sécurité dédiées vont devoir voir le jour, dans ce projet nous allons nous pencher sur la sécurité des réseaux définis par logiciels, l'objectif étant de détecter les risques et de les détailler. Dans la première partie de ce papier, nous allons donner une vue d'ensemble sur le SDN, en mettant l'accent sur l'objectif de cette technologie, et l'architecture de normalisation ONF (*Open Network Fundation*) Dans la seconde partie, nous allons énumérer les différentes failles de sécurité subsistantes dans le SDN, tout en présentant les solutions mises en place pour éliminer chaque faille. Nous finaliserons ce papier, par une conclusion, en dressant un bilan sur la sécurité du SDN. De plus, nous allons tenter d'explorer de nouvelles pistes qui peuvent s'avérer intéressantes pour le futur de la sécurité dans le SDN.

II. ETAT DE L'ART

A. Le SDN qu'est-ce que c'est ?

Le réseau défini par logiciels ou Software Defined Network (SDN) en anglais, est un concept réseau qui permet de séparer le plan de contrôle du plan de données. En effet dans les architectures réseaux traditionnelles, les tables de transfert (forwarding) sont calculées de façon distribuée au niveau de chaque équipement réseau (routeur ou commutateur), le routage ou la commutation se décident au niveau de ces mêmes équipements. Dans le SDN, les tables de transfert sont calculées au niveau d'un nouvel équipement appelé le contrôleur, celui-ci, exploite sa capacité à avoir une vision de la topologie entière du réseau afin de décider de la façon dont les paquets seront routés.

B. Son objectif

L'objectif du SDN est clairement de simplifier les processus de déploiement, d'administration et de gestion des réseaux, en exploitant certaines technologies existantes telles que la virtualisation et le cloud computing, il est globalement reconnu aujourd'hui comme une architecture permettant d'introduire une notion de programmabilité des réseaux [1] [2], un exemple concret des révolutions apportées par le SDN, serait la possibilité d'exécuter n'importe quel système d'exploitation sur n'importe quelle plateforme matériel, indépendamment du caractère propriétaire de celle-ci.

C. L'architecture ONF

Comme toute technologie émergente, voulant se frayer un chemin au milieu des plus grands constructeurs et fournisseurs d'équipements réseaux, le SDN a du être soumis à une phase de normalisation, et c'est à l'ONF (Open Network foundation) qu'incombe cette tâche, l'architecture proposée par l'ONF constituée de trois couches est décrite par la figure suivante :

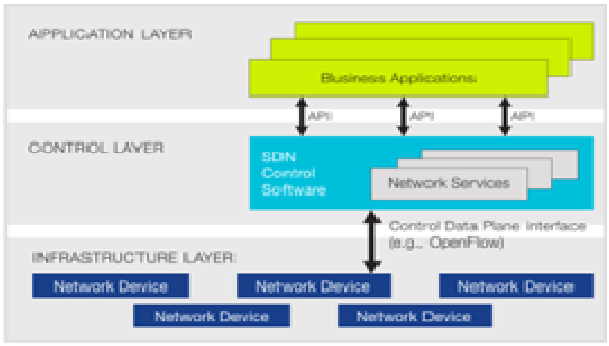

Fig. 1. Architecture ONF [3]

1) La couche application

Cette couche est constituée d'un ensemble d'applications permettant de configurer, gérer, sécuriser, et optimiser les ressources du réseau de façon dynamique et automatisée.
La couche application introduit la programmabilité des applications en faisant descendre vers le contrôleur à travers l'API nord tout les éléments nécessaires pour la gestion du réseau.

2) Couche contrôle

Cette couche permet de centraliser l'intelligence du réseau au niveau du contrôleur, ce dernier donne les instructions nécessaires au plan de données pour l'acheminement des paquets, en exploitant sa capacité à avoir une vue sur la topologie entière du réseau à travers l'API sud.

3) Couche infrastructure

La couche infrastructure est constituée par l'ensemble des périphériques et équipements réseaux physiques et virtuels permettant la commutation et le routage des paquets.

4) L'API nord

Cette API constitue l'interface entre la couche application et la couche contrôle (figure 2 ci-dessous) et permet de de faire transiter les informations nécessaires à la configuration du réseau en provenance des applications vers le contrôleur, le protocole de base pour réaliser ces transmissions est fondé sur l'API REST (*Representative State Transfert*), ou l'API Java [2].

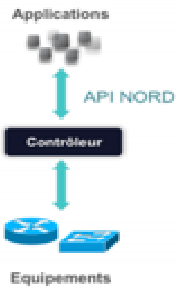

Fig. 2. API nord

5) L'API sud

Elle permet de faire communiquer le plan de contrôle avec le plan infrastructure (figure 3 ci-dessous), dans un sens elle remonte les informations statistiques nécessaires au contrôleur pour cartographier la topologie du réseau et dans l'autre sens, elle envoie les instructions du contrôleur vers le plan de données. Il existe de nombreux protocoles pour cette API, nous pouvons citer OpenFlow, LISP, SNMP…

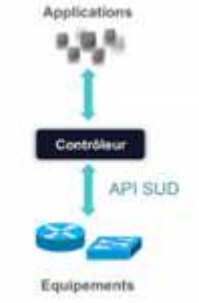

Fig. 3. L'API sud

6) API ouest et est

L'interface donne la possibilité aux contrôleurs du même sous-réseau de communiquer entre eux, afin de prendre des décisions communes. L'interface ouest quant à elle, donne la possibilité aux contrôleurs de communiquer entre eux même s'ils ne font pas partie du même sous-réseau.

D. Exemple de fonctionnement du SDN

Afin d'illustrer le fonctionnement du SDN de façon simple nous allons traiter un exemple pratique, en utilisant l'application HP Network Protector (HPNP) qui permet d'intercepter les interrogations DNS et les vérifier à travers sa base de données pour s'assurer qu'elle ne pointe pas vers des sites web malveillants, cette implémentation utilise OpenFlow, l'application HPNP est installée dans le contrôleur SDN et utilise l'API nord java pour communiquer avec le contrôleur, ce dernier utilise OpenFlow afin de mettre à jour les switch pour intercepter les requêtes DNS, si un ordinateur sur le réseau envoie une requête DNS vers un site web malveillant, quand l'interrogation DNS du site web atteint le switch, cette dernière sera redirigée vers le contrôleur qui à son tour la redirige vers l'application, cette dernière interroge la base de données, si le site est malicieux l'application informe que le domaine n'existe pas, dans le cas où le site web n'est pas malicieux l'ordinateur aura l'accès au site, la figure suivante illustre schématiquement l'exemple :

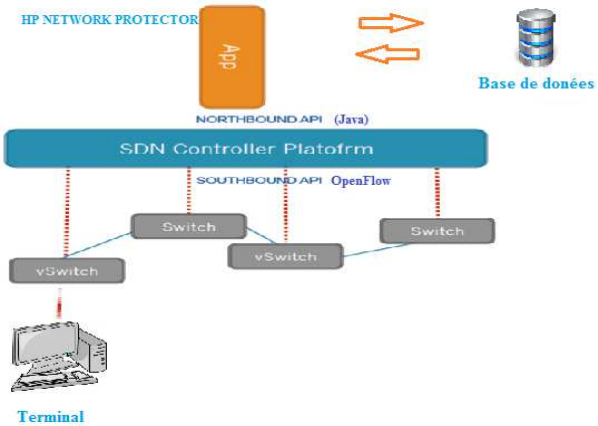

Fig. 4. Exemple de fonctionnement du SDN

III. LES SERVICES DE SÉCURITÉ DANS LE SDN :

A. *Confidentialité*

La confidentialité empêche la divulgation d'informations à des entités non autorisées. Pour assurer la confidentialité, deux méthodes communes de cryptage et de contrôle d'accès sont utilisées.
[4] Le cryptage du canal de communication entre le plan de données et le contrôleur signifie qu'un attaquant a accès au texte chiffré, mais qu'il n'est pas en mesure de récupérer le texte en clair correspondant. Par exemple, un canal chiffré peut être établi par TLS comme proposé par la spécification de commutateur OpenFlow [5].
[6] L'octroi d'accès après l'authentification via les interfaces de gestion des périphériques réseau et le contrôleur signifie que seules les entités autorisées ont accès aux structures de données. Le contrôle d'accès peut être appliqué par le système d'exploitation. Plusieurs techniques pour crypter les canaux de communication réseau et appliquer le contrôle d'accès sont déjà développées et pourraient être adaptées.

B. Authenticité

L'authenticité décrit la propriété que les entités sont réellement celles qu'ils prétendent être. Une méthode cryptographique bien connue pour assurer l'authenticité est une signature (par exemple un code d'authentification de message (MAC) pour des données en bloc). En outre, les dispositifs réseau ainsi que le contrôleur doivent échanger des clés (soit les clés secrètes pour la génération / validation d'un MAC ou les clés publiques pour les signatures asymétriques) [5]. Pour assurer la confiance entre les applications et le contrôleur, [7] propose l'utilisation d'un système autonome de gestion de confiance (par exemple [8]) pour empêcher que les applications malveillantes se lient au contrôleur pour effectuer des actions malveillantes.

