Carlos Félix Barros Rodrigues Silva

FRAMEWORK DE SEGURANÇA PARA REDES SDN EM CENÁRIOS 5G

Dissertação no âmbito do Mestrado em Segurança Informática orientada pelo Professor Doutor Bruno Sousa, coorientada pelo professor Doutor João Vilela e à Faculdade de Ciências e Tecnologia / Departamento de Engenharia Informática.

Outubro de 2021

Esta página foi intencionalmente deixada em branco.

Resumo

Nos recentes anos com o desenvolvimento e expansão das tecnologias, as redes tradicionais estão a ser substituídas por soluções como as redes SDN (Software Defined Network) que oferecem maior flexibilidade, e capacidade de gestão para redes de grandes dimensões. Com a inovação das redes SDN também se intensificaram os problemas de segurança que podem comprometer a rede.
A presente dissertação tem como objetivo principal o desenvolvimento e validação de uma framework de segurança para redes definidas por software, em cenários 5G. Para este estudo foi analisado o estado da arte das redes SDN, em específico a sua arquitetura, protocolos de comunicação, e os seus componentes e os controladores disponíveis no mercado. Terminado o processo de análise do estado da arte foi feito o mesmo processo, mas com o foco nos aspetos de segurança que abrangem as redes SDN, em específico fazer o levantamento de todas as vulnerabilidades e ameaças reportadas, e os mecanismos de segurança que possam existir de forma a mitigar estas ameaças e proteger os dados da rede, onde foi desenvolvido uma metodologia de forma a classificar as vulnerabilidades seguindo diversas métricas, desta forma podemos filtrar e identificar quais são as que apresentam maior relevância e impacto para os controladores. Por fim foi implementado e documentado todo o processo de configuração de segurança nas interfaces do controlador, e também foi feita a avaliação preliminar do impacto das configurações de segurança nas diferentes interfaces do controlador.

Palavras-Chave

Redes definidas por software, SDN,ONOS,Segurança,ONOS Security Mode,TLS,CROCUS.

Esta página foi intencionalmente deixada em branco.

Abstract

In the recent years with the development and expansion of technologies, traditional networks are increasingly being replaced by more versatile solutions such as SDN (Software Defined Network) which offers greater flexibility, and management capacity for large networks. The innovation of SDN networks also introduced security problems that can compromise the network.
The main objective of this dissertation the specification and validation of a security framework for software defined networks, in 5G scenarios. To do this study we analyzed the state of the art of SDN networks, specifically their architecture, communication protocols, and their components and controllers available on the market. Once the state-of-the-art process was completed, the same process was done, but with the focus on the security aspects that cover SDN networks, specifically to survey all reported vulnerabilities and threats, and the security mechanisms that may exist to mitigate these threats and protect the network data. It should be noted that a methodology has been developed to classify the vulnerabilities following several metrics, in this way we can filter and identify which are the most relevant ones and their impact on SDN controllers. The entire security configuration process in the controller interfaces was implemented and documented, and a preliminary assessment of the impact of the security configurations on the different controller interfaces was also carried out.

Keywords

Software defined networks, SDN ,ONOS,Security,ONOS Security Mode,TLS, CROCUS.

Esta página foi intencionalmente deixada em branco.

Agradecimentos

Aproveito este espaço para agradecer a minha família pelo apoio incondicional durante todo meu percurso académico e pessoal, percurso este marcado por momentos felizes e outros menos afortunados, que mesmo estando longe fisicamente estiveram sempre presentes.

Aos meus orientadores Professor Doutor Bruno Sousa e Professor Doutor João Vilela um obrigado pelo apoio, conhecimentos partilhados, e sobretudo a paciência e disponibilidade em todas as nossas reuniões semanais de trabalho.

A todos que de forma direta ou indireta contribuíram para a minha formação académica e pessoal durante todo este percurso.
.

Um muito Obrigado a todos.

Esta página foi intencionalmente deixada em branco.

Índice

Acrónimos ..... **x**
Lista de Figuras ..... **xii**
Lista de Tabelas ..... **xiv**
1 Introdução **1**
1.1 Contexto 1
1.2 Objetivos 2
1.2.1 Objetivos Finais 2
1.3 Estrutura e Organização do Documento 3
1.4 Planeamento 4
1.5 Contribuições 6
2 Redes Definidas por Software **7**
2.1 Conceito 7
2.2 Arquitetura 8
2.3 Interfaces 9
2.4 Controladores 11
2.4.1 ONOS-Open Network Operating System 15
2.5 Mininet 17
2.6 Atomix 19
2.7 Protocolo Transport Layer Security-TLS 20
2.7.1 Diferenças entre TLS 1.2 e TLS 1.3 21
2.8 Finalização do Capítulo 21
3 Segurança **22**
3.1 Ativos SDN 22
3.2 Ameaças e Vulnerabilidades 23
3.2.1 Ameaças SDN Genéricas 23
3.2.2 Ameaças SDN no plano de dados 24
3.2.3 Ameaças SDN no plano de aplicação 25
3.2.4 Ameaças SDN no plano de controlo 26
3.3 Metodologia de avaliação de vulnerabilidades em controladores 27
3.3.1 Passo 1-Identificar os controladores 27

3.3.2 Passo 2- Recolher as vulnerabilidades 29
3.3.3 Passo 3-Classificação final das vulnerabilidades reportadas 33
3.4 Finalização do Capítulo 36
4 Desenvolvimento e Implementação **37**
4.1 ONOS SecurityMode 37
4.2 Implementação 38
4.2.1 Cluster 39
4.2.2 Configuração TLS 43
4.2.2.1 TLS na Interface Northbound 46
4.2.2.2 TLS na Interface East/Westbound 47
4.2.2.3 TLS na Interface SouthBound 49
4.3 Finalização do Capítulo 51
5 Validação e Análise **52**
5.1 Estratégias de Validação 52
5.2 Avaliação do impacto dos diferentes algoritmos na NBI 53
5.2.1 Avaliação do impacto do controlador com interfaces Seguras e sem Segurança 57
5.2.1.1 Interface NBI 57
5.2.1.2 Avaliação do impacto do controlador com as Interfaces SBI e EBI/WBI Seguro 58
5.3 Finalização do capítulo 59
6 Conclusão **60**
Referências **62**
Anexo A **66**
Anexo B **87**
Anexo C **94**
Anexo D **100**

Acrónimos

AHP	Analytic Hierarchy Process
AES	Advanced Encryption Standard
APIs	Application Programming Interface
CBE	Container Based Emulation
CVSS	Common Vulnerability Scoring System
DCCP	Datagram Congestion Control Protocol
DDoS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DH	Diffie-Hellman
DoS	Denial of Service
EBI	Eastbound Interface
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
GUI	Graphical User Interface
IDS	Intrusion Detection System
IPS	Intrusion Prevention Systems
JSON	JavaScript Object Notation
MiTM	Man In The Middle
NBI	Northbound Interface
NBIWG	Northbound Interface Work Group
NETCONF	Network Configuration Protocol
NRENs	National Research and Education Network
OVS	OpenvSwitch
ONF	Open Networking Foundation
ONOS	Open Network Operating System
SAN	Subject Alternative Name
RTT	Round-trip time
RSA	Rivest-Shamir-Adleman
SBI	Southbound Interface
SDN	Software Defined Network
SNMP	Simple Network Management Protocol
SPOF	Single Point of Failure
TLS	Transport Layer Security
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WBI	Westbound Interface

Esta página foi intencionalmente deixada em branco.

Lista de Figuras

Figura 1-Rede SDN 1a) vs Rede Tradicional 1b) baseado em [2] 8
Figura 2-Interfaces nas redes SDN baseado em [8] 11
Figura 3-Sincronização entre os nodos do controlador ONOS [19] 16
Figura 4-Rede Física emulada [22] 18
Figura 5-Equação rCric- rácio de solvabilidade de vulnerabilidade críticas 29
Figura 6- Métricas CROCUS na etapa de classificação 33
Figura 7-Vulnerabilidades OpenDayLight 34
Figura 8-Vulnerabilidade ONOS 35
Figura 9-Cenário de implementação de segurança nas interfaces (NBI, SBI, EBI e WBI) 38
Figura 10-Descoberta ONOS/Atomix [37] 42
Figura 11-Cenario de avaliação TLS 52
Figura 12-Gráfico comparativo do RSA e ECDHE 55
Figura 13-Gráfico comparativo das interfaces com e sem segurança 57
Figura 14-output do comando ´ss´ 59
Figura 15-Exemplo de permissões[34] 89
Figura 16-Revisão e Validação e Permissões [35] 90
Figura 17-Mecanismo de controlo de acesso[36] 91
Figura 18-Serviços administrativos e não administrativo [37] 92
Figura 19-Exemplo de um ficheiro de políticas 93

Esta página foi intencionalmente deixada em branco.

Lista de Tabelas

Tabela 1-Tarefas proposta para entrega intermédia 4
Tabela 2-Entrega intermédia 4
Tabela 3- Cronograma da entrega Final da tese 5
Tabela 4-Tarefas Realizadas 5
Tabela 5-Entrega Intermédia atualizada 6
Tabela 6-Entrega Final 6
Tabela 7-Lista adaptada dos controladores do mercado SDN 12
Tabela 8-Quadro comparativo dos controladores SDN 15
Tabela 9-Comandos mininet 17
Tabela 10-Fases da Metodologia de avaliação 27
Tabela 11-Métricas de segurança funcionais de CROCUS 28
Tabela 12-Classificação CVSS3 29
Tabela 13-Categorias de avaliação 30
Tabela 14-Métricas de Explorabilidade e Impacto 30
Tabela 15-Métricas adicionais 32
Tabela 16-Atomix.conf 40
Tabela 17-Cluster.json 42
Tabela 18-Gerar certificados 44
Tabela 19-Comando para distribuir certificado 44
Tabela 20-Extrair ficheiro PEM 45
Tabela 21-Copiar e importar certificados 45
Tabela 22-Alterar ficheiro ONOS 46
Tabela 23-Cluster.messaging.tls 47
Tabela 24-Ficheiro onos-service 48
Tabela 25-Obter certificado em formato .p12 49
Tabela 26-Exportar certificados p12 para pem 49
Tabela 27-Obter certificado para ser partilhado 49
Tabela 28-Copiar certificado para OVS 50
Tabela 29-Gerar certificados OvS 50
Tabela 30-Habilitar o OvS para utilizar as novas chaves 50
Tabela 31-Copiar e importar chave OvS para ONOS 51
Tabela 32-Métricas de Avaliação 53
Tabela 33- Encriptação e algoritmos de assinatura 53
Tabela 34-Gerar certificados para validação 53
Tabela 35-Legenda das métricas de avaliação 54
Tabela 36-Resultado algoritmo RSA 54
Tabela 37-Resultado algoritmo ECDHE 54
Tabela 38-OpenSSL s_time 56
Tabela 39-OpenSSL ECDHE 56
Tabela 40-OpenSSL RSA 56
Tabela 41-Controlador com TLS vs sem TLS 57

Tabela 42-Permissões de Aplicações [33] 89
Tabela 43-Descrição dos serviços administrativos e regulares 92

Esta página foi intencionalmente deixada em branco.

1 Introdução

Neste capítulo é feita a contextualização do trabalho de pesquisa e a apresentação da estrutura do documento. Será apresentado os objetivos propostos e alcançados, o planeamento de trabalho feito para ambas as entregas (Intermédia e Final) e as contribuições que resultaram do desenvolvimento da dissertação.

1.1 Contexto

As redes definidas por software (SDN) vieram mudar o paradigma tradicional das redes tracionais trazendo uma rede programável, escalável e com uma maior capacidade de gestão. O paradigma da rede SDN faz a separação entre o plano de controlo com o plano de dados o que resulta na facilidade de configuração e gestão das alterações numa rede, desta forma o controlador fica responsável por decisões de encaminhamento quando estes não podem ser tratados pelos outros componentes, também permite a implementação de melhores e mais consistentes políticas de controle de uma forma uniforme para toda a rede através do controlador.
Com a introdução de novos elementos na rede e da alteração da arquitetura, a segurança das redes SDN passou a ser um ponto de preocupação pois trouxe novos desafios e vulnerabilidades que podem levar a consequências mais graves para a rede. Entre os desafios mais relevantes temos o problema dos controladores centralizados que introduzem pontos únicos de falha (Single Point of Failures), sendo vulneráveis a ataques de negação de serviço distribuídos (DDos). Acrescem ainda os problemas com a privacidade dos dados, autenticação, integridade, e o acesso dos dados na rede por terceiros. Neste sentido de forma a ultrapassar estes desafios de segurança, uma escolha objetiva do controlador revela-se ser um ponto muito importante, onde deve-se levar em conta o seu funcionamento em modo cluster de modo a ultrapassar os pontos únicos de falhas, as informações das vulnerabilidades que este apresenta, as suas funcionalidades e mecanismos de segurança.

1.2 Objetivos

Este trabalho, centra-se no desenvolvimento de conhecimentos aprofundados da rede SDN, os seus protocolos, sua arquitetura e conceito, tendo como objetivo final o desenvolvimento e validação de uma framework de segurança para redes SDN, em cenários de redes 5G.

De forma a alcançar o objetivo final do desenvolvimento e sua validação propõe-se:

1. Estudo aprofundado das redes SDN.
    - Análise do estado da arte
    - Estudo da arquitetura, protocolo de comunicação, e componentes da rede SDN.

2. Desenho e validação de Framework de segurança
    - Identificação das vulnerabilidades e ameaças da rede SDN.
    - Identificação e definição de mecanismos de proteção para componentes das redes SDN
    - Identificação definição de mecanismos que garantem a privacidade dos dados.

3. Implementação e validação de funcionalidades da Framework de segurança.
    - Implementação de mecanismos que asseguram a privacidade dos dados.

4. Avaliação da Framework de segurança implementada.
    - Verificar o impacto dos mecanismos implementados em situações de ataque a rede.

1.2.1 Objetivos Finais

. Ao longo do trabalho os objetivos 2 e 4 da secção 1.2 definidos até a entrega intermédia tiveram de ser reformulados, onde foi encontrado limitações nas soluções de segurança em um dos controladores(ONOS) onde só foi identificado os possíveis mecanismos de proteção, neste sentido os objetivos finais foram adaptados onde não foi desenhado uma framework de segurança mas sim aproveitando os mecanismos já existentes no controlador foi desenvolvido um guia atualizado de como pode-se implementar um cenário com mecanismos que asseguram a privacidade de dados.

5. Estudo aprofundado das redes SDN.
    - Análise do estado da arte
    - Estudo da arquitetura, protocolo de comunicação, e componentes da rede SDN.

6. Desenho e validação de Framework de segurança
    - Identificação das vulnerabilidades e ameaças da rede SDN.
    - Identificação de mecanismos de proteção para componentes das redes SDN.

7. Implementação e validação de funcionalidades da Framework de segurança.
    - Implementação de mecanismos que asseguram a privacidade dos dados.

8. Avaliação dos mecanismos de segurança implementado.

1.3 Estrutura e Organização do Documento

O restante do documento está organizado da seguinte forma:

Capítulo 2:

- Apresenta-se o estado da arte com foco no conceito de redes SDN, sua arquitetura, e todos os outros elementos que a constituem.

Capítulo 3:

- Neste capítulo é apresentado a segurança em redes SDN, identificando as suas vulnerabilidades e ameaças, seguido da avaliação das mesmas de modo a classificar as que apresentam um risco elevado para a rede e fazer a relação com a escolha de um controlador.

Capítulo 4:

- Neste capítulo é apresentado o processo de desenvolvimento e implementação dos mecanismos que garantem a privacidade das comunicações nas interfaces. É apresentado em detalhes o processo de configurações desde a criação do cluster as configurações de segurança implementadas.

Capítulo 5:

- Neste capítulo são apresentados o processo de validação e análise do impacto dos mecanismos de segurança implementado e as estratégias de validação utilizadas.

Capítulo 6:

- Neste capítulo são apresentadas as conclusões da dissertação, os desafios encontrados ao longo da sua realização e o trabalho futuro.

1.4 Planeamento

Esta tese é marcada por três momentos importantes:

- **M1** (21/01/2021) - Entrega intermédia da tese.
- **M2** (28/03/2021) - Foi submetido o artigo "*CROCUS: An objective approach for SDN controllers' security assessment* "e aceite na conferência EAI SecureComm 2021, 17th EAI International Conference on Security and Privacy in Communication Networks.
- **M3** (30/10/2021) - Entrega final da tese.

Foram definidas um total de sete (7) tarefas até o final da entrega intermédia da dissertação:

Tabela 1-Tarefas proposta para entrega intermédia

Tarefa	**Descrição**
T1	Análise do estado da arte da arquitetura de redes SDN, os controladores existentes, e outros mecanismos que assegurem o funcionamento da rede.
T2	Análise do estado da arte dos mecanismos de segurança para redes SDN, as vulnerabilidades e ameaças existentes.
T3	Descrição e desenho preliminar de um cenário pratico, para avaliação de segurança.
T4	Testes no cenário prático com recursos a emuladores SDN
T5	Implementação da Framework de segurança com os mecanismos de segurança propostos.
T6	Avaliação Final da Framework de segurança implementada segundo os objetivos definidos.
T7	Escrita do relatório Final para efeitos da dissertação

O cronograma de trabalho até a entrega intermédia está representado no diagrama de Gantt:

Tabela 2-Entrega intermédia

Tarefa	Data Início	Duração/Dias	Data da Conclusão	Estado
T1	21-09-2020	21	12-10-2020	Concluído
T2	13-10-2020	31	13-11-2020	Concluído
T3	16-11-2020	25	11-12-2020	Atrasado
T4				Por fazer

Tabela 3- Cronograma da entrega Final da tese

Tarefa	Data Início	Duração/Dias	Data da Conclusão	Estado
T5	01-03-2021	31	01-04-2021	Por fazer
T6	02-04-2021	30	02-05-2021	Por fazer
T7	03-05-2021	31	03-06-2021	Por fazer

O cronograma de trabalho sofreu alterações onde a entrega foi estendida para época especial (30/10/2021). Na segunda fase do trabalho a seguir à entrega intermédia, foi alterado a tarefa anteriormente proposta onde foi adicionado a escrita do artigo, como mencionado na secção 1.2 as tarefas foram alteradas onde foi encontrado limitações nas soluções de segurança em um dos controladores em específico no ONOS SecurityMode onde após vários testes, validações e esclarecimentos da comunidade do ONOS foi possível concluir que este mecanismo não pode ser implementado pois foi uma funcionalidade que foi descontinuada o que provocou atrasos durante esta fase de testes do SecurityMode. A fase de descrição e implementação do cenário prático para avaliação de segurança foi um dos processos mais demorados do trabalho pois houve imprevistos em relação a validação do SecurityMode do ONOS o que levou a reformulação do cenário e os mecanismos que podem ser adotados para melhorar a segurança do controlador, o que normalmente é demorado e exige um processo de tentativa e erro para as configurações. A alteração do cronograma pode ser verificada na Tabela *4*.

Tabela 4-Tarefas Realizadas

Tarefa	**Descrição**
T1	Análise do estado da arte da arquitetura de redes SDN, os controladores existentes, e outros mecanismos que assegurem o funcionamento da rede.
T2	Análise do estado da arte dos mecanismos de segurança para redes SDN, levantamento das vulnerabilidades e ameaças existentes.
T3	Escrita da tese para entrega intermédia.
T4	Escrita do artigo para submissão.
T5	Descrição e Implementação do cenário prático para avaliação de segurança do controlador.
T6	Escrita da tese para entrega final.

Tabela 5-Entrega Intermédia atualizada

Diagrama de Gant - Entrega Intermédia				
Tarefa	**Data Início**	**Duração/Dias**	**Data da Conclusão**	**Estado**
T1	21/09/2020	40	31/10/2020	Concluído
T2	01/11/2020	35	06/12/2020	Concluído
T3	07/12/2020	42	18/01/2021	Concluído

Tabela 6-Entrega Final

Diagrama de Gant - Entrega Final				
Tarefa	**Data Início**	**Duração/Dias**	**Data da Conclusão**	**Estado**
T4	**02/02/2021**	52	26/03/2021	Concluído
T5	01/04/2021	110	20/07/2021	Concluído
T6	02/08/2021	88	29/10/2021	Concluído

1.5 Contribuições

O esforço de pesquisa da dissertação teve um conjunto de contribuições:

1. Escrita e publicação de um artigo científico na conferência EAI SecureComm, que pode ser consultado no anexo A. Bruno Sousa, João Vilela, Carlos Silva, “CROCUS: An objective approach for SDN controllers security assessment”, setembro 2021.

2. Proposta de metodologia para a escolha de controladores SDN, na fase de desenho e na fase de produção tendo presente informação de vulnerabilidades.

3. Documentação do processo de implementação de segurança das diversas interfaces do controlador ONOS (implementação do protocolo TLS).

4. Disponibilização do código produzido e configurações em modo Open-Source (repositório git, acessível em https://github.com/carlosfelix12/ONOS).

2 Redes Definidas por Software

Neste capítulo é apresentado o estado da arte das redes SDN, o seu conceito, arquitetura, componentes, os controladores, e as tecnologias utilizadas durante a realização do trabalho. Pretende-se com este capítulo ajudar a perceber como funciona uma rede SDN, os controladores utilizados atualmente e ser um apoio de forma a perceber os próximos capítulos.

2.1 Conceito

As redes de computadores foram baseadas na utilização de equipamentos que pudessem trabalhar de forma descentralizada, desta forma cada equipamento quando adicionado à rede contém aplicações com todos os comandos necessários à integração. Este tipo de abordagem, pode levantar problemas com utilização de aplicações proprietários que não podem ser substituídos dificultando o processo de gestão e manutenção deste tipo equipamentos na rede.
Com o passar dos tempos aumentou a procura de soluções que pudessem simplificar a capacidade de gestão e a aplicação de novas configurações nos diversos dispositivos de rede. Uma das propostas sugeria que o plano de dados e o plano de controlo e ficassem separados fisicamente, alterando desta forma a conceção inicial dos elementos de rede onde o funcionamento era independente e distribuído [1]. Esta proposta serviu como princípio básico para o aparecimento das redes definidas por software.
O conceito de rede SDN (Software Defined Network) teve origem na universidade de Stanford no final do ano 2006, onde Martin Casado, apresentou o projeto denominado de Ethane, tendo como objetivo a centralização e simplificação na forma como os mecanismos de gestão e segurança eram executados. A flexibilidade do Ethane permitia aos administradores implementar regras de tráfego, diretamente e de forma global para toda rede, em vez de executar o mesmo processo individualmente em cada equipamento existente. Esta centralização é assegurada com a separação do plano de controlo e o plano de dados por um controlador através de interfaces entre os planos, desta forma a configuração da rede passa ser feito por APIs.

2.2 Arquitetura

A implementação da rede SDN levou com que houvesse alterações na arquitetura das redes tradicionais onde anteriormente cada dispositivos tinham instalados os seus próprios serviços, plano de dados, e plano de controlo o que pode levar a uma maior carga de processamento nestes dispositivos e complexidade na arquitetura da rede. Esta alteração na arquitetura levou com que na implementação das redes SDN cada dispositivo ficasse responsável apenas pelo plano de dados de forma a garantir o encaminhamento de informação, enquanto os planos de aplicações e controlo são centralizados.
A Figura *1*a) representa a arquitetura SDN enquanto a b) representa as redes tradicionais.

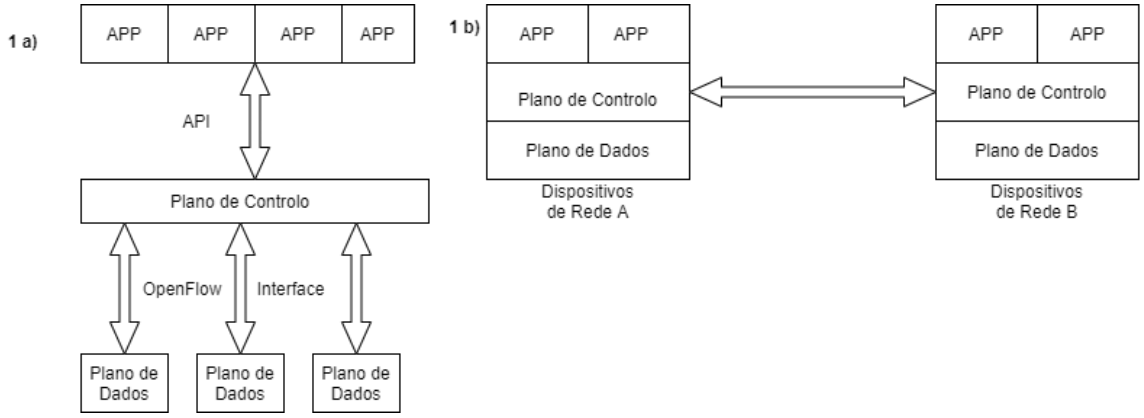


Figura 1-Rede SDN 1a) vs Rede Tradicional 1b) baseado em *[2]*

A principal característica da arquitetura de uma rede SDN, é a separação dos diferentes planos, com o foco na centralização do plano de controlo e aplicação o que permite uma maior flexibilidade nas configurações das aplicações de rede. A sua arquitetura de funcionamento divide-se em três níveis separados propostas pela Open Networking Foundation (ONF) [3]:

Plano de Controlo:

- Camada responsável pelo controlo de toda a rede, onde são enviados para o controlador pedidos de forma a ser tomadas medidas de encaminhamento de dados determinas por um fluxo, permitindo que os administradores possam configurar toda a rede e criar aplicações para a monitorização de tráfego, criação de firewalls, e configuração de políticas de seguranças.

- É importante realçar que dependendo do controlador este poderá suportar uma arquitetura distribuída.

Plano de Dados:

- Plano responsável pela abstração dos componentes físicos da rede como switches que recebem e gerem o tráfego, firewalls ou outros mecanismos que auxiliam na gestão da rede. Desta forma podemos ter um plano apenas responsável pelo encaminhamento dos pacotes.

- As decisões de encaminhamento são baseadas nas informações dos pacotes, ou seja, em caso de um equipamento não tenha regras definidas para um pacote, este geralmente é enviada para o controlador onde este é questionado sobre as medidas de encaminhamento pretendido, ou seja, ser encaminhado ou descartado.

Plano de Aplicações:

- Plano com aplicações que são responsáveis pela gestão da rede em tempo real podendo desta forma executar tarefas como encaminhamento, balanceamento de carga, políticas de segurança, sendo este gerenciamento feito através de uma ou várias APIs.

2.3 Interfaces

Na arquitetura SDN pode-se ter duas arquiteturas de implementação a centralizada ou distribuída dependendo do tipo de controlador utilizado, a topologia da rede, desempenho, e os objetivos da organização. O controlador faz uso de APIs responsáveis por enviar encaminhar informações para os restantes componentes e planos em uma arquitetura SDN.A *Figura 2* representa as diferentes interfaces existentes em uma arquitetura SDN.

Northbound interface:

- Esta interface aumenta a flexibilidade de toda rede pois permite configurá-la de uma forma geral através de um canal único. As interfaces Northbound estabelecem uma ligação entre os controladores e as aplicações, onde são definidas através de uma API (Aplication Program Interface), na qual os administradores podem definir comandos para receber dados de equipamentos, definir regras de encaminhamento entre outras informações.

Ao contrário do Openflow (protocolo mais utilizado na Southbound interface), na Northbound interface não existe um protocolo ou API padronizado que organizações ou utilizadores poderão utilizar, uma das razões para isto é a diversidade de aplicações utilizadas no plano de aplicações cada um com requisitos próprios. De forma a continuar o desenvolvimento da SDN a ONF criou a Northbound Interface Work Group (NBIWG) [4] com a finalidade de ter uma padronização desta interface. Apesar de ter sido fundado em 2013 até a data corrente ainda não foi publicado nenhuma proposta de standard para a NBI.

Southbound interface:

- Representa a interface que faz a comunicação entre os planos de dados, com o plano de controlo, ou seja, permite a comunicação entre os dispositivos responsáveis por ações de encaminhamento encontrados no plano de dados com os componentes do plano de controlo. A Southbound veio resolver o problema da heterogeneidade dos elementos da rede, onde temos diferentes fabricantes de equipamentos com arquiteturas de fabrico próprio e que suportam diferentes linguagens, este problema é solucionado ao oferecer uma interface padronizada e aberta para todos os diferentes componentes e vendedores. Existe uma variedade de protocolos [5], nos quais esta interface pode basear-se tais como: P4, ForCES, NETCONF, SNMP, POF, OpFlex, OpenState e outros como OpenFlow, OVSD [6]. Ao suportar diversos protocolos podemos manter um maior nível de compatibilidade entre o controlador e os equipamentos. De realçar que uma grande maioria dos controladores suportam apenas o protocolo OpenFlow sendo que este é o protocolo sugerido pela Open Networking Foundation [7].

Eastbound/Westbound interface:

- São interfaces especiais encontrados em cenários de utilização de controladores distribuídos, conforme pode ser verificado na Figura *2*. Tanto a Westbound como a Eastbound são necessários para a comunicação entre os controladores. A principal vantagem destas interfaces é que aumenta a robustez da rede SDN em caso de falha no controlador pois possui mecanismos que permitem a replicação dos dados. Outras funções que estas interfaces desempenham passam pela monitorização, o transporte de dados, e notificações aos controladores. A Eastbound e Westbound interface à semelhança as outras interfaces também utiliza suas próprias APIs.

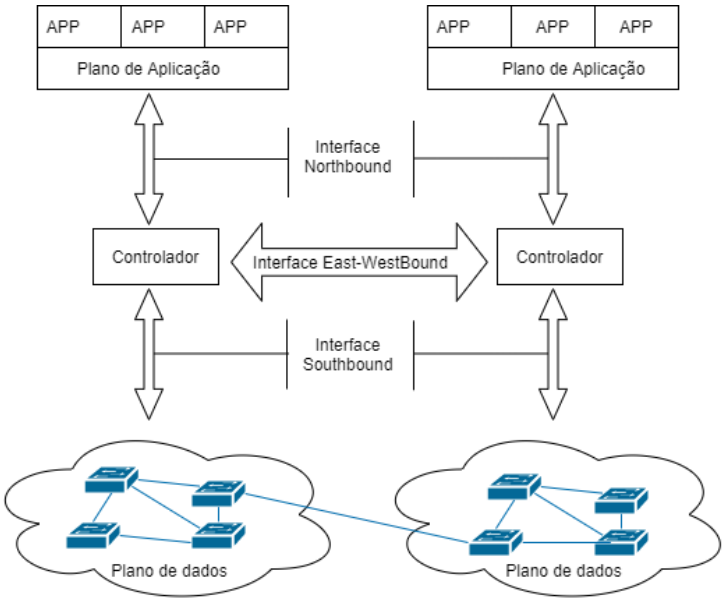


Figura 2-Interfaces nas redes SDN baseado em *[8]*

2.4 Controladores

Os controladores SDN funcionam como o núcleo de uma rede SDN, pois é através deles que são focadas todas as operações de rede e informação tornando em um elemento crítico. Desta forma a sua implementação deve levar em consideração a disponibilidade dos dados recolhidos, e os recursos que o controlador fornece às aplicações no plano de aplicações. A implementação de um controlador não segue uma norma em específico, mas deve garantir a comunicação entre o plano de dados e o plano de aplicações utilizando as interfaces Southbound e Northbound.
Atualmente existem vários controladores SDN [9], cada uma com características e arquiteturas distintas [10]. Esta diversidade de controladores e características, pode dificultar no momento da escolha de um controlador para um determinado projeto.
No estudo comparativo [11] feito é apresentado uma tabela analisando um total de 34 controladores, não seguindo um critério em específico, mas sim, listando todos os controladores disponíveis no mercado e suas características, e outros controladores já descontinuados.

Algumas das seguintes propriedades, foram comparadas entre os controladores Tabela *7* tais como:

- linguagem de programação
- arquitetura.
- APIs
- documentação
- modularidade.
- plataforma e interface.

Tabela 7-Lista adaptada dos controladores do mercado SDN

Nome	#LP	#Arq	#NBI	#SBI	EBI WBI	#Plat.S	#InTf	#Licença	#ModLa
Beacon	Java	Centralizado	Ad-hoc	OpenFlow 1.0	-	Linux, MacOs, Windows	CLI, Web UI	GPL 2.0	Razoável
NOX	C++	Centralizado	Ad-hoc	OpenFlow 1.0	-	Linux	CLI, Web UI	GPL 3.0	Baixo
ONOS	Java	Distribuído	Rest, Neutron	OpenFlow 1.0, 1.3	Raft	Linux, MacOs, Windows	CLI, Web UI	Apache 2.0	Alto
POX	Python	Centralizado	Ad-hoc	OpenFlow 1.0	-	Linux, MacOs, Windows	CLI, GUI	Apache 2.0	Baixo
Ryu	Python	Centralizado	REST	OpenFlow 1.0-1.5	-	Linux, MacOs	CLI	Apache 2.0	Razoável
Floodlight	Java	Centralizado	REST, Java, RPC, Quantum	OpenFlow 1.0-1.3	-	Linux, MacOs, Windows	CLI, Web UI	Apache 2.0	Razoável
OpendayLight	Java	Distribuído	REST RESTCONF XMPP NETCONF	OpenFlow 1.0-1.3	Akka, Raft	Linux, MacOs, Windows	CLI, Web UI	Apache 2.0	Alto
ZeroSDN	C++	Distribuído	REST	OpenFlow 1.0-1.3	ZeroMQ	Linux	CLI, Web UI	Apache 2.0	Alto
HyperFlow	C++	Distribuído	-	OpenFlow 1.0	publicar e assinar mensagens	-	-	Proprietário	Razoável
OpenIRIS	Java	Distribuído	REST	OpenFlow 1.0-1.3	Protocolo Personalizado	Linux	CLI, Web UI	Apache 2.0	Razoável
Disco	Java	Distribuído	REST	OpenFlow 1.0	AMQP	-	-	Proprietário	Bom

De modo a saber quais são os melhores controladores fizeram uma análise baseado na implementação e utilização dos controladores, seguindo este modelo foram selecionados os cinco melhores controladores para serem analisados, são eles:

- OpenDaylight
- FloodLight
- Ryu
- Pox

Neste estudo foi utilizado o método Analytic Hierarchy Process (AHP) [12] que auxilia na tomada de decisões, a sua utilização é justificada por dois motivos a primeira por utilizar a priorização em par e segundo por ter um mecanismo integrado de verificação de consistência, ao longo do artigo é apresentado em detalhes a utilização deste método no estudo.

Em termos das propriedades escolhidas para comparação o processo passou por ter uma metodologia de investigação de propriedades, a segunda etapa passa por consultar as propriedades, a terceira e quarta etapa consistia em analisar artigos, e workshops técnicos sobre os controladores em estudo. Com o processo terminado os controladores foram comparados segundo:

- Interfaces
- Suporte para switch virtuais
- GUI
- Suporte para APIs
- Linguagem de programação suportada
- Modularidade
- Idade
- Suporte OpenFlow
- Documentação
- Open source
- Suporte TLS (Transport Layer Security)

Foi analisado outro estudo [13] de forma a comparar as performances dos controladores em termos da sua carga na rede, e simultaneamente comparar as propriedades que cada uma apresenta. Para isso foram verificadas informações de fontes distintas para cada propriedade, e sempre que houvesse discrepância nas informações esta propriedade não seria considerado para comparação. Seguindo este método as seguintes propriedades foram consideradas para avaliação:

Southbound interfaces.
- GUI
- Modularidade
- Documentação
- Virtualização
- Arquitetura
- Linguagem Programação

Como mencionado nos estudos acima, existem diversos controladores, muitos dos quais já foram descontinuados ou já não tem uma presença ativa na comunidade. Para este trabalho foram escolhidos através, da consulta de artigos já feitos, e dos próprios websites de cada controlador, os mais utilizados atualmente e em constante desenvolvimento, são estes:

OpenDaylight:
- É um dos controladores mais populares do mercado suportando o protocolo Openflow e um conjunto de outras Southbound APIs.

Ryu:
- É um sistema operativo para SDN, focado em fornecer um controlador centralizado com APIs bem definidos, de forma a tornar mais fácil o gerenciamento da rede e suas aplicações de controlos.

Pox:

- É um controlador open-source baseado em python, utilizado principalmente para efeitos de pesquisa meios académicos.

Floodlight:

- É um controlador de nível empresarial, que suporta um conjunto de switches do tipo OpenFlow virtual e físico, de realçar que apesar de não estar em constante atualização, ainda continua sendo um dos controladores mais utilizados.

ONOS:

- É um controlador open-source criado com o propósito de proporcionar alta disponibilidade. A sua arquitetura permite ser modular e distribuídos, dispondo do plano de controlo responsável pelo gerenciamento dos componentes de rede, e executando aplicações do plano de aplicações.

Tendo identificado os controladores, o próximo passo por determinar quais as propriedades que possam ser utilizadas para fazer a comparação entre os mesmos, para isto a abordagem foi escolher as propriedades de maior relevância com base nas que foram apresentadas nos estudos referidos anteriormente e pesquisas das caraterísticas de cada controlador, tendo sido escolhidas as seguintes propriedades:

- Modularidade
- Interfaces
- GUI
- Documentação **-** Apresentada em três níveis**:**
  - **Fraco** -documentação disponibilizado no âmbito geral)
  - **Razoável** - Descrição geral, e documentação de APIs, e das funções disponibilizadas.
  - **Bom**
- Linguagem de programação
- **Arquitetura-** Distribuída ou Centralizada.
- Suporte TLS
- Idade
- Parceiros

A Tabela *8* permite destacar que o ONOS e OpenDaylight, são as que apresentam as propriedades mais completas entre os controladores, tendo uma amigável interface gráfica para os utilizadores, ambos possuem uma comunidade open-source e parceiros tecnológicos que contribuem constantemente para o seu desenvolvimento, e ideias. Outra característica que ambos apresentam de relevante é a sua arquitetura distribuída, o que os torna fortes candidatos em casos de implementação real de uma rede SDN. Outro controlador a destacar é a Ryu que possui propriedades relevantes, em específico para organizações pequenos e propósitos de pesquisas, no entanto e baixa modularidade limita a sua utilização no mercado de aplicações.
Considerando as propriedades analisadas dos controladores acima apresentadas, o OpenDaylight é o mais completo, tendo em conta as características especificas do

projeto de estágio e as propriedades analisadas o ONOS será o controlador a ser utilizado.

Tabela 8-Quadro comparativo dos controladores SDN

	ONOS	**OpenDaylight**	**Ryu**	**Floodlight**	**Pox**
Southbound APIs	OF1.0, 1.3, NETCO NF	OF1.0, 1.3, 1.4, NETCONF/ YANG, OVSDB, PCEP, BGP/LS, LISP, SNMP	OF 1.0, 1.2, 1.3, 1.4, NETCO NF, OFCONFIG	OF 1.0, 1.3	OF 1.0
Northbound APIs	REST API	REST API	REST API	REST API	REST API
GUI	Baseado em Web	Baseado em Web	GUI Patch	Baseado em Web e Java	Python+ QT4
Modularidade	Alto	Alto	Médio	Médio	Baixo
Linguagem de programação	Java	Java	Python	Java	Python
Suporte TLS	SIM	SIM	SIM	SIM	SIM
Arquitetura	Distribuída	Distribuída	Centralizada	Centralizada	Centralizada
Idade	2015	2013	2013	2012	2012
Parceiros	ON.LAB, At&T, Ciena, Cisco, Ericsson, Fujitsu, Huawei,Intel, Nec,Nsf.Ntt Comunnication, Sk Telecom	Linux Foundation com associações em empresas como Cisco, IBM, NEC	Nippo Telegraph And Telephone Corporation	Big Switch Networks	Nicira
Documentação	Bom	Bom	Razoável	Bom	Fraco

2.4.1 ONOS-Open Network Operating System

O projeto ONOS (Open Network Operating System) foi iniciado no ano de 2014, e até a presente data já conta com o total de dezoito (18) versões, e com um número de membros a aumentar constantemente tendo como parceiras organizações como AT&T, Google, Cisco (ONOS 2020) [14].

O ONOS apresenta uma framework distribuído que consiste em várias aplicações de gestão de rede, desenvolvidos sobre o container Karaf OSGI, suportando vários protocolos de comunicação no plano de controlo. As informações dos estados da rede são armazenadas utilizando a base de dados da framework Atomix [15] desenvolvido para o ONOS, onde é utilizado para coordenação de tarefas entre os controladores. A sua arquitetura distribuída é dividida em domínios com um controlador por domínio, e em que todos partilham informações entre si. Esta partilha de informações é feita utilizando o mecanismo de distribuição mais apropriado, desta forma desenvolvedores podem escolher entre dois modelos [16]:

Eventualmente consistente:

- Este modelo faz a replicação total do seu estado atual para todos os nós do cluster, onde cada nó terá uma réplica completa das suas informações. Este modelo garante uma melhor performance de escrita e leitura de dados em favor da consistência, pois estes dados são guardados em memória e eliminadas após reinicialização total do cluster. Este modelo pode ser configurado para registar a hora das atualizações (ClockService) [17], garantindo que cada nó possa aplicar as atualizações de estado em ordem correta. No caso de haver nós que não estejam sincronizados, este executa em segundo plano o processo conhecido por anti-entropia que faz o monitoramento de dados inconsistente e faz as atualizações necessários realizando cópias entre as réplicas garantindo que os nós com informações replicados convertam para o estado correto.

Consistente:

- Este modelo utiliza o algoritmo de consenso RAFT [18] para gestão e consistência das informações replicadas entres os nós do cluster garantindo que quaisquer alterações feitas no mapa de memória sejam apresentadas e partilhadas entre todos os nós. Por exemplo num modelo consistente cada mapa de memória é partilhado igualmente pelos nós dos clusters, em caso de falha num dos nós, as informações não são perdidas pois estão replicados em outros nós da rede.

A Figura *3* representa a replicação de dados entre os nós. O ONOS utiliza o modelo eventualmente consistente para gestão de dispositivos de rede, listas e hosts. O domínio do dispositivo (controlador "Master") utiliza o modelo consistente para gestão dos dados.

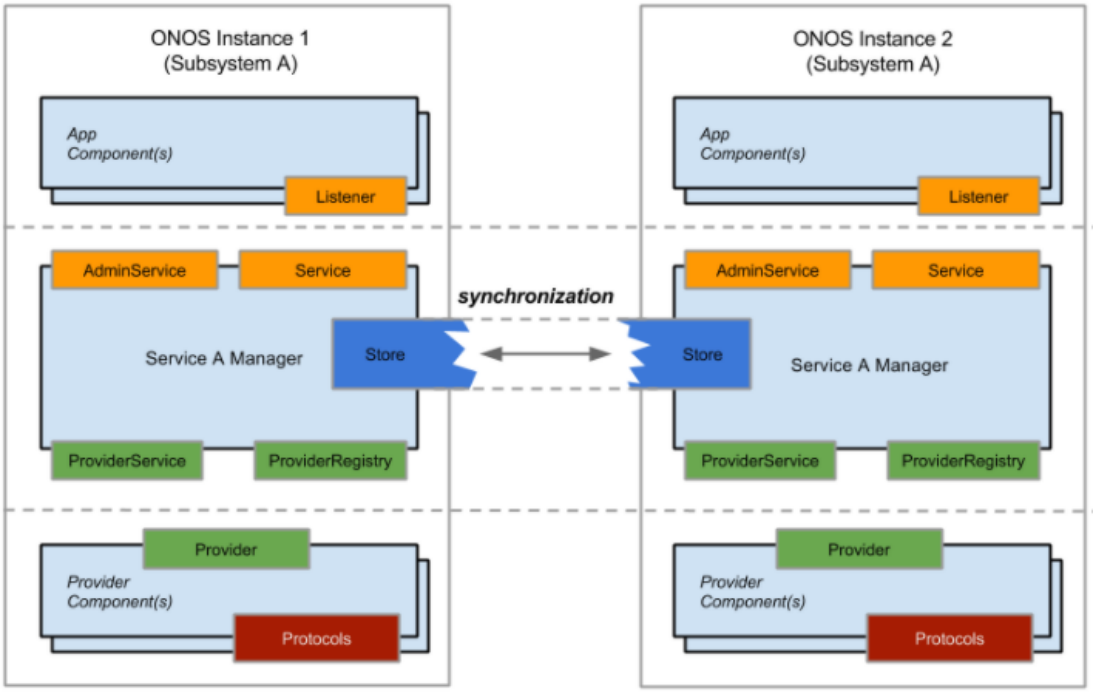


Figura 3-Sincronização entre os nodos do controlador ONOS *[19]*

2.5 Mininet

O mininet [20] é um emulador de rede, do tipo CBE (Container-Based Emulation) que faz a virtualização a nível de processo onde os recursos como estrutura de dados, sistemas de ficheiros e kernel são partilhados tornando o processo de emulação mais rápida e escalável. Com o mininet pode-se emular os seguintes componentes:

Switches:

- São os dispositivos com suporte ao protocolo OpenFlow, que são configuradas e geridas pelo controlador.

Controladores:

- Os controladores podem estar localizados em qualquer parte da rede (rede física ou virtual), e devem estar conectados aos switches quando estes estiverem ligados.

Links:

- É um link Ethernet virtual que atua como cabo de rede conectado a duas interfaces virtuais, com a função de enviar pacotes entre as interfaces.

Para controlar e gerir os dispositivos emulados o Mininet permite através de uma CLI (Command Line Interface) gerir toda rede controlar e todos os dispositivos. Também permite o desenvolvimento de topologias de redes personalizadas através de uma API (Application Programming Interface) para linguagem de programação python onde podem ser utilizados os comandos representados naTabela *9*:

Tabela 9-Comandos mininet

Comandos	**Descrição**
addSwitch	Adicionar um Switch a rede.
addLink	Adiciona um link bidirecional numa topologia de rede
addhost	Adiciona um host a rede.
stop	Parar a rede.
start	Inicializa a rede.
hosts	Lista todos os hosts da rede.
pingAll	Testa toda a conectividade da rede.
net.host:	listar todos os hosts da rede.

O Mininet apresenta algumas vantagens como:

- Velocidade de arranque.
- Maior escalabilidade – consegue suportar o aumento de hosts e switches.
- Maior largura de banda- geralmente na ordem dos 2 Gbps.
- Permite testes de topologias de rede complexos, sem ter a necessidade de conectar a rede física.

Apesar de apresentar estas vantagens o mininet possui algumas limitações [21] são elas:

- Utiliza um único kernel Linux para todos os hosts- Desta forma não é possível executar softwares que dependem de outros kernels de sistemas operativos.
- O mininet isola por defeito a topologia de rede da rede local e internet.
- O mininet não escreve comandos OpenFlow para o controlador.
- Não pode exceder a utilização do CPU ou largura de banda de um único servidor.

A Figura *4* representa uma rede física emulada pela mininet.

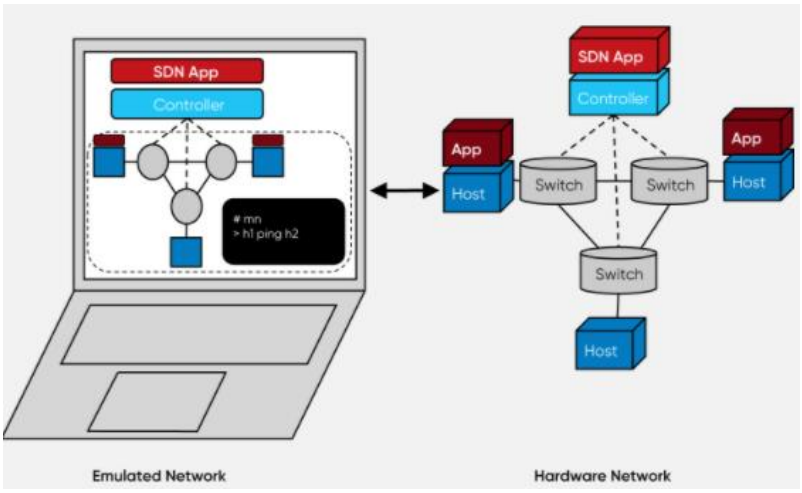


Figura 4-Rede Física emulada *[22]*

2.6 Atomix

O Atomix é uma framework desenvolvida como parte do projeto ONOS [23] para a implementação de sistemas distribuídos que sejam tolerantes a falhas. O ONOS como já visto na secção 2.4.1 é um dos controladores mais populares e utiliza o Atomix para o seu armazenamento de dados partilhados para uma implementação logicamente centralizado, mas fisicamente distribuída. O Atomix implementa o protocolo de consenso Raft, para o qual o número e a localização das instâncias do Atomix não podem ser alterados. A partir da versão 1.14 e posteriores do ONOS, o Atomix e ONOS foram separados, o que permite ao ONOS ser atualizado e escalado horizontalmente de forma mais fácil o que anteriormente não acontecia.

O Atomix apresenta cinco conceitos [24] importantes na sua estrutura são elas:

Cluster:

- Consiste em um conjunto de nodos que se comunicam entre si, utilizando diferentes protocolos para partilha do estado. Na prática quando um nodo é iniciado, dependendo da sua configuração este conecta-se a um conjunto de pares, e para isso pode ser escolhido entre três protocolos de descoberta: 1-Bootstrap, 2-Multicast, 3-DNS.

Protocolos de Replicação:

- São protocolos específicos para sistemas distribuídos implementados pelo Atomix para partilhar o estado entre os nodos. O Atomix utiliza vários protocolos diferentes para replicação do seu estado descritos anteriormente, podendo variar entre consistência fraca a forte. Os três protocolos que podem ser utilizados são
    - **Raft** – É um protocolo de consenso utilizado pelo Atomix para consistência forte. Este faz a gerência dos logs replicados das mudanças nos primitivos. Isso é feito elegendo um líder e replicando as mudanças de forma síncrona para os outros nodos “seguidores”. A consistência é mantida elegendo apenas os líderes que tenham todas as alterações mais recentes. No entanto, uma propriedade importante do protocolo Raft é que este pode avançar quando a maioria do cluster esta disponível. No caso de uma partição de rede, apenas o lado da maioria da partição continuará a progredir.
    - **Primary-backup-** É um protocolo de replicação mais simples com propriedades de consistência mais fracas. O protocolo Primary-backup pode suportar a perda de todos, as alterações podem ser replicadas de forma síncrona ou assíncrona para qualquer número de nodos. Isso torna o protocolo Primary-backup muito mais adequado para casos de uso de alto desempenho.
    - **Gossip-** É um protocolo de replicação do tipo eventualmente consistente que depende da partilha periódica de informações e de forma aleatória entre os nodos.

Grupos de Partição:

- São um conjunto de instâncias do protocolo Raft ou Primary-Backup utilizado para replicar instâncias primitivos especificas. Quando um cluster é formado, os membros devem definir o conjunto e partições de grupo na qual vai participar.

Primitivas Distribuídas:

- Estão no centro do processo de como o Atomix faz a replicação do estado e coordena as alterações de estados em sistemas distribuídos

Protocolos Primitivos:

- Os primitivos do Atomix são replicados e armazenados com base no protoloco implementado pelo grupo de participação na qual foram atribuídos. Cada primitivo pode ser armazenado numa participação Raft ou Primary-backup, e o comportamento específico de cada protocolo pode ser definido nas configurações do protocolo de primitivos.

2.7 Protocolo Transport Layer Security-TLS

O *Transport Layer Security* -TLS é um protocolo do tipo cliente-servidor que roda em cima dum protocolo de transporte confiável e orientado para conexão, como por exemplo o Transmission Control Protocol (TCP) . Tem como principais objetivos fornecer integridade onde uma entidade externa não pode alterar os dados e confidencialidade onde é garantido que uma entidade externa não possa ler os dados partilhados.
Uma comunicação TLS é estabelecida em duas fases, onde na primeira fase os pares envolvidos na comunicação fazem autenticação entre si e fazem negociações dos parâmetros como algoritmos de encriptação e chaves secretas, na segunda etapa os dados são trocados utilizando os parâmetros anteriormente estabelecidos, esta troca de mensagens e autenticação necessários para estabelecer uma sessão TLS é possível através de uma troca previamente estabelecida sendo denominada de *handshake.* Atualmente o protocolo TLS pode ser implementado utilizando a versão 1.2 ou 1.3, sendo que em 2018 foi introduzido a versão 1.3 [25] que apresenta maior privacidade segurança e performance.

2.7.1 Diferenças entre TLS 1.2 e TLS 1.3

Com a introdução da versão 1.3 do protocolo TLS houve mudanças significativas [26] em relação à versão 1.2. Esta ainda pode ser implementada com segurança, mas existe o risco de vulnerabilidade explorarem partes opcionais do protocolo e algoritmos que não estejam atualizados, desta forma o TLSv1.3 faz a remoção das opções vulneráveis, tais como:

- Cifras que não oferecem uma encriptação autenticada.
- Utilização dos algoritmos de hash MD5 e SHA-223 com assinaturas.
- Troca das chaves *Rivest-Shamir-Adleman*-RSA e Static *Diffie-Hellman*.

Utilização dos algoritmos Diffie-Hellman (DH) e Elliptic Curve Diffie-Helman (ECDHE):

- O TLSv1.3 não suporta o RSA porque este utiliza uma única chave privada durante todo o processo de handshake o que em caso de acesso indevido, todas as comunicações feitas utilizando esta chave estarão comprometidas. O que o TLSv1.3 propõem é a utilização dos algoritmos DH e ECDHE que gera uma chave em cada fase do handshake.

2.8 Finalização do Capítulo

Neste capítulo foram introduzidos o conceito e os componentes importantes que vão ser abordados ao longo do trabalho. Em concreto, foi feito uma breve introdução ao conceito da rede SDN, as alterações na arquitetura de uma rede tradicional para uma rede SDN e os 3 planos que separa uma arquitetura SDN. Também foi abordado as diferentes interfaces da arquitetura SDN, e a grande variedade de controladores existentes e suas características. Foi analisado o controlador ONOS desde o conceito, e os seus modelos de mecanismos de distribuição, de realçar que o controlador ONOS foi uma parte vital na realização do trabalho. Foi introduzido a framework Atomix introduzindo o seu conceito e os componentes fundamentais que fazem parte do seu funcionamento, desde a formação do cluster aos protocolos de replicação distribuídos. Por fim foi apresentado o protocolo TLS e as principais diferenças entre as atuais versões.
No próximo capítulo é apresentado os aspetos de segurança que abrange as redes SDN.

3 Segurança

Esta secção fornece uma visão sobre os aspetos de segurança que abrange as redes SDN.A secção 3.1 faz o levantamento e detalha elementos que acrescentam um valor para a rede, podendo ser hardware, software, humano, ou ambiente físico de uma rede SDN. A secção 3.2 são apresentadas as principais ameaças genéricas que uma rede SDN pode sofrer. Estas ameaças podem estar relacionadas com os diferentes ativos apresentados anteriormente, e os diferentes planos da arquitetura SDN. A secção 3.1 explica o processo utilizado na identificação e escolha do controlador a ser utilizado, também apresenta a metodologia de avaliação das vulnerabilidades dos controladores SDN, no sentido de identificar as que têm mais impacto, e uma probabilidade maior de acontecer, desta forma podemos fazer uma análise das que precisam de mais investigação de forma a tornar os controladores SDN mais seguros.

3.1 Ativos SDN

Com o crescimento das redes SDN os aspetos da segurança tornaram-se uma prioridade pois a flexibilidade das redes SDN veio acompanhado de riscos de segurança. Como já mencionado nos capítulos anteriores as comunicações relacionadas com tomadas de decisão, passam pelo controlador tornando-o em um ponto critico pois em caso de problemas toda a rede pode ser afetada.
Tendo isto em conta foi feito um levantamento dos ativos (são todos os elementos que acrescentam um valor para a rede, podendo ser hardware, software, humano, ou ambiente físico) de uma rede SDN, bem como as ameaças genéricas e específicas que cada plano pode sofrer.
Os seguintes ativos representam os constituintes em uma arquitetura SDN:

Ativos no Plano de dados:

- Inclui todos os componentes físicos (Switches, Routers) no plano de dados de uma rede SDN.