C. Intégrité

L'intégrité signifie que l'information n'est pas modifiée pendant son cycle de vie. Dans les réseaux SDN, l'intégrité des règles de flux et des messages transférés entre les couches doit avant tout être assurée. L'intégrité des messages pourrait également être mise en œuvre, par exemple, par un code d'authentification de message (MAC).
On considère la question de l'intégrité en ce qui concerne les règles d'écoulement / d'acheminement comme étant essentielles dans un réseau , car des règles modifiées pourraient avoir des effets indésirables

D. Disponibilité

La disponibilité est la propriété d'accéder aux données, aux périphériques et aux services chaque fois qu'il est nécessaire. Le goulot d'étranglement évident dans notre échantillon OpenFlow réseau est le contrôleur. Si le contrôleur n'est pas disponible en raison d'une mauvaise configuration, d'une erreur technique ou d'une attaque (par exemple, attaque de déni de service), Les périphériques réseau sont uniquement en mesure d'appliquer des règles prédéfinies. Si un commutateur SDN est en panne en raison d'erreurs techniques ou d'une attaque DoS, le contrôleur peut reprogrammer dynamiquement les chemins réseau. Les solutions possibles pour atténuer une attaque DoS sont la mise en œuvre d'une limitation de débit, la réduction du délai d'attente des entrées de la table de flux ou la suppression des paquets d'une attaque DoS (voir [9]). [10] discute du placement et du nombre de contrôleurs pour obtenir la redondance.

E. Cohérence

Si différentes applications sont utilisées pour définir des règles de flux, il est possible que les règles de flux ne soient pas cohérentes. Par conséquent, un médiateur entre les applications et le contrôleur est nécessaire pour traiter les règles contradictoires.
Une mise en œuvre pour détecter et gérer les conflits des règles est FortNOX [11]. Nous évaluons l'impact des règles conflictuelles comme étant critiques pour les réseaux conventionnels comme pour les réseaux SDN, Conduisent à un comportement réseau imprévisible.

IV. LES DEFIS DE SECURITE DU SDN :

La séparation des plans et l'agrégation de la fonctionnalité du plan de contrôle à un système centralisé (par exemple, un contrôleur OpenFlow) peuvent être fondamentales pour les réseaux futurs ; Cependant, elle ouvre également de nouveaux défis de sécurité. Par exemple, les canaux de communication entre des plans isolés peuvent être ciblés. En effet, en masquant un plan, l'attaquant peut attaquer l'autre plan. Le plan de contrôle est plus susceptible aux attaques de sécurité, et précisément aux attaques DoS et DDoS, en raison de sa nature visible. Le contrôleur SDN peut devenir le seul point d'échec et rendre ainsi le réseau entier en panne dans le cas où il y'a un compromis de sécurité. La visibilité des ressources réseau est d'une importance primordiale dans le SDN, mais ces ressources ne doivent pas être visibles par toutes les applications ou par celles qui ne sont pas concernées.

La liste des défis en matière de sécurité dans le SDN devrait croître avec le déploiement progressif des technologies SDN. Afin de tirer pleinement parti de SDN, ces défis doivent être mis en évidence afin que les mesures de sécurité appropriées peuvent être prises de manière proactive. Par conséquent, les défis de sécurité et les menaces existant dans SDN sont abordés dans cette section et elles sont résumés dans le Tableau I ci-dessous [12]. D'un point de vue principal, les vulnérabilités de sécurité dans le SDN sont concentrées autour des trois domaines principaux : applications, plan de contrôle et plan de données. Par conséquent, les défis de sécurité et les menaces existant dans les trois plans (ou couches SDN) sont décrits ci-dessous.

A- Défis de sécurité au niveau du plan d'application :

SDN a deux propriétés principales qui constituent la base de l'innovation en réseau d'une part, et la base des défis de sécurité d'autre part. Tout d'abord, la capacité de contrôler un réseau par logiciel, et deuxièmement, la centralisation de

Plan du SDN	Type de Menaces	Description
Plan d'Application	Manque d'Authentification et d'Autorisation	Pas de mécanismes d'authentification et d'autorisation convaincants pour les applications et plus de menaces en cas de grand nombre d'applications tierces.
	Insertion de règles de flux frauduleuses	Les applications malicieuses et compromises peuvent générer de fausses règles de flux et il est difficile de vérifier si une application est compromise.
	Manque de Contrôle d'Accès et Imputabilité	Difficile à implémenter le contrôle d'accès et l'imputabilité dans les applications de tierces parties et les applications imbriquées qui consomment les ressources réseau.
Plan Contrôle	Attaques DoS	La nature visible, l'intelligence centralisée et les ressources limitées du plan de contrôle sont les raisons principales de l'attirance des attaques DoS.
	Accès non autorisé au contrôleur	Pas de mécanismes convaincants pour imposer le contrôle d'accès aux applications.
	Scalabilité et disponibilité	La centralisation d'intelligence dans une entité aura très probablement des défis de scalabilité et de de disponibilité.
Plan Données	Règles de flux frauduleuses	Le Plan de Données est passif et par conséquent plus susceptible aux règles de flux frauduleuses.
	Attaques par inondation	Les tables de flux des commutateurs OpenFlow peuvent stocker un nombre fini ou limité de règles de flux.
	Détournement de contrôleur ou compromis	Le plan de données dépend uniquement du plan de contrôle, ce qui rend la sécurité du plan de données dépendante de la sécurité du contrôleur.
	Attaques au niveau TCP	TLS est susceptible à des attaques au niveau TCP.
	Attaques Man-in-the-middle	Ceci est dû à l'utilisation optionnelle de TLS, et la complexité dans la configuration de TLS.

Tableau I : Les menaces de sécurité majeurs dans les plans du SDN.

l'intelligence du réseau dans les contrôleurs de réseau. Puisque la plupart des fonctions du réseau peuvent être mises en œuvre sous la forme d'applications SDN, les applications malveillantes si elles ne sont pas arrêtées assez tôt peuvent provoquer des dégâts sur un réseau. Par conséquent, nous décrivons les problèmes de sécurité liés aux applications SDN dans cette section.

Comme il n'existe pas de normes ou de spécifications afin de faciliter les API ouvertes aux applications pour contrôler des services et des fonctions réseau via le plan de contrôle [13], les applications peuvent poser de sérieuses menaces à la sécurité des ressources réseau, des services et des fonctions. Bien qu'OpenFlow permette le déploiement d'algorithmes de détection de sécurité basés sur le flux sous la forme d'applications de sécurité, il n'existe pas d'applications de sécurité OpenFlow convaincantes [14]. En outre, il n'existe pas d'environnements de développement convenus qui soient convaincants et de modèles de programmation de réseau ou de paradigmes. La variété et la multitude d'applications de fournisseurs et de tierces parties développées dans différents environnements de développement indépendants utilisant des modèles de programmation et des paradigmes différents pourraient créer des limitations d'interopérabilité et une collision entre les politiques de sécurité. Certains des défis de sécurités menaçants à relever et qui concerne les applications SDN sont décrits ci-dessous :

1) Authentification et Autorisation :

L'authentification des applications dans les tendances rapides actuelles de l'ingénierie logicielle et les exploits émergents des hacks sera un enjeu majeur dans les domaines du SDN. Dans OpenFlow, les applications exécutées sur le contrôleur implémentent une majorité des fonctionnalités du plan de contrôle et sont généralement développées par d'autres parties que les fournisseurs de contrôleurs. Ces applications héritent des privilèges d'accès aux ressources du réseau et de la manipulation du comportement des réseaux, la plupart du temps sans mécanismes de sécurité adéquats pour protéger les ressources du réseau contre les activités malveillantes [15]. Par conséquent, l'authentification du nombre croissant d'applications dans des réseaux programmables avec une architecture de contrôle centralisée constitue un défi de sécurité majeur.

Kreutz et al. [7] ont présenté des vecteurs de menace pour décrire les vulnérabilités de sécurité dans SDN. Il est décrit qu'il n'existe pas de mécanismes convaincants pour établir une relation de confiance entre le contrôleur et les applications dans les SDN. Par conséquent, une application malveillante peut potentiellement créer des ravages dans le réseau puisque les contrôleurs SDN fournissent des abstractions qui sont traduites en commandes de configuration pour l'infrastructure sous-jacente par les applications. De même, si un serveur d'applications qui stocke les détails des utilisateurs est compromis, les informations d'identification des utilisateurs légitimes peuvent être utilisées pour injecter des flux autorisés mais faux dans le réseau. De plus, diverses techniques existent pour certifier les périphériques réseau dans un réseau, mais il n'existe aucun mécanisme pour certifier les applications réseau. Comme la fonctionnalité du réseau dans le SDN est implémentée dans les applications, un système centralisé pour certifier les applications SDN est requis, mais il n'est pas encore disponible.

2) Contrôle d'Accès et Imputabilité :

L'imputabilité est la propriété qui garantit que les actions d'une entité sont tracées et attribuées à cette seule entité. L'imputabilité assure de pouvoir identifier, pour toutes les actions accomplies, les personnes, les systèmes ou les

processus qui les ont initiées (identification) et de garder trace de l'auteur et de l'action (traçabilité) [16].

Comme les applications implémentent la plupart des services réseau dans le SDN, des mécanismes de contrôle d'accès et d'imputabilité appropriés sont nécessaires pour assurer la sécurité d'un réseau. Pour comprendre les menaces de sécurité possibles liées au contrôle d'accès et à l'imputabilité dans le SDN, considérez les exemples d'applications suivants. Hartman et al. [17] identifient trois classes d'applications qui peuvent affecter la sécurité du réseau dans le SDN. Premièrement, les applications sensibles au réseau qui nécessitent des caractéristiques particulières du réseau telles que les caractéristiques des chemins, le coût des flux de trafic, etc. Deuxièmement, les applications qui fournissent des services au réseau tel que le contrôle d'accès ou le pare-feu et l'inspection du contenu ou les services de détection d'intrusion. Troisièmement, des services réseau packagées qui combinent des applications de la première et de la deuxième classe ou des applications demandant l'instanciation d'une autre application en tant qu'élément virtuel dans le réseau. Par conséquent, une application malveillante peut contourner le contrôle d'accès en utilisant une instance de l'application de deuxième classe.