Ativos no Plano de Controlo:

- Inclui os ativos associados ao plano de controlo podendo ser hardware (controladores, interfaces) e software (protocolos de comunicação) utilizado no processo de controlo da rede.

Ativos no Plano de Aplicações:

- Inclui aplicações utilizadas na implementação da rede, juntamente com o hardware(servidores) necessário para executar estas aplicações.

Utilizador SDN:

- Inclui todos os utilizadores que utilizam um equipamento encontrado no plano de dados de uma arquitetura SDN.

3.2 Ameaças e Vulnerabilidades

Nesta secção são apresentadas as principais ameaças genéricas que uma rede SDN pode sofrer. Estas ameaças podem estar relacionadas com os diferentes ativos apresentados anteriormente, e os diferentes planos da arquitetura SDN.É importante realçar que as ameaças e vulnerabilidades apresentadas estão representadas com nomenclatura em inglês pois alguns nomes não possuem uma tradução direta.

3.2.1 Ameaças SDN Genéricas

As principais ameaças genéricas às redes SDN são:

Single Point of Failure (SPOF):

- A utilização de um controlador central nas soluções de controladores de uma rede SDN, faz com que qualquer ameaça que possa levar à falha do mesmo, possa comprometer toda rede. Esta ameaça pode ser solucionada utilizando soluções de controladores distribuídos, de relembrar que nem todos os controladores SDN permitem uma configuração distribuída [27].

Denial of Service (DoS):

- Ataques DoS estão relacionados com ataques intencionais de sobrecarga em redes SDN, com o objetivo de reduzir ou interromper o mesmo. Este tipo de ameaça pode ocorrer em todos os níveis da arquitetura de uma rede SDN, no plano de dados este pode ser causado sobrecarregando a largura de banda ou os recursos dos elementos da rede, isto muitas vezes pode ser feito utilizando recursos como botnets que sobrecarregam um switch com tráfego excessivo de pacotes. O plano de controlo por ser a central de uma rede SDN, onde podemos ter controladores que não suportam uma configuração distribuída, tornando-o num único ponto de falha [28].

Vulnerabilidade nos Canais de comunicação:

- A segurança na comunicação entre o plano de dados e plano de controlo é assegurada com a utilização do TLS com o protocolo OpenFlow, mas, contudo, a Open Network Foundation não exige que o TLS seja implementado. A não obrigatoriedade na utilização de um protocolo de encriptação na comunicação permite ter controladores vulneráveis a oportunidades de ataques como Man in The Middle (MiTM)., ou até mesmo levar a problemas de interoperabilidade entre controladores.

3.2.2 Ameaças SDN no plano de dados

O plano de dados está localizado na parte inferior da arquitetura SDN onde contém muitas dispositivos conectados entre si e responsáveis pelo encaminhamento de pacotes. são apontadas as seguintes vulnerabilidades:

Integridade dos Componentes:

- Consiste em comprometer um componente SDN como controladores, switches, routers, de modo a alterar dados da rede como as regras de fluxo. Um outro exemplo passa por forjar pedidos de controladores de forma a ganhar controlo e redefinir a topologia de redes.

Side-Channel Attack:

- Este tipo de ataque passa por extrair informações existentes nas tabelas de regras de fluxos nos elementos da rede como switches de forma a encaminhar pacotes. Um atacante executa um ataque side channel para determinar se já existe uma regra na tabela de fluxo, isto é possível analisando o tempo de estabelecimento de uma nova conexão à rede, pois um switch compatível com OpenFlow e SDN demora mais tempo processando um pacote que pertence a um novo fluxo [29].

Flooding Attack:

- Consiste em comprometer um dos componentes de rede SDN, por exemplo enviar uma quantidade enorme de pacotes de forma a esgotar os recursos de processamento de um switch. E com um dos componentes comprometido, por exemplo um switch o atacante pode inundar intencionalmente o controlador com tráfego de dados esgotando os seus recursos.

Exploits:

- Consiste em explorar vulnerabilidades com o objetivo de causar o mau funcionamento, escuta, ou paragem dos serviços da rede e comprometer os dados. De realçar que esta ameaça pode ocorrer em qualquer um dos planos da arquitetura de uma rede SDN.

3.2.3 Ameaças SDN no plano de aplicação

No plano de Aplicação são apontadas aas seguintes vulnerabilidades:

API Exploitation:

- Como já mencionado, a Northbound interface é responsável pela comunicação entre o controlador com o plano de aplicação e a Southbound interface fornece a comunicação entre o controlador e os elementos de controle da rede (switches, routers). De realçar que em cada uma das interfaces pode ser utilizados diferentes APIs, com diferentes vulnerabilidades associadas permitindo ao atacante, por exemplo recolher informações da tabela de fluxos explorando vulnerabilidades da Northbound interface (NBI).

Traffic sniffing:

- Este tipo de ataque acontece geralmente na comunicação entre uma aplicação no plano de aplicação com o plano de controlo de forma a ganhar acesso ao controlador ou credenciais no plano de aplicação. Na prática isto é possível explorando um canal de comunicação não encriptado entre os planos. De realçar que o Traffic sniffing pode ser utilizado para propósitos legítimos de monitorização da rede, e que neste caso não representa uma ameaça. São necessárias implementações de políticas de segurança por parte do controlador, como controlo de acesso, autenticação das aplicações entre outros, pois as aplicações terceiros utilizados no plano de aplicações poderão ter acesso a informações confidenciais.

Exploração de vulnerabilidades das aplicações:

- Outra via de ataque são as aplicações de terceiros utilizadas na rede SDN que podem apresentar vulnerabilidades e servir de entrada de um possível ataque. Sendo assim importante a implementação de controlos de autenticação e níveis de autorização para aplicações de modo a limitar a sua exposição com o controlador.

3.2.4 Ameaças SDN no plano de controlo

Na arquitetura de uma rede SDN, o plano de controlo é o ponto crucial na rede com impacto direto de segurança nos outros planos no caso do controlador for comprometido. No plano de Controlo são apontadas aas seguintes vulnerabilidades:

Software exploits:

- Esta ameaça explora vulnerabilidades do software de modo a causar mau funcionamento, ou paragem do serviço. Estes exploits podem ser encontrados em todos os planos de uma arquitetura SDN. Um exemplo em casos de switches no plano de dados os exploits explorados podem criar ou forjar tráfego na rede de forma a esgotar os recursos de outros switches e os controladores conectados.

Spoofing:

- Neste plano um ataque spoofing tem como alvo a identidade do controlador, desta forma o atacante poderá forjar a identidade do controlador e interagir com os outros elementos da rede controlados pelo controlador infetado.

Ataque de aplicações terceiros:

- Um atacante poderá ganhar acesso à rede através do plano de aplicação explorando alguma vulnerabilidade das aplicações e aceder a dados sensíveis sobre a rede o que poderá levar a um ataque direto ao plano de controlo.

3.3 Metodologia de avaliação de vulnerabilidades em controladores

Esta secção resume a metodologia de avaliação das vulnerabilidades dos controladores SDN, no sentido de identificar as que têm mais impacto, e uma probabilidade maior de acontecer, desta forma podemos fazer uma análise das que precisam de mais investigação de forma a tornar os controladores SDN mais seguros.
Os processos do método de avaliação de vulnerabilidades foram divididos em três passos, resumidos na Tabela *10*:

Tabela 10-Fases da Metodologia de avaliação

Etapas	Descrição
1	Identificar os controladores.
2	Recolher as vulnerabilidades.
3	Classificação final das vulnerabilidades reportadas.

3.3.1 Passo 1-Identificar os controladores

A adoção de redes SDN leva à disponibilidade de vários controladores, que possuem diversas características em termos de desempenho e suporte a segurança. Os controladores SDN são um alvo comum para ataques na rede pois o seu comprometimento leva a prejudicar toda a rede. Assim, a escolha de um controlador SDN deve ser um processo meticuloso desde o início das fases (da concepção à produção). O estudo realizado no Anexo A propõe, um mecanismo denominado de "CROCUS" para permitir uma avaliação objetiva e de suporte de segurança dos controladores SDN. O CROCUS depende das informações fornecidas pelo sistema de pontuação de vulnerabilidade comum “The Common Vulnerability Scoring System” (CVSS v3) [30] e leva em consideração os recursos derivados de cenários com requisitos de segurança rigorosos. Os atributos de segurança necessários para a implementação e design de uma rede SDN seguro, e resiliente pode ser categorizado em três grupos: Design, Serviços de segurança, e Interface, representadas na Tabela *11*.Cada uma destas categorias representam os controlos que são utilizados para implementação da segurança , por exemplo a categoria interface avalia os controlos que o controlador disponibiliza para garantir uma comunicação segura desde o suporte a protocolos de encriptação, e controlos de autenticação para o acesso a interface gráfica do utilizador, por outro lado temos a categoria de serviços que avalia serviços ou softwares como o sistema de monitoramento de instruções (IDS) ou o sistema de prevenção de intrusão(IPS), que podem ser adicionados ao controlador para proteger a framework e as interfaces. Os recursos de segurança são avaliados em três escalas, onde 0-representa sem suporte, 1-Suporte parcial, e 2-Suporte total. Esta classificação tem por base a documentação oficial dos controladores e as recomendações dos recursos que são vistos como obrigatórios na implementação da rede SDN em ambiente empresarial [31]. Considerando um cenário de comunicação veicular suportado por múltiplas tecnologias, foi restringindo a seleção dos controladores SDN para OpenDayLight e ONOS onde os resultados mostrados (Anexo A) colocam em evidência que ambos os controladores têm

recursos de segurança relevantes para cenários que exigem uma alta disponibilidade da rede, autenticação, e uma comunicação segura entre os controladores e todos intervenientes na rede de forma proteger os dados transmitidos , destas exigências de segurança o ONOS se destaca em alguns aspetos.

Tabela 11-Métricas de segurança funcionais de CROCUS

Categoria	Métrica	Descrição
Serviços	Integração IDPS(**#IDPS**)	A integração com sistemas de prevenção de intrusão é realizada em um modo contínuo.
	Suporte AAA(**#AAA**)	A utilização de recursos requer autenticação.
	Monitoramento de recursos (**#MR**)	Os recursos são monitorizados (por exemplo, alteração na topologia)
	Logs e auditoria (**#AuD**)	As informações dos planos SDN são registadas.
Interface	Comunicação segura nas interfaces(**#CCS**)	Comunicações com o controlador SDN são protegidos (por exemplo TLS).
	Segurança das API GUI/REST **(#API**)	Interfaces que expõem o controlador SDN, em em particular, os NBI são protegidos (por exemplo, TLS) Controlos para autenticação / autorização necessária para acesso a GUI ou aplicada para as chamadas REST.
Design	Recursos de segurança(**#RS**)	Mecanismos para proteger recursos da rede
	resolução de conflitos de políticas(**#IRP**)	Esquemas para garantir que as políticas não introduzam comportamentos suspeitos
	Instâncias múltiplas do controlador-Resilencia(**#IMC**)	Suporte para cluster de forma a evitar SPoF
	Armazenamento seguro **#AS**	As informações são armazenadas de forma segura e com esquemas de verificação de integridade

A decisão final na escolha do controlador depende da abordagem que combina as vulnerabilidades dos diversos controladores (*V-So*), e outros funcionalidades de segurança relevantes (*SeSo*) agrupados em 3 categorias (Design, Interface e Serviço de segurança) representados na Tabela *11*. O rácio de solvabilidade corresponde a uma métrica composta que avalia o tempo por parte de uma entidade/comunidade responsável pelo controlador para resolver vulnerabilidades críticas. O "*rCRic*" em relação ao controlador c é determinado pela equação da Figura *5* que considera a média dos dias necessários para resolução das vulnerabilidades "*uptDayCric*", que tem um determinado número de vulnerabilidades críticas nVulCric. Uma vulnerabilidade critica tem uma pontuação CVSS acima de quatro (CVSSS3 $\geq$4). O "*TOTuptDayCri*" representa o somatório das médias de dias da resolução das vulnerabilidades críticas em todo os

controladores. Valores inferiores do rCri são mais relevantes pois representam que uma vulnerabilidade leva menor tempo para ser resolvido.

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

Figura 5-Equação rCric- rácio de solvabilidade de vulnerabilidade críticas

3.3.2 Passo 2- Recolher as vulnerabilidades

Para o levantamento das vulnerabilidades foram utilizados dois websites [32] [33], por serem referência no mercado, e pelas características avaliadas sendo que o primeiro apresenta mais informações por isso foi utilizado como fonte principal, mas ambos disponibilizam uma lista de vulnerabilidades reportadas para diferentes produtos e vendedores.
As vulnerabilidades reportadas utilizam o framework de classificação CVSSv3 pois não existe muita diferença significativa com a versão v3.1, que depende sobretudo das informações de vulnerabilidades e aspetos de esclarecimentos. O CVSS v3 combina diferentes categorias de métricas numa escala de 0 a 10, onde o primeiro é classificado como informativo e o último como crítico. A vulnerabilidade ambiental refere-se ao ambiente onde a vulnerabilidade pode ocorrer, incluindo informações sobre os privilégios necessários para explorara-lo. A pontuação CVSS é determinado considerando as métricas na categoria base, que é necessária para determinar a gravidade associada a uma vulnerabilidade. Além disso, a categoria base considera dois subtipos de métricas a explorabilidade e o impacto, sendo o primeiro relacionado com a disponibilidade e facilidade de explorar as vulnerabilidades, enquanto o impacto está associado com a extensão que afeta uma exploração bem-sucedida da vulnerabilidade. A Tabela *14* resume as diversas métricas da categoria base.

Tabela 12-Classificação CVSS3

Escala	**Classificação**
9.0-10.0	Crítico
7.0-9.0	Alto
4.0-6.9	Médio
0-3.9	Baixo e informativo

Esta escala é calculada utilizando diversos critérios agrupados em três categorias que avaliam diferentes características das vulnerabilidades, resumido na Tabela *13*:

Tabela 13-Categorias de avaliação

Categorias	Descrição
Base	Avalia as características essenciais das vulnerabilidades, que não mudam com o tempo.
Temporal	Avalia as características de uma vulnerabilidade que pode alterar-se com o tempo
Ambiental	Avalia como é que o ambiente na qual a vulnerabilidade está inserida pode ou não contribuir para a sua exploração

Tabela 14-Métricas de Explorabilidade e Impacto

Explorabilidade	
Quanto maior a disponibilidade é menos difícil de ser explorado, maior será a gravidade da vulnerabilidade. Esta possui três métricas:	
#VT **Vetor de Ataque**- Possibilidade de o ataque ser feito remotamente ou local	**Local**- O atacante deve ter acesso físico ao sistema.
	Rede - O invasor deve ter acesso, onde normalmente as vulnerabilidades são exploráveis remotamente.
#C.AT **Complexidade de Ataque** Nível de dificuldade na exploração da vulnerabilidade	**Alto**- São exigidas condições especializadas, como por exemplo métodos de engenharia social
	Médio-São exigidos requisitos adicionais para o ataque, como por exemplo limite de pacotes enviados para que o sistema vulnerável seja executado fora do padrão.
	Baixo: Não existem condições especiais para explorar a vulnerabilidade
#E **Escopo** Escopo dos componentes que podem ser exploradas	**Alto-** Tem impacto nos componentes para além do escopo de segurança
	Baixo- Apenas tem impacto em um componente
#UI **Interação de utilizador**	**Alto-**É necessário a interação do utilizador.
	Nenhuma- Não é necessária interação do utilizador
#Pr **Privilégios -** Nível de acesso necessário para explorar a vulnerabilidade.	**Múltiplo-**Para explorar a vulnerabilidade é necessário múltiplas autenticações.
	Único- O atacante precisa autenticar-se uma vez para explorar a vulnerabilidade.

	Nenhuma- Não há necessidade de autenticação.
Impacto	
avalia os impactos provocados em caso de uma vulnerabilidade for explorada.	
#Conf **Confidencialidade** Avalia o nível de exposição de dados confidenciais quando uma vulnerabilidade é explorada.	**Nenhuma-** A confidencialidade do sistema não é comprometida.
	Parcial- Há alguma perda da confidencialidade, onde o atacante pode obter informações restritas, mas não tem o controlo sobre estas informações. Mesmo que as informações forem divulgadas não apresentam um risco direto para o sistema.
	Completo-Existe a perda total da confidencialidade, resultando na divulgação total de todos os recursos do sistema, apresentando assim um impacto direto ao sistema.
#Disp **Disponibilidade** Avalia a acessibilidade aos recursos após a exploração de uma vulnerabilidade	**Nenhuma-** A disponibilidade do sistema não é comprometida
	Parcial- Existe um desempenho reduzido ou interrupções na disponibilidade do sistema, onde mesmo que a vulnerabilidade for explorada constantemente o atacante não consegue provocar uma negação de serviço completo ao todo sistema
	Completo- Existe a perda total da disponibilidade do sistema, tendo como resultado o atacante provocar uma negação de serviços em todo o sistema.
#Int **Integridade-** Avalia o impacto da integridade quando a vulnerabilidade é explorada	**Nenhuma-** Não há impacto na integridade do sistema quando as vulnerabilidades forem exploradas.
	Parcial- A integridade do sistema é comprometida parcialmente, onde o atacante não tem controlo sobre a modificação do sistema pois o escopo é limitado.
	Completo- Existe a perda total da integridade do sistema, onde o atacante pode alterar um ficheiro protegido do componente comprometido.

Para além das métricas CVSS3 que permitem determinar uma pontuação base foram adicionadas métricas que permitem fazer o levantamento de informações relevantes como o estado de resolução, o impacto nos diferentes planos (controlo, dados, aplicação), a data de atualização entre outros que são considerados importantes no processo de decisão.

As métricas do estado e dias de atualização, fornecem informações sobre a correção da vulnerabilidade (na forma de correções de software, novas versões de software), podem ser recolhidas com recurso a informações disponíveis na base de dados NVD "***National Vulnerabilities Databases***". Deve-se realçar que os dias de atualização não significam necessariamente que a vulnerabilidade esteja resolvida, pois pode incluir atualizações sobre as informações disponíveis, por exemplo, para aplicar configurações temporárias a fim de reduzir o impacto. Essas métricas também são relevantes para destacar o suporte da comunidade para corrigir e aprimorar recursos num determinado

controlador SDN. Por exemplo, controladores SDN sem modificações no período de um ano podem indicar o baixo suporte da comunidade para adicionar novas funcionalidades a um controlador SDN (por exemplo, adicionar suporte para P4) ou para corrigir vulnerabilidades identificadas. A probabilidade de ocorrência e o nível de severidade foram introduzidos, considerando a combinação da avaliação das vulnerabilidades com a determinação de riscos em ambiente corporativo [34]. Neste contexto a métrica correspondente aos ativos dos diversos planos mostra quais os componentes afetados nos planos SDN, conforme descrito na secção 3.1. Por exemplo verificar se uma vulnerabilidade afeta apenas aplicações que correm no topo do controlador utilizando a NBI, ou se em simultâneo afeta o plano de controlo e de dados. O número de medidas de mitigação foi adicionado como uma métrica relevante sobretudo em casos onde as vulnerabilidades não estão resolvidas na sua totalidade.

Tabela 15-Métricas adicionais

Métrica	**Valor**	**Descrição**
Estado **#Stat**	{0,1}	1- Resolvido, ou 0- ainda sem resolução
Data de Atual.#UptDay	[0; ∞]	Número de dias desde a identificação e publicação da vulnerabilidade e sua resolução
Ativos no Plano de aplicação **#PlanoAp**	{0,1}	1- Se afeta ativos no plano de aplicação.
Ativos no Plano de controlo **#PlantoCtrl**	{0,1}	1- Se afeta ativos no plano de controlo.
Ativos no Plano de dados **#PlanoDados**	{0,1}	1- Se afeta ativos no plano de dados
Medidas de metição **#MitMea**	{0,1,2}	0-Nenhuma medida, enquanto 2- se por mais de uma medida
Nível de severidade**#NS**	[1; 6]	6-Significa extremo, enquanto 1- significa negligenciável
Probabilidade de ocorrência **#Prob**	[1; 6]	6- Significa máximo, enquanto 1- é negligenciável

Com as métricas identificadas é necessário fazer o ranking das vulnerabilidades tendo como base o algoritmo MeTHODICAL [35] que é uma abordagem de decisão multicritério utilizado em cenários de decisão QoS, seleção de caminhos e migração de recursos. O primeiro passo é classificar as vulnerabilidades em duas categorias: custos e benefícios. O custo leva em consideração as métricas cujo valor deve ser minimizado, por exemplo os valores da pontuação CVSS deve ser o mais próximo de zero pois apresenta menor probabilidade de exploração e redução do impacto no sistema e serviços, por outro lado as métricas tidas como benefícios os valores devem ser maximizados, por exemplo o número de medidas de mitigação deve ser maior para reduzir o impacto da exploração da vulnerabilidade.

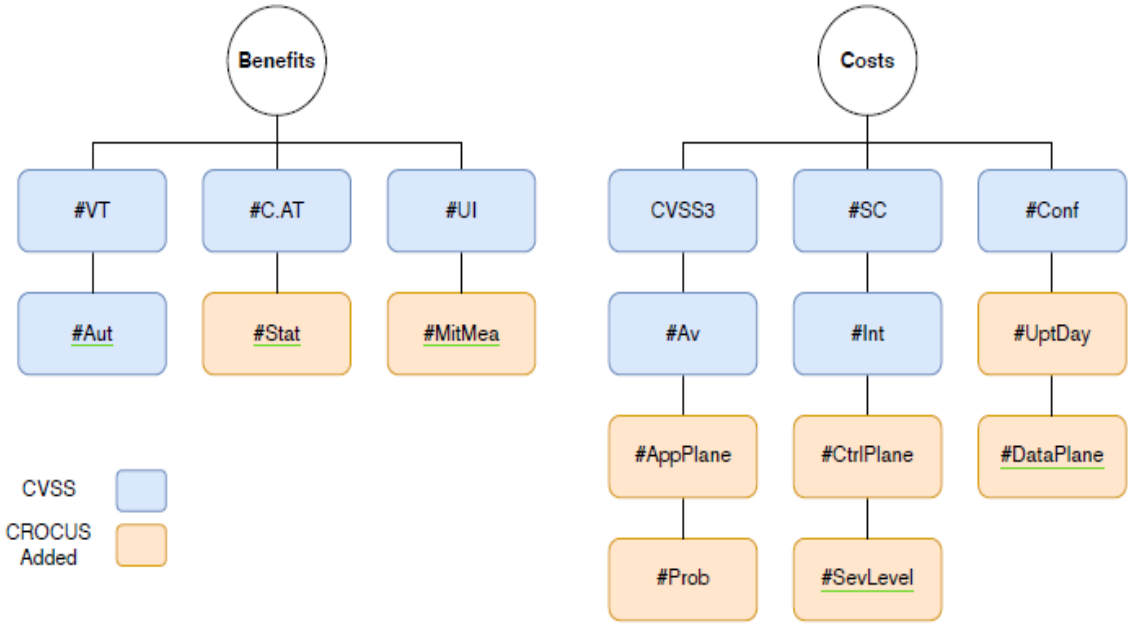


Figura 6- Métricas CROCUS na etapa de classificação

A Figura *6* representa as métricas já fornecidas pelos padrões CVSS e também destaca as métricas propostas pelo CROCUS. Estas foram propostas porque as métricas CVSS não abordavam questões de segurança importantes como o número de mitigação disponibilizados para uma vulnerabilidade ou o número de ativos afetados nos diferentes planos de uma rede SDN, desta forma as métricas do CROCUS complementa e aborda os aspetos de segurança para uma rede SDN. As diversas métricas representadas são associadas a um determinado peso que vai de acordo com as preferências do utilizador e o cenário ou serviço que este é utilizado. Por exemplo um utilizador pode atribuir uma maior importância para pontuação CVSS do que a probabilidade de ocorrência de uma vulnerabilidade. Além disso as próprias categorias podem ser-lhe atribuídas um peso onde a categorias de benefícios pode ser considerado mais relevante do que a de custos. Por fim temos uma classificação com uma pontuação ordenada em que os valores mais baixos são os mais relevantes, pois estão próximos das soluções ideais que o algoritmo MeTHODICAL leva internamente em consideração.

3.3.3 Passo 3-Classificação final das vulnerabilidades reportadas

As classificações finais das vulnerabilidades reportadas para os controladores têm por base as tabelas 14 e ,15 apresentadas na secção 3.3.2. Como já mencionado na secção 3.3.1 os controladores para o estudo foram restringidos ao controlador OpenDayLight e ONOS.É importante realçar que a linha temporal da análise das vulnerabilidades para o ONOS e OpenDayLight foi no período de 2014 a 2020. A Figura *7* representa as vulnerabilidades do OpenDayLight apresentando o total de doze (12) vulnerabilidade de risco alto a critico, enquanto a *Figura 8* representa o ONOS com o total de vinte e nove

(29) vulnerabilidades sendo treze (13) de risco critico. As tabelas das vulnerabilidades para ambos os controladores estão no Anexo C.

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

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

Figura 7-Vulnerabilidades OpenDayLight

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

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

Figura 8-Vulnerabilidade ONOS

A média de dias para atualização do estado de uma vulnerabilidade crítica no OpenDaylight é de 352 dias o que é maior quando comparado com o tempo do ONOS que é de aproximadamente 156 dias. A Figura *7* demonstra que OpenDayLight tem menos vulnerabilidades do que o ONOS, mas por outro lado a sua percentagem do rácio solvabilidade(rCriODL) é maior quando comparado com os 30.65% do ONOS. De realçar que a média de dias para resolução da vulnerabilidade críticas do OpenDaylight é de 444 dias e nenhuma das vulnerabilidades analisadas durante o período de tempo do estudo está no estado aberto ou sem informação. O menor rácio solvabilidade(rCriONOS) do ONOS demonstra que este tem um desenvolvimento ativo e com novas funcionalidades a serem incorporadas visto que tem vulnerabilidades mais recentes (depois de 2019) em comparação ao OpenDayLight. Outro aspeto importante são as vulnerabilidades reportados como médios no controlador ONOS onde estes resultam de eventos e interações entre componentes do controlador. O principal problema é que estes tipos de vulnerabilidades são mais complicados de detetar, pois exigem um conhecimento profundo do controlador e de seus componentes internos. Mas, por outro lado, este é

do interesse para os atacantes, pois sua exploração é difícil, mas também mais difícil de detetar.

3.4 Finalização do Capítulo

Este capítulo abordou uma das partes fundamentais do trabalho que é a segurança nas redes SDN e uma metodologia objetiva para escolha do controlador. Iniciou-se por identificar os elementos que acrescem um valor para a rede seguido das ameaças genéricas e especificas que cada plano pode sofrer.
Foi introduzido a metodologia de avaliação das vulnerabilidades que permite identificar as que tem maior probabilidade de acontecer e que apresentam um maior impacto, de forma a perceber como tornar os controladores mais seguros. A metodologia de avaliação é dividida em 3 etapas: 1- identificar os controladores, 2-Recolher as vulnerabilidades, 3-classificação final das vulnerabilidades reportadas
Na primeira fase foi introduzida o mecanismo denominado de "CROCUS" que permite uma avaliação objetiva e suporte de segurança dos controladores SDN, onde este faz uso de uma variedade de métricas existentes e outras que foram introduzidas desta forma através de uma equação utilizando estas métricas conseguimos avaliar de forma objetiva o tempo de reação em que uma entidade/comunidade resolve as vulnerabilidades. Com este capítulo foi possível concluir que o controlador ONOS apesar de apresentar mais vulnerabilidades quando comparado com o OpenDaylight, tem uma melhor percentagem de resolução das vulnerabilidades e as vulnerabilidades do ONOS resultam de eventos e interações entre componentes do controlador.