En outre, le contrôle d'accès et l'imputabilité des applications imbriquées (par exemple, une application utilisant une instance d'une autre application) constitueront un réel défi dans le SDN. Un exemple de ces applications est mentionné dans [18], indiquant que les applications dans les SDN peuvent être SDN-aware ou SDN-unaware. Les applications SDN-aware sont capables de localiser et de communiquer directement avec les contrôleurs SDN tandis que les applications SDN-unaware communiquent indirectement avec des datagrammes d'application dans des formats spécifiques [18]. Dans ce dernier cas, une application SDN légitime mais compromise peut devenir une passerelle pour un accès non autorisé au plan de contrôle du réseau. De même, le maintien de l'imputabilité pour l'utilisation des ressources réseau par des applications imbriquées est un autre défi.

B- Défis de sécurité au niveau du plan de Contrôle :

Dans SDN, le plan de commande (par exemple, le contrôleur OpenFlow) est une entité décisionnelle centralisée. Par conséquent, le contrôleur peut être très ciblé pour compromettre le réseau ou mener des activités malveillantes dans le réseau en raison de son rôle pivot. Les principaux défis et menaces de sécurité existants dans le plan de contrôle sont décrits ci-dessous :

1) Menaces des applications :

Les applications implémentées au-dessus du plan de contrôle peuvent constituer de sérieuses menaces de sécurité pour le plan de contrôle. Généralement, la sécurité du contrôleur est un défi du point de vue de la capacité du contrôleur à authentifier les applications et autorise les ressources utilisées par les applications avec un isolement, une vérification et un suivi appropriés [17]. Ajouté à cela, il est nécessaire de séparer différentes applications en fonction de leurs implications en matière de sécurité avant d'avoir accès aux informations et aux ressources du réseau. La séparation des applications est très importante pour authentifier et autoriser distinctement les applications tierces et les applications d'opérateur ayant établi un audit pour chaque application.

Différentes applications ayant des exigences fonctionnelles différentes du contrôleur sous-jacent et du chemin de données doivent satisfaire à différentes exigences de sécurité. Certaines applications, par exemple l'application à répartition de charge, peuvent avoir besoin de statistiques de réseau telles que des octets ou des valeurs de compteur de paquets du commutateur pour effectuer l'équilibrage de charge. D'autres applications, par exemple, l'application de détection d'intrusion peuvent avoir besoin d'inspecter les champs d'en-tête de paquets. Par ailleurs, les fournisseurs et les applications tierces doivent disposer de privilèges différents pour accéder aux informations et aux ressources du réseau. Par exemple, la mise en réseau participative dans SDN est présentée dans [19] pour permettre aux utilisateurs finaux et à leurs applications de participer à la configuration du réseau. Ces types d'applications utilisateur doivent être examinés correctement avant que l'accès au réseau ne soit assuré en ayant des privilèges comparativement inférieurs à ceux des applications fournisseurs. Par conséquent, un mécanisme de mise en application de sécurité personnalisé pour divers types d'applications est requis dans l'API nord du contrôleur. De telles procédures de sécurité personnalisées basées sur le type ou les catégories d'applications n'ont pas encore été démontrées.

2) Menaces dues à la scalabilité :

Dans OpenFlow, la majeure partie de la complexité est poussée vers un contrôleur où les décisions de transfert sont prises de manière logiquement centralisée [20]. Si les contrôleurs sont tenus d'installer des règles de flux pour chaque nouveau flux dans le chemin de données, le contrôleur peut facilement devenir un goulot d'étranglement. Les auteurs dans [21] ont analysé que les implémentations des contrôleurs d'aujourd'hui ne sont pas capables de gérer le grand nombre de nouveaux flux lors de l'utilisation d'OpenFlow dans les réseaux à haut débit avec des liens de 10 Gbps. Il est décrit dans [22] que le manque de scalabilité permet aux attaques ciblées de provoquer une saturation du plan de contrôle qui a des résultats plus préjudiciables dans les SDN que les réseaux traditionnels. Par conséquent, la scalabilité du contrôleur en fait un choix favori pour les attaques DoS et DoS distribuées.

Un autre défi pour les implémentations de contrôleur actuellement disponibles est de spécifier le nombre de périphériques de transfert à gérer par un seul contrôleur pour faire face aux contraintes de retard. Si le nombre de flux sur le contrôleur augmente, il y'a une forte probabilité que le temps de séjour augmente, ce qui dépend fortement de la puissance de traitement du contrôleur. Cette limitation des capacités des contrôleurs peut conduire au seul point d'échec. Pour éviter les défis qu'un contrôleur soit le seul point de défaillance, l'utilisation de plusieurs contrôleurs est suggérée. Cependant, il est démontré que la simple utilisation de plusieurs contrôleurs dans les SDN ne peut pas protéger le réseau à partir d'un seul point d'échec. La raison en est que la charge des contrôleurs portant la charge du contrôleur défaillant peut dépasser leur

capacité et par conséquent entraînera une situation plus catastrophique, par exemple, des défaillances en cascade des contrôleurs [23].

3) Attaques DoS :

Les attaques DoS et DoS distribuées sont les défis de sécurité les plus menaçants pour le contrôleur SDN. L'attaque DoS est une tentative de rendre une ressource (réseau) indisponible pour les utilisateurs légitimes. Une attaque DoS sur SDN exploite la logique de séparation des plans de contrôle-données du SDN. Un outil de balayage de réseau est développé pour pouvoir identifier un réseau de SDN avec l'aide des temps de réponse de flux. Puisque pour chaque nouveau flux, le chemin de données interroge le contrôleur, il existe une différence dans les temps de réponse de flux pour les flux nouveaux et existants. Le scanner recueille les valeurs de temps à l'aide d'un balayage de changement de champ d'en-tête, qui scanne les réseaux en changeant les champs d'en-tête de réseau. Après avoir trouvé le réseau SDN, des demandes de flux spécialement conçues sont transmises au réseau cible, elles sont envoyées par le chemin de données au contrôleur. En augmentant le nombre de flux dans le chemin de données, le contrôleur se verra bombarder avec des demandes de configuration de flux provenant des commutateurs et par conséquent, cela provoquera éventuellement sa rupture.

4) Défis dans le plan de contrôle distribué :

Pour gérer un grand nombre et une variété de périphériques qui ne peuvent pas être gérés par un seul contrôleur SDN, plusieurs contrôleurs doivent être déployés, ce qui divisera le réseau en différents sous-domaines. Mais si le réseau est divisé en plusieurs sous-réseaux définis par logiciel, l'agrégation des informations et le maintien de règles de confidentialité différentes dans chaque sous-réseau constitueront un défi.

5) Autres défis :

Étant donné que le contrôleur est responsable de l'application de la politique à l'échelle du réseau, l'écart sémantique du contrôleur-commutateur, la distribution du contrôle d'accès prenant en charge les flux agrégés, les contrôleurs multi-entité et les contrôleurs multiples dans un seul domaine peuvent créer des conflits de configuration. Dans plusieurs infrastructures OpenFlow, l'incohérence dans les configurations du contrôleur entraînera des conflits inter-fédérés potentiels [24]. Par exemple, si l'état du réseau change, tous les contrôleurs peuvent ne pas recevoir les informations réseau en même temps. Par conséquent, certaines applications avec état telles que les pare-feu pourraient se comporter incorrectement en raison de la non-visibilité du contrôleur sur les événements passés.

C-Défis de sécurité au niveau du Plan de Données :

Dans les réseaux OpenFlow, le contrôleur OpenFlow installe des règles de flux dans les tables de flux du commutateur OpenFlow. Ces règles de flux peuvent être installées avant qu'un nouvel hôte envoie des paquets (installation de règle proactive) ou sur le premier paquet d'un nouvel hôte (installation de règle réactive). Un commutateur a un nombre limité de tables de flux dans lesquels les règles de flux sont installées selon la vue du contrôleur sur le réseau. Depuis, la capacité décisionnelle a été prise hors des commutateurs, le premier défi de sécurité avant tout est de reconnaître les véritables règles de flux et de les différencier des règles fausses ou malveillantes. Le deuxième défi est basé sur le nombre d'entrées de flux qu'un commutateur peut maintenir.

Dans OpenFlow, un commutateur doit stocker les flux jusqu'à ce que le contrôleur émette des règles de flux. Cela rend le plan de données sujet à des attaques de saturation, car il a des ressources limitées pour tamponner les flux (TCP / UDP) non sollicités.

Dans le SDN, la sécurité du plan de contrôle a des implications directes sur le plan de données. Cela signifie que si un contrôleur est compromis, le réseau entier comprenant une variété de nœuds de plan de données sera compromis. Dans les architectures partagées telles que SDN, si un commutateur ne reçoit pas d'instructions de transfert du plan de contrôle, soit en raison d'une défaillance de plan de contrôle, ou soit d'une déconnexion de plan de contrôle, le plan de données devient pratiquement hors ligne [25]. Par conséquent, la liaison commutateur-contrôleur peut être un choix favorable pour attaquer le réseau. La séparation du plan de contrôle et du plan de données peut permettre à un attaquant de manipuler les flux de façon furtive à travers la manipulation des règles OpenFlow, ce qui entraîne diverses attaques réseau actives, comme l'attaque man-in-the-middle et les attaques de trou noir.

La spécification originale d'OpenFlow définit la sécurité de couche de transport (TLS : Transport Layer Security) [26] et la sécurité de couche de transport de datagramme (DTLS : Datagram Transport Layer Security) [27] pour la communication du contrôleur-commutateur. Cependant, les dernières versions d'OpenFlow (par exemple, OpenFlow v1.3.0) rendent l'utilisation de TLS facultative. De plus, il est décrit que TLS a une plus grande barrière technique pour les opérateurs en raison de sa complexité de la configuration. Les étapes de configuration comprennent la génération d'un certificat à l'échelle du site, la génération de certificats de contrôleur et de commutateur, la signature des certificats avec la clé privée du site et l'installation de clés et de certificats corrects sur tous les périphériques. Par conséquent, de nombreux fournisseurs ont ignoré la prise en charge de TLS dans leurs commutateurs OpenFlow.

La complexité dans la configuration et l'utilisation facultative de TLS peuvent rendre le canal de contrôle vulnérable à divers types d'attaques. Benton et al. [28] décrivent comment les attaques man-in-the-middle sont plus létales dans les réseaux OpenFlow que les réseaux traditionnels en raison d'une connectivité constante et d'un manque d'authentification, car les messages qui circulent sur le canal de contrôle TCP d'OpenFlow ne sont pas forcément chiffrés. En conséquence, un attaquant peut immédiatement saisir le contrôle total de tous les commutateurs en aval et d'exécuter des attaques par écoute [28]. Ajouté à cela, il est démontré que l'utilisation de TLS ne fournit aucune protection au niveau TCP et, en tant que tel, peut être sujettes à des attaques au niveau TCP.

De plus, SDN peut permettre au trafic réseau d'être acheminé à travers un pare-feu centralisé pour sécuriser le plan de données. Cependant, la surveillance des messages entre un commutateur et un contrôleur peut prendre assez de temps, par conséquent, les ressources du commutateur sont épuisées par de faux flux conduisant à une inondation ou à un compromis d'attaque DoS Les retards dans la mise en place des règles de flux peuvent introduire des difficultés dans l'autorisation et l'authentification et l'inspection des contenus de trafic et des règles de flux par les applications de sécurité [12].

V. *LES MENACES DE SECURITE SUR LES DIFFERENTS PLANS DU SDN*

Les réseaux SDN possèdent deux propriétés qui peuvent être considérées comme des pots de miel attrayants pour les utilisateurs malveillants et une source de maux de tête pour les opérateurs de réseaux moins préparés.

Tout d'abord, la capacité de contrôler le réseau au moyen d'un logiciel (toujours sujet à des bugs et une vingtaine d'autres vulnérabilités).

Deuxièmement, la centralisation de l'intelligence du réseau dans le (s) contrôleur (s). Toute personne ayant accès aux serveurs qui hébergent le logiciel de contrôle peut potentiellement contrôler l'ensemble du réseau.

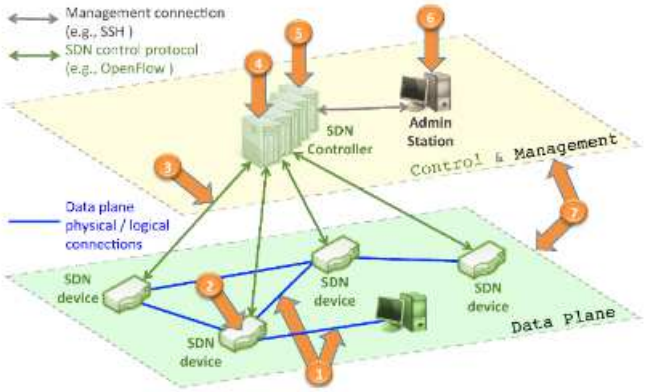

Fig. 5. Menace de sécurité dans le SDN [7]

Nous allons décrire les différentes menaces potentielles identifiées dans les différents plans des SDN.

1) Plan de données

flux de trafic faux ou falsifié, qui peut être utilisé pour attaquer les switchs et les contrôleurs. Cette menace peut être déclenchée par des périphériques défectueux (non malveillants) ou par un utilisateur malveillant. Un attaquant peut utiliser des éléments de réseau (par exemple, des switchs, des serveurs ou des ordinateurs personnels) pour lancer une attaque DoS contre les switch OpenFlow et les ressources du contrôleur. Un mécanisme d'authentification simple pourrait atténuer le problème, mais si un attaquant a le contrôle d'un serveur d'applications qui stocke les détails de nombreux utilisateurs. L'utilisation de systèmes de détection d'intrusion avec prise en charge de l'analyse des causes de l'exécution peut aider à identifier les flux anormaux. Cela pourrait être couplé avec des mécanismes pour le contrôle dynamique du comportement du switch (par exemple, les limites de débit pour les demandes de plan de commande).

Attaques sur les vulnérabilités dans les switchs, ce qui peut facilement faire des ravages avec le réseau. Un seul commutateur pourrait être utilisé pour supprimer ou ralentir des paquets dans le réseau, cloner ou dévier le trafic réseau (par exemple pour le vol de données), ou même injecter du trafic ou des demandes falsifiées pour surcharger le switch par exemple un attaquant peut saturer la capacité d'un nœud de différentes manières. Une première technique consiste à envoyer une grande quantité de trafic au nœud qui sera mise dans sa mémoire tampon en attente d'être envoyé au contrôleur. Ainsi, le nœud sera surchargé et ne pourra donc pas recevoir les instructions à suivre par le contrôleur étant donné que ce dernier ne recevra pas le trafic. Une solution à ce problème consiste à désactiver l'utilisation de la mémoire tampon. Cependant, en désactivant son utilisation, le nœud transmettra au contrôleur le trafic reçu au risque de le surcharger. La version 1.3 d'OpenFlow permet à l'administrateur de pallier ce problème en définissant les actions à appliquer à un tel trafic notamment sa suppression.

Une deuxième technique consiste à surcharger la table de transfert d'un nœud. La version 1.3 d'OpenFlow permet d'envoyer un message d'erreur au contrôleur qui tenterait d'ajouter une règle. Néanmoins aucune nouvelle règle ne sera acceptée par le nœud.

L'utilisation de mécanismes d'attestation logicielle, tels que les solutions de gestion autonome de la confiance pour les composants logiciels [29], est un facteur d'atténuation possible. Les mécanismes permettant de surveiller et de détecter les comportements anormaux des périphériques réseau peuvent également être utiles pour contrer ce type de menaces.

2) Plan de contrôle

attaques sur les communications du plan de contrôle, qui peuvent être utilisées pour générer des attaques DoS ou pour le vol de données. Comme il est bien connu dans la communauté de la sécurité, L'utilisation de TLS / SSL ne garantit pas en soi une communication sécurisée et compromet le contrôleur. Divers documents rapportent les faiblesses des communications TLS / SSL et son principal ancrage de la confiance, l'infrastructure PKI [30]. La sécurité de ces communications est aussi forte que son lien le plus faible, qui peut être un certificat auto-signé, une autorité de certification compromise ou des applications et des bibliothèques vulnérables. Par exemple, il existe de nombreuses implémentations vulnérables de SSL dans le monde entier dans le système critique mondial [31]. En outre, le modèle TLS / SSL n'est pas suffisant pour établir et assurer la confiance entre contrôleurs et switchs. Une fois qu'un attaquant a accès au plan de contrôle, il peut être capable d'agréger suffisamment de force de puissance (en termes de nombre de commutateurs sous son contrôle) pour lancer des attaques DDoS. Ce manque de confiance garantirait même la création d'un réseau virtuel de trous noirs (par exemple, en utilisant des techniques de découpage basées sur OpenFlow [32]) permettant des fuites de données pendant que le trafic normal circule. L'utilisation de modèles de confiance oligarchiques avec de multiples autorités

de certification de confiance-ancrage (par exemple, une par sous-domaine ou par instance de contrôleur) est une possibilité. Une autre solution consiste à sécuriser la communication avec la cryptographie de seuil à travers les répliques du contrôleur [33] (où le commutateur aura besoin d'au moins n actions pour obtenir un message de contrôleur valide). De plus, l'utilisation de mécanismes d'association de dispositifs dynamiques, automatisés et assurés peut être envisagée, afin de garantir la confiance entre le plan de contrôle et les dispositifs de plan de données.

Attaques et vulnérabilités dans les contrôleurs, qui sont probablement les menaces les plus graves pour les SDN. Un contrôleur défectueux ou malveillant pourrait compromettre un réseau entier. L'utilisation d'un système commun de détection d'intrusion peut ne pas être suffisante, car il peut être difficile de trouver la combinaison exacte d'événements qui déclenchent un comportement particulier et, plus important encore, de l'étiqueter comme malveillant. De même, une application malveillante peut potentiellement faire ce qu'elle veut dans le réseau, puisque les contrôleurs fournissent uniquement des abstractions qui se traduisent par l'émission de commandes de configuration vers l'Infrastructures sous-jacentes. Plusieurs techniques peuvent être utilisées, telles que la réplication (pour détecter, enlever ou masquer un comportement anormal), en employant la diversité (des contrôleurs, des protocoles, des langages de programmation, des logicielles, etc.) et la récupération (rafraîchissant périodiquement le système vers un environnement propre et fiable). Il est également important de sécuriser tous les éléments sensibles à l'intérieur du contrôleur (par exemple, les clés / secrets cryptographiques). De plus, les politiques de sécurité appliquant un comportement correcte peuvent être mises en correspondance avec ces techniques, ce qui limite les interfaces qu'une application peut utiliser et le type de règles qu'il peut générer pour programmer le réseau façon dont la certification est faite. Les techniques utilisées pour (selon la hiérarchisation basée sur la sécurité [11]).

Hidjacking, le détournement du contrôleur ou les accès non autorisés aux contrôleurs visent le principe de confidentialité de la sécurité. Des vulnérabilités dans le contrôleur peuvent avoir des conséquences qui peuvent mettre en danger tout le réseau. L'attaquant peut prendre en charge la gestion des contrôleurs mal configurés et vulnérables ainsi que la gestion du réseau. Ensuite, l'attaquant peut programmer des commutateurs dans le plan de données pour supprimer le trafic venant au contrôleur peut utiliser pour lancer des attaques sur d'autres cibles.

Attaques et vulnérabilités dans les terminaux administratifs, Les ordinateurs d'administration qui sont directement connectés au contrôleur peuvent provoquer un point d'entrée dans le réseau. Si ces ordinateurs ont une certaine vulnérabilité, l'attaquant peut utiliser ces vulnérabilités pour obtenir le contrôle des ordinateurs pour accéder facilement au contrôleur. Il devient assez facile de reprogrammer le réseau à partir d'un seul emplacement.