4 Desenvolvimento e Implementação

Neste capítulo são apresentados as decisões de desenvolvimento e o processo de implementação percorrido. Na secção 4.1 é feito um breve resumo da funcionalidade ONOS SecurityMode onde mais detalhes podem ser encontrados no Anexo B, como já foi mencionado na secção 1.4 esta funcionalidade foi descontinuada pelo ONOS, mas inicialmente foi uma referência importante para a proposta de implementação de segurança, onde na prática o trabalho passaria por utilizar uma aplicação por exemplo a DHCP do ONOS, para avaliar o comportamento dos mecanismos de controlo de acesso (ONOS SecurityMode), ou seja, configurar toda a infraestrutura da rede através do DHCP e perceber como é que os controladores distribuídos podem controlar as permissões que esta aplicação precisa para efetuar alterações nos diferentes componentes das redes dos controladas pelos controladores. Na secção 4.2 é apresentado o cenário proposto para o processo de implementação e documentação de segurança das diversas interfaces do controlador ONOS.

4.1 ONOS SecurityMode

O controlador ONOS fornece APIs para a interface Northbound que permite a comunicação e desenvolvimento de aplicações no plano de aplicação. Estas APIs são fáceis de utilizar, mas por outro lado devido ao nível de acesso que lhe é concedido de forma a ter o máximo de desempenho possível, permite que as aplicações do ONOS realizem qualquer ação na rede.
As aplicações do ONOS devido ao nível de acesso que possuem, podem ser considerados um ponto vulnerável na rede, que se forem utilizadas com fins maliciosos comprometem a rede, pois algumas dessas aplicações possuem privilégios que podem alterar o comportamento da rede. O ONOS disponibiliza o recurso SecurityMode para redes com determinados requisitos, desta forma os administradores de rede podem configurar o controlador de forma a restringir a capacidade de acesso das aplicações.
Na prática o SecurityMode adiciona dois recursos principais ao controlador ONOS:

- **Autenticação da aplicação.**
- **Controlo de acesso baseado em regras e permissões.**

O SecurityMode exige que os administradores de rede revejam e aceitem as políticas de segurança de cada aplicação antes desta ser ativada. Quando o controlo de acesso é baseado em permissões uma aplicação não pode utilizar os recursos de forma indiscriminada, ou seja, estes têm de ser explicitamente permitidos. Estas permissões estão relacionadas com os serviços e APIs que uma aplicação poderá utilizar para interagir com o controlador ONOS, por exemplo uma aplicação com permissão do tipo APP READ pode aceder em modo de leitura às informações das aplicações instalados.
Estas permissões são definidas no ficheiro *app.xml* das aplicações, desta forma sempre que a aplicação for instalada estas permissões são revistas e sujeita a aprovação, caso

contrário se este processo não for feito, a ativação da aplicação é tida como uma violação de políticas de segurança.

4.2 Implementação

Nesta secção é apresentado o processo de implementação e documentação de segurança das diversas interfaces do controlador ONOS (NBI, SBI, EBI/EWI). Inicialmente a proposta de implementação de segurança tinha por base o uso do ONOS SecurityMode, mas esta funcionalidade por ter sido descontinuada levou a uma alteração da abordagem. Na abordagem inicial os mecanismos do SecurityMode permitiriam avaliar as permissões que uma aplicação utiliza durante a sua execução. Nesta nova abordagem o objetivo é utilizar os mecanismos existentes no próprio controlador para implementar mecanismos de segurança para todas as interfaces de forma a proteger as comunicações (confidencialidade e integridade). É importante realçar que as documentações referentes as implementações do TLS nas diversas interfaces estão dispersas e as que estão disponíveis não fornecem informações de configurações claras e atualizadas das versões dos controladores existentes. Como já mencionado no capítulo 3 as interfaces podem ser pontos vulneráveis do controlador. A Figura *9* representa o cenário de implementação do protocolo TLS em todos as interfaces do controlador:

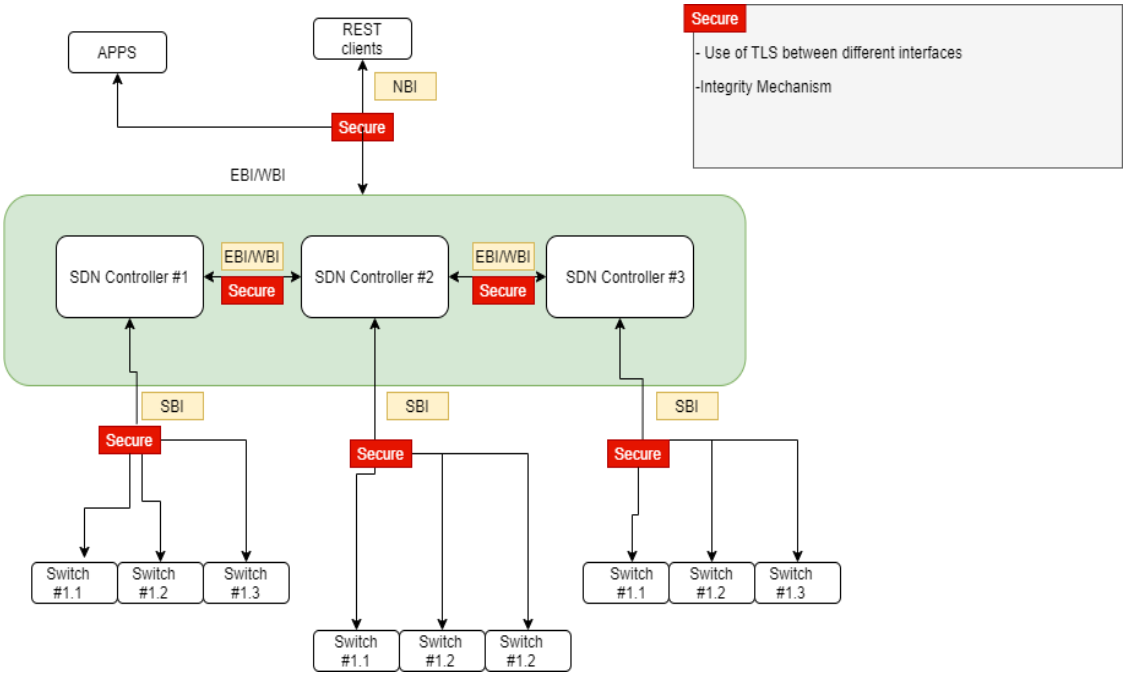


Figura 9-Cenário de implementação de segurança nas interfaces (NBI, SBI, EBI e WBI).

4.2.1 Cluster

O primeiro passo na implementação do cenário foi criar um ambiente cluster com três controladores ONOS. Os mecanismos de cluster acompanharam a evolução do ONOS onde as versões anteriores ao 1.14 utilizavam o serviço Apache Zookeeper para fornecer informações de configuração, sincronização e serviços para clusters em sistemas distribuídos, já nas versões posteriores ao 1.14 o ONOS utiliza a framework denominada de Atomix para coordenação do ambiente em cluster. Esta partilha de informações é feita utilizando o mecanismo de distribuição mais apropriado, desta forma os programadores podem escolher entre os dois modelos (consistente e eventualmente consistente) apresentados na secção 2.4.1.

Configuração do Atomix

O processo de criação de um cluster requer a configuração de dois ficheiros diferentes em cada um dos nodos do cluster. O primeiro é o ficheiro de configuração Atomix em formato JSON que requer três componentes essenciais:

Cluster
- Especifica como os nodos do Atomix se comunicam e descobrem uns aos outros no cluster.

Grupos de Partição
- Faz a configuração dos grupos de participação para armazenamento e replicação.

Grupo de Gestão
- Configura os grupos de partições para gerir primitivos.

Os grupos de partição foram introduzidos a partir da versão 3.0 do Atomix. Este pode ser definido como um conjunto de partes que implementam um protocolo específico de sistemas distribuídos (por exemplo, protocolo Raft). As primitivas distribuídas são armazenadas num grupo de partição específico e todas as primitivas podem operar em qualquer grupo de partição, independentemente do protocolo que seja implementa. Essa arquitetura genérica significa que o protocolo subjacente às primitivas distribuídas do Atomix é configurável. No ONOS é esperado que o cluster Atomix seja configurado com um grupo de partição Raft para primitivos distribuídos (ONOS wiki, 2018) [36].
A Tabela *16* representa o ficheiro Atomix criado para cada nodo do cluster.

Tabela 16-Atomix.conf

Atomix.conf

```
1: {
2:   "cluster": {
3:     "node": {
4:       "id": "atomix-1",
5:     "address": "172.20.0.10:5679"
6:     },
7:     "clusterId": "onos",
8:     "discovery": {
9:       "nodes": [
10:          {
11:            "id": "atomix-1",
12:             "address": "172.20.0.10:5679"
13:          },
15:          {
16:            "id": "atomix-2",
17:           "address": "172.20.0.11:5679"
18:          },
19:
20:          {
21:            "id": "atomix-3",
22:            "address": "172.20.0.12:5679"
23:          }
24:
25:        ],
26:        "type": "bootstrap"
27:     }
28:  },
29:   "partitionGroups": {
30:     "raft": {
31:       "partitionSize": 3,
32:       "storage": {
33:         "level": "mapped"
34:       },
35:       "type": "raft",
36:      "members": [
37:         "atomix-1",
38:         "atomix-2",
39:         "atomix-3"
40:       ],
41:       "partitions": 3
42:    }
43:   },
44:  "managementGroup": {
45:     "partitionSize": 2,
46:    "storage": {
47:                 "level": "mapped"
48:     },
49:     "type": "raft",
50:     "members": [
51:       "atomix-1",
52:       "atomix-2",
53:         "atomix-3"
54:     ],
55:    "partitions": 1
```

Na configuração do cluster os nodos fornecidos para o protocolo de descoberta são o conjunto de todos os nodos do Atomix (linha 8 a 22). De realçar que nodos do ONOS não precisam ser mencionados, pois estes ligam-se ao Atomix numa arquitetura do tipo cliente-servidor. O parâmetro de endereço para cada nodo é representado pelo tuplo host:-port. Como já mencionado na secção 2.6 existem protocolos de descoberta adicionais, no entanto o protocolo de bootstrap é recomendado para nodos de consenso, uma vez que identidades de nodos fixos é um requisito do protocolo Raft.

Na configuração dos grupos de partição (linha 29 a 41) é definido um conjunto de grupos nomeados para armazenar primitivos distribuídos.

O grupo de gestão (linha 44 a 55) define o protocolo e a configuração a serem utilizados para administrar o cluster Atomix. Como o ONOS depende muito do Atomix para obter consenso, esse deve ser um grupo de partição Raft. O grupo de gestão pode ser configurado com apenas uma única partição, pois a carga no grupo é baixa.

Configuração do ONOS

Com o Atomix configurado em cada nodo o passo seguinte na criação do cluster é a configuração do ONOS. As versões mais recentes, do ONOS e do Atomix obrigou a adaptação do ficheiro *cluster.json* de forma a receber as alterações nos padrões de comunicação dentro dos clusters ONOS e Atomix. Nas versões anteriores, os nodos do ONOS formavam um cluster comunicando-se entre eles para eleger o líder utilizando o protocolo Raft. Por ser um protocolo de consenso, o Raft exigia informações sobre os membros do cluster para preservar a consistência ao formar o cluster por meio da contagem de votos. Em versões recentes, um nodo do ONOS pode descobrir seus pares utilizando mecanismos de descoberta dinâmicos. A descoberta dos nodos é feita através do cluster Atomix representado na Figura *10* abaixo:

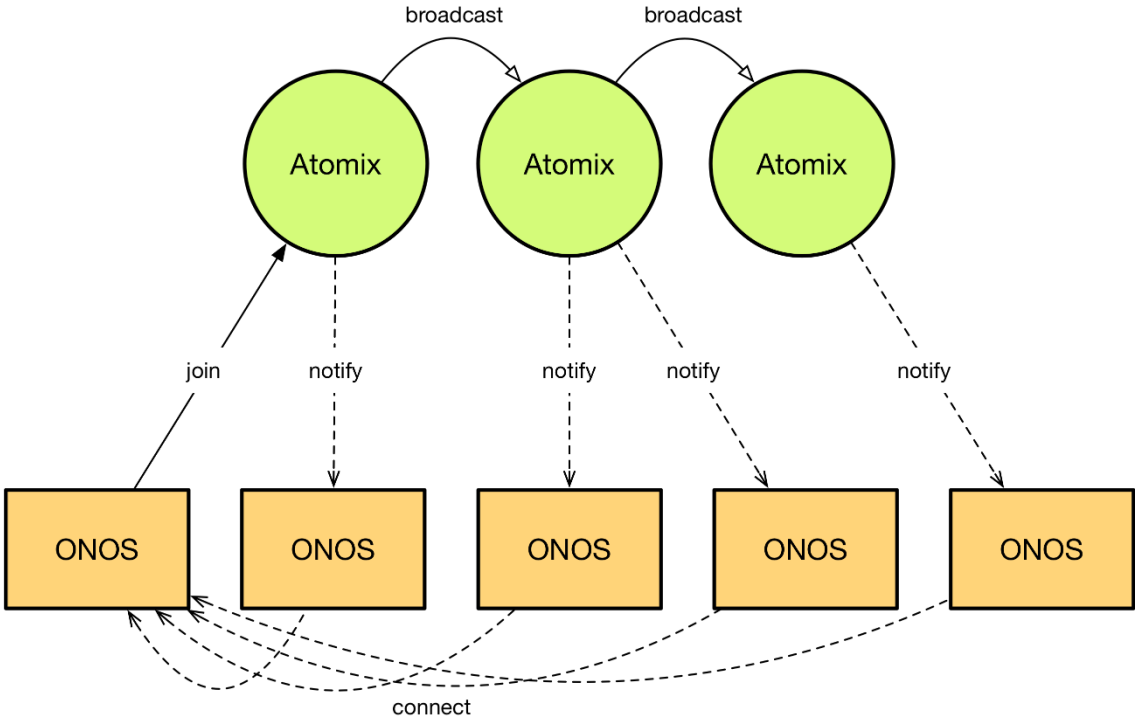


Figura 10-Descoberta ONOS/Atomix *[37]*

Quando um nodo ONOS é inicializado, este conecta-se ao cluster Atomix externo para armazenamento de dados e coordenação. Ao conectar-se, o nodo ONOS notifica o Atomix sobre sua existência e localização, este por sua vez transmite as informações do novo nodo para todos os outros nodos ONOS conectados, e os nodos ONOS que estão conectados consequentemente conectam-se diretamente de volta ao novo nodo ONOS para uma comunicação ponto a ponto. Isso permite que os nodos do ONOS se descubram dinamicamente. A Tabela *17* representa o ficheiro *cluster.json* criado para cada nodo ONOS onde nestes são listados os nodos do Atomix.

Tabela 17-Cluster.json

Cluster.json

1:{
2: "node": {
3: "ip": "172.20.0.20",
4: "id": "172.20.0.20",
5: "port": 9876
6: },
7: "storage": [
8: {
9: "ip": "172.20.0.10",
10: "id": "atomix-1",
11: "port": 5679
12: },
13: {
14: "ip": "172.20.0.11",
15: "id": "atomix-2",

```
16:          "port": 5679
17:       },
18:      {
19:          "ip": "172.20.0.12",
20:          "id": "atomix-3",
21         "port": 5679
22:       }
23:   ],
24:    "name": "onos"
25:       }
```

4.2.2 Configuração TLS

Com o cluster criado o próximo passo passou por identificar quais os mecanismos que já existem no próprio controlador que permitem proteger as suas interfaces de comunicação. A implementação do TLS foi feita utilizando o algoritmo RSA, de forma a ter uma configuração semelhante as documentações disponibilizadas pelo ONOS, mas é importante realçar que este algoritmo não é aconselhado nas versões recentes do TLS(secção 2.7.1).A base de segurança do RSA é o nível de dificuldade em determinar fatores de números muitos grandes, onde quanto maior for o tamanho da chave maior o nível de segurança, desta forma o tamanho recomendado e implementado foi uma chave de tamanho 2048 bits que também é a base fornecida por uma autoridade de certificação. Para cada nodo do cluster será apresentado o processo de configuração do protocolo TLS.

Gerar certificado e chave

O comando descrito na Tabela *18* vai gerar uma nova chave e adicioná-la ao armazenamento de chaves. Os valores de 'alias', 'storepass', 'validade' e 'tamanho da chave' podem ser modificados de acordo com os requisitos exigidos. O valor 'keyalg' é recomendado utilizar. Alguns browsers aceitam o certificado gerado, mas apresentam mensagem de aviso pois estes esperam que alguns parâmetros estejam presentes pois por serem certificados autoassinados não é possível determinar a entidade de certificação, desta forma para indicar que este é confiável foi adicionado o parâmetro SAN (*Subject Alternative Name*) onde é passado os valores *hostname* e seu IP correspondente. É necessário aplicar este comando em todos os nodos do cluster alterando os valores dos parâmetros para os seus correspondentes.

Tabela 18-Gerar certificados

Generate a Key/Certificate
keytool -genkey -alias teastoressl -keyalg RSA -keysize 2048 -validity 360 -keystore "/rootapache-karaf-4.2.8/etc/keystore/rc.keystore" -ext SAN=dns: YourONOSInstanceName, ip:HostIP What is your first and last name? [CV]: **InstanceName** What is the name of your organizational unit? [CV]: **UnitName** What is the name of your organization? [CV]: **OrganizationName** What is the name of your City or Locality? [CV]: **CityName** What is the name of your State or Province? [CV]: **StateName** What is the two-letter country code for this unit? [CV]: **CountryName** Is CN=**InstanceName**, OU=**dei**, O=**uc**, L=**coimbra**, ST=**coimbra**, C=**pt** correct? [no]: yes, Enter key password for (RETURN if same as keystore password):

Distribuição dos certificados

Com o certificado gerado em cada nodo do ONOS é necessário distribuí-lo entre outros nodos do ONOS. Para isso, a chave deve ser convertida num arquivo, em formato. p12 e, em seguida, num ficheiro em formato .pem, esta conversão é necessária pois quando o ficheiro está em formato .p12 este contém as chaves geradas e de forma a guardar o certificado é necessário que este esteja em formato .pem , pode ser executado com os seguintes comandos:

Tabela 19-Comando para distribuir certificado

Obter certificado
1 **:** keytool -importkeystore -srckeystore /root/onos/apache-karaf-4.2.8/etc/keystore/rc.keystore -destkeystore onos2.p12 -srcstoretype jks -deststoretype pkcs12
2:openssl pkcs12 -in onos2.p12 -out onos2.pem

O certificado pode ser extraído do ficheiro PEM utilizando o comando da Tabela *20* o ficheiro de saída deste comando deve ser ajustado para garantir que seja exclusivo para cada nodo do ONOS, em concreto deve-se indicar o nodo na qual pertence o certificado.

Tabela 20-Extrair ficheiro PEM

Extrair ficheiro PEM
awk 'split_after == 1 {n++; split_after=0} /-----END ENCRYPTED PRIVATE KEY-----/ {split_after=1} {print > "onoscert" n ". pem"}' < onos1.pem; mv cacert1.pem onos1cert.pemd

Copiar e Importar os certificados para cada nodo do ONOS

Para cada nodo do ONOS presente no cluster este precisará partilhar o seu certificado com os restantes nodos. Os certificados podem ser copiados utilizando os comandos da Tabela *21* e deverá ser repetido para cada certificado e cada instância do ONOS. Com os certificados copiados para cada instância este precisará de os ter no respetivo armazenamento de chaves(keychain).

Tabela 21-Copiar e importar certificados

Copiar certificados do container ONOS para o host
docker cp "container ID" /root/onos/apache-karaf-4.2.8/etc/onos1cert.pemd /home/node
Partilhar certificado para restantes nodos ONOS
scp /home/node-3/onos3.pem node-1@192.168.5.145:/home/node-1/ scp /home/node-3/onos3.pem node-1@192.168.5.146:/home/node-2/
Enviar os certificados recebidos para o keystore do ONOS
docker cp /home/node-2/onos1.pem onos3:/root/onos/apache-karaf-4.2.8/etc/keystore docker cp /home/node-2/onos3.pem onos3:/root/onos/apache-karaf-4.2.8/etc/keystore
Converter os certificados em formato. pem para .crt para ser adicionado ao Keystore
openssl x509 -outform der -in onos1.pem -out onos1cert.crt openssl x509 -outform der -in onos3.pem -out onos3cert.crt
Importar os certificados convertidos para a keystore
keytool -import -alias onos3 -file onos2cert.crt -keystore /root/onos/apache-karaf-4.2.8/etc/keystore/rc.keystore
Verificar os certificados importados na keystore de cada nodo ONOS
keytool -v -list -keystore /root/onos/apache-karaf-4.2.8/etc/keystore/rc.keystore

4.2.2.1 TLS na Interface Northbound

O acesso à interface Northbound-NBI é feito através de Web UI e API REST fornecido pelo protocolo HTTP. Apesar de ser exigida a autenticação para aceder ao Web UI a transferência entre o cliente e o servidor não é segura, o que pode permitir a um atacante alterar e observar dados trocados entre o cliente e o servidor, por isso é necessário configurar e ativar o protoloco HTTPS de forma a garantir a comunicação segura com a interface Northbound.

Alterar Ficheiro de configurações ONOS

Com a chave criada (Tabela *18*)é necessário alterar o ficheiro "**org.ops4j.pax.web.cfg**" do ONOS de acordo com a Tabela *22*, onde os parâmetros passwords e keystore deve ser definida de acordo com o que foi configurado na etapa de gerar certificado e chave.

Tabela 22-Alterar ficheiro ONOS

org.ops4j.pax.web.cfg
1: org.osgi.service.http.port=**8181** 2: org.osgi.service.http.port.secure=**8443** 3: 4: org.osgi.service.http.enabled=**true** 5: org.osgi.service.http.secure.enabled=**true** 6: 7: org.ops4j.pax.web.ssl.keystore=etc/keystore/rc.keystore 8: org.ops4j.pax.web.ssl.password=**onos22** 9: org.ops4j.pax.web.ssl.keypassword=**onos22** 10: 11: 12: org.ops4j.pax.web.session.timeout=1440 13: 14 : org.ops4j.pax.web.session.url=none 15: org.ops4j.pax.web.config.file=./etc/jetty.xml

Por omissão os browsers não aceitam ou apresentam uma mensagem de aviso quando é utilizado certificados autoassinados, dadas as questões de segurança associadas a este processo, desta forma é necessário adicionar uma exceção de segurança. Isto pode ser feito utilizando uma autoridade de certificação para assinar o certificado uma outra solução e a que foi utilizado no trabalho foi adicionar o certificado na lista de certificados fidedignos do sistema (exemplo: Keychain no MacOS).
Por fim com os certificados gerados, o ONOS web UI pode ser acedido através:
https://localhost:8443/onos/ui **.**

4.2.2.2 TLS na Interface East/Westbound

A comunicação entre os controladores é feita na interface EBI e WBI onde por padrão não fornece uma comunicação segura, o que pode permitir a observação e alteração do tráfego entre os controladores. O TLS deve ser habilitado entre controladores para garantir que a comunicação dentro do cluster ONOS seja segura. O processo de configuração de TLS pode ser realizado tendo por base a informação no website do ONOS [38], de realçar que alguns parâmetros e outras etapas(secção 4.2.2) foram adicionados devido às alterações nas versões mais recentes do ONOS e na formação do cluster através do Atomix, em concreto foi adicionado o parâmetro "*cluster.messaging.tls*" no ficheiro de configuração dos nodos Atomix, alterações estas que o website do ONOS não apresenta onde pode ser visto que teve sua última data de atualização no ano de 2018.Antes de prosseguir é necessário efetuar as etapas da distribuição e importação dos certificados para cada nodo do Atomix apresentadas na secção 4.2.2.

Alterar Ficheiro Atomix.conf

Nas versões mais recentes do ONOS a formação do cluster alterou-se significativamente onde temos a separação do Atomix e do ONOS, o que antes era embutido dentro do ONOS. Desta forma é necessário alterar em cada nodo Atomix do cluster o ficheiro Atomix.conf onde é adicionado um novo parâmetro "*cluster.messaging.tls*"**.** É necessário adicionar o certificado gerado anteriormente no *keyStore* do nodo Atomix (linha 4 e 6), isto pode ser feito replicando as etapas representadas na Tabela *21*.

Tabela 23-Cluster.messaging.tls

Atomix.conf
1:cluster.messaging.tls: 2: { 3: enabled: true 4: keyStore: /opt/atomix/onos/apache-karaf-4.2.8/etc/keystore/rc.keystore 5: keyStorePassword: onos22 6: trustStore: /opt/atomix/onos/apache-karaf-4.2.8/etc/keystore/rc.keystore 7: trustStorePassword: onos22 8: }

Alterar ficheiro no ONOS

Para cada instância do ONOS é necessário alterar o script do ficheiro "*onos-service*" localizado no diretório *ONOS_ROOT / bin /,* onde este contém uma linha para ativar o protocolo TLS para Netty, que é a estrutura de comunicação utilizada entre os controladores. É necessário remover o comentário da linha e certificar de que os valores dos parâmetros '*keystore*' e o '*truststore*' apontam para o keystore usado na geração dos certificados, por último deve-se alterar o parâmetro *enableNettyTLS* para *io.atomix.enableNettyTLS.*

Tabela 24-Ficheiro onos-service

Alterar ficheiro onos-service-EBI/WBI
Export JAVA_OPTS=" ${JAVA_OPTS: -- io.atomix.enableNettyTLS =true -Djavax.net.ssl.keyStore=/root/onos/apache-karaf-4.2.8/etc/rc.keystore - Djavax.net.ssl.keyStorePassword=onos22 -Djavax.net.ssl.trustStore=/root/onos/apache-karaf-4.2.8/etc/rc.keystore - Djavax.net.ssl.trustStorePassword=onos22}"

4.2.2.3 TLS na Interface SouthBound

A interface Southbound não faz a encriptação dos dados ou autenticação dos dispositivos conectados na rede. Com o TLS ativado a comunicação entre o ONOS e os dispositivos no plano de dados (exemplo: routers, switchers) são encriptados garantido a privacidade nas comunicações, e alteração do tráfego de dados.

Converter Ficheiros

Com o certificado gerado anteriormente (Tabela *18*), este será usado pelo OvS (OpenvSwitch), que suporta ficheiros em formato .pem, desta forma deverá ser convertida através de um processo de duas etapas. No primeiro, o keystore deve ser convertido para o formato .p12:

Tabela 25-Obter certificado em formato .p12

Obter certificado em formato .p12
1: keytool -importkeystore -srckeystore apache-karaf-3.0.8/etc/keystore/rc.keystore --destkeystore onos1.p12 -srcstoretype jks -deststoretype pkcs12: Enter destination keystore password: Re-enter new password: Enter source keystore password: Entry for alias onos successfully imported. Import command completed: 1 entries successfully imported, 0 entries failed or cancelled

Com o ficheiro em formato .p12 este já pode ser convertido para um ficheiro em formato. pem

.

Tabela 26-Exportar certificados p12 para pem

Exportar ficheiro do formato p12 para pem
1:openssl pkcs12 -in onos1.p12 -out onos1.pem Enter Import Password: MAC verified OK Enter PEM pass phrase: Enter PEM pass phrase

O certificado contido no ficheiro ‘onos1.pem’ deverá ser copiado para um ficheiro separado, que posteriormente será partilhado entre os hosts.

Tabela 27-Obter certificado para ser partilhado

Exportar ficheiro do formato p12 para pem
1: awk 'split_after == 1 {n++;split_after=0} /-----END ENCRYPTED PRIVATE KEY-----/ {split_after=1} {print > "cacert" n ".pem"}' < onos1.pem; mv cacert1.pem onos1-pub.pem

Copiar o certificado para diretoria do OpenvSwitch

O ficheiro *nos1-pub.pem* (Tabela 27) deve ser copiado para diretório apropriado do OpenvSwitch (OvS) para que este possa aceitar o certificado nos nodos do ONOS.

Tabela 28-Copiar certificado para OVS

Copiar certificado para OVS
1: scp ./ onos1-pub.pem admin@ovs :/var/lib/openvswitch/pki/controllerca

Gerar certificado SSL para OpenvSwitch

No host responsável pelos componentes OvS vai ser gerado um certificado que será distribuído e utilizado pelo ONOS, e por último é configurado o OvS para utilizar as chaves geradas na Tabela 29.O host host responsável corresponde ao nodo onde se emula a rede com recurso ao mininet.