Manque de mécanismes pour assurer la confiance entre le contrôleur et les applications de gestion, de sorte que de la même façon que la menace numéro 3, les contrôleurs Et les applications n'ont pas la capacité d'établir des relations de confiance. La principale différence par rapport à la menace mentionnée réside dans la certifier les dispositifs réseau sont différentes de celles utilisées pour les applications. Des mécanismes de gestion de la confiance autonome pourraient être utilisés pour garantir la fiabilité de l'application pendant sa durée de vie.

Le manque de ressources fiables et réhabilitation, ce qui permettrait de comprendre la cause d'un problème détecté et de procéder à une récupération en mode rapide et sécurisé. Afin d'étudier et d'établir des faits sur un incident, nous avons besoin d'informations fiables de tous les composants et domaines du réseau. En outre, ces données ne seront utiles que si leur fiabilité (intégrité, authenticité, etc.) peut être assurée. De même, la réhabilitation nécessite des systèmes instantanés sûrs et fiables pour garantir une récupération rapide et correcte des éléments du réseau vers un état de fonctionnement connu. L'enregistrement **(logging)** et le traçage sont les mécanismes communs utilisés et sont nécessaires à la fois dans les plans de données et de contrôle. Cependant, pour être efficaces, ils doivent être indélébiles (un journal qui est garanti pour être immuable et sûr). En outre, les journaux doivent être stockés dans des environnements distants et sécurisés.

Attaques DoS, les attaques de déni de service distribué (DDoS) contre le contrôleur et les attaques d'inondation des tables de flux des commutateurs dans le plan de données ciblent le principe de disponibilité ou le service de la sécurité. Le contrôleur logiquement centralisé est l'une des principales caractéristiques du SDN. Bien que cette fonctionnalité offre une perspective globale sur le réseau, il apparaît comme l'une des principales faiblesses en termes de sécurité de l'architecture SDN. Dans [7], il est montré que les flux de trafic falsifiés peuvent être utilisés pour faire des attaques DoS à des commutateurs OpenFlow ainsi que les contrôleurs. L'attaquant peut exposer des attaques DoS au contrôleur en envoyant beaucoup de paquets au contrôleur. De même, plus d'un attaquant ou des botnets peuvent envoyer une grande quantité de paquets aux commutateurs d'une manière systématique. Comme toutes les règles ne sont pas incluses dans les tableaux de flux des commutateurs, cela entraînera un grand nombre de requêtes à envoyer au contrôleur. Dans ce cas, le contrôleur sera exposé à une attaque DDoS et sera incapable de répondre aux demandes légitimes. Les attaques DoS et DDoS sur le contrôleur peuvent avoir un effet négatif sur le fonctionnement de l'ensemble du réseau. De même, l'attaque DoS peut être également possible pour les commutateurs dans le plan de données. Les tableaux de flux des commutateurs qui ont un cache limité seront vulnérables aux attaques d'inondation lorsque les attaquants envoient de gros paquets appartenant à des flux différents. Comme indiqué précédemment, certaines des règles de flux ne sont pas disponibles dans le tableau de flux, donc des requêtes sont envoyées au contrôleur. En attendant la réponse à ces requêtes, le cache des commutateurs se remplira rapidement. Ce type d'attaque est également appelé DoS Switch.

Menaces	Spécifique (SDN) ?	Conséquences
1) Flux de trafic faux ou falsifié	Non	Vulnérable aux attaques DoS
2) Vulnérabilités des switchs	Non	L'impact a potentiellement augmenté

3) Vulnérabilités des communications du plan de contrôle	Oui	La communication avec le contrôleur peut être exploitée
4) Vulnérabilité dans les contrôleurs	Oui	Le contrôle du contrôleur peut compromettre le réseau sous jacent
5) Manque de confiance entre les contrôleurs et les applications de gestion	Oui	Les applications malveillantes peuvent être développées et déployées dans les contrôleurs
6) Vulnérabilité dans les terminaux administratifs	Non	L'impact a potentiellement augmenté
7) Manque de ressources fiables	Non	Reste critique pour assurer une récupération rapide et un diagnostique quand une erreur se produit

Tableau II. Les menaces spécifiques et non spécifiques au SDN [7]

Le tableau ci-dessus résume les différentes menaces spécifiques et non spécifiques au SDN, comme on peut le constater, les menaces 3, 4 et 5 ne sont pas présentes dans les réseaux traditionnels. Ils sont spécifiques aux SDN car ils résultent de la séparation des plans de contrôle et de données et de l'introduction conséquente d'une nouvelle entité dans ces réseaux. Le contrôleur logique centralisé. D'autre part, les menaces 1, 2, 6 et 7 étaient déjà présentes dans les réseaux traditionnels. Cependant, l'impact de ces menaces peut être potentiellement augmenté ou du moins il peut être exprimé différemment et en conséquence il peut avoir besoin d'être traité de manière dissemblable.

Ces sept menaces montrent que le nombre d'attaque dans le SDN est plus élevé par rapport aux réseaux traditionnels et que des techniques d'atténuation différentes doivent être mises en place. Nous croyons que cela offre des arguments solides pour la nécessité d'envisager la sécurité.

VI. *SOLUTION DE SÉCURITÉ DU SDN.*

Dans SDN, le plan de contrôle est logiquement centralisé pour prendre des décisions centralisées basées sur la vue globale du réseau. SDN facilite l'identification rapide et adaptative des menaces au moyen d'un cycle de collecte d'informations à partir du réseau pour analyser, mettre à jour la politique et reprogrammer le réseau en conséquence. Les SDN facilitent la modification de la stratégie de sécurité dynamique pour définir des politiques de sécurité et réduire les risques de mauvaise configuration et de conflits de règles sur le réseau en temps d'exécution. En raison de la visibilité globale du réseau, les services de sécurité tels que les pare-feux et les systèmes de détection d'intrusion (IDS) peuvent facilement être déployés sur un trafic spécifié conformément aux politiques de sécurité définies.

Dans ce chapitre, des mesures de sécurité, des propositions et des plateformes pour sécuriser les applications et les plans de contrôle et de données sont discutées. Les solutions de sécurité qui renforcent explicitement la sécurité de chaque plan sont résumées dans le tableau II [53].

VII. *SOLUTIONS DE SÉCURITÉ DU PLAN D'APPLICATION*

Dans SDN, le contrôleur agit comme une couche intermédiaire entre le matériel réseau et les applications et cache la complexité réseau des applications. Par conséquent, l'architecture de contrôle centralisée par SDN facilite le déploiement de nouvelles applications qui permettraient de récupérer les statistiques du réseau et les caractéristiques des paquets via le contrôleur pour mettre en œuvre de nouveaux services de sécurité. Par conséquent, plusieurs langages de programmation de réseau, tels que Frentic, Procera et NetCore sont proposés, ces langages simplifient le développement des applications dans SDN. De plus, FRESCO [37] est utilisé pour permettre le développement d'applications de sécurité OpenFlow.

Le langage de script FRESCO permet aux développeurs de mettre en œuvre de nouvelles applications de sécurité qui peuvent être déployées sur n'importe quel contrôleur OpenFlow ou switch. De même, plusieurs frameworks de sécurité sont proposés pour vérifier si les applications respectent ou non les politiques de sécurité réseau.

1) Contrôle d'accès et d'autorisation :

Les applications doivent travailler dans leurs limites fonctionnelles et avoir un accès contrôlé aux ressources du réseau. PermOF [34] est un système d'autorisation à grain fin (fine-grained permission) utilisé pour fournir un accès contrôlé au contrôleur OpenFlow et un chemin de données aux applications OpenFlow. La conception est basée sur un ensemble d'autorisations et de mécanismes d'isolement pour appliquer le contrôle d'autorisation. L'ensemble d'autorisations est classé en lecture, notification, écriture et autorisations système.

L'autorisation de lecture est utilisée pour gérer la disponibilité d'informations sensibles sur une application. La permission de notification est utilisée pour savoir si une application doit être notifiée lors des événements en temps réel. Les autorisations d'écriture gèrent la capacité des applications à modifier les états du contrôleur ou les commutateurs, et en fin les autorisations système gèrent l'accès d'une application aux ressources locales fournies par le système d'exploitation. Le framework d'isolement maintient la supériorité du contrôleur sur les applications, ce framework isole le flux de contrôle et le flux de données, et permet au contrôleur de médiatiser toutes les activités des applications avec le monde extérieur. Cependant, il n'existe pas d'évaluation expérimentale du framework proposé.

Plans du SDN	Solution de sécurité	Menace	Type de solution
Plan d'application	FRESCO [37]	Menace dans les applications	Framework de développement
	PermOF [34]	Contrôle d'accès	Système d'autorisations et de permissions
	Assertion [38]	Contradiction des règles de flux	Framework de débogage
	Flover [39]	Violation de politique de sécurité	Vérification de la politique de sécurité
	OFTesting [40]	Défaut de programmation	Framework de test
Plan de contrôle	SE-Floodlight [41]	Autorisation d'applications.	Sécuriser l'architecture et les API (App-Ctrl)
	HybridCtrl [42]	Passage à l'échelle du contrôleur	Architecture hybride (réactif / proactif)
	DISCO [43]	Passage à l'échelle du contrôleur	Architecture de contrôleur distribué
	Ctrl-Placement [44]	Disponibilité du contrôleur	Framework de placement du contrôleur
	HyperFlow [46]	Disponibilité du contrôleur	Plan de contrôle distribué
	DDoSDetection [47]	Attaque DDoS	Framework de détection
Plan de données	FortNOX [48]	Contradiction des règles de flux	Framework de contrôleur
	FlowChecker [35]	Défaut des règles de flux	Outil de vérification de la configuration
	VeriFlow [49]	Défaut des règles de flux	Outil de débogage réseau

Tableau III : Les menaces et les solutions de sécurité dans les plans du SDN.

1) Conformité avec la sécurité réseau :

Dans SDN, les applications doivent avoir une vue cohérente du réseau et être notifiées de l'évolution du réseau. Une méthode pour vérifier et déboguer les applications SDN pour connaître les changements de topologie et rester compatible avec les conditions du réseau dynamique est présentée dans [38]. Le langage de débogage et de vérification basé sur l'assertion est développé pour permettre aux développeurs d'applications de vérifier les propriétés dynamiques de l'application du contrôleur via des instructions de programme de haut niveau.

Les méthodes basées sur l'assertion aident à détecter les bugs dans les programmes avant qu'ils ne soient déployés. Puisque VeriFlow [49] fournit des mécanismes pour inspecter les règles de flux en temps d'exécution, la procédure de vérification proposée dans [38] utilise l'algorithme de vérification VeriFlow avec une structure de données incrémentale pour vérifier efficacement les propriétés avec des conditions de vérification dynamiques.

Flover [39], un système de vérification de modèle pour OpenFlow, il vérifie que l'ensemble des politiques de flux ne viole pas les règles de sécurité du réseau. Flover est implémenté comme une application OpenFlow s'exécutant sur le contrôleur pour vérifier que les nouvelles règles de flux créées par le contrôleur sont cohérentes avec un ensemble de propriétés spécifiées. Il existe d'autres propositions telles que [40], où les auteurs proposent une procédure de test automatique pour identifier les bugs dans les programmes OpenFlow. La structure ndb [51] fournit un outil de débogage pour les programmeurs de réseau pour trouver la cause racine des bugs dans un réseau. L'OFRewind [52] peut enregistrer et rejouer le trafic sélectionné pour retracer les anomalies du réseau. Les frameworks ndb et OFRewind peuvent être utilisés pour tracer les applications qui provoquent des menaces de sécurité dans un réseau.

2) Remarque:

Les plates-formes et les propositions discutées ci-dessus aident à développer des applications de sécurité et à fournir une sécurité au plan de contrôle à partir d'applications malveillantes. Il est important de mentionner qu'il y a très peu d'efforts pour renforcer la sécurité des données des applications et des applications elles-mêmes. En outre, il n'existe pas de mécanismes permettant de différencier des applications entre tiers ou utilisateurs et des applications de service d'opérateur ou de réseau. De plus, les procédures de contrôle d'accès pour les applications imbriquées n'ont pas encore été vérifiées.

VIII. SOLUTIONS DE SÉCURITÉ DU PLAN DE CONTRÔLE

Les solutions de sécurité du plan de contrôle sont classés en propositions et approches pour sécuriser le plan de contrôle i) des applications malveillantes ou défectueuses, II) de contourner la sécurité du SDN en ciblant le passage à l'échelle du plan de contrôle, III) des attaques Dos et IV) assurer le plan de contrôle de sécurité et de disponibilité grâce à un contrôleur fiable.

1) Applications :

Puisque les applications accèdent aux ressources réseau et aux informations par le plan de contrôle, il est très important de sécuriser le plan de contrôle contre les applications malveillantes ou défectueuses. De plus, le plan de contrôle doit assurer l'accès aux applications légitimes selon leurs besoins fonctionnels, mais aussi en fonction des contraintes de sécurité.

« Security-Enhanced (SE) Floodlight controler » [41], une version étendue du contrôleur de projecteur d'origine, est une tentative vers une couche de contrôle SDN sécurisée idéale. Le contrôleur « Floodlight » de SE fournit des mécanismes pour la séparation de privilèges en ajoutant une API nord programmable sécurisée au contrôleur pour fonctionner comme un médiateur entre les applications et le plan de données. Il introduit un module de vérification et d'exécution d'application OpenFlow pour valider l'intégrité des modules de classe qui produisent les règles de flux. Pour la résolution des conflits basés sur les rôles, SE-Floodlight assigne des rôles d'autorisation à des applications OpenFlow pour résoudre les conflits de règles en comparant les rôles faisant autorité des producteurs de règles contradictoires. De la même façon, il peut restreindre les messages PACKET_OUT produits par diverses applications et, par conséquent, assurer la médiation des règles de flux sécurisé. Le contrôleur SE-Floodlight introduit également un nouveau sous-système d'audit OpenFlow qui permet d'éliminer tous les événements liés à la sécurité qui se produisent dans la couche de contrôle d'OpenFlow.

2) Passage à l'échelle du contrôleur :

Dans la norme OpenFlow du SDN, un contrôleur installe des règles distinctes pour chaque connexion d'un client, également appelé "microflux," conduisant à l'installation d'un grand nombre de flux dans les commutateurs et une lourde charge sur le contrôleur. Par conséquent, diverses approches sont suggérées soit pour minimiser la charge sur un contrôleur, distribuer des fonctionnalités de plan de contrôle, soit pour maximiser la puissance de traitement et la mémoire des contrôleurs. Outre cela, OpenFlow prend en charge l'utilisation de caractères génériques de sorte que le contrôleur dirige un ensemble de requêtes client vers des serveurs. Les mécanismes génériques exploitent le support de commutateur pour des règles génériques pour obtenir une plus grande évolutivité, en plus de maintenir une charge équilibrée sur le contrôleur. Ces techniques utilisent des algorithmes qui calculent des règles génériques concises qui permettent d'atteindre la répartition cible du trafic et s'adaptent automatiquement aux changements des stratégies d'équilibrage de charge sans perturber les connexions existantes.

Une analyse comparative de plusieurs modèles de régulateurs OpenFlow réactifs et proactifs pour le passage à l'échelle est présentée dans [42]. Les contrôleurs réactifs reçoivent le premier paquet de flux à partir du commutateur pour remplir la table de flux dans le commutateur pour ce flux particulier, alors que les contrôleurs proactifs définissent des règles de flux avant que les flux arrivent au commutateur sur la base de certaines règles de transfert prédéfinies. Le document [42] démontre que les contrôleurs proactifs sont plus évolutifs que leurs homologues. Cependant, un contrôleur proactif pur devrait connaître tous les flux de trafic à l'avance ce qui n'est pas pratiquement possible. Par conséquent, l'auteur suggère une architecture de contrôleur hybride dans laquelle les contrôleurs agissent de manière réactive pour configurer les routes et ont une certaine intelligence pour agir de manière proactive pour comprendre le comportement du trafic et définir un chemin à l'avance.

Il y a des efforts pour augmenter la puissance de traitement des contrôleurs et partager les responsabilités entre un ensemble de contrôleurs. McNettle [54] est un contrôleur SDN qui passe l'échelle avec des noyaux CPU multiples supportant les algorithmes de contrôle. Elle requiert une visibilité globale des changements d'état se produisant à des débits d'écoulement. Les programmeurs peuvent étendre McNettle avec un langage de programmation fonctionnel de haut niveau. Comparé au NOX, McNettle est plus évolutif car il peut s'échelonner jusqu'à 46 cœurs tandis que NOX peut s'échelonner jusqu'à 10 cœurs, en plus d'être plus efficace en termes de performance. Dans [54], le parallélisme par des processeurs multi cœurs est proposé pour augmenter les performances de traitement des contrôleurs pour une plus grande évolutivité et disponibilité.

Le plan de commande SDN distribué (DISCO) est présenté dans [43] pour fournir des fonctionnalités de plan de contrôle à des réseaux distribués et hétérogènes. DISCO se compose de deux parties: intra-domaine et inter-domaine. Les modules intra-domaine permettent la surveillance du réseau et la hiérarchisation du flux de gestion pour le contrôleur de calculer les chemins des flux prioritaires. Ces modules aident à réagir dynamiquement aux problèmes de réseau en réorientant et/ou arrêtant le trafic réseau en fonction de la criticité des flux. La partie inter-domaine gère la communication entre contrôleurs et se compose d'un messager et des agents. Le module messager découvre des contrôleurs voisins et fournit un canal de contrôle entre ces domaines voisins. Les agents utilisent les canaux fournis par les messagers pour échanger des informations à l'échelle du réseau avec d'autres contrôleurs.

HyperFlow [46] est une plateforme de contrôle évolutive physiquement distribuée et logiquement centralisée basée sur des événements. HyperFlow permet aux opérateurs de réseau de déployer plusieurs contrôleurs, étant capables de prise de décision locale, afin de maximiser le passage à l'échelle du contrôleur et de minimiser le temps d'installation du flux. Heller et al. [44] fournissent un compromis entre la disponibilité et la distribution d'état et suggèrent de placer les contrôleurs pour minimiser la latence comme point de départ, puis utilisent des algorithmes d'équilibrage de charge pour équilibrer la charge entre les contrôleurs.

3) Mitigation du DoS :

Les attaques DoS ou DDoS peuvent être atténuées en analysant le comportement de flux et les statistiques de flux stockées dans les commutateurs OpenFlow. Puisque les statistiques de commutation peuvent être facilement récupérées dans le contrôleur OpenFlow, le processus de collecte de statistiques dans OpenFlow est comparativement rentable en raison de frais généraux. Une détection d'attaque DDoS légère utilisant les cartes auto-organisatrices (Self-Organizing Maps) (SOM) [55] est présentée dans [47]. SOM est un réseau de neurones artificiel utilisé pour transformer un modèle donné de n-dimensions de données en une carte à 1 ou 2 dimensions. Le processus de transformation effectue l'ordonnancement topologique, où des profils de données avec des caractéristiques statistiques similaires sont rassemblés pour un traitement ultérieur. Dans [47], le mécanisme de SOM est utilisé pour trouver des relations cachées entre les flux entrant dans le réseau.