Tabela 29-Gerar certificados OvS

Gerar certificado para OvS
1: ovs-pki req+sign sc switch

A Tabela 30 apresenta a configuração que é necessário no host virtual em termos de certificados, chave privada e certificado do controlador ONOS para encriptar as comunicações. Estas configurações são utilizadas por todos os elementos OvS.

Tabela 30-Habilitar o OvS para utilizar as novas chaves

OVS a utilizar as novas chaves
1: ovs-vsctl set-ssl /etc/openvswitch/sc-privkey.pem \ /etc/openvswitch/sc-cert.pem /var/lib/openvswitch/pki/controllerca/ onos1-pub.pem

Copiar chave para o ONOS

É necessário copiar o certificado OvS para o diretório do ONOS para que este seja adicionado à sua keystore. Os comandos da *Tabela 31* representam a cópia da chave *sc-cert.pem (geradas na* Tabela 29) e sua importação para o keystore do host ONOS.

Tabela 31-Copiar e importar chave OvS para ONOS

Copiar chave OVS para ONOS
1: scp ./sc-cert.pem node-1@192.168.5.145:/ home/node-1/
Importar chave OvS para keystore do ONOS
keytool -importcert -file sc-cert.pem -keystore /root/onos/apache-karaf-4.2.8/etc/keystore/rc.keystore

Alterar ficheiro no ONOS

Com o certificado adicionado na keystore do ONOS (*Tabela 31*) é necessário adicionar novos parâmetros no ficheiro "onos-service" no diretório *ONOS_ROOT / bin /* de forma a ativar o TLS entre o ONOS e os switches OvS.

Alterar ficheiro onos-service
export JAVA_OPTS="${JAVA_OPTS:--DenableOFTLS=true -Djavax.net.ssl.keyStore=/root/onos/apache-karaf-4.2.8/etc/rc.keystore -Djavax.net.ssl.keyStorePassword=onos22 -Djavax.net.ssl.trustStore=/root/onos/apache-karaf-4.2.8/etc/rc.keystore -Djavax.net.ssl.trustStorePassword=onos22}"

4.3 Finalização do Capítulo

Este capítulo abordou brevemente as funcionalidades do ONOS SecurityMode que inicialmente foi a abordagem escolhida para implementação de segurança, antes deste ser alterado para uma nova abordagem porque foi verificado que o controlador já não oferecia suporte a esta funcionalidade. Outro ponto abordado neste capítulo e que foi uma das contribuições do trabalho, é documentação do processo de implementação de segurança com recurso ao TLS nas diversas interfaces (NBI, SBI, EBI/WBI) do controlador ONOS, onde foi apresentado o cenário que representa o cluster com todas as interfaces seguras seguido do processo de configuração de segurança para cada uma das interfaces. Esta foi uma tarefa desafiadora, pois as documentações de implementações com recurso ao TLS existentes são dispersas e não estão atualizados para os deployment actuais. Por exemplo, não têm em consideração a introdução do Atomix no processo de criação de clusters. Desta forma foi produzido e disponibilizado um guia atualizado com o código e configurações em modo Open-Source (repositório git) *[39]* para que possa ser utilizado quando é necessário deixar as interfaces do controlador ONOS mais seguras com recurso a configuração do protocolo TLS.

5 Validação e Análise

Neste capítulo, é apresentado e analisado os resultados obtidos no capítulo do Desenvolvimento e Implementação. Em suma, os objetivos são de validar e analisar se os mecanismos foram implementados (Seção 4.2), e verificar se existe alterações de desempenho, e outras que possam ocorrer no controlador. Em termos de segurança, o objetivo é ter as interfaces de comunicação encriptadas através da implementação do protoloco TLS. Na secção 5.1 é apresentado estratégia de validação, detalhando o os testes realizados. A figura representa o cenário de validação utilizado.

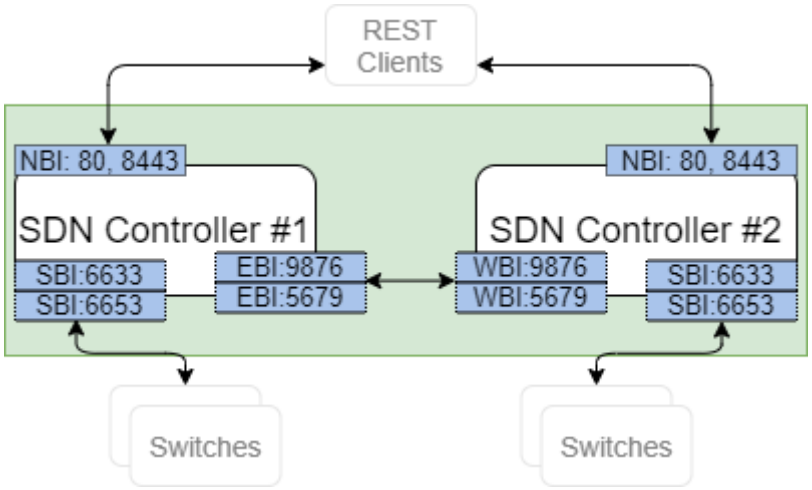


Figura 11-Cenario de avaliação TLS

5.1 Estratégias de Validação

A estratégia de validação adotada foca-se na implementação do protocolo TLS nas interfaces de comunicação NBI e o seu impacto nos controladores e na rede. Vai ser analisada em duas vertentes, na primeira é avaliado como é que a implementação de diferentes algoritmos de encriptação na NBI pode impactar o desempenho, e na segunda será avaliado o impacto quando temos um controlador com TLS implementado e outro controlador sem o TLS.

A medição do impacto será feita com recurso à captura dos pacotes e respetiva análise, desta forma podemos acompanhar todo o processo de estabelecimento da conexão segura do controlador e avaliar o impacto que este tem no sistema, na prática esta análise será realizada três vezes para cada um dos algoritmos de encriptação de forma a termos dados suficientes que possam validar a análise. A avaliação do impacto com recurso a análise dos pacotes é feita através das seguintes métricas:

Tabela 32-Métricas de Avaliação

Métricas	**Descrição**
Estabelecimento da Sessão	Tempo do primeiro pedido até ao envio da resposta. Do início até o primeiro pacote de dados de aplicativo.
Estabelecimento de conexão	Num segundo tempo de solicitação do primeiro pacote até ao envio da resposta. Do *client hello* até os primeiros dados do aplicativo
Tempo de carregamento	Avalia o tempo de carregamento total da página de login no browser. Na primeira, segunda e terceira solicitação.

Sendo a NBI destinada a fazer a interface com o utilizador, foi também medido o tempo de carregamento da página através das métricas disponibilizadas em modo programador do browser Chrome e Edge.O utilitário openSSL foi utilizado para aferir o tempo de ligação e performance (número de pedidos por segundo).

5.2 Avaliação do impacto dos diferentes algoritmos na NBI

Definido as métricas e o processo de avaliação, é necessário implementar os algoritmos de encriptação RSA e ECDHE na NBI, processo apresentado anteriormente na secção 4.2.2.1 com a diferença que serão geradas chaves de diferentes tamanhos associados ao certificado.

Tabela 33- Encriptação e algoritmos de assinatura

Algo Enc.	**Tamanho Chave**	**Algo. Ass**
RSA	2048	SHA256withRSA
ECDHE	512	SHA512withECDSA

Tabela 34-Gerar certificados para validação

Gerar certificado RSA
1: keytool -genkeypair -alias ONOS_RSA -keyalg RSA -keysize 2048 -sigalg SHA256withRSA -validity 360 -keystore "/root/onos/apache-karaf-4.2.8/etc/keystore_RSA" -ext SAN=dns:onos1,ip:192.168.5.145
Gerar certificado ECDHE
2: keytool -genkeypair -alias ONOS_EC -keyalg EC -sigalg SHA512withECDSA -validity 360 -keystore "/root/onos/apache-karaf-4.2.8/etc/keystore/keystore_ECDSA" -ext SAN=dns:onos1,ip:192.168.5.145

Com as chaves e certificados gerados é iniciado o processo de análise de pacotes (através do wireshark) para o algoritmo RSA e ECDHE.O teste é repetido três vezes para cada algoritmo, e são gerados três pedido de solicitações desde primeiro pacote até ao envio da resposta. As métricas Estabelecimento da Sessão, e conexão é calculado fazendo a diferença entre o tempo final e inicial no intervalo de tempo registado nos pedidos de solicitações.

Tabela 35-Legenda das métricas de avaliação

Métricas
Algoritmo TLS #algo
Número de teste #run
Estabelecimento da Sessão #tSessão
Estabelecimento da conexão #tConexão
2ºEstabelecimento de conexão #tConexãoReuse
1º Tempo de carregamento #tCar1
2º Tempo de carregamento #tCar2
3º Tempo de carregamento #tCar3

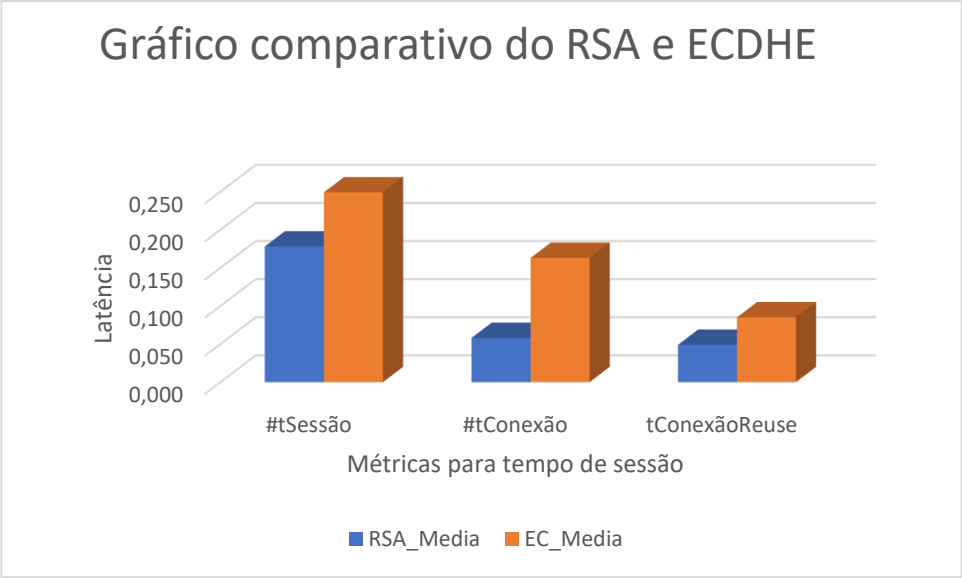
Para o algoritmo RSA e ECDHE foram registados os seguintes dados que podem ser analisados nas Tabela *36* e Tabela *37*.O registo completo da análise de pacotes pode ser consultado no Anexo D.

Tabela 36-Resultado algoritmo RSA

#algo	#run	#tSessão	#tConexão	#tConexãoReuse	#tCar1	#tCar2	# tCar3
RSA	1	0,107	0,054	0,047	320ms	117ms	119ms
RSA	2	0,283	0,055	0,071	598ms	153ms	98ms
RSA	3	0,143	0,063	0,029	437ms	150ms	71ms
Média		0,178	0,058	0,049	452ms	140ms	96ms
D.Padrão		0,076	0,004	0,017	113,966	16,310	19,647

Tabela 37-Resultado algoritmo ECDHE

#algo	#run	#tSessão	#tConexão	#tConexãoReuse	#tCar1	#tCar2	# tCar3
EC	1	0,288	0,315	0,098	788ms	402ms	183ms
EC	2	0,282	0,108	0,080	834ms	263ms	144ms
EC	3	0,176	0,067	0,078	635ms	181ms	165ms
Média		0,249	0,163	0,085	752ms	282ms	164ms
D.Padrão		0,051	0,108	0,009	85,066	91,218	15,937



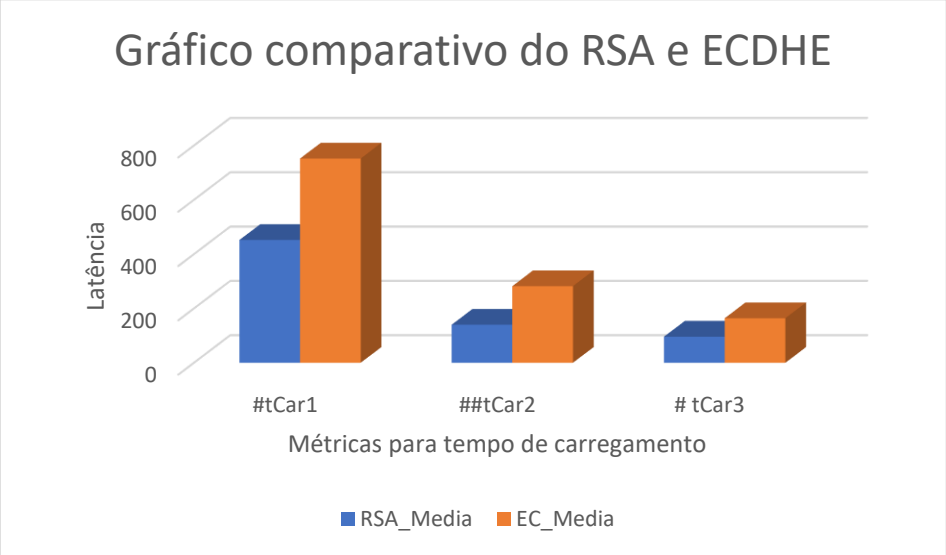


Figura 12-Gráficos comparativos do RSA e ECDHE

Os resultados obtidos na análise dos pacotes demonstram que o algoritmo ECDHE apresenta valores superiores ao algoritmo RSA em todas as métricas analisadas o que é refletido no tempo de carregamento nas três vezes que foi feito o teste, onde apresenta maior carga quando comparado ao RSA. De forma a validar que o ECDHE apresenta maior impacto que o RSA foi feito um teste simples de desempenho com recurso ao comando *s_time* do OpenSSL que solicita a página de login do UI do ONOS e inclui o tempo da transferência de dados nas suas medições. É recolhido o número de conexões num determinado período e calcula o tempo médio gasto para uma conexão. A Tabela *38* representa o comando *s_time* que vai recolher o número de conexões por 15 segundo na página web do ONOS.

Tabela 38-OpenSSL s_time

OpenSSL s_time
1: openssl s_time -connect 192.168.5.145:443 -www / -time 15

O teste foi feito para os algoritmos RSA e ECDHE tendo sido obtidos os seguintes resultados:

ECDHE

Tabela 39-OpenSSL ECDHE

ECDHE	Recolha das estatísticas de conexão por 15 segundos		
1	978 connections in 1.23s; 795.12 connections/user sec, bytes read 117360		
	978 connections in 16 real seconds, 120 bytes read per connection		
	tempo com a reutilização do id de sessão		
	1276 connections in 1.56s; 817.95 connections/user sec, bytes read 153120		
	1276 connections in 16 real seconds, 120 bytes read per connection		

RSA

*T*abela 40-OpenSSL RSA

RSA	Recolha das estatísticas de conexão por 15 segundos		
1	626 connections in 0.73s; 857.53 connections/user sec, bytes read 75120		
	626 connections in 16 real seconds, 120 bytes read per connection		
	tempo com a reutilização do id de sessão		
	921 connections in 1.08s; 852.78 connections/user sec, bytes read 110520		
	921 connections in 16 real seconds, 120 bytes read per connection		

Os resultados da *Tabela 39* Tabela 40 confirmam que o algoritmo de encriptação ECDHE tem mais impacto no sistema apresentando números superiores quando comparado com o RSA. Contudo o número de ligação suportada pelo ECDHE é superior atingindo um valor de 978 no mesmo espaço de tempo que o RSA (15 segundos), o valor médio de ligações por segundo é superior no RSA (852.98 em contraste com 817.95), o que está alinhado com os resultados das métricas observadas na recolha e análise de pacotes.

5.2.1 Avaliação do impacto do controlador com interfaces Seguras e sem Segurança

5.2.1.1 Interface NBI

O objetivo principal desta avaliação é saber o impacto direto que um controlador apresenta quando utiliza o TLS na NBI em comparação quando não é utilizado. Para isso é comparado os valores do tempo de carregamento total da página do ONOS no browser, na primeira, segunda e terceira solicitação.

Tabela 41-Controlador com TLS vs sem TLS

Configuração	1º Tempo de carregamento	2º Tempo de carregamento	3º Tempo de carregamento
TLS com RSA	320ms	117ms	119ms
	598ms	153ms	98ms
	437ms	150ms	71ms
Sem TLS	1º Tempo de carregamento	2º Tempo de carregamento	3º Tempo de carregamento
	139ms	111ms	52ms
	281ms	48ms	49ms
	262ms	44ms	51ms

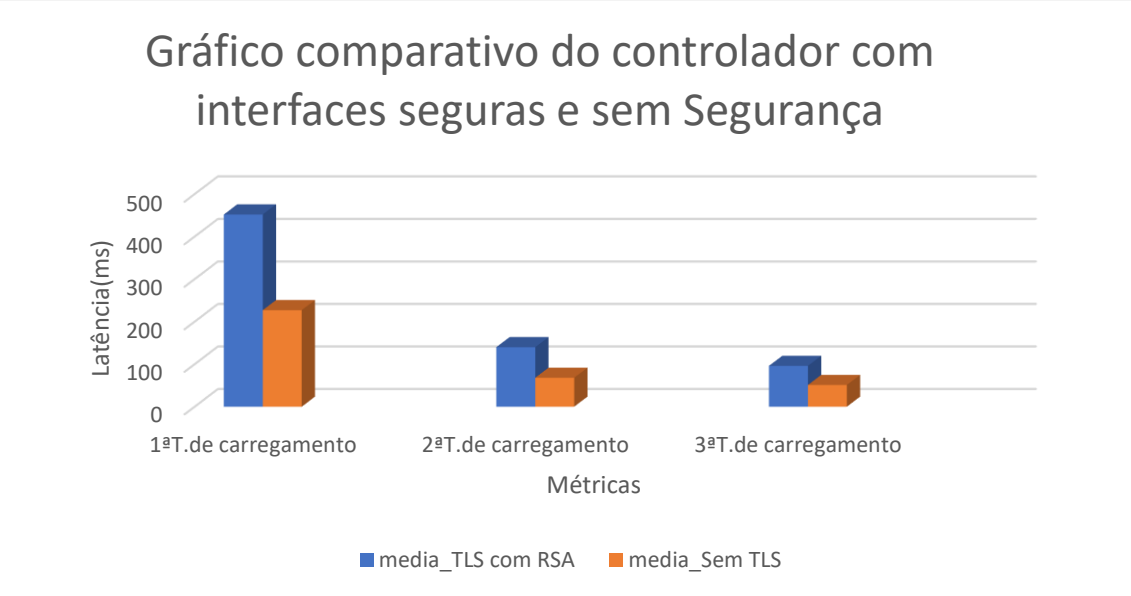


Figura 13-Gráfico comparativo das interfaces com e sem segurança

Os dados da Figura 13 demonstram uma grande diferença em relação ao tempo de carregamento da página ONOS, onde fica claro que um controlador que utiliza o TLS no NBI demora mais tempo estabelecer comunicação quando comparado com um controlador que não utiliza o TLS, isto pode estar associado ao processo de negociação das chaves que é estabelecida quando se faz uma solicitação.

5.2.1.2 Avaliação do impacto do controlador com as Interfaces SBI e EBI/WBI Seguro

A estratégia de avaliação prevista foca-se no impacto que estas interfaces (SBI, EBI,WBI) seguras podem ter nos controladores. Semelhante a estratégia adotada na avaliação do NBI (secção 5.1), o objetivo é analisar todas as comunicações nestas interfaces seguras e ter dados suficientes para fazer o estudo comparativo do impacto num controlador que não tenha estas interfaces em modo seguro. Contudo esta estratégia não foi avaliada pois ao longo dos testes não foi possível verificar a estabilidade entre a ligação dos switches OvS e o controlador ONOS. Verificou-se nos logs do controlador que a ligação segura era estabelecida, mas passado algum tempo, a ligação era terminada de forma aleatória, não permitindo verificar a informação da topologia na UI do ONOS.

Na prática esta análise passaria por avaliar todos os pacotes trocados nas comunicações entre os controladores e ver todo o processo de encriptação que ocorre durante esta transmissão e também avaliar os pacotes transmitidos na SBI com o plano de dados (exemplo: pacotes provindos de switches, de routers) em diferentes topologias de rede. Porém como já foi mencionado não havia estabilidade na ligação entre os switches OvS e o controlador e optou-se por não fazer as medições, pois a informação nas diferentes topologias testadas não era reportada de forma correta.

Para o processo de avaliação fez-se o estudo da ferramenta *´ss´* disponibilizada pelo Linux que permite analisar toda as estatísticas dos sockets em específico as informações dos protocolos Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Datagram Congestion Control Protocol (DCCP). Utilizando o comando *´ss´* podemos obter informações mais detalhados das ligações que ocorrem na rede desta forma podemos acompanhar o processo de estabelecimento de comunicação entre os controladores e também entre os controladores e os dispositivos no plano de dados.
Uma das métricas fornecidas pela ferramenta ss é o Round-trip time (RTT) que permite determinar o tempo que um pedido na rede demora para ir de um ponto de origem para o destino e de volta para o ponto de partida. Esta métrica permite avaliar a nível de segurança na integridade nas comunicações entre os controladores e o plano de dados, na prática saber a velocidade do processo de encriptação e os valores de RTT quando estes dados são transmitidos. Devido à dimensão dos resultados do comando *´ss´*, é apresentado através da Figura 14 um trecho dos resultados gerados entre a comunicação do controlador ONOS e o Atomix. Os resultados obtidos podem ser consultados no repositório git disponibilizado.

```
#########:1635346889
Netid  State      Recv-Q Send-Q     Local Address:Port       Peer Address:Port
tcp    ESTAB      0      0            172.20.0.10:56976       172.20.0.21:9876
         skmem:(r0,rb425984,t0,tb425984,f0,w0,o0,bl0) cubic wscale:7,2 rto:204 rtt:1.017/0.16 ato:40 mss:1398 cwnd:10 send 110.0Mbps p
tcp    ESTAB      0      0            172.20.0.10:5679        172.20.0.20:56198
         skmem:(r0,rb425984,t0,tb425984,f0,w0,o0,bl0) cubic wscale:2,7 rto:204 rtt:0.121/0.015 ato:40 mss:1398 cwnd:10 send 924.3Mbps
tcp    ESTAB      0      0            172.20.0.10:5679        172.20.0.20:56238
         skmem:(r0,rb425984,t0,tb425984,f4096,w0,o0,bl0) cubic wscale:2,7 rto:204 rtt:0.124/0.059 ato:40 mss:1398 cwnd:10 send 901.9Mb
tcp    ESTAB      0      0            172.20.0.10:33860       172.20.0.11:5679
         skmem:(r0,rb425984,t0,tb425984,f0,w0,o0,bl0) cubic wscale:7,2 rto:204 rtt:3.654/0.85 ato:40 mss:1398 cwnd:10 send 30.6Mbps pa
tcp    ESTAB      0      0            172.20.0.10:5679        172.20.0.20:56256
         skmem:(r0,rb425984,t0,tb425984,f0,w0,o0,bl0) cubic wscale:2,7 rto:204 rtt:0.121/0.03 ato:40 mss:1398 cwnd:10 send 924.3Mbps p
tcp    ESTAB      0      0            172.20.0.10:5679        172.20.0.20:56218
         skmem:(r0,rb425984,t0,tb425984,f0,w0,o0,bl0) cubic wscale:2,7 rto:204 rtt:0.118/0.02 ato:204 mss:1398 cwnd:10 send 947.8Mbps
tcp    ESTAB      0      0            172.20.0.10:57980       172.20.0.20:9876
         skmem:(r0,rb425984,t0,tb425984,f0,w0,o0,bl0) cubic wscale:7,2 rto:204 rtt:0.509/0.855 ato:40 mss:1398 cwnd:10 send 219.7Mbps
tcp    ESTAB      0      0            172.20.0.10:5679        172.20.0.11:41644
         skmem:(r0,rb425984,t0,tb425984,f0,w0,o0,bl0) cubic wscale:2,7 rto:204 rtt:1.204/0.433 ato:40 mss:1398 cwnd:10 send 92.9Mbps p
```

Figura 14-output do comando ´ss´

5.3 Finalização do capítulo

Neste capítulo foi abordado a validação e análise do impacto da implementação de segurança nas diferentes interfaces do controlador ONOS. Na primeira avaliação foi na NBI onde através de uma análise em duas vertentes onde, na primeira foi avaliado como é que a implementação de diferentes algoritmos de encriptação na NBI pode impactar o desempenho, e na segunda foi avaliado o impacto quando temos um controlador com TLS implementado e outro controlador sem o TLS. Na primeira vertente ficou provado que o algoritmo ECDHE tem mais impacto no sistema apresentando números superiores nas métricas de impacto quando comparado com o RSA, na segunda avaliação ficou provado que um controlador que o TLS na NBI demora mais tempo em estabelecer comunicação quando comparado com controlador que não utilizada o TLS.
Foi apresentado a estratégia de avaliação prevista para o SBI, EBI e WBI que não foram avaliados porque não foi possível determinar a estabilidade entre as ligações dos switches OvS e o controlador ONOS. Podemos concluir que a implementação do TLS nas interfaces vai depender do cenário e os objetivos na qual esta inserido o controlador, pois através dos teste feitos ainda não é possível verificar de forma conclusiva nas interfaces SBI, EBI e WBI o real impacto que estes podem ter no controlador.

6 Conclusão

O trabalho realizado, demonstra que as redes SDN terão um papel importante no futuro das redes sobretudo as de grandes dimensões. Com a flexibilidade que a arquitetura SDN fornece, cada vez mais teremos redes a expandirem em dimensão ao mesmo tempo que reduzem os seus custos operacionais da sua infraestrutura. De realçar que apesar da implementação de redes SDN já ser a realidade de muitas organizações e infraestruturas, esta ainda se encontra em uma fase inicial sobretudo na parte da segurança.

Tendo isto em mente a primeira contribuição do trabalho foi uma análise sistemática das vulnerabilidades onde foi feito o levantamento de todas vulnerabilidades e ameaças que cada controlador possa ter, onde os dados recolhidos serviram de base para a metodologia de classificação que nos permite classificar estas vulnerabilidades através de várias métricas desta forma percebemos os planos mais afetados e quais as áreas da rede SDN precisam de mais proteção.

A flexibilidade que uma rede SDN fornece leva a disponibilidade de uma diversidade de controladores com diversas características próprias em termos de segurança e desempenho, tendo isto em mente a escolha do controlador deve ser um processo detalhado abordando aspetos de design à produção. O presente trabalho propõe um mecanismo denominado CROCUS que permite realizar uma avaliação objetiva na escolha dos controladores tendo em conta o seu suporte para segurança. O CROCUS é feito levando em consideração os dados disponibilizados pelo Common Vulnerability Scoring System (CVSS) e considera cenários com requisitos de segurança rigorosos. Com o CROCUS foi possível concluir que o controlador ONOS apesar de apresentar mais vulnerabilidades quando comparado com o OpenDaylight, tem uma melhor percentagem de resolução das vulnerabilidades e as vulnerabilidades do ONOS resultam de eventos e interações entre componentes do controlador.

Um dos desafios encontrados na realização do trabalho foi a falta de documentação correta e atualizada das configurações de segurança nos controladores, tendo isto em conta foi disponibilizado em modo Open-Source (repositório git[1]) o código e documentação de configuração do TLS para as interfaces (NBI, SBI, EBI, WBI) do ONOS, para que possa ser utilizado quando é necessário deixar as interfaces mais seguras com recurso a configuração do protocolo TLS.

O trabalho futuro passa por melhorar a avaliação do impacto da implementação de segurança nas interfaces (NBI, SBI, EBI, WBI), isto pode ser feito utilizando novas métricas de avaliação ou a integração de outras tecnologias, outro aspeto seria automatizar e simplificar o processo de implementação da segurança nas interfaces do controlador. Outra melhoria seria a integração do CROCUS em controladores SDN para melhorar o suporte de segurança em tempo real.

[1] Acessível em: https://github.com/carlosfelix12/ONOS

Esta página foi intencionalmente deixada em branco.

Referências

[1] F. M. J. P. J. ,. L. J. ,. M. N. S. S. Casado Martín, “Ethane: Taking Control of the Enterprise,” 2009.

[2] L. T. Majd Latah, “Application of Artificial Intelligence to Software Defined Networking: A Survey,” 2017.

[3] “Open Networking Foundation,” [Online]. Available: http://opennetworking.org/wp-content/uploads/2014/11/TR_SDN-ARCH-1.0-Overview-12012016.04.pdf. [Acedido em Setembro 2020].

[4] “Northbound Interface Work Group:,” [Online]. Available: http://opennetworking.org/news-and-events/press-releases/open-networking-foundation-introduces-northbound-interface-working-group/ . [Acedido em setembro 2020].

[5] “Open Networking Foundation,” [Online]. Available: https://wiki.onosproject.org/display/ONOS/Southbound+protocols . [Acedido em Setembro 2020].

[6] “IETF RFC 7047,” [Online]. Available: https://tools.ietf.org/html/rfc7047. [Acedido em Setembro 2020].

[7] T. Z. T. H. P. T.-G. K. Michael Jarschel, “Interfaces, Attributes, and Use Cases: A Compass for SDN,” 2014.

[8] C. J. S. I. J. H.Cox Jacob, “Advancing Software-Defined Networks: A Survey,” Outubro 2017.

[9] I. H. E. A. K. A. C. Ola Salman, “SDN Controllers: A Comparative Study,” em *Conference: 2016 18th Mediterranean Electrotechnical Conference (MELECON)*, 2016.

[10] S. S. A. M. F. Bannour, “Distributed SDN Control: Survey, Taxonomy, and Challenges,” p. 333–354.

[11] K. M. S. K. Z. Liehuang, “SDN Controllers: Benchmarking & Performance Evaluation,” Fevereiro 2019.

[12] Z. A. M. R. “.-b. K. Rahamatullah, “Comparison and Selection of Sofware Defined Networking (SDN) Controllers,” Janeiro 2014.

[13] A. L. S. D. – L. Mamushiane, “Comparative Evaluation of the Performance of popular SDN controllers,” em *10th Wireless Days Conference*, 2018.

[14] “ONOS Foundation,” [Online]. Available: https://opennetworking.org/onos/. [Acedido em Outubro 2020].

[15] “Atomix,” [Online]. Available: https://atomix.io/docs/latest/user-manual/introduction/what-is-atomix. [Acedido em Outubro 2020].

[16] “Wiki onosproject,” [Online]. Available: https://wiki.onosproject.org/display/ONOS/Cluster+Coordination. [Acedido em Outubro 2020].

[17] “onosproject,” [Online]. Available: https://wiki.onosproject.org/display/ONOS/Distributed+Primitives. [Acedido em outubro 2020].

[18] “Raft,” [Online]. Available: https://raft.github.io/raft.pdf. [Acedido em Outubro 2020].

[19] “wiki onosproject,” [Online]. Available: https://wiki.onosproject.org/display/ONOS/Cluster+Coordination. [Acedido em Outubro 2020].

[20] “mininet,” [Online]. Available: http://mininet.org. [Acedido em Outubro 2020].

[21] “mininet,” [Online]. Available: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet. [Acedido em Outubro 2020].

[22] “opennetworking,” [Online]. Available: https://opennetworking.org/mininet/. [Acedido em Outubro 2020].

[23] “atomix,” [Online]. Available: https://atomix.io/community/#contributors. [Acedido em Agosto 2021].

[24] “Atomix,” [Online]. Available: https://atomix.io/docs/latest/user-manual/concepts/cluster/. [Acedido em Setembro 2021].

[25] E. Rescorla, “RFC 8446,” [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8446. [Acedido em Setembro 2021].

[26] W. Stallings, Cryptography and Network Security: Principles and Practice, Prentice Hall Press, 2017.

[27] A. L. a. S. D. L. Mamushiane, “A comparative evaluation of the performance of popular SDN controllers,” p. 54–59, Abril 2018.

[28] K. A. a. R. J. S. Dong, *A Survey on Distributed Denial of Service (DDoS) Attacks in SDN and Cloud Computing Environments,* 2019.

[29] “opennetworking.org,” [Online]. Available: https://www.opennetworking.org/wp-content/uploads/2014/10/Threat_Analysis_for_the_SDN_Architecture.pdf. [Acedido em Novembro 2020].

[30] “first org,” [Online]. Available: https://www.first.org/cvss/specification-document. [Acedido em Fevereirro 2021].

[31] S.Scott-Hayward, “Design and deployment of secure, robust, and resilient sdn controllers,” pp. 1-5, 2015.

[32] “cve.mitre,” [Online]. Available: https://cve.mitre.org/cve/. [Acedido em Março 2021].

[33] “cvedetails,” [Online]. Available: https://www.cvedetails.com.

[34] M. F.-B. E. J. W. &. S. Schumacher, Security Patterns: Integrating security and systems engineering, John Wiley & Sons.

[35] K. P. a. M. C. B. Sousa, *Methodical: Towards the next generation multihomed applications,* 2014.

[36] “ONOS wiki,” [Online]. Available: https://wiki.onosproject.org/pages/viewpage.action?pageId=28836788. [Acedido em junho 2021].

[37] H. Jordan, “opennetworking,” [Online]. Available: https://opennetworking.org/wp-content/uploads/2018/12/Distributed-Systems-in-ONOS-with-Atomix-3.pdf. [Acedido em Junho 2021].

[38] “onosproject,” [Online]. Available: https://wiki.onosproject.org/display/ONOS/Configuring+TLS+for+inter-controller+communication. [Acedido em Junho 2021].

[39] C. Silva, “GitHub,” [Online]. Available: https://github.com/carlosfelix12/ONOS.

Esta página foi intencionalmente deixada em branco

Anexo A

Artigo Científico

Este anexo incluí o artigo produzido e publicado na conferência EAI SecureComm[2].

C.Silva , B.Sousa, J.Vilela, “CROCUS: An objective approach for SDN controllers security assessment”, Setembro 2021.

[2] https://securecomm.eai-conferences.org/2021/accepted-papers/

CROCUS: An objective approach for SDN controllers security assessment

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

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

2CRACS/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 research [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

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

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 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

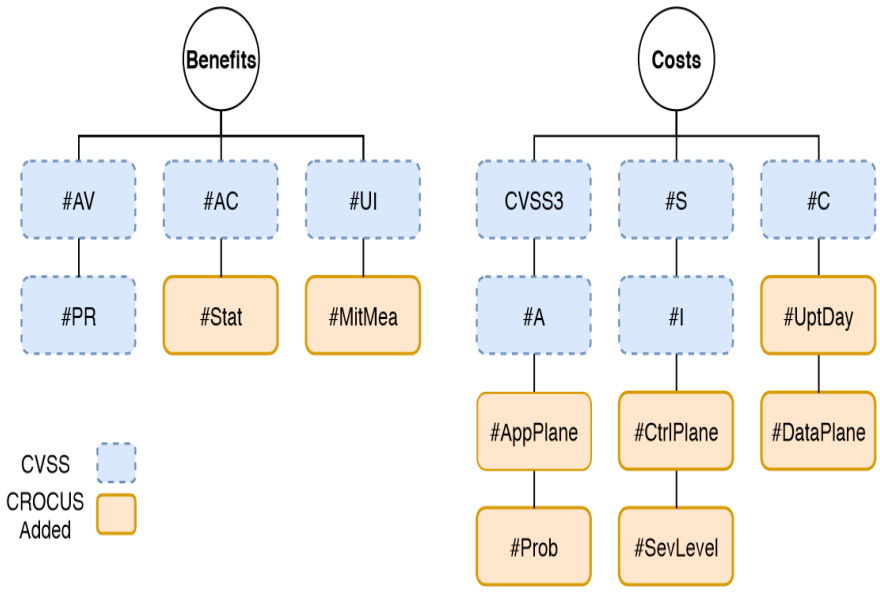


Fig.1: CROCUS benefits and costs metrics from Tables 1 & 2 in ranking step.

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 ≥ 4). The $TOTuptDayCri$ corresponds to the sum of the averages days to solve critical vulnerabilities in all N controllers.

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

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 $wR + wV + wS = 1$.

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

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 summarized 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 ≈ 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

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.

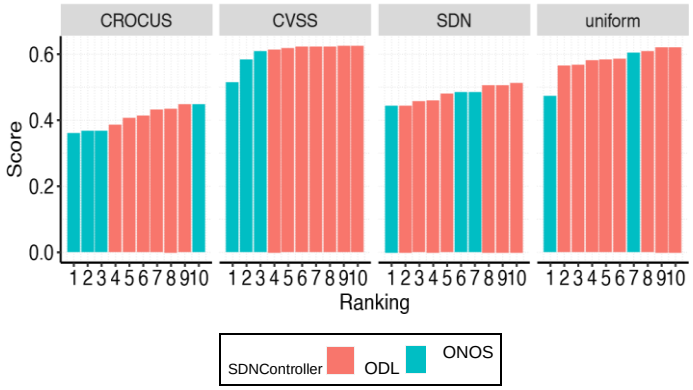


Fig.2: TopTen per weight set and *equalGrp* weight category

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.

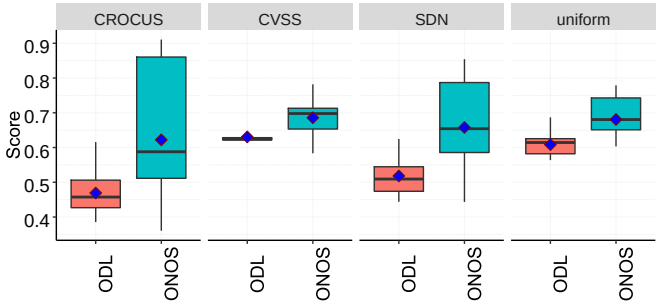


Fig.3: Variation of vulnerability score per controller and weight set

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

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

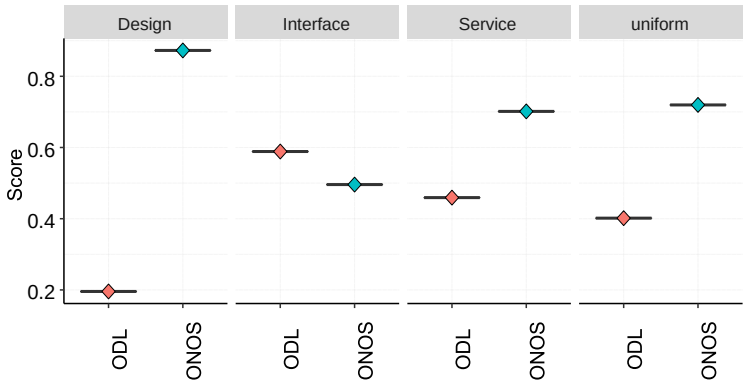


Fig.4: Variation of score per controller and weight set

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 - *V So* and the average score of security functionalities *SeSo*. Fig. 5 illustrates the CROCUS score for the ODL and ONOS controllers.

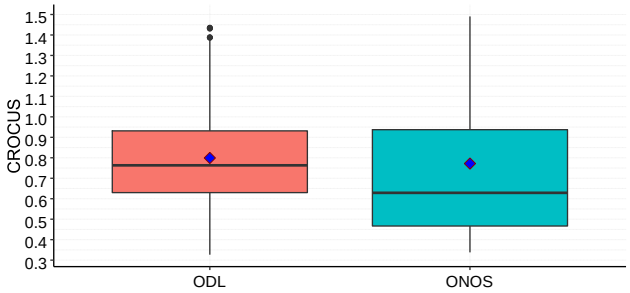


Fig.5: CROCUS

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 (≈ 1.5), while ODL has around ≈ 1.4 (with some outliers). Nonetheless, the variation between the minimum value and the first quartile (25 th percentile) is lower for ONOS (≈ 0.45, while ODL has ≈ 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. Lu´ıs, D. Raposo, F. Michel, S. Sargento, and M. M´edard, “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. Arau´jo, 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.

Anexo B

Este anexo apresenta informações sobre o ONOS Security Mode mencionado no capítulo 4.

O controlador ONOS (Open Network Operantig System) fornece APIs para a interface Northbound que permite a comunicação e desenvolvimento de aplicações no plano de aplicação. Estas APIs são fáceis de utilizar, mas por outro lado devido ao nível de acesso que lhe é concedido de forma a ter o máximo de desempenho possível, permite que as aplicações do ONOS realizem qualquer ação na rede.
As aplicações do ONOS devido ao nível de acesso que possuem, podem ser considerados um ponto vulnerável na rede, que se forem utilizadas com fins maliciosos comprometem a rede pois algumas dessas aplicações possuem recursos que podem alterar o comportamento da rede. O ONOS disponibiliza o recurso Security Mode para redes com determinados requisitos, desta forma os administradores de rede podem configurar o controlador de forma a restringir a capacidade de acesso das aplicações.
Na prática o Security Mode adiciona dois recursos principais ao controlador ONOS:

- **Autenticação da aplicação.**
- **Controlo de acesso baseado em regras e permissões.**

O Security mode exige que os administradores de rede revejam e aceitem as políticas de segurança de cada aplicação antes deste ser ativado. Quando o controlo de acesso é baseado em permissões uma aplicação não pode utilizar os recursos de forma indiscriminada, ou seja, estes têm de ser explicitamente permitidos. Estas permissões estão relacionadas com os serviços e APIs que uma aplicação poderá utilizar para interagir com o controlador ONOS, por exemplo uma aplicação com permissão do tipo APP READ pode aceder em modo de leitura às informações das aplicações instalados.
Estas permissões são definidas no ficheiro app.xml das aplicações, desta forma sempre que a aplicação for instalada estas permissões são revistas e sujeita a aprovação, caso contrário se este processo não for feito, a ativação da aplicação é tida como uma violação de políticas de segurança.
A representa a lista de algumas das permissões que o ONOS disponibiliza para serem atribuídas para o funcionamento de uma aplicação. De realçar que as permissões se encontram descritas em inglês por não haver uma tradução direta para algumas das permissões.

TYPE	Description	SERVICES	APIs
APP_READ	Permission to read various information about installed applications	Application Service	getApplication(s) getState getId getPermissions
		Core Service	version getAppId(s) getIdGenerator
APP_WRITE	Permission to register new application	Core Service	registerApplication
APP_EVENT	Permission to receive application lifecycle events	Application Service	addListener removeListener
CONFIG_READ	Permission to read configuration properties	Component Config Service	getComponentNames getProperties
		Network Config Service	getSubjectClasees getSubjectFactory getConfigClass getSubjects getConfig
CONFIG_WRITE	Permission to write configuration properties	Component Config Service	registerProperties unregisterProperties setProperty unsetProperty preSetProperty
		Network Config Service	addConfig applyConfig removeConfig
CODEC_READ	Permission to read codec information	Codec Service	getCodec(s)
CODEC_WRITE	Permission to add/remove entityclass from codecs	Codec Service	registerCodec unregisterCodec
CLOCK_WRITE	Permission to write clock properties	Logical Clock Service	getTimestamp
CLUSTER_READ	Permission to read cluster information	Cluster Service	getLocalNode getNode(s) getState getLastUpdated

		Cluster Metadata Service	getClusterMetadata getLocalNode
		MastershipTermService	getMastershipTerm
		Leadership Service	getLeader getLeadership ownedTopics getLeaderboard getCandidates
		Mastership Service	getLocalRole getMasterFor getNodesFor getDevicesOf
CLUSTER_WRITE	Permission to modify the cluster	Leadership Service	runForLeadership withdraw
		Mastership Service	requestRoleFor relinquishMastership
CLUSTER_EVENT	Permission receive cluster events	Cluster Service	addListener removeListener
		Leadership Service	addListener removeListener
		Mastership Service	addListener removeListener

Tabela 42-Permissões de Aplicações [33]

Como já mencionado anteriormente o SecurityMode exige que os administradores de rede revejam e aceitem as permissões de segurança (representados na Tabela *42*) de cada aplicação antes deste ser ativado. A Figura 15 representa este processo de validação e aceitação de permissões. No exemplo da Figura 16 temos uma aplicação de monitorização de tráfego da rede onde é necessário dar permissões explícitas para receber os eventos da rede, ver as alterações na topologia da rede, e poder adicionar e remover regras de fluxo.

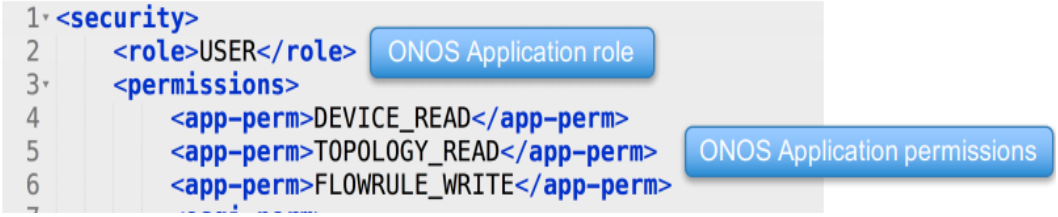


Figura 15-Exemplo de permissões[34]

Figura 16-Revisão e Validação e Permissões [35]

Controlo de acesso das aplicações ONOS

O processo de validação do controlo de acesso (representado na Figura 15) das aplicações ONOS consiste em três etapas:

- **Controlo de acesso baseado em função do nível de pacote.**
- **Controlo de acesso baseado em função do nível de aplicação**
- **Controlo de acesso baseado em permissão do nível de API.**

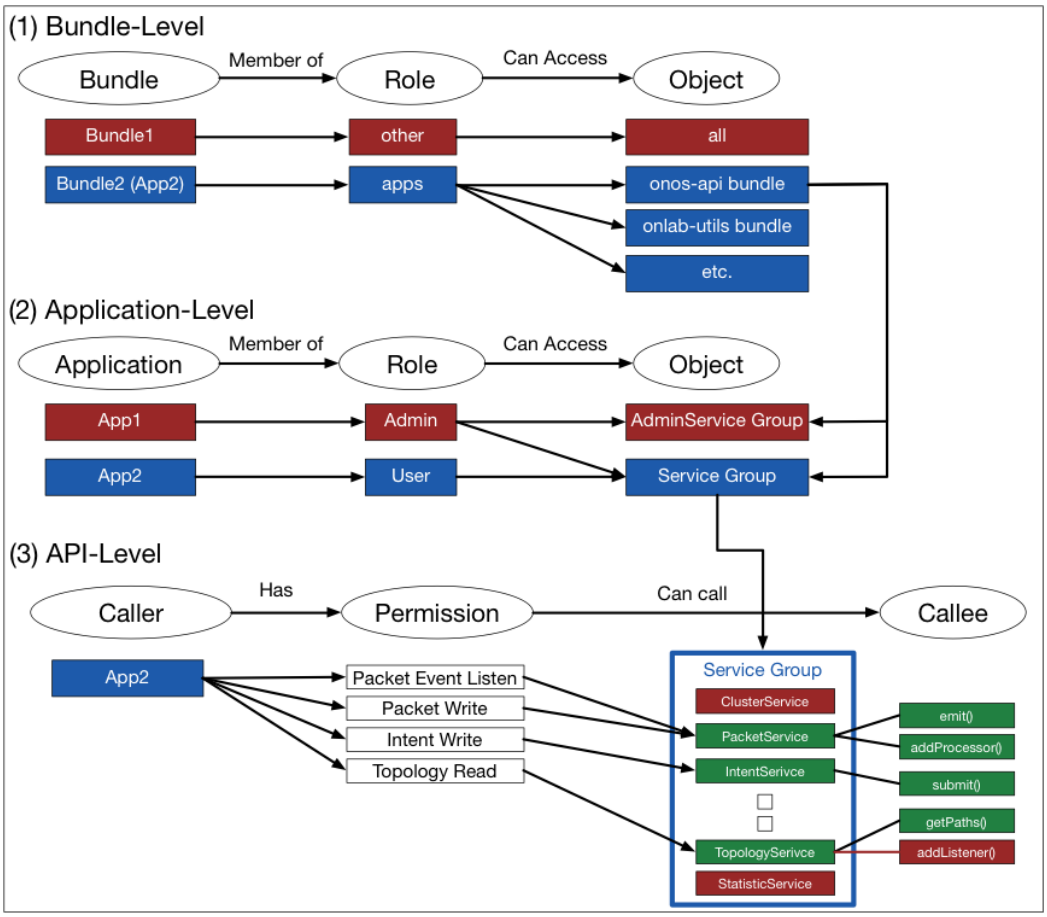


Figura 17-Mecanismo de controlo de acesso[36]

Controlo de acesso baseado em função do nível de pacote:

- Este controlo propõe que todos os pacotes tenham uma de duas funções**:** "aplicações" e "não aplicações", sendo que este último não permite que a aplicação aceda as APIs da interface Northbound.Com base na função atribuída o controlo de acesso baseado em função é aplicado a cada pacote, conforme apresentado na Figura 16(1). Esse tipo de controlo de acesso permite apenas pacotes com a função " aplicações " possam aceder aos pacotes das APIs da interface Northbound e APIs de utilitários selecionados.

Controlo de acesso baseado em função do nível de aplicação:

- Existem APIs na interface Northbound com funções administrativas utilizadas para operações sensíveis na rede, em que o administrador pode querer restringir o acesso a determinadas aplicações. De acordo com a "role" especificada no arquivo de configuração da aplicação, o nível de controlo de acesso será aplicado, ou seja, se uma aplicação for especificada como "administrador, esta pode ter acesso aos serviços administrativos e não administrativos como pode ser verificado na figura 14 (2), por outro lado se for do tipo "não administrativa", só pode aceder a serviços regulares que não executam operações sensíveis que podem alterar a topologia de rede.

  A Figura *18* apresenta os serviços administrativos e não administrativos do ONOS (a partir da versão 1.0.0) , e a descrição de alguns desses serviços na Tabela *43*.

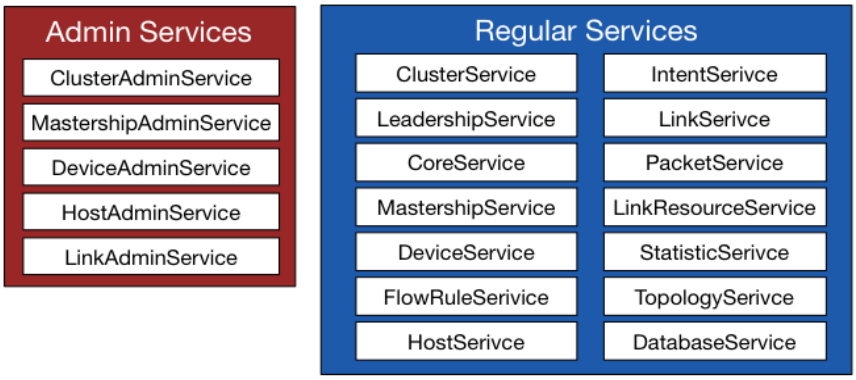


Figura 18-Serviços administrativos e não administrativo [37]

Tabela 43-Descrição dos serviços administrativos e regulares

Serviços Administrativos	**Descrição**
ClusterAdminService	Serviço para administrar a associação dos nós do cluster
MastershipAdminService	Serviço para administrar o inventário dos nós com posição de Master
DeviceAdminService	Serviço para administrar o inventário dos dispositivos da infraestrutura
HostAdminService	Serviço para administrar o inventário de hosts no plano de dados
LinkAdminService	Serviço de administração de inventario dos links da infraestrutura
Serviços Regulares	**Descrição**
ClusterService	Serviço para obter informações dos nós individuais no cluster
FlowRuleService	Serviço para injetar regras de fluxo no ambiente e obter informações sobre regras de fluxo já existentes no ambiente
TopologyService	Serviço de fornecimento de informações da topologia de rede
PacketService	Serviço para intercetar pacotes no plano de dados e emitir pacotes de saídas.
CoreService	Serviço de interação com o controlador

Controlo de acesso baseado em permissão do nível de API:

- Para que uma aplicação possa utilizar as permissões o ONOS propõe que estes devem estar explicitamente permitidas no ficheiro de configuração de forma a poder utilizar as APIs que precisa para o seu funcionamento. Para utilizar uma permissão esta deve ser representado de forma lógica e mapeado para os conjuntos de APIs que vai utilizar Por exemplo na Figura 16, a permissão para o envio de intenções deve ser representada como "onos.permission.INTENT_WRITE".

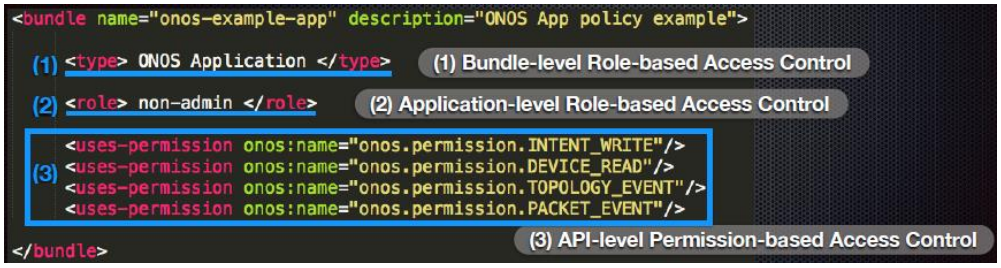


Figura 19-Exemplo de um ficheiro de políticas

O ONOS Security Mode apresenta diversas vantagens, a nível de segurança, contudo não permite o controlo de acesso refinado. Ou seja, não permite garantir de forma efetiva o acesso total ou negar o acesso a determinados recursos. Por exemplo não permite ter a noção das funções dos perfis de utilizadores e saber o nível de acesso das aplicações utilizadas por eles. Outra limitação é que a fim de utilizar de forma eficaz a Security mode as políticas de permissões devem ser criadas para todas as aplicações em uso, invés de ser gerada de forma automática, por exemplo o ONOS através de uma análise estática da aplicação, pudesse atribuir as permissões necessárias para seu funcionamento.

Anexo C

Este anexo apresenta informações das tabelas das vulnerabilidades para ambos os controladores apresentados na secção 3.3.3.

Vulnerabilidades OpenDaylight

Métricas de segurança									
#RS	**#IRP**	**#IMC**	**#AS**	**#CCS**	**#API**	**#IDS**	**#AA**	**#MR**	**#AuD**
1	1	1	1	1	1	1	1	1	1

Opendaylight															Ativos Afetados			Impacto nos planos				
CVE-id	**Estado**	**Data de Public.**	**Data de Atual.**	**Days**	**CVSS3**	**#VT**	**#C.AT**	**#PR**	**#E**	**#IU**	**#Conf**	**#Int.**	**#Disp.**	**#M.miti**	**A.plano aplicação**	**A.plano de controlo**	**A.plano de dados**	**P.de conf**	**P.de Inte**	**P.dispo**	**#N.de sev**	**#P.de ocorrência**
CVE-2018-10898	1	30/07/2018	09/10/2019	436	3	2	1	0	1	0	2	2	2	1	0	1	0	1	0	0	3	3
CVE-2018-1132	1	20/06/2018	09/10/2019	476	4	1	1	0	1	0	2	2	2	1	0	1	0	1	1	0	3	3

CVE-2018-1078	1	16/03/2018	09/10/2019	572	4	1	1	0	1	0	2	2	2	2	0	1	0	0	0	1	3	3
CVE-2017-1000411	1	31/01/2018	03/10/2019	610	3	1	1	0	1	0	0	0	2	1	0	0	0	0	0	1	2	3
CVE-2017-1000406	1	30/11/2017	20/12/2017	20	3	1	1	0	1	0	0	2	0	1	0	0	0	0	1	0	2	3
CVE-2015-1778	1	27/06/2017	05/07/2017	8	4	1	1	0	1	0	2	2	2	2	1	1	0	1	0	0	3	3
CVE-2014-8149	1	27/06/2017	03/07/2017	6	3	1	1	1	1	0	2	2	2	2	1	1	0	1	1	1	3	3
CVE-2017-1000361	1	24/04/2017	03/10/2019	892	3	1	1	0	1	0	0	0	2	1	0	1	0	0	0	1	2	3
CVE-2017-1000357	1	24/04/2017	02/10/2019	891	3	1	1	0	1	0	0	0	2	1	0	1	0	0	0	1	2	3
CVE-2016-4970	1	13/04/2017	14/02/2021	1403	3	1	1	0	1	0	0	0	2	1	0	1	0	0	0	1	2	3
CVE-2015-1612	1	04/04/2017	11/04/2017	7	3	1	1	0	1	0	0	2	0	2	0	1	1	0	1	0	2	3
CVE-2015-1611	1	04/04/2017	11/04/2017	7	3	1	1	0	1	0	0	2	0	2	0	1	1	0	1	0	2	3

Vulnerabilidades ONOS

Métricas de segurança									
#RS	**#IRP**	**#IMC**	**#AS**	**#CCS**	**#API**	**#IDS**	**#AA**	**#MR**	**#AuD**
1	1	1	1	1	1	0	1	1	1

ONOS															Ativos Afetados			Impacto nos planos				
CVE-id	**Estado**	**Data de Public.**	**Data de Atual.**	**Days**	**CVSS 3**	**#VT**	**#C.AT**	**#PR**	**#E**	**#IU**	**#Conf**	**#Int.**	**#Disp.**	**#M.miti**	**A.plano aplicação**	**A.plano de controlo**	**A.plano de dados**	**P.de conf**	**P.de Inte**	**P.dispo**	**#N.de sev**	**#P.de ocorrência**
CVE-2020-35604	1	21/12/2020	25/12/2020	4	4	1	1	0	1	0	2	2	2	1	1	1	0	1	0	0	3	3
CVE-2019-16302	0	20/02/2020	25/02/2020	5	3	1	1	0	1	0	0	0	2	0	1	1	0	0	0	1	2	2
CVE-2019-16301	0	20/02/2020	25/02/2020	5	3	1	1	0	1	0	0	0	2	0	1	1	0	0	0	1	2	2
CVE-2019-16300	0	20/02/2020	25/02/2020	5	3	1	1	0	1	0	0	0	2	0	1	1	0	0	0	1	2	2

CVE-2019-16299	0	20/02/2020	25/02/2020	5	3	1	1	0	1	0	0	0	2	0	1	1	0	0	0	1	2	2
CVE-2019-16298	0	20/02/2020	25/02/2020	5	3	1	1	0	1	0	0	0	2	0	1	1	0	0	0	1	2	2
CVE-2019-16297	0	20/02/2020	25/02/2020	5	3	1	1	0	1	0	0	0	2	0	1	1	0	0	0	1	2	2
CVE-2019-11189	0	20/02/2020	28/02/2020	8	3	1	1	0	1	0	0	2	0	0	0	1	1	0	1	0	2	3
CVE-2020-8495	1	30/01/2020	06/02/2020	7	3	1	2	1	1	0	2	2	2	1	1	1	1	1	1	0	3	2
CVE-2020-8494	1	30/01/2020	06/02/2020	7	3	1	1	1	1	0	2	2	2	1	1	1	1	1	1	0	3	2
CVE-2019-18418	0	24/10/2019	29/10/2019	5	4	1	1	0	1	0	2	2	2	0	1	1	0	1	0	0	3	2
CVE-2019-12587	1	04/09/2019	24/08/2020	355	3	2	1	0	1	0	2	2	0	1	0	1	1	0	1	1	2	3
CVE-2019-15571	1	26/08/2019	03/09/2019	8	4	1	1	0	1	0	2	2	2	1	1	1	0	1	1	0	3	3

CVE-2019-10102 34	1	22/07/20 19	25/07/20 19	3	4	1	1	0	1	0	2	2	2	1	1	1	0	1	1	1	3	3
CVE-2019-10102 45	1	19/07/20 19	25/07/20 19	6	4	1	1	0	1	0	2	2	2	2	1	1	0	1	1	1	3	2
CVE-2019-13624	1	16/07/20 19	19/07/20 19	3	4	1	1	0	1	0	2	2	2	2	1	1	0	1	1	1	3	2
CVE-2018-15868	1	21/06/20 19	24/06/20 19	3	4	1	1	0	1	0	2	2	2	1	1	1	0	1	1	0	3	2
CVE-2018-10006 16	1	09/07/20 18	04/09/20 18	57	4	1	1	0	1	0	2	2	2	2	0	1	0	1	1	1	3	2
CVE-2018-10006 14	1	09/07/20 18	04/09/20 18	57	4	1	1	0	1	0	2	2	2	2	0	1	0	1	1	1	3	2
CVE-2018-11316	1	03/07/20 18	11/09/20 18	70	4	1	1	0	2	1	2	2	2	1	0	1	1	1	1	0	3	4
CVE-2018-11314	1	03/07/20 18	11/09/20 18	70	4	1	1	0	2	1	2	2	2	1	0	1	1	1	1	0	3	4

CVE-2018-10001155	1	24/05/2018	03/10/2019	497	4	1	1	0	1	0	2	2	2	2	0	1	1	0	1	1	3	3
CVE-2014-8129	1	01/03/2018	06/04/2018	36	3	1	1	0	1	1	2	2	2	1	0	1	0	0	0	1	3	2
CVE-2018-5452	1	07/03/2018	18/09/2020	926	3	1	1	0	1	0	0	0	2	1	0	1	1	0	0	1	3	2
CVE-2017-13763	1	29/08/2017	03/10/2019	765	3	1	1	0	1	0	0	0	2	2	0	1	1	0	1	1	2	2
CVE-2015-7516	1	24/08/2014	30/08/2017	1102	3	1	1	0	1	0	0	0	2	2	0	1	1	0	0	1	3	2
CVE-2017-1000081	1	17/07/2017	07/12/2020	1239	4	1	1	0	1	0	2	2	2	2	1	1	0	1	1	1	3	3
CVE-2017-1000080	1	17/07/2017	07/12/2020	1239	3	1	1	0	1	0	0	2	0	2	1	1	0	1	1	0	3	2
CVE-2017-1000079	1	17/07/2017	07/12/2020	1239	3	1	1	0	1	0	0	0	2	2	1	1	0	0	0	1	3	3

Anexo D

Este anexo apresenta o registo dos pacotes registados para os algoritmos RSA e ECDHE nas três vezes que foram analisadas (secção 5.2)

RSA teste número 1

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.5.1	192.168.5.145	TCP	66	54476 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.002612	192.168.5.145	192.168.5.1	TCP	66	8443 > 54476 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
3	0.002811	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
4	0.003406	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
5	0.004118	192.168.5.145	192.168.5.1	TCP	60	8443 > 54476 [ACK] Seq=1 Ack=518 Win=30336 Len=0
6	0.030584	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
7	0.031688	192.168.5.1	192.168.5.145	TLSv1.3	486	Change Cipher Spec, Client Hello
8	0.075701	192.168.5.145	192.168.5.1	TCP	60	8443 > 54476 [ACK] Seq=199 Ack=950 Win=31360 Len=0
9	0.109351	192.168.5.145	192.168.5.1	TLSv1.3	214	Server Hello
10	0.109526	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec

11	0.109660	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=950 Ack=365 Win=130816 Len=0							
12	0.110173	192.168.5.145	192.168.5.1	TLSv1.3	1332	Application Data							
13	0.116130	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data							
14	0.116567	192.168.5.1	192.168.5.145	TLSv1.3	765	Application Data							
15	0.118707	192.168.5.145	192.168.5.1	TCP	60	8443 > 54476 [ACK] Seq=1643 Ack=1024 Win=31360 Len=0							
16	0.120505	192.168.5.145	192.168.5.1	TCP	60	8443 > 54476 [ACK] Seq=1643 Ack=1735 Win=32768 Len=0							
17	0.125852	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data							
18	0.128754	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=1731 Ack=1735 Win=32768 Len=1460 [TCP segment of a reassembled PDU]							
19	0.128774	192.168.5.145	192.168.5.1	TLSv1.3	636	Application Data							
20	0.131272	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=1735 Ack=3773 Win=131328 Len=0							
21	0.205706	192.168.5.1	192.168.5.145	TLSv1.3	718	Application Data							
22	0.239791	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=3773 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]							
23	0.239816	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=5233 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]							
24	0.239831	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=6693 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]							
25	0.239902	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=8153 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]							

26	0.239920	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=9613 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
27	0.240084	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=11073 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
28	0.240137	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=12533 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
29	0.240161	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=13993 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
30	0.240183	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=15453 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
31	0.240202	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=16913 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
32	0.241848	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=2399 Ack=18373 Win=131328 Len=0
33	0.242199	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=18373 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
34	0.242219	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
35	0.242760	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=20178 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
36	0.242777	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=21638 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
37	0.242793	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=23098 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
38	0.242817	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=24558 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
39	0.242836	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=26018 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
40	0.242855	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=27478 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

41	0.242872	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=28938 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
42	0.242890	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=30398 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
43	0.242907	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=31858 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
44	0.242923	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=33318 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
45	0.243095	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=34778 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
46	0.243112	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
47	0.243717	192.168.5.145	192.168.5.1	TLSv1.3	258	Application Data
48	0.246760	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=2399 Ack=31858 Win=131328 Len=0
49	0.247509	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=36787 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
50	0.247556	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=38247 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
51	0.247572	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=39707 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
52	0.247642	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=41167 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
53	0.247660	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=42627 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
54	0.247675	192.168.5.145	192.168.5.1	TLSv1.3	1165	Application Data, Application Data
55	0.249240	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=2399 Ack=44087 Win=131328 Len=0

56	0.289589	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=2399 Ack=45198 Win=130048 Len=0							
57	0.464836	192.168.5.1	192.168.5.145	TLSv1.3	715	Application Data							
58	0.506284	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=45198 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]							
59	0.506301	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=46658 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]							
60	0.506311	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=48118 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]							
61	0.506321	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54476 [ACK] Seq=49578 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]							
62	0.506329	192.168.5.145	192.168.5.1	TLSv1.3	113	Application Data							
63	0.508049	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [ACK] Seq=3060 Ack=51097 Win=131328 Len=0							
64	1.134241	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [FIN, ACK] Seq=3060 Ack=51097 Win=131328 Len=0							
65	1.136332	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data							
66	1.136506	192.168.5.145	192.168.5.1	TCP	60	8443 > 54476 [FIN, ACK] Seq=51137 Ack=3061 Win=35712 Len=0							
67	1.137311	192.168.5.1	192.168.5.145	TCP	54	54476 > 8443 [RST, ACK] Seq=3061 Ack=51137 Win=0 Len=0							
68	92.244725	192.168.5.1	192.168.5.145	TCP	66	54564 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1							
69	92.245423	192.168.5.145	192.168.5.1	TCP	66	8443 > 54564 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128							
70	92.247248	192.168.5.1	192.168.5.145	TCP	54	54564 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0							

71	92.247969	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello							
72	92.249654	192.168.5.145	192.168.5.1	TCP	60	8443 > 54564 [ACK] Seq=1 Ack=518 Win=30336 Len=0							
73	92.284915	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request							
74	92.286392	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello							
75	92.301102	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello							
76	92.301499	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec							
77	92.302110	192.168.5.1	192.168.5.145	TCP	54	54564 > 8443 [ACK] Seq=1045 Ack=371 Win=130816 Len=0							
78	92.302288	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data							
79	92.303465	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data							
80	92.303892	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data							
81	92.304185	192.168.5.145	192.168.5.1	TCP	60	8443 > 54564 [ACK] Seq=489 Ack=1886 Win=32896 Len=0							
82	92.306791	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data							
83	92.308409	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
84	92.309157	192.168.5.1	192.168.5.145	TCP	54	54564 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0							
85	92.340311	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data							

86	92.355510	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
87	92.367192	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data							
88	92.386284	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
89	92.429082	192.168.5.1	192.168.5.145	TCP	54	54564 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0							
90	101.156464	192.168.5.1	192.168.5.145	TCP	54	54564 > 8443 [FIN, ACK] Seq=3271 Ack=877 Win=130304 Len=0							
91	101.157514	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data							
92	101.157698	192.168.5.145	192.168.5.1	TCP	60	8443 > 54564 [FIN, ACK] Seq=917 Ack=3272 Win=35968 Len=0							
93	101.158190	192.168.5.1	192.168.5.145	TCP	54	54564 > 8443 [RST, ACK] Seq=3272 Ack=917 Win=0 Len=0							
94	125.841013	192.168.5.1	192.168.5.145	TCP	66	54586 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1							
95	125.842666	192.168.5.145	192.168.5.1	TCP	66	8443 > 54586 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128							
96	125.842925	192.168.5.1	192.168.5.145	TCP	54	54586 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0							
97	125.843365	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello							
98	125.843567	192.168.5.145	192.168.5.1	TCP	60	8443 > 54586 [ACK] Seq=1 Ack=518 Win=30336 Len=0							
99	125.859144	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request							
100	125.861225	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello							

101	125.889212	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello							
102	125.889373	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec							
103	125.889722	192.168.5.1	192.168.5.145	TCP	54	54586 > 8443 [ACK] Seq=1045 Ack=371 Win=130816 Len=0							

RSA teste número 2

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.5.1	192.168.5.145	TCP	66	51208 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.004905	192.168.5.145	192.168.5.1	TCP	66	8443 > 51208 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
3	0.005162	192.168.5.1	192.168.5.145	TCP	54	51208 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
4	0.012021	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
5	0.018011	192.168.5.145	192.168.5.1	TCP	60	8443 > 51208 [ACK] Seq=1 Ack=518 Win=30336 Len=0
6	0.189728	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
7	0.190706	192.168.5.1	192.168.5.145	TLSv1.3	486	Change Cipher Spec, Client Hello
8	0.190966	192.168.5.145	192.168.5.1	TCP	60	8443 > 51208 [ACK] Seq=199 Ack=950 Win=31360 Len=0
9	0.293360	192.168.5.145	192.168.5.1	TLSv1.3	214	Server Hello
10	0.293499	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
11	0.293702	192.168.5.1	192.168.5.145	TCP	54	51208 > 8443 [ACK] Seq=950 Ack=365 Win=130816 Len=0
12	0.294832	192.168.5.145	192.168.5.1	TLSv1.3	1332	Application Data
13	0.305997	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data

14	0.306311	192.168.5.145	192.168.5.1	TCP	60	8443 > 51208 [ACK] Seq=1643 Ack=1024 Win=31360 Len=0							
15	0.306473	192.168.5.1	192.168.5.145	TLSv1.3	795	Application Data							
16	0.306848	192.168.5.145	192.168.5.1	TCP	60	8443 > 51208 [ACK] Seq=1643 Ack=1765 Win=32896 Len=0							
17	0.322143	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data							
18	0.361561	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
19	0.361833	192.168.5.1	192.168.5.145	TCP	54	51208 > 8443 [ACK] Seq=1765 Ack=1831 Win=131072 Len=0							
20	0.390652	192.168.5.1	192.168.5.145	TCP	54	51208 > 8443 [FIN, ACK] Seq=1765 Ack=1831 Win=131072 Len=0							
21	0.420997	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data							
22	0.421367	192.168.5.1	192.168.5.145	TCP	54	51208 > 8443 [RST, ACK] Seq=1766 Ack=1871 Win=0 Len=0							
23	0.421489	192.168.5.145	192.168.5.1	TCP	60	8443 > 51208 [FIN, ACK] Seq=1871 Ack=1766 Win=32896 Len=0							
24	0.476202	192.168.5.1	192.168.5.145	TCP	66	51210 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1							
25	0.477846	192.168.5.145	192.168.5.1	TCP	66	8443 > 51210 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128							
26	0.478452	192.168.5.1	192.168.5.145	TCP	54	51210 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0							
27	0.479954	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello							
28	0.482838	192.168.5.145	192.168.5.1	TCP	60	8443 > 51210 [ACK] Seq=1 Ack=518 Win=30336 Len=0							

29	0.490489	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request							
30	0.492861	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello							
31	0.533335	192.168.5.145	192.168.5.1	TCP	60	8443 > 51210 [ACK] Seq=199 Ack=1045 Win=31360 Len=0							
32	0.533532	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello							
33	0.533703	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec							
34	0.534314	192.168.5.1	192.168.5.145	TCP	54	51210 > 8443 [ACK] Seq=1045 Ack=371 Win=130816 Len=0							
35	0.534986	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data							
36	0.536714	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data							
37	0.537682	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data							
38	0.541563	192.168.5.145	192.168.5.1	TCP	60	8443 > 51210 [ACK] Seq=489 Ack=1119 Win=31360 Len=0							
39	0.542012	192.168.5.145	192.168.5.1	TCP	60	8443 > 51210 [ACK] Seq=489 Ack=1813 Win=32768 Len=0							
40	0.544794	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data							
41	0.567813	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
42	0.568446	192.168.5.1	192.168.5.145	TCP	54	51210 > 8443 [ACK] Seq=1813 Ack=677 Win=130560 Len=0							
43	0.598901	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data							

44	0.606666	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
45	0.648237	192.168.5.1	192.168.5.145	TCP	54	51210 > 8443 [ACK] Seq=2504 Ack=777 Win=130560 Len=0							
46	10.394722	192.168.5.1	192.168.5.145	TCP	54	51210 > 8443 [FIN, ACK] Seq=2504 Ack=777 Win=130560 Len=0							
47	10.396089	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data							
48	10.396340	192.168.5.1	192.168.5.145	TCP	54	51210 > 8443 [RST, ACK] Seq=2505 Ack=817 Win=0 Len=0							
49	10.396534	192.168.5.145	192.168.5.1	TCP	60	8443 > 51210 [FIN, ACK] Seq=817 Ack=2505 Win=34176 Len=0							
50	31.746196	192.168.5.1	192.168.5.145	TCP	66	51230 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1							
51	31.748913	192.168.5.145	192.168.5.1	TCP	66	8443 > 51230 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128							
52	31.749680	192.168.5.1	192.168.5.145	TCP	54	51230 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0							
53	31.750093	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello							
54	31.750323	192.168.5.145	192.168.5.1	TCP	60	8443 > 51230 [ACK] Seq=1 Ack=518 Win=30336 Len=0							
55	31.754935	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request							
56	31.755874	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello							
57	31.796985	192.168.5.145	192.168.5.1	TCP	60	8443 > 51230 [ACK] Seq=199 Ack=1045 Win=31360 Len=0							
58	31.803705	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello							

59	31.820241	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec							
60	31.820647	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data							
61	31.821915	192.168.5.1	192.168.5.145	TCP	54	51230 > 8443 [ACK] Seq=1045 Ack=489 Win=130816 Len=0							
62	31.822263	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data							
63	31.822494	192.168.5.145	192.168.5.1	TCP	60	8443 > 51230 [ACK] Seq=489 Ack=1119 Win=31360 Len=0							
64	31.822522	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data							
65	31.822756	192.168.5.145	192.168.5.1	TCP	60	8443 > 51230 [ACK] Seq=489 Ack=1886 Win=32896 Len=0							
66	31.831952	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data							
67	31.840549	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
68	31.843814	192.168.5.1	192.168.5.145	TCP	54	51230 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0							
69	31.882339	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data							
70	31.884661	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
71	31.900734	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data							
72	31.904773	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
73	31.946238	192.168.5.1	192.168.5.145	TCP	54	51230 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0							

74	40.420592	192.168.5.1	192.168.5.145	TCP	54	51230 > 8443 [FIN, ACK] Seq=3271 Ack=877 Win=130304 Len=0							
75	40.421723	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data							
76	40.421875	192.168.5.145	192.168.5.1	TCP	60	8443 > 51230 [FIN, ACK] Seq=917 Ack=3272 Win=35968 Len=0							
77	40.422165	192.168.5.1	192.168.5.145	TCP	54	51230 > 8443 [RST, ACK] Seq=3272 Ack=917 Win=0 Len=0							
78	62.700752	192.168.5.1	192.168.5.145	TCP	66	51244 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1							
79	62.701627	192.168.5.145	192.168.5.1	TCP	66	8443 > 51244 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128							
80	62.701838	192.168.5.1	192.168.5.145	TCP	54	51244 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0							
81	62.702249	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello							
82	62.702468	192.168.5.145	192.168.5.1	TCP	60	8443 > 51244 [ACK] Seq=1 Ack=518 Win=30336 Len=0							
83	62.706052	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request							
84	62.706656	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello							
85	62.737663	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello							
86	62.737835	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec							
87	62.738145	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data							
88	62.738972	192.168.5.1	192.168.5.145	TCP	54	51244 > 8443 [ACK] Seq=1045 Ack=489 Win=130816 Len=0							

89	62.739879	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data							
90	62.740133	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data							
91	62.740321	192.168.5.145	192.168.5.1	TCP	60	8443 > 51244 [ACK] Seq=489 Ack=1886 Win=32896 Len=0							
92	62.745794	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data							
93	62.747092	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
94	62.747709	192.168.5.1	192.168.5.145	TCP	54	51244 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0							
95	62.778105	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data							
96	62.780690	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
97	62.805894	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data							
98	62.830852	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data							
99	62.871829	192.168.5.1	192.168.5.145	TCP	54	51244 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0							
100	70.433450	192.168.5.1	192.168.5.145	TCP	54	51244 > 8443 [FIN, ACK] Seq=3271 Ack=877 Win=130304 Len=0							
101	70.441971	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data							
102	70.442133	192.168.5.145	192.168.5.1	TCP	60	8443 > 51244 [FIN, ACK] Seq=917 Ack=3272 Win=35968 Len=0							
103	70.442894	192.168.5.1	192.168.5.145	TCP	54	51244 > 8443 [RST, ACK] Seq=3272 Ack=917 Win=0 Len=0							

RSA teste número 3

	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.5.1	192.168.5.145	TCP	66	50857 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.001423	192.168.5.145	192.168.5.1	TCP	66	8443 > 50857 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
3	0.002507	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
4	0.002840	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
5	0.003029	192.168.5.145	192.168.5.1	TCP	60	8443 > 50857 [ACK] Seq=1 Ack=518 Win=30336 Len=0
6	0.023291	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
7	0.024747	192.168.5.1	192.168.5.145	TLSv1.3	486	Change Cipher Spec, Client Hello
8	0.067758	192.168.5.145	192.168.5.1	TCP	60	8443 > 50857 [ACK] Seq=199 Ack=950 Win=31360 Len=0
9	0.145169	192.168.5.145	192.168.5.1	TLSv1.3	214	Server Hello
10	0.145404	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
11	0.146176	192.168.5.145	192.168.5.1	TLSv1.3	1332	Application Data
12	0.147137	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=950 Ack=1643 Win=131328 Len=0
13	0.153614	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data

14	0.153986	192.168.5.1	192.168.5.145	TLSv1.3	765	Application Data
15	0.154237	192.168.5.145	192.168.5.1	TCP	60	8443 > 50857 [ACK] Seq=1643 Ack=1024 Win=31360 Len=0
16	0.154497	192.168.5.145	192.168.5.1	TCP	60	8443 > 50857 [ACK] Seq=1643 Ack=1735 Win=32768 Len=0
17	0.164673	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data
18	0.187652	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=1731 Ack=1735 Win=32768 Len=1460 [TCP segment of a reassembled PDU]
19	0.187672	192.168.5.145	192.168.5.1	TLSv1.3	636	Application Data
20	0.188602	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=1735 Ack=3773 Win=131328 Len=0
21	0.319365	192.168.5.1	192.168.5.145	TLSv1.3	718	Application Data
22	0.353468	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=3773 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
23	0.353492	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=5233 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
24	0.353506	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=6693 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
25	0.353755	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=8153 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
26	0.353778	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=9613 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
27	0.353797	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=11073 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
28	0.354030	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=12533 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

29	0.354052	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=13993 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
30	0.354124	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=15453 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
31	0.354624	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=2399 Ack=16913 Win=131328 Len=0
32	0.355386	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=16913 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
33	0.355409	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=18373 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
34	0.355422	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
35	0.358223	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=20178 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
36	0.358245	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=21638 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
37	0.358318	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=23098 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
38	0.358338	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=24558 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
39	0.358351	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=26018 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
40	0.358366	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=27478 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
41	0.358390	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=28938 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
42	0.358469	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=2399 Ack=23098 Win=131328 Len=0
43	0.359319	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=30398 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

44	0.359343	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=31858 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
45	0.359375	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=33318 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
46	0.359396	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=34778 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
47	0.359413	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
48	0.359909	192.168.5.145	192.168.5.1	TLSv1.3	258	Application Data
49	0.360995	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=36787 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
50	0.361019	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=38247 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
51	0.361096	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=39707 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
52	0.361114	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=41167 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
53	0.361128	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=42627 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
54	0.361145	192.168.5.145	192.168.5.1	TLSv1.3	1120	Application Data
55	0.361181	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=2399 Ack=36583 Win=131328 Len=0
56	0.361871	192.168.5.145	192.168.5.1	TLSv1.3	99	Application Data
57	0.362861	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=2399 Ack=45198 Win=131328 Len=0
58	0.440091	192.168.5.1	192.168.5.145	TLSv1.3	715	Application Data

59	0.443726	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=45198 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
60	0.443752	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=46658 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
61	0.443767	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=48118 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
62	0.443783	192.168.5.145	192.168.5.1	TCP	1514	8443 > 50857 [ACK] Seq=49578 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
63	0.443798	192.168.5.145	192.168.5.1	TLSv1.3	113	Application Data
64	0.444985	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [ACK] Seq=3060 Ack=51097 Win=131328 Len=0
65	8.777213	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [FIN, ACK] Seq=3060 Ack=51097 Win=131328 Len=0
66	8.778158	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data
67	8.778266	192.168.5.145	192.168.5.1	TCP	60	8443 > 50857 [FIN, ACK] Seq=51137 Ack=3061 Win=35712 Len=0
68	8.778643	192.168.5.1	192.168.5.145	TCP	54	50857 > 8443 [RST, ACK] Seq=3061 Ack=51137 Win=0 Len=0
69	19.525400	192.168.5.1	192.168.5.145	TCP	66	50871 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
70	19.527288	192.168.5.145	192.168.5.1	TCP	66	8443 > 50871 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
71	19.528686	192.168.5.1	192.168.5.145	TCP	54	50871 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
72	19.529594	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
73	19.530650	192.168.5.145	192.168.5.1	TCP	60	8443 > 50871 [ACK] Seq=1 Ack=518 Win=30336 Len=0

74	19.542471	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request									
75	19.544529	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello									
76	19.585235	192.168.5.145	192.168.5.1	TCP	60	8443 > 50871 [ACK] Seq=199 Ack=1045 Win=31360 Len=0									
77	19.592221	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello									
78	19.592357	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec									
79	19.592703	192.168.5.1	192.168.5.145	TCP	54	50871 > 8443 [ACK] Seq=1045 Ack=371 Win=130816 Len=0									
80	19.592979	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data									
81	19.594962	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data									
82	19.595302	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data									
83	19.595582	192.168.5.145	192.168.5.1	TCP	60	8443 > 50871 [ACK] Seq=489 Ack=1119 Win=31360 Len=0									
84	19.595692	192.168.5.145	192.168.5.1	TCP	60	8443 > 50871 [ACK] Seq=489 Ack=1886 Win=32896 Len=0									
85	19.605361	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data									
86	19.623792	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data									
87	19.624658	192.168.5.1	192.168.5.145	TCP	54	50871 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0									
88	19.657631	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data									

89	19.663155	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data									
90	19.678662	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data									
91	19.687746	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data									
92	19.730584	192.168.5.1	192.168.5.145	TCP	54	50871 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0									
93	38.451917	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data									
94	38.462928	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data									
95	38.492879	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data									
96	38.517205	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data									
97	38.540539	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data									
98	38.570103	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data									
99	38.612944	192.168.5.1	192.168.5.145	TCP	54	50871 > 8443 [ACK] Seq=5423 Ack=1177 Win=130048 Len=0									
100	58.789168	192.168.5.1	192.168.5.145	TCP	54	50871 > 8443 [FIN, ACK] Seq=5423 Ack=1177 Win=130048 Len=0									
101	58.790178	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data									
102	58.790328	192.168.5.145	192.168.5.1	TCP	60	8443 > 50871 [FIN, ACK] Seq=1217 Ack=5424 Win=40576 Len=0									
103	58.790863	192.168.5.1	192.168.5.145	TCP	54	50871 > 8443 [RST, ACK] Seq=5424 Ack=1217 Win=0 Len=0									

104	77.346282	192.168.5.1	192.168.5.145	TCP	66	49672 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
105	77.347649	192.168.5.145	192.168.5.1	TCP	66	8443 > 49672 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
106	77.353805	192.168.5.1	192.168.5.145	TCP	54	49672 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
107	77.354280	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
108	77.354488	192.168.5.145	192.168.5.1	TCP	60	8443 > 49672 [ACK] Seq=1 Ack=518 Win=30336 Len=0
109	77.359565	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
110	77.360388	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello
111	77.373399	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello
112	77.377973	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
113	77.378203	192.168.5.1	192.168.5.145	TCP	54	49672 > 8443 [ACK] Seq=1045 Ack=371 Win=130816 Len=0
114	77.382800	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data
115	77.384038	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data
116	77.384347	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data
117	77.397755	192.168.5.145	192.168.5.1	TCP	60	8443 > 49672 [ACK] Seq=489 Ack=1886 Win=32896 Len=0
118	77.400189	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data

119	77.401157	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data
120	77.408238	192.168.5.1	192.168.5.145	TCP	54	49672 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0
121	77.439328	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data
122	77.444325	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data
123	77.457211	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data
124	77.483975	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data
125	77.528319	192.168.5.1	192.168.5.145	TCP	54	49672 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0
126	83.794490	192.168.5.1	192.168.5.145	TCP	54	49672 > 8443 [FIN, ACK] Seq=3271 Ack=877 Win=130304 Len=0
127	83.797587	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data
128	83.797791	192.168.5.145	192.168.5.1	TCP	60	8443 > 49672 [FIN, ACK] Seq=917 Ack=3272 Win=35968 Len=0
129	83.798922	192.168.5.1	192.168.5.145	TCP	54	49672 > 8443 [RST, ACK] Seq=3272 Ack=917 Win=0 Len=0
130	102.990394	192.168.5.1	192.168.5.145	TCP	66	64923 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
131	102.991121	192.168.5.145	192.168.5.1	TCP	66	8443 > 64923 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
132	102.993197	192.168.5.1	192.168.5.145	TCP	54	64923 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0

ECDHE teste número 1

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.5.1	192.168.5.145	TCP	66	49828 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.004587	192.168.5.145	192.168.5.1	TCP	66	8443 > 49828 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
3	0.005712	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
4	0.020238	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
5	0.020566	192.168.5.145	192.168.5.1	TCP	60	8443 > 49828 [ACK] Seq=1 Ack=518 Win=30336 Len=0
6	0.097409	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
7	0.098317	192.168.5.1	192.168.5.145	TLSv1.3	486	Change Cipher Spec, Client Hello
8	0.098544	192.168.5.145	192.168.5.1	TCP	60	8443 > 49828 [ACK] Seq=199 Ack=950 Win=31360 Len=0
9	0.306937	192.168.5.145	192.168.5.1	TLSv1.3	214	Server Hello
10	0.307118	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
11	0.307348	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=950 Ack=365 Win=130816 Len=0
12	0.307830	192.168.5.145	192.168.5.1	TLSv1.3	756	Application Data
13	0.311695	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data

14	0.312492	192.168.5.1	192.168.5.145	TLSv1.3	765	Application Data								
15	0.314734	192.168.5.145	192.168.5.1	TCP	60	8443 > 49828 [ACK] Seq=1067 Ack=1024 Win=31360 Len=0								
16	0.315598	192.168.5.145	192.168.5.1	TCP	60	8443 > 49828 [ACK] Seq=1067 Ack=1735 Win=32768 Len=0								
17	0.321613	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data								
18	0.342952	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=1155 Ack=1735 Win=32768 Len=1460 [TCP segment of a reassembled PDU]								
19	0.342976	192.168.5.145	192.168.5.1	TLSv1.3	636	Application Data								
20	0.343718	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=1735 Ack=3197 Win=131328 Len=0								
21	0.444766	192.168.5.1	192.168.5.145	TLSv1.3	718	Application Data								
22	0.486777	192.168.5.145	192.168.5.1	TCP	60	8443 > 49828 [ACK] Seq=3197 Ack=2399 Win=34304 Len=0								
23	0.668195	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=3197 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]								
24	0.668220	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=4657 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]								
25	0.668236	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=6117 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]								
26	0.668383	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=7577 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]								
27	0.668410	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=9037 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]								
28	0.668428	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=10497 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]								

29	0.668685	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=11957 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
30	0.668711	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=13417 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
31	0.668729	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=14877 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
32	0.669506	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=2399 Ack=16337 Win=131328 Len=0
33	0.680759	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=16337 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
34	0.680793	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=17797 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
35	0.680807	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
36	0.681257	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=2399 Ack=19602 Win=131328 Len=0
37	0.755227	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=19602 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
38	0.755246	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=21062 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
39	0.755262	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=22522 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
40	0.755275	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=23982 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
41	0.755288	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=25442 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
42	0.755304	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=26902 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
43	0.755504	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=28362 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

44	0.755636	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=2399 Ack=29822 Win=131328 Len=0				
45	0.755684	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=29822 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
46	0.755779	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=31282 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
47	0.755796	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=32742 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
48	0.755809	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=34202 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
49	0.755822	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data				
50	0.757320	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=2399 Ack=36007 Win=131328 Len=0				
51	0.764308	192.168.5.145	192.168.5.1	TLSv1.3	258	Application Data				
52	0.781779	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=36211 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
53	0.781807	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=37671 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
54	0.781819	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=39131 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
55	0.781832	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=40591 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
56	0.781844	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=42051 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]				
57	0.781857	192.168.5.145	192.168.5.1	TLSv1.3	1120	Application Data				
58	0.782630	192.168.5.145	192.168.5.1	TLSv1.3	99	Application Data				

59	0.782938	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=2399 Ack=44622 Win=131328 Len=0								
60	0.791106	192.168.5.1	192.168.5.145	TLSv1.3	715	Application Data								
61	0.791491	192.168.5.145	192.168.5.1	TCP	60	8443 > 49828 [ACK] Seq=44622 Ack=3060 Win=35712 Len=0								
62	0.836529	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=44622 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]								
63	0.836552	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=46082 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]								
64	0.836563	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=47542 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]								
65	0.836573	192.168.5.145	192.168.5.1	TCP	1514	8443 > 49828 [ACK] Seq=49002 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]								
66	0.836609	192.168.5.145	192.168.5.1	TLSv1.3	113	Application Data								
67	0.837966	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [ACK] Seq=3060 Ack=50521 Win=131328 Len=0								
68	5.282355	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [FIN, ACK] Seq=3060 Ack=50521 Win=131328 Len=0								
69	5.284853	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data								
70	5.284991	192.168.5.145	192.168.5.1	TCP	60	8443 > 49828 [FIN, ACK] Seq=50561 Ack=3061 Win=35712 Len=0								
71	5.285242	192.168.5.1	192.168.5.145	TCP	54	49828 > 8443 [RST, ACK] Seq=3061 Ack=50561 Win=0 Len=0								
72	49.893520	192.168.5.1	192.168.5.145	TCP	66	49854 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1								
73	49.896543	192.168.5.145	192.168.5.1	TCP	66	8443 > 49854 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128								

74	49.896789	192.168.5.1	192.168.5.145	TCP	54	49854 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0							
75	49.897305	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello							
76	49.897564	192.168.5.145	192.168.5.1	TCP	60	8443 > 49854 [ACK] Seq=1 Ack=518 Win=30336 Len=0							
77	50.130583	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request							
78	50.131805	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello							
79	50.132021	192.168.5.145	192.168.5.1	TCP	60	8443 > 49854 [ACK] Seq=199 Ack=1045 Win=31360 Len=0							
80	50.203519	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello							
81	50.211396	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec							
82	50.212080	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data							
83	50.212955	192.168.5.1	192.168.5.145	TCP	54	49854 > 8443 [ACK] Seq=1045 Ack=489 Win=130816 Len=0							
84	50.213168	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data							
85	50.213311	192.168.5.145	192.168.5.1	TCP	60	8443 > 49854 [ACK] Seq=489 Ack=1119 Win=31360 Len=0							
86	50.213419	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data							
87	50.213560	192.168.5.145	192.168.5.1	TCP	60	8443 > 49854 [ACK] Seq=489 Ack=1886 Win=32896 Len=0							
88	50.221474	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data							

89	50.237403	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data								
90	50.238384	192.168.5.1	192.168.5.145	TCP	54	49854 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0								
91	50.265485	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data								
92	50.290625	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data								
93	50.299746	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data								
94	50.321972	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data								
95	50.364885	192.168.5.1	192.168.5.145	TCP	54	49854 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0								
96	55.290690	192.168.5.1	192.168.5.145	TCP	54	49854 > 8443 [FIN, ACK] Seq=3271 Ack=877 Win=130304 Len=0								
97	55.293107	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data								
98	55.293419	192.168.5.145	192.168.5.1	TCP	60	8443 > 49854 [FIN, ACK] Seq=917 Ack=3272 Win=35968 Len=0								
99	55.294560	192.168.5.1	192.168.5.145	TCP	54	49854 > 8443 [RST, ACK] Seq=3272 Ack=917 Win=0 Len=0								
100	81.320264	192.168.5.1	192.168.5.145	TCP	66	49873 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1								
101	81.321642	192.168.5.145	192.168.5.1	TCP	66	8443 > 49873 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128								
102	81.323399	192.168.5.1	192.168.5.145	TCP	54	49873 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0								
103	81.323964	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello								

104	81.324400	192.168.5.145	192.168.5.1	TCP	60	8443 > 49873 [ACK] Seq=1 Ack=518 Win=30336 Len=0								
105	81.355018	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request								
106	81.355727	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello								
107	81.395590	192.168.5.145	192.168.5.1	TCP	60	8443 > 49873 [ACK] Seq=199 Ack=1045 Win=31360 Len=0								
108	81.421135	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello								
109	81.421308	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec								
110	81.421641	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data								
111	81.423293	192.168.5.1	192.168.5.145	TCP	54	49873 > 8443 [ACK] Seq=1045 Ack=489 Win=130816 Len=0								

ECDHE teste número 2

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.5.1	192.168.5.145	TCP	66	54484 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.001572	192.168.5.145	192.168.5.1	TCP	66	8443 > 54484 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
3	0.002109	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
4	0.013747	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
5	0.014134	192.168.5.145	192.168.5.1	TCP	60	8443 > 54484 [ACK] Seq=1 Ack=518 Win=30336 Len=0
6	0.113544	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
7	0.115085	192.168.5.1	192.168.5.145	TLSv1.3	486	Change Cipher Spec, Client Hello
8	0.115427	192.168.5.145	192.168.5.1	TCP	60	8443 > 54484 [ACK] Seq=199 Ack=950 Win=31360 Len=0
9	0.294514	192.168.5.145	192.168.5.1	TLSv1.3	214	Server Hello
10	0.294705	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
11	0.295560	192.168.5.145	192.168.5.1	TLSv1.3	756	Application Data
12	0.302890	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=950 Ack=1067 Win=130304 Len=0
13	0.306688	192.168.5.145	192.168.5.1	TLSv1.3	756	[TCP Spurious Retransmission] , Application Data

14	0.306860	192.168.5.1	192.168.5.145	TCP	66	[TCP Dup ACK 12#1] 54484 > 8443 [ACK] Seq=950 Ack=1067 Win=130304 Len=0 SLE=365 SRE=1067
15	0.311362	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data
16	0.311647	192.168.5.145	192.168.5.1	TCP	60	8443 > 54484 [ACK] Seq=1067 Ack=1024 Win=31360 Len=0
17	0.311816	192.168.5.1	192.168.5.145	TLSv1.3	765	Application Data
18	0.312518	192.168.5.145	192.168.5.1	TCP	60	8443 > 54484 [ACK] Seq=1067 Ack=1735 Win=32768 Len=0
19	0.321305	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data
20	0.330472	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=1155 Ack=1735 Win=32768 Len=1460 [TCP segment of a reassembled PDU]
21	0.330497	192.168.5.145	192.168.5.1	TLSv1.3	636	Application Data
22	0.331641	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=1735 Ack=3197 Win=131328 Len=0
23	0.417943	192.168.5.1	192.168.5.145	TLSv1.3	718	Application Data
24	0.462869	192.168.5.145	192.168.5.1	TCP	60	8443 > 54484 [ACK] Seq=3197 Ack=2399 Win=34304 Len=0
25	0.763416	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=3197 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
26	0.763445	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=4657 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
27	0.763458	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=6117 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
28	0.763473	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=7577 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

29	0.763487	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=9037 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
30	0.763660	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=10497 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
31	0.763675	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=11957 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
32	0.763689	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=13417 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
33	0.763712	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=14877 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
34	0.763732	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=16337 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
35	0.764295	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=2399 Ack=17797 Win=131328 Len=0
36	0.764648	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=17797 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
37	0.764669	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
38	0.765134	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=2399 Ack=19602 Win=131328 Len=0
39	0.771496	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=19602 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
40	0.771518	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=21062 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
41	0.771532	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=22522 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
42	0.771544	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=23982 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
43	0.771562	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=25442 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

44	0.771640	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=26902 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
45	0.771655	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=28362 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
46	0.771677	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=29822 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
47	0.771694	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=31282 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
48	0.771712	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=32742 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
49	0.771891	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=34202 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
50	0.771908	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
51	0.773028	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=2399 Ack=32742 Win=131328 Len=0
52	0.774315	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=2399 Ack=36007 Win=131328 Len=0
53	0.782938	192.168.5.145	192.168.5.1	TLSv1.3	258	Application Data
54	0.824430	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=2399 Ack=36211 Win=131072 Len=0
55	0.828670	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=36211 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
56	0.828692	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=37671 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
57	0.828706	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=39131 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
58	0.828722	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=40591 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

59	0.828735	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=42051 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
60	0.828898	192.168.5.145	192.168.5.1	TLSv1.3	1120	Application Data
61	0.829127	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=2399 Ack=44577 Win=131328 Len=0
62	0.830816	192.168.5.145	192.168.5.1	TLSv1.3	99	Application Data
63	0.838121	192.168.5.1	192.168.5.145	TLSv1.3	715	Application Data
64	0.841902	192.168.5.145	192.168.5.1	TCP	60	8443 > 54484 [ACK] Seq=44622 Ack=3060 Win=35712 Len=0
65	0.864195	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=44622 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
66	0.864225	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=46082 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
67	0.864240	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=47542 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
68	0.864255	192.168.5.145	192.168.5.1	TCP	1514	8443 > 54484 [ACK] Seq=49002 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
69	0.864270	192.168.5.145	192.168.5.1	TLSv1.3	113	Application Data
70	0.865133	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [ACK] Seq=3060 Ack=50521 Win=131328 Len=0
71	5.735299	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [FIN, ACK] Seq=3060 Ack=50521 Win=131328 Len=0
72	5.737014	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data
73	5.737255	192.168.5.145	192.168.5.1	TCP	60	8443 > 54484 [FIN, ACK] Seq=50561 Ack=3061 Win=35712 Len=0

74	5.737613	192.168.5.1	192.168.5.145	TCP	54	54484 > 8443 [RST, ACK] Seq=3061 Ack=50561 Win=0 Len=0
75	24.663939	192.168.5.1	192.168.5.145	TCP	66	54501 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
76	24.665457	192.168.5.145	192.168.5.1	TCP	66	8443 > 54501 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
77	24.665832	192.168.5.1	192.168.5.145	TCP	54	54501 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
78	24.666742	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
79	24.667077	192.168.5.145	192.168.5.1	TCP	60	8443 > 54501 [ACK] Seq=1 Ack=518 Win=30336 Len=0
80	24.715977	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
81	24.717146	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello
82	24.717438	192.168.5.145	192.168.5.1	TCP	60	8443 > 54501 [ACK] Seq=199 Ack=1045 Win=31360 Len=0
83	24.773585	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello
84	24.773816	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
85	24.774244	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data
86	24.774698	192.168.5.1	192.168.5.145	TCP	54	54501 > 8443 [ACK] Seq=1045 Ack=489 Win=130816 Len=0
87	24.775259	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data
88	24.775492	192.168.5.145	192.168.5.1	TCP	60	8443 > 54501 [ACK] Seq=489 Ack=1119 Win=31360 Len=0

89	24.775526	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data								
90	24.775691	192.168.5.145	192.168.5.1	TCP	60	8443 > 54501 [ACK] Seq=489 Ack=1886 Win=32896 Len=0								
91	24.784763	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data								
92	24.812633	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data								
93	24.818418	192.168.5.1	192.168.5.145	TCP	54	54501 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0								
94	24.850471	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data								
95	24.890964	192.168.5.145	192.168.5.1	TCP	60	8443 > 54501 [ACK] Seq=677 Ack=2580 Win=34432 Len=0								
96	24.911934	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data								
97	24.943888	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data								
98	24.944265	192.168.5.145	192.168.5.1	TCP	60	8443 > 54501 [ACK] Seq=777 Ack=3271 Win=35968 Len=0								
99	24.963386	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data								
100	25.004693	192.168.5.1	192.168.5.145	TCP	54	54501 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0								
101	25.741397	192.168.5.1	192.168.5.145	TCP	54	54501 > 8443 [FIN, ACK] Seq=3271 Ack=877 Win=130304 Len=0								
102	25.743194	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data								
103	25.743323	192.168.5.145	192.168.5.1	TCP	60	8443 > 54501 [FIN, ACK] Seq=917 Ack=3272 Win=35968 Len=0								

104	25.743905	192.168.5.1	192.168.5.145	TCP	54	54501 > 8443 [RST, ACK] Seq=3272 Ack=917 Win=0 Len=0
105	48.157326	192.168.5.1	192.168.5.145	TCP	66	54515 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
106	48.158347	192.168.5.145	192.168.5.1	TCP	66	8443 > 54515 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
107	48.158825	192.168.5.1	192.168.5.145	TCP	54	54515 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
108	48.159266	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
109	48.159774	192.168.5.145	192.168.5.1	TCP	60	8443 > 54515 [ACK] Seq=1 Ack=518 Win=30336 Len=0
110	48.176425	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
111	48.179667	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello
112	48.219958	192.168.5.145	192.168.5.1	TCP	60	8443 > 54515 [ACK] Seq=199 Ack=1045 Win=31360 Len=0
113	48.237281	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello
114	48.238027	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
115	48.238872	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data
116	48.239582	192.168.5.1	192.168.5.145	TCP	54	54515 > 8443 [ACK] Seq=1045 Ack=489 Win=130816 Len=0
117	48.240538	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data

ECDHE teste número 3

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.5.1	192.168.5.145	TCP	66	60238 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.000661	192.168.5.145	192.168.5.1	TCP	66	8443 > 60238 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
3	0.001836	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
4	0.002213	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
5	0.006310	192.168.5.145	192.168.5.1	TCP	60	8443 > 60238 [ACK] Seq=1 Ack=518 Win=30336 Len=0
6	0.060455	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
7	0.064366	192.168.5.1	192.168.5.145	TLSv1.3	486	Change Cipher Spec, Client Hello
8	0.064714	192.168.5.145	192.168.5.1	TCP	60	8443 > 60238 [ACK] Seq=199 Ack=950 Win=31360 Len=0
9	0.172125	192.168.5.145	192.168.5.1	TLSv1.3	214	Server Hello
10	0.177727	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
11	0.178051	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=950 Ack=365 Win=130816 Len=0
12	0.178572	192.168.5.145	192.168.5.1	TLSv1.3	755	Application Data
13	0.182257	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data

14	0.182485	192.168.5.145	192.168.5.1	TCP	60	8443 > 60238 [ACK] Seq=1066 Ack=1024 Win=31360 Len=0
15	0.182549	192.168.5.1	192.168.5.145	TLSv1.3	765	Application Data
16	0.182709	192.168.5.145	192.168.5.1	TCP	60	8443 > 60238 [ACK] Seq=1066 Ack=1735 Win=32768 Len=0
17	0.196516	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data
18	0.211691	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=1154 Ack=1735 Win=32768 Len=1460 [TCP segment of a reassembled PDU]
19	0.211711	192.168.5.145	192.168.5.1	TLSv1.3	636	Application Data
20	0.217097	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=1735 Ack=3196 Win=131328 Len=0
21	0.317084	192.168.5.1	192.168.5.145	TLSv1.3	718	Application Data
22	0.358876	192.168.5.145	192.168.5.1	TCP	60	8443 > 60238 [ACK] Seq=3196 Ack=2399 Win=34304 Len=0
23	0.409223	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=3196 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
24	0.409250	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=4656 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
25	0.409432	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=6116 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
26	0.409451	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=7576 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
27	0.410049	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=2399 Ack=9036 Win=131328 Len=0
28	0.410135	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=9036 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

29	0.410162	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=10496 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
30	0.410402	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=11956 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
31	0.410421	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=13416 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
32	0.411028	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=2399 Ack=14876 Win=131328 Len=0
33	0.412236	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=14876 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
34	0.412261	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=16336 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
35	0.412377	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=17796 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
36	0.412390	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
37	0.413063	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=2399 Ack=19601 Win=131328 Len=0
38	0.427521	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=19601 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
39	0.429248	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=21061 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
40	0.429275	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=22521 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
41	0.429301	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=23981 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
42	0.429323	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=25441 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
43	0.429340	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=26901 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]

44	0.430072	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=2399 Ack=28361 Win=131328 Len=0
45	0.430260	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=28361 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
46	0.430276	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=29821 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
47	0.430290	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=31281 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
48	0.430304	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=32741 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
49	0.430325	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=34201 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
50	0.430343	192.168.5.145	192.168.5.1	TLSv1.3	399	Application Data
51	0.430790	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=2399 Ack=36006 Win=131328 Len=0
52	0.439123	192.168.5.145	192.168.5.1	TLSv1.3	258	Application Data
53	0.455810	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=36210 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
54	0.455835	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=37670 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
55	0.455853	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=39130 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
56	0.455868	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=40590 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
57	0.455882	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=42050 Ack=2399 Win=34304 Len=1460 [TCP segment of a reassembled PDU]
58	0.455896	192.168.5.145	192.168.5.1	TLSv1.3	1120	Application Data

59	0.457136	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=2399 Ack=44576 Win=131328 Len=0
60	0.471098	192.168.5.145	192.168.5.1	TLSv1.3	99	Application Data
61	0.471933	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=2399 Ack=44621 Win=131328 Len=0
62	0.644640	192.168.5.1	192.168.5.145	TLSv1.3	715	Application Data
63	0.646166	192.168.5.145	192.168.5.1	TCP	60	8443 > 60238 [ACK] Seq=44621 Ack=3060 Win=35712 Len=0
64	0.696815	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=44621 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
65	0.696848	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=46081 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
66	0.696866	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=47541 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
67	0.696932	192.168.5.145	192.168.5.1	TCP	1514	8443 > 60238 [ACK] Seq=49001 Ack=3060 Win=35712 Len=1460 [TCP segment of a reassembled PDU]
68	0.696948	192.168.5.145	192.168.5.1	TLSv1.3	113	Application Data
69	0.697866	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [ACK] Seq=3060 Ack=50520 Win=131328 Len=0
70	6.638215	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [FIN, ACK] Seq=3060 Ack=50520 Win=131328 Len=0
71	6.640931	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data
72	6.641136	192.168.5.145	192.168.5.1	TCP	60	8443 > 60238 [FIN, ACK] Seq=50560 Ack=3061 Win=35712 Len=0
73	6.641330	192.168.5.1	192.168.5.145	TCP	54	60238 > 8443 [RST, ACK] Seq=3061 Ack=50560 Win=0 Len=0

74	21.486627	192.168.5.1	192.168.5.145	TCP	66	60252 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
75	21.490766	192.168.5.145	192.168.5.1	TCP	66	8443 > 60252 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
76	21.491429	192.168.5.1	192.168.5.145	TCP	54	60252 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
77	21.492003	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
78	21.492439	192.168.5.145	192.168.5.1	TCP	60	8443 > 60252 [ACK] Seq=1 Ack=518 Win=30336 Len=0
79	21.502551	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
80	21.503382	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello
81	21.543711	192.168.5.145	192.168.5.1	TCP	60	8443 > 60252 [ACK] Seq=199 Ack=1045 Win=31360 Len=0
82	21.549267	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello
83	21.549489	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
84	21.554476	192.168.5.1	192.168.5.145	TCP	54	60252 > 8443 [ACK] Seq=1045 Ack=371 Win=130816 Len=0
85	21.559263	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data
86	21.560638	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data
87	21.560926	192.168.5.145	192.168.5.1	TCP	60	8443 > 60252 [ACK] Seq=489 Ack=1119 Win=31360 Len=0
88	21.560990	192.168.5.1	192.168.5.145	TLSv1.3	821	Application Data

89	21.561149	192.168.5.145	192.168.5.1	TCP	60	8443 > 60252 [ACK] Seq=489 Ack=1886 Win=32896 Len=0					
90	21.569840	192.168.5.145	192.168.5.1	TLSv1.3	142	Application Data					
91	21.571776	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data					
92	21.571959	192.168.5.1	192.168.5.145	TCP	54	60252 > 8443 [ACK] Seq=1886 Ack=677 Win=130560 Len=0					
93	21.598288	192.168.5.1	192.168.5.145	TLSv1.3	748	Application Data					
94	21.646010	192.168.5.145	192.168.5.1	TCP	60	8443 > 60252 [ACK] Seq=677 Ack=2580 Win=34432 Len=0					
95	21.662168	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data					
96	21.670074	192.168.5.1	192.168.5.145	TLSv1.3	745	Application Data					
97	21.675937	192.168.5.145	192.168.5.1	TCP	60	8443 > 60252 [ACK] Seq=777 Ack=3271 Win=35968 Len=0					
98	21.697486	192.168.5.145	192.168.5.1	TLSv1.3	154	Application Data					
99	21.741125	192.168.5.1	192.168.5.145	TCP	54	60252 > 8443 [ACK] Seq=3271 Ack=877 Win=130304 Len=0					
100	26.650717	192.168.5.1	192.168.5.145	TCP	54	60252 > 8443 [FIN, ACK] Seq=3271 Ack=877 Win=130304 Len=0					
101	26.651808	192.168.5.145	192.168.5.1	TLSv1.3	94	Application Data					
102	26.651964	192.168.5.145	192.168.5.1	TCP	60	8443 > 60252 [FIN, ACK] Seq=917 Ack=3272 Win=35968 Len=0					
103	26.652241	192.168.5.1	192.168.5.145	TCP	54	60252 > 8443 [RST, ACK] Seq=3272 Ack=917 Win=0 Len=0					

104	42.498983	192.168.5.1	192.168.5.145	TCP	66	60267 > 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
105	42.499939	192.168.5.145	192.168.5.1	TCP	66	8443 > 60267 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
106	42.500136	192.168.5.1	192.168.5.145	TCP	54	60267 > 8443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
107	42.500530	192.168.5.1	192.168.5.145	TLSv1.3	571	Client Hello
108	42.500811	192.168.5.145	192.168.5.1	TCP	60	8443 > 60267 [ACK] Seq=1 Ack=518 Win=30336 Len=0
109	42.520140	192.168.5.145	192.168.5.1	TLSv1.3	252	Hello Retry Request
110	42.522061	192.168.5.1	192.168.5.145	TLSv1.3	581	Change Cipher Spec, Client Hello
111	42.561634	192.168.5.145	192.168.5.1	TCP	60	8443 > 60267 [ACK] Seq=199 Ack=1045 Win=31360 Len=0
112	42.564950	192.168.5.145	192.168.5.1	TLSv1.3	220	Server Hello
113	42.577290	192.168.5.145	192.168.5.1	TLSv1.3	60	Change Cipher Spec
114	42.577804	192.168.5.1	192.168.5.145	TCP	54	60267 > 8443 [ACK] Seq=1045 Ack=371 Win=130816 Len=0
115	42.578465	192.168.5.145	192.168.5.1	TLSv1.3	172	Application Data
116	42.580131	192.168.5.1	192.168.5.145	TLSv1.3	128	Application Data