La méthode de détection DDoS dans [47] utilise trois modules, c'est-à-dire le collecteur de flux, l'extracteur de fonctions et le classificateur. Le module de collecteur de flux recueille les entrées de flux de toutes les tables de flux des commutateurs OpenFlow pendant des intervalles prédéterminés. Le module extracteur de fonctions extrait les fonctions qui sont importantes pour les attaques DDoS, les rassemble en 6-tuples et les transmet au classificateur. Les caractéristiques extraites comprennent la moyenne des paquets par flux, la moyenne des Bytes par flux, la moyenne de la durée par flux, le pourcentage des flux de paires, la croissance des flux uniques et la croissance des différents ports. Le classificateur analyse si un 6-tuple donné répond à une attaque ou un trafic normal en utilisant SOM. SOM est entraîné avec un ensemble suffisamment grand d'échantillons de 6-tuple recueillis soit pendant une attaque ou un trafic normal pour créer une carte topologique où différentes régions présentent chaque type de trafic. Dorénavant, chaque fois que le SOM entraîné est stimulé avec un 6-tuple extrait des entrées de flux collectées à partir de commutateurs OpenFlow, il sera capable de classer un trafic soit comme un trafic normal ou une attaque.

4) Emplacement du régulateur fiable :

Le problème de placement du contrôleur est présenté dans [44], où l'on montre que le nombre de contrôleurs et les emplacements topologiques des contrôleurs sont des défis multiples pour l'évolutivité du réseau et la résilience dans SDN. Par conséquent, le placement optimal du contrôleur a attiré beaucoup d'attention de la communauté de recherche et un certain nombre d'algorithmes pour un placement optimal ont été examinés et testés. L'Algorithme « Simulated Annealing » (SA), un algorithme probabiliste générique, a été privilégié comme l'algorithme le plus optimal pour le positionnement du contrôleur dans [45].

Le problème de placement du contrôleur est décrit dans [45] afin de maximiser la fiabilité des opérations de contrôle SDN tout en répondant aux exigences de temps de réponse. Les auteurs suggèrent que la synchronisation d'état et la coordination de contrôle entre les contrôleurs est nécessaire et peut être utilisée avec des techniques telles que la hiérarchie du contrôleur. Pour un placement optimal, le pourcentage attendu de perte de chemin de contrôle est utilisé comme une métrique de fiabilité, où la perte de chemin de contrôle est considérée comme le nombre de chemins de commande brisés en raison de défaillances de réseau. Les mécanismes proposés dans [45] optimisent le réseau en minimisant le pourcentage attendu de perte de contrôle. Plusieurs algorithmes avec leurs avantages pour le placement du contrôleur sont examinés en utilisant des topologies réelles, et les compromis entre la fiabilité et les latences sont présentées.

Problème de Provisionnement de Contrôleur Dynamique (DCPP) a été réglé dans [56]. Les auteurs proposent un framework pour le déploiement dynamique de plusieurs contrôleurs dans un WAN, dans lequel le nombre et l'emplacement des contrôleurs sont ajustés en fonction de la dynamique du réseau. DCPP a été formulé comme un programme linéaire d'entier (IGP) qui utilise des modèles de trafic pour minimiser les coûts de la collecte d'état de commutation, la synchronisation entre contrôleurs et la réaffectation du commutateur au contrôleur. De plus, la stratégie de déploiement du contrôleur décrite dans [56] fournit un échange équitable entre le temps d'établissement du flux et la charge de communication dans les SDN.

5) Échange intelligent entre plan de contrôle et de données :

L'échange intelligent entre le contrôleur et les commutateurs peut être utilisé pour augmenter la disponibilité du contrôleur. « Devolved OpenFlow » ou DevoFlow [57] est une telle approche qui modifie le modèle OpenFlow afin de minimiser l'interaction entre les données de contrôle et les données. L'architecture renvoie un certain contrôle aux commutateurs tout en maintenant la commande centrale du contrôleur. Les auteurs suggèrent que la visibilité centrale de tous les flux pourrait ne pas être nécessaire mais plutôt coûteuse en termes d'évolutivité. Par conséquent, DevoFlow est conçu de telle sorte qu'il utilise des règles OpenFlow strictes, et les commutateurs peuvent prendre des décisions de routage local où le contrôle par flux par le contrôleur pourrait ne pas être nécessaire. La plupart des microflux de DevoFlow sont traités dans le plan de données, mais les opérateurs peuvent gérer ou contrôler tout flux qui leur importe à des fins de gestion.

Le langage de programmation de base de réseau (NetCore) [50] est un langage déclaratif de haut niveau pour exprimer des politiques de transmission de paquets dans SDNs. NetCore propose de nouveaux algorithmes de compilation et un système d'exécution pour concilier le retard engendré par l'implication

d'un contrôleur dans toutes les décisions de traitement des paquets. NetCore divise les responsabilités de traitement des paquets entre les contrôleurs et les commutateurs à l'aide des algorithmes de compilation couplés au système d'exécution.

IX. Solutions de sécurité du Plan de données

Comme point de départ, le plan de données doit être sécurisé des applications malintentionnées qui peuvent installer ou modifier des flux dans le chemin de données. Par conséquent, des mécanismes de contrôle de sécurité à grain fin « fine-grained security » tels que l'authentification et l'autorisation sont utilisés pour les applications qui peuvent modifier les règles de flux. FortNox [48] est une telle plate-forme qui permet au contrôleur NOX OpenFlow de régler les contradictions de flux de contrôle en temps réel et d'autoriser les applications OpenFlow avant qu'elles puissent changer les règles de flux.

FortNOX fournit une autorisation basée sur les rôles via des signatures numériques et l'application des contraintes de sécurité via une extension logicielle dans le contrôleur NOX OpenFlow. À l'aide d'un moteur de détection de conflit hépatique, FortNOX intervient dans la médiation de toutes les requêtes d'insertion OpenFlowrule à l'aide d'un algorithme d'analyse des conflits de règles, c'est-à-dire «alias set rule reduction». Dans le même réseau OpenFlow.

FlowChecker [35] est un outil de vérification de configuration utilisé pour identifier les incohérences dans les règles OpenFlow dans un seul commutateur ou plusieurs éléments de chemin de données inter-fédérés. FlowChecker peut être utilisé comme une application OpenFlow ou un contrôleur principal pour valider, analyser et appliquer les configurations OpenFlow de bout en bout au moment de l'exécution. VeriFlow [49] est un outil de débogage de réseau utilisé pour trouver des règles défectueuses insérées par les applications SDN et les empêcher de causer un comportement anormal du réseau.

Puisque la connectivité du contrôleur est essentielle pour la fonctionnalité d'un commutateur, des connexions redondantes ou des mécanismes de récupération de liaison rapide doivent être en place pour la communication entre les deux. Le protocole OpenFlow lui-même contribue à la récupération du commutateur en utilisant des techniques de détection de connexion pour vérifier la connexion au contrôleur, comme l'envoi périodique de messages d'analyse d'activité au contrôleur. Le protocole OpenFlow offre également la flexibilité de configurer une connexion secondaire avec un contrôleur de sauvegarde à utiliser si le premier contrôleur échoue.

Une planification et une segmentation du réseau appropriées peuvent également aider à renforcer la résilience des commutateurs OpenFlow et maximiser leur connectivité avec les contrôleurs. Un commutateur OpenFlow qui est toujours connecté au contrôleur sera moins sujet aux attaques de saturation, du fait qu'il ne sera pas nécessaire de stocker des flux non sollicités pour une durée plus longue. Il est démontré dans [36] que la longueur du chemin entre un commutateur et un contrôleur est directement proportionnelle à la perte de connectivité. Par conséquent, il est suggéré dans [36] que, outre le nombre optimal d'interrupteurs sous un contrôleur, la longueur des chemins entre les contrôleurs et les commutateurs doit être optimale. Cela permettra non seulement d'améliorer les performances du système en termes de contraintes de délai, mais aussi de restaurer rapidement, d'améliorer la disponibilité du contenu pour les applications de sécurité et d'effectuer une analyse de sécurité rapide.

X. Conclusion

Dans cet article nous avons présenté le modèle d'architecture SDN, ces différents défis de sécurité, les menaces auxquelles cette architecture se heurte et finalement les solutions qui ont été proposé pour combler le manque de sécurité.

Les réseaux définis par logiciel (SDN) apportent une nouvelle approche dans le monde des réseaux. En effet, les architectures réseau traditionnelles se transforment en architecture de réseau dynamique avec SDN. Bien que ce dernier offre une plate-forme ouverte et programmable une multitude d'attaques peuvent être perpétrées à l'encontre des différents plans du modèle d'architecture SDN. Parmi ces attaques, les plus répondues et dangereuses nous pouvons retrouver l'attaque DoS ou DDoS au niveau du plan de contrôle, les flux de trafic faux ou falsifiés au niveau du plan de donnée mais encore des applications non authentifiées au niveau du plan d'application. Ainsi le SDN doit être conçu de manière appropriée en prenant en considération l'aspect sécurité, comme par exemple en introduisant plus d'applications orientées sécurité et en effectuant une étude au préalable des éventuelles failles engendrées par ces applications.

References

[1] SEZER, Sakir, SCOTT-HAYWARD, Sandra, CHOUHAN, Pushpinder Kaur, et al. Are we ready for SDN? Implementation challenges for software-defined networks. IEEE Communications Magazine, 2013, vol. 51, no 7, p. 36-43.

[2] Réseaux logiciels, virtualisation, SDN, 5G et sécurité. Volume 1. Guy Pujolle, ISTE éditions.

[3] www.opennetworking.org

[4] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," Open Networking Foundation, ONF White Paper, April 2012.

[5] OpenFlow Switch Specification, Open Networking Foundation, ONF Specification Version 1.4.0, October 2013.

[6] S. Shirali-Shahreza and Y. Ganjali, "Empowering Software Defined Network controller with packet-level information," in Proceedings of the 2013 IEEE International Conference on Communications Workshops, ser. ICC '13.IEEE Computer Society, June 2013, pp. 1335–1339.

[7] D. Kreutz, F. M. Ramos, and P. Verissimo, "Towards Secure and Dependable Software-Defined Networks," in Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined

Networking, ser. HotSDN '13. ACM, August 2013, pp. 55–60.

[8] Z. Yan and C. Prehofer, "Autonomic Trust Management for a Component-Based Software System,"IEEE Transactions on Dependable and Secure Computing, vol. 8, no. 6, pp. 810–823, Nov 2011.

[9] R. Kloti, V. Kotronis, and P. Smith, "OpenFlow: A Security Analysis," In Proceedings of the 8th Workshop on Secure Network Protocols (NPSec), part of IEEE ICNP , ser. NPSec '13. IEEE, October 2013.

[10] B. Heller, R. Sherwood, and N. McKeown, "The Controller Placement Problem," in Proceedings of the First Workshop on Hot Topics in Software Defined Networks , ser. HotSDN '12. ACM, August 2012, pp. 7–12.

[11] P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu, "A Security Enforcement Kernel for OpenFlow Networks," in Proceedings of the First Workshop on Hot Topics in Software Defined Networks, ser. HotSDN '12.ACM, August 2012, pp. 121–126.

[12] Ijaz Ahmad, Suneth Namal, Mika Ylianttila, and Andrei V. Gurtov. Security in software defined networks: A survey. IEEE Communications Surveys and Tutorials, 17(4):2317–2346, 2015.

[13] T. Nadeau, "Software driven networks problem statement," Network Working Group Internet-Draft, Sep. 30, 2011. [Online]. Available: https://tools.ietf.org/html/draft-nadeau-sdn-problem-statement-00

[14] S. Shin, P. Porras, V. Yegneswaran, and G. Gu, "A framework for integrating security services into software-defined networks," in *Proc. ONS*, 2013, pp. 1–2.

[15] X. Wen, Y. Chen, C. Hu, C. Shi, and Y. Wang, "Towards a secure controller platform for OpenFlow applications," in *Proc. 2nd ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw.*, 2013, pp. 171–172.

[16] Privacy Commission jeudi 12 Janvier 2017 https://www.privacycommission.be/fr/taxonomy/term/140

[17] S. Hartman, M. Wasserman and D. Zhang, "Software driven networks problem statement," Network Working Group Internet-Draft, Oct. 15, 2013. [Online]. Available: https://tools.ietf.org/html/drafthartman- sdnsec-requirements-00

[18] H. Xie, T. Tsou, D. Lopez, H. Yin, and V. Gurbani, "Use cases for ALTO with software defined networks," Working Draft, IETF Secretariat, Internet-Draft, 2012. [Online]. Available: https://tools.ietf.org/html/draft-xie-alto-sdn-use-cases-01

[19] A. D. Ferguson, A. Guha, J. Place, R. Fonseca, and S. Krishnamurthi, "Participatory networking," in *Proc. Hot-ICE*, 2012, pp. 327–338.

[20] J. Naous, D. Erickson, G. A. Covington, G. Appenzeller, and N. McKeown, "Implementing an OpenFlow switch on the NetFPGA platform," in *Proc. 4th ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, 2008, pp. 1–9.

[21] M. Jarschel *et al.*, "Modeling and performance evaluation of an OpenFlow architecture," in *Proc. 23rd ITCP*, 2011, pp. 1–7.

[22] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "Avant-guard: Scalable and vigilant switch flow management in software-defined networks," in *Proc. CCS*, 2013, pp. 413–424.

[23] G. Yao, J. Bi, and L. Guo, "On the cascading failures of multi-controllers in software defined networks," in *Proc. 21st IEEE ICNP*, Oct. 2013, pp. 1–2.

[24] E. Al-Shaer and S. Al-Haj, "FlowChecker: Configuration analysis and verification of federated openflow infrastructures," in *Proc. 3rd ACM Workshop SafeConfig*, 2010, pp. 37–44.

[25] Y. Zhang, N. Beheshti, and M. Tatipamula, "On resilience of splitarchitecture networks," in *Proc. IEEE GLOBECOM*, Dec. 2011, pp. 1–6.

[26] T. Dierks, "The Transport Layer Security (TLS) protocol version 1.2," 2008. [Online]. Available: http://tools.ietf.org/html/rfc5246

[27] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security Version 1.2," 2012. [Online]. Available: http://tools.ietf.org/html/rfc6347

[28] K. Benton, L. J. Camp, and C. Small, "OpenFlow vulnerability assessment," in *Proc. 2nd ACM SIGCOMM Workshop HotSDN*, 2013, pp. 151–152.

[29] Z. Yan and C. Prehofer. "Autonomic Trust Management for a Component-Based Software System". In: IEEE Trans. on Dep. and Sec. Computing 8.6 (2011).

[30] R. Holz et al. "X.509 Forensics: Detecting and Localising the SSL/TLS Men-in-the-Middle". In: Computer Security. LNCS. 2012.

[31] M. Georgiev et al. "The most dangerous code in the world: validating SSL certificates in non-browser software". In: ACM CCS. 2012.

[32] R. Sherwood et al. FlowVisor: A Network Virtualization Layer. Tech. rep. Deutsche Telekom Inc. R&D Lab, Stanford, Nicira Networks, 2009.

[33] Y. G. Desmedt. "Threshold cryptography". In: European Trans. on Telecommunications 5.4 (1994).

[34] X. Wen, Y. Chen, C. Hu, C. Shi, and Y. Wang, "Towards a secure controller platform for OpenFlow applications," in *Proc. 2nd ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw.*, 2013, pp. 171–172

[35] E. Al-Shaer and S. Al-Haj, "FlowChecker: Configuration analysis and verification of federated openflow infrastructures," in *Proc. 3rd ACM Workshop SafeConfig*, 2010, pp. 37–44.

[36] Y. Zhang, N. Beheshti, and M. Tatipamula, "On resilience of splitarchitecture networks," in *Proc. IEEE GLOBECOM*, Dec. 2011, pp. 1–6.

[37] S. Shin *et al.*, "FRESCO: Modular composable security services for software-defined networks," in *Proc. Netw. Distrib. Security Symp.*, Feb. 2013, pp. 1–16. [Online]. Available: http://www.csl.sri.com/users/vinod/papers/fresco.pdf

[38] R. Beckett *et al.*, "An assertion language for debugging SDN applications," in *Proc3rd ACM Workshop Hot Topics Softw. Defined Netw.*, 2014, pp. 91–96.

[39] S. Son, S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "Model checkin invariant security properties in openflow," in *Proc. IEEE ICC*, 2013, pp. 1974–1979.

[40] M. Canini, D. Kostic, J. Rexford, and D. Venzano, "Automating the testing of OpenFlow applications," in *Proc. 1st Int. WRiPE*, 2011, pp. 1–6.

[41] "Security-enhanced floodlight," SDx Central, Sunnyvale, CA, USA. [Online]. Available: http://www.sdncentral.com/education/towardsecure-sdn-control-layer/2013/10/

[42] M. Fernandez, "Comparing openflow controller paradigms scalability: reactive and proactive," in *Proc. IEEE 27th Int. Conf. AINA*, Mar. 2013, pp. 1009–1016.

[43] K. Phemius, M. Bouet, and J. Leguay, "DISCO: Distributed multidomain SDN controllers," in *Proc. IEEE NOMS*, May 2014, pp. 1–4.

[44] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proc. 1st ACM Workshop HotSDN*, 2012, pp. 7–12.

[45] Y. Hu, W. Wang, X. Gong, X. Que, and S. Cheng, "On reliabilityoptimized controller placement for software-defined networks," *Commun., China*, vol. 11, no. 2, pp. 38–54, Feb. 2014.

[46] A. Tootoonchian and Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow," in *Proc. Internet Netw. Manag. Conf. Res. Enterprise Netw. USENIX Assoc.*, 2010, p. 3.

[47] R. Braga, E. Mota, and A. Passito, “Lightweight DDoS flooding attack detection using NOX/OpenFlow,” in *Proc. IEEE 35th Conf. LCN*, Oct. 2010, pp. 408–415.

[48] P. Porras *et al.*, “A security enforcement kernel for OpenFlow networks,” in *Proc. 1st Workshop HotSDN*, 2012, pp. 121–126.

[49] A. Khurshid, W. Zhou, M. Caesar, and P. B. Godfrey, “Veriflow: Verifying network-wide invariants in real time,” *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 467–472, Sep. 2012.

[50] C. Monsanto, N. Foster, R. Harrison, and D. Walker, “A compiler and run-time system for network programming languages,” *SIGPLAN Notices*, vol. 47, no. 1, pp. 217–230, Jan. 2012

[51] N. Handigol, B. Heller, V. Jeyakumar, D. Mazières, and N. McKeown,

“Where is the debugger for my software-defined network?” in *Proc. 1st Workshop Hot Topics Softw. Defined Netw.*, 2012, pp. 55–60.

[52] A. Wundsam, D. Levin, S. Seetharaman, A. Feldmann, “OFRewind: Enabling record and replay troubleshooting for networks.” in *Proc. USENIX Annu. Tech. Conf.*, 2011, p. 29.

[53] Ijaz Ahmad, Suneth Namal, Mika Ylianttila, and Andrei V. Gurtov. Security in software de_ned networks: A survey. IEEE Communications Surveys and Tutorials, 17(4):2317{2346, 2015}.

[54] A. Voellmy and J. Wang, “Scalable software defined network controllers,” in *Proc. ACM SIGCOMM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2012, pp. 289–290.

[55] T. Kohonen, “The self-organizing map,” *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.

[56] M. Bari *et al.*, “Dynamic controller provisioning in software defined networks,” in *Proc. Int. CNSM*, Oct. 2013, pp. 18–24.

[57] J. C. Mogul *et al.*, “DevoFlow: Cost-effective flow management for high performance enterprise networks,” in *Proc. 9th ACM SIGCOMM Workshop Hotnets-IX*, 2010, pp. 1–